Performance Analysis is Data Science

Presentation at HPC U. Ghent

Todd Gamblin
Computer Scientist
Center for Applied Scientific Computing

Feb 1, 2018

LLNL-PRES-745859
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

LLNL is a multidisciplinary national security laboratory

Experimental Test Site
(11 miles^2 near Tracy, CA)

- Established in 1952
- Approximately 6,000 employees
- 1 square mile, 684 facilities
- Annual federal budget: ~ $1.42B

The picture can't be displayed.

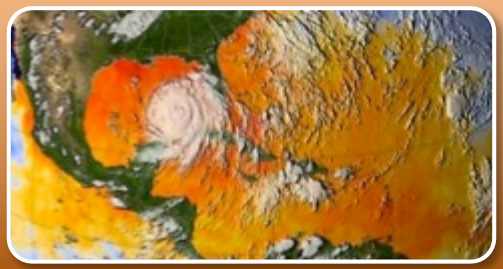
LLNL is focused on solving global security challenges for the nation

Multidisciplinary science, technology, and engineering – R&D that the private sector either can't do or won't do

Nuclear Security

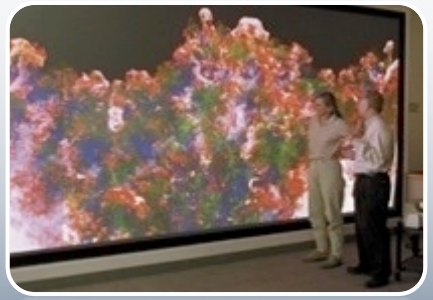
Domestic and International Security

Energy and Environmental Security



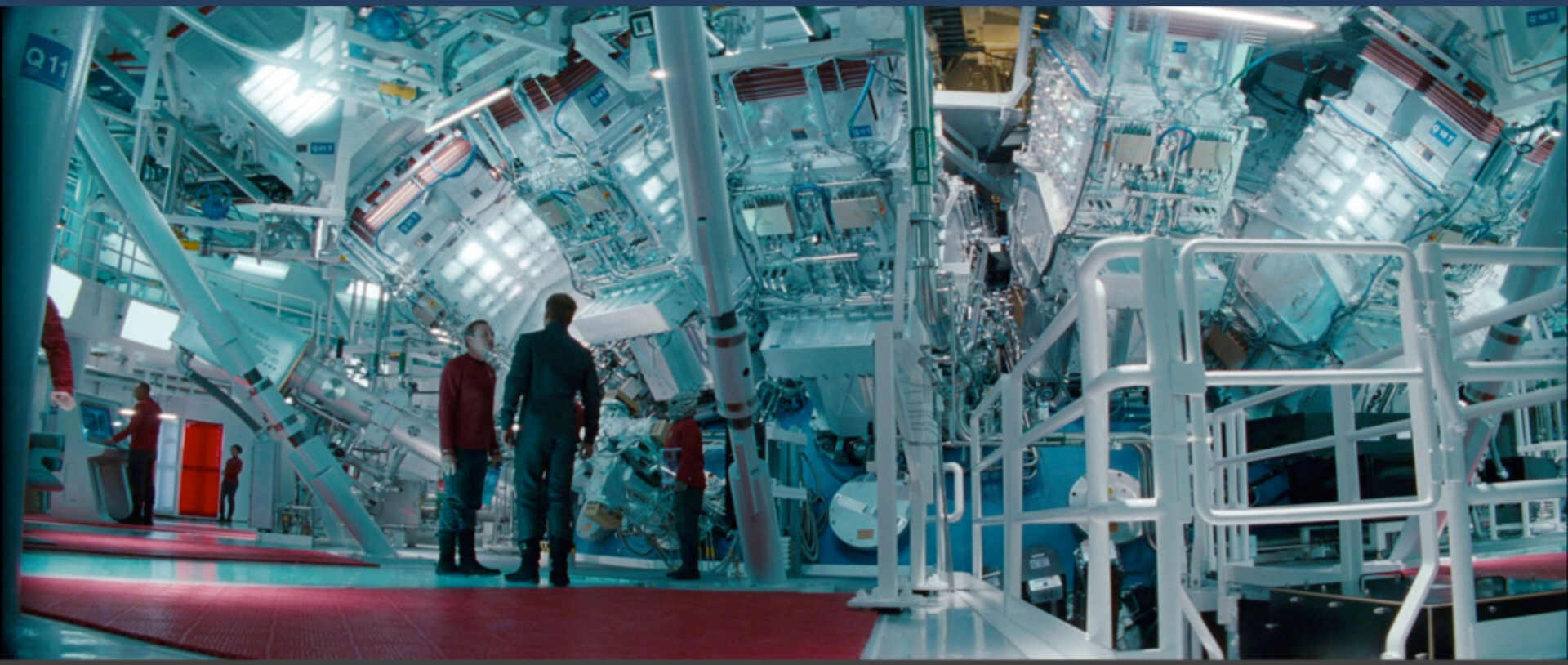
Applied Science

Engineering



Computing

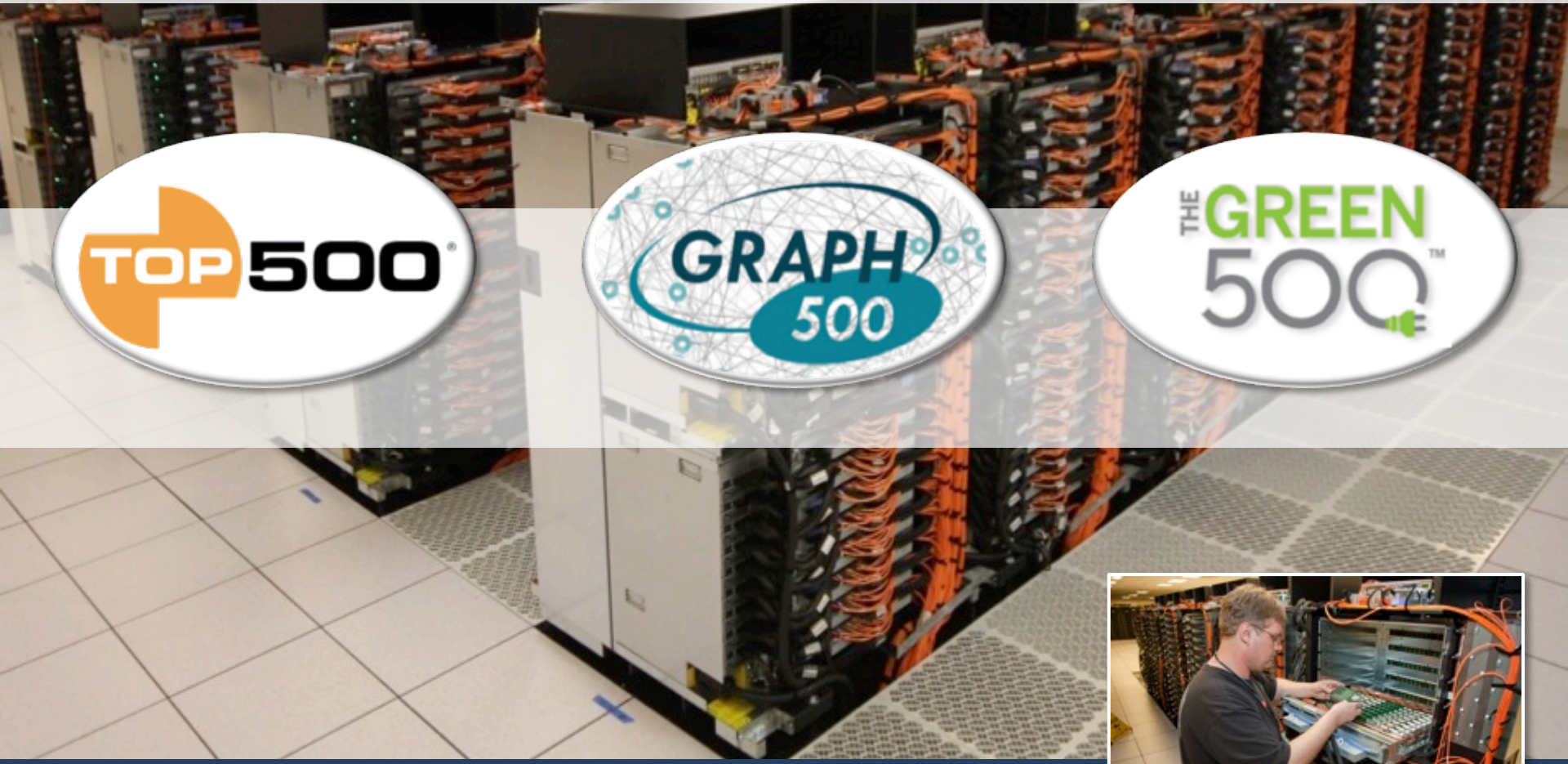
We combine world class experiments with large scale simulations



National Ignition Facility

explores the extremes of energy, temperature, and pressure that occur in stars, supernovae, and nuclear explosions.

High-Performance Computing (HPC) is part of the Laboratory's DNA

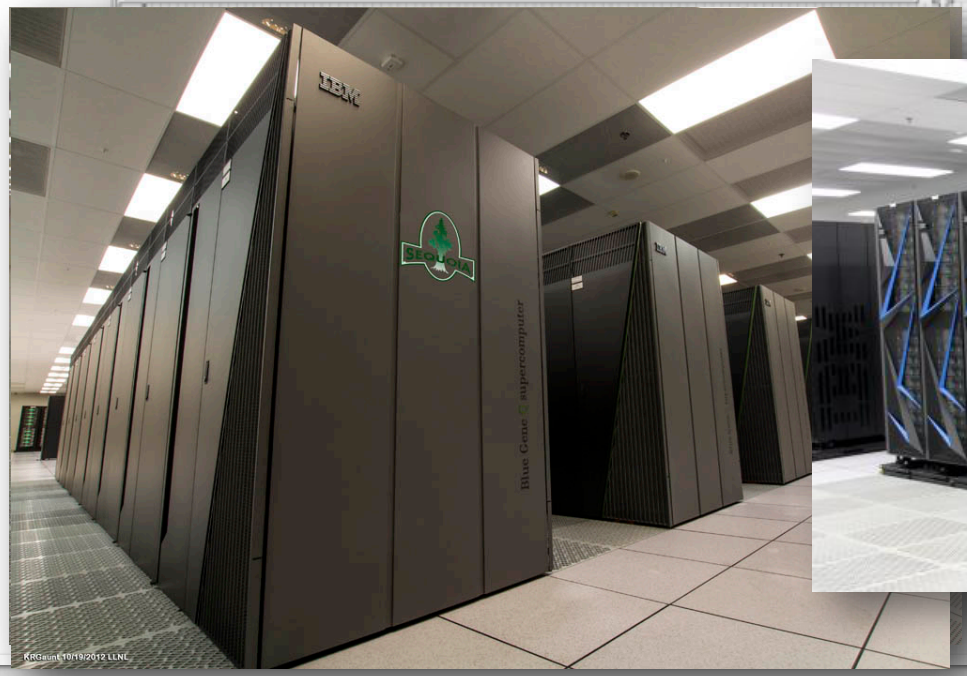


Throughout its history, LLNL has been at the forefront of enabling both the hardware and software of HPC.

The Livermore Computing Complex is massive

B453: 48,000 sqft

B654: 6,000 sqft (expandable)



Sequoia (16 PF)

Sierra (125 PF)

https://hpc.llnl.gov/hardware/platforms

The picture can't be displayed.

To optimize the Livermore Computing Complex, we need answers to several key questions

1. What is the workload?
2. How well *should* we expect a given application to perform on our machines?
3. How can we optimize the latency of a single application run?
4. How can we optimize the throughput of all jobs?
5. What machines should we purchase to handle our future workloads?

Answering performance questions is becoming increasingly complex

- **Problems:**
  - System variability complicates performance modeling
  - Runtime of two identical runs of the same code may differ by a factor of 2
  - Applications and workload change constantly
- **Many causes contribute to variability:**
  - **System level:**
    - Shared resources (Network, file system, memory bus)
    - Processor manufacturing variability
  - **Application level:**
    - Nondeterminism in applications
    - Data-dependent algorithms (input decks, multi-material, etc.)

It is increasingly hard to assess the characteristics of a "normal" run of an HPC application

We must enable automated analysis to understand the performance of our compute center

Data Management

Sonar Data Cluster (ISCP/ATDM)
Continuous monitoring from LC resources.
Applications contribute job data.
User-level security enables a shared database.

