EasyChair Preprint

№ 2779

Bear: Cyberinfrastructure for Long-Tail Researchers at the Federal Reserve Bank of Kansas City

Bj Lougee, Michael Robinson, Chris Stackpole, Mark Watson and Greg Woodward

February 26, 2020

Bear: Cyberinfrastructure for Long-Tail Researchers at the Federal Reserve Bank of Kansas City

BJ Lougee
Federal Reserve Bank of Kansas City
bj.lougee@kc.frb.org

Michael Robinson
Federal Reserve Bank of Kansas City
michael.robinson@kc.frb.org

Chris Stackpole
Federal Reserve Bank of Kansas City
chris.stackpole@kc.frb.org

Mark Watson
Federal Reserve Bank of Kansas City
mark.watson@kc.frb.org

Greg Woodward
Federal Reserve Bank of Kansas City
greg.woodward@kc.frb.org

ABSTRACT

The Federal Reserve Bank of Kansas City has developed a new cyberinfrastructure environment uniquely tailored to the long-tail researchers in and around the field of economics. We based our design on our researchers' usage modalities and our experience delivering advanced research computing to domains that are now in more need of computational power and the ability to work with large datasets.

KEYWORDS

cyberinfrastructure, long-tail, economics, training

1 BACKGROUND

Starting in the late '90s increased changes in environmental factors affected research computing environments tailored for economics[5]. Some of the changes were economic models growing in computational complexity, economic research consuming more and larger datasets, and a larger but more sophisticated software and tooling selection. The Center for the Advancement of Data and Research in Economics (CADRE) at the Federal Reserve Bank of Kansas City (FRBKC) developed a strategy that encompassed computing, data, people and training to facilitate better research. Staff designed our computing and data warehousing environments, introducing multiple technology changes needed to support large distributed memory jobs, shared memory jobs, high throughput computing jobs and access to and the provisioning of large quantities of data. Over the last decade, CADRE has iterated and developed a cyberinfrastructure [1] that is tailored for long-tail [4] [6] researchers and their co-authors inside the Federal Reserve System (FRS). The last such iteration, an environment we call Bear, was deployed in the 3rd quarter of 2019 and is the culmination of FRBKC's shared experience from working directly with our researchers and being the leaders in offering cyberinfrastructure to researchers inside of the FRS.

2 USAGE MODALITIES

The vast majority of our jobs are single core or single node jobs. We have a mix of 16 and 28 core nodes for our Bear cluster. For our interactive partitions, we have a default setting of four core jobs, which is why the number of four core jobs is larger than the single and dual core jobs as shown in Figure 1. One issue we look at quite heavily is the time our researchers spend waiting in our queues. We are very sensitive to the time a researcher has to wait for their research to run. Our researchers do policy work in addition to research, and this creates a cycle in which a researcher will only have a short window to complete their computation runs, as shown in Figure 2. Our mean wait time has been about half an hour, with only one very large computational run that increased our wait time to about four and a half days over a small period of time. Although we occasionally receive as many as 2500 jobs per day, the typical number of daily jobs submitted is around 250. Most of the jobs submitted to our Bear cluster need 64 GB of memory or less. We do, however, have jobs that require 128GB or 256GB of memory for interactive exploratory analysis, as shown in Figure 4. We surveyed our researchers and found that 80% require access to a virtual desktop either to do prototyping or working with data that lives within our environment. In addition, a large portion of our researchers only want access to graphical applications such as STATA and Matlab. Because of these usage requirements, we have interactive desktops in our oVirt environment as well as compute nodes strictly used for interactive computing. The type of jobs as well as the requirements of our users has guided our decisions in developing our environment.

3 LESSONS LEARNED AND ARCHITECTURE

3.1 Related Work

Work has been done to reach long-tail researchers by providing access to advanced research computing that better favors their computational needs. The work done by San Diego Supercomputing Center (SDSC) in targeting allocation and scheduling policies that help long-tail researchers is an example of this [8]. While a portion of our long-tail researcher base can take advantage of a more traditional type of cyberinfrastructure, our researchers needs do not fit well with how most systems are currently architected. More specifically, the cyberinfrastructure needs of long-tail researchers in and around the field of economics led to different architectures to better facilitate research and lower the barriers of entry. One such example is how Bently University has architected their environment to solve this issue [9].

