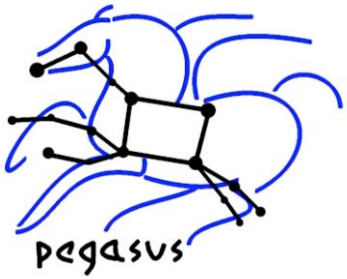
Time and Space Optimizations for Executing Scientific Workflows in Distributed Environments



Ewa Deelman

Information Sciences Institute

University of Southern California

Scientific Applications Today

- Complex
  - Involve many computational steps
  - Require many (possibly diverse resources)
- Composed of individual application components
  - Components written by different individuals
  - Components require and generate large amounts of data
  - Components written in different languages
- Reuse of individual intermediate data products
- Need to keep track of how the data was produced

Ewa Deelman
deelman@isi.edu

Execution environment

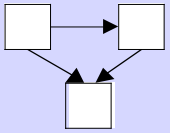
- Many resources are available
- Resources are heterogeneous and distributed in the WAN
- Access to resources is often remote
- Resources come and go because of failure or policy changes
- Data is replicated at more than one location
- Application components can be found at various locations or staged in on demand

- **Problem:** How to compose and map applications onto the environment?
  - Efficiently &Reliably
- Structure the application as a workflow
  - Define the application components, the dependencies between them
- Tie the resources together into a Grid
- Develop a mapping strategy to map from the workflow description to the Grid resources

Scientific Analysis

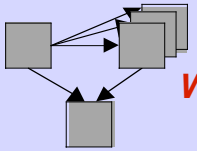
Workflow Evolution

Construct the Analysis



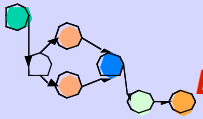
Workflow Template

Select the Input Data



Workflow Instance

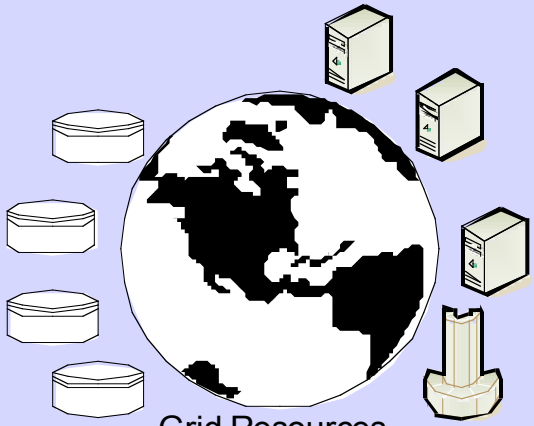
Map the Workflow onto Available Resources



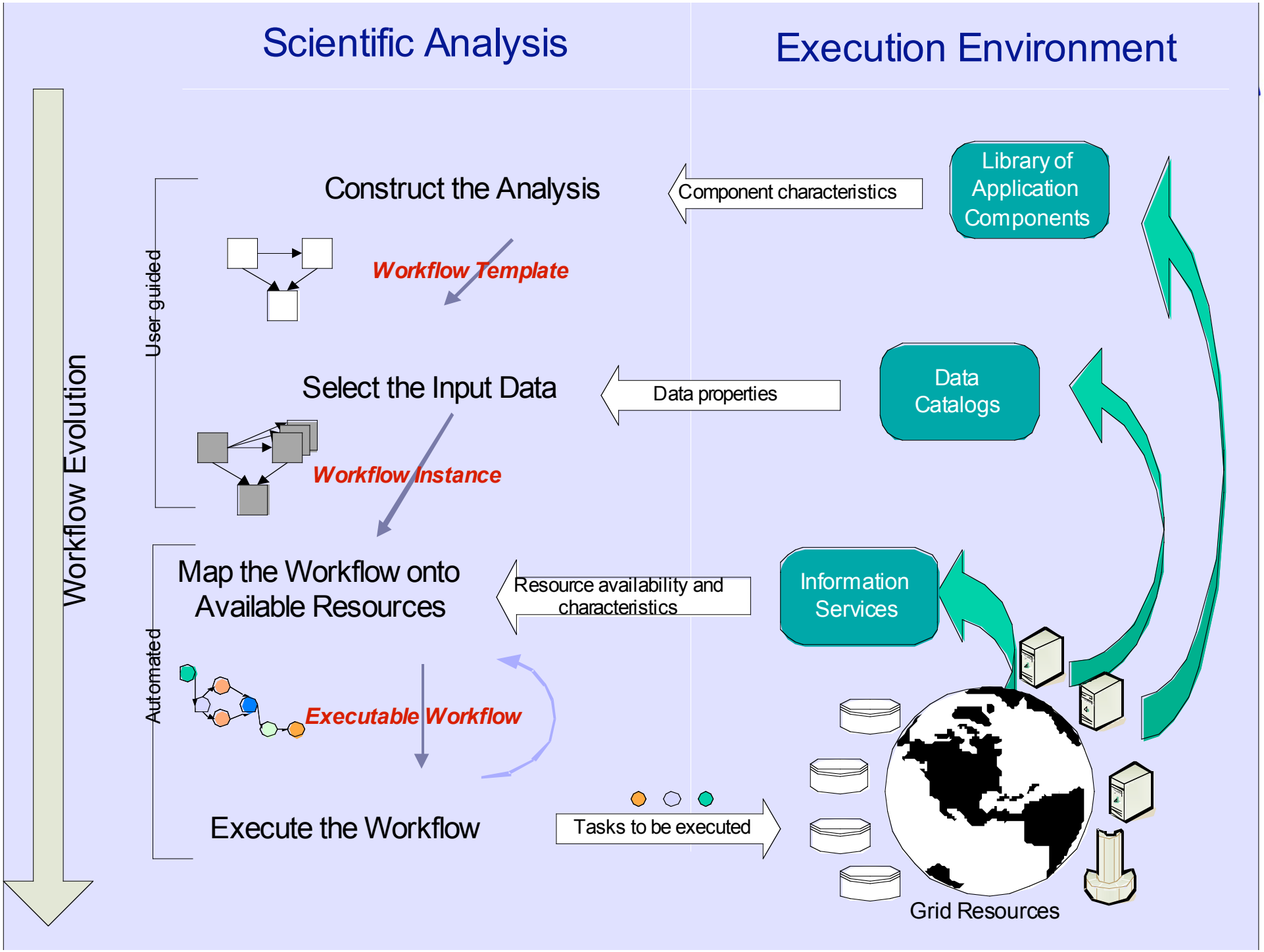
Executable Workflow

Execute the Workflow

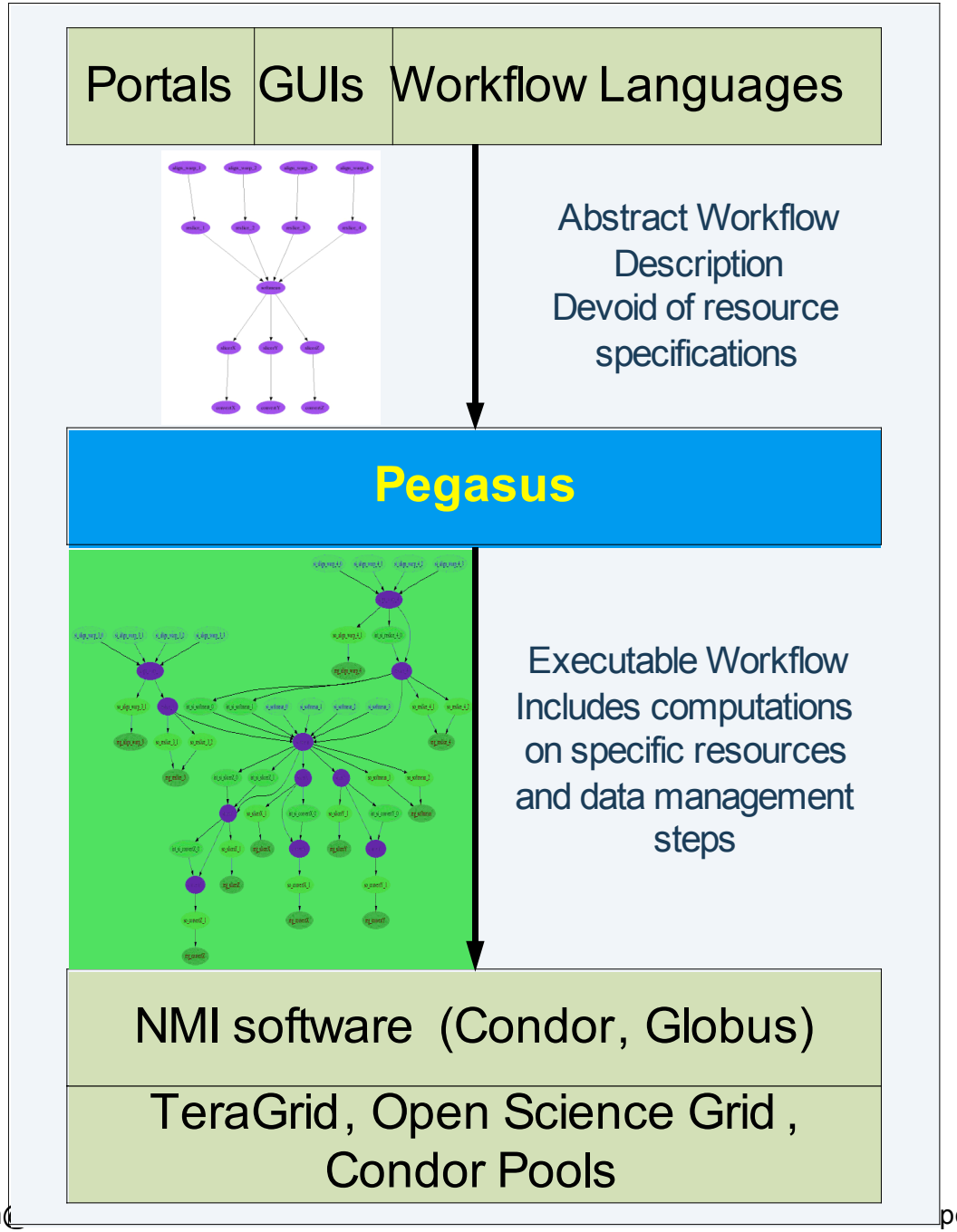
Tasks to be executed



Grid Resources


Scientific Analysis
Execution Environment
Workflow Evolution
User guided
Automated
Construct the Analysis
Workflow Template
Select the Input Data
Workflow Instance
Map the Workflow onto Available Resources
Executable Workflow
Execute the Workflow
Component characteristics
Library of Application Components
Data properties
Data Catalogs
Resource availability and characteristics
Information Services
Tasks to be executed
Grid Resources

Pegasus in Practice



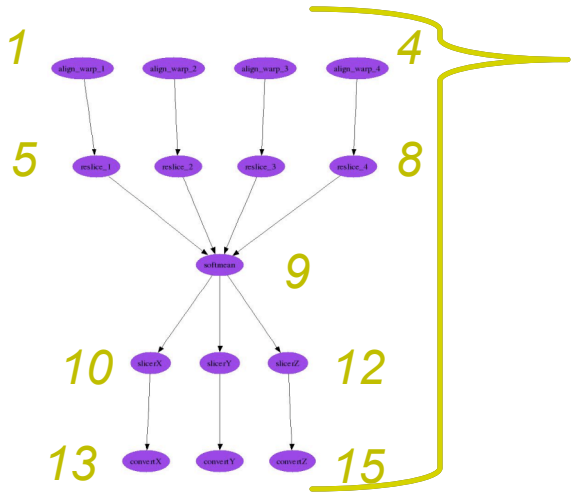
Pegasus:
Planning for Execution in Grids

- Maps from a workflow instance to an executable workflow
- Automatically locates physical locations for both workflow components and data
- Finds appropriate resources to execute the components
- Reuses existing data products where applicable
- Publishes newly derived data products
  - Provides provenance information

Information Components used by Pegasus

- Pegasus maintains interfaces to support a variety of information sources
- Information about resources
  - Globus Monitoring and Discovery Service (MDS)
    - Finds resource properties
    - Dynamic: load, queue length
    - Static: location of GridFTP server, RLS, etc
- Information about data location
  - Globus Replica Location Service
    - Locates data that may be replicated
    - Registers new data products
- Information about executables
  - Transformation Catalog

