Rapport d'activité de l'axe AIC (2013-2018)

Centre de Recherche en Informatique de Lens

février 2019

Produits et activités de recherche

Bilan scientifique

Nous articulons la présentation des travaux réalisés au sein de l'axe AIC autour de cinq thématiques principales : compilation de connaissances, fondements pour le raisonnement par contraintes, outils pour la modélisation, explications de conflits, fouille de données - apprentissage et contraintes.

Compilation de connaissances

La compilation de connaissances a pour objectif l'identification et l'étude de langages de représentation de connaissances dans la perspective des compromis temps/espace possibles, la conception, le développement et l'expérimentation d'outils de traduction entre langages, mais aussi d'outils de raisonnement ou de prise de décision opérant à partir de représentations compilées. Compiler une représentation consiste à construire une autre représentation (très souvent équivalente à la première) lors d'une phase de pré-traitement, puis à utiliser la représentation compilée obtenue pour répondre plus efficacement à certaines requêtes qui sont difficiles du point de vue calculatoire. La compilation est profitable quand le temps mis à compiler peut être amorti en considérant un nombre suffisant de requêtes. Construire une carte de compilation qui identifie les requêtes traitables (et celles qui ne le sont pas) et précise l'efficacité spatiale des langages de compilation permet de faire un choix informé parmi les langages cibles possibles.

Le thème de la compilation de connaissances est étudié au CRIL depuis plus de vingt ans (voir (Marquis, 2015)). Diverses contributions ont été apportées lors du précédent contrat (c'est-à-dire, depuis 2013), tant d'un point de vue théorique que pratique. Sur ce dernier point, un petit groupe de collègues intéressés par le sujet participe au projet Compile! du laboratoire, qui donne lieu à un développement logiciel conséquent. En particulier, les codes exécutables de divers programmes dont les compilateurs (évoqués ci-dessous) cnf2eadt, d4, dmc, cn2mddg et les préprocesseurs pmc et B+E sont disponibles (mais on trouve aussi divers jeux d'essai au format CNF ou donnés sous forme de réseaux de contraintes au format XCSP).

- **Langages de compilation** Dans (Fargier et Marquis, 2014) nous avons défini et étudié de nouveaux langages de compilation propositionnels, obtenus en permettant des transformations implicites (la disjonction et l'oubli) sur des formules issues de langages incomplets pour lesquels le test de cohérence est en temps polynomial. Dans (Le Berre et al., 2018), nous avons analysé les langages des contraintes pseudo-booléennes et des contraintes de cardinalité selon les critères retenus dans la carte de compilation des langages propositionnels. Ces langages offrent essentiellement les mêmes requêtes et transformations traitables que le langage CNF, mais avec une efficacité spatiale meilleure. Dans (ECAI'14), nous avons proposé d'étendre le langage des diagrammes de décision binaires décomposables (aussi appelé decision-DNNF) pour y intégrer des symétries entre littéraux. Cette extension conduit à un langage ayant une efficacité spatiale plus grande, le prix à payer étant la perte d'une opération de conditionnement réalisable en temps polynomial dans le cas général (un conditionnement restreint restant traitable, ce qui suffit toutefois pour certains problèmes). Dans (IJCAI'13) nous avons proposé un cadre formel permettant la comparaison (selon les principes de la carte de compilation) de langages de représentation hétérogènes. Dans (IJCAI'13) et (Fargier et al., 2014) nous avons proposé et étudié de nouveaux langages de compilation pour la représentation de fonctions multivariées à valeurs dans le domaine d'une structure de valuation qui reste « simple » d'un point de vue algébrique (un monoïde) et assure l'existence d'une forme canonique des fonctions ; le conditionnement et l'optimisation sont en temps polynomial pour les formules de ces langages, ce qui est essentiel pour les applications visées, en particulier la configuration (le thème central du projet ANR BR4CP auquel nous avons participé). Dans (Lagniez et Marquis, 2017a), nous avons proposé un nouveau compilateur vers le langage decision-DNNF, appelé d4. En plus des composants des compilateurs existants, d4 tire parti d'une approche de décomposition dynamique (quoique parcimonieuse) basée sur la

construction d'une partition de l'hypergraphe dual de la formule CNF donnée en entrée. Certaines règles de simplification sont utilisées pour minimiser le temps passé dans les étapes de partition et pour promouvoir la qualité des décompositions. Dans (Lagniez, Marquis et Szczepanski, 2018), nous avons proposé un compilateur distribué vers le langage decision-DNNF, appelé dmc. Ce compilateur s'appuie sur d4 et suit une stratégie de répartition des charges qui est proche du work stealing. Empiriquement, les gains de performance obtenus par distribution du calcul sont souvent très significatifs. Dans (Koriche, Lagniez et Marquis, 2015), nous avons présenté le langage MDDG des diagrammes de décision décomposables multivalués qui généralise le langage decision-DNNF à des variables à domaines finis qui ne sont pas forcément booléennes. Nous avons décrit un compilateur de réseaux de contraintes, appelé cn2mddg, ciblant le langage MDDG et montré son intérêt pratique en comparaison à une approche procédant par encodage du réseau sous forme CNF puis compilation de cet encodage en une représentation en decision-DNNF. Dans (CP'17), nous avons comparé les performances d'une vingtaine d'heuristiques de choix de variable (certaines étant classiques, d'autres étant de nouvelles heuristiques proposées) quand elles sont utilisées au sein du compilateur cn2mddg dans l'objectif de calculer des diagrammes de décision multivalués, décomposables ou non. Pour finir, dans (Audemard, Lagniez et Simon, 2013), nous utilisons une approche incrémentale pour répondre à des requêtes sur des bases de connaissances. Plutôt que de compiler ces bases, nous réalisons ces requêtes à la volée et profitons des appels précédents pour amortir le coût de chacune d'elles.