ScrubJay (ASCR / ATDM / ECRP)
Query/analysis system for Sonar data
Allows all users to access data

Data Analysis

Job Performance Prediction (ECRP / ASC)
Use monitoring data to predict job runtimes.
Feed back data to resource managers.

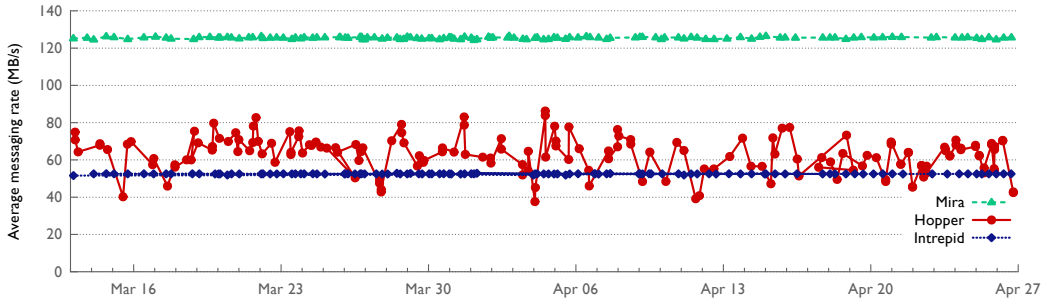
Apollo AMR auto-tuning (ECRP / ASC)
Performance portability for data-dependent apps
Select runtime strategies based on input

ECRP	ASCR Early Career Research Program	**ATDM**	NNSA Advanced Tech. Dev. & Mitigation
ASCR	ASCR XStack2 PIPER Project	**ISCP**	LLNL Institutional Sci. Capability Portfolio
ASC	NNSA Advanced Simulation & Computing		

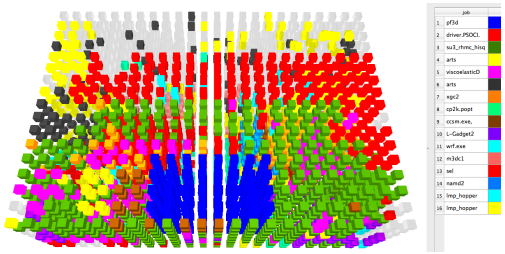
LLNL offers a unique environment where we can bring together research and programmatic work

Network contention can lead to severe performance degradation

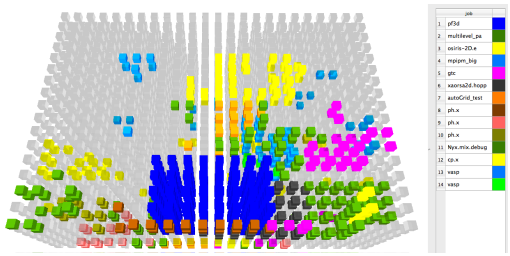
- We ran pF3D on BG/Q, BG/P, and Cray systems with I/O disabled
  - One run per day for 2 months
- Communication performance on Cray is affected by other jobs
  - Up to 2x slowdown
- Future machines will not have isolated networks like BlueGene
  - I/O performance is not isolated on BG/Q
  - Network is not the only source of nondeterminism.



Performance variability over time with and without network congestion. Blue Gene systems (Mira & Intrepid) have isolated per-job network partitions, while Cray XE6 systems use a shared network.



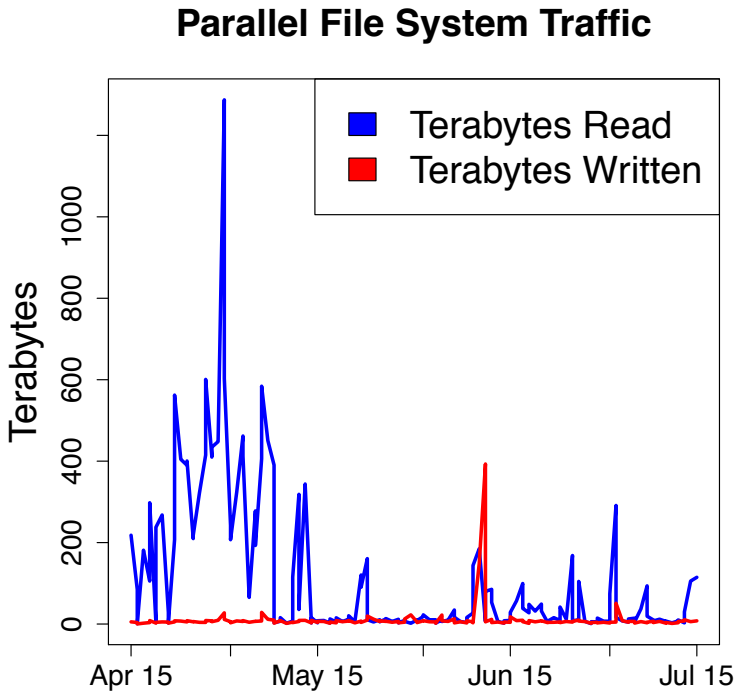
Slow run of pf3d on Cray XE6 system.



