Before Tutorial Starts
Hands on Tutorial Exercises: Setup

Please claim an instance by putting your name next to an unused instance in:

https://tinyurl.com/pegasus-htc23

Follow the link next to your name.

(This is the same (but hosted) as the self-guided tutorial available in the Pegasus documentation: https://pegasus.isi.edu/documentation/user-guide/tutorial.html)

If we are not finishing here today, feel free to keep exploring on your own.
Introduction to Pegasus

Karan Vahi

Information Sciences Institute, University of Southern California

vahi@isi.edu
1. Introduction
What are Scientific Workflows

- Conducts a series of computational tasks.
  - Resources distributed across Internet.

- Chaining (outputs become inputs) replaces manual hand-offs.
  - Accelerated creation of products.

- Ease of use - gives non-developers access to sophisticated codes.
  - Resources distributed across Internet.

- Provides framework to host or assemble community set of applications.
  -Honors original codes. Allows for heterogeneous coding styles.

- Framework to define common formats or standards when useful.
  - Promotes exchange of data, products, codes. Community metadata.

- Multi-disciplinary workflows can promote even broader collaborations.
  - E.g., ground motions fed into simulation of building shaking.

- Certain rules or guidelines make it easier to add a code into a workflow.
Why Pegasus?

- Automates Complex, Multi-stage Processing Pipelines
- Enables Parallel, Distributed Computations
- Automatically Executes Data Transfers
- Reusable, Aids Reproducibility
- Records How Data was Produced (Provenance)
- Handles Failures with to Provide Reliability
- Keeps Track of Data and Files
- Ensures Data Integrity during workflow execution

NSF funded project since 2001, with close collaboration with HTCondor team

https://pegasus.isi.edu
Some of The Success Stories...
Southern California Earthquake Center’s CyberShake

Mix of MPI and single-core jobs, mix of CPU, GPU codes. Large data sets (10s of TBs), ~300 workflows with 420,000 tasks each
Supported since 2005: changing CI, x-platform execution

First Physics-Based "Shake map" of Southern California

Laser Interferometer Gravitational-Wave Observatory (LIGO)

High-throughput computing workload, access to HPC resources, ~ 21K Pegasus workflows, ~ 107M tasks
Supported since 2001, distributed data, opportunistic computing resources

First direct detection of a gravitational wave (colliding black holes)

XENONnT - Dark Matter Search

Custom data management
Rucio for data management
MongoDB instance to track science runs and data products.

Monte Carlo simulations and the main processing pipeline.
Data Flow for LIGO Pegasus Workflows in OSG

60,000 Compute Tasks
Input Data: 5000 files (10GB total)
Output Data: 60,000 files (60GB total)
Processed Data: 725 GB

Executed on LIGO Data Grid, EGI,
Open Science Grid and XSEDE

Advanced LIGO Laser Interferometer Gravitational Wave Observatory

https://pegasus.isi.edu
Processing instrument data in real time
Automated Quality Control of Phenotypic Datasets

The NIMH Center for Collaborative Genomic Studies on Mental Disorders, now known as the NIMH Repository and Genomics Resource (NRGR), maintains biomaterials, demographic, and phenotypic data from over 200,000 well-characterized individuals with a range of psychiatric illnesses, their family members, and unaffected controls.

Easy to Use Web-Based Interface
- Simple Submission
- Real-time Monitoring and Error Reports
- After automated QC, submit corrected files for expert curation

Scalable
- Workflow based architecture using Pegasus WMS

Extensible Design
- Easily add new QC steps, and checks

Enables Complex checks
- Pedigree Checks
- QC Checks validating data with external sources
- QC Checks can correlate data across multiple files and across multiple fields within files

