Pegasus WMS: Before Tutorial Starts

- **Presentation handouts**

  /home/rcf-proj/workshop/handouts/HPCPegasusWorkflows*

  /home/rcf-proj/workshop/handouts/HPCPegasusWorkflowsExample*

- **Tutorial page**

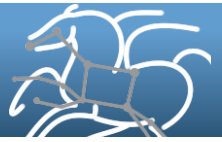
  https://pegasus.isi.edu/tutorial/usc
  https://pegasus.isi.edu/documentation/pegasus-user-guide.pdf

- **To do exercises logon to hpc-pegasus.usc.edu**

  ssh <USCNetId>@hpc-pegasus.usc.edu

- **In case of problems, we have people to help you out**

  Karan Vahi, Rajiv Mayani, Cesar Sul and Erin Shaw



Scientific Workflows through Pegasus WMS on the USC HPC Cluster

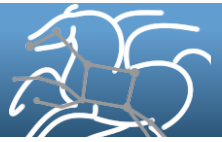
Karan Vahi, Mats Rynge

Science Automation Technologies Group

USC Information Sciences Institute

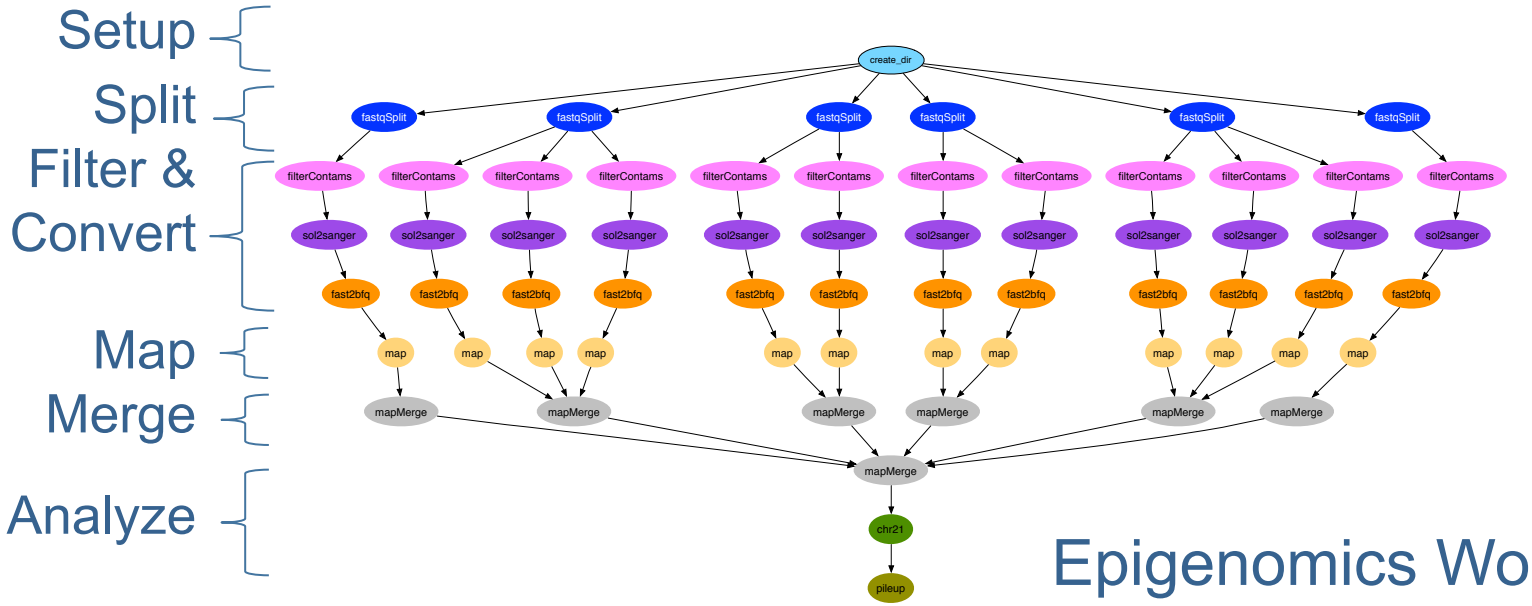
Agenda

- 2:30 – 3:00 Introduction on Workflows and Pegasus
- 3:00 – 4:00 Hands on Exercises
- 4:00 – 4:30 Dashboard and Monitoring



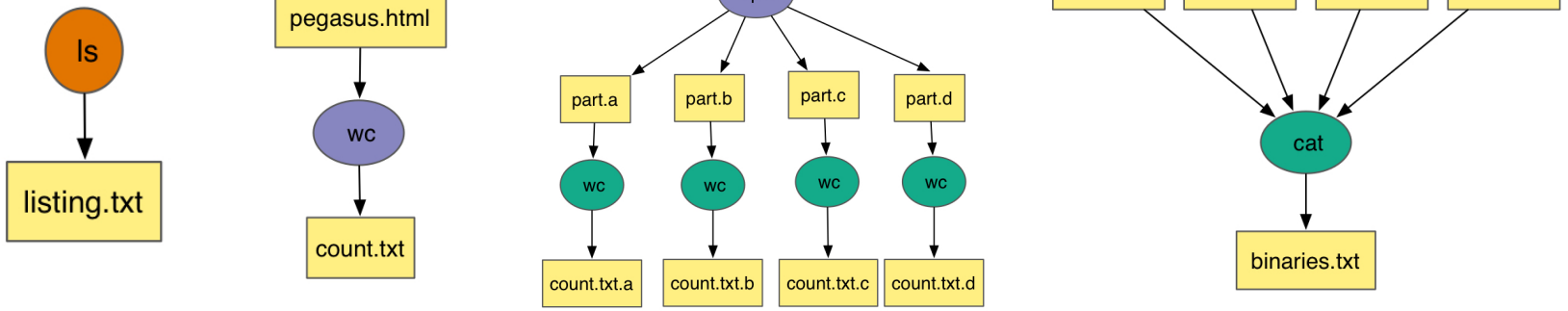
Scientific Workflows

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

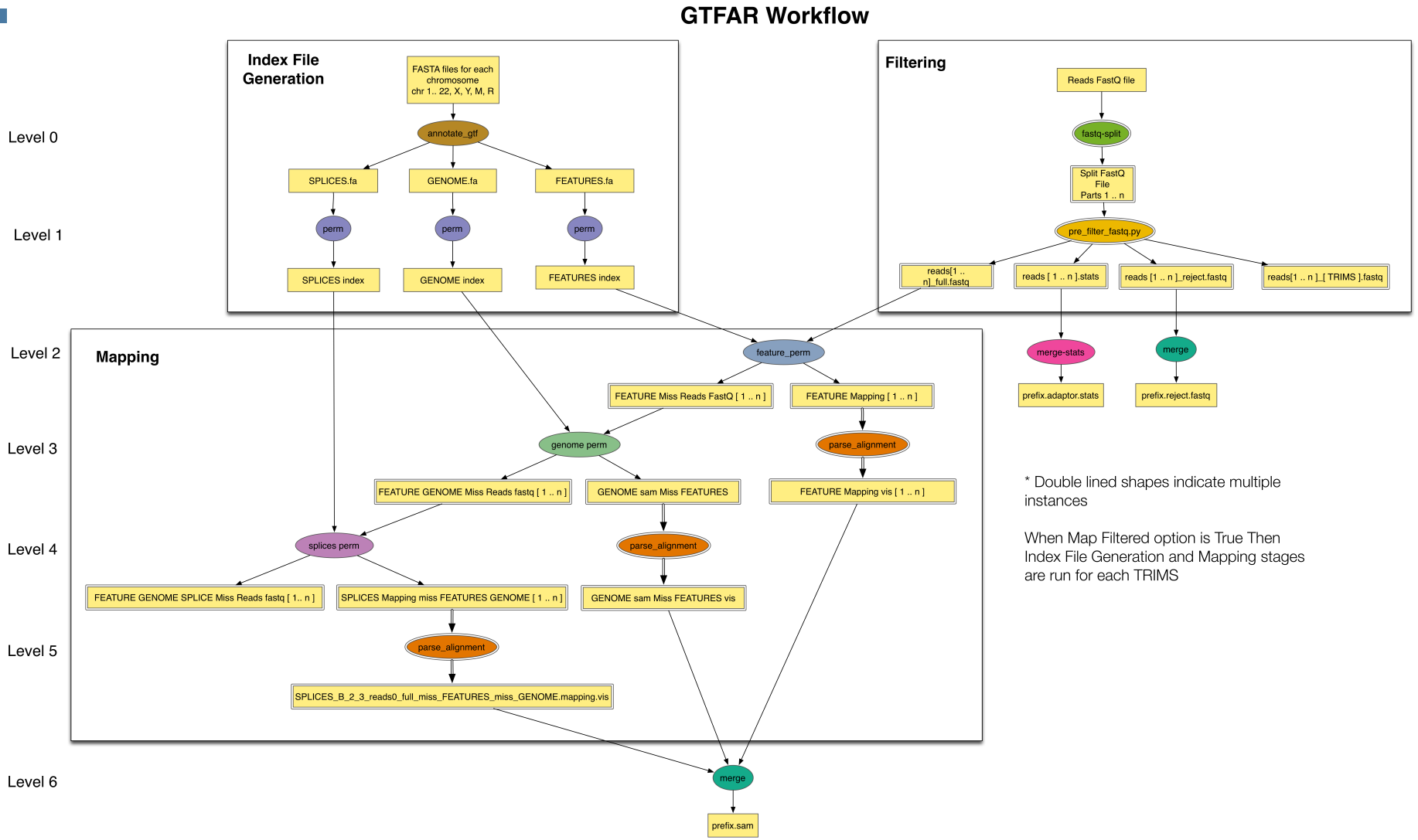


Epigenomics Workflow

Simple Workflows – Building Blocks



Complex RNA-Sequencing Workflow - GTFAR



* Double lined shapes indicate multiple instances

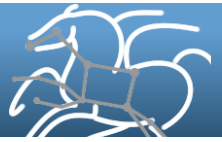
When Map Filtered option is True Then Index File Generation and Mapping stages are run for each TRIMS

