View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 1 (2012) 2053–2054

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Science, ICCS 2010

Practical Aspects of High-Level Parallel Programming PAPP 2010

Anne Benoit[a,*], Frédéric Gava[b]

[a] *Laboratoire de l'Informatique du Parallélisme UMR 5668 CNRS – ENS Lyon – INRIA – UCBL*
Ecole Normale Supérieure de Lyon; 46, allée d'Italie, 69364 Lyon cedex 07 – France
[b] *Laboratory of Algorithms, Complexity and Logic, University of Paris-East*
61 avenue du Général de Gaulle, 94010 Créteil cedex – France

Computational science applications are more and more complex to develop and require more and more computing power. The trend is towards the increase of cores in processors, the number of processors and the need for scalable computing everywhere. Major companies in the computing industry now recognises the urgency of reorienting an entire industry towards massively parallel computing. However, parallel and distributed programming is still dominated by low-level techniques such as send/receive message passing. Thus high-level approaches should play a key role in the shift to scalable computing in every computer.

The seventh international workshop on Practical Aspects of High-Level Parallel Programming (PAPP), affiliated to the International Conference on Computational Science (ICCS), presents recent work of researchers in these fields. The PAPP workshop is aimed both at researchers involved in the development of high-level approaches for parallel and grid computing and computational science researchers who are potential users of these languages and tools.

The topics of the PAPP workshop include high-level models (BSP, LogP, *etc.*) and tools for parallel and grid computing; high-level parallel language design, implementation and optimisation; functional, logic, constraint programming for parallel, distributed and grid computing systems; algorithmic skeletons, patterns and high-level parallel libraries; generative (*e.g.* template-based) programming with algorithmic skeletons, patterns and high-level parallel libraries; applications in all fields of high-performance computing (using high-level tools); industrial uses of a high-level parallel language; and benchmarks and experiments using such languages and tools.

The papers of the PAPP 2010 workshop are classified depending on the size of the parallel machine host: from Grid to CPGPU, passing by clusters and multi-cores.

The Java programming language increases the productivity of programmers for example by taking care of memory management, by providing a wide collection of data structures (safer with the recent introduction of genericity in the language), and many other features. Thus Java is an interesting choice as a basis for high-level parallelism. The first paper *Map, Reduce and MapReduce, the skeleton way* focuses on implementation and experimentation of two skeletons, the now well known *map, reduce* (which are massively used by Google) in a Java/Grid environment called ProActive/GCM. This prototype called *MareMare* provides better performances than the implementation of these skeletons provided by the ProActive framework.

*Corresponding author
Email addresses: `Anne.Benoit@ens-lyon.fr` (Anne Benoit), `frederic.gava@univ-paris-est.fr` (Frédéric Gava)
URL: `http://graal.ens-lyon.fr/~abenoit/` (Anne Benoit), `http://lacl.fr/gava/` (Frédéric Gava)

1877-0509 © 2012 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
doi:10.1016/j.procs.2010.04.229

Ubiquitous multicore architectures require that many levels of parallelism have to be found in codes. Dependence and data-flow analysis are the main approaches in compilers for the detection of parallelism. Adapting these analyses to ubiquitous architectures is the problematic of the second paper, *FADAlib: an Open Source C++ Library for Fuzzy Array Dataflow Analysis*. Authors provide new technics called *Fussy Array Dataflow*, implemented in a C++ library called *FADAlib*, that analyse codes to detect parallelism. Some comparisons with regular and non regular codes are also provided.

Benchmarks and experiments are the main subject of the third paper of the workshop: *Boosting the Performance of Computational Fluid Dynamics Codes for Interactive Supercomputing*. The authors focus on an experimental study of a problem on fluid dynamics. By developing visualisation utilities, they restructure a program transformation that makes the numerical code run well on multicore CPUs.

Multi-cores are also at the core of the fourth paper. But the adding of stream-processing makes the need of language of coordination for the interactions between asynchronous computations. This paper, *Parallel Signal Processing with S-Net*, describes the design of a high-level framework called *S-Net*, which provides higher-level abstraction of the coordination and reuse facility of its components. Application of this framework is provided by implementing a real radar-data processing application.

To finish the workshop, we consider low-level architectures such as graphic processors (GPUs). These GPUs are programmed with one of the most high-level language, the pure functional and well-known *Haskell*. This last paper, *GPGPU Kernel Implementation and Refinement using Obsidian*, describes how working with a domain specific language for data-parallel programming on GPUs, called *Obsidian*, and which is based on Haskell. This language provides high-level descriptions of required functions and able refining them for fine-control of the computation on the GPUs. New coordination patterns are introduced to able multi-kernel computations. Experiments and comparisons with another higher-level embedded language for GPU programming in Haskell are discussed.

We would like to thank all the people who made the PAPP workshop possible: the organisers of the ICCS conference, the other members of the programme committee: Marco Aldinucci (University of Torino, Italy), Umit V. Catalyurek (The Ohio State University, USA), Emmanuel Chailloux (University of Paris 6, France), Frédéric Dabrowski (University of Orléans, France), Alexandros Gerbessiotis (NJIT, USA), Clemens Grelck (University of Amsterdam, Netherlands), Hideya Iwasaki (The University of Electro-communications, Japan), Christoph Kessler (Linkopings Universitet, Sweden), Rita Loogen (University of Marburg, Germany), Kiminori Matsuzaki (Kochi University of Technology, Japan), Samuel Midkiff (Purdue University, USA), Susanna Pelagatti (University of Pisa, Italy), Bruno Raffin (INRIA, France), and Casiano Rodriguez-Leon (University La Laguna, Spain). We also thank the referees external to the PC for their efficient help. Finally we thank all authors who submitted papers for their interest in the workshop, the quality and variety of the research topics they proposed.