Changing landscape of computing at BNL

Shared Pool and New Users and Tools

HTCondor Week
May 2018

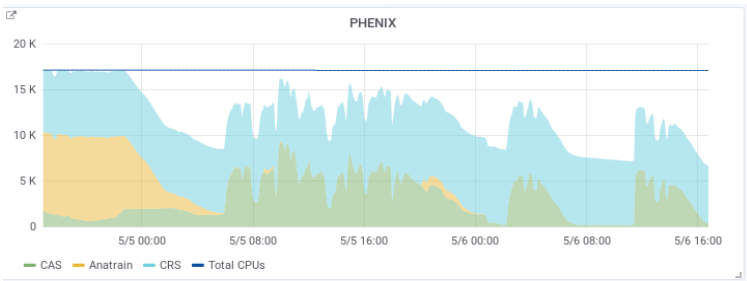
William Strecker-Kellogg <willsk@bnl.gov>

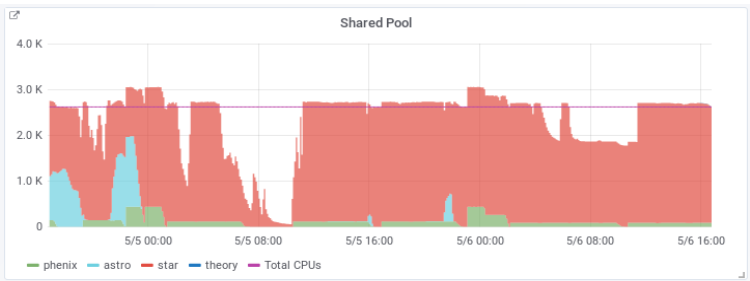
Shared Pool

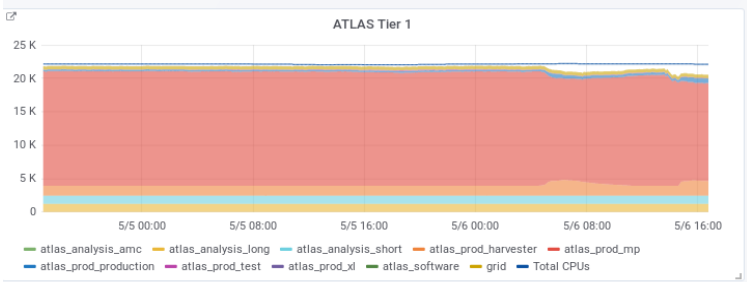
Merging 6 HTCondor Pools into 1

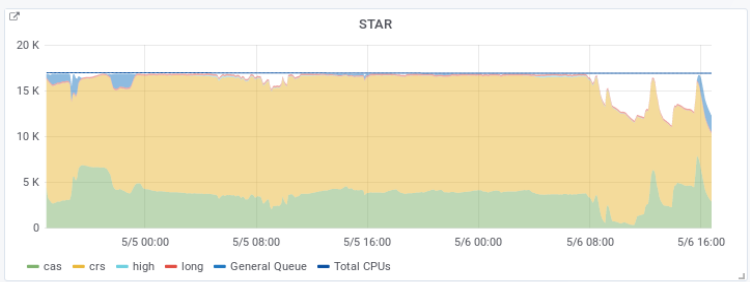
What?

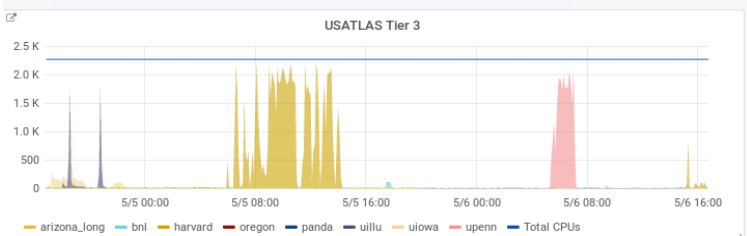
- **Current Situation**
  - Many pools, not all as well utilized as possible
  - Wildly different policies
  - Emulate “queues” with policy statements

