Data Management for LSST Image Simulation on OSG

Overview

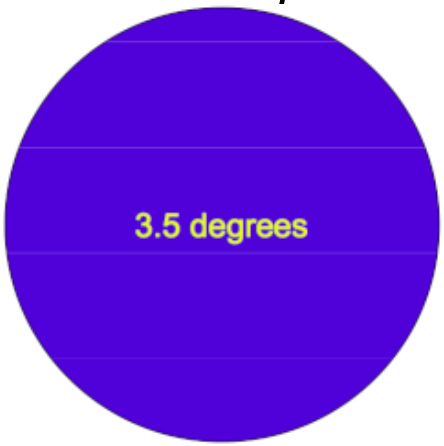
- Introduction to LSST and OSG
- Description of LSST simulation on OSG
- Two kinds of data transfer
- Performance
- Some practical details

Marko Slyz
for the OSG Task Force on LSST
Computing Division, Fermilab

LSST

The **L**arge **S**ynoptic **S**urvey **T**elescope is designed to record wide-angle images of the night sky. Can photograph the entire sky in a few nights, and produce near real time reports of interesting events. Expected to record 30 TB of data a night using a 3.2 gigapixel camera.

– *see http://www.lsst.org/lsst/faq*



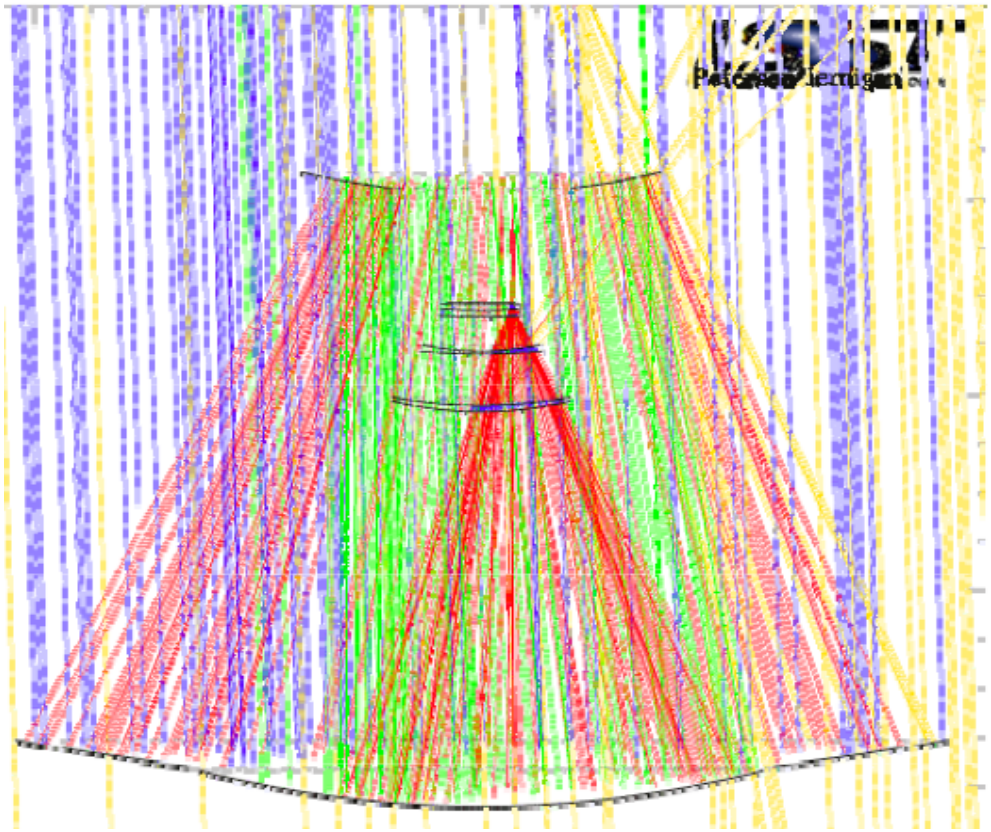
LSST field of view

0.5 degrees

--from I. Shipsey's talk at FNAL, April 2010.

