TLP **18** (5–6): 722–724, 2018. © Cambridge University Press 2018
doi:10.1017/S1471068418000406

Past and present (and future) of parallel and distributed computation in (constraint) logic programming

FERDINANDO FIORETTO
University of Michigan, Ann Arbor, MI 48109, USA
(e-mail: fioretto@umich.edu)

ENRICO PONTELLI
New Mexico State University, Las Cruces, NM 88003, USA
(e-mail: epontell@cs.nmsu.edu)

submitted 05 July 2018; revised 07 July 2018; accepted 08 July 2018

1 Introduction

Declarative languages offer unprecedented opportunities for the use of parallelism to speed up execution. A declarative language, being not procedural, removes the need to perform operations in a strict order and reduces the number of dependencies among operations, thus opening the doors for concurrent execution. The potential for transparent exploitation of parallelism in logic programming emerged almost immediately with the birth of the paradigm (Pollard 1981).

An extensive literature has been developed exploring issues like automated parallelization of logic programs, the use of logic programs to describe parallel and distributed computations, and logic programming models to capture concurrency and facilitate the development of provably correct concurrent applications. The literature on parallel execution of logic-based languages has extensively explored many of the issues related to exploitation of parallelism in traditional execution models of languages like Prolog (e.g., Gupta *et al.* 2001) as well as execution models of more recent logic-based paradigms, including Satisfiability Modulo Theory (Hyvarinen and Wintersteiger 2018)and Answer Set Programming (Dovier *et al.* 2018). After over 30 years of research in these domains, the state of the art has reached a stage where technologies are highly complex and sophisticated, and applications are plentiful. Yet, the continuous development of novel architectures (e.g., the onset of GPU-based computing; the widespread use of simple inter-connected devices, like Arduino and Raspberry Pi), the appearance of new domains and potential applications (e.g., big data), and the developments in novel logic programming languages and paradigms are creating new research opportunities and fueling new ideas and developments.

2 The papers

The goal of this special issue is to provide a three-fold perspective on research at the junction between parallel and distributed computation and (constraint) logic programming:

(1) Well-thought assessments of the state of the art (e.g., in the form of well organized surveys).
(2) Cutting-edge coverage of new developments (e.g., novel execution models, innovative systems, and implementations).
(3) New research directions, offering clear motivations, new perspectives, and solid foundations for other researchers to build upon.

The following is a brief summary of the papers that have been accepted as part of this special issue—organized in two groups: *Survey Papers* and *Original Contributions*.

2.1 Survey papers

The special issue presents three outstanding survey papers that explore the issue of parallelism and concurrency in three core domains: *Constraint Solving*, *Constraint Handling Rules,* and *Datalog-based Big Data*.

The paper by Gent, McCreesh, Miguel, Moore, Nightingale, Prosser, and Unsworth, titled *A Review of Literature on Parallel Constraint Solving,* provides a thorough review of the techniques explored to parallelize the process of constraint solving, with a focus on the use of multi-core platforms and an eye on challenges and opportunities that can be explored in the future. The paper categorizes existing work into parallel consistency and propagation, parallelizing the search process, multi-agent search, and portfolios.

The paper by Gent *et al.* is nicely complemented by the survey by Thom Früwirth, titled *Parallelism, Concurrency and Distribution in Constraint Handling Rules: A Survey.* This survey provides an overview of concurrent, parallel, and distributed semantics for the Constraint Handling Rules framework. On the more practical side, the survey explores parallel implementations that have been proposed alongside with meaningful examples and benchmarks. Additionally, it reviews concurrency models that have been encoded in Constraint Handling Rules to get a better understanding of them and sometimes to extend them. The survey also identifies common topics and issues that lead to open research questions.

The third survey moves into the world of Big Data. The paper by Condie, Shkapsky, Das, Interlandi, Yang, and Zaniolo, titled *Scaling-Up Reasoning and Advanced Analytics on BigData* examines the difficulties and reviews solutions for challenges at the language level and at the system level arising when creating an efficient and scalable systems using logic to express queries and reasoning. The survey explores extensions of Datalog suitable to enhance performance and scalability on distributed computing platforms and multi-core architectures, including platforms like Apache Spark.

2.2 Original papers

The special issue includes a number of original papers that explore different dimensions of parallelism and concurrency in different constraint and logic-based paradigms.

The work by Calegari, Denti, Mariani, and Omicini, titled *Logic Programming as a Service* takes a novel approach to concurrency and distribution, by introducing the notion of Logic Programming as a Service. The Logic Programming as a Service framework provides a solution to the problem of using logic engines as distributed services. The paper

presents a prototype of Logic Programming as a Service built on top of the *tuProlog* system. Among the advantages of the proposed framework, exploiting a Logic Programming approach in pervasive systems encourages representing and reasoning with situations using a declarative language, promotes cooperation and inter-operation between different entities of a pervasive system, and enables reasoning over data streams as collected by sensors.

The paper by Interlandi and Tanca, titled *A Datalog-based Computational Model for Coordination-free, Data-Parallel Systems,* proposes a study of a logic programming-based computational model for eventually consistent, data-parallel systems, the keystone of which is provided by the recent finding that the class of programs that can be computed in an eventually consistent, coordination-free way is that of monotonic programs. The paper analyzes this principle, called CALM, under synchronous and reliable settings, and shows that the CALM principle holds also in *rsync* settings, but in general only for the subclass of monotonic queries defined as connected.

The paper by Pakin (*Performing Fully Parallel Constraint Logic Programming on a Quantum Annealer*) takes us into one of the first ever proposed studies that explores the use of quantum annealing techniques in the context of parallel constraint logic programming. Quantum annealers provide a method to solve particular classes of optimization problems, the paper shows how to compile a subset of Prolog, enhanced with support for constraint logic programming, into a model suitable for execution on a quantum annealer. To express constraint logic programming in the form accepted by quantum-annealing hardware, the paper identifies an analogy between expression minimization in an Ising-model Hamiltonian and unification in constraint logic programming. Based on that insight, the author implemented *QA Prolog,* a compiler that converts Prolog programs into 2-local Ising-model Hamiltonians, runs these on a D-Wave quantum annealer, and reports the results in terms of program variables.

Last but not least, the paper by Areias and Rocha, titled *Table Space Designs For Implicit and Explicit Concurrent Tabled Evaluation,* offers a comprehensive overview of the techniques for concurrent tabled evaluation and describes the design and implementation challenges of several alternative table space designs for implicit and explicit concurrent tabled evaluation, as explored in the YAP Prolog system. The paper further rises important questions and identifies future challenging research directions.

References

DOVIER, A., FORMISANO, A. AND PONTELLI, E. 2018. Parallel answer set programming. In *Handbook of Parallel Constraint Reasoning*. Y. Hamadi and L. Sais, Eds. Springer International Publishing: Heidelberg, Germany, 237–282.

GUPTA, G., PONTELLI, E., CARLSSON, M., HERMENEGILDO, M. AND ALI, K. 2001. Parallel execution of prolog programs: A survey. *ACM Transactions on Programming Languages and Systems 23,* 4, 472–602.

HYVARINEN, A. AND WINTERSTEIGER, C. 2018. Parallel satisfiability modulo theories. In *Handbook of Parallel Constraint Reasoning*. Y. Hamadi and L. Sais, Eds. Springer: Heidelberg, Germany, 141–178.

POLLARD, G. H. 1981. *Parallel Execution of Horn Clause Programs*. PhD Thesis, Department of Computing, Imperial College, London.