ELSEVIER

Science of Computer Programming 30 (1998) 1–2

Science of Computer Programming

Foreword

Philippe Codognet

INRIA - Rocquencourt, B.P. 105, 78 153 Le Chesnay Cedex, France

This special issue contains papers on Concurrent Constraint Programming, originating from CCP'95, the first international workshop on Concurrent Constraint Programming, that took place on 29–31 May 1995, at the University of Venice, Italy. Among the seven papers that are composing this special issue, three papers correspond to invited talks and four to papers presented at the workshop that have been selected for publication in *Science of Computer Programming*. They have been thoroughly revised after an additional reviewing process.

Concurrent Constraint Programming (CCP) has been proposed a few years ago by V. Saraswat as a new programming paradigm that can be seen as the merging and generalization of Constraint Logic Programming and Concurrent Logic Languages. It makes it possible to combine both approaches, that is, on the one hand, the ability to reason (symbolically) and to compute (numerically) on specific domains (*constraints*) and, on the other hand, the possibility to have a dynamic data-driven control of the execution flow (*concurrency*). The CCP paradigm brings new expressive power and makes it possible to study new topics emerging from the conjunction in a single framework of these two paradigms. Let us mention in particular on the theoretical side the simpler and clearer formalization of synchronization mechanisms in CCP (thanks to the use of constraints and logical operations such as implication) and, on the practical side, the ability to dynamically control the behavior of constraint solvers by using concurrent agents. The fundamental idea of Concurrent Constraint Languages is the use of constraints for defining the synchronization and control mechanisms. Therefore, several agents (processes) could communicate and synchronize through a global store where all information is added in a monotonic way through the time line. Each agent can either add a new constraint (*Tell* operation) or check if some constraint is already true in the current store (*Ask* operation), i.e., from a logical point of view, implied by it. The *Tell* operation corresponds to the classical CLP addition of a new constraint. Synchronization is achieved through a *blocking Ask* operation: if it cannot be stated whether the constraint is true or false in the current store, i.e. more information is needed to decide, then the asking agent is suspended until other concurrently running agents add (*Tell*) new constraints strong enough to decide.

PII S0167-6423(97)00005-1

Computing with continuous change by V. Gupta, R. Jagadeesan and V.A. Saraswat describes a new CCP language called HYBRID CC, which is intented to be used for modeling and programming hybrid (continuous and discrete) systems. It extends the classical CCP framework with a default construct and a temporal construct in order to provide a declarative and expressive language.

Partial order and contextual net semantics for atomic and locally atomic CC programs by F. Bueno, M. Hermenegildo, U. Montanari and F. Rossi studies two concurrent semantics for CCP with atomic tell operations. One is based on simple partial orders of computation steps, while the other is based on contextual nets.

Newton: Constraint programming over nonlinear constraints by Pascal Van Hentenryck, Laurent Michel and Frédéric Benhamou presents the NEWTON constraint programming language over nonlinear real constraints. The paper also presents several programming examples and applications to numerous problems in science and engineering.

A hierarchy of constraint systems for data-flow analysis of constraint logic-based languages by R. Bagnara is interested in the data-flow analysis of CCP in the abstract interpretation framework. The basic idea is to use constraint domains as abstract domains for the properties to analyse.

GOFFIN: Higher-order functions meet concurrent constraints by M.M.T. Chakravarty, Yike Guo, Martin Köhler and H.C.R. Lock merges Higher-Order Programming and CCP in the GOFFIN programming language. It is an extension of the purely functional language Haskell with a coordination layer based on concurrent constraints.

Constraint-based protocols for distributed problem solving by U.M. Borghoff, R. Pareschi, F. Arcelli and F. Formato is not into the mainstream area of CCP but addresses a related field. It explores an approach to decompose problems into subproblems to be solved by interacting, cooperative software agents.

A distributed arc-consistency algorithm by T. Nguyen and Y. Deville discusses the distributed implementation of constraint satisfaction. It presents a distributed arc-consistency algorithm for constraints over finite domains.

Last but not least, I would like to thank the members of the program committee for their help in the reviewing and selection process, and the conference organizer, Agostino Cortesi from Università di Venezia, for making CCP'95 a cheerful and enjoyable event.