Pegasus WMS: Features For Pipelines Across Diverse Resources

Karan Vahi, Ewa Deelman, Gideon Juve, Mats Rynge, Rajiv Mayani, Rafael Ferreira da Silva
University of Southern California / Information Sciences Institute

Overview

- Pegasus is a system for mapping and executing abstract application workflows over a range of execution environments.
- The same abstract workflow can, at different times, be mapped different execution environments such as XSEDE, OSG, commercial and academic clouds, campus grids, and clusters.
- Pegasus can easily scale both the size of the workflow, and the resources that the workflow is distributed over. Pegasus runs workflows ranging from just a few computational tasks up to 1 million.
- Pegasus Workflow Management System (WMS) consists of three main components: the Pegasus Mapper, HTCondor DAGMan, and the HTCondor Schedd.

HTCondor
High Throughput Computing

Workflow Design and Mapping

```
#!/usr/bin/env python

from Pegasus.DAX3 import *
import sys
import os

# Create a abstract dag
dax = ADAG("hello_world")

# Add the hello job
hello = Job(namespace="hello_world",
            name="hello", version="1.0")
b = File("f.b")
hello.uses(a, link=Link.INPUT)
hello.uses(b, link=Link.OUTPUT)
dax.addJob(hello)

# Add the world job (depends on the hello job)
world = Job(namespace="hello_world",
            name="world", version="1.0")
c = File("f.c")
world.uses(b, link=Link.INPUT)
world.uses(c, link=Link.OUTPUT)
dax.addJob(world)

# Add control-flow dependencies
dax.addDependency(Dependency(parent=hello,
                             child=world))

# Write the DAX to stdout
dax.writeXML(sys.stdout)
```

DAX Generator API

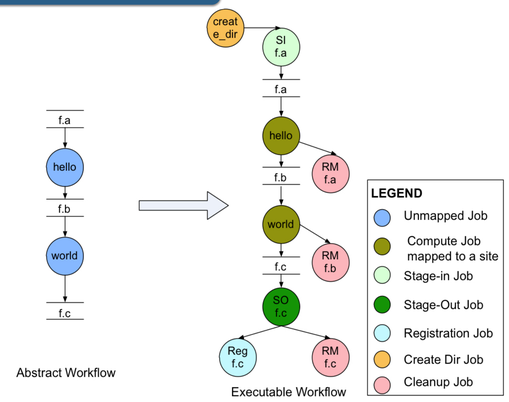
Easy to use APIs in Python, Java and Perl to generate an abstract workflow describing the users computation.

Above is a simple two node hello world example.

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- generator: python -->
<adag xmlns="http://pegasus.isi.edu/schema/DAX"
      version="3.4" name="hello_world">

    <!-- describe the jobs making
         up the hello world pipeline -->
    <job id="ID0000001" namespace="hello_world"
                 name="hello" version="1.0">

        <uses name="f.b" link="output"/>
        <uses name="f.a" link="input"/>
    </job>

    <job id="ID0000002" namespace="hello_world"
                 name="world" version="1.0">

        <uses name="f.b" link="input"/>
        <uses name="f.c" link="output"/>
    </job>

