Scientific Workflows with Pegasus and DAGMan

Karan Vahi

Science Automation Technologies Group

USC Information Sciences Institute

Before we begin

- The tutorial involves hands on exercises
  - https://pegasus.isi.edu/tutorial/chtc15/index.php
- If you have a CHTC account, then you can logon to submit-5.chtc.wisc.edu or submit-3.chtc.wisc.edu to do the tutorial.
  - ssh <username>@submit-5.chtc.wisc.edu
  - Replace <username> with your username e.g. nu_vahi
- The tutorial can be done anytime. It is self contained.

Agenda

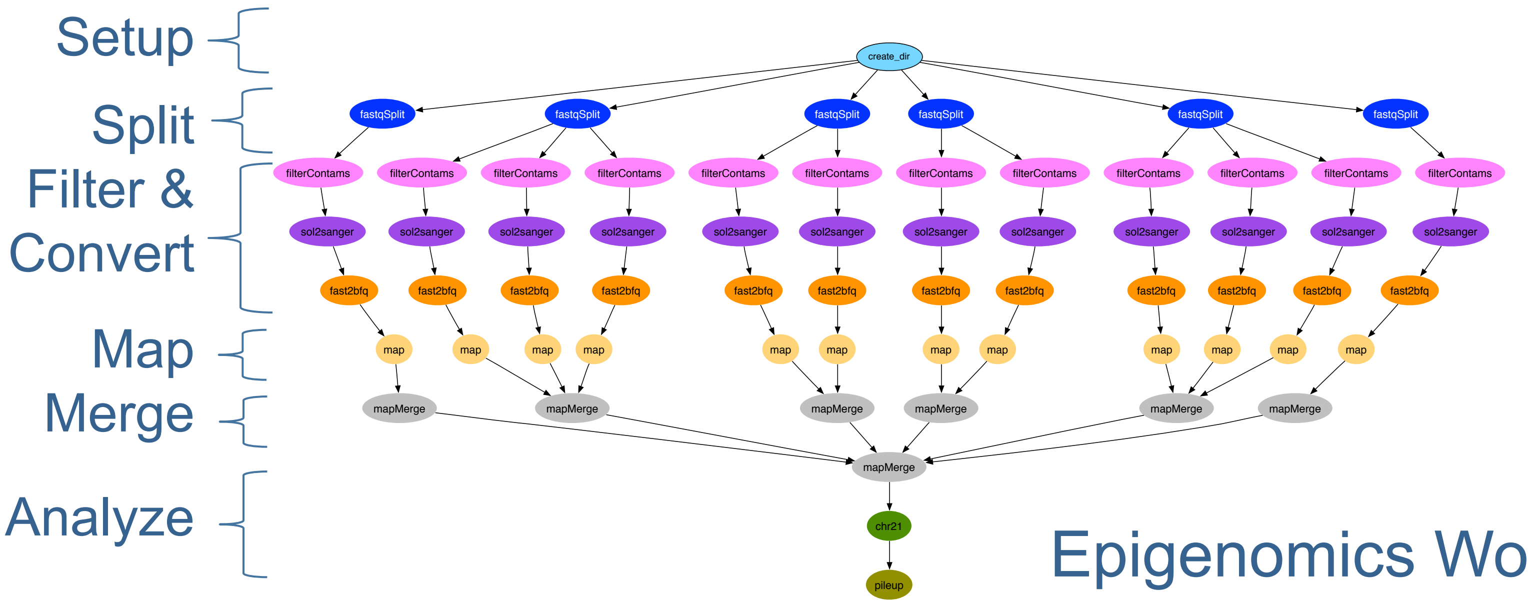
- Introduction to Workflows and Pegasus

- Hands-on Pegasus Tutorial Demonstration
  - Compose, setup and run a simple workflow
  - Monitor , debug and generate statistics of a workflow.
  - Run the same workflow with clustering turned on
  - Compose and submit a workflow with a MPI job.

- Advanced Topics

Scientific Workflows

- Orchestrate complex, multi-stage scientific computations
- Often expressed as directed acyclic graphs (DAGs)
- Capture analysis pipelines for sharing and reuse
- Can execute in parallel on distributed resources



Epigenomics Workflow

Scientific Workflow Challenges

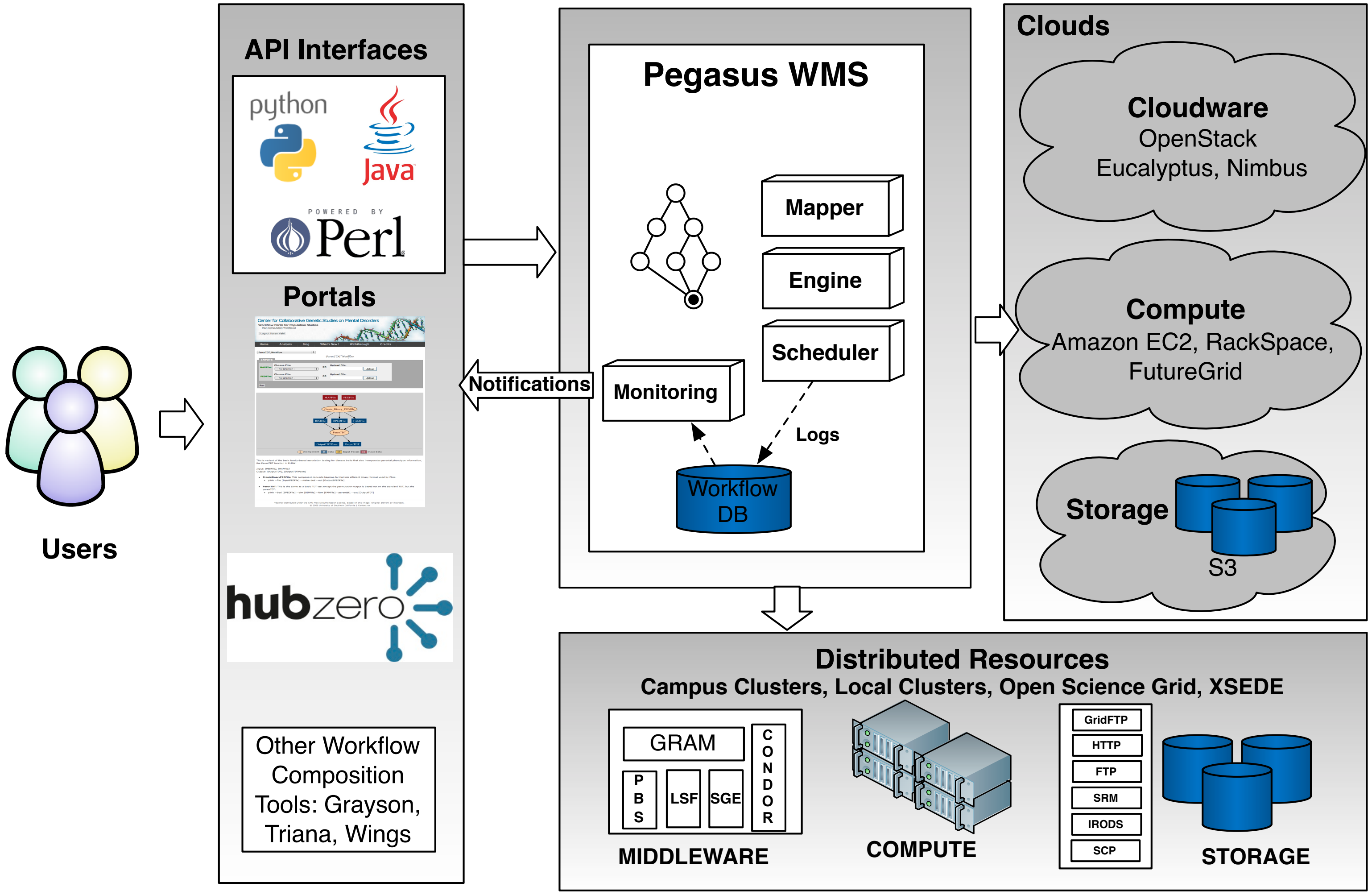
- **Portability**
  - How can you run a pipeline on Amazon EC2 one day, and a PBS cluster the next?
- **Data Management**
  - How do you ship in the small/large amounts data required by your pipeline?
  - Different protocols for different sites: Can I use SRM? How about GridFTP? HTTP and Squid proxies?
  - Can I use Cloud based storage like S3 on EC2?
- **Debug and Monitor Computations.**
  - 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
- **Restructure Pipelines for Improved Performance**
  - Short running tasks?
  - Data placement?

