Opportunities for GPU Acceleration of OpenFOAM

Stan Posey[1], Frederic Pariente[2]

[1]NVIDIA Corporation, Santa Clara, USA, CFD Solutions Development, sposey@nvidia.com
[2]NVIDIA Ltd., Toulouse, FR, Developer Relations, fpariente@nvidia.com

Current trends in high performance computing include the use of graphics processing units (GPUs) as massively parallel co-processors to CPUs in order to accelerate numerical operations that are common to computational fluid dynamics (CFD) solvers. This talk will examine the state of play in various GPU implementations for OpenFOAM, and will describe recent GPU hardware and software developments that provide new HPC opportunities for industry CFD applications that deploy OpenFOAM solvers.

Introduction

The efficient use of hardware system resources and CFD simulation turn-around times continue to be important factors behind engineering decisions to expand CFD as a technology to support product design. While the progress in recent years of CFD simulation verses physical experimentation has been remarkable, the availability of inexpensive workstations and clusters with conventional multi-core CPU parallel solvers has not been enough to motivate broad industry deployment of high-fidelity modelling and use of design optimization procedures.

During recent years, CFD has become increasingly reliant on clusters of multi-core CPUs to enable more detailed simulations within the limits of engineering project times. For this reason, the scalability of a CFD solver across multiple servers became more important than single-processor performance. Most computational performance has relied strictly on compilers to achieve a nominal level of single processor speedups, and development investments have gone towards distributed memory parallel.

Developments in GPU-parallel CFD achieve substantial speedups from a fine grain, or second-level of parallelism under an existing distributed memory, or first-level 'scalable' parallelism. For most GPU implementations in CFD, the focus is on implicit sparse iterative solvers whereby linear algebra matrix operations that are normally processed on CPUs, are off-loaded to the GPU for acceleration, resulting in an overall simulation speedup.

Iterative solvers have become the standard for commercial CFD owing to their computational and storage efficiency. In order to aid the adoption of GPUs for applied CFD with the off-load computing model and assist in the CFD community's broad range of GPU development interests (including particle-based CFD), NVIDIA continues to invest in high performance iterative solvers in several areas:

- cuBLAS lib of basic linear algebra subroutines (BLAS) for dense matrices
- cuSPARSE lib of kernels and solver components for sparse matrices with a variety of formats
- Thrust, an open source C++ template library of high performance parallel primitives
- AmgX complete iterative solver lib with emphasis on multigrid schemes and smoothers

GPU implementations of OpenFOAM made use of one or more of these methods in ways that helped simplify the programming effort while providing GPU library-level numerical performance.

GPU History

Implicit sparse solvers are the current GPU focus of most OpenFOAM implementations owing to their favorable execution profile of having a 'hotspot' and the simpler requirement that only a small % of lines of code must be ported to the GPU for potentially significant gains. All developments make use of GPU libraries at some level, which provide optimized computational kernels.

Starting from 2011, at least a dozen GPU implementations for OpenFOAM can be found from search and literature review that include both research and commercial interests. A list is provided in an order of the year of introduction for 10 that have achieved a level of advanced capability and/or functionality:

- 2011: Symscape, UK (Smith) - GPU Linear Solver Library for OpenFOAM
- 2011: Cufflink, US (Combest – now ENGYS) - Cuda For FOAM Link (cufflink) ; Also github
- 2012: Vratis, PL (Miroslaw – now Microsoft Azure) - SpeedIT Plugin to OpenFOAM
  - 2012 Paper from Wroclaw University of Technology, Poland (Tomczak, et al.): Complete PISO and SIMPLE solvers on Graphics Processing Units
- 2013: PARALUTION, DE (Lukarski – now Apple) - Plugin for OpenFOAM ; Also GTC Talk
- 2014: FluiDyna/Altair, DE, (Indinger) - Culises for GPU-Based Acceleration of OpenFOAM
- 2015: simFlow, PL (Jasiński) - RapidCFD OpenFOAM running on GPU
  - 2016 Paper from Norwegian University of Science and Technology, Norway (Arslan) A benchmark test for OpenFOAM using GPU cards : Flow past a centrifugal pump
- 2016: DoD NSWCCD NavyFOAM, US (Williams, Sarofeen – now NVIDIA)
  - Accelerated Iterative Linear Solver with GPUs for CFD Calculations of Unstructured Grids
- 2017: Guthrie Engineering, UK (Guthrie) - TurboCFD - OpenFOAM on GPUs
- 2017: University of Moratuwa, LK (Rathnayake – now Ph.D. student UIUC)
  - Paper: OPENFOAM ON GPUS USING AMGX
- 2018: Brunel University London, UK (Dyson – now Watkinson Motors)
  - GPU Accelerated Linear System Solvers for OpenFOAM and Their Application to Sprays

This talk will review important attributes and contributions from the list, and their availability status on current GPU platforms.

