Resource Partitioning and Power Management in the Exascale Era

ANL: Sridutt Bhalachandra, Kamil Iskra, Swann Perarnau, Valentin Reis, Kazutomo Yoshii
LLNL: Tapasya Patki

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.

Resource Partitioning

Overview

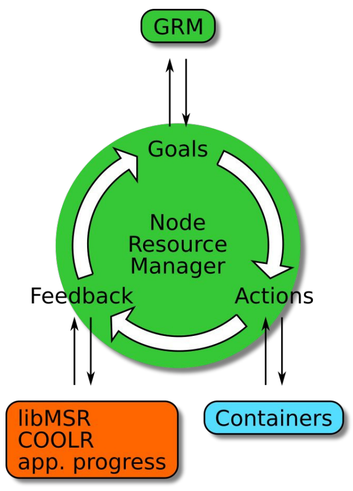
- Hierarchical resource partitioning
- Enclaves for job partitioning at node granularity
  - Enable subjobs, services above node level
- Containers for intra-node resource partitioning
  - Using the cgroups mechanism of Linux
- More efficient “packing” of multi-component applications
- Arbitrate resources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

Enclaves

- Groups of nodes managed as a single entity
- Application-facing abstraction to create, configure and monitor subpartitions of a job
- Extendable through distributed services: fault-tolerance, performance monitoring
- Rely on existing hierarchical job scheduling facilities
- ECP goals: integration with job schedulers, development of new services

Node Resource Manager

- Single API endpoint for all node management services
- Map containers to topology, interact with container runtime
- Monitoring through libMSR API, safe access to MSRs through msr-safe
- Upstream API: interact with GRM, publish node events
- Downstream API: user access to container management, application reporting
- ECP goals: integration across hierarchy levels, collaboration with job schedulers, MPI

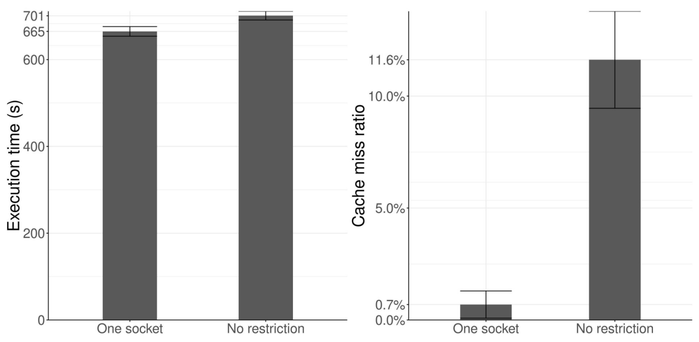


Compute Containers

- Users provide resource requirements inside a container manifest
- Fully compatible with 3d party solutions (Singularity, Docker, Shifter)
- ECP goals: finer-grained resource control, better integration with the ECP ecosystem (job scheduler, MPI)

Partitioning results

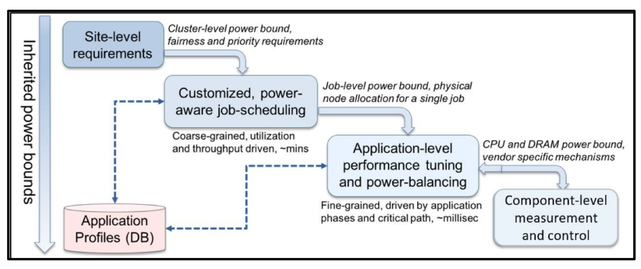
- Dual Xeon Gold 6126
- Two MPI jobs co-scheduled using libEnsemble
- Execution time improved thanks to reduced cache miss ratio

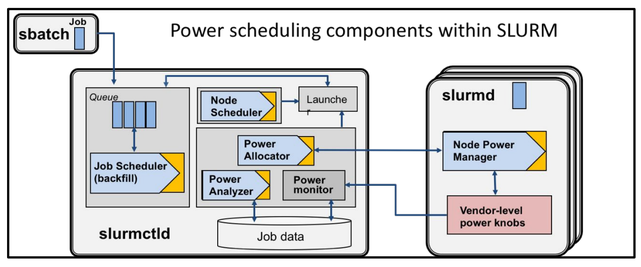


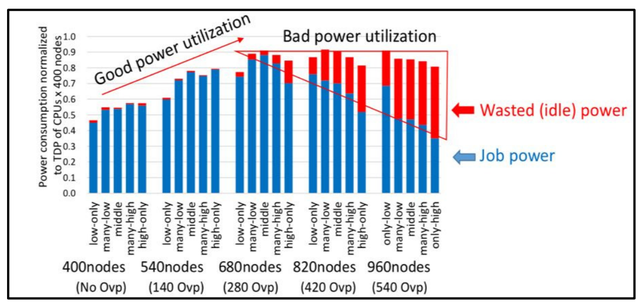
Power Management

Scheduler-Aware Hierarchical Power 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, Intel GEOPM (production codes) and Adagio, Conductor, RMAP, PowSched, P-SLURM (research codes)



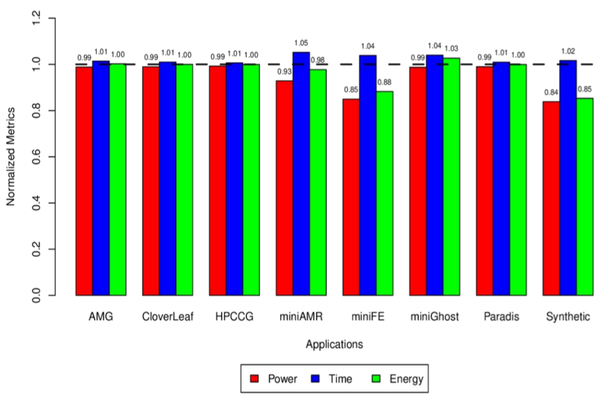




As part of PowerStack, a Power-aware design of SLURM (GRM) has been developed and tested on 960-nodes on the HA8K supercomputer in Japan (collaborators). Results show the benefits of hardware overprovisioning at scale as well as sensitivity to the degree of overprovisioning. Development of interfaces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

Application-Aware Node Power Control

- 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



A bulk synchronous parallel (BSP) application-aware power policy with de-coupled monitoring and control identifies critical path across application phases to dynamically adapt power, showing up to 15% energy reduction

This research was supported by the Exascale Computing Project (ECP), Project Number: 17-SC-20-SC, a collaborative effort of two DOE organizations - the Office of Science and the National Nuclear Security Administration, responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering and early testbed platforms, to support the nation’s exascale computing imperative.
Part of this work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-POST-