# Scientific Workflows - How Pegasus can enhance your DAGMan experience

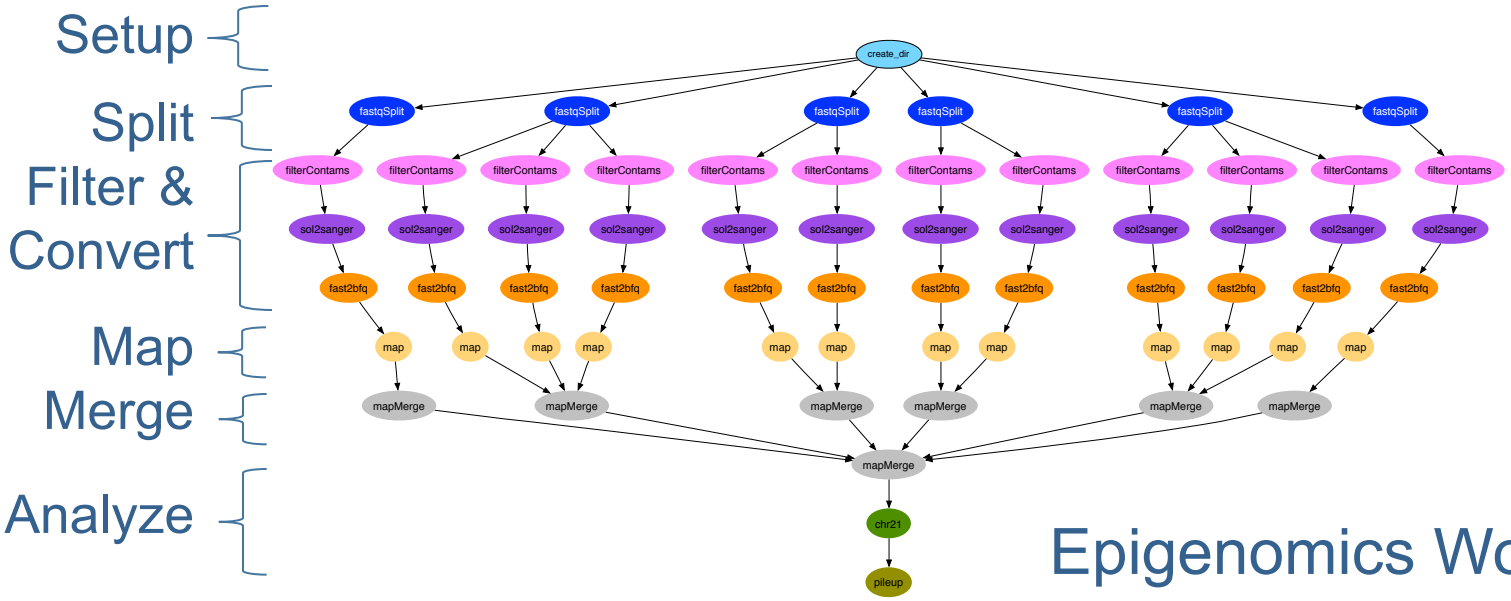
Karan Vahi

Science Automation Technologies Group

USC Information Sciences Institute

# Scientific Workflows

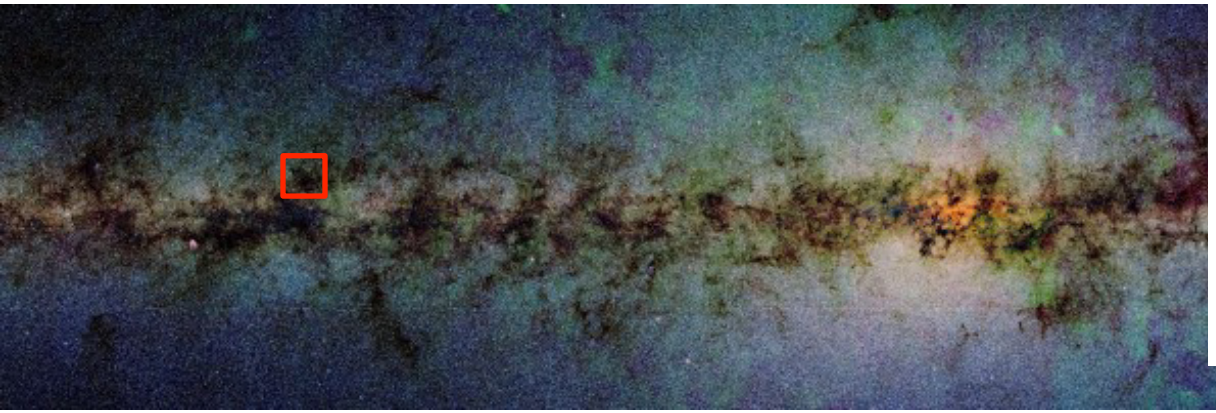
- Orchestrate complex, multi-stage scientific computations
- Often expressed as directed acyclic graphs (DAGs)
- Capture analysis pipelines for sharing and reuse
- Can execute in parallel on distributed resources

