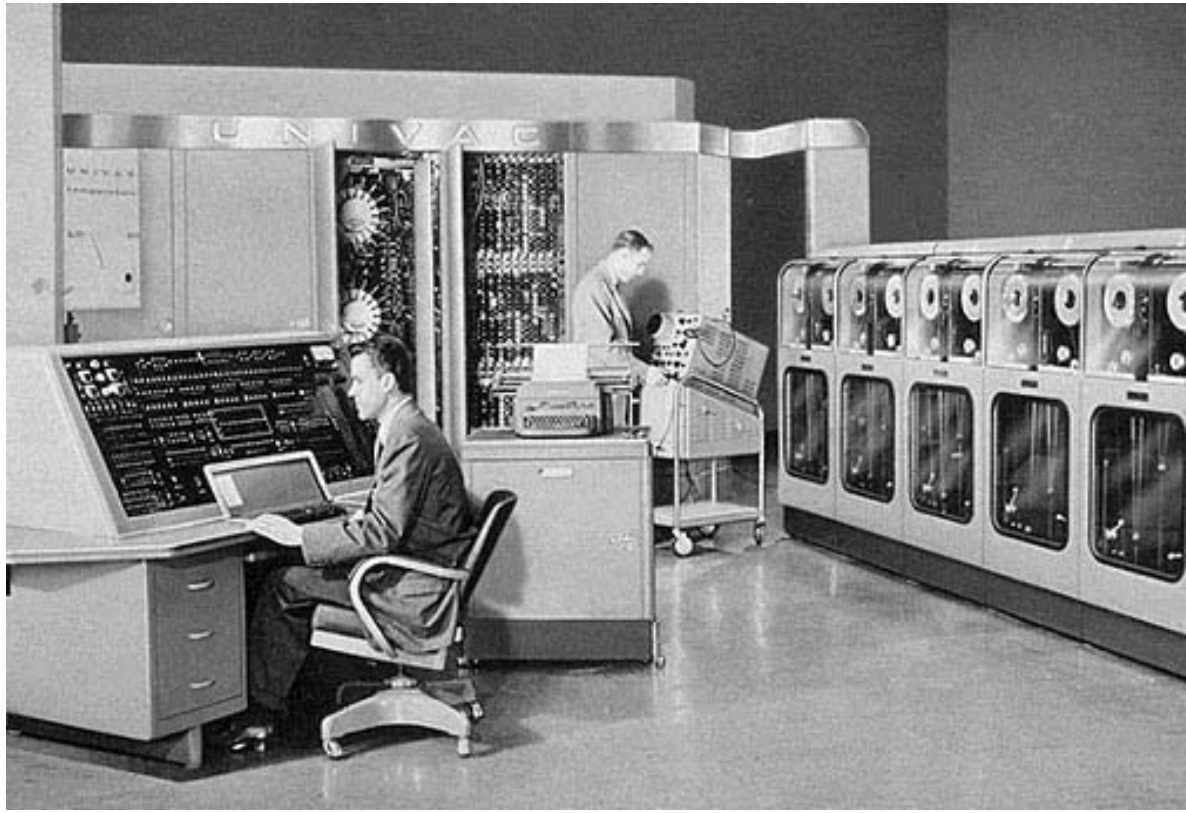
User Perspective and Requirements

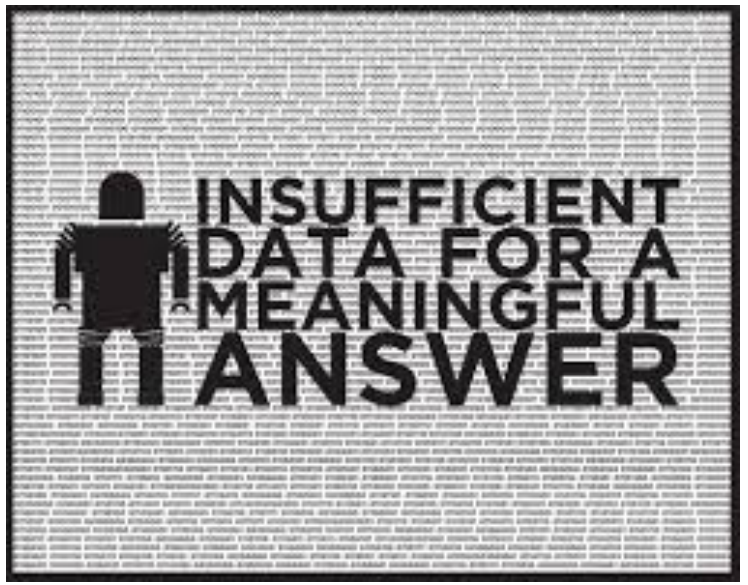


Graham Heyes

Wednesday Sept 30th 2020

What is this talk about?

