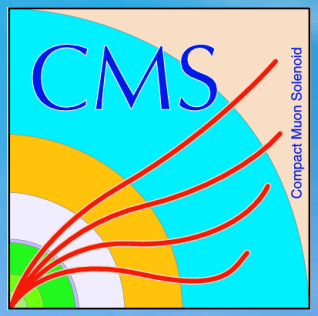


Evolution of CMS workload management towards multicore job support

A. Pérez-Calero Yzquierdo,
J. Hernandez, F. Khan, K. Larson, J. Letts,
A. Malta, A. McCrea, E. Vaandering

for CMS Computing

OIST, Okinawa, 13th April, 2015

Ciemat
Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas

PIC
port d'informació científica

Outline

- Multicore jobs for LHC run 2
- CMS workload management and submission infrastructure
- Results from latest tests
- Monitoring deployment
- Conclusions & Outlook for future developments

Multicore jobs for LHC run 2

- **Motivation for multithreaded applications:**
  - **Hardware evolution:** best exploitation of current **multicore CPUs**
  - **Evolution of LHC conditions**: increased **data volumes** and **event complexity**

- New era for HEP computing with the integration of elements of Grid Computing and High Performance Computing (***distributed parallel computing)*** which requires to adapt different levels of our computing:
  - Multithreaded applications
  - **Grid-wide scheduling**
  - Site scheduling

- **CMS priority in 2015 is Tier1 sites**: Prompt data reconstruction will be run using **T0+50% of T1 CPUs**
  - Drives first phase of deployment, focused on T1s
  - Simulation and digitisation will follow, with deployment to T2s

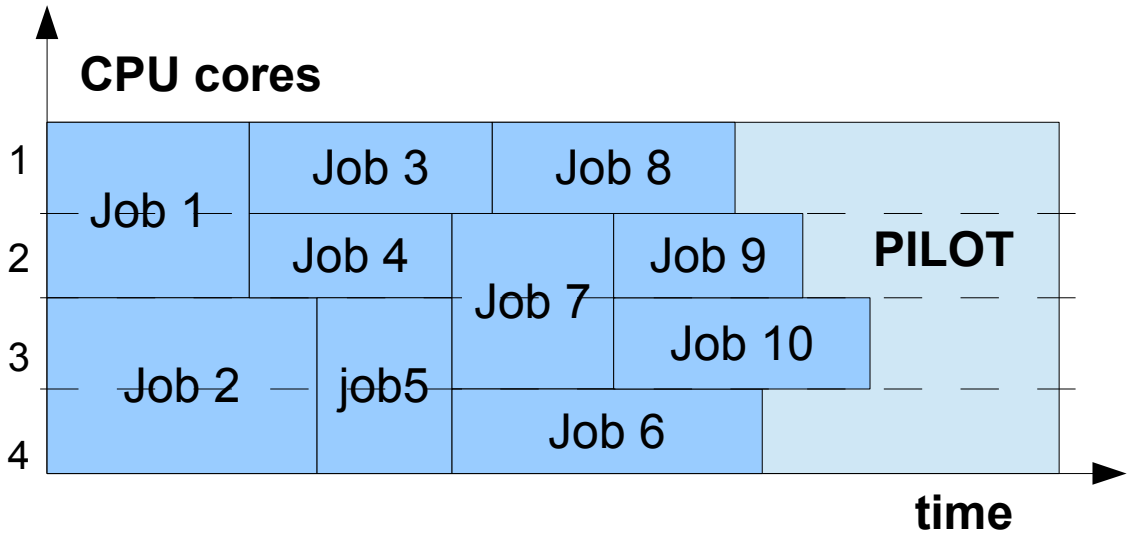
- **Single core and multicore jobs will coexist during run 2: mixed scheduling is mandatory**

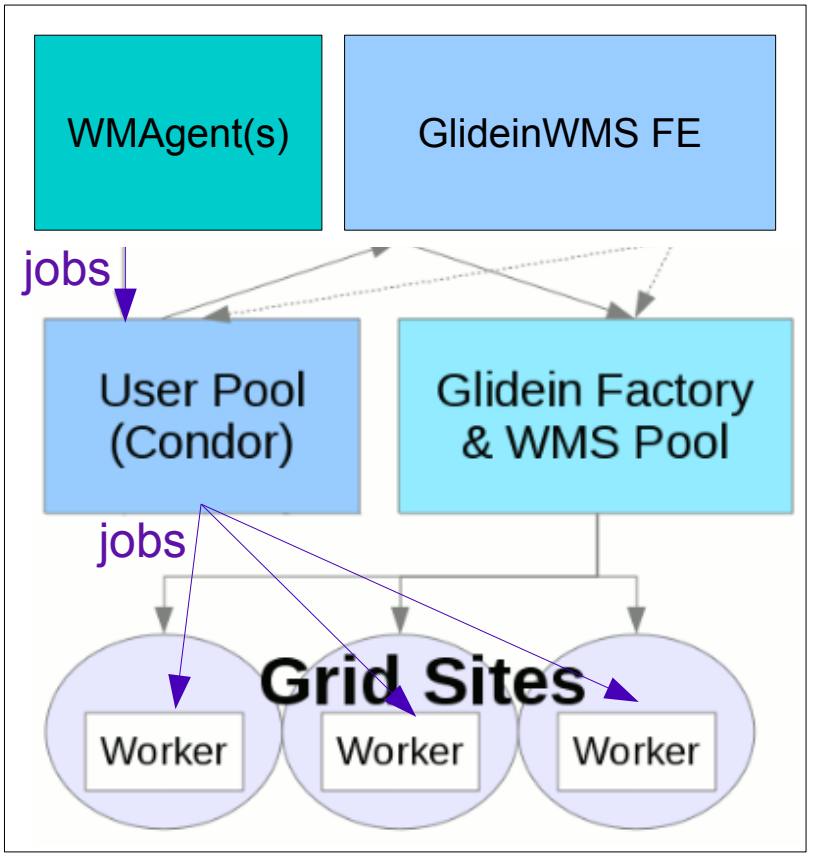
See talk "**Using the CMS Threaded Application in a Production Environment**" for details on CMS multithreaded applications

