OpenACC Implementation of Radiative Transfer Simulation Code

Ryohei Kobayashi[1], Norihisa Fujita[1], Yoshiki Yamaguchi[2], Taisuke Boku[1],
Kohji Yoshikawa[1], Makito Abe[1] and Masayuki Umemura[1]
Center for Computational Sciences, University of Tsukuba[1]
Faculty of Engineering, Information and Systems, University of Tsukuba[2]
Tsukuba, Ibaraki, Japan
kobayashi@cs.tsukuba.ac.jp

1 ABSTRACT

Graphics processing units (GPUs) offer good peak performance and high memory bandwidth. They have been widely used in high-performance computing (HPC) systems as accelerators. However, they are not suitable for all applications, and there are applications where they don't efficiently work on. One of such applications is multiphysics simulation. Multiphysics is defined as the coupled processes or systems involving more than one simultaneously occurring physical fields and the studies of and knowledge about these processes and systems. Therefore, multiphysics applications perform simulations with multiple interacting physical properties and there are various computations within a simulation, and GPU-non-suited ones can be included. Because of that, accelerating simulation speed by GPU only is quite difficult and this is why we try to combine GPU and FPGA and make the FPGA cover GPU-non suited computation. We call this concept Cooperative Heterogeneous Acceleration with Reconfigurable Multidevices (CHARM) and have been working on GPU-FPGA-accelerated computation for radiative transfer simulation in astrophysics as a proof of concept [1].

The implementation method of GPU-FPGA-accelerated computation is a mixture of CUDA and OpenCL programming, which means that the computation kernels running on GPUs are written in CUDA and those running on FPGAs are written in OpenCL. We do not write all computation kernels in OpenCL for the following three reasons. First, since most of the existing HPC applications are CUDA-based implementations, it is very burdensome for programmers to rewrite the entire code in OpenCL. Secondly, even if OpenCL is a platform that is designed to run applications in a heterogeneous environment, in order to use both GPUs and FPGAs at the same time, it is necessary to separately compile and link the computation kernels using the OpenCL compiler for GPUs and the OpenCL compiler for FPGAs. This is essentially the same as the CUDA and OpenCL programming environments. Finally, most of the GPUs used in HPC are made by NVIDIA, and it is not hard to imagine that it is easier to maximize the performance of GPUs by using CUDA, which is a programming model that follows the GPU architecture. For these reasons, we use a mixture of CUDA and OpenCL programming. On the other hand, such a multi-lingual programming imposes a heavy burden on programmers, and therefore, a programming environment with higher usability is required.

We are currently working on a programming environment that enables to use both accelerators in a GPU-FPGA equipped HPC cluster system with OpenACC. Since it is a directive-based programming model, we can specify to the compiler by directives which part of the application should be offloaded to which accelerator. In addition, Oak Ridge National Laboratory (ORNL) is developing a compiler that can write computation kernels for FPGAs as well as GPUs in OpenACC. We are currently collaborating with ORNL with the goal of realizing cooperative computation of both accelerators in a GPU-FPGA equipped HPC cluster system, and as part of this collaboration, we use the compiler being developed by ORNL to realize the high usability GPU-FPGA-accelerated computation described above.

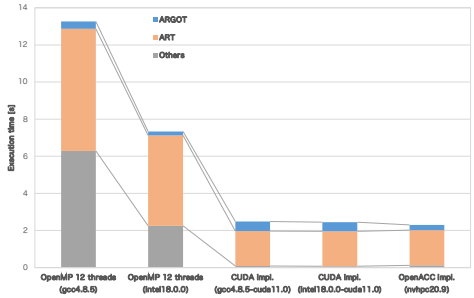

Figure 1: Performance comparison between OpenMP, CUDA, and OpenACC implementations. The problem size was 64^3.

Given the above background, we implement the radiative transfer simulation code with OpenACC and evaluate the performance by comparing with those of OpenMP-based CPU implementation and CUDA-based GPU implementation. The compilation software are GCC version 4.8.5 and Intel compiler 18.0.0 for OpenMP-based implementation, CUDA version 11.0, and NVHPC 20.9 for OpenACC-based implementation. Our experimental machine is the Cygnus supercomputer and we used a single computation node for the performance comparison. This is a heterogeneous platform composed of three types of devices: two Intel® Xeon® Gold 6126 CPUs, a four NVIDIA V100 GPUs for PCIe-based servers (Gen3 x16), and two BittWare 520N boards equipped with Intel® FPGA connected to the CPU through a PCIe Gen3 x16 interface. In this evaluation, we used a single CPU and GPU as located on the same CPU socket to avoid the performance degradation caused by a PCIe access over the Intel UPI (Ultra Path Interconnect). Fig. 1 shows the comparison result on a computation node of the Cygnus supercomputer and shows that there is almost no difference between the CUDA and OpenACC implementations.

REFERENCES

[1] Ryohei Kobayashi, Norihisa Fujita, Yoshiki Yamaguchi, Taisuke Boku, Kohji Yoshikawa, Makito Abe, and Masayuki Umemura. 2020. Multi-Hybrid Accelerated Simulation by GPU and FPGA on Radiative Transfer Simulation in Astrophysics. *Journal of Information Processing* 28 (2020), 1073–1089. https://doi.org/10.2197/ipsjjip.28.1073