- Data center managers and data center users have different perspectives.
- The aim of the talk is to start discussion:
  - What matters to the user?
  - What matters to the data center manager?
  - How are they reconciled? (or are they?)
  - How are the user's needs met?
    - Management
    - Resources
    - Operation
- This is a workshop on trends in nuclear physics computing so I will confine the talk to scientific computing users.



If you don't tell me how to design a data center, I won't tell you how to do science.

What matters to a user?

- Having not been a user myself for a while I found a few tame ones and asked them for feedback.

- Amazingly the feedback was pretty consistent.

- The user is here to do science, the things that matter most include:
  - Access to resources.
  - Knowing how to use them.
  - Knowing where and how to get help.
  - Getting the work done.

- The following slides cover these in more detail:

Access to resources

- You would think this was the easy part:
  - Doesn't everyone know what they want and it's readily available exactly when they need it?

- Having the needed resources on hand
  - A user doesn't necessarily know what they want.
  - A user doesn't necessarily know what is available.
  - Resources are finite and shared in a way that is hard to understand.

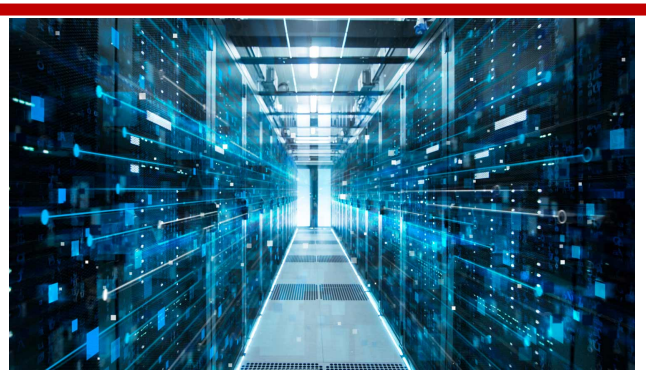
- Having resources assigned.
  - Having identified that a resource is needed a user has to gain access.
  - This is often a source of frustration.

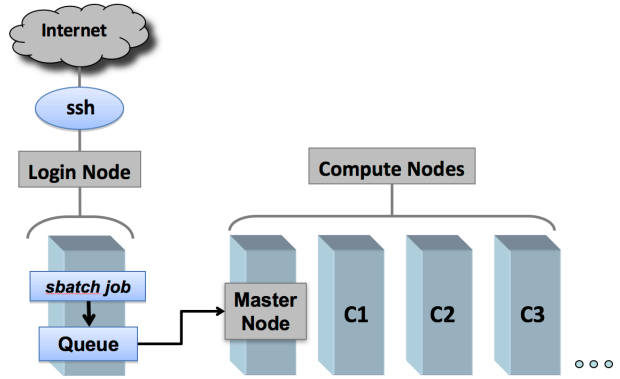
- Priority.
  - To a user their own work and time are the most important things.
  - There are always bigger fish in the sea.
  - Fixed priority is administratively easy but not the best tactic.

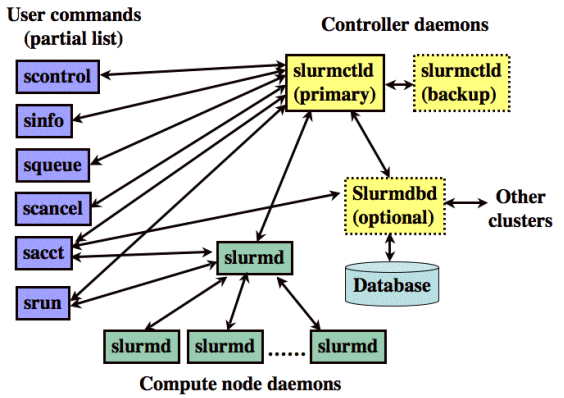
- The key to these frustrations is to plan ahead (see later)

Knowing how to use them

- The sort of computing environment available at a lab is complex and frequently bewildering.
  - How to write code that efficiently uses resources is an art.
  - With a complex interplay between software and hardware it is difficult to chose configuration parameters that make sense.
  - Software is provided by a software group, hardware, batch system and tools by the data center and the user has no idea how any of it works.
  - Many systems, software and hardware, require too much user education.
- Examples things users do because of this :
  - Requesting more resources than needed - unnecessarily queued for a long time.
  - Showing up after a long absence and expecting nothing to have changed.
  - Repeatedly running jobs that skim large numbers of tapes.
  - Storing important and irreproducible data on a volume called /volatile …
- Sometimes it boils down to knowing the right question to ask. Which leads to the next slide.







