

Un algorithme tabou trivalué hybride pour le problème MAX-SAT

Frédéric Lardeux, Frédéric Saubion et Jin-Kao Hao
LERIA, Université d'Angers,
2 Bd Lavoisier, F-49045 Angers Cedex 01
email: {lardeux,saubion,hao}@info.univ-angers.fr

1 Introduction

Le problème de satisfiabilité (SAT) consiste à trouver une affectation booléenne validant une formule en logique propositionnelle. Une instance de ce problème est définie par un ensemble de variables booléennes (aussi appelées atomes) $\mathcal{X} = \{x_1, \dots, x_n\}$ et une formule booléenne $\phi : \{0, 1\}^n \rightarrow \{0, 1\}$. Un littéral est une variable ou sa négation; une clause est une disjonction de littéraux. On considère en général des formules en forme normale conjonctive (CNF) c'est à dire des conjonctions de clauses. Une affectation est une fonction $v : \mathcal{X} \rightarrow \{0, 1\}$. Une formule est vraie si et seulement si toutes ses clauses sont vraies. La formule ϕ est dite satisfiable s'il existe une affectation rendant ϕ vraie et est non satisfiable s'il n'en existe pas.

Le problème MAX-SAT, auquel nous nous intéressons plus particulièrement ici, peut alors être vu comme une formulation alternative du problème SAT qui consiste à trouver une affectation maximisant le nombre de clauses de la formule satisfaites pour une affectation.

Deux classes de méthodes peuvent être utilisées pour résoudre le problème SAT : les méthodes complètes qui permettent de trouver toutes les solutions d'une instance ou, le cas échéant, de déterminer sa non satisfiabilité et les méthodes incomplètes, principalement des algorithmes évolutionnaires et des algorithmes à base de recherche locale, qui cherchent à maximiser, sur l'espace des affectations, une fonction correspondant au nombre de clauses satisfaites.

L'objectif de cet article est de proposer un cadre algorithmique homogène permettant de combiner des stratégies incomplètes et complètes, afin d'améliorer la résolution. De telles hybridations ont déjà été étudiées [6, 2, 5] mais sans réelle interaction entre les deux stratégies.

Chacune de ces méthodes possède ses avantages propres. Les méthodes complètes telle que la procédure Davis Putnam Loveland [3] (DPL) permettent d'effectuer une recherche exhaustive en travaillant sur des affectations partielles, qui sont progressivement complétées au fur et à mesure de la descente dans l'arbre de recherche, et donc d'exploiter

complètement l'espace de recherche. Les méthodes de recherche locale telle que Tabou [4] qui travaillent sur des affectations complètes limitant leur exploration à certaines zones, ce qui s'avère particulièrement intéressant pour des problèmes de grande taille. Notre idée est donc d'intégrer ce pouvoir d'intensification de DPL (exploration exhaustive d'un espace de recherche) au sein d'un algorithme Tabou qui possède des facultés de diversification et ce, afin d'obtenir un bon compromis entre ces deux stratégies fondamentales de recherche. Afin d'unifier ces approches, nous définissons un cadre de résolution en introduisant trois valeurs de vérité. Cette représentation permet d'intégrer de manière plus homogène un processus de diversification à la recherche Tabou ainsi qu'une stratégie d'intensification par le biais d'une procédure DPL. Nous procédons alors à une étude expérimentale de notre algorithme afin de mettre en évidence les atouts d'une telle hybridation.

2 Tabou à 3 valeurs

Pour que l'hybridation avec DPL soit efficace, il faut unifier ces deux méthodes dans une même représentation des éléments de base, c.à.d les affectations. L'ajout d'une troisième valeur de vérité *indéterminé* permet de faire travailler Tabou sur une affectation partielle (incomplète). Tabou et DPL peuvent alors s'enchaîner de manière homogène. Outre une meilleure cohésion entre les deux méthodes, cette troisième valeur permet également de diversifier la recherche Tabou sans ajout d'heuristique externe.

En effet, alors que le Tabou à deux valeurs de vérité parcourt l'espace de recherche en passant d'une affectation à une autre par une succession de flips¹, le Tabou à trois valeurs permet de se déplacer dans l'espace de recherche avec une affectation partiellement indéterminée qui peut être considérée comme la représentation d'un ensemble d'affectations totales. Dès que la réduction de cet ensemble permet d'augmenter le nombre de clauses vraies sans créer de nouvelles clauses fausses, Tabou instancie certaines variables aux valeurs *vrai* ou *faux*. De cette manière la recherche se diversifie lors de la création de variables indéterminées et s'intensifie lors de la valuation de variables à *vrai* ou à *faux*.

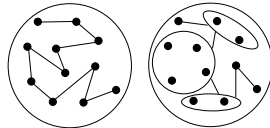


FIG. 1 – *Différentes évolutions de la recherche pour les deux Tabous : à gauche, le Tabou à deux valeurs de vérité se déplace dans l'espace de recherche d'affectation en affectation. À droite, le Tabou à trois valeurs de vérité peut, quant à lui, passer d'une affectation à un ensemble d'affectations en rendant indéterminées une ou plusieurs variables ou inversement, d'un ensemble d'affectations à un ensemble plus petit en valant une variable indéterminée.*

Afin d'inciter l'algorithme à rendre des variables indéterminées, une clause est considérée indéterminée si et seulement si toutes ses variables sont indéterminées. Ce point de vue

1. un flip représente le changement de valeur d'une variable

plutôt pessimiste permet d'éviter la création massive de clauses unitaires et d'utiliser ainsi au maximum la puissance de la procédure DPL.

