An Interactive GPU Metric Dashboard for HPC clusters

Wentao Shi[2], Brandon Cook[1], Ronnie Chatterjee[1] , and Johannes Blaschke[1]

NERSC, Lawrence Berkeley National Laboratory, Berkeley, CA[1]

Department of Electrical and Computer Engineering, Louisiana State University, Baton Rouge, LA[2]

Scaling up programs to run at the scale of a modern high-performance computing (HPC) center can be a daunting task. One of the first questions developers ask is: “is my program using all the hardware available?” Many tools can extract detailed performance data on applications. But the level of detail that these tools deliver comes at a cost: significant resource and time must be invested in collecting and analyzing such performance data. But to answer a question like “am I using all 4 GPUs per node?”, this level of detail is overkill. So, this project aims to provide developer a user-friendly application to have a peek in their program’s GPU utilization with a novel combination of existing tools.

Introduction

The goal of this project is to develop a web application that displays the utilization of GPUs. Existing monitoring tools such as Lightweight Distributed Metric Service (LDMS) [1] provide means to retrieve metrics data from HPC clusters. However, it is not trivial to extract and exploit LDMS data. Having users parse the LDMS output is an undue burden on new users. So, this work also serves as a template for other HPC centers to implement similar tools. As this tool is available all the time, non-invasive, and scalable to system-sized jobs, we envision that the data provided gives users insight into when more detailed profiling is needed. Furthermore, first-time users of HPC systems can be overwhelmed by the complexity of performance profiling tools (e.g., Nsight Systems). We develop a simple tool such as these as a "gentle on-ramp" to more sophisticated performance profilers.

Data Processing Pipeline

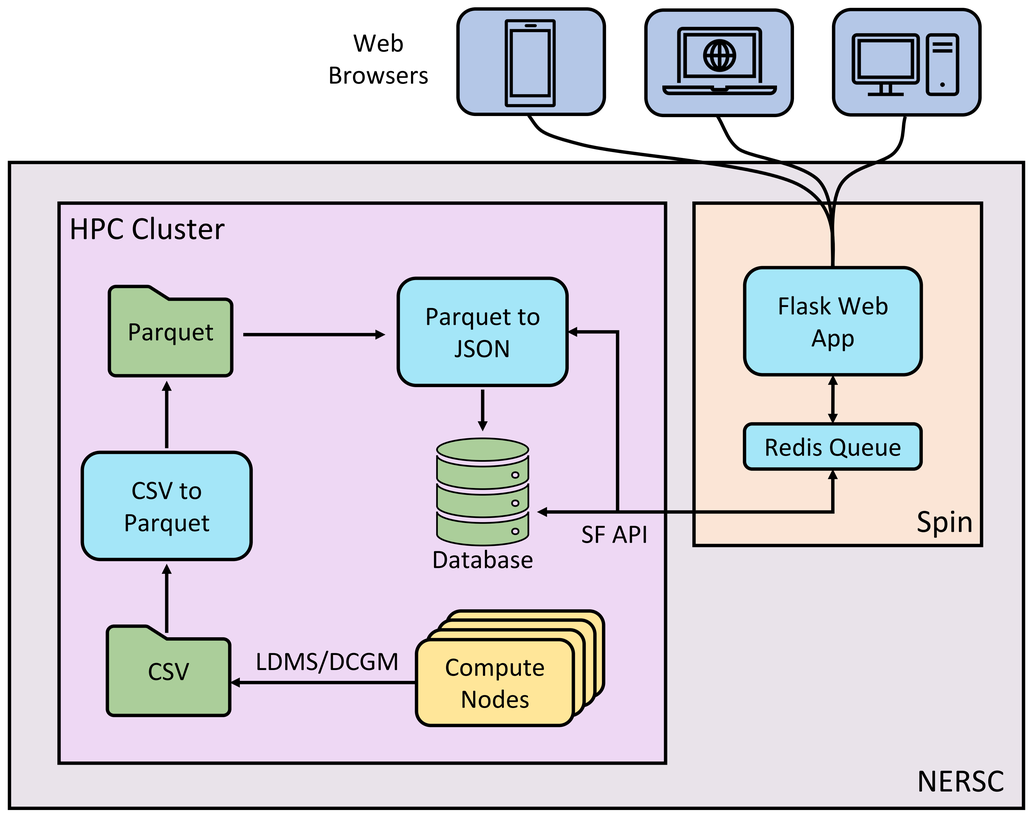
As illustrated in the right figure, the backbone of the application is a data processing pipeline deployed on the cluster. The processed data will be stored in a database and be fetched by the Flask web app later via the Super Facility (SF) API [5]. The users access the information by interacting with a web application deployed on Spin.

The initial stage of the data processing pipeline collects data from the compute nodes using LDMS. LDMS is integrated with Nvidia Data Center GPU Manager (DCGM) [2] to collect GPU metrics. The collected data are dumped to CSV files. Since the the CSV files are massive (at the scale of 100 GB), NERSC has developed a Python program to convert the CSV files into Apache Parquet [3] files which uses columnar storage technique. As a result, the file sizes are reduced approximately 10x to 100x depending on the property of data. Upon a user input, the Flask web app initializes an SF API call, then retrieves the data from a database via a second SF API call. Finally, the metric data are displayed in the web app.

