Des Données aux Connaissances : Modèles et Algorithmes

Saïd Jabbour

UNIVERSITÉ D'ARTOIS

UNIVERSITÉ D'ARTOIS

ECOLE DOCTORALE SPI 072
SCIENCES POUR L'INGÉNIEUR

Mémoire

pour l'obtention du titre de

Habilitation à Diriger les Recherches

de l'Université d'Artois
Mention : INFORMATIQUE

Présentée par

Said JABBOUR

Des Données aux Connaissances : Modèles et Algorithmes

Soutenue publiquement le
6 décembre 2018

Composition du jury :

Rapporteurs :	Sihem AMER-YAHYA (Directrice de Recherche)	LIG - CNRS, Université Grenoble Alpes
	Philippe-Viger FOURINER (Professeur)	Harbin Institute of Technology (Chine)
	Felip MANYA (Professeur)	IIIA-CSIC (Espagne)
Examinateurs :	Anthony HUNTER (Professeur)	University College London (UK)
	Jean-Charles RÉGIN (Professeur des Universités)	Université de Nice-Sophia Antipolis
	Philippe BESNARD (Directeur de Recherche)	IRIT - CNRS, Université Paul Sabatier

CENTRE DE RECHERCHE EN INFORMATIQUE DE LENS (CRIL CNRS UMR 8188)
Université d'Artois, rue Jean Souvraz, S.P. 18 F-62307, Lens Cedex France
Secrétariat : Tél. : +33 (0)3 21 79 17 23 – Fax : +33 (0)3 21 79 17 70
http://www.cril.univ-artois.fr

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction générale

Mes thématiques de recherche s'inscrivent à la frontière de plusieurs domaines différents mais de premier plan incluant l'Intelligence Artificielle (IA), la fouille de données et les systèmes d'information. Elles peuvent être caractérisées par les mots clés suivants : programmation par contraintes, représentation des connaissances, la problématique de raisonnement sur des ontologies conflictuelles (incohérentes et/ou inconsistantes) via l'argumentation déductive, fouille de motifs sous contraintes, clustering, détection de communautés, compression de données et composition de services web. Cette multidisciplinarité a plusieurs motivations et sources d'explications. Tout d'abord ma volonté à explorer de nouvelles thématiques afin de mettre en lumière leurs connexions, leurs complémentarités et leurs spécificités en vue d'une réelle fertilisation croisée. De plus, bien que les formalismes ou les modèles de représentation utilisés ne sont pas les mêmes, certaines interrogations et problématiques étudiées dans ces domaines sont similaires et récurrentes. On peut citer par exemple la problématique de la qualité des données, l'algorithmique d'énumération de structures combinatoire, les paradigmes utilisés pour l'intégration ou la fusion de données/connaissances et la détection d'incohérences. Une autre motivation vient de la place de plus en plus grande qu'occupent les recherches autour de l'analyse et de la valorisation des données via des techniques issues de la fouille de données et de l'apprentissage automatique. Ces dernières ont contribué à élargir considérablement l'éventail des applications qui touche à de nouveaux secteurs dans le monde industriels et des services. Au-delà de ces motivations indéniables, mon ouverture à d'autres thématiques de recherche s'est concrétisée au gré des rencontres et des projets de recherche multidisciplinaires auxquels j'ai participé.

Dans un ordre chronologique, mes premiers travaux ont porté sur les problèmes de décision ou d'optimisation associés à l'inférence et à la prise de décision en intelligence artificielle (axe II du CRIL). Ces problèmes sont typiquement intraitables, i.e. il n'existe usuellement pas d'algorithme déterministe en temps polynomial dans le pire des cas pour ceux-ci (et la théorie de la complexité pousse à conjecturer fortement qu'il n'en existera jamais). Cette difficulté calculatoire impose le développement de solutions palliatives ayant pour objectif d'élargir la classe des problèmes traitables en pratique, en permettant la résolution d'instances de tailles de plus en plus grande, dans un laps de temps raisonnable. Dans ce contexte, je me suis particulièrement intéressé à deux problèmes représentatifs de cette difficulté calculatoire, il s'agit du problème de la satisfiabilité d'une formule propositionnelle (SAT) et de son extension les formules booléennes quantifiées (QBF), deux exemples de références de la classe des problèmes NP et PSPACE complet respectivement. Mes travaux de recherche autour de ces deux problèmes ont portés principalement sur les aspects algorithmiques incluant l'apprentissage de clauses, les techniques de simplifications, les stratégies de résolution et la détection et l'exploitation de propriétés structurelles des problèmes traités (e.g. symétries, dépendances fonctionnelles). Nos contributions ont donné lieu à divers solveurs parmi les plus performants, attesté par les diverses médailles que nous avions obtenue lors des compéti-

tions internationales. Nous avons donc eu la chance de participer à cette effervescence particulière qu'a connu la résolution du problème SAT ces dernières décennies, dont les résultats ont permis un réel passage à l'échelle. Les solveurs SAT modernes sont aujourd'hui capables de résoudre efficacement des instances de SAT issues de problèmes réels avec plusieurs millions de variables et de clauses. Ce saut qualitatif a contribué à élargir les domaines d'applications traditionnels de SAT (vérification formelle de logiciels et de matériels) à d'autres champs d'applications comme la planification, la bio-informatique et la cryptographie. La modélisation d'applications du monde réel en logique propositionnelle et l'extension des résultats de SAT à des problématiques autour de SAT est devenue aujourd'hui une pratique courante. Cette nouvelle situation suscite un réel débat sur le fait que la complexité au pire cas généralement privilégié par la théorie de la complexité est loin de refléter la réalité de la résolution de problèmes en pratique (voir article de Moshe Y. Vardi "On P, NP, and computational complexity" ACM 2010).

Depuis ma thèse, plusieurs rencontres et faits marquants ont contribué à concrétiser mon projet d'ouverture à d'autres thématiques de recherche. Notre expertise reconnue sur la résolution du problème de la satisfiabilité propositionnelle a fortement intéressé l'équipe "Constraint Reasoning" de Microsfot Research Cambridge (UK) dirigée par Youssef Hamadi. Dans le cadre d'un stage à Cambridge et d'un postdoctorat au laboratoire commun Inria-Microsoft à Paris, je me suis intéressé au développement d'approches parallèles pour SAT exploitant les architectures multi-cœurs. Dans ce cadre, nous avons proposé la première approche parallèle de type portfolio avec de nombreuses améliorations ayant permis de dominer la compétition SAT (parallel track) de 2008 à 2011. Cette approche a ensuite été intégré dans le solveur SMT (Sat Modulo Theory) Z3 développé par Leonardo de Moura à Microsoft, un solveur état-de-l'art, très utilisé dans la vérification de logiciels. Ce succès a contribué grandement au regain d'intérêt pour le développement d'approches parallèles pour SAT et CP.

Le CRIL est un laboratoire structuré autour de deux axes complémentaires, la représentation des connaissances et raisonnements (Axe 1), et Algorithmes pour l'inférence et contraintes (Axe 2). Cette proximité entre les deux axes a été propice à mon intérêt pour la thématique de la représentation des connaissances. Nous avons proposé plusieurs contributions à deux des problématiques de cet axe : la théorie de l'argumentation et les mesures d'incohérentes. Pour l'argumentation, nous avons entre autres introduit des relations de préférences, exprimés en fonction des relations d'attaque et de défense entre arguments, pour le calcul des Top-k extensions non conflictuelles. Pour quantifier l'incohérence des bases de connaissances, nous avons proposés de nouvelles mesures et de nouvelles propriétés que toute mesure d'incohérence est censée vérifier. Nos mesures prennent mieux en compte la structure de l'hypergraphe induit par les sous-formules minimalement incohérences. Elles nous ont permis d'exhiber de nouveaux problèmes comme celui du set packing fermé ou l'extension du problème "Symmetric Intersecting Monotone UNSAT".

Un autre fait marquant ayant influencé mes travaux récents est sans aucun doute le projet ANR Défi DAG [1] "Approches déclaratives pour l'énumération de motifs intéressants à partir de données" 2009-2013. Ce projet auquel j'ai eu la chance de participer a un double objectif, celui de (1) définir des langages déclaratifs de haut niveau (logiques ou algébriques) pour exprimer et représenter les problèmes de fouille de données et (2) exhiber des sous-classes de problèmes pour lesquelles il existe des algorithmes génériques et efficaces. Les nombreux résultats que nous

1. https ://projet.liris.cnrs.fr/dag/

avions obtenus dans le cadre de ce projet multidisciplinaire regroupant le CRIL (Programmation par contraintes/logique), le LIRIS (Bases de données/Fouille de données) et le LIMOS (Algorithmique d'énumération/théorie des graphes), ont été à l'origine de la création d'un groupe de recherche au CRIL autour de la fouille de données et contraintes. Dans ce cadre et par fertilisation croisée entre la fouille de données et l'intelligence artificielle et plus particulièrement CP/SAT, nous avons proposé diverses formulations des problèmes de fouille de données en SAT, incluant la fouille d'itemsets fréquents, maximaux ou fermés, l'énumération de règles d'association et de ses nombreuses variantes et la fouille de séquences. Les apports de cette fertilisation croisée sont particulièrement illustrés par deux de nos contributions, à savoir (1) l'extension des symétries (largement étudiées en SAT/CP) en fouille de données, (2) l'exploitation de la fouille de données pour compresser des formules booléennes ou des contraintes relationnelles (ou en extension) et (3) la mise en œuvre de nouvelles techniques de clustering d'objets complexes représentés par des formules booléennes. Les effets de bord de cette ouverture thématique sont nombreux et enrichissants. Le premier exemple vient du fait que la modélisation de problèmes de fouilles de données en logique propositionnelle fait souvent intervenir des contraintes linéaires 0/1 (contraintes de cardinalité et contraintes pseudo booléennes) dont le codage SAT a fait l'objet de nombreux travaux. Dans ce cadre, nous avons proposé une formulation de ce type de contraintes en utilisant le fameux principe des tiroirs (ou Pigeon-Hole Problem). Le second vient de notre étude sur la compression de formules booléennes en particulier de clauses binaires. Ces dernières admettent une représentation sous forme de graphes. Nous avons observé que les cliques ou bi-cliques de clauses binaires représentent des inéquations linéaires 0/1 particulières. Cette observation est à l'origine de la proposition de nouvelles classes de graphes à l'origine de notre approche de compression de graphes par une représentation sous forme d'une disjonction d'inéquations linéaires 0/1 dont les solutions donnent les arêtes du graphe originale. Notre approche de compression par l'exploitation de certaines classes de graphes admet des connexions avec une autre thématique de recherche importante : la détection de communautés. Nous avons proposé diverses approches déclaratives et spécialisés en exploitant diverses classes de graphes.

La description (non exhaustive) chronologique ci-dessus est donnée pour illustrer mes ouvertures thématiques qui vont de SAT aux graphes et à la détection de communautés en passant par la représentation des connaissances et la fouille de données. Cette ouverture thématique se poursuivra avec le projet QDoSSI « Qualité des données multi-sources : un double défi pour les sciences sociales et les sciences de l'informatique » financé par la mission pour l'interdisciplinarité du CNRS depuis 2016. L'objet étant d'analyser les données (structurés et non structurés) traitant des migrations internationales afin de comprendre les processus à l'œuvre dans la construction desD parcours migratoires. Ce projet offre donc une excellente opportunité pour contribuer à d'autres thématiques comme la qualité des données, la fouille de texte, la fouille de graphes attribués et la visualisation de données.

Une synthèse détaillée de mes travaux et des perspectives sera donnée dans la suite de ce mémoire d'habilitation à diriger les recherches qui est organisé en 10 chapitres.

Le second chapitre est consacré à nos travaux en lien avec la résolution effective du problème SAT comprenant notre approche parallèle portfolio, les travaux sur l'énumération de modèles, l'amélioration des encodages et la compression de formules. Dans le troisième chapitre nous reviendrons sur nos contributions en fouille de données. Nous parlerons de ceux effectués pour la fouille des motifs ensemblistes et séquentiels et les différentes améliorations proposées. Nous

présentons également le problème de clustering d'objets représentés sous forme de formules propositionnelle et les solutions proposées. Le quatrième chapitre est dédié à nos travaux dans les graphes. Nous abordons deux problématiques différentes à savoir la détection de communautés et la question de trouver des représentations compactes. Nous parlerons notamment de notre nouvelle technique de compression de graphes à base de contraintes pseudo-booléennes. Le cinquième chapitre adresse nos contributions dans le cadre de la représentation de connaissances. Nous présentons les travaux dans le cadre de la gestion de l'incohérence mais également ceux en lien avec l'argumentation. Le septième chapitre décrit les approches proposées dans le cadre de la composition des services web. Nous présentons les deux approches visant à trouver une composition optimale en prenant en compte la qualité de service. Nous concluons, dans le chapitre neuf, la description des travaux par un chapitre sur le bilan et les perspectives de nos travaux de recherche. Le dernière chapitre est consacré à la description de mes activités de recherche annexes.

Chapitre 2

Satisfiabilité propositionnelle

2.1 Introduction

Ce premier chapitre est consacré à une description synthétique de mes contributions dédiées à la résolution du problème de satisfiabilité propositionnelle (SAT). Nous commençons par introduire un ensemble de notations et de définitions élémentaires, suivi par un bref aperçu de solveurs SAT modernes. La suite du chapitre est consacrée à la description de nos travaux dans ce domaine. Ces contributions peuvent être scindées naturellement en cinq catégories : (i) les approches dédiées à l'amélioration de la résolution du problème SAT, (ii) l'utilisation du parallélisme en satisfiabilité propositionnelle, (iii) l'énumération des modèles d'une formule CNF, (iv) l'encodage de la contrainte de cardinalité (conditionnelle) et finalement (v) la compression de formules booléennes et contraintes tables.

2.2 Définitions et notations

Dans cette section, nous allons présenter les principaux éléments de la logique propositionnelle nécessaire pour la suite du manuscrit.

Soit $\mathcal{L}$ un langage propositionnel défini à partir d'un ensemble fini $\mathcal{PS}$ de symboles propositionnels (p, q, r, etc.). L'ensemble des formules est défini de manière inductive à partir de $\mathcal{PS}$, la constante $\bot$ (*Faux*), la constante $\top$ (*Vrai*), et utilisant les connecteurs logiques usuels $\neg$, $\wedge$, $\vee$, $\rightarrow$ ainsi que le connecteur d'équivalence $\leftrightarrow$. Pour chaque deux formules propositionnelles Φ et Ψ de $\mathcal{L}$, le connecteur $\leftrightarrow$ est défini par $\Phi \leftrightarrow \Psi \equiv (\Phi \rightarrow \Psi) \wedge (\Psi \rightarrow \Phi)$. Dans la suite, nous nous référons à l'ensemble des symboles de $\mathcal{PS}$ qui apparaissent dans la formule Φ comme $\mathcal{S}(\Phi)$.

Étant donnée une formule $\Phi \in \mathcal{L}$, une *interprétation booléenne* $\mathcal{B}$ de Φ est une fonction de $\mathcal{S}(\Phi)$ dans $\{0, 1\}$. Soient deux formules $\Phi, \Psi \in \mathcal{L}$, nous avons les équivalences suivantes : $\mathcal{B}(\bot) = 0$, $\mathcal{B}(\top) = 1$, $\mathcal{B}(\neg\Phi) = 1 - \mathcal{B}(\Phi)$, $\mathcal{B}(\Phi \wedge \Psi) = min(\mathcal{B}(\Phi), \mathcal{B}(\Psi))$, $\mathcal{B}(\Phi \vee \Psi) = max(\mathcal{B}(\Phi), \mathcal{B}(\Psi))$, et $\mathcal{B}(\Phi \rightarrow \Psi) = max(1 - \mathcal{B}(\Phi), \mathcal{B}(\Psi))$.

Un *modèle* d'une formule $\Phi \in \mathcal{L}$ est une interprétation booléenne $\mathcal{B}$ qui satisfait Φ, i.e.,a $\mathcal{B}(\Phi) = 1$. $\mathcal{M}(\Phi)$ représente l'ensemble des modèles de la formule Φ, i.e., $\mathcal{M}(\Phi) = \{M \mid M \models \Phi\}$. Une formule Φ est *satisfiable* ssi $\mathcal{M}(\Phi) \neq \emptyset$. Soient $X \subseteq \mathcal{S}(\Phi)$ et $\mathcal{B} \in \mathcal{M}(\Phi)$, nous définissons $\mathcal{B}^X = \{x \in X \mid \mathcal{B}(x) = 1\}$ et $\mathcal{M}^X(\Phi) = \{\mathcal{B}^X \mid \mathcal{B} \in \mathcal{M}(\Phi)\}$. Une interprétation $\mathcal{B}$ de Φ est dite *impliquant premier* (PI en court) si et seulement si pour tout littéral $a \in \mathcal{B}$, $\mathcal{B} \setminus \{a\} \not\models \Phi$. Nous notons par $PI(\Phi)$ la disjonction de tous les impliquants premiers de Φ. Évidemment, $PI(\Phi)$ est logiquement équivalent à Φ. Nous utilisons le symbole $\vdash$ pour désigner la relation de déduction ou d'inférence. Notons que le symbole $\vdash$ n'est pas un nouveau symbole du langage mais appartient au métalangage. $\vdash_*$ désigne la relation de déduction modulo la propagation unitaire.

Une formule sous *forme normale conjonctive* (CNF) est une conjonction (finie) de clauses. Une *clause* est une disjonction (finie) de littéraux. Un *littéral* est soit une variable propositionnelle $x \in \mathcal{PS}$ ou sa négation $\neg x$. Notons également que toute formule propositionnelle peut être transformée en une formule CNF équivalente pour la satisfiabilité, en utilisant l'encodage linéaire de Tseitin [Tseitin 1968]. Une clause c est dite *impliqué premier* d'une formule satisfiable Φ si $\Phi \vdash c$, c n'est pas tautologique, et pour toute clause $c' \subset c$, $\Phi \nvdash c'$.

Le problème de la satisfiabilité propositionnelle, appelé *SAT*, est le problème de décision qui consiste à déterminer si une formule CNF est satisfiable ou non. Ce problème a été montré NP-complet [Cook 1971]. Depuis les années soixante et les fameux travaux de Davis et Puttnam [Davis *et al.* 1962], le problème SAT a fait l'objet de nombreux travaux sur le plan théorique et pratique. Plusieurs algorithmes ont été proposés et raffinés tout le long des décennies passées. Ces recherches ont aboutis à des procédures (solveurs SAT modernes) efficaces en pratique conduisant à une vraie révolution dans la résolution des instances SAT et un succès indéniable sur le plan industriel. Au delà de l'application phare qui est la vérification formelle (Bounded Model Checking) [Clarke *et al.* 2001], le champ d'application de SAT n'a cessé de s'accroître au cours des années allant de la bio-informatique à la correction de circuits électroniques vis-à-vis de leur spécification en passant par l'ordonnancement [Crawford & Baker 1994], et la fouille de données [Boudane *et al.* 2016]. Cette expansion du champ d'application s'accompagne par une difficulté croissante dûe à la structure intrinsèque de certains problèmes mais parfois à la simplicité du langage propositionnel menant à des formules de plus en plus volumineuses.

2.3 Approches de résolution du problème SAT

Les algorithmes dédiés à la résolution du problème SAT peuvent être scindés en deux principales catégories : complet et incomplet. Dans la catégorie des solveurs complets, le solveur dit solveur SAT moderne (Conflict Driven Clause Learning) [Marques-Silva & Sakallah 1996, Moskewicz *et al.* 2001] est l'archetype des solveurs de cette catégorie et le symbole de leurs succès. C'est une amélioration notable de l'algorithme DPLL [Davis *et al.* 1962].

Un algorithme SAT de type CDCL est une procédure de recherche arborescente de type retour-arrière (backtrack) qui développe un arbre binaire. En chaque nœud, une variable de la formule est choisie et affectée à l'une de ces valeurs de vérités suivi de la propagation unitaire. Si tous les littéraux sont affectés ou propagés sans contradiction, alors la formule est déclarée satisfiable et l'interprétation courante est un modèle de la formule. Dans le cas contraire, un conflit a lieu, i.e., une clause est rendue fausse par l'interprétation courante (tous ces littéraux sont affectés à $faux$).

CDCL [Zhang *et al.* 2001] se distingue par la manière de capturer la propagation unitaire en utilisant des structures intelligentes ("Watched Literals"), de traiter les conflits (application de la résolution en utilisant les graphes d'implications), et par l'utilisation des heuristiques de choix de variables (VSIDS) pour circonscrire la partie conflictuelle de l'espace de recherche. Cette procédure a été également enrichie par des méthodes visant à améliorer la résolution en recommençant régulièrement la recherche (redémarrage) pour diversifier la recherche et en maintenant une base de clauses apprises de qualité (supprimer régulièrement des clauses jugées non pertinentes).

2.4 Élimination de symétries

Dans cette section, nous parlons de symétries, un concept largement étudié en littérature et qui a fait l'objet de certains de nos travaux. La symétrie est un concept fondamental en informatique, mathématiques, physiques et plein d'autres domaines. Elle existe dans divers problèmes réels. Les symétries sont communément exploitées dans la résolution de problèmes combinatoires, tels que les problèmes d'ordonnancement. Par exemple, dans un problème d'ordonnancement où certaines machines peuvent être interchangeables, partant d'un ordonnancement valide, on peut en obtenir un autre valide en permutant ces machines. Exploiter les symétries permet de réduire le coût de la recherche de solutions, en évitant d'explorer des branches symétriques de l'espace de recherche. Beaucoup de travaux ont ainsi porté sur les symétries dans les problèmes de satisfaction de contraintes (CSP) [Gent & Smith 2000, Cohen *et al.* 2006], satisfaisabilité propositionnelle [Benhamou & Sais 1994, Crawford 1992], en formules booléennes quantifiées [Audemard *et al.* 2007], et en recherche opérationnelle [Margot 2003, Liberti 2012].

Tout d'abord, introduisons quelques concepts de la théorie des groupes. Un groupe $(\mathcal{G}, \circ)$ est un ensemble fini $\mathcal{G}$ avec un opérateur binaire associatif $\circ : \mathcal{G}\times\mathcal{G} \rightarrow \mathcal{G}$ admettant un élément neutre et un élément inverse. L'ensemble de toutes les permutations $\mathcal{P}$ sur un ensemble fini E associé à l'opérateur de composition $\circ$, noté $(\mathcal{P}, \circ)$, forme un groupe. En outre, chaque permutation $\sigma \in \mathcal{P}$ peut être représentée par un ensemble de cycles $\{c_1 \dots c_n\}$ tel que chaque cycle c_i est un ensemble d'éléments de E $(l_{i_1} \dots l_{i_{n_i}})$ avec $\forall 1 \leqslant k < n_i, \sigma(l_{i_k}) = l_{i_{k+1}}$ et $\sigma(l_{i_{n_i}}) = l_{i_1}$.

Soient Φ une formule CNF et σ une permutation sur $Lit(\Phi)$. On peut étendre le concept de permutation σ à Φ comme suit : $\sigma(\Phi) = \{\sigma(c) \mid c \in \Phi\}$ et $\sigma(c) = \{\sigma(l) \mid l \in c\}$.

Définition 1. *Soient Φ une formule CNF et σ une permutation sur les littéreaux de Φ. σ est une symétrie de Φ si elle satisfait les conditions suivantes :*

— $\sigma(\neg x) = \neg\sigma(x)$, $\forall x \in \mathcal{L}(\Phi)$,
— $\sigma(\Phi) = \Phi$.

Par conséquent, une symétrie σ définit une relation d'équivalence sur l'ensemble des affectations possibles. L'élimination des symétries consiste à supprimer toutes les interprétations symétriques sauf une dans chaque classe d'équivalence. Usuellement, l'élimination d'une symétrie $\sigma = (x_1, y_1)(x_2, y_2)\dots(x_n, y_n)$ s'effectue par ajout de contraintes appelées *SBP* (Symmetry Breaking Predicates) [Crawford 1992, Crawford *et al.* 1996, Aloul *et al.* 2003a, Aloul *et al.* 2003b, Walsh 2006] définit formellement comme suit :

$$(x_1 \leqslant y_1)$$

$$(x_1 = y_1) \rightarrow (x_2 \leqslant y_2)$$

$$\vdots$$

$$(x_1 = y_1) \wedge (x_2 = y_2) \wedge \dots \wedge (x_{n-1} = y_{n-1}) \rightarrow (x_n \leqslant y_n)$$

L'ajout de ces contraintes permet de préserver l'équivalence pour la satisfiabilité entre la formule originale et celle augmentée par le SBP. De plus, la suppression d'un ensemble de symétries s'effectue en ajoutant le SBP associé à chacune d'entre elles.

2.5 Résolution séquentielle

Durant les années qui ont suivi ma thèse, nous nous sommes intéressés à améliorer la résolution effective du problème SAT. Les limites observés dans sa résolution nous ont encouragés à chercher d'autres alternatives capables de rendre la résolution plus efficace. Nous avons explorés plusieurs directions de recherche allant de l'amélioration du schéma d'apprentissage des clauses à l'introduction d'une approche parallèle. Nous nous somme également intéressés aux techniques de pré-traitement en amont de la résolution en explorant des encodages de certaines contraintes usuellement utilisées lors des encodages ou en proposant des représentations plus compactes.

Dans la suite nous présentons les travaux réalisés en lien étroit avec la résolution du problème SAT.

2.5.1 Amélioration du schéma d'apprentissage de clauses

L'ajout des clauses obtenues par résolution à chaque conflit (apprentissage de clauses) permet d'éviter de revisiter des sous-espaces de recherche déjà exploré. Toutefois, le nombre de ces clauses est connu pour être exponentiel dans le pire cas. Dans [Jabbour 2009, Hamadi *et al.* 2009, Jabbour *et al.* 2013b, Hamadi *et al.* 2012, Hamadi *et al.* 2016], nous nous sommes intéressés à améliorer le schéma d'apprentissage de clauses en proposant plusieurs modèles. En effet, lors de l'encodage de certains problèmes d'application vers SAT, et afin de réduire la taille de ce dernier, souvent des variables supplémentaires sont ajoutées. Ces variables sont généralement exprimées en fonction de celles du problème initial par le biais de fonctions booléennes. Ces fonctions booléennes de la forme $y \leftrightarrow f(x_1, \ldots, x_n)$, $f \in \{\wedge, \vee\}$ peuvent êtres découvertes soit syntaxiquement ou sémantiquement en utilisant la propagation unitaire [Grégoire *et al.* 2004]. Dans [Jabbour *et al.* 2013b], nous avons proposé d'améliorer la qualité des clauses apprises en substituant certaines de leurs variables par leurs définitions. Le but est de faire en sorte que les contraintes portent plus sur les variables originales du problème.

Améliorer le schéma d'apprentissage de clauses a fait également l'objet d'un autre travail [Hamadi *et al.* 2009]. Dans ce dernier, nous avons exploré dans quelle mesure utiliser le graphe d'implications pour déduire de nouvelles clauses utiles pour la recherche i.e., ayant du *pouvoir de propagation*. Dans [Pipatsrisawat & Darwiche 2008], les auteurs ont exploré cette notion en considérant les clauses assertives (au moment de leur génération) en se basant sur une forme de résolution particulière appelée *merge résolution* [Andrews 1968], i.e., $\Phi \vdash x \vee y \vee \alpha$ et $\Phi \wedge \neg x \wedge \neg \alpha \nvdash_* y$. Dans [Jabbour *et al.* 2013b], nous avons proposé de générer par résolution des clauses particulières dites *bi-assertives* dans l'esprit de celles proposées dans [Pipatsrisawat & Darwiche 2008] en partant de chacun des deux littéraux en contradiction lors d'un conflit et en effectuant des résolutions jusqu'au littéral assertif.

2.5.2 Simplification des bases des clauses apprises

Le nombre de clauses pouvant être générées par résolution est connu pour être exponentiel dans le pire des cas. Bien que la résolution est appliquée uniquement en cas de conflit, ce nombre reste également exponentiel. Garder toutes les clauses apprises présente l'inconvénient de ralentir la propagation unitaire et par conséquent la résolution. Afin de maintenir un équilibre entre l'apprentissage de clauses et l'efficacité de la propagation unitaire, il est commode

dans les solveurs SAT de supprimer régulièrement certaines clauses jugées non pertinentes pour le reste de la recherche. Plusieurs travaux [Audemard & Simon 2009] ont essayé de mettre en exergue des stratégies capable de juger de la pertinence des clauses apprises et par conséquent de décider de celles à maintenir et celles à supprimer. Dans [Jabbour *et al.* 2014b], nous avons exploré ce champ en partant de l'observation que les clauses de taille réduite ont plus de probabilité d'être utilisées dans le processus de propagation ou dans la construction des preuves d'insatisfiabilité. Nous avons proposé plusieurs stratégies privilégiant les clauses de taille réduite et faisant un choix arbitraire sur le reste des clauses. Choisir de manière arbitraire les clauses à garder consiste à opérer une forme de diversification de la recherche. Nous avons proposé également d'autres variantes comme celle basée sur la taille relative de chaque clause, i.e., la taille modulo la propagation unitaire. Certaines stratégies se sont montrées particulièrement efficaces sur plusieurs classes de problèmes, principalement sur des problèmes issues de la cryptographie [Soos *et al.* 2009, Massacci & Marraro 2000]. Ces stratégies ont été implantées dans *Minisat* donnant lieu à un nouveau solveur appelé *X-Minisat* [1].

2.6 Résolution parallèle du problème SAT

Dans cette section, nous donnons un aperçu de nos travaux effectués dans le cadre de la résolution parallèle du problème SAT.

2.6.1 Approche portfolio

L'avénement récent des nouvelles architectures multicœurs a permis de raviver l'intérêt pour le parallélisme et l'emergence d'une génération de solveurs SAT parallèles [Heule *et al.* 2018, Audemard *et al.* 2012, Chu *et al.* 2008] visant à pousser les limites des problèmes pouvant être résolus en pratique. Certaines approches ont remis au goût du jour le paradigme *diviser pour régner* usuellement utilisant les chemins de guidage dans les premières générations de solveurs SAT [Chrabakh & Wolski 2003, Böhm & Speckenmeyer 1996, Jurkowiak *et al.* 2001]. Ce principe consiste à diviser l'espace de recherche entre différents cœurs de calcul de façon disjointe en attribuant à chacun un sous-espace à explorer (sous-formule). Toutefois, cette approche est confrontée à certaines problèmes, principalement l'équilibrage de charge. De plus, le nombre de variables des instances SAT actuelles est élevé rendant l'approche moins efficace lorsque le nombre de cœurs de calcul est limité. Partant du constat que les solveurs SAT sont sensibles aux valeurs de leur paramètres, i.e., un changement minimal dans la valeur d'un paramètre peut avoir une grande influence sur le temps de calcul, nous avons proposé *ManySAT* [Hamadi *et al.* 2009], le premier solveur parallèle dont l'architecture est basée sur un portfolio de solveurs de type CDCL changés en leur principal composants. ManySAT permet de lancer plusieurs solveurs de type CDCL (cœur de calcul) incarnant chacun une stratégie différente. Le but est de permettre une exploration différente de l'espace de recherche. D'autres part, afin d'éviter de revisiter des sous espaces de recherche redondants, les différents cœurs de calcul échangent régulièrement certaines clauses entre eux. Intuitivement les clauses de taille réduite sont échangées permettant d'améliorer considérablement la résolution. Empiriquement, dans la première version de ManySAT, nous

1. http ://www.cril.univ-artois.fr/ sais/X-SAT/

avons constaté qu'échanger des clauses de taille inférieure à 10 permet d'obtenir les meilleures performances.

Le solveur ManySAT s'est distingué en remportant la première compétition dédiée aux solveurs SAT parallèles organisée en 2009 [2].

2.6.2 Échange de clauses

La stratégie de partage de clauses proposée dans [Hamadi *et al.* 2009] a certaines limitations. En effet, pour certains problèmes, la résolution passe par des phases où les clauses générées ont une taille plus élevée que la limite fixée. Pour certains autres, le nombre de clauses ayant une taille inférieure au seuil fixée peut s'avérer très élevé forçant les cœurs de calcul à passer plus de temps à échanger des clauses qu'à explorer l'espace de recherche. Dans [Hamadi *et al.* 2011], nous avons proposé une politique dynamique de partage de clauses qui utilise des limites de taille par paires pour contrôler l'échange entre chaque deux cœurs de calcul. L'évolution des limites par paires suit une loi AIMD (*Additive-Increase-MultiplicativeDecrease*), un algorithme de contrôle par rétroaction [Chiu & Jain 1989]. L'algorithme Additive Increase/Multiplicative Decrease (AIMD) est un algorithme de contrôle de débit utilisé pour éviter la congestion dans les réseaux de communication. Le problème résolu par AIMD est de deviner la bande passante de communication disponible entre deux nœuds communicants. L'algorithme effectue des sondes successives, augmentant le taux de communication w linéairement tant qu'aucune perte de paquets n'est observée, et la diminuer exponentiellement quand une perte est rencontrée. Plus précisément, l'évolution de w est définie par la formule AIMD(a, b) suivante :

$$w = \begin{cases} w - a \times w & \text{si une perte est détectée} \\ w + b & \text{sinon.} \end{cases}$$

Pour remédier au problème de la pertinence des clauses partagées, nous étendons notre politique pour intégrer la qualité des échanges. Chaque cœur de calcul évalue la qualité des clauses reçues, et le contrôle est capable d'augmenter/diminuer les limites par paires basées sur la qualité sous-jacente des clauses récemment reçues. La justification étant que les informations récemment reçues d'une source particulière est qualitativement liée à l'information qui pourrait être reçu de lui dans un avenir très proche.

Dans [Audemard *et al.* 2012], nous avons continué l'exploration de l'échange de clauses en parallèle. Une nouvelle alternative issue des travaux dans le cadre de la résolution séquentiel de SAT a été développée. Il s'agit d'adapter l'heuristique basée sur la stratégie *psm* définie dans [Audemard *et al.* 2011] au cadre parallèle basé sur un portfolio de solveurs. Plus précisément, les clauses sont échangées entre les cœurs de calcul si elles n'excèdent pas une taille fixe. Ces clauses ne sont pas réellement intégrées à la recherche que si elles sont potentiellement utiles à la recherche, i.e., très proche de l'interprétation en cours d'exploration. Dans le cas contraire elles sont gelées et vérifiées périodiquement.

Dans [Lazaar *et al.* 2012], nous nous sommes intéressés à l'échange de clauses à grande échelle d'une architecture parallèle. Nous avons montré que la topologie de communication complète (e.g., ManySAT2.0) arrive très souvent à son limite lorsque le nombre de coeurs est important. Pour pallier ce problème, nous avons introduit une nouvelle technique décentralisée qui

2. http ://www.satcompetition.org/

permet de contrôler l'échange et le partage de clauses entre les différents coeurs. Notre approche consiste à sélectionner pour chaque cœur de calcul, les potentiels cœurs pouvant communiqués avec lui, i.e, autorisés à partager des clauses apprises avec ce dernier de manière décentralisée et dynamique. Notons que la longueur maximale des clauses partagées est fixe préalablement. Pour ce faire, nous nous sommes basés sur une approche bandit multi-armée baptisée Bandit Ensemble for Parallel SAT Solving (BESS) afin de gérer de manière dynamique l'échange entre les différents coeurs de calcul.

2.6.3 Intensification et diversification

Dans les solveurs SAT modernes, les redémarrages constitue une forme de diversification de la recherche alors que la pondération des variables des conflits vise à intensifier la recherche en focalisant la recherche sur des sous-espaces conflictuels. Dans les approches parallèles de type portfolio, la diversification est facilement mise en œuvre par le biais des paramètres des solveurs. Par contre l'intensification de la recherche n'a pas été suffisamment explorée. Afin de trouver un compromis entre ces deux principes d'intensification et de diversification, nous avons proposé dans [Guo *et al.* 2010] une nouvelle approche parallèle basée sur le principe *maître/esclave*. Plus précisément, certains cœurs de calcul sont choisis pour être des maîtres et ont pour objectif de diversifier la recherche. Les cœurs de calcul restants, appelés esclaves, ont pour but d'intensifier la recherche de leurs maîtres. L'intensification consiste à forcer les esclaves à explorer différemment des sous-espaces proche de ceux en cours d'exploration par les maîtres. Formellement, dans [Guo *et al.* 2010], les maîtres partagent régulièrement avec les esclaves les affectations partielles à considérer. Ces dernières sont composées de littéraux des derniers conflits. Cette stratégie a montré son efficacité principalement sur les instances connues pour être insatisfiables.

Dans [Guo *et al.* 2014], nous avons considéré une autre alternative pour la diversification de la recherche. Dans ce travail, nous nous sommes intéressés à l'impact des stratégies de réduction de la base des clauses apprises sur la parallélisation des approches SAT de type portfolio. Nous avons proposé de diversifier la recherche en forçant chacun des cœurs à réduire sa base de clauses apprises en choisissant une stratégies différente parmi celles proposées dans la littérature [Audemard & Simon 2009] tout en proposant de nouvelles.

2.6.4 Parallèle & déterminisme

L'approche parallèle permet de diminuer le temps nécéssaire à la résolution de certains problèmes SAT et d'en résoudre d'autres jusqu'au là inaccessible par les solveurs séquentiels. Toutefois, cela s'accompagne par une problématique liée au déterminisme de ces solveurs. En effet, à cause de l'échange de clauses, lorsque l'instance est satisfiable le modèle trouvé peut être différent d'une exécution à une autre. Même constat peut être fait pour la preuve par résolution pour la réfutation. Pour remédier à ce problème, nous avons proposé un solveur SAT parallèle déterministe [Hamadi *et al.* 2011]. Ses résultats sont entièrement reproductibles, c'est-à-dire, une exploration parallèle reproductible de l'espace de recherche, qui comprend le même modèle rapporté ou preuve de l'insatisfiabilité. Nous avons proposé un solveur qui définit un environnement contrôlé basé sur un ordre total des interactions des solveurs à l'aide de barrières de synchronisation. Pour maximiser l'efficacité, l'échange d'informations (clauses de conflit) et la vérification de

la terminaison sont effectués régulièrement. La fréquence de ces échanges influence grandement sur la performance de notre solveur. Nous avons proposé dans ce cadre une stratégie dynamique pour décider de la fréquence de synchronisation.

2.6.5 Approche distribuée

Dans [Audemard *et al.* 2014], nous avons proposé un autre cadre pour résoudre le problème SAT en parallèle en se basant sur le principe de diviser pour régner dans un environnement distribué afin de pouvoir profiter des architectures distribuées offrant un large nombre de cœurs de calcul. En se basant sur une architecture maître/esclave utilisant le principe de diviser pour régner, nous avons conçu un solveur où le maître se charge d'attribuer les sous-espaces de recherche à explorer lorsqu'un esclave a fini d'exécuter sa tâche et qui utilise l'équilibrage de charge pour gérer l'inactivité des cœurs de calcul [Chrabakh & Wolski 2003, Chu *et al.* 2008]. Cependant, ce cadre est exposé à certains cas pathologiques dans le contexte SAT, comme l'effet *ping pong* ou encore la *division inutile* pouvant considérablement ralentir les performances de l'approche. Nous avons proposé plusieurs améliorations visant à éviter ces cas problématiques en munissant les esclaves de capacités à refuser de diviser l'espace de recherche sous certaines conditions. Ce travail a donné lieu à un solveur appelé *Dolius* [3].

2.7 Énumération de modèles

Comme ça été mentionné auparavant, avec l'expansion des domaines d'application de SAT, le problème d'énumération de tous ou une partie des modèles de formules sous forme CNF connait un vrai regain d'intérêt. Nous allons à présent détailler nos travaux portant sur l'énumération des modèles d'une formule propositionnelle.

2.7.1 Énumération de tous les modèles

Généralement, les approches d'énumération de modèles sont basés sur les solveurs SAT modernes. Ces implantations sont basées sur l'ajout de clauses, appelées *clauses bloquantes*, permettant d'éviter la génération des modèles redondants [McMillan 2002, Chauhan *et al.* 2003, Jin *et al.* 2005]. Ces clauses peuvent correspondre à la négation du modèle trouvé. D'autres approches ont essayé d'améliorer cette procédure [Jin *et al.* 2005]. Dans [Morgado & Marques-Silva 2005], les auteurs proposent plusieurs optimisations obtenues par apprentissage et simplification des clauses bloquantes. Le but est de trouver la plus petite interprétation partielle satisfaisant la formule et couvrant le modèle trouvé par le solveur. Une interprétation partielle de taille k sur une formule contenant n variables représente les 2^{n-k} modèles et ajouter la clause bloquante associée permet d'éviter l'énumération d'un tel ensemble de modèles. Cependant, cette approche naïve présente certaines limites. Tout d'abord, de point de vue complexité, le nombre de modèles à énumérer peut être exponentiel. Cumuler à la fois des clauses apprises issues des conflits et celles dites bloquantes peut altérer l'efficacité de la résolution. Le deuxième problème est lié à l'utilisation de l'heuristique de branchement VSIDS connue pour être plus appropriée pour trouver un seul modèle ou prouver qu'aucun modèle n'existe. De plus, il est bien connu que l'ordre

3. http ://www.cril.univ-artois.fr/ hoessen/dolius.html

utilisé pour l'énumération de tous les modèles peut changer radicalement la complexité de calcul [Creignou *et al.* 2011]. Dans [Dechter & Itai 1992], les auteurs ont montré qu'énumérer tous les modèles d'une théorie donnée peut être effectuée dans le temps proportionnel au produit du nombre de modèles à l'effort nécessaire pour générer chaque modèle séparément. Par conséquent, un moyen efficace pour améliorer le processus d'énumération consisterait à chercher tous les modèles à proximité du modèle trouvé à une itération donnée.

Motivé par les observations ci-dessus, nous avons introduit dans [Jabbour *et al.* 2014a] une extension des solveurs CDCL pour énumérer tous les modèles d'une formule CNF. Notre approche procède de la manière suivante : la première étape consiste à chercher le premier modèle en utilisant un solveur CDCL classique. Lorsque ce dernier est trouvé, le solveur continue la recherche des autre modèles en inhibant les redémarrages.

2.7.2 Enumération des impliquants premiers

Les implicants premiers (PI) des fonctions booléennes est un sous-ensemble de générateurs de tous les modèles. Ils ont été utilisé dans le contexte de la minimisation des fonctions booléennes par Quine [Quine 1952, Quine 1959] et McCluskey [McCluskey 1956]. Cette première application de la forme canonique des impliquants premiers est importante car cela permet de réduire la taille et le coût du circuit numérique tout en améliorant la vitesse (par exemple, [Tison 1967]). En plus de l'analyse et de l'optimisation des circuits numériques, les impliquants premiers ont trouvé plusieurs autres domaines d'application, y compris l'analyse de l'arbre de défaillance [Coudert & Madre 1993, Dutuit & Rauzy 1997], la bio-informatique [Acuña *et al.* 2012], les bases de données [del Val 1994], le diagnostique [de Kleer *et al.* 1990], et la représentation des connaissances et du raisonnement [Cadoli & Donini 1997]. Il est également importante dans de nombreux domaines de l'intelligence artificielle tels que le raisonnement automatique et non monotone [Ginsberg 1989], la compilation de connaissances [Darwiche & Marquis 2002], les systèmes multi-agents [Slavkovik & Agotnes 2014]. Cependant, le problème de génération de tous les impliquants premiers d'une théorie propositionnelle est une tâche très complexe.

Dans [Jabbour *et al.* 2014f], nous avons proposé une approche originale qui réécrit une formule CNF Φ comme une nouvelle formule CNF Φ' telle que les impliquants premiers de Φ correspondent aux modèles de Φ'. Notre encodage emprunte l'idée de renommage des littéraux utilisée dans les formulations en programmation linéaire (ILP) proposées dans [Pizzuti 1996, Manquinho *et al.* 1997, Manquinho *et al.* 1998]. Formellement, soit Φ une formule CNF. Nous associons à chaque littéral a de $Lit(\Phi)$ une variable propositionnelle x_a. Nous définissons la formule CNF Φ^R comme la formule obtenue à partir de Φ en renommant chaque littéral $a \in Lit(\Phi)$ par sa variable propositionnelle correspondante x_a, et en ajoutant les clauses binaires suivantes :

$$\bigwedge_{a \in Var(\Phi)} (\neg x_a \vee \neg x_{\neg a}) \tag{2.1}$$

La contrainte $\mathcal{M}(\Phi^R)$ définie dans (2.2) permet d'établir une bijection entre les impliquants premiers de la formule Φ et les modèles de la formule résultante.

$$\mathcal{M}(\Phi^R) = \bigwedge_{a \in Lit(\Phi)} (x_a \rightarrow \neg \bigwedge_{c \in \Phi^R,\ x_a \in c} c \setminus \{x_a\}) \tag{2.2}$$

L'énumération des impliquants premiers est alors réduite au problème de l'énumération de tous les modèles d'une formule CNF. Une telle correspondance permet de bénéficier des performances des solveurs SAT modernes. Cependant, mettre (2.2) sous forme CNF peut augmenter considérablement le nombre de variables et de clauses additionnelles. Dans notre deuxième contribution, nous avons étendu notre approche en proposant un encodage plus compacte en exploitant la structure des formules. Plus précisément, grâce aux fonctions booléennes généralement présentes dans les problèmes réels, des réductions significatives peuvent être opérées.

2.7.3 Enumération des modèles Top-k

Souvent dans le monde réel, l'énumération de tous les modèles n'est pas requise et est remplacée par l'énumération de certains modèles dit *préférés*. La notion de préférence joue un rôle primordial dans plusieurs disciplines telles que l'économie, la recherche opérationnelle, et la théorie de la décision en général. Les préférences sont pertinentes pour la conception de systèmes intelligents qui prennent en charge les décisions. La modélisation et le raisonnement avec les préférences jouent un rôle croissant en intelligence artificielle et ses domaines connexes tels que le raisonnement non monotone, la planification, le diagnostique, la configuration, la programmation par contraintes (CP), etc. Par exemple, dans le raisonnement non monotonique, l'introduction de la sémantique préférentielle par Shoham [Shoham 1988] donne un cadre unificateur où la logique non monotone est réduite à une logique standard avec une relation de préférence (ordre) sur les modèles de cette logique. Plusieurs modèles de représentation et de raisonnement à propos des préférences ont été proposés. Par exemple, les contraintes souples [Meseguer *et al.* 2006] sont l'un des moyens les plus généraux pour traiter les préférences quantitatives, tandis que les CP-net (réseaux de préférences conditionnelles) [Boutilier *et al.* 2004] sont plus pratique pour les préférences qualitatives. Il existe une littérature abondante sur les préférences (voir [Walsh 2007, Brafman & Domshlak 2009, Domshlak *et al.* 2011] pour plus de détails). Dans l'exploration de données, les préférences ont également été étudiées par plusieurs auteurs [Bistarelli & Bonchi 2007, de Amo *et al.* 2012, Ugarte Rojas *et al.* 2014]. Par exemple, dans [Bistarelli & Bonchi 2007], les auteurs ont présenté un nouveau paradigme de découverte de modèle basé sur des contraintes souples, où les contraintes ne sont plus des fonctions booléennes rigides. Plus récemment, Ugarte et al. [Ugarte Rojas *et al.* 2014] ont introduit une méthode générique et efficace basée sur la programmation par contraintes pour l'extraction de skypatterns (soft-) qui permettent d'exprimer la préférence de l'utilisateur en fonction des relations de dominance. Partant de l'observation que les relations de dominance sont présentes dans de nombreux problèmes, dans [Négrevergne *et al.* 2013], les auteurs proposent une algèbre qui combine les contraintes et les relations de dominance qui peuvent être utilisées pour décrire un large éventail de paramètres d'exploration de modèles.

Dans [Jabbour *et al.* 2013d, Jabbour *et al.* 2017d], nous nous sommes intéressés aux préférences qualitatives définies par une relation de préférence sur les modèles d'une formule propositionnelle qui n'ont pas reçu beaucoup d'attention [Rosa *et al.* 2010, Castell *et al.* 1996]. Nous avons introduit un cadre générique pour traiter les préférences qualitatives des formules propositionnelle. Nos préférences qualitatives sont définies à l'aide d'une relation réflexive et transitive $\geq$ (pré-ordre) sur les modèles d'une formule propositionnelle. Cette relation de préférence sur les modèles est d'abord utilisée pour introduire un nouveau problème, appelé Top-k SAT, défini

comme le problème de l'énumération des modèles Top-k d'une formule propositionnelle. Ici, un *modèle Top-k* est défini comme un modèle n'ayant pas plus que $(k-1)$ modèles préférés à lui par rapport à la relation de préférence considérée. Nous avons également montré que Top-k SAT généralise les deux problèmes bien connus à savoir le problème MaxSAT partiel et le problème de génération de modèles minimaux. Un algorithme général pour calculer les modèles Top-k est également proposé pour des relations de préférences particulières.

2.8 Contraintes de cardinalité & problème des pigeons

Parmi les contraintes les plus récurrentes lors de l'encodage de certains problèmes d'applications vers SAT, les contraintes de cardinalité sont sans doute les plus fréquentes. Plusieurs applications impliquent le comptage d'arguments exprimés sous forme de contraintes de cardinalité ou contraintes pseudo-booléennes. Ce type de contraintes est souvent dû à l'encodage de problèmes du monde réel, tels que l'assignation de fréquences radio, la programmation temporelle et les problèmes de configuration de produits pour n'en citer que quelques-uns. Ces contraintes sont exprimées sous la forme suivante :

$$\sum_{i=1}^{n} x_i \# k \qquad x_i \in \{0,1\}, \quad k \in \mathbb{Z}, \qquad \# \in \{\leq, \geq, =\} \tag{2.3}$$

La transformation de la contrainte (2.3) en CNF a fait l'objet de nombreux travaux visant à proposer des encodages efficaces en utilisant plusieurs modèles incluant les *BDD*, les *réseaux de tri*, etc. [Warners 1996, Sinz 2005, Silva & Lynce 2007, Asín *et al.* 2009, Eén & Sörensson 2006, Bailleux *et al.* 2009]. L'efficacité fait référence à la fois à la compacité de la représentation (taille de la formule CNF) et à la capacité d'atteindre le même niveau de propagation de contraintes (cohérence d'arc généralisée) sur la formule CNF. Souvent ces transformations se basent sur l'ajout de nouvelles variables et clauses pour réduire la taille de l'encodage. Nous avons proposé dans [Jabbour *et al.* 2013d, Jabbour *et al.* 2014g] une approche originale basée sur le principe des tiroirs (pigeon-hole problem). Ce dernier affirme qu'il n'y a pas de correspondance injective de k pigeons à n cases telle que $n < k$. Le problème des pigeons peut être transformé en un problème SAT obtenu par l'introduction de variables additionnelles p_{ij} exprimant que le pigeon i est mis dans la case j. La formulation standard du principe des tiroirs est le résultat de la conjonction des contraintes (2.4) et (2.5) suivantes :

$$\bigwedge_{i=1}^{k} \left(\bigvee_{j=1}^{n} p_{ij} \right) \tag{2.4}$$

$$\bigwedge_{l=1}^{k} \bigwedge_{1 \leq i < j < n} (\neg p_{il} \vee \neg p_{jl}) \tag{2.5}$$

Pour transformer la contrainte de cardinalité $\sum_{i=1}^{n} x_i \geqslant k$ en CNF, nous nous sommes basés sur l'encodage erronée de Hooker, dans une note non publiée [Warners 1996], pour donner la bonne formulation en utilisant le principe des tiroirs ci-dessous.

$$\bigwedge_{i=1}^{n}\bigwedge_{j=1}^{k}(x_i \vee \neg p_{ji}) \tag{2.6}$$

$$\bigwedge_{i=1}^{k}(\bigvee_{j=1}^{n} p_{ij}) \tag{2.7}$$

$$\bigwedge_{l=1}^{k}\bigwedge_{1\leq i<j<n}(\neg p_{il} \vee \neg p_{jl}) \tag{2.8}$$

La transformation est polynômiale et nécessite l'ajout de $n(k+1)$ variables et $k + n(k + \frac{k(k-1)}{2})$ clauses. Par conséquent, la complexité globale est en $O(kn)$ variables et $O(nk^2)$ clauses. Toutefois, cet encodage ne permet pas de satisfaire la contrainte de la cohérence d'arc généralisée. Le problème des tiroirs est connu pour être difficile à résoudre. Pour pallier ce problème, dans [Jabbour *et al.* 2013d, Jabbour *et al.* 2014g] nous nous sommes basés sur les symétries très présentes dans cette formulation. Nous avons proposé de combiner les contraintes d'élimination de symétries (voir section 2.4) et les contraintes (2.6), (2.7), et (2.8). Formellement, au lieu d'ajouter conjonctivement les contraintes de suppression de symétries à l'encodage initial, nous avons montré qu'une application intelligente de la résolution entre ces deux ensembles permet d'obtenir le nouveau encodage constitué de 2.9), (2.10) et (2.11) qui satisfait la cohérence d'arc généralisée.

$$\bigwedge_{1\leq i\leq k}(\bigvee_{1\leq j\leq n-k+1} p_{ij}) \tag{2.9}$$

$$\bigwedge_{1\leq i\leq k}\bigwedge_{1\leq j\leq n-k+1}(x_{i+j-1} \vee \neg p_{ij}) \tag{2.10}$$

$$\bigwedge_{1\leq i<k}\bigwedge_{1\leq j<n-k+1}(\neg p_{(i+1)j} \vee \bigvee_{1\leq l\leq j} p_{il}) \tag{2.11}$$

Cet encodage présente l'inconvénient d'être en $O(k(n-k)^2)$ en le nombre d'occurrences des littéraux. Dans [Hattad *et al.* 2017], nous avons montré qu'il est possible de compacter cet encodage afin de maintenir la même complexité en nombres de clauses et de variables mais en réduisant la taille en le nombre de littéraux de $O(k(n-k)^2)$ à $O(k(n-k))$. En effet, comme illustré dans la figure 2.1 représentant un sous-ensemble de clauses issues de (2.11), la sous clause $(p_{11} \vee p_{12} \vee p_{13} \vee p_{23} \vee \ldots \vee p_{2\frac{n-k}{2}})$ est fréquente dans $\frac{k}{2}$ clauses. Par conséquent, le principe d'extension de Tseitin par ajout de variables peut être utilisé permettant de substituer $\frac{n-k}{2} \times \frac{k}{2}$ littéraux par seulement $\frac{kn}{2} + \frac{k}{2}$ autres.

Par ailleurs, nous avons montré que notre approche peut facilement être rendue incrémentale i.e., le changement de la borne de la contrainte de cardinalité ne nécessite pas un encodage depuis le début. Cette approche a été intégrée au solveur *QMaxSAT* [Koshimura *et al.* 2012] pour résoudre des problèmes d'optimisation de type MaxSAT ou MaxSAT partiel.

$$
\begin{array}{l}
p_{11} \vee \neg p_{21} \\
p_{11} \vee p_{12} \vee \neg p_{22} \\
p_{11} \vee p_{12} \vee p_{13} \vee \neg p_{23} \\
\ldots \\
p_{11} \vee p_{12} \vee p_{13} \vee p_{23} \ldots \neg p_{2\frac{n-k}{2}} \\
\ldots \\
p_{11} \vee p_{12} \vee p_{13} \vee p_{23} \ldots p_{2\frac{n-k}{2}} \vee \ldots \vee p_{2(n-k)}
\end{array}
$$

FIGURE 2.1 – Représentation compacte de contraintes

2.9 Contrainte de cardinalité conditionnelle

Dans [Jabbour *et al.* 2017a], nous nous sommes intéressés à une variante de la contrainte de cardinalité appelée *contraintes de cardinalité conditionnelle* exprimée sous la forme $y \rightarrow \sum_{i=1}^{n} x_i \leq k$. Une manière naturelle de coder cette contrainte en CNF consiste à encoder $\sum_{i=1}^{n} x_i \leq k$ sous forme clausale et d'ajouter $\neg y$ à chacune des clauses. Toutefois, cette manière de procéder ne permet pas de préserver l'arc cohérence généralisée. En effet, considérons le cas où $k = 1$. En utilisant la transformation de $\sum_{i=1}^{n} x_i \leq 1$ décrite dans [Sinz 2005] (équivalence à celle obtenue avec le principe des tiroirs), $y \rightarrow \sum_{i=1}^{n} x_i \leq k$ est expriméee comme suit :

$$
\begin{array}{l}
(\neg y \vee \neg x_1 \vee p_1) \wedge (\neg y \vee \neg x_n \vee \neg p_{n-1}) \wedge \\
\bigwedge_{1<i<n} (\neg y \vee \neg x_i \vee p_i) \wedge (\neg y \vee \neg p_{i-1} \vee p_i) \wedge (\neg y \vee \neg x_i \vee \neg p_{i-1})
\end{array}
\tag{2.12}
$$

En utilisant (2.12), on peut déduire que l'affectation de plusieurs littéraux x_i à *vrai* ne permet pas de déduire $\neg y$ par propagation unitaire. En effet, la formule restante est binaire et sans clause unitaire. Afin de pallier ce problème, nous avons montré que si Φ est une formule de horn insatisfiable sous une interprétation partielle de littéraux positifs, alors il est possible de déduire par propagation sur ses clauses mixtes d'autres littéraux positifs rendant l'une de ses clauses négatives totalement fausse. Ce résultat permet de décider des clauses à augmenter par $\neg y$ afin de préserver l'arc cohérence généralisée à savoir les clauses totalement négatives. Notons que la majorité des encodages de la contrainte de cardinalité $\sum_{i=1}^{n} x_i \leq k$ sont des formules de horn. Dans le cas particulier de la contrainte $y \rightarrow \sum_{i=1}^{n} x_i \leq 1$, cela donne lieu à la formulation suivante :

$$
\begin{array}{l}
(\neg x_1 \vee p_1) \wedge (\neg y \vee \neg x_n \vee \neg p_{n-1}) \wedge \\
\bigwedge_{1<i<n} (\neg x_i \vee p_i) \wedge (\neg p_{i-1} \vee p_i) \wedge (\neg y \vee \neg x_i \vee \neg p_{i-1})
\end{array}
\tag{2.13}
$$

2.10 Représentation compacte

Dans cette section, nous présentons deux approches visant à compacter la représentation des formules CNF et contraintes tables afin d'améliorer l'efficacité de la résolution.

2.10.1 Forme normale conjonctive

Réduire la taille des représentations CNF est une question cruciale dans l'efficacité de la résolution à l'heure des données volumineuses (Big Data). Les travaux visant à encoder de manière concise certaines contraintes spécifiques comme les contraintes de cardinalité, etc. ont contribué à réduire considérablement la taille des formules issues de transformations vers CNF et constituent une étape dans ce but. Les méthodes de simplification par application de la règle de résolution en constituent une autre [Eén & Biere 2005].

Dans [Jabbour *et al.* 2013f], nous avons montré comment la fouille de données peut être exploitée de manière originale et élégante pour compacter la représentation CNF des formules. Nous avons montré que les techniques d'extraction d'itemsets sont très appropriées pour découvrir des sous-clauses fréquentes dans les formules CNF. Plus précisément, une formule CNF Φ est codée sous la forme d'une table de transactions D contenant chacune les littéraux d'une clause. Les itemsets fréquents (fermés) de D correspondants dans ce cas aux sous-clauses qui se répètent dans Φ. Ces motifs sont ensuite utilisés pour réécrire la formule CNF de manière plus compacte grâce au principe d'extension de Tseitin [Tseitin 1968] par ajout de variables additionnelles comme illustré dans l'exemple 1.

Exemple 1. *La sous-clause $x_4 \vee x_5 \vee x_6$ est fréquente dans la formule Φ suivante. L'application du principe d'extension de Tseitein permet d'obtenir la nouvelle CNF Φ' par ajout d'une variable supplémentaire y.*

$$
\begin{aligned}
\Phi = \quad & \neg x_0 \vee x_2 \\
& \neg x_1 \vee x_4 \\
& \neg x_3 \vee x_4 \\
& \neg x_5 \vee x_6 \\
& \neg x_0 \vee x_1 \quad \vee \quad \mathbf{x_4 \vee x_5 \vee x_6} \\
& \qquad x_3 \quad \vee \quad \mathbf{x_4 \vee x_5 \vee x_6} \\
& \neg x_1 \vee x_2 \quad \vee \quad \mathbf{x_4 \vee x_5 \vee x_6} \\
& \neg x_2 \vee x_3 \quad \vee \quad \mathbf{x_4 \vee x_5 \vee x_6}
\end{aligned}
\qquad
\begin{aligned}
\Phi' = \quad & \neg x_0 \vee x_2 \\
& \neg x_1 \vee x_4 \\
& \neg x_3 \vee x_4 \\
& \neg x_5 \vee x_6 \\
& \neg x_0 \vee x_1 \quad \vee \quad \mathbf{y} \\
& \qquad x_3 \quad \vee \quad \mathbf{y} \\
& \neg x_1 \vee x_2 \quad \vee \quad \mathbf{y} \\
& \neg x_2 \vee x_3 \quad \vee \quad \mathbf{y} \\
& \qquad \neg \mathbf{y} \quad \vee \quad x_4 \vee x_5 \vee x_6
\end{aligned}
$$

La recherche de l'ordre de remplacement optimale peut être encodée en problème d'optimisation. Nous avons montré également que l'ensemble des clauses binaires, très présentes dans les transformations de problèmes d'application vers SAT, peut considérablement être réduit. Cette réduction est possible en considérant une représentation intermédiaire, sous forme d'implications, $((x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (x_1 \vee x_4) \wedge (x_1 \vee x_5)$ est équivalente à $x_1 \rightarrow x_2 \wedge x_3 \wedge x_4 \wedge x_5)$. Cette formulation permet de représenter les conséquences des implications sous forme de table de transactions et par conséquent de calquer le processus de la fouille des itemsets préalablement décrit pour compacter l'ensemble des clauses binaires. Nous avons également prouvé que notre approche peut réaliser des réductions similaires à celle de [Rintanen 2006]. C'est le cas pour un ensemble de clauses binaires formant un graphe biparti ou une clique. Cela offre une réelle alternative pour trouver des encodages réduits de certaines contraintes standards qui sont récurrentes lors des transformations vers SAT.

2.10.2 Contraintes tables

La programmation par contraintes (CP) est un paradigme puissant pour résoudre des problèmes combinatoires, basée sur un large éventail de méthodes de l'intelligence artificielle. Bien que ce paradigme appartient à la même classe de complexité que SAT (NP-Complet) et a motivé beaucoup de fertilisations allant de la traduction de problèmes de satisfaction de contraintes en SAT à l'hybridation d'approches, CP se distingue par un langage plus expressif. Un problème de satisfaction de contraintes (CSP) est un triple $\langle X, Dom, C\rangle$ où :

— $X = \langle X_1, X_2, \ldots, X_n\rangle$ est un n-tuple de variables,
— $Dom = \langle Dom(X_1), Dom(X_2), \ldots, Dom(X_n)\rangle$ est un n-tuple de domaines tel que $X_i \in Dom(X_i)$,
— $C = \{C_1, C_2, \ldots, C_k\}$ est un ensemble de contraintes où chaque contrainte C_i porte sur un sous-ensemble de X.

Les contraintes entre variables peuvent être exprimées soit en intension (contraintes globales e.g., contrainte de cardinalité, alldiff, etc.) ou en extension (contraintes tables).

Les contraintes tables sont considérées depuis longtemps comme particulièrement importantes dans les problèmes de satisfaction de contraintes. L'une des formulations de CSP les plus utilisées consiste à exprimer chaque contrainte en extension ou en tant que relation entre les variables avec domaines finis associés. Cependant, la taille de ce type de contraintes en extension pourrait être exponentiel dans le pire des cas. La complexité de l'espace de ce type de contraintes est un obstacle. Des tentatives de compression ont été proposées [Katsirelos & Walsh 2007] où une représentation alternative de l'ensemble des tuples d'une relation est représentée par un ensemble de tuples compressés. Cependant, les tuples compressés peuvent être plus grands que l'arité de la contrainte d'origine.

Dans [Jabbour *et al.* 2015a], nous avons proposé une approche de compression de contraintes tables qui combine les deux techniques d'extraction d'éléments fréquents et le principe d'extension de Tseitin pour dériver une nouvelle représentation compacte des contraintes tables. Nous avons montré que notre approche de compression de formules propositionnelles peut être parfaitement étendue et appliquée au cas des CSP en considérant le graphe de contraintes comme base de données, où les transactions correspondent aux contraintes et les items aux variables du CSP. Un itemset fréquent fermé correspond à un sous-ensemble de variables partagées le plus souvent par les différentes contraintes du CSP. Puis, en utilisant l'extension (variables auxiliaires), nous avons montré comment de telles contraintes peuvent être réécrites tout en préservant l'équivalence pour satisfiabilité. Dans [Jabbour *et al.* 2015a], nous avons également considéré chaque contrainte table individuellement donnant lieu à une nouvelle base de transactions formée d'une séquence de tuples, c'est-à-dire un ensemble de tuples indexés. Plus précisément, chaque valeur d'un tuple est indexé avec sa position dans la contrainte. En énumérant les itemsets fréquents fermés de cette base de transactions, nous sommes en mesure d'identifier le plus grand rectangle de la contrainte de la table. De même, avec le principe d'extension, nous avons montré comment une telle contrainte peut être représentée de manière plus succinte.

Chapitre 3

Fouille de données & Clustering

3.1 Introduction

L'analyse de données incluant l'extraction des connaissances à partir de données et la classification est à l'heure actuelle de domaine de recherche majeure accentué par la libéralisation de l'accès aux données qui rend ce domaine au centre de beaucoup de travaux de recherche.

L'extraction des motifs ensemblistes est l'une des tâches fondamentales de la fouille de données. Elle vise à découvrir des nouvelles relations intéressantes cachées dans de grandes bases de données. Depuis la première application, largement connue sous la dénomination de panier de la ménagère [Agrawal & Srikant 1994], plusieurs nouveaux domaines d'application ont été identifiés comme la bio-informatique, le diagnostic médical, la détection d'intrusion, la fouille du web et l'analyse des données scientifiques. Ce large spectre d'applications a permis à la fouille d'itemsets ou règles d'association d'être appliquée à une variété de bases de données, y compris les données séquentielles, spatiales, et les données des graphes.

D'autres part, l'extraction des motifs séquentiels est largement utilisée en bio-informatique comme un moyen d'extraire des connaissances significatives à partir d'un grand volume de données telles que la découverte de motifs dans les protéines, la prédiction de gènes et l'alignement de séquences. Ce problème est au cœur de nombreux autres domaines d'applications tels que les bases de données et l'exploration de texte. Elle constitue une tâche centrale dans la biologie calculatoire, l'analyse de séquence temporelle et l'exploration de texte.

Comme souligné dans [Raedt *et al.* 2008, Raedt *et al.* 2011], les contraintes font souvent partie de la spécification de plusieurs problèmes de fouille de données. Cette observation a conduit à un nouveau domaine de recherche actif et multidisciplinaire permettant une fertilisation croisée entre la fouille de données et l'intelligence artificielle [Raedt *et al.* 2008, Davidson *et al.* 2010, Dao *et al.* 2013, Kemmar *et al.* 2014, Gebser *et al.* 2016]. Plusieurs modèles de représentation et de résolution en intelligence artificielle ont été utilisés pour modéliser et résoudre plusieurs problèmes de fouille de données.

Dans la suite, nous présentons nos différentes approches pour la fouille des motifs ensemblistes et séquentiels.

3.2 Définitions et cadre formel

Soit Ω un ensemble fini non vide de symboles appelés *items*. À partir de maintenant, nous supposons que cet ensemble est fixe. Nous utilisons les lettres a, b, c, etc. pour désigner les éléments de Ω.

— Un *itemset* I sur Ω est défini comme un sous-ensemble de Ω, c'est-à-dire, $I \subseteq \Omega$. Nous utilisons 2^{Ω} pour désigner l'ensemble des itemsets sur Ω et les lettres majuscules I, J, K, etc. pour représenter les éléments de 2^{Ω}.
— Une *transaction* est une paire ordonnée (i, I) tel que i est une entier naturel, appelé *identifiant de transaction*, et I un itemset, i.e., $(i, I) \in \mathbb{N} \times 2^{\Omega}$.
— Une *base transactionnelle* D (ou table de transactions) est définie comme un ensemble fini de transactions non vides ($D \subseteq \mathbb{N} \times 2^{\Omega}$) où chaque identifiant de transaction fait référence à un ensemble d'éléments unique.

La figure 3.2 montre deux représentations de base d'une table de transactions. Dans celle de droite, les valeurs sont des booléens indiquant que l'item est présent (1) ou pas (0).

id	*transactions*						
1			*C*	*D*	*E*	*F*	*G*
2			*C*	*D*	*E*	*F*	*G*
3	*A*	*B*	*C*	*D*			
4	*A*	*B*	*C*	*D*		*F*	
5	*A*	*B*	*C*	*D*			
6			*C*		*E*		

id	*transactions*						
1	0	0	1	1	1	1	1
2	0	0	1	1	1	1	1
3	1	1	1	1	0	0	0
4	1	1	1	1	0	1	0
5	1	1	1	1	0	0	0
6	0	0	1	0	1	0	0

TABLE 3.1 – Deux représentations d'une table des transactions

Étant donnés une table de transactions D et un itemset I, la *couverture* de I dans D, notée $C(I, D)$, est définie par $C(I, D) = \{i \in \mathbb{N} \mid (i, J) \in D \text{ et } I \subseteq J\}$. Le *support* de I dans D, noté $Supp(I, D)$, corresponds à la cardinalité de $C(I, D)$, i.e., $Supp(I, D) = |C(I, D)|$.

Les itemsets fréquents peuvent être obtenus à partir de ceux dits fermés en énumérant tous les sous-ensembles.

Définition 2 (Itemset Fermé)**.** *Soit D une table de transactions. Un itemset $I \subseteq \Omega$ tel que $Supp(I, D) \geqslant 1$ est fermé si pour tout itemset J tel que $I \subset J$, $Supp(J, D) < Supp(I, D)$.*

La propriété la plus importante pour élaguer la recherche l'espace dans les algorithmes traditionnels est *l'anti-monotonicité* i.e., si I n'est pas un itemset fréquent alors tout itemset $I \subset J$ n'est pas fréquent également.

Le problème de fouille des itemsets fréquents consiste à énumérer, étant donné un seuil de support minimum *minupp*, l'ensemble $\{I \in \Omega \mid supp(I) \geq minsupp\}$. La figure 3.3.5 représente l'espace des $2^{|\Omega|}$ itemsets possible connu pour former un treillis.

Définition 3 (Règle d'Association)**.** *Une Règle d'Association est un motif de la forme $X \rightarrow Y$ tel que X (appelé antécédent) et Y (appelé conséquence) sont deux itemsets disjoints.*

En fouille de règles d'association, le prédicat d'intérêt est défini en utilisant les notions de support et de confiance. Le *support d'une règle d'association* $X \rightarrow Y$ dans une table de transactions D, est définit comme suit :

$$Supp(X \rightarrow Y, D) = \frac{Supp(X \cup Y, D)}{|D|}$$

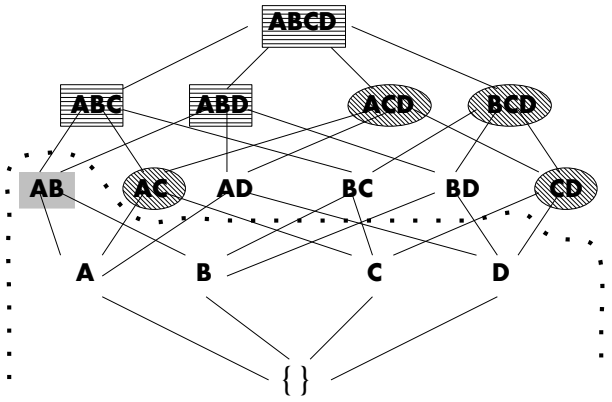


FIGURE 3.1 – Visualisation de l'espace des itemsets possibles

Clairement, le support d'une règle d'association détermine la fréquence d'occurrence de la règle.

On peut alors définir la *confidence* de $X \rightarrow Y$ dans D comme suit :

$$Conf(X \rightarrow Y, D) = \frac{Supp(X \cup Y, D)}{Supp(X, D)}$$

Intuitivement, la confidence fournit une estimation de la probabilité conditionnelle de Y étant donné X. Lorsqu'il n'y a pas d'ambiguïté, nous omettons de mentionner la table de transactions D, et nous notons simplement $Supp(X \rightarrow Y)$ et $Conf(X \rightarrow Y)$.

Une règle d'association *valide* est une règle d'association avec un support et une confiance supérieure ou égale au seuil de support minimum ($minsupp$) et au seuil de confiance minimum ($minconf$), respectivement.

Le problème de fouille des règles d'associton consiste à énumérer, étant donné un seuil de support minimum $minupp$ et un seuil de confidence $minconf$, l'ensemble $\{X \rightarrow Y \mid X,\ Y \in \Omega,\ supp(X \rightarrow Y) \geq minsupp,\ conf(X \rightarrow Y) \geq minconf\}$.

Définition 4 (Règle d'association fermée)**.** *Une règle $X \rightarrow Y$ est dite fermée ssi $X \rightarrow Y$ est une règle d'association valide et $X \cup Y$ est un itemset fermé.*

3.3 Fouille des itemsets

Dans cette section, nous présentons nos différents travaux réalisés dans le cadre de la résolution du problème de l'énumération des itemsets fréquents. Nous décrivons notre approche déclarative basée sur une formulation de ce problème en logique propositionnelle. Nous détaillons les différentes améliorations proposées ainsi que des extensions pour énumérer des motifs particuliers.

3.3.1 Approche symbolique

Dans [Jabbour *et al.* 2013d], nous avons proposé d'exprimer le problème de la fouille des itemsets en un problème d'énumération de modèles d'une formule propositionnelle. Cet encodage permet d'avoir une bijection entre les itemsets de la table de transactions et les modèles de la formule propositionnelle correspondante. Formellement, nous introduisons deux types de variables booléennes, celles qui représentent les items et celles qui représentent les transactions. Pour chaque item a (resp. transaction t_i) une variable propositionnelle, notée p_a (resp. q_i), est associée. Les itemsets recherchés correspondent aux modèles de la formule constituée des contraintes (3.1), (3.2) et (3.3).

La contrainte (3.1), décrite également dans [Henriques *et al.* 2012]), permet de capturer la table de transactions. Plus précisément, un itemset apparaît dans une transaction d'identifiant i ($q_i = vrai$) si cet itemset est formé d'un sous-ensemble des items de la transaction i. Ce qui peut être exprimé via une simple fonction booléenne (3.1). Pour exiger que l'itemset recherché apparaît dans au moins $minsupp$ transactions, nous utilisons la fameuse contrainte de cardinalité (3.2). Nous avons mentionné que cette contrainte est récurrente dans beaucoup de transformation de problèmes d'applications vers CP ou SAT. D'ailleurs c'est autour de la réflexion sur cette contrainte que l'encodage basé sur le principe des tiroirs a été proposé (section 2.8).

$$\bigwedge_{i=1}^{m}(\neg q_i \leftrightarrow \bigvee_{a\in\Omega\setminus T_i} p_a) \tag{3.1}$$

$$\sum_{i=1}^{m} q_i \geqslant minsupp \tag{3.2}$$

$$\bigwedge_{a\in\Omega}(\neg p_a \rightarrow \bigvee_{i\in[1..n]\ |\ a\notin T_i} q_i) \tag{3.3}$$

De plus, afin d'exprimer qu'un itemset est fermé, nous considérons la contrainte (3.3). En effet, pour un item a, si l'itemset recherché apparaît uniquement dans un sous-ensemble de transactions contenant a, alors cet itemset doit contenir a. Notons que la contrainte de fermeture exige l'ajout de $|\Omega|$ clauses, une clause pour chaque item.

L'encodage par formulation vers SAT présente un avantage majeur. En effet, la recherche de certains itemsets particuliers peut se traduire aisément par ajout de contraintes booléennes à la formulation de base contrairement aux approches spécialisés nécessitant un changement dans le programme lui même. En effet, l'énumération des itemsets de supports inférieurs à up nécessite l'ajout de la contrainte $\sum_{i=1}^{m} q_i \leq up$ ou encore pour restreindre l'énumération aux itemsets de tailles supérieurs à lb est obtenu en ajoutant $\sum_{a\in\Omega} p_a \geq lb$. Ce pouvoir d'expressivité permet de disposer d'une approche déclarative capable d'enrichir la formulation de base par de nouvelles contraintes utilisateurs dont l'intégration est assez simple.

La contrepartie de la déclarativité de notre approche réside dans la taille des formules obtenues. En effet, le nombre de clauses de la contrainte (3.1) est égale à $\sum_{i=1}^{m}(|\Omega| - |T_i| + 1)$. Or, très souvent, $|T_i| \ll |\Omega \setminus T_i|$ ce qui rend le nombre de clauses très proche de $|\Omega| \times |D|$. De plus la taille des clauses issues de la contrainte (3.3) est large. Tout cela constitue un vrai obstacle lorsque

l'on considère des données volumineuses. Pour remédier à ce problème, nous nous sommes intéressés à trouver des solutions prometteuses permettant de passer à l'échelle en s'appuyant sur la décomposition ou la parallélisme comme nous l'exposons dans la suite de ce manuscrit.

3.3.2 Solveur adaptatif

La traduction d'un problème dans un nouveau formalisme ne s'accompagne pas nécessairement par l'efficacité escomptée. En effet, certaines propriétés structurelles des problèmes initiaux ne sont pas facilement expressible dans la formulation. Dès lors, il est nécessaire de voir dans quelle mesure ces informations structurelles peuvent être prises en compte dans la conception d'un solveur dédié. Pour le problème de la fouille d'itemsets fréquents, plusieurs questions sont soulevées. L'heuristique VSIDIS est elle la plus appropriée pour ces problèmes ? l'apprentissage de clauses est il encore nécessaire ? Quelle polarité pour les variables ? Pour répondre à ces questions, nous avons mené une étude empirique [Dlala *et al.* 2016b] pour déterminer le paramétrage le plus approprié pour ce type de problèmes. Tout d'abord, le nombre de modèles est souvent assez élevé (des millions voir des dizaines de millions de modèles) pour les problèmes d'énumération. Par conséquent, nous avons constaté qu'un solveur CDCL utilisant simultanément l'apprentissage de clauses et les clauses bloquantes ralentit considérablement la recherche. Par contre une approche simple de type DPLL s'est montrée plus efficace. Concernant l'heuristique de choix de variables, nous avons comparé les performances de VSDIS [Zhang *et al.* 2001] à *MOMS* [Jeroslow & Wang 1990] qui procède en choisissant la variable la plus fréquente dans la CNF ce qui est équivalent à choisir l'item le moins fréquent dans la table de transactions. MOMS s'est distinguée en dominant VSIDS sur les problèmes considérés. Finalement, pour la polarité des variables, la conclusion est qu'affecter les variables à *vrai* permet d'obtenir les meilleurs performances. Toutefois dans le cas particulier de l'énumération des itemsets top-k, l'affectation à $faux$ est, par contre, plus efficace.

Un autre problème lié à l'approche déclarative est celui de la contrainte (3.2). Bien que des algorithmes de transformation de contraintes de cardinalité polynomiaux existent, la traduction en CNF de cette contrainte n'est pas recommandée. Nous avons pu constater que gérer cette contrainte par un simple propagateur permet de garantir des performances plus élevées au détriment probablement d'une légère perte en élagage de l'espace de recherche.

3.3.3 Motifs maximaux

Parmi les approches utilisées pour réduire la taille de la sortie pour l'extraction des itemsets fréquents, celles qui consistent à énumérer les itemsets fréquents fermés ont été largement étudiées permettant de réduire considérablement la taille de la sortie et par conséquent une meilleure exploration des données en pratique. Néanmoins, dans de nombreuses applications, notamment pour les données denses, la taille de l'ensemble des itemsets fermés [Zaki & Hsiao 2002] reste très élevé [Gouda & Zaki 2005]. Pour réduire encore plus la taille de la sortie, il est commode de considérer un sous-ensemble d'itemsets fermés dits *maximaux* qui sont une forme de condensation des itemsets fréquents bien qu'il ne permettent pas de retrouver les supports de chacun des itemsets. Un itemset fréquent est maximal s'il n'a pas de sur-ensemble qui est fréquent [Jr. 1998, Lin & Kedem 1998, Burdick *et al.* 2001, Zou *et al.* 2002, Gouda & Zaki 2005].

Les approches déclaratives peuvent s'adapter parfaitement à ce sous-problème. En effet, dans l'approche SAT, pour énumérer les itemsets maximaux, il suffit d'ajouter la contrainte (3.4) qui permet de forcer l'itemset à être maximale.

$$\bigwedge_{a \in \Omega} (\neg p_a \rightarrow \sum_{i \mid a \in T_i} q_i < minsupp) \tag{3.4}$$

Toutefois, cette contrainte concentre les problèmes récurrents liée à la transformation de problèmes issues d'applications vers SAT. En effet, elle est composée de plusieurs contraintes de cardinalité conditionnelle dont la transformation en CNF peut devenir rapidement fastidieuse. Dans [Dlala *et al.* 2018], nous avons présenté un nouvel algorithme *SATMAX* qui fait une utilisation originale des solveurs SAT pour énumérer efficacement tous les motifs maximaux d'une table transactions. Plus précisément, *SATMAX* utilise une recherche arborescente basée sur un solveur de type DPLL affectant les items en priorité à *vrai* afin de maximiser le taille de l'itemset recherché. Pour éviter d'énumérer les itemsets non-maximaux, une clause bloquante est ajoutée après chaque modèle trouvé, constituée des items affectés à $faux$, forçant les prochains itemsets à ne pas être sous-ensembles de ceux déjà trouvés. Des techniques de simplifications ont été également intégrées pour rendre la résolution plus rapide. Par ailleurs, nous avons utilisé la décomposition comme celle décrite dans la section 3.3.4 pour passer à l'échelle. Les premiers résultats obtenus sont très satisfaisants. En effet, pour certaines données et certaines valeurs du seuil du support minimum, notre approche obtient de meilleurs performances que *Eclat* [Borgelt 2012], un des meilleurs algorithmes spécialisés pour énumérer les itemsets fréquents maximaux.

3.3.4 Décomposition & approche parallèle

Si les approches déclaratives comme celles que nous avons proposé [Jabbour *et al.* 2013e, Boudane *et al.* 2016] permettent d'intégrer aisément des contraintes utilisateurs au problème initiale, la taille des problèmes transformés est toutefois un vrai obstacle quand à la scalabilité de ces approches. À titre d'exemple, pour la table *retail* [1] contenant 88128 transactions et 14245 items, le nombre de clauses de la contrainte (3.1) dépasse les 120 millions de clauses. S'il est prouvé que les solveurs CDCL peuvent résoudre des problèmes de grande taille, énumérer les modèles d'une formule de plusieurs dizaine de millions de clauses reste toutefois un palier difficile à franchir. Afin de permettre à notre approche déclarative de passer à l'échelle, nous avons proposé dans [Jabbour *et al.* 2015b] d'utiliser la décomposition comme mécanisme de division du problème de la fouille des itemsets fréquents. Notre approche consiste à ramener le problème de l'énumération des itemsets d'une table de transactions en l'énumération disjointes de plusieurs sous-table de petite taille en utilisant le paradigme *diviser pour régner*. L'idée principale est de décomposer la base de transactions en utilisant des chemins de guidage (voir figure 3.2) basées sur les items. Pour illustrer notre approche, considérons un item $a \in \Omega$. L'ensemble des itemsets fréquents peut être divisé en deux catégories : ceux contenant a (les modèles de $\Phi_D \wedge p_a$) et ceux où a n'apparait pas (les modèles de $\Phi_D \wedge \neg p_a$). Ceux qui contiennent a sont alors ceux de la table constituée des transactions contenant a alors qu'énumérer ceux qui ne contiennent pas a consiste à supprimer a de la table initiale. De plus, comme les sous-formules obtenues sont indépendantes,

1. http ://fimi.ua.ac.be/data/

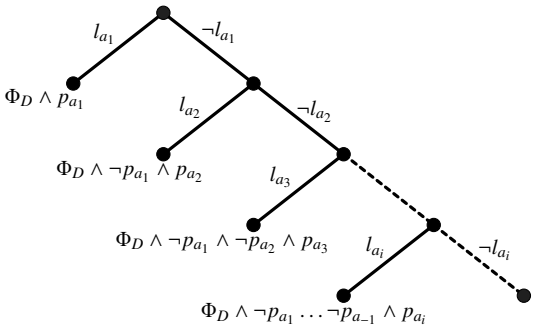


FIGURE 3.2 – Arbre des chemins de guidage

celles-ci peuvent être résolues en parallèle. Dans [Dlala *et al.* 2015, Dlala *et al.* 2010], nous avons conçu un solveur *paraSATMiner* [2] qui permet de lancer l'énumération des itemsets fréquents fermés en séquentiel ou en parallèle en utilisant la décomposition. Nous avons défini un nouveau problème permettant de trouver l'ordre de décomposition le plus efficace. Plus intéressant encore, cette nouvelle approche s'est montrée particulièrement plus efficace que les approches CP récentes [Lazaar *et al.* 2016, Schaus *et al.* 2017], proposant l'énumération des itemsets fréquents fermés en utilisant des contraintes globales, et permet de résoudre des problèmes de grande taille jusqu'au là inaccessibles.

3.3.5 Symétries et motifs ensemblistes

Les symétries sont principalement étudiées en fouille de graphes (e.g. [Huan *et al.* 2003, Desrosiers *et al.* 2007, Vanetik 2010]). Nous avons poursuivi nos travaux en fouille de données afin de trouver des fertilisations croisées entre la fouille de données et l'intelligence artificielle mais cette fois ci en exportant les techniques proposées pour l'élimination des symétries dans SAT vers la fouille des itemsets fréquents. La symétrie dans une table de transactions est définie comme une permutation entre les items de la table la laissant invariante. D'ailleurs, comme le problème des itemsets fréquents peut être exprimé par des contraintes, les symétries de la table de transactions ne sont autre que les symétries de la formule CNF associée.

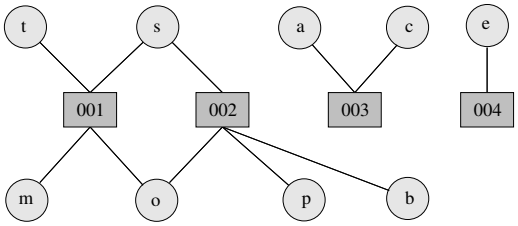
Pour découvrir les symétries d'une table de transactions, nous avons proposé de la transformer en un graphe. Les sommets représentent les items et les transactions et des arêtes lient tout sommet représentant un item aux transactions auxquelles il appartient. Les sommets des items sont différenciés de ceux des transactions dans le but d'éviter de générer des symétries entre un item et une transaction. La construction d'un tel graphe permet de profiter des nombreux algorithmes existants pour la découverte de symétries [McKay 1990, Ramani *et al.* 2006]. Par exemple, la permutation $\sigma = (b, m)(p, t)$ est une symétrie de la table D_1.

Afin d'exploiter les symétries en fouille de données, nous avons proposé deux approches différentes. La première [Jabbour *et al.* 2012] consiste à exploiter les symétries comme moyen de simplification de la table de transactions. En effet, notre approche consiste à exploiter les symétries pour réduire la table de transactions en supprimant certains items. À titre d'exemple si $\sigma = (a, b)$ est une symétrie binaire d'une table de transactions D, alors il est possible de supprimer

2. http ://www.cril.univ-artois.fr/decMining/

tid	itemset
001	*s*, *o*, *t*, *m*
002	*s*, *p*, *o*, *b*
003	*c*, *a*
004	*e*

Table transactionnelle D_1



Graphe associé à D_1

b de toute transaction ne contenant pas *a*. Ce mécanisme a été est généralisé pour une symétrie de taille quelconque.

Dans un autre travail de recherche [Jabbour *et al.* 2013a], nous avons intégré l'élimination des symétries dans un algorithme de type *a priori*. À chaque niveau, lors de la génération de candidats, si un itemset est fréquent tous ses symétriques le sont aussi et inversement. Par conséquent, l'utilisation des symétries permet d'éviter d'interroger la table de transactions pour certains itemsets candidats et par conséquent d'obtenir un gain en temps de traitement. Notons que contrairement à l'approche proposée dans [Jabbour *et al.* 2012] où la sortie contient un sous-ensemble des itemsets (un représentant pour chaque classe d'équivalence).

3.3.6 Modèles Top-k et motifs Top-k

Dans les problèmes issues du monde réel, comme c'est le cas de la fouille de données, un problème important concerne l'énorme taille de la sortie rendant difficile la tâche d'extraire des informations pertinentes. Par conséquent, il est essentiel de contrôler la taille de la sortie lors de l'exploration des masses de données. Restreindre la génération des itemsets à des formes condensées comme les fermés, les maximaux, sont quelques-unes des approches largement étudiées dans la littérature. La contrainte de support consistue l'autre manière de limiter la taille de la sorite. Cette contrainte permet à l'utilisateur de contrôler, au moins dans une certaine mesure, la taille de la sortie en ne considérant que les itemsets dont le support dépasse un seuil prédéfini. Cependant, il est difficile pour l'utilisateur de fixer un seuil approprié. Comme indiqué dans [Fu *et al.* 2000], un seuil trop petit peut conduire à la génération d'un grand nombre de motifs, alors qu'une valeur trop élevée mène à un nombre très réduit. Basé sur un ordre complet entre les ensembles d'items, dans [Fu *et al.* 2000], les auteurs ont proposé d'extraire les *k* itemsets les plus pertinents de longueur arbitraire. Dans [Han *et al.* 2002], la tâche proposée consiste à extraire des itemsets fréquents et fermés dits top-*k* de longueur supérieure à une borne inférieure *min*, où *k* est le nombre d'itemsets fermés fréquents à extraire, et *min* est la longueur minimale de chaque ensemble d'éléments. Il a été démontré qu'il est beaucoup plus facile de définir la longueur minimale des ensembles d'objets à extraire que de définir le seuil de support habituel.

