# Harvester for HPC

Tadashi Maeno (BNL)
on behalf of Harvester team

ATLAS TIM,
18-22 September 2017, CERN, Switzerland

# Constraints for Workload Management 1/2

- Preemptable or very short walltime limit
  - To shorten the execution time of jobs
    - Decreasing the number of events per job, and/or
    - Increasing the number of CPU cores per job
  - Or to enable event-level bookkeeping (event service)
- Limitation on number of concurrent workers in the batch system (e.g. ~10)
  - To increase the number of CPU cores per worker
    - Combining multiple jobs to a single payload which is given to a worker (multi-job or ManyToOne)
    - Increasing the number of events per job (jumbo job)
- No outbound network connectivities on compute nodes
  - Edge service on edge node to mediate communication between Panda and workers
- Long waiting time in the batch queue
  - To assign only low priority jobs
  - Or to enable parallel event consumption on pledged resources (multiple consumers or jumbo job)

# Constraints for Workload Management 2/2

- Intermittent and/or spiky resource availability
  - To send “fake” pilot requests from edge service (get_job requests for job pre-fetching or update_job requests for jobs in stating state)
  - Or to request jobs before resources become available (proactive workload assignment)
- Regular downtime
  - To introduce a new queue status to temporarily prolong various timeout values

# Custom Tasks

- Good to use the resource anyway
  - Optimization of the number of events per job for the resource and workflow
    - E.g. very small number of events per job to minimize losses due to preemption
  - Dedicated workqueue
    - Tasks can generate jobs without competing with other tasks based on task priorities
    - A pool of activated job due to nQueueLimit even if the resource is temporarily inactive
  - Preassigned to the resource
    - Bypassing the brokerage which skips inactive resources
    - No competition with other resources
- Not good for automation
  - Production managers have to look for good tasks for HPCs, empirically set event set sizes, and define downstream processing steps accordingly

# Goal

- To have full automation without custom tasks in order to release production managers and operation people from babysitting
  - No special tuning for job sizing
  - Without dedicated workqueue
  - With the standard brokerage
  - No slowness in task completion time due to usage of HPCs
- A common pilot/worker provisioning machinery
  - Each HPC can use different plugins and workflows
  - Commonize monitoring views and operational knowledge
- More optimal usage of compute resources

# Workflows 1/4 : Push+True Pilot

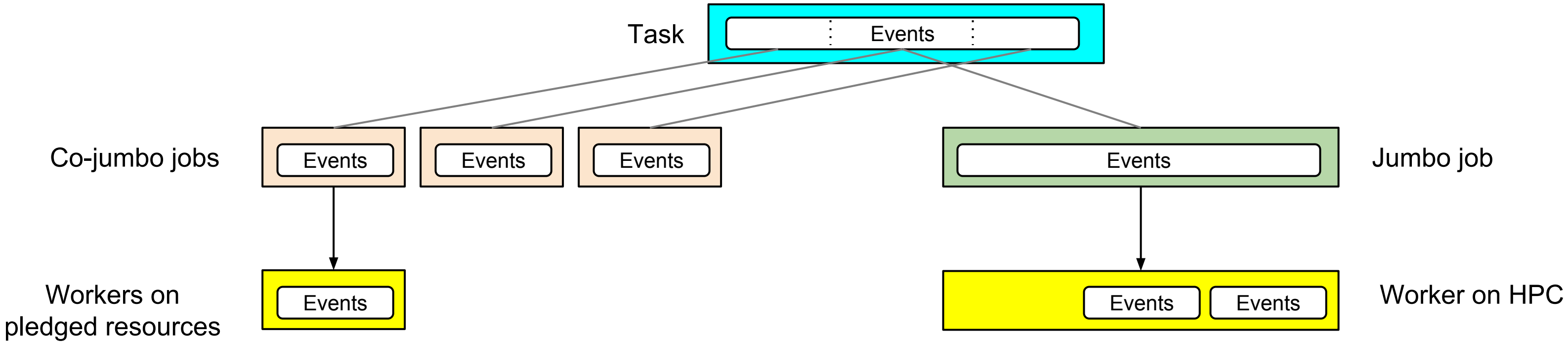
- Prefetches jobs, submits workers(pilots)+jobs to the batch system, and lets workers communicate with panda once they get CPUs
- Advantages
  - Easy to send get_job requests without empty workers to attract jobs before the resource becomes available
  - A pool of prefetched jobs as a buffer for fluctuated CPU availability
  - Automatic throttling of worker submission in case of no jobs
  - A well matured workflow in ATLAS as it has been used for some grid sites for a long time
- Caveats
  - Requires less restrictive operation policy
    - Outbound network connection on compute nodes, many batch workers running in parallel, long walltime limit with allocation
  - High prio jobs cannot get the first available CPUs

