Applying Nested Monte Carlo and Sequential Halving on unweighted and weighted MaxSAT

Théophane Vallaeys, Marc Schachtsiek

April 2023

1 Introduction

Weighted Maximum Satisfiability (Weighted MaxSAT) is a combinatorial optimization problem that extends Maximum Satisfiability (MaxSAT) by assigning non-negative weights to clauses in a CNF formula. The goal is to find an assignment of truth values to variables that maximizes the sum of weights of satisfied clauses, while minimizing the number of unsatisfied clauses. MaxSAT is NP-hard and has applications in various areas such as artificial intelligence, operations research, and scheduling. Previous work has shown the success in applying Monte Carlo Tree Search (MCTS) and other related methods to MaxSAT and achieving robust performance against a variety of scenarios.

In this work, we partially replicate the results of [3] and extend the functionality to the Weighted MaxSAT problem. We show that the performance of the weighted adaptation performs as good as the unweighted version on an equally weighted problem and achieves robust performance on a variety of Weighted MaxSAT benchmarks. The implementation covers the Monte Carlo Tree Search (MCTS) and its nested variant, as well as sequential halving. Several heuristics and flipping strategies were implemented for both MaxSAT and Weighted MaxSAT.

The code is available at `https://github.com/webalorn/mc_maxsat`. The MaxSAT implementation is in the *main* branch and the adaptation to Weighted MaxSAT in the *weighted* branch. The data was taken from Max-SAT 2015 (`http://www.maxsat.udl.cat/15/benchmarks/`).

2 Algorithms & Heuristics

The following algorithms were implemented:

- Monte Carlo Tree Search (MCTS) is an algorithm that selects nodes in a search tree based on their estimated value and exploration potential. The quality of each node is evaluated by using rollouts and their statistics are analyzed to select nodes for further exploration.

- Zero Nested Monte Carlo Search (ZNMCS) is an extension of NMCS that only considers the best action from a pool of actions instead of the entire set of possible actions. It is effectively a limit on the search space covered during each iteration and was chosen here since it significantly reduces the computational complexity while only having slightly worse performance. The number of actionable variables was limited to 4.

- Sequential Halving (SH) is a round-based algorithm where each arm is sampled the same amount of times and at each round the worse half of arms is removed. In the next round, the remaining arms divide the same budget amongst a smaller number with the aim to provide more robust estimation of their rewards leading to a progressively better convergence towards the best solution.

Additionally, the implementation included multiple heuristics (with H3 being used for the experiments) and two flipping strategies that were adapted for Weighted MaxSAT.

Heuristics

All heuristics set to true the variables which have more positive than negated occurrences inside clauses, and false otherwise. They differ in the order of assigning.

- H1 assigns the values in order of the variables.

- H2 assigns the values starting with the variable that occurs most frequently.

- H3 assigns the values starting with the literal that occurs most frequently.

We also experiment with a *dynamic* version of the heuristic, where the value of a variable was assigned by taking into account only *unsatisfied* clauses. But, surprisingly, it only hindered the results, while being slower to compute.

Flipping Algorithms

- WalkSAT [1] classically uses a random initial assignment and then attempts to flip unverified variables to solve the satisfiability problem. Applied in the scope of Monte Carlo methods, it usually starts with the current best solution and performs the flipping within some budget. Variables are flipped at random a certain percentage of times, but otherwise the a breaking score based on a clause count and selecting the best variable from that list.

- Novelty [2] is similar to WalkSAT but considers all variables instead of only unverified ones with a guard to prevent flipping in a loop.

- MaxWalkSAT is the weighted adaptation of the aforementioned WalkSAT algorithm. Instead of using the count of of clauses as the breaking score, the weight considered.

- Weighted-Novelty is a similar adaptation of Novelty.

3 Results

For the following results, the values are the average number of or the average sum of weights of the remaining unsatisfied clauses which was attempted to be minimized for each instance.

We observed that NMCS with a depth of 2 took significantly more time to execute than the other methods. While its results might be better, we were not even able to obtain them on every test set. For instance, on the unweighted case with 70 variables, NMCS with depth 2 took on average 629s to execute each instance, against 33.6s for SH, which is an order of magnitude more.