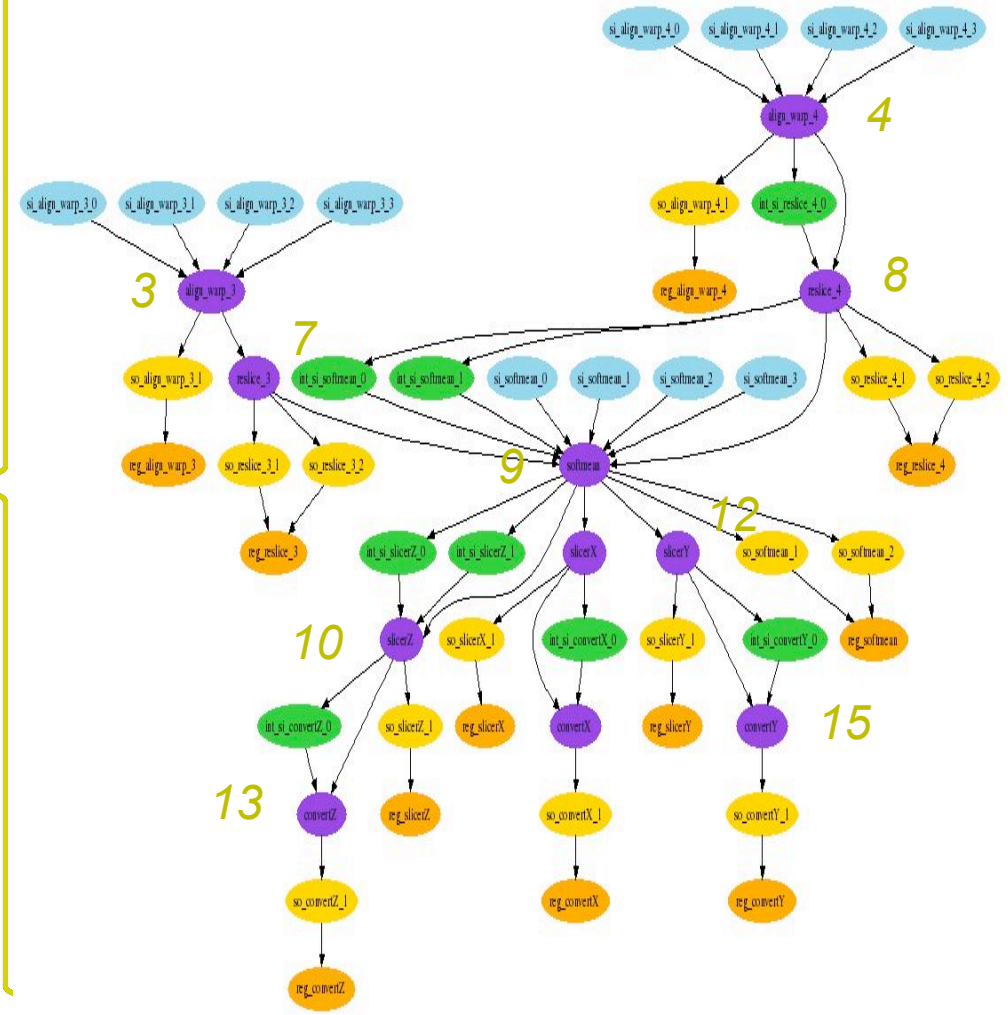
Pegasus Workflow Mapping



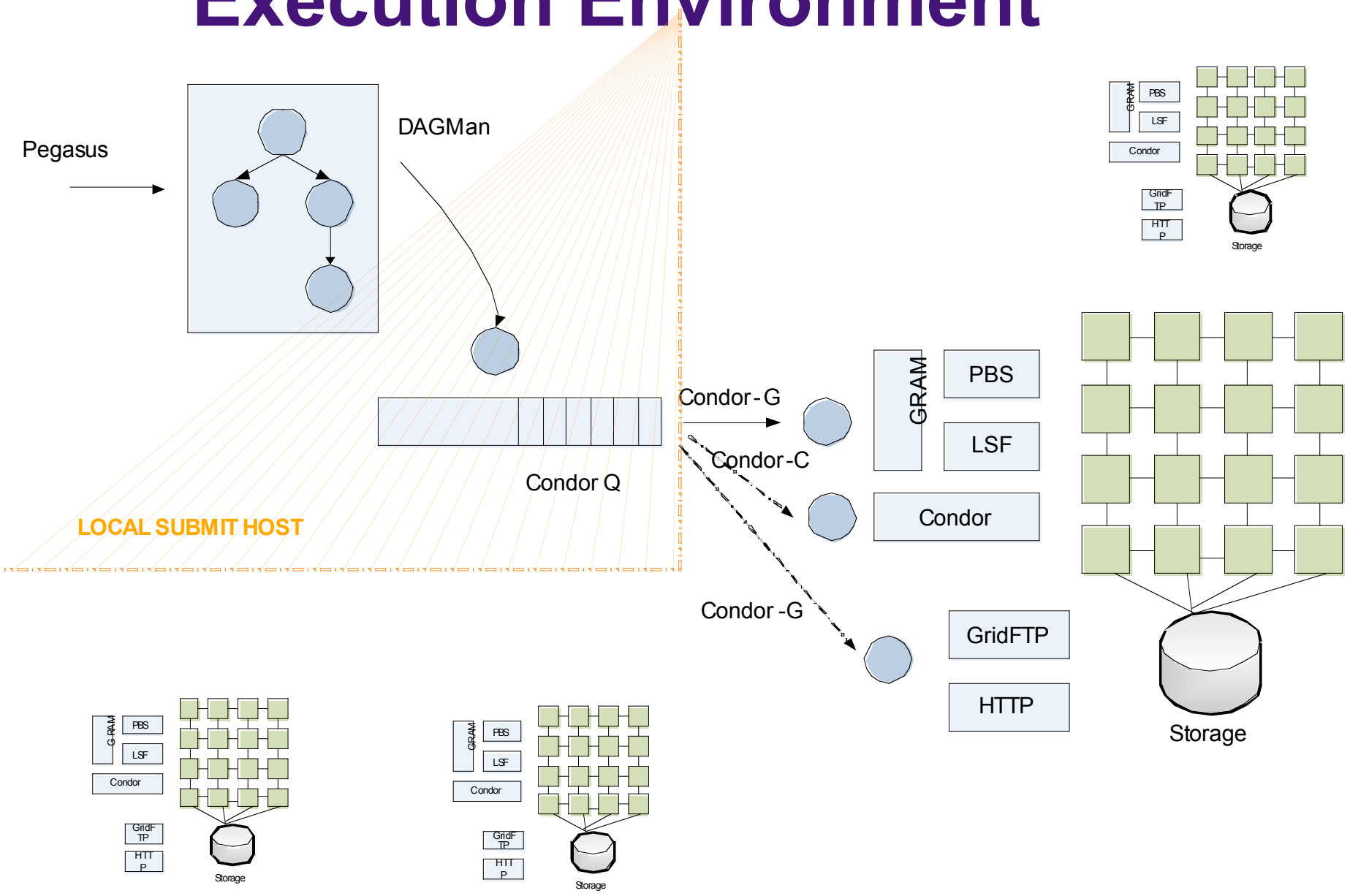
Original workflow: 15 compute nodes devoid of resource assignment

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)



Execution Environment

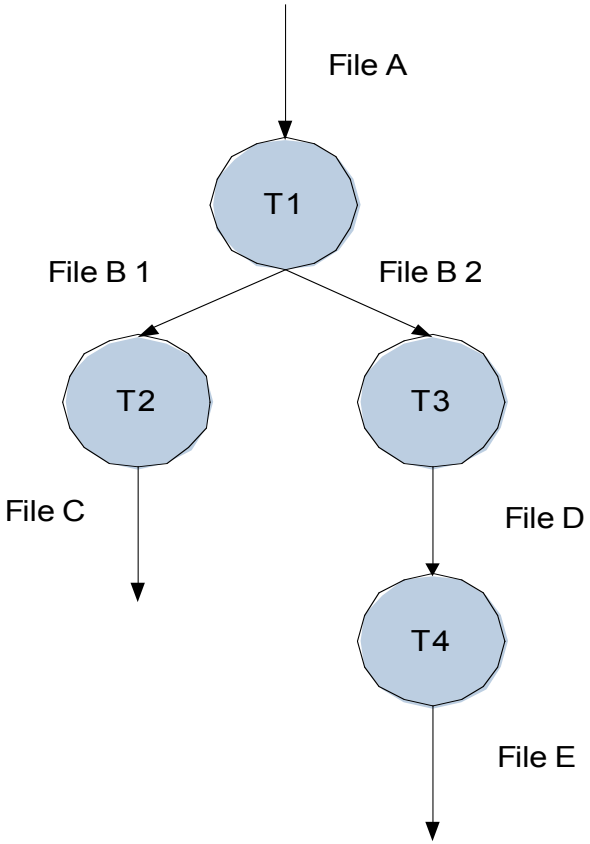


Outline

- Pegasus
- Time Optimizations
  - Data reuse
  - Workflow restructuring
  - Resource provisioning
- Space Optimizations
  - Workflow-level data management
  - Task-level data management
- Application Experiences and Science Impacts
- Conclusions

