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.
  - Avoids need to download-install-learn how to use someone else's code.
- 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.

Workflow Building Blocks

Slide Content Courtesy of David Okaya, SCEC, USC

https://pegasus.isi.edu
Workflow Challenges Across Domains

• Need to describe complex workflows in a simple way

• Need to access distributed, heterogeneous data and resources (heterogeneous interfaces)

• Need to deal with resources/software that change over time

• Ease of use. Ability to debug and monitor large workflows

Our Focus

• Separation between workflow description and workflow execution

• Workflow planning and scheduling (scalability, performance)

• Task execution (monitoring, fault tolerance, debugging, web dashboard)

• Provide additional assurances that a scientific workflow is not accidentally or maliciously tampered with during its execution.

https://pegasus.isi.edu
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...
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)

CPU jobs
(Mesh generation, seismogram synthesis)
1,094,000 node-hours

GPU jobs:
439,000 node-hours
AWP-ODC finite-difference code
5 billion points per volume, 23,000 timesteps
200 GPUs for 1 hour

Titan:
421,000 CPU node-hours, 110,000 GPU node-hours

Blue Waters:
673,000 CPU node-hours, 329,000 GPU node-hours

SCEC/USC

Pre-processing

Configuration Files (2 GB)

Mesh Generation

Velocity Mesh (120 GB)

SGT X Simulation

SGT Y Simulation

Post-processing

Strain Green Tensors (1.5 TB)

Seismograms & intensity measures (8TB)

286 Sites
4 Models
Each Workflow Has 420,000 Tasks

https://pegasus.isi.edu
Data Flow for LIGO Pegasus Workflows in OSG

Advanced LIGO Laser Interferometer Gravitational Wave Observatory

- 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

https://pegasus.isi.edu
Two Workflows

Monte Carlo simulations and the main processing pipeline.

- Workflows execute across Open Science Grid (OSG) & European Grid Infrastructure (EGI)
- Rucio for data management
- MongoDB instance to track science runs and data products.

<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>4000</td>
<td>0</td>
<td>0</td>
<td>4000</td>
<td>267</td>
<td>4267</td>
</tr>
<tr>
<td>Jobs</td>
<td>4484</td>
<td>0</td>
<td>0</td>
<td>4484</td>
<td>267</td>
<td>4751</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: 5 hrs, 2 mins
Cumulative job wall time: 136 days, 9 hrs
Cumulative job wall time as seen from submit side: 141 days, 16 hrs
Cumulative job badput wall time: 1 day, 2 hrs
Cumulative job badput wall time as seen from submit side: 4 days, 20 hrs

Main processing pipeline is being developed for XENONnT - data taking will start at the end of 2019. Workflow in development:

https://pegasus.isi.edu
Basic Concepts...
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
Portable Description

Users do not worry about low level execution details

Directed-acyclic graphs

Input Workflow Specification YAML formatted

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

Cleanup Job
Removes unused data

Registration Job
Registers the workflow output data

https://pegasus.isi.edu
Pegasus also provides tools to generate the Abstract Workflow

```python
#!/usr/bin/env python3
import os
import logging
from pathlib import Path
from argparse import ArgumentParser

logging.basicConfig(level=logging.DEBUG)

# --- Import Pegasus API ---------------------
from pegasus.api import *

# --- Create Abstract Workflow ------------
wf = Workflow(pipeline=True)

webpage = file('pegasus.html')

# --- Create Parent Job -------------------
curl_job = {
    'id': 'curl',
    'addArgigor': 'a',
    'webpage': 'http://pegasus.isi.edu/',
    'addOutputs': webpage, 'stage_out=True, register_replica=True'
}

count = File('count.txt')

# --- Create Dependent Job ---------------
wc_job = {
    'id': 'wc',
    'addOutput': webpage,
    'setStdout': (count, stage_out=True, register_replica=True)
}

# --- Add Jobs to Abstract Workflow ------
wf.add_jobs(curl_job, wc_job)

# --- Add Control Flow dependency -------
wf.add_dependency(wc_job, parent=curl_job)

# --- Write out the Abstract Workflow ----
wf.write()
```

---

```yaml
- pegasus:
  name: pipeline
  jobs:
    - type: job
      name: curl
      id: 100000001
      arguments:
        - pipe
        - pegasus.html
        - http://pegasus.isi.edu
      output:
        stageOut: false
        registerReplica: false
    - type: job
      name: wc
      id: 100000002
      stdout: count.txt
      arguments:
        - -1
        - pegasus.html
  uses:
    - ifn: pegasus.html
      type: output
      stageOut: true
      registerReplica: true
    - ifn: pegasus.html
      type: input
    - id: 100000001
      children:
        - 100000002
```

---

https://pegasus.isi.edu
System Architecture

Interfaces
- Python
- Jupyter
- Hubzero
- CyVerse
- WINGS

Other workflow composition tools

Pegasus WMS
- Catalogs
  - Transformation
  - Replica
  - Sites

Mapper
Engine
Scheduler

Monitoring & Provenance
Workflow DB

Loggs
Submit Host

Clouds
- OpenStack
- Amazon EC2 and AWS Batch
- Google Cloud
- RackSpace
- Chameleon
- Amazon S3
- OpenStack
- Google Cloud Storage

Distributed Resources
- HTCondor / GRAM
  - PBS
  - SLURM
  - LSF
  - SGE

Compute
- Containers
  - GridFTP
  - HTTP
  - FTP
  - SRM
  - IRODS
  - SCP

Middleware

Storage

Users
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
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.

