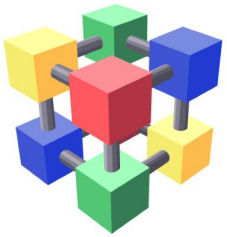
Multicore Task Force

Alessandra Forti
Antonio Perez-Calero Yzquierdo

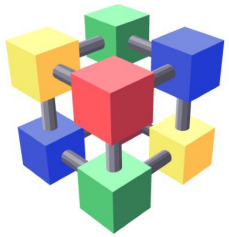
WLCG Multicore TF

21st January 2014



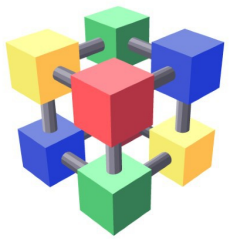
Main points

- Multicore motivation
- What we are trying to solve
- What levels of complications there are
- Multicore job scheduling
- Waste of resources problem
- Different experiment approaches
- What sites are doing
- What the task force should do



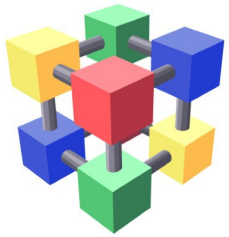
Multicore Motivation

- Hardware evolution
  - Increasing number of cores
  - Cores power remaining more or less constant
  - Memory/core ratio constant
- LHC evolution
  - Higher luminosity
    - Increased number of data volumes and event size
      - Longer processing times and increased memory usage
- Parallelization (multicore)
  - Reduced memory usage
  - Reduced time to process each event
  - Reduced number of jobs and output files to handle



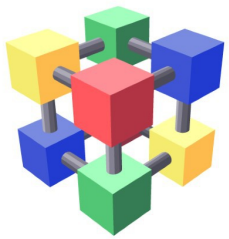
Different Levels

- Modifications at different levels
  a) ~~Application~~
     - Not for this TF to discuss
  b) Grid-wide scheduling
     - Experiments framework and brokering algorithms
  c) Site scheduling
     - Site batch system configuration
- The objective of this WLCG Task Force is to explore, develop and propose ways to connect b) and c) in the most efficient way, with reasonable effort from sites and experiments, and in a reasonable time.



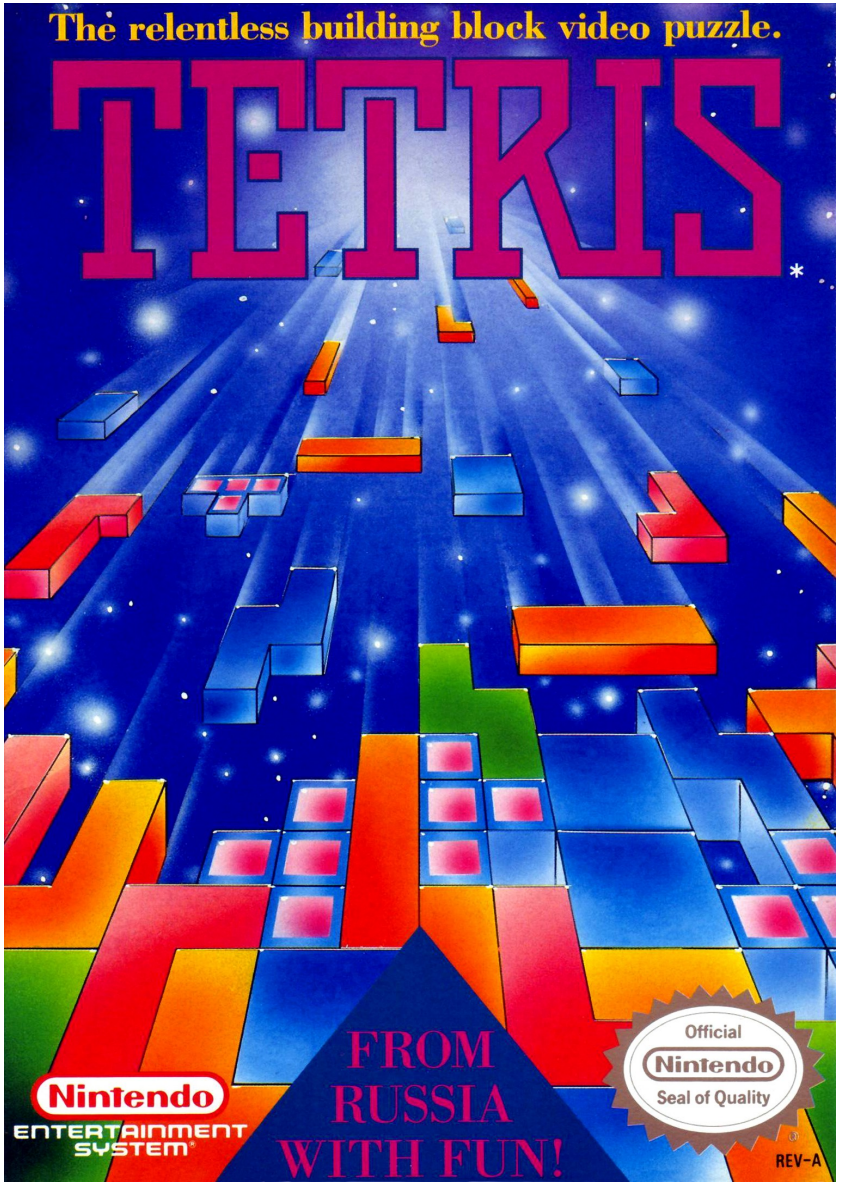
Multicore job scheduling (1)