- **Comptage de modèles/solutions** Ces dernières années, nous avons en particulier ciblé la requête du comptage des solutions pour son importance (comme reflétée par le théorème de Toda) et sa difficulté (problème #P-complet), lorsque les représentations traitées sont de nature booléenne et données par des formules propositionnelles CNF ou des réseaux de contraintes à domaines finis. Dans (IJCAI'13) nous avons introduit le langage des arbres de décision affine et avons montré que ce langage permet, en temps polynomial, le comptage des modèles de la formule représentée ; nous avons aussi développé un compilateur appelé cnf2adt qui se révèle particulièrement efficace, comparé à certains compteurs de modèles dédiés (pour certaines instances, le temps de compilation est amorti dès la première requête de comptage de modèles). Au-delà des algorithmes de compilation, nous avons conçu et développé des méthodes de pré-traitement, qui visent à réduire les représentations fournies en entrée dans le but de simplifier les traitements réalisés à partir de celles-ci, sans pour autant passer par la construction de nouvelles représentations. Nous nous sommes particulièrement concentrés sur la tâche de comptage de modèles. Ainsi dans (AAAI'14) et (Lagniez et Marquis, 2017b), nous avons présenté un préprocesseur appelé pmc qui inclut de nombreuses techniques de prétraitement élémentaires, notamment la réduction des occurrences, la vivification, l'identification du backbone, ainsi que l'équivalence, l'identification et le remplacement des portes ET et XOR. Nous avons montré que l'utilisation de ce préprocesseur permet d'améliorer notablement les temps de calcul requis par le comptage de modèles pour beaucoup d'instances de familles différentes. Vu l'efficacité apportée par l'identification et le remplacement de portes, nous avons ensuite conçu un algorithme permettant l'identification et le remplacement implicites de portes quelconques, en exploitant de « vieux » résultats bien connus en logique classique et portant sur la définissabilité (le théorème de Beth et la méthode de Padoa). Le préprocesseur correspondant, appelé B+E et décrit dans (Lagniez, Lonca et Marquis, 2016), a montré très largement son intérêt pratique. Dans (Chen et Mengel, 2016), nous nous sommes intéressés à la complexité de compter le nombre de solutions d'une formule existentielle positive sur une structure finie et avons prouvé un théorème de trichotomie sur ces classes de formules lorsque l'arité est bornée. Ce théorème généralise et unifie plusieurs résultats existants concernant la complexité des requêtes conjonctives et les unions de requêtes conjonctives. Ce travail a été étendu dans (Chen et Mengel, 2017). Nous avons présenté une extension de la logique du premier ordre avec laquelle les algorithmes pour ce problème de comptage peuvent être naturellement et facilement exprimés, dans un sens qui est clarifié et motivé par le désir de comprendre quels sont les cas faciles de ce problème.

- **Formes normales négatives** Dans (Bova et al., 2016), nous exhibons une relation entre la taille des circuits décomposables en forme normale négative (DNNF) et la complexité de communication multi-partition. Cela nous permet de transformer directement des bornes inférieures existantes dans le domaine de la complexité de communication en des bornes inférieures sur la taille des représentations par DNNFs. Nous utilisons cette approche pour prouver une séparation exponentielle entre les DNNFs et les DNNFs déterministes ainsi qu'entre les CNFs et les DNNFs. Nous utilisons ces résultats dans (SAT'16), pour montrer des bornes inférieures non conditionnelles sur la taille des circuits DNNF encodant des formules CNF restreintes à plusieurs mesures de graphes. Dans (ICALP'17), nous étudions le problème d'énumération des valuations satisfaisant un circuit tout en limitant le délai, c'est-à-dire le temps nécessaire pour calculer chaque valuation successive, en nous concentrant sur la classe des circuits d-DNNF. Nous proposons un algorithme pour ces circuits qui énumère les valuations avec un prétraitement linéaire et un délai linéaire dans la taille de chaque valuation. Notre cadre d'énumération efficace s'applique ainsi à tous les problèmes dont les solutions peuvent être compilées dans une d-DNNF. En particulier, nous l'utilisons pour redémontrer des résultats classiques dans la théorie des bases de données, pour les bases de données factorisées et pour l'évaluation MSO.
  Dans (ICDT'18), nous montrons que ce cadre est aussi adaptable en cas de mise à jour des données. Nous nous

sommes également intéressés à la requête d'optimisation sous contraintes DNNF d'une fonction de coût donnée à la volée. Cette requête apparaît naturellement dans de nombreux problèmes, en particulier en configuration de produits de nature combinatoire. Alors que la minimisation linéaire était déjà connue comme traitable sous contraintes DNNF, nous avons montré dans (ECAI'16) que la minimisation quadratique et la minimisation sous-modulaire sont NP-difficiles en général, mais FPT pour différents sous-ensembles de DNNF. En particulier, le caractère FPT de la minimisation sous-modulaire a été établi pour un paramètre naturel capturant la dissemblance structurelle entre la fonction de coût considérée et la représentation DNNF des contraintes.

- **Apprentissage par compilation** Les problèmes d'apprentissage séquentiel sous contraintes symboliques apparaissent dans de nombreuses applications (systèmes de recommandation, gestion d'emplois du temps, allocation de ressources, problèmes de congestion dans les réseaux, etc.). Dans ce contexte, l'espace des décisions possibles est de nature combinatoire et il est impossible d'appliquer directement des stratégies de prédiction séquentielle (*Hedge*, *Stochastic Gradient Descent*, *Online Mirror Descent*) pour traiter le problème. Afin de pallier cette difficulté, nous avons utilisé dans (Koriche, 2018) une compilation des contraintes symboliques sous forme d'un circuit d-DNNF, et modifié les stratégies de prédiction afin d'apprendre de manière rapide et efficace en présence de telles contraintes. En particulier, nous avons fait tomber la complexité temporelle de la stratégie *Online Mirror Descent* (connue comme étant optimale), de $O(n^6)$ à $O(n^2)$ (modulo une complexité linéaire en la taille du circuit).

Raisonnement par contraintes (SAT/CSP) : fondements

- **Cohérences locales et algorithmes de filtrage** Les cohérences locales sont des propriétés qui portent sur les réseaux de contraintes et qui sont utiles pour filtrer l'espace de recherche (typiquement, en détectant puis éliminant des valeurs incohérentes). Au cours de la période de référence, nous avons contribué à ce domaine de recherche « historique » du CRIL en proposant de nouvelles cohérences et/ou algorithmes de filtrage. Tout d'abord, dans (CP'13) et (Bessiere, Fargier et Lecoutre, 2016), nous avons proposé un algorithme complet pour les systèmes de configuration de produits. Il repose sur une propriété appelée cohérence inverse globale (GIC pour *Global Inverse Consistency*), assurant que toute valeur proposée à l'utilisateur lors d'un processus de configuration de produit puisse conduire à un produit existant. Cette approche a été appliquée avec succès au processus de configuration d'automobile de la firme Renault. Ensuite, dans (AAAI'13), nous avons proposé un algorithme (basée sur STR, la méthode de réduction tabulaire qui a montré son efficacité pour le filtrage des contraintes définies en extension) assurant l'inter-cohérence complète (FPWC), une cohérence plus forte que la cohérence d'arc (GAC pour *Generalized Arc Consistency*) et maxRPWC (*max Restricted Pairwise Consistency*). Finalement, dans (Paparrizou et Stergiou, 2017) nous avons proposé une famille de singleton cohérences plus faibles que la variante actuelle de SAC (*Singleton Arc Consistency*) obtenant des résultats qui surpassent les techniques de propagation existantes. En ce qui concerne le filtrage des contraintes définies en extension (contraintes table), nous avons proposé un nouvel algorithme STR3 (Lecoutre, Likitvivatanavong et Yap, 2015), qui est assez complémentaire de STR2 (pour lequel nous avons étudié des extensions dans (Constraints'15)) : lorsque la réduction tabulaire est faible (c'est-à-dire, lorsque les tables ne se réduisent pas rapidement), STR3 prend le pas sur STR2. Nous avons, plus récemment, proposé un algorithme de filtrage surpassant les algorithmes existants (CP'16) et qui combine des techniques issues de plus de 10 ans de recherche sur les contraintes table. Ce nouvel algorithme, testé de manière intensive, est clairement le plus rapide de sa génération. Il est adopté aujourd'hui par la plupart des solveurs (Choco, Gecode, JacoP, etc.). Enfin, pour finir, nous avons proposé (CP'14) une approche de filtrage générale, SND en abrégé pour *Scoring-based Neighborhood Dominance*, pour le problème d'isomorphisme de sous-graphe.
- **Recherche de solutions** Pour conduire la recherche de solutions (sur la base d'une approche complète), plusieurs ingrédients sont particulièrement importants. Parmi eux, on trouve l'heuristique de branchement, l'enregistrement de no-goods, et l'utilisation du parallélisme. Nous nous sommes toujours intéressés aux approches génériques permettant de guider la recherche (voir, par exemple, nos travaux plus anciens sur dom/wdeg et last-conflict). Récemment, nous avons proposé une variante de last-conflict qui prend la main sur l'heuristique de recherche plutôt que de la réparer, en minimisant récursivement les ensembles de variables en conflit. Cette approche générique, appelée *Conflict Ordering Search* (CP'15), se comporte particulièrement bien sur les problèmes d'ordonnancement, pour lesquels elle est compétitive face aux approches dédiées. L'utilité de l'enregistrement de nogoods (instanciations partielles globalement incohérentes) lors de la recherche, notamment pour le problème SAT, n'est plus à démontrer. Depuis 2009, le solveur SAT glucose est développé conjointement par un collègue du CRIL et un membre du LaBRI : glucose permet une mesure judicieuse des clauses (nogoods) apprises. Ce solveur fait toujours partie des solveurs de l'état de l'art et a remporté de nombreux prix lors des compétitions SAT ces dernières années. Dans (SAT'16), nous avons montré que, parmi les problèmes SAT proposés actuellement, nombreux sont ceux qui exhibent des comportements extrêmes, des aberrations. Partant de quelques indicateurs très simples, nous avons montré comment laisser glucose choisir