**PEGASUS DASHBOARD**

web interface for monitoring and debugging workflows

---

**Real-time Monitoring**

**Reporting**

**Debugging**

**Troubleshooting**

**RESTful API**

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 1 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>5</td>
<td>0</td>
<td>5</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
Pegasus-Statistics
or
Used for Debugging
Pegasus-Analyzer

https://pegasus.isi.edu
Understanding Pegasus Features...
Data Staging Configurations

**HTCondor I/O** (HTCondor pools, OSG, ...)
- Worker nodes do not share a file system
- Data is pulled from / pushed to the submit host via HTCondor file transfers
- Staging site is the submit host

**Non-shared File System** (clouds, OSG, ...)
- Worker nodes do not share a file system
- Data is pulled / pushed from a staging site, possibly not co-located with the computation

**Shared File System** (HPC sites, XSEDE, Campus clusters, ...)
- I/O is directly against the shared file system
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)

HTTP
SCP
GridFTP
Globus
Online
iRods
Amazon S3
Google Storage
SRM
FDT
Stashcp
Rucio
cp
ln -s

https://pegasus.isi.edu
Workflow wrapped as an MPI job
Allows sub-graphs of a Pegasus workflow to be submitted as monolithic jobs to remote resources
Modern IT systems are not perfect - errors creep in.

At modern “Big Data” sizes we are starting to see checksums breaking down.

Plus there is the threat of intentional changes: malicious attackers, insider threats, etc.

User Perception: “Am I not already protected? I have heard about TCP checksums, encrypted transfers, checksum validation, RAID and erasure coding – is that not enough?”
Pegasus performs integrity checksums on input files right before a job starts on the remote node.

- For raw inputs, checksums specified in the input replica catalog along with file locations
- All intermediate and output files checksums are generated and tracked within the system.
- Support for sha256 checksums

Job failure is triggered if checksums fail
Pegasus Container Support

Users can refer to **containers** in the **Transformation Catalog** with their executable preinstalled

Users can **refer** to a **container** they want to **use** – **Pegasus stages** their executables and containers to the node

- Useful if you want to use a site recommended/standard container image.
- Users are using generic image with executable staging.

**Future Plans**

- Users can **specify an image buildfile** for their jobs.
- **Pegasus will build the Docker image as separate jobs in the executable workflow, export them as a tar file and ship them around**
Data Management for Containers

Containers are data too!

Pegasus treats containers as input data dependency

- Staged to compute node if not present
- Docker or Singularity Hub URL’s
- Docker Image exported as a TAR file and available at a server, just like any other input dataset

Scaling up for larger workflows

- The image is pulled down as a tar file as part of data stage-in jobs in the workflow
- The exported tar file is then shipped with the workflow and made available to the jobs
- Pricing considerations. You are now charged if you exceed a certain rate of pulls from Hubs

Other Optimizations

- Symlink against existing images on shared filesystem such as CVMFS
- The exported tar file is then shipped with the workflow and made available to the jobs

https://pegasus.isi.edu
Performance.
Why not improve it?

Clustered Job
Groups small jobs together to improve performance

Task
Small granularity
Pegasus also handles large-scale workflows

When abstract workflow with only compute jobs is encountered

Recursion ends

https://pegasus.isi.edu
Data Reuse prune jobs if output data already exists

Jobs which output data is already available are pruned from the DAG.

data already available

workflow reduction

data also available

data reuse

data reuse

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
Job Submissions

**Both Glite and BOSCO build on HTCondor BLAHP**

**Current supported schedulers:**
- SLURM
- SGE
- PBS
- MOAB

### LOCAL

**Submit Machine**
Personal HTCondor

**Local Campus Cluster accessible via Submit Machine**
HTCondor via BLAHP

### REMOTE

**BOSCO + SSH**
Each node in executable workflow submitted via SSH connection to remote cluster

**BOSCO based Glideins**
SSH based submission of glideins

**PyGlidein**
IceCube glidein service

**OSG using glideinWMS**
Infrastructure provisioned glideins

**CREAMCE**
Uses CondorG

**Globus GRAM**
Uses CondorG

https://pegasus.isi.edu
Credentials Management

- **Credentials required for two purposes**
  - Job Submission
  - Data transfers to *stage-in* input and *stage-out* generated outputs when a job executes

- **Specifying Credentials**
  - Users can specify credentials in a *generic credentials file* on submit host
  - Associate credentials with sites in site catalog

- **Approach**
  - Planner will *automatically* associate the *required credentials* with each job
  - The credentials are *transferred* along with the job
  - Usually available *only for the duration* of the job execution

- **Supported Credentials**
  - X.509 grid proxies
  - Amazon AWS S3 keys,
  - Google Cloud Platform OAuth token (.boto file),
  - iRods password
  - SSH keys
  - Web Dav
Running **Pegasus** workflows with **Jupyter**

https://pegasus.isi.edu
• New and fresh Python3 API to compose, submit and monitor workflows, and configure catalogs
• New Catalog Formats
• Python 3 Support
  • All Pegasus tools are Python 3 compliant
  • Python PIP packages for workflow composition and monitoring
• Zero configuration required to submit to local HTCondor pool.
• Data Management Improvements
  • New output replica catalog that registers outputs including file metadata such as size and checksums
  • Improved support for hierarchical workflows
• Reworked Documentation and Tutorial
  https://pegasus.isi.edu/documentation/
Get Started

- **Pegasus Website**
  https://pegasus.isi.edu

- **Users Mailing List**
  pegasus-users@isi.edu

- **Support**
  pegasus-support@isi.edu

- **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*