International Journal of Parallel Programming (2019) 47:161–163
https://doi.org/10.1007/s10766-018-0606-6

EDITORIAL

Guest Editorial: High-Level Parallel Programming and the Road to High Performance

J. Daniel García[1] · Arturo Gonzalez-Escribano[2]

Published online: 3 November 2018

1 The Challenges

Parallel programming in an abstract enough form has been advocated to be easy and intuitive [3]. Productive programming development, and software engineering, relies in programming abstractions with efficient implementations. Abstractions that hide the low-level details and decisions. However, a trade-off between abstraction and performance is constantly appearing in the complex environments of parallel computing.

The complexity of new platforms for high-performance computing is continuously increasing. This trend affects all kind of systems at any scale level. The list of the more powerful computational systems worldwide is regularly published on the Top500 list [4]. The observation of the trends and evolution of these systems shows that both the amount of processing units and their heterogeneity is growing at a fast pace, with exascale computing on mind. On the other hand, more parallel, heterogeneous, and complex systems-on-chip, containing a diversity of core types, are being delivered for low-end systems. Exploiting the performance power of all these systems is becoming more and more difficult.

The parallel programming languages and tools commonly used as *de-facto* standards (such as OpenMP, message-passing, CUDA, or OpenCL), or even more generic approaches [1], are not yet providing a really intuitive and interoperable parallel programming experience. In the quest for high-performance, the generality is overwhelmed by the details. Both the system and the applications programmers still face many challenges. For example, those related to learning different programming models

✉ Arturo Gonzalez-Escribano
arturo@infor.uva.es

J. Daniel García
josedaniel.garcia@uc3m.es

1 Computer Science and Engineering Department, University Carlos III of Madrid, Av Universidad, 30, 28911 Leganés, Madrid, Spain

2 Computer Science Department, University of Valladolid, Campus Miguel Delibes s/n, 47011 Valladolid, Spain

for different types of devices, exploring their interoperability, becoming an expert on specific platform optimizations, studying load balancing and scheduling mechanism for heterogeneity, and taking partition and mapping decisions in terms of the trade-offs between computation versus communication costs in hierarchical environments.

New parallel programming techniques, tools, and models with a higher level of abstraction, and the ability to deliver the whole performance potential of the new and future systems, are demanded. New proposals are needed to introduce abstractions, and at the same time solve the challenges implied in the compilation techniques, run-time systems, programming frameworks, and development tools. The new programming ecosystems should internally model and analyze the programs and platforms to adapt the computation to the specific target system, taking most of the challenging decisions needed to obtain an optimized and efficient implementation. The future of high-performance computing relies on the introduction and adoption of such simpler and higher-level parallel programming models and tools.

2 Special Issue Contents

Since 2001, the HLPP series of workshops/symposia has been a forum for researchers developing state-of-the-art concepts, and tools and applications for high-level parallel programming. The general emphasis is on software quality, programming productivity, and high-level performance models. This special issue of the International Journal of Parallel Programming contains revised papers, selected from those presented at The 10th International Symposium on High-Level Parallel Programming and Applications (HLPP 2017) [2], held in Valladolid, Spain, July 10–11th, 2017.

The program committee of HLPP 2017 accepted 14 papers, out of 20 full paper submissions, covering both foundational and practical issues in high-level parallel programming and applications. Authors of several selected papers were invited to submit extended versions to this special issue. Eight papers went through the IJPP peer review process. Finally, six papers were selected to be published in this special issue. The final rate of acceptance comparing with the submissions to the HLPP 2017 symposium has been a 30%.

In their paper, Grelck and Wiesinger study the idea of introducing a persistent layer of specialized versions of functions to asynchronously adapt generic array programs with minimal impact due to recompilation time. This approach increases the usability of generic array programming in data-parallel computations with an improved trade-off between abstraction and performance.

Jakobsson contributes a new approach to introduce automatic cost analysis of programs in the classic Bulk-Synchronous-Parallel (BSP) programming model, whose abstraction derives in predictable scalability. The technique is based on classic cost analysis and approximation of polyhedral integer volumes, only requiring textually aligned synchronization and textually aligned, polyhedral communication. The formula obtained provide tight upper-bounds of the cost of running data-oblivious programs with any input sizes, or number of processing elements. This approach provide a highly valuable tool to automatically predict the behaviour of parallel programs, and in the future, to derive proper optimization paths.

Navarro and her colleagues introduce a new approach to build a high-level parallel-loop template, targeting heterogeneous devices comprising mixed CPU and GPU devices. The solution adapts at run-time the size of the chunk of iterations that is delivered to each device on a work request. It dynamically adapts the computation to the different computation capabilities of the mixed devices, obtaining a better load balance transparently.

Wrede and his co-authors present an approach to apply swarm intelligence meta-heuristics, specifically the Fish School Search, to solve hard optimization problems in the context of skeleton based libraries for high-level parallel programming. Their approach allows a better mapping of the abstract parallel skeletons to different types of parallel hardware and devices.

Griebler and his co-authors focus on the stream-parallel programming model. They study the applicability of several state-of-the-art parallel programming frameworks and libraries. They focus on Pthreads, TBB, FastFlow, and specifically Spar, a Domain-Specific Language that introduces a new approach for parallel stream programming. They use as case-study several well-known stream basic applications, such as Dedup, Ferret and Bzip2. They do comparative analysis of the different implementations and their performance, providing new insights on the abstraction and effectivity of the Spar approach.

The contribution of López-Fandiño and his colleagues present an application of abstract parallel programming to a case study: a GPU implementation of changes detection in Multitemporal Hyperspectral Images. Their framework exploits Change Vector Analysis with the Spectral Angle Mapper distance and Otsu's thresholding, to obtain a high-level approach to this kind of programs with efficient implementations for GPUs.

These works study and propose solutions for important challenges to introduce high-level parallel programming approaches and applications. They provide solutions to increase the easy of programming and thus, contribute to the widespread adoption of parallel programming, paving the road to leverage the high-performance potential of the new and future parallel computing platforms.

Acknowledgements We would like to thank all the authors, reviewers, and editors involved in the elaboration of this special issue, including also the reviewers who were involved in the HLPP'2017 symposioum, where short versions of the papers were previously selected. We are especially grateful to Dr. Alex Nicolau, editor in chief of the International Journal of Parallel Programming, for approving this special issue and for his help along the process of its preparation.

References

1. Balaji, P. (ed.): Programming Models for Parallel Computing. Scientific and Engineering Computation. The MIT Press, Cambridge (2015)
2. HLPP'2017 Organization Committee: 10th international symposium on high-level parallel programming and applications (2017). https://hlpp2017.infor.uva.es/
3. Kirkpatrick, Keith: Parallel computational thinking. ACM Commun. **60**(12), 17–19 (2017)
4. TOP500.org: TOP500 supercomputer sites (2017). https://www.top500.org/