parmi quelques stratégies afin de résoudre efficacement ces problèmes extrêmes. (AoR'16) présente des travaux autours de l'apprentissage de clauses à partir des conflits. Dans (CP'17), nous nous sommes intéressés pour le problème CSP à l'enregistrement de nogoods pouvant être extraits systématiquement lors du redémarrage d'un algorithme de recherche complet. Dans ce contexte, nous avons proposé plusieurs techniques de simplification et de combinaison de nogoods, dans le but d'accroître leur capacité de filtrage. L'avènement des processeurs multi-cœurs et du cloud-computing nous a conduit à développer des solveurs SAT parallèles. Nous avons tout d'abord travaillé autour de la parallélisation de glucose. Dans (SAT'14), nous avons montré comment sélectionner attentivement les clauses échangées entre deux solveurs concurents. Cette version parallèle de glucose (nommée syrup) fait toujours partie des meilleurs solveurs SAT parallèles. Dans le cadre du projet ANR SATAS, nous avons proposé des solveurs SAT capables de travailler dans le cloud. Dans (CP'16), nous avons présenté Ampharos, un nouveau solveur SAT parallèle fondé sur le paradigme « *diviser pour mieux régner* ». Dans (SAT'17), nous avons étendu la version parallèle de syrup aux environnements distribués. Un livre présentant un large aperçu du parallélisme dans les formalismes de raisonnement basés sur les contraintes a été co-édité par Youssef Hamadi et Lakhdar Saïs (Hamadi et Saïs, 2018). Pour finir, dans (Grégoire, Lagniez et Mazure, 2013) et (ECAI'14), plusieurs questions liées à la dynamique des réseaux de contraintes ont été explorées à la fois conceptuellement et algorithmiquement. Nous avons notamment étudié comment préserver un ensemble de solutions partielles lorsque de nouvelles contraintes sont introduites dans le réseau. Nous avons également proposé et exploré une notion de contraintes de relaxation, dites permissives, lesquelles visent à élargir l'ensemble initial de solutions du réseau.

- **Recherche de structures** Dans (LPAR'18), nous avons cherché à transfomer des instances SAT en instances du problème de la recherche d'un stable maximum pour définir de nouvelles classes traitables. Dans (JELIA'14), nous avons proposé une nouvelle approche pour énumérer les impliquants premiers (PI) d'une formule booléenne sous forme normale conjonctive (CNF). Elle est basée sur un codage de la formule d'entrée en une nouvelle formule dont les modèles correspondent à l'ensemble des PIs de la théorie originale. Cette première approche d'énumération des PIs est ensuite améliorée par une utilisation originale des fonctions ou portes booléennes habituellement impliquées dans de nombreuses instances CNF codant des problèmes du monde réel. Dans (SAT'14), nous avons étudié les structures de communautés présentes dans les instances industrielles et avons montré qu'elles ont un lien avec la mesure LBD (*Literal Block Distance*) utilisée dans glucose. Dans (PAKDD'17), nous avons proposé une nouvelle approche pour la détection de communautés dans des réseaux de grande taille. Nous avons introduit un nouveau concept de communauté paramétrique, appelée *k-linked community*, permettant de mieux caractériser des communautés avec un diamètre borné. La détection est ensuite formulée comme un problème Partial-Max-SAT. Notre approche s'est montrée plus efficace que de nombreuses méthodes parmi les plus populaires de l'état de l'art.

Outils pour la modélisation

La modélisation est une étape cruciale en programmation par contraintes (PPC). Aujourd'hui, il nous semble nécessaire de simplifier cette étape afin de rendre la PPC accessible à un public plus large. Le développement de la suite MCSP3/XCSP3 soutient cet objectif, notamment car elle repose sur des standards de l'informatique (Java, XML, JSON). Proposer de nouvelles briques de modélisation, simples et génériques, est un autre apport : les contraintes « *smart* » permettent la représentation de contraintes délicates à traiter autrement (notamment, la prise en compte de la disjonction).

XCSP3 est un format de représentation, et plus généralement un projet de longue haleine autour de ce format, mené de front au CRIL depuis 2014. La production de ce document, spécifications 3.0.5 du format XCSP3, a nécessité une réflexion intense sur une période d'environ 2 ans. Avec un groupe de travail composé de 4 personnes, nous avons cherché à évaluer méthodiquement l'importance des différents concepts (notamment, les contraintes) introduits dans la littérature et les systèmes. Nous avons également sollicité de nombreux collègues d'autres laboratoires afin d'approfondir au maximum notre regard sur la question. Notre objectif est que le format XCSP3 décrit dans ce document devienne le format pivot pour les outils de programmation par contraintes (un peu comme le bytecode pour Java, toutes proportions gardées).