Pegasus Workflow Management System (est. 2001)

- A collaboration between USC and the Condor Team at UW Madison (includes DAGMan)
- Maps a resource-independent “abstract” workflow onto resources and executes the “executable” workflow
- Used by a number of applications in a variety of domains
- Provides reliability—can retry computations from the point of failure
- Provides scalability—can handle large data and many computations (kbytes-TB of data, 1-10^6 tasks)
- Infers data transfers, restructures workflows for performance
- Automatically captures provenance information
- Can run on resources distributed among institutions, laptop, campus cluster, Grid, Cloud

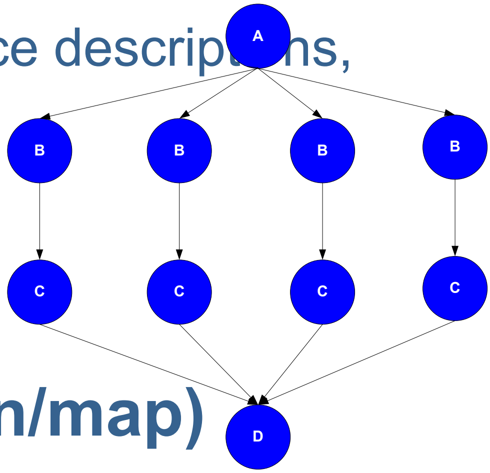
USC Viterbi School of Engineering

Pegasus WMS Environment



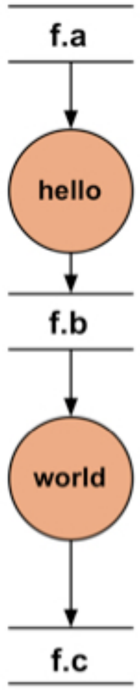
Pegasus Workflow Management System

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



- **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

DAX – XML format to describe Abstract Workflows



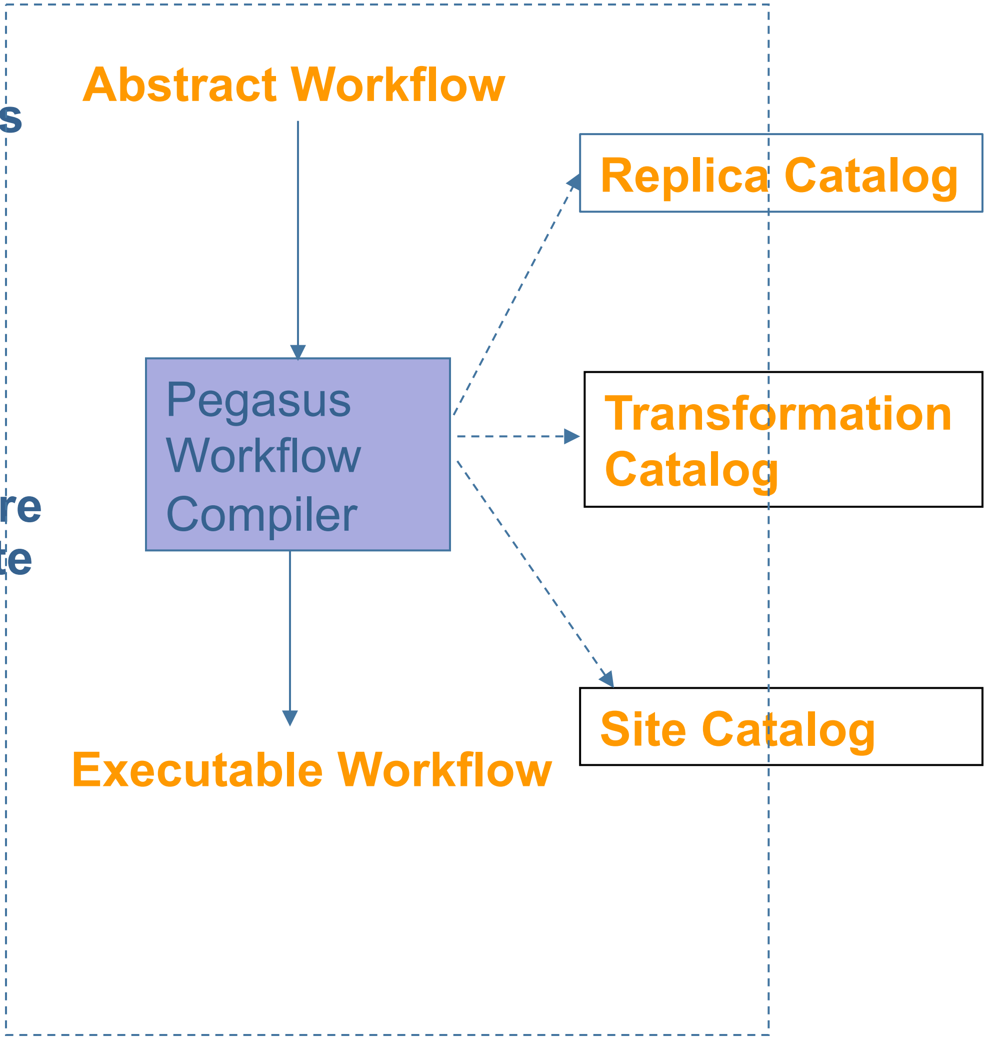
Abstract Workflow

```
<?xml version="1.0" encoding="UTF-8"?>
<adag version="3.4" name="hello-world" index="0" count="1">

<!-- Section: Job's, DAX's or Dag's - Defines a JOB or DAX or
DAG (Atleast 1 required) -->

   <job id="j1" namespace="pegasus" name="hello" version="4.0">
      <argument>-a hello -T 60 -i  <file name="f.a"/>
        -o <file name="f.b"/>
    </argument>
    <uses name="f.a" link="input" transfer="true"
register="true"/>
     <uses name="f.b" link="output" transfer="false"
register="false"/>
   </job>

   <job id="j2" namespace="pegasus" name="world" version="4.0">
      <argument>-a world -T 60 -i  <file name="f.b"/>
            -o  <file name="f.c"/>
      </argument>
    <uses name="f.b" link="input" transfer="true"
register="true"/>
     <uses name="f.c" link="output" transfer="false"
register="false"/>
   </job>

<!-- Section: Dependencies - Parent Child relationships (can be
empty) -->

   <child ref="j2">
      <parent ref="j1"/>
   </child>

</adag>
```

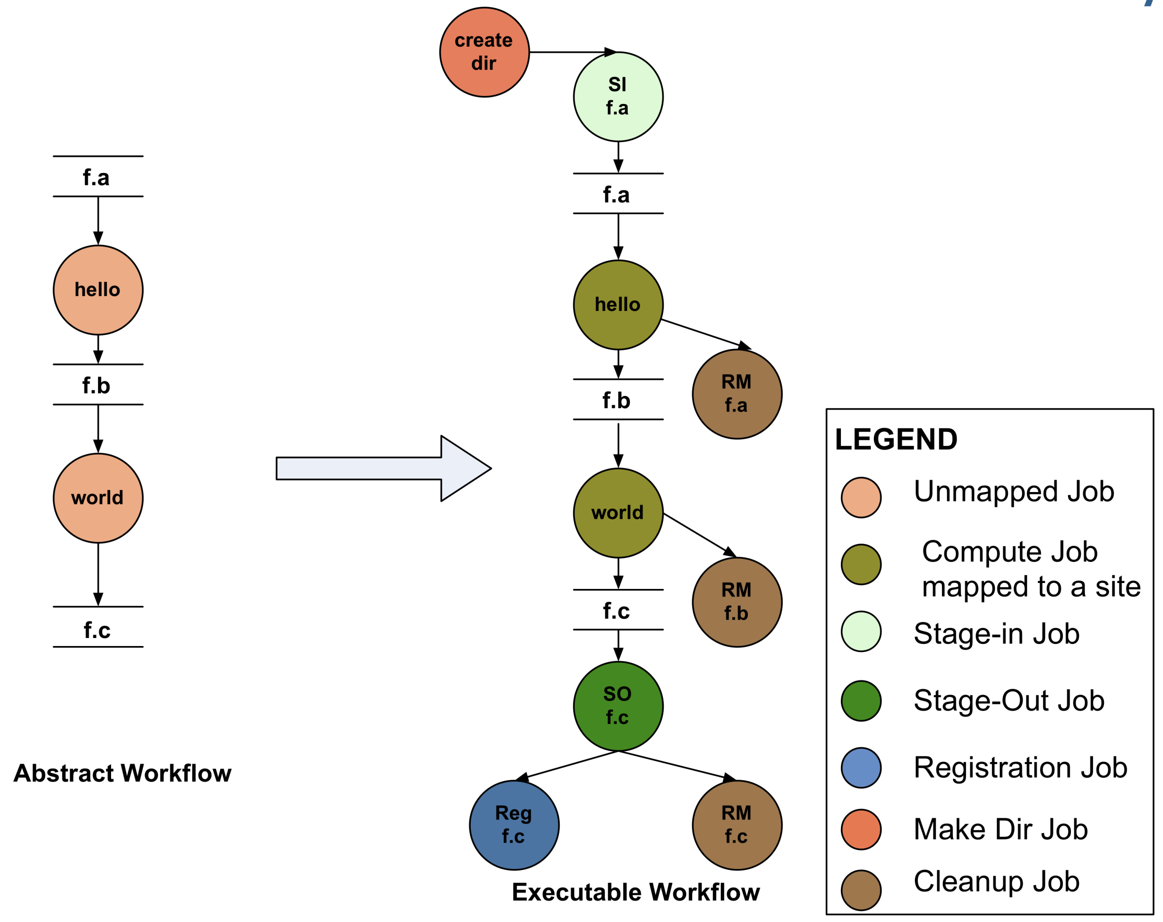
We provide Python, Java and Perl DAX API's!

