Guest Editors' Introduction

Matthias S. Müller · Eduard Ayguadé

Received: 27 April 2010 / Accepted: 27 April 2010 / Published online: 30 May 2010
© Springer Science+Business Media, LLC 2010

OpenMP is a portable, scalable programming model that provides a simple and flexible interface for developing shared-memory parallel applications in Fortran, C, and C++. Since its introduction in 1997, as a result of the work by a group of major computer hardware and software vendors, OpenMP has gained support from all major high-performance compiler and hardware vendors. Under the direction of the OpenMP Architecture Review Board (ARB), the OpenMP standard has continued to evolve. Version 3.0 was released in 2008, adding several new features to the OpenMP specification, including: tasking, loop collapse, enhanced loop schedules, and better definition of and control over nested parallel regions. Many compilers now support OpenMP 3.0 allowing for a broad use of the new features.

The community of OpenMP researchers and developers in academia and scientific institutions is organized under cOMPunity, a forum for the dissemination and exchange of information about OpenMP. It also serves as a forum for discussing the experiences with this programming API and for debating ideas that might improve it. IWOMP is the annual series of international workshops dedicated to the promotion and advancement of all aspects focusing on parallel programming with OpenMP. The event is a premier opportunity to learn more about parallel programming with OpenMP and other programming models, and to interact with OpenMP users, developers

M. S. Müller

Zentrum für Informationsdienste und Hochleistungsrechnen (ZIH), Technische Universität Dresden,
Dresden, Germany

E. Ayguadé (✉)

Computer Architecture Department, Technical University of Catalonia (UPC), Jordi Girona, 1-3.
Module D6., 08034 Barcelona, Spain
e-mail: eduard@ac.upc.edu

E. Ayguadé

Computer Sciences Department, Barcelona Supercomputing Center (BSC), Barcelona, Spain

and people working on the next release of the standard. The workshop also serves as a forum to present the latest research ideas and results related to OpenMP.

The fifth International Workshop on OpenMP (IWOMP-2009) took place at Technische Universität Dresden, Germany from June 3–5, 2009. The program included 14 papers, 2 keynotes and one invited talk. Over 80 researchers attended the workshop. This special issue of the International Journal of Parallel Programming is devoted to extended versions of the best papers that were presented at IWOMP-2009.

The first two papers analyze tasking, a concept introduced with OpenMP version 3.0. The first paper “Comparison of OpenMP 3.0 and Other Task Parallel Frameworks on Unbalanced Task Graphs” by Stephen L. Olivier and Jan F. Prins compares the performance of different programming paradigms and their implementation. They also identify potential improvements for future OpenMP runtime libraries. The second paper “Productivity and Performance Portability of the OpenMP 3.0 Tasking Concept when applied to an Engineering Code written in Fortran 95” by Paul Kapinos and Dieter an Mey applies the tasking construct to a simulation code. The simplicity of the parallel program and its scalability is compared to solutions like nested parallelism.

The paper “Performance Evaluation of Mixed-mode OpenMP/MPI Implementations” by J. Mark Bull, James Enright, Xu Guo, Chris Maynard, and Fiona Reed proposes a benchmark to analyze the performance aspects of the combination of two major programming models. It also highlights the strong influence of the implementation, especially when multiple OpenMP threads communicate simultaneously via MPI. The source-to-source compiler framework ROSE is presented in “Semantic-Aware Automatic Parallelization of Modern Applications Using High-Level Abstractions” by Chunhua Liao, Daniel J. Quinlan, Jeremiah J. Willcock and Thomas Panas. ROSE is able to analyze C++ programs, detect high-level abstractions in the abstract syntax tree and automatically introduce OpenMP directives to parallelize the program.

The important aspect of an efficient runtime implementation is discussed in “ForestGOMP: an efficient OpenMP environment for NUMA architectures” by François Broquedis et al. The presented approach takes the hardware topology and the dynamic load distribution of the application into account. The performance is evaluated with different versions of the *stream* benchmark. The paper “Extending OpenMP to Survive the Heterogeneous Multi-core Era” by Eduard Ayguadé et al. proposes extensions to the existing OpenMP standard to better support the use of hardware accelerators like GPGPUs with OpenMP.

More than a year after the introduction of OpenMP 3.0 there is still much work to do to evolve OpenMP in the new era of multicore parallelism, in terms of programming model definition, its implementation (compiler and runtime) on novel platforms, tools to make programming more productive, and its use in the emerging applications that will run on these systems. We hope that this collection of papers will prove to be interesting and useful to readers, and that the issues raised will stimulate further research in this area.

Enjoy!