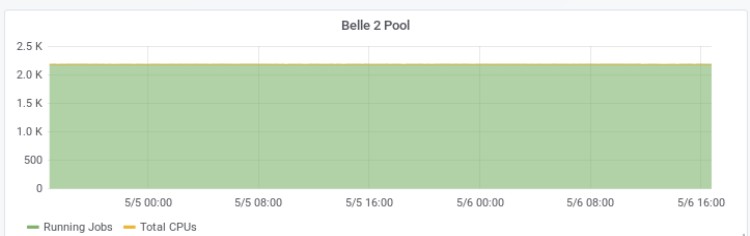




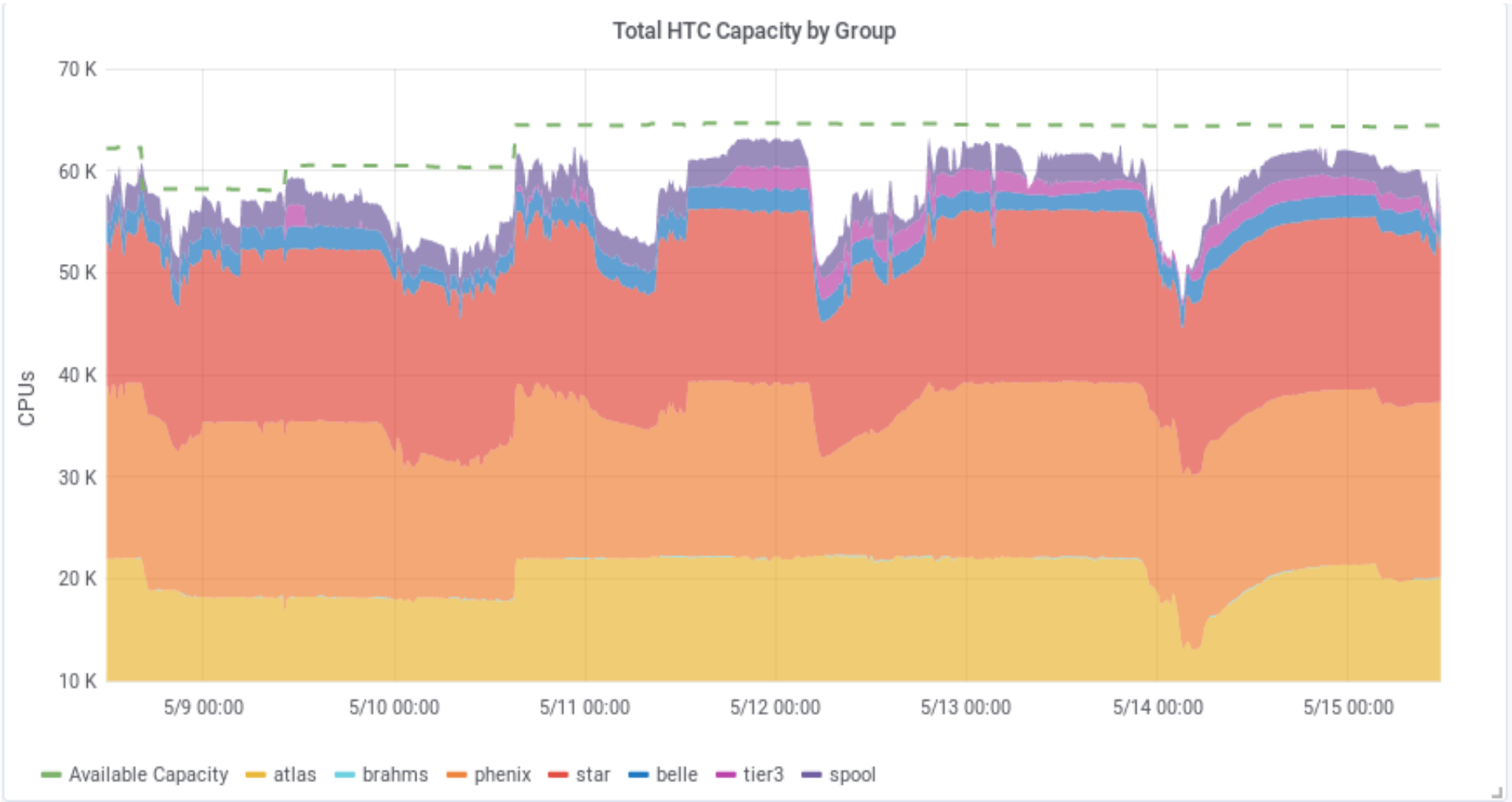








