SArTagnan

Description of the parallel solver SArTagnan
submitted for the SAT-Race 2010

Stephan Kottler

Eberhard Karls Universität Tübingen, Germany
kottlers@informatik.uni-tuebingen.de

Brief overview of SArTagnan

SArTagnan is a new parallel SAT-solver that runs different algorithms and search strategies on different threads. The solver is implemented in C++ using OpenMP and was submitted as 64-bit binary version.

Clause sharing All threads are allowed to share clauses [3, 14, 7] logically and physically. However, the set of clauses of different threads may differ and not all clauses have to be shared. One criterion to decide on which clauses to share is the LBD value [1]. All sharing is generally realised without mutex locks of the operating system.

Different strategies Most threads use CDCL [11] with the VSIDS heuristic [12] for variables [4], Luby restarts [10] and phase-saving [13]. However, three threads use geometric restarts and one thread uses activity values for literals as in the original VSIDS heuristic [12]. Most threads apply lazy hyper binary resolution as proposed in [2].

Sharing clauses physically allows for easily sharing different kinds of information among several threads. E.g. if one thread detects a clause for "on the fly improvement" [8] all threads may profit from this immediately. In this spirit two threads (when run with 8 threads) mainly attempt to improve the clause set for the other solvers:
One thread uses reference points for decision making (DMRP) as proposed in [5, 6] and similar to [9]. It frequently computes a reference point which attempts to reflect the search direction of several solvers: This is done by choosing the predominant assignment values of all solvers for the reference point. Subsequently the DMRP thread focuses on the set of clauses that are not fulfilled by this reference point. Considering these clauses for decision making often allows for learning valuable lemmata.

Another thread tries to simplify the clause database by eliminating or replacing variables. It also performs subsumption and backward subsumption checks and searches for autarkies.

References

[1] Gilles Audemard and Laurent Simon. Predicting learnt clauses quality in modern sat solvers. In *IJCAI*, pages 399–404, 2009.

[2] Armin Biere. Lazy hyper binary resolution. Algorithms and Applications for Next Generation SAT Solvers, Dagstuhl Seminar 09461, Dagstuhl, Germany, 2009.

[3] Wolfgang Blochinger, Carsten Sinz, and Wolfgang Küchlin. Parallel propositional satisfiability checking with distributed dynamic learning. *Parallel Computing*, 29(7):969–994, 2003.

[4] Niklas Eén and Niklas Sörensson. An extensible SAT-solver. In *SAT 2003*, 2003.

[5] Eugene Goldberg. Determinization of resolution by an algorithm operating on complete assignments. In *SAT 2006*, 2006.

[6] Eugene Goldberg. A decision-making procedure for resolution-based SAT-solvers. In *SAT 2008*, 2008.

[7] Youssef Hamadi and Lakhdar Sais. Manysat: a parallel sat solver. *Journal on Satisfiability, Boolean Modeling and Computation (JSAT*, 2009.

[8] HyoJung Han and Fabio Somenzi. On-the-fly clause improvement. In *SAT*, 2009.

[9] Stephan Kottler. SAT Solving with Reference Points. In *SAT 2010*, 2010.

[10] Michael Luby, Alistair Sinclair, and David Zuckerman. Optimal speedup of las vegas algorithms. In *ISTCS*, pages 128–133, 1993.

[11] J. Marques-Silva. The impact of branching heuristics in propositional satisfiability algorithms. In *EPIA '99: Proceedings of the 9th Portuguese Conference on Artificial Intelligence*, pages 62–74, London, UK, 1999. Springer-Verlag.

[12] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: engineering an efficient SAT solver. In *DAC*, 2001.

[13] Knot Pipatsrisawat and Adnan Darwiche. A lightweight component caching scheme for satisfiability solvers. In *SAT*, pages 294–299, 2007.

[14] Tobias Schubert, Matthew D. T. Lewis, and Bernd Becker. Pamira - a parallel sat solver with knowledge sharing. In *MTV*, pages 29–36, 2005.