Contraintes sur des domaines paramétriques

Université de Nantes, France

Charlotte Truchet
Université de Nantes
Charlotte.Truchet@univ-nantes.fr

David Cachera
ENS Rennes
David.Cachera@irisa.fr

Contexte

La Programmation par Contraintes (CP) étudie des problèmes formalisés comme une conjonction de prédicats du premier ordre, appelés contraintes, qui définissent des relations entre les variables du problème [5]. La CP offre des méthodes de résolution génériques efficaces avec des applications variées en logistique, conception, bioinformatique, musique... Ces méthodes de résolution intègrent notamment des raisonnements à partir des contraintes, qui permettent de réduire l'espace de recherche. Si ces raisonnements ne sont pas suffisants pour trouver une solution, les solveurs de contraintes parcourent alors l'espace de recherche avec une méthode brute force.

L'une des limitations des méthodes de contraintes (comme d'autres méthodes en optimisation combinatoire) est que le format des problèmes est assez restreint. Le problème doit être écrit avec un nombre fixé de variables, chacune à valeur dans un domaine également fixé : une variable peut prendre des valeurs dans le domaine [1..10], mais pas dans un domaine paramétré comme par exemple $[1..n]$. Cette restriction est nécessaire pour appeler les algorithmes de résolution, qui souvent travaillent sur des structures de données construites à partir des domaines (par exemple, graphe biparti entre les domaines de deux variables, dans le cas d'une des contraintes les plus utilisées, alldifferent).

Cela limite l'application des contraintes à des problèmes complètement spécifiés. Or, dans certains cas, on souhaiterait étendre les raisonnements implémentés à des domaines paramétrés. C'est par exemple le cas lorsque l'on utilise les contraintes pour des problèmes issus de la vérification de programmes, par exemple pour le calcul de Worst Case Execution Time (WCET) [1] : il faut pouvoir tenir compte des paramètres d'entrée du programme. Une solution de facilité consiste à donner une borne, fixée, à ces paramètres, mais la solution obtenue dans ce cas risque d'être très imprécise. Au delÃ de cet exemple particulier, il n'existe pas de solution générique aux problèmes de contraintes avec domaines paramétrés.

Sujet

Le sujet de ce stage est l'extension des méthodes de programmation par contraintes à des domaines paramétrés. Dans le cas général, c'est un sujet difficile car tous les raisonnements effectués sur des contraintes doivent être redéfinis. On étudiera donc dans un premier temps le cas de la consistance de bornes (les raisonnements sont effectués uniquement aux bornes des domaines) sur des domaines entiers, et d'abord sur un langage de contraintes simplifié. Le/la

stagiaire pourra dans un premier temps étudier le cadre CLP (en particulier CLP(FD) [2]) qui implémente les raisonnements sur les contraintes à base de règles, puis proposer une extension de ce cadre au cas des domaines paramétrés. Il pourra ensuite définir une méthode d'instanciation générique pour ces mêmes domaines.

L'implémentation sera faite dans le solveur de contraintes Absolute [4]. Absolute est basé sur la notion de domaine abstrait en interprétation abstraite (AI), une théorie de l'approximation de sémantiques pour la preuve de programmes [3]. Les domaines utilisés par Absolute redéfinissent les opérateurs de résolution de contraintes de façon générique, le solveur étant pour l'instant plus développé sur les domaines numériques (boîtes, octogones, polyèdres) que sur les domaines entiers. Il s'agira ici d'ajouter un nouveau domaine, les domaines entiers paramétrés, au solveur.

Informations pratiques

Ce stage est basé à l'Université de Nantes, et encadré par Charlotte Truchet, au LINA (UMR 6241). Il fera l'objet d'une collaboration avec David Cachera, IRISA, Rennes.
Absolute étant développé en OCaml, le candidat ou la candidate doit avoir des bases de programmation en OCaml.

Le stage sera financé, si besoin, par le projet ANR Coverif. Il pourra éventuellement donner lieu à une continuation en thèse, avec un financement par le même projet.

References

[1] Stefan Bygde, Andreas Ermedahl, and Björn Lisper. An efficient algorithm for parametric WCET calculation. *Journal of Systems Architecture - Embedded Systems Design*, 57(6):614–624, 2011.

[2] Philippe Codognet and Daniel Diaz. Compiling constraints in clp(fd). *The Journal of Logic Programming*, 27(3):185 – 226, 1996.

[3] Patrick Cousot and Radhia Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, 1977. ACM Press, New York, NY.

[4] Marie Pelleau, Antoine Miné, Charlotte Truchet, and Frédéric Benhamou. A Constraint Solver based on Abstract Domains. In Roberto Giacobazzi, Josh Berdine, and Isabella Mastroeni, editors, *Verification, Model Checking, and Abstract Interpretation*, volume 7737 of *Lecture Notes in Computer Science*, pages 434–454, Rome, Italie, 2013. Springer-Verlag.

[5] Francesca Rossi, Peter van Beek, and Toby Walsh. *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. Elsevier, 2006.