

Resource Partitioning and Power Management in the Exascale Era

ANL: Valentin Reis, Florence Monna, Swann Perarnau, Kamil Iskra, Kazutomo Yoshii
 LLNL: Tapasya Patki, Stephanie Brink, Aniruddha Marathe, Barry Rountree

Improving all layers of the open-source resource management ecosystem

The goal of the Argo resource management effort is to provide user-facing advanced mechanisms to control and monitor resource usage across the system. This includes performance isolation, support for advanced workloads such as workflows and coupled-codes, and comprehensive power management.

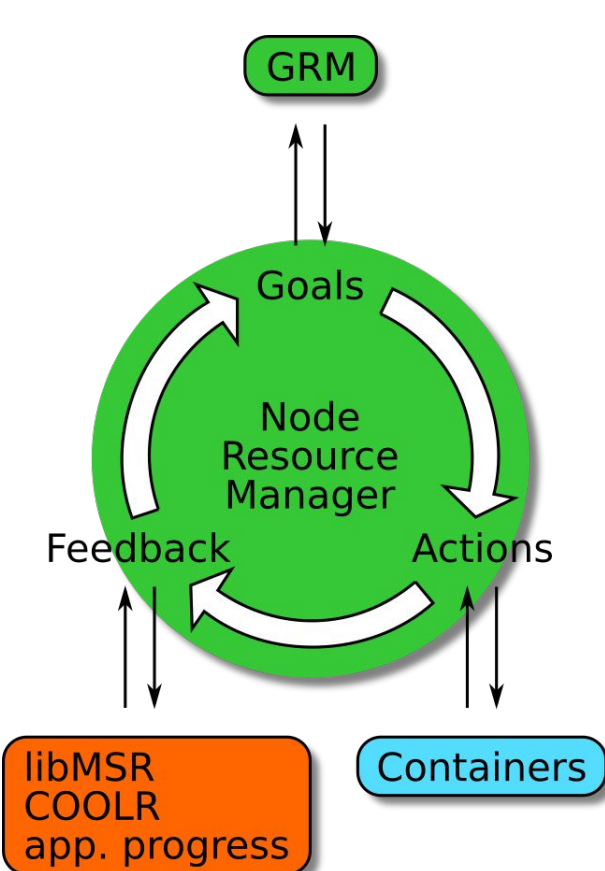
Local (Node) Resource Management

Overview

- Hierarchical resource partitioning
- Containers for intra-node resource partitioning
 - Using the cgroups mechanism of Linux
- More efficient “packing” of multi-component applications
- Arbitrate resources between applications and runtime services
- Reconfigurable, dynamically tracking resource changes
- Integration with batch schedulers, power management

Node Resource Manager

- Single API endpoint for all node resource management services
- Map containers to topology, interact with container runtime
- Upstream API: interact with GRM, publish node events
- Downstream API: user access to container management, application reporting, power command
- Goals: integration across hierarchy levels, collaboration with job schedulers, MPI



Application-Aware Sensors

- Monitor application progress through hardware performance counters, self-reporting
- Monitor hardware sensors: power, temperature, fan speed, frequency
- GRM-provided power limit
- Closed-loop control mechanism, p-state/RAPL actuators
- ECP goals: improve control-loop, integrate more sensor data into policies

Resource Abstraction and Control Synthesis

NRM performs abstracted resource accounting in the form of “sensors” and “actuators”, enables their discoverability, and optimizes reconfigurable sensor optimization goals.

- Reconfigurable for any number of sensors and actuators
- Specified through a Control Problem Description format
- Synthesizes a Multi-Armed Bandit controller

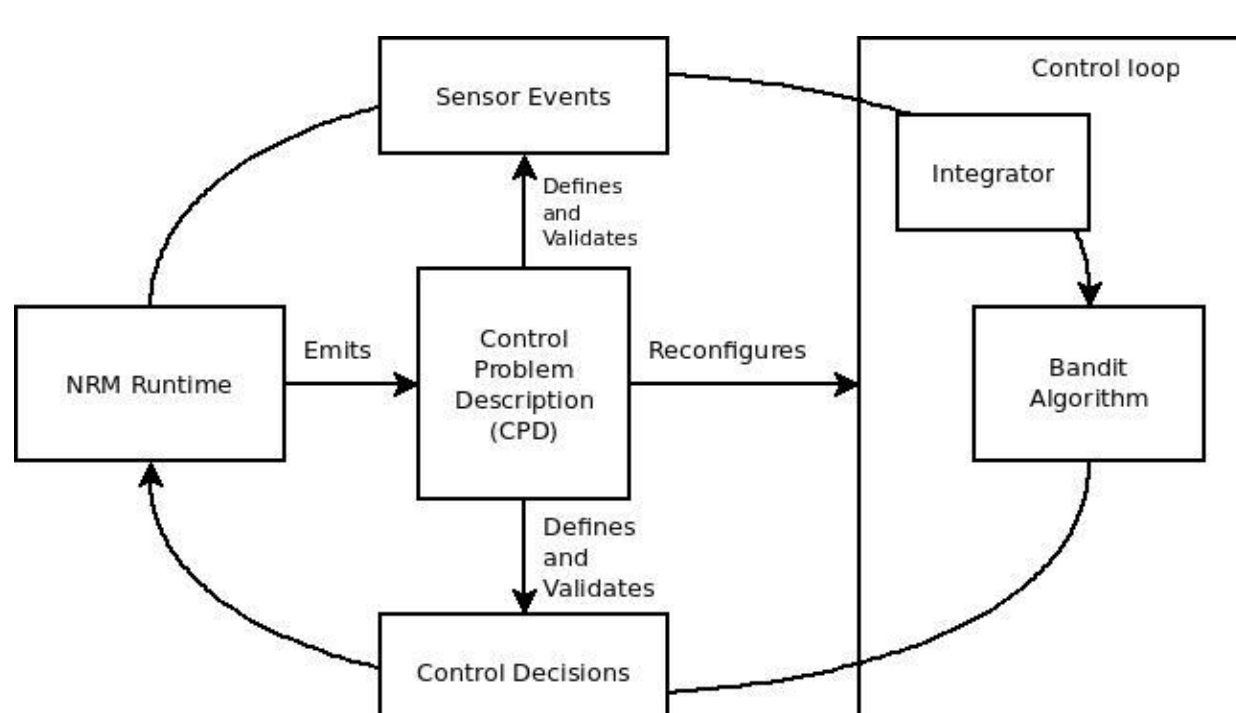
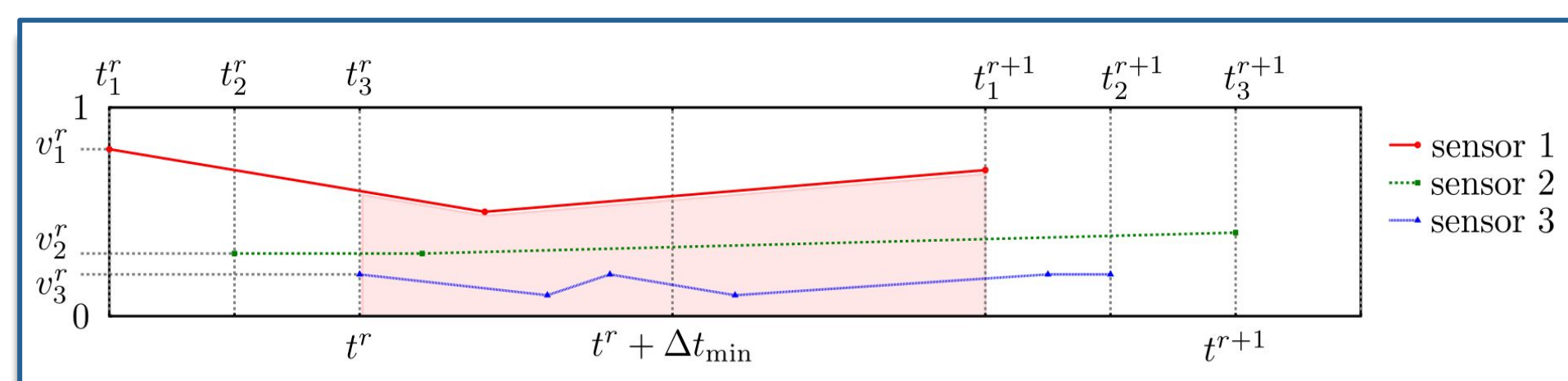
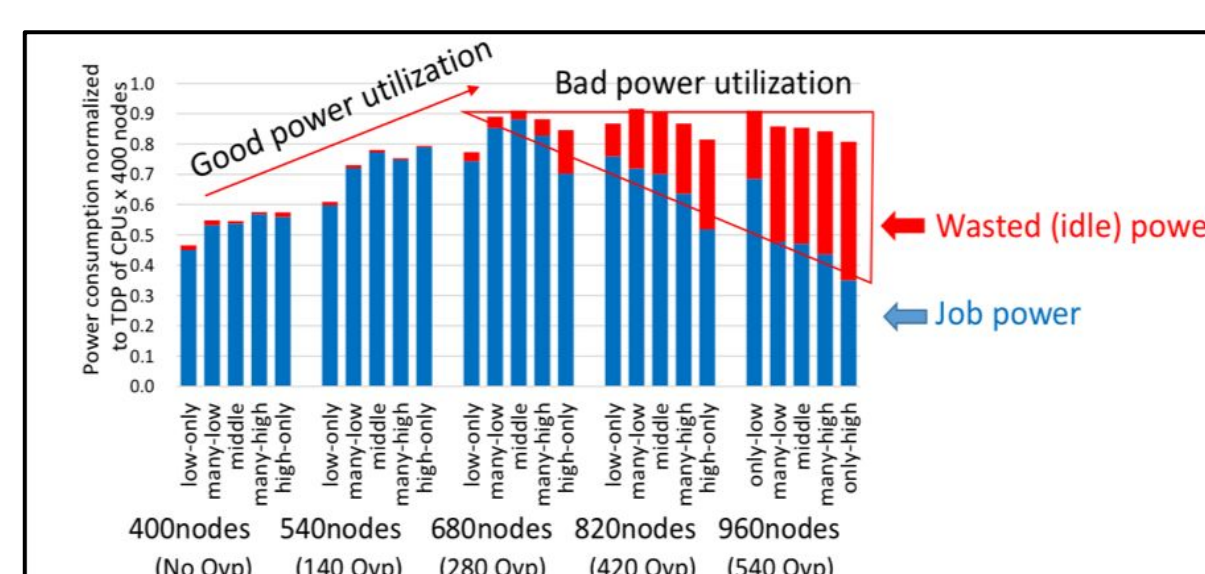