LSST Image Simulation

Simulate the path of billions of photons from their sources through the atmosphere, telescope optics, and the sensor. --see http://lsst.astro.washington.edu/intro/overview/



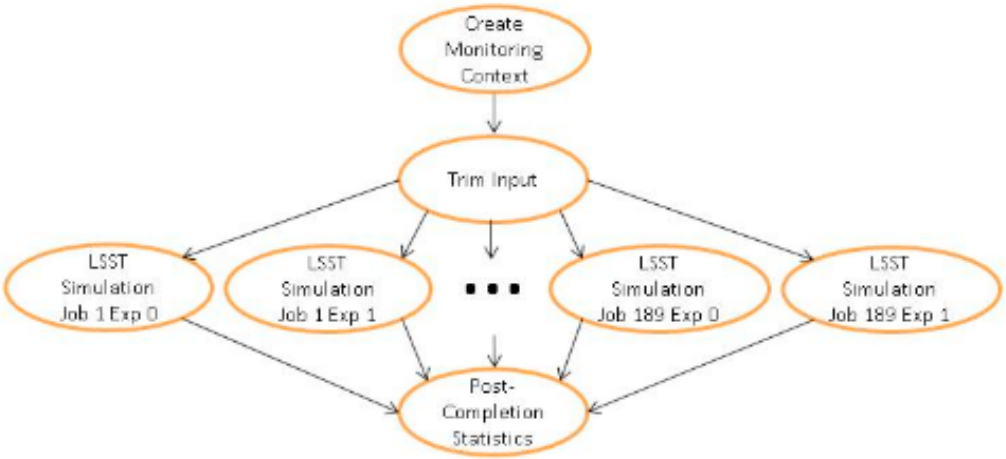
--from I. Shipsey's talk at FNAL, April 2010.

OSG Involvement

- LSST at Purdue (Ian Shipsey) and OSG are collaborating to explore the use of the OSG to run LSST computations
- We have integrated the LSST “current” version of the image simulation with OSG
- We have produced about 500 image pairs and have completed the validation
- Goal of OSG is to empower LSST to use OSG resources independently

Basic Job Workflow

- LSST simulation of 1 image: 189 trivially parallel jobs for the 189 chips
- Input to the workflow:
  - SED catalog and focal plane conf. files: 15 GB uncompr.
  - Instance Catalog (SED files + wind speed, moon position, atmosphere, etc.): 2.8 MB compr. per chip (~.5 GB total)
- Workflow:
  - Split catalog file into 189 chip-specific files
  - Submit 2 x 189 jobs:
    1 image pair (same image w/ 2 exposures)
- Output: 2 x 189
  FITS files, 25 MB
  per chip each compr.