# Workflows 2/4 : ManyToOne

- Prefetches multiple jobs, combines them into a single payload, and submits the payload to the batch system
- No MPI : one job per rank/node
- Essentially the same as “multi-job pilot”
  - One major difference is that jobs are prefetched and input files are asynchronously pre-staged before CPU slots become available, while multi-job pilot fetches jobs and stages input files once free CPU slots are found
- Advantage
  - The number of concurrent workers in the batch system can be reduced
- Caveats
  - Needs jobs with similar execution time so that all jobs in the same worker finish simultaneously to avoid having idle nodes
    - E.g., jobs from the same task or request. Cannot accept jobs from random tasks → Custom tasks
  - Or needs to enable event service
    - When the first job finishes all the rest could be killed

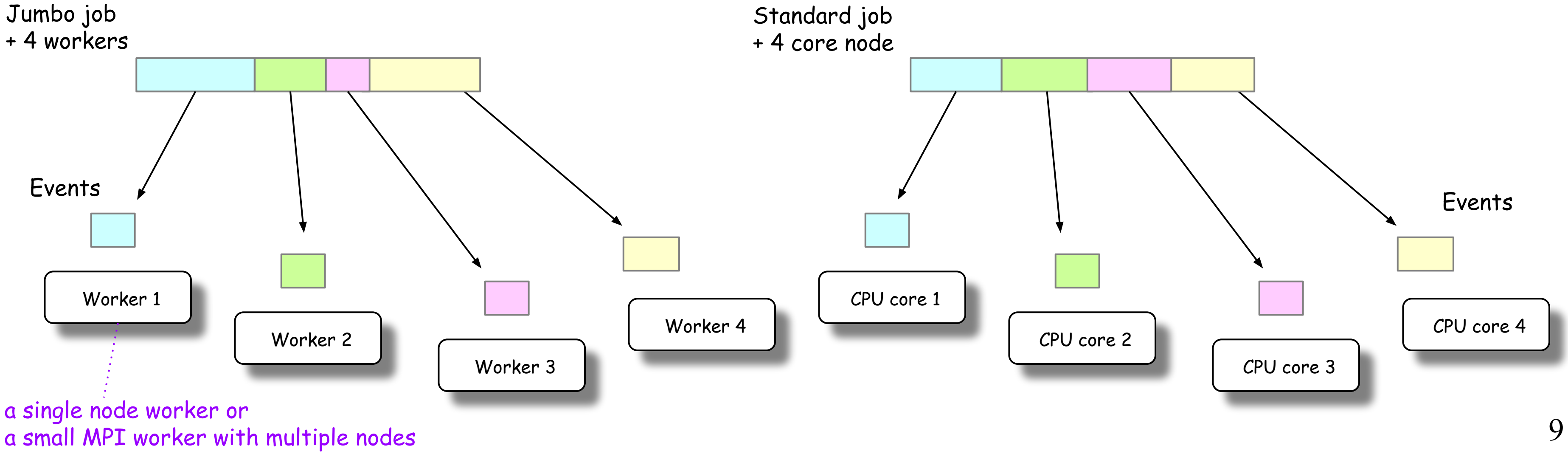
# Workflows 3/4 : Jumbo Jobs

- One single huge event set (jumbo job) including all events from one task
  - A huge event set + event-level bookkeeping allows a big batch worker to process events at HPCs as much as possible
  - Multiple jumbo jobs per task to be assigned to different HPCs
  - Don't have to estimate optimal event sizes for each HPC
- The huge event set is partitioned at the same time to small event sets (co-jumbo jobs)
  - They are good to be processed by small batch workers at pledged resources
- Workers for jumbo and co-jumbo jobs compete to grab events
  - Each event is exclusively processed by one worker
  - Events are being consumed at pledged resources even if big workers are waiting in long HPC batch queues



# Workflows 4/4 : Multi Workers

- Many workers contributing to the same job
- Typical use-case : Jumbo jobs + small workers
  - Single node workers
  - Small MPI workers with backfill mode
- Job and file records for each jumbo job is huge in the database
  - Not good to have one jumbo job for each small worker
- One standard job is processed by many CPU cores → One MPI job is processed by many compute nodes → One jumbo job could be processed by many workers
  - Workers don't have to pop-up simultaneously → Workload sharing with asynchronous workers without node-boundaries



# Theta/Titan Workflow with ALCC

- Limitation on the number of concurrent batch workers → needs a large workload for each worker
- Current workflow and plugins
  - ManyToOne
  - Cobalt or SAGA plugins
  - Pilot mover (rucio download/upload) or GlobusOnline plugins
- Issues
