Making the Breakout Algorithm Complete Using Systematic Search

Carlos Eisenberg and Boi Faltings
Swiss Federal Institute of Technology (EPFL)
1015 Ecublens, Switzerland
{eisenberg,faltings}@lia.di.epfl.ch

Abstract

Local search algorithms have been very successful for solving constraint satisfaction problems (CSP). However, a major weakness has been that local search is unable to detect unsolvability and is thus not suitable for tightly and overconstrained problems. We present a hybrid solving scheme where we combine a local search algorithm - the breakout algorithm, with a systematic search algorithm - backtracking. The breakout algorithm is used for identifying hard or unsolvable subproblems and the backtracking algorithm proves the solvability of these subproblems. The resulting hybrid algorithm is complete and is tested on randomly generated graph 3-colouring problems. The algorithm performs extremely well for all areas of the phase transition and outperforms the individual methods by several orders of magnitude.

1 Introduction

The breakout algorithm is an efficient, local search algorithm for solving Constraint Satisfaction Problems (CSPs). The roots of the algorithm go back to [Minton *et al.*, 1992] and to [Morris, 1993]. [Minton *et al.*, 1992], developed, a local search algorithm, called min-conflict heuristic, which iteratively repairs a given assignment until all conflicts are eliminated. One drawback of the method is that it can get caught in local, non solution minima. Morris eliminated this drawback by extending the algorithm with a breakout method, which allows to escape from local, non solution-minima. The strengths of the breakout algorithm are simplicity, robustness, low memory requirement and high efficiency for solving under constrained problems. [Minton *et al.*, 1992] demonstrated the performance superiority of the min-conflict heuristic by solving large-scale scheduling problems; it performed by orders of magnitude better than traditional backtracking techniques. However, the major weak point of the algorithm is its incompleteness.

In this paper we present a hybrid algorithm that is complete where we combine a local search algorithm, the breakout algorithm, with a complete search algorithm, backtracking. The combination of the two algorithms compensates each others weakness to deal with under and tightly constrained problems. We discover that the algorithm combination also leads to synergies. The weight information from the breakout algorithm can locate and order hard or unsolvable subproblems and guide backtracking by a fail-first variable order. In addition, we introduce a weight sum constraint that can be used to identify unsolvable subproblems of a certain size.

2 The Scheme

When a problem contains an unsolvable subproblem, we observe that the average constraint weight of a constraint belonging to an unsolvable subproblem is higher than the average constraint weight of the problem. In addition, the smaller the unsolvable subproblem, the higher the average constraint weight. This observation is supported by Lemma 1:

Lemma 1*: After increasing the weights n times, the sum of the constraint weights w_{P_k} of an unsolvable subproblem P_k with k constraints must be greater than or equal to $n + k$.*

Proof: If a subproblem is unsolvable, then in every breakout step, one or more of the subproblem constraints is always violated, thus every time the algorithm is caught in a local non solution minimum, one or more of the corresponding weights must be increased. The lower bound of w_{P_k} can be derived by assuming that in every breakout step only one constraint is violated.

By applying Lemma 1 we can define a weight sum constraint, which is extremely useful for tracing unsolvable subproblems of different sizes.

Definition 1 (Weight sum constraint for an unsolvable subproblem P_k): *According to Lemma 1, in an unsolvable subproblem P_k consisting of k constraints after n breakout steps, the following condition*

$$\sum_{i=1}^{k} w(c_i) \geq n + k$$

is satisfied, where c_i are all the constraints of the constraint set C_{P_k} of the subproblem P_k.

The weight sum constraint is a powerful tool for pruning the search for unsolvable subproblems. For example, if we search for an unsolvable subproblem of size 3, we only have to consider the constraints whose weight sum is greater $n+3$.

We also observe that the average constraint weight of an unsolvable subproblem of size k is higher than that of an unsolvable subproblem of size $k + s$ ($s > 0$). This is due to

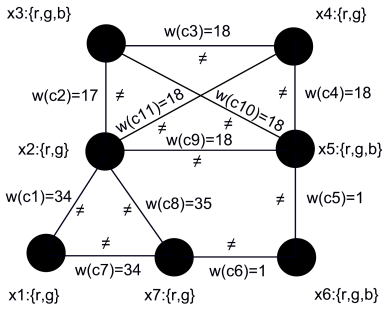

Figure 1: The weight graph of an unsolvable graph colouring problem after 100 breakout steps, containing two unsolvable subproblems of size 3 (c1,c7,c8) and 6 (c2,c3,c4,c9,c10,c11).

the fact that violations cycle over the constraint graph and the less constraints are available, the greater the average constraint weight value. This is demonstrated in Figure 1, containing two unsolvable subproblems of size 3 and 6. After n=100 breakout steps, the average constraint weight of the subproblem of size 3 is approximately n/3 + 1 =34 and for the subproblem of size 6, it is n/6 + 1 =18. Sorting the variables according to increasing constraint weights, orders the subproblem of size 3 before the subproblem of size 4, and thus separates the two.