Jefferson Lab

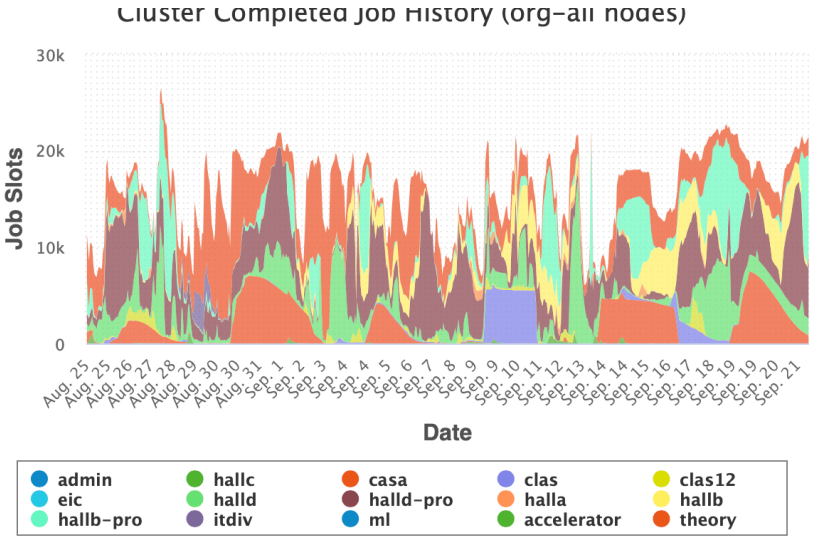
Knowing where and how to get help.

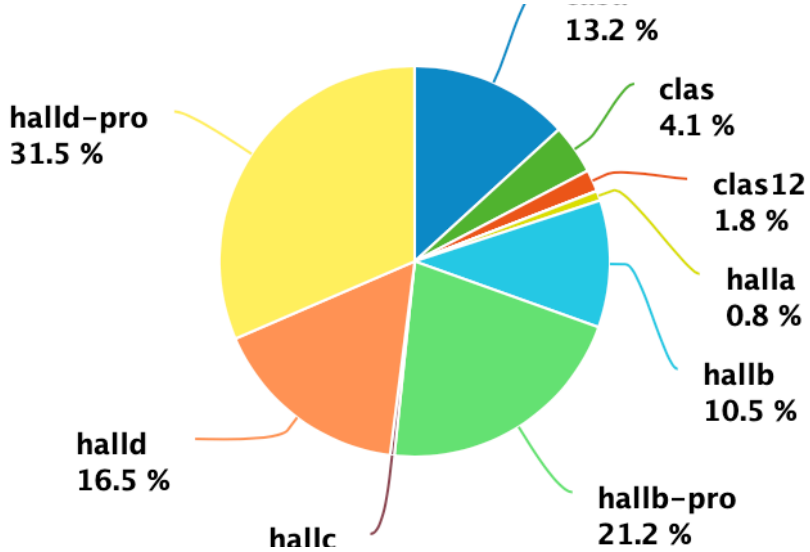
- At a lab there are many experts. Knowing who they are and how to contact them is a problem users frequently face.
  - This leads to reliance on “folk lore” and “here’s what worked for me”.
- Providing clear instruction on how to get help, and encouraging it, is important.
  - People often leave asking for help until all other avenues are exhausted.
- Responsiveness is important.
  - Leaving a user feeling that nobody cares and that they must work it out themselves can lead to more problems.
  - Perception is important – letting people know help is there is as important as providing it.
- Documentation (or lack thereof)
  - A constant struggle is to provide adequate documentation in an accessible way.
  - Much documentation and institutional knowledge is now provided in an electronic format – it is often hard to find.
    - A searchable knowledge base is very important.
  - Out of date information is trouble for both user and service provider.

Jefferson Lab

Getting the work done

- Fairness (real or perceived).
  - One of the most frequent complaints is:
    ”I am not getting my fair share”.
  - It is important that users can clearly see why they get the performance that they do out of the system.
- Availability/uptime - track and make visible outages.
  - Users believe outages are more common than they are.
- Organization and tracking of jobs.
  - Take some of the hard work out of repetitive tasks.
- Diagnostics for job failure modes.
  - If a job fails, make sure the user can find out why.
- Keep systems simple and easy to use.
  - It is hard to efficiently use a system when you don’t know how it works.

