What Do We (want to) Measure and Why?

Miron Livny
John P. Morgridge Professor of Computer Science
UW Center for High Throughput Computing
Morgridge Institute for Research

CENTER FOR HIGH THROUGHPUT COMPUTING

OSG

PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING

Thank you for joining the 7th

European  HTCondor Software Suite workshop

(year 37 of the (HT)Condor project)

(year 15 of the CHTC)

CENTER FOR HIGH THROUGHPUT COMPUTING

OSG

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING

PATh project - a Partnership between UW-Madison Center for High Throughput Computing (**CHTC**) and **OSG** Consortium to advance **Throughput Computing**

HTCondor Software Suite Software Suite (HTCSS) manages High Throughput workloads across all forms of research computing resources from campus clusters to commercial clouds and HPC facilities

OSG services enable science collaborations and campuses to build and operate private distributed computing environments across >70 sites that deliver over **2B** core hours annually

LSC LIGO Scientific Collaboration

Office of Research HOLLAND COMPUTING CENTER

Fermilab

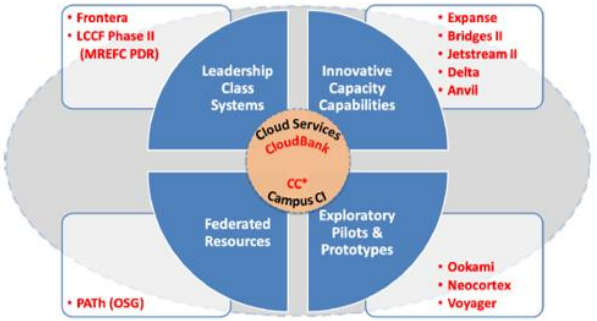
IceCube South Pole Neutrino Observatory

Jefferson Lab

CMS CERN The CMS Experiment at CERN

GlueX

ATLAS EXPERIMENT



Aligned with NSF CI Eco-system

Campus Cyberinfrastructure (CC*)

PROGRAM SOLICITATION
NSF 21-528

REPLACES DOCUMENT(S):
NSF 20-507

National Science Foundation

Directorate for Computer and Information Science and Engineering
Office of Advanced Cyberinfrastructure
Division of Computer and Network Systems

Proposals are required to commit to a minimum of 20% shared time on the cluster and describe their approach to making the cluster available as a shared resource external to the campus, with access and authorization according to local administrative policy. Conversely, the proposal should describe the approach to providing **on-demand** access to additional external computing resources for its targeted on-campus users and projects. One possible approach to implementing such a federated distributed computing solution is joining a multi-campus or national federated system such as the Open Science Grid. Whatever opportunistic, federated, scalable, distributed computing platform is

CENTER FOR HIGH THROUGHPUT COMPUTING

OSG

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING

Cyberinfrastructure for Sustained Scientific Innovation (CSSI)

PROGRAM SOLICITATION
NSF 21-617

REPLACES DOCUMENT(S):
NSF 20-592

National Science Foundation

Directorate for Computer and Information Science and Engineering
Office of Advanced Cyberinfrastructure
Division of Computing and Communication Foundations
Division of Information and Intelligent Systems

High-Throughput Computing Resources

Proposals may request high-throughput computing (HTC) resources through the Partnership to Advance Throughput Computing (PATh) project supported by NSF.

Proposers should describe the request in a Supplementary Document no longer than two pages with a technical description of, and justification for, the requested HTC resources that includes (a) the expected number of self-contained tasks per ensemble – note that each task can be packaged into one or more batch job; (b) the resource requirements for each task type in the ensemble – for example, requirements for cores, memory, wall-time, and scratch space; (c) the expected number of ensembles; (d) the expected input and output data requirements for each task type; and (e) the expected number and size of shared input files within an ensemble – expected number of times each file is read per ensemble.

