Combining Disparate Data Sources in the HPC Ecosystem

Alfredo Giménez, Todd Gamblin, Peer-Timo Bremer, Abhinav Bhatele, Martin Schulz

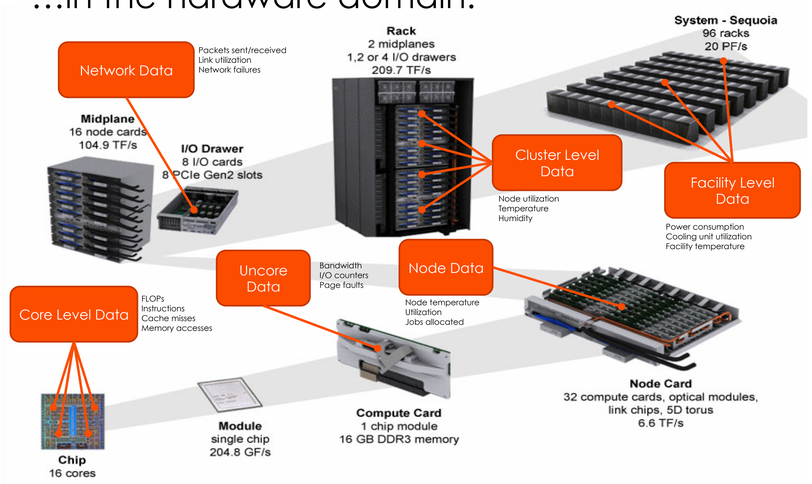
Abstract

Understanding and improving the performance and efficiency of HPC centers requires detailed analysis of running systems. To this end, modern HPC facilities provide extensive capabilities for collecting performance-related data for analysis. However, these data sources are most often disparate from one another, measuring different components in different domains. It is not clear, for example, how to correlate per-rack temperature readings with mesh input sizes recorded for a particular physics simulation.

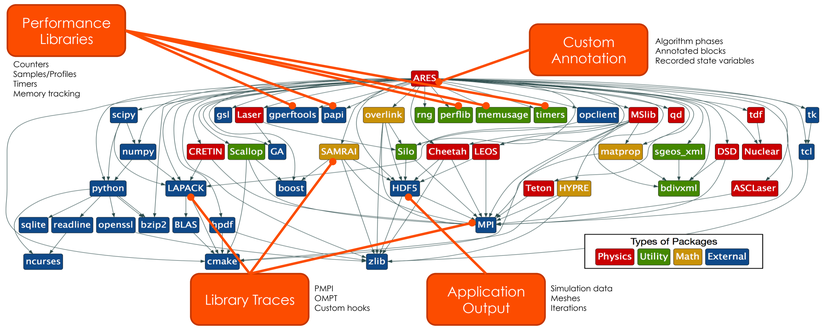
We are developing a performance analysis system that combines disparate data sources into a centralized database and automatically performs complex transformations on the data to yield indirect relationships between them.

HPC Performance Data Sources

…in the hardware domain:



…in the software domain:



Merging Disparate Data

Disparate data sources often require **more advanced merging** than a simple SQL JOIN operation.

Case: No one-to-one mapping

Time	FLOP Counter
10:00	6453
10:01	34
10:02	786
10:03	244556

Time	Temperature
10:01	55.6
10:03	58.2

Basic merge (JOIN)

Time	Temperature	FLOP Counter
10:00	UNDEFINED	6453
10:01	55.6	34
10:02	UNDEFINED	786
10:03	58.2	244556

Better merge

Time	Temperature	FLOPs
10:01	55.6	6453+34
10:03	58.2	244556+786

Case: Same domain, different units

T	FLOP Counter
54453	6453
64453	34
74453	786
84453	244556

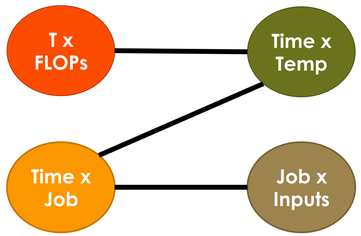
Time	Temperature
10:01	55.6
10:03	58.2

No columns to JOIN!

Solution: Semantic Table

Column	Units	Aggregator	Conversions
T	cycles	N/A	f(T) => Time
Time	HH:MM	N/A	f^{-1}(T) => T
FLOP Counter	count	Sum	none
Temperature	Celsius	Average	c(T) => Fahrenheit

This tells us: 1) **if** two data sources may be merged, and
2) **how** to merge them
And in turn: 3) **possible datasets** that may be produced by different sequences of merges (see below)



How do my application's FLOPs vary with different inputs?

We can perform 3 merges to obtain a new dataset with **Inputs x FLOPs**

1. **What is the best sequence of merges to perform?**
2. **How much error does each merge produce?**

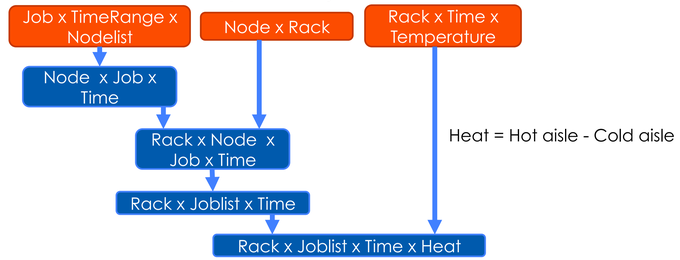
Preliminary Results

Dedicated Access Time (DAT) for 2 days on Cab (1296 nodes)

Data collected:

1. Job queue information (slurmq)
2. Facility temperature (9 sensors per rack, 23 racks)
3. Facility layout (assignment of nodes to racks)

How much heat is generated by different jobs?



Rack x Time x Heat x Joblist (sorted by heat)

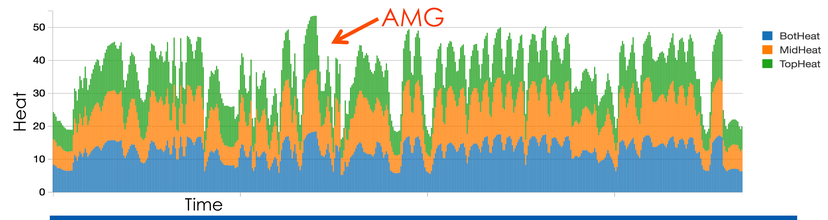
Rack	Time	TotalHeat	JobList
17	2015-08-05T22:44:00.000+0000	53.5423584	[('amg', 70)]
17	2015-08-05T22:42:00.000+0000	53.52637481999999	[('amg', 70)]
17	2015-08-05T22:40:00.000+0000	53.519630439999986	[('amg', 70)]
17	2015-08-05T22:38:00.000+0000	53.395622259999996	[('amg', 70)]
17	2015-08-05T22:36:00.000+0000	53.21107483	[('amg', 70)]
17	2015-08-05T22:34:00.000+0000	52.10317992	[('amg', 70)]
8	2015-08-05T22:42:00.000+0000	51.37504958999998	[('amg', 65)]
8	2015-08-05T22:40:00.000+0000	51.30201721	[('amg', 65)]
8	2015-08-05T22:44:00.000+0000	51.2350502	[('amg', 65)]
8	2015-08-05T22:38:00.000+0000	51.06603241	[('amg', 65)]
17	2015-08-05T22:32:00.000+0000	50.995285030000005	[('amg', 70)]
8	2015-08-05T22:36:00.000+0000	50.86448288	[('amg', 65)]

Left:

- Rack id, heat, list of running jobs (and number of nodes used)
- **AMG generated the most heat**

Below:

- **Heat** over time for rack 17
- Generated heat = temperature difference between hot and cold aisles



The SONAR Data Cluster

Above results used **>8GB data** (only 2 days worth)

Soon will be collecting continuous HPC performance data

- Power
- Temperature
- LDMS (counters on cores, uncore, and motherboard)

Need long-term massive storage, large-scale data processing

cassandra Spark

SONAR: newly deployed data cluster

- 13 nodes, SSDs, data software stack
- Apache Cassandra distributed database
- Apache Spark distributed data-local processing (used here)