3.2 Improvements

For our researchers, our previous HPC environment, named Bull, was a great introduction that allowed us to implement advanced computing concepts. Although we added many new capabilities and flexibility to our new cluster, Bear, we also discovered some pain

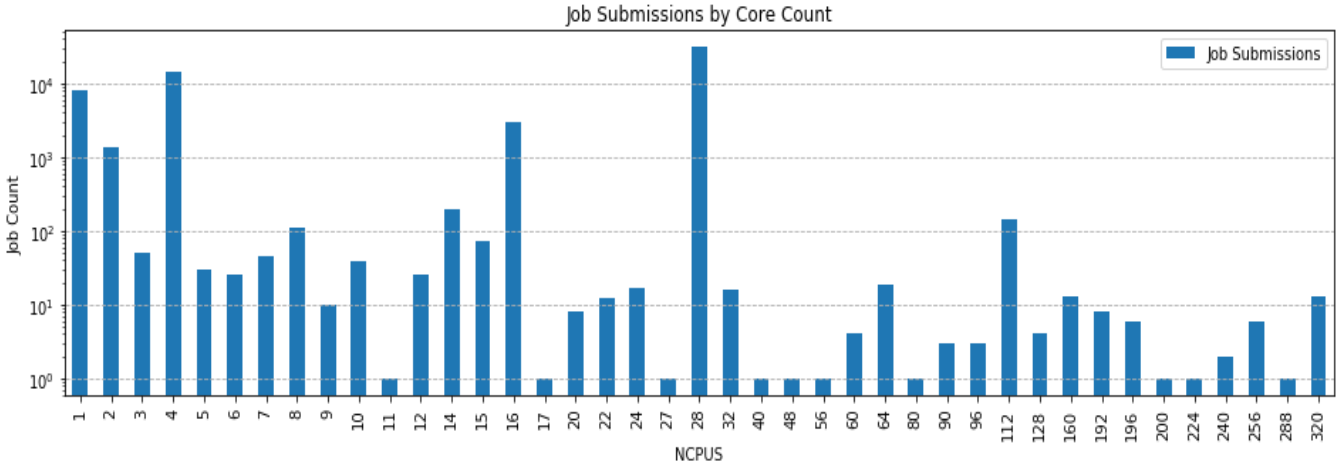

Figure 1: Number of Jobs by Core Count

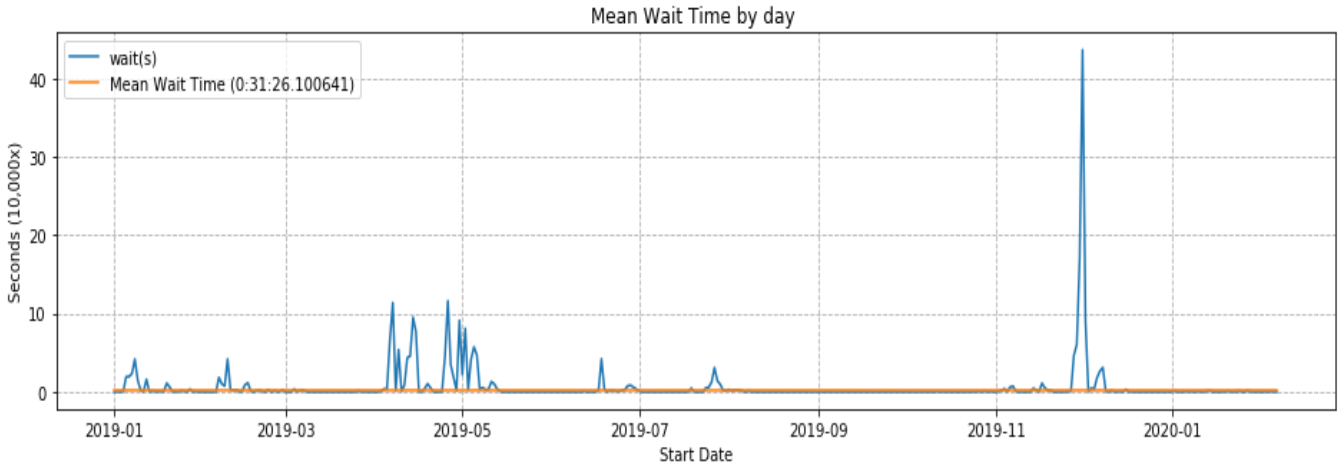

Figure 2: Average Wait Times

points in our previous environment that needed to be addressed. To best understand the pain points and our solutions, it is helpful to understand the architecture of both environments. Bull was more closely aligned with a more traditional High Performance Computing environment that one might find in academia. Users had log-in nodes to access the cluster where they could then launch jobs onto compute resources. Users who were adept at the command-line could use nodes that were configured for just shell access, users who preferred a graphical method could log into a web page that provided graphical options. In both cases, users were then required to submit jobs to the cluster, where different resources were available in different partitions or queues. A few specific design choices on Bull were evaluated for improvement on Bear.

3.3 Interactive computing

Many of our researchers do not have any formal training in Computer Science topics. The most common path to our resources is first exposure through traditional laptops and desktops. The Graphical User Interface (GUI) is comforting and familiar, and thus an important first step. Our goal is to help researchers spend more time focusing on their research than learning entirely new concepts all at once. One way the Bull cluster assisted with this was that we provided a web-based portal and compute resources we called 'guinodes'. This helped our researchers significantly by providing them a familiar environment in which to conduct research, proof-of-concept ideas, test code, and prepare code before requesting more efficient, powerful, and finite shared compute resources. To access these available GUI resources our researchers were required to use Java to start and connect to their desktop session, which would be spawned on the 'guinodes'. There were two very common issues that caused problems for our users and our service desk in this model. The first was that we had no control or ownership of what versions of Java were on our client-end devices. A frequent problem occurred when the client or the cluster updated Java, and

