UO-CS Graduate Research Open House

SIMON: A Simple Monitoring Framework for Heterogeneous System Observability

Allen Malony, Michael Dushkoff, Grace McLewee, Kevin Huck

Background

Observability: ability to make measurements of a system's operation in order to understand how and how well it performs when used by applications.

Crucial to explain system/application behavior and manage/improve operational outcomes.

The Problem

Most HPC performance measurement systems will collect detailed performance data and wait until the end of the application's execution to make it available.

Furthermore, current application monitoring technologies are limited, specialized, difficult to program, or hard to integrate with evolving HPC execution platforms.

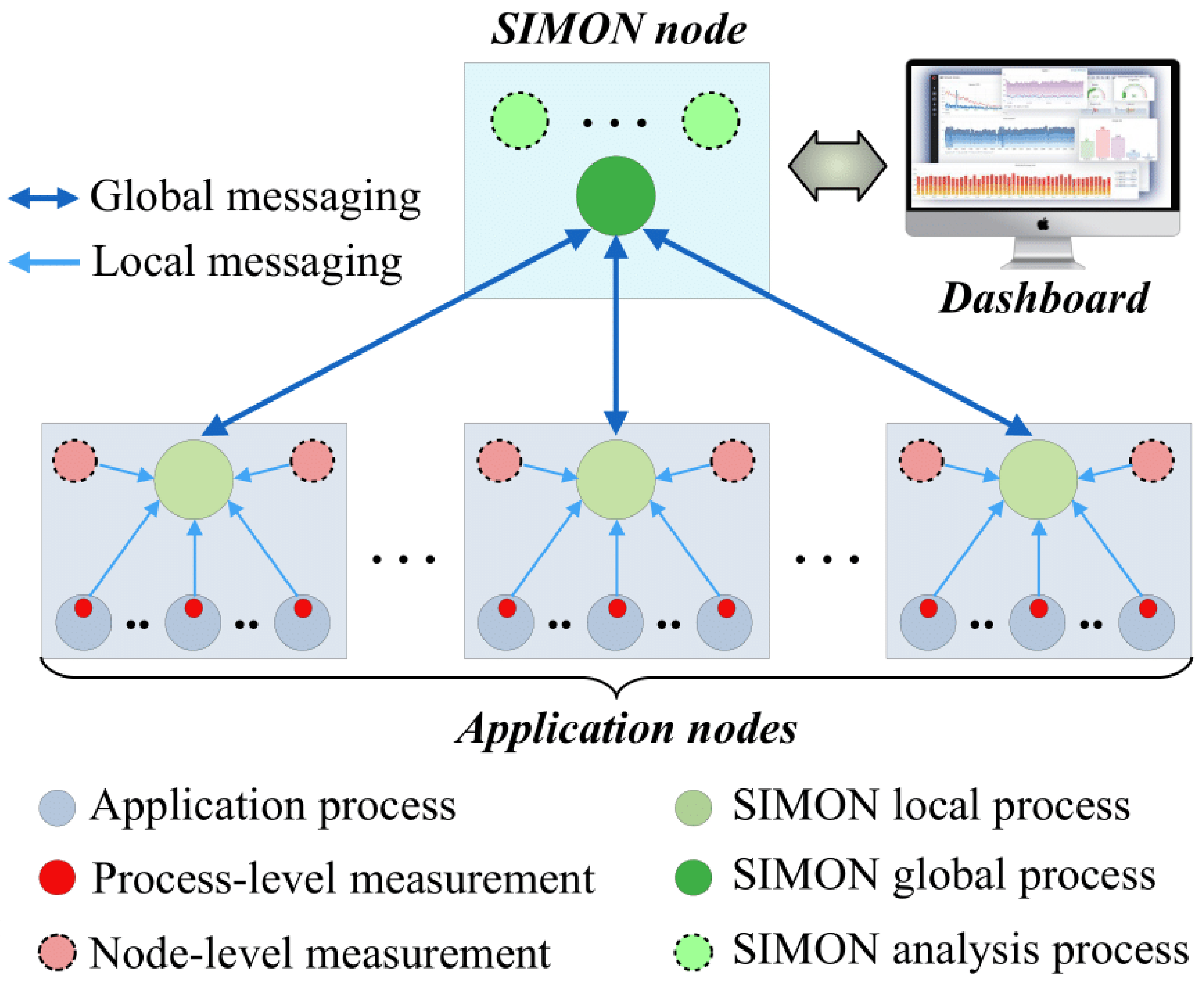
Solution

We present SIMON, a solution that prioritizes simplicity through:

- **easy to use** functionality
- operates **out of the box**
- **programmable**
- **extensible**
- **configurable**

Our Contributions:

- SIMON design approach and architecture;
- SIMON prototype developed with Python and ZeroMQ;
- demonstration and evaluation of SIMON prototype;
- development of a monitoring tool for XGC and ZeroSum; and
- evaluation of XGC monitoring tool on Frontier for up to 8 nodes and 64 ranks and with a real-time analysis dashboard



SIMON node

Dashboard

Global messaging

Local messaging

Application nodes

● Application process

● Process-level measurement

● Node-level measurement

● SIMON local process

● SIMON global process

● SIMON analysis process

SIMON Design Approach and Architecture

Assumes an observational model based on time-series telemetry and in situ processing, where measurement providers execute either with an application process or as a daemon process.

Providers send data to SIMON *collector* processes, which reside on the application node. To aggregate the node data, SIMON sends the measurement data to a single global *aggregator* running on a separate global node. Data processing can occur on the *collector* or *aggregator* process. Final analysis is produced by the *aggregator*.

Functionality of SIMON processes and data processing is entirely determined by the monitor system builder.

Prototype Development

Developed using ZeroMQ for transport and Python for telemetry processing to reduce software technologies in the stack and increase programming flexibility in constructing the monitoring system. Slurm is used to launch a SIMON monitor with an application.

SIMON Experiments

Used to build three monitoring systems for evaluation.

1. Distributed Top Monitor

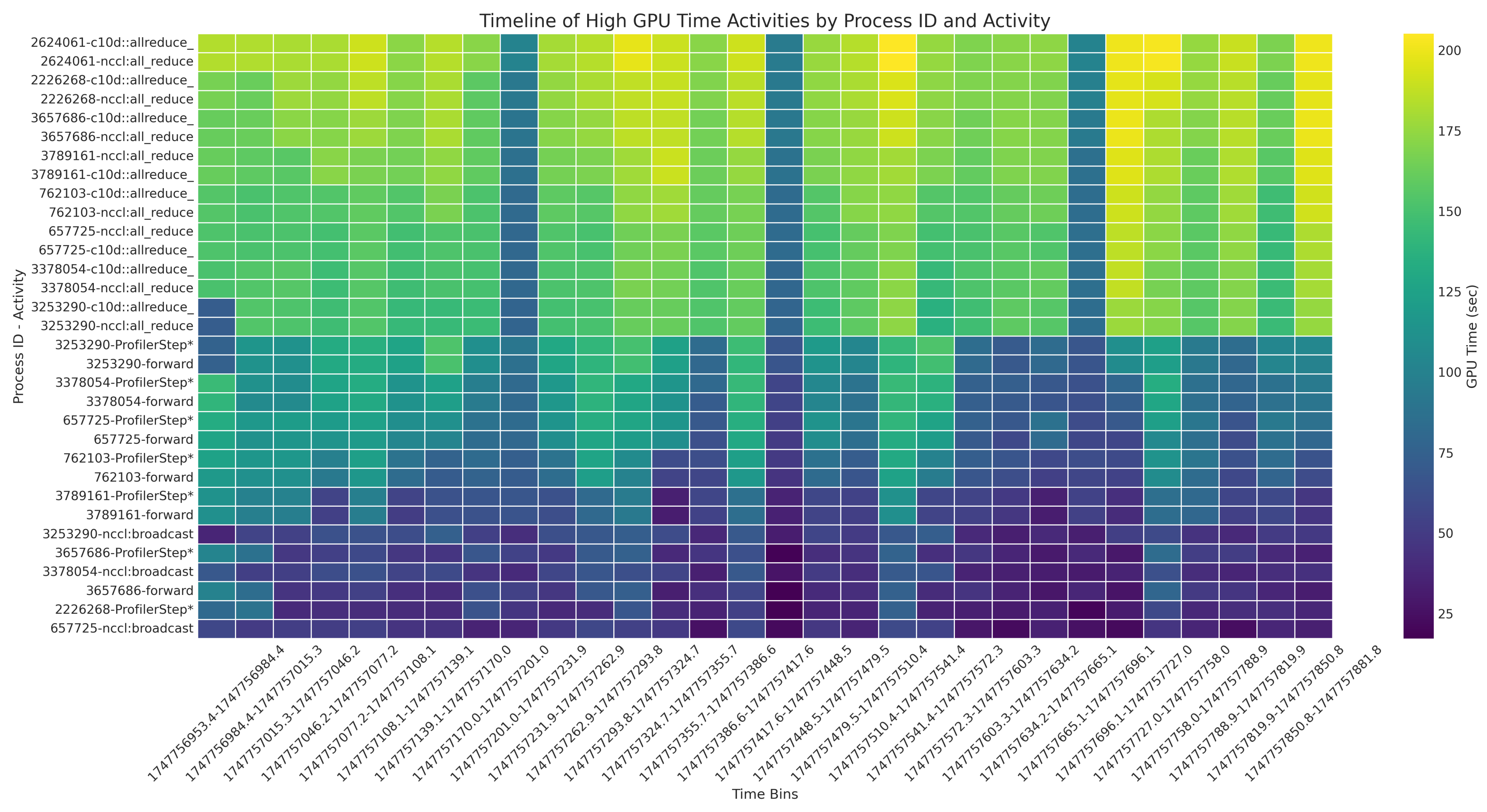
A basic SIMON monitoring test case that builds off the Linux top command. top data is collected across multiple nodes for a distributed heterogeneous application during execution.

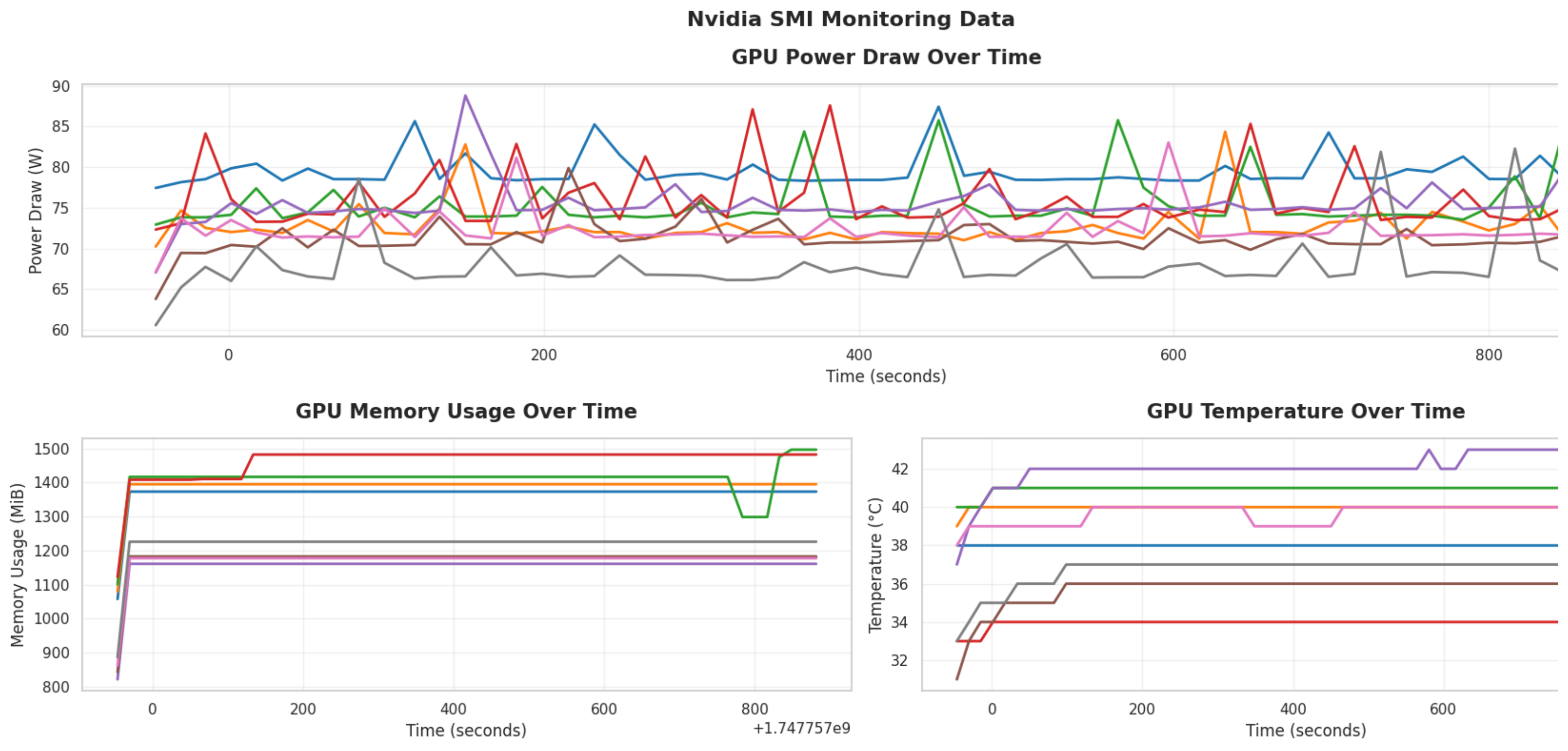
2. Distributed Pytorch Training

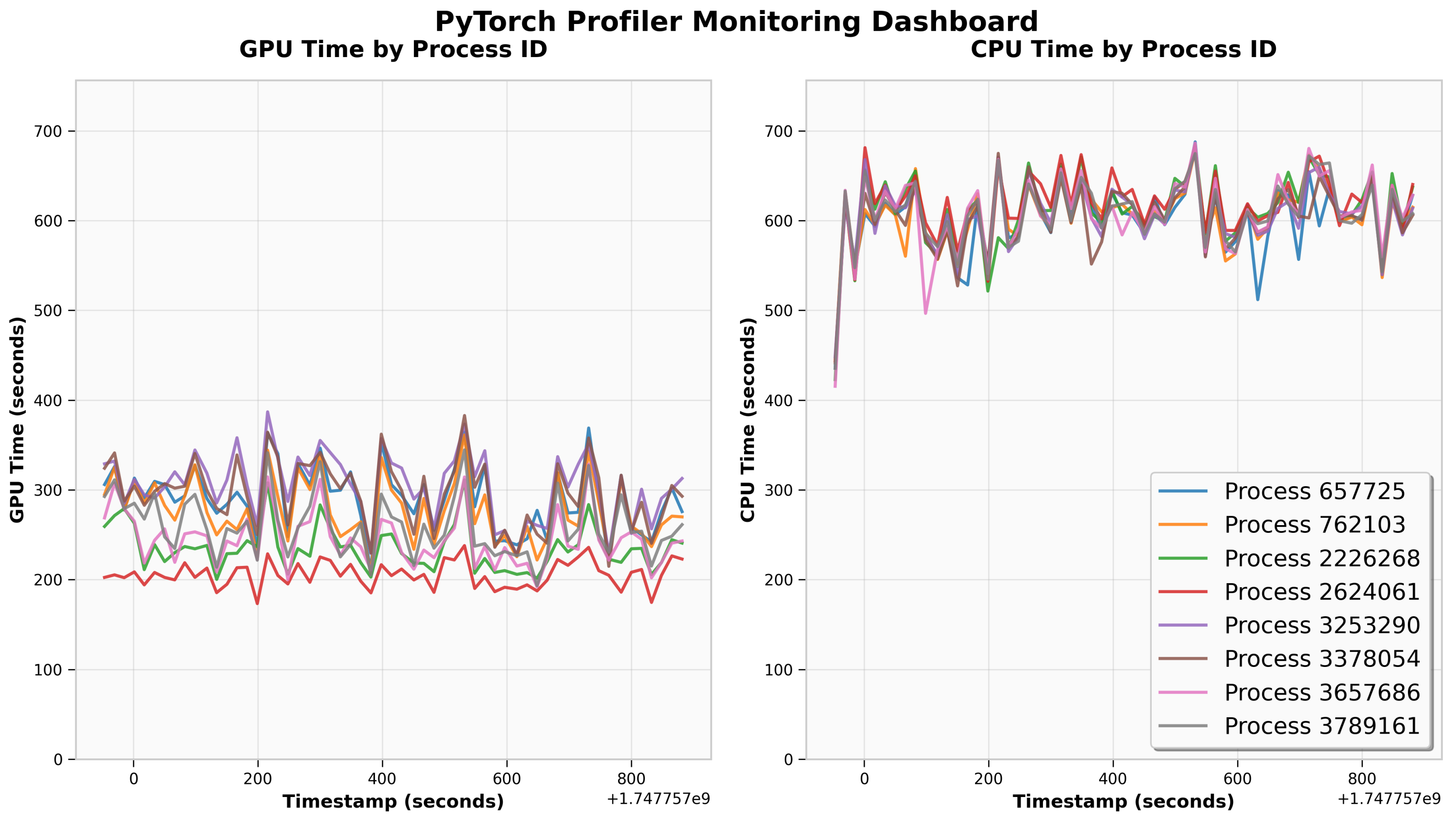
Our second example shows how SIMON can be used with an application where measurements are provided from data producers embedded in the application code with Pytorch Profiler and externally from NVIDIA System Management Interface. The training was done using 8 nodes with NVIDIA A100 80GB PCIe gpus.

3. XGC Monitor

A SIMON-based monitor was built to demonstrate how monitoring support can be integrated with a real application (XGC). We used ZeroSum to collect monitoring data every second of application execution.







UNIVERSITY OF OREGON