# Multicore Deployment Task Force report

2016/02/18


**Alessandra Forti**

**Antonio Pérez-Calero Yzquierdo**


WLCG

Worldwide LHC Computing Grid

# WLCG Multicore Deployment TF

**Goals:**

- Explore:
  - Multicore capabilities of local batch systems
  - Compatibility of approaches to multicore job distribution by different LHC Vos
- Produce **guidelines** for efficient multicore scheduling
- Get the sites to **run multicore** (together with VOs)

The multicore TF **period of higher activity** has been Jan-2014 to early 2015, well participated by sites (mainly from T1s) and experiments (ATLAS and CMS) representatives

Once **main objectives were achieved**, full deployment in charge of VOs, so the TF has been kept **open but in a latent state**

Full documentation:

**Project twiki**: https://twiki.cern.ch/twiki/bin/view/LCG/DeployMultiCore

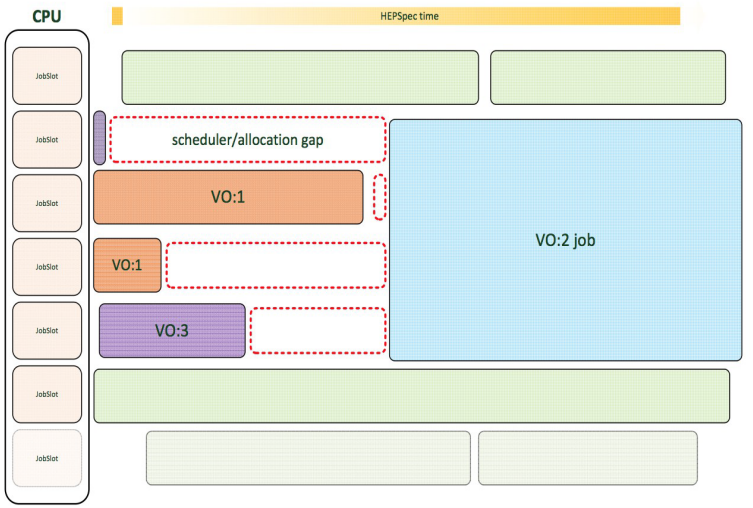
CHEP2015 note: http://iopscience.iop.org/article/10.1088/1742-6596/664/6/062016/pdf

CHEP2015 slides: https://indico.cern.ch/event/304944/session/4/contribution/333/attachments/578522/796661/20150414-chep_mcore.pdf

# Scheduling multicore jobs

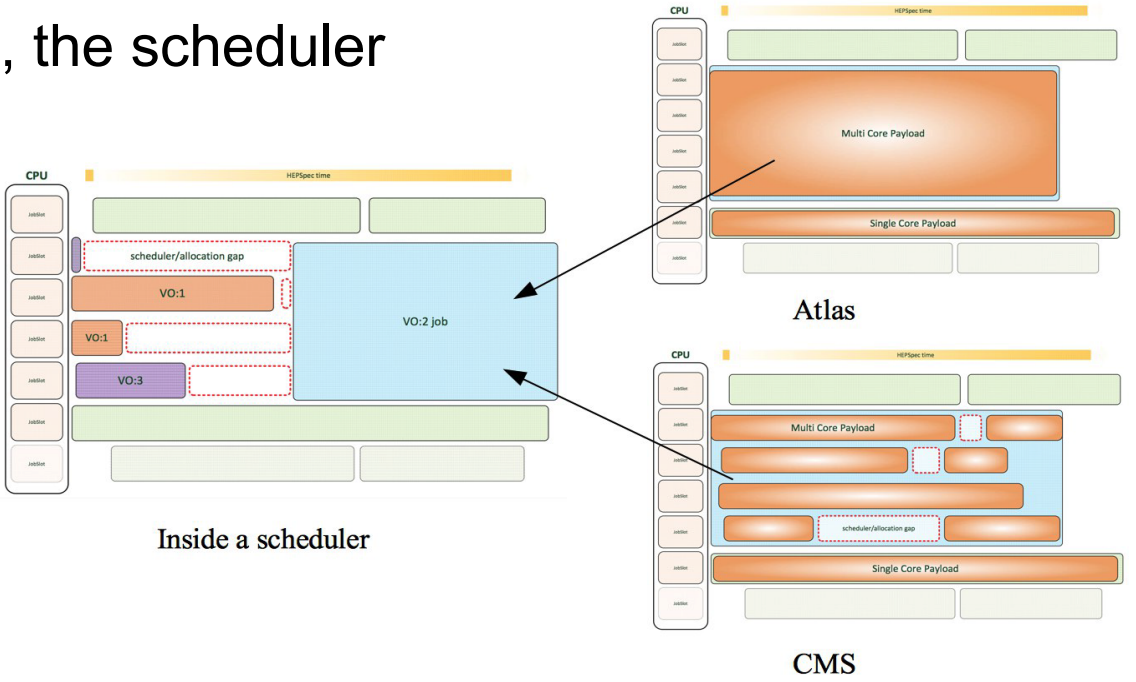
Key problem: in order for a multicore job to start in a non-dedicated environment, a machine needs to be sufficiently ***drained***

