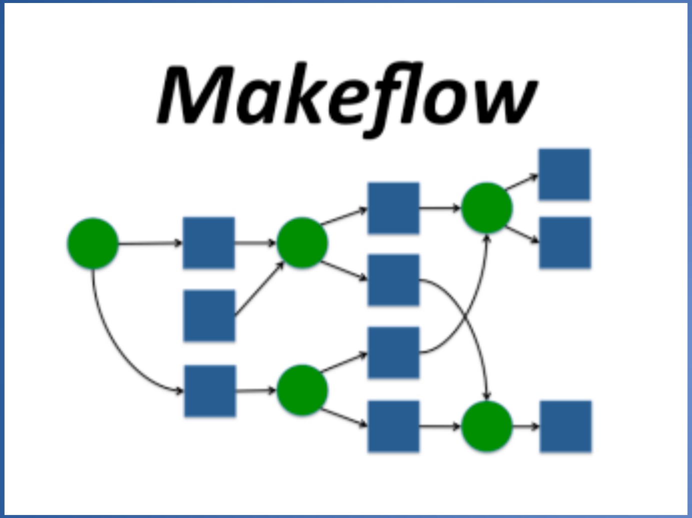
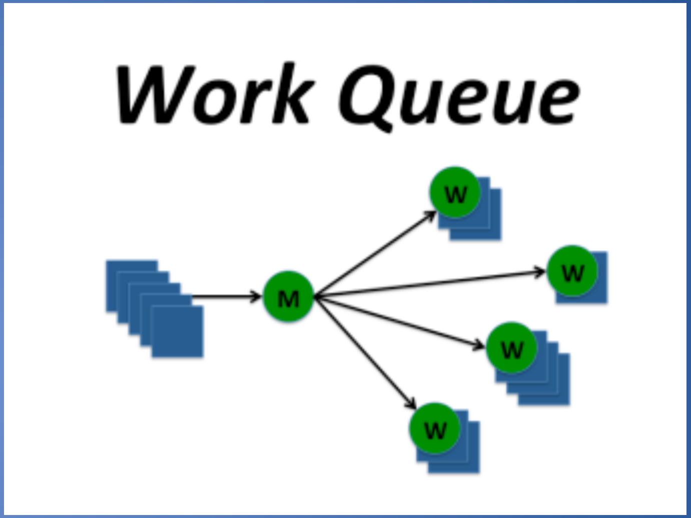
Introduction to Makeflow and Work Queue

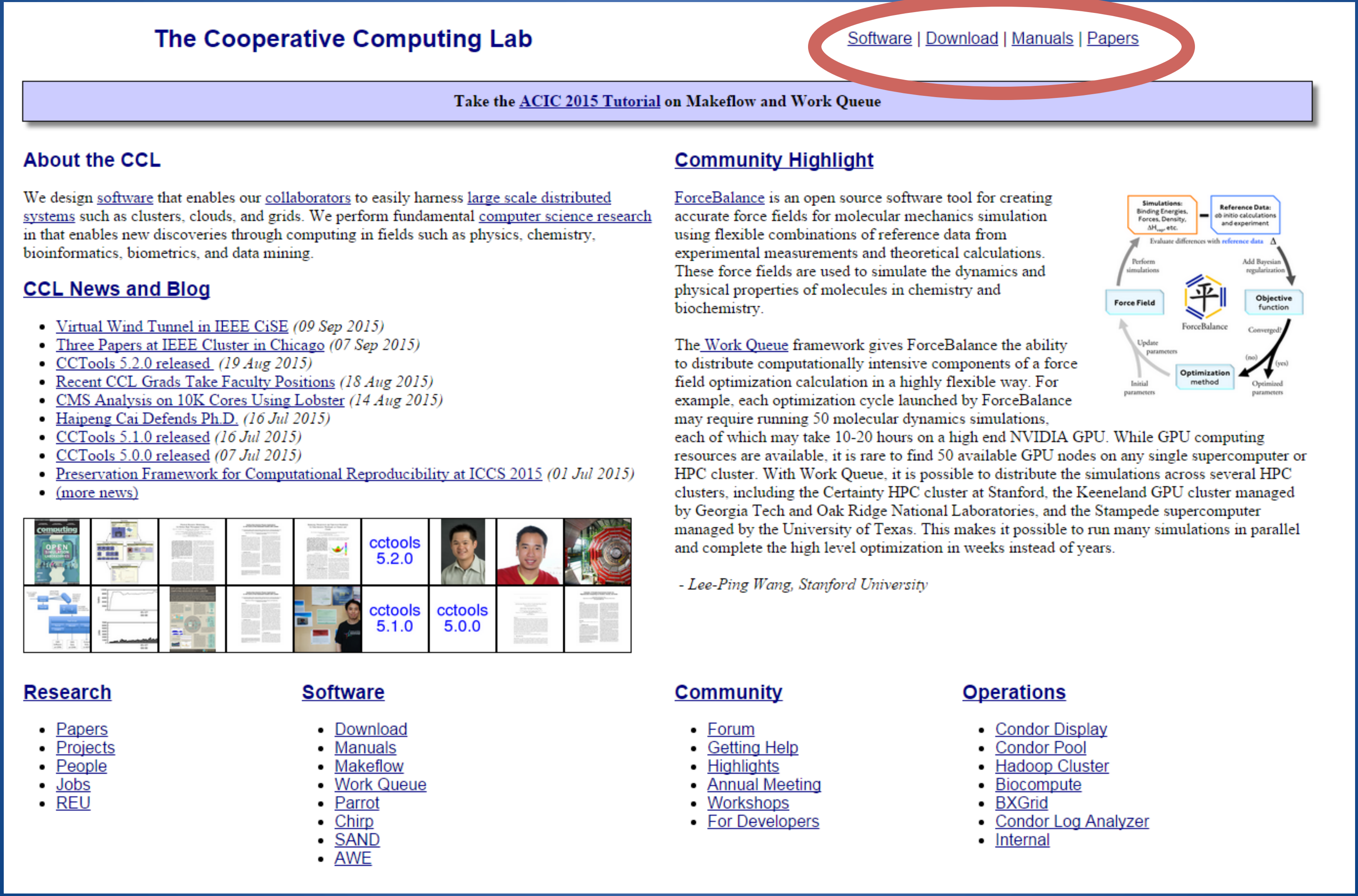




Prof. Douglas Thain, University of Notre Dame

http://www.nd.edu/~dthain
dthain@nd.edu
@ProfThain

Lots of information here: http://ccl.cse.nd.edu



The Cooperative Computing Lab

Software | Download | Manuals | Papers

Take the ACIC 2015 Tutorial on Makeflow and Work Queue

About the CCL

We design software that enables our collaborators to easily harness large scale distributed systems such as clusters, clouds, and grids. We perform fundamental computer science research in that enables new discoveries through computing in fields such as physics, chemistry, bioinformatics, biometrics, and data mining.

CCL News and Blog

- Virtual Wind Tunnel in IEEE CiSE *(09 Sep 2015)*
- Three Papers at IEEE Cluster in Chicago *(07 Sep 2015)*
- CCTools 5.2.0 released *(19 Aug 2015)*
- Recent CCL Grads Take Faculty Positions *(18 Aug 2015)*
- CMS Analysis on 10K Cores Using Lobster *(14 Aug 2015)*
- Haipeng Cai Defends Ph.D. *(16 Jul 2015)*
- CCTools 5.1.0 released *(16 Jul 2015)*
- CCTools 5.0.0 released *(07 Jul 2015)*
- Preservation Framework for Computational Reproducibility at ICCS 2015 *(01 Jul 2015)*
- (more news)

Community Highlight

ForceBalance is an open source software tool for creating accurate force fields for molecular mechanics simulation using flexible combinations of reference data from experimental measurements and theoretical calculations. These force fields are used to simulate the dynamics and physical properties of molecules in chemistry and biochemistry.

Simulations: Binding Energies, Forces, Density, ΔH_vap, etc.
–
Reference Data: ab initio calculations and experiment
Evaluate differences with reference data Δ
Add Bayesian regularization
Objective function
Converged?
(no)
(yes)
Optimized parameters
Optimization method
Initial parameters
Update parameters
Force Field
Perform simulations
ForceBalance

The Work Queue framework gives ForceBalance the ability to distribute computationally intensive components of a force field optimization calculation in a highly flexible way. For example, each optimization cycle launched by ForceBalance may require running 50 molecular dynamics simulations, each of which may take 10-20 hours on a high end NVIDIA GPU. While GPU computing resources are available, it is rare to find 50 available GPU nodes on any single supercomputer or HPC cluster. With Work Queue, it is possible to distribute the simulations across several HPC clusters, including the Certainty HPC cluster at Stanford, the Keeneland GPU cluster managed by Georgia Tech and Oak Ridge National Laboratories, and the Stampede supercomputer managed by the University of Texas. This makes it possible to run many simulations in parallel and complete the high level optimization in weeks instead of years.

