Community Resources for Enabling Research in Distributed Scientific Workflows

Rafael Ferreira da Silva, Weiwei Chen, Gideon Juve, Karan Vahi, and Ewa Deelman

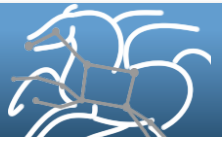
University of Southern California

Funding from NSF and DOE

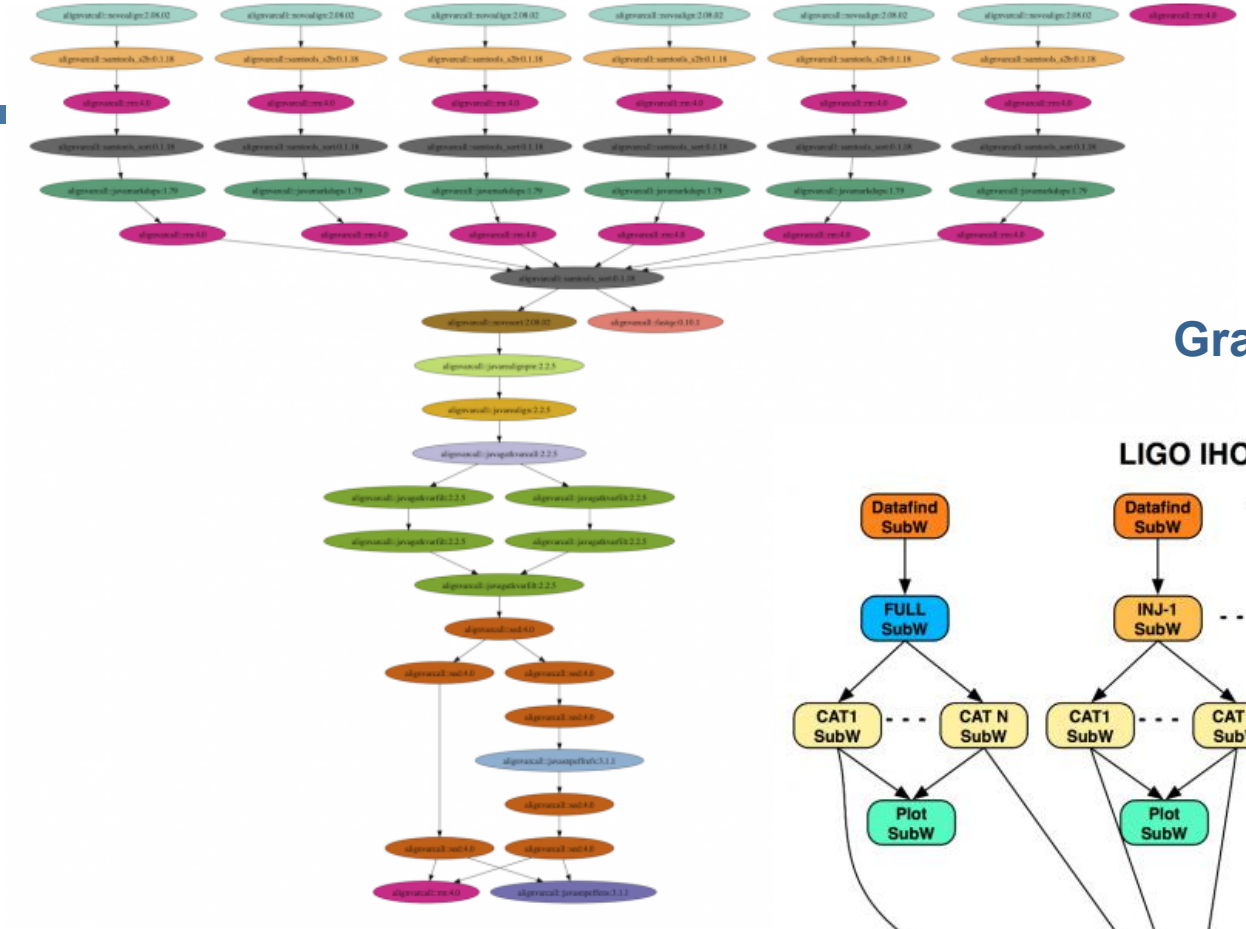
Community Resources for Enabling Scientific Workflow Research

