GridPP

UK Computing for Particle Physics

Batch System Status at the RAL Tier-1

Andrew Lahiff, Alastair Dewhurst, John Kelly, Ian Collier

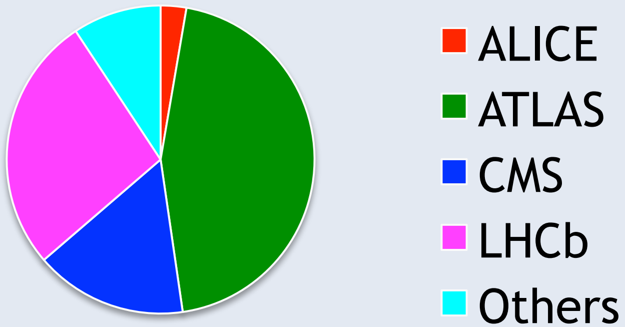
29 Oct 2013, HEPiX Fall 2013 Workshop

Outline

- RAL batch system
  - Background
  - Issues
- Choosing a new batch system
  - Criteria
  - Testing
- Compatibility with middleware
- Testing with VOs
- New batch system configuration & monitoring
- Migration to the new batch system

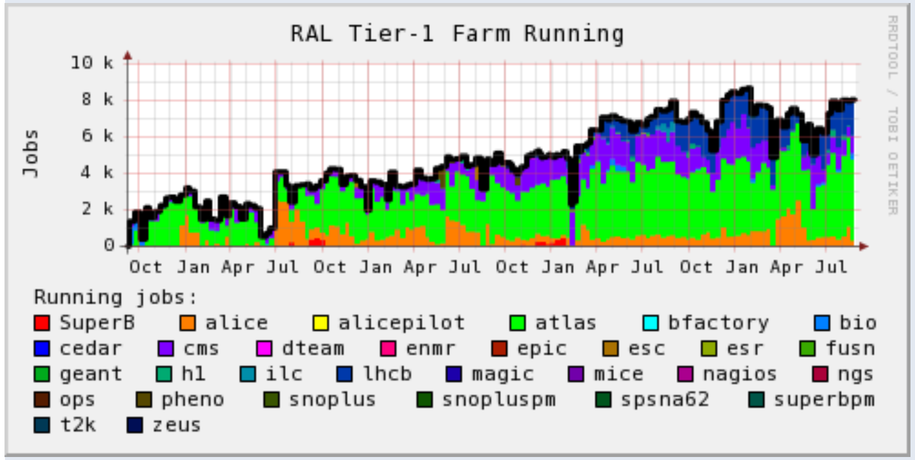
RAL batch system

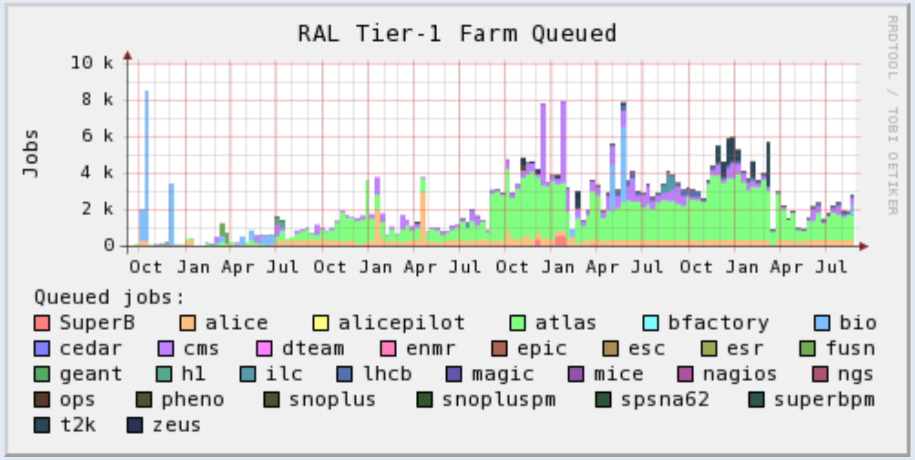
- Batch system at the RAL Tier-1
  - 656 worker nodes, 9312 slots, 93000 HEPSPEC06
- VOs supported
  - All LHC experiments. RAL provides:
    - 2% of ALICE T1 requirements
    - 13% of ATLAS T1 requirements
    - 8% of CMS T1 requirements
    - 19% of LHCb T1 requirements
  - Many non-LHC experiments, including non-HEP
- Allocations