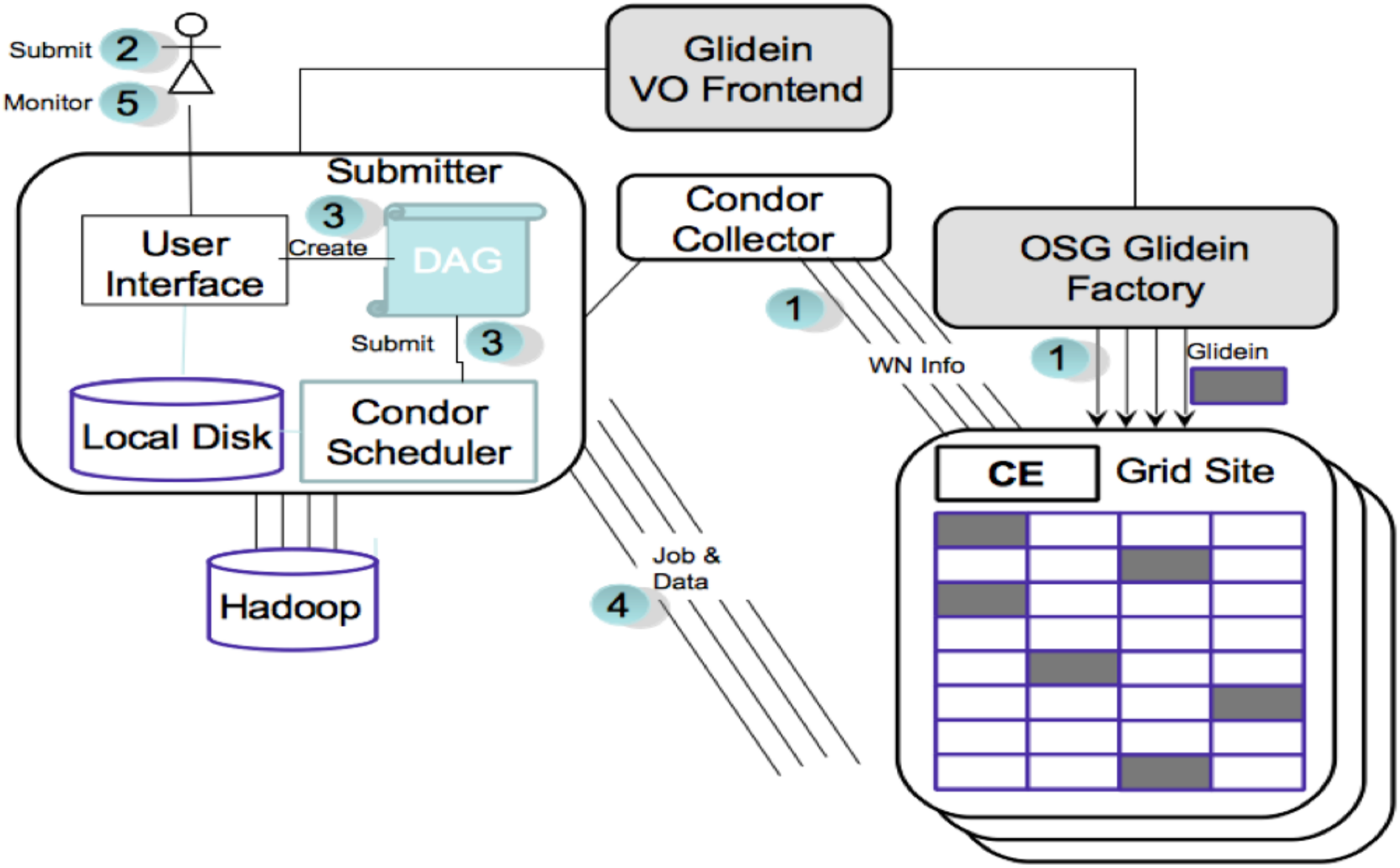
Production Estimate

Some back-of-the-envelope arithmetic:

Goal: simulate 1 night of LSST data collection: 500 pairs

- 200k simulation jobs (1 chip at a time) + 500 trim jobs
- Assume 4 hours / job for trim and simulation (over-est.)
  → 800,000 CPU hours
- Assume 2000 jobs running on average → ~50,000 CPU hours / day => ~17 days to complete (w/o counting failures)
- 12,000 jobs / day i.e. 31 image pairs / day
- 50 GB / day of input files *moved* (different for every job)
- 300 GB / day of output
- Total number of files = 400,000 (50% input - 50% output)
- Total output compressed = 5.0 TB (25 MB per job) ⌘

Architecture



Handling Different Data Types

Static Data:
The 15GB SED catalog changes rarely. => Preinstall this and the simulation application at all the sites.

Dynamic data:
The ~3.0 MB compressed instance catalog is different for every job, as are the pair of 25MB output images. => Use glideinWMS file transfer to upload and download with every run.

--see https://twiki.grid.iu.edu/bin/viewauth/ReleaseDocumentation/StorageEndUser

glideinWMS File Transfer

Files get transferred directly from submit host to worker node. Specify which files in job description.

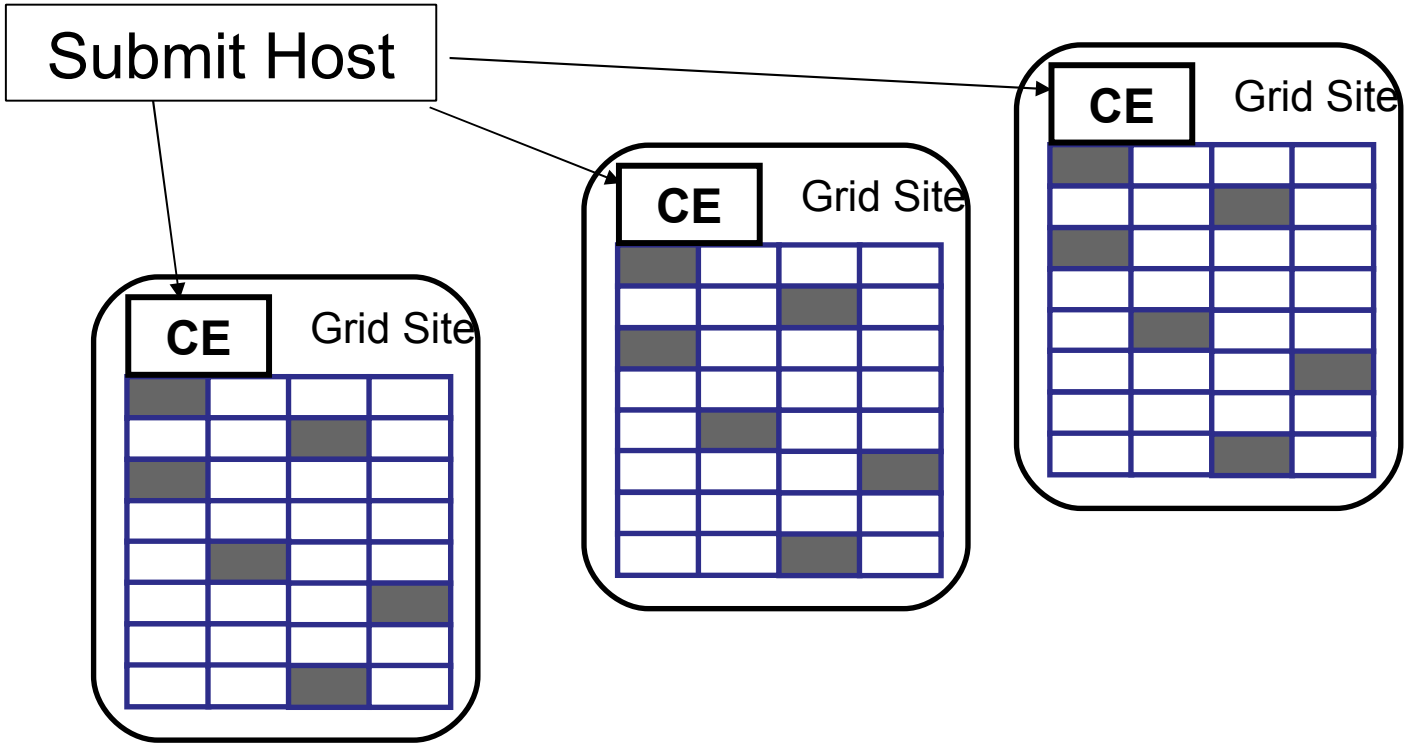
Convenient, but:

- Transferred files are not stored from run to run. Important if files are big.
- Need to be careful not to transfer unneeded data.

Note that intermediate files in the DAG get sent back to the submit host.

http://www.cs.wisc.edu/condor/manual/v7.4/2_5Submitting_Job.html#SECTION003540000000000000000