Le cadre logique des top-*k* SAT défini dans [Jabbour *et al.* 2013e, Jabbour *et al.* 2017d] peut aisément être combiné avec l'approche déclarative basée sur la logique propositionnelle pour énumérer les itemsets top-*k*. Nous avons étudié dans [Jabbour *et al.* 2013e] une telle adaptation. Deux algorithmes sont introduits pour calculer les itemsets top-*k* fréquents où la relation de préférence est le support. La borne de la contrainte du support est alors modifiée dynamiquement pour éviter l'énumération des itemsets qui ne sont pas top-*k*. Ce cadre a été également étendu au cas des

motifs séquentiels [Jabbour *et al.* 2017d].

3.3.7 Motifs skyPatterns

Pour pouvoir extraire les informations les plus pertinentes, il est nécessaire tout d'abord d'inclure au maximum dans le formalisme utilisé toutes les connaissances utilisateurs. Leur absence rend l'extraction fastidieuse et coûteuse et ne permet pas de distinguer les connaissances relevantes qui sont noyés dans le reste des connaissances extraites. En pratique, les items et les transactions peuvent avoir plus au moins d'importance selon le problème considéré. Les tentatives faites consistent à pondérer chaque item en lui associant un poids [Tao *et al.* 2003, Cai *et al.* 1998, Yun & Leggett 2005, Tao *et al.* 2003]. De plus, un poids sur les transactions est généralement dérivé en agrégeant les poids des items de chacune des transactions. La motivation principale derrière ceci, est de définir un support pondéré permettant de satisfaire la propriété de l'anti-monotonie, propriété qui est généralement violée dans ce cadre. Cependant, dans les applications du monde réel, l'importance d'une transaction donnée n'est pas toujours corrélée aux poids de ces items. Par exemple, supposons que la base de données des transactions code un ensemble de documents publiés par les membres d'une institution de recherche dans les conférences internationales. Chaque publication associée à une conférence donnée (identifiant de transaction), implique un ensemble d'auteurs (articles). On peut associer un poids à chaque conférence en fonction de sa signification ou classement des conférences, tandis que les auteurs peuvent être pondérés en fonction de leurs âges, par exemple, les jeunes auteurs sont préférés. Dans ce contexte, un modèle intéressant peut être vu comme l'ensemble des jeunes auteurs publiant dans les conférences les plus sélectives.

Dans [Jabbour *et al.* 2016a], nous avons proposé un cadre logique traitant des préférences pour la fouille des itemsets fréquents prenant en compte les pondérations sur les items et les transactions. Ce modèle exploite la logique de pénalité [Walsh 2007, Brafman & Domshlak 2009, Kaci 2011, Domshlak *et al.* 2011] et les encodages basés sur SAT du problème d'extraction d'itemset. Les itemsets recherchés sont ceux *pareto dominant* ou *skypatterns* [Soulet *et al.* 2011].

3.3.8 Motifs ensemblistes sous incertitude

Jusqu'à présent, nous nous sommes intéressés à la fouille de données certaines c'est-à-dire que chaque élément apparaissant dans une transaction est connu avec certitude. Cependant, dans certaines applications du monde réel, les quantités massives de données sont souvent entachées d'imperfection et d'incertitude. Par exemple, les positions des objets obtenus par *GPS* ou *RFID*, les données collectées à partir de capteurs, et les calculs statistiques sont intrinsèquement bruyants. Par conséquent, le problème de trouver des itemsets sous incertitude a gagné un réel intérêt dans la communauté fouille de données. En effet, plusieurs approches et algorithmes ont été proposés [Chui *et al.* 2007, Chui & Kao 2008, Leung *et al.* 2008]. Le but des techniques mentionnées ci-dessus est de générer des modèles satisfaisant certaines contraintes spécifiques. En effet, de nombreuses propriétés (par exemple fréquence, maximalité, anti-monotonie) ont été étudiées et mises en œuvre dans différentes techniques de fouille de données.

En fouille d'itemsets sous incertitude, une table de transaction incertaine $D = \{T_1, T_2, ..., T_n\}$ est un ensemble de transactions où chaque item a_i d'une transaction j est associé avec une proba-

bilité $p(a_j, T_i) = p_{ji}$. La probabilité existentielle d'une itemset I dans T_i ($1 \leqslant i \leqslant n$), noté $p(I, T_i)$ est définie par $p(I, T_i) = \prod_{a_j \in I, I \subseteq T_i} p_{ji}$. Le support d'un itemset appelé *expected support* et noté ExpSN est défini $\text{ExpSN}(I, D) = \sum_{T_i \in D} p(I, T_i)$.

Le problème de la fouille des itemsets fréquents d'une table de transactions incertaine D consiste à énumérer l'ensemble $\{I \in \Omega \mid ExpSN(I, D) \geq minexpsupp\}$.

Dans [Dlala *et al.* 2016a], nous avons proposé une extension de notre cadre déclaratif du cas certain au cas incertain. Dans ce cas, la contrainte de l'expected support peut s'écrire comme suit :

$$\sum_{i=1}^{n} \prod_{a \in T_i} (p(a, T_i) \times p_a \wedge q_i + \neg p_a \wedge q_i) \geqslant minexpsupp \tag{3.5}$$

La contrainte (3.5) est particulière. Elle est non-linéaire et ses coefficients ne sont généralement pas entiers. Afin de tirer profit de cette contrainte, nous avons proposé une approche incrémentale énumérant les itemsets d'une taille fixée. L'objectif étant de rendre la contrainte de l'expected support linéaire pour une meilleure intégration dans les solveurs SAT. Notre approche permet de limiter le nombre des itemsets fréquents sans faux négatifs, i.e., les itemsets qui ne sont pas fréquents et que la relaxation n'a pas pu éliminer.

3.4 Fouille de règles d'association

Les résultats encourageants obtenus dans le contexte de la fouille des itemsets fréquents permettent d'étendre le cadre déclaratif aux règles d'association. À cela deux motivations : la première est que les techniques usuelles d'extraction de règles d'association suivent une approche en deux étapes. La seconde, est que la pertinence des règles à extraire est exprimée par des contraintes. Dans la suite nous présentons nos travaux dans ce cadre.

3.4.1 Approche symbolique

Dans [Boudane *et al.* 2016], nous avons proposé d'étendre l'encodage proposé pour les itemsets pour extraire les règles d'association. Cette extension est assez naturelle et obtenue en dupliquant la contrainte (3.1) pour capturer les deux itemsets X et Y de la règle $X \rightarrow Y$. Les contraintes (3.6),(3.7), (3.8), (3.9), (3.10) et (3.11) traduisent le problème de recherche des règles d'association fréquentes et fermées.

$$\bigwedge_{a \in \Omega} (\neg x_a \vee \neg y_a) \tag{3.6}$$

$$\bigwedge_{i=1}^{m} (\neg p_i \leftrightarrow \bigvee_{a \in \Omega \setminus T_i} x_a) \tag{3.7}$$

$$\bigwedge_{i=1}^{m} (\neg q_i \leftrightarrow \neg p_i \vee (\bigvee_{a \in \Omega \setminus T_i} y_a)) \tag{3.8}$$

$$\sum_{i=1}^{m} q_i \geqslant minsupp \tag{3.9}$$

$$\frac{\sum_{i=1}^{m} q_i}{\sum_{i=1}^{m} p_i} \geqslant minconf \tag{3.10}$$

$$\bigwedge_{a\in\Omega}(\bigwedge_{i=1}^{m}(q_i \rightarrow a \in T_i) \rightarrow x_a \vee y_a) \tag{3.11}$$

Notons que la contrainte (3.10) est réécrite sous la forme $\sum_{i=1}^{m} q_i - minconf \sum_{i=1}^{m} p_i \geq 0$. Souvent $minconf$ est exprimée en pourcentage i.e., $minconf = k\%$ (avec $k \in \mathbb{N}$) permettant de réécrire (3.10) sous forme de la contrainte pseudo-booléenne suivante $100 \sum_{i=1}^{m} q_i + k \sum_{i=1}^{m} \neg p_i \geq k \times m$.

En plus de la fouille des règles d'association standards, dans [Boudane *et al.* 2016], nous avons montré que d'autres type de règles peuvent être encodées efficacement. C'est le cas par exemple des règles indirectes [Tan *et al.* 2000] dont le but est d'extraire les paires d'items $\{a, b\}$ tel que $a \rightarrow b$ n'est pas une règle valide (rare) par contre les deux items sont impliqués séparément dans des règles d'association fréquentes avec la même conséquence, i.e., il existe un itemset X tel que $\{a\} \rightarrow X$ et $\{b\} \rightarrow X$ sont des règles valides.

L'avantage de la recherche en une seule phase s'est confirmé dans les résultats expérimentaux qui ont montré que notre approche peut être compétitive vis à vis des approches spécialisées [Fournier-Viger *et al.* 2014].

3.4.2 Règles non-redondantes

De façon similaire au problème de la fouille des itemsets fréquents, l'ensemble des règles d'association peut être également très large. La taille d'un tel ensemble de règles n'aide pas l'utilisateur à retrouver facilement les informations potentiellement pertinentes. Une telle observation a conduit à différentes approches visant à supprimer les règles jugées redondantes [Bastide *et al.* 2000, Fournier-Viger & Tseng 2013, Zaki 2004, Gasmi *et al.* 2005]. Les règles de type *conditions minimales conséquences maximales* (MNR) [Kryszkiewicz 1998b] (voir définition 5) sont celles qui couvrent toutes les règles d'association [Kryszkiewicz 1998a].

Définition 5 (Règle non redondante)**.** *$X \rightarrow Y$ est une règle non redondante minimale ssi il n'existe pas de règle $X' \rightarrow Y'$ différente de $X \rightarrow Y$ t.q. (i) $Supp(X \rightarrow Y) = Supp(X' \rightarrow Y')$, (ii) $Conf(X \rightarrow Y) = Conf(X' \rightarrow Y')$ et (iii) $X' \subseteq X$ et $Y \subseteq Y'$.*

Pour éliminer les règles redondantes, généralement les approches spécialisées partagent les deux étapes suivantes : (1) trouver l'ensemble des générateurs minimaux (voir définition 6) et l'ensemble des itemsets fermés, et (2) générer les règles non-redondantes en considérant les deux ensembles déjà extraits dans la première étape.

Définition 6 (Générateur minimal)**.** *Soit X un itemset fermé. Un itemset $X' \subseteq X$ est un générateur minimal de X ssi $Supp(X', D) = Supp(X, D)$ et il n'existe pas $X'' \subseteq X$ t.q. $X'' \subset X'$ et $Supp(X'', D) = Supp(X, D)$.*

Nous avons proposé dans [Boudane *et al.* 2017b] d'étendre le cadre déclaratif proposé dans [Boudane *et al.* 2016] pour extraire les règles minimales non redondantes en une seule étape. La redondance est éliminée élégamment par ajout de la contrainte (3.12).

$$\sum_{b\in\Omega} x_b \neq 1 \rightarrow (\bigwedge_{a\in\Omega} x_a \rightarrow \bigvee_{(T_i \mid a\notin T_i)} (\bigwedge_{b\notin T_i\cup\{a\}} \neg x_b)) \tag{3.12}$$

(3.12) est composée de plusieurs contraintes de cardinalité conditionnelles ($y \rightarrow \sum_{i=1}^{n} x_i \leq k$) (voir section 2.9). Par ailleurs, nous avons montré également qu'une restriction de notre encodage permet d'extraire les générateurs minimaux.

Nous avons exploré également une autre caractérisation des règles non-redondantes établi par Zaki et al. [Zaki 2004]. Il s'agit de règles d'association appelées *règles les plus générales* dont l'antécédent est le plus réduit et la conséquence (en termes d'inclusion) est une classe d'équivalente des règles ayant le même support et la même confidence.

3.5 Fouille de motifs séquentiels

Dans cette section, nous présentons un autre cadre déclaratif à base de contraintes pour extraire les motifs séquentiels. Une séquence de données peut être vue comme une séquence d'éléments, tandis que le motif peut être vu comme une sous-séquence pouvant contenir des caractères génériques ou des jokers en ce sens qu'ils correspondent à n'importe quel item [Parida *et al.* 2000, Pisanti *et al.* 2003, Arimura & Uno 2007].

3.5.1 Définitions et cadre formel

Soit Ω un ensemble fini d'items, appelé alphabet. Une *sequence d'items* s sur Ω est une simple séquence de symboles $s_0 \cdots s_{n-1}$ appartenant à Ω.

Nous notons par $|s|$ sa taille et par $\mathcal{P}_s$ l'ensemble $\{0, \ldots, |s| - 1\}$ de toutes les positions de ces symboles. *Un jocker* est un nouveau symbole $\circ$ qui n'est pas dans Ω. Ce symbole peut représenter n'importe quel symbole de l'alphabet.

Un *motif* sur Ω est une séquence $p = p_0 \ldots p_{m-1}$, telle que $p_0 \in \Omega$, $p_{m-1} \in \Omega$ et $p_i \in \Omega \cup \{\circ\}$ pour $i = 1, \ldots, m-2$.

Nous dirons que p est inclu dans $s = s_0 \ldots s_{n-1}$ à la position $l \in \mathcal{P}_s$, notée $p \sqsubseteq_l s$, si $\forall i \in \{0 \ldots m-1\}$, $p_i = s_{l+i}$ or $p_i = \circ$. Nous dirons également que p est inclut dans s, noté $p \sqsubseteq s$, si $\exists l \in \mathcal{P}_s$ tel que $p \sqsubseteq_l s$. La *couverture* de p dans s est définie comme l'ensemble $\mathcal{L}_s(p) = \{l \in \mathcal{P}_s | p \sqsubseteq_l s\}$. De plus, le *support* de p dans s est définit comme la valeur de $|\mathcal{L}_s(p)|$.

Définition 7 (Problème $\mathcal{FPS}$)**.** *Soit s une séquence, p un motif et $\lambda \geqslant 1$ un seuil minimum, appelé également quorum. Nous dirons que p est un motif fréquent dans s if $|\mathcal{L}_s(p)| \geqslant \lambda$.*

Le problème de la fouille des motifs fréquents dans une séquence d'items ($\mathcal{FPS}$) consiste à calculer l'ensemble $\mathcal{FPS}(s, \lambda)$ de tous les motifs fréquents.

Par exemple, considérons la séquence $s = aaccbcabcba$ et le motif $p = a \circ c$. Nous avons $\mathcal{L}_s(p) = \{0, 1, 6\}$, puisque $p \sqsubseteq_0 s$, $p \sqsubseteq_1 s$ et $p \sqsubseteq_6 s$. dans ce cas, si le seuil minimum est égale à 3, alors le motif p est un motif fréquent dans s.

Définition 8 (Motif fermé)**.** *Un motif fréquent p d'une séquence s est dit fermé si pour tout motif fréquent q satisfaisant $p \sqsubset q$, il n'existe pas d'entier d tel que $\mathcal{L}_s(q) = \mathcal{L}_s(p) + d$, avec $\mathcal{L}_s(p) + d = \{l + d \mid l \in \mathcal{L}_s(p)\}$.*

Clairement, l'ensemble des motifs fréquents fermés est une représentation condensée de l'ensemble des motifs fréquents. En effet, les motifs fréquents peuvent être obtenus à partir des fermés en remplaçant les éléments par des caractères génériques (jockers).

Comme pour les motifs ensemblistes, la propriété de l'anti-monotonie est importante. Elle stipule que si p_1 et p_2 sont deux motifs tel que $p_1 \sqsubseteq p_2$, alors si $|\mathcal{L}_s(p_2)| \geqslant \lambda$ alors $|\mathcal{L}_s(p_1)| \geqslant \lambda$.

3.5.2 Approche symbolique

Nous avons proposé deux approches pour la fouille des motifs séquentiels [Coquery *et al.* 2012, Jabbour *et al.* 2013c]. Nous décrivons ici l'encodage SAT pour le problème d'extraction de motifs séquentiels fréquents avec des caractères génériques dans le cas d'une séquences d'items proposée dans [Jabbour *et al.* 2013c]. À cette fin, nous fixons, sans perte de généralité, un alphabet $\Omega = \{a_1, \ldots, a_k\}$, une séquence $s = s_0, \ldots, s_{m-1}$ sur Ω et un seuil minimum $n \geqslant 1$. Nous associons à chaque item a apparaissant dans s un ensemble distinct de variables propositionnelles k_a, notées $p_{a,0}, \ldots, p_{a,k_a-1}$ où $k_a = max(\mathcal{L}_s(a)) + 1$. La variable $p_{a,l}$ est utilisée pour représenter le fait que l'élément a est à la position l dans le modèle candidat. De plus, nous utilisons $k_a = max(\mathcal{L}_s(a)) + 1$ variables propositionnelles pour chaque item a car l'ensemble $\{0, \ldots, max(\mathcal{L}_s(a))\}$ représente toutes les positions possibles de a. Clairement, la dernière position de a dans chaque modèle candidat ne peut pas être supérieur à sa dernière position dans la séquence s. De plus, nous associons pour chaque position possible l dans $0..m-1$ du modèle candidat une variable propositionnelle distincte, notée q_l.

Nous avons proposé plusieurs contraintes visant à capturer le motif recherché. (3.13) permet de forcer le modèle candidat à commencer par un item non générique. Celle de (3.14) permet de capturer les positions du motif candidat. elle est obtenue en exprimant que les valeurs de vérité des variables représentant les positions possibles du motif candidat (les variables de la forme q_l) sont déterminées à partir de celles des variables représentant les items et leurs positions possibles (les variables de la forme $p_{a,l}$)

$$\bigvee_{a \in \Omega} p_{a,0} \tag{3.13}$$

$$\bigwedge_{0 \leqslant l \leqslant m-1} \Big(\bigvee_{a \in \Omega, 0 \leqslant i \leqslant k_a - 1} (p_{a,i} \wedge (l + i > m - 1 \vee s_{l+i} \neq a)) \Big) \leftrightarrow \neg q_l \tag{3.14}$$

Cette formule exprime le fait que le modèle candidat n'est pas à la position l si et seulement s'il existe un élément a appartenant à ce modèle à la position i tel qu'il n'est pas à la position $l+i$ dans la séquence. Par exemple, lorsque le modèle candidat commence par a et que cet élément n'est pas à la position l dans la séquence, nous savons que l n'est pas une position pour le motif candidat. Notons que pour un modèle $\mathcal{B}$ de (3.14) et une position l dans son modèle correspondant, si nous avons $\mathcal{B}(p_{a,l}) = 0$ pour chaque item a avec $l \leqslant k_a - 1$, alors cela signifie que le motif contient un joker à la position l.

Étant donnée un modèle $\mathcal{B}$ de (3.13) $\wedge$ (3.14) tel qu'il existe $0 \leqslant l_0 \leqslant m-1$ avec $\mathcal{B}(q_{l_0}) = 1$, nous utilisons $P_{\mathcal{B}}$ pour représenter le motif candidat associé à $\mathcal{B}$.

Exemple 2. *Considérons la séquence s = abaca construite sur $\Omega = \{a, b, c\}$. L'ensemble des variables associées à a, b, c sont $\{p_{a,0}, p_{a,1}, p_{a,2}, p_{a,3}, p_{a,4}\}$, $\{p_{b,0}, p_{b,1}\}$, et $\{p_{c,0}, p_{c,1}, p_{c,2}, p_{c,3}\}$ respectivement. La formule (3.13) $\wedge$ (3.14) dans ce cas est :*

$$(p_{a,0} \vee p_{b,0} \vee p_{c,0}) \wedge$$

$$((p_{b,0} \vee p_{c,0} \vee p_{a,1} \vee p_{c,1} \vee p_{c,2} \vee p_{a,3}) \leftrightarrow \neg q_0) \wedge$$
$$((p_{a,0} \vee p_{c,0} \vee p_{b,1} \vee p_{c,1} \vee p_{a,2} \vee p_{c,3} \vee p_{a,4}) \leftrightarrow \neg q_1) \wedge$$
$$((p_{b,0} \vee p_{c,0} \vee p_{a,1} \vee p_{b,1} \vee p_{c,2} \vee p_{a,3} \vee p_{c,3} \vee p_{a,4}) \leftrightarrow \neg q_2) \wedge$$
$$((p_{a,0} \vee p_{b,0} \vee p_{c,1} \vee p_{b,1} \vee p_{a,2} \vee p_{c,2} \vee p_{a,3} \vee p_{c,3} \vee p_{a,4}) \leftrightarrow \neg q_3) \wedge$$
$$((p_{b,0} \vee p_{c,0} \vee p_{a,1} \vee p_{b,1} \vee p_{c,1} \vee p_{a,2} \vee p_{c,2} \vee p_{a,3} \vee p_{c,3} \vee p_{a,4}) \leftrightarrow \neg q_4)$$

Soit $\mathcal{B}$ une interprétation booléenne sur la formule définie comme suit :

- $\mathcal{B}(p_{a,0}) = 1$ *and* $\mathcal{B}(p_{b,0}) = \mathcal{B}(p_{c,0}) = 0$*;*
- $\mathcal{B}(p_{a,1}) = \mathcal{B}(p_{b,1}) = \mathcal{B}(p_{c,1}) = 0$*;*
- $\mathcal{B}(p_{a,2}) = 1$ *and* $\mathcal{B}(p_{c,2}) = 0$*;*
- $\mathcal{B}(p_{a,3}) = \mathcal{B}(p_{c,3}) = 0$*;*
- $\mathcal{B}(p_{a,4}) = 0$*;*
- $\mathcal{B}(q_0) = \mathcal{B}(q_2) = 1$ *and* $\mathcal{B}(q_1) = \mathcal{B}(q_3) = \mathcal{B}(q_4) = 0$.

On peut aisément remarquer que $\mathcal{B}$ est un modèle de la formule précédente. De plus, le motif $P_{\mathcal{B}}$ corresponds à $a \circ a$.

Afin d'énumérer les motifs avec un support supérieur ou égale au seuil de support minimal $minsupp$, il suffit d'ajouter la contrainte de cardinalité suivante :

$$\sum_{l=0}^{m-1} q_l \geqslant minsupp \tag{3.15}$$

Pour énumération des motifs fréquents fermés, nous avons introduit une condition nécessaire, mais pas suffisante. L'idée de base consiste à maximiser le nombre d'éléments (différents du jocker) du côté droit du motif candidat.

$$\bigwedge_{a\in\Omega, 0\leqslant i\leqslant k_a-1} \Big(\bigwedge_{0\leqslant l\leqslant m-1} q_l \rightarrow s_{l+i} = a \Big) \rightarrow p_{a,i} \tag{3.16}$$

En d'autres termes, cette formule exprime le fait qu'un caractère générique est utilisé sur le côté droit du modèle candidat uniquement lorsque l'utilisation d'un item à la même position réduit le nombre de positions de ce motif. Par exemple, considérons la séquence *abcabc* et le motif $a \circ c$. L'interprétation booléenne qui correspond à ce modèle ne satisfait pas la formule (3.16) puisque nous pouvons remplacer $\circ$ par b sans réduire le nombre de positions ($\{0,2\}$).

De la même manière que la formule (3.16), la formule suivante permet de maximiser le nombre d'éléments sur le côté gauche du motif candidat :

$$\bigwedge_{a\in\Omega, 1\leqslant i\leqslant k'_a-1} \neg\Big(\bigwedge_{0\leqslant l\leqslant m-1} q_l \rightarrow s_{l-i} = a \Big) \tag{3.17}$$

où $k'_a = n - min(\mathcal{L}_s(a))$. Les entiers de 1 à $k'_a - 1$ représentent toutes les positions possibles de l'item a sur le côté gauche du motif candidat. La formule (3.17) spécifie qu'aucun motif n'est accepté si nous pouvons ajouter un élément sur son côté gauche et conserver le même nombre de positions.

3.6 Clustering de formules booléennes

Les techniques de classification sont considérées comme une étape importante de l'analyse de données. On distingue deux types de classification : la classification est dite *supervisée* si on connaît à priori les descriptions des classes qu'on veut. Dans l'autre cas, celle dite *non-supervisée* ou *clustering* c'est à dire lorsque les classes sont construites à partir des caractéristiques des objets et dont le but est de regrouper des objets dans des catégories ou des classes de sorte que ceux qui appartiennent au même groupe ont des propriétés similaires en se basant sur des critères bien définis.

Nous nous sommes intéressés au clustering, une technique visant à partitionner un ensemble d'objets en sous-ensembles homogènes et bien séparés appelés clusters. L'homogénéité signifie que les objets dans un même cluster sont très similaire, par contre la séparation exprime qu'un objet dans un cluster doit être différent des objets des autres clusters. Le problème de clustering a été largement étudié pour son application dans plusieurs domaines, à savoir la biologie [Wittkop *et al.* 2010], l'analyse des images [Matas & Kittler 1995] et dans le commerce [Wang *et al.* 1999], etc. Les données peuvent être transactionnelles, séquentielles, arborescentes, des graphes, textes, ou même de nature symbolique. Ce dernier type de données est particulièrement adapté à la modélisation d'objets complexes et hétérogènes généralement décrits par un ensemble de variables de différents types e.g., intervalles, multi-catégorielles ou modaleds [Billard & Diday 2012, Diday & Esposito 2003]. Nous pouvons également citer le regroupement conceptuel, proposé il y a plus de trente ans dans [Michalski 1980], défini comme une tâche d'apprentissage automatique qui considère un ensemble de descriptions d'objets (événements, faits, observations, etc.) et produit un système de classification sur eux. Le regroupement conceptuel ne se limite pas à partitionner les données, mais génère également des clusters qui peuvent être résumés par une description conceptuelle.

Aujourd'hui, les données sont de nature complexe et hétérogène. Ces données complexes peuvent représenter les désirs des clients ou les préférences recueillies par différentes façons possibles issues de sondages ou de questionnaires. Par exemple, on peut citer les systèmes de configuration généralement conçus pour fournir des produits personnalisés répondant aux différentes exigences du client habituellement modélisé par des contraintes ou des formules logiques [Hotz *et al.* 2014].

Dans [Boudane *et al.* 2017a], nous avons introduit un nouveau cadre de clustering, dans lequel les objets complexes sont décrits par des formules booléennes. Cela est tout à fait plausible. Considérons le cadre d'un sondage où les questions posées sont ordonnées et certaines questions dépendent des réponses des précédentes questions. Cela peut parfaitement être formulé en logique propositionnelle.

La question que nous avons considéré est comment partitionner un ensemble des formules propositionnelles S. Ce problème qu'on note $\mathcal{P}(k, S)$ est défini formellement comme suit.

Définition 9 (Solution de $\mathcal{P}(k, S)$)**.** *Soit $S = \{\Phi_1, \ldots, \Phi_n\}$ un ensemble de formules propositionnelle. C est une solution de $\mathcal{P}(k, S)$ ssi C est une partition de S de taille k et pour tout $C_i \in C$, $C_i \nvdash \bot$.*

Notons que la contrainte $C_i \nvdash \bot$, dite contrainte de cohérence, exige la cohérence logique de chaque cluster, i.e., si $C_i = \{\Phi^i_1, \ldots, \Phi^i_m\}$, alors $\Phi^i_1 \wedge \ldots \wedge \Phi^i_m \nvdash \bot$.

Les algorithmes de l'état de l'art de clustering peuvent être adaptés au cas des objets de type formules propositionnelles. Néanmoins, cette adaptation requiert de définir une distance (ou similarité) entre deux formules, ce qu'est un centroïde d'un cluster de formule, et une fonction objective à optimiser.

Pour définir une distance entre formules, nous avons considéré le cœfficient de similarité de Jaccard, issue de la mesure de similiarité de Tversky [Tversky 1977], et définit comme ci-dessous. En effet, une formule booléenne peut être vue comme l'ensemble de ces modèles (forme normale disjonctive DNF).

$$s_J(\Phi_1, \Phi_2) = \frac{|M(\Phi_1 \wedge \Phi_2)|}{|M(\Phi_1 \vee \Phi_2)|}$$

À partir de ce cœfficient, on peut définir la mesure de distance $d_J(\Phi_1, \Phi_2) = 1 - s_J(\Phi_1, \Phi_2)$.

Intuitivement, étant donné un cluster C_i (sous-ensemble de formules), le centroïde peut être défini comme la conjonction de ses formules $O_{C_i} = \bigwedge_{\Phi \in C_i} \Phi$. Cette formulation peut être améliorée en considérant la *distance de hamming* avec les modèles qui ne sont pas dans l'intersection. La formulation de la fonction objective est similaire à celle définie de manière standard suivante :

$$C^* = \underset{C}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{C_i \in C} Q(C_i)$$

Suivant la définition de la qualité du cluster, plusieurs fonctions peuvent être instanciées. $Q(c)$ peut être substituée par la somme des distances au centroïde (distance euclidienne) $\sum_{\Phi \in C_i} d_J(\Phi, O_{C_i})$. Dans [Boudane *et al.* 2017a], nous avons considéré celle de la définition 10.

Définition 10. *Soit $C = \{\Phi_1, \ldots, \Phi_n\}$ un cluster de formules booléennes. Nous définissons la qualité de C comme :*

$$Q(C) = \frac{|\mathcal{M}(\Phi_1 \wedge \cdots \wedge \Phi_n)|}{|\mathcal{M}(\Phi_1 \vee \cdots \vee \Phi_n)|}$$

Dans [Boudane *et al.* 2017a], nous avons montré que les adaptions proposées des algorithmes de l'état de l'art du clustering comme le *k-means* [Macqueen 1967], les *algorithmes hiérarchiques* [Sneath & Sokal 1973, Karypis *et al.* 1999] et les algorithmes basés sur la densité [Ester *et al.* 1996] peuvent échouer à fournir un regroupement adéquat en satisfaisant le nombre de clusters fixé à cause de la contrainte de cohérence des clusters. Afin de pallier ce problème, nous avons introduit un nouvel algorithme hiérarchique descendant pour une représentation ensembliste des formules. Plus précisément, notre algorithme procède par une phase de division en se basant sur la notion d'ensemble intersectant [3] (en anglais Hitting Set) de l'hypergraphe dont les sommets est l'union des modèles des formules et les hyper-arêtes sont les modèles de chaque formule. Le calcul du premier ensemble intersectant minimum H^0 permet de disposer des premires centroïdes. Cela permet de former les clusters de formules en attribuant chaque formule à un cluster si l'un de ces modèles est un centroïde. D'ailleurs, $|H^0|$ constitue la borne inférieure du clustering qu'il est possible de réaliser sous la contrainte de cohérence des clusters. Si $|H^0| > k$, alors la division est réitérée en choisissant le cluster de mauvaise qualité C_m (voir Définition 10) et en remplaçant C

3. Un sous-ensemble H est dit ensemble intersectant d'un ensemble d'ensembles F si $\forall E \in F, H \cap E \neq \emptyset$. De plus, H est dit irréductible (ou minimal) si tout sous-ensemble strict de H n'est pas un ensemble intersectant

par $C = \bigcup_{\Phi \in C_m} (\Phi \setminus (\bigcap_{\Phi' \in C_m} \Phi'))$ et $k = k - |H^0|$ où les modèles communes aux formules du cluster sont préalablement supprimés avant la division, jusqu'à l'obtention des k clusters demandés.

Le calcul d'ensemble intersectant minimal est connu pour être un problème NP-difficile et l'évaluation de la qualité des clusters fait intervenir l'appel à un oracle #SAT appartenant à la classe #P-Complet lorsque la forme normale disjonctive des formules est considérée. Cela constitue deux problèmes de complexité élevée. Pour réduire la complexité calculatoire et éviter principalement l'appel à l'oracle #SAT, nous avons proposé une formulation en MaxSAT incrémentale permettant de réaliser un tel clustering.

Chapitre 4

Graphes

4.1 Introduction

Les graphes constituent un outil général et puissant de modélisation des relations structurelles entre les données. Ils ont été largement utilisé dans plusieurs domaines d'application, incluant les interactions en bio-informatique, l'analyse de liens dans les réseaux, les documents XML, etc. L'importance prise par les réseaux sociaux depuis quelques années a rendu ce domaine de plus en plus attractif et incontournable et remis au goût du jour certaines questions comme l'étude des structures topologiques des graphes, la détection de communautés et regroupement [Newman & Girvan 2004, Xie *et al.* 2011, Gregory 2009, Shen *et al.* 2009, Yang & Leskovec 2013] mais dans un environnement où les données sont volumineuses nécessitant des approches pour compacter ces représentations en vue de leur traitement.

Généralement, les réseaux réels impliquent des concentrations élevées d'arêtes dans des groupes spéciaux de sommets, et de faibles concentrations entre ces groupes. À titre d'exemple, dans le graphe de la toile mondiale (www), des concentrations denses peuvent correspondre à des groupes de pages traitant de sujets identiques ou connexes. En raison de leur degré de connectivité, ces communautés/clusters peuvent parfois correspondre à certaines classes de graphes tels que les (quasi)-cliques, les graphes bipartis, etc. Cependant, l'extraction de structures dans les graphes des bases de données sont coûteuses en raison de la complexité des graphes de données de plus en plus volumineux. Par conséquent, il est difficile de comprendre l'information/structure codée dans ces graphes par de simples visualisations.

Dans la suite nous tentons à travers les travaux effectués dans le cadre de la détection de communautés dans les graphes et l'approche de compression pour une représentation plus concise de ces derniers de comprendre les structures des réseaux complexes.

4.1.1 Définitions et notations

Nous considérons un réseau comme un graphe non orienté $G = (V, E)$ constitué d'un ensemble de sommets V et d'un ensemble d'arêtes $E \subseteq V \times V$ qui lient des paires de sommets. Pour de raisons de simplicité, n désigne le nombre de sommets dans G (c'est-à-dire $n = |V|$) et m le nombre d'arêtes de G (c'est-à-dire $m = |E|$). Le degré d'un sommet $u \in V$, désigné par $d(u)$, est défini comme étant le nombre d'arêtes incidentes. Pour deux sommets $u, v \in V$, on note par $dist(u, v)$ la distance entre u et v. Étant donné une arête $e = (u, v) \in E$, la distance entre e et un sommet $w \in V$ est définie comme étant $dist(e, w) = min\{dist(u, w), dist(v, w)\}$. Si pour tout $u, v \in V$, $(u, v) \in E$, alors G représente une clique. Si $V = X \cup Y$ tel que $X \cap Y = \emptyset$ et $X \times X \cap E = \emptyset$ et $Y \times Y \cap E = \emptyset$, alors G est dit biparti. De plus, si $X \times Y = E$, alors G est dit biparti complet. Nous notons un graphe biparti par $G(X \cup Y, E)$.

En théorie des graphes, une communauté est définit comme un *groupe de sommets plus étroitement connectés les uns avec les autres qu'avec le reste du réseau*. Dans les graphes réels, les sommets s'organisent en groupes étroitement liés, appelés communément *communautés*, clusters ou modules. Souvent, certaines communautés dans les réseaux peuvent se chevaucher car les sommets peuvent appartenir à plusieurs communautés à la fois.

Dans la littérature, différentes mesures ont été proposées pour quantifier la qualité des communautés. Parmi celles-ci, on peut citer la *modularité* [Newman & Girvan 2004] et la fonction F_1 *score*.

Modularité : Étant donné un découpage en communautés disjointes ou recouvrantes, la modularité permet d'évaluer leurs qualités par le calcul de la différence entre la proportion de liens se trouvant à l'intérieur des communautés et la proportion de liens qui pourraîent être à l'intérieur de ces mêmes communautés, en moyenne, dans un graphe aléatoire de même distribution de degrés. Cette fonction est définie comme suit :

$$Q = \frac{1}{2m} \sum_{C \in C_G} \sum_{i,j \in C} \frac{1}{O_i O_j} \left[A_{ij} - \frac{d_i d_j}{2m} \right] \tag{4.1}$$

où C_G est l'ensemble de communautés dans le graphe G ; O_u est le nombre de communautés auxquelles appartient le sommet u ; et d_u est le degré du sommet u.

F_1 score : Soient $\hat{C}$ (resp. C^*) l'ensemble de communautés détectées (resp. réelles) associées au graphe G. La fonction F_1 score vise à quantifier le niveau de correspondance entre C^* et $\hat{C}$ [Yang & Leskovec 2013]. Autrement dit, nous devons déterminer quelle communauté $C_i \in C^*$ correspond à $\hat{C}_i \in \hat{C}$. Ensuite, le score F_1 est défini comme la moyenne de la communauté réelle la plus proche de chaque communauté détectée et la meilleure communauté détectée correspondant à chaque communauté réelle. Formellement, cette fonction est définie de la manière suivante :

$$\frac{1}{2} \left(\frac{1}{|C^*|} \sum_{C_i \in C^*} F_1(C_i, \hat{C}_{g(i)}) + \frac{1}{|\hat{C}|} \sum_{\hat{C}_i \in \hat{C}} F_1(C_{g'(i)}, \hat{C}_i) \right) \tag{4.2}$$

où la meilleure correspondance entre g et g' est définie comme suit :

$$g(i) = \arg\max_j F_1(C_i, \hat{C}_j), \quad g'(i) = \arg\max_j F_1(C_j, \hat{C}_i)$$

et $F_1(C_i, \hat{C}_j)$ est la moyenne harmonique de la précision [1] et le rappel [2].

4.2 Détection de communautés

La découverte des structures cachées d'un graphe est un problème fondamental dans l'analyse de grands graphes de données. Dans la litérature, plusieurs approches ont été proposées pour résoudre ce problème difficile. De plus, la détection de communautés dans un graphe est une méthode efficace pour comprendre les caractéristiques des réseaux de données. Elle constitue une étape cruciale dans l'étude de la structure et la dynamique des réseaux sociaux et biologiques

1. La précision entre deux communautés C et C' est définie comme suit : $\frac{|C \cap C'|}{|C|}$.
2. Le rappel entre deux communautés C et C' est définie comme suit : $\frac{|C \cap C'|}{|C'|}$.

[Newman 2010, Adamcsek *et al.* 2006]. Par exemple, la détection de communautés permet de mieux comprendre les mécanismes métaboliques et les interactions protéine-protéine (IPP), les réseaux trophiques écologiques, les réseaux sociaux, les réseaux de collaboration, les réseaux d'information de documents interconnectés, et même les réseaux de produits co-achetésc. Dans la suite, nous présentons nos différents travaux effectués dans cette thématique de recherche.

4.2.1 Communauté k-liée centrée

Dans [Jabbour *et al.* 2017c], nous avons proposé une nouvelle approche de détection de communautés avec chevauchement dans les grands réseaux en se basant sur la logique propositionnelle. Utiliser la satisfiabilité pour étudier les réseaux complexes met en lumière une nouvelle perspective pour combiner les progrès dans la résolution du problème SAT et la détection des communautés. Notre approche consiste à décomposer un graphe $G(V, E)$ donné en communautés tout en fixant le diamètre de chaque communauté comme c'est le cas dans [Alba 1973]. Pour ce faire, nous avons défini la notion de communauté *k-liée centrée* associée à un sommet u comme l'ensemble de sommets qui sont à une distance au plus k de u, i.e., $C(u, k) = \{v \in V \mid dist(u, v) \leq k\}$. Cela constitue une manière particulière de fixer le diamètre de la communauté. Dès lors, nous avons formulé le problème de détection de communautés en un problème de recherche d'une décomposition minimale ou maximale en groupes k-liées. En d'autre termes, nous recherchons un sous-ensemble minimal ou maximal $S \subseteq V$ de sommets centraux tels que tout autre sommet est à une distance au plus k d'un des sommets de S.

$$min/max\ \{|S|,\ S \subseteq V \text{ et } f(S) = V\} \qquad f(S) = \bigcup_{u \in S} C(u, k) \tag{4.3}$$

Ensuite, nous avons exprimé une variante de (4.3) en un problème d'optimisation MaxSAT dont la formulation est la suivante :

$$\bigwedge_{u \in V} (x_u \rightarrow \bigwedge_{v \in C(u, \frac{k}{2})} \neg x_v) \tag{4.4}$$

$$\bigwedge_{u \in V} \bigvee_{v \in C(u, \frac{k}{2})} x_v \tag{4.5}$$

$$\min / \max \sum_{u \in V} x_u \qquad \text{subject to } (4.4) \wedge (4.5) \tag{4.6}$$

Des évaluations empiriques ont permis de conclure que la valeur $k = 4$ est la plus appropriée, permettant de maximiser les valeurs de modularité et de F_1 score. Nous avons également proposé une nouvelle méthode à base de préférences pour étendre le modèle défini dans [Jabbour *et al.* 2017c]. L'idée consiste à prendre en considération l'importance de chaque sommet par rapport à son voisinage en donnant plus de poids aux sommets dont le voisinage est plus connecté. Plusieurs notions existent pour caractériser l'importance d'un sommet dans un graphe. Parmi lesquelles, on trouve le degré de centralité, la densité, etc. Notre approche consiste à prendre en compte les préférences dans le modèle [Jabbour *et al.* 2017c] afin de sélectionner un centroïde parmi les sommets candidats. Pour cela, le score de chaque sommet u est calculé en fonction de

la densité de la communauté C_u où u est le centroïde de C_u. Formellement, la densité de u est calculée par la fonction suivante :

$$g(C_u) = \frac{2 \times m_{C_u}}{n_{C_u}(n_{C_u} - 1)}$$

où m_{C_u} est le nombre d'arêtes entre les sommets de C_u et $n_{C_u} = |C_u|$.

Ensuite, le sommet u est un centroïde si $g(C_u)$ est maximale. Le modèle utilise une nouvelle fonction objective qui intègre des poids dans le premier modèle [Jabbour *et al.* 2017c]. Cette fonction est exprimée de la manière suivante :

$$\min / \max \sum_{u \in V} w_u \times x_u \qquad \text{subject to (4.4)} \wedge \text{(4.5)} \tag{4.7}$$

4.2.2 Communauté k-clique-star

L'identification des communautés dans un réseau social peut être considérée comme un problème de densité des sous-graphes. Dans ce contexte, différents modèles mathématiques ont été étudiés pour l'extraction des communautés dans les réseaux du monde réel. Le modèle le plus intuitif pour l'analyse de réseaux sociaux est sans aucun doute la clique [Luce 1950]. Une telle structure, de préférence clique maximale, constitue la structure communautaire idéale, que l'on voudrait trouver. Cependant, générer des communautés avec une telle propriété structurelle est intraitable. De plus, la clique est trop restrictive. En effet, les cliques de petite taille sont les plus fréquentes dans les réseaux du monde réel, tandis que des cliques de grandes tailles sont généralement rares [Cohen 2008]. Autrement dit, les graphes qui apparaissent dans de nombreux réseaux réels ont plutôt une topologie de "petit monde" dans laquelle les sommets sont fortement groupés avec une longueur du chemin entre eux très faible [Wasserman & Faust 1994]. Par conséquent, d'autres formes plus souples de sous-graphes cohésifs ont été proposées. Leskovec a présenté un modèle basé sur la distance appelé k-clique [Leskovec *et al.* 2010], et Alba a proposé un modèle basé sur le diamètre appelé k-club [Alba 1973]. D'une manière générale, ces modèles relaxent l'accessibilité entre les sommets de 1 à k. Cependant, ces structures ne permettent pas de résoudre non plus les problèmes d'énumération ou l'intraitabilité du calcul. En outre, d'autres travaux se sont focalisés sur le degré de contrainte de la clique, comme le k-plex [Saito *et al.* 2008], le k-core[Seidman 1978], le DN-Graph [Wang & Cheng 2012], et le k-truss [Shao *et al.* 2014].

Dans [Jabbour *et al.* 2018], grâce à l'observation qu'une clique de taille k peut être vue comme deux cliques de taille k' et k'' telle que $k = k' + k''$ et les deux cliques forment un graphe biparti, nous avons proposé de relaxer l'une des deux cliques donnant lieu à la notion de *k-clique-star* (voir définition 11). Cette structure consiste en une clique et un ensemble de sommets (externe à la clique) connectés à tous ses éléments. Ici la clique est une sorte de centroïde.

Définition 11 (k-clique-star)**.** *Un graphe $G = (V, E)$ est dit k-clique-star ($k > 1$) s'il existe un sous-graphe $G' = (V', E')$ de G tel que G' est un k-clique, et $\forall\, u \in V \setminus V'$, $\forall\, v \in V'$, $(u, v) \in E$. G' est appelé le centroïde de G.*

Nous avons également montré que cette structure permet de garantir une borne de densité. Nous avons proposé de décomposer le graphe en un ensemble de *k-clique-star* maximaux. Cet

ensemble n'est autre que les communautés recherchées. Nous avons montrons aussi que chaque *k-clique-star* représente une extension d'une clique maximale par des arêtes additionnelles.

4.2.3 Compression de graphes à base contraintes pseudo-booléennes

Beaucoup de problèmes du monde réel sont modélisés efficacement comme des graphes. Cependant, ces représentations sont généralement de très grande taille conduisant à un calcul de complexité croissant dans le traitement de tels graphes. Partant de l'observation qu'un graphe arbitraire contient des structures cachées correspondantes à des classes particulières de graphes, dans [Jabbour *et al.* 2016c] nous nous sommes intéressés à la question de découvrir ces structures spécifiques et de les exploiter pour améliorer la compacité en se basant sur le formalisme logique. Pour cela, nous avons associé à chaque sommet une variable propositionnelle. Lorsqu'il n'y pas d'ambiguïté, nous confondons chaque sommet et sa variable booléenne associé.

Pour illustrer ce propos, considérons le cas de trois exemples de graphes représentés dans la figure 4.1. Le premier est la clique. Il est facile de déduire qu'il existe une bijection entre les arêtes de G et les impliquants premiers de la contrainte linéaire suivante. En effet, toute paire (x_i, x_j), $i \neq j$ est un modèle de la contrainte.

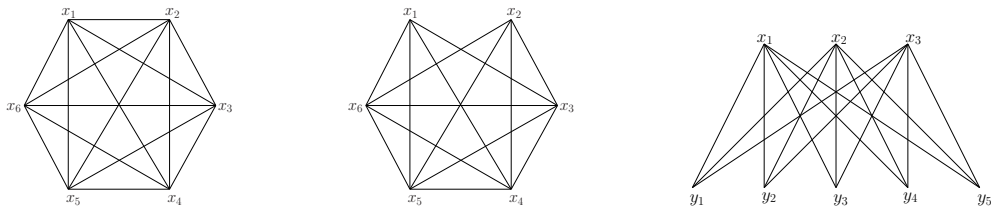
$$\sum_{i=1}^{n} x_i = 2$$

Le second est le cas de la quasi-clique, i.e., la clique avec une arête en moins (x_1, x_2). Tout couple de variables solution de l'équation suivante est une arête de la quasi-clique.

$$x_1 + x_2 + 2\sum_{i=3}^{n} x_i \geq 3$$

Le dernier cas intéressant est celui des graphes bipartis complets. Dans ce cas, la même correspondance existe entre les arêtes du graphe et les modèles de l'équation linéaire suivante :

$$2\sum_{u \in U} x_u + 3\sum_{v \in V} x_v = 5$$



a. clique b. quasi-clique c. biparti parfait

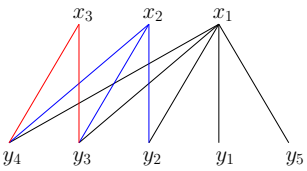
FIGURE 4.1 – Exemples de graphes

Les exemples précédents montrent que le formalisme logique peut offrir une alternative de représentation de choix pour les graphes tout en réduisant la taille de la représentation. Dans

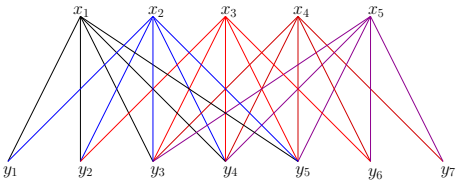
[Jabbour *et al.* 2016c], nous avons décrit les classes de graphes bipartis $G = (X \cup Y, E)$ pouvants être exprimés par des contraintes pseudo booléennes de la forme (4.8).

$$k \leq AX^T + BY^T \leq k', \quad A \in \mathbb{Z}^n,\ B \in \mathbb{Z}^m \tag{4.8}$$

Nous avons prouvé que les graphes imbriqués (Figure 4.2.a) et les graphes séquences (Figure 4.2.b), peuvent être exprimés d'une manière plus succincte en utilisant ces contraintes. Nous avons également présenté un algorithme basé sur une heuristique capable de partitionner un graphe en un ensemble de sous-graphes imbriqués disjoints.



c. Graphe biparti imbriqué



d. Graphe séquence

FIGURE 4.2 – Quelques classes de graphe

Chapitre 5

Représentation de connaissances & Raisonnement

5.1 Introduction

Le raisonnement en présence d'incohérence est un thème qui a reçu une attention particulière dans la littérature. En effet, il existe différentes situations où nous pouvons être ramener à travailler avec des informations conflictuelles. À titre d'exemple, lorsque différents experts ou capteurs fournissent des données contradictoires, plusieurs règles sont simultanément applicables dont les conclusions sont contradictoires, ou encore plusieurs avis divergents concernant l'évaluation d'un candidat ou d'un objet. Il est usuel qu'en présence d'incohérences de ne pas utiliser les méthodes classiques de déduction, qui trivialisent en inférant toutes les formules du langage. Il est donc primordial que la capacité de raisonner en présence de ces incohérences soit assurée.

Il peut s'avérer utile d'évaluer à quel point une base de connaissances est incohérente pour s'en servir afin de choisir par exemple, parmi un ensemble de bases, la base la moins conflictuelle. De même, savoir quelles sont les formules qui apportent beaucoup ou peu de conflit à la base de connaissances, peut être d'une grande utilité pour de nombreuses tâches, par exemple pour identifier les formules les plus problématiques afin de résoudre les conflits. L'autre possibilité lorsque l'on veut raisonner non trivialement à partir d'une base incohérente est de considérer certaines approches comme les logiques paraconsistantes, la théorie de l'argumentation, ou encore les méthodes basées sur les ensembles maximaux cohérents.

Ce qui suit résume nos contributions portant sur l'évaluation de la qualité des données dont les modes d'acquisition sont variés et présentent différentes incohérences et imperfections. Nous présentons ensuite nos travaux sur les approches argumentatives pour le raisonnement en présence des ontologies incohérentes.

5.2 Gestion et mesure de l'incohérence

L'incohérence est souvent inévitable dans les bases de données et dans les systèmes multi-agents. Quantifier le degré l'incohérence des bases de connaissances des agents facilite la compréhension d'un agent de son environnement et lui fournit une aide pour la prise de décision. Certains travaux ont émergé ces dernières années pour quantifier d'incohérence des bases de connaissances comme celles fondés sur les sous-ensembles minimaux incohérents [Hunter & Konieczny 2010, Grant & Hunter 2011, Mu *et al.* 2011, Xiao & Ma 2012], les sous-ensembles maximaux cohérents, ou celles qui consistent à calculer la proportion du langage affectée par l'incohérence

[Konieczny *et al.* 2003, Oller 2004, Grant & Hunter 2008, Ma *et al.* 2011]. Les mesures appartenant à cette deuxième catégorie sont souvent basées sur une sémantique paraconsistante car cela offre des modèles à des bases de connaissances logiquement incohérentes.

5.2.1 Définitions et notations

Nous considérons une base de connaissances K formée d'un ensemble fini de formules propositionnelles. Pour un sous-ensemble S, $|S|$ indique sa cardinalité. Nous dirons que K est *incohérente* si $K \vdash \bot$.

Un ensemble de formules M est dit *ensemble minimal incohérent* (MUS) de K ssi (1) M est incohérent et (2) tout sous-ensemble strict de M est cohérent. L'ensemble des ensembles minimaux incohérents de K est noté $MUSes(K)$. Les MUSes constituent les explications les plus fines, en termes du nombre de formules, de l'incohérence de la base de connaissances K. Bien évidement, les formules qui ne font parti d'aucun MUS sont dites des formules *libres*. L'ensemble des formules libres de K est noté $free(K)$. Dans le cas contraire, elle est dite non libre. Nous notons cet ensemble $unfree(K)$. Parallèlement à la notion de MUS, on peut définir les ensembles maximaux cohérents. Un ensemble M est dit *maximal cohérent* si (1) il est cohérent et (2) tout sur-ensemble strict de M est incohérent. Nous notons $MSSes(K)$ l'ensemble des sous-ensemble maximaux cohérents de K. Il existe une dualité entre les MUSes et les MSSes via la notion d'ensemble intersectant.

Formellement, une mesure d'incohérence peut être définie comme suit :

Définition 12. *Etant donné un langage propositionnel $\mathcal{L}$. Une mesure d'incohérence est une fonction $I : \mathcal{L} \to \mathbb{R}^+$ qui associe une valeur réelle pour toute base $K \in \mathcal{L}$.*

Afin de définir des mesures d'incohérence rationnelles, plusieurs recherches se sont intéressés à définir des propriétés qu'une mesure idéale doit satisfaire [Hunter & Konieczny 2010, Jabbour *et al.* 2014c, Besnard 2014]. Dans la suite nous listons les plus importantes parmi elles.

— *Cohérence :* $I(K) = 0$ ssi K est cohérente.
— *MinInc :* $I(M) = 1$ if $M \in MUSes(K)$.
— *Indépendance Libre :* $I(K \cup \{\alpha\}) = I(K)$ si $\alpha \in free(K \cup \{\alpha\})$.
— *Monotonie :* si $K \subseteq K'$, alors $I(K) \leq I(K')$.
— *Additivité :* $I(K_1 \cup \ldots \cup K_n) = \Sigma_{i=1}^{n} I(K_i)$ si $MUSes(K_1 \cup \ldots \cup K_n) = MUSes(K_1) \uplus \ldots \uplus MUSes(K_n)$, tel que $\uplus$ est l'union disjointe sur la famille d'ensembles.
— *Dominance :* si $\alpha \nvdash \bot$ et $\alpha \vdash \beta$, alors $I(K \cup \{\beta\}) \leq I(K \cup \{\alpha\})$.

Dans nos travaux présentés ci-dessous, nous avons pris le soin à la fois de critiquer certaines de ces mesures et d'en proposer des nouvelles tout en argumentant à travers de nouvelles propriétés.

5.2.2 Mesure de degré d'incohérence via les preuves minimales

Peu de travaux ont été proposés pour la caractérisation de la responsabilité de chaque formule d'une base de connaissances dans l'incohérence de la base. Le problème avec ces mesures est qu'elles ne donnent pas des résultats très satisfaisants parce qu'elles affectent d'une manière uniforme le degré d'incohérence entre les différents formules responsables du conflit. Pour améliorer

le comportement de cette famille de mesures, nous avons introduit dans [Jabbour & Raddaoui 2013] la notion de *preuve minimale* (ensemble minimal de formules à l'origine de l'inférence d'un littéral) afin de caractériser dans un cadre uniforme la responsabilité/participation de chaque formule dans l'incohérence de toute la base originale. Intuitivement, dans cette approche le degré d'incohérence d'une formule est le nombre de fois où cette formule intervient dans la production de conflit dans la base. Cela permet d'inspecter plus finement les conflits au niveau des formules. De manière assez naturelle, notre mesure s'étend pour quantifier à quel point une base de connaissances est incohérente. Notre approche satisfait les propriétés logiques de base que toute mesure d'incohérence intuitive doit satisfaire comme la consistance et la monotonie. Par ailleurs, nous avons montré l'utilité de cette mesure de conflit pour la résolution de l'incohérence dans les bases de connaissances.

Il est intéressant de noter qu'il a été prouvé dans [Thimm 2016] que notre mesure d'incohérence est la mesure la plus expressive de toutes les mesures de l'état de l'art.

5.2.3 Additivité restreinte

Dans [Jabbour *et al.* 2014c], nous nous somme intéressés à la propriété d'additivité. Cette propriété permet d'additionner le degré de conflit des sous-bases sous la condition que les seuls MUSes déductibles à partir de leur union est l'union des MUSes des sous-bases. Nous avons argumenté en faveur de l'insuffisance de cette condition et l'étendons en requérant également la disjonction en termes de formules non libre entre les sous-bases (voir Définition 13).

Définition 13 (Indépendance-Additivité)**.** *Soient K et K' deux bases de connaissances. $I(K \cup K') = I(K)+I(K')$ si $unfree(K)\cap unfree(K') = \emptyset$ et $MUSes(K\cup K') = MUSes(K)\uplus MUSes(K')$.*

À la lumière de cette nouvelle propriété, nous avons établi une borne inférieure des mesures d'incohérences. Cette borne a donné lieu à une nouvelle métrique qui consiste à calculer la taille maximale de l'ensemble de MUSes disjoints deux à deux (i.e, formant un set packing maximal) et qui est fermé par union (Définition 14).

Définition 14 (Set packing)**.** *Soit U un univers et S une famille de sous-ensemble de U. Un set packing est un sous-ensemble $P \subseteq S$ tel que, $\forall S_i, S_j \in P$ avec $S_i \neq S_j$, $S_i \cap S_j = \emptyset$.*

Dans [Jabbour *et al.* 2016b], nous avons généralisé cette notion aux ensembles. Cela a donné lieu à un nouveau problème appelé *set packing fermé* qui est une extension du problème bien connu de *set packing* proposé et étudié en recherche opérationnelle. Nous avons montré que ce nouveau problème est aussi difficile que le set packing standard de point de vue de complexité (Décider s'il existe un set packing fermé de taille inférieur ou égale à k est NP-Complet). La fermeture par union peut être obtenue en utilisant la définition suivante :

Définition 15 (Fermeture)**.** *Soit U un univers et S une famille de sous-ensemble de U. Nous définissons la fonction $f_S : 2^S \rightarrow 2^S$ comme $f_S(P) = \{S_i \in S \mid S_i \subseteq \cup_{S'\in P} S'\}$. Alors, un set packing $P \subseteq S$ est dit fermée si P est un point fixe de la fonction f_S, i.e., $f_S(P) = P$.*

Par ailleurs, nous avons proposé une extension de la mesure basée sur la borne inférieure en associant un vecteur pour chaque base de connaissances. Ce vecteur est une partition en set packing fermés maximaux de la base en question. De plus, deux mesures d'incohérences ont été

instanciées en agrégeant les valeurs de ce vecteur soit en considérant une somme pondérée ou en utilisant la fraction continue de ce vecteur. Nous avons également formulé le problème de set packing fermé et sa version pondéré en problèmes d'optimisation basés sur MaxSAT et la programmation linéaire 0/1.

5.2.4 Sous-additivité

Dans [Jabbour *et al.* 2017c], nous avons poursuivi l'exploration des mesures d'incohérences de point de vue syntaxique. Partant de l'observation que la disjonction entre MUS est un facteur accentuant la valeur de l'incohérence, nous avons proposé une nouvelle propriété appelée *sous-additivité* (Définition 16) qui est une contre-partie de la propriété de l'indépendance-additivité décrite précédemment.

Définition 16 (Sous-Additivité)**.** *Si* $MUSes(K \cup K') = MUSes(K) \uplus MUSes(K')$ *et* $unfree(K \cup K') \neq unfree(K) \uplus unfree(K')$*, alors* $I(K \cup K') < I(K) + I(K')$*.*

Afin de définir une mesure satisfaisant la propriété de la sous-additivité, nous définissons $G^K_{mus}(V, E)$ comme le graphe associé aux MUSes de K tel que $V = MUSes(K)$, et $(M, M') \in E$ ssi $M \cap M' \neq \emptyset$. De plus, nous dirons que S est une *couverture en MUS* (MC) de K ssi S est une couverture par sommets de $G^K_{mus}(V, E)$.

Lorsque deux MUSes partagent des formules, leur incohérence doit être définie sur des valeurs différentes pour satisfaire la propriété de sous-additivité. À cette fin, nous définissons d'abord une partition d'arêtes du graphe des MUSes de K.

Définition 17. *Nous définissons une partition en arêtes de* G^K_{mus} *comme*

$$P_e = \biguplus_{M \in V} P_e(M) \text{ t.q. } P_e(M) \subseteq \{(M, M') \in E\}$$

La définition 17 stipule que chaque arête entre deux MUSes M et M' appartient soit à $P_e(M)$ soit à $P_e(M')$. De plus, une partition en arêtes P_e peut contenir des ensembles vides et sa taille $|P_e| = |MUSes(K)|$. Et cela permet de quantifier la contribution de chaque MUS M en fonction du nombre d'arêtes de $P_e(M)$.

Maintenant, nous définissons la mesure d'incohérence d'une partition en arêtes comme suit :

Définition 18. *Soit* P_e *une partition en arêtes* G^K_{mus}*. Nous définissons la mesure d'incohérence de* P_e *comme*

$$Inc^f(P_e) = \sum_{M \in V} f(|P_e(M)|)$$

avec f *est une fonction strictement décroissante sur* $\mathbb{N}$ *qui satisfait* $f(0) = 1$ *et bornée par un réel* $0 < c < 1$*.*

Comme f est une fonction strictement décroissante sur la taille des éléments de la partition, cela signifie que si une arête est ajoutée à $P_e(M)$, la contribution de M à la mesure d'incohérence de P_e diminue. En d'autres termes, le degré d'incohérence d'un MUS M dépend du nombre d'arêtes attribuées à $P_e(M)$. Différentes fonctions f peuvent être identifiées. Par exemple, on peut considérer $f_1(n) = 1 - c + c^{n+1}$ ou encore $f_2(n) = 1 - c + \frac{c}{n+1}$ tel que $0 < c < 1$.

En utilisant la partition P_e, nous avons défini le degré d'incohérence de K comme

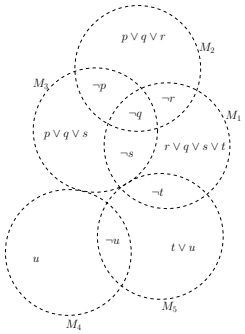
$$Inc(K) = \max_{P_e} \; Inc^f(P_e)$$

Le calcul de $Inc(K)$ dépend de la fonction f utilisée. Comme on peut le voir avec l'exemple suivant, dans le cas où $f = f_1$ ou f_2, on obtient une fonction non-linéaire à optimiser.

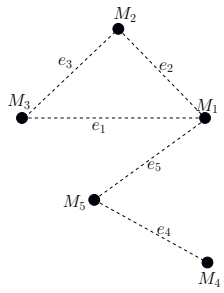
Exemple 3. *Pour l'exemple de la figure 5.1, en utilisant les fonctions f_1 et f_2, le calcul des mesures d'incohérences revient à optimiser les fonctions suivantes :*

$$Inc^{f_1}(P_e) = 5(1-c) + c^{1+x_{e_1}+x_{e_2}+x_{e_5}} + c^{1+\neg x_{e_2}+x_{e_3}} + c^{1+\neg x_{e_1}+\neg x_{e_3}} + c^{1+x_{e_4}} + c^{1+\neg x_{e_4}+\neg x_{e_5}} \tag{5.1}$$

$$Inc^{f_2}(P_e) = 5(1-c) + \frac{1}{1+x_{e_1}+x_{e_2}+x_{e_5}} + \frac{1}{1+\neg x_{e_2}+x_{e_3}} + \frac{1}{1+\neg x_{e_1}+\neg x_{e_3}} + \frac{1}{1+x_{e_4}} + \frac{1}{1+\neg x_{e_4}+\neg x_{e_5}} \tag{5.2}$$



Hypergraphe des MUSes de K



Graphe des MUSes de K

FIGURE 5.1 – De l'hypergraphe au graphe des MUSes de K

Exploiter la structure des MUSes pour définir de nouvelles mesures d'incohérence a fait l'objet d'un autre travail. Dans [Jabbour & Sais 2016], nous avons considéré les ensembles indépendants de MUSes, i.e., ensemble de MUSes disjoints deux à deux. Cette structure permet de mieux capturer la structure de MUSes en termes d'intersection et offre la possibilité d'étendre plusieurs mesures d'incohérence existantes. À la lumière de cette notion, nous avons associé à chaque base de connaissances la séquence $\langle \delta_0 \ldots \delta_i \ldots \delta_n \rangle$ où δ_i est le nombre des sous-ensemble de MUS indépendants de taille i permettant de définir la mesure $Inc(K) = log(\delta_1 + \ldots + \delta_n)$. Cette mesure vaut $Inc(K) = \sum_{i=1}^{n} C_n^i = log(2^n) = n = |MUSes(K)|$ lorsque les MUSes sont disjoints deux à deux.

5.2.5 Dominance faible

Dans [Jabbour *et al.* 2014d, Jabbour *et al.* 2014e, Jabbour *et al.* 2017b], nous nous sommes intéressés à l'une des propriétés des mesures d'incohérences nommée dominance. Cette propriété stipule que le degré de conflit ne peut pas accroître si une formule est substituée par l'une de ces conséquences logiques. Cette propriété est rarement satisfaite par les différentes

mesures de l'état de l'art. Nous avons proposé dans [Jabbour *et al.* 2017c, Jabbour *et al.* 2014d, Jabbour *et al.* 2014d] une version moins forte de cette propriété appelée *dominance faible*, jugée plus intuitive que l'originale à notre avis car elle est basée sur une notion de conséquence logique plus restreinte mais rationnelle.

Définition 19 (Dominance Faible)**.** *Si $\alpha \nvdash \bot$ et $PI(\beta) \subseteq PI(\alpha)$, alors $I(K \cup \{\alpha\}) \geqslant I(K \cup \{\beta\})$.*

Basée sur cette notion de conséquence logique restreinte, nous avons à la fois explorer deux questions fondamentales. De point de vue sémantique, nous avons donné une caractérisation logiques des variables conflictuelles et montrer leur utilité face aux approches existantes basées sur des sémantiques multi-valuées. Par ailleurs, nous avons introduit la notion de MUS déduit (DMUS) permettant de caractériser des MUSes construits en utilisant la conséquence logique restreinte définie précédemment. Nous avons également montré que le raisonnement avec les MUSes déduits permet de capturer plus finement les sous-ensembles insatisfiables au sein d'un ensemble de formules. Nous avons présenté également une nouvelle mesure permettant de satisfaisant la propriété de la dominance faible.

5.2.6 Restauration de la cohérence

Nous nous somme également intéressés dans [Jabbour & Sais 2016] à la restauration de la cohérence d'une base de connaissance incohérente, l'autre question fondamentale dans le cadre de la gestion de l'incohérence. Nous nous sommes focalisés précisément sur les méthodes basées sur la restauration de la cohérence par suppression de formules. Dans ce contexte, nous avons introduit une approche permettant de supprimer le nombre minimum de formules nécessaire pour restaurer la cohérence tout en évitant de supprimer complètement des MUSes. L'objectif est de préserver le plus possible les connaissances de la base originale. Nous avons formulé ce problème sous l'appellation de *couverture minimale de MUSes* définit comme suit.

Définition 20 (Couverture Minimale de MUSes)**.** *Le problème de Couverture Minimale de MUSes est défini comme le problème de trouver le plus petit ensemble intersectant qui couvre le nombre minimum de $MUSes(K)$.*

Intuitivement, pour un ensemble intersectant H, on associe l'ensemble $C_{mus}(H)$ définit par $C_{mus}(H) = \{M \in MUSes(K),\ M \subseteq H\}$. Cela permet de définir une relation de préférences entre les ensembles intersectants comme suit :

$$H_1 \preceq H_2 \text{ ssi } |C_{mus}(H_1)| \leq C_{mus}(H_1) \text{ ou } |C_{mus}(H_1)| = |C_{mus}(H_2)| \text{ et } |H_1| \leq |H_2|$$

.

Notre objectif est donc de trouver les ensembles intersectants préférés par rapport à la relation $\preceq$. En relaxant ou en contraignant la relation de préférence $\preceq$, nous avons pu définir plusieurs problèmes sous-jacents. À titre d'exemple, le problème qui consiste à trouver un ensemble intersectant ne couvrant aucun MUS (lorsque les MUSes sont préalablement calculées), est identique à *SIM-UNSAT* défini dans le cadre propositionnel et dont les instances sont formées de clauses totalement positives P et d'autres totalement négatives obtenue à partir de P en inversant chaque littéral p en $\neg p$. Ce problème fait partie d'un ensemble de problèmes proposés par Thomas Eiter en 1996 [Eiter 1996] et dont la complexité est encore une question ouverte. Le problème de

décision associé à notre approche généralise celui de *SIM-UNSAT*. Dans [Jabbour 2016], nous avons étudié plusieurs variantes de notre nouveau problème. Nous avons également formulé des encodages en MaxSAT de ce problème de couverture minimale et ses variantes.

5.3 Théorie de l'argumentation

Il est parfois difficile d'assurer la cohérence d'une base de connaissances. Il faut alors être capable de raisonner en présence de contradictions. Une classe d'approches de raisonnement en présence d'informations incertaines et incohérentes est récemment proposée, dite théorie de l'argumentation [Dung 1995]. L'argumentation est un modèle attrayant pour raisonner avec des connaissances imparfaites ou conflictuelles ; elle est essentiellement fondée sur la justification d'une conclusion plausible par des raisons en faveur de celle-ci. Depuis quelques années, l'argumentation formelle est devenue un thème de recherche très populaire et elle a un large éventail d'applications pour plusieurs domaines comme le juridique, le médical, la prise de décision, la négociation entre agents, etc. Elle est aussi étudiée dans diverses disciplines comme la psychologie, la philosophie et la linguistique.

Nous présentons dans cette section nos approches autour du raisonnement à base d'argumentation.

5.3.1 Enumération des extensions Top-k

Le cadre d'argumentation abstrait initié par Dung dans [Dung 1995] représente simplement un système d'argumentation par un graphe où les sommets sont les arguments et les arcs les attaques entre ces arguments. Ensuite le problème est de définir les arguments qui peuvent être conjointement acceptés, c'est le rôle des différentes sémantiques proposées.

Ce cadre argumentatif abstrait a récemment reçu un intérêt considérable. Malgré leur simplicité, ce cadre fournit un outil puissant qui capte différentes approches de raisonnement nonmonotone. Malheureusement, dans ces systèmes, la plupart des tâches de raisonnement souffrent d'une complexité très élevée. En particulier, l'énumération de toutes les extensions [1] est généralement un problème intraitable, puisque le nombre d'extensions peut être exponentiel dans le pire des cas. Ainsi, la réduction de la taille de la sortie dans le cadre d'argumentation abstrait est clairement une question de recherche très importante.

Dans [Jabbour *et al.* 2016d], nous nous sommes intéressés au problème qui consiste à sélectionner les "meilleures" extensions d'un sytsème d'argumentation abstrait. Cette sélection se fait par agrégation afin d'optimiser la recherche et de manipuler moins de quantités de données. Plus précisément, vue que l'espace des alternatives considérées est de grande taille à cause de sa structure combinatoire, il est illusoire de vouloir représenter et de maintenir des préférences données sous forme explicite. Pour se faire, nous avons exploité les préférences au niveau de la sémantique de choix des extensions afin de réduire la taille de sortie. Contrairement aux approches existantes qui consistent à injecter les préférences au niveau atomique (sur les arguments), nous avons défini des relations de préférence sur des ensembles d'arguments. Ensuite, nous avons présenté un

1. Une extension est un ensemble d'arguments acceptables. Dans le système de Dung [Dung 1995], plusieurs sémantiques sont données à l'acceptabilité des arguments.

algorithme de calcul des extensions Top-k en se basant sur une relation de préférence spécifiée par l'utilisateur. En effet, une extension est dite Top-k sous une sémantique donnée si elle admet moins de k extensions préférées à celle-ci par rapport à une relation de préférence donnée. Différentes notions de préférence ont été introduites conduisant à différentes politiques de choix des extensions. Finalement, une étude expérimentale sur divers benchmarks a démontré l'efficacité de l'approche proposée.

5.3.2 Raisonnement sur des ontologies conflictuelles

Dans le cadre de nos travaux sur l'argumentation dans un cadre de logique, nous avons étudié le problème de raisonnement sur des ontologies en présence de conflit [Bouzeghoub *et al.* 2017]. En effet, le web sémantique constitue un environnement dans lequel les agents humains et machines vont communiquer selon une base sémantique. Il utilise la notion d'ontologies pour la conceptualisation et l'extraction des connaissances du domaine et les stocke en termes de concepts et de propriétés dans la machine d'une manière compréhensible et traitable. En raison de leurs capacités de décidabilité et d'expressivité, les ontologies ont joué un rôle fondamental pour décrire la sémantique des données non seulement dans le web sémantique émergents, mais aussi dans l'ingénierie des connaissances traditionnelles et les systèmes de traitement de l'information. Néanmoins, l'augmentation du nombre d'ontologies développées et maintenues sur le web, génère inévitablement des incohérences. Dans ce contexte, nous avons étendu la théorie de l'argumentation déductive de Besnard & Hunter dans le contexte de la logique $\mathcal{SHIOQ}$ [Baader *et al.* 2010], un fragment expressif de la logique de description. Différentes notions de conflit ont été étudiées, à savoir l'incohérence qui signifie qu'une des classes de l'ontologie est nécessairement vide d'individu et l'inconsistance qui implique l'absence d'un modèle. Cela nous a permis, contrairement aux approches existantes, la détection automatique et la localisation des incohérences sémantiques. Nous avons également étendu notre approche argumentative au cadre des ontologies incertaines afin de prendre en compte l'aspect incertain des données disponibles et d'analyser les réponses aux requêtes en se basant sur des logiques de description possibilistes. Dans ce contexte, nous avons exploré les différentes manières d'utiliser les poids sur les arguments pour définir des nouvelles relations d'inférence, ce qui permet de prendre en compte l'incertitude tout au long du processus d'inférence. Nous avons en particulier montré que plusieurs relations d'inférences de l'état de l'art peuvent être capturées via notre cadre d'argumentation possibiliste.

Chapitre 6

Composition de services web

6.1 Introduction

Les évolutions impressionnantes du monde de l'informatique et de l'internet ont entrainé le développement de nouveaux paradigmes d'interaction entre applications. Notamment, l'architecture orientée service (Service Oriented Architecture, ou SOA) qui a été mise en avant afin de permettre des interactions entre applications distantes. Cette architecture est construite autour de la notion de service, qui est matérialisé par un composant logiciel assurant une fonctionnalité particulière et accessible via son interface. Un service est toujours accompagné d'une description fournissant aux applications les informations nécessaires à son utilisation. Cette évolution liée aux technologies internet et web est un facteur important dans la création et l'explosion de l'informatique dans les nuages ou *Cloud Computing*. Ce dernier fournit un ensemble de ressources informatiques partagées telles que les applications, le stockage, le calcul, la mise en réseau et le développement. Cette technologie vise à permettre l'accès à de larges ressources de calcul de manière totalement virtualisée, en les regroupant et en offrant une vue système unique. La diffusion rapide des services web et des services orientés systèmes constitue un pas important vers un changement radical dans le processus de développement logiciel.

Les services web constituent un bon paradigme pour concevoir et créer des applications complexes. Ce paradigme est utilisé par l'informatique dans les nuages, plus précisément par le SaaS, en tant qu'élément principal avec des descriptions d'interface bien définies. Ces services web sont définis comme un ensemble d'opérations abstraites et de messages qui doivent souvent être combinés pour former un service complexe à l'aide du mécanismes de composition.

La composition dynamique des services Web exige que les consommateurs de services découvrent des fournisseurs de services répondant à des exigences fonctionnelles et non fonctionnelles données, notamment des exigences de coût et de qualité de service, telles que la performance et la disponibilité. Les attributs de qualité de service d'un service web incluent des mesures telles que le temps de réponse, le débit, la sécurité et la disponibilité faisant référence à ce qu'on appelle la *qualité de services*.

Au fil des années, le nombre de services web avec des fonctionnalités similaires et une qualité de service irrégulière a considérablement augmenté. La sélection des services web devient compliquée, ce qui rend nécessaire de garantir une composition avec une meilleure qualité de service bien que cette tâche est difficile (problème NP-hard).

Le problème de composition de services web consiste à trouver automatiquement une séquence optimale (ou de meilleure qualité) de services web satisfaisant la requête de l'utilisateur. La littérature est riche d'approches de composition de services web [Nacer *et al.* 2015, Shiaa *et al.* 2008, Rodriguez-Mier *et al.* 2011, Okutan & Cicekli 2010, Wu & Khoury 2012, Balbiani *et al.* 2010].

Dans la suite nous donnes une description des méthodes à base de contraintes proposées dans le cadre de la composition de services web.

6.2 Définitions et cadre formel

Les applications en tant que service sont construites à partir d'un ensemble de services web accessibles via internet. Un fournisseur différent peut proposer un ensemble de services Web avec les mêmes fonctionnalités. Un service web est une technologie qui permet aux applications de communiquer à distance via Internet, et cela indépendamment des plates-formes et des langues sur lesquelles ils s'appuient. Pour cela, les services web s'appuient sur un ensemble de protocoles internet largement utilisés, tels que XML, HTTP, pour communiquer.

Un service web est considéré comme un composant logiciel avec ses paramètres d'entrée I et ses paramètres de sortie O.

Définition 21. *Un service web est un triplet $w = (I, O, Q)$ tel que :*

- *I : est un ensemble de paramètres d'entrée,*
- *O : est un ensemble de paramètres de sortie,*
- *Q : représente la qualité de service associée à w.*

Lorsque c'est utile, pour un service web $w = (I, O, Q)$, nous notons $I(w) = I$, $O(w) = O$, et $Q(w) = Q$. De plus, nous omettons de mentionner la qualité de service $w = (I, O)$ lorsqu'elle n'est pas requise.

Pour décider de la relation d'invocation de $w_1 = (I_1, O_1)$ à $w_2 = (I_2, O_2)$ dans la composition, il est nécessaire de comparer sémantiquement les sorties O_1 de w_1 avec les entrées I_2 de w_2. Pour cela, nous devons calculer une similarité sémantique entre deux paramètres ; c'est-à-dire qu'il faut trouver une relation entre deux représentations de connaissances. Une fonction de jonction décrit la correspondance sémantique entre deux paramètres et formulée comme suit.

Définition 22 (Fonction de Jonction)**.** *Étant donné deux services Web $w_1 = (I_1, O_1, Q_1)$ et $w_2 = (I_2, O_2, Q_2)$, une fonction de jonction renvoie true si i.type = o.type tel que $i \in I_2$ et $o \in O_1$, ou i.type est un sous-type de o.type.*

Étant donné un ensemble W de services web disponibles et une requête $w_r = (I_r, O_r)$, *le problème de composition de services web* consiste à trouver un service web $w = (I, O) \in W$ tel que $I_r \subseteq I$ et $O_r \in O$. Cependant, souvent, il n'existe pas de service web unique répondant aux exigences de la requête. Dans ce cas, le but est de trouver un ensemble de services web à combiner pour satisfaire la requête.

Nous avons proposé des transformations vers la logique propositionnelle afin de trouver une composition de services web optimale maximisant la qualité de service QoS. Deux approches ont été proposées.

6.3 Composition de services web à base d'ensembles minimaux incohérents

Dans [Wakrime & Jabbour 2015], nous avons montré que la recherche d'une composition optimale est équivalente à la recherche du plus petit MUS. Pour une telle modélisation, nous avons associé à chaque paramètre d'entre i_k (resp. de sortie o_k) une variable propositionnelle p_{i_k} (resp. q_{o_k}). Pour chaque service web w une variable propositionnelle r_w est également associée. La jonction est exprimée par l'égalité entre un paramètre d'entrée i_k et un paramètre de sortie $i_{k'}$ par $p_{i_k} = q_{o_{k'}}$ lorsque cette dernière est possible.

L'encodage est obtenu en considérant la contrainte de l'équation (6.1). Pour chaque service web w, cette contrainte stipule que si les paramètres d'entrée de w sont actifs, alors le service web est actif pour composition ; et par conséquent le service pourrait être utilisé dans la composition. De plus, si le service web w est actif alors ces paramètres de sortie le sont également.

$$\bigwedge_{(\langle i_1 \ldots i_k \rangle, \langle o_1 \ldots o_{k'} \rangle) \in W} (p_{i_1} \wedge \ldots \wedge p_{i_k} \rightarrow r_w) \wedge (r_w \rightarrow q_{o_1} \wedge \ldots \wedge q_{o_{k'}}) \tag{6.1}$$

Notons que pour une requête utilisateur de la forme $w_r = (\langle i_1 \ldots i_n \rangle, \langle o_1 \ldots, o_m \rangle)$, une composition est possible si la formule $\Phi_w = (6.1) \wedge p_{i_1} \wedge \ldots \wedge p_{i_n} \wedge \neg q_{o_1} \wedge \ldots \wedge \neg q_{o_m}$ est insatisfiable par propagation unitaire ($\Phi_w \vdash_* \bot$). De plus, trouver la composition optimale est équivalent à trouver un MUS de la plus petite taille possible contenant $p_{i_1} \wedge \ldots \wedge p_{i_n} \wedge \neg q_{o_1} \wedge \ldots \wedge \neg q_{o_m}$ et contenant le nombre minimum de services web W. Dès lors, les techniques de recherche du MUS de taille minimum peuvent être utilisées directement pour trouver une composition optimale. Nous avons proposé également une extension de cette approche pour prendre en compte la qualité de service (QoS).

6.4 Composition de services web à base d'impliquants premiers

Dans [Wakrime & Jabbour 2017], nous avons proposé une autre approche *bottom-up* pour trouver une composition de services web optimale. L'encodage est composé des deux contraintes suivantes :

$$r_w \rightarrow \bigwedge_{i \in I \setminus I_r} \bigvee_{w' \mid i \in O(w')} r_{w'} \quad \text{pour tout } w = (I, O) \in W \tag{6.2}$$

$$\bigvee_{w \mid o \in O(w)} r_w \quad \text{pour tout } o \in O_r \tag{6.3}$$

La contrainte (6.2) est obtenue en fusionant deux contraintes. La première stipule qu'un service web est actif à la composition si tous ces paramètres d'entrée le sont aussi, i.e., $r_w \rightarrow \bigwedge_{i_k \in I(w)} p_{i_k}$. La seconde indique qu'un paramètre d'entrée i_k est actif s'il fait partie de l'entrée de la requête de l'utilisateur $i \in X$ ou s'il est également le paramètre de sortie d'au moins un service web actif, i.e., $p_{i_k} \rightarrow \bigvee_{w' \mid i \in O(w')} r_{w'}$. La seconde contrainte (6.3) traduit le fait que les paramètres de sorties de la requête doivent être actifs. Cela est facilement exprimé par le fait que chacun de ces paramètres doit être la sortie d'au moins un service web qui est lui même actif.

Contrairement à l'approche [Wakrime & Jabbour 2015], l'avantage de ce nouvel encodage est qu'il est restreint aux variables booléennes représentant les services web. Ensuite la formule obtenue est reverse-horn. Par conséquent, l'existance d'une composition est équivalente à la satisfiabilité de la formule reverse-horn qui peut être vérifié en temps linéaire. Nous avons prouvé que les impliquants premiers correspondent aux compositions minimales et ceux de la plus petite taille correspondent exactement aux compositions optimales en nombre de services web.

La formulation générale du problème de la composition de services web avec la QoS (à minimiser) lorsque cette dernière est additive, peut être formulée via le problème bien connue de MinSAT de la manière suivante :

$$min \sum_{w \in W} Q(w) w_r \text{ subject to } (6.2) \wedge (6.3)$$

Par ailleurs dans [Wakrime *et al.* 2015], nous avons traité le cas des requêtes qui n'aboutissent pas. Nous avons proposé un cadre à base de contraintes pour trouver le minimum de changement par ajout de paramètres d'entrée afin de rendre la réponse à la requête possible. L'objectif étant d'offrir à l'utilisateur une alternative à sa requête qui pourrait éventuellement répondre à ses exigences.

Chapitre 7

Bilan et perspectives

Durant ces dix dernières années, nous avons exploré diverses thématiques de recherche. Certaines en adéquation avec les axes de recherche du CRIL, d'autres constituent des thématiques d'ouverture comme la fouille de données, l'apprentissage et la composition de services web. Nous avons mis en lumière de multiples connexions qui nous ont permis d'opérer de riches fertilisations croisées. Nous avons essayé de répondre à plusieurs questions liées à la résolution effective des problèmes de décision, au raisonnement en présence d'incohérence et à l'efficacité des approches logiques pour la fouille de motifs ensemblistes et séquentiels. Par exemple, nous avons défini un cadre logique pour calculer les Top-k motifs via la définition d'un nouveau problème, appelé Top-k SAT, un concept nouveau en logique propositionnelle. Nous avons étendu le concept de symétries largement exploré en SAT/CP vers la fouille de motifs ensemblistes et proposé des approches logiques pour la fouille des motifs ensemblistes et séquentiels. Certaines contraintes, comme la contrainte de cardinalité et sa variante conditionnelle, récurrentes dans ces transformations, ont fait l'objet d'une étude spécifique ayant conduit à de nouveaux encodages efficaces. Nos contributions à l'amélioration de résolution pratique du problème SAT sont essentielles et touchent à différents aspects, allant de l'amélioration de l'apprentissage des clauses, à la définition de nouvelles mesures de qualité des clauses apprises en passant par la mise en œuvre de stratégies de recherche plus efficaces (e.g., redémarrages dynamiques). Notre approche parallèle de type portfolio a poussé encore plus loin les limites des solveurs SAT modernes. Nos propositions d'amélioration de l'échange de clauses et du contrôle de leurs qualités a été mise à profit dans ce cadre et dans le cas distribué.

Nos contributions au raisonnement en présence d'incohérence ont fait émerger d'autres questions intermédiaires aussi complexes mais autant importantes comme l'extension des problèmes bien connus de "Set Packing" et "SIM-UNSAT", en respectivement "Set Packing Fermé" et "SIM-UNSAT Généralisé" que nous avons étudié et analysé du point de vue de la complexité.

Parmi les autres problématiques en vogue à l'heure actuelle et que nous avons abordé, on peut citer celle liée à la taille des données de plus en plus volumineuses. Nous avons proposé plusieurs techniques pour différentes représentations. D'un côté, nous avons introduit un cadre pour compacter les problèmes de contraintes (ensemble de clauses (SAT) et contraintes tables ou en extension (CP)) en utilisant la fouille de données. Pour les graphes, nous avons défini une approche de compression basée sur les contraintes pseudo-booléennes. Nous avons également proposé des approches symboliques pour la détection de communautés dans les grands graphes de données. Finalement, nous avons utilisé des techniques de décomposition et l'approche parallèle étudiée en logique pour améliorer sensiblement l'efficacité des approches déclaratives de fouille de motifs ensemblistes sur des données de grandes tailles.

Sur les aspects liés à la modélisation de problèmes, nous avons montré que la composition de services web peut être encodé en logique propositionnelle, tout en proposant des approches pour

une composition optimale qui prend en compte la qualité des services web.

Pour résumer ce bilan, nous citons certains de nos travaux que nous pouvons modestement qualifier de précurseur (ou parmi les premiers) dans le domaine. Il s'agit de nos contributions autours de(s) :

1. approches de type portfolio pour la résolution parallèle du problème SAT,
2. restarts dynamiques,
3. l'apprentissage de clauses à partir des succès,
4. la fouille de séquences par des approches contraintes,
5. l'utilisation des symétries en QBF,
6. la détection et l'élimination des symétries en fouille de données,
7. l'utilisation de la fouille de données pour extraire des motifs pour la compression de formules booléennes et de réseaux de contraintes,
8. la représentation des graphes par une disjonction de contraintes pseudo-booléennes,
9. le clustering de données complexes représentées par des formules logiques,
10. approches logiques pour la détection de communautés.

Nos travaux antérieurs ouvrent de nombreuses perspectives qui méritent d'être explorées. J'ai fait le choix de limiter ma présentation autours de trois grandes directions de recherche :

Fouille de données & Contraintes : Encouragé par les récents développements, notre projet a pour cadre la fouille de données par contraintes. Il s'articulera autour de deux problématiques de recherche :

1. Extraction de motifs sous contraintes : Il s'agit de poser un cadre plus flexible et générique pour prendre en compte les contraintes, les préférences et les critères d'intérêts que l'utilisateur souhaite optimiser. Cette étude ne se limitera pas aux données transactionnelles, nous proposons d'élargir le cadre de cette étude à d'autres types de données comme les séquences et les graphes attribués. Pour l'extraction de connaissances à partir des bases de données transactionnelles, nous avons initié l'étude de deux problématiques avec un potentiel applicatif plus large. Il s'agit de la découverte de motifs profitables (générant un profit élevé), appelé « high utility itemset mining », une problématique ayant émergée récemment. Ce problème vient répondre à une limitation importante du cadre classique, qui considère que les items ont la même importance. Or dans de nombreuses applications, les items peuvent être différenciés par plusieurs attributs associés comme la quantité, le prix ou plus généralement l'utilité. Dans ce cadre, les motifs ne sont ni monotones ni anti-monotones rendant le problème d'énumération plus difficile. Or les approches déclaratives à base de contraintes peuvent être très utiles dans ce cadre, car elles disposent de mécanismes d'élagages de l'espace de recherche plus spécifique. Le second problème concerne la découverte de motifs graduels à partir de bases de données numériques, permettant de capturer des relations entre attributs en termes de variations. Ce problème trouve de très nombreuses applications dans divers domaines incluant les données médicales, biologiques ou statistiques.

2. Fouille qualitative de données : l'objectif est de proposer de nouvelles caractérisations des motifs à extraire en intégrant des notions de pertinences plus fortes comme la robustesse et la stabilité. Ceci nous permettra de réduire significativement la taille de la sortie en se focalisant sur les motifs les plus pertinents. Certains types de modèles robustes définis dans le cadre de la logique peuvent trouver ici une possible extension en fouille de données. Nous intégrerons également diverses propriétés structurelles comme les symétries pour réduire la taille de l'espace de recherche des algorithmes d'énumération.
3. Analyse logique des données : L'analyse logique des données (LAD) est une méthodologie principalement basée sur la notion de motif. Considérant une base de données constituée de deux groupes d'observations booléennes (appelées observations positives et négatives), la méthode LAD consiste à trouver un sous-ensemble d'attributs ayant les mêmes valeurs sur certaines observations de l'ensemble positif alors que ces valeurs ne peuvent pas être observées dans le groupe négatif. Ce sous-ensemble s'appelle un motif et le but est de trouver des motifs partagés par autant d'observations positives que possible. Cette approche proposée il y'a plus de 20 ans par Peter L. Hammer, est essentiellement connue dans la communauté des mathématiques discrètes par ces liens aux travaux autours sur les fonctions booléennes partiellement définies. Elle reste largement ignorée par la communauté fouille de données. Les nombreuses formulations connues sont basées sur des modélisations sous forme de programmes lineaires 0/1 avec un nombre quadratique de contraintes linéaires. La première piste serait de réduire cette complexité en espace pour un réel passage à l'échèle. Nous envisageons également d'élargir le champ d'application de la LAD au domaine du test logiciel.
4. Fouille de données symbolique : Aujourd'hui les données peuvent être plus complexes, de nature hétérogènes, contextuelles et multi-sources. Elles peuvent représenter par exemple les désirs ou préférences des clients collectés de différentes manières. Extraire des connaissances à partir de ces données complexes passe par une étape de pré-traitement qui consiste à trouver la représentation la plus appropriée. La logique peut constituer un formalisme de choix pour décrire ces données ou entités complexes. Il s'agit ensuite de mettre en œuvre des nouvelles approches d'extraction de connaissances adaptées à de telles représentations. Notre premier travail autour du clustering symbolique est un pas dans cette direction.
5. Détection de communautés : Dans ce cadre, nous souhaitons élargir nos travaux à la détection de communautés dans les graphes orientés et attribués. Il s'agit de tenir compte des informations (attributs, flux) étiquetant les liens ou les arcs. La non prise en compte de ces informations est une réelle simplification de la notion de communauté qui est caractérisée non pas par les liens mais aussi par les informations attachées aux sommets et aux arêtes.

