Fast, Reliable, Loosely Coupled Parallel Computation

Tiberiu Stef-Praun

Computation Institute

University of Chicago & Argonne National Laboratory

tiberius@ci.uchicago.edu

www.ci.uchicago.edu/swift

Goals of using grids through scripting

- Provide an easy on-ramp to the grid
- Utilize massive resources with simple scripts
  - Leverage multiple grids like a workstation
- Empower script-writers to empower end users
- Track and leverage provenance in the science process

VDS – The Virtual Data System

- Introduced Virtual Data Language - VDL
  - A location-independent parallel language
- Several Planners
  - Pegasus: main production planner
  - Euryale: experimental “just in time” planner
  - GADU/GNARE – user application planner (D. Sulahke, Argonne)
- Provenance
  - Kickstart – app launcher and tracker
  - VDC – virtual data catalog

VDL/VDS Limitations

- Missing VDL language features
  - Data typing & data mapping
  - Iterators and control-flow constructs
- Run time complexity in VDS
  - State explosion for data-parallel applications
  - Computation status hard to provide
  - Debugging information complex & distributed
- Performance
  - Still many runtime bottlenecks

the globus alliance
www.globus.org

Swift System

- Clean separation of logical/physical concerns
  - **XDTM** specification of logical data structures
+ Concise specification of parallel programs
  - **SwiftScript**, with iteration, etc.
+ Efficient execution on distributed resources
  - Lightweight threading, dynamic provisioning, Grid interfaces, pipelining, load balancing
+ Rigorous provenance tracking and query
  - Virtual data schema & automated recording

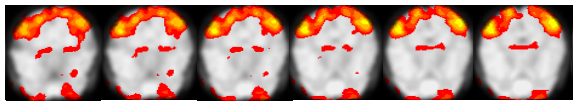
→ **Improved usability and productivity**

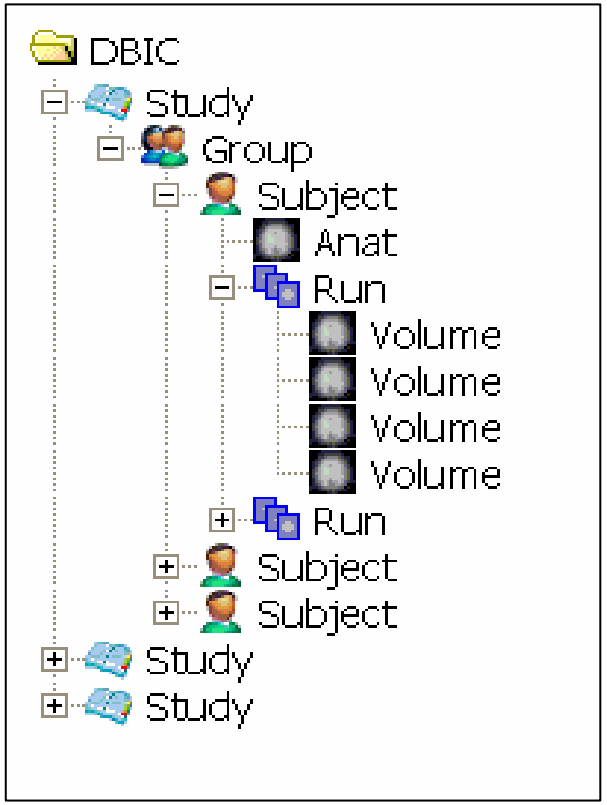
- Demonstrated in numerous applications

the globus alliance
www.globus.org

The Messy Data Problem

- Scientific data is typically logically structured
  - E.g., hierarchical structure
  - Common to map functions over dataset members
  - Nested map operations can scale to millions of objects





The Messy Data Problem

- But physically "messy"
- Heterogeneous storage format and access protocol
  - Logically identical dataset can be stored in textual File (e.g. CSV), spreadsheet, database, ...
  - Data available from filesystem, DBMS, HTTP, WebDAV, ..
- Metadata encoded in directory and file names
- Hinders program development, composition, execution

```
./Group23
total 58
drwxr-xr-x  4 yongzh users 2048 Nov 12 14:15 AA
drwxr-xr-x  4 yongzh users 2048 Nov 11 21:13 CH
drwxr-xr-x  4 yongzh users 2048 Nov 11 16:32 EC

./Group23/AA:
total 4
drwxr-xr-x  5 yongzh users 2048 Nov  5 12:41 04nov06aa
drwxr-xr-x  4 yongzh users 2048 Dec  6 12:24 11nov06aa

. /Group23/AA/04nov06aa:
total 54
drwxr-xr-x  2 yongzh users  2048 Nov  5 12:52 ANATOMY
drwxr-xr-x  2 yongzh users 49152 Dec  5 11:40 FUNCTIONAL

. /Group23/AA/04nov06aa/ANATOMY:
total 58500
-rw-r--r--  1 yongzh users      348 Nov  5 12:29 coplanar.hdr
-rw-r--r--  1 yongzh users 16777216 Nov  5 12:29 coplanar.img

. /Group23/AA/04nov06aa/FUNCTIONAL:
total 196739
-rw-r--r--  1 yongzh users     348 Nov  5 12:32 bold1_0001.hdr
-rw-r--r--  1 yongzh users  409600 Nov  5 12:32 bold1_0001.img
-rw-r--r--  1 yongzh users     348 Nov  5 12:32 bold1_0002.hdr
-rw-r--r--  1 yongzh users  409600 Nov  5 12:32 bold1_0002.img
-rw-r--r--  1 yongzh users     496 Nov 15 20:44 bold1_0002.mat
-rw-r--r--  1 yongzh users     348 Nov  5 12:32 bold1_0003.hdr
-rw-r--r--  1 yongzh users  409600 Nov  5 12:32 bold1_0003.img
```