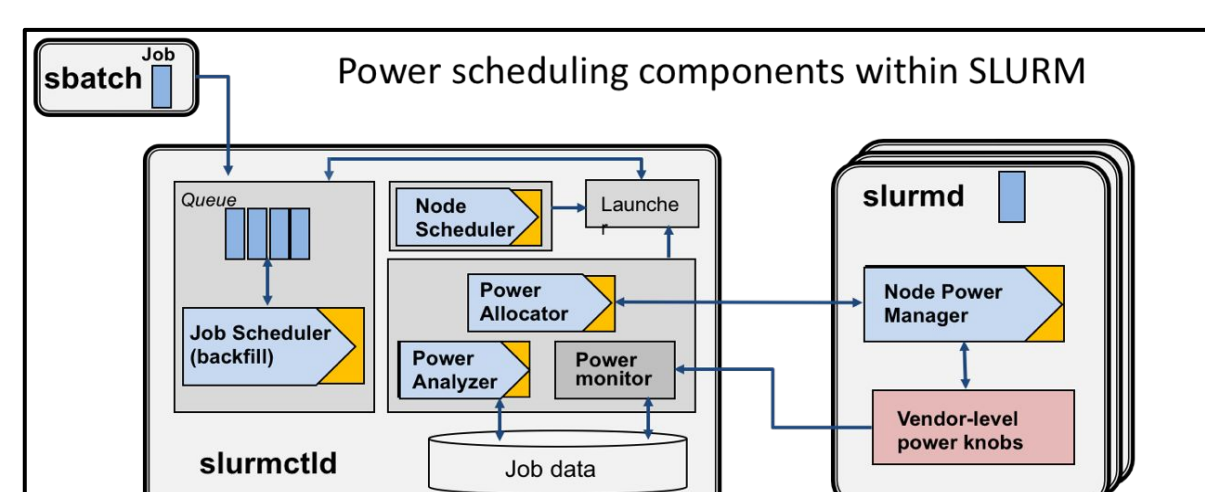
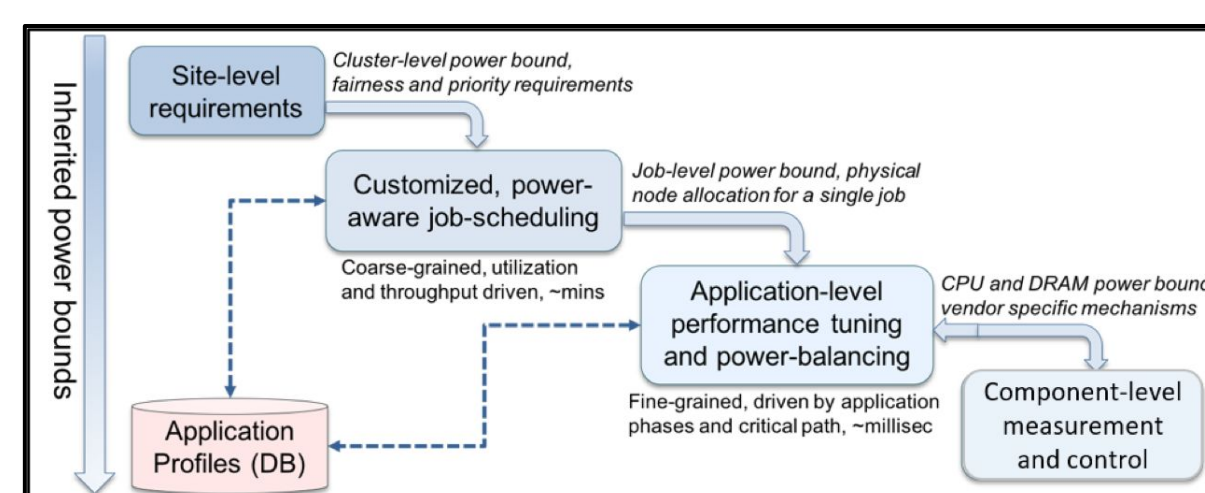


