BROOKHAVEN NATIONAL LABORATORY

Harvester
An edge service harvesting heterogeneous resources for ATLAS

T Maeno, F H Barreiro Megino, D Benjamin, D Cameron, J T Childers, K De, A De Salvo, A Filipcic, J R Hover, F Lin, D Oleynik, on behalf of ATLAS collaboration

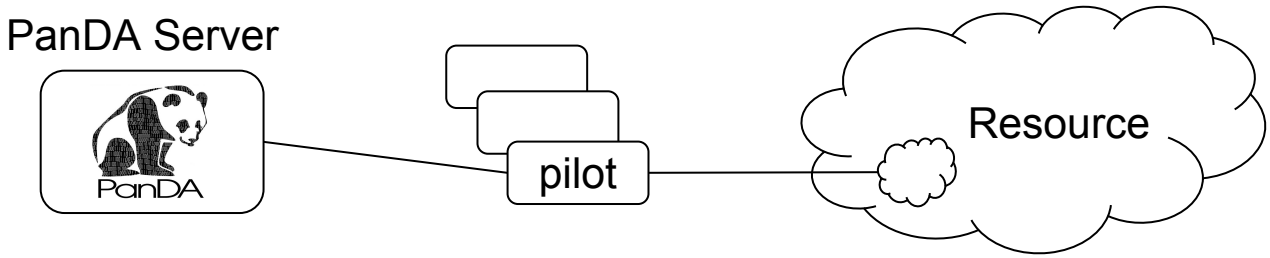
Brookhaven National Laboratory, USA
University of Texas at Arlington, USA
Duke University, USA
University of Oslo, Norway
Argonne National Laboratory, USA
Sapienza Universita e INFN, Roma I, Italy
Jozef Stefan Institute, Slovenia
Academia Sinica, Taiwan

CHEP2018, 9-13 July 2018, Sofia, Bulgaria

Introduction

BROOKHAVEN NATIONAL LABORATORY

- PanDA used to rely on server-pilot paradigm
  - Worked well for the grid with 250k cores 24x7 as underlying resources are not very heterogeneous