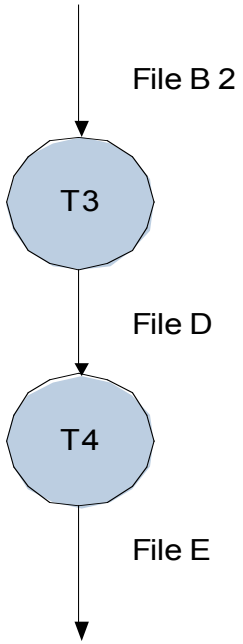
Data Reuse

Sometimes it is cheaper to access the data than to regenerate it



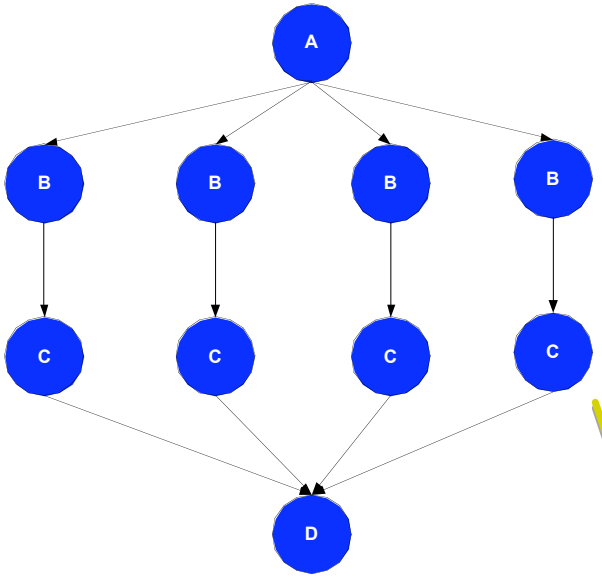
Original Workflow

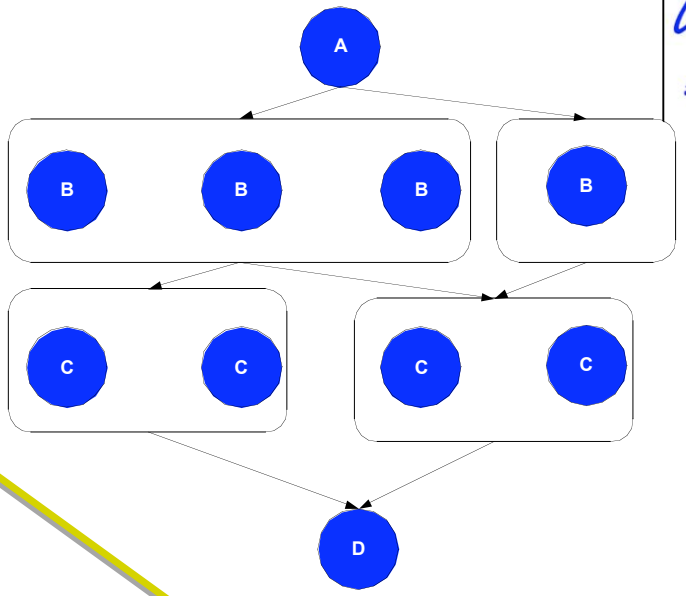
Files C and B 2 are available



Reduced Workflow

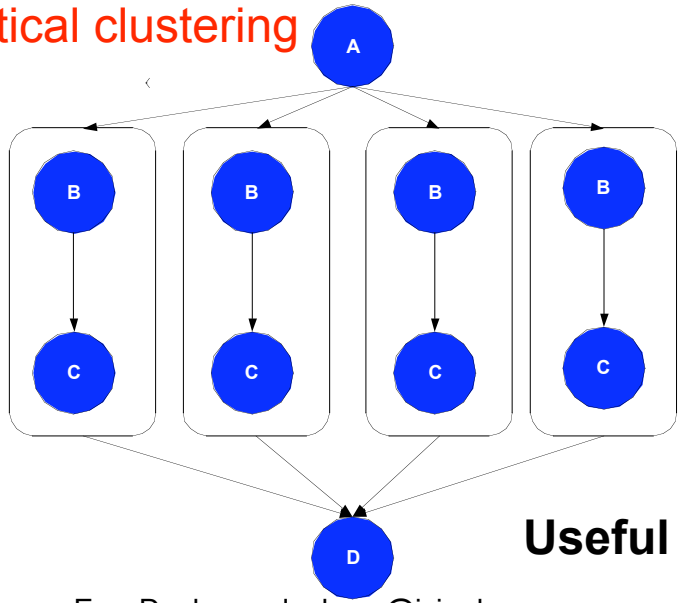
Node clustering



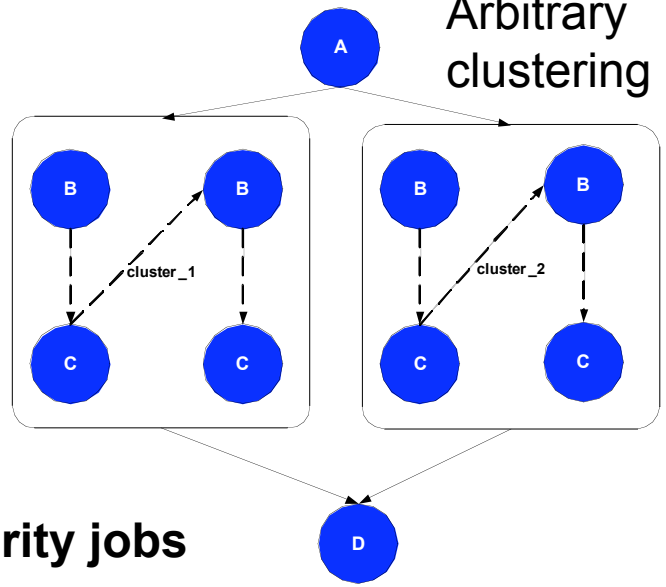


Level-based clustering

Vertical clustering



Arbitrary clustering

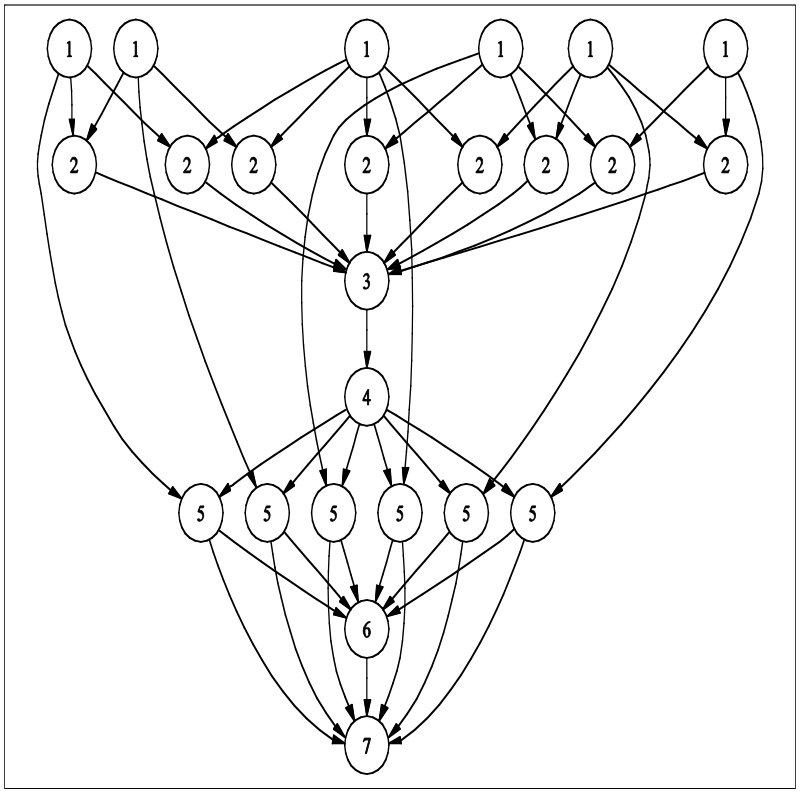


Useful for small granularity jobs

Ewa Deelman, deelman@isi.edu

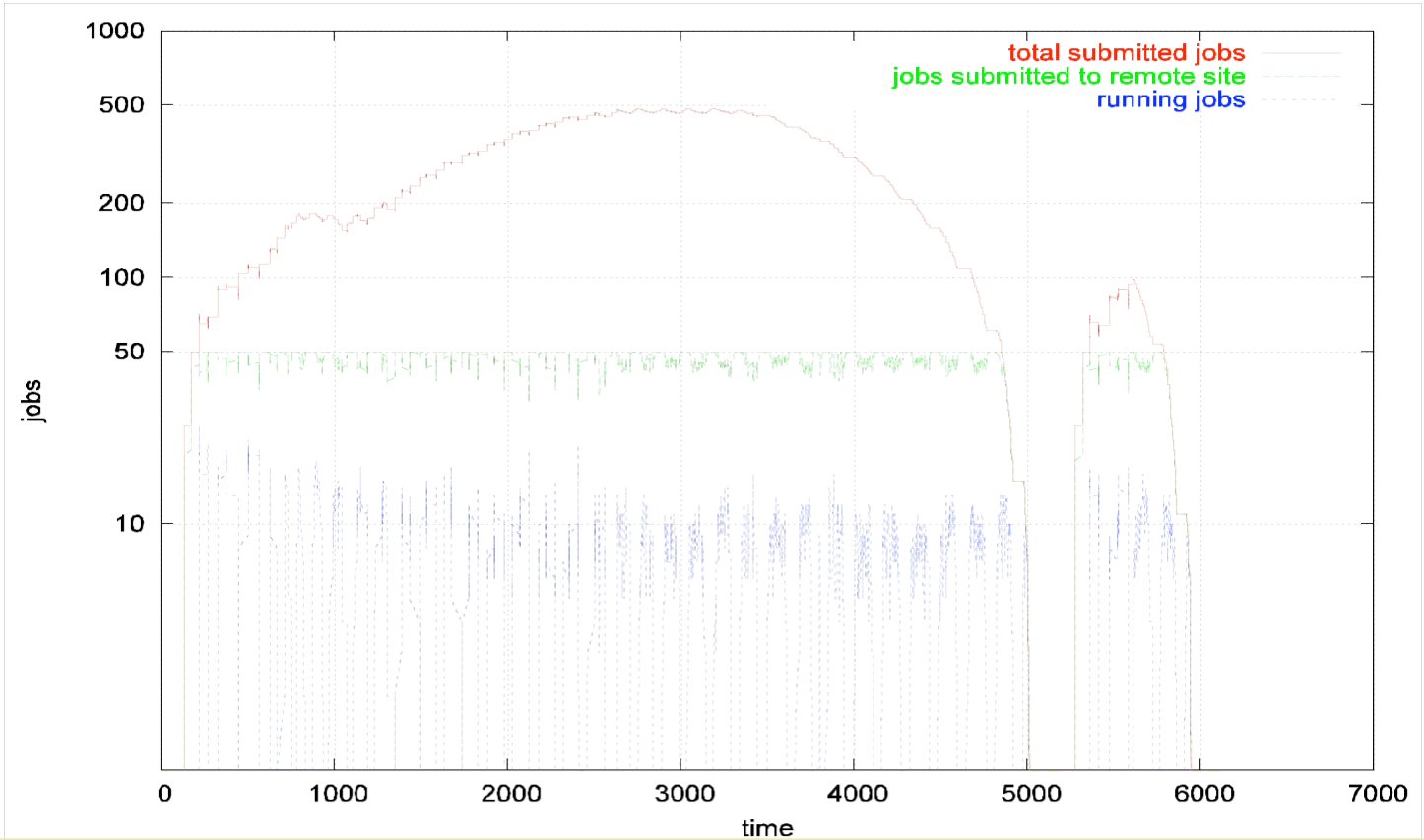
Montage Workflow of ~1,500 nodes

Level	Transformation Name	No. of jobs at level	Runtime of a job at level (in seconds)
1	mProject	180	6
2	mDiffFit	1010	1.4
3	mConcatFit	1	44
4	mBgModel	1	32
5	mBackground	180	0.8
6	mImgtbl	1	3.5
7	mAdd	1	60



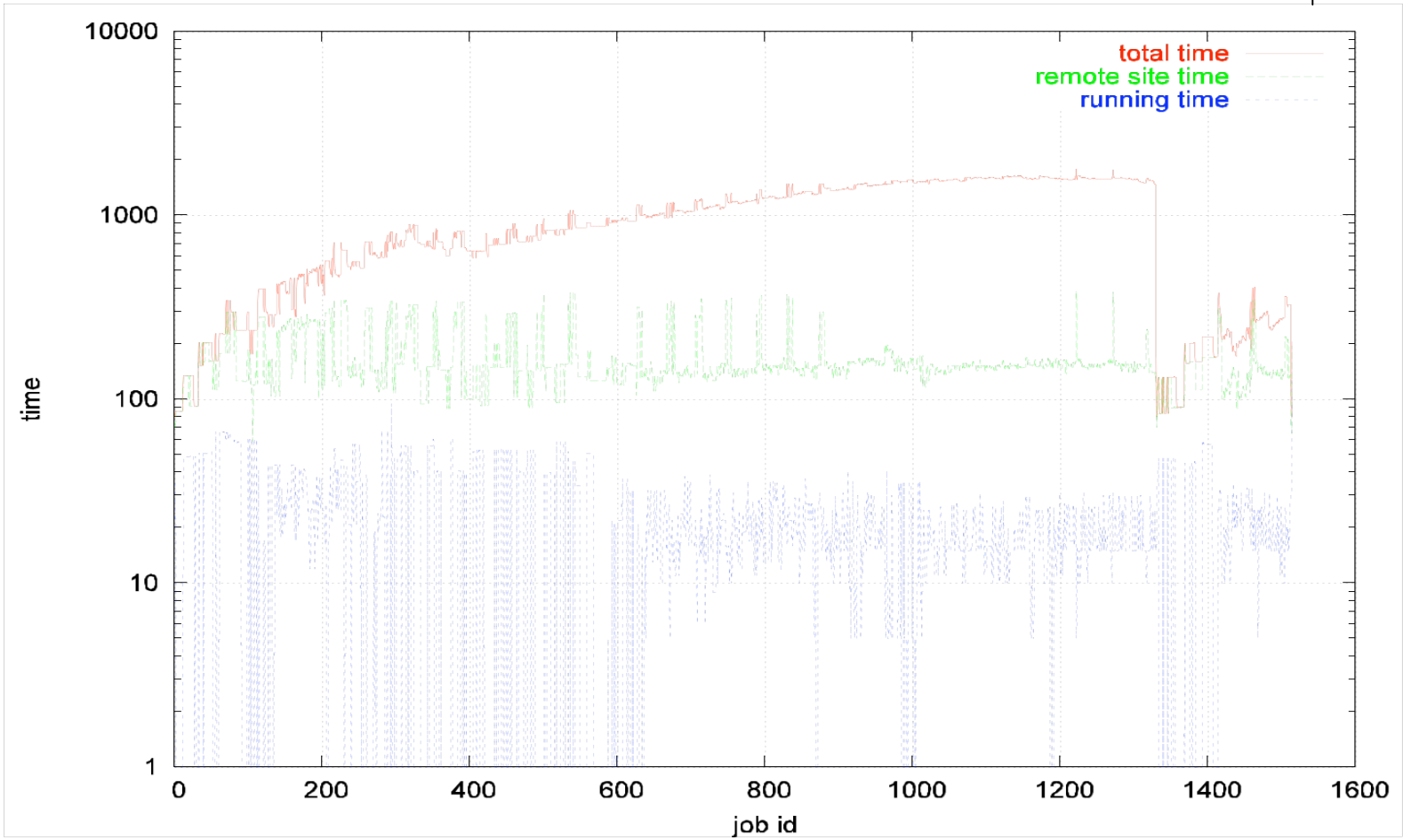
Montage Workflow running on the TeraGrid

