Resource provisioning for high throughput workloads on the national cyberinfrastructure

Mats Rynge

USC Information Sciences Institute

Outline

- Workloads HTC / HPC
- National cyberinfrastructures
- Corral and GlideinWMS
- Example applications
  - NASA IPAC Kepler
  - SCEC
- pegasus-mpi-cluster
- OSG as an XSEDE service provider

High Throughput Computing

Sustained computing over long periods of time. Usually serial codes, or low number of cores threaded/MPI.

vs.

High Performance Computing

Great performance over relative short periods of time. Large scale MPI.

Why High Throughput Computing?

For many experimental scientists, scientific progress and quality of research are strongly linked to computing throughput. In other words, they are less concerned about instantaneous computing power. Instead, what matters to them is the amount of computing they can harness over a month or a year --- they measure computing power in units of scenarios per day, wind patterns per week, instructions sets per month, or crystal configurations per year.

Slide credit: Miron Livny

The Open Science Grid

A framework for large scale distributed resource sharing

addressing the technology, policy, and social requirements of sharing

OSG is a consortium of software, service and resource providers and researchers, from universities, national laboratories and computing centers across the U.S., who together build and operate the OSG project. The project is funded by the NSF and DOE, and provides staff for managing various aspects of the OSG.

Brings petascale computing and storage resources into a uniform grid computing environment

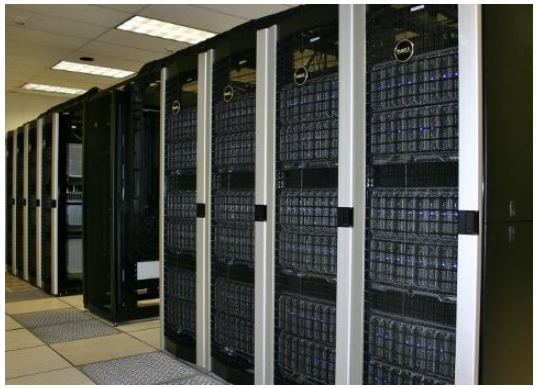
Integrates computing and storage resources from over 100 sites in the U.S. and beyond

XSEDE

- XSEDE supports 16 supercomputers and high-end visualization and data analysis resources
- Follow-on to TeraGrid
- 17 institutions (NCSA, SDSC, TACC, PSU, NICS, …)
- 120 FTE
- Funded by NSF OCI

XSEDE

Extreme Science and Engineering Discovery Environment



Bringing National Cyberinfrastructure Resources to the Scientist's Desktop

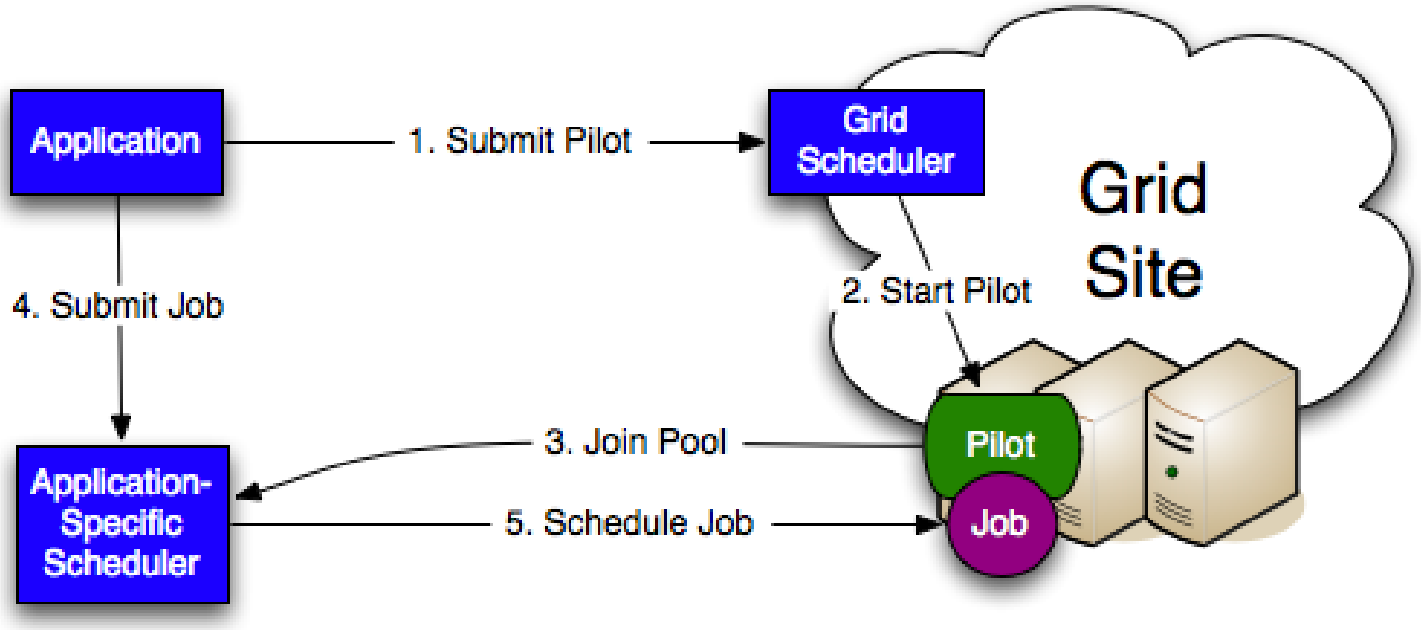
Traditional HPC/HTC

- ssh/scp access
- Grid interfaces?
- Copy data / log in to head node / set up environment / submit jobs
- Using more than one resource? Repeat.

Desktop anchored Virtual Resource

- Familiar environment
- Access to local data
- Output location?
- Flexibility
- Running across multiple infrastructures protects the scientist from downtimes, technical site problems, allocation issues, and resource availability

Pilot Jobs



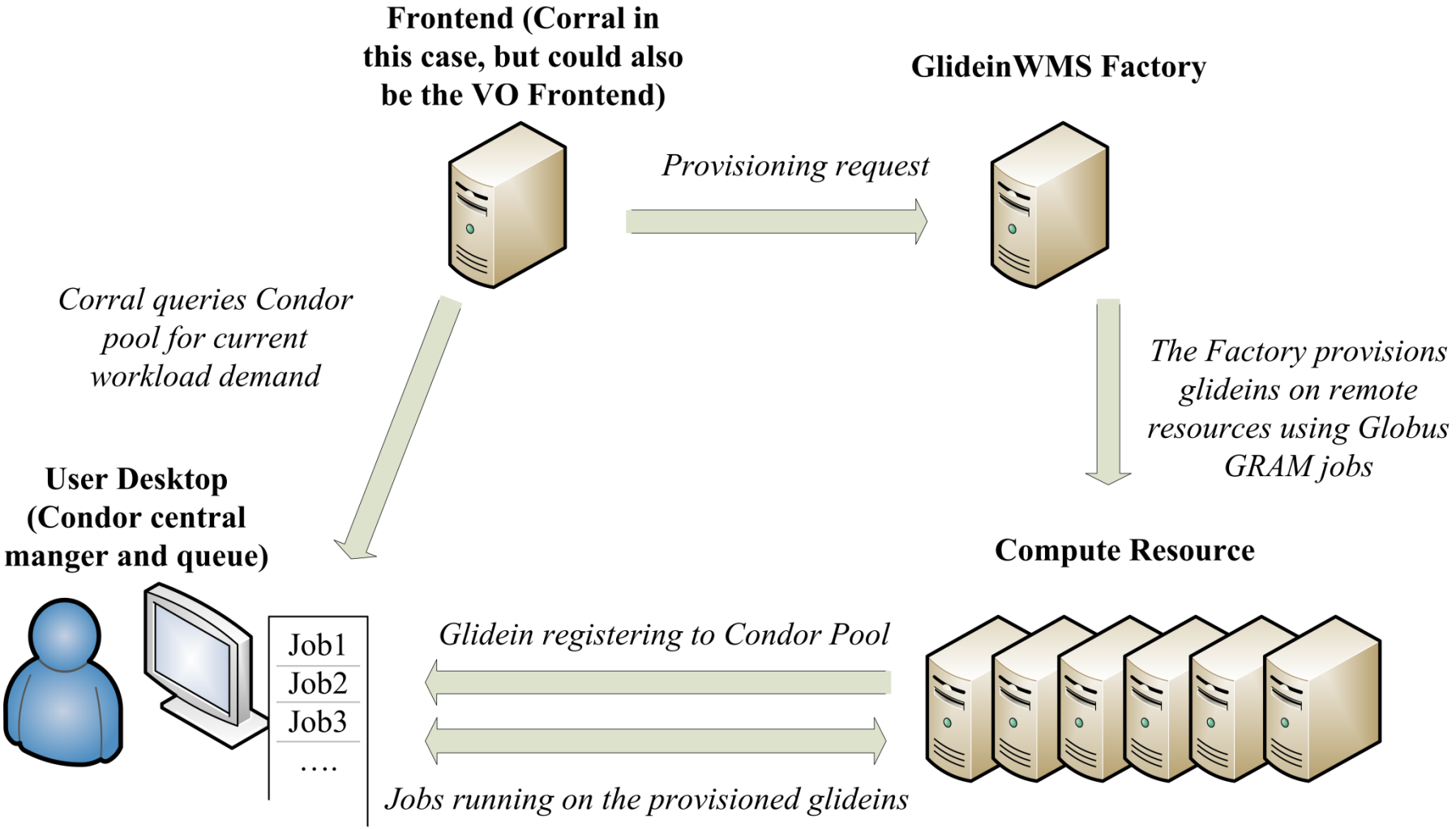
- Overlay a personal cluster on top of grid resources
- Condor based pilots: **Glideins**

GlideinWMS Overview

- Developed to meet the needs for the CMS (Compact Muon Solenoid) experiment at the LHC (Large Hadron Collider)
- Frontend watches job queue for demand
- Factory uses grid interface to submit jobs (Condor startds)
- >100,000 concurrent jobs in production

Corral - History

- Corral was a standalone provisioning tool targeting HPC resources
- Developed by Pegasus Workflow Management System team
- Short jobs
- Mixed HPC/HTC workloads
- Repurposed as a glideinWMS frontend
- Single user mode


Frontend (Corral in this case, but could also be the VO Frontend)
GlideinWMS Factory
Provisioning request
Corral queries Condor pool for current workload demand
The Factory provisions glideins on remote resources using Globus GRAM jobs
User Desktop (Condor central manger and queue)
Compute Resource
Job1
Job2
Job3
....
Glidein registering to Condor Pool
Jobs running on the provisioned glideins

glideinWMS Frontends

VO Frontend

Concept of VOs

Service certificates

Glideins shared/reused between users

Corral

Individual users

Personal certificates

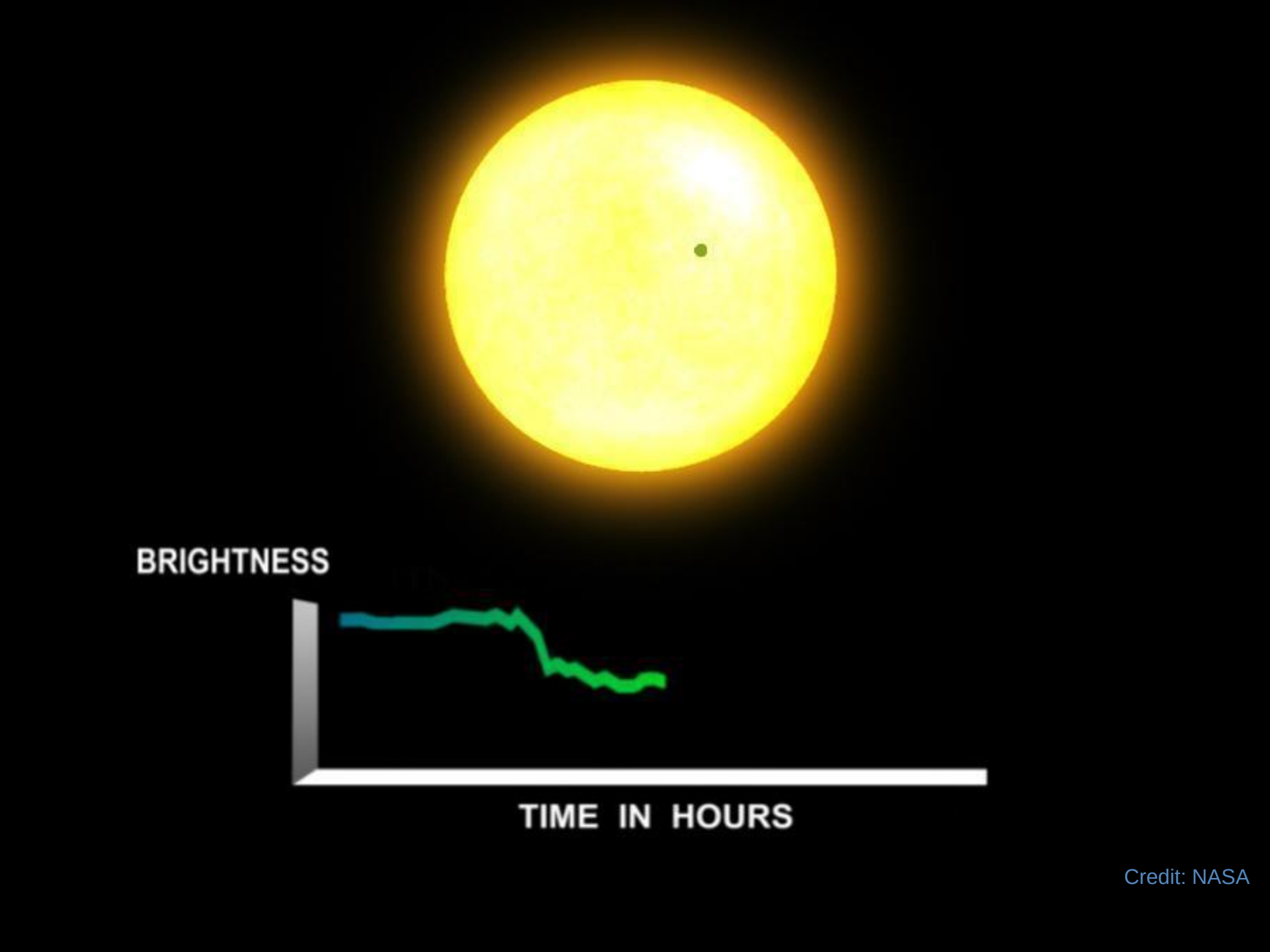
Glideins tied to user

This flexibility allows Corral to acquire a mix of resources with different user/group mappings when running across infrastructures

Desktop Setup

- Condor central manager
  - Collector – for the glideins to register to
  - Schedd – submit jobs
- X.509 security
- 10 sub collectors
- From the users point of view:

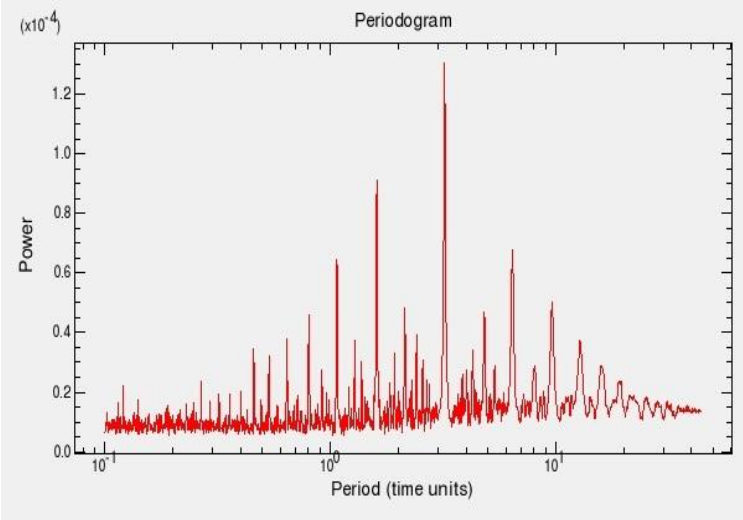
Standard Condor pool



Credit: NASA

Periodograms

- Dataset: ~210,000 stars
- Calculates the significance of different frequencies in time-series data to identify periodic signals.
- Light curve -> Periodogram -> Event -> Event database
- FFT
- Three different algorithms

