Introduction to OSG

Tim Cartwright

OSG Deputy Executive Director and OSG User School Director

University of Wisconsin–Madison

Overview

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

Sometimes, that is not enough!

(Don't let computing hold back your science, remember?)

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

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

What Do We Want?

- Lots of resources – available, stretchy, & reliable

- Submit locally, run globally (as close as possible)

- Automation to get resources, manage them, and run jobs

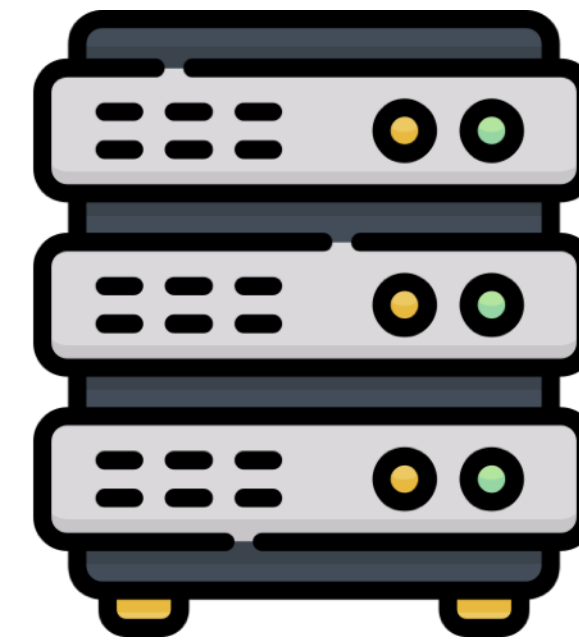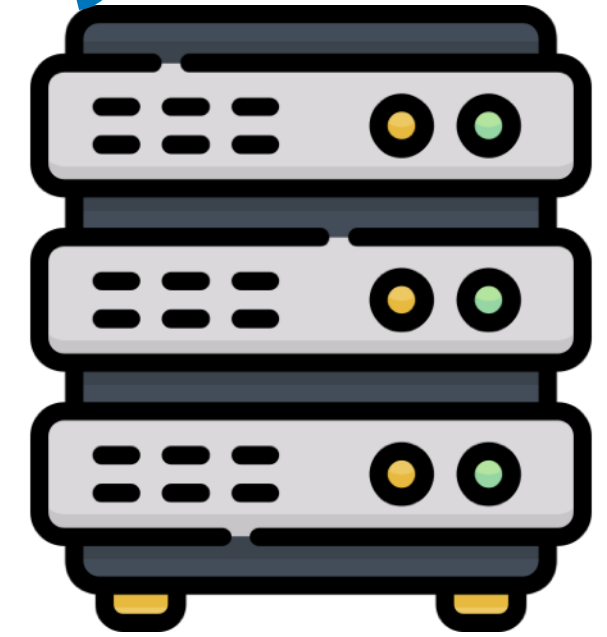- Free would be nice! (But who is the owner?)

Yesterday

Access Point

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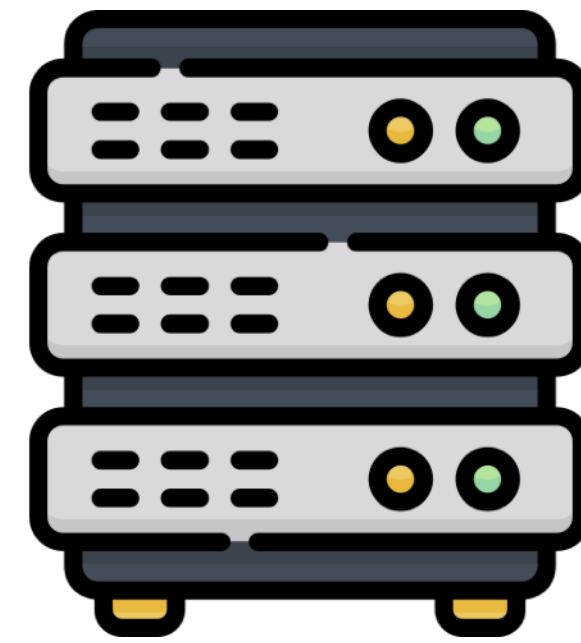
CHTC

