USC VITERBI SCHOOL OF ENGINEERING

pegasus

# Hosted Science:
## *Managing Computational Workflows in the Cloud*

Ewa Deelman

USC Information Sciences Institute

http://pegasus.isi.edu

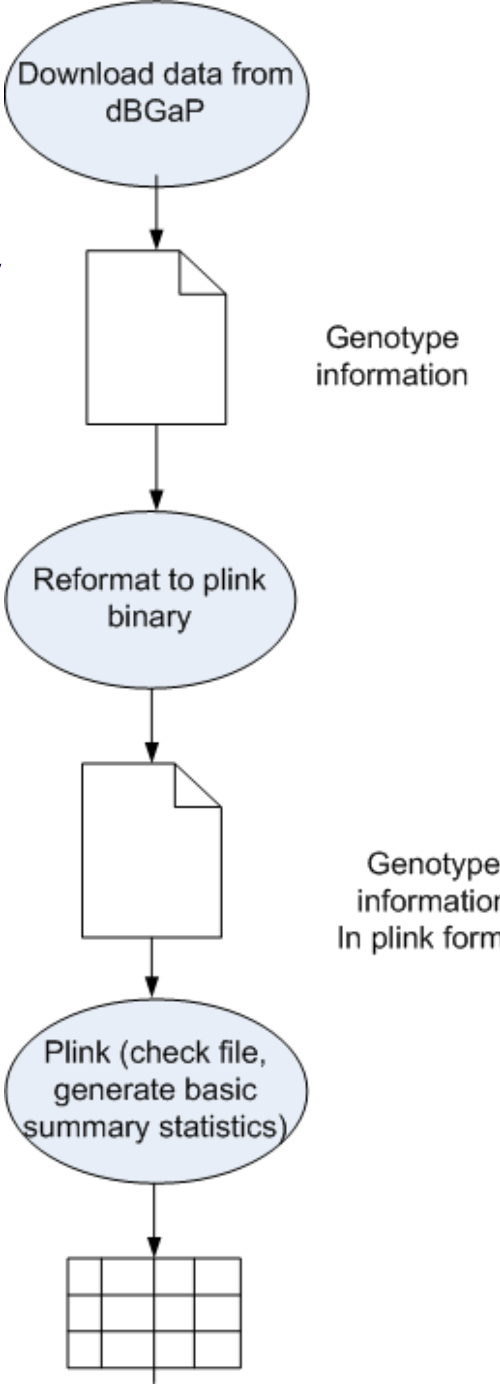
deelman@isi.edu

# The Problem

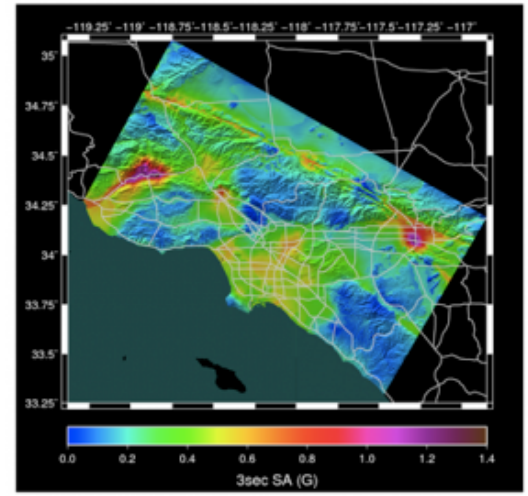
- Scientific data is being collected at an ever increasing rate
  - The “old days” -- big, focused experiments– LHC
  - Today “cheap” DNA sequencers – and an increasing number of them
- The complexity of the computational problems is ever increasing
- Local compute resources are often not enough (too small, limited availability)
- The computing infrastructure keeps changing
  - Hardware, software, but also computational models

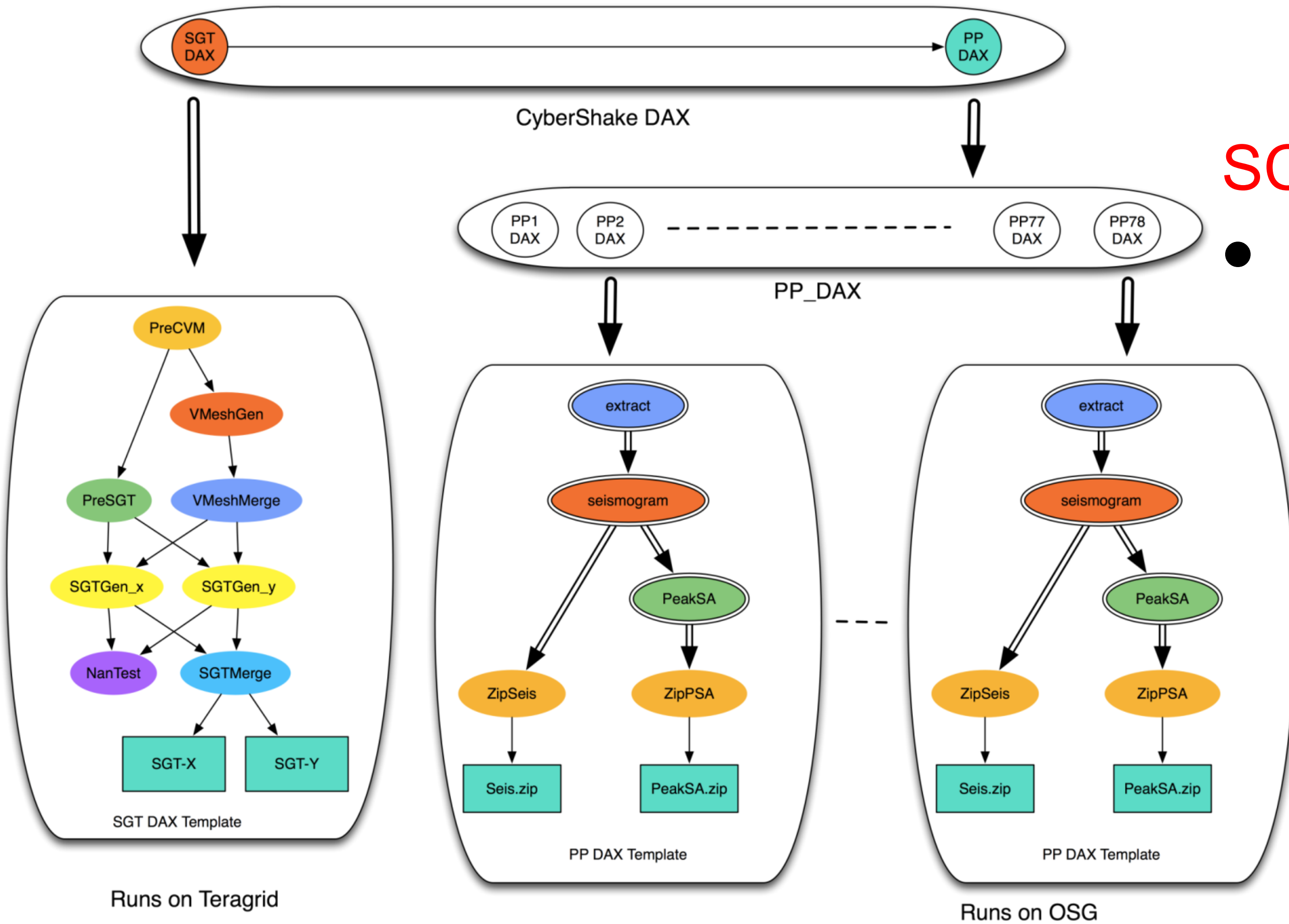
# Computational workflows --managing application complexity

- Help express multi-step computations in a declarative way
- Can support automation, minimize human involvement
  - Makes analyses easier to run
- Can be high-level and portable across execution platforms
- Keep track of provenance to support reproducibility
- Foster collaboration—code and data sharing



# So far applications have been running on local/campus clusters or grids



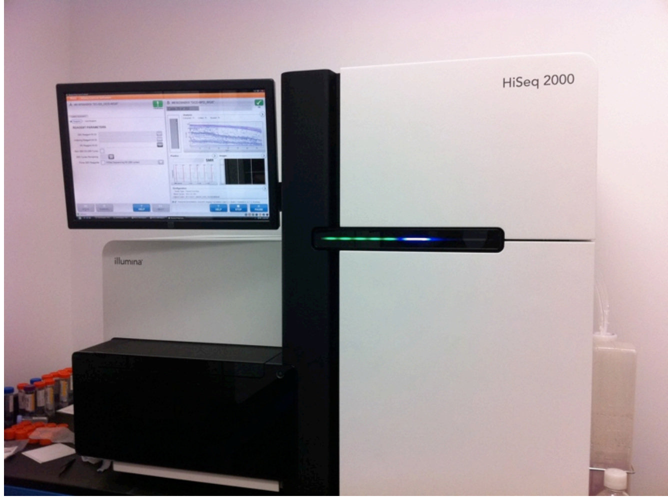




~ 850,000 tasks

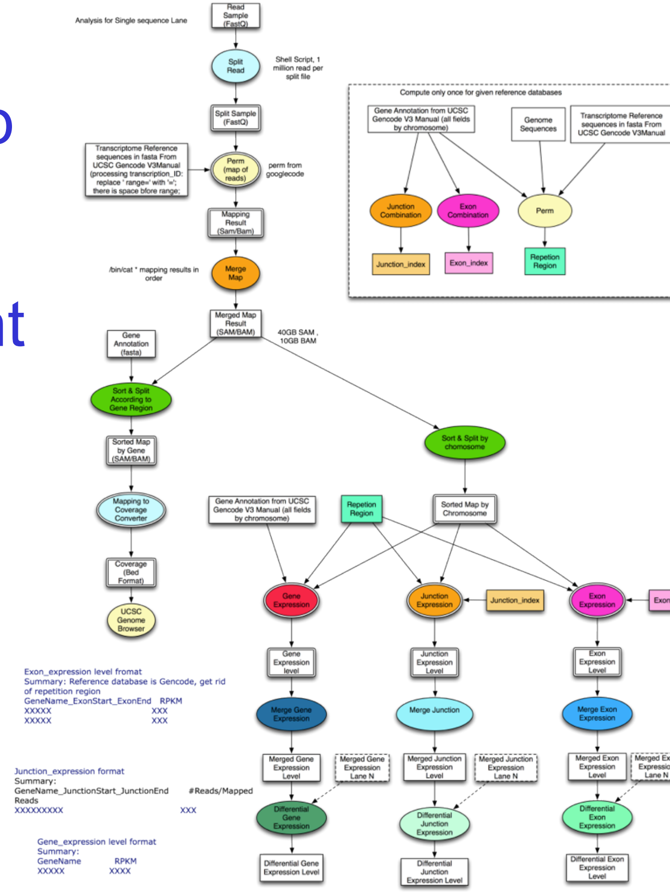
## SCEC CyberShake

- Uses physics-based approach
  - 3-D ground motion simulation with anelastic wave propagation
  - Considers ~415,000 earthquakes per site
    - <200 km from site of interest
    - Magnitude >6.5

# DNA sequencing, a new breed of data-intensive applications




- Data collected at a sequencers
  - Needs to be filtered for noisy data
  - Needs to be aligned
  - Needs to be collected into a single map
- Vendors provide some basic tools
  - you may want to try the latest alignment algorithm
  - you may want to use a remote cluster
- Challenges:
  - automation of analysis, reproducibility
  - Portability
  - provenance USERS!

