Extension de Compact-Diagram aux smart MVD

Hélène Verhaeghe[1*] **Christophe Lecoutre**[2] **Pierre Schaus**[1]

[1]UCLouvain, ICTEAM, Place Sainte Barbe 2, 1348 Louvain-la-Neuve, Belgique
[2]CRIL-CNRS UMR 8188, Université d'Artois, F-62307 Lens, France
[1]{*prenom.nom*}@*uclouvain.be* [2]*lecoutre@cril.fr*

Résumé

Les diagrammes de décision multi-valués (MDD), et plus généralement les diagrammes variables multi-valués (MVD), sont des structures qui peuvent s'avérer utiles pour la modélisation de problèmes combinatoires sous contraintes. Ces dernières années, de nombreux algorithmes de filtrage ont été proposés tels que mddc, MDD4R et Compact-Diagram. Par ailleurs, diverses formes compressées de tables ont également été proposées au fil du temps menant notamment à une hybridation 'smart' entre les représentations des contraintes dite en extension et en intension. Cela peut se résumer à intégrer des contraintes arithmétiques simples dans les tables définissant la sémantique des contraintes. L'algorithme état de l'art Compact-Table a été récemment étendu pour gérer ces tables 'smart simple', i.e. des tables contenant des éléments de la forme '$*$', '$\neq v$', '$\leq v$', '$\geq v$' et '$\in S$'. Dans notre article, intitulé "Extending Compact-Diagram to Basic Smart Multi-Valued Variable Diagrams"[2] et publié à CPAIOR-19, nous introduisons le concept de MVD 'smart simple' (*bs*-MVD) correspondant à la possibilité d'utiliser ces formes de contraintes arithmétiques au niveau des étiquettes des diagrammes. Nous montrons comment étendre l'algorithme Compact-Diagram pour gérer ce type de diagrammes.

1 Introduction

La représentation efficace de contraintes sous forme de tables ou diagrammes de décision est l'un des sujets de recherche importants de la dernière décennie dans le contexte de la programmation par contraintes. Ces améliorations peuvent être classées en deux grandes catégories : i) l'amélioration des algorithmes de filtrage (introduction de la technique de réduction tabulaire, exploitation des opérations bit à bit, ...) et ii) l'amélioration de la représentation (tables compressées, formes compactes de graphes, ...).

Compact-Diagram (appelé précédemment Compact-MDD) est un algorithme de filtrage introduit récemment [1]. Il exploite notamment les opérations bit à bit adaptées à toute contrainte définie par un diagramme variable multi-valué (MVD), qui est une généralisation du diagramme de décision multi-valué autorisant le non-déterminisme. Dans cet article, nous proposons une extension de l'algorithme Compact-Diagram (CD) permettant de gérer des diagrammes 'smart' (simples), i.e. des diagrammes tels que les arêtes peuvent avoir des étiquettes de la forme '$= v$', '$*$', '$\neq v$', '$\leq v$', '$\geq v$' et '$\in S$'. Nous décrivons également comment il est possible de générer ces diagrammes à partir de tables.

Au delà de la compression potentielle obtenue avec ces nouveaux diagrammes, une application directe est la représentation de la contrainte regular de manière compacte.

2 Transformation de Tables en *bs*-MVD

La transformation peut être effectuée de deux manières. La première possibilité est de transformer une table en diagramme en utilisant l'un des algorithmes déjà connu tel que pReduce (pour générer un MDD) ou sReduce (pour générer un semi-MDD). Ensuite, le diagramme est réduit en regroupant certaines arêtes. Cette opération consiste à examiner tous les groupes d'arêtes partageant le même nœud source et le même nœud destination. Les étiquettes présentes dans le groupe sont alors examinées et comparées au domaine de la variable associée. Des expressions unaires peuvent alors être dérivées et les arêtes concernées sont remplacées par de nouvelles arêtes qui sont étiquetées avec ces expressions. Par exemple, si toutes les valeurs du domaine sont représentées dans le groupe, cela correspond à '$*$', si toutes les valeurs inférieures à v sont représentées, cela correspond à '$\leq v$',....

*Papier doctorant : Hélène Verhaeghe[1] est auteur principal.

La deuxième possibilité est de commencer par transformer la table en une table smart (simple). Ensuite, celle-ci est transformée en diagramme par l'adaptation des algorithmes connus pour gérer les nouvelles catégories d'étiquettes. Pour pReduce et sReduce, cette adaptation ne requiert que la définition d'un ordre total sur toutes les étiquettes possibles.