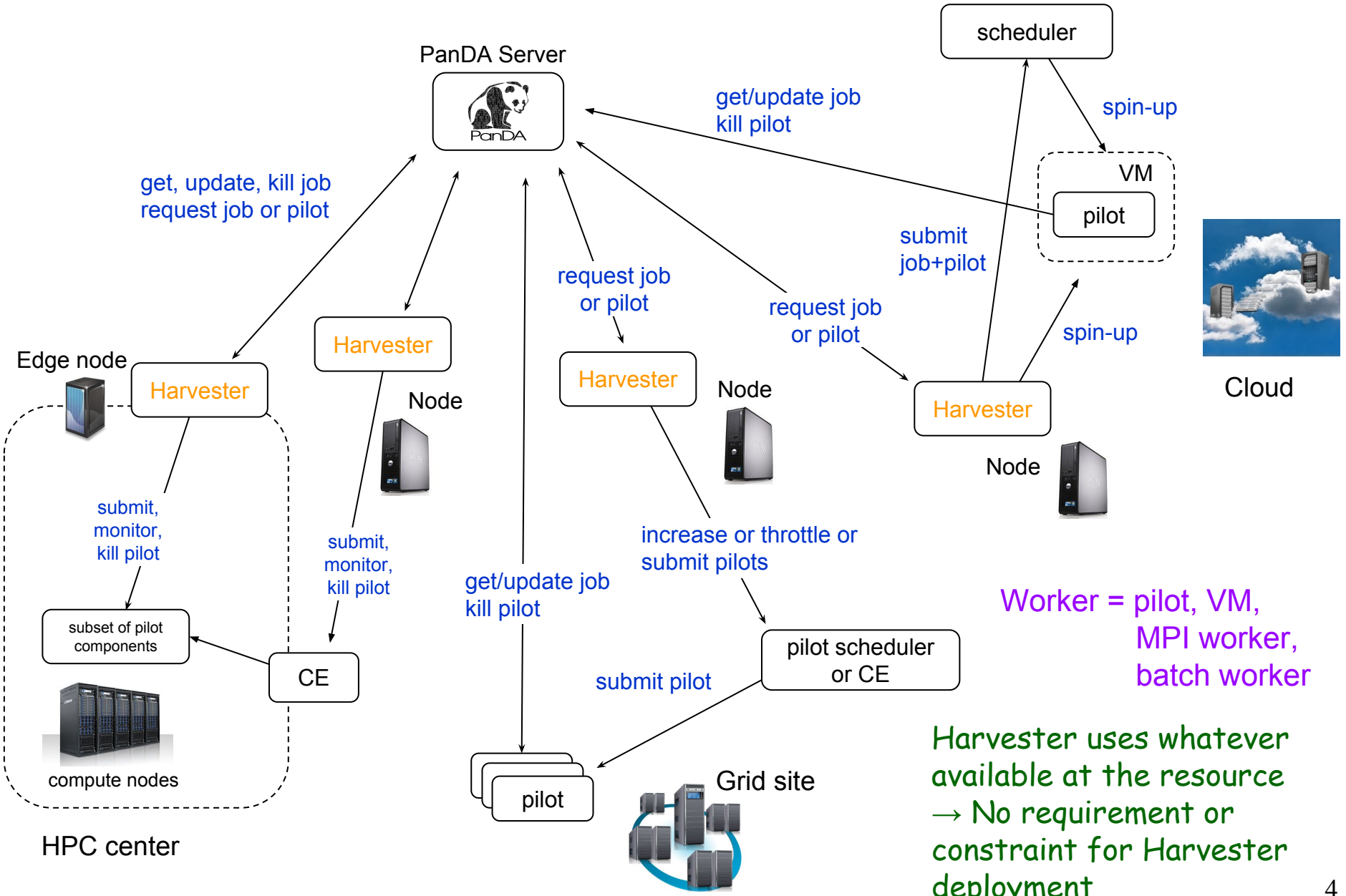
- Not very well for opportunistic resources, especially for HPCs
  - A different edge service and operational policy at each HPC center, leading to over-stretched pilot architecture and incoherence in implementation at different HPCs
  - Too many manual interventions to effectively fill available CPU resources at all HPC centers
- New project launched in Dec 2016 to address the issue

Introduction (cont'd)

BROOKHAVEN NATIONAL LABORATORY

- New model : server-harvester-pilot
  - Harvester is a resource-facing service between PanDA server and collection of pilots (workers)
  - Stateless service plus database (sqlite or MariaDB) for local bookkeeping
  - Modular design for different resource types and workflows
  - Many harvester instances running in parallel
- Objectives
  - A common machinery for pilot provisioning on all computing resources
  - Coherent implementations for HPCs
  - Timely optimization of CPU allocation among various resource types and removal of batch-level partitioning
  - Better resource monitoring
  - Tight integration between PanDA system and resources for new workflows