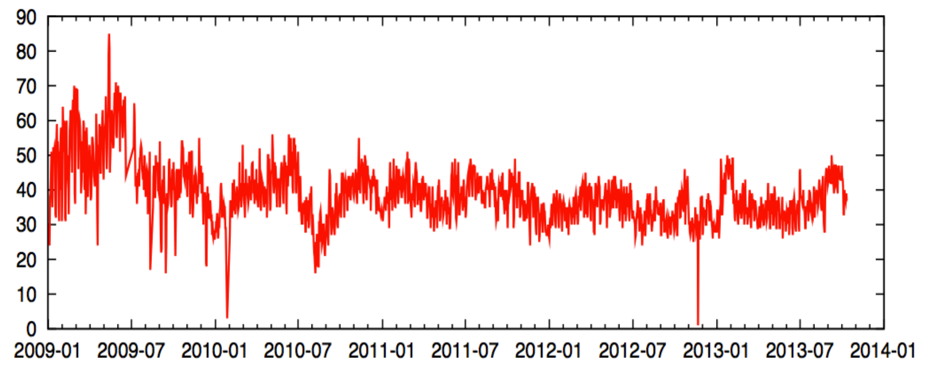
RAL batch system

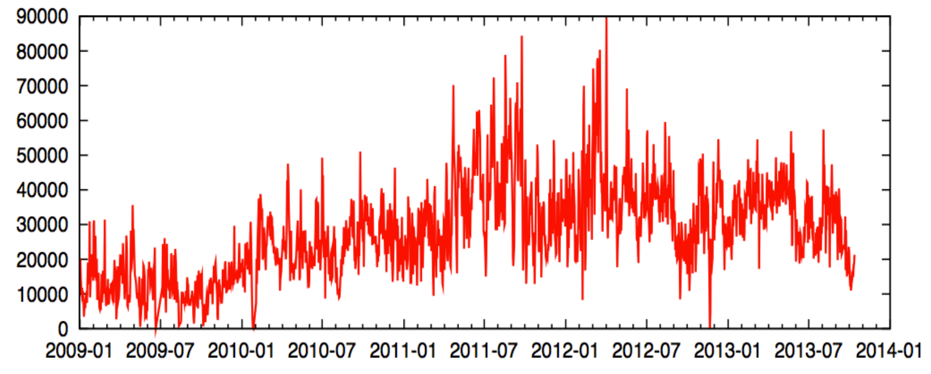
- Jobs running & queued over past 4 years





- Distinct users per day, jobs completed per day





Issues

- Torque/Maui have been used for many years at RAL
  - Currently Torque 2.5.12, Maui 3.3.1
- Many issues with Torque/Maui
  - pbs_server, maui sometimes unresponsive
  - pbs_server needs to be restarted sometimes due to excessive memory usage
  - Job start rate sometimes not high enough to keep the farm full
  - Regular job submission failures on CEs - *Connection timed out-qsub: cannot connect to server*
  - Unable to schedule jobs to the whole-node queue
    - We wrote our own simple scheduler for this, running in parallel to Maui
  - Didn’t handle mixed farm with SL5 and SL6 nodes well
  - DNS issues, network issues & problematic worker nodes cause it to become very unhappy
- Significant effort just to keep it working

Choosing a new batch system

- In August 2012 started looking for an alternative
- Initially proposed the following technologies as candidates
  - Torque 4 + Maui
  - LSF
  - Grid Engine
  - SLURM
  - HTCondor

Choosing a new batch system

- Criteria
  - Integration with WLCG community
    - Compatible with grid middleware
    - APEL accounting
  - Integration with our environment
    - e.g. does it require a shared filesystem
  - Scalability
    - Number of worker nodes
    - Number of cores
    - Number of jobs per day
    - Number of running, pending jobs
  - Robustness
    - Effect of problematic worker nodes on batch server
    - Effect if batch server is down temporarily
    - Effect of other problems (e.g. network issues)

Choosing a new batch system

- Criteria (cont'd)
  - Software support
  - Procurement cost
    - Licenses, support
    - Avoid commercial products unless all open source products unsuitable
  - Maintenance cost
    - FTE required to keep it running
  - Essential functionality
    - Hierarchical fairshares
    - Ability to limit resources (CPU time, wall time, memory, …)
    - Ability to schedule whole-node/multi-core jobs effectively
    - Ability to place limits on numbers of running jobs for particular users, groups or VOs
  - Desirable functionality
    - High availability
    - Ability to handle dynamic resources
    - Power management
    - IPv6 compatibility

