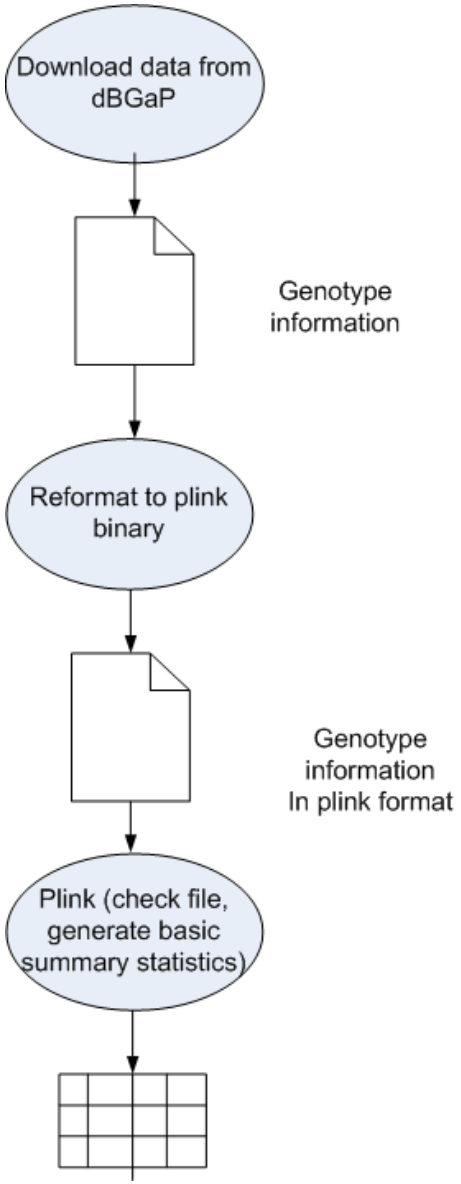
Tricks of the Trade for Running Workflows on HPC Systems

Gideon Juve

Information Sciences Institute
University of Southern California
gideon@isi.edu

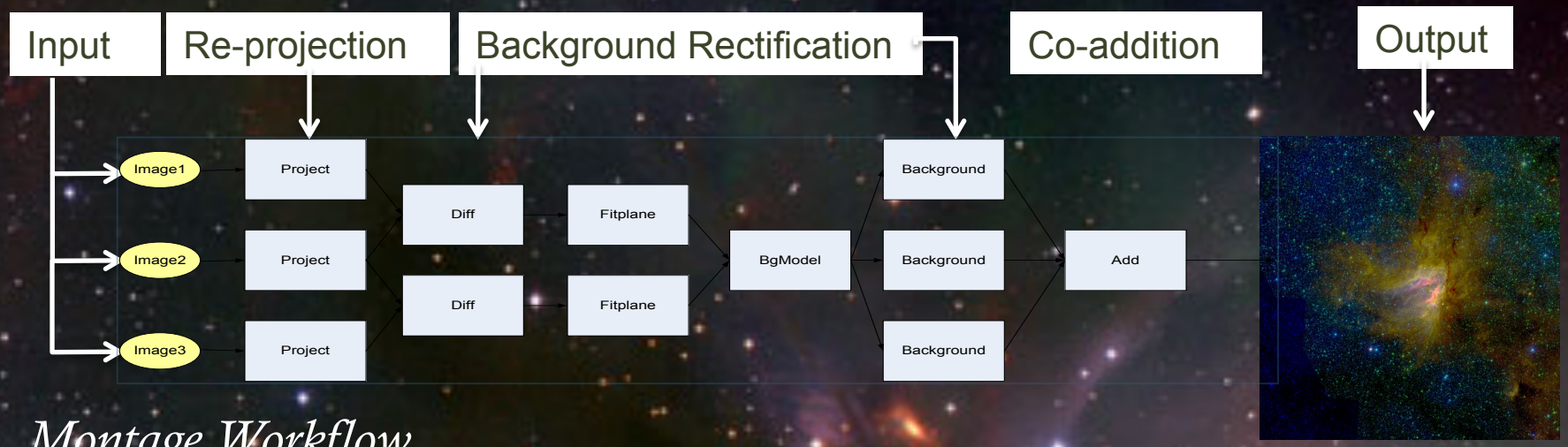
Scientific Workflows

- Enable **automation** of complex, multi-step pipelines
- Provide **reliable** execution on unreliable infrastructure
- Support **reproducibility** of computations
- Can be **shared** and **reused** with other data/ parameters/algorithms
- Enable recording of data **provenance**
- Support **distributed**, **parallel** execution to reduce time to solution



Science-grade Mosaic of the Sky

Science-grade Mosaic of the Sky



Montage Workflow

Size of mosaic in degrees square	Number of input data files	Number of tasks	Number of intermediate files	Total data footprint	Cummulative wall time
1	84	387	850	1.9 GB	21 mins
2	300	1442	3176	6.8 GB	54 mins
4	685	3738	8258	18 GB	3 hours, 18 mins
6	1461	7462	16458	37 GB	7 hours, 7 mins
8	2565	12757	28113	64 GB	11 hours, 44 mins

HPC versus HTC

- **High Performance Computing**
  - Solve one large problem
  - Single job performance
  - Low-latency network
  - Homogeneous
  - Parallel file system
  - PBS
  - Parallel (MPI) Jobs
  - Capability

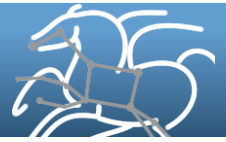
- **High Throughput Computing**
  - Solve many small problems
  - Workload performance
  - Commodity network
  - Heterogeneous
  - No shared file system
  - Condor
  - Serial or Multi-threaded Jobs
  - Capacity

Workflows as HTC Applications

- Throughput is more important than peak performance
- Care about time to finish entire workflow
- Loosely-coupled
- Workflow jobs are typically serial or multithreaded
- Usually contain lots of small tasks

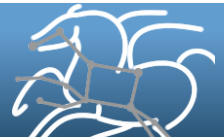
Most workflows are HTC applications

Workflow	Tasks	Avg Task Duration (s)	Avg I/O Read (MB)	Avg I/O Write (MB)	Peak Memory (MB)
Montage	10,429	1.7	14.3	4.9	17
CyberShake	815,823	40.6	272.8	1.2	1870
Broadband	770	44.3	1558.2	233.3	942
Epigenome	529	50.7	46.7	10.4	197
LIGO	2,041	90.3	105.0	0.0	969
SIPHT	31	142.8	56.2	48.9	116