    <!-- describe the edges in the DAG -->
    <child ref="ID0000002">
        <parent ref="ID0000001"/>
    </child>
</adag>
```

Abstract Workflow (DAX)

The abstract workflow rendered as XML. It only captures the computations the user wants to do and is devoid of any physical paths. Input and output files are identified by logical identifiers. This representation is portable between different execution environments.

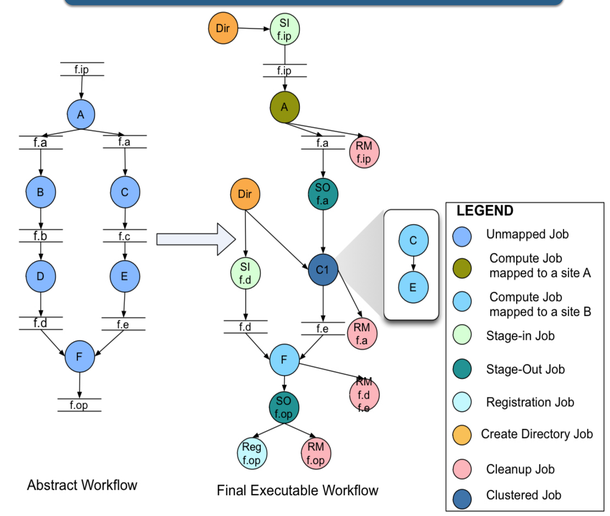


Abstract to Executable Workflow (Condor DAG) Mapping

The DAX is passed to the Pegasus Mapper and it generates a **HTCondor DAGMan** workflow that can be run on actual resource.

The above example highlights addition of ***data movement nodes*** to staging in the input data and stage out the output data; addition of ***data cleanup nodes*** to remove data that is no longer required; and ***registration nodes*** to catalog output data locations for future discovery.

Data Reuse Example



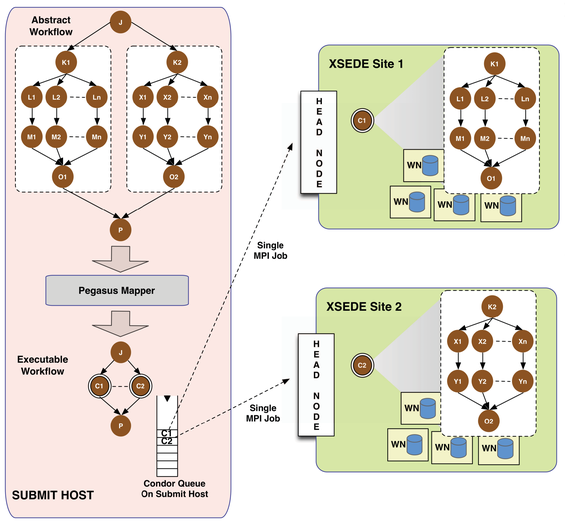
Additional Capabilities Highlighted

Data Reuse: Jobs B and D are removed from the workflow as file f.d already exists. The f.d is staged in , instead of regenerating it by executing jobs B and D.

Job Clustering: Jobs C and E are clustered together into a single clustered job.

Cross Site Run: Single Workflow can be executed on multiple sites, with Pegasus taking care of the data movement between the sites.

Pegasus Workflows with PMC on XSEDE



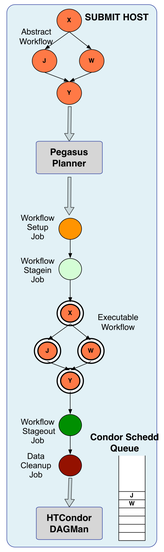
XSEDE Challenges

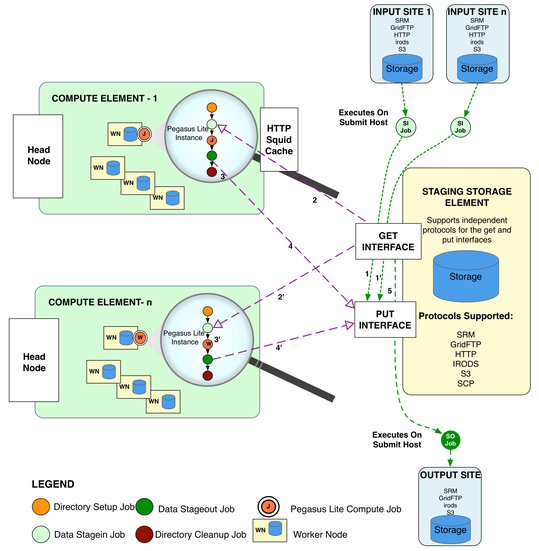
- HPC resources with remote job submission via GRAM, with strict queue limits per user (=~50)
- LIGO workflows have a large number of tasks that cannot be all submitted through remote job submission interface

Solution: The workflow is partitioned into independent sub graphs, which are submitted as self-contained Pegasus MPI Cluster (PMC) jobs to the remote sites.

A PMC job is expressed as a DAG and PMC uses the master-worker paradigm to farm out individual tasks to worker nodes. PMC acts a scheduler and considers core and memory requirements of the tasks when making scheduling decisions. PMC can be easier to setup than pilot jobs / glideins as no special networking is required. PMC relies on standard MPI constructs.

Pegasus Workflows on OSG and Virgo Resources





Data Flow For a Workflow with Pegasus

1. Stagein Jobs transfer input data for the workflow to the staging site
2. Pegasus Lite wrapped jobs , when they start on compute worker nodes, pull in the input data from staging site
3. The compute job executes on a local directory on the worker node.
4. The PegasusLite wrapper pushes the output data from the worker node back to the staging site
5. The Stageout Jobs transfer the relevant output data out to the output site from staging site

Pegasus Data Staging Configurations

Non Shared Filesystem with Staging Site : Data is staged by Pegasus Lite at Runtime from an external staging site. Popular on Virgo and OSG resources with SRM as data staging server.

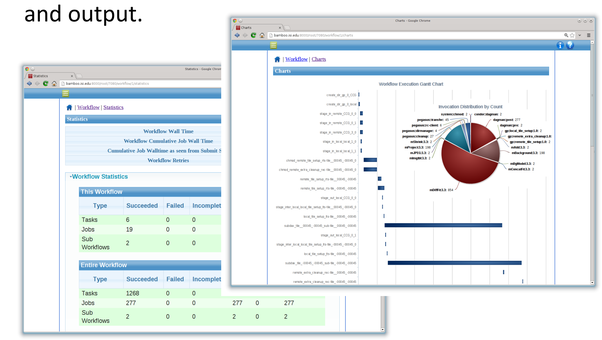
CondorIO Data is staged using Condor File Transfers from submit node. Popular on OSG and Amazon EC2.

Shared Filesystem (Head Node and the worker nodes of execution sites share a filesystem). Popular on XSEDE and clusters.

Monitoring and Debugging

At runtime, a database is populated with workflow and task runtime provenance, including which software was used and with what parameters, execution environment, runtime statistics and exit status.

Pegasus comes with command line monitoring and debugging tools. A web dashboard now allows users to monitor their running workflows and check jobs status and output.



Acknowledgments:

- Pegasus WMS is funded by the National Science Foundation OCI SDCI program grant #1148515.
- HTCondor : Miron Livny, Kent Wenger, University of Wisconsin Madison

http://pegasus.isi.edu