SAT_Power SAT Solver: System Description

Abdorrahim Bahrami, Seyed Rasoul Mousavi, Maryam Farshchian

June 7, 2010

1. Introduction

The SAT_Power is a two-phase SAT solver. The first phase is a simplification phase which converts a CNF formula to an equivalent but simpler CNF formula, and the second phase is to call a SAT solver to solve the simplified CNF formula. The emphasis is on the first phase; any sat solver may be used as for the second phase. This solver uses the latest available version of Precosat. Actually, the first phase is a preprocessing phase and tries to simplify the CNF formula using unary and binary clauses of the formula. This phase has three parts which are described next.

2. Resolution on binary Clauses

The first part of the simplification is to use the resolution rule on binary clauses to extract unary ones. If two binary clauses exist in the CNF formula in the forms $(l_i \vee l_j)$ and $(\bar{l}_i \vee l_j)$, a unary clause in the form (l_i) can be extracted from them. In this part of preprocessing, all binary clauses in the given formula are checked to find every pair from which a unary clause can be extracted.

For speed up, this part is performed at the same time the CNF formula is being read and prepared into the data structures.

3. Unit-propagation

All the unary clauses are stored in a queue while the CNF formula is being read. This includes the new unary clauses extracted from the binary ones. Then, a unit-propagation task is performed to further simplify the formula and also to eliminate all the unary clauses.

4. Finding Strongly Connected Component

After eliminating all unary and binary clauses in the forms $(l_i \vee l_j)$ and $(\bar{l}_i \vee l_j)$, an Implication Graph with all of the remaining binary clauses are constructed. As in 2-SAT, each strongly-connected component can be used for the purpose of binary equivalence of some literals. For example, if a strongly-connected component with three nodes l_i, l_j and l_k exist in the implication graph, then there will be paths from l_i to l_j and from l_j to l_k and from l_i to l_k, and vice versa. Recall that in an implication graph, if a literal l_i has a path to another literal l_j then l_i implies l_j or if $l_i = true$ then $l_j = true$ either. So if a strongly-connected component with three nodes l_i, l_j and l_k exist in an implication graph, then all these literals will be equivalent. So l_j and l_k can be replaced with l_i , and also $\bar{l}_j$

and $\bar{l}_k$ can be replaced with l_i. Therefore, by finding strongly-connected components, some variables are eliminated and the CNF formula is simplified and is expected to be solved in less time than would otherwise be the case.

Having this accomplished, the simplification phase is completed, and a sat solver, Precosat must be invoked to solve the simplified formula.