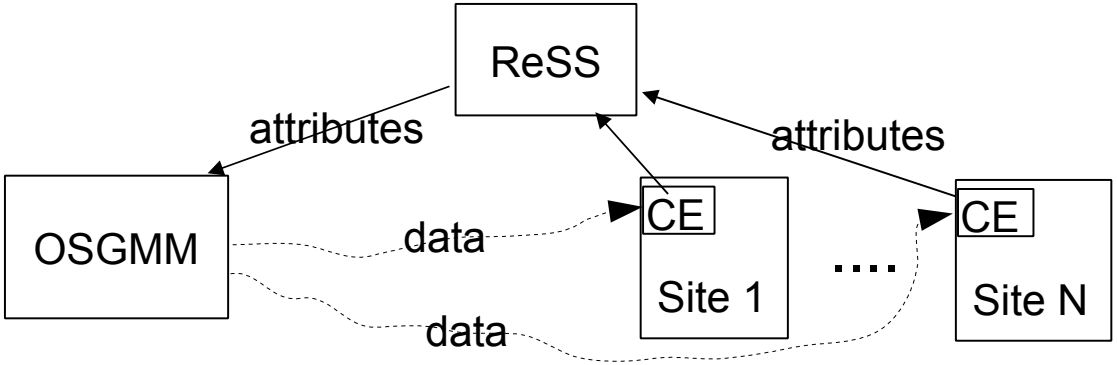
Detail of Architecture for Preinstalling Data



Each grid site has a shared file system readable by the CE and worker nodes, but worker nodes can't always write to it.
=> Transfer the files to the CEs only.

Preinstalling Data

1. The sites report their attributes (i.e. “I have 64-bit x86-based machines”) to a server called *ReSS*.
2. A service called *OSGMM* gets these attributes and decides what sites are good matches.



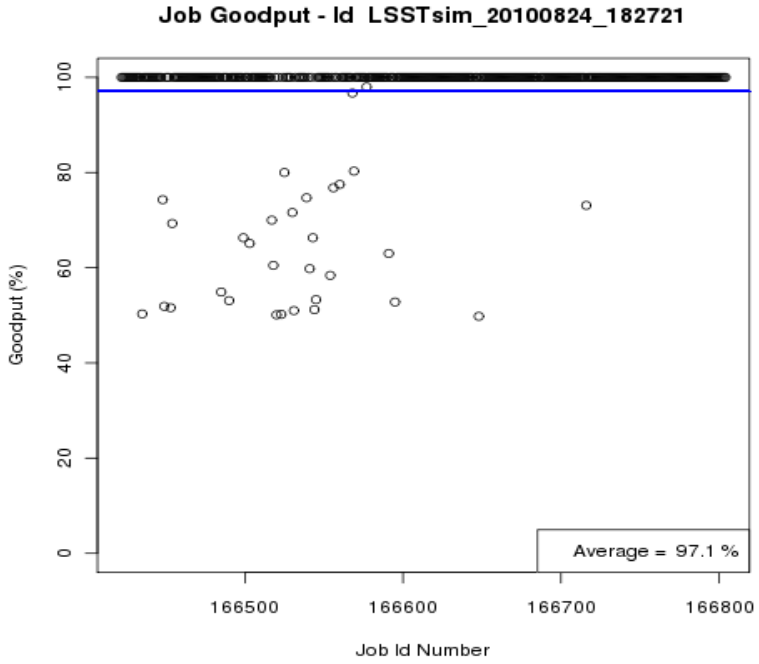
3. OSGMM installs the data there.
4. After all the static data is installed, the glideinWMS pilots check for that data. glidinWMS then only runs jobs where the data is.