- No modifications, 50 jobs throttled at Condor level
- Total time ~ 6,000 seconds



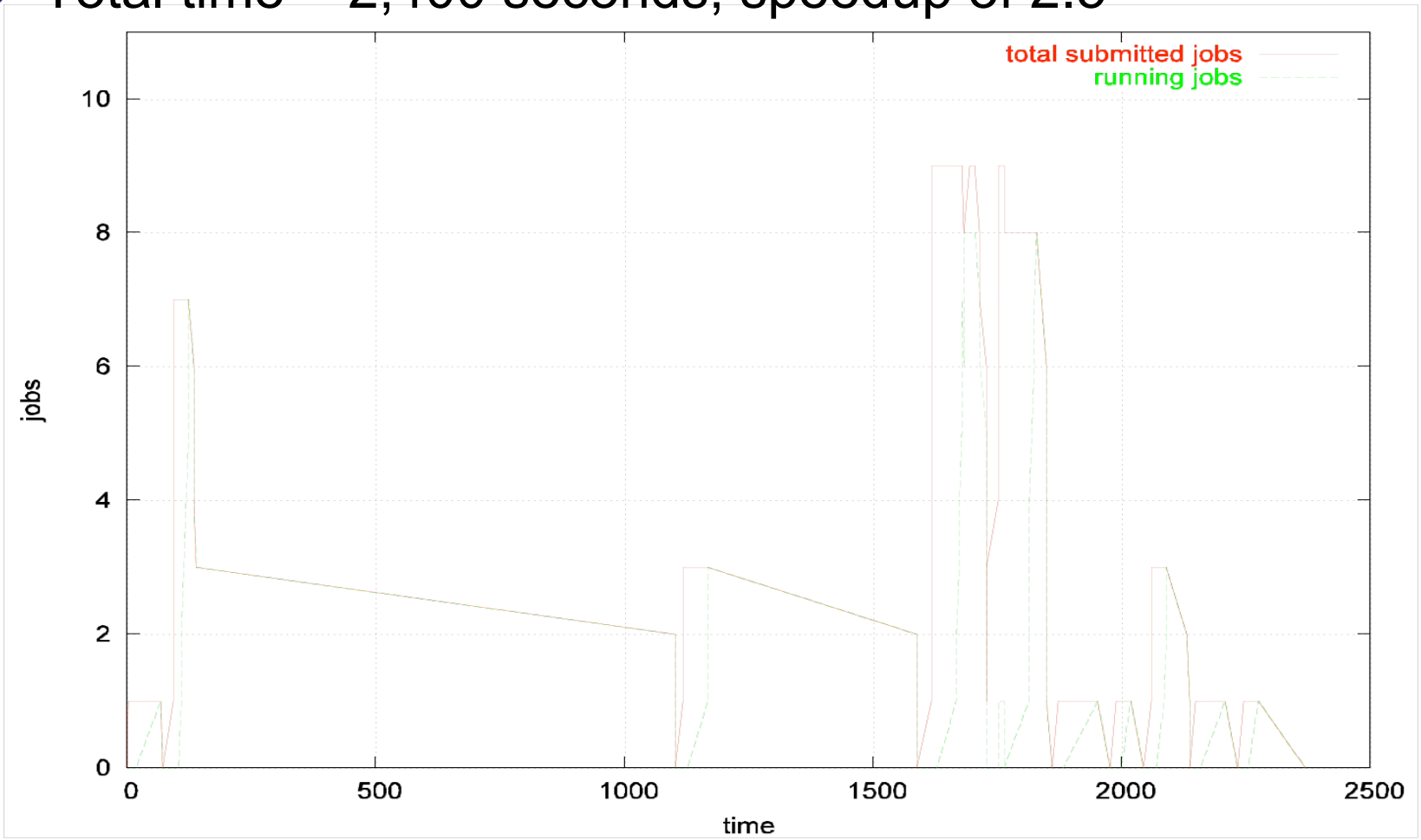
E. Deelman, et al. Pegasus: a Framework for Mapping Complex Scientific Workflows onto Distributed Systems, Scientific Programming Journal, Volume 13, Number 3, 2005

Breakdown of overheads (in seconds)

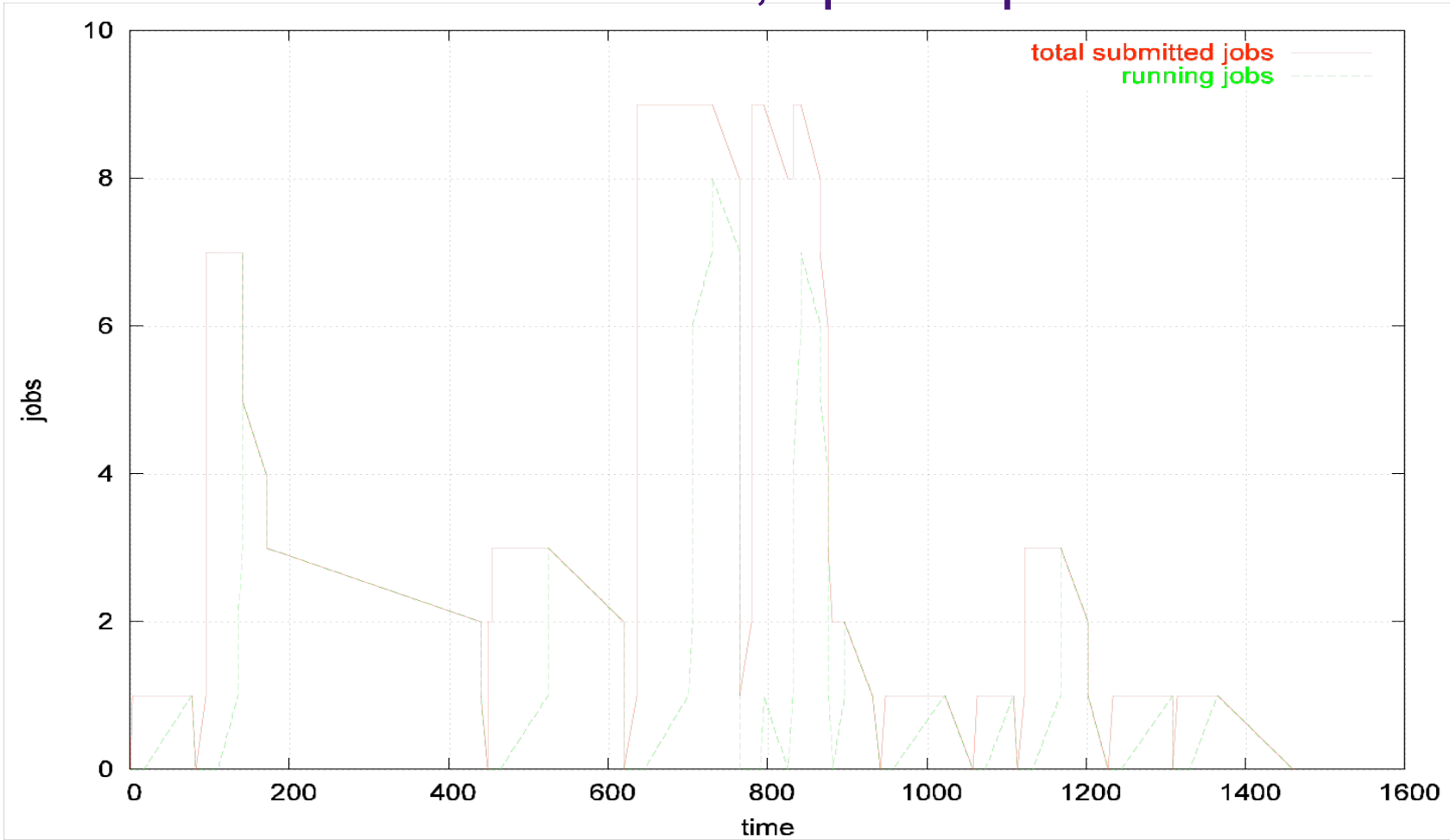


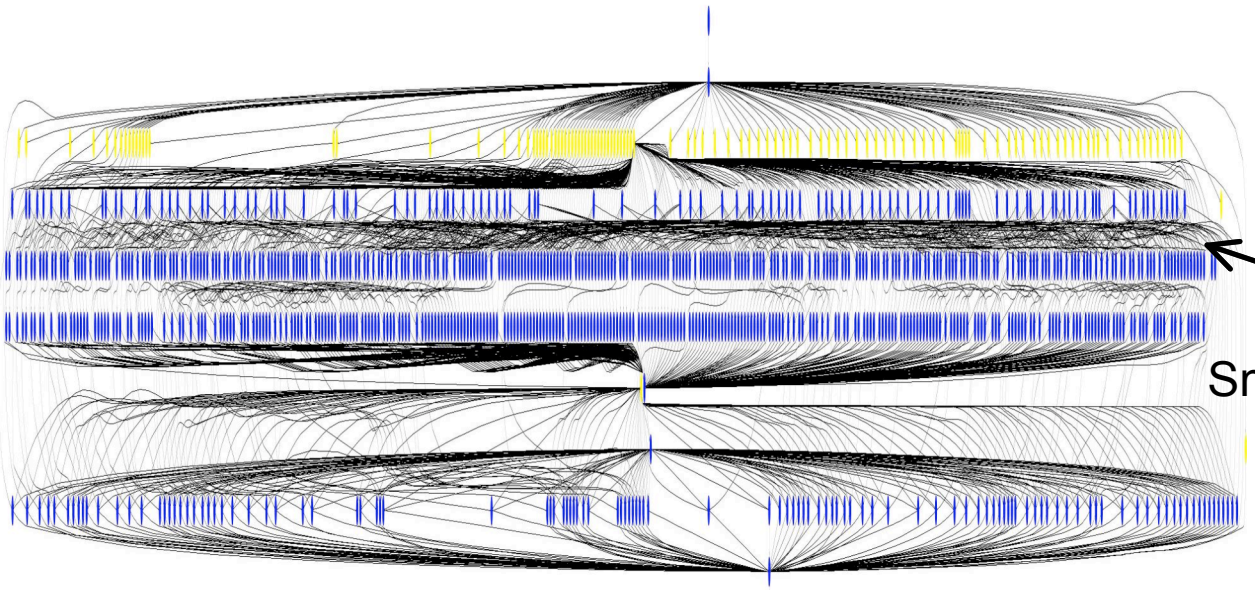
Clustering of 60 jobs per cluster at each level

- Total jobs = 35, no delays in the condor queue
- Total time ~ 2,400 seconds, speedup of 2.5



60 jobs per cluster
MPI-based Master/Slave execution in each cluster using 10 processors
total runtime 1420 seconds, speedup of 4.2

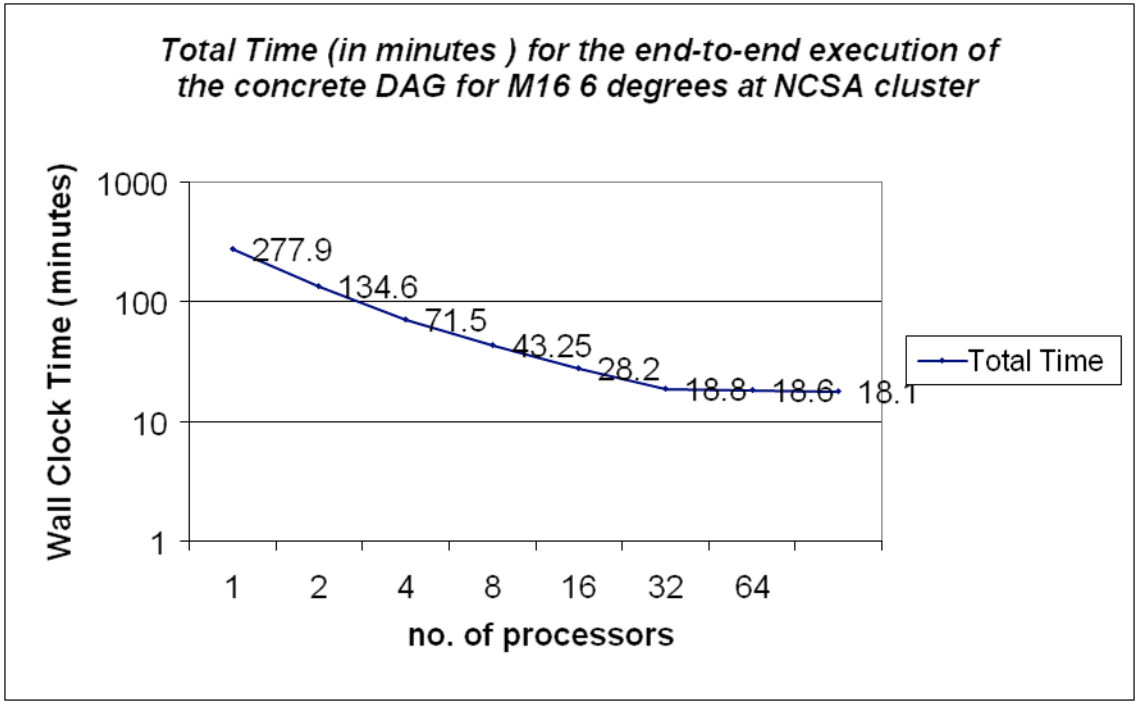




Small 1,200 Montage Workflow

Montage application

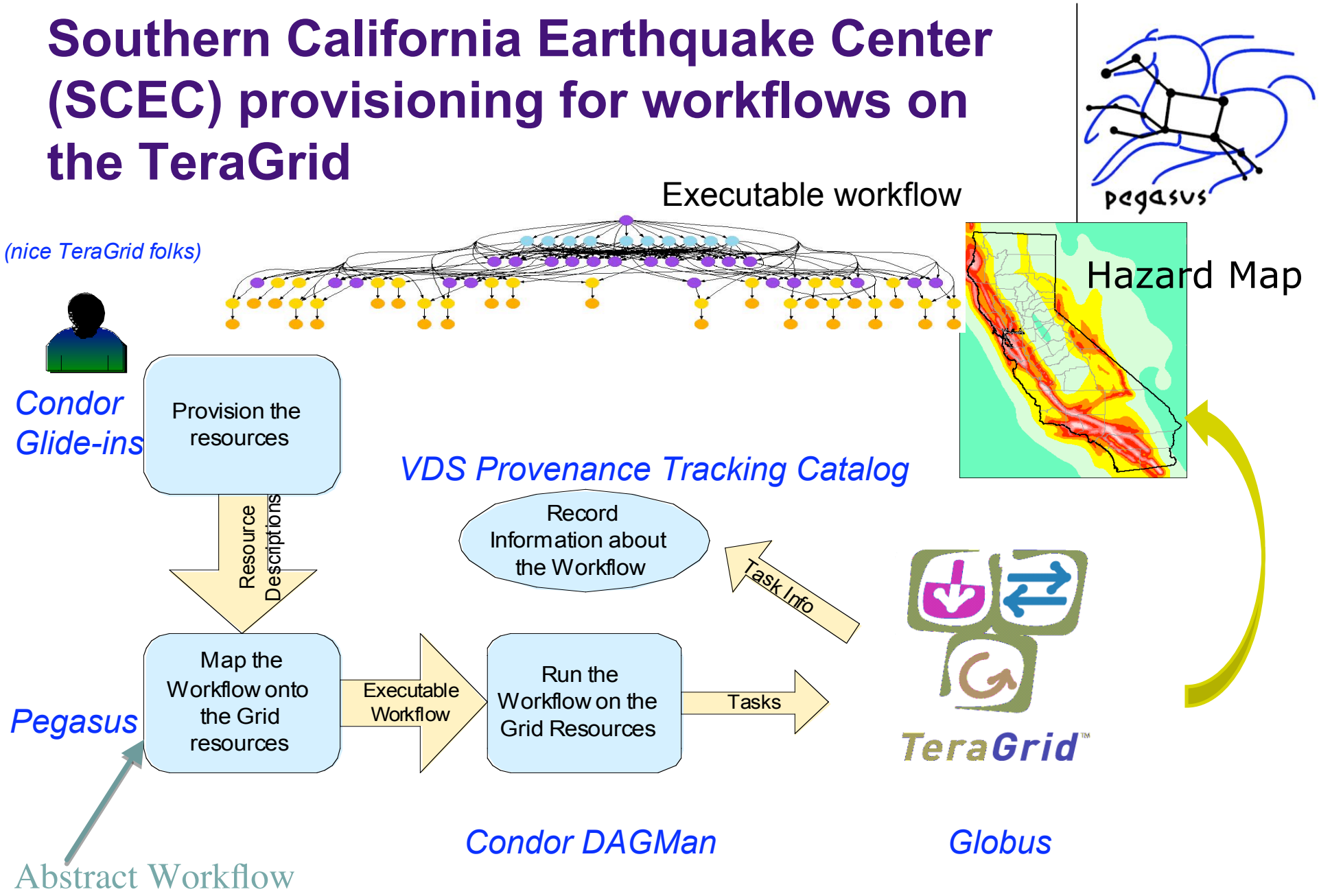
- **~7,000 compute jobs**
- **~10,000 nodes in the executable workflow**
- **same number of clusters as processors**
- **speedup of ~15 on 32 processors**



Outline

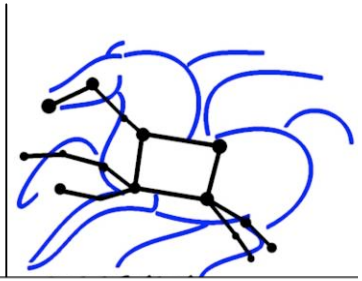
- Pegasus
- Time Optimizations
  - Data reuse
  - Workflow restructuring
  - Resource provisioning
- Space Optimizations
  - Workflow-level data management
  - Task-level data management
- Application Experiences and Science Impacts
- Conclusions

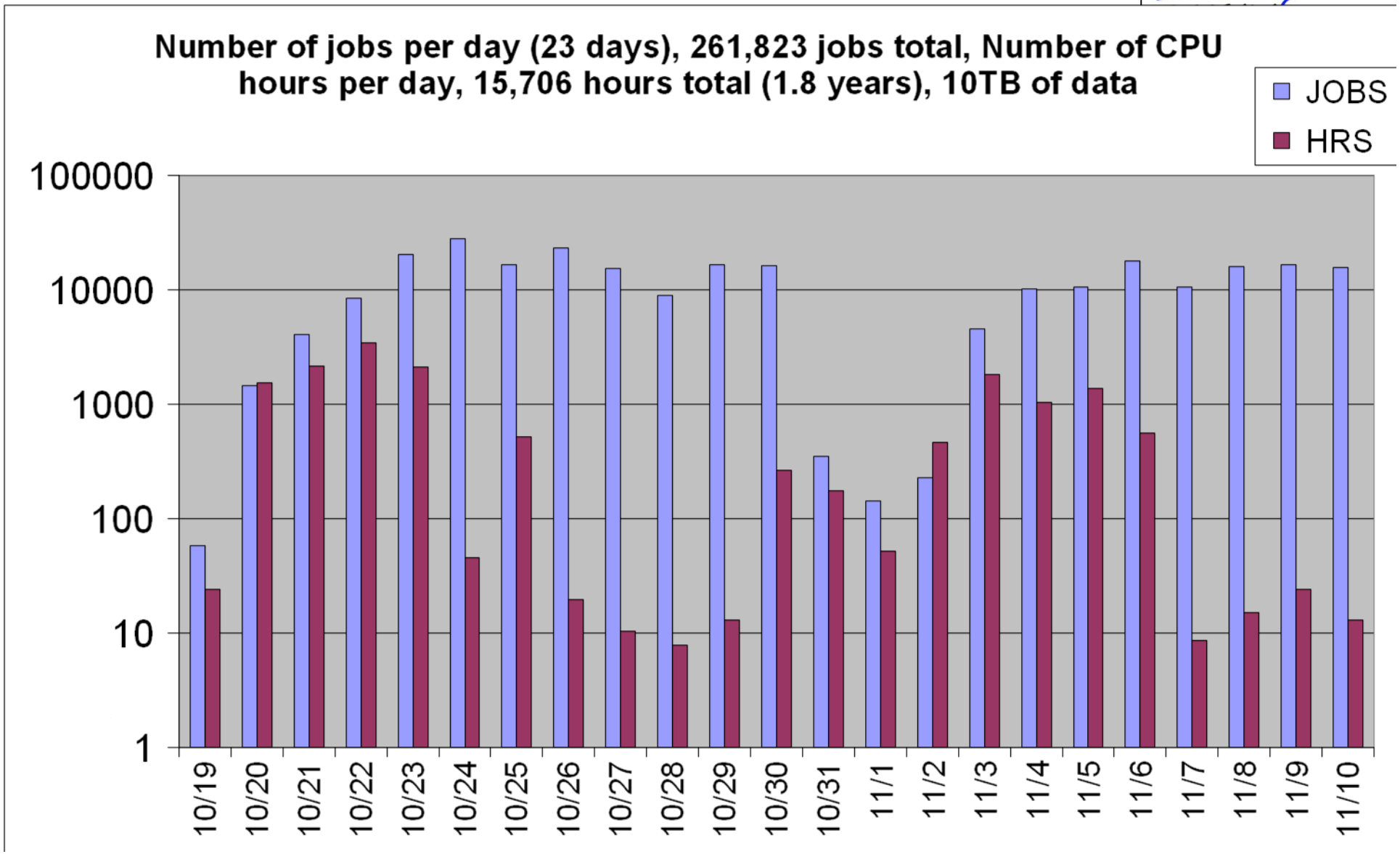
Southern California Earthquake Center (SCEC) provisioning for workflows on the TeraGrid



Joint work with: R. Graves, T. Jordan, C. Kesselman, P. Maechling, D. Okaya & others

