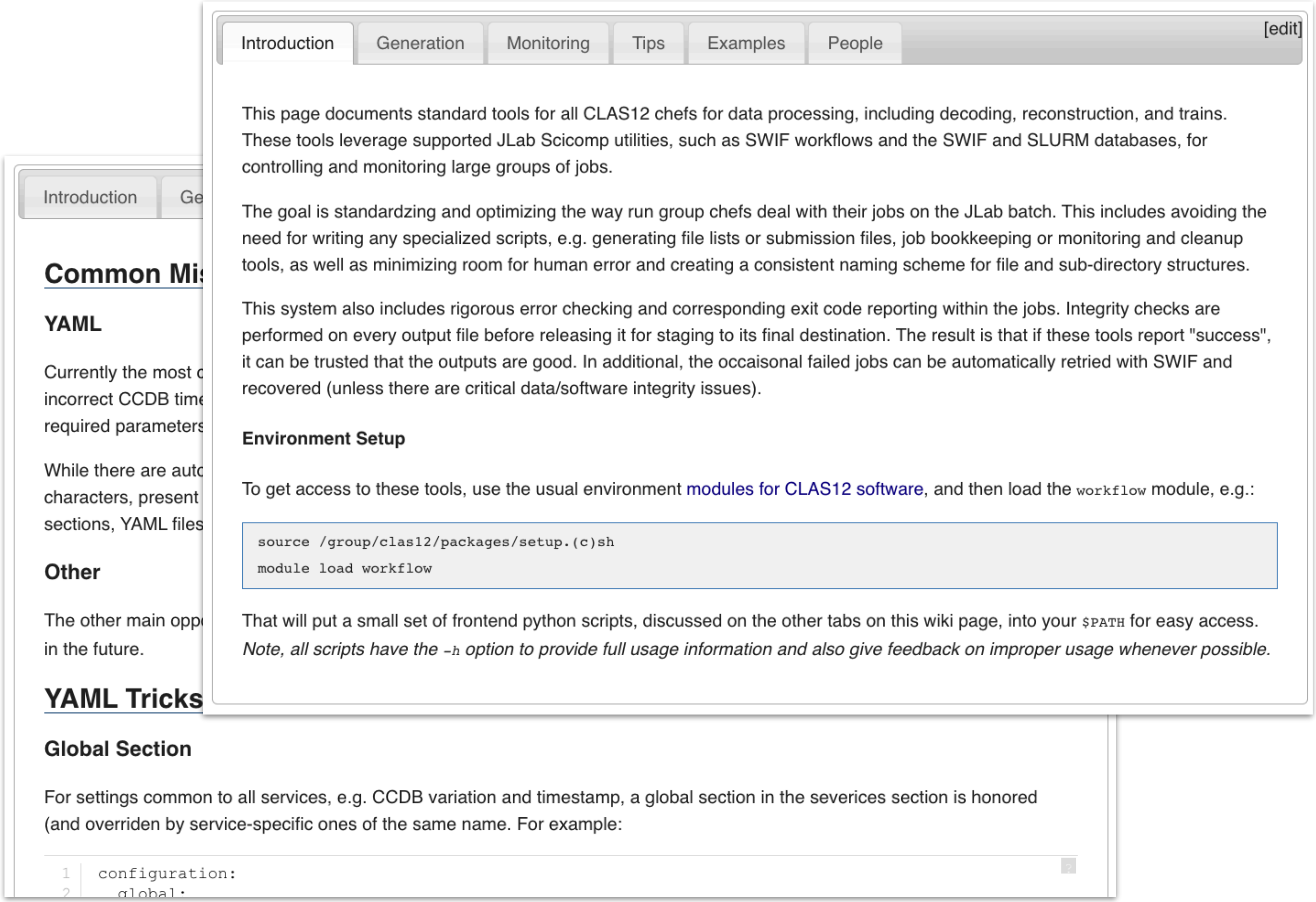
Computing Update

N. Baltzell - CLAS Collaboration Meeting - March 2, 2021

CLAS12 Chef Documentation

- Weekly meetings with run group chefs and software/computing coordinators
- Documentation cleaned up, expanded, more guidelines and examples, FAQs, common issues and concerns
- Linked from the reconstruction tab of the software wiki
- Useful for new chefs and future run groups coming online in the collaboration, soliciting feedback on what's missing



This page documents standard tools for all CLAS12 chefs for data processing, including decoding, reconstruction, and trains. These tools leverage supported JLab Scicomp utilities, such as SWIF workflows and the SWIF and SLURM databases, for controlling and monitoring large groups of jobs.

