Une nouvelle heuristique de choix de variables

Christophe Lecoutre[1] **Gilles Audemard**[1] **Charles Prud'homme**[2]
[1] Univ. Artois, CNRS, CRIL, France
[2] TASC, IMT-Atlantique, LS2N-CNRS, France
{lecoutre,audemard}@cril.fr charles.prudhomme@imt-atlantique.fr

Résumé

Il est bien connu que les heuristiques de choix de variables jouent un rôle central dans la résolution efficace des instances du problème de satisfaction de contraintes (CSP). Pendant plus de deux décennies à partir du début des années 80, l'heuristique `dom` (et ses variantes) sélectionnant la variable avec le plus petit domaine s'est imposée. Puis, à partir du milieu des années 2000, certaines heuristiques adaptatives ont été introduites : leur principe est de collecter des informations pendant le processus de recherche afin de prendre des décisions mieux informées. Parmi ces heuristiques adaptatives, `wdeg/dom` (et ses variantes) reste particulièrement robuste. Dans cet article, nous introduisons une heuristique originale basée sur le traitement *intermédiaire* des conflits (résultant de la propagation des contraintes) : cette heuristique appelée `pick/dom` piste les variables qui sont directement impliquées dans le processus de propagation des contraintes, lorsqu'il se termine par un conflit. La robustesse de cette nouvelle heuristique est démontrée à partir d'une large expérimentation menée avec le solveur de contraintes ACE. Il est intéressant de noter que l'on peut observer une certaine complémentarité entre les formes précoce, intermédiaire et tardive de traitement des conflits.

1 Introduction

La recherche arborescente reste une approche classique pour résoudre les instances du problème de satisfaction de contraintes (CSP pour Constraint Satisfaction Problem). Elle est basée sur une exploration en profondeur avec retours-arrières, qui s'appuie sur une heuristique de choix de variables. Le choix de la bonne heuristique pour résoudre un réseau de contraintes donné est une question clé, car des heuristiques différentes peuvent conduire à des arbres de recherche radicalement différents.

Dans cet article, et sa version longue [1], nous introduisons une heuristique originale appelée `pick/dom` qui apprend des conflits en identifiant les variables qui sont directement impliquées dans le processus de propagation des contraintes (en cas d'échec). La robustesse de cette nouvelle heuristique est démontrée en menant une vaste expérimentation avec les solveurs de contraintes bien connus ACE [4] et Choco [6].

2 Discussion

Les trois façons d'exploiter les conflits pour guider la recherche, se concrétisent par trois heuristiques différentes :

- `wdeg/dom` [2, 3, 7]
- `frba/dom` [5]
- `pick/dom`

qui montrent des comportements quelque peu complémentaires. Cela peut s'expliquer par le fait que l'information est extraite à différents moments : au tout début du processus conduisant à un conflit (c'est-à-dire au moment de la décision), pendant la propagation des contraintes, ou au moment où le dernier propagateur (algorithme de filtrage) est sollicité. On peut alors parler d'approches telles que le traitement opérationnel des conflits est *early* (E), *midway* (M) ou *late* (L). Ceci est illustré par la figure 1 où une nouvelle décision $x = a$ est prise, lors de la résolution d'un réseau CN P, dans la continuité de deux décisions prises précédemment $v = a$ et $w \neq b$. Dans notre scénario, l'exécution de la propagation de contraintes ϕ sur (l'état actuel de) P après avoir assigné la valeur a à x, c'est-à-dire $\phi(P|_{x=a})$, révèle une nouvelle situation conflictuelle (sans issue) (désignée par $\bot$). Le traitement précoce de ce nouveau conflit consiste à considérer la variable x impliquée dans la décision comme le principal coupable. C'est le principe de l'heuristique `frba/dom`. Le traitement intermédiaire de ce conflit consiste à considérer que toutes les variables ayant joué un rôle (c'est-à-dire ayant été

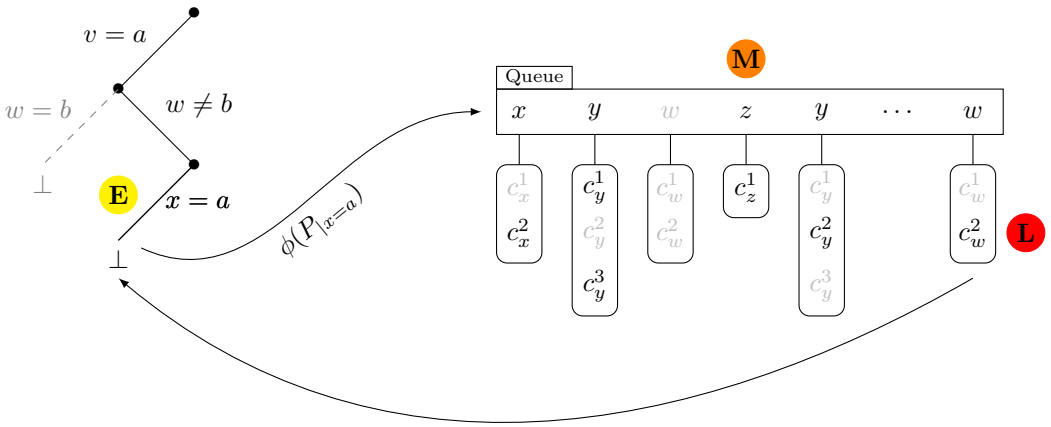

FIGURE 1 – Illustration des moments charnières pour la collecte d'informations sur les conflits : ils correspondent au traitement précoce (E), intermédiaire (M) et tardif (L) des conflits.

choisies) au cours de la propagation ont contribué à l'échec. C'est le principe qui sous-tend l'heuristique `pick/dom`. Le traitement tardif de ce conflit consiste à considérer la dernière contrainte (ici, c_w^2) provoquant un domain-wipeout (c'est-à-dire supprimant la dernière valeur d'un domaine) comme l'objet d'intérêt. C'est le principe de la pondération de contraintes, comme dans `wdeg/dom`.

3 Conclusion

Dans cet article, nous avons introduit une nouvelle façon d'exploiter les conflits pendant la recherche arborescente afin de construire une heuristique de choix de variables bien informée. Contrairement aux heuristiques existantes qui s'appuient sur le traitement précoce et tardif des conflits, cette nouvelle heuristique, `pick/dom`, consiste à traiter les conflits en suivant les variables pendant la propagation de contraintes. La robustesse de `pick/dom` a été démontrée par une campagne expérimentale portant sur plus de 120 problèmes (et environ 2000 instances). Il est intéressant de noter que les trois formes différentes d'exploitation des conflits, basées sur des moments différents de collecte d'informations, entraînent des comportements quelque peu complémentaires du solveur. Cela ouvre des perspectives pour la construction d'un solveur encore plus robuste en combinant ces heuristiques basées sur les conflits d'une manière intelligente. L'identification des caractéristiques ou des propriétés des instances (par exemple, la largeur d'arbre, la taille de l'épine dorsale, la présence de communautés, la structure des tableaux de variables, etc.) qui peuvent apparaître comme favorables à une certaine forme de traitement des conflits est une question qui mériterait également d'être étudiée.

Remerciements. Ce travail a bénéficié du soutien de l'ANR (France 2030), dans le cadre du projet MAIA (ANR-22-EXES-0009).

Références

[1] Gilles AUDEMARD, Christophe LECOUTRE et Charles PRUD'HOMME : Guiding backtrack search by tracking variables during constraint propagation. *In Proceedings of CP'23*, pages 9–17, 2023.

[2] F. BOUSSEMART, F. HEMERY, C. LECOUTRE et L. SAIS : Boosting systematic search by weighting constraints. *In Proceedings of ECAI'04*, pages 146–150, 2004.

[3] D. HABET et C. TERRIOUX : Conflict history based heuristic for constraint satisfaction problem solving. *Journal of Heuristics*, 27(6):951–990, 2021.

[4] C. LECOUTRE : ACE, a generic constraint solver. *CoRR*, abs/2302.05405, 2023.

[5] H. LI, M. YIN et Z. LI : Failure based variable ordering heuristics for solving CSPs. *In Proceedings of CP'21*, 2021.

[6] Charles PRUD'HOMME et Jean-Guillaume FAGES : Choco-solver : A Java library for constraint programming. *Journal of Open Source Software*, 7(78): 4708, 2022.

[7] H. WATTEZ, C. LECOUTRE, A. PAPARRIZOU et S. TABARY : Refining constraint weighting. *In Proceedings of ICTAI'19*, pages 71–77, 2019.