QMaxSATpb: A Certified MaxSAT-Solver

Dieter Vandesande, Wolf De Wulf and Bart Bogaerts

Vrije Universiteit Brussel

16th International Conference on
Logic Programming and Non-monotonic Reasoning

ARTIFICIAL
INTELLIGENCE
RESEARCH GROUP

Thanks to Jakob Nordström for sharing his proof logging slides.

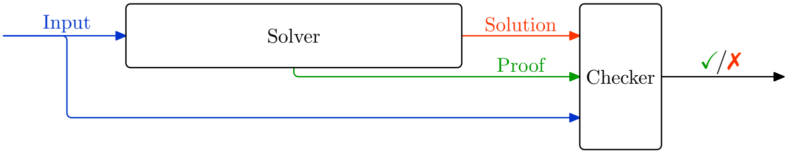
COMBINATORIAL SOLVING AND OPTIMISATION

- Revolution last couple of decades in combinatorial solvers for
  - Boolean satisfiability (SAT) solving [BHvMW21]
  - Satisfiability modulo theories (SMT) solving [BHvMW21]
  - Constraint programming (CP) [RvBW06]
  - Mixed integer linear programming (MIP) [AW13, BR07]
  - Answer Set Programming (ASP) [GKKS12]
- Solve NP problems (or worse) very successfully in practice!
- Except solvers are sometimes wrong... [BLB10, CKSW13, AGJ+18, GSD19, GS19]
- Software testing doesn't suffice to resolve this problem
- Formal verification techniques cannot deal with level of complexity of modern solvers

CERTIFIED RESULTS WITH PROOF LOGGING

Design certifying algorithms [ABM+11, MMNS11] that

- not only solve problem but also
- do proof logging to certify that solution is correct



Workflow:

1. Run solver on problem input
2. Get as output not only solution but also proof
3. Feed input + solution + proof to proof checker
4. Verify that proof checker says solution is correct

YET ANOTHER SAT SUCCESS STORY

Well established — required in main track of SAT competitions

Many proof logging formats for SAT solving using CNF clausal format:

- *DRAT* [HHW13a, HHW13b, WHH14]
- *GRIT* [CMS17]
- *LRAT* [CHH+17]
- . . .

Formally verified proof verifiers exist.

But efficient proof logging has remained out of reach for other paradigms,
e.g. Maximum Satisfiability (MaxSAT)

OUTLINE OF THIS PRESENTATION

The rest of this presentation:

- MaxSAT solver *QMaxSAT* [KZFH12]
- *VeriPB* [BGMN22, EGMN20b] as proof system.
- Our contribution: *QMaxSATpb*, A certified MaxSAT solver, by example
- Experimental results
- Future work & Conclusions

PARTIAL MAXSAT

A partial MaxSAT-instance is a tuple (F, S) with:

- F the set of hard clauses.
- S the set of soft clauses.

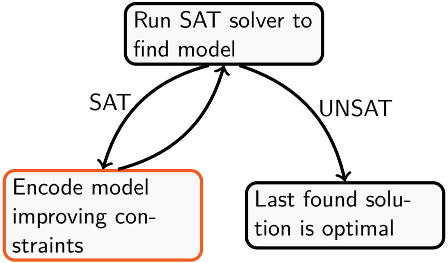
A solution is an assignment for all variables such that:

- All hard clauses are satisfied.
- No other satisfying assignment satisfies more soft clauses.

QMaxSAT: IDEA BEHIND THE SOLVER

QMaxSAT [KZFH12] is an Iterative Satisfiability-Based MaxSAT solver.

- Given a satisfying assignment, QMaxSAT searches for another assignment with fewer soft clauses falsified.
- Totalizer encoding of cardinality constraints [BB03]



VeriPB: A GENERAL PURPOSE PROOF SYSTEM

VeriPB is a proof system for pseudo-Boolean optimisation [BGMN22, EGMN20b].

