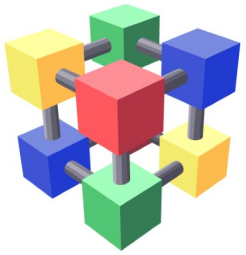
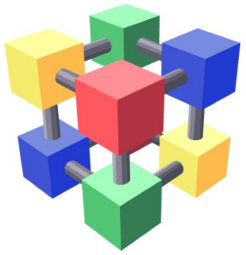


WLCG Multicore Deployment TF

WLCG pre-GDB meeting
Bologna, March 11th 2014

Alessandra Forti

Antonio Pérez-Calero Yzquierdo



Task Force objectives reminder

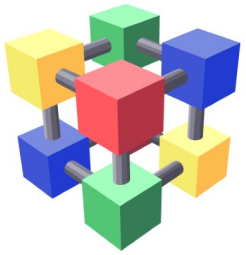
- **Objectives for a successful multicore scheduling strategy:**
  - Integrate scheduling of both **multicore and single-core jobs**,
  - **Maximize CPU usage**
  - **Avoid the need for dedicated resources at sites**

- Job scheduling involves two main elements :

  a) **Grid-wide job submission by experiments**

  b) **Resource allocation at the sites**

The purpose of the WLCG Multicore Deployment TF is to explore, develop and propose ways to connect a) and b) in the most efficient way, with reasonable effort from sites and experiments, and in a reasonable time in order to achieve our multicore scheduling objectives.



Batch systems reviews

- We have just **started reviewing** batch systems in the context of the Multicore TF in terms of
  - **Functionalities** useful for multicore scheduling
  - **Experience** so far
    - ATLAS jobs in production
    - CMS only limited testing

- **Mini workshops** dedicated to each technology
  - Done: HTCondor (RAL), UGE (KIT)
  - To be continued with more input from Torque/Maui, Slurm, LSF, etc.

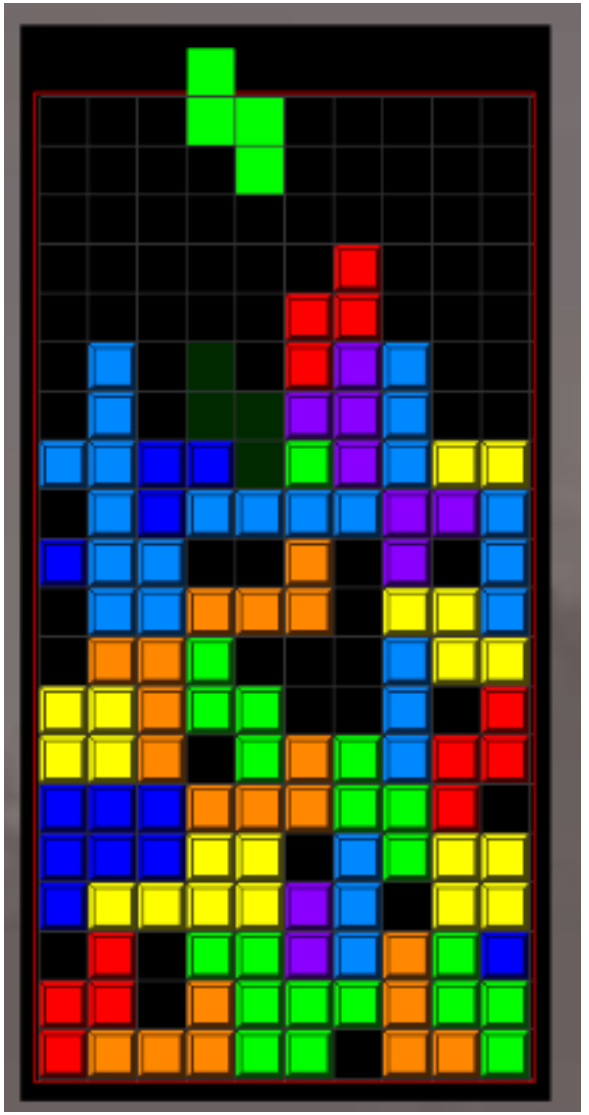
- Even at this early stage we understand we will need **more than one iteration**

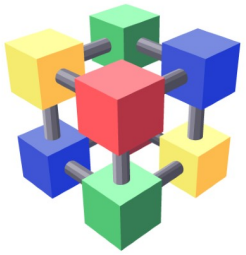
Multicore job scheduling

Scheduling mixed single core and multicore jobs:

- Main tool: s**cheduler with backfilling,** fill gaps with appropriate jobs in the queue, to avoid simple draining and maximize CPU usage.

- Needs reasonably accurate **job lifetime estimation**



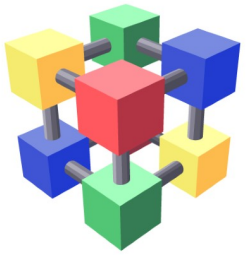


Recurring questions

Scheduling multicore jobs using **backfilling** abilities of local schedulers is based on the concepts of **entropy** and **predictability**

- **Entropy**: a variety of jobs with different walltime requirements
  - what scenario is easier, high or low entropy?
  - can we run in a mixed scenario?

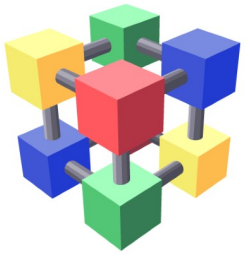
- **Predictability**: can we actually provide an accurate prediction for jobs running time?
  - How accurate this prediction needs to be?



Recurring questions

- How do we **account** for CPU wastage resulting from the scheduling of multicore jobs?

- Is it a site issue or should it be billed to the VOs?
  - Currently no prediction is provided by WLCG jobs: providing precise walltime predictions is not considered a duty of VOs
  - If a job run time deviates from the prediction, thus affecting the scheduling at the sites, would it be reasonable to account to its related VO?



Some conclusions

- Systems reviewed so far are **technically capable** of handling multicore jobs
  - however **reservation** of resources/**draining** of the system is required to be able to absorb them when running together with single core jobs
  - a (so far) small **degradation** of CPU usage has been noticed as a consequence of draining
    - the impact depends on the size of the site
- Job **submission patterns** affect tuning, performance and wastage of the system
  - Wavelike patterns require to constantly tune the amount of draining needed
- Multicore support results from the **interaction** of VO submission models with local batch system capabilities and scheduler tuning
  - **Iterative process**, feedback exchange is needed: sites ↔ VOs
- **Need to continue and extend our tests**: shared ATLAS and CMS sites with diverse batch system technologies