Pegasus – Enhancing LIGO DAGMan Experience

Karan Vahi

Science Automation Technologies Group
USC Information Sciences Institute
LIGO Gravitational Wave Detection

- LIGO recently announced first ever detection of gravitational waves.
  - Created as a result of coalescence of a pair of dense, massive black holes.
  - Confirms major prediction of Einstein Theory of Relativity

- Detection Event
  - Detected by both of the operational Advanced LIGO detectors (4km long L shaped interferometers)
  - Event occurred at September 14, 2015 at 5:51 a.m. Eastern Daylight Time


Image Credits: 0.2 Second before the black holes collide: SXS/LIGO
Signals of Gravitational Waves Detected: Caltech/MIT/LIGO Lab
LIGO Detection – Behind the Scenes

- A variety of complex analysis pipelines were executed.
- Some were low latency that initially alerted people to look at a specific piece of data containing the signal.
- However, to verify that signal is a valid candidate,
  - a large amount of data needs to be analyzed.
  - Statistical significance of the detection should be at 5-sigma level
- Pipelines are mainly executed on LSC Data Grid
  - Consists of approximately 11 large clusters at various LIGO institutions and affiliates
  - Each cluster has Grid middleware and HTCondor installed.
  - GridFTP used for data transfers.
- Pipelines are modeled as scientific workflows
One of the main pipelines to measure the statistical significance of data needed for discovery.

Contains 100's of thousands of jobs and accesses on order of terabytes of data.

Uses data from multiple detectors.

Exclusively managed by Pegasus WMS and an earlier version was used for the blind injection test in 2011

For the detection, the pipeline was executed on Syracuse and Albert Einstein Institute Hannover

PyCBC Paper: An improved pipeline to search for gravitational waves from compact binary coalescence. Samantha Usman, Duncan Brown et al.

PyCBC Detection GW150914: First results from the search for binary black hole coalescence with Advanced LIGO. B. P. Abbott et al.
Pegasus Workflow Management System

- **NSF funded project since 2001**
  - Developed as a collaboration between USC Information Sciences Institute and the HTCondor Team at UW Madison

- **Builds on top of HTCondor DAGMan.**

- **Abstract Workflows - Pegasus input workflow description**
  - Workflow “high-level language”
  - Only identifies the computation, devoid of resource descriptions, devoid of data locations
  - File Aware – For each task you specify the input and output files

- **Pegasus is a 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
  - Collects runtime provenance
Pegasus Deployment

Users

Interfaces

APIs

- python
- Java
- Perl
- hubzero

Other workflow composition tools: Wings

Pegasus Dashboard

Submit Host

Pegasus WMS

Mapper

Engine

Scheduler

Monitoring & Provenance

Workflow DB

Logs

Job Queue

Notifications

Distributed Resources

Clouds

Cloudware
- OpenStack, Eucalyptus, Nimbus

Compute
- Amazon EC2, Google Cloud, RackSpace, Chameleon

Storage
- Amazon S3, Google Cloud Storage, OpenStack

Campus Clusters, Local Clusters, Open Science Grid, XSEDE

Middleware

HTCondor / GRAM

Compute
- PBS
- LSF
- SGE

Storage
- GridFTP
- HTTP
- FTP
- SRM
- IRODS
- SCP

Submit Host

Distributed Resources

Clouds

Cloudware
- OpenStack, Eucalyptus, Nimbus

Compute
- Amazon EC2, Google Cloud, RackSpace, Chameleon

Storage
- Amazon S3, Google Cloud Storage, OpenStack

Campus Clusters, Local Clusters, Open Science Grid, XSEDE

Middleware

HTCondor / GRAM

Compute
- PBS
- LSF
- SGE

Storage
- GridFTP
- HTTP
- FTP
- SRM
- IRODS
- SCP
Benefits to LIGO provided by Pegasus - Expanded Computing Horizons

- No longer limited to a single execution resource
  - Non Pegasus LIGO pipelines can often only run on LIGO clusters
  - Input is replicated out of band, in a rigid directory layout.
  - Rely on the shared filesystem to access data.

- Made it possible to leverage Non LDG Computing Resources
  - Open Science Grid
    - Dynamic – Best Effort Resource with no shared filesystem available
  - Large NSF Supercomputing Clusters XSEDE
    - No HTCondor
    - Geared for Large MPI jobs, not thousands of single node jobs
    - LIGO tried to setup XSEDE cluster as a LDG site but mismatch in setup.
    - Pegasus enabled LIGO to use XSEDE without changes at LIGO or at XSEDE
  - VIRGO Resources in Europe
    - Clusters with no shared filesystem and different storage management infrastructure than LDG
    - No HTCondor

- Pegasus enables users to run workflows across different computing environments!
Data Flow for LIGO Pegasus Workflows in OSG

1. Workflow Stagein Job stages the input data for workflow from user server.
2. PegasusLite instance looks up input data on the compute node/CVMFS. If not present, stage-in data from remote data staging server.
3. PegasusLite instance stages out job output data from worker node to data staging server.
4. Workflow StageOut Job stages produced data from data staging server to LIGO Output Data Server.