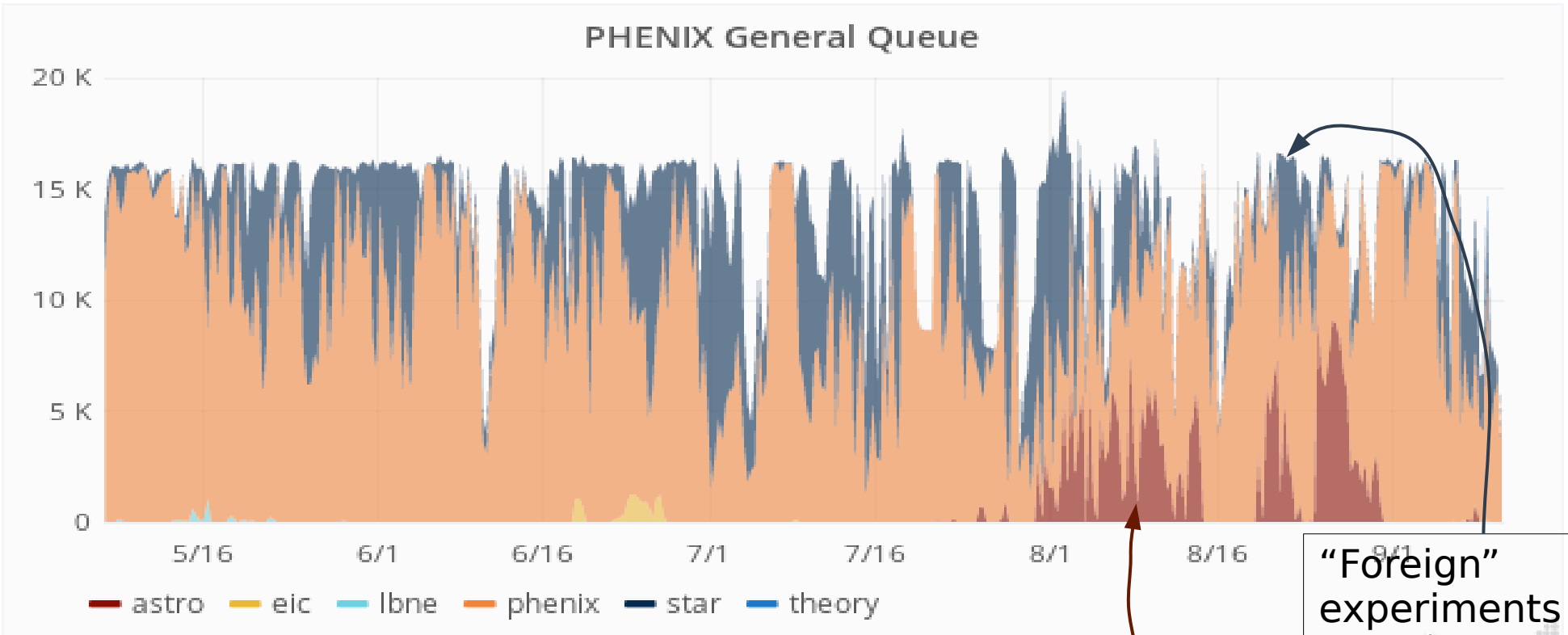
Current Situation



What?