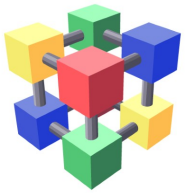
- **Creating a multicore slot:**
  - Prevent single core jobs from taking freed resources
    - **draining = idle CPUs**
- **Conserving the multicore slot:**
  - Higher priority single core jobs taking the resources, destroying mcore slots
    - **wasted draining = need to start again**
- **Limiting draning:**
  - As a protection of farm utilization
    - **Slow ramp up of multicore jobs**



# Experiments submission models

- CMS and ATLAS are using different job submission models:
  - Both VOs **still employ single core payloads together with multicore**
- **CMS:**
  - moves the mixed scheduling inside the pilot
  - one pilot, multiple payloads
- **ATLAS:**
  - mcore and score in parallel, the scheduler does the job
  - one payload per pilot





# Main guidelines from the TF work

With no backfilling available to reduce the draining penalty, reduce the level of draining to the required mcore job pressure and, once the cost of draining machines has been paid, avoid multicore slot destruction. **Recommendations**:

## Experiments:

- Provide a **continuous and stable supply of multicore jobs**
  - vacated slots can be filled with new multicore jobs
  - avoid bursty submission patterns, which force the system to continue and re-adjust the level of draining
- Avoid short jobs, which increase the number of scheduling cycles, potentially leading to increased draining and wastage
- Different VOs should use a **common slot size** for shared sites.
  - the default value is **N_cores = 8**

## Sites: Several techniques explored in the TF thanks to the contribution of participating Tier-1s

- Dynamic partitioning: Torque (**Nikhef, PIC**), LSF (**CNAF**)
- Dynamic scheduling with preferential mcore treatment and adaptable draining: HTCondor (**RAL**)
- Dynamic scheduling with capacity to limit the number of drained slots: SGE (**KIT**)
- Static partitioning not favored or recommended