Challenges while Executing Compute Pipelines

- **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? 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**

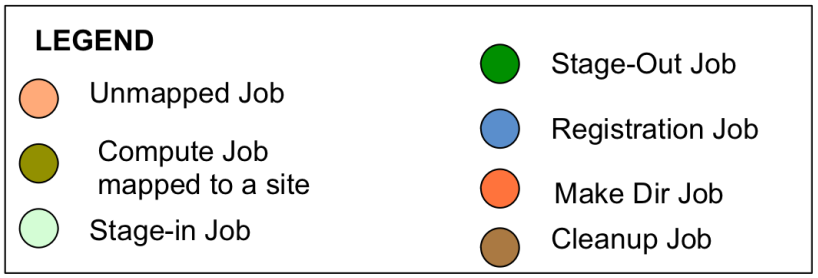
Pegasus Workflow Management System

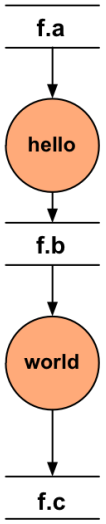
- **NSF funded project since 2001**
  - Developed as a collaboration between USC Information Sciences Institute and the HTCondor Team at UW Madison
- **Builds on top of HTCondor DAGMan**
  - **HTCondor** is a specialized workload management system for compute-intensive jobs. **DAGMan** (Directed Acyclic Graph Manager) is a meta-scheduler for **HTCondor**
- **Pegasus is a workflow "compiler"**
  - Does planning and mapping
  - Targets are DAGMan DAGs and Condor submit files



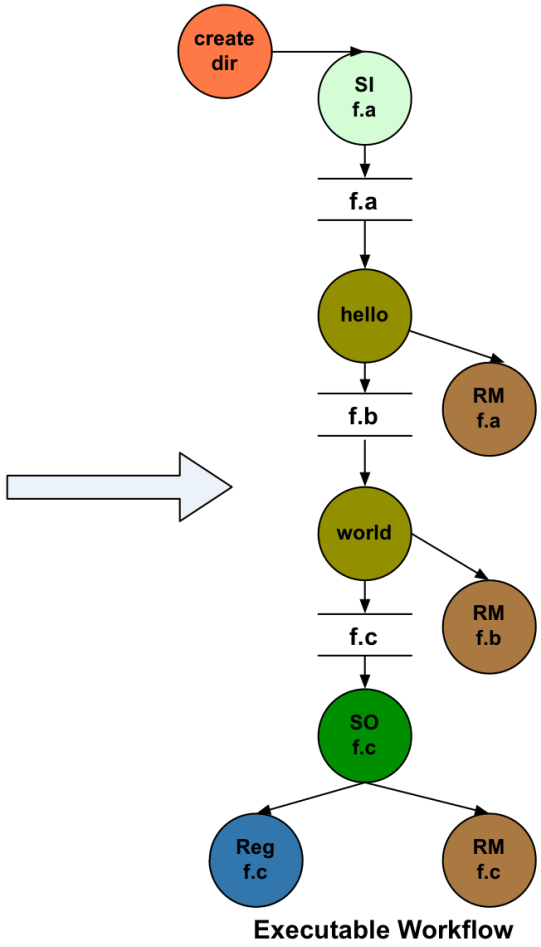
Pegasus Workflow Management System

- **Abstracts workflows**
  - Workflow describe in "high-level language"
- **Compiles workflows**
  - Maps a resource-independent "abstract" workflow onto resources and executes the "executable" workflow