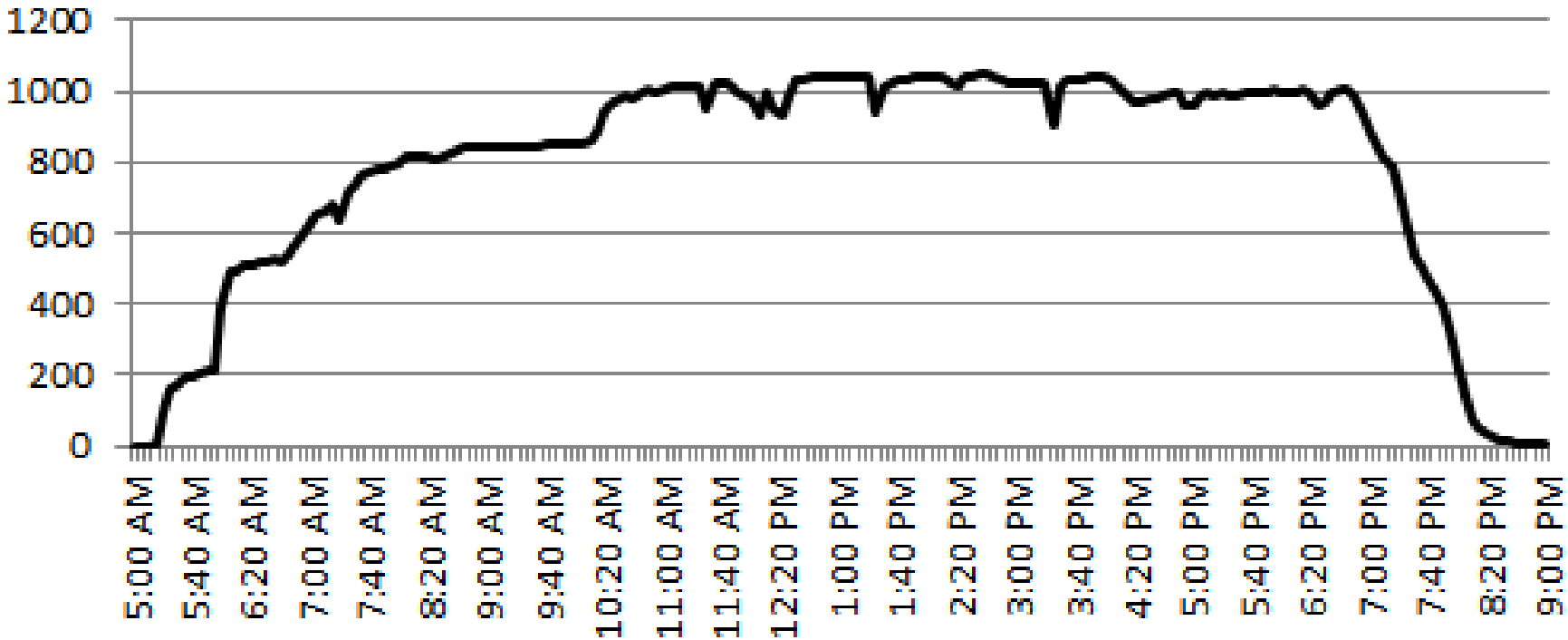


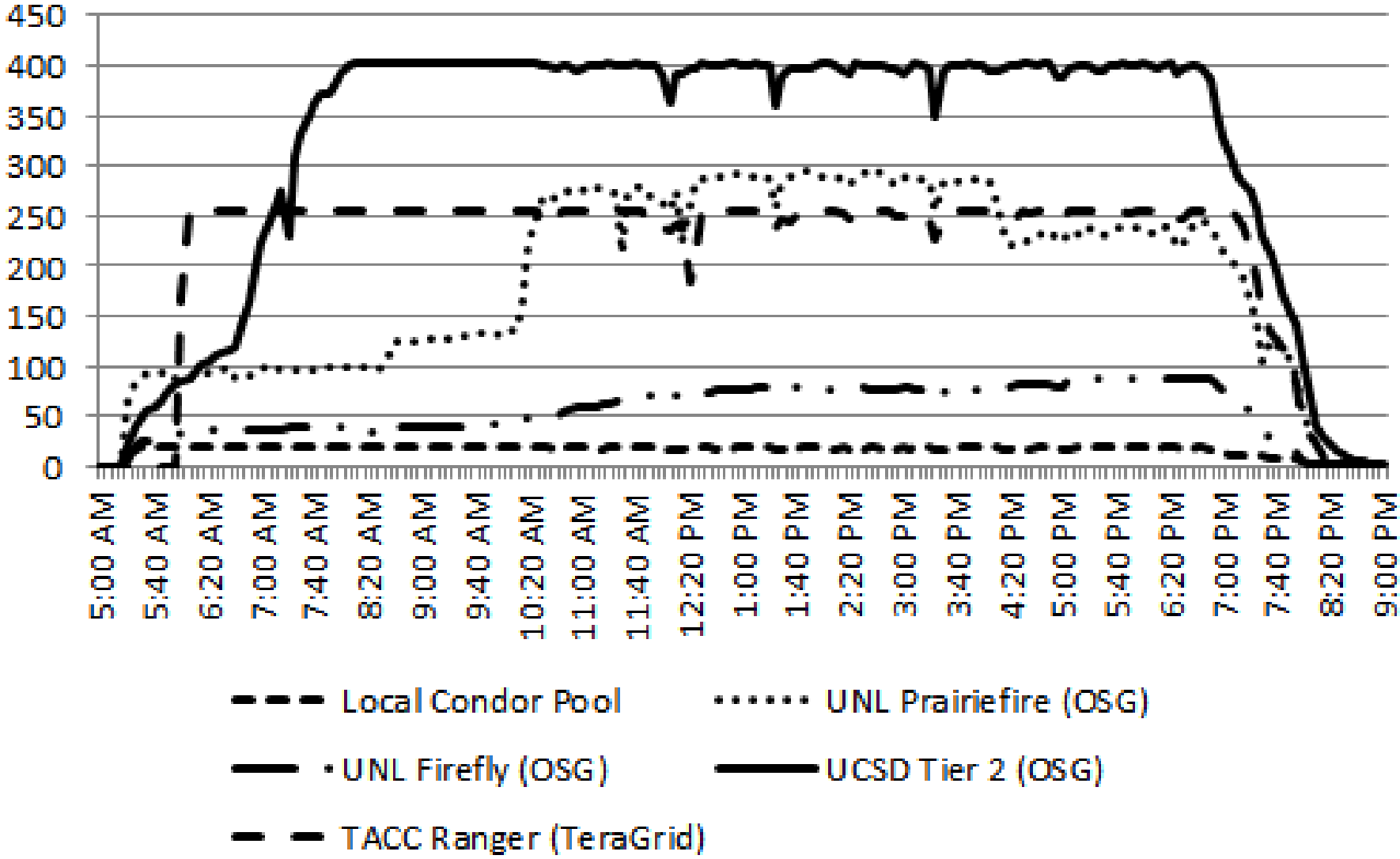
BLS periodogram for Kepler -4b, the smallest transiting exoplanet discovered by Kepler to date.

Workflow Details

- 11 sub workflows, ~ 50000 tasks each
- Wall time based job clustering
  - Simple binning
  - Target: 1 hour
- ~ 800 jobs per sub workflow
- Execute across available resources: Local, Open Science Grid, TeraGrid

