http://scalability.llnl.gov/

Power Constrained HPC

Martin Schulz
Center or Applied Scientific Computing (CASC)
Lawrence Livermore National Laboratory

With many collaborators and Co-PIs, incl.:
LLNL: Barry Rountree, Tapasya Patki, Aniruddha Marathe
U Arizona: David Lowenthal, Sam Cotter
Intel: Jonathan Eastep and the GEOPM team, Matthias Maiterth (also LRZ)
ANL: Pete Beckman, Kamil Iskra and the ARGO team
U Oregon: Al Malony and Dan Ellsworth

LLNL-PRES-733699
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

Constraints are a Fact of Life (and also in HPC)

Modern systems are built around constraints

- Number of nodes and cores
- Amount of memory per core
- Memory and network bandwidth
- …

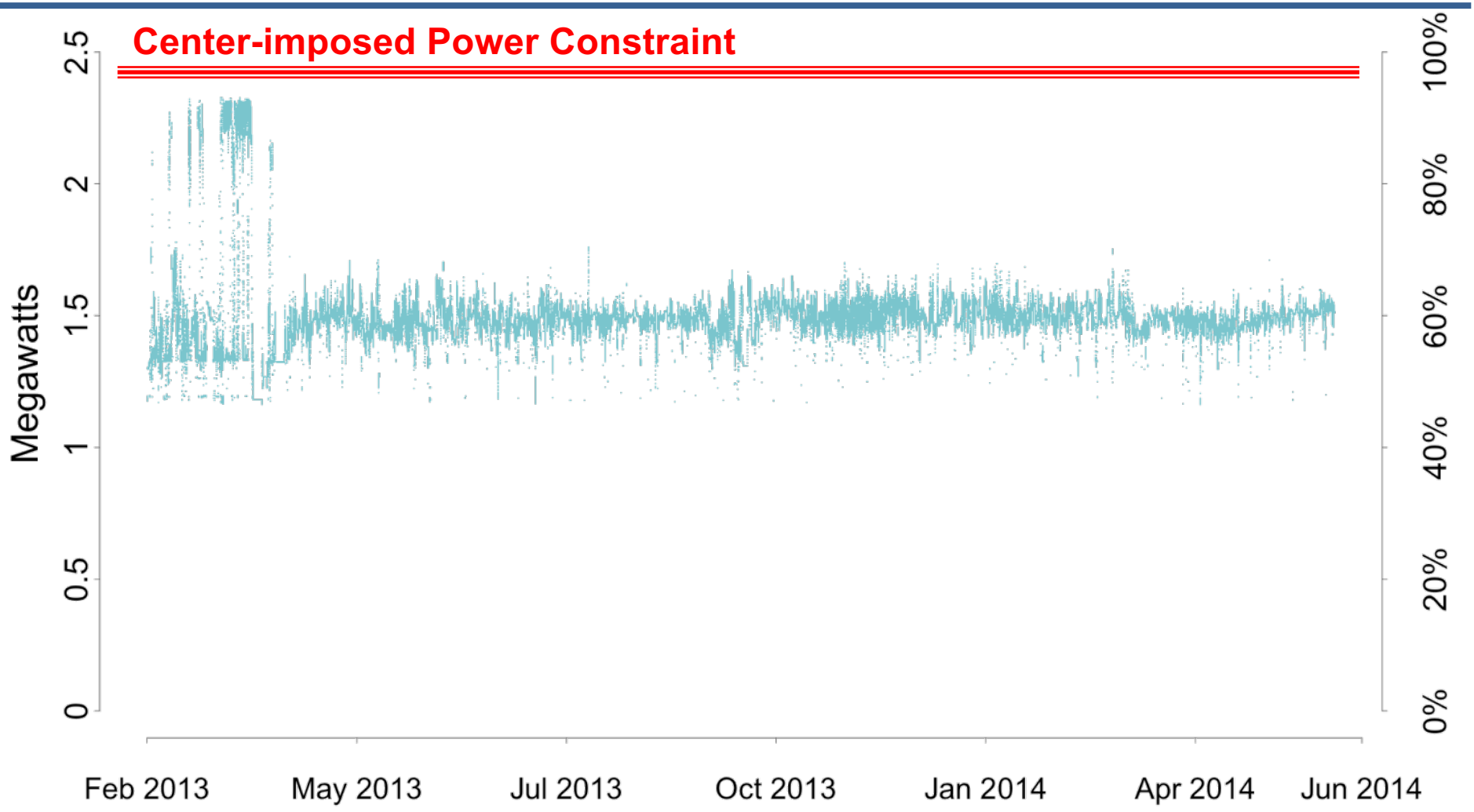
Power and Energy add two new constraints for centers

- Hard limits caused by physical and/or budget constraints
- Typically not an optimization target
- Some centers are already starting to hit such limits today

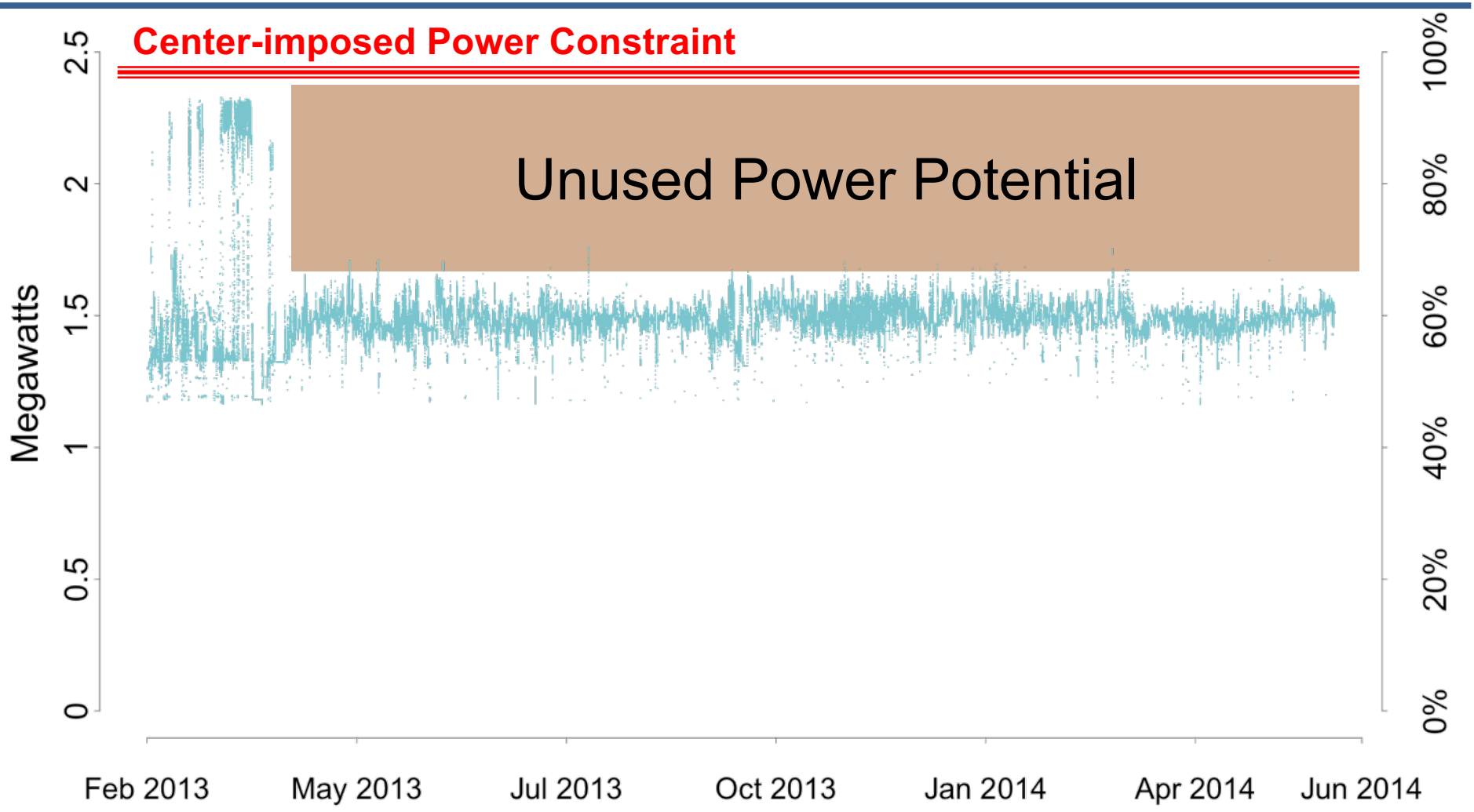
But: power and energy have special properties