Performance results for 2 SCEC sites (Pasadena and USC) on the TeraGrid

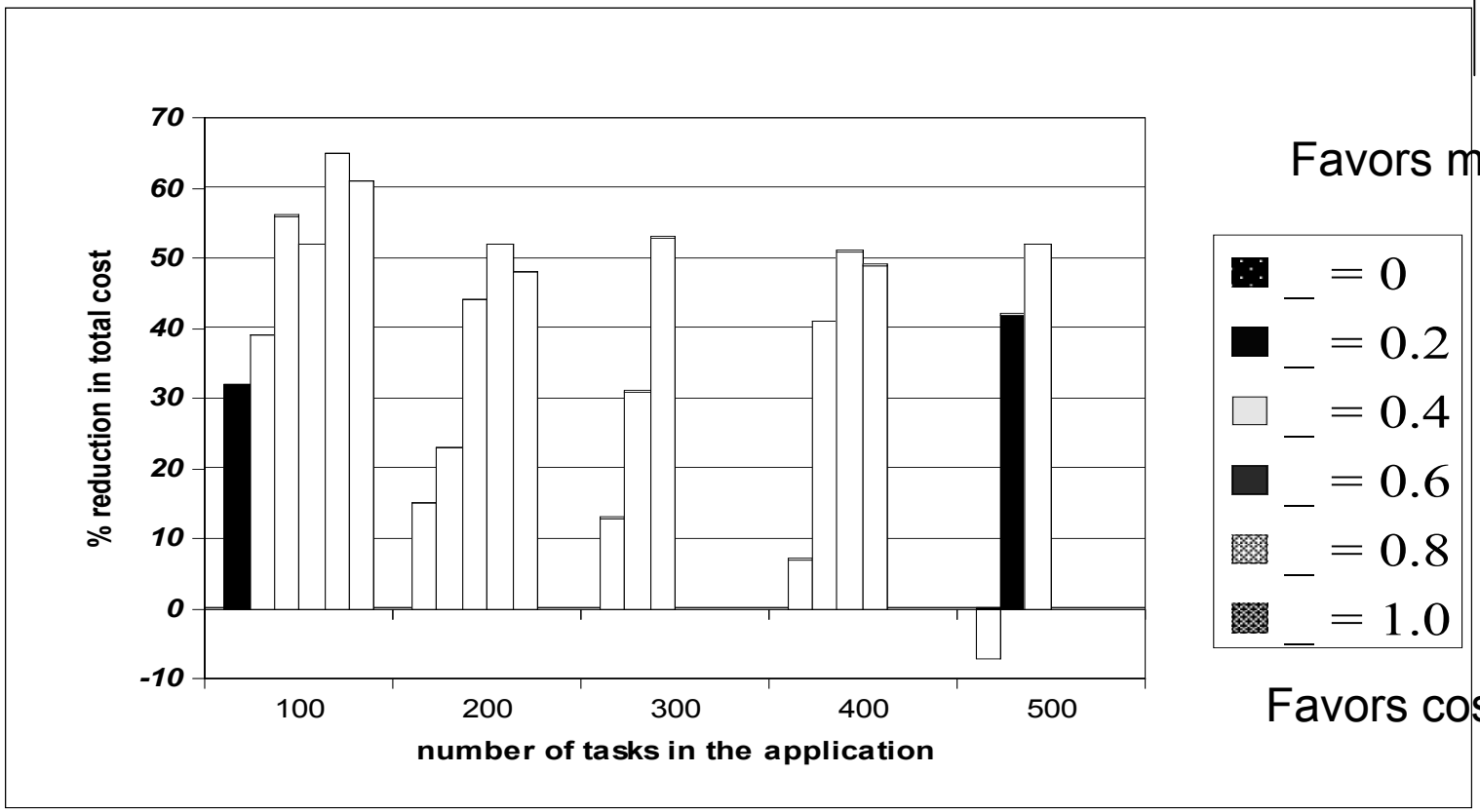




Approach to Provisioning Resources Ahead of the Execution

- Assume resources publish their availability in the form of “slot
- Pick the slots that would
  - Minimize the workflow makespan, and
  - Minimize the cost of the allocation (proportional to the allocation size)
    - Initially slots are indivisible
- Evaluate using Min-min for choosing the slots and Genetic-type algorithms
- Evaluate using random workflows

% reduction in total cost (combines makespan and allocation costs)



4 compute sites, ~ 100 processors total, ~200 slots

GA in general achieves a 25-30% reduction in the total cost over Min-Min

In 30% of cases, Min-Min could not complete the schedule

G. Singh, C. Kesselman, E. Deelman, Application-level Resource Provisioning on the Grid, e-Science 2006, to appear

Outline

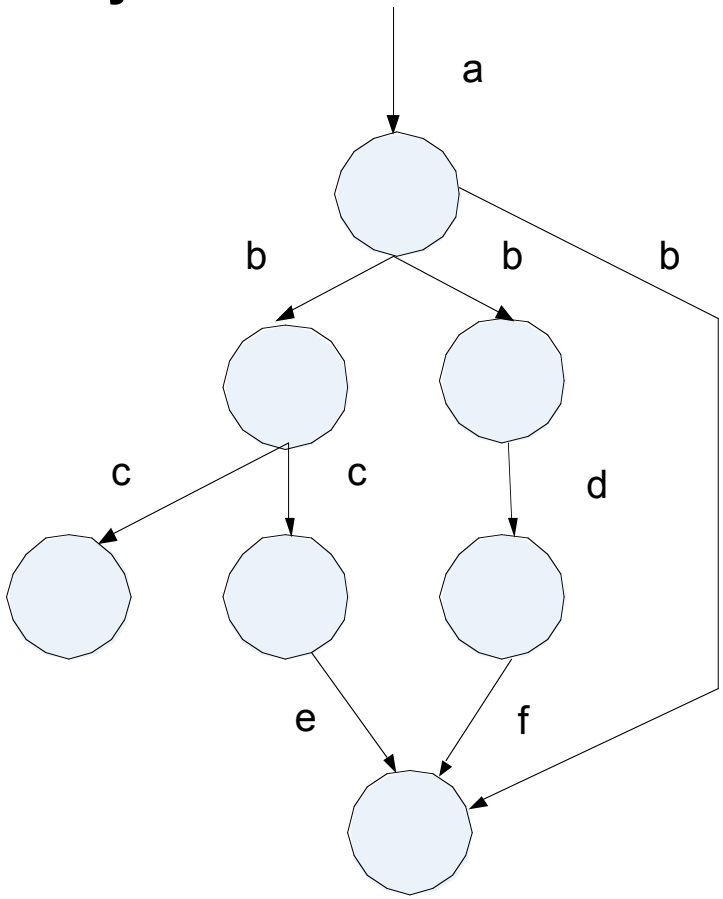
- Pegasus
- Time Optimizations
  - Data reuse
  - Workflow restructuring
  - Resource provisioning
- Space Optimizations
  - Workflow-level data management
  - Task-level data management
- Application Experiences and Science Impacts
- Conclusions

Optimizing Space

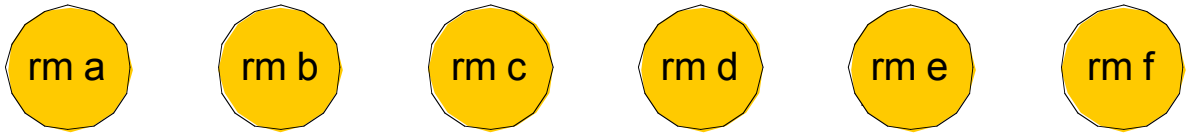
- Input data is staged dynamically to remote sites
- New data products are generated during execution
- For large workflows 10,000+ files
  - Similar order of intermediate and output files
  - Total space occupied is far greater than available space—failures occur
- Solution 1: Generate a “cleanup DAG” which can be run after the workflow completes
- Issues:
  - May not be able to complete the workflow due to lack of space

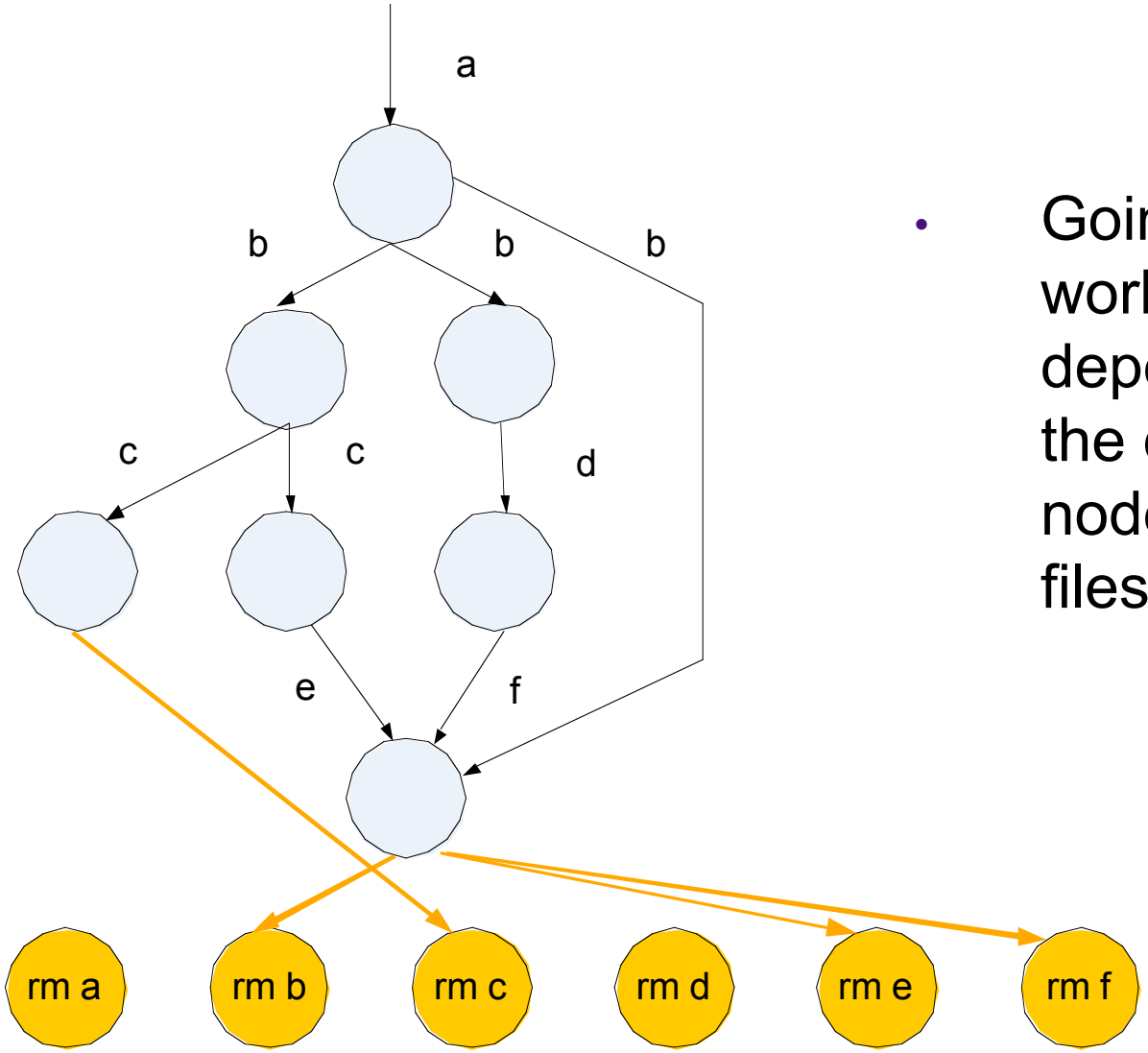
Solution2:

Determine which data are no longer needed and when
Add nodes to the workflow do cleanup data along the way