the globus alliance
www.globus.org

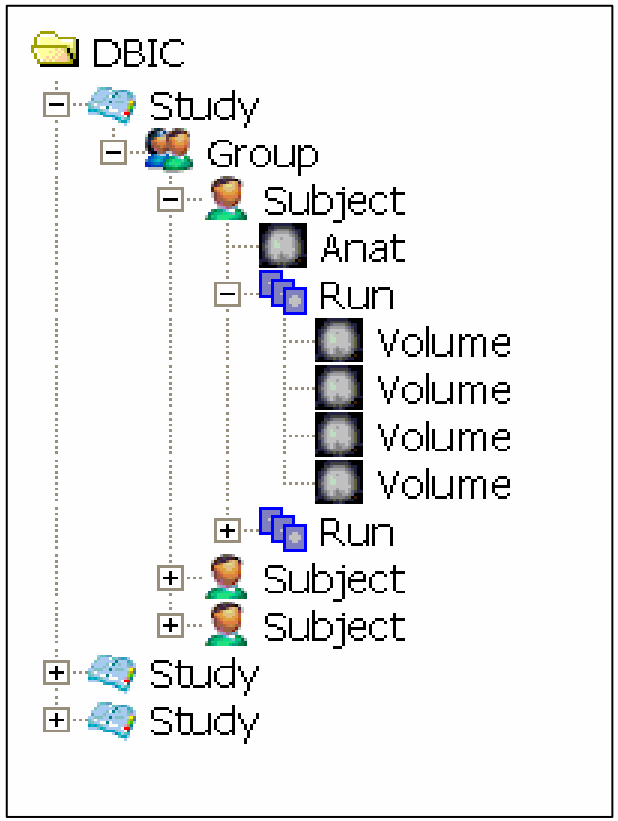
SwiftScript

- Typed parallel programm [SIGMOD05, Springer06]
  - XDTM as data model and type system
  - Typed dataset and procedure definitions
- Scripting language
  - Implicit data parallelism
  - Program composition from procedures
  - Control constructs (foreach, if, while, ...)

Clean application logic
Type checking
Dataset selection, iteration

A Notation & System for Expressing and Executing Cleanly Typed Workflows on Messy Scientific Data [SIGMOD Record Sep05]

fMRI Type Definitions in SwiftScript



Simplified declarations of fMRI AIRSN (Spatial Normalization)

```
type Study {
        Group g[ ];
}

type Group {
        Subject s[ ];
}

type Subject {
        Volume anat;
        Run run[ ];
}

type Run {
        Volume v[ ];
}

type Volume {
        Image img;
        Header hdr;
}
```

```
type Image {};

type Header {};

type Warp {};

type Air {};

type AirVec {
        Air a[ ];
}

type NormAnat {
        Volume anat;
        Warp aWarp;
        Volume nHires;
}
```

AIRSN Program Definition

```
(Run snr) functional ( Run r, NormAnat a,
                       Air shrink ) {
    Run yroRun = reorientRun( r , "y" );
    Run roRun = reorientRun( yroRun , "x" );
    Volume std = roRun[0];
    Run rndr = random_select( roRun, 0.1 );
    AirVector rndAirVec = align_linearRun( rndr, std, 12, 1000, 1000, "81 3 3" );
    Run reslicedRndr = resliceRun( rndr, rndAirVec, "o", "k" );
    Volume meanRand = softmean( reslicedRndr, "y", "null" );
    Air mnQAAir = alignlinear( a.nHires, meanRand, 6, 1000, 4, "81 3 3" );
    Warp boldNormWarp = combinewarp( shrink, a.aWarp, mnQAAir );
    Run nr = reslice_warp_run( boldNormWarp, roRun );
    Volume meanAll = strictmean( nr, "y", "null" )
    Volume boldMask = binarize( meanAll, "y" );
    snr = gsmoothRun( nr, boldMask, "6 6 6" );
}
```

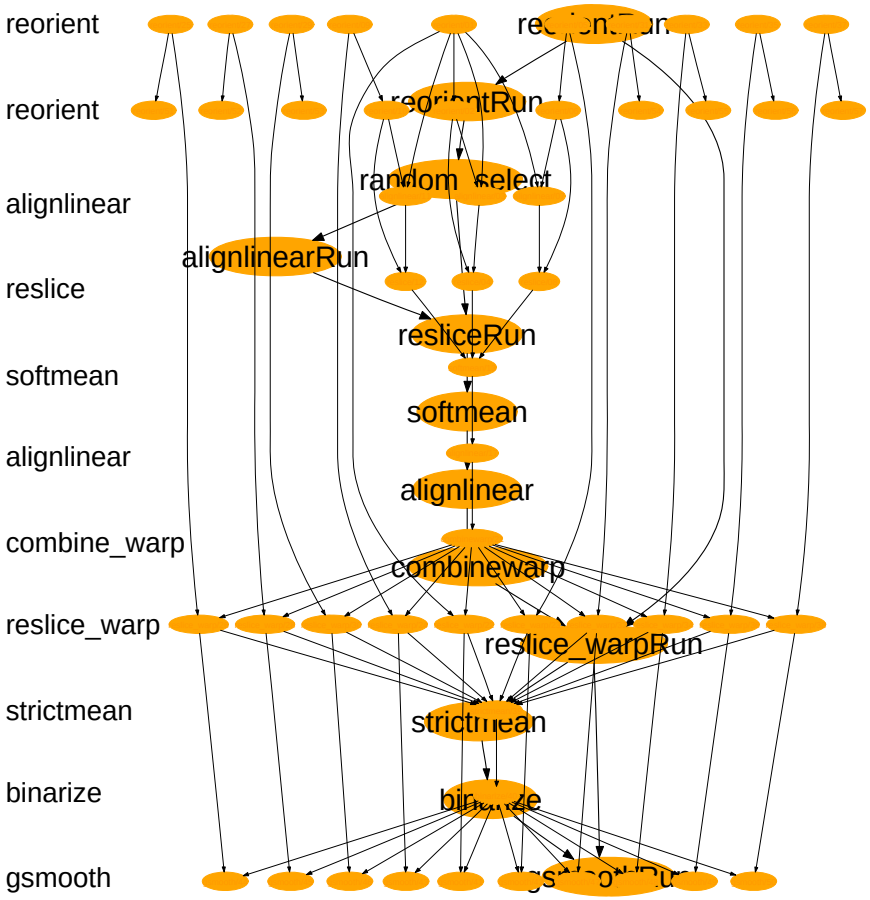
```
(Run or) reorientRun (Run ir,
                      string direction) {
    foreach Volume iv, i in ir.v {
        or.v[i] = reorient(iv, direction);
    }
}
```