- Lee-Ping Wang, Stanford University

Research

- Papers
- Projects
- People
- Jobs
- REU

Software

- Download
- Manuals
- Makeflow
- Work Queue
- Parrot
- Chirp
- SAND
- AWE

Community

- Forum
- Getting Help
- Highlights
- Annual Meeting
- Workshops
- For Developers

Operations

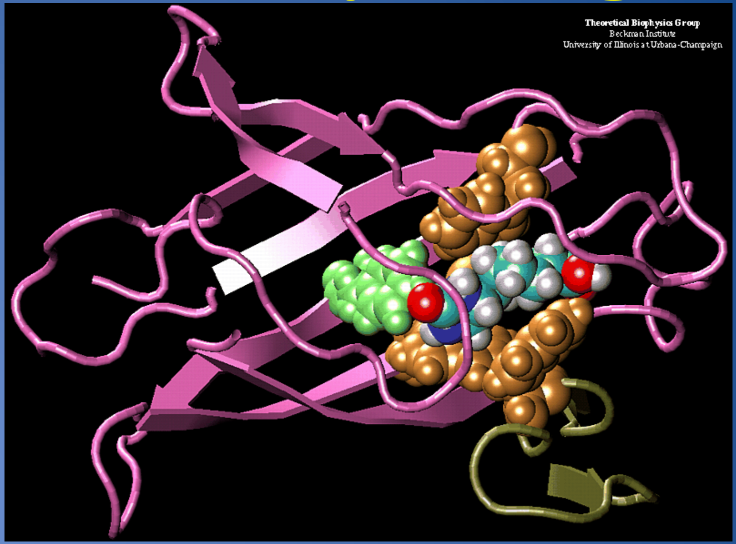
- Condor Display
- Condor Pool
- Hadoop Cluster
- Biocompute
- BXGrid
- Condor Log Analyzer
- Internal

The Cooperative Computing Lab

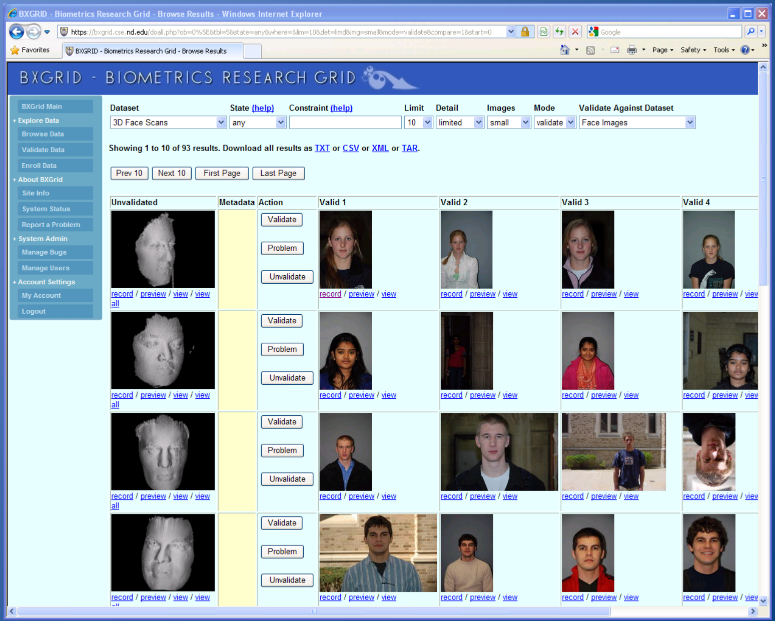
- We *collaborate with people* who have large scale computing problems in science, engineering, and other fields.
- We *operate computer systems* on the O(10,000) cores: clusters, clouds, grids.
- We *conduct computer science* research in the context of real people and problems.
- We *develop open source software* for large scale distributed computing.

http://www.nd.edu/~ccl

Science Depends on Computing!


Theoretical Biophysics Group
Beckman Institute
University of Illinois at Urbana-Champaign

AGTCCGTACGATGCTATTAGCGAGCGTGA…



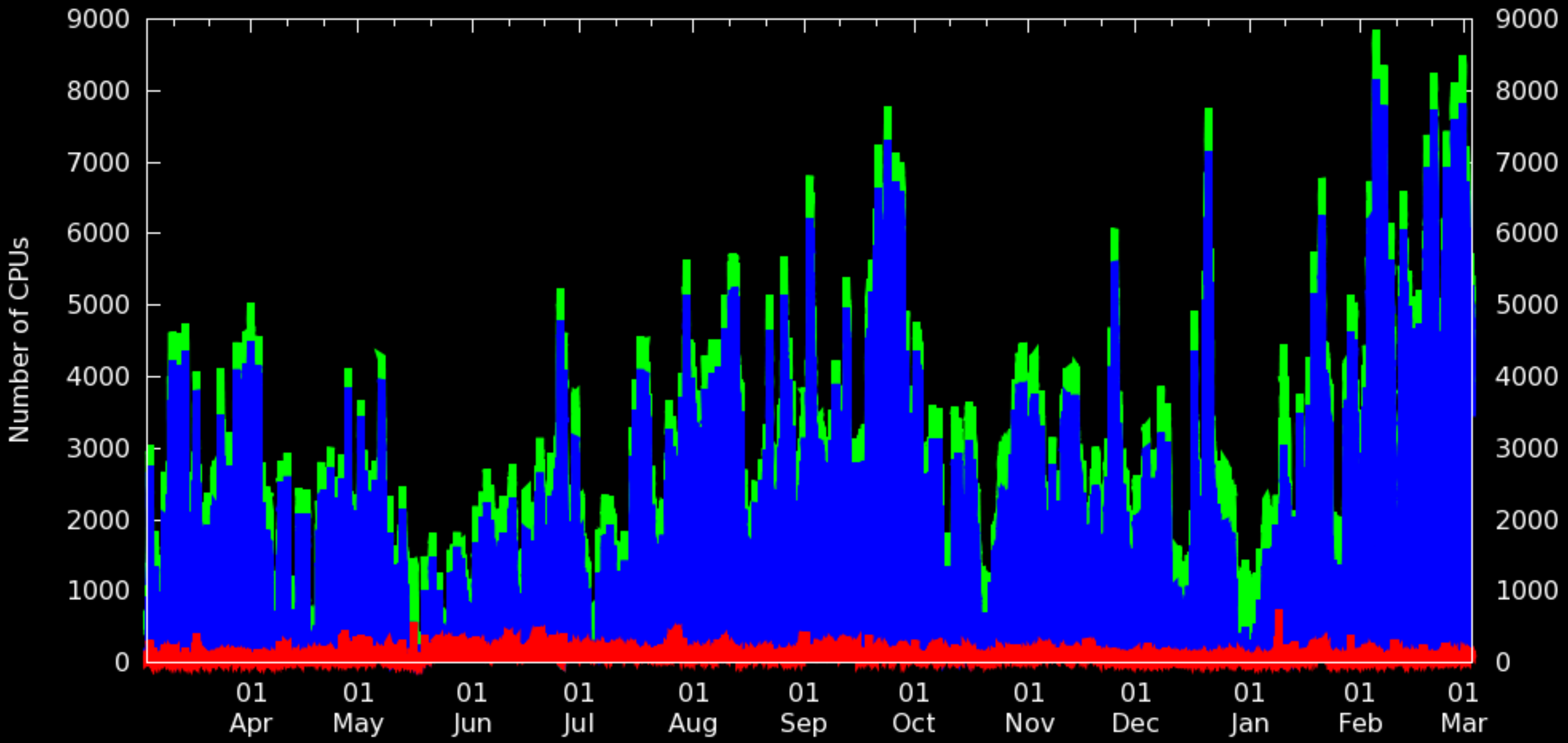
https://bxgrid.cse.nd.edu/doall.php?ob=0%5E&tbl=5&state=any&where=&lim=10&det=limd&img=small&mode=validate&compare=1&start=0

Dataset	State (help)	Constraint (help)	Limit	Detail	Images	Mode	Validate Against Dataset
3D Face Scans	any		10	limited	small	validate	Face Images

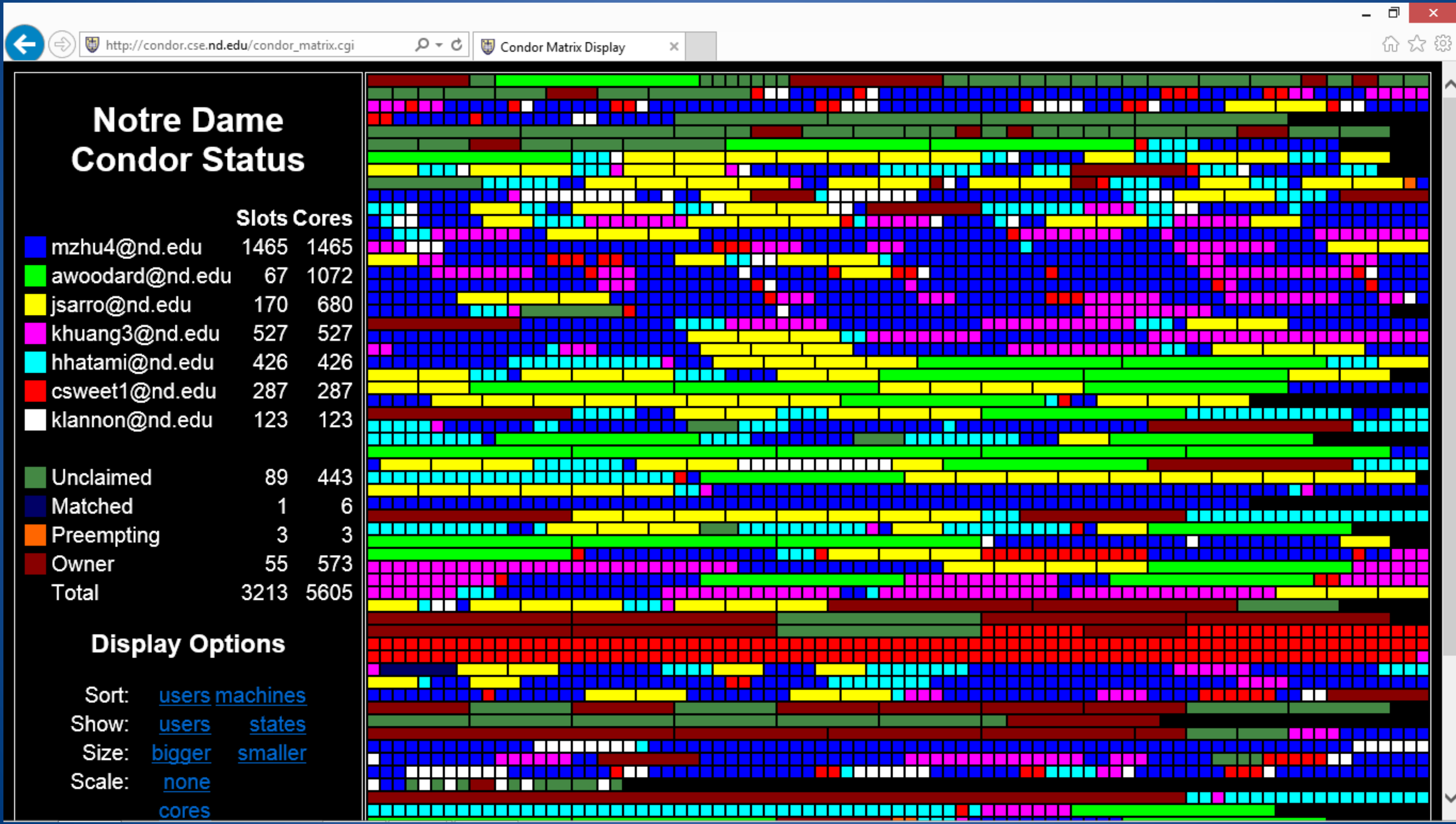
Showing 1 to 10 of 93 results. Download all results as TXT or CSV or XML or TAR.