CMS WM and SI

- CMS workload management and submission infrastructure is based on:
  - **WMAgents**: **manage centralized workflows** populating job queues, assigning job priorities, handling errors and job retrials, etc.
  - **GlideinWMS: matches jobs to resources** managing a transient pool of computing resources controlled by **pilot jobs**

- **Main tool: multicore pilots** with internal dynamic partitioning of resources





See talk "**Using the glideinWMS System as a Common Resource Provisioning Layer in CMS**" for more details

Multicore pilot model

Advantages of managing all CMS workflows with multicore partitionable pilots:

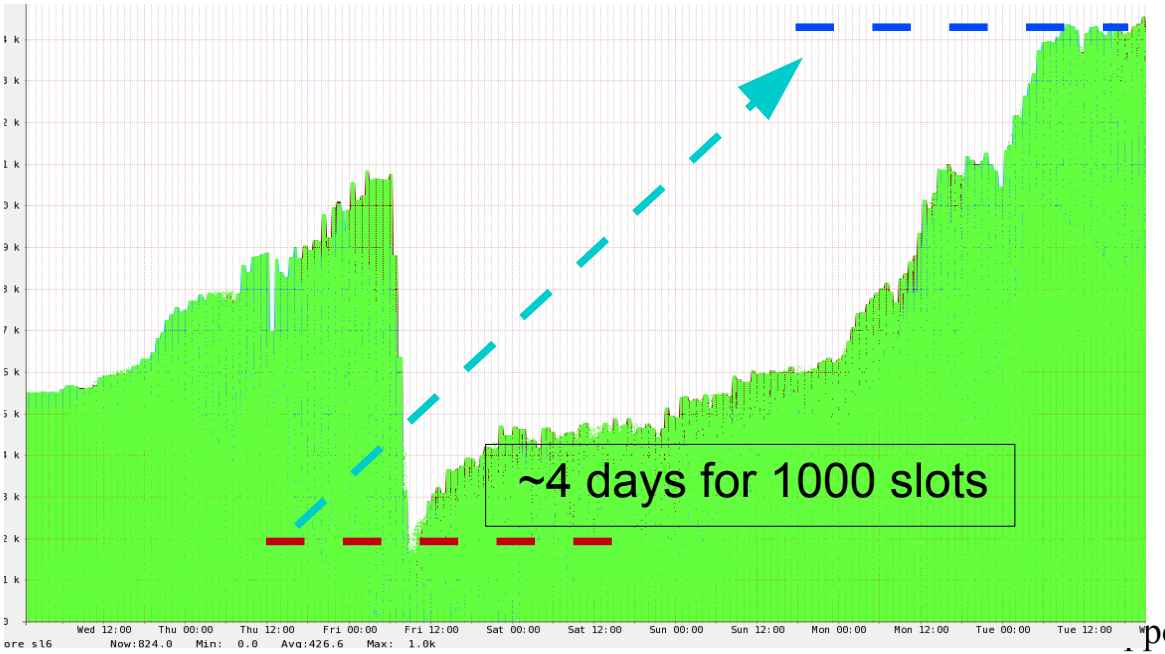
- Total **control of scheduling priorities** of single and multicore jobs
- **Remove unwanted effects** from single core and multicore pilot competition
  - for resources at the sites
  - for matching jobs once running
- **Reduced number of pilots** in the system