Abstract Workflow

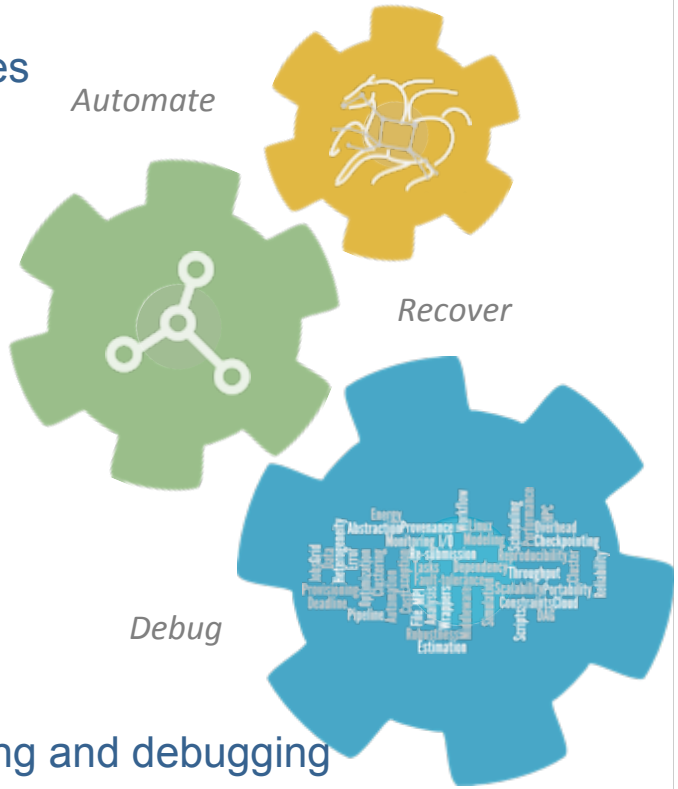


Executable Workflow

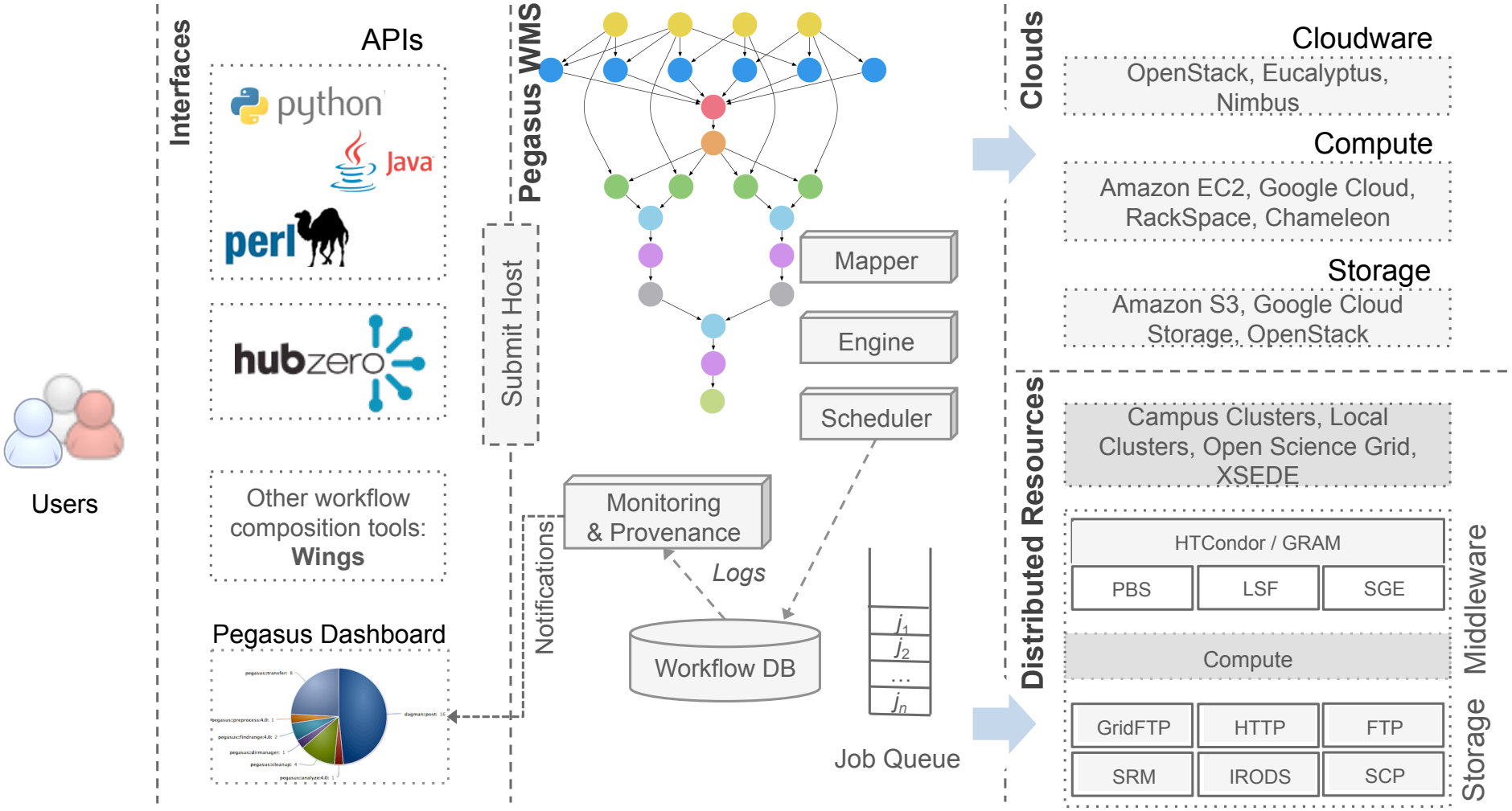
Pegasus Workflow Management System

Benefits

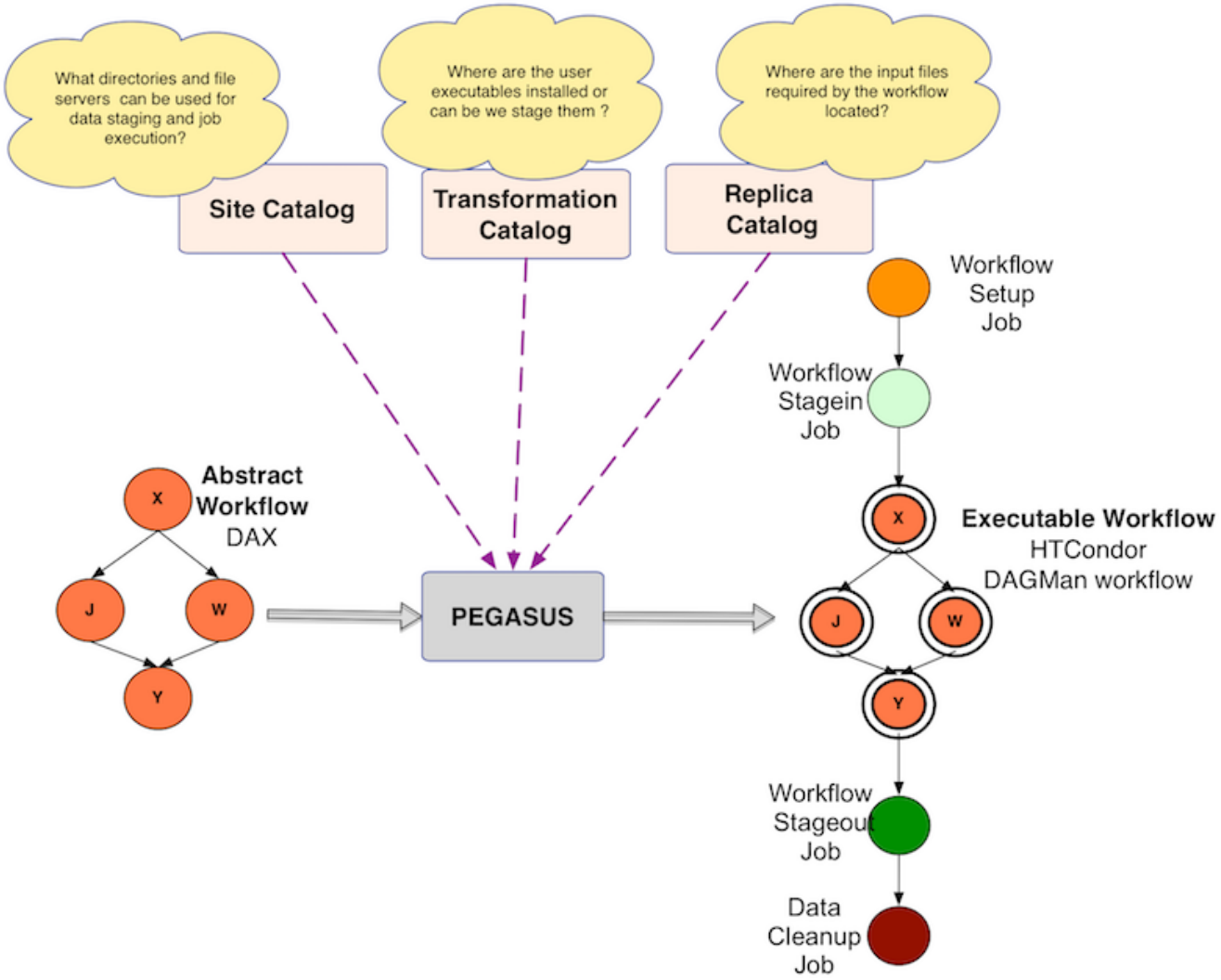
- **Automation**
  - Automates complex, multi-stage processing pipelines
  - Enables parallel, distributed computations
  - Automatically executes data transfers
  - Reusable portable description of workflows that aids reproducibility
- **Recovery**
  - Automatic job retries
  - Ability to restart pipelines without repeating steps
- **Debug**
  - Records how data was produced (provenance)
  - Simple easy to use command line tools for monitoring and debugging
  - Web Based Dashboard Available



Pegasus Deployment



Abstract to Executable Workflow Mapping - Discovery

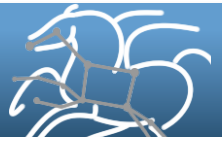


Simple Steps to Run Pegasus

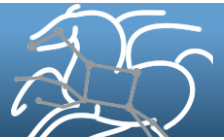
1. **Specify your computation in terms of DAX**
   - Write a simple DAX generator
   - Python, Java , Perl based API provided with Pegasus
2. **Set up your catalogs**
   - Replica catalog, transformation catalog and site catalog.
3. **Plan and Submit your workflow**
   - Use *pegasus-plan* to generate 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.

Agenda

- 2:30 – 3:00 Introduction and Tutorial on Pegasus
- 3:00 – 4:00 Hands on Exercises
- 4:00 – 4:30 Dashboard and Monitoring

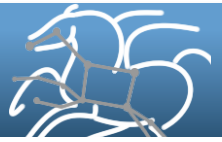


Move to Hands on Tutorial



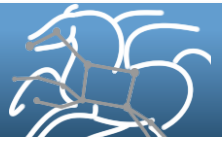
Hands on Tutorial Outline

- **Topics Covered**
  - What are Scientific Workflows (already done)
  - Submission of an already generated example workflow with Pegasus.
  - How to use the Pegasus Workflow Dashboard for monitoring workflows.
  - Command line tools for monitoring, debugging and generating statistics.
  - Recovery from failures
  - Creation of workflow using system provided API
  - Information catalogs configuration
  - Running the whole workflow as a MPI job

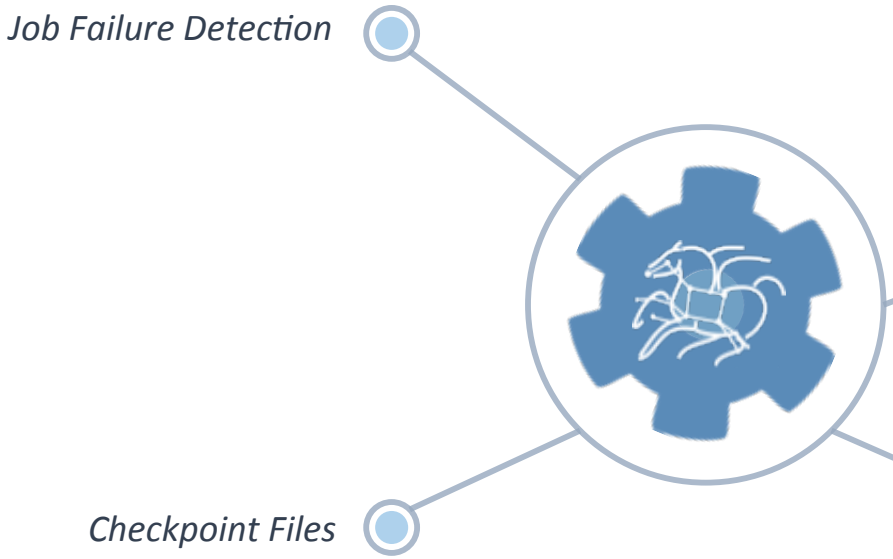


Agenda

- 2:30 – 3:00 Introductions on Workflows and Pegasus
- 3:00 – 4:00 Hands on Exercises
- 4:00 – 4:30 Features addressing user problems



Failure Recovery



Job Failure Detection

Job Retry

helps with transient failures
set number of retries per job and run

Checkpoint Files

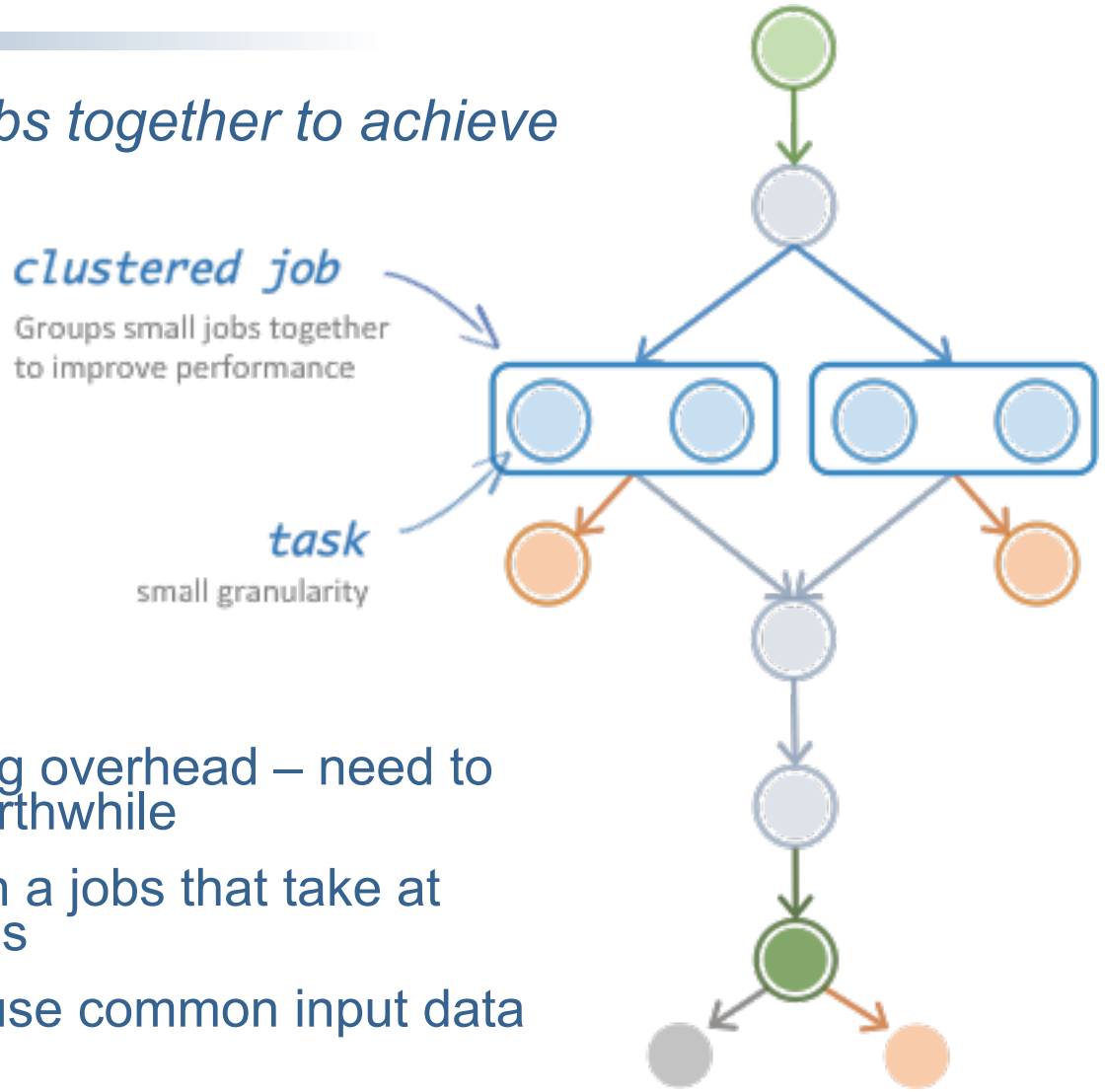
job generates checkpoint files
staging of checkpoint files is
automatic on restarts