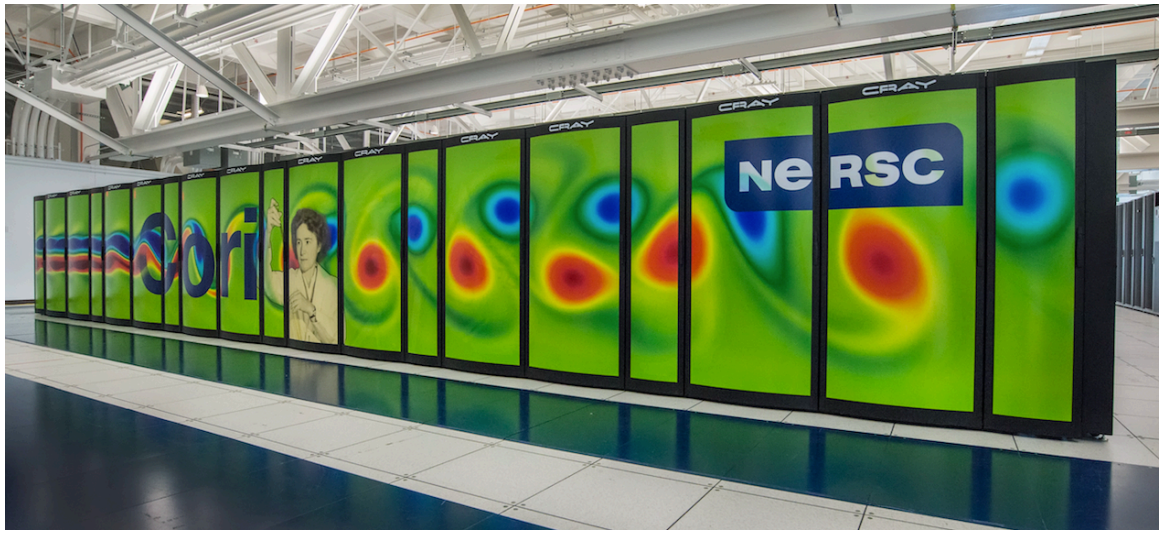


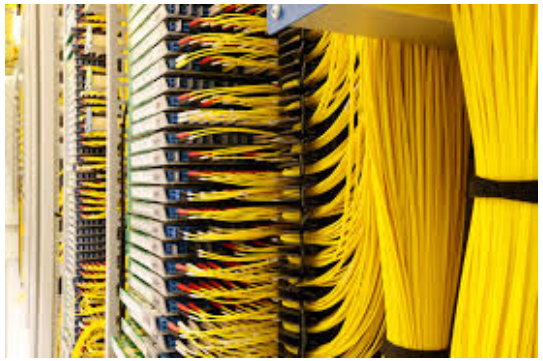


Jefferson Lab

Part 2 – The view from the data center

- The goal of the data center is to support the science mission.
- Tangible things.
  - Infrastructure and operation
  - Compute resources
  - Storage
  - Networking
- Intangible things
  - Computing environment
    - Batch systems
    - Software frameworks and libraries
    - Data management tools
    - Data movement tools
  - User support
    - Application integration
    - Interaction with users



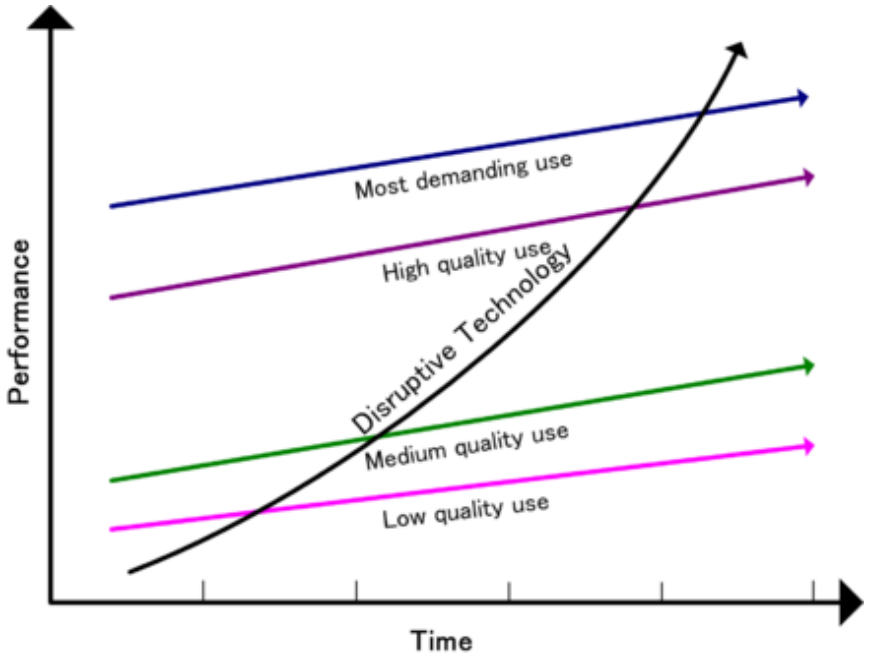


Challenges

- Running a data center to support a user community presents some challenges:
  - What do the users need?
    - Many user communities are notoriously bad at estimating computing needs.
  - When do they need it?
    - Timing of computing needs is driven by external factors:
      - Experiment schedule which is subject to change.
      - Deadlines such as conferences, publication, etc.
  - How is it going to be funded?
    - Many users seem to forget that someone has to pay for computing resources.
  - The landscape is constantly changing
    - Experiments come and go – people change their ideas
    - Technology trends.
    - Disruptive technologies – AI/ML.

We're going to need 10 nodes for a month. But we may make a mistake and need to do over, so let's double that and ask for 20 nodes.

Two weeks in…
Oh, we weren't ready, so we missed the first two weeks, and we have to get it all done in only two weeks, so can we have 40 nodes now?



Proactively estimating need

- To meet these challenges we use a requirements-based approach:
  - Develop metrics to measure scope.
    - Statistics - NP tasks scale linearly with dataset size.
    - Time - users expect that compute will be done in a specific time period.
  - Gather user requirements using these metrics.
  - Review the requirements.
    - sometimes users are very bad at the guesswork part.
  - Develop a plan to meet these requirements
    - Do the resources exist onsite?
    - Are resources available offsite?
  - Iterate with users.
  - Identify funding sources (if needed).

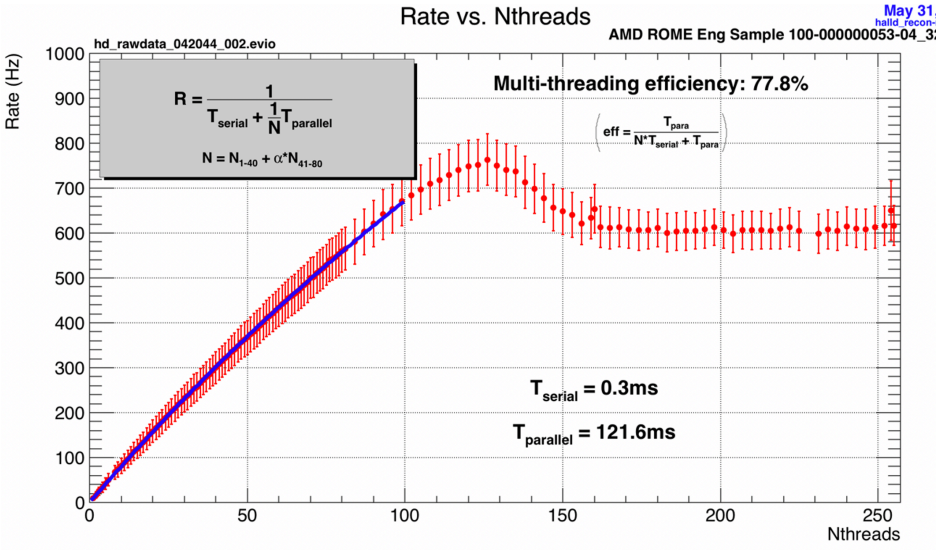
How many events do I need to process in parallel?

$$\frac{\text{(Number of events)}}{\text{(required total time)*(processing rate in ev/sec)}}$$

	Parameter	Constants	Comissioning FY18	FY19	FY20	FY21
5	Weeks of running		22	28	32	10
7	Efficiency		0.6	0.6	0.6	0.6
8	Effective weeks of beam time		13.2	16.8	19.2	6
9	Effective hours of beam time		2218	2822	3226	1008
11	Average event rate during beam, in Hz		12000	12000	12000	12000
13	**RECONSTRUCTION**					
14	Time to reconstruct a single event, in seconds		0.35	0.35	0.35	0.35
15	Reconstruction in CPU weeks per week of running		4200	4200	4200	4200
16	**Reconstruction - hours of CPU time per pass**		9.3 M	11.9 M	13.5 M	4.2 M
17	Reconstruction passes	2				
18	**Reconstruction - total hours pf CPU**		**19 M**	**24 M**	**27 M**	**8 M**
20	**Calibration**					
21	Fraction calibrated	5%				
22	Passes to calibrate	5				
23	**Calibration - total hours of CPU time**		**2.3 M**	**3.0 M**	**3.4 M**	**1.1 M**
25	**Analysis**					
26	Percentage of events surviving trains etc	10%				
27	Percentage of reconstruction load	10%				
28	**Analysis runs**	5				
29	Analysis		**0.47 M**	**0.59 M**	**0.68 M**	**0.21 M**
31	Total non-MC		**21 M**	**27 M**	**31 M**	**10 M**
34	**MC**					
35	Time to generate - seconds per event		0.5	0.5	0.5	0.5
36	Time to reconstruct - seconds per event		0.35	0.35	0.35	0.35
37	Total CPU time per event - seconds per event		0.85	0.85	0.85	0.85
38	Physics dataset size - events		9.6E+09	1.2E+10	1.4E+10	4.4E+09
39	MC - Hours to reconstruct raw dataset once		**2.26 M**	**2.88 M**	**3.29 M**	**1.03 M**
40	MC dataset ratio to raw dataset size	6				
41	MC runs	1.5				
42	**Total MC**		**20.4 M**	**25.9 M**	**29.6 M**	**9.3 M**

Benchmarks

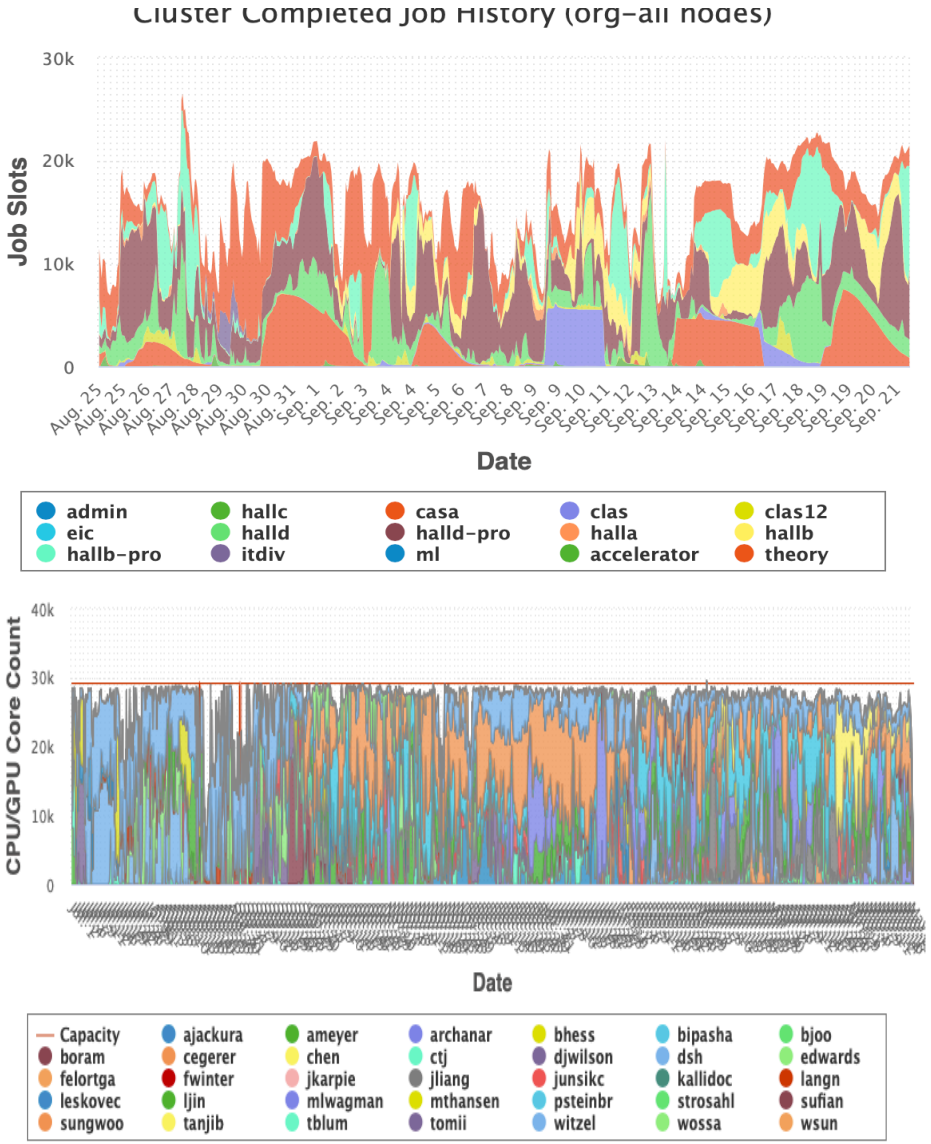
- There are many benchmarks that could be used to estimate computing requirements.
- NP computing workflows are complex, and it is difficult to find a set of general-purpose benchmarks that are a reliable proxy for running user code.
  - Test running user code on a node and measure event processing rate.
  - Usually there is a maximum event rate that can be processed in parallel (jobs). (top figure).
- Given the rate per node we can estimate how many nodes of a particular flavor are needed to run a workflow.
- Lower table shows the Jlab ENP cluster.
  - The 76 newest nodes deliver ~4700 compute units vs ~4100 for the 236 older ones.
  - Can run fewer parallel threads on those nodes ,~5000 vs 7500 – but they run faster.



	Type	clock GHz	mem GB	disk TB	nodes	cores / node	Total cores	Rate / node.	Core perf factor	Normalized cores
2014	Xeon - Haswell	2.3	32	1	104	24	2496	62	0.50	1251
2016	Xeon E5-26797V4 - Broadwell	2.3	64	1	44	36	1584	94	0.51	802
2018	Xeon - Skylake	2.4	96	0.48	88	40	3520	120	0.58	2048
2019/20	AMD EPYC 7502	2.5	256	0.96	76	64	4864	330	1.00	4864

Now we know what the user wants what do we deliver?

