Tenth International Conference on Computational Fluid Dynamics (ICCFD10), Barcelona, Spain, July 9-13, 2018

ICCFD10-2018-xxxx

Extending the HPC^2 framework capabilities for heterogeneous computing

X. Álvarez*, F. X. Trias*, A. Gorobets*,** and N. Valle*
Corresponding author: xavier@cttc.upc.edu

* Heat and Mass Transfer Technological Center, Technical University of Catalonia, C/ Colom 11, Terrassa (Barcelona), 08222, Spain
** Keldysh Institute of Applied Mathematics, Miusskaya Sq. 4, Moscow, 125047, Russia

1 Introduction

Continuous enhancement in hardware technologies enables scientific computing advancing incessantly to reach farther aims. After hitting petascale speeds in 2008, several organizations and institutions began the well-known global race for exascale high-performance computing (HPC) [1]. Thence, hardware developers have to face two big challenges. Firstly, the energy efficiency of the devices ought to be drastically incremented. Namely, the FLOPS per megawatt ratio has to be increased by a factor of 100 respect to the first petascale machines. Secondly, the memory bandwidth of the devices also needs to be increased. The common FLOP-oriented architectures (*i.e.* very high and growing FLOPS to memory bandwidth ratios) are not efficient dealing with the numerical methods used in scientific computing, which typically involve sparse matrix and vector operations (algorithms with a very low arithmetic intensity). Therefore, the maximum achievable performance is normally reduced to a small fraction of the peak performance as demonstrated by the HPCG Benchmark [2] results. In consequence, massively-parallel devices of various architectures are being incorporated into the newest supercomputers.

This trend is being reflected in most of the fields that rely on large-scale simulation codes such as computational fluid dynamics (CFD). Scientists and software developers have to rethink algorithms and rewrite codes, sometimes from the scratch. The computing operations that form the algorithm, the so-called kernels, must be compatible with distributed- and shared-memory MIMD parallelism and, more importantly, with stream processing, which is a more restrictive parallel paradigm. Consequently, the fewer the kernels of an application, the easier it is to provide portability. Furthermore, the hybridization of HPC systems makes the design of simulation codes a rather complex problem. Heterogeneous implementations such as an MPI+OpenMP+OpenCL parallelization [3] can target a wide range of architectures and combine different kinds of parallelism. Hence, they are becoming increasingly necessary in order to engage all available computing power and memory throughput of CPUs and accelerators. In Ref.[4] the performance of the PyFR framework was tested on a hybrid node with a multicore CPU, NVIDIA and AMD GPUs. Further, in Ref.[5] the scalability of the HOSTA code was tested up to 1024 TianHe-1A hybrid nodes.

In this context of accelerated innovation, power consumption reduction, and considering the enormous complexity of porting existing codes, the software portability and efficiency become of crucial importance. Hence, we presented in a previous work **??** the HPC^2 (Heterogeneous Portable Code for HPC). It is a fully-portable, algebra-based framework capable of heterogeneous computing with many potential applications in the fields of computational physics and mathematics.

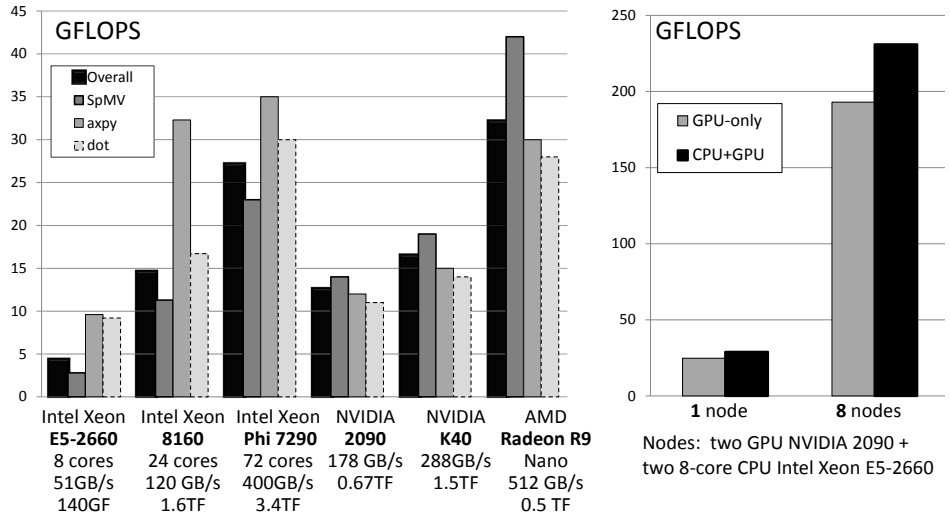

Figure 1: Left: performance of the overall DNS algorithm and the three basic kernels tested on different devices. Right: heterogeneous execution of the overall DNS algorithm vs GPU-only.

2 Problem Statement

The HPC^{2}aims at providing a user-friendly environment for writing algorithms in the fields of computational physics and mathematics. In its application to CFD, the algorithm of the time-integration phase relies on a reduced set of only three algebraic operations: the sparse matrix-vector product, the linear combination of vectors and the dot product. This algebraic approach combined with a multilevel MPI+OpenMP+OpenCL parallelization naturally provides portability. The performance has been studied on different architectures (Figure 1) including multicore CPUs, Intel Xeon Phi accelerators and GPUs of AMD and NVIDIA. The multi-GPU scalability is demonstrated up to 256 devices. The heterogeneous performance will be demonstrated on a CPU+GPU hybrid cluster.

The algebra-based implementation approaches has gained popularity in recent years. Studies based on such formulation have benefited from its power for the analysis and the construction of accurate discretizations. For this reason, our team aims at extending the capabilities of HPC^{2}towards new applications such as operator-based algorithms for multi-phase flow simulations.

References

[1] Dongarra et al. The International Exascale Software Project roadmap. *The International Journal of High Performance Computing Applications*, 25(1):3–60, 2011.

[2] J. Dongarra and M. Heroux. HPCG Benchmark: a new metric for ranking high performance computing systems. Technical Report June, 2013.

[3] A. Gorobets, F. X. Trias, and A. Oliva. A parallel MPI+OpenMP+OpenCL algorithm for hybrid supercomputations of incompressible flows. *Computers and Fluids*, 88:764–772, 2013.

[4] F. D. Witherden, B. Vermeire, and P. Vincent. Heterogeneous computing on mixed unstructured grids with PyFR. *Computers and Fluids*, 120:173–186, 2015.

[5] Chuanfu Xu, Xiaogang Deng, Lilun Zhang, Jianbin Fang, Guangxue Wang, Yi Jiang, Wei Cao, Yonggang Che, Yongxian Wang, Zhenghua Wang, Wei Liu, and Xinghua Cheng. Collaborating CPU and GPU for large-scale high-order CFD simulations with complex grids on the TianHe-1A supercomputer. *Journal of Computational Physics*, 278(1):275–297, 2014.