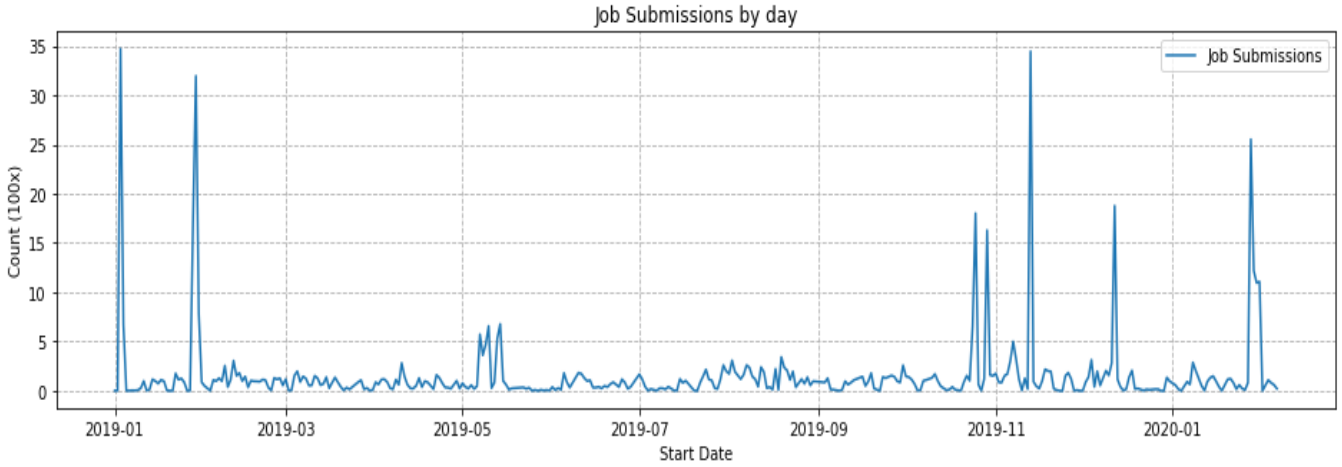

Figure 3: Number of Job Submission Per Day

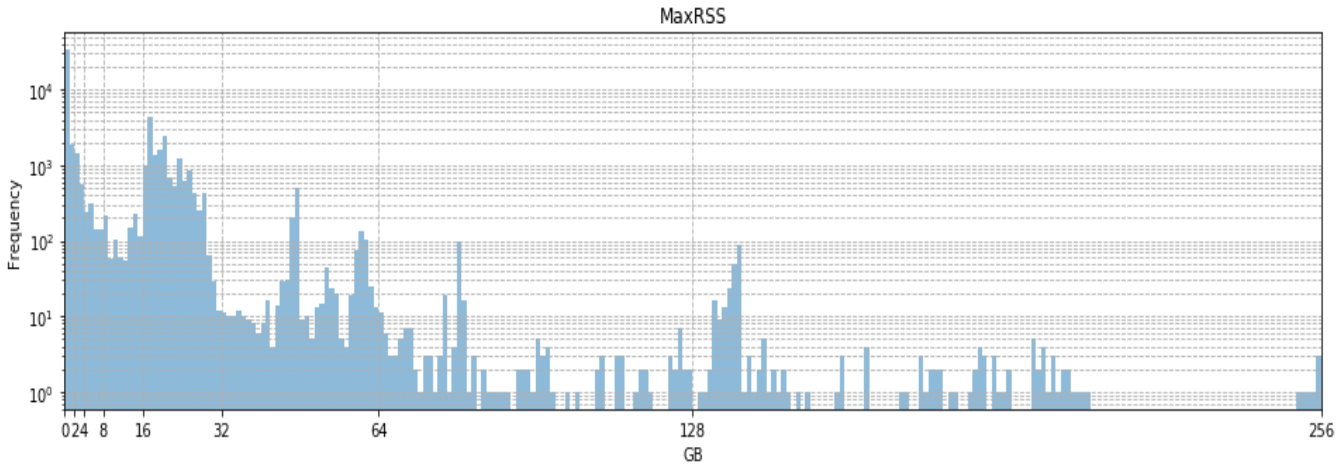

Figure 4: Max Memory

researchers and staff desired to move away from such a constant issue. To avoid the ongoing Java version struggle, we investigated several other technologies but implemented Ohio Supercomputer Center's "OSC Open OnDemand" platform [3]. With this platform, we were able to eliminate the dependency of a Java solution entirely, since all functionality is HTML5 driven. We retained the GUI web interface our researchers enjoyed while adding and enhancing many of the features at the same time.

3.4 Need for Better Interactive Isolation

Another common issue with our Bull cluster was resource constraints when using Linux cgroups ("control group based traffic control filter") pertaining to our available GUI nodes. We successfully set limits on our 'guinodes' processes governing CPU resources; however, we did not implement cgroups for memory. Many of our researchers did not have a firm grasp on their applications' memory requirements. This is even more apparent when they were still in the early stages of proof-of-concept testing and working with large datasets. When memory cgroups were enforced, applications requiring more memory than allocated by their cgroup would be terminated by the Linux kernel by the process known as "Out of Memory Killer" or oom-killer. When this kernel process was enacted, it often left the researchers confused about what happened. Not only did this increase the number of issues filed with our service desk, but it also enabled the feeling among the researchers that the Bull cluster was "unstable". To minimize these initial issues and allow researchers the freedom to refine their code without issue, we removed the memory cgroup which left the Bull environment open to other consequences. Primarily, these consequences were situations in which a user exhausted the available memory and consumed swap space of a shared 'guinode', which eventually affected all users with processes running on that node. While this was not ideal, it was the more acceptable issue in our environment. Using Zabbix, an open-source monitoring solution, we monitored each