GPU Suitability

For OpenFOAM and most other CFD, the computational characteristic of most concern is the rate of memory bandwidth for a system architecture in order to efficiently solve sparse matrices. Current GPU architectures provide factors more in memory bandwidth performance (and FLOPS) vs. the latest generation of x86 CPUs, and the reason GPUs have become one of the most critical components of HPC architectures for Top500 systems.

Iterative sparse solvers are widely used in CFD for simulations that deploy implicit schemes, owing to their computational and storage efficiency. Implementation of an iterative sparse solver such as a conjugate gradient or multi-grid method on a GPU would consist mainly of concerns with kernel performance of sparse-matrix-vector-multiply (SpMV) and a few additional BLAS-1 kernels. But because of an SpMV kernel's low arithmetic intensity, it is memory-bound and optimization of the memory access pattern from potential re-design of the sparse matrix data structures, can achieve peak performance.

The common focus of current and historical GPU implementations of OpenFOAM has been performance of sparse solvers, and in each case with an external solver. These GPU-based solvers deployed a range of methods, but typically used a GPU library such as cuSPARSE for sparse SpMV in some conjugate gradient method, or cuBLAS for some vector-vector calculations, or in some cases the use of a multi-grid scheme from the AmgX library of solvers, preconditioners, and smoothers.

These solver "plugin" procedures start with OpenFOAM to set up the system of linear equations for solution to Ax=b, and the A matrix and b vector are sent from OpenFOAM to a separately compiled, external GPU based solver. Before solving the linear system, the external solvers usually convert the LDU matrix and the right-hand vector to a different format such as CSR that is better suited to the GPU solver. After solving the system, the solution vector must be converted back to an OpenFOAM vector before solution transfer back to OpenFOAM. This conversion between vectors and matrices requires substantial work, but is required for GPU performance.

Also notable is that while each of the 10 implementations listed have made good GPU progress in various ways, none have reached a state of production-ready OpenFOAM on GPU systems in CFD practice. There are a variety of both technical and (mostly) non-technical reasons for this, and the talk will examine the requirements to achieve this production-ready state in collaboration with the OpenFOAM community.

References

Combest, DP and Day, J 2011. “Cufflink: a library for linking numerical methods based on CUDA C/C++ with OpenFOAM”. http://cufflink-library.googlecode.com.

CUSPARSE, NVIDIA 2017. “CUBLAS libraries”. https://developer.nvidia.com/cusparse.

Spisso, I., Amati, G., HPC Comparison of Hypre Petsc vs Pstream as external linear algebra library for OpenFOAM : Project Presentation, ESI OpenFOAM Conference, Hamburg, 23-25 Oct 2018.

Lukarski, Dimitar and Trost, Nico 2014. “PARALUTION project”. http://www.paralution.com.

Naumov, M., M. Arsaev, P. Castonguay, J. Cohen, J. Demouth, J. Eaton, S. Layton, N. Markovskiy, N. Sakharnykh, R. Strzodka et al. 2014. “AmgX: Scalability and performance on massively parallel platforms”. SIAM workshop on exascale applied mathematics challenges and opportunities. SIAM.

NVIDIA AmgX 2014. “AmgX Documentation”. https://developer.nvidia.com/amgx.

NVIDIA, C. 2008. “Cublas library”. NVIDIA Corporation, Santa Clara, California vol. 15, pp. 27.

OpenFOAM-Wiki 2019. “Matrices in OpenFOAM”. https://openfoamwiki.net/index.php/OpenFOAM_guide/Matrices_in_OpenFOAM.

OpenFOAM-Wiki 2019. “GPGPU”. https:// https://openfoamwiki.net/index.php/GPGPU.