J Supercomput (2013) 63:323–325
DOI 10.1007/s11227-012-0743-4

Guest Editors' introduction

Abhinav Vishnu · Pavan Balaji · Yong Chen

Published online: 21 February 2012

The last decade has observed a tremendous rise in high-end computing architectures from commercial off the shelf clusters to system-on-a-chip architectures and accelerator-based systems. Significant advances in many aspects of overall architectures with multi-core design (Intel, AMD, IBM), upcoming memory architectures such as memory cubes with 3D design, a variety of SIMD units (GPUs, Intel MIC, AMD Fusion and IBM Cell Architectures), commodity networking technologies (Ethernet, InfiniBand) and proprietary technologies (Cray Gemini/Seastar, IBM BlueGene, Myrinet, Quadrics) are playing a critical role in addressing the computational needs of scientific applications. With Exascale systems on the horizon, there is a significant push for revolutionary approaches in hardware design.

However, much of the above effort would not be of use without an appropriate system software stack, which "rides" the architectural wave to provide the best performance, while addressing the impending energy and reliability challenges. This special issue is a small, but an important step toward research and best practices for designing programming models and systems software for the future.

A. Vishnu (✉)
Pacific Northwest National Laboratory, Richland, USA
e-mail: abhinav.vishnu@pnnl.gov

P. Balaji
Argonne National Laboratory, Lemont, USA
e-mail: balaji@mcs.anl.gov

Y. Chen
Texas Tech University, Lubbock, USA
e-mail: yong.chen@ttu.edu

1 In this issue

In "Deadline and Energy Constrained Dynamic Resource Allocation in a Heterogeneous Computing Environment", Young et al. present that Energy-efficient resource allocation within clusters and data centers is important because of the growing cost of energy. They study the problem of energy-constrained dynamic allocation of tasks to a heterogeneous cluster computing environment. The overall goal is to complete as many tasks by their individual deadlines and within the system energy constraint as possible given that task execution times are uncertain and the system is oversubscribed at times. They use Dynamic Voltage and Frequency Scaling (DVFS) to balance the energy consumption and execution time of each task. They design and evaluate (via simulation) a set of heuristics and filtering mechanisms for making allocations in our system. They show that the appropriate choice of filtering mechanisms improves performance more than the choice of heuristic.

In "Restricted Admission Control in View Oriented Transactional Memory", Leung et al. present a Restricted Admission Control (RAC) scheme for View-Oriented Transactional Memory. The scheme can control the number of threads concurrently accessing a view in order to reduce the number of aborts of transactions. The RAC scheme has the merits of both the locking mechanism and the transactional memory. A theoretical model is proposed to analyze the performance of the RAC scheme and to provide guidance for dynamic adjustment of the number of concurrent threads accessing the same view. Experimental results demonstrate that theoretical RAC model can mostly provide correct guidance to transactional concurrency control. Their RAC implementation shows that RAC can optimize concurrency control of transactions and performs much better than conventional transactional memory systems such as TinySTM that have no dynamic admission control.

In "Multi-Domain Job Scheduling for Leadership Computing Systems", Tang et al. present that current supercomputing centers usually deploy a large-scale compute system together with an associated data analysis or visualization system. Multiple scenarios have driven the demand that some associated jobs co-execute on different machines. They propose a multi-domain co-scheduling mechanism, providing the ability to coordinate execution between jobs on multiple resource management domain without manual intervention. They have evaluated their mechanism based on real job traces from Intrepid and Eureka, the production Blue Gene/P system and a cluster with the largest GPU installation, deployed at Argonne National Laboratory. The experimental results show that co-scheduling can be achieved with limited impact on system performance under varying workloads.

In "Concurrent Programming Constructs for Parallel MPI Applications", Berka et al. present that Concurrency and parallelism have long been viewed as important, but somewhat distinct concepts. While concurrency is extensively used to amortize latency (for example, in web- and database-servers, user interfaces, etc.), parallelism is traditionally used to enhance performance through execution on multiple functional units. Motivated by an evolving application mix and trends in hardware architecture, there has been a push toward integrating traditional programming models for concurrency and parallelism. Use of conventional threads APIs (POSIX, OpenMP) with messaging libraries (MPI), however, leads to significant programmability concerns,

owing primarily to their disparate programming models. In this paper, they describe a novel API and associated runtime for concurrent programming, called MPI Threads (MPIT), which provides a portable and reliable abstraction of low-level threading facilities. They describe various design decisions in MPIT, their underlying motivation, and associated semantics. They provide performance measurements for their prototype implementation to quantify overheads associated with various operations. Finally, they discuss two real-world use cases: an asynchronous message queue and a parallel information retrieval system. They demonstrate that MPIT provides a versatile and a low overhead programming model that can be leveraged to program large parallel ensembles.

We hope the articles in this special issue will provide relevant insights into the emerging trends in parallel programming models and systems software for HEC systems.