Introduction to OSG

Tim Cartwright

University of Wisconsin–Madison

OSG School Director

OSG Deputy Executive Director

OSG Campus Coordinator

Overview

So far, we have seen how to use HTC on one cluster

Sometimes, that is not enough!

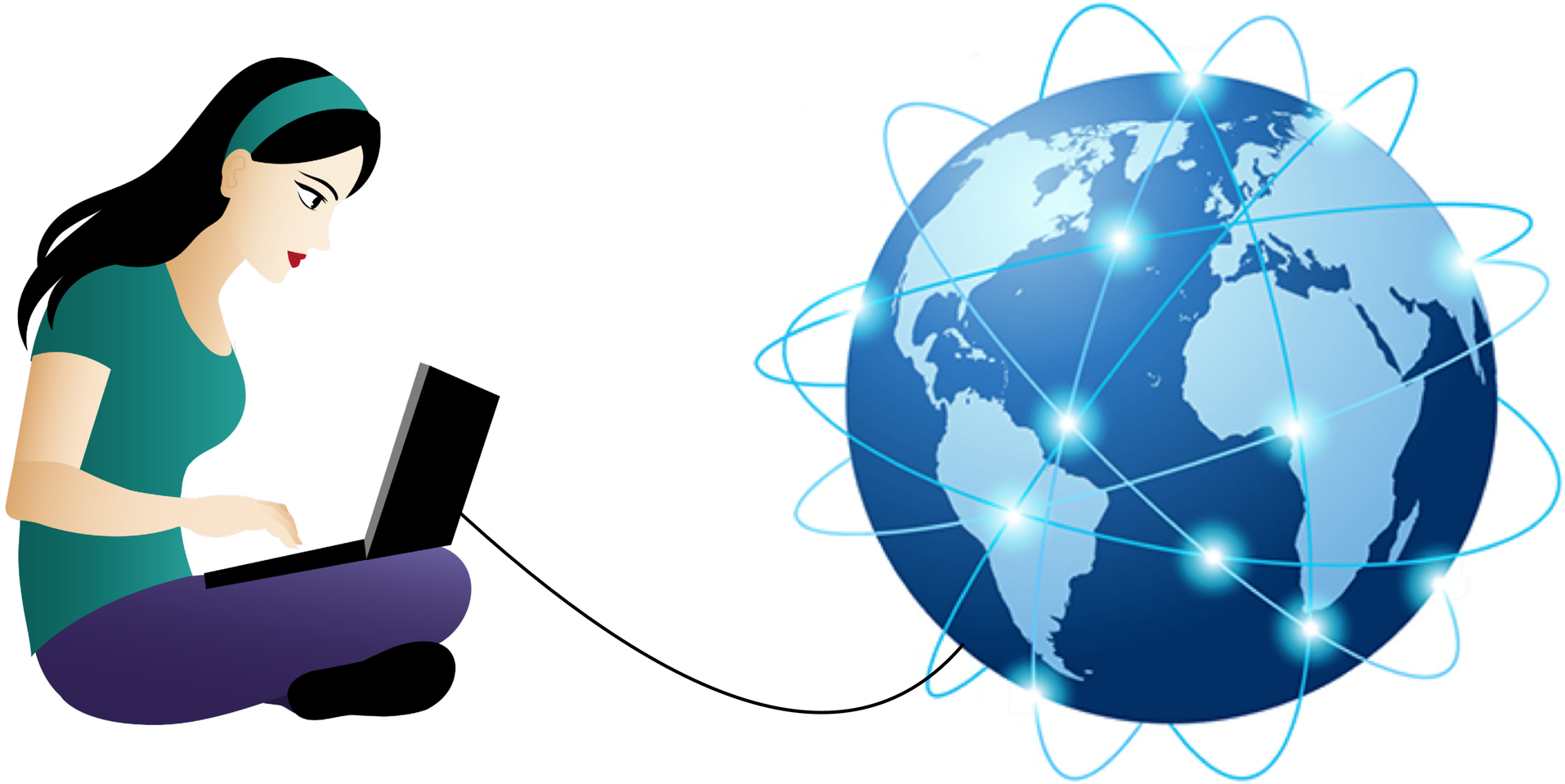
(Don't let computing be a barrier to research, remember?)

