Corrective Explanation for Interactive Constraint Satisfaction *

Barry O'Sullivan, Barry O'Callaghan and **Eugene C. Freuder**
Cork Constraint Computation Centre
Department of Computer Science, University College Cork, Ireland
{b.osullivan|b.ocallaghan|e.freuder}@4c.ucc.ie

Abstract

Interactive tasks such as online configuration can be modeled as constraint satisfaction problems. These can be solved interactively by a user assigning values to variables. Explanations for failure in constraint programming tend to focus on conflict. However, what is often desirable is an explanation that is *corrective* in the sense that it provides the basis for moving forward in the problem-solving process. This paper defines this notion of *corrective explanation* and demonstrates that a greedy search approach performs very well on a large real-world configuration problem.

1 Introduction

To demonstrate the distinction between corrective explanations and more traditional conflict-based explanations, consider the example in Figure 1.

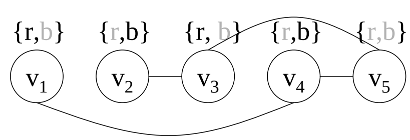


Figure 1: Example of corrective versus conflict-based explanations.

This figure presents assignments of colors to variables in a coloring problem. We have assigned colors to the first 4 variables (highlighted in black), only two colors are available, and we are running a forward checking algorithm, so we have encountered a domain wipeout on variable v_5 at this point. We will say that variables v_3 and v_4 form a "culprit set". The assignments to those variables account for the wipeout. The variables v_3 and v_4 do not provide a corrective explanation because it is not possible by changing their instantiations alone to restore any choices to v_5. $\{v_1, v_4\}$ is a *corrective explanation of inconsistency*, as alternative assignments can be found for the these variables that enables the user to assign at least one more variable. Making v_1 b(lue) and v_4 r(ed) allows us to proceed to assign a value to v_5, blue.

*This work has received support from Enterprise Ireland (Grant SC/02/289) and Science Foundation Ireland (Grant 00/PI.1/C075).

A *corrective explanation for value recovery* is useful when the user would like to restore a value that was removed due to some previous decision(s). Such an explanation identifies modifications to the user's decisions guaranteing that the desired value is restored and can be selected consistently. A corrective explanation is minimal if no (proper) subset is itself a corrective explanation. We define an optimal corrective explanation as one of minimum cardinality.

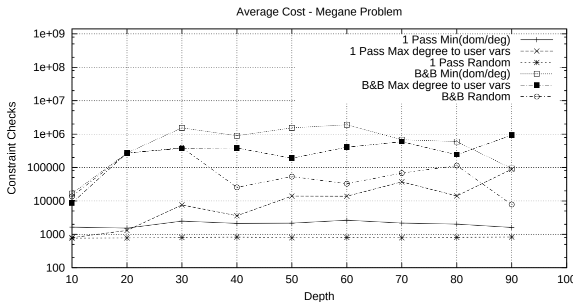
2 Computing Corrective Explanations

Instead of using compilation methods [Amilhastre *et al.*, 2002] or standard model-based diagnosis techniques [Junker, 2004; Reiter, 1987], we advocate the use of a heuristic search approach to find corrective explanations. While heuristic search cannot guarantee that minimum length explanations are found, near optimal explanations can often be found quite efficiently. We generate corrective explanations by returning any differences between the assignments made by the user and the assignments in a solution found using carefully chosen *variable and value ordering heuristics*. These heuristics attempt to maximize the number of user assignments in such a solution. The *value ordering heuristic* favors values chosen by the user during search, by selecting those first whenever possible, and uses standard heuristics for the remaining ones. We partition the set of variables into a set of user-assigned variables and a set of unassigned variables. User-assigned variables are considered first, and within each subset of variables a particular *variable ordering heuristic* is used. The intuition here is that during search the user's choices are always considered first, so they are more likely to participate in the solution used to generate an explanation.

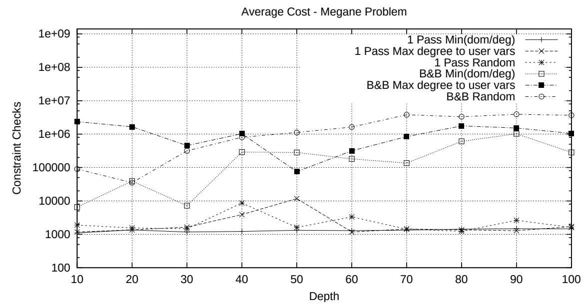
3 Experiments

We ran a suite of experiments based on the Renault Megane Configuration benchmark [Amilhastre *et al.*, 2002]. The problem consists of 101 variables, domain sizes vary from 2 to 43, and there are 113 non-binary constraints. The number of solutions to the problem is in excess of 1.4×10^{12}. The solver we used in our experiments was based on generalized forward checking.

