Exact: evaluating a pseudo-Boolean solver on MaxSAT problems

Jo Devriendt
KU Leuven
Leuven, Belgium
jo.devriendt@kuleuven.be

***Abstract*—Weighted MaxSAT solving is a special case of pseudo-Boolean optimization, also known as binary linear programming. This submission aims to investigate whether Exact, a conflict-driven cutting planes learning pseudo-Boolean solver, is competitive on MaxSAT problems.**

***Index Terms*—binary linear programming, pseudo-Boolean solving, cutting planes, core-guided optimization**

I. Introduction

It is well-known[1] that a weighted MaxSAT formula can be written as a binary linear program (BLP):

$$\begin{aligned} \text{Minimize} \quad & \sum_{c \in C} w_c z_c \\ \text{s.t.} \quad & z_c + \sum_{x \in c^+} x + \sum_{y \in c^-} (1-y) \geqslant 1 \qquad \forall c \in C \end{aligned}$$

where C is a set of clauses, c^+ and c^- are the set of positive and negative literals in a clause $c \in C$ respectively, w_c is the cost of not satisfying c, and all variables x, y and z are binary.

Even though this BLP formulation is natural, the state-of-the-art in previous MaxSAT evaluations employs repeated calls to Boolean satisfiability (SAT) solvers instead of one straightforward call to an integer linear programming (ILP) solver. Most likely, the reason for this is that ILP solvers rely heavily on exploiting the linear relaxation of a BLP, while all constraints in the above BLP are clauses, which have a particularly weak linear relaxation.

A third technology that could natively handle the above BLP however is pseudo-Boolean (PB) solving. Similar to ILP technology, PB technology natively takes linear constraints over binary variables as input. However, in contrast to ILP solvers, a PB solver does not chiefly depend on reasoning on the linear relaxation of a BLP. Instead, so-called *conflict-driven cutting-planes learning* (CDCPL) PB solvers derive (*learn*) from each conflict in the search tree an implied linear constraint that, if it had been derived previously, would have prevented the conflict through unit propagation. In this way, CDCPL PB solvers are a generalization of *conflict-driven clause learning* (CDCL) SAT solvers, where a CDCPL solver can learn stronger constraints than clauses.

II. Submission

We submit the CDCPL solver Exact[2] to the MaxSAT evaluation. Exact is a fork of the CDCPL solver RoundingSat[3] [1]. For this submission, we do not employ RoundingSat's linear programming integration [2], as we expect the linear relaxations of the instances to be too weak. We do make use of its optimized propagation routines [3] and its hybrid core-guided optimization technique [4].

Exact improves upon its predecessor through a myriad of refactorings, extensions and improvements. We highlight three important ones for this MaxSAT evaluation submission.

A first one is the *stratification* routine of Exact's core-guided optimization. Instead of core-guided stratification based on [5], Exact uses a simple routine that ignores all soft clauses with a cost lower than some τ, which initially is set to the highest clause cost (the highest weight in the objective of the BLP representation). If Exact does not find a core with this τ (i.e., it finds a solution where all hard and non-ignored soft clauses are satisfied, or timeouts in the core-guided search) τ is halved, to consider more soft clauses. This process is repeated until the maximum cost is halved to 1, at which point all soft clauses are taken into account.

A second improvement is the exploitation of the observation that a single *PB core* may yield multiple *cardinality cores*, which can be used during the core-guided lower bound derivation and objective reformulation process [4]. For instance, given an objective function $4x+3y+2z+w$ to be minimized, and a PB core $2x + 2y + z + w \geqslant 4$, Exact constructs an initial implied cardinality core $x + y + z \geqslant 2$, reformulating the objective to $2x + y + w + 2a + 4$ through the *extension constraint* $x + y + z = 2 + a$. But as $2x + 2y + z + w \geqslant 4$ also implies $x + y + w \geqslant 2$, Exact can further reformulate the objective to $x + 2a + b + 6$ with the extension constraint $x+y+w = 2+b$, increasing the objective lower bound from 4 to 6 without any new core-guided solver call.