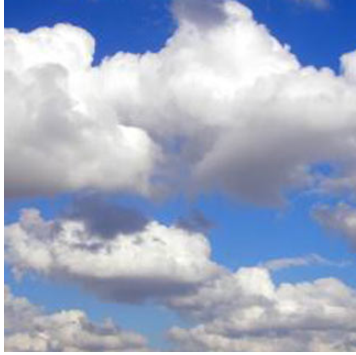


# Outline

- Role of hosted environments
- Workflows on the Cloud
  - Challenges in running workflows on the cloud
  - Data management aspects
- Hosted Science
  - Managing workflow ensembles on the cloud
  - Within user-defined constraints
- Conclusions

*Acknowledgements: Gideon Juve (USC), Maciej Malawski (AGH), Jarek Nabrzyski ( ND), Applications: Bruce Berriman et al (Caltech), Tom Jordan et al (USC), Ben Berman, James Knowles et al (USC Medical School), Pegasus Projects: Miron Livny (UWM), Gideon Juve, Gaurang Mehta, Rajiv Mayani, Mats Rynge, Karan Vahi, (USC/ISI)*

# New applications are looking towards Clouds

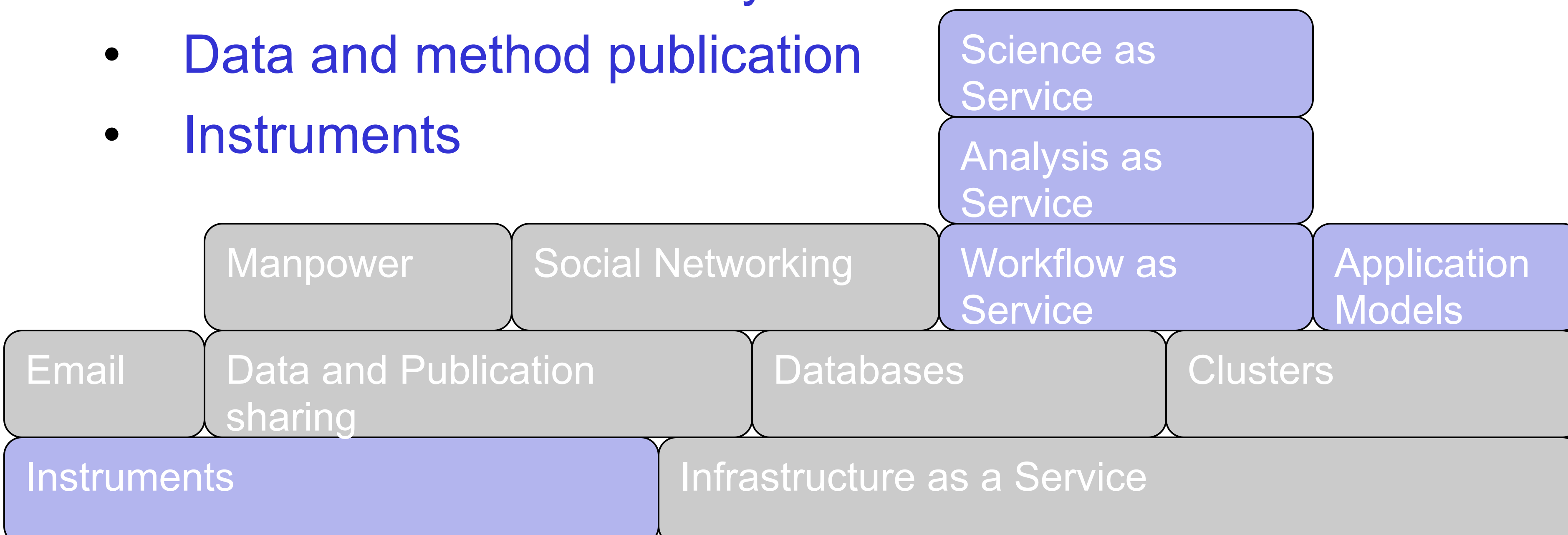


- Originated in the business domain
- Outsourcing services to the Cloud (successful for business)
- Pay for what you use, elasticity of resources
- Provided by data centers that are built on compute and storage virtualization technologies
- Scientific applications often have different requirements
  - MPI
  - Shared file system
  - Support for many dependent jobs

Google's Container-based Data Center in Belgium
http://www.datacenterknowledge.com/

# Hosted Science

- Today applications are using the cloud as a resource provider (storage, computing, social networking)
- In the future more services will be migrating to the cloud (more integration)
  - Hosted end-to-end analysis
  - Data and method publication
  - Instruments

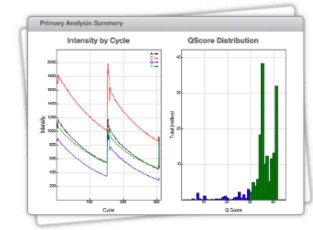


# The Future is Now Illumnia's BaseSpace

## Data Analysis

BaseSpace now performs one alignment and variant detection for free on all Illumina data! To learn more about what's included, click here.

BaseSpace makes data analysis easy. Push-button tools let researchers easily leverage all types of analysis applications and seamlessly view their results. Our flexible "app store" environment is being developed to bring the industry's best tools to your fingertips, with new tools added constantly.




Currently, BaseSpace can perform the following analyses on your data:

**RESEQUENCING ALIGNMENT**
Sequencing of an enriched portion of the human genome, or of a small genome (such as e.coli). Reads are aligned against the reference, and variants are noted.

**AMPLICON SEQUENCING**
Sequencing of PCR amplicons from probes targeting particular genome positions (up to ~384 loci from up to ~96 samples).

**DE NOVO ASSEMBLY**
Assembly of small (< 20MB) genome from 16S ribosomal RNA reads without the use of a genomic reference.

**SMALL RNA ANALYSIS**
Resequencing workflow applied to microRNAs.

**LIBRARY QC**
Fast resequencing of a reference genome to QC the DNA library

**METAGENOMICS**
The 16S metagenomics workflow is used to classify organisms from a metagenomic sample by amplifying specific regions in the 16S ribosomal RNA. The main output of this workflow is a classification of reads at several taxonomic levels (kingdom, phylum, class, order, family, genus)


Workflow times include dual surface scanning and v2 kits.

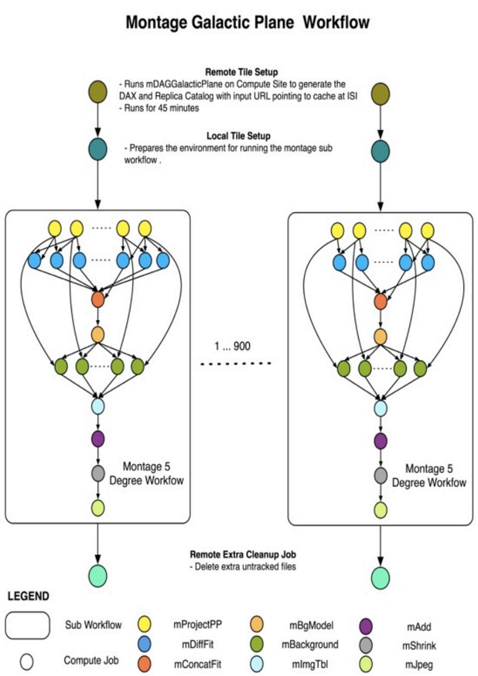
# Outline

- Role of hosted environments
- Workflows on the Cloud
  - Challenges in running workflows on the cloud
  - Data management aspects
- Hosted Science
  - Managing workflow ensembles on the cloud
  - Within user-defined constraints
- Conclusions

# Issues

- It is difficult to manage cost
  - How much would it cost to analyze one sample?
  - How much would it cost to analyze a set of samples?
  - The analyses may be complex and multi-step (*workflows*)
- It is difficult to manage deadlines
  - “I would like all the results to be done in a week”
  - “I would like the most important analyses done in a week”
  - “I have a week to get the most important results and $500 to do it”