Rescue DAGs

workflow can be restarted from checkpoint file
recover from failures with minimal loss

Task Clustering

Cluster small running jobs together to achieve better performance



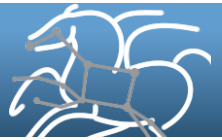
Why?

- Each job has scheduling overhead – need to make this overhead worthwhile
- Ideally users should run a jobs that take at least 10/30/60/? minutes
- Clustered tasks can reuse common input data – less data transfers

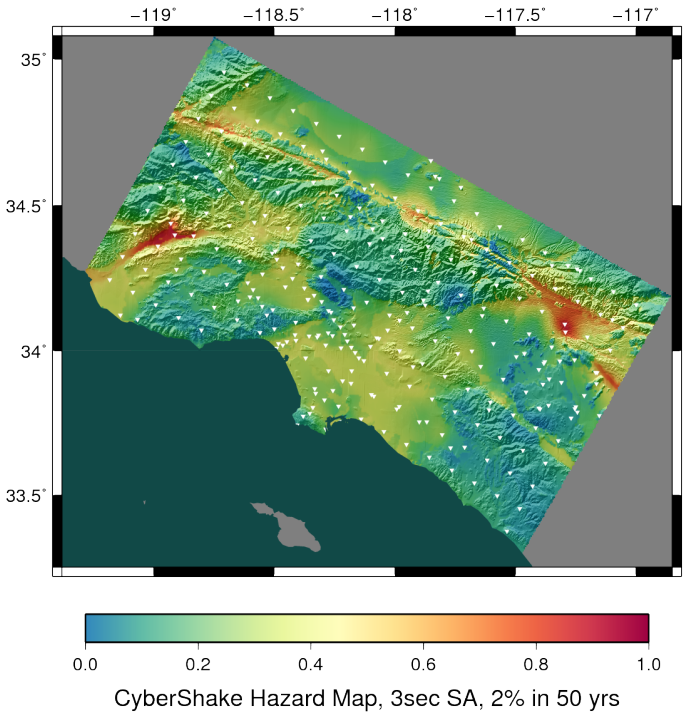
Fine-Grained Workflows

- **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



Fine Grained Workflows



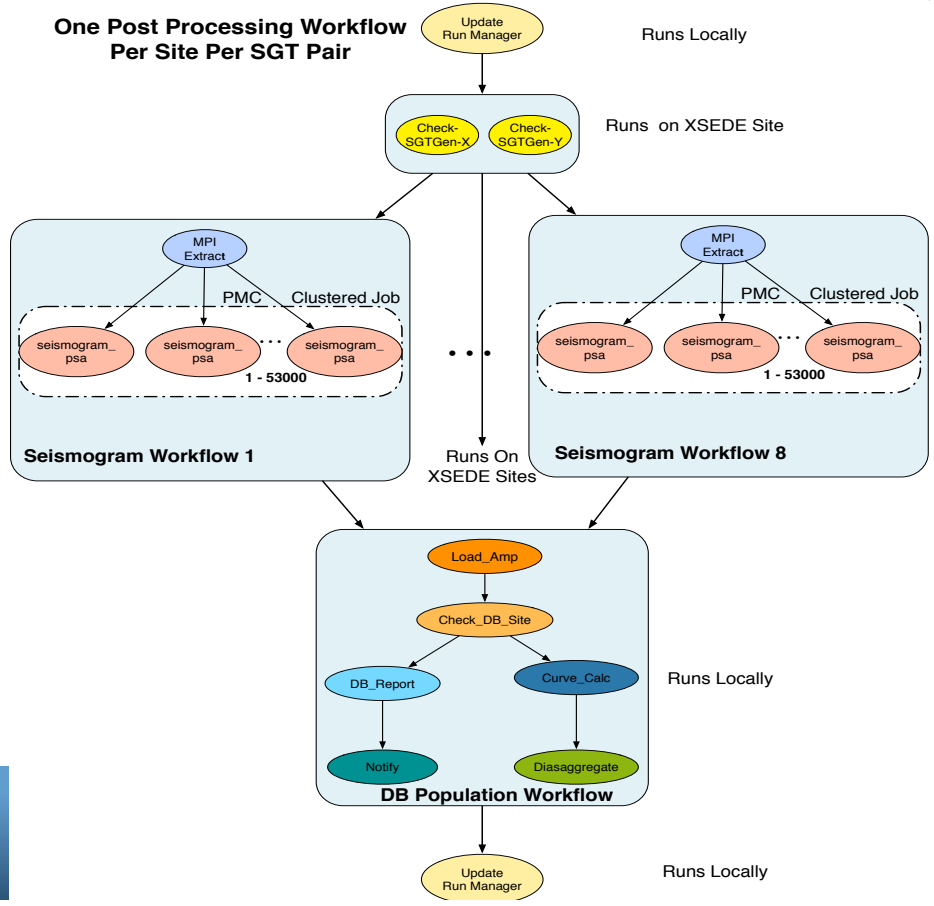
CyberShake Hazard Map, 3sec SA, 2% in 50 yrs

2014: 286 Sites, 4 models

- Each site = one workflow
- Each workflow has 420,000 tasks in 21 jobs

CyberShake PSHA Workflow

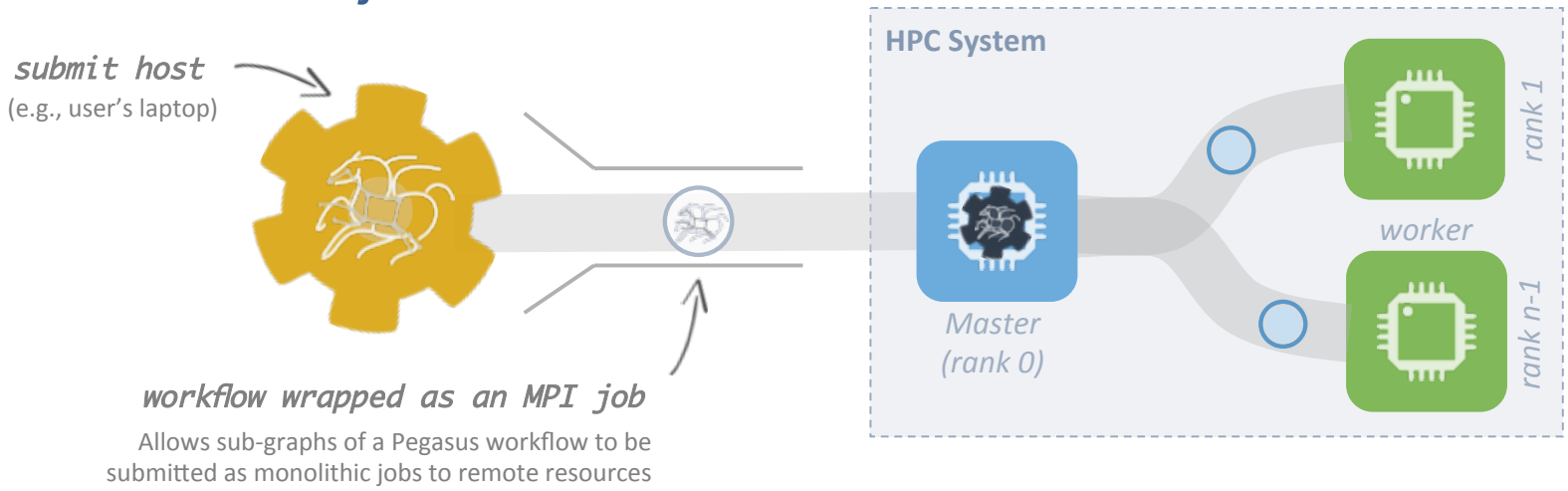
- Builders ask seismologists: "What will the peak ground motion be at my new building in the next 50 years?"
- Seismologists answer this question using Probabilistic Seismic Hazard Analysis (PSHA)



Fine Grained Workflows

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

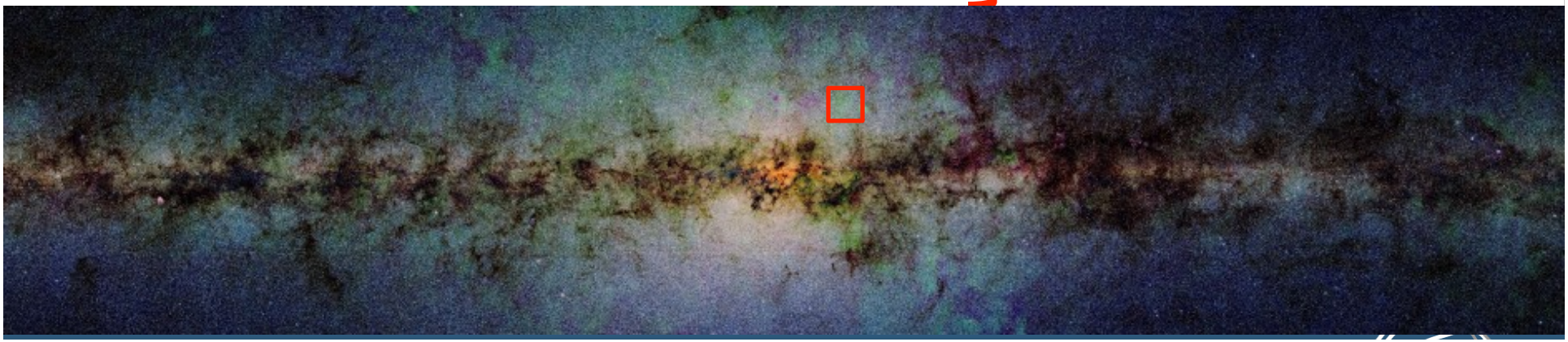


Connecting Pipelines

- **Problem**
  - For large scale analysis, scientists often want to compose larger constructs from smaller working pipelines.
  - Groups from the same domain collaborate and want to connect pipelines as part of larger analysis
  - Also want to scale to workflows with millions of tasks in total.