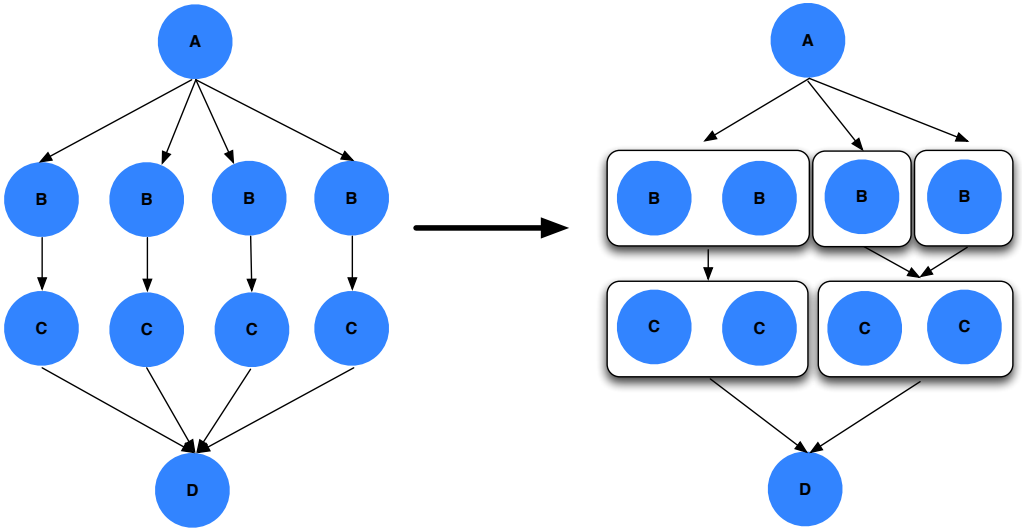
Workflows on HPC Systems

- **Much of the available infrastructure was designed for HPC**
  - Results in many problems for workflows
- **Queue Delays**
  - Significantly decreases throughput
- **Unfavorable Policies**
  - Max jobs queued
  - Priority for large jobs
  - Can't express application policies
- **Mismatched I/O patterns**
  - Parallel file system designed for few, large parallel files, not lots of small files
- **Lack of (remote) APIs for job submission**
  - Many systems do not deploy GRAM, UNICORE, etc.
- **Security Policies**
  - Firewalls prevent access outside local network
  - 2 factor authentication cannot be automated