# Scientific Environment
## How to manage complex workloads?


Work definition

Local Resource

Campus Cluster

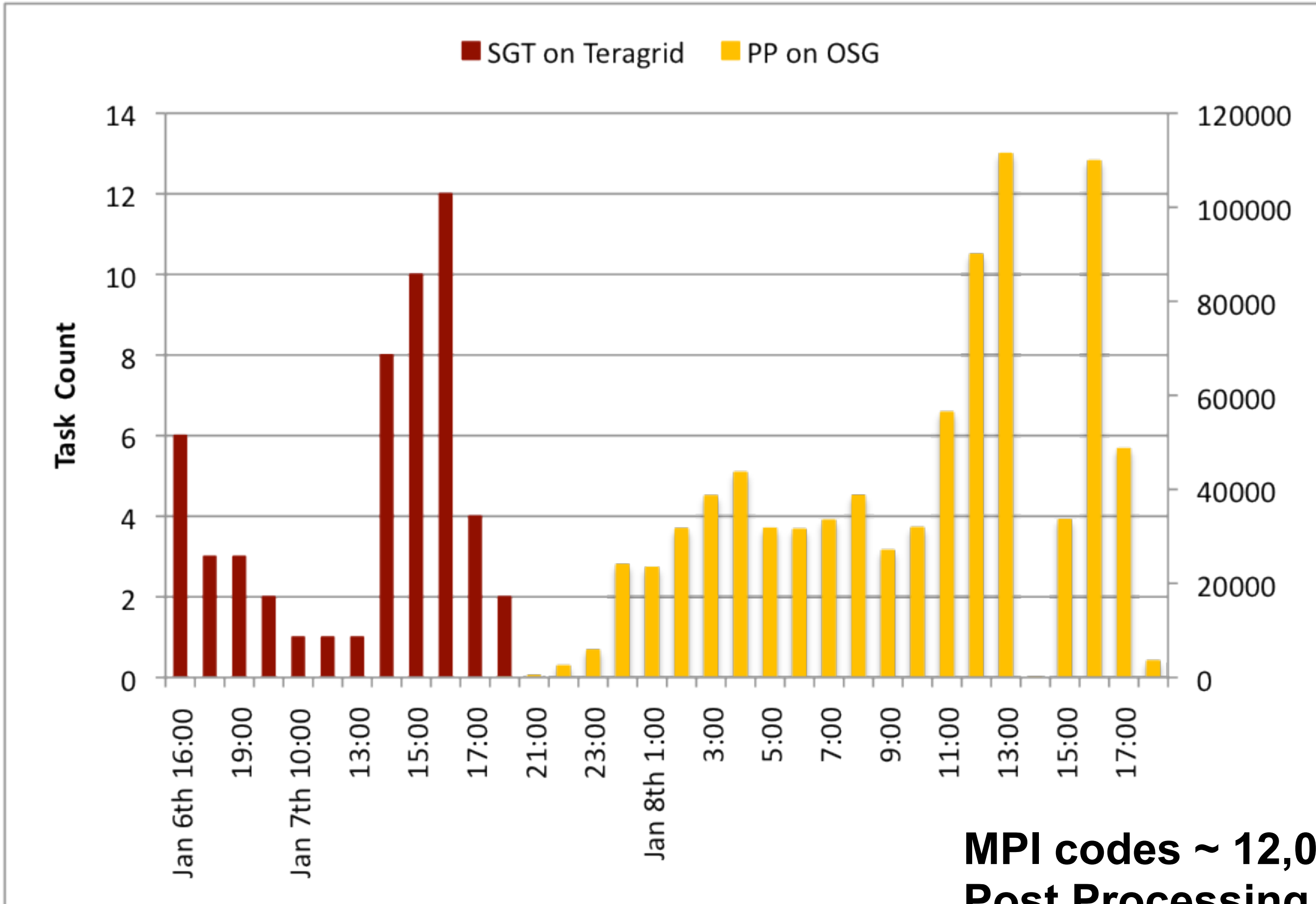
EGI

TeraGrid/XSEDE

Open Science Grid

Amazon Cloud

# Workflows have different computational needs --need systems to manage their execution




**SoCal Map needs 239 of those**

**MPI codes ~ 12,000 CPU hours,**
**Post Processing 2,000 CPU hours**
**Data footprint ~ 800GB**

**Peak # of cores on OSG 1,600**
**Walltime on OSG 20 hours, could be done in 4 hours on 800 cores**

# Workflow Management

You may want to use different resources within a workflow or over time

- Need a high-level workflow specification
- Need a planning capability to map from high-level to executable workflow
- Need to manage the task dependencies
- Need to manage the execution of tasks on the remote resources

- Need to provide scalability, performance, reliability

# Our Approach

- Analysis Representation
  - Support a declarative representation for the workflow (dataflow)
  - Represent the workflow structure as a Directed Acyclic Graph (DAG)
  - Use recursion to achieve scalability
- System (Plan for the resources, Execute the Plan, Manage tasks)
  - Layered architecture, each layer is responsible for a particular function
  - Mask errors at different levels of the system
  - Modular, composed of well-defined components, where different components can be swapped in
  - Use and adapt existing graph and other relevant algorithms

# Use the given Resources

Data Storage

data

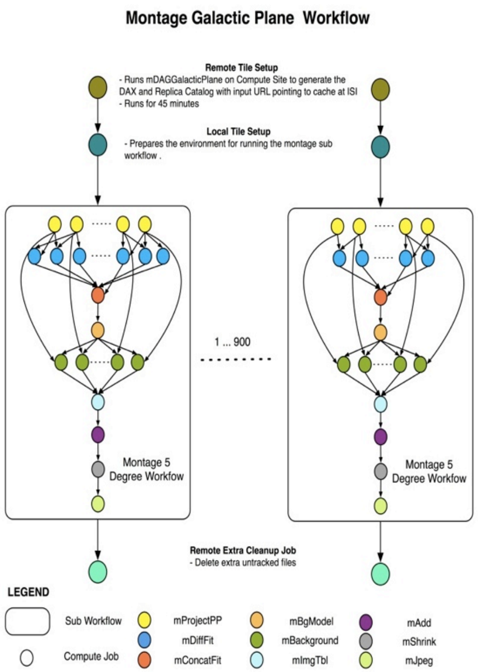
Campus Cluster

EGI

TeraGrid/XSEDE

Open Science Grid

**Amazon Cloud**




Work definition
As a WORKFLOW

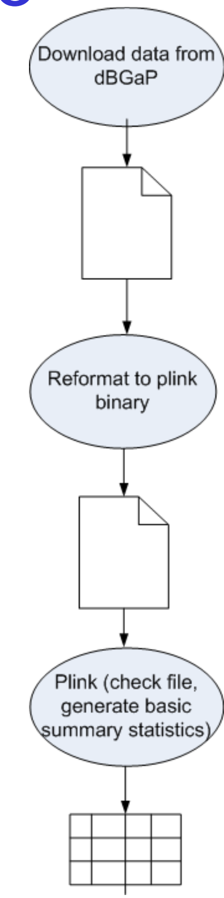
Workflow Management System

Local Resource

work

# Challenges of running workflows on the cloud

- Clouds provide resources, but the software is up to the user
- Running on multiple nodes may require cluster services (e.g. scheduler)
- Dynamically configuring such systems is not easy
  - Manual setup is error-prone and not scalable
  - Scripts work to a point, but break down for complex deployments
- Some tools are available
- Workflows need to communicate data—often through files, need filesystems
- Data is an important aspect of running on the cloud



# Outline

- Role of hosted environments
- Workflows on the Cloud
  - Challenges in running workflows on the cloud
  - Data management aspects
- Hosted Science
  - Managing workflow ensembles on the cloud
  - Within user-defined constraints
- Conclusions

# Workflow Data In the Cloud

## Executables

- Transfer into cloud
- Store in VM image

## Input Data

- Transfer into cloud
- Store in cloud

## Intermediate Data

- Use local disk (single node only)
- Use distributed storage system

## Output Data

- Transfer out of cloud
- Store in cloud

# Amazon Web Services (AWS)

IaaS Cloud, Services

- Elastic Compute Cloud (EC2)
  - Provision virtual machine instances
- Simple Storage Service (S3)
  - Object-based storage system
  - Put/Get files from a global repository
- Elastic Block Store (EBS)
  - Block-based storage system
  - Unshared, SAN-like volumes
- Others (queue, RDBMS, MapReduce, Mechanical Turk etc.)

*We want to explore data management issues for workflows on Amazon*

# Applications

- Not CyberShake SoCal map (PP) could cost at least $60K for computing and $29K for data storage (for a month) on Amazon (one workflow ~$300)
- Montage (astronomy, provided by IPAC)
  - 10,429 tasks, 4.2GB input, 7.9GB of output
  - I/O: High (95% of time waiting on I/O)
  - Memory: Low, CPU: Low

- Epigenome (bioinformatics, USC Genomics Center)
  - 81 tasks 1.8GB input, 300 MB output
  - I/O: Low, Memory: Medium
  - CPU: High (99% time of time)

- Broadband (earthquake science, SCEC)
  - 320 tasks, 6GB of input, 160 MB output
  - I/O: Medium
  - Memory: High (75% of task time requires > 1GB mem)
  - CPU: Medium

# Storage Systems

## Local Disk

RAID0 across available partitions with XFS

## NFS: Network file system

1 dedicated node (m1.xlarge)

