ParCFD'2020
32^{nd} International Conference on Parallel Computational Fluid Dynamics
May-11-13 2020, Nice FRANCE

NUMA-AWARE STRATEGIES FOR THE EFFICIENT EXECUTION OF CFD SIMULATIONS ON CPU SUPERCOMPUTERS

Xavier ÁLVAREZ-FARRÉ[1*], Andrey GOROBETS[2], F. Xavier TRIAS[1] and Assensi OLIVA[1]

[*]Technical University of Catalonia,
Heat and Mass Transfer Technological Center,
Carrer Colom 11, 08222 Terrassa (Barcelona), Spain

[†]Keldysh Institute of Applied Mathematics,
Miusskaya Sq. 4, 125047 Moscow, Russia

Key words: Parallel CFD applications, Incompressible flows, NUMA, Heterogeneous computing

Abstract. The growing variety of computing architectures and the hybridization of high-performance computing systems encourage the research for portable implementations of numerical methods in simulation codes. However, the pursuit of efficient and portable implementations is a rather complex problem. The present work is devoted to the development of portable parallel algorithms, primarily for scale-resolving time-accurate simulations of incompressible flows with turbulent heat and mass transfer. The heterogeneous computing capability allows for engaging both processors and accelerators efficiently. In addition to computing on accelerators, special attention is paid at efficiency on multiprocessor nodes with significant non-uniform memory access factor. In this work, we study in detail the parallel efficiency and performance for different execution modes on up to ten thousand cores of MareNostrum 4 supercomputer.

1 INTRODUCTION

Current HPC systems consist of multiple hybrid computing nodes interconnected via a communication infrastructure. Hybrid nodes may be composed of several hardware devices of different architectures, such as central processing unit (CPU), many integrated core (MIC) or graphics processing unit (GPU), among others. The computing operations that form the algorithms must be compatible with distributed- and shared-memory multiple instruction, multiple data (*i.e.*, DM-MIMD and SM-MIMD respectively) parallelism, and more importantly, with stream processing (SP), which is a more restrictive parallel paradigm. Therefore, a complex multilevel parallel implementation model is required to combine the different parallel paradigms and their corresponding computing frameworks.

Nowadays, the use of the GPU architecture for computational fluid dynamics (CFD) simulations has become rather mature, and there are many successful examples in the literature [1, 2]. One of the most significant examples of large-scale, GPU-based simulation

can be found in [3], reporting an extraordinary sustained performance of 13.7PF. These popular GPU-only implementations may derive from the assumption that GPU is powerful enough to neglect the CPU's capabilities. However, both architectures are still advancing, and their competition is far from over. Therefore, if either CPUs or GPUs can be used efficiently for CFD simulations, why not to use them both concurrently? Considering a 1:3 ratio between CPU's and GPU's bandwidth, the heterogeneous parallel model can increase the sustained performance in 30% as proved in [4]. Nevertheless, heterogeneous implementations are still relatively uncommon. For instance, see [5, 6].

In our previous work [7], we proposed an algebra-based framework as a portable solution for direct numerical and large eddy simulation (DNS and LES respectively) of incompressible turbulent flows on unstructured meshes. However, the multilevel parallel implementation in [7] became ineffective on multiprocessor supercomputers with significant non-uniform memory access factor (NUMA). In the present work, we have solved this issue and significantly improved the efficiency on multiprocessor nodes and, consequently, increased the gain of heterogeneous computing. Therefore, we present an in-depth performance study on up to 200 multiprocessor nodes of the MareNostrum 4 supercomputer.

2 THE ALGEBRA-BASED APPROACH FOR CFD SIMULATIONS

Consider a simulation code composed of a minimal set of basic operations, preferably standard linear algebra operations, which are simple and compatible with the restrictive stream processing paradigm. Following an algebraic approach, we replace the traditional stencil data structures and sweeps on CFD codes by algebraic data structures and kernels. The discrete operators and mesh functions are stored as sparse matrices and vectors, respectively. Thus, the algorithm for DNS and LES of incompressible turbulent flows relies on a reduced set of three algebraic kernels: the sparse matrix-vector product (SpMV), the linear combination of vectors (axpy) and the dot product (dot). Specifically, the SpMV often receives much attention because it is prevalent in many computing applications. Indeed, this key operation is a bottleneck in scientific computing because it is a memory-bounded operation with a very low arithmetic intensity, and it leads to indirect memory accesses with unavoidable cache misses. Therefore many authors strive to adapt sparse matrix storage formats for different architectures and matrix properties. For instance, see [8, 9].

3 PERFORMANCE OF THE SPMV ON CPU SUPERCOMPUTERS

In modern supercomputers, the CPU consist of a pool of cores grouped into NUMA nodes (*i.e.*, different CPU sockets, even clusters of cores inside each socket). Modern CPU architectures integrate dozens of cores and allow for simultaneous multithreading (SMT). Therefore, a fine-tuned SM-MIMD parallelisation is required to engage all the CPU cores. In this work, the performance of our in-house SpMV kernel on multiprocessor nodes is improved by a NUMA-aware OpenMP implementation. It sets thread affinity to properly bind threads to NUMA nodes to ensure compact data locality.

The benefits of the NUMA-aware implementation have been studied on MareNostrum

4 supercomputer at Barcelona Supercomputing Center (BSC). Its nodes are composed of two Intel Xeon Platinum 8160 CPUs (24 cores, 2.1 GHz, 6 DDR4-2666 memory channels, 128 GB/s memory bandwidth, 33 MB L3 cache), and are interconnected through a high-speed interconnection network, the Intel Omni-Path (OPA).

