Argonne NATIONAL LABORATORY

Performance portability evaluation of OpenCL benchmarks across Intel and NVIDIA platforms

C. Bertoni*, J. Kwack, T. Applencourt, Y. Ghadar, B. Homerding,
C. Knight, B. Videau, H. Zheng, V. Morozov, and S. Parker

Argonne National Laboratory

* Corresponding author, email: bertoni@anl.gov

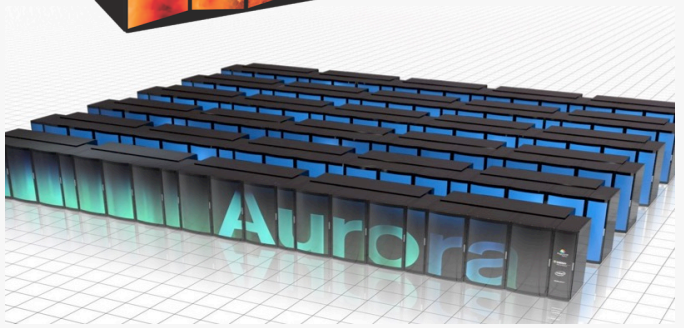
Motivation, Context, and Questions

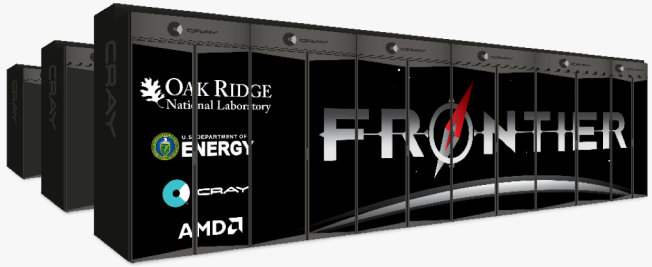
Pre-exascale and exascale landscape is diverse

- Aurora (Intel CPUs + Intel X^e GPUs) [1]
- Frontier (AMD CPUs + AMD GPUs) [2]
- Perlmutter (AMD CPUs + NVIDIA GPUs) [3]

For developers wanting to target all platforms:

- How will code developed on one architecture perform on another architecture?
- What programming model should I use to develop code?
- Is it worth the effort to port to a different programming model for every architecture and maintain separate branches of code?





1. https://www.anl.gov/article/us-department-of-energy-and-intel-to-deliver-first-exascale-supercomputer
2. https://www.olcf.ornl.gov/frontier/
3. https://www.nersc.gov/systems/perlmutter/

How to answer the questions?

Before we can answer the previous questions, we need to consider how to quantify the performance of codes across architectures

- In our case, we decided to use roofline-based performance efficiencies to compute **performance portability metric*** for a benchmark set across architectures

$$\text{PP}(a,p,H) = \begin{cases} \dfrac{|H|}{\sum_{i \in H} \dfrac{1}{e_i(a,p)}} & \text{if } i \text{ is supported } \forall i \in H \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

H: set of platforms
a: an application that solves problem *p*
$e_i(a,p)$: performance efficiency of application *a* solving problem *p* on platform *i*

This is the harmonic mean across platforms of the "performance efficiency" e_i of an application

*S.Pennycook, J.Sewall, and V.Lee, "Implications of a metric for performance portability" (2016)

How to answer the questions?

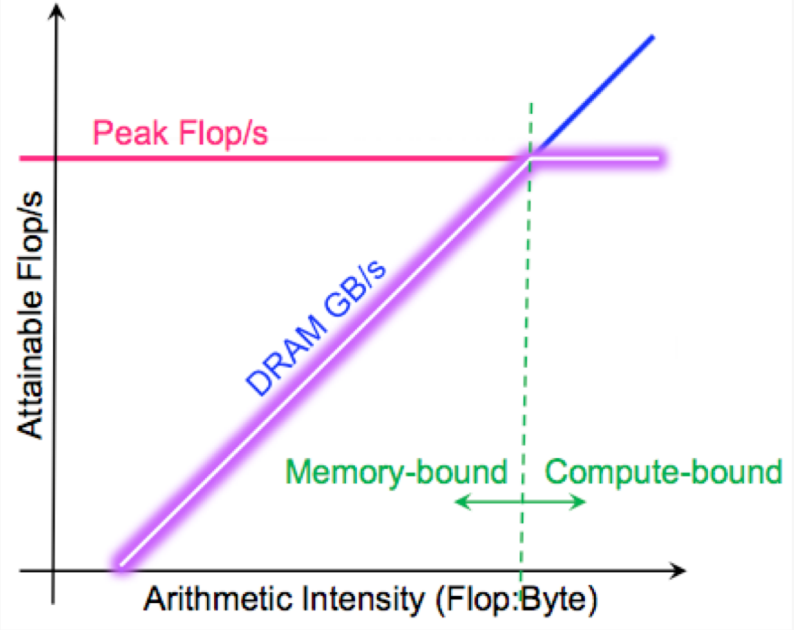
Before we can answer the previous questions, we need to consider how to quantify the performance of codes across architectures