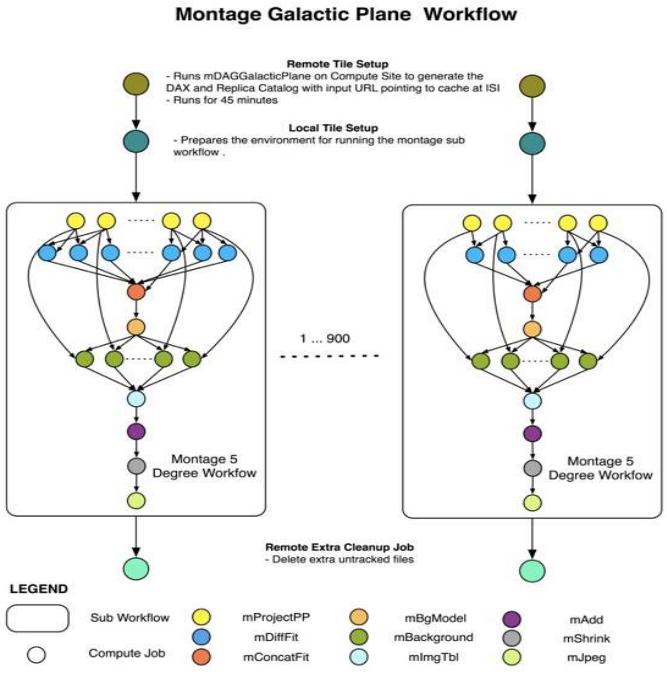



Epigenomics Workflow

# Workflows can be simple

# 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**

# HTCondor DAGMan

- **All the previous workflow pipeline are DAG based**
- **DAGMan is a reliable and a scalable workflow executor**
  - Sits on top of HTCondor Schedd
  - Can handle very large workflows ( to order of million tasks)
- **Has useful reliability features in-built**
  - Automatic Job retries
  - Rescue DAG's ( recover from where you left off in case of failures)
- **Throttling for jobs in a workflow**
- **Easy way to describe workflows**
  - Users can directly express their workflows as DAGMan Dags and condor submit files.

# However - !

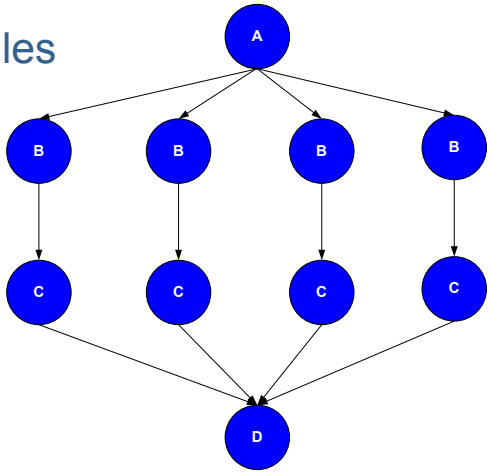
- **If you code directly against DAGMan**
  - **You are potentially limiting yourself to a single execution resource**
  - **You are responsible for figuring out how to access the data**
    - Where do the inputs for your pipeline exist and what file servers to use?
    - How do you ship in the small/large amounts data required by the workflows?
    - Can I use SRM? How about GridFTP? HTTP and Squid proxies?
    - Can I use Cloud based storage like S3 on EC2?
  - **How do you leverage underlying infrastructure setups**
    - E.g. On a HPC cluster in XSEDE, you can rely on the shared filesystem to store data, and use it for all jobs in the workflow
    - On OSG and in computational clouds each job needs to bring it's own inputs
  - **What happens if somebody wants setup your pipleine on their own resource?**
    - Can your pipeline on Amazon EC2 one day, and a PBS cluster the next?

# Other Workflow Challenges

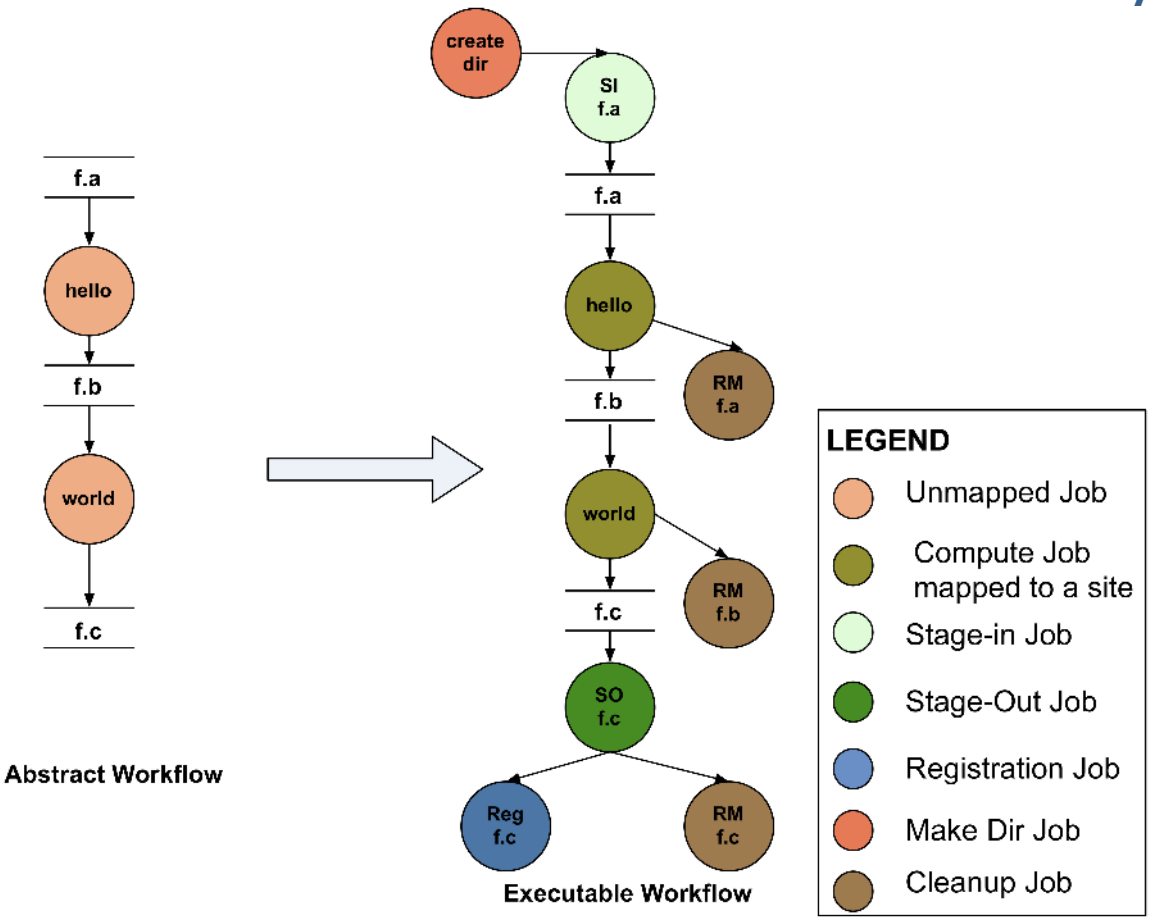
- **Provenance**
  - Can you go back and find out how and where data was produced?
- **Debug and Monitor Workflows**
  - Users need automated tools to go through the log files
  - Need to correlate data across lots of log files
  - Need to know what host a job ran on and how it was invoked
- **Restructure Workflows for Improved Performance**
  - Short running tasks?
  - Data placement?
- **Integrate with higher level tools such as HubZero and provisioning infrastructure**
  - such as GlideinWMS, BOSCO