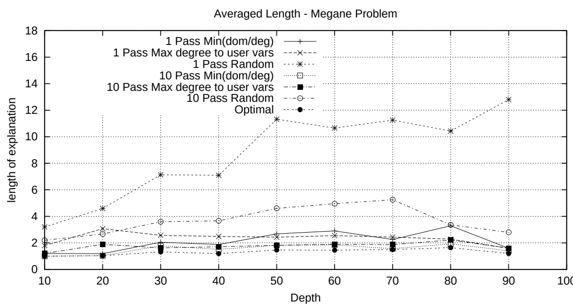
We studied three *variable ordering* heuristics for generating corrective explanations using the approach described above: *minimum ratio of domain over forward degree* (also



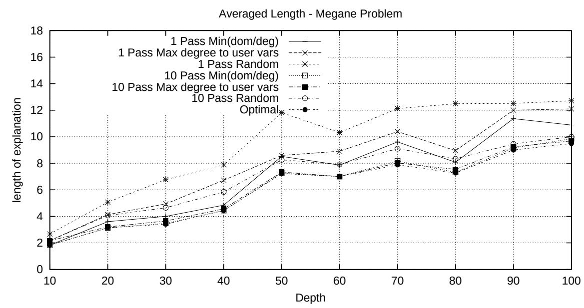
(a) Average cost of inconsistency explanations.



(b) Average cost of value recovery explanations.



(c) Average length of inconsistency explanations.



(d) Average length of value recovery explanations.

Figure 2: Results for finding explanations for the Renault Megane Configuration Benchmark.

referred to as minimum domain/degree), *maximum degree to the user's assigned variables* and *random*. We used a *random value ordering*, since it seemed to perform best overall.

Based on the variable and value ordering heuristics, we studied three different search techniques: 1) a branch-and-bound search seeking to minimize explanation length; 2) a greedy heuristic search for a solution that returns a single explanation; 3) running multiple greedy heuristic searches (10 in these experiments), providing multiple explanations, of which the one of minimum length was returned. The simulated user chose variables randomly and values lexicographically. We considered both the task of generating corrective explanations for inconsistency (over 800 configuration sessions), and corrective explanations for value recovery (over 100 sessions) in which the user chose to restore an inconsistent value with a probability of 0.1. We report averages at intervals of 10 variables.

In Figure 2(a) and Figure 2(b) we see that minimum length explanations tend to be very expensive to compute using a branch-and-bound search, thus not a viable option in an interactive context. In Figure 2(c) and Figure 2(d) we plot average explanation lengths. The greedy approaches, with the possible exception of that based on the random variable ordering, find almost minimum length corrective explanations using a fraction of the search cost of branch-and-bound.

The best approach of those studied was based on minimum domain/degree. For the inconsistency explanations experiment the average length of the explanations found was 2.15 variables, for the one solution version, and 1.60 variables when we proposed the best of 10 solutions. This compares with an average optimal explanation length of 1.32 variables. The average search effort required to find a single explanation was 2,142 constraint checks. For the value recovery experiment, minimum domain/degree found explanations of length 8.22 and 7.16 for the one solution and best of 10 solutions, respectively. The average optimal length was 7.03. The search effort required to find a single explanation was 1,354 checks.

4 Future Work and Conclusion

The notion of corrective explanation is new in the domain of constraint satisfaction, where existing work on explanation is focused on conflict. Using a greedy search approach with good variable and value ordering heuristics seems an attractive option as it represents a good compromise between search cost and explanation length. Building upon the work of Junker [Junker, 2004] we plan to explore the potential for a preference-oriented relaxation-focused corrective explanation algorithm.

References

[Amilhastre *et al.*, 2002] J. Amilhastre, H. Fargier, and P. Marguis. Consistency restoration and explanations in dynamic CSPs – application to configuration. *Artificial Intelligence*, 135:199–234, 2002.

[Junker, 2004] U. Junker. QUICKXPLAIN: Preferred explanations and relaxations for over-constrained problems. In *Proceedings AAAI-04,*, pages 167 – 172, 2004.

[Reiter, 1987] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.