Unvalidated	Metadata	Action	Valid 1	Valid 2	Valid 3	Valid 4
record / preview / view / view all		Validate Problem Unvalidate	record / preview / view	record / preview / view	record / preview / view	record / preview / view
record / preview / view / view all		Validate Problem Unvalidate	record / preview / view	record / preview / view	record / preview / view	record / preview / view
record / preview / view / view all		Validate Problem Unvalidate	record / preview / view	record / preview / view	record / preview / view	record / preview / view
record / preview / view / view		Validate Problem Unvalidate	record / preview / view	record / preview / view	record / preview / view	record / preview / view

Condor Cycles at Notre Dame



Users of Condor Cycles



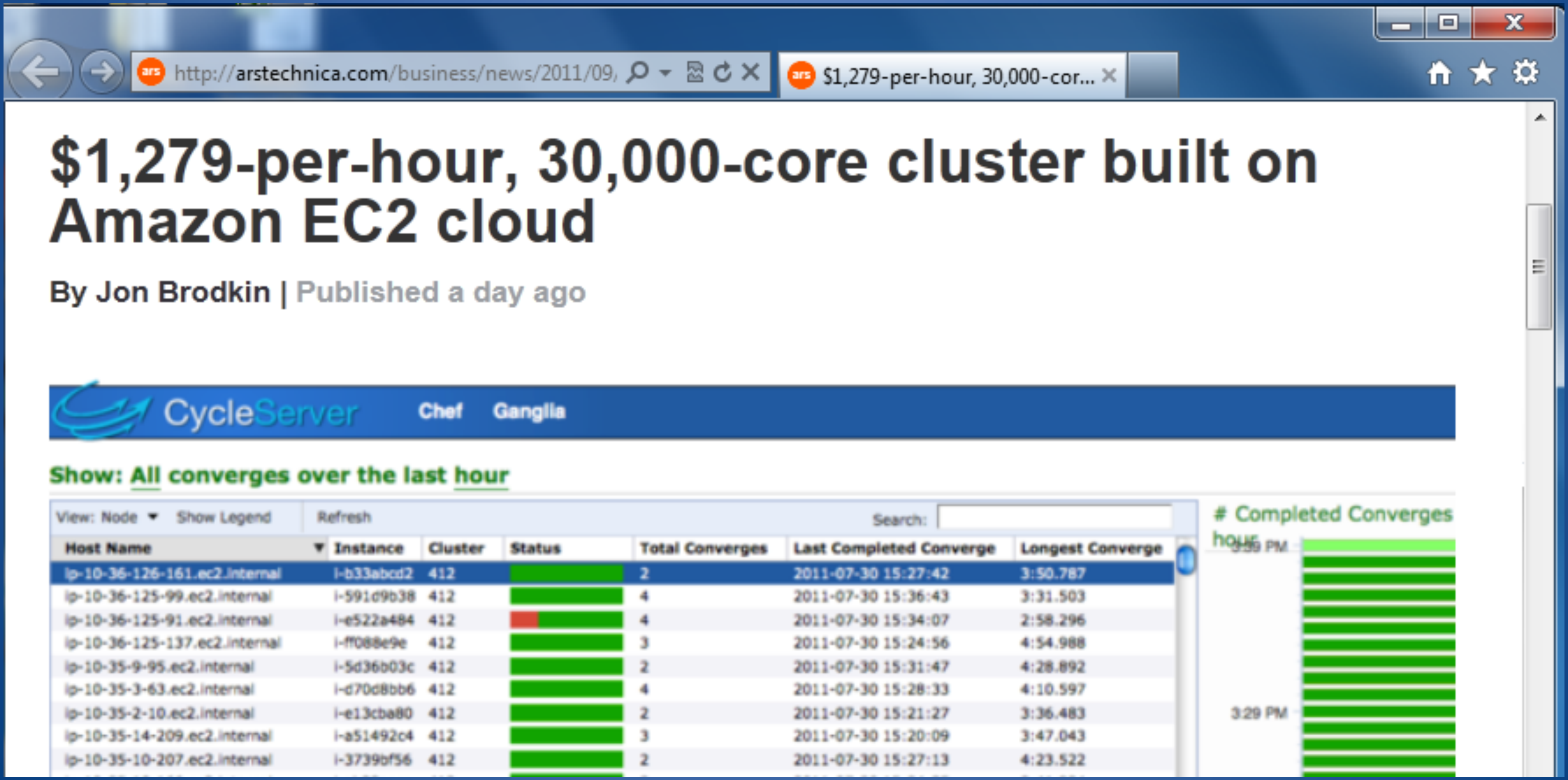
Notre Dame Condor Status

		Slots	Cores
🟦	mzhu4@nd.edu	1465	1465
🟩	awoodard@nd.edu	67	1072
🟨	jsarro@nd.edu	170	680
🟪	khuang3@nd.edu	527	527
🟦	hhatami@nd.edu	426	426
🟥	csweet1@nd.edu	287	287
⬜	klannon@nd.edu	123	123
🟩	Unclaimed	89	443
🟦	Matched	1	6
🟧	Preempting	3	3
🟫	Owner	55	573
	Total	3213	5605

Display Options

Sort: users machines
Show: users states
Size: bigger smaller
Scale: none
cores

Superclusters at Amazon


http://arstechnica.com/business/news/2011/09/
$1,279-per-hour, 30,000-cor...
CycleServer
Chef
Ganglia
# Completed Converges
hour
3:59 PM
3:29 PM

$1,279-per-hour, 30,000-core cluster built on Amazon EC2 cloud

By Jon Brodkin | Published a day ago

Show: All converges over the last hour

View: Node ▾ Show Legend Refresh Search:

Host Name ▾	Instance	Cluster	Status	Total Converges	Last Completed Converge	Longest Converge
ip-10-36-126-161.ec2.internal	i-b33abcd2	412	🟩	2	2011-07-30 15:27:42	3:50.787
ip-10-36-125-99.ec2.internal	i-591d9b38	412	🟩	4	2011-07-30 15:36:43	3:31.503
ip-10-36-125-91.ec2.internal	i-e522a484	412	🟥🟩	4	2011-07-30 15:34:07	2:58.296
ip-10-36-125-137.ec2.internal	i-ff088e9e	412	🟩	3	2011-07-30 15:24:56	4:54.988
ip-10-35-9-95.ec2.internal	i-5d36b03c	412	🟩	2	2011-07-30 15:31:47	4:28.892
ip-10-35-3-63.ec2.internal	i-d70d8bb6	412	🟩	4	2011-07-30 15:28:33	4:10.597
ip-10-35-2-10.ec2.internal	i-e13cba80	412	🟩	2	2011-07-30 15:21:27	3:36.483
ip-10-35-14-209.ec2.internal	i-a51492c4	412	🟩	3	2011-07-30 15:20:09	3:47.043
ip-10-35-10-207.ec2.internal	i-3739bf56	412	🟩	2	2011-07-30 15:27:13	4:23.522

http://arstechnica.com/business/news/2011/09/30000-core-cluster-built-on-amazon-ec2-cloud.ars

I can get as many machines
on the cloud/grid as I want!

How do I organize my application
to run on those machines?

The Cooperative Computing Tools

Our Philosophy:

- Harness all the resources that are available: desktops, clusters, clouds, and grids.
- Make it easy to scale up from one desktop to national scale infrastructure.
- Provide familiar interfaces that make it easy to connect existing apps together.
- Allow portability across operating systems, storage systems, middleware…
- Make simple things easy, and complex things possible.
- ***No special privileges required.***

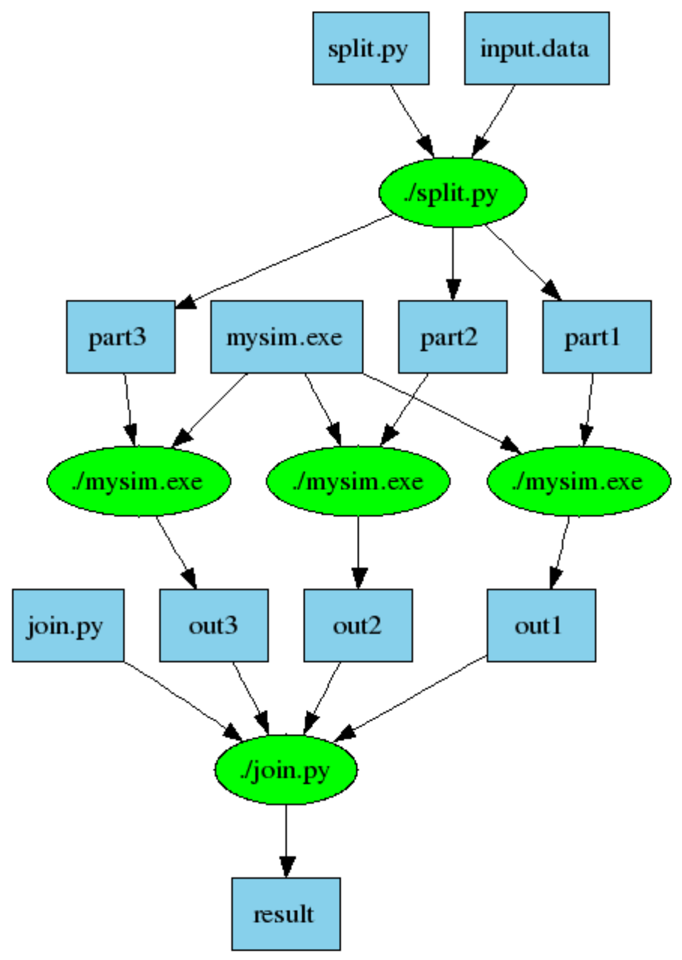
A Quick Tour of the CCTools

- Open source, GNU General Public License.
- Compiles in 1-2 minutes, installs in $HOME.
- Runs on Linux, Solaris, MacOS, Cygwin, FreeBSD, …
- Interoperates with many distributed computing systems.
  - Condor, SGE, Torque, Globus, iRODS, Hadoop…
- Components:
  - Makeflow – A portable workflow manager.
  - Work Queue – A lightweight distributed execution system.
  - All-Pairs / Wavefront / SAND – Specialized execution engines.
  - Parrot – A personal user-level virtual file system.
  - Chirp – A user-level distributed filesystem.