node and enabled it to take some preventative measures, which sometimes alleviated the situation. Still, a researcher would occasionally render a 'guinode' unusable. Because of the very nature of these 'guinode' resources, we lacked any features to move processes or users affected by such events to another node, resulting in an occasional loss of research progress. Addressing the memory limitations were more challenging. We did not have the resources to give a dedicated physical desktop system to each of our users, and every technology we explored for sharing physical systems had similar limitations for our scenario. Thus, we looked to virtual systems to address the "cgroup" issue. By implementing Red Hat's oVirt open source virtualization management platform into our new Bear environment, we could provision each researcher a single desktop virtual machine (VM). This provided our researchers the flexibility when they created their virtual desktops to choose and set CPU and memory selections for the work they would need. We configured the maximum values they are allowed to request and should they crash their VM, they only affected themselves. The infrastructure remained scalable on the back-end, and no single user could demand enough resources to affect the whole oVirt system. This technology has given us the ability to do live migrations within the oVirt infrastructure addressing resource contention and usage issues that can arise in a shared environment. By moving the desktop virtualization away from compute hardware, we were able to free up resources while also solving our issue of not being able to migrate, load balance, and isolate our researchers from affecting each other. This not only improved system administration but also improved the researchers' mentality toward the system, giving them the perception that it is more stable and available.

3.5 Parallel Filesystem Choice

Another difficulty with the Bull cluster was with our 'Out of the Box' storage solution that lacked flexibility and performance in certain areas. Although this storage solution was able to scale in performance and capacity, doing so required us to expand in a manner that was cost prohibitive. In addition, one thing that caused frequent concern was that implementing updates to the filesystem required downtime of the entire cluster, reducing our researchers availability to conduct research. These primary difficulties shaped many choices for our next HPC environment, as we not only wished to address these struggles but also to introduce new and developed technologies to enhance the research computing experience. We moved to the Open Source Ceph file-system, commonly referred to as CephFS, and it has been a very welcomed change. Not only are we given much more flexibility in tuning the performance to fit our resources and needs, but the Ceph storage cluster is very resilient when doing rolling updates or against the failure of a node or a disk. The Ceph storage cluster can be upgraded, one host at a time, without cluster downtime. Ceph is also easier for us to achieve flexible performance tuning by replacing parts or expanding Ceph storage blocks on an as-needed basis for a significantly more reasonable price. Moving to CephFS allowed us to have a filesystem that is both cost competitive, but, more importantly, open source, and allowing greater fine tuning for our specific use cases. Lastly, being able to do rolling updates with almost no downtime allows us to have minimal impact on our researchers workflows.

4 TRAINING

4.1 Related Work

It's not enough to have good technology or resources. Researchers also need good training in how to use it. A lot of work has been done to solve the issues around teaching long-tail researchers to effectively use advanced research computing, both as for-credit courses [12] or as workshops for faculty and students [10] [2]. While most of the work done focuses on students, faculty, or staff in academia, we found a lot of this work relevant for our training as well. We have used these ideas as guideposts and are in the process of developing more tailored training.

4.2 Current Training

We focus and tailor our cluster and data ecosystem to the long-tail researchers and their development workflows. Over time, we have found that the cluster and data ecosystem are applicable to other business units within the FRS. The primary user base consists of PhD research economists, economists, research assistants, and data and computer scientists that leverage the systems for academic research. The four related workflows are iterative interactive programming, batch submissions, pipeline and CI/CD. The need for ad-hoc SQL access to research data hosted in the environment is also an important factor. The secondary user base represents more traditional business units performing research and other activities particular to risk modelers, quants, data analysts and data engineers. These users also require the same types of workflows. The technical skill sets required for users to efficiently leverage the cyberinfrastructure ranges from beginner to experienced. Because of these different experience levels we went through an iterative process that CADRE has developed with the following methods for meeting the training needs of it's disparate user base:

- Documentation with examples: We found documentation alone was not enough. Some users required real examples to make the leap from documentation to implementation.
- Lunch-n-Learns: We designed One-hour demonstrations on new or existing technology or tools for users who did not have a lot of time to spare. The key to successful lunch-n-learns has been polling the users to find a topic that is interesting, useful, and timely for our user base. Selecting topics from the perspective of a Cyberinfrastructure Engineer or ACI-REF [7] did not appeal to our user base and resulted in low attendance and very low retention.
- One-on-one training: While not realistic or efficacious for training large groups, we have found one-on-one training produce the best results and retention. Training at this level allows for personalization of the material, often using the trainees' code to demonstrate a concept, skill or tool. This personalization leaves the user with real examples they can apply to their daily work, which aids in retention. Personalized material from one-on-one training provides specific examples for documentation.
- Software Carpentry Workshops (SWC): Hands-on workshops like those provided by the Software Carpentry Foundation [11] are fantastic for teaching large groups, particularly

large groups of people who are new to the topics being covered. However, SWC workshops and workshops in general become less effective when participants have some degree of knowledge on the topics being presented.