A third improvement is meant to address the fact that, given a search conflict implied by only clausal constraints, CDCPL solvers can only learn a clause, which is identical to regular CDCL SAT solving (which has a more efficient implementation). For CDCPL to work well, non-clausal constraints need to appear in the conflict implication graph, so that

[1]See, e.g., https://en.wikipedia.org/wiki/Maximum_satisfiability_problem#(1-1/e)-approximation

[2]https://gitlab.com/JoD/exact

[3]https://gitlab.com/miao_research/roundingsat

strong non-clausal constraints can be learned [6]. On MaxSAT instances, Exact introduces non-clausal constraints in three ways. Firstly, a derived upper or lower bound on the objective function is typically a non-clausal constraint. Secondly, a core-guided extension constraint also typically is equivalent to a conjunction of non-clausal constraints. Thirdly, implied cardinality constraints can be detected from a conjunction of clauses. Work on cardinality detection in RoundingSat exists [7], where an investigation of the implication graph during conflict analysis yields the right information to construct cardinality constraints. Exact uses a different approach, where repeated *probing* (deciding a single variable and running unit propagation) yields the necessary edges in the implication graph to derive *at-most-one* cardinality constraints.

III. Conclusion

By combining the effectiveness of CDCLP and core-guided optimization, PB solving technology may have become competitive to SAT-based approaches on MaxSAT problems. Exact's submission to 2021's MaxSAT evaluation will provide experimental data to support or reject this hypothesis.

IV. Acknowledgment

Exact is a fork of RoundingSat, which in turn uses code from MiniSat [8]. The author of Exact is Jo Devriendt (KU Leuven). The authors of RoundingSat are Jan Elffers (formerly Lund University), Jo Devriendt (KU Leuven), Stephan Gocht (Lund University) and Jakob Nordström (University of Copenhagen, Lund University). The authors of MiniSat are Niklas Eén (formerly University of California) and Niklas Sörensson (formerly Chalmers University of Technology).

References

[1] J. Elffers and J. Nordström, "Divide and conquer: Towards faster pseudo-Boolean solving," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI '18)*, Jul. 2018, pp. 1291–1299.

[2] J. Devriendt, A. Gleixner, and J. Nordström, "Learn to relax: Integrating 0-1 integer linear programming with pseudo-Boolean conflict-driven search," in *Proceedings of the 17th International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR '20)*, ser. Lecture Notes in Computer Science, vol. 12296. Springer, Sep. 2020, pp. xxiv–xxv.

[3] J. Devriendt, "Watched propagation of 0-1 integer linear constraints," in *Proceedings of the 26th International Conference on Principles and Practice of Constraint Programming (CP '20)*, ser. Lecture Notes in Computer Science, vol. 12333. Springer, Sep. 2020, pp. 160–176.

[4] J. Devriendt, S. Gocht, E. Demirović, J. Nordström, and P. Stuckey, "Cutting to the core of pseudo-Boolean optimization: Combining core-guided search with cutting planes reasoning," vol. 35, no. 5, May 2021, pp. 3750–3758. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/16492

[5] C. Ansótegui, M. L. Bonet, J. Gabàs, and J. Levy, "Improving SAT-based weighted MaxSAT solvers," in *Proceedings of the 18th International Conference on Principles and Practice of Constraint Programming (CP '12)*, ser. Lecture Notes in Computer Science, vol. 7514. Springer, Oct. 2012, pp. 86–101.

[6] S. R. Buss and J. Nordström, "Proof complexity and SAT solving," in *Handbook of Satisfiability*, 2nd ed., ser. Frontiers in Artificial Intelligence and Applications, A. Biere, M. J. H. Heule, H. van Maaren, and T. Walsh, Eds. IOS Press, Feb. 2021, vol. 336, ch. 7, pp. 233–350.

[7] J. Elffers and J. Nordström, "A cardinal improvement to pseudo-Boolean solving," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI '20)*, Feb. 2020, pp. 1495–1503.

[8] N. Eén and N. Sörensson, "An extensible SAT-solver," in *6th International Conference on Theory and Applications of Satisfiability Testing (SAT '03), Selected Revised Papers*, ser. Lecture Notes in Computer Science, vol. 2919. Springer, 2004, pp. 502–518.