Scheduling for future HPC systems

Gonzalo P. Rodrigo - gonzalo@cs.umu.se
Erik Elmroth – elmroth@cs.umu.se
Lavanya Ramakrishnan – lramakrishnan@lbl.gov
P-O Östberg – p-o@cs.umu.se

Distributed Systems Group – Umeå University, Sweden
Data Science & Technology – Lawrence Berkeley National Lab


UMEÅ UNIVERSITET
BERKELEY LAB

What is this talk about?

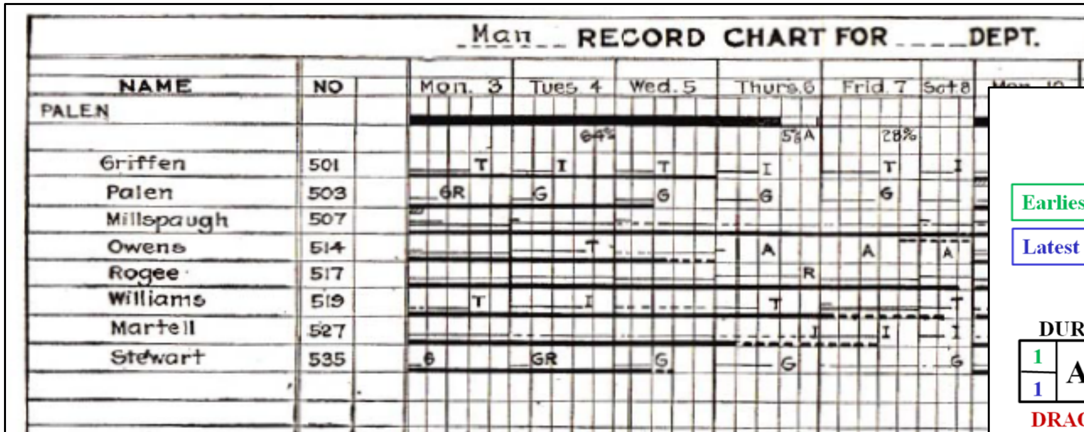
We analyzed the lifetime workloads of three last Berkeley Lab's HPC systems: **Job and application heterogeneity is giving the scheduler a hard time.**

Surveyed trends toward Exascale: **How systems heterogeneity and extreme parallelization will challenge schedulers**.

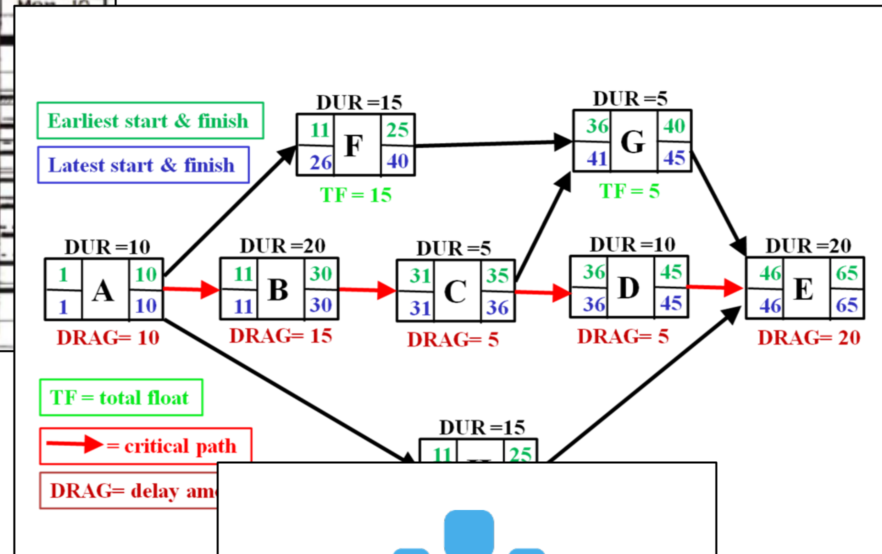
We talked with HPC users using workflows: **They wait forever or waste resources. We propose a solution.**

We designed a cloud inspired **scheduling model to cope with systems and workload heterogeneity.**

Present the toolset that made this scheduling research possible.



[2]



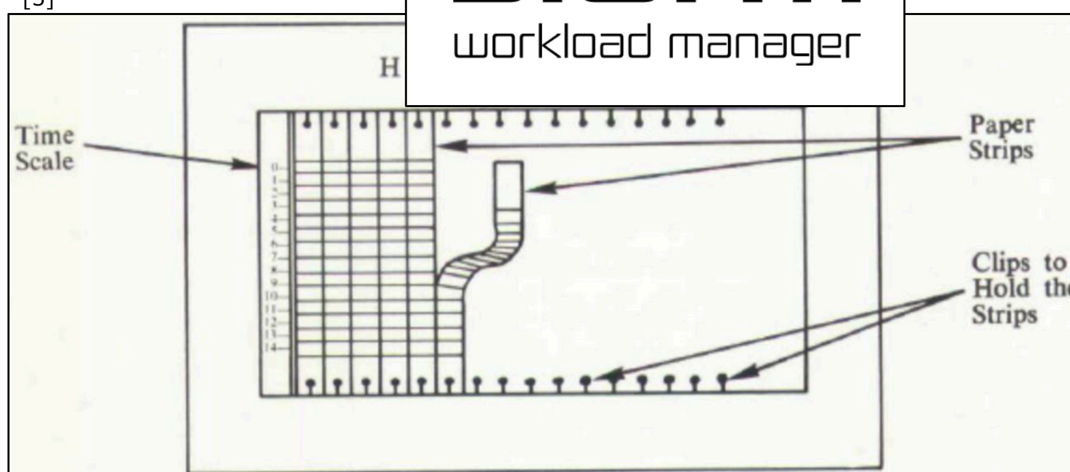
[1]

Batch scheduling basics

slurm

workload manager

[3]



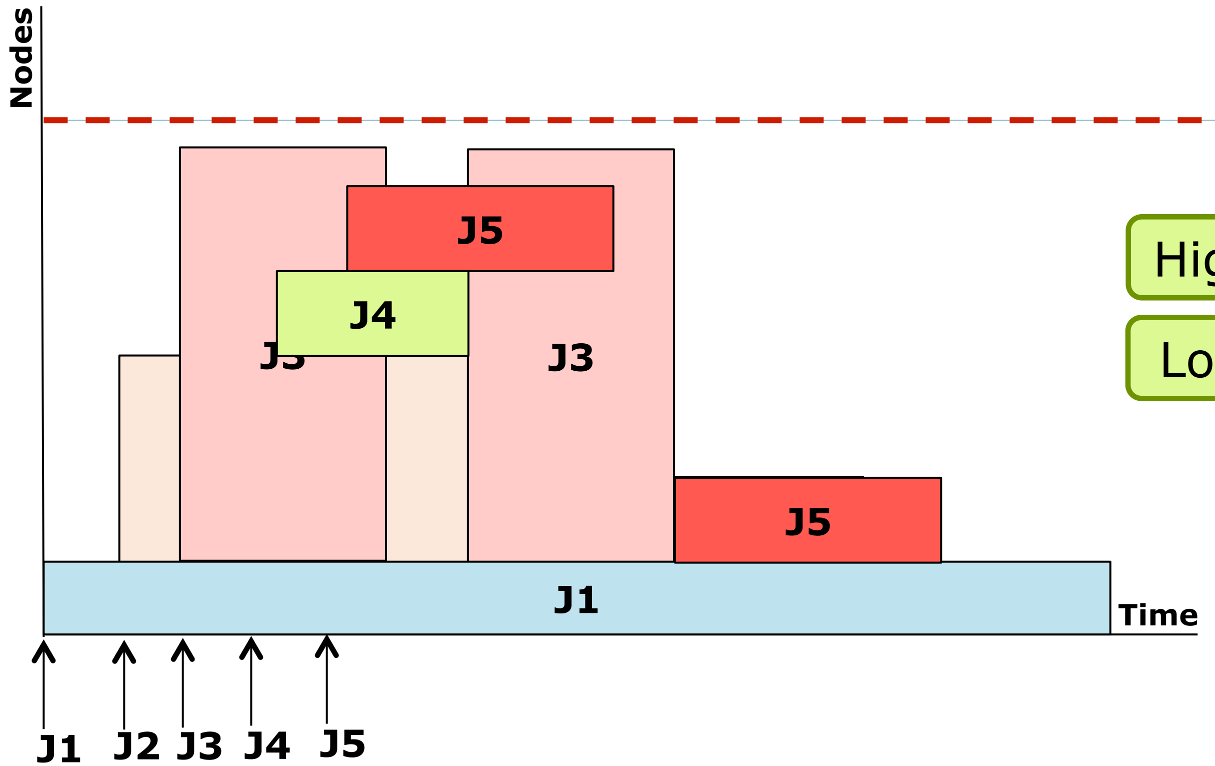
[1] CPM example

[2] Original Gantt chart by Henry Gantt

[3] Harmonogram by Karol Adamiecki

FCFS: Jobs execute in arrival order

Back-filling: Job can start if it does not delay previous jobs.



High Utilization

Low Wait Time

Gonzalo P. Rodrigo – gonzalo@cs.umu.se

Batch Schedulers: Steering the system

Mechanisms to reorder the waiting queue

Fairness

Don’t starve jobs or users

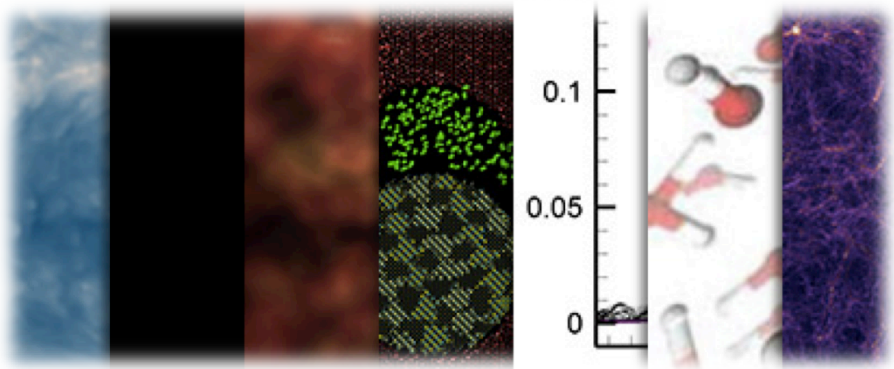
Priority

Run more important jobs first

[1]

Workloads and Systems

[2]



[1] Aurora Supercomputer: http://aurora.alcf.anl.gov

[2] Visualization elements from climate science, design accelerator design, biological research, transportation improvement, chemistry, and cosmology: http://aurora.alcf.anl.gov