The goal is standardzing and optimizing the way run group chefs deal with their jobs on the JLab batch. This includes avoiding the need for writing any specialized scripts, e.g. generating file lists or submission files, job bookkeeping or monitoring and cleanup tools, as well as minimizing room for human error and creating a consistent naming scheme for file and sub-directory structures.

This system also includes rigorous error checking and corresponding exit code reporting within the jobs. Integrity checks are performed on every output file before releasing it for staging to its final destination. The result is that if these tools report "success", it can be trusted that the outputs are good. In additional, the occaisonal failed jobs can be automatically retried with SWIF and recovered (unless there are critical data/software integrity issues).

Environment Setup

To get access to these tools, use the usual environment modules for CLAS12 software, and then load the `workflow` module, e.g.:

```
source /group/clas12/packages/setup.(c)sh
module load workflow
```

That will put a small set of frontend python scripts, discussed on the other tabs on this wiki page, into your `$PATH` for easy access. *Note, all scripts have the `-h` option to provide full usage information and also give feedback on improper usage whenever possible.*

Common Mis

YAML

Currently the most c
incorrect CCDB time
required parameters

While there are auto
characters, present
sections, YAML files

Other

The other main opp
in the future.

YAML Tricks

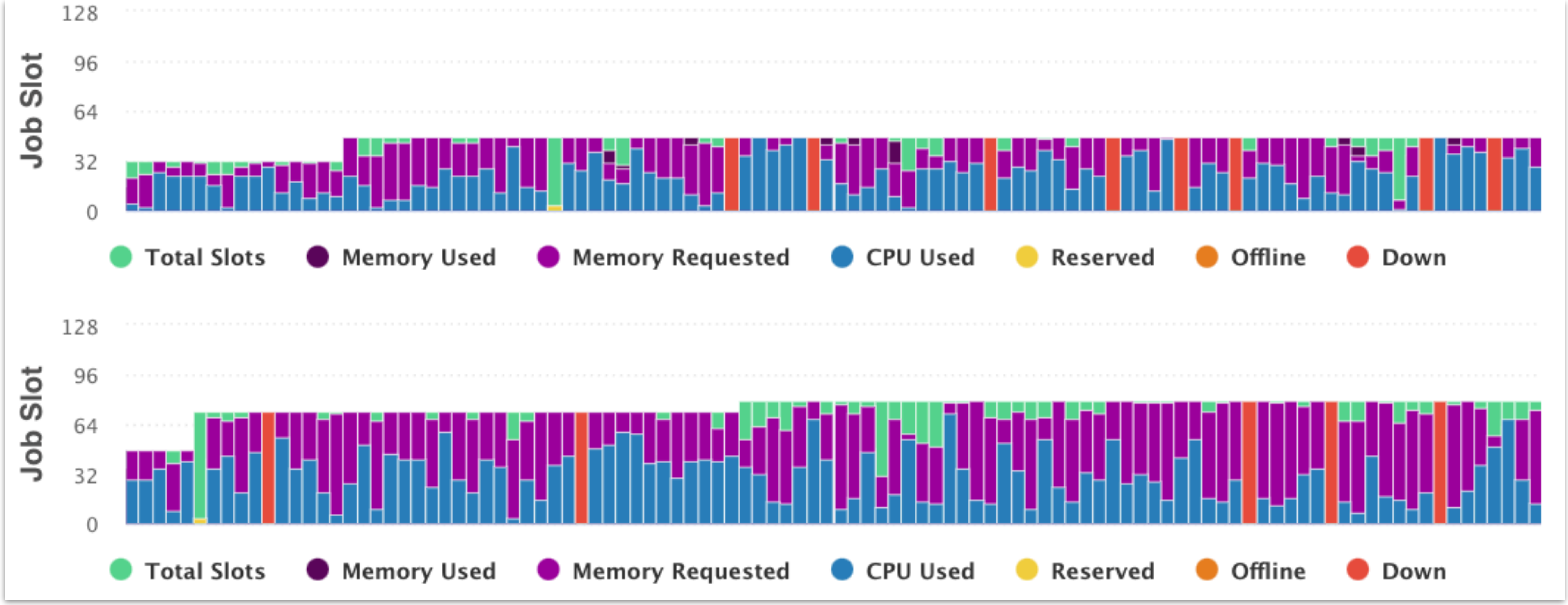
Global Section

For settings common to all services, e.g. CCDB variation and timestamp, a global section in the severices section is honored (and overriden by service-specific ones of the same name. For example:

```
configuration:
  global:
```