SwiftScript Expressiveness

Lines of code with different workflow encodings

fMRI Workflow	Shell Script	VDL	Swift
ATLAS1	49	72	6
ATLAS2	97	135	10
FILM1	63	134	17
FEAT	84	191	13
AIRSN	215	~400	34

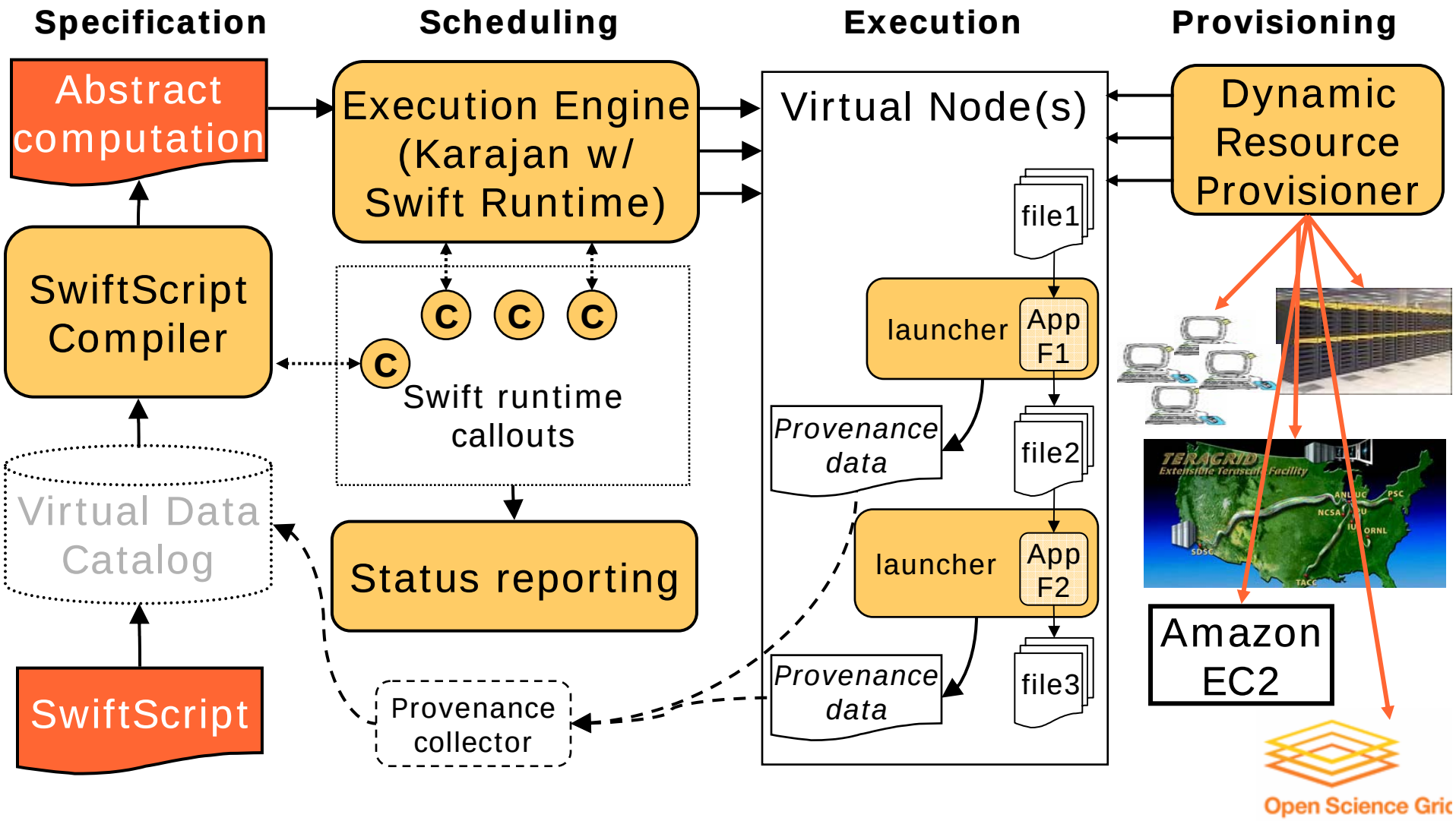
AIRSN workflow:expanded:



Collaboration with James Dobson, Dartmouth [SIGMOD Record Sep05]