http://ccl.cse.nd.edu/software

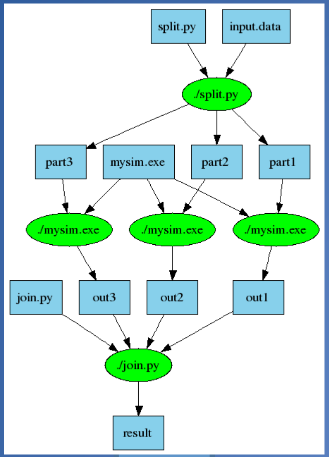
Makeflow:
A Portable Workflow System

An Old Idea: Makefiles

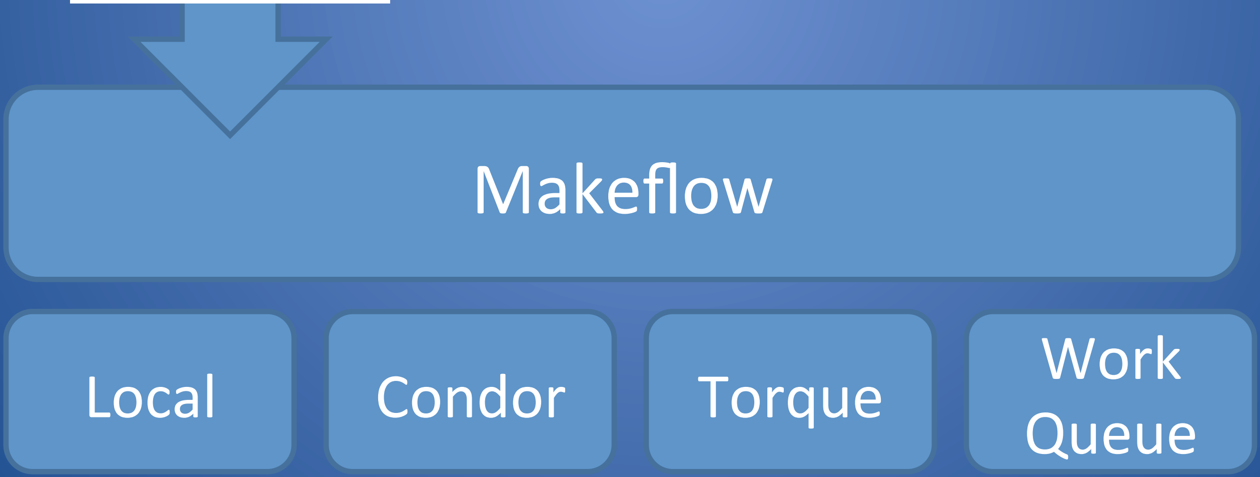


```
part1 part2 part3: input.data split.py
      ./split.py input.data

out1: part1 mysim.exe
     ./mysim.exe part1 >out1

out2: part2 mysim.exe
     ./mysim.exe part2 >out2

out3: part3 mysim.exe
     ./mysim.exe part3 >out3

result: out1 out2 out3 join.py
     ./join.py out1 out2 out3 > result
```

Makeflow = Make + Workflow



- Provides portability across batch systems.
- Enable parallelism (but not too much!)
- Trickle out work to batch system.
- Fault tolerance at multiple scales.
- Data and resource management.



Makeflow

Local

Condor

Torque

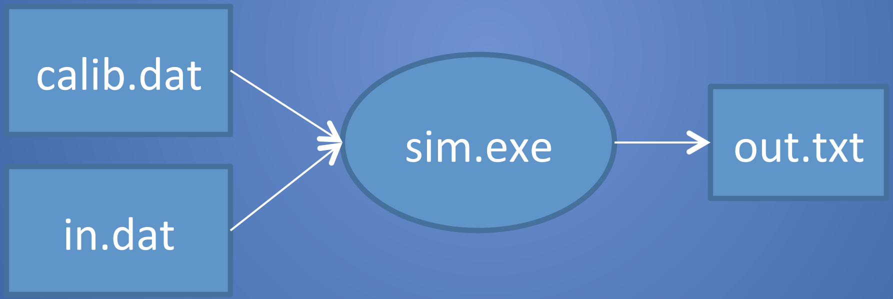
Work Queue

http://ccl.cse.nd.edu/software/makeflow

Makeflow Syntax

```
[output files] : [input files]
        [command to run]
```

One Rule



```
out.txt : calib.dat in.dat sim.exe
        ./sim.exe –p 50 in.data > out.txt
```

You must state
all the files
needed by the command.

sims.mf

```
out.10 : in.dat calib.dat sim.exe
        ./sim.exe –p 10 in.data > out.10


out.20 : in.dat calib.dat sim.exe
        ./sim.exe –p 20 in.data > out.20


out.30 : in.dat calib.dat sim.exe
        ./sim.exe –p 30 in.data > out.30
```

How to run a Makeflow

- Run a workflow locally, using multiple cores:
  - makeflow -T local sims.mf
- Run the workflow on Torque:
  - makeflow –T torque sims.mf
- Run the workflow on Condor:
  - makeflow –T condor sims.mf
- Run the workflow on SLURM:
  - makeflow –T slurm sims.mf

You should see this:

```
% makeflow -T local sims.mf
parsing sims.mf...
checking sims.mf for consistency...
sims.mf has 3 rules.
starting workflow....
submitting job: ./sim.exe –p 30 in.data > out.30
submitted job 2035
submitting job: ./sim.exe –p 20 in.data > out.20
submitted job 2036
submitting job: ./sim.exe –p 10 in.data > out.10
submitted job 2037
job 2035 completed
job 2036 completed
job 2037 completed
nothing left to do.
```

If you do the same thing twice:

```
% makeflow -T local sims.mf
parsing sims.mf...
checking sims.mf for consistency...
sims.mf has 3 rules.
recovering from log file sims.mf.makeflowlog...
starting workflow....
nothing left to do.
```

Makeflow keeps a log of operations, so it knows which jobs have been sent to the batch system, and which files have already been created.

Automatically clean outputs:

```
% makeflow --clean sims.mf
parsing sims.mf...
checking sims.mf for consistency...
sims.mf has 3 rules.
recovering from log file sims.mf.makeflowlog...
cleaning filesystem...
```

Note that you do *not* have to write a “clean” rule.
Makeflow just figures it out for you.

Some more handy options:

- Limit the number of jobs running at once:

```
--max-local #
--max-remote #
```

- Retry jobs that have a tendency to fail:

```
--retry-count=5
```

- Send email when the workflow is done:

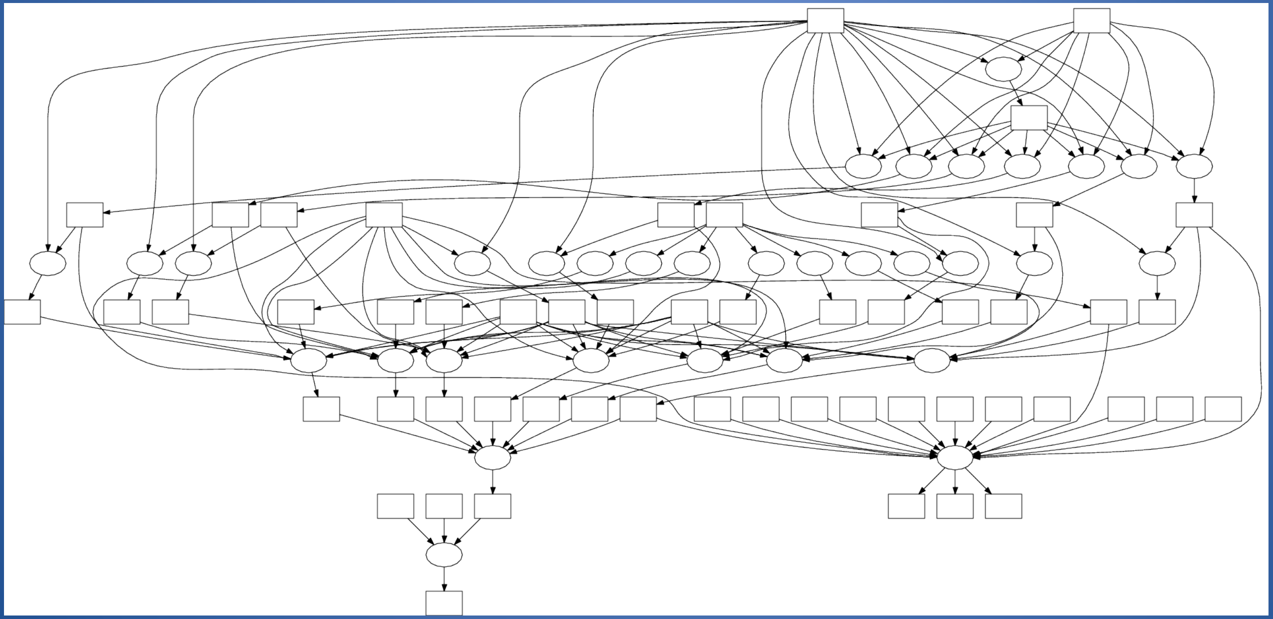
```
--email user@domain.com
```

- Monitor the resources consumed by each job:

```
--monitor <output-dir>
```

Visualization with DOT

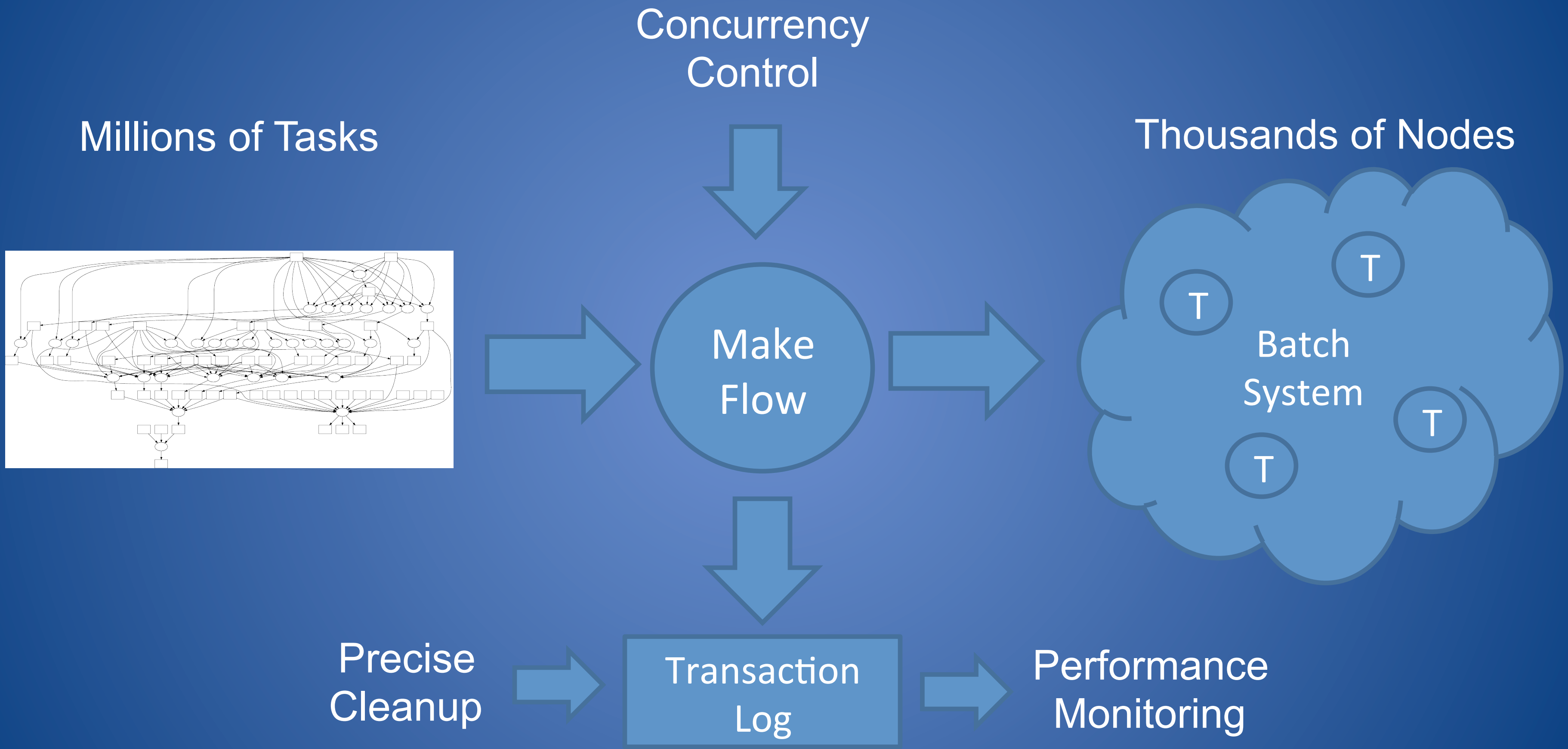
- makeflow_viz –D example.mf > example.dot
- dot –T gif < example.dot > example.gif



DOT and related tools:

http://www.graphviz.org

Makeflow Shapes a Workflow



Concurrency Control

Millions of Tasks

Thousands of Nodes

Make Flow

Batch System

T

T

T

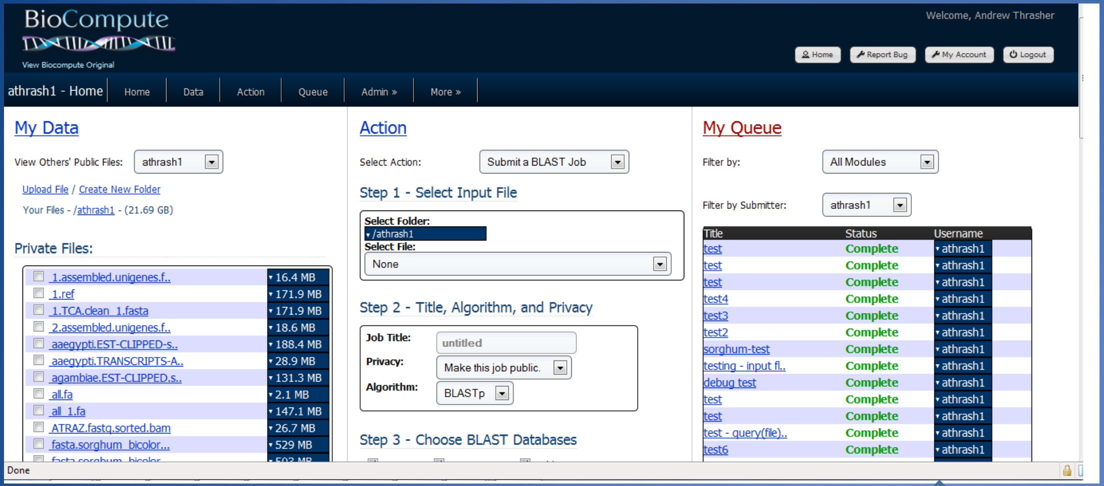
T

Precise Cleanup

Transaction Log

Performance Monitoring

Example: Biocompute Portal



BioCompute

View Biocompute Original

Welcome, Andrew Thrasher

My Data

View Others' Public Files: athrash1

Upload File / Create New Folder

Your Files - /athrash1 - (21.69 GB)

Private Files:

□ 1.assembled.unigenes.f..	16.4 MB
□ 1.ref	171.9 MB
□ 1.TCA.clean_1.fasta	171.9 MB
□ 2.assembled.unigenes.f..	18.6 MB
□ aaegypti.EST-CLIPPED-s..	188.4 MB
□ aaegypti.TRANSCRIPTS-A..	28.9 MB
□ agambiae.EST-CLIPPED.s..	131.3 MB
□ all.fa	2.1 MB
□ all_1.fa	147.1 MB
□ ATRAZ.fastq.sorted.bam	26.7 MB
□ fasta.sorghum_bicolor...	529 MB
□ fasta.sorghum_bicolor	503 MB

Action

Select Action: Submit a BLAST Job

Step 1 - Select Input File

Select Folder: /athrash1

Select File: None

Step 2 - Title, Algorithm, and Privacy

Job Title: untitled

Privacy: Make this job public.

Algorithm: BLASTp

Step 3 - Choose BLAST Databases

My Queue

Filter by: All Modules

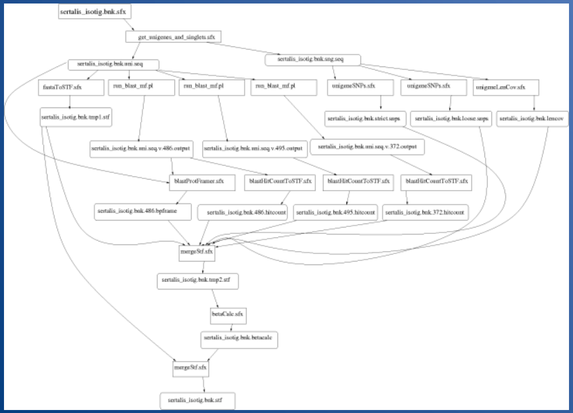
Filter by Submitter: athrash1

Title	Status	Username
test	Complete	athrash1
test	Complete	athrash1
test	Complete	athrash1
test4	Complete	athrash1
test3	Complete	athrash1
test2	Complete	athrash1
sorghum-test	Complete	athrash1
testing - input fl..	Complete	athrash1
debug test	Complete	athrash1
test	Complete	athrash1
test	Complete	athrash1
test - query(file)..	Complete	athrash1
test6	Complete	athrash1

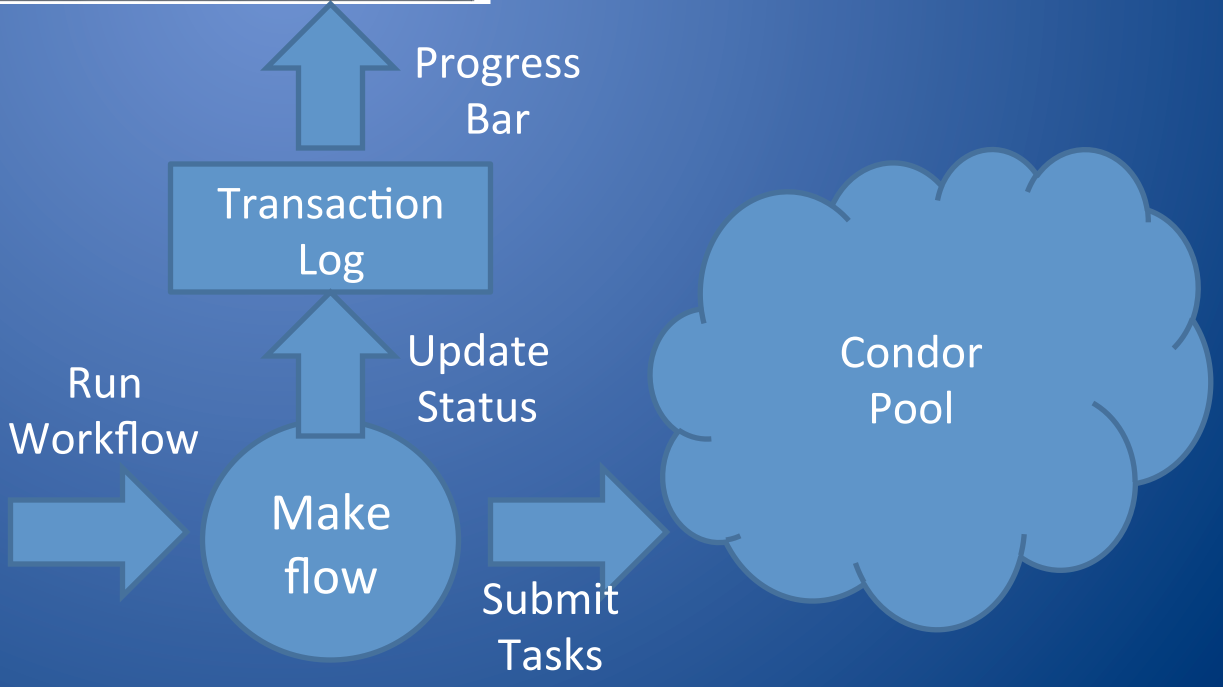
Done

BLAST
SSAHA
SHRIMP
EST
MAKER
…

Generate Makefile



sertalis_isotig.bnk.sfx
get_unigenes_and_singlets.sfx
sertalis_isotig.bnk.uni.seq
sertalis_isotig.bnk.sng.seq
fastaToSTF.sfx
run_blast_mf.pl
run_blast_mf.pl
run_blast_mf.pl
unigeneSNPs.sfx
unigeneSNPs.sfx
unigeneLenCov.sfx
sertalis_isotig.bnk.tmp1.stf
sertalis_isotig.bnk.strict.snps
sertalis_isotig.bnk.loose.snps
sertalis_isotig.bnk.lencov
sertalis_isotig.bnk.uni.seq.v.486.output
sertalis_isotig.bnk.uni.seq.v.495.output
sertalis_isotig.bnk.uni.seq.v.372.output
blastProtFramer.sfx
blastHitCountToSTF.sfx
blastHitCountToSTF.sfx
blastHitCountToSTF.sfx
sertalis_isotig.bnk.486.bpframe
sertalis_isotig.bnk.486.hitcount
sertalis_isotig.bnk.495.hitcount
sertalis_isotig.bnk.372.hitcount
mergeStf.sfx
sertalis_isotig.bnk.tmp2.stf
betaCalc.sfx
sertalis_isotig.bnk.betacalc
mergeStf.sfx
sertalis_isotig.bnk.stf



Progress Bar

Transaction Log

Update Status

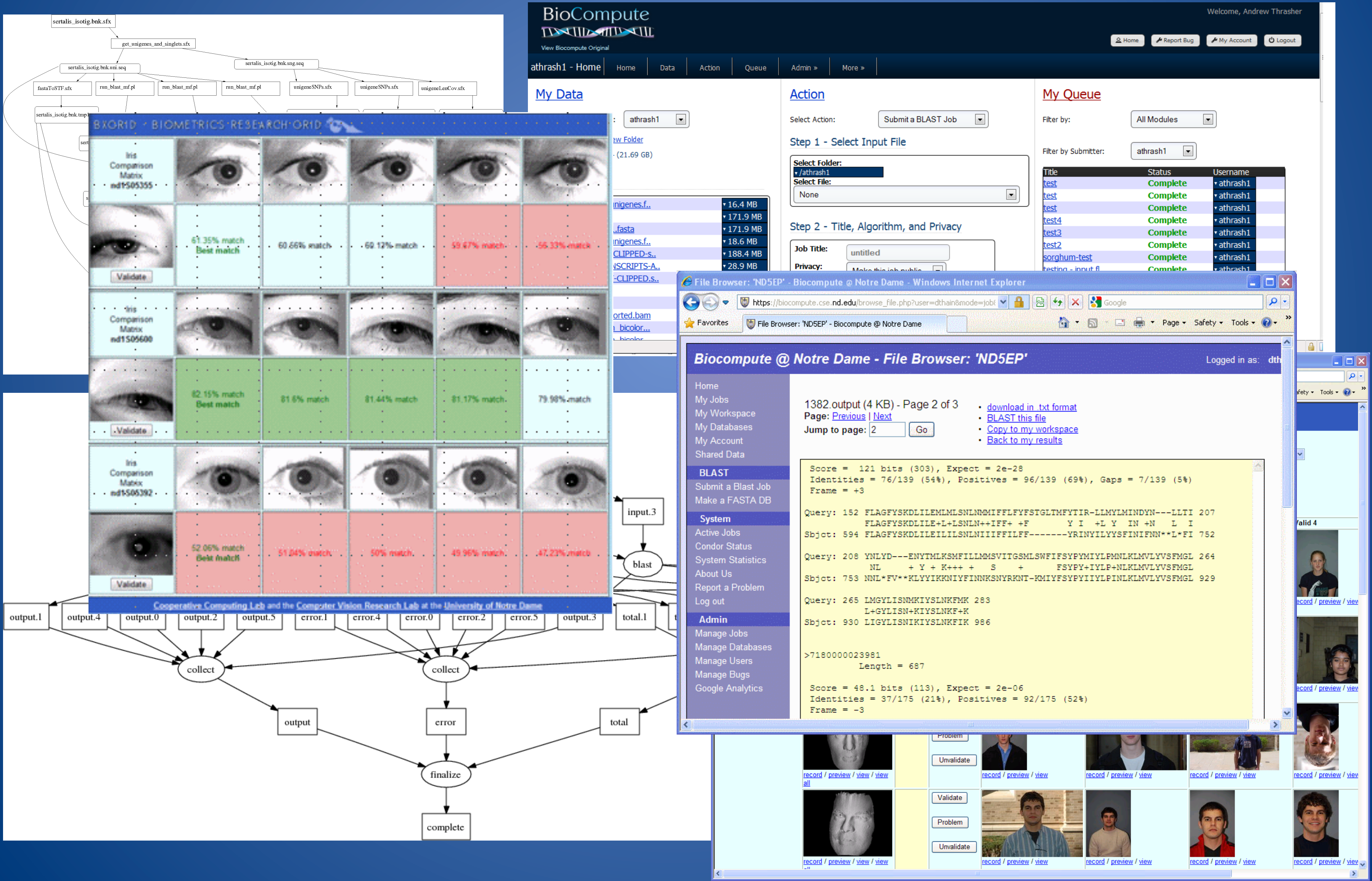
Run Workflow

Make flow

Submit Tasks

Condor Pool

Makeflow Applications



Title	Status	Username
test	Complete	athrash1
test	Complete	athrash1
test	Complete	athrash1
test4	Complete	athrash1
test3	Complete	athrash1
test2	Complete	athrash1
sorghum-test	Complete	athrash1
testing - input fi	Complete	athrash1

1382.output (4 KB) - Page 2 of 3

Page: Previous | Next

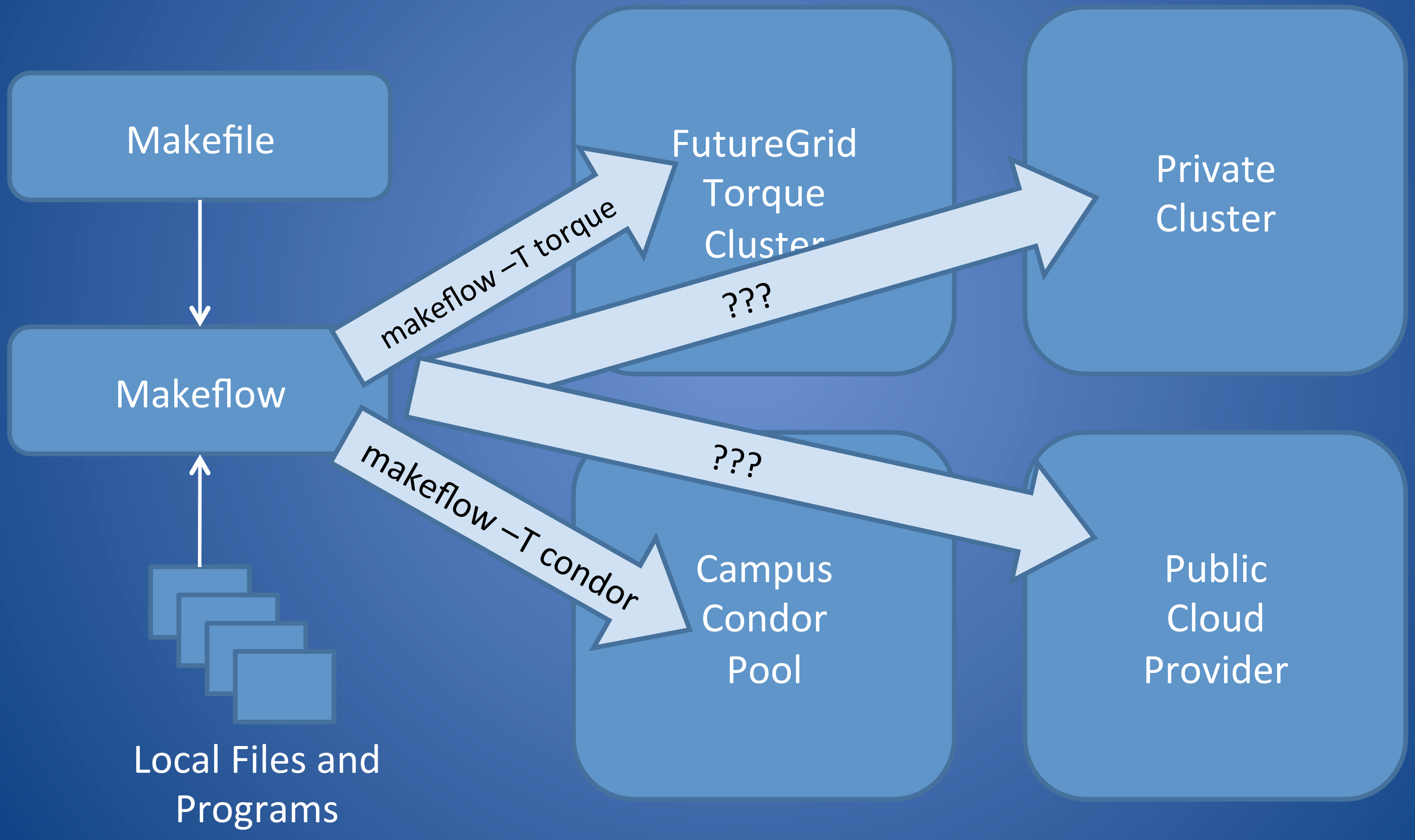
Jump to page: 2 Go

- download in .txt format
- BLAST this file
- Copy to my workspace
- Back to my results