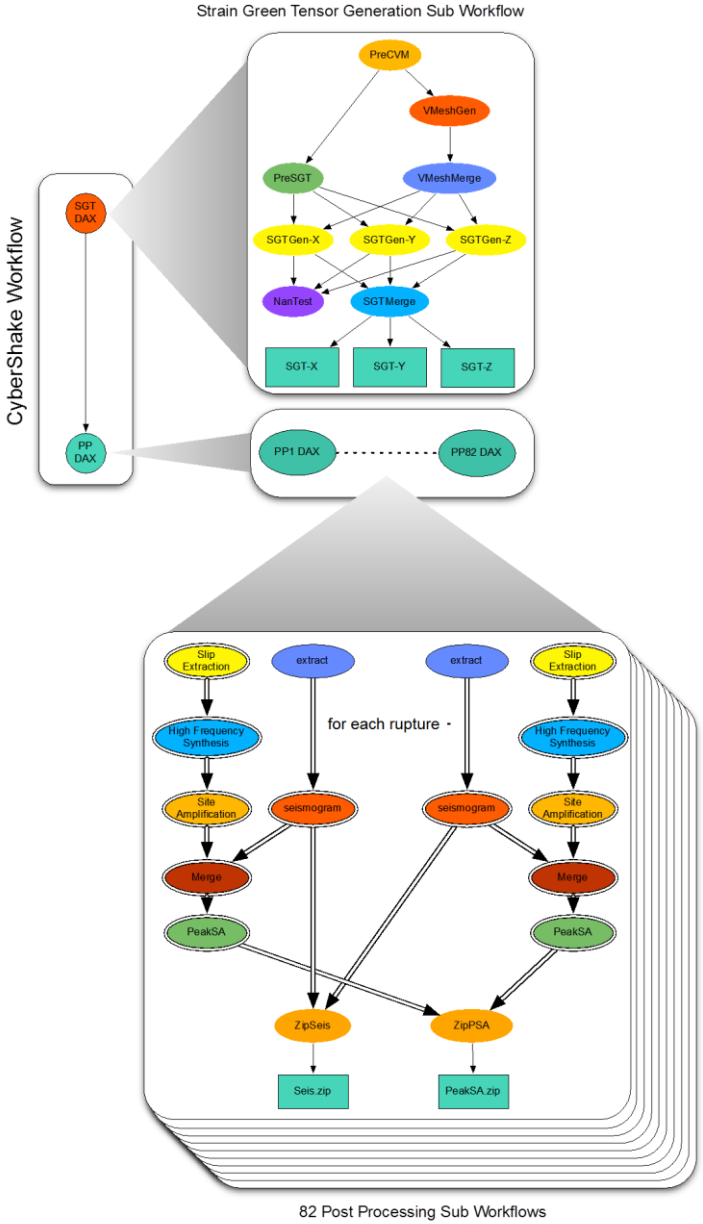
Size of Condor Pool

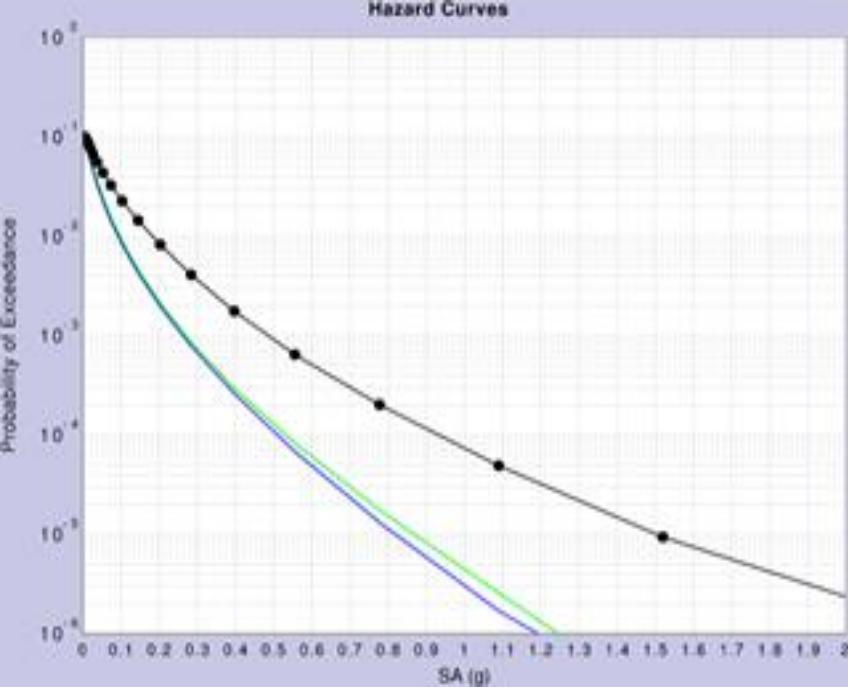



450
400
350
300
250
200
150
100
50
0
5:00 AM
5:40 AM
6:20 AM
7:00 AM
7:40 AM
8:20 AM
9:00 AM
9:40 AM
10:20 AM
11:00 AM
11:40 AM
12:20 PM
1:00 PM
1:40 PM
2:20 PM
3:00 PM
3:40 PM
4:20 PM
5:00 PM
5:40 PM
6:20 PM
7:00 PM
7:40 PM
8:20 PM
9:00 PM
Local Condor Pool
UNL Prairiefire (OSG)
UNL Firefly (OSG)
UCSD Tier 2 (OSG)
TACC Ranger (TeraGrid)

CyberShake

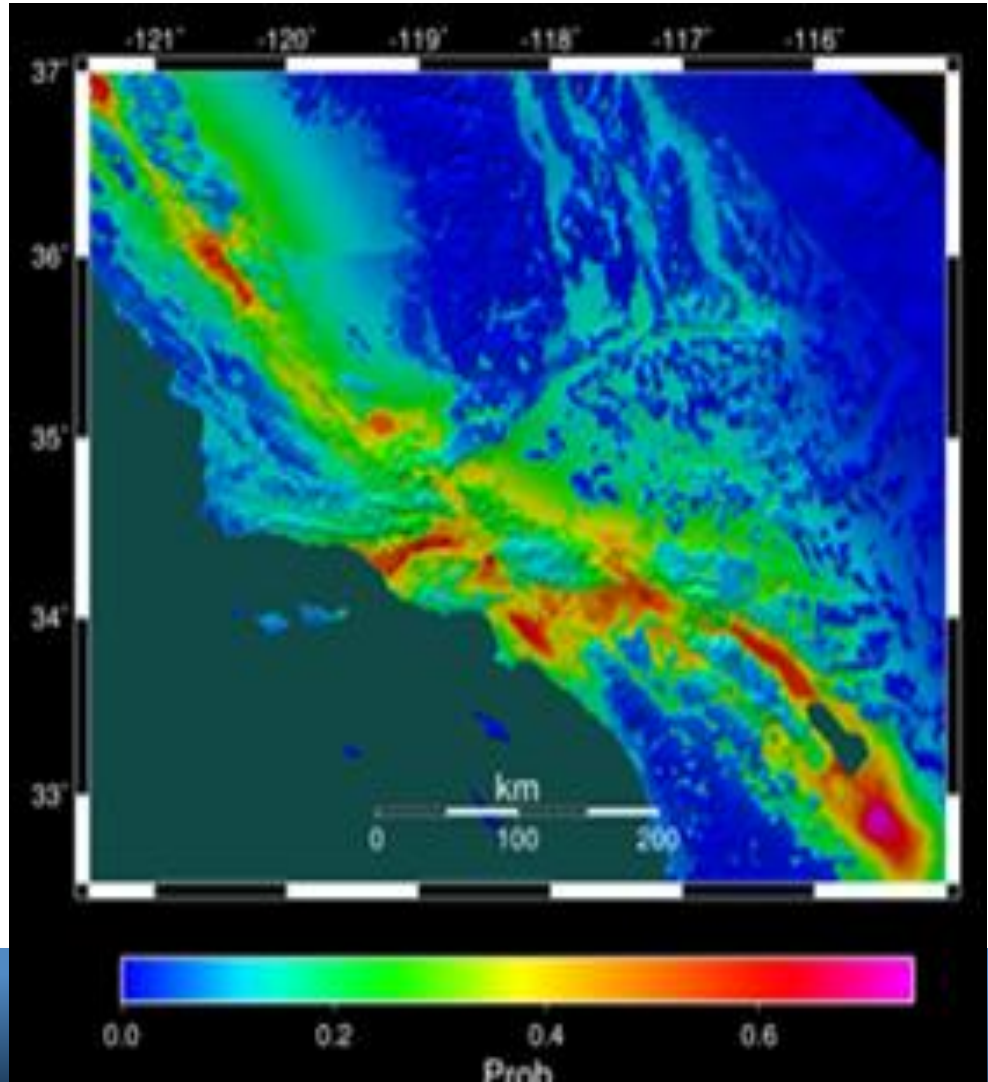
- Probabilistic seismic hazard analysis workflow
  - How hard will the ground shake in the future?
  - Considers a set of possible large earthquakes
  - 415,000 earthquakes is typical
- Uses Pegasus and HTCondor DAGMan for workflow management
  - Hierarchal workflows
  - Small set of large parallel jobs
  - 840,000 serial jobs, in 78 sub workflows