CENTER FOR HIGH THROUGHPUT COMPUTING

OSG

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING

One of the services in the OSG fabric of Distributed High Throughput Computing Services (dHTC) is a HTCondor pool that serves Open Science in the US. We refer to this pool as the Open Science Pool (**OSPool**)

- Organizations contribute capacity to the OSPool following local (autonomous) policies. 20 of these organizations are CC* awards.
- Any researcher with affiliation to a US research institute can gain “fair-share” access to the OSPool capacity via Access Points provided by PATh

OSPool (typical) Weekly Numbers:

- runs more then **2M** jobs
- placed by more than **70** users
- from more than **60** projects
- at more than **35** Access Points
- on more than **35K** cores
- at more than **50** sites
- that consume more than **4M** core hours

PATh monitors the OSPool because we

- are committed to maximize the impact of the OSPool capacity on open science,
- are accountable to the organizations that contribute their capacity to the OSPool,
- are accountable to the entities that fund us

CENTER FOR HIGH THROUGHPUT COMPUTING

OSG

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING

The information we collect is used to

- guide the PATh **Research Computing Facilitators** in their interaction with researchers
- pinpoint bugs, miss configurations, operational mistakes, system failures
- identify opportunities to advance functionality, features, documentation, data collection, user interfaces

Key to the impact of the OSPool monitoring effort is how quickly we can generate, collect and report the information needed to answer a question

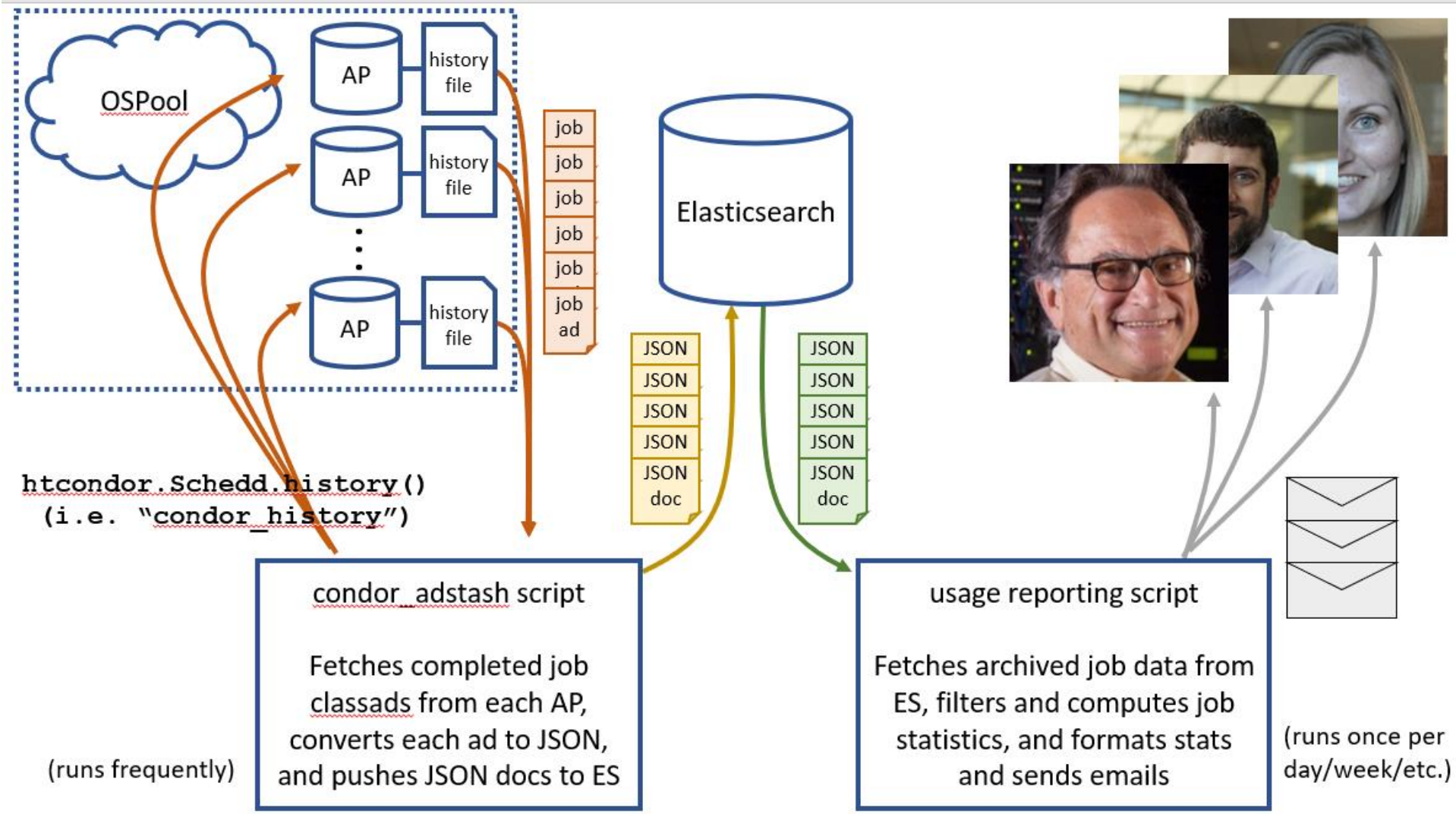
- PATh controls the OSPool software (HTCSS)
- PATh has partial control on the deployment and the operation of the GlideIns
- PATh is working on methodologies and procedures that shorten the “need to repot” time