- **Sharing done between some pools with HTCondor Flocking**
  - Policy requirements for “general queue” jobs much stricter than for native jobs
  - Collaborations negotiate with other over these parameters
- **Not possible for many stakeholders**
  - E.g. ATLAS, using group quotas and auto-balancing (see my previous talks)

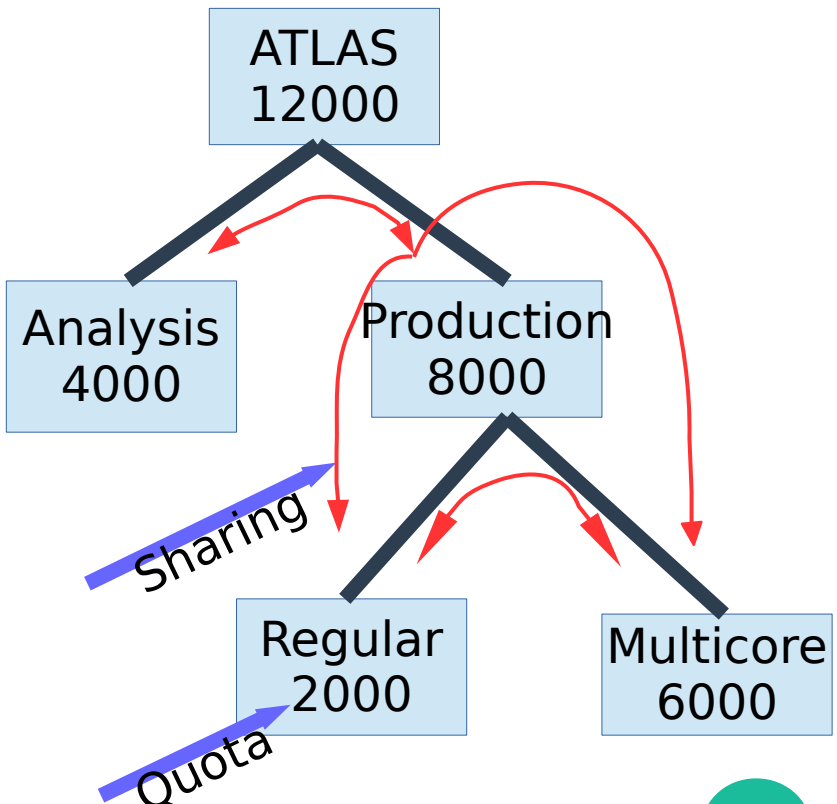
Flocking



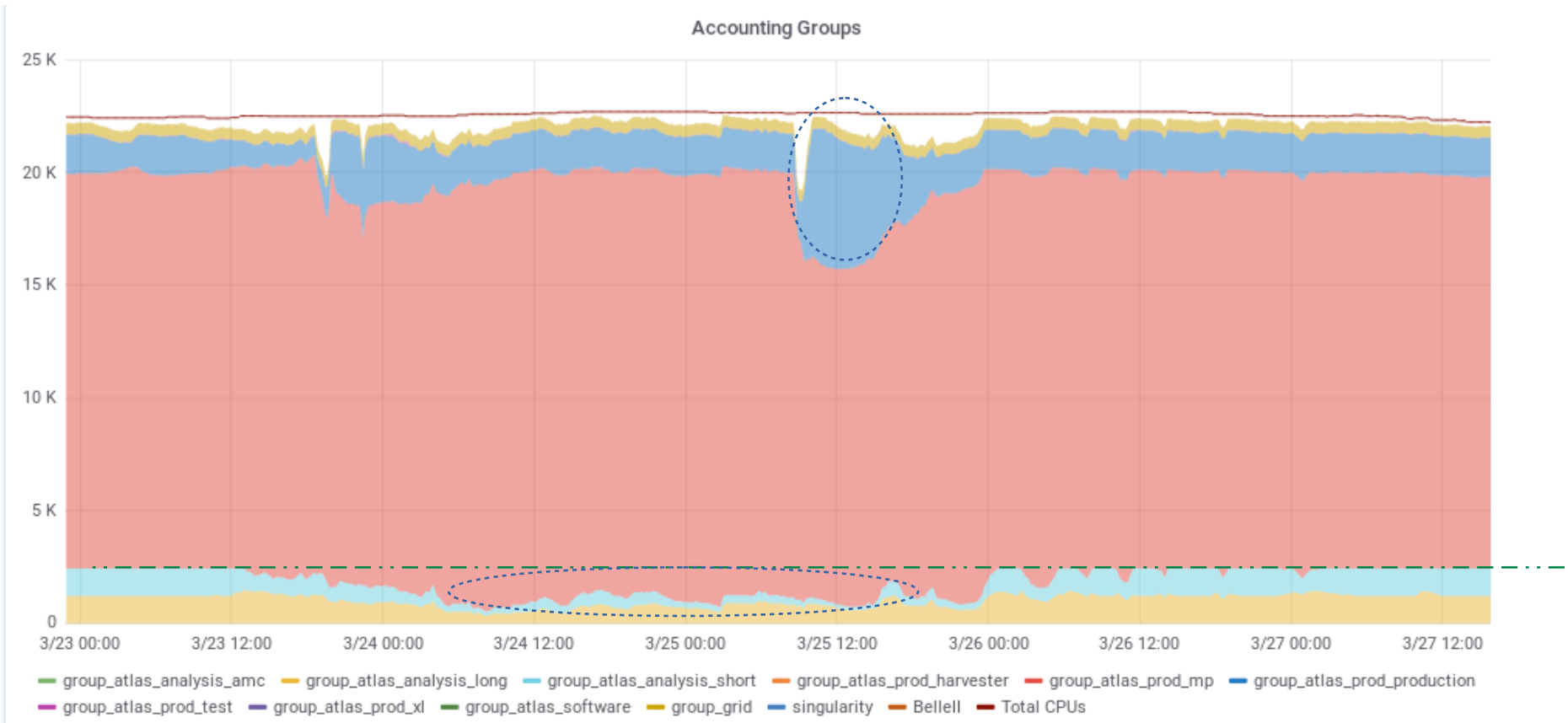
"Foreign" experiments flocked to PHENIX

Group Quota Model

- **Hierarchy of job classes with “quotas” assigned to each class (how many CPUs can they get)**
  - Jobs “spill” between groups freely



Group Surplus in ATLAS



"level" for analysis

Areas where other queues "fill in" with surplus-sharing automatically

How?

- **Will adopt the group quota model**
  - Experiments→Top-Level Groups
  - Quotas→Set by contribution
  - Flocking→Surplus sharing
  - Queues→Sub-Groups in experiment
  - Fair-share→Fair-Share (between users within group)

