# MiniSat v2.0 (Beta)

Niklas Eén
Cadence Berkeley Labs, Berkeley, USA.
niklas@cadence.com

Niklas Sörensson
Chalmers University of Technology, Göteborg, Sweden.
nik@cs.chalmers.se

August 15, 2006

## Abstract

This document describes the current state of the open source SAT solver MiniSat. The latest version brings new features that was proven to be successful during the SAT competition 2005, namely the variable elimination based simplification techniques of SatELiteGTI (winner of the industrial category). The resulting solver is mainly intended to be easier to use, understand, and extend — therefore performance is estimated to be roughly similar to that of SatELiteGTI.

## 1 Description

Originally inspired by ZChaff [4] and Limmat [1], MiniSat features the now commonplace two-literal watcher scheme for BCP, first-UIP conflict clause learning, and the VSIDS variable heuristic (see [3] for a detailed description). Additionally, it has support for incremental SAT solving, and it exists in variations that support user defined boolean constraints and proof-logging.

**Version 1.14** This is the latest public release, which mainly constitutes a cleaned-up version of the MiniSat that participated in the SAT competition 2005. At the time of the competition, unique new features were: an improved VSIDS inspired variable heuristic, optimization of binary clauses, and most notably, conflict clause minimization. In the new version, the optimization for binary clauses was removed to ease the implementation of the simplification techniques. It will probably be added again in the future.

**What's new in 2.0** The main new feature is a CNF simplification similar to that of SatELite [2]; namely variable elimination using clause distribution, and simplification using subsumption and subsumption resolution. Since it has been integrated into MiniSat it can now be used incrementally, as opposed to SatELiteGTI. Moreover, the implementation has been significantly simplified and made more robust.

A minor new feature, that has no significance for the competition, is an extension to the incremental interface: given that a call to 'solve' with a certain set of assumptions is unsatisfiable, a special conflict clause is returned that is made false by the assumptions alone. This feature has its uses in applications where you benefit from knowing if an assumption is unnecessary for the contradiction.

**SAT-Race hacks** Although the simplification of MiniSat is much more robust than that of

SatELite in unrestricted mode, there are still extreme cases where it takes too long. To avoid this we implemented the simplest possible kind of restriction: if the number of clauses is greater than 4000000, the simplification is turned off.

Finally, a simple change to the restart strategy was implemented: until the first conflict after each restart the random-frequency for the variable heuristic is increased. The intention was to help diversify the search, possibly improving on satisfiable instances. But because it was a pretty late hack, it wasn't evaluated thoroughly, and since then we have had indications that it is in fact detrimental overall.

# References

[1] A. Biere. The evolution from limmat to nanosat. In *Technical Report 444, Dept. of Computer Science, ETH Zürich*, 2004.

[2] N. Een and A. Biere. Effective preprocessing in SAT through variable and clause elimination. In *Proc. of the 8$^{th}$ Int. Conference on Theory and Applications of Satisfiability Testing*, volume 3569 of *LNCS*, 2005.

[3] N. Een and N. Sörensson. An extensible sat solver. In *Proc. of the 6$^{th}$ Int. Conference on Theory and Applications of Satisfiability Testing*, 2003.

[4] M. W. Moskewizc, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient sat solver. In *Proc. of 12$^{th}$ Int. Conference on Computer Aided Verification*, volume 1855 of *LNCS*, 2001.