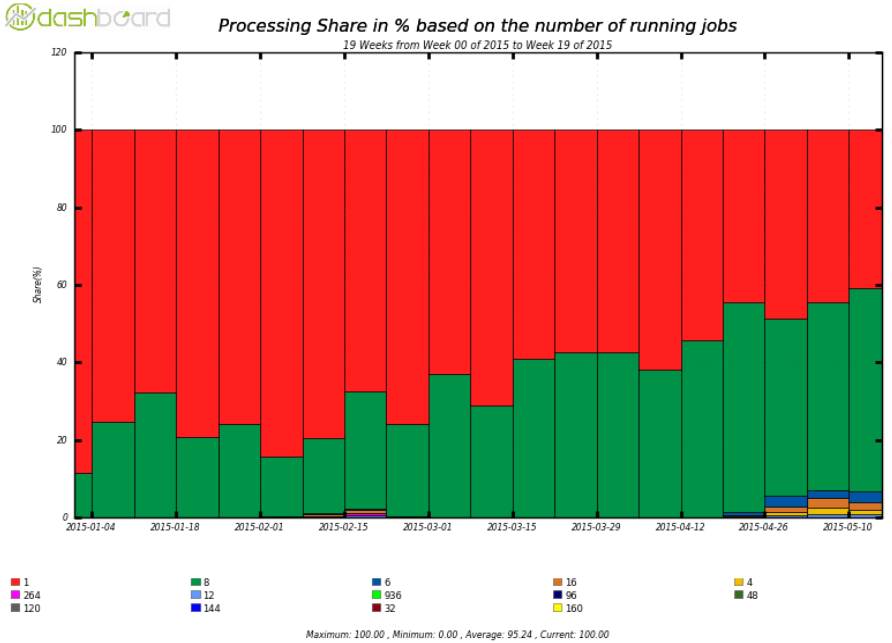
# Passing parameters to the BS

- Multicore TF has also investigated the submission chain, from brokering to kernel for the resource request parameters
  - Main interest from ATLAS
  - An agreement of what parameters should be used in each case has been reached (details in backup slide)
- Status:
  - ATLAS is enabling brokering and passing the parameters to the batch system
    - In particular RSS and walltime values
    - Cputime and vmem are considered at best not useful and at worst harmful.
  - Not in current CMS model: payload scheduling is performed inside the glideinWMS multicore pilots, not at the Batch System

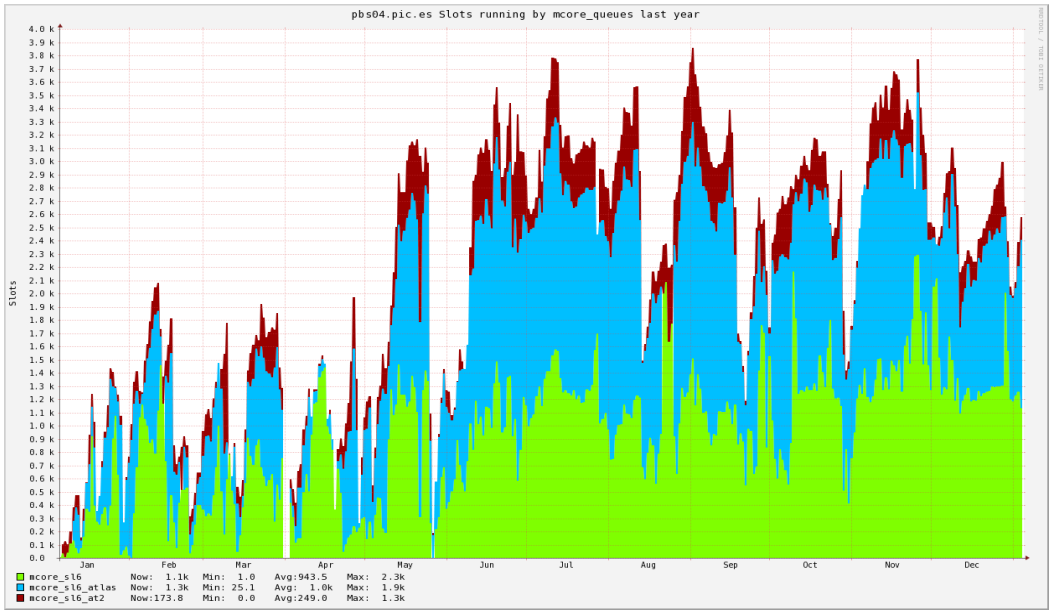
# Results

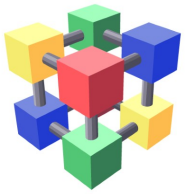
- Main result: successful deployment and utilization of multicore resources
- Increased expertise of the community: sites and experiments