Why?

- More standard setup—everyone gets same features
- Sub-groups give experiments flexibility to define own policies / “queues”
- Surplus sharing automatically ensures maximum occupancy
- One unified policy (helps manage user expectations during growth)
- Easy scaling of offloading work. E.g. Backfilling HPC (slurm) resources with routed overflow jobs

Implementation Details

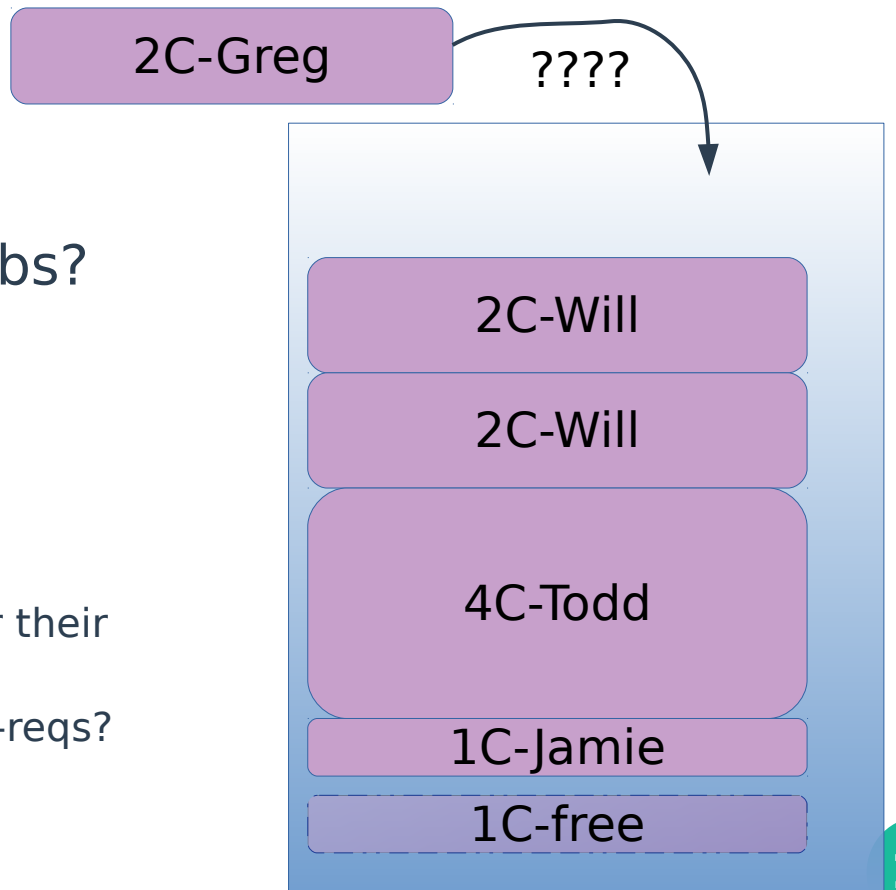
- **Preemption**
  - Needed for two reasons (assuming latency constraints)
    1) Intra-group: most collaborations want to be able to evict a resource hog sooner than the maximum runtime allowed (Latency)
    2) Inter-group: if surplus sharing is on, a group can monopolize the pool, not acceptable for all other colloborations to wait the maximum time to get their own resources back (preempt a group down to its quota)
- Currently has major issues with Partitionable-Slots
  - HTCondor team promises progress here…

Limitations

- **Latency vs. Throughput:**
  - Most fundamental limit—all groups need to agree on allowable job run-time and acceptable latency
    - Currently able to be set per-experiment, but by its nature it is pool-wide
  - Manifests in several places
    - Maximum job lengths, how to fairly allow differences
    - How much to defragment to allow “large” jobs

Implementation Details

- **Partitionable Slots w/ Preemption**
  - How to make room for larger jobs?
  - All slots that meet Preempt-Requirements
    - How to choose what to evict
      - Users with worst integrated priority?
      - Users belonging to groups most over their quota?
      - Least badput respecting preemption-reqs?
    - This is non-trivial and will require experimentation



Preemption

- **Do we need it?**
  - Poll: What are the maximum-allowed and minimum-guaranteed run-times at your site?
  - Poll: What are your user expectations for latency?

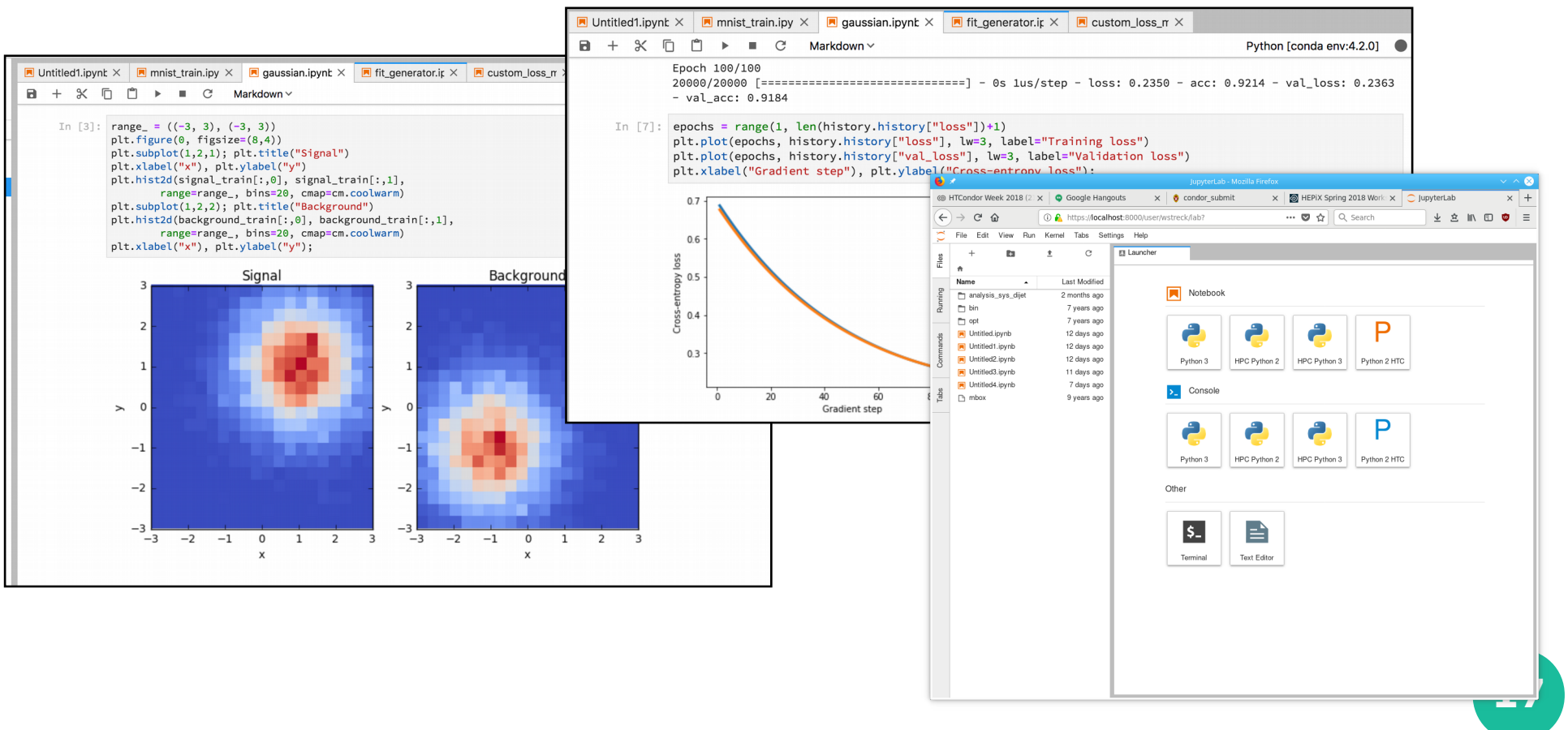
New Computing Paradigms

HTC and Jupyter for Iteractive Scaling

Jupyter

- **Came from IPython project**
  - Ipython→Jupyter (Hub/Lab/whatever)
- **Interactive Python Interperter and Login Shell in Browser**
- **Why not just log in normally?**
  - This is (to most admins inexplicably) a major barrier to science getting done
- **Very useful session management and portability via browser**

Jupyter


Untitled1.ipynb
mnist_train.ipy
gaussian.ipynb
fit_generator.ip
