Towards Application-Agnostic HPC Profiling

Hari Teja Jajula
The University of Alabama
Tuscaloosa, AL, USA
hjajula@crimson.ua.edu

Dhruva Kulkarni
Lawrence Berkeley National Laboratory
Berkeley, CA, USA
dkulkarni@lbl.gov

Brian Austin
Lawrence Berkeley National Laboratory
Berkeley, CA, USA
baustin@lbl.gov

Purushotham V. Bangalore
The University of Alabama
Tuscaloosa, AL, USA
pvbangalore@ua.edu

Abstract

In modern HPC systems, performance metrics are collected at multiple levels. At the network level, telemetry includes packet counts for sent and received traffic, as well as counters such as latency distributions and packet size breakdowns. On GPU nodes, performance can be monitored with minimal overhead using NVIDIA Data Center GPU Manager (DCGM) counters[3]. The main research question is whether we can collect and use these metrics in an application-agnostic way to identify performance bottlenecks. Specifically, can such hardware-level data provide insights into application performance—comparable to those obtained from traditional instrumentation tools like Nsight Compute—without directly instrumenting the application code?

To address this, we leverage Perlmutter's existing Sandia-led Lightweight Distributed Metric Service (LDMS) component[4] and Performance Monitoring API to extract meaningful post-runtime job insights. Our approach focuses on determining whether telemetry data can be used to generate real-time, application-level performance views purely from hardware metrics. In particular, we explore generating a roofline heatmap [1] using only DCGM counters, in an application-agnostic manner. We use `dcgm_gr_engine_active` to filter for active samples, and `dcgm_fp64_active` to capture double-precision floating-point activity.

For each sample, we compute performance (y-axis) as:

$$\text{Performance} = (\text{fp64\_active} \times \text{Peak FP64 FLOPs})$$

and arithmetic intensity (x-axis) as:

$$\text{AI} = \frac{\text{fp64\_active} \times \text{Peak FP64 FLOPs}}{\text{dram\_active} \times \text{Peak DRAM Bandwidth}}$$

We validated these calculations using the Mixbench GPU microbenchmark [2], finding close agreement between our telemetry-based values and Mixbench results. The small differences observed were attributable to the sampling rate of DCGM counters rather than the computation method, confirming that the approach is accurate enough to reproduce roofline models directly from runtime telemetry without invasive instrumentation. This provides a path for continuous and application-agnostic performance characterization on production workloads.

Building on this GPU-side analysis, we extended our framework to incorporate network-level telemetry. Using Perlmutter's topology, we mapped compute nodes to their associated switches and developed a tool to automatically identify the set of switches relevant to any job. We then investigated whether NIC-level congestion signals—such as pause events—could be attributed to switch-level congestion. Our method compared pause signals observed on job nodes against those seen on neighboring nodes not associated with the job but connected to the same switch. If both groups exhibited similar pause activity, we inferred that the congestion originated at the switch rather than from the job itself. While we have not yet validated these inferences against switch-level counters, this approach provides a scalable framework for attributing performance bottlenecks to either job-level or network-level causes, which is critical for diagnosing problems in large-scale HPC environments.

To make the results accessible, we generate a summary report that converts raw DCGM and NIC counters into interpretable metrics. On the GPU side, the report includes the mean arithmetic intensity and mean performance (FP64), along with the proportion of memory-bound and active samples. It also summarizes SM activity, FP64 compute activity, GPU memory bandwidth utilization, memory usage, and power consumption, both at an aggregate level and for each individual GPU. On the network side, the report captures NIC utilization and imbalance by recording sent, received, and total packet counts for each interface, together with utilization percentages. These results are presented in the poster to highlight the combined insights from GPU and NIC telemetry.

In summary, this work demonstrates the feasibility of generating application-level performance insights directly from hardware metrics in an application-agnostic way. By validating against Mixbench, building GPU roofline heatmaps from DCGM counters, attributing NIC pause events to switch-level congestion, and presenting results in user-friendly summary reports, we bridge the gap between raw telemetry and actionable performance diagnosis. In conclusion, a parallel throughput check against `iperf3` showed that, because our LDMS deployment did not yet collect packet-size distribution counters, we had to assume an average packet size (8,192 bytes) to approximate throughput; this close-but-approximate match highlighted that collecting packet-size distributions is essential to identify the *type of packets* in flight and to recover throughput exactly. Future work will focus on validating NIC-based inferences against actual switch counters, integrating cross-layer analyses that combine GPU, CPU, and NIC metrics, and automating alerts for performance anomalies. This approach lays the groundwork for production-ready, low-overhead profiling tools capable of guiding optimization and scheduling decisions on current and future HPC systems.

Keywords

HPC telemetry, LDMS, DCGM, Cassini NIC, topology-aware profiling, roofline

Acknowledgments

I thank Dr. Sam Wellborn for discussions on iperf scripts for validation of NIC counters. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility (project m888-2025).

References

[1] Brian Austin, Dhruva Kulkarni, Brandon Cook, Samuel Williams, and Nicholas J. Wright. 2024. System-wide Roofline Profiling: A Case Study on NERSC's Perlmutter Supercomputer. In *Proceedings of the 15th International Workshop on Performance Modeling, Benchmarking and Simulation of High-Performance Computing Systems (PMBS'24) at SC Workshops*. IEEE, 1398–1404.

[2] Elias Konstantinidis and Yiannis Cotronis. 2017. A quantitative roofline model for GPU kernel performance estimation using micro-benchmarks and hardware metric profiling. *J. Parallel and Distrib. Comput.* 107 (2017), 37–56. doi:10.1016/j.jpdc.2017.04.002

[3] NVIDIA Corporation. 2024. *NVIDIA Data Center GPU Manager (DCGM) User Guide*. https://docs.nvidia.com/datacenter/dcgm/latest/user-guide Accessed: 2025-08-16.

[4] Sandia National Laboratories. [n. d.]. *Lightweight Distributed Metric Service (LDMS) Documentation*. https://www.sandia.gov/sandia-computing/high-performance-computing/lightweight-distributed-metric-service-ldms/ Accessed: 2025-08-16.