Hormis les spécifications, XCSP3 offre aussi :

- un site vitrine www.xcsp.org, où il est possible de sélectionner et de télécharger de nombreuses instances de problèmes au format XCSP3,
- un jeu d'instances au format XCSP3. À ce jour, plus de 23 000 instances issues d'une centaine de problèmes sont disponibles. Nombre de ces instances ont été générées à l'aide du solveur AbsCon (et maintenant à l'aide du compilateur MCSP3 décrit ci-dessous),
- des outils, dont un parseur C++ et un parseur Java, ainsi qu'un outil de vérification de solutions. Le code est disponible sur github.

Egalement, nous avons développé *MCSP3*, une API de modélisation basée sur Java 8, permettant de modéliser des problèmes combinatoires sous contraintes. L'utilisateur définit simplement et naturellement un modèle, et traduit ensuite ce modèle en instance(s) au format XCSP3, en utilisant le compilateur fourni par l'API (et en indiquant les données). L'API MCSP3, accompagnée de sa documentation, est disponible sur github.

Au delà des tables ordinaires, nous nous sommes intéressés aux tables concises (*short tables*) intégrant des valeurs universelles (*), et aussi aux tables « *smart* ». Une contrainte « *smart* » permet de gérer un ensemble (disjonction) d'entrées construites à partir d'opérations arithmétiques élémentaires. Nous avons montré que de nombreuses contraintes (dites globales) peuvent se coder de manière très compacte sous cette forme. Nous avons proposé une adaptation de l'algorithme CT (*Compact Table*) aux tables concises et/ou tables négatives (listant les combinaisons interdites) (Verhaeghe, Lecoutre et Schauss, 2017), ainsi qu'aux tables « *smart* » élémentaires (CP'17). Nous avons également montré comment une table « smart » pouvait être générée automatiquement à partir d'une table ordinaire (Le Charlier et al., 2017).

Explications de conflits

La détection de noyaux incohérents (ou de sous-ensembles maximaux cohérents) est importante à double titre : elle permet d'apporter une explication compacte à une situation conflictuelle, et elle peut être exploitée dans le cadre d'une recherche SAT incrémentale.

- **SAT incrémental** Les avancées spectaculaires obtenues dans le cadre de la résolution pratique du problème SAT ont rejailli bien au-delà de ses frontières. Ainsi, à l'heure actuelle, de nombreux problèmes dont la classe de complexité est au delà de NP peuvent être traités de manière pratique via l'utilisation de solveurs SAT. Dans un grand nombre de cas, la résolution de ces problèmes consiste à appeler un solveur SAT sur plusieurs instances analogues. Ce type de résolution, appelé résolution incrémentale de SAT, est en passe de devenir l'état de l'art dans bien des domaines. Dans (SAT'13), nous avons montré comment améliorer le solveur SAT glucose pour le rendre efficace dans le cadre incrémental. Ainsi, nous améliorons sensiblement les performances du solveur Muser qui recherche des MUS (*Minimum Unsatisfiable Subset*) dans une formule booléenne. glucose est aujourd'hui utilisé dans de nombreux solveurs nécessitant la technologie SAT incrémental (en particulier, des solveurs Max-SAT).
- **Sous-ensembles cohérents ou incohérents** Dans (Constraints'15), nous avons amélioré les techniques expérimentalement les plus efficaces pour extraire un sous-ensemble minimal incohérent (alias MUC pour *Minimal Unsatisfiable Core*) de contraintes d'un réseau sur-contraint. En particulier, nous avons montré comment on peut avantageusement éviter la première étape de ces techniques qui vise à déterminer une première sur-approximation fine d'un MUC à l'aide d'une recherche brute. Pour finir, dans (Grégoire, Lagniez et Mazure, 2014), nous avons proposé un algorithme original de calcul d'un ensemble maximal (pour l'inclusion) cohérent d'une instance SAT qui surclasse expérimentalement l'ensemble de ses concurrents.
- **Ensembles mutuellement contradictoires** Dans (Besnard, Grégoire et Lagniez, 2015) et (Grégoire, Izza et Lagniez, 2016), nous avons montré la grande diversité de problèmes d'IA qui reposent sur une capacité à raisonner déductivement sans conflit logique avec des ensembles mutuellement contradictoires d'informations. Une technique calculatoire originale de réécriture de problèmes a permis de proposer une technique souvent efficace de calcul de sous-ensembles maximaux d'informations propositionnelles qui ne contredisent aucun des contextes hypothétiques mutuellement contradictoires envisagés.

Fouille de données, apprentissage et contraintes

Etablir des passerelles entre les domaines de l'apprentissage et la fouille de données d'un côté, et de la résolution par contraintes de l'autre, nous intéresse depuis plusieurs années. Les deux domaines peuvent se fertiliser l'un l'autre, via le transfert ou l'exploitation d'approches issues de l'un des domaines à l'autre.