The Goal

Access Point

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

Syracuse

CHTC

Nebraska

UCSD

https://sweetclipart.com/
https://www.flaticon.com/free-icons/server
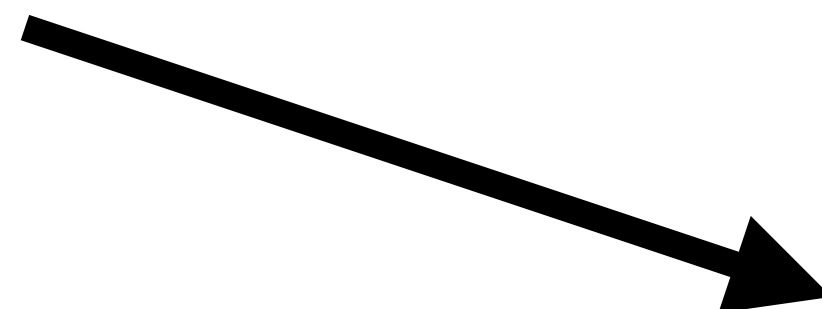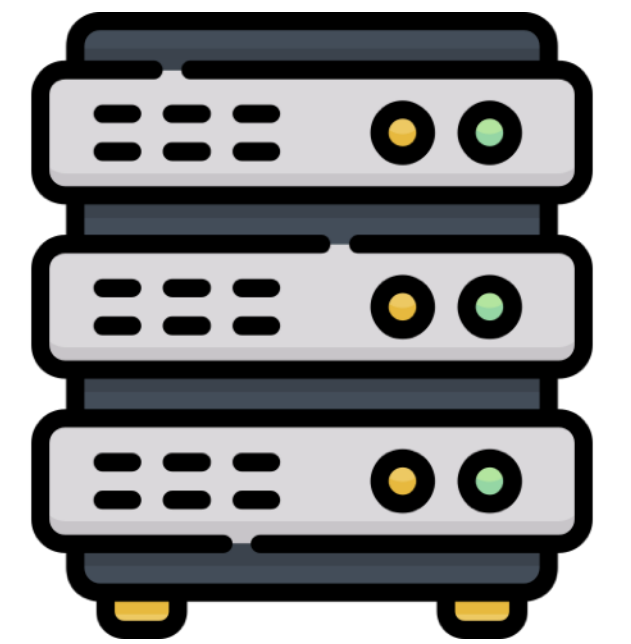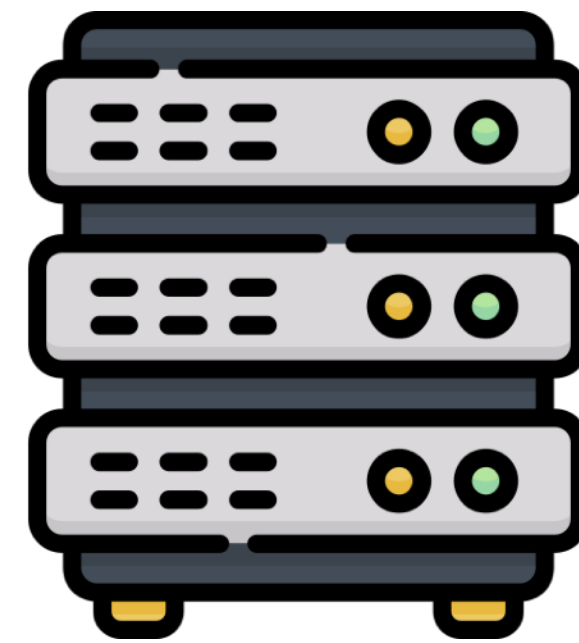
Demo, Part I

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 a pool or AP gets resources
  - How the OSPool differs from a local (CHTC) pool

What is OSG?

OSG Defined, Version 1

- ## OSG is a Consortium:

  The OSG consortium of research collaborations, campuses, national laboratories, and software providers is dedicated to the advancement of all open science via the practice of distributed High Throughput Computing (dHTC), and to the advancement of its state of the art. Established in 2005, the OSG operates a fabric of dHTC services for the National S&E community *[OSG home page]*

- ## In this view, OSG *is* people, including:

  - ## Users: Researchers like you!

    - From individual PIs to international collaborations of thousands

  - ## Resource Providers

    - Fun fact: OSG does not own or operate most of the resources!

  - ## Team

    - We provide support, infrastructure, new features, etc.

OSG Defined, Version 2

- Some people think of OSG as pools of resources (even though OSG does not own resources!)

- What is a pool of resources?