Batch Farm - Memory Requests

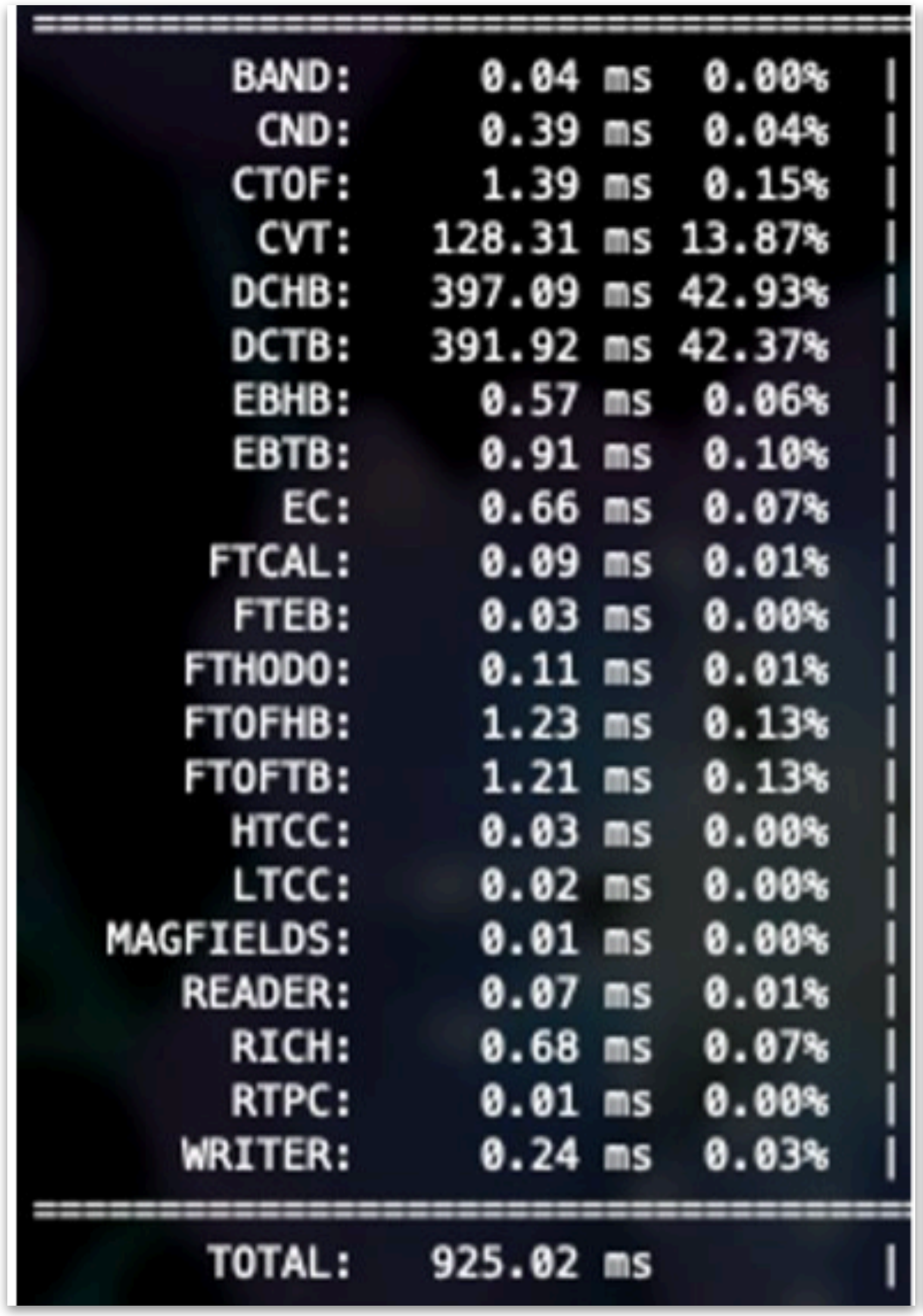
- Over the past year, we've gotten much better regarding CLAS/CLAS12 user jobs, but still occasionally some outliers.
- The plot below was yesterday and abnormal. It shows a 50% idle farm, even though plenty of jobs in the queue, purely due to unnecessarily-bloated memory requests!
- Before running many jobs on the batch farm, make sure your requests are appropriate for the jobs in question. Run some test jobs, check memory usage at scicomp.jlab.org, and set requests accordingly.