Fertilisation croisée entre la fouille de données et l'IA : Nous avons montré que les domaines de l'IA et de la fouille de données peuvent tirer un réel bénéfice d'une fertilisation croisée. Dans ce cadre, nous souhaitons approfondir cette voie de recherche fructueuse dans plusieurs directions. La première consiste à étudier les possibilités d'intégration de divers paradigmes de l'IA comme par exemple le raisonnement sur les connaissances extraites (post-traitement), l'intégration des symétries dans diverses tâches de fouille de données, le pré-traitement des données par

des approches issues de la compilation des bases de connaissances et la prise en compte des préférences dans le processus de fouille de données. Divers types de solutions en logique peuvent être étendu pour définir des motifs plus robustes. Il s'agit aussi de voir dans quelle mesure la fouille de données (extraction de motifs, clustering, détection de communautés) peut être utile pour analyser et extraire diverses formes de structures dans les formules SAT, dans les réseaux de contraintes CSP et dans les graphes, pour compresser et améliorer la résolution de problèmes dans ces formalismes.

Extraction, analyse et visualisation de motifs complexes à partir des données issues de processus migratoires : Dans le cadre du projet CNRS Mastodons QDoSSI [1], une collaboration est en cours avec des chercheurs en sciences humaines et sociales (SHS) autours de l'analyse et de la compréhension des processus à l'œuvre dans la construction des parcours migratoires. Notre domaine d'analyse se concentre sur des données structurées (e.g., biographie migratoires) et non structurés (e.g., récits de vie) collectées par les laboratoires SHS sur des terrains différents (e.g., Afrique subsaharienne, et moyen orient) et portent sur des populations diverses (e.g., migrants, réfugiées, etc.).

Ce projet multidisciplinaire ouvre la voie au développement de nouvelles techniques en rupture avec l'existant dans le domaine des bases de données et de l'extraction des connaissances :

- Qualité des données et pré-traitement : les données mobilisées sont hétérogènes avec des temporalités et des représentativités variables. Elles présentent diverses imperfections, comme la présence de données manquantes, de données mal orthographiées ou encore de données entachées d'incertitudes et d'imprécisions. Ceci pose le problème de la mesure, de l'intégration et de l'évaluation de l'impact de la qualité des données sur les résultats d'analyses dans un contexte spatio-temporel évolutif.
- Extraction de motifs complexes à partir de données complexes : notamment, l'extraction des connaissances complexes (ici les parcours/graphes) à partir de données textuelles et/ou relationnelles. Les graphes extraits, et enrichis sémantiquement, ouvrent la voie à de nouveaux champs d'extraction de connaissances, d'analyses et d'interrogations de ces structures combinatoires.
- Visualisation de graphes attribués : les graphes attribués associés aux parcours migratoires sont complexes et renseignent sur les lieux et les étapes parcourues par les migrants. L'objectif est de proposer des techniques de visualisations de graphes capable de mettre en valeur la richesse et la diversité des informations associées aux sommets et aux arêtes. Cette visualisation doit être dynamique et interactive. Il s'agit également de projeter ce graphe sur une carte géographique.

1. Qualité des données multi-Sources : un double défi pour les sciences sociales et les sciences de l'Informatique – url : http ://www.cril.univ-artois.fr/qdossi/

Chapitre 8

Activités de recherche

— Titulaire de la PEDR depuis octobre 2013.

— Délégation CNRS, CRIL CNRS UMR 8188, 2017, 6 mois.

8.1 Valorisation scientifique

8.1.1 Distinctions

— Prix du meilleur papier (**Best Student Paper Awards**), Imen Ouled Dlala, Saïd Jabbour, Lakhdar Saïs, Yakoub Salhi, andBouteina Ben Yaghlane, Parallel SAT Based Closed Frequent Itemsets Enumeration, (AICCSA'2015), Marrakech, Maroc.
— Nomination pour le prix du meilleur papier (Best Student Paper Runner-Up), Saïd Jabbour et Badran Raddaoui, Measuring Inconsistency Through Minimal Proofs, Twelve European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU), 2013, (89 soumissions – 44 papiers acceptés – 4 papiers nominés)
— Prix du meilleur article de la conférence (**Best Paper Awards**), Youssef Hamadi, Saïd Jabbour, Lakhdar Saïs, "Learning for Dynamic Subsumption", 21st IEEE International Conference on Tools with Artificial Intelligence (ICTAI), Newark, New Jersey, USA, 2-4 November 2009

8.1.2 Logiciels

— **Solveurs SAT**

X-MiniSAT (2014) [1] : solveur SAT de type CDCL basé sur nos derniers résultats sur les stratégies d'élimination de clauses apprises. Sur la instances de la dernière compétition internationale SAT 2013, X-MiniSAT obtient des résultats significativement meilleurs (sur les instances de la catégorie Application) que les vainqueurs de cette compétition. X-MiniSAT peut être considéré comme un solveur état-de-l'art.
ManySAT [2] : un solveur SAT de type CDCL incluant toutes les fonctionnalités des solveurs SAT modernes (analyse de conflits, heuristique basée sur les activités, etc.) et celles issues de nos résultats sur SAT et sur la résolution parallèle de SAT.
Transferts : ManySAT est utilisé en
— Cryptography : Macquarie University, Australia

1. http ://www.cril.univ-artois.fr/ Saïs/X-SAT/X-MiniSAT.html
2. http ://www.msr-inria.inria.fr/ jabbour/manysat.htm

— Parity games : University of Munich
— Bounded Model Checking : IBM Germany Research & Development GmbH in Boeblingen
— Software Verification : ManySAT adapté et intégré dans Microsoft Z3 SMT solver, Microsoft Redmonds, USA

Distinctions : (Résultats des différentes éditions de la compétition internationale et de la SAT race : http ://www.satcompetition.org/)
— Meilleur solveur SAT parallèle (médaille d'or) à la SAT Race 2008 (track parallèle), Guangzhou, Chine, 2008
— Jury special prize (1st rank) Compétition SAT 2009, Swansea, United Kingdom, 2009
— Médaille d'argent et de bronze à la SAT Race 2010, Edinburgh, Scotland, UK, 2010.

LySAT [3] : solveur SAT de type CDCL incluant toute les fonctionnalités des solveurs SAT modernes (analyse de conflits, heuristique basée sur les activités, etc.) et celles issues de nos résultats récents sur l'amélioration des techniques d'analyse de conflits, subsumption dynamique, restarts dynamique, etc.
Distinction : 2 médailles de bronze à la compétition SAT 2009 (track séquentiel) [4]

— **Outils de Fouille de Données** [5]
Sym4IM : Détection et élimination des symétries dans les problèmes de fouille d'itemsets.
paraSATMiner : solveur parallèle pour l'extraction des itemsets fréquents fermée.
SatMax : solveur pour l'extraction des itemsets fréquents maximaux.
Sat4Seq : librairie pour la recherche de motifs intéressants en utilisant la programmation par contraintes et SAT.
MineToCompress : outil utilisant la fouille d'itemsets pour compresser des formules CNF en utilisant le principe d'extension de Tseitin.

8.1.3 Distinctions : compétitions de solveurs

— ManySAT 1.1 et 1.5 : Médaille d'argent et médaille de bronze à la SAT Race 2010, Edinburgh, Scotland, UK, 2010 (Track parallèle)
— LySAT : 2 médailles de bronze, Competition internationale SAT2009, Swansea, Wales, UK, 2009 (Track séquentiel)
— ManySAT : prix du meilleur solveur SAT parallèle (prix spécial du jury), Competition internationale SAT2009, Swensee, UK, 2009 (Track parallèle)
— ManySAT : prix du meilleur solveur SAT parallèle (Médaille d'or), SAT Race 2008, Guangzhou, China, 2008

8.1.4 Brevets

— "Controlled constraint sharing in parallel problem solvers" [6], en collaboration avec Microsoft Research, Cambridge, UK.

3. http ://www.msr-inria.inria.fr/ jabbour/ (éxecutable disponible)
4. http ://www.cs.swan.ac.uk/ csoliver/SAT2009/
5. http ://www.cril.univ-artois.fr/decMining/
6. https ://patents.google.com/patent/US8346704

8.1.5 Collaboration industriels (passées et en cours)

— **Microsoft Research Cambridge (UK)**
La collaboration entre le CRIL et Microsoft a donnée lieu à de nombreux résultats de recherche incluant la mise en oeuvre du premier solveur SAT parallèlle de type portfolio ManySAT ayant dominé les compétitions internationales de 2008 à 2010. Le solveur ManySAT est intégré dans Z3, le solveur SMT de Microsoft Research Redmonds. Ce résultat a par exemple fait la une du site Microsoft Research News & Highlights [7]. Il est à noter que de nombreux solveurs parallèles de type portfolio ont été ensuite mis en oeuvre.

Bourse de thèse : Financement : Microsoft Research Cambridge, UK. Durée : 3 ans (septembre 2009 - septembre 2012). Budget : 100K Euros. Thèse soutenue à l'Université d'Artois par Long Guo (Co-encadrée avec Lakhdar Saïs)

— Membre **Projet PAJERO** soutenu par le programme, "Innovation Stratégique Industrielle" d'OSEO, (2011-2015).
Ce projet comprend 3 laboratoires (PRISM (Versailles), I3S (NICE) et CRIL (Lens)) ainsi que 3 entreprises (HSW, ICAPS et EquiTime). L'objectif consiste à développer une plateforme logicielle permettant, la gestion complexe de ressources multiples (ressources humaines, planification, gestion des temps et activités associées).

— **RATP - ING/STF/QS/AQL - Paris** : Département Ingénierie, Unité Systèmes du Transport Ferroviaire, Entité Qualification des Systèmes, Atelier de Qualification des Logiciels. Une collaboration est actuellement en cours avec la RATP (Paris), autour des outils de model checking. Le but du projet est d'intégrer nos solveurs SAT dans l'atelier de preuve de la RATP.
Objectifs : Preuve formelle pour l'évaluation des systèmes de sécurité ferroviaire et notamment leur partie logicielle. La preuve formelle permet d'explorer l'exhaustivité des comportements d'un système et, ainsi, d'améliorer significativement le niveau de confiance dans la sécurité. Participants RATP : David Bonvoisin, Responsable de l'Atelier de Qualification Logicielle du département ING et Nazim Benaissa, Responsable d'études en sécurité ferroviaire au sein du laboratoire AQL.
Participants CRIL : Yakoub Salhi, Jean-François Condotta, Saïd Jabbour et Lakhdar Saïs.
Durée : 2 ans, 2015 - 2017. Budget : 150K Euros (Hors Taxes)

8.1.6 Projets de recherche nationaux

— Membre au niveau CRIL, du projet Mastodons - CNRS, *QDoSSI : Qualité des données multi-Sources : un double défi pour les sciences sociales et les sciences de l'Informatique*, janvier 2016 - décembre 2018.
— Membre au niveau CRIL, du projet ANR programme Défis *DAG : Approche déclarative pour énumérer des motifs intéressants*, janvier 2010 - décembre 2013.

7. http ://research.microsoft.com/news/featurestories/publish/ManySAT.aspx ?0hp= n1

— Co-Responsable du projet *Bonus Qualité Recherche (BQR)*, "Vers un calcul efficace des mesures d'incohérence" CRIL, Université d'Artois, 2014-2015.
— Co-Responsable du projet *Bonus Qualité Recherche (BQR)*, "Raisonnement efficace en logique modale", CRIL, Université d'Artois, 2011-2013.

8.2 Encadrement de la recherche et animation scientifique

8.2.1 Activités d'encadrement

Sommaire

— Co-Encadrement de 4 thèses soutenues
— Co-encadrement de 4 thèses en cours dont 3 en Co-Tutelle
— Encadrement de 8 Masters recherche
— Co-Encadrement d'un postdoctorant (Yakoub Salhi) dans le cadre du projet ANR DAG, 01 octobre 2011 au 31 septembre 2012
— Co-Encadrement d'un postdoctorant (Mehdi Khiarri) dans le cadre du projet ANR DAG, 01 septembre 2012 au 31 juillet 2013
— Co-Encadrement d'un postdoctorant (Florian Legendre) dans le cadre du projet ANR TUPLES, 01 octobre 2014 au 30 avril 2015

Co-Encadrement (en cours)

1. Ikram Nakkache Mana, depuis Janvier 2018. *Approches déclaratives pour la fouille de motifs séquentiels* (Co-Tutelle avec l'Algérie)
2. Fatima Zahra Mana, depuis Janvier 2017. *Les systèmes de reconnaissances basées sur la fusion de données multi-biométriques* (Co-Tutelle le Maroc)
3. Nizar Mhadhbi, depuis octobre 2015. *Fouilles de graphes et applications aux réseaux sociaux* (Bourse du gouvernement tunisien). (Soutenance fixée : décembre 2018)
4. Imen Ouled Dlala, depuis septembre 2015. *Enumération de motifs dans les données incertaines.* (Co-Tutelle avec la Tunisie) (Soutenance 12 décembre 2018)

Thèses soutenues

1. Abdelhamid Boudane, septembre 2015 - août 2018. Fouilles de données par Contraintes (Bourse du gouvernement d'Algérie). *Soutenance* : septembre 2018.
2. Jerry Lonlac, septembre 2010 - octobre 2014. Contribution à la résolution du problème de la Satisfiabilité Propositionnelle (SAT) (financée par le MAE). *Soutenance* : octobre 2014
3. Benoît Hoessen, septembre 2011 - octobre 201. Solving the Boolean Satisfiability problem using the parallel paradigm (financée par le Projet Pajero). *Soutenance* : octobre 2014
4. Long Guo, septembre 2009 - Juillet 2013. Résolution séquentielle et parallèle du problème de la satisfiabilité propositionnelle (financement par Microsoft research Cambridge (UK)). *Soutenance* : décembre 2013

Master recherche

1. Amal El Hidouri, février 2018 - juillet 2018, *Approches contraintes pour la fouille des itemsets high utility*, 2018.
2. Abderrahim Rachid, février 2018 - juillet 2018, *Fouille de données textuelles*, 2018.
3. Abdesattar Mhadhbi, février 2017 - juillet 2017, *Approches logiques pour la compression des grands graphes*, 2017.
4. Soukaine Hattad, février 2016 - juillet 2016, *Résolution des problèmes d'optimisation en SAT*, 2016.
5. Abdelhamid Boudane, février 2015 - juillet 2015, *Fouille de données symbolique*, 2015.
6. Rym Meghnous, février 2014 - juillet 2014, *automatisation des systèmes de preuves étendus*, 2014.
7. Fares Boudraa, février 2014 - juillet 2014, *Symétries en fouille de séquences*, 2014.
8. Essid Mohamed, février 2014 - juillet 2014, *Exploitation des classes traitables pour la simplification des formules booléennes*, 2011.

Internship & Post-doc

1. Florian Legendre, septembre 2014 - avril 2015, *Fouille de données par contraintes*, Post-doc (6 mois), projet TUPLES
2. Mehdi Khiarri, septembre 2012 - Août 2013, *Fouille de données par contraintes*, Post-doc (11 mois), projet DAG
3. Yakoub Salhi, octobre 2011 - septembre 2012, *Modèles logiques pour la fouille de données*, Post-doc (12 mois), projet DAG

8.2.2 Membre de comités de programme

— Membre du comité de programme de "*The 23rd International Conference on Artificial Intelligence*", IJCAI'2018, août 28 au 1 septembre, 2018, Melbourne, Sweden, 2018
— Membre du comité de programme de "*The 23rd International Conference on Principles and Practice of Constraint Programming*", CP'2017, août 28 au 1 septembre, 2018, Lille, France, 2018
— Membre du comité de programme de "*The 23rd International Conference on Principles and Practice of Constraint Programming*", CP'2017, août 28 au 1 septembre, 2017, Melbourne, Australie, 2017
— Membre du comité de programme de "*The 19th International Conference on Principles and Practice of Constraint Programming*", CP'2013, September 16-20, 2013, Uppsala, Suède, 2013

8.2.3 Relecture d'articles

Revues : Artificial Intelligence Journal 2017-2018, FUNDAMENTA INFORMATICAE 2017, Internationational Journal of Artificial Intelligence Tools 2013, 2014, 2016, Journal of Experimental algorithmics 2015.

Conférences : CP'2013, CP'2018, IJCAI'2018, SAT'2013, CIIA'2013, ICTAI'2013, SAT'2012, JFPC'2011, ICTAI'2010, POS'2010, SAT'2010.

8.2.4 Expertise de projets

— Expert auprès du ministère des affaires étrangères, Projet Chili, 2012

8.2.5 Collaborations internationales et bourses de thèse

— Joao Marques Silva, University College of Dublin, Ireland (2014), "Computing Prime implicant of CNF formulae" (Publié à JELIA'2014).
— Takeaki Uno, National Institute of Informatics, Tokyo, Japan. Collaboration autour de la compression de formules bolléennes par les approches de fouille de données (Publication commune à CIKM'2013).
— Clémentin Tayou Ndjemini, Université de Schang, Cameroun. Collaboration autours de SAT et plus particulièrement sur l'intégration de la substitution de fonctions booléennes dans les solveurs SAT (Publication commune à la revue RIA en 2014). Cette collaboration a donné lieu au **financement de la thèse de Jerry Lonlac par le ministère des affaires étrangères Français (soutenue en 2014)**
— Youssef Hamadi, Microsoft Research Cambridge (UK) : nos collaborations ont porté sur la résolution parallèle de SAT et sur l'amélioration des solveurs SAT modernes [ECAI'2008, SAT'2008, IJCAI'2009, ICTAI'2009, CP'2010, etc.]. Le solveur ManySAT est né de cette collaboration.
— Lucas Bordeaux, Microsoft Research Cambridge (UK) : nous avons proposé un cadre général pour l'analyse des conflits à l'origine du solveur eSAT, un des meilleurs solveurs SAT actuellement [SAT'2008].

8.2.6 Collaborations nationales

— Badran Raddaoui, Télécom SudParis, SAMOVAR - CNRS UMR 5157 (CP'18, IDA'18, PAKDD'17, BigData'16, etc.)
— Salima Benbernou, université Paris Descartes, LIPADE. "Relaxation based SaaS for Repairing Failed Queries over the Cloud Computing" (Publié à ICEBE 2015)
— Yue Ma, Université Paris Sud, Orsay, LRI : Plusieurs publications autours des mesures d'inconsistances (ECAI'2014, AAMAS'2015)
— Belaid Benhamou, Université d'Aix-Marseille, LSIS : Lors de la délégation CNRS de Belaid Benhamou au CRIL (2013-2014), nous avons travaillé sur les symétries en fouille de données (publication KDIR'2014)
— Emmanuel Coquery, Université de Lyon I : Fouille de séquences par la programmation par contraintes - ANR DAG (ECAI'12, Wks ICDM'11)

8.2.7 Responsabilités collectives

— Membre du COS pour le recrutement d'un maître de conférences (MCF), Université d'Artois, 2016.
— Membre de la commission API pour le recrutement des Attachés Temporaires d'Enseignement et de Recherche (ATER), 2014-2016.
— Membre du conseil du laboratoire CRIL, 2013-2017.

— Co-responsable des emplois de temps, 2014-2018.

Bibliographie

[Acuña *et al.* 2012] Vicente Acuña, Paulo Vieira Milreu, Ludovic Cottret, Alberto Marchetti-Spaccamela, Leen Stougie et Marie-France Sagot. Algorithms and complexity of enumerating minimal precursor sets in genome-wide metabolic networks. Bioinformatics, vol. 28, no. 19, pages 2474–2483, 2012. (Cité en page 13.)

[Adamcsek *et al.* 2006] Balázs Adamcsek, Gergely Palla, Illés J. Farkas, Imre Derényi et Tamás Vicsek. CFinder : locating cliques and overlapping modules in biological networks. Bioinformatics, vol. 22, no. 8, pages 1021–1023, 2006. (Cité en page 41.)

[Agrawal & Srikant 1994] Rakesh Agrawal et Ramakrishnan Srikant. Fast Algorithms for Mining Association Rules in Large Databases. Dans Proceedings of VLDB'94, pages 487–499, 1994. (Cité en page 21.)

[Alba 1973] Richard D. Alba. A graph-theoretic definition of a sociometric clique. Journal of Mathematical Sociologys, vol. 3, pages 113–126, 1973. (Cité en pages 41 et 42.)

[Aloul *et al.* 2003a] F. A. Aloul, A. Ramani, I. L. Markov et K. A. Sakallah. Solving difficult instances of Boolean satisfiability in the presence of symmetry. IEEE Trans. on CAD of Integrated Circuits and Systems, vol. 22, no. 9, pages 1117–1137, 2003. (Cité en page 7.)

[Aloul *et al.* 2003b] Fadi A. Aloul, Arathi Ramani, Igor L. Markov et Karem A. Sakallah. Efficient Symmetry-Breaking for Boolean Satisfiability. Dans Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, 2003. (Cité en page 7.)

[Andrews 1968] Peter B. Andrews. Resolution With Merging. Journal of the ACM, vol. 15, no. 3, pages 367–381, Juillet 1968. (Cité en page 8.)

[Arimura & Uno 2007] Hiroki Arimura et Takeaki Uno. An efficient polynomial space and polynomial delay algorithm for enumeration of maximal motifs in a sequence. Journal of Combinatorial Optimization, vol. 13, 2007. (Cité en page 32.)

[Asín *et al.* 2009] Roberto Asín, Robert Nieuwenhuis, Albert Oliveras et Enric Rodríguez-Carbonell. Cardinality Networks and Their Applications. Dans 12th International Conference on Theory and Applications of Satisfiability Testing (SAT 2009), pages 167–180, 2009. (Cité en page 15.)

[Audemard & Simon 2009] Gilles Audemard et Laurent Simon. Predicting Learnt Clauses Quality in Modern SAT Solvers. Dans IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009, pages 399–404, 2009. (Cité en pages 9 et 11.)

[Audemard *et al.* 2007] Gilles Audemard, Saïd Jabbour et Lakhdar Sais. Symmetry Breaking in Quantified Boolean Formulae. Dans IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007, pages 2262–2267, 2007. (Cité en page 7.)

[Audemard *et al.* 2011] Gilles Audemard, Jean-Marie Lagniez, Bertrand Mazure et Lakhdar Sais. On Freezing and Reactivating Learnt Clauses. Dans Theory and Applications of

Satisfiability Testing - SAT 2011 - 14th International Conference, SAT 2011, Ann Arbor, MI, USA, June 19-22, 2011. Proceedings, pages 188–200, 2011. (Cité en page 10.)

[Audemard *et al.* 2012] Gilles Audemard, Benoît Hoessen, Saïd Jabbour, Jean-Marie Lagniez et Cédric Piette. Revisiting Clause Exchange in Parallel SAT Solving. Dans Theory and Applications of Satisfiability Testing - SAT 2012 - 15th International Conference, Trento, Italy, June 17-20, 2012. Proceedings, pages 200–213, 2012. (Cité en pages 9 et 10.)

[Audemard *et al.* 2014] Gilles Audemard, Benoît Hoessen, Saïd Jabbour et Cédric Piette. An Effective Distributed D&C Approach for the Satisfiability Problem. Dans 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, PDP 2014, Torino, Italy, February 12-14, 2014, pages 183–187, 2014. (Cité en page 12.)

[Baader *et al.* 2010] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi et Peter F. Patel-Schneider. The description logic handbook : Theory, implementation and applications. Cambridge University Press, New York, NY, USA, 2nd édition, 2010. (Cité en page 52.)

[Bailleux *et al.* 2009] O. Bailleux, Y. Boufkhad et O. Roussel. New Encodings of Pseudo-Boolean Constraints into CNF. Dans SAT'2009, pages 181–194, 2009. (Cité en page 15.)

[Balbiani *et al.* 2010] Philippe Balbiani, Fahima Cheikh Alili, P.-C. Héam et Olga Kouchnarenko. Composition of Services with Constraints. Electronic Notes in Theoretical Computer Science, vol. 263, no. 0, pages 31 – 46, 2010. Proceedings of the 6th International Workshop on Formal Aspects of Component Software (FACS 2009). (Cité en page 53.)

[Bastide *et al.* 2000] Yves Bastide, Nicolas Pasquier, Rafik Taouil, Gerd Stumme et Lotfi Lakhal. Mining Minimal Non-redundant Association Rules Using Frequent Closed Itemsets. Dans Computational Logic - CL 2000, volume 1861, pages 972–986, 2000. (Cité en page 31.)

[Benhamou & Sais 1994] Belaid Benhamou et Lakhdar Sais. Tractability through Symmetries in Propositional Calculus. Journal of Automated Reasoning, vol. 12, no. 1, pages 89–102, Février 1994. (Cité en page 7.)

[Besnard 2014] Philippe Besnard. Revisiting Postulates for Inconsistency Measures. Dans JELIA, pages 383–396, 2014. (Cité en page 46.)

[Billard & Diday 2012] Lynne Billard et Edwin Diday. Symbolic data analysis : Conceptual statistics and data mining. John Wiley & Sons, May 2012. (Cité en page 35.)

[Bistarelli & Bonchi 2007] Stefano Bistarelli et Francesco Bonchi. Soft constraint based pattern mining. Data & Knowledge Engineering, vol. 62, no. 1, pages 118 – 137, 2007. (Cité en page 14.)

[Böhm & Speckenmeyer 1996] Max Böhm et Ewald Speckenmeyer. A Fast Parallel SAT-Solver - Efficient Workload Balancing. Ann. Math. Artif. Intell., vol. 17, no. 3-4, pages 381–400, 1996. (Cité en page 9.)

[Borgelt 2012] Christian Borgelt. Frequent item set mining. Wiley Interdisc. Rew. : Data Mining and Knowledge Discovery, vol. 2, no. 6, pages 437–456, 2012. (Cité en page 26.)

[Boudane *et al.* 2016] Abdelhamid Boudane, Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. A SAT-Based Approach for Mining Association Rules. Dans Proceedings of IJCAI'16, pages 2472–2478, 2016. (Cité en pages 6, 26, 30 et 31.)

[Boudane *et al.* 2017a] Abdelhamid Boudane, Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. Clustering Complex Data Represented as Propositional Formulas. Dans Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part II, pages 441–452, 2017. (Cité en pages 35 et 36.)

[Boudane *et al.* 2017b] Abdelhamid Boudane, Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. Enumerating Non-redundant Association Rules Using Satisfiability. Dans Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I, pages 824–836, 2017. (Cité en page 31.)

[Boutilier *et al.* 2004] Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, David L. Poole et Holger H. Hoos. CP-nets : A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. Journal of Artificial Intelligence Research (JAIR), vol. 21, pages 135–191, 2004. (Cité en page 14.)

[Bouzeghoub *et al.* 2017] Amel Bouzeghoub, Saïd Jabbour, Yue Ma et Badran Raddaoui. Handling conflicts in uncertain ontologies using deductive argumentation. Dans Proceedings of the International Conference on Web Intelligence, Leipzig, Germany, August 23-26, 2017, pages 65–72, 2017. (Cité en page 52.)

[Brafman & Domshlak 2009] R. I. Brafman et Carmel Domshlak. Preference Handling - An Introductory Tutorial. AI Magazine, vol. 30, no. 1, pages 58–86, 2009. (Cité en pages 14 et 29.)

[Burdick *et al.* 2001] Doug Burdick, Manuel Calimlim et Johannes Gehrke. MAFIA : A maximal frequent itemset algorithm for transactional databases. Dans In ICDE, pages 443–452, 2001. (Cité en page 25.)

[Cadoli & Donini 1997] Marco Cadoli et Francesco M. Donini. A Survey on Knowledge Compilation. AI Commun., vol. 10, no. 3-4, pages 137–150, 1997. (Cité en page 13.)

[Cai *et al.* 1998] C. H. Cai, Ada W. C. Fu, C. H. Cheng et W. W. Kwong. Mining Association Rules with Weighted Items. Dans In Proc. of IDEAS Symp, pages 68–77, 1998. (Cité en page 29.)

[Castell *et al.* 1996] Thierry Castell, Claudette Cayrol, Michel Cayrol et Daniel Le Berre. Using the Davis and Putnam Procedure for an Efficient Computation of Preferred Models. Dans ECAI'96, pages 350–354, 1996. (Cité en page 14.)

[Chauhan *et al.* 2003] Pankaj Chauhan, Edmund M. Clarke et Daniel Kroening. Using SAT based image computation for reachability analysis. Rapport technique, Technical Report CMU-CS-03-151, 2003. (Cité en page 12.)

[Chiu & Jain 1989] D-M Chiu et R. Jain. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. Comp. Networks, vol. 17, pages 1–14, 1989. (Cité en page 10.)

[Chrabakh & Wolski 2003] Wahid Chrabakh et Richard Wolski. GridSAT : A Chaff-based Distributed SAT Solver for the Grid. Dans Proceedings of the ACM/IEEE SC2003 Conference on High Performance Networking and Computing, 15-21 November 2003, Phoenix, AZ, USA, CD-Rom, page 37, 2003. (Cité en pages 9 et 12.)

[Chu *et al.* 2008] Geoffrey Chu, Peter J. Stuckey et Aaron Harwood. PMiniSAT : A Parallelization of MiniSAT 2.0. Rapport technique, 2008. (Cité en pages 9 et 12.)

[Chui & Kao 2008] Chun Kit Chui et Ben Kao. A Decremental Approach for Mining Frequent Itemsets from Uncertain Data. Dans Proceedings of the Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, pages 64–75, 2008. (Cité en page 29.)

[Chui *et al.* 2007] Chun-Kit Chui, Ben Kao et Edward Hung. Mining Frequent Itemsets from Uncertain Data. Dans Proceedings of the Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, PAKDD'07, pages 47–58, 2007. (Cité en page 29.)

[Clarke *et al.* 2001] Edmund M. Clarke, Armin Biere, Richard Raimi et Yunshan Zhu. Bounded Model Checking Using Satisfiability Solving. Formal Methods in System Design, vol. 19, no. 1, pages 7–34, 2001. (Cité en page 6.)

[Cohen *et al.* 2006] David A. Cohen, Peter Jeavons, Christopher Jefferson, Karen E. Petrie et Barbara M. Smith. Symmetry Definitions for Constraint Satisfaction Problems. Constraints, vol. 11, no. 2-3, pages 115–137, 2006. (Cité en page 7.)

[Cohen 2008] J. Cohen. Trusses : Cohesive subgraphs for social network analysis. Dans Technical report, National Security Agency, 2008. (Cité en page 42.)

[Cook 1971] S. A. Cook. The complexity of theorem-proving procedures. Dans Proceedings of the Third Annual ACM Symposium on Theory of Computing, pages 151–158, New York (USA), 1971. Association for Computing Machinery. (Cité en page 6.)

[Coquery *et al.* 2012] Emmanuel Coquery, Saïd Jabbour, Lakhdar Saïs et Yakoub Salhi. A SAT-Based Approach for Discovering Frequent, Closed and Maximal Patterns in a Sequence. Dans ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012, pages 258–263, 2012. (Cité en page 33.)

[Coudert & Madre 1993] O. Coudert et J.C. Madre. Fault tree analysis : 10^{20} prime implicants and beyond. Dans Reliability and Maintainability Symposium, 1993. Proceedings., Annual, pages 240–245, Jan 1993. (Cité en page 13.)

[Crawford & Baker 1994] James M. Crawford et Andrew B. Baker. Experimental Results on the Application of Satisfiability Algorithms to Scheduling Problems. Dans Proceedings of the Twelfth National Conference on Artificial Intelligence (Vol. 2), AAAI'94, pages 1092–1097, 1994. (Cité en page 6.)

[Crawford *et al.* 1996] James M. Crawford, Matthew L. Ginsberg, Eugene M. Luks et Amitabha Roy. Symmetry-Breaking Predicates for Search Problems. Dans Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR'96), Cambridge, Massachusetts, USA, November 5-8, 1996., pages 148–159, 1996. (Cité en page 7.)

[Crawford 1992] James Crawford. A theorical Analysis of Reasoning by Symmetry in First Order Logic. Dans Proceedings of Workshop on Tractable Reasonning, AAAI92, pages 17–22, Juillet 1992. (Cité en page 7.)

[Creignou *et al.* 2011] Nadia Creignou, Frédéric Olive et Johannes Schmidt. Enumerating All Solutions of a Boolean CSP by Non-decreasing Weight. Dans 14th International Conference on Theory and Applications of Satisfiability Testing (SAT'11), pages 120–133, 2011. (Cité en page 13.)

[Dao *et al.* 2013] Thi-Bich-Hanh Dao, Khanh-Chuong Duong et Christel Vrain. A Declarative Framework for Constrained Clustering. Dans Proceedings of ECML PKDD'13, pages 419–434, 2013. (Cité en page 21.)

[Darwiche & Marquis 2002] Adnan Darwiche et Pierre Marquis. A Knowledge Compilation Map. J. Artif. Intell. Res. (JAIR), vol. 17, pages 229–264, 2002. (Cité en page 13.)

[Davidson *et al.* 2010] Ian Davidson, S. S. Ravi et Leonid Shamis. A SAT-based Framework for Efficient Constrained Clustering. Dans Proceedings of SDM'10, pages 94–105, 2010. (Cité en page 21.)

[Davis *et al.* 1962] M. Davis, G. Logemann et D. W. Loveland. A machine program for theorem-proving. Communications of the ACM, vol. 5, no. 7, pages 394–397, 1962. (Cité en page 6.)

[de Amo *et al.* 2012] Sandra de Amo, Mouhamadou Saliou Diallo, Cheikh Talibouya Diop, Arnaud Giacometti, Haoyuan D. Li et Arnaud Soulet. Mining Contextual Preference Rules for Building User Profiles. Dans Proceedings of the 14th International Conference on Data Warehousing and Knowledge Discovery, DaWaK'12, pages 229–242, 2012. (Cité en page 14.)

[de Kleer *et al.* 1990] Johan de Kleer, Alan K. Mackworth et Raymond Reiter. Characterizing Diagnoses. Dans Proceedings of the 8th National Conference on Artificial Intelligence (AAAI'90), pages 324–330, 1990. (Cité en page 13.)

[Dechter & Itai 1992] Rina Dechter et Alon Itai. Finding All Solutions if You can Find One. Dans In AAAI-92 Workshop on Tractable Reasoning, pages 35–39, 1992. (Cité en page 13.)

[del Val 1994] A. del Val. Tractable Databases : How to Make Propositional Unit Resolution Complete Through Compilation. Dans Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning (KR'94), pages 551–561, 1994. (Cité en page 13.)

[Desrosiers *et al.* 2007] Christian Desrosiers, Philippe Galinier, Pierre Hansen et Alain Hertz. Improving Frequent Subgraph Mining in the Presence of Symmetry. Dans Proceedings of MLG'2007, 2007. (Cité en page 27.)

[Diday & Esposito 2003] Edwin Diday et Floriana Esposito. An introduction to symbolic data analysis and the SODAS software. Intell. Data Anal., vol. 7, no. 6, pages 583–601, 2003. (Cité en page 35.)

[Dlala *et al.* 2010] Imen Ouled Dlala, Said Jabbour, Badran Raddaoui et Lakhdar Sais. A SAT Based Framework for Itemset Mining in Parallel. Dans In Proceedings of CP'18, pages 552–567. Springer, 2010. (Cité en page 27.)

[Dlala *et al.* 2015] Imen Ouled Dlala, Saïd Jabbour, Lakhdar Sais, Yakoub Salhi et Boutheina Ben Yaghlane. Parallel SAT based closed frequent itemsets enumeration. Dans 12th IEEE/ACS International Conference of Computer Systems and Applications, AICCSA 2015, Marrakech, Morocco, November 17-20, 2015, pages 1–8, 2015. (Cité en page 27.)

[Dlala *et al.* 2016a] Imen Ouled Dlala, Saïd Jabbour, Badran Raddaoui, Lakhdar Sais et Boutheina Ben Yaghlane. A SAT-Based Approach for Enumerating Interesting Patterns from Uncertain Data. Dans 28th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2016, San Jose, CA, USA, November 6-8, 2016, pages 255–262, 2016. (Cité en page 30.)

[Dlala *et al.* 2016b] Imen Ouled Dlala, Saïd Jabbour, Lakhdar Saïs et Boutheina Ben Yaghlane. A Comparative Study of SAT-Based Itemsets Mining. Dans Research and Development in Intelligent Systems XXXIII - Incorporating Applications and Innovations in Intelligent Systems XXIV. Proceedings of AI-2016, The Thirty-Sixth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, Cambridge, UK, December 13-15, 2016, pages 37–52, 2016. (Cité en page 25.)

[Dlala *et al.* 2018] Imen Ouled Dlala, Saïd Jabbour, Badran Raddaoui et Lakhdar Sais. A Parallel SAT-Based Framework for Closed Frequent Itemsets Mining. Dans Principles and Practice of Constraint Programming - 24th International Conference, CP 2018, Lille, France, August 27-31, 2018, Proceedings, pages 570–587, 2018. (Cité en page 26.)

[Domshlak *et al.* 2011] Carmel Domshlak, Eyke Hüllermeier, Souhila Kaci et Henri Prade. Preferences in AI : An overview. Artificial Intelligence, vol. 175, no. 7-8, pages 1037–1052, 2011. (Cité en pages 14 et 29.)

[Dung 1995] Phan Minh Dung. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Persson Games. Artif. Intell., vol. 77, no. 2, pages 321–358, 1995. (Cité en page 51.)

[Dutuit & Rauzy 1997] Y. Dutuit et A. Rauzy. Exact and Truncated Computations of Prime Implicants of Coherent and non-Coherent Fault Trees within Aralia. Reliability Engineering and System Safety, vol. 58, no. 2, pages 127–144, 1997. (Cité en page 13.)

[Eén & Biere 2005] Niklas Eén et Armin Biere. Effective Preprocessing in SAT Through Variable and Clause Elimination. Dans SAT, volume 3569 de Lecture Notes in Computer Science, pages 61–75. Springer, 2005. (Cité en page 18.)

[Eén & Sörensson 2006] N. Eén et N. Sörensson. Translating Pseudo-Boolean Constraints into SAT. JSAT, vol. 2, no. 1-4, pages 1–26, 2006. (Cité en page 15.)

[Eiter 1996] Thomas Eiter. Open Problems in Satisfiability. Dans http ://www.ece.uc.edu/ franco/Sat-workshop/sat-workshop-open-problems.html, 1996. (Cité en page 50.)

[Ester *et al.* 1996] Martin Ester, Hans-Peter Kriegel, Jörg Sander et Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Dans Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA, pages 226–231, 1996. (Cité en page 36.)

[Fournier-Viger & Tseng 2013] Philippe Fournier-Viger et Vincent S. Tseng. TNS : mining top-k non-redundant sequential rules. Dans Proceedings of SAC '13, pages 164–166, 2013. (Cité en page 31.)

[Fournier-Viger *et al.* 2014] Philippe Fournier-Viger, Antonio Gomaric, Ted Gueniche, Azadeh Soltani, Cheng-Wei Wu et Vincent S Tseng. SPMF : a java open-source pattern mining

library. The Journal of Machine Learning Research, vol. 15, no. 1, pages 3389–3393, 2014. (Cité en page 31.)

[Fu *et al.* 2000] Ada Wai-Chee Fu, Renfrew W. w. Kwong et Jian Tang. Mining *N*-most Interesting Itemsets. Dans Proceedings of the 12th International Symposium on MÈthodologies for Intelligent Systems (ISMIS 2000), Lecture Notes in Computer Science, pages 59–67. Springer, 2000. (Cité en page 28.)

[Gasmi *et al.* 2005] Ghada Gasmi, Sadok Ben Yahia, Engelbert Mephu Nguifo et Yahya Slimani. IGB : A New Informative Generic Base of Association Rules. Dans Proceedings of PAKDD'05, pages 81–90, 2005. (Cité en page 31.)

[Gebser *et al.* 2016] Martin Gebser, Thomas Guyet, René Quiniou, Javier Romero et Torsten Schaub. Knowledge-Based Sequence Mining with ASP. Dans IJCAI, pages 1497–1504. IJCAI/AAAI Press, 2016. (Cité en page 21.)

[Gent & Smith 2000] Ian P. Gent et Barbara Smith. Symmetry breaking in constraint programming. Dans Proceedings of the Thirteenth European Conference on Artificial Intelligence(ECAI'00), pages 599–603, 2000. (Cité en page 7.)

[Ginsberg 1989] MatthewL. Ginsberg. A circumscriptive theorem prover. Dans M. Reinfrank, J. Kleer, M.L. Ginsberg et E. Sandewall, editeurs, Non-Monotonic Reasoning, volume 346 de Lecture Notes in Computer Science, pages 100–114. Springer Berlin Heidelberg, 1989. (Cité en page 13.)

[Gouda & Zaki 2005] Karam Gouda et Mohammed Javeed Zaki. GenMax : An Efficient Algorithm for Mining Maximal Frequent Itemsets. Data Min. Knowl. Discov., vol. 11, no. 3, pages 223–242, 2005. (Cité en page 25.)

[Grant & Hunter 2008] John Grant et Anthony Hunter. Analysing inconsistent first-order knowledgebases. Artif. Intell., vol. 172, no. 8-9, pages 1064–1093, 2008. (Cité en page 46.)

[Grant & Hunter 2011] John Grant et Anthony Hunter. Measuring the Good and the Bad in Inconsistent Information. Dans IJCAI, pages 2632–2637, 2011. (Cité en page 45.)

[Grégoire *et al.* 2004] Éric Grégoire, Richard Ostrowski, Bertrand Mazure et Lakhdar Sais. Automatic Extraction of Functional Dependencies. Dans SAT 2004 - The Seventh International Conference on Theory and Applications of Satisfiability Testing, 10-13 May 2004, Vancouver, BC, Canada, Online Proceedings, 2004. (Cité en page 8.)

[Gregory 2009] Steve Gregory. Finding Overlapping Communities Using Disjoint Community Detection Algorithms. Dans International Workshop on Complex Networks, pages 47–61, 2009. (Cité en page 39.)

[Guo *et al.* 2010] Long Guo, Youssef Hamadi, Saïd Jabbour et Lakhdar Sais. Diversification and Intensification in Parallel SAT Solving. Dans Principles and Practice of Constraint Programming - CP 2010 - 16th International Conference, CP 2010, St. Andrews, Scotland, UK, September 6-10, 2010. Proceedings, pages 252–265, 2010. (Cité en page 11.)

[Guo *et al.* 2014] Long Guo, Saïd Jabbour, Jerry Lonlac et Lakhdar Sais. Diversification by Clauses Deletion Strategies in Portfolio Parallel SAT Solving. Dans 26th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2014, Limassol, Cyprus, November 10-12, 2014, pages 701–708, 2014. (Cité en page 11.)

[Hamadi *et al.* 2009] Youssef Hamadi, Saïd Jabbour et Lakhdar Sais. Control-Based Clause Sharing in Parallel SAT Solving. Dans IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009, pages 499–504, 2009. (Cité en pages 8, 9 et 10.)

[Hamadi *et al.* 2011] Youssef Hamadi, Saïd Jabbour, Cédric Piette et Lakhdar Sais. Deterministic Parallel DPLL. JSAT, vol. 7, no. 4, pages 127–132, 2011. (Cité en pages 10 et 11.)

[Hamadi *et al.* 2012] Youssef Hamadi, Said Jabbour et Lakhdar Sais. Learning from conflicts in propositional satisfiability. 4OR, vol. 10, no. 1, pages 15–32, 2012. (Cité en page 8.)

[Hamadi *et al.* 2016] Youssef Hamadi, Saïd Jabbour et Lakhdar Saïs. What we can learn from conflicts in propositional satisfiability. Annals OR, vol. 240, no. 1, pages 13–37, 2016. (Cité en page 8.)

[Han *et al.* 2002] Jiawei Han, Jianyong Wang, Ying Lu et Petre Tzvetkov. Mining Top-K Frequent Closed Patterns without Minimum Support. Dans Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), pages 211–218. IEEE Computer Society, 2002. (Cité en page 28.)

[Hattad *et al.* 2017] Soukaina Hattad, Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. Enhancing Pigeon-Hole based Encoding of Boolean Cardinality Constraints. Dans Proceedings of the 9th International Conference on Agents and Artificial Intelligence, ICAART 2017, Volume 2, Porto, Portugal, February 24-26, 2017., pages 299–307, 2017. (Cité en page 16.)

[Henriques *et al.* 2012] Rui Henriques, Inês Lynce et Vasco M. Manquinho. On When and How to use SAT to Mine Frequent Itemsets. CoRR, vol. abs/1207.6253, 2012. (Cité en page 24.)

[Heule *et al.* 2018] Marijn J. H. Heule, Oliver Kullmann et Armin Biere. Cube-and-Conquer for Satisfiability. Dans Handbook of Parallel Constraint Reasoning., pages 31–59. 2018. (Cité en page 9.)

[Hotz *et al.* 2014] Lothar Hotz, Alexander Felfernig, Markus Stumptner, Anna Ryabokon, Claire Bagley et Katharina Wolter. Chapter 6 - Configuration Knowledge Representation and Reasoning. Dans Knowledge-Based Configuration, pages 41 – 72. Morgan Kaufmann, 2014. (Cité en page 35.)

[Huan *et al.* 2003] Jun Huan, Wei Wang et Jan Prins. Efficient Mining of Frequent Subgraphs in the Presence of Isomorphism. Dans Proceedings of the 3rd IEEE ICDM'2003, pages 549–552, 2003. (Cité en page 27.)

[Hunter & Konieczny 2005] Anthony Hunter et Sébastien Konieczny. Approaches to Measuring Inconsistent Information. Dans Inconsistency Tolerance, pages 191–236, 2005. (Non cité.)

[Hunter & Konieczny 2010] Anthony Hunter et Sébastien Konieczny. On the measure of conflicts : Shapley Inconsistency Values. Artif. Intell., vol. 174, no. 14, pages 1007–1026, 2010. (Cité en pages 45 et 46.)

[Jabbour & Raddaoui 2013] Saïd Jabbour et Badran Raddaoui. Measuring Inconsistency through Minimal Proofs. Dans ECSQARU, pages 290–301, 2013. (Cité en page 47.)

[Jabbour & Sais 2016] Saïd Jabbour et Lakhdar Sais. Exploiting MUS Structure to Measure Inconsistency of Knowledge Bases. Dans ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016), pages 991–998, 2016. (Cité en pages 49 et 50.)

[Jabbour *et al.* 2012] Saïd Jabbour, Lakhdar Sais, Yakoub Salhi et Karim Tabia. Symmetries in Itemset Mining. Dans ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012, pages 432–437, 2012. (Cité en pages 27 et 28.)

[Jabbour *et al.* 2013a] Saïd Jabbour, Mehdi Khiari, Lakhdar Sais, Yakoub Salhi et Karim Tabia. Symmetry-Based Pruning in Itemset Mining. Dans 25th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2013, Herndon, VA, USA, November 4-6, 2013, pages 483–490, 2013. (Cité en page 28.)

[Jabbour *et al.* 2013b] Saïd Jabbour, Jerry Lonlac et Lakhdar Saïs. Adding New Bi-Asserting Clauses for Faster Search in Modern SAT Solvers. Dans Proceedings of the Tenth Symposium on Abstraction, Reformulation, and Approximation, SARA 2013, 11-12 July 2013, Leavenworth, Washington, USA., 2013. (Cité en page 8.)

[Jabbour *et al.* 2013c] Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. Boolean satisfiability for sequence mining. Dans 22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013, pages 649–658, 2013. (Cité en page 33.)

[Jabbour *et al.* 2013d] Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. A Pigeon-Hole Based Encoding of Cardinality Constraints. TPLP, vol. 13, 2013. (Cité en pages 14, 15, 16 et 24.)

[Jabbour *et al.* 2013e] Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. The Top-k Frequent Closed Itemset Mining Using Top-k SAT Problem. Dans Proceedings of ECML/PKDD'13, pages 403–418, 2013. (Cité en pages 26 et 28.)

[Jabbour *et al.* 2013f] Saïd Jabbour, Lakhdar Sais, Yakoub Salhi et Takeaki Uno. Mining-based compression approach of propositional formulae. Dans 22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013, pages 289–298, 2013. (Cité en page 18.)

[Jabbour *et al.* 2014a] Saïd Jabbour, Jerry Lonlac, Lakhdar Sais et Yakoub Salhi. Extending modern SAT solvers for models enumeration. Dans Proceedings of the 15th IEEE International Conference on Information Reuse and Integration, IRI 2014, Redwood City, CA, USA, August 13-15, 2014, 2014. (Cité en page 13.)

[Jabbour *et al.* 2014b] Saïd Jabbour, Jerry Lonlac, Lakhdar Sais et Yakoub Salhi. Revisiting the Learned Clauses Database Reduction Strategies. CoRR, vol. abs/1402.1956, 2014. (Cité en page 9.)

[Jabbour *et al.* 2014c] Said Jabbour, Yue Ma et Badran Raddaoui. Inconsistency Measurement Thanks to MUS-Decomposition. Dans International conference on Autonomous Agents and Multi-Agent Systems (AAMAS'14), pages 877–884, 2014. (Cité en pages 46 et 47.)

[Jabbour *et al.* 2014d] Saïd Jabbour, Yue Ma, Badran Raddaoui et Lakhdar Sais. On the Characterization of Inconsistency : A Prime Implicates Based Framework. Dans 26th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2014, Limassol, Cyprus, November 10-12, 2014, pages 146–153, 2014. (Cité en pages 49 et 50.)

[Jabbour *et al.* 2014e] Saïd Jabbour, Yue Ma, Badran Raddaoui et Lakhdar Saïs. Prime Implicates Based Inconsistency Characterization. Dans ECAI, pages 1037–1038, 2014. (Cité en page 49.)

[Jabbour *et al.* 2014f] Saïd Jabbour, Joao Marques-Silva, Lakhdar Sais et Yakoub Salhi. Enumerating Prime Implicants of Propositional Formulae in Conjunctive Normal Form. Dans Logics in Artificial Intelligence - 14th European Conference, JELIA 2014, Funchal, Madeira, Portugal, September 24-26, 2014. Proceedings, pages 152–165, 2014. (Cité en page 13.)

[Jabbour *et al.* 2014g] Saïd Jabbour, Lakhdar Saïs et Yakoub Salhi. A Pigeon-Hole Based Encoding of Cardinality Constraints. Dans International Symposium on Artificial Intelligence and Mathematics, ISAIM 2014, Fort Lauderdale, FL, USA, January 6-8, 2014, 2014. (Cité en pages 15 et 16.)

[Jabbour *et al.* 2015a] Saïd Jabbour, Stéphanie Roussel, Lakhdar Sais et Yakoub Salhi. Mining to Compress Table Constraints. Dans 27th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2015, Vietri sul Mare, Italy, November 9-11, 2015, pages 405–412, 2015. (Cité en page 19.)

[Jabbour *et al.* 2015b] Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. Decomposition Based SAT Encodings for Itemset Mining Problems. Dans Proceedings of PAKDD'15, pages 662–674, 2015. (Cité en page 26.)

[Jabbour *et al.* 2016a] Saïd Jabbour, Souhila Kaci, Lakhdar Sais et Yakoub Salhi. Itemset Mining with Penalties. Dans 28th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2016, San Jose, CA, USA, November 6-8, 2016, pages 962–966, 2016. (Cité en page 29.)

[Jabbour *et al.* 2016b] Saïd Jabbour, Yue Ma, Badran Raddaoui, Lakhdar Sais et Yakoub Salhi. A MIS Partition Based Framework for Measuring Inconsistency. Dans Principles of Knowledge Representation and Reasoning : Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016., pages 84–93, 2016. (Cité en page 47.)

[Jabbour *et al.* 2016c] Saïd Jabbour, Nizar Mhadhbi, Abdesattar Mhadhbi, Badran Raddaoui et Lakhdar Sais. Summarizing big graphs by means of pseudo-boolean constraints. Dans 2016 IEEE International Conference on Big Data, BigData 2016, Washington DC, USA, December 5-8, 2016, pages 889–894, 2016. (Cité en pages 43 et 44.)

[Jabbour *et al.* 2016d] Saïd Jabbour, Badran Raddaoui, Lakhdar Sais et Yakoub Salhi. On the Computation of Top-k Extensions in Abstract Argumentation Frameworks. Dans ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016), pages 913–920, 2016. (Cité en page 51.)

[Jabbour *et al.* 2017a] Said Jabbour, Abdelhamid Boudane, Badran Raddaoui et Lakhdar Sais. Efficient SAT-Based Encodings of Conditional Cardinality Constraints. Dans LPAR-22, 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning, 2017. (Cité en page 17.)

[Jabbour *et al.* 2017b] Saïd Jabbour, Yue Ma, Badran Raddaoui et Lakhdar Sais. Quantifying conflicts in propositional logic through prime implicates. Int. J. Approx. Reasoning, vol. 89, pages 27–40, 2017. (Cité en page 49.)

[Jabbour *et al.* 2017c] Saïd Jabbour, Nizar Mhadhbi, Badran Raddaoui et Lakhdar Sais. A SAT-Based Framework for Overlapping Community Detection in Networks. Dans Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part II, pages 786–798, 2017. (Cité en pages 41, 42, 48 et 50.)

[Jabbour *et al.* 2017d] Saïd Jabbour, Lakhdar Sais et Yakoub Salhi. Mining Top-k motifs with a SAT-based framework. Artif. Intell., vol. 244, pages 30–47, 2017. (Cité en pages 14, 28 et 29.)

[Jabbour *et al.* 2018] Said Jabbour, Nizar LMhadhbi, Badran et Raddaoui Lakhdar Sais. Pushing the Envelope in Overlapping Communities Detection. Dans Advances in Intelligent Data Analysis XVI - 16th International Symposium, IDA 2018, Proceedings, volume – de Lecture Notes in Computer Science. Springer, 2018. (Cité en page 42.)

[Jabbour 2009] Saïd Jabbour. Learning for Dynamic Assignments Reordering. Dans ICTAI 2009, 21st IEEE International Conference on Tools with Artificial Intelligence, Newark, New Jersey, USA, 2-4 November 2009, pages 336–343, 2009. (Cité en page 8.)

[Jabbour 2016] Saïd Jabbour. On Inconsistency Measuring and Resolving. Dans ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016), pages 1676–1677, 2016. (Cité en page 51.)

[Jeroslow & Wang 1990] Robert G. Jeroslow et Jinchang Wang. Solving propositional Satisfiability Problems. Annals of Mathematics and Artificial Intelligence, vol. 1, pages 167–187, 1990. (Cité en page 25.)

[Jin *et al.* 2005] Hoonsang Jin, Hyojung Han et Fabio Somenzi. Efficient conflict analysis for finding all satisfying assignments of a boolean circuit. Dans In TACAS'05, LNCS 3440, pages 287–300. Springer, 2005. (Cité en page 12.)

[Jr. 1998] Roberto J. Bayardo Jr. Efficiently Mining Long Patterns from Databases. Dans SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA., pages 85–93, 1998. (Cité en page 25.)

[Jurkowiak *et al.* 2001] Bernard Jurkowiak, Chu Min Li et Gil Utard. Parallelizing Satz Using Dynamic Workload Balancing. Electronic Notes in Discrete Mathematics, vol. 9, pages 174–189, 2001. (Cité en page 9.)

[Kaci 2011] Souhila Kaci. Working with preferences : Less is more. Cognitive Technologies. Springer, 2011. (Cité en page 29.)

[Karypis *et al.* 1999] George Karypis, Eui-Hong Han et Vipin Kumar. Chameleon : Hierarchical Clustering Using Dynamic Modeling. IEEE Computer, vol. 32, no. 8, pages 68–75, 1999. (Cité en page 36.)

[Katsirelos & Walsh 2007] George Katsirelos et Toby Walsh. A Compression Algorithm for Large Arity Extensional Constraints. Dans Proceedings of the 13th International Conference on Principles and Practice of Constraint Programming - CP 2007, volume 4741 de Lecture Notes in Computer Science, pages 379–393. Springer, 2007. (Cité en page 19.)

[Kemmar *et al.* 2014] Amina Kemmar, Willy Ugarte, Samir Loudni, Thierry Charnois, Yahia Lebbah, Patrice Boizumault et Bruno Crémilleux. Mining Relevant Sequence Patterns with CP-Based Framework. Dans 26th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2014, Limassol, Cyprus, November 10-12, 2014, pages 552–559, 2014. (Cité en page 21.)

[Konieczny *et al.* 2003] Sébastien Konieczny, Jérôme Lang et Pierre Marquis. Quantifying information and contradiction in propositional logic through test actions. Dans IJCAI, pages 106–111. Morgan Kaufmann, 2003. (Cité en page 46.)

[Koshimura *et al.* 2012] Miyuki Koshimura, Tong Zhang, Hiroshi Fujita et Ryuzo Hasegawa. QMaxSAT : A Partial Max-SAT Solver. JSAT, vol. 8, no. 1/2, pages 95–100, 2012. (Cité en page 16.)

[Kryszkiewicz 1998a] Marzena Kryszkiewicz. Representative Association Rules. Dans Proceedings of PAKDD'98, pages 198–209, 1998. (Cité en page 31.)

[Kryszkiewicz 1998b] Marzena Kryszkiewicz. Representative Association Rules and Minimum Condition Maximum Consequence Association Rules. Dans Proceedings of PKDD '98, pages 361–369, 1998. (Cité en page 31.)

[Lazaar *et al.* 2012] Nadjib Lazaar, Youssef Hamadi, Saïd Jabbour et Michel sebag. Cooperation control in Parallel SAT Solving : a Multi-armed Bandit Approach. 2012. (Cité en page 10.)

[Lazaar *et al.* 2016] Nadjib Lazaar, Yahia Lebbah, Samir Loudni, Mehdi Maamar, Valentin Lemière, Christian Bessiere et Patrice Boizumault. A Global Constraint for Closed Frequent Pattern Mining. Dans Principles and Practice of Constraint Programming - 22nd International Conference, CP 2016, Toulouse, France, September 5-9, 2016, Proceedings, pages 333–349, 2016. (Cité en page 27.)

[Leskovec *et al.* 2010] Jure Leskovec, Kevin J. Lang et Michael W. Mahoney. Empirical comparison of algorithms for network community detection. Dans WWW, pages 631–640, 2010. (Cité en page 42.)

[Leung *et al.* 2008] Carson Kai-Sang Leung, Mark Anthony F Mateo et Dale A Brajczuk. A tree-based approach for frequent pattern mining from uncertain data. Dans Advances in Knowledge Discovery and Data Mining, pages 653–661. Springer, 2008. (Cité en page 29.)

[Liberti 2012] Leo Liberti. Reformulations in mathematical programming : automatic symmetry detection and exploitation. Math. Program., vol. 131, no. 1-2, pages 273–304, 2012. (Cité en page 7.)

[Lin & Kedem 1998] Dao-I Lin et Zvi M. Kedem. Pincer-search : A new algorithm for discovering the maximum frequent set, pages 103–119. Springer Berlin Heidelberg, 1998. (Cité en page 25.)

[Luce 1950] R. Duncan Luce. Connectivity and generalized cliques in sociometric group structure. Psychometrika, vol. 15, no. 2, pages 169–190, Jun 1950. (Cité en page 42.)

[Ma *et al.* 2011] Yue Ma, Guilin Qi et Pascal Hitzler. Computing inconsistency measure based on paraconsistent semantics. J. Log. Comput., vol. 21, no. 6, pages 1257–1281, 2011. (Cité en page 46.)

[Macqueen 1967] J. Macqueen. Some methods for classification and analysis of multivariate observations. Dans In 5-th Berkeley Symposium on Mathematical Statistics and Probability, pages 281–297, 1967. (Cité en page 36.)

[Manquinho *et al.* 1997] Vasco M. Manquinho, Paulo E Flores, Jo P. Marques Silva et Arlindo L. Oliveira. Prime implicant computation using satisfiability algorithms. Dans in Proc. of the IEEE International Conference on Tools with Artificial Intelligence, pages 232–239, 1997. (Cité en page 13.)

[Manquinho *et al.* 1998] Vasco M. Manquinho, Arlindo L. Oliveira et João P. Marques Silva. Models and Algorithms for Computing Minimum-Size Prime Implicants. Dans In Proc. International Workshop on Boolean Problems (IWBP'98, 1998. (Cité en page 13.)

[Margot 2003] François Margot. Exploiting orbits in symmetric ILP. Math. Program., vol. 98, no. 1-3, pages 3–21, 2003. (Cité en page 7.)

[Marques-Silva & Sakallah 1996] Joao P. Marques-Silva et Karem A. Sakallah. GRASP - A New Search Algorithm for Satisfiability. Dans Proceedings of IEEE/ACM CAD, pages 220–227, 1996. (Cité en page 6.)

[Massacci & Marraro 2000] Fabio Massacci et Laura Marraro. Logical Cryptanalysis as a SAT Problem. J. Autom. Reasoning, vol. 24, no. 1/2, pages 165–203, 2000. (Cité en page 9.)

[Matas & Kittler 1995] Jiri Matas et Josef Kittler. Spatial and Feature Space Clustering : Applications in Image Analysis. Dans Computer Analysis of Images and Patterns, 6th International Conference, CAIP'95, Prague, Czech Republic, September 6-8, 1995, Proceedings, pages 162–173, 1995. (Cité en page 35.)

[McCluskey 1956] Jr. McCluskey E.J. Minimization of Boolean Functions. Bell System Technical Journal, vol. 35, no. 6, pages 1417–1444, 1956. (Cité en page 13.)

[McKay 1990] B. McKay. nauty user's guide (version 1.5). Rapport technique, 1990. (Cité en page 27.)

[McMillan 2002] Kenneth L. McMillan. Applying SAT Methods in Unbounded Symbolic Model Checking. Dans Proceedings of the 14th International Conference on Computer Aided Verification (CAV'02), pages 250–264, 2002. (Cité en page 12.)

[Meseguer *et al.* 2006] Pedro Meseguer, Francesca Rossi et Thomas Schiex. Soft Constraints. Dans Handbook of Constraint Programming, volume 2 de Foundations of Artificial Intelligence, pages 281–328. Elsevier, 2006. (Cité en page 14.)

[Michalski 1980] R. S. Michalski. Knowledge Acquisition Through Conceptual Clustering : A Theoretical Framework and an Algorithm for Partitioning Data into Conjunctive Concepts. Journal of Policy Analysis and Information Systems, vol. 4, no. 3, pages 219–244, 1980. (Cité en page 35.)

[Morgado & Marques-Silva 2005] António R. Morgado et João P. Marques-Silva. Good Learning and Implicit Model Enumeration. Dans International Conference on Tools with Artificial Intelligence (ICTAI'2005), pages 131–136. IEEE, 2005. (Cité en page 12.)

[Moskewicz *et al.* 2001] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang et S. Malik. Chaff : Engineering an Efficient SAT Solver. Dans Proceedings of the 38th Design Automation Conference, pages 530–535, 2001. (Cité en page 6.)

[Mu *et al.* 2011] Kedian Mu, Weiru Liu, Zhi Jin et David A. Bell. A Syntax-based approach to measuring the degree of inconsistency for belief bases. Int. J. Approx. Reasoning, vol. 52, no. 7, pages 978–999, 2011. (Cité en page 45.)

[Nacer *et al.* 2015] Amina Ahmed Nacer, Kahina Bessai, Samir Youcef et Claude Godart. A Multi-criteria Based Approach for Web Service Selection Using Quality of Service (QoS). Dans Services Computing (SCC), 2015 IEEE International Conference on, pages 570–577. IEEE, 2015. (Cité en page 53.)

[Négrevergne *et al.* 2013] Benjamin Négrevergne, Anton Dries, Tias Guns et Siegfried Nijssen. Dominance Programming for Itemset Mining. Dans 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7-10, 2013, pages 557–566, 2013. (Cité en page 14.)

[Newman & Girvan 2004] M. E. J. Newman et M. Girvan. Finding and evaluating community structure in networks. Phys. Rev. E, vol. 69, no. 2, page 026113, Février 2004. (Cité en pages 39 et 40.)

[Newman 2010] M. E. J. Newman. Networks : An introduction. Oxford Univ. Press, Oxford, U.K., 2010. (Cité en page 41.)

[Okutan & Cicekli 2010] Cagla Okutan et Nihan Kesim Cicekli. A Monolithic Approach to Automated Composition of Semantic Web Services with the Event Calculus. Know.-Based Syst., vol. 23, no. 5, pages 440–454, Juillet 2010. (Cité en page 53.)

[Oller 2004] Carlos A. Oller. Measuring coherence using LP-models. J. Applied Logic, vol. 2, no. 4, pages 451–455, 2004. (Cité en page 46.)

[Parida *et al.* 2000] Laxmi Parida, Isidore Rigoutsos, Aris Floratos, Dan Platt et Yuan Gao. Pattern Discovery on Character Sets and Real-valued Data : Linear Bound on Irredundant Motifs and an Efficient Polynomial Time Algorithm. Dans ACM-SIAM Symposium on Discrete Algorithms, pages 297–308, 2000. (Cité en page 32.)

[Pipatsrisawat & Darwiche 2008] Knot Pipatsrisawat et Adnan Darwiche. A New Clause Learning Scheme for Efficient Unsatisfiability Proofs. Dans Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008, pages 1481–1484, 2008. (Cité en page 8.)

[Pisanti *et al.* 2003] Nadia Pisanti, Maxime Crochemore, Roberto Grossi et Marie france Sagot. Bases of motifs for generating repeated patterns with wild cards. IEEE/ACM TCBB'2003, vol. 2, page 2005, 2003. (Cité en page 32.)

[Pizzuti 1996] Clara Pizzuti. Computing Prime Implicants by Integer Programming. Dans Proceedings of the 8th International Conference on Tools with Artificial Intelligence, ICTAI '96, pages 332–336, Washington, DC, USA, 1996. IEEE Computer Society. (Cité en page 13.)

[Quine 1952] W. V. Quine. The Problem of Simplifying Truth Functions. The American Mathematical Monthly, vol. 59, no. 8, pages 521–531, 1952. (Cité en page 13.)

[Quine 1959] W. Quine. On cores and prime implicants of truth functions. American Mathematical Monthly, pages 755–760, 1959. (Cité en page 13.)

[Raedt *et al.* 2008] Luc De Raedt, Tias Guns et Siegfried Nijssen. Constraint Programming for Itemset Mining. Dans Proceedings of SIGKDD'08, pages 204–212, 2008. (Cité en page 21.)

[Raedt *et al.* 2011] Luc De Raedt, Siegfried Nijssen, Barry O'Sullivan et Pascal Van Hentenryck. Constraint Programming meets Machine Learning and Data Mining. Dagstuhl Reports, vol. 1, no. 5, pages 61–83, 2011. (Cité en page 21.)

[Ramani *et al.* 2006] A. Ramani, I. L. Markov, K. A. Sakallah et F. A. Aloul. Breaking Instance-Independent Symmetries In Exact Graph Coloring. Journal of Artificial Intelligence Research (JAIR), vol. 26, pages 289–322, 2006. (Cité en page 27.)

[Rintanen 2006] J. Rintanen. Compact representation of sets of binary constraints. Dans Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006), pages 143–147. IOS Press, 2006. (Cité en page 18.)

[Rodriguez-Mier *et al.* 2011] Pablo Rodriguez-Mier, Manuel Mucientes et Manuel Lama. Automatic Web Service Composition with a Heuristic-Based Search Algorithm. Dans Proceedings of the 2011 IEEE International Conference on Web Services, ICWS '11, pages 81–88, Washington, DC, USA, 2011. IEEE Computer Society. (Cité en page 53.)

[Rosa *et al.* 2010] Emanuele Di Rosa, Enrico Giunchiglia et Marco Maratea. Solving satisfiability problems with preferences. Constraints, vol. 15, no. 4, pages 485–515, 2010. (Cité en page 14.)

[Saito *et al.* 2008] Kazumi Saito, Takeshi Yamada et Kazuhiro Kazama. Extracting Communities from Complex Networks by the k-Dense Method. IEICE Transactions, vol. 91-A, no. 11, pages 3304–3311, 2008. (Cité en page 42.)

[Schaus *et al.* 2017] Pierre Schaus, John O. R. Aoga et Tias Guns. CoverSize : A Global Constraint for Frequency-Based Itemset Mining. Dans Principles and Practice of Constraint Programming - 23rd International Conference, CP 2017, Melbourne, VIC, Australia, August 28 - September 1, 2017, Proceedings, pages 529–546, 2017. (Cité en page 27.)

[Seidman 1978] Stephen B. Seidman. A graph-theoretic generalization of the clique concep. Journal of Mathematical sociology, vol. 5, no. 3, page 139–154, 1978. (Cité en page 42.)

[Shao *et al.* 2014] Yingxia Shao, Lei Chen et Bin Cui. Efficient cohesive subgraphs detection in parallel. Dans SIGMOD, pages 613–624, 2014. (Cité en page 42.)

[Shen *et al.* 2009] Huawei Shen, Xueqi Cheng, Kai Cai et Mao-Bin Hu. Detect overlapping and hierarchical community structure in networks. Physica A, vol. 388, no. 8, pages 1706–1712, 2009. (Cité en page 39.)

[Shiaa *et al.* 2008] M.M. Shiaa, J.O. Fladmark et B. Thiell. An Incremental Graph-based Approach to Automatic Service Composition. Dans Services Computing, 2008. SCC '08. IEEE International Conference on, volume 1, pages 397–404, July 2008. (Cité en page 53.)

[Shoham 1988] Yoav Shoham. Reasoning about change : time and causation from the standpoint of artificial intelligence. MIT Press, Cambridge, MA, USA, 1988. (Cité en page 14.)

[Silva & Lynce 2007] J. P. Marques Silva et I. Lynce. Towards Robust CNF Encodings of Cardinality Constraints. Dans 13th International Conference on Principles and Practice of Constraint Programming (CP 2007), pages 483–497, 2007. (Cité en page 15.)

[Sinz 2005] C. Sinz. Towards an Optimal CNF Encoding of Boolean Cardinality Constraints. Dans 11th International Conference on Principles and Practice of Constraint Programming - CP 2005, pages 827–831, 2005. (Cité en pages 15 et 17.)

[Slavkovik & Agotnes 2014] M. Slavkovik et T. Agotnes. A Judgment Set Similarity Measure Based on Prime Implicants. Dans Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '14, page to appear, 2014. (Cité en page 13.)

[Sneath & Sokal 1973] P.H.A. Sneath et R.R. Sokal. Numerical taxonomy. the principles and practice of numerical classification. Freeman, 1973. (Cité en page 36.)

[Soos *et al.* 2009] Mate Soos, Karsten Nohl et Claude Castelluccia. Extending SAT Solvers to Cryptographic Problems. Dans Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings, pages 244–257, 2009. (Cité en page 9.)

[Soulet *et al.* 2011] Arnaud Soulet, Chedy Raïssi, Marc Plantevit et Bruno Crémilleux. Mining Dominant Patterns in the Sky. Dans ICDM'11, pages 655–664, 2011. (Cité en page 29.)

[Tan *et al.* 2000] Pang-Ning Tan, Vipin Kumar et Jaideep Srivastava. Indirect Association : Mining Higher Order Dependencies in Data. Dans Proceedings of PKDD'00, pages 632–637, 2000. (Cité en page 31.)

[Tao *et al.* 2003] Feng Tao, Fionn Murtagh et Mohsen Farid. Weighted Association Rule Mining using Weighted Support and Significance Framework. Dans Proceedings of SIGKDD'03, pages 661–666, 2003. (Cité en page 29.)

[Thimm 2016] Matthias Thimm. On the expressivity of inconsistency measures. Artif. Intell., vol. 234, pages 120–151, 2016. (Cité en page 47.)

[Tison 1967] P. Tison. Generalized consensus theory and applications to the minimization of boolean circuits. IEEE Transactions on Computers, pages 446–456, 1967. (Cité en page 13.)

[Tseitin 1968] G.S. Tseitin. On the complexity of derivations in the propositional calculus. Dans H.A.O. Slesenko, editeur, Structures in Constructives Mathematics and Mathematical Logic, Part II, pages 115–125, 1968. (Cité en pages 6 et 18.)

[Tversky 1977] A. Tversky. Features of similarity. Psychological Review, vol. 84, no. 4, page 327–352, 1977. (Cité en page 36.)

[Ugarte Rojas *et al.* 2014] Willy Ugarte Rojas, Patrice Boizumault, Samir Loudni, Bruno Crémilleux et Alban Lepailleur. Mining (Soft-) Skypatterns Using Dynamic CSP. Dans Integration of AI and OR Techniques in Constraint Programming (CPAIOR'14), pages 71–87, 2014. (Cité en page 14.)

[Vanetik 2010] Natalia Vanetik. Mining Graphs with Constraints on Symmetry and Diameter. Dans International Workshops on Web-Age Information Management - WAIM 2010, pages 1–12, 2010. (Cité en page 27.)

[Wakrime & Jabbour 2015] Abderrahim Ait Wakrime et Saïd Jabbour. Minimum Unsatisfiability based QoS Web Service Composition over the Cloud Computing. Dans 15th International Conference on Intelligent Systems Design and Applications, ISDA 2015, Marrakech, Morocco, December 14-16, 2015, pages 540–545, 2015. (Cité en pages 55 et 56.)

[Wakrime & Jabbour 2017] Abderrahim Ait Wakrime et Saïd Jabbour. Formal Approach for QoS-Aware Cloud Service Composition. Dans 26th IEEE International Conference on Enabling Technologies : Infrastructure for Collaborative Enterprises, WETICE 2017, Poznan, Poland, June 21-23, 2017, pages 30–35, 2017. (Cité en page 55.)

[Wakrime *et al.* 2015] Abderrahim Ait Wakrime, Salima Benbernou et Saïd Jabbour. Relaxation Based SaaS for Repairing Failed Queries over the Cloud Computing. Dans 12th IEEE International Conference on e-Business Engineering, ICEBE 2015, Beijing, China, October 23-25, 2015, pages 245–250, 2015. (Cité en page 56.)

[Walsh 2006] T. Walsh. General Symmetry Breaking Constraints. Dans 12th International Conference on Principles and Practice of Constraint Programming (CP 2006), pages 650–664, 2006. (Cité en page 7.)

[Walsh 2007] Toby Walsh. Representing and Reasoning with Preferences. AI Magazine, vol. 28, no. 4, pages 59–70, 2007. (Cité en pages 14 et 29.)

[Wang & Cheng 2012] Jia Wang et James Cheng. Truss Decomposition in Massive Networks. PVLDB, vol. 5, no. 9, pages 812–823, 2012. (Cité en page 42.)

[Wang *et al.* 1999] Ke Wang, Chu Xu et Bing Liu. Clustering Transactions Using Large Items. Dans Proceedings of the 1999 ACM CIKM International Conference on Information and Knowledge Management, Kansas City, Missouri, USA, November 2-6, 1999, pages 483–490, 1999. (Cité en page 35.)

[Warners 1996] J. P. Warners. A Linear-Time Transformation of Linear Inequalities into Conjunctive Normal Form. Information Processing Letters, 1996. (Cité en page 15.)

[Wasserman & Faust 1994] S. Wasserman et K. Faust. Social network analysis : Methods and applications. Cambridge Univ Press, 1994. (Cité en page 42.)

[Wittkop *et al.* 2010] Tobias Wittkop, Dorothea Emig, Sita Lange, Sven Rahmann, Mario Albrech et John H. Morris. On the maximum quasi-clique problem. Nature Methods, vol. 7, no. 6, 2010. (Cité en page 35.)

[Wu & Khoury 2012] Ching-Seh Wu et Ibrahim Khoury. Tree-based search algorithm for web service composition in SaaS. Dans Information Technology : New Generations (ITNG), 2012 Ninth International Conference on, pages 132–138. IEEE, 2012. (Cité en page 53.)

[Xiao & Ma 2012] Guohui Xiao et Yue Ma. Inconsistency Measurement based on Variables in Minimal Unsatisfiable Subsets. Dans ECAI, pages 864–869, 2012. (Cité en page 45.)

[Xie *et al.* 2011] Jierui Xie, Boleslaw K. Szymanski et Xiaoming Liu. SLPA : Uncovering Overlapping Communities in Social Networks via a Speaker-Listener Interaction Dynamic Process. Dans IEEE International Conference on Data Mining, pages 344–349, 2011. (Cité en page 39.)

[Yang & Leskovec 2013] Jaewon Yang et Jure Leskovec. Overlapping community detection at scale : a nonnegative matrix factorization approach. Dans WSDM, pages 587–596, 2013. (Cité en pages 39 et 40.)

[Yun & Leggett 2005] Unil Yun et John J. Leggett. WFIM : Weighted Frequent Itemset Mining with a Weight Range and a Minimum Weight. Dans InProceedings of the Fifth SIAM International Conference on Data Mining, Pages, pages 636–640, 2005. (Cité en page 29.)

[Zaki & Hsiao 2002] Mohammed Javeed Zaki et Ching-Jiu Hsiao. CHARM : An Efficient Algorithm for Closed Itemset Mining. Dans Proceedings of SDM'02, pages 457–473, 2002. (Cité en page 25.)

[Zaki 2004] Mohammed Javeed Zaki. Mining Non-Redundant Association Rules. Data Mining Knowledge Discovery, vol. 9, pages 223–248, 2004. (Cité en pages 31 et 32.)

[Zhang *et al.* 2001] Lintao Zhang, Conor F. Madigan, Matthew W. Moskewicz et Sharad Malik. Efficient Conflict Driven Learning in Boolean Satisfiability Solver. Dans Proceedings of the 2001 IEEE/ACM International Conference on Computer-Aided Design, ICCAD 2001, San Jose, CA, USA, November 4-8, 2001, pages 279–285, 2001. (Cité en pages 6 et 25.)

[Zou *et al.* 2002] Qinghua Zou, Wesley W. Chu et Baojing Lu. SmartMiner : A Depth First Algorithm Guided by Tail Information for Mining Maximal Frequent Itemsets. Dans Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan, pages 570–577, 2002. (Cité en page 25.)

Chapitre 9

Publications jointes

1. A Parallel SAT-based Framework for Closed Frequent Itemsets Mining (CP'2018)
2. A SAT-based Framework for Overlapping Community Detection in Network (PAKDD'2017)
3. Clustering Complex Data Represented as propositional formulas (PAKDD'2017)
4. Summarizing Large Graphs by Means of Pseudo-Boolean Constraints (BigData'2016)
5. Inconsistency Measurement Thanks to MUS Decomposition (AAMAS'2014)
6. Mining-Based Compression Approach of Propositional Formulae (CIKM'2013)
7. Boolean Satisfiability for Sequence Mining (CIKM'2013)
8. The Top-k Frequent Closed Itemset Mining Using Top-k SAT Problem (ECML/PKDD'2013)
9. Symmetries in Itemset Mining (ECAI'2012)
10. Control-based Clause Sharing in Parallel SAT Solving (IJCAI'2009)

Control-based Clause Sharing in Parallel SAT Solving

Youssef Hamadi
Microsoft Research
7 J J Thomson Avenue, CB3 0FB Cambridge,
United Kingdom, youssefh@microsoft.com

Said Jabbour and **Lakhdar Sais**
CRIL-CNRS, Université d'Artois
Rue Jean Souvraz SP18, F-62307 Lens,
France, {jabbour,sais}@cril.fr

Abstract

Conflict driven clause learning, one of the most important component of modern SAT solvers, is also recognized as very important in parallel SAT solving. Indeed, it allows clause sharing between multiple processing units working on related (sub-)problems. However, without limitation, sharing clauses might lead to an exponential blow up in communication or to the sharing of irrelevant clauses. This paper, proposes two innovative policies to dynamically adjust the size of shared clauses between any pair of processing units. The first approach controls the overall number of exchanged clauses whereas the second additionally exploits the relevance quality of shared clauses. Experimental results show important improvements of the state-of the-art parallel SAT solver.

1 Introduction

The recent successes of SAT solvers in traditional hardware and software applications have extended their applicability to important new domains. Today, they represent essential low level reasoning components used in general theorem proving, computational biology, AI, etc. This popularity gain is related to their breakthrough on real world instances involving million of clauses and hundred of thousands of variables. These solvers, called modern SAT solvers [Moskewicz *et al.*, 2001; Eén and Sörensson, 2003], are based on a nice combination of (i) clause learning [Marques-Silva and Sakallah, 1996; Moskewicz *et al.*, 2001], (ii) activity-based heuristics [Moskewicz *et al.*, 2001], and (iii) restart policies [Gomes *et al.*, 1998] enhanced with efficient data structures (e.g. watched-literals [Moskewicz *et al.*, 2001]). This architecture is now standard, and today only minor improvements have been observed (cf. SAT competitions). Therefore, it seems difficult to bet on others orders of magnitude gains without a radical algorithmic breakthrough.

Fortunately, the recent generalization of multicore hardware gives parallel processing capabilities to standard PCs. This represents a real opportunity for SAT researchers which can now consider parallel SAT solving as an obvious way toward substantial efficiency gains.

Recent works on parallel SAT are all based on the modern SAT architecture [Moskewicz *et al.*, 2001], and therefore systematically exploit clause learning as an easy way to extend the cooperation between processing units. When a unit learns a new clause, it can share it with all other units in order to prune their search spaces. Unfortunately, since the number of potential conflicts is exponential, the systematic sharing of learnt clauses is not practically feasible. The solution is to exchange up to some predefined size limit. This has the advantage of reducing the overhead of the cooperation while focusing the exchange on short clauses, recognized as more powerful in term of search pruning.

In this work, our goal is to improve the clause sharing scheme of modern parallel SAT solvers. Indeed, the approach based on some predefined size limit has several flaws. The first and most apparent being that an overestimated value might induce a very large cooperation overhead, while an underestimated one might completely inhibit the cooperation. The second flaw comes from the observation that the size of learnt clauses tends to increase over time (see section 3.1), leading to an eventual halt of the cooperation. The third flaw is related to the internal dynamic of modern solvers which tend to focus on particular subproblems thanks to the activity/restart mechanisms. In parallel SAT, this can lead two search processes toward completely different subproblems where clause sharing becomes pointless.

We propose a dynamic clause sharing policy which uses pairwise size limits to control the exchange between any pair of processing units. Initially, high limits are used to enforce the cooperation, and allow pairwise exchanges. On a regular basis, each unit considers the number of foreign clauses received from other units. If this number is below/above a predefined threshold, the pairwise limits are increased/decreased. This mechanism allows the system to maintain a throughput. It addresses the flaws one and two. To address the last flaw related to the poor relevance of the shared clauses, we extend our policy to integrate the quality of the exchanges. Each unit evaluates the quality of the received clauses, and the control is able to selectively increase/decrease the pairwise limits based on the underlying quality of the recently communicated clauses. The rationale being that the information recently received from a particular source is qualitatively linked to the information which could be received from it in the very near future. The evolution

of the pairwise limits w.r.t., the throughput or quality criterion follows an AIMD (Additive-Increase-Multiplicative-Decrease) feedback control-based algorithm [Chiu and Jain, 1989].

The paper is organized as follows. After some preliminaries (section 2), our dynamic control-based clause sharing policies are motivated and presented in section 3. The section 4 presents extensive experimental evaluation of our policies as opposed to a standard static one. The section 5 details previous parallel SAT works. Finally, we conclude by providing some interesting future paths of research.

2 Technical background

In this section, we introduce the computational features of modern SAT solvers. Then, we briefly describe the principle of the AIMD feedback control-based algorithm usually applied to solve TCP congestion control problems.

2.1 Computational features of modern SAT solvers

Most of the state of the art SAT solvers are based on the Davis, Putnam, Logemann and Loveland procedure, commonly called DPLL [Davis *et al.*, 1962]. DPLL is a backtrack search procedure; at each node of the search tree, a decision literal is chosen according to some branching heuristic. Its assignment to one of the two possible values (true or false) is followed by an inference step that deduces and propagates some forced unit literal assignments. The assigned literals (decision literal and the propagated ones) are labeled with the same decision level starting from 1 and increased at each decision (or branching) until finding a model or reaching a conflict (or a dead end). In the first case, the formula is answered to be satisfiable, whereas in the second case, we backtrack to the last decision level and assign the remaining value to the last decision literal. After backtracking, some variables are unassigned, and the current decision level is decreased accordingly. The formula is answered to be unsatisfiable when backtracking to level 0 occurs. In addition to this basic scheme, modern SAT solvers use additional important component such as restart policy, conflict driven clause learning and activity based heuristics. Let us give some details on these last two important features. First, to learn from conflict, they maintain a central data-structure, the implication graph, which records the partial assignment that is under construction together with its implications. When a dead end occurs, a conflict clause (called asserting clause) is generated by resolution following a bottom-up traversal of the implication graph. The learning process stops, when a conflict clause containing only one literal from the current decision level is generated. Such a conflict clause (or learnt clause) expresses that such a literal is implied at a previous level. Modern SAT solvers backtrack to the implication level and assign that literal to true. Let us mention, that the activity of each variable encountered during such resolution process is increased. The variable with greatest activity is selected to be assigned next.

As the number of learnt clauses can grow exponentially, even in the sequential case, modern SAT solvers regularly reduce the data-base of learnt clauses. In the SAT solver Minisat [Eén and Sörensson, 2003], such a reduction (called reduceDB) is achieved as follows. When the size of the learnt data base exceeds a given upper bound (B), it is reduced by half. The set of deleted clauses corresponds to the less active ones. Initially, B is set to $\frac{1}{3} \times |\mathcal{F}|$) where $|\mathcal{F}|$ is the number of clauses in the original formula $\mathcal{F}$. At each restart, B is increased by 10%.

2.2 AIMD feedback control based algorithm

The Additive Increase/Multiplicative Decrease (AIMD) algorithm is a feedback control algorithm used in TCP congestion avoidance. The problem solved by AIMD is to guess the communication bandwidth available between two communicating nodes. The algorithm performs successive probes, increasing the communication rate w linearly as long as no packet loss is observed, and decreasing it exponentially when a loss is encountered. More precisely, the evolution of w is defined by the following $AIMD(a, b)$ formula:

- $w = w - a \times w$, if loss is detected
- $w = w + \frac{b}{w}$, otherwise

Different proposals have been made in order to prevent congestion in communication networks based on different numbers for a and b. Today, AIMD is the major component of TCP's congestion avoidance and control [Jacobson, 1988]. On probe of network bandwidth increasing too quickly will overshoot limits (underlying capacities). On notice of congestion, decreasing too slowly will not be reactive enough.

In the context of clause sharing, our control policies want to achieve a particular throughput or a particular throughput of maximum quality. Since any increase in the size limit can potentially generate a very large number of new clauses, AIMD's slow increase can help us to avoid a quick overshoot of the throughput. Similarly, in case of overshooting, aggressive decrease can help us to quickly reduce clause sharing by a very large amount.

3 Control-based clause sharing in parallel SAT solving

3.1 Motivation

To motivate further our proposed framework, we conducted a simple experiment using the standard Minisat algorithm [Eén and Sörensson, 2003]. In figure 1 we show the evolution of the percentage of learnt clauses of size less than or equal to 8 on a particular family of 16 industrial instances *AProVE07_**. This limit represents the default static clause sharing limit the ManySAT parallel solver [Hamadi *et al.*, 2009].

This percentage is computed every 10000 conflicts, and as it can be observed, it decreases over time[1]. Initially, nearly 17% of the learnt clauses could be exchanged but as search goes on, this percentage falls below 4%. Our observation is very general and can be performed on different instances with similar or different clause size limits. It illustrates the second flaw reported above: the size of learnt clauses tends to increase over time. Consequently, in a parallel SAT setting, any static limit might lead to an halt of the clause sharing process. Therefore, if one wants to maintain a quantity of exchange

[1]The regular small raises are the result of the cyclic reduction of the learnt base through reduceDB.

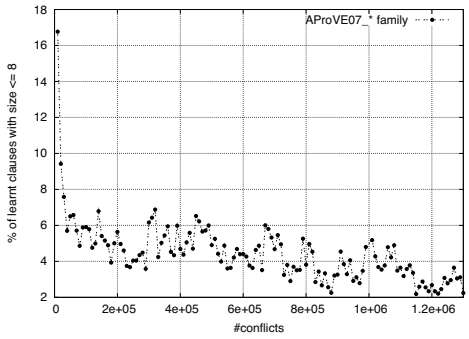


Figure 1: Evolution of the percentage of learnt-clauses with size ≤ 8

over time, there does not exist an optimal *static* policy for that. This clearly shows the importance of a dynamic control clause sharing policy.

3.2 Throughput and quality based control policies

In this section, we describe our dynamic control-based clause sharing policies which control the exchange between any pair of processing units through dynamic pairwise size limits.

The first policy controls the throughput of clause sharing. Each unit considers the number of foreign clauses received from other units. If this number is below/above a predefined throughput-threshold, the pairwise limits are all increased/decreased using an AIMD feedback algorithm. The second policy is an extension of the previous one. It introduces a measure of the quality of foreign clauses. With this information, the increase/decrease of the pairwise limits become proportional to the underlying quality of the clauses shared by each unit. The first (respectively second) policy allows the system to maintain a throughput (respectively throughput of better quality).

We consider a parallel SAT solver with n different processing units. Each unit u_i corresponds to a SAT solver with clause learning capabilities. Each solver can either work on a subspace of the original instance as in divide-and-conquer techniques, or on the full problem, as in ManySAT (see details in sections 5 and 4.1). We assume that these different units communicate through a shared memory (as in multicore architectures).

In our control strategy, we consider a control-time sequence as a set of steps t_k with $t_0 = 0$ and $t_k = t_{k-1} + \alpha$ where α is a constant representing the time window defined in term of number of conflicts. The step t_k of a given unit u_i corresponds to the conflict number $k \times \alpha$ encountered by the solver associated to u_i. In the sequel, when there is no ambiguity, we sometimes note t_k simply k. Then, each unit u_i can be defined as a sequence of states $S_i^k = (\mathcal{F}, \Delta_i^k, R_i^k)$, where $\mathcal{F}$ is a CNF formula, Δ_i^k the set of its proper learnt clauses and R_i^k the set of foreign clauses received from the other units between two consecutive steps $k-1$ and k. The different units achieve pairwise exchange using pairwise limits. Between two consecutive steps $k-1$ and k, a given unit u_i receives from all the other remaining units u_j where $0 \leq j < n$ and $j \neq i$ a set of learnt clauses $\Delta_{j\to i}^k$ of length less or equal to a size limit $e_{j\to i}^k$ i.e., $\Delta_{j\to i}^k = \{c \in \Delta_j^k / \, |c| \leq e_{j\to i}^k\}$. Then, the set R_i^k can be formally defined as $\cup_{0\leq j<n, j\neq i} \Delta_{j\to i}^k$.

Using a fixed throughput threshold T of shared clauses, we describe our control-based policies which allow each unit u_i to guide the evolution of the size limit $e_{j\to i}$ using an AIMD feedback mechanism.

Throughput based control

As illustrated in figure 2, at step k a given unit u_i checks whether the throughput is exceeded or not. if $|R_i^k| < T$ (respectively $|R_i^k| > T$) the size limit $e_{j\to i}^{k+1}$ is additively increased (respectively multiplicatively decreased). More formally, the upper bound $e_{j\to i}^{k+1}$ on the size of clauses that a solver j shares with the solver i between k and $k+1$ are changed using the following AIMD function:

$aimdT(R_i^k)\{$
$\forall j | 0 \leq j < n, j \neq i$
$e_{j\to i}^{k+1} = \begin{cases} e_{j\to i}^k + \frac{b}{e_{j\to i}^k}, if(|R_i^k| < T) \\ e_{j\to i}^k - a \times e_{j\to i}^k, if(|R_i^k| > T) \end{cases}$ } where a and b are positive constants.

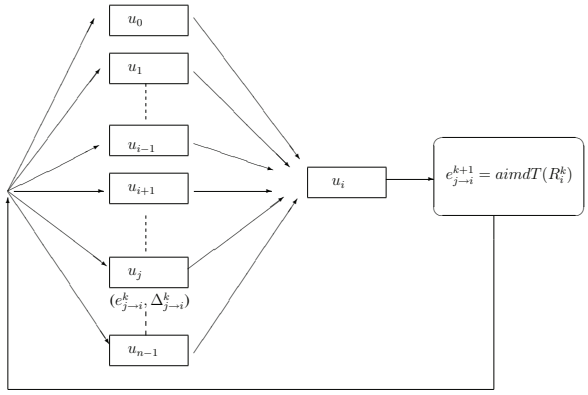


Figure 2: Throughput based control policy

Throughput and quality based control

In this policy, to control the throughput of a given unit u_i, we introduce a quality measure $Q_{j\to i}^k$ (see definition 1) to estimate the relative quality of the clauses received by u_i from u_j. In the throughput and quality based control policy, the evolution of the size limit $e_{j\to i}^k$ is related to the estimated quality.

Our quality measure is defined using the activity of the variables at the basis of VSIDS heuristic [Moskewicz *et al.*, 2001] another important component of modern SAT solvers. The variables with greatest activity represent those involved in most of the (recent)-conflicts. Indeed, when a conflict occurs, the activity of the variables whose literals appear in the clauses encountered during the generation of a learnt clause are updated. The most active variables are those related to the current part of the search space. Consequently, our quality measure exploits these activities to quantify the relevance of a clause learnt by unit u_j to the current state of a given unit u_i. To define our quality measure, suppose that, at any

time of the search process, we have $\mathcal{A}_i^{max}$ the current maximal activity of u_i's variables, and $\mathcal{A}_i(x)$ the current activity of a given variable x.

Definition 1 (Quality) *Let c be a clause and $\mathcal{L}_{\mathcal{A}_i}(c) = \{x/x \in c \text{ s.t. } \mathcal{A}_i(x) \geq \frac{\mathcal{A}_i^{max}}{2}\}$ the set of active literals of c with respect to unit u_i. We define $\mathcal{P}_{j\to i}^k = \{c/c \in \Delta_{j\to i}^k$ s.t. $|\mathcal{L}_{\mathcal{A}_i}(c)| \geq Q\}$ be the set of clauses received by i from j between steps $k-1$ and k with at least Q active literals. We define the quality of clauses sent by u_j to u_i at a given step k as $Q_{j\to i}^k = \frac{|\mathcal{P}_{j\to i}^k|+1}{|\Delta_{j\to i}^k|+1}$*

Our throughput and quality based control policy change the upper bound $e_{j\to i}^{k+1}$ on the size of clauses that a solver j shares with the solver i between k and $k+1$ using the following AIMD function:

$aimdTQ(R_i^k)\{$
$\forall j | 0 \leq j < n, j \neq i$

$$e_{j\to i}^{k+1} = \begin{cases} e_{j\to i}^k + (\frac{Q_{j\to i}^k}{100}) \times \frac{b}{e_{j\to i}^k}, if(|R_i^k| < T) \\ e_{j\to i}^k - (1 - \frac{Q_{j\to i}^k}{100}) \times a \times e_{j\to i}^k, if(|R_i^k| > T) \end{cases}$$

$\}$ where a and b are positive constants.

As shown by the AIMD function of the throughput and quality based control policy, the adjustment of the size limit depends on the quality of shared clauses. Indeed, as it can be seen from the above formula, when the exchange quality between u_j and u_i ($Q_{j\to i}^k$) tends to 100% (respectively 0%), then the increase in the limit size tends to be maximal (respectively minimal) while the decrease tends to be minimal (respectively maximal). Our aim in this second policy is to maintain a throughput of good quality. The rationale being that the information recently received from a particular source is qualitatively linked to the information which could be received from it in the very near future.

4 Evaluation

4.1 The parallel SAT solver

Our policies were implemented and tested on top of the ManySAT parallel SAT solver [Hamadi *et al.*, 2009] which won the parallel track of the 2008 SAT-Race[2]. ManySAT includes all the classical features like two-watched-literal, unit propagation, activity-based decision heuristics, lemma deletion strategies, and clause learning. In addition to the classical first-UIP scheme, it incorporates a new technique which extends the classical implication graph used during conflict-analysis to exploit the satisfied clauses of a formula [Audemard *et al.*, 2008].

Unlike others parallel SAT solvers, ManySAT does not implement a divide-and-conquer strategy based on some dynamic partitioning of the search space. At contrary, it uses a portfolio philosophy which lets several sequential DPLLs compete and cooperate to be the first to solve the common instance. These DPLLs are differentiated in many ways. They use different and complementary restart strategies, VSIDS and polarity heuristics, and learning schemes. Additionally, all the DPLLs are exchanging learnt clauses up to some static size limit.

[2]http://baldur.iti.uka.de/sat-race-2008/

4.2 Experiments

Our tests were done on Intel Xeon Quadcore machines with 16GB of RAM running at 2.3Ghz. We used a timeout of 1500 seconds for each problem. ManySAT was used with 4 DPLLs strategies each one running on a particular core (unit). To alleviate the effects of unpredictable threads scheduling, each problem was solved three times and the average was taken.

Our dynamic clause sharing policies were added to ManySAT and compared against ManySAT with its default static policy *ManySAT e=8* which exchanges clauses up to size 8. Remark that since each pairwise limit is read by a unit, and updated by another one, our proposal can be integrated without any lock.

We have selected $a = 0.125, b = 8$ for aimdT and aimdTQ, associated to a time window of $\alpha = 10000$ conflicts. The throughput T is set to $\frac{\alpha}{2}$ and the upper bound Q on the number of active literals per clause c is set to $\frac{|c|}{3}$ (see definition 1). Each pairwise limit $e_{j\to i}$ was initialized to 8.

4.3 Industrial problems

The Table 1 presents the results on the 100 industrial problems of the 2008 SAT-Race. The problem set contains families with several instances or individual instances.

From left to right we present, the family/instance name, the number of instances per family. Results associated to the standard ManySAT, with the number of problems solved before timeout, and the associated average runtime. The right part reports results for the two dynamic policies. For each dynamic policy we provide $\bar{e}$, the average of the $e_{j\to i}$ observed during the computation. The last row provides for each method, the total number of problems solved, and the cumulated runtime. For the dynamic policies, it also presents the average of the $\bar{e}$ values.

At that point we have to stress that the static policy ($e = 8$) is optimal in the way that it gives the best average performance on this set of problems. We can observe that the static policy solves 83 problems while the dynamic policies aimdT and aimdTQ solve respectively 86 and 89 problems. Except on the *ibm_** and *manol_** families, the dynamic policies always exhibit a runtime better or equivalent to the static one. Unsurprisingly, when the runtime is significant but does not drastically improve over the static policy, the values of $\bar{e}$ are often close to 8, i.e., equivalent to the static size limit. When we consider the last row, we can see that the aimdT is faster than the aimdTQ. However, this last policy solves more problems. We can explain this as follows. The quality-based policy intensifies the search by favoring the exchange of clauses related to the current exploration of each unit. This intensification leads to the resolution of more difficult problems. However, it increases the runtime on easier instances where a more diversified search is often more beneficial. Overall these results are very good since our dynamic policies are able to outperform the best possible static tuning.

		ManySAT e=8		ManySAT aimdT			ManySAT aimdTQ		
family/instance	#inst	#Solved	time(s)	#Solved	time(s)	$\bar{e}$	#Solved	time(s)	$\bar{e}$
ibm_*	20	19	**204**	19	218	7	19	286	6
manol_*	10	10	**117**	10	**117**	8	10	205	7
mizh_*	10	6	762	7	746	6	**10**	**441**	5
post_*	10	9	325	9	**316**	7	9	375	7
velev_*	10	8	585	8	**448**	5	8	517	7
een_*	5	5	2	5	2	8	5	2	7
simon_*	5	5	111	5	84	10	5	**59**	9
bmc_*	4	4	7	4	7	7	4	**6**	9
gold_*	4	1	1160	1	**1103**	12	1	1159	12
anbul_*	3	2	742	**3**	**211**	11	3	689	11
babic_*	3	3	2	3	2	8	3	2	8
schup_*	3	3	129	3	**120**	5	3	160	5
fuhs_*	2	2	90	2	**59**	11	2	77	10
grieu_*	2	1	783	1	**750**	8	1	**750**	8
narain_*	2	1	786	1	**776**	8	1	792	8
palac_*	2	2	20	2	**8**	3	2	54	7
aloul-chnl11-13	1	0	1500	0	1500	11	0	1500	10
jarvi-eq-atree-9	1	1	70	1	69	25	1	**43**	17
marijn-philips	1	0	1500	**1**	1133	34	1	**1132**	29
maris-s03-gripper11	1	1	11	1	11	10	1	11	8
vange-col-abb313gpia-9-c	1	0	1500	0	1500	12	0	1500	12
Total/(average)	100	83	10406	86	9180	(10.28)	89	9760	(9.61)

Table 1: SAT-Race 2008, industrial problems

4.4 Crafted problems

We present here results on the crafted category (201 problems) of the 2007 SAT-competition. These problems are hand made and many of them are designed to beat all the existing SAT solvers. It contains for example Quasi-group instances, forced random SAT instances, counting, ordering and pebbling instances, social golfer problems, etc.

The scatter plot (in log scale) given in figure 3 (left hand side) illustrates the comparative results of the static and dynamic throughput version of ManySAT. The x-axis (resp. y-axis) corresponds to the CPU time tx (resp. ty) obtained by ManySAT e=8 (resp. ManySAT aimdT). Each dot with (tx, ty) coordinates corresponds to a SAT instance. Dots below (resp. above) the diagonal indicate that ManySAT aimdT is faster (resp. slower) than ManySAT e=8. The results clearly exhibit that the throughput based policies outperform the static policy on the crafted category. These improvements are illustrated in the figure 3 (right hand side) which shows the time in seconds needed to solve a given number of instances ($\#instances$). We can observe that both aimdT ($\bar{e} = 23.14$ with a peak at 94) and aimdTQ ($\bar{e} = 26.17$ with a peak at 102) solve 7 more problems than the static policy. Like for the previous problem category, aimdT remains faster than aimdTQ. We can explain this as follows. It seems that since by definition, these problems do not have a structure which can be advantageously exploited by an intensification process, the higher diversification provided by aimdT allows better performances.

4.5 The dynamic of $\bar{e}$

It is interesting to consider the evolution of $\bar{e}$, the average of the $e_{j \to i}$ observed during the computation. In Figure 1 it was shown on the 16 instances of the *AProVE07_** family that the size of learnt clauses was increasing over time.

We present in figure 4 the evolution of $\bar{e}$ with ManySAT aimdT on the same family. The evolution is given for each unit (core0 to core3), and the average of the units is also presented (average). We can see that our dynamic policy overcomes the disappearing of "small" clauses by the incremental raising of the pairwise limits. It presents a typical saw tooth behavior that represents the probe for throughput T. This Figure and our results on the industrial and crafted problems show that the evolution of the pairwise limits if not bounded, does not reach unacceptable levels.

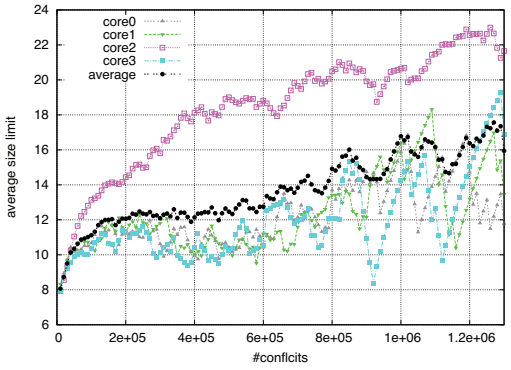


Figure 4: ManySAT aimdT : $\bar{e}$ on the AProVE07_* family

5 Previous works

We do not present parallel SAT solvers which predate the introduction of the modern DPLL architecture since these works did not use clause sharing.

The first parallel SAT solver based on a modern DPLL is Gradsat [Chrabakh and Wolski, 2003] which extends the zChaff solver with a master-slave model, and implements *guiding-paths* to divide the search space of a problem. These paths are represented by a set of unit clauses added to the original formula. Additionally, learned clauses are exchanged between all clients if they are smaller than a predefined limit on the number of literals.

[Blochinger *et al.*, 2003] uses an architecture similar to

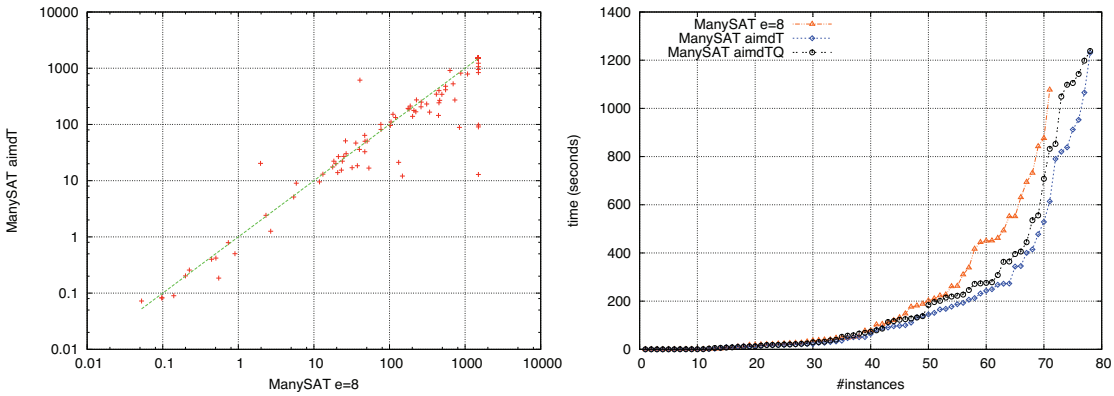


Figure 3: SAT-Competition 2007, crafted problems

Gradsat. However, a client incorporates a foreign clause if it is not subsumed by the current guiding-path constraints.

MiraXT is designed for shared memory multiprocessors systems [Lewis *et al.*, 2007]. It uses a divide and conquer approach where threads share a unique clause database which represents the original and the learnt clauses. Therefore in this system all the learnt clauses are shared.

pMiniSat uses a standard divide-and-conquer approach based on guiding-paths [Chu and Stuckey, 2008]. It exploits these paths to improve clause sharing. When considered with the knowledge of the guiding path of a particular thread, a clause can become small and therefore highly relevant. This allows pMiniSat to extend the sharing of clauses since a large clause can become small in another search context.

Let us remark that pMiniSat and MiraXT were respectively ranked second and third of the 2008 SAT-Race while ManySAT (presented in section 4.1) finished first.

6 Conclusion

We have presented two dynamic clause sharing policies for parallel SAT solving. They use an AIMD feedback control-based algorithm to dynamically adjust the size of shared clauses between any pair of processing units. Our first policy maintains an overall number of exchanged clauses (throughput) whereas the second additionally exploits the relevance quality of shared clauses. These policies have been devised as an efficient answer to the various flaws of the classical static size limit policy. The experimental results comparing our proposed dynamic policies against the static policy show important improvements for the state-of the-art parallel SAT solver ManySAT. It allows this solver to solve 6 more industrial and 7 more crafted problems. Our proposed framework opens interesting perspectives. For example, the design of new relevant quality measures for clause sharing is of great importance. It could benefit to sequential solver to improve their learnt base reduction strategy and as demonstrated by this work have an important impact in parallel SAT solving.

References

[Audemard *et al.*, 2008] G. Audemard, L. Bordeaux, Y. Hamadi, S. Jabbour, and L. Sais. A generalized framework for conflict analysis. In *Proc. of SAT*, pages 21–27, 2008.

[Blochinger *et al.*, 2003] W. Blochinger, C. Sinz, and W. Küchlin. Parallel propositional satisfiability checking with distributed dynamic learning. *Parallel Computing*, 29(7):969–994, 2003.

[Chiu and Jain, 1989] D-M Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Comp. Networks*, 17:1–14, 1989.

[Chrabakh and Wolski, 2003] W. Chrabakh and R. Wolski. GrADSAT: A parallel sat solver for the grid. Technical report, UCSB CS TR N. 2003-05, 2003.

[Chu and Stuckey, 2008] G. Chu and P. J. Stuckey. Pminisat: a parallelization of minisat 2.0. Technical report, SAT-Race 2008, solver description, 2008.

[Davis *et al.*, 1962] M. Davis, G. Logemann, and D. W. Loveland. A machine program for theorem-proving. *Comm. of the ACM*, 5(7):394–397, 1962.

[Eén and Sörensson, 2003] N. Eén and N. Sörensson. An extensible sat-solver. In *Proc. of SAT'03*, pages 502–518, 2003.

[Gomes *et al.*, 1998] C. P. Gomes, B. Selman, and H. Kautz. Boosting combinatorial search through randomization. In *Proc. AAAI'98*, pages 431–437, 1998.

[Hamadi *et al.*, 2009] Y. Hamadi, S. Jabbour, and L. Sais. ManySAT: a parallel SAT solver. *Journal on Satisfiability, Boolean Modeling and Computation (JSAT)*, to appear, 2009.

[Jacobson, 1988] V. Jacobson. Congestion avoidance and control. In *Proc. SIGCOMM '88*, pages 314–329, 1988.

[Lewis *et al.*, 2007] M. Lewis, T. Schubert, and B. Becker. Multithreaded sat solving. In *Proc. ASP DAC'07*, 2007.

[Marques-Silva and Sakallah, 1996] J. P. Marques-Silva and K. A. Sakallah. GRASP - A New Search Algorithm for Satisfiability. In *Proc. of IEEE/ACM CAD'96*, pages 220–227, 1996.

[Moskewicz *et al.*, 2001] M.W. Moskewicz, C.F. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient SAT solver. In *DAC'01*, pages 530–535, 2001.

Symmetries in Itemset Mining

Said Jabbour and **Lakhdar Sais** and **Yakoub Salhi** and **Karim Tabia**[1]

Abstract. In this paper, we describe a new framework for breaking symmetries in itemset mining problems. Symmetries are permutations between items that leave invariant the transaction database. Such kind of structural knowledge induces a partition of the search space into equivalent classes of symmetrical itemsets. Our proposed framework aims to reduce the search space of possible interesting itemsets by detecting and breaking symmetries between items. Firstly, we address symmetry discovery in transaction databases. Secondly, we propose two different approaches to break symmetries in a preprocessing step by rewriting the transaction database. This approach can be seen as an original extension of the symmetry breaking framework widely used in propositional satisfiability and constraint satisfaction problems. Finally, we show that Apriori-like algorithms can be enhanced by dynamic symmetry reasoning. Our experiments clearly show that several itemset mining instances taken from the available datasets contain such symmetries. We also provide experimental evidence that breaking such symmetries reduces the size of the output on some families of instances.

1 Introduction

The problem of mining frequent itemsets is well-known and essential in data mining, knowledge discovery and data analysis. It has applications in various fields and becomes fundamental for data analysis as datasets and datastores are becoming very large. Since the first article of Agrawal [15] on association rules and itemset mining, the huge number of works, challenges, datasets and projects show the actual interest in this problem (see [17] for a recent survey of works addressing this problem). Note that several data mining tasks are closely related to the itemset mining problem such as the ones of association rule mining, frequent pattern mining in sequence data, data clustering, episode mining, etc. Important progress has been achieved for data mining and knowledge discovery in terms of implementations, platforms, libraries, etc. Nevertheless, the majority of works developed solutions and tools specifically tuned and designed to achieve very specific goals on specific data mining tasks. The algorithmic issues represent the most important problem for the majority of works. As pointed out in [17], lot of works deal with designing highly scalable itemset mining algorithms for large scale datasets. The existing algorithms mainly differ in the way they explore the itemset search space, how the anti-monotonicity property is used and how the dataset is handled. Another important problem of itemset mining and data mining problems in general, concerns the huge size of the output, from which it is difficult to retrieve useful information. Consequently, for practical data mining, it is important to reduce the size of the output, by exploiting the structure of the itemsets data. Computing for example, closed, maximal, condensed, discriminative itemset patterns are some of the well-known and useful techniques.

Symmetry between items is another kind of structural information that might be recovered from the transaction databases and used to further reduce the size of the output by limiting the search space. Symmetries are a fundamental concept in computer science, mathematics, physics and many other domains. Many human artifacts (e.g. classroom in a university, aircraft seats, circuit patterns) and entities in nature (e.g. plants, molecules, DNA sequences, atoms) exhibits symmetries. Such symmetries allow us to reason and to understand more complex entities and systems. For example, this kind of structures has been widely exploited in constraint satisfaction (CSP) and propositional satisfiability (SAT) problems. As far as we know, symmetry reasoning has been first introduced by Krishnamurthy [9] to shorten resolution proofs in propositional logic. He shows that a resolution proof system augmented with the symmetry rule is more powerful than resolution. In [2], the authors showed how dynamic symmetry detection and elimination can lead to significant improvements of several automated deduction techniques. In constraint satisfaction problems, a weaker form of symmetry called value interchangeability is first introduced by E. Freuder in [6], while variable and value symmetry are studied in [14]. Symmetry breaking using additional constraints has been proposed independently by James Crawford in the context of first order [3] and propositional [4] logic theories and by Jean-François Puget in CSP [14]. More recent contributions clearly show that symmetry remains an active research area in both CSP and SAT.

To the best of our knowledge, in the context of data mining and particularly for the itemset mining problem, symmetries have not received much attention. Let us mention two related works addressing a particular case of general symmetries considered in this paper [13, 12]. Indeed, this restricted form of symmetry, called pairwise items symmetry, is obtained by exchanging two items, while leaving the remaining items unchanged. This is clearly a restriction of the general symmetry principle. In [13], an efficient ZBDD algorithm for detecting pairwise symmetries is proposed and the property of symmetric items in transaction database are discussed. Pairwise items symmetries, called clone items, have been proposed by Medina and Nourine [7] to explain why sometimes, the number of rules of a minimum cover of a relation is exponential in the number of items of the relation. More recently, in the context of constraint programming for k-pattern set mining, Guns et al. [8] used symmetry breaking constraints to impose a strict ordering on the patterns in the pattern set.

In this paper, we propose a theoretical framework for dealing with general symmetries in itemset mining problems. We first propose an adaptation of the symmetry detection method usually used in propositional satisfiability [3, 1] to transaction databases. As this hidden structure can be very helpful for reducing the search space, we show how they can be integrated dynamically in Apriori-like al-

[1] Univ Lille Nord de France, F-59000 Lille, France. UArtois, CRIL UMR CNRS 8188, F-62300 Lens, France. {jabbour, sais, salhi, tabia}@cril.univ-artois.fr

gorithms to prune the set of possible candidate patterns. Then, we propose two symmetry breaking approaches to eliminate symmetries in transaction databases in a preprocessing step. Our proposed symmetry breaking methods eliminate items from the original transaction database. The frequent itemsets generated using the new transaction database together with the symmetry group can be used to retrieve the whole set of frequent itemsets of the original transaction database. This can be seen as a form of condensed representation of the output. A condensed representation is originally proposed by Mannila and Toivenen in [10]. For frequent itemsets, a condensed representation is a collection of itemsets that still contains the same information.

On the practical side and to show the usefulness of exploiting symmetries in itemset mining problems, we present an experimental evaluation of both symmetry detection and symmetry breaking on some benchmark instances.

2 Preliminary definitions and notations

Let $\mathcal{I}$ be a set of *items*. A set $I \subseteq \mathcal{I}$ is called an *itemset*. A *transaction* is a couple (tid, I) where tid is the *transaction identifier* and I is an itemset. A *transaction database* is a finite set of transactions over $\mathcal{I}$ where for each two different transactions, they do not have the same transaction identifier. We note $\mathcal{T}_{id}(\mathcal{D}) = \{tid|\ (tid, I) \in \mathcal{D}\}$ the set of transaction identifiers associated to $\mathcal{D}$. We use $\mathcal{I}_{items}(O)$ to denote the set of all the items appearing in the syntactic object O (e.g. a transaction database, itemset, etc). We say that a transaction (tid, I) *supports* an itemset J if $J \subseteq I$.

The *cover* of an itemset I in a transaction database $\mathcal{D}$ is the set of identifiers of transactions in $\mathcal{D}$ supporting I: $\mathcal{C}(I, \mathcal{D}) = \{tid \mid (tid, J) \in \mathcal{D} \text{ and } I \subseteq J\}$. The *support* of an itemset I in $\mathcal{D}$ is defined by: $\mathcal{S}upp(I, \mathcal{D}) = |\mathcal{C}(I, \mathcal{D})|$. Moreover, the frequency of I in $\mathcal{D}$ is defined by: $\mathcal{F}(I, \mathcal{D}) = \frac{\mathcal{S}upp(I,\mathcal{D})}{|\mathcal{D}|}$.

Example 1 *Let us consider the following transaction database over the set of items* $\mathcal{I} = \{Spaghetti, Tomato, Parmesan, Beef, Olive\ oil, Mozzarella, Chili\ pepper, Anchovies, Eggs\}$*:*

tid	itemset
001	$Spaghetti, Olive\ oil, Tomato, Mozzarella$
002	$Spaghetti, Parmesan, Olive\ oil, Beef$
003	$Chili\ pepper, Anchovies$
004	$Eggs$

Table 1. An example of transaction database $\mathcal{D}$

For instance, we have $\mathcal{S}upp(\{Spaghetti, Olive\ \ oil\}, \mathcal{D})$= $|\{001, 002\}|$=2 *and* $\mathcal{F}(\{Spaghetti, Olive\ oil\}, \mathcal{D})$=0.5.

Let $\mathcal{D}$ be a transaction database over $\mathcal{I}$ and λ a minimal support threshold.

Proposition 1 (Anti-Monotonicity) *Let I_1 and I_2 be two itemsets such that $I_1 \subseteq I_2$. If $\mathcal{S}upp(I_2, \mathcal{D}) \geq \lambda$ then $\mathcal{S}upp(I_1, \mathcal{D}) \geq \lambda$.*

Definition 1 (Frequent Itemset Mining Problem) *The frequent itemset mining problem consists in computing the following set:*

$$\mathcal{FIM}(\mathcal{D}, \lambda) = \{I \subseteq \mathcal{I} \mid \mathcal{S}upp(I, \mathcal{D}) \geq \lambda\}$$

Definition 2 (Transaction Renaming) *Let $\mathcal{D}$ be a transaction database. A renaming f over $\mathcal{T}_{id}(\mathcal{D})$ is a bijective mapping from $\mathcal{T}_{id}(\mathcal{D})$ to $\mathcal{T}_{id}(\mathcal{D})$.*

We can extend a renaming f to $\mathcal{D}$ as follows: $f(\mathcal{D}) = \{(f(tid), I)|(tid, I) \in \mathcal{D}\}$.

3 Symmetry in Frequent Itemset Mining

Definition 3 (Permutation) *A permutation σ over $\mathcal{I}$ is a bijective mapping from $\mathcal{I}$ to $\mathcal{I}$.*

Let $\mathcal{D}$ be a transaction database. We can extend a permutation σ to $\mathcal{D}$ as follows: $\sigma(\mathcal{D}) = \{(tid, \sigma(I))|(tid, I) \in \mathcal{D}\}$ where $\sigma(I) = \{\sigma(a)|a \in I\}$.

We denote by $\mathcal{P}(\mathcal{I})$ the set of all the permutations over $\mathcal{I}$ and $\circ$ is the composition operation over the elements of $\mathcal{P}(\mathcal{I})$. It is easy to see that $(\mathcal{P}(\mathcal{I}), \circ)$ is a group where the identity permutation is a neutral element. $(S', \circ)$ is a sub-group of the group $(\mathcal{P}(\mathcal{I}), \circ)$ if and only if $(S', \circ)$ is a group and $S' \subseteq \mathcal{P}(\mathcal{I})$. A *generating set of the group* $(S, \circ)$ is a subset Σ of S such that every element of the group can be obtained from the combination of finitely many elements of Σ and their inverses, and we write $S =< \Sigma >$.

Each permutation σ can be represented by a set of cycles $c_1 \dots c_n$ where each cycle $c_i = (a_1, \dots, a_k)$ is a list of elements of $\mathcal{I}$ such that $\sigma(a_j) = a_{j+1}$ for $j = 1, \dots, k-1$, and $\sigma(a_k) = a_1$. In the sequel, for clarity reasons, we omit cycles of the form (a, a) in the description of permutations and symmetries,.

Definition 4 (Orbit) *Let $(S, \circ)$ be a sub-group of $(\mathcal{P}(\mathcal{I}), \circ)$. The orbit a^S of an item $a \in \mathcal{I}$ on which $(S, \circ)$ acts is $a^S = \{\sigma(a)|\sigma \in S\}$.*

When there is no ambiguity, we note a^S simply $[a]$.

Definition 5 (Symmetry) *Let $\mathcal{D}$ be a transaction database. A symmetry of $\mathcal{D}$ is a permutation $\sigma \in \mathcal{P}(\mathcal{I})$ such that there exists a transaction renaming f over $\mathcal{T}_{id}(\mathcal{D})$ where $\sigma(\mathcal{D}) = f(\mathcal{D})$ i.e. $f^{-1}(\sigma(\mathcal{D})) = \mathcal{D}$.*

Example 2 *Let us consider again the transaction database given in Table 1.* $\sigma = (Tomato, Parmesan)(Mozzarella, Beef)$ $(Chili\ pepper, Anchovies)$ *is a symmetry because $\sigma(\mathcal{D}) = f(\mathcal{D})$ where f is a transaction renaming defined as follows:*

$$f(x) = \begin{cases} 002 & \text{if x=001} \\ 001 & \text{if x=002} \\ x & \text{otherwise} \end{cases}$$

Let $\mathcal{S}(\mathcal{D})$ be the set of all the symmetries. One can see that $(\mathcal{S}(\mathcal{D}), \circ)$ is a sub-group of $(\mathcal{P}(\mathcal{I}), \circ)$.

Definition 6 *Let $\mathcal{D}$ be a transaction database. Two items $a, b \in \mathcal{I}_{items}(\mathcal{D})$ are symmetric if there exists a symmetry σ of $\mathcal{D}$ such that $\sigma(a) = b$.*

The following proposition is immediate:

Proposition 2 *Let $\mathcal{D}$ be a transaction database and $a \in \mathcal{I}_{items}(\mathcal{D})$. All the items in $a^{\mathcal{S}(\mathcal{D})}$ are symmetrical two by two.*

Proposition 3 *Let $\mathcal{D}$ be a transaction database, σ a symmetry of $\mathcal{D}$, λ a minimal support threshold and I an itemset. $I \in \mathcal{FIM}(\mathcal{D}, \lambda)$ iff $\sigma(I) \in \mathcal{FIM}(\mathcal{D}, \lambda)$.*

Proof: Since there exists a transaction renaming f such that $\sigma(\mathcal{D}) = f(\mathcal{D})$, it is easy to see that $\sigma(I) \in \mathcal{FIM}(f(\mathcal{D}), \lambda)$ and consequently $\sigma(I) \in \mathcal{FIM}(\mathcal{D}, \lambda)$. □

4 Symmetry Detection in Transaction Databases

In this section, we describe symmetry detection in transaction databases. One of the most popular means of discovering symmetries of a problem is to first convert the problem into a graph, and employ a general-purpose graph symmetry tool to uncover the symmetries [1]. These symmetries can then be reflected back into the original problem. The oldest and most established graph symmetry program is McKays Nauty [11]. Most of the available symmetry detection tools convert the original problem into a colored undirected graph, where vertices are labeled with colors. Such colored vertices are considered when searching for automorphism on the graph (i.e. vertices with different colors can not be permuted to each other).

Definition 7 *A colored undirected graph is a triplet $\mathcal{G} = (V, E, \eta)$ with vertex set V and edge set $E \in V^2$ and η is a function from V to N that associates a positive integer (a color) to each vertex.*

Definition 8 *A permutation σ of V is a symmetry of $\mathcal{G}$ iff $\sigma(\mathcal{G}) = \mathcal{G}$ or equivalently $\sigma(E) = E$.*

Definition 9 *A symmetry σ of $\mathcal{G}$ respects a partition π of V if for each $v \in V$, v and $\sigma(v)$ belong to the same cell of π. The set of all symmetries of $\mathcal{G}$ with respect to a partition π is called the automorphism group of $\mathcal{G}$ under π and is denoted $\mathcal{A}(\mathcal{G})_\pi$.*

We now show how a transaction database can be encoded as a colored undirected graph.

Definition 10 (From $\mathcal{D}$ to $\mathcal{G}$) *Let $\mathcal{I}$ be a set of items and $\mathcal{D}$ a transaction database over $\mathcal{I}$. We define the colored undirected graph $\mathcal{G}$ associated to $\mathcal{D}$ as $\mathcal{G}(\mathcal{D}) = (V, E, \eta)$ where $V = \mathcal{I} \cup \mathcal{T}_{id}(\mathcal{D})$, $E = \{(tid, i) | \exists (tid, I) \in \mathcal{D}, i \in I\}$ and $\forall v \in V$,*

$$\eta(v) = \begin{cases} 0 & \text{if } v \in \mathcal{I} \\ 1 & \text{otherwise } v \in \mathcal{T}_{id}(\mathcal{D}) \end{cases}$$

Let us note that $\pi = \{\{v | v \in V, \eta(v) = 0\}, \{v | v \in V, \eta(v) = 1\}\}$ is the initial partition of V. Several refinements have been proposed to improve the partition π. The goal of these refinements is to reduce the search space by distinguishing as much as possible non symmetric vertices using vertex invariant such as vertex degree (e.g. [5]).

The conversion of the transaction database $\mathcal{D}$ given in Example 1 to a colored undirected graph $\mathcal{G}(\mathcal{D})$ is depicted in Figure 1. For simplicity, we use the first letter of each item name in lower case (e.g. e for Eggs). In the figure we distinguish two kinds of nodes, items represented by circles (color 1) and transaction identifiers represented by rectangles (color 2).

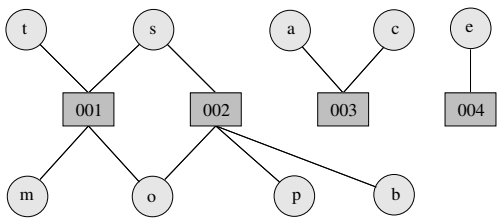


Figure 1. From transaction database $\mathcal{D}$ to colored graph $\mathcal{G}(\mathcal{D})$

With this construction, there is a one to one mapping between symmetries in the transaction database and the automorphisms of the colored undirected graph. Given $\mathcal{G}(\mathcal{D})$ and an initial partition π, the goal is to compute $\mathcal{A}(\mathcal{G}(\mathcal{D}))_\pi$. Using NAUTY [11], one can find the automorphisms that leave $\mathcal{G}(\mathcal{D})$ invariant. For example, $A = (t,p)(m,b)(c,a)(o)(s)(e)$ $[(001,002)(003)(004)]$ is an automorphism of $\mathcal{G}(\mathcal{D})$. The corresponding symmetry σ is obtained from A by projection on the set of items $\mathcal{I}$ of $\mathcal{D}$.

5 Symmetry Pruning

In this section, we show how symmetries can be used by classical enumeration algorithms such as Apriori to reduce the set of possible candidate patterns.

The Apriori algorithm is an algorithm for mining frequent itemsets for association rules [15]. It proceeds by a level-wise search of the elements of $\mathcal{FIM}(\mathcal{D}, \lambda)$. Namely, it starts by computing the elements of $\mathcal{FIM}(\mathcal{D}, \lambda)$ of size one. Then, assuming the elements of $\mathcal{FIM}(\mathcal{D}, \lambda)$ of size n known, it computes a set of candidates of size $n+1$ so that I is a candidate if and only if all its subsets are in $\mathcal{FIM}(\mathcal{D}, \lambda)$. This procedure is iterated until no more candidate is found. Let $\mathcal{D}$ be a transaction database such that $\mathcal{I}_{item}(\mathcal{D}) =$

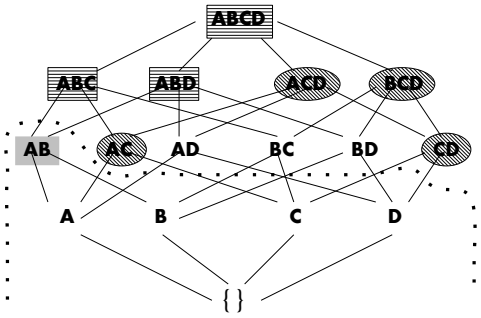


Figure 2. Symmetry pruning of the search space

$\{A, B, C\}$, $\sigma = (B, C)$ and $\sigma' = (A, C)(B, D)$ two symmetries of $\mathcal{D}$ and λ a minimal support threshold. We consider that we are in an intermediary step of an Apriori like algorithm and we have $\{A\}, \{B\}, \{C\}, \{D\} \in \mathcal{FIM}(\mathcal{D}, \lambda)$ and $\{A, B\} \notin \mathcal{FIM}(\mathcal{D}, \lambda)$. Using the anti-monotonicity property (Proposition 1), we know that for all itemset I containing $\{A, B\}$, $I \notin \mathcal{FIM}(\mathcal{D}, \lambda)$, then we obtain $\{A, B, C\}, \{A, B, D\}, \{A, B, C, D\} \notin \mathcal{FIM}(\mathcal{D}, \lambda)$. Moreover, using the symmetries σ and σ', we know that $\{A, C\} \notin \mathcal{FIM}(\mathcal{D}, \lambda)$ and $\{C, D\} \notin \mathcal{FIM}(\mathcal{D}, \lambda)$. Indeed, this is a consequence of Proposition 3, $\sigma(\{A, B\}) = \{A, C\}$ and $\sigma'(\{A, B\}) = \{C, D\}$. Finally, using the anti-monotonicity property, we deduce that also $\{A, C, D\}, \{B, C, D\} \notin \mathcal{FIM}(\mathcal{D}, \lambda)$. This is illustrated in Figure 2 where the inclined dashed circles (nodes) correspond to the itemsets pruned by symmetry.

6 Symmetry Breaking

In this section, we propose two approaches for breaking symmetries in a preprocessing step. Our proposed symmetry breaking methods eliminate items from the original transaction database. The frequent itemsets generated using the new transaction database together with the symmetry group can be used to retrieve the whole set of frequent itemsets of the original transaction database. This can be seen as a form of condensed representation of the output.

6.1 Orbit-based Symmetry Breaking

Definition 11 *Let D be a transaction database and $[a]$ an orbit following $(\mathcal{S}(\mathcal{D}), \circ)$. $\mathcal{D}^a$ is the transaction database obtained from $\mathcal{D}$ as follows: $\forall b \in [a]$ such that $b \neq a$ and for every transaction $T = (tid, I) \in \mathcal{D}$, if $b \in I$ and $a \notin I$, then b is removed from I in the transaction T.*

Let D be a transaction database, a an item and λ a minimal support threshold, we denote by $\mathcal{FIM}^a(\mathcal{D}, \lambda)$ all the elements of $\mathcal{FIM}(\mathcal{D}, \lambda)$ containing a.

Proposition 4 *Let D be a transaction database, $a, b \in \mathcal{I}_{items}(\mathcal{D})$ such that they are in the same orbit following the group $(\mathcal{S}(\mathcal{D}), \circ)$ and λ a minimal support threshold. There exists a symmetry σ such that $\mathcal{FIM}^b(\mathcal{D}, \lambda) = \sigma(\mathcal{FIM}^a(\mathcal{D}, \lambda))$.*

Proof: Since a and b are in the same orbit, there exists a symmetry σ such that $b = \sigma(a)$. Using Proposition 3, we have (1) $\mathcal{FIM}^b(\mathcal{D}, \lambda) \supseteq \sigma(\mathcal{FIM}^a(\mathcal{D}, \lambda))$ and (2) $\mathcal{FIM}^a(\mathcal{D}, \lambda) \supseteq \sigma^{-1}(\mathcal{FIM}^b(\mathcal{D}, \lambda))$. Using (2), we have $\sigma(\mathcal{FIM}^a(\mathcal{D}, \lambda)) \supseteq \mathcal{FIM}^b(\mathcal{D}, \lambda)$. Therefore, $\mathcal{FIM}^b(\mathcal{D}, \lambda) = \sigma(\mathcal{FIM}^a(\mathcal{D}, \lambda))$ holds. □

Proposition 5 *Let D be a transaction database, $[a]$ an orbit following $(\mathcal{S}(\mathcal{D}), \circ)$ of size n and λ a minimal support threshold. Then, there exist n symmetries $\sigma_1, \ldots, \sigma_n$ such that $\mathcal{FIM}(\mathcal{D}, \lambda) = \mathcal{FIM}(\mathcal{D}^a, \lambda) \cup \sigma_1(\mathcal{FIM}(\mathcal{D}^a, \lambda)) \cup \ldots \cup \sigma_n(\mathcal{FIM}(\mathcal{D}^a, \lambda))$.*

Proof: By induction on n (it is a consequence of Proposition 4). □

Let $\mathcal{D}$ be a transaction database, $\mathcal{O} = \{[a_1], \ldots, [a_k]\}$ the set of all the orbits following $(\mathcal{S}(\mathcal{D}), \circ)$ and λ a minimal support threshold. Then, the transaction database $(\ldots(\mathcal{D}^{a_1})\ldots)^{a_k}$ and $(\mathcal{S}(\mathcal{D}), \circ)$ are sufficient to compute $\mathcal{FIM}(\mathcal{D}, \lambda)$. This is a direct consequence of Proposition 5.

Let us take again the transaction database given in Example 1 and the associated symmetry σ given in section 2. Using the orbit $[Tomato]^\sigma$, we eliminate $Parmesan$ in transaction 002 as it does not contain the item $Tomato$. While using the orbit $[Mozarella]^\sigma$, we eliminate the item $Beef$ from transaction 002 as it does not contain $Mozarella$. However, using the orbit $[Chili\ pepper]^\sigma$ the item $Anchovies$ can not be eliminated from the transaction 003 as both items, $Chili\ pepper$ and $Anchovies$, occur together in the same transaction.

6.2 Itempair-Based Symmetry Breaking

In this section, we propose another method to reduce the search space of possible interesting itemsets by detecting and breaking symmetries between itempairs.

We only consider the particular case of the symmetries with binary cycles in our description. Moreover, we fix an order $\preceq$ on $\mathcal{I}$ and we denote by $min(I)$, with I an itemset, the least item of I following $\preceq$.

Definition 12 (Ordered Symmetry) *Let $\mathcal{D}$ be a transaction database and $\sigma = (a_1, a'_1)\cdots(a_k, a'_k)$ be a symmetry of $\mathcal{D}$. σ is an ordered symmetry iff for all $i, j \in \{1, \ldots, k\}$ such that $i \leq j$, $a_i \preceq a'_i$ and $a_i \preceq a_j$.*

In the rest of this paper, we consider all symmetries as ordered. Let $\mathcal{D}$ be a transaction database and $\sigma = (a_1, a'_1)\cdots(a_k, a'_k)$ be a symmetry of $\mathcal{D}$, the set of items $\mathcal{L}(\sigma)$ ($\mathcal{L}$ for Left) is defined by $\mathcal{L}(\sigma) = \{a_1, \ldots, a_k\}$, $\mathcal{R}(\sigma)$ ($\mathcal{R}$ for Right) by $\mathcal{R}(\sigma) = \{a'_1, \ldots, a'_k\}$ and $\mathcal{H}(\sigma)$ by $\mathcal{H}(\sigma) = \{\{a_i, a'_j\} | a_i \in \mathcal{L}(\sigma), a'_j \in \mathcal{R}(\sigma)$ and $i \leq j\}$. Moreover, let E be a set of itempairs, we denote by $E(\sigma)$ the following set of itempairs: $\sigma(E) = \{\{x, y\} \in E | x, y \notin \mathcal{R}(\sigma)$ or $\{x, y\} \in \mathcal{H}(\sigma)\}$. Intuitively, $\sigma(E)$ is obtained from E by removing itempairs that can be recovered using σ.

In the following two propositions, we particularly show how some itempairs can be recovered from the others using symmetries. This will allow us to reduce transaction databases by removing items.

Proposition 6 *Let $\mathcal{D}$ be a transaction database, $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ a set of symmetries of $\mathcal{D}$, $E_\Sigma = \{\{x, y\} | x, y \in \mathcal{I}_{items}(\Sigma)\}$ and $W_\Sigma = \sigma_1(\sigma_2(\cdots(\sigma_n(E_\Sigma))\cdots))$. Then, for all $\{x, y\} \in \mathcal{I}_{items}(\Sigma)$, there exist $\{x', y'\} \in W_\Sigma$ and a symmetry σ such that $\sigma(\{x', y'\}) = \{x, y\}$.*

Proof: One can easily prove that $W_\Sigma = \sigma_1(E_\Sigma) \cap \cdots \cap \sigma_n(E_\Sigma)$ (by induction on the value of n). Let $x', y' \in \mathcal{I}_{items}(\Sigma)$ such that **(1)** there exists a symmetry σ where $\sigma(\{x', y'\}) = \{x, y\}$, **(2)** and for all symmetry σ', there are no $x'', y'' \in \mathcal{I}_{items}(\Sigma)$ such that $x'' \prec x'$ (that means in particular $x' \prec y'$) and $\sigma'(\{x'', y''\}) = \{x, y\}$. Intuitively, $\{x', y'\}$ is the smallest itempair allowing to recover $\{x, y\}$. If $\{x', y'\} \notin W_\Sigma$, then there exists $\sigma_i \in \Sigma$ such that $x', y' \in \mathcal{R}(\sigma_i)$ or $\{x', y'\} \notin \mathcal{H}(\sigma_i)$. In both cases, there exists $\{x'', y''\} \in \sigma_i(E_\Sigma)$ such that $x'' \prec x'$ and $\sigma_i(\{x'', y''\}) = \{x', y'\}$. Thus, $\sigma(\sigma_i(\{x'', y''\})) = \sigma(\{x', y'\}) = \{x, y\}$ holds and we get a contradiction. □

Proposition 7 *Let $\mathcal{D}$ be a transaction database, $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ be a set of symmetries of $\mathcal{D}$, $E = \{\{x, y\} | x, y \in \mathcal{I}_{items}(\mathcal{D})\}$ and $W = \sigma_1(\sigma_2(\cdots(\sigma_n(E_\Sigma))\cdots))$. The three following properties are satisfied:*

1. *there exists $x \in \mathcal{I}_{items}(W)$ such that for all $y \in \mathcal{I}_{items}(\mathcal{D}) \backslash \mathcal{I}_{items}(\Sigma)$, $\{x, y\} \in W$;*
2. *for all $x \in \mathcal{I}_{items}(\mathcal{D})$, there exists $y \in \mathcal{I}_{items}(W)$ and a symmetry σ such that $x = \sigma(y)$;*
3. *for all $\{x, y\} \in E \backslash W$, there exists $\{x', y'\} \in W$ and a symmetry σ such that $\sigma(\{x', y'\}) = \{x, y\}$.*

Proof: The *first property* is proved by taking $x = min(\mathcal{I}_{items}(\Sigma))$. Indeed, by construction of W we have $min(\mathcal{I}_{items}(\Sigma))$ is in $\mathcal{I}_{items}(W)$ and for all $y \in \mathcal{I}_{items}(\mathcal{D}) \backslash \mathcal{I}_{items}(\Sigma)$, $\{x, y\} \in W$.
The *second property* is proved by using the fact that the minimum of each set of the items symmetrical two by two is in W.
Let us now prove the *third property*. If $x \notin \mathcal{I}_{items}(\Sigma)$ (resp. $y \notin \mathcal{I}_{items}(\Sigma)$) then by using the second property we know that there exist $y' \in W$ (resp. $x' \in W$) and a symmetry σ such that $\sigma(\{x, y'\}) = \{x, y\}$ (resp. $\sigma(\{x', y\}) = \{x, y\}$). Otherwise, x and y are in $\mathcal{I}_{items}(\Sigma)$ and, by using Proposition 6, the property is satisfied. □

Definition 13 *Let $\mathcal{D}$ be a transaction database, $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ a set of symmetries of $\mathcal{D}$, $E = \{\{x, y\} | x, y \in \mathcal{I}_{items}(\mathcal{D})\}$ and $W = \sigma_1(\sigma_2(\cdots(\sigma_n(E_\Sigma))\cdots))$. $\mathcal{D}[W]$ is the transaction database obtained from $\mathcal{D}$ as follows: for all $a \in \mathcal{I}_{items}(\mathcal{D})$ and for all transaction $T = (tid, I) \in \mathcal{D}$ with $|I| \geq 2$ and $a \in I$, if for all $b \in I$ we have $\{a, b\} \notin W$, then a is removed from I in the transaction T.*

Let λ be a minimal support threshold. From Proposition 3 and Proposition 7 (Property 3), we deduce that the transaction database $\mathcal{D}[W]$ and Σ are sufficient to compute $\mathcal{FIM}(\mathcal{D}, \lambda)$.

Let us show an example where Itempair-Based Symmetry Breaking approach eliminates more items in the transaction database than the Orbit-based Symmetry Breaking one. Let $\mathcal{D}$ be a transaction database and $\sigma_1 = (a, b)$, $\sigma_2 = (b, c)$ and $\sigma_3 = (c, d)$ three symmetries of $\mathcal{D}$. It is easy to see that a, b, c and d are in the same orbit. Thus, using orbit-based symmetry breaking, $\mathcal{FIM}(\mathcal{D}, \lambda)$ can be computed from the transaction database obtained from $\mathcal{D}$ by removing b, c and d from all transactions that do not contain a. However, using the itempair-based symmetry breaking, we can remove more items from $\mathcal{D}$. Indeed, in $\mathcal{D}[\sigma_1(\sigma_2(\sigma_3(E)))]$, with $E = \{\{x, y\}|x, y \in \mathcal{I}_{items}(\mathcal{D})\}$, c and d are removed from all transactions, and b is removed from all transactions that do not contain a.

7 Experimental evaluation

Our experimental studies are carried out on both real, public and simulated datasets widely used in the data-mining community. We provide several experiments in order to show the improvements that can be achieved using our symmetry breaking-based itemset mining approach. In particular, we provide experiments on challenging high dimensional data with thousands of items.

7.1 Experimentation setup

In our study, we carried out a series of three experimentations using different datasets:

1. Simulated datasets: In this experimentation, we use the well-known IBM itemset data generator to generate data with different features (such as dataset size, density, etc.). Note that this generator is open source and publically available[2]. We generated different datasets with different parameters regarding the number of transactions, average number of items per transaction, number of different items, etc. In Table 2, we provide the features of each simulated dataset.
2. Public datasets: The datasets used in this evaluation are from the well-known itemset FIMI repository [3]. However, the datasets from FIMI are mostly simulated and not high dimensional. Therefore, we used $SIDO1$, a dataset containing pharmacology real data form the Causality challenge[4]. This dataset contains massive data of 12678 transactions with 4932 items.
3. Real datasets: In this experimentation, we used real data regarding intrusion detection alerts. The raw data is collected from real and recent alert log files produced by Snort intrusion detection system (IDS) [5] monitoring a university campus network. These alert logs represent three months activity. The alert log files are preprocessed into alert windows. Each alert window represents 1 hour of alerts in dataset $Alert1h$ and 8 hours in dataset $Alert8h$. In the obtained datasets, each transaction represents the alerts triggered during an alert window.

In our experiments, we exploit Saucy[6], a new implementation of the Nauty system. It is originally proposed in [1] and significantly improved in [5]. The latest version of Saucy outperforms all the existing tools by many orders of magnitude, in some cases improving runtime from several days to a fraction of a second.

Because of lack of space, we only provide experiments using itempair-based symmetry breaking approach.

In the sequel, we note, $|\mathcal{D}|$ (resp. $|\mathcal{I}|$) the number of transactions (resp. different items) in the dataset $\mathcal{D}$. $|Iocc|$ denotes the total number of item occurrences in $\mathcal{D}$, while $\%\ dens$ represents the density of the input transaction dataset. $time(s)$ denotes the cumulated time in seconds required to search and break symmetries. $|Irem|$ represents the number of removed item occurrences from the initial dataset. #sym denotes the number of symmetries discovered in the dataset. Finaly, $Iset$ and $Isset$ corresponds to the number of frequent itemsets obtained on the transaction dataset before and after breaking symmetries respectively.

7.2 Results on simulated datasets

Table 2 provides details on the generated datasets and the results of our symmetry breaking approach.

Dataset	$\|\mathcal{D}\|$	$\|\mathcal{I}\|$	$\|Iocc\|$	$\%dens$	time(s)	$\|Irem\|$	#sym
Dataset1	100	1006	1020	0.099%	1.29	894	899
Dataset2	778	3506	4089	0.011%	1.67	2774	2642
Dataset3	6110	13680	30886	0.014%	2.64	10577	8371
Dataset4	7822	17266	40506	0.023%	2.89	7520	9415
Dataset5	10000	26445	200102	0.075%	3.87	8330	9283
Dataset6	60808	27141	246817	0.073%	7.72	13396	9074
Dataset7	100	27151	199869	7.37%	8.5	13032	11607

Table 2. Details and results on simulated datasets

Table 2 clearly shows the significant improvements (in terms of the number of removed item occurrences) that can be performed as the number of symmetries grows. For instance, in experimentation on $Dataset3$ with 6110 transactions, the number of symmetries is 8371. The second point to mention is that on the generated datasets, the number of symmetries is important in datasets with low densities.

In Table 3, we provide results pointing out the correlation between the size of the preprocessed datasets (in terms of item occurrences) and the size of outputs (in terms of frequent itemsets) with the number of discovered symmetries. Note that this experiment is carried out on datasets we simulated specifically to involve interesting symmetries. To compare the output size, we use the $LCMv3$ itemset mining implementation[7] [16] on each dataset before and after preprocessing.

$\#sym$	$\|\mathcal{I}\|$	$\|Irem\|$	λ	$\|Iset\|$	$\|Isset\|$
6	900	421	5	1015830	4370
10	5586	3711	90	17467891	12952
16	37350	32482	375	2679034	467
22	148471	128471	900	1462982	1232
26	311850	289847	1450	22156	33

Table 3. Comparison of the number of removed item occurences and output size in the number of discovered symmetries.

One can see for instance in Table 3 that as the number of symmetries in the dataset increases, the size of the preprocessed dataset is significantly reduced. In particular, in our simulated dataset, when the number of symmetries equals 26, the item occurrences is reduced from 311850 to 66272 (ensuring a reduction rate of 78%). Note also that the output size (see the number of frequent itemsets $|Iset|$ before preprocessing and the number frequent itemsets $|Isset|$ on the preprocessed dataset) is significantly reduced on all the datasets of Table 3. The output size is computed for each dataset using the same support λ.

[2] IBM Quest Market-Basket Synthetic Data Generator: http://sourceforge.net/projects/ibmquestdatagen/

[3] Frequent Itemset Mining Implementations Repository: http://fimi.ua.ac.be/

[4] http://www.causality.inf.ethz.ch/challenge.php

[5] http://snort.org

[6] Saucy2: Fast symmetry discovery http://vlsicad.eecs.umich.edu/BK/SAUCY/

[7] http://research.nii.ac.jp/~uno/codes.htm

7.3 Results on public datasets

In Table 4, we provide details on the used datasets and the obtained results. The results of Table 4 show that in the tested public datasets,

Dataset	$\|\mathcal{D}\|$	$\|\mathcal{I}\|$	$\|\mathcal{I}occ\|$	$\%dens$	$\|Irem\|$	#sym
Mushroom	8124	186852	198169866	13%	1208	11
Retail	88162	908576	4106009	0.005%	89	217
BMS-WebView2	77512	3341	358278	0.14%	4	10
BMS-POS	515597	1658	3367020	0.4%	3	16
SIDO1	9297	4926	4514145	9.85%	568	190

Table 4. Details and results on public datasets

symmetries can be found. For instance, the $SIDO1$ dataset contains 190 symmetries allowing to remove 568 item occurrences from the initial dataset. However, the number of symmetries is not as important as in some simulated datasets of Table 2. Also, the number of removed item occurrences is not significant, consequently the size of the outputs in terms of the number of frequent itemsets is not significantly reduced.

7.4 Results on real datasets

In Table 5, we find details on the used alert datasets and the results of our approach. In Table 5, we see that our datasets regarding real intrusion detection alerts contain some symmetries. More importantly, on the dataset $Alert8h$, the size of the output in terms of the number of frequent itemsets (with a support λ=60) is decreased by a ratio of 50%. The two datasets have different densities and the number of removed items does not depend only on the number of discovered symmetries but also on the nature of the data itself.

Dataset	$\|\mathcal{D}\|$	$\|\mathcal{I}\|$	$\|\mathcal{I}occ\|$	$\%dens$	$\|Irem\|$	#sym
Alert1h	22203	167	27049	0.73%	86	23
Alert8h	123	167	3453	16.81%	248	19

Table 5. Details and results on the alert dataset

Let us summarize the experimental evaluation of our symmetry framework for frequent itemset mining problems. From the obtained results, we can make several interesting observations:

- Several datasets contain symmetries and are discovered by our symmetry detection method in a reasonable amount of time.
- The size of the output (number of frequent itemsets) can be significantly reduced by our symmetry-based framework.
- The reduction rate in terms of the number of eliminated item occurrences in the transaction database does not depend only on the number of found symmetries but also on their form. Indeed, some of the discovered symmetries are made of permutations of items in the same transactions, leading to no reduction using our symmetry breaking approaches. Such kind of symmetries can be better exploited in a dynamic way by Apriori-like algorithms as explained in Section 5 (symmetry pruning). This issue will be a subject of future investigations. The goal is to combine both symmetry breaking and symmetry pruning in the same framework.

8 Conclusion and Future Works

Symmetry is an important structural property widely exploited to reduce the search space of many combinatorial problems. In this paper, we have presented the theoretical foundation for discovering and using symmetries in the context of itemset mining problems. This study is important for several reasons. First, this kind of structures can be present in structured data and can be exploited for reducing both the search space and the size of the output. Secondly, even if such output is reduced, the eliminated combinatorial structures can be recovered by symmetry. Symmetries can also be used to help the user for computing either non-symmetric or symmetric patterns. The symmetries discovered from a given set of data, might be valuable in analyzing data themselves. Our theoretical framework includes symmetry detection, symmetry breaking and symmetry pruning. Our experimental results, demonstrate that this kind of structure can be hidden in some classes of transaction databases. We also show that when symmetries are present, their exploitation leads to interesting reduction of the output size.

The symmetry framework proposed in this paper for the itemset mining problem opens interesting research directions in data mining in general. We plan to extend our symmetry framework to other data mining problems such us sequence, tree or graph mining. In the context of itemset mining, there remains many rooms for future improvements. The integration of symmetry pruning in Apriori-like algorithm is clearly an important issue as many datasets contain symmetries between items appearing in the same transactions.

REFERENCES

[1] Fadi A. Aloul, Arathi Ramani, Igor L. Markov, and Karem A. Sakallah, 'Solving difficult instances of boolean satisfiability in the presence of symmetry', Transactions on Computer Aided Design, (2003).

[2] B. Benhamou and L. Sas, 'Tractability through symmetries in propositional calculus', Journal of Automated Reasoning, **12**(1), 89–102, (1994).

[3] J. Crawford, 'A theorical analysis of reasoning by symmetry in first order logic', in Workshop on Tractable Reasonning (AAAI'92), pp. 17–22, (1992).

[4] J. Crawford, M. L. Ginsberg, E. Luck, and A. Roy, 'Symmetry-breaking predicates for search problems', in Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR'96), 148–159, Morgan Kaufmann, (1996).

[5] Paul T. Darga, Karem A. Sakallah, and Igor L. Markov, 'Faster symmetry discovery using sparsity of symmetries', in Proceedings of the 45th Design Automation Conference (DAC'08), pp. 149–154, (2008).

[6] Eugene C. Freuder, 'Eliminating interchangeable values in constraint satisfaction problems', in Proceedings of the 9th National Conference on Artificial Intelligence (AAAI'91), pp. 227–233, (1991).

[7] Alain Gély, Raoul Medina, Lhouari Nourine, and Yoan Renaud, 'Uncovering and reducing hidden combinatorics in guigues-duquenne bases', in Proceedings of the third International ConferenceFormal Concept Analysis (ICFCA'05), pp. 235–248, (2005).

[8] Tias Guns, Siegfried Nijssen, and Luc de Raedt, 'k-pattern set mining under constraints', IEEE TKDE, **99**(PrePrints), (2011).

[9] Balakrishnan Krishnamurthy, 'Shorts proofs for tricky formulas', Acta Informatica, **22**, 253–275, (1985).

[10] H. Mannila and H. Toivonen, 'Multiple uses of frequent sets and condensed representations', in KDD, pp. 189–194, (1996).

[11] B. D. McKay, 'Practical graph isomorphism', Congressus Numerantium, **30**, 47–87, (1981).

[12] Raoul Medina, Caroline Noyer, and Olivier Raynaud, 'Efficient algorithms for clone items detection', in CLA'05, pp. 70–81, (2005).

[13] Shin-ichi Minato, 'Symmetric item set mining based on zero-suppressed bdds', in DS'06, pp. 321–326, (2006).

[14] Jean-Francois Puget, 'On the satisfiability of symmetrical constrained satisfaction problems', in Proceedings of the 7th International Symposium on Methodologies for Intelligent Systems (ISMIS'93), pp. 350–361, (1993).

[15] T. Imielinski R. Agrawal and A. N. Swami, 'Mining association rules between sets of items in large databases', in ACM SIGMOD International Conference on Management of Data, pp. 207–216, Baltimore, (1993). ACM Press.

[16] H. Arimura T. Uno, M. Kiyomi, 'Lcm ver.3: Collaboration of array, bitmap and prefix tree for frequent itemset mining', in OSDM Workshop on Frequent Pattern Mining Implementations, (2005).

[17] A. Tiwari, RK Gupta, and DP Agrawal, 'A survey on frequent pattern mining: Current status and challenging issues', Inform. Technol. J, **9**, 1278–1293, (2010).

Mining-Based Compression Approach of Propositional Formulae

Said Jabbour, Lakhdar Sais, Yakoub Salhi
CRIL - CNRS, University of Artois
F-62307 Lens Cedex, France
{jabbour, sais, salhi}@cril.fr

Takeaki Uno
National Institute of Informatics
2-1-2, Hitotsubashi, Chiyoda-ku
Tokyo 101-8430, Japan
uno@nii.jp

ABSTRACT

In this paper, we propose a first application of data mining techniques to propositional satisfiability. Our proposed mining based compression approach aims to discover and to exploit hidden structural knowledge for reducing the size of propositional formulae in conjunctive normal form (CNF). It combines both frequent itemset mining techniques and Tseitin's encoding for a compact representation of CNF formulae. The experimental evaluation of our approach shows interesting reductions of the sizes of many application instances taken from the last SAT competitions.

Categories and Subject Descriptors

F.4.1 [**Mathematical logic and formal languages**]: Mathematical Logic—*Logic and constraint programming*; H.2.8 [**Database management**]: Database applications—*Data mining*

Keywords

Compression; Data mining; Propositional satisfiability and modeling

1. INTRODUCTION

Propositional satisfiability (SAT) i.e., the problem of checking whether a Boolean formula in conjunctive normal form (CNF) is satisfiable or not, became a core technology in many application domains, such as formal verification, planning and various new applications derived by the recent impressive progress in practical SAT solving. SAT has gained a considerable audience with the advent of a new generation of solvers able to solve SAT instances with millions of variables and clauses [9, 4]. Today, these solvers represent an important low-level building block for many important fields, e.g., SAT modulo theory, Theorem proving, Model checking, Quantified Boolean formulas, Maximum Satisfiability, Pseudo Boolean, etc. In addition to the traditional applications of SAT to hardware and software formal verification, this impressive progress led to increasing use of SAT technology to solve new real-world applications such as in bioinformatics, cryptography and data mining.

Propositional formulae in CNF is the standard input format for propositional satisfiability. Indeed, most of the state-of-the-art SAT solvers are based on this normal form. Such convenient CNF representation is derived from a general Boolean formula using the well-known Tseitin encoding [14]. In practice, the general Boolean formula itself is usually derived from high level language in which problem structure is explicit.

Two important flaws were identified on the CNF form and largely discussed in the literature (e.g. [6]). First, it is often argued that by encoding arbitrary propositional formulae in CNF, structural properties of the original problem are not reflected in the CNF formula. Secondly, even if such translation is linear in the size of the original formula, a huge CNF formula might result when encoding real-world problems. Some instances exceed the capacity of the available memory, and even if the instance can be stored, the time needed for reading the input instance might be higher than its solving time. The obstacle here is not the potential difficulty of the instance, but its size. The growing success obtained in solving real-world SAT problems highlights a real transition to industrial and commercial scale. This results in a rapid growth in the size of the CNF instances encoding real-world problems. Consequently, the design of new efficient models for representing and for solving SAT instances of very large sizes ("Big" instances) is clearly an important challenge.

To address this problem, developing a more compact representation of CNF formulae is an interesting research issue. By compact encoding of formulae, we have in mind a representation model which through its use of structural knowledge results in the most compact possible formula equivalent with respect to satisfiability. By structural knowledge, we mean patterns that can be recognized or discovered. To be useful, such algorithm must be incremental.

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

http://dx.doi.org/10.1145/2505515.2505576.

Two promising models were proposed these last years. The first, proposed by H. Dixon et al [3], uses group theory to represent several classical clauses by a single clause called an "augmented clause". The second model was proposed by M. L. Ginsberg et al [5], called QPROP ("quantified propositional logic"), which may be seen as a propositional formula extended by the introduction of quantifications over finite domains, i.e. first order logic limited to finite types and without functional symbols. The problem rises in finding efficient solving techniques of formulae encoded using such models.

More recently, an original approach for compacting sets of binary clauses was proposed by J. Rintanen in [12]. Binary clauses are ubiquitous in propositional formulae that represent real-world problems ranging from model-checking problems in computer-aided verification to AI planning problems. In [12], using auxiliary variables, it is shown how constraint graphs that contain big cliques or bi-cliques of binary clauses can be represented more compactly than the quadratic and explicit representation. The main limitation of this approach lies in its restriction to particular sets of binary clauses whose constraints graph represents cliques or bi-cliques. Such particular regularities can be caused by the presence of an at-most-one constraint over a subset of Boolean variables, forbidding more than one of them to be true at a time.

In the data mining community, several models and techniques for discovering interesting patterns in large databases has been proposed in the last few years. The problem of mining frequent itemsets is well-known and essential in data mining, knowledge discovery and data analysis. Since the first article of Agrawal [1] on association rules and itemset mining, the huge number of works, challenges, datasets and projects show the actual interest in this problem (see [13] for a recent survey).

Our goal in this work is to address the problem of finding a compact representation of CNF formulae. Our proposed mining based compression approach aims to *discover hidden structures* from arbitrary CNF formulae and to exploit them to reduce the overall size of the CNF formula while preserving satisfiability. It is the first application of data mining techniques to Boolean Satisfiability.

Recently, a first constraint programming (CP) based data mining framework was proposed by Luc De Raedt et al. in [11, 2]. This new framework offers a declarative and flexible representation model. It allows data mining problems to benefit from several generic and efficient CP solving techniques [8]. This first study leads to the first CP approach for itemset mining displaying nice declarative opportunities while opening interesting perspectives to cross fertilization between data-mining, constraint programming and propositional satisfiability.

In this paper, we are particularly interested in the other side of this innovative connection between these two research domains, namely how data-mining can be helpful for SAT. We present the first data-mining approach for Boolean Satisfiability. We show that itemset mining techniques are very suitable for discovering interesting patterns from CNF formulae. Such patterns are then used to rewrite the CNF formula more compactly. We also show how sets of binary clauses can be also compacted by our approach. We also prove that our approach can automatically achieve similar reductions as in [12], on bi-cliques and cliques of binary clauses. It is also important to note, that our proposed mining4SAT approach is incremental. Indeed, our method can be applied incrementally or in parallel on the subsets of any partition of the original CNF formula. This will be particularly helpful for huge CNF formula that can not be entirely stored in memory.

tid	itemset
1	$Joyce, Beckett, Proust$
2	$Faulkner, Hemingway, Melville$
3	$Joyce, Proust$
4	$Hemingway, Melville$
5	$Flaubert, Zola$
6	$Hemingway, Golding$

Table 1: An example of transactions database $\mathcal{D}$

2. FREQUENT ITEMSET MINING PROBLEM

2.1 Preliminary Notations and Definitions

Let $\mathcal{I}$ be a set of *items*. A set $I \subseteq \mathcal{I}$ is called an *itemset*. A *transaction* is a couple (tid, I) where tid is the *transaction identifier* and I is an itemset. A *transactions database* $\mathcal{D}$ is a finite set of transactions over $\mathcal{I}$ where for all two different transactions, they do not have the same transaction identifier. We say that a transaction (tid, I) *supports* an itemset J if $J \subseteq I$.

The *cover* of an itemset I in a transactions database $\mathcal{D}$ is the set of identifiers of transactions in $\mathcal{D}$ supporting I: $\mathcal{C}(I, \mathcal{D}) = \{tid \mid (tid, J) \in \mathcal{D} \texttt{ and } I \subseteq J\}$. The *support* of an itemset I in $\mathcal{D}$ is defined by: $\mathcal{S}(I, \mathcal{D}) =\mid \mathcal{C}(I, \mathcal{D}) \mid$. Moreover, the *frequency* of I in $\mathcal{D}$ is defined by: $\mathcal{F}(I, \mathcal{D}) = \frac{\mathcal{S}(I,\mathcal{D})}{|\mathcal{D}|}$.

For example, let us consider the transactions database in Table 1. Each transaction corresponds to the favorite writers of a library member. For instance, we have :
$\mathcal{S}(\{Hemingway, Melville\}, \mathcal{D}) = |\{2, 4\}| = 2$ and
$\mathcal{F}(\{Hemingway, Melville\}, \mathcal{D}) = \frac{1}{3}$.
Let $\mathcal{D}$ be a transactions database over $\mathcal{I}$ and λ a minimal support threshold. The frequent itemset mining problem consists of computing the following set: $\mathcal{FIM}(\mathcal{D}, \lambda) = \{I \subseteq \mathcal{I} \mid \mathcal{S}(I, \mathcal{D}) \geqslant \lambda\}$.

The problem of computing the number of frequent itemsets is $\#P$-hard [7]. The complexity class $\#P$ corresponds to the set of counting problems associated with a decision problem in NP. For example, counting the number of models satisfying a CNF formula is a $\#P$ problem.

2.2 Maximal and Closed Frequent Itemsets

Let us now recall two popular condensed representations of the set of all frequent itemsets: maximal and closed frequent itemsets.

Definition 1 (Maximal Frequent Itemset). *Let $\mathcal{D}$ be a transactions database, λ a minimal support threshold and $I \in \mathcal{FIM}(\mathcal{D}, \lambda)$. I is called maximal when for all $I' \supset I$, $I' \notin \mathcal{FIM}(\mathcal{D}, \lambda)$ (I' is not a frequent itemset).*

We denote by $\mathcal{MAX}(\mathcal{D}, \lambda)$ the set of all maximal frequent itemsets in $\mathcal{D}$ with λ as a minimal support threshold. For instance, in the previous example, we have $\mathcal{MAX}(\mathcal{D}, 2) = \{\{Joyce, Proust\}, \{Hemingway, Melville\}\}$.

DEFINITION 2 (CLOSED FREQUENT ITEMSET). *Let $\mathcal{D}$ be a transactions database, λ a minimal support threshold and $I \in \mathcal{FIM}(\mathcal{D}, \lambda)$. I is called closed when for all $I' \supset I$, $\mathcal{C}(I, \mathcal{D}) \neq \mathcal{C}(I', \mathcal{D})$.*

We denote by $\mathcal{CLO}(\mathcal{D}, \lambda)$ the set of all closed frequent itemsets in $\mathcal{D}$ with λ as a minimal support threshold. For instance, we have $\mathcal{CLO}(\mathcal{D}, 2) = \{\{Hemingway\}, \{Joyce, Proust\}, \{Hemingway, Melville\}\}$. In particular, let us note that we have $\mathcal{C}(\{Hemingway\}, \mathcal{D}) = \{2, 4, 6\}$ and $\mathcal{C}(\{Hemingway, Melville\}, \mathcal{D}) = \{2, 4\}$. That explains why $\{Hemingway\}$ and $\{Hemingway, Melville\}$ are both closed. One can easily see that if all the closed (resp. maximal) frequent itemsets are computed, then all the frequent itemsets can be computed without using the corresponding transactions database. Indeed, the frequent itemsets correspond to all the subsets of the closed (resp. maximal) frequent itemsets.

The number of maximal (resp. closed) frequent itemsets is significantly smaller than the number of frequent itemsets. Nonetheless, this number is not always polynomial in the size of the database [16]. In particular, the problem of counting the number of maximal frequent itemsets is $\#P$-complete (see also [16]).

Many algorithm has been proposed for enumerating frequent closed itemsets. One can cite Apriori-like algorithm, originally proposed in [1] for mining frequent itemsets for association rules. It proceeds by a level-wise search of the elements of $\mathcal{FIM}(\mathcal{D}, \lambda)$. Indeed, it starts by computing the elements of $\mathcal{FIM}(\mathcal{D}, \lambda)$ of size one. Then, assuming the element of $\mathcal{FIM}(\mathcal{D}, \lambda)$ of size n is known, it computes a set of candidates of size $n + 1$ so that I is a candidate if and only if all its subsets are in $\mathcal{FIM}(\mathcal{D}, \lambda)$. This procedure is iterated until no more candidates are found. Obviously, this basic procedure is enhanced using some properties such as the anti-monotonicity property that allow us to reduce the search space. Indeed, if $I \notin \mathcal{FIM}(\mathcal{D}, \lambda)$, then $I' \notin \mathcal{FIM}(\mathcal{D}, \lambda)$ for all $I' \supseteq I$. In our experiments, we consider one of the state-of-the-art algorithms LCM for mining frequent closed itemsets proposed by Takeaki Uno et al. in [15]. In theory, the authors prove that LCM exactly enumerates the set of frequent closed itemsets within polynomial time per closed itemset in the total input size. Let us mention that LCM algorithm obtained the best implementation award of FIMI'2004 (Frequent Itemset Mining Implementations).

3. FROM CNF FORMULA TO TRANSACTION DATABASE

We first introduce the satisfiability problem and some necessary notations. We consider the conjunctive normal form (CNF) representation for the propositional formulas. A *CNF formula* Φ is a conjunction of clauses, where a *clause* is a disjunction of literals. A *literal* is a positive (p) or negated ($\neg p$) propositional variable. The two literals p and $\neg p$ are called *complementary*. A CNF formula can also be seen as a set of clauses, and a clause as a set of literals. The size of the CNF formula Φ is defined as $|\Phi| = \sum_{c \in \Phi} |c|$, where $|c|$ is equal to the number of literals in c. A *unit* clause is a clause containing only one literal (called *unit literal*), while a binary clause contains exactly two literals. A formula containing only binary clauses is called *2-CNF* formula. An *empty clause*, denoted $\bot$, is interpreted as false (unsatisfiable), whereas an *empty CNF formula*, denoted $\top$, is interpreted as true (satisfiable).

Let c_1 and c_2 be two clauses of a formula Φ. We say that c_1 *subsumes* c_2 iff $c_1 \subseteq c_2$. If c_1 subsumes c_2, then the clause c_2 can be deleted from Φ while preserving satisfiability. A CNF formula Φ is *closed under subsumption* iff $\forall c \in \Phi$, $\nexists c' \in \Phi$ such that $c \neq c'$ and c' subsumes c. We denote by Φ^s, the formula obtained from Φ by eliminating all the subsumed clauses.

We note $\bar{l}$ the *complementary* literal of l. More precisely, if $l = p$ then $\bar{l}$ is $\neg p$ and if $l = \neg p$ then $\bar{l}$ is p. Let us recall that any propositional formula can be translated to CNF using Tseitin's linear encoding [14]. We denote by $\mathcal{V}_\Phi$ the set of propositional variables appearing in Φ, while the set of literals of Φ is defined as $\mathcal{L}_\Phi$.

An *interpretation* $\mathcal{B}$ of a propositional formula Φ is a function which associates a value $\mathcal{B}(p) \in \{0, 1\}$ (0 corresponds to $false$ and 1 to $true$) to the variables $p \in \mathcal{V}_\Phi$. A *model* of a formula Φ is an interpretation $\mathcal{B}$ that satisfies the formula: $\mathcal{B}(\Phi) = 1$. The *SAT problem* consists in deciding if a given CNF formula admits a model or not.

We define $\Phi|_x$ as the formula obtained from Φ by assigning x the truth-value 1. Formally $\Phi|_x = \{c \mid c \in \Phi, \{x, \neg x\} \cap c = \emptyset\} \cup \{c \backslash \{\neg x\} \mid c \in \Phi, \neg x \in c\}$. Φ^* denotes the formula Φ *closed under unit propagation*, defined recursively as follows: (1) $\Phi^* = \Phi$ if Φ does not contain any unit clause, (2) $\Phi^* = \bot$ if Φ contains two unit-clauses $\{x\}$ and $\{\neg x\}$, (3) otherwise, $\Phi^* = (\Phi|_x)^*$ where x is the literal appearing in a unit clause of Φ.

A CNF formula can be considered as a transactions database, called CNF database, where the items correspond to literals and the transactions to clauses. Note that complementary literals correspond to two different items.

DEFINITION 3 (CNF TO $\mathcal{D}$). *Let $\Phi = \bigwedge_{1 \leqslant i \leqslant n} c_i$ be a CNF formula. The set of items $\mathcal{I} = \mathcal{L}_\Phi$ and the transactions database associated to Φ is defined as $\mathcal{D}^c_\Phi = \{(tid_i, c_i) | 1 \leqslant i \leqslant n\}$*

For instance, the CNF formula $(x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_2 \vee x_4) \wedge x_1 \wedge (x_3 \vee \neg x_4)$ corresponds to the following transactions database:

tid	itemset
1	$x_1, \neg x_2, \neg x_3$
2	$x_1, \neg x_2, x_4$
3	x_1
4	$x_3, \neg x_4$

In this context, a frequent itemset corresponds to a frequent set of literals: the number of clauses containing these literals is greater or equal to the minimal support threshold. For instance, if we set the minimal threshold λ to 2, we get $\{x_1, \neg x_2\}$ as a frequent itemset in the previous database. The set of maximal frequent itemsets is the smallest set of frequent set of literals where each frequent set of literals is included in at least one of its elements. For instance, the unique maximal frequent itemset in the previous example is $\{x_1, \neg x_2\}$ ($\lambda = 2$). Furthermore, the set of closed frequent itemsets is the smallest set of frequent set of literals where each frequent itemset is included in at least one of its elements *having the same support*. For instance, the set of the closed frequent itemsets is $\{\{x_1, \neg x_2\}, \{x_1\}\}$.

In the definition of a transaction database, we did not require that the set of items in a transaction to be unique. Indeed, two different transactions can have the same set of items and different identifiers. A CNF formula may contain the same clause more than once, but in practice this does not provide any information about satisfiability. Thus, we can consider a CNF database as just a set of itemsets (sets of literals).

4. MINING-BASED APPROACH FOR SIZE-REDUCTION OF CNF FORMULAE

In this section, we describe our mining based approach, called Mining4SAT, for reducing the size of CNF formulae. The key idea consists in searching for frequent sets of literals (sub-clauses) and substituting them with new variables using Tseitin's encoding [14].

4.1 Tseitin's Encoding

Tseitin's encoding consists in introducing fresh variables to represent sub-formulae in order to represent their truth values. For example, given a Boolean formula, containing the variables a and b, and v a fresh variable, one can add the definition $v \leftrightarrow a \vee b$ (called extension) to the formula while preserving satisfiability. Tseitin's extension principle is at the basis of the linear transformation of general Boolean formulae into CNF. Two decades later, after Tseitin's seminal paper, Plaisted and Greenbaum presented an improved CNF translation that essentially produces a subset of Tseitin's representation [10]. They noticed that by keeping track of polarities of sub-formulae, one can remove large parts of Tseitin translation. In the sequel, we use Plaisted and Greenbaum improvement. More precisely, as the disjunction $a \vee b$ is a sub-clause with positive polarity, it is sufficient to add the formula $v \rightarrow a \vee b$ i.e. a clause $(\neg v \vee a \vee b)$.

Let consider the following DNF formula (Disjunctive Normal Form: a disjunction of conjunctions):

$$(x_1 \wedge \cdots \wedge x_l) \vee (y_1 \wedge \cdots \wedge y_m) \vee (z_1 \wedge \cdots \wedge z_n)$$

A naive way of converting such a formula to a CNF formula consists in using the distributivity of disjunction over conjunction $(A \vee (B \wedge C) \leftrightarrow (A \vee B) \wedge (A \vee C))$:

$$(x_1 \vee y_1 \vee z_1) \wedge (x_1 \vee y_1 \vee z_2) \wedge \cdots \wedge (x_l \vee y_m \vee z_n)$$

Such a naive approach is clearly exponential in the worst case. In Tseitin's transformation, fresh propositional variables are introduced to prevent such combinatorial explosion, mainly caused by the distributivity of disjunction over conjunction and vice versa. With additional variables, the obtained CNF formula is linear in the size of the original formula. However the equivalence is only preserved w.r.t satisfiability:

$$(t_1 \vee t_2 \vee t_3) \wedge (t_1 \rightarrow (x_1 \wedge \cdots \wedge x_l)) \wedge (t_2 \rightarrow (y_1 \wedge \cdots \wedge y_m))$$
$$\wedge (t_3 \rightarrow (z_1 \wedge \cdots \wedge z_n))$$

4.2 A Size-Reduction Method

Let us now describe in more details, how itemset mining techniques can be combined with Tseitin's principle to compress CNF formula.

To illustrate the main ideas behind our mining based compression approach, we consider the CNF formula Φ:

$$(x_1 \vee \cdots \vee x_n \vee \alpha_1) \wedge \cdots \wedge (x_1 \vee \cdots \vee x_n \vee \alpha_k)$$

where $n \geqslant 2$, $k > \frac{n+1}{n-1}$ and $\alpha_1, \ldots, \alpha_k$ are clauses. As we can observe, the sub-clause $(x_1 \vee \ldots, \vee x_n)$ appears in each clause of Φ. Using Tseitin's encoding, we can rewrite Φ as follows:

$$(y \vee \alpha_1) \wedge \cdots \wedge (y \vee \alpha_k) \wedge (x_1 \vee \cdots \vee x_n \vee \neg y)$$

where y is a fresh propositional variable. Indeed, $n \times k$ literals are replaced with $k + n + 1$ literals leading to a gain in terms of number of literals of $(n \times k) - (n + k + 1)$.

Now, if we consider the CNF database corresponding to Φ, $\{x_1, \ldots, x_n\}$ is a frequent itemset where the minimal support threshold is greater or equal to k. It is easy to see that to reduce the number of literals n must be greater or equal to 2. Indeed, if $n < 2$ then there is no reduction of the number of literals, on the contrary, their number is increased. Regarding the value of k, one can also see that such a transformation is interesting only when $k > \frac{n+1}{n-1}$. Thus, there are three cases : if $n = 2$, then $k \geqslant 4$, else if $n = 3$ then $k \geqslant 3$, $k \geqslant 2$ otherwise. Therefore, the number of literals is always reduced when $k \geqslant 4$.

Obviously, a boolean interpretation is a model of the formula obtained after reduction if and only if it is a model of Φ.

In the previous example, we illustrate how the problem of finding frequent itemsets can be used to reduce the size of a CNF formula. One can see that, in general, it is more interesting to consider a condensed representation of the frequent itemsets (closed and maximal) to reduce the number of literals. Indeed, by using a condensed representation, we consider all the frequent itemsets and the number of fresh propositional variables and new clauses (in our example, y and $(x_1 \vee \cdots \vee x_n \vee \neg y)$) introduced is smaller than that of those introduced by using all the frequent itemsets. For instance, in the previous formula, it is not interesting to introduce a fresh propositional variable for each subset of $\{x_1, \ldots, x_n\}$.

EXAMPLE 1. *Let us consider a formula Φ containing the following 8 clauses:*

$$\begin{array}{l}
\neg x_0 \vee x_2 \\
\neg x_1 \vee x_4 \\
\neg x_3 \vee x_4 \\
\neg x_5 \vee x_6
\end{array}$$

$$\begin{array}{rcl|l}
\neg x_0 \vee x_1 & \vee & & x_4 \vee x_5 \vee x_6 \\
x_3 & \vee & & x_4 \vee x_5 \vee x_6 \\
\neg x_1 \vee x_2 & \vee & & x_4 \vee x_5 \vee x_6 \\
\neg x_2 \vee x_3 & \vee & & x_4 \vee x_5 \vee x_6
\end{array}$$

Suppose that the minimal support threshold is less than 4, then the sub-clause $(x_4 \vee x_5 \vee x_6)$ is frequent. Using our approach, the formula Φ can be rewritten as:

$$\begin{array}{l}
\neg x_0 \vee x_2 \\
\neg x_1 \vee x_4 \\
\neg x_3 \vee x_4 \\
\neg x_5 \vee x_6
\end{array}$$

$$\begin{array}{rcl|l}
\neg x_0 \vee x_1 & \vee & & \mathbf{y} \\
x_3 & \vee & & \mathbf{y} \\
\neg x_1 \vee x_2 & \vee & & \mathbf{y} \\
\neg x_2 \vee x_3 & \vee & & \mathbf{y}
\end{array}$$

$$\neg \mathbf{y} \quad \vee \quad x_4 \vee x_5 \vee x_6$$

In this simple example, the original formula contains 27 *literals, while the new formula involves only* 23 *literals.*

Closed vs. Maximal.

In Section 2.2, we introduced two condensed representations of the frequent itemsets: closed and maximal. The question is, which condensed representation is better? We know that the set of maximal frequent itemsets is included in that of the closed ones. Thus, a small number of fresh variables and new clauses are introduced using the maximal frequent itemsets. However, there are cases where the use of the closed frequent itemsets is more suitable. For example, let us consider the following formula:

$$\begin{array}{ll}(x_1 \vee \ldots \vee x_k \vee \ldots \vee x_n \vee \alpha_1) & \wedge \\ \vdots & \vdots \\ (x_1 \vee \ldots \vee x_k \vee \ldots \vee x_n \vee \alpha_m) & \wedge \\ (x_1 \vee \ldots \vee x_k \vee \beta_1) & \wedge \\ \vdots & \vdots \\ (x_1 \vee \ldots \vee x_k \vee \beta_{m'}) & \end{array}$$

where $k \geqslant 2$, $m, m' \geqslant 4$ and $n > k$. We assume that the frequent itemsets are only the subsets of $\{x_1, \ldots, x_n\}$. Therefore, $\{x_1, \ldots, x_n\}$ is the unique maximal itemset and the closed itemsets are $\{x_1, \ldots, x_n\}$ and $\{x_1, \ldots, x_k\}$. Let us start by using the closed frequent itemset $\{x_1, \ldots, x_n\}$ in the reduction of the number of literals:

$$\begin{array}{ll}(y \vee \alpha_1) & \wedge \\ \vdots & \vdots \\ (y \vee \alpha_m) & \wedge \\ (x_1 \vee \ldots \vee x_k \vee \beta_1) & \wedge \\ \vdots & \vdots \\ (x_1 \vee \ldots \vee x_k \vee \beta_{m'}) & \wedge \\ (x_1 \vee \ldots \vee x_n \vee \neg y) & \end{array}$$

Now, by using $\{x_1, \ldots, x_k\}$, we get the following formula:

$$\begin{array}{ll}(y \vee \alpha_1) & \wedge \\ \vdots & \vdots \\ (y \vee \alpha_m) & \wedge \\ (z \vee \beta_1) & \wedge \\ \vdots & \vdots \\ (z \vee \beta_{m'}) & \wedge \\ (z \vee x_{k+1} \vee \ldots \vee x_n \vee \neg y) & \wedge \\ (x_1 \vee \ldots \vee x_k \vee \neg z) & \end{array}$$

In this example, it is more interesting to consider the closed frequent itemsets in our Mining4SAT approach.

