Evaluating Performance Portability of HPC Applications and Benchmarks Across Diverse HPC Architectures

Date: April 13, 2022

Presented by: JaeHyuk Kwack (Argonne National Laboratory)

(The slides are available under “Materials from the Webinar” in the above link.)

Q. Why are we discussing old systems? Is this to just discuss the rationale behind the benchmarks? All these vendors, except for Nvidia, have all redesigned their cpu and gpu architectures. The 2nd decade belonged to Nvidia but it will not go their way in the 3rd decade.

- **A.** The first part was done in 2018, so the systems used in the study were pretty outdated. In the study of the first part, we were looking for a good methodology about how to define, measure and compare application performance across different HPC architectures. In this aspect, you may find some useful information. The main part of this talk was done in 2021, and we could use the latest GPUs from NVIDIA and AMD for 2021. Due to NDA, we unfortunately could not use data from the latest Intel XeHPC GPU. We will keep working on performance portability study on new architecture and we will keep HPC communities posted with newer data.

C. [From the chat] But I would argue these are the key DOE supercomputer systems we have *now*, and most applications don’t use them very well, certainly in a roofline sense.

- **A.** You are right. In a roofline sense, we don’t see all applications have high enough efficiency now. It means they don’t fully utilize memory bandwidth, or flop-rates of the systems. As we know, HPC applications have complex algorithms for their science; therefore, we can expect their performance bottlenecks can be something else such as cache performance, atomic operation, instruction throughput, memory latency, network bandwidth, I/O bandwidth, and so on.

Q. Why is TDP used for perf/Watt? For the GPU system a CPU is still needed to run the system which does not appear to be counted and TDP does not well correspond to wall power usage in the first place.

- **A.** I agree with you. TDP does not well represent actual power usage of applications. As you mentioned, it misses power usage of other components such as CPU for GPU applications, DRAM, NIC, and so on. In our earlier study, we just wanted to get approximated power efficiency of the applications as a rule of thumb. For more accurate power efficiency, we will need to use more elaborated tools (e.g.,GEOPM) to measure power consumption.

Q. I’ve browsed the [slides] deck and you have 5 apps. Why did you choose these? There is a suite of HPC apps we need benchmarks for. This will serve the whole industry if we can get an honest benchmark framework in place for Exascale. Let me see if I can get an image(s) for future reference. I’m complaining but I do see the value of where this presentation is going :-) (See end of the document for HPC Apps. I have more but if these apps could be benchmarked

or if a framework for benchmarking these applications is available, that would be a step forward.)

- **A.** Thanks for your comments. As you mentioned, it would be great if we could have more HPC apps in this study. This talk is about our effort to find out a good way to measure performance portability of HPC applications across diverse architectures. We were seeking more elaborated performance measurements than FOM (Figure-of-Merit) or execution time, so we couldn't start this study with a suite of HPC applications. Instead, we started with several applications popularly used at our LCF systems. We plan to extend our work with more applications, more portability layers, and in-depth analysis. We will keep you updated.

Q. I am assuming you were comparing 1 V100 vs 2 sockets of SKL and TX2 and one socket of KNL. As mentioned above even if the CPUs are not being used there is power drawn from the mainboard (CPU+memory+network) that is not measured in this manner when you normalize with TDP. A fairer comparison would use the V100 TDP and ¼ or ⅙ or ⅛ (depending on the configuration) of the dual socket CPU TDP. Of course actual power draw is far better as a measure - most codes do not get TDP level powers from either device (CPU/GPU).

- **A.** You are right. The actual power usages are different from TDPs, since there are other components (e.g., DRAM, NIC, and so on) and TDP is a power cap of a processor. In the comparison, we just wanted to get approximated power efficiency of the applications as a rule of thumb. For more accurate power efficiency, we will need to use more elaborated tools (e.g.,GEOPM) to measure power consumption.

Q. I may have missed this, was a unified programming model / compiler (OpenACC, OpenCL, etc.) used for the benchmarks?

- **A.** In our 2021 study, we used applications with portability layers such as SYCL, OpenMP target offloading, Kokkos, RAJA, and AMReX framework. With these portability layers, we measure application performance on AMD, Intel and NVIDIA GPUs with the same source base. We could not use applications with the OpenCL model, because NVIDIA tools don't provide performance data for OpenCL applications. The OpenACC model doesn't work on Intel GPU, so we could not have OpenACC applications in this study.

Q. Where do you see standard language features playing a role in performance portability for heterogeneous systems? I don't see it addressed.

- **A.** It is a good point. I think the ISO programming features can be very useful in performance portability for heterogeneous systems. Once these features are supported by multiple vendors and HPC communities, I believe HPC application developers will pick up these features for their application development. As we have experienced with our current programming models (e.g., OpenMP, OpenACC, OpenCL, CUDA, SYCL, HIP, Kokkos, RAJA, OCCA, and so on), I think it will take time. Once it becomes popular, we will try to understand their performance portability across architecture with applications.

Q. Beside GPUs, are there any other accelerators going into exascale systems?

A. In my understanding, we will have multiple GPU accelerated exascale systems in 2 to 3 years. I heard about continuous efforts with FPGA-based HPC architecture. I hope to see big scale systems with new accelerators like FPGA in the near future.

Q. Let's say that a code is benchmarked as a point on the roofline plot. When the same code runs on a different machine, will the coordinates of the point change on the roofline plot?

A. The arithmetic intensity (i.e., the x-coordinate) is a ratio of "flop count" over "memory traffic". Even for the same workload (i.e. the same source code with the same input), we saw different memory traffic across architectures, due to different size of L1 & L2 caches. Flop count can also vary across architecture, since math instructions (e.g., reciprocal, log, sine, sqrt, and so on) can be counted differently per architecture. As a result, the coordinates of a kernel on the roofline plot can be different across systems.