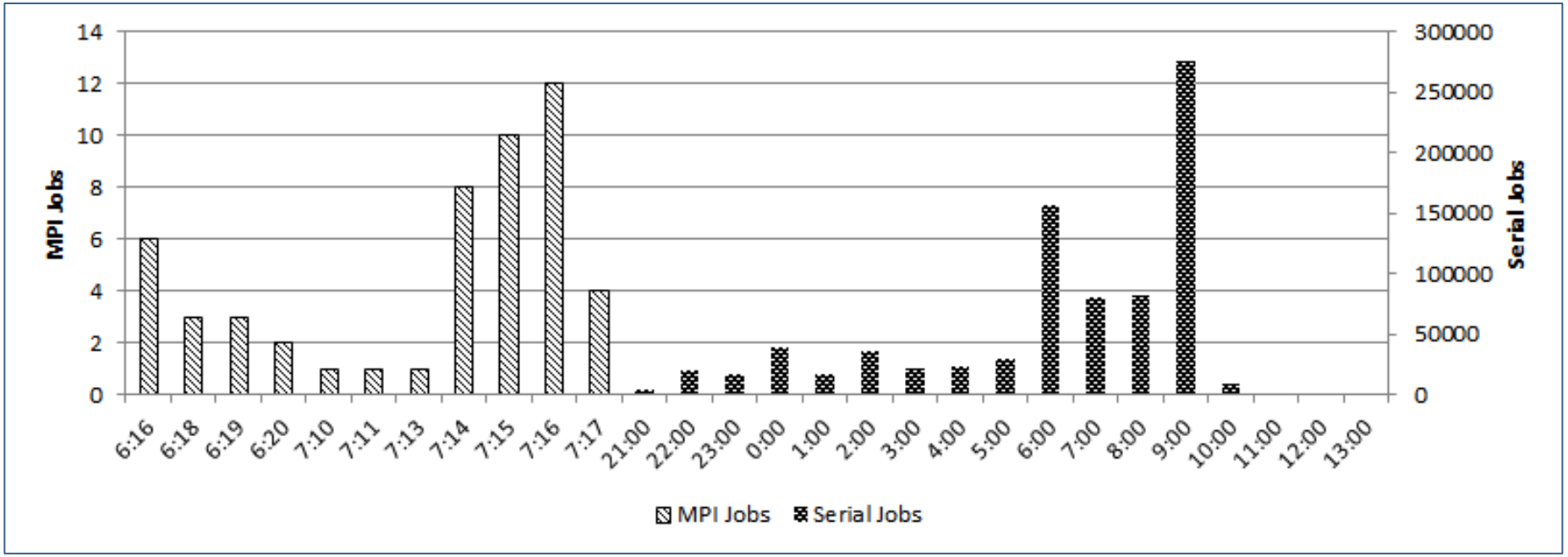


Probabilistic Seismic Hazard Analysis (PSHA) curve. Estimates the probability that earthquake ground motions will exceed some intensity measure.

Set of PSHA curves interpolated creates hazard map for an area

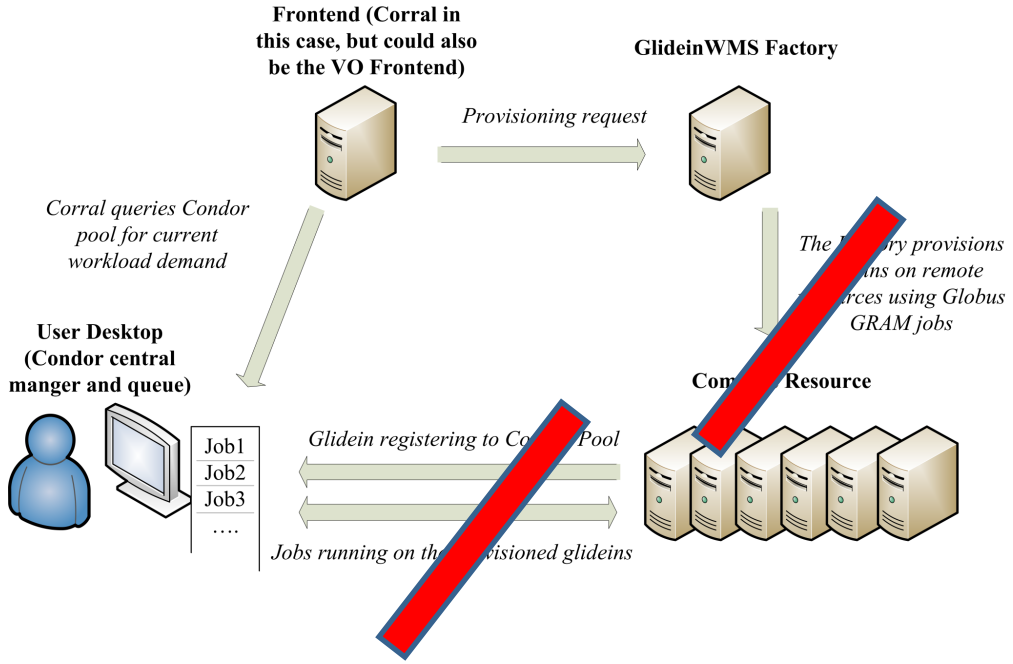


A mix of MPI and serial jobs



Glideins on NICS Kraken?

- Cray XT System Environment / ALPS / aprun
  - Login node
  - aprun node
  - Compute node

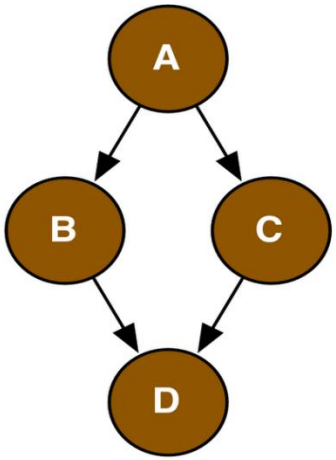


Approach

- Partition workflow into subgraphs
- Execute partition as a self-contained MPI job

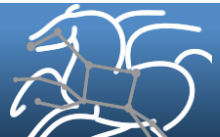
pegasus-mpi-cluster

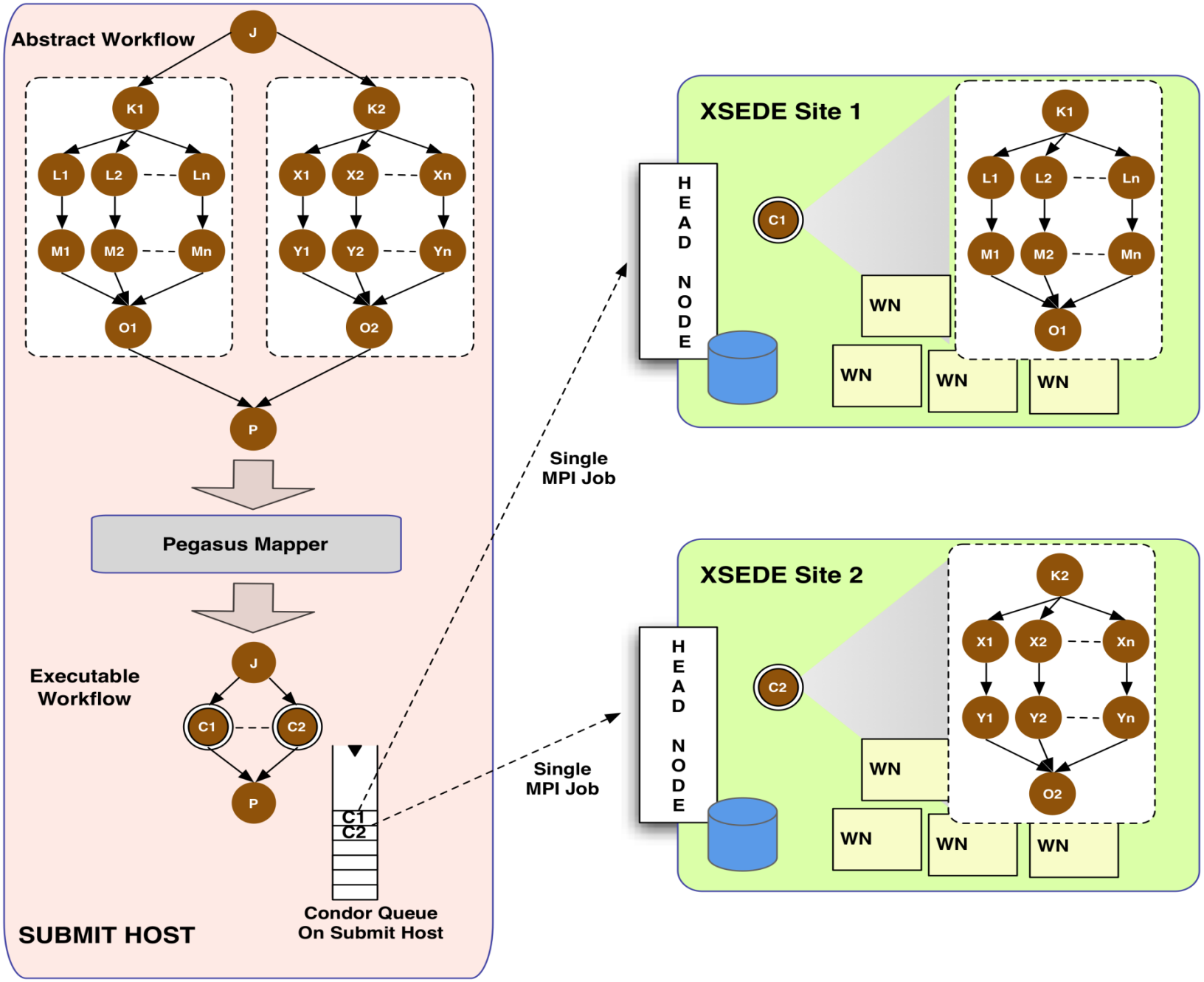
- Master/worker paradigm
- Master manages the subgraph tasks, handing out work to the workers
- Efficient scheduling / handling of input/outputs
- Subgraph described in a DAG-similar format
- Failure management / rescue DAG

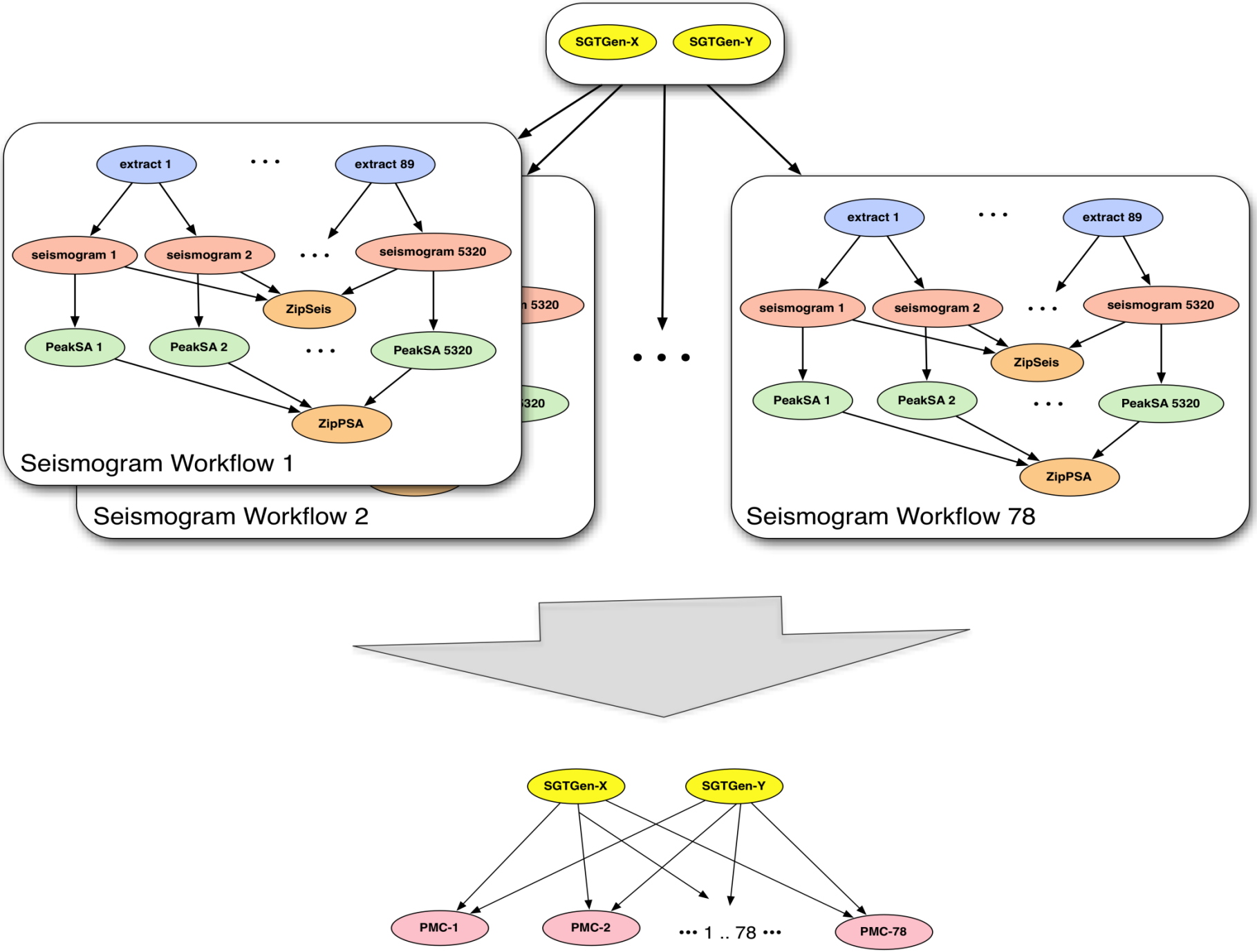


diamond.dag

```
TASK A /bin/echo "I am task A"
TASK B /bin/echo "I am task B"
TASK C /bin/echo "I am task C"
TASK D /bin/echo "I am task D"
EDGE A B
EDGE A C
EDGE B D
EDGE C D
```


Abstract Workflow
J
K1
K2
L1
L2
Ln
X1
X2
Xn
M1
M2
Mn
Y1
Y2
Yn
O1
O2
P
Pegasus Mapper
Executable Workflow
J
C1
C2
P
C1
C2
Condor Queue On Submit Host
SUBMIT HOST
Single MPI Job
Single MPI Job
XSEDE Site 1
HEAD NODE
C1
WN
WN
WN
WN
K1
L1
L2
Ln
M1
M2
Mn
O1
XSEDE Site 2
HEAD NODE
C2
WN
WN
WN
WN
K2
X1
X2
Xn
Y1
Y2
Yn
O2


SGTGen-X
SGTGen-Y
extract 1
extract 89
seismogram 1
seismogram 2
seismogram 5320
ZipSeis
PeakSA 1
PeakSA 2
PeakSA 5320
ZipPSA
Seismogram Workflow 1
Seismogram Workflow 2
Seismogram Workflow 78
SGTGen-X
SGTGen-Y
PMC-1
PMC-2
... 1 .. 78 ...
PMC-78

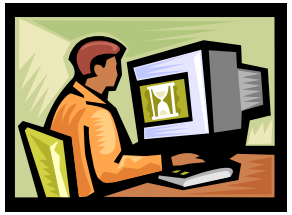
PMC - Future Work

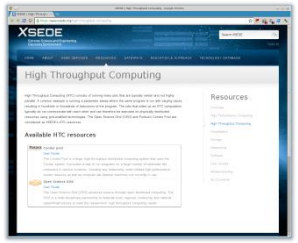
- Demonstrated efficient execution of fine-grained workflows on petascale resources by partitioning workflow into MPI master/worker jobs
- Size of partition?
- Size of MPI job?
- Handing tasks with mixed requirements?
  - pegasus-mpi-cluster now considers memory to be a consumable resource

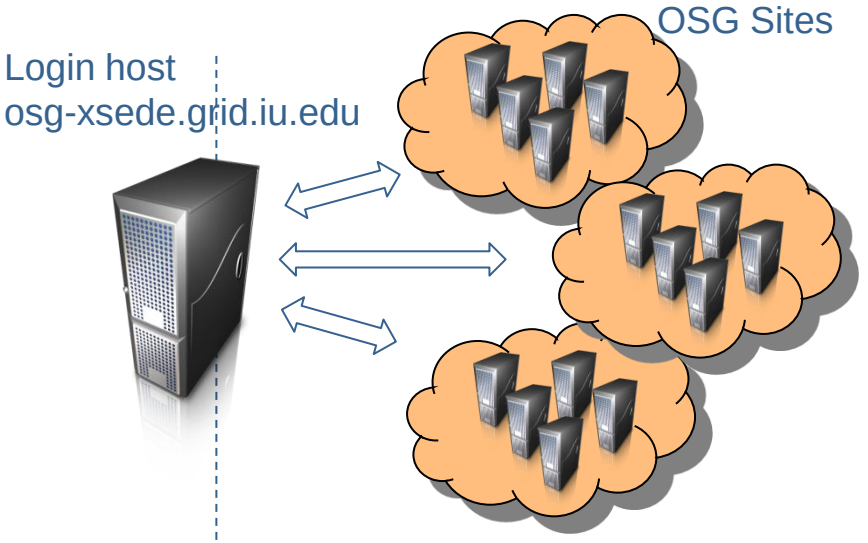
OSG is now an XSEDE service provider

- HTC / HTPC
- Implemented as a login host with a dynamically sized Condor pool drawn from opportunistic cycles available at OSG sites
  - (i.e. a GlideinWMS frontend)
- Currently contributing 2M SUs / quarter
- Challenges
  - XSEDE is based on allocations / OSG on opportunistic use
  - XSEDE has a central database with allocations and users / OSG has distributed VOs
  - Collecting and aggregating usage data to both XSEDE and OSG