Choosing a new batch system

- Some products were quickly rejected
  - Requirement: avoid all commercial solutions unless all open source products are found to be unsuitable
  - Therefore rejected
    - LSF
    - Univa Grid Engine
    - Oracle Grid Engine
  - Also rejected the open source Grid Engines (Son of Grid Engine, Open Grid Scheduler)
    - Competing products, not clear which has best long-term future
    - Neither seems to have communities as active as SLURM & HTCondor
- Note we did do some minimal testing with LSF and Son of Grid Engine
  - E.g. to see how easy to install & configure, setting up fairshares, …

Choosing a new batch system

- Also rejected
  - Torque 4 + Maui
    - Still need to use Maui (Maui causes us problems in the current batch system)
    - Testing with high job submission rates / query rates revealed problems
      - Success rate:

	Job submission	Job status
Torque 2.5.12	10%	20%
Torque 4.x	>90%	>90%
Grid Engine	100%	100%
HTCondor	100%	100%
LSF	100%	100%
SLURM	100%	100%

Choosing a new batch system

- Left with 2 choices

SLURM vs HTCondor

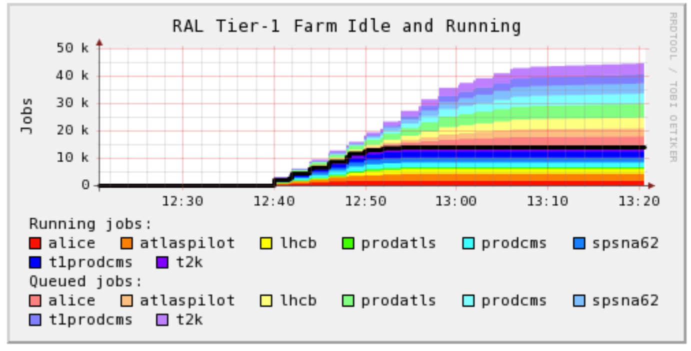
- **Critical test:** *can the batch system successfully maintain 10000 running jobs?*
  - No point migrating to a batch system which fails this test
- Testing
  - 110 old worker nodes (8 cores, 16 GB), using 16, 64, 100 job slots per node
  - Sleep jobs with random durations submitted from a variety of different users
- Setup

  Enabled features which would be required in a production service
  - HTCondor
    - Single central manager (collector, negotiator), schedd on another host
    - Hierarchical fairshares
    - Partitionable slots
  - SLURM
    - Consumable resource allocation plugin
    - Multi-factor job priority plugin
    - Backfill scheduler
    - Accounting (external MySQL database)

GridPP
UK Computing for Particle Physics

SLURM vs HTCondor

- HTCondor
  - No problems running > 10000 jobs
  - No problems with > 200000 pending jobs



- SLURM
  - Stability problems experienced when running > ~6000 jobs
    - Everything fine when no jobs are completing or new jobs starting (!)
  - Queries (sinfo, squeue, …) and job submission failed:

    Socket timed out on send/recv operation
  - Using FIFO scheduling helped
    - Cannot use this in production!
  - Some activities (e.g. restarting SLURM controller) triggered unresponsiveness
    - Took many hours to return to a stable situation

SLURM vs HTCondor

- SLURM
  - Tried a number of things
    - Identical configuration, same version as used at another site which has 5500 slots
    - Tried “large cluster” & “high-throughput” suggestions from documentation
    - Asked other people using SLURM, asked on the mailing list
  - Despite a lot of effort we were unable to solve these problems, therefore rejected SLURM
    - At the time didn’t know of any WLCG sites with more than 5500 slots using or testing SLURM
- Conclusion
  - Chose HTCondor as the prime candidate for replacing Torque/Maui

Compatibility with Middleware

- EMI-3 CREAM CE
  - HTCondor not officially supported
    - BLAH supports HTCondor
      - Job submission works!
    - Script for publishing dynamic information doesn’t exist in EMI-3
      - Wrote our own based on the scripts in old CREAM CEs
    - APEL parser for HTCondor doesn’t exist in EMI-3
      - Wrote our own
  - Relatively straightforward to get an EMI-3 CREAM CE working with HTCondor

Compatibility with Middleware

- Another possibility – EMI-3 ARC CE
  - Successfully being used by some ATLAS & CMS Tier-2s outside of Nordugrid (with SLURM, Grid Engine, …)
    - LRZ-LMU, Estonia Tier 2, Imperial College, Glasgow
  - Benefits of ARC CEs
    - Support HTCondor better than CREAM CEs do
    - Simpler than CREAM CEs (no YAIM, no Tomcat, no MySQL, …)
    - ARC CE accounting publisher (JURA) can send accounting records directly to APEL using SSM. APEL publisher node not required
  - Decided it was worthwhile to try ARC CEs
    - Internal testing initially
    - Moved on to testing with real ATLAS jobs, pilots submitted from the standard pilot factories

Compatibility with Middleware

- Which VOs can use ARC CEs?
  - ATLAS, CMS (both use HTCondor-G to submit pilots)
  - LHCb (recently added to DIRAC the ability to submit to ARC)
  - Non-LHC VOs which use EMI WMS for job submission
- Which VOs can't?
  - ALICE, don't currently have any available effort to work on this
    - ALICE can submit directly to HTCondor, which is something we might consider
- Our configuration of ARC CEs
  - Each CE configured with a single generic queue
  - Using the philosophy: *jobs **must** request the resources they require*. For example
    - CMS jobs request 2.5 GB memory
    - ATLAS jobs request 3 GB or 4 GB memory as required
    - ATLAS multicore jobs request 8 cores, 16 GB memory
    - Jobs which don't specifically request much memory don't get any
  - We think this approach is better than having lots of queues

HTCondor testing with VOs

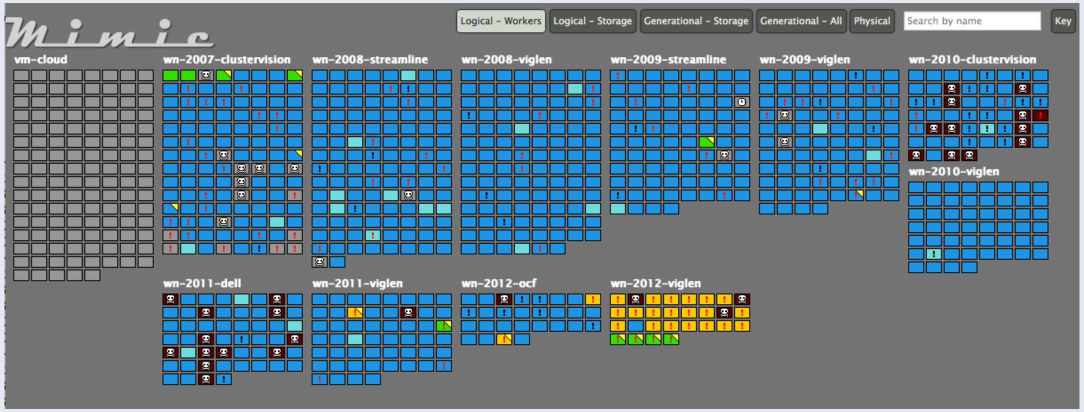
- Next stage of testing with HTCondor
- “Almost” production quality service setup in late May
  - HTCondor 7.8.8 with highly-available central manager (2 nodes)
  - 2 EMI-3 ARC CEs, using LCAS/LCMAPS + Argus
  - 112 8-core EMI-2 SL6 worker nodes
- Testing
  - Evaluation using resources beyond WLCG pledges
  - Aim to gain experience running ‘real’ work
    - Stability, reliability, functionality, dealing with problems, …
  - Initial testing mainly with ATLAS, but also CMS
    - ATLAS: production & analysis SL6 queues
    - CMS: initially testing with integration testbed, then added to production glideinWMS
  - After sorting out initial teething problems, worked very successfully

HTCondor setup

- All configuration managed by Quattor
- Features we're using
  - High-availability of central manager
    - Easy to setup, doesn't require shared filesystem
  - Hierarchical fairshares
  - Partitionable slots
  - condor_defrag daemon
    - Currently not many multicore jobs are submitted
  - Concurrency limits
  - Per-job PID namespaces
  - Python API (for Nagios checks)
- Startd cron
  - Worker node health check script prevents new jobs from starting by some/all VOs as appropriate if problems detected (e.g. disk full or read-only, CVMFS broken, …)
- Currently testing
  - cgroups