Training benefits our user base in many ways, but our primary goal is to enable research. For example, before working one-on-one with a Research Assistant (RA), the RA's software was limited to using a single core and would take hours to complete. After showing them how and where to parallelize their code, the RA's cluster use changed to using multiple cores which would complete in a shorter period of time which resulted in getting faster results and increased research productivity. After demonstrating Slurm [13] job arrays and providing examples, their cluster use increased again until they were using their max allocation on the cluster (Figure 5). We determine our success and the training efficacy based on a researcher's increase in cluster use after the training. We also look at continued use of the cluster in conjunction with continued requests for assistance.

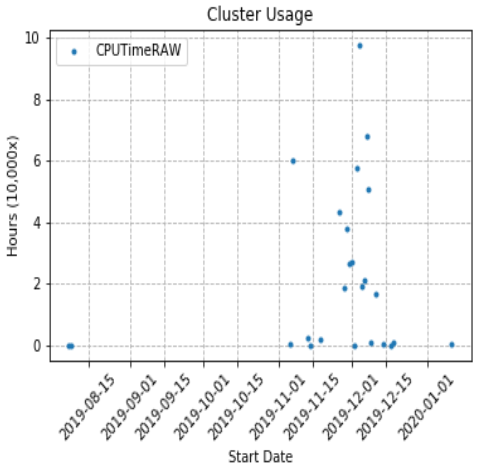

Figure 5: Training Impact

Working with users in the programming language they are familiar with, rather than trying to introduce them to a possibly more efficient programming language, has also improved use and retention. We found most users had little interest in learning a new language unless they could see significant benefits. The added complexity of learning a new language plus the new skill to accelerate their research proves to be too much for most. Enabling research is not the only benefit of our training, as it also provides personal growth opportunities for each user. As an example, the environment prepares RAs to be fluent in navigating and efficiently leveraging most cyberinfrastructures in the country. The entry-level training we provide, a modified Software Carpentry lesson called HPC Carpentry, gives the RAs basic fundamentals that will help them succeed when they leave to pursue their PhD or enter an industry.

5 CONCLUSION

CADRE has a long and storied tradition of continually evolving and improving our environment to meet our researchers' needs within the FRS. Our new Bear environment is a continuation of our success as a leader in providing advanced research capabilities to the FRS. Having the unique position of being inside the FRS and working solely with our long-tail researchers has allowed us to specialize and focus on providing an environment that is both unique and well-positioned to continue to give researchers the tools necessary for their work.

6 FUTURE WORK

6.1 Reproducibility

On top of the great work we have already done, we still have more planned to enhance the environment we have built for our long-tail researchers. We see a need to enhance our environment by giving our researchers the ability to access containerization both for workflow management and to help solve issues around reproducible science. In the last few years, several of our researchers started to use containerization. The technology those researchers and our cyberinfrastructure team uses is Singularity, the same technology used in many national centers and academic institutions. Our first adopters of Singularity are those which have built containers using more sophisticated methods than our average user base. Because of this, we see a need to have Singularity containers built on our cluster with an intuitive, easy-to-use interface that will lower the barrier of adoption. In addition to workflow management, we see more researchers using machine learning and data science techniques. Using readily available containers created by upstream projects has been the method of choice for most of these researchers. The ability to obtain containers with pre-set environments already set up and then converting those into Singularity containers that can run inside of our environment reduces the burden on our researchers to build the complex environments required to conduct research. Another reason for using containers is that our researchers have co-authors at different academic institutions and Reserve Banks who want to share a single file that contains their work without the hassle of setting up and replicating the environment to do research.