UMEÅ UNIVERSITET

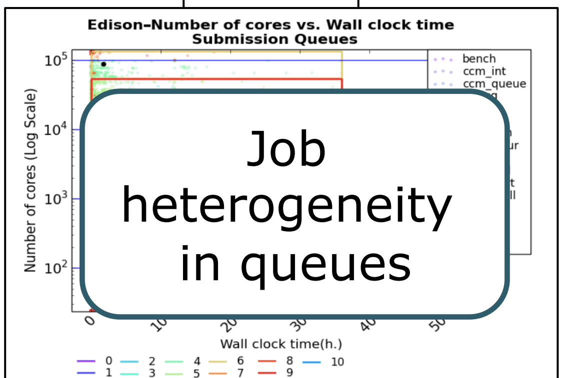
BERKELEY LAB

Present and future workload: Diversity and evolution

Job diversity

Data intensive Applications

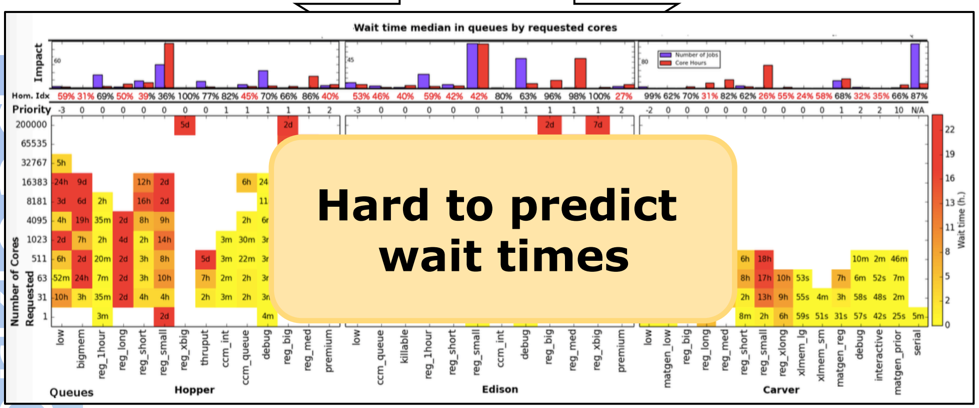
Application diversity



Job heterogeneity in queues

Different performance Metrics

Different performance Model



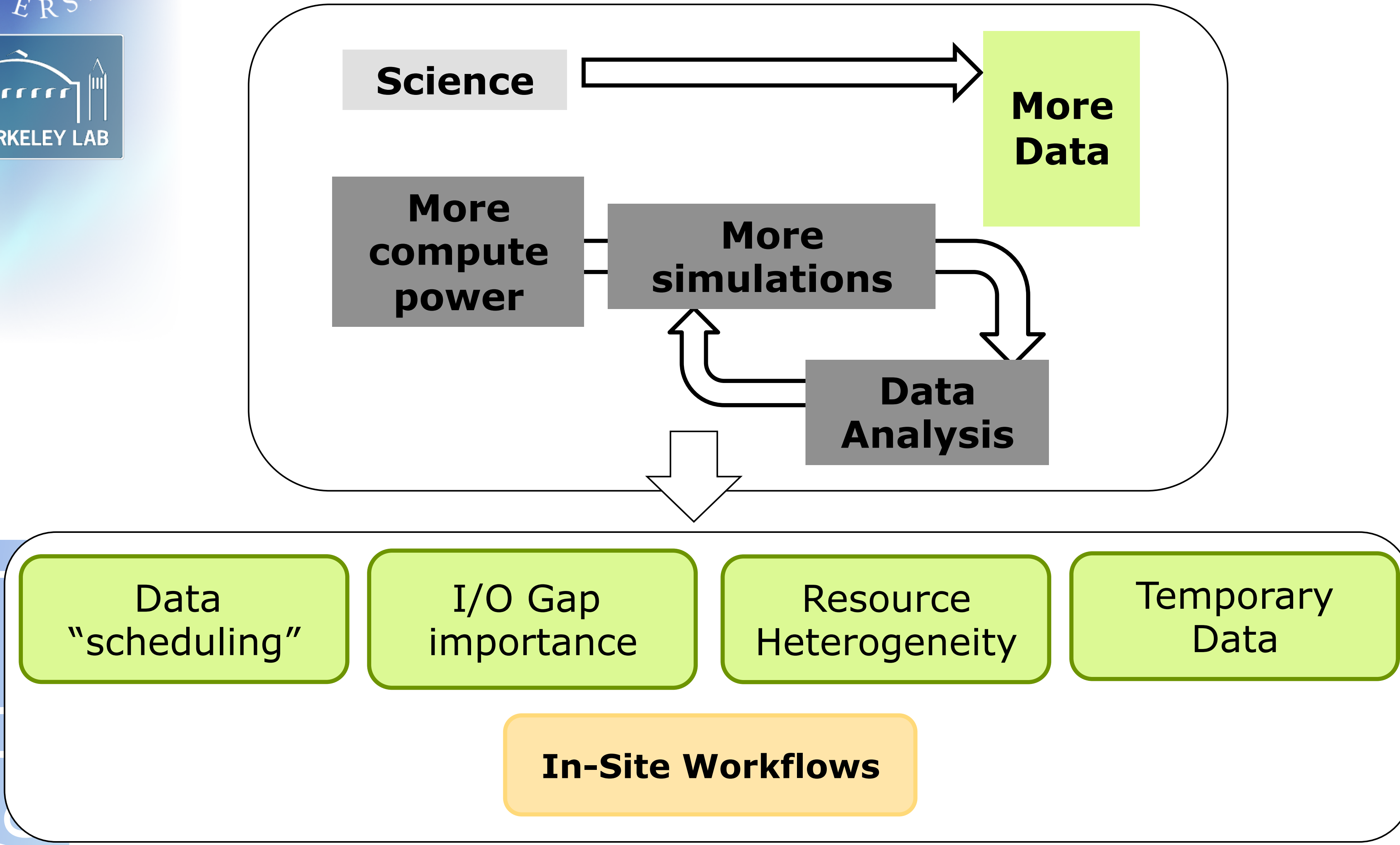
Hard to predict wait times

Scheduler Comparing Apples & Pears

G. Rodrigo, P-O. Östberg, E. Elmroth, K. Antypas, R. Gerber, and L. Ramakrishnan. Towards Understanding Job Heterogeneity in HPC: A NERSC Case Study. CCGrid 2016 - The 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2016.

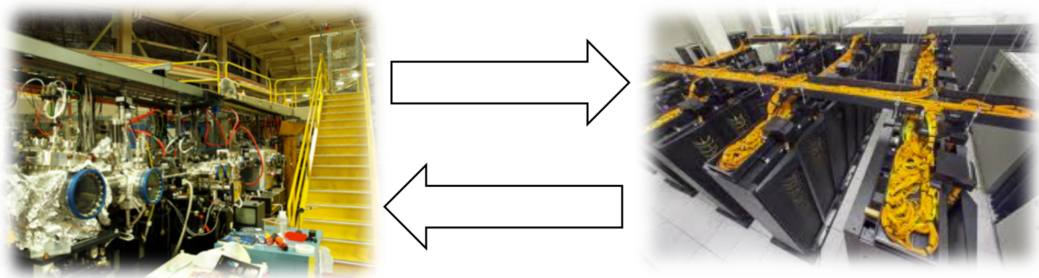
G. Rodrigo, P-O. Östberg, E. Elmroth, K. Antypas, R. Gerber, and L. Ramakrishnan. (2015, June). HPC System Lifetime Story: Workload Characterization and Evolutionary Analyses on NERSC Systems. In Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing (pp. 57-60)

Present and future workload: Data explosion [1]



[1] Tansley, Stewart, and Kristin Michele Tolle, eds. The fourth paradigm: data-intensive scientific discovery. Vol. 1. Redmond, WA: Microsoft research, 2009.

Present Workload: New, "old" applications



Real Time Applications

Grid workflows = In-Site Workflows

Gonzalo P. Rodrigo – gonzalo@cs.umu.se

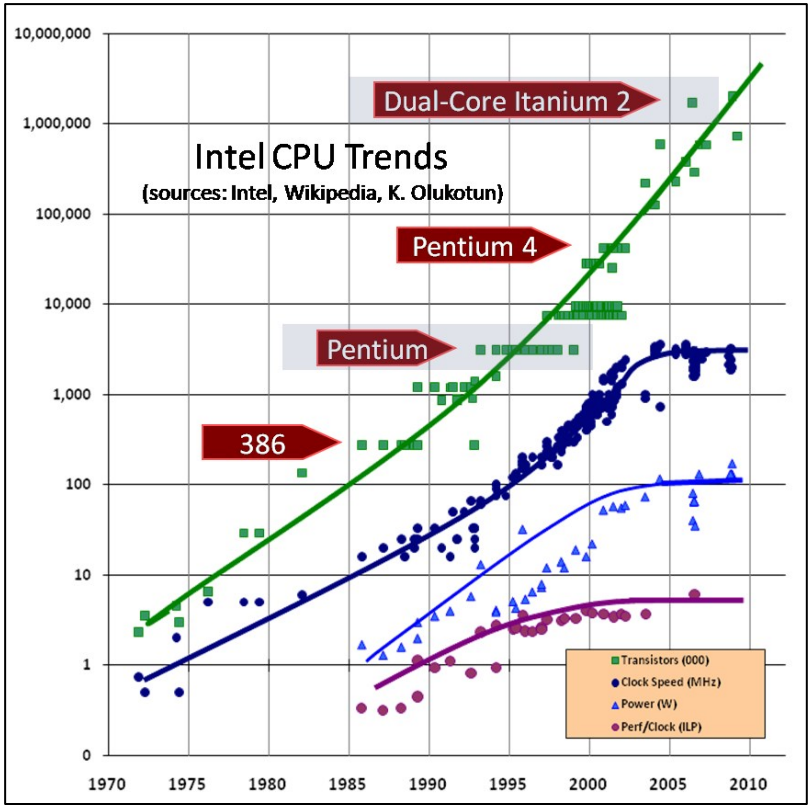
Why Exascale?

Science is fueled by computation:
More power, more science.

Grid based simulations (e.g. climate) require more resolution:
More parallelism.

Exascale: What is the challenge?

It's all about power and cost



[1]

Higher degree parallelization

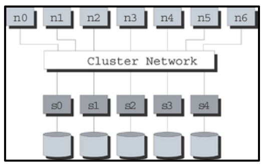
Break down of Dennard scaling

Systems

[1] http://www.extremetech.com/computing/116561-the-death-of-cpu-scaling-from-one-core-to-many-and-why-were-still-stuck

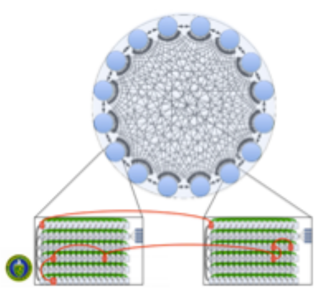
Gonzalo P. Rodrigo – gonzalo@cs.umu.se

Raw Exaflops are possible by increasing the number of CPUs but…



I/O — Only scalable in parallel!

RAM — Power hungry!



Interconnect — More parallelism => More complexity
Less uniform latency

The Exascale paradox

Heterogeneous HW

Per Thread

Systems

Very little

Compute Power	Operation Specific HW Coordination, more stages. **Workflows!**
RAM	
PFS I/O BW	Complex I/O Hierarchy: **Burst Buffers**
Network BW	More in-chip Comms. **Placement**
Electric Power	Reduced Resilience!

A pre-Exascale system: HPC2N's Kebnekaise

Freshly deployed

- 432 classical compute nodes (12 096 cores)
- 20 large memory nodes (3 terabytes/node)
- 32 2xGPU Nodes (319 488 gpu cores)
- 4 4xGPU Nodes (79 872 gpu cores)
- 36 KNL Nodes (9 792 threads)

- 416 352 Cores
- 437 232 Threads
- 128 TBytes RAM
- 4 types of CPUS
- Non Uniform Memory BW

Heterogeneity: memory, compute, interconnect, and programming models.

A pre-Exascale system: Sunway TaihuLight

Systems

[1] **Sunway TaihuLight**
93.014 PFLOPS
US$273M
15 MW (No cooling)

X 11 →

1 Exaflop
US$3003M
165 MW

256+4 cores/CPU
Scratchpad memory
In Chip network
X3 Gflops/Watt

Bad HPCG Benchmark

Hard to program
Slow memory
Modest interconnect

[1] Dongarra, Jack. "Report on the Sunway TaihuLight system." www.netlib.org. Retrieved June 20 (2016).

Gonzalo P. Rodrigo – gonzalo@cs.umu.se

All challenges

System's Heterogeneity

Coordination, more stages,.

Job diversity

Real Time Applications

Application diversity

More in-chip Comms.

Data explosion

Complex I/O

Burst Buffers

Grid workflows

=

In-Site Workflows

Comparing Apples & Pears

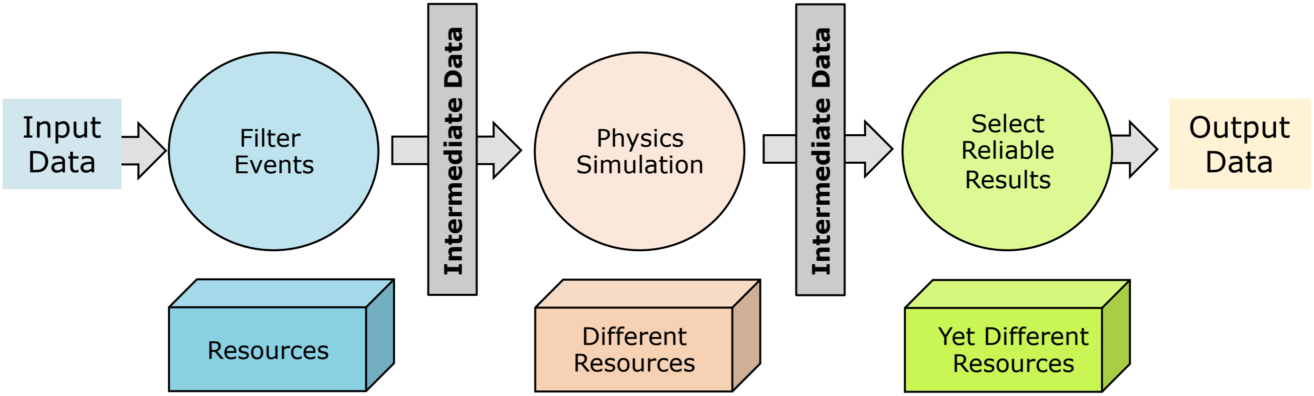
Hard to predict wait times

Workflow management & scheduling

Job and Application diversity

Data scheduling (workflows?)

