

CHEP 2016 Distributed Computing Talks Overview

Tanya Levshina

Common Problems:

- Hungry for CPU. Started to target local (HLT)/ opportunistic (CMS and ATLAS using not owned OSG resource; LHC@Home) / Private and Commercial Cloud/ HPC
- Increased appetite for memory
- Necessity to use resources efficiently
 - Network forecasting and provisioning
 - Systematic studies of jobs' resource utilization
 - Proactive workload assignment
 - Multicore jobs
 - Partitionable slots
 - Improved production workflow
 - Fine grained (event-level) workload partitioning
 - Help user by providing automated jobs splitting

Workflow Management:

Problems:

- Inefficiency due to old resource partitioning based on geographical grouping of computing centers (Monarch hierarchy issues)
- Suboptimal usages of non-traditional resources due to job-based workload management
- Incoherent implementations for various HPC workflows
- Overstretched architecture of the pilot to support non-traditional resources (Atlas)
- Lack of prediction capabilities for resource availability

ATLAS (PanDA):

- Resource consolidation:
 - There are two new site concepts:
 - Nuclei sites are the Tier 1s and large Tier 2s, where tasks will be assigned and the output aggregated
 - Satellites are the sites that will execute the jobs and send the output to their nucleus
 - Nuclei and satellite sites are dynamically paired by PanDA for each task based on the input data availability, capability matching, site load and network connectivity

- The new model has impact on an improved storage usage and more flexibility in resource usage, while avoiding network bottlenecks
- Intelligent brokerage:
 - More intelligence to the brokerage based on:
 - Job retry history
 - Network forecast
 - Cache hit rate
 - Memory consumption
- Proactive workload assignment:
 - Assigning jobs just before resources become available
 - Removing jobs just after resources become unavailable
 - Based on (quasi) real-time resource information
 - Workload assignment could trigger input data transfers
 - Should be smart enough for proactive assignment to minimize redundant data transfers
- In-House security

To authenticate requests from the pilot running on special environment where standard X509-based auth is unavailable
- Enhancement of event service:
 - Fine grained (event-level) workload partitioning
 - Allowing jobs to be revoked in the middle of processing with minimized losses

CMS (GlideIn WMS):

- Global Condor pool
 - Almost each workflow can run anywhere
 - All CPU joined to one Global HTCondor pool + dedicated Tier-0 pool
- Tier-1 & Tier-2 disk managed via Dynamic Data Management (DDM)
- One central Request Manager:
 - Unified Tool
 - Handling of input data staging and placement
- Assignment of request to resources:
 - Largely automated based on configurations for different campaigns
 - Recover failed parts of a request
 - Job splitting and submission to Global Pool
 - Supervise placement of input data (primary, secondary)
 - Utilize as many resource as possible over the grid
 - Spreads the load to all possible sites
 - Integrates opportunistic resources for processing
 - Dynamically re-route work to reduce latencies
- Partitionable slots

LHcb, LSST, Pheno (DIRAC)

- Testing Data Availability Matching (DAM). The DAM model takes into account a job input data availability on a requesting tier. For the data management model the current replication strategy was used.
 - <https://indico.cern.ch/event/505613/contributions/2230718/attachments/1347085/2041853/Poster-296.pdf>
- Testing elastic cloud resource scheduling is an extension of DIRAC. VMDIRAC starts VMs equipped with Job agents through cloud managers according to the demands of DIRAC task queue.
 - https://indico.cern.ch/event/505613/contributions/2230747/attachments/1347682/2038770/Poster_354.pdf
- Context-aware CloudScheduler +Shoal, Squids and DynaFed (federation of storage elements using http or WebDAV protocol)
 - <https://indico.cern.ch/event/505613/contributions/2230405/attachments/1345978/2045227/Oral-052.pdf>
- Adopted a new approach which detaches the payload running from the Belle II DIRAC pilot which is a customized pilot pulling and processing jobs from the Belle II DIRAC system, so the payload can run without requiring credential.
 - <https://indico.cern.ch/event/505613/contributions/2227710/attachments/1336660/2030003/Poster-007.pdf>
 -

Resource Hunt:

