Empowering Research Computing at Your Organization Througfrdoyle@stmichaelcp.orgh the Open Science Grid

PEARC21
Monday, July 19, 11am-2pm Eastern Time (Conversion here)

Schedule & Materials

Time	Topic	Materials
0:00 - 0:10	Welcome, Accounts	See below “Accounts for Hands-On”
0:10 - 0:30 (20 min)	Research on the OSG	Slides
0:30 - 1:15 (40 min)	Exercises	Tutorial: osg-locations (group) Breakouts • Submitting Multiple Jobs • Using Data / Blast • Using Containers • Creating a Software installation
1:20 - 1:30 (15 min)	Break	Working with Tensorflow, GPUs, and containers
1:25 - 1:55 (30 min)	**Facilitating** Research on the OSG	Slides
1:55 - 2:00 (5 min)	Break	
2:00 - 2:55 (55 min)	**Campus Engagement** with the OSG	Slides Panel: (AMNH, UCSD, Michigan)

Accounts for Hands-On

To request an account, follow the instructions under “Sign in to or create an account” here: https://support.opensciencegrid.org/support/solutions/articles/5000632072-registration-and-login-for-osg-connect#sign-in-to-or-create-an-account

Please indicate “PEARC21 tutorial” in the comments for your account request, and then immediately add your SSH key to your account in the OSG Connect website. We will approve accounts before the first tutorial.

Sign In

- Name / Institution / Follow-up email

OSG Staff

- Christina Koch / UW - Madison / ckoch5@wisc.edu
- Lauren Michael / UW-Madison / lmichael@wisc.edu
- Emelie Fuchs / UNL / efuchs@unl.edu
- Mats Rynge / USC / rynge@isi.edu

Participants

- Jason Miller, Shepherd University, jmill02@shepherd.edu
- Shelly Johnson, University of Michigan, jshelly@umich.edu
- Trevor Cooper, San Diego Supercomputer Center, tcooper@sdsc.edu
- Kimberly Thomas, University of California at San Diego, kkt008@ucsd.edu
- Kirk M. Anne / Rochester Institute of Technology / kirk.m.anne@rit.edu
- Tony Weaver / Franklin and Marshall College / aweaver1@fandm.edu
- Tom Milledge / Duke University / tom.milledge@duke.edu
- Kevin Bryan, University of Rhode Island, bryank@uri.edu
- Feng Chen / Louisiana State University, fchen14@lsu.edu
- Wirawan Purwanto / Old Dominion University / wpurwant@odu.edu
- August Hays-Ekeland, Arizona State University ahayseke@asu.edu
- Andy Anderson, Amherst College, aanderson@amherst.edu

Panelists

- Bennet Fauber <bennet@umich.edu> University of Michigan
- Sajesh Singh <ssingh@amnh.org> American Museum of Natural History
- Alan Moxley

Links and Notes

Commands to get started with the tutorial:

```
$ tutorial list
$ tutorial osg-locations
$ cd tutorial-osg-locations
$ condor_submit scalingup.submit
```

To look at jobs, use “condor_q” or “condor_watch_q”
Once jobs are done, run this command:

```
$ cat job.*.output | sort -u
```

And paste the output into https://www.mapcustomizer.com/ after clicking on “Bulk Entry” on the right.

https://support.opensciencegrid.org/support/solutions/articles/12000078187-using-julia-on-the-osg

Things Learned/Questions

At first break: what is one thing you learned and one question you have?

- Jason M - Condor has great support for repeating some action on many files. Q: can a condor job submit a condor job?
    - A(Lauren): sometimes... Generally, what people are looking for when they ask this question is: “How can I have jobs submitted/run in a particular order?” **HTCondor has a built-in workflow tool called DAGMan** (and the less HTCondor-specific Pegasus workflow tool also uses it). Jobs within a DAG can be specified to run in a particular order, and only if prior jobs succeeded (zero exit code). Technically, submitting a DAG results in a job that runs on the access point and submits the jobs defined in the DAG workflow. DAGMan YouTube intro: https://www.youtube.com/watch?v=1MvVHxRs7iU
    - Christina also mentioned **self-checkpointing**, where a job that needs to run with sequential work that is longer than the OSPool’s ideal (10 hrs) can exit after writing a checkpoint, and then HTCondor can start the executable again (maybe in a new compute slot) with the checkpoint file(s) present.
    - We (OSG, or for UW-Madison’s HTCondor pool) generally DON’T want people implementing other mechanisms for jobs submitting jobs (or even scripts that submit numerous jobs instead of using native HTCondor submit file ‘queue’ statements), as they almost always go awry and cause major issues for everyone involved with the access point.
- Jason M - Is there a ‘module’ command that sets up a particular software environment?
    - **The OSPool has *some* softwares support via modules** (in the OSG User Documentation). Generally, bringing along your software (static binary, .tar.gz of the install folder, container, etc.) is more robust than relying on modules, in terms of dependency-handling across diverse sites. There are certain softwares for which those other options just don’t work well (e.g. code/database/library too large for a container, etc.), so modules are more of a last-resort/exception and not a ‘first line’ for software portability in the OSPool.
- Shelly J - GPU jobs have their own special requirements. I like the *condor_watch_q* command.
- Trevor C - In my test job in the .log output file there are a number of metrics reported. This is great, is there a way to collect even more metrics and add them to this report?
    - A(Lauren): Yes! While jobs are queued (haven’t completed and are still in the queue), there are additional options to ‘**condor_q**’ (like ‘condor_q <job or cluster