- In our case, we decided to use **roofline-based performance efficiencies*** to compute performance portability metric for a benchmark set across architectures

Roofline-based Performance Efficiency: how close the application is to the ceilings in the roofline

$$e_i(a,p) = \frac{P_i(a,p)}{min\{F_i, B_i \text{ x } CI_i(a,p)\}}$$

For application *a* solving problem *p* on platform *i:*
$e_i(a,p)$*:* performance efficiency
$P_i(a,p)$*:* peak floating point throughput achieved
$CI_i(a,p)$*:* computational intensity
F_i*:* peak floating point performance *of i*
D_i*:* memory bandwidth of *i*

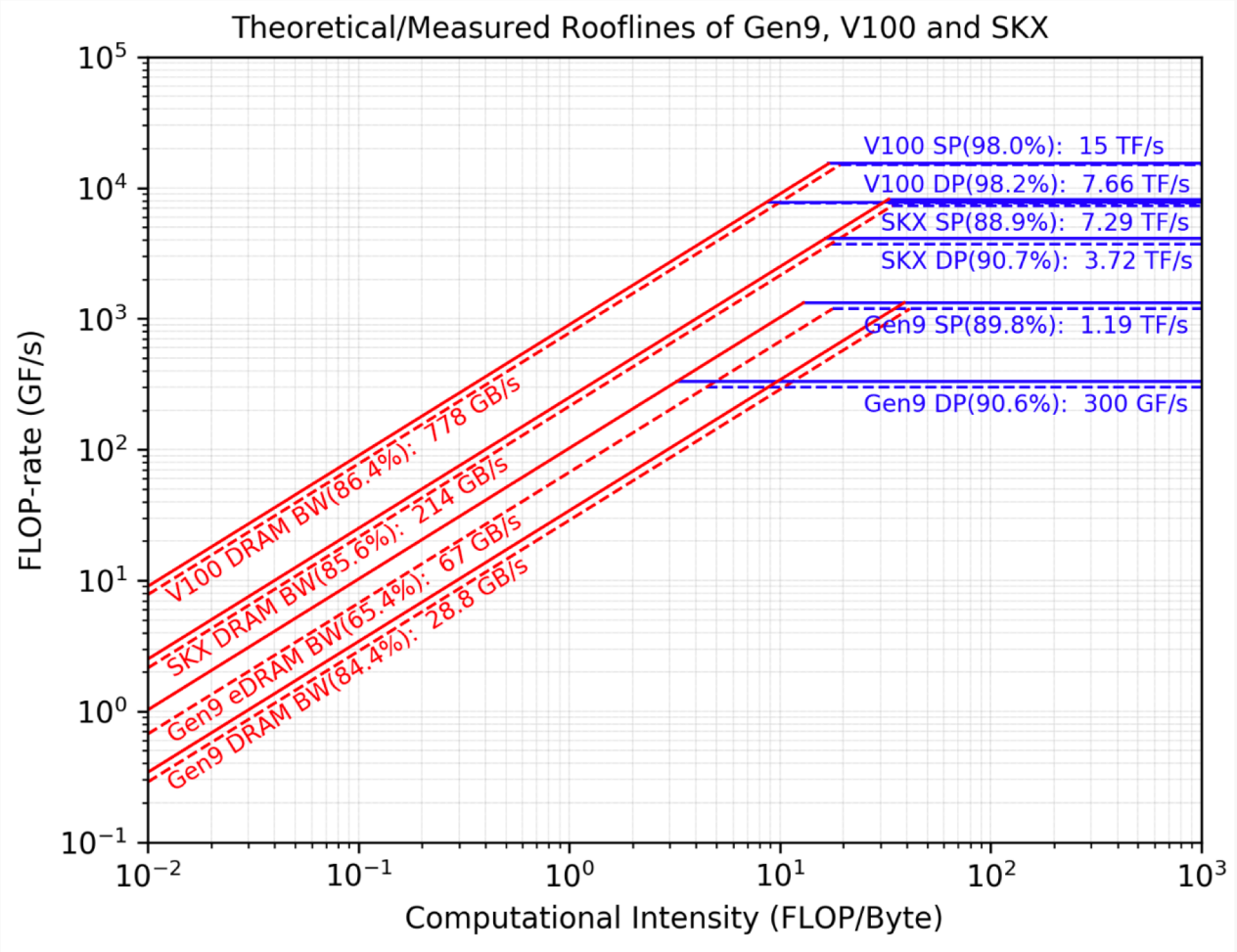


- C. Yang, R. Gayatri, T. Kurth, P. Basu, Z. Ronaghi, A. Adetokunbo, B. Friesen, B. Cook, D. Doerfler, L. Oliker, J. Deslippe, and S. Williams, “An empirical roofline methodology for quantitatively assessing performance portability.” (2018)
- https://ideas-productivity.org/wordpress/wp-content/uploads/2017/08/webinar010-Roofline-slides.pdf

Scope of this study

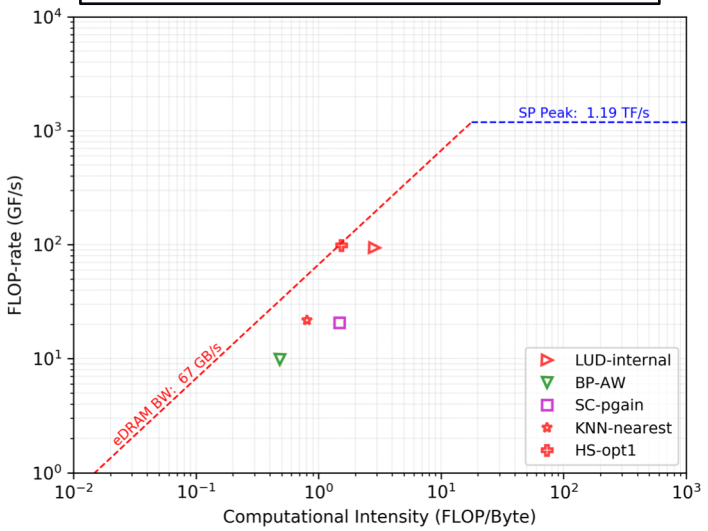
For this work we studied how the OpenCL implementations of the Rodinia Benchmarks performed across three platforms:

- Intel Gen9 GT4e integrated GPU
- Intel dual socket Skylake CPUs
- NVIDIA V100 discrete GPU