- Execution traces
- Synthetic workflow generator
- Workflow execution simulator

- www.workflowarchive.org

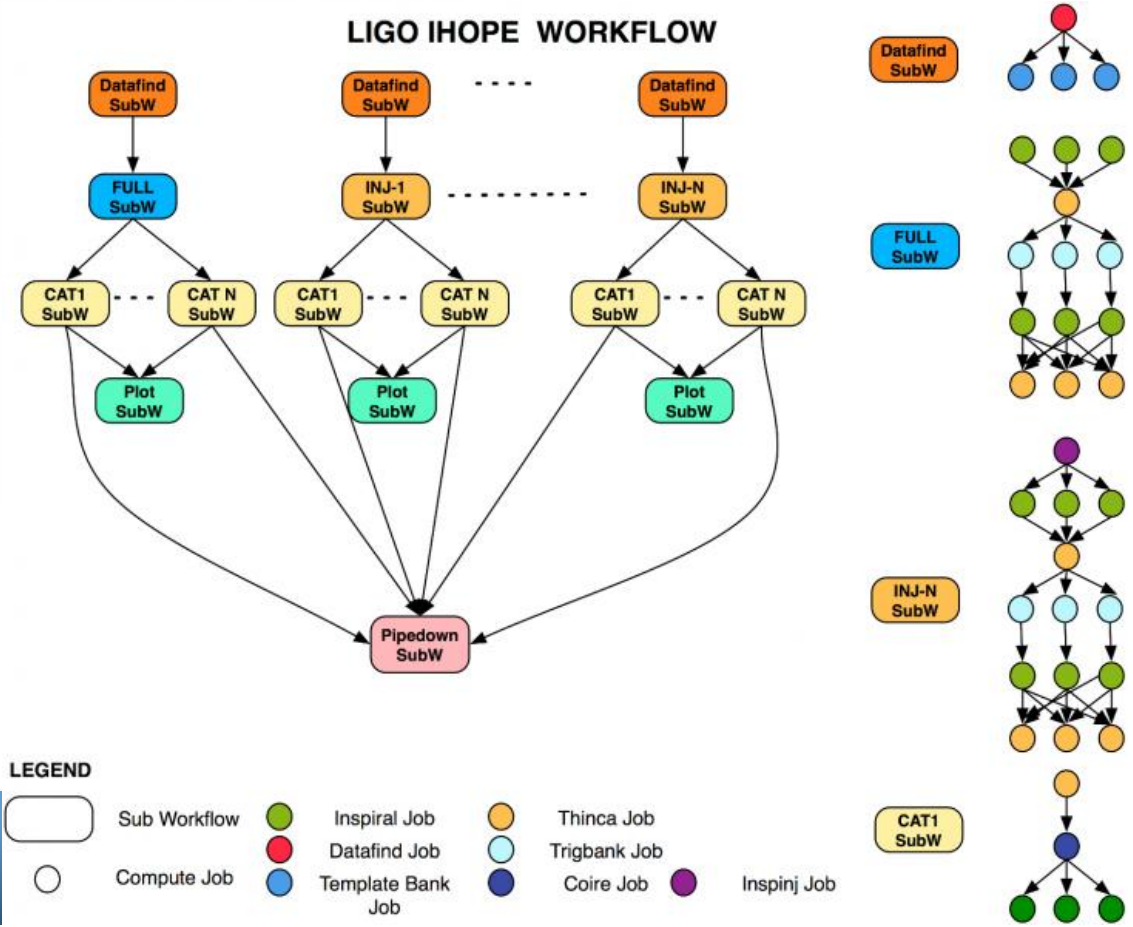


Some workflows are structurally complex

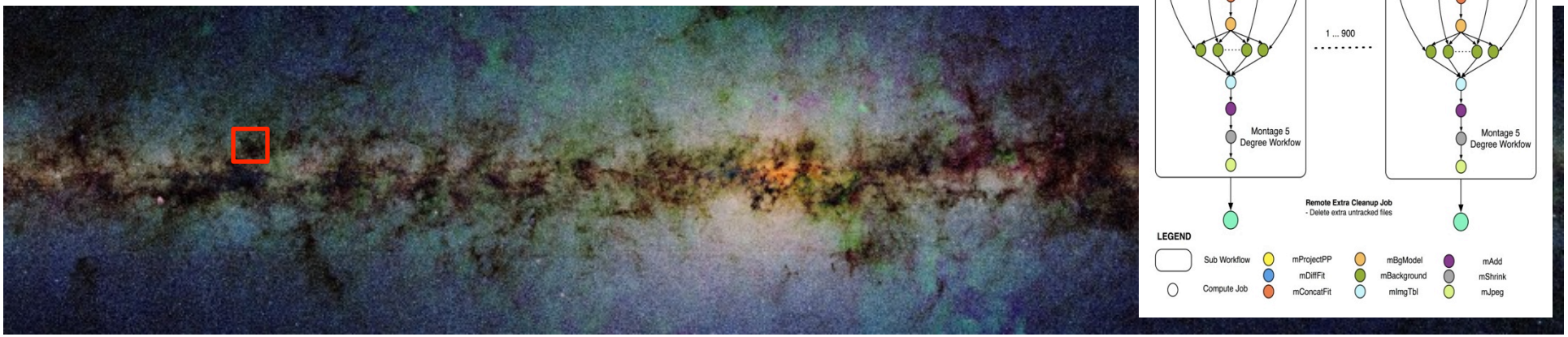


Genomics Workflow

Gravitational-wave physics



Some workflows are large-scale and data-intensive



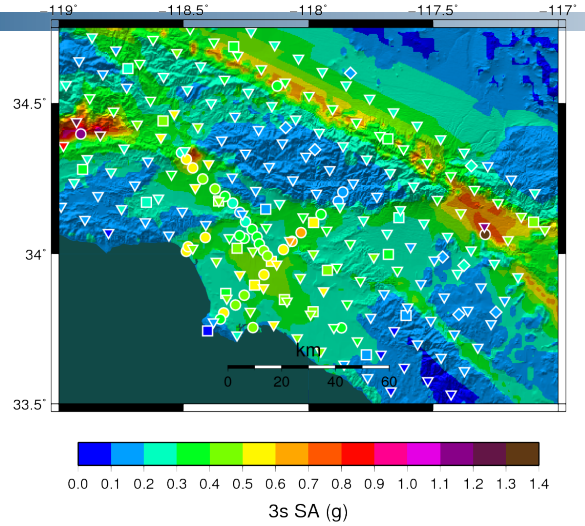
John Good (Caltech)

- **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

- **Need to support hierarchical workflows and scale, workflow ensembles**

Southern California Earthquake Center, T. Jordan, USC



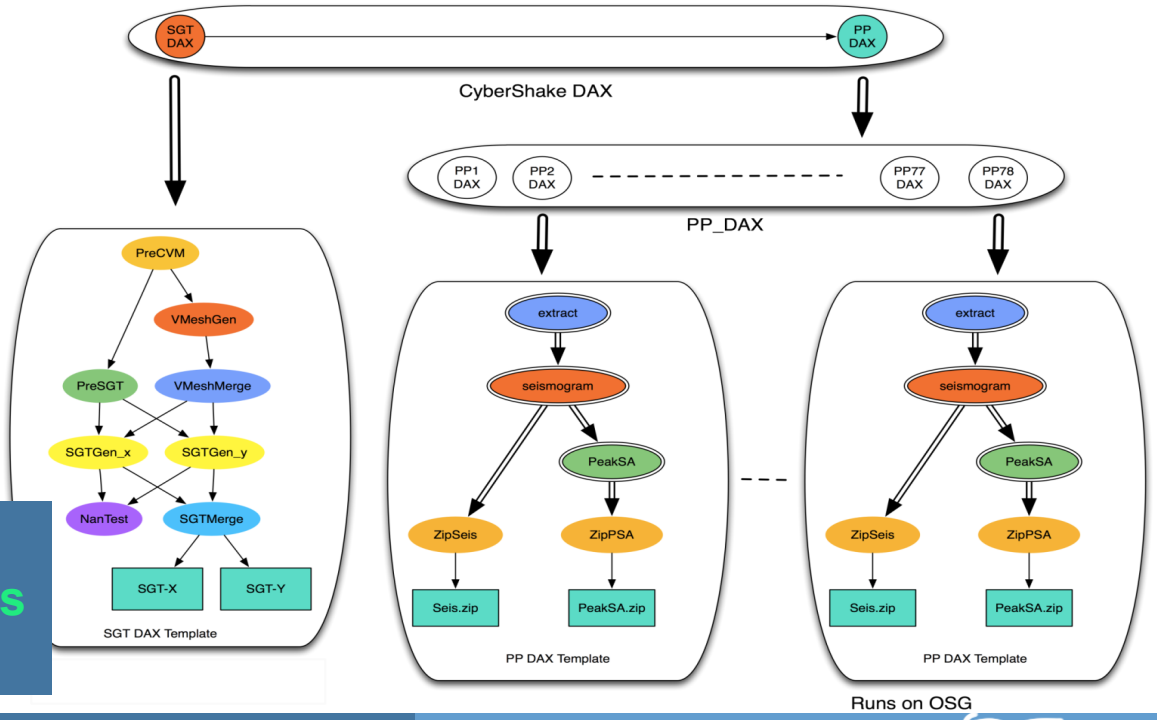
CyberShake PSHA Workflow

- **Description**
  - 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)

239 Workflows

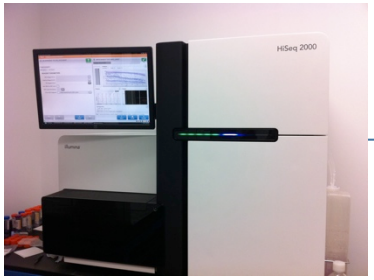
- **Each site in the input map corresponds to one workflow**
- **Each workflow has:**
  - **820,000 tasks**

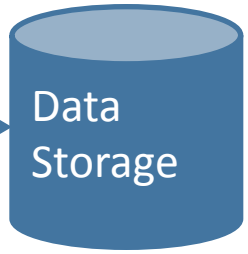
MPI codes ~ 12,000 CPU hours,
Post Processing 2,000 CPU hours
Data footprint ~ 800GB

