



# Lancaster's Multicore Adventure.

## *HEPSYSMAN, 13th January 2014.*

Matt Doidge, Peter Love, Robin Long

`m.doidge@lancaster.ac.uk`

Lancaster University

# Disclaimer.

Given the...rushed nature of the multicore deployment at Lancaster this is a tale of how we enabled multicore jobs rather than a how to do it yourselves. We have a far from optimum solution. I apologise for these slides being dryer than a Sandworm's Backside! Over the next few pages we'll discuss:

- Our (naive) strategy.
- How we set ourselves up for multicore (on torque and SGE).
- Our experiences.
- How we're going to clean up our multicore mess.

# Strategy.

- The main problem with a sustainable multicore setup is scheduling alongside normal jobs.
  - But we decided to screw sustainability - at least in the short term, and just try to get things to work.
  - So we decided to use dedicated nodes to run these jobs on.
- To allow us granularity and agility we adopted separate queues for our multicore jobs.
  - This allowed us to be dynamic(ish) with the allocation of resources for multicore-ness.

# Torque Setup, Part One

- First we created a new mcore queue, with only atlas groups in the acs.
  - The other exciting thing was setting `resources_default.neednodes = multi`
  - We gave some nodes the "multi" attribute in `server_priv/nodes`
  - (We gave the rest of our nodes a "single" attribute, and had our normal queue require that).
- After a server restart we found that multicore jobs just worked.

# Torque Setup, Part Deux: Maui

```
RESDEPTH          64
RESERVATIONDEPTH  64
ENABLENEGJOBPRIORITY TRUE
ENABLEMULTINODEJOBS TRUE
ENABLEMULTIREQJOBS TRUE
BACKFILLPOLICY          FIRSTFIT
ALLOCATIONPOLICY          MINRESOURCE
RESERVATIONPOLICY          HIGHEST
NODECFG[DEFAULT]  PRIORITYF=JOBCOUNT
```

Also our ops etc reservations need to require "single" nodes.

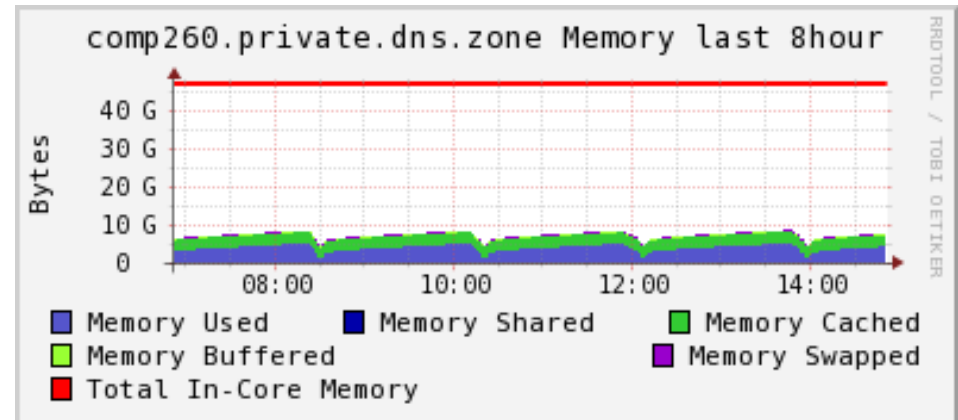
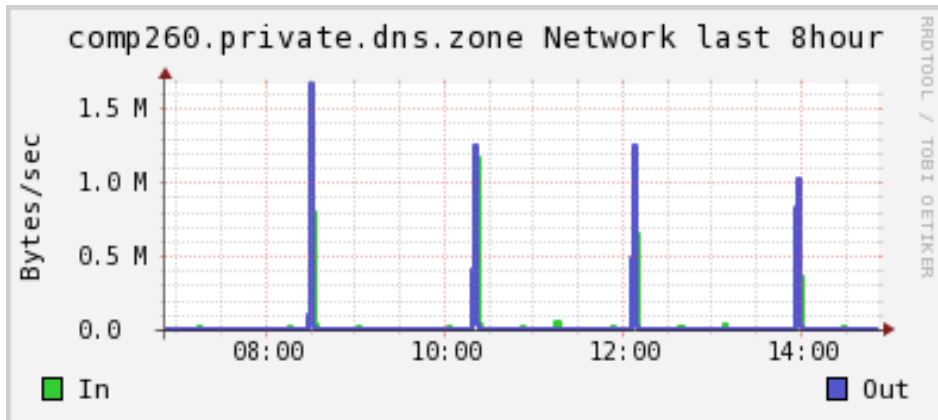
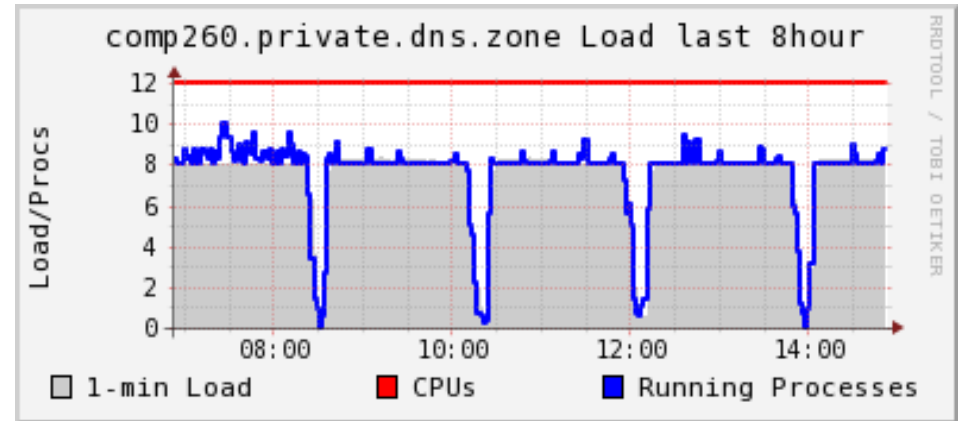
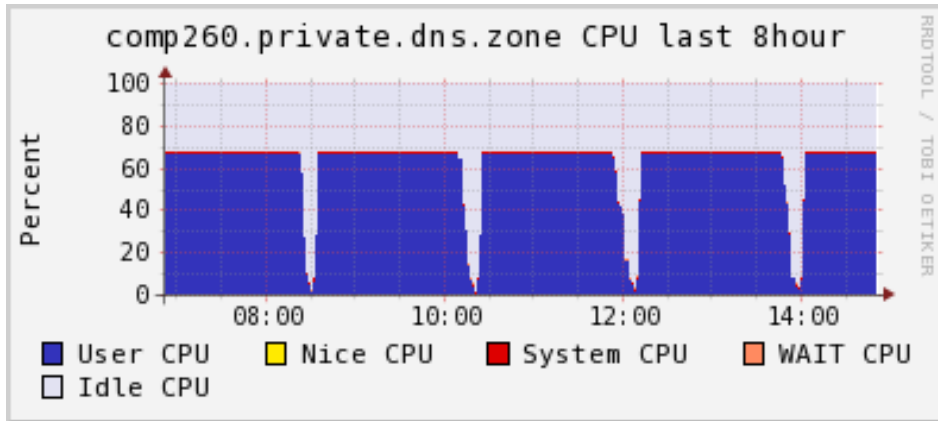
# (Son of) Grid Engine Setup