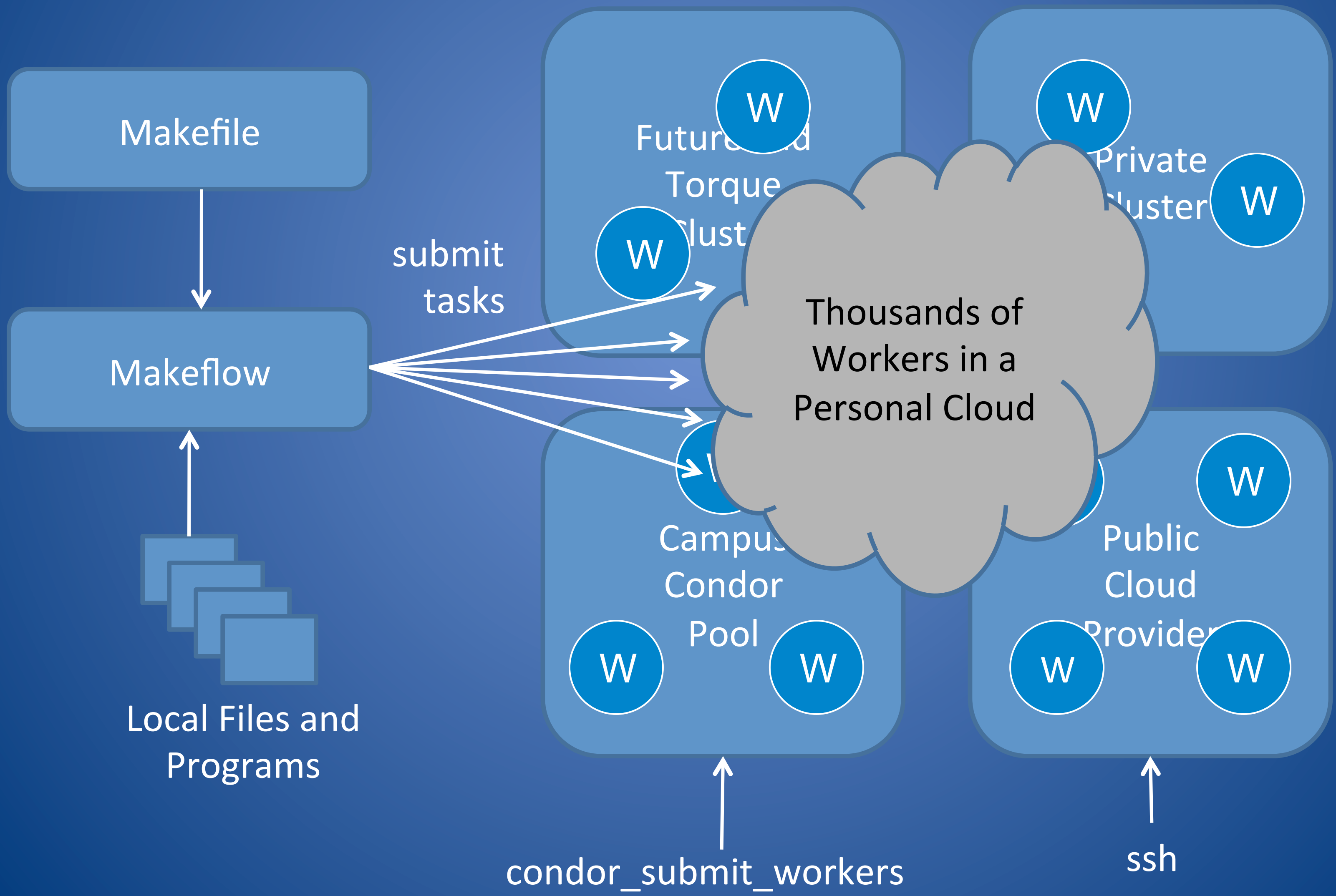
```
 Score =  121 bits (303), Expect = 2e-28
 Identities = 76/139 (54%), Positives = 96/139 (69%), Gaps = 7/139 (5%)
 Frame = +3

Query: 152 FLAGFYSKDLILEMLMLSNLNMMIFFLFYFSTGLTMFYTIR-LLMYLMINDYN---LLTI 207
           FLAGFYSKDLILE+L+LSNLN++IFF+ +F       Y I  +L Y  IN +N    L  I
Sbjct: 594 FLAGFYSKDLILEILILSNLNIIIFFILFF-------YRINYILYSFINIFNN**L*FI 752

Query: 208 YNLYD---ENYTMLKSMFILLMMSVITGSMLSWFIFSYPYMIYLPMNLKLMVLYVSFMGL 264
           NL      + Y + K+++ +    S    +         FSYPY+IYLP+NLKLMVLYVSFMGL
Sbjct: 753 NNL*FV**KLYYIKKNIYFINNKSNYRKNT-KMIYFSYPYIIYLPINLKLMVLYVSFMGL 929

Query: 265 LMGYLISNMKIYSLNKFMK 283
           L+GYLISN+KIYSLNKF+K
Sbjct: 930 LIGYLISNIKIYSLNKFIK 986


>7180000023981
          Length = 687

 Score = 48.1 bits (113), Expect = 2e-06
 Identities = 37/175 (21%), Positives = 92/175 (52%)
 Frame = -3
```

Makeflow + Work Queue

Makeflow can send jobs to one batch system at a time.



Makeflow + Work Queue can harness multiple clusters at once.



Advantages of Work Queue

- Harness multiple resources simultaneously.
- Hold on to cluster nodes to execute multiple tasks rapidly. (ms/task instead of min/task)
- Scale resources up and down as needed.
- Better management of data, with local caching for data intensive tasks.
- Matching of tasks to nodes with data.

Makeflow and Work Queue

First, start the Makeflow:

```
% makeflow –T wq  sims.mf
Could not create work queue on port 9123.
```

Whoops, try again:

```
% makeflow –T wq --port 0 sims.mf
Listening for workers on port 8374…
```

Start one worker and tell it where to find makeflow:

```
% work_queue_worker  master.hostname.org 8374
```

Start 25 Workers in Batch System

Submit workers to Condor:

```
condor_submit_workers master.hostname.org 8374 25
```

Submit workers to SGE:

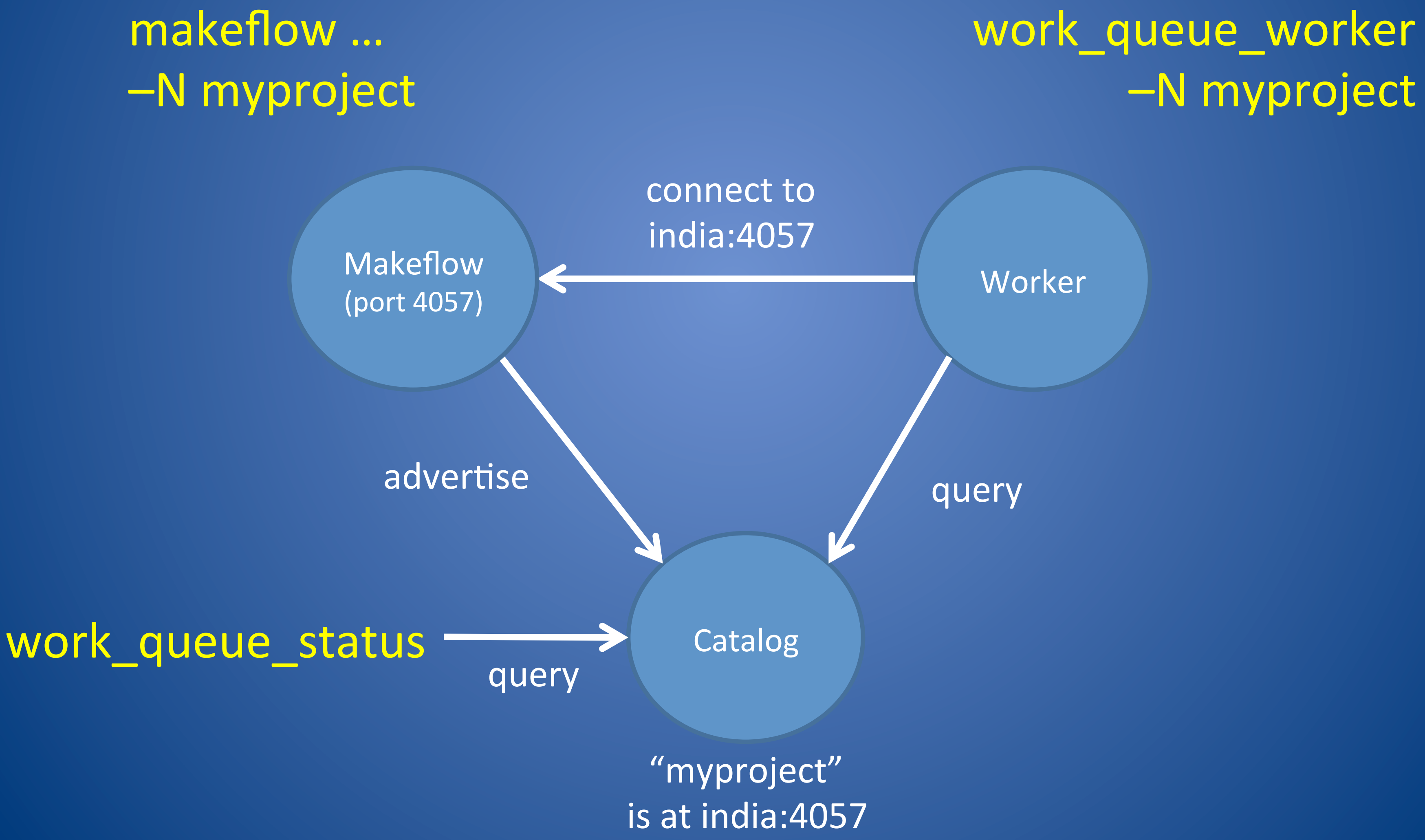
```
sge_submit_workers master.hostname.org 8374 25
```

Submit workers to Torque:

```
torque_submit_workers master.hostname.org 8374 25
```

Keeping track of port numbers gets old fast…

Project Names



makeflow …
–N myproject

work_queue_worker
–N myproject

connect to
india:4057

Makeflow
(port 4057)

Worker

advertise

query

work_queue_status

query

Catalog

“myproject”
is at india:4057

Project Names

Start Makeflow with a project name:

```
% makeflow –T wq –N myproject  sims.mf
Listening for workers on port XYZ…
```