- **Example Application Use Case - Montage Galactic Plane Workflow**
  - Existing montage workflow that can generate 5 degree mosiacs
  - 18 million input images (~2.5 TB)
  - 900 output images (2.5 GB each, 2.4 TB total)
  - 10.5 million tasks (34,000 CPU hours)

× 17

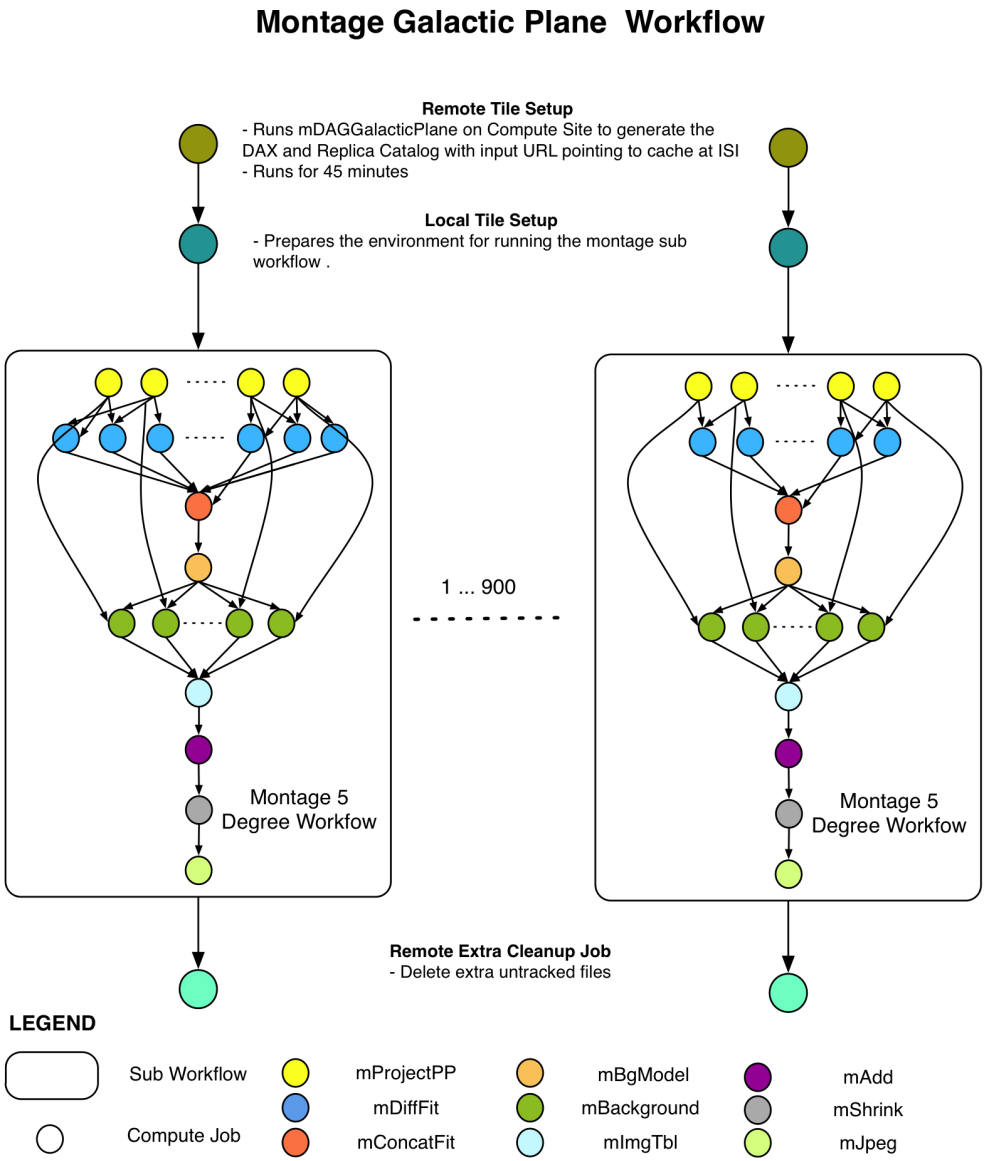


Connecting Pipelines

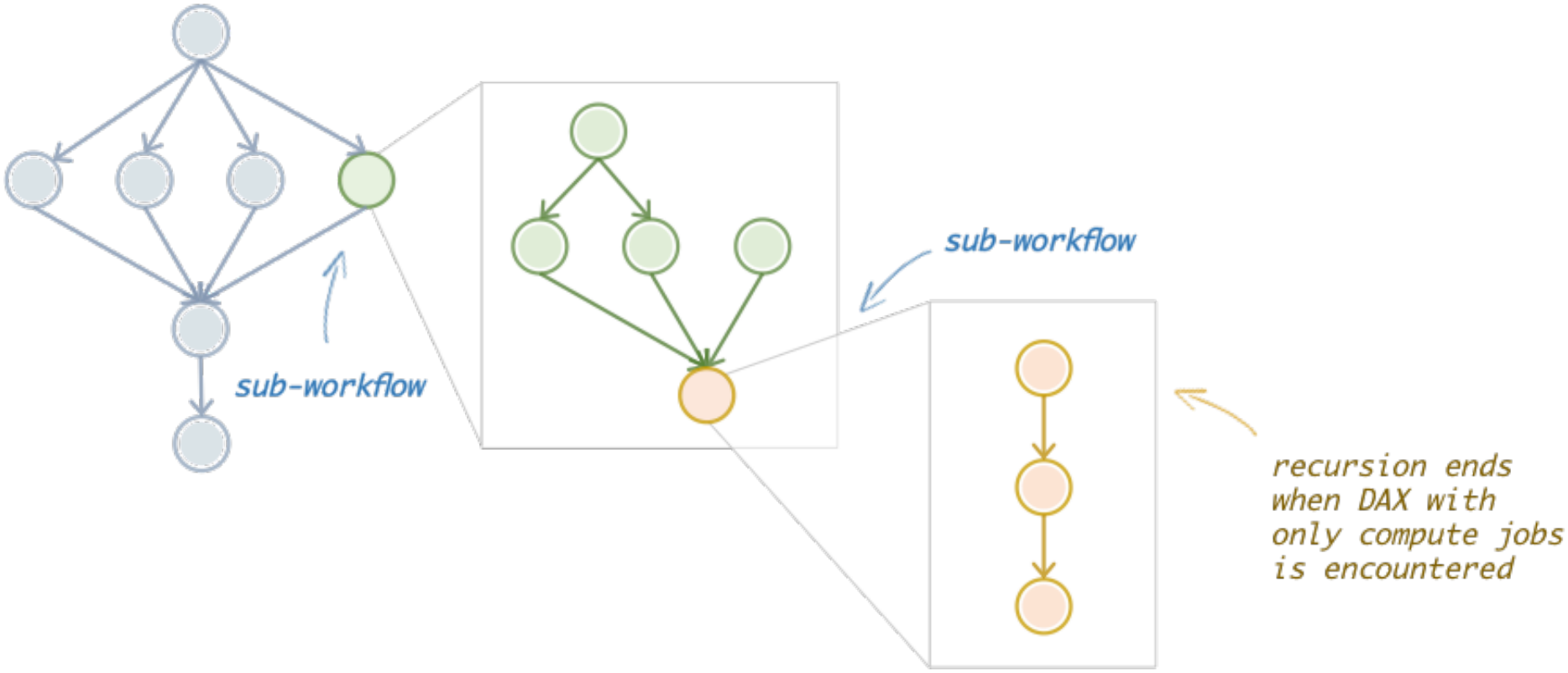
Solution

- Pegasus allows jobs/nodes in a workflow to refer to other workflows, allowing users to connect distinct workflows for analysis.

- Updated our debugging tools to handle these workflow of workflows.