Abstract to Executable Workflow Mapping - Discovery

- **Data**
  - Where do the input datasets reside?
- **Executables**
  - Where are the executables installed ?
  - Do binaries exist somewhere that can be staged to remote grid sites?
- **Site Layout**
  - What does a execution site look like?



Abstract to Executable Workflow Mapping



LEGEND

●	Unmapped Job
●	Compute Job mapped to a site
●	Stage-in Job
●	Stage-Out Job
●	Registration Job
●	Make Dir Job
●	Cleanup Job

- **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 resources and/or across them)**
  - **Gives opportunities for optimization and fault tolerance**
    - **automatically restructure the workflow**
    - **automatically provide fault recovery (retry, choose different resource)**

Pegasus Guarantee - Wherever and whenever a job runs it's inputs will be in the directory where it is launched.

Simple Steps to Run Pegasus

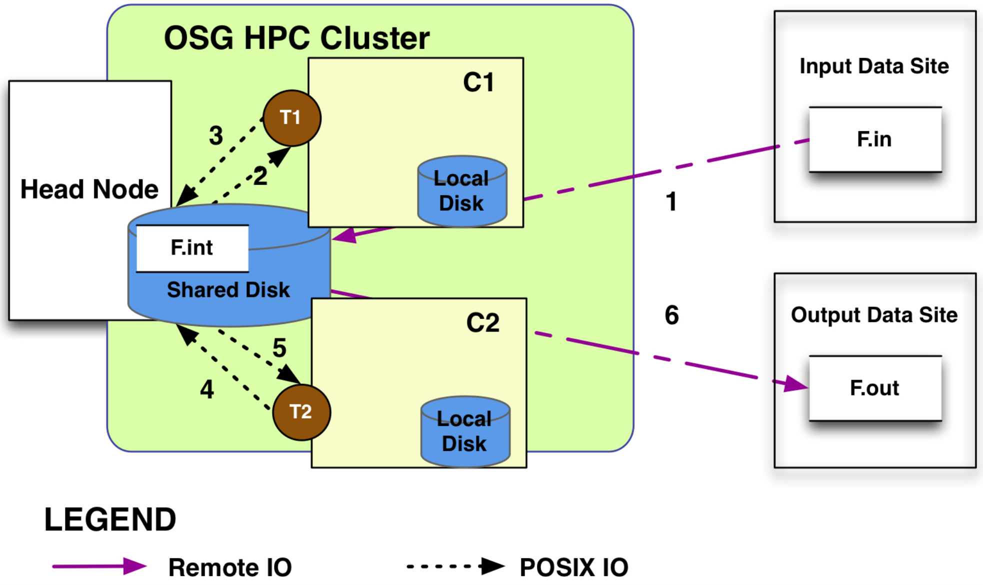
1. **Specify your computation in terms of DAX**
   - Write a simple DAX generator
   - Python, Java , Perl based API provided with Pegasus
2. **Set up your catalogs**
   - Replica catalog, transformation catalog and site catalog.
3. **Plan and Submit your workflow**
   - Use *pegasus-plan* to generate your executable workflow that is mapped onto the target resources and submits it for execution
4. **Monitor and Analyze your workflow**
   - Use *pegasus-status* | *pegasus-analyzer* to monitor the execution of your workflow
5. **Workflow Statistics**
   - Run pegasus-statistics to generate statistics about your workflow run.

Different Directories used by Pegasus

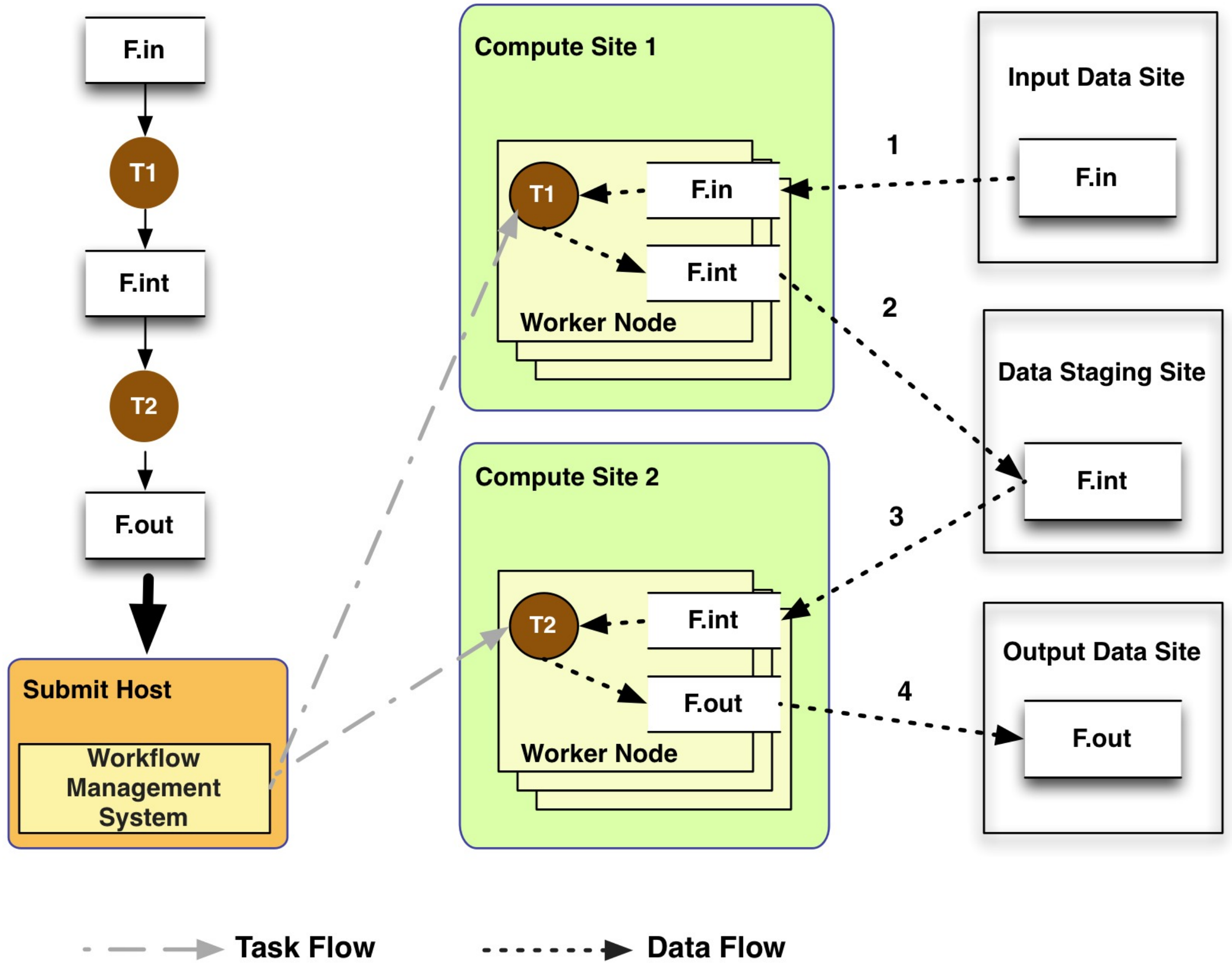
1. **Submit Directory**
   - The directory where pegasus-plan generates the executable workflow i.e HTCondor DAGMan and job submit files.
   - Specified by ***--dir*** option to pegasus-plan
2. **Input Directory**
   - Mostly input file locations are catalogued in the Replica Catalog.
   - However, if inputs are on the submit host (i.e. where Pegasus is installed), then you can pass ***–input-dir*** option to pegasus-plan
3. **Scratch Directory**
   - Workflow specific directory created by the **create-dir** job on the shared filesystem of HPCC. This is where all the jobs run.
   - The base directory specified in the site catalog entry for HPCC in ***sites.xml*** file.
4. **Output Directory**
   - The output directory where the outputs of the workflow appear.
   - Specified in the output site ("***local***") entry in the ***sites.xml*** file.
   - Can also be optionally specified by ***–output-dir*** option to pegasus-plan

How does Pegasus view a compute resource as?