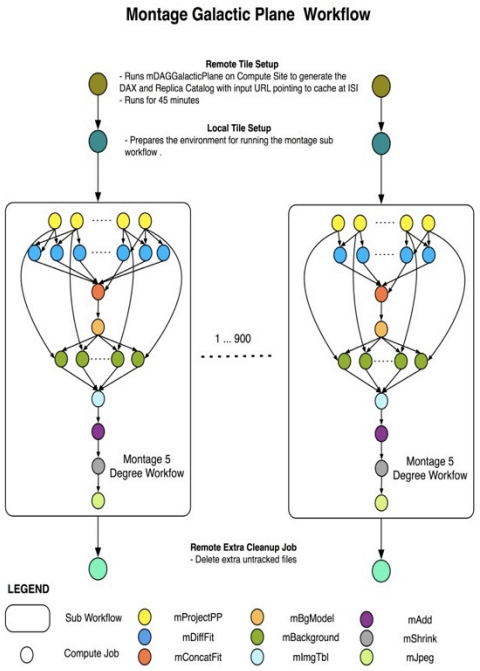


Coordination between resources is needed

Sometimes the environment is complex







Work definition

Local Resource

Campus Cluster

XSEDE

NERSC

ALCF

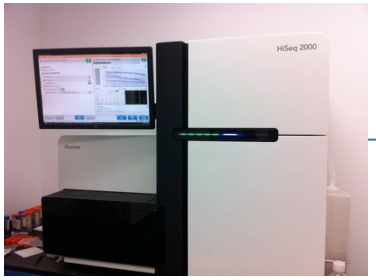
OLCF

Open Science Grid

FutureGrid

Amazon Cloud

Sometime you want to change or combine resources

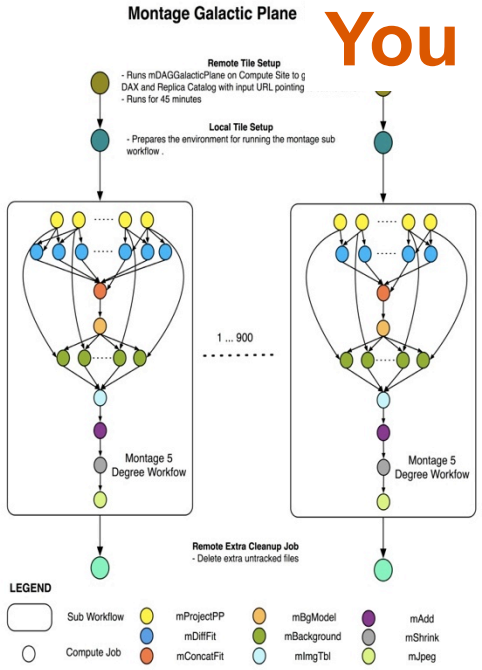


Data Storage

data

Campus Cluster

XSEDE



You don't want to recode your workflow

Local Resource

work

ALCF

OLCF

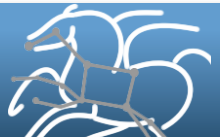
Open Science Grid

FutureGrid

Amazon Cloud

Pegasus Workflow Management System

- A workflow "compiler"
  - Input: abstract workflow description, resource-independent
  - Output: executable workflow with concrete resources
  - Transforms the workflow for performance and reliability (task clustering, data cleanup, etc.)
  - Automatically locates physical locations for both workflow tasks and data
- A workflow engine (DAGMan)
  - Executes the workflow on local or distributed resources (HPC, clouds)
  - Task executables are managed by Condor *schedd*
- Provenance and execution traces are collected and stored
- Traces and DB can be mined for performance and overhead information

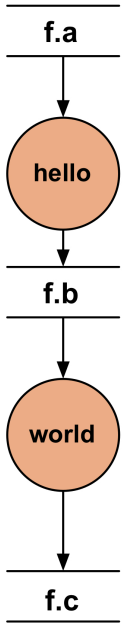


Generating executable workflows

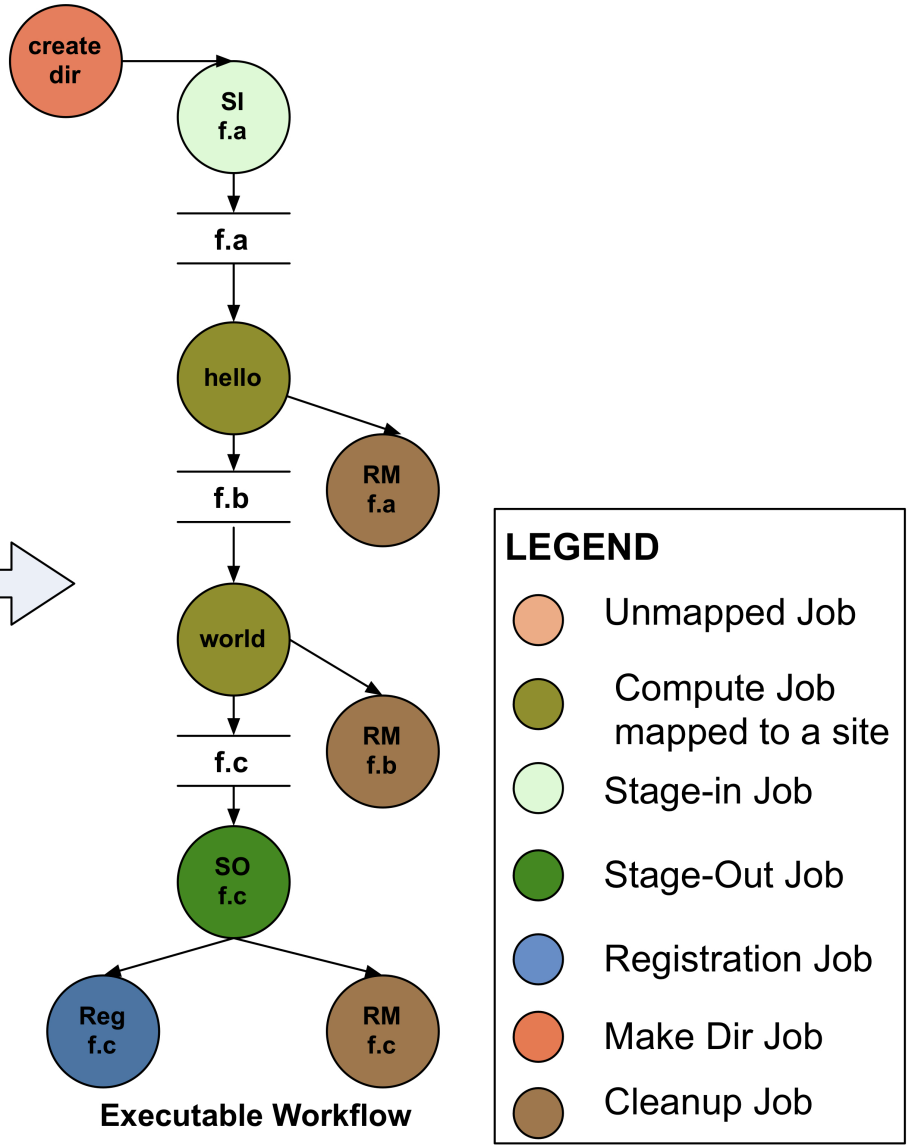
(DAX)

APIs for workflow specification (DAX--- DAG in XML)