It reasons on 0–1 integer linear inequalities $\sum_i a_i l_i \geq A$ (a.k.a. pseudo-Boolean constraints) with:

- Cutting Planes (CP) proof system [CCT87]
  - e.g., adding up two constraints
- Reverse Unit Propagation [GN03]
  - allows deriving constraints without providing an explicit derivation
- Redundance-Based Strenghtening [GN21, BGMN22]
  - generalisation of the RAT-rule [BT19]
  - allows introducing "fresh" reification variables, such as $r \Leftrightarrow (\sum_i a_i l_i \geq A)$.
- Support for Optimisation [BGMN22]
  - allows deriving model-improving constraints

QMaxSATpb: AN EXAMPLE

Hard Clauses	Soft Clauses
$\overline{x}_1 \vee x_2$	$\overline{x}_1 \vee \overline{x}_2 \vee r_1$
$x_1 \vee \overline{x}_2$	$x_1 \vee x_2 \vee r_2$
$\overline{x}_2 \vee x_3$	$x_2 \vee x_4 \vee r_3$
$\overline{x}_3 \vee x_4$	

- Relaxation variables r_i such that C_i falsified implies r_i true.
- We want to minimize $\sum_i r_i$

QMaxSATpb BY EXAMPLE

Objective: *min* $\sum_i r_i$

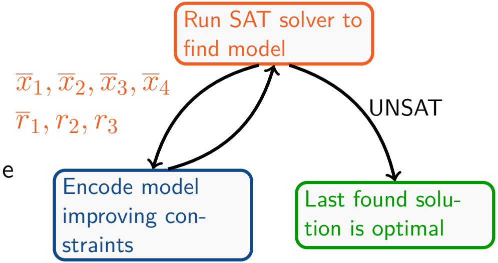
VeriPB proof:

derived	justification
$x_2 + r_2 \geq 1$	Reverse Unit Propagation
$\{\overline{x}_1, \ldots, \overline{x}_4, \overline{r}_1, r_2, r_3\}$	Incumbent solution
$\sum_i r_i \leq 1$	Objective Improvement Rule
$\mathrm{PB}(p_1 \Leftrightarrow (\sum_i r_i \geq 1))$	Fresh variable (RBS)
$\mathrm{PB}(p_2 \Leftrightarrow (\sum_i r_i \geq 2))$	
$\mathrm{CNF}(p_j \Leftrightarrow (\sum_i r_i \geq j))$	Explicit CP derivation
$\overline{p}_2 \geq 1$	Explicit CP derivation
$x_4 \geq 1$	Reverse Unit Propagation
$\{\overline{x}_1, \overline{x}_2, \overline{x}_3, x_4, \overline{r}_1, r_2, \overline{r}_3\}$	Incumbent solution
$\sum_i r_i \leq 0$	Objective Improvement Rule
$\overline{p}_1 \geq 1$	Explicit CP derivation
$0 \geq 1$	Reverse Unit Propagation

$\overline{x}_1 \vee x_2$	$\overline{x}_1 \vee \overline{x}_2 \vee r_1$
$x_1 \vee \overline{x}_2$	$x_1 \vee x_2 \vee r_2$
$\overline{x}_2 \vee x_3$	$x_2 \vee x_4 \vee r_3$
$\overline{x}_3 \vee x_4$	$x_2 \vee r_2$

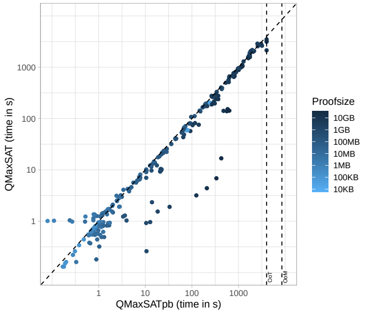
$\mathrm{CNF}(p_j \Leftrightarrow (\sum_i r_i \geq j))$

$\overline{p}_2$	x_4
$\overline{p}_1$	$\perp$

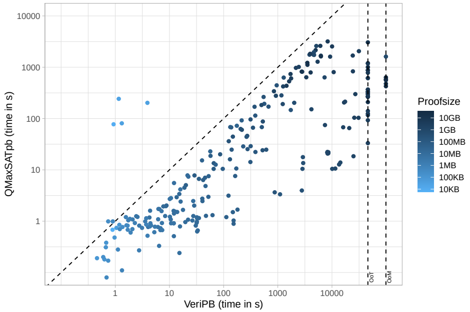


EXPERIMENTAL RESULTS

MaxSAT evaluations 2021
Resource Limits: *QMaxSAT* (1h, 32GB) — *VeriPB* (10h, 64GB)
10.2% OoT, 2.4% OoM



(a) Performance overhead of proof logging



(b) Performance of proof verification

FUTURE RESEARCH DIRECTIONS

Other MaxSAT Algorithms

- Iterative MaxSAT algorithms (e.g., Pacose [PRB18])
  - Uses different encodings of cardinality constraints
- Core-guided MaxSAT algorithms [Ave21, IMMS19, JA17, MTNJ+17]
  - Heavily rely on encodings of cardinality constraints
- Implicit hitting sets solvers [DB11]
  - Challenge: certifying the minimality of the hitting sets.