# Pegasus Workflow Management System

- **NSF funded project since 2001**
  - Developed as a collaboration between USC Information Sciences Institute and the Condor Team at UW Madison
- **Builds on top of HTCondor DAGMan.**
- **Abstract Workflows - Pegasus input workflow description**
  - Workflow “high-level language”
  - Only identifies the computation, devoid of resource descriptions, devoid of data locations
  - File Aware – For each task you specify the input and output files
- **Pegasus is a workflow “compiler” (plan/map)**
  - Target is DAGMan DAGs and Condor submit files
  - Transforms the workflow for performance and reliability
  - Automatically locates physical locations for both workflow components and data
  - Collects runtime provenance



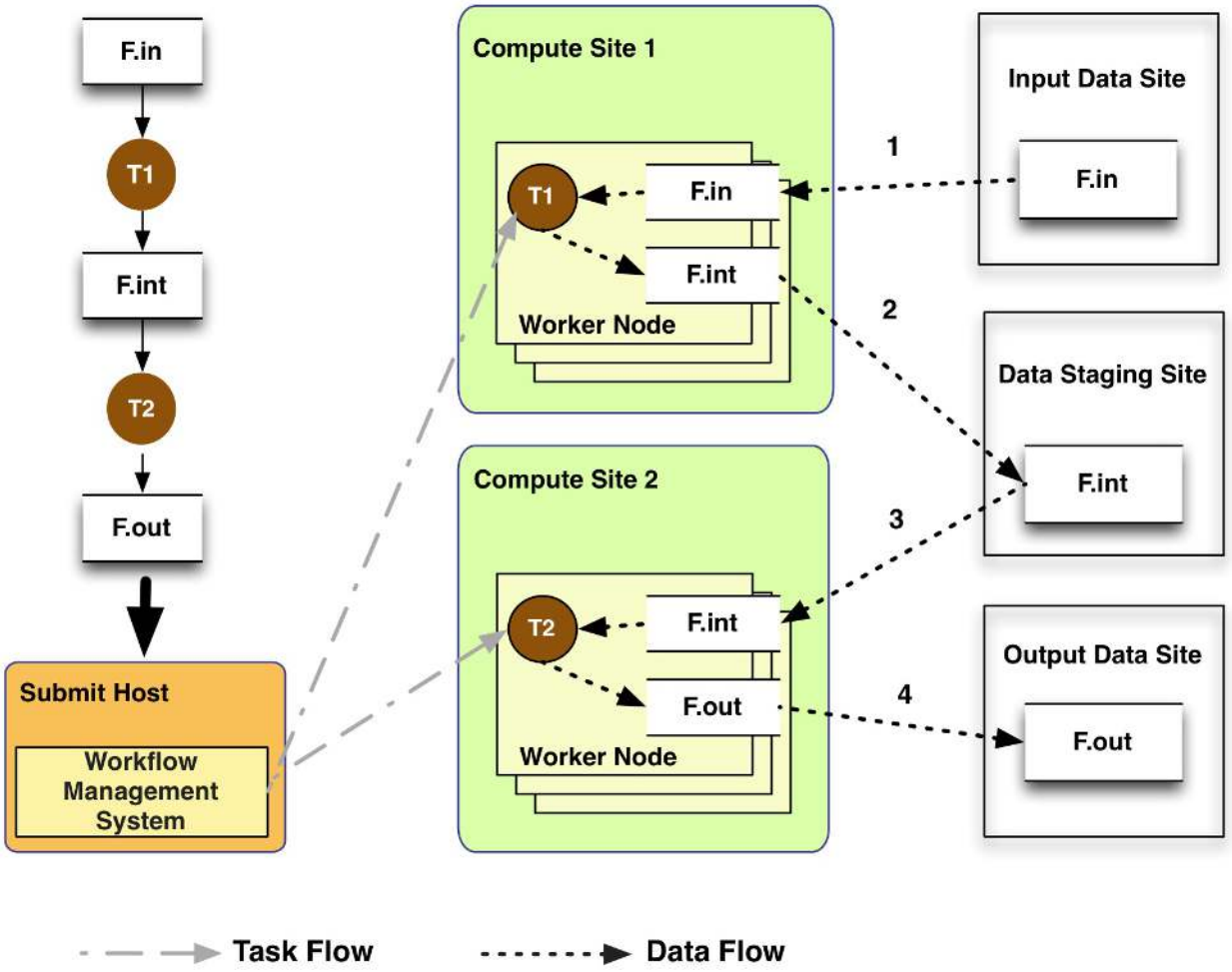
# Abstract to Executable Workflow Mapping




- Abstraction provides
  - Ease of Use (do not need to worry about low-level execution details)
  - Portability (can use the same workflow description to run on a number of resources and/or across them)
  - Gives opportunities for optimization and fault tolerance
    - automatically restructure the workflow
    - automatically provide fault recovery (retry, choose different resource)

**Pegasus Guarantee -** Wherever and whenever a job runs it's inputs will be in the directory where it is launched.

# General Workflow Execution Model



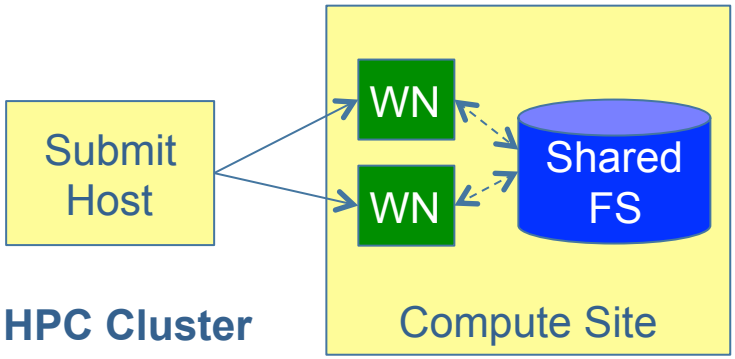

- Most of the tasks in scientific workflow applications require POSIX file semantics
  - Each task in the workflow opens one or more input files
  - Read or write a portion of it and then close the file.