6.2 Further Research

Training is another key area in which we can improve. We have been teaching HPC Carpentry courses to our researchers for a few years. We want to expand our training beyond what we currently have and more directly target training that will enable our researchers to advance their research. To achieve this goal, we need to understand our researchers' computational skill level as well as the software stacks they use for their research. We will be conducting a survey to achieve these goals.

ACKNOWLEDGMENTS

A huge thank you to Brett Currier and Elizabeth Willoughby for their work on this paper.

REFERENCES

[1] Daniel E Atkins, Kelvin K Droegemeier, Stuart I Feldman, Hector Garcia-Molina, Michael L Klein, David G Messerschmitt, Paul Messina, Jeremiah P Ostriker,

and Margaret H Wright. 2003. Revolutionizing science and engineering through cyberinfrastructure. *Report of the National Science Foundation blue-ribbon advisory panel on cyberinfrastructure* 1 (2003).

[2] Andrew Fitz Gibbon, David A. Joiner, Henry Neeman, Charles Peck, and Skylar Thompson. 2010. Teaching High Performance Computing to Undergraduate Faculty and Undergraduate Students. In *Proceedings of the 2010 TeraGrid Conference (TG '10)*. Association for Computing Machinery, New York, NY, USA, Article Article 7, 7 pages. https://doi.org/10.1145/1838574.1838581

[3] Dave Hudak, Doug Johnson, Alan Chalker, Jeremy Nicklas, Eric Franz, Trey Dockendorf, and Brian L. McMichael. 2018. Open OnDemand: A web-based client portal for HPC centers. *Journal of Open Source Software* 3, 25 (May 2018), 622. https://doi.org/10.21105/joss.00622

[4] D. S. Katz, D. Hart, C. Jordan, A. Majumdar, J. P. Navarro, W. Smith, J. Towns, V. Welch, and N. Wilkins-Diehr. 2011. Cyberinfrastructure Usage Modalities on the TeraGrid. In *2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum*. 932–939. https://doi.org/10.1109/IPDPS.2011.239

[5] BJ Lougee, Tim Morley, and Mark Watson. 2018. The Road to Cyberinfrastructure at the Federal Reserve Bank of Kansas City. *Technical Briefings* (April 2018). https://doi.org/10.18651/tb/tb1802

[6] Richard L Moore, Chaitan Baru, Diane Baxter, Geoffrey C Fox, Amit Majumdar, Phillip Papadopoulos, Wayne Pfeiffer, Robert S Sinkovits, Shawn Strande, Mahidhar Tatineni, et al. 2014. Gateways to discovery: Cyberinfrastructure for the long tail of science. In *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment*. 1–8.

[7] Henry Neeman, Aaron Bergstrom, Dana Brunson, Carrie Ganote, Zane Gray, Brian Guilfoos, Robert Kalescky, Evan Lemley, Brian G Moore, Sai Kumar Ramadugu, et al. 2016. The advanced cyberinfrastructure research and education facilitators virtual residency: Toward a national cyberinfrastructure workforce. In *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale*. 1–8.

[8] Shawn M Strande, Haisong Cai, Trevor Cooper, Karen Flammer, Christopher Irving, Gregor von Laszewski, Amit Majumdar, Dmistry Mishin, Philip Papadopoulos, Wayne Pfeiffer, et al. 2017. Comet: Tales from the long tail: Two years in and 10,000 users later. In *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact*. 1–7.

[9] Jason Wells and J Eric Coulter. 2019. Research Computing at a Business University. In *Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning)*. 1–5.

[10] Greg Wilson. 2006. Software Carpentry: Getting Scientists to Write Better Code by Making Them More Productive. *Computing in Science & Engineering* (November–December 2006).

[11] Greg Wilson. 2014. Software Carpentry: lessons learned. *F1000Research* 3 (2014).

[12] Lucas A. Wilson and S. Charlie Dey. 2016. Computational Science Education Focused on Future Domain Scientists. In *Proceedings of the Workshop on Education for High Performance Computing (EduHPC '16)*. IEEE Press, 19–24.

[13] Andy B Yoo, Morris A Jette, and Mark Grondona. 2003. Slurm: Simple linux utility for resource management. In *Workshop on Job Scheduling Strategies for Parallel Processing*. Springer, 44–60.