Pegasus- Advanced Features and Optimizations

Griphyn-LIGO Meeting, Caltech
July 20th, 2006

Karan Vahi, Ewa Deelman, Gaurang Mehta,
Center for Grid Technologies
USC Information Sciences Institute
vahi, deelman,gmehta@isi.edu
Advanced Features Outline

- Deferred Planning
- Job Clustering
- Transfer Configurations
- Transfer of Executables
- Replica Selection
- Running in different GRID setups
Simple scheduling: random or round robin using well-defined scheduling interfaces.
Deferred Planning through Partitioning

Partitioning techniques implemented
- Breadth First
- Label based (User specifies in the DAX what his partitions are)
- Node by Node (Each Node is a separated partition)
Label Based Partitioning(1)

- The partitions are explicitly tagged in the DAX by the user.
  - Tagging is done by associating VDS profiles with the jobs.
  - Jobs with the same profile value are considered to belong to the same partition.
  - Profiles can either be added in DAX generator or in the VDL.

- Which VDS profile key to use for partitioning?
  > You can specify any key to be used.
  > Set the property vds.label.key
Label Based Partitioning (2)

IN THE DAX:
<adag>

...  
<job id="ID000004" namespace="vahi" name="analyze" version="1.0" level="1">
  <argument>-a bottom -T60 -i <filename file="vahi.f.c1"/> -o <filename file="vahi.f.d"/></argument>
  <profile namespace="vds" key="ligo_label">p1</profile>
  <uses file="vahi.f.c1" link="input" dontRegister="false" dontTransfer="false"/>
  <uses file="vahi.f.c2" link="input" dontRegister="false" dontTransfer="false"/>
  <uses file="vahi.f.d" link="output" dontRegister="false" dontTransfer="false"/>
</job>
...

</adag>

PROPERTY FILE:
vds.label.key = ligo_label

- The above states that the VDS profiles with key as ligo_label are to be used for designating partitions.
- Each job with the same value for VDS profile key ligo_label appears in the same partition.
Mega DAG is created by Pegasus and then submitted to DAGMan.

Pegasus(X): Pegasus generated the concrete workflow and the submit files for Partition X -- Su(X).

DAGMan(Su(X)): DAGMan executes the concrete workflow for X.
Partitioned Workflow Processing

- Create workflow partitions
  - partition the abstract workflow into smaller workflows using partitiondax.
  - create the xml partition graph (pdax) that lists out the dependencies between partitions.

- Create the MegaDAG (creates the dagman submit files)
  - transform the xml partition graph to it’s corresponding condor representation.

- Submit the MegaDAG
  - Each job invokes Pegasus on a partition and then submits the plan generated back to condor.
Job Clustering (1)

- 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 minutes to execute

Or $VDS_HOME/doc/userguide/VDSUG_PegasusJobClustering.xml
• **Horizontal Clustering**
  - Jobs on the same level are clustered into larger jobs
  - Clustering parameters can be configured by associating profiles in Transformation Catalog or Site Catalog.

• **Vertical Clustering (Soon)**

• **The clustered job can be run on the remote site**
  - Sequentially using VDS tool seqexec.
  - In Parallel using using VDS MPI wrapper mpiexec
Planning & Scheduling Granularity

- **Partitioning**
  - Allows to set the granularity of planning ahead

- **Node aggregation**
  - Allows to combine nodes in the workflow and schedule them as one unit (minimizes the scheduling overheads)
  - May reduce the overheads of making scheduling and planning decisions

- **Related but separate concepts**
  - **Small jobs**
    > High-level of node aggregation
    > Large partitions
  - **Very dynamic system**
    > Small partitions
Transfer Configurations

- Variety of transfer clients may be used
  - Set `vds.transfer.*.implementation` property
  - Support for clients like
    - RFT
    - Stork
    - T2 (VDS client that retries in case of failures)
    - Transfer (VDS client wrapper around g-u-c)
    - SRM (preliminary support)

- Variety of refinement strategies maybe used for adding transfer nodes
  - Set `vds.transfer.refiner` property.