ATLAS jobs by Ncores (first months of 2015)




Mixed utilization of PIC CPUs
for multicore by CMS and ATLAS (2015)





# Main question

- The task force is **currently open but inactive**, having **achieved the main goals**
- **Specific tasks** in the hands of the experiments
  - CMS deployment to T1s complete but deployment to T2s ongoing
  - Effort on passing parameters for ATLAS model lead by the VO
- LHCb moving to multicore:
  - LHCb own multicore submission model to consider pros and cons understood from the comparison of the ATLAS and CMS models
  - Deployment to LHCb sites profit from TF experience
- Should the task force continue open or be closed?

# Memory from brokering to kernel

**Parameters table**

| Batch system | corecount | rss | rss+swap | vmem (address space) | cputime | walltime |
|---|---|---|---|---|---|---|
| Torque/maui | ppn | mem | - | vmem | cput | walltime |
| *GE | -pe | s_rss | - | s_vmem | s_cpu | s_rt |
| UGE 8.2.0(*) | -pe | m_mem_free | h_vmem | s_vmem | s_cpu | s_rt |
| HTCondor(**) | RequestCpus | RequestMemory | No default (Recipe) | No default (Recipe) | Recipe | Recipe |
| SLURM | ntasks,nodes | mem-per-cpu | - | No option | No option | time |
| LSF | ? | ? | ? | ? | ? | ? |

(*) with cgroups support enabled
(**) ARC-CE has a HTCondor backend with *Limit parameters which make it simpler

https://twiki.cern.ch/twiki/bin/view/LCG/BSPassingParameters

**What really happens with the memory?** i.e. what can we really limit? So far it seems we can limit only the address space if

| Batch system | rss | rss+swap | vmem | needs cgroups to do sensible things |
|---|---|---|---|---|
| Torque/maui | - | - | RLIMIT_AS | N/A |
| *GE | - | - | RLIMIT_AS | N/A |
| UGE >=8.2.0 | yes | yes | RLIMIT_AS | yes |
| HTCondor | yes | in 8.3.1 | - | yes |
| SLURM | yes | - | - | yes |
| LSF | in 9.1.1 | in 9.1.1 | RLIMIT_AS | yes |

| Experiments | corecount | rss | rss+swap | vmem | cputime | walltime | comment |
|---|---|---|---|---|---|---|---|
| ALICE | - | - | - | - | - | - | - |
| ATLAS current | corecount | maxmemory | maxmemory | - | maxtime*ncores | maxtime | - |
| ATLAS future | corecount | maxrss | maxrss+maxswap | - | maxtime*ncores | maxtime | maxrss+maxswap really usable only by cgroups enabled sites |
| CMS | - | - | - | - | - | - | - |
| LHCb | - | - | - | - | - | - | - |

| Computing Element | corecount | rss | rss+swap | vmem | cputime | walltime |
|---|---|---|---|---|---|---|
| CREAM-CE Glue1 | JDL: CpuNumber= corecount; WholeNodes=false; SMPGranularity= corecount | GlueHostMainMemoryRAMSize | GlueHostMainMemoryVirtualSize | GlueHostMainMemoryVirtualSize(*) | GlueCEPolicyMaxCPUTime | GlueCEPolicyMaxWallClockTime |
| CREAM-CE Glue2 | JDL: CpuNumber= corecount; WholeNodes=false; SMPGranularity= corecount | GLUE2ComputingShareMaxMainMemory | GLUE2ComputingShareMaxVirtualMemory(*) | GLUE2ComputingShareMaxVirtualMemory(*) | GLUE2ComputingShareMaxCPUTime | GLUE2ComputingShareMaxWallTime |
| ARC-CE | (count = corecount) (countpernode = corecount) | memory(*) | - | memory(*) | cputime | walltime |
| HTCondor-CE | xcount | maxMemory | N/A | N/A | N/A | maxWallTime |

## Experiments