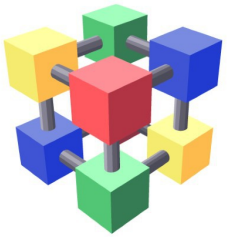
- Objectives:
  - Avoid splitting resources at sites, such as dedicated whole node slots, separated queues, etc:
  - Additional source of CPU inefficiency
  - Increased complexity in sites resources configuration and management
- Therefore we should aim at finding a model to integrate scheduling of both multicore and single-core jobs,
  - Single core jobs will probably still be used by LHC experiments, at least in the near future
  - Single core jobs will also remain in use by other VOs in shared sites



Multicore job scheduling (2)

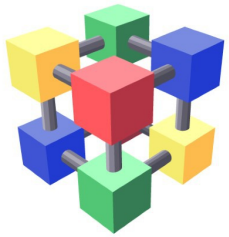
- CPU usage
  - Avoid CPUs being idle if there is job to be done
  - Minimize CPU inefficiencies deriving from scheduling





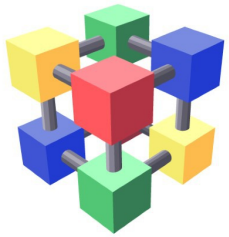
Multicore Job Scheduling (3)

- Solutions should also provide:
  - Proper accounting of the usage of resources.
  - A way of implementing priority policies both at experiments and sites, depending on the VOs (fair share), types of jobs, user privileges, etc.
  - Tools for status monitoring to be used by system operators
  - An intensively automated system to reduce manpower requirements needed to run and maintain experiment workflows active



Draining problem

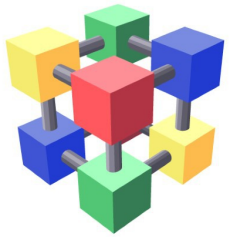
- LHC experiments multicore jobs require 8 cores on the same node
  - 8 cores was arbitrary to avoid whole node resources
- There is no need to setup MPIs.
  - So getting multicore on the WNs is “easy” enough
- The problem is running them without wasting resources
  - The main factor in wasting resources is draining the cores for multicore jobs.
- Draining stems from having to run multicore and single core jobs on the same nodes.