- Varying third party transfer settings
  - Set `vds.transfer.*.thirdparty.sites`
  - Allows you to specify for which compute sites you want to use for third party party staging.

  Explained in more detail at `$VDS_HOME/doc/properties.pdf`
Transfer Throttling

- **Large Sized Workflows** result in a large number of transfer jobs being executed at once. Results in
  - Grid FTP server overload (connection refused errors etc)
  - May result in a high load on the head node if transfers are not configured for being executed as third party transfers

- **Need to throttle transfers**
  - Set vds.transfer.refiner property.
  - Allows you to create chained transfer jobs or bundles of transfer jobs
Transfer Throttling by Chaining

Original Workflow

Workflow After Adding the Stage-In Nodes

Legends:
- Stage-In Transfer Node
- Compute Job scheduled at same site

Explainined in more detail at $VDS_HOME/doc/properties.pdf
Transfer Throttling by Bundling

Original Workflow

Workflow After Adding the Stage-In Nodes

Explained in more detail at $VDS_HOME/doc/properties.pdf
Transfer of Executables

- Allows the user to dynamically deploy scientific code on remote sites
- Makes for easier debugging of scientific code.
- The executables are transferred as part of the workflow
- Currently, only statically compiled executables can be transferred
- Selection of what executable to transfer
  - Set vds.transformation.selector property.


Also explained in the properties file at $VDS_HOME/doc/properties.pdf
Replica Selection

- **Default replica selection**
  - Always prefer data present at the compute site, else select randomly a replica

- **Restricted Replica Selection**
  - Can specify preferred sites from which to stage in data per compute site.
  - Can specify sites to ignore for staging in data per compute site.

- **Properties to Set (\* in name replaced by site name. \* means all sites)**
  - vds.replica.selector
  - vds.replica.\*ignore.stagein.sites
  - vds.replica.\*ignore.stagein.sites

Explained in more detail at $VDS_HOME/doc/properties.pdf
Running in different grid setups

● Need to specify vds namespace profile keys with the sites in the site catalog.

● Submitting directly to condor pool
  - The submit host is a part of a local condor pool
  - Bypasses CondorG submissions avoiding Condor/GRAM delays.

● Using Condor GlideIn
  - User glides in nodes from a remote grid site to his local pool
  - Condor is deployed dynamically on glided in nodes for e.g. you glide in nodes from the teragrid site running PBS.
  - Only have to wait in the remote queue once when gliding in nodes.

Or $VDS_HOME/doc/userguide/VDSUG_RunningPegasus.xml
Condor GlideIn

Cluster on a public network

Submit Node
(Collectors, Master, Negotiator, Schedd)

Head Node
(GT4 PBS GRAM)

Connect to Collector

PBS runs Glidein request

Execute Jobs

Glidein request
For further information

- **VDS and Pegasus:**
  - [http://vds.isi.edu](http://vds.isi.edu)
  - [http://pegasus.isi.edu](http://pegasus.isi.edu)

- **Mailing Lists**
  - [vds-support@griphyn.org](mailto:vds-support@griphyn.org)
  - [vds-discuss@griphyn.org](mailto:vds-discuss@griphyn.org)

- **Workflow Management research group in GGF:**
  - [www.isi.edu/~deelman/wfm-rg](http://www.isi.edu/~deelman/wfm-rg)

- **Workshops**
  - Works06 ([http://www.isi.edu/works06/](http://www.isi.edu/works06/)) in conjunction with HPDC 2006.
Pegasus - Further Reading

- VDS Documents in VDS distribution in $VDS_HOME/doc directory
  - configuration via properties $VDS_HOME/doc/properties.pdf
  - Userguide in $VDS_HOME/doc/userguide directory

- On the web (often lags latest release)
  - http://vds.uchicago.edu/twiki/bin/view/VDSWeb/VDSDocs
Pegasus Papers

- Papers on Pegasus (more at http://pegasus.isi.edu)
  - "Artificial Intelligence and Grids: Workflow Planning and Beyond," Yolanda Gil, Ewa Deelman, Jim Blythe, Carl Kesselman, and Hongsuda Tangmurarunkit. IEEE Intelligent Systems, January 2004