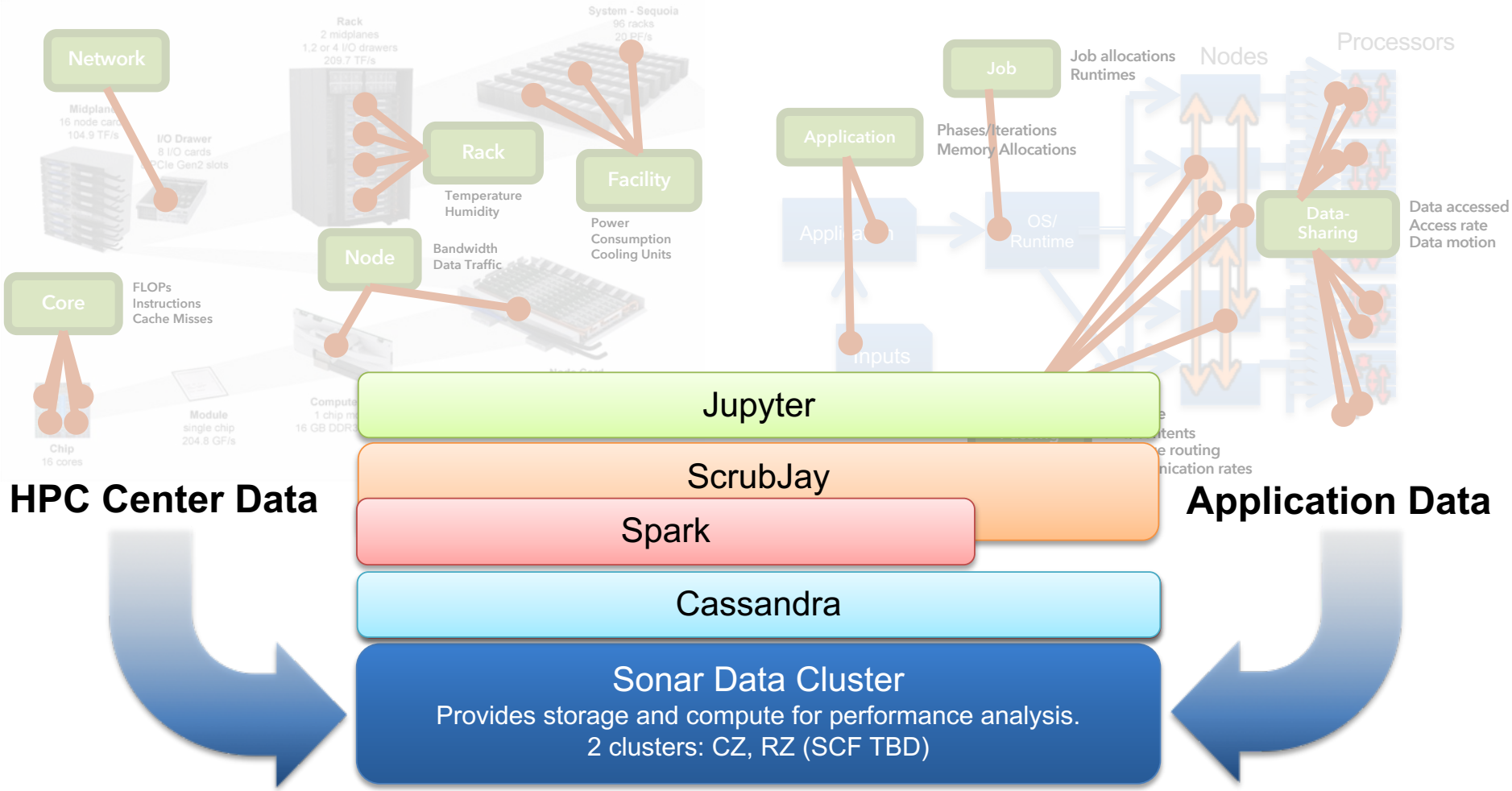
25% faster messaging rate without congestion.

File system load is also highly variable

- Lustre traffic over 4 months on a 1,296-node Sandy Bridge cluster (cab) at LLNL
  - Aggregate writes and reads to Lustre on LLNL clusters vary considerably over time.
  - Surprisingly, workload is very read-intensive
- Understanding a job's I/O performance depends on other jobs and other clusters
  - Particularly pronounced at LC, where file systems are shared



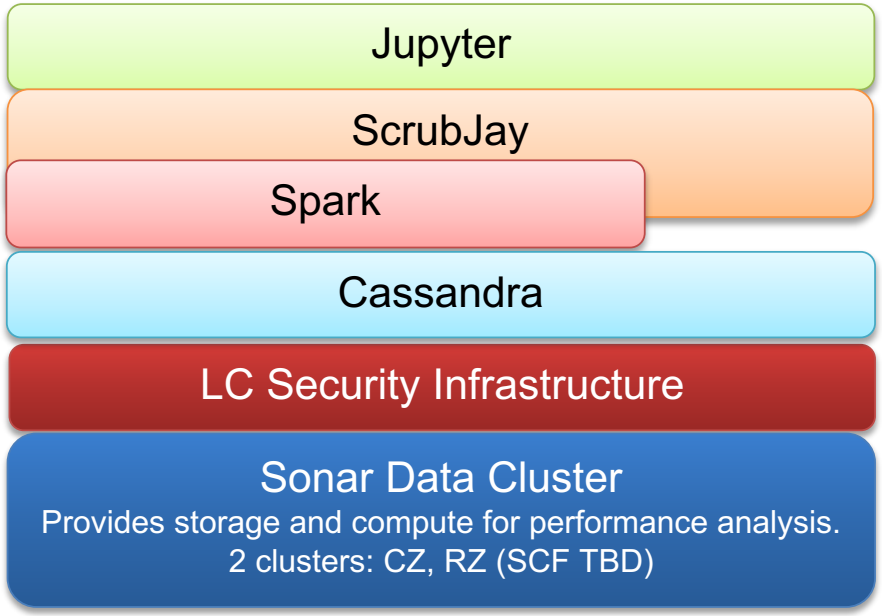
Sonar collects data from the HPC Center and applications, allowing users to access it with secure permissions



Sonar enables all LC users to research into the root causes of performance variation

Sonar allows users to access data previously available only to facility administrators

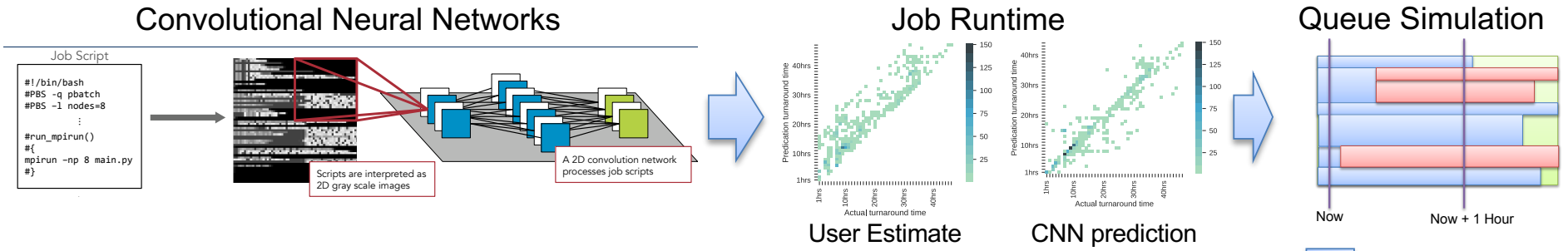
- *Some* data at Livermore Computing is sensitive
  - Path names, users, groups, detailed job information, etc.
  - Data about ECI, UCNI, and other sensitive codes
- Access to this data was previously all-or-nothing
- Sonar now integrates with LC's user/group permissions
  - Allows scientists in LLNL's research divisions to access data for the first time
  - Users can manage their own permissions in the system
  - Each table can have its own users, groups, and delegated management.
- Permissions will allow us to federate and manage a much larger volume of data, with more people using the system.



Sonar will allow us to create open data sets for HPC facility performance

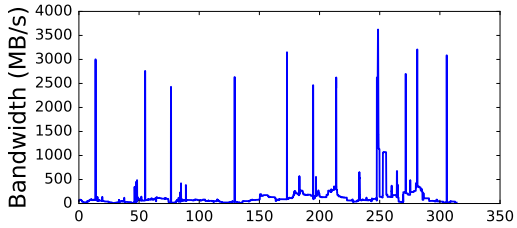
How can we use Sonar data to understand *facility* performance?

We combine neural networks with queue simulation to predict resource utilization

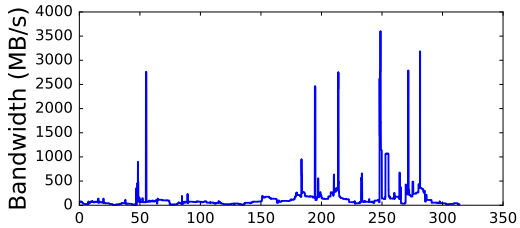


- CNN prediction:
  - Takes *only* job scripts and queue data as input
  - Leverage unstructured information (coding style, job scripts)
  - No preprocessing of job scripts required; fully automated
- Queue simulation:
  - Use predicted runtimes to simulate future job schedule
- We can predict many I/O bursts.
  - Some bursts can't be predicted b/c jobs enter the queue and run immediately
  - There may be periodicity or other patterns to these bursts
- We are investigating additional modeling techniques to predict bursts that cannot be simulated

Observed



Predicted (30 min horizon)



These results will provide input for resource-aware scheduling on the Flux project

How can we use Sonar data to understand *application* performance?

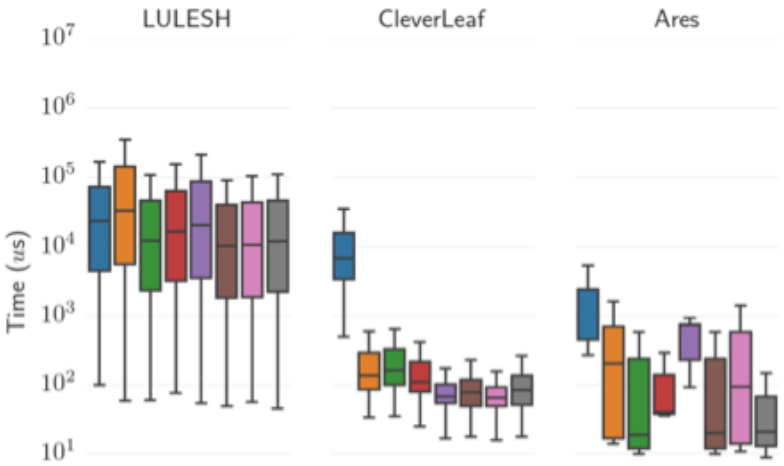
The right execution policy for a RAJA kernel depends on the data

RAJA-style loop

```
forall<exec_policy> (index_set , [=] (int i) {
     y[i] += a * x[i] ;
     tsum += y[i];
     tmin.min( y[i] );
});
```

Execution policy: scheduling, execution, OpenMP/CUDA/ other programming models

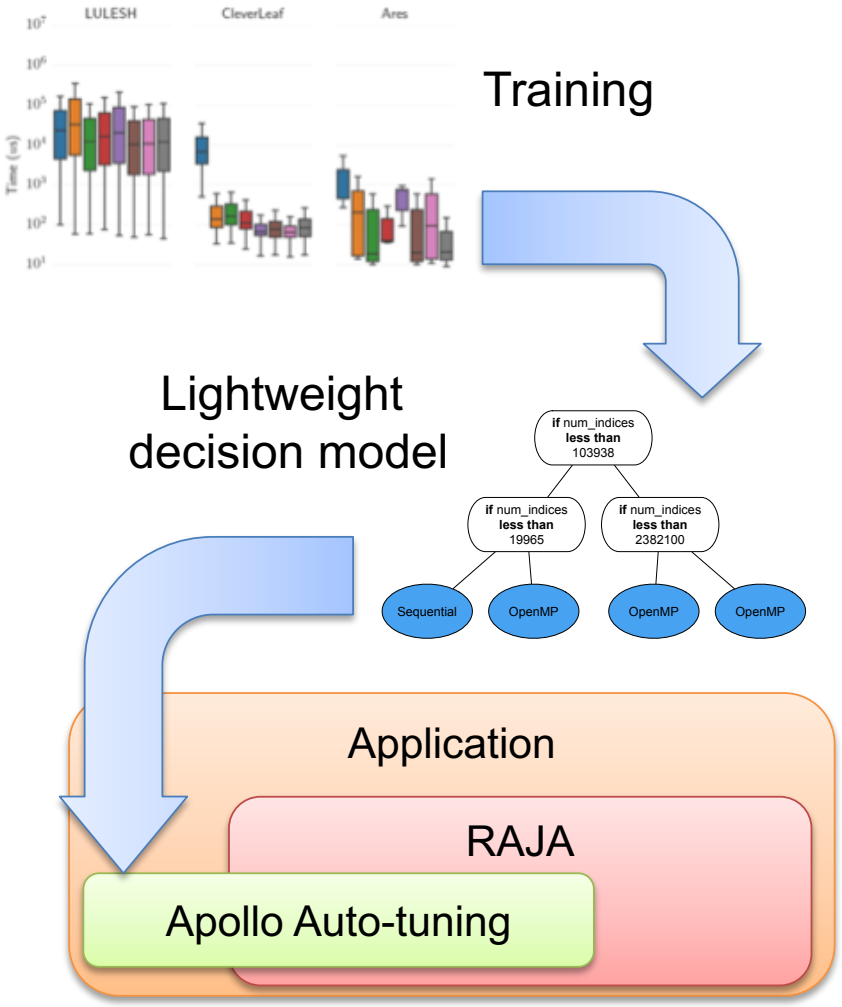
IndexSet: iteration space partition, ordering, dependencies, data placement, etc.



Runtimes of key hydro kernels vary by input and by programming model

- Performance of the same RAJA loop can vary depending on its execution model
  - CUDA, OpenMP, sequential, thread counts, etc.
- Fastest execution model depends on the data
  - Orders of magnitude difference bt/w best and worst
- We would like to select the best execution policy each time a kernel is invoked
  - Requires detailed knowledge of code performance and size of the arrays processed by each kernel
  - Execution policy is set statically to get better compiler optimization, but we want to switch dynamically

Apollo is a RAJA extension for fast auto-tuning of the execution policy

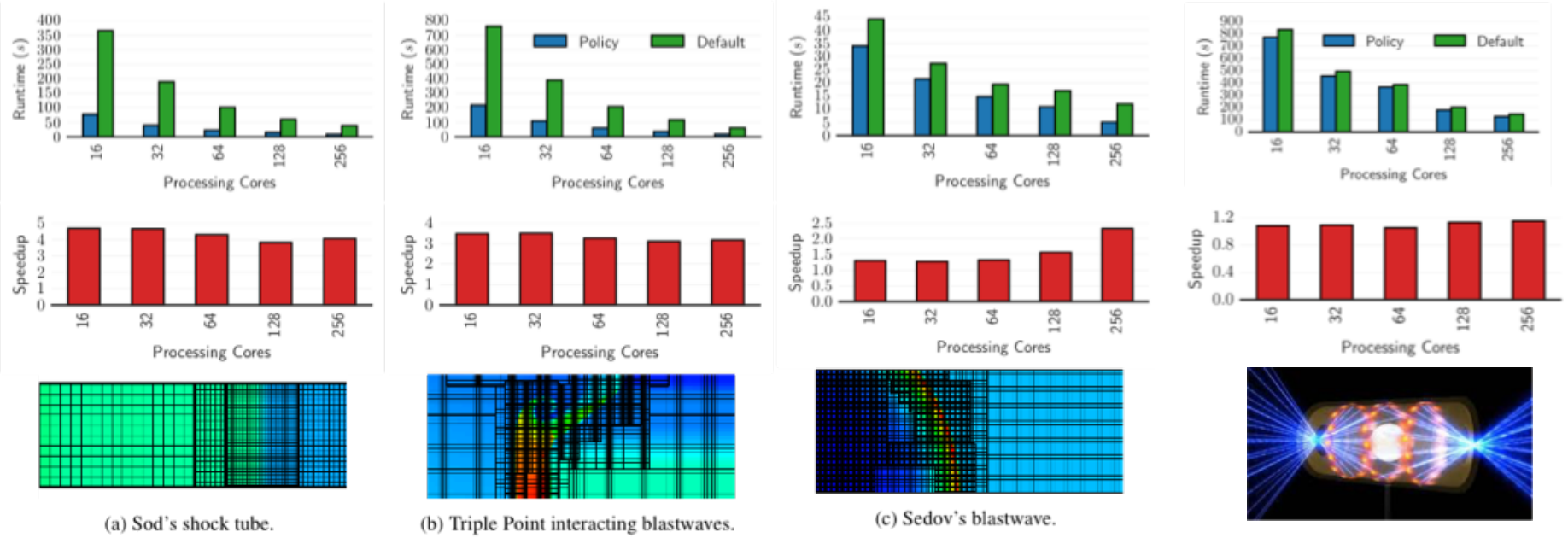


- RAJA provides performance *tunability* but doesn't guarantee *portability*
  - i.e., across different input decks/kernel invocations
- Apollo allows us to train decision models that can pick an execution model online
  - Generates code to pick among different template instantiations (code variants)
  - Templates are chosen on-line, but we retain benefits of static optimization (inlining, etc.)
- Decision models in Apollo are *fast*
  - Can pick an execution policy for every patch in an AMR loop within a single time step
- Apollo decision trees are lightweight
  - Traditional auto-tuners choose parameters on-line based on a costly runtime search
  - Apollo requires offline training, but its model chooses an execution model direction (no runtime test executions required)

Scaling CleverLeaf and ARES with Apollo

CleverLeaf AMR Mini-application

ARES



(a) Sod's shock tube.

(b) Triple Point interacting blastwaves.

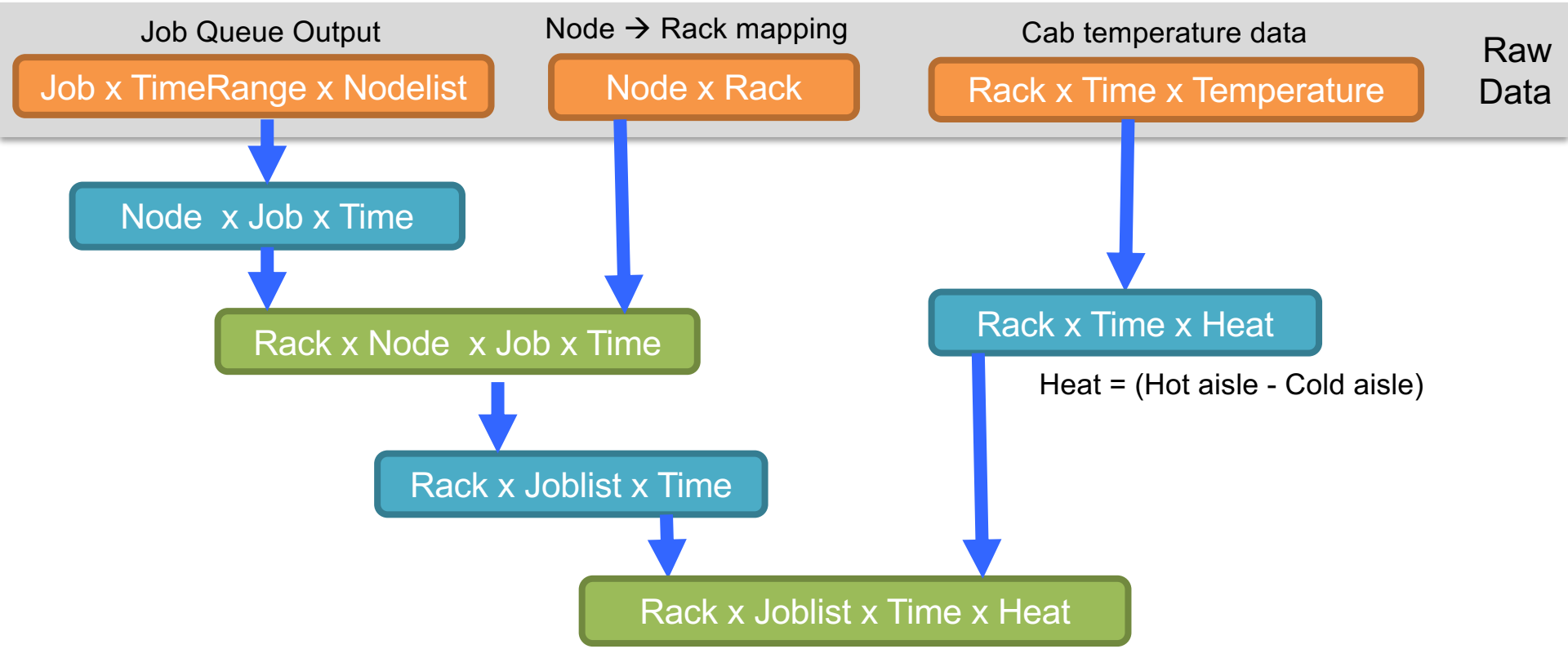
(c) Sedov's blastwave.

- Up to 4.8x speedup for CleverLeaf AMR mini-application
- 8-15% speedup of ARES (with only Lagrange converted to RAJA)
- Speedup *increases* with strong scaling due to increasingly fine granularity
  - For ARES & Cleverleaf Sedov, highest speedup was for the largest run (256 cores)

How can we use Sonar data to understand interaction between applications *and* the facility?

Analyzing performance data requires tedious data scrubbing and transformation

- Which applications (jobs) are correlated with hot racks over time?
  - We collected job queue, machine configuration, and temperature sensor data
  - Stored data exactly as it came from different data sources, but need to project to analyze



This analysis required ~6hrs of running, debugging, and validating data

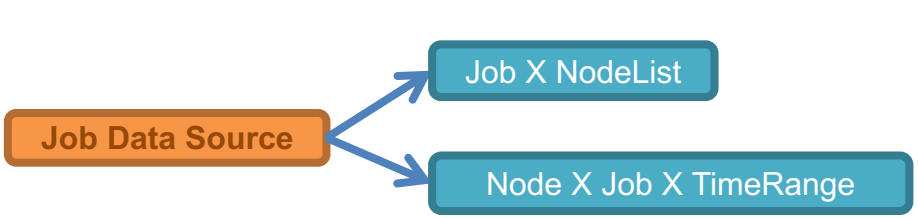
ScrubJay automates the analysis of HPC performance data

- Users do not know all the relationships necessary to piece together facility-wide analysis
- ScrubJay allows users to leverage expert knowledge
  - Experts (admins, developers, HPC architects) tell system the possible transformations in advance
  - Users include these transformations in their own analysis

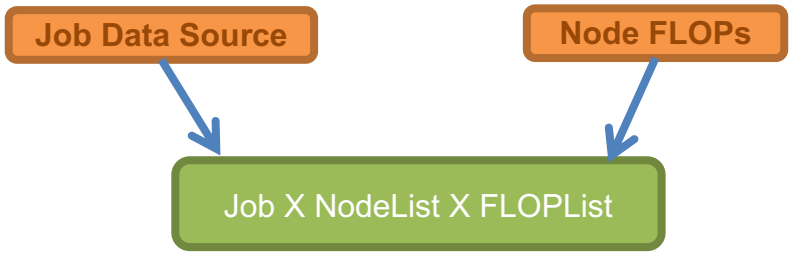
1. **Store *semantics* (types, units, data domain) of data sources**

Job Data Source
Encodes a unique identifier JOBID for every job, the start time of the job, as a human-readable timestamp, as START…

2. **Define *transformations* and *combinations* of sources based on semantics**



Transformations



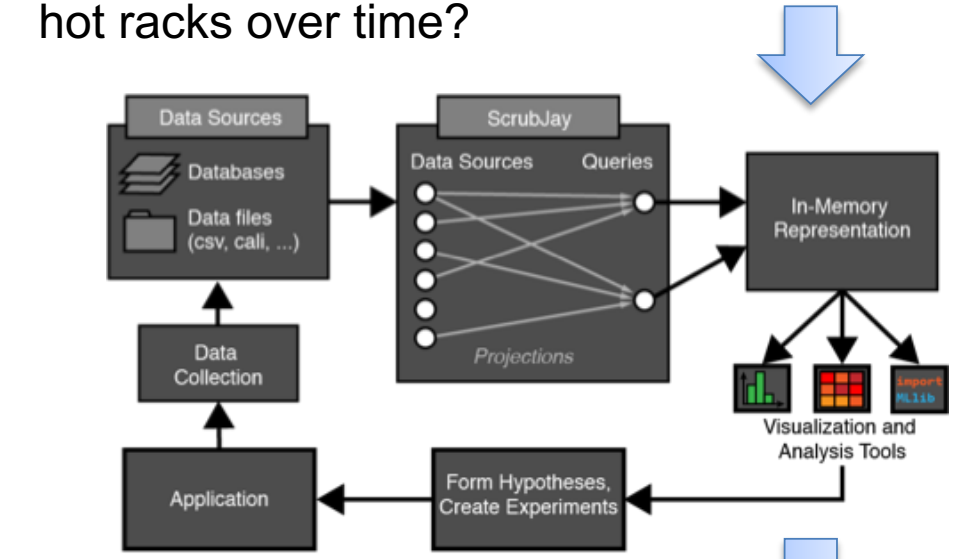
Combinations (Joins, incl. Interpolative Joins)

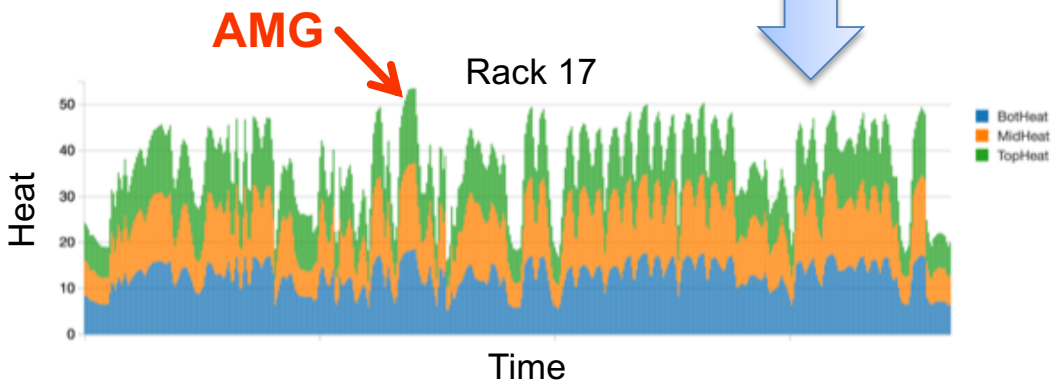
3. **Satisfy queries by automatically transforming/combining data sources**
   - Finding valid paths of joins and transformations is a constraint-solving problem

ScrubJay will allow end users to understand how their jobs interact with the HPC facility

- ScrubJay uses a constraint solve to deduce necessary transformations given an arbitrary query
  - Users do not need to understand raw data formats
  - Users query only a logical data taxonomy
- Projections are parallelized using Spark
  - Users can run Spark jobs through dashboards such as Jupyter Notebook
- ScrubJay allows users to easily run analysis on data from applications and facility data

Which applications (jobs) are correlated with hot racks over time?



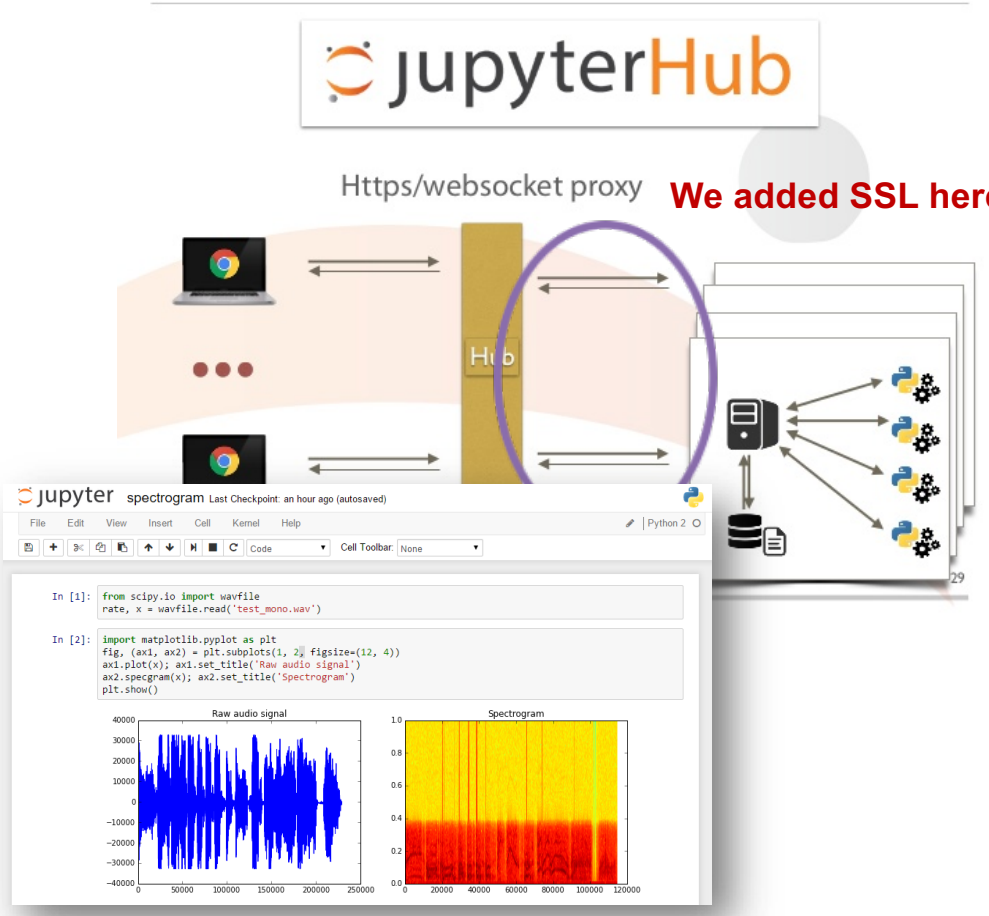


Sonar and ScrubJay will evolve beyond Livermore Computing

- Sonar and ScrubJay are currently services for Livermore Computing users.
- We plan to release ScrubJay as open source
  - Has general applicability to scientific/industry data
  - We plan to develop a strategy to build a community around it
  - ScrubJay could become a commonplace addition to Spark environments
- Caliper is already open source and has a growing community
  - We've striven at LLNL to make it easy for Sonar to ingest Caliper data
- Sharing Sonar stack with other centers requires more thought
  - Implementation and security are tied to LC infrastructure in some ways
  - Platforms like Kubernetes + Containers seem promising
  - Good ways to share distributed services like these with other sites.

We have implemented end-to-end SSL security for JupyterHub, so that it can be used securely at HPC centers

- JupyterHub allows users to write interactive analysis scripts in a Python "notebook" server
  - Out of the box, server is insecure and allows anyone to connect
- We have added security features:
  - Spawn notebooks on LC machines
  - Jupyter proxy certificate authority (CA) manages unique certificates for each notebooks
- Our additions ensure that user identity is preserved from the client, through the proxy, to the notebooks.
  - We are contributing this upstream
- LLNL Next-generation code team (MARBL) will be using Jupyter Notebooks as the steering interface for their code.



We rolled this out to our first network zone on Tuesday

Our infrastructure and analysis techniques will enable center-wide HPC performance analysis

- Performance analysis is becoming more data intensive!
  - Parallel performance analysis IS big data analysis.
  - We need better tools to analyze and mine data about our facilities.

- LLNL is expanding the scope of data it is collecting in order to tune the entire HPC center
  1. Understand the real production workload
  2. Focus more on job throughput and quantifiable improvement
  3. Look at training tuning models based on *historical* performance data
  4. Allow users to explore data for the entire center (no more anecdotes)


Lawrence Livermore
National Laboratory