The 8th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2022
5-9 June 2022, Oslo, Norway

Design And Analysis Of Task-based Parallelization Of A Discontinuous Galerkin Euler Flow Solver On Heterogeneous Architectures

Sangeeth Simon[1,2,*], Vincent Perrier[1,2], Jonathan Jung[2,1] and Matthieu Haefele[3]

[1]Team Cagire, INRIA Bordeaux Sud-Ouest
[2] Laboratoire de Mathématiques et de leurs applications, Université de Pau et des Pays de l'Adour Bâtiment IPRA, Avenue de l'Université, Pau, France-64 013.
[3]Universite de Pau et des Pays de lAdour, E2S UPPA, CNRS, LMAP, Pau, France-64 013.

Keywords: *Task-based Parallelization, StarPU, Discontinuous Galerkin, Higher order, Heterogeneous computing, Euler System.*

Modern high performance computing hardware is increasingly a complex heterogeneous architecture. They commonly comprise of various dissimilar computing units like multi-core processors, specialized co-processors, GPGPUs, FPGAs etc., coexisting on an array of fast interconnected nodes. In combination with cache-coherent unified virtual memory systems, these machines are capable of high-bandwidth memory access, and offers scope for improved application performances at reduced power consumption. However, from the perspective of an application developer, this heterogeneity is a double-edged sword: on one hand it offers multiple layers of parallelism for useful exploitation, while on the other it poses a serious challenge in designing portable applications capable of approaching the theoretical peak performances on such architectures.

An interesting alternative to efficiently parallelize applications on heterogeneous architectures is the *task-based programming model*[1]. The central idea in this model is to express the application parallelism as a series of asynchronous non-blocking 'tasks' with explicit data dependencies among them. These tasks are then submitted to an underlying run-time scheduler, which arranges them into a directed acyclic graph. On such a task graph, each node represents a task and the connectivity edges represents the data dependencies. The task graph enables the scheduler to identify tasks that can be executed in parallel and map them onto the appropriate computational resources on-the-fly. Additionally, the data transfers between various devices on the hardware are also handled exclusively by the scheduler with little intervention from the programmer. Among the various schedulers that exist today, StarPU[2] is a compelling option due to its simple interfaces, expressive data management library, dynamic scheduling capabilities, and the flexibility it allows toward designing custom scheduling strategies.

Although StarPU has been successfully used to design parallel linear algebra libraries [3], only a few attempts have explored their effectiveness in the field of Computational Fluid Dynamics (CFD) [4, 5, 6, 7]. These efforts have focused exclusively on building task-parallel CFD applications based upon conventional lower-order Finite Difference (FD) and

Finite Volume (FV) method. However, there is a general contention that task parallelism would be inefficient for such applications because of the inherent low arithmetic intensity (AI) associated with low order versions of these methods. Moreover, low AI also results in tasks with smaller execution times which may not justify the scheduling overhead incurred in executing them. To overcome this, some researchers have proposed designing artificially larger tasks or grouping together various algorithmic steps into tasks to improve their execution times [5, 4]. An alternative option could be to focus on improving the floating point operations (flops) of the application kernels by employing spatially higher order versions of these methods. Unfortunately, achieving such accuracy improvements for the FD or the FV methods is neither straightforward nor computationally efficient. Also, any potential gain in flops achieved from such higher-order variants is usually offset by the communication cost incurred in dealing with the wider stencils required for such accuracy gains.

In this regard, modern higher order CFD methods based on the Discontinuous Galerkin (DG) framework offers an opportunity. The compact stencil, the naturally high AI per mesh element, fewer communication between elements, hp-adaptability, an invertible block diagonal mass matrix etc. makes them interesting candidates for exploiting task-based parallelism [8]. In this work, we report on our efforts towards developing a hardware agnostic, scalable, DG based CFD code using the StarPU framework. The proposed code is targeted toward computing solutions to the Euler system: a classic non-linear system of equations governing compressible fluid flows. We will report on the strong and weak scalability results on various shared memory architectures at our disposal like Intel Xeon Haswell E5-2680, Intel Xeon Skylake Gold, and Intel Xeon Phi KNL. We will also discuss our findings on the domain partitioning and task granularity required to achieve the reported scalability.

REFERENCES

[1] Thoman, Peter Et al.. A Taxonomy of Task-Based Parallel Programming Technologies for High-Performance Computing. *J. Supercomput.* Vol. **74**, pp. 1422–1434, 2018.

[2] Augonnet, C. Thibault, S. Namyst, R. and Wacrenier, P. A. StarPU: a unified platform for task scheduling on heterogeneous multicore architectures. *Concurrency and Computation: Practice and Experience*, Vol. **23(2)**, pp. 187–198, 2011.

[3] S. Thibault Et al.. Matrices Over Runtime Systems at Exascale. *High Performance Computing, Networking Storage and Analysis*, pp.1330–1331, 2012.

[4] Couteyen C. J. M. and Roman, J. and Brenner, P. Design and analysis of a task-based parallelization over a runtime system of an explicit finite-volume CFD code with adaptive time stepping. *Journal of Computational Science* Vol. **28**, pp. 439–454, 2018.

[5] Lucas, L. N. and Serpa, M. da S. and Schnorr, M. L. and Navaux, P. O. A. Task-based parallel strategies for computational fluid dynamic application in heterogeneous CPU/GPU resources. *Concurrency and Computation: Practice and Experience* Vol. **32**, 2020.

[6] Jeannot, Emmanuel and Fournier, Yvan and Lorendeau, Benjamin, Experimenting task-based runtimes on a legacy Computational Fluid Dynamics code with unstructured meshes. *Computers and Fluids*, Vol. **173**, pp. 51–58, 2018.

[7] Essadki, M. Jung, J. Larat, A. Pelletier, M. Perrier, V. A Task-Driven Implementation of a Simple Numerical Solver for Hyperbolic Conservation Laws. *ESAIM: ProcS*. Vol. **63**, pp. 228–247, 2018.

[8] Brenger Bramas, Philippe Helluy, Laura Mendoza, Bruno Weber, Optimization of a discontinu- ous Galerkin solver with OpenCL and StarPU.*International Journal on Finite Volumes*, Vol. **15**, pp. 1–19, 2020.