Geophysical Research Abstracts
Vol. 17, EGU2015-1276, 2015
EGU General Assembly 2015

Acceleration of a Particle-in-Cell Code for Space Plasma Simulations with OpenACC

Ivy Bo Peng (1), Stefano Markidis (1), Andris Vaivads (2), Juris Vencels (1), Jan Deca (3), Giovanni Lapenta (3), Alistair Hart (4), and Erwin Laure (1)

(1) HPCViz, KTH Royal Institute of Technology, Stockholm, Sweden (ivybopeng@kth.se, markidis@pdc.kth.se, vencels@kth.se, erwinl@pdc.kth.se), (2) Disciplinary Domain of Science and Technology, Swedish Institute of Space Physics, Uppsala, Sweden (andris@irfu.se), (3) Department of Mathematics, Centre for Mathematical Plasma Astrophysics (CmPA), KU Leuven, Belgium (jandeca@gmail.com,giovanni.lapenta@wis.kuleuven.be), (4) Cray Exascale Research Initiative Europe, UK (ahart@cray.com)

Abstract

We simulate space plasmas with the Particle-in-cell (PIC) method that uses computational particles to mimic electrons and protons in solar wind and in Earth magnetosphere. The magnetic and electric fields are computed by solving the Maxwell's equations on a computational grid. In each PIC simulation step, there are four major phases: interpolation of fields to particles, updating the location and velocity of each particle, interpolation of particles to grids and solving the Maxwell's equations on the grid. We use the iPIC3D code, which was implemented in C++, using both MPI and OpenMP, for our case study.

By November 2014, heterogeneous systems using hardware accelerators such as Graphics Processing Unit (GPUs) and the Many Integrated Core (MIC) coprocessors for high performance computing continue growth in the top 500 most powerful supercomputers world wide. Scientific applications for numerical simulations need to adapt to using accelerators to achieve portability and scalability in the coming exascale systems. In our work, we conduct a case study of using OpenACC to offload the computation intensive parts: particle mover and interpolation of particles to grids, in a massively parallel Particle-in-Cell simulation code, iPIC3D, to multi-GPU systems. We use MPI for inter-node communication for halo exchange and communicating particles. We identify the most promising parts suitable for GPUs accelerator by profiling using CrayPAT. We implemented manual deep copy to address the challenges of porting C++ classes to GPU. We document the necessary changes in the exiting algorithms to adapt for GPU computation. We present the challenges and findings as well as our methodology for porting a Particle-in-Cell code to multi-GPU systems using OpenACC.

In this work, we will present the challenges, findings and our methodology of porting a Particle-in-Cell code for space applications as follows:

- We profile the iPIC3D code by Cray Performance Analysis Tool (CrayPAT) and identify the most promising part to be ported to GPU;
- We use OpenACC to offload the computation intensive parts in particle mover and interpolation of particle to fields to GPU;
- We analyse the performance and implement necessary changes to the existing algorithms to adapt to GPU computations;
- We identify the difficulties in porting C++ class and template to GPU due to the needs of manual deep copy and also the difficulties in debugging the application on GPU;
- We identify new algorithms that are needed for optimizing computations on GPU and for minimizing data movement between GPU and CPU memory space;

Acknowledgement. This work was funded by the Swedish VR grant D621-2013-4309 and by the European Commission through the EPiGRAM project (grant agreement no. 610598. epigram-project.eu). The work on OpenACC also used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

References

[1] Stefano Markidis, Giovanni Lapenta, and Rizwan-uddin. Multi-scale simulations of plasma with iPIC3D. *Mathematics and Computers in Simulation*, 80(7):1509 – 1519, 2010.

[2] Top 500 supercomputer sites. http://www.top500.org/lists/. Accessed: 2014-12-01.

[3] The OpenACC application programming interface version 2.0, August 2013.

[4] Epigram project website. http://www. epigram-project.eu. Accessed: 2014-12-01.