- Ensures high-quality uniform data deposited at NRGR
- Better resource utilization: solve most QC problems automatically, use expert curation for hard cases

https://pegasus.isi.edu
Key Pegasus Concepts

- **Pegasus WMS** == Pegasus planner (mapper) + DAGMan workflow engine + HTCondor scheduler/broker
  - Pegasus maps workflows to infrastructure
  - DAGMan manages dependencies and reliability
  - HTCondor is used as a broker to interface with different schedulers

- **Workflows are DAGs**
  - Nodes: jobs, edges: dependencies
  - No while loops, no conditional branches
  - Jobs are standalone executables

- **Planning occurs ahead of execution**

- **Planning converts an abstract workflow into a concrete, executable workflow**
  - Planner is like a compiler

https://pegasus.isi.edu
Input Workflow Specification **YAML formatted**

**Portable Description**
Users do not worry about low level execution details

**Logical Filename (LFN)**
Platform independent (abstraction)

**Transformation**
Executables (or programs) platform independent

**Output Workflow**

**Stage-in Job**
Transfers the workflow input data

**Cleanup Job**
Removes unused data

**Stage-out Job**
Stage-out generated output data

**Registration Job**
Registers the workflow output data

https://pegasus.isi.edu
Pegasus Deployment

- **Workflow Submit Node**
  - Pegasus WMS
  - HTCondor

- **One or more Compute Sites**
  - Compute Clusters
  - Cloud
  - OSG

- **Input Sites**
  - Host Input Data

- **Data Staging Site**
  - Coordinate data movement for workflow

- **Output Site**
  - Where output data is placed
Pegasus-transfer

Pegasus’ internal data transfer tool with support for a number of different protocols

- **Directory creation, file removal**
  - If protocol can support it, also used for cleanup

- **Two stage transfers**
  - e.g., GridFTP to S3 = GridFTP to local file, local file to S3

- **Parallel transfers**

- **Automatic retries**

- **Credential management**
  - Uses the appropriate credential for each site and each protocol (even 3rd party transfers)

<table>
<thead>
<tr>
<th>Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
</tr>
<tr>
<td>SCP</td>
</tr>
<tr>
<td>GridFTP</td>
</tr>
<tr>
<td>Globus</td>
</tr>
<tr>
<td>Online</td>
</tr>
<tr>
<td>iRods</td>
</tr>
<tr>
<td>Amazon S3</td>
</tr>
<tr>
<td>Google Storage</td>
</tr>
<tr>
<td>SRM</td>
</tr>
<tr>
<td>FDT</td>
</tr>
<tr>
<td>Stashcp</td>
</tr>
<tr>
<td>Rucio</td>
</tr>
<tr>
<td>cp</td>
</tr>
<tr>
<td>ln -s</td>
</tr>
</tbody>
</table>

https://pegasus.isi.edu
Real-time monitoring of workflow executions. It shows the status of the workflows and jobs, job characteristics, statistics and performance metrics.

Provenance data is stored into a relational database.

https://pegasus.isi.edu
command-line...

$ pegasus-status pegasus/examples/split/run0001
STAT IN_STATE JOB
Run 00:39 split-0 (/home/pegasus/examples/split/run0001)
Idle 00:03 split_ID0000001
Summary: 2 Condor jobs total (I:1 R:1)

UNRDY READY PRE IN_Q POST DONE FAIL %DONE STATE DAGNAME
14 0 0 I 0 2 0 11.8 Running *split-0.dag

$ pegasus-analyzer pegasus/examples/split/run0001
pegasus-analyzer: initializing...

