Big PanDA integration with Titan LCF.

Danila Oleynik

UTA / JINR

Outline

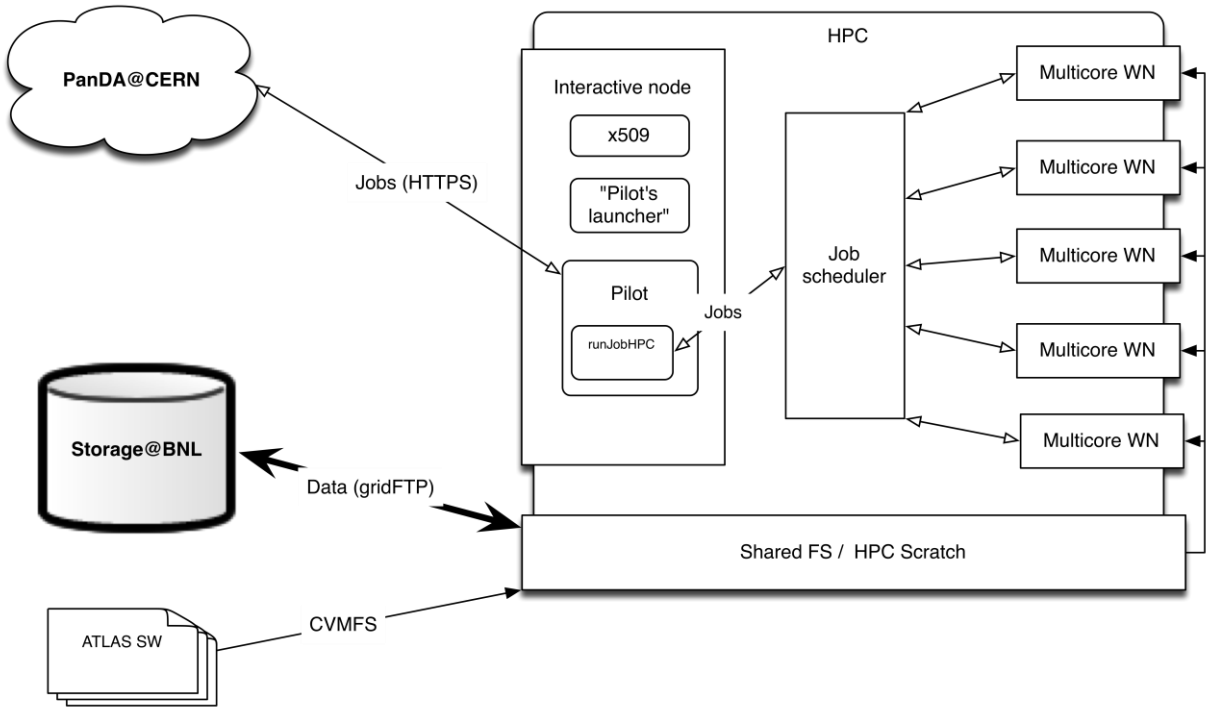
- Titan LCF specialty
- BigPanDA architecture for Titan
- PanDA Pilot changes
- Backfill
- Titan allocation policy
- Occupation algorithm. Initial testing.
- Issues.

Titan LCF specialty

- Titan Cray XT7
  - 18,688 nodes
  - node: 16 core, 32 + 6 GB RAM (2GB per core)
- Parallel file system shared between nodes, recently upgraded: project workspace 100TB quota (30 PB total capacity)
- Highly restricted access:
  - One-Time Password Authentication;
  - No network connection with worker nodes;
- 3 layers of nodes:
  - Interactive nodes: user interactive login;
  - Service nodes: job setup operations, managed through PBS/Torque directives;
  - Worker nodes: job executions, managed through ALPS (Application Level Placement Scheduler);
- Limitation on number of jobs in scheduler for one user
- Special data transfer nodes (high speed stage in/out)
- System naturally designed for parallel execution

BigPanDA architecture for Titan

- Pilot runs on HPC interactive node
- Pilot interacts with local job scheduler (PBS)
- Number of executing pilots number of available slots in local scheduler (???)



PanDA Pilot changes

- Native PanDA pilot was successfully started on Titan interactive nodes.
  - Correct definition of PanDA queue was needed.
- Main modification was performed for payload execution part: runJobTitan.py module was developed based on runJob.py module.
  - Method, which call payload execution was changed for run and collect results of job execution through PBS;
  - Interface with PBS job manager was implemented by using SAGA API
- Some minor modifications of cleanup procedures was done (subdirectories cleanup).
- Proper setup and execution of MPI jobs through ALPS.
- Function for collecting information about available resources for backfill was implemented
  - Full PanDA workflow on Titan was done.

Opportunistic job backfill on Titan

- As a first step a simple algorithm was implemented:
  - Pilot queries MOAB scheduler about unused transient resources
  - Information about available resources returns in a format that includes a number of currently unscheduled nodes and period of their availability
  - Pilot chooses the largest available block of free nodes and generates appropriate job submission parameters taking into account Titan's scheduling policy limitations

Titan scheduling policy

- Job's wall-time limit depends on number of requested nodes.

Min Nodes	Max Nodes	Max Walltime (Hours)	Aging Boost (Days)
11,25	–	24.0	15
3,75	11,249	24.0	5
313	3,749	12.0	0
125	312	6.0	0
1	124	2.0	0

- For example one can't request 100 nodes (1600 cores) for more than two hours (regardless of declared period of backfill availability)

Initial backfill tests on Titan

Submitted	Account	Nodes	Cores	Wait	Walltime limit	Runtime	State	Completed
Mar, 04 16:26	CSC108	**6**	**96**	0.00	1:59:00	0,01	Completed	Mar, 04 16:27
Mar, 04 16:52	CSC108	**185**	**2960**	0.07	5:59:00	0,02	Completed	Mar, 04 16:58
Mar, 04 17:32	CSC108	**608**	**9728**	0.01	11:59:00	0,02	Completed	Mar, 04 17:34
Mar, 04 17:45	CSC108	**578**	**9248**	0.01	11:59:00	0,03	Completed	Mar, 04 17:47
Mar, 04 17:51	CSC108	**1,649**	**26,384**	0.00	11:59:00	0,03	Completed	Mar, 04 17:53
Mar, 04 18:03	CSC108	**636**	**10176**	0.01	11:59:00	0,02	Completed	Mar, 04 18:05
Mar, 04 18:09	CSC108	**740**	**11840**	0.13	11:59:00	0,02	Completed	Mar, 04 18:18
Mar, 04 18:21	CSC108	**577**	**9232**	0.00	11:59:00	0,03	Completed	Mar, 04 18:22
Mar, 04 18:25	CSC108	**596**	**9536**	0.04	11:59:00	0,02	Completed	Mar, 04 18:28

- “Backfill capture” is almost instantaneous!
- No competition for the resource?
- More studies are planned

Issues

There are two scenarios of using resources:

1. Using MPI (“natural” payloads for HPC):
   - Unpredictable size of output (output size may changes with number of involved cores)
2. “Multiple Simultaneous Jobs”
   - 100 simultaneous jobs in one submission;
   - Job occupy minimum one node (not core);
   - Significant redesign of Pilot (to take care about set of PanDA jobs, against one)