Condor near future talk

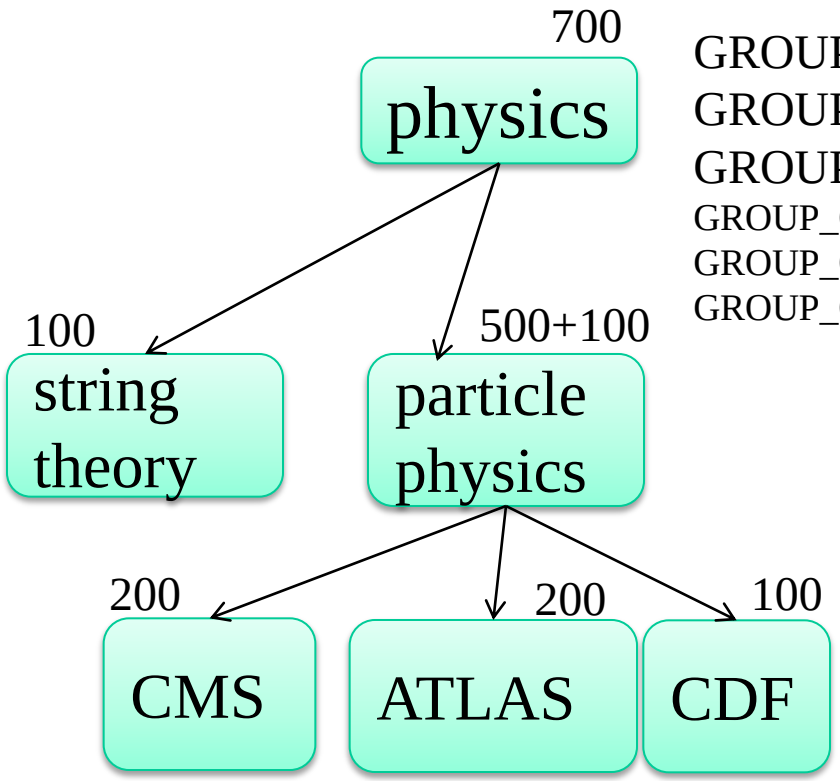
glideinWMS Training @ UCSD
January 2012

Condor Project
Computer Sciences Department
University of Wisconsin-Madison

Release Situation

- Stable Series
  - Current: Condor v7.6.6 (maybe the last?)
- Development Series
  - Current: Condor v7.7.4
  - Condor v7.7.5 (end of January)
  - Condor v7.7.6 == Release Candidate for v7.8.0

Hierarchical Group Quotas




```
GROUP_QUOTA_physics = 700
GROUP_QUOTA_physics.string_theory = 100
GROUP_QUOTA_physics.particle_physics = 600
GROUP_QUOTA_physics.particle_physics.CMS = 200
GROUP_QUOTA_physics.particle_physics.ATLAS = 200
GROUP_QUOTA_physics.particle_physics.CDF = 100
```

group.sub-group.sub-sub-group…

Group Accounting changes

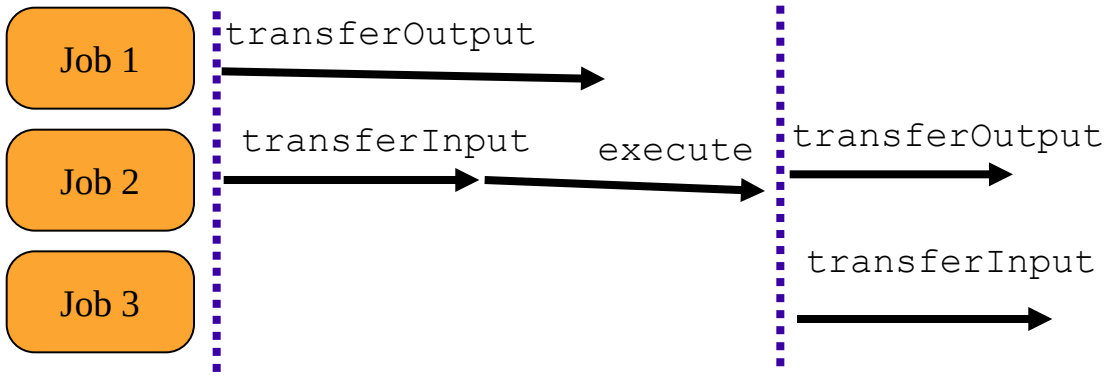
- Condor_userprio gives group information
- GROUP_AUTOREGROUP changed to work as it was in v7.4
  - Now it is NOT equivalent to GROUP_ACCEPT_SURPLUS
  - Desired by CDF, FermiGrid

Grid/Cloud/VM Universe Changes

- Cloud service handled as a job scheduler type in grid universe
  - Condor moved from Soap to Query (Rest) interface for EC2
- Support for submission to SGE, Globus 5.x, improved performance via batching for CREAM
- BOSCO (Blahp Over Ssh Condor Overlay)
  - Help for individuals to build community grids
- Cisco UCS integration

Asynchronous sandbox transfer

Interleave output sandbox transfer and next job's input transfer/execution within same claim



Scheduling Improvements

- Mechanism for balancing how much of the pool is devoted to single-core vs. full-machine jobs (or big –vs- small RAM, etc). Key Issues:
  - Will we make use of time estimates for job completion?
  - How will we choose a machine to drain?
  - How will drained machines change personality?
  - What needs to be monitored?
  - How do we decide when to initiate draining?
- So far led to improvements in statistics and partitionable slots

Statistics

- Expose operational data oftentimes buried in the Condor services via ClassAd attributes. Examples:
  - Claimed time spent transferring files –vs- running jobs
  - Goodput –vs- badput
  - Negotiation times
- Attribute Types
  - Since startup, sliding window average, histograms

Paritionable Slot Improvements

- Startd advertises everything (ex: 8 cores 8GB ram), job takes what it needs (1 core, 6GB ram), leftovers available for the next job (7 core, 2GB).
- Addressed two shortcoming
  - Fragmentation
  - Many negotitation cycles required to fill a node
- Hope will simplify HTPC scheduling

Other near-term items

- Finish workflows faster
  - Machine disappears, jobrescheduled in 20 minutes (instead of 2 hrs)
  - Non-leaf nodes in a DAG will keep the claim
  - IPv6 first-pass support
- More useful machine ad attributes
  - Distribution info, RSS, PSS, GPU info
- File-remaps (need root), cgroups support (needs RHEL6, root)
- Shared Libs put Condor on a diet

Near term to medium-term…

- Better glexec handling
  - Handle proxy expiration, perform retries, … [http://tinyurl.com/7852msc]
- Checkpoint / Resume in Vanilla Universe
- Improve job sandbox I/O transfer situation
  - Bittorrent file transfer plugin? Transfer daemon(s) outside of the schedd? Auto-squid caching?

Interested in what could Condor do to increase the happiness of glideinWMS admins

Thoughts so far…

1. Mechanism to *require* users to *explicitly* state memory, cores, etc?
2. Matchmaking analyze help for startd requirements just like job requirements?
3. More automatic parameter tuning?
4. Simplify configuration/setup for GSI authentication and authorization?
5. Differential schedd/collector polling?
6. More site scheduler glidein-awareness – communicate deadline changes, condor_who / condor_ps ?
7. Automatically flag/analyze jobs that do not match?
8. More direct link from daemon log messages to documentation?
9. Python bindings for ClassAd library?

…your thoughts?