eSAT: solver description

Gilles Audemard[2], Lucas Bordeaux[1], Youssef Hamadi[1], Said Jabbour[2], and Lakhdar Sais[2]

[1] Microsoft Research
7 J J Thomson Avenue, Cambridge, United Kingdom
{lucasb, youssefh}@microsoft.com
[2] CRIL-CNRS, Université d'Artois
Rue Jean Souvraz SP18, F-62307 Lens Cedex France
{audemard, jabbour, sais}@cril.fr

Overview

eSat is a DPLL-based satisfiability solver which includes all the classical features like lazy data-structures and activity-based decision heuristics. It differs from state-of-the-art satisfiability solvers such as Rsat [5] and MiniSAT [3] on many important components such as restart strategies and clause learning. 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]. It also include a dynamic restart strategy, where the cut-off value of the next restart is computed using information gathered in the two previous runs. Finally, a new phase-learning [4, 5] policy based on the occurrences of literals in the learnt clauses is used.

Code

The system is written in C++ and has about 2000 lines of code. It was submitted to the race as a 32 bit binary. It is written on top of minisat 2.02 [3], which was extended to accommodate the new learning scheme and the new policies. SatElite was also applied systematically as a pre-processor [2].

References

1. G. Audemard, L. Bordeaux, Y. Hamadi, S. Jabbour, and L. Sais. A generalized framework for conflict analysis. In *SAT (to appear)*, 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. Daniel Frost and Rina Dechter. In search of the best constraint satisfaction search. In *AAAI*, pages 301–306, 1994.

5. Knot Pipatsrisawat and Adnan Darwiche. A lightweight component caching scheme for satisfiability solvers. In João Marques-Silva and Karem A. Sakallah, editors, *SAT*, volume 4501 of *Lecture Notes in Computer Science*, pages 294–299. Springer, 2007.