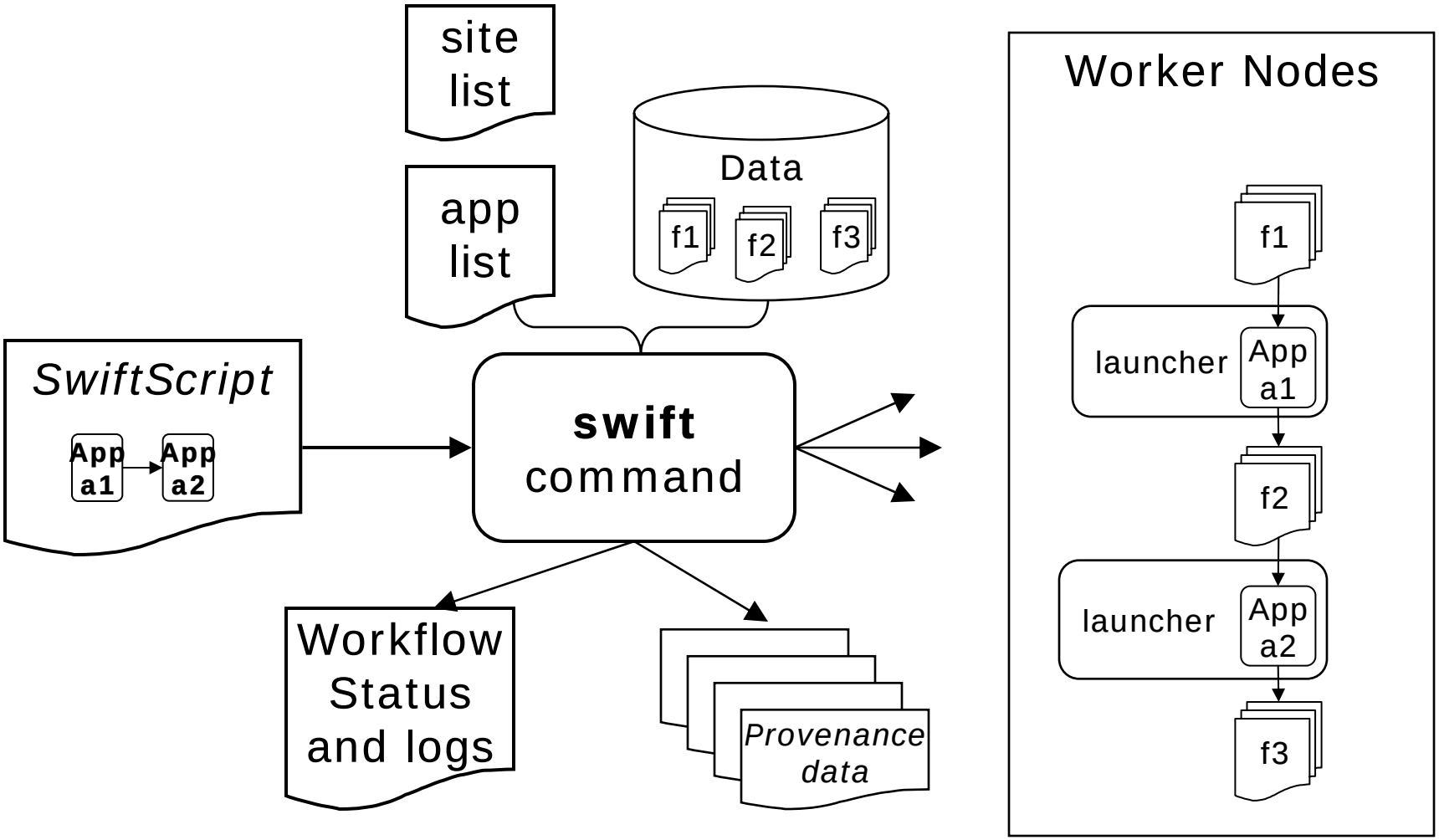
the globus alliance
www.globus.org

Swift Architecture



the globus alliance
www.globus.org

Using Swift



the globus alliance
www.globus.org

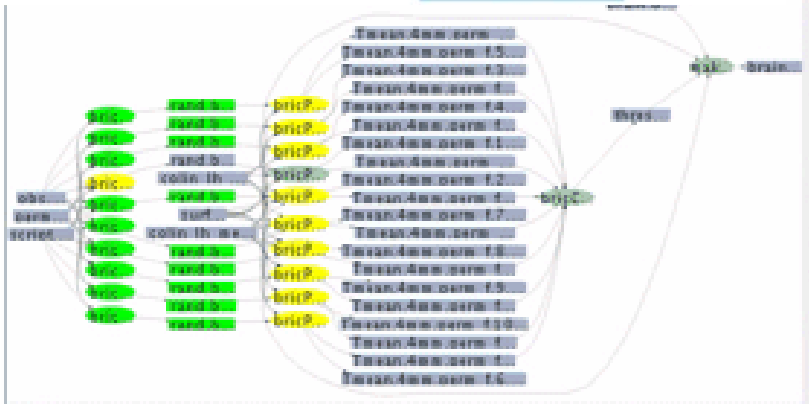
Swift uses Karajan Workflow Engine

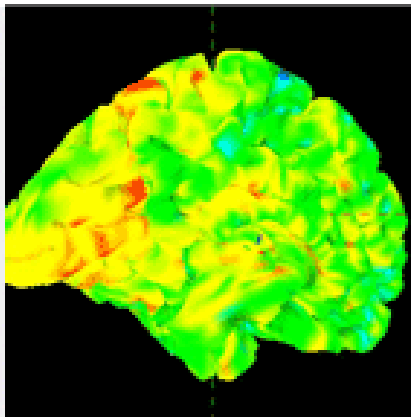
- Fast, scalable threading model
- Suitable constructs for control flow
- Flexible task dependency model
  - “Futures” enable pipelining
- Flexible provider model allows for use of different run time environments
  - Job execution and data transfer
  - Flow controlled to avoid resource overload
- Workflow client runs from a Java container

the globus alliance
www.globus.org

Application example: ACTIVAL: Neural activation validation

Identifies clusters of neural activity not likely to be active by random chance: switch labels of the conditions for one or more participants; calculate the delta values in each voxel, re-calculate the reliability of delta in each voxel, and evaluate clusters found. If the clusters in data are greater than the majority of the clusters found in the permutations, then the null hypothesis is refuted indicating that clusters of activity found in our experiment are not likely to be found by chance.





Work by S. Small and U. Hasson, UChicago.

SwiftScript Workflow ACTIVAL – Data types and utilities

```
type script {}
type brainMeasurements{}
type precomputedPermutations{}
type brainClusterTable {}
type brainDatasets{ brainDataset b[]; }
type brainClusters{ brainClusterTable c[]; }
type fullBrainData {}
type fullBrainSpecs {}
type brainDataset {}

// Procedure to run "R" statistical package
(brainDataset t) bricRInvoke (script permutationScript, int iterationNo,
   brainMeasurements dataAll, precomputedPermutations dataPerm) {
      app { bricRInvoke @filename(permutationScript) iterationNo
                        @filename(dataAll) @filename(dataPerm); }
}

// Procedure to run AFNI Clustering tool
(brainClusterTable v, brainDataset t) bricCluster (script clusterScript,
  int iterationNo, brainDataset randBrain, fullBrainData brainFile,
  fullBrainSpecs specFile) {
      app { bricPerlCluster @filename(clusterScript) iterationNo
                        @filename(randBrain) @filename(brainFile)
                        @filename(specFile); }
}

// Procedure to merge results based on statistical likelhoods
(brainClusterTable t) bricCentralize ( brainClusterTable bc[]) {
     app { bricCentralize @filenames(bc); }
}
```

ACTIVAL Workflow – Dataset iteration procedures

```
// Procedure to iterate over the data collection

(brainClusters randCluster, brainDatasets dsetReturn) brain_cluster
   (fullBrainData brainFile, fullBrainSpecs specFile)
{
   int sequence[]=[1:2000];

   brainMeasurements          dataAll<fixed_mapper; file="obs.imit.all">;
   precomputedPermutations dataPerm<fixed_mapper; file="perm.matrix.11">;
   script                          randScript<fixed_mapper; file="script.obs.imit.tibi">;
   script                          clusterScript<fixed_mapper; file="surfclust.tibi">;
   brainDatasets                randBrains<simple_mapper; prefix="rand.brain.set">;

   foreach int i in sequence {
      randBrains.b[i] = bricRInvoke(randScript,i,dataAll,dataPerm);
      brainDataset rBrain = randBrains.b[i] ;
      (randCluster.c[i],dsetReturn.b[i]) =
         bricCluster(clusterScript,i,rBrain, brainFile,specFile);
   }
}
```