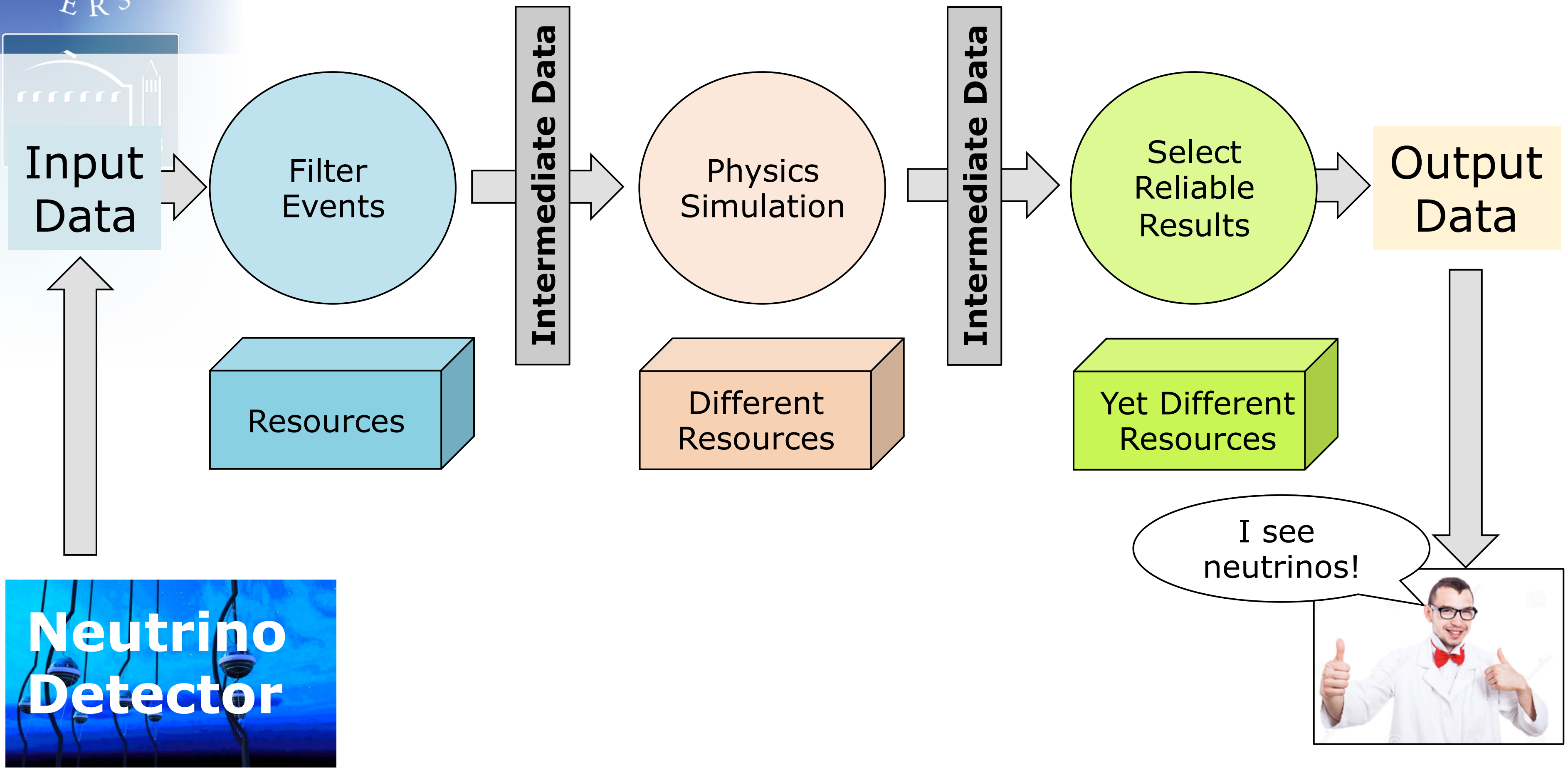
New time requirements



No Wait, No Waste: Workflow aware scheduling

To be submitted to IPDPS 2017

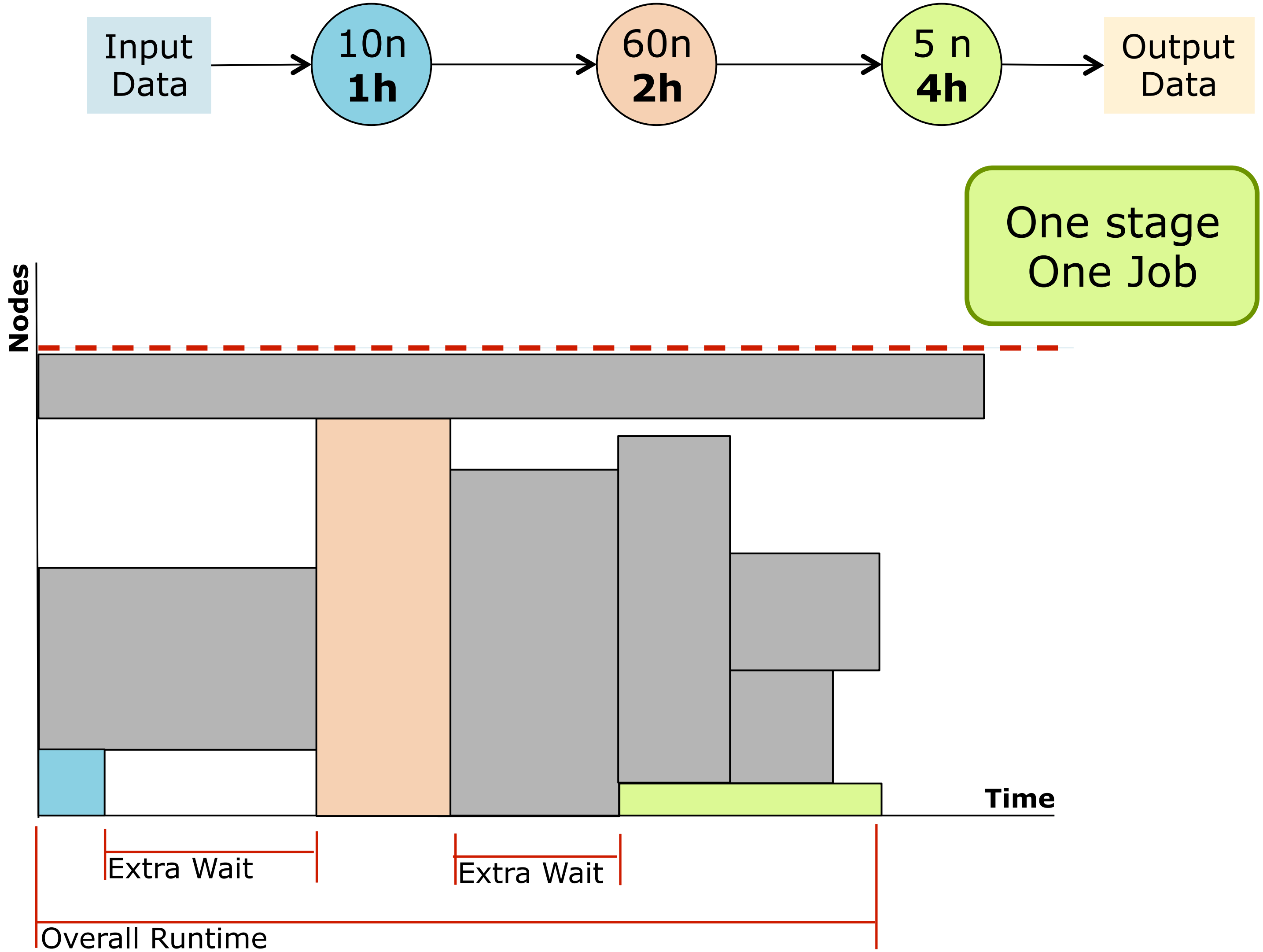
But before… What is a workflow?



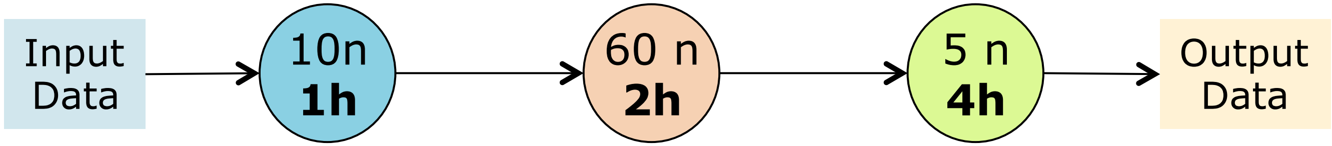
But..

How does a scheduler deal with **Workflows**?

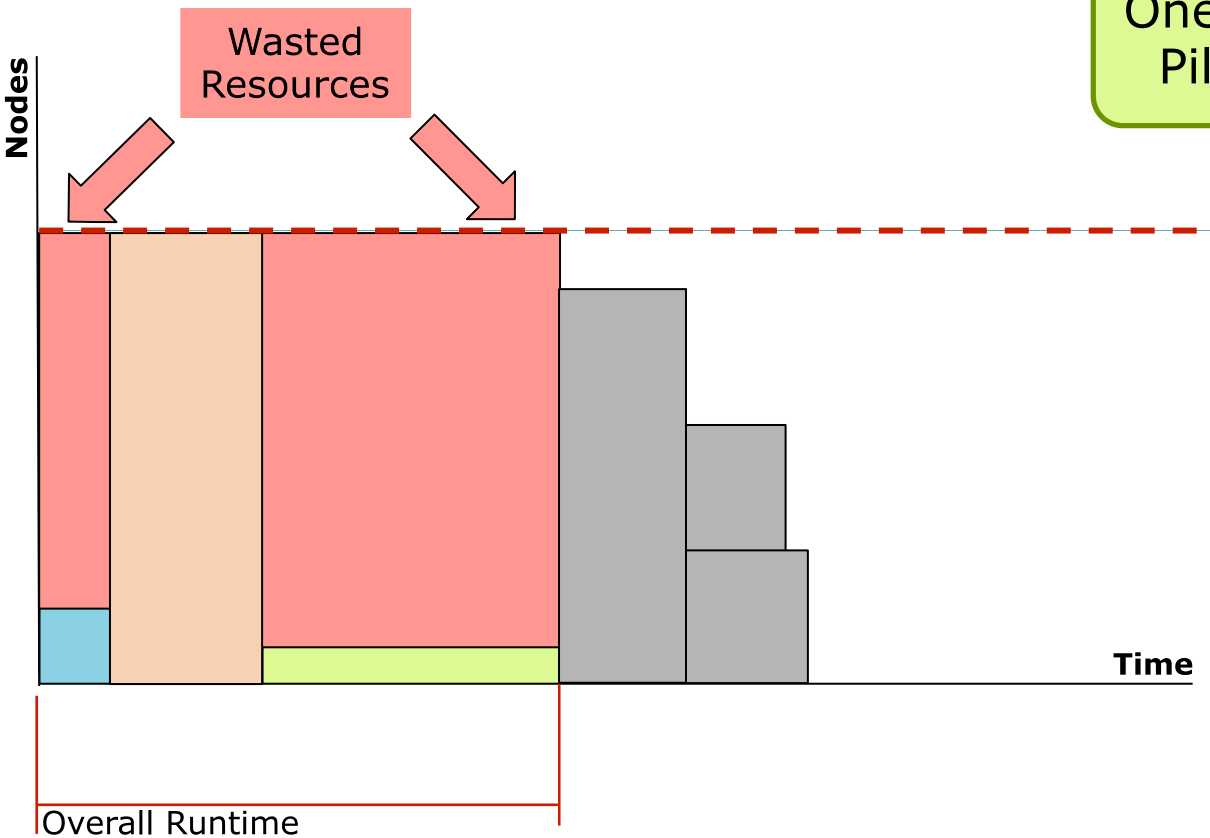
Submitting a workflow: Wait! (approach)