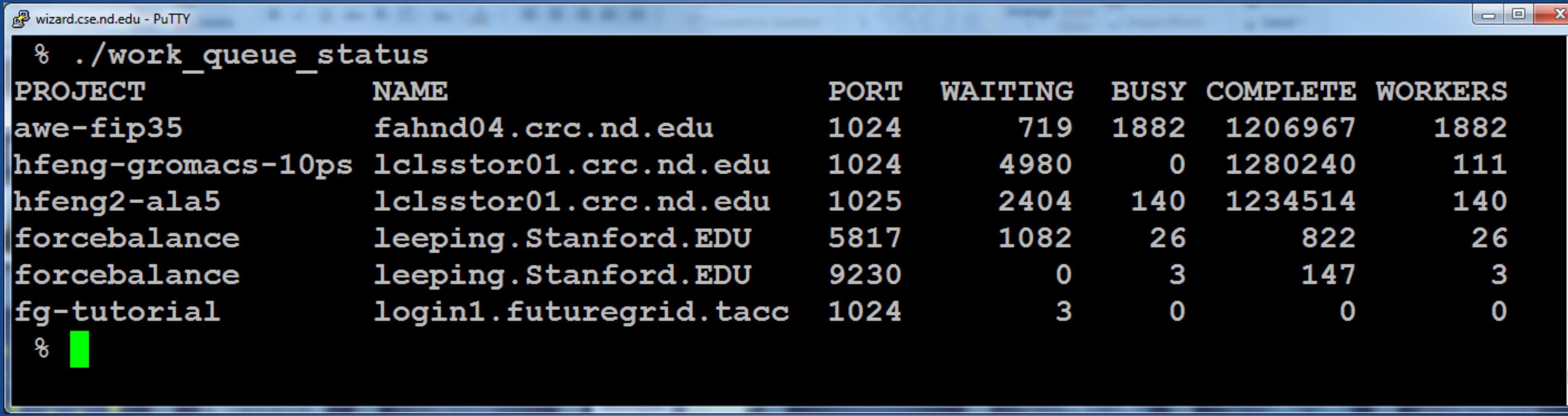
Start one worker:

```
% work_queue_worker -N myproject
```

Start many workers:

```
% torque_submit_workers –N myproject  5
```

work_queue_status

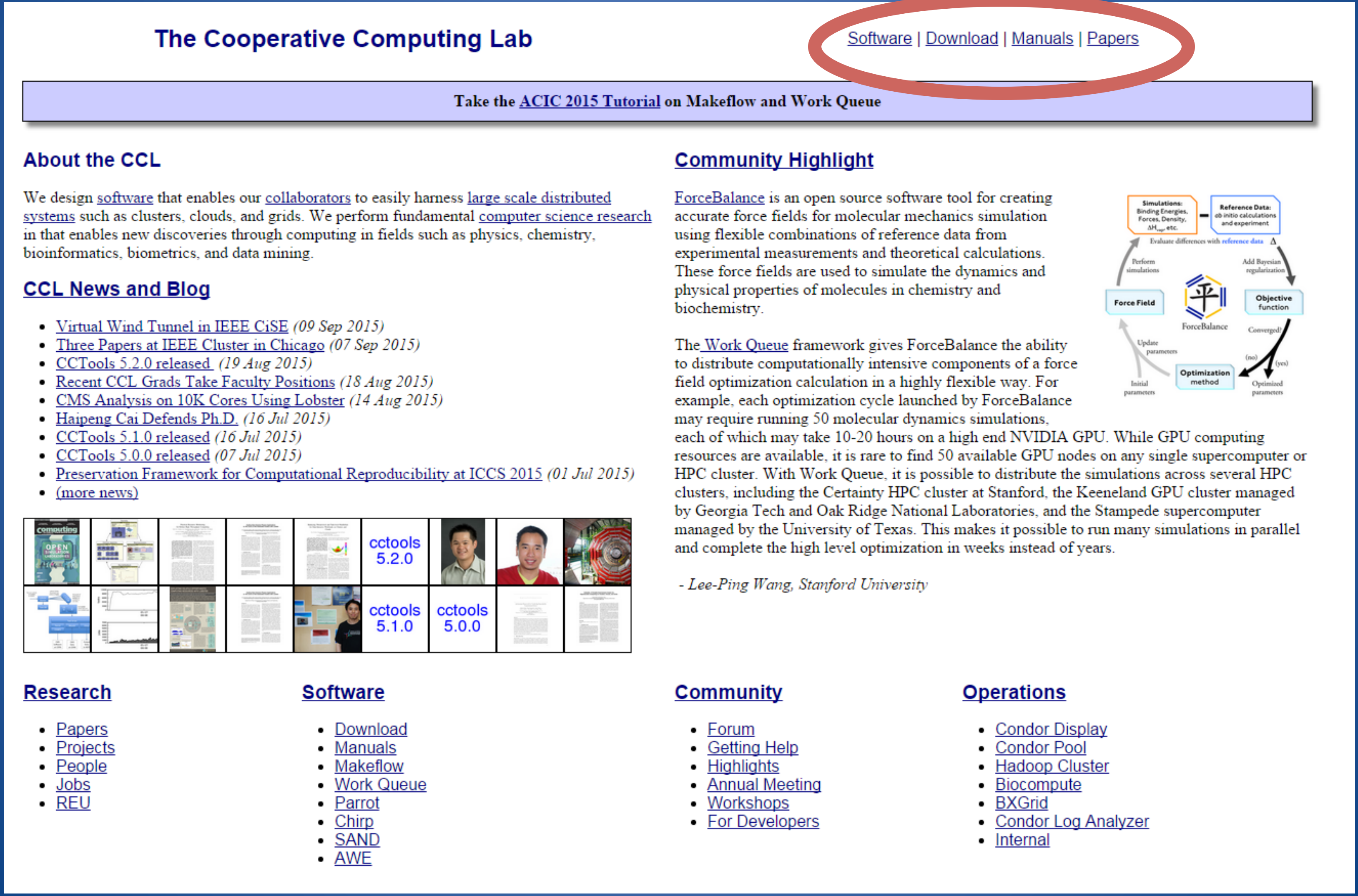


```
 % ./work_queue_status
PROJECT            NAME                     PORT  WAITING  BUSY COMPLETE WORKERS
awe-fip35          fahnd04.crc.nd.edu       1024      719  1882  1206967    1882
hfeng-gromacs-10ps lclsstor01.crc.nd.edu    1024     4980     0  1280240     111
hfeng2-ala5        lclsstor01.crc.nd.edu    1025     2404   140  1234514     140
forcebalance       leeping.Stanford.EDU     5817     1082    26      822      26
forcebalance       leeping.Stanford.EDU     9230        0     3      147       3
fg-tutorial        login1.futuregrid.tacc   1024        3     0        0       0
 %
```

Resilience and Fault Tolerance

- MF +WQ is fault tolerant in many different ways:
  - If Makeflow crashes (or is killed) at any point, it will recover by reading the transaction log and continue where it left off.
  - Makeflow keeps statistics on both network and task performance, so that excessively bad workers are avoided.
  - If a worker crashes, the master will detect the failure and restart the task elsewhere.
  - Workers can be added and removed at any time during the execution of the workflow.
  - Multiple masters with the same project name can be added and removed while the workers remain.
  - If the worker sits idle for too long (default 15m) it will exit, so as not to hold resources idle.

Lots more information here: http://ccl.cse.nd.edu



The Cooperative Computing Lab

Software | Download | Manuals | Papers

Take the ACIC 2015 Tutorial on Makeflow and Work Queue

About the CCL

We design software that enables our collaborators to easily harness large scale distributed systems such as clusters, clouds, and grids. We perform fundamental computer science research in that enables new discoveries through computing in fields such as physics, chemistry, bioinformatics, biometrics, and data mining.

CCL News and Blog

- Virtual Wind Tunnel in IEEE CiSE *(09 Sep 2015)*
- Three Papers at IEEE Cluster in Chicago *(07 Sep 2015)*
- CCTools 5.2.0 released *(19 Aug 2015)*
- Recent CCL Grads Take Faculty Positions *(18 Aug 2015)*
- CMS Analysis on 10K Cores Using Lobster *(14 Aug 2015)*
- Haipeng Cai Defends Ph.D. *(16 Jul 2015)*
- CCTools 5.1.0 released *(16 Jul 2015)*
- CCTools 5.0.0 released *(07 Jul 2015)*
- Preservation Framework for Computational Reproducibility at ICCS 2015 *(01 Jul 2015)*
- (more news)

Community Highlight

ForceBalance is an open source software tool for creating accurate force fields for molecular mechanics simulation using flexible combinations of reference data from experimental measurements and theoretical calculations. These force fields are used to simulate the dynamics and physical properties of molecules in chemistry and biochemistry.

The Work Queue framework gives ForceBalance the ability to distribute computationally intensive components of a force field optimization calculation in a highly flexible way. For example, each optimization cycle launched by ForceBalance may require running 50 molecular dynamics simulations, each of which may take 10-20 hours on a high end NVIDIA GPU. While GPU computing resources are available, it is rare to find 50 available GPU nodes on any single supercomputer or HPC cluster. With Work Queue, it is possible to distribute the simulations across several HPC clusters, including the Certainty HPC cluster at Stanford, the Keeneland GPU cluster managed by Georgia Tech and Oak Ridge National Laboratories, and the Stampede supercomputer managed by the University of Texas. This makes it possible to run many simulations in parallel and complete the high level optimization in weeks instead of years.

- Lee-Ping Wang, Stanford University

Simulations: Binding Energies, Forces, Density, ΔH_vap, etc.
−
Reference Data: ab initio calculations and experiment
Evaluate differences with reference data
Δ
Perform simulations
Add Bayesian regularization
Force Field
平
ForceBalance
Objective function
Converged?
Update parameters
(no)
(yes)
Initial parameters
Optimization method
Optimized parameters

Research

- Papers
- Projects
- People
- Jobs
- REU

Software

- Download
- Manuals
- Makeflow
- Work Queue
- Parrot
- Chirp
- SAND
- AWE

Community

- Forum
- Getting Help
- Highlights
- Annual Meeting
- Workshops
- For Developers

Operations

- Condor Display
- Condor Pool
- Hadoop Cluster
- Biocompute
- BXGrid
- Condor Log Analyzer
- Internal