Comet - Tales from the Long Tail - Two Years In and 10,000 Users Later

Shawn M. Strande[a]
strande@sdsc.edu

Haisong Cai[a]
cai@sdsc.edu

Trevor Cooper[a]
tcooper@sdsc.edu

Karen Flammer[a]
kflammer@sdsc.edu

Christopher Irving[a]
cirving@sdsc.edu

Gregor von Laszewski[b]
kevinwangfg@gmail.com

Amit Majumdar[a]
majumdar@sdsc.edu

Dmistry Mishin[a]
dmishin@sdsc.edu

Philip Papadopoulos[a]
phil@sdsc.edu

Wayne Pfeiffer[a]
wayne@sdsc.edu

Robert S. Sinkovits[a]
sinkovit@sdsc.edu

Mahidhar Tatineni[a]
mahidhar@sdsc.edu

Rick Wagner[c]
wagner@globus.org

Fugang Wang[b]
kevinwangfg@gmail.com

Nancy Wilkins-Diehr[a]
nwilkins@sdsc.edu

Nicole Wolter[a]
nickel@sdsc.edu

Michael L. Norman[a]
mlnorman@sdsc.edu

[a] San Diego Supercomputer Center, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093-0021
[b] School of Informatics and Computing, Indiana University, 107 S. Indiana Avenue, Bloomington, IN 47408-7000
[c]University of Chicago, 5802 South Ellis Avenue, Chicago, IL 60637

ABSTRACT

The *Comet* petascale system (ACI #1341698) was put into production as an XSEDE resource in early 2015 with the goal of serving a much larger user community than HPC systems of similar size. The *Comet* project set an audacious goal of reaching over 10,000 users in its 4-years of planned operation, a goal which has now been achieved in less than two years, due in large part to the adoption of science gateways, and as a result of policies that favor smaller allocations, and thus support more users. Here we describe our experiences in operating and supporting this system, highlight some of the important science that has been enabled by *Comet*, and provide some practical lessons we have learned by operating a system designed for the long tail.

Categories and Subject Descriptors

C.5.1 Super (very large) computers, K.6 Management of computing and information systems

General Terms

Design, Management

Keywords

High performance computing, high throughput computing, science gateways, virtualization, GPU, solid-state drive, parallel file system, scientific applications, user support

PEARC17, July 09-13, 2017, New Orleans, LA, USA

ACM ISBN 978-1-4503-5272-7/17/07.
http://dx.doi.org/10.1145/3093338.3093383

1. INTRODUCTION

Comet was designed and is operated with scheduling and allocations policies targeted at the "long tail" of science [1]. This means that *Comet* is targeted primarily at small and modest scale computing jobs, and those that require specialized software environments that are not typically found on traditional clusters. *Comet* is also tailored to serve science gateways, which provide easy to use web interfaces to HPC systems on behalf of well-defined research communities. Additional flexibility is provided via Virtual Clusters (VCs), which provide a near-bare metal performing computing resource upon which research teams can provision custom software for their communities.

Key strategies that *Comet* uses to serve the long tail include:

- Cap allocations for standard research allocations at 10M SUs in order to support more allocations as smaller levels.
- Encