Workflow tools for the TeraGrid

Ewa Deelman  
deelman@isi.edu

Karan Vahi  
vahi@isi.edu

USC Information Sciences Institute  
Marina del Rey, CA
Outline

- Introduction to scientific workflows
- Introduction to Pegasus-WMS
- Hand-on exercises
  - Workflow creation
  - Environment setup
  - Local execution of workflows
  - Executing a workflow on the TeraGrid
  - Advanced features
Scientific Workflows

- Capture individual data transformation and analysis steps
- Allow to compose larger applications from individual application components
- The components can be independent or connected by some control flow/data flow dependencies.
- Provide automation
- Support easy analysis modifications
- Provide scaled up execution over several computational resources
- Foster collaborations
Types of Workflow Applications

- **Providing a service to a community** (Montage project)
  - Data and derived data products available to a broad range of users
  - A limited number of small computational requests can be handled locally
  - For large numbers of requests or large requests need to rely on shared cyberinfrastructure resources
  - On-the fly workflow generation, portable workflow definition

- **Supporting community-based analysis** (SCEC project)
  - Codes are collaboratively developed
  - Codes are “strung” together to model complex systems
  - Ability to correctly connect components, scalability

- **Processing large amounts of shared data on shared resources** (LIGO project)
  - Data captured by various instruments and cataloged in community data registries.
  - Amounts of data necessitate reaching out beyond local clusters
  - Automation, scalability and reliability
Issues Critical to Scientists

- **Reproducibility** of scientific analyses and processes is at the core of the scientific method
  - Scientific versus Engineering reproducibility
  - Workflows give us the opportunity to provide reproducibility

- Scientists consider the “capture and generation of provenance information as a critical part of the workflow-generated data”

- “Sharing workflows is an essential element of education, and acceleration of knowledge dissemination.”

NSF Workshop on the Challenges of Scientific Workflows, 2006, [www.isi.edu/nsf-workflows06](http://www.isi.edu/nsf-workflows06)

Ewa Deelman  http://pegasus.isi.edu  deelman@isi.edu
*The full moon is 0.5 deg. sq. when viewed from Earth, Full Sky is ~ 400,000 deg. sq.*
Workflow Lifecycle

**Reuse**
- Data Products
- Adapt, Modify
- Workflow and Component Libraries

**Creation**
- Workflow Template
- Populate with data
- Workflow Instance
- Data, Metadata Catalogs

**Scheduling/Execution**
- Executable Workflow
- Compute, Storage and Network Resources
- Distributed

**Mapping**
- Map to available resources
- Resource, Application Component Descriptions
Workflow Creation

- Design a workflow
  - Find the right components
  - Set the right parameters
  - Find the right data
  - Connect appropriate pieces together
  - Find the right fillers

- Support both experts and novices
Wings: Workflow Instance Generation and Selection (Yolanda Gil, USC/ISI)

Slide courtesy Yolanda Gil, USC/ISI
Workflow Creation with Wings

Workflow Template (Edited in Wings)

Workflow Instance (Automatically generated by Wings)

Used in Earthquake science and data mining domains

Slide courtesy Yolanda Gil, USC/ISI
Creating Workflows with WINGS

- Separates analysis spec from data
  - Workflow template as reusable well-defined acceptable analysis process
  - Workflow instance binds template to data for particular analyses
- Represents data collections within the workflow specification
- Ensures that the data complies with the component specifications and their constraints within the workflow
- Attaches descriptions and metadata to projected new data products
- Records data provenance (workflow instance) and pedigree (w. template)
- Submits workflow instance to Pegasus, which creates the executable workflow and oversees execution
- Supports efficient and correct creation of new workflows
  - By retrieving a template and filling in the data
  - By reusing existing data
  - By isolating scientists from complexities of distributed computing
- Framework ensures adherence to scientific methodology

Slides courtesy Yolanda Gil, USC/ISI
Bioinformatics applications

Slide Courtesy of Carole Goble
Warehouses and Databases

Taverna

Protocol

Create a gene list in Excel
Go to NCBI
Retrieve FASTA for each gene
DragonDB Blast each sequence
Copy/paste IDs into a spreadsheet
Run Repeat Masker on each sequence
Copy/paste masked sequence into Excel
Run MacVector cut each seq with EcoRI