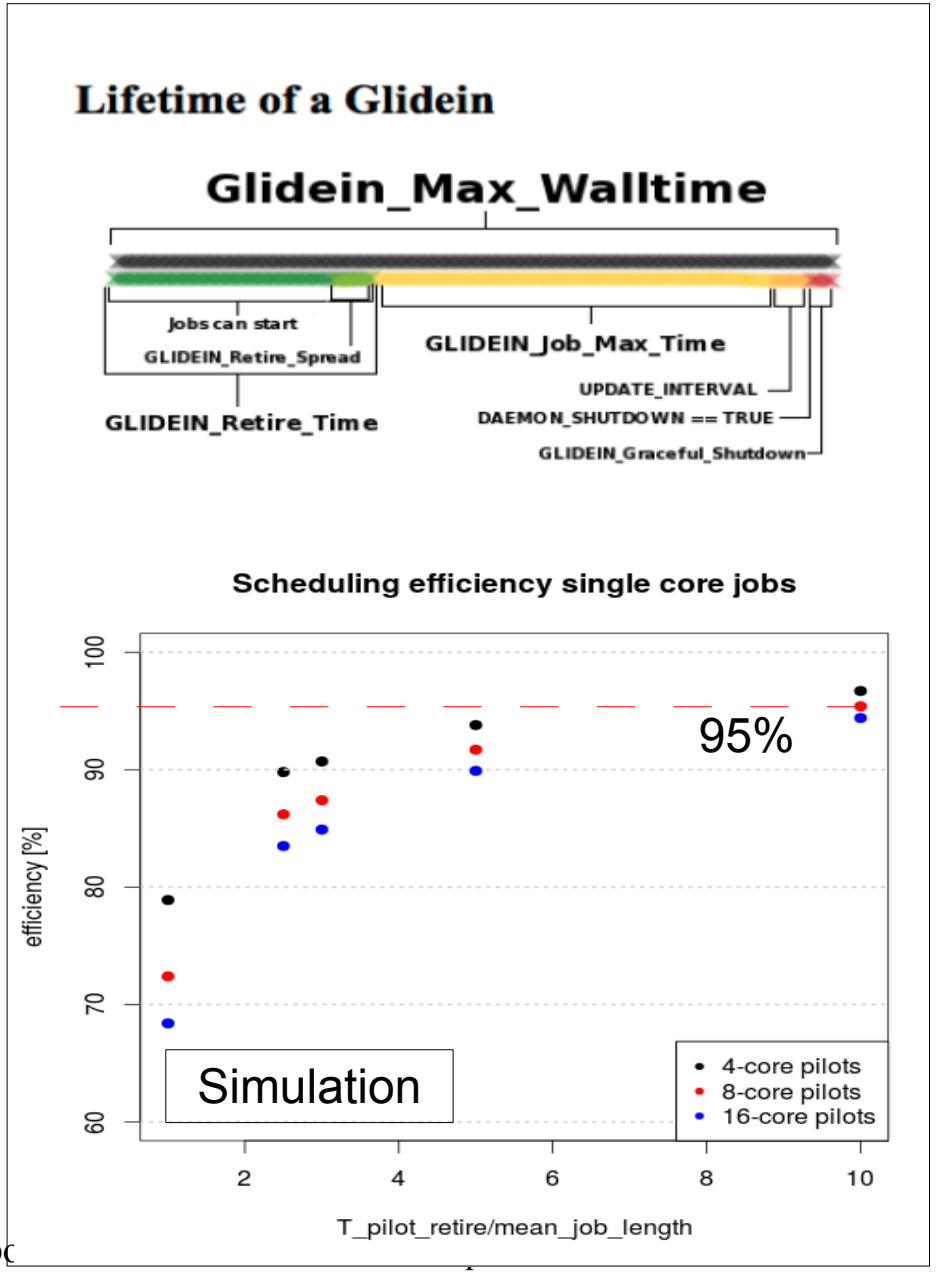
Multicore pilot model

Disadvantages of multicore pilots:

- **Inefficiency** in draining of retiring pilots
- **Slow ramp up** of resources in shared sites who protect their farms from excessive WN draining

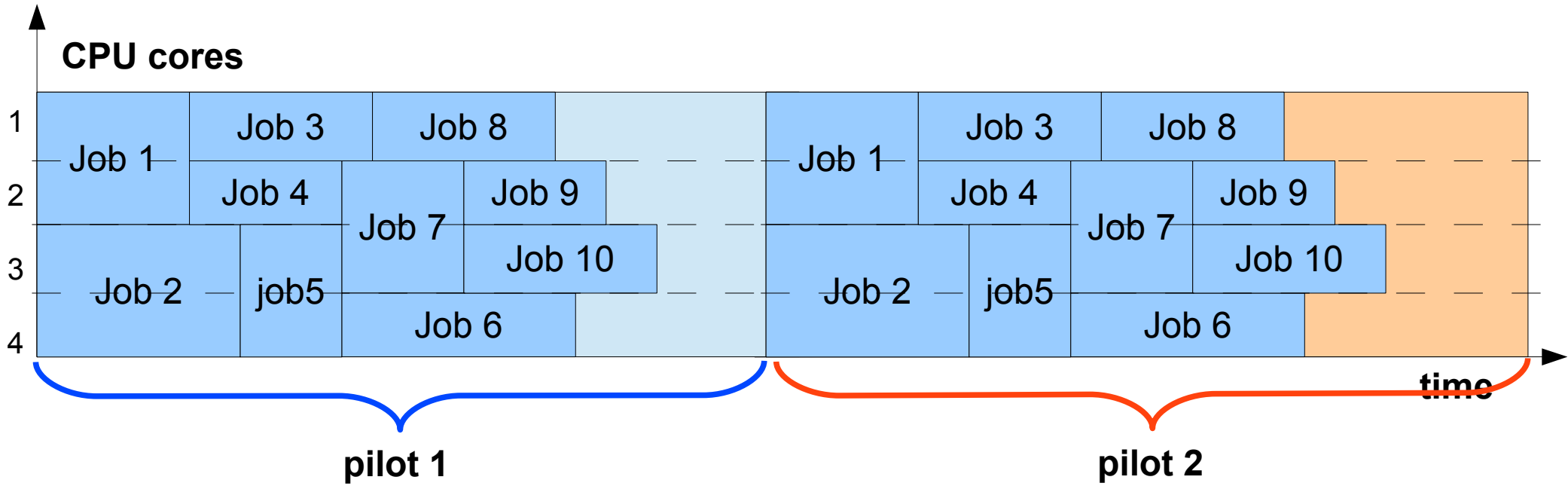
Slots for 8-core pilots available at PIC over a week





Multicore pilot model

- **Fragmentation of the pilot internal resources** limits pilot ability to pull new multicore jobs.
- Renewal of finite-lifetime pilots provides fresh **non-fragmented pilots continuously**
  - **No forced defragmentation** inside the pilots needed
- **Tuning of the system is essential**: performance dependence on ratio of job and pilot running times, mixture of single and multicore jobs, number of cores, job ranks, etc.



