

ManySAT 1.1: solver description

Youssef Hamadi^{1,2}, Said Jabbour³, and Lakhdar Sais⁴

¹ Microsoft Research, 7 J J Thomson Avenue, Cambridge, United Kingdom
youssefh@microsoft.com

² LIX École Polytechnique, F-91128 Palaiseau, France

³ Microsoft-INRIA joint-lab, 28, rue Jean Rostand, 91893 Orsay, France
said.jabbour@inria.fr

⁴ CRIL-CNRS, Université Lille Nord de France
Rue Jean Souvraz SP18, F-62307 Lens Cedex France
sais@cril.fr

Overview

ManySAT is a parallel DPLL-engine which includes all the classical features like two-watched-literal, unit propagation, activity-based decision heuristics, lemma deletion strategies, and clause learning [4, 5]. In addition to the classical first-UIP scheme, it incorporates a new technique which extends the classical implication graph used during conflict-analysis to exploit the satisfied clauses of a formula [1].

When designing ManySat we decided to take advantage of the main weakness of modern DPLLs: their sensitivity to parameter tuning. For instance, changing parameters related to the restart strategy or to the variable selection heuristic can completely change the performance of a solver on a particular problem. In a multi-threading context, we can easily take advantage of this lack of robustness by designing a system which will run different incarnation of a core DPLL-engine on a particular problem. Each incarnation would exploit a particular parameter set and their combination should represent a set of orthogonal strategies.

To allow ManySAT to perform better than any of the selected strategy, conflict-clause sharing was added. Technically, this is implemented through lock-less shared data structures. The version 1.1 implements innovative dynamic clause sharing policies [6].

Code

The system is written in C++ and has about 4000 lines of code. It is written on top of minisat 2.02 [3], which was extended to accommodate the new learning scheme, the various strategies, and our multi-threading clause sharing policy. SatElite is systematically applied as a pre-processor [2].

References

1. Gilles Audemard, Lucas Bordeaux, Youssef Hamadi, Saïd Jabbour, and Lakhdar Sais. A generalized framework for conflict analysis. In *11th International Conference*

- on *Theory and Applications of Satisfiability Testing - SAT'2008*, volume 4996 of *Lecture Notes in Computer Science*, pages 21–27. Springer, 2008.
2. Niklas Eén and Armin Biere. Effective preprocessing in sat through variable and clause elimination. In Fahiem Bacchus and Toby Walsh, editors, *SAT*, volume 3569 of *Lecture Notes in Computer Science*, pages 61–75. Springer, 2005.
 3. Niklas Eén and Niklas Sörensson. An extensible sat-solver. In Enrico Giunchiglia and Armando Tacchella, editors, *SAT*, volume 2919 of *Lecture Notes in Computer Science*, pages 502–518. Springer, 2003.
 4. Y. Hamadi, S. Jabbour, and L. Sais. Manysat: solver description. Technical Report MSR-TR-2008-83, Microsoft Research, May 2008.
 5. Y. Hamadi, S. Jabbour, and L. Sais. Manysat: a parallel SAT solver. *Journal on Satisfiability, Boolean Modeling and Computation (JSAT)*, under submission, 2009.
 6. Youssef Hamadi, Saïd Jabbour, and Lakhdar Sais. Control-based clause sharing in parallel sat solving. In Craig Boutilier, editor, *IJCAI*, pages 499–504, 2009.