11th World Congress on Computational Mechanics (WCCM XI)
5th European Conference on Computational Mechanics (ECCM V)
6th European Conference on Computational Fluid Dynamics (ECFD VI)
July 20 - 25, 2014, Barcelona, Spain

Toward Exascale Simulations with the CFD code Nek5000

Dan Henningson[1*], Philipp Schlatter[1*], Adam Peplinski[1*], Stefano Markidis[2*], Michael Schliephake[2*], Erwin Laure[2*], Jing Gong[2*], Alistair Hart[4], David Henty[5], Paul Fischer[6] and Katherine Heisey[6]

[1] Department of Mechanics, KTH Royal Institute of Technology, SE-10044, Stockholm, Sweden
[2] PDC Center for High Performance Computing, KTH Royal Institute of Technology, SE-10044, Stockholm, Sweden
[3] Cray U.K. Limited, Reading, Berkshire RG7 5AH, UK
[4] EPCC, The University of Edinburgh, Edinburgh, EH9 3JZ, UK
[5] Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, USA

Key Words: *CRESTA Project, Nek5000, Adaptive Mesh Refinement, OpenACC*

Nek5000 is an open-source code for simulating incompressible flows [1]. Its discretization scheme is based on the spectral-element method. In this approach, the incompressible Navier-Stokes equations are discretized in space by using high-order weighted residual techniques employing tensor-product polynomial bases. The resultant linear systems are computed using CG and GMRES with convenient pre-conditioners. The tensor-product-based operator evaluation can be implemented as matrix-matrix products. The code is widely used in a broad range of applications.

We are preparing the CFD code Nek5000 for the exascale era. We focus both on algorithmic improvements using adaptive mesh refinement (AMR) and on software challenges using hybrid computer architectures with accelerators for exascale simulations.

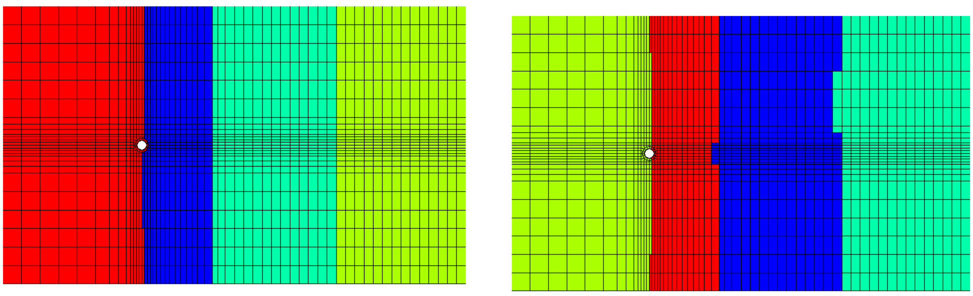

Figure 1. (left) ParMetis Graph partitioning (right) Nek5000 static partitioning

The implementation of AMR algorithms in Nek5000 gives the possibility to increase grid resolution in flow regions where small scales need to be resolved. Grid refinement, either automatic or by user intervention, is a crucial feature for the future scalability of the code on exascale machines. It is in particular important for the simulation of large-scale CFD problems, involving stability calculations, where error reduction in the sensitive flow regions is crucial.

In our implementation we adopt octree structure of the grid in which every parent element can be divided into 4 or 8 children elements in 2 and 3 dimensional cases respectively.

* Affiliation to the Swedish e-Science Research Center (SeRC)

Refinement, coarsering and balancing of the mesh is managed by p4est library [2], and dynamical grid decomposition is done with the ParMetis, which is a parallel library for dynamical graph partitioning. Mesh partitioning of 2D flow past circular cylinder on four processes in Nek5000 is presented in Figure 1. It shows that ParMetis can obtain similar quality of partition when compared with native Nek5000 static partitioning.

On the road to exscale, we are porting the CFD code Nek5000 on massive parallel hybrid CPU/GPU systems for the first time. This requires several steps: first, we have shown how to use hybrid massively parallel computing on a simplified version of Nek5000 code. Second we are investigating possible strategies to speed up the full Nek5000 code when running computer systems equipped with GPU. This study provides an example of how to port real-world application to CPU/GPU massively parallel system, and also accesses the effectiveness of OpenACC [3] and compiler support for GPU programming with a view to future hybrid exascale simulations. This is important to avoid the inefficiencies of rewriting large portions of the code in a low-level language, thus ending up with multiple version of the source code to maintain. OpenACC enables existing HPC application codes to run on accelerators with minimal source code changes. This is done using compiler directives and API calls, with the compiler being responsible for generating optimized code and the user guiding performance only when necessary.

In Ref. [4], we presented a case study of porting NekBone to a parallel GPU-accelerated system. We reached a parallel efficiency of 68.7% on 16,384 GPUs of the Titan XK7 supercomputer at the Oak Ridge National Laboratory. The scalability study is presented in Figure 1. Currently we are working on porting the full Nek5000 code to multi-GPU systems**.**

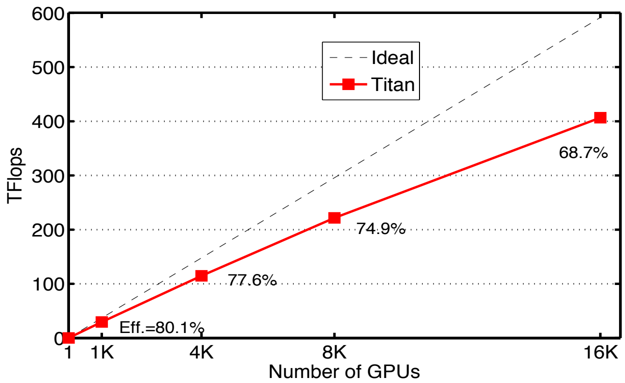

Figure 2 The weak scaling of NekBone with optimized subroutines on Titan

REFERENCES

[1] P. F. Fischer, J. W. Lottes, and S. G. Kerkemeier, *The Nek5000 code,* http://nek5000.mcs.anl.gov

[2] C. Burstedde, L. C. Wilcox, and O. Ghattas, *p4est: Scalable Algorithms for Parallel Adaptive Mesh Refinement on Forests of Octrees.* SIAM Journal on Scientific Computing, Vol. **10**, pp 1103-1133, 2011

[3] OpenACC standard, http://www.openacc-standard.org

[4] S. Markidis, J. Gong, M. Schliephake, E. Laure, A. Hart, D. Henty, K. Heisey, and P. F. Fischer, *OpenACC Acceleration of Nek5000, Spectral Element Code*, submitted to Advances in Engineering Software Journal.