## PVFS: Parallel, striped cluster file system

Workers host PVFS and run tasks

## GlusterFS: Distributed file system

Workers host GlusterFS and run tasks

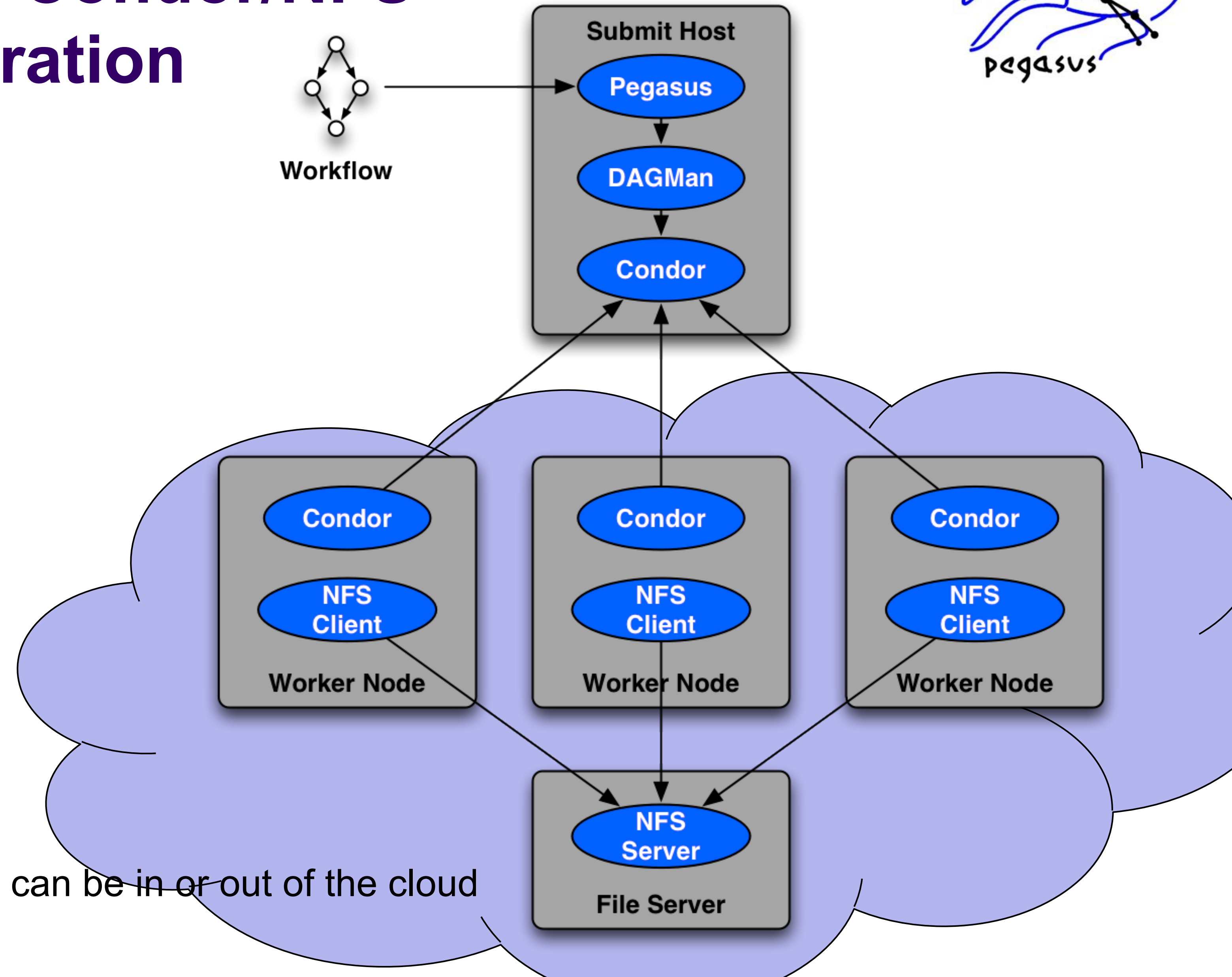
NUFA, and Distribute modes

## Amazon S3: Object-based storage system

Non-POSIX interface required changes to Pegasus

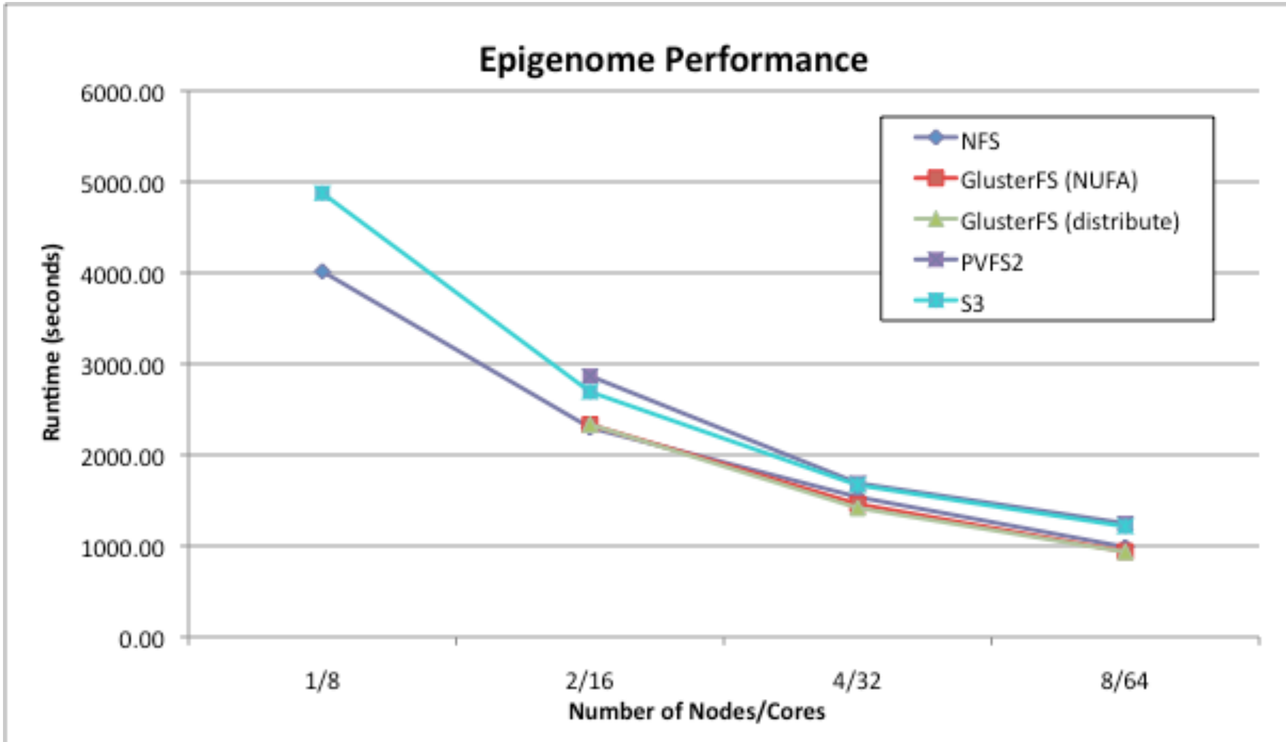
Data is cached on workers

# A cloud Condor/NFS configuration

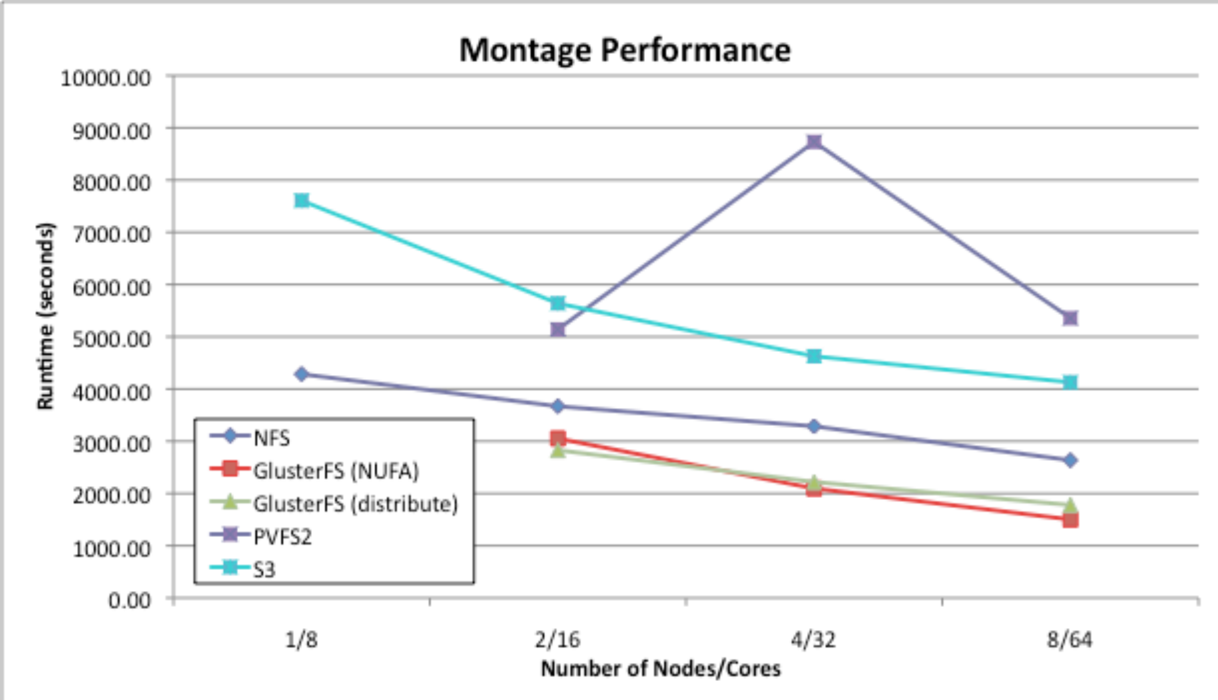



The submit host can be in or out of the cloud

# Storage System Performance

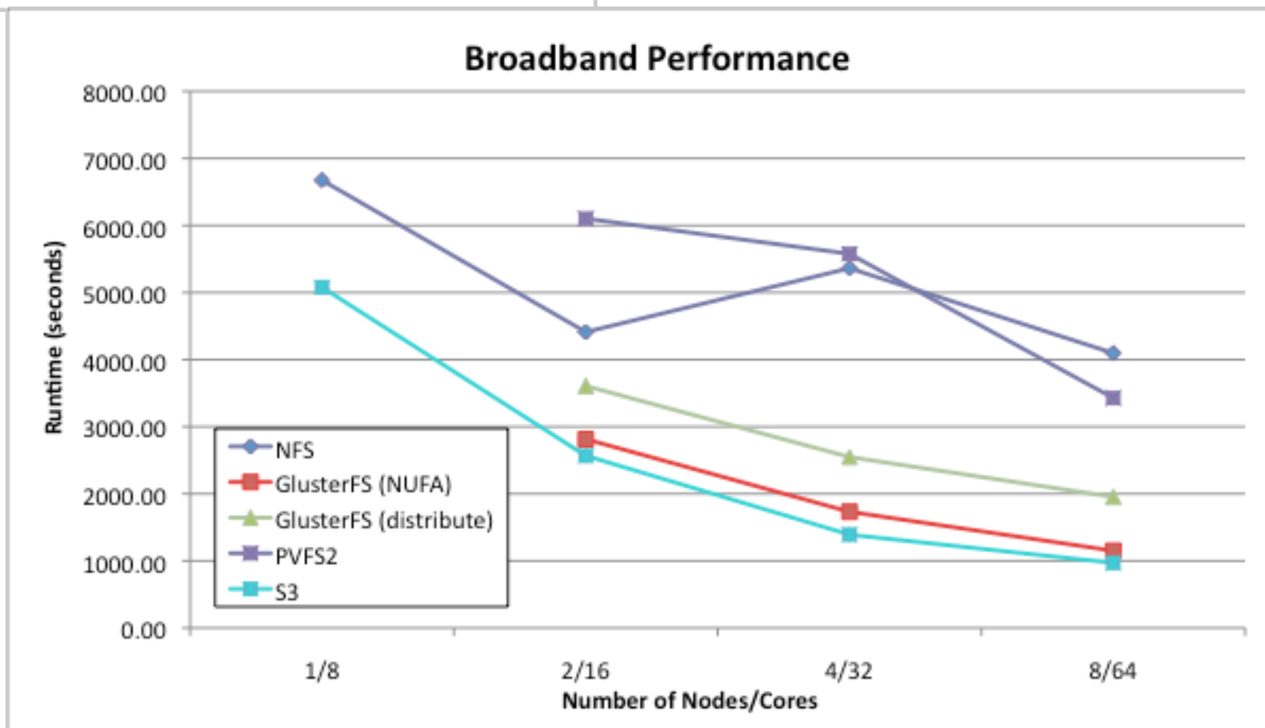



- NFS uses an extra node
- PVFS, GlusterFS use workers to store data, S3 does not
- PVFS, GlusterFS use 2 or more nodes
- We implemented whole file caching for S3