Schematic View



Worker = pilot, VM,
MPI worker,
batch worker

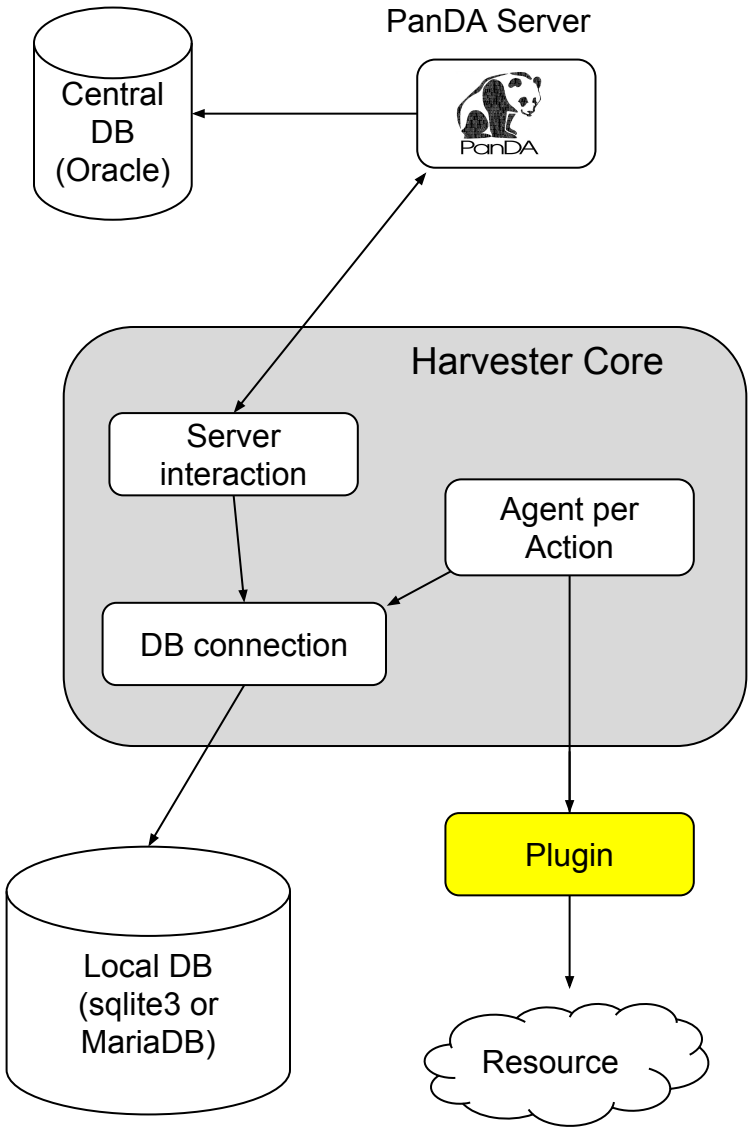
Harvester uses whatever available at the resource → No requirement or constraint for Harvester deployment

Main Functions of Harvester

BROOKHAVEN NATIONAL LABORATORY

- Submission and monitoring of workers
  - With batch systems, computing elements, submission services, orchestration services, ...
  - Utilization of real-time information on resources
- Communication with PanDA
  - Getting jobs and reporting their status
  - Sending requests and receiving commands
- Bridge service between PanDA and workers
  - Propagating heartbeats and commands
  - Dynamic optimization of payload size
- Data management
  - Asynchronous pre-staging and shipping-out of data
- Cleanup
  - Disk cleanup, database cleanup, deletion or killing of (orphaned) workers

Harvester Architecture



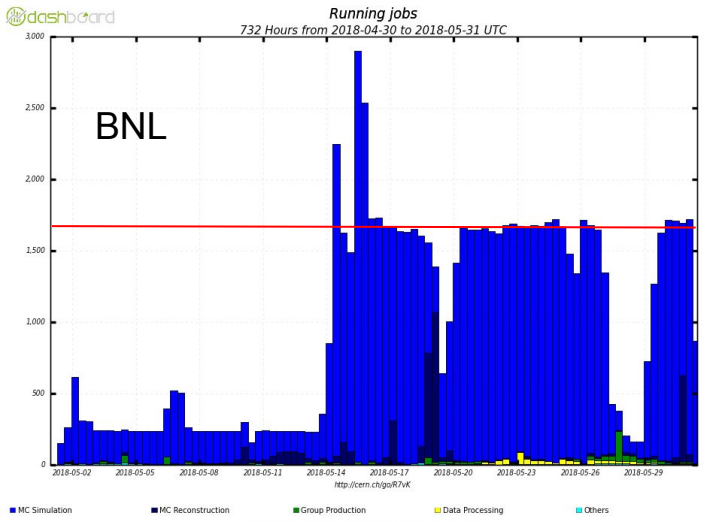
- Central + local DBs for scalability
- Multiprocessing, multithreading, multi-nodes
- One core component (agent) per action
  - Action : job fetching, pre-staging, worker submission, …
- Taking actions based on transition of job/worker status in local DB
- No direct communication (messaging) between agents
- Plugins developed by each resource experts

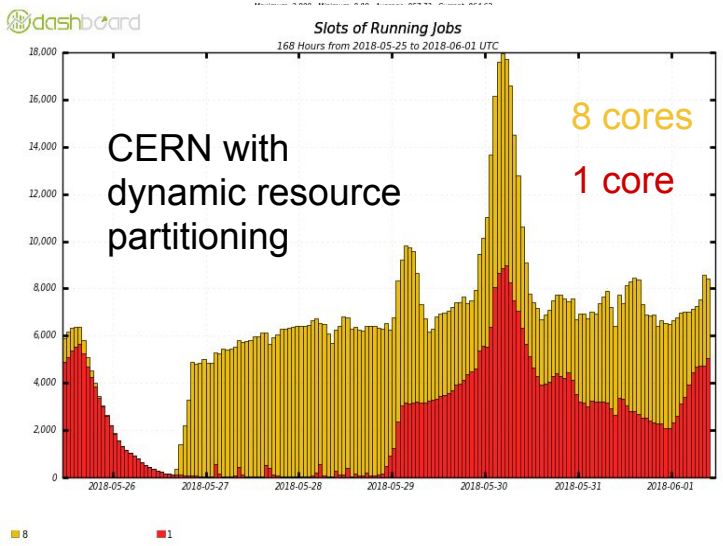
Current Status

The Grid

- Status
  - Ongoing migration for large scale production at CERN, BNL and Taiwan
  - Evaluating two submission engines, condor-C and aCT
  - Migration of runtime tests for ATLAS offline software to harvester due to intermittent workload submission
- Goals
  - Full migration to Harvester as a single mechanism for pilot provisioning on all ATLAS grid resources
  - Dynamic resource partitioning based on current physics needs while getting rid of static batch-level partitioning
    - Details in CHEP poster : F H Barreiro Megino, ATLAS Global Shares Implementation in the PanDA Workload Management System
  - Better site description for more optimal resource usage

Running jobs at BNL and CERN harvester queues

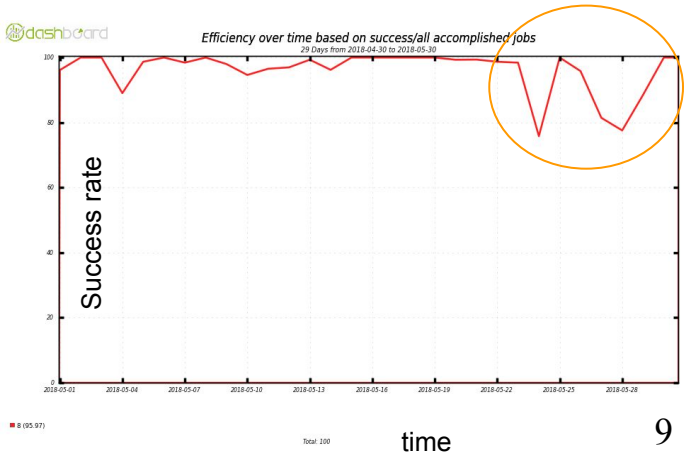




Cloud

- Production status
  - Condor-based cloud resources at CERN + Leibniz + Edinburgh with 1.2k CPU cores in production
- Two major developments
  - Condor-based for ATLAS High Level Trigger (HLT) CPU farm with 50k cores, aka Sim@P1
    - Resource availability depending on needs for the original usage
    - Proactive assignment of workload to the resource for quick ramp up before the resource becomes available
    - Details in CHEP Talk : F Berghaus, Sim@P1: Using Cloudscheduler for offline processing on the ATLAS HLT farm
  - Using native cloud API for GCE and EC2
    - Use-cases
      - In context of the data ocean project with GCE + GCE API + Google Storage + preemptible VMs
      - Openstack instance with EC2 API at Taiwan for non-ATLAS
    - New plugins with Kubernetes to be developed

Effect of switching from normal VMS to preemptible VMs on GCE

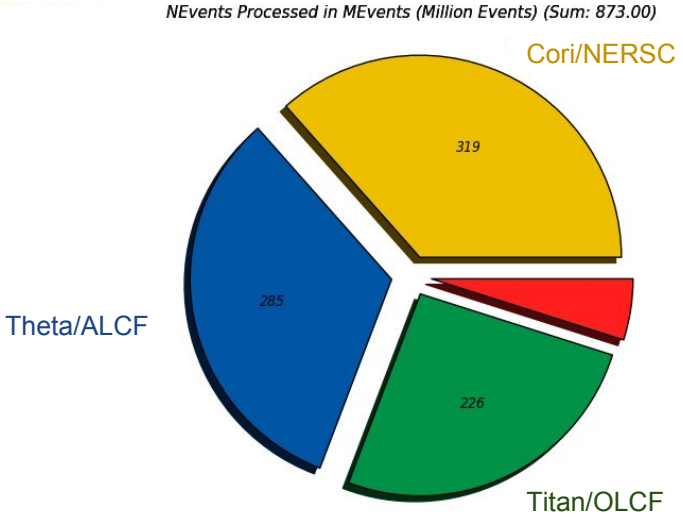


HPC 1/2

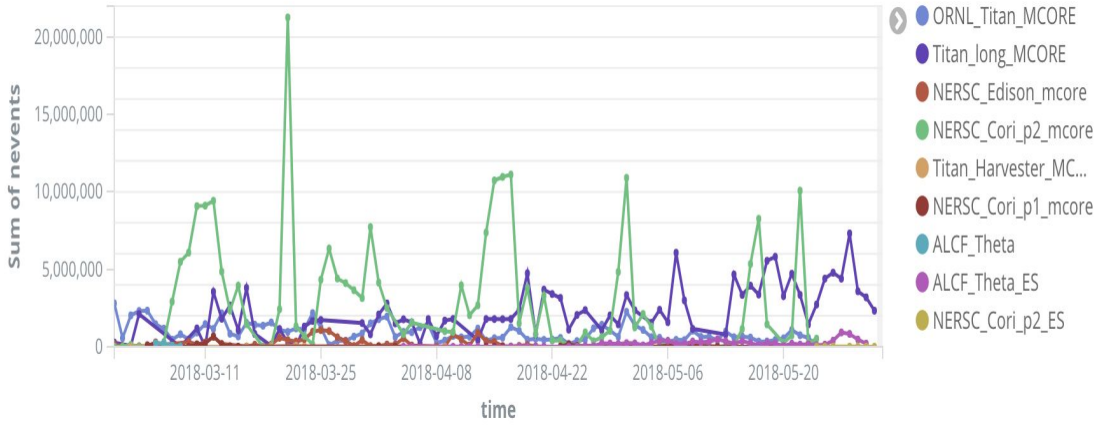
BROOKHAVEN NATIONAL LABORATORY

- Production status
  - US DOE HPCs
    - In production with a mechanism to dynamically combine many PanDA jobs to a single batch submission
    - Theta/ALCF, Titan/OLCF, Cori/NERSC

The total number of events (in M events) processed for ATLAS simulation production for last 6 months at US DOE HPCs



The number of events processed per day at US DOE HPCs for last 3 months



availability is intermittent

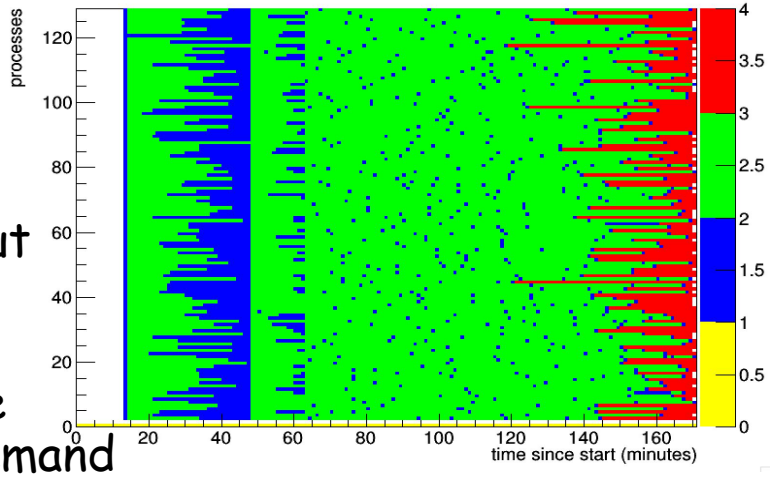
rce

  - ASGC
    - In production for non-ATLAS experiments
  - EU or US NSF HPCs
    - Under discussion

