Glueminisat-ActivityMini

Seongsoo Moon
Graduate School of Information Science and Technology,
The University of Tokyo, Japan

Inaba Mary
Graduate School of Information Science and Technology,
The University of Tokyo, Japan

***Abstract*—We briefly introduce our solver, Glueminisat-Activitymini, submitted to SAT-Race 2015. This solver is an implementation of minor enhancements of VSIDS, most commonly used decision heuristic. As a base solver, Glueminisat is used.**

I. Introduction

Decision heuristic is one of the most important elements in modern SAT solvers. The most prominent method is VSIDS[1]. There were lots of attempts to surpass VSIDS [2] [3] [4], but VSIDS is still most popular decision heuristic because of its robustness.

In VSIDS, each variable has a *Activity*. And when a clause is added to learned clauses database, the *Activity* of each literal in the clause is incremented. When we pick a variable based on VSIDS, a variable with the highest *Activity* is chosen. However, from time to time, the number of variable with the highest *Activity* is not just 1. These ties are broken randomly in VSIDS.

In our program, we propose decision heuristic when ties occur.

II. *ActivityMini*

Consider the random picking method from ties in VSIDS when ties occur. It is well known feature of CDCL SAT solvers, that their running time can vary substantially due to small changes of decision order. Therefore, when we choose a variable from ties randomly, this choice will change running time, and the reduced running time will appear a fifty-fifty chance or lower because of the cost of random picking.

We propose *ActivityMini* to subserve *Activity* in *VSIDS* when ties occur. In our program, each variable has a *ActivityMini* as well as *VSIDS*.

The pseudo code of *ActivityMini* is exhibited in Figure 1. The function "attach_clauses", "detach_clauses" updates *ActivityMini*. We add (or subtract) $\frac{1}{learnt\ clause\ length}$ for each variable in a learnt clause to allocate extra weight for short clauses because we want to pick the variable having high probability of unit propagation from ties in VSIDS.

The function "get_lookup_size" returns the size of look-up table of 2-literal watching[1]. Although this size doesn't exactly correspond to the actual number of that variable in the database of learnt clauses, picking a variable with a large look-up table might be a better choice than picking a variable having a small look-up table entirely.

We define some parameters MAX_CN, TH1, TH2 and TH3 in pseudo code. In this time, We set MAX_CN = 50, TH1 = 1.0, TH2 = TH3 = 0.8 based on some empirical tests.

```
attach_clauses() {
  for(var in learnt clause) {
    actMini[var] += 1 / (learnt clause size);
  }
}

detach_clauses() {
  for(var in learnt clause) {
    actMini[var] -= 1 / (learnt clause size);
  }
}

get_lookup_size(var) {
  return lookup size of variable var;
}

pick_literal() {
  topAct = (max act from all variables);
  topActMini = (max actMini from ties);
  while(candidates.size() < MAX_CN &&
        act[i] > topAct * TH1 &&
        actMini[i] > topActMini * TH2) {
    candidates.add(i);
  }
  for(i in candidates) {
    max_lookup_size
      = max(max_lookup_size, get_lookup_size(i));
  }
  for(i in candidates) {
    if(lookup_size[i] > max_lookup_size * THRES3)
      if(max_act < (actMini[i] * lookup_size[i]))
        update max_activity, max_idx;
      }
    }
  }
  return max_idx;
}
```

Fig. 1. Pseudo code of *ActivityMini*

REFERENCES

[1] Moskewicz, M.W., Madigan, C.F., Zhao, Y., Zhang, L., Malik, S. Chaff: Engineering an Efficient SAT Solver. In Proceedings of the 38th Design Automation Conference, pp 530–535, 2001.

[2] Dershowitz, Nachum and Hanna, Ziyad and Nadel, Alexander. A Clause-Based Heuristic for SAT Solvers. Theory and Applications of Satisfiability Testing, pp 46–60, 2005.

[3] Goldberg, Evgueni and Novikov, Yakov. BerkMin: A Fast and Robust Sat-Solver. Design, Automation, and Test in Europe, pp 465–478, 2008.

[4] L.Ryan. Efficient algorithms for clause-learning SAT solvers. Matser's thesis, Simon Fraser University, 2004.