- Input Data Site, Compute Site and Output Data Sites can be co-located
  - Example: Input data is already present on the compute site.

# Supported Data Staging Approaches - I

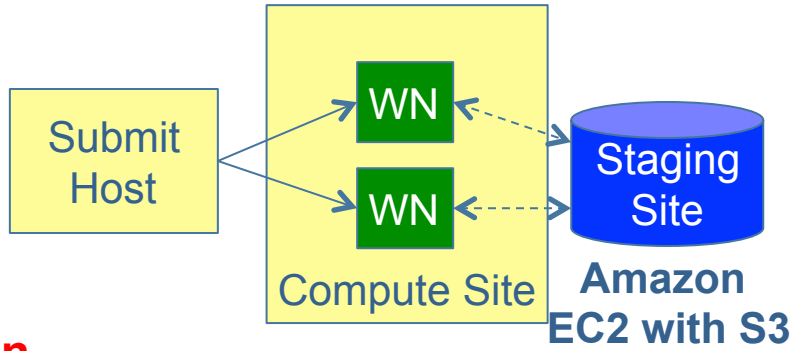
## Shared Filesystem setup (typical of XSEDE and HPC sites)

- Worker nodes and the head node have a shared filesystem, usually a parallel filesystem with great I/O characteristics
- Can leverage symlinking against existing datasets
- Staging site is the shared-fs.




## Non-shared filesystem setup with staging site (typical of OSG and EC 2)

- Worker nodes don't share a filesystem.
- Data is pulled from / pushed to the existing storage element.
- A separate staging site such as S3.



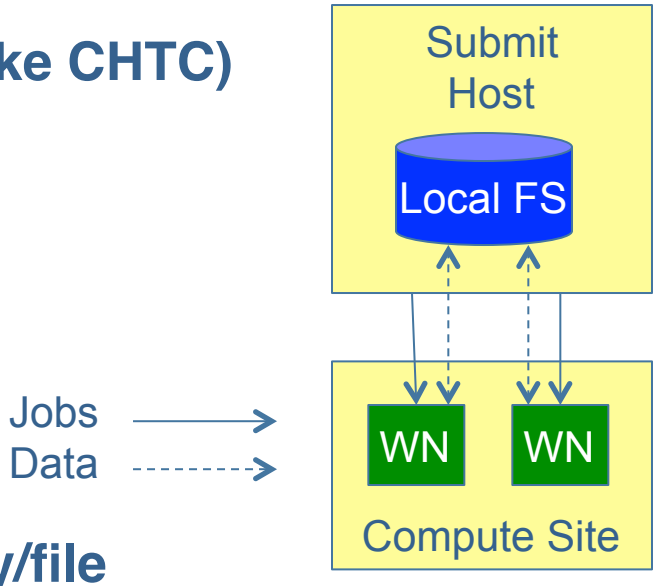

**HubZero uses Pegasus to run a single application worklow across sites, leveraging shared filesystem at local PBS cluster and non shared filesystem setup at OSG!**

Jobs →
Data ⇢

# Supported Data Staging Approaches - II

## Condor IO ( Typical of large Condor Pools like CHTC)

- Worker nodes don't share a filesystem
- Symlink against datasets available locally
- Data is pulled from / pushed to the submit host via Condor file transfers
- Staging site is the submit host.




## Supported Transfer Protocols – for directory/file creation and removal, file transfers

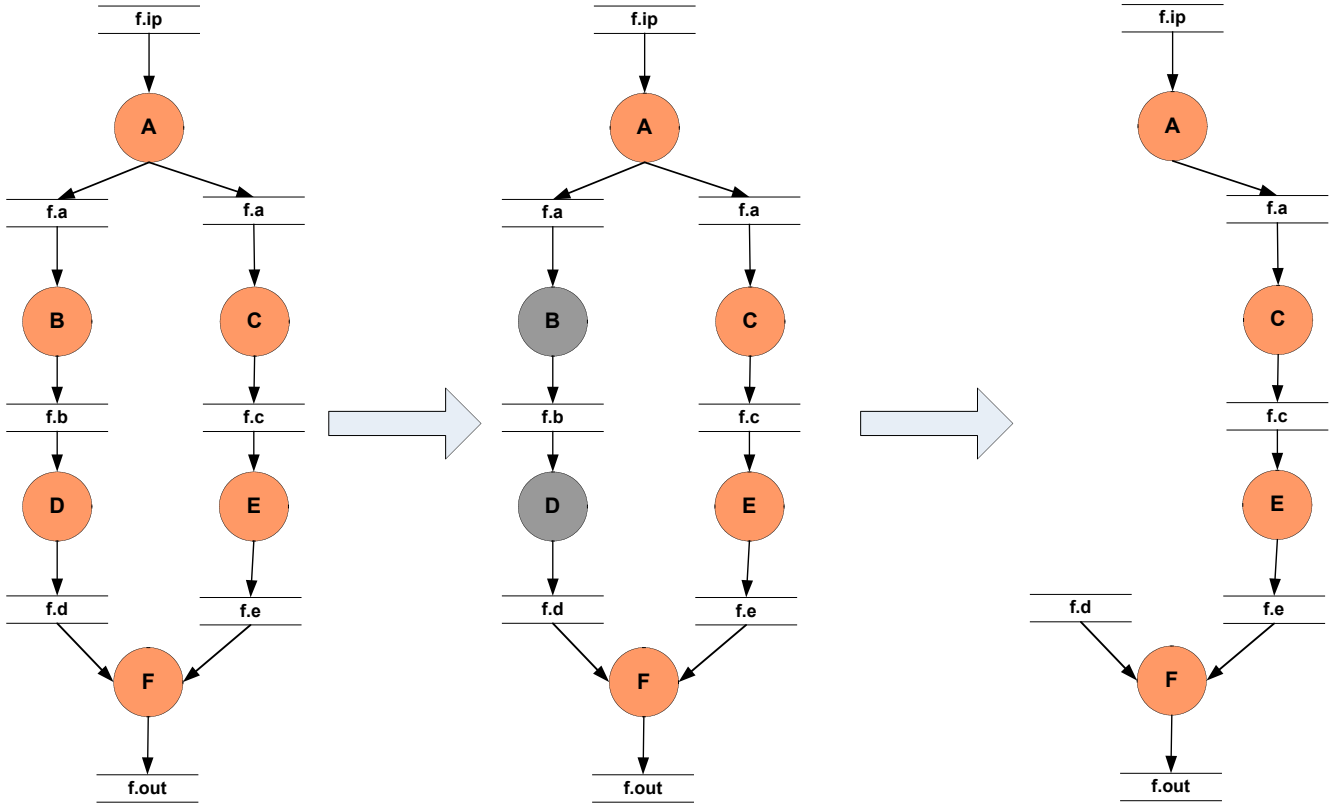
- HTTP
- SCP
- GridFTP
- IRODS
- S3 / Google Cloud Storage
- Condor File IO
- File Copy
- OSG Stash