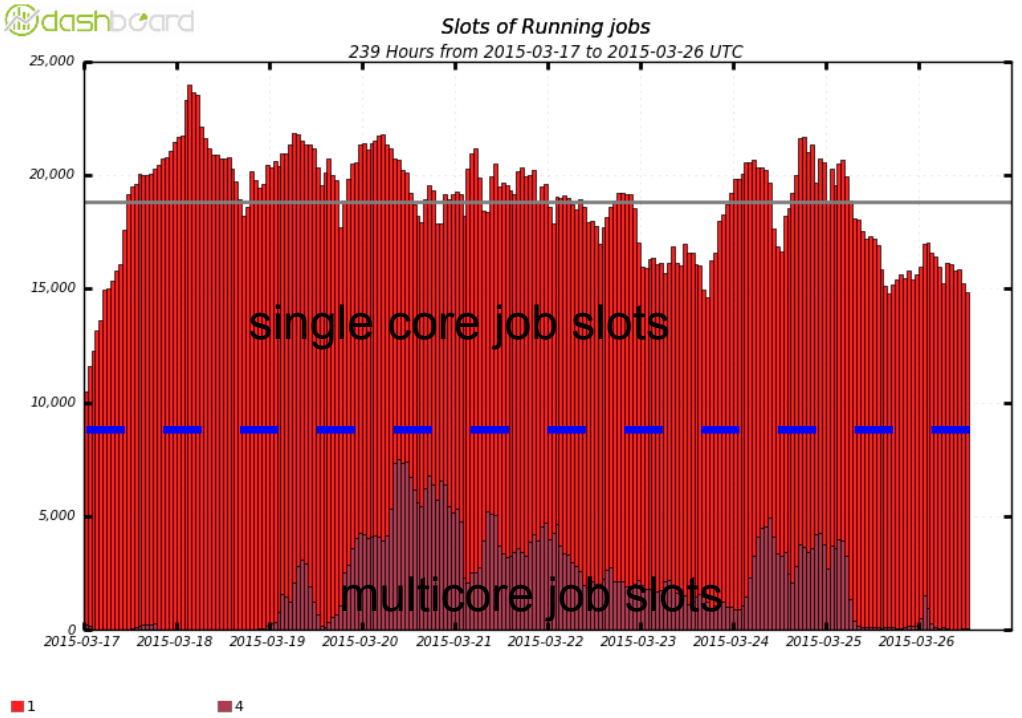
Multicore job scheduling overview

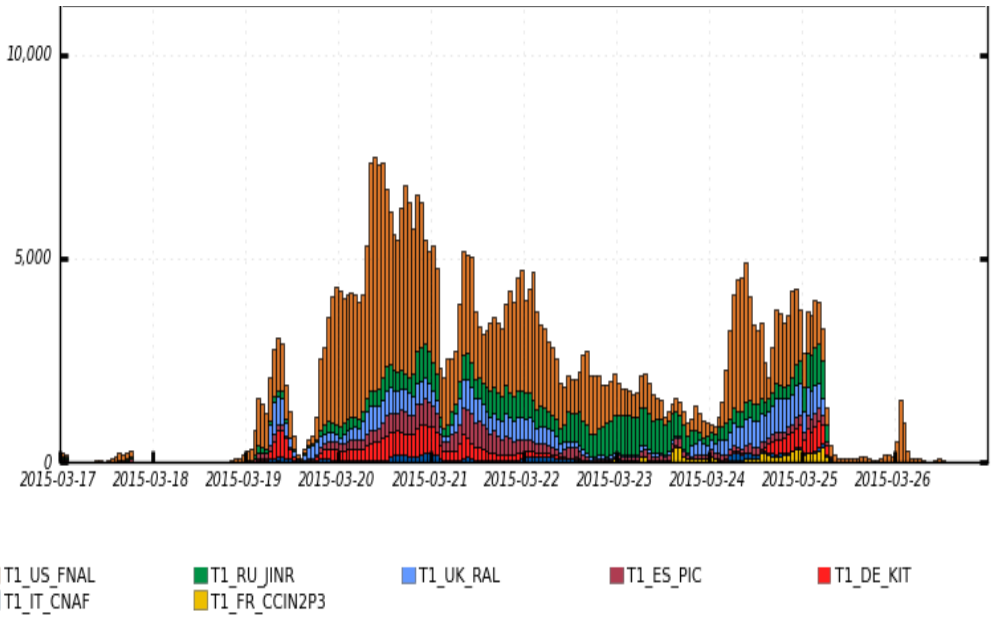
- **CMS target for 2015:** 50% of the T1 resources in use by multicore jobs for PromptReco
- **Objectives:**
  - Integrate scheduling of both **multicore and single-core jobs: control fragmentation of resources in use**
  - High efficiency CPU usage, **minimizing any inefficiencies** deriving from scheduling
- **Components** in WM and SI which required **modifications and tuning**
  - **WMAgent**, managing mixed single core and multicore loads with correct job prioritization
  - Pilot allocation to the sites:
    - **GlideinWMS pilot factories** produce enough multicore pilots for each site
    - Local resource managers (**T1s batch systems**)
  - **GlideinWMS FE**: pilot performance in scheduling mixed loads inside the running pilots
- Tuning and debugging the system requires **fully developed monitoring:**
  - **Pilots**
  - **Jobs**

Latest results

Scale test to Tier1s

- **Multicore job submission works: successfully submitting and running multicore jobs to all CMS T1s**
- **PromptReco multicore jobs** regularly submitted to T1s to test scale:
  - Targetting **18k cores at KIT, PIC, CCIN2P3, CNAF, JINR, RAL and FNAL**
  - Using **4-core jobs** inside **8-core pilots**
  - Tested under **heavy pressure from single core jobs**
- **Good results, close to target overall**
  - **Peaks of 8k cores achieved**
  - **Results vary from site to site**: work ongoing to reach target at each site independently

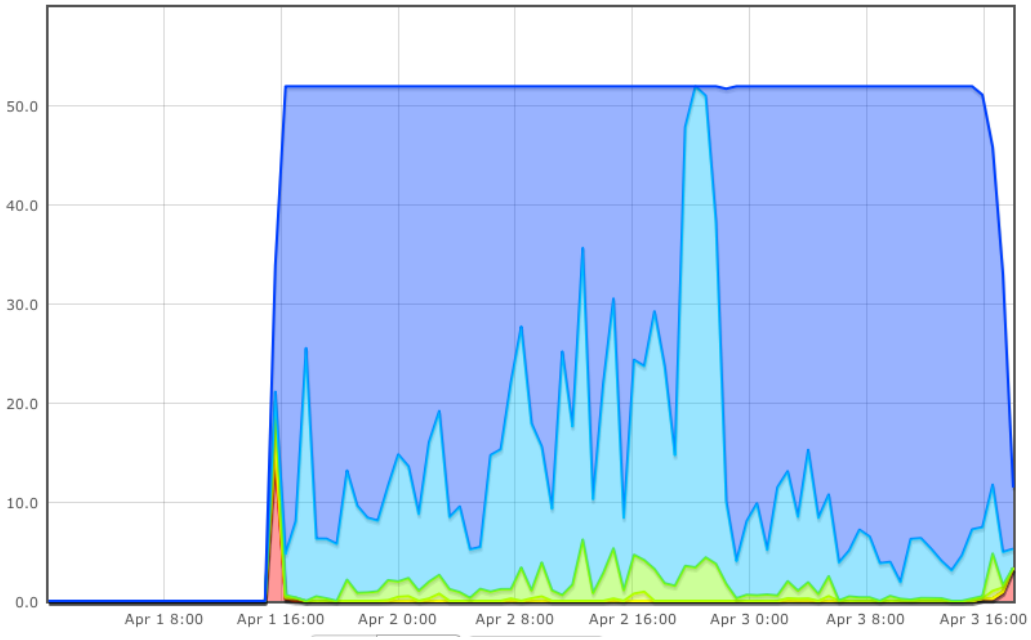




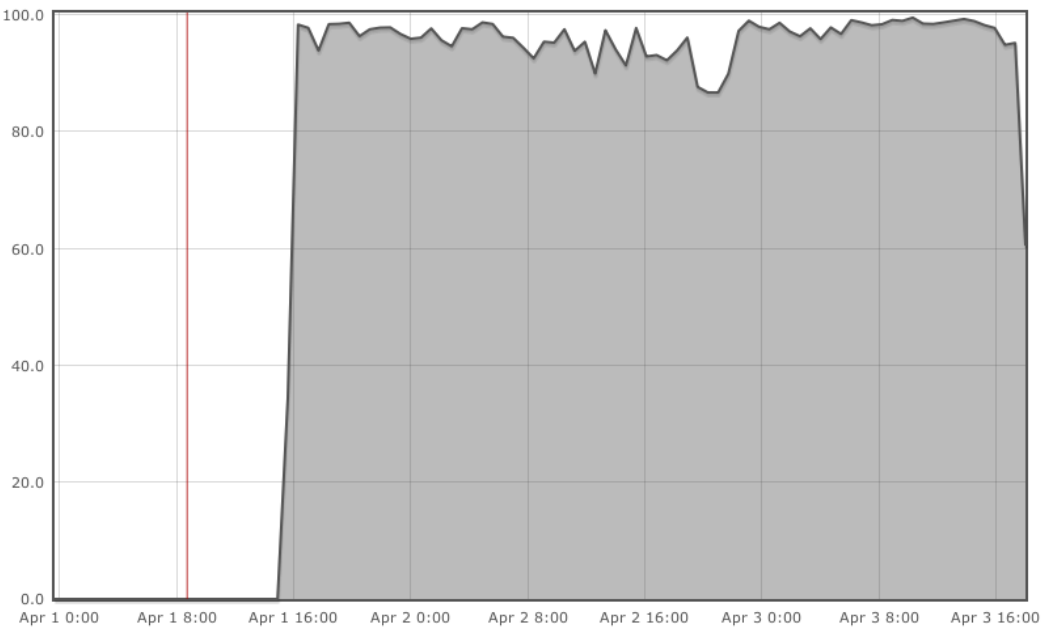
Multicore pilot scheduling efficiency

- **Multicore pilots** running filled with **single core jobs**
  - 40h long pilots
  - 1-2h long jobs
- Results: with sufficient job pressure, **pilot internal inefficiencies are negligible**

Slots occupancy by pilots



Overall slots occupancy (%)

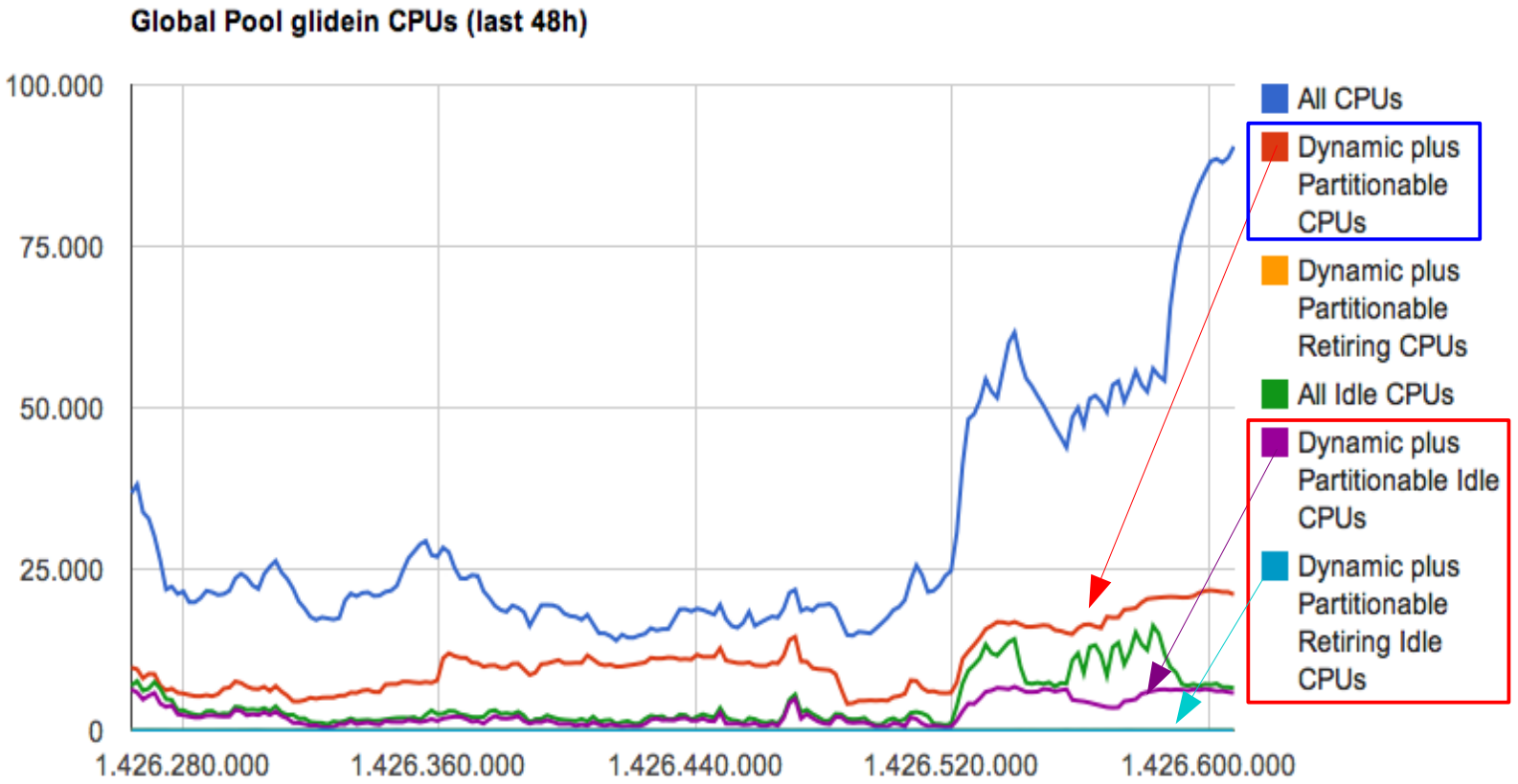


Monitoring deployment

Global pilot monitoring

Central glideinWMS monitoring:

- Pilot internal scheduling efficiency: **fraction of the time pilot resources are spent in running jobs vs. total pilot running time**
  - Prototype of global pool view of **pilots running and idle weighted by Ncores**
- **Also developing monitors for multicore pilot pressure at each site**



Conclusions & Outlook

Conclusions & Outlook

In summary:

- CMS will use **multithreaded applications for LHC run 2**
  - **Prompt data reconstruction in 2015**
- **CMS tools for multicore job management are ready**
- **Target for 2015** multicore resources (**50% of T1 CPUs**) **already achieved**
- **Ready by the restart of data taking!**

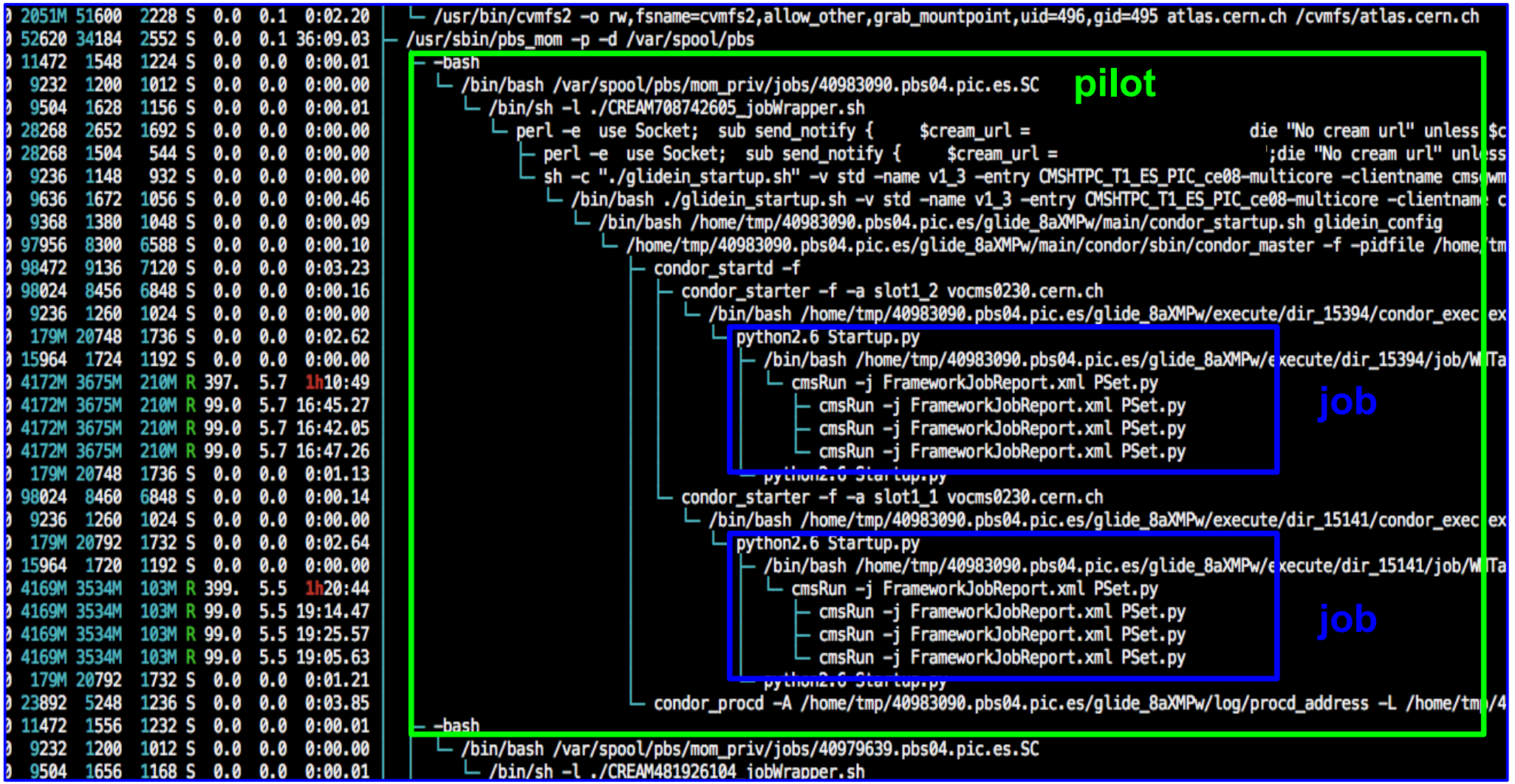
Milestones for the coming months

- Complete deployment of monitoring tools
- Continue tests for scheduling optimization
- Optimize site by site results

Extra slides

CMS multithreaded jobs running at T1s

- Example of 4-core jobs running inside 8-core pilot (PIC)

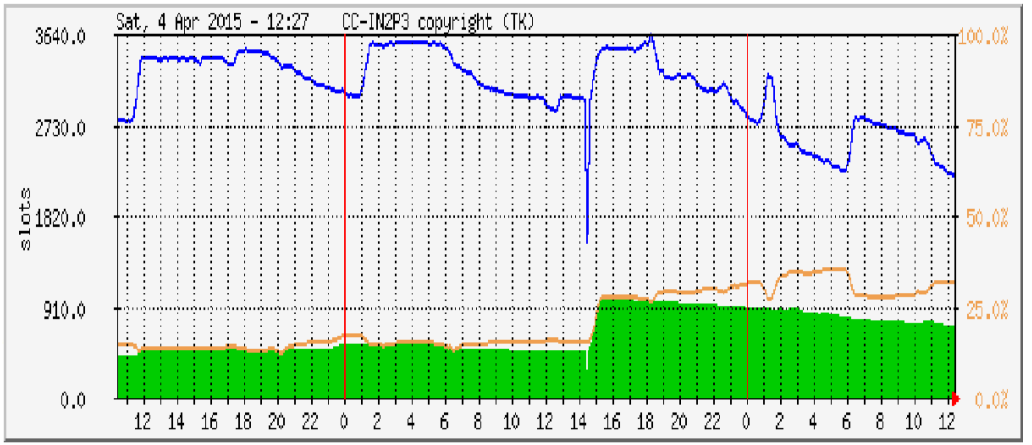


Additional pilot monitoring at T1 sites

- Cores in use by CMS by type of pilot running
  - Single core and multicore
  - Production and analysis