Future COATJAVA Releases - Java 11

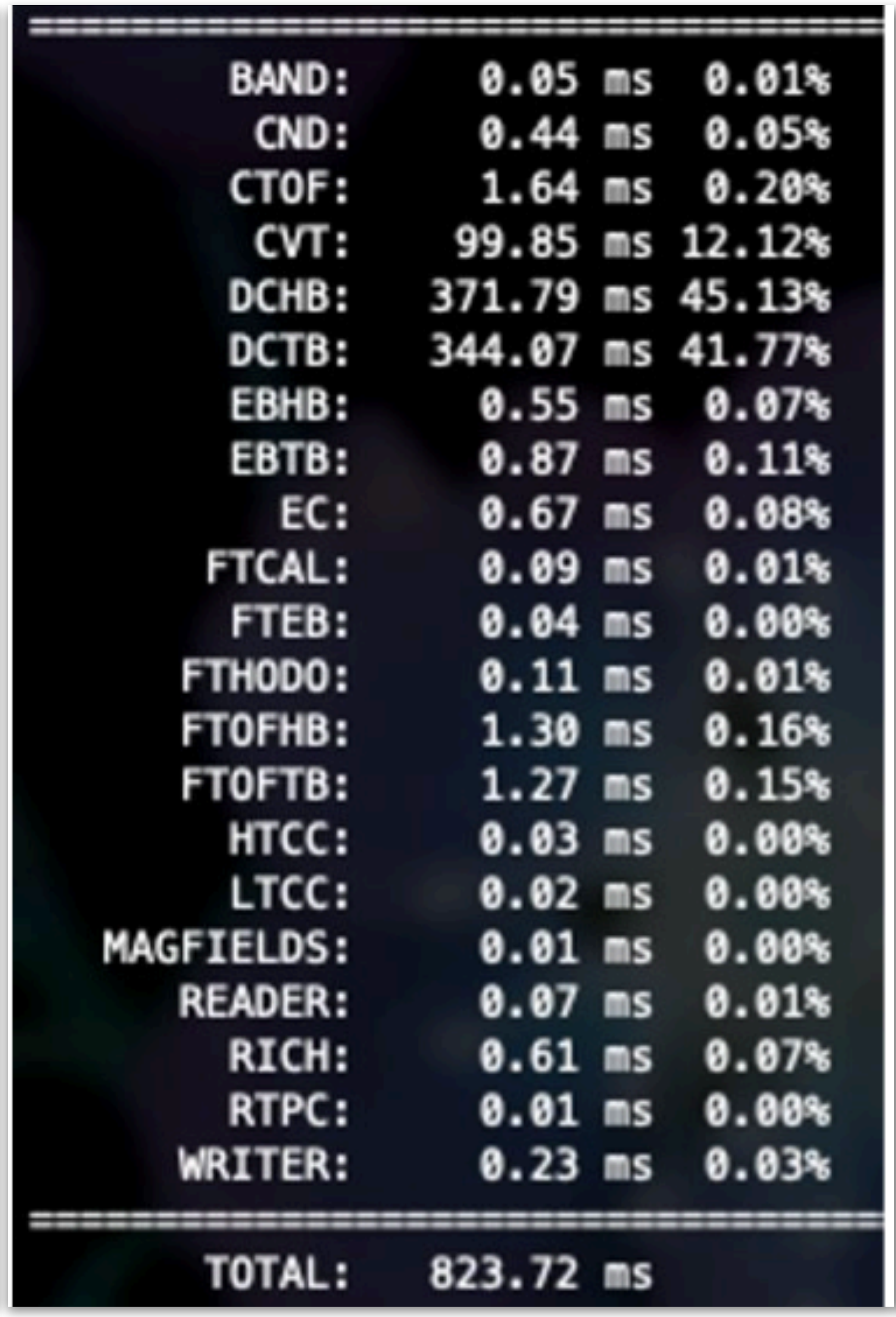
- 10-20% reconstruction speedup by compiling and running in Java 11
  - wasn’t effective in multi-threaded jobs until CLARA also compiled in 11, recently
- Dependencies also starting to get benefits from going to 11
  - e.g. recent GROOT improvements
  - In order to accommodate that, production code will not be compilable in 8 anymore (without manually reverting to older dependency versions, but we can keep a branch available for that if necessary)
- Java 11 is standard on all/most modern operating systems and is the current LTS release
  - and available in our CUE environment modules
- It’s (past) time to make the switch; **expect future releases to not support Java 8 at compile nor runtime …**