**Using Pegasus allows you to move from one deployment to another without changing the workflow description!**

**Pegasus Data Management Tools**

pegasus-transfer, pegasus-create-dir, pegasus-cleanup support client discovery, parallel transfers, retries, and many other things to improve transfer performance and reliability

# Workflow Reduction (Data Reuse)




Abstract Workflow

File f.d exists somewhere.
Reuse it.
Mark Jobs D and B to delete

Delete Job D and Job B

**Useful when you have done a part of computation and then realize the need to change the structure. Re-plan instead of submitting rescue DAG!**

# File cleanup

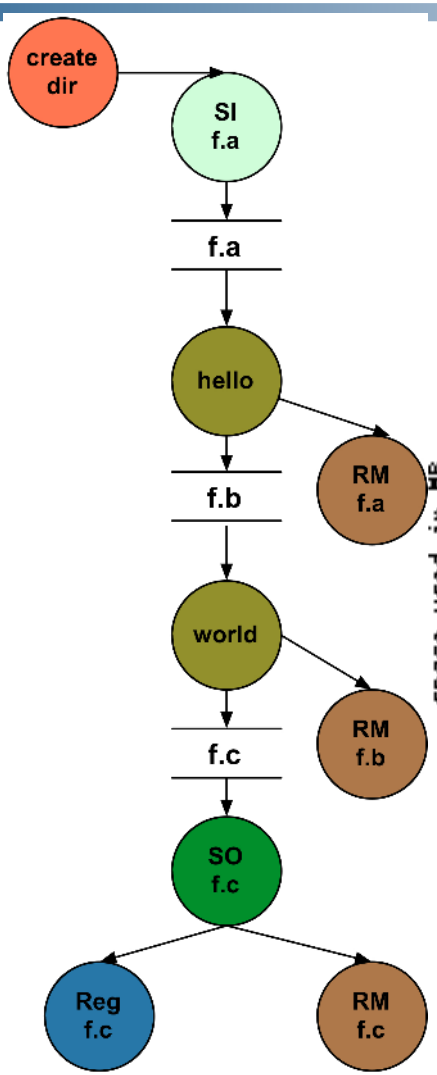
- **Problem: Running out of disk space during workflow execution**

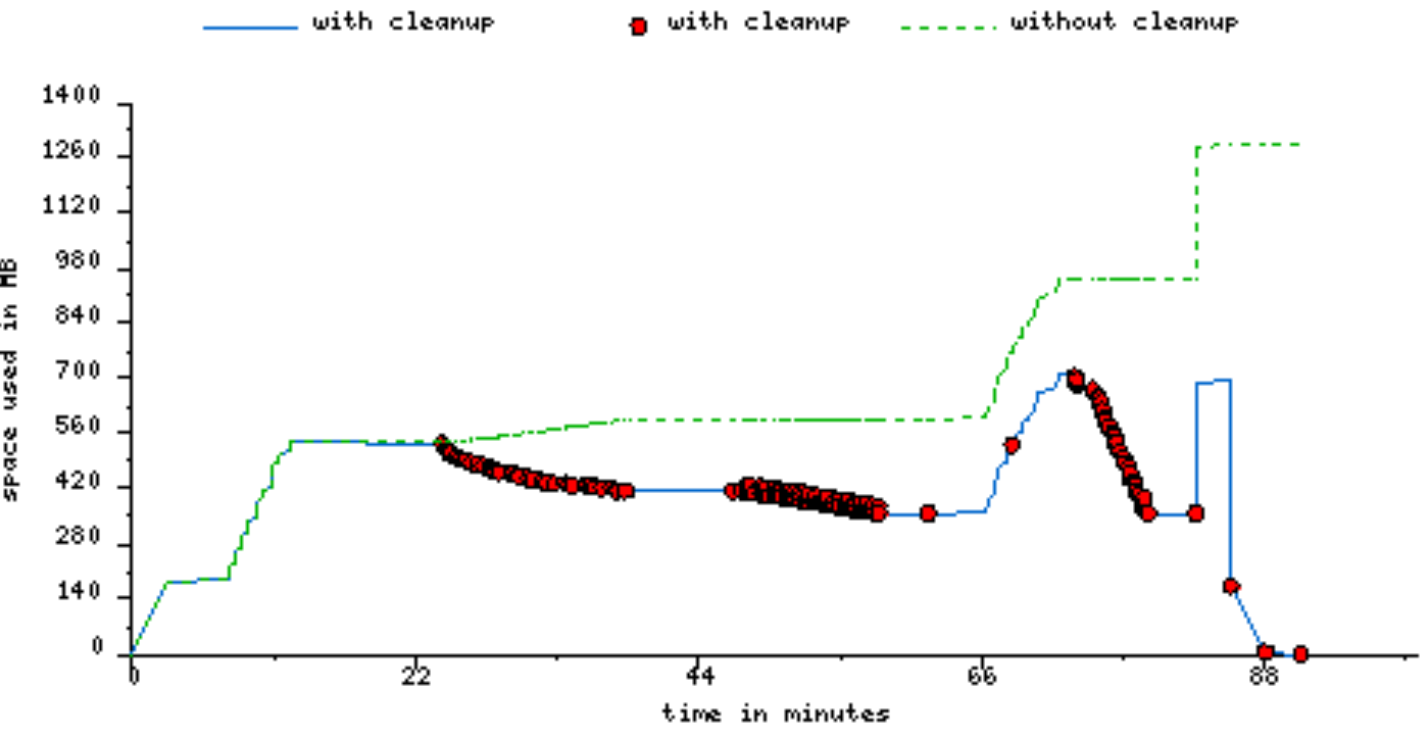
- **Why does it occur**
  - Workflows could bring in huge amounts of data
  - Data is generated during workflow execution
  - Users don't worry about cleaning up after they are done

