Actin - Activity Initialization

Raihan H. Kibria
Computer Systems Lab, Dept. of Electrical Engineering and Information Technology
Darmstadt University of Technology, D-64283 Darmstadt, Germany
http://www.rs.e-technik.tu-darmstadt.de/
kibria@rs.tu-darmstadt.de

July 3, 2006

Abstract

Actin (MiniSAT+i) is a DPLL SAT solver written in C++, based on MiniSAT v1.14 [1] by Niklas Een and Niklas Sörensson. It adds *activity initialization*, but is otherwise unchanged from the original program.

1 Introduction

MiniSAT is a complete DPLL-based SAT-solver with conflict-driven learning. The decision heuristic is based on variable activity, similar to the *VSIDS* heuristic introduced in Chaff. All activities have the initial value zero.

In Actin the activities are initialized with values computed from the SAT problem's CNF (only once, after the preprocessing BCP stage). The algorithm to compute the activities was found utilizing *genetic programming* (GP), a type of evolutionary algorithm [2].

2 Activity initialization

The algorithm for computing the initial activities requires the variable counts (how often each variable occurs in the clauses) for all variables of the CNF. For each variable x the *raw initial activity* $a_{raw}(x)$ is computed:

$$a_{raw}(x) = \sum_{\substack{\text{clauses } C,\\ x \in C\\ \vee \neg x \in C}} \; \sum_{\substack{\text{literals}\\ l \in C,\\ l \neq x\\ \wedge l \neq \neg x}} \text{varcount}(l) \tag{1}$$

varcount(l) is the number of occurences of the variable of literal l in the CNF. The $a_{raw}(x)$ are *normalized* by finding the largest raw initial activity $a_{raw,max}$ and dividing all $a_{raw}(x)$ by it, yielding the actual initial activity $a_0(x)$ of each variable x:

$$a_0(x) = \frac{a_{raw}(x)}{a_{raw,max}} \Rightarrow a_0(x) \in [0,1] \tag{2}$$

3 Usage and expected behaviour

Actin only takes one argument on the command line, this is the name of the .cnf file. If no argument is given, standard input is read. It is not possible to give a time limit.

Actin is a complete DPLL SAT solver and will eventually return SATISFIABLE or UNSATISFIABLE for any SAT problem. If the problem is satisfiable, a model is printed out containing assignments to all variables in the problem.

References

[1] The MiniSAT page.
http://www.cs.chalmers.se/Cs/Research/FormalMethods/MiniSat/Main.html

[2] R. H. Kibria, Y. Li: Optimizing the Initialization of Dynamic Decision Heuristics in DPLL SAT Solvers Using Genetic Programming. EuroGP 2006, 331-340.
http://dx.doi.org/10.1007/11729976_30