J. Parallel Distrib. Comput. 118 (2018) 118–119

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Editorial

VSI: Edu*-2016 - Keeping up with technology: Teaching parallel, distributed and high-performance computing

Editorial Introduction

This special issue is devoted to progress in one of the most important challenges facing computing education. The work published here is of relevance to those who teach computing related topics at all levels, with greatest implications for undergraduate education. Parallel and distributed computing (PDC) has become ubiquitous to the extent that even casual users feel their impact. This necessitates that every programmer understands how parallelism and a distributed environment affect problem solving. Thus, teaching only traditional, sequential programming is no longer adequate. For this reason, it is essential to impart a range of PDC and high performance computing (HPC) knowledge and skills at various levels within the educational fabric woven by Computer Science (CS), Computer Engineering (CE), and related computational science and engineering curricula. However, rapid changes in hardware platforms, languages, programming environments, and advances in research increasingly challenge educators in deciding what to teach and how to teach it, in order to prepare students for their careers in computing.

In recognition of the importance of these issues and the underlying challenges, a curriculum working group from the IEEE Technical Committee on Parallel Processing (TCPP), the National Science Foundation (NSF), and sister communities such as the ACM, has taken up proposing and refining a curriculum for computer science (CS) and computer engineering (CE) undergraduates on PDC. The goal of this working group has been to propose a core curriculum for CS/CE undergraduates, with the premise that every such undergraduate should achieve a specified skill level regarding PDC-related topics as a result of their required coursework. This effort has resulted in a preliminary curriculum in 2010 and its formal version in 2012. Since 2011, this curriculum initiative has been coordinated by the NSF-supported Center for Parallel and Distributed Computing Curriculum Development and Educational Resources (CDER), with both the initiative and the center receiving additional support from Intel, nVIDIA, and IBM. The NSF/TCPP curriculum has over 100 early adopter institutions worldwide and the ACM/IEEE CS2013 Computer Science Curricula explicitly refers to this for comprehensive coverage of parallelism (and provides a direct hyperlink to it). In 2011, to facilitate sharing of findings and experiences and for fostering the community, the EduPar workshop series was established at TCPP's flagship IPDPS conference. Inaugurated in 2013, the EduHPC workshop series at SC conference, with greater emphasis on HPC, followed the success of EduPar. Reflecting the truly global nature of the community, in 2015 the workshops were expanded with the first Euro-EduPar at the EuroPar conference. In summary, there are now three workshops per year devoted to PDC and HPC Education. The workshops are very successful, which indicates the community's interest in Parallel and Distributed Computing, in accordance with the necessity of initiating today's students to a technology they will work with in their professional life.

This special issue sought high quality contributions in the fields of PDC and HPC education. Submissions were on the topics of EduPar 2016, Euro-EduPar 2016 and EduHPC 2016 workshops, but the submission was open to all. We received 23 submissions from all over the world. The submissions were rigorously reviewed by at least three expert external reviewers, and further evaluated by the guest editors. Of the manuscripts that advanced to the second round for revision and review, 12 were finally accepted for publication. Unlike regular journal papers, timeline and size were additional constraints because of the nature of special issue. We thank all the reviewers for their critical and expert help. Special thanks are due to our CDER center colleagues, Anshul Gupta, Arnold Rosenberg, Alan Sussman, and Chip Weems, for their guidance and help throughout.

Below, we provide an overview of the papers appearing in this volume. We have organized these papers as those most relevant to (i) pedagogy, (ii) experience, or (iii) tools, frameworks, and environments. These could also serve as sources of useful exemplars to illustrate and teach PDC/HPC concepts.

We begin with the papers most relevant to pedagogy.

In "Teaching Distributed Memory Programming from Mental Models", Victor Eijkhout argues that the mental model of the beginning parallel programmer (master–worker hierarchy) is counterintuitive to MPI model. Typically, the progression of topics in teaching parallel programming using MPI is point-to-point communication first, followed by collective communication, and other advanced topics. In this interesting paper, the author argues that this is dictated by the level of complexity in the implementation, rather than by conceptual considerations. In his opinion, MPI should be taught by sequencing of topics and use of examples that explicitly target the required mental model of the parallelism model underlying MPI. The author describes his experience of teaching MPI with collective communication first.

In "Designing Lab Sessions Focusing on Real Processors for Computer Architecture Courses: A Practical Perspective", Josue Feliu et al. propose a new approach that complements the use of simulation frameworks in lab sessions of computer architecture

https://doi.org/10.1016/j.jpdc.2018.03.007

courses. This approach is based on performing experiments on current commercial processors, where multiple hardware events related to the performance of the computer components under study are monitored, and students analyze the measured events and explore how they impact the overall performance. The paper also describes experiences from teaching a computer architecture class with accompanying lab exercises on real multicore hardware. The difficulty level of the labs ranging from basic to intermediate to advanced levels can be adopted for different levels of Architecture classes. The authors discussed the learning goals and development for each lab. The labs presented in this paper covered wide range of topics such as Cache Hierarchy Performance and System Performance, Prefetching and Issue Stalls, Inter-thread Interferences, Memory Bandwidth-Aware Scheduling, and Core Allocation in SMT Processors.