Proof logging for other combinatorial optimization

- Pseudo-Boolean optimization
- Mixed integer linear programming *(work on SCIP in [CGS17, EG21])*
- Satisfiability modulo theories (SMT) solving *(work by Bjørner and others)*
- Answer Set Programming *(ASP–DRUPE [ADF+19], but no (native) support for optimization or aggregates (PB constraints). ASP-VeriPB(?))*

SUMMING UP

Proof logging helps:

- Ensuring correctness of a result.
- Debugging in case of a bug.
- Building trust in solvers

VeriPB: general-purpose proof system

- Subgraph Isomorphism Problem [GMN20]
- Parity (XOR) reasoning [GN21]
- All Different reasoning (CP) [EGMN20a]

QMaxSATpb: Certified MaxSAT-solver

- Clauses derived by SAT oracle are RUP
- Totalizer encoding of cardinality constraints can be proven by an explicit CP derivation
- Proof logging is possible without too much overhead, verifying proofs is harder.

Thank you for your attention!

REFERENCES

[ABM+11] Eyad Alkassar, Sascha Böhme, Kurt Mehlhorn, Christine Rizkallah, and Pascal Schweitzer. An introduction to certifying algorithms. *it - Information Technology Methoden und innovative Anwendungen der Informatik und Informationstechnik*, 53(6):287–293, December 2011.

[ADF+19] Mario Alviano, Carmine Dodaro, Johannes Klaus Fichte, Markus Hecher, Tobias Philipp, and Jakob Rath. Inconsistency proofs for ASP: the ASP - DRUPE format. *Theory Pract. Log. Program.*, 19(5-6):891–907, 2019.

[AGJ+18] Özgür Akgün, Ian P. Gent, Christopher Jefferson, Ian Miguel, and Peter Nightingale. Metamorphic testing of constraint solvers. In *Proceedings of the 24th International Conference on Principles and Practice of Constraint Programming (CP '18)*, volume 11008 of *Lecture Notes in Computer Science*, pages 727–736. Springer, August 2018.

[Ave21] Florent Avellaneda. A short description of the solver EvalMaxSAT. *MaxSAT Evaluation Solver and Benchmark Descriptions*, pages 10–11, 2021.

[AW13] Tobias Achterberg and Roland Wunderling. Mixed integer programming: Analyzing 12 years of progress. In Michael Jünger and Gerhard Reinelt, editors, *Facets of Combinatorial Optimization*, pages 449–481. Springer, 2013.

REFERENCES

[BB03] Olivier Bailleux and Yacine Boufkhad. Efficient CNF Encoding of Boolean Cardinality Constraints. In Francesca Rossi, editor, *Principles and Practice of Constraint Programming CP 2003*, Lecture Notes in Computer Science, pages 108–122, Berlin, Heidelberg, 2003. Springer.

[BGMN22] Bart Bogaerts, Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. Certified symmetry and dominance breaking for combinatorial optimisation. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI '22)*, pages 3698–3707, February 2022.

[BHvMW21] Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, volume 336 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2nd edition, February 2021.

[BLB10] Robert Brummayer, Florian Lonsing, and Armin Biere. Automated testing and debugging of SAT and QBF solvers. In *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing (SAT '10)*, volume 6175 of *Lecture Notes in Computer Science*, pages 44–57. Springer, July 2010.

[BN21] Samuel R. Buss and Jakob Nordström. Proof complexity and SAT solving. In Biere et al. [BHvMW21], chapter 7, pages 233–350.

REFERENCES

[BR07] Robert Bixby and Edward Rothberg. Progress in computational mixed integer programming—A look back from the other side of the tipping point. *Annals of Operations Research*, 149(1):37–41, February 2007.

[BT19] Samuel R. Buss and Neil Thapen. DRAT proofs, propagation redundancy, and extended resolution. In *Proceedings of the 22nd International Conference on Theory and Applications of Satisfiability Testing (SAT '19)*, volume 11628 of *Lecture Notes in Computer Science*, pages 71–89. Springer, July 2019.

