Robustness and Stability in Constraint Programming under Dynamism and Uncertainty*

(Extended Abstract)

Laura Climent[1], Richard J. Wallace[1], Miguel A. Salido[2], and Federico Barber[2]

[1] Insight Center for Data Analytics, University College Cork, Ireland
[2] Instituto de Automática e Informática Industrial, Universitat Politècnica de València, Spain
{laura.climent,richard.wallace}@insight-centre.org,
{msalido,fbarber}@dsic.upv.es

1 Introduction

Because of the dynamism and uncertainty associated with many real life problems, these problems and their associated Constraint Satisfaction Problem (CSP) models may change over time; thus an earlier solution found for the latter may become invalid. Moreover, many approaches proposed in the literature cannot be applied when the required information about dynamism is unknown ([9], [4], [5], [11], [10], etc.). This fact has motivated us to consider dynamic situations where, in addition, only limited assumptions about changes can be made. Our analysis focuses on CSPs with ordered and discrete domains that model problems for which the order over the elements of the domain is significant. In these cases, a common type of change that problems may undergo is restrictive modifications over the bounds of the solution space. A discussion of these assumptions, their motivation and real life examples can be found in [3].

In this paper, we present an algorithm that meets the goal of combining solution *stability* (meaning that solutions can often be repaired using other similar values if they undergo a value loss) and robustness (meaning that solutions have a high likelihood of remaining solutions after changes). The desireability of this combination of features was noted in the survey [8]. Furthermore, in this work we have extended both concepts to apply to the type of CSP analyzed. The paper is organized as follows. Section 2 presents the new conceptions of robustness and stability. Section 3 describes our approach for finding solutions that simultaneously meet both these criteria. In Section 4 we present some experimental results. Section 5 gives conclusions.

2 Extending Robustness and Stability Concepts

Given CSPs with ordered domains, where only limited assumptions are made about changes in the problem that are related to their inherent structure, it is reasonable to

* This is a summary of the paper: L. Climent, R. J. Wallace, M. A. Salido, and F. Barber. Robustness and Stability in Constraint Programming under Dynamism and Uncertainty. Journal of Artificial Intelligence Research, 49:49-78, 2014.
http://www.jair.org/media/4126/live-4126-7626-jair.pdf

B. O'Sullivan (Ed.): CP 2014, LNCS 8656, pp. 923–927, 2014.

assume that the original *bounds* of the solution space (delimited by the domains and constraints of the CSP) can only be restricted or relaxed, even if this does not cover all possible changes. Note that the possibility of solution loss only exists in the restrictive case. For this reason, we specialize the definition of robustness as follows.

Definition 1. *The most robust solution of a CSP with ordered domains without detailed dynamism data is the solution that maximizes the distance from all the dynamic bounds of the solution space.*

In addition, we can define the notion of stability more precisely in this framework because it is possible to define a more specific notion of closeness between two solutions thanks to the existent order over domain values than the one introduced in [6]. Here, we use the Manhattan distance ($\sum_{i=1}^{n} |s1_i - s2_i|$, where $s1$ and $s2$ are solutions).

Definition 2. *Given an order relationship over the values of a set of solutions, a solution $s1$ is more stable than another solution $s2$ iff, in the event of a change that invalidates them, there exists an alternative solution to $s1$ with lower Manhattan distance than the Manhattan distance of any alternative solution to $s2$.*

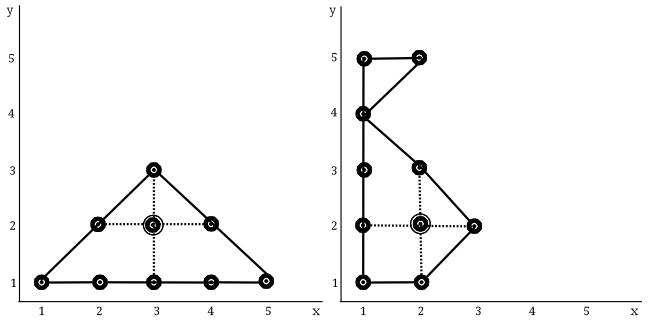
3 Searching for Robust and Stable Solutions

In this section we explain our strategy for searching for solutions that combine robustness and stability according to the definitions of Section 2. The measure of the distance from the dynamic bounds of the solution space (required for the robustness measurement) is not always obvious or easy to derive, since the constraints of the CSP may be extensionally expressed. However, some deductions about minimum distances to the bounds can be made based on the feasibility of the neighbours of a solution. This idea is first motivated with a very simple example and then is formalized.