- The upper chart shows the utilization of the Jlab cluster used for experimental nuclear physics (ENP).
- The lower chart shows the utilization of the Jlab LQCD clusters.
- The LQCD clusters are 94% loaded on average.
- The use for ENP users is much lower, why?
  - The ENP use is, for the most part, data driven, and researchers are cautious.
    - Before bulk processing can start calibration must be understood.
    - After one batch and before the next researchers validate that the processed data makes sense.
      - Need to account for “head scratching time”.
  - The number of active large resource users in ENP is less than on the LQCD side.
    - There are fewer large “jobs” to consume unused cycles.

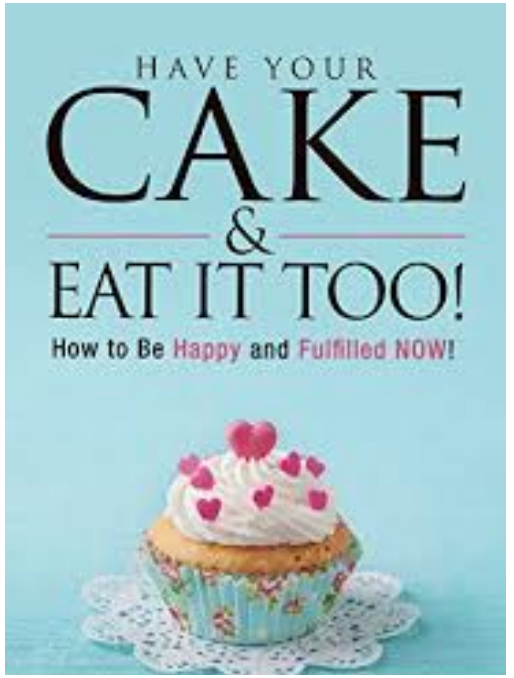


Jefferson Lab

Over provisioning

- Budget for a cluster large enough to accommodate the the sum of all resource requests from the users even though the resource is on average underutilized.
  - In an ideal world everyone gets what they want.
  - Leads to idle resources.
  - Can really upset funding sources.

- It is also impractical to realize.
  - As discussed earlier, resource need timing is driven by factors outside the immediate control of the user.
  - Over provisioning by a small amount doesn't help with tall narrow peaks of demand.
  - Nobody is going to pay for over provisioning by a large amount.
    - (see later for caveat).

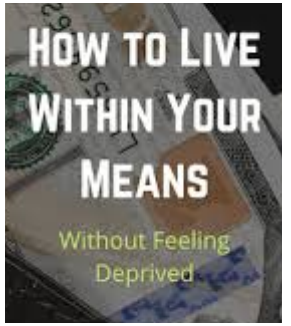


Delayed allocation model

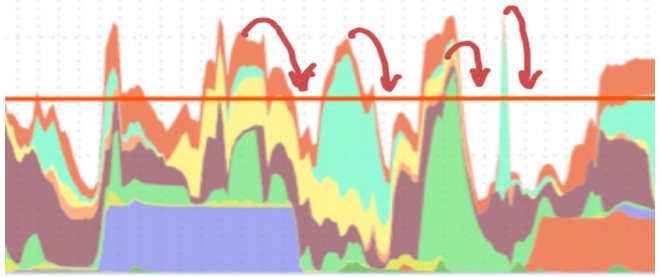
- Constrain the users to a fixed pace defined by an affordable cluster within a constrained budget.
  - It isn't an ideal world, there isn't enough to go around and you have to make do.

- The delayed allocation model:
  - Give the users an allocation and priority.
    - Sum of allocations = capacity – planned_downtime
  - Requests over capacity are delayed but guaranteed.
    - Allocations expire if not used - use it or lose it.
    - Expired allocation is distributed to other users.
  - This is what is happening with the LQCD cluster.

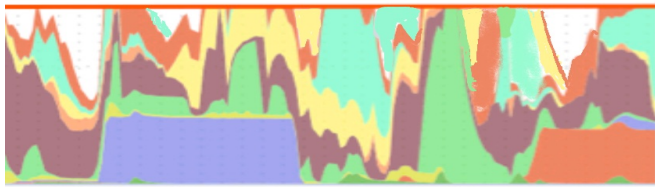
- Problematic for experimenters.
  - There is an expectation that a dataset will be processed in a reasonable time.
    - The definition of "reasonable" varies.
  - Detectors generate data at some rate that needs to match the rate of processing – can't let work pile up!
  - Running the experiment is the cost driver.
    - I can't tell an experiment "tough luck, you had an allocation but didn't use it".



Demand peaks exceed capacity



Delay allocation fulfillment



The beg, borrow, and (as a last resort) buy, approach

- Budget for a cluster large enough to accommodate the average pace and pre-arrange offsite resources to cover the peak load.
  - Allows you to live within the budget and get most of what you want.
  - Use underutilization valleys to pay back what was borrowed.

- There are many large computing resources available that are of a useful scale that can be used to supplement.
  - Collaborating institution
    - Compute resources as a contribution to an experiment – needs management.
  - Open Science Grid
    - Borrow what you need to cover demand peaks and pay it back in the dips.
    - Is an argument for slight overprovision so that there are always cycles to pay back from.
  - Leadership class facility (NERSC etc.)
    - Apply for an allocation and hope you get it – risk of denial.
  - Commercial cloud
    - Buy what you want when you need it – needs money and management.

The challenge is all in the details

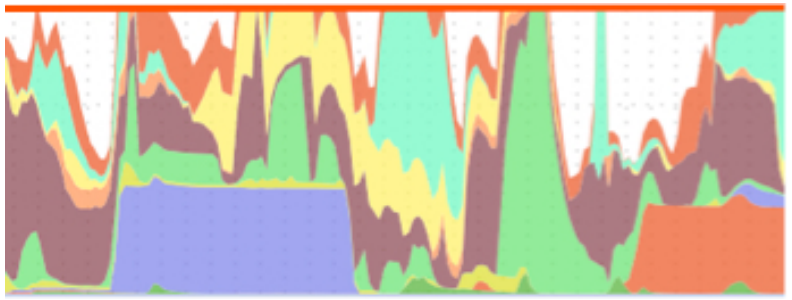
- To run effectively in a distributed mode the details of how a job is run offsite should not be something that concerns the user.

- Ideally users submit jobs, and the scheduling software decides where to run them based on:
  - Available resources.
  - Current load.
  - Type of job.

- In the simple model shown on the right we simply load up the local cluster until full and throw everything over capacity off site.
  - The user sees a virtual system with a larger capacity than exists onsite.

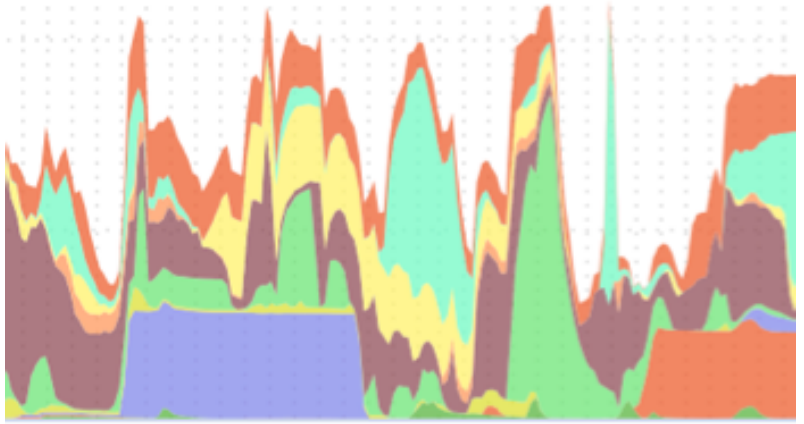
- The real world is more complex. For example jobs that are data intensive are better run close to the data.

What runs onsite – limited capacity



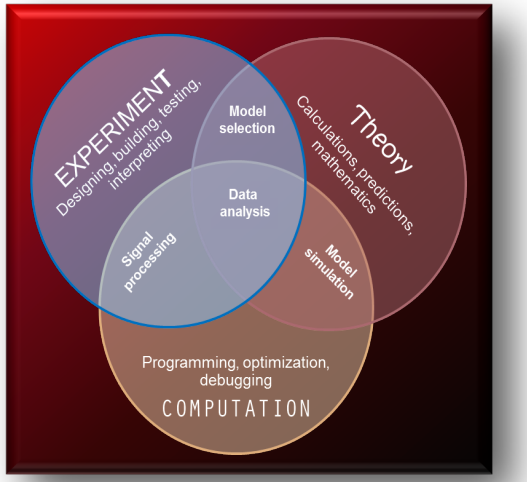
What runs offsite

What the user sees – no capacity limit.



What is the role of the local data center in a distributed computing model?

- Everything at the start of this talk
  - Understanding and interacting with the users.
  - Providing help, documentation, and tools to enable resource use.
  - Enabling access to resources, both local and remote.
  - Making sure that the users can get their work done.



- Everything in the middle of this talk
  - Defining metrics and gathering requirements.
  - Reviewing requirements.
  - Identifying resources and managing them (allocation, procurement, contracting…)
  - Auditing and reporting.

- Everything at the end of this talk
  - Operating local resources for storage and compute that meet the average load.
  - Providing the tools to support load balancing using remote resources.
  - Work with remote resource providers.

- In the end it is the science that is important.

- Computing is a partnership between the user community and data center professionals.


That's all Folks!