

Summary of the Thesis: “Soft Constraints in Concurrent Constraint Programming: Design and Implementation”

Alberto DELGADO-ORTEGÓN and Jorge Andrés PÉREZ-PARRA

Thesis Information

- Work carried out within the AVISPA research group, and supported by COLCIENCIAS (the Colombian Agency for Science and Technology).
- Supervisor: Prof. Camilo RUEDA. Dean of the Department of Sciences and Engineering of Computing, Pontificia Universidad Javeriana, Cali, Colombia.
- Submitted: November 8, 2005. Defended: November 22, 2005.
- Awarded for the achievements obtained, February 2006.

A wide variety of problems can be modeled as Constraint Satisfaction Problems (or CSPs). In several scenarios, to describe a problem in terms of some mandatory conditions and the available resources to satisfy them turns out to be extremely convenient. For instance, imagine some of the conditions that the program chair of a conference must consider when assigning reviewers for submissions: every paper must be revised by a fixed number of reviewers, both reviewers and authors should belong to different institutions, there is a maximum of papers a program committee member can revise, among other similar conditions. A possible modelling approach for this problem would be, for instance, to define n different *variables* for each submission, each one representing a selected reviewer (that is, each paper is reviewed by n persons). That is, the possible values for such variables (their *domain*) would be all available reviewers. Finally, a satisfactory *solution* would be an assignment of papers to reviewers in such a way that *every constraint* in the problem is met.

However, there are situations that can not be faithfully represented by using this *classical* formulation of CSPs. For instance, those problems stating preferences or costs over the constraints, or those asserting contradictory constraints in such a way that it is impossible to find a suitable solution for the problem. Human descriptions of preferences are specially prone to be contradictory constraints. Returning to our paper reviewing example: it might be the case where the program chair must review more papers than it was originally planned, thus violating the constraint stating a maximum number of submissions per reviewer. Also, the program chair may be tempted to decide that some constraints are more important than others. In these cases, sometimes it is necessary to consider only a subset of such constraints or to satisfy them in an incomplete manner in order to obtain a solution.

Multiple research efforts (both theoretical and practical) have been carried out in order to shed light in modelling and solving this kind of “relaxed” constraints, commonly referred to as *soft constraints*. The goal of the theoretical proposals has been to devise a *generic* framework that associate preferences and similar notions to several kinds of CSPs. Therefore, properties and generic results obtained from using such frameworks apply to diverse contexts. Unfortunately, few works have tried to take to the practical side the main notions and properties derivable from theoretical frameworks. Indeed, some prototypes implementing toy examples of the techniques are available, however, none of the theoretical frameworks has been used in producing a *programming technique* on its own. In contrast, the practical works have proposed ad-hoc solutions that albeit useful, are not applicable to other contexts. As a consequence, properties that can be derived from such solutions are only applicable to restricted contexts.

Our B.Sc. thesis focuses on one formal framework for handling soft constraints. More precisely, we study a framework based on a mathematical structure known as *semiring*. Intuitively, the framework gives

a mathematical account of the basic elements of a CSP (i.e., variables and constraints) while formally associating some elements representing preferences (known as *valuations*) and the constraints in the CSP. In this context, a semiring (an ordered set plus two operations) thus represents the set of valuations and the operations over them. This *semiring-based constraints* framework was originally proposed by Bistarelli, Rossi and Montanari in [1] and later consolidated in Bistarelli's doctoral dissertation [2].

From a global perspective, the aim of our work is to devise tools that enable Mozart [3], a concurrent constraint programming language, to effectively use soft constraints statements in its programs. A brief description of solving processes for a CSP in Mozart follows. In Mozart solving processes are divided in two phases: *propagation* and *distribution* (or *labelling*). In propagation the goal is to remove those values in variables' domains that are inconsistent wrt each constraint. Solving processes for each constraint are governed by a concurrent agent (*propagator*) that waits for events that may lead to further pruning in variables' domains. Since it is possible that after propagation the CSP is not fully solved (i.e., that every variable's domain has a single value), it is necessary to *distribute* some (unsolved) variables in the CSP, including additional constraints representing the choice of a particular value in those variables. Such new constraints must be carefully chosen so that they induce further propagation processes. As a result, propagation and distribution complement to each other to solve a CSP.

The choice of Mozart as base language is supported by the consolidated expertise our research group has in using and extending such a language and, in general, in the theory and practice of the concurrent constraint programming (CCP) model [9]. Therefore, the idea of including soft constraints tools in Mozart is not only convenient but, given the outlined background, is also a relevant research topic. From the harmonic integration of these tools within the concurrent model of Mozart, a straightforward inclusion of soft constraints statements in real-life constraint problems would be possible.

We follow a two-fold approach to conduct this study. Firstly, we propose a set of soft constraints for Mozart. An initial subject of research consists in selecting the most appropriate mechanisms for including soft constraints in Mozart. In this issue we consider two alternatives. While the first one is based on exploiting the functional approach of Mozart for implementing semiring operations, the second proposes the implementation of low-level propagators, taking advantage of the standard (low-level) extension tools provided by the language. *A priori*, both approaches present concrete advantages and disadvantages wrt the other. In [4] we evaluate prototype implementations of both approaches and conclude that the propagator approach is more convenient for our purposes. An influential factor in this choice resides in the fact that propagators are highly compatible with Mozart's computation model. This not only allows for transparent integration between soft and classical constraints in Mozart programs, but also ensures a significant level of efficiency.

Extended capabilities of the propagator approach for soft constraints are studied in [5]. There, some new propagators enforcing additional soft constraints are presented and evaluated. One of the most remarkable features of this extended implementation is an economical approach for storing valuations associated with a soft CSP. In such a scheme, only a value for each constraint and a value valid for the whole problem need to be explicitly saved. These values are the main criteria when discarding values from variables' domains, and they are only stored when a solution is found. This *implicit* approach for valuation manipulation is the source of the good performance of our implementation as well as constitutes a distinguishing feature of our work wrt some related efforts.

Development in these extended capabilities is carried out in parallel with a conservative modification to the original semiring-based model for soft constraints. This modification is required as the way of calculating valuations, using the two kinds of parameters mentioned before, is quite different from the calculation defined in the model (where every tuple representing a constraint has explicitly associated a valuation). We show how our valuation scheme is just a specific instance of the original model. An initial version of our formal modification is presented in [6]; a more complete description is given in our thesis document ([8]).

The second major strand of our work is the study of a theoretical proposal for *abstracting* soft constraints. In certain cases, this proposal provides hints that may help to speed up solving processes. As such, two challenges are involved in this part of our work: (i) the study of applicability of the proposal and (ii)

the relationship of the implicit valuation scheme with the abstraction results. We develop an iterative procedure that implements this proposal for Mozart. For doing so, we define a mechanism for implementing soft constraints only using classical constraints. The idea is to define (classical) constraints that *simulate* the behaviour of soft ones. We find that these special constraints, so-called *classical counterparts*, are particularly efficient when soft CSPs have a large number of soft constraints. By means of an example, we show how to prove the equivalence (in terms of propagation power) between a soft constraint and a classical counterpart. We also study the influence parameters for the iterative procedure have in overall performance. Our results in this matter are condensed in [7]. Evaluation in real-world problems is included there.

The final part of our research consists in the analysis of the developed implementations wrt a category of CSPs. Such a category is formed by four kinds of CSPs, which differ in the number of soft constraints they include. In addition for the results suggesting the use of the abstraction procedures in problems with a considerable number of soft constraints, we find that the soft propagators are more convenient when only a few soft constraints are present in the problem. This basic observation comes from the extensive performance evaluations we carry out. Hints wrt the use of soft constraints in existing constraint programming applications are also given. Most of these applicability results have not been published yet; they are also condensed in our thesis document [8].

Summing up, the models and tools proposed in this thesis contribute to the state of the art in constraint processing by giving a very concrete idea of the applicability of soft constraints techniques in concurrent constraint programming environments. We propose a set of general guidelines for soft constraint implementation and take them to the practical side in a particular programming language (Mozart). These guidelines are inherently efficient and also general: every notion proposed is independent from Mozart's capabilities. As a matter of fact, the first author is currently participating in a research project which intends to implement our soft constraints ideas in the generic constraint library GECODE [10]. Since our implementations directly emerge from consolidated theoretical results they inherit a great deal of support.

References

- [1] S. Bistarelli, U. Montanari, and F. Rossi. Constraint Solving over Semirings. In *Proc. IJCAI95*, San Francisco, CA - USA, 1995. Morgan Kaufman.
- [2] Stefano Bistarelli. *Semirings for Soft Constraint Solving and Programming*. Number 2962 in Lecture Notes in Computer Science. Springer-Verlag, 2004.
- [3] Mozart Consortium. The Mozart Programming Language. <http://www.mozart-oz.org>.
- [4] Alberto Delgado, Carlos A. Olarte, Jorge Andr es P erez, and Camilo Rueda. Implementing Semiring-Based Constraints using a Concurrent Constraint Programming Language. In Stefano Bistarelli and Francesca Rossi, editors, *Proceedings of the Sixth International Workshop on Preferences and Soft Constraints (Soft 2004)*, October 2004. Held in conjunction with the Tenth International Conference on Principles and Practice of Constraint Programming (CP 2004).
- [5] Alberto Delgado, Carlos A. Olarte, Jorge Andr es P erez, and Camilo Rueda. Implementing Semiring-Based Constraints Using Mozart. In Peter Van Roy, editor, *Multiparadigm Programming in Mozart/Oz: Extended Proceedings of the Second International Conference MOZ 2004*, volume 3389 of *Lecture Notes in Computer Science*, pages 224–236. Springer-Verlag, 2005.
- [6] Alberto Delgado, Carlos A. Olarte, Jorge Andr es P erez, and Camilo Rueda. Semiring-based Fuzzy Constraints in Concurrent Constraint Programming. In Juan F. D iaz, Camilo Rueda, and Antal A. Buss, editors, *Proceedings of CLEI2005*, pages 735–746, October 2005. ISBN 958-670-426-2.
- [7] Alberto Delgado, Jorge Andr es P erez, and Camilo Rueda. Implementing an Abstraction Framework for Soft Constraints. In Jean-Daniel Zucker and Lorenza Zaitta, editors, *Abstraction, Reformulation, and Approximation, Proceedings of the 6th International Symposium SARA 2005*, volume 3607 of *Lecture Notes in Artificial Intelligence*, pages 60–75. Springer, July 2005.
- [8] Alberto Delgado and Jorge Andr es P erez. An alisis e Implementaci on de Mecanismos de Restricciones D ebiles para Programaci on Concurrente por Restricciones, November 2005. BSc. Thesis - Pontificia Universidad Javeriana, Cali.
- [9] V. Saraswat, M. Rinard and P. Panangaden. The semantic foundations of concurrent constraint programming. In *POPL '91*, pages 333–352, jan 1991.
- [10] Christian Schulte. GECODE: Generic Constraint Development Environment. <http://www.gecode.org>.