ACTIVAL Workflow – Main Workflow Program

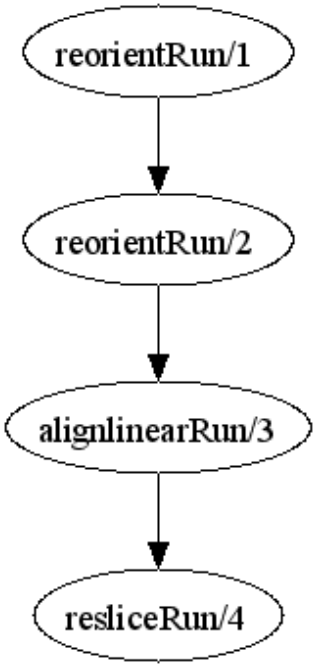
```
// Declare datasets

fullBrainData       brainFile<fixed_mapper; file="colin_lh_mesh140_std.pial.asc">;
fullBrainSpecs      specFile<fixed_mapper; file="colin_lh_mesh140_std.spec">;

brainDatasets       randBrain<simple_mapper; prefix="rand.brain.set">;
brainClusters       randCluster<simple_mapper; prefix="Tmean.4mm.perm",
                        suffix="_ClstTable_r4.1_a2.0.1D">;
brainDatasets       dsetReturn<simple_mapper; prefix="Tmean.4mm.perm",
                        suffix="_Clustered_r4.1_a2.0.niml.dset">;
brainClusterTable   clusterThresholdsTable<fixed_mapper; file="thresholds.table">;
brainDataset        brainResult<fixed_mapper; file="brain.final.dset">;
brainDataset        origBrain<fixed_mapper; file="brain.permutation.1">;


// Main program – executes the entire workflow

(randCluster, dsetReturn) = brain_cluster(brainFile, specFile);

clusterThresholdsTable = bricCentralize (randCluster.c);

brainResult = makebrain(origBrain,clusterThresholdsTable,brainFile,specFile);
```

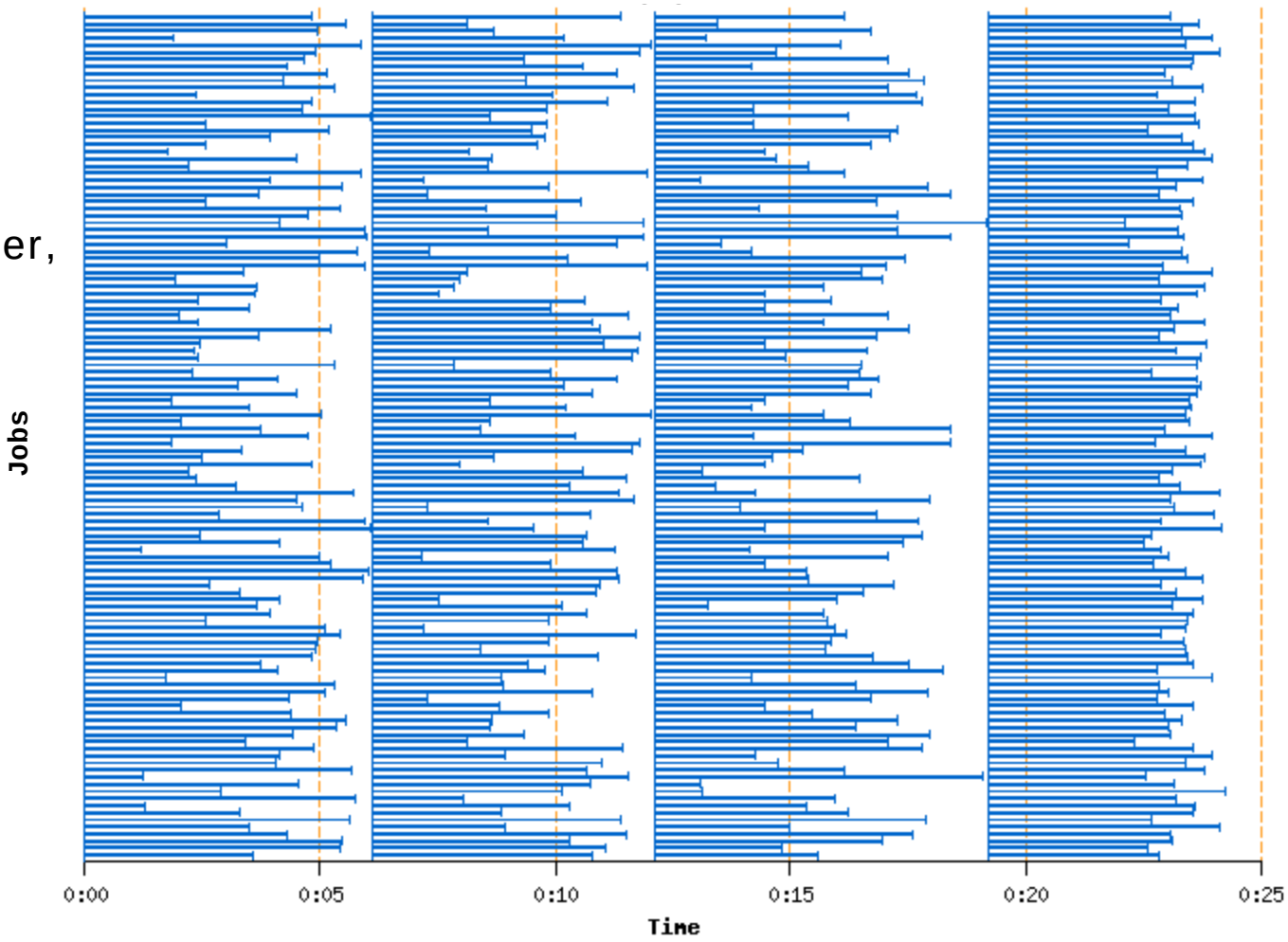
the globus alliance
www.globus.org

Performance example: fMRI workflow

4-stage workflow (subset of AIRSN)
476 jobs, <10 secs CPU each, 119 jobs per stage.