**************************Summary**************************
Total jobs : 7 (100.00%)
# jobs succeeded : 7 (100.00%)
# jobs failed : 0 (0.00%)
# jobs unsubmitted : 0 (0.00%)

$ pegasus-statistics -s all pegasus/examples/split/run0001
-------------------------------------------------------------------
<table>
<thead>
<tr>
<th>Type</th>
<th>Succeeded</th>
<th>Failed</th>
<th>Incomplete</th>
<th>Total</th>
<th>Retries</th>
<th>Total+Retries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tasks</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>Jobs</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>0</td>
<td>17</td>
</tr>
<tr>
<td>Sub-Workflows</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
-------------------------------------------------------------------
Workflow wall time : 2 mins, 6 secs
Workflow cumulative job wall time : 38 secs
Cumulative job wall time as seen from submit side : 42 secs
Workflow cumulative job badput wall time :
Cumulative job badput wall time as seen from submit side :

Provenance Data can be Summarized

or

Used for Debugging

https://pegasus.isi.edu
And if a job fails?

Postscript
- detects non-zero exit code output
- parsing for success or failure
- message exceeded timeout do not produced expected output files

Job Retry
- helps with transient failures
- set number of retries per job and run

Checkpoint Files
- job generates checkpoint files
- staging of checkpoint files is automatic on restarts

Rescue DAGs
- workflow can be restarted from checkpoint file
- recover from failures with minimal loss

https://pegasus.isi.edu
Pegasus is part of the ACCESS support strategy

Pegasus is be used as a tier 1 tool

Central Open OnDemand instance with Pegasus, HTCondor and Jupyter

It is be easy to run HTC workflows across ACCESS sites
ACCESS Pegasus

Bring your workflows to ACCESS!

- Execute scientific workflows across ACCESS resources

- OpenOnDemand Portal: **has all you need**: Jupyter Notebooks, ACCESS authentication, Pegasus workflow management, and HTCondor job management

- **Bring your own ACCESS capacity**: HTCondor Annex - pilot jobs automatically create a virtual HTCondor pool

https://access.pegasus.isi.edu

More at: support.access-ci.org/pegasus
2. Hands on Exercises
Before Tutorial Starts

Hands on Tutorial Exercises: Setup

Please claim an instance by putting your name next to an unused instance in:

https://tinyurl.com/pegasus-htc23

Follow the link next to your name.

(This is the same (but hosted) as the self-guided tutorial available in the Pegasus documentation: https://pegasus.isi.edu/documentation/user-guide/tutorial.html)

If we are not finishing here today, feel free to keep exploring on your own.
2.1 API

https://pegasus.isi.edu
Key Pegasus Concepts

Pegasus WMS == Pegasus planner (mapper) + DAGMan workflow engine + HTCondor scheduler/broker

- Pegasus maps workflows to infrastructure
- DAGMan manages dependencies and reliability
- HTCondor is used as a broker to interface with different schedulers

Workflows are DAGs

- Nodes: jobs, edges: dependencies
- No while loops, no conditional branches
- Jobs are standalone executables

Planning occurs ahead of execution

Planning converts an abstract workflow into a concrete, executable workflow

- Planner is like a compiler

https://pegasus.isi.edu
Input Workflow Specification

YAML formatted

Portable Description
Users do not worry about low level execution details

Directed-acyclic graphs

Output Workflow

Logical Filename (LFN)
Platform independent (abstraction)

Transformation
Executables (or programs)
Platform independent

Stage-in Job
Transfers the workflow input data

Cleanup Job
Removes unused data

Stage-out Job
Stage-out generated output data

Registration Job
Registers the workflow output data

https://pegasus.isi.edu
2.2 Debugging

https://pegasus.isi.edu
2.3 Command Line Tools

https://pegasus.isi.edu
Get Started

Pegasus Website
https://pegasus.isi.edu

Users Mailing List
pegasus-users@isi.edu

Support
pegasus-support@isi.edu

Slack
Ask for an invite by trying to join pegasus-users.slack.com in the Slack app

Pegasus Online Office Hours
https://pegasus.isi.edu/blog/online-pegasus-office-hours/

Bi-monthly basis on second Friday of the month, where we address user questions and also apprise the community of new developments.

YouTube Channel
https://www.youtube.com/channel/UCwJQln1CqBvTJqiNr9X9F1Q/featured

Pegasus in 5 Minutes