Today, we see what it takes to get more resources *

* ***Caveat:*** *I will focus on compute resources; Wed. will focus on data.*

What Researchers Want

Submit locally, run globally



What That Entails

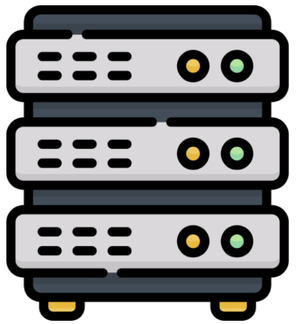
- Lots of networked computing resources
  - Most OSPool resources are contributed!
  - (PATh Facility, used yesterday, is owned by PATh)
- Resource owners are motivated to share
- Unified purpose *and* local autonomy
- Automation to make it work at scale
- Free would be nice!

Yesterday

Access Point

```
$ ls
science.sub
input.dat
$ condor_q
```

PATh



https://sweetclipart.com/
https://www.flaticon.com/free-icons/server

The Goal

Access Point

```
$ ls
science.sub
input.dat
$ condor_q
```

Syracuse

PATh

Nebraska

UCSD

https://sweetclipart.com/
https://www.flaticon.com/free-icons/server

Demo, Part I

Behind the Curtain

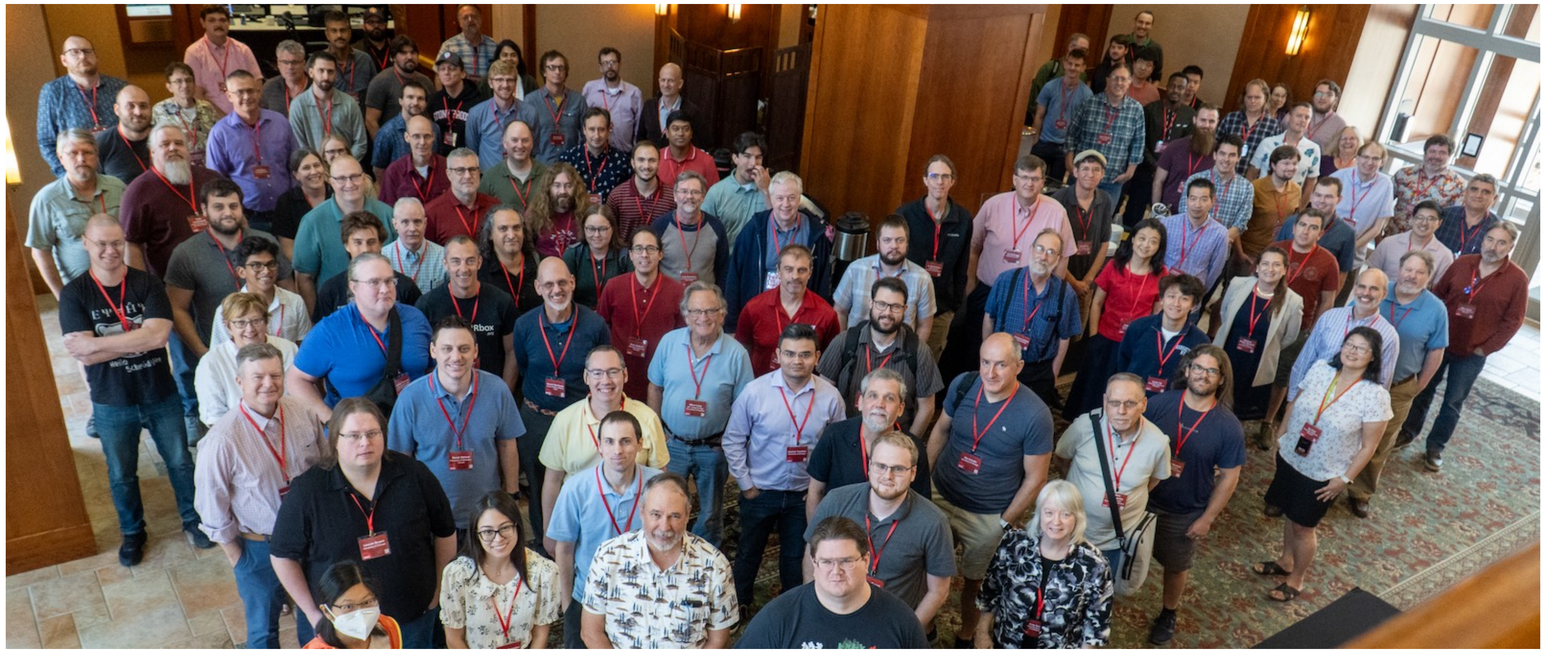
Reasons for Continuing

- So why learn more about OSG and the OSPool?
  - May change how you plan to run computing there
  - May change the way you use the Access Point
  - May change how you handle issues that arise
- What is there to learn? (outline of rest of talk)
  - Concepts of OSG, Pool, and Access Point
  - How the OSPool gets resources
  - How the OSPool differs from a local cluster

What is OSG?

OSG Defined, Version 1

- OSG Consortium – in this view, OSG ***is*** people:
  - Users: individuals PI/students to collabs. of 1,000s
  - Resource owners/contributors
  - Team: provides infrastructure, support, features, …



OSG Defined, Version 2

- Pools of resources
  - **Resources:** compute, storage, and other systems that can be used for research workflows
  - **Services:** software infrastructure that manages resources and makes features available
  - **Access Points:** where users go

https://www.pngall.com/wp-content/uploads/5/Server-Rack-PNG-Free-Image.png

OSG Defined, Version 3

- OSG Access Point
  - Where you go to do computing
  - Has access to resources (constantly changing)
  - Provides means for accessing data (see Wednesday)

```
[tim.cartwright@ap40 ~]$ condor_version
$CondorVersion: 10.7.0 2023-07-10 BuildID: 659788 PackageID: 10.7.0-0.659788 RC $
$CondorPlatform: x86_64_AlmaLinux8 $
[tim.cartwright@ap40 ~]$
```

Getting Resources for OSPool

The Goal

Access Point

```
$ ls
science.sub
input.dat
$ condor_q
```

Syracuse

PATh

Nebraska

UCSD

https://sweetclipart.com/
https://www.flaticon.com/free-icons/server

The Goal

Access Point

```
$ ls
science.sub
input.dat
$ condor_q
```

Syracuse

PATh

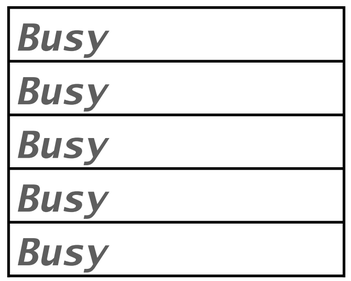
Nebraska

UCSD

OSPool dHTC Diagram

1. Before OSPool

Nothing available at Wisc. 😭



Access Point

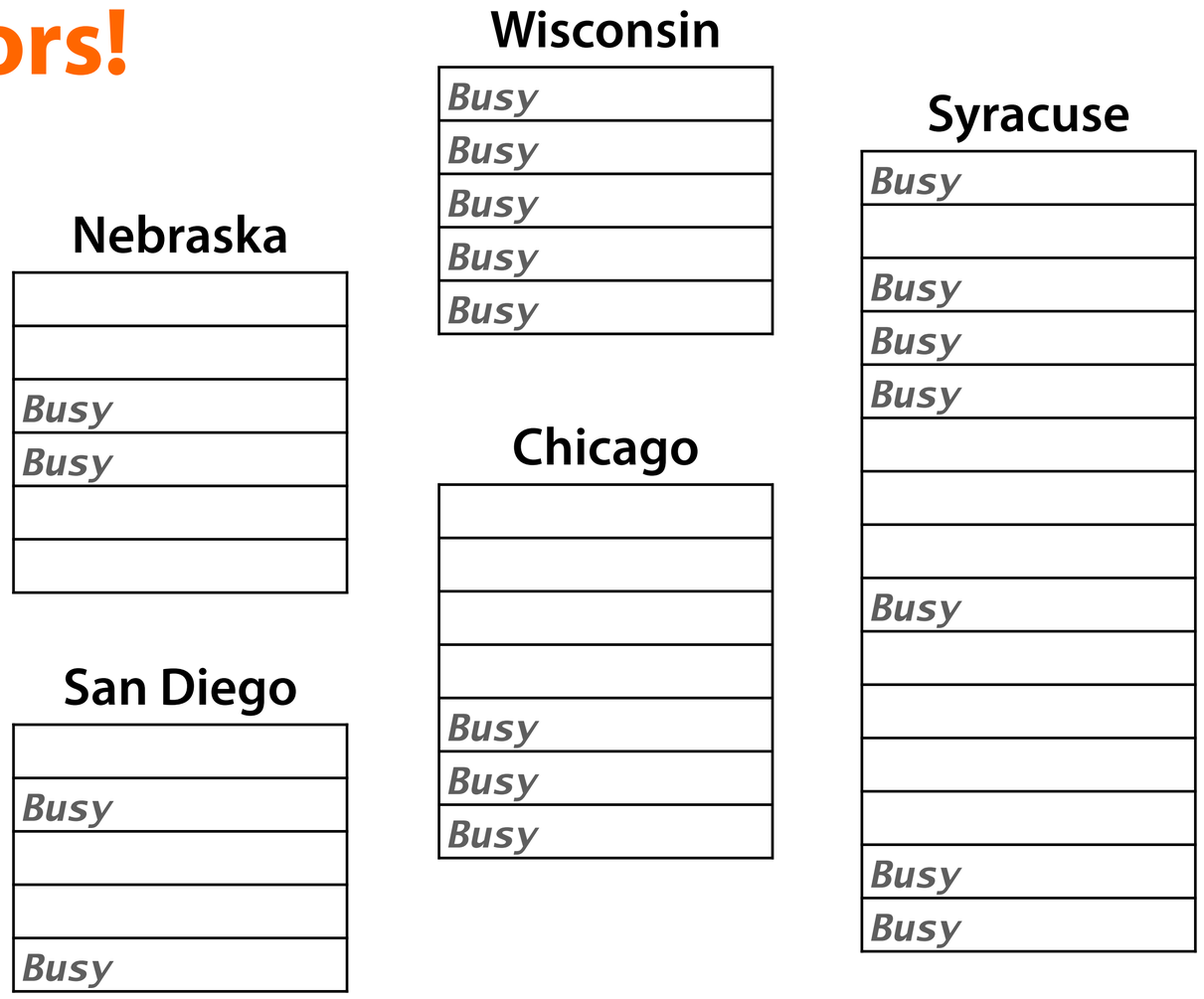
```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

OSPool dHTC Diagram

2. Add resource contributors!

Access Point

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```



OSPool dHTC Diagram

3. Get resources (method #1)

Try to run “glideins” at sites

Access Point

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

Nebraska

OSPool GI NU1
OSPool GI NU2
Busy
Busy

Wisconsin

Busy
Busy
Busy
Busy
Busy

Syracuse

Busy
OSPool GI SU1
Busy
Busy
Busy
OSPool GI SU4
OSPool GI SU2
OSPool GI SU3
Busy
Busy
Busy

San Diego

OSPool GI SD2
Busy
OSPool GI SD1
OSPool GI SD3
Busy

Chicago

OSPool GI UC2
OSPool GI UC1
Busy
Busy
Busy

OSPool dHTC Diagram

4. Glideins add resources to Pool

(I am not explaining how yet)

AP

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

Pool

OSPool GI NU1	*idle*
OSPool GI NU2	*idle*
OSPool GI SD1	*idle*
OSPool GI SD2	*idle*
OSPool GI SD3	*idle*
OSPool GI UC1	*idle*
OSPool GI UC2	*idle*
OSPool GI SU1	*idle*