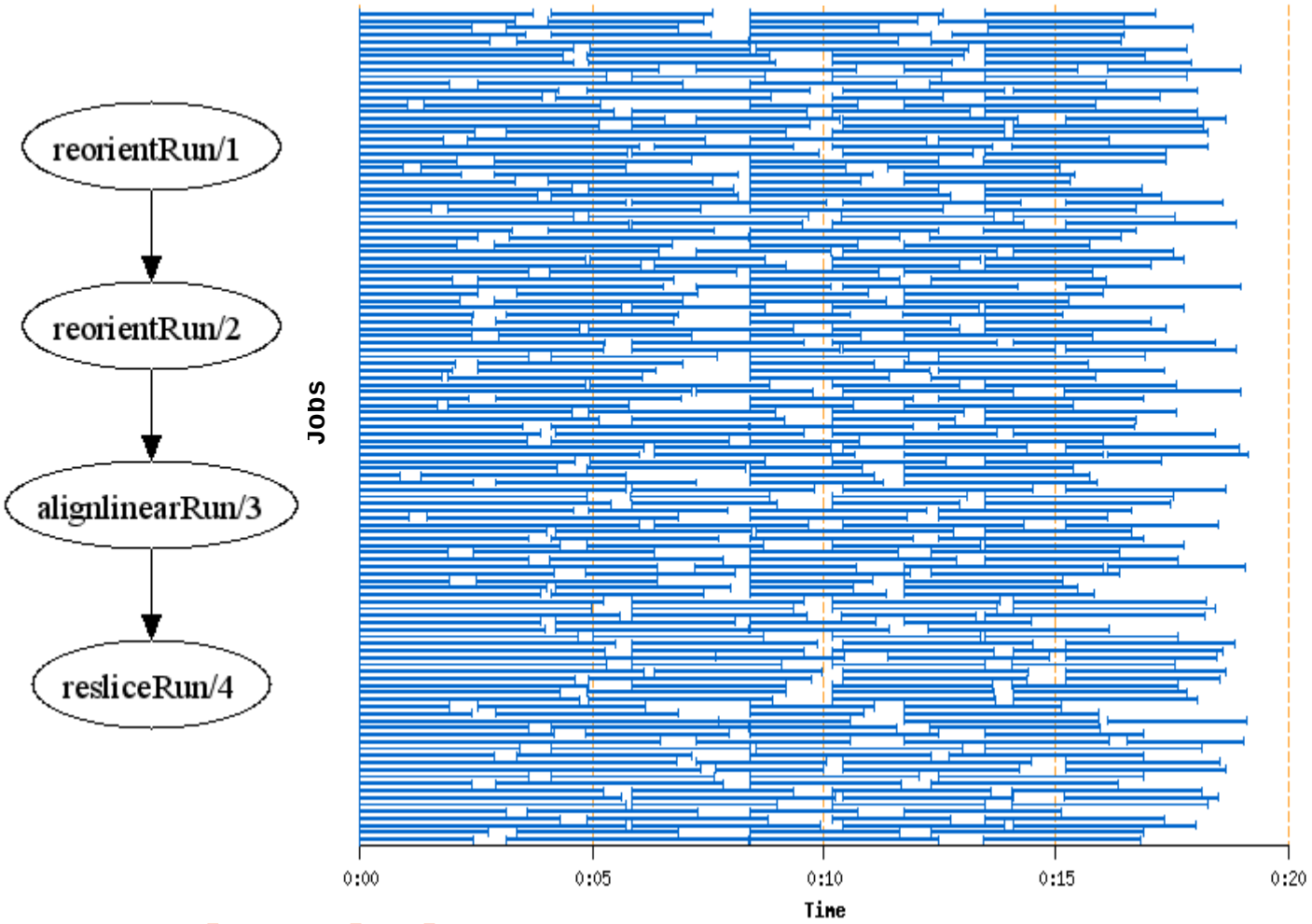
No pipelining:
24 minutes
(idle uc-teragrid cluster, via GRAM to Torque)





the globus alliance
www.globus.org

Example Performance Optimizations



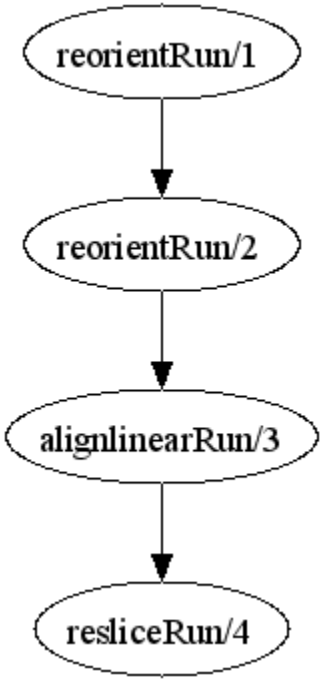
Jobs pipelined between stages:

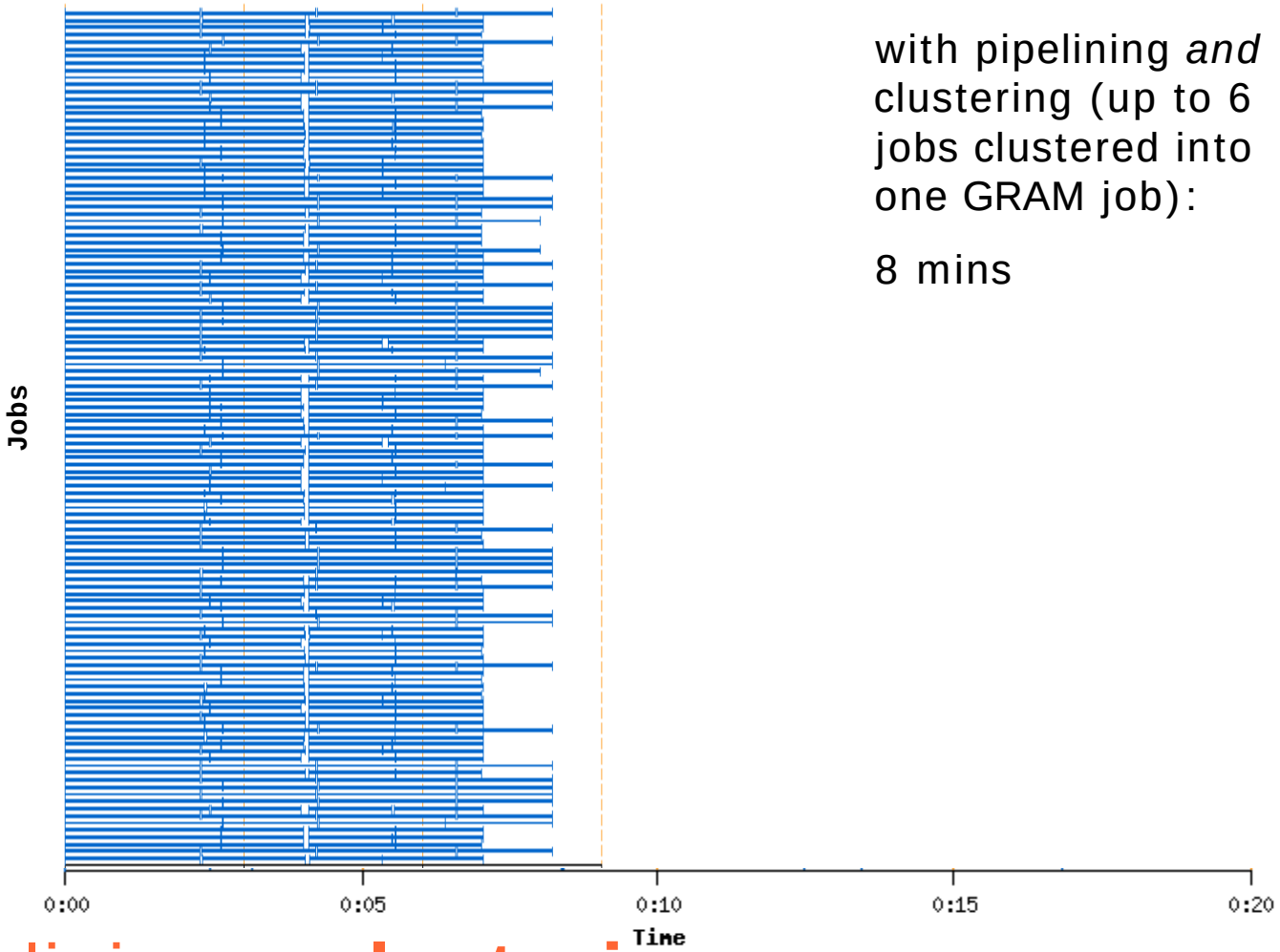
19 minutes

Pipelining

the globus alliance
www.globus.org

Example Performance Optimizations





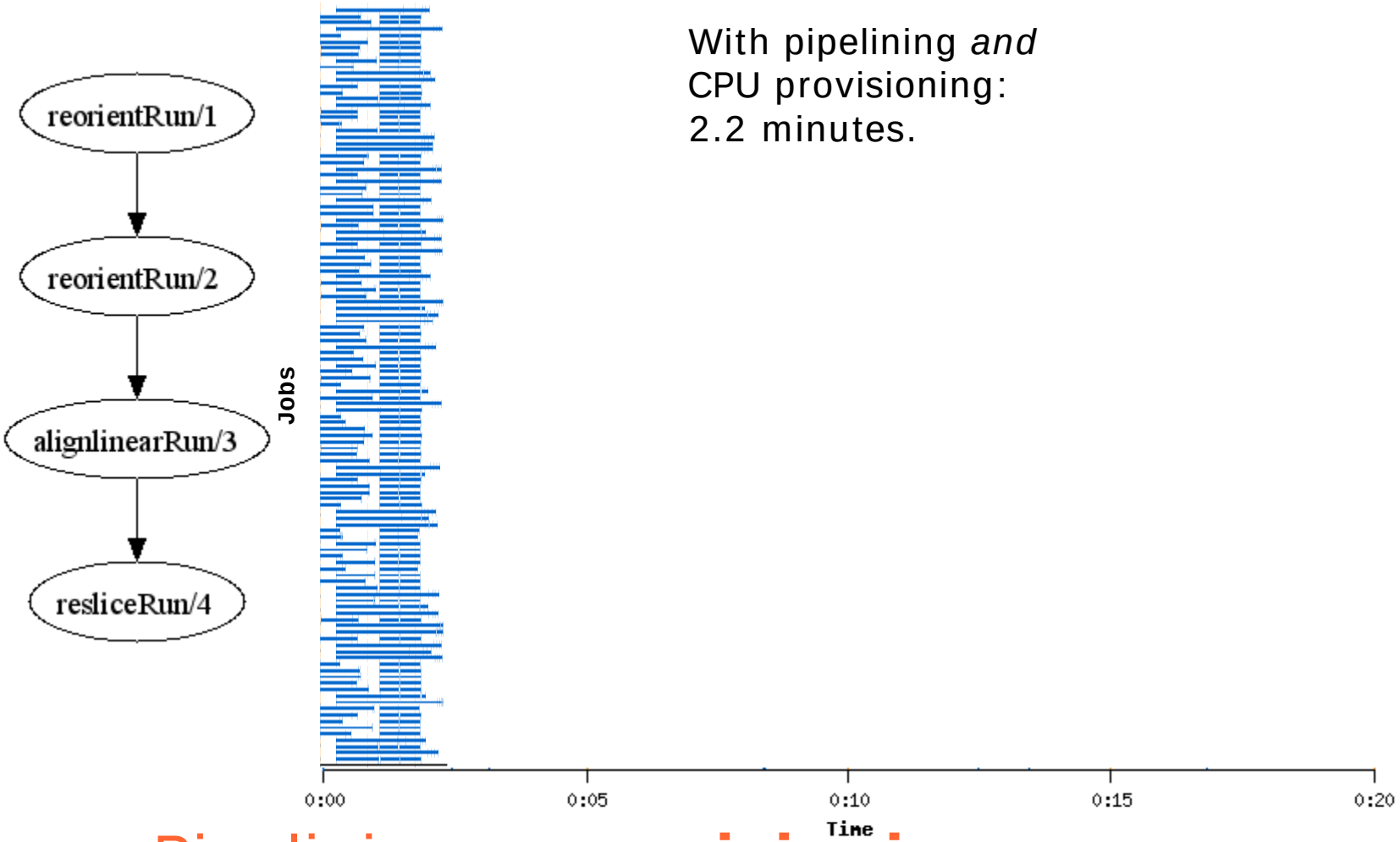
with pipelining *and* clustering (up to 6 jobs clustered into one GRAM job):

