Méta-contraintes pour la modélisation des problèmes sur-contraints

Thierry Petit *
ILOG Sophia Antipolis - LIRMM †,
les Taissounières, 1681 route des Dolines, 06560 Valbonne
email: `tpetit@ilog.fr, tpetit@lirmm.fr`

Lorsqu'on modélise une application réelle sous la forme d'un réseau de contraintes on obtient souvent un problème sur-contraint, qui n'a pas de solution si on impose que toutes les contraintes posées soient satisfaites. Dans ce cas il est necessaire de tolérer qu'un certain nombre de contraintes soient violées dans les solutions. Le but est alors de trouver le meilleur compromis possible. On qualifie généralement de *dures* les contraintes qui doivent nécessairement être satisfaites, comme par exemple des contraintes physiques, et de *molles* celles qui peuvent éventuellement être violées.

Une attention particulière doit être portée sur l'efficacité de la recherche et sur l'expression des critères que doivent vérifier les solutions pour être réalistes. Bien qu'il soit difficile de séparer totalement ces deux aspects, nous nous concentrons essentiellement, dans ce résumé, sur le deuxième. Afin de résoudre des problèmes réels, un modèle théorique doit fournir la possibilité d'exprimer des règles sur les contraintes qui doivent être violées ou satisfaites dans les solutions, et des règles sur les degrés de violation, sachant que certaines contraintes peuvent être plus ou moins fortement violées.

Le modèle le plus simple est Max-CSP [FW92], où l'objectif consiste à minimiser le nombre de contraintes violées. Afin de caractériser d'une manière plus fine les solutions acceptables, d'autres modèles ont été proposés [Sch92, DFP93, SFV95]. Ils sont inclus dans deux paradigmes génériques, appellés "Valued CSP" et "Semiring CSP" [BMR+99]. Leur point commun est le suivant : une valuation est associée à chaque contrainte ou bien à chaque combinaison de valeurs possibles pour les variables impliquées dans chaque contrainte; l'ensemble des valuations est alors intégré dans une fonction d'optimisation. Selon les approches, cette fonction est basée sur des opérateurs monotones ou idempotents. Ce mécanisme permet de spécifier quelles sont les contraintes dures, d'exprimer des règles de priorité sur les contraintes molles – i.e, de favoriser la satisfaction de celles qui ont la plus grande importance – et/ou de distinguer plusiseurs degrés de violation des contraintes.

* Ce travail a été réalisé dans le cadre d'une thèse CIFRE co-encadrée par Jean-Charles Régin (ILOG) et Christian Bessière (LIRMM-CNRS).

† Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier, 161 rue Ada, 34392 Montpellier Cedex 5.

Cependant, il est souvent nécessaire d'exprimer des règles plus complexes, définies sur des sous-ensembles de contraintes indépendamment de la fonction d'optimisation. Ces règles ne peuvent pas toujours être modélisées à l'aide d'une fonction définie sur des valuations. Deux types de règles apparaîssent fréquemment dans les problèmes réels :

1. *répartition homogène des violations*
   On peut considérer l'exemple d'un problème de gestion d'emplois du temps : chaque personne exprime des préférences sur les horaires ou sur les tâches à effectuer. Une solution violant toutes les préférences de certaines personnes et satisfaisant toutes les préférences d'autres personnes ne serait pas équitable.
2. *dépendances entre les contraintes*
   Il est parfois nécessaire de définir des règles de dépendance : si certaines contraintes sont violées alors d'autres doivent impérativement être satisfaites – elles deviennent dures –, ou bien de nouvelles contraintes qui n'appartenaient pas au réseau initial doivent être déclenchées, etc.

Afin d'exprimer de telles règles, nous proposons de modifier le réseau de contraintes. Un ensemble de variables est ajouté afin de caractériser l'état de chaque contrainte molle : *violée* ou *satisfaite*. Le domaine d'une variable d'état s est consitué d'un ensemble de valuations totalement ordonné, possédant un élément minimal 0 qui correspond à l'état de satisfaction de la contrainte C à laquelle elle est attachée. C est alors remplacée par la contrainte C' : $[C \wedge (s = 0)] \vee [\neg C \wedge (s > 0)]$. On traite donc un problème d'optimisation dont l'objectif est relatif à l'ensemble des valeurs affectées aux variables d'état.

Un des avantages de ce modèle est que l'on peut poser de nouvelles contraintes sur les variables d'état, que l'on qualifie de *méta-contraintes*, car leur rôle est de modéliser les interactions entre les contraintes molles du problème. Par exemple, l'homogénéité (1) est exprimée par des contraintes de "global cardinality" [Rég96], posées sur des sous-ensembles de variables d'état. Les dépendances (2) sont modélisées par des contraintes d'implication. De nombreux autres types de règles peuvent être exprimés à l'aide de méta-contraintes.

Dans la perspective de traiter des problèmes réels, nous étudions l'adaptation d'algorithmes de filtrage à notre modèle, capables de prendre en compte des contraintes de type et d'arité quelconques.

Références

[BMR+99] S. Bistarelli, U. Montanari, F. Rossi, T. Schiex, G. Verfaillie, and H. Fargier. Semiring-based csps and valued csps: Frameworks, properties, and comparison. *Constraints*, 4:199–240, 1999.

[DFP93] D. Dubois, H. Fargier, and H. Prade. The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction. *FUZZ-IEEE'93*, 1993.

[FW92] E.C. Freuder and R.J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 58:21–70, 1992.

[Rég96] J.C. Régin. Generalized arc consistency for global cardinality constraint. *Proceedings AAAI*, pages 209–215, 1996.

[Sch92] T. Schiex. Possibilistic constraint satisfaction problems, or "how to handle soft constraints?". *Proceedings of 8th Conf. of Uncertainty in AI*, 1992.

[SFV95] T. Schiex, H. Fargier, and G. Verfaillie. Valued constraint satisfaction problems: hard and easy problems. *Proceedings IJCAI*, 1995.