Hierarchical Workflows



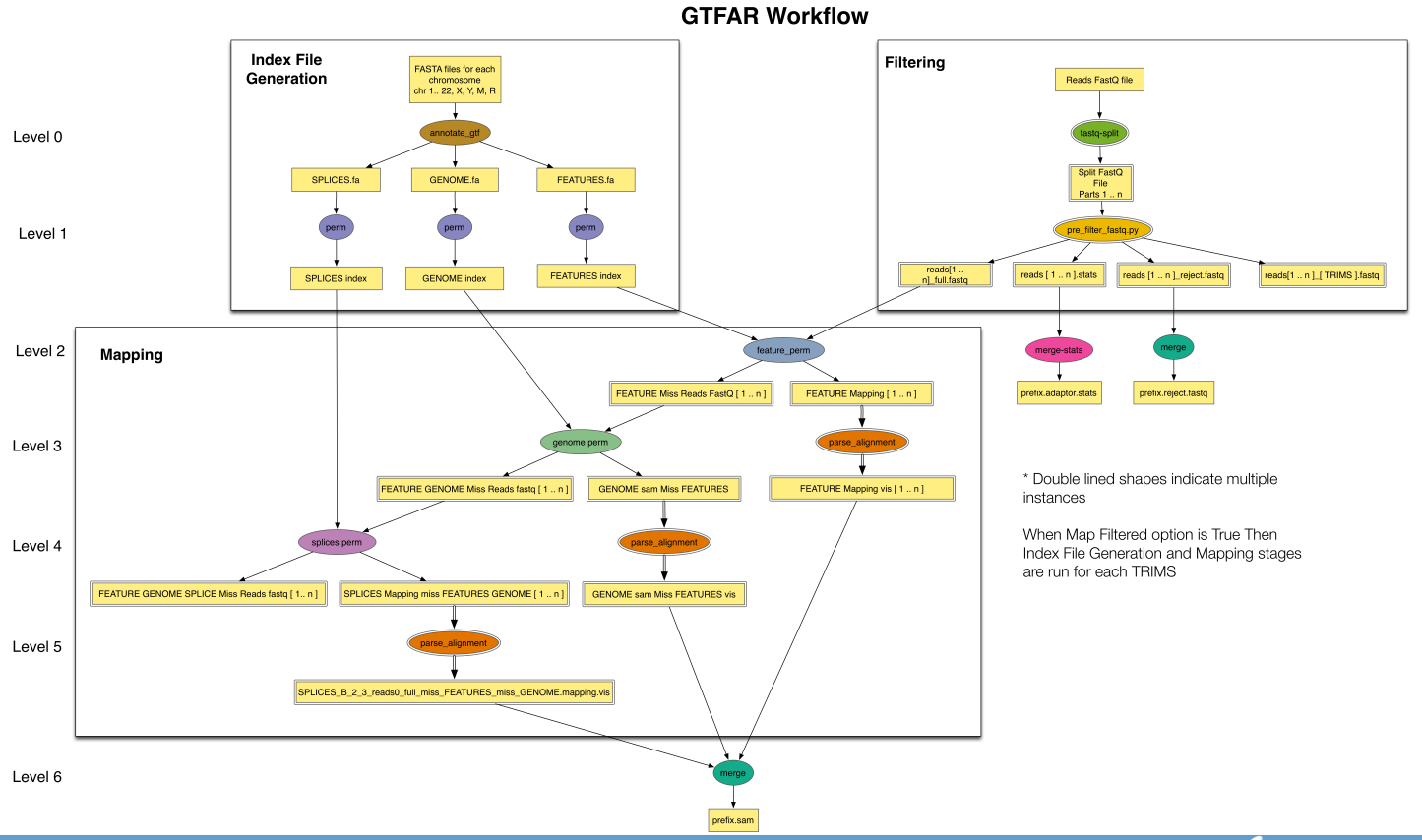
Reusing Data Products

- **Problem**

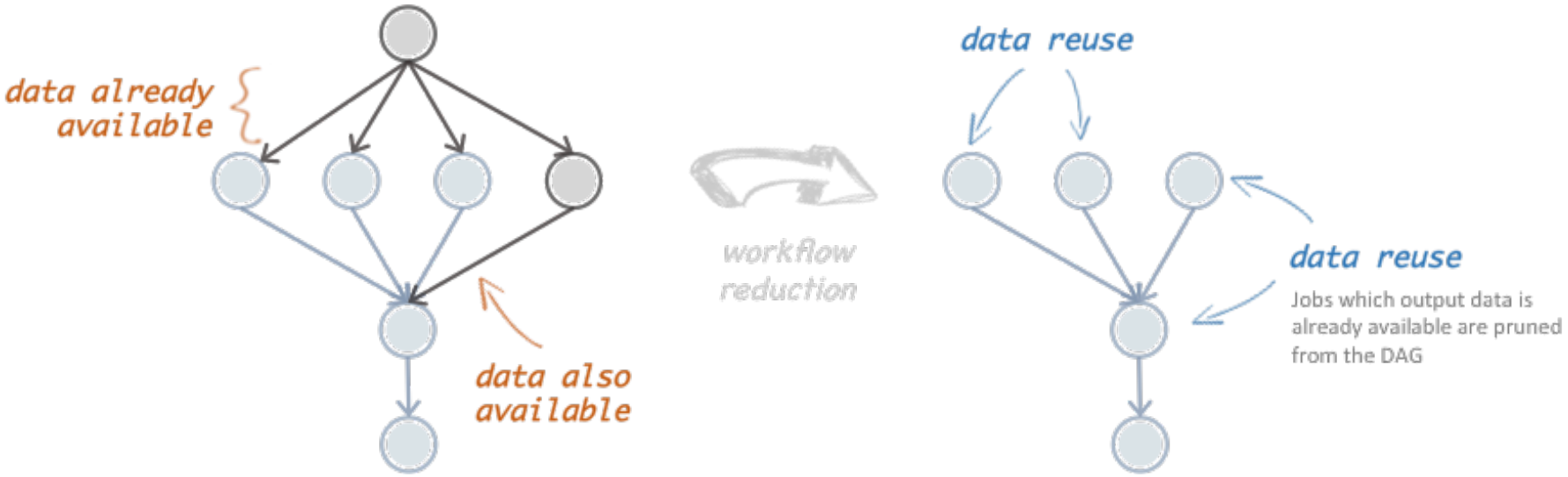
  Users want to be able to intelligently reuse previously generated outputs.

  Genomic Workflows want to avoid index generation steps for raw input files

Long running analysis fail, and users realize they need to regenerate workflow description, as wrong arguments set or missing jobs.



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

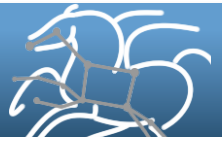
File cleanup

- **Problem**

  Users run out out of disk space during workflow execution

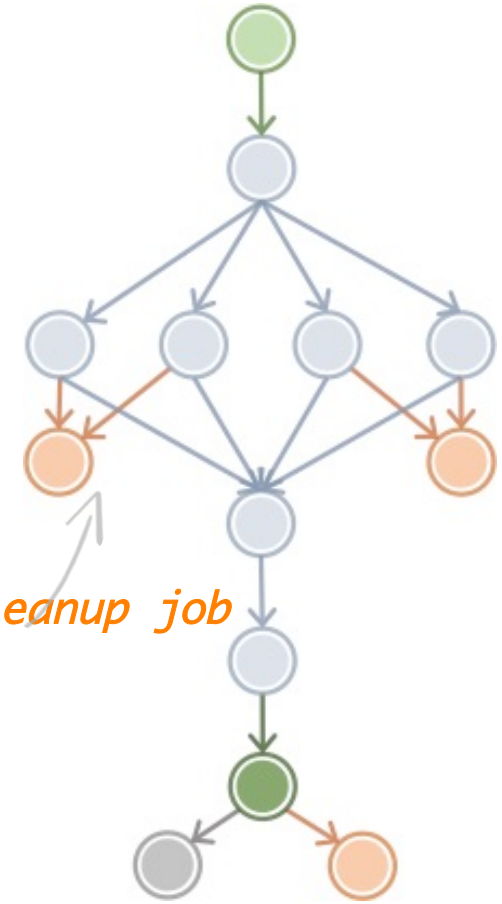
- **Why does it occur**
  - Workflows could bring in huge amounts of data
  - Data is generated during workflow execution
  - Users don't worry about cleaning up after they are done

- **Example Application**
  - SoyKB workflows from iPlant
  - Epigenomics Pipelines generating 60TB of data in a single execution