- We were in luck here, the shared cluster already knew how to handle multicore jobs for local users thanks to Mike Pacey.
- This meant that we could just use the existing smp parallel environment and reverse engineer our starter script from what was already there.
  - I'll happily share these if anyone is interested.
- And once again, after ironing out a few bugs in our starter scripts, multicore jobs just worked.
  - Although a pain for us here is that we're using 12-core nodes...

# Bits and Bobs.

- The two things left were to tweak the bdii and apel.
  - Being yaimophobes we edited our Idifs by hand - which worked surprisingly well.
  - In our apel parser.cfg we set the line "parallel = true" to enable the parsing of parallel jobs.
  - \*But\* I only just set that parameter so we have a reasonable amount of work published without it - I've asked the APEL chaps what this means/what we can do about it.

# Some Multi-Core Ganglia Charts.



"Gap" between jobs is stagein/out (see the network spikes) or AthenaMP taking a long time to ramp up?



# Where we are, Where we're going.

- Our current setup is to have a pool of dedicated multicore resources on each cluster.
  - This gurantees that some multicore jobs can always run.
- In addition to this we have enabled multicore and single core queues in tandem across most of our nodes.
  - This means if we go through a "normal job" drought the multicore work can step up.
- Of course we want to do away with the need for dedicated multicore resources, and schedule dynamically.

# Brain Splurge: Run Time.

- So far MSCORE jobs are very uniform. This could make dynamic reservations easier.
  - Pity the same can't be said for the regular jobs (especially outside of atlas).
- Maybe we need to drastically reduce the maximum allowed job time to 12-24 hours from our usual 72? Three days is longer than most (should) need.
  - Being draconian on run time is one (successful) strategy to get parallel and serial jobs to get along at big clusters (e.g. Polaris).

# Final Musings.

- I'm left wondering if going for separate queues was not the best plan.
  - Although for our shared cluster we have no choice (it is, and always will be, a multi-queue environment).
  - But learn from our mistakes!
- The 8-core hard-coding is a bit of a dog.
  - 4-core jobs would fit a lot better into slots, being easier to schedule, but might not be worth the AthenaMP Overhead.

**This is the End.**

Questions on a Postcard Please.