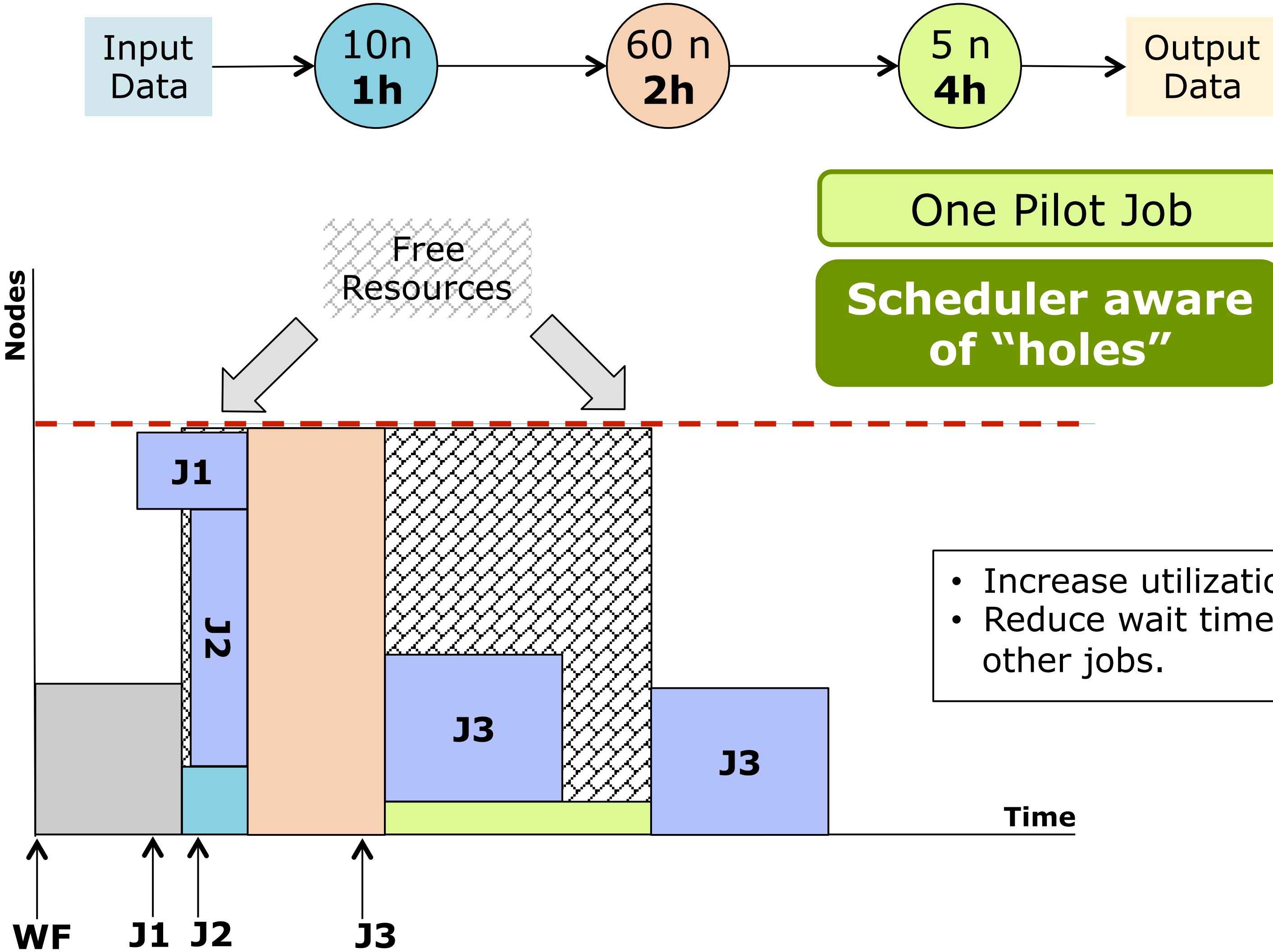
Submitting a workflow: Waste! (approach)



One single Pilot Job



Workflow aware scheduling: Backfilling

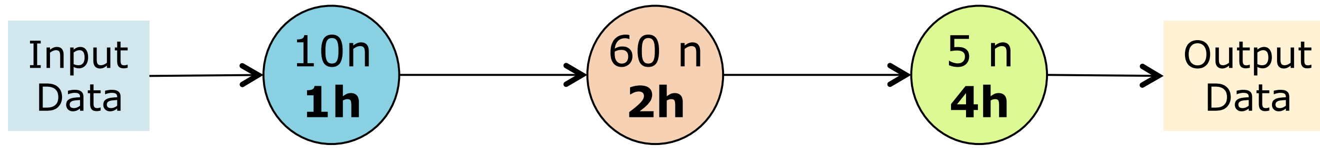


One Pilot Job

Scheduler aware of "holes"

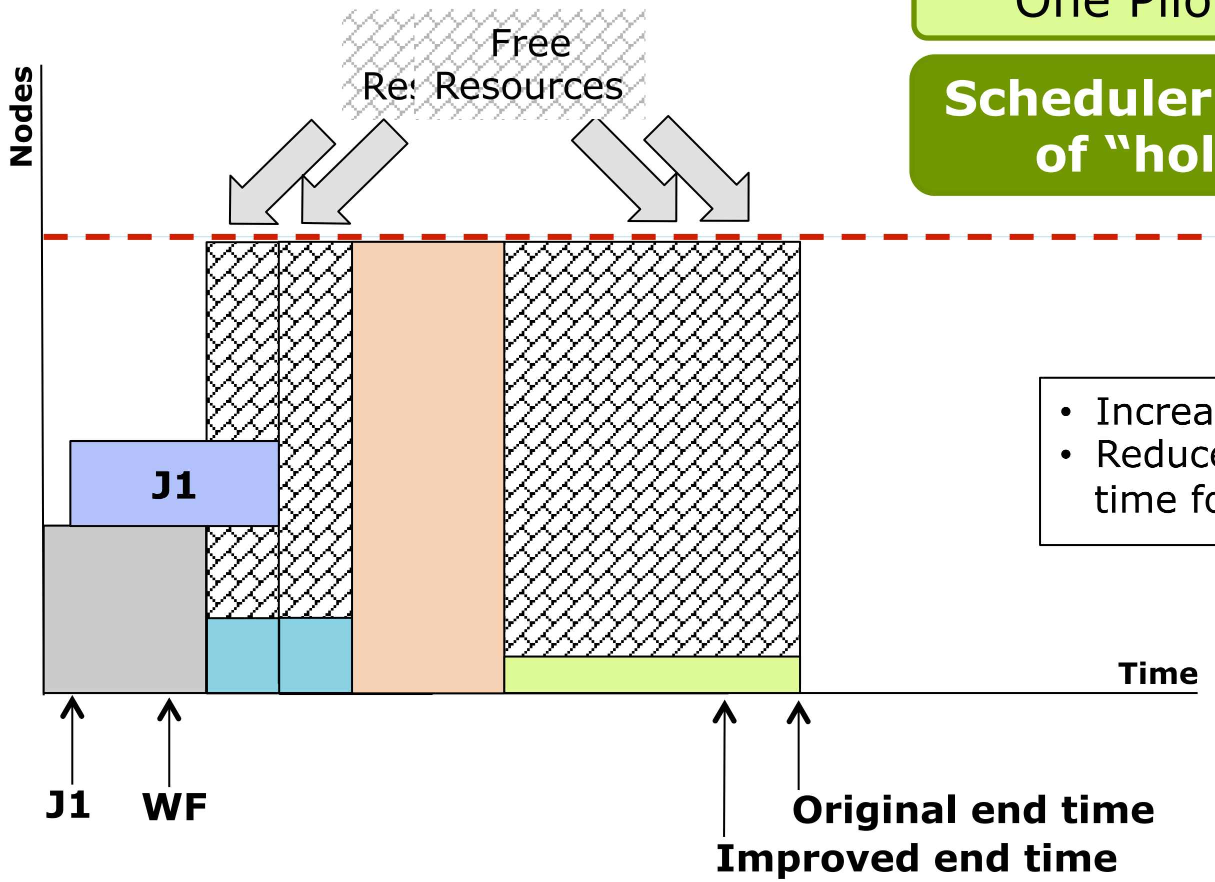
- Increase utilization.
- Reduce wait time of other jobs.

Workflow aware scheduling: FCFS



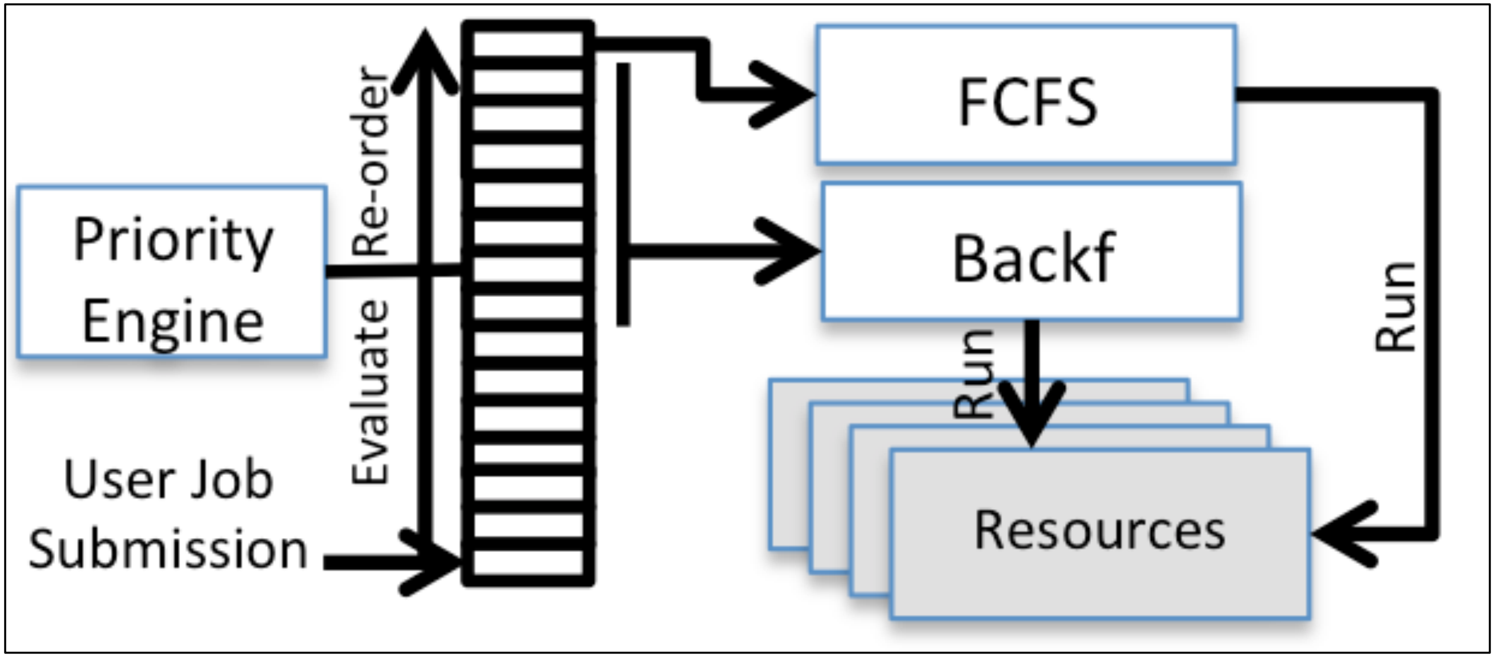
One Pilot Job

Scheduler aware of "holes"



- Increase utilization.
- Reduce turn around time for workflows.