Figure: NRM's control synthesis and sensor integration mechanism.

Global Power Management

Scheduler-Aware Hierarchical Control

- Integrate job scheduler, enclaves to control power across jobs
- Use NRM data to monitor power and application performance
- Steer power where it can most advance the application's progress
- ECP goals: production-ready GRM, integration across different hierarchy levels
- **PowerStack**: Develop first prototype and lead community effort toward capturing power management details from the microarchitecture-level up to the site-level.
- **PowerStack** will leverage and extend existing tools such as msr-safe, libmsr, variorium, Intel GEOPM (production codes) and Adagio, Conductor, RMAP, PowSched, P-SLURM (research codes)



As part of PowerStack, a Power-aware design of SLURM (GRM) has been developed and tested on 960-nodes on the HA8K supercomputer in Japan (collaborators). Results show the benefits of hardware overprovisioning at scale as well as sensitivity to the degree of overprovisioning.

Variorium v1.0 Released:

- <https://variorium.readthedocs.io>
- Extensible, open-source library for exposing low-level hardware knobs
- Vendor-neutral API for power management, with current support across six Intel architectures, IBM Power9, and NVIDIA Volta.

SLURM SPANK Plugin with GEOPM Released:

- <https://github.com/geopm/geopm-slurm>
- Allows for end-to-end control across jobs with and without GEOPM on Intel architectures
- Integration into TOSS deployment framework is underway
- Variorium + GEOPM integration is planned for FY20, allowing vendor-neutral implementation of PowerStack with SLURM.