- Getting more memory is impossible
- Changing the number of nodes of a job on the fly is hard
- But: Power and energy can easily manipulated

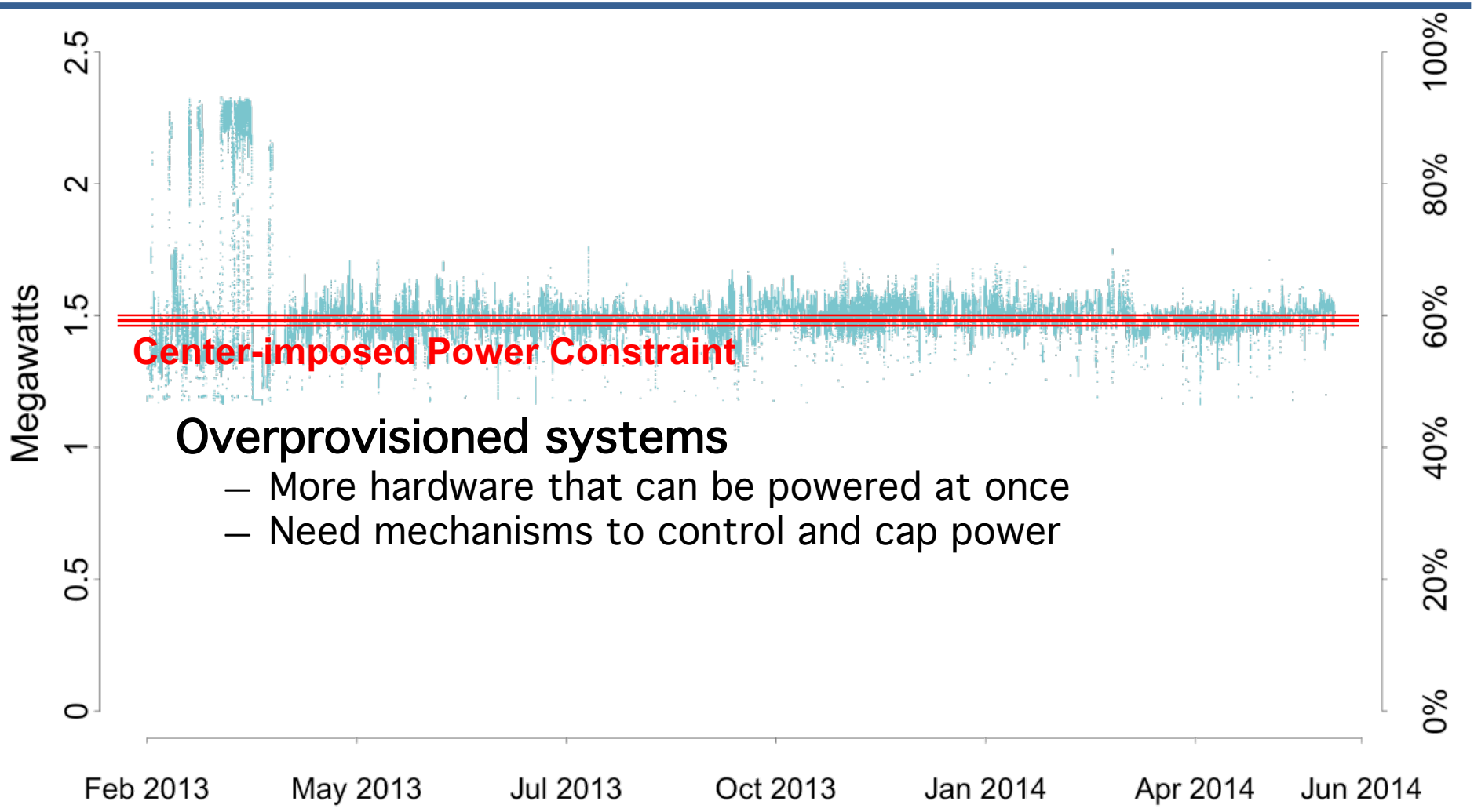
Power Consumption on Vulcan, a BG/Q System