- HPC
 - HTCondor + ROCED (Cloud Scheduler) - n 5 million CPU hours within the month
 - ATLAS:
 - Titan at OLCF
 - Edison/Cori at NERSC
 - SuperMUC at LRZ
 -
- Opportunistic
 - HTCondor + Docker (local) - 10k CPU hours over weekend
 - @Home - volunteer computing for MC processing
 - BELLE II DIRAC
 - HighLevel Trigger for off-line production
- Commercial Cloud
 - 1&1 Internet Cloud (HTCondor + ROCED) - 800 simultaneous slots
 - HTCondor + Decision Engine+ AWS - 10000 simultaneous slots
 - ARC CE+ SLURM+SWITCHengines (Open Stack) - 300 cores 24/7
 - Private Cloud

- OpenStack 1100 cores, ~ 40 TB
- CNAF OpenStack + LSF Dynfarm extension
- All Resources as One Big Cluster
 - HELIX NEBULA Science Cloud: is a Pre-Commercial Procurement project with a budget of more than 5M€ that is co-funded by the European Commission The objective is to produce a hybrid cloud platform for the European research community
 - HEP CLOUD Project: Add disparate resources (HPC slots, Cloud VM, OSG nodes, local resources) into an HTCondor pool.

<https://indico.cern.ch/event/505613/contributions/2230729/attachments/1346953/2043387/Oral-400.pdf>

<https://indico.cern.ch/event/505613/contributions/2230753/attachments/1347601/2038816/Poster-545.pdf>

<https://indico.cern.ch/event/505613/contributions/2230742/attachments/1345726/2028848/CHEP16-ATLAS-ID86.pdf>

https://indico.cern.ch/event/505613/contributions/2230743/attachments/1344438/2026306/chep_2016_poster.pdf

<https://indico.cern.ch/event/505613/contributions/2230727/attachments/1348598/2046641/2016-10-13-CHEP-HNSciCloud.pdf>

<https://indico.cern.ch/event/505613/contributions/2230752/attachments/1346565/2030523/Poster-414.pdf>

<https://indico.cern.ch/event/505613/contributions/2230739/attachments/1346645/2039021/garzo-glio-chep2016.pdf>

Workflow Management

- LOBSTER: a workflow management tool for exploiting volatile opportunistic computing resources:
 - System resource specifications for a task category can now be modified while a project is running, avoiding the need to restart the project if resource requirements differ from the initial estimates.
 - Lobster now implements time limits on each task category to voluntarily terminate tasks. This allows partially completed work to be recovered.
 - Allows dependent tasks to begin as soon as sufficient input data has accumulated.
 - Utilizes a new capability in Work Queue to monitor the system resources each task requires in order to identify bottlenecks and optimally assign tasks.

- <https://indico.cern.ch/event/505613/contributions/2230715/attachments/1347614/2041496/Oral-224.pdf>
- JSUB is a lightweight and extensible task submission and management tool
 - Small experiments or experiments in early stage can extend it for quick access of distributed resources for massive production
 - JSUB has been implemented in Python. Dirac and Condor supported
 - Goals: Automatically manage massive jobs Highly extensible for new experiments. Ease the procedure to use heterogeneous resources
 - Life cycle of a task (a batch of jobs in one submission)
split->submit->run->status monitor->output retrieval -> reschedule

<https://indico.cern.ch/event/505613/contributions/2230719/attachments/1347681/2038982/oral-330.pdf>

- CRAB3 - using DAG for job splitting:
 - Implemented automatic task splitting
 - Tail-splitting feature is currently disabled, will be enabled pending more tweaks to address resource consumption

<https://indico.cern.ch/event/505613/contributions/2230719/attachments/1347681/2038982/oral-330.pdf>

Monitoring evolution

- Rise of data visualization such as plots, histograms, and task chain diagrams
- Necessity to do analytics
- Trying to do predictive analytics:
 - expected task completion time and comparison with actual task computation progress
 - Expected memory usage
- Tools used:
 - ES
 - Kibana
 - Logstash, Flume
 - Graphite/Grafana
 - Ganglia
 - Sensu
 - Jython