Open-WBO @ MaxSAT Evaluation 2019

Ruben Martins
rubenm@cs.cmu.edu
CMU, USA

Norbert Manthey
nmanthey@conp-solutions.com
Dresden, Germany

Miguel Terra-Neves, Vasco Manquinho, Inês Lynce
{neves,vmm,ines}@inesc-id.pt
INESC-ID/IST, Portugal

I. INTRODUCTION

Open-WBO [1] is an open source MaxSAT solver that supports several MaxSAT algorithms [2], [3], [4], [5], [6], [7], [8] and SAT solvers [9], [10], [11]. Open-WBO is particularly efficient for unweighted MaxSAT and has been one of the best solvers in the MaxSAT Evaluations from 2014 to 2017. Three versions of Open-WBO were submitted to the MaxSAT Evaluation 2019: open-wbo-g, open-wbo-ms and open-wbo-ms-pre. The remainder of this document describes the differences between these versions.

II. SAT SOLVERS

OPEN-WBO is based on the data structures of MINISAT 2.2 [9], [12]. Therefore, solvers based on MINISAT 2.2 can be used as a potential back-end solvers. For the MaxSAT Evaluation 2019, we use GLUCOSE 4.1 4.1 [10], [13], [14] as the back-end SAT solver of the open-wbo-g version and MERGESAT [11] as the back-end SAT solver of the versions open-wbo-ms and open-wbo-ms-pre.

MERGESAT is a new CDCL solver developed by Norbert Manthey and it is based on the SAT competition winner of 2018, MAPLELCMDISTCHRONOBT [15], and adds several known techniques. For restarts, only partial backtracking is used, learned clause minimization is implemented more efficiently, and also applies simplification again in case the first swipe resulted in a simplification. Finally, the time-based decision heuristic switch is made deterministic by using solving steps. To support being used inside MaxSAT solvers, the incremental search feature had to been enabled again.

III. MAXSAT ALGORITHMS

In this section we briefly describe the algorithms used for the complete and incomplete tracks at the MSE2019.

A. Complete Track

For the complete track, OPEN-WBO uses a variant of the unsatisfiability-based algorithm MSU3 [3] and the OLL algorithm [7]. These algorithms work by iteratively refining a lower bound λ on the number of unsatisfied soft clauses until an optimum solution is found. Both MSU3 and OLL use the Totalizer encoding for incremental MaxSAT solving [4]. For unweighted MaxSAT, we extended the incremental MSU3 algorithm [4] with resolution-based partitioning techniques [8]. We represent a MaxSAT formula using a resolution-based graph representation and iteratively join partitions by using a proximity measure extracted from the graph representation of the formula. The algorithm ends when only one partition remains and the optimal solution is found. Since the partitioning of some MaxSAT formulas may be unfeasible or not significant, we heuristically choose to run either MSU3 with partitions or the OLL algorithm. In particular, we do not use partition-based techniques when one of the following criteria is met: (i) the formula is too large ($>$ 1,000,000 clauses), (ii) the ratio between the number of partitions and soft clauses is too high ($>$ 0.8), (iii) the sparsity of the graph is too small ($<$ 0.04), or (iv) there exist some at-most-one relations between soft clauses ($>$ 10), i.e. if one soft clause is satisfied it implies that some other soft clauses will be unsatisfied.

For weighted MaxSAT, we use a variant of the OLL algorithm [7] without optimizations. Potential avenues for improvements involve reusing the soft cardinality via assumptions instead of cloning them [16], extending the OLL algorithm to use lexicographic optimization [17], and perform core minimization.

B. Incomplete Track

For the unweighted incomplete track, OPEN-WBO uses an incremental variant of the MSU4 algorithm [18], [19] with the incremental Totalizer encoding [4]. This is a complete MaxSAT algorithm that performs a linear search SAT-UNSAT but lazily expands the soft clauses that can be relaxed, i.e. unsatisfied. This approach is particularly effective for benchmarks with thousands of soft clauses [19].

For the weighted complete track, OPEN-WBO uses a variant of the lexicographical optimization algorithm [17] that does not guarantee optimality [20], [21]. This algorithm considers n objective functions where n is the number of distinct weights in the MaxSAT formula. This is done by performing a sequence of calls to a SAT solver and refining an upper bound μ on the number of unsatisfied soft clauses. To restrict μ at each iteration, we need to encode cardinality constraints into CNF, for which, incremental Totalizer encoding [4] has been used. Once for a given objective function the upper bound μ cannot be improved, it is frozen, and the next objective function in the order is optimized. If an optimal solution is found when using this algorithm, then it is not necessarily an optimal solution of the input formula. Once this happens, we change to a complete algorithm based on linear search SAT-UNSAT that uses the Adder [22] or Generalized Totalizer encoding (GTE) [23] to encode pseudo-Boolean constraints. In order to maintain the lexicographic structure as long as possible, we only relax the previous lexicographical restrictions if they are the reason for unsatisfiability.

IV. PREPROCESSING

We integrated the MaxSAT preprocessor MaxPre [24] with Open-WBO via MaxPre API into the version open-wbo-ms-pre. To avoid spending too much time in preprocessing, we limit the number of tries for each preprocessing technique with the flag `-skiptechnique=100` and the time limit taken by the preprocessor to 10% of the total time (or 180 seconds if smaller). MaxPre can simplify the formula using a variety of techniques, such as, blocked clause elimination, unit propagation, bounded variable elimination, subsumption elimination, self subsuming resolution, subsumed label elimination, binary core removal, bounded variable addition, group subsumed label elimination, equivalence elimination, unhiding techniques, structure labeling and failed label probing.

In addition to the simplifications performed by MaxPre, we also perform identification of unit cores and at-most-one relations between soft clauses by using unit propagation. A similar technique is done in RC2 [25], the winner of the MaxSAT Evaluation 2018.