**Lots of small files**

**Re-reading the same file**



# Cost Components

## Resource Cost

- Cost for VM instances
- Billed by the hour

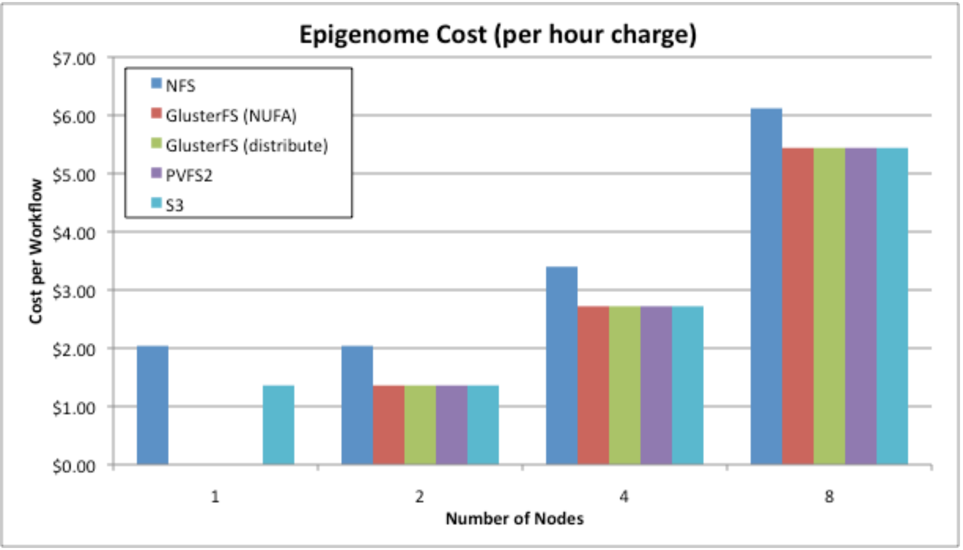
## Transfer Cost

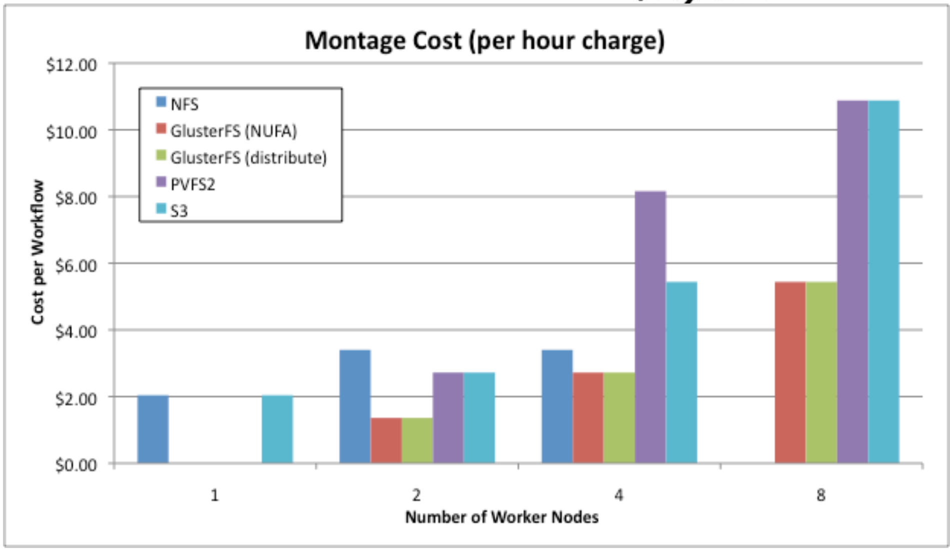
- Cost to copy data to/from cloud over network
- Billed by the GB

## Storage Cost

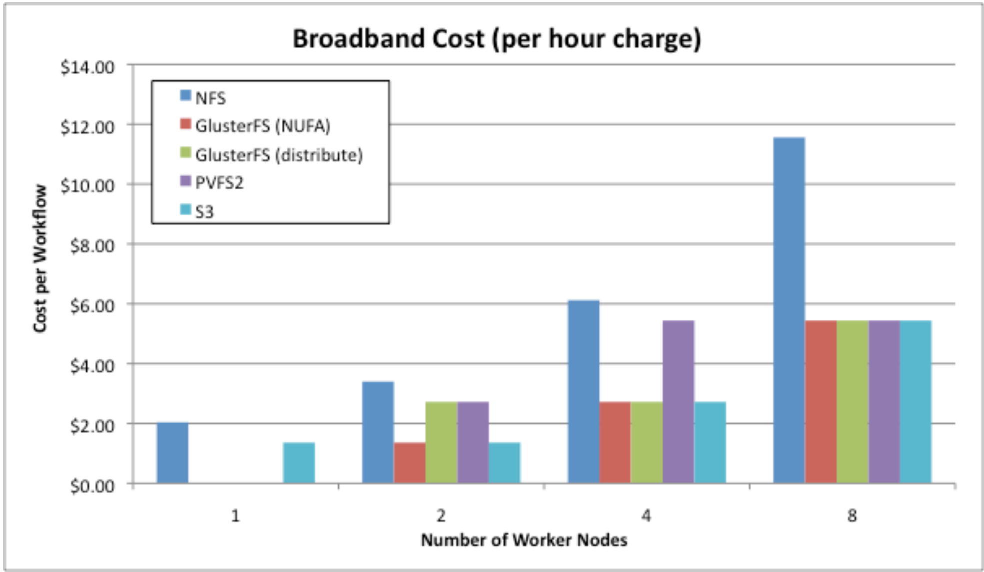
- Cost to store VM images, application data
- Billed by the GB, # of accesses

# Resource Cost (by Storage System)




- Cost tracks performance
- Price not unreasonable
- Adding resources does not usually reduce cost



# Transfer Cost

| Application | Input | Output | Logs |
|---|---|---|---|
| Montage | 4291 MB | 7970 MB | 40 MB |
| Broadband | 4109 MB | 159 MB | 5.5 MB |
| Epigenome | 1843 MB | 299 MB | 3.3 MB |

Transfer Sizes

| Application | Input | Output | Logs | Total |
|---|---|---|---|---|
| Montage | \$0.42 | \$1.32 | < \$0.01 | \$1.75 |
| Broadband | \$0.40 | \$0.03 | < \$0.01 | \$0.43 |
| Epigenome | \$0.18 | \$0.05 | < \$0.01 | \$0.23 |

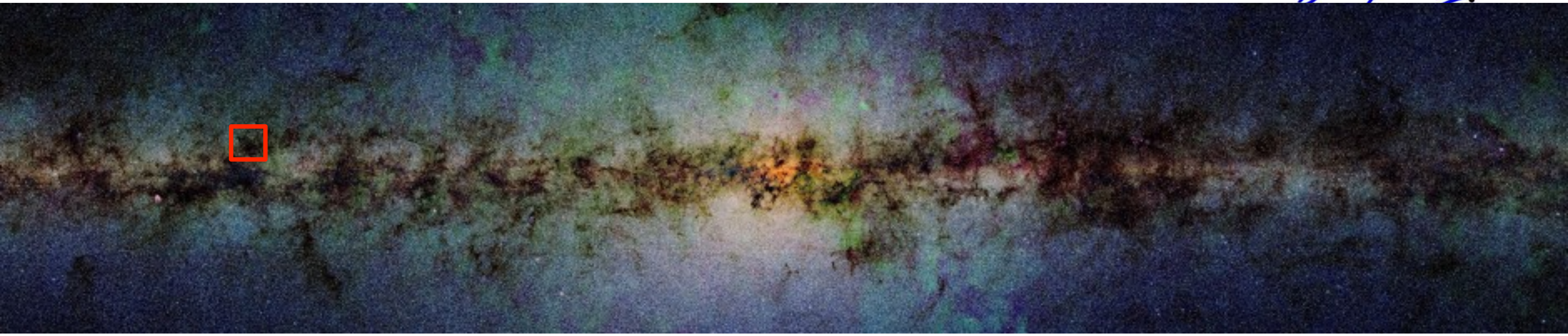
Transfer Costs

- Cost of transferring data to/from cloud
  - Input: \$0.10/GB
  - Output: \$0.17/GB
- Transfer costs are a relatively large
  - For Montage, transferring data costs more than computing it (\$1.75 > \$1.42)
- Costs can be reduced by storing input data in the cloud and using it for multiple workflows

# Outline

- Role of hosted environments
- Workflows on the Cloud
  - Challenges in running workflows on the cloud
  - Data management aspects
- Hosted Science
  - Managing workflow ensembles on the cloud
  - Within user-defined constraints
- Conclusions

# Large-Scale, Data-Intensive Workflows



Montage Galactic Plane Workflow

- 18 million input images (~2.5 TB)
- 900 output images (2.5 GB each, 2.4 TB total)
- 10.5 million tasks (34,000 CPU hours)