In fact, a (closed) frequent itemset I and one of its subsets I' (which can be closed) are both interesting if $\mathcal{S}(I') - \mathcal{S}(I) > \frac{|I'|+1}{|I'|-1} - 1$. Indeed, if we apply our transformation using I, then the support of I' in the resulting formula is equal to $\mathcal{S}(I') - \mathcal{S}(I) + 1$, and we know that I' is interesting in the resulting formula if its support is greater to $\frac{|I'|+1}{|I'|-1}$.

Overlap.

Let Φ be a set of itemsets. Two itemsets I and I' of Φ overlap if $I \cap I' \neq \emptyset$. Moreover, I and I' are in the same *overlap class* if there exist k itemsets $I_1, \ldots, I_k$ of Φ such that $I = I_1, I_k = I'$ and for all $1 \leqslant i \leqslant k-1$, I_i and I_{i+1} overlap.

In our transformation, one can have some problems when two frequent itemsets overlap. For example, if $\{x_1, x_2, x_3\}$ and $\{x_2, x_3, x_4\}$ are two frequent itemsets (3 is the minimal support threshold) such that $\mathcal{S}(\{x_1, x_2, x_3\}) = 3$, $\mathcal{S}(\{x_2, x_3, x_4\}) = 3$ and $\mathcal{S}(\{x_1, x_2, x_3, x_4\}) = 2$, then if we apply our transformation using $\{x_1, x_2, x_3\}$, then the support of $\{x_2, x_3, x_4\}$ is equal to 2 (infrequent) in the resulting formula and vice versa. Thus, we can not use both of them in the transformation.

Le us note that the overlap notion can be seen as a generalization of the subset one. Let I and I' be frequent itemsets such that they overlap. They are both interesting in our transformation if:

1. $\mathcal{S}(I) - \mathcal{S}(I \cup I') > \frac{|I|+1}{|I|-1} - 1$ or $\mathcal{S}(I') - \mathcal{S}(I \cup I') > \frac{|I'|+1}{|I'|-1} - 1$. This comes from the fact that if we apply the transformation using I (resp. I'), then the support of I' (resp. I) is equal to $\mathcal{S}(I') - \mathcal{S}(I \cup I') + 1$ (resp. $\mathcal{S}(I) - \mathcal{S}(I \cup I') + 1$).

2. $|I \backslash I'| \geqslant k$ (resp. $|I' \backslash I| \geqslant k$) where $k = 2$ if $\mathcal{S}(I) \geqslant 4$ (resp. $\mathcal{S}(I') \geqslant 4$), $k = 3$ if $\mathcal{S}(I) = 3$ (resp. $\mathcal{S}(I') = 3$), $k = 4$ otherwise. Indeed, in the previous cases, $I \backslash I'$ (resp. $I' \backslash I$) can be used in our transformation.

Mining4SAT algorithm.

We now describe our compression algorithm, called Mining4SAT, using the set of closed frequent itemsets. Let us note that the optimal transformation using the set of all the closed frequent itemsets can be obtained by an optimal transformation using separately the overlap classes of this set. Actually, since any two distinct overlap classes do not share any literal, the reduction applied to a given formula using the elements of an overlap class does not affect the supports of the elements of the other classes. Moreover, one can easily compute the set of all the overlap classes of the set of the closed frequent itemsets: let $G = (V, E)$ be an undirected graph such that V is the set of the closed frequent itemsets and (I_1, I_2) is an edge of G if and only if I_1 and I_2 overlap; C is an overlap class if and only if it corresponds to the set of vertices of a connected component of G which is not included in any other connected component of G. For this reason, we restrict here our attention to the reductions that can be obtained using a single overlap class. The whole size reduction process can be performed by iterating on all the overlap classes.

Let I be a closed frequent itemset, we denote by $\alpha(I)$ the value $\mathcal{S}(I) \times (|I|-1) - |I| - 1$ that corresponds to the number of literals reduced by applying our transformation with I on a CNF formula.

Algorithm 1 takes as input a CNF formula ϕ and an overlap class C, and returns ϕ after applying size-reduction transformations. It iterates until there is no element in C. In each iteration, it first selects one of the most interesting elements in C (line 2): an element I of C such that there is no element $I' \in C$ satisfying $\alpha(I') > \alpha(I)$. Note that this element is not necessarily unique in C. This instruction means that Algorithm 1 is a greedy algorithm because it makes a locally optimal choice at each iteration. Then, it applies

Algorithm 1 Size Reduction

Require: A formula ϕ, an overlap class of closed frequent itemsets C

1: **while** $C \neq \emptyset$ **do**
2: $I \leftarrow MostInterstingElment(C)$;
3: $replace(\phi, I, x)$;
4: $Add(\phi, I, x)$:
5: $remove(C, I)$;
6: $replaceSubset(C, I, x)$;
7: $removeUninterestingElements(C)$;
8: $updateSupports(C)$;
9: **end while**
10: **return** ϕ

our transformation using $I = \{y_1, \ldots, y_n\}$: it replaces the occurrences of I with a fresh propositional variable x (line 3); and it adds the clause $y_1 \vee \ldots \vee y_n \vee \neg x$ to ϕ (line 4). It next removes I from C (line 5) and replaces I in the the other elements of C with x (line 6). The next instruction (line 7) consists in removing the elements of C that could increase the number of literals: the elements that overlap with I and are not included in I. As explained before, an element of C overlapping with I does not necessarily increase the number of literals. Thus, by removing elements from C because only they overlap with I, our algorithm can remove closed frequent itemsets decreasing the number of literals. A partial solution to this problem consists in recomputing the closed frequent itemsets in the formula returned by Algorithm 1. The last instruction in the while loop (line 8) consists in updating the supports of the elements remaining in C following the new value of ϕ: a support of an element I' remaining in C changes only when it is included in I and its new support is equal to $\mathcal{S}(I') - \mathcal{S}(I) + 1$. This instruction also removes all the elements of C becoming uninteresting because of the new supports and sizes.

5. APPLICATION: A COMPACT REPRESENTATION OF SETS OF BINARY CLAUSES

Binary clauses (2-CNF formula) are ubiquitous in CNF formula encoding real-world problems. Some of them contain more than 90% of binary clauses. One of the main reason is that the encoding of several kinds of constraints to CNF leads to big sets of binary clauses. As an example, expressing that the variables $x_1, \ldots, x_n$ must take different values in $\{v_1, \ldots, v_m\}$ can leads to $n^2 \times m^2$ binary clauses (cliques of binary clauses) with a naive encoding. For $n = 100$ and $m = 10$, we get about one million of binary clauses or 10 Megabytes if each binary clause takes 10 bytes. Another example given by Rintanen in [12], concerns the encoding of planning problem to SAT and particularly of some invariants such as the one expressing that an object cannot be in two locations at the same time. For n state variables there are $\frac{n \times (n-1)}{2}$ invariants that are binary clauses. In the case of a planning problem with $n = 5000$ state variables and a formula that encodes the search for plans of 100 time points, if only one of the state variables is true at any given time, the total size of the set of binary clauses is about 12 Gigabytes.

Table 2 illustrates the proportion of binary clauses on a sample of SAT instances (application category) taken from the last SAT competitions [17]. For each instance (first column), we give the total number of clauses (#cls), the number of binary clauses (#bin) and the ratio of binary clauses ((%)bin).

instance	#cls	#bin	(%) bin
velev-pipe-o-uns-1.1-6	304026	268354	88,26 %
9dlx_vliw_at_b_iq2	542253	500227	92,24 %
1dlx_c_iq57_a	8562505	7567948	88,38 %
7pipe_k	751116	722278	96,16 %
SAT_dat.k100.debugged	670701	523153	78,00 %
BM_FV_2004_rule_batch	445444	339588	76,23 %
sokoban-sequential-p145-*.040-*	1413816	1364160	96,48 %
openstacks-*-p30_1.085-*	1621926	1601145	98,71 %
aaai10-planning-ipc5-*-12-step16	1029036	991140	96,31 %
k2fix_gr_rcs_w8.shuffled	271393	270136	99,53 %
homer17.shuffled	1742	1716	98,50 %
gripper13u.shuffled-as.sat03-395	38965	35984	92,34 %
grid-strips-grid-y-3.045-*	2750755	2695230	97,98 %

Table 2: Ratio of binary clauses in some SAT instances

In our Mining4SAT general approach presented previously, binary clauses are not taken into account. Indeed, to reduce the size of a formula, we only search for itemsets of size at least two. The case where a binary clause representing a closed frequent itemset can be considered by our Mining4SAT algorithm is when it appears at least four times in a formula Φ. For example, when Φ contains the following clauses $c_1 = (a \vee b)$, $c_2 = (a \vee b \vee \alpha_1)$, $c_3 = (a \vee b \vee \alpha_2)$, and $c_4 = (a \vee b \vee \alpha_3)$, where α_i for $1 \leqslant i \leqslant 3$ are clauses. In this case, the last three clauses are subsumed by the first binary clause. The clauses c_2, c_3 and c_4 are redundant and can be eliminated from Φ. Consequently, if we suppose that a formula is closed under subsumption, this case do not happen. Let us give a more general formulation of this particular case. Let Φ be the following formula:

$$\begin{array}{ll} (x_1 \vee \cdots \vee x_n) & \wedge \\ (x_1 \vee \cdots \vee x_n \vee \alpha_1) & \wedge \\ \vdots & \vdots \\ (x_1 \vee \cdots \vee x_n \vee \alpha_k) & \end{array}$$

where $\alpha_1, \ldots, \alpha_k$ are clauses. As we can see, the first clause subsumes all the remaining clauses. Then we obtain the formula closed under subsumption $\Phi^s = (x_1 \vee \cdots \vee x_n)$. Suppose that $I = \{x_1, \cdots, x_n\}$ is a frequent closed itemset of $\mathcal{D}^c_\Phi$. Using I, we obtain the following formula Φ':

$$(y) \wedge (y \vee \alpha_1) \wedge \cdots \wedge (y \vee \alpha_k) \wedge (\neg y \vee x_1 \vee \cdots \vee x_n)$$

Applying unit propagation, we obtain $\Phi'^* = (x_1 \vee \cdots \vee x_n)$. As we can remark $\Phi^s = \Phi'^*$. To summarize, our general Mining4SAT can derive unit literals (y). Then by applying unit propagation closure, some redundant clauses are eliminated automatically.

5.1 Compacting Arbitrary Sets of Binary Clauses

In this section, we first show how our mining based approach can be used to achieve a compact representation of arbitrary set of binary clauses. Then, we consider two interesting special cases corresponding to sets of binary clauses representing either a clique or a bi-clique. It is important to note that, in [12], the authors investigated only these particular cases.

In order to make strong reductions in terms of literals but also in terms of clauses, we propose a four step approach for the compression of a set of binary clauses. In the first step, we rewrite the set of binary clauses using another more suitable representation. Secondly, from this intermediary representation, we derive a new transactions database. Then we search for frequent closed itemsets. Finally, we apply the compression algorithm obtained by a slight modification to the compression Algorithm 1 on the set of binary clauses .

Let us first introduce a more convenient and equivalent representation of a set binary of clauses.

DEFINITION 4 (B-IMPLICATION). *A B-implication is a Boolean formula of the following from :* $x \vee \beta(x)$ *where* $\beta(x)$ *is a conjunction of literals.*

Let S be the following set of binary clauses: $\{(x \vee y_1), \ldots, (x \vee y_k)\}$, with $k \geqslant 1$. Note that the B-implication $B(S) = x \vee (y_1 \wedge \cdots \wedge y_k)$ is equivalent to the conjunction of the elements of S. Hence, each 2-CNF formula Φ can be transformed into a set of B-implications, noted $B_{[\vee(\wedge)]}(\Phi)$. The original formula Φ can be obtained from $B_{[\vee(\wedge)]}(\Phi)$ by distributing $\vee$ over $\wedge$. However, there exist several ways for rewriting a 2-CNF as a conjunction of B-implications. A naive way is to simply fix a complete order relation over $\mathcal{L}_\Phi$.

We define a complete order relation over $\mathcal{L}_\Phi$. Let f be a bijective mapping from $\mathcal{L}_\Phi$ to $\{1 \ldots |\mathcal{L}_\Phi|\}$. A literal x is smaller than a literal y, noted $x \preceq y$, iff $f(x) \leqslant f(y)$. In this way, each literal of Φ is mapped to a unique natural number. $\preceq$ is a complete order. Now, using this order, we get a unique way to rewrite a 2-CNF formula as a set of B-implications. Let Φ be a 2-CNF formula and $(x \vee y) \in \Phi$, we conjunctively add y (respectively x) to $\beta(x)$ (respectively $\beta(y)$), if $x \prec y$ (respectively $y \prec x$).

Algorithm 2 aims to compute the set of B-implications associated to a 2-CNF formula Φ. It takes a 2-CNF formula Φ and a complete order $\preceq$ over $\mathcal{L}(\Phi)$ as inputs, and provides a set of B-implications $B_{[\vee(\wedge)]}(\Phi)$ as output. In line 1, we initialize $\mathcal{C}(x)$ to the empty set for all the literals of Φ. Following our order relation, in lines $2-9$, we build the set $\beta(x)$ for each literal $x \in \mathcal{L}_\Phi$. Indeed, for each binary clause $(x \vee y) \in \Phi$, y is added to $\mathcal{C}(x)$ (respectively x is added to $\mathcal{C}(y)$) if $x \prec y$ (respectively $y \prec x$).

In lines $10-14$, if according to the chosen ordering, we have $x \prec y$ and $\beta(x)$ contains only one literal, then we try to enhance the compression of the set of B-implications $B_{[\vee(\wedge)]}(\Phi)$. In this case, we add x to the set $\mathcal{C}(y)$, only if it contains at least one literal, and we set $\mathcal{C}(x)$ to the emptyset. In line 16, we return only the B-implications of the form $[x \vee \beta(x)]$ only when $\mathcal{C}(x)$ is not empty.

Assuming that adding an element to a set can be performed in constant time, the worst case complexity of the Algorithm 2 is in $\mathcal{O}(|\Phi| + |\mathcal{L}(\Phi)|)$.

Now, we explain how a transactions database is associated to a 2-CNF formula, using a set of B-implications which we consider as an intermediate representation.

DEFINITION 5 (2-CNF TO $\mathcal{D}$). *Let* Φ *be a 2-CNF formula. The transactions database associated to* Φ *is defined as* $\mathcal{D}^b_\Phi = \{(tid_{x_i}, \beta_i) | x_i \vee \beta_i \in B_{[\vee(\wedge)]}(\Phi)\}$.

Mining4Binary algorithm.

Let us describe our approach to compact a 2-CNF formula Φ, called Mining4Binary (for reducing the size of a set of binary clauses). First, after rewriting Φ as $B_{[\vee(\wedge)]}(\Phi)$, we build the transactions database $\mathcal{D}^b_\Phi$. Then the set of closed frequent itemsets and its associated overlap classes $\mathcal{C}$ are computed. The last step aims to reduce the size of the 2-CNF Φ using a slightly modified version of the Algorithm 1. We only need to add two modifications. First, our 2-CNF compression algorithm takes as input $B_{[\vee(\wedge)]}(\Phi)$ and returns a compressed set of B-implications. Secondly, for an itemset $I = \{y_1, \ldots, y_n\}$, in line 4 of the Algorithm 1, we introduce a fresh variable x and we add a B-implication $[\neg x \vee (y_1 \wedge y_2 \wedge \cdots \wedge y_n)]$ to $B_{[\vee(\wedge)]}(\Phi)$. This modified algorithm returns a set of compressed B-implications. The last step is a trivial translation of the obtained B-implications to 2-CNF.

Obviously, the compression rate depends on the chosen ordering. Indeed, the intermediate representation (B-implications) is build according to a total order, and the transactions database depends on this intermediary representation.

Algorithm 2 B-implications

Require: A formula Φ, a complete order $\preceq$ over $\mathcal{L}(\Phi)$
1: $\mathcal{C}(x) = \emptyset,\ \forall x \in \mathcal{L}(\Phi)$
2: **for** $c = (x \vee y) \in \Phi$ **do**
3: **if** $x \prec y$ **then**
4: $\mathcal{C}(x) \leftarrow \mathcal{C}(x) \cup \{y\}$
5: **else**
6: $\mathcal{C}(y) \leftarrow \mathcal{C}(y) \cup \{x\}$
7: **end if**
8: **end for**
9: **for** $x \in \mathcal{L}(\Phi)$ **do**
10: **if** $\mathcal{C}(x) = \{y\}$ *and* $|\mathcal{C}(y)| > 1$ **then**
11: $\mathcal{C}(y) \leftarrow \mathcal{C}(y) \cup \{x\}$
12: $\mathcal{C}(x) \leftarrow \emptyset$
13: **end if**
14: **end for**
15: **return** $\{[x \vee (\bigwedge_{y \in \mathcal{C}(x)} y)] | x \in \mathcal{L}(\Phi) \mid \mathcal{C}(x) \neq \emptyset\}$

EXAMPLE 2. *Let us consider the following 2-CNF* Φ*:*

$$\begin{aligned}\Phi = \ & (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (x_1 \vee x_4) \wedge (x_1 \vee x_5) \wedge \\ & (x_1 \vee x_6) \wedge (x_1 \vee x_7) \wedge (x_2 \vee x_3) \wedge (x_2 \vee x_4) \wedge \\ & (x_2 \vee x_5) \wedge (x_2 \vee x_6) \wedge (x_2 \vee x_7) \wedge (x_3 \vee x_4) \wedge \\ & (x_3 \vee x_6) \wedge (x_3 \vee x_7) \wedge (x_3 \vee x_5) \wedge (x_4 \vee x_5) \wedge \\ & (x_4 \vee x_6) \wedge (x_4 \vee x_7) \wedge (x_5 \vee x_6) \wedge (x_5 \vee x_7) \wedge \\ & (x_6 \vee x_7)\end{aligned}$$

Using the complete order relation $x_1 \prec \ldots \prec x_7$ *over* $\mathcal{L}_\Phi$*, we can rewrite* Φ *as the following set of B-implications* $B^1_{[\vee(\wedge)]}(\Phi)$*:*

$$\begin{aligned}B^1_{\vee[\wedge]}(\Phi) = \ & \{[x_1 \vee (x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6 \wedge x_7)], \\ & [x_2 \vee (x_3 \wedge x_4 \wedge x_5 \wedge x_6 \wedge x_7)], \\ & [x_3 \vee (x_4 \wedge x_5 \wedge x_6 \wedge x_7)], \\ & [x_5 \vee (x_6 \wedge x_7)], \\ & [x_6 \vee (x_7)]\}\end{aligned}$$

The transactions database representation $\mathcal{D}^b_\Phi$ *is built from* $B^1_{[\vee(\wedge)]}(\Phi)$ *by considering only* $\beta(x_1), \ldots, \beta(x_6)$*:*

tid	$itemset$					
tid_{x_1}	x_2	x_3	x_4	x_5	x_6	x_7
tid_{x_2}		x_3	x_4	x_5	x_6	x_7
tid_{x_3}			x_4	x_5	x_6	x_7
tid_{x_4}				x_5	x_6	x_7
tid_{x_5}					x_6	x_7
tid_{x_6}						x_7

The itemset mining process is done on the conjunctive part of $B^1_{\vee[\wedge]}(\Phi)$ represented in the transactions database. Setting the minimum support threshold to 4, we get as a frequent itemset $\{x_5, x_6, x_7\}$. Using 2-CNF compression algorithm described above, we can rewrite $B^1_{[\vee(\wedge)]}(\Phi)$ as $B^2_{[\vee(\wedge)]}(\Phi)$:

$$\begin{aligned} B^2_{\vee[\wedge]}(\Phi) = \quad & \{[x_1 \vee (x_2 \wedge x_3 \wedge y)] \ , \\ & [x_2 \vee (x_3 \wedge x_4 \wedge y)] \ , \\ & [x_3 \vee (x_4 \wedge y)] \ , \\ & [x_5 \vee (x_6 \wedge x_7)] \ , \\ & [x_6 \vee (x_7)] \ , \\ & [\neg y \vee (x_5 \wedge x_6 \wedge x_7)]\} \end{aligned}$$

Finally a simple encoding of $B^2_{[\vee(\wedge)]}(\Phi)$ as CNF formula leads to following compressed 2-CNF formula:

$$\begin{aligned} \Phi = \quad & (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (x_1 \vee x_4) \wedge (x_2 \vee x_3) \quad \wedge \\ & (x_2 \vee x_4) \wedge (x_3 \vee x_4) \wedge (x_5 \vee x_6) \wedge (x_5 \vee x_7) \quad \wedge \\ & (x_6 \vee x_7) \quad \wedge \\ & (x_1 \vee y) \wedge (x_2 \vee y) \wedge (x_3 \vee y) \wedge (x_4 \vee y) \quad \wedge \\ & (x_5 \vee \neg y) \wedge (x_6 \vee \neg y) \wedge (x_7 \vee \neg y) \end{aligned}$$

The substitution of the itemset $\{x_5, x_6, x_7\}$ allows us to reduce the size of the 2-CNF formula Φ. Indeed, the resulting 2-CNF contains 5 binary clauses less.

5.2 Special Case of (Bi-)cliques of Binary Clauses

In [12], J. Rintanen addressed the problem of representing big sets of binary clauses compactly. He particularly shows that constraint graphs arising from practically interesting applications (eg. AI planning) contain big cliques or bi-cliques of binary clauses. An identified bi-clique involving the two sets of literals $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ and $\mathcal{Y} = \{y_1, y_2, \ldots, y_m\}$ expresses the propositional formula $\Phi = (x_1 \wedge x_2 \wedge \cdots \wedge x_n) \vee (y_1 \wedge y_2 \wedge \cdots \wedge y_m)$, while a clique involving the literals $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ expresses that at-most one literal from $\mathcal{X}$ is $false$.

Bi-cliques of Binary Clauses.

Let us explain how a bi-clique can be compacted with Mining4Binary method. Let $\Phi = [(x_1 \vee y_1) \wedge (x_1 \vee y_2) \vee \cdots \vee (x_1 \vee y_m)] \ldots [(x_n \vee y_1) \wedge (x_n \vee y_2) \vee \cdots \vee (x_n \vee y_m)]$ a bi-clique of $n \times m$ binary clauses (see Figure 1). Considering the complete order relation defined by: $f(x_i) = i, f(y_j) = n + j$. Using this order relation $B_{[\vee(\wedge)]}(\Phi)$ corresponds exactly to $\{(x_i \vee [y_1 \wedge y_2 \wedge \cdots \wedge y_m]) | 1 \leqslant i \leqslant n\}$. Obviously, the transactions database $\mathcal{D}^b_\Phi$ contains a single closed frequent itemset $\{y_1, y_2, \ldots, y_m\}$. Applying our algorithm leads to the following compact representation of $\Phi' = [\bigwedge_{1 \leqslant i \leqslant n}(x_i \vee z)] \wedge [\bigwedge_{1 \leqslant j \leqslant m}(\neg z \vee y_j)]$. We obtain exactly the same gain as in [12] ($\mathcal{O}(n+m)$ binary clauses and one additional variable).

Cliques of Binary Clauses.

Let $\Phi = \bigwedge_{1 \leqslant i \leqslant n-1}[(x_i \vee x_{i+1}) \wedge \cdots \wedge (x_i \vee x_n)]$ be a clique of n^2 binary clauses (see Figure 2). The set $B_{[\vee(\wedge)]}(\Phi) = \{[x_i \vee (x_{i+1} \wedge \cdots \wedge x_n)] | 1 \leqslant i \leqslant n-1\}$ using the order relation defined by: $f(x_i) = i$. If we take a closer look to D^b_Ψ, the closed frequent itemset I with the greatest value $\alpha(I)$ corresponds to $\{x_{n/2}, \ldots, x_n\}$. In the first $\frac{n}{2}$ rows of D^b_Φ, I is substituted by a fresh variable x and a new set of binary clauses $[x \vee (x_{\frac{n}{2}} \wedge, \cdots \wedge x_n)]$) is added to it, leading to two subproblems of size $\frac{n}{2} + 1$. Obviously, the same treatment is done on the set $B_{[\vee(\wedge)]}(\Phi)$. Consequently, the number of variables is defined by the following recurrence equation:

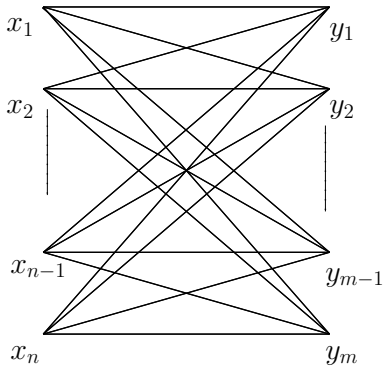


Figure 1: Bi-clique representation of the $n \times m$ clauses

$$\mathcal{V}(n) = 2 \times \mathcal{V}(\frac{n}{2} + 1) + 1 \tag{1}$$

$$\mathcal{V}(6) = 1. \tag{2}$$

The basic case is reached for $n = 6$, where the last fresh variable is introduced to represent the conjunction $x_4 \wedge x_5 \wedge x_6$. For $n < 6$ no fresh variable is introduced because no frequent closed itemset can lead to a reduction of the size of the formula. Consequently, from the solution of the previous recurrence equation, we obtain that our encoding is in $\mathcal{O}(n)$ auxiliary variables. Using the same reasoning, we also obtain the same complexity $\mathcal{O}(n)$ for the number of binary clauses. This corresponds to the complexity obtained in [12].

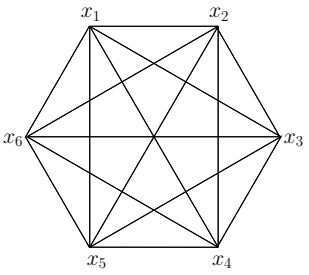


Figure 2: Clique representation of n^2 clauses

The two special cases of cliques and bi-cliques of binary clauses considered in this section, allow us to show that when a constraint is not well encoded, our approach can be used to correct and to derive a more efficient and compact encodings automatically.

6. EXPERIMENTS

In this section, we present an experimental evaluation of our proposed approaches. Two kind of experiments have been conducted. The first one deals with size reduction of arbitrary CNF formulas using Mining4SAT algorithm, while the second one attempts to reduce the size of the 2-CNF sub-formulas only, using Mining4Binary algorithm.

Both algorithms are tested on different benchmarks taken from the last SAT challenge 2012. From the 600 instances

Instance	orig.	comp.	% red
1dlx_c_iq57_a	190 Mb	164 Mb	13.68 %
6pipe_6_ooo.*-as.sat03-413	11 Mb	7.7 Mb	30.00 %
9dlx_vliw_at_b_iq6.*-*04-347	76 Mb	65 Mb	14.47 %
abb313GPIA-9-c.*.sat04-317	21 Mb	6.9 Mb	67.14 %
E05F18	3.7 Mb	2.2 Mb	40.54 %
eq.atree.braun.11.unsat	120 Kb	72 Kb	40.00 %
eq.atree.braun.12.unsat	144 Kb	88 Kb	38.88 %
k2mul.miter.*-as.sat03-355	1.5 Mb	1.3 Mb	13.33 %
korf-15	1.2 Mb	752 Kb	37.33 %
rbcl_xits_08_UNSAT	1.1 Mb	856 Kb	22.18 %
SAT_dat.k45	3.5 Mb	2.6 Mb	25.71 %
traffic_b_unsat	18 Mb	12 Mb	33.33 %
x1mul.miter.*-as.sat03-359	1.1 Mb	928 Kb	15.63 %
9dlx_vliw_at_b_iq3	19 Mb	15 Mb	21.05 %
9dlx_vliw_at_b_iq4	31 Mb	26 Mb	16.12 %
AProVE07-09	2.8 Mb	2.7 Mb	3.57 %
eq.atree.braun.10.unsat	96 Kb	56 Kb	41.66 %
goldb-heqc-frg1mul	348 Kb	328 Kb	5.74 %
minand128	7.7 Mb	2.6 Mb	66.23 %
ndhf_xits_09_UNSAT	2.6 Mb	2.1 Mb	19.23 %
velev-pipe-o-uns-1.1-6	5.5 Mb	4.4 Mb	20.00 %

Table 3: Results of Mining4SAT : a general approach

of the application category submitted to this challenge, we selected 100 instances while taking at least one instance from each family. All tests were made on a Xeon 3.2GHz (2 GB RAM) cluster and the timeout was set to 4 hours.

In Table 3 and Table 4, we indicate the size in Kilobytes (`Kb`) or Megabytes (`Mb`) of each SAT instance before (`orig.`) and after reduction (`comp.`). We also provide *%red*, the reduction percentage.

Table 3 highlights the results obtained by Mining4SAT general approach. In this experiments, and to allow possible reductions, we only search for frequent closed itemsets of size greater or equal to 4. Consequently, binary clauses are not considered. As we can observe, our Mining4SAT reduction approach allows us to reduce the size more than 20% on the majority of instances. Let us also note that the maximum (67.14 %) is reached in the case of the instance *abb313GPIA-9-c.*.sat04-317*: its original size is `21 Mb` and its size after reduction is `6.9 Mb`.

In Table 4, we present a sample of the results obtained by Mining4Binary algorithm on compacting only binary clauses. We observe similar behavior as in the first experiment in terms of size reduction.

To study the influence of our size reduction approaches on the solving time, we also run the SAT solver MiniSAT 2.2 on both the original instance and on those obtained after reduction. As a summary, our approach achieve significant reductions in the size of instances without loosing solving effectiveness.

In our experiments, we have presented the two compression algorithms, Mining4SAT (for clauses of arbitrary size) and Mining4Binary (for 2-CNF formulae). As the two algorithms can be applied independently, we have not presented the results using a two steps compression algorithm: application of the general Mining4SAT algorithm in the first step on the original formula, followed by the specialized Mining4Binary algorithm on the compressed formula derived in the first step. The results can be derived by cumulating the reductions obtained in both steps. For example, if we take the SAT instance `1dlx_c_iq57_a` presented in both Table 3 and Table 4, the reduction by the two steps algorithm is

Instance	orig.	comp.	% red
velev-pipe-o-uns-1.1-6	5.5 Mb	3.2 Mb	41.81 %
9dlx_vliw_at_b_iq2	11 Mb	6 Mb	44.45 %
1dlx_c_iq57_a	190 Mb	124 Mb	34.73 %
7pipe_k	14 Mb	5.4 Mb	61.42 %
SAT_dat.k100.debugged	16 Mb	13 Mb	18.75 %
IBM_FV_2004_rule_batch _2_31_1_SAT_dat.k80.debugged	9.7 Mb	7.5 Mb	22.68 %
sokoban-sequential-p145-*.040-*	24 Mb	14 Mb	41.66 %
openstacks-*-p30_1.085-*	30 Mb	26 Mb	13.33 %
aaai10-planning-ipc5-*-12-step16	17 Mb	12 Mb	29.41 %
k2fix_gr_rcs_w8.shuffled	3.4 Mb	1.7 Mb	50.00 %
homer17.shuffled	20 Kb	16 Kb	20.00 %
gripper13u.shuffled-as.sat03-395	524 Kb	364 Kb	30.35 %
grid-strips-grid-y-3.045-*	52 Mb	42 Mb	19.23 %

Table 4: Results of Mining4Binary: a 2-CNF approach

computed as follows : $(190 - 164) + (190 - 124) = 92Mb$. The reduction ratio is equal to 48.42%.

7. AKNOWLEDGMENTS

We thank the reviewers for their helpful comments. This work has been supported in part by the French ANR project *"DAG: Declarative Approaches for Enumerating Interesting Patterns"* under the Défis program 2009.

8. CONCLUSION AND FUTURE WORKS

In this paper, we propose the first data-mining approach, called Mining4SAT, for reducing the size of Boolean formulae in conjunctive normal form (CNF). It can be seen as a preprocessing step that aims to discover hidden structural knowledge that are used to decrease the number of literals and clauses. Mining4SAT combines both frequent itemset mining techniques for discovering interesting substructures, and Tseitin-based approach for a compact representation of CNF formulae using these substructures. Thus, we show in this work, inter alia, that frequent itemset mining techniques are very suitable for discovering interesting patterns in CNF formulae.

Since we use a greedy algorithm in our approach, the formula obtained after transformation is not guaranteed to be optimal w.r.t. size. An important open question, which we will study in future works, is how to optimally use the closed frequent itemsets ranging in an overlap class. For the special case of sets of binary clauses, finding a better complete ordering leading to an optimal compression is clearly an important challenge.

Finally, our framework can be extended to constraint satisfaction problems (CSP), Pseudo Boolean etc.

9. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD International Conference on Management of Data*, pages 207–216, Baltimore, 1993. ACM Press.

[2] E. Coquery, S. Jabbour, L. Saïs, and Y. Salhi. A sat-based approach for discovering frequent, closed and maximal patterns in a sequence. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*, pages 258–263, 2012.

[3] H. E. Dixon, M. L. Ginsberg, D. K. Hofer, E. M. Luks, and A. J. Parkes. Implementing a generalized version of resolution. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI 2004)*, pages 55–60, 2004.

[4] N. Eén and N. Sörensson. An extensible sat-solver. In *Proceedings of the Sixth International Conference on Theory and Applications of Satisfiability Testing (SAT'03)*, pages 502–518, 2003.

[5] M. L. Ginsberg and A. J. Parkes. Search, subsearch and qprop. In *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR 2000)*, 2000.

[6] É. Grégoire, R. Ostrowski, B. Mazure, and L. Saïs. Automatic extraction of functional dependencies. In *Proceedings of the Seventh International Conference on Theory and Applications of Satisfiability Testing (SAT'04)*, pages 122–132, 2004.

[7] D. Gunopulos, R. Khardon, H. Mannila, S. Saluja, H. Toivonen, and R. S. Sharma. Discovering all most specific sentences. *ACM Trans. Database Syst.*, 28(2):140–174, June 2003.

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

[9] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient sat solver. In *DAC*, pages 530–535, 2001.

[10] D. A. Plaisted and S. Greenbaum. A structure-preserving clause form translation. *Journal of Symbolic Computation*, 2(3):293–304, 1986.

[11] L. D. Raedt, T. Guns, and S. Nijssen. Constraint programming for itemset mining. In *ACM SIGKDD*, pages 204–212, 2008.

[12] J. Rintanen. Compact representation of sets of binary constraints. In *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006)*, pages 143–147. IOS Press, 2006.

[13] A. Tiwari, R. Gupta, and D. Agrawal. A survey on frequent pattern mining: Current status and challenging issues. *Inform. Technol. J*, 9:1278–1293, 2010.

[14] G. Tseitin. On the complexity of derivations in the propositional calculus. In H. Slesenko, editor, *Structures in Constructives Mathematics and Mathematical Logic, Part II*, pages 115–125, 1968.

[15] T. Uno, M. Kiyomi, and H. Arimura. Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In R. J. B. Jr., B. Goethals, and M. J. Zaki, editors, *FIMI*, volume 126 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004.

[16] G. Yang. The complexity of mining maximal frequent itemsets and maximal frequent patterns. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 344–353, New York, NY, USA, 2004.

[17] International sat competition. http://www.satcompetition.org. Organized in conjunction with the International Conference on Theory and Applications of Satisfiability Testing.

Boolean Satisfiability for Sequence Mining

Said Jabbour
jabbour@cril.fr

Lakhdar Sais
sais@cril.fr

Yakoub Salhi
salhi@cril.fr

CRIL - CNRS, University of Artois
F-62307 Lens Cedex
France

ABSTRACT

In this paper, we propose a SAT-based encoding for the problem of discovering frequent, closed and maximal patterns in a sequence of items and a sequence of itemsets. Our encoding can be seen as an improvement of the approach proposed in [8] for the sequences of items. In this case, we show experimentally on real world data that our encoding is significantly better. Then we introduce a new extension of the problem to enumerate patterns in a sequence of itemsets. Thanks to the flexibility and to the declarative aspects of our SAT-based approach, an encoding for the sequences of itemsets is obtained by a very slight modification of that for the sequences of items.

Categories and Subject Descriptors

F.4.1 [**Mathematical logic and formal languages**]: Mathematical Logic—*Logic and constraint programming*; H.2.8 [**Database management**]: Database applications—*Data mining*

Keywords

Data mining; Propositional satisfiability and modeling

1. INTRODUCTION

Frequent sequence data mining is the problem of discovering frequent patterns shared across time among an input data-sequence. Sequence mining is a central task in computational biology, temporal sequence analysis and text mining.

In this paper, we consider the pattern discovery problem for a specific class of patterns with wildcards in a sequence. The data-sequence can be seen as a sequence of items, while the pattern can be seen as a subsequence that might contains wildcards or jokers in the sense that they match any item [18, 20, 2]. At the first sight, allowing wildcards to occur in a pattern can be seen as an even more restrictive type of patterns in general. However as argued in [18] "*studying patterns with wildcards has the merit of capturing one important aspect of biological features that often concerns isolated positions inside a motif that are not part of the biological feature being captured*". The enumeration problem for maximal and closed motifs with wildcards has been investigated recently by several authors [19, 20, 2, 8]. One of the major problem is that the number of motifs can be of exponential size. This combinatorial explosion is tackled using different approaches. For example, in Parida et al. [18], the number of patterns is reduced by introducing the maximal non redundant q-patterns (patterns occurring at least q times in a sequence). Arimura and Uno [2] proposed a polynomial space and polynomial delay algorithm MaxMotif for maximal pattern discovery of the class of motifs with wildcards.

In this work, we follow the constraint programming (CP) based data mining framework proposed recently by Luc De Raedt et al. in [10] for itemset mining. This new framework offers a declarative and flexible representation model. New constraints often require new implementations in specialized approaches, while they can be easily integrated in such a CP framework. It allows data mining problems to benefit from several generic and efficient CP solving techniques. The authors show how some typical constraints (e.g. frequency, maximality, monotonicity) used in itemset mining can be formulated for use in CP [14]. This first study leads to the first CP approach for itemset mining displaying nice declarative opportunities without neglecting efficiency. More recently, Coquery et al. [8] have proposed a SAT-Based approach for Discovering for enumerating frequent, closed and maximal patterns with wildcards in a sequence of items. In this paper, we first propose a new SAT encoding of the problem of enumerating frequent, closed and maximal patterns with wildcards in a sequence of items. Our contribution can be seen as an improvement of the approach proposed in [8]. Indeed, the experimental results clearly show that the new encoding is significantly better than the original SAT encoding proposed in [8].

Encouraged by these promising results, we propose in our second contribution a new variant of the problem of discovering patterns with wildcards in a sequence, by considering a sequence of itemsets instead of a sequence of items. In this extension the emptyset will simply play the same role as the wildcard symbol. Indeed, one can use the emptyset to match any itemset. This new problem admits some similarities and differences with the classical sequential pattern

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

http://dx.doi.org/10.1145/2505515.2505577 .

mining problem introduced in [1]. Indeed, given an alphabet or a set of items Σ, in both problems we consider a sequence s as an ordered list of itemsets $s_0, \ldots, s_n$ where $s_i \subseteq \Sigma$ for $i = 0, \ldots, n$. However, the first difference resides in the definition of a subsequence. Indeed, in the sequential patterns, we say that s' is a subsequence of s if there exists a one-to-one order-preserving function f that maps (inclusion relation) itemsets in s' with itemsets in s. In our new setting, the notion of subsequence is defined w.r.t. to a given location and by using empty itemsets as wildcards. The other difference is that in the sequential pattern mining we consider a database of sequences of itemsets, while in our setting, we consider only a single sequence of itemsets. These differences leads also to different definitions of the notions of support, closeness and maximality. In summary, our new problem of discovering patterns in a sequence of itemsets can be seen as a simple and natural extension of the same problem in a sequence of items. As we can show later, the SAT encoding can be derived from the one used for the sequences of items with a very slight modification demonstrating its flexibility.

The paper is organized as follows. In the next section, we give a short overview of necessary definitions and notations about Boolean Satisfiability (SAT) and Frequent Pattern mining in a Sequence of items (FPS). The extension to the Frequent Pattern mining in a Sequence of Itemsets (FPSI) is presented in Section 3, followed by a discussion of some related works. Then our new SAT-based approach for FPS is described in Section 4, while the closeness and maximality constraints are discussed in Section 5. In Section 6, we show how the SAT encoding of FPSI, can be obtained by a slight modification of the SAT encoding of FPS. Finally, experimental results are conducted and discussed before concluding.

2. PRELIMINARIES

2.1 Boolean Satisfiability

In this section, we introduce the Boolean satisfiability problem, called SAT. It corresponds to the problem of deciding if a formula of propositional classical logic is consistent or not. It is one of the most studied NP-complete decision problem. In this work, we consider the associated problem of Boolean model enumeration.

We consider the conjunctive normal form (CNF) representation for the propositional formulas. A *CNF formula* Φ is a conjunction of clauses, where a *clause* is a disjunction of literals. A *literal* is a positive (p) or negated ($\neg p$) propositional variable. The two literals p and $\neg p$ are called *complementary*.

A CNF formula can also be seen as a set of clauses, and a clause as a set of literals. Let us recall that any propositional formula can be translated to CNF using linear Tseitin's encoding [22]. We denote by $Var(\Phi)$ the set of propositional variables occurring in Φ.

An *interpretation* $\mathcal{B}$ of a propositional formula Φ is a function which associates a value $\mathcal{B}(p) \in \{0, 1\}$ (0 corresponds to $false$ and 1 to $true$) to the variables $p \in Var(\Phi)$. A *model* of a formula Φ is an interpretation $\mathcal{B}$ that satisfies the formula. *SAT problem* consists in deciding if a given formula admits a model or not.

We denote by $\bar{l}$ the complementary literal of l, i.e., if $l = p$ then $\bar{l} = \neg p$ and if $l = \neg p$ then $\bar{l} = p$. For a set of literals L, $\bar{L}$ is defined as $\{\bar{l} \mid l \in L\}$. Moreover, $\overline{\mathcal{B}}$ ($\mathcal{B}$ is an interpretation over $Var(\Phi)$) corresponds to the clause $\bigvee_{p \in Var(\Phi)} f(p)$, where if $\mathcal{B}(p) = 0$ then $f(p) = p$, otherwise $f(p) = \neg p$.

Let us informally describe the most important components of modern SAT solvers. They are based on a reincarnation of the historical Davis, Putnam, Logemann and Loveland procedure, commonly called DPLL [9]. It performs a backtrack search; selecting at each level of the search tree, a decision variable which is set to a Boolean value. This assignment is followed by an inference step that deduces and propagates some forced unit literal assignments. This is recorded in the implication graph, a central data-structure, which encodes the decision literals together with there implications. This branching process is repeated until finding a model or a conflict. In the first case, the formula is answered satisfiable, and the model is reported, whereas in the second case, a conflict clause (called learnt clause) is generated by resolution following a bottom-up traversal of the implication graph [17, 24]. The learning or conflict analysis process stops when a conflict clause containing only one literal from the current decision level is generated. Such a conflict clause asserts that the unique literal with the current level (called asserting literal) is implied at a previous level, called assertion level, identified as the maximum level of the other literals of the clause. The solver backtracks to the assertion level and assigns that asserting literal to *true*. When an empty conflict clause is generated, the literal is implied at level 0, and the original formula can be reported unsatisfiable.

In addition to this basic scheme, modern SAT solvers use other components such as activity based heuristics and restart policies. An extensive overview about propositional Satisfiability can be found in [6, 15].

2.2 Frequent Pattern Mining in a Sequence of Items (FPS)

In this section, we present the frequent pattern mining problem of enumerating frequent, closed and maximal patterns with wildcards in a sequence of items [18, 20, 2]. Let us first give some preliminary definitions and notations.

Sequences of items.

Let Σ be a finite set of items, called alphabet. A *sequence of items* s over Σ is a simple sequence of symbols $s_0 \cdots s_{n-1}$ belonging to Σ. We denote by $|s|$ its length and by $\mathcal{P}_s$ the set $\{0, \ldots |s| - 1\}$ of all the locations of its symbols. *A wildcard* is a new symbol $\circ$ which is not in Σ. This symbol matches any symbol of the alphabet.

Pattern.

A *pattern* over Σ is a sequence $p = p_0 \ldots p_{m-1}$, where $p_0 \in \Sigma$, $p_{m-1} \in \Sigma$ and $p_i \in \Sigma \cup \{\circ\}$ for $i = 1, \ldots, m-2$. We say that p is included in $s = s_0 \ldots s_{n-1}$ at the location $l \in \mathcal{P}_s$, denoted $p \preceq_l s$, if $\forall i \in \{0 \ldots m-1\}$, $p_i = s_{l+i}$ or $p_i = \circ$. We also say that p is included in s, denoted $p \preceq s$, if $\exists l \in \mathcal{P}_s$ such that $p \preceq_l s$. The *cover* of p in s is defined as the set $\mathcal{L}_s(p) = \{l \in \mathcal{P}_s | p \preceq_l s\}$. Moreover, The *support* of p in s is defined as the value $|\mathcal{L}_s(p)|$.

FPS problem.

Let s be a sequence, p a pattern and $\lambda \geq 1$ a minimal support threshold, called also a quorum. We say that p is a *frequent pattern in s w.r.t. λ* if $|\mathcal{L}_s(p)| \geq \lambda$. *The frequent pattern mining problem in a sequence of items (FPS)* consists in computing the set $\mathcal{M}_s^\lambda$ of all the frequent patterns w.r.t. λ. For instance, let us consider the sequence $s = aaccbcabcba$ and then pattern $p = a \circ c$. We have $\mathcal{L}_s(p) = \{0, 1, 6\}$, since $p \preceq_0 s$, $p \preceq_1 s$ and $p \preceq_6 s$. In this case, if we consider that the minimal support threshold is equal to the value 3, then the pattern p is a frequent pattern of s.

Closed and Maximal Patterns.

A frequent pattern p of a sequence s is said to be *closed* if for any frequent pattern q satisfying $q \succ p$, there is no integer d such that $\mathcal{L}_s(q) = \mathcal{L}_s(p) + d$, where $\mathcal{L}_s(p) + d = \{l + d | l \in \mathcal{L}_s(p)\}$. Moreover, it is said to be *maximal* if for any frequent pattern q, $q \not\succ p$. Clearly, the set of closed frequent patterns (resp. maximal frequent patterns) is a condensed representation of the set of frequent patterns. Indeed, the frequent patterns can be obtained from the closed (resp. maximal) ones by replacing items with wildcards.

Note that if p_1 and p_2 are two patterns such that $p_1 \preceq p_2$, then if $|\mathcal{L}_s(p_2)| \geq \lambda$ then $|\mathcal{L}_s(p_1)| \geq \lambda$. This property is called anti-monotonicity.

3. FREQUENT PATTERN MINING IN A SEQUENCE OF ITEMSETS (FPSI)

In this section, we define a new variant of the problem of discovering patterns with wildcards in a sequence, by considering a sequence of itemsets instead of a sequence of items. The role of wildcard symbol is nicely played by the empty itemset as it match any itemset. As mentioned in the introduction, this new problem admits some similarities and differences with the classical sequential pattern mining problem introduced in [1]. The main difference resides in the definition of the notion of subsequence (inclusion), where empty itemsets are used as wildcards, and in the use or not of a single or several sequences

A *sequence of itemsets* s over an alphabet Σ is defined as a sequence $s_0, \ldots, s_{n-1}$, where $s_i \subseteq \Sigma$ for $i = 0, \ldots, n-1$. Similarly to the sequences of items, we denote by $|s|$ its length ($|s| = n$) and by $\mathcal{P}_s$ the set $\{0, \ldots |s| - 1\}$ of the locations.

A *pattern* $p = p_0, \ldots, p_{m-1}$ over Σ is also defined as a sequence of itemsets where the first and the last elements are different from the empty itemset. In this context, let us mention that we do not need the wildcard symbol. Indeed, one can use the empty itemset to match any itemset. Furthermore, we say that p is included in $s = s_0 \ldots s_{n-1}$, denoted $p \preceq_l s$, at the location $l \in \mathcal{P}_s$ if $\forall i \in \{0 \ldots m-1\}$, $p_i \subseteq s_{l+i}$. The relation $\preceq$ and the set $\mathcal{L}_s(p)$ are defined in the same way as in the case of the sequences of items. The *cover* (resp. *support*) of p in s is defined as the set $\mathcal{L}_s(p)$ (resp. as the value $|\mathcal{L}_s(p)|$).

The frequent, closed and maximal patterns are also defined in the same way. For instance, a frequent pattern p of a sequence s is said to be *closed* if for any frequent pattern q satisfying $q \succ p$, there is no integer d such that $\mathcal{L}_s(q) = \mathcal{L}_s(p) + d$, where $\mathcal{L}_s(p) + d = \{l + d | l \in \mathcal{L}_s(p)\}$. The frequent patterns can be obtained from the closed (resp. maximal) ones by replacing itemsets with their subsets.

For example, let us consider the sequence of itemsets $s = \{a, b\}, \{a, b\}, \{c, d\}, \{c, e\}, \{f\}, \{g\}, \{d\}, \{a, b, d\}, \{f\}, \{c\}$ and the pattern $p = \{a, b\}, \{\}, \{c\}$. If we set the minimal support threshold to 3, then p is a frequent pattern in s, since $\mathcal{L}_s(p) = \{0, 1, 7\}$. The pattern p is also a closed frequent pattern, but $p' = \{a\}, \{\}, \{c\}$ is not closed, since $p \prec p'$.

The pattern mining task that we consider in the sequences of itemsets allows to exhibit a high degree of self similarity for better understandings of large volumes of data. For instance, a sequence of itemsets can be seen as a record of the articles bought by a customer over a period of time. In such a case, a frequent pattern could be "the customer bought acetylsalicylic acid two days after buying beer and wine in 20% of the days from 2008 to 2012".

3.1 Related Works and Motivations

SAT-based encodings for enumerating frequent, closed and maximal patterns in the sequences of items have been proposed in [8]. They follows the constraint programing (CP) based approach proposed recently by Luc De Raedt et al. in [10] for itemset mining. The SAT and CP based approaches in data mining are proposed in order to offer declarative and flexible frameworks. Indeed, new constraints require often new implementations in specialized approaches, while they can be easily integrated in such frameworks.

In this paper, we propose a SAT-based encodings for enumerating frequent, closed and maximal patterns in the sequences of items and the sequences of itemsets. The choice of SAT comes from our desire to exploit the efficiency of modern SAT solvers [6]. In this context, our encodings can be seen as an improvement and an extension of the encodings proposed in [8]. Indeed, we show experimentally on real world data that our encodings are better than those in [8]. Furthermore, we show that encodings in the case of the sequences of itemsets are obtained by a very slight modification of that for the sequences of items.

4. A NEW SAT-BASED APPROACH FOR FPS

We describe here our new Boolean encoding for the problem of enumerating the frequent patterns in a sequence of items FPS. The base idea consists in using a propositional variable to represent the location of an element of the alphabet in the candidate pattern. Moreover, we use the well-known cardinality constraint to reason about the support of the candidate pattern.

Let $\Sigma = \{a_1, \ldots, a_m\}$ be an alphabet, s a sequence over Σ of length n and λ a minimal support threshold. We associate to each character a appearing in s a set of k_a propositional variables $p_{a,0}, \ldots, p_{a,(k_a-1)}$ such that $k_a = min(max(\mathcal{L}_s(a)) + 1, n - \lambda + 1)$. The variable $p_{a,i}$ means that a is in the candidate pattern at the location i. In fact, that explains why we associate only $min(max(\mathcal{L}_s(a)) + 1, n - \lambda + 1)$ variables to each character a, because $\{0, \ldots, min(max(\mathcal{L}_s(a)), n - \lambda)\}$ corresponds to the set of all possible locations of a in the candidate patterns.

We first need to encode that the first symbol must be a solid character (different from the wildcard symbol). This property is expressed by the following simple clause:

$$\bigvee_{a \in \Sigma} p_{a,0} \tag{1}$$

The following constraint composed of binary clauses allows us to capture the locations where the candidate pattern does not appear:

$$\bigwedge_{a \in \Sigma, 0 \leq l \leq n-1, 0 \leq i \leq k_a - 1} (p_{a,i} \wedge s_{l+i} \neq a) \rightarrow b_l \tag{2}$$

where $b_0, \ldots, b_{n-1}$ are n new propositional variables. In the previous formula $b_j = 1$ if the candidate pattern does not appear in s at the location j. Let us recall that, in classical propositional logic, we have $A \rightarrow B := \neg A \vee B$, and that explains why the previous formula can be seen as a set of binary clauses (the expressions of the for $s_{l+i} \neq a$ are constants, i.e. $s_{l+i} \neq a \in \{0, 1\}$).

In the problem of enumerating all the frequent patterns in s w.r.t. λ, we need to express that the candidate pattern occurs at least λ times. This property is obtained by the following *cardinality constraint*:

$$\sum_{l=0}^{n-1} b_l \leq n - \lambda \tag{3}$$

Indeed, if this constraint is not satisfied, then we know that there exist at least $n - \lambda + 1$ locations where the candidate pattern does not appear. This is equivalent to say that there exist at most $\lambda - 1$ locations where the candidate pattern appears, i.e. it is not frequent. Otherwise, there exist at least λ locations of the candidate pattern , i.e. it is frequent. Hence this constraint allows us to reason about the support of the considered candidate pattern an to decide whether it is greater or equal to the minimal support threshold or not.

The previous constraint involves the well known cardinality constraint (0/1 linear inequality). Several polynomial encoding of this kind of constraints into a CNF formula have been proposed in the literature. The first linear encoding of general linear inequalities to CNF have been proposed by J. P. Warners [23]. Recently, efficient encodings of the cardinality constraint to CNF have been proposed, most of them try to improve the efficiency of constraint propagation (e.g. [4, 21, 3, 16]).

PROPOSITION 1. *The problem of enumerating all frequent patterns in a given sequence s is expressed by the constraints (1), (2) and (3).*

PROOF. We first prove that if $p = a_0, \ldots, a_{k-1}$ is a frequent pattern, then there exists an extension of its corresponding Boolean interpretation $\mathcal{B}_p$ which is a model of (1), (2) and (3). Note that $\mathcal{B}_p$ is defined as follows: for all $a \in \Sigma$ and for all $i \in \{0, \ldots, k_a\}$, if $a_i = a$ then $\mathcal{B}_p(p_{a,i}) = 1$. One can easily see that the constraint (1) is satisfied by $\mathcal{B}_p$, since $\mathcal{B}_p(p_{a_0,0}) = 1$. Let us now extend the Boolean Interpretation $\mathcal{B}_p$ to the variables $b_0, \ldots, b_{n-1}$. This extension is obtained as follows: for all $0 \leq i \leq n-1$, if $p \not\preceq_i s$ then $\mathcal{B}_p(b_i) = 1$. Clearly this extension corresponds to a Boolean interpretation that satisfies (2). Finally, it also satisfies (3), since p is a frequent pattern, i.e. its support is greater or equal to the minimal support threshold λ.

Conversely, we have to prove that if a Boolean interpretation $\mathcal{B}$ is a model of (1), (2) and (3), then there exists a unique pattern $p_{\mathcal{B}}$ corresponding to $\mathcal{B}$ which is frequent. Note that, using the constraints (2) and (3), we have, for all $a, a' \in \Sigma$ and for all $i \in \{0, \ldots, k_a - 1\}$, if $\mathcal{B}(p_{a,i}) = \mathcal{B}(p_{a',i}) = 1$, then $a = a'$. Indeed, if there exists $a \neq a'$ such that $\mathcal{B}(p_{a,i}) = \mathcal{B}(p_{a',i}) = 1$, then we get $\sum_{l=0}^{n-1} b_l = n$ and this is in contradiction with $\sum_{l=0}^{n-1} b_l \leq n - \lambda$ ($\lambda \neq 0$). Furthermore, using the constraint (1), we know that the first symbol of p is different from $\circ$. Therefore, we deduce that there exists a unique pattern associated to $\mathcal{B}$. This pattern corresponds to $p_{\mathcal{B}} = a_0 \cdots a_{k-1}$ such that, for all $i \in \{0, \ldots, k-1\}$ with $a_i \neq \circ$, $\mathcal{B}(p_{a_i,i}) = 1$, and $\mathcal{B}(p_{a,i}) = 0$ for all $a \in \Sigma$ with $a \neq a_i$. Moreover, using the constraint (2), we know that if $\mathcal{B}(b_i) = 1$, then $p \not\preceq_i s$. Hence, using the cardinality constraint (3), we deduce that the support of p is greater or equal to the support threshold λ. □

Example. Consider the frequent pattern mining problem in the case of the sequence *aabb* with 2 as minimal support threshold. Our encoding corresponds to the following formulae:

$p_{a,0} \vee p_{b_0}$
$p_{a,0} \rightarrow (b_2 \wedge b_3)$
$p_{a,1} \rightarrow (b_1 \wedge b_2 \wedge b_3)$
$p_{a,2} \rightarrow (b_0 \wedge b_1 \wedge b_2 \wedge b_3)$
$p_{b,0} \rightarrow (b_0 \wedge b_1)$
$p_{b,1} \rightarrow (b_0 \wedge b_3)$
$p_{b,2} \rightarrow (b_2 \wedge b_3)$
$b_0 + b_1 + b_2 + b_3 \leq 2$

Note that, for all Boolean interpretation $\mathcal{B}$, if $\mathcal{B}(p_{a,i}) = \mathcal{B}(p_{b,i})$, then $b_0 + b_1 + b_2 + b_3 = 4$. Hence we cannot have different solid characters at the same position. Moreover, using the last constraint, for all Boolean interpretation $\mathcal{B}$ which is a model of the encoding, we must have $\mathcal{B}(p_{a,1}) = \mathcal{B}(p_{a,2}) = 0$ and $\mathcal{B}(p_{a,0}) \neq \mathcal{B}(p_{b,1})$. If we describe each Boolean model of the formula by a subset of $\{p_{a,0}, p_{a,1}, p_{a,2}, p_{b,0}, p_{b,1}, p_{b,2}\}$, then we obtain as models $\{p_{a,0}\}$, $\{p_{b,0}\}$ and $\{p_{a,0}, p_{b,2}\}$. These Boolean models correspond to the patterns a, b and $a \circ b$.

Note that in order to consider the frequent patterns with at least *min* solid characters, we just have to add the following constraint:

$$\sum_{a \in \Sigma, 0 \leq i \leq k_a - 1} p_{a,i} \geq min \tag{4}$$

Conversely, in order to only consider the frequent patterns with at most *max* solid characters, we add:

$$\sum_{a \in \Sigma, 0 \leq i \leq k_a - 1} p_{a,i} \geq max \tag{5}$$

Moreover, the combination of the two previous constraints allows us to only consider with the number of solid characters between *min* and *max*. Let us mention that such extensions show that our approach is flexible.

5. ENUMERATING CLOSED AND MAXIMAL MOTIFS

5.1 Enumerating Closed Motifs (CPS)

In order to provide constraints allowing to enumerate the closed frequent patterns, we associate to each symbol a a set of $k_a + (n - min(\mathcal{L}_s(a)) - 1)$ propositional variables:

$$p_{a,-k'_a}, \ldots, p_{a,k_a-1}$$

where $k'_a = n - min(\mathcal{L}_s(a)) - 1$. Similarly to our previous encoding, the propositional variables $p_{a,0}, \ldots, p_{a,k_a-1}$ allow us to reason about the possible locations of a in the candidate pattern. The variables with negative indices are used to force the candidate pattern to be closed. Our encoding of the problem of enumerating the closed frequent patterns in a sequence of items CPS is obtained by extending the previous one with new constraints.

We first have to capture all the locations where the candidates pattern appears. This is obtained by the following constraint:

$$\bigwedge_{l=0}^{n-1} (b_l \rightarrow \bigvee_{a \in \Sigma, 0 \leq i \leq k_a - 1} (p_{a,i} \wedge s_{l+i} \neq a)) \quad (6)$$

Indeed, the previous constraint combined to the constraint (2) allows us to obtain that, if the Boolean interpretation $\mathcal{B}$ is a model of the constraints (1), (2) and (6), then the candidate pattern that corresponds to $\mathcal{B}$ appears only in the locations $\{0 \leq l \leq n-1 | \mathcal{B}(b_l) = 0\}$.

Now, we introduce a necessary, but not sufficient, constraint, w.r.t. the previous constraints, for obtaining a closed frequent pattern:

$$\bigwedge_{a \in \Sigma, 0 \leq i \leq k_a - 1} (\bigwedge_{l=0}^{n-1} \overline{b_l} \rightarrow s_{l+i} = a) \rightarrow p_{a,i} \quad (7)$$

Intuitively, the previous constraint maximizes the number of the symbols different from wildcard on the right side of the symbol represented by the propositional variable having 0 as index.

We now define a constraint with the propositional variables having the negative indices. Conversely to the previous constraint, the following constraint allows us to to maximize the number of the symbols different from wildcard on the left side:

$$\bigwedge_{a \in \Sigma, 1 \leq i \leq k'_a} (\bigwedge_{l=0}^{n-1} \overline{b_l} \rightarrow s_{l-i} = a) \leftrightarrow p_{a,-i} \quad (8)$$

Let us note that if the Boolean interpretation $\mathcal{B}$ is a model of $(1) \wedge (2) \wedge (3) \wedge (6) \wedge (7) \wedge (8)$ and $\mathcal{B}(p_{a,i}) = 1$, then, for all $b \in \Sigma$ such that $a \neq b$ and $p_{b,i}$ exists, $\mathcal{B}(p_{b,i}) = 0$ holds. This property is mainly obtained from the constraints (2) and (8) (see arguments used in the proof of Proposition 1). A closed motif is obtained from a model by using the propositional variables associated to the elements of Σ and evaluated to 1 by this model. Let $p_{a_0,i_0}, p_{a_1,i_1}, \ldots, p_{a_{k-1},i_{k-1}}$ be these variables. In this case, the closed motif is:

$$a_0 \overbrace{\circ \cdots \circ}^{i_1 - i_0 - 1} a_1 \cdots a_{k-1}$$

We now provide another encoding without using propositional variables with negative indices. The idea consists in excluding each interpretation whenever its corresponding pattern is not closed. This encoding is obtained by replacing the constraint (8) with the following constraint:

$$\bigwedge_{a \in \Sigma, 1 \leq i \leq k'_a} \neg(\bigwedge_{l=0}^{n-1} \overline{b_l} \rightarrow s_{l-i} = a) \quad (9)$$

By the previous constraint, we simply force the candidate pattern to be closed.

PROPOSITION 2. *The problem of enumerating all the closed frequent patterns in a given sequence s is expressed by the constraints (1), (2), (3), (6), (7) and (9).*

PROOF. Let $p = a_0, \ldots, a_{k-1}$ be a closed frequent pattern. We define its corresponding Boolean interpretation $\mathcal{B}_p$ as follows: for all $a \in \Sigma$ and for all $i \in \{0, \ldots, k_a\}$, if $a_i = a$ then $\mathcal{B}_p(p_{a,i}) = 1$. We extend $\mathcal{B}_p$ to the propositional variables $\{b_0, \ldots, b_{n-1}\}$ as follows: $p \not\preceq_i s$ iff $\mathcal{B}_p(b_i) = 1$, for $i = 0, \ldots, n-1$. By using similar arguments as in our proof of Proposition 1, we know that $\mathcal{B}_p$ satisfies the constraints (1), (2) and (3). It also satisfies (6), since if $\mathcal{B}_p(b_i) = 1$ then $p \not\preceq_i s$. In this context, the support of p is equal to $|\{b_i \mid \mathcal{B}_p(b_i) = 0\}|$. This allows us to deduce that $\mathcal{B}_p$ satisfies (7) and (9), since p is closed. Indeed, if (7) or (9) are not satisfied by $\mathcal{B}_p$, then there exists a pattern p' such that $p \prec p'$ with a support greater or equal to that of p and we get a contradiction because that means that p is not a closed pattern.

Conversely, consider $\mathcal{B}$ a model of (1), (2), (3), (6), (7) and (9). Using the constraints (2) and (3), we have, for all $a, a' \in \Sigma$ and for all $i \in \{0, \ldots, k_a - 1\}$, if $\mathcal{B}(p_{a,i}) = \mathcal{B}(p_{a',i}) = 1$, then $a = a'$. Hence, there exists a unique pattern associated to $\mathcal{B}$ that corresponds to $p_{\mathcal{B}} = a_0 \cdots a_{k-1}$ such that, for all $i \in \{0, \ldots, k-1\}$ with $a_i \neq \circ$, $\mathcal{B}(p_{a_i,i}) = 1$, and $\mathcal{B}(p_{a,i}) = 0$ for all $a \in \Sigma$ with $a \neq a_i$. Using Prposition 1, we know that the pattern $p_{\mathcal{B}}$ is frequent. The constraint (6) allows us to obtain that the support of p is equal to $|\{b_i \mid \mathcal{B}(b_i) = 0\}|$. Using the constraints (7) and (9), we now that there is no frequent pattern q having the same support as p such that $p \prec q$. Therefore, we deduce that p is a closed frequent pattern. □

Note that in order to enumerate all frequent patterns without any condition on their supports, we only have to remove the constraint 3.

5.2 Enumerating Maximal Motifs (MPS)

In our encoding of the problem of enumerating the maximal frequent patterns in a sequence of items MPS, we only use the propositional variables associated to the elements of Σ with positive indices, i.e. we associate to each symbol a a set of k_a propositional variables $p_{a,0}, \ldots, p_{a,(k_a-1)}$. Our encoding of MPS is obtained by extending the one of FPS in a similar way as our encoding of CPS.

In order to enumerate the maximal frequent patterns, we

need to capture all the locations where the candidates pattern appears. To this end, similarly to our encodings of CPS, we use the constraint (6). Indeed, the combination of the constraints (2) and (6) allows us to obtain, if $\mathcal{B}$ is a Boolean model of these two constraints, then $\{0 \leq l \leq n-1 | \mathcal{B}(b_l) = 0\}$ corresponds to the set of the locations where the candidate pattern appears.

We now provide the constraint allowing to maximize the number of symbols different from wildcard on the right side of the symbol represented by the propositional variable having 0 as index:

$$\bigwedge_{a \in \Sigma, 1 \leq i \leq k_a - 1} (\sum_{l=0}^{n-1} \overline{b_l} \wedge s_{l+i} = a \geq \lambda) \rightarrow p_{a,i} \quad (10)$$

Intuitively, the constraint means that if $p = a_0 \cdots a_{k-1}$ is the patter candidate and there exists $a \in \Sigma$ such that then pattern $a_0 \cdots a_{k-1} \circ \cdots \circ a$ have the same support as p, then p is not a maximal frequent pattern.

Coversely to the previous constraint, we finally introduce the constraint allowing to maximize the number of symbols different from wildcard on the left side of the symbol represented by the propositional variable having 0 as index:

$$\bigwedge_{a \in \Sigma, 1 \leq i \leq k'_a} \neg(\sum_{l=0}^{n-1} \overline{b_l} \wedge s_{l-i} = a \geq \lambda) \quad (11)$$

One can easily see that it is equivalent to the following constraint:

$$\bigwedge_{a \in \Sigma, 1 \leq i \leq k'_a} \sum_{l=0}^{n-1} \overline{b_l} \wedge s_{l-i} = a \leq \lambda - 1 \quad (12)$$

Indeed, the constraint $\neg(\sum_{l=0}^{n-1} x \geq \lambda)$ is equivalent to cardinality constraint $\sum_{l=0}^{n-1} x \leq \lambda - 1$.

PROPOSITION 3. *The problem of enumerating all the maximal frequent patterns in a given sequence s is expressed by the constraints (1), (2), (3), (6), (10) and (12).*

PROOF. Similar to our proof of Proposition 2. □

Let us mention that we can also use the constraints (4) and (5) to reason about the number of the solid characters in the considered patterns in the cases of CPS and MPS. Furthermore, we can use a constraint in order to only consider the closed and maximal patterns with support between λ and λ'. This constraint is the following:

$$\sum_{l=0}^{n-1} b_l \geq n - \lambda' \quad (13)$$

6. SAT-BASED ENCODINGS AND SEQUENCE OF ITEMSETS

In this section, we extend our SAT-based approach for discovering frequent, closed and maximal patterns in a sequence of itemsets. We will show that our encodings in this case can be obtained from the previous ones with a very slight modification.

Our encoding of the problem of enumerating the frequent patterns in a sequence of itemsets FPSI can be easily obtained from the one of FPS. We only have to replace the equalities of the form $s_{l+i} \neq a$ with $a \notin s_{l+i}$:

$$\bigvee_{a \in \Sigma} p_{a,0} \quad (14)$$

$$\bigwedge_{a \in \Sigma, 0 \leq l \leq n-1, 0 \leq i \leq k_a - 1} (p_{a,i} \wedge a \notin s_{l+i}) \rightarrow b_l \quad (15)$$

$$\sum_{l=0}^{n-1} b_l \leq n - \lambda \quad (16)$$

In this case, the variable $p_{a,i}$ means that the symbol a is in the candidate pattern in the itemset at the location i. Let us recall that we use the empty itemset as wildcard.

We denote by CPSI (resp. MPSI) the problem of enumerating the closed (resp. maximal) frequent patterns in a sequence of itemsets. Similarly to FPSI, Boolean encodings of CPSI and MPSI can be directly obtained from the ones of respectively CPS and MPS by replacing the expressions of the form $s_{l+i} \neq a$ (resp. $s_{l+i} = a$) with $a \notin s_{l+i}$ (resp. $a \in s_{l+i}$).

Constraints of closeness:

$$\bigwedge_{l=0}^{n-1} (b_l \rightarrow \bigvee_{a \in \Sigma, 0 \leq i \leq k_a - 1} (p_{a,i} \wedge a \notin s_{l+i})) \quad (17)$$

$$\bigwedge_{a \in \Sigma, 0 \leq i \leq k_a - 1} (\bigwedge_{l=0}^{n-1} \overline{b_l} \rightarrow a \in s_{l+i}) \rightarrow p_{a,i} \quad (18)$$

$$\bigwedge_{a \in \Sigma, 1 \leq i \leq k'_a} \neg(\bigwedge_{l=0}^{n-1} \overline{b_l} \rightarrow a \in s_{l-i}) \quad (19)$$

Constraints of maximality: to express the maximality, we add the following two constraints to (17):

$$\bigwedge_{a \in \Sigma, 1 \leq i \leq k_a - 1} (\sum_{l=0}^{n-1} \overline{b_l} \wedge a \in s_{l+i} \geq \lambda) \rightarrow p_{a,i} \quad (20)$$

$$\bigwedge_{a \in \Sigma, 1 \leq i \leq k'_a} \sum_{l=0}^{n-1} \overline{b_l} \wedge a \in s_{l-i} \leq \lambda - 1 \quad (21)$$

The slight modification of our encodings in the case of the sequences of items in order to obtain encodings for the sequences of itemsets clearly shows the high flexibility of our proposed framework.

7. IMPLEMENTATION AND EXPERIMENTS

In our study, we carried out a preliminary experimental evaluation of our proposed approaches using two different datasets.

1. *Bioinformatics*: proteinic data encoded as a sequence of items, where an item is an amino-acid [1].

[1] http://www.biomedcentral.com/1471-2105/11/175/additional/

2. *Synthetic datasets:* we use the well-known IBM itemset data generator[2] to derive datasets with different features. We used this generator to get sequences of itemsets, that can be seen as an ordered set of transactions.

To make fair our comparison with the approach proposed in [8], we adopted the similar choices in our implementations. First, as we deal with the problem of enumerating all the models of a given CNF formula encoding our sequence mining problem, we implemented a model enumeration solver based on the CDCL-based solver MiniSAT 2.2 [3]. To enumerate all the models, each time a model is found, we add only the negation of the sub-model restricted to the literals encoding the pattern to the formula and we restart the search. Secondly, as our SAT encodings include cardinality constraints, we also use the BDD encoding [5] using BoolVar/PB open source java library [4].

In the first experiment, we compare our new SAT encoding (noted CPS1) against the SAT encoding proposed in [8] (noted CPS2), on bioinformatics datasets (sequences of items). This first evaluation concerns the enumeration of frequent closed patterns with wild cards in a sequence of items. We consider a sequence of fixed length and we measure the evolution of computation time with respect to the minimal support threshold (quorum) λ. The quorum evolves linearly ($\lambda_0 = 5$ and $\lambda_i = \lambda_{i-1} + 5$). Several datasets have been considered, their evaluation shows similar behavior. The results obtained on a representative dataset are depicted in Figure 1. This experiment confirms that in the case of sequences of items, our new SAT-based sequence mining approach outperforms (in terms of CPU time) the approach proposed recently in [8]. We also obtain significant improvement w.r.t. the size of the encoding. For instance, if we consider the most difficult dataset of the Figure 1 (quorum $\lambda = 5$), the obtained CNF formula using our encoding contains about 19 millions of clauses and 3 millions of variables, whereas with the encoding proposed in [8] the formula contains about 30 millions of clauses and 7.5 millions of variables.

The second experiment concerns the new extension of the problem to the case of sequence of itemsets. Our goal is show the feasibility of our proposed extension and its associated encodings. For our evaluation, we considered synthetic datasets, generated using the approach outlined in [1] and also used by several authors (e.g. [12]). In our context, we only consider a single sequence of itemsets with different features (size of the sequence, number of items, average size of the itemsets). In Figure 2, we illustrate the results obtained on two datasets: dataset1 (size of the sequence = 100, avg. size of itemsets = 15, number of items = 40) and dataset2 obtained from dataset1 by cutting the sequence at 50th position. The quorum is also varied linearly as in the first experiment. The main observation that can be made, is that the hardness of the enumeration problem increase as the quorum decrease. Indeed, for smaller values of λ, the number of frequent closed patterns is huge, leading to even harder problems. However for higher values of λ, the enumeration problem becomes easy as the number of interesting patterns decreases.

[2]http://sourceforge.net/projects/ibmquestdatagen/
[3]MiniSAT: http://minisat.se/
[4]BoolVAR/PB : http://boolvar.sourceforge.net/

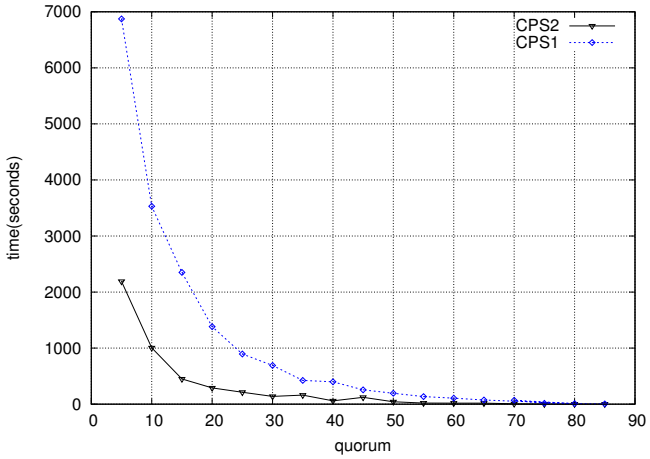


Figure 1: Bioinfo: time Vs quorum (Sequence of Items - Frequent Closed Patterns)

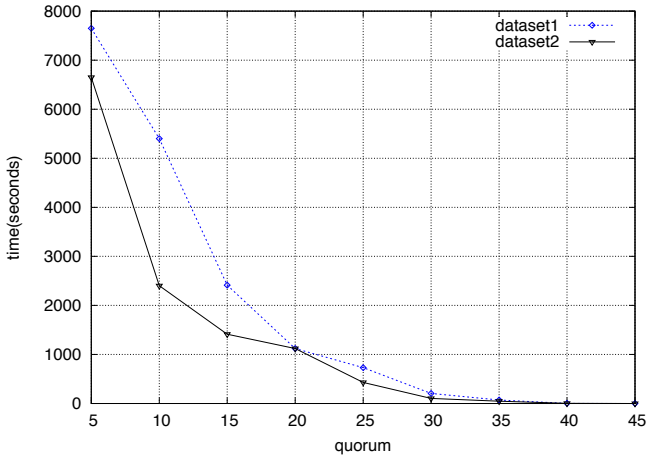


Figure 2: Synthetic datasets: time Vs quorum (Sequence of Itemsets - Frequent Closed Patterns)

As a summary, in the above experiments, we have shown that our encoding significantly improve the one proposed in [8, 7]. For comparison purposes, we used the same encoding of the cardinality constraint and also the same algorithm for enumerating all models of a CNF formula. We think that several room for future improvements can be obtained by using the state-of-the-art encoding of the cardinality [3], and by using more efficient model enumeration algorithm [13, 11, 25]. Obviously, our SAT based approach is less efficient than dedicated approaches such as the state-of-the-art algorithm proposed by Arimura et al. in [2]. However, our SAT model is declarative and highly flexible. Indeed, the SAT encoding for a sequence of itemsets is obtained with a very slight modification of the SAT encoding of a sequence of items. Also, one can easily combine several kind of constraints. Finally, SAT-based data mining benefits from the continuous progress of SAT community.

8. AKNOWLEDGMENTS

We thank the reviewers for their helpful comments. This work has been supported in part by the CNRS and the French ANR project *"DAG: Declarative Approaches for Enumerating Interesting Patterns"* under the Défis program 2009.

9. CONCLUSION AND PERSPECTIVES

The contributions of this paper are twofolds. First, we proposed an interesting improvement of the SAT-based encodings introduced in [8] for enumerating frequent, closed and maximal patterns with wildcards in a sequence of items. Secondly, we introduced a new and natural extension of the problem to deal with the sequences of itemsets. Interestingly, its encoding to SAT is obtained with a slight modification of the SAT encoding of the problem dealing with the sequences of items. This clearly shows the high flexibility of our proposed framework and opens several issues for future research. We first plan to investigate other variants of the problem such as sequences of sequences of items or itemsets. It would be interesting to extend our encoding with constraints on the form of the enumerated patterns (restriction on the number of consecutive wildcards, regular expressions, etc). Finally, on the Boolean satisfiability side, the design of efficient model generation procedures is an important issue for SAT-based data mining framework in general and to other important application domains.

10. REFERENCES

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In A. L. P. C. Philip S. Yu, editor, *Proceedings of the Eleventh International Conference on Data Engineering (ICDE'1995)*, pages 3–14. IEEE Computer Society, 1995.

[2] H. Arimura and T. Uno. An efficient polynomial space and polynomial delay algorithm for enumeration of maximal motifs in a sequence. *Journal of Combinatorial Optimization*, 13, 2007.

[3] R. Asin, R. Nieuwenhuis, A. Oliveras, and E. Rodriguez-Carbonell. Cardinality networks: a theoretical and empirical study. *Constraints*, 16(2):195–221, 2011.

[4] O. Bailleux and Y. Boufkhad. Efficient CNF Encoding of Boolean Cardinality Constraints. In *9th International Conference on Principles and Practice of Constraint Programming - CP 2003*, pages 108–122, 2003.

[5] O. Bailleux, Y. Boufkhad, and O. Roussel. A translation of pseudo boolean constraints to sat. *Journal on Satisfiability, Boolean Modeling and Computation (JSAT)*, 2(1-4), 2006.

[6] A. Biere, M. J. H. Heule, H. van Maaren, and T. Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in AI and Applications*. IOS Press, 2009.

[7] E. Coquery, S. Jabbour, and L. Sais. A constraint programming approach for enumerating motifs in a sequence. In M. Spiliopoulou, H. Wang, D. J. Cook, J. Pei, W. Wang, O. R. Zaïane, and X. Wu, editors, *ICDM Workshops*, pages 1091–1097. IEEE, 2011.

[8] E. Coquery, S. Jabbour, L. Saïs, and Y. Salhi. A SAT-Based Approach for Discovering Frequent, Closed and Maximal Patterns in a Sequence. In *20th European Conference on Artificial Intelligence ECAI*, pages 258–263, 2012.

[9] M. Davis, G. Logemann, and D. W. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, 1962.

[10] L. De Raedt, T. Guns, and S. Nijssen. Constraint Programming for Itemset Mining. In *Proceedings of the 14th International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD'2008)*, pages 204–212, Las Vegas, Nevada, USA, August 24-27 2008.

[11] M. Gebser, B. Kaufmann, A. Neumann, and T. Schaub. *clasp* : A conflict-driven answer set solver. In *9th International Conference Logic Programming and Nonmonotonic Reasoning (LPNMR'2007)*, volume 4483 of *Lecture Notes in Computer Science*, pages 260–265. Springer, 2007.

[12] K. Gouda, M. Hassaan, and M. J. Zaki. Prism: A primal-encoding approach for frequent sequence mining. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007)*, pages 487–492, Omaha, Nebraska, USA, October 28-31 2007. IEEE Computer Society.

[13] O. Grumberg, A. Schuster, and A. Yadgar. Memory efficient all-solutions sat solver and its application for reachability analysis. In *In Proceedings of the 5th International Conference on Formal Methods in Computer-Aided Design (FMCAD)*, pages 275–289. Springer, 2004.

[14] T. Guns, S. Nijssen, and L. D. Raedt. Itemset mining: A constraint programming perspective. *Artificial Intelligence*, 175(12-13):1951–1983, 2011.

[15] Y. Hamadi, S. Jabbour, and L. Sais. Learning from conflicts in propositional satisfiability. *4OR*, 10(1):15–32, 2012.

[16] S. Jabbour, L. Sais, and Y. Salhi. A pigeon-hole based encoding of cardinality constraints. In *In proceedings of the 29th International Conference on Logic Programming (ICLP'2013)*, August 24-29 2013.

[17] J. P. Marques-Silva and K. A. Sakallah. GRASP - A New Search Algorithm for Satisfiability. In *Proceedings of IEEE/ACM CAD*, pages 220–227, 1996.

[18] L. Parida, I. Rigoutsos, A. Floratos, D. Platt, and Y. Gao. Pattern Discovery on Character Sets and Real-valued Data: Linear Bound on Irredundant Motifs and an Efficient Polynomial Time Algorithm. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 297–308, 2000.

[19] L. Parida, I. Rigoutsos, and D. Platt. An output-sensitive flexible pattern discovery algorithm. In *In proceedings of the 12th Annual Symposium on Combinatorial Pattern Matching (CPM'2001)*, volume 2089 of *Lecture Notes in Computer Science*, pages 131–142. Springer, 2001.

[20] N. Pisanti, M. Crochemore, R. Grossi, and M.-F. Sagot. Bases of motifs for generating repeated patterns with wild cards. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB'2005)*, 2(1):40–50, 2005.

[21] C. Sinz. Towards an Optimal CNF Encoding of Boolean Cardinality Constraints. In *11th International*

Conference on Principles and Practice of Constraint Programming - CP 2005, pages 827–831, 2005.

[22] G. Tseitin. On the complexity of derivations in the propositional calculus. In H. Slesenko, editor, *Structures in Constructives Mathematics and Mathematical Logic, Part II*, pages 115–125, 1968.

[23] J. P. Warners. A linear-time transformation of linear inequalities into conjunctive normal form. *Information Processing Letters*, 68(2):63–69, 1998.

[24] L. Zhang, C. F. Madigan, M. W. Moskewicz, and S. Malik. Efficient conflict driven learning in Boolean satisfiability solver. In *IEEE/ACM CAD'2001*, pages 279–285, 2001.

[25] W. Zhao and W. Wu. Asig: An all-solution sat solver for cnf formulas. In *11th International Conference on Computer-Aided Design and Computer Graphics, CAD/Graphics 2009, Huangshan, China, August 19-21 (CAD/Graphics)*, pages 508–513. IEEE, 2009.

The Top-k Frequent Closed Itemset Mining Using Top-k SAT Problem

Said Jabbour, Lakhdar Sais and Yakoub Salhi

CRIL - CNRS, Université d'Artois, France
F-62307 Lens Cedex, France
{jabbour, sais, salhi}@cril.fr

Abstract. In this paper, we introduce a new problem, called Top-k SAT, that consists in enumerating the Top-k models of a propositional formula. A Top-k model is defined as a model with less than k models preferred to it with respect to a preference relation. We show that Top-k SAT generalizes two well-known problems: the partial Max-SAT problem and the problem of computing minimal models. Moreover, we propose a general algorithm for Top-k SAT. Then, we give the first application of our declarative framework in data mining, namely, the problem of enumerating the Top-k frequent closed itemsets of length at least min ($\mathcal{FCIM}^k_{min}$). Finally, to show the nice declarative aspects of our framework, we encode several other variants of $\mathcal{FCIM}^k_{min}$ into the Top-k SAT problem.

1 Introduction

The problem of mining frequent itemsets is well-known and essential in data mining, knowledge discovery and data analysis. It has applications in various fields and becomes fundamental for data analysis as datasets and datastores are becoming very large. Since the first article of Agrawal [1] on association rules and itemset mining, the huge number of works, challenges, datasets and projects show the actual interest in this problem (see [2] for a recent survey of works addressing this problem). Important progress has been achieved for data mining and knowledge discovery in terms of implementations, platforms, libraries, etc. As pointed out in [2], several works deal with designing highly scalable data mining algorithms for large scale datasets. An important problem of itemset mining and data mining problems, in general, concerns the huge size of the output, from which it is difficult for the user to retrieve relevant informations. Consequently, for practical data mining, it is important to reduce the size of the output, by exploiting the structure of the itemsets data. Computing for example, closed, maximal, condensed, discriminative itemset patterns are some of the well-known and useful techniques. Most of the works on itemset mining require the specification of a minimum support threshold λ. This constraint allows the user to control at least to some extent the size of the output by mining only itemsets covering at least λ transactions. However, in practice, it is difficult for users to provide an appropriate threshold. As pointed out in [3], a too small threshold may lead to the generation of a huge number of itemsets, whereas a too high value of the threshold may result in no answer. In [3], based on a total ranking between patterns, the authors propose to mine the n most interesting itemsets of arbitrary length. In [4], the proposed task consists in mining Top-k frequent closed itemsets of

length greater than a given lower bound min, where k is the desired number of frequent closed itemsets to be mined, and min is the minimal length of each itemset. The authors demonstrate that setting the minimal length of the itemsets to be mined is much easier than setting the usual frequency threshold. Since the introduction of Top-k mining, several research works investigated its use in graph mining (e.g. [5, 6]) and other datamining tasks (e.g. [7, 8]). This new framework can be seen as a nice way to mine the k preferred patterns according to some specific constraints or measures. Starting from this observation, our goal in this paper is to define a general logic based framework for enumerating the Top-k preferred patterns according to a predefined preference relation.

The notion of preference has a central role in several disciplines such as economy, operations research and decision theory in general. Preferences are relevant for the design of intelligent systems that support decisions. Modeling and reasoning with preferences play an increasing role in Artificial Intelligence (AI) and its related fields such as nonmonotonic reasoning, planning, diagnosis, configuration, constraint programming and other areas in knowledge representation and reasoning. For example, in nonmonotonic reasoning the introduction of preferential semantics by Shoham [9] gives an unifying framework where nonmonotonic logic is reduced to a standard logic with a preference relation (order) on the models of that standard logic. Several models for representing and reasoning about preferences have been proposed. For example, soft constraints [10] are one of the most general way to deal with quantitative preferences, while CP-net (Conditional Preferences networks) [11] is most convenient for qualitative preferences. There is a huge literature on preferences (see [12–14] for a survey at least from the AI perspective). In this paper we focus on qualitative preferences defined by a preference relation on the models of a propositional formula. Preferences in propositional satisfiability (SAT) has not received a lot of attention. In [15], a new approach for solving satisfiability problems in the presence of qualitative preferences on literals (defined as partial ordered set) is proposed. The authors particularly show how DPLL procedure can be easily adapted for computing optimal models induced by the partial order. The issue of computing optimal models using DPLL has also been investigated in SAT [16].

The contribution of this paper is twofold. Firstly, we propose a generic framework for dealing with qualitative preferences in propositional satisfiability. Our qualitative preferences are defined using a reflexive and transitive relation (preorder) over the models of a propositional formula. Such preference relation on models is first used to introduce a new problem, called Top-k SAT, defined as the problem of enumerating the Top-k models of a propositional formula. Here a Top-k model is defined as a model with no more than k-1 models preferred to it with respect to the considered preference relation. Then, we show that Top-k SAT generalizes the two well-known problems, the partial Max-SAT problem and the problem of generating minimal models. We also define a particular preference relation that allows us to introduce a general algorithm for computing the Top-k models.

Secondly, we introduce the first application of our declarative framework to data mining. More precisely, we consider the problem of mining Top-k frequent closed itemsets

of minimum length min [17]. In this problem, the minimum support threshold usually used in frequent itemset mining is not known, while the minimum length can be set to 0 if one is interested in itemsets of arbitrary length. In itemset mining, the notion of Top-k frequent itemsets is introduced as an alternative to finding the appropriate value for the minimum support threshold. It is also an elegant way to control the size of the output. Consequently, itemset mining is clearly a nice application of our new defined Top-k SAT problem. In this paper, we provide a SAT encoding and we show that computing the Top-k closed itemsets of length at least min corresponds to computing the Top-k models of the obtained propositional formula. Finally, to show the nice declarative aspects of our framework, we encode several other variants of this data mining problem as Top-k SAT problems. Finally, preliminary experiments on some datasets show the feasibility of our proposed approach.

2 Preliminary definitions and notations

In this section, we describe the Boolean satisfiability problem (SAT) and some necessary notations. We consider the conjunctive normal form (CNF) representation for the propositional formulas. A *CNF formula* Φ is a conjunction of clauses, where a *clause* is a disjunction of literals. A *literal* is a positive (p) or negated ($\neg p$) propositional variable. The two literals p and $\neg p$ are called *complementary*. A CNF formula can also be seen as a set of clauses, and a clause as a set of literals. Let us recall that any propositional formula can be translated to CNF using linear Tseitin's encoding [18]. We denote by $Var(\Phi)$ the set of propositional variables occuring in Φ.

An *interpretation* $\mathcal{M}$ of a propositional formula Φ is a function which associates a value $\mathcal{M}(p) \in \{0, 1\}$ (0 corresponds to $false$ and 1 to $true$) to the variables p in a set V such that $Var(\Phi) \subseteq V$. A *model* of a formula Φ is an interpretation $\mathcal{M}$ that satisfies the formula. The *SAT problem* consists in deciding if a given CNF formula admits a model or not.

We denote by $\bar{l}$ the complementary literal of l. More precisely, if $l = p$ then $\bar{l}$ is $\neg p$ and if $l = \neg p$ then $\bar{l}$ is p. For a set of literals L, $\bar{L}$ is defined as $\{\bar{l} \mid l \in L\}$. Moreover, we denote by $\overline{\mathcal{M}}$ ($\mathcal{M}$ is an interpretation over $Var(\Phi)$) the clause $\bigvee_{p \in Var(\Phi)} s(p)$, where $s(p) = p$ if $\mathcal{M}(p) = 0$, $\neg p$ otherwise. Let Φ be a CNF formula and $\mathcal{M}$ an interpretation over $Var(\Phi)$. We denote by $\mathcal{M}(\Phi)$ the set of clauses satisfied by $\mathcal{M}$. Let us now consider a set X of propositional variables such that $X \subseteq Var(\Phi)$. We denote by $\mathcal{M} \cap X$ the set of variables $\{p \in X | \mathcal{M}(p) = 1\}$. Moreover, we denote by $\mathcal{M}_{|X}$ the restriction of the model $\mathcal{M}$ to X.

3 Preferences and Top-k Models

Let Φ be a propositional formula and Λ_Φ the set of all its models. A preference relation $\succeq$ over Λ_Φ is a reflexive and transitive binary relation (a preorder). The statement $\mathcal{M} \succeq$

$\mathcal{M}'$ means that $\mathcal{M}$ is at least as preferred as $\mathcal{M}'$. We denote by $P(\Phi, \mathcal{M}, \succeq)$ the subset of Λ_Φ defined as follows:

$$P(\Phi, \mathcal{M}, \succeq) = \{\mathcal{M}' \in \Lambda_\Phi \mid \mathcal{M}' \succ \mathcal{M}\}$$

where $\mathcal{M}' \succ \mathcal{M}$ means that $\mathcal{M}' \succeq \mathcal{M}$ holds but $\mathcal{M} \succeq \mathcal{M}'$ does not. It corresponds to all models that are strictly preferred to $\mathcal{M}$.

We now introduce an equivalence relation $\approx_X$ over $P(\Phi, \mathcal{M}, \succeq)$, where X is a set of propositional variables. It is defined as follows:

$$\mathcal{M}' \approx_X \mathcal{M}'' \text{ iff } \mathcal{M}' \cap X = \mathcal{M}'' \cap X$$

Thus, the set $P(\Phi, \mathcal{M}, \succeq)$ can be partitioned into a set of equivalence classes by $\approx_X$, denoted by $[P(\Phi, \mathcal{M}, \succeq)]^X$. In our context, this equivalence relation is used to take into consideration only a subset of propositional variables. For instance, we introduce new variables in Tseitin's translation [18] of propositional formula to CNF, and such variables are not important in the case of some preference relations.

Definition 1 (Top-k Model). *Let Φ be a propositional formula, $\mathcal{M}$ a model of Φ, $\succeq$ a preference relation over the models of Φ and X a set of propositional variables. $\mathcal{M}$ is a Top-k model w.r.t. $\succeq$ and X iff $|[P(\Phi, \mathcal{M}, \succeq)]^X| \leq k - 1$.*

Let us note that the number of the Top-k models of a formula is not necessarily equal to k. Indeed, it can be strictly greater or smaller than k. For instance, if a formula is unsatisfiable, then it does not have a Top-k model for any $k \geq 1$. Furthermore, if the considered preference relation is a total order, then the number of Top-k models is always smaller than or equal to k.

It is easy to see that we have the following *monotonicity property*: if $\mathcal{M}$ is a Top-k model and $\mathcal{M}' \succeq \mathcal{M}$, then $\mathcal{M}'$ is also a Top-k model.

Top-k SAT problem. Let Φ be propositional formula, $\succeq$ a preference relation over the models of Φ, X a set of propositional variables and k a strictly positive integer. The Top-k SAT problem consists in computing a set $\mathcal{L}$ of Top-k models of Φ with respect to $\succeq$ and X satisfying the two following properties:

1. for all $\mathcal{M}$ Top-k model, there exists $\mathcal{M}' \in \mathcal{L}$ such that $\mathcal{M} \approx_X \mathcal{M}'$; and
2. for all $\mathcal{M}$ and $\mathcal{M}'$ in $\mathcal{L}$, if $\mathcal{M} \neq \mathcal{M}'$ then $\mathcal{M} \not\approx_X \mathcal{M}'$.

The two previous properties come from the fact that we are only interested in the truth values of the variables in X. Indeed, the first property means that, for all Top-k model, there is a model in $\mathcal{L}$ equivalent to it with respect to $\approx_X$. Moreover, the second property means that $\mathcal{L}$ does not contain two equivalent Top-k models.

In the following definition, we introduce a particular type of preference relations, called δ-preference relation, that allows us to introduce a general algorithm for computing Top-k models.

Definition 2. *Let Φ be a formula and $\succeq$ a preference relation on the models of Φ. Then $\succeq$ is a δ-preference relation, if there exists a polytime function $f_{\succeq}$ from Boolean interpretations to the set of CNF formulae such that, for all $\mathcal{M}$ model of Φ and for all $\mathcal{M}'$ Boolean interpretation, $\mathcal{M}'$ is a model of $\Phi \wedge f_{\succeq}(\mathcal{M})$ iff $\mathcal{M}'$ is a model of Φ and $\mathcal{M} \not\succ \mathcal{M}'$.*

Note that, given a model $\mathcal{M}$ of a CNF formula Φ, $f_{\succeq}(\mathcal{M})$ is a formula such that when added to Φ together with $\overline{\mathcal{M}}$, the models of the resulting formula are different from $\mathcal{M}$ and they are at least as preferred as $\mathcal{M}$. Intuitively, this can be seen as a way to introduce a lower bound during the enumeration process. From now, we only consider δ-preference relations.

3.1 Top-k SAT and Partial MAX-SAT

In this section, we show that the Top-k SAT problem generalizes the Partial MAX-SAT problem (e.g. [19]). In Partial MAX-SAT each clause is either relaxable (soft) or non-relaxable (hard). The objective is to find an interpretation that satisfies all the hard clauses together with the maximum number of soft clauses. The MAX-SAT problem is a particular case of Partial MAX-SAT where all the clauses are relaxable.

Let $\Phi = \Phi_h \wedge \Phi_s$ be a partial MAX-SAT instance such that Φ_h is the hard part and Φ_s the soft part. The relation denoted by $\succeq_{\Phi_s}$ corresponds to preference relation defined as follows: for all $\mathcal{M}$ and $\mathcal{M}'$ models of Φ_h defined over $Var(\Phi_h \wedge \Phi_s)$, $\mathcal{M} \succeq_{\Phi_s} \mathcal{M}'$ if and only if $|\mathcal{M}(\Phi_s)| \geq |\mathcal{M}'(\Phi_s)|$.

Note that $\succeq_{\Phi_s}$ is a δ-preference relation. Indeed, we can define $f_{\succeq_{\Phi_s}}$ as follows:

$$f_{\succeq_{\Phi_s}}(\mathcal{M}) = (\bigwedge_{C \in \Phi_s} p_C \leftrightarrow C) \wedge \sum_{C \in \Phi_s} p_C \geq |\mathcal{M}(\Phi_s)|$$

where p_C for $C \in \Phi_s$ are fresh propositional variables.

The Top-1 models of Φ_h with respect to $\succeq_{\Phi_s}$ and $Var(\Phi)$ correspond to the set of all solutions of Φ in Partial Max-SAT. Naturally, they are the most preferred models with respect to $\succeq_{\Phi_s}$, and that means they satisfy Φ_h and satisfy the maximum number of clauses in Φ_s. Thus, the Top-k SAT problem can be seen as a generalization of Partial MAX-SAT.

The formula $f_{\succeq_{\Phi_s}}(\mathcal{M})$ involves the well-known cardinality constraint (0/1 linear inequality). Several polynomial encodings of this kind of constraints into a CNF formula have been proposed in the literature. The first linear encoding of general linear inequalities to CNF has been proposed by Warners [20]. Recently, efficient encodings of the cardinality constraint to CNF have been proposed, most of them try to improve the efficiency of constraint propagation (e.g. [21, 22]).

3.2 Top-k SAT and X-minimal Model Generation Problem

Let $\mathcal{M}$ and $\mathcal{M}'$ be two Boolean interpretations and X a set of propositional variables. Then, $\mathcal{M}$ is said to be smaller than $\mathcal{M}'$ with respect to X, written $\mathcal{M} \preceq_X \mathcal{M}'$, if

Algorithm 1: Top-k

Input: a CNF formula Φ, a preorder relation $\succeq$, an integer $k \geq 1$, and a set X of Boolean variables
Output: A set of Top-k models $\mathcal{L}$

$\Phi' \leftarrow \Phi$;
$\mathcal{L} \leftarrow \emptyset$; /* Set of all Top-$k$ models */
while (solve(Φ')) **do** /* $\mathcal{M}$ is a model of Φ' */
  if ($\exists \mathcal{M}' \in \mathcal{L}. \mathcal{M} \approx_X \mathcal{M}'$ & $\mathcal{M} \succ \mathcal{M}'$) **then**
    replace($\mathcal{M}, \mathcal{M}', \mathcal{L}$);
  else if ($\forall \mathcal{M}' \in \mathcal{L}. \mathcal{M} \not\approx_X \mathcal{M}'$ & $|\text{preferred}(\mathcal{M}, \mathcal{L})| < k$) **then**
    $S \leftarrow$ min_top($k, \mathcal{L}$);
    add($\mathcal{M}, \mathcal{L}$);
    remove($k, \mathcal{L}$);
    $S \leftarrow$ min_top($k, \mathcal{L}$) $\setminus S$;
    $\Phi' \leftarrow \Phi' \wedge \bigwedge_{\mathcal{M}' \in S} f_{\succeq}(\mathcal{M}')$;
  else
    $\Phi' \leftarrow \Phi' \wedge f_{\succeq}(\mathcal{M})$
  $\Phi' \leftarrow \Phi' \wedge \overline{\mathcal{M}}$;
return $\mathcal{L}$;

$\mathcal{M} \cap X \subseteq \mathcal{M}' \cap X$. We now consider $\preceq_X$ as a preference relation, i.e., $\mathcal{M} \preceq_X \mathcal{M}'$ means that $\mathcal{M}$ is at least as preferred as $\mathcal{M}'$.

We now show that $\preceq_X$ is a δ-preference relation. We can define $f_{\preceq_X}$ as follows:

$$f_{\preceq_X}(\mathcal{M}) = (\bigvee_{p \in \mathcal{M} \cap X} \overline{p}) \vee \bigwedge_{p' \in X \setminus \mathcal{M}} \overline{p'}$$

Absolutely, $\mathcal{M}'$ is a model of a formula $\Phi \wedge \overline{\mathcal{M}} \wedge f_{\preceq_X}(\mathcal{M})$ if and only if $\mathcal{M}'$ is a model of Φ, $\mathcal{M}' \neq \mathcal{M}$, and either $\mathcal{M}' \cap X = \mathcal{M} \cap X$ or $(\mathcal{M} \cap X) \setminus (\mathcal{M}' \cap X) \neq \emptyset$. The two previous statements mean that $\mathcal{M} \not\approx_X \mathcal{M}'$. In fact, if $\mathcal{M}'$ satisfies $\bigwedge_{p' \in X \setminus \mathcal{M}} \overline{p'}$, then $\mathcal{M}' \cap X \subseteq \mathcal{M} \cap X$ holds. Otherwise, $\mathcal{M}'$ satisfies $\bigvee_{p \in \mathcal{M} \cap X} \overline{p}$ and that means that $(\mathcal{M} \cap X) \setminus (\mathcal{M}' \cap X) \neq \emptyset$. This latter statement expresses that either $\mathcal{M}' \cap X \subset \mathcal{M} \cap X$ or $\mathcal{M}$ and $\mathcal{M}'$ are incomparable with respect to $\preceq_X$.

Let Φ be a propositional formula, X a set of propositional variables and $\mathcal{M}$ a model of Φ. Then $\mathcal{M}$ is said to be an *X-minimal model* of Φ if there is no model strictly smaller than $\mathcal{M}$ with respect to $\preceq_X$. In [23], it was shown that finding an X-minimal model is $P^{NP[O(log(n))]}$-hard, where n is the number of propositional variables.

The set of all X-minimal models corresponds to the set of all top-1 models with respect to $\preceq_X$ and $Var(\Phi)$. Indeed, if $\mathcal{M}$ is a top-1 model, then there is no model $\mathcal{M}'$ such that $\mathcal{M}' \prec_X \mathcal{M}$, and that means that $\mathcal{M}$ is an X-minimal model. In this context, let us note that computing the set of Top-k models for $k \geq 1$ can be seen as a generalization of X-minimal model generation problem.

3.3 An algorithm for Top-k SAT

In this section, we describe our algorithm for computing Top-k models in the case of the δ-preference relations (Algorithm 1). The basic idea is simply to use the formula $f_{\succeq}(\mathcal{M})$ associated to a model $\mathcal{M}$ to obtain models that are at least as preferred as $\mathcal{M}$.

This algorithm takes as input a CNF formula Φ, a preference relation $\succeq$, a strictly positive integer k, and a set X of propositional variables allowing to define the equivalence relation $\approx_X$. It has as output a set $\mathcal{L}$ of Top-k models of Φ satisfying the two properties given in the definition of the Top-k SAT problem.

Algorithm description In the while-loop, we use lower bounds for finding optimal models. These lower bounds are obtained by using the fact that the preorder relation considered is a δ-preference relation. In each step, the lower bound is integrated by using the formula:

$$\bigwedge_{\mathcal{M}' \in S} f_{\succeq}(\mathcal{M}')$$

- **Lines 4 – 5.** Let us first mention that the procedure `replace(`$\mathcal{M}$`,`$\mathcal{M}'$`,`$\mathcal{L}$`)` replaces $\mathcal{M}'$ with $\mathcal{M}$ in $\mathcal{L}$. We apply this replacement because there exists a model $\mathcal{M}'$ in $\mathcal{L}$ which is equivalent to $\mathcal{M}'$ and $\mathcal{M}$ allows to have a better bound.
- **Lines 6 – 11.** In the case where $\mathcal{M}$ is not equivalent to any model in $\mathcal{L}$ and the number of models in $\mathcal{L}$ preferred to it is strictly less than k ($|\texttt{preferred}(\mathcal{M}, \mathcal{L})| < k$), we add $\mathcal{M}$ to $\mathcal{L}$ ($\texttt{add}(\mathcal{M},\mathcal{L})$). Note that $\mathcal{S}$ contains first the models of $\mathcal{L}$ before adding $\mathcal{M}$ that have exactly $k-1$ models preferred to them in this set. After adding $\mathcal{M}$ to $\mathcal{L}$, we remove from $\mathcal{L}$ the models that are not Top-k, i.e., they have more than $k-1$ models in $\mathcal{L}$ that are strictly preferred to them ($\texttt{remove}(k, \mathcal{L})$). Next, we modify the content of S. Note that the elements of S before adding $\mathcal{M}$ are used as bounds in the previous step. Hence, in order to avoid adding the same bound several times, the new content of S corresponds to the models in $\mathcal{L}$ that have exactly $k-1$ models preferred to them in $\mathcal{L}$ ($\texttt{min\_top}(k, \mathcal{L})$) deprived of the elements of the previous content of S. In line 11, we integrate lower bounds in Φ' by using the elements of S. Indeed, for all model $\mathcal{M}$ of a formula $\Phi' \wedge \bigwedge_{\mathcal{M}' \in S} f_{\succeq}(\mathcal{M}')$, $\mathcal{M}' \not\succ \mathcal{M}$ holds, for any $\mathcal{M}' \in S$.
- **Lines 12 – 13.** In the case where $\mathcal{M}$ is not a Top-k model, we integrate its associated lower bound.
- **Line 14.** This instruction enables us to avoid finding the same model in two different steps of the while-loop.

Proposition 1. *Algorithm 1 (Top-k) is correct.*

Proof. The proof of the partial correctness is based on the definition of the δ-preference relation. Indeed, the function $f_{\succeq}$ allows us to exploit bounds to systematically improve the preference level of the models. As the number of models is bounded, adding the negation of the found model at each iteration leads to an unsatisfiable formula. Consequently the algorithm terminates.

As explained in the algorithm description, we use lower bounds for finding optimal models. These bounds are obtained by using the function $f_{\succeq}$.

4 Total Preference Relation

We here provide a second algorithm for computing Top-k models in the case of the total δ-preference relations (Algorithm 2). Let us recall that a δ-preference relation $\succeq$ is total

if, for all models $\mathcal{M}$ and $\mathcal{M}'$, we have $\mathcal{M} \succeq \mathcal{M}'$ or $\mathcal{M}' \succeq \mathcal{M}$.

Our algorithm in this case is given in Algorithm 2:

- **Lines 3 – 8**. In this part, we compute a set $\mathcal{L}$ of k different models of Φ such that, for all $\mathcal{M}, \mathcal{M}' \in \mathcal{L}$ with $\mathcal{M} \neq \mathcal{M}'$, we have $\mathcal{M} \not\approx_X \mathcal{M}'$. Indeed, if $\mathcal{M}$ is a model of Φ and $\mathcal{M}'$ is a model of $\Phi \wedge \overline{\mathcal{M}^1_{|X}} \wedge \cdots \wedge \overline{\mathcal{M}^n_{|X}} \wedge \overline{\mathcal{M}_{|X}}$, then it is trivial that $\mathcal{M} \not\approx_X \mathcal{M}'$.

- **Line 9**. Note that the set $min(\mathcal{L})$ corresponds to the greatest subset of $\mathcal{L}$ satisfying the following property: for all $\mathcal{M} \in min(\mathcal{L})$, there is no model in $\mathcal{L}$ which is strictly less preferred than $\mathcal{M}$. The assignment in this line allows us to have only models that are at least as preferred as an element of $min(\mathcal{L})$. Indeed, we do not need to consider the models that are less preferred than the elements of $min(\mathcal{L})$ because it is clear that they are not Top-k models. Note that all the elements of $min(\mathcal{L})$ are equivalent with respect to the equivalence relation $\approx$ induced by $\succeq$, since this preorder relation is total.

- **Line 10 – 21**. This while-loop is similar to that in Algorithm 1 (Top-k). We only remove the condition $|preferred(\mathcal{M}, \mathcal{L})| < k$ and replace $min\_top(k, \mathcal{L})$ with $min(\mathcal{L})$. In fact, since the preference relation $\succeq$ is a total preorder, it is obvious that we have $|preferred(\mathcal{M}, \mathcal{L})| < k$ because of the lower bounds added previously. Moreover, as $\succeq$ is total, the set of removed models by $remove(k, \mathcal{L})$ (Line 16) is either the empty set or $min(\mathcal{L})$.

Proposition 2. *Algorithm 2 (Top-k^T) is correct.*

Correctness of this algorithm is obtained from that of the algorithm Top-k and the fact that the considered δ-preference relation is total.

5 An application of Top-*k* SAT in data mining

The problem of mining frequent itemsets is well-known and essential in data mining [1], knowledge discovery and data analysis. Note that several data mining tasks are closely related to the itemset mining problem such as the ones of association rule mining, frequent pattern mining in sequence data, data clustering, etc. Recently, De Raedt et al. in [24, 25] proposed the first constraint programming (CP) based data mining framework for itemset mining. This new framework offers a declarative and flexible representation model. It allows data mining problems to benefit from several generic and efficient CP solving techniques. This first study leads to the first CP approach for itemset mining displaying nice declarative opportunities.

In itemset mining problem, the notion of Top-k frequent itemsets is introduced as an alternative to finding the appropriate value for the minimum support threshold. In this section, we propose a SAT-based encoding for enumerating all closed itemsets. Then we use this encoding in the Top-k SAT problem for computing all Top-k frequent closed itemsets.

Algorithm 2: Top-k^T

Input: a CNF formula Φ, a total preorder relation $\succeq$, an integer $k \geq 1$,and a set X of Boolean variables
Output: the set of all Top-k models $\mathcal{L}$

$\Phi' \leftarrow \Phi$;
$\mathcal{L} \leftarrow \emptyset$; /* Set of all Top-$k$ models */
for $(i \leftarrow 0$ **to** $k-1)$ **do**
  if $(\texttt{solve}(\Phi'))$ **then**
    $\texttt{add}(\mathcal{M}, \mathcal{L})$; /* $\mathcal{M}$ is a model of Φ' */
    $\Phi' \leftarrow \Phi' \wedge \overline{\mathcal{M}_{|X}}$;
  else
    return $\mathcal{L}$;
$\Phi' \leftarrow \Phi \wedge \bigwedge_{\mathcal{M} \in \mathcal{L}} \overline{\mathcal{M}} \wedge \bigwedge_{\mathcal{M}' \in min(\mathcal{L})} f_{\succeq}(\mathcal{M}')$;
while $(\text{solve}(\Phi'))$ **do** /* $\mathcal{M}$ is a model of Φ' */
  if $(\exists \mathcal{M}' \in \mathcal{L}. \mathcal{M} \approx_X \mathcal{M}' \ \& \ \mathcal{M} \succ \mathcal{M}')$ **then**
    $\texttt{replace}(\mathcal{M}, \mathcal{M}', \mathcal{L})$;
  else if $(\forall \mathcal{M}' \in \mathcal{L}. \mathcal{M} \not\approx_X \mathcal{M}')$ **then**
    $S \leftarrow \texttt{min}(\mathcal{L})$;
    $\texttt{add}(\mathcal{M}, \mathcal{L})$;
    $\texttt{remove}(k, \mathcal{L})$;
    $S \leftarrow \texttt{min}(\mathcal{L}) \setminus S$;
    $\Phi' \leftarrow \Phi' \wedge \bigwedge_{\mathcal{M}' \in S} f_{\succeq}(\mathcal{M}')$;
    else
      $\Phi' \leftarrow \Phi' \wedge f_{\succeq}(\mathcal{M})$
  $\Phi' \leftarrow \Phi' \wedge \overline{\mathcal{M}}$;
return $\mathcal{L}$;

5.1 Problem statement

Let $\mathcal{I}$ be a set of *items*. A *transaction* is a couple (tid, I) where tid is the *transaction identifier* and I is an *itemset*, i.e., $I \subseteq \mathcal{I}$. A *transaction database* is a finite set of transactions over $\mathcal{I}$ where, for all two different transactions, they do not have the same transaction identifier. We say that a transaction (tid, I) *supports* an itemset J if $J \subseteq I$.

The *cover* of an itemset I in a transaction database $\mathcal{D}$ is the set of transaction identifiers in $\mathcal{D}$ supporting I: $\mathcal{C}(I, \mathcal{D}) = \{tid \mid (tid, J) \in \mathcal{D}, I \subseteq J\}$. The *support* of an itemset I in $\mathcal{D}$ is defined by: $\mathcal{S}(I, \mathcal{D}) = \mid \mathcal{C}(I, \mathcal{D}) \mid$. Moreover, the frequency of I in $\mathcal{D}$ is defined by: $\mathcal{F}(I, \mathcal{D}) = \frac{\mathcal{S}(I, \mathcal{D})}{|\mathcal{D}|}$.

For instance, consider the following transaction database:

tid	itemset
1	a, b, c, d
2	a, b, e, f
3	a, b, c, m
4	a, c, d, f, j
5	j, l
6	d
7	d, j

Transaction database $\mathcal{D}$

In this database, we have $\mathcal{S}(\{a,b,c\},\mathcal{D}) = |\{1,3\}| = 2$ and $\mathcal{F}(\{a,b\},\mathcal{D}) = \frac{3}{7}$.

Let $\mathcal{D}$ be a transaction database over $\mathcal{I}$ and λ a minimum support threshold. The *frequent itemset mining problem* consists in computing the following set:

$$\mathcal{FIM}(\mathcal{D},\lambda) = \{I \subseteq \mathcal{I} \mid \mathcal{S}(I,\mathcal{D}) \geq \lambda\}$$

Definition 3 (Closed Itemset). *Let $\mathcal{D}$ be a transaction database (over $\mathcal{I}$) and I an itemset ($I \subseteq \mathcal{I}$) such that $\mathcal{S}(I,\mathcal{D}) \geq 1$. I is closed if for all itemset J such that $I \subset J$, $\mathcal{S}(J,\mathcal{D}) < \mathcal{S}(I,\mathcal{D})$.*

One can easily see that all frequent itemsets can be obtained from the closed frequent itemsets by computing their subsets. Since the number of closed frequent itemsets is smaller than or equal to the number of frequent itemsets, enumerating all closed itemsets allows us to reduce the size of the output.

In this work, we mainly consider the problem of mining Top-k frequent closed itemsets of minimum length min. In this problem, we consider that the minimum support threshold λ is not known.

Definition 4 ($\mathcal{FCIM}^k_{min}$). *Let k and min be strictly positive integers. The problem of mining Top-k frequent closed itemsets consists in computing all closed itemsets of length at least min such that, for each one, there exist no more than $k-1$ closed itemsets of length at least min with supports greater than its support.*

5.2 SAT-based encoding for $\mathcal{FCIM}^k_{min}$

We now propose a Boolean encoding of $\mathcal{FCIM}^k_{min}$. Let $\mathcal{I}$ be a set of items, $\mathcal{D} = \{(0,t_i),\ldots,(n-1,t_{n-1})\}$ a transaction database over $\mathcal{I}$, and k and min are strictly positive integers. We associate to each item a appearing in $\mathcal{D}$ a Boolean variable p_a. Such Boolean variables encode the candidate itemset $I \subseteq \mathcal{I}$, i.e., $p_a = true$ iff $a \in I$. Moreover, for all $i \in \{0,\ldots,n-1\}$, we associate to the i-th transaction a Boolean variable b_i.

We first propose a constraint allowing to consider only the itemsets of length at least min. It corresponds to a cardinality constraint:

$$\sum_{a \in \mathcal{I}} p_a \geq min \tag{1}$$

We now introduce a constraint allowing to capture all the transactions where the candidate itemset does not appear:

$$\bigwedge_{i=0}^{n-1} (b_i \leftrightarrow \bigvee_{a \in \mathcal{I} \setminus t_i} p_a) \tag{2}$$

This constraint means that b_i is true if and only if the candidate itemset is not in t_i.

By the following constraint, we force the candidate itemset to be closed:

$$\bigwedge_{a \in \mathcal{I}} (\bigwedge_{i=0}^{n-1} \overline{b_i} \rightarrow a \in t_i) \rightarrow p_a \tag{3}$$

Intuitively, this formula means that if $\mathcal{S}(I) = \mathcal{S}(I \cup \{a\})$ then $a \in I$ holds. Thus, it allows us to obtain models that correspond to closed itemsets.

In this context, computing the Top-k closed itemsets of length at least min corresponds to computing the Top-k models of (1), (2) and (3) with respect to $\succeq_B$ and $X = \{p_a | a \in \mathcal{I}\}$, where $B = \{b_0, \ldots, b_{n-1}\}$ and $\succeq_B$ is defined as follows: $\mathcal{M} \succeq_B \mathcal{M}'$ if and only if $|\mathcal{M}(B)| \leq |\mathcal{M}'(B)|$. This preorder relation is a δ-preference relation. Indeed, one can define $f_{\succeq_B}$ as follows:

$$f_{\succeq_B}(\mathcal{M}) = (\sum_{i=0}^{n-1} b_i \leq |\mathcal{M}(B)|)$$

Naturally, this formula allows us to have models corresponding to closed itemsets with supports greater or equal to the support of the closed itemset obtained from $\mathcal{M}$.

5.3 Some Variants of $\mathcal{FCIM}^k_{min}$

In this section, our goal is to illustrate the nice declarative aspects of our proposed framework. To this end, we simply consider slight variations of the problem, and show that their encodings can be obtained by simple modifications.

Variant 1 ($\mathcal{FCIM}^k_{max}$) In this variant, we consider the problem of mining Top-k closed itemsets of length at most max. Our encoding in this case is obtained by adding to (2) and (3) the following constraint:

$$\sum_{a \in \mathcal{I}} p_a \leq max \tag{4}$$

In this case, we use the δ-preference relation $\succeq_B$ defined previously.

Variant 2 ($\mathcal{FCIM}^k_\lambda$) Let us now propose an encoding of the problem of mining Top-k closed itemsets of supports at least λ (minimal support threshold). In this context, a Top-k closed itemset is a closed itemset such that, for each one, there exist no more than k - 1 closed itemsets of length greater than its length. Our encoding in this case is obtained by adding to (2) and (3) the following constraint:

$$\sum_{i=0}^{n} \overline{b_i} \geq \lambda \tag{5}$$

The preference relation used in this case is $\succeq_I$ defined as follows: $\mathcal{M} \succeq_I \mathcal{M}'$ if and only if $|\mathcal{M}(I)| \geq |\mathcal{M}'(I)|$. It is a δ-preference relation because $f_{\succeq_I}$ can be defined as follows:

$$f_{\succeq_I}(\mathcal{M}) = \sum_{a \in I} p_a \geq |\mathcal{M}(I)|$$

Variant 3 ($\mathcal{FMIM}^k_\lambda$) We consider here a variant of the problem of mining maximal frequent itemsets. It consists in enumerating Top-k maximal itemsets of supports at least λ and for each one, there exist no more than k - 1 maximal itemsets of length greater than its length. Our encoding of this problem consists of (2) and (5). We use in this case the δ-preference relation $\succeq_I$.

6 Experiments

This section evaluates the performance of our Algorithm for Top-k SAT empirically. The primary goal is to assess the declarativity and the effectiveness of our proposed framework. For this purpose, we consider the problem $\mathcal{FCIM}^k_{min}$ of computing the Top-k frequent closed itemsets of minimum length min described above.
For our experiments, we implemented the Algorithm 1 (Top-k) on the top of the state-of-the-art SAT solver MiniSAT 2.2 [1]. In our SAT encoding of $\mathcal{FCIM}^k_{min}$, we used the sorting networks, one of the state-of-the-art encoding of the cardinality constraint (0/1 linear inequality) to CNF proposed in [26].

We considered a variety of datasets taken from the `FIMI` repository [2] and `CP4IM` [3]. All the experiments were done on Intel Xeon quad-core machines with 32GB of RAM running at 2.66 Ghz. For each instance, we used a timeout of 4 hours of CPU time.
The table 1 details the characteristics of the different transaction databases ($\mathcal{D}$). The first column mentions the name of the considered instance. In the second and third column, we give the size of $\mathcal{D}$ in terms of number of transactions (#trans) and number of items (#items) respectively. The fourth column shows the density (dens) of the transaction database, defined as the percentage of 1's in $\mathcal{D}$. The panel of datasets ranges from sparse (e.g. mushroom) to dense ones (e.g. Hepatitis). Finally, in the two last columns, we give the size of the CNF encoding (#vars, #clauses) of $\mathcal{FCIM}^k_{min}$. As we can see, our proposed encoding leads to CNF formula of reasonable size. The maximum size is obtained for the instance *connect* (67 815 variables and 5 877 720 clauses).

In order to analyze the behavior of our Top-k algorithm on $\mathcal{FCIM}^k_{min}$, we conducted two kind of experiments. In the first one, we set the minimum length min of the itemsets to 1, while the value of k is varied from 1 to 10000. In the second experiment, we fix the parameter k to 10, and we vary the minimal length min from 1 to the maximum size of the transactions.

Results for a representative set of datasets are shown in Figure 1 (log scale). The other instances present similar behavior. As expected, the CPU time needed for computing the Top-k models increase with k. For the *connect* dataset, our algorithm fails to

[1] MiniSAT: http://minisat.se/
[2] FIMI: http://fimi.ua.ac.be/data/
[3] CP4IM: http://dtai.cs.kuleuven.be/CP4IM/datasets/

instance	#trans	#items	$dens(\%)$	#vars	#clauses
zoo-1	101	36	44	173	2196
Hepatitis	137	68	50	273	4934
Lymph	148	68	40	284	6355
audiology	216	148	45	508	17575
Heart-cleveland	296	95	47	486	15289
Primary-tumor	336	31	48	398	5777
Vote	435	48	33	531	14454
Soybean	650	50	32	730	22153
Australian-credit	653	125	41	901	48573
Anneal	812	93	45	990	39157
Tic-tac-toe	958	27	33	1012	18259
german-credit	1000	112	34	1220	73223
Kr-vs-kp	3196	73	49	3342	121597
Hypothyroid	3247	88	49	3419	143043
chess	3196	75	49	3346	124797
splice-1	3190	287	21	3764	727897
mushroom	8124	119	18	8348	747635
connect	67558	129	33	67815	5877720

Table 1. Characteristics of the datasets

compute the Top-k models for higher value of $k > 1000$ in the time limit of 4 hours. This figure clearly shows that finding the Top-k models (the most interesting ones) can be computed efficiently for small values of k. For example, on all datasets the top-10 models are computed in less than 100 seconds of CPU time. When a given instance contains a huge number of frequent closed itemsests, the Top-k problem offers an alternative to the user to control the size of the output and to get the most preferred models. In Figure 2, we show the results obtained on the hardest instance from Table 1. On splice-1, the algorithm fails to solve the problem under the time limit for $k > 20$.

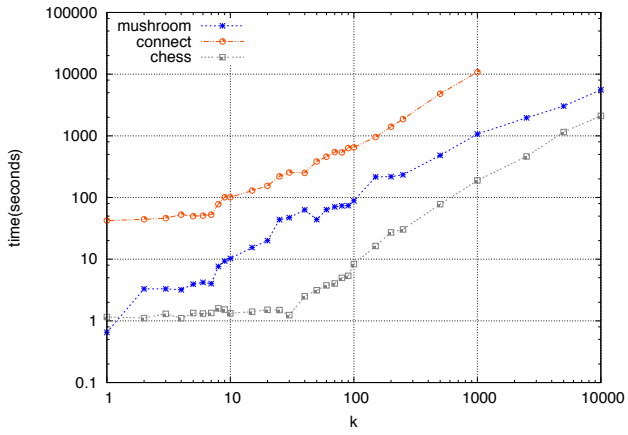


Fig. 1. $\mathcal{FCIM}_1^k$ results for different values of k

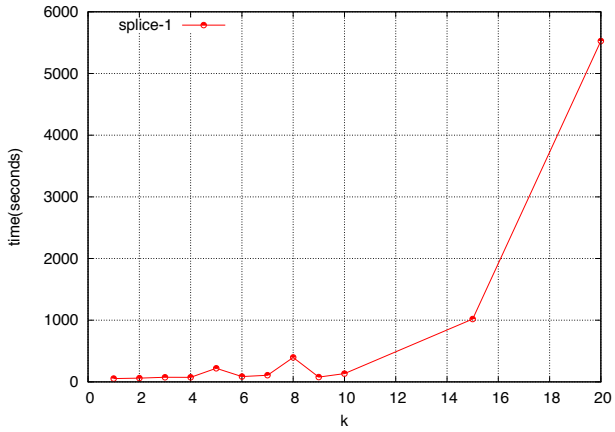


Fig. 2. Hardest $\mathcal{FCIM}_1^k$ instance

In our second experiment, our goal is to show the behavior of our algorithm when varying the minimum length. In Figure 3, we give the results obtained on the three representative datasets (mushroom, connect and chess) when k is fixed to 10 and min is varied from 1 to the maximum size of the transactions. The problem is easy at both the under-constrained (small values of min - many Top-k models) and the over-constrained (high values of min - small number of Top-k models) regions. For the connect dataset, the algorithm fails to solve the problem for $min > 15$ under the time limit. For all the other datasets, the different curves present a pick of difficulty for medium values of the minimal length of the itemsets.

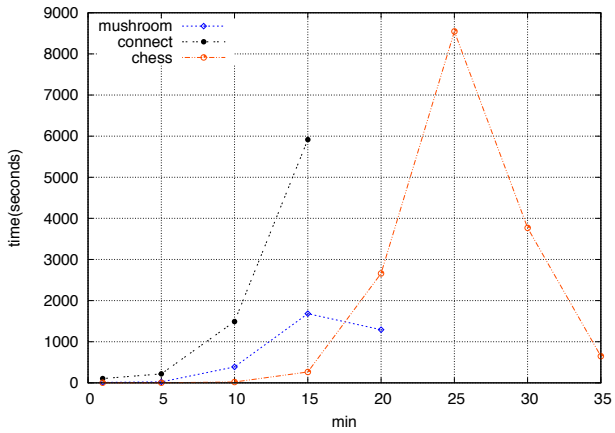


Fig. 3. $\mathcal{FCIM}_{min}^{10}$ results for different values of min

7 Acknowledgments

This work was supported by the French Research Agency (ANR) under the project DAG "Declarative Approaches for Enumerating Interesting Patterns" - Défis program 2009.

8 Conclusion and Perspectives

In this paper, we introduce a new problem, called Top-k SAT, defined as the problem of enumerating the Top-k models of a propositional formula. A Top-k model is a model having no more than k-1 models preferred to it with respect to the considered preference relation. We also show that Top-k SAT generalizes the two well-known problems: the partial Max-SAT problem and the problem of computing minimal models. A general algorithm for this problem is proposed and evaluated on the problem of enumerating top-k frequent closed itemsets of length at least min.

While our new problem of computing the Top-k preferred models in Boolean satisfiability is flexible and declarative, there are a number of questions that deserve further research efforts. One direction is the study of (preferred/Top-k) model enumeration algorithm so as to achieve a further speedup of the runtime. This fundamental problem has not received a lot of attention in the SAT community, except some interesting works on enumerating minimal/preferred models.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: ACM SIGMOD International Conference on Management of Data, Baltimore, ACM Press (1993) 207–216
2. Tiwari, A., Gupta, R., Agrawal, D.: A survey on frequent pattern mining: Current status and challenging issues. Inform. Technol. J **9** (2010) 1278–1293
3. Fu, A.W.C., w. Kwong, R.W., Tang, J.: Mining *n*-most interesting itemsets. In: Proceedings of the 12th International Symposium on Mthodologies for Intelligent Systems (ISMIS 2000). Lecture Notes in Computer Science, Springer (2000) 59–67
4. Han, J., Wang, J., Lu, Y., Tzvetkov, P.: Mining top-k frequent closed patterns without minimum support. In: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), IEEE Computer Society (2002) 211–218
5. Ke, Y., Cheng, J., Yu, J.X.: Top-k correlative graph mining. In: Proceedings of the SIAM International Conference on Data Mining (SDM 2009). (2009) 1038–1049
6. Valari, E., Kontaki, M., Papadopoulos, A.N.: Discovery of top-k dense subgraphs in dynamic graph collections. In: Proceedings of the 24th International Conference on Scientific and Statistical Database Management (SSDBM 2012). (2012) 213–230
7. Lam, H.T., Calders, T.: Mining top-k frequent items in a data stream with flexible sliding windows. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2010). (2010) 283–292
8. Lam, H.T., Calders, T., Pham, N.: Online discovery of top-k similar motifs in time series data. In: Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011 (SDM 2011). (2011) 1004–1015

9. Shoham, Y.: Reasoning about change: time and causation from the standpoint of artificial intelligence. MIT Press, Cambridge, MA, USA (1988)
10. Meseguer, P., Rossi, F., Schiex, T.: 9. In: Soft Constraints. Elsevier (2006)
11. Boutilier, C., Brafman, R.I., Domshlak, C., Poole, D.L., Hoos, H.H.: CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. Journal of Artificial Intelligence Research (JAIR) **21** (2004) 135–191
12. Walsh, T.: Representing and reasoning with preferences. AI Magazine **28**(4) (2007) 59–70
13. Brafman, R.I., Domshlak, C.: Preference Handling - An Introductory Tutorial. AI Magazine **30**(1) (2009) 58–86
14. Domshlak, C., Hüllermeier, E., Kaci, S., Prade, H.: Preferences in AI: An overview. Artificial Intelligence **175**(7-8) (2011) 1037–1052
15. Rosa, E.D., Giunchiglia, E., Maratea, M.: Solving satisfiability problems with preferences. Constraints **15**(4) (2010) 485–515
16. Castell, T., Cayrol, C., Cayrol, M., Berre, D.L.: Using the davis and putnam procedure for an efficient computation of preferred models. In: ECAI. (1996) 350–354
17. Wang, J., Han, J., Lu, Y., Tzvetkov, P.: TFP: An efficient algorithm for mining top-k frequent closed itemsets. IEEE Transactions on Knowledge Data Engineering **17**(5) (2005) 652–664
18. Tseitin, G.: On the complexity of derivations in the propositional calculus. In: Structures in Constructives Mathematics and Mathematical Logic, Part II. (1968) 115–125
19. Fu, Z., Malik, S.: On Solving the Partial MAX-SAT Problem. In: Proceedings of the Ninth International Conference on Theory and Applications of Satisfiability Testing (SAT'06). (2006) 252–265
20. Warners, J.P.: A linear-time transformation of linear inequalities into conjunctive normal form. Information Processing Letters (1996)
21. Bailleux, O., Boufkhad, Y.: Efficient CNF Encoding of Boolean Cardinality Constraints. In: 9th International Conference on Principles and Practice of Constraint Programming - CP 2003. (2003) 108–122
22. Sinz, C.: Towards an optimal cnf encoding of boolean cardinality constraints. In: 11th International Conference on Principles and Practice of Constraint Programming - CP 2005. (2005) 827–831
23. Cadoli, M.: On the complexity of model finding for nonmonotonic propositional logics. In: 4th Italian conference on theoretical computer science. (1992) 125–139
24. Raedt, L.D., Guns, T., Nijssen, S.: Constraint programming for itemset mining. In: ACM SIGKDD. (2008) 204–212
25. Guns, T., Nijssen, S., De Raedt, L.: Itemset mining: A constraint programming perspective. Artificial Intelligence **175**(12-13) (August 2011) 1951–1983
26. Eén, N., Sörensson, N.: Translating pseudo-boolean constraints into SAT. JSAT **2**(1-4) (2006) 1–26

Inconsistency Measurement Thanks to MUS Decomposition

Said Jabbour
CRIL-CNRS, Univ. Artois
Lens, France
jabbour@cril.fr

Yue Ma
Institute of TCS
TU Dresden, Germany
mayue@tu-dresden.de

Badran Raddaoui
CRIL-CNRS, Univ. Artois
Lens, France
raddaoui@cril.fr

ABSTRACT

Bearing contradictory knowledge is often unavoidable among multi-agents. Measuring inconsistency degrees of knowledge bases of different agents facilitates the understanding of an agent to her environment. Several semantics or syntax-based approaches have been proposed to quantify inconsistencies. In this paper, we propose a new inconsistency measuring framework based on both minimal unsatisfiable sets and maximal consistent sets. Firstly, we define a graph representation of knowledge bases, based on which we furthermore explore the logical property of the *Additivity* condition. Then, we show how the structure of the proposed graph representation can be used to discriminate, in a fine-grained way, the responsibility of each formula or a set of formulae for the inconsistency of a knowledge base. Finally, we extend our framework to provide an inconsistency measure for a whole knowledge base. All the proposed measures are shown satisfying the desired properties.

Categories and Subject Descriptors

H.3 [**Agent Reasoning**]: Knowledge representation

General Terms

Theory

Keywords

Measuring Inconsistency, Classical Logic

1. INTRODUCTION

Measuring inconsistency has proven useful and attractive in diverse scenarios including software specifications [21], e-commerce protocols [5], belief merging [28], news reports [13], integrity constraints [8], requirements engineering [21], databases [22, 11], ontologies [33], semantic web [33], and network intrusion detection [23].

Indeed, we cannot expect large-sized knowledge bases free of inconsistency in real applications, such as multi-agents communicating with each other to build a common knowledge base or to perform certain actions in a complex environment. For illustration purpose, we consider an example simplified from the Multi-agent System ARTOR (ARTificial ORganizations [30]) which simulates the partnership of organizations – each organization owns agents responsible for purchasing and selling products. We consider a scenario where each product has N grades of price and M grades of quantity (higher grade, larger value, e.g. \$10 for the price grade 1 and \$20 for the grade 2). Selling agents wish to sell products at a price as expensive as possible and with a quantity as large as possible, but buying agents wish as cheap as possible and have quantity upper bounds. By $p_i (1 \leq i \leq N)$ and $q_j (1 \leq j \leq M)$, we denote a trade of price grade i and of quantity grade j, respectively. Consider the following two agents:

Buying agent :		Selling agent :	
$p_{i+1} \rightarrow p_i$	|	$p_i \rightarrow p_{i+1}$	(1)
$q_{j+1} \rightarrow q_j$	|	$q_j \rightarrow p_{j+1}$	(2)
$p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4$	|	$\neg p_5 \wedge p_6 \wedge \cdots \wedge p_N$	(3)
$q_1 \wedge q_2 \wedge \neg q_3$	|	$\neg q_6 \wedge q_7 \wedge \cdots \wedge q_M$	(4)

Item 1 (resp. 2) says that if the buying agent accepts the price (resp. quantity) at grade $i+1$ (resp. $j+1$), then he accepts lower prices (resp. less quantities); Whilst the selling agent accepts the price (resp. quantity) at grade i (resp. j), then he accepts higher prices (resp. larger quantities). Together with Item 3 (resp. 4), we can read that the buying agent agrees to pay at most at price grade 3 and with the maximal quantity grade 2, but the selling agent accepts minimal price grade 6 and quantity grade 7. It is easy to see that there are contradictions between these two agents (e.g. $\{p_3, \neg q_7\}$ for the buying agent but $\{\neg p_3, q_7\}$ for the selling agent). To make the trade successful, agents should be equipped with the ability of reasoning with inconsistency. Moreover, if an agent knows the different inconsistency degrees between him and others, a better trade plan could be arranged when multiple sellers and buyers are available.

Analyzing conflicting information has gained a considerable attention and become an important issue in computer science recently [2]. Indeed, measuring inconsistency is helpful to compare different knowledge bases and to evaluate their quality [7]. Consider again the above example, giving the opportunity for a buyer agent to choose among different selling agents, naturally he may try to choose the one that is less inconsistent.

To understand the nature of inconsistency and to quantify it in turn, a number of logic-based inconsistency measures have been studied, including the maximal η-consistency [18], measures based on variables or via multi-valued models [7,

Appears in: *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), Lomuscio, Scerri, Bazzan, Huhns (eds.), May, 5–9, 2014, Paris, France.*

12, 27, 13, 9, 20, 31, 19], n-consistency and n-probability [6], minimal inconsistent subsets based inconsistency measures [15, 24, 25, 32], the Shapley inconsistency value [14, 16], and the inconsistency measurement based on minimal proofs [17]. Although it is hardly possible to have a complete comparison of the proposed measures. One way to categorize the existing measures can be based on the dependence of syntax or semantics: Semantics based ones aim to compute the proportion of the language affected by inconsistency. The inconsistency values belonging to this class are often based on some paraconsistent semantics and syntax independent. Whilst, syntax based ones are concerned with the minimal number of formulae that cause inconsistencies. Viewing Minimal Unsatisfiable Subsets (MUSes) as the cornerstones of inconsistency, it is natural to derive inconsistency measures from MUSes of a knowledge base. Another possible classification of different measures originates in the measuring objective: formula oriented or knowledge base oriented. For example, the inconsistency measures in [14, 16] are to measure the contribution of a formula to the inconsistency of a whole knowledge base that it belongs to, while the rest measures mentioned above are to measure the inconsistency degree of a whole base. Some basic properties [16], such as *Consistency, Monotony, Free Formula Independence, and MinInc*, are also proposed to qualify inconsistency measures.

In this paper, we propose a new approach for measuring inconsistency, both formula oriented and knowledge base oriented. It is inspired, on the one hand, by the observation that existing measures fail to distinguish certain knowledge bases which should have different inconsistency degrees. On the other hand, a specific property is explored, namely *Additivity* that is rarely discussed in the literature due to its modeling difficulty [16]. Our study is based on a novel analysis of connections among MUSes, which is shown useful and general to quantify more finely the inconsistency responsibility of a formula and the inconsistency degree of a whole base. Note that this is unlike the oriented links between MUSes explored in [1] which says that the resolution of one MUS allows for the resolution of the other. By looking inside MUSes and taking into account the correlations between them, the proposed analysis of MUSes structure lead to several interesting measures different from the existing ones. Moreover, we propose an *enhanced additivity* property and show that our measures satisfy such a property as well as several basic properties.

The paper is organized as follows: Section 2 reviews approaches to measuring inconsistencies based on minimal inconsistent subsets and maximal consistent subsets. In Section 3, we revisit the additivity property and propose the graph representation of a knowledge base by which we introduce the notion of MUS-decomposition. This notion is then used in Section 4 to evaluate the degree of inconsistency of each formula in the knowledge base. In Section 5, we generalize our framework to quantify inconsistency of a whole base. Section 6 concludes the paper and discusses some perspectives.

2. PRELIMINARIES

Through this paper, we consider the propositional language $\mathcal{L}$ built from a finite set of propositional symbols $\mathcal{P}$ under connectives $\{\neg, \wedge, \vee, \rightarrow\}$. We will use $a, b, c, \ldots$ to denote propositional variables, and Greek letters $\alpha, \beta, \gamma, \ldots$ for formulae. The symbol $\bot$ denotes contradiction. For a set S, we denote $|S|$ its cardinality.

A knowledge base K consists of a finite set of propositional formulae. K is inconsistent if there is a formula α such that $K \vdash \alpha$, and $K \vdash \neg\alpha$, where $\vdash$ is the deduction in classical propositional logic. If K is inconsistent, then one can define the notion of *Minimal Unsatisfiable Subset* (MUS) as an unsatisfiable set of formulae $\mathcal{M}$ in K such that any of its subsets is satisfiable. Formally,

Definition 1 (MUS). *Let K be a knowledge base and $\mathcal{M} \subseteq K$. $\mathcal{M}$ is a minimal unsatisfiable (inconsistent) subset of K iff $\mathcal{M} \vdash \bot$ and $\forall \mathcal{M}' \subset \mathcal{M}$, $\mathcal{M}' \nvdash \bot$.*

Clearly, an inconsistent knowledge base K can have multiple minimal inconsistent subsets. The set of minimal inconsistent subsets of K is denoted as $MUSes(K) = \{\mathcal{M} \subseteq K \mid \mathcal{M} \text{ is a } MUS\}$. When a MUS is singleton, the single formula in it is called a *self-contradictory formula*. A formula α not involved in any MUS of K is called a *free formula*, meaning that α does not have any relationship with the inconsistency of K. Formally, $SelfC(K) = \{\alpha \in K \mid \{\alpha\} \vdash \bot\}$, and $free(K) = \{\alpha \mid \text{ for all } \mathcal{M} \in MUSes(K), \alpha \notin \mathcal{M}\}$.

At the same time, we can define the *Maximal Satisfiable Subset* (MSS), and Hitting set as follows:

Definition 2 (MSS). *Let K be a knowledge base and $\mathcal{M}$ a subset of K. $\mathcal{M}$ is a maximal satisfiable (consistent) subset of K iff $\mathcal{M} \nvdash \bot$ and $\forall \alpha \in K \setminus \mathcal{M}$, $\mathcal{M} \cup \{\alpha\} \vdash \bot$.*

We denote by $MSSes(K)$ the set of all maximal consistent subsets of K.

Definition 3 (Hitting set). *Given a universe U of elements and a collection $\mathcal{S}$ of subsets of U, $H \subseteq U$ is a hitting set of $\mathcal{S}$ if $\forall E \in \mathcal{S}, H \cap E \neq \emptyset$. We say that H is a minimal hitting set of $\mathcal{S}$ if H is a hitting set of $\mathcal{S}$ and each $H' \subset H$ is not a hitting set of $\mathcal{S}$.*

We denote by $LB_{HS}(K)$ the smallest size of a hitting set of K, i.e., $LB_{HS}(K) = min\{|H| \mid \text{H is a hitting set of } K\}$.

2.1 Inconsistency Measures

In this section, we review some inconsistency measures important and related to the rest of this paper.

There have been several contributions for measuring inconsistency in knowledge bases defined through minimal inconsistent subsets theories. In [16], Hunter and Konieczny introduce a scoring function allowing to measure the degree of inconsistency of a subset of a knowledge base. In details, for a subset $K' \subseteq K$, the scoring function is defined as the diminution of the number of minimal inconsistent subsets while K' is removed, i.e., $|MUSes(K)| - |MUSes(K \setminus K')|$. By extending the scoring function, the authors introduce an inconsistency measure of a whole base, defined as the number of minimal inconsistent subsets of K. Formally, $I_{MI}(K) = |MUSes(K)|$. In the same paper, a family of "MinInc inconsistency values" MIV based on minimal inconsistent subsets is also presented:

- $MIV_D(K, \alpha)$ is a simple measure that values 1 if α belongs to a minimal inconsistent subset and 0 otherwise.
- $MIV_\#$ is defined by the scoring function: $MIV_\#(K, \alpha) = |\{\mathcal{M} \in MUSes(K) \mid \alpha \in \mathcal{M}\}|$.

- MIV_C takes into account the size of each minimal inconsistent subset in addition to the number of $MUSes$ of K, formally $MIV_C(K,\alpha) = \sum_{\alpha\in\mathcal{M}|\mathcal{M}\in MUSes(K)} \frac{1}{|\mathcal{M}|}$.

Combining both minimal inconsistent subsets and maximal consistent subsets, Mu et al. present in [26] an approach to quantify the degree of inconsistency of a knowledge base. Another inconsistency value, called I_M measure, that deals with that combination, has been introduced in [10]. The I_M measure counts for a given knowledge base, the number of its $MSSes$ and its Self-contradictory formulae (subtraction of 1 is required to make $I_M(K) = 0$ when K is consistent):

$$I_M(K) = |MSSes(K)| + |SelfC(K)| - 1.$$

Recently, in [17] a new framework based on the notion of minimal proof is proposed to measure inconsistency degrees. This framework uses conflicting minimal proofs to characterize the inconsistency of any subset of formulae in a base.

3. MUS PARTITIONING IN KNOWLEDGE BASES

There are a set of well accepted basic properties that inconsistency measures should satisfy (see Definition 4), while leaving one property *Additivity* debatable [15]. In this section, we propose an enhancement of the additivity property to make it more intuitive and give a way to modify the I_M measure introduced above to satisfy the enhance additivity.

DEFINITION 4 ([16]). *Let K and K' be two knowledge bases, α and β two formulae. A set of properties of an inconsistency measure I is defined as follows:*

(1) Consistency: $I(K) = 0$ iff K is consistent,

(2) Monotony: $I(K) \leq I(K \cup K')$,

(3) Free Formula Independence: if α is a free formula in $K \cup \{\alpha\}$, then $I(K \cup \{\alpha\}) = I(K)$,

(4) MinInc: If $M \in MUSes(K)$, then $I(M) = 1$.

The monotony property shows that the inconsistency value of a knowledge base has to be non-decreasing while adding new formulae. The free formula independence property states that the set of formulae not involved in any minimal inconsistent subset should have no contribution to inconsistency. And MinInc means that a single MUS as a whole has inconsistency value 1. Besides, another property called *Additivity*[1] has been proposed in [16].

DEFINITION 5 (ADDITIVITY). *Let $K_1, \ldots, K_n$ be knowledge bases and I an inconsistency measure. I is additive if it satisfies the following condition: If $MUSes(K_1 \cup \ldots \cup K_n) = MUSes(K_1) \oplus \ldots \oplus MUSes(K_n)$[2], then $I(K_1 \cup \ldots \cup K_n) = I(K_1) + \ldots + I(K_n)$.*

The additivity was proposed to ensure that the amount of conflict of a knowledge base K can be obtained by summing up the degrees of its sub-bases K_i under the condition that $\{MUSes(K_i)\}$ is a partition of $MUSes(K)$. However, Luce and Raiffa have pointed out that the interaction of sub-bases (sub-games in [29]) is not taken into account by Additivity, which is one of the criticisms about this condition [29, 16]. Although the partitionability of $MUSes$ is used to describe a sort of interaction in Definition 5, we argue that it is not enough. Consider the following example:

EXAMPLE 1. *Let $K_1 = \{a, \neg a\}, K_2 = \{\neg a, a \wedge b\}$, and $K_3 = \{c, \neg c\}$, each of which has only one single MUS. Consider two bases $K = K_1 \cup K_2$, and $K' = K_1 \cup K_3$. Clearly, $MUSes(K) = MUSes(K_1) \oplus MUSes(K_2)$, and $MUSes(K') = MUSes(K_1) \oplus MUSes(K_3)$. For any measure I, if I satisfies Additivity by Definition 5, we should have $I(K) = I(K_1)+I(K_2)$, and $I(K') = I(K_1)+I(K_3)$. If I furthermore satisifes MinInc (see Definition 4), single MUS leads to the same inconsistency value. Then $I(K) = I(K')$, which is counterintuitive because the components of $MUSes(K')$ are less interactive, thus more spreading, than that of $MUSes(K)$. And in turn K' should, intuitively, contains more inconsistencies than K.*

To enhance the consideration of interaction among sub-bases, we propose the following Enhanced Additivity:

DEFINITION 6 (ENHANCED ADDITIVITY). *Let $K_1, \ldots, K_n$ be knowledge bases and I an inconsistency measure. If $MUSes(K_1 \cup \ldots \cup K_n) = MUSes(K_1) \oplus \ldots \oplus MUSes(K_n)$ and $\{\alpha \in \mathcal{M} \mid \mathcal{M} \in MUSes(K_i)\} \cap \{\beta \in \mathcal{M} \mid \mathcal{M} \in MUSes(K_j)\} = \emptyset$ for all $1 \leq i \neq j \leq n$, $I(K_1 \cup \ldots \cup K_n) = I(K_1)+\ldots+I(K_n)$. I is then called an independent-additive measure.*

Note that the enhanced additivity requires an extra precondition, which is to encode a stronger independence among sub-bases to perform additivity. The enhanced additivity can exclude the counterintuitive conclusion as given in Example 1: suppose I satisfies the enhanced additivity, then we have $I(K') = I(K_1)+I(K_3)$, but not necessarily $I(K') = I(K_1)+I(K_2)$. Hence $I(K)$ is not necessarily equal to $I(K')$.

Alternatively, the enhanced-additivity, given below, can be defined through multi-set partition instead of set partition as used in Definition 6.

DEFINITION 7. *I is called multi-set additive if it satisfies: $I(K_1 \cup \ldots \cup K_n) = \sum_{i=1}^{k} I(K_i)$ when $MUSes(\bigcup_{i=1}^{n} K_i) = \biguplus_{i=1}^{n} MUSes(K_i)$, where $\biguplus$ is the multi-set union over sets.*

For Example 1, whilst $MUSes(K) = \{\{a, \neg a\}, \{\neg a, a \wedge b\}\}$, we have $MUSes(K_1) \biguplus MUSes(K_2) = \{\{a, \neg a\}, \{\neg a, a \wedge b\}, \{a, \neg a\}\}$. This leads to the conclusion that it is unnecessarily to have $I(K) = I(K_1) + I(K_2)$, the same as using Definition 6. In fact, it is easy to see that I is multi-set additive if and only if it is enhanced additive.

Clearly, additivity implies enhanced additivity. While we can see that the MIV measure family satisfies the additivity and the enhanced additivity, it is not the case for the I_M measure as showed below.

PROPOSITION 1. *I_M is neither additive nor enhanced additive.*

PROOF. Consider the counter-example: $K_1 = \{a, \neg a\}$, $K_2 = \{b, \neg b\}$, and $K = K_1 \cup K_2$. It is easy to check that K and K_i $(i = 1, 2)$ satisfy the conditions of additivity and enhanced additivity. But we have $I_M(K_1 \cup K_2) = 3$ while $I_M(K_1) = I_M(K_2) = 1$. □

[1]The Additivity condition is named Decomposability in [15].
[2]We denote a partition $\{A, B\}$ of a set C by $C = A \oplus B$, i.e., $C = A \cup B$ and $A \cap B = \emptyset$.

Indeed, the following theorem states that under certain constraints, $MSSes$ is multiplicative instead of additive.

PROPOSITION 2. *Let K_1 and K_2 be two knowledge bases such that $\{\alpha \in MUSes(K_i)\} \cap \{\beta \in MUSes(K_j)\} = \emptyset$. Then, $|MSSes(K_1 \cup K_2)| = |MSSes(K_1)| \times |MSSes(K_2)|$.*

As the enhanced additivity gives a more intuitive characterization of interaction among sub-sets to be added up, in the following, we are interested in restoring the enhanced additivity of the I_M measure. To simplify terminology, in the rest of the paper, we call the enhanced additivity simply Additivity by default unless other claims are made.

To reach this goal, let us first define three fundamental concepts: *MUS-graph*, *MUS-decomposition*, and *elementary MSS*.

DEFINITION 8 (MUS-GRAPH). *Given an inconsistent knowledge base K, the MUS-graph of K, denoted $\mathcal{G}_{\mathcal{MUS}}(K)$, is an undirected graph where:*

(1) $MUSes(K)$ is the set of vertices, and

(2) $\forall \mathcal{M}, \mathcal{M}' \in MUSes(K)$, $\{\mathcal{M}, \mathcal{M}'\}$ is an edge iff $\mathcal{M} \cap \mathcal{M}' \neq \emptyset$.

A MUS-graph of a knowledge base K gives us a structural representation of its interconnected minimal unsatisfiable subsets.

Moreover, $\mathcal{G}_{\mathcal{MUS}}(K)$ leads to a partition of a knowledge base K, named *MUS-decomposition*, as defined below.

DEFINITION 9 (MUS-DECOMPOSITION). *Let K be a knowledge base, and $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ with $\mathcal{C}_i \subseteq K$ for $1 \leq i \leq p$. $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ is the MUS-decomposition of K iff $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ is the set of the connected components of $\mathcal{G}_{\mathcal{MUS}}(K)$.*

A MUS-decomposition $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ of a knowledge base K represents a partition of the minimal unsatisfiable subsets of K into connected components.

By the fact that $MUSes(K) \neq \emptyset$ and the uniqueness of connected components of a graph, we can easily see:

PROPOSITION 3. *MUS-decomposition exists and is unique for an inconsistency knowledge base.*

Definition 9 allows us to associate to a given knowledge base K a set of sub-bases $K_1, \ldots, K_p$ such that the elements of each sub-base K_i are the formulae of the connected component $\mathcal{C}_i$.

In the following, we present an alternative to the inconsistency measure I_M so as to make it additive. To this end, we introduce the concept of *elementary MSS* by using the MUS-decomposition.

DEFINITION 10. *Let K be a knowledge base, $K' \subseteq K$, and $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ the MUS-decomposition of K. K' is an elementary MSS of K iff there exists a connected component $\mathcal{C}_i$ of K such that $K' \in MSSes(\mathcal{C}_i)$. We denote by $EMSS(K)$ the set of all elementary MSS of K, i.e., $EMSS(K) = \bigcup_{i=1}^{p} MSSes(K_i)$.*

That is, an elementary maximal consistent subset should be locally restricted by a connected component of $MUSes(K)$.

EXAMPLE 2. [3] *Let $K = \{a \wedge d, \neg a, \neg b, b \vee \neg c, \neg c \wedge d, \neg c \vee e, c, \neg e, e \wedge d\}$. The MUS-decomposition of K contains two subsets, $\mathcal{C}_1 = \{\mathcal{M}_1\}$, and $\mathcal{C}_2 = \{\mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_5\}$ where $\mathcal{M}_1 = \{\neg a, a \wedge d\}$, $\mathcal{M}_2 = \{c, \neg b, b \vee \neg c\}$, $\mathcal{M}_3 = \{c, \neg c \wedge d\}$, $\mathcal{M}_4 = \{\neg c \vee e, c, \neg e\}$, and $\mathcal{M}_5 = \{\neg e, e \wedge d\}$ (see figure 1). Then, $EMSS(K) = \{\{a \wedge d\}, \{\neg a\}, \{\neg b, b \vee \neg c, \neg c \wedge d, \neg c \vee e, e \wedge d\}, \{\neg b, b \vee \neg c, \neg c \wedge d, \neg c \vee e, \neg e\}, \{b \vee \neg c, \neg c \vee e, c, e \wedge d\}, \{\neg b, \neg c \vee e, c, e \wedge d\}, \{b \vee \neg c, c, \neg e\}, \{\neg b, c, \neg e\}\}$.*

Now, we use the notion of *EMSS* to define an alternative of the I_M measure, named I'_M:

$$I'_M(K) = \begin{cases} |EMSS(K)| + |SelfC(K)| & \text{if } K \vdash \bot; \\ 0 & \text{otherwise.} \end{cases}$$

PROPOSITION 4. *the I'_M measure is additive.*

PROOF. The conclusion follows immediately from the fact that both $|EMSS(K)|$ and $|SelfC(K)|$ are additive. □

That is, by taking into account the connections between minimal inconsistent subsets, MUS-decomposition gives us a way to define an inconsistency measure which still satisfies the additivity.

4. MUS-DECOMPOSITION BASED INCONSISTENCY MEASURE

In this section, we use the MUS-decomposition of a knowledge base, defined in the previous section, to estimate the responsibility of each formula to the inconsistency of its base.

Given a MUS-graph of a knowledge base K, a distance, as defined next, is an assignment of a real number to each MUS pair of K.

DEFINITION 11. *Let K be a knowledge base. We denote $d_{MUS}(\mathcal{M}, \mathcal{M}')$ the shortest path between $\mathcal{M}$ and $\mathcal{M}'$ in the MUS-graph $\mathcal{G}_{\mathcal{MUS}}(K)$.*

As easily seen, $d_{MUS}(\mathcal{M}, \mathcal{M}')$ is a distance, that is, it satisfies: (1) $d_{MUS}(\mathcal{M}, \mathcal{M}') \geq 0$; (2) $d_{MUS}(\mathcal{M}, \mathcal{M}') = d_{MUS}(\mathcal{M}', \mathcal{M})$; and (3) $d_{MUS}(\mathcal{M}_1, \mathcal{M}_2) \leq d_{MUS}(\mathcal{M}_1, \mathcal{M}_3) + d_{MUS}(\mathcal{M}_3, \mathcal{M}_2)$. So, we call it the distance between $\mathcal{M}$ and $\mathcal{M}'$.

Next, we will extend Definition 11 to compute the distance between a formula and a MUS.

DEFINITION 12. *Let K be a knowledge base, $\alpha \in K \setminus free(K)$, and $\mathcal{M} \in MUSes(K)$. The distance between α and $\mathcal{M}$ is defined as $d_{MUS}(\alpha, \mathcal{M}) = min\{d_{MUS}(\mathcal{M}, \mathcal{M}') \mid \alpha \in \mathcal{M}'\}$.*

In fact, the distance between a given formula $\alpha \in K$ and a MUS $\mathcal{M}$ corresponds to the shortest path from α to $\mathcal{M}$ along a sequence of intersecting *MUSes*. Note that if α and $\mathcal{M}$ do not belong to the same connected component of $\mathcal{G}_{\mathcal{MUS}}(K)$, this means that $\mathcal{M}$ is not reachable from α and in this case, the distance is assigned an infinite value, i.e., $d_{MUS}(\alpha, \mathcal{M}) = +\infty$.

EXAMPLE 3. *(Example 2 Contd.) Let $\alpha = \neg b$, we have*
$d_{MUS}(\alpha, \mathcal{M}_2) = 0$ $\quad$ $d_{MUS}(\alpha, \mathcal{M}_3) = 1$
$d_{MUS}(\alpha, \mathcal{M}_4) = 1$ $\quad$ $d_{MUS}(\alpha, \mathcal{M}_5) = 2$

[3] For a better display of the new notions introduced in this paper, we use a different example from the illustrative one given in Section 1.

We note that the distance d_{MUS} allows for an ordering over the minimal inconsistent subsets of K according to their distances from α.

In the following, we quantify the inconsistency value of α in the light of the distances from α to reachable *MUSes* of K. Indeed, for each formula belonging to some MUSes, there exists at least one such a finite distance. To compare different formulae by their inconsistency values, only finite distances are meaningful. For free formulae, all the distances will be $+\infty$. But by *Free Formula Independence* principle, they should not be contributors to inconsistency anyway. Let us note $d^{max}_{MUS}(\alpha) = max\{d_{MUS}(\alpha, \mathcal{M}) \mid \mathcal{M} \in MUSes(K),\ d_{MUS}(\alpha, \mathcal{M}) \neq +\infty\}$. Note that the maximum distance is not more than the cardinality of the connected component that α belongs to, that is, $d^{max}_{MUS}(\alpha) < |\mathcal{C}|$.

Definition 13. *Let K be a knowledge base, and $\alpha \in K$. Write $\mathcal{S}_{MUS}(\alpha, k) = \{\mathcal{M} \in MUSes(K) \mid d_{MUS}(\alpha, \mathcal{M}) = k\}$. $\mathcal{SQ}_{MUS}(\alpha)$ is defined as follows: $\mathcal{SQ}_{MUS}(\alpha) = \{\mathcal{S}_{MUS}(\alpha, 0), \mathcal{S}_{MUS}(\alpha, 1), \ldots, \mathcal{S}_{MUS}(\alpha, d^{max}_{MUS}(\alpha))\}$.*

Note that $\mathcal{S}_{MUS}(\alpha, k)$ represents the set of MUSes with a distance k from α, and $\mathcal{SQ}_{MUS}(\alpha)$ gives the sequence of MUSes distributed in terms of the distance d_{MUS}.

Example 4. *(Example 2 Contd.) For $\alpha = \neg b$, we have $\mathcal{S}_{MUS}(\alpha, 0) = \{\mathcal{M}_2\}$, $\mathcal{S}_{MUS}(\alpha, 1) = \{\mathcal{M}_3, \mathcal{M}_4\}$, $\mathcal{S}_{MUS}(\alpha, 2) = \{\mathcal{M}_5\}$. Then, $\mathcal{SQ}_{MUS}(\alpha) = \{\{\mathcal{M}_2\}, \{\mathcal{M}_3, \mathcal{M}_4\}, \{\mathcal{M}_5\}\}$. Figure 1 depicts a graphical representation of the connected components of K. Also, Figure 2 represents the distribution of MUSes according to their distances to α. We remark that the MUSes are arranged into different levels.*

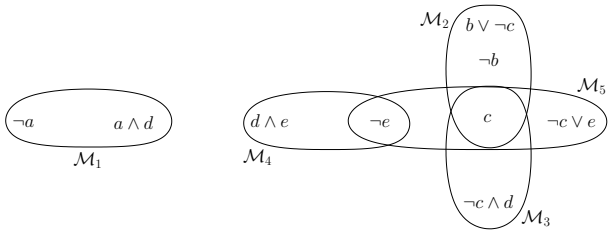


Figure 1: MUS-graph of K

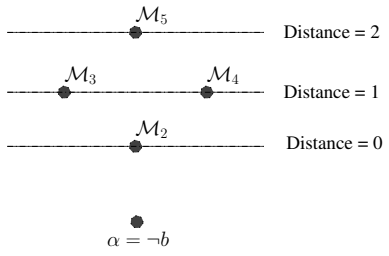


Figure 2: Distribution of MUSes according to α

It is easy to check that the $MIV_{\#}$ measure is just a characterization of the set $\mathcal{S}_{MUS}$ when applying uniquely the minimum distance from the formulae. More formally,

Proposition 5. *Let K be a knowledge base, and $\alpha \in K$. Then, $MIV_{\#}(K, \alpha) = |\mathcal{S}_{MUS}(\alpha, 0)|$.*

Proposition 5 shows that the $MIV_{\#}$ value considers only the nearest neighbors of α. However, this measure is not sufficiently discriminating for our purposes, since it takes into account only the first level of MUSes. Indeed, let us consider the formulae $d \wedge e$, and $\neg c \vee e$ of Example 2. According to $MIV_{\#}$ or MIV_C measure, these two formulae have the same inconsistency value. However, according to Figure 1 $d \wedge e$ and $\neg c \vee e$ do not have the same structural properties. Indeed, the MUS containing $\neg c \vee e$ is more connected than the one of $d \wedge e$. Hence, one has to go beyond the nearest neighbors to obtain a finer-grained measure.

A first inconsistency measure can be defined as follows:

Definition 14. *Let K be a knowledge base, and $\alpha \in K$. The inconsistency value of α is defined as:*

$$DIM_C(\alpha, K) = \frac{|\mathcal{S}_{MUS}(\alpha, 0)|}{d^{max}_{MUS}(\alpha)+1}$$

Unlike $MIV_{\#}$, the DIM_C value takes into account the structure of connected components by considering the nearest and the farthest MUSes. More precisely, the DIM_C measure aims to assign a higher value to a formula that has numerous nearest neighbors with the remaining *MUSes* concentrated around it. Put differently, while two formulae have the same number of neighbors of distance 0, the distance from the farthest MUSes allows us to find out the most inconsistent one. Note that if α is a self-contradictory formula, the DIM_C measure takes value one, i.e., $DIM_C(\alpha, K) = 1$.

Let us now illustrate the behavior of the DIM_C measure in the next example.

Example 5. *(Example 2 Contd.) It is not hard to see:*

$DIM_C(a \wedge d, K) = 1$	$DIM_C(\neg a, K) = 1$
$DIM_C(\neg b, K) = \frac{1}{3}$	$DIM_C(\neg c \wedge d, K) = \frac{1}{3}$
$DIM_C(\neg e, K) = 1$	$DIM_C(b \vee \neg c, K) = \frac{1}{3}$
$DIM_C(d \wedge e, K) = \frac{1}{3}$	$DIM_C(c, K) = \frac{3}{2}$
$DIM_C(\neg c \vee e, K) = \frac{1}{2}$	

Notice that now we can make a distinction between $d \wedge e$, and $\neg c \vee e$, since $DIM_C(d \wedge e, K) < DIM_C(\neg c \vee e, K)$.

However, the problem remains between the formulae $d \wedge e$, and $\neg b$. In order to make the measure more accurate, we propose to extend our framework by not only considering the farthest MUSes, but the whole structure of the connected components, which leads to a more useful inconsistency measure defined as follows:

Definition 15. *Let K be a knowledge base, and $\alpha \in K$. The inconsistency value of α is defined as:*

$$DIM_H(\alpha, K) = \sum_{\substack{\mathcal{M} \in MUSes(K) \\ d_{MUS}(\alpha, \mathcal{M}) \neq +\infty}} \frac{1}{d_{MUS}(\alpha, \mathcal{M}) + 1}$$

We can see more clearly by the following example that DIM_H gives a more precise view of the conflict brought by each formula.

Example 6. *(Example 2 Contd.) We have:*

$DIM_H(a \wedge d, K) = 1$	$DIM_H(\neg a, K) = 1$
$DIM_H(\neg b, K) = \frac{7}{3}$	$DIM_H(\neg c \wedge d, K) = \frac{7}{3}$
$DIM_H(\neg e, K) = 3$	$DIM_H(b \vee \neg c, K) = \frac{7}{3}$
$DIM_H(d \wedge e, K) = \frac{13}{6}$	$DIM_H(c, K) = \frac{7}{2}$
$DIM_H(\neg c \vee e, K) = \frac{7}{3}$	

Using the DIM_H measure, the formula $d \wedge e$ has now an inconsistency value $\frac{13}{6}$ less than that of $\neg b$ which is $\frac{7}{3}$.

The DIM_H measure could be refined by using the following notion of a weighting function that assigns a weight to each MUS in the MUS-graph. The idea is that a weight represents the significance of a MUS with respect to their distance from a given formula, thus leading to a better assignment of inconsistency responsibility to formulae. These weights can take into consideration other criteria like the degree of each MUS in the MUS-graph. A general definition is stated as follows.

DEFINITION 16. *Let K be a knowledge base, and $w(\mathcal{M}) \in \mathbb{R}$ a given weight function. The inconsistency value of α is defined as:*

$$DIM_W(\alpha, K) = \sum_{\substack{\mathcal{M} \in MUSes(K) \\ d_{MUS}(\alpha, \mathcal{M}) \neq +\infty}} \frac{w(\mathcal{M})}{d_{MUS}(\alpha, \mathcal{M}) + 1}$$

The following result shows that the DIM_W measure can be expressed by using the sequence $\mathcal{SQ}_{MUS}(\alpha)$. The idea is that $w(\mathcal{M})$ only depends on the distance between $\mathcal{M}$, and α.

PROPOSITION 6. *Let K be a knowledge base, and $\alpha \in K$. Then, the following equation holds:*

$$DIM_W(\alpha, K) = \sum_{\mathcal{S}_{MUS} \in \mathcal{SQ}_{MUS}} w(i) \times \frac{|\mathcal{S}_{MUS}(\alpha, i)|}{(i+1)}$$

Note that $w(i)$ represents the weight associated to each MUS at distance i from α.

4.1 Measuring Inconsistency of Sub-bases

Instances of a DIM (such as those given in Definitions 14, 15, and 16) can be obviously extended to a set of formulae. For this purpose it will be convenient to define the distance between a subset $K' \subseteq K$, and a MUS $\mathcal{M} \in MUSes(K)$.

DEFINITION 17. *Let K be a knowledge base, K' a subset of K, and $\mathcal{M} \in MUSes(K)$. The distance between K', and $\mathcal{M}$ is defined as:*

$$d_{MUS}(K', \mathcal{M}) = min\{d_{MUS}(\alpha, \mathcal{M}) \mid \alpha \in K'\}$$

Clearly, the set of *MUSes* can be partitioned into different subsets according to their distances to the subset K'. Here, we denote by $d^{max}_{MUS}(K')$ the maximum distance of *MUSes* from K'.

Now, the inconsistency measure of K' through DIM_C value is defined as follows:

DEFINITION 18. *Let K be a knowledge base, and $K' \subseteq K$. The inconsistency degree of K' is defined as:*

$$DIM_C(K', K) = \frac{|\mathcal{S}_{MUS}(K', 0)|}{d^{max}_{MUS}(K') + 1}$$

Interestingly, we note that while considering more than one formula, different connected components should be taken into account. In the light of the property of additivity, $DIM_C(K', K)$ can be rewritten as stated in Proposition 7.

PROPOSITION 7. *Let K be a knowledge base, $K' \subseteq K$, and $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ the MUS-decomposition of K. Then,*

$$DIM_C(K', K) = \sum_{i=1}^{i=p} DIM_C(K' \cap K_i, K)$$

EXAMPLE 7. *Let us consider again the knowledge base $K = \{a \wedge d, \neg a, \neg b, b \vee \neg c, \neg c \wedge d, \neg c \vee e, c, \neg e, e \wedge d\}$. Then, $DIM_C(\{a \wedge d, c\}, K) = 1 + \frac{3}{2} = \frac{5}{2}$.*

Similarly, DIM_C, DIM_H, and DIM_W inconsistency values can be naturally extended to a set of formulae. In particular, if we consider the case $K' = K$, then $DIM_C(K) = \frac{|\mathcal{S}_{MUS}(K,0)|}{d^{max}_{MUS}(K)+1} = |MUSes(K)|$, which corresponds to the I_{MI} measure. The same result is obtained when using DIM_H, i.e., $DIM_H(K) = |MUSes(K)|$.

4.2 Monotonicity

As seen earlier, the DIM_C value combines the distances to the nearest $MUSes$, and the inverse of that to the farthest in order to quantify the participation of each formula in the inconsistencies. Note that adding new formulae to a knowledge base may grow the distance to the farthest MUS, and consequently the DIM_C value decreases. This means that DIM_C is not monotonic. For illustration, consider the knowledge base $K = \{a, \neg a \wedge b, \neg b \wedge c, \neg c \wedge d, \neg d \wedge e\}$. Then, we have $DIM_C(a) = \frac{1}{4}$. Now if we add to K the formula $\neg e$, then the value of a in $K \cup \{\neg e\}$ becomes $\frac{1}{5}$.

In contrast, DIM_H counts the inverse of all distances from a formula to each MUS. Moreover, adding new formulae cannot decrease the number of $MUSes$, and cannot increase the distance of existing $MUSes$ from the formula, thus the inverse of the distances will be non-dereasing. Consequently, the DIM_H measure is monotonic.

5. MEASURING INCONSISTENCIES OF A WHOLE BASE

This section is devoted to the definition of an inconsistency measurement for a whole knowledge base.

To address this question, let us firstly give a general characterization of our measure with respect to the additivity property. Then, we discuss different measures obtained by different instantiations of the general case.

DEFINITION 19. *Let K be a knowledge base, and $\mathcal{CC} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ the connected components of K. The inconsistency measure of K, denoted $I_{\mathcal{CC}}(K)$, is defined as $I_{\mathcal{CC}}(K) = \sum_{i=1}^{n} \delta(\mathcal{C}_i)$ where δ is a function valuing in $\mathbb{R}$.*

The general definition given above allows for a range of possible measures. Next we first instantiate some $I_{\mathcal{CC}}$ measures by varying the δ function. The simplest one is obtained when $\delta(\mathcal{C}_i) = 1$. In this case, we get a measure that assigns to K the number of its connected components, i.e., $I_{\mathcal{CC}}(K) = |\mathcal{CC}|$. Note that this measure is not monotonic. Indeed, adding new formulae to the knowledge base can decrease the number of connected components of the base. For instance, let us consider the knowledge base $K = \{a, \neg a, b, \neg b\}$ that contains two connected components $\mathcal{C}_1 = \{a, \neg a\}$, and $\mathcal{C}_2 = \{b, \neg b\}$; now adding the formula $a \vee b$ to K leads to a new knowledge base containing a unique connected component $\mathcal{C} = \{a, \neg a, b, \neg b, a \vee b\}$. Note that this simple measure considers each connected component as an inseparable entity.

Moreover, when we consider $\delta(\mathcal{C}_i) = |\mathcal{C}_i|$, the $I_{\mathcal{CC}}$ measure leads to an existing inconsistency measure. More precisely, as $|\mathcal{C}_i|$ corresponds exactly to the number of $MUSes$ involved in the connected component $\mathcal{C}_i$, it is obvious to see that $I_{\mathcal{CC}}(K)$ is equal to I_{MI} measure, i.e., $I_{\mathcal{CC}}(K) = |MUSes(K)|$. This second value is of little interest, since it states exactly the fact that the inconsistency value only takes into account the number of the minimal inconsistent subsets of a base.

In the following, we propose to deeply explore the properties of additivity, and monotony to define a new interesting inconsistency measure.

DEFINITION 20. *Let K be a knowledge base, and $K_1, \ldots, K_n$ subsets of K. The set $\{K_1, \ldots, K_n\}$ is called a conditional independent MUS partition of K if the following conditions are satisfied:*

(1) $K_i \vdash \bot$, *for* $1 \leq i \leq n$,

(2) $MUSes(K_1 \cup \ldots \cup K_n) = \bigoplus_{1 \leq i \leq n} MUSes(K_i)$,

(3) $K_i \cap K_j = \emptyset$, $\forall i \neq j$.

According to Definition 20, K can be written as $K = K_1 \cup \ldots \cup K_n \cup R$ where R is a subset of K, and $\{K_1, \ldots, K_n\}$ is a conditional independent MUS partition of K. That is, when removing the set of formulae R from K the remaining base can be partitioned into sub-bases $K_1, \ldots, K_n$ satisfying conditions (1), (2), and (3) (Definition 20).

In general, for a given knowledge base K, there exist different subsets $R \subseteq K$ such that Definition 20 holds. Moreover, if $K = K_1 \cup \ldots \cup K_n \cup R$, then there exists $\{\mathcal{M}_1, \ldots, \mathcal{M}_n\} \subseteq MUSes(K)$ being a conditional independent MUS partition of K. In other words, $K = \mathcal{M}_1 \cup \ldots \cup \mathcal{M}_n \cup R'$ where $\mathcal{M}_i \cap \mathcal{M}_j = \emptyset$, $\forall\ 1 \leq i \neq j \leq n$, and $MUSes(\mathcal{M}_1 \cup \ldots \cup \mathcal{M}_n) = \oplus_{1 \leq i \leq n} \mathcal{M}_i$. Indeed, it is sufficient to pick a MUS $\mathcal{M}_i$ from each K_i since $K_i \vdash \bot$, and consider $R' = R \cup \{K_1 \setminus \mathcal{M}_1\} \cup \ldots \cup \{K_n \setminus \mathcal{M}_n\}$.

Let us now characterize our inconsistency measure I in the light of both additivity, and monotony properties. Using Definition 20, $I(K) = I(K_1 \cup \ldots \cup K_n \cup R)$. Using monotony property, we have $I(K) \geq I(K_1 \cup \ldots \cup K_n)$. Finally by additivity, we conclude that $I(K) \geq I(K_1) + \ldots + I(K_n)$.

Now, let us denote by $\mu_{max}(K)$ the maximal cardinality of sets $\{K_1, \ldots, K_n\}$ satisfying the conditions (1), (2), and (3) of Definition 20. That is, $\mu_{max}(K)$ corresponds to the number of maximal connected components that can be obtained while removing some formulae from K, i.e., the maximum value taken by n. By considering the maximal conditional independent partition value μ_{max}, one can deduce that if the measure I is additive then, $I(K) \geq I(K_1) + \ldots + I(K_{\mu_{max}(K)})$. Now, by using this bound one can define a new inconsistency measure as stated in Definition 21.

DEFINITION 21. *Let K be a knowledge base. We define the inconsistency measure of K as:*

$$I_{\mathcal{CC}}(K) = \mu_{max}(K)$$

So now we can show that:

PROPOSITION 8. *The inconsistency measure $I_{\mathcal{CC}}$ is additive, and monotonic.*

PROOF. Let K, and K' be two knowledge bases such that $K = K_1 \cup \ldots \cup K_{\mu_{max}(K)} \cup R$. Then, $K \cup K' = K_1 \cup \ldots \cup K_{\mu_{max}(K)} \cup R \cup K'$. Hence, $K \cup K' = K_1^{'} \cup \ldots \cup K_{\mu_{max}(K)}^{'} \cup R'$. Thus, $\mu_{max}(K \cup K') \geq \mu_{max}(K)$ which proves that $I_{\mathcal{CC}}$ is monotonic. Suppose now that $(K \setminus free(K)) \cap (K' \setminus free(K')) = \emptyset$, and $MUSes(K) \cap MUSes(K') = \emptyset$, it is easy to conclude that $\mu_{max}(K \cup K') = \mu_{max}(K) + \mu_{max}(K')$. As consequently, $I_{\mathcal{CC}}$ is additive. □

EXAMPLE 8. *Let us consider $K = \{a, \neg a, a \vee b, \neg b, b, c, \neg c \wedge d, \neg d \wedge e \wedge f, \neg e, \neg f\}$. The figure 3 depicts its connected components. K contains two connected components $\mathcal{C}_1$, and $\mathcal{C}_2$ such that $\mathcal{C}_1 = \{a, \neg a, a \vee b, \neg b, b\}$, and $\mathcal{C}_2 = \{c, \neg c \wedge d, \neg d \wedge e \wedge f, \neg e, \neg f\}$. The conditional independent MUS partition of $\mathcal{C}_1$ of maximum size is equal to 2, and can be obtained from $\mathcal{C}_1$ by removing $a \vee b$. For $\mathcal{C}_2$, for all removed subsets of $\mathcal{C}_2$, the number of resulting connected components can not exceed 1. Then, we have $I_{\mathcal{CC}}(K) = I_{\mathcal{CC}}(\mathcal{C}_1) + I_{\mathcal{CC}}(\mathcal{C}_2) = 3$.*

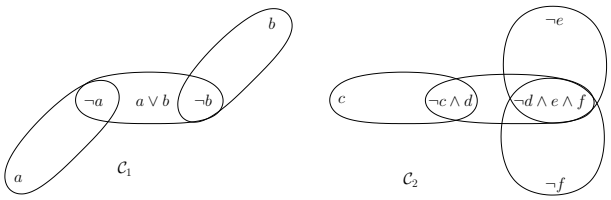


Figure 3: Connected components of the knowledge base K

Let us note that among measures that are additive, monotonic, and MinInc, I_{CC} is the smallest one, as shown below.

PROPOSITION 9. *For any I that is additive, monotonic, and MinInc, $I_{CC}(K) \leq I(K)$.*

PROOF. Indeed, an additive, and monotonic measure must attribute to a knowledge base a value greater than $I(K_1) + \ldots + I(K_{\mu_{max}(K)})$. When a measure satisfies MinInc, and monotonicity, $I(K_i) \geq 1$. Hence, $I(K) \geq \mu_{max}(K)$. □

Moreover, Proposition 10 shows the relationship between the maximal conditional independent MUS partition, and the smallest minimal hitting set.

PROPOSITION 10. *Let K be a knowledge base. Then,*

$$I_{CC}(K) \leq LB_{HS}(K)$$

PROOF. As K can be partitioned into $\mu_{max}(K)$ independent components by removing some formulae, then a minimal hitting set of K must contain at least one formula from each independent component of the maximal partition. Hence, $\mu_{max}(K) \leq LB_{HS}(K)$. □

EXAMPLE 9. *Let us consider again Example 8, and its second connected component $\mathcal{C}_2$. We have $LB_{HS}(K) = 2$ while the conditional independent MUS partition of K can not exceed 1.*

6. CONCLUSION

We proposed in this paper a new framework for defining inconsistency values that allow to associate each formula with its degree of responsibility for the inconsistency of a

whole knowledge base. This approach is based on the correlation between minimal inconsistent subsets which is shown a useful way to quantify the amount of inconsistencies in a finer way. We showed that such a framework can be extended to quantify the inconsistency of a whole knowledge base. We also proposed an enhanced additivity property to better capture its intuition according to the debat existing in the literature.

In the future, we plan to investigate deeply the architecture of the connected components. For instance, we plan to use logical argumentation theory [4] to deepen the analysis of the graph representation of a knowledge base [3]. We are also going to investigate results coming from graph theory in order to offer a finer grained evaluation of the inconsistencies. The theoretical complexity of our inconsistency measures, and practical algorithms are under investigation.

7. ACKNOWLEDGMENTS

The first author benefits from the support of both CNRS and OSEO within the ISI project Pajero.

8. REFERENCES

[1] S. Benferhat and L. Garcia. A local handling of inconsistent knowledge and default bases. In Applications of Uncertainty Formalisms, pages 325–353, 1998.

[2] L. E. Bertossi, A. Hunter, and T. Schaub. Introduction to inconsistency tolerance. In Inconsistency Tolerance, pages 1–14, 2005.

[3] P. Besnard, É. Grégoire, C. Piette, and B. Raddaoui. Mus-based generation of arguments and counter-arguments. In IRI, pages 239–244, 2010.

[4] P. Besnard and A. Hunter. A logic-based theory of deductive arguments. Artif. Intell., 128(1-2):203–235, 2001.

[5] Q. Chen, C. Zhang, and S. Zhang. A verification model for electronic transaction protocols. In APWeb, pages 824–833, 2004.

[6] D. Doder, M. Raskovic, Z. Markovic, and Z. Ognjanovic. Measures of inconsistency and defaults. Int. J. Approx. Reasoning, 51(7):832–845, 2010.

[7] J. Grant. Classifications for inconsistent theories. Notre Dame Journal of Formal Logic, 19(3):435–444, 1978.

[8] J. Grant and A. Hunter. Measuring inconsistency in knowledgebases. J. Intell. Inf. Syst., 27(2):159–184, 2006.

[9] J. Grant and A. Hunter. Analysing inconsistent first-order knowledgebases. Artif. Intell., 172(8-9):1064–1093, 2008.

[10] J. Grant and A. Hunter. Measuring consistency gain and information loss in stepwise inconsistency resolution. In ECSQARU, pages 362–373, 2011.

[11] J. Grant and A. Hunter. Distance-based measures of inconsistency. In ECSQARU, pages 230–241, 2013.

[12] A. Hunter. Measuring inconsistency in knowledge via quasi-classical models. In AAAI/IAAI, pages 68–73, 2002.

[13] A. Hunter. How to act on inconsistent news: Ignore, resolve, or reject. Data Knowl. Eng., 57(3):221–239, 2006.

[14] A. Hunter and S. Konieczny. Shapley inconsistency values. In KR, pages 249–259, 2006.

[15] A. Hunter and S. Konieczny. Measuring inconsistency through minimal inconsistent sets. In KR, pages 358–366, 2008.

[16] A. Hunter and S. Konieczny. On the measure of conflicts: Shapley inconsistency values. Artif. Intell., 174(14):1007–1026, 2010.

[17] S. Jabbour and B. Raddaoui. Measuring inconsistency through minimal proofs. In ECSQARU, pages 290–301, 2013.

[18] K. Knight. Measuring inconsistency. J. Philosophical Logic, 31(1):77–98, 2002.

[19] Y. Ma, G. Qi, and P. Hitzler. Computing inconsistency measure based on paraconsistent semantics. J. Log. Comput., 21(6):1257–1281, 2011.

[20] Y. Ma, G. Qi, G. Xiao, P. Hitzler, and Z. Lin. Computational complexity and anytime algorithm for inconsistency measurement. Int. J. Software and Informatics, 4(1):3–21, 2010.

[21] A. B. B. Martinez, J. J. P. Arias, and A. F. V. and. On measuring levels of inconsistency in multi-perspective requirements specifications. In PRISE'04, pages 21–30, 2004.

[22] M. V. Martinez, A. Pugliese, G. I. Simari, V. S. Subrahmanian, and H. Prade. How dirty is your relational database? an axiomatic approach. In ECSQARU, pages 103–114, 2007.

[23] K. McAreavey, W. Liu, P. Miller, and K. Mu. Measuring inconsistency in a network intrusion detection rule set based on snort. Int. J. Semantic Computing, 5(3), 2011.

[24] K. Mu, W. Liu, and Z. Jin. A general framework for measuring inconsistency through minimal inconsistent sets. Knowl. Inf. Syst., 27(1):85–114, 2011.

[25] K. Mu, W. Liu, and Z. Jin. Measuring the blame of each formula for inconsistent prioritized knowledge bases. J. Log. Comput., 22(3):481–516, 2012.

[26] K. Mu, W. Liu, Z. Jin, and D. A. Bell. A syntax-based approach to measuring the degree of inconsistency for belief bases. Int. J. Approx. Reasoning, 52(7):978–999, 2011.

[27] C. A. Oller. Measuring coherence using lp-models. J. Applied Logic, 2(4):451–455, 2004.

[28] G. Qi, W. Liu, and D. A. Bell. Measuring conflict and agreement between two prioritized belief bases. In IJCAI, pages 552–557, 2005.

[29] H. R. Robert Duncan Luce. Games and Decision. Wiley, 1957.

[30] M. A. H. Shmeil and E. Oliveira. Detecting the opportunities of learning from the interactions in a society of organizations. In SBIA, pages 242–252, 1995.

[31] G. Xiao, Z. Lin, Y. Ma, and G. Qi. Computing inconsistency measurements under multi-valued semantics by partial max-sat solvers. In KR, 2010.

[32] G. Xiao and Y. Ma. Inconsistency measurement based on variables in minimal unsatisfiable subsets. In ECAI, pages 864–869, 2012.

[33] L. Zhou, H. Huang, G. Qi, Y. Ma, Z. Huang, and Y. Qu. Measuring inconsistency in dl-lite ontologies. In Web Intelligence, pages 349–356, 2009.

Summarizing Large Graphs by Means of Pseudo-Boolean Constraints

Said Jabbour[1] and Nizar Mhadhbi[1] and Abdesattar Mhadhbi[2] and Badran Radaoui[3] and Lakhdar Sais[1]

[1] *CRIL - CNRS UMR 8188, University of Artois, France*
{*jabbour,mhadahbi,sais*}*@cril.fr*
[2] *Faculty of Sciences of Gafsa, Tunisia*
mhadhbiabdesattar@gmail.com
[3] *LIAS - ENSMA, University of Poitiers, France*
badran.raddaoui@ensma.fr

***Abstract*—Many real-world problems are effectively modeled as graphs. However, the resulting graph-based representation are usually of very large size leading to an increasing computational complexity in processing such graphs. Starting from the fact that any arbitrary graph contains hidden structures corresponding to particular classes of graphs, our goal in this paper is to discover such specific structures and to exploit them to summarize the main components of the graph using a more compact representation. More precisely, we first show that some special graph classes such as cliques and bi-cliques can be represented efficiently as *Pseudo-Boolean constraints*. Then, we propose three new interesting graph classes, called *nested*, *sequence* and *clique-nested bi-partite* graphs. All these classes are bi-partite graphs satisfying some specific properties. We also show that such graph classes can be expressed in a more succinct manner using Pseudo-Boolean constraints. We then present a more general approach for a partial or a complete summarization of an arbitrary graph as a disjunction of Pseudo-Boolean constraints. This new representation can be seen as an original way to represent the edges of the graph, as they correspond to particular solutions of the Pseudo-Boolean constraints. Experimental results on real-world datasets, show that our approach allows us to efficiently summarize large graphs in a compact way and enable further analysis.**

***Keywords*-Graph Mining, Graph Summarization, Pseudo Boolean Constraints.**

I. Introduction

Graphs present a general tool for modeling structural relationships between data objects. They have been widely used in several application domains, including interactions in bioinformatics, link analysis in social networks, XML documents, etc. Unfortunately, correlation mining in graph databases is expensive due to the complexity of graph data. In fact, in most of these applications, graphs are very large, with thousands or even millions of nodes and edges. Subsequently, it is difficult to understand the information/structure encoded in these graphs by simple visualization.

Taking as an example protein-protein interaction (PPI) networks [1]. These representations are depicted by graphs where nodes and edges represent proteins and interactions, respectively. Such dependencies provide relevant insights into the underlying mechanisms of biological processes within a cell. Indeed, interacting proteins are likely to have related cellular roles. As pointed out in [2], identifying the sets of proteins that are involved in the same biological processes is an important computational task. A common approach to this issue, is to partition the interaction graph into modules (subsets of proteins) based on the connectivity of the nodes. Proceeding from the related processes common proteins tend to be situated near one another within the network [3]. In fact, identifying such modules can help to understand the structure of the cell.

Revealing the topological structure of the graph is at the basis of many approaches in social networks, such as community detection, and clustering (e.g. [4]). Generally, real networks involve high concentrations of edges within special groups of vertices, and low concentrations between these groups. As an example, in the graph of the World Wide Web, high concentrations may correspond to groups of pages dealing with the same or related topics. Due to their degree of connectedness, these communities, clusters or modules might sometimes match some graph classes such as (almost) cliques, bicliques, bipartite graphs, etc. In this paper, we aim at discovering such specific structures, which enables efficient summarization. Moreover, we exploit the discovered structures to summarize the main components of the graph using a more compact representation while maintaining the possibility to uncover the original graph. Indeed, effective graph summarization approaches are crucial since they can assist in uncovering useful insights about the patterns hidden in the underlying data. Therefore, graph summarization and compression have been a subject of intense research activities (see the related work section).

To the best of our knowledge, it is the first time where a graph is represented using Pseudo-Boolean constraints (from now on "PB-constraints") (PB constraints for short), i.e., 0-1 linear inequalities. The main originality is that each PB constraint allows us to implicitly represent large parts

of the graph. We particularly show that any graph can be represented as a disjunction of PB constraints. Since a PB constraint can be encoded as a Boolean formula in conjunctive normal form, our results suggest that any graph can be represented as a Boolean formula. Consequently, we believe that our new compact representation opens a new research avenue for reasoning on large graphs and their underlying structures. Without loss of generality, in this paper we focus on undirected graphs and on the discovery of its most connected components corresponding to special graphs classes.

We first show that some special graph classes such as cliques and bicliques can be represented efficiently as PB constraints. Then, we propose three new interesting graph classes, called *nested*, *sequence* and *nested-sequence* bipartite graphs. All these classes are bipartite graphs satisfying some specific properties. We also show that such graph classes can be represented using PB constraints. We then present a more general approach for a partial or a complete summarization of an arbitrary graph as a disjunction of PB constraints. This new representation can be seen as an original way to represent the edges of the graph, as they correspond to particular solutions of the PB constraints.

The roadmap of this paper is organized as follows: We start by introducing some necessary definitions about PB constraints and graphs. Then, in Section III we present our proposed approach that uses PB constraints to identify and encode particular subgraphs. In addition to existing types of graphs that can be encoded, we propose a number of additional structures that can be represented efficiently as PB constraints. Section IV reports our experimental results on different real-world datasets. In Section V, we briefly overview related work. Finally, we conclude in Section VI.

II. Preliminaries

Let us introduce some necessary definitions about PB constraints and graphs. First, we assume a countable set $\mathcal{V}$ of propositional variables. The set of *propositional formulas* $\mathcal{F}_{\mathcal{V}}$ is defined inductively from $\mathcal{V}$ under the logical connectives ($\vee$, $\wedge$, $\neg$, $\rightarrow$), and the Boolean constants $\top$ (true) and $\bot$ (false). We will use $\phi, \psi, \ldots$ to denote propositional formulas. The set of propositional variables occurring in a formula ϕ is denoted $\mathcal{V}_\phi$. A propositional formula is said to be in CNF, *conjunctive normal formula*, if it is a set (interpreted as a conjunction) of *clauses*, where a clause is a set (interpreted as a disjunction) of *literals*. A literal is either x, or its negation $\neg x$, for a propositional variable x. Using linear Tseitin encoding [5], any formula ϕ can be translated to a CNF formula equivalent w.r.t. satisfiability.

An *interpretation* $\mathcal{B}$ of a formula ϕ is a function that assigns a truth value $\mathcal{B}(x) \in \{0, 1\}$ (0 corresponds to $false$ and 1 to $true$) to some of the variables $x \in \mathcal{V}_\phi$. It can be inductively extended to a set of formulas using the usual semantic. $\mathcal{B}$ is said to be *complete* if it assigns a value to every $x \in \mathcal{V}_\phi$, and *partial* otherwise. An interpretation is alternatively represented by a set of literals. A *model (or implicant)* of a formula ϕ is an interpretation $\mathcal{B}$ that satisfies the formula, denoted $\mathcal{B} \models \phi$. An implicant $\mathcal{B}$ of ϕ is called a *prime implicant*, iff for all literals $l \in \mathcal{B}$, $\mathcal{B} \backslash \{l\} \not\models \phi$. A *satisfiability problem* consists in deciding whether a given CNF formula Φ admits a model or not.

Let us now introduce PB constraints (or 0-1 linear inequalities). A PB constraint C_{pb} is defined as follows:

$$\sum_{i=1}^{n} a_i x_i \ op \ b \tag{1}$$

where a_i and b are integer coefficients, x_i $(1 \leq i \leq n)$ are propositional variables, and the relation operator op belongs to $\{=, \leq, <, >, \geq\}$. it can be noticed that one can assume that op is $\leq$ and the a_i and b are positive since all the other cases can be easily reduced to this normalized form [6]. Additionally, when it is more convenient we will consider the following two-sided PB constraint C_{pb} by introducing a lower and upper bounds:

$$b_{min} \leq \sum_{i=1}^{n} a_i x_i \leq \ b_{max} \tag{2}$$

An important special case of PB constraints is the one of *cardinality constraints*, in which all coefficients a_i $(1 \leq i \leq n)$ are equal to 1. Obviously, any clause $x_1 \vee \ldots \vee x_n$ can be written as a cardinality constraint as follows:

$$\sum_{i=1}^{n} x_i \geq 1 \tag{3}$$

Let us note that a negative literal $\neg x$ can also be written as $(1 - x)$.

Next, we extend the notion of interpretation to apply to PB constraints. Let C_{pb} be a PB constraint as defined in equation (1) and $\mathcal{B}$ an interpretation of the variables of C_{pb}, we define the interpretation $\mathcal{B}(C_{pb})$ as follows:

$$\sum_{i=1}^{n} a_i \mathcal{B}(x_i) \ op \ b \tag{4}$$

We say that $\mathcal{B}$ is a model of C_{pb} iff $\mathcal{B}(C_{pb}) = 1$. In the similar way we can extend $\mathcal{B}$ to two-sided PB constraint. For a PB constraint C_{pb}, $\mathcal{M}(C_{pb})$ denotes the set of models of C_{pb}. We note $\mathcal{M}^t_{\mathcal{B}}(C_{pb}) = \{x \in \mathcal{V}_{C_{pb}} | \mathcal{B}(x) = 1\}$ the set of variables of $\mathcal{V}_{C_{pb}}$ that are true in the model $\mathcal{B}$. A model $\mathcal{B} \in \mathcal{M}(C_{pb})$ is called a $k\text{-}model$ of the PB constraint C_{pb} iff $|\mathcal{M}^t_{\mathcal{B}}(C_{pb})| = k$. The set of k-models of C_{pb} is denoted $k\text{-}models(C_{pb})$. Note that the concept of prime implicant can be extended to PB constraint in a straightforward way.

PB constraints are at the heart of integer linear programming and propositional satisfiability. They are essential for modeling a wide variety of applications. Indeed, several polynomial encodings of this kind of constraints into a CNF

formula have been introduced in the literature. The first linear encoding of general 0-1 linear inequalities to CNF has been proposed by Warners [7]. Recently, efficient encodings of the cardinality constraint to CNF have been proposed (see [8], [9]). Similarly, various CNF polynomial encodings have been studied for general PB constraints of the form (2) (see [9]).

Now, we present some terminology about graphs that will be used throughout the paper. A *graph* is a pair of sets $G = (X, E)$. X is the set of vertices where the number of vertices $|X| = n$ is the *order* of the graph. The set $E \subseteq X \times X$ contains the *edges* of the graph with $|E| = m$. In an undirected graph each edge is an unordered pair $\{u, v\}$. We note $\langle X \rangle$ a sequence of nodes. We say that v is a *neighbor* of u, if $\{u, v\} \in E$. The set of neighbors of a given vertex v is called the *neighborhood* of v, denoted $\Gamma(v, G)$. When there is no ambiguity, we note $\Gamma(v, G)$ simply as $\Gamma(v)$. We will use $|\Gamma(v)|$ to refer to the *degree* of v. A graph $G^s = (S, E^s)$ is a *subgraph* of a graph $G = (X, E)$ iff $S \subseteq X$ and $E^s = \{\{v, u\} \in E \mid u, v \in S\}$. A graph is called *complete* if every pair of distinct vertices is connected by a unique edge. A *clique* of an undirected graph is a complete subgraph. A *maximal clique* is a clique that cannot be extended by including one more adjacent vertex. Note that enumerating all maximal cliques of a graph requires $3^{|X|/3}$ time in the worst case [10]. The problem of testing whether a graph contains a clique of size k is NP-Complete. Moreover, a *bipartite* graph is a graph $G = (X, Y, E)$ such that $E \subseteq X \times Y$. Alternatively, a bipartite graph is a graph that does not contain any odd-length cycles. A *biclique* is a complete bipartite graph, i.e., $E = X \times Y$. Similarly, the problem of identifying bicliques under certain size conditions is NP-Complete. Bipartite graphs are ubiquitous in many applications where the vertices represent two distinct classes of objects, such as customers and products.

III. Pseudo-Boolean Based Graph Summarization

In this section, we present our graph summarization approach. Our aim is to discover hidden structures corresponding to some graph classes and to exploit them to represent more succinctly the main components of the graph using a more compact representation, namely PB constraints.

A. Illustrative Examples and Motivations

Before introducing our framework, we first present some simple motivating examples. Let us consider a clique, a well-known graph class, $G = (X, E)$ s.t. $X = \{x_1, \ldots, x_n\}$ as depicted in Figure 1.a (left hand side). Then, G can be encoded by the following PB constraint $C_{pb}(G) = \sum_{1 \leq i \leq n} x_i = 2$.

Clearly, the 2-$models$ of $C_{pb}(G)$ correspond exactly to the edges of G. More precisely, the edge $\{x_i, x_j\}$ corresponds to the 2-$model$ $\mathcal{B}$ where $\mathcal{B}(x_i) = 1$, $\mathcal{B}(x_j) = 1$ and $\mathcal{B}(x_k) = 0$ for all $1 \leq k \leq n$ s.t. $k \neq i$ and $k \neq j$. So any clique can be encoded as a cardinality constraint. This allows us to summarize a graph with n nodes and n^2 edges by only one cardinality constraint with n variables, 1 positive integer and a sum operator.

Note that even if an arbitrary graph might contain several cliques that can be encoded as a *disjunction* of PB constraints, it would be better to represent also other less restrictive graph classes. Let us consider for example a *near clique*, a complete graph with one missing edge $\{x_1, x_2\}$ (see Figure 1 - right hand side). This graph remains expressible through the following equation: $2(x_n + \ldots + x_3) + x_2 + x_1 \geq 3$. Indeed, the 2-$models$ of the PB constraint, with two variables assigned to 1, corresponds exactly to the edges of the near clique. It is important to note that the 2-$models$ of the two previous PB constraints are prime implicants.

These two examples show the potential benefits behind using PB constraints to summarize in a compact way some graph classes. In the sequel, we generalize our approach to other well-known graph classes including three novel ones.

B. Pseudo Boolean Formulas

As an arbitrary graph can not be represented as a single PB constraint, its summarization can be achieved through a disjunction of PB constraints, where each constraint encode a subgraph of the original graph. Obviously, to encode an arbitrary graph G, one need to partition the set of edges in G. In fact, each element of the partition corresponds to a subgraph expressible as a PB constraint.

Definition 1 (Edges Partition): Let $G = (X, E)$ be an undirected graph. We define an *edges partition* of G as a set of graphs $P_e(G) = \{G_1, \ldots, G_k\}$ where $\forall 1 \leq i \leq k$, $G_i = (X_i, E_i)$ s.t. $\{E_1, \ldots, E_k\}$ is a partition of E and $X_i = \{x \in X \mid \exists \{x, y\} \in E_i\}$.

Next, we define a *pseudo boolean formula* as a disjunction of PB constraints s.t. each constraint is a two sided PB constraint of the form (1) or (2).

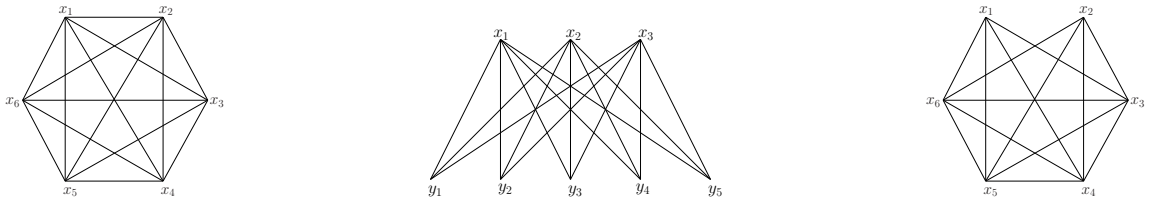
Definition 2: Let $G = (X, E)$ be an undirected graph s.t. $P_e(G) = \{G_1, \ldots, G_k\}$ is an edges partition of G. Then, the *PB constraints based summarization* of G, denoted $F_{pb}(G)$, is defined as $F_{pb}(G) = \mathcal{C}_{pb}(G_1) \vee \ldots \vee C_{pb}(G_k)$.

To ensure correctness, our PB constraints based summarization must satisfy some principles as shown by the following definition.

Definition 3: Let $G = (X, E)$ be an undirected graph. A *PB formula* $F_{pb}(G)$ is a *correct* PB constraints based summarization of G if it satisfies the following properties:

- $\mathcal{B} \models F_{pb}(G)$ can be answered in linear time,
- there is a one to one mapping between 2-$models(F_{pb}(G))$ and E,
- 1-$models(F_{pb}(G)) = \emptyset$.

Let us note that a straightforward representation of the whole graph as a PB formula consists in encoding each sin-



a. Clique b. Biclique c. Almost Clique

Figure 1: (Almost) Clique and Biclique

gle edge $\{x_i, x_j\}$ as a PB constraint of the form $x_i + x_j = 2$. Clearly, such summarization is not compact.

C. From Clique-Bipartite Graphs to New and Hybrid Classes

In this subsection, we present new classes of graphs that can be expressed using PB constraints. These structures are obtained by relaxing the definition of well-known graph classes (e.g. bipartite, cliques) or by building hybrid classes from them.

1) Nested-Bipartite Graph: Before defining our first new graph class, we consider the case where a bipartite graph is complete (biclique). Let us recall that traditional approaches for compressing bipartite graphs use an additional node, called virtual node, connecting one side of the bipartite graph to that virtual node and the virtual node to the other side of the bipartite graph (e.g. [11], [12]). Such approaches allow to reduce the number of edges from a quadratic number to a linear number. To highlight the expressive power of PB constraints, we now consider biclique graphs class (see Figure 1.b) as stated in the following proposition.

Proposition 1: Let $G = (X, Y, E)$ be a biclique where $|X| = n$ and $|Y| = m$. Then, G can be expressed as:

$$2 \times \sum_{i=1}^{n} x_i + 3 \times \sum_{j=1}^{m} y_j = 5 \tag{5}$$

Clearly, equation (5) is a two sided PB constraint of the form (2) where $b_{min} = b_{max} = 5$.

In the following, we extend the class of bipartite graphs (see Definition 5). For that we need some further notation.

Definition 4: Let $G = (X, E)$ be a graph and $u, v \in X$. The nodes u and v are called *nested nodes*, denoted $u \subseteq_\Gamma v$, iff $\Gamma(u) \subseteq \Gamma(v)$. Then, $\langle X \rangle = \langle x_1 \dots x_n \rangle$ is a *nested sequence of nodes* if X is totally ordered under the binary relation $\subseteq_\Gamma$, i.e., $x_n \subseteq_\Gamma x_{n-1} \subseteq_\Gamma \dots \subseteq_\Gamma x_1$.

Definition 5: Let $G = (X, Y, E)$ be a bipartite graph. Then, G is called a *Nested-Bipartite graph* w.r.t X (NB graph for short) if $\langle X \rangle$ is a nested sequence of nodes.

That is, a biclique is specific case of the NB graph where $\Gamma(x_1) = \Gamma(x_2) = \dots = \Gamma(x_n)$.

The following Definition is useful in order to derive the PB constraint for NB graphs.

Definition 6: Let $G = (X, Y, E)$ be a bipartite graph. Then, G is a *Nested-Ordered Bipartite graph* (NOB graph for short) if there exists a sequence of nested nodes $\langle X \rangle = \langle x_1 \dots x_n \rangle$ and a sequence $\langle Y \rangle = \langle y_1 \dots y_m \rangle$ s.t. $\forall i \in \{1 \dots n\}, \langle \Gamma(x_i) \rangle = \langle y_1 \dots y_{m_i} \rangle$ where $m_i = |\Gamma(x_i)|$.

Definition 6 states that a NOB graph $G = (X, Y, E)$ is a NB graph where Y can be reordered as a sequence $\langle Y \rangle$ s.t. each $\Gamma(x_i)$ is represented as a sub-sequence of $\langle Y \rangle$ starting with y_1 (All the neighborhood sequences of x_i start on the same node y_1). Let us note that $m = m_1 \geq m_2 \geq \dots \geq m_n$, since $\langle x_1 \dots x_n \rangle$ is a sequence of nested nodes.

Next, we show that a NB graph w.r.t X is a NOB graph.

Proposition 2: If $G = (X, Y, E)$ is a NB graph w.r.t. X then G is a NOB graph.

By Proposition 2, if a graph is a NB graph w.r.t. X, then it is also a NB graph w.r.t. Y.

In the sequel, we denote the NOB representation associated to a NB graph G by G_{nob}.

Example 1: Figure 2 shows how a NB graph can be reordered as a NOB graph. As we can see, the graph on left side of the Figure is a NB graph w.r.t. to X, i.e., $x_3 \subseteq_\Gamma x_2 \subseteq_\Gamma x_1$. The graph on the right side shows the obtained NOB structure, where $y_4 \subseteq_\Gamma y_3 \subseteq_\Gamma y_2 \subseteq_\Gamma y_1 \subseteq_\Gamma y_5$ and $\langle \Gamma(x_3) \rangle = \langle y_4, y_3 \rangle$, $\langle \Gamma(x_2) \rangle = \langle y_4, y_3, y_2 \rangle$ $\langle \Gamma(x_3) \rangle = \langle y_4, y_3, y_2, y_1, y_5 \rangle$. Then, all the neighborhood sequences of nodes of the variables from X admit the same starting node y_2.

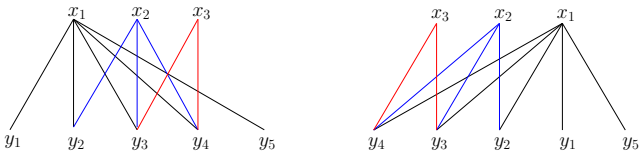


Figure 2: From NB Graph to NOB Graph

In Proposition 2, we have shown that a NB graph can be represented as a NOB graph. Such representation allows us to obtain the PB constraint expressing the NB graph.

Proposition 3: Let $G = (X, Y, E)$ be a NB graph w.r.t. X. Let $G_{nob} = (\langle X \rangle, \langle Y \rangle, E)$ be a NOB representation of G where $\langle X \rangle = \langle x_1 \ \dots \ x_n \rangle$ and $\langle Y \rangle = \langle y_1 \ \dots \ y_m \rangle$. Then, $G = (X, Y, E)$ is expressible with the following two sided

PB constraint:

$$-m \leq \sum_{i=1}^{m}(m+j) \times y_j - \sum_{j=1}^{n}(m+m_i) \times x_i \leq 0 \quad (6)$$

As for NB graphs, the PB constraint expressed bicliques is as follows.

$$-m \leq \sum_{i=1}^{m}(m+j) \times y_j - 2m\sum_{j=1}^{n} x_i \leq 0 \quad (7)$$

Example 2: Let us consider the graph depicted in Figure 2. This graph can be expressed as: $-5 \leq 6y_4 + 7y_3 + 8y_2 + 9y_1 + 10y_5 - 10x_1 - 8x_2 - 7x_3 \leq 0$.

Let us now state an interesting property of NB graphs.

Definition 7: We define the complement of a bipartite graph $G = (X, Y, E)$ as the bipartite graph $G^c = (X, Y, E^c)$, where $E^c = \{\{x, y\} \in X \times Y | \{x, y\} \notin E\}$.

Proposition 4: If $G = (X, Y, E)$ is a NB graph, then $G^c = (X, Y, E^c)$ is a NB graph.

2) Sequence-Bipartite Graphs: Let us now present our second graph class, called *Sequence-Bipartite graph*.

Definition 8: A bipartite graph $G = (X, Y, E)$ is a *Sequence-Bipartite graph* (SB graph for short) if X and Y can be written as sequences $\langle X \rangle = \langle x_1 \dots x_n \rangle$ and $\langle Y \rangle = \langle y_1 \dots y_m \rangle$ s.t. there exists an integer $k > 0$ where $\forall i \in \{1 \dots n\}$, $\exists k_i \in \{1 \dots m\}$ s.t. $\langle \Gamma(x_i) \rangle = \langle y_{1+k_i} \dots y_{k+k_i} \rangle$.

That is, a SB graph is a new class of the bipartite graph that consists of a set of nodes X where their neighbors are sub-sequences of size k successively translated with k_i over $\langle Y \rangle$.

In the following, let $G_{sb} = (\langle X \rangle, \langle Y \rangle, E)$ represents the SB graph associated to $G = (X, Y, E)$ where $\langle X \rangle$ and $\langle Y \rangle$ are two sequences of the nodes of X and Y, respectively, satisfying the conditions of Definition 8.

Proposition 5: If $G = (X, Y, E)$ is a SB graph, then $G^c = (X, Y, E^c)$ is a SB graph.

Proposition 6: A SB graph is expressible through the following two sided PB constraint:

$$1 \leq \sum_{j=1}^{m}(k+j) \times y_j - \sum_{i=1}^{n}(k+k_i) \times x_i \leq k \quad (8)$$

Example 3: Figure 3 depicts an SB graph. We have $\forall x \in X$, $|\Gamma(x)| = 5$. The neighborhood sets of $x_1, \dots, x_5$ are obtained by translation with $k_1 = 0$, $k_2 = 0$, $k_3 = 1$, $k_4 = 2$, $k_5 = 2$, respectively. This graph can be expressed by the following two-sided PB constraint: $0 \leq 8(y_1 - x_3) + 9(y_2 - x_4 - x_5) + 10y_3 + 11y_4 + 12(y_5 + y_7) + 13y_6 - 7(x_1 + x_2) \leq 5$.

We note that a biclique is specific case of SB graphs where $k_i = 0$, $\forall i \in \{1 \dots n\}$ and $k = m$. Hence, a biclique can be expressed differently using the two-sided PB constraint:

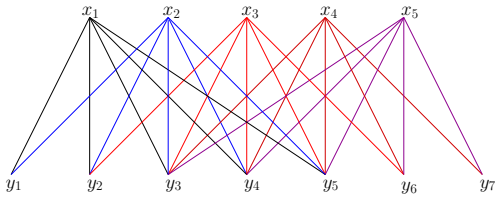


Figure 3: A Sequence-Bipartite Graph

$$1 \leq \sum_{j=1}^{m}(m+j) \times y_j - m\sum_{i=1}^{n} x_i \leq m \quad (9)$$

3) Clique-Nested-Bipartite graph: Here, we define a third hybrid structure combining both cliques and NB graphs.

Definition 9: A graph $G_{cnb} = (X, Y, E)$ is called *Clique-Nested-Bipartite graph* (CNB graph for short) if it satisfies the following conditions:

1) The graph $G_c = (X, E')$ with $E' = \{\{u, v\} \in E \mid u, v \in X\}$ is a clique, and
2) The graph $G_{nb} = (X, Y, E'')$ with $E'' = \{\{u, v\} \in E \mid u \in X, v \in Y\}$ is a NB graph w.r.t. X.

Proposition 7: A CNB graph can be expressed using the following two sided PB constraint.

$$\sum_{j=1}^{m}(-m-1+j) \times y_j - \sum_{i=1}^{n}(m-1+m_i) \times x_i \leq -2m \quad (10)$$

Note that for the particular case of *clique-biclique* graphs (see Figure 4 right side), the PB constraint can be expressed as follows:

$$\sum_{i=1}^{m}(-m-1+j) \times y_j - (2m-1)\sum_{j=1}^{n} x_i \leq -2m \quad (11)$$

Example 4: Let us consider the CNB graph depicted in Figure 4 (left side). This graph contains two sets of nodes $X = \{x_1, x_2, x_3\}$ and $Y = \{y_1, \dots, y_5\}$. As we can see, the graph consists of two subgraphs, a clique involving X and a NB graph on X and Y w.r.t. $\{x_1, x_2, x_3\}$, i.e, $x_3 \subseteq_\Gamma x_2 \subseteq_\Gamma x_1$. This graph can be expressed as: $5y_1 + 4y_2 + 3y_3 + 2y_4 + y_5 + 9x_1 + 8x_2 + 6x_3 \geq 10$.

4) Graph classes: a Summary: Let us now summarize the different graph classes discussed above. In Figure 5, we describe the relations between them. Let A and B be two graph classes, an arc $A \rightarrow B$ means that A is a subclass of B. In other words, there are graphs in the class B that do not belong to A. For clarity, we depict the new graph classes by grey color.

5) Summarization by Subgraph Discovery: an Heuristic Approach: Let us now address the problem of summarizing arbitrary graphs as PB formulas. Our approach is based on subgraph classes discovery. We are interested to detect the

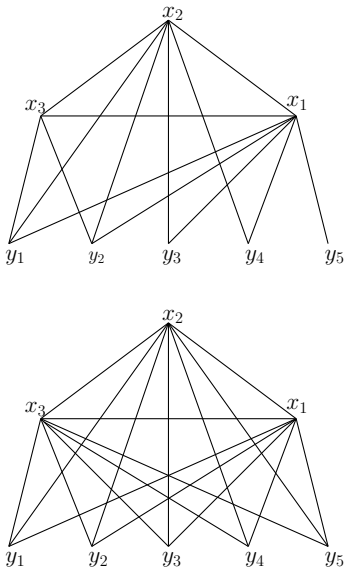


Figure 4: Clique-Nested-Bipartite Graphs

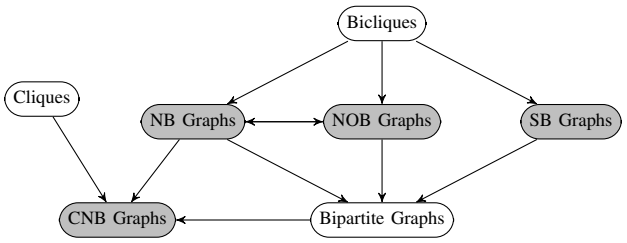


Figure 5: Graph Hierarchy

subgraphs that can be represented through PB constraints and then iterate such process until no "interesting" part of the graph can be represented compactly. As discussed in Section II, discovering some structures is NP-Complete. So, an exact discovery technique, which considers all the possible ordered combinations of the candidate subgraph classes and chooses the best summarization is intractable and not usable in practice. Thus, we need a heuristic-based approach that will give an efficient but approximate summarization of the whole graph. To show the feasibility of our proposed framework, we adopt a greedy approach limited to the discovery of NB graphs. Our first graph summarization approach using the NB graphs class as a PB formula is described in Algorithms 1 and 2. More precisely, the first algorithm is called to search for the next NB graph, while the second one encodes these NB subgraphs as a PB formula.

Let us start our explanation with Algorithm 1. It takes as input the current graph, and output the NB subgraph and the remaining graph (or graph residue). The graph residue is obtained by deleting the edges of the NB subgraph from the current graph, and then by deleting the single nodes (nodes with degree 0). The second Algorithm 1 proceeds as follows. It starts by initializing the graph residue to the current graph and the NB graph to an empty graph (line 3). In line 4, we look for the node x with the maximum degree in the current graph ($maxDegreeNode$). The set Y is set to the neighbors of x (line 5). In the repeat loop (lines 6-13), we search for the next node with a maximum overlap ($maxOverlapNeighbors$) with Y in the graph residue (line 13). It is important to note that this function return nil if there is no node x sharing neighbors with Y. At each iteration, we update the current set of neighbors Y (line 7), the NB graph G_{nb} (line 8) and the residue graph G_r (line 9-11) accordingly. As we can see, in line 10 and 11, we also delete the single nodes from the graph residue.

In Algorithm 2, we present our NB Graph Summarization approach based on PB constraints (called $PBnbGS$). We take an arbitrary graph G and return a PB formula encoding all the NB subgraphs discovered by successive calls to Algorithm 1. Our algorithm also outputs the waste subgraph G_w that can not be encoded compactly using our approach. More precisely, we search for the next NB subgraph (line 5). The PB constraint encoding the returned NB subgraph is added disjunctively to the PB formula only if its number of nodes is at least two (lines 6-7). Otherwise, there is no benefit to encode a NB subgraph with only one node and its neighbors. Obviously, this particular NB subgraph (a biclique) can be encoded using the PB constraint (5), but not in a compact way. In this case the returned NB subgraph is added to the waste graph (line 8). The algorithm ends when the set of edges of the graph residue is empty.

Finally, our approach is also able to encode an arbitrary graph with a PB formula without waste subgraph. Indeed, the NB subgraphs that we decided to not encode are bicliques of the form $(\{x\}, Y, E)$. Indeed, using the PB constraint (5), we obtain $2 \times x + 3 \times \sum_{y \in Y} y = 5$.

IV. Experimental Evaluation

In this section, we consider NB graphs as described previously in Section III-C and we present some empirical results regarding the coverage that we can achieve using them. We also report the maximum, the minimum, and the average size among the NB graphs extracted from an input graph.

The experiments are carried out on a PC with an Intel Core 2 Quad Q9400 processor, with 8 GB RAM, and the Precise Pangolin x86_64 OS. Our implementation of Algorithm 2 was written in Python and run with PyPy 2.4.0 (http://pypy.org/). Only one CPU core was used.

In these preliminary experiments, we emphasize on the summarization aspect. We are particularly interested in the structure identification that will help the user to better understand the main graph characteristics.

We implemented and tested our proposed approach on a wide variety of large scale graphs: biological networks,

Algorithm 1: nextNB_Graph

Input: a graph $G = (X, E)$
Output: (a NB subgraph $G_{nb} = (X^{nb}, E^{nb})$, a graph residue $G_r = (X^r, E^r)$)

$(X^r, E^r) \leftarrow (X, E)$; $(X^{nb}, E^{nb}) \leftarrow (\emptyset, \emptyset)$;
$x \leftarrow\ maxDegreeNode(G_r)$;
$Y \leftarrow \Gamma(x, G_r)$;
repeat
  $Y \leftarrow Y \cap \Gamma(x, G_r)$;
  $(X^{nb}, E^{nb}) \leftarrow (X^{nb} \cup \{x\}, E^{nb} \cup \{\{x, v\} | v \in Y\})$;
  $(X^r, E^r) \leftarrow (X^r, E^r \setminus E^{nb})$;
  if $(|\Gamma(x, G_r)| = 0)$ **then**
    $X^r \leftarrow X^r \setminus \{x\}$;
  end
  $x \leftarrow\ maxOverlapNeighbors(G_r, Y)$;
until $(x = nil)$;
return (G_{nb}, G_r);
common

Algorithm 2: Summarization using Linear Inequalities (SuLI)

Input: a graph $G = (X, E)$
Output: (a PB Formula F_{pb}, a Waste Graph $G_w = (X^w, E^w)$)

$F_{pb} = \top$;
repeat
  $(G^{nb}, G_r) \leftarrow nextNB\_Graph(G)$;
  if $(|X^{nb}| > 1)$ **then**
    $F_{pb} \leftarrow F_{pb} \vee C_{pb}(G_{nb})$;
  end
  else
    $(X^w, E^w) \leftarrow (X^w \cup X^{nb}, E^w \cup E^{nb})$;
  end
until $(E^r = \emptyset)$;
return (F_{pb}, G_w);

citation networks, web networks, qualitative constraint networks, peer-to-peer networks, collaboration networks, networks with ground-truth communities (see Table I).

`Global-hgnc` is a global human protein interactome dataset that was complied from public databases (HPRD, BIND, DIP, IntAct, MINT, BioGRID). `Facebook` is a social network that describes circles (or friends lists) from Facebook where nodes are humans. `Ca-AstroPh` (Astro Physics), `GR-QC` (General Relativity and Quantum Cosmology), and `COND-MAT` (Condense Matter Physics) are three collaboration networks that cover scientific collaborations between authors of papers submitted to Astro Physics category, General Relativity and Quantum Cosmology category, and Condense Matter category respectively. These data are collected from e-print arXiv and cover papers in the period from January 1993 to April 2003 (124 months). `Adm1`, `Gadm1` and `Gag` are three qualitative constraint networks that represent the administrative geography of Great Britain, the German administrative units and the Greek administrative geography, respectively. `Flickr` is a social dataset built by forming links between images sharing common metadata from Flickr. The original images are collected from PASCAL, ImageCLEF, MIR, and NUS-wide. The `DBLP` graph is a co-authorship network where two authors are connected if they publish at least one paper together. This data of computer science bibliography provides a comprehensive list of research papers in computer science. `com-LiveJournal` is a social network that describes free on-line blogging community where users declare friendship each other. LiveJournal also allows users to form a group which other members can then join. `web-google` is an undirected (originally directed) web graph extremely large which comes from different web pages and the links between them on the internet. `cnr-2000` is a web graph from a crawl of the Italian CNR domain. `epinions` is a who-trust-whom online social network of a general consumer review site Epinions.com. Members of the site can decide whether to "trust" each other. This network was originally directed. `gnutella-31` is a peer-to-peer network where a sequence of snapshots of the Gnutella peer-to-peer file sharing network from August 2002. This graph was originally directed. The `Cit-hep-th` and `CoAuth-hep-ph` high-energy physics theory citation graphs are from the e-print arXiv and covers all the citations within a data that covers papers in the period from January 1993 to April 2003 (124 months). These networks were originally directed.
The aforementioned graphs vary in category, size, and type, and are therefore very good candidates for examining the effectiveness of our proposed method.

How does our algorithm $PBnbGS$ summarize real graphs? Is a NB graph a frequently encountered pattern in the structure of a given graph, and if so, what is its size approximately?

Table I shows the results of our algorithm for different real-datasets. Interestingly, our experimental results show that NB graphs occur frequently in real datasets. This fact alone allows us to describe a given graph more succinctly, and then help to understand the main graph characteristics in a simple manner. In Table I, we also observe that in many networks, like `Global-hgnc`, `Facebook`, `Ca-AstroPh`, `Flickr`, `com-LiveJournal`, `Cit-hep-th`, `CoAuth-hep-ph`, and `Amazon_2003_all`, our method attained an impressive coverage of the original graphs with NB graphs. As it can be seen in Table I, there is a significant number of NB graphs in `Flickr`, `DBLP`, `Cit-hep-th`, and `Amazon_2003_all`, among other networks. Moreover, as our experimental results show, for web graphs, especially

web-google network, our algorithm does not allow a high coverage of the original graph. This is because these graphs are very sparse.

From Table I, the CPU time needed to compute all NB graphs is less than 30 minutes; for most graphs whose number of edges is below one million, the computing time is less than one minute.

As a summary, we noticed that NB graphs appear very often in fourteen of real-world graphs (e.g., in citation network, social networks, qualitative networks and collaboration networks). These preliminary results by considering only one graph class (NB graphs) among those described in Section III-C are very promising. As a next step, we plan to consider all the graph classes both individually and in combination. The most important issue is then to design efficient algorithms for their identification.

V. Related Work

During past decades, many algorithms were proposed for graph compression, graph summarization and graph clustering. In this section, we only provide a very brief survey on some of the most related research work. We particularly focus on graph compression and summarization.

Graph compression and graph summarization (e.g. [21], [22]) are active research issues. The goal is twofolds, simplify the graph for better understanding and extract relevant structural knowledge. Several approaches are based on Subdue [23], one of the classical graph compression algorithm. They exploit the Minimum Description Length (MDL) principle to evaluate the interestingness of the mined substructures in term of compression. Among them, one can cite the compression model proposed by Navlakha et al. [22] that provide a highly compact two-part representation of a given graph consisting of a graph summary and a set of corrections. In [21], the authors propose a novel algorithm, called SCMiner, integrating graph summarization, graph clustering, link prediction and the discovery of the hidden structure on the basis of data compression. It proceeds by reduction of a bipartite input graph to a compact representation useful for several data mining tasks. Many other approaches try to discover interesting substructures using clustering and mining, community detection particularly in web graphs and social networks to achieve compression or summarization. In [11], a frequent pattern mining approach to extract meaningful connectivity formations. The authors propose a virtual node miner to achieve graph compression by generating virtual nodes from frequent itemsets in vertex adjacency lists. Motivated by the complexity of social, biological networks and other relations, several contributions have been proposed to deal with more complex graphs (e.g. [24]). Finally, an interesting graph summarization system, called VoG, is proposed by Danai Koutra et al. in [25]. The main ideas behind the proposed system is first to identify a set of subgraph-types that often occur in real graphs (e.g., stars, cliques, chains), find the most succinct description of a graph in terms of this graph classes. This last step is achieved by means of the MDL principle: a subgraph is included in the summary if it decreases the total description length of the graph. This last mentioned approach clearly share the idea of discovering subgraphs classes to achieve better summarization. Our proposed approach have significant difference, in both the considered graph classes, on how they are detected and on the way the graph is summarized (using constraints instead of MDL principle). From this brief description of the related works, our approach can benefit from the use of preprocessing techniques based for example on clustering or community detection to circumscribe our target subgraphs. These kind of approaches might be helpful to overcome the intractability of the subgraph detection problem.

In the second experiment, we compare our $PBnbGS$ compression algorithm to the state of the art VOG tool [25]. Let us recall that VOG is a standard tool for graph summarization. It uses connectivity structures (stars, cliques, etc) represented as special vocabulary of graph primitives to summarize graphs. In Table II we report for each graph (column 1) its original size (column 2), the ratio compression using VOG (column 3) and the ratio compression using $PBnbGS$ approach (column 4). We also consider other graphs. As we can remark, on many graphs, our $PBnbGS$ outperforms VOG (e.g., $Youtube$, $Twitter$, $epinion$, etc.). On the other side, there exists graphs where VOG performs well. More especially $KARATE$, $Com-Google$, etc.
Let us recall that the obtained results are very interesting since our compression approach is limited to nested bipartite graphs.

Let us remark that our new approach even if limited to nested bipartite graph is able to outperform VOG on many graphs e.g., $Youtube$, $Twitter$, $epinion$, etc.

VI. Conclusion

In this paper, we proposed a novel approach for summarizing large graphs. Starting from the observation, that arbitrary graphs might contain hidden connected graph structures that can be represented by some graph classes, our method aims to discover such structures and summarize them using a more compact representation. Our contributions cover several aspects of the summarization process, including (i) the definition of new graph classes that can also be of interest for graph community, (ii) a first encoding of these new and some other well known graph classes as PB constraints, (iii) a new PB constraints based graph summarization algorithm and (iv) an experimental evaluation on real world graphs. In addition to the ability of our approach to summarize large graphs as shown by the experimental evaluation, the results obtained in this paper opens new research opportunities for cross-fertilization between data mining, graph theory and artificial intelligence.

Table I: Summarization results on real-world dataset

Network	nodes/edges	# NB graphs	coverage	min/max size	avg size	CPU time (s)
Global-hgnc [13]	11 120/84 766	1 562	76.04%	2/2 402	40	112.26
Facebook [14]	4 039/88 234	731	81.23%	2/2 876	98	25.31
Ca-AstroPh [15]	18 772/198 110	3 119	81.60%	2/2 180	51	340.93
GR-QC [15]	5 242/14 496	360	51.63%	2/272	20	3.74
COND-MAT [15]	23 133/93 497	2 838	66.27%	2/501	21	225.20
Adm1 [16]	11 761/44 832	1 525	54.66%	2/137	16	42.40
Gadm1 [17]	42 749/159 600	1 459	41.57%	12/4 612	595	166.17
Gag [17]	1 732 999/5 236 270	2 635	65.12%	24/5 200	1 294	303.46
Flickr [18]	105 938/2 316 948	8 084	75.65%	2/52 071	216	4 837.76
DBLP [18]	317 080/1 049 866	8 281	30.88%	2/690	39	5 785.81
com-LiveJournal [18]	3 997 962/34 681 189	4 365	73.15%	43/7 948	1476	3 643.52
web-google [19]	855 802/4 291 352	2 582	15.12%	2/6 831	251	3 166.22
cnr-2000 [20]	325 557/3 216 152	487	37.81%	8/194 103	2 126	417.23
epinions [19]	75 877/405 739	924	26.20%	2/2 334	115	1 387.08
gnutella-31 [19]	62 561/147 878	3 400	30%	2/65	13	804.28
Cit-hep-th [19]	27 400/352 021	9 388	91.52%	2/4 203	34	1 765.15
CoAuth-hep-ph [19]	11 204/117 619	774	72.37%	2/3 213	109	43.97
Amazon_2003_all [19]	473 315/3 505 519	141 148	82.28%	2/1 521	20	9 856.43

Table II: VOG cs PBnbGS

instance	Original Size	VOG (%) Comp. Rate	PBnbGS (%) Comp. Rate
OSM	454.8KB	44.04	28.03
Amazon	12.1MB	4.95	0.83
Cnr-com	41.5MB	39.03	**40.24**
Com-DBLP	13.4MB	19.40	14.92
KARATE	420B	39.28	0.24
Youtube	38.2MB	13.08	**30.36**
Com-email	3.7MB	54.05	51.35
Facebook	47MB	68.08	62.97
Com-Gadm	1.8MB	68.01	11.11
Com-Gag	76.8MB	86.97	84.24
Com-Google	770.3KB	69.14	35.77
Roadnet-ttx	82.2MB	59.24	50.24
Twitter	4MB	65	**75.14**
Yahoo	24.9MB	48.99	**54.61**
Flickr	48.7MB	59.54	39.01
epinion	380.4KB	47.42	**73.71**
Ca-ostrop	207.7KB	25	**27.78**
Ca-thepth	658.6KB	33	**45.41**
Com-lj-com	50.4MB	80	67.46
Toto	3.7MB	32.43	**56.75**
As-skitter	129.1MB	69.01	51.89
Epinion	380.4KB	60.63	47
www.Barbassi	2.4MB	41.66	25
Enron	4MB	32.5	**47.5**
Chocolate	940.3KB	39.14	**64.14**

There are several directions in which this work can be developed further. New graph classes that can be expressed as PB constraints can be characterized. Design of new techniques for discovering subgraphs by exploiting for clustering or community detection approaches. Reasoning on graphs represented as PB formulas is also an important research issue. For example, how to mine graphs, how to reason on graphs and how to solve graph problems using this new representation?

REFERENCES

[1] Y. Cho and A. Zhang, "Predicting protein function by frequent functional association pattern mining in protein interaction networks," *IEEE Trans. on Info. Techno. in Biomed.*, vol. 14, no. 1, pp. 30–36, 2010.

[2] S. Navlakha, M. C. Schatz, and C. Kingsford, "Revealing biological modules via graph summarization," *J. Comput. Biol.*, vol. 16, no. 2, pp. 253–264, 2009.

[3] D. Bu, Y. Zhao, L. Cai, H. Xue, X. Zhu, H. Lu, J. Zhang, S. Sun, L. Ling, N. Zhang, G. Li, and R. Chen, "Topological structure analysis of the protein-protein interaction network in budding yeast," *Nucleic Acids Research*, vol. 31, pp. 2443–2450, 2003.

[4] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999.

[5] G. Tseitin, "On the complexity of derivations in the propositional calculus," in *Structures in Constructives Mathematics and Mathematical Logic, Part II*, 1968, pp. 115–125.

[6] N. Eén and N. Sörensson, "Translating pseudo-boolean constraints into SAT," *JSAT*, vol. 2, no. 1-4, pp. 1–26, 2006.

[7] J. P. Warners, "A linear-time transformation of linear inequalities into conjunctive normal form," *Info. Process. Lett.*, 1996.

[8] R. Asin, R. Nieuwenhuis, A. Oliveras, and E. Rodriguez-Carbonell, "Cardinality networks: a theoretical and empirical study," *Constraints*, vol. 16, no. 2, pp. 195–221, 2011.

[9] I. Abio, R. Nieuwenhuis, A. Oliveras, E. Rodriguez-Carbonell, and V. Mayer-Eichberger, "A new look at bdds for pseudo-boolean constraints." *J. Artif. Intell. Res.*, vol. 45, pp. 443–480, 2012.

[10] "The worst-case time complexity for generating all maximal cliques and computational experiments," *Theoretical Computer Science*, vol. 363, no. 1, pp. 28 – 42, 2006.

[11] G. Buehrer and K. Chellapilla, "A scalable pattern mining approach to web graph compression with communities," in *WSDM*, 2008, pp. 95–106.

[12] C. Karande, K. Chellapilla, and R. Andersen, "Speeding up algorithms on compressed web graphs," *Internet Math.*, vol. 6, no. 3, pp. 373–398, 2009.

[13] G. Scardoni, M. Petterlini, and C. Laudanna, "Analyzing biological network parameters with CentiScaPe," *Bioinformatics*, vol. 25, no. 21, pp. 2857–2859, 2009.

[14] J. J. McAuley and J. Leskovec, "Discovering social circles in ego networks," *TKDD*, vol. 8, no. 1, 2014.

[15] J. Leskovec, J. M. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *TKDD*, vol. 1, no. 1, 2007.

[16] J. Goodwin, C. Dolbear, and G. Hart, "Geographical Linked Data: The Administrative Geography of Great Britain on the Semantic Web," *Transactions in GIS*, vol. 12, pp. 19–30, 2008.

[17] C. Nikolaou and M. Koubarakis, "Fast Consistency Checking of Very Large Real-World RCC-8 Constraint Networks Using Graph Partitioning," in *AAAI*, 2014, pp. 2724–2730.

[18] J. Leskovec and A. Krevl, "SNAP Datasets," http://snap.stanford.edu/data.

[19] http://www.cs.yale.edu/homes/mmahoney/NetworkData/.

[20] P. Boldi and S. Vigna, "Law," http://law.di.unimi.it/.

[21] J. Feng, X. He, B. Konte, C. Böhm, and C. Plant, "Summarization-based mining bipartite graphs," in *KDD*, 2012, pp. 1249–1257.

[22] S. Navlakha, R. Rastogi, and N. Shrivastava, "Graph summarization with bounded error," in *SIGMOD*. ACM, 2008, pp. 419–432.

[23] D. J. Cook and L. B. Holder, "Substructure discovery using minimum description length and background knowledge," *J. Artif. Intell. Res.*, vol. 1, no. 1, pp. 231–255, Feb. 1994.

[24] H. Toivonen, F. Zhou, A. Hartikainen, and A. Hinkka, "Compression of weighted graphs," in *SIGKDD*, 2011, pp. 965–973.

[25] D. Koutra, U. Kang, J. Vreeken, and C. Faloutsos, "VoG: summarizing and understanding large graphs," in *SDM*, 2014, pp. 91–99.

Clustering Complex Data Represented as propositional formulas

Abdelhamid Boudane, Said Jabbour, Lakhdar Sais, and Yakoub Salhi

CRIL-CNRS, Université d'Artois, F-62307 Lens Cedex, France
{boudane,jabbour,sais,salhi}@cril.fr

Abstract. Clustering has been extensively studied to deal with different kinds of data. Usually, datasets are represented as a n-dimensional vector of attributes described by numerical or nominal categorical values. Symbolic data is another concept where the objects are more complex such as intervals, multi-categorical or modal. However, new applications might give rise to even more complex data describing for example customer desires, constraints, and preferences. Such data can be expressed more compactly using logic-based representations. In this paper, we introduce a new clustering framework, where complex objects are described by propositional formulas. First, we extend the two well-known k-means and hierarchical agglomerative clustering techniques. Second, we introduce a new divisive algorithm for clustering objects represented explicitly by sets of models. Finally, we propose a propositional satisfiability based encoding of the problem of clustering propositional formulas without the need for an explicit representation of their models. Preliminary experimental results validating our proposed framework are provided.

1 Introduction

Clustering is a technique used to recover hidden structure in a dataset obtained by grouping data into clusters of similar objects. It is derived by several important applications ranging from scientific data exploration, to information retrieval, and computational biology (e.g. [1]). Such diversity in terms of application domains induces a variety of data types and clustering techniques (see. [2] for a survey). Indeed, data can be transactional, sequential, trees, graphs, texts, or even of a symbolic nature [6, 7, 10]. This last kind of data is particularly suitable for modeling complex and heterogeneous objects usually described by a set of multivalued variables of different types (e.g. intervals, multi-categorical or modal) (e.g. [3, 4, 8]). We can also mention conceptual clustering proposed more than thirty years ago by Michalski [14] and defined as a machine learning task. It accepts a set of object descriptions (events, facts, observations, ...) and produces a classification scheme over them. Conceptual clustering not only partitions the data, but generates clusters that can be summarized by a conceptual description. As a summary, conceptual and symbolic clustering are twoparadigm proposed to deal with kinds of data other than those usually described by numerical values.

In today's data-driven digital era, data might be even more complex and heterogenous. Such complex data might represent customers desires or preferences

collected in different possible ways using surveys and quizzes. As an example, one can cite configuration systems usually designed to provide customized products satisfying the different requirements of the customer, usually modeled by constraints or logic-formulas (e.g. [11]). These customers requirements-data or the data-models provided by the configuration systems are some kind of complex data that we are interested in. These data can be represented by logic-formulas (requirements) or by models (the products satisfying the requirements). Data can also represent more complex entities such as transaction databases. Indeed, suppose that we collected several transaction databases from stores chain selling the same products, one can be interested in determining similar stores (clusters) or stores with the same behavior. This could help the manager of the stores chain to better define its trade policy. In the two previous examples, data can be better represented as a set of propositional formulas or as sets of models.

In this paper, we introduce a new clustering framework, where complex objects are described by propositional formulas. We first extend the two well known k-means and hierarchical agglomerative clustering techniques. Then, we introduce a new divisive algorithm for clustering objects represented explicitly by sets of models. Finally, we propose a propositional satisfiability based encoding of clustering propositional formulas without the need for an explicit representation of their models. Preliminary experimental results validating our proposed framework are provided before concluding.

1.1 Propositional Satisfiability

Let $\mathcal{P}$ be a countably infinite set of propositional variables. The set of *propositional formulas*, denoted $F_{\mathcal{P}}$, is defined inductively starting from $\mathcal{P}$, the constant $\bot$ denoting absurdity, the constant $\top$ denoting true, We use the greek letters ϕ, ψ to represent formulas. A *Boolean interpretation* $\mathcal{I}$ of a formula ϕ is defined as a function from $\mathcal{P}(\phi)$ to $\{0,1\}$ (0 for $false$ and 1 for $true$). A *model* of a formula ϕ is a Boolean interpretation $\mathcal{I}$ that satisfies ϕ (written $\mathcal{I} \vDash \phi$), i.e. $\mathcal{I}(\phi) = 1$. We denote the set of models of ϕ by $\mathcal{M}(\phi)$. A formula ϕ is satisfiable (or consistent) if there exists a model of ϕ; otherwise it is called unsatisfiable (or inconsistent).

Let ϕ and ψ be two propositional formulas, we say that ψ is a logical consequence of ϕ, written $\phi \vDash \psi$, iff $\mathcal{M}(\phi) \subseteq \mathcal{M}(\psi)$. The two formulas ϕ and ψ are called equivalent iff $\phi \vDash \psi$ and $\psi \vDash \phi$, i.e $\mathcal{M}(\phi) = \mathcal{M}(\psi)$.

A CNF formula is a conjunction ($\wedge$) of clauses, where a *clause* is a disjunction ($\vee$) of literals. A *literal* is a propositional variable (p), called positive literal, or ($\neg p$), called negative literal. The *SAT problem* consists in deciding whether a given CNF formula admits a model or not. Another problem related to SAT is the SAT model enumeration problem. Enumeration requires generating all models of a problem instance without duplicates. Models enumeration is related to #SAT, the problem of computing the number of models for a given propositional formula. Model counting is the canonical #P-complete problem. On the practical side, for model counting, *SampleCount* a sampling based approach proposed by Gomes et al in [9], provides very good lower bounds with high confidence. Similarly, an efficient model enumeration algorithm has been proposed in [12, 5].

2 Motivating Example

To motivate our proposed framework, let us consider a simple example of a car dealer selling different cars bands with several possible options. For each car brand, several colors and types of fuels are available. The car dealer collected the preferences of four customers through a survey questionnaire. The first customer does not want red cars. The second wants a car with a diesel fuel, while the third wants a red car with gasoline fuel. Finally, the fourth customer prefers brand Peugeot cars. In addition to these customer desires, we also consider mutual exclusion constraints (mutex), allowing to express that each car must have only one color, one type of fuel and one car brand.

To express the different customer desires in propositional logic, we consider the following propositional variables: r (resp. b) represents red (resp. black) colors, p (resp. c) represents the Peugeot (resp. Citroen) car brand and d (resp. g) represents cars with diesel (resp. gasoline) fuel.

The mutex constraints are expressed by the following formula: $\mu = [(r \wedge \neg b) \vee (b \wedge \neg r)] \wedge [(g \wedge \neg d) \vee (d \wedge \neg g)] \wedge [(p \wedge \neg c) \vee (c \wedge \neg p)]$.

In Figure 1 (left hand side), for each customer c_i, we associate a propositional formula ϕ_{c_i} expressing its desires. We also provide the set of models satisfying both the desires of the customer and the mutex constraints ($\mathcal{M}(\phi_{c_i} \wedge \mu)$). The presentation of the models follows the variables ordering: $r \prec b \prec d \prec g \prec c \prec p$. In Figure 1 (right hand side), we give a graphical representation of the preferences of the four customers. This illustrative example highlights the expressiveness of

Customers	c_1	c_2	c_3	c_4
ϕ_{c_i}	$\neg r$	d	$g \wedge r$	p
$\mathcal{M}(\phi_{c_i} \wedge \mu)$	010101	011001		010101
	010110	011010		011001
	011001	101001	100101	100101
	011010	101010	100110	101001

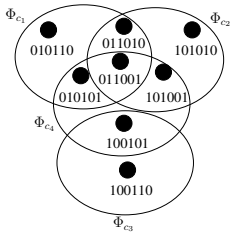


Fig. 1. Logical and Graphical Representation of Customers Preferences

logic-based data representation while allowing the possibility to define both user and background constraints.

3 Adapting Standard Clustering Algorithms

In this section, we present our extension of the well-known k-means and agglomerative hierarchical clustering algorithms to handle objects expressed as propositional formulas. Let us first fix some necessary notations and definitions.

We use $\mathcal{P}(k, \Phi)$ to denote the problem of clustering the set of propositional formulas $\Phi = \{\phi_1, \ldots, \phi_n\}$ into a set of k clusters with $k \leqslant n$. Let $\mathcal{C}$ be a family of sets over Φ. $\mathcal{C}$ is a solution of $\mathcal{P}(k, \Phi)$ if and only if $|\mathcal{C}| = k$, $\bigcup_{\mathcal{C}_i \in \mathcal{C}} \mathcal{C}_i = \Phi$ with $C_i \cap C_j = \emptyset$ for $1 \leqslant i < j \leqslant k$, and $\mathcal{M}(\bigwedge_{\phi \in \mathcal{C}_i} \phi) \neq \emptyset$ for every $\mathcal{C}_i \in \mathcal{C}$. We say that a clustering problem $\mathcal{P}(k, \Phi)$ is *consistent* if it admits a solution.

3.1 *k*-Means Algorithm for propositional formulas Clustering

Given a set of n data points in d-dimensional space $\mathbb{R}^d$ and a positive integer k, the k-means algorithm determines a set of k points in $\mathbb{R}^d$, called centers, so as to minimize an objective function such as the mean squared distance from each data point to its nearest center. To extend the k-means algorithm to clustering of objects described by propositional formulas, we need to define,

1. a distance between two formulas;
2. a centroid representing a given cluster;
3. an objective function to optimize.

Let us recall that a propositional formula ϕ can be equivalently expressed by its set of models $\mathcal{M}(\phi)$. With this representation in mind, one can consider that two formula ϕ_1 and ϕ_2 are similar if their set of common models $\mathcal{M}(\phi_1) \cap \mathcal{M}(\phi_2)$ is higher with respect to the remaining (distinctive) models $\mathcal{M}(\phi_1) \setminus \mathcal{M}(\phi_2) \cup \mathcal{M}(\phi_2) \setminus \mathcal{M}(\phi_1)$. This kind of similarity is related to the well-known contrast model of similarity proposed in a seminal paper by Tversky [15].

Definition 1 (Tversky [15]). *Let a and b be two objects described by two sets of features A and B respectively. Similarity between a and b, denoted $s(a, b)$, is defined as:*

$$s\,(a, b) = \frac{f\,(A \cap B)}{f\,(A \cap B) + \alpha f\,(A - B) + \beta f\,(B - A)} \quad \alpha, \beta \geqslant 0$$

The positive coefficients α and β reflects the weights given to the distinctive features of the two objects a and b. We usually assume that f is a matching function satisfying the additivity property $f(A \cup B) = f(A) + f(B)$, whenever A and B are disjoint. The ratio model defines a normalized value of similarity such that $0 \leqslant s(a, b) \leqslant 1$.

Contrast similarity model is particularly suitable in our context. To extend Definition 1, we consider the relationship between set operations and logical connectives. Indeed, the set union (resp. intersection) corresponds to disjunction (resp. conjunction). The difference between sets can be expressed using both conjunction and negation connectives, while the symmetric difference between sets can be expressed using the xor ($\oplus$) logical connective. Indeed, we have $\mathcal{M}(\phi_1) \setminus \mathcal{M}(\phi_2) \cup \mathcal{M}(\phi_2) \setminus \mathcal{M}(\phi_1) = \mathcal{M}((\phi_1 \wedge \neg\phi_2) \vee (\phi_2 \wedge \neg\phi_1)) = \mathcal{M}(\phi_1 \oplus \phi_2)$.

Using these relationships, we derive the following extension of the ratio model[16].

Definition 2. *Let a and b be two objects described by two propositional formulas ϕ_1 and ϕ_2 respectively. Similarity between a and b is defined as:*

$$s(a,b) = \frac{f(\phi_1 \wedge \phi_2)}{f(\phi_1 \wedge \phi_2) + \alpha f(\phi_1 \wedge \neg\phi_2) + \beta f(\phi_2 \wedge \neg\phi_1)} \quad \alpha, \beta \geqslant 0$$

In our context, as no distinction is made between the measure of $\phi_1 \wedge \neg\phi_2$ and $\phi_2 \wedge \neg\phi_1$, we derive the following similarity measure.

Definition 3. *Let a and b be two objects described by two propositional formulas ϕ_1 and ϕ_2 respectively. Similarity between a and b is defined as:*

$$s(a,b) = \frac{f(\phi_1 \wedge \phi_2)}{f(\phi_1 \wedge \phi_2) + \gamma f(\phi_1 \oplus \phi_2)}, \gamma \geqslant 0$$

From Definition 2 (resp. Definition 3), instantiating $\alpha = \beta = 1$ (resp. $\gamma = 1$), we derive a logic-based variant of the well known Jaccard similarity coefficient (resp. distance) [13]:

Definition 4. *Let a and b be two objects described by two propositional formulas ϕ_1 and ϕ_2 respectively. Similarity and distance between a and b or between ϕ_1 and ϕ_2 are defined respectively as:*

$$s_J(a,b) = s_J(\phi_1,\phi_2) = \frac{f(\phi_1 \wedge \phi_2)}{f(\phi_1 \vee \phi_2)} \text{ and } d_J(a,b) = 1 - s_J(a,b) = d_J(\phi_1,\phi_2)$$

As mentioned previously, considering the model based representation of propositional formulas, we define the function f as:

$$f : \left| \begin{array}{l} F_{\mathcal{P}} \longrightarrow \mathbf{N} \\ \phi \longmapsto |\mathcal{M}(\phi)| \end{array} \right.$$

Clearly, the function f satisfies the additive property. Indeed, we have $\mathcal{M}(\phi_1 \vee \phi_2) = \mathcal{M}(\phi_1) \cup \mathcal{M}(\phi_2)$. Computing f involves solving a #P-Complete model counting problem as discussed in Section 1.1.

Let us now define the representative of a cluster of propositional formulas.

Definition 5. *Let $\mathcal{C}_i$ be a cluster involving n_i formulas $\{\phi_{1_i}, \phi_{2_i}, \ldots, \phi_{n_i}\}$. We define the cluster representative (also called centroid) $\mathcal{O}_{\mathcal{C}_i}$ of the cluster $\mathcal{C}_i$ as:*

$$\mathcal{O}_{\mathcal{C}_i} = \phi_{1_i} \wedge \phi_{2_i} \wedge \ldots \wedge \phi_{n_i}$$

It is important to note that in our proposed extension, the goal is to group formulas into consistent clusters. Consequently, the *formula representing a given cluster must be consistent.*

We use the classical k-means objective function introduced in Definition 6

Definition 6. *Let $\mathcal{P}(k,\Phi)$ be the problem of clustering a set of propositional formulas $\Phi = \{\phi_1, \ldots, \phi_n\}$ to $k(\leqslant n)$ clusters $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$. The objective function is defined using Absolute-Error Criterion (AEC):*

$$C^* = \arg\min_{\mathcal{C}} \sum_{i=1}^{k} \sum_{\phi \in \mathcal{C}_i} d_J(\phi, \mathcal{O}_{\mathcal{C}_i}) \tag{1}$$

Our clustering algorithm of a set of propositional formulas can now be derived from the classical k-means algorithm using the new components (distance, centroid and objective function) defined above.

3.2 Hierarchical Agglomerative Algorithm for propositional formulas Clustering

Hierarchical algorithms can behave better than the k-means. The base idea of hierarchical agglomerative algorithms is to build a dendrogram such that at each level the two closest clusters are merged. By applying a hierarchical algorithm, we will ensure that if there are two objects that are closest to each other, they will necessarily be in the same cluster. In this adaptation, the similarity between two clusters is identical to the similarity between their representatives. Similarly to Definition 5, the conjunction of all formulas in a cluster represents its centroid. To merge clusters, we combine the two clusters with the smallest centroid distance. Using this adaptation, we can applay a standard hierarchical agglomerative algorithm on data represented as boolean formulas as illustrated in figure 2.Note that this algorithm needs at least $\mathcal{O}(n^2)$ calls to a # SAT oracle.

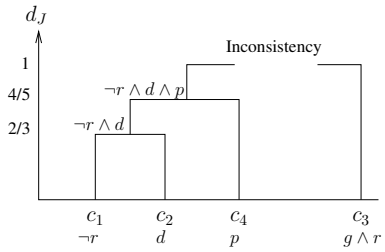


Fig. 2. Agglomerative Clustering on the Car Dealer Example

4 Divisive Algorithm for Model Based Representation

As mentioned previously, when we consider the problem of clustering a set of formulas $\Phi = \{\phi_1, \phi_1, \ldots, \phi_n\}$ without common model, i.e., $\Phi \vdash \bot$, agglomerative algorithm and k-means can fail to find a clustering with the desired number of clusters. In the sequel, we propose a top-down hierarchical (or divisive) algorithm for clustering a set of propositional formulas. Our proposed adaptation makes use of the well-known minimum hitting sets problem, that we recall.

Definition 7. *H is a hitting set of a set of sets Ω if $\forall S \in \Omega$, $H \cap S \neq \emptyset$. A hitting set H is irreducible if there is no other hitting set H' s.t $H' \subset H$. H is called minimum hitting set if there is no hitting set H' such that $|H'| < |H|$.*

Example 1. Let $\Phi = \{\phi_1, \phi_2, \phi_3\}$ be a set of propositional formulas such that $\mathcal{M}(\phi_1) = \{m_1, m_2, m_3\}$, $\mathcal{M}(\phi_2) = \{m_1, m_4\}$ and $\mathcal{M}(\phi_3) = \{m_3, m_5\}$. The set $H = \{m_1, m_3\}$ is a minimum and irreducible hitting set of the models of Φ.

In our adaptation, we choose the worst cluster to divide according to the following quality measure.

Definition 8. *Let $\mathcal{C}_i = \{\phi_{1_i}, \ldots, \phi_{n_i}\}$ be a cluster of n_i propositional formulas. We define the quality of $\mathcal{C}_i$ as:*

$$\mathcal{Q}(\mathcal{C}_i) = \frac{|\mathcal{M}(\phi_{1_i} \wedge \cdots \wedge \phi_{n_i})|}{|\mathcal{M}(\phi_{1_i} \vee \cdots \vee \phi_{n_i})|}$$

The quality of a cluster is obtained by extending the similarity measure between two formulas to a set of formulas. Indeed, a cluster is qualified to be of poor quality, when its formulas admits a great number of models while sharing a small number of models. Consequently, the worst cluster is obtained as follows:

$$\mathcal{C}_i^* = \underset{\mathcal{C}_i \in \mathcal{C}}{\operatorname{argmin}}\ \mathcal{Q}(\mathcal{C}_i)$$

Definition 9. *Let Φ be a set of propositional formulas and $\mathcal{I}$ a Boolean interpretation. We define the subset of formulas of Φ sharing the model $\mathcal{I}$ as* $\mathcal{S}(\mathcal{I}, \Phi) = \{\phi \in \Phi \mid \mathcal{I} \vDash \phi\}$

To build consistent clusters, Algorithm 1 starts by computing a minimum hitting set H of the set of sets of models of the formulas in Φ (line 1). The main idea behind our algorithm is to use the models of the computed minimum hitting set to divide a cluster into several consistent clusters. Each cluster is obtained by selecting for each model m of the minimum hitting set, the set of formulas admitting m as a model. In this way, the formulas in the obtained clusters share at least one model. If the size of the minimum hitting set H is greater than k, then no clustering is possible, and the algorithm returns an empty set (line 3), otherwise a consistent clustering can be obtained. In this last case, the algorithm starts by a clustering $\mathcal{C}$ where all the formulas in Φ are grouped into a single cluster (line 6). We start an iterative top-down divisive process (lines 7-20), until generating k clusters. At each iteration, we choose a cluster to divide (line 8) which is one of those with the worst quality (see Definition 8). Then, we build Ω the set of sets of models of the formulas involved in the selected cluster, while removing the set of common models M (lines 9-10). A minimum hitting set H of Ω is then computed (line 11). It is important to note that by removing the common models M from the models of each formula of the selected cluster, we avoid the trivial minimum hitting sets of size 1. Now, we use the hitting set H to divide the chosen cluster $\mathcal{C}_i^*$ into $|H|$ clusters (line 12). Indeed, for each model m in H, we associate a cluster Ψ_m made of formulas of $\mathcal{C}_i^*$ sharing the model m. In this way, we maintain the consistency property on each new cluster Ψ_m. Now, we substitute in $\mathcal{C}$ the cluster of poor quality $\mathcal{C}_i^*$ with the new set of clusters (line 18). However, this is only done when the size of the new

Algorithm 1: Model-Based Divisive Algorithm for Clustering Boolean Formulas

Input: A set of formulas $\Phi = \{\phi_1, \ldots, \phi_n\}$ and an integer $k \geqslant 1$
Output: A set of clusters $\mathcal{C} = \{C_1, \ldots, C_k\}$

1 $H \leftarrow minHittingSet(\{\mathcal{M}(\phi_1), \ldots, \mathcal{M}(\phi_n)\})$;
2 **if** $(|H| > k)$ **then**
3 | **return** $\emptyset$;
4 **end**
5 **else**
6 | $\mathcal{C} \leftarrow \{\Phi\}$;
7 | **while** $(|\mathcal{C}| \; != k)$ **do**
8 | | $\mathcal{C}_i^* = \{\phi_{i_1} \ldots \phi_{i_{n_i}}\} \leftarrow \underset{\mathcal{C}_i \in \mathcal{C}, |\mathcal{C}_i|>1}{\arg\min} \mathcal{Q}(\mathcal{C}_i)$, $\triangleright \; n_i = |\mathcal{C}_i^*|$;
9 | | $M = \mathcal{M}(\phi_{i_1}) \cap \cdots \cap \mathcal{M}(\phi_{i_{n_i}})$;
10 | | $\Omega = \{\mathcal{M}(\phi_{i_1}) \setminus M, \ldots, \mathcal{M}(\phi_{i_{n_i}}) \setminus M\}$;
11 | | $H \leftarrow minHittingSet(\Omega)$;
12 | | $\forall m \in H$, $\Psi_m \leftarrow \mathcal{S}(m, \mathcal{C}_i^*)$;
13 | | **if** $(|\mathcal{C}| + |H| - 1 > k)$ **then**
14 | | | $\Psi \leftarrow merge(\{\Psi_{m_1}, \ldots, \Psi_{m_{|\mathcal{C}|+|H|-1-k}}\})$;
15 | | | $\mathcal{C} \leftarrow (\mathcal{C} \setminus \mathcal{C}_i^*) \cup \{\Psi\} \cup \{\Psi_{m_{|\mathcal{C}|+|H|-k}}, \ldots, \Psi_{m_{|H|}}\}$
16 | | **end**
17 | | **else**
18 | | | $\mathcal{C} \leftarrow (\mathcal{C} \setminus \mathcal{C}_i^*) \cup \{\Psi_{m_1}, \ldots, \Psi_{m_{|H|}}\}$
19 | | **end**
20 | **end**
21 **end**
22 $\mathcal{C} \leftarrow eliminateOverlap(\mathcal{C})$;
23 **return** $\mathcal{C}$

clustering does not exceed k (line 13); otherwise to obtain exactly k clusters, we merge (function `merge`) the first $|\mathcal{C}| + |H| - (k+1)$ of these new clusters (line 14) before applying substitution (line 15). Note that in the divisive step (line 12), a formula can belong to several new clusters. The reason comes from the fact that a given formula can share several models of the minimum hitting set. Consequently, a last step is then performed to produce non overlapping clusters (line 20 - function *eliminateOverlap*). To do this, for each formula occurring in several clusters, we keep it in the cluster with the best quality, while removing it in the remaining clusters. Obviously, depending on applications, overlapping clusters might be more suitable. In this case, one only need to skip the call to the overlap elimination function.

Algorithm 1, involves $\mathcal{O}(n)$ calls to model enumeration problem (line 1), $\mathcal{O}(k)$ calls to # SAT oracle (line 8) and $\mathcal{O}(k)$ calls to minimum hitting set problem (line 1 and 11).

Let us now gives some interesting properties of our propositional formulas based divisive algorithm. The first one states the correctness of our algorithm.

Proposition 1. *If $\mathcal{P}(k, \Phi)$ is consistent, then Algorithm 1 produces a clustering.*

The proof trivially follows from the previous detailed explanation on how the algorithm operates.

The second property allows us to establish that two equivalent formulas might be located in the same cluster when overlaps between clusters are allowed.

Proposition 2. *Let $\mathcal{P}(k, \Phi)$ be a clustering problem with overlaps, $\mathcal{C}$ a clustering of $\mathcal{P}(k, \Phi)$ and $\phi_1, \phi_2 \in \Phi$. If $\phi_1 \equiv \phi_2$ then $\forall \mathcal{C}_i \in \mathcal{C}$, $\phi_1 \in \mathcal{C}_i$ iff $\phi_2 \in \mathcal{C}_i$.*

The last property generalizes the previous property to the case of two formulas where one is a logical consequence of the other.

Proposition 3. *Let $\mathcal{P}(k, \Phi)$ be a clustering problem with overlaps, $\mathcal{C}$ a clustering of $\mathcal{P}(k, \Phi)$ and $\phi_1, \phi_2 \in \Phi$. If $\phi_1 \vdash \phi_2$ then $\forall \mathcal{C}_i \in \mathcal{C}$, if $\phi_1 \in \mathcal{C}_i$ then $\phi_2 \in \mathcal{C}_i$.*

5 SAT encoding for a Bounded Consistent Clustering

As discussed in the previous section, when the propositional formulas are not represented by their models, our proposed model based divisive algorithm requires $\mathcal{O}(n)$ calls to model enumeration oracle, to compute the set of models of each formula. Such set of models might be of exponential size in the worst case. In addition to these limitations, one also need to compute a minimum hitting set of a set of sets of models ($\mathcal{O}(k)$ calls). In this section, we present an alternative approach that significantly reduces the overall complexity of our Algorithm. To this end, we introduce a SAT-based encoding that allows to find a bounded consistent clustering of a given set of propositional formulas.

Let $\Phi = \{\phi_1, \ldots, \phi_n\}$ be a set of propositional formulas and k a positive integer. To define our encoding, we associate to each propositional variable p appearing in Φ a set of k fresh propositional variables, denoted $p^1, \ldots, p^k$. Then, for every formula $\phi_i \in \Phi$ and $j \in \{1, \ldots, k\}$, we use ϕ_i^j to denote the formula obtained from ϕ_i by replacing each propositional variable p with the fresh variable p^j. The formula ϕ_i^j is used to model the fact that ϕ_i is in the j^{th} cluster.

The following formula expresses that each formula in Φ has to be true in at least one consistent cluster:

$$\bigwedge_{i=1}^{n} \left(\bigvee_{j=1}^{k} \phi_i^j \right) \tag{2}$$

One can easily see that (2) is satisfiable if and only if Φ can be partitioned in k consistent clusters. It is worth noting that in a model of (2) a formula can belong to more than one cluster. To obtain a bounded consistent clustering from a model m, we only have to consider for each formula $\phi_i \in \Phi$ a single positive integer j in the set $\{1 \leqslant j \leqslant k \mid m(\phi_i^j) = 1\}$. This problem can be avoided by reformulation. To this end, we associate to each formula ϕ_i in Φ a set of k fresh propositional variables, denoted $q_{\phi_i}^1, \ldots, q_{\phi_i}^k$. The variable $q_{\phi_i}^j$ is used to represent the fact that ϕ_i is in the j^{th} cluster by using the following formula:

$$\bigwedge_{i=1}^{n} \left(\bigwedge_{j=1}^{k} q_{\phi_i}^j \Leftrightarrow \phi_i^j \right) \tag{3}$$

Then, to express that each formula in Φ belongs to exactly one consistent cluster, we use the following formula:

$$\bigwedge_{i=1}^{n}(\sum_{j=1}^{k} q_{\phi_i}^{j} = 1) \tag{4}$$

Our second SAT encoding of the bounded consistent clustering problem $\mathcal{P}(k, \Phi)$ is defined by the formula $\mathcal{P}_{SAT}(k, \Phi) = (3) \wedge (4)$. From a model m of $\mathcal{P}_{SAT}(k, \Phi)$, a clustering can be easily extracted. Indeed, if $m(q_{\phi_i}^{j}) = true$ then $\phi_i \in \mathcal{C}_j$ otherwise $\phi_i \notin \mathcal{C}_j$.

Definition 10. *Let $\Phi = \{\phi_1, \ldots, \phi_n\}$. C is called a minimum consistent clustering of Φ if there is no consistent clustering $\mathcal{C}'$ of Φ such that $|\mathcal{C}'| < |\mathcal{C}|$.*

As we can observe, clustering propositional formulas can be done using Algorithm 1 by replacing the computation of the minimum hitting set with the computation of the minimum consistent clustering (Definition 10) using $\mathcal{P}_{SAT}(k, \Phi)$. Similarly to Algorithm 1, Properties 1, 2 and 3 holds.

6 Experimentation

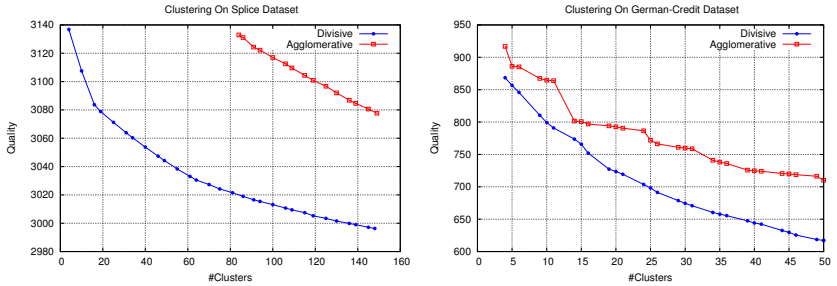
In this section, we carried out an experimental evaluation of the performance of our divisive and agglomerative algorithms for the clustering of a set of propositional formulas. Our goal is to assess the feasibility and effectiveness of our proposed framework.

We performed our experiments on a machine with Intel Core2 Quad CPU of 2.66GHz and 8G of RAM. Our first aim is to compare the performance of our divisive and agglomerative algorithms. To this end, We consider two datasets `splice`, and `german-credit`[1]. We consider each data set as a set of transactions, where each transaction is a formula (a set of models). Consequently, an item is assimilated to a model.

Figure 3 shows the performances of agglomerative (Algorithm **??**) and divisive (Algorithm 1) methods on the problem of clustering transaction databases. First, our divisive algorithm outperforms the agglomerative algorithm on `splice` and `german-credit`. Nevertheless, as illustrated in section 3.2, the agglomerative algorithm is unable to find a clustering all the time. This is the case on `splice` data, where such approach can not provide clustering answer when the number of desired clusters is less than 84.

To further investigate the expressiveness and the ability of our approach to scale, we enlarge our experiments of the previous problem by studying the clustering of a set of formulas resulting from a random-generated poll with 100 to 1000 participants where each participant is invited to report its preferences. The questions of the poll are organized in four levels. At the first level, the participant is invited to select its 3 preferred options among 5. According to the

[1] https://dtai.cs.kuleuven.be/CP4IM/



splice german-credit

Fig. 3. Model approach: Agglomerative vs Divisive

preferences of the participant, she/he is invited to select other preferences from the second level and so on until the last level (level 4). For illustration, assume that in the first level we consider a set S of courses (e.g. Artificial Intelligence, Data Mining, Databases, Networks and Web Programming). A student selects three courses from S (level 1). Then, for each selected course, she/he chooses chapters (level 2), and so on. The preferences of each participant are encoded as a propositional formula (the resulting formulas have between 567 and 1813 models). Agglomerative approach is not considered since it can not guaranty to find a clustering solution if it exists.

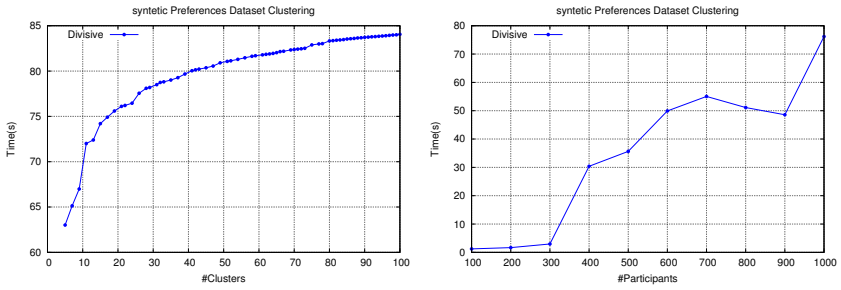


Fig. 4. Time vs #Clusters vs #Participants

The time needed to obtain a clustering, Figure 4, does not exceed 100 seconds for all values of k. This shows that our approach scale well. Finally, we study the evolution of the time needed to find a clustering when the number of clusters is fixed to 20 and the number of participants is varied from 100 to 1000 (Figure 4). Here again the time needed is reasonable, i.e., less than 100 seconds.

7 Conclusion et perspectives

In this work we introduced the concept of consistent clustering propositional formulas. We show how well-known k-means, agglomerative and divisive algorithms

can be adapted to this new framework. We then, propose two new solutions. The first one called model based, assume that the set of models of each formula are given. We then show how the hitting set notion is used to efficiently give a consistent clustering. In the second part, we propose an encoding into SAT of the divisive algorithm that make a linear number of calls to a #SAT oracle to count the set of models during the clustering steps. As a future work, we plan to explore other similarity measure, to define intuitive distance between propositional formulas. Improving our divisive algorithm by exploiting efficiently the overlaps deserves further investigation.

References

1. C. C. Aggarwal and C. K. Reddy. *Data clustering: algorithms and applications*. CRC Press, 2013.
2. P. Berkhin. A survey of clustering data mining techniques. In J. Kogan, C. K. Nicholas, and M. Teboulle, editors, *Grouping Multidimensional Data - Recent Advances in Clustering*, pages 25–71. Springer, 2006.
3. L. Billard and E. Diday. *Symbolic Data Analysis: Conceptual Statistics and Data Mining*. John Wiley & Sons, May 2012.
4. H. H. Bock. *Analysis of Symbolic Data: Exploratory Methods for Extracting Statistical Information from Complex Data*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2000.
5. S. Chakraborty, K. Meel, and M. Vardi. A scalable approximate model counter. In *CP'2013)*, pages 200–216, 2013.
6. F. d. A. de Carvalho, M. Csernel, and Y. Lechevallier. Clustering constrained symbolic data. *Pattern Recognition Letters*, 30(11):1037–1045, 2009.
7. R. M. de Souza and F. d. A. De Carvalho. Clustering of interval data based on city–block distances. *Pattern Recognition Letters*, 25(3):353–365, 2004.
8. E. Diday and F. Esposito. An introduction to symbolic data analysis and the SODAS software. *Intell. Data Anal.*, 7(6):583–601, 2003.
9. C. P. Gomes, J. Hoffmann, A. Sabharwal, and B. Selman. From sampling to model counting. In *IJCAI'1997*, pages 2293–2299, 2007.
10. K. C. Gowda and E. Diday. *New Approaches in Classification and Data Analysis*, chapter Symbolic Clustering Algorithms using Similarity and Dissimilarity Measures, pages 414–422. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.
11. L. Hotz, A. Felfernig, M. Stumptner, A. Ryabokon, C. Bagley, and K. Wolter. Chapter 6 - configuration knowledge representation and reasoning. In *Knowledge-Based Configuration*, pages 41 – 72. Morgan Kaufmann, 2014.
12. S. Jabbour, J. Lonlac, L. Sais, and Y. Salhi. Extending modern SAT solvers for models enumeration. In *IEEE-IRI'2014*, pages 803–810, 2014.
13. P. Jaccard. The distribution of the flora of the alpine zon. *New Phytologist*, 11:37–50, 1912.
14. R. S. Michalski. Knowledge acquisition through conceptual clustering: A theoretical framework and an algorithm for partitioning data into conjunctive concepts. *Journal of Policy Analysis and Information Systems*, 4(3):219–244, 1980.
15. A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.
16. A. Tversky. *Preference, Belief, and Similarity*. The MIT Press, November 2003.

A SAT-based Framework for Overlapping Community Detection in Networks

Abstract. In this paper, we propose a new approach to the classic problem of detecting overlapping community in large complex networks. We first introduce a parametrized notion of communities, called *k-linked community*, allowing us to characterize node/edge centered k-linked community with bounded diameter. Next, we show how the problem of centered k-linked community detection can be expressed as a partial Max-SAT optimization problem. Last, we propose a post-processing strategy to limit the overlaps between communities. An extensive experimental evaluation on real-world networks shows that our approach outperforms several popular algorithms for community detection. Interestingly, when the diameter upper bound is set to four, our proposed approach outperforms the existing ones.