- **Approches déclaratives pour la fouille de données** Les approches déclaratives pour la fouille visent en particulier deux principaux objectifs. Premièrement, de telles approches permettent à l'utilisateur de caractériser avec facilité les résultats qu'il souhaite extraire. Deuxièmement, elles dissocient la caractérisation des résultats à trouver de la façon de les trouver. Cette dissociation permet de développer des approches génériques qui vont exploiter les progrès faits en résolution SAT et/ou CSP. Cette dissociation exige bien sûr un encodage (SAT et/ou CSP) pour pouvoir utiliser les solveurs existants. Dans (Jabbour, Saïs et Salhi, 2017), (ECML/PKDD'13) et (CIKM'13), nous avons introduit un nouveau cadre générique fondé sur le problème SAT permettant de résoudre de nombreux problèmes en fouille de données. Ce cadre permet en particulier l'extraction de connaissances avec une sélection faite à l'aide de relations de préférences. Cela permet, entre autres, d'avoir plus de contrôle et de précision sur la nature des connaissances obtenues de manière automatique. Dans ce cadre, nous avons formulé les problèmes d'énumération des top-k motifs intéressants à partir

de données transactionnelles, des séquences, et plus généralement de séquences d'itemsets. Ainsi, notre cadre peut en particulier être utilisé pour l'extraction de connaissances comme les anomalies et les exceptions. Par ailleurs, dans (PAKDD'15), nous avons proposé une approche permettant d'améliorer l'efficacité de méthodes déclaratives en fouille de données. Dans cette approche, on réalise la décomposition d'une tâche déclarative en fouille de données en sous-tâches peu nombreuses et plus simples. Notre approche de décomposition ouvre différentes perspectives quant à la parallélisation des méthodes déclaratives en fouille de données. Pour finir, dans (Boudane et al., 2016) et (PAKDD'17), nous avons proposé des méthodes déclaratives pour des tâches complexes en fouille de données, comme l'énumération des règles d'associations fermées, indirectes, minimales et non redondantes. L'évaluation expérimentale a montré que notre approche déclarative permet d'améliorer sensiblement les méthodes dédiées de l'état de l'art.

- **Clustering symbolique** Le clustering est l'un des problèmes majeurs en fouille de données. Ce processus d'apprentissage non supervisé a fait l'objet de nombreuses études conduisant à plusieurs algorithmes traitant divers types de données. Habituellement, les objets sont représentés comme des vecteurs n-dimensionnels d'attributs numériques. Cependant, dans de nombreuses applications, les données sont de nature encore plus complexe, décrivant, par exemple, les désirs ou les préférences des clients. De telles données peuvent être exprimées de manière plus compacte en utilisant des représentations logiques. Dans (PAKDD'17), nous avons introduit un nouveau cadre pour le clustering, où les objets complexes sont décrits par des formules booléennes. Nous avons étendu diverses techniques de clustering comme le k-means, pour traiter des objets complexes représentés sous forme de formules booléennes, et un nouvel algorithme de clustering permettant de regrouper des objets représentés explicitement par des ensembles de modèles. Enfin, nous avons proposé un codage basé sur SAT sans avoir besoin de passer par une représentation explicite des modèles.
- **Compression de formules, de contraintes et de graphes** Dans (CIKM'13), nous avons proposé une approche basée sur la fouille de données pour découvrir des motifs ou structures cachés dans les formules booléennes. Nous avons montré qu'il est possible d'exploiter ces motifs pour la compression de ces formules.
- **Apprentissage et contraintes** Ce thème s'articule autour de deux axes : (i) utiliser la programmation par contraintes pour résoudre des problèmes d'apprentissage, et (ii) utiliser l'apprentissage pour la modélisation par contraintes. Concernant le premier axe, de nombreuses tâches d'apprentissage statistique ou en-ligne peuvent se formuler comme des problèmes d'optimisation stochastique ou séquentielle sous contraintes. Dès lors que la tâche d'apprentissage fait intervenir des contraintes symboliques (par exemple, la structuration de l'espace d'hypothèses), le problème correspondant est souvent NP-difficile. Pour l'apprentissage de fonctions linéaires discrètes, où les hypothèses sont contraintes dans un hypercube, nous avons proposé un algorithme stochastique d'approximation (avec borne de convergence) dans (ICML'13). Pour l'apprentissage de modèles graphiques probabilistes, où les hypothèses sont structurées en graphes (orientés ou non orientés), nous avons formulé le problème sous contrainte de matroïde, et résolu le problème d'apprentissage séquentiel de forêts markoviennes (Koriche, 2016). Concernant le deuxième axe, la modélisation d'un problème de décision sous forme de contraintes nécessite un véritable effort cognitif de la part de l'utilisateur, non expert du domaine. Dans ce contexte, l'apprentissage automatique peut aider l'utilisateur à formuler son problème combinatoire, en exploitant une bibliothèque de contraintes (par exemple dans XCSP3), et en faisant appel aux instances (par exemple des emplois du temps) que l'utilisateur a déjà résolu « à la main ». Dans (Bessiere et al., 2017), nous avons montré que l'apprentissage avec requêtes (modèle d'Angluin) permet de converger sur le choix d'un réseau de contraintes, en utilisant très peu de requêtes d'appartenance et d'équivalence.

Quelques cas d'utilisation

- **Planification de visites de musée** Dans (ICAPS'13), nous avons formalisé et étudié la complexité du problème de personnalisation de visites de musée et développé un solveur dédié pour la résolution de ce problème (PMV). Nous avons aussi développé un prototype à destination du grand public sous forme d'une application web, baptisé Tech-A-Way.
- **Résolution efficace des problèmes en logique modale** Les logiques modales sont utilisées dans la modélisation de plusieurs problèmes comme la planification contingente et la compilation de connaissances. Pourtant, la résolution pratique de ces problèmes est encore un défi, parce que le problème de la cohérence dans ces logiques est généralement NP-complet ou PSPACE-complet. Dans (Caridroit et al., 2017), nous avons proposé une nouvelle traduction de la logique modale S5 vers la logique propositionnelle. Nous avons montré que la réduction du nombre de variables et des clauses dans le résultat de la traduction permet d'augmenter considérablement l'efficacité de l'algorithme pour le test de cohérence en logique modale S5. Cet travail a été étendu pour obtenir des modèles de Kripke minimaux dans (Lagniez et al., 2018). La suite de ce travail a été présenté dans (Lagniez et al., 2017) : nous avons proposé dans cet article un algorithme générique pour la résolution du problème de cohérence de problèmes PSPACE-complets, appelé RECAR. Cette approche a été

implémentée dans le solveur Mosaic pour la résolution du problème de cohérence en logique modale K. Les résultats expérimentaux montrent que Mosaic est actuellement le solveur le plus efficace de la littérature pour résoudre les instances de ce problème.

- **Contraintes et génie logiciel** La période de référence a été l'occasion de renforcer le travail à l'intersection de l'IA et du génie logiciel, notamment par la collaboration avec l'équipe de recherche Spirals du laboratoire CRIStAL. Au niveau des lignes de produits logiciels (considérées par une communauté de recherche qui utilise Sat4j depuis 2005), le formalisme logique utilisé a été étendu à l'aide de contraintes de cardinalité dans (SPLC'14). Dans le cadre de la réparation automatique de logiciels, une approche à base de solveur SMT a été proposée dans (Xuan et al., 2017).

Faits marquants

- 2018 : organisation de la conférence internationale CP'18 par le CRIL à Lille (Euratechnologies).
- 2018 : parution du *Handbook of Parallel Constraint Reasoning*, co-édité par Lakhdar Saïs.
- 2017 : le projet Mastodons est présenté au Salon Innovatives SHS du CNRS, à Marseille, les 16 et 17 mai
- 2016 : spécifications 3.0.0 du format XCSP3, accompagné de son écosystème.
- 2016 : Daniel Le Berre est co-président du comité de programme de la conférence internationale SAT 2016.
- 2013-2017 : glucose, solveur SAT, primé à toutes les compétitions SAT depuis 2013.
- 2013 : Cédric Piette devient président de l'Association Française pour la Programmation par Contraintes (AFPC).

Sélection des publications

Audemard Gilles, Lagniez Jean-Marie, Simon Laurent, 2013, « Just-In-Time Compilation of Knowledge Bases », *23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*, p. 447-453.

Besnard Philippe, Grégoire Éric, Lagniez Jean-Marie, 2015, « On Computing Maximal Subsets of Clauses that Must Be Satisfiable with Possibly Mutually-Contradictory Assumptive Contexts », *29th AAAI Conference on Artificial Intelligence (AAAI'15)*, p. 3710-3716.

Bessiere Christian, Fargier Hélène, Lecoutre Christophe, 2016, « Computing and restoring global inverse consistency in interactive constraint satisfaction », *Artificial Intelligence (AIJ)*, 241, p. 153-169.

Bessiere Christian, Koriche Frédéric, Lazaar Nadjib, O'Sullivan Barry, 2017, « Constraint Acquisition », *Artificial Intelligence (AIJ)*, 244, p. 315-342.

Boudane Abdelhamid, Jabbour Saïd, Saïs Lakhdar, Salhi Yakoub, 2016, « A SAT-based Approach for Mining Association Rules », *25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, p. 2472-2478.

Bova Simone, Capelli Florent, Mengel Stefan, Slivovsky Friedrich, 2016, « Knowledge Compilation Meets Communication Complexity » Kambhampati Subbarao (dir.), *25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, p. 1008-1014.

Caridroit Thomas, Lagniez Jean-Marie, Le Berre Daniel, De Lima Tiago, Montmirail Valentin, 2017, « A SAT-Based Approach for Solving the Modal Logic S5-Satisfiability Problem », *31st AAAI Conference on Artificial Intelligence (AAAI'17)*, p. 3864-3870.

Chen Hubie, Mengel Stefan, 2016, « Counting Answers to Existential Positive Queries: A Complexity Classification » Milo Tova, Tan Wang-Chiew (dir.), *35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS'16)*, p. 315-326.