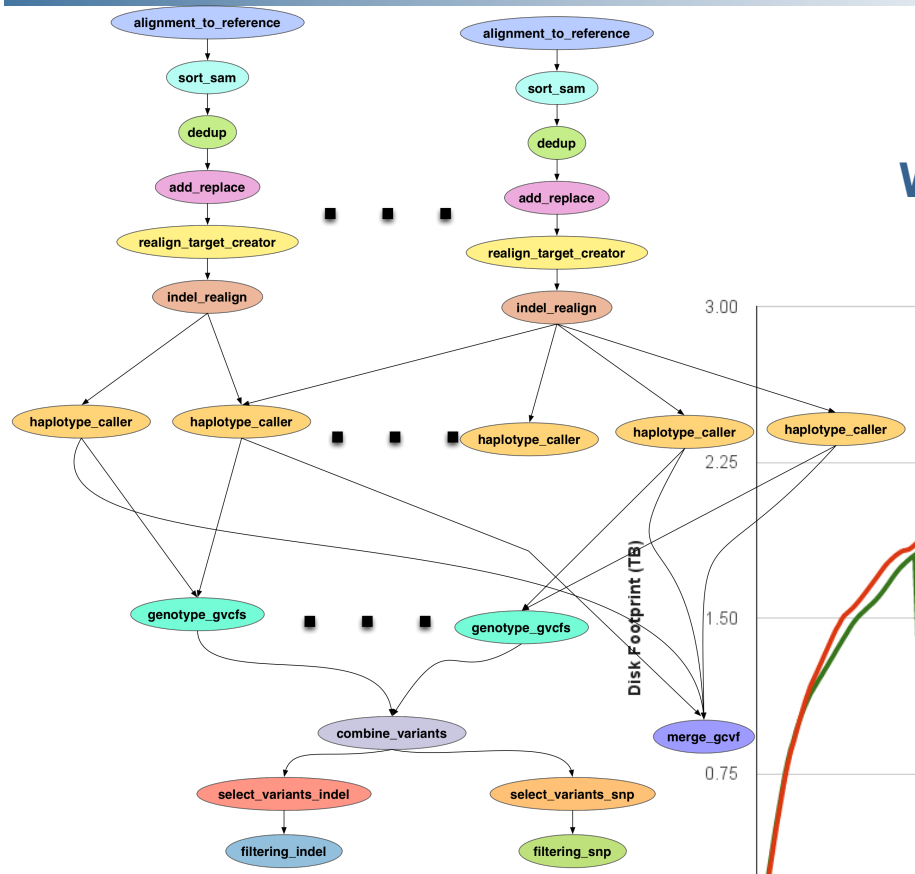


File cleanup

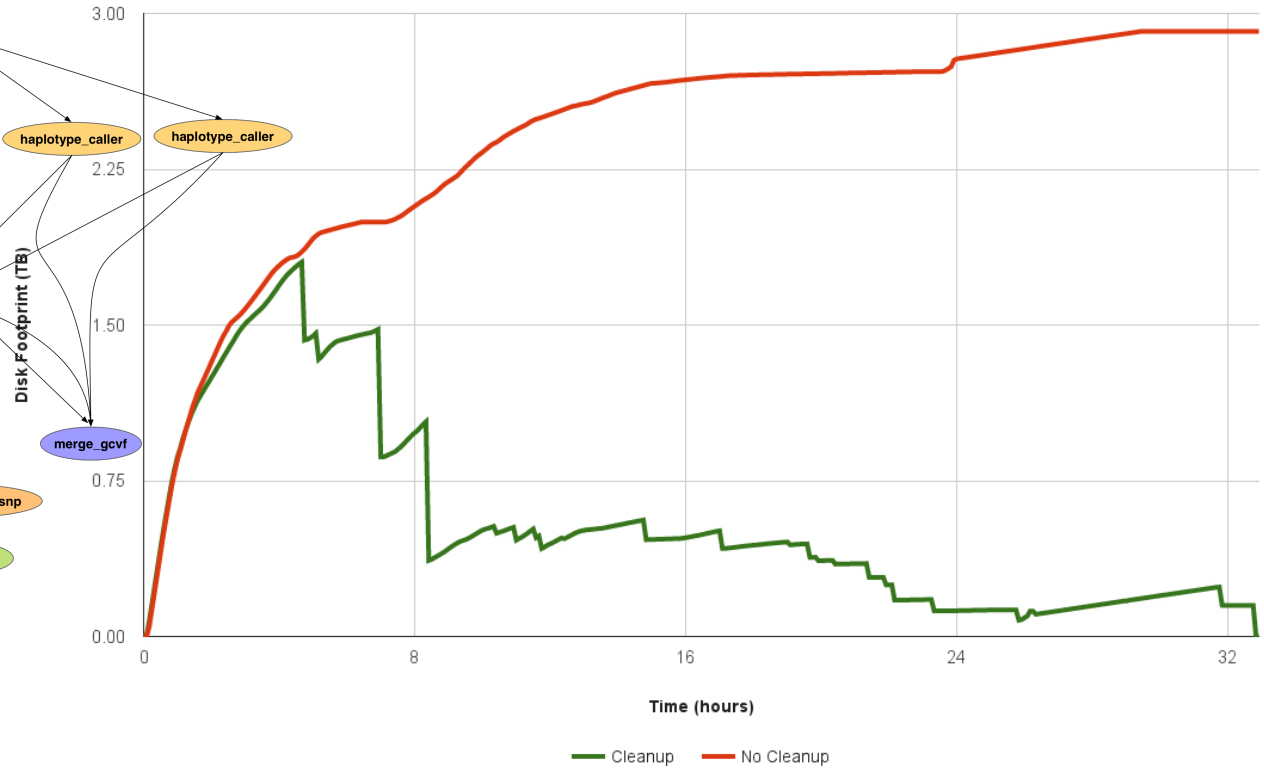
- **Solution**
  - **Do cleanup after workflows finish**
    - Does not work as the scratch may get filled much before during execution
  - **Interleave cleanup automatically during workflow execution.**
    - Requires an analysis of the workflow to determine, when a file is no longer required
  - **Cluster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



File cleanup (cont)

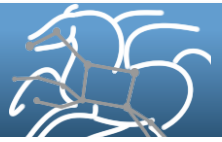


Single SoyKB NGS Pegasus Workflow with 10 input reads.



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

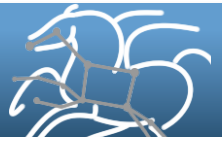
- **Provenance**
  - Provenance data is collected in a database, and the data can be summaries with tools such as pegasus-statistics, pegasus-plots, or directly with SQL queries.

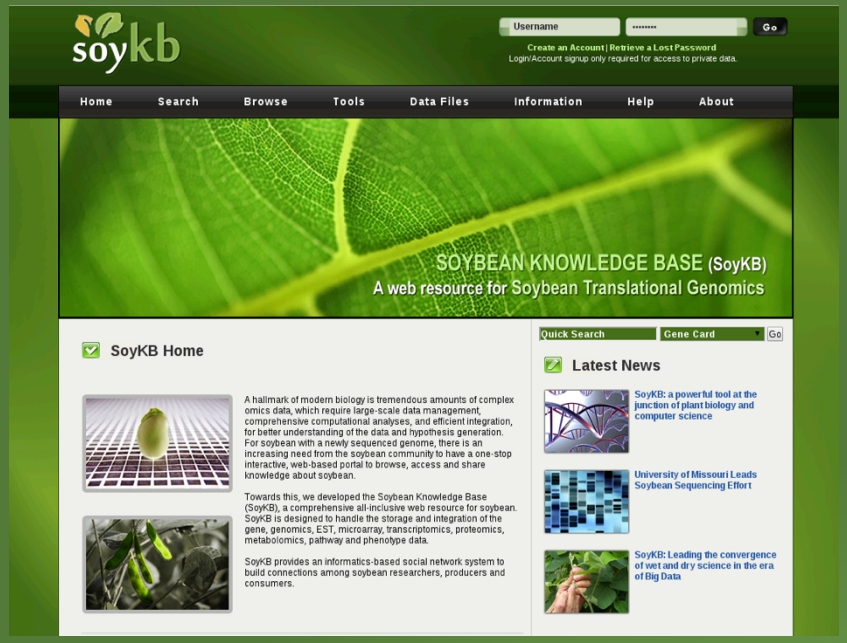
- **Reliability and Debugging Tools**
  - Jobs and data transfers are automatically retried in case of failures. Debugging tools such as pegasus-analyzer helps the user to debug the workflow in case of non-recoverable failures.

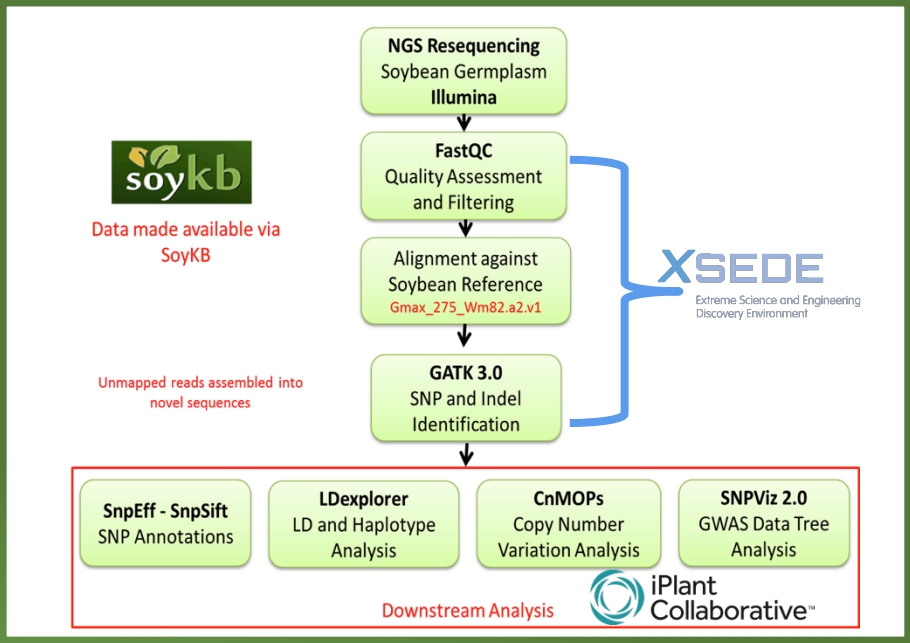
- **Scalability**
  - Hierarchal workflows
  - Scale to hundreds of thousands of nodes in a workflow.

Pegasus in Bioinformatics

- **PAGE Study** https://www.pagestudy.org
  - Quality Control
  - Imputation Workflows
- **CGSMD** https://www.nimhgenetics.org/submit_data/
  - Data Submission to NRGR
- **iPlant**
  - SoyKB http://soykb.org
  - International Rice Research Institute (IRRI)
- **Splinter(Structural Protein Ligand Interactome)**
- **Children's Hospital of Philadelphia Pediatric Genome Analysis**
- **Bioinformatic discovery of bacterial regulatory RNAs using SIPHT** http://newbio.cs.wisc.edu/sRNA/
- **RNA Sequencing workflows**
  - https://genomics.isi.edu/gtfar