1 Introduction

Many complex interactions can be represented by networks, which are set of nodes connected by edges. Such connections might represent different type of relations between individuals or entities. Nodes in networks can be organized into *communities*, which often correspond to groups of nodes that share common properties, roles or functionnalities, such as functionally related proteins, social communities, or topically related webpages.

One of the most important tasks when studying networks is that of identifying communities. Indeed, detecting and analyzing communities is of great interest in several concrete applications, including clustering web clients who have similar interests, identifying clusters of customers in the network of customers-products purchase relationships of online retailers (e.g. Amazon), etc. However, designing an efficient algorithm for discovering communities in complex networks remains highly non-trivial for many reasons. For example, the most popular algorithm based on non-negative matrix factorisation [12] and spectral clustering methods [17] depend on a set of parameters such as the number of expected communities. Other approaches based on the edge betweenness [8] involve the computation of the shortest paths between pairs of nodes. So, the major motivation of this paper is to develop an algorithm which does not require any knowledge about the number of expected communities or the original division of the communities and discover the community structure in a reasonable amount of time.

To this end, we first introduce a parametrized notion of communities, called k-linked community, allowing us to characterize node/edge centered k-linked community with bounded diameter. We then show how to use off-the-shelf satisfiability solvers for community discovering using an appropriate encoding of the

centered k-linked community detection task as a partial maximum satisfiability (Partial Max-SAT) optimisation problem. Our proposed framework allows us to benefit from the recent advances in propositional satisfiability and its optimisation variants. Finally, we propose a post-processing strategy to limit the overlaps between communities. Our proposed framework, follows the recent data mining research trend exploiting two powerful declarative models, namely constraint programming and propositional satisfiability. Indeed, several data mining tasks including pattern mining [10] and clustering [7] have been modeled and solved using these two well-known declarative and flexible models.

2 Formal Preliminaries

2.1 Propositional Logic and SAT Problem

Let $\mathcal{L}$ be a propositional language defined inductively from a finite set $\mathcal{PS}$ of propositional symbols, the boolean constants $\top$ (*true* or 1) and $\bot$ (*false* or 0) and the standard logical connectives $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ in the usual way. We use the letters x, y, z, etc. to range over the elements of $\mathcal{PS}$. Formulas of $\mathcal{L}$ are denoted by A, B, C, etc. A *literal* is a propositional variable (x) of $\mathcal{PS}$ or the negation of a variable ($\neg x$). The two literals x and $\neg x$ are called complementary. A *clause* is a (finite) disjunction of literals, i.e., $a_1 \vee \ldots \vee a_n$. For every propositional formula $\mathcal{A}$ from $\mathcal{L}$, $\mathcal{P}(\mathcal{A})$ denotes the symbols of $\mathcal{PS}$ occurring in $\mathcal{A}$.

A *Boolean interpretation* $\mathcal{I}$ of a formula $\mathcal{A}$ is a truth assignement of $\mathcal{PS}$, that is, a total function from $\mathcal{P}(\mathcal{A})$ to $\{0, 1\}$. A *model* of a formula $\mathcal{A}$ is a Boolean interpretation $\mathcal{I}$ that satisfies $\mathcal{A}$, i.e. $\mathcal{I}(\mathcal{A}) = 1$. A formula $\mathcal{A}$ is satisfiable if there exists a model of $\mathcal{A}$. We denote by $\mathcal{M}(\mathcal{A})$ is the set of all models of $\mathcal{A}$.

As usual, every finite set of formulas is considered as the conjunctive formula whose conjuncts are the elements of the set. A formula in *conjunctive normal form* (CNF) is a (finite) conjunction of clauses. The SAT problem consists in deciding wether a given CNF formula admits a model or not. This well-known NP-Complete problem has seen spectacular progress these recent years.

SAT has seen many successful applications in various fields such as electronic design automation, debugging of hardware designs, artificial intelligence, and data mining. Several SAT extensions have been proposed to deal with optimisation problems. For example, the Max-SAT Problem seeks the maximum number of clauses that can be satisfied. In this paper, we consider one of these optimisation variants referred to as Partial Max-SAT problem. Partial Max-SAT sits between SAT and Max-SAT problems. While SAT requires all clauses to be satisfied, Partial Max-SAT relaxes this requirement by considering two kind of clauses, hard and soft. Partial MaxSAT is the problem of finding an optimal assignment to the variables that satisfies all the hard clauses, while satisfying the maximum number of soft clauses.

2.2 Overlapping Community Detection

In this subsection, we discuss the classic problem of detecting overlapping community structure in networks.

A network is an undirected graph $\mathcal{N} = (V, E)$ where V is a set of nodes and $E \subseteq V \times V$ is a set of edges. We denote by n (respectively m) the number of nodes (respectively edges) in $\mathcal{N}$. The *degree* of a node $u \in V$, denoted d_u, is the number of edges connected to it. The length of the shortest path between two nodes $u, v \in V$ is called the *distance* between the nodes, noted $dist(u, v)$. Given an edge $e = (u, v) \in E$ and a node $w \in V$, the distance between e and w is defined as $dist(e, w) = \min\{dist(u, w), dist(v, w)\}$. In graph theory, a *community* is described as a set of nodes densely connected internally. In real-world networks, nodes are organized into densely linked sets of nodes that are commonly referred to as *network communities*, clusters or modules. Notice that communities in networks often overlap as nodes can belong to multiple communities at once. Network *overlapping community detection* problem consists in dividing a network of interest into (overlapping) communities for intelligent analysis. It has recently attracted significant attention in diverse application domains. Identifying the community structure is crucial for understanding structural properties of the real-world networks. Various methods have been proposed to identify the community structure of complex networks (see [6, 15] for an overview).

Quality Metrics:

Several measures have been proposed for quantifying the quality of communities in networks (see [13] for a comparative study of quality measures). In this paper, we adopt two well-known metrics to assess the performance of our method:

Modularity. The most widely used metric for measuring the quality of network's partition into communities (without a ground-truth) is Newman's *modularity* function [19]. The idea of modularity-based community detection is to try to assign each node of the given network to a community such that it maximizes the modularity value of the whole network. Modularity quantifies the community strength by comparing the fraction of edges within the community with such fraction when random connections between the nodes are made. Networks with high modularity have dense connections between the nodes within communities but sparse connections between nodes in different communities. The modularity function has several variants, but these variants share the same principle. Without the loss of generality, we use the following equation of modularity, an extension of Newman's modularity function designed to support overlapping communities proposed in [20]. For the given community partition of a network $\mathcal{N} = (V, E)$ with m edges, an extended modularity EQ is given by:

$$EQ = \frac{1}{2m} \sum_{C \in C_{\mathcal{N}}} \sum_{u,v \in C} \frac{1}{O_u O_v} \left[A_{uv} - \frac{d_u d_v}{2m} \right] \tag{1}$$

with $C_{\mathcal{N}}$ the set of communities in $\mathcal{N}$; O_u the number of communities to which the node u belongs and A_{uv} is the element of the adjacency matrix representing the network.

F1 score. Let $\mathcal{N} = (V, E)$ be a network, and $\hat{C}$ (respectively C^*) the set of (respectively ground truth) communities associated to $\mathcal{N}$. The average F1 score measure aims to quantify the level of correspondence between C^* and $\hat{C}$. More precisely, we need to determine which $C_i \in C^*$ corresponds to which $\hat{C}_i \in \hat{C}$. The F1 score is defined as the average of F1 score of the best matching ground-truth community to each detected community, and the F1 score of the best matching detected community to each ground-truth community [25]. More formally, this function is defined as follows:

$$\frac{1}{2}\left(\frac{1}{|C^*|}\sum_{C_i \in C^*} F_1(C_i, \hat{C}_{g(i)}) + \frac{1}{|\hat{C}|}\sum_{\hat{C}_i \in \hat{C}} F_1(C_{g'(i)}, \hat{C}_i)\right) \tag{2}$$

where the best matching g and g' is defined as follows: $g(i) = \arg\max_j F_1(C_i, \hat{C}_j)$, $g'(i) = \arg\max_j F_1(C_j, \hat{C}_i)$, and $F_1(C_i, \hat{C}_j)$ is the harmonic mean of Precision and Recall.

3 A SAT-based Framework for Community Detection

Fundamentally, communities allow us to discover groups of interacting objects and the relations between them. A community (also referred to as a cluster) is a set of cohesive nodes that have more connections inside the set than outside. In this section, we propose to use off-the-shelf satisfiability solvers for community detection using an appropriate encoding of the community detection task as a SAT optimization problem. Proximity between nodes have been expressed as direct edges expressing formally a direct relation. Individuals can be grouped into the same cluster even if they are not linked directly. Relationships between individuals can be expressed via some proximity conditions. For instance, individuals having much common friends could be considered as very closed to each other. Consequently, the definition of individuals proximity is clearly a fundamental issue, as it have a great impact on the outcome. Next, we establish the main definitions which will be used to formulate our problem.

Definition 1 (k-linked community). *A community is k-linked if the nodes are pairwise k-linked, i.e., the distance between each two nodes is less or equal than k.*

According to Definition 1, a k-linked community has a diameter less or equal than k. Now, to allow for the use of SAT problem in discovering overlapping communities, we need to focus on the following kinds of k-linked communities called k-linked *centered* communities: those having a centroid node or centroid edge that possesses a distance at most $\frac{k}{2}$ from each other node of the community.

Definition 2 (Node/Edge Centered k-linked Community). *Let $\mathcal{N} = (V, E)$ be a network and $k > 1$ a positive integer. A community $C \subseteq V$ is node*

(resp. edge) centered k-linked community of $\mathcal{N}$ *iff there exists* $c \in C$ *(resp.* $e = (u, v) \in E$ *with* $u, v \in C$*) s.t.* $\forall\ w \in C$, $dist(c, w) \leqslant \frac{k}{2}$ *(resp.* $dist(e, w) \leqslant \frac{k}{2}$*).*

Obviously, a node centered k-linked community is an edge centered k-linked community, while the converse is not true. Note also that a k-linked community is not necessarily a centered k-linked community. A counter-example consists of the network $\mathcal{N} = (V, E)$ where $V = \{x_1, \ldots, x_8\}$ and $E = \{(x_1, x_2), (x_2, x_3), (x_3, x_4), (x_5, x_6), (x_6, x_7), (x_7, x_8), (x_1, x_5), (x_4, x_8)\}$. Then, $C = V$ is a 4-linked community, while there is neither a node $x_i \in V$ nor edge $e \in E$ with distance at most 2 from all the remaining nodes of C.

Lemma 1. *Let* $\mathcal{N} = (V, E)$ *be a network,* $C \subseteq V$ *a community and* $k > 1$ *a positive integer. If* C *is a centered* k*-linked community, then* C *is also a* k*-linked community.*

Now, based on the notion of centered k-linked community, community detection is defined as an optimization problem, solving partial Max-SAT. To do so, our starting point is to find a set of centroids S in the given network. The next step is to formed the communities around the centroids based on a predefined parameter k which represents the diameter of the communities. Clearly, we distinguish the following two cases: k-linked node (resp. edge) centered communities corresponding to an even (resp. odd) value of k.

Next, we propose two appropriate reformulations as an optimization problem for the community detection problem corresponding to node and edge centered k-linked communities, respectively. To achieve this, propositional variables are used for representing the network. Indeed, we associate each node u (resp. edge e) with a propositional variable denoted x_u (resp. y_e) where $x_u, y_e \in \{0, 1\}$. The key idea is that the variables assigned to 1 represent the centroids nodes (resp. edges), i.e., $S_v = \{u \in V \mid \mathcal{I}(x_u) = 1\}$ (resp. $S_e = \{e \in E \mid \mathcal{I}(y_e) = 1\}$). We now describe our SAT-based encodings using such propositional variables.

Node Centered *k*-linked Community: Our encoding consists of a set of constraints. The first propositional formula expresses the fact that if a node u is a centroid ($\mathcal{I}(x_u) = 1$), then the nodes with a distance at most $\frac{k}{2}$ from u are placed to the same community that possesses u as a centroid.

$$\bigwedge_{u \in V} (x_u \rightarrow \bigwedge_{v \in V \mid dist(u,v) \leqslant \frac{k}{2}} \neg x_v) \tag{3}$$

Let us remark that constraint (3) can be expressed by a set of binary clauses:

$$\bigwedge_{u \in V} \bigwedge_{v \in V \mid dist(u,v) \leqslant \frac{k}{2}} (\neg x_u \vee \neg x_v)$$

After finding the centroids, we still have to determine whether a node u belongs to community C or not depending on the value of k. To achieve this, we use the following formula that affects nodes of the network to communities

where they belong to, i.e., nodes that have a distance at most of $\frac{k}{2}$ from the centroid.

$$\bigwedge_{u \in V} \bigvee_{v \in V | dist(u,v) \leqslant \frac{k}{2}} x_v \tag{4}$$

Proposition 1. *If the constraints (3) ∧ (4) are satisfied, then for all $u \notin S_v$ there exists $v \in S_v$ s.t. $dist(u, v) \leqslant \frac{k}{2}$.*

Proposition 1 ensures that if (3) ∧ (4) admits a model $\mathcal{I}$, then the nodes corresponding to the variables assigned to 1 ($\{u \in V \mid \mathcal{I}(x_u) = 1\}$) are the centroids and the network can be partitioned into $|S|$ communities. The communities can then be constructed by finding the nodes with a distance at most $\frac{k}{2}$ from each centroid.

Obviously, the formula (3) ∧ (4) may admits many candidate solutions (i.e. models). However, choosing an arbitrary model do not always guarantee a best partition of the network into communities. To alleviate this problem, we will consider an objective function to optimize over the space of solutions. Then, the node centered k-linked community detection problem can be formulated as the following optimisation problem:

$$\min/\max \sum_{u \in V} x_u \qquad \text{subject to (3) } \wedge \text{ (4)} \tag{5}$$

Edge Centered *k*-linked Community: Now, to derive the formulation of edge centered k-linked community detection problem, we use similar reasoning as for node centered k-linked community, except that we consider centroid edges instead of centroid nodes. To do so, a community is built around an edge $e = (u, v)$ by considering nodes with a distance at most $\frac{k}{2}$ from the edge e. This is equivalent to partition the set of edges into modules and from that modules we can deduce the set of communities of nodes.

In the same way as for centroid nodes, the following formula expresses the fact that if an edge $e = (u, v)$ is a centroid edge ($\mathcal{I}(y_e) = 1$), then the nodes with a distance at most $\frac{k}{2}$ from u or v are assigned to 0.

$$\bigwedge_{e=(u,v) \in E} (y_e \rightarrow \bigwedge_{e' \in E | dist(e',u) \leqslant \frac{k}{2} || dist(e',v) \leqslant \frac{k}{2}} \neg y_{e'}) \tag{6}$$

Let us now introduce the following formula that affects nodes of the network to their associated communities, i.e. nodes that have a distance of $\frac{k}{2}$ from the centroid edge e.

$$\bigwedge_{e=(u,v) \in E} \bigvee_{e' \in E | dist(e',u) \leqslant \frac{k}{2} || dist(e',v) \leqslant \frac{k}{2}} y_{e'} \tag{7}$$

After fixing the centroids edges, the constraint 7 allows to identify whether a node u belongs to a community C or not from the value of k.

Similarly, to improve the quality of the detected communities, our edge centered k-linked community detection problem is formulated as the following optimisation problem:

$$\min/\max \sum_{e \in E} y_e \qquad \text{subject to } (6) \wedge (7) \tag{8}$$

We will use the notation $\text{CDSAT}^k_{\min/\max}$ to denote the optimization problems (5) and (8).

Example 1. Let us consider the undirected network $\mathcal{N} = (V, E)$ depicted in Figure 1. Setting $k = 4$ can lead to the following solution of $\text{CDSAT}^4_{\max}$: $\mathcal{I} = \{\neg x_1, \neg x_2, \neg x_3, \neg x_4, \neg x_5, x_6, \neg x_7, x_8, \neg x_9, \neg x_{10}, \neg x_{11}\}$. So for that solution, $\mathcal{N}$ can be partitioned into the two communities $C_1 = \{x_1, \ldots, x_6, x_7, x_{11}\}$ and $C_2 = \{x_1, x_2, x_5, x_6, x_7, \ldots, x_{11}\}$. In contrast, $\text{CDSAT}^4_{\min}$ leads to one community with centroid x_1 and containing all the nodes of $\mathcal{N}$.

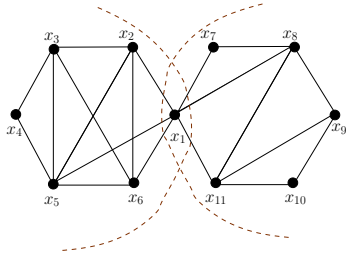


Fig. 1. A simple undirected network

Overlapping Enhancement: As said before, once the node/edge centroids are found, the communities are formed arround them based on a predefined parameter k. As a result, some nodes can belong to multiple communities as illustrated in Example 1. However, such overlapping can be huge and not significant enough w.r.t. real communities. To overcome this drawback and to allow for an accurate partition of the network, we propose a simple but effective overlaps reduction technique in order to correctly identify dense community overlaps. Starting from a set of communities, each overlapping node will be assigned to its closest communities according to its distance from the centroids of these communities.

Example 2. Let us consider again the network $\mathcal{N} = (V, E)$ of Figure 1. By enhancing the overlapping, the two communities are reduced to $C_1 = \{x_1, \ldots, x_6\}$ and $C_2 = \{x_1, x_7, \ldots, x_{11}\}$.

Algorithm 1 describes the general feature of our SAT-based node centered k-linked community detection procedure [1]. The algorithm takes as input the

[1] Algorithm 1 can be slightly modified to deal with edge centered k-linked community detection problem.

network and even integer k and returns a set of overlapping communities. It proceeds as follows: First, we generate the corresponding optimization problem that can be represented as a Partial MaxSAT problem (line 1). Then, a state-of-the-art Weighted Partial MaxSAT solver `WPM3` is used to get an optimal solution (i.e. model) $\mathcal{I}$. Next, the centroids are determined from the obtained model (lines 4-7). Using such centroids, the next step is to build communities by finding the nodes with a distance at most $\frac{k}{2}$ from each centroid. Finally, the cleaning step is called to improve the quality of detected communities (lines 11-13).

Algorithm 1: $\text{CDSAT}^{k}_{\text{min/max}}$

Input: A network $\mathcal{N} = (V, E)$ and an integer $k > 1$
Output: A set of overlapping communities

```
1  Φ = encodeToOpt(k, G);
2  I = solve(Φ) ;
3  S ← ∅;
4  for u_x ∈ I do
5  |  if I(u_x) == 1 then
6  |  |  C_u ← {u};
7  |  |  S ← S ∪ C_u
8  |  end
9  end
10 for v_x ∈ I do
11 |  for C_u ∈ S do
12 |  |  if dist(u, v) ⩽ k/2 then  C_u ← C_u ∪ {v} ;
13 |  end
14 end
15 for C_u, C_v ∈ S × S do
16 |  for w ∈ V do
17 |  |  if dist(w, u) < dist(w, v) then  C_v ← C_v \ {w} ;
18 |  end
19 end
20 return S
```

4 Performance Evaluation

4.1 Experiment Settings

In this section, we present an experimental evaluation of our proposed approach. It was conducted on fourteen networks that cover a variety of application areas and are briefly described in Table 1 (columns 1 and 2). Some of these networks have ground-truth communities as presented in column 2 of Table 2. We have also chosen three large networks (`Facebook`, `DBLP`, and `Amazon` taken from SNAP [14]) to show the scalability of our model.

We evaluate the performance of our approaches by comparing them with the following most prominent state-of-the-art overlapping community detection algorithms: (i) *Community-Affiliation Graph Model* (`AGM`) [24], (ii) *Clique Percolation Method* (`CPM`) [1], (iii) *Cluster Affiliation Model for Big Networks* (`BIGCLAM`) [25], and (iv) *Communities from Edge Structure and Node Attributes* (`CESNA`) [26]. For the `CPM` algorithm, we use the cliques of size equal to 3. For `BIGCLAM`

method, user can specify the number of communities to detect, or let the program determine the number of communities in the network from the topology of the network. We opt for the second case where the number of communities is not fixed in advance.

The proposed system, referred to as $\text{CDSAT}^{k}_{\text{min/max}}$, was written in Python. Given an input network as a set of edges, our algorithm starts by generating the corresponding optimization problem represented as a Partial MaxSAT problem. To solve this problem, we consider the state-of-the-art Weighted Partial MaxSAT solver `WPM3` (best solver at the last MaxSAT competition [2]) [2]. For our experimental study, all algorithms have been run on a PC with an Intel Core 2 Duo (2 GHz) processor and 2 GB memory. We imposed one hour time limit for all the methods. Last, we use the symbol (-) in Tables 1 and 2 to indicate that the method is not able to scale on the considered network under the time limit.

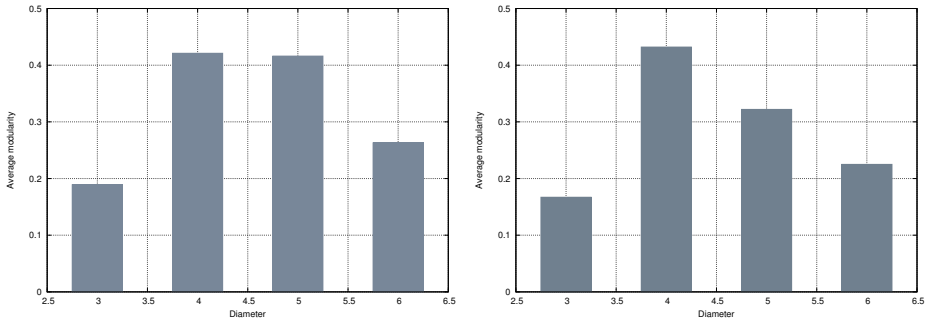
4.2 Choosing the Best Value of the Diameter

Our $\text{CDSAT}^{k}_{\text{min/max}}$ algorithm takes as input a network and a positive integer k and returns a set of overlapping communities. In order to determine the best diameter k, we run our $\text{CDSAT}^{k}_{\text{min/max}}$ algorithms on the fourteen considered networks, while varying k from 3 to 6. The Figure 2 summarises the relationship between the average modularity and k. In both algorithms $\text{CDSAT}^{k}_{\text{min}}$ and $\text{CDSAT}^{k}_{\text{max}}$, we see that the best average modularity is achieved for the diameter $k = 4$. As Figure 2 reveals, the best average modularity is obtained by $\text{CDSAT}^{4}_{\text{min}}$ and $\text{CDSAT}^{4}_{\text{max}}$ with a value of 0.421 and 0.432 respectively. As we can observe, the average modularity obtained by both algorithms decreases beyond $k = 4$. Overall, for both algorithms the best average modularity is obtained for $k = 4$. There performances are relatively close. This can be explained by the fact that real-world social networks possess small (average or effective) diameter (e.g. [5]). This can be related to the property of the small-world phenomenon observed by several authors on real graphs (e.g. [21]).

4.3 Comparison with Baseline Algorithms

Experiments with modularity metric. Table 1 reports the performance comparison between our $\text{CDSAT}^{4}_{\text{min/max}}$ approaches and the considered methods. Experiments show that our methods outperform every baseline, in most cases, by an interesting margin as shown by the average modularity reported in the last line of Table 1. We observe that across all datasets and modularity metric, $\text{CDSAT}^{4}_{\text{min}}$ yields the best performance in 8 out of 14 networks. We also note that $\text{CDSAT}^{4}_{\text{min}}$ shows a high margin in performance gain against the baselines in two large networks `DBLP` and `Amazon`, and in a collaborations network such as `Coauthorship`. In terms of average performance, $\text{CDSAT}^{4}_{\text{min}}$ outperforms CPM by 111.55%, BIGCLAM by 26.42%, and CESNA by 40.80%. Similarly, we note

[2] http://maxsat.ia.udl.cat/introduction/



$\text{CDSAT}^k_{\text{min}}$ $\text{CDSAT}^k_{\text{max}}$

Fig. 2. Average modularity for $\text{CDSAT}^k_{\text{min/max}}$

that $\text{CDSAT}^4_{\text{max}}$ outperforms all the other methods in 7 out of 14 datasets. In terms of average performance, $\text{CDSAT}^4_{\text{max}}$ outperforms CPM by 117.08%, BIGCLAM by 29.72%, and CESNA by 44.48%. We also observe that $\text{CDSAT}^4_{\text{max}}$ gives an important improvement against the baselines in two large networks `Facebook`, `DBLP`, and also in a collaborations network like `Coauthorship`. On the `Lemis`, `Power grid`, `Pilgrim`, and `Jazz` datasets, our methods remain relatively competitive with the best baseline. A possible explanation for this phenomenon is that the `WPM3` solver don't return the optimal solution for these datasets. Overall, our methods outperform BIGCLAM, which is the most competing algorithm, on all large real datasets.

Table 1. Modularity based performance on fourteen datasets

Networks	nodes/edges	AGM	CPM	BIGCLAM	CESNA	$\text{CDSAT}^4_{\text{min}}$	$\text{CDSAT}^4_{\text{max}}$
Dolphin [16]	62/159	−0.040	0.304	0.053	0.095	**0.438**	0.297
Karate [23]	34/78	0.200	0.230	0.195	0.180	**0.310**	**0.311**
Risk map [4]	42/83	0.415	0.488	0.194	0.504	**0.571**	**0.528**
Lemis [11]	77/254	0.162	0.205	**0.444**	0.311	0.064	0.419
Word adjacencies [18]	112/425	0.139	0.031	0.154	0.111	**0.175**	0.098
Football [8]	115/615	0.222	0.199	0.343	0.390	0.286	**0.404**
Facebook [14]	4039/88234	–	–	0.391	0.539	0.449	**0.701**
DBLP [14]	317080/1049866	–	0.293	0.216	0.202	**0.520**	**0.436**
Amazon [14]	334863/925872	–	0.195	0.341	0.430	**0.616**	**0.502**
Books [8]	105/441	0.366	0.265	0.308	0.255	**0.439**	0.345
Power grid [22]	4941/6594	–	0.007	**0.840**	0.586	0.679	0.547
Coauthership [18]	1462/2742	0.619	0.456	0.679	0.031	**0.923**	**0.852**
Pilgrim [3]	34/128	0.368	0.096	**0.415**	0.321	0.312	0.407
Jazz [9]	196/2742	**0.310**	0.022	0.099	0.231	0.112	0.208
Average	N/A	N/A	0.199	0.333	0.299	**0.421**	**0.432**

Results on ground-truth communities. After finding communities in a given network, we can gauge the performance of each community that an algorithm has discovered and whether a ground-truth community has been successfully identified. Table 2 summarizes the evaluation results, with F1 scores of all algorithms

on each network. Interestingly, it can be seen that $\text{CDSAT}^4_{\text{min}}$ and $\text{CDSAT}^4_{\text{max}}$ produce more accurate average w.r.t. the ground-truth setting than all the other baseline algorithms. In terms of average performances, $\text{CDSAT}^4_{\text{min}}$ outperforms CPM by 16%, BIGCLAM by 22%, and CESNA by 35.05%. Moreover, notice that $\text{CDSAT}^4_{\text{max}}$ outperforms CPM by 6.49%, BIGCLAM by 12%, and CESNA by 23.98%. In the cases of `Karate`, `Risk map` and `DBLP` data instances, $\text{CDSAT}^4_{\text{min}}$ and $\text{CDSAT}^4_{\text{max}}$ achieves a closely gain in the F1 score compared to the best baseline (CPM in this case).

As a summary, experimental results confirm that $\text{CDSAT}^4_{\text{min/max}}$ methods achieve the overall best performance in terms of the accuracy of the detected overlapping communities.

Table 2. F1 Score based performance of methods on fourteen datasets

Networks	Communities	AGM	CPM	BIGCLAM	CESNA	$\text{CDSAT}^4_{\text{min}}$	$\text{CDSAT}^4_{\text{max}}$
Dolphin	2	0.120	0.579	0.628	0.100	**0.749**	**0.659**
Karate	2	0.864	**0.857**	0.629	0.663	0.847	0.851
Risk map	6	0.641	**0.884**	0.694	0.842	0.779	0.769
DBLP	13477	–	**0.596**	0.370	0.310	0.470	0.483
Amazon	75149	–	0.519	0.498	0.642	**0.695**	0.399
Books	3	0.684	0.557	0.549	0.591	**0.804**	0.652
Pilgrim	4	0.773	0.427	0.835	0.652	0.785	**0.892**
Average	N/A	N/A	0.631	0.600	0.542	**0.732**	**0.672**

Evaluating scalability. Finally, we evaluate the scalability of the different community detection methods by measuring the CPU time (see Table 3). From the results, it can be seen that our algorithms make few seconds to generate all communities for small networks. However, the CPM, BIGCLAM and CESNA baselines are faster than our methods for small networks (up to 200 nodes). We can observe that $\text{CDSAT}^4_{\text{min}}$ and $\text{CDSAT}^4_{\text{max}}$ are third-fastest method overall, when the network becomes larger. Interestingly, we also notice that our algorithms are the second-fastest methods, next BIGCLAM, for `DBLP` and `Amazon`.

Table 3. Comparison in terms of running Time (s)

Networks	AGM	CPM	BiGCLAM	CESNA	$\text{CDSAT}^4_{\text{max}}$	$\text{CDSAT}^4_{\text{min}}$
Dolphin	6.77	0.09	0.24	0.07	14	8
Karate	35	0.07	0.29	0.07	11	7.15
Risk map	62	0.09	2.84	0.59	38	17
Lemis	200	0.10	0.55	0.09	16	12
Word adjacencies	60.35	0.09	0.97	0.13	60.60	11
Football	47.71	0.08	1.78	0.13	120.20	420
Facebook	$> 1h$	$> 1h$	240.38	4.81	360.30	480.7
DBLP	$> 1h$	3240	60.56	900.34	720.50	780.40
Amazon	$> 1h$	$> 1h$	60.09	1200.49	780.20	900
Books	19.83	0.12	2.71	0.10	14.35	60.20
Power grid	$> 1h$	0.66	0.81	4.48	420.15	480.25
Co-authership science	360.17	0.07	14.08	0.05	360.58	360.20
Pilgrim	0.61	0.09	0.35	0.07	8.2	7
Jazz	60.02	0.09	2.84	0.59	360.12	120

5 Conclusion

In this paper, we developed a novel community detection approach that accurately discovers the overlapping community structure of real-world networks. Our method is based on a partition of the network into modules having with maximum diameter k. We identified two kind of community structures: centroid node based community and centroid edge based community. Different encodings into Partial MaxSAT problem are provided and a policy to limit the overlapping is proposed. Experiments show that our approach outperforms the stateof-the-art methods in accurately discovering network communities.

Our work has several implications: First, our encoding can lead to big optimization problems for that the identification of an optimal solution is challenging. Decomposition and parallel algorithms would be very helpful to improve the resolution process. Second, the optimal solution is not unique which can affect the quality of clusters. Looking for particular solutions is another interesting research direction to improve the quality of detected communities.

References

1. B. Adamcsek, G. Palla, I. J. Farkas, I. Derényi, and T. Vicsek. Cfinder: locating cliques and overlapping modules in biological networks. *Bioinformatics*, 22(8):1021–1023, 2006.
2. C. Ansótegui, F. Didier, and J. Gabàs. Exploiting the structure of unsatisfiable cores in MaxSAT. In *IJCAI*, pages 283–289, 2015.
3. Brian, B. Dickinson, W. Valyou, and Hu. A genetic algorithm for identifying overlapping communities in social networks using an optimized search space. *Social Networking*, 02No.04:1–9, 2013.
4. J. Cheng, M. Leng, L. Li, H. Zhou, and X. Chen. Active semi-supervised community detection based on must-link and cannot-link constraints. *PLoS ONE*, 9(10):1–18, 2014.
5. F. Comellas, J. Ozón, and J. G. Peters. Deterministic small-world communication networks. *Information Processing Letters*, 76(1):83 – 90, 2000.
6. S. Fortunato. Community detection in graphs. *CoRR*, abs/0906.0612, 2009.
7. S. Gilpin and I. N. Davidson. Incorporating SAT solvers into hierarchical clustering algorithms: an efficient and flexible approach. In *KDD*, pages 1136–1144, 2011.
8. M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proc.Natl.Acad.Sci*, 99:7821, 2002.
9. P. Gleiser and L. Danon. Community structure in jazz. *Advances in Complex Systems*, 6:565, 2003.
10. T. Guns, S. Nijssen, and L. D. Raedt. Itemset mining: A constraint programming perspective. *Artif. Intell.*, 175(12-13):1951–1983, 2011.
11. D. E. Knuth. *The Stanford GraphBase - a platform for combinatorial computing.* ACM, 1993.
12. D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, 2001.
13. J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg. Predicting positive and negative links in online social networks. In *WWW*, pages 641–650, 2010.

14. J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.
15. J. Leskovec, K. J. Lang, and M. W. Mahoney. Empirical comparison of algorithms for network community detection. In *WWW*, pages 631–640, 2010.
16. D. Lusseau, K. Schneider, O. Boisseau, P. Haase, E. Slooten, and S. Dawson. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405, 2003.
17. M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74, 2006.
18. M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74:036104, 2006.
19. M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, Feb. 2004.
20. H. Shen, X. Cheng, K. Cai, and M. Hu. Detect overlapping and hierarchical community structure in networks. *Physica A*, 388(8):1706–1712, 2009.
21. D. J. Watts, P. S. Dodds, and M. E. J. Newman. Collective dynamics of small-world networks. *Nature*, (393):440–442, 1998.
22. D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *nature*, 393(6684):440–442, 1998.
23. Z. W.W. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.
24. J. Yang and J. Leskovec. Community-affiliation graph model for overlapping network community detection. In *ICDM*, pages 1170–1175, 2012.
25. J. Yang and J. Leskovec. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *WSDM*, pages 587–596, 2013.
26. J. Yang, J. J. McAuley, and J. Leskovec. Community detection in networks with node attributes. *CoRR*, abs/1401.7267, 2014.

A Parallel SAT-based Framework for Closed Frequent Itemsets Mining

Imen Ouled Dlala [1,3], Said Jabbour[1], Badran Raddaoui [2], and Lakhdar Sais[1]

[1]CRIL-CNRS, Université d'Artois, F-62307 Lens Cedex, France
[2]SAMOVAR, Télécom SudParis, CNRS, Univ. Paris-Saclay, Evry, France
[3]LARODEC, University of Tunis, Tunisia
{dlala,jabbour,sais}@cril.fr,badran.raddaoui@telecom-sudparis.eu

Abstract. Constraint programming (CP) and propositional satisfiability (SAT) based framework for modeling and solving pattern mining tasks has gained a considerable audience in recent years. However, this nice declarative and generic framework encounters a scaling problem. The huge size of constraints networks/propositional formulas encoding large datasets is identified as the main bottleneck of most existing approaches. In this paper, we propose a parallel SAT based framework for itemset mining problem to push forward the solving efficiency. The proposed approach is based on a divide-and-conquer paradigm, where the transaction database is partitioned using item-based guiding paths. Such decomposition allows us to derive smaller and independent Boolean formulas that can be solved in parallel. The performance and scalability of the proposed algorithm are evaluated through extensive experiments on several datasets. We demonstrate that our partition-based parallel SAT approach outperforms other CP approaches even in the sequential case, while significantly reducing the performances gap with specialized approaches.

1 Introduction

Frequent itemset mining (abbreviated as FIM) [1,4] is a fundamental research topic in data mining (DM). It aims to discover important relationships between items in a database arising from numerous applications, ranging from marketing to scientific data analytics. Different kinds of interesting patterns and constraints have been introduced, including closeness and maximality constraints, association rules and its variants [1,3,36]. Such progress results in various scalable algorithms designed to deal with specific data mining tasks or constraints. However, such dedicated data mining systems are highly difficult to maintain when additional or combination of constraints came into play. This observation together with the increasing need in terms of user-preference led to the first and seminal paper by De Raedt et al. [28], opening a research avenue on the design of declarative approaches for itemset mining and for pattern mining in general. This new framework offers a flexible representation model that does not require any deep changes in the implementations. In other words, new constraints can be

easily integrated without developing specialized methods. These existing declarative approaches for data mining clearly help pattern selection strategies such as minimizing the redundancy or combining patterns on the basis of their usefulness in a given context.

Encouraged by these promising results, several contributions addressed different data mining tasks using the well-known *constraint programming* (CP), *propositional satisfiability* (SAT) and *answer set programming* (ASP) AI formalisms, such as frequent sequence mining [18,26,11], frequent itemset mining [15], closed frequent itemset mining [22,30], association rules mining [5], clustering [7,6], and community detection [16,10]. A high-level language for constraint-based mining, called MiningZinc, is introduced by Guns et al. [12]. It supports a wide variety of different solvers, including DM algorithms and general purpose solvers, and uses a significantly more high-level modeling language. This combination of generic and specialized solvers gives MiningZinc the ability to be both generic and efficient with respect to the performance of the state-of-the-art algorithms on common data mining tasks.

In the first CP-based proposal for itemset mining [13], the problem is expressed through different linear and reified constraints over Boolean variables. This widely adopted model does not exploit some of the well established algorithmic skills from the state-of-the-art specialized algorithms. For the closed itemset mining task, to enhance the efficiency of such CP model, Lazaar et al. [22] proposed a new formulation using a global constraint that incorporates some of the propagation properties borrowed from specialized algorithms. This approach allows to capture the closed frequent itemset mining problem without requiring reified constraints or extra variables, leading to significantly better performances. More recently, Schaus et al. [30] introduced the CoverSize constraint for the itemset mining problem, a global constraint for counting and constraining the number of transactions covered by the itemset decision variables. The authors showed that compared to the ClosedPattern approach [22] using a global constraint for frequent closed itemset mining, both generality and efficiency can be significantly improved. A relation is established between the CoverSize constraint and the well-known table constraint, where the underlined filtering algorithm internally exploits the reversible sparse bitset data structure used for the filtering table. By expressing the size of the cover as a variable, the proposed approach opens up new modelling perspectives. Moreover, a SAT-based framework for enumerating Top-k Motifs in both transaction databases, sequences and sequences of itemsets is proposed in [20]. As a summary, this new framework offers a nice declarative and flexible representation model, while facing a real challenge in terms of scalability. Indeed, all these CP/SAT-based itemset mining approaches flag out good performance on datasets of reasonable size and high support threshold. However, the performance decreases as the database increases in size or the support threshold turns to be low. In this latter case, the size of constraints network/propositional formulas encoding the itemset mining task tends to be huge. Space complexity is clearly identified as

the main bottleneck behind the competitiveness of these new declarative and flexible models w.r.t. specialized data mining approaches.

Some early efforts tried to speed up the specialized pattern mining algorithms by running them in parallel [35,24,27]. Several parallel approaches use Spark or MapReduce frameworks [33,23,38], through multi-core processors or distributed computing platforms. For an overview of parallel FIM specialized methods, readers are kindly referred to [25]. Moreover, Savasere et al. [29] adressed the problem of generating association rules from large databases by introducing a new algorithm which divides the database into a number of non-overlapping partitions. The partitions are considered one at a time and all large itemsets for that partition are generated.

In spite of the significance of the declarative framework for frequent pattern mining, and despite of their main scalability bottleneck, no advances have been made on parallelizing CP/SAT-based pattern mining approaches. To avoid the generation of a single huge propositional formula encoding the whole transaction database, in [19], the authors proposed an incremental approach allowing to partition the whole problem into sub-problems of reasonable size while maintaining incremental solving. It takes as input a transaction database and a partition of the set of items, then it incrementally generates and sequentially solves a sequence of sub-problems while ensuring completeness. In this paper, we propose a new parallel SAT based framework for Closed Frequent Itemset Mining (in short **paraSatMiner**). Our proposed method is based on a divide-and-conquer paradigm. The main idea is to decompose the input transaction database, using item-based guiding paths, leading to sub-formulas that can be solved independently in parallel. Our paraSatMiner approach is carefully implemented on a multicore architecture, while maintaining workload balanced between the different processor units. Extensive experimental comparative evaluation on several datasets achieves significant performance improvements w.r.t. the two most recent and effective CP approches [22,30] even on the sequential case, while reducing the performance gap against LCM, one of the state-of-the-art specialized algorithms. We also show additional performance gains up to 8 cores. Contrary to CP mining systems based on the design of specific global constraints, our approach does not integrate any technique or property from specialized algorithms.

2 Technical Background and Preliminary Definitions

2.1 Propositional Logic and SAT Problem

Let $\mathcal{L}$ be a propositional language defined from a finite set $\mathcal{PS}$ of propositional symbols (p, q, r, etc.). The set of formulas is defined inductively from $\mathcal{PS}$, the constant $\bot$ denoting false, the constant $\top$ denoting true, and using the classical logical connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$ and the equivalence connective $\leftrightarrow$. For every two propositional formulas Φ and Ψ from $\mathcal{L}$, the connective $\leftrightarrow$ is defined by $\Phi \leftrightarrow \Psi \equiv (\Phi \rightarrow \Psi) \wedge (\Psi \rightarrow \Phi)$. Moreover, we refer to the set of symbols from $\mathcal{PS}$ that occur in the formula Φ as $\mathcal{S}(\Phi)$.

A *boolean interpretation* $\mathcal{B}$ of a formula Φ is a truth assignement of $\mathcal{PS}$, that is, a total function from $\mathcal{S}(\Phi)$ to $\{0,1\}$ (0 corresponds to false and 1 corresponds to true). For every two formulas Φ and Ψ from $\mathcal{L}$, we have the following equivalences: $\mathcal{B}(\bot) = 0$, $\mathcal{B}(\top) = 1$, $\mathcal{B}(\neg\Phi) = 1 - \mathcal{B}(\Phi)$, $\mathcal{B}(\Phi \wedge \Psi) = min(\mathcal{B}(\Phi), \mathcal{B}(\Psi))$, $\mathcal{B}(\Phi \vee \Psi) = max(\mathcal{B}(\Phi), \mathcal{B}(\Psi))$, and $\mathcal{B}(\Phi \rightarrow \Psi) = max(1 - \mathcal{B}(\Phi), \mathcal{B}(\Psi))$.

A *model* of a formula Φ is a boolean interpretation $\mathcal{B}$ that satisfies Φ, i.e., $\mathcal{B}(\Phi) = 1$. A formula Φ is *satisfiable* if there exists a model of Φ. We denote by $\mathcal{M}(\Phi)$ the set of all models of Φ. Let $X \subseteq \mathcal{S}(\Phi)$ and $\mathcal{B} \in \mathcal{M}(\Phi)$, we define $\mathcal{B}^X = \{x \in X \mid \mathcal{B}(x) = 1\}$ and $\mathcal{M}^X(\Phi) = \{\mathcal{B}^X \mid \mathcal{B} \in \mathcal{M}(\Phi)\}$.

As usual, every finite set of formulas is considered as the conjunctive formula whose conjuncts are the elements of the set. A formula in *conjunctive normal form* (CNF) is a (finite) conjunction of clauses. A *clause* is a (finite) disjunction of literals. A *literal* is either a propositional variable x of $\mathcal{PS}$ or its negation $\neg x$. Let us also mention that any propositional formula can be translated to a CNF formula equivalent w.r.t. satisfiability, using linear Tseitin's encoding [32]. The propositional satisfiability problem, called **SAT**, is the decision problem of determining the satisfiability of a CNF formula.

2.2 Parallel SAT Solving

Parallel SAT solving has received a lot of attention in the last years. This comes from several factors like the wide availability of cheap multicore platforms combined with the relative performance stall of sequential SAT solvers.

Many parallel SAT solvers have been previously proposed. Most of them are based on the divide-and-conquer principle. These solvers either divide the search space using for example guiding-paths [37,31] or the formula itself using decomposition techniques. The main issue behind these approaches rises in (i) getting the workload balanced between the different processing units, and (ii) selecting the most relevant guiding paths.

Portfolio-based parallel SAT solving has been recently introduced [14]. It avoids the previous problem by letting several DPLL engines compete and cooperate to be the first to solve a given instance. Each solver works on the original formula, and search spaces are not split or decomposed anymore. To be efficient, the portfolio has to use diversified search engines. This maximizes the chance of having one of them solving the problem. However, when clause sharing is added, diversification has to be restricted in order to maximize the impact of a foreign clause whose relevance is more important in a similar or related search effort. A challenging question is to maintain a good and relevant distance between the parts of the search space explored by the different search efforts which is equivalent to finding of a better diversification and intensification tradeoff.
These two parallel solving paradigms are complementary and admit their own strengths and weaknesses. As our goal is to partition the transactions database in order to generate several formulas of reasonable size, the divide-and-conquer based paradigm is clearly the most convenient for our purpose.

2.3 Itemset Mining Problem based on Boolean Satisfiability

Let Ω denote a universe of items (or symbols), called alphabet. A *transaction database* is a finite set of n data records, called *transactions*, denoted by $\mathcal{D} = \{T_1, T_2, ..., T_n\}$. Each transaction T_i $(1 \leqslant i \leqslant n)$ is defined as a couple $(tid_i, \mathcal{I}_i)$, where tid_i is the *transaction identifier* and $\mathcal{I}_i \subseteq \Omega$ an itemset, i.e., unordered collection of items from Ω. We use 2^{Ω} to denote the set of all possible itemsets over Ω. Let $a \in \Omega$, we define $\mathcal{D}\downarrow_a = \{T \mid T = (tid, \mathcal{I}) \in \mathcal{D}, a \in \mathcal{I}\}$ the set of transactions containing a. We also define $\mathcal{D}\uparrow_a = \{(tid, \mathcal{I}\backslash\{a\}) \mid (tid, \mathcal{I}) \in \mathcal{D}\}$.

We associate to each $a \in \Omega$ a propositional variable denoted l_a. We note $\mathcal{P}$ such set of variables encoding the items in Ω and representing the candidate pattern. We also associate with each transaction T_i $(1 \leqslant i \leqslant n)$ a propositional variable q_i. We also note $\mathcal{Q}$ the set of variables associated to transactions. These variables will be used to express the following notions as propositional formulas.

Definition 1 (Cover). *Let $\mathcal{D} = \{T_1, T_2, ..., T_n\}$ be a transaction database. The cover of an itemset $\mathcal{I} \subseteq \Omega$ in $\mathcal{D}$, denoted $\mathcal{C}(\mathcal{I}, \mathcal{D})$, corresponds to the following set of transaction identifiers: $\mathcal{C}(\mathcal{I}, \mathcal{D}) = \{tid \mid (tid, \mathcal{J}) \in \mathcal{D}$ and $\mathcal{I} \subseteq \mathcal{J}\}$.*

For instance, if we consider the transaction database of Table 1, we have $\mathcal{C}(\{b\}, \mathcal{D}) = \{1, 2, 3, 4\}$ while $\mathcal{C}(\{a, b\}, \mathcal{D}) = \{1, 2, 3\}$.

TID	Transactions
T_1	a b c d
T_2	a b c e
T_3	a b e
T_4	b d
T_5	d f g
T_6	f g h
T_7	h

Table 1: Transaction Database

Cover constraint: The following constraint allows us to capture all the transactions where the candidate itemset does not appear:

$$\Phi_{\mathcal{D}}^{cov} = \bigwedge_{i=1}^{n} (\neg q_i \leftrightarrow \bigvee_{a \in \Omega \backslash \mathcal{I}_i} l_a) \tag{1}$$

This constraint means that q_i is true if and only if the candidate itemset is in the transaction T_i.

Definition 2 (Support). *Let $\mathcal{D} = \{T_1, T_2, ..., T_n\}$ be a transaction database. The support of an itemset $\mathcal{I} \subseteq \Omega$ in $\mathcal{D}$, denoted by $\mathcal{S}(\mathcal{I}, \mathcal{D})$, is defined as follows: $\mathcal{S}(\mathcal{I}, \mathcal{D}) = |\mathcal{C}(\mathcal{I}, \mathcal{D})|$.*

From the transaction database of Table 1, we have $\mathcal{S}(\{a, b\}, \mathcal{D}) = 3$.

Definition 3 (Frequent Itemset Mining). *Given a transaction database D and θ an explicit frequency user-specified support threshold. The problem of frequent itemset mining, denoted $FIM(\mathcal{D}, \theta)$, can be defined as follows:*

$$FIM(\mathcal{D}, \theta) = \{I \subseteq \Omega \mid \mathcal{S}(I, \mathcal{D}) \geqslant \theta\}$$

As an example, consider again the transaction database depicted by Table 1, we have $FIM(\mathcal{D}, 2) = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{g\}, \{h\}, \{a, b\}, \{a, c\}, \{a, e\}, \{b, c\}, \{b, e\}, \{b, d\}, \{f, g\}, \{a, b, c\}, \{a, b, e\}\}$.

Let us note that the frequency is a monotonic property w.r.t. set inclusion, meaning that if an itemset is not frequent, none of its supersets are frequent. Similarly, if an itemset is frequent, all of its subsets are frequent.

Frequency constraint: To express that the candidate pattern occurs at least θ times, we use the following 0/1 linear inequality:

$$\Phi^{freq}_{\mathcal{D},\theta} = \sum_{i=1}^{n} q_i \geqslant \theta \tag{2}$$

Then, the SAT-based encoding of $FIM(\mathcal{D}, \theta)$ is expressed as: $\Phi^{fim}_{\mathcal{D},\theta} = \Phi^{cov}_{\mathcal{D}} \wedge \Phi^{freq}_{\mathcal{D},\theta}$. Obviously, the monotonic property is implicitly satisfied by the frequency constraint. Notice also that the constraint (2) corresponds to the well known boolean cardinality constraint, subject of several efficient CNF encoding that maintain generalized arc consistency via unit propagation [34,2,17].

Now, in order to reduce the size of the huge number of extracted itemsets, condensed representations have been introduced, by exploiting the structure of the itemsets data. We define below *closed frequent itemsets* as one of these condensed representations.

Definition 4. *(Closed Itemsets): Let $\mathcal{D}$ be a transaction database. We say that the itemset $\mathcal{I}$ is closed if and only if for all $\mathcal{J} \supset \mathcal{I}$, $\mathcal{S}(\mathcal{I}, \mathcal{D}) > \mathcal{S}(\mathcal{J}, \mathcal{D})$.*

Closeness constraint: The constraint allowing to force the candidate itemset to be closed can be expressed by the following propositional formula:

$$\Phi^{clos}_{\mathcal{D}} = \bigwedge_{a \in \Omega} ((\bigvee_{(tid_i, \mathcal{I}_i) \in \mathcal{D},\ a \notin \mathcal{I}_i} q_i) \vee l_a) \tag{3}$$

That is, this formula means that if we have $\mathcal{S}(\mathcal{I}, \mathcal{D}) = \mathcal{S}(\mathcal{I} \cup \{a\}, \mathcal{D})$, then $a \in \mathcal{I}$ holds. This condition is necessary and sufficient to force the candidate itemset to be closed. Let us note that an expression of the form $a \in \mathcal{I}_i$ corresponds to a constant, i.e., $a \in \mathcal{I}_i$ corresponds to $\top$ if the item a is in $\mathcal{I}_i$, to $\bot$ otherwise.

The closed frequent itemset mining task **CFIM** can then be encoded as $\Phi^{cfim}_{\mathcal{D},\theta} = \Phi^{clos}_{\mathcal{D}} \wedge \Phi^{fim}_{\mathcal{D},\theta}$, i.e., the conjunction of the formulas (1), (2) and (3). In the sequel, for clarity reasons and when there is no ambiguity, we simply note $\Phi^{cfim}_{\mathcal{D},\theta}$ as $\Phi_{\mathcal{D}}$.

Let us consider again the transaction database $\mathcal{D}$ of Table 1. Assume that the minimum support threshold $\theta = 2$. Then, the closed frequent itemsets are: $\{b\}$, $\{h\}$, $\{a, b\}$, $\{b, d\}$, $\{f, g\}$, $\{a, b, c\}$ and $\{a, b, e\}$.

3 Partition-based Parallel SAT Approach for CFIM

Through an illustrative example, we start by highlighting some weaknesses of the SAT-based encoding of CFIM task described in Section 2.3.

Example 1. Let $\mathcal{D}$ be a transaction database made of two sub-bases $\mathcal{D}_1$ and $\mathcal{D}_2$ built over two disjoint sets of items Ω_1 and Ω_2, respectively. Formally, let $\mathcal{D}_1 = \{T_1, T_2, ..., T_j\}$ and $\mathcal{D}_2 = \{T_{j+1}, T_{j+2}, .., T_n\}$, such that for each $T_i = (tid_i, \mathcal{I}_i)$, we have $\mathcal{I}_i \subseteq \Omega_1$ (resp. $\mathcal{I}_i \subseteq \Omega_2$) for $1 \leqslant i \leqslant j$ (resp. $j+1 \leqslant i \leqslant n$). In the SAT encoding of $CFIM(\mathcal{D}, \theta)$, the propositional variables associated to the items in Ω_1 are used in the encoding of the sub-base $\mathcal{D}_2$ and vis-versa. Indeed, the cover constraint $\Phi_{\mathcal{D}}^{cov}$ (resp. closeness constraint $\Phi_{\mathcal{D}}^{clos}$) involves for each transaction $T_i = (tid_i, \mathcal{I}_i) \in \mathcal{D}$ the propositional variables associated to the items $a \in \Omega \setminus \mathcal{I}_i$ (resp. $a \in \Omega$), whereas $\mathcal{D}$ is made of two independent transaction databases $\mathcal{D}_1$ and $\mathcal{D}_2$. In this worst case illustrative example, the encoding leads to a formula with a high number of large clauses. Indeed, the weakness of SAT-based approaches is the size of the encoding, which for large formulas can outgrow available memory or can make SAT solving otherwise inefficient.

To overcome this drawback, a possible encoding is to express the problem as two independent propositional formulas $\Phi_{\mathcal{D}_1}$ and $\Phi_{\mathcal{D}_2}$. An alternative compact encoding of the itemset mining task on the whole database $\mathcal{D}$ can be expressed as a single propositional formula:

$$\Phi_{\mathcal{D}} = [y \rightarrow (\Phi_{\mathcal{D}_1} \wedge \bigwedge_{l_a \in \Omega_2} \neg l_a)] \wedge [\neg y \rightarrow (\Phi_{\mathcal{D}_2} \wedge \bigwedge_{l_a \in \Omega_1} \neg l_a)] \tag{4}$$

The formula (4) allows us to split in an efficient way the patterns of $\mathcal{D}_1$ and $\mathcal{D}_2$. Assigning y to $true$ (resp. $false$) leads to the enumeration of the models of $\Phi_{\mathcal{D}_1}$ (resp. $\Phi_{\mathcal{D}_2}$), where the propositional variables associated to the items from Ω_2 (resp. Ω_1) are assigned by unit propagation to $false$. Interestingly, the formula (4) can be easily translated into clausal form.

Clearly, the previous example illustrates the motivation behind partitioning the transaction database for an efficient SAT based itemsets enumeration tasks. The benefits are twofold: the size of the encoding can be reduced significantly while improving solving efficiency. Unfortunately, disjoint sub-bases are not very common in real datasets and there recognition is not an easy task. Nevertheless, partitioning can be performed differently. In fact, for a transaction database $\mathcal{D}$ and an item a, the set of frequent closed itemsets can be partitioned into those containing a (models of $\Phi_{\mathcal{D}} \wedge l_a$) and those without a (models of $\Phi_{\mathcal{D}} \wedge \neg l_a$). Interestingly, the models of $\Phi_{\mathcal{D}} \wedge l_a$ are those of $\Phi_{\mathcal{D}\downarrow_a}$. While the models of $\Phi_{\mathcal{D}} \wedge \neg l_a$ correspond to those of $\Phi_{\mathcal{D}\uparrow_a} \wedge \Psi_{\mathcal{D},a}$, where $\Psi_{\mathcal{D},a}$ is defined as:

$$\Psi_{\mathcal{D},a} = \bigvee_{(tid_i, \mathcal{I}_i) \in \mathcal{D},\ a \notin \mathcal{I}_i} q_i$$

Adding $\Psi_{\mathcal{D},a}$ allows to avoid such redundancies. In this way, any model of $\Phi_{\mathcal{D}\uparrow_a} \wedge \Psi_{\mathcal{D},a}$ must correspond to a pattern that covers at least one transaction not

containing a. Let us note that the constraint $\Psi_{\mathcal{D},a}$ can be derived from the closure constraint (3) by assigning l_a to $false$.
As a summary, the assignment of the variable l_a to $true$ allows to restrict the mining process to the transactions containing the item a, while setting l_a to $false$ allows to remove the item a from the database $\mathcal{D}$ with the addition of $\Psi_{\mathcal{D},a}$, so that to avoid redundancy.

Clearly, a less frequent item a_1 might lead to a CNF formula $\Phi_{\mathcal{D}} \wedge l_{a_1}$ of reasonable size. But, $\Phi_{\mathcal{D}} \wedge \neg l_{a_1}$ can remain very huge. Fortunately, the previous partitioning principle can be applied recursively on $\Phi_{\mathcal{D}} \wedge \neg l_{a_1}$ by choosing other items. For example, if we choose a second item a_2, the formula $\Phi_{\mathcal{D}} \wedge \neg l_{a_1}$ can also be divided into $\Phi_{\mathcal{D}} \wedge \neg l_{a_1} \wedge l_{a_2}$ and $\Phi_{\mathcal{D}} \wedge \neg l_{a_1} \wedge \neg l_{a_2}$. Let us note that $\Phi_{\mathcal{D}} \wedge \neg l_{a_1} \wedge l_{a_2}$ is equivalent to $\Phi_{\mathcal{D}\uparrow_{a_1}\downarrow_{a_2}}$. Figure 1 shows this recursive partitioning process.

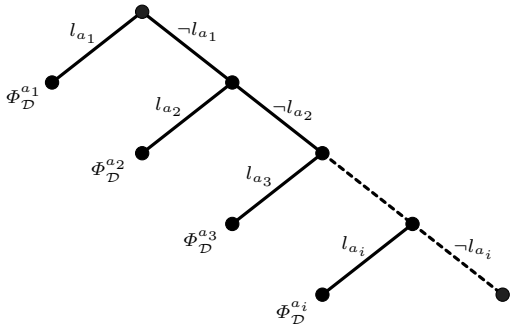


Fig. 1: Item-based guiding paths tree

Finally, if $\Omega = \{a_1, \ldots, a_m\}$, then the models of $\Phi_{\mathcal{D}}$ can be partitioned into disjoint sets of models as stated in Proposition 1.

Proposition 1. *Let $\mathcal{D}$ be a transaction database over Ω. Then,*

$$\mathcal{M}^{\mathcal{P}}(\Phi_{\mathcal{D}}) = \bigcup_{i=1}^{m} \mathcal{M}^{\mathcal{P}}(\Phi_{\mathcal{D}}^{a_i}), \text{ and } \mathcal{M}^{\mathcal{P}}(\Phi_{\mathcal{D}}^{a_i}) \cap \mathcal{M}^{\mathcal{P}}(\Phi_{\mathcal{D}}^{a_j}) = \emptyset \ (1 \leqslant i < j \leqslant m)$$

where $\Phi_{\mathcal{D}}^{a_i} = (\Phi_{\mathcal{D}\downarrow_{a_i}\uparrow_{a_1}\ldots\uparrow_{a_{i-1}}} \wedge \bigwedge_{1\leqslant j<i}(\Psi_{\mathcal{D}\downarrow_{a_i},a_j}))$.

Proof (Sketch). Using the sequence of totally ordered set of items $a_1, a_2, \ldots, a_m$, the formula $\Phi_{\mathcal{D}}$ can be decomposed into a sequence of formulas $\Phi_{\mathcal{D}}^{a_1}$, $\Phi_{\mathcal{D}}^{a_2}$,..., and $\Phi_{\mathcal{D}}^{a_m}$. This partition by the set of item-based guiding paths is complete as depicted in Figure 1. Indeed, starting with the item a_1, the models of $\Phi_{\mathcal{D}}^{a_1}$ correspond to the patterns of $\mathcal{D}$ restricted to transaction containing a_1. The models of $\Phi_{\mathcal{D}}^{a_2}$ correspond to the patterns of $\mathcal{D}$ restricted to transactions containing a_2 while removing the item a_1. The constraint $\Psi_{\mathcal{D}\downarrow_{a_2},a_1} = \bigvee_{(tid_i,\mathcal{I}_i)\in\mathcal{D}\downarrow_{a_2},\ a_1\notin\mathcal{I}_i} q_i$ allows us to generate patterns that cover at least one additional transaction containing a_2 and not a_1, and so on.

Remark 1. As $\Phi_{\mathcal{D}}$ corresponds to the SAT-based encoding of $CFIM(\mathcal{D}, \theta)$, the constraint $\Psi_{\mathcal{D},a}$ avoiding patterns redundancy is derived from the closeness constraint (3) each time an item a is removed from $\mathcal{D}$ (i.e., $\mathcal{D} \uparrow_a$) or equivalently l_a is assigned to $false$ (i.e., $\Phi_{\mathcal{D}} \wedge \neg l_a$). Obviously, the formulas $\Phi_{\mathcal{D}}^{a_i} \wedge l_{a_i}$ and $\Phi_{\mathcal{D}} \wedge l_{a_i} \wedge \neg l_{a_1} \wedge \cdots \wedge \neg l_{a_{i-1}}$ admit the same models over $\mathcal{P}$. However, as our goal is to partition the encoding of $\mathcal{D}$ into several independent formulas of reasonable size, it is better to consider $\Phi_{\mathcal{D}}^{a_i}$, where the encoding is done after restricting $\mathcal{D}$ to transactions containing a_i, removing the items $a_1, \ldots, a_{i-1}$ and adding the constraint $\bigwedge_{1 \leqslant j < i}(\Psi_{\mathcal{D}\downarrow_{a_i}, a_j})$, than propagating the literals $l_{a_i} \wedge \neg l_{a_1} \wedge, \ldots, \wedge \neg l_{a_{i-1}}$ on the formula encoding the whole transaction database.

As explained in Section 2.2, divide-and-conquer is an usual way for parallelizing SAT solvers. This strategy uses the notion of guiding path to split the search space. Each derived sub-formula is solved using a sequential SAT solver running on a particular processor. In fact, partitioning the search space can be done statically or dynamically in order to avoid the idleness of threads. Additionally, workload balancing is another criteria allowing to distribute approximately equal amounts of work among processors over time. In this paper, we consider a static divide-and-conquer approach, i.e., the set of guiding paths are generated in a preprocessing step. Our partition-based approach is based on the set of formulas $\Phi_{\mathcal{D}}^{a_i}$ (see Proposition 1). We denote by g^{a_i} the ith guiding path defined as: $g^{a_i} = \neg l_{a_1} \wedge \ldots \wedge \neg l_{a_{i-1}} \wedge l_{a_i}$. The guiding path g^{a_i} allows us to both restrict the search on a subset of transactions involving only a_i, with the items a_j $(1 \leqslant j < i)$ removed. Each guiding path g^{a_i} leads to a formula $\Phi_{g^{a_i}} = \Phi_{\mathcal{D}} \wedge g^{a_i}$ associated to the ith branch of the item-based guiding paths tree depicted in Figure 1. As $\Phi_{\mathcal{D}}$ is the propositional formula encoding $CFIM(\mathcal{D}, \theta)$, the two formulas $\Phi_{g^{a_i}}$ and $\Phi_{\mathcal{D}}^{a_i}$ are equivalent. Naturally, each guiding path g^{a_i} can be extended with additional items to further reduce the size of the associated formula.

Notice that the total ordering over Ω used to generate the guiding paths greatly impacts the size of the associated sequence of formulas and therefore it can significantly affect the performance. Finding the best ordering for an efficient parallelization is clearly a challenging issue. Our goal is then to decompose the encoding into smaller propositional formulas in order to efficiently balance the load between the different cores. Definitions 5 and 6 formalize this issue.

Definition 5. *Given a transaction database $\mathcal{D}$ on $\Omega = \{a_1, \ldots, a_m\}$. Let σ be a permutation over Ω. We define $G^{\Omega,\sigma} = \{g^{\sigma(a_1)}, \ldots, g^{\sigma(a_m)}\}$ as the set of guiding paths over Ω w.r.t. σ. We also define $\Phi(\mathcal{D}, \sigma)$ as the set of propositional formulas encoding $\mathcal{D}$ w.r.t. σ, i.e., $\Phi(\mathcal{D}, \sigma) = \{\Phi_{g^{\sigma(a_1)}}, \ldots, \Phi_{g^{\sigma(a_m)}}\}$.*

Finding an appropriate permutation is a hard task. In fact, the relevance of such permutation depends on the overall complexity of the model enumeration process on the associated formulas. Considering the size of the formula as a complexity measure, the issue can be formulated as an optimization problem:

Definition 6. *Let $\mathcal{D}$ be a transaction database on Ω. We define the Best Items Decomposition Ordering Problem as the problem of finding the best permutation σ that leads to a set of formulas encoding $\mathcal{D}$ while minimizing the size of*

the largest formula in $\Phi(\mathcal{D},\sigma)$*:* $\sigma^* = \arg\min_{\sigma\in\mathcal{P}erm(\Omega)} max\{|\phi|,\ \phi \in \Phi(\mathcal{D},\sigma)\}$*, where* $\mathcal{P}erm(\Omega)$ *is the set of all permutations over* Ω*.*

The main idea is to reduce the overall computational complexity of the model enumeration task among all the generated formulas under the set of guiding paths. Note that the size of each formula $\Phi_{g^{\sigma(a_i)}}$ depends on the number of transactions involving $\sigma(a_i)$ and on the the frequency of $\sigma(a_j)$ for ($j < i$) in such transactions. From this observation, as a decomposition ordering we consider items from the less frequent to the most frequent one. This static ordering, easy to compute, proved to be a good compromise as shown in our experiments.

Algorithm 1, called paraSatMiner, summarizes the main components of our partition-based parallel SAT approach for CFIM. It takes as input a transaction database $\mathcal{D}$ over Ω, a permutation σ on Ω, a minimum support threshold θ and a fixed number of threads (cores), and returns the set of closed frequent itemsets as output. The number of threads is less or equal to the number of items ($n \leqslant m$). The set of items are sorted in ascending order according to their frequency, i.e., the permutation σ over Ω satisfies the following condition: $\forall 1 \leqslant i < j \leqslant m, \mathcal{S}(\mathcal{D},\{\sigma(a_i)\}) \leqslant \mathcal{S}(\mathcal{D},\{\sigma(a_j)\})$. First, the SAT-based model enumeration solvers are initialized. Each one is associated to a given thread or core i, while initializing the set of models $\mathcal{M}_i$ returned by each thread i to an empty-set (lines 1-4). Now, the n model enumeration solvers are launched in parallel (lines 7-11). For example, the solver i is run successively on the formulas $\Phi_{\mathcal{D}}^{\sigma(a_{i+k\times n})}$ corresponding to the guiding paths (or branches) number $(i + k \times n) \leqslant m$. The set of models computed by the solver i are collected in the set variable $\mathcal{M}_i$. In this way, the solver i is run on the formulas $\Phi_{\mathcal{D}}^{\sigma(a_i)}, \Phi_{\mathcal{D}}^{\sigma(a_{i+n})}, \Phi_{\mathcal{D}}^{\sigma(a_{i+2\times n})}, \ldots, \Phi_{\mathcal{D}}^{\sigma(a_{i+k\times n})}$, with $i + k \times n \leqslant m$. As the items are ordered according to their frequency, the sequences of formulas associated to the n model enumeration solvers are approximately of closer sizes. This allow us to ensure load balancing between the different solvers. Finally, in lines 12-14, the set of closed frequent itemsets are collected by merging the model enumerated by each thread, projected on the variables encoding the itemsets $\mathcal{P}$.

4 Experimental Results

In this section, we evaluate the performance of paraSatMiner. The proposed parallel SAT mining solver is implemented using a model enumeration solver based on MiniSAT [9]. As our approach is based on the enumeration of all models of a propositional formula, we propose an extension of MiniSAT solver based on the DPLL (Davis-Putnam-Logemann-Loveland) procedure [8] to deal with this fundamental problem, marginally explored in the SAT community. More precisely, each time a model is found, a chronological backtracking is performed while inhibiting the restart component to ensure completeness. We also use `OpenMP` as an API that supports multi-platform shared memory multiprocessing programming in C and C++. Let us note that the cardinality constraints representing the support constraint is managed on the fly inside the solver. We also use the well-known MOMS variable ordering heuristic [21].

Algorithm 1: Parallel SAT for Closed Frequent Itemset Mining (paraSat-Miner)

Input: $\mathcal{D}$: a transaction database, $\Omega = \{a_1, \ldots, a_m\}$: a set of items, σ: a permutation over Ω (ordering), θ: a minimum support threshold, n: number of Threads

Output: The set of Closed Fequent Itemsets $CFIM(\mathcal{D}, \theta)$

foreach $i \in \{1, \ldots, n\}$ **do**
  $initEnumSatSolver(i)$;
  $\mathcal{M}_i = \emptyset$;
end
$\mathcal{S} = \emptyset$;
$k = 0$;
\# in parallel;
if $(i + k \times n \leq |\Omega|)$ **then**
  $\mathcal{M}_i \leftarrow \mathcal{M}_i \cup enumModels(enumSatSolver_i, \Phi_{\mathcal{D}}^{\sigma(a_{i+k\times n})})$;
  k++;
end
foreach $i \in \{1, \ldots, n\}$ **do**
  $S = S \cup \mathcal{M}_i^{\mathcal{P}}$;
end
Return S

All the experiments were done on Intel Xeon quad-core machines with 32GB of RAM running at 2.66 Ghz. To evaluate the practical performance of our approach, we run the experiments on different datasets, taken from FIMI [1] and CP4IM[2] repositories. For each instance, we fix a timeout of 1000 seconds of CPU time. Table 2 presents the characteristics of the evaluated datasets. For each instance, we report the number of transactions (#Transactions), the number of items (#Items), the density, and the size of the dataset in bytes.

For our experiments, we denote by `paraSatMiner-*c`[3] our parallel SAT solver based itemset enumeration where $(*)$ indicates the number of threads. We compare the performance of our approach with the following most prominent state-of-the-art CP-based CFIM systems: `ClosedPattern`[4,5] and `CoverSize`[6]. As mentioned previously, `ClosedPattern` and `CoverSize` encode the closeness constraint as global constraints. These CP mining systems are implemented in C++ and Java using the or-tools solver [22] (or `choco` for the recent version) and the OscaR solver [30], respectively. In addition, we consider `LCM`[7] known as the best specialized solver for frequent (closed) itemset mining. We perform two

[1] `http://fimi.ua.ac.be/data/`
[2] `http://dtai.cs.kuleuven.be/CP4IM/datasets/`
[3] `http://www.cril.univ-artois.fr/decMining/`
[4] `http://www.lirmm.fr/~lazaar/cpminer.html`
[5] `https://lemierev.users.greyc.fr/closedpattern/`
[6] `https://www.info.ucl.ac.be/~pschaus`
[7] `http://research.nii.ac.jp/~uno/code/lcm.html`

kinds of experiments. In the first evaluation, we carry out a comparison with all the considered itemset mining systems. In this case, `paraSatMiner` is run using 1 core as a sequential solver. In the second, we show that the performance of our `paraSatMiner` can be improved by increasing the number of threads. In this case, we vary the number of cores from 1 to 8.

Instance	#Transactions	#Items	Density	Size
chess	3196	75	49.0%	340K
mushroom	8124	119	19.0%	516K
BMS-WebView-1	59601	497	0.5%	940K
T10I4D100K	100000	870	1.0%	3.9M
retail	88162	16470	0.06%	4,2M
connect	67558	129	35.62%	8.9M
T40I10D100K	100000	942	4.31%	15M
pumsb	49046	2113	3.0%	16,7M
kosarak	990002	41267	0.01%	32M
accidents	340183	468	7.0%	34M

Table 2: Data Characteristics

4.1 Sequential Evaluation

Table 3 reports the comparative results of the considered systems using different minimum support threshold values. For each dataset, the number of models (patterns) and the total CPU time (in seconds) are given. For `CoverSize`, we also mention in parenthesis the solving CPU time without including the time spent for loading and encoding the instance. (`--`) means that the solver is not able to finish the enumeration under the fixed time out. According to the results, our ParaSatMiner-1c (with 1 core) solver outperforms considerably the CP approaches on almost all the datasets with several factors of magnitudes of gain. For instance, our approach allows to enumerate all the models of `connect` data in 180 seconds for $\theta = 5000$ when both ClosedPattern and CoverSize timed out. Except for `T10I4D100K` and $\theta = 50$, where CoverSize is the best. For all the remaining tested values, `ParaSatMiner-1c` is clearly the best. Interestingly, on the `kosarak` data with 990002 transactions the two CP systems are not able to enumerate the whole set of models under the time limit for all θ values. For the comparison with specialized algorithms, even if our approach reduced the performance gap, the `LCM` algorithm remains the best system.

Table 3: paraSatMiner vs (ClosedPattern, CoverSize, LCM)

Instance	θ	Closed Pattern	Cover Size	paraSat Miner-1c	LCM	#Models
	80	–	265.10 (42.84)	14.06	0.21	$> 8.10^3$
	60	–	295.47 (72.04)	18.07	0.24	$> 1.10^4$
retail	40	–	334.23 (113.14)	25.33	0.28	$> 2.10^4$
	20	–	439.94 (216.68)	41.93	0.35	$> 5.10^4$
	10	–	586.16 (361.71)	76.49	0.56	$> 1.10^5$
	40000	7.54	14.95 (9.67)	7.25	0.17	$> 7.10^4$
	20000	50.22	75.48 (70.40)	22.73	0.55	$> 5.10^5$
connect	10000	526.43	431.19 (426.34)	68.88	2.68	$> 3.10^6$
	5000	–	–	179.17	9.82	$> 1.10^7$

Continued on next page

Table 3: paraSatMiner vs (ClosedPattern, CoverSize, LCM)

Instance	θ	Closed Pattern	Cover Size	paraSat Miner-1c	LCM	#Models
	2000	1.51	1.22 (0.51)	0.25	0.04	$\simeq 7.10^4$
	1500	6.30	4.09 (3.38)	0.8	0.20	$> 5.10^5$
	1000	51.35	28.62 (27.93)	5.52	1.75	$> 4.10^6$
chess	500	577.29	311.47 (310.74)	49.50	18.25	$> 45.10^6$
	250	–	–	186.11	72.96	$\simeq 2.10^8$
	100	–	–	484.41	215.30	$> 5.10^8$
	100000	101.68	145.96 (81.84)	74.14	1.72	$\simeq 1.10^5$
accidents	80000	319.25	283.98 (220.86)	141.29	3.53	$\simeq 4.10^5$
	60000	–	866.21 (804.33)	299.94	5.66	$> 1.10^6$
	40000	–	–	735.39	10.59	$\simeq 6.10^6$
	40000	20.99	389.43 (1.83)	4.78	0.13	$> 2.10^4$
	35000	103.20	325.42 (19.54)	10.33	0.25	$\simeq 2.10^5$
pumbs	30000	434.25	404.26 (97.85)	27.72	0.59	$\simeq 9.10^5$
	25000	–	994.35 (690.09)	147.64	3.18	$\simeq 6.10^6$
	20000	–	–	669.93	17.69	$> 3.10^7$
	10000	4.16	51.43 (0.145)	3.15	0.34	$\simeq 1.10^2$
	8000	5.08	51.13 (0.352)	4.07	0.54	$> 1.10^2$
T40I10D100K	6000	10.38	52.39 (0.912)	6.43	0.79	$> 2.10^2$
	4000	30.51	53.26 (2.221)	9.59	1.00	$> 4.10^2$
	2000	144.89	58.22 (7.378)	22.74	1.64	$> 1.10^3$
	500	106.87	24.04 (6.91)	2.90	0.35	$> 1.10^3$
	400	147.14	25.56 (8.32)	3.32	0.37	$> 1.10^3$
T10I4D100K	300	217.40	27.73 (10.69)	4.31	0.40	$> 4.10^3$
	200	314.17	29.12 (11.98)	6.16	0.44	$> 1.10^4$
	100	497.10	32.40 (15.14)	14.66	0.48	$> 2.10^4$
	50	–	45.05(27.80)	76.37	0.58	$> 4.10^4$
	70	–	211.71 (97.76)	31.66	0.82	$\simeq 6.10^3$
	40	–	233.34 (120.48)	34.64	0.85	$> 6.10^3$
Online-Retail	10	–	253.64 (141.61)	39.94	0.82	$\simeq 8.10^3$
	5	–	267.96 (156.74)	41.79	0.90	$> 8.10^3$
	4000	–	–	29.37	1.32	$> 2.10^3$
	3000	–	–	39.98	1.61	$> 4.10^3$
Kosarak	2000	–	–	65.12	2.16	$> 3.10^4$
	1000	–	–	145.65	3.43	$\simeq 5.10^5$
	250	1.14	2.54 (0.50)	1.33	0.05	$> 1.10^4$
	100	1.64	1.91 (1.07)	1.94	0.06	$> 3.10^4$
mushroom	50	2.70	2.47 (1.63)	2.44	0.09	$> 5.10^4$
	25	2.82	3.16 (2.27)	2.90	0.11	$\simeq 8.10^4$
	5	5.57	4.20 (3.33)	3.90	0.17	$> 1.10^5$
	48	203.51	220.67 (1.43)	8.69	0.06	$> 9.10^3$
	36	833.76	220.20 (3.32)	10.08	0.16	$> 6.10^4$
BMS-WebView-1	34	–	219.19 (4.79)	10.50	0.21	$> 8.10^4$
	32	–	227.80 (8.19)	10.89	0.32	$> 1.10^5$
	30	–	231.89 (14.88)	11.37	0.53	$> 1.10^5$

4.2 Parallel Evaluation

On the parallel side, we perform the same experiments by varying the number of cores from 1 to 8 and by considering several minimum support threshold values. Figure 2 shows the obtained results on a representative sample of dataset. A first remark is that the parallel approach allows us to reduce considerably the

computation time. By increasing the number of cores, the time always decreases. The impressive gain is obtained from 1 to 2 cores as shown by Figure 2. For the transition from 4 to 8 cores, the gain is not very impressive. Overall, the strategy used to generate guiding paths by considering the less frequent items first is successful. For instance for `pumsb` data and $\theta = 2000$, the time is reduced from 700 seconds with 1 core to less than 200 seconds with 8 cores.

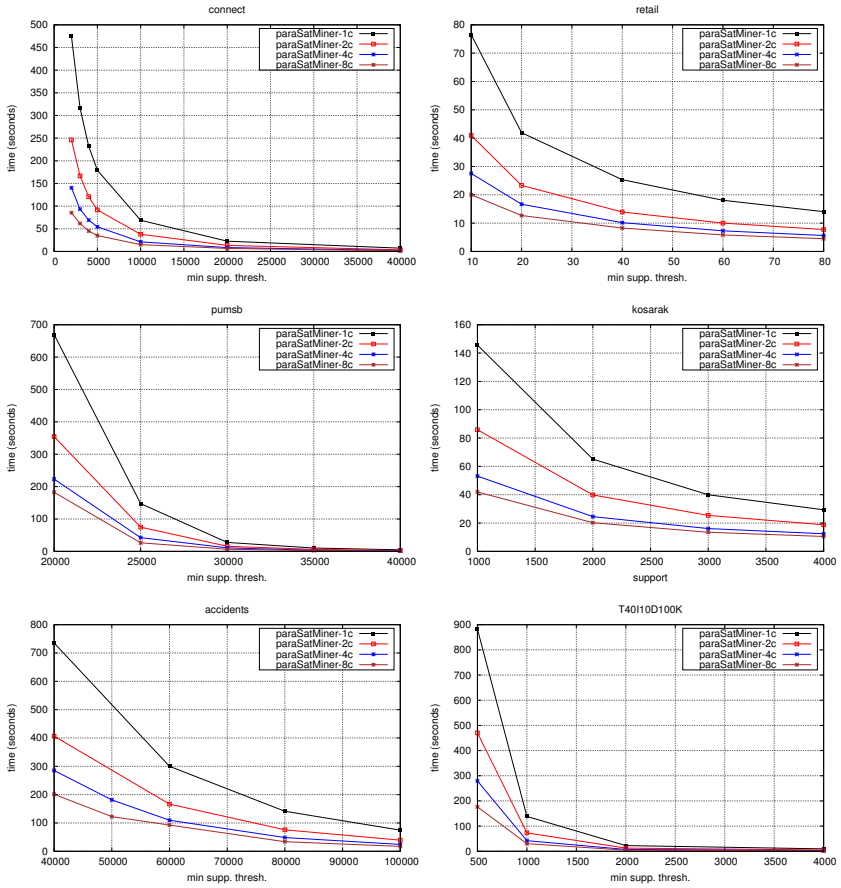


Fig. 2: Performance gain w.r.t. the number of cores

4.3 Load Balancing

Finally, to assess the suitability of our load balancing strategy, we consider the datasets `pumsb` and `T10I4D100K` and we provide an empirical analysis of the number of models discovered by the different cores. Using the 4 (resp. 8) cores configuration, for each value of θ we report the average number of models among the 4 (resp. 8) cores (the curve), while mentioning the minimum and the maximum number of models (the interval) (see Figure 3). We note that the load

balancing strategy is better when the minimum and maximum number of models among those returned by the different cores are close to the average number of models. As we can observe, for `T10I4D100K`, our strategy is clearly interesting. Indeed, the relative load unbalancing is very limited. All the cores find approximatively the same number of models. However, the load unbalancing is relatively high for `pumsb`, mainly for the transition between 4 to 8 cores.
Overall, our guiding paths generation strategy allows to balance the number of models between the different threads. These results indicate that our method is both efficient and scalable.

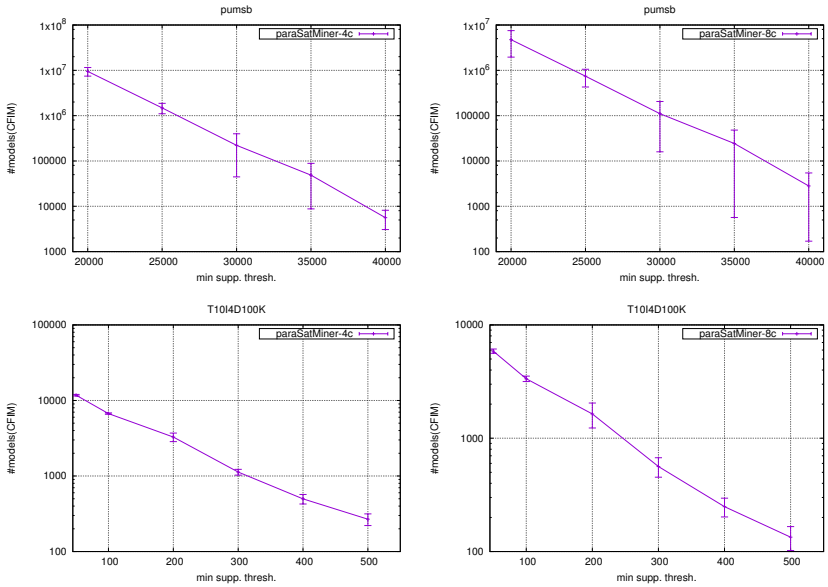


Fig. 3: Load unbalancing between cores

5 Conclusion

In this paper, we proposed a parallel SAT-based framework for CFIM. We show that partitioning the formula using predefined guiding paths allows to push forward the performance of SAT-based itemset mining frameworks. Even in the sequential configuration, our approach outperforms the existing CP-based approaches while reducing considerably the time needed to compute all the itemsets. We have shown that such approach scales well and presents good load balancing among the different cores.

This work can be extended in different ways. First, we would like to consider a dynamic partitioning strategy. We also plan to implement a distributed version to handle very large datasets. Last, we plan to tackle other DM problems like discriminative itemsets, association rules mining, and skypatterns tasks.

References

1. R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD International Conference on Management of Data*, pages 207–216, 1993.
2. O. Bailleux and Y. Boufkhad. Efficient CNF encoding of boolean cardinality constraints. In *International Conference on Principles and Practice of Constraint Programming CP*, pages 108–122, 2003.
3. Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal. Mining minimal non-redundant association rules using frequent closed itemsets. In *Computational Logic*, volume 1861, pages 972–986, 2000.
4. C. Borgelt. Frequent item set mining. *Wiley Int. Rev. Data Min. and Knowl. Disc.*, 2(6):437–456, Nov. 2012.
5. A. Boudane, S. Jabbour, L. Sais, and Y. Salhi. A sat-based approach for mining association rules. In *IJCAI*, pages 2472–2478, 2016.
6. A. Boudane, S. Jabbour, L. Sais, and Y. Salhi. Clustering complex data represented as propositional formulas. In *Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part II*, pages 441–452, 2017.
7. T. Dao, K. Duong, and C. Vrain. Constrained clustering by constraint programming. *Artif. Intell.*, 244:70–94, 2017.
8. M. Davis, G. Logemann, and D. Loveland. A machine program for theorem proving. *Communications of the ACM*, 5:394–397, 1962.
9. N. En and N. Srensson. An extensible sat-solver. In *Proceedings of the Sixth International Conference on Theory and Applications of Satisfiability Testing (SAT'03)*, pages 502–518, 2002.
10. M. Ganji, J. Bailey, and P. J. Stuckey. A declarative approach to constrained community detection. In *International Conference on Principles and Practice of Constraint Programming*, pages 477–494, 2017.
11. M. Gebser, T. Guyet, R. Quiniou, J. Romero, and T. Schaub. Knowledge-based sequence mining with ASP. In *International Joint Conference on Artificial Intelligence*, pages 1497–1504, 2016.
12. T. Guns, A. Dries, G. Tack, S. Nijssen, and L. D. Raedt. Miningzinc: A modeling language for constraint-based mining. In *International Joint Conference on Artificial Intelligence*, pages 1365–1372, 2013.
13. T. Guns, S. Nijssen, and L. D. Raedt. Itemset mining: A constraint programming perspective. *Artificial Intelligence*, 175(12-13):1951–1983, 2011.
14. Y. Hamadi, S. Jabbour, and L. Sais. Manysat: a parallel SAT solver. *JSAT*, 6(4):245–262, 2009.
15. R. Henriques, I. Lynce, and V. M. Manquinho. On when and how to use sat to mine frequent itemsets. *CoRR*, abs/1207.6253, 2012.
16. S. Jabbour, N. Mhadhbi, B. Raddaoui, and L. Sais. A sat-based framework for overlapping community detection in networks. In *Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 786–798, 2017.
17. S. Jabbour, L. Sais, and Y. Salhi. A pigeon-hole based encoding of cardinality constraints. *TPLP*, 13(4-5-Online-Supplement), 2013.
18. S. Jabbour, L. Sais, and Y. Salhi. The top-k frequent closed itemset mining using top-k SAT problem. In *ECML/PKDD*, pages 403–418, 2013.
19. S. Jabbour, L. Sais, and Y. Salhi. Decomposition based SAT encodings for itemset mining problems. In *Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 662–674, 2015.

20. S. Jabbour, L. Sais, and Y. Salhi. Mining top-k motifs with a SAT-based framework. *Artif. Intell.*, 244:30–47, 2017.
21. R. G. Jeroslow and J. Wang. Solving propositional satisfiability problems. *Annals of Mathematics and Artificial Intelligence*, 1:167–187, 1990.
22. N. Lazaar, Y. Lebbah, S. Loudni, M. Maamar, V. Lemière, C. Bessiere, and P. Boizumault. A global constraint for closed frequent pattern mining. In *International Conference on Principles and Practice of Constraint Programming*, pages 333–349, 2016.
23. Y. C. Lin, C. Wu, and V. S. Tseng. Mining high utility itemsets in big data. In *Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 649–661, 2015.
24. L. Liu, E. Li, Y. Zhang, and Z. Tang. Optimization of frequent itemset mining on multiple-core processor. In *International Conference on Very Large Data Bases*, 2007.
25. S. Moens, E. Aksehirli, and B. Goethals. Frequent itemset mining for big data. In *IEEE International Conference on Big Data*, pages 111–118, 2013.
26. B. Négrevergne and T. Guns. Constraint-based sequence mining using constraint programming. In *International Conference on Integration of AI and OR Techniques in Constraint Programming*, pages 288–305, 2015.
27. B. Négrevergne, A. Termier, J. Méhaut, and T. Uno. Discovering closed frequent itemsets on multicore: Parallelizing computations and optimizing memory accesses. In *International Conference on High Performance Computing & Simulation*, pages 521–528, 2010.
28. L. D. Raedt, T. Guns, and S. Nijssen. Constraint programming for itemset mining. In *ACM SIGKDD*, pages 204–212, 2008.
29. A. Savasere, E. Omiecinski, and S. B. Navathe. An efficient algorithm for mining association rules in large databases. In *International Conference on Very Large Data Bases*, pages 432–444, 1995.
30. P. Schaus, J. O. R. Aoga, and T. Guns. Coversize: A global constraint for frequency-based itemset mining. In *International Conference on Principles and Practice of Constraint Programming*, pages 529–546, 2017.
31. T. Schubert, M. D. T. Lewis, and B. Becker. Pamiraxt: Parallel SAT solving with threads and message passing. *JSAT*, 6(4):203–222, 2009.
32. G. Tseitin. On the complexity of derivations in the propositional calculus. In *Studies in Mathematics and Mathematical Logic*, pages 115–125, 1968.
33. S. Wang, Y. Yang, Y. Gao, G. Chen, and Y. Zhang. Mapreduce-based closed frequent itemset mining with efficient redundancy filtering. In *IEEE International Conference on Data Mining Workshops, ICDM*, pages 449–453, 2012.
34. J. P. Warners. A linear-time transformation of linear inequalities into conjunctive normal form. *Information Processing Letters*, 68(2):63–69, 1998.
35. O. R. Zaïane, M. El-Hajj, and P. Lu. Fast parallel association rule mining without candidacy generation. In *IEEE International Conference on Data Mining*, pages 665–668, 2001.
36. M. J. Zaki. Mining non-redundant association rules. *Data Min. Knowl. Discov.*, 9(3):223–248, 2004.
37. H. ZHANG, M. P. BONACINA, and J. HSIANG. Psato: a distributed propositional prover and its application to quasigroup problems. *Journal of Symbolic Computation*, 21(4):543 – 560, 1996.
38. M. Zitouni, R. Akbarinia, S. B. Yahia, and F. Masseglia. Massively distributed environments and closed itemset mining: The DCIM approach. In *International Conference on Advanced Information Systems Engineering*, pages 231–246, 2017.