8



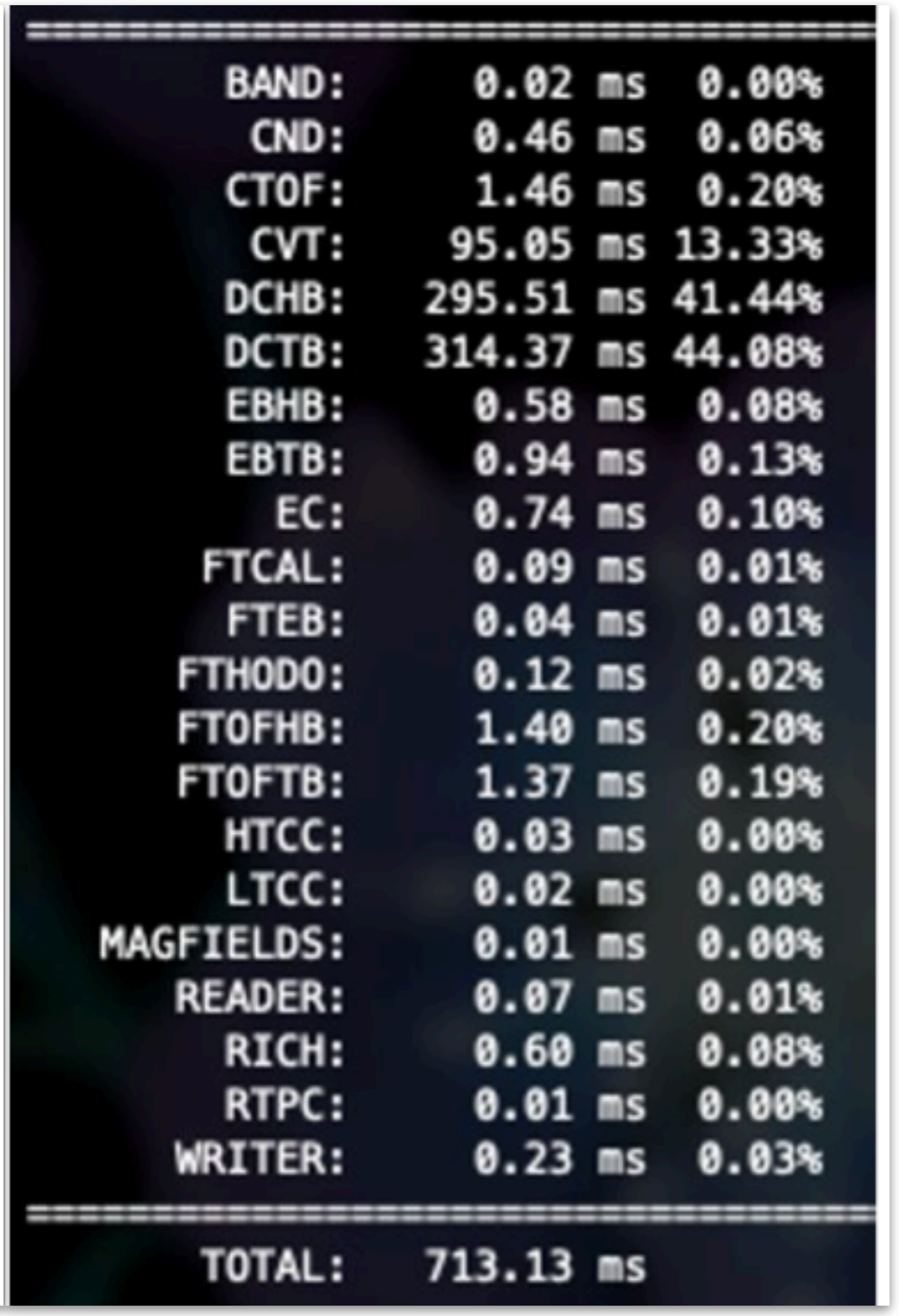
```
=================================================
       BAND:      0.04 ms   0.00%  |
        CND:      0.39 ms   0.04%  |
       CTOF:      1.39 ms   0.15%  |
        CVT:    128.31 ms  13.87%  |
       DCHB:    397.09 ms  42.93%  |
       DCTB:    391.92 ms  42.37%  |
       EBHB:      0.57 ms   0.06%  |
       EBTB:      0.91 ms   0.10%  |
         EC:      0.66 ms   0.07%  |
      FTCAL:      0.09 ms   0.01%  |
       FTEB:      0.03 ms   0.00%  |
     FTHODO:      0.11 ms   0.01%  |
     FTOFHB:      1.23 ms   0.13%  |
     FTOFTB:      1.21 ms   0.13%  |
       HTCC:      0.03 ms   0.00%  |
       LTCC:      0.02 ms   0.00%  |
  MAGFIELDS:      0.01 ms   0.00%  |
     READER:      0.07 ms   0.01%  |
       RICH:      0.68 ms   0.07%  |
       RTPC:      0.01 ms   0.00%  |
     WRITER:      0.24 ms   0.03%  |
=================================================
      TOTAL:    925.02 ms          |
```