V. AVAILABILITY

The latest release of Open-WBO is available under a MIT license in GitHub at *https://github.com/sat-group/open-wbo*.

ACKNOWLEDGMENTS

We would like to thank Laurent Simon and Gilles Audemard for allowing us to use GLUCOSE 4.1 in the MaxSAT Evaluation. We would also like to thank Niklas Eén and Niklas Sörensson for the development of MINISAT 2.2. We would like to thank Jeremias Berg and Matti Järvisalo for their help with the MaxSAT preprocessor MaxPre. Additionally, we would like to thank all the collaborators on previous versions of OPEN-WBO, namely Saurabh Joshi and Mikoláš Janota. Finally, we would like to thank David Chen for his study on the impact of disjoint cores, unit cores, and at-most-one relations between soft clauses that were done in the scope of Independent Studies at CMU.

REFERENCES

[1] R. Martins, V. Manquinho, and I. Lynce, "Open-WBO: a Modular MaxSAT Solver," in *SAT*, ser. LNCS, vol. 8561. Springer, 2014, pp. 438–445.

[2] V. Manquinho, J. Marques-Silva, and J. Planes, "Algorithms for Weighted Boolean Optimization," in *SAT*. Springer, 2009, pp. 495–508.

[3] J. Marques-Silva and J. Planes, "On Using Unsatisfiability for Solving Maximum Satisfiability," *CoRR*, 2007.

[4] R. Martins, S. Joshi, V. Manquinho, and I. Lynce, "Incremental Cardinality Constraints for MaxSAT," in *CP*. Springer, 2014, pp. 531–548.

[5] R. Martins, V. Manquinho, and I. Lynce, "On Partitioning for Maximum Satisfiability," in *ECAI*. IOS Press, 2012, pp. 913–914.

[6] R. Martins, V. M. Manquinho, and I. Lynce, "Community-based partitioning for maxsat solving," in *SAT*. Springer, 2013, pp. 182–191.

[7] A. Morgado, C. Dodaro, and J. Marques-Silva, "Core-Guided MaxSAT with Soft Cardinality Constraints," in *CP*. Springer, 2014, pp. 564–573.

[8] M. Neves, R. Martins, M. Janota, I. Lynce, and V. M. Manquinho, "Exploiting Resolution-Based Representations for MaxSAT Solving," in *SAT*. Springer, 2015, pp. 272–286.

[9] N. Eén and N. Sörensson, "An Extensible SAT-solver," in *SAT*. Springer, 2003, pp. 502–518.

[10] G. Audemard and L. Simon, "Predicting Learnt Clauses Quality in Modern SAT Solvers," in *IJCAI*, 2009, pp. 399–404.

[11] N. Manthey, "Mergesat," in *Proceedings of SAT Competition 2019: Solver and Benchmark Descriptions*, 2019.

[12] N. Sörensson, N. Een, and N. Manthey. (2018, May) GitHub repository for MiniSat. https://github.com/conp-solutions/minisat.

[13] G. Audemard, J.-M. Lagniez, and L. Simon, "Improving glucose for incremental sat solving with assumptions: Application to mus extraction," in *SAT*. Springer, 2013.

[14] G. Audemard and L. Simon. (2018, May) Glucose's home page. http://www.labri.fr/perso/lsimon/glucose.

[15] A. Nadel and V. Ryvchin, "Chronological backtracking," in *SAT*. Springer, 2018, pp. 111–121.

[16] J. Berg and M. Järvisalo, "Weight-Aware Core Extraction in SAT-Based MaxSAT Solving," in *CP*. Springer, 2017, pp. 652–670.

[17] J. Marques-Silva, J. Argelich, A. Graça, and I. Lynce, "Boolean lexicographic optimization: algorithms & applications," *Annals of Mathematics and Artificial Intelligence*, vol. 62, no. 3-4, pp. 317–343, 2011.

[18] J. Marques-Silva and J. Planes, "Algorithms for maximum satisfiability using unsatisfiable cores," in *DATE*. ACM, 2008, pp. 408–413.

[19] R. Martins, V. Manquinho, and I. Lynce, "Improving linear search algorithms with model-based approaches for maxsat solving," *J. Exp. Theor. Artif. Intell.*, vol. 27, no. 5, pp. 673–701, 2015. [Online]. Available: https://doi.org/10.1080/0952813X.2014.993508

[20] S. Joshi, P. Kumar, R. Martins, and S. Rao, "Approximation Strategies for Incomplete MaxSAT," in *CP*. Springer, 2018.

[21] S. Joshi, P. Kumar, S. Rao, and R. Martins, "Open-WBO-Inc: Approximation Strategies for Incomplete Weighted MaxSAT," in *JSAT*. IOS Press, 2019.

[22] J. P. Warners, "A Linear-Time Transformation of Linear Inequalities into Conjunctive Normal Form," *Information Processing Letters*, vol. 68, no. 2, pp. 63–69, 1998.

[23] S. Joshi, R. Martins, and V. M. Manquinho, "Generalized Totalizer Encoding for Pseudo-Boolean Constraints," in *CP*. Springer, 2015, pp. 200–209.

[24] T. Korhonen, J. Berg, P. Saikko, and M. Järvisalo, "MaxPre: An Extended MaxSAT Preprocessor," in *SAT*. Springer, 2017, pp. 449–456.

[25] A. Ignatiev, A. Morgado, and J. Marques-Silva, "PySAT: A Python Toolkit for Prototyping with SAT Oracles," in *Proc. SAT*, ser. Lecture Notes in Computer Science, O. Beyersdorff and C. M. Wintersteiger, Eds., vol. 10929. Springer, 2018, pp. 428–437.