Travelling in the World of Local Searches in the Space of Partial Assignments*

Cédric Pralet and Gérard Verfaillie

LAAS-CNRS, Toulouse, France
{cpralet,gverfail}@laas.fr

Abstract. In this paper, we report the main results of a study which has been carried out about the multiple ways of parameterising a local search in the space of the partial assignments of a *Constraint Satisfaction Problem* (CSP), an algorithm which is directly inspired from the *decision repair* algorithm [1]. After a presentation of the objectives of this study, we present the generic algorithm we started from, the various parameters that must be set to get an actual algorithm, and some potentially interesting algorithm instances. Then, we present experimental results on randomly generated, but not completely homogeneous, binary CSPs, which show that some specific parameter settings allow such *a priori* incomplete algorithms to solve almost all the consistent and inconsistent problem instances on the whole constrainedness spectrum. Finally, we conclude with the work that remains to do if we want to acquire a better knowledge of the best parameter settings for a local search in the space of partial assignments.

1 Local Search in the Space of Partial Assignments

1.1 Depth-First Tree Search in Constraint Satisfaction

The basic algorithm, designed to solve *Constraint Satisfaction Problems* (CSP [2]), consists of a *depth first* search in the space of the partial assignments, organised into a tree the root of which is the empty assignment, leaves are complete assignments, and each node, except the root, results from the assignment of an unassigned variable in its father node. Variable and value heuristics are used to choose pertinently the next variable to assign and the value to assign to it [3]. Constraint propagation algorithms are also used to enforce any local consistency property, like for example *arc consistency*, at each node of the tree [4, 5]. These algorithms allow domains of the unassigned variables to be reduced and eventually wiped out. In the latter case, inconsistency of the associated subproblem is proven and another value for the last assigned variable is chosen. Producing and recording value removal explanations allow other forms of backtracking than this chronological one to be performed, like for example *conflict directed backjumping* [6].

* The study reported in this paper has been initiated when both authors were working at ONERA, Toulouse, France.

J.-C. Régin and M. Rueher (Eds.): CPAIOR 2004, LNCS 3011, pp. 240–255, 2004.

The main advantage of such a tree search is its completeness: if the problem instance is consistent, a solution is found; if not, inconsistency is established. Another advantage is its limited space complexity, which is only a linear function of the instance size. But its lack of scalability in terms of CPU-time prevents it from being used for solving large instances: at least for the hardest instances at the frontier between consistency and inconsistency, its time complexity is an exponential function of the instance size.

1.2 Local Search in Constraint Satisfaction

On the contrary, local search algorithms perform an unordered search in the space of the complete assignments: each move goes from a complete assignment to another one in its neighbourhood, generally randomly chosen with a heuristic-biased probability distribution (choice of the variable to unassign and of the new value to assign to it). When solving CSPs, each neighbour assignment is evaluated via constraint checking [7]. Meta-heuristics, such as *simulated annealing*, *tabu search*, or others, allow various search schemes to be defined, including *restart* mechanisms [8].

A first advantage of local search is its limited space complexity, which is, as with depth first tree search, only a linear function of the instance size. Its main advantage is however its far better scalability in terms of CPU-time. But it suffers from at least three shortcomings:

– its incompleteness: if the problem instance is consistent, there is no guarantee that a solution will be found; if not, there is no mechanism which allows inconsistency to be established;
– its non deterministic behaviour: when random choice mechanisms are used, running twice the same algorithm on the same instance may give different results;
– its difficulty handling constraints: on the one hand, because local search is basically an optimisation method, it has some difficulty handling correctly together constraints and criterion; on the other hand, because only complete assignments are considered, constraint propagation is not used and is replaced by a simple constraint checking.

1.3 Hybridisations between Tree Search, Local Search, and Constraint Propagation

This landscape, which has been recognised for a long time [9], pushed researchers and practitioners to explore combinations of tree search, local search, and constraint propagation and to propose various hybridisation schemes.

Many of these schemes led to loose combinations: two kinds of search on the same problem in sequence or in parallel [10, 11]; a kind of search on a master subproblem and another kind on the complementary slave subproblem [12]. Stronger combinations have been however proposed with large neighbourhood local searches [13, 14, 15]: when neighbourhoods become large, they can no longer