OSG-XSEDE Interface

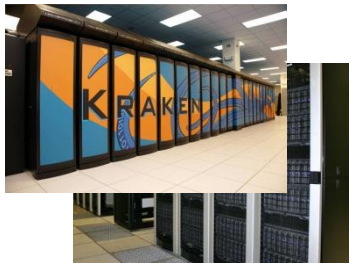






Flocking / user submit hosts

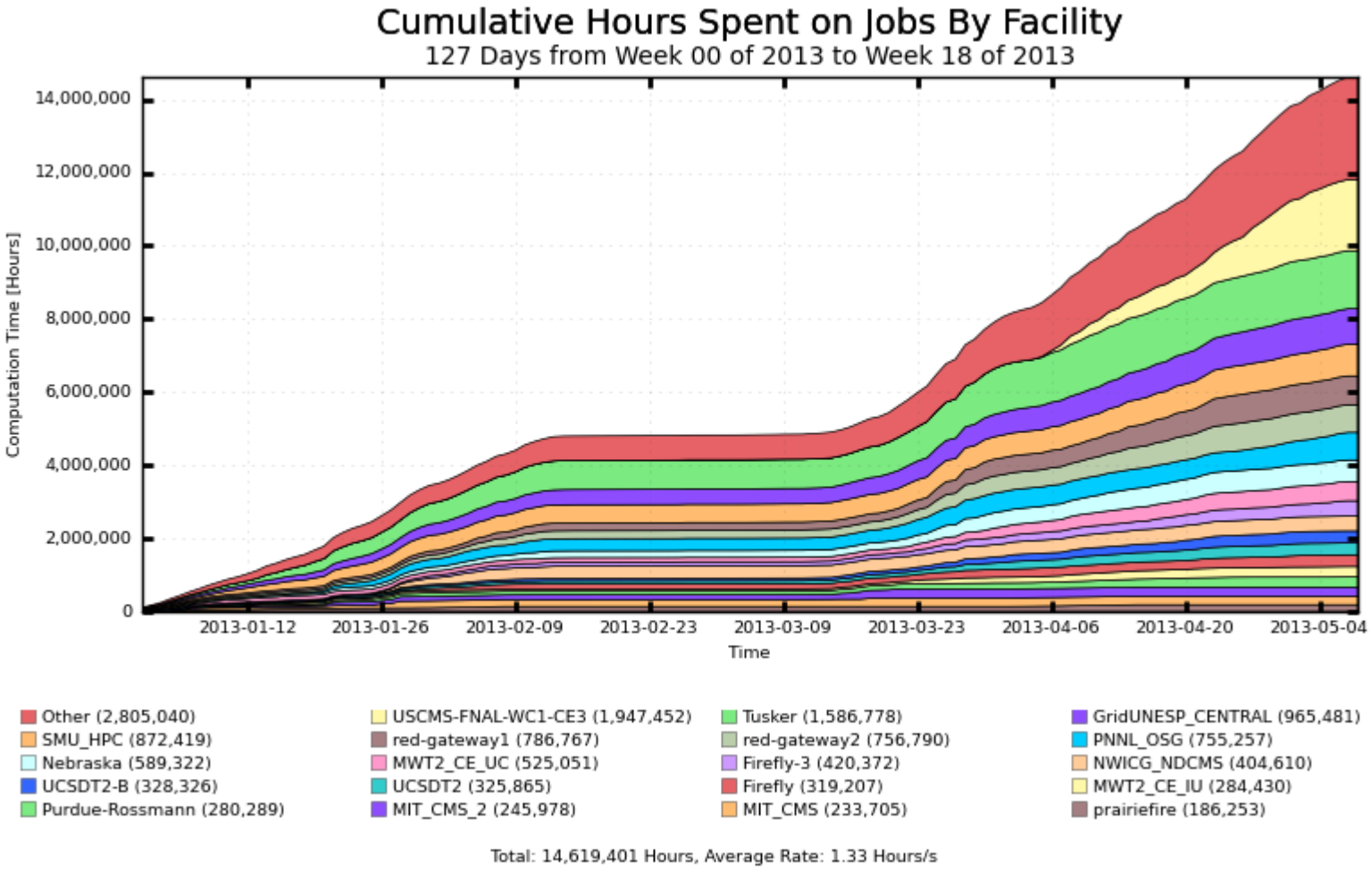
workflow.isi.edu



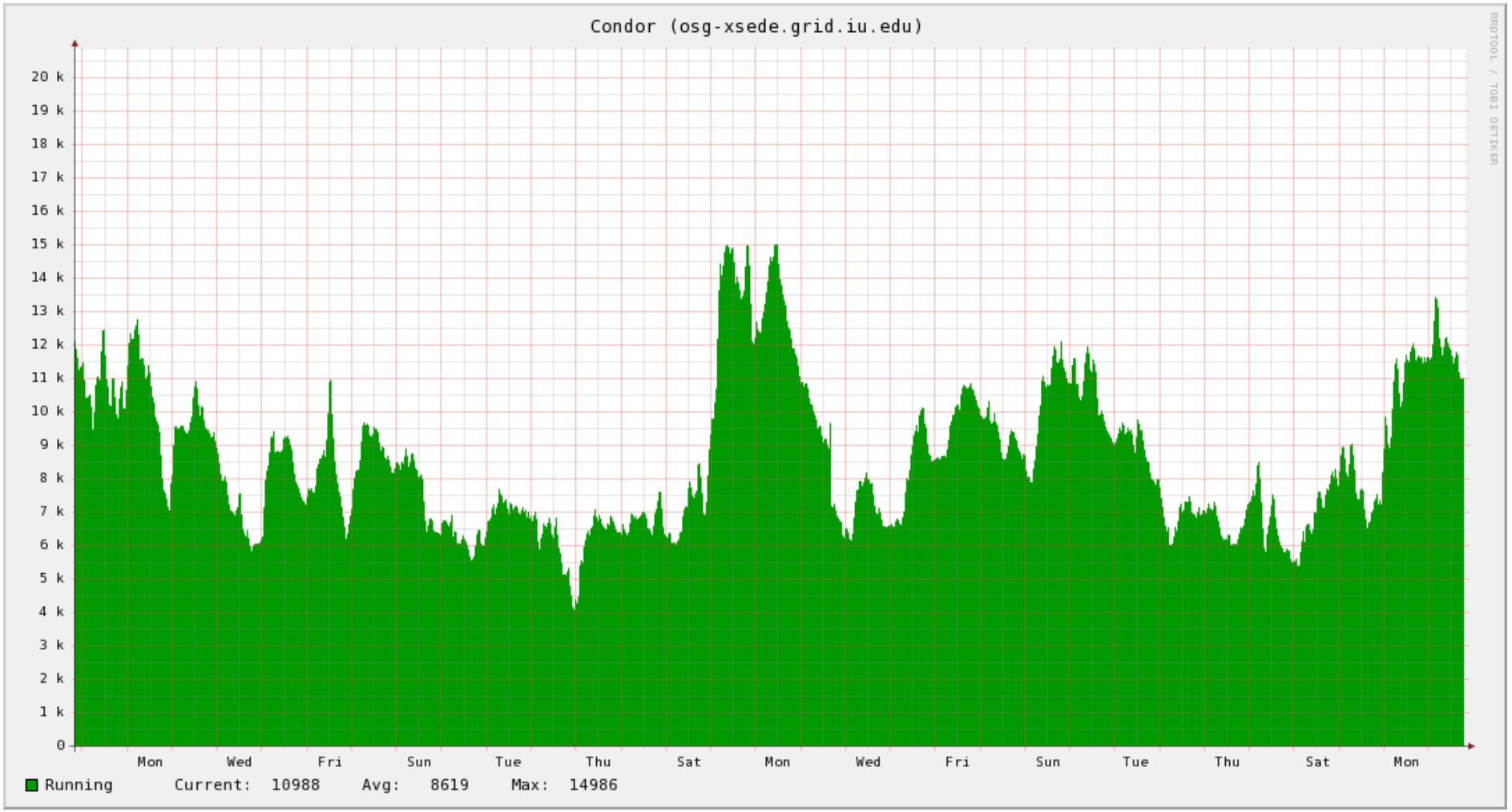
XSEDE Service Providers

XSEDE users login to the “OSG Virtual Cluster”, which provides an abstraction layer to access the distributed OSG fabric. This interface allows XSEDE users to view the OSG as one resource where they submit their jobs, provide the inputs and retrieve the outputs.