custom_loss_n
Markdown
Python [conda env:4.2.0]
In [3]:
range_ = ((-3, 3), (-3, 3))
plt.figure(0, figsize=(8,4))
plt.subplot(1,2,1); plt.title("Signal")
plt.xlabel("x"), plt.ylabel("y")
plt.hist2d(signal_train[:,0], signal_train[:,1],
range=range_, bins=20, cmap=cm.coolwarm)
plt.subplot(1,2,2); plt.title("Background")
plt.hist2d(background_train[:,0], background_train[:,1],
range=range_, bins=20, cmap=cm.coolwarm)
plt.xlabel("x"), plt.ylabel("y");
Signal
Background
Epoch 100/100
20000/20000 [==========] - 0s 1us/step - loss: 0.2350 - acc: 0.9214 - val_loss: 0.2363
- val_acc: 0.9184
In [7]:
epochs = range(1, len(history.history["loss"])+1)
plt.plot(epochs, history.history["loss"], lw=3, label="Training loss")
plt.plot(epochs, history.history["val_loss"], lw=3, label="Validation loss")
plt.xlabel("Gradient step"), plt.ylabel("Cross-entropy loss");
Cross-entropy loss
Gradient step
JupyterLab - Mozilla Firefox
HTCondor Week 2018
Google Hangouts
condor_submit
HEPiX Spring 2018 Work
JupyterLab
https://localhost:8000/user/wstreck/lab?
File Edit View Run Kernel Tabs Settings Help
Files
Running
Commands
Tabs
Launcher
Name
Last Modified
analysis_sys_dijet 2 months ago
bin 7 years ago
opt 7 years ago
Untitled.ipynb 12 days ago
Untitled1.ipynb 12 days ago
Untitled2.ipynb 12 days ago
Untitled3.ipynb 11 days ago
Untitled4.ipynb 7 days ago
mbox 9 years ago
Notebook
Python 3
HPC Python 2
HPC Python 3
Python 2 HTC
Console
Other
Terminal
Text Editor