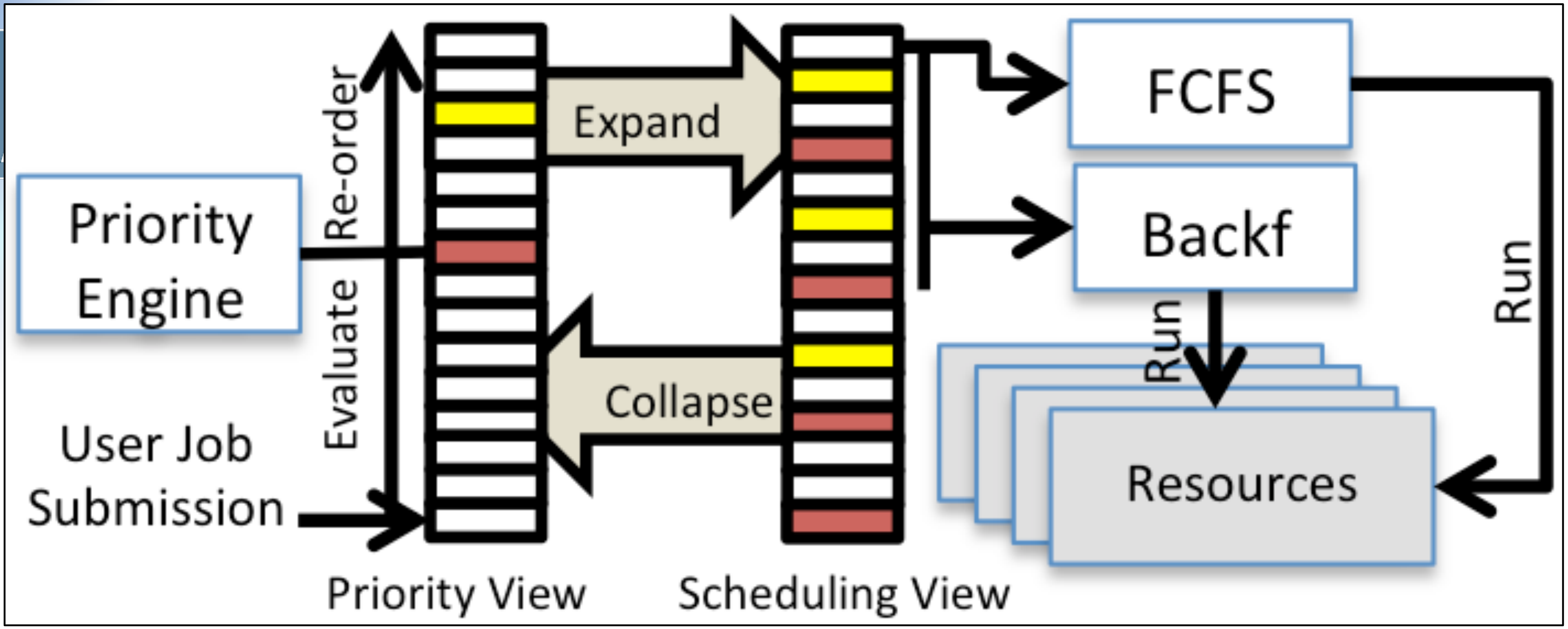
Workflow aware scheduling: Before



Scheduler Independent

Avoid System Gaming

Workflow aware scheduling: Scheduler View

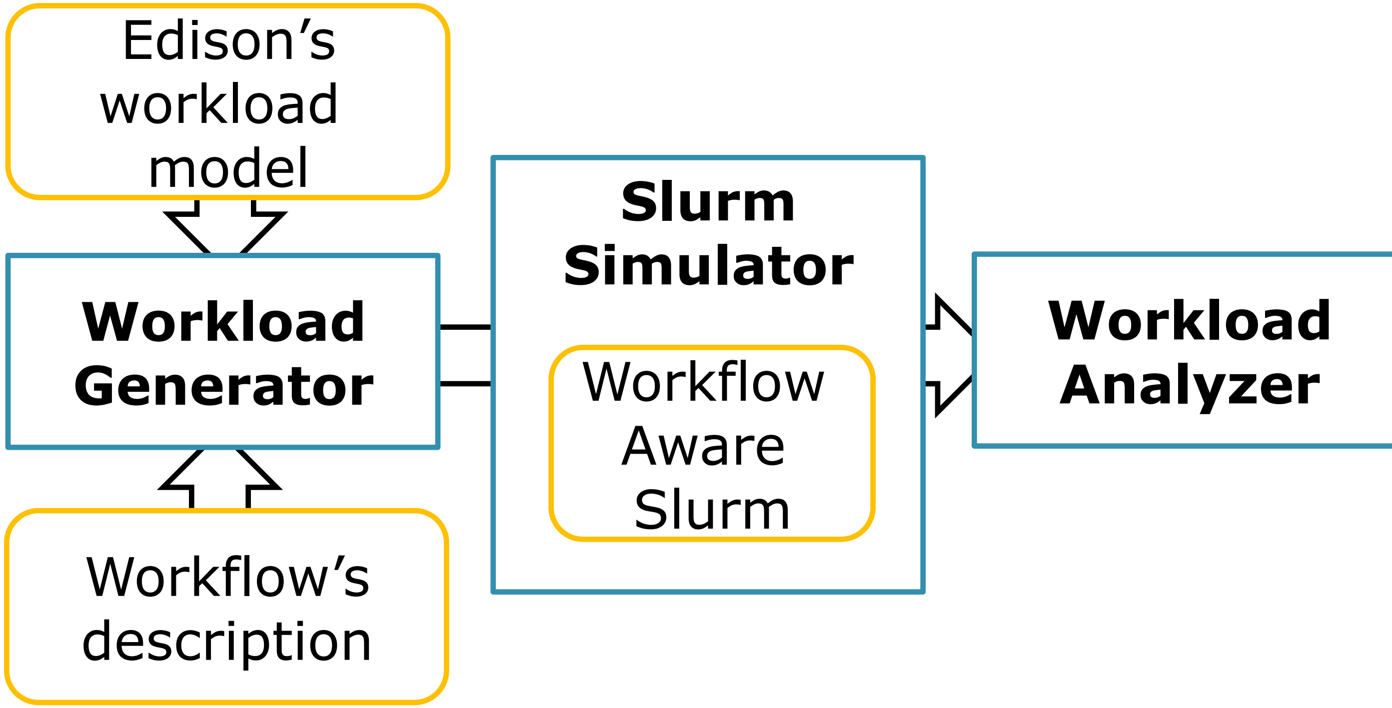


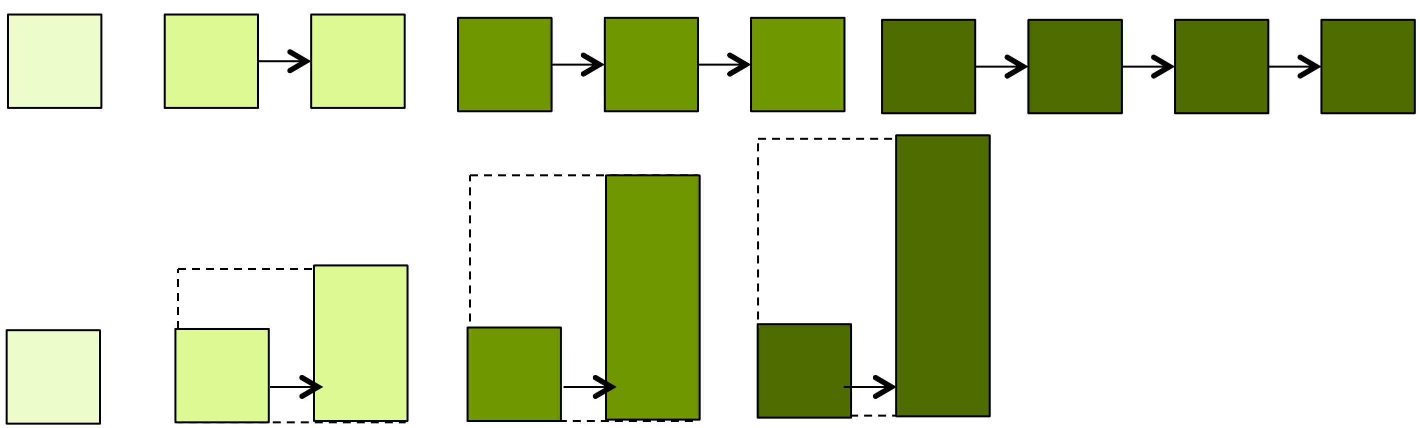
Wrap Scheduling Algorithms

Priority Workflow by Bounding job

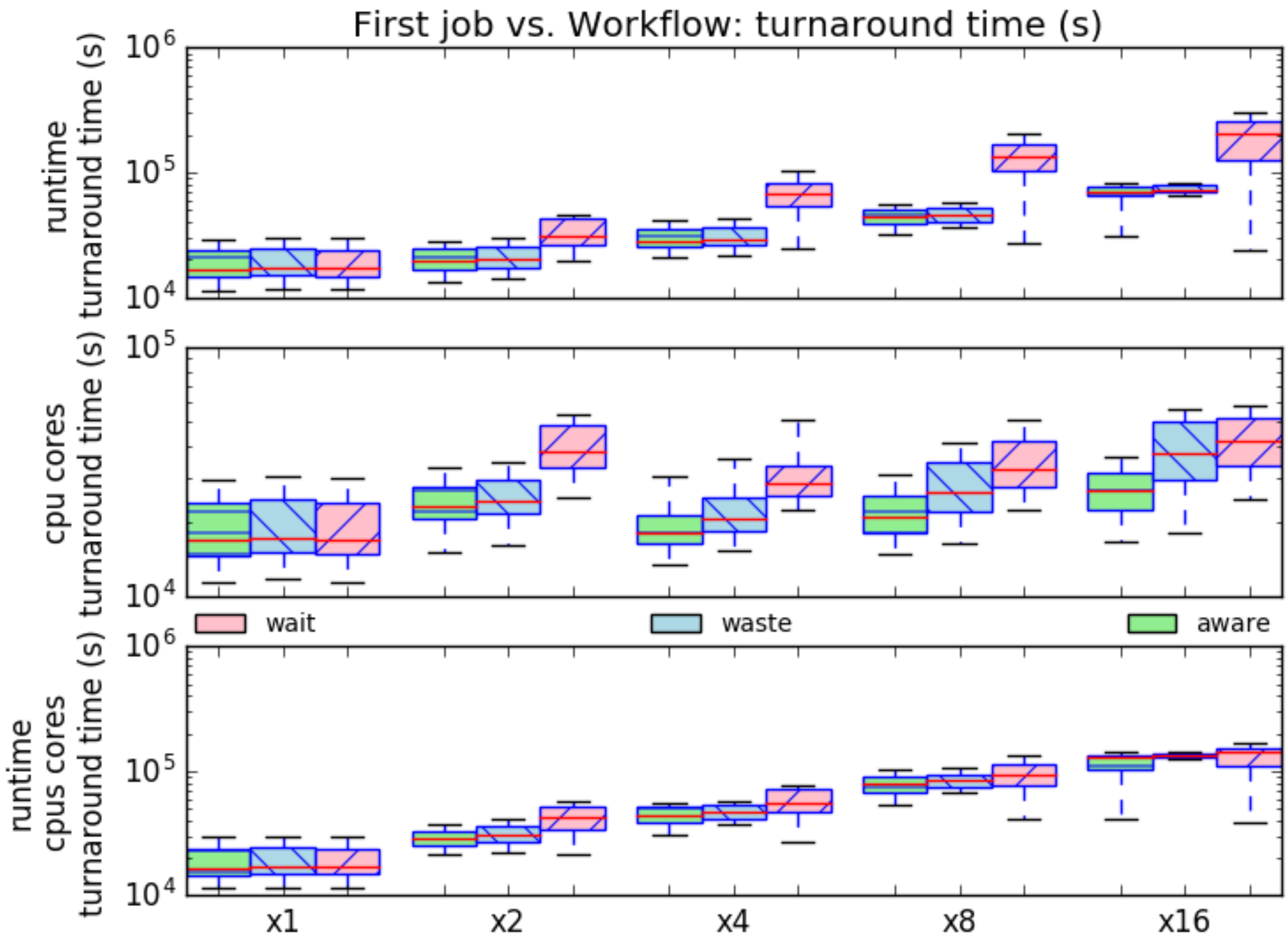
Experimentation: Modeling NERSC's Edison

Cray XC30
Aries Network
5,576 Nodes, 24 cores/node 133,824 cores
2.57 Pflops/s
SLURM





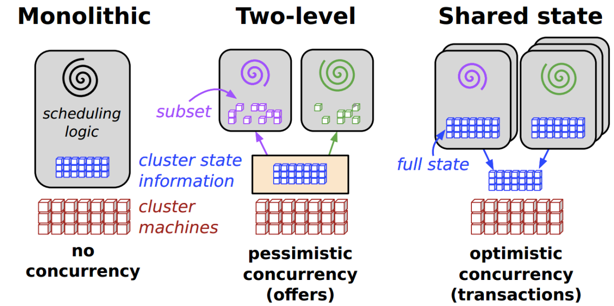
Workflow aware scheduling: Some results



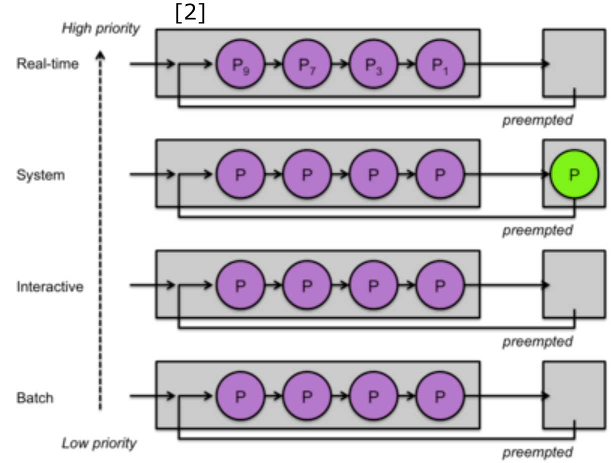
[3]

A2L2: Long term proposal

[1]



[2]



[1] Scheduling taxonomy: Schwarzkopf, Malte, et al. "Omega: flexible, scalable schedulers for large compute clusters." Proceedings of the 8th ACM European Conference on Computer Systems. ACM, 2013.

[2] Multilevel Queues: https://www.cs.rutgers.edu/~pxk/416/notes/07-scheduling.html

[3] Rodrigo Álvarez, G. P., Östberg, P. O., Elmroth, E., & Ramakrishnan, L. (2015, June). A2L2: An Application Aware Flexible HPC Scheduling Model for Low-Latency Allocation. In Proceedings of the 8th International Workshop on Virtualization Technologies in Distributed Computing (pp. 11-19)

Looking for inspiration… in the clouds.

Cloud infrastructures have faced similar challenges…

Hypothesis: Cloud scheduling techniques can be applied to tackle new HPC challenges.
Method: Compared study on techniques and application circumstances (Survey)

Similarities

Applications

Cloud	HPC
Heterogeneous Workload	**Heterogeneous Workload**
Batch Jobs	Many non tightly coupled
Data is Key	Non-classical HPC
Wait Time is important	Response time
SSDs on Nodes	**Burst Buffer**
Distributed Filesystems	
Heterogeneous resources	**Accelerator HW**, **BB nodes**, **Compute nodes**

Infrastructure

A2L2: Application Aware Flexible HPC Scheduling Model for Low-Latency Allocation

Application aware scheduling: Aware of characteristics, performance models, different rules for different types of job.

Dynamically malleable management: runtime re-scaling of jobs, performance based allocation.