User Interface

The only input the users need to provide is the Slurm job ID. After entering the job ID, the users can see a default plot in the web browser. The user can change the plot by re-selecting the part(s) of data to display. First, users can specify which node to display; all the nodes included in this job will be included in the selection. Second, users can select which metric to display; all the supported metrics are loaded, so users can quickly switch between different metrics. Lastly, users can specify which GPU(s) to display by checking the boxes on the left of the dashboard.

System Overview



Lightweight Distributed Metric Service (LDMS). LDMS is a scalable, lightweight, and system-wide monitoring tool for collecting system metrics. It collects high-fidelity data at low cost [1]. After being deployed the data collector instances of LDMS will be running on compute nodes to collect metrics, and data aggregator instances of LDMS will gather the collected data then dump them to CSV files.

Data Center GPU Manager (DCGM). DCGM is a software suite developed by Nvidia for managing and monitoring GPUs in cluster environments [2]. It is integrated with LDMS and the GPU metrics are stored as CSV files.

Apache Parquet File Format. The data are converted from CSV files to Parquet files for faster and more efficient access. Apache Parquet is a data format that leverages the columnar data storage technique, supporting efficient compressing and encoding schemes [3].

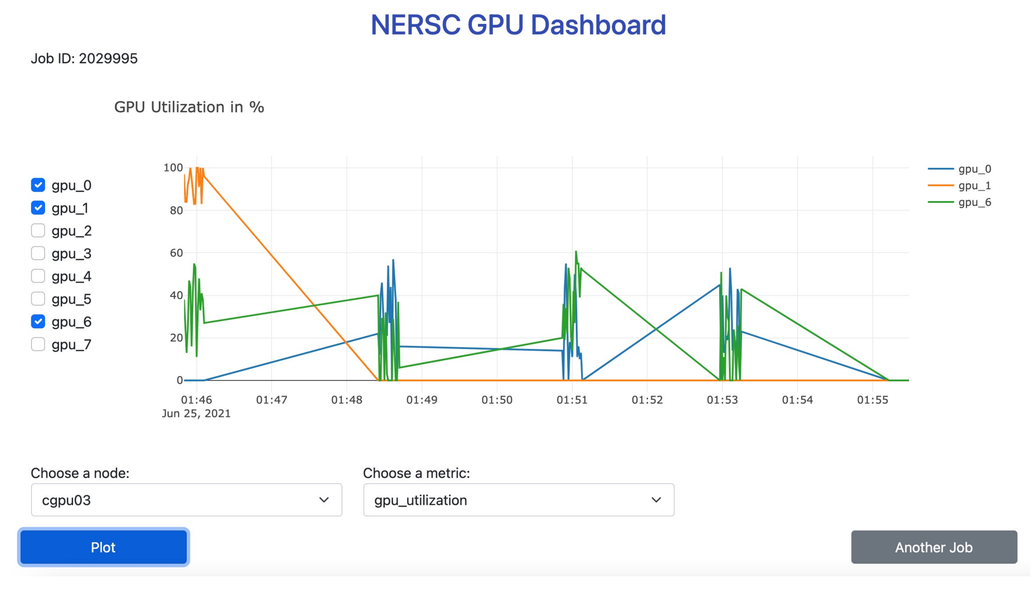
The GPU Dashboard Web App. In this project, the web application is built with the Flask [4] micro web framework written in Python. The webpages are rendered by the Jinja template engine that comes with Flask. The code that generates the interactive plots are based on Plotly.js.

The Super Facility (SF) API. The SF API is provided by NERSC to users for building applications that interact with the clusters [5]. In this project, our web app uses the “command” SF API to call the Python program that converts Parquet files to JSON. Then, the web app retrieves the JSON string according to the response by using the “task” SF API. SF APIs are available at https://api.nersc.gov/.

Redis Queue. Redis Queue is a job queue which is implemented based on Redis. The user requests are submitted to the queue and the Redis Queue workers fetch the jobs and access the SF API. This implementations provides a mechanism to prevent flooding of the server. Additionally, it enables the users to know their current positions in the queue.

Interactive Plot with Plotly.js. The interactive plot is implemented using Plotly.js [6]. Users can select different sections of data to visualize, zoom in, and zoom out.

User Interface



References

[1] Agelastos, Anthony, et al. "The lightweight distributed metric service: a scalable infrastructure for continuous monitoring of large scale computing systems and applications." *SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2014.

[2] *Nvidia DCGM*, https://developer.nvidia.com/dcgm. Accessed 20 July 2021.

[3] Vohra, Deepak. "Apache parquet." *Practical Hadoop Ecosystem*. Apress, Berkeley, CA, 2016. 325-335.

[4] Grinberg, Miguel. "Flask web development: developing web applications with python." *O'Reilly Media, Inc.*, 2018.

[5] Enders, Bjoern, et al. "Cross-facility science with the Superfacility Project at LBNL." *2020 IEEE/ACM 2nd Annual Workshop on Extreme-scale Experiment-in-the-Loop Computing (XLOOP)*. IEEE, 2020.

[6] Plotly Technologies Inc. Collaborative data science. Montréal, QC, 2015. https://plot.ly.

Contact

Johannes Blaschke: jpblaschke@lbl.gov

Wentao Shi: wentao771@gmail.com