[CCT87] William Cook, Collette Rene Coullard, and György Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18(1):25–38, November 1987.

[CGS17] Kevin K. H. Cheung, Ambros M. Gleixner, and Daniel E. Steffy. Verifying integer programming results. In *Proceedings of the 19th International Conference on Integer Programming and Combinatorial Optimization (IPCO '17)*, volume 10328 of *Lecture Notes in Computer Science*, pages 148–160. Springer, June 2017.

[CHH+17] Luís Cruz-Filipe, Marijn J. H. Heule, Warren A. Hunt Jr., Matt Kaufmann, and Peter Schneider-Kamp. Efficient certified RAT verification. In *Proceedings of the 26th International Conference on Automated Deduction (CADE-26)*, volume 10395 of *Lecture Notes in Computer Science*, pages 220–236. Springer, August 2017.

REFERENCES

[CKSW13] William Cook, Thorsten Koch, Daniel E. Steffy, and Kati Wolter. A hybrid branch-and-bound approach for exact rational mixed-integer programming. *Mathematical Programming Computation*, 5(3):305–344, September 2013.

[CMS17] Luís Cruz-Filipe, João P. Marques-Silva, and Peter Schneider-Kamp. Efficient certified resolution proof checking. In *Proceedings of the 23rd International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS '17)*, volume 10205 of *Lecture Notes in Computer Science*, pages 118–135. Springer, April 2017.

[DB11] Jessica Davies and Fahiem Bacchus. Solving MAXSAT by solving a sequence of simpler SAT instances. In *Proceedings of the 17th International Conference on Principles and Practice of Constraint Programming (CP '11)*, volume 6876 of *Lecture Notes in Computer Science*, pages 225–239. Springer, September 2011.

[EG21] Leon Eifler and Ambros Gleixner. A computational status update for exact rational mixed integer programming. In *Proceedings of the 22nd International Conference on Integer Programming and Combinatorial Optimization (IPCO '21)*, volume 12707 of *Lecture Notes in Computer Science*, pages 163–177. Springer, May 2021.

REFERENCES

[EGMN20a] Jan Elffers, Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. Justifying all differences using pseudo-boolean reasoning. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 1486–1494, 2020.

[EGMN20b] Jan Elffers, Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. Justifying all differences using pseudo-Boolean reasoning. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI '20)*, pages 1486–1494, February 2020.

[GKKS12] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. *Answer Set Solving in Practice.* Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012.

[GMN20] Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. Subgraph isomorphism meets cutting planes: Solving with certified solutions. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI '20)*, pages 1134–1140, July 2020.

REFERENCES

[GN03] Evgueni Goldberg and Yakov Novikov. Verification of proofs of unsatisfiability for CNF formulas. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '03)*, pages 886–891, March 2003.

[GN21] Stephan Gocht and Jakob Nordström. Certifying parity reasoning efficiently using pseudo-Boolean proofs. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI '21)*, pages 3768–3777, February 2021.

[GS19] Graeme Gange and Peter Stuckey. Certifying optimality in constraint programming. Presentation at KTH Royal Institute of Technology. Slides available at `https://www.kth.se/polopoly_fs/1.879851.1550484700!/CertifiedCP.pdf`, February 2019.

[GSD19] Xavier Gillard, Pierre Schaus, and Yves Deville. SolverCheck: Declarative testing of constraints. In *Proceedings of the 25th International Conference on Principles and Practice of Constraint Programming (CP '19)*, volume 11802 of *Lecture Notes in Computer Science*, pages 565–582. Springer, October 2019.

[HHW13a] Marijn J. H. Heule, Warren A. Hunt Jr., and Nathan Wetzler. Trimming while checking clausal proofs. In *Proceedings of the 13th International Conference on Formal Methods in Computer-Aided Design (FMCAD '13)*, pages 181–188, October 2013.

REFERENCES

[HHW13b] Marijn J. H. Heule, Warren A. Hunt Jr., and Nathan Wetzler. Verifying refutations with extended resolution. In *Proceedings of the 24th International Conference on Automated Deduction (CADE-24)*, volume 7898 of *Lecture Notes in Computer Science*, pages 345–359. Springer, June 2013.