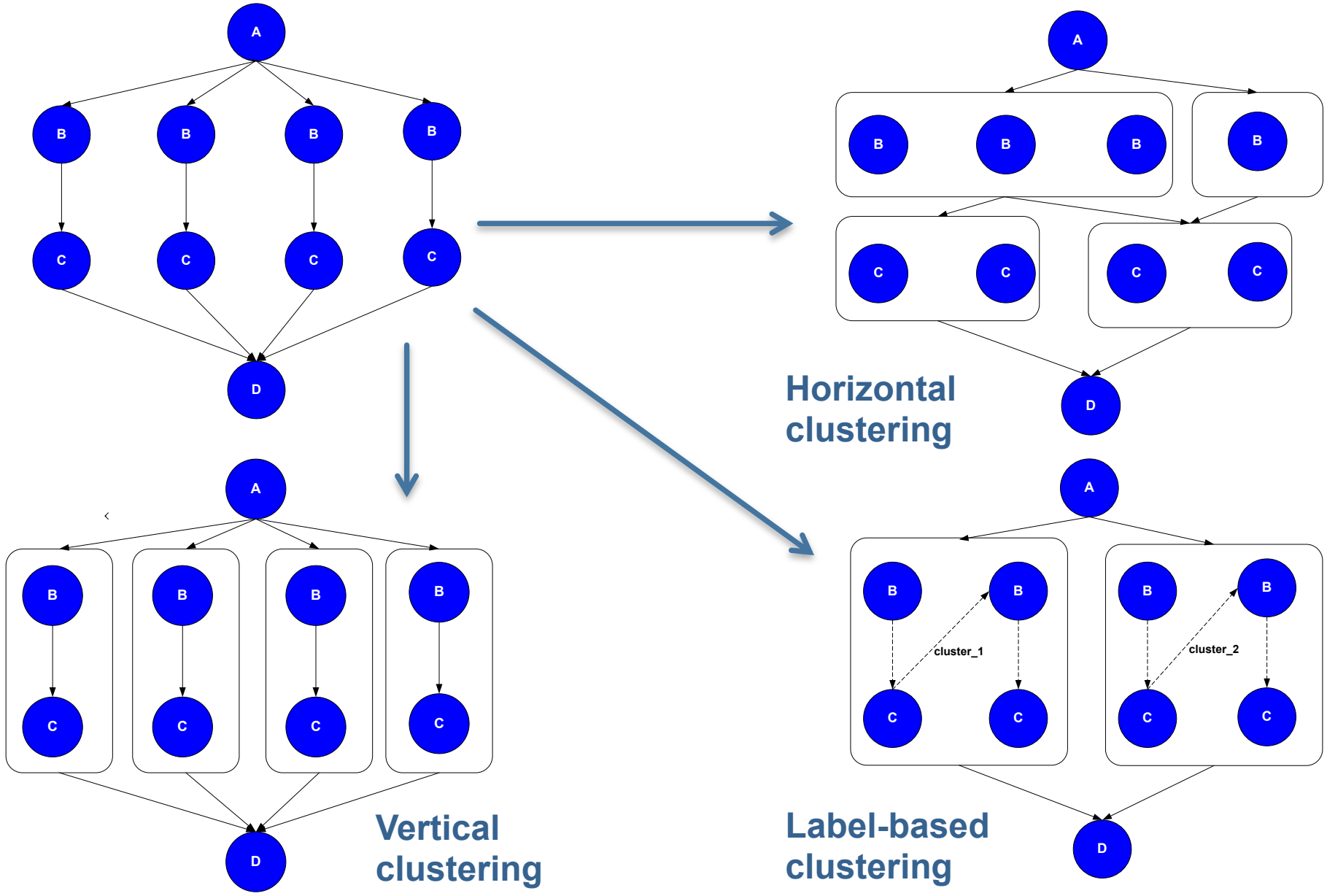


Task Clustering

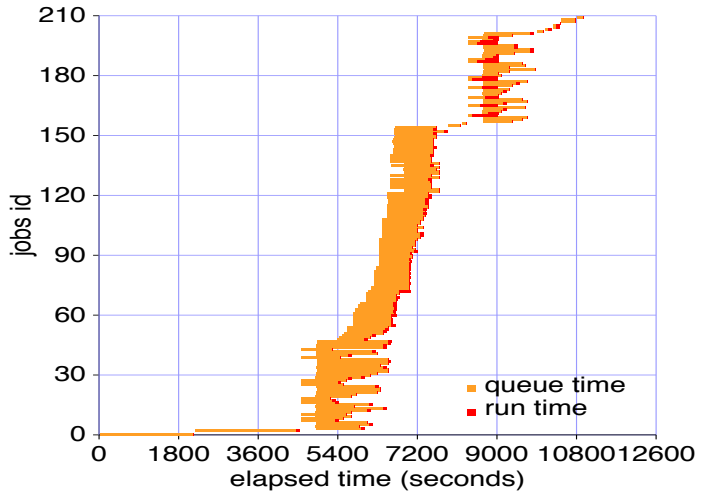
- **Cluster short-running tasks together to reduce queue delays and achieve better performance**
- **Why?**
  - Each job has scheduling overhead – need to make overhead worthwhile
  - Ideally users should run a job that takes at least 10-30 minutes
  - Clustered tasks can reuse common input data – less data transfers

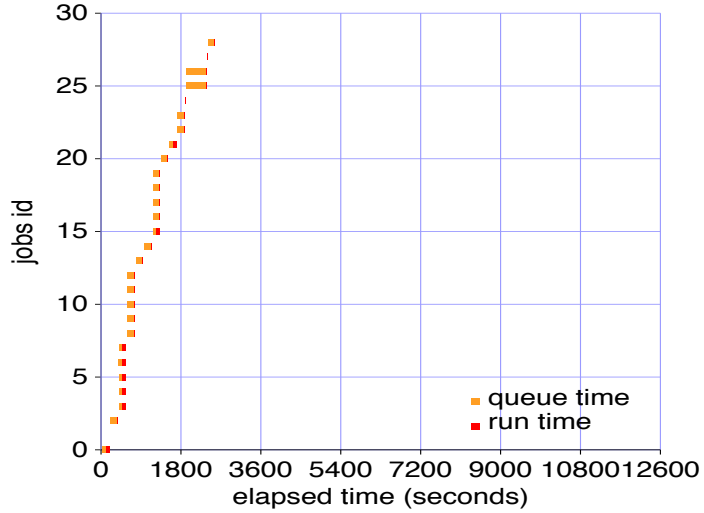


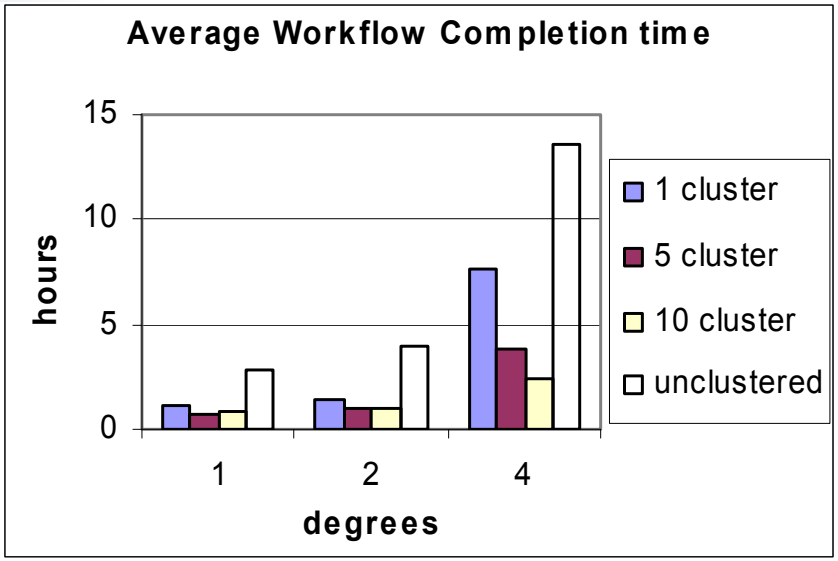
Level-based clustering


A
B
B
B
B
C
C
C
C
D
Horizontal clustering
A
B
B
B
B
C
C
C
C
D
Vertical clustering
A
B
B
B
B
C
C
C
C
D
Label-based clustering
A
B
B
B
B
cluster_1
cluster_2
C
C
C
C
D

Task Clustering Results



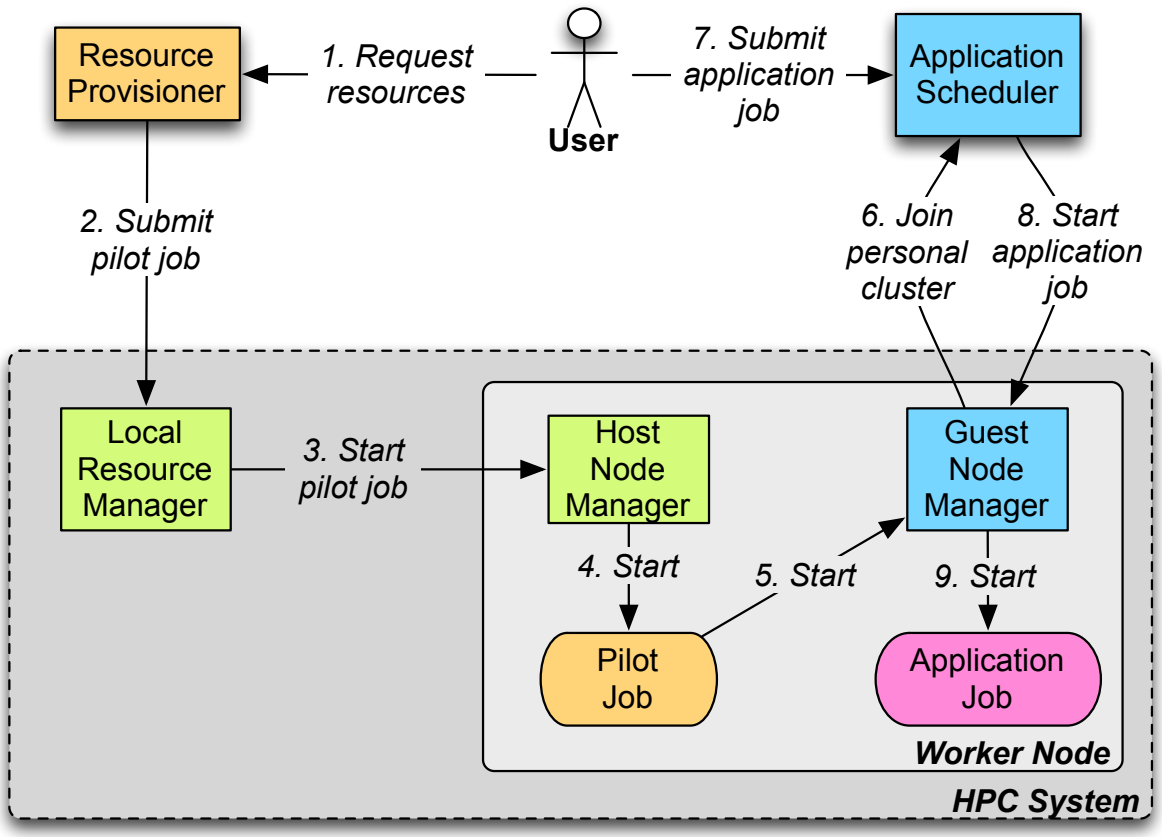




- Finding the "best" clustering parameters can be difficult
- Too much clustering can sacrifice parallelism