- For Pegasus a compute resource or a site is associated with the following
  - An entry point or a scheduler contact to submit jobs to e.g PBS/LSF/Condor
  - File servers to stage data to the cluster
  - Different types of directories on the site
    - Shared-scratch - shared across all the worker nodes in the site
    - Local – a directory/filesystem local to the node where a job executes
  - Site wide information like environment variables to be set when a job is run.



General Workflow Execution Model



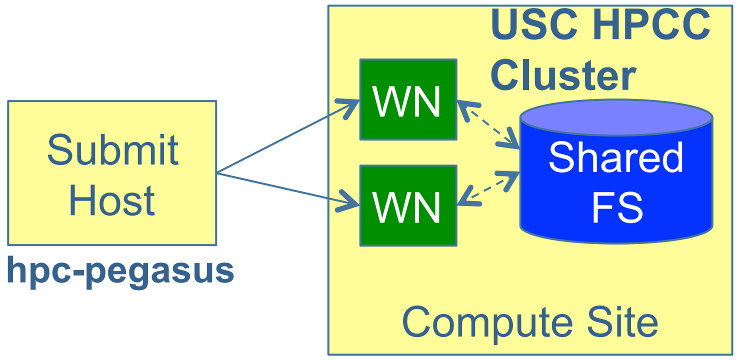
- Most of the tasks in scientific workflow applications require POSIX file semantics
  - Each task in the workflow opens one or more input files
  - Read or write a portion of it and then close the file.
- Data Staging Site can be the shared filesystem on the compute cluster!

- Input Data Site, Compute Site and Output Data Sites can be co-located
  - Example: Input data is already present on the compute site.

Supported Data Staging Approaches - I

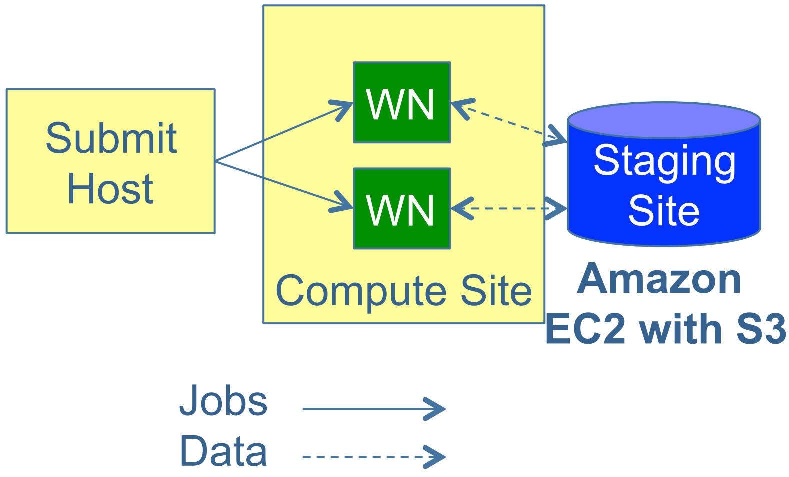
Shared Filesystem setup (typical of XSEDE and HPC sites)

- Worker nodes and the head node have a shared filesystem, usually a parallel filesystem with great I/O characteristics
- Can leverage symlinking against existing datasets
- Staging site is the shared-fs.



Non-shared filesystem setup with staging site (typical of OSG and EC 2)

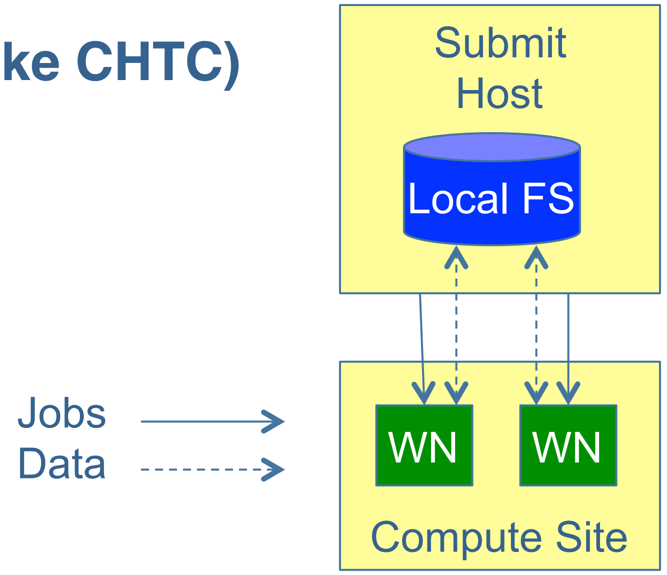
- Worker nodes don't share a filesystem.
- Data is pulled from / pushed to the existing storage element.
- A separate staging site such as S3.



Supported Data Staging Approaches - II

Condor IO (Typical of large Condor Pools like CHTC)

- Worker nodes don't share a filesystem
- Symlink against datasets available locally
- Data is pulled from / pushed to the submit host via Condor file transfers
- Staging site is the submit host.