OSPool GI SU2	*idle*
OSPool GI SU3	*idle*
OSPool GI SU4	*idle*

Nebraska

OSPool GI NU1
OSPool GI NU2
Busy
Busy

San Diego

OSPool GI SD2
Busy
OSPool GI SD1
OSPool GI SD3
Busy

Wisconsin

Busy
Busy
Busy
Busy
Busy

Chicago

OSPool GI UC2
OSPool GI UC1
Busy
Busy
Busy

Syracuse

Busy
OSPool GI SU1
Busy
Busy
Busy
OSPool GI SU4
OSPool GI SU2
OSPool GI SU3
Busy
Busy
Busy

OSPool dHTC Diagram

5. Run jobs

HTCondor with AP & Pool

AP

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

Pool

OSPool GI NU1	Job1.4
OSPool GI NU2	*idle*
OSPool GI SD1	Job1.0
OSPool GI SD2	Job1.3
OSPool GI SD3	*idle*
OSPool GI UC1	Job1.2
OSPool GI UC2	Job1.6
OSPool GI SU1	Job1.8
OSPool GI SU2	Job1.12
OSPool GI SU3	Job1.10
OSPool GI SU4	*idle*

Nebraska

NU1 > Job1.4
NU2 > *idle*
Busy
Busy

San Diego

SD2 > Job1.3
Busy
SD1 > Job1.0
SD3 > *idle*
Busy

Wisconsin

Busy
Busy
Busy
Busy
Busy

Chicago

UC2 > Job1.6
UC1 > Job1.2
Busy
Busy
Busy

Syracuse

Busy
SU1 > Job1.8
Busy
Busy
Busy
SU4 > *idle*
SU2 > Job1.12
SU3 > Job1.10
Busy
Busy
Busy

OSPool dHTC Diagram

6. Get resources (method #2)

Direct contributions

Queue
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999

Pool	
OSPool GI NU1	Job1.4
OSPool GI NU2	*idle*
OSPool GI SD1	Job1.0
OSPool GI SD2	Job1.3
OSPool GI SD3	*idle*
OSPool GI UC1	Job1.2
OSPool GI UC2	Job1.6
OSPool GI SU1	Job1.8
OSPool GI SU2	Job1.12
OSPool GI SU3	Job1.10
OSPool GI SU4	*idle*
SU Contrb OG1	*idle*

Nebraska

NU1 > Job1.4