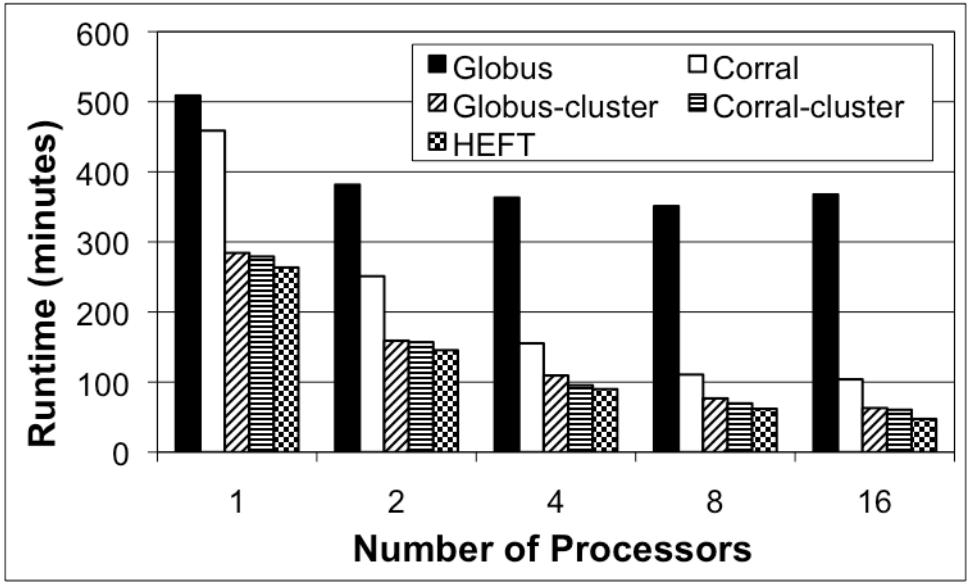
Pilot Jobs

- Key idea: Use HPC scheduler to run application scheduler
- Parallel pilot jobs
- Amortize queue delays over many jobs
- Apply application-specific policy



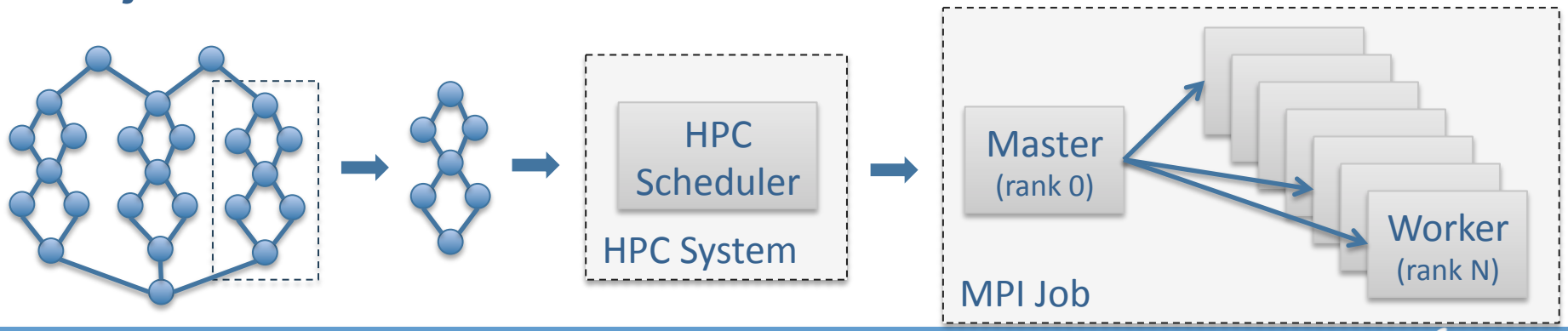
Pilot Jobs

- Lots of different pilot job systems have been developed
  - DIRAC, PanDA, glideinWMS, Corral
- Benefits
  - Higher throughput
  - Lower makespan
  - Better resource utilization
  - Reduced load on LRM
  - Easier to compete
- Drawbacks
  - Complexity
  - User infrastructure
  - Resource Provisioning
- Too complex for average users



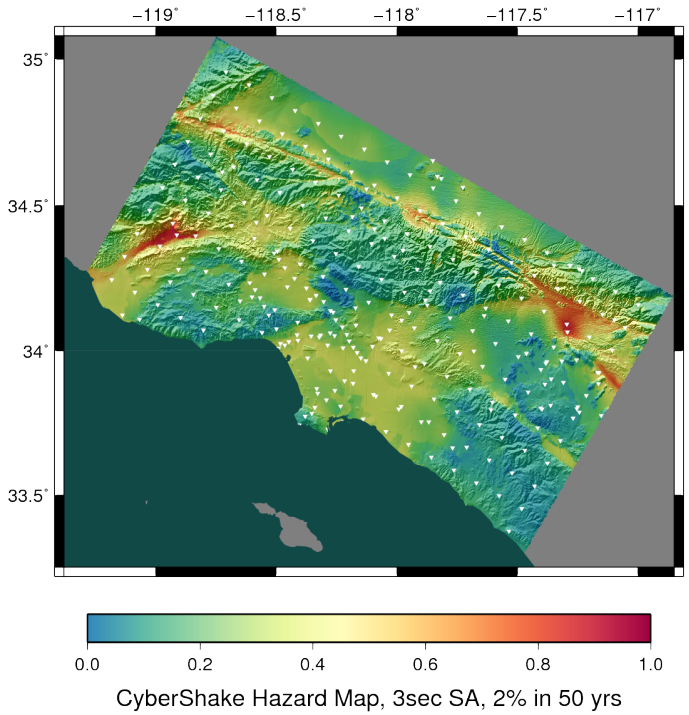
Pegasus-MPI-Cluster

- A master/worker task scheduler for running fine-grained workflows on batch systems
- Runs as an MPI job
  - Uses MPI to implement master/worker protocol
- Works on most HPC systems
  - Requires: MPI, a shared file system, and fork()
- Allows sub-graphs of a workflow to be submitted as monolithic jobs to remote resources