More than 35 Access Points (SchedD) are recognized by the OSPool Collector and Matchmaker. More than 40 Compute Entrypoints (CE) are used to launch GlideIns

- Operated by different organizations
- Located at different administrative domains
- Using different HTCSS versions

Every morning at 7 am central time we mail out four daily reports that cover the jobs that completed the previous day

- A report for all jobs
- A report for removed jobs
- A report for held jobs
- A report for max running time jobs


OSPool
AP
history file
AP
history file
AP
history file
job
job
job
job
job
job ad
Elasticsearch
JSON
JSON
JSON
JSON
JSON doc
JSON
JSON
JSON
JSON
JSON doc
htcondor.Schedd.history()
(i.e. "condor_history")
condor_adstash script
Fetches completed job classads from each AP, converts each ad to JSON, and pushes JSON docs to ES
(runs frequently)
usage reporting script
Fetches archived job data from ES, filters and computes job statistics, and formats stats and sends emails
(runs once per day/week/etc.)

Understand Good-Put and Bad-Put

- Measure how much wall-clock time is devoted to the last (successful?) execution epoch of a job
- Profile execution epochs of a job – number and duration. Focus on wall-clock duration distribution (25%, 50%, 75%, 95%, max) of last epoch.

Understand Shadow invocations that do not lead to an execution

- Identify the different reasons/causes an invocation does not result in an execution
- Characterize the responsibilities for failures that cause "wasted" invocation
- Measure and record information on such invocations – failed to connect, failed to activate claim, failed to transfer input sandbox, failed to launch job executable, wall-clock used

Understand removed jobs

- Profile number and wall-clock consumptions of removed jobs
- Should removed jobs be considered Good-Put?
- Provide Research Computing Facilitators with information to follow up with researchers
- Can we automatically identify the reason for removal?

Understand Held Jobs

- Measure number of jobs that were placed on hold
- Profile the reasons for the hold
- Measure the number of times a job was released from hold
- Profile the reasons for the release
- Categorize reasons and assign responsiblities

Understand long running jobs

- Identify the longest running job for each user
- Profile the execution history of these jobs
- Quantify the impact of these jobs on Bad-Put

We maintain a (Ganglia) Dash-Board for GlideIns

- Information is provided by the Compute Entrypoints (CE) (note that more OSPool sites do not launch GlideIns via CEs)
- Measure running GlideIns
- Measure queued GlideIns

Understand missed capacity contributions

- Measure the “pressure” of GlideIn requests presented to the CEs
- Measure failed launching of GlideIns
- Measure delays in launching of GlideIns

Understand Waste-Put

- Measure wall-clock assigned to GlideIns but not claimed
- Measure GlideIns that were never claimed

The road ahead:

- Faster “need to report” turnaround times for the OSPool
- Less effort to translate a need to a report for the OSPool
- Improve robustness of monitoring infrastructure – report missed data prevent duplication
- Integrate the reporting infrastructure into HTCSS

Thank you for building

a thriving (d)HTC community

CENTER FOR HIGH THROUGHPUT COMPUTING

OSG

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING