

This paper appears in the *First International Competition and Symposium on Satisfiability Testing*, Beijing, CHINA, 1996.

SUN: a Multi-Strategy Platform for SAT

Mazure Bertrand

CRIL

Université d'Artois

rue de l'Université SP 16

F-62307 Lens Cedex

France

mazure@lens.lifl.fr

Saïs Lakhdar

CRIL - IUT de Lens

Université d'Artois

rue de l'Université SP 16

F-62307 Lens Cedex

France

sais@lens.lifl.fr

Grégoire Eric

CRIL

Université d'Artois

rue de l'Université SP 16

F-62307 Lens Cedex

France

gregoire@lens.lifl.fr

Abstract

SUN is a multi-strategy platform for SAT. It includes a whole family of local search techniques and some of the best Davis and Putnam's strategies for checking propositional satisfiability. Most notably, it features an optimized tabu-based local search method and includes a new logically complete approach that combines the respective strengths of local and systematic searches. SUN is a comprehensive toolkit provided with most current SAT instances generators and with various user-friendly tools.

1 Introduction

SUN (**S**at or **UN**sat) is a multi-strategy platform for SAT that has been implemented at the Université d'Artois in France. It provides the user with a selection of both local and systematic searches. More precisely, it includes many variants of GSAT- like techniques that prove efficient in showing the consistency of SAT instances. In particular, it features an optimized tabu-based local search method. SUN also includes a new logically complete approach that combines the power of local search methods with the completeness of Davis and Putnam's procedure (DP), allowing for both large satisfiable and unsatisfiable instances to be proved. It thus shows that local search methods can play a key role with respect to proving unsatisfiability.

In this extended abstract, first the methods and tools covered by SUN are listed. Then, the focus is laid on the main two features of SUN: an optimized tabu-based local search method and its use as an heuristic to guide the branching strategy of DP.

2 Current Available Search Techniques and Generators in SUN

SUN has been developed in standard ANSI C and should run on most UNIX systems. Our experimentations have been conducted on 133 Pentium under Linux 1.2.13. All the available search techniques in SUN have been implemented by our own (when these techniques have been previously proposed by other authors, our implementation is as efficient as the original code).

2.1 Uncomplete techniques

- *GSAT* : the well-known basic local search method by Selman *et al.* [12]
- *TWSAT* : an optimized tabu-based local search method by Mazure *et al.* [9]

with the following available (combinable) strategies

- *-rws* : Random Walk Strategy by Selman *et al.* [13]

- *-weight* : proposed by Selman *et al.* [13] and further used by Cha and Iwama [2]
- *-unbalance* : generates the initial interpretation taking into account the difference between the numbers of positive and negative literals occurring in the clauses

2.2 Complete techniques

- *DP* : Davis and Putnam's procedure [3]

with the following available branching strategies

- *jeroslow* : Jeroslow and Wang's heuristic [6]
- *ffis* : First-Failed in Shortened Clause by Rauzy [11]
- *matrix_order* : lexicographic order
- *max_S-U* : maximizes the difference of the number of occurrences in the shortened clauses and in the untouched ones
- *gsat* : uses Gsat (and its options) to deliver the next literal to be assigned [10]
- *twsat* : uses Twsat (and its options) to deliver the next literal to be assigned [10]

with the following available options

- *-ms* : applies an extension of Oxusof and Rauzy's model-splitting theorem [8]
- *-am* : restricted *-ms* extension [8]
- *-use_clauses* : selects the next literal as the literal that has appeared most often in the most often falsified clauses by Gsat or Twsat [10]

2.3 Generators

- *ar*: generates K-sat instances according to the fixed-length clause model
- *qh*: generates satisfiable ar K-sat instances (forbidding positive clauses) [1]
- *2-3-SAT*, *3-4-SAT*, *2-3-3-SAT*, *2-4-4-SAT*, *2-3-4-SAT*: mixed-clause length model generators by Gent and Walsh [5]
- *ramsey*: generates instances of Ramsey's graph-coloring theorems
- *queens*, *pigeons*: generates N-Queens and Pigeons-Holes problems
- *krishnamurthy*: generates problems proposed by Krishnamurthy [7]
- *dubois*: generates Dubois' problems [4]

The other various software tools in SUN are described in an interactive help module. Thanks to its open-ended architecture, SUN can easily be augmented with other techniques, heuristics, options and generators.