PIC

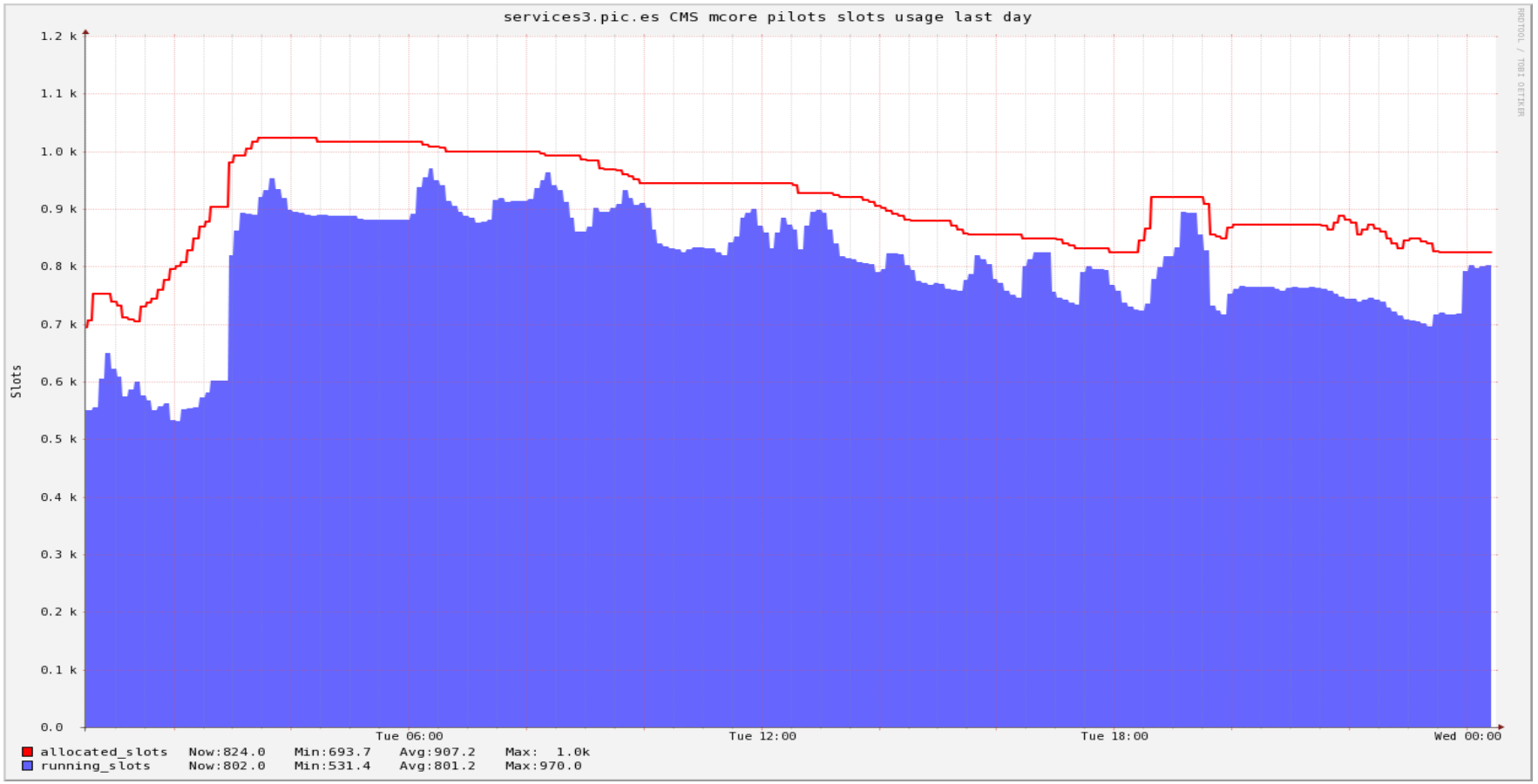
CCIN2P3



	Max	Average	Current
cmsf_T1prod_mc project used slots:	968	645	712
cmsf group used slots:	3623	3075	2218
percentage:	35.0 %	21.0 %	32.0 %

Additional pilot monitoring at T1 sites

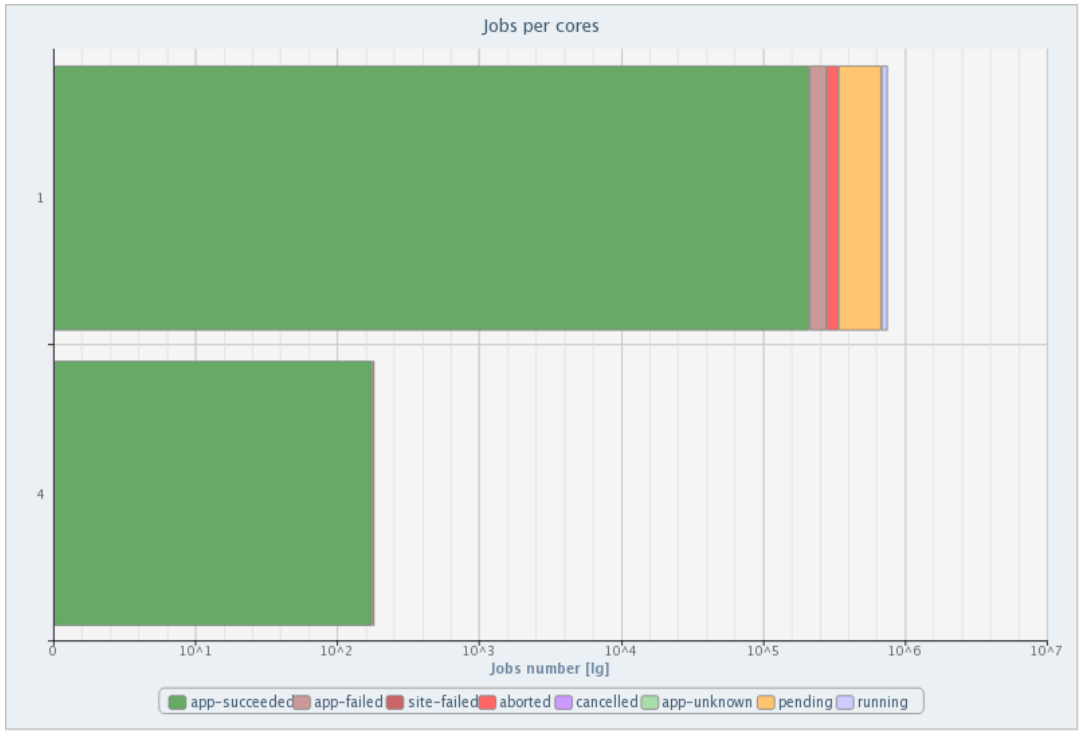
- Being developed at T1s in addition to central monitors
  - Example: multicore pilot occupancy monitors at PIC
    - number of allocated cores vs. cores in use by jobs