Power Consumption on Vulcan, a BG/Q System



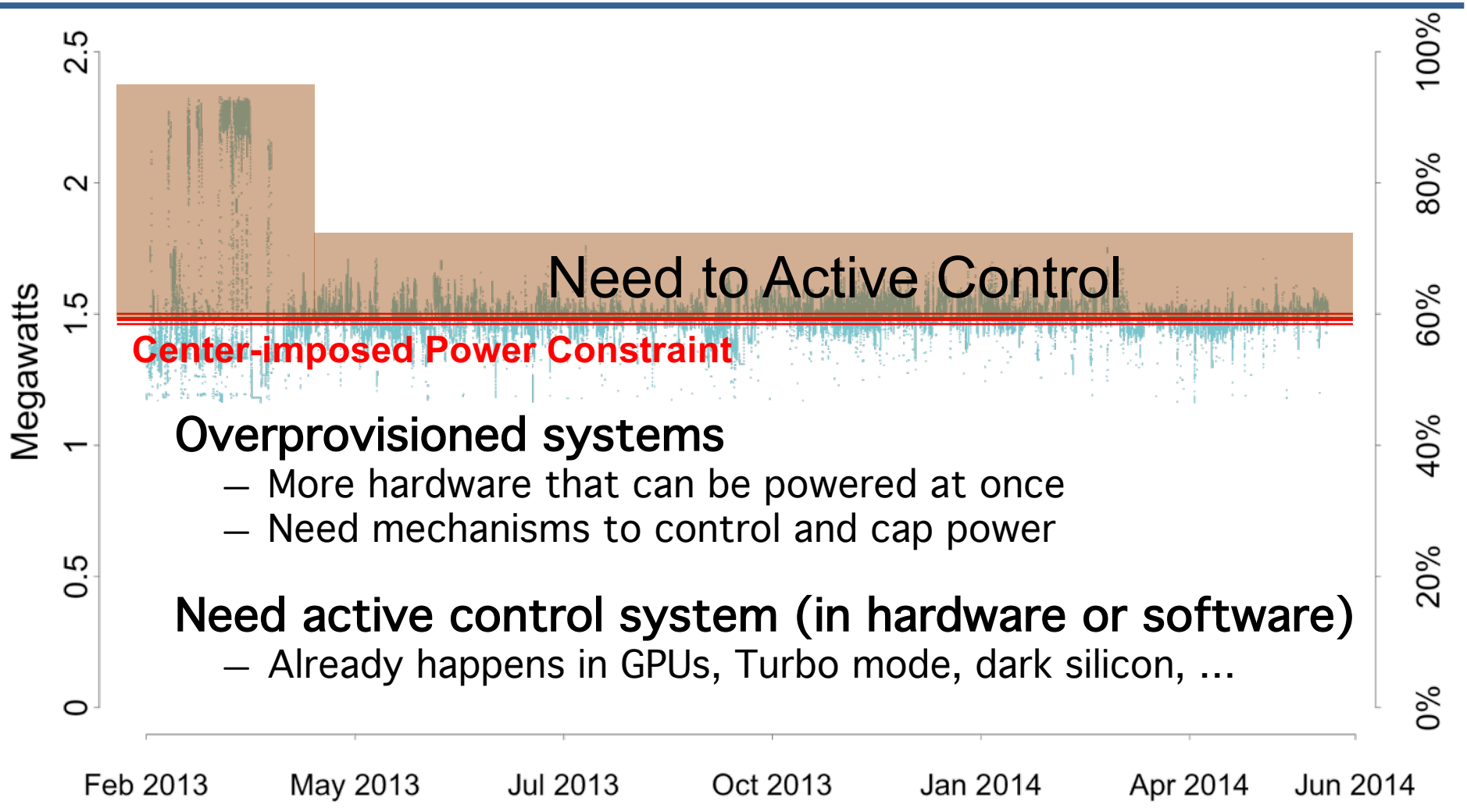
What if …



Overprovisioned systems

- More hardware that can be powered at once
- Need mechanisms to control and cap power

What if …



Overprovisioned systems

- More hardware that can be powered at once
- Need mechanisms to control and cap power

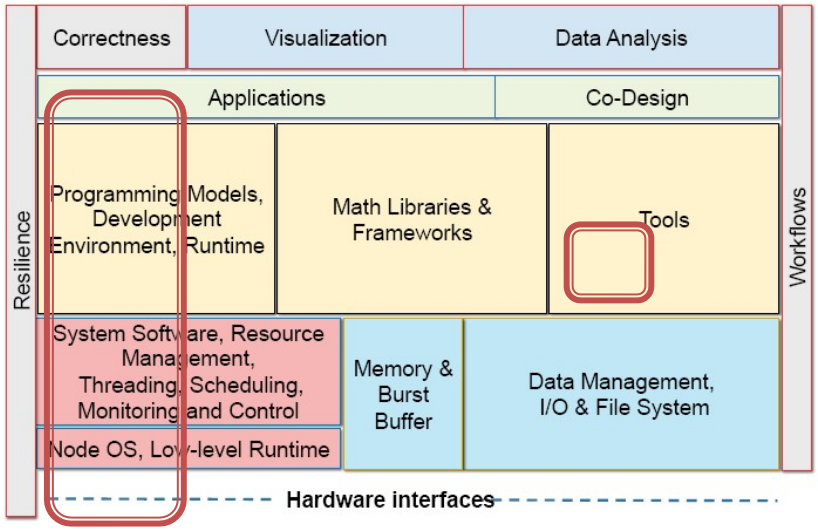
Need active control system (in hardware or software)

- Already happens in GPUs, Turbo mode, dark silicon, …

New Software Stack to Manage Power/Energy

Vertically integrated software stack

- Part of global system software
- Integration with node software
- Scalable communication
- Interaction with applications
- Low overhead



Support for Multiple Constraints

- Initial target: power constraints
- Ultimately any constraint
- Work on energy ongoing as well
- Configurable for each site

Developed as part of the Exscale Computing Project (ECP)

- Multiple projects (ARGO, PowerStack)
- Integration with Caliper to enable application feedback
- Close collaboration with the hardware column

Four Questions

Managing the power hierarchy

- How to Manage on Node Power/Energy?
- How to Manage on Job Power/Energy?
- How to Manage on Global Power/Energy?

How to coordinate to schedule resources?

- Interfaces between layers
- Integration into the machine resource manager

(1) How to Manage on Node Power/Energy?

Arbiter between system constraints and runtime requests

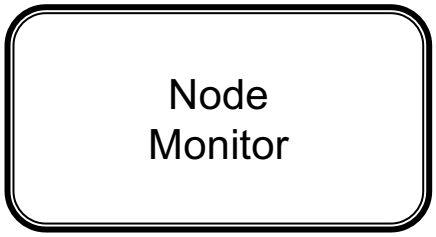
- Central point for power and energy measurements
- Power limits vs. runtime demands
- Energy limits vs. runtime demands
- Thermal control

Locally supported by libmsr

- https://github.com/LLNL/libmsr
- Access to RAPL on Intel systems

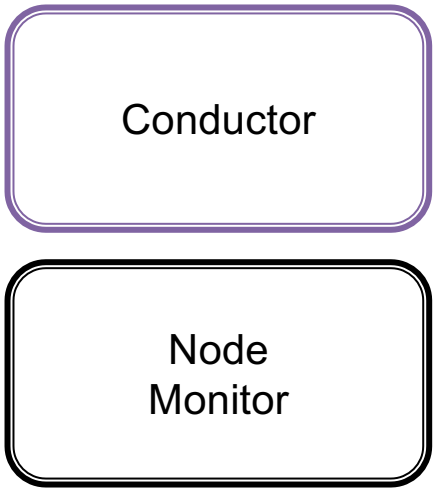
Open challenges

- Enforcement across runtimes
- APIs for runtime
- Coordination backchannel



(2) How to Manage on Job Power/Energy?

Given a job-level power constraint,
how do we optimize application performance?