Flexible backfilling: for better utilization

Low latency allocation: To allow allocation of jobs a short time after submission (stream job)

Position Paper

Scheduler model

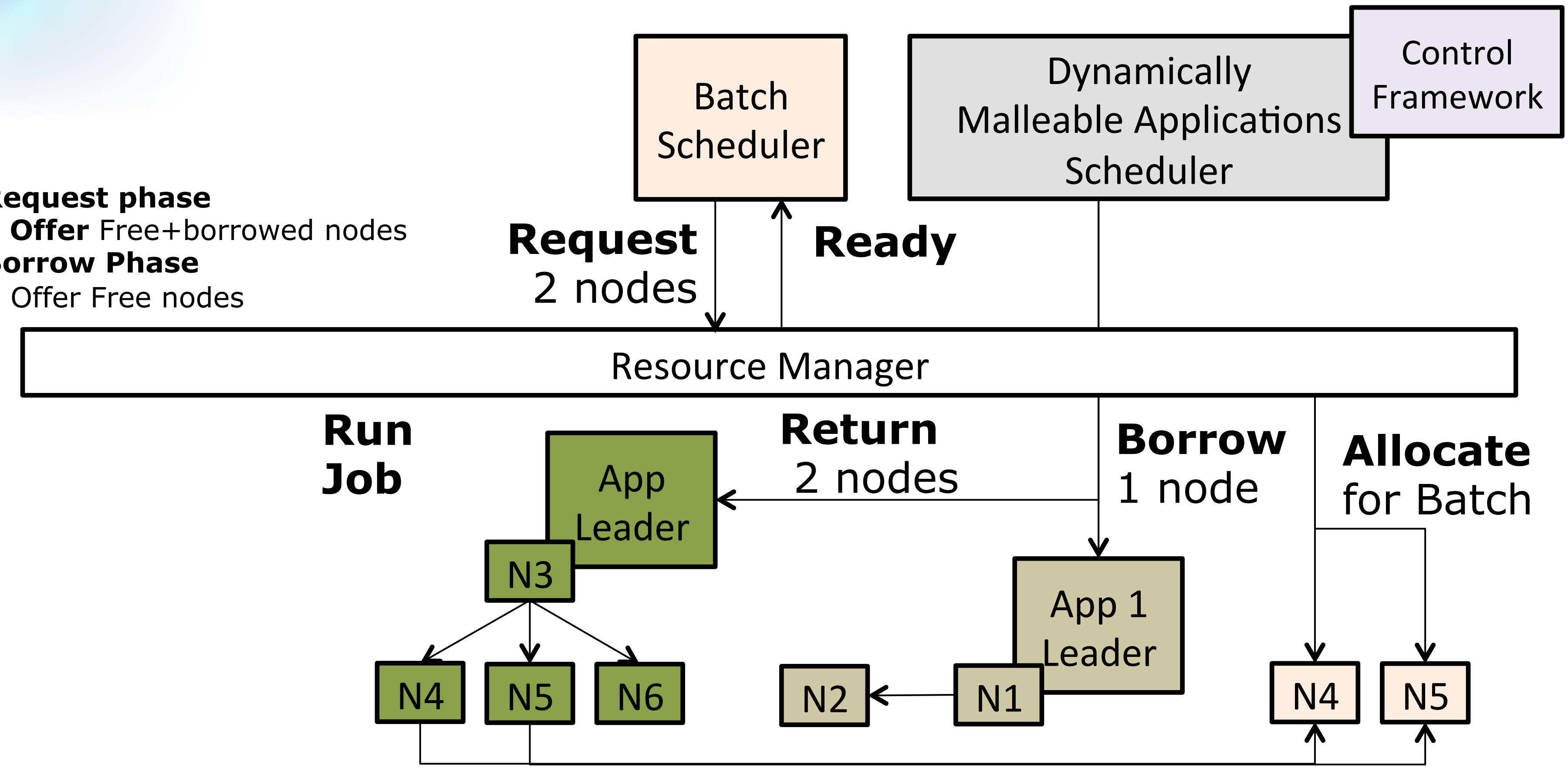
Cloud borrowed solution: **Two level scheduling**

One scheduler per application + smart RM

Malleable Applications: Dynamic allocation

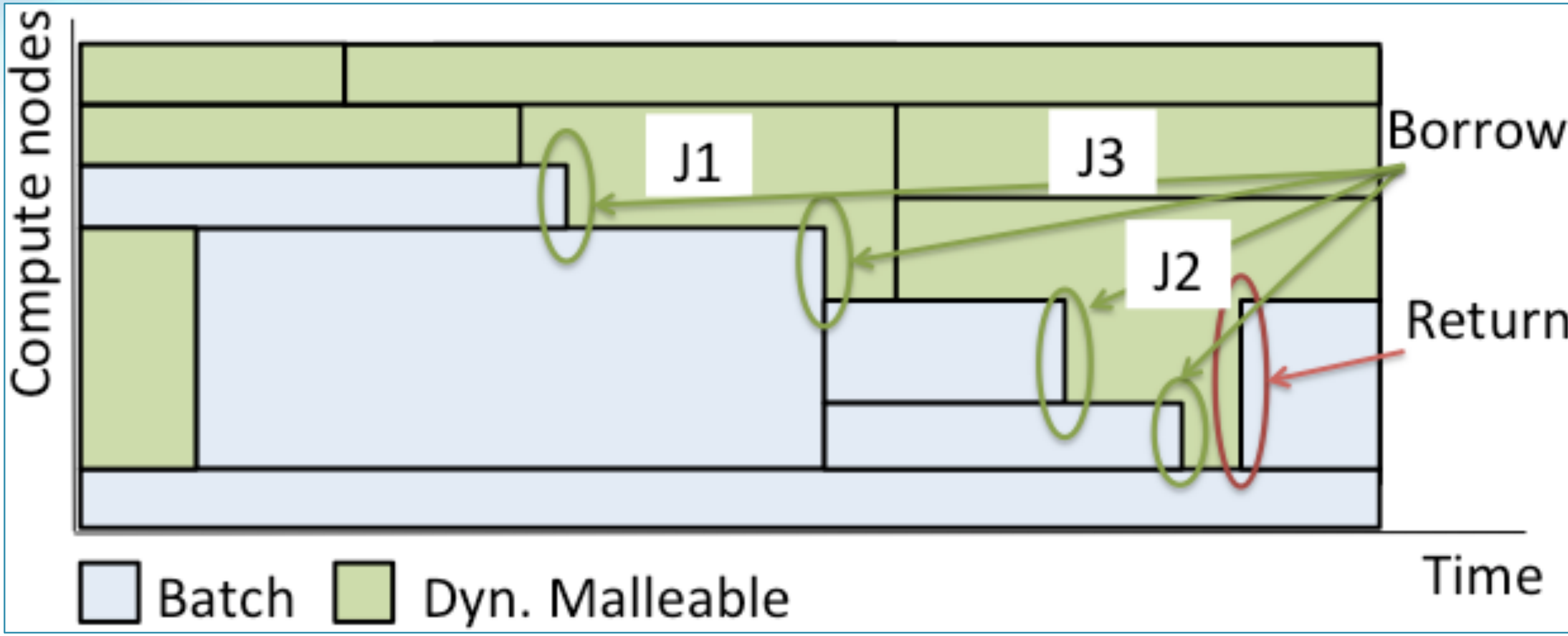
Low latency allocation

Request phase
Offer Free+borrowed nodes
Borrow Phase
Offer Free nodes



Gonzalo P. Rodrigo – gonzalo@cs.umu.se

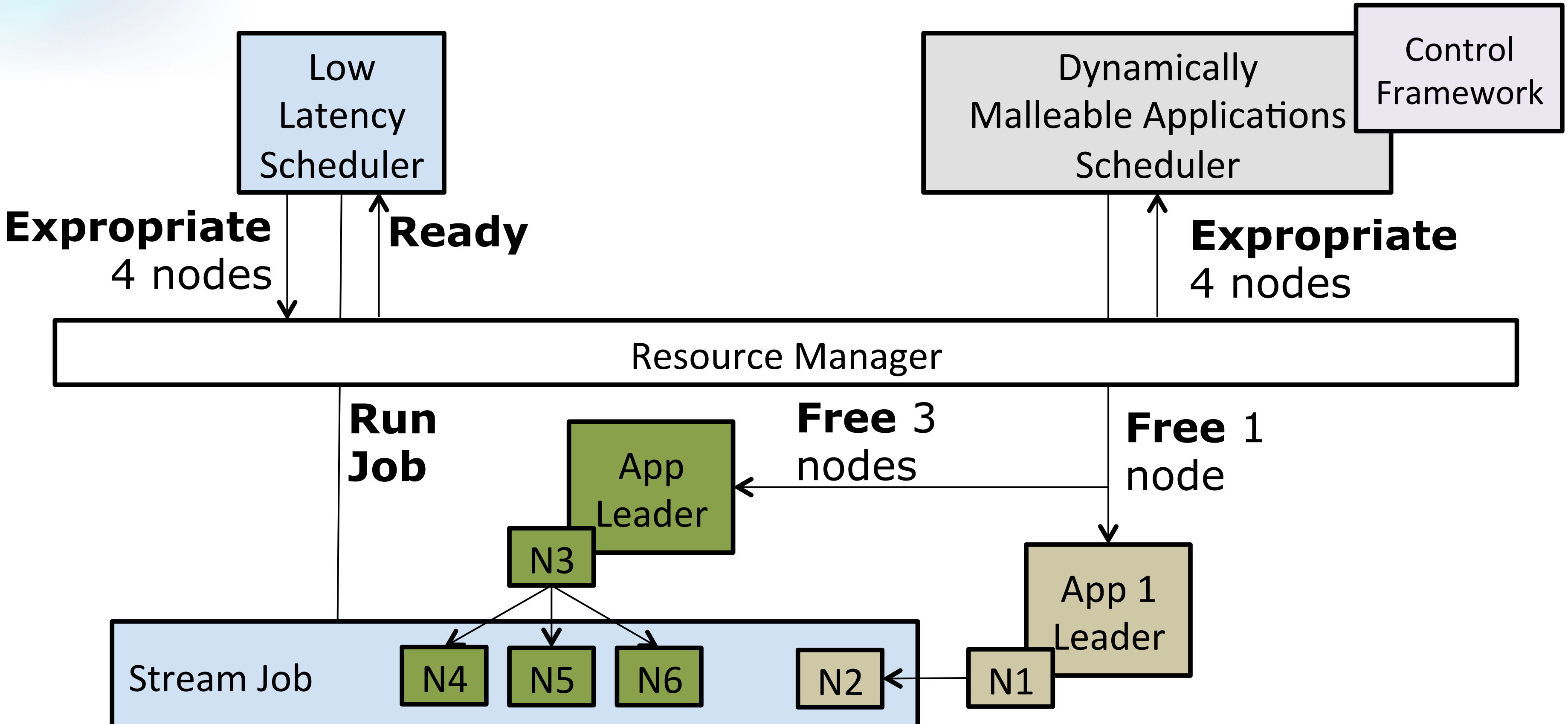
Flexible backfilling



Resource Expropriation: Low latency allocation

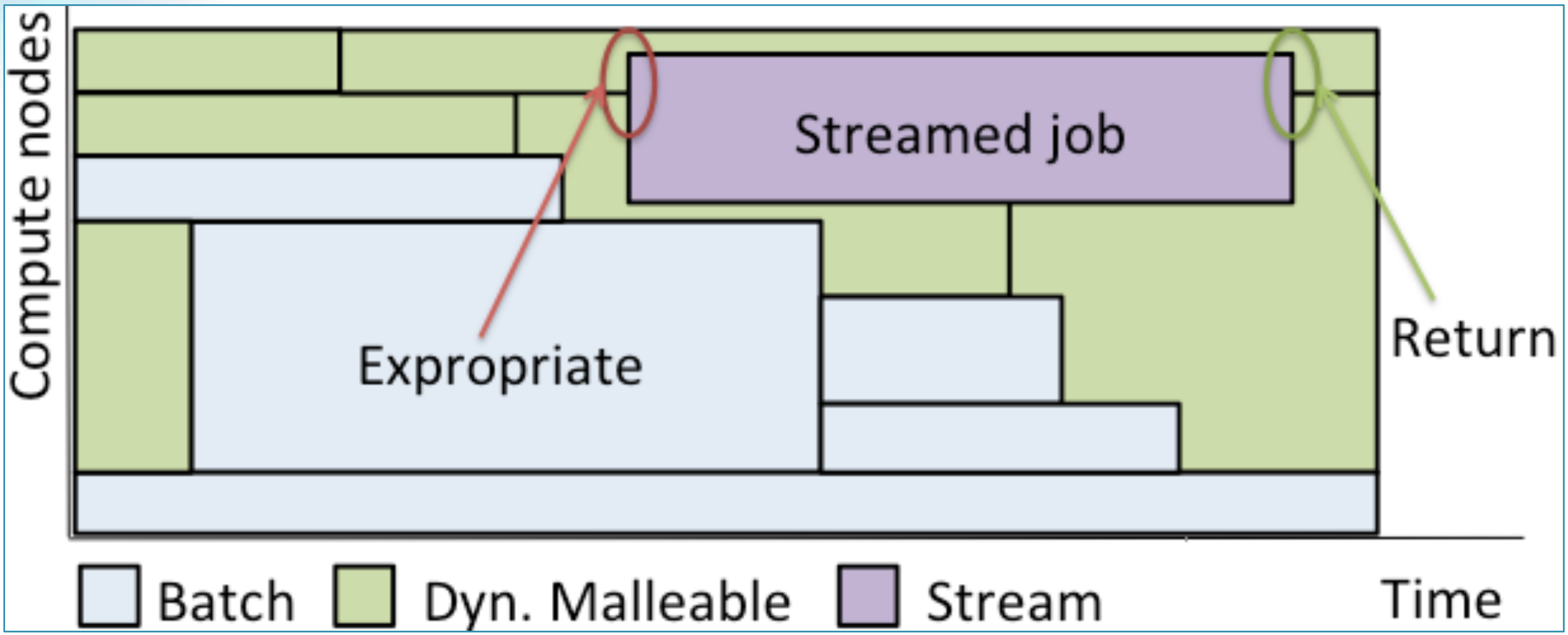
Temporary “expropriation” of resources assigned assigned to dynamically malleable applications