× 17

An analysis is composed of a number of related workflows– *an ensemble*

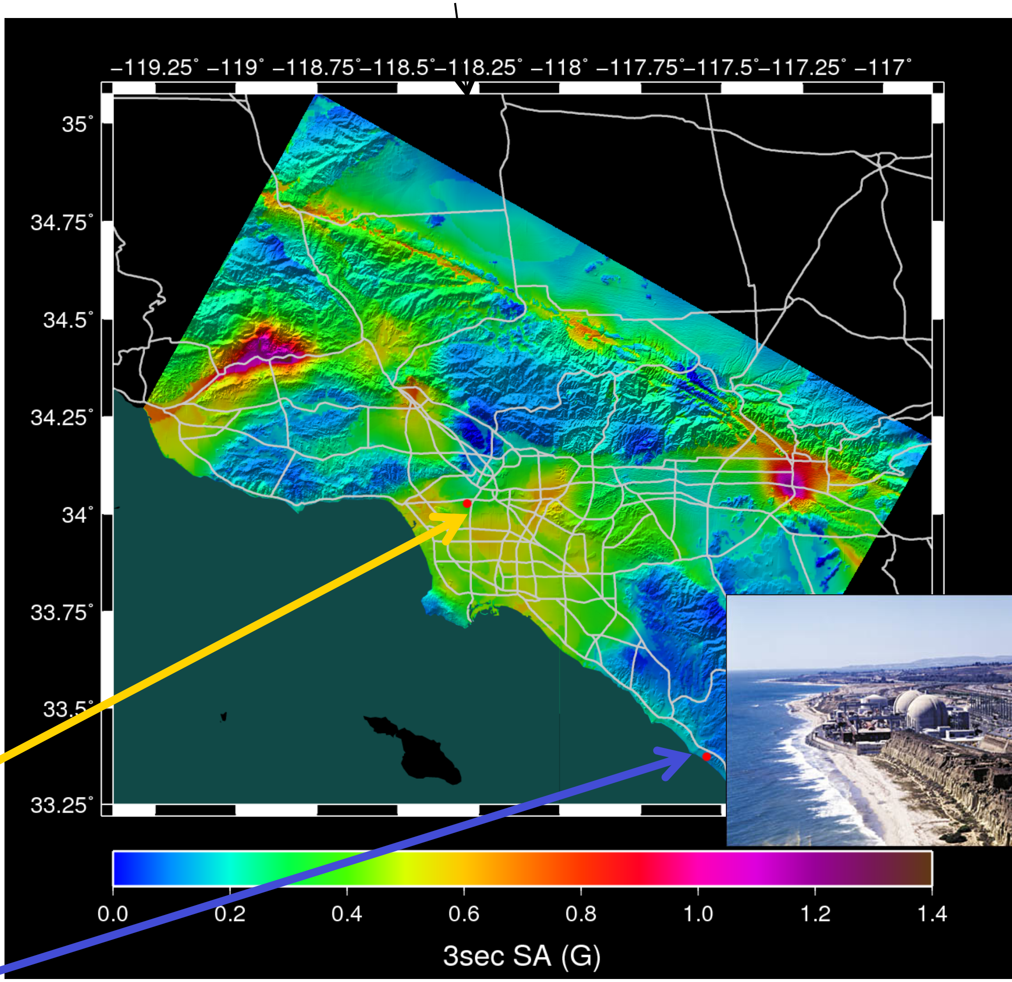
# Workflow Ensembles

## Set of workflows

- Workflows have different parameters, inputs, etc.

## Prioritized

- Priority represents user's utility




USC

San Onofre Nuclear Power Plant

# Problem Description

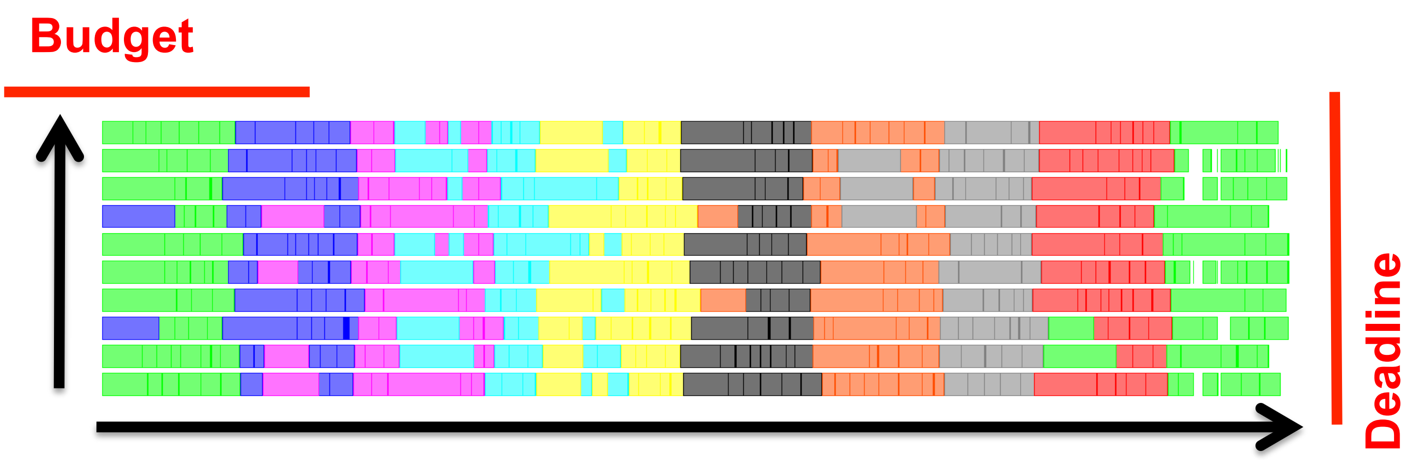
How do you manage ensembles in hosted environments ?

Typical research question:

*How much computation can we complete given the limited time and budget of our research project?*

Constraints: **Budget** and **Deadline**

**Goal**: given budget and deadline, maximize the number of prioritized workflows in an ensemble



# Explore provisioning and task scheduling decisions

## Inputs:

Budget, deadline, prioritized ensemble, and task runtime estimates

## Outputs:

**Provisioning**: Determines # of VMs to use over time

**Scheduling**: Maps tasks to VMs

## Algorithms:

**SPSS**: Static Provisioning, Static Scheduling

**DPDS**: Dynamic Provisioning, Dynamic Scheduling

**WA-DPDS**: Workflow-Aware DPDS

# SPSS

- Plans out all provisioning and scheduling decisions ahead of execution (offline algorithm)
- Algorithm:
  - For each workflow in priority order
  - Assign sub-deadlines to each task
  - Find a minimum cost schedule for the workflow such that each task finishes by its deadline
  - If the schedule cost <= the remaining budget: accept the workflow
  - Otherwise: reject the workflow
- Static plan may be disrupted at runtime

# DPDS

- Provisioning and scheduling decisions are made at runtime (online algorithm)
- Algorithm:
  - Task priority = workflow priority
  - Tasks are executed in priority order
  - Tasks are mapped to available VMs arbitrarily
  - Resource utilization determines provisioning
- May execute low-priority tasks even when the workflow they belong to will never finish
  - We assume no pre-emption of tasks

# WA-DPDS

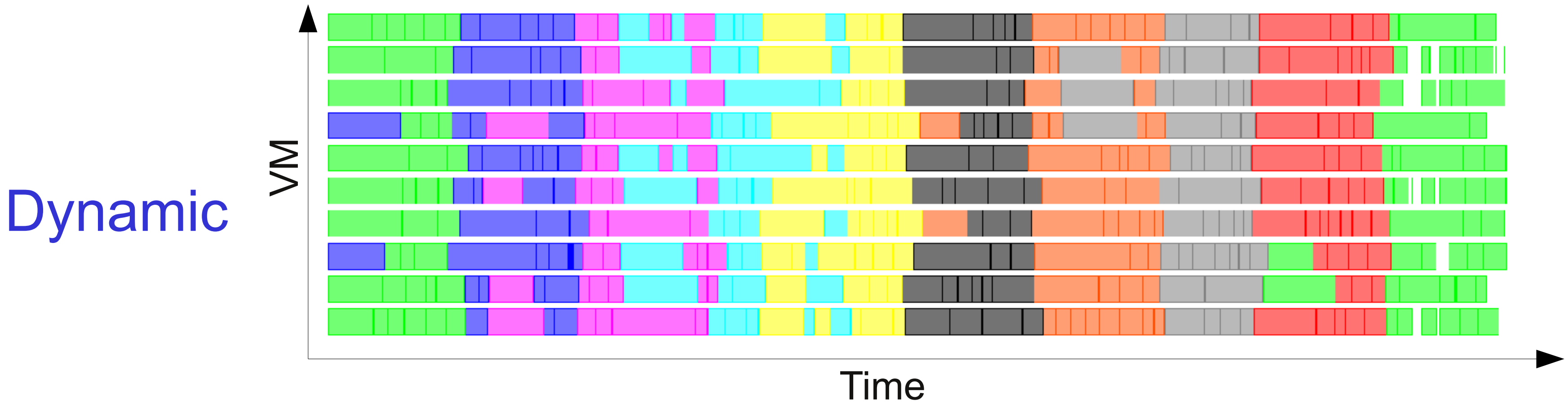
DPDS with additional workflow admission test:

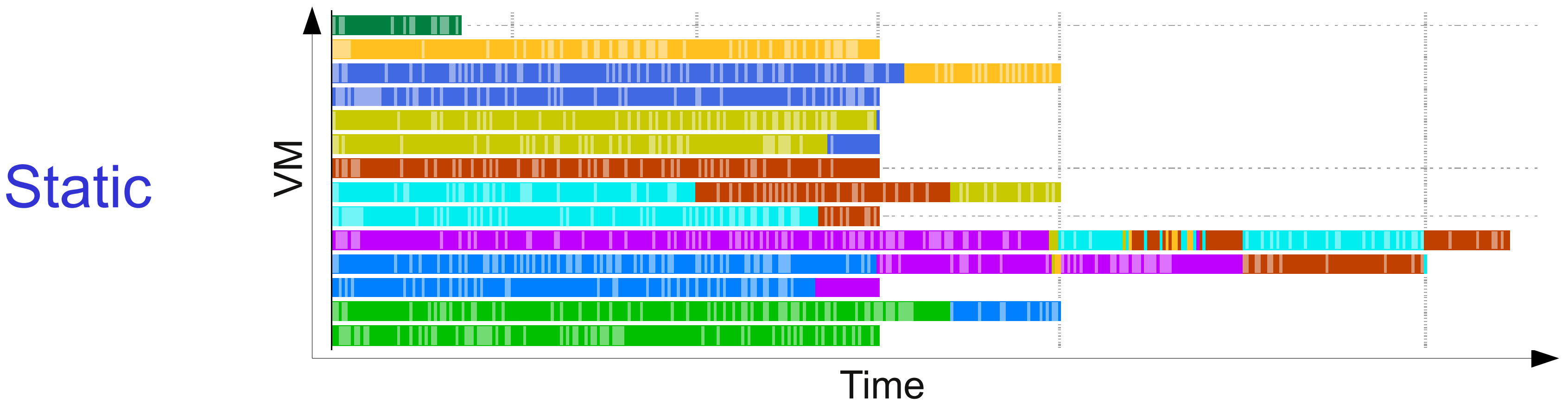
- Each time a workflow starts
- Add up the cost of all the tasks in the workflow
- Determine critical path of workflow
- If there is enough budget: accept workflow
- Otherwise: reject workflow

Other admissions tests are possible

- e.g. Critical path <= time remaining

# Dynamic vs. Static
# Task execution over time



# Evaluation

## Simulation

- Enables us to explore a large parameter space
- Simulator uses CloudSim framework