3 Adaptation de DPL pour MAX-SAT

Dans le cadre du problème SAT, l'idée de la procédure DPL est de couper une branche dès qu'une clause fausse est atteinte. DPL élague alors l'arbre de recherche jusqu'à l'obtention d'une solution ou l'élagage de toutes les branches et donc la preuve que le problème est insatisfiable. Ce mécanisme permet de couper très rapidement des branches mais pour le problème MAX-SAT, il n'est plus utilisable tel quel. En effet, il ne faut pas couper une branche dès qu'une clause fausse est rencontrée mais développer toutes les branches et, pour chacune d'elles, compter le nombre de clauses fausses rencontrées. La feuille qui génère le moins de clauses fausses est la meilleure affectation pour le problème MAX-SAT. Bien sûr, il est beaucoup trop coûteux d'explorer ainsi tout l'arbre de recherche et il est possible de couper des branches afin de réduire la taille de l'arbre. La première branche est alors explorée entièrement et fournit un nombre de clauses fausses de référence. Ensuite DPL continue son exploration mais ne coupe une branche que si le nombre de clauses fausses est supérieur à la meilleure valeur déjà trouvée pour une affectation complète. Cette approche est une méthode de type branch and bound comme celles proposées dans [1, 7].

4 Hybridation Tabou-DPL

Le processus d'hybridation (figure 2) passe constamment d'une méthode à l'autre en conservant les informations obtenues par chacune d'elles afin de mieux guider la recherche d'un point de vue global.

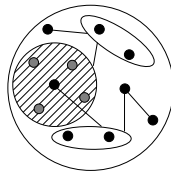


FIG. 2 – Hybridation de Tabou à trois valeurs de vérité et de DPL pour MAX-SAT. Après une phase de recherche Tabou, DPL cherche la meilleure affectation du sous-ensemble sélectionné par Tabou (zone hachurée). Ensuite, Tabou redémarre sa recherche jusqu'au prochain appel à DPL.

Tabou effectue sa recherche jusqu'à obtenir une affectation contenant moins de k (empiriquement fixé à 50) variables ayant la valeur indéterminée avec comme condition qu'au moins n flips aient été opérés depuis le dernier appel à DPL (afin d'équilibrer le travail entre Tabou et DPL). Si le nombre de variables indéterminées est alors inférieur ou égal à k , d'autres variables sont aléatoirement sélectionnées parmi celles déjà évaluées et rendues indéterminées. Ainsi, DPL peut être exécuté sur un sous-arbre (vis à vis du problème initial) et retourne la meilleure affectation du sous-espace de recherche

représenté par l'affectation ayant k variables indéterminées. Tabou reprend ensuite sa recherche à partir de la meilleure affectation trouvée par DPL et, afin d'éviter de chercher de nouveau parmi le sous-ensemble d'affectations étudiées par DPL, toutes les variables qui étaient indéterminées sont ajoutées à la liste taboue.

5 Expérimentations

Dans cette section, l'hybridation entre le Tabou trivalué et DPL est comparée à un algorithme Tabou classique sur différents jeux de test bien connus. Pour comparer la qualité des deux algorithmes, trois critères sont utilisés. Les deux premiers sont logiquement la moyenne sur l'ensemble des exécutions du nombre minimum de clauses non satisfaites obtenues et son écart type (e.t.). Un autre critère important est le nombre de flips (fl.) nécessaires pour atteindre cette valeur.

Benchmarks			Tabou			Tabou+DPL		
instances	var	cls	clauses moy.	fausses e.t.	fl.	clauses moy.	fausses e.t.	fl.
color-10-3	300	6475	2.60	0.50	219359	2.45	0.76	256962
color-15-4	900	45675	5.30	0.52	263991	5.20	0.57	341175
par32-5-c	1339	5350	14.70	11.38	303836	11.70	6.66	219720
par32-5	3176	10325	23.70	30.04	361385	10.20	1.32	473116
ssa7552-038	1501	3575	10.00	13.06	407185	8.55	1.88	553387
aim-100-2_0-yes1-1	100	200	2.90	2.69	165.55	1.20	0.70	5992.25
aim-200-2_0-yes1-3	200	400	3.95	3.53	1306.90	1.35	1.14	16363
gencnf	85	7539	1.00	0.00	56248	0.80	0.45	197866

TAB. 1 – Comparaisons entre le Tabou trivalué hybridé avec DPL et un Tabou classique

Le tableau 1 montre que l'hybridation Tabou+DPL permet d'obtenir de meilleurs résultats qu'une méthode Tabou classique. En plus d'un nombre de clauses fausses moyen plus faible, Tabou+DPL a une meilleure fiabilité dans ses résultats comme le prouve les écarts types très faibles par rapport au Tabou classique.

Références

- [1] T. Alsinet, F. Manya, and J. Planes. Improved branch and bound algorithms for MAX-SAT. In *Proc of the 6th Int. Conf. SAT2003*, pages 408–415, 2003.
- [2] B. Borchers and J. Furman. A two-phase exact algorithm for MAX-SAT and weighted MAX-SAT problems. *Journal of Combinatorial Optimization*, 2:299–306, 1999.
- [3] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, Jul 1962.
- [4] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, 1997.
- [5] D. Habet, CM. Li, L. Devendeville, and M. Vasquez. A Hybrid Approach for SAT. In *Int. Conf. on Constraint Programming (CP), LNCS*, 2002.
- [6] B. Mazure, L. Saïs, and E. Grégoire. Boosting complete techniques thanks to local search methods. *Annals of Mathematics and Artificial Intelligence*, 22:319–331, 1998.
- [7] H. Zhang, H. Shen, and F. Manyà. Exact algorithms for MAX-SAT. *Electronic Notes in Theoretical Computer Science*, volume 86. Elsevier, 2003.