- **Solution**
  - **Do cleanup after workflows finish**
    - Add a leaf Cleanup Job ( Available in 4.4 Release onwards)
  - **Interleave cleanup automatically during workflow execution.**
    - Requires an analysis of the workflow to determine, when a file is no longer required
  - **Cluster the cleanup jobs by level for large workflows**
  - In 4.6 release, users should be able to specify maximum disk space that should not be exceeded. Pegasus will restructure the workflow accordingly.

**Real Life Example: Used by a UCLA genomics researcher to delete TB's of data automatically for long running workflows!!**

# File cleanup (cont)




**Montage 1 degree workflow run with cleanup**

# Job Checkpoint Files

- **A job can specify that it uses one or more checkpoint files**
- **Checkpoint files are both input files and output files**
  - Recommended - application code should create checkpoint files periodically.
- **Users specify checkpoint.time ( the time at which the job creates a checkpoint file ) and the maxwalltime of a site.**
- **Pegasus will stage-out these files in the case that job fails**
  - Typically due to a timeout on long-running jobs
  - They are sent a TERM signal at checkpoint.time associated with the jobs.
  - A KILL signal is sent K seconds after the TERM signal ( where K is (maxwalltime – checkpoint.time)/2
- **Pegasus will stage-in these files before retrying the job**
  - They will appear in the working directory of the job

**Works for Grid Universe not just vanilla!**

**Used by LIGO to run long running inspiral jobs on VIRGO compute resources, where maxwalltime is 12 hours per site policy.**



# Workflow and Task Notifications

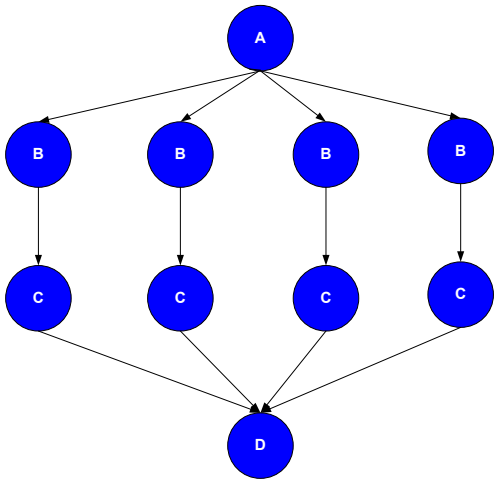
- **Users want to be notified at certain points in the workflow or on certain events.**

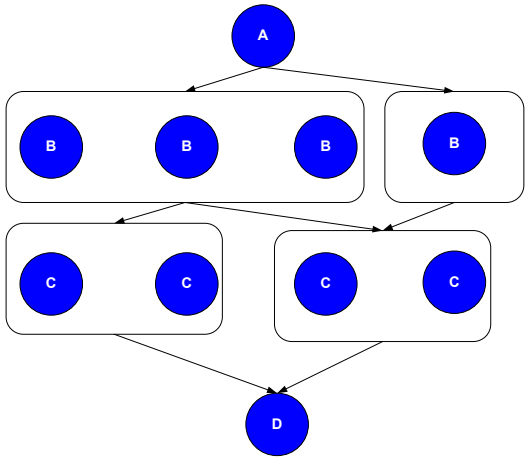
- **Support for adding notification to workflow and tasks**

- **Event based callouts**
  - On Start, On End, On Failure, On Success
  - Provided with email and jabber notification scripts
  - Can run any user provided scripts
  - Defined in the DAX

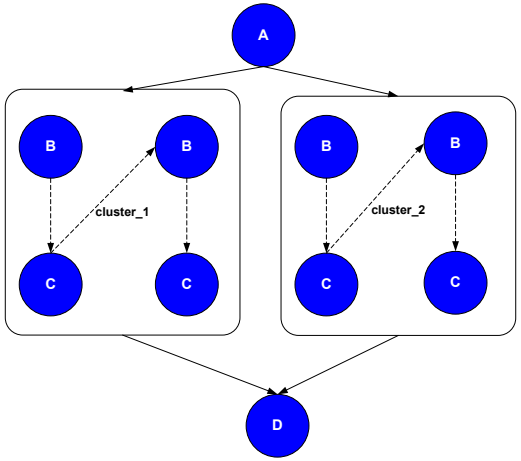
# Workflow Restructuring to improve application performance

- **Cluster small running jobs together to achieve better performance**
- **Why?**
  - Each job has scheduling overhead – need to make this overhead worthwhile
  - Ideally users should run a job on the grid that takes at least 10/30/60/? minutes to execute
  - Clustered tasks can reuse common input data – less data transfers




Horizontal clustering




Label-based clustering

# Workflow Monitoring - Stampede

- **Leverage Stampede Monitoring framework with DB backend**
  - Populates data at runtime. A background daemon monitors the logs files and populates information about the workflow to a database
  - Stores workflow structure, and runtime stats for each task.