3 TWSAT: a brief description

TWSAT departs from basic GSAT by making a systematic use of a tabu list of variables in order to avoid recurrent flips and thus escape from local minima. This allows a better and more uniform coverage of the search space. More precisely, TWSAT keeps a fixed length - chronologically-ordered FIFO - list of flipped variables and prevents any of the variables in the list from being flipped again during a given amount of time. The length of the tabu list has been experimentally optimized (for K-SAT instances) with respect to the size of the problems.

4 DP + local search: a brief description

Let us now describe a basic algorithm using GSAT-like procedures to guide the branching strategy of logically- complete algorithms based on DP.

```

Procedure DP + TWSAT
Input : a set of clauses S
Output : a satisfying truth assignment of S, if found
           or a definitive statement that S is inconsistent
Begin
  Unit_propagate(S);
  if the empty clause is generated then return (false);
  else if all variables are assigned then return (true)
    else begin
      if TWSAT(S) succeeds then return (true)
      else begin
        p := the most often falsified literal
           during TWSAT search;
        return (DP+TWSAT(S ∧ p) ∨
                DP+TWSAT(S ∧ ¬p));
      end;
    end;
End

```

Figure 1: DP + TWSAT: basic version

TWSAT (or another local search procedure) is run to deliver the next literal to be assigned by DP. This literal is selected as the one with the highest score according to the following counting. A trace of TWSAT is recorded: for each literal, taking each flip as a step of time, we count the number of times the literal appears in the falsified clauses. (Intuitively, it seemed to us that the most often falsified clauses should normally belong to an inconsistent kernel of the SAT instance if this instance is actually inconsistent. This hypothesis proves most often correct).

Such an approach can be seen as using the trace of TWSAT as an heuristic for selecting the next literal to be assigned by DP, and a way to extend the partial assignment made by DP towards a model of the SAT instance when this instance is satisfiable. Each time DP needs to select the next variable to be considered, such a call to TWSAT can be performed with respect to

the remaining part of the SAT instance. This algorithm is given in Figure 1. Let us stress that this combination schema was simply designed to show its feasibility: in this respect, it is a very primitive one that can be optimized in several ways [10].

References

- [1] T. Castell (1995). Personal communication.
- [2] B. Cha, K. Iwama (1995). Performance Test of Local Search Algorithms Using New Types of Random CNF formulas. *Proc. IJCAI-95*, pp. 304-310.
- [3] M. Davis, H. Putnam (1960). A Computing Procedure for Quantification Theory. *Journ. of the ACM*, vol. 7, pp. 201-215.
- [4] DIMACS (1993). Second challenge organized by the Center for Discrete Mathematics and Computer Science of Rutgers University. (The benchmarks used in our tests can be obtained by anonymous ftp from Rutgers University Dimacs Center: ftp dimacs.rutgers.edu cd pub/challenge/sat/benchmarks/cnf)
- [5] I.P. Gent, T. Walsh (1994). The SAT Phase Transition. *Proc. ECAI-94*, pp. 105-109.
- [6] R.E. Jeroslow, J. Wang (1990). Solving Propositional Satisfiability Problems. *Annals of Math. and Art. Int.*, vol. 1, pp. 167-187.
- [7] B. Krishnamurthy (1985). Short Proofs for Tricky Formulas. *Acta Informatica*, vol 22, pp 253-275.
- [8] P. Marquis, B. Mazure, L. Saïs, E. Grégoire (1996). Some Extension of Model Splitting Theorem. *In preparation*.
- [9] B. Mazure B., L. Saïs, E. Grégoire (1995). TWSAT: a New Local Search Algorithm for SAT. Performance and Analysis. *Proc. CP-95 Workshop on Studying and Solving Really Hard Problems*, pp. 127-130.

- [10] B. Mazure, L. Saïs, E. Grégoire E. (1996). Detecting Logical Inconsistencies. *Proc. AI/MATH-96*, pp. 116-121.
- [11] A. Rauzy (1994). On the Complexity of the Davis and Putnam's Procedure on Some Polynomial Sub-Classes of SAT. *LaBRI Technical Report 806-94*, Université de Bordeaux 1.
- [12] B. Selman, H. Levesque, D. Mitchell (1992). A New Method for Solving Hard Satisfiability Problems. *Proc. AAAI-92*, pp. 440-446.
- [13] B. Selman, H.A. Kautz, B. Cohen (1993). Local Search Strategies for Satisfiability Testing. *Proc. 1993 DIMACS Workshop on Maximum Clique, Graph Coloring, and Satisfiability*.