Expropriate and return actions



Gonzalo P. Rodrigo – gonzalo@cs.umu.se

Resource Expropriation: Low latency allocation



A2L2: Conclusions

Application heterogeneity is a trait of both cloud and HPC applications

Application Aware

Two level scheduling

Flexible nature of malleable applications can be useful (and there maybe enough malleable workload to make be useful)

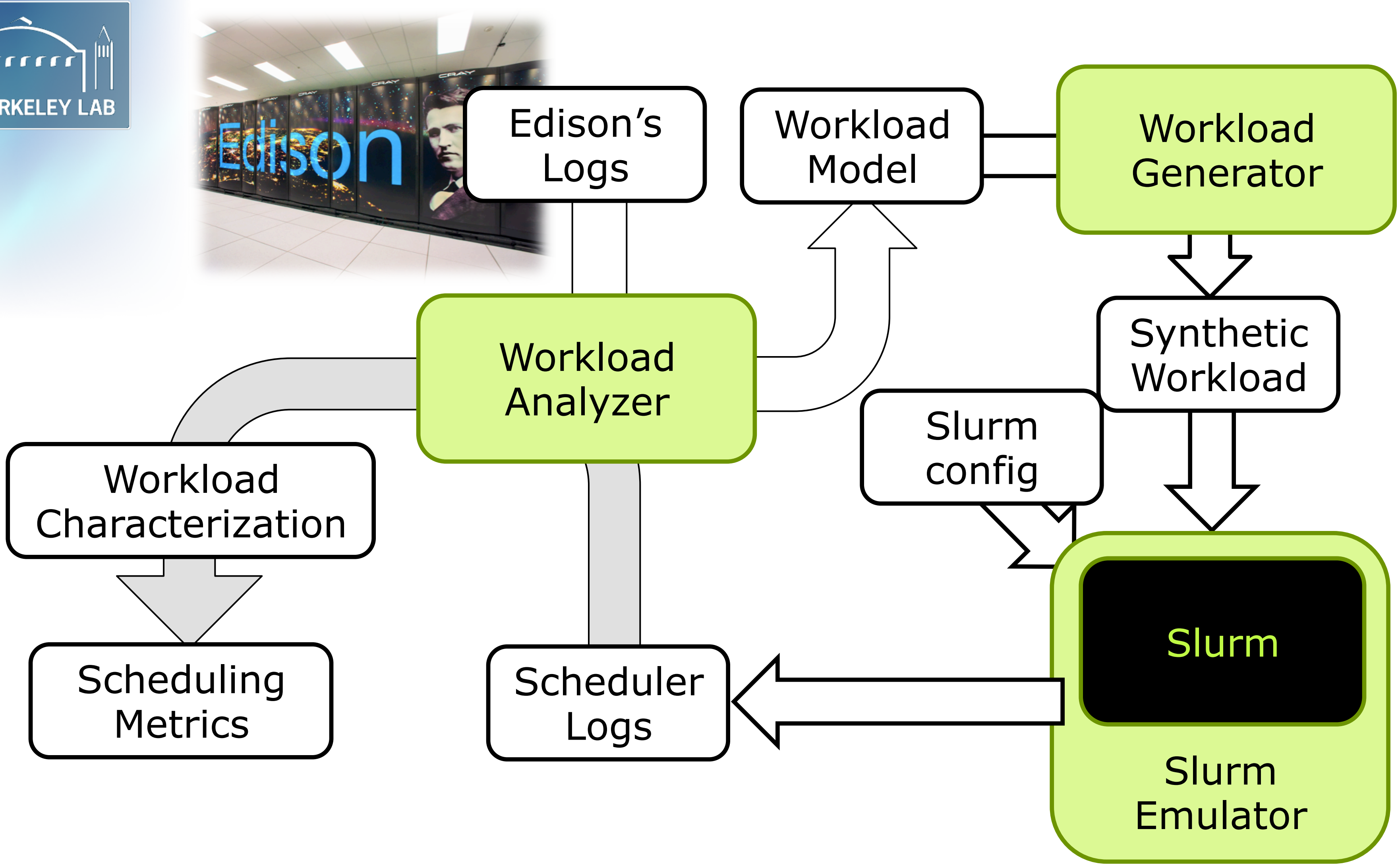
Application Management

Better utilization

Stream job allocation

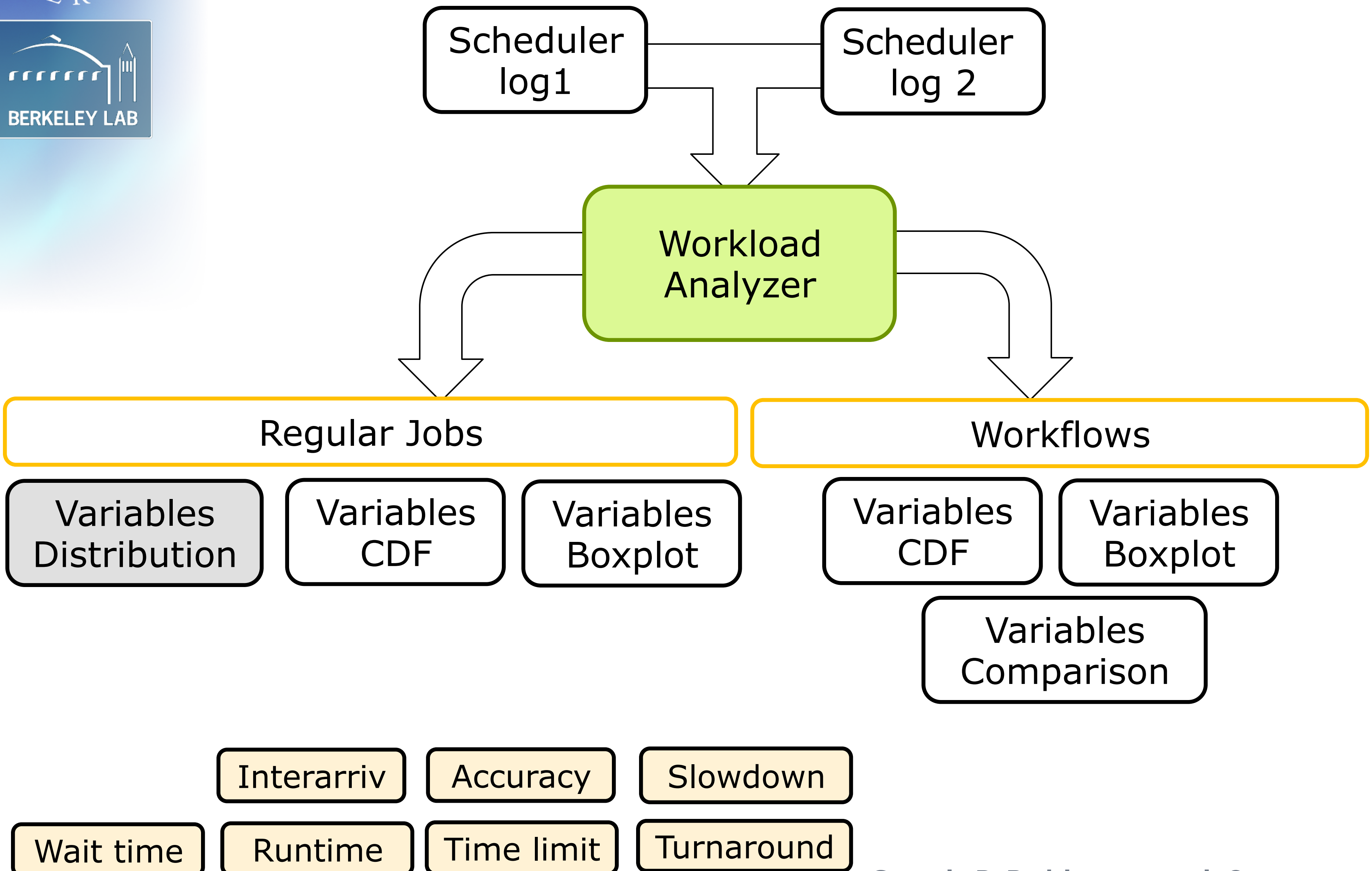
Scheduling: Challenges Research & Operational

