HAL open science

Max-réfutations et oracles SAT

Matthieu Py, Mohamed Sami Cherif, Djamal Habet

▶ **To cite this version:**

Matthieu Py, Mohamed Sami Cherif, Djamal Habet. Max-réfutations et oracles SAT. JFPC 2022, Jun 2022, Saint-Etienne, France. hal-03737731

HAL Id: hal-03737731

https://hal-amu.archives-ouvertes.fr/hal-03737731

Submitted on 25 Jul 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Max-réfutations et oracles SAT

Matthieu Py, Mohamed Sami Cherif, Djamal Habet

Aix-Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France

{ matthieu.py, mohamedsami.cherif, djamal.habet}@univ-amu.fr

Résumé

Adapter une réfutation par résolution en max-réfutation sans en augmenter considérablement sa taille est une question ouverte depuis l'introduction de la max-résolution. Cet article contribue à cette problématique en proposant un algorithme nommé algorithme de génération de remplaçants, capable d'adapter n'importe quelle réfutation par résolution en max-réfutation grâce à des appels à un oracle SAT. En particulier, on démontre que cet algorithme adapte efficacement les motifs en diamant, dont l'adaptation est exponentielle dans la littérature. Cet article résume le travail publié à la conférence ICTAI 2021 [3].

Mots-clés

Max-SAT, Résolution, Max-Réfutation

1 Introduction

Étant donnée une formule sous Forme Normale Conjonctive, le problème Max-SAT consiste à déterminer le nombre maximum de clauses qu'il est possible de satisfaire par une affectation des variables alors que le problème SAT consiste simplement à déterminer si la formule est satisfiable. Dans le contexte du problème SAT, une formule peut être démontrée insatisfiable à l'aide d'une séquence de résolutions [4], appelée réfutation par résolution, qui déduit de la formule initiale de nouvelles clauses jusqu'à en déduire la clause vide. De même, pour Max-SAT, on utilise un système de preuve bien connu basé sur la règle de la max-résolution [1], qui étend la règle de résolution utilisée dans SAT. Les séquences de max-résolutions sont plus contraintes que les séquences de résolutions car la max-résolution remplace les prémisses par les conclusions, ce qui consomme les prémisses et empêche de les utiliser à nouveau. Adapter une réfutation SAT (par résolution) en une réfutation valide pour Max-SAT est par conséquent un problème difficile et jusqu'à il y a peu, on ne savait le faire que si la formule en entrée est *read-once*, c'est à dire si chaque clause est utilisée une seule fois comme prémisse d'une résolution. Des travaux récents proposent une adaptation de n'importe quelle réfutation par résolution pour Max-SAT (exemple 2) mais celle-ci entraîne une augmentation exponentielle de la taille de la formule dans le pire des cas [2].

Dans cet article, on propose d'apporter une autre alternative pour l'adaptation des réfutations par résolution pour Max-SAT (on parle de max-réfutation). Pour cela, on utilise un algorithme qui suit la réfutation et remplace chaque résolution par une max-résolution et, lorsqu'il a besoin d'un remplaçant pour une clause qui a été consommée, fait appel à un oracle SAT pour que ce remplaçant soit créé.

Exemple 1. *Soit* $\phi = (\overline{x_1} \lor x_3) \land (x_1) \land (\overline{x_1} \lor x_2) \land (\overline{x_2} \lor \overline{x_3})$. *Une réfutation par résolution de* ϕ *est représentée dans la Figure 1.*

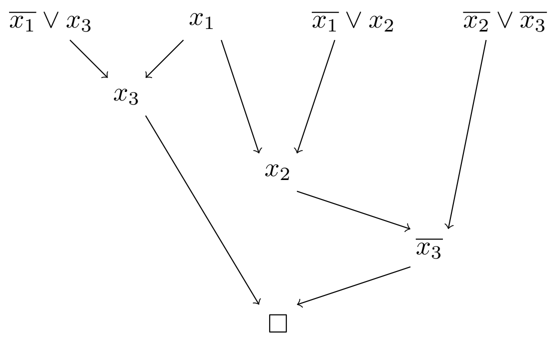

FIGURE 1 – Resolution refutation

Exemple 2. *On considère la résolution par réfutation de l'exemple 1. L'adaptation de cette réfutation pour Max-SAT d'après [2] est représenté dans la Figure 2.*

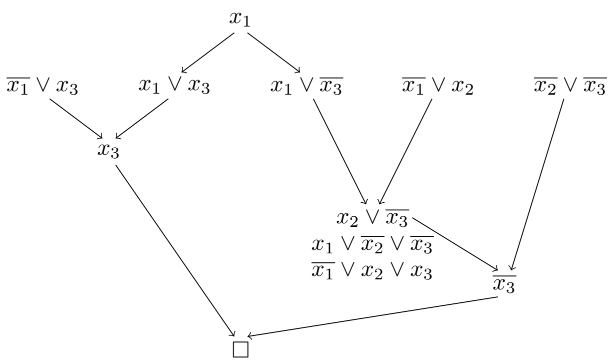

FIGURE 2 – Adaptation d'une en max-réfutation [2]