8 mins

Pipelining + **clustering**

the globus alliance
www.globus.org

Example Performance Optimizations

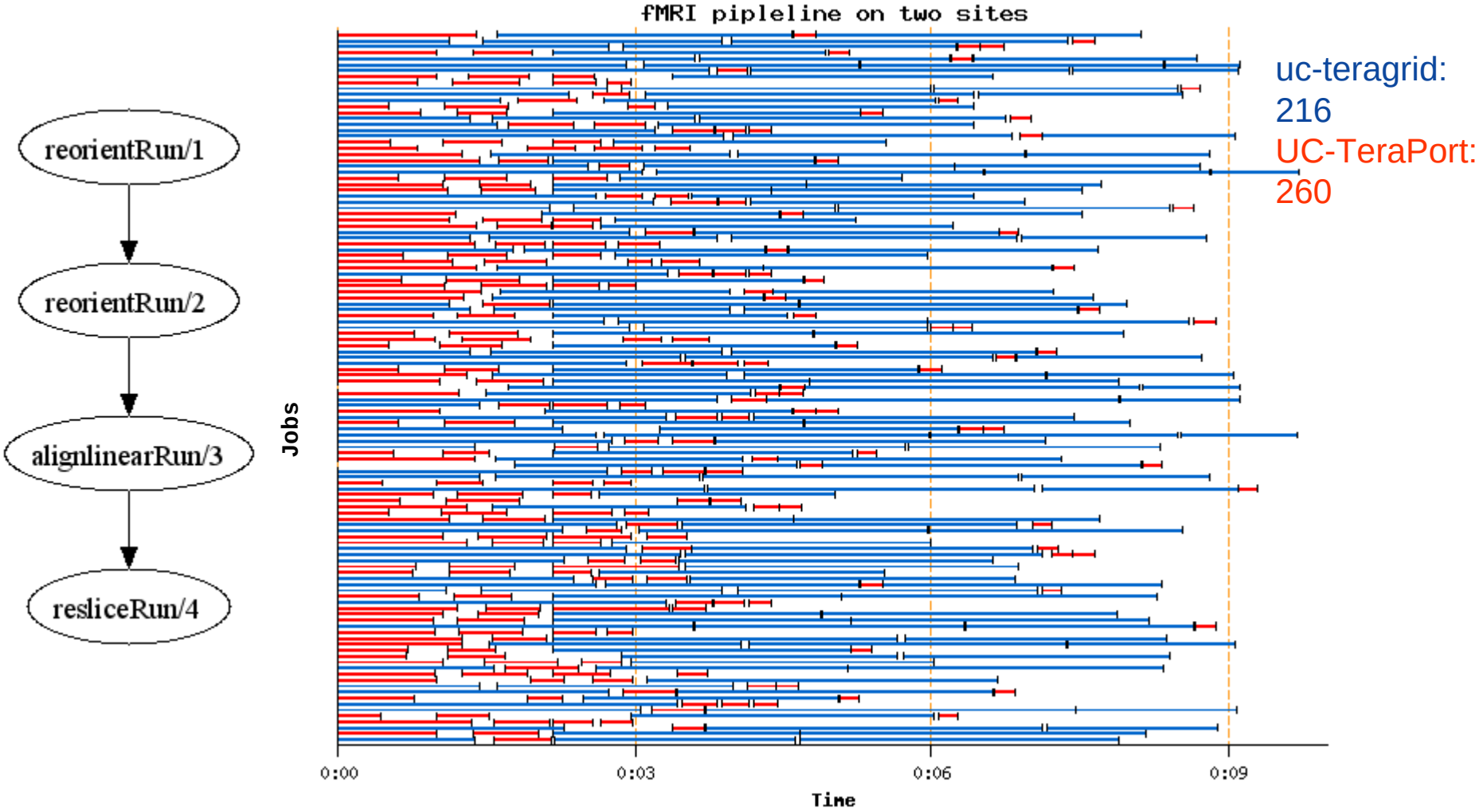


With pipelining *and* CPU provisioning: 2.2 minutes.

Pipelining + **provisioning**

the globus alliance
www.globus.org

Load Balancing



Load balancing between UC TeraPort (OSG) and UC-TeraGrid (IA32)

Development Status

- Initial release is available for evaluation
- Performance measurement and tuning efforts active
- Adapting to OSG Grid info and site conventions
- Many applications in progress and eval
  - Astrophysics, molecular dynamics, neuroscience, psychology, radiology
- Provisioning mechanism progressing
- Virtual data catalog re-integration starting ~ April
- Collating language feedback – focus is on mapping
- Web site for docs, downloads and more info:
  www.ci.uchicago.edu/swift

Conclusion

- Swift is in its early stages of development and its transition from the VDS virtual data language
- Application testing is underway in neuroscience, molecular dynamics, astrophysics, radiology, and other applications.
  - Providing valuable feedback for language refinement and finalization
- SwiftScript is proving to be a productive language
  - while feedback from usage is still shaping it
  - Positive comments from VDL users – radiology in particular
- Ongoing performance evaluation and improvement is yielding exciting results
- Major initial focus is usability – good progress on improving time-to-get-started and on ease of debugging

the globus alliance
www.globus.org

Acknowledgements

- Swift effort is supported by DOE (Argonne LDRD), NSF (i2u2,GriPhyN, iVDGL), NIH, and the UChicago Computation Institute
- Team
  - Ben Clifford, Ian Foster, Mihael Hategan, Veronika Nefedova, Tiberiu Stef-Praun, Mike Wilde, Yong Zhao
- Java CoG Kit
  - Mihael Hategan, Gregor Von Laszewski, and many collaborators
- User contributed workflows and Swift applications
  - ASCI Flash, I2U2, UC Human Neuroscience Lab, UCH Molecular Dynamics, UCH Radiology

Thank you!

Questions?