HPC 2/2

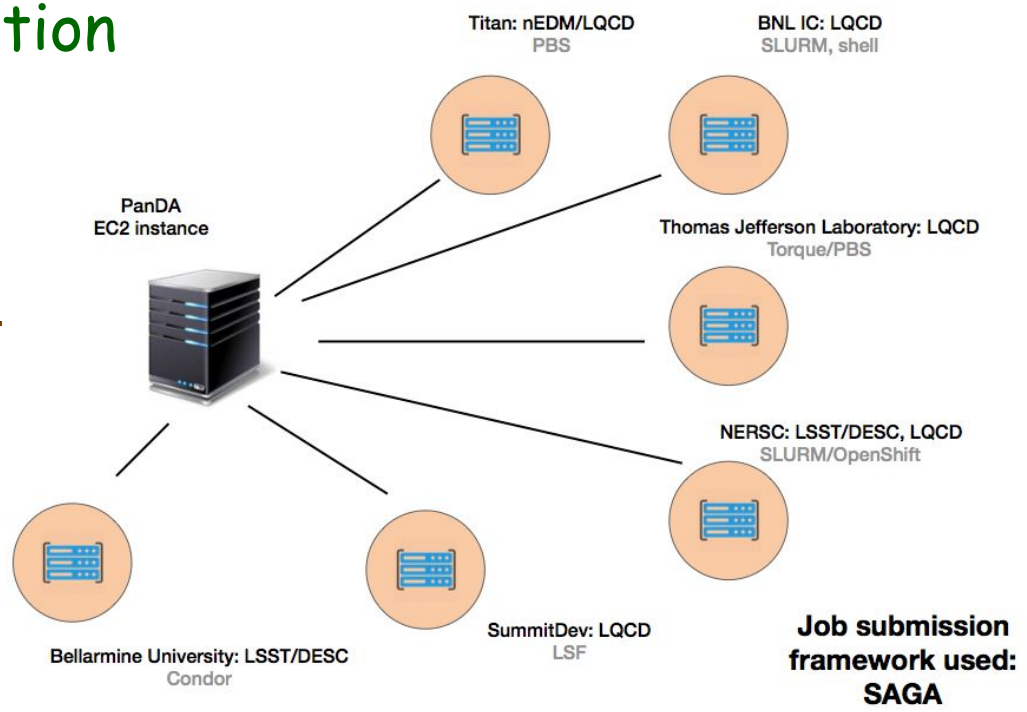
- Current developments
  - Jumbo payload with event service on HPC, a.k.a Yoda
    - Local site policies drive the need for large payloads
    - Event-level bookkeeping as a protection against early termination due to preemption and/or inaccurate estimation of execution
    - In validation
  - HPC + computing element
    - Integration of HPC resources to the grid with HTCondor or ARC CE
  - Dynamic payload sizing based on real-time information from HPC batch system
    - Usage of preemptible resources without a capability of event-level bookkeeping
  - Caching
    - Streaming service + local cache service to deliver data to compute nodes on demand
    - To be coherent with developments for ATLAS Event Streaming service which is reported in a CHEP talk : N Magini, Towards an Event Streaming Service for ATLAS data processing

A Yoda job at Theta/ALCF with 16k cores
Idle, Processing events which completed,
Processing events which didn't complete



Beyond ATLAS

- 6 instances of harvester configured and ready to use for non-ATLAS experiment in BigPanDA project
  - One regional instance at Thomas Jefferson Lab maintained by TJLab team
- Tested for nEDM, LQCD, LSST, also with Next Generation Executer (NGE)
- The first LQCD production successful at BNL
- Details in CHEP talk :
  P Svirin, PanDA and RADICAL-Pilot Integration: Enabling the Pilot Paradigm on HPC Resources



Plans

- Migration of the entire ATLAS grid to harvester
- Dynamic resource partitioning based on current physics needs in ATLAS
- Seamless integration of HPCs with other resources without any manual interventions by using jumbo payload + Yoda
- Full integration of ATLAS HLT farm to harvester with proactive workload assignment
- Expanding harvester usage beyond ATLAS

Summary

- Harvester project was launched in Dec 2016
  - Wide collaboration of resource and PanDA experts
- Many development activities in parallel with various resources
  - Coherent implementations to meet different requirements
- Already in production for various resources
- Still a lot of challenges to come
  - Further optimization and automation