11



```
=================================================
       BAND:      0.05 ms   0.01%
        CND:      0.44 ms   0.05%
       CTOF:      1.64 ms   0.20%
        CVT:     99.85 ms  12.12%
       DCHB:    371.79 ms  45.13%
       DCTB:    344.07 ms  41.77%
       EBHB:      0.55 ms   0.07%
       EBTB:      0.87 ms   0.11%
         EC:      0.67 ms   0.08%
      FTCAL:      0.09 ms   0.01%
       FTEB:      0.04 ms   0.00%
     FTHODO:      0.11 ms   0.01%
     FTOFHB:      1.30 ms   0.16%
     FTOFTB:      1.27 ms   0.15%
       HTCC:      0.03 ms   0.00%
       LTCC:      0.02 ms   0.00%
  MAGFIELDS:      0.01 ms   0.00%
     READER:      0.07 ms   0.01%
       RICH:      0.61 ms   0.07%
       RTPC:      0.01 ms   0.00%
     WRITER:      0.23 ms   0.03%
=================================================
      TOTAL:    823.72 ms
```

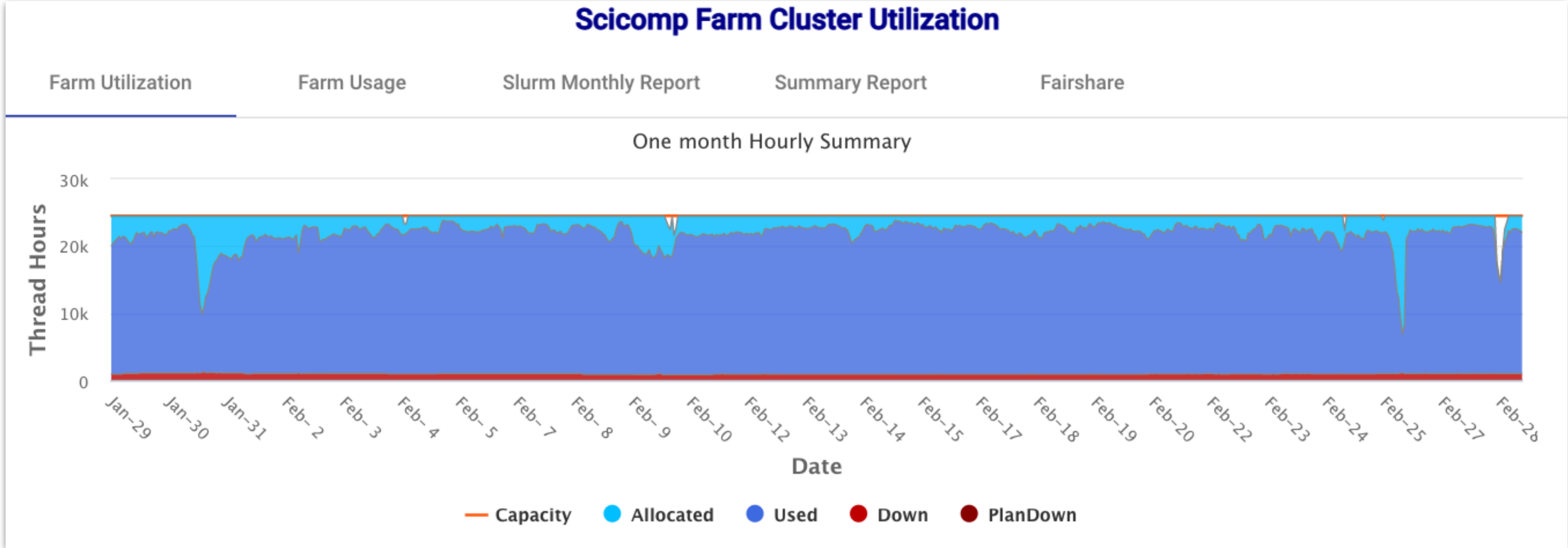
11+opts



```
=================================================
       BAND:      0.02 ms   0.00%
        CND:      0.46 ms   0.06%
       CTOF:      1.46 ms   0.20%
        CVT:     95.05 ms  13.33%
       DCHB:    295.51 ms  41.44%
       DCTB:    314.37 ms  44.08%
       EBHB:      0.58 ms   0.08%
       EBTB:      0.94 ms   0.13%
         EC:      0.74 ms   0.10%
      FTCAL:      0.09 ms   0.01%
       FTEB:      0.04 ms   0.01%
     FTHODO:      0.12 ms   0.02%
     FTOFHB:      1.40 ms   0.20%
     FTOFTB:      1.37 ms   0.19%
       HTCC:      0.03 ms   0.00%
       LTCC:      0.02 ms   0.00%
  MAGFIELDS:      0.01 ms   0.00%
     READER:      0.07 ms   0.01%
       RICH:      0.60 ms   0.08%
       RTPC:      0.01 ms   0.00%
     WRITER:      0.23 ms   0.03%
=================================================
      TOTAL:    713.13 ms
```

There’s also potential speedup from 14 in garbage collection, which we have a lot of, and we may test and run production chef jobs in that soon. But since 14’s not a LTS, and less easily available in many distributions, standard releases will not be migrated yet.

