

EBGlucose: learnt clause quality prediction over a new class.

Bryan Matsuo

Department of Computer Science, UCSC

March 2, 2011

Abstract

For SAT Competition 2011 we are presenting a solver EBGLUCOSE based of the winner in the industrial UNSAT category of the 2009 competition GLUCOSE . EBGLUCOSE is part of an experiment examining the usefulness of a new clause class proposed by the authors of RSAT in [PD08]. This report summarizes the techniques employed and implementation decisions made in EBGLUCOSE . The “EB” prefix in the name EBGLUCOSE comes from the 1-Empowering Bi-asserting (EB) clause learning technique which has been embedded in this solver.

1 Introduction

In the 2009 SAT Competition the solver GLUCOSE introduced a static measure of clause usefulness with an aggressive clause removal. In the competition as well as [AS09], the authors demonstrated GLUCOSE ’s ability to solve UNSAT instances from industrial applications. Also recently, the authors of RSAT introduced and studied a new class of learnt clause known as the 1-empowering bi-asserting clause. When implemented, the new clause class reportedly gave RSAT an enhanced ability to solve UNSAT instances of an industrial nature [PD08]. The solver presented here, EBGLUCOSE , was developed to give more experimental knowledge about the usefulness of this new clause class. EBGLUCOSE ’s source is based on GLUCOSE version 1.0. The new clause learning technique was then embedded and tuned for performance.

2 A new class of learnt clause

A class of learnt clauses was presented in [PD08] and implemented in the solver RSAT for SAT competition 2009. This technique involves attempting to learn clauses lying in the intersection of the following two clause classes if they

have low enough backtrack levels (in relation to the FUIP).

Definition 2.1 (1-Empowerment). *A clause $c = (\alpha \Rightarrow \ell)$ for some literal ℓ and conjunction of literals α is 1-empowering with respect to a clause set Δ if*

1. $\Delta \models c$: Δ logically implies clause c .
2. $\Delta \wedge \alpha$ is 1-consistent: Asserting α does not result in a conflict detectable by unit resolution.
3. $\Delta \wedge \alpha \not\models \ell$: Literal ℓ cannot be derived from $\Delta \wedge \alpha$ via unit-propagation.

The literal ℓ is known as an empowering literal of c .

Definition 2.2 (Bi-asserting Clause). *A conflict clause c in at a given state solver state is bi-asserting if contains exactly two literals on the highest decision level.*

Clauses in this class intersection are non-asserting but are generally shorter than the FUIP. Checking for 1-Empowerment in conflict clauses with respect to the entire clause database is too costly to do. However, [PD08] gives an efficient algorithm for detecting 1-Empowerment with respect to the clauses used in the derivation of the conflict clause (which tends to be accurate enough) incurring minimal overhead.

As with the authors of [PD08], 1-empowering bi-asserting clauses are learned if their decision level is at least two levels below that of the FUIP corresponding to the same conflict.

This feature is main addition made to the original GLUCOSE solver. The technique was embedded to study the usefulness this class of clauses under GLUCOSE ’s static clause quality measure and aggressive database management.

3 Using bi-asserting clauses

Because learnt bi-asserting clauses are non-asserting, it isn’t immediately clear how a solver

should proceed after learning them. There are a number of combinations of ways you can affect the next decision, including a forceful assumption. Preliminary experiments run by the author have shown no significant benefit to any particular action. Thus, EBGlucoSE simply defers the next decision to the main decision engine inherited from GLUCOSE .

4 Predicting clause quality

EBGLUCOSE judges clauses based on a static measure described first in [AS09]. The measure, known as the *Literal Blocks Distance (LBD)* of a learnt clause, is calculated when the clause is learnt, prior to backtracking. It is defined as follows.

Definition 4.1 (Literal Blocks Distance (LBD)). *Given a clause C and a partition of its literal into n subsets s.t. literals are partitioned w.r.t. their decision level in the current assignment, the LBD of C is exactly n .*

A practical justification of this measure for learnt FUIP clauses is described in [AS09]. It should be somewhat obvious that a bi-asserting clause found before the FUIP will have a lower LBD measure than the FUIP. This could indicate that GLUCOSE 's clause deletion mechanism will be kind to learnt bi-asserting clauses and tend to keep them in the database. The because of this the GLUCOSE solver seems like an ideal scenario in which to study the usefulness of of bi-asserting clauses.

5 Aggressive clause deletion

As in GLUCOSE [AS09], the clause deletion mechanism in EBGlucoSE deletes at most half of the clauses in the database on a static schedule. The number of conflicts between two database reductions (and before the first) follows the sequence $\{t_k\}_{k=1}^{\infty}$ where $t_k = 20000 + 500(k - 1)$. [AS09] displayed that this technique drastically reduced long term database size and has the effect of increasing the unit-propagation rate when embedded in minisat.

In SAT Race 2010, GLUCOSE 1.1 was entered and was described to have a new clause deletion heuristic which would delete clauses with higher LBD first and break ties with MINISAT 's

clause activity measure [SGH⁺10]. This feature of GLUCOSE 1.1 was re-implemented in EBGlucoSE .

6 Other embedded techniques

The GLUCOSE solver (and EBGlucoSE by extension) makes use of many state-of-the-art techniques besides a non-standard clause quality measure and aggressive clause deletion scheme. It uses a dynamic restart strategy, and handles variable polarity by using a phase-caching schema [PD07]. Also, using the VSIDS heuristic for variables presented in [MMZ⁺01], GLUCOSE manipulates future decision to promote production of clauses with small LBD. For more information see [AS09].

7 Acknowledgements

Many thanks are due to Allen Van Gelder for helpful implementation discussions and guidance in benchmarking.

References

- [AS09] Gilles Audemard and Laurent Simon, *Predicting learnt clauses quality in modern sat solvers*, Proceeding of the International Joint Conferences on Artificial Intelligence, 2009.
- [MMZ⁺01] Matthew W. Moskewicz, Conor F. Madigani, Ying Zhao, Lintao Zhang, and Sharad Malik, *Chaff: Engineering an efficient sat solver*, 2001, pp. 530–535.
- [PD07] Knot Pipatsrisawat and Adnan Darwiche, *A lightweight component caching scheme for satisfiability solvers*, Proceedings of 10th International Conference on Theory and Applications of Satisfiability Testing (SAT), 2007, pp. 294–299.
- [PD08] Knot Pipatsrisawat and Adnan Darwiche, *A new clause learning scheme for efficient unsatisfiability proofs*, Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI), 2008.
- [SGH⁺10] Carsten Sinz, Aarti Gupta, Youssef Hamadi, Himanshu Jain, Daniel Le Berre, Panagiotis Manolios, Yakov Novikov, and Florian Merz, *SAT race*, 2010, <http://baldur.iti.uka.de/sat-race-2010/index.html>.