  - It's a scheduling problem batch system developers have tried to solve. Even Maui has a full chapter on backfilling

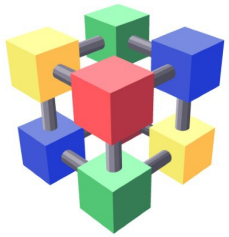
Draining problem in WLCG

- Draining for parallel jobs problem not only a WLCG
  - Even MPI jobs but they request cores on different nodes
    - Easier requirement to satisfy
- In WLCG
  - Dedicated sites vs non dedicated sites
    - Dedicated sites will have more freedom but even within an experiment there will be single core and multi-core jobs
  - In Europe multi-LHC and smaller VOs
    - Race condition among different groups
  - In quite few countries funding depends on CPU efficiency
    - Mishandled multicore wasting resources can really do some damage.
    - Smaller sites more affected



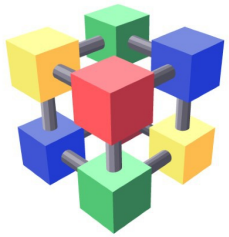
Approaches

1) Scheduling multicore is **only** a site problem
   - Just request 1 node and 8 cores in the JDL
   - Blah parser will convert the request into batch system requirements
   - Batch system will handle multicore/single core jobs race conditions
2) Scheduling multicore is **also** an experiment problem
   - Using multi user pilots run more payloads requesting different number of cores until pilot time is exhausted
   - Dynamic scheduling done internally
     - Internal backfilling with different payloads
   - Aim at avoiding sites allocating dedicated resources
   - Draining minimised by length of pilot



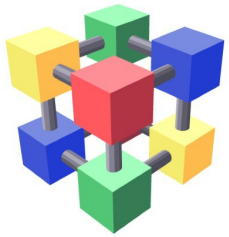
Sites

- Some sites accept multicore jobs
  - Static dedicated resources
  - Dynamic scheduling
    - Aim of everyone working on this should be this
- Different batch systems have different approaches
  - Torque+maui, LSF, Htcondor, SGE, Slurm
    - Each batch system to be reviewed
- Quite few MC sites currently working in Atlas have dedicated resources
  - It's not clear how many resources sites with dynamic scheduling are wasting.
    - We need to measure this



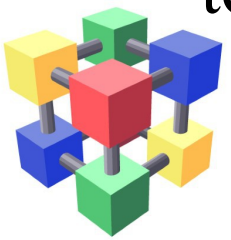
Task Force plans

- Decide on common definitions
  - In particular define what dynamic means
  - If it means job can run on any node but there is a 30% loss of resources perhaps not acceptable.
- Dig further in each approach
  - Experiments experiences
  - Pros/cons
  - Is there a middle way?
- Dig further in sites situations
  - Dedicated batch systems meetings to discuss configurations
  - Sites experience so far
  - Measure how each solution affects site efficiency
    - How this can this be improved?



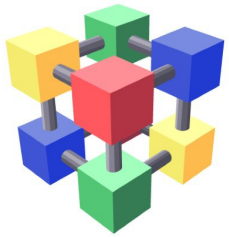
Fixed points

- Neither ATLAS nor CMS can run without multicore after LS1
- Sites cannot waste too many resources through bad scheduling
- Scheduling should not affect other users
- Experiments cannot go each their own way on shared sites resources
- We are not aiming at the perfect solution to deploy but we need the best solutions possible before asking all sites to adapt.



Work already done

- ATLAS
  - CHEP2012: Multi-core job submission and grid resource scheduling for ATLAS AthenaMP
- CMS
  - CHEP2013: CMS multicore jobs scheduling strategy
  - CHEP2013: Minimizing draining waste through extending the lifetime of pilot jobs in Grid environments
- Sites
  - Atlas multicore twiki



Target & Meetings

- Target date
  - A date by which we want to have a working solution for all sites to apply.
    - CMS aiming at October seems reasonable
- Today kickstart meeting
  - Next meetings frequency and time
    - Suggested to start once a week
      - Experiments approaches
      - Sites scheduling mini-workshops
      - Common language definitions