These observations inspired us to use the constraint weight information for localizing potentially unsolvable subproblems and to derive a fail-first variable ordering heuristic for backtracking. In this heuristic, variables are sorted so that the constraints with the highest weights are treated first.

2.1 Hybrid Solver BOBT

We have implemented this scheme into a hybrid algorithm (BOBT), where we combine the breakout algorithm (BO) with backtracking (BT). In this algorithm, we first execute BO and terminate if no solution is available after a bounded number of breakout steps. Then we derive a fail-first variable order from the constraint weights and the graph structure and start backtrack search. BT will then either prove that the problem is unsolvable, or return a solution. This method guarantees the completeness of the hybrid algorithm. In the case where we find an unsolvable subproblem, we terminate the algorithm and give a failure explanation by returning the minimal unsolvable subproblem. The minimal unsolvable subproblem is computed from the unsolvable subproblem using the technique described in [Faltings, 2002].

3 Experiments and Results

For testing the scheme, we generated a large set of 10,000 random graph 3-colouring problems according to the method described in [Davenport et.al. 1995]. The generated problem graphs consisted of 30 variables with a connectivity of 2-6. The ratio of solvable to unsolvable problems is 1:1. Figure 2 shows the results of the experiments. We show the number of constraint checks for the three algorithms, BO, BT and BOBT on a logarithmic scale over the graph connectivity. For unsolvable problems, we bound the maximum number of iterations for the BO to $4.37 \cdot 10^5$.

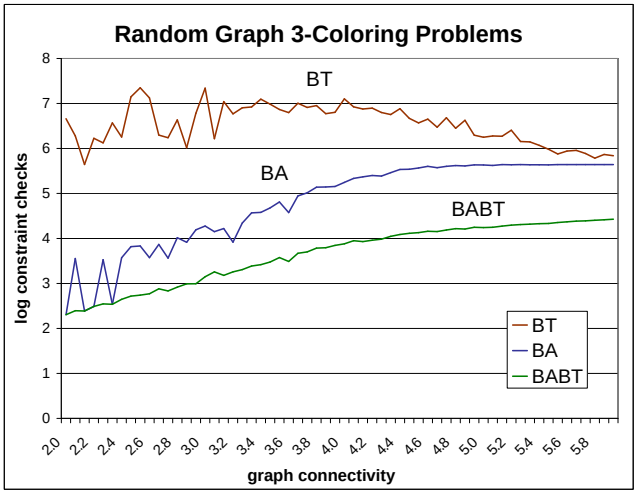

Figure 2: Number of constraint checks on a logarithmic scale for solving 10,000 random generated, 30 node graph 3-colouring problems, solved with BO, BT and BOBT.

4 Conclusion

The main contribution of this paper is the presented hybrid scheme that combines the breakout algorithm with a systematic search method, backtracking, and results in a new algorithm that is complete. In the scheme we use the constraint weight information of the breakout algorithm to identify hard and unsolvable subproblems of increasing sizes and derive in combination with the graph structure a fail-first variable order for a backtracking algorithm. With our results we prove, that the new hybrid search scheme performs extremely well. BOBT needs orders of magnitude less constraint checks than BO and BT. In the future we plan to extend the scheme by identifying the ordered set of all unsolvable sub problems in order to perform a spectral analysis that gives us the distribution of unsolvable subproblems for random graph colouring problems.

References

[Davenport *et al.*, 1992] Andrew Davenport and Edward Tsang. *An empirical investigation into the exceptionally hard problems.* Technical Report CSM-239, Department of Computer Science, University of Essex, U.K., 1995.

[Faltings, 2002] Boi Faltings and Macho-Gonzalez. Open Constraint Satisfaction. *Proceedings of the 8th International Conference, CP 2002* Ithaca, NY, USA, Sept. 2002.

[Minton *et al.*, 1992] Steven Minton and Mark D. Johnston and Andrew B. Philips and Philip Laird. *Minimizing Conflicts: A Heuristic Repair Method for Constraint Satisfaction and Scheduling Problems*. Artificial Intelligence, Vol. 58, P. 161-205, 1992.

[Morris, 1993] Paul Morris. The breakout method for escaping from local minima. *Proc. of the 11th National Conf. on Artificial Intelligence*, Washington, DC, pp. 40–45, 1993.