WMaxCDCL in MaxSAT Evaluation 2023

1st Jordi Coll
Artificial Intelligence Research Institute
CSIC, Bellaterra, Spain
jcoll@iiia.csic.es

2nd Shuolin Li
Aix Marseille Univ,Université de Toulon
CNRS, LIS, Marseille, France
shuolin.li@etu.univ-amu.fr

3rd Chu-Min Li
Université de Picardie Jules Verne,
Amiens, France
Aix Marseille Univ,Université de Toulon
CNRS, LIS, Marseille, France
chu-min.li@u-picardie.fr

4th Felip Manyà
Artificial Intelligence Research Institute
CSIC, Bellaterra, Spain
felip@iiia.csic.es

5th Djamal Habet, 6th Mohamed Sami Cherif
Aix Marseille Univ,Université de Toulon
CNRS, LIS, Marseille, France
Djamal.Habet@univ-amu.fr, mohamed-sami.cherif@univ-amu.fr

7th Kun He
Huazhong University of Science and Technology
Wuhan,China
brooklet60@hust.edu.cn

I. Introduction

WMaxCDCL participated for first time in MaxSAT Evaluation 2022. Its main algorithm is MaxCDCL [1], an algorithm that combines Branch and Bound and clause learning. WMaxCDCL solves Weighted Partial MaxSAT, and its first version was developed as a modification of the MaxCDCL solver for unweighted Partial MaxSAT from [1]. Both MaxCDCL and WMaxCDCL have been independently updated since the version in [1], though some common features have been implemented in both solvers.

II. WMaxCDCL Algorithm

The MaxCDCL algorithm is an extension for MaxSAT of the CDCL algorithm which combines Branch and Bound and clause learning. Similarly as done in CDCL, the MaxCDCL algorithm roughly alternates decisions and unit propagation with conflict analysis and clause learning. Moreover, at some selected nodes of the search tree, MaxCDCL computes a lower bound (LB) of the number of soft clauses that will be falsified in any solution that satisfies the hard clauses. If the bounding procedure detects that the current assignment cannot be extended to a satisfying assignment that improves the best solution found so far, i.e. $LB \geq UB$, a *soft conflict* is detected. Similarly to (hard) conflicts in CDCL, which can also occur in MaxCDCL, and where a hard clause is falsified, MaxCDCL detects an implicit clause that is falsified when a soft conflict occurs. Both after hard and soft conflicts, conflict analysis is used to find the first unique implication point and backtrack. In addition, when the lower bounding procedure does not detect a soft conflict but $LB + \omega(c) \geq UB$, for some soft clause c with weight $\omega(c)$, such soft clause can be hardened. This hardening is done by unit propagation after introducing new clauses explaining the reason of the hardening.

The computation of the lower bound is based on the detection of local unsatisfiable cores, i.e. cores that depend on the current partial assignment. Roughly, the detection of a local core is done by assuming soft clauses to be true and applying unit propagation until some conflict is found [2]–[4]. For every detected local core (set of soft clauses), the lower bound can be increased by the minimum weight of its soft clauses.

III. Implementation Details

The solver first tries to find an initial feasible cost, by solving the problem with a sequence of increasing upper bound values UB starting at 1, and increased by $UB \leftarrow min(1.5 \cdot UB, initUB)$ until a feasible solution is found or *initUB* is reached. Here, *initUB* is either a trivial upper bound (sum of weights of soft clauses) or an upper bound computed externally and received as input. Then, the optimization procedure continues by decreasing UB until the optimal solution is found and proven.

Before starting the search we find incompatible subsets of soft clauses by unit propagation, i.e. sets of soft clauses such that at most one of them can be satisfied according to hard clauses. Then, for every set of weighted clauses $(c_1, \omega(c_1)), \ldots, (c_n, \omega(c_n))$, we derive a fixed cost of $m(n-1)$, where m is the minimum of the weights $\omega(c_1), \ldots, \omega(c_m)$.