#> -l' , which will display ALL attributes that HTCondor stores about the job). Using '**condor_history**' can list the same for jobs that recently completed and left the queue (e.g. 'condor_history <job or cluster #>'. The '-l' option will list ALL attributes, and you can list specific attributes with '-af' (autoformat). For example: 'condor_history <cluster#> -af ProcId MemoryUsage' will show the memory usage (MBs) for all jobs of that cluster that have already left the queue. See also: full-length HTCondor User Tutorial video, starting here (31min in): https://youtu.be/p2X6s_7e51k?t=1882

- Kimberly T - testing a container image to ensure that it works
- Kirk M
- Tony W -- "ideal" jobs
- Tom M
- Kevin B - distributed environment uses directory form of singularity images, likely for caching purposes; How do errors get reported if it seems like a node configuration issue? (Received: "tensorflow.python.framework.errors_impl.InternalError: CUDA runtime implicit initialization on GPU:0 failed. Status: device kernel image is invalid" using tutorial for GPU)
- Feng C
- Wirawan P - An overview of running jobs on OSG. How do you help researchers cope with enormously large number of files? Examples:
    - Where calculations succeeded, or failed
        - A(Lauren):User can run **condor_history** commands to see the 'ExitCode' recorded from each job's executable: condor_history <clusterID> -af ProcID ExitCode
          (**See also the answer to Trever, above**.)
            - Or, if the user's executable didn't produce a non-zero exit code, reviewing output files is the best way. (Using 'ls' and comparing sizes can be useful, but will really depend on the user's particular output.)
    - Any recommendations for storing, analyzing large number of files?
        - A(Lauren): the OSG Connect access points use local disk for /home, to better handle large numbers of files and because a network/shared filesystem (that would struggle) is not necessary. Additionally, the HTCondor submit file allows for job files to be organized into subdirectories, according to the user's preferred data organization. It's rare that we have users with files so numerous that they have to change to more of a tree hierarchy to avoid filesystem issues, when they don't already think of and organize their jobs in such a tree structure.
        -
- Andy — methods for transferring files to remote systems (since I'm only used to using a shared network file system with Condor).
    - Q: How to impose memory usage limits by individual jobs?
- August - If you have lots of short TensorFlow jobs you should run it on CPUs because there aren't very many GPUs available.

    - Lauren: Depends, a bit. If your TensorFlow jobs can run on ANY GPU, there are a LOT in the OSPool, and the performance difference between CPUs and GPUs may direct you to one option versus the other (e.g., if your CPU-based ML jobs would be too long, >10hrs on average).

Question from TrevorK: Is the **algorithm for calculating priority** documented?
Answer from the HTCondorTeam:
https://htcondor.readthedocs.io/en/latest/admin-manual/user-priorities-negotiation.html

- This is perfect. The case will be users wanting to know why their submitted jobs are still IDLE for any given amount of time. This provides the raw information needed to help user understand scheduling priority.
- Lauren: In a large HTC system like the OSPool, a user can have up to thousands of cores in use, even with low priority, IF their jobs require less resources than other jobs. So, fair-share applies most directly between users with similarly-sized jobs. We rarely see that a user has NO jobs running for hours, unless each job is on the VERY large size for the OSPool. With single-core, laptop-sized tasks, the user will have dozens of jobs running in the first several minutes, hundreds within hours, and up to thousands within a day of submitting.

Closing questions

- What are the benefits to specifically supporting researchers with HTC-type workflows in using a system like the OSG?
- What challenges do you anticipate in introducing researchers to the OSG (or high throughput computing approaches generally)?
- What else, beyond your current resources, would you need to support researchers using HTC approaches and/or the OSG?
- Based on the above, what is at least ONE goal and ONE concrete action that you can take in the next month?