3 CD^{bs} : CD appliqué aux *bs*-MVD

CD et CT sont relativement similaires en terme de conception. Les deux algorithmes utilisent des vecteurs de bits pré-calculés appelés `supports` pour identifier les arêtes et tuples qui doivent être éliminés durant la phase de mise à jour (suite à la suppression de valeurs dans les domaines des variables associées). Pour gérer les contraintes unaires, CD peut être modifié selon la logique suivie par CT. Tout comme pour CT^{bs}, trois vecteurs de bits pré-calculés sont introduits : `supports`*, `supportsMin` et `supportsMax`. Des modifications s'opèrent également comme pour CT pendant la phase de mise à jour.

Contrairement aux tables pour lesquelles un bit est associé à un tuple et donc à plusieurs variables simultanément, ici, chaque bit est associé à une arête et donc à une seule variable. L'ordre des bits d'un niveau peut donc être facilement changé sans impacter les autres niveaux du diagramme. Une partition des arêtes est donc possible suivant leur type d'étiquette. Les ensembles de bits, implantés par des tableaux de "mots" (i.e. groupements de 64 bits quand on utilise en Java un entier de type Long), peuvent donc être construits selon cette partition (tout les bits d'un même mot sont associés à des arêtes d'une même catégorie). Cela permet à CD^{bs} d'appliquer un nombre restreint d'opérations pour chacun des mots, suivant la catégorie de celui-ci contrairement à CT^{bs} qui ne pouvait pas faire d'hypothèses sur le contenu des mots.

4 Résultats

Les résultats portant sur la transformation de tables en *bs*-MVD nous montre tout d'abord que la méthode transitant par les diagrammes génère moins de nœuds et d'arcs que la méthode transitant par les *bs*-tables. Nous expliquons cela par une probabilité plus faible d'obtenir un regroupement de nœuds lors du passage par la *bs*-table dû à l'augmentation du nombre d'étiquettes possibles amenée par l'introduction des étiquettes 's-mart' présentes dans la *bs*-table.

Les résultats concernant l'algorithme de filtrage montrent une amélioration en temps lorsque le diagramme compressé est plus petit en taille que le diagramme non compressé correspondant. Cela est le cas pour les instances générées par transformation de tables en *bs*-MDD en transitant préalablement par la transformation en diagrammes.

En comparaison avec CT^{bs}, CD^{bs} reste toutefois moins performant. Néanmoins l'écart est de plus en plus réduit entre les deux approches (tables versus diagrammes) et un plus grand nombre d'instances est résolu (de manière plus rapide) par CD^{bs} que CD. Ces instances sont sans grande surprise celles où la compression en diagrammes est la plus importante. Il est également connu que certains types de problèmes (tels que ceux impliquant la contrainte globale AllDifferent) ne sont pas du tout appropriés à une représentation sous forme de diagrammes dû une possibilité de compression très limitée (lors du processus de regroupement de nœuds). Il est de ce fait, et selon notre opinion, utile d'avoir les deux formes de contraintes, et leurs propagateurs CD et CT, pour pouvoir sélectionner le plus pertinent en fonction de la capacité de compression en diagrammes.

5 Conclusion

Dans cet article, nous avons proposé une extension des diagrammes classiques. Nous avons montré comment générer ces *bs*-MVD à partir de tables, de tables 'smart simple' et de diagrammes. Nous avons présenté un algorithme de filtrage pour cette nouvelle forme de diagramme. Cet algorithme est efficace et permet de réduire encore un peu plus l'écart entre les méthodes basées sur les tables et les méthodes basées sur les diagrammes. Pour plus de détails concernant les structures, algorithmes et résultats, veuillez consulter l'article original [2].

Remerciements

Le second auteur est financé par le projet CPER Data des Hauts-de-France.

Références

[1] Hélène Verhaeghe, Christophe Lecoutre et Pierre Schaus : Compact-MDD : Efficiently filtering (s)MDD constraints with reversible sparse bit-sets. *In IJCAI18*, pages 1383–1389, 2018.

[2] Hélène Verhaeghe, Christophe Lecoutre et Pierre Schaus : Extending compact-diagram to basic smart multi-valued variable diagrams. *In Proceedings of CPAIOR'17*, 2019.