Monitoring

- Torque batch system
  - Lots of custom monitoring & accounting scripts written over the years
  - All would need to be modified for HTCondor
  - Only a few so far have been updated for HTCondor, e.g. Mimic
  -

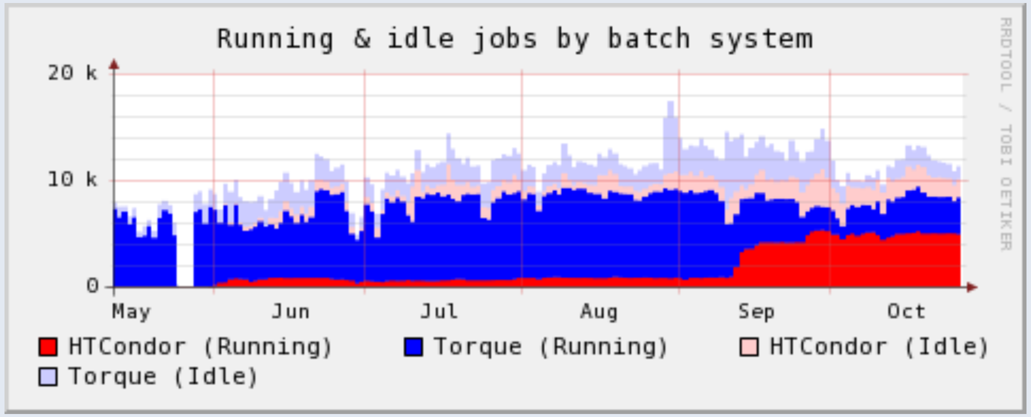


- Mostly trying to use existing tools, e.g.
  - HTCondor Job Overview Monitor (http://sarkar.web.cern.ch/sarkar/doc/condor_jobview.html)
  - condor_gangliad *(since last week)*
  - Gangliarc (ARC CE ganglia monitoring)
  - ARC Grid Monitor

Migration to HTCondor

- Timeline

2012 Aug - Started evaluating alternatives to Torque/Maui
2013 June - Began testing HTCondor with ATLAS & CMS
2013 Aug - Choice of HTCondor approved by RAL Tier-1 management
2013 Sept - Declared HTCondor & ARC CEs production services
- Moved 50% of pledged CPU resources to HTCondor
(upgraded WNs to SL6 as well as migrating to HTCondor)
2013 Nov - Migrate remaining resources to HTCondor

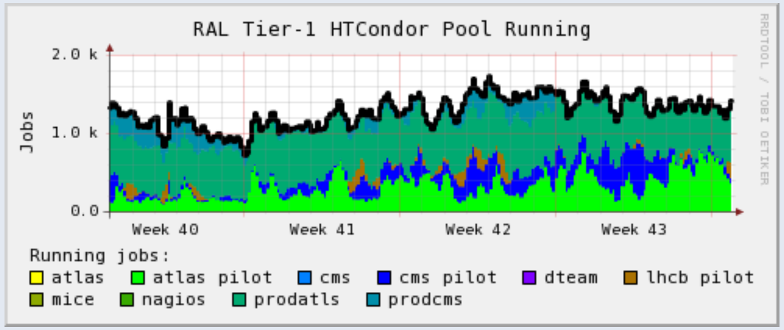


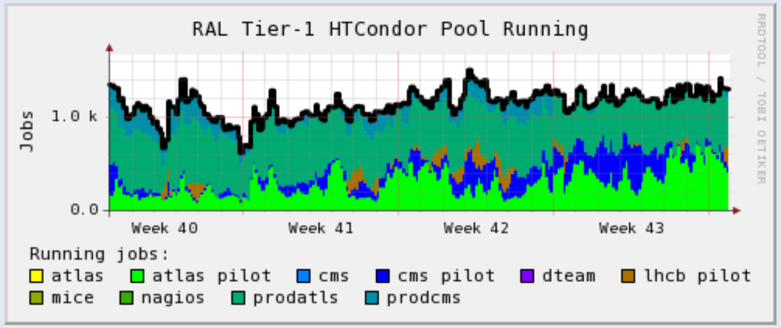
GridPP
UK Computing for Particle Physics