Example 1. Figure 1 shows two solution spaces (composed by the variables x and y) whose dynamic bounds are marked by contiguous lines. The most robust solutions according to Definition 1 are highlighted. Note that there are two contiguous feasible neighbours on both sides of each assignment (discontinuous lines).

From Example 1, we conclude that *we can only ensure that a solution s is located at least at a distance d from a bound in a certain direction of the n-dimensional space if all the tuples at distances lower or equal to d from s in this direction are feasible*. Therefore, the number of feasible contiguous surrounding neighbours of the solution is a measure of its robustness and also of its stability. Because if the value assigned to a variable has at least one feasible neighbour value, then this variable is repairable.

Let $\mathcal{N}_k$ denote the neighbourhood of feasible contiguous surrounding values for a given value (assuming a specific variable and a feasible partial or complete assignment) at distance not greater than k in increasing, or decreasing, or both directions with respect to the order relationship. For the general case of CSPs with ordered domains, the desirable goal is to find contiguous surrounding feasible neighbours on both sides. For instance, in Figure 1(b), considering the partial assignment $\{x = 2\}$, $\mathcal{N}_k = \{1, 3\}$ for



(a) Convex Space (b) Non-convex Space

Fig. 1. Most robust solutions for different solution spaces

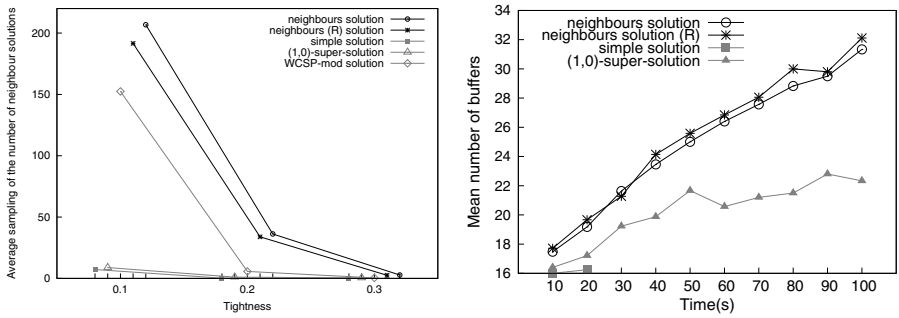
the value 2 in the y axis (for any k value). However, for some problems it is important to consider the feasibility of neighbours only in an increasing/decreasing order [2].

In order to find solutions with the maximum number of contiguous feasible neighbours, we implemented a Branch & Bound algorithm that maximizes the sum of the sizes of $\mathcal{N}_k$ for all the variables of the assignment s. If s is an incomplete assignment, we calculate the maximum size of $\mathcal{N}_k$ of the analyzed variable, for each value of its feasible domain with respect s. Note that this objective function is an upper bound on the final total number of feasible contiguous neighbours of the solution. For the inference process, we developed an extension of the well-known Generalized Arc Consistency (GAC) that checks the feasiblity of both the analyzed value and its $\mathcal{N}_k$. An earlier description of this search algorithm can be found in [1]. For the highlighted solutions of both Figures 1(a) and 1(b) the objective function is equal to four for $k \geq 1$ (every value assigned to each solution has two contiguous neighbours on both sides).

4 Experimental Results

In this section, we describe a very limited part of the evaluation that was carried out. Experiments were run on an Intel Core i5-650 Processor (3.20 Ghz) with a time cutoff of 100 seconds. Figure 2 shows, for a fixed $k = 1$, the solutions obtained by our search algorithm ("neighbour solutions" and "(R)" is a variant). We also evaluated an ordinary CSP solver ("simple solutions") and two other methods: a WCSP modeling technique [3] ("WCSP-mod solutions") and the (1, 0)-super-solutions [6].

Figure 2(a) shows an analysis of robustness as a function of the tightness of the constraints (ratio of the number of forbidden assignments to the total number possible) of random CSPs with 25 variables, domain size 30 and 200 binary constraints. Here we made 500 random changes to each solution by increasing or decreasing two of their values and then checking if they were still solutions. The more neighbours that are not solutions of the CSP, the higher the likelihood of the solution becoming infeasible after changes over the bound/s. Note that our search algorithm outperformed the other approaches, specially for lower tightness. At higher tightness values, there is a lower probability of neighbour solutions (i.e. *all* solutions are located close to the bounds).