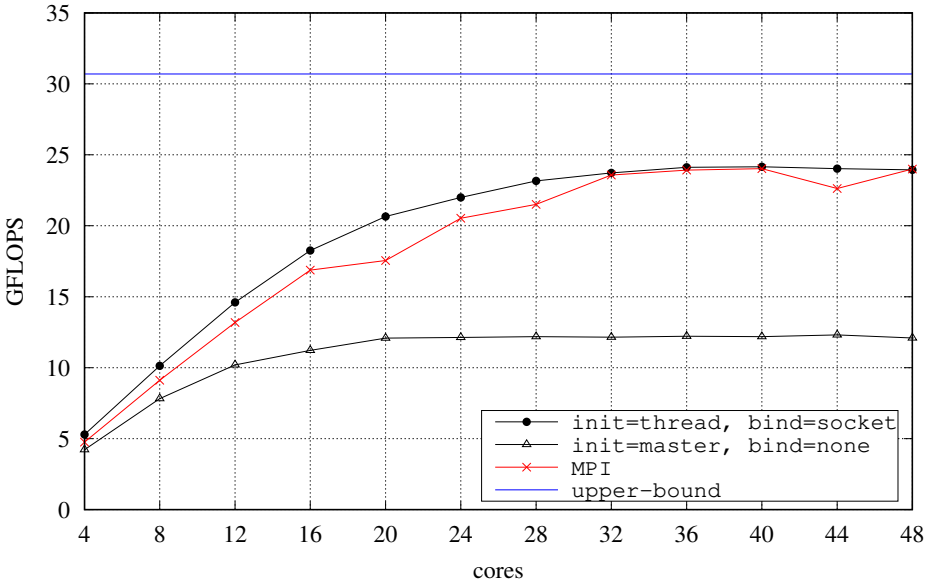

Figure 1: Comparison of different parallel modes of the SpMV kernel on a dual-processor node with two Intel Xeon Platinum 8160 CPUs.

The single-node results for different parallel modes in Figure 1 reveal the effects of thread binding and NUMA initialisation. The sparse matrix used in this study arise from the symmetry-preserving discretisation [10] of the Laplacian operator on an unstructured hex-dominant mesh of 17M cells. The MPI mode, which leads to the most compact data placement, is slightly overcome by the OpenMP mode with thread binding to sockets and thread data initialisation (denoted in the plot as `init=thread, bind=socket`) due to the absence of communications in the latter mode. In both cases, the performance begins to stagnate from 24 cores and reaches the maximum performance at 40 cores. The OpenMP mode with master initialisation and no affinity results in a deficient performance comparable to using a single CPU socket.

Finally, the multi-node results on up to ten thousand cores of MareNostrum 4 supercomputer are going to be presented and discussed at the conference.

Acknowledgments

The work has been financially supported by a competitive R+D project (ENE2017-88697-R) by the Spanish Research Agency. X. Á. F. is supported by a predoctoral contract (2019FI_B2-00076) by the Government of Catalonia. The work has been carried out using

the computing resources of the Barcelona Supercomputing Center (MareNostrum 4); The authors thankfully acknowledge the institution.

REFERENCES

[1] A. Yamanaka, T. Aoki, S. Ogawa, and T. Takaki, "GPU-accelerated phase-field simulation of dendritic solidification in a binary alloy," *Journal of Crystal Growth*, vol. 318, no. 1, pp. 40–45, 2011.

[2] P. Zaspel and M. Griebel, "Solving incompressible two-phase flows on multi-GPU clusters," *Computers & Fluids*, vol. 80, no. 1, pp. 356–364, 2013.

[3] P. E. Vincent, F. Witherden, B. Vermeire, J. S. Park, and A. Iyer, "Towards Green Aviation with Python at Petascale," in *SC16: International Conference for High Performance Computing, Networking, Storage and Analysis*, no. November, pp. 1–11, IEEE, nov 2016.

[4] S. A. Soukov and A. V. Gorobets, "Heterogeneous Computing in Resource-Intensive CFD Simulations," *Doklady Mathematics*, vol. 98, pp. 472–474, sep 2018.

[5] F. D. Witherden, B. C. Vermeire, and P. E. Vincent, "Heterogeneous computing on mixed unstructured grids with PyFR," *Computers & Fluids*, vol. 120, pp. 173–186, oct 2015.

[6] A. Gorobets, S. Soukov, and P. Bogdanov, "Multilevel parallelization for simulating compressible turbulent flows on most kinds of hybrid supercomputers," *Computers & Fluids*, vol. 173, pp. 171–177, sep 2018.

[7] X. Álvarez, A. Gorobets, F. Trias, R. Borrell, and G. Oyarzun, "HPC^2 – A fully-portable, algebra-based framework for heterogeneous computing. Application to CFD," *Computers & Fluids*, vol. 173, pp. 285–292, sep 2018.

[8] A. Monakov, A. Lokhmotov, and A. Avetisyan, "Automatically Tuning Sparse Matrix-Vector Multiplication for GPU Architectures," in *High Performance Embedded Architectures and Compilers* (Y. N. Patt, P. Foglia, E. Duesterwald, P. Faraboschi, and X. Martorell, eds.), pp. 111–125, Springer Berlin Heidelberg, 2010.

[9] J. L. Greathouse and M. Daga, "Efficient Sparse Matrix-Vector Multiplication on GPUs Using the CSR Storage Format," in *SC14: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 769–780, IEEE, nov 2014.

[10] F. X. Trias, O. Lehmkuhl, A. Oliva, C. D. Pérez-Segarra, and R. W. C. P. Verstappen, "Symmetry-preserving discretization of NavierStokes equations on collocated unstructured grids," *Journal of Computational Physics*, vol. 258, pp. 246–267, feb 2014.