- **Tools for querying the monitoring framework**
  - **pegasus-status**
    - Status of the workflow
  - **pegasus-statistics**
    - Detailed statistics about your finished workflow

```
------------------------------------------------------------------------------
Type           Succeeded Failed  Incomplete  Total     Retries   Total+Retries
Tasks          135002    0       0           135002    0         135002
Jobs           4529      0       0           4529      0         4529
Sub-Workflows  2         0       0           2         0         2
------------------------------------------------------------------------------

Workflow wall time                                : 13 hrs, 2 mins, (46973 secs)
Workflow cumulative job wall time                 : 384 days, 5 hrs, (33195705 secs)
Cumulative job walltime as seen from submit side : 384 days, 18 hrs, (33243709 secs)
```



# Workflow Debugging Through Pegasus

- **After a workflow has completed, we can run pegasus-analyzer to analyze the workflow and provide a summary of the run**

- **pegasus-analyzer's output contains**
  - **a brief summary section**
    - showing how many jobs have succeeded
    - and how many have failed.
  - **For each failed job**
    - showing its last known state
    - exitcode
    - working directory
    - the location of its submit, output, and error files.
    - any stdout and stderr from the job.

**Alleviates the need for searching through large DAGMan and Condor logs!**

# Workflow Monitoring Dashboard: pegasus-dashboard

- **A python based online workflow dashboard**
  - Uses the FLASK framework
  - Packaged with Pegasus 4.5 release.
  - Queries the STAMPEDE database
- **Lists all the user workflows on the home page and are color coded.**
  - Green indicates a successful workflow,
  - Red indicates a failed workflow
  - Blue indicates a running workflow
- **Explore Workflow and Troubleshoot ( Workflow Page )**
  - Has identifying metadata about the workflow
  - Tabbed interface to
    - List of sub workflows
    - Failed jobs
    - Running  jobs
    - Successful jobs.



# Pegasus Dashboard

## Workflow Listing Page Shows Successful, Failed and Running Workflows


Show results for last week

Show 10 entries Search:

| Workflow Label | Submit Host | Submit Directory | State | Submitted On |
|---|---|---|---|---|
| blackdiamond | cartman | /lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-WT-T19A/test/core/019-black-label/work/bamboo/pegasus/blackdiamond/20150514T092718-0700 | Successful | Thu, 14 May 2015 09:27:18 |
| rosetta | cartman | /lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-WT-SSHFTP030/test/core/030-pegasuslite-sshftp/work/bamboo/pegasus/rosetta/20150514T092732-0700 | Failed | Thu, 14 May 2015 09:27:32 |
| horizontal-clustering-test | cartman | /lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-WT-T10D/test/core/010-runtime-clustering/work/submit/bamboo/pegasus/horizontal-clustering-test/20150514T092741-0700 | Successful | Thu, 14 May 2015 09:27:41 |

| Workflow Wall Time | 7 mins 9 secs |
|---|---|
| Workflow Cumulative Job Wall Time | 2 mins 51 secs |
| Cumulative Job Walltime as seen from Submit Side | 5 mins 49 secs |
| Workflow Retries | 0 |

Workflow Statistics

Job Breakdown Statistics

## Workflow Statistics

Show 10 entries Search:

| Transformation | Count | Succeeded | Failed | Min | Max | Mean | Total |
|---|---|---|---|---|---|---|---|
| dagman::post | 32 | 32 | 0 | 5 | 5 | 5 | 160 |
| mAdd:3.3 | 1 | 1 | 0 | 1.284 | 1.284 | 1.284 | 1.284 |
| mBackground:3.3 | 32 | 32 | 0 | 0.119 | 0.211 | 0.174 | 5.562 |
| mBgModel:3.3 | 1 | 1 | 0 | 16.949 | 16.949 | 16.949 | 16.949 |
| mConcatFit:3.3 | 1 | 1 | 0 | 1.022 | 1.022 | 1.022 | 1.022 |
| mDiffFit:3.3 | 73 | 73 | 0 | 0.088 | 0.370 | 0.192 | 14.044 |
| mImgtbl:3.3 | 1 | 1 | 0 | 0.128 | 0.128 | 0.128 | 0.128 |
| mJPEG:3.3 | 1 | 1 | 0 | 0.529 | 0.529 | 0.529 | 0.529 |
| mProjectPP:3.3 | 32 | 32 | 0 | 1.875 | 2.040 | 1.945 | 62.246 |
| mShrink:3.3 | 1 | 1 | 0 | 0.488 | 0.488 | 0.488 | 0.488 |