  – Resources – a set of compute, storage, and other systems that can come and go over time

  – Services – software infrastructure that manages resources and makes features available
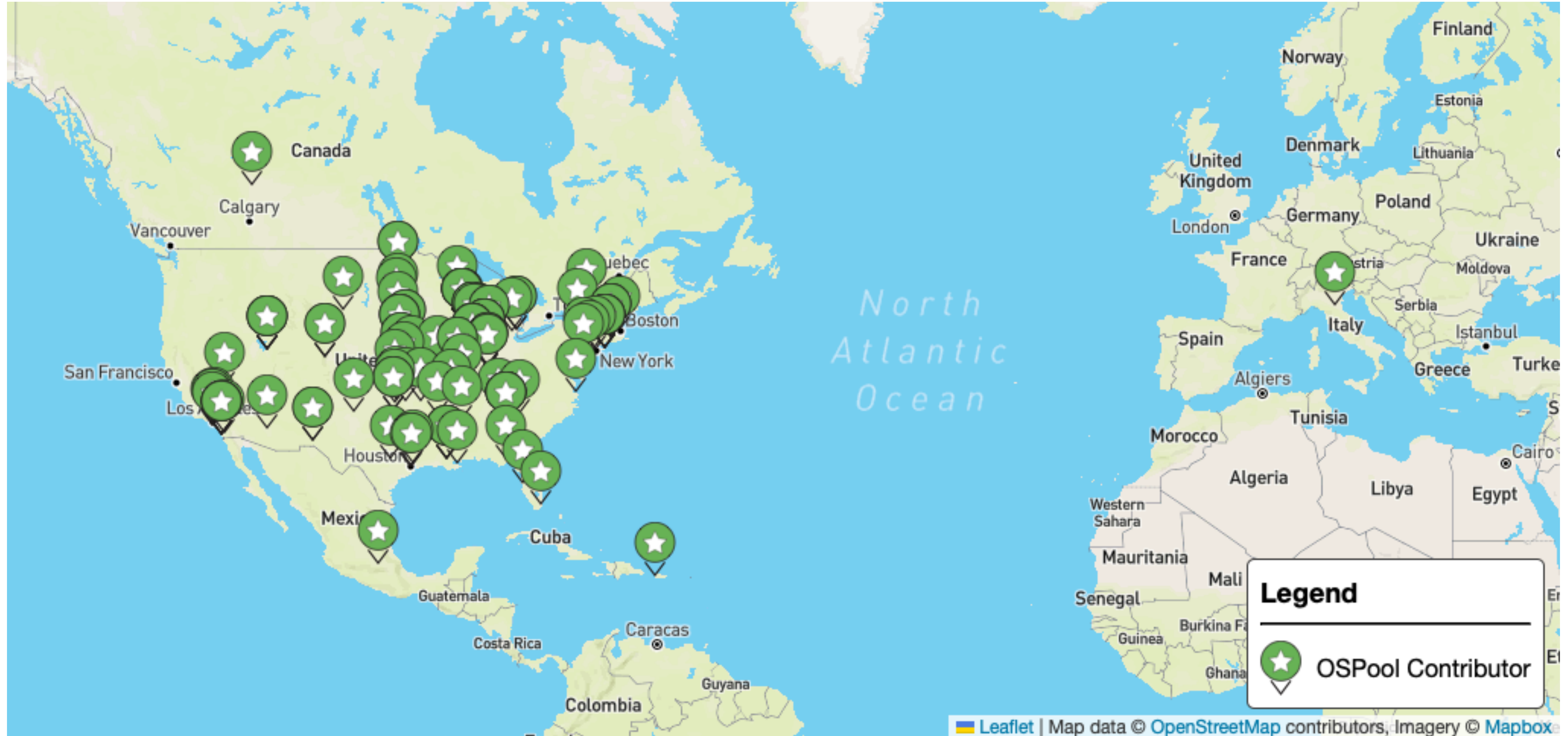
  – Access Point(s) – where users go

OSG Defined, Version 3

- For a user, the Access Point **is** OSG (or part of it)

  – Where you go to do computing

  – Has access to resources (constantly changing)

  – Provides means for accessing data (see Wednesday)

OSPool Contributors (most)

