Using Pegasus 3.0 for data-based workflows on the OSG

Mats Rynge
rynge@isi.edu

USC Information Sciences Institute
Pegasus: Planning for Execution in Grids

- **Abstract Workflows** - Pegasus input workflow description
  - Workflow “high-level language”
  - Only identifies the computation, devoid of resource descriptions, devoid of data locations

- **Pegasus**
  - 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
  - Provides runtime provenance
How to generate a DAX

• Use the Pegasus Java, Perl, Python APIs

• Use Wings for semantically rich workflow composition ([http://www.isi.edu/ikcap/wings/](http://www.isi.edu/ikcap/wings/))

• Write the XML directly

```xml
<!-- part 1: list of all files used (may be empty) -->
<filename file="f.input" link="input"/>
<filename file="f.intermediate" link="input"/>
<filename file="f.output" link="output"/>

<!-- part 2: definition of all jobs (at least one) -->
<job id="ID000001" namespace="pegasus" name="preprocess" version="1.0">
  <argument>-a top -T 6 -i <filename file="f.input"/> -o <filename file="f.intermediate"/>
</argument>
  <uses file="f.input" link="input" dontRegister="false" dontTransfer="false"/>
  <uses file="f.intermediate" link="output" dontRegister="true" dontTransfer="false"/>
</job>
```
Basic Workflow Mapping

• Select where to run the computations
  • Change task nodes into nodes with executable descriptions
    • Execution location
    • Environment variables initializes

• Select which data to access
  • Add stage-in nodes to move data to computations
  • Add stage-out nodes to transfer data out of remote sites to storage
  • Add data transfer nodes between computation nodes that execute on different resources
Additional Mapping Elements

- Add data cleanup nodes to remove data from remote sites when no longer needed
  - reduces workflow data footprint
- Cluster compute nodes in small computational granularity applications
- Add nodes that register the newly-created data products
- Provide provenance capture steps
  - Information about source of data, executables invoked, environment variables, parameters, machines used, performance
- Scale matters - today we can handle:
  - 1 million tasks in the workflow instance (SCEC)
  - 10TB input data (LIGO)
Original workflow: 15 compute nodes devoid of resource assignment
**Original workflow:** 15 compute nodes devoid of resource assignment

Assume the results of these computations are already available
Original workflow: 15 compute nodes devoid of resource assignment

Resulting workflow mapped onto 3 Grid sites:

- **13 data stage-in nodes**
- **11 compute nodes** (4 reduced based on available intermediate data)
- **8 inter-site data transfers**
- **14 data stage-out nodes to long-term storage**
- **14 data registration nodes** (data cataloging)
Catalogs used for discovery

- To execute in a distributed environment Pegasus needs to discover
  - **Data** (the input data that is required by the workflows)
    - Replica catalog, data registry, db, dax
  - **Executables** (application executables already installed or can that be dynamically staged)
    - Transformation catalog, dax
  - **Site Layout** (site services and environment)
    - Site catalog
Discovery of Data

• Replica Catalog stores mappings between logical files and their target locations.

• Interfaces with a variety of replica catalogs
  • File based Replica Catalog
    • useful for small datasets
    • cannot be shared across users
  • Database based Replica Catalog
    • useful for medium sized datasets.
    • can be used across users
Discovery of Site Layout

- Pegasus queries a site catalog to discover site layout
  - Job submission points for different types of schedulers
  - Data transfer servers
  - Local Replica Catalogs where data residing in that site has to be catalogued
  - Site Wide Profiles like environment variables
  - Work and storage directories

The pegasus-sc-client can pull the site information from ReSS or OSGMM
DATA FLOW TO COMPUTE JOBS ON THE WORKER NODES RELYING ON A SHARED FILESYSTEM

COMPUTE AND STAGING SITE ARE SAME
Optimizations during Mapping

• Node clustering for fine-grained computations
  • Can obtain significant performance benefits for some applications (in Montage ~80%, SCEC ~50%)

• Data reuse in case intermediate data products are available
  • Performance and reliability advantages—workflow-level checkpointing

• Data cleanup nodes can reduce workflow data footprint
  • by ~50% for Montage, applications such as LIGO need restructuring
Job clustering

Vertical clustering

Level-based clustering

Arbitrary clustering

Useful for small granularity jobs
Abstract Workflow

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

Delete Job D and Job B
Data Cleanup

Adding cleanup nodes to the workflow

1.25GB versus 4.5 GB

Remote Storage used over time. 1 degree cleanup:fork. node:bundle=1
- with cleanup nodes
- without cleanup node
LIGO Workflows

Full workflow:
185,000 nodes
466,000 edges
10 TB of input data
1 TB of output data.

166 nodes

Remote Storage used over time

26% improvement

56% improvement
Job Priorities – Overlapping Data Staging and Computations

• Pegasus assigns default priorities to jobs (new feature in 3.0)
• Compute jobs
  • Based on what level the job is in the workflow (10, 20, ...)
  • Useful when running multiple workflows
• Auxiliary jobs
  • Create dir – 800
  • Stage in – 700
  • Stage out – 900
  • Cleanup – 1000

Jobs belonging to the same workflow can run in different universes. For example: compute jobs in “grid” and staging jobs in “local”
Pegasus 3.1 Upcoming Features

• Advanced transfer features with Storage Servers
  • Allows to share intermediate advanced storage infrastructure with several remote sites
  • No need for shared file system on local site
  • Can be enabled or disabled based on compute site as well as file level.
• Define metadata in DAX and populate automatically to a given metadata server
• Notification hooks on tasks, DAX, DAGs events (maybe!)
DATA FLOW TO COMPUTE JOBS ON THE WORKER NODES RELYING ON A SHARED FILESYSTEM

COMPUTE AND STAGING SITE ARE SAME
DATA FLOW TO COMPUTE JOBS ON THE WORKER NODES AND NO SHARED FILESYSTEM

COMPUTE AND STAGING SITE ARE DIFFERENT
Pegasus: http://pegasus.isi.edu/

Email: pegasus@isi.edu

QUESTIONS?