[IMMS19] Alexey Ignatiev, Antonio Morgado, and Joao Marques-Silva. RC2: an Efficient MaxSAT Solver. *Journal on Satisfiability, Boolean Modeling and Computation*, 11:53–64, 09 2019.

[JA17] Eivind Jahren and Roberto Asın Achá. The MSUSorting MaxSAT solver. *MaxSAT Evaluation Solver and Benchmark Descriptions*, page 15, 2017.

[KZFH12] Miyuki Koshimura, Tong Zhang, Hiroshi Fujita, and Ryuzo Hasegawa. QMaxSAT: A Partial Max-SAT Solver. *Journal on Satisfiability, Boolean Modeling and Computation*, 8(1-2):95–100, January 2012.

[MMNS11] Ross M. McConnell, Kurt Mehlhorn, Stefan Näher, and Pascal Schweitzer. Certifying algorithms. *Computer Science Review*, 5(2):119–161, May 2011.

[MTNJ+17] Ruben Martins, Miguel Terra-Neves, Saurabh Joshi, Mikoláš Janota, Vasco Manquinho, and Inês Lynce. Open-wbo in MaxSAT evaluation 2017. *MaxSAT Evaluation Solver and Benchmark Descriptions*, page 17, 2017.

REFERENCES

[PRB18] Tobias Paxian, Sven Reimer, and Bernd Becker. Pacose: An iterative sat-based maxsat solver. *MaxSAT Evaluation*, 2018:20, 2018.

[RvBW06] Francesca Rossi, Peter van Beek, and Toby Walsh, editors. *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*. Elsevier, 2006.

[WHH14] Nathan Wetzler, Marijn J. H. Heule, and Warren A. Hunt Jr. DRAT-trim: Efficient checking and trimming using expressive clausal proofs. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 422–429. Springer, July 2014.

0–1 INTEGER LINEAR (A.K.A. PSEUDO-BOOLEAN) CONSTRAINTS

Pseudo-Boolean (PB) constraints are 0–1 integer linear constraints

$$C \doteq \sum_i a_i \ell_i \geq A$$

- $a_i, A \in \mathbb{Z}$
- literals ℓ_i: x_i or $\overline{x}_i$ (where $x_i + \overline{x}_i = 1$)
- variables x_i take values $0 = \mathit{false}$ or $1 = \mathit{true}$

Pseudo-Boolean formulas are conjunctions of pseudo-Boolean constraints

A pseudo-Boolean optimisation problem is a formula F with a linear objective function.

SOME TYPES OF PSEUDO-BOOLEAN CONSTRAINTS

1. Clauses

$$x \vee \overline{y} \vee z \quad \Leftrightarrow \quad x + \overline{y} + z \geq 1$$

2. Cardinality constraints

$$x_1 + x_2 + x_3 + x_4 \geq 2$$

3. General pseudo-Boolean constraints

$$x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$$

PSEUDO-BOOLEAN REASONING: CUTTING PLANES [CCT87]

Literal axioms $$\frac{}{\ell_i \geq 0}$$

Linear combination $$\frac{\sum_i a_i \ell_i \geq A \qquad \sum_i b_i \ell_i \geq B}{\sum_i (c_A a_i + c_B b_i) \ell_i \geq c_A A + c_B B} \quad [c_A, c_B \in \mathbb{N}]$$

Division $$\frac{\sum_i c a_i \ell_i \geq A}{\sum_i a_i \ell_i \geq \lceil A/c \rceil} \quad [c \in \mathbb{N}^+]$$

Toy example:

$$\dfrac{\text{Lin comb}\ \dfrac{2x + 4y + 2z + w \geq 5 \qquad 2x + y + w \geq 2}{6x + 6y + 2z + 3w \geq 9} \qquad \overline{z} \geq 0}{\text{Div}\ \dfrac{6x + 6y + 3w \geq 7}{2x + 2y + w \geq 3}}\ \text{Lin comb}$$

EXTENSION RULE: REDUNDANCE-BASED STRENGTHENING

C is redundant with respect to F if F and $F \wedge C$ are equisatisfiable

Want to allow adding redundant constraints

Redundance-based strengthening [BT19, GN21, BGMN22]

C is redundant with respect to F if and only if there is a substitution ω (mapping variables to truth values or literals), called a witness, for which

$$F \wedge \overline{C} \models (F \wedge C)\restriction_{\omega}$$