Legend
⭐ OSPool Contributor

Leaflet | Map data © OpenStreetMap contributors, Imagery © Mapbox

OSPool Usage

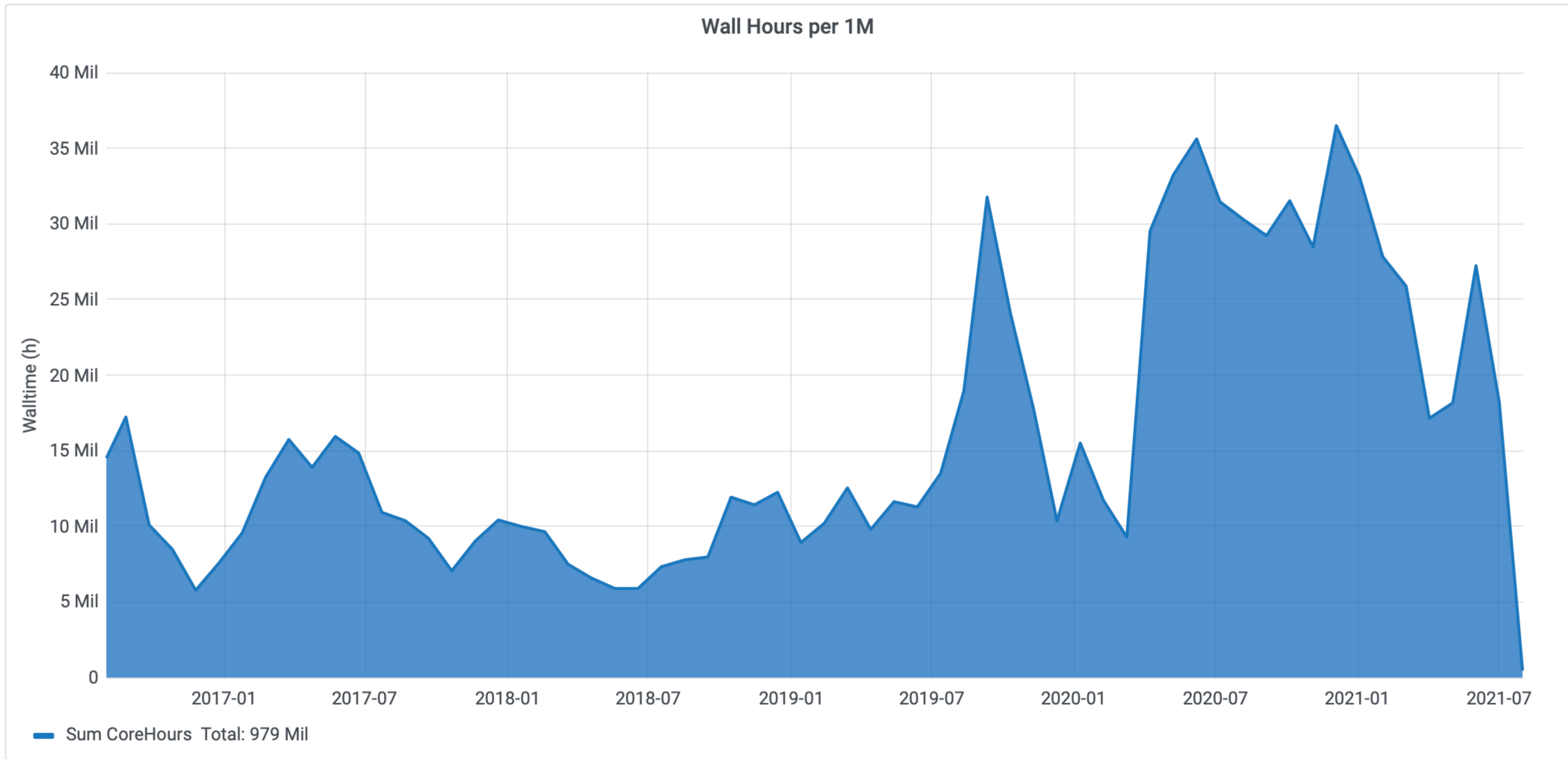Wall Hours per 1M

OSG: Getting Resources

How OSG Gets Resources

- Individual resources are contributed by owners
- OSG services consolidate resources into a Pool
- Provide users with an Access Point into the Pool
- Automate management of resources and jobs
- Called distributed High Throughput Computing (dHTC)

OSG dHTC Diagram

1. Before OSG

Nothing available at Wisc. 😭

Access Point

Job1.0
Job1.1
Job1.2
Job1.3
...
Job1.1999

Wisconsin

Busy
Busy
Busy
Busy
Busy

2. Add resource contributors!

Access Point

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

Wisconsin

Busy
Busy
Busy
Busy
Busy

Syracuse

Busy
Busy
Busy
Busy
Busy
Busy
Busy

Nebraska

Busy
Busy

Chicago

Busy
Busy
Busy

San Diego

Busy
Busy

3. Get resources (method #1)

Try to run "pilots" at sites

Access Point

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

Wisconsin

Busy
Busy
Busy
Busy
Busy

Syracuse

Busy
OSG Pilot SU1
Busy
Busy
Busy
OSG Pilot SU4
OSG Pilot SU2
OSG Pilot SU3
Busy
Busy
Busy