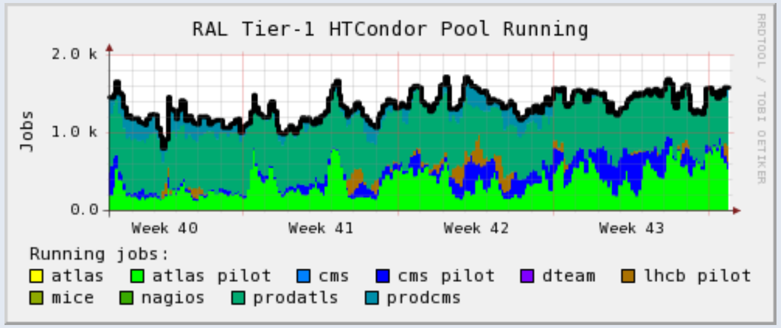
Migration to HTCondor

- CE usage over past month

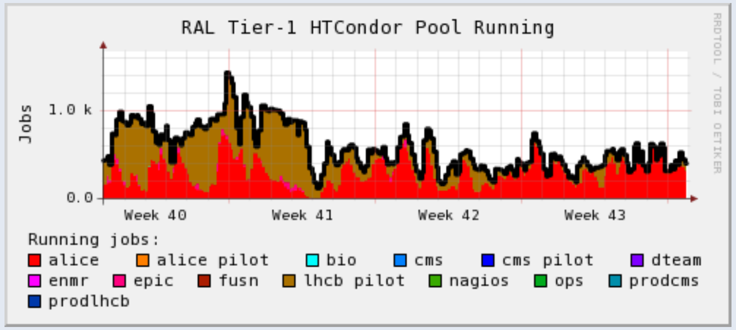
ARC CEs

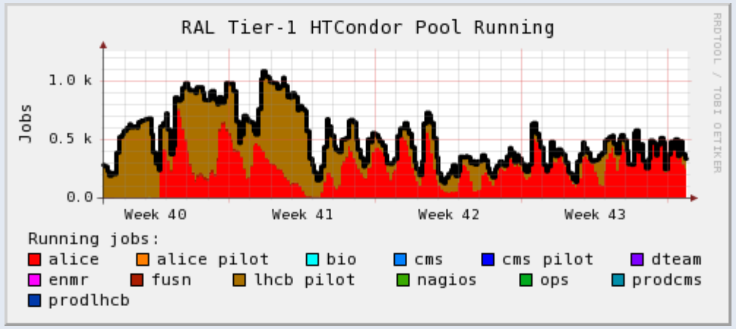






CREAM CEs

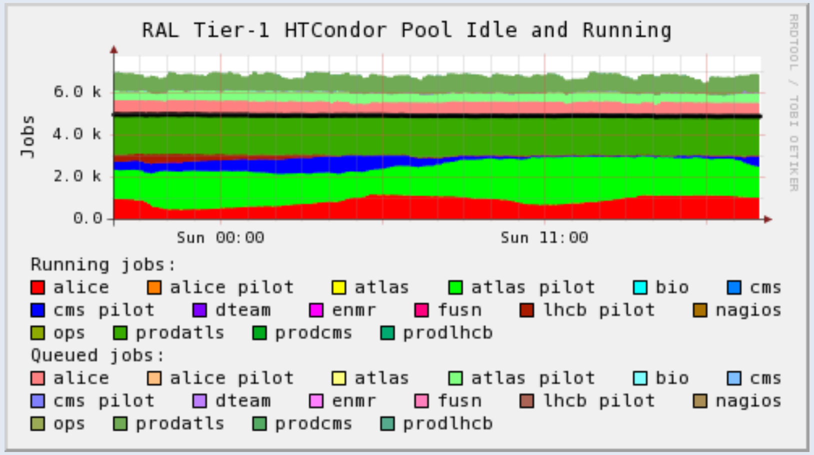




GridPP
UK Computing for Particle Physics

HTCondor: experience so far

- No major problems
  - In some ways this is not good: admins not gaining experience in diagnosing problems
- Support very good
  - E.g. issue found affecting high availability of central manager, quickly fixed & released in 8.0.2
- Even when throttled, job start rate faster than Torque/Maui
- Trivial to extend batch system into a private cloud
  - See talk on Friday



Summary

- Scaling problems with Torque/Maui
- Investigated alternatives
  - HTCondor chosen as replacement
- Current status
  - No major problems with ARC CEs or HTCondor
  - Migration in progress
    - 50% CPU capacity in Torque/Maui, 50% in HTCondor
    - Will complete migration in early November