## Ensembles

- Use synthetic workflows generated using parameters from real applications
- Randomized using different distributions, priorities

## Experiments

- Determine relative performance
- Measure effect of low quality estimates and delays

# Ensemble Types

## Ensemble size

Number of workflows (50)

## Workflow size

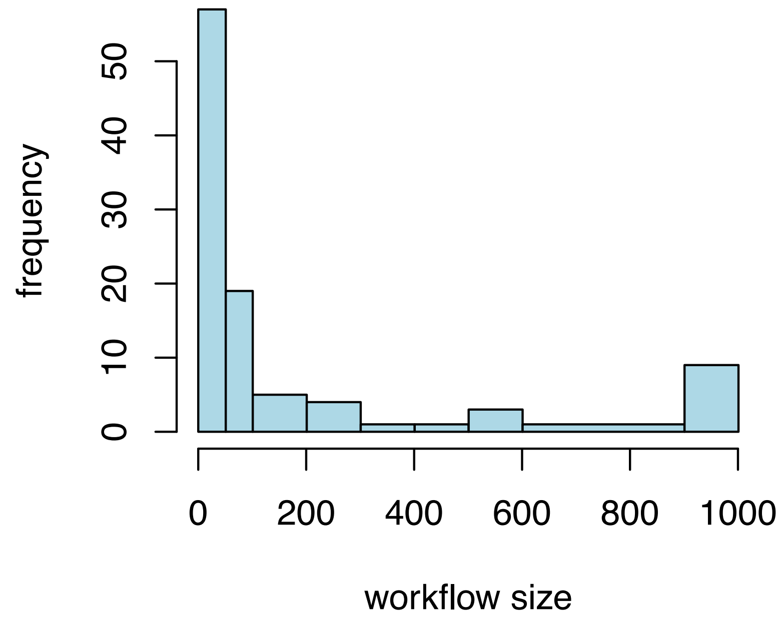
{100, 200, 300, 400,

500, 600, 700, 800, 900, and 1000}

**Constant size**

**Uniform distribution**

**Pareto distribution**




## Priorities

**Sorted**: Priority assigned by size

**Unsorted**: Priority not correlated with size

# Performance Metric

Exponential score:

$$Score(e) = \sum_{w \in Completed(e)} 2^{-Priority(w)}$$

**Key:** High-priority workflows are more valuable than all lower-priority workflows combined:

$$2^{-p} > \sum_{i = p+1, \dots} 2^{-i}$$

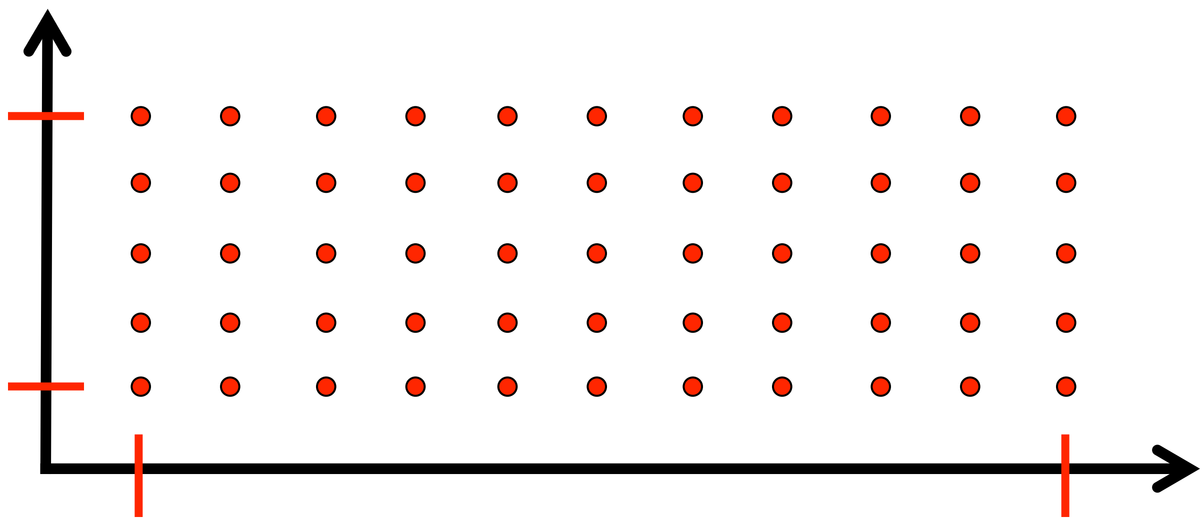
Consistent with problem definition

# Budget and Deadline Parameters

## Goal: cover space of interesting parameters

$$\left[\min_{w\,\in\,e} Cost(w), \sum_{w\,\in\,e} Cost(w)\right]$$

$$\left[\min_{w\,\in\,e} CriticalPath(w), \sum_{w\,\in\,e} CriticalPath(w)\right]$$



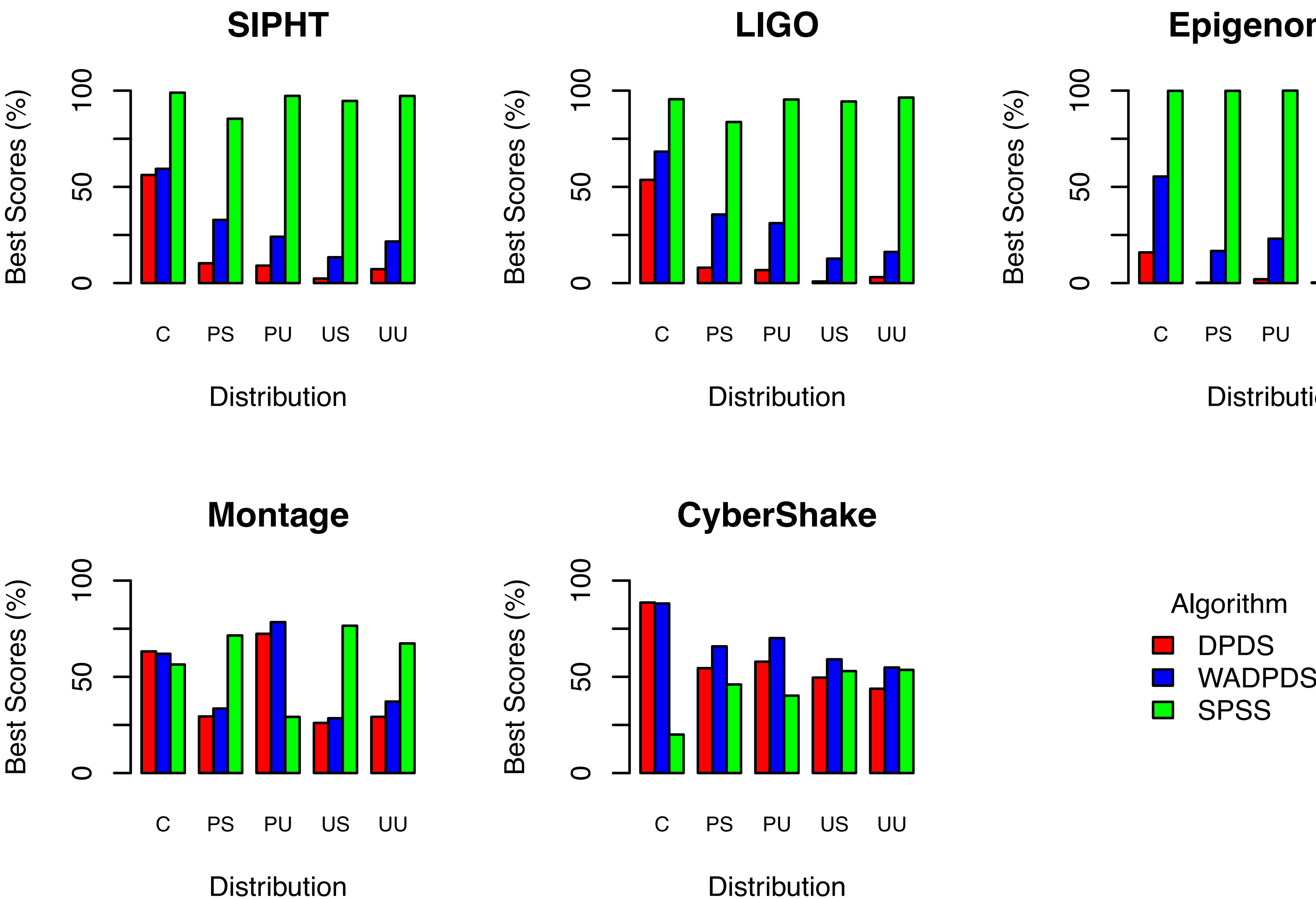
# Relative Performance

How do the algorithms perform on different applications and ensemble types?

Experiment:

- Compare relative performance of all 3 algorithms on 5 applications
- 5 applications, 5 ensemble types, 10 random seeds, 10 budgets, 10 deadlines

**Goal:** Compare % of ensembles for which each algorithm gets the highest score




C = constant, PS = Pareto sorted, PU=Pareto unsorted, US=uniform sorted, UU=uniform

# Inaccurate Runtime Estimates

What happens if the runtime estimates are inaccurate?