NU2 > *idle*
Busy
Busy

San Diego

SD2 > Job1.3
Busy
SD1 > Job1.0
SD3 > *idle*
Busy

Wisconsin

Busy
Busy
Busy
Busy
Busy

Chicago

UC2 > Job1.6
UC1 > Job1.2
Busy
Busy
Busy

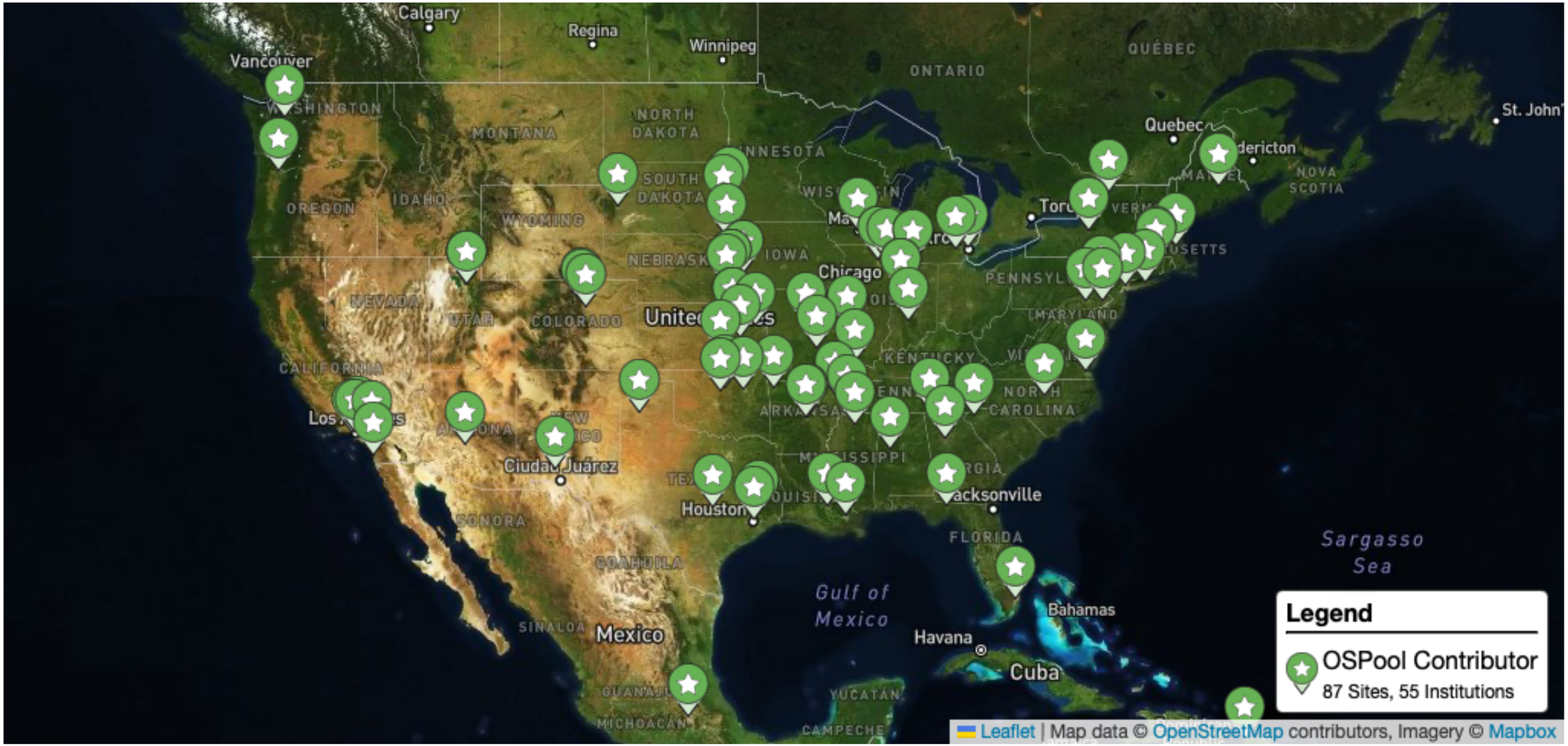
Syracuse

Busy
SU1 > Job1.8
Busy
Busy
Busy
SU4 > *idle*
SU2 > Job1.12
SU3 > Job1.10
Busy
SU Contrb OG1
Busy
Busy

OSPool dHTC – A Few Details

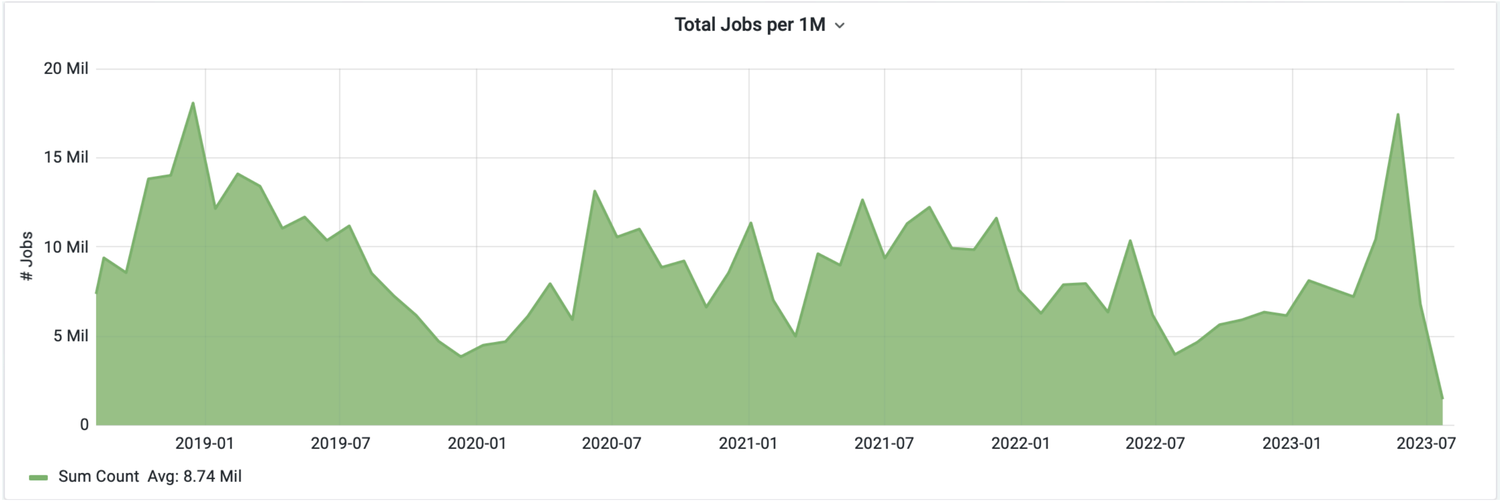
- A few details on the ***Glidein*** system
  - OSG submits glideins to contributing sites
  - Glideins are submitted based on *idle* OSPool jobs
  - Local site policy determines when to run glideins
  - A glidein creates an execution environment (EP), and reports back to central services
  - Glideins (hence capacity) go away when not needed
- OSG/HTCondor manage & automate the details!

OSPool Contributors *(United States)*

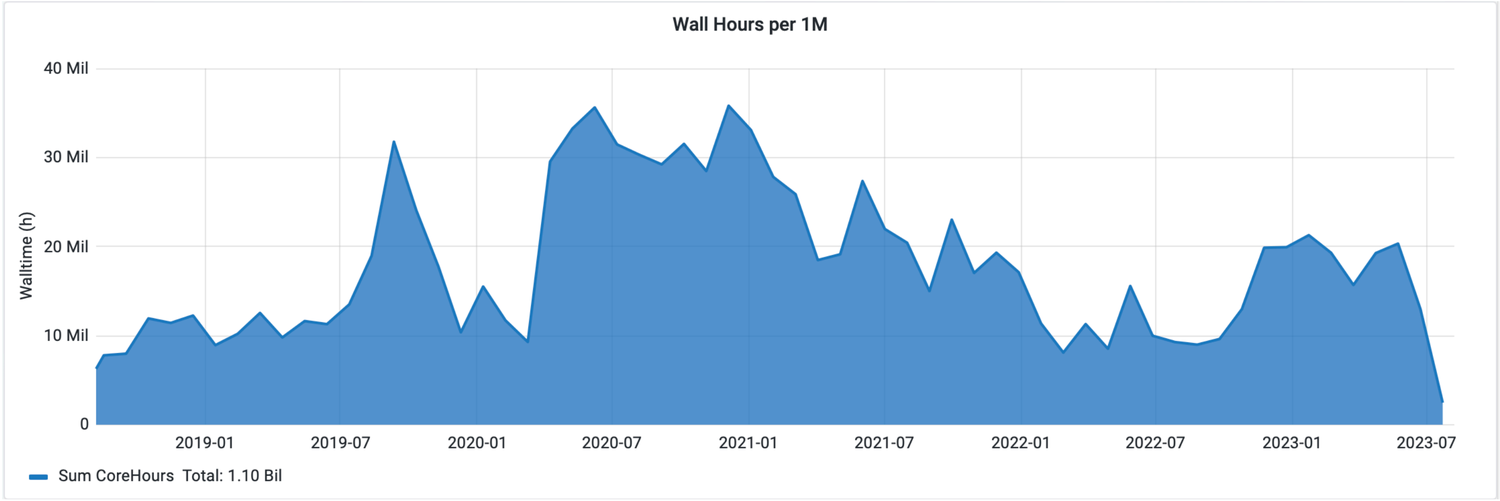


OSPool Usage

Jobs



Hours



Using OSPool

OSPool Is HTCondor

- OSPool is an HTCondor pool:
  You have condor_q, condor_submit, DAGMan, etc.
- OSPool bonus features!
  - More resources (usually) than a typical local system
  - Some storage on Access Point (Data lecture, Wed.)
  - Some special resources, like GPUs (GPU topic, Thu.)
- How does OSPool differ from local one? ***Variety***

Varied Hardware

- Wider variety of CPUs (type, speed), memory, …
- Request what you need in submit files (**`request_cpus`**, **`request_memory`**, **`request_disk`**)
