Look-Ahead Versus Look-Back for Satisfiability Problems

Chu Min Li and Anbulagan

LaRIA, Université de Picardie Jules Verne
33, Rue St. Leu, 80039 Amiens Cédex 01, France
tel: (33) 3 22 82 78 75, fax: (33) 3 22 82 75 02
e-mail: {cli@laria.u-picardie.fr, Anbulagan@utc.fr}

Abstract. CNF propositional satisfiability (SAT) is a special kind of the more general Constraint Satisfaction Problem (CSP). While look-back techniques appear to be of little use to solve hard random SAT problems, it is supposed that they are necessary to solve hard structured SAT problems. In this paper, we propose a very simple DPL procedure called Satz which only employs some look-ahead techniques: a variable ordering heuristic, a forward consistency checking (Unit Propagation) and a limited resolution before the search, where the heuristic is itself based on unit propagation. Satz is favorably compared on random 3-SAT problems with three DPL procedures among the best in the literature for these problems. Furthermore on a great number of problems in 4 well-known SAT benchmarks Satz reaches or outspeeds the performance of three other DPL procedures among the best in the literature for structured SAT problems. The comparative results suggest that a suitable exploitation of look-ahead techniques, while very simple and efficient for random SAT problems, may allow to do without sophisticated look-back techniques in a DPL procedure.

1 Introduction

Consider a set of Boolean variables \{x_1, x_2, ..., x_n\}, a literal \(l\) is a variable \(x\) or its negated form \(\bar{x}\), a clause \(c\) is a logical OR of some literals such as \(x_1 \lor \bar{x}_2 \lor x_3\). A propositional formula \(F\) in Conjunctive Normal Form (CNF) is a logical AND of several clauses such as \(c_1 \land c_2 \land ... \land c_m\). \(F\) is often simply written as a set \{\(c_1, c_2, ..., c_m\)\} of clauses.

Given \(F\), the CNF propositional satisfiability (SAT) problem consists in testing whether clauses in \(F\) can all be satisfied by some consistent assignment of truth values \{true, false\} to the variables. If it is the case, \(F\) is said satisfiable; otherwise, \(F\) is said unsatisfiable. SAT is a specific kind of finite-domained Constraint Satisfaction Problem (CSP) in which every variable ranges over the values \{true, false\} and is the first NP-complete problem [4]. When each clause in \(F\) exactly contains \(r\) literals, the restricted SAT problem is called \(r\)-SAT. 3-SAT is the smallest NP-complete subproblem of SAT. We distinguish two types of SAT problems: problems having structures such as regularities, symmetries etc... and random problems without any structure. While real world problems are often
structured, random problems represent the "core" of SAT and are independent of any particular domain.

The most effective systematic algorithms are based on the popular Davis-Putnam procedure in Loveland's form (DPL procedure) [6]. DPL procedures such as C-SAT [7], Tableau [5], and POSIT [8] usually employ a variable ordering heuristic and a forward consistency checking (Unit Propagation) known as look-ahead techniques in CSP terms. These algorithms actually have more or less difficulties to solve structured SAT problems. Recently several authors propose to embed (and emphasize) look-back techniques such as backjumping (also known as intelligent backtracking or non-chronological backtracking) and learning (also known as nogood or constraint recording) in a DPL procedure to attack structured SAT problems. GRASP [17] and relsat(4) [2] are such DPL procedures employing both look-ahead and look-back techniques, which are efficient for structured SAT problems but are not effective for random SAT problems.

In this paper we propose a very simple DPL procedure called Satz which only employs look-ahead techniques and a simple preprocessing of the input CNF formula to add some resolvents of length \( \leq 3 \) into the clause database. The broad experimental comparative results of Satz with several state-of-the-art DPL procedures (C-SAT, Tableau, POSIT, GRASP, relsat(4)) suggest that a suitable exploitation of unit propagation and the preprocessing may be effective for both random SAT problems and a lot of structured ones. Our experience with Satz also enforces the belief that if a DPL procedure is efficient for random SAT problems, it should be also efficient for a lot of structured ones.

The paper is organized as follows. Section 2 presents Satz by discussing its heuristic and the preprocessing of the CNF formula. Section 3 compares Satz on random 3-SAT problems with C-SAT, Tableau, POSIT, the three DPL procedures among the best in the literature for random 3-SAT problems. Section 4 compares Satz on 4 well-known SAT benchmarks with GRASP, POSIT, relsat(4), the three DPL procedures among the best in the literature for structured SAT problems. All experiments are made on a SUN Sparc 20 workstation with a 125 MHz CPU. Section 5 discusses the look-ahead and look-back techniques. Section 6 concludes.

2 About Satz

We roughly sketch the DPL procedure in Figure 1.

DPL procedure essentially constructs a binary search tree through the space of possible truth assignments until it either finds a satisfying truth assignment or concludes that no such assignment exists, each recursive call constituting a node of the tree. Recall that all leaves (except eventually one for a satisfiable problem) of a search tree represent a dead end where an empty clause is found. Look-ahead techniques such as variable ordering heuristics play a determinant role to reach the dead end early to minimize the length of the current path in the search tree.