  - GO transfers files efficiently but limitation is tight
    - To reduce transfer tasks by grouping files in harvester like an example or using Rucio
  - Jobs with similar execution time are required to avoid having idle nodes. I.e. custom tasks are still needed
- Future workflow for full automation
  - ManyToOne + Event service
    - The first successful job would terminate all the rest in the same worker, in order to release all nodes simultaneously
  - Jumbo jobs
    - As ManyToOne + Event service still has the problem with small jobs which could terminate workers too quickly

# NERSC Workflow

- Like a big computer cluster
  - Outbound network connections are available on compute nodes, many batch workers can run in parallel, and walltime limit is long enough due to allocation
  - The number of available CPUs can fluctuate
- Possible workflow and plugins
  - Push + True pilot
  - Slurm plugins
  - Pilot mover or GO plugins
- Just a matter of when
  - E.g. Edison and Cori-1 try first and Cori-2 migrates if they are successful
- Ordinary jobs first to get rid of custom tasks
- Event service or jumbo jobs for optimal CPU usage

# Titan Workflow with backfill

- Workers can be terminated by preemption → walltime (i.e. optimal size of event chunk) is unpredictable
- Possible workflow and plugins
  - Jumbo job + Multi-workers
  - SAGA plugins
  - Pilot mover plugins
  - Backfill module
- Challenges
  - Jumbo jobs and multi-workers are available, but not yet tried in production environment
    - Jumbo job is essentially just a large event service job, but largeness is always a killer
  - New Yoda
  - New monitoring since traditional 1-to-1 mapping between job and pilot is broken
  - Backfill module to be integrated in harvester

# Plans for HPC

- ➢ Bringing Theta/ALCF into production
  - To fix a bug in mini-pilot which reports job was successful although local transfer was not done → job is flagged as failed in panda since output file is missing
  - To reduce memory consumption of harvester
  - To improve GO plugins to reduce the number of active transfer tasks (up to 3)
  - To improve mini-pilot to report missing job data, e.g. maxRSS, cpuConsumptionTime, ...
- ➢ Migration of NERSC to Harvester
  - Mini-pilot + traditional jobs first
  - Yoda + event service jobs next
- ➢ Testing event service at Theta and Titan
- ➢ Getting rid of custom tasks for HPCs
- ➢ Full integration of HPCs with other pledge resources without any manual interventions
- ➢ Trying advanced workflows like jumbo jobs and multi workers