AI Communications 17 (2004) 171–174
IOS Press


Thesis

# A Generic, Collaborative Framework for Interval Constraint Solving *

Antonio J. Fernández
*Dpto. Lenguajes y Ciencias de la Computación, ETSII, University of Málaga, D.P. 29071, Spain*
*E-mail: afdez@lcc.uma.es*

**Abstract.** The paper abstracts the contents of a PhD dissertation entitled *A Generic, Collaborative Framework for Interval Constraint Solving* which has been recently defended. This thesis presents a generic framework for defining and solving interval constraints on any set of domains (finite or infinite) that are lattices. This framework combines a number of characteristics desirable in any constraint system such as *transparency*, on both constraints and computation domains (i.e., it follows a glass box approach where new constraints can be user defined and new, possibly compound, constraint domains can be constructed from existing domains using lattice combinators), *cooperativity* (so that different solvers, possibly on distinct domains, can communicate and hence, cooperate in solving a problem) and *genericity* (i.e., it can be applied on any computation domain with lattice structure).

Keywords: Constraint, lattice, cooperation, propagation, indexical

## 1. Abstract of PhD dissertation

The basic idea in Constraint Logic Programming (CLP) [16] is to replace the classical logic programming unification by constraint solving on a given computation domain. This idea gave rise to the CLP($\mathcal{X}$) schema [17] where $\mathcal{X}$ is a computation domain over which constraints are solved. Different instances of $\mathcal{X}$ (e.g., with reals, integers, sets, Booleans, etc.) generate different instances of the CLP($\mathcal{X}$) schema, and the computation domain determines the nature of the constraints and their solvers; this means to have distinct constraint solving methods for different computation domains. In practice, constraint problems are often not specific to any particular system domain and thus their formulation has to be artificially adapted to fit a given solver.

Most constraint solvers, called *black box* solvers, have the control fixed by the system. This approach enables very efficient implementations and can provide practical tools for the common constraint applications. However, such black box solvers lack adaptability for use in solving non-standard problems. To overcome this lack of flexibility, some constraint systems provide *glass box* constraints [1,15]. These allow the user to define new constraints for specific applications. However, these solvers are often restricted to just the built-in domains, usually the integers. This restricts the flexibility of this approach since as already discussed, in practice, problems are heterogeneous and often have a natural formulation which uses domains other than the built-in domains.

Moreover, many problems are most naturally expressed using heterogeneous constraints over more than one domain and there exist constraints defined on multiple domains that require the collaboration of distinct domains by sending and receiving information to and from another different domain (e.g., $w = x > y$).

As consequence, in existing CLP systems the formulation of real problems has to be artificially adapted to a single domain (i.e., one of the supported by the system).

This thesis proposes a generic and cooperative schema for CLP(Interval($\mathcal{X}$)) where $\mathcal{X}$ is any computation domain with lattice structure. This schema, based on interval lattices, is a general framework for

*This work has been partially supported by the projects TIC2001-2705-C03-02, and TIC2002-04498-C05-02 funded by both the Spanish Ministry of Science and Technology and FEDER.

interval constraint satisfaction and interval solver cooperation on domains with lattice structure independently of its cardinality. This proposal assures a complete glass box setting on which both constraints and domains as well as the intended propagation and cooperation mechanisms among constrained variables can be easily defined from the user level. The main body of the thesis consists of a formal specification of this schema.

This thesis presents the following main results:

(A) A comprehensive comparison, on both the efficiency and certain aspects of the expressiveness, of a number of constraint systems. This comparison, done over the Boolean and the finite domains, illustrates main differences between existing constraint systems.

(B) A proposal of a constraint satisfaction framework for CLP(Interval($\mathcal{X}$)). This proposal is done by describing the whole process of interval constraint solving on any domain with lattice structure, detailing separately the processes of interval propagation and interval branching. One of the advantages of the proposal is that monotonicity of constraints is implicitly defined in the theory. Also, the thesis presents a statement of a number of interesting properties that, subject to certain conditions, are satisfied by any instances of the schema. Moreover, the thesis shows that many existing constraint systems satisfy these conditions and points out other non-trivial interesting instances of the framework.

(C) A novel proposal to extend the schema for CLP(Interval($\mathcal{X}$)) in order to enable solver cooperation by allowing the information flow between distinct computation domains. This enables the mix of different instances of the schema, e.g., well known instances such as CLP(Interval($\Re$)), CLP(Interval(*Integer*)), CLP(Interval(*Set*)), and CLP(Interval(*Bool*)) among others, and new instances resulting from user defined domains or even from the combination of existing domains in the way CLP(Interval($\mathcal{X}_1 \times \cdots \times \mathcal{X}_n$)). Therefore, $\mathcal{X}$ may be instantiated to a set of lattice structure computation domains and the corresponding CLP(Interval($\mathcal{X}$)) schema allows multiple flexibility in the definition of (probably user defined) domains in $\mathcal{X}$ and interaction between them.

(D) By means of a prototype implementation, the thesis shows that a single system based on the proposed CLP(Interval($\mathcal{X}$)) schema may provide support for classical interval constraint satisfaction and optimization as well as for interval solver cooperation over a multiple set of computations domains. Moreover the system is a glass box approach from a double perspective since the user can define not only new constraints and the intended propagation mechanism but also new domains on which constraints can be solved and the expected cooperation mechanism between all the (user or system defined) computation domains.

## 2. Structure

This thesis is composed of 4 parts relatively independent.

`Part I, Introduction and background`, is composed of two chapters. The first chapter motivates the generic, cooperative and transparent system for interval constraint solving described in this thesis. This is done by discussing the limitations of the current instances of CLP and by showing how these are solved in our proposal. The second chapter provides a general overview of the basis over which CLP is founded.

`Part II, the comparative framework`, composed of Chapter 3, describes the election of an adequate approach to support the generic schema of the thesis. There are two key reasons for adopting CLP technology for solving a problem. The first is its expressiveness enabling a declarative solution with readable code which is vital for maintenance, and the second is the provision of an efficient implementation for the computationally expensive procedures. However, CLP systems differ significantly both in how solutions may be expressed and the efficiency of their execution, and it is important that both these factors are taken into account when choosing the best CLP system for a particular application. Also, among the domains of CLP, the finite domain (FD) is one of the most studied since a lot of problems involve variables ranging in discrete domains.

Currently, there are several techniques to support constraint solving on FD in the CLP systems. The thesis includes a comparison on the efficiency of a number of CLP systems in the setting of finite domains as well as two specific aspects of their expressiveness (those concerning reification and meta-constraints). This comparison involves eight systems that, strictly speaking, are glass boxes since they allow user defined constraints in a more or less clear way. The comparison illustrates differences between the systems, indicating their particular strengths and weaknesses and helps in the choice of the best technique for building our generalized framework for interval constraint solving.

`Part III, the theoretical framework`, is composed of three chapters dedicated to the formalization of the generic and collaborative theoretical framework.

This part begins by constructing a generic theoretical framework (called *the basic framework*) to propagate interval constraints on any domain with lattice structure (Chapter 4). From the comparison done previously, the transparent approach called *indexical* [1] was chosen due to its flexibility, its simplicity and its performance. This approach, defined for the FD, was generalized for interval constraint propagation to domains with lattice structure. The thesis provides the theoretical foundations for this framework, a schematic procedure for the operational semantics, and numerous examples illustrating how it can be used over both classical and new domains. Also, the thesis shows how lattice combinators can be used to generate new domains and, hence, new constraint solvers for these domains. As most of the existing domains are lattices, this framework provides support for them.

In the CLP(Interval($\mathcal{X}$)) schema for constraint propagation devised previously, the interval constraint solvers are each based on the same generic solver so that they are completely independent from each other and there is no provision for any cooperation between them. Therefore, Chapter 5 extends the basic generic theoretical framework to enable solver cooperation and allow information to flow between different computation domains. This is done by means of a novel technique allowing constraint operators to be defined over multiple domains enabling thus a one-way communication channel between different domains. To allow for a two way channel the generic concept of *high level constraint* was also defined. As consequence, the different solvers can communicate and hence, cooperate in solving a constraint satisfaction problem (CSP).

Also, often constraint propagation is not enough to solve completely a CSP and some additional strategy must be applied for it. For this reason, Chapter 6 extends the cooperative CLP(Interval($\mathcal{X}$)) schema for complete constraint solving. This was done by proposing a parameterized CLP(Interval($\mathcal{X}$)) schema for constraint branching that (with suitable instantiations of the parameters) can solve completely CSPs defined via interval constraints on any set of domains $\mathcal{X}$ with lattice structure. The resulting schema allows classical constraint solving as well as constraint optimization.

This part is also dedicated to study a number of interested properties of the schema, to develop numerous examples to show the declarativeness of the generic setting, and to treat other issues such as constraint monotonicity, high level constraints and combination of domains.

`Part IV, the practical framework`, is composed of one chapter devoted to describe *clp*($\mathcal{L}$), an interval constraint logic programming language that allows constraint solving on any set $\mathcal{L}$ of lattices that is based on the schema described in the thesis. This part also gives an outline of a prototype implementation and provides as number of non-standard examples to show the declarativeness and flexibility of the resulting system. This prototype implementation demonstrates that a single system, based on our CLP(Interval($\mathcal{X}$)) schema, is enough to provide support for multiple domains, solver cooperation, solver satisfaction and solver optimization integrated in a glass box setting on both constraints and domains.

The thesis terminates with a chapter that briefly summarizes the results and gives major directions for future works and improvements.

## 3. Further information

The main results achieved in the thesis have been widely published. The original motivations of our work (Part I) were initially presented in [2] and Part II was almost integrally published in the *Constraints* journal [11]. Moreover, a recently-published paper [18] examines the positive and negative aspects of the benchmarking process performed in [11]. Also, some preliminary results of Part II were presented in [5,4,8].

With respect to Part III, some results were published in [6,7,9,10,12,13]. Also, a journal has published most of the work done in this Part [14].

Part IV is available, as user manual, in [3].

After providing an insight to the thesis contents, we will finish detailing some further information on the thesis:

*Author*: Antonio J. Fernández.

*Title*: A Generic, Collaborative Framework for Interval Constraint Solving.

*Language of the thesis*: Written in English.

*Promotor*: Patricia M. Hill (Leeds University).

*Evaluation Committee*: Fernando Orejas, María Alpuente, Francisco López-Fraguas, Francisco Bueno and Ernesto Pimentel.

The thesis, written in English, and further information, are available in the following URL:
http://www.lcc.uma.es/~afdez/papers.html.

## Acknowledgement

The author thanks Patricia M. Hill for the improvement of the ideas applied in this thesis, making them coherent and presentable at the same time.

## References

[1] P. Codognet and D. Diaz, Compiling constraints in clp(FD), *The Journal of Logic Programming* **27**(3) (1996), 185–226.

[2] A.J. Fernández, Towards a glass-box typed CLP language, in: *Workshop on Automated Reasoning (WAR'97)*, AISB Workshop and Tutorial Series, M. Fisher, ed., Manchester, University of Manchester, 1997, pp. 7–8.

[3] A.J. Fernández, clp(L) version 0.21, user manual. Available at http://www.lcc.uma.es/∼afdez/generic, 2000.

[4] A.J. Fernández and P.M. Hill, Boolean and finite domain solvers compared using self referential quizzes, in: *Joint Conference on Declarative Programming (APPIA-GULP-PRODE'97)*, M. Falaschi, M. Navarro and A. Policriti, eds, Grado, Italy, 1997, CLEUP, pp. 533–544. Also available as Technical Report ref.97.03, School of Computer Studies, University of Leeds, January, 1997.

[5] A.J. Fernández and P.M. Hill, Finite domain solvers compared using self referential quizzes, in: *Sixièmes Journées Francophones de Programmation Logique et Programmation par Contraintes (JFPLC'97)*, Orléans, France, 1997. LIFO. (Poster Session). Rapport de Recherche.

[6] A.J. Fernández and P.M. Hill, A design for a generic constraint solver for ordered domains, in: *Types for Constraint Logic Programming (TCLP'98)*, Manchester, 1998.

[7] A.J. Fernández and P.M. Hill, A generic execution model for CLP(X), Research Report LCC-ITI 98/16, Universidad de Málaga, Departamento de Lenguajes y Ciencias de la Computación, August, 1998.

[8] A.J. Fernández and P.M. Hill, An impartial efficiency comparison of FD constraints systems, in: *4th International Conference on Principles and Practice of Constraint Programming (CP'98)*, number 1520 in LNCS, M. Maher and J.-F. Puget, eds, Pisa, Italy, Springer-Verlag, 1998, p. 468. Also available as Technical Report ref.98.18, School of Computer Studies, University of Leeds, September, 1998.

[9] A.J. Fernández and P.M. Hill, Constrant solving on lattices, in: *International Joint Conference on Declarative Programming (APPIA-GULP-PRODE'99)*, C. Meo and M. Vilares, eds, L'Aquila, Italy, 1999, pp. 105–120. Gruppo Tipografico Editoriale.

[10] A.J. Fernández and P.M. Hill, An interval lattice-based constraint solving framework for lattices, in: *4th International Symposium on Functional and Logic Programming (FLOPS'99)*, number 1722 in LNCS, A. Middeldorp and T. Sato, eds, Tsukuba, Japan, Springer-Verlag, 1999, pp. 194–208.

[11] A.J. Fernández and P.M. Hill, A comparative study of eight constraint programming languages over the Boolean and finite domains, *Constraints* **5**(3) (2000), 275–301.

[12] A.J. Fernández and P.M. Hill, Constraint propagation on multiple domains, in: *9th International Workshop on Functional and Logic Programming (WFLP'00)*, M. Alpuente, ed., Benicássim, Spain, Universidad Politécnica de Valencia, 2000, pp. 455–469.

[13] A.J. Fernández and P.M. Hill, Branching: the essence of constraint solving, in: *ERCIM Workshop on Constraints*, K. Apt, R. Barták, E. Monfroy and F. Rossi, eds, Prague, Czech Republic, Charles University/Faculty of Mathematics and Physics, 2001.

[14] A.J. Fernández and P.M. Hill, An interval constraint system for lattice domains, *ACM Transactions on Programming Languages and Systems* **26**(1) (2004), 1–46.

[15] T. Frühwirth, Theory and practice of constraint handling rules, *The Journal of Logic Programming* **37** (1998), 95–138.

[16] J. Jaffar and J.L. Lassez, Constraint logic programming, in: *14th ACM Symposium on Principles of Programming Languages (POPL'87)*, Munich, Germany, ACM Press, 1987, pp. 111–119.

[17] J. Jaffar and M. Maher, Constraint logic programming: a survey, *The Journal of Logic Programming* **19–20** (1994), 503–581.

[18] M. Wallace, J. Schimpf, K. Shen and W. Harvey, On benchmarking constraint logic programming platforms. Response to Fernandez and Hill's "A comparative study of eight constraint programming languages over the Boolean and finite domains". *Constraints* **9**(1) (2004), 5–34.