Southern California Earthquake Center

CyberShake PSHA Workflow

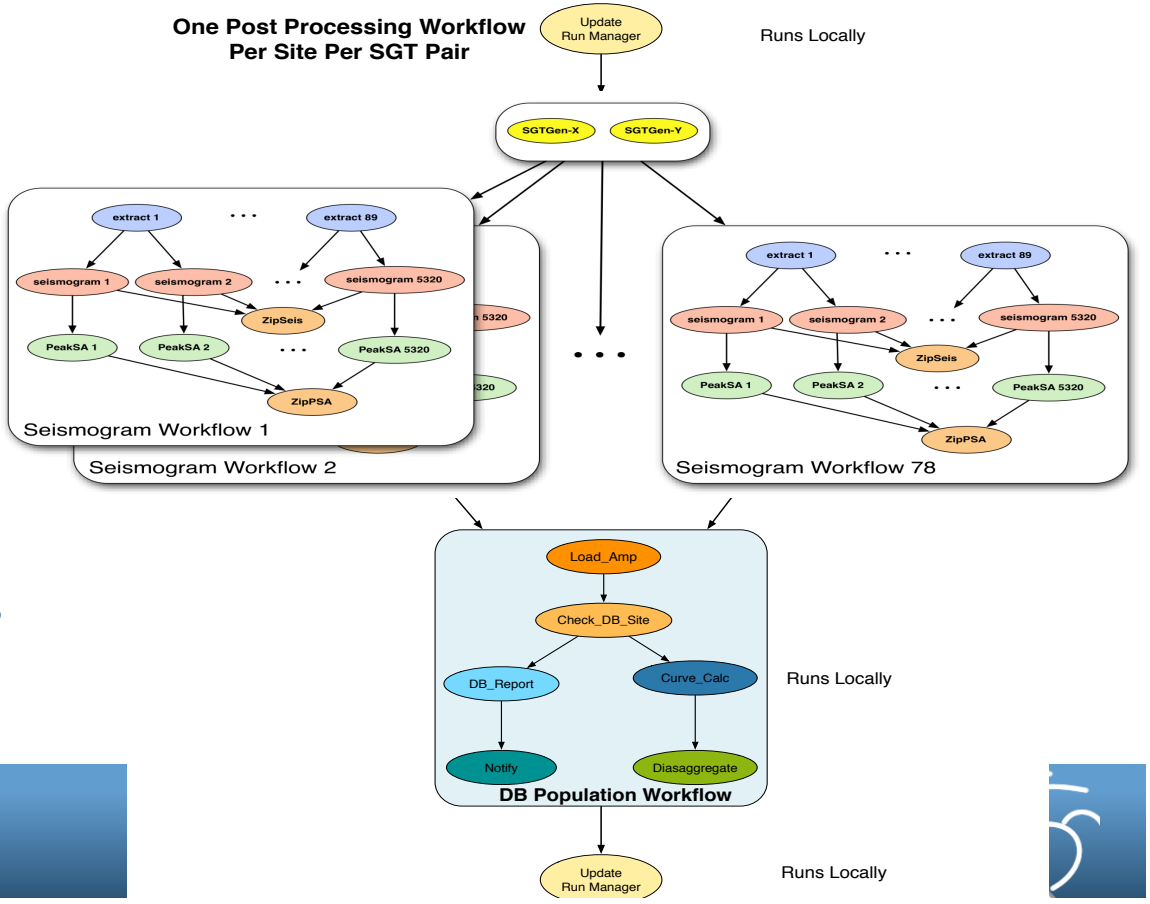


CyberShake Hazard Map, 3sec SA, 2% in 50 yrs

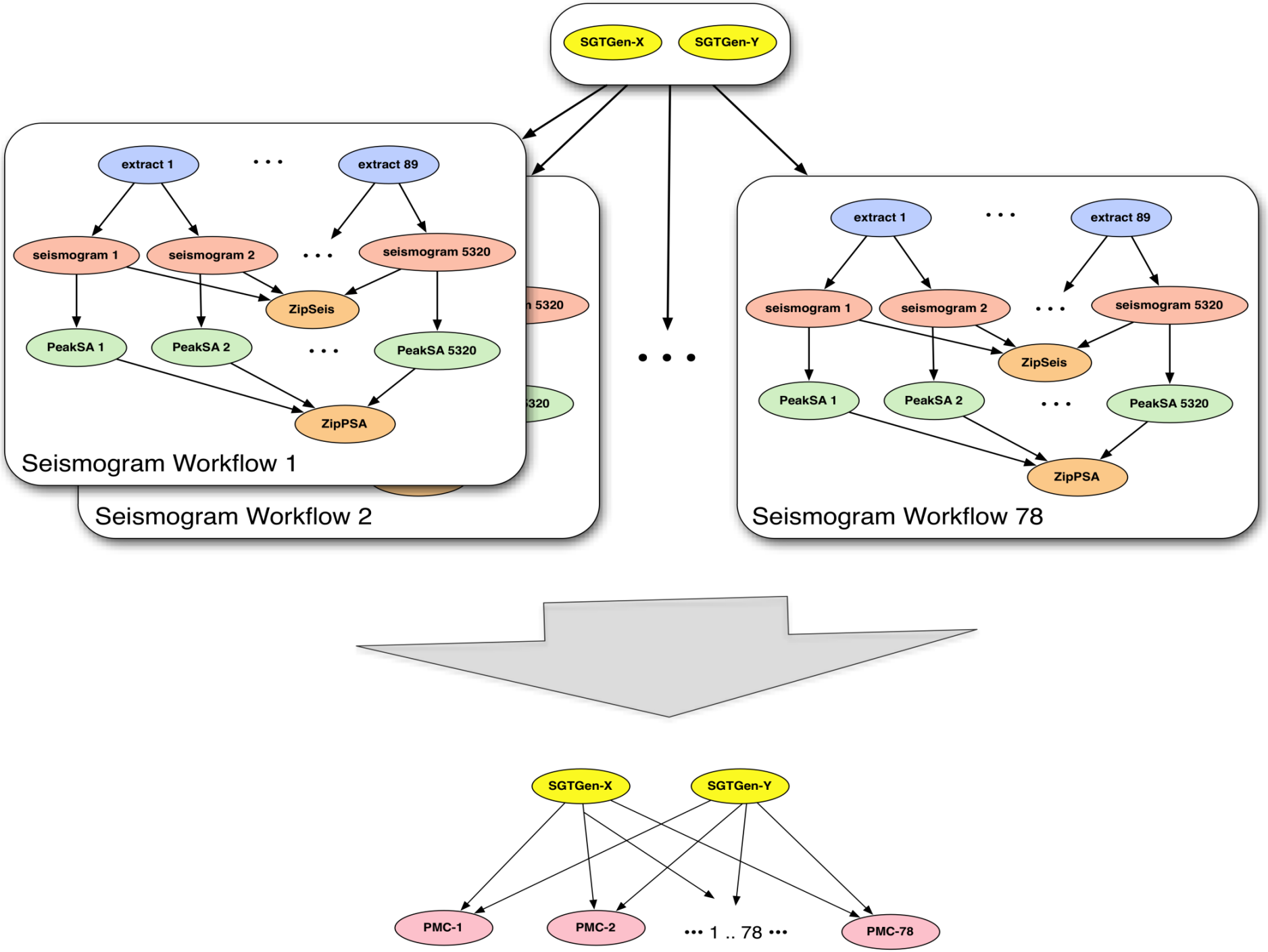
286 Sites, 4 models

- **Each site = one workflow**
- **Each workflow has 420,000 tasks in 21 jobs**

- Builders ask seismologists: "What will the peak ground motion be at my new building in the next 50 years?"
- Seismologists answer this question using Probabilistic Seismic Hazard Analysis (PSHA)



USC Viterbi School of Engineering


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

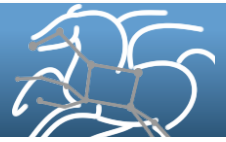
I/O Forwarding in PMC

- **Parallel file systems are designed for large, parallel files**
  - Striping, concurrent writes, etc.
- **Workflows generate lots of small files**
  - Metadata-intensive, no concurrent writes, many files in one directory

Type	Files / Workflow	Avg Size
Seismogram	404,864	200K
PSA	404,864	200B

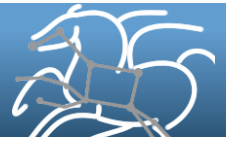
CyberShake File Stats

- **PMC has a feature called I/O forwarding for this case**