Identify critical path

- Only the critical path needs full power
- The rest can work with reduced power

Measure power headroom

- Execute application for controlled period of time and measure power
- Can be distributed based on process criticality

Execute repeatedly during application execution

- Typically on time step boundaries
- Intended for repetitive applications

Step I: Configuration Selection

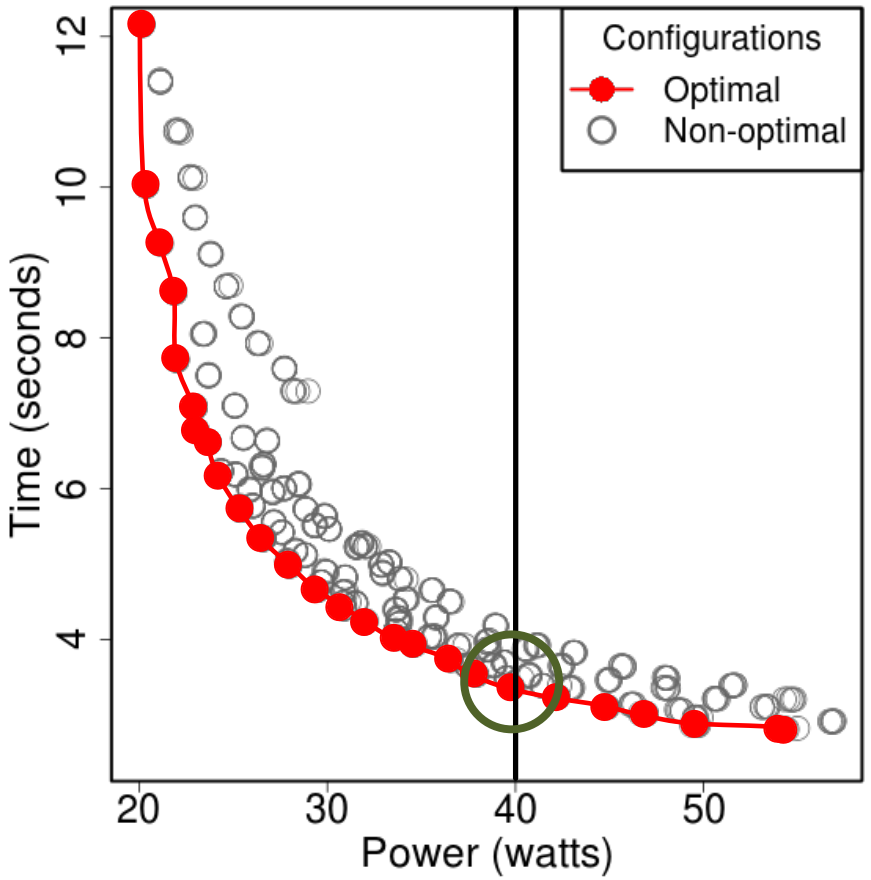
Profile the configuration space on-line

1) Run each computation operation on individual nodes at distinct configurations (e.g., # threads)