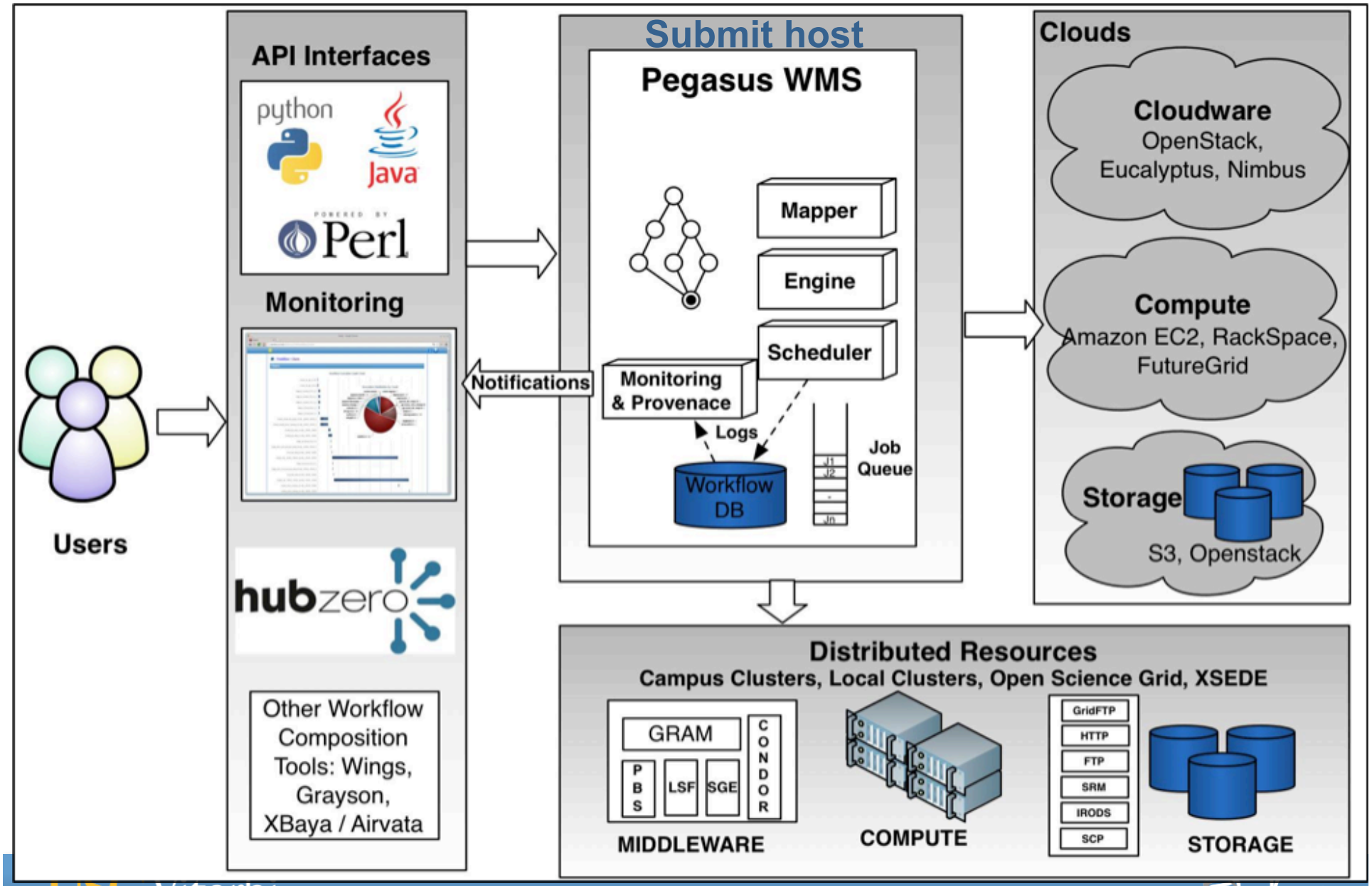
Java, Perl, Python



Abstract Workflow

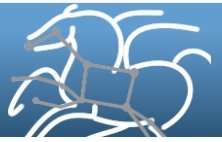


Executable Workflow


API Interfaces
python
Java
Perl
Monitoring
Users
hubzero
Other Workflow Composition Tools: Wings, Grayson, XBaya / Airvata
Notifications
Submit host
Pegasus WMS
Mapper
Engine
Scheduler
Monitoring & Provenance
Logs
Workflow DB
Job Queue
J1
J2
Jn
Clouds
Cloudware
OpenStack, Eucalyptus, Nimbus
Compute
Amazon EC2, RackSpace, FutureGrid
Storage
S3, Openstack
Distributed Resources
Campus Clusters, Local Clusters, Open Science Grid, XSEDE
GRAM
PBS
LSF
SGE
CONDOR
MIDDLEWARE
COMPUTE
GridFTP
HTTP
FTP
SRM
IRODS
SCP
STORAGE

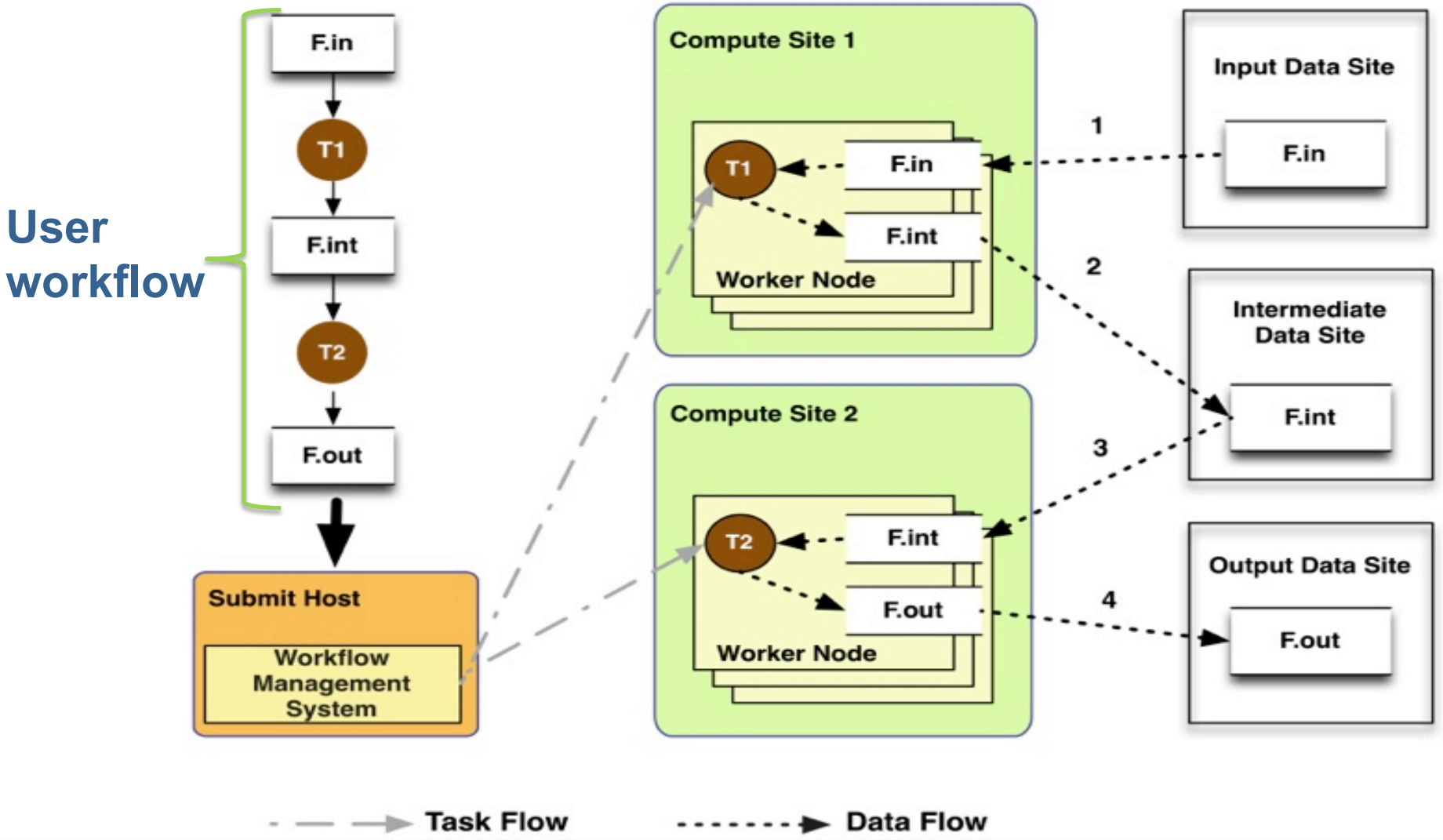
Pegasus optimizations address issues of:

- Failures in the execution environment or application
- Data storage limitations on execution sites
- Performance
  - Small workflow tasks
- Heterogeneous execution architectures
  - Different file systems (shared/non-shared)
  - Different system architectures (Cray XT, Blue Gene, …)