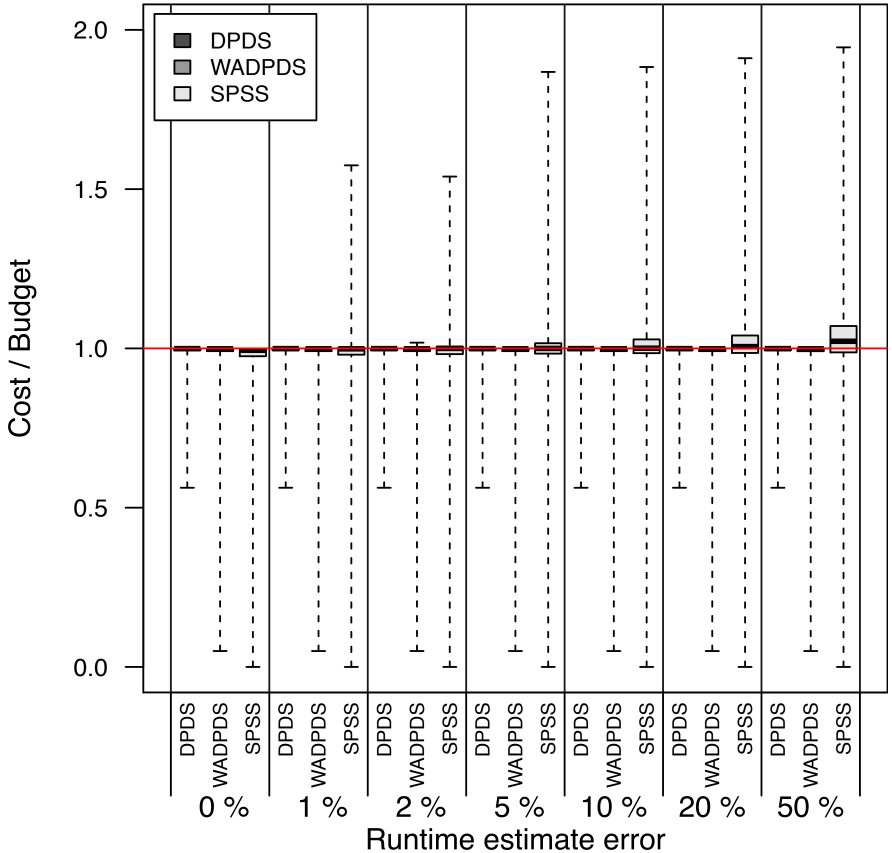
Experiment:

- Introduce uniform error of ±p% for p from 0 to 50
- Compare ratios of actual cost/budget and actual makespan/deadline
- All applications, all distributions, and 10 ensembles, budgets and deadlines each

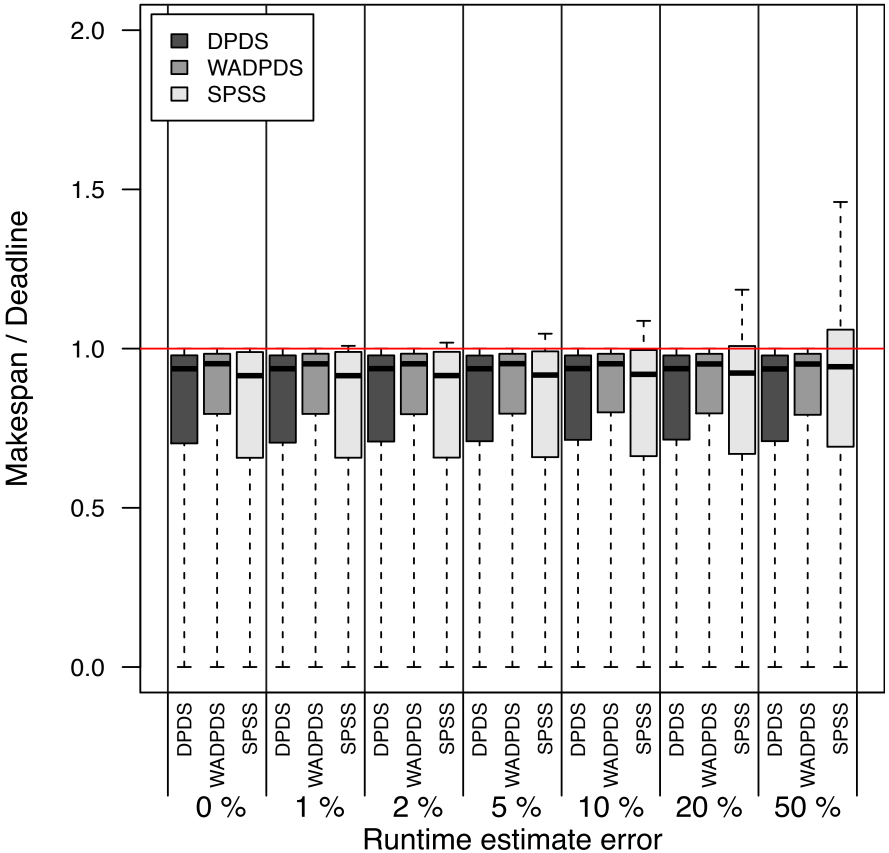
**Goal:** See how often each algorithm exceeds budget and deadline

# Inaccurate Runtime Estimate Results

Cost / Budget




Makespan / Deadline

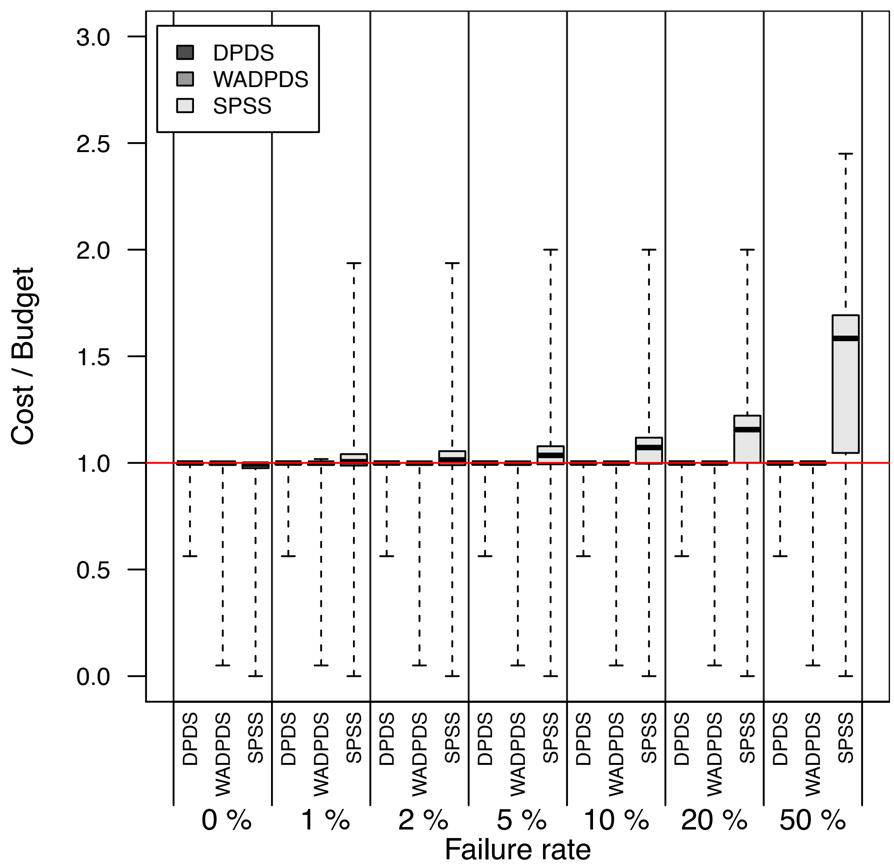


# Task Failures

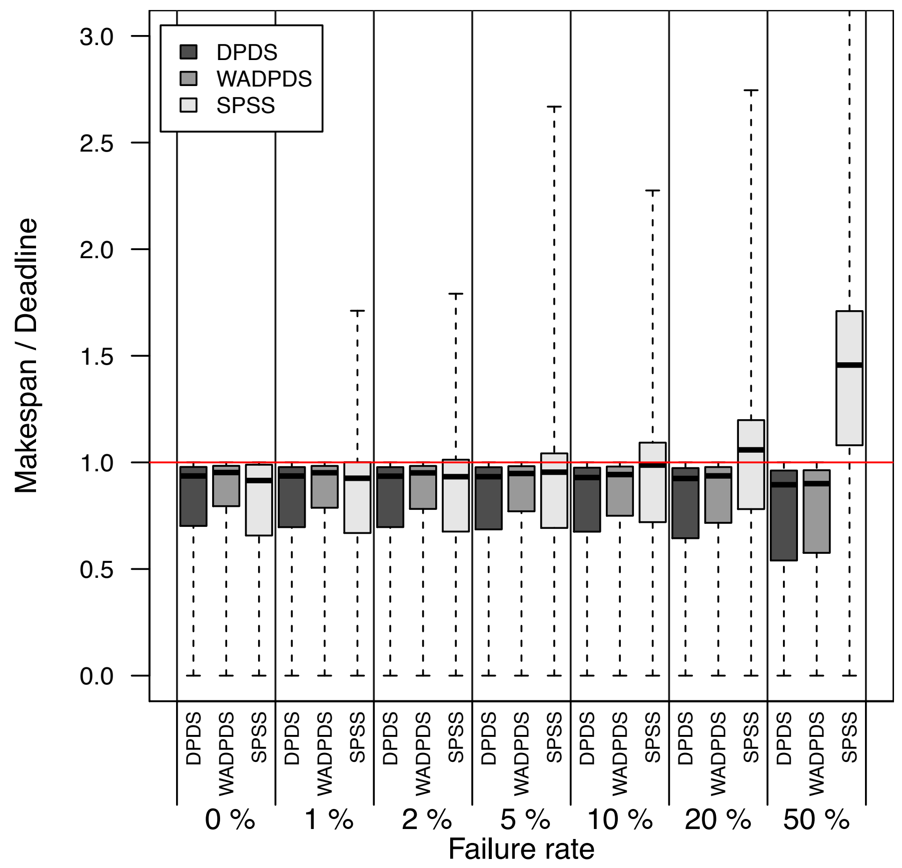
- Large workflows on distributed systems often have failures
- Experiment:
  - Introduce a uniform task failure rate between 0% and 50%
  - All applications, all distributions, and 10 ensembles, budgets and deadlines
- **Goal:** Determine if high failure rates lead to significant constraint overruns

# Task Failure Results

Cost / Budget




Makespan / Deadline



# Summary I--observations

- Commercial clouds are usually a reasonable alternative to grids for a number of workflow applications
  - Performance is good
  - Costs are OK for small workflows
  - Data transfer can be costly
  - Storage costs can become high over time
- Clouds require additional configurations to get desired performance
  - In our experiments GlusterFS did well overall
- Need tools to help evaluate costs for entire computational problems (ensembles), not just one workflows
- Need tools to help manage the costs, the applications, and the resources

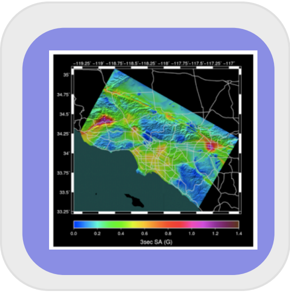
# Summary II—looking into the future

There is a move to hosting more services in the cloud

Hosting science will require

- a number of integrated services
- seamless support for managing resource usage and thus cost and performance
- ease of use---can you do science as an app?

References: **http://pegasus.isi.edu**



Paper on ensembles at SC'12 in Salt Lake City

**deelman@isi.edu**