2) Record the power/perf. profile characteristics of each computation operation

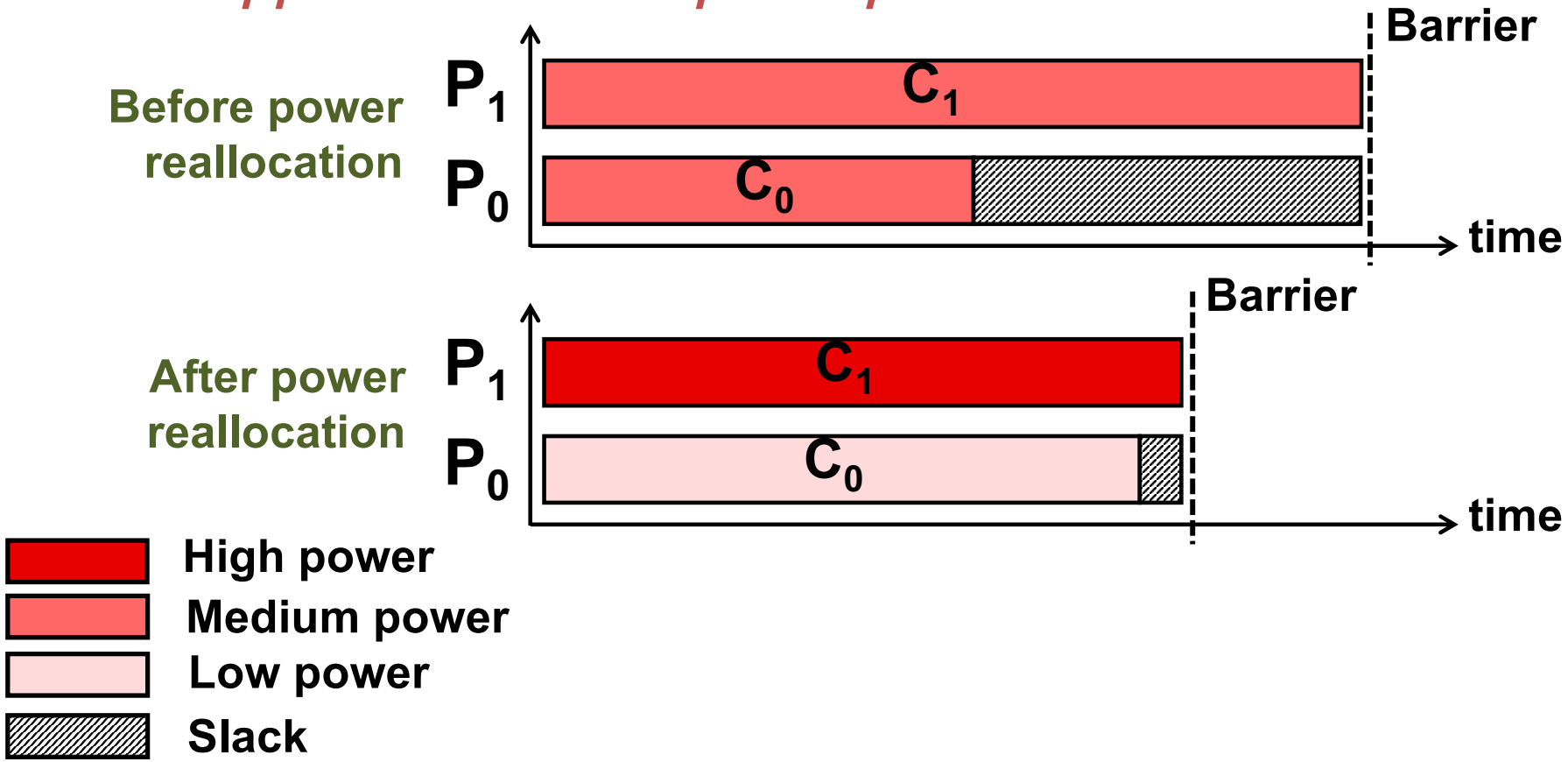
3) Construct Pareto frontier

4) Pick best configuration under a given power bound



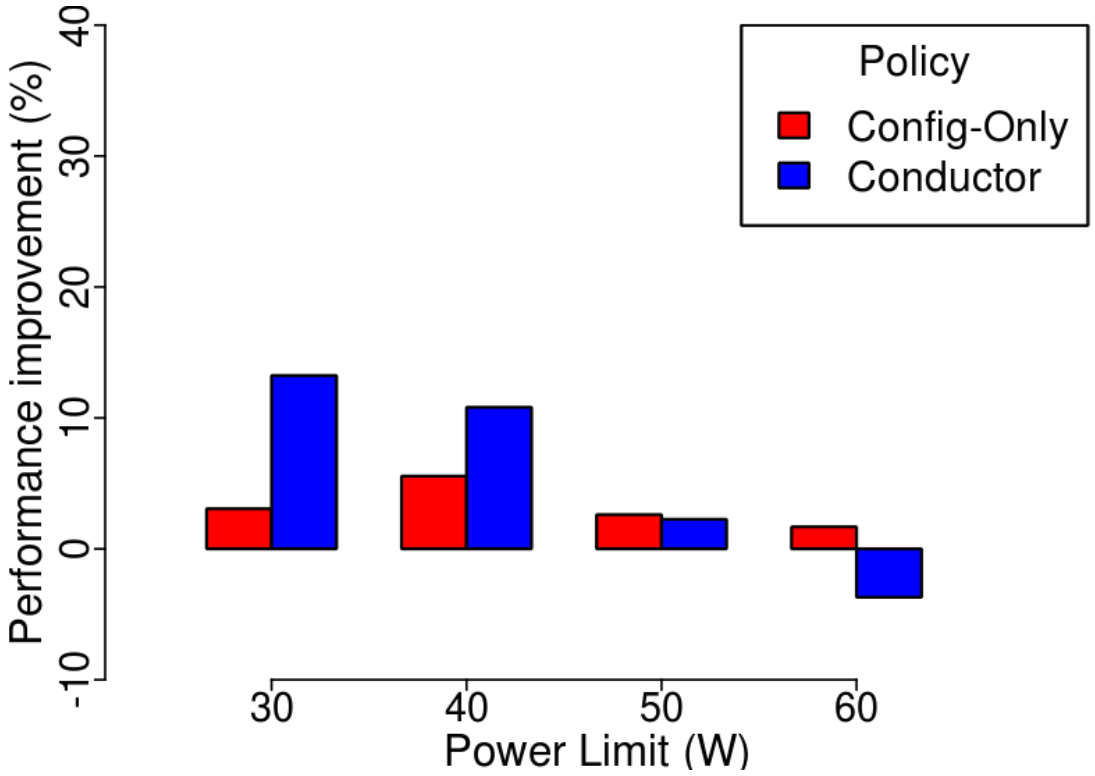
Step II: Power Reallocation

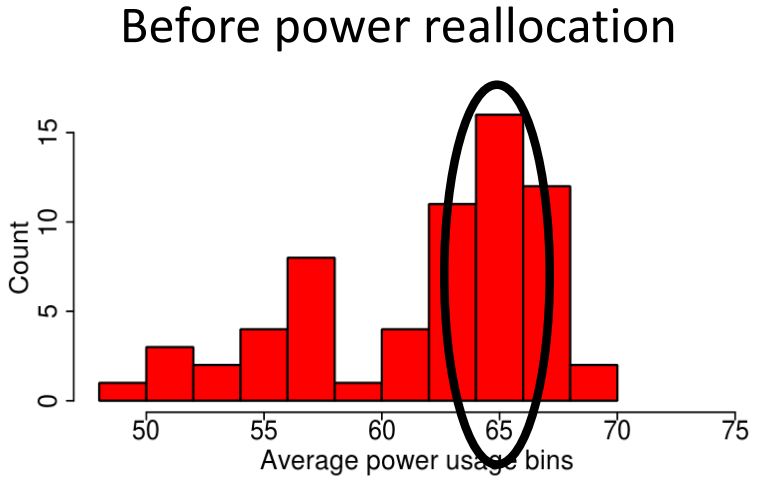
How can we allocate power to the critical operations in an application and improve performance?

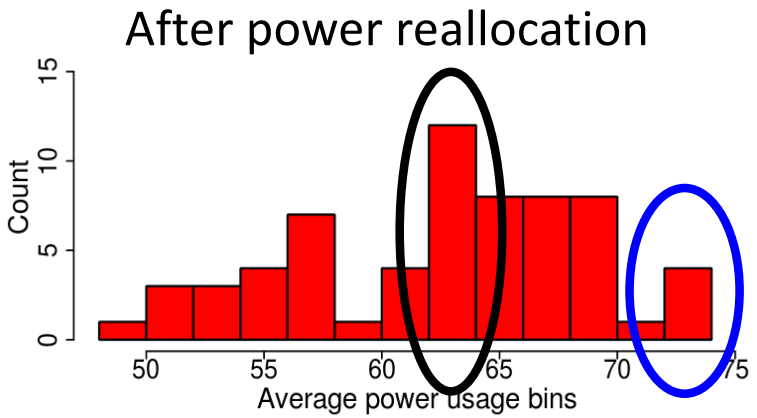


Conductor Benefits Dynamic Apps.

Example: Crystal code on 512 SandyBridge CPUs





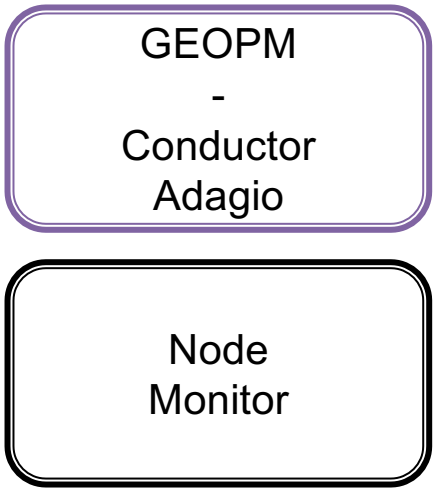


- Up to 13% speedup over Static scheme
- Benefits from process-level imbalance of power usage

ECP PowerStack Project

Goal: Creating a production ready job-level power runtime

- Conductor techniques for power-aware computing
- Option to enable energy-aware computing (Adagio)
- Scalable and extensible base infrastructure
- Portable across platforms



Global Extensible Open Power Manager

Free open source power management framework

- Job-level power management runtime
- Focus on scalability

Scalable, self-configuring tree-based communication infrastructure

Plug-in architecture for extensibility

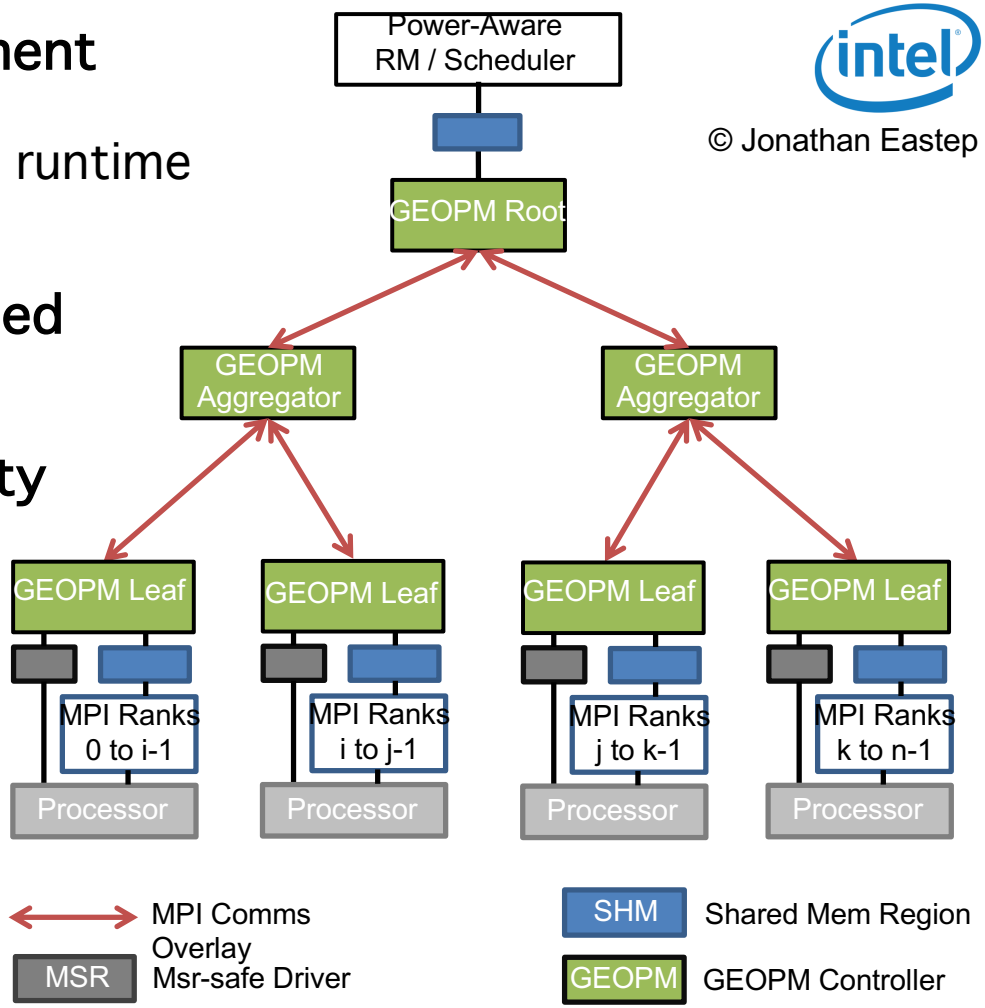
- Control algorithms
- Platform plug-ins