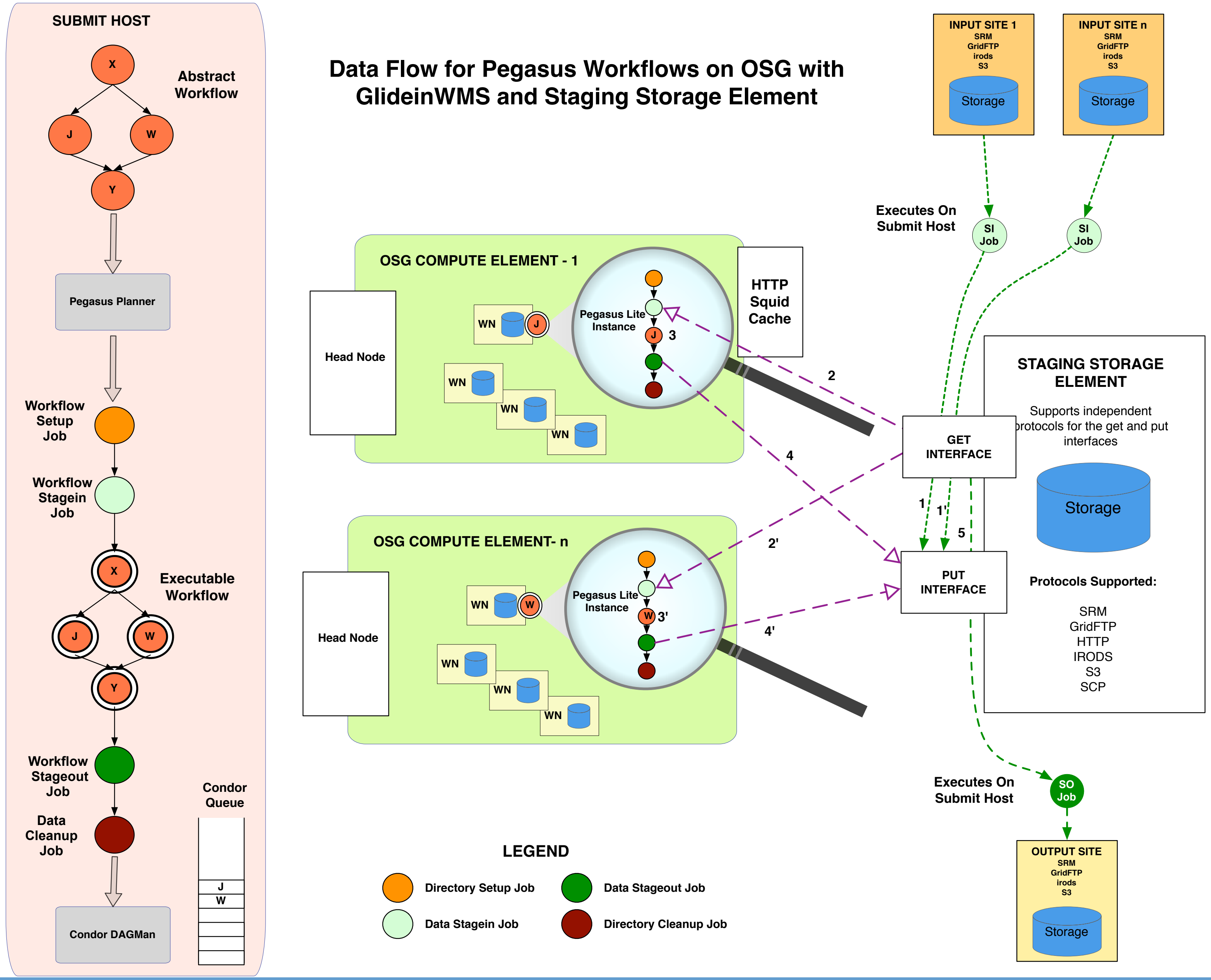


Supported Transfer Protocols

- HTTP
- SCP
- GridFTP
- IRODS
- S3
- Condor File IO
- File Copy

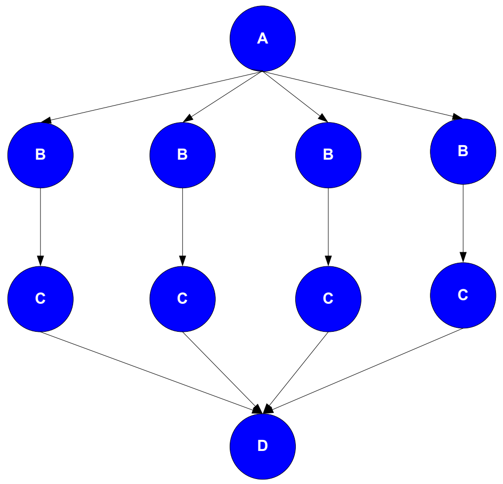
Using Pegasus allows you to move from one deployment to another without changing the workflow description!

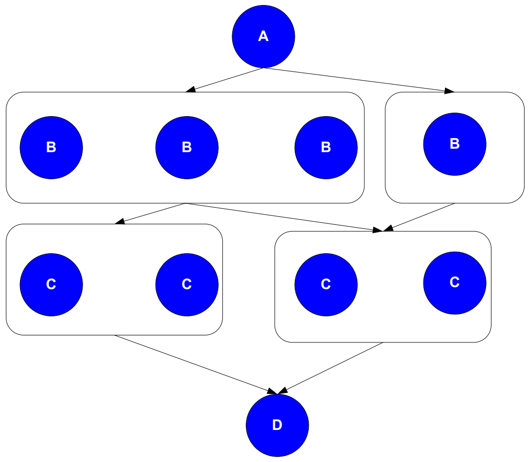
Data Flow for Pegasus Workflows on OSG with GlideinWMS and Staging Storage Element



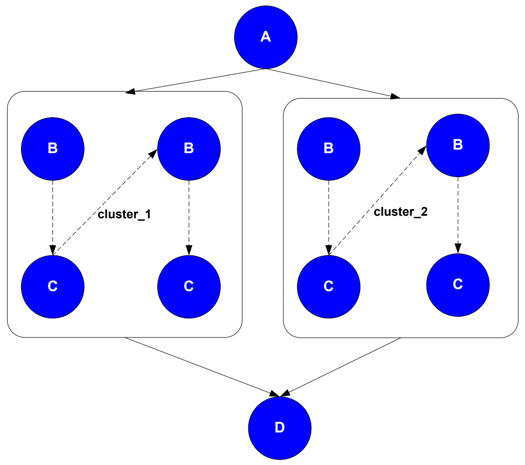
Workflow Restructuring to improve application performance

- **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 10/30/60/? minutes to execute
  - Clustered tasks can reuse common input data – less data transfers





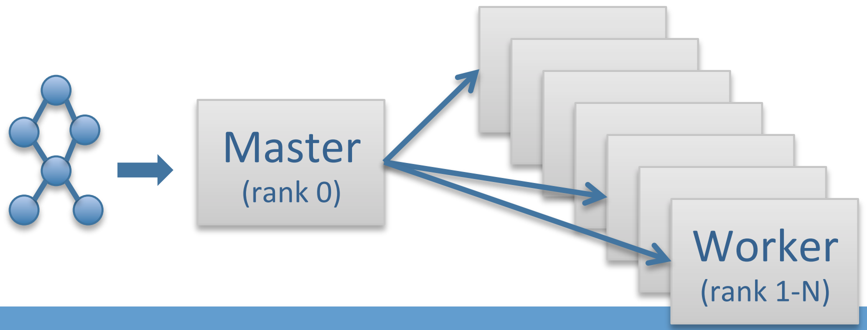
Horizontal clustering



Label-based clustering

Pegasus-MPI-Cluster

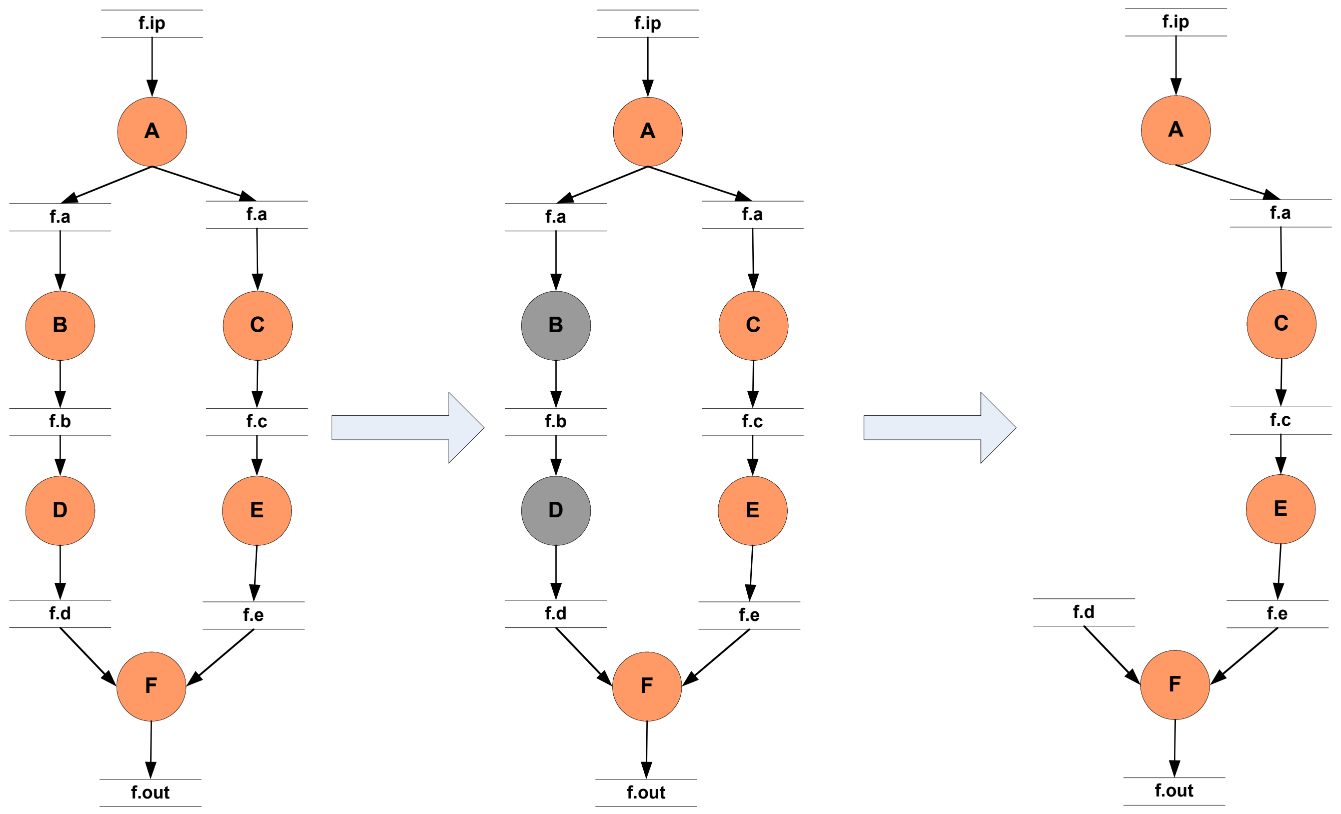
- A master/worker task scheduler for running fine-grained workflows on batch systems
- Runs as an MPI job
  - Uses MPI to implement master/worker protocol
- Works on most HPC systems
  - Requires: MPI, a shared file system, and fork()
- Allows sub-graphs of a Pegasus workflow to be submitted as monolithic jobs to remote resources



PMC Features

- **Fault Tolerance**
  - Retries at the task level (master resends task to another worker)
  - Retries at the workflow level (using a transaction log to record progress)
- **Resource-aware scheduling**
  - Many HPC machines have low memory/core
  - PMC can allocate memory and cores to a task, and force other slots on the same node to be idle
- **I/O Forwarding**
  - Small tasks == small I/O == poor performance
  - PMC reads data off of pipes from worker and forwards it using MPI messages to a central I/O process, which collects the data and writes it to disk
  - Writes are not interleaved, no locking required for synchronization

Workflow Reduction (Data Reuse)



Abstract Workflow

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

Delete Job D and Job B

Useful when you have done a part of computation and then realize the need to change the structure. Re-plan instead of submitting rescue DAG!

Data cleanup

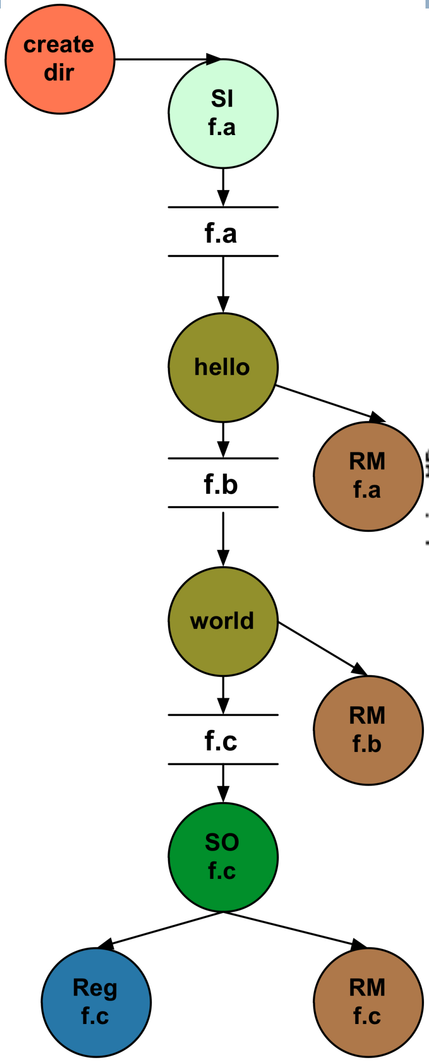
- **Problem: Running out of disk space during workflow execution**

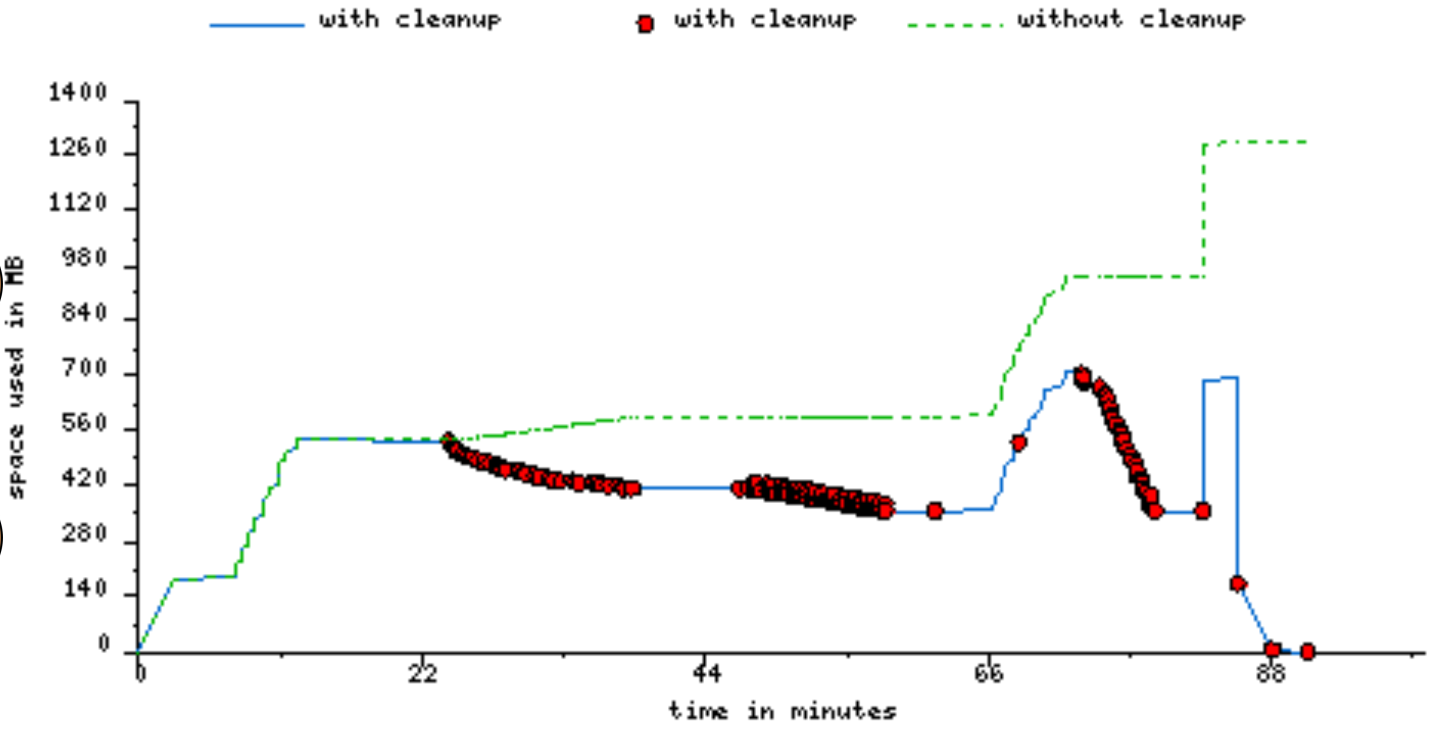
- **Why does it occur**
  - Workflows could bring in huge amounts of data
  - Data is generated during workflow execution
  - Users don't worry about cleaning up after they are done

- **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**

Real Life Example: Used by a UCLA genomics researcher to delete TB's of data automatically for long running workflows!!

Data cleanup (cont)

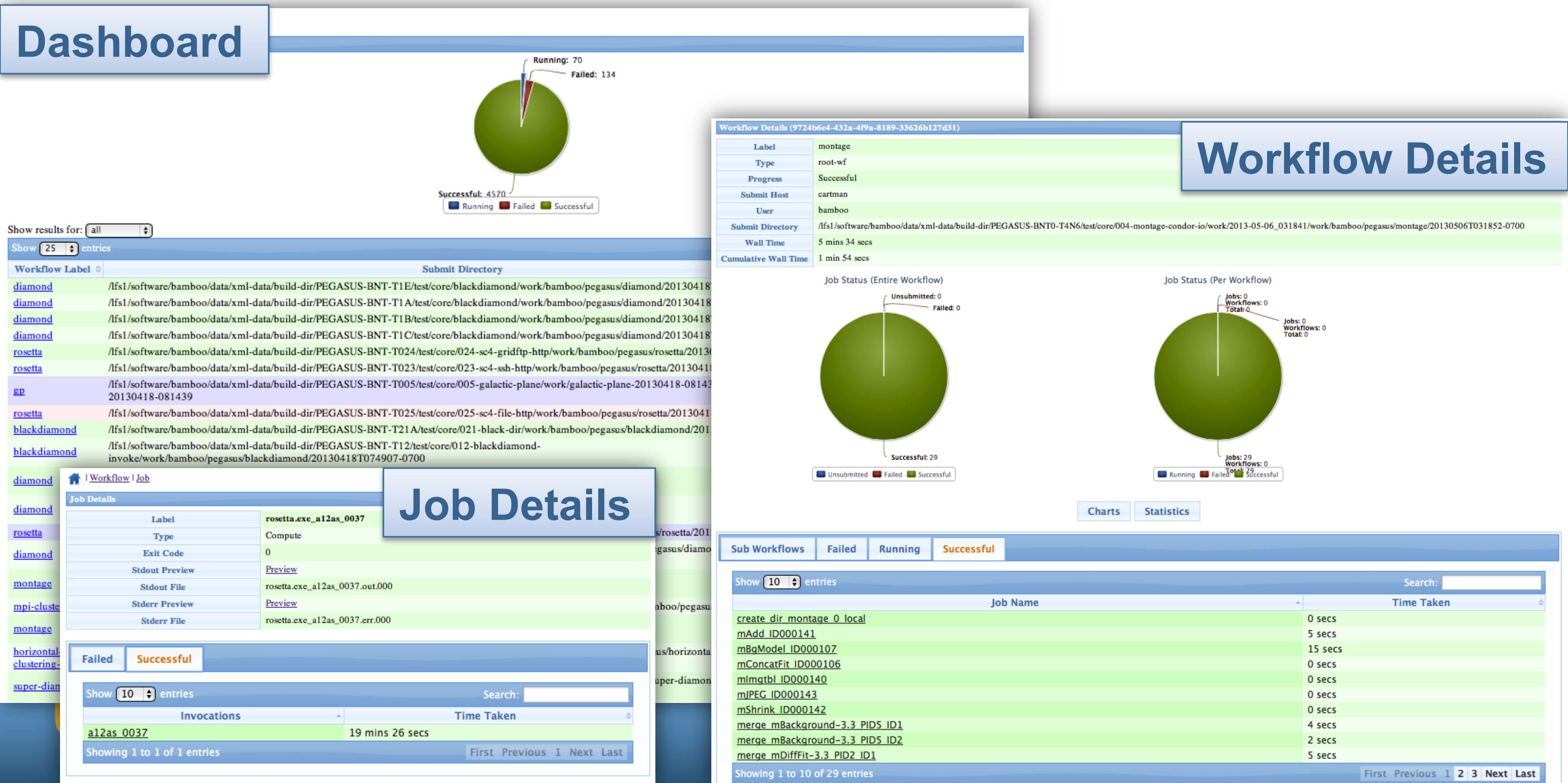




Montage 1 degree workflow run with cleanup

Pegasus Dashboard

- Web-based workflow monitoring GUI
  - Data comes from monitoring database
  - Supports monitoring, troubleshooting, and reporting



Show results for: all

Show 25 entries

Workflow Label	Submit Directory
diamond	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T1E/test/core/blackdiamond/work/bamboo/pegasus/diamond/20130418
diamond	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T1A/test/core/blackdiamond/work/bamboo/pegasus/diamond/20130418
diamond	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T1B/test/core/blackdiamond/work/bamboo/pegasus/diamond/20130418
diamond	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T1C/test/core/blackdiamond/work/bamboo/pegasus/diamond/20130418
rosetta	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T024/test/core/024-sc4-gridftp-http/work/bamboo/pegasus/rosetta/2013
rosetta	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T023/test/core/023-sc4-ssh-http/work/bamboo/pegasus/rosetta/201304
gp	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T005/test/core/005-galactic-plane/work/galactic-plane-20130418-0814 20130418-081439
rosetta	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T025/test/core/025-sc4-file-http/work/bamboo/pegasus/rosetta/201304
blackdiamond	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T21A/test/core/021-black-dir/work/bamboo/pegasus/blackdiamond/201
blackdiamond	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT-T12/test/core/012-blackdiamond-invoke/work/bamboo/pegasus/blackdiamond/20130418T074907-0700
diamond	
diamond	
rosetta	/rosetta/201
diamond	gasus/diamo
montage	
mpi-cluste	nboo/pegasu
montage	
horizontal-clustering-	us/horizonta
super-dian	uper-diamon

Workflow Details (9724b6e4-432a-4f9a-8189-33626b127d31)

Label	montage
Type	root-wf
Progress	Successful
Submit Host	cartman
User	bamboo
Submit Directory	/lfs1/software/bamboo/data/xml-data/build-dir/PEGASUS-BNT0-T4N6/test/core/004-montage-condor-io/work/2013-05-06_031841/work/bamboo/pegasus/montage/20130506T031852-0700
Wall Time	5 mins 34 secs
Cumulative Wall Time	1 min 54 secs

Sub Workflows | Failed | Running | Successful

Show 10 entries   Search:

Job Name	Time Taken
create_dir_montage_0_local	0 secs
mAdd_ID000141	5 secs
mBgModel_ID000107	15 secs
mConcatFit_ID000106	0 secs
mImgtbl_ID000140	0 secs
mJPEG_ID000143	0 secs
mShrink_ID000142	0 secs
merge_mBackground-3.3_PID5_ID1	4 secs
merge_mBackground-3.3_PID5_ID2	2 secs
merge_mDiffFit-3.3_PID2_ID1	5 secs

Showing 1 to 10 of 29 entries   First Previous 1 2 3 Next Last

| Workflow | Job

Job Details

Label	rosetta.exe_a12as_0037
Type	Compute
Exit Code	0
Stdout Preview	Preview
Stdout File	rosetta.exe_a12as_0037.out.000
Stderr Preview	Preview
Stderr File	rosetta.exe_a12as_0037.err.000

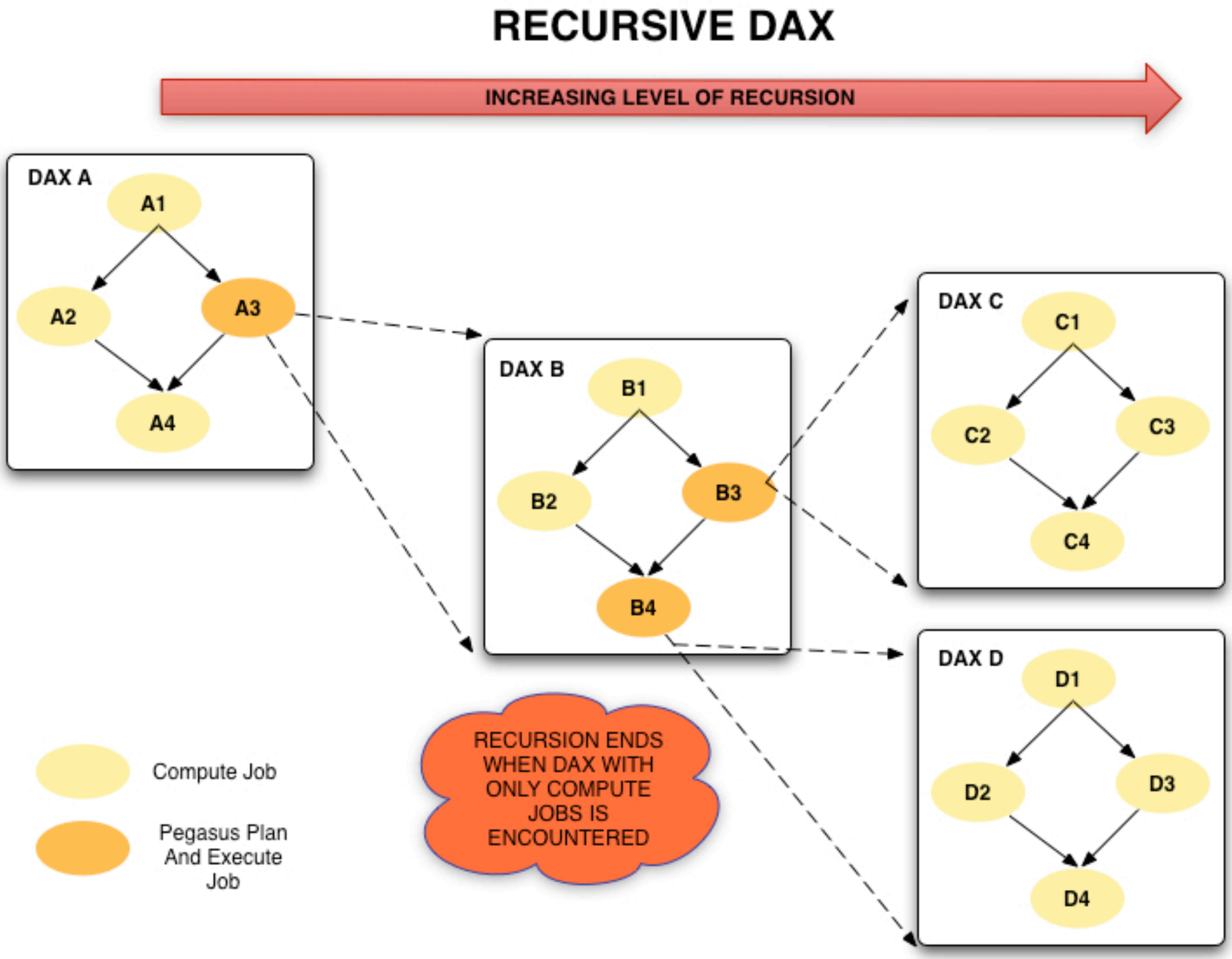
Failed | Successful

Show 10 entries   Search:

Invocations	Time Taken
a12as_0037	19 mins 26 secs

Showing 1 to 1 of 1 entries   First Previous 1 Next Last

Hierarchical Workflows


RECURSIVE DAX
INCREASING LEVEL OF RECURSION
DAX A
A1
A2
A3
A4
DAX B
B1
B2
B3
B4
DAX C
C1
C2
C3
C4
DAX D
D1
D2
D3
D4
RECURSION ENDS WHEN DAX WITH ONLY COMPUTE JOBS IS ENCOUNTERED
Compute Job
Pegasus Plan And Execute Job

Example Hierarchical Workflow

- <dax> element behaves like <job>
  - Arguments are for pegasus-plan (most are inherited)
- Planner is invoked when DAX job is ready to run

```
<?xml version="1.0" encoding="UTF-8"?>
<adag version="3.4" name="multi-level">
    <job id="ID0000001" namespace="example" name="sleep">
        <argument>5</argument>
    </job>
    <dax id="ID0000002" file="sub.dax">
        <argument>--output-site local</argument>
    </dax>
    <job id="ID0000003" namespace="example" name="sleep">
        <argument>5</argument>
    </job>
    <child ref="ID0000002">
        <parent ref="ID0000001"/>
    </child>
    <child ref="ID0000003">
        <parent ref="ID0000002"/>
    </child>
</adag>
```

What Does Pegasus provide an Application - I

- **Portability / Reuse**
  - User created workflows can easily be mapped to and run in different environments without alteration.

- **Data Management**
  - Pegasus handles replica selection, data transfers and output registrations in data catalogs. These tasks are added to a workflow as auxiliary jobs by the Pegasus planner.

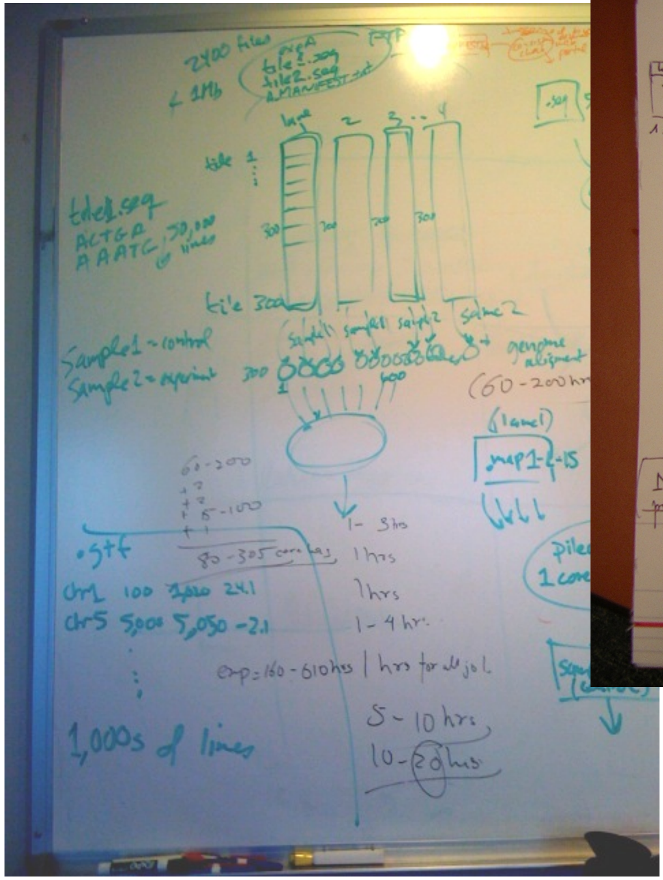
- **Performance**
  - The Pegasus mapper can reorder, group, and prioritize tasks in order to increase the overall workflow performance.

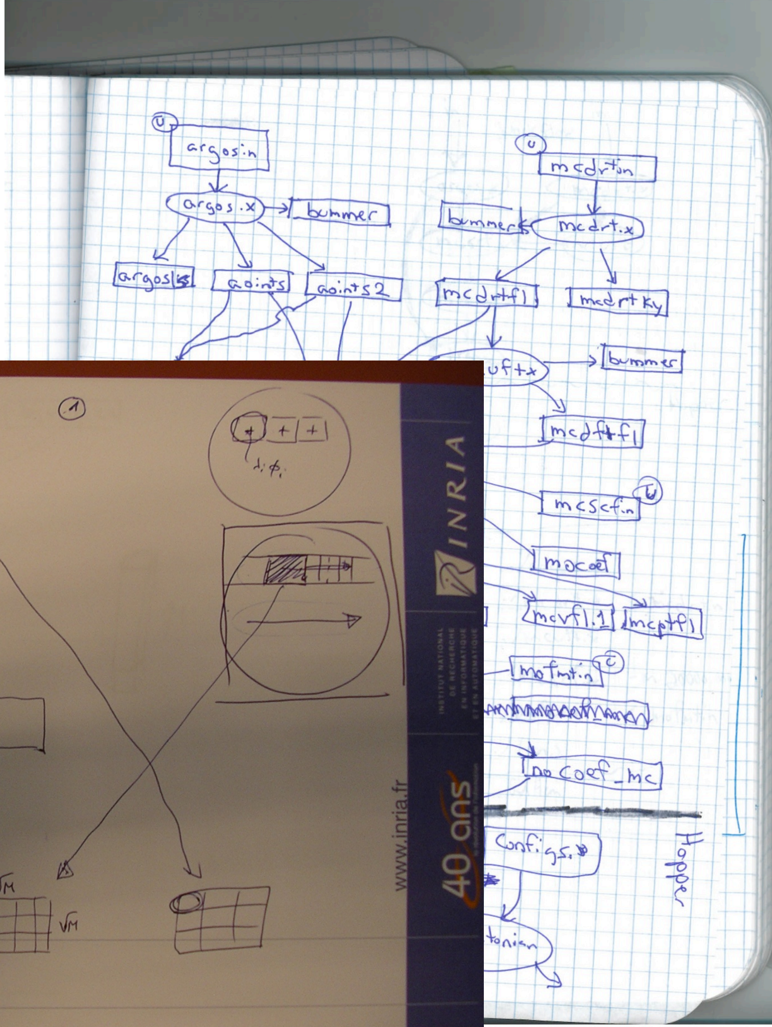
What Does Pegasus provide an Application - II

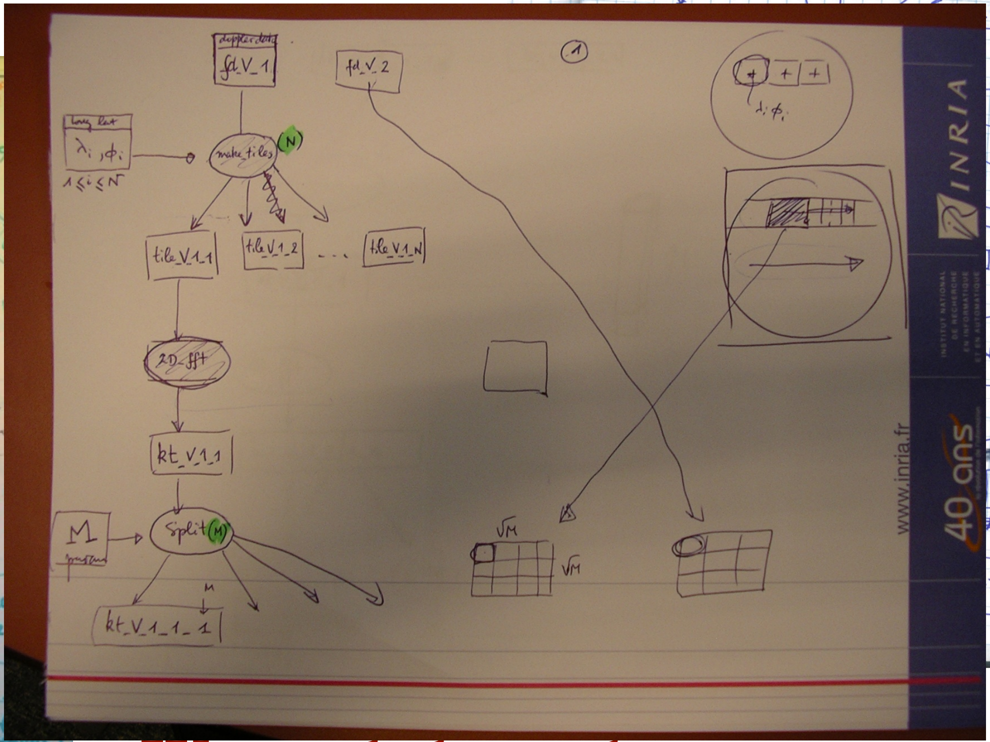
- **Provenance**
  - Provenance data is collected in a database, and the data can be summaries with tools such as pegasus-statistics, pegasus-plots, or directly with SQL queries.
- **Reliability and Debugging Tools**
  - Jobs and data transfers are automatically retried in case of failures. Debugging tools such as pegasus-analyzer helps the user to debug the workflow in case of non-recoverable failures.
- **Scalability**
  - Hierarchal workflows
  - Scale to hundreds of thousands of nodes in a workflow.

If you get stuck…

And you can draw….







We can help you!

More Information

- Pegasus Website:
  - http://pegasus.isi.edu
- Tutorial:
  - http://pegasus.isi.edu/wms/docs/latest/tutorial.php
- Documentation:
  - http://pegasus.isi.edu/documentation
- Email addresses:
  - Pegasus users list (public): pegasus-users@isi.edu
  - Pegasus support (private): pegasus-support@isi.edu