When the number of free soft clauses n and the upper bound UB are small, we add as implied constraints a CNF encoding of pseudo-Boolean constraints, expressing that the cost of the solution must be smaller than the best found upper bound. In particular, we add the MDD encoding [5] when $n \leq 50$ and $n \cdot K \leq 10^5$, and otherwise the GGPW encoding [6] when $n \leq 500$ or $n \leq 5000$ and $n \cdot K \leq 10^5$.

We also implement a number of inprocessing algorithms, namely failed literal detection, equivalent literal substitution,

clause simplification, and solution improvement with local search by means of a custom implementation of the local search method described in [7].

IV. WMaxCDCL in MaxSAT Evaluation 2023

Three versions of WMaxCDCL are submitted in MaxSAT Evaluation 2023.

The first one is the pure WMaxCDCL solver without using any third-party solver.

The second one precedes the execution of WMaxCDCL with, first, 10 minutes of the SCIP solver [8]. If no solution is found, then the MaxHS solver [9] is run for 20 more minutes. Finally, if the instance is not solved, WMaxCDCL is run for the remaining time, taking the last solution found by MaxHS as *initUB*. In order to run the SCIP solver, we use the UWrMaxSAT-SCIP implementation [10], which includes instance parsing and preprocessing. We specify that the 900 seconds must used by SCIP, unless UWrMaxSAT-SCIP internally decides to stop or avoid the SCIP computation and continue with other solving techniques.

The third version is the same as the second one but giving 15 minutes to SCIP and 15 minutes to MaxHS.

Acknowledgments

This work has been partially supported by AI CHAIR reference ANR-19-CHIA-0013-01 (MASSAL'IA) and project ANR-20-ASTR-0011 (POSTCRYPTUM) funded by the French Agence Nationale de la Recherche, and projects PID2019-111544GB-C21 and TED2021-129319B-I00 funded by MCIN/AEI/10.13039/501100011033, and partially supported by Archimedes Institute, Aix-Marseille University. We thank the Université de Picardie Jules Verne for providing the Matrics Platform.

References

[1] C.-M. Li, Z. Xu, J. Coll, F. Manyà, D. Habet, and K. He, "Combining clause learning and branch and bound for maxsat," in *27th International Conference on Principles and Practice of Constraint Programming (CP 2021)*, 2021.

[2] C. M. Li, F. Manyà, N. O. Mohamedou, and J. Planes, "Exploiting cycle structures in Max-SAT," in *In Proceedings of SAT 2009*, ser. LNCS, vol. 5584. Springer, 2009, pp. 467–480.

[3] C. M. Li, F. Manyà, and J. Planes, "Exploiting unit propagation to compute lower bounds in branch and bound Max-SAT solvers," in *Proceedings of CP 2005*, ser. LNCS, vol. 3709. Springer, 2005, pp. 403–414.

[4] ——, "Detecting disjoint inconsistent subformulas for computing lower bounds for Max-SAT," in *Proceedings of AAAI 2006*, 2006, pp. 86–91.

[5] M. Bofill, J. Coll, J. Suy, and M. Villaret, "An mdd-based SAT encoding for pseudo-boolean constraints with at-most-one relations," *Artificial Intelligence Review*, vol. 53, no. 7, pp. 5157–5188, 2020.

[6] M. Bofill, J. Coll, P. Nightingale, J. Suy, F. Ulrich-Oltean, and M. Villaret, "SAT encodings for pseudo-boolean constraints together with at-most-one constraints," *Artificial Intelligence*, vol. 302, p. 103604, 2022.

[7] J. Zheng, K. He, J. Zhou, Y. Jin, C.-M. Li, and F. Manya, "Bandmaxsat: A local search MaxSAT solver with multi-armed bandit," in *Proceedings of 31st International Joint Conference on Artificial Intelligence (To appear)*, 2022.

[8] K. Bestuzheva et al., "The SCIP Optimization Suite 8.0," Optimization Online, Technical Report, December 2021.

[9] F. Bacchus, "Maxhs in the 2022 maxsat evaluation," *MaxSAT Evaluation 2022*, pp. 17–18, 2022.

[10] M. Piotrów, "Uwrmaxsat entering the maxsat evaluation 2022," *MaxSAT Evaluation 2022*, pp. 21–22, 2022.