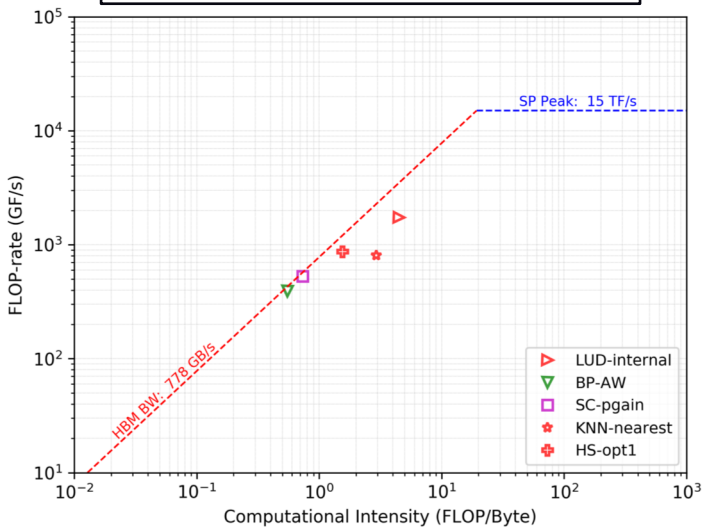


Roofline Analysis for each platform

EDRAM-based Roofline Plot on Gen9



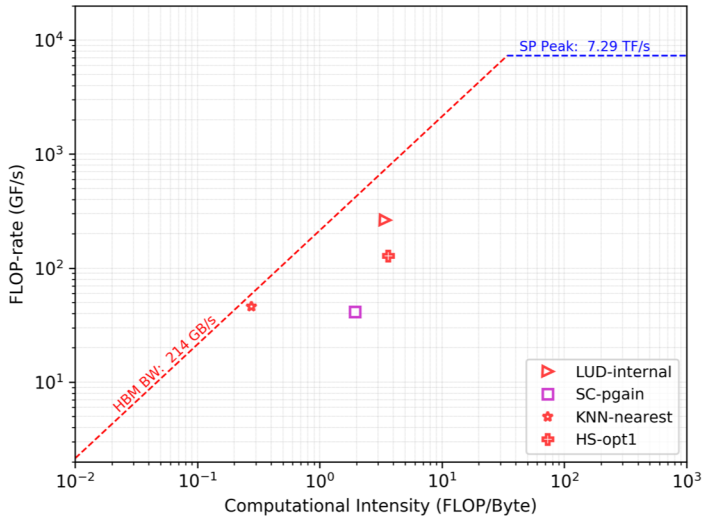
HBM-based Roofline Plot on V100



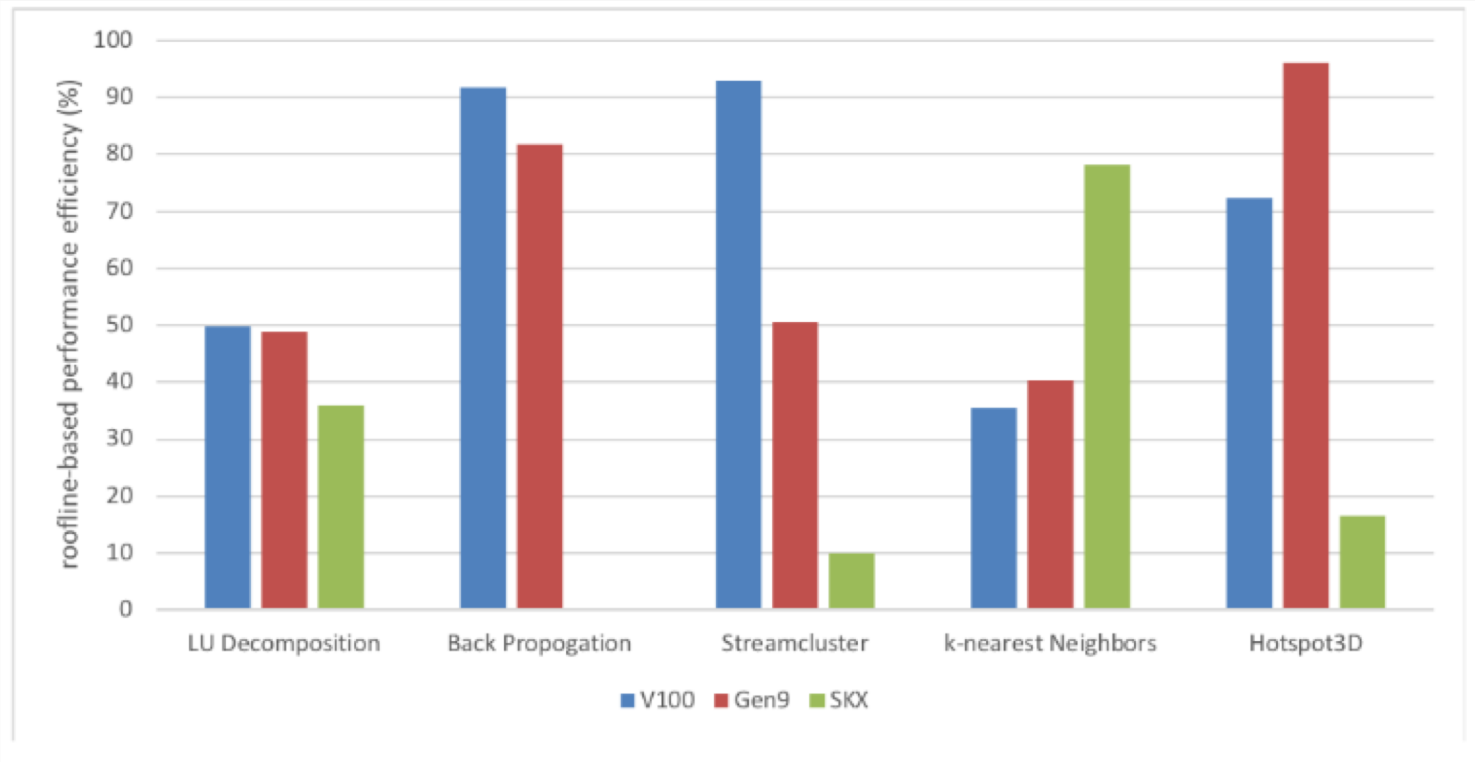
Measured Computational Intensities

Benchmark kernel	SKX	Gen9	V100
LUD-internal	3.428	2.870	4.473
BP-AW	–	0.512	0.549
SC-pgain	1.937	0.816	0.731
KNN-nearest	0.276	0.801	2.920
HS-opt1	3.621	1.536	1.546