3.1 MaxSAT

Table 1: Unweighted MaxWalkSAT performance on unweighted instances

Benchm.		Unweighted MaxWalkSAT					Optimal Solution
Vars	Inst	MCTS	NMCS *d=1*	NMCS *d=2*	SH	SH (AMAF)	
70	50	48.6	49.3	**47.2**	48.6	48.7	46.8
80	50	27.9	28.3	**27.1**	27.8	27.7	26.9
90	49	40.2	40.4	—	**40.0**	40.2	—
120	50	**211.7**	214.2	—	212.9	213.2	196.1
140	50	**202.5**	205.7	—	203.6	203.8	184.8
200	49	**194.7**	195.9	—	194.9	195.3	171.0
300	25	8.16	8.7	—	**8.1**	8.2	6.3

3.2 Weighted MaxSAT

The first set of experiments was performed to verify the correct workings of the weighted adaptations. The four datasets listed here were created on the basis of the respective unweighted dataset by giving each clause an equal weight of 1, making the results directly comparable since the cumulative weight equals the count. As previously shown the approaches covered in this work perform generally well on the data and are robust to the different scenarios presented in Table 3.2. Unexpectedly, the weighted algorithms perform marginally but consistently better on the same data as the unweighted implementation with every other part having been kept the same.

Table 2: Weighted MaxWalkSAT performance on equal weighted instances

Benchm.		Weighted MaxWalkSAT					Optimal Solution
Vars	Inst	MCTS	NMCS *d=1*	NMCS *d=2*	SH	SH (AMAF)	
70	50	47.6	48.1	**46.9**	47.6	47.6	46.8
140	50	200.4	201.1	—	**200.3**	200.4	184.8
200	49	**190.1**	191.4	—	190.7	190.7	171.0
300	25	7.7	8.0	—	**7.5**	7.8	6.3

The next set of experiments were performed on weighted MaxSAT problems from Max-SAT 2015. Each of the clauses has an associated weight $w \in [1, 10]$ and the score is the cumulative weight of all remaining unverified clauses. Optimal information is available for some of the problem instances and gives a good indication on the overall performance of the algorithms. For Weighted MaxWalkSAT, Table 3, the performance is as expected, performing generally well across the board especially with NMCS (depth = 2) and Sequential Halving.

Table 3: Weighted MaxWalkSAT performance on weighted instances

Benchm.		Weighted MaxWalkSAT					Optimal Solution
Vars	Inst	MCTS	NMCS *d=1*	NMCS *d=2*	SH	SH (AMAF)	
70	45	270.2	274.9	**263.5**	267.7	267.8	254.4
80	80	223.6	226.7	**216.2**	220.5	220.1	—
90	49	177.3	180.1	**170.0**	175.7	175.0	166.6
100	60	134.1	136.7	**127.9**	131.2	132.1	—
110	50	97.2	100.6	**93.7**	96.4	96.0	91.4

Finally, the aforementioned algorithms were run with the Weighted Novelty flipping algorithm instead of Weighted MaxWalkSAT. The results are shown in Table 4 and show that the performance is worse than Weighted MaxWalkSAT accross the board while taking between 1.5x and 3x more time to compute.

Table 4: Weighted Novelty performance on weighted instances

Benchm.		Weighted Novelty					Optimal Solution
Vars	Inst	MCTS	NMCS *d=1*	NMCS *d=2*	SH	SH (AMAF)	
70	45	290.6	**284.0**	269.7	287.2	284.2	254.4
80	80	242.3	240.3	—	**232.0**	234.3	—
90	49	189.1	192.1	—	**184.0**	184.1	166.6
100	60	145.9	154.0	—	**141.6**	143.1	—
110	50	114.9	125.6	—	111.9	**111.8**	91.4

4 Conclusion

This paper explored the extension of Monte Carlo methods from MaxSAT problems to Weighted MaxSAT. Our results show that the robustness of Monte Carlo methods for MaxSAT problems extend to the weighted domain and furthermore seems to even marginally improve the results for equally weighted problem instances that are direct solutions to their unweighted counterparts. It is clear that ZNMCS with a depth of 2 is the best performing algorithm here; however, another method might be preferred in scenarios were reasonable runtimes are required.

We showed in our results that, **in the weighted case**, using **Successive Halving** (with or without AMAF) was most of the time the best performing method that executes less that a few minutes (therefore excluding NMCS with a depth of 2).

There are several aspects that could improve this work. First of all, WalkSAT and Novelty and their weighted adaptations are all a similar approach to SAT solving. There are other methods such as CCLS that might lead to better results here. Additionally, while the flipping algorithms were adapted for the weighted domain, the heuristics not. Perhaps improved heuristics specifically designed for Weighted MaxSAT problems might lead to further improvement.

References

[1] Kautz, H., Selman, B., and McAllester, D. Walksat in the 2004 sat competition. In *Proceedings of the International Conference on Theory and Applications of Satisfiability Testing* (2004).

[2] Menai, M. E. B., and Batouche, M. C. Efficient initial solution to extremal optimization algorithm for weighted maxsat problem. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (2003).

[3] Wang, H., Saffidine, A., and Cazenave, T. Towards tackling maxsat by combining nested monte carlo with local search. *ArXiv abs/2302.13225* (2023).