Our work required too much engineering…

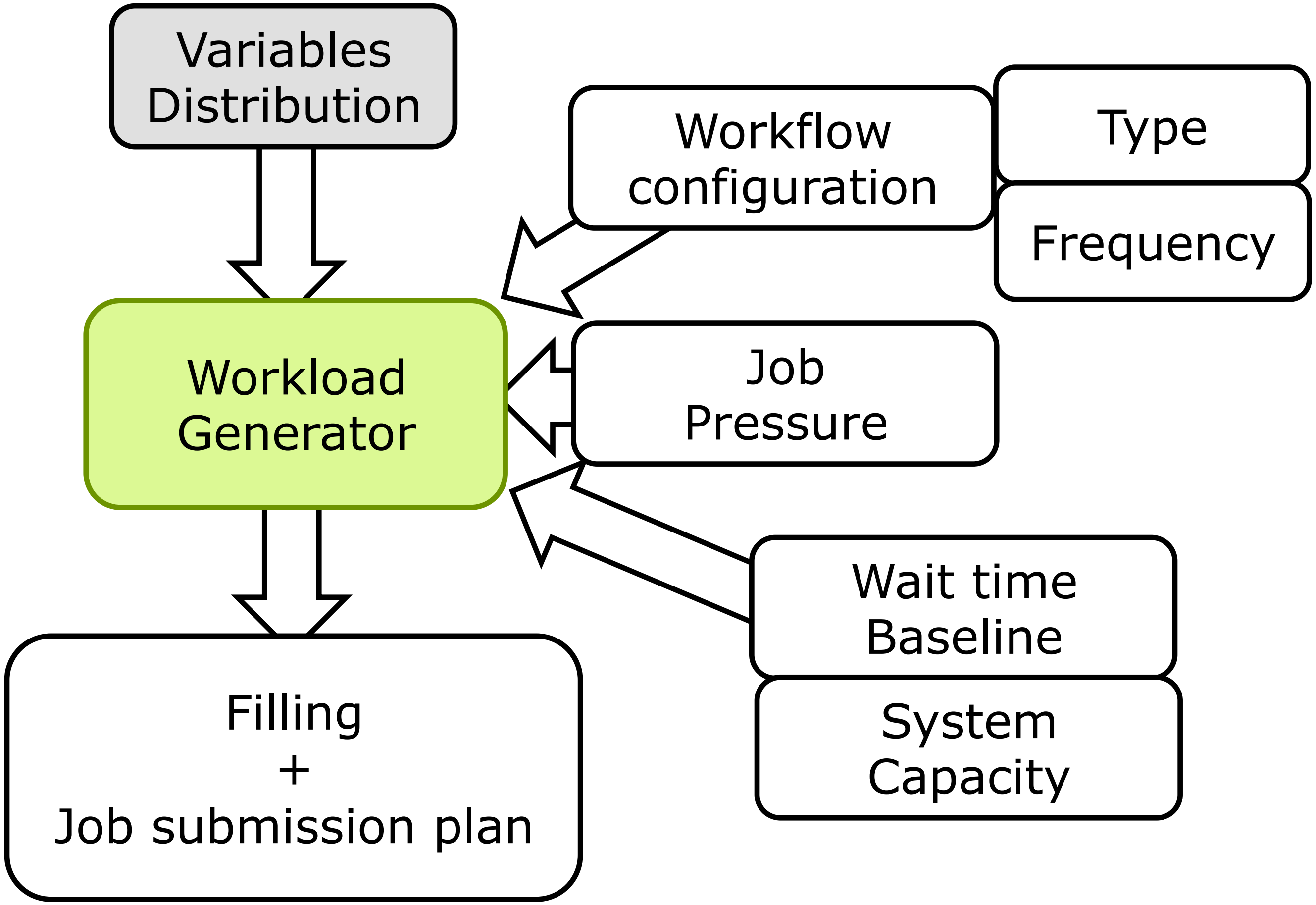


Gonzalo P. Rodrigo – gonzalo@cs.umu.se

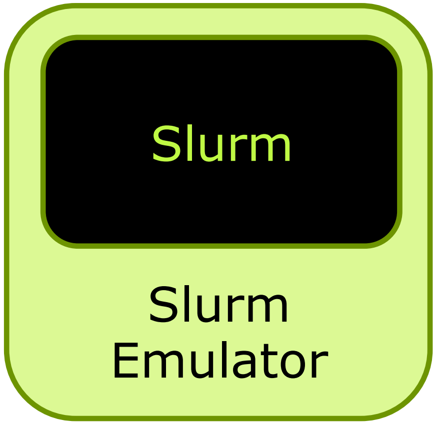
Workload analyzer



Workload generator



Slurm Simulator



Based on BSC original work

Based on CSCS work

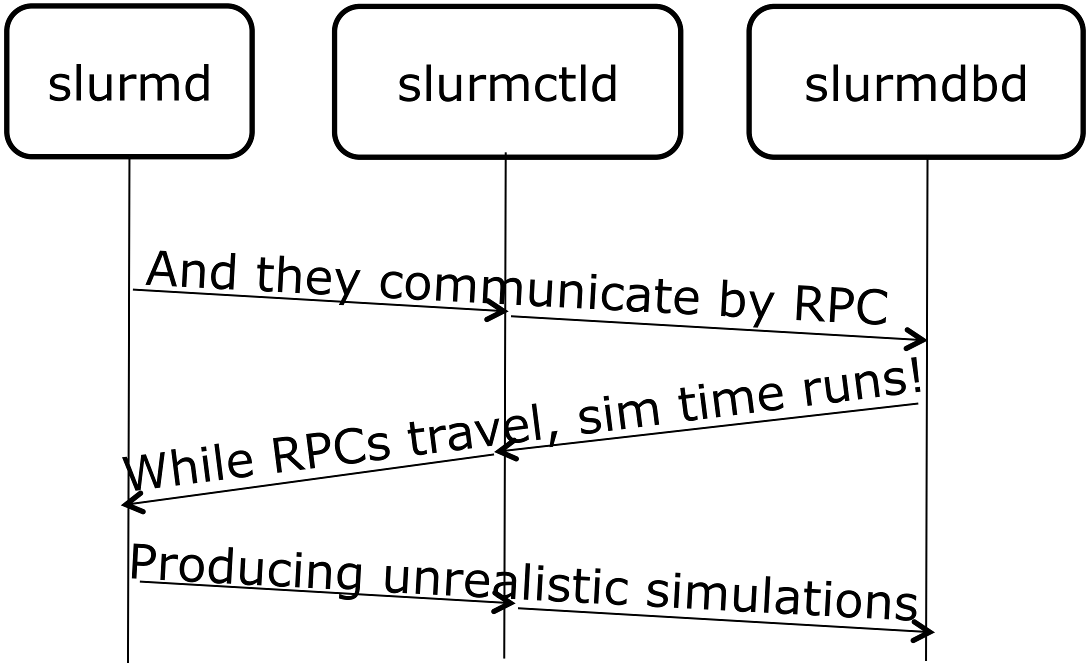
Still, it required quite some work!

Functions

- Emulate Hardware
- Time speed-up: By hacking time/sleep calls
- Job submission from submission plan

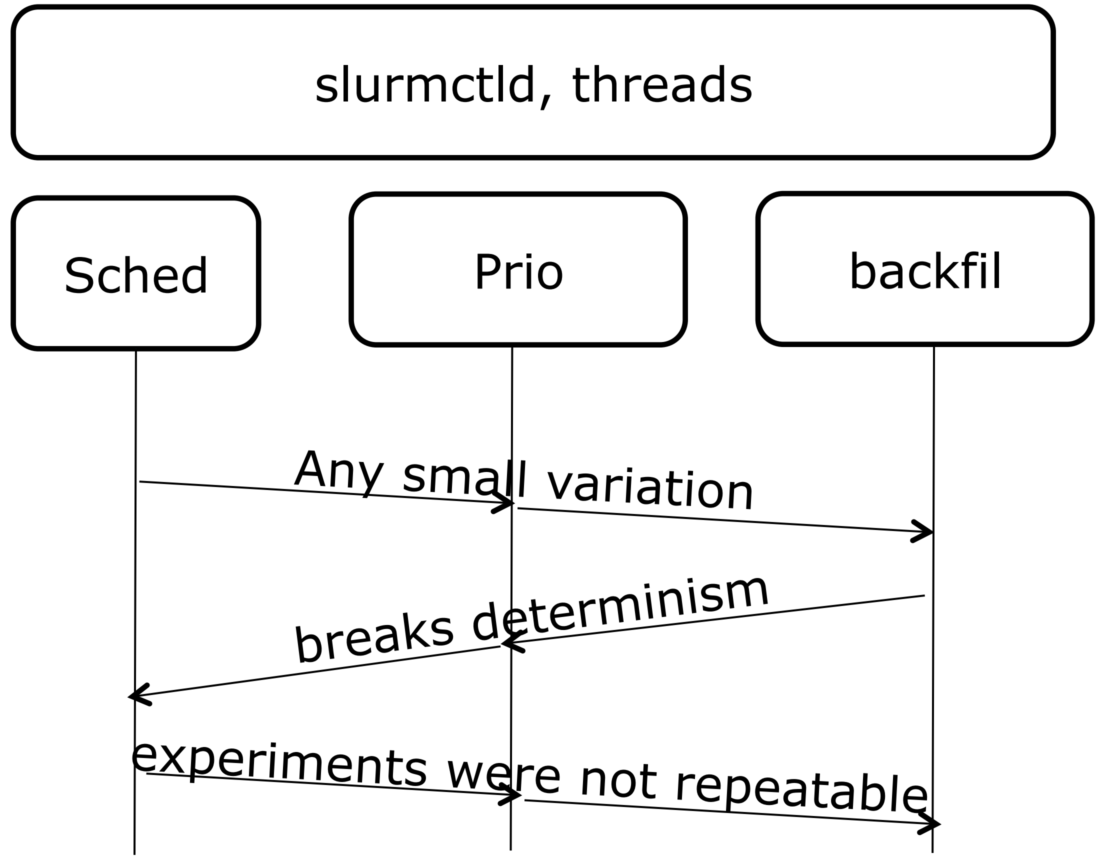
Slurm Simulator: The challenges (1)

SLURM
is composed by
Daemons



Slurm Simulator: The challenges (2)

SLURM
Daemons are
Super threaded



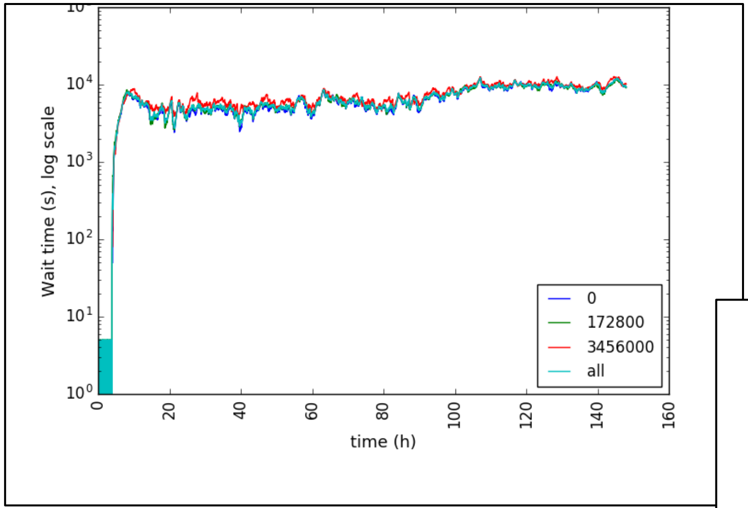
Slurm Simulator: The work

- sim_mgr and scheduling loops RPC synchronized
- Every significant thread is synced with the simulation controller.

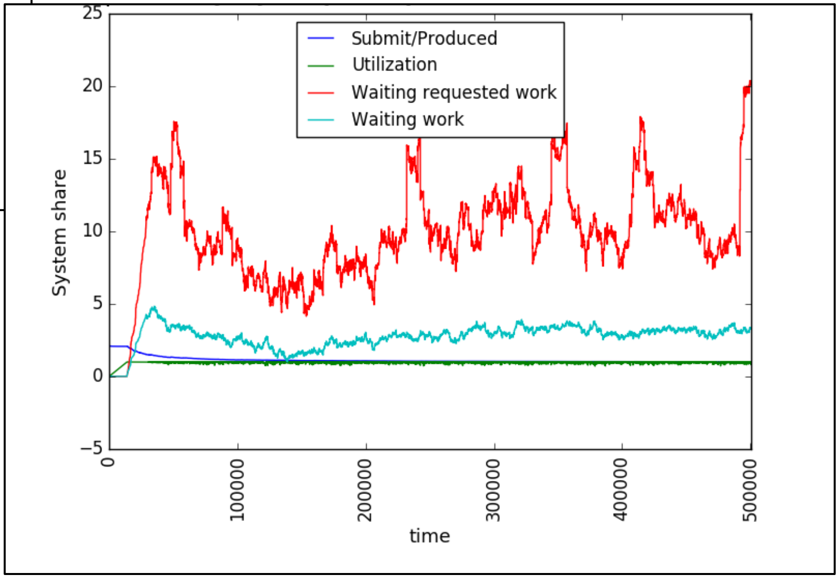
- Real vs simulation time: x10-x20 speedup
- Priority, Scheduling, and Accouting are synced.
- Scheduler can achieve high utilization.
- Results are “semi” deterministic.

Workload analyzer: Evaluating scheduler

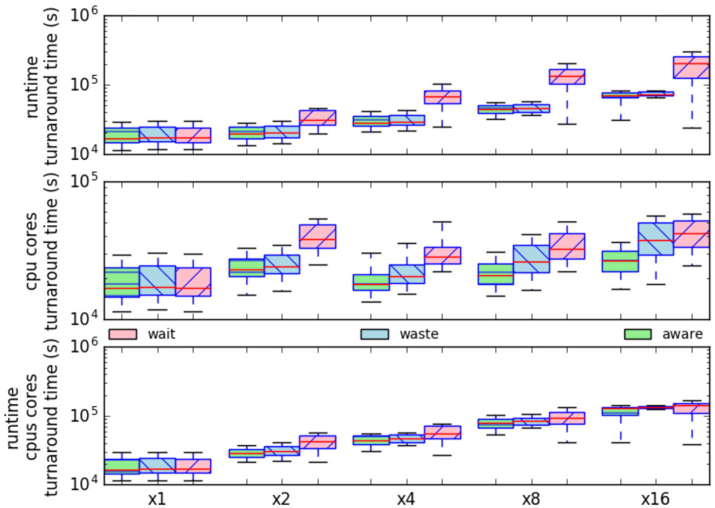
Wait time evolution



Utilization, submitted, pending



Workflows behavior



Gonzalo P. Rodrigo – gonzalo@cs.umu.se

Schedulling tools: Value

- To be fully described in upcoming paper
- Analysis, generation, running, and workflow aware scheduler **will be open sourced**

- Scheduling research: save "engineering hours"
- Admins: capacity to play with configurations of **their own systems and their own workloads.**

Summary of take-aways

Systems and workloads require new scheduling:
We propose two-level cloud inspired model.

In-site workflows are very important:
There is a better way to schedule them.

Good **tools** are fundamental **for good research and operations**.

UMEÅ UNIVERSITET

BERKELEY LAB

Thanks for your time… questions?

Gonzalo P. Rodrigo – gonzalo@cs.umu.se

To know more….

Contact:

gonzalo@cs.umu.se - gprodrigoalvarez@lbl.gov

Rodrigo Álvarez, G. P., Östberg, P. O., Elmroth, E., Antypas, K., Gerber, R., & Ramakrishnan, L. Towards Understanding Job Heterogeneity in HPC: A NERSC Case Study. CCGrid 2016 - The 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2016.

Rodrigo Álvarez, G. P., Östberg, P. O., Elmroth, E., Antypas, K., Gerber, R., & Ramakrishnan, L. (2015, June). HPC System Lifetime Story: Workload Characterization and Evolutionary Analyses on NERSC Systems. In Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing (pp. 57-60). ACM.

Rodrigo Álvarez, G. P., Östberg, P. O., Elmroth, E., & Ramakrishnan, L. (2015, June). A2L2: An Application Aware Flexible HPC Scheduling Model for Low-Latency Allocation. In Proceedings of the 8th International Workshop on Virtualization Technologies in Distributed Computing (pp. 11-19). ACM. Citation

Rodrigo, G. P., Östberg, P-O. & Elmroth, E. (2014).Priority Operators for Fairshare Scheduling. 18th Workshops on Job Scheduling Strategies for Parallel Processing (JSSPP 2014) hosted at the IPDPS-2014 conference.

Rodrigo, G. P. Establishing the equivalence between operators: theorem to establish a sufficient condition for two operators to produce the same ordering in a Faishare prioritization system. January 2014.

Rodrigo, G. P. Proof of compliance for the relative operator on the proportional distribution of unused share in an ordering fairshare system. January 2014.