  - Each worker opens a pipe to the tasks as it forks them
  - Task writes output on pipe
  - PMC uses MPI messages to transfer data
  - One process aggregates data and writes it to a file
  - Requires no modification to application code



Dedicated Head Node

- **No remote job submission interface? Run on head node!**
  - Policy often prevents users from having long-running processes on head node (e.g. workflow systems)
  - Some systems do allow it (Titan), others are willing to install a node for you if you pay for it (HPCC)
- **Benefits**
  - Submit directly to HPC scheduler
  - No firewall issues
  - Great for communities of users
- **Drawbacks**
  - Inconvenient
  - Cost
  - Administration

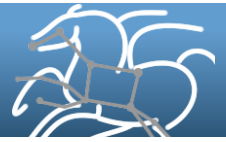


Workflows on Titan

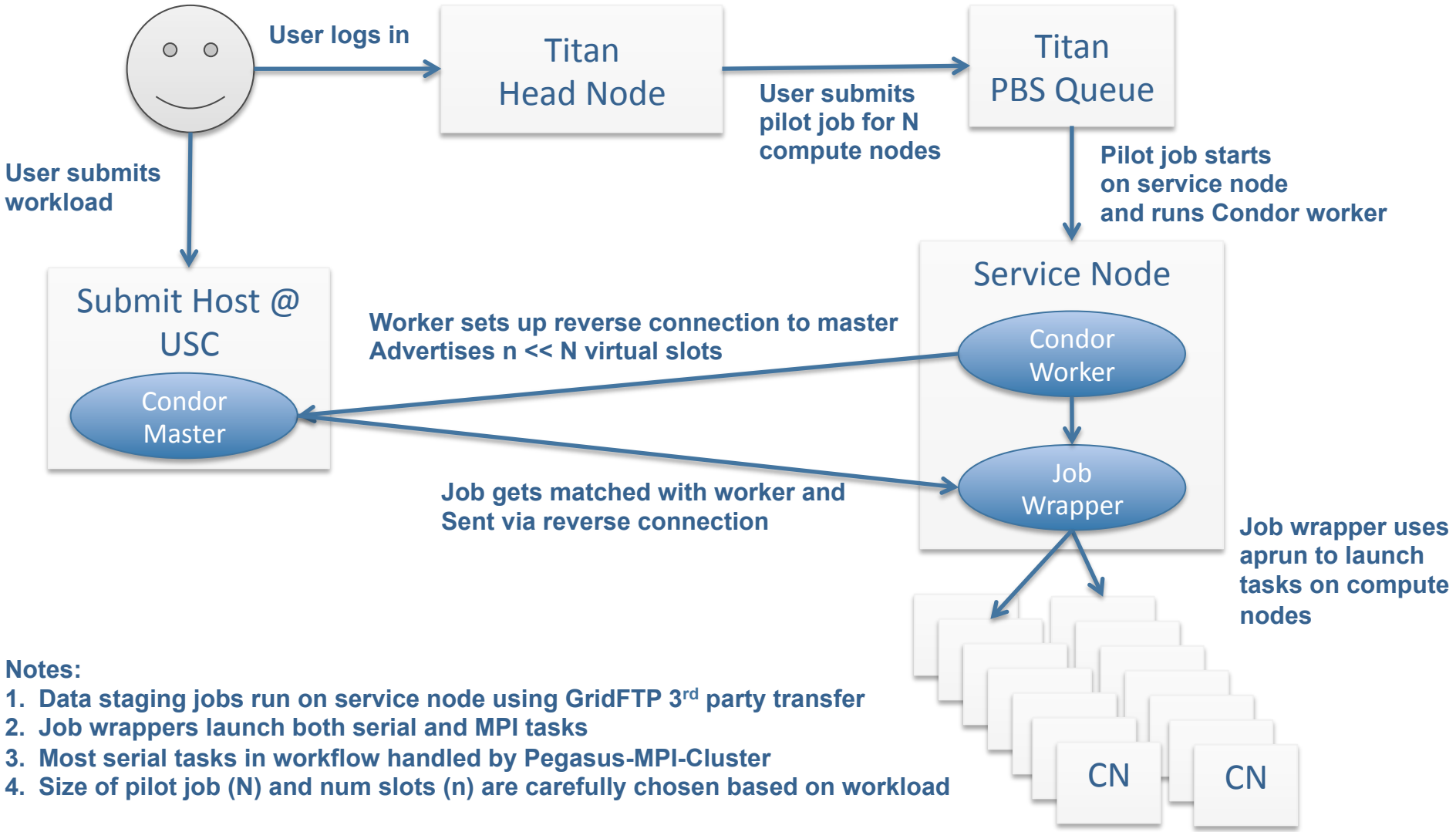
- Titan has no remote job submission interface
  - Other systems like Kraken and Blue Waters have GRAM
  - Security policy prohibits GRAM and similar on Titan
  - Incoming connections require 2-factor authentication
- Running on head node is possible, but very inconvenient
- Solution: Run pilot jobs