[Mark Wilkinson, 2006 BioMOBY]
Scufl Simple Conceptual Unified Flow Language
Taverna Writing, running workflows & examining results
SOAPLAB Makes applications available as services

Slides courtesy Katy Wolstencroft
Kepler (UCSD)

- Kepler is a software application for the analysis and modeling of scientific data
- Actor Oriented Modeling
  - Each actor has input/output ports (data flows through the actors)
  - Parameters are static ports
- Data Connections
  - Unidirectional communication channels connect output to input ports
- Composite Actors
  - Wrap sub workflows
  - Arbitrary Nesting
- Directors
  - Define the **execution semantics** of workflow graph
  - executes workflow graph
  - sub-workflows may have different directors promotes reusability
... to an executable workflow (here: in KEPLER)

Slide Courtesy of Bertram Ludaescher
LIGO Inspiral Search Application

Use of high-level languages to describe workflows
Textual workflow representations

Inspiral workflow application is the work of Duncan Brown, Caltech, Scott Koranda, UW Milwaukee, and the LSC Inspiral group
Workflow Lifecycle

Data Products

Adapt, Modify

Workflow and Component Libraries

Workflow Template

Populate with data

Workflow Instance

Map to available resources

Mapping

Data, Metadata Catalogs

Resource, Application Component Descriptions

Execute

Data, Metadata, Provenance Information

Compute, Storage and Network Resources

Executable Workflow
Why mapping?

- Many workflow systems support only executable workflow composition
- Abstraction provides
  - ease of use (do not need to worry about low-level execution details)
  - portability (can use the same workflow description to run on a number of TG resources and/or across them)
  - gives opportunities for optimization and fault tolerance
    - automatically restructure the workflow
    - automatically provide fault recovery (retry, choose different resource)
Specification: Place Y = F(x) at L
Execution Environment: Distributed

- Find where x is--- {S1,S2, …}
- Find where F can be computed--- {C1,C2, …}
- Choose c and s subject to constraints (performance, space availability,…..)
- Move x from s to c
  - Move F to c
- Compute F(x) at c
- Move Y from c to L
- Register Y in data registry
- Record provenance of Y, performance of F(x) at c

Error!  x was not at s!
Error!  F(x) failed!
Error!  c  crashed!
Error!  there is not enough space at L!
Some challenges in workflow mapping

- Automated management of data
- Efficient mapping of workflow instances to resources
  - Runtime Performance
  - Data space optimizations
  - Fault tolerance (involves interfacing with the workflow execution system)
    - Recovery by replanning
    - plan “B”
- Scalability
- Providing feedback to the user
  - Feasibility, time estimates
Mapping Correctly (Pegasus-WMS)

- Select where to run the computations
  - Apply a scheduling algorithm
    - HEFT, min-min, round-robin, random
    - Schedule in a data-aware fashion (data transfers, amount of storage)
    - The quality of the scheduling depends on the quality of information
  - Transform task nodes into nodes with executable descriptions
    - Execution location
    - Environment variables initialization
    - Appropriate command-line parameters setting

- 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
  - Add nodes to create an execution directory on a remote site
Additional Mapping Elements (Pegasus-WMS)

- Cluster compute nodes in small granularity applications
- Add data cleanup nodes to remove data from remote sites when no longer needed
  - reduces workflow data footprint
- 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)
SCEC and Pegasus-WMS on the TeraGrid

Total number of jobs 884,787
Total time : 230 CPU days
Execution Environment

Globus and Condor Services for job scheduling
Globus Services for data transfer
Data Registries (Globus RLS)

Information Services:
--- information about data location
--- information about the execution sites
Challenges in Workflow Execution

- Resource provisioning
  - Which resources to provision if many possibilities?
  - How many resources to provision?
  - For how long?
- Fault Tolerance
  - How to recognize different types of failures
  - How to recover from failures?
- Efficient collaboration between the data and computation management systems
- Debugging
  - How to relate the workflow result (outcome) to workflow specification
- Executing in a number of environments
  - Grid, P2P, Web services
Grid Computing:
- Job Submission, File services
- A Graphical Grid Computing Environment or Portal

Service Based Computing:
- Deployment, discovery and communication with distributed services e.g. P2P and (GSI) Web services

GAT Interface
- Condor
- Unicore
- GridFTP
- GridLab
- SSH
- SGE
- LDR
- Other

GAP Interface
- P2PS
- JXTA
- Web Services
- UDDI
- SOAP
- Grid services

Slide Courtesy of Ian Taylor
Efficient data handling (Pegasus)

- Input data is staged dynamically
- New data products are generated during execution
- For large workflows 10,000+ files
  - Similar order of intermediate and output files
  - Total space occupied is far greater than available space—failures occur
- Solution:
  - Analyze the workflow
    - determine which data are no longer needed and when
  - Add nodes to the workflow to cleanup data along the way

Joint work with Rizos Sakellariou, Manchester University, CCGrid 2007, Scientific Programming Journal, 2007
Dynamic Cleanup with Montage Workflow

Adding cleanup nodes to the workflow

1.25GB versus 4.5 GB
Full workflow:
185,000 nodes
466,000 edges
10 TB of input data
1 TB of output data.

26% improvement
56% improvement
Interaction Between Workflow Planner and Data Placement Service for Staging Data

Joint work with Ann Chervenak, USC/ISI

Significant potential for multiple workflows
Execution Times with Default Input Sizes

Combination of prestaging data with DRS followed by workflow execution using Pegasus improves execution time approximately 10% over Pegasus performing explicit data staging.
Workflow Lifecycle

- Data Products
- Data, Metadata, Provenance information
- Adapt, Modify
- Workflow and Component Libraries
- Workflow Template
- Populate with data
- Workflow Instance
- Executable Workflow
- Map to available resources
- Compute, Storage and Network Resources
- Resource, Application Component Descriptions

Ewa Deelman
http://pegasus.isi.edu
deeleman@isi.edu
Challenges in reuse and sharing

- How to find what is already there
- How to determine the quality of what’s there
- How to invoke an existing workflow
- How to share a workflow with a colleague
- How to share a workflow with a competitor
Workflow Mapping and Execution Connected

- For each data item, we can find the executable workflow steps that produced it and other data items that contributed to those steps.
- For each workflow step, we can find its connection to the workflow instance jobs from which it was refined.

Joint work with Luc Moreau, Southampton University  e-Science 2007
Workflow Reuse within a system

Template and component libraries
Keyword and semantic search capabilities
Composite Workflows
Broad sharing: the new frontier

- MyExperiment in the UK (University of Manchester), a repository of workflows [http://www.myexperiment.org/](http://www.myexperiment.org/)

- How do you share workflows across different workflow systems?
  - How to write a workflow in Wings and execute in Taverna?

- How do you interpret results from one workflow when you are using a different workflows system (provenance-level interoperability)
  - Open provenance model [http://eprints.ecs.soton.ac.uk/14979/1/opm.pdf](http://eprints.ecs.soton.ac.uk/14979/1/opm.pdf)
Workflows do not need to be exposed to the end user
Portals, Providing high-level Interfaces

Earthworks
TeraGrid Science Gateway

EarthWorks Project (SCEC), lead by with J. Muench P. Maechling, H. Francoeur, and others

TG 2006
Scientific Workflow Systems Summary

- Scientific workflows are used in a number of disciplines: astronomy, bioinformatics, earth sciences, ecology, physics, ...
- Many workflow systems, each with their own strengths and weaknesses
  - Taverna-- good UI, built for bioinformatics applications (web service-based)
  - Kepler--good UI, support for different semantics between workflow components (streaming data)
  - Triana--good UI, support for execution in grid-based and service-based environments
  - **Pegasus-WMS = Pegasus + DAGMan**
    - deals with mapping abstract workflows to resources, issues of performance, scalability (portable across environments)
    - scalable and reliable workflow execution engine

- BPEL and associated engines—developed within the business community, often too complex for scientific workflows
  - Scientific tools often either abstract the BPEL layer or provide a hand-tailored solution
Relevant Links

- DAGMan: [www.cs.wisc.edu/condor/dagman](http://www.cs.wisc.edu/condor/dagman)
- Kepler: [http://www.kepler-project.org/](http://www.kepler-project.org/)
- Pegasus: [http://pegasus.isi.edu](http://pegasus.isi.edu)