Cascaded Directed Arc Consistency and No-Good Learning for the Maximal Constraint Satisfaction Problem *

Richard J. Wallace
University of New Hampshire, Durham, NH 03824 USA

Abstract. This paper describes new branch and bound methods for overconstrained CSPs. The first method is an extension of directed arc consistency preprocessing, used in conjunction with forward checking. After computing directed arc consistency counts, inferred counts are derived for each value, based on the counts of supporting values in future variables. This inference process can be 'cascaded' from the end to the beginning of the search order, to augment the initial counts. The second method is a form of wipeout-driven nogood learning: the method for finding nogoods is described and conditions for the validity of the nogood are established. In tests with random problems, significant improvements in efficiency were found with cascaded DACCs; in contrast, no-good learning did not enhance performance when used alone or in any combination of strategies.

1 Introduction

Constraint satisfaction problems (CSPs) involve assigning values to variables which satisfy a set of constraints. Algorithms for ordinary CSPs are designed to return a complete solution, i.e., one that satisfies all of the constraints. But if a problem has no complete solution, these algorithms simply report this.

For problems that have only partial solutions, it may still be useful to have an assignment of values to variables that satisfies the most important constraints or, if constraints have equal weight, one that satisfies as many constraints as possible. (For situations in which good partial solutions to overconstrained problems are useful, see [2] [1] [6] [5].) The latter case, in which an optimal solution is one with a maximal number of satisfied constraints, has been termed the maximal constraint satisfaction problem (MAX-CSP).

In recent years, complete algorithms have been developed for MAX-CSPs that are based on branch and bound methods, using depth-first search [8] [7]. Enhancements have also been developed that use information obtained from local consistency tests carried out before search. This information takes the form of inconsistency counts, i.e., tallies for each value \( a \) of variable \( v_i \), of the number of domains \( v_j \) that do not have any values consistent with \( a \) in the constraint

* This material is based on work supported by the National Science Foundation under Grant Nos. IRI-9207633 and IRI-9504316.
between \( v_i \) and \( v_j \). These counts can be used to compute tighter lower bounds and to order domain values to find solutions with fewer inconsistencies earlier in search. The most powerful techniques of this sort use directed arc consistency counts (DAC-counts or DACCs) that are based on an initial variable ordering, and that refer to variables that are before or after the current value in this order \cite{10}.

In this paper a further elaboration of directed arc consistency counts is considered. Given a domain some of whose values have DACCs, inconsistency counts can sometimes be inferred for values in neighboring domains. For example, if value \( a \) in the domain of \( v_i \) is supported by values in the domain of \( v_j \) that have counts of 1 or more, then a count can be deduced for \( a \) on this basis. If such inferences are carried out systematically, DACCs can be carried forward or backward to one or the other end of the search order. In this way, it may be possible to derive much tighter lower bounds at the beginning of search.

This paper also examines nogood learning in conjunction with algorithms for MAX-CSPs. An appealing feature of this technique in this context is that, once discovered, nogoods are independent of the error factors that go into lower bound calculations. A complication here is that it cannot be assumed that a nogood tuple discovered at one point in search is still no-good at a later point. But, as shown below, a simple test can be done to verify that the tuple is still a valid nogood.

The next section, 2, presents some background for this work. Section 3 describes cascaded DAC procedures. Section 4 describes the no-good learning procedure for MAX-CSPs. Section 5 presents experimental comparisons between these new methods and the best branch and bound algorithms studied in earlier work. Concluding remarks are given in Section 6.

2 Background: Basic Concepts

A constraint satisfaction problem (CSP) involves assigning values to variables that satisfy a set of constraints among subsets of these variables. The set of values that can be assigned to one variable is called the domain of that variable. In the present work all constraints are binary, i.e., they are based on the Cartesian product of the domains of two variables. A binary CSP is associated with a constraint graph, where nodes represent variables and arcs represent constraints.

Branch and bound algorithms associate each path through a search tree with a cost function that is non-decreasing in the length of the path. Search down a given path can stop when the cost of the partial assignment of values to variables is at least as great as the lowest cost yet found for a full assignment. The latter, therefore, sets an upper bound on the cost function. In addition to calculating the cost at a particular node, projected cost can be calculated to produce a higher, and therefore more effective, lower bound. The present algorithms use the number of violated constraints incurred by the partial assignment of values to variables as a cost function; this is called the distance of a partial solution from a complete solution. Maximal solutions are associated with minimum distances.