Pilot Jobs on Titan: Challenges

- How to enable network connections from USC to Titan?
  - Network policy prevents incoming connections to Titan w/o 2-factor auth
  - Solution: Condor connection brokering
  - Condor worker makes persistent *outgoing* connections to Condor master at USC, which arranges connections (similar to passive FTP)
- Where to run Condor worker?
  - Compute nodes can't talk to outside network
  - Solution: Use aprun from service nodes to launch jobs on compute nodes
- How to run MPI jobs?
  - Condor has very poor support for MPI
  - Use wrapper scripts to call aprun directly
- Submitting pilot jobs is still a problem (resource provisioning)



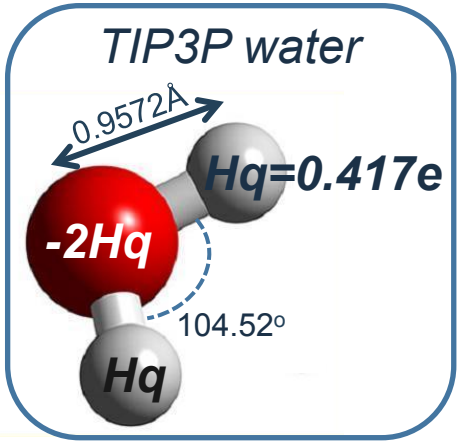
Pilot Jobs on Titan: How it works



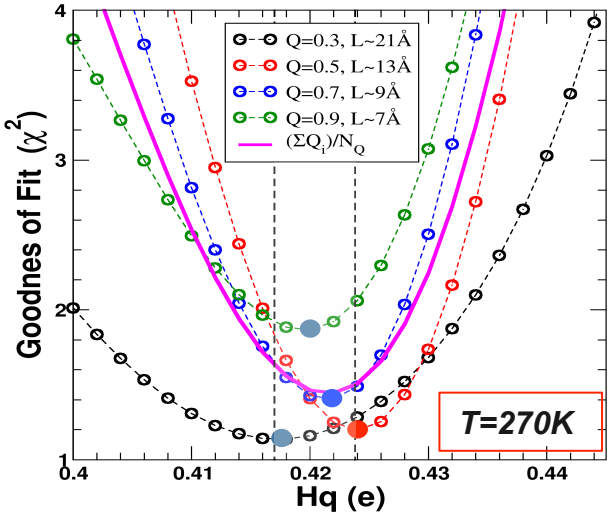
Notes:

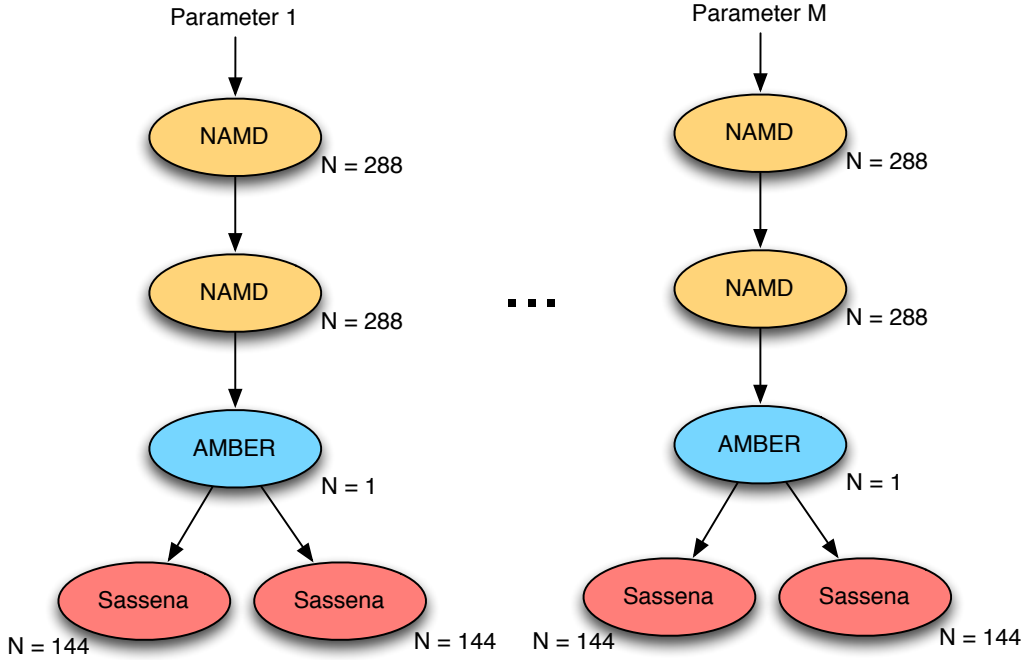
1. Data staging jobs run on service node using GridFTP 3rd party transfer
2. Job wrappers launch both serial and MPI tasks
3. Most serial tasks in workflow handled by Pegasus-MPI-Cluster
4. Size of pilot job (N) and num slots (n) are carefully chosen based on workload

SNS Refinement Workflow



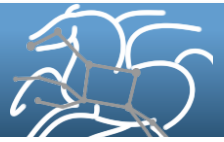
- Trying to fit parameter to experimental data to improve water model
- Pipeline of simulations for each value (M=20)
- Simulation code is MPI





Pegasus-HPC-Cluster

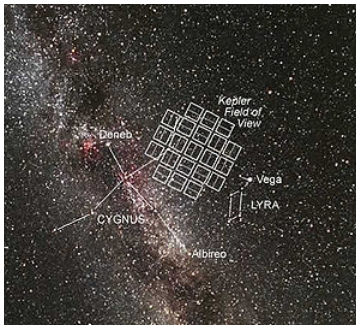
- **Problem: PMC cannot handle MPI jobs**
  - MPI launching MPI is tricky
  - PMC scheduler is pull-based (not good for parallel jobs)
- **Pegasus-HPC-Cluster: Like PMC, but for MPI jobs**
  - Task graph contains MPI jobs
  - PHC job starts running on service node (or PBS MOM)
  - PHC schedules parallel jobs on available compute nodes
  - aprun/mpiexec used to launch MPI jobs on compute nodes
  - PHC workflow can contain PMC jobs
- **Work in progress**

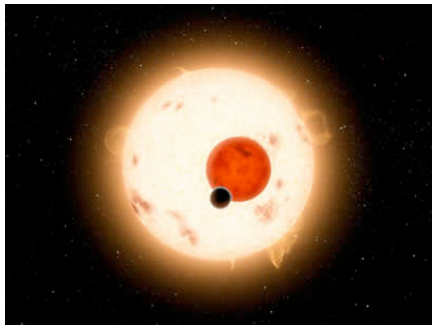


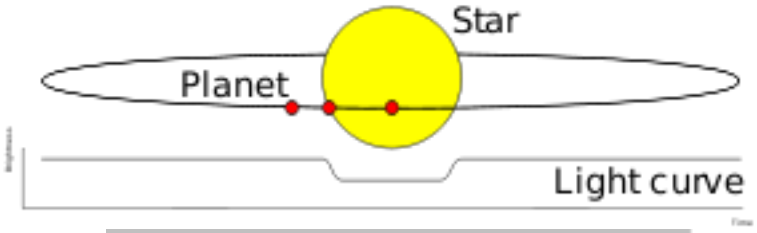
Hunting Exoplanets with Kepler

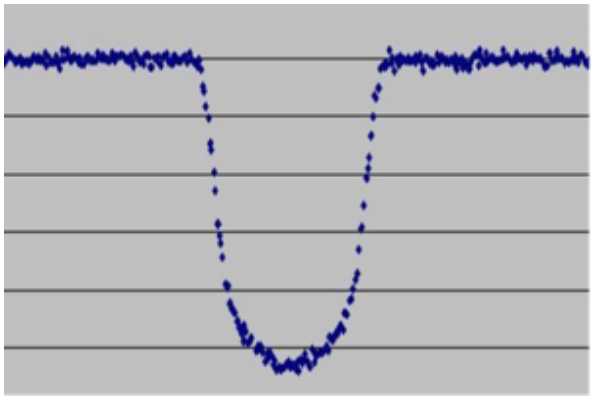
http://kepler.nasa.gov

- Kepler continuously monitors the brightness of over 175,000 stars
  - Search for periodic dips in signals as Earth-like planets transit in front of host star.
- Need to perform a bulk analysis of all the data when it is released to search for these periodic signals
- Over 380,000 light curves have been released (x 3 algorithms x 2 parameter sets = 2.2 M tasks)







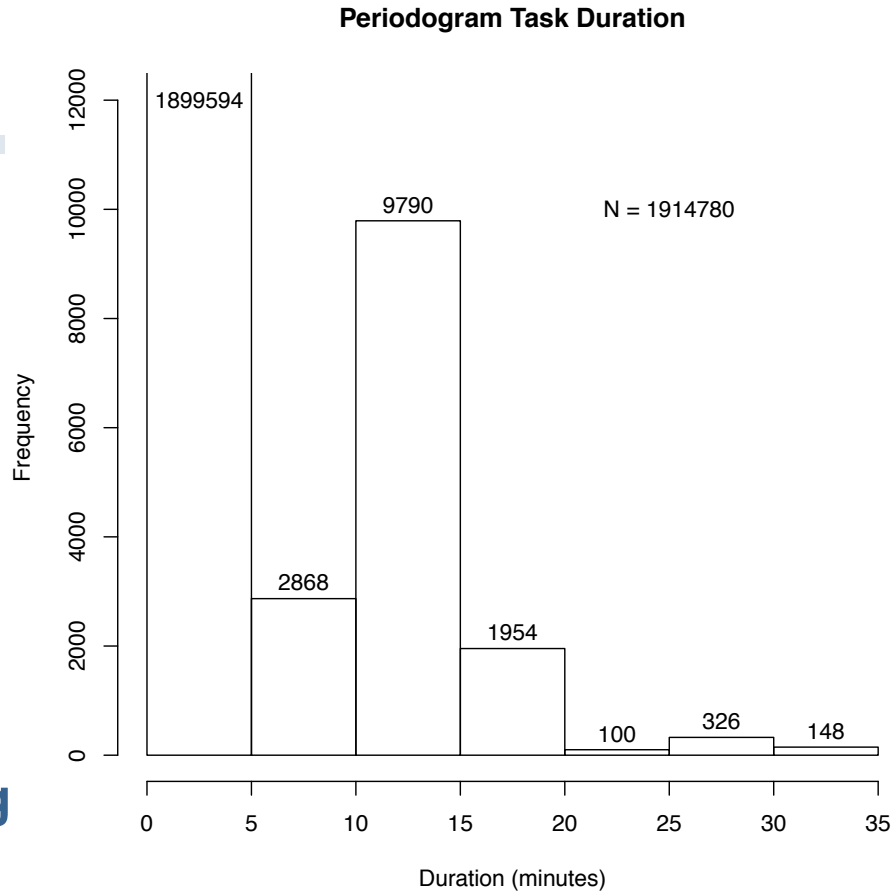


Kepler 6-b transit

Kepler Evolution

- Lots of small tasks
- A few large tasks
- Runtime varies over a wide range

2010: Pilot jobs / glideins
2011: Time-based task clustering
2012: Pegasus-MPI-Cluster



Year	Site	Inputs	Input Size	Outputs	Output Size	Jobs	Tasks	CPU Cores	CPU Hours
2010	Amazon EC2	210 K	17.3 GB	2.53 M	182 GB	51 K	1.26 M	128	2,417
2010	TACC Ranger	210 K	17.3 GB	1.26 M	3 TB	25 K	632 K	256	50,019
2011	Amazon EC2	210 K	17.3 GB	3.8 M	316 GB	7,065	632 K	256	5,300
2011	FutureGrid	210 K	17.3 GB	3.8 M	316 GB	7,065	632 K	256	5,300
2011	Open Science Grid	210 K	17.3 GB	3.8 M	316 GB	7,065	632 K	1300	5,300
2012	SDSC Trestles	1.1 M	1,650 GB	12.7 M	16 TB	372	2.2 M	640	101,614

Conclusion

- HPC and HTC are different
- Most scientific workflows are HTC applications
- There are some tricks to running workflows on HPC systems
  - Task clustering
  - Pilot jobs
  - Pegasus-MPI-Cluster / Pegasus-HPC-Cluster
  - Dedicated head node
  - Combinations of the above