A Holistic View of Memory Utilization on Perlmutter

Jie Li
Department of Computer Science
Texas Tech University
Lubbock, TX, USA
jie.li@ttu.edu

George Michelogiannakis
AMCR
Berkeley Lab
Berkeley, CA, USA
mihelog@lbl.gov

Brandon Cook
NERSC
Berkeley Lab
Berkeley, CA, USA
bgcook@lbl.gov

Yong Chen
Department of Computer Science
Texas Tech University
Lubbock, TX, USA
yong.chen@ttu.edu

***Abstract*—HPC systems are at risk of being underutilized due to various resource requirements of applications and the imbalance of utilization among subsystems. This work provides a holistic analysis and view of memory utilization on a leadership computing facility, the Perlmutter system at NERSC, through which we gain insights about the resource usage patterns of the memory subsystem. The results of the analysis can help evaluate current system configurations, offer recommendations for future procurement, provide feedback to users on code efficiency, and motivate research in new architecture and system designs.**

***Index Terms*—HPC, Large-scale Characterization, Resource Utilization**

I. Introduction

The current resource allocation method in HPC allocates resources to applications in units of nodes where every node is identical. On the other hand, the demands of HPC applications vary significantly in terms of resource usage. The gap between the coarse-grained resource and the varying resource demands makes the system risk in substantially under-utilization. In this work, we perform a large-scale analysis of metrics sampled in NERSC's Perlmutter, a state-of-the-art HPC system containing both CPU-only and GPU-accelerated nodes. This data-driven analysis gives us a holistic view of how memory resources are used and what characteristics of applications have regarding utilizing the HPC memory resources.

II. Background

A. Perlmutter: System Overview

Perlmutter is being installed in two phases: phase 1 includes GPU-accelerated nodes and scratch file system, which is available in summer of 2021, and phase2 adds CPU-only nodes in 2022 [1]. The phase 1 system is currently ranked 7th in the top500 list [2]. Perlmutter has more than 1,500 GPU nodes and over 3,000 CPU nodes. Each GPU-accelerated node features four NVIDIA A100 Tensor Core GPUs and one AMD "Milan" CPU. The memory subsystem in each GPU node includes 40GB of HBM2 per GPU and 256GB of host DRAM. Each CPU node features two AMD "Milan" CPUs with 512GB of memory per node. The system uses SLURM version 21.08.8 for resource management and job scheduling.

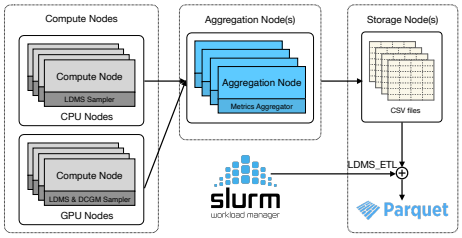
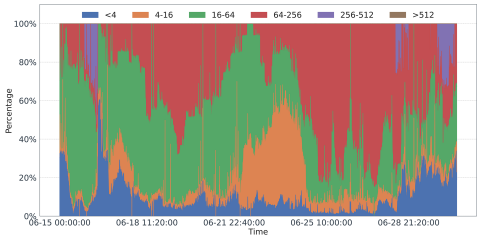
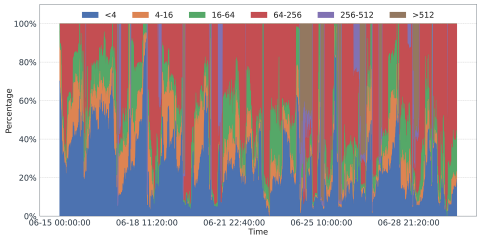


Figure. 1: Workflow of Monitoring Data Collection



(a) CPU Job size distribution

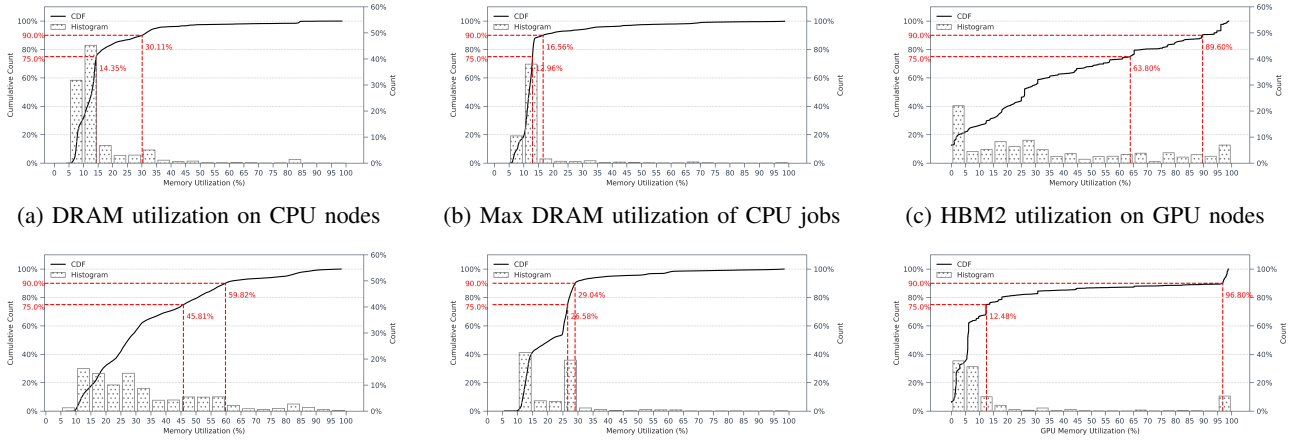


(b) GPU Job size distribution

Figure. 2: Job Size Distribution On CPU and GPU Nodes

B. Samplers and Metrics

The system-wide monitoring data are collected through LDMS [3] and DCGM [4]. The metrics are collected at an interval of 10 seconds. To support job-level analysis and improve the usability of the monitoring data, we use *LDMS_ETL* to join the raw monitoring CSV files with SLURM *sacct* data and write to parquet files for further analysis. The data collection and aggregation workflow is illustrated in Figure 1. For this study, we sample Perlmutter from June 15 to July 1, 2022 and we only focus on the memory resource utilization.