Alpha version available

- v1.0 product coming soon
- github.com/geopm/geopm

Intial target ANL's Theta

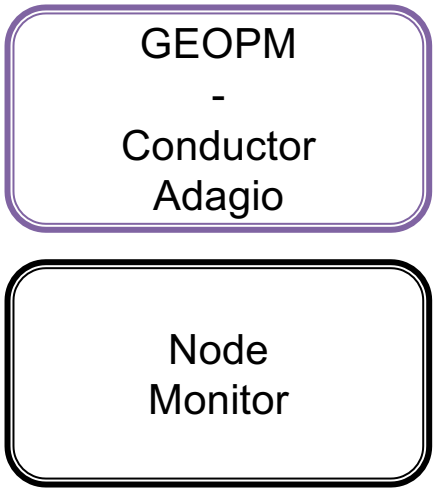
- Other systems soon



ECP PowerStack Project

Goal: Creating a production ready job-level power runtime

- Conductor techniques for power-aware computing
- Option to enable energy-aware computing (Adagio)
- Scalable and extensible base infrastructure
- Portable across platforms



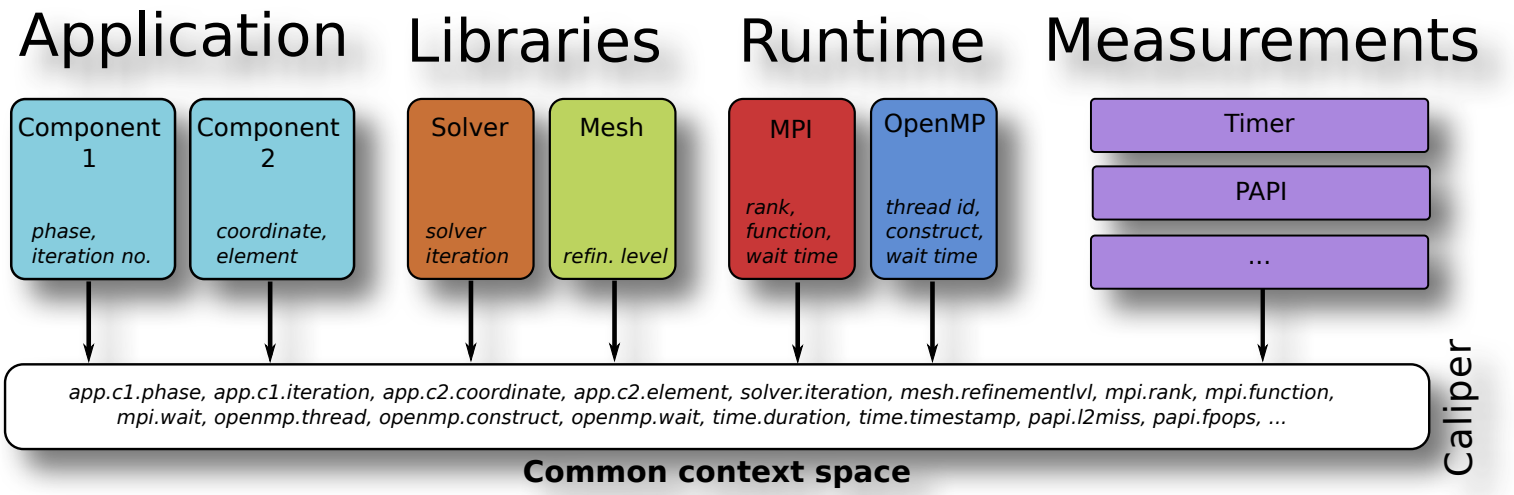
GEOPM as integration base

- Inclusion of Conductor's mechanisms
- Platform extensions
- Deployment on ECP testbeds

Caliper for application annotations

- Low overhead annotation API
- Portable across apps and tools
- Demarcation of phases and regions
- Mapping to GEOPM annotations

Application Introspection with Caliper
https://github.com/LLNL/Caliper



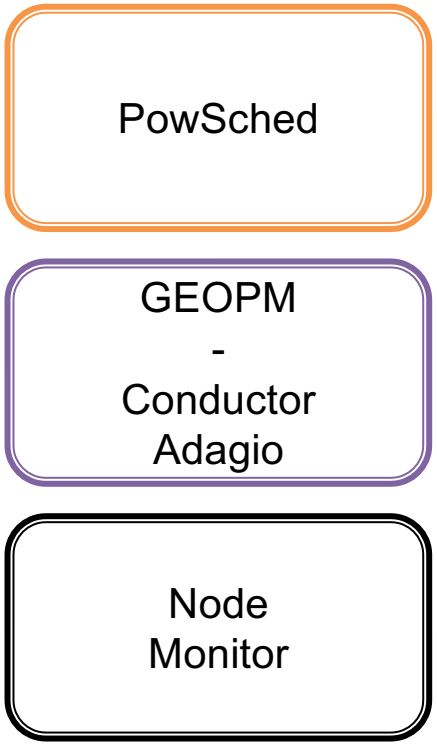
Shared application/workflow wide context

- Simple annotation API for applications and libraries
- Re-usable annotations across tools, codes, runtimes, …
- Automatic context management across SW components