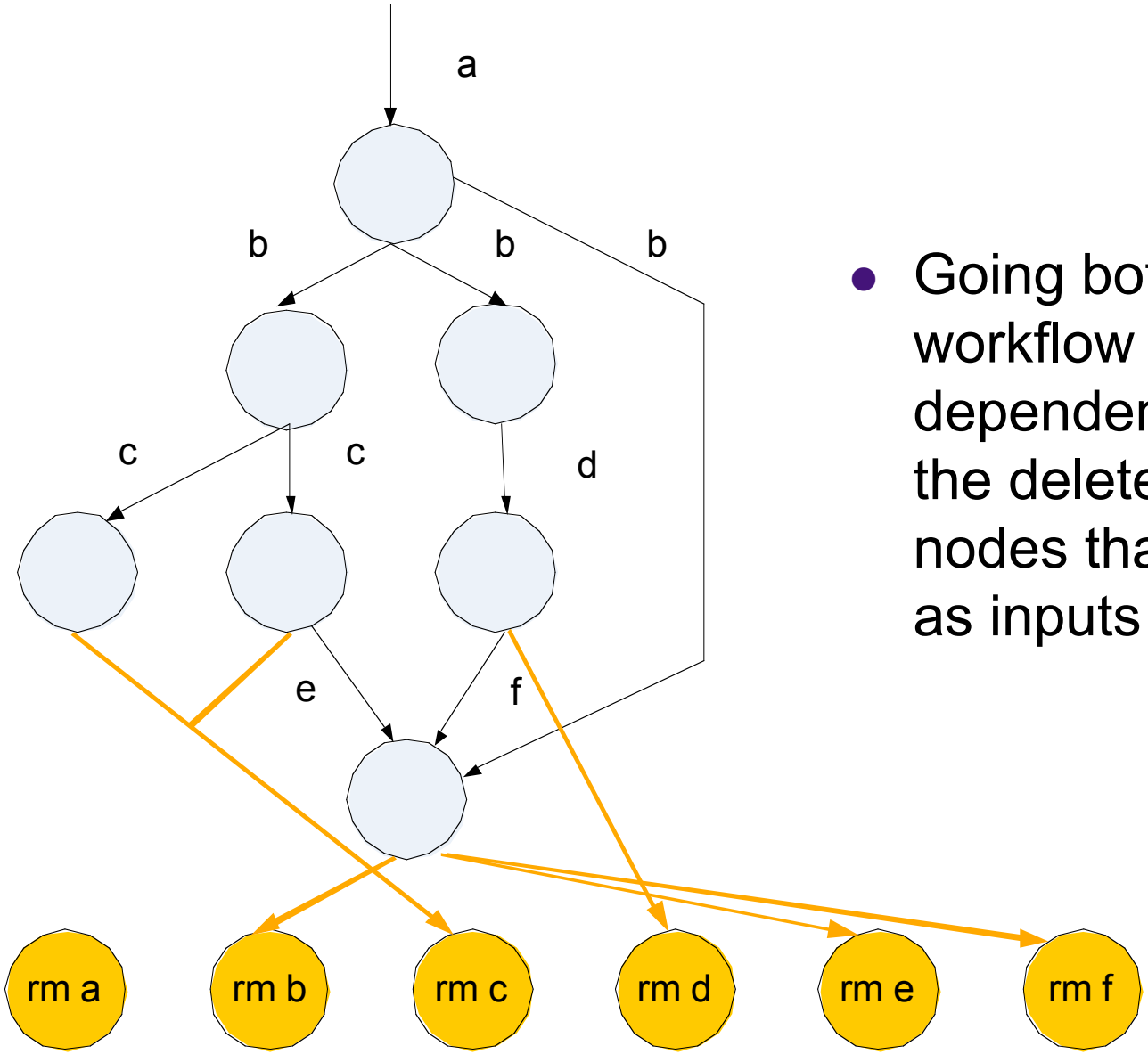


1. Add nodes representing each file

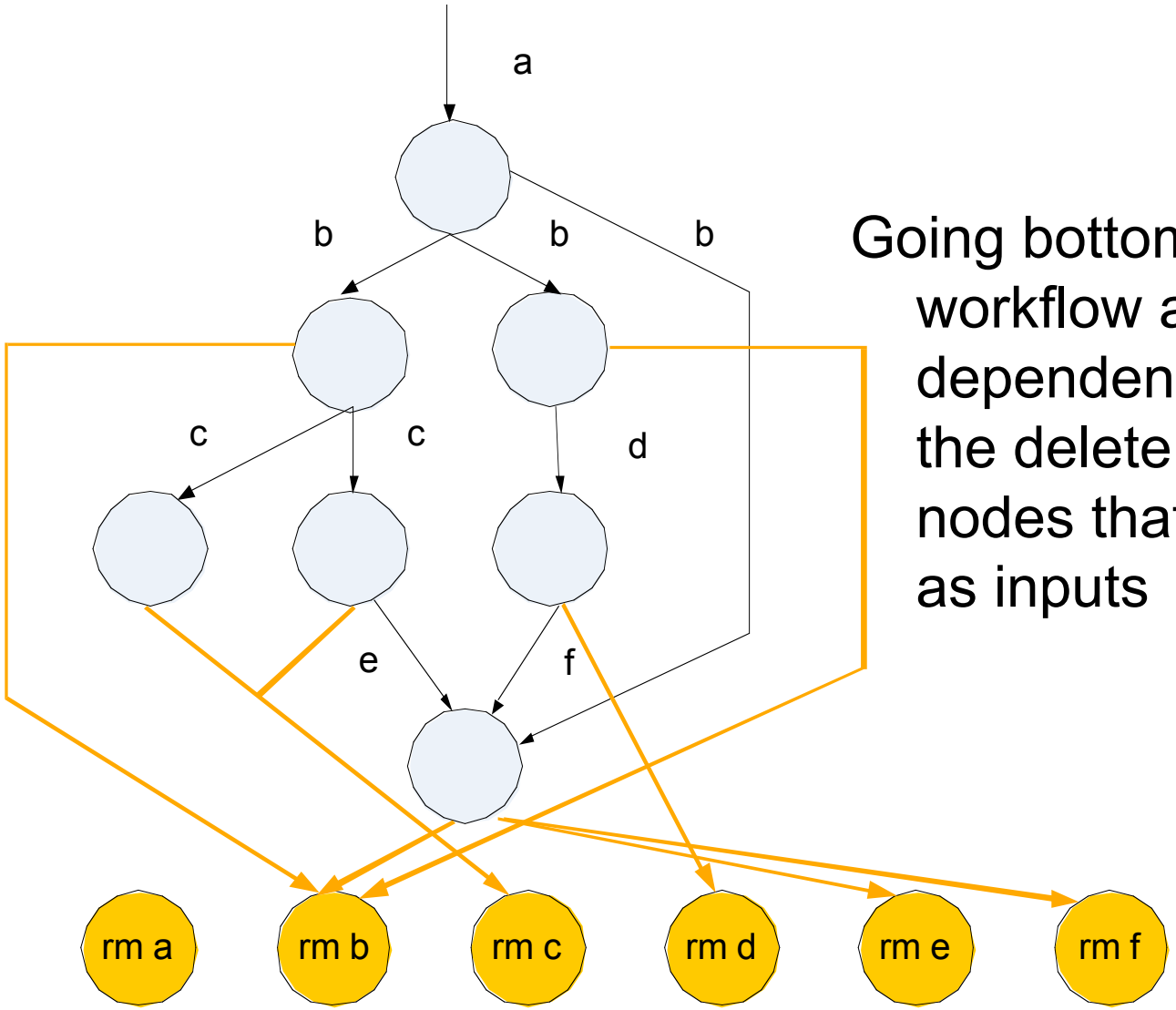




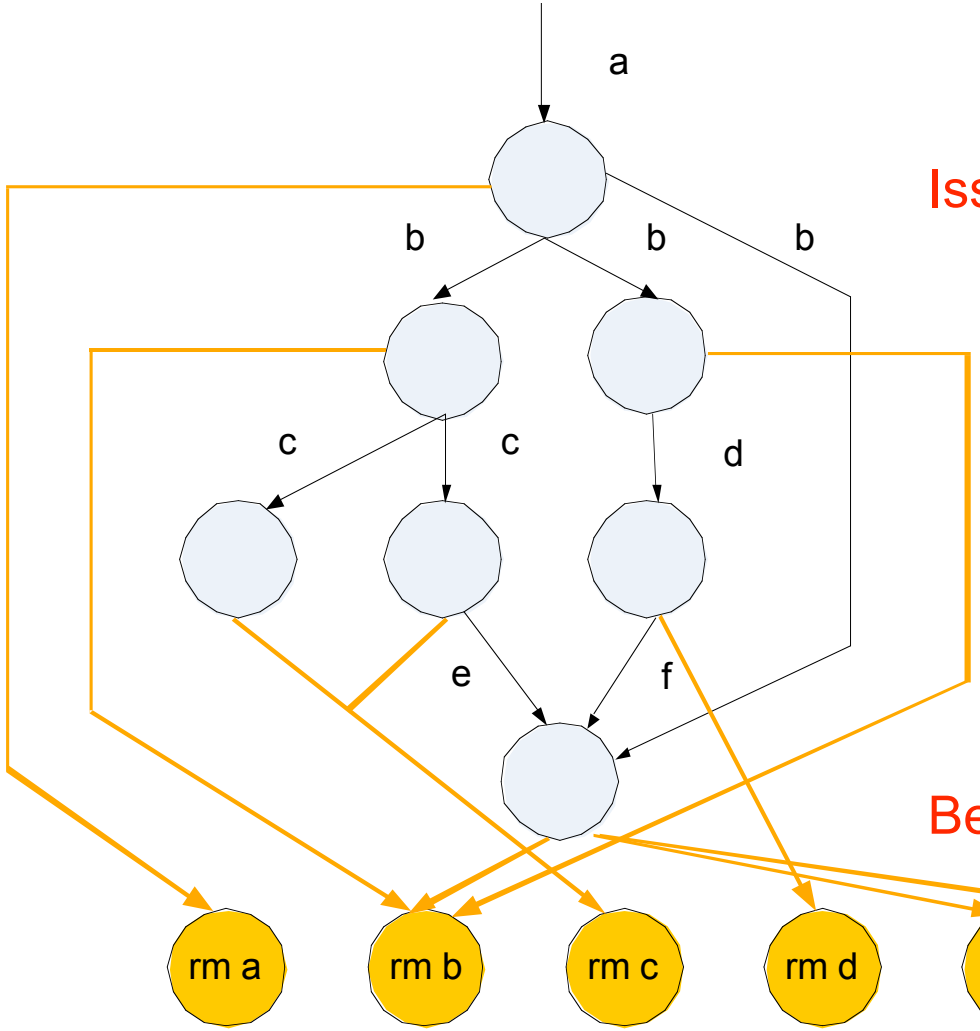
- Going bottom up in the workflow add dependencies between the delete node and the nodes that have the files as inputs



- Going bottom up in the workflow add dependencies between the delete node and the nodes that have the files as inputs



Going bottom up in the workflow add dependencies between the delete node and the nodes that have the files as inputs



Issues:

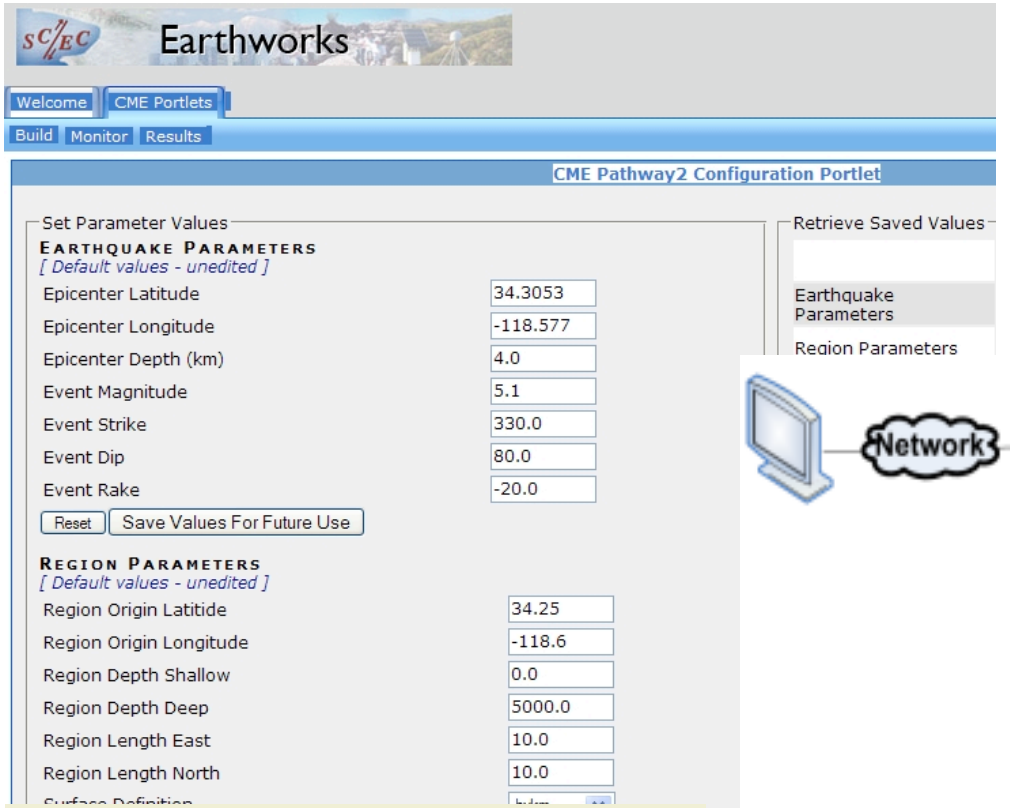
- minimize the number of nodes and dependencies added so as not to slow down workflow execution
- deal with portions of workflows scheduled to multiple sites
- deal with files on partition boundaries