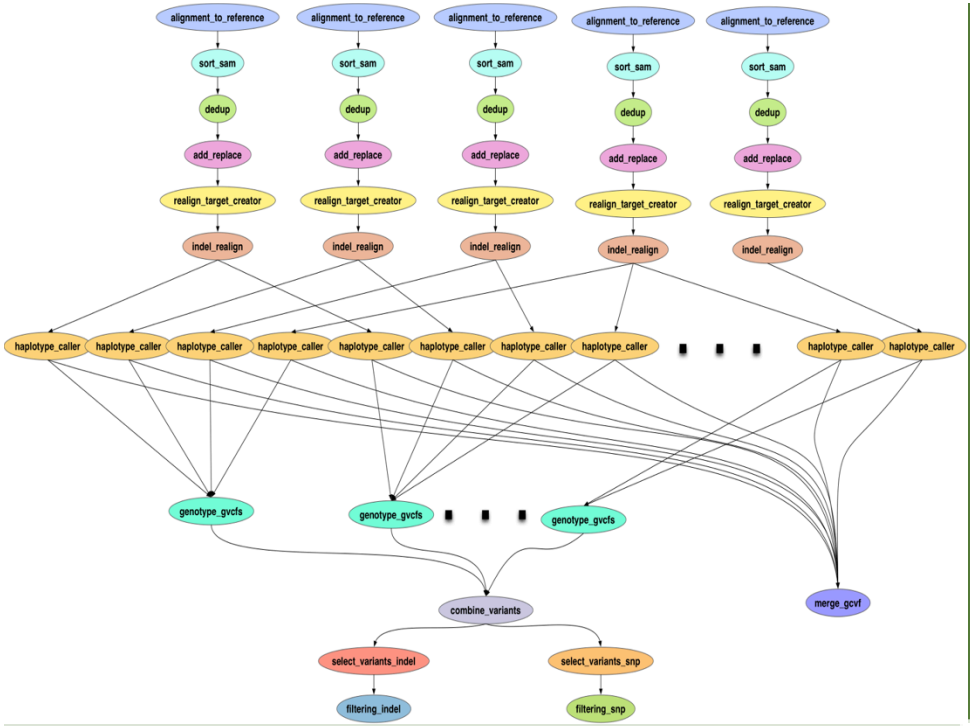


http://soykb.org

XSEDE Allocation

PI: Dong Xu

Trupti Joshi, Saad Kahn, Yang Liu, Juexin Wang, Badu Valliyodan, Jiaojiao Wang



Task	Base Code	Cores (Threads)	Memory (GB)
Alignment_to_reference	BWA	7	8
Sort_sam	Picard	1	21
Dedup	Picard	1	21
Add_replace	Picard	1	21
Realign_target_creator	GATK	15	10
Indel_realign	GATK	1	10
Haplotype_caller	GATK	1	3
Genotype_gvcfs	GATK	1	10
Merge_gvcf	GATK	10	20
Combine_variants	GATK	1	10
Select_variants	GATK	14	10
Filtering	GATK	1	10

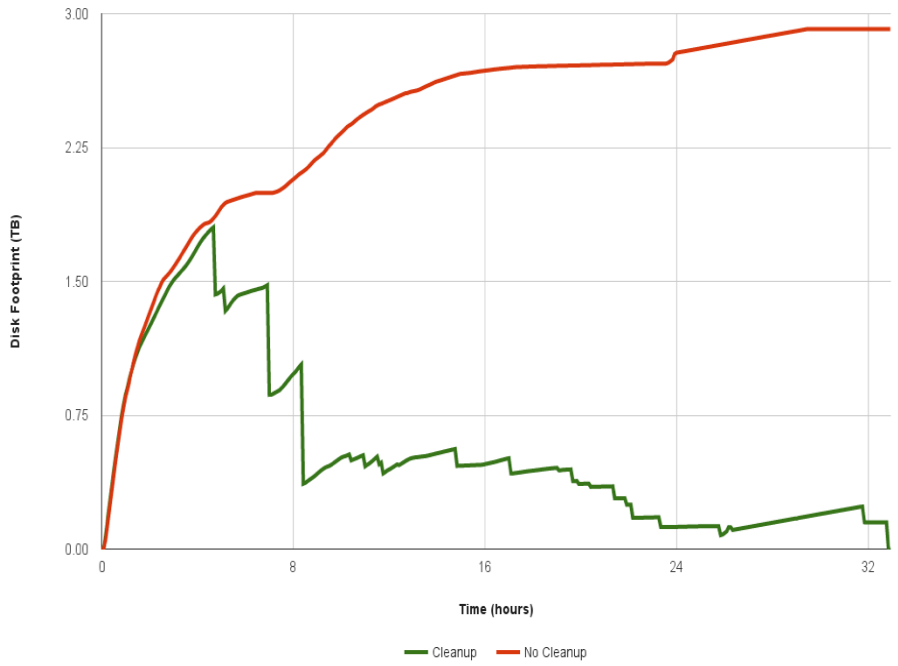
TACC Wrangler as Execution Environment

Flash Based Shared Storage

Switched to glideins (pilot jobs) - Brings in remote compute nodes and joins them to the HTCondor pool on in the submit host - Workflow runs at a finer granularity

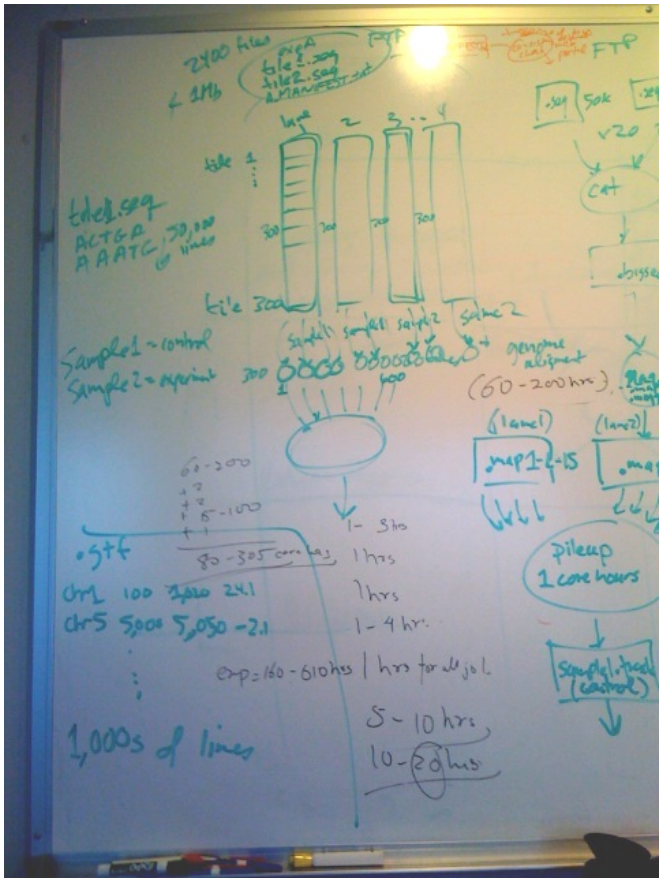
Works well on Wrangler due to more cores and memory per node (48 cores, 128 GB RAM)

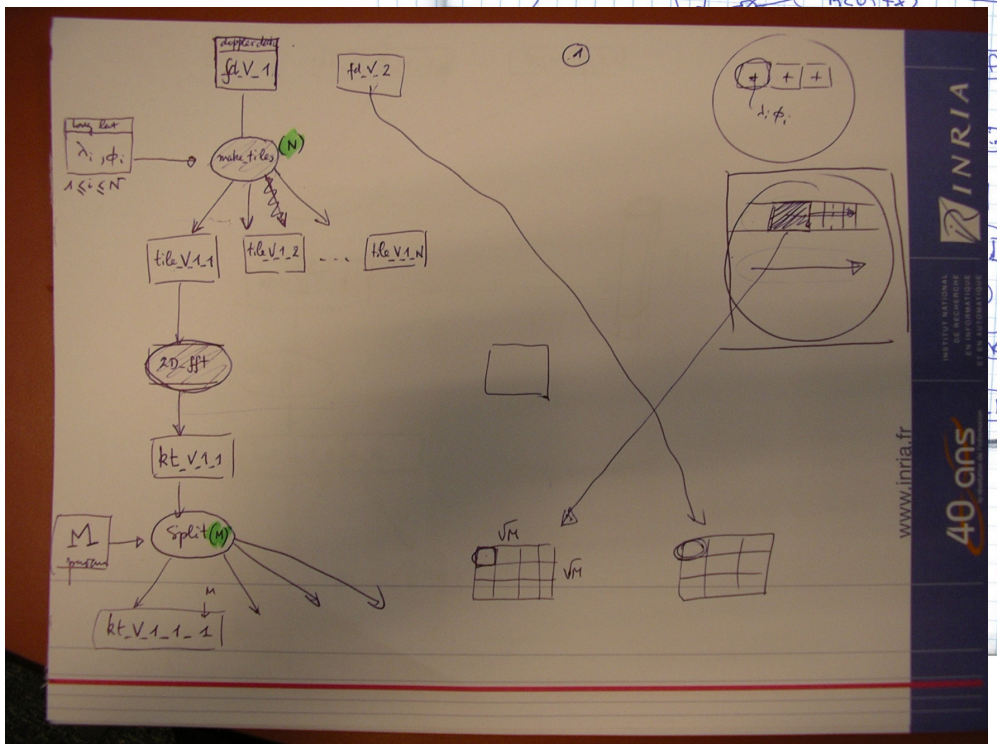
Pegasus



If you get stuck…

And you can draw….





We can help you!

36

Pegasus est. 2001

Automate, recover, and debug scientific computations.

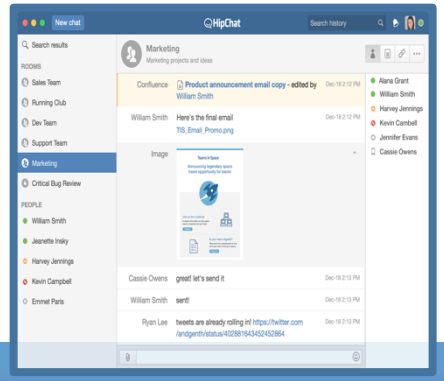
Get Started

Pegasus Website
http://pegasus.isi.edu

Users Mailing List
pegasus-users@isi.edu

Support
pegasus-support@isi.edu

HipChat



USC Viterbi School of Engineering

Pegasus est. 2001

Automate, recover, and debug scientific computations.

Thank You

Questions?

Meet our team


Ewa Deelman


Karan Vahi


Gideon Juve


Mats Rynge


Rajiv Mayani


Rafael Ferreira da Silva