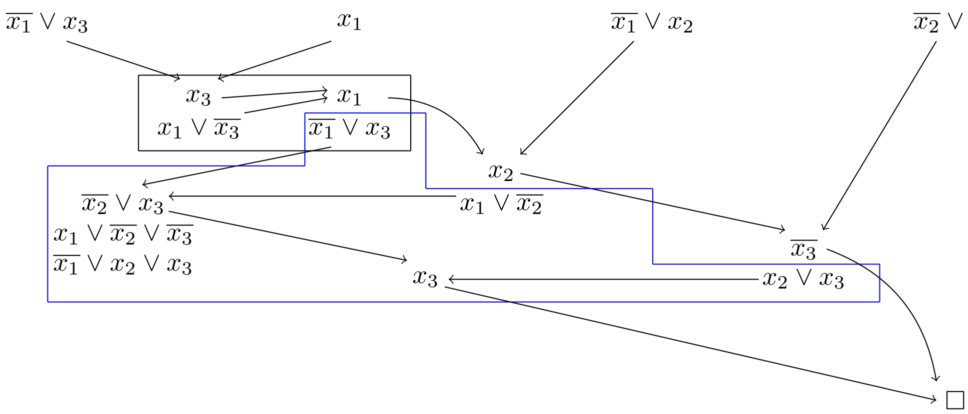

FIGURE 3 – Max-Réfutation calculée grâce à l'algorithme de génération de remplaçants

2 L'algorithme de génération de remplaçants

L'idée de l'algorithme de génération de remplaçants est de suivre la liste des étapes de résolution de la réfutation initiale, de remplacer chaque étape de résolution par une max-résolution, et d'appliquer cette étape sur la formule courante. Au cours du processus, si une clause nécessaire à une étape de max-résolution est manquante, on fait appel à un oracle SAT pour régénérer cette clause. Pour cela, on propage la falsification de la clause manquante et toute réfutation par résolution de la formule obtenue donne une transformation vers la clause manquante. Notez que le procédé est ré-appliqué récursivement pour générer le remplaçant.

Exemple 3. *On considère la réfutation par résolution de l'exemple 1 (plus de détails dans [2]).*

Première résolution : *La première étape est sur $(\overline{x_1} \vee x_3)$ et (x_1) qui sont dans la formule. On remplace donc la résolution par la max-résolution que l'on applique, et la formule est maintenant $\phi = (\overline{x_1} \vee x_2) \wedge (\overline{x_2} \vee \overline{x_3}) \wedge (x_3) \wedge (x_1 \vee \overline{x_3})$.*

Deuxième résolution (début) : *La deuxième étape est sur (x_1) et $(\overline{x_1} \vee x_2)$. On doit générer un remplaçant pour la clause (x_1). Pour cela, on propage $\overline{x_1}$, on calcule une réfutation par résolution et on annule la propagation, on obtient une étape de résolution sur (x_3) et $(x_1 \vee \overline{x_3})$ pour avoir notre remplaçant.*

Première résolution (niveau 2) : *Les clauses (x_3) et $(x_1 \vee \overline{x_3})$ sont dans la formule courante, on remplace par une max-résolution, on applique et on obtient $\phi = (\overline{x_1} \vee x_2) \wedge (\overline{x_2} \vee \overline{x_3}) \wedge (x_1) \wedge (\overline{x_1} \vee x_3)$.*

Deuxième résolution (fin) : *Les clauses (x_1) et $(\overline{x_1} \vee x_2)$ sont maintenant dans la formule. On remplace la résolution par la max-résolution, on l'applique, et la formule est maintenant $\phi = (\overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_3) \vee (x_2) \wedge (x_1 \vee \overline{x_2})$.*

Troisième résolution : *La troisième étape est sur (x_2) et $(\overline{x_2} \vee \overline{x_3})$ qui sont dans la formule. On remplace donc la résolution par la max-résolution que l'on applique, et la formule est maintenant $\phi = (\overline{x_1} \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge (\overline{x_3}) \wedge (x_2 \vee x_3)$.*

Quatrième résolution (début) : *La quatrième étape est sur (x_3) et $(\overline{x_3})$. On doit générer un remplaçant pour (x_3). Pour cela, on propage $\overline{x_3}$, on calcule une réfutation et on annule la propagation ; on obtient deux étapes de résolution sur $(x_1 \vee \overline{x_2})$ et $(\overline{x_1} \vee x_3)$ puis sur $(\overline{x_2} \vee x_3)$ et $(x_2 \vee x_3)$.*

Première résolution (niveau 2) : *Les clauses $(x_1 \vee \overline{x_2})$ et $(\overline{x_1} \vee x_3)$ sont dans la formule, on remplace par une max-résolution, on applique et on obtient $\phi = (\overline{x_3}) \wedge (x_2 \vee x_3) \wedge (\overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_3)$.*

Première résolution (niveau 2) : *Les clauses $(\overline{x_2} \vee x_3)$ and $(x_2 \vee x_3)$ sont dans la formule, on remplace par une max-résolution, on applique et on obtient $\phi = (\overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_3) \wedge (x_3)$.*

Quatrième résolution (fin) : *Les clauses (x_3) and $(\overline{x_3})$ sont maintenant dans la formule. On remplace la résolution par la max-résolution, on l'applique, et la formule est maintenant $\phi = (\overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_3) \wedge (x_3)$. L'exécution de l'algorithme est terminée et la max-réfutation obtenue est représentée Figure 3.*

Références

[1] María Luisa Bonet, Jordi Levy, and Felip Manyà. Resolution for Max-SAT. *Artificial Intelligence Volume 171, Issues 8–9, June 2007, Pages 606-618*, 2007.

[2] Matthieu Py, Mohamed Sami Cherif, and Djamal Habet. Towards Bridging the Gap Between SAT and Max-SAT Refutations. In *32nd IEEE International Conference on Tools with Artificial Intelligence, (ICTAI)*, pages 137–144. IEEE, 2020.

[3] Matthieu Py, Mohamed Sami Cherif, and Djamal Habet. Computing Max-SAT Refutations using SAT Oracles. In *33rd IEEE International Conference on Tools with Artificial Intelligence, (ICTAI)*. IEEE, 2021.

[4] J. A. Robinson. A machine-oriented logic based on the resolution principle. In *Journal of the Association for Computing Machinery, vol. 12, pages 23–41*, 1965.