(a) DRAM utilization on CPU nodes (b) Max DRAM utilization of CPU jobs (c) HBM2 utilization on GPU nodes

(d) DRAM utilization on GPU nodes (e) Max DRAM utilization of GPU jobs (f) Max HBM2 utilization of jobs

Figure. 3: DRAM and HBM2 Utilization in Node-level and Job-level.

III. Analysis

A. Job Size Distribution

We divide jobs into six groups by the number of allocated nodes and calculate the percentage of each group in the total number of nodes; the results are shown in the area plots in Figure 2a and Figure 2b. In summary, 27.29% of jobs on GPU nodes request less than 4 nodes, much larger than the percentage of small-scale jobs on CPU nodes. Moreover, extremely large-scale jobs (request over 512 nodes) are only observed on GPU nodes, whereas large-scale jobs occupy only 2.96% of total jobs on CPU nodes.

B. Node-level Memory Utilization

Figure 3a and Figure 3d summarize the cumulative distribution function (CDF) of DRAM usage on CPU nodes and GPU nodes, respectively. From the figures we can see that, CPU nodes have their 90th percentile in DRAM utilization at 30.11%, indicating that less than 154 GB memory is used for 90% of the time. The long tail of the CDF in the Figure 3a indicates that the chances of a CPU node to exhaust its memory resource are low. For GPU host DRAM, it is used by almost 60% for 90% of the time, corresponding to 154 GB, which is similar to the DRAM used on CPU nodes, even their utilization rates differ significantly.

C. Job-level Memory Utilization

Figure 3b plots the CDF of the maximum DRAM utilization of CPU jobs; it shows two significant ranges, 5-10% and 10-15%, where the number of the corresponding jobs account for 20% and 70% of the total number of jobs. The CDF plot has a long tail, indicating that very few jobs will take up all the memory capacity on CPU nodes. For jobs on GPU nodes (Figure 3e), 90% of the jobs' maximum DRAM usage is no larger than 29.04% of the total memory capacity, i.e., 74 GB. The jobs using 10-15% and 25-30% of the total memory accounts for about 40% and 37% of the total number of jobs, respectively. The long tail is also observed in the CDF.

D. GPU Memory (HBM2) Utilization

Figure 3c plots CDF for HBM2 of GPU nodes. GPU HBM2 has a relatively balanced utilization across all time; for 90% of the time, the HBM2 utilization is nearly 90%, that is corresponding to 143 GB. Even though the memory utilization rate differs significantly (59.82% vs. 89.69%) at the same cumulative count between GPU host DRAM and GPU HBM2; their actual memory occupancy are close (153 GB vs. 143 GB). The job's maximum GPU HBM2 utilization is illustrated in Figure 3f. 75% of jobs have the maximum HBM2 utilization less than 12.48%, that is about 20 GB, which is only half of the HBM2 for each GPU. However, we observe about 10% of jobs use over 95% of the whole HBM2 capacity, making a spike at the end of the CDF line.

IV. Summary

In this work, we conducted a preliminary analysis on the memory resources usage of a leadership HPC cluster. Based on the analysis, we observed a significant amount of memory resources are under-utilized both on CPU nodes and GPU nodes. The DRAM utilization and the maximum memory utilization of CPU jobs indicate that the DRAM resources are over-provisioned. As the memory subsystem is one of the most expensive components in HPC that makes up over 20% of the total cost [5], it is desirable to reduce the memory capacity while having insignificant performance affects on jobs. The memory utilization on GPU nodes also reveals possibilities for optimizing memory configuration for GPU nodes. We observed that there are quite a significant amount of jobs cannot use GPU resources effectively. These under-utilized HBMs are ideal candidates for disaggregated memory that provide not only memory capacity but also high memory bandwidth for applications [6]. In our future work, we will continuously work on the memory resources analysis, especially considering the temporal and spatial characteristics. In addition, we will include metrics from other subsystems such as CPU, GPU and NIC to have a whole view of the resource utilization.

ACKNOWLEDGMENT

This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231.

REFERENCES

[1] NERSC. (2022) Perlmutter. [Online]. Available: https://www.nersc.gov/systems/perlmutter/

[2] Top500. (2022) Top500 list. [Online]. Available: https://www.top500.org/lists/top500/2022/06/

[3] A. Agelastos, B. Allan, J. Brandt, P. Cassella, J. Enos, J. Fullop, A. Gentile, S. Monk, N. Naksinehaboon, J. Ogden *et al.*, "The lightweight distributed metric service: a scalable infrastructure for continuous monitoring of large scale computing systems and applications," in *SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2014, pp. 154–165.

[4] NVIDA. (2022) NVIDIA DCGM. [Online]. Available: https://developer.nvidia.com/dcgm

[5] I. Peng, I. Karlin, M. Gokhale, K. Shoga, M. Legendre, and T. Gamblin, "A holistic view of memory utilization on hpc systems: Current and future trends," in *The International Symposium on Memory Systems*, 2021, pp. 1–11.

[6] G. Michelogiannakis, B. Klenk, B. Cook, M. Y. Teh, M. Glick, L. Dennison, K. Bergman, and J. Shalf, "A case for intra-rack resource disaggregation in hpc," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 19, no. 2, pp. 1–26, 2022.