In "Teaching High-Performance Service in a Cluster Computing Course", Pedro López and Elvira Baydal propose several teaching strategies for a course on cluster computing showing how to configure, test and evaluate a high-availability/load-balanced Internet server. They present an outline of an upper-level undergraduate or graduate course on cluster computing with emphasis on high-performance service and availability, and consider a practical and hands-on approach where students build, configure, test and evaluate their own cluster-based web server. The authors aim to fill the gap observed in many cluster computing courses, and improve students' skills in this important application of computer clusters technology.[1]

The second set of papers is most relevant to experience and content of PDC/HPC courses.

In "An Approach to Task-based Parallel Programming for Undergraduate Students", Eduard Ayguade and Daniel Jiménez-González present a parallel programming course for undergraduate students, where the focus is on the shared-memory programming paradigm which facilitates the presentation of fundamental aspects and notions of parallel computing. The course follows a task-based approach, and uses OpenMP to express task decomposition strategies and simple performance models to understand the potential of task decomposition strategies.

In "Computational Science and HPC Education for Graduate Students: Paving the Way to Exascale", Alexander Antonov et al. summarize their experience of teaching a graduate course on supercomputer disciplines in a class of more than 250 students. The authors report on their approach of assigning specific practical projects to the students, with an emphasis on important properties of parallel algorithms which are critical for developing efficient applications for any parallel computing platform.

In "A Course on Big Data Analytics", Joshua Eckroth describes experience from a course designed for undergraduate junior and senior computer science students. Some of the projects used in the course are discussed, which open the door to several PDC topics including MapReduce, GPU computing, and performance analysis. The projects also provide exposure to tools and environments such as Hadoop, Apache Hive, SQL, R, and ggplot.

Our final set of papers is focused on tools and environments that address various aspects of PDC/HPC.

In "Building Web-Based Services for Practical Exercises in Parallel and Distributed Computing", Oleg Sukhoroslov describes an approach for the design and implementation of web-based environments to support development and testing of practical exercises in parallel and distributed computing courses. The approach relies on Everest, a general-purpose platform for building computational web services and the author shows how it can be exploited for the purpose of developing assignments by students and testing of the assignments by lecturers.

In "A Visual Programming Environment for Introducing Distributed Computing to Secondary Education", Brian Broll et al. introduce a Snap!-like development environment, "NetsBlox". Key to the work are blocks for message passing and remote procedure calls that can be used to explore several distributed computing concepts and environments. A cloud-based infrastructure with access to NetsBlox through a web interface is also discussed.

In "Unifying Computing Resources and Access Interface to Support Parallel and Distributed Computing Education", Linh Bao Ngo et al. present a web-based Python interface, "JupyterHub", that facilitates the teaching of PDC concepts. Several sample modules are presented in support; these include scaling, distributed locks, parallel programming, communication patterns (broadcast, reduce, scatter and gather), scheduling, and MapReduce.

In "Let's HPC: A Web-Based Platform to aid Parallel, Distributed and High Performance Computing Education", Bhaskar Chaudhury et al. propose a web-based supplement to traditional HPC and PDC courses. The platform centers around an archive of performance characteristics of parallel programs running on different environments and architectures. It serves as a handy tool for both instructors and students to experiment with factors impacting performance.

In "TSGL: A Tool for Visualizing Multithreaded Behavior", Joel C. Adams et al. present the "Thread-safe Graphics Library (TSGL)" that allows the user to add graphics calls to a multithreaded program to visualize the parallel behavior. The paper also includes examples of programs' use of the tool and directions in which student comprehension can be enhanced.

In "Preparing the Software Engineer for a Modern Multi-Core World", Nasser Giacaman and Oliver Sinnen describe the "Modern Parallel Programming Framework (MPPF)" that addresses a range of PDC concepts, including GUI concurrency and object-oriented parallelism. Experiences of integrating this framework into a course have also been reported, with the recognition that successful software engineering in this domain involves a combination of hard and soft skills.

We end with sincere thanks to all the authors for their high-quality contributions. We hope that you will find this volume rich and exciting and a valuable resource for your future teaching endeavors.

Sushil K. Prasad
Georgia State University, United States
National Science Foundation, United States
E-mail address: sprasad@gsu.edu.

Sheikh Ghafoor
Tennessee Tech University, United States
E-mail address: sghafoor@tntech.edu.

Christos Kaklamanis
University of Patras, Greece
CTI "Diophantus", Greece
E-mail address: kakl@ceid.upatras.gr.

Ramachandran Vaidyanathan
Louisiana State University, United States
E-mail address: vaidy@lsu.edu.

Available online 31 March 2018

[1] This article appears in a separate volume due to production issues but is a part of this collection: http://dx.doi.org/10.1016/j.jpdc.2018.02.027.