BatchSpawner

- **Using Condor BatchSpawner**
  - Jupyter sessions spawned in batch jobs that proxy back to the jupyter node
  - Allows reduction of dedicated interactive nodes
  - Greatly-enhanced scalability of interactive workload
- **This is timely, considering growth of userbase and the shared pool**
  - Raises issues of latency again!
  - I vote for “condor_NOW” tool name, Todd

HTCondor Integration

- **Different Levels**
  1) Batch → Interactive (already discussed)
  2) Programmatic job / workflow management
  3) Leveraging batch horsepower “invisibly”

Job Management

- **Job Driver**

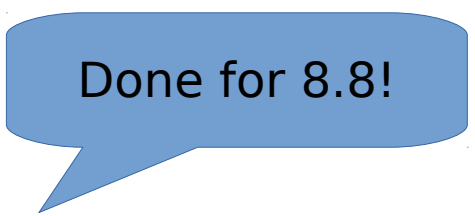
```
[willsk@willsktop condor (master)]$ python
Python 2.7.15 (default, May  9 2018, 11:18:37)
[GCC 7.3.1 20180303 (Red Hat 7.3.1-5)] on linux2
Type "help", "copyright", "credits" or "license" for more informati
on.
>>> import job
>>> j = job.Job({"Executable": "/bin/sleep", "Arguments": '2000'})
>>> print j.submit()
78
>>> j.status
1
>>> j.hold()
[ TotalChangedAds = 1; TotalSuccess = 1; TotalBadStatus = 0; TotalP
ermissionDenied = 0; TotalError = 0; TotalNotFound = 0; TotalJobAds
 = 1; TotalAlreadyDone = 0 ]
>>> j.status
5
>>> j.remove()
[ TotalChangedAds = 1; TotalSuccess = 1; TotalBadStatus = 0; TotalP
ermissionDenied = 0; TotalError = 0; TotalNotFound = 0; TotalJobAds
 = 1; TotalAlreadyDone = 0 ]
>>> j._jobdata
>>>
```

```
[willsk@willsktop condor (master)]$ cat tests/jdfs/sleep.job
Executable      = /bin/sleep
Arguments       = 10
Log             = sleep.$(cluster).log

Queue 4
[willsk@willsktop condor (master)]$ python
Python 2.7.15 (default, May  9 2018, 11:18:37)
[GCC 7.3.1 20180303 (Red Hat 7.3.1-5)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import job
>>> c = job.JobCluster.from_jdf('tests/jdfs/sleep.job')
>>> c.status
{0L: 1L, 1L: 1L, 2L: 1L, 3L: 1L}
>>> c.status
{0L: 2L, 1L: 2L, 2L: 2L, 3L: 2L}
>>> c.wait()
```

Job Management

- **Python bindings somewhat un-Pythonic/lacking**
  - UserLog watching broken
  - ~~Batch submit not in same cluster~~
  - ~~Support advanced “Queue” iterables~~
    - Which “level”, in language or in library?



Job Management

- **DAGs would be great!**
  - Ipython Parallel project API?
- **Which interfaces most useful to users?**
  - Which can generalize to other batch systems or HPC?

“Higher” Level Interface

- **Make “interactive” and “batch” indistinguishable**
  - Run code over one input locally, run over 10000 in batch, from within Jupyter
  - Condor-leveraged map_async(<fn>, <iter>)
    - Made test case, use cloudpickle to serialize code + data, distribute naively 1:1 to condor jobs

Conclusions

- **Consolidating many pools**
  - Must enforce common policy
    - Need p-Slot preemption to sell to users
  - Better scaling to many user groups
- **Users love Jupyter**
  - Trying to think how HTCondor can be useful in this use case—happy to talk to users and share code

Questions?
Comments?

Thank you!

GPU Hackathon @ BNL
This September
See this site for info