(a) Robustness analysis based on the tightness (b) Stability analysis based on computing time

Fig. 2. Robustness and stability analysis for random CSPs and scheduling problems

Figure 2(b) shows a stability analysis based on the computing time of the algorithms (discretization of 10 seconds) of the CSPs of the "e0ddr1" scheduling benchmark [7]. The mean number of buffer times is a measure of the stability because the start time of a task with an associated buffer can be delayed, for instance when there are delays in previous tasks. The most noteworthy aspect is that our search algorithm clearly outperformed the other approaches, specially when the computation time cutoff was higher.

5 Conclusions

In this paper we extend the concept of robustness and stability to deal with CSPs with discrete and ordered domains where only limited assumptions can be made about changes in these problems due to a lack of detailed dynamism information. Furthermore, we present a new search algorithm that combines criteria for both robustness and stability in this framework by searching for a solution that maximizes the sum of contiguous feasible surrounding neighbours at distances of k or less from the values of the solution. The obtained solutions have a higher probability of remaining valid after possible future restrictive changes over the constraints and domains of the original problem (robustness criterion), and they also have a high number of variables that can be easily repaired with a value at a distance lower or equal to k if they undergo a value loss (stability criterion).

Our experiments showed that our search algorithm outperformed other approaches that need only limited information about dynamism, with respect to robustness and stability as we have defined them, in cases where there were real differences in the robustness of solutions that could be obtained. The latter occurs when the problem is not so constrained that there are only a few valid solutions.

Acknowledgements. The first author is supported by the research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289. The work was also supported by the FPU fellowship (Min. de Ciencia e Innovación, Spain) and the project TIN2013-46511-C2-1 (pending).

References

1. Climent, L., Wallace, R., Salido, M., Barber, F.: An algorithm for finding robust and stable solutions for constraint satisfaction problems with discrete and ordered domains. In: 24th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2012), pp. 874–879 (2012)
2. Climent, L., Wallace, R.J., Salido, M.A., Barber, F.: A constraint programming approach to solve scheduling problems under uncertainty. In: Workshop on Constraint Satisfaction Techniques for Planning and Scheduling Problems (COPLAS 2013) in ICAPS 2013, pp. 28–37 (2013)
3. Climent, L., Wallace, R.J., Salido, M.A., Barber, F.: Finding robust solutions for constraint satisfaction problems with discrete and ordered domains by coverings. In: Artificial Intelligence Review (AIRE) (2013), doi:10.1007/s10462-013-9420-0
4. Fargier, H., Lang, J.: Uncertainty in constraint satisfaction problems: A probabilistic approach. In: Moral, S., Kruse, R., Clarke, E. (eds.) ECSQARU 1993. LNCS, vol. 747, pp. 97–104. Springer, Heidelberg (1993)
5. Fargier, H., Lang, J., Schiex, T.: Mixed constraint satisfaction: A framework for decision problems under incomplete knowledge. In: Proceedings of the 13th National Conference on Artificial Intelligence (AAAI 1996), pp. 175–180 (1996)
6. Hebrard, E.: Robust Solutions for Constraint Satisfaction and Optimisation under Uncertainty. PhD thesis, University of New South Wales (2006)
7. Sadeh, N., Fox, M.: Variable and value ordering heuristics for the job shop scheduling constraint satisfaction problem. Artificial Intelligence 86(1), 1–41 (1996)
8. Verfaillie, G., Jussien, N.: Constraint solving in uncertain and dynamic environments: A survey. Constraints 10(3), 253–281 (2005)
9. Wallace, R., Freuder, E.: Stable solutions for dynamic constraint satisfaction problems. In: Maher, M.J., Puget, J.-F. (eds.) CP 1998. LNCS, vol. 1520, pp. 447–461. Springer, Heidelberg (1998)
10. Walsh, T.: Stochastic constraint programming. In: Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002), pp. 111–115 (2002)
11. Yorke-Smith, N., Gervet, C.: Certainty closure: Reliable constraint reasoning with incomplete or erroneous data. Journal of ACM Transactions on Computational Logic (TOCL) 10(1), 3 (2009)