Chen Hubie, Mengel Stefan, 2017, « The logic of counting query answers », *32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'17)*, p. 1-12.

Fargier Hélène, Marquis Pierre, 2014, « Disjunctive closures for knowledge compilation », *Artificial Intelligence (AIJ)*, 216, p. 129-162.

Fargier Hélène, Marquis Pierre, Niveau Alexandre, Schmidt Nicolas, 2014, « A Knowledge Compilation Map for Ordered Real-Valued Decision Diagrams », *28th AAAI Conference on Artificial Intelligence (AAAI'14)*, p. 1049-1055.

Grégoire Éric, Izza Yacine, Lagniez Jean-Marie, 2016, « On the Extraction of One Maximal Information Subset That Does Not Conflict with Multiple Contexts », *30th AAAI Conference on Artificial Intelligence (AAAI'16)*, p. 3404-3410.

Grégoire Éric, Lagniez Jean-Marie, Mazure Bertrand, 2013, « Preserving Partial Solutions while Relaxing Constraint Networks », *23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*, p. 552-558.

Grégoire Éric, Lagniez Jean-Marie, Mazure Bertrand, 2014, « An Experimentally Efficient Method for (MSS, CoMSS) Partitioning », *28th Conference on Artificial Intelligence (AAAI'14)*, p. 2666-2673.

Hamadi Youssef, Saïs Lakhdar, 2018, « Handbook of Parallel Constraint Reasoning », *in* Springer International Publishing.

Jabbour Saïd, Saïs Lakhdar, Salhi Yakoub, 2017, « Mining Top-k motifs with a SAT-based framework », *Artificial Intelligence (AIJ)*, 244, p. 30-47.

Koriche Frédéric, 2016, « Online Forest Density Estimation », *32nd International Conference on Uncertainty in Artificial Intelligence (UAI'16)*, p. 357-366.

Koriche Frédéric, 2018, « Compiling Combinatorial Prediction Games » Dy Jennifer, Krause Andras (dir.), *35th International Conference on Machine Learning (ICML'18)*.

Koriche Frédéric, Lagniez Jean-Marie, Marquis Pierre, 2015, « Compiling Constraint Networks into Multivalued Decomposable Decision Graphs », *24th International Joint Conference on Artificial Intelligence (IJCAI'15)*, p. 332-338.

Lagniez Jean-Marie, Le Berre Daniel, De Lima Tiago, Montmirail Valentin, 2017, « A Recursive Shortcut for CEGAR: Application To The Modal Logic K Satisfiability Problem », *26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, p. 674-680.

Lagniez Jean-Marie, Le Berre Daniel, De Lima Tiago, Montmirail Valentin, 2018, « An Assumption-Based Approach for Solving The Minimal S5-Satisfiability Problem », *9th International Joint Conference on Automated Reasoning (IJCAR'18)*.

Lagniez Jean-Marie, Lonca Emmanuel, Marquis Pierre, 2016, « Improving Model Counting by Leveraging Definability », *25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, p. 751-757.

Lagniez Jean-Marie, Marquis Pierre, 2017a, « An Improved Decision-DNNF Compiler », *26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, p. 667-673.

Lagniez Jean-Marie, Marquis Pierre, 2017b, « On Preprocessing Techniques and Their Impact on Propositional Model Counting », *Journal of Automated Reasoning (JAR)*, 58(4), p. 413-481.

Lagniez Jean-Marie, Marquis Pierre, Szczepanski Nicolas, 2018, « DMC: A Distributed Model Counter », *27th International Joint Conference on Artificial Intelligence (IJCAI'18)*.

Le Berre Daniel, Marquis Pierre, Mengel Stefan, Wallon Romain, 2018, « Pseudo-Boolean Constraints from a Knowledge Representation Perspective », *27th International Joint Conference on Artificial Intelligence (IJCAI'18)*.

Le Charlier Baudouin, Khong Minh Thanh, Lecoutre Christophe, Deville Yves, 2017, « Automatic Synthesis of Smart Table Constraints by Abstraction of Table Constraints », *26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, p. 681-687.

Lecoutre Christophe, Likitvivatanavong Chavalit, Yap Roland, 2015, « STR3: A path-optimal filtering algorithm for table constraints », *Artificial Intelligence (AIJ)*, 220, p. 1-27.

Marquis Pierre, 2015, « Compile! », *29th AAAI Conference on Artificial Intelligence (AAAI'15)*, p. 4112-4118.

Paparrizou Anastasia, Stergiou Kostas, 2017, « On Neighborhood Singleton Consistencies », *26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, p. 736-742.

Verhaeghe Hélène, Lecoutre Christophe, Schauss Pierre, 2017, « Extending Compact-Table to Negative and Short Tables », *31st AAAI Conference on Artificial Intelligence (AAAI'17)*, p. 3951-3957.

Xuan Jifeng, Martinez Matias, Demarco Favio, Clement Maxime, Lamelas Sebastian, Durieux Thomas, Le Berre Daniel, Monperrus Martin, 2017, « Nopol: Automatic Repair of Conditional Statement Bugs in Java Programs », *IEEE Transactions on Software Engineering (TSE)*, 43(1), p. 34-55.