

WMaxCDCL in MaxSAT Evaluation 2024

1st Shuolin Li

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

2nd Chu-Min Li*

Université de Picardie Jules Verne,
MIS, Amiens, France
Aix Marseille Univ, Université de Toulon
CNRS, LIS, Marseille, France
chu-min.li@u-picardie.fr

3rd Jordi Coll

Universitat de Girona
Girona, Spain
jordicoll@udg.edu

4th Djamal Habet

Aix Marseille Univ, Université de Toulon
CNRS, LIS, Marseille, France
Djamal.Habet@univ-amu.fr

5th Felip Manyà

Artificial Intelligence Research Institute
CSIC, Bellaterra, Spain
felip@iiia.csic.es

6th Kun He

Huazhong University of
Science and Technology
Wuhan, China
brooklet60@hust.edu.cn

Abstract—This document is an introduction of our submitted solvers to MaxSAT Evaluation 2024.

I. INTRODUCTION

WMaxCDCL2024 solves Weighted Partial MaxSAT instances. It is developed from WMaxCDCL2023 [1] which won the weighted partial category of MaxSAT Evaluation 2023. Its main algorithm is MaxCDCL [3], an algorithm that combines **Branch and Bound(BnB)** and **clause learning**. As a BnB solver, it computes for a partial assignment a lower bound (LB) of the total weight of soft clauses that will be falsified when the partial assignment is extended, and compares LB with an imposed upper bound (UB). This process is called *lookahead*. If $LB \geq UB$, no solution better than UB can be found. So, the solver prunes and backtracks; otherwise the search continues. To better achieve clause learning, it transforms each soft clause c into a soft literal s with the same weight, so that the value of s is true if c is satisfied and vice versa. The soft clauses are removed from the formula after the transformation, and the solver works with soft literals.

Some new features have been added into WMaxCDCL2024 w.r.t. WMaxCDCL2023.

II. IN-PROCESSING BOUNDED VARIABLE ELIMINATION AND EQUIVALENT LITERAL SUBSTITUTION

The implementation details of Bounded Variable Elimination(BVE) and Equivalent Literal Substitution(ELS) are similar to those in MaxCDCL2023 [2], but some modifications are done in ELS to fit the property of Weighted Partial MaxSAT.

If soft literals s_1 and s_2 are complementary in MaxCDCL, i.e., s_1 is satisfied if and only if s_2 is falsified, then the constant cost is increased by 1, and both literals are removed, while in WMaxCDCL the literal with higher weight is kept and its weight updated to the difference of two weights. If soft literal s_1 and s_2 are equivalent, nothing is done in MaxCDCL, while in WMaxCDCL the two literals could be unified by removing

one of them and the sum of their weights becomes the weight of the remaining one.

III. IMPROVEMENTS IN LOOKAHEAD

In WMaxCDCL, the key to efficient pruning is to compute a high-quality LB for a partial assignment α . For this purpose, it detects local cores w.r.t. α using unit propagation, where a local core w.r.t. α is a subset of soft literals that cannot all be satisfied without falsifying a hard clause under α . So, the key property of a local core is that there is at least one soft literal falsified by any complete assignment extended from α , and we define the weight of a core to be the minimum weight of the soft literals it contains. After detecting the core, the weight of any soft literal in the core is reduced by the weight of the core. We say that a core locks a part of weight of each literal in it. To guarantee the soundness of LB , the locked weight cannot be considered in other cores, and only the soft literals with non-zero (remaining) weight participate in the detection of other cores. Thus, the sum of weights of all detected cores and of weights of all soft literal already falsified under α is a lower bound LB .

Two main improvements are introduced in lookahead in WMaxCDCL2024 that are described in this section.

A. Reinforcement learning for hiring Lookahead

Lookahead is time consuming, and if the computed LB does not reach UB , i.e., if the LB computation is not successful, the effort spent to compute LB is lost. Consequently, WMaxCDCL does not detect local cores for all partial assignments. Instead, the previous versions of WMaxCDCL use a probing technique to estimate the average a and standard deviation d of the number of disjoint local cores detected in a successful LB computation, and decides whether the LB computation should be hired for a partial assignment based on a and d .

In WMaxCDCL2024, we use reinforcement learning with 2 armed-bandit mechanism to decide if LB computation should be hired. Concretely, two actions, *yes* and *no*, are associated with each k , where k is the number of disjoint local cores for

*Corresponding author

a LB computation to be successful. Initially, the score of *yes* and *no* is initialized to 0. If a LB computation is hired and is successful, then the score are updated as follows.

$$\text{score}(\text{yes}) \leftarrow \text{score}(\text{yes}) + \text{stepSize} * (1 - \text{score}(\text{yes}))$$

$$\text{score}(\text{no}) \leftarrow \text{score}(\text{no}) * \text{stepSize} * (1 - \text{score}(\text{no}))$$

If the LB computation is not successful, then

$$\text{score}(\text{yes}) \leftarrow \text{score}(\text{yes}) * \text{stepSize} * (1 - \text{score}(\text{yes}))$$

$$\text{score}(\text{no}) \leftarrow \text{score}(\text{no}) + \text{stepSize} * (1 - \text{score}(\text{no}))$$

where *stepSize* is a parameter initialized to 0.4 and is decreased by 10^{-5} before every LB computation until 0.04.

LB computation should be hired if $c * \text{score}(\text{yes}) \geq \text{score}(\text{no})$, where *c* is initialized to 1 and is adjusted to maintain the success rate of LB computation to be between 0.6 and 0.75.

B. Unlocking Cores during Lookahead

In previous versions of WMaxCDCL, a soft literal with remaining weight 0 cannot participate in any further core, because all its weight is locked in existing cores. We say that such a literal is *locked*. The LB computation is based on the very conservative hypothesis that there is only one falsified soft literal *s* in each core and that *s* necessarily has the minimum weight in the core. In WMaxCDCL2024, we develop a core unlocking mechanism to detect cores that have more than one falsified soft literals, so that better LB can be computed.

We illustrate the core unlocking mechanism using an example. When we say we propagate a literal *s*, we mean that we assign true to *s* and execute unit propagation (UP); and when we say UP falsifies *s*, we mean that UP produced a hard unit clause $\neg s$. Let $c_1 = \{s_1, s_2, s_3\}$ be a core and the remaining weight of *s*₁ and *s*₂ is 0 so that they cannot participate in any further core in previous versions of WMaxCDCL. The left of Fig. 1 shows the detection of *c*₁: propagating *s*₁ and then *s*₂ falsifies *s*₃, meaning that *s*₁, *s*₂, *s*₃ cannot be all satisfied without falsifying a hard clause w.r.t. α .

To detect a new core, we propagate a soft literal *s*₄ with non-zero remaining weight. Suppose that UP falsifies *s*₁. Then we unlock *c*₁, so that *s*₂ can be propagated, which falsifies a soft literal *s*₅ with non-zero remaining weight. We now show that the subset $\{s_1, s_2, s_3, s_4, s_5\}$ has at least two falsified soft literals under any assignment extending α . If *s*₄ is false, then the two falsified literals are *s*₄ and one of *s*₁, *s*₂, *s*₃ because $\{s_1, s_2, s_3\}$ is a core (case 1). If *s*₄ is true, then the two falsified literals are *s*₁, *s*₂ (case 2), or *s*₁, *s*₅ (case 3) according to UP. The right of Fig. 1 shows the three cases modulo UP. Let *w*(*s*) (*w*(*c*)) denote the weight of soft literal *s* (core *c*), where *w*(*s*) is the sum of the remaining weight of *s* and the part of weight of *s* locked in *c*₁. The weight of the core $c_2 = \{s_1, s_2, s_3, s_4, s_5\}$ is defined to be the minimum among $w(s_4) + w(c_1)$, $w(s_2) + w(s_1)$ and $w(s_5) + w(s_1)$. The part of weight of *s*₁, *s*₂, *s*₃ locked in *c*₂ is equal to their part locked in *c*₁, and the part of weight of *s*₄ and *s*₅ locked in *c*₂ is

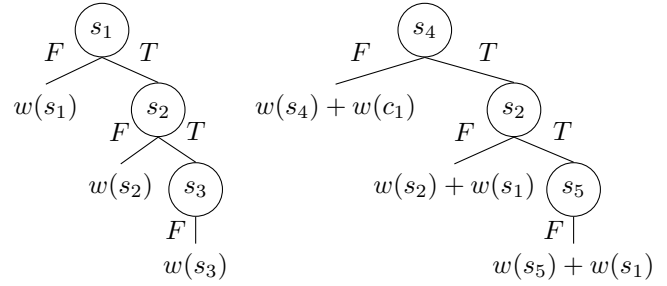


Fig. 1. Example illustrating core unlocking mechanism

equal to $w(c_2) - w(c_1)$. The remaining weight of all literals is computed accordingly. Finally, *c*₁ is removed.

Note that the new core $c_2 = \{s_1, s_2, s_3, s_4, s_5\}$ may not be detected if *c*₁ is not unlocked and *s*₂ is not propagated.

In the general case, an existing core is unlocked and its unassigned literals can be propagated during the detection of a new core once the total weight of falsified soft literals in the core is equal to or greater than the weight of the core.

In practice, WMaxCDCL2024 first executes normal lookahead without unlocking any existing core. If the computed LB does not reach UB because any new core cannot be detected after propagating all soft literals with non-zero weight, it re-executes lookahead with unlocking to detect more cores.

IV. VERSIONS SUBMITTED TO MSE2024

We submit two versions to MSE2024. The first is WMaxCDCL2024 itself. The second combines WMaxCDCL2024 and Open-WBO [4] (version MSE2023) in a portfolio in which Open-WBO is first executed 1200s, followed by WMaxCDCL2024. Note that Open-WBO, WMaxCDCL2023 and MaxCDCL2023 are the only exact solvers of MSE2023 which pass successfully the regression test required in MSE2024.

ACKNOWLEDGMENTS

This work is 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 Matrices Platform.

REFERENCES

- [1] Jordi Coll, Shuolin Li, Chu-Min Li, Felip Manyà, Djamal Habet, Mohamed Sami Cherif, and Kun He. Wmaxcdcl in maxsat evaluation 2023. *MaxSAT Evaluation 2023*, pages 16–17, 2023.
- [2] Chu-Min Li, Jordi Coll, Shuolin Li, Djamal Habet, Felip Manyà, and Kun He. Maxcdcl in maxsat evaluation 2023. *MaxSAT Evaluation 2023*, pages 14–15, 2023.
- [3] Chu-Min Li, Zhenxing Xu, Jordi Coll, Felip Manyà, Djamal Habet, and Kun He. Combining clause learning and branch and bound for maxsat. In *27th International Conference on Principles and Practice of Constraint Programming (CP 2021)*, 2021.
- [4] Ruben Martins, Vasco M. Manquinho, and Inês Lynce. Open-WBO: A modular MaxSAT solver. In *Proceedings of SAT 2014*, volume 8561 of LNCS, pages 438–445. Springer, 2014.