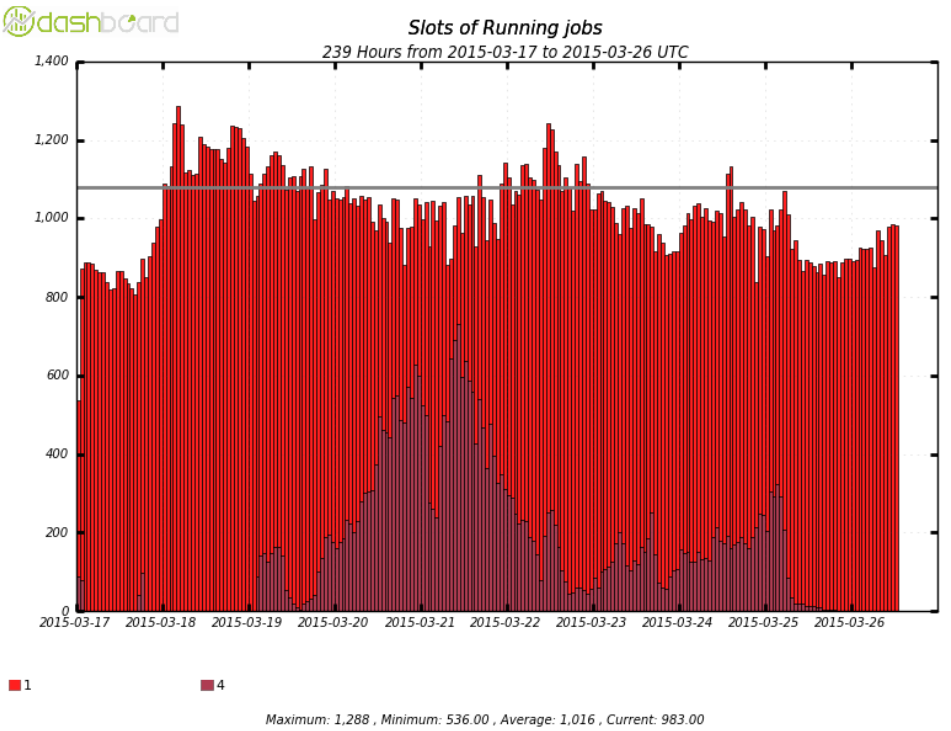
Job monitoring in dashboard

- **CMS job dashboard is getting ready for multicore:**
  - Use Ncores to **classify single core and multicore jobs**
  - Job weight to display **correct resource utilization of sites**
  - **Scaled walltimes with Ncores**, so redefined CPU efficiency stays in [0,1]

Interactive view



Historical view



Related contributions at CHEP'15

CMS:

- Using the CMS Threaded Application in a Production Environment
- Using the glideinWMS System as a Common Resource Provisioning Layer in CMS
- The CMS Tier-0 goes Cloud and Grid for LHC Run 2

Non CMS:

- Multicore job scheduling in the Worldwide LHC Computing Grid
- Scheduling multicore workload on shared multipurpose clusters

Other references

- A new era for central processing and production in CMS, E Fajardo *et al* 2012 *J. Phys.: Conf. Ser.* **396** 042018, http://dx.doi.org/10.1088/1742-6596/396/4/042018
- GlideinWMS Homepage: http://www.uscms.org/SoftwareComputing/Grid/WMS/glideinWMS/doc.prd/index.html
- HTCondor Homepage: http://research.cs.wisc.edu/htcondor/
- The pilot way to Grid resources using glideinWMS, I. Sfiligoi et al. http://dx.doi.org/10.1109/CSIE.2009.950
- WLCG Multicore Task Force: https://twiki.cern.ch/twiki/bin/view/LCG/DeployMultiCore

Abstract

The successful exploitation of the multicore processor architectures available at the computing sites is a key element of the LHC distributed computing system in the coming era of the LHC Run 2. High-pileup complex-collision events represent a challenge for the traditional sequential programming in terms of memory and processing time budget. The CMS data production and processing framework has introduced the parallel execution of the reconstruction and simulation algorithms to overcome these limitations.

CMS plans to execute the data reconstruction and simulation as multicore processing yet supporting single-core processing for other tasks difficult to parallelize, such as user analysis. The CMS strategy for job management across the Grid thus aims at integrating single and multicore job scheduling. This is accomplished by scheduling multicore pilots with dynamic partitioning of the allocated resources, capable of running jobs with various core counts within a single pilot.

An extensive test programme has been conducted to enable multicore scheduling with the various local batch systems available at CMS sites. Scale tests have been run to optimize the scheduling strategy and to ensure the most efficient use of the distributed resources. This contribution will present in detail the evolution of the CMS job management and resource provisioning systems in order to support this hybrid scheduling model, as well as its optimization and deployment, which will enable CMS to transition to a multicore production model by the restart of the LHC.