Performance for glideinWMS

Define

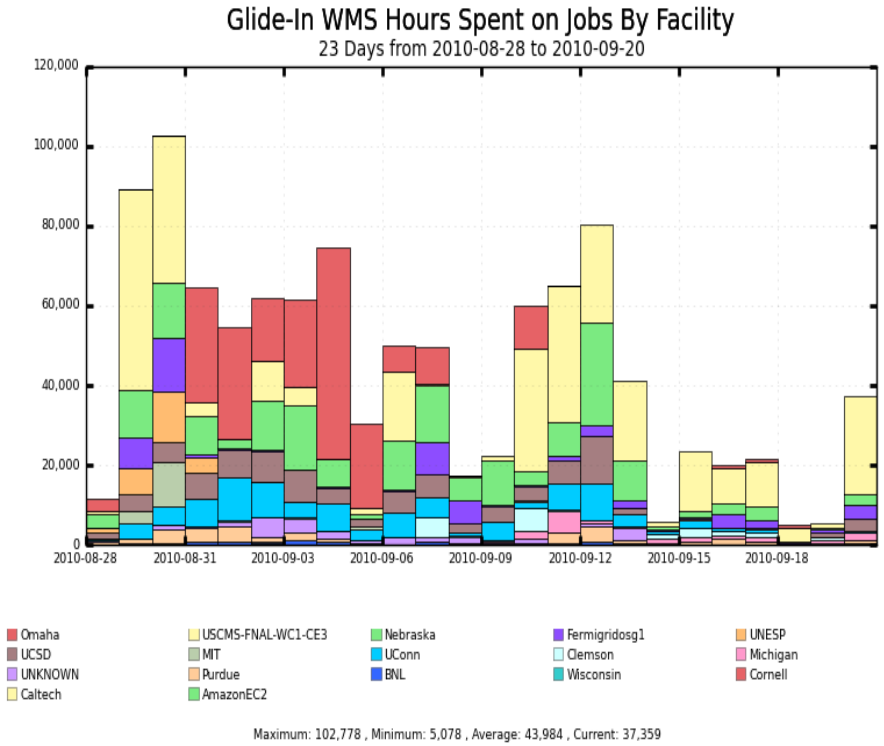
Goodput = CPU time used/Wall time

Wall time includes time for file transfer and other system calls. This ratio is close to 100% for LSST, ignoring restarted runs, so file transfer time is small.

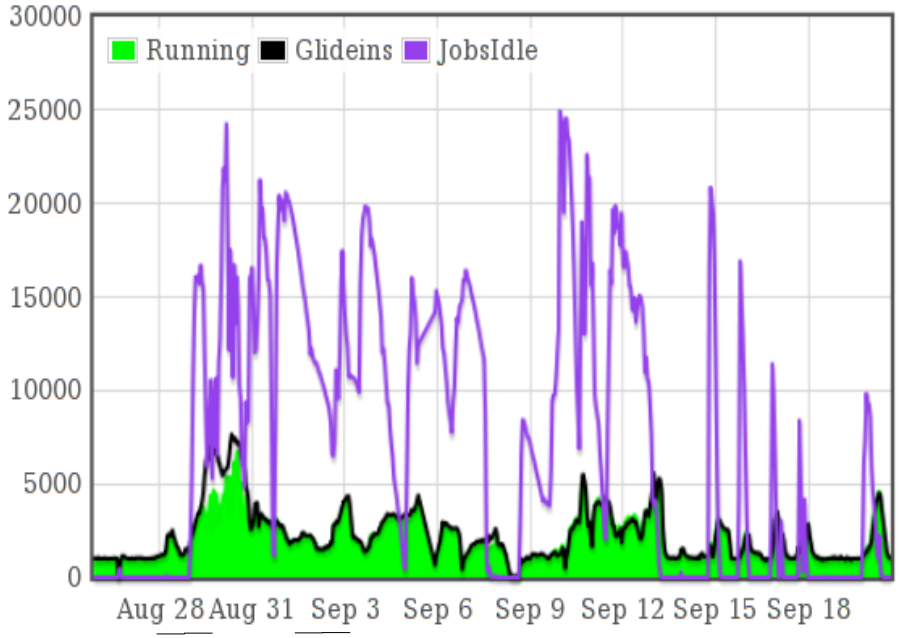


Actual Production

By Sep 3, produced 150 pairs in 5 days using 13 sites.
400 / 529 pairs are produced (some chips job may require recovery)



Gratia Resource Utilization plots



150 pairs produced

Frontend Status: Jobs & Glideins

– see http://gratia-osg-prod-reports.opensciencegrid.org/gratia-reporting/

Some Details

- Two ways to get static data from submit host to sites:
  - Initiate the transfer from the CEs. *Used by LSST.*
  - Initiate the transfer from the submit host. *May even out load on the server that stores it but infrastructure for this is not complete.*