Upper layers can query context and annotate their own data

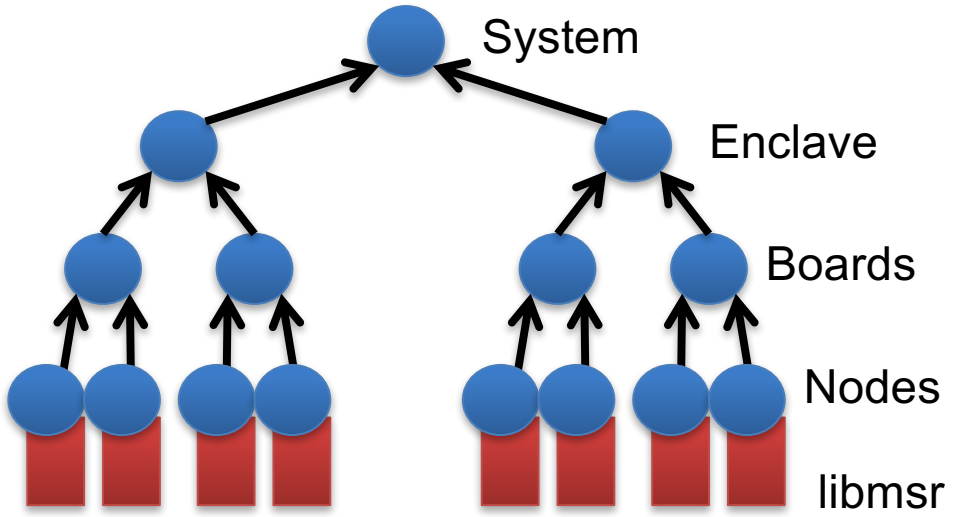
- Existing tools can use Caliper as a module
- New tools can be integrated into Caliper as “services”

(3) How to Manage on Global Power/Energy?



Transparent Runtime Adaptation

- Part of a global operating system
- Reallocation of unused power
- Transparent to application
- Scalable communication scheme



PowSched (Part of ECP ARGO Project)

Predetermine per job power budgets

(1) Collect the readings for all components

- Scalable aggregation

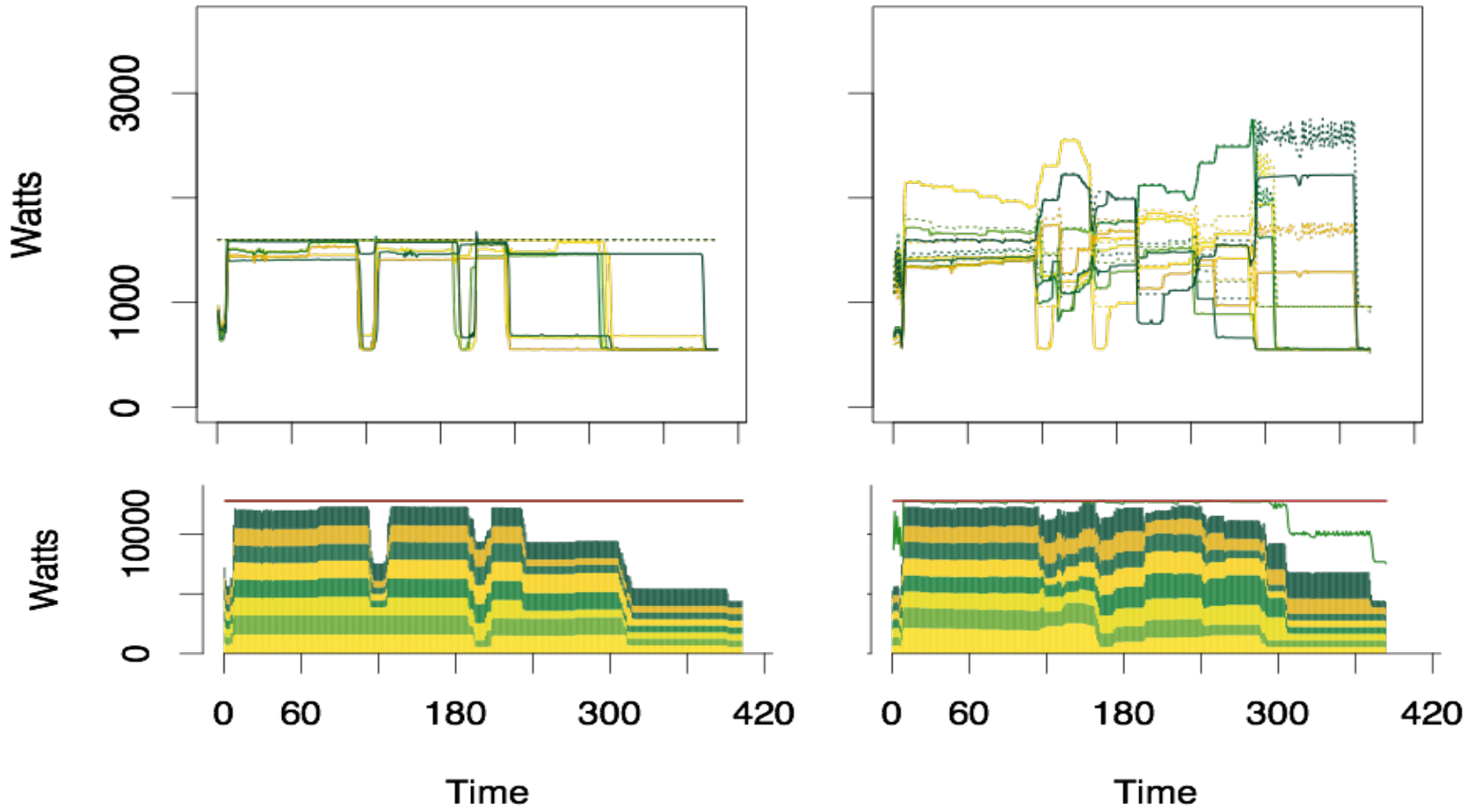
(2) Allocate less power to components that are consuming below their budget

- Find appropriate jobs
- Calculate new power per job
- Distribute to jobs

(3) Allocate more power to components that are consuming near their current allocation

- Calculate power headroom
- Split headroom among eligible jobs
- Distribute to jobs

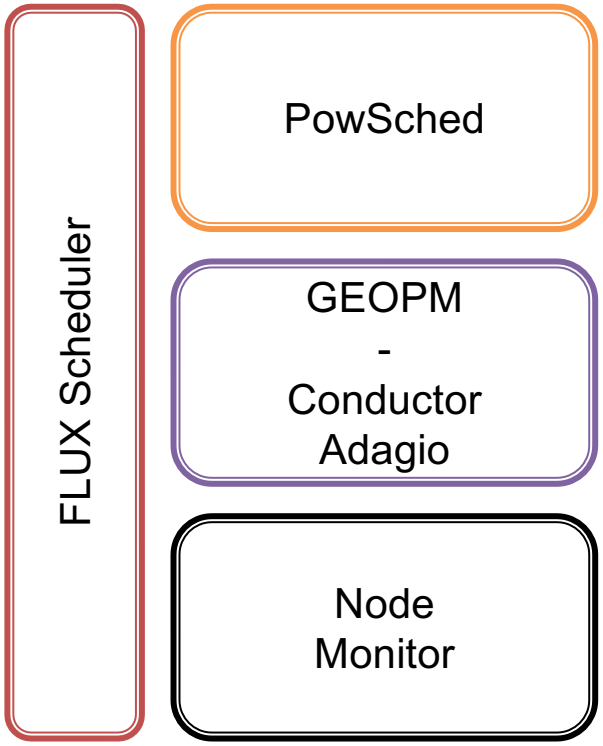
Power-aware Resource Management



Power-aware Resource Management

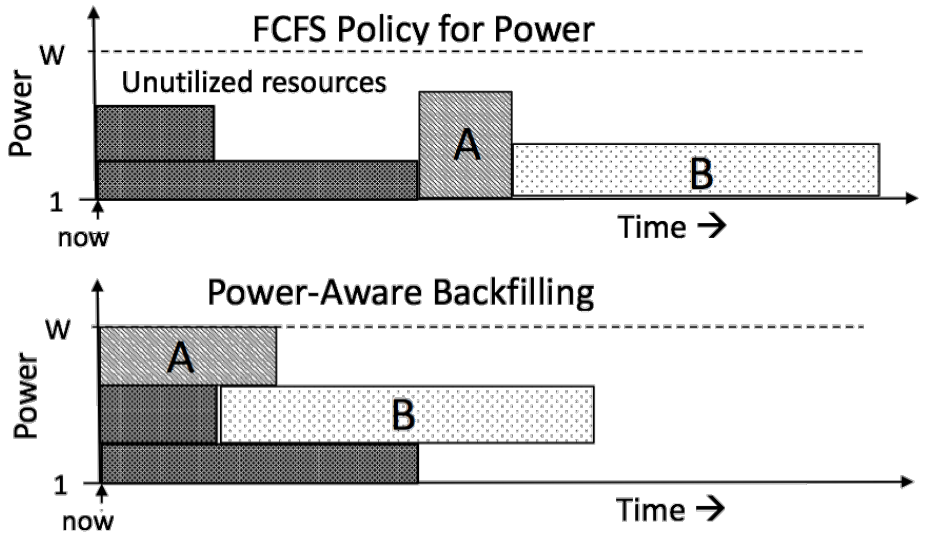
Experiment	Runtime	Stddev	Improvement
115W static	278.26	9.57	
115W dynamic	276.24	4.84	0.7%
90W static	284.63	3.20	
90W dynamic	277.13	5.04	2.6%
70W static	323.83	4.90	
70W dynamic	278.02	4.97	14.1%
50W static	407.21	18.00	
50W dynamic	371.92	13.23	8.7%

(4) How to Coordinate to Schedule Resources?



Power-aware Job Scheduling

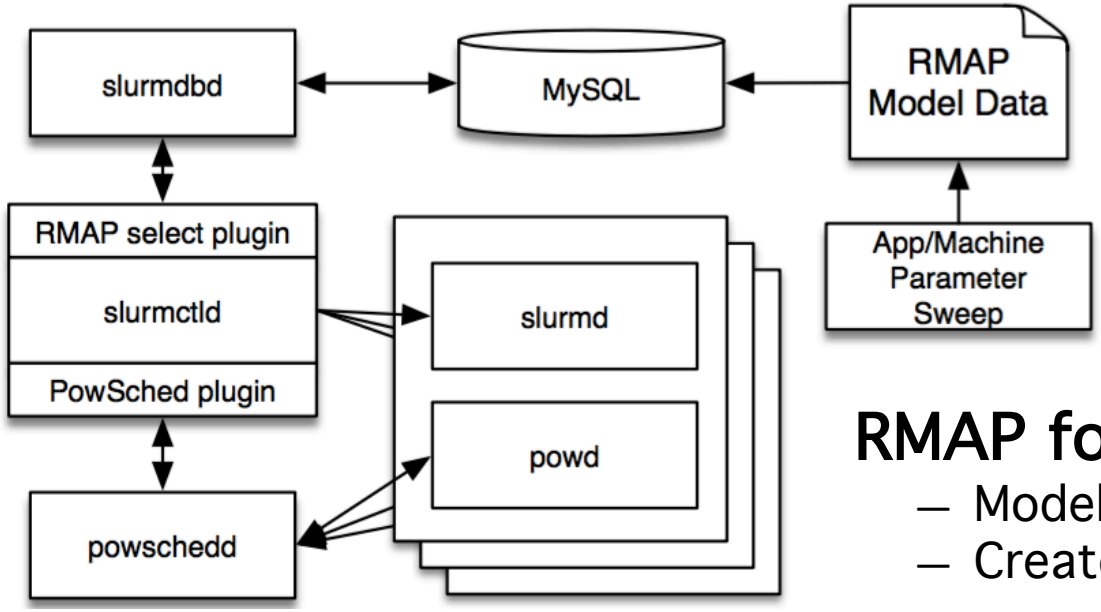
- Power as a controlled resource
- RMAP approach
  - Power aware backfilling



FLUX Resource Manager

- Hierarchical resource control

Integrating Static & Dynamic Power Management



Prototyping in SLURM

- Using SLURM plugins
- Currently being implemented in a large scale live testbed

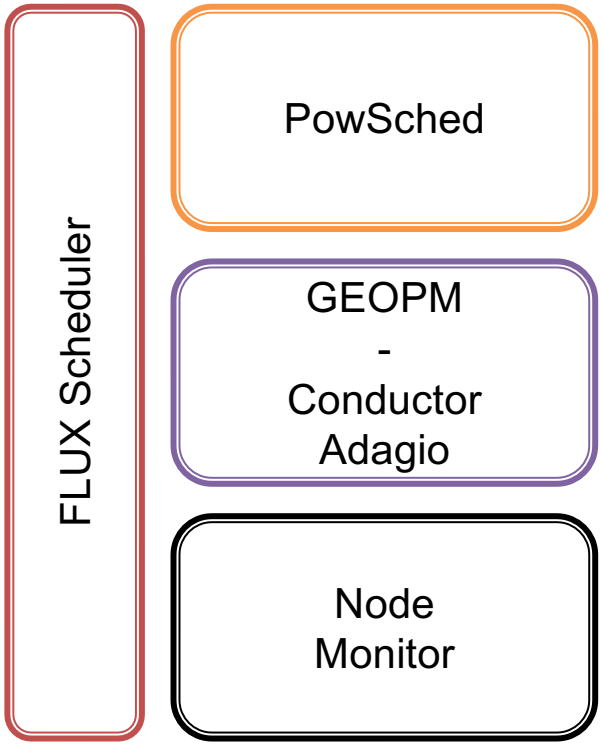
RMAP for static allocation

- Model data based on prior runs
- Creates job power budgets

PowSched for dynamic optimization

- Dynamically adjust budgets
- Maintain global total
- Ensure fairness

Open Challenges



Interfaces for node management

- Adjust to requests
- Enforce limits
- Global coordination on limits

Application input

- Caliper is the mechanism
- Need proper taxonomy
- Phases as a start, what else?

Coordination between PowSched & GEOPM

- Running at the same time
- Avoid conflicts

Interfaces to FLUX

- Need better prediction mechanisms

Conclusions


Power and energy are critical constraints for HPC

- Gaining in importance for centers
- Need software infrastructure to manage

Need full system software stack approach

- Node local management
- Per job management
- System-wide dynamic management
- Resource management

Each component shows promising results

- Prototypes in simulation and on real hardware
- Integration currently in progress
- Challenge: Interfaces

Towards a power-aware software stack for ECP