DRAM-based Roofline Plot on SKX

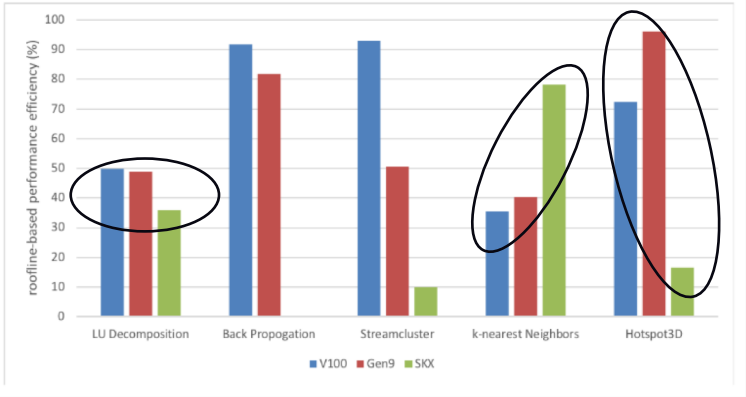


Roofline-based performance efficiencies



- Based on measured computational intensities, the roofline-based performance efficiencies are computed for each architecture and kernel

Computed Performance Portability Metric Values

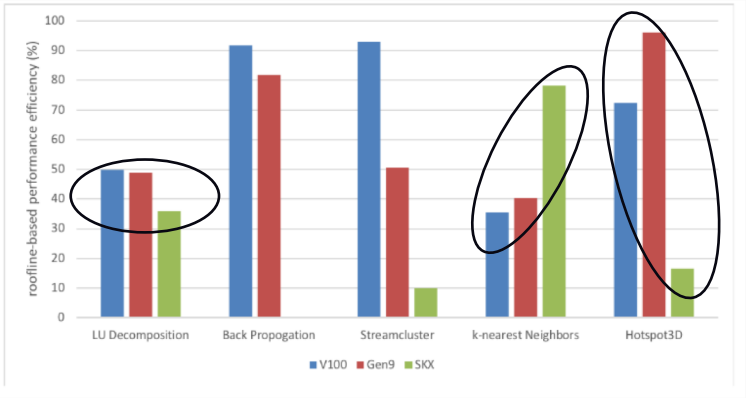


Benchmark kernel	Perf. portability (%)
LUD-internal	43.815
BP-AW*	86.419*
SC-pgain	22.947
KNN-nearest	45.617
HS-opt1	35.332

* Only considering the V100 and Gen9

- LUD-internal, KNN-nearest, and HS-opt1 have similar performance portability values
- However, we saw that spread of efficiencies across architectures is very different for each benchmark
- We see similar performance portability values because the performance portability metric is defined as a harmonic mean

Computed Performance Portability Metric Values



Benchmark kernel	Perf. portability (%)	Std. Dev.
LUD-internal	43.815	8.39
BP-AW*	86.419*	7.00*
SC-pgain	22.947	25.93
KNN-nearest	45.617	16.82
HS-opt1	35.332	35.1

* Only considering the V100 and Gen9

- To distinguish between instances where the performance portability is similar, it is useful to consider the **variability in performance efficiencies**
- To quantify this, we argue that the **standard deviation** for the performance portability metric should be included in performance portability studies.
- If you have a quantity that is defined as a mean, it's often useful to include a standard deviation to look at variability, too

Closing remarks and contributions

- We analyzed the performance and computed the **roofline-based efficiencies** of the Rodinia benchmarks on several architectures. In particular, roofline efficiency of kernels on Intel GPUs has not been evaluated previously in detail.
- We also quantified the **performance portability** of the benchmarks with a standard performance portability metric.
- The **standard deviation of roofline efficiencies** of the benchmarks across platforms is presented to further quantify their performance portability
- Finally, we discuss the issues encountered in quantifying performance portability