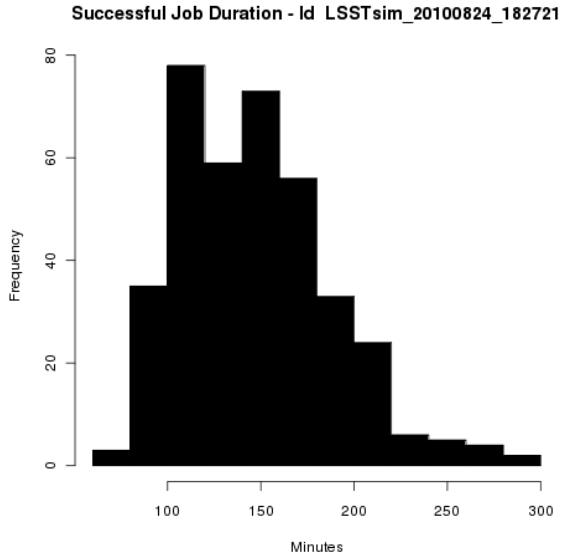
- OSGMM is due to be phased out in 2012.

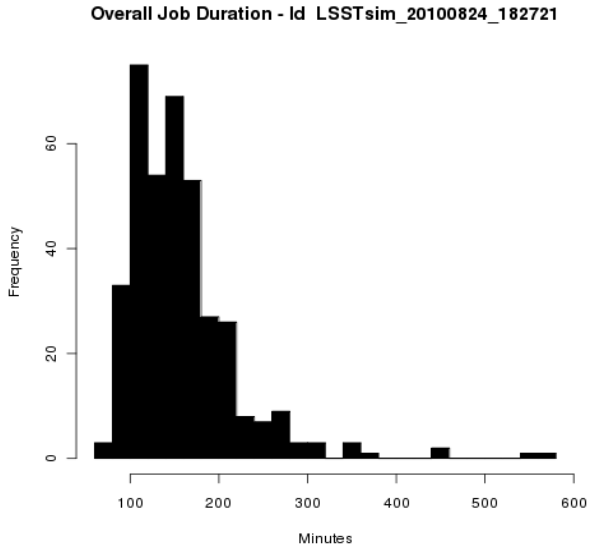
- Transfers may take a long time. To get 7GB to some sites took overnight.

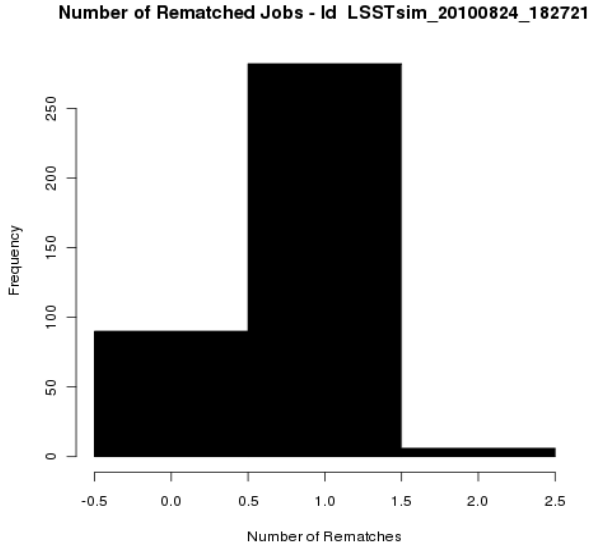
- No explicit catalog for output data, just pick directory names carefully.

Ramping Up Production

- Ramping up production took months, which is typical.
- Fix simulation program to be OSG compatible.
- Deal with limitations of grid sites:
  - Some batch systems didn't allot enough RAM
  - Storage unavailable due to maintenance at some of the most productive sites







Conclusions

We used two methods to move the LSST simulation data to and from the sites:

- direct transfer, controlled by OSGMM, for static data,
- and glideinWMS for data that changes from run to run.

This efficiently gets the data to where it's needed.

– see https://twiki.grid.iu.edu/bin/viewauth/Engagement/EngageLSSTPhase2

Acknowledgements: Based on contributions from the OSG Task Force—especially Brian Bockelman, Parag Mhashilkar, and Derek Weitzel —Bo Xin from LSST, Chris Green, Tanya Levshina, and Gabriele Garzoglio.