Exact at the Pseudo-Boolean Competition 2025

Jo Devriendt
Nonfiction Software
Koksijde, Belgium
jo.devriendt@nonfictionsoftware.com
Orestis Lomis
KU Leuven
Leuven, Belgium
orestis.lomis@kuleuven.be

***Abstract*—EXACT is a cutting-planes learning integer programming solver built upon the foundations of CDCL-CUTTINGPLANES and ROUNDINGSAT. This document summarizes the main features of the version of EXACT submitted to the Pseudo-Boolean Competition 2025.**

I. INTRODUCTION

EXACT is a cutting-planes learning integer programming solver. It supports integer linear constraints, integer multiplicative constraints and reifications of integer linear constraints as input. EXACT eagerly translates these constraints during parsing to 0-1 linear inequalities, also known as *pseudo-Boolean* (PB) constraints. As such, the core search routines deal only with Boolean variables and linear constraints, which allows a tight conflict-driven cutting-planes learning (CDCPL) depth-first search loop based on the division method introduced in ROUNDINGSAT [1]. In fact, EXACT is a fork of ROUNDINGSAT, which participated in the last PB competition in 2016 under the name of CDCL-CUTTINGPLANES.

Complementary to its CDCPL core, EXACT features

- simplex LP solving integration [2] with SOPLEX[1] as backend solver;
- watched unit propagation [3];
- hybrid core-guided optimization [4], dynamically interleaving top-down and bottom-up optimization;
- arbitrary-sized coefficients, transparently switching to the most efficient internal representation for individual constraints;
- support for the VERIPB [5] proof format, to log certificates of unsatisfiability;
- advanced reduction and conflict analysis techniques for cutting-plane conflict analysis [6];
- in-processing using probing, dominance breaking, binary implication analysis and cardinality detection;
- support for integer variables and non-linear constraints, i.a., parsing .lp and .mip formats;
- a fully stateful Python interface with support for assumptions, unsat core generation, objective function modification, solution counting and solution intersection (*full propagation*);
- a branch[2] with full support for VERIPB's proof format version 2.

II. SUBMISSIONS

Four versions of EXACT were submitted to the following tracks:

- *Exact*: DEC-LIN, DEC-NLC, OPT-LIN, OPT-NLC, PARTIAL-LIN, SOFT-LIN
- *ExactNoDomBreak*: DEC-LIN, DEC-NLC, OPT-LIN, OPT-NLC, PARTIAL-LIN, SOFT-LIN
- *ExactNoDBNoLS*: DEC-LIN, DEC-NLC, OPT-LIN, OPT-NLC, PARTIAL-LIN, SOFT-LIN
- *Exact_proof*: DEC-LIN-CERT, OPT-LIN-CERT

The regular *Exact*, *ExactNoDomBreak* and *ExactNoDBNoLS* submissions use the same commit [3] while the proof-generating submission *Exact_proof* uses a slightly different one [4]. *Exact_proof* disables dominance breaking inprocessing and replaces core-guided optimization by a simpler lower bound assumption routine, to simplify the proof generation implementation.

ExactNoDomBreak is identical to the *Exact* submission, except that it disables dominance breaking (*ExactNoDomBreak*) as we observed this may not always be beneficial.

The biggest change to this year's submission is the incorporation of a recent integer programming local search routine [7]. In the future, Exact may incorporate this routine running in a seperate thread, but for the competition, this routine was run as a preprocessing step for 360 seconds (a tenth of the timeout). To gauge the effectiveness of the local search routine, the submission *ExactNoDBNoLS* does not use it. So comparing *ExactNoDBNoLS* and *ExactNoDomBreak* hence indicates how much improvement local search adds.

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.

[1] https://github.com/scipopt/soplex

[2] https://gitlab.com/nonfiction-software/exact/-/tree/veripb_2?ref_type=heads

[3] https://gitlab.com/nonfiction-software/exact/-/commit/ef45f5cb3382e45acbc743e8a594b26b68795eaf

[4] https://gitlab.com/nonfiction-software/exact/-/commit/8590e14c77b1930a92e3b6e8fe367e46bcb025ea

[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] J. Elffers, S. Gocht, C. McCreesh, and J. Nordström, "Justifying all differences using pseudo-Boolean reasoning," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI '20)*, Feb. 2020, pp. 1486–1494.

[6] O. Lomis, J. Devriendt, H. Bierlee, and T. Guns, "Improving Reduction Techniques in Pseudo-Boolean Conflict Analysis," in *28th International Conference on Theory and Applications of Satisfiability Testing (SAT 2025)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), J. Berg and J. Nordström, Eds., vol. 341. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025, pp. 21:1–21:17. [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SAT.2025.21

[7] P. Lin, M. Zou, and S. Cai, "An Efficient Local Search Solver for Mixed Integer Programming," in *30th International Conference on Principles and Practice of Constraint Programming (CP 2024)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), P. Shaw, Ed., vol. 307. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, pp. 19:1–19:19. [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.CP.2024.19