Benefits: study underway

Outline

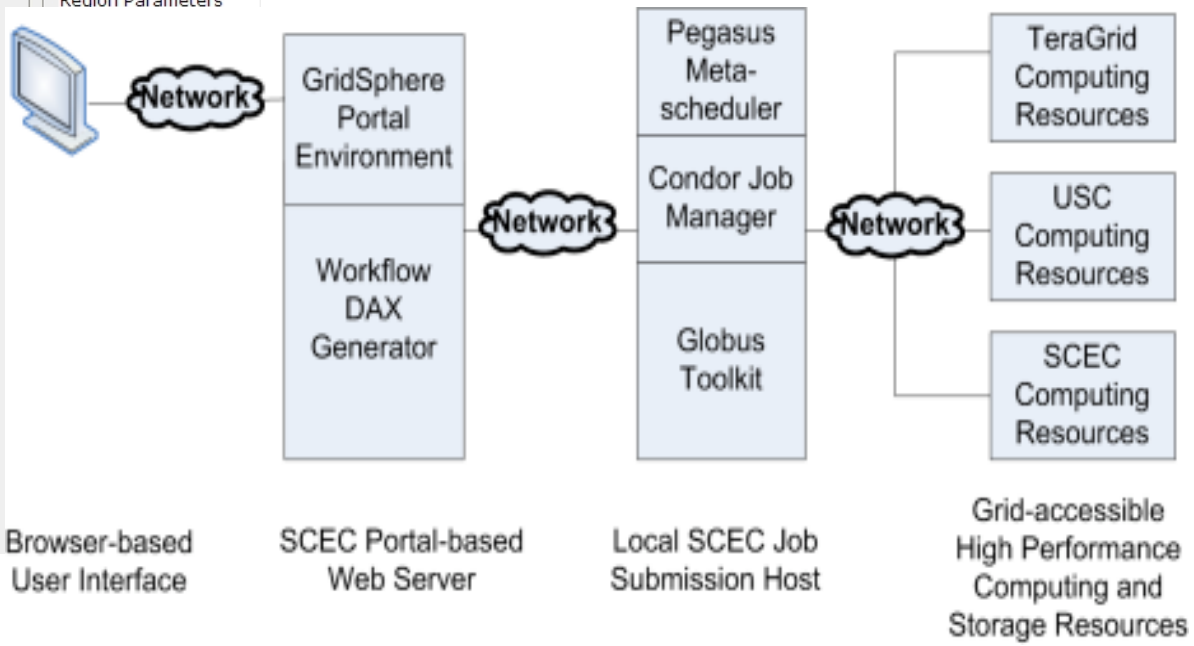
- Pegasus
- Time Optimizations
  - Data reuse
  - Workflow restructuring
  - Resource provisioning
- Space Optimizations
  - Workflow-level data management
  - Task-level data management
- Application Experiences and Science Impacts
- Conclusions

Portals, Providing high-level Interfaces



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

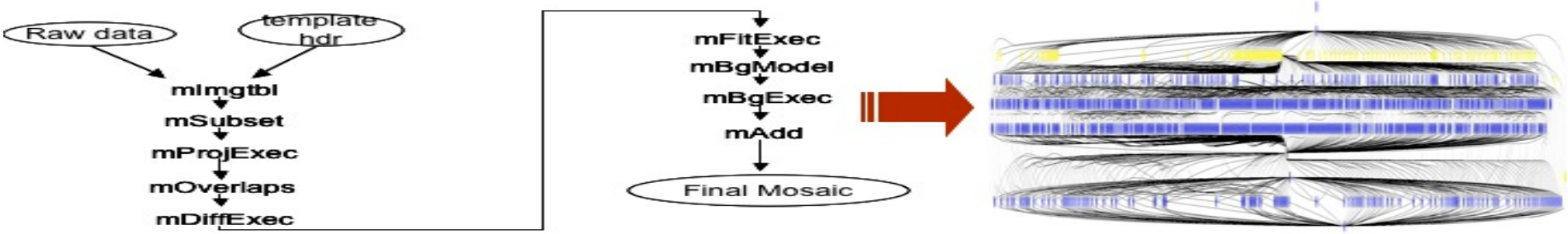
TG Science Gateway, Washington University

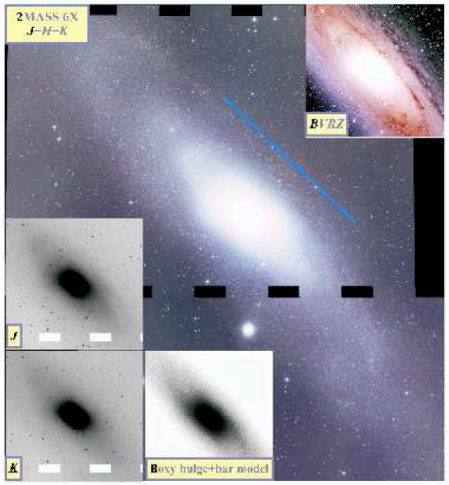


SCEC Earthworks: Community Access to Wave Propagation Simulations, J. Muench, H. Francoeur, D. Okaya, Y. Cui, P. Maechling, E. Deelman, G. Mehta, T. Jordan
TG 2006

National Virtual Observatory and Montage: Building Science-Grade Mosaics of the Sky

Workflow technologies were used to transform a single-processor code into a complex workflow and parallelized computations to process larger-scale images.





- Pegasus maps workflows with thousands of tasks onto NSF's TeraGrid
- Pegasus improved overall runtime by 90% through automatic workflow restructuring and minimizing execution overhead

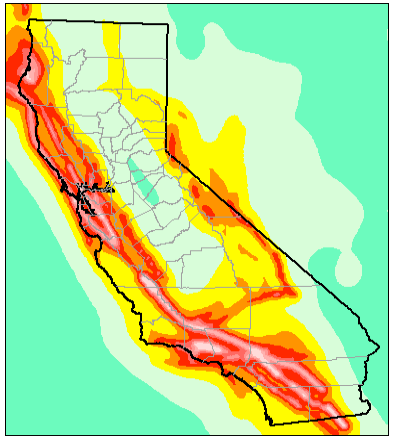
Montage Science Result : Verification of a Bar in the Spiral Galaxy M31, Beaton et al. *Ap J Lett* in press

Eleven major projects and surveys world wide, such as the Spitzer Space Telescope Legacy teams have integrated Montage into their pipelines and processing environments to generate science and browse products for dissemination to the astronomy community.

Montage is a collaboration between IPAC, JPL and CACR

Ewa Deelman, deelman@isi.edu www.isi.edu/~deelman pegasus.isi.edu

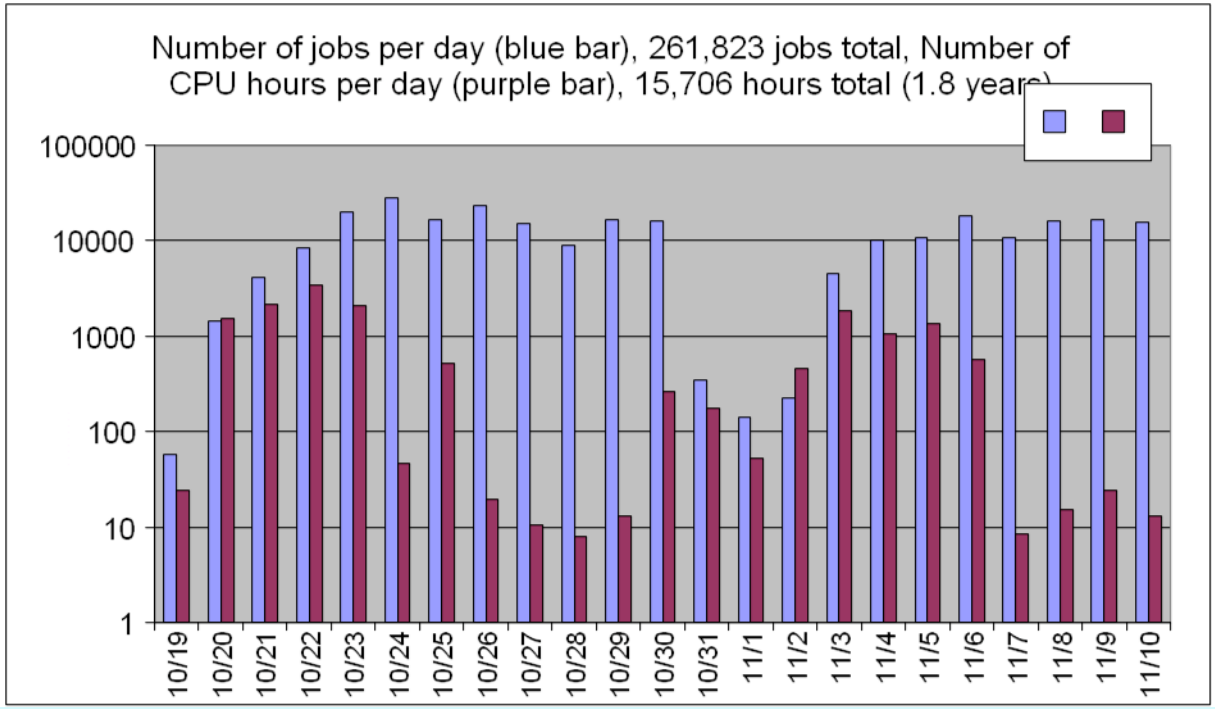
Southern California Earthquake Center (SCEC)



- SCEC's Cybershake is used to create Hazard Maps that specify the maximum shaking expected over a long period of time

- Used by civil engineers to determine building design tolerances

Pegasus mapped SCEC CyberShake workflows onto the TeraGrid in Fall 2005. The workflows ran over a period of 23 days and processed 20TB of data using 1.8 CPU Years. Total tasks in all workflows: 261,823.



***CyberShake Science result**: CyberShake delivers new insights into how rupture directivity and sedimentary basin effects contribute to the shaking experienced at different geographic locations. As a result more accurate hazard maps can be created.*

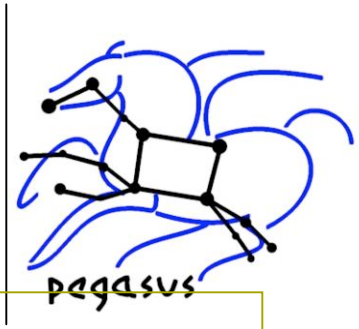
SCEC is led by Tom Jordan, USC

Ewa Deelman, deelman@isi.edu

Pegasus: Planning for Execution in Grids

- Pegasus bridges the scientific domain and the execution environment
- Pegasus enables scientists to construct workflows in abstract terms without worrying about the details of the underlying CyberInfrastructure
- Pegasus is used day-to-day to map complex, large-scale scientific workflows with thousands of tasks processing TeraBytes of data
- Pegasus applications include NVO's Montage, SCEC's CyberShake simulations, LIGO's Binary Inspiral Analysis, and others
- Pegasus improves the performance of applications through:
  - Data reuse to avoid duplicate computations and provide reliability
  - Workflow restructuring to improve resource allocation
  - Automated task and data transfer scheduling to improve overall runtime
- Pegasus provides reliability through dynamic workflow remapping
- Pegasus uses Condor's DAGMan for workflow execution and Globus to provide the middleware for distributed environments

Current and Future Research



- Resource selection
- Resource provisioning
- Workflow restructuring
- Adaptive computing
  - Workflow refinement adapts to changing execution environment
- Workflow provenance
- Management and optimization across multiple workflows
- Workflow debugging
- Streaming data workflows
- Automated guidance for workflow restructuring
- Support for long-lived and recurrent workflows

Acknowledgments

- The Pegasus team consists of Ewa Deelman, Gaurang Mehta, Mei-Hui Su, and Karan Vahi (ISI)
- Thanks to Yolanda Gil (ISI) for collaboration on scientific workflow issues
- Thanks to Montage collaborators: Bruce Berriman, John Good, Dan Katz, and Joe Jacobs
- Thanks to SCEC collaborators: Tom Jordan, Robert Graves, Phil Maechling, David Okaya, Li Zhao
- Thanks to LIGO collaborators: Kent Blackburn, Duncan Brown, and David Meyers
- Thanks to the National Science Foundation for the support of this work

Relevant Links

- Pegasus: pegasus.isi.edu
  - released as part of VDS, joint work with Ian Foster
- NSF Workshop on Challenges of Scientific Workflows: vtcpc.isi.edu/wiki/, E. Deelman and Y. Gil (chairs)
- Workflows for e-Science, Taylor, I.J.; Deelman, E.; Gannon, D.B.; Shields, M. (Eds.), Dec. 2006, *to appear*
- Globus: www.globus.org
- Condor: www.cs.wisc.edu/condor/
- TeraGrid: www.teragrid.org
- Open Science Grid: www.opensciencegrid.org
- SCEC: www.scec.org
- Montage: montage.ipac.caltech.edu/
- LIGO: www.ligo.caltech.edu/