Nebraska

OSG Pilot NU1
OSG Pilot NU2
Busy
Busy

Chicago

OSG Pilot UC2
OSG Pilot UC1
Busy
Busy
Busy

San Diego

OSG Pilot SD2
Busy
OSG Pilot SD1
OSG Pilot SD3
Busy

4. Pilots add resources to Pool

(I am not explaining how yet)

AP

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

Pool

OSG Pilot NU1	idle
OSG Pilot NU2	idle
OSG Pilot SD1	idle
OSG Pilot SD2	idle
OSG Pilot SD3	idle
OSG Pilot UC1	idle
OSG Pilot UC2	idle
OSG Pilot SU1	idle
OSG Pilot SU2	idle
OSG Pilot SU3	idle
OSG Pilot SU4	idle

Nebraska

OSG Pilot NU1
OSG Pilot NU2
Busy
Busy

San Diego

OSG Pilot SD2
Busy
OSG Pilot SD1
OSG Pilot SD3
Busy

Wisconsin

Busy
Busy
Busy
Busy
Busy

Chicago

OSG Pilot UC2
OSG Pilot UC1
Busy
Busy
Busy

Syracuse

Busy
OSG Pilot SU1
Busy
Busy
Busy
OSG Pilot SU4
OSG Pilot SU2
OSG Pilot SU3
Busy
Busy
Busy

5. Run jobs

HTCondor with AP & Pool

AP

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

Pool

OSG Pilot NU1	Job1.4
OSG Pilot NU2	idle
OSG Pilot SD1	Job1.0
OSG Pilot SD2	Job1.3
OSG Pilot SD3	idle
OSG Pilot UC1	Job1.2
OSG Pilot UC2	Job1.6
OSG Pilot SU1	Job1.8
OSG Pilot SU2	Job1.12
OSG Pilot SU3	Job1.10
OSG Pilot 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

6. Get resources (method #2)

Site contributes directly

Queue

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

Pool

OSG Pilot NU1	Job1.4
OSG Pilot NU2	idle
OSG Pilot SD1	Job1.0
OSG Pilot SD2	Job1.3
OSG Pilot SD3	idle
OSG Pilot UC1	Job1.2
OSG Pilot UC2	Job1.6
OSG Pilot SU1	Job1.8
OSG Pilot SU2	Job1.12
OSG Pilot SU3	Job1.10
OSG Pilot SU4	idle
SU Pilot 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 Pilot OG1
Busy
Busy

OSG dHTC – A Few Details

- Notes on the *pilot* method of getting resources
  - Pilots are special jobs to request resources from other sites
  - OSG runs a service to submit pilots to contributing sites
  - A site runs a pilot when they want to contribute more resources
  - A pilot job runs part of HTCondor itself, reporting the actual resource back to the Pool
  - Pilots expire after a while or when idle, or are kicked off
- OSG and HTCondor manage/automate the details!

Open Science Pool

- **Open Science Pool** (OSPool): for all of Open Science

- It has many Access Points (e.g., projects, campuses)

- **OSG Connect** provides Access Points for US projects (including collaborators)

- Other pools exist for specific groups
  - Collaborations (e.g., gravitational-wave projects)
  - Projects (e.g., DUNE neutrino physics project)
  - Campuses (e.g., HCC at University of Nebraska)

OSG: Usage

OSG Is HTCondor

- OSG (e.g., OSPool) has HTCondor pools:
  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, Fri.)

- 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 pilot)

- 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!	Still very advantageous	Maybe not, but get in touch!
Expected Throughput, per user	1000s concurrent cores	100s concurrent cores	Let's discuss!
CPU	1 per job	< 8 per job	> 8 per job
Walltime	< 10 hrs*	< 20 hrs*	> 20 hrs
RAM	< few GB	< 40 GB	> 40 GB
Input	< 500 MB	< 10 GB	> 10 GB**
Output	< 1 GB	< 10 GB	> 10 GB**
Software	pre-compiled binaries, containers	Most other than →	Licensed Software, non-Linux

*or checkpointable

** per job; you can work with a large dataset on OSG if it can be split into pieces

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, please acknowledge us!

https://osg-htc.org/acknowledging

Acknowledgements

- OSG team, especially Brian Lin, Christina Koch, Mats Rynge, and Jason Patton

- This work was supported by NSF grants MPS-1148698, OAC-1836650, and OAC-2030508

Demo, Part II