- Some specific hardware may be specified; search for documentation or contact us
  - Often in submit-file **`requirements`** expression
  - Example: GPU needs (ask us!)

Varied OSs and Software

- Varied Operating Systems
  - All Linux, *mostly* recent, but lots of variation
  - Changes to CentOS 8 => new variants (e.g., Alma)
  - Software on the Access Point **probably won't exist** on Execution Points! (e.g., specific Python version)
- Your software
  - Never assume your software is on Execution Points
  - The Software lecture (later today) is on this topic!

Varied Access to Data

- No shared filesystem
  - Unlike some local clusters with shared filesystems
  - Thus, files must be transferred to Execution Points
- There are many ways to handle data
  - Data lecture is Wednesday morning

Varied Policies

- Individual sites/clusters have their own policies
  - Example: Whether to kill jobs that exceed memory
  - Example: Maximum run-time of a job (or its glidein)
- If possible, set requirements for what you need
  - But this does not help with, e.g., maximum run-time
- Generally, try to make “OSG-sized” jobs (see next)

What Makes a Good OSG Job?

	Ideal Jobs! (up to 10,000 cores across Jobs, per user!)	Still Very Advantageous!	Less-so, but maybe
Cores (GPUs)	**1** (1; non-specific type)	**<8** (1; specific GPU type)	**>8 (or MPI)** (multiple)
Walltime	**<10 hrs*** *or checkpointable	**<20 hrs*** *or checkpointable	**>20 hrs**