Showing 1 to 10 of 14 entries First Previous 1 2 Next Last

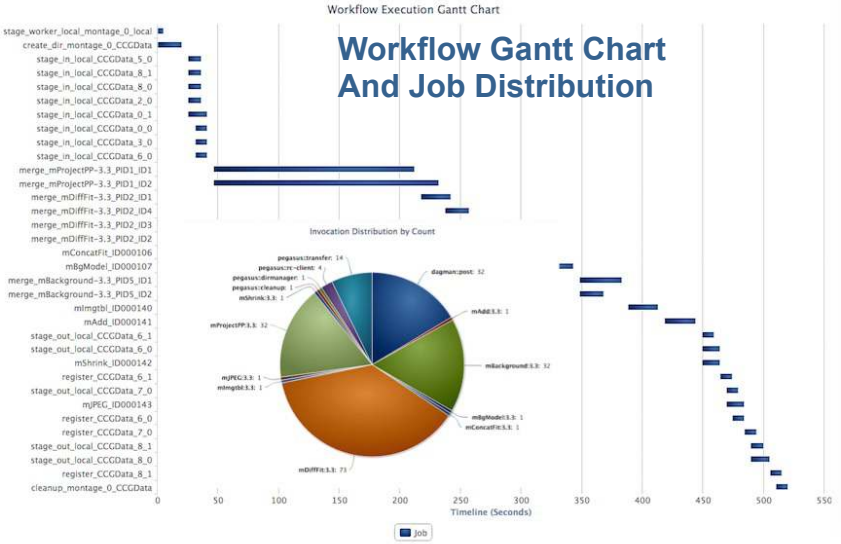
Charts

Job Distribution

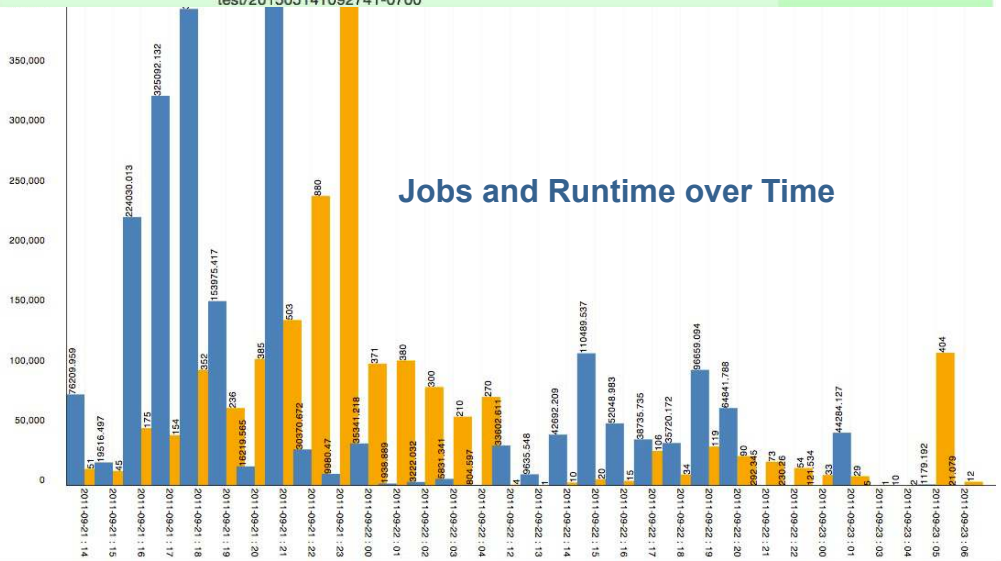
Time Chart

Gantt Chart

## Workflow Gantt Chart And Job Distribution




## Jobs and Runtime over Time



Home Planner Metrics DAGMan Metrics Downloads Maps Admin May 20, 2015

Show results from start of May / 2014 to end of May / 2015 Update

Showing 2014-04-30 17:00:00 to 2015-05-31 17:00:00

## Metametrics

| | |
|---|---|
| Number of raw objects | 259,898 |
| Number of invalid objects | 5 |
| Number of processed objects | 259,893 |

## Planner Metrics

| | |
|---|---|
| Workflows Planned | 230,673 |
| Tasks Planned | 1,288,322,926 |
| Jobs Planned | 234,209,972 |
| Errors Reported | 5,883 |

## DAGMan Metrics

| | |
|---|---|
| Workflow Runs | 27,446 |
| Total Jobs | 2,928,140 |
| Jobs Submitted | 1,954,608 |
| Jobs Succeeded | 1,939,640 |
| Jobs Failed | 14,968 |
| Total Runtime (hrs) | 0.00 |

## Download Metrics

| | |
|---|---|
| Number of downloads | 1,751 |

**Metrics Usage May 2014-May2015**

**DAGMan metrics reporting only in 4.5 onwards**

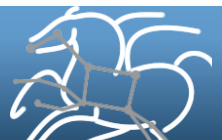
## Top Planner Domains

| Domain | Workflows | Tasks | Jobs |
|---|---|---|---|
| isi.edu | 50,028 | 289,840,396 | 128,095,184 |
| nanohub.org | 45,948 | 71,070 | 272,610 |
| mps.mpg.de | 40,960 | 1,300,183 | 1,626,657 |
| opensciencegrid.org | 23,629 | 501,008,491 | 33,269,886 |
| grid.iu.edu | 18,906 | 372,352,884 | 10,851,866 |

## Top Planner Hosts

| Host | Workflows | Tasks | Jobs |
|---|---|---|---|
| cartman.isi.edu | 45,270 | 279,893,163 | 122,228,810 |
| seismo3.mps.mpg.de | 40,914 | 1,299,455 | 1,625,707 |
| scatter.nanohub.org | 34,109 | 47,405 | 188,697 |
| xd-login.opensciencegrid.org | 23,629 | 501,008,491 | 33,269,886 |
| osg-xsede.grid.iu.edu | 18,906 | 372,352,884 | 10,851,866 |





# Summary –
# What Does Pegasus provide an Application - I

- **All the great features that DAGMan has**
  - Scalability / hierarchal workflows
  - Retries in case of failure.

- **Portability / Reuse**
  - User created workflows can easily be mapped to and run in different environments without alteration.

- **Performance**
  - The Pegasus mapper can reorder, group, and prioritize tasks in order to increase the overall workflow performance.

# Summary –
# What Does Pegasus provide an Application - II

- **Provenance**
  - Provenance data is collected in a database, and the data can be summaries with tools such as pegasus-statistics, pegasus-plots, or directly with SQL queries.

- **Reliability and Debugging Tools**
  - Jobs and data transfers are automatically retried in case of failures. Debugging tools such as pegasus-analyzer helps the user to debug the workflow in case of non-recoverable failures.

- **Data Management**
  - Pegasus handles replica selection, data transfers and output registrations in data catalogs. These tasks are added to a workflow as auxiliary jobs by the Pegasus planner.


## Relevant Links

- **Pegasus: http://pegasus.isi.edu**
- **Tutorial and documentation: http://pegasus.isi.edu/wms/docs/latest/**
- **Support: pegasus-users@isi.edu**
  **pegasus-support@isi.edu**


Acknowledgements

Pegasus Team, Condor Team, funding agencies, NSF, NIH, and everybody who uses Pegasus.