Dynamic Resource Partitioning for Complex Workloads using Containers

Swann Perarnau @ Argonne National Laboratory

NNSA National Nuclear Security Administration
U.S. DEPARTMENT OF ENERGY | Office of Science

Significant Changes in Workloads at the Node Level

New workloads are becoming popular across HPC facilities, with more complex resource requirements inside a node.

- Still single-user jobs but more complex than MPI+X
- Provide opportunities for more complex optimizations at the OS-level

Ensemble Simulations

Large collections of small jobs:

- Small # of ranks (even single rank), short runtime,
- Parameter-space exploration: jobs possibly generated on-the-fly,
- Co-scheduling multiple jobs on same node for throughput
- Independent simulations, sharing hardware resources (e.g. caches)

Coupled Codes

Several components of the same job running side-to-side on the nodes

- Ex: simulation + in-situ analysis + checkpointing + …
- Each component has its own resource requirements (# cores, mem B/W)
- Synchronisation/communication between all components
- No need for namespacing between components

Issues with Co-scheduling / Node Sharing

Not all node resources can be split evenly or have users isolated from each other:

- Shared hardware resources: caches, memory bandwidth
- Bindings required on discrete resources (cores)
- Best-effort QoS from the OS (disk, virtual memory)

Argo Node Resource Manager

Node-local infrastructure to manage hardware node resources:

- User-space daemon to monitor/partition/optimize resources across components
- Manages: cpu & memory bindings, power, cache bandwidth (soon)
- Can monitor MPI activity, hardware performance counters

Argo Containers

NRM uses containers to control resource utilization

- Each component runs in its own container,
- Users can create containers on-the-fly, with specific resource requests,
- OS and hardware enforce the resource constraints between containers

Not Your Regular Containers

Argo Containers are focused on resource control:

- No namespacing: all processes can communicate with each other
- No image infrastructure: all processes share the same node environment
- Focus on OS-enforced QoS for hardware, and advanced hardware features like Intel RDT.

Argo Containers are Compatible with Others

Users can directly run singularity or docker inside an Argo container

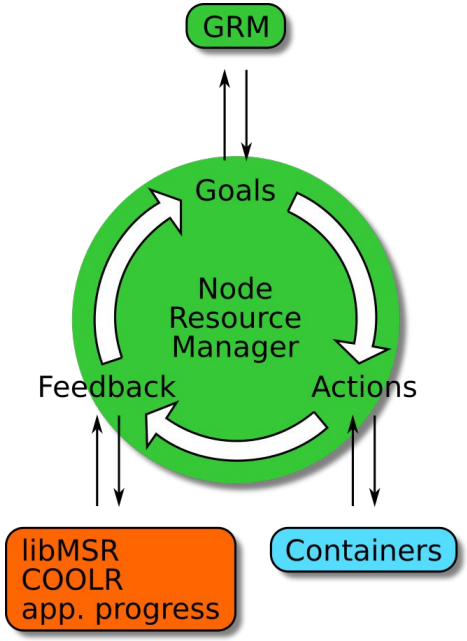
- Without any patches or specific configuration
- Working towards direct Open Container Initiative Runtime Spec support

How NRM manages resources

The inner design of the NRM is a control loop:

- Goals: received from scheduler/user
- Actions: software & hardware control
- Feedback: software & hardware monitoring

NRM configuration and user hints are used to decide applicable policies and actions.

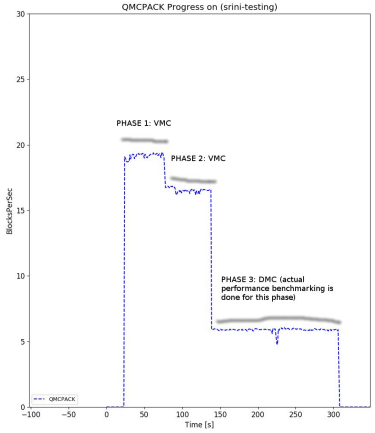


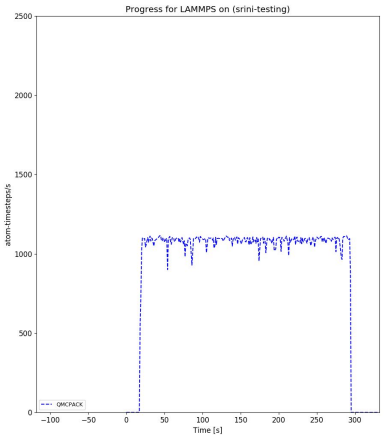
EXASCALE COMPUTING PROJECT

Application-level Progress

Application-specific indicator that work is advancing

- Act as a performance heartbeat,
- Used by NRM for feedback on actions,
- Can detect application phases,
- Sent through lightweight local messages.





EXASCALE COMPUTING PROJECT

National Nuclear Security Administration

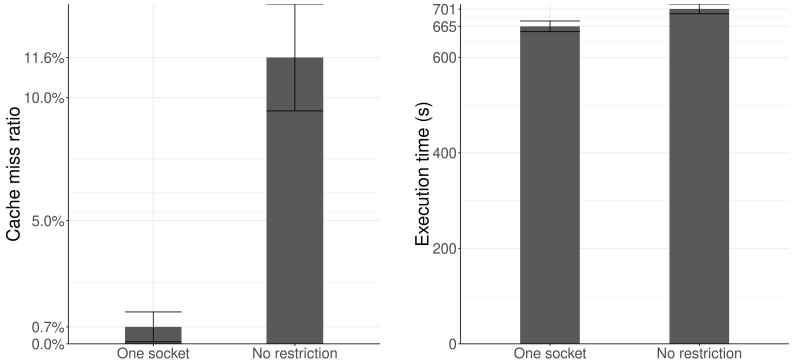
U.S. DEPARTMENT OF ENERGY | Office of Science

Cache Interference Management

Topology-aware core allocation policies

- NRM detects topology on startup
- Containers are packed to avoid shared cache interference

Working on Intel's Cache Allocation Technology to add cache bandwidth limits to our control.



EXASCALE COMPUTING PROJECT

Looking for early users / collaborators

- Workflows already using co-scheduling
- Applications with multiple phases, work imbalance
- Users with complex container setups
- Workload-specific resource allocation policies