RAM	**<few GB**	**<10s GB**	**>10s GB**
Input	**<500 MB**	**<10 GB**	**>10 GB**
Output	**<1 GB**	**<10 GB**	**>10 GB**
Software	*'portable' (pre-compiled binaries, transferable, containerizable, etc.)*	*most other than* →	*Licensed software; non-Linux*

More OSG Tips – Security

- Computer security is hard — read the headlines!
- OSG does its best, but no system is perfect
- Some suggestions:
  - Use strong, distinct passwords for each account
  - Do not share your account
  - Avoid world-writable directories and files
  - Avoid sensitive software and data (no HIPAA!)
  - Do not try to work around security barriers; contact us to help meet your goals in a safe way

Acknowledgements

You Can Acknowledge OSG!

If you publish or present results that benefitted from using OSG services, please acknowledge us!

https://osg-htc.org/acknowledging

Acknowledgements

- OSG team, especially Christina Koch; in past years: Brian Lin, Mats Rynge, and Jason Patton
- This work was supported by NSF grants OAC-1836650 and OAC-2030508

A Few Suggestions

- **Exercises**
  - Today, some exercises will specify less, so try to use what you learned yesterday — first, from memory, if possible, then look things up
  - Use Slack! There are staff online who can help, too
- **Consultations**
  - Please consider signing up for a consultation, if you haven't already
  - If the slots fill up, we will try to make more!

Demo, Part II