LEGEND
- Directory Setup Job
- Data Stageout Job
- Pegasus Lite Compute Job
- Data Stagein Job
- Directory Cleanup Job
- Worker Node
LIGO on XSEDE

- **Problem:** Many scientific workflows are fine-grained
  - Thousands of tasks
  - Short duration
  - Serial

- Collectively, these tasks require distributed resources to finish in a reasonable time, but individually they are relatively small
  - Touch many GB or TB of data
  - Consume thousands of CPU hours

- Many large-scale compute resources are optimized for a few, large, parallel jobs, not many small, serial jobs
  - Serial tasks face long queue times due to low priority
  - Batch schedulers have low throughput

Results in poor workflow performance
Pegasus MPI Cluster

Solution: Pegasus-MPI-Cluster

- A master/worker task scheduler for running fine-grained workflows on batch systems
- Runs as an MPI job
  - Uses MPI to implement master/worker protocol
- Allows sub-graphs of a Pegasus workflow to be submitted as monolithic jobs to remote resources
Benefits to LIGO provided by Pegasus - Smart Data Management

- **Automated Discovery of Data**
  - Symlink against locally available inputs
  - Fallback to remote file servers if data not available locally
  - Support for retrieving data using various protocols

- **Automated Cleanup of Data**
  - Data that is no longer required is automatically cleaned up.
  - Reduces peak storage requirements.

- **Data Reuse**
  - If output data is already computed or exists, Pegasus automatically prunes the pipeline accordingly
  - Reduces amount of computing resources used!

- **Job Checkpoint Files**
  - Long running jobs write out checkpoint files that are managed by Pegasus
  - Can run long running jobs on sites where limits on runtime of a single job.
Reusing Data Products

Solution: Workflow Reduction

- Don’t execute jobs at runtime for which data products already exist.
- Similar to make style semantics for compiling code
File cleanup

- **Solution**
  - *Do cleanup after workflows finish*
    - Does not work as the scratch may get filled much before during execution
  - *Interleave cleanup automatically during workflow execution.*
    - Requires an analysis of the workflow to determine, when a file is no longer required
  - *Cluster the cleanup jobs by level for large workflows*
    - Too many cleanup jobs adversely affect the walltime of the workflow.
Benefits to LIGO provided by Pegasus - Performance Improvements

- **Task Clustering**
  - LIGO workflows are mix of long running and short running tasks.
  - Pegasus clusters short running tasks into larger chunks to overcome scheduling overheads.
  - LIGO used Pegasus MPI Cluster framework for running large workflows on XSEDE.
    - Sub graphs of Pegasus Workflows submitted to remote resources as single MPI job.

- **Separation of Directories**
  - Non Pegasus LIGO pipelines rely on the shared filesystem of clusters
  - Use of Pegasus allowed workflow submit directories to be moved to local filesystems
Benefits to LIGO provided by Pegasus: Monitoring and Debugging

- **Failure Recovery**
  - Automatic retry of failed jobs as a workflow is running
  - Workflows can be restarted from where they left off

- **Debug and Monitor Workflows**
  - Users need automated tools to go through the log files
  - Need to correlate data across lots of log files
  - Need to know what host a job ran on and how it was invoked

- **Pegasus Dashboard**
  - Used by LIGO users to monitor and debug workflows

- **Especially useful for LIGO users because of the size of their workflows!**
Pegasus LIGO Collaboration - Timeline

2001  Griphyn (Grid Physics Network) funded. Pegasus development started
2002  Pegasus LIGO demonstration at SC 2002 highlighting Virtual Data
2002  HPDC paper on Griphyn and LIGO focusing on Virtual Data
2003  Support for LDR – Globus RLS based LIGO data discovery service
2004  Development of replica selection strategies to optimize data access on LIGO Data Grid
2004  Long term collaboration with LIGO for running workflows on OSG
2005  Use of task clustering for performance improvements
2006  Development of cleanup algorithm to reduce peak storage requirements
2010  Hierarchal Workflows used by LIGO iHope workflows
2010  Developed pegasus-analyzer – a workflow debugging tool
2010-2011 Pegasus managed iHope workflows used for blind injection test
2012  Enabled LIGO iHope workflows to use VIRGO computing resources
2013  Introduced Pegasus Dashboard for LIGO users
2014  Enabled LIGO to leverage XSEDE for computations
2015  Pegasus managed pyCBC workflows used to verify gravitational wave detection
LIGO Pegasus – What’s Next

- **Continued use of Pegasus to detect other interesting events**

- **Support for Metadata**
  - Automatic collection of static and runtime metadata attributes
  - Accessible via Pegasus Dashboard
  - Use for smarter data reuse – identifying what are the existing relevant data sets

- **Automatic organization of files in efficient directory structure**
  - Having thousands of files in a directory degrades filesystem performance.
  - Pegasus will automatically place them in a hierarchal data organization.

- **Increased use of Open Science Grid**
  - Seamless overflow of jobs to OSG
  - Improved Data Discovery

- **Improved error debugging and analysis via dashboard**
Pegasus  
est. 2001

Automate, recover, and debug scientific computations.

Get Started

Pegasus Website  
http://pegasus.isi.edu

Users Mailing List  
pegasus-users@isi.edu

Support  
pegasus-support@isi.edu

Tutorial  
http://pegasus.isi.edu/documentation/
Pegasus  est. 2001
Automate, recover, and debug scientific computations.

Thank You

Questions?

Acknowledgements
LIGO – Duncan Brown, Stuart Anderson, Larne Pekowsky, Alex Nitz, Peter Couvares and many others.

Meet our team
- Ewa Deelman
- Karan Vahi
- Gideon Juve
- Mats Rynge
- Rajiv Mayani
- Rafael Ferreira da Silva