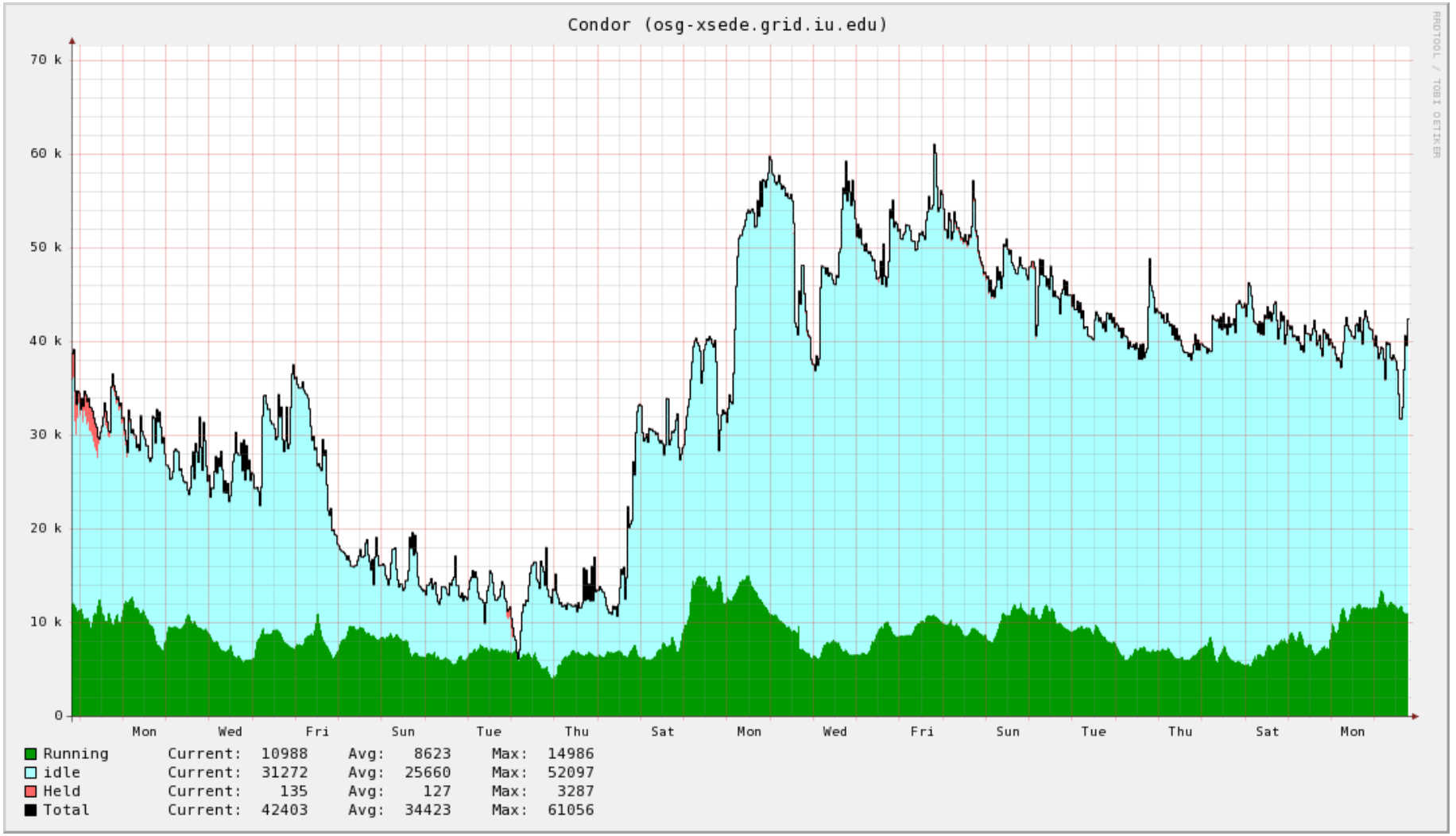
OSG-XSEDE: Where do the resources come from?



OSG-XSEDE Running jobs



OSG-XSEDE Running and pending jobs

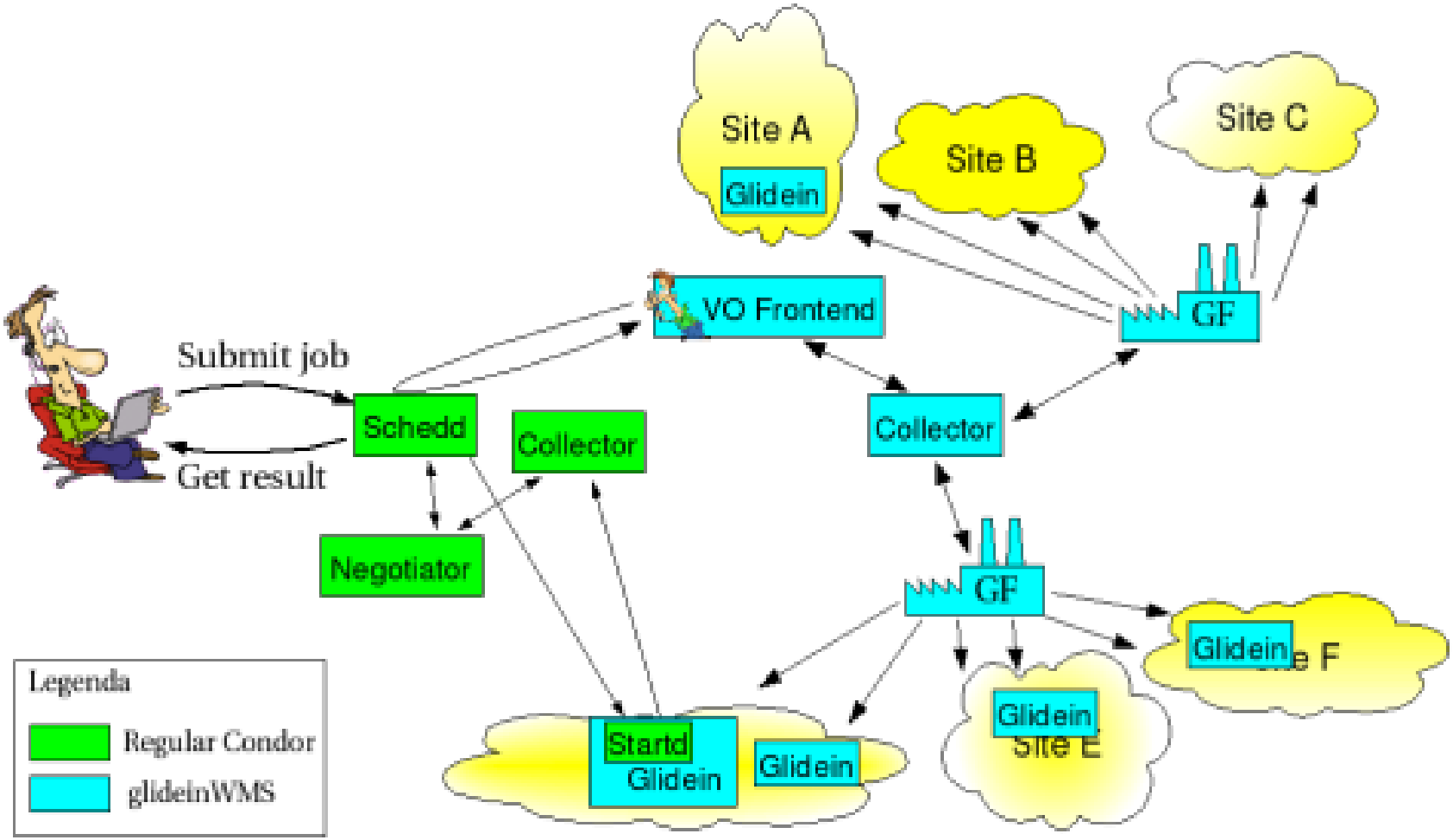


Looking forward

- GlideinWMS
  - 3.1 release
  - Clouds
- Pegasus-mpi-cluster sizing advanced features
- Continue to operate and evolve OSG-XSEDE

Questions?

GlideinWMS


Site A
Glidein
Site B
Site C
VO Frontend
GF
Submit job
Get result
Schedd
Collector
Collector
Negotiator
GF
Glidein
Startd
Glidein
Glidein
Glidein
Legenda
Regular Condor
glideinWMS

GlideinWMS groups and two-level matching

- First level to determine what type of glidein is needed
  - Maps job to a group: main, large memory, long job, HTPC, …
- Second level to match a job to a provisioned glidein
  - Startd limits jobs to a particular group
  - User can use job requirements to limit within the group

Multislot Requests

- Mapping demand from user job queue to a factory request to a single grid job requesting N slots
- Efficiency – grow the pool quickly
- Queue limits – only allowed 7 jobs in the queue

Periodogram Jobs Running on the Open Science Grid