DOI 10.12694/scpe.v18i2.1282
Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 18, Number 2, pp. iii–iv. http://www.scpe.org

INTRODUCTION TO THE SPECIAL ISSUE ON HIGH PERFORMANCE COMPUTING SOLUTIONS FOR COMPLEX PROBLEMS

PEDRO VALERO-LARA* MAWUSSI ZOUNON† MAKSIMS ABALENKOVS‡ AND FERNANDO L. PELAYO§

Scientific and engineering problems are becoming increasingly complex. Such complex problems include numerical simulations, molecular dynamics, computational fluid dynamics, bioinformatics, image processing, and deep-learning, to name a few. In addition to these complexities, improvements in high performance computing (HPC) compromise important modifications on computer architectures, bridging the gap gap between the general scientific user community (in need of an easy access to efficient high performance computations) and the HPC programmers community (in charge of the implementation of such a complex problems efficiently).

Today, HPC programmers and scientific applications have to deal with numerous computer architectures details to take advantage of modern computing systems. Thus, strategies and tools that can help us to adapt our codes over different computing architectures is of vital importance. However, previously, we must know and identify what are the efficient programming strategies and architectonic features. This is a difficult task, as it depends on the particular problem to be computed and the computational platform on which the problem must be computed.

While only a few computational architectures were available during the last decade, today the diversity of HPC systems is more extensive. This range from clusters of common processors to heterogeneous clusters equipped with accelerators (GPU) co-processors. These platforms usually use their own compilers, languages, etc., being the implementation of portable codes very complex.

This special issue provides several studies, which involve different applications and strategies to improve performance and to achieve better usage of modern HPC systems. It is composed by five works.

The work "ARank: A Multi-Agent based Approach for Ranking of Cloud Computing Services" by A. Jahani, F. Derakhsan, and L. Mohammad-Khanl, proposes a new multi-agent based method named ARank, which is applied for ranking algorithm to reduce the waiting time of users. ARank method uses intelligent agents which they choose some candidate services and rank these services based on the quality of service values. Furthermore, the agents of ARank include the satisfaction rate of the earlier users in ranking process. The results of this evaluation show reduction in the waiting time of the users using ARank method, compared to the existing related work, Analatic Hierarchical Process (AHP) and Singular Value Decomposition (SVD).

The paper "Automatic Tuning on Many-Core Platform for Energy Efficiency via Support Vector Machine Enhanced Differential Evolution" by Z. Yang, Z. I. Rauen, and C. Liu, proposes SVM-JADE, a machine learning enhanced version of an adaptive differential evolution algorithm (JADE). They monitor the energy and EDP values of different frequency and voltage combinations of the cores, or power islands, as the algorithm evolves through generations. By adding a well-tuned support vector machine (SVM) to JADE, creating SVM-JADE, they are able to achieve energy-aware computing on many-core platform when running multiple-program workloads. Their experimental results show that their algorithm can further improve the energy by 8.3% and further improve EDP by 7.7% than JADE.

In the next study "Integrating Generic FEM Simulations into Complex Simulation Applications" by R. Dietze, M. Hofmann, and G. Rünger, it is described the efforts for integrating alternative FEM codes into a complex simulation application from the area of engineering optimisation. The application area, as well as the software components, and their interactions are presented. The integration of two different FEM codes is demonstrated based on a dedicated FEM data conversion component.

The work "SaaS for Energy Efficient Utilization of HPC Resources of Linear Algebra Calculations" by H. Astsatryan and G. da Costa, focuses on one of the most important factor of High performance computing (HPC) systems nowadays, that is to limit or decrease the power consumption while preserving a high utilization. With the availability of alternative energy, which powers such systems, there is a need to maximize the usage of

*Barcelona Supercomputing Center (BSC), Barcelona, Spain. (pedro.valero@bsc.es)
†The University of Manchester, Manchester, UK. (mawussi.zounon@manchester.ac.uk)
‡The University of Manchester, Manchester, UK. (m.abalenkovs@manchester.ac.uk)
§The University of Catilla La-Mancha, Albacete, Spain. (fernandol.pelayo@uclm.es)

alternative energy over brown power. For now, the usage of alternative energy is varying in time due to different factors such as sunny days, the wind, etc. and it is crucial to have an energy-aware algorithm to maximize the usage of this energy. In this work, a SaaS service is presented to optimize a usage of alternative energy, to reduce the power consumption and to preserve a best possible percentage of resource utilization.

Today one of the most important challenges in HPC is the development of computers with a low power consumption. In this context, the paper "Towards HPC-Embedded. Case Study: Kalray and Message-Passing on NoC" by P. Valero-Lara, E. Krishnasamy, and J. Jansson, analyses one of the new "low-cost" parallel computers, Kalray. Unlike other many-core architectures, Kalray is not a co-processor (self-hosted). One interesting feature of the Kalray architecture is the Network on Chip (NoC) connection. Habitually, the communication in many-core architectures is carried out via shared memory. However, in Kalray, the communication among processing elements can also be via Message-Passing on the NoC. One of the main motivations of this study is to present the main constraints to deal with the Kalray architecture. In particular, memory management and communication. They assess the use of NoC and shared memory on Kalray. Unlike shared memory, the implementation of Message-Passing on NoC is not transparent from programmer point of view. The synchronization among processing elements and NoC is other of the challenges to deal with in the Karlay processor. Although the synchronization using Message-Passing is more complex and consuming time than using shared memory, it is achieved an overall speedup close to 6× when using Message-Passing on NoC with respect to the use of shared memory. Additionally, we have measured the power consumption of both approaches. Despite of being faster, the use of NoC presents a higher power consumption with respect to the approach that exploits shared memory. This additional consumption in Watts is about a 50%.

Last, but not less, we would like to thank the editorial board of SCPE and reviewers for their effort and time.