Q. Were any of the test codes Fortran?

A. In our work in 2018, GAMESS code on CPU was a Fortran application. In our recent work in 2021, we didn't have a Fortran application on GPUs. We know several teams with Fortran applications are porting their codes to GPUs. We plan to test them out on GPUs in the near future.

Q. Is there a tech report or paper where the details of this work are published?

A. This talk is composed of two conference papers. One is a conference proceeding of CUG 2019 (title: Roofline-based performance efficiency of HPC benchmark and applications on current generation of processor architecture, https://cug.org/proceedings/cug2019_proceedings/includes/files/pap115s2-file1.pdf). The other is a conference proceeding of P3HPC workshop at SC21 (title: Evaluation of performance portability of applications and mini-apps across AMD, Intel and NVIDIA GPUs, https://scwpub21:conf21%2f%2f@conferences.computer.org/scwpub/#!/toc/14).

Q. Will we get FLOP counts from AMD tools anytime soon?

A. AMD plans to release a reliable tool for FLOP counts for MI-250 GPUs soon. The AMD team will present it at 2022 ECP annual meeting (Tutorial title: Performance tuning with the roofline model on GPUs and CPUs, https://whova.com/portal/webapp/ecpan_202205/Agenda/2237005)

Q. What kind of analysis/profiling tools do you think will be helpful for HPC developers to improve performance portability?

A. The roofline performance analysis and corresponding tools (i.e., Advisor, Nsight Compute, Rocm profilers) presented in this talk can be a good starting point. Via this process, you can characterize the kernels as compute bound or memory traffic bound kernels. For kernels with low roofline efficiency, you can consider in-depth analysis to find out other performance bottlenecks such as cache performance, atomic operations, instruction throughput, latency, and so on.

Q. When do you think we'll see you next version of your presentation. All these new systems are due to be delivered in the [next] 1-3 years but you will get to see some aspects of these systems in non-production environments. I'm sure HPC software vendors would love to get their apps profiled.

- **A.** Thanks a lot for your interest. We plan to share our updated work via a workshop at SC22. We plan to use newer hardware (hopefully, we will have more new hardware publicly available then), more applications, and additional steps for in-depth performance analysis.

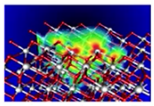
Q. Could you comment on PPM vs programming models like hip/openmp/openacc/etc ?

- **A.** As you probably know, each application has different performance characteristics (i.e., instruction type, memory traffic & access pattern, performance bottleneck, and so on) even on the same architecture. In this study, we didn't compare PPM of portability layers with the same workload, so we cannot comment on PPM vs. portability layers.

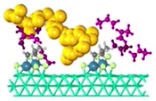
Q. You must have chosen problem sizes for each application that would fit in a GPU's memory - do you have a repo where these configurations can be found for anyone willing to try this on other GPUs?

- **A.** You can find the detailed configurations from our conference papers. One is a conference proceeding of CUG 2019 (title: Roofline-based performance efficiency of HPC benchmark and applications on current generation of processor architecture, https://cug.org/proceedings/cug2019_proceedings/includes/files/pap115s2-file1.pdf). The other is a conference proceeding of P3HPC workshop at SC21 (title: Evaluation of performance portability of applications and mini-apps across AMD, Intel and NVIDIA GPUs, https://scwpub21:conf21%2f%2f@conferences.computer.org/scwpub/#!/toc/14).

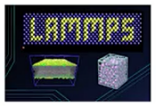
5 COMMON LIFE & MATERIALS SCIENCE HPC APPLICATIONS



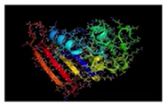
VASP*
Computes the material properties of molecules, solid-state compounds, and nanostructure systems at the atomic level.



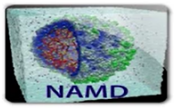
NWChem*
Focuses on chemical kinetics, dynamics of chemical transformations, and other chemical processes at the atomic level.



LAMMPS*
Calculates the properties of materials from liquids, gases, and gels through solid-state materials to mesoscale assemblies.



GROMACS*
Designed for high-performance simulation of large biomolecules, such as proteins, lipids, and nucleic acids.

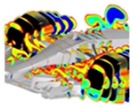


NAMD*
Simulates biomolecules in realistic environments. It uses VMD, the molecular graphics program, to run simulations.

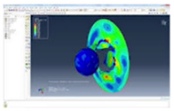
VASP (Compute & Memory Intensive)
NWChem (Memory Intensive)
LAMMPS (Compute Intensive)
GROMACS (Compute Intensive)
NAMD (Compute Intensive)

TOP 5 MANUFACTURING HPC APPLICATIONS

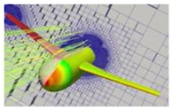
LS-DYNA*
Simulates complex real world problems so engineers can run through design options and test scenarios quickly



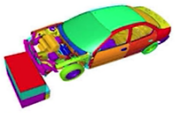
ANSYS Fluent*
Used for model flow, turbulence, heat transfer, and reactions for industrial applications



Abaqus*
Used for finite element analysis and multi-physics engineering simulations



OpenFOAM*
Open source CFD software for engineers and researchers for finite element analysis and multi-physics simulations



Radioss*
Front car crash refined model , can solve both linear and non-linear problems. Finite element solver using implicit and explicit integration schemes to solve engineering problems

Data-intensive HPC application and CAE simulation analysis workload combinations stress systems

LS-DYNA (Compute Intensive)
Fluent (Computer & Memory intensive
Abaqus (Compute & Memory Intensive)
OpenFOAM (Memory intensive)
Radioss (Compute & Memory Intensive)