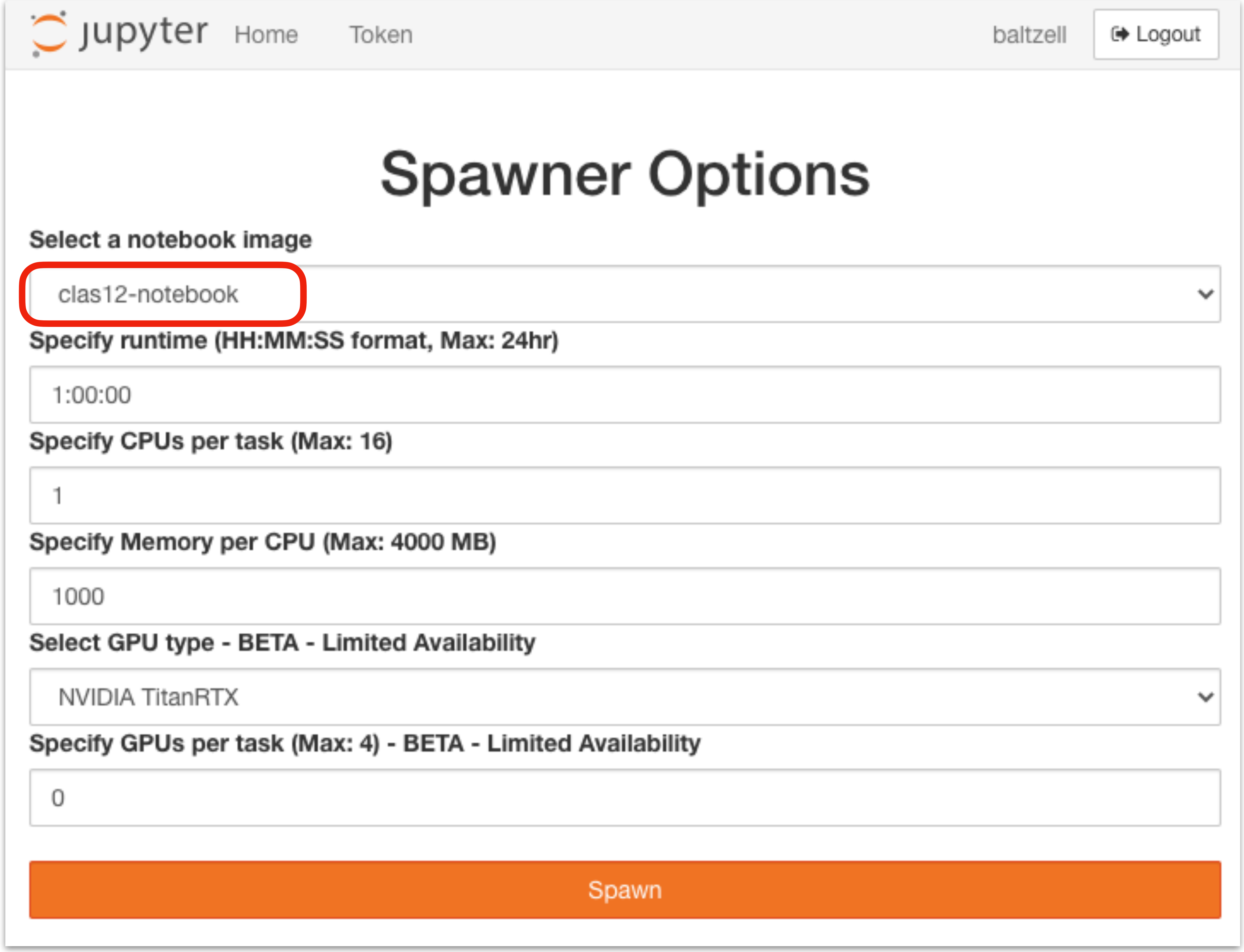
Scicomp (1)



- New, more I/O-performant /work fileserver in procurement stages
    - should be available later this year
- New scicomp-2020.jlab.org website developed and available
    - Much faster load times, more user-friendly, more batch farm bookkeeping, e.g. quarterly, yearly
    - Does not provide support for Auger jobs, and Auger will eventually be removed and replaced by SWIF
        - *If you're still submitting jobs with Auger, you should consider updating to SLURM, or SWIF if you need file staging*
- swif2 is available for testing
    - see /site/bin/swif*, overhauled for offsite and grid jobs
    - user interface is very similar to the original swif (now called swif1)
    - see scicomp's website for documentation

Scicomp (2)

- New jupyterhub.jlab.org notebooks requested by CLAS collaboration members and in development
  - one with python kernel and pyROOT
  - another with C++/ROOT kernel, and clas12root support coming soon
- Full access to JLab filesystems
- Must be in a SLURM account for access, which is already true if you've run a batch job before
  - otherwise we submit a request at cc.jlab.org



Spawner Options

Select a notebook image: clas12-notebook

Specify runtime (HH:MM:SS format, Max: 24hr): 1:00:00

Specify CPUs per task (Max: 16): 1

Specify Memory per CPU (Max: 4000 MB): 1000

Select GPU type - BETA - Limited Availability: NVIDIA TitanRTX

Specify GPUs per task (Max: 4) - BETA - Limited Availability: 0

Summary

- CLAS12 chef documentation improved and expanded, soliciting feedback from chefs
- Batch farm job memory requests still important (and not always good, sometimes severely!)
- Moving COATJAVA and some dependencies (at least GROOT) to Java 11 for future releases
  - with a major version bump
- Scicomp
  - New and improved scicomp website
    - Doesn’t support Auger, which will be getting phased out eventually, move your job submissions to SLURM/SWIF
  - New notebooks at jupyterhub.jlab.org for CLAS in progress
  - New /work fileserver in procurement, anticipated ready for use this year