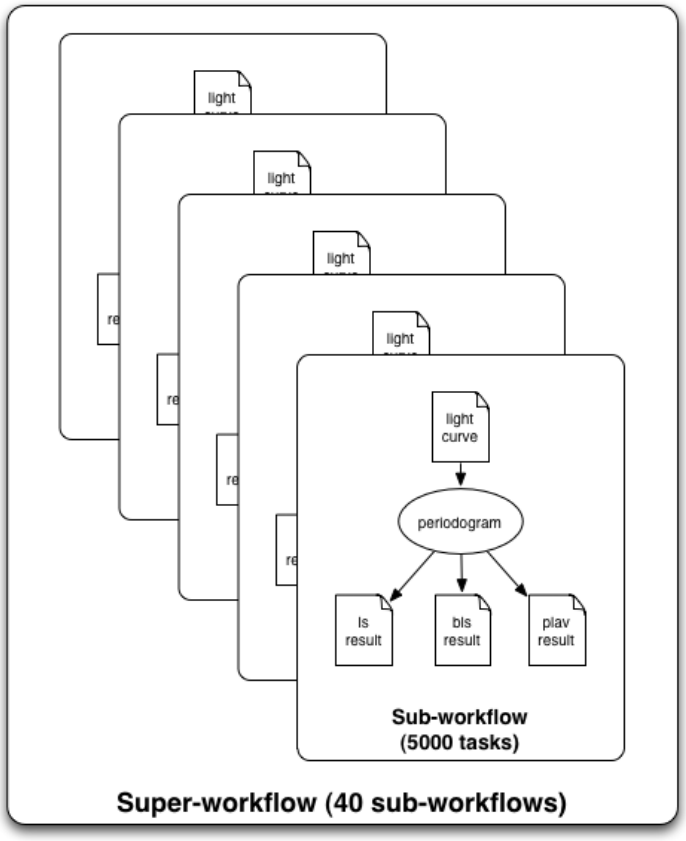
Storage limitations

Variety of file system deployments: shared vs non-shared



Sometimes the environment is just not exactly right

Single core workload

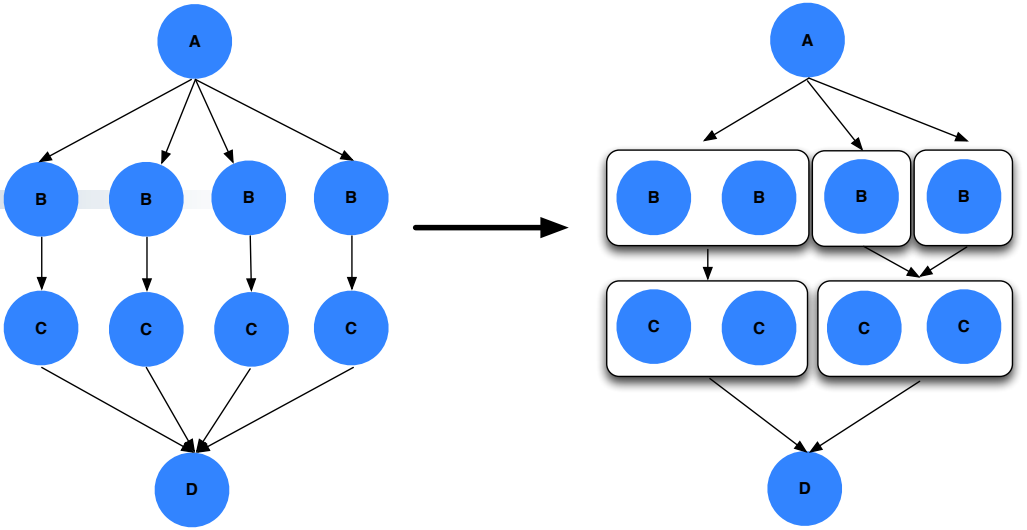


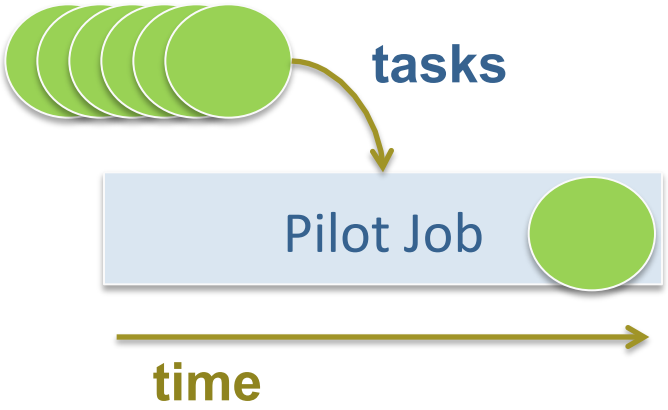
Cray XT System Environment / ALPS / aprun

- Designed for MPI codes

Solutions

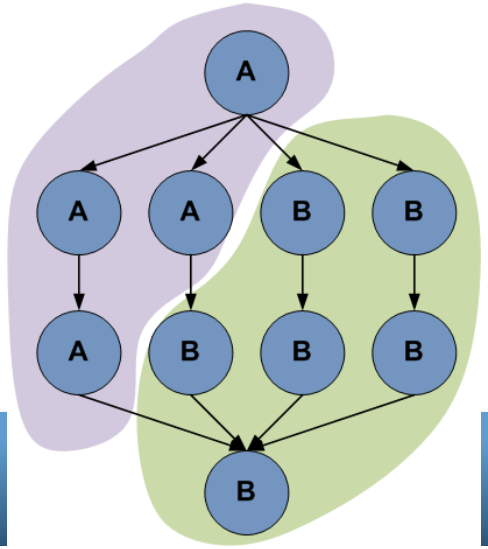
Cluster tasks





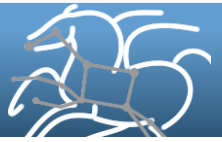
Use "pilot" jobs to dynamically provision a number of resources at a time

Develop an MPI-based workflow management engine to manage sub-workflows



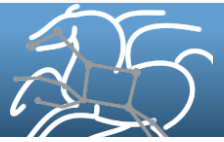
Pegasus-kickstart

- Lightweight C based executable to launch jobs
- Captures job runtime provenance and logs it as a XML record
- Following information is captured about each job on all supported platforms
  - exit code with which the job it launched exited
  - start time and duration of the job
  - hostname and IP address of the host the job ran on
  - stdout and stderr of the job
  - arguments with which it launched the job
  - directory in which the job was launched
  - environment that was set for the job before it was launched

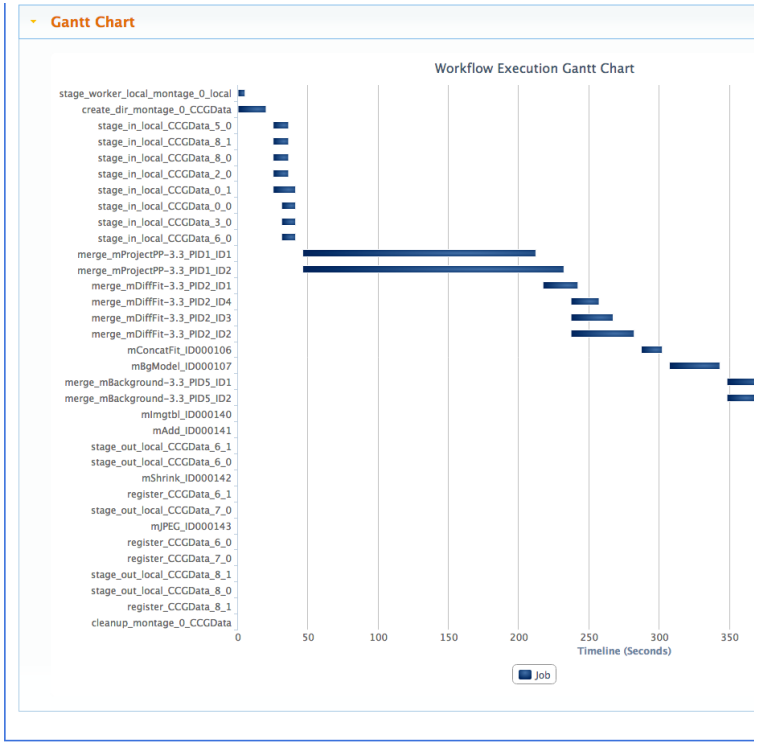


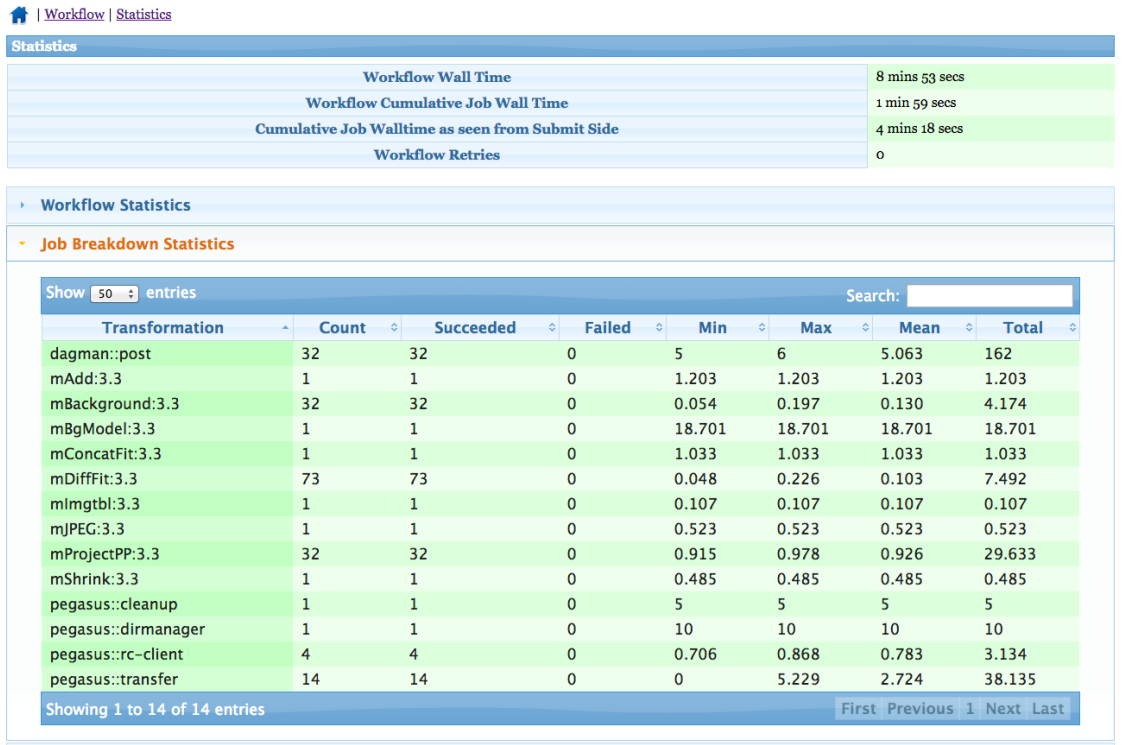
Pegasus-kickstart with extra tracing enabled

- Captures extra tracing information (optional) on Linux based hosts
- Collects, for each process in process tree
  - peak memory usage (resident set size, and vm size)
  - total I/O read and write,
  - runtime,
  - start and end time
  - Pid
  - all files accessed (total read and write per file)
- Traces also include DAGMan and Condor logs (release of jobs to the scheduling system, sending jobs to remote resources, etc..)

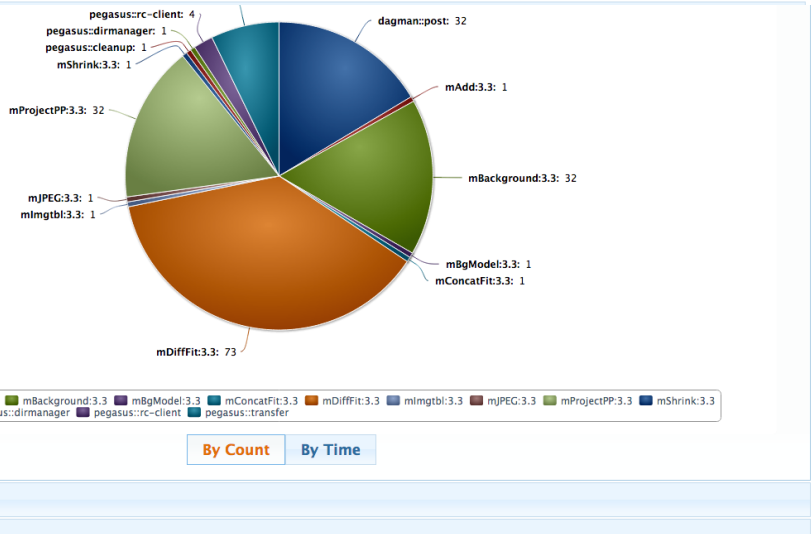


Workflow Monitoring Dashboard – *pegasus-dashboard*





Status, statistics, timeline of jobs

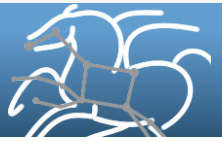


USC Viterbi School of Engineering

Overview of the Community Resources

www.workflowarchive.org

- **Execution Traces** of a range of real workflow applications
- **Synthetic Workflow Generator** produces realistic workflows based on profiles extracted from execution traces (astronomy, gravitational-wave physics, bioinformatics, earthquake science)
- **Workflow Simulator** mimics the execution of synthetic workflows on realistic infrastructures



Workflow Traces Archive

- **Workflow Gallery currently has 11 workflow applications, most with multiple runs**

08/24 12:24:55 submitting: condor_submit -a dag_node_name' '=' 'stage_in_remote_usc_1 -a +DAGManJobId' '=' '11443 -a DAGManJobId' '=' '11443 -a submit_event_notes' '=' 'DAG' 'Node:' 'stage_in_remote_usc_1 -a +DAGParentNodeNames' '=' '"create_dir_61HE2AAXX_2_HSB_135_S1C_R_0_usc" stage_in_remote_usc_1.sub

08/24 12:24:55 From submit: Submitting job(s).

08/24 12:24:55 From submit: Logging submit event(s).

08/24 12:24:55 From submit: 1 job(s

08/24 12:24:55 assigned Condor ID

08/24 12:24:55 Submitting Condor N

08/24 12:24:55 submitting: condor_s +DAGManJobId' '=' '11443 -a DAGM 'stage_in_remote_usc_2 -a +DAGPa '"create_dir_61HE2AAXX_2_HSB_1

08/24 12:24:55 From submit: Submit

08/24 12:24:55 From submit: Loggin

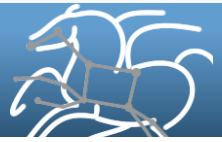
08/24 12:24:55 From submit: 1 job(s

08/24 12:24:55 assigned Condor ID

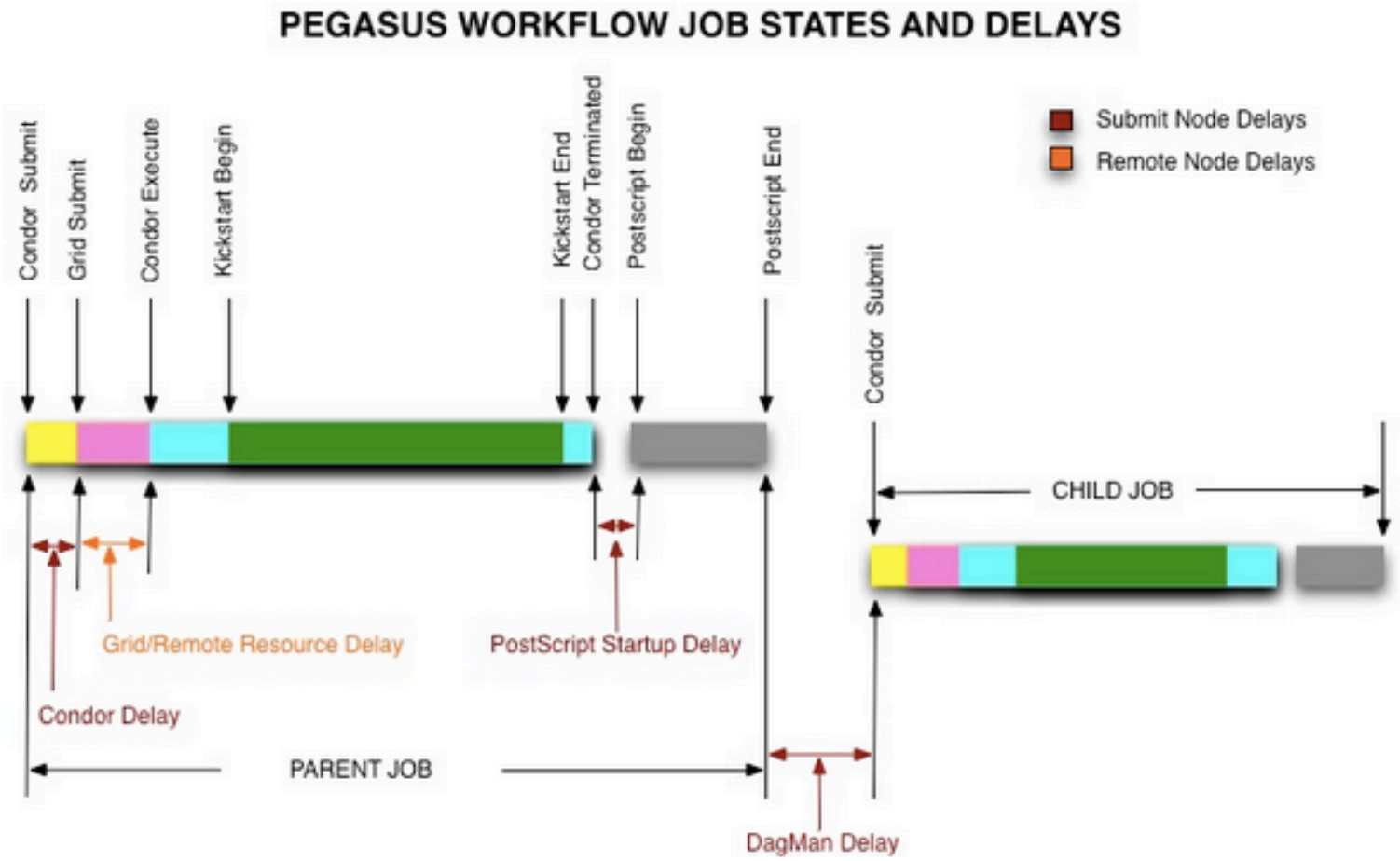
08/24 12:24:55 Submitting Condor N

DAGMan, Condor, Pegasus logs, submit files, stdout, stderr

1314221995 SamToBam_SamToBam-27-1 EXECUTE 11601.0 usc 7200 158
1314221995 SamToMrf_SamToMrf-0-1 EXECUTE 11602.0 usc 7200 159
1314221995 SamToBam_SamToBam-26-1 EXECUTE 11600.0 usc 7200 157
1314222155 SamToBam_SamToBam-27-1 JOB_TERMINATED 11601.0 usc
1314222155 SamToBam_SamToBam-27-1 JOB_SUCCESS 0 usc 7200 158
1314222155 SamToBam_SamToBam-27-1 POST_SCRIPT_STARTED 11601
1314222160 SamToBam_SamToBam-27-1 POST_SCRIPT_TERMINATED 1
1314222160 SamToBam_SamToBam-27-1 POST_SCRIPT_SUCCESS 0 usc
1314222166 stage_out_remote_usc_5_1 SUBMIT 11605.0 usc - 162
1314222176 stage_out_remote_usc_5_1 GLOBUS_SUBMIT 11605.0 usc - 1
1314222176 stage_out_remote_usc_5_1 GRID_SUBMIT 11605.0 usc - 162
1314222176 stage_out_remote_usc_5_1 EXECUTE 11605.0 usc - 162



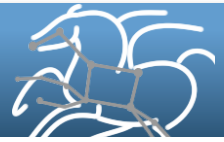
Traces include information about system overheads



Tools to calculate job statistics

Task Type	Count	Runtime(s)	IO Read (MB)	IO Write (MB)	Memory Peak(MB)	CPU Utilization(%)
mProjectPP	2102	1.73	2.05	8.09	11.81	86.96
mDiffFit	6172	0.66	16.56	0.64	5.76	28.39
mConcatFit	1	143.26	1.95	1.22	8.13	53.17
mBgModel	1	384.49	1.56	0.10	13.64	99.89
mBackground	2102	1.72	8.36	8.09	16.19	8.46
mImgtbl	17	2.78	1.55	0.12	8.06	3.48
mAdd	17	282.37	1102	775.45	16.04	8.48
mShrink	16	66.10	412	0.49	4.62	2.30
mJPEG	1	0.64	25.33	0.39	3.96	77.14

Table 1. Execution profile of the Montage workflow, averages calculated



Automatic Workflow Characterization

- Characterize tasks based on their estimation capability
  - **Runtime, I/O write, memory peak ➔ estimated from I/O read**

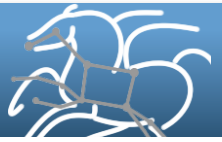
- Use correlation statistics to identify statistical relationships between parameters
  - High correlation values yield accurate estimations, **Estimation based on the ratio: parameter/input data size**
  - Low correlation: use mean value

Task	Runtime		I/O Write		Memory Peak	
	ρ	σ	ρ	σ	ρ	σ
fastqSplit	0.98	9.00	1.00	297.15	0.00	0.01
filterContams	-0.03	0.27	0.99	1.46	0.00	0.01
sol2sanger	0.21	0.41	0.90	1.49	0.00	0.01
fast2bfq	0.18	0.27	0.56	0.87	0.00	0.01
map	0.02	18.96	0.06	0.70	0.01	1.43
mapMerge	0.98	13.33	0.99	189.81	-0.36	2.15
pileup	0.99	4.73	0.17	249.78	0.87	25.70

Constant values

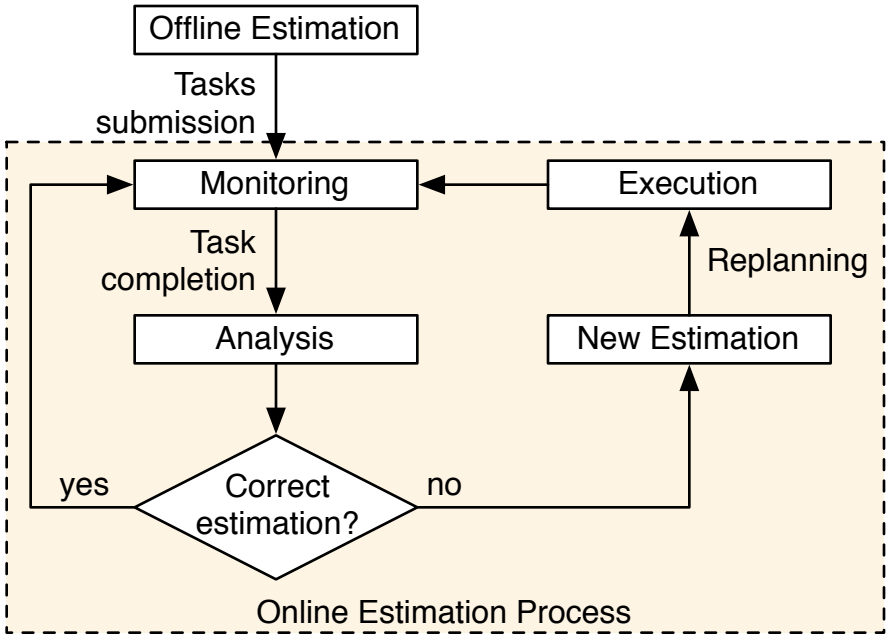
Correlated if $\rho > 0.8$

Epigenomics workflow



Workflow Resource Usage Prediction

- **Profile and Characterize Workflow Execution**
  - **I/O, runtime, memory usage, and CPU utilization**
  - **Predict task runtime, e.g. using correlations based techniques**
  - **Predict resource usage (disk space, memory consumption) using an online estimation process**



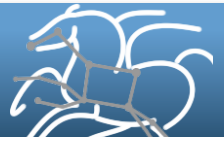
DOE project: dV/dt Accelerating the Rate of Progress towards Extreme Scale Collaborative Science

Predictive Modeling and Diagnostic Monitoring of Extreme Science Workflows

Objective: Understand complex scientific workflow applications and infrastructure behaviors and to translate this understanding into flexible, end-to-end analytical models that can effectively predict the behavior of extreme scale workflows on current and future infrastructures

Approach:

- Engage DOE science teams from simulation (e.g., Earth Systems Modeling (ESM) and instrument facilities (e.g., Spallation Neutron Source(SNS) to create example workflow scenarios
- Develop a general analytical modeling methodology that captures the end-to-end performance of these workflow scenarios using a structured modeling approach;
- Validate the analytical models using empirical measurement and simulation
- Employ the analytical performance models to facilitate prototype capabilities that include anomaly detection and diagnosis, resource management and adaptation, and infrastructure design and planning.



Workflow Generator Toolkit

- Inputs: the size of the workflow in number of jobs, or the input data size, or a scaling factor
- Inputs are combined with probability distributions of file sizes and runtimes from real executions of the workflow, to generate random values that are used in constructing the synthetic workflow.
- Generators have application-specific code and parameters that are designed to reproduce the structure and characteristics of the application.
- **Currently supports 20 workflow applications from astronomy, earth science, bioinformatics, weather, and ocean modeling**
- **Pre-generated workflows: a large collection of synthetic workflow samples is available (5 workflow applications and 2,840 workflow instances)**

Epigenomics
The epigenomics workflow created by the USC Epigenome Center and the Pegasus Team is used to automate various operations in genome sequence processing.

24 Node DAX
46 Node DAX
100 Node DAX
997 Node DAX

USC Viterb
School of Engi

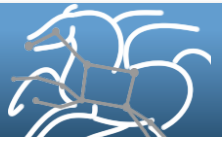
Workflow Simulator: WorkflowSim

A workflow simulator for distributed environments

- Workflow-level support (data dependency) on top of CloudSim
- Contains a model of system overheads
- Supports task failures, monitoring, task retry, task clustering

- Can be used to evaluate algorithms and techniques in task scheduling, task clustering, resource provisioning, and data placement etc.
- Input: DAX files from Synthetic Workflow Generator or other DAX generators
- Output: makespan, resource usage, cost, etc.

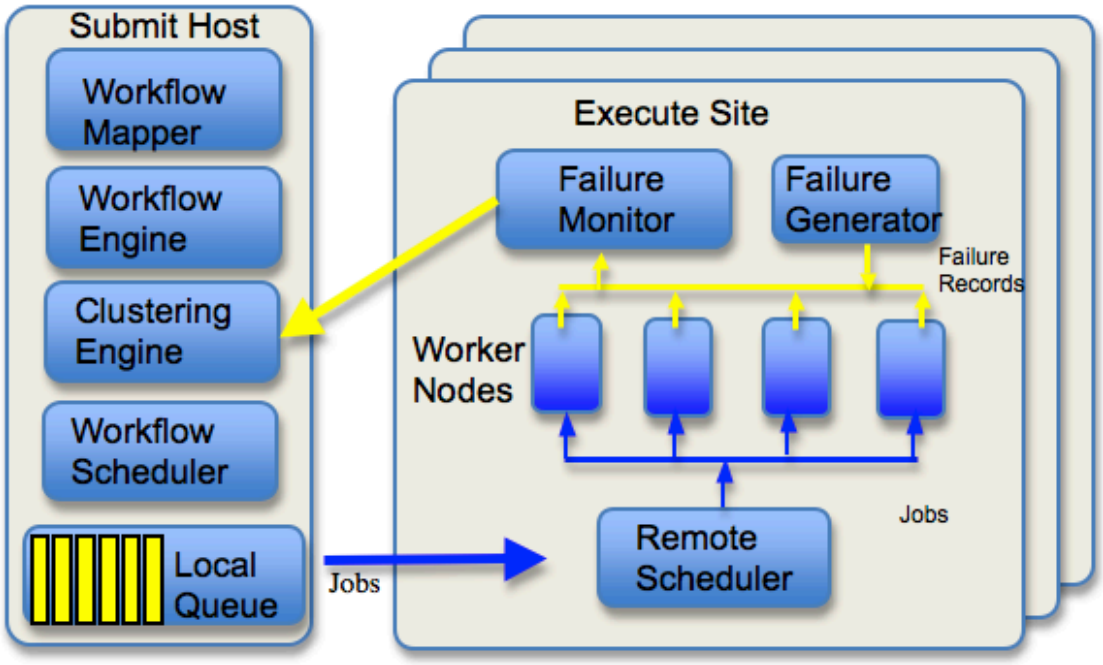
Source available on github and open to contributions
https://github.com/WorkflowSim



Workflow Simulator: WorkflowSim

Components

- **Behavior based on Pegasus**
  - Workflow Mapper/Compiler, Workflow Engine, Clustering Engine, Workflow Scheduler
- **Additional features**
  - Failure Generator
  - Failure Monitor
  - System Overheads
  - Monetary Cost

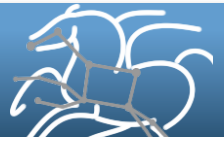


Simulation Usage Examples

- **Balanced Task Clustering: investigate the dependency and runtime imbalance problem in task clustering**
  - **Used synthetic workflow generator to create a large set of workflow instances**

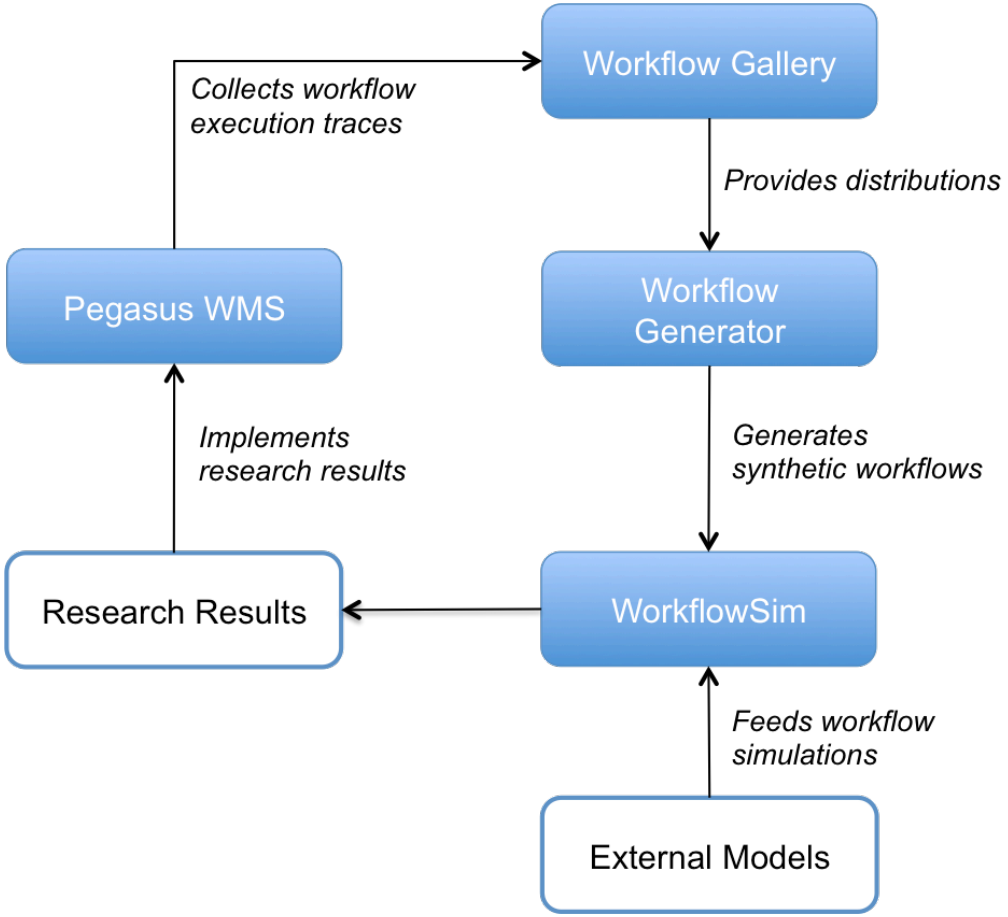
- **Fault-Tolerant Task Clustering: improve the task clustering in a faulty environment**
  - **Relied on the generation of transient failures**
  - **Synthetic workflows**

- **Energy-Efficiency: develop an energy consumption model for large-scale infrastructure**



Conclusion

- **A collection of tools and data that have enabled research in new methods and systems**
  - Execution traces
  - Synthetic workflow generator
  - Workflow simulator
- **We use these tools to**
  - **Generate traces**
  - **Analyze and profile traces**
  - **Vary system configurations and workflow instances**
  - **Evaluate the results in simulator**
  - **Implement promising approaches in Pegasus**
- **The tools are not limited to the Pegasus community**



Current Community Resources

Contributions Welcome!

- **Pegasus Workflow Management System:** http://pegasus.isi.edu
- **Workflow Archive and Workflow Generator:** www.workflowarchive.org
- **WorkflowSim:** www.workflowsim.org **and** https://github.com/WorkflowSim