HPC^3 Policy

Executive Summary

Some background

- HPC and GreenPlanet catalyzed shared computing at UCI
- HPC survey indicated importance to faculty for research, overall utility *and room for improvement*
- Scalability of HPC has reached limitations in that every owner is given a different queue
    - 70+ queues (submission points) are time consuming to manage. Confusing to use
    - Some queues have long waits, others are sporadically used (missing opportunity to share unused resource and reduce wait times)
- Recent MRI award coupled with UCI Campus investment provides an opportunity to adjust shared computing and improve upon the existing cluster

HPC^3 - Goals

- Enables users to have **access to a larger compute/analysis system** than they could reasonably afford “on their own”
- Enables **access to specialized nodes** (e.g. Large memory, GPU (64-bit), deep-learning (32-bit), ..)
- **Fosters a growing community** across UCI to utilize scalable computing (HPC and HTC)* for their scientific research program and teaching
- **Provides a well-managed software environment** that forms the basis of a “reproducible” scientific instrument
- **Fits “seamlessly”** into the progression of : desktop, lab cluster, campus, national (e.g. XSEDE) and commercial cloud
- Enables **construction of more secure research environments**

*HPC – High-Performance Computing,   HTC – High-Throughput Computing

Facilitating sharing of resources

- Fundamental issue – How can we enable three different models of acquiring computing cycles to work together on the same platform
    1. Condo-style – researchers purchase hardware, RCIC manages
    2. Granted – Fulfill aspirational goal of 200K core hours/year to any UCI researcher who requests (RCI Vision document)
    3. Cycle Purchase – enable researchers to buy cycles in a manner similar to commercial cloud

➢ Approach: Use Core-hour accounting + “free” (non-accounted) cycles

Each job draws from a core-hours accounting bank

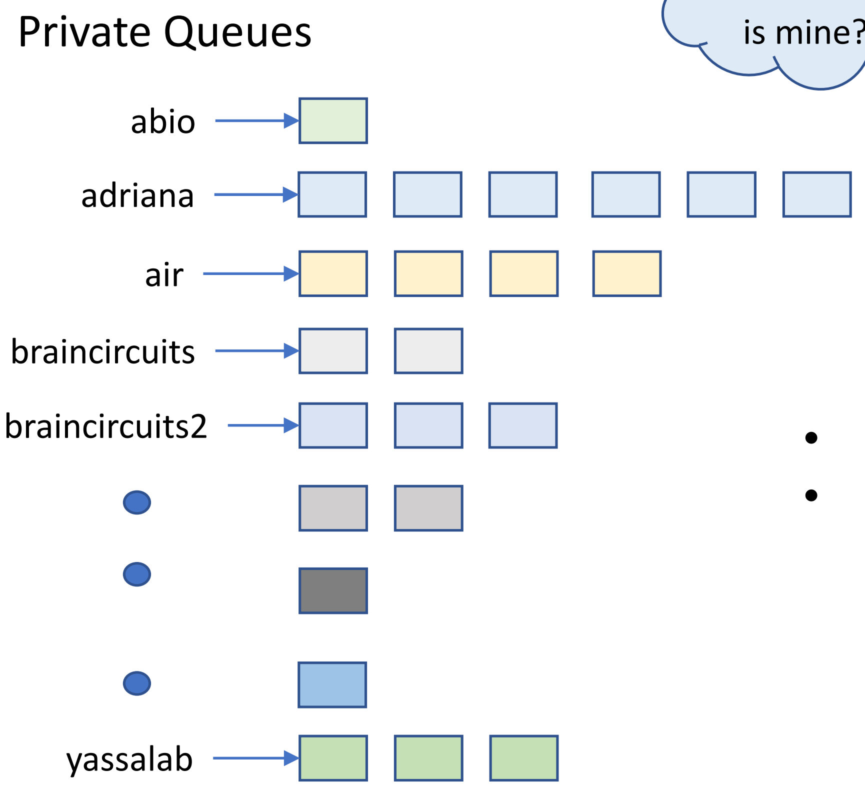
- Accounted jobs vs. “free” cycles
  - Accounted – Once a job is started, it cannot be killed or pre-empted
  - Free – a free (non-accounted) can be killed at anytime
- Three ways of filling your account*
  - Granted cycles. (UCI core funds purchase hardware to provide enough resource to support granted cycles)
  - Converted. Condo nodes capability converted to core-hours
    - Physical hardware can deliver N-core-hours/year. 0.95N are deposited into an owners account each year he/she has a node (or nodes) in the cluster.
  - Purchased. Hours are pre-purchased (nominally $.0125/core-hour) in reasonable chunks (e.g., $100 increments -> ~ 8000 core-hours)

* Account is a flexible notion. It can be per-user, per-lab or per-group. Multiple users can be authorized to use the same bank

Queueing in HPC vs. HPC3

HPC: Which hardware is mine?

HPC^3 : What kind of resource does my job need?



Private Queues

abio

adriana

air

braincircuits

braincircuits2

●

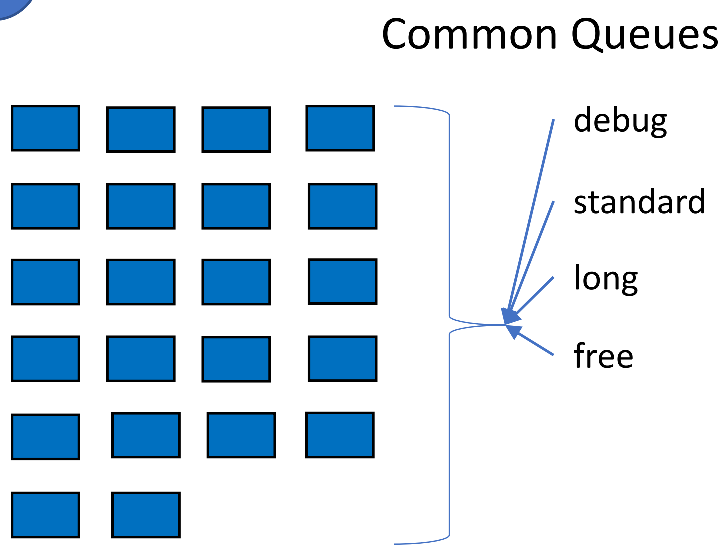
●

●

yassalab

HPC

- 70+ queues vs. ~5 queues
- Owner preemption vs. free queue preemption



Common Queues

debug

standard

long

free

HPC^3

Major Differences Between HPC and HPC3

HPC	HPC3
Node owners can kill jobs on their nodes	Only “free” jobs can be killed
Most users have access to a small number of queues	Users have access to nearly all queues
Only “free” jobs can span owner and non-owner jobs	All jobs can span nodes as needed
	Users can be granted core-hours
	Users can purchase core-hours
Users can purchase hardware	Users can purchase hardware, but HPC3 steering group defines “supported hardware configs”

Fair Queueing + No Oversubscription

- No oversubscription
  - If you own X% of the total cluster, your starting account balance is ~ X% of the total number hours that can be delivered in a year by the entire cluster.
- Fair Queuing
  - Non-FIFO. Jobs arriving earlier in the queue are not guaranteed to schedule first.
    - Want to prevent a large number of jobs from User A blocking a small number of jobs from user Z
- Bias towards “interactive” turnaround for small debugging jobs
  - Optimize people time for the “debug” process. Small core count + short time duration jobs should schedule as quickly as possible
- Fair running
  - If you are running an accounted job, once your job is started, it will not be pre-empted/killed
- Free cycles
  - Users who “pick up spare cycles” can have their jobs killed so that accounted jobs can run as soon as possible

Some detailed but important policies

1. Hardware sunset – HPC3 will define an upfront policy of when nodes are taken out of the cluster/no-longer count for core-hours
    - Likely: Hardware Warranty + 1 year
2. Cores are “cores + memory”. Jobs requiring more memory will need to request more cores to fulfil needs
3. GPUs – accounted for separately. Policy still being worked out.
4. Overlapping operation of HPC and HPC3
    - Goal : overlap does not last for more than a year. More discussion is planned to ensure any impacts are addressed.
5. Converting nodes from HPC to HPC3
    - Supported and encouraged.
    - Only “compatible” (e.g. connected to network, able to run CentOS7,…)
    - Treated as condo nodes for computing core-hours.
    - Will sunset per standard HPC3 policy.
    - No additional $$ cost to owner to make this transfer.

Flexibility in “Who runs when” Policy

- Will put in place a “next-to-run” queue that costs about 2X/core-hour.
  - Provides a mechanism for users to elevate their priority to meet paper/grant/other deadlines
  - “kill” free jobs to make room for next-to-run job.
  - Jumps ahead of all other standard jobs, but will NOT kill any accounted job
  - Competes fairly with all other next-to-run jobs when there are conflicting requests
- Admin intervention
  - In rare cases, admins can elevate priority of jobs to meet grant/paper deadlines
- “Fair” queueing isn’t ever perfect (no priority system is perfect!), specific details of policy can be adjusted over time so that HPC3 works better for UCI researchers.

The key “shortcoming” of core-hour accounting

- An unused core is “lost forever” computing
- ➔ If banks are not spent down on a regular basis, It is possible that the sum of all remaining “funds” are more than the cluster can physically deliver
  - This is the case where the cluster is not over-allocated but is *under-utilized*.
- HPC^3 may need a “use it or lose it” policy on hours
  - Will compute an “automated debit” to deduct hours ***only if the total utilization*** is less than a pre-determined threshold (~80%).
  - For several months, will only perform the computation, no actual debiting will occur.
  - Utilization is defined as “allocated hours”, not how efficiently different codes run.
  - Want to support common, cyclic usage mode of “perform a large amount of computing, then do no computing while analyzing results”

FAQ

1. Who defined the policy?
   - The HPC3 subcommittee of the RCIC advisory committee crafted the initial policy. The RCIC Advisory committee approved the policy
2. Does this sharing cause problems with granting agencies?
   - We don’t believe so. The “condo” conversion factor (0.95) essentially enables an owner to turn around and spend their converted hours on their owned hardware. The 5% reductions is a rational estimate of lack of availability of hardware when accounting for software maintenance, reboots, and other downtime. Grants should purchase the hardware capacity they require. Not more.
3. How do I prevent my grad student from “draining my account” before I know about it?
   - RCIC will allow you to set up “charge limits” for any particular user. If a student hits their limit, they ask you for more, or use the free queue.
4. I don’t have any funds to purchase cycles or buy hardware, can I use HPC3?
   - Yes, if you are faculty member, you have granted cycles that are yours to use anyway you see fit for research. There is also the “free” queue, where jobs are not charged
5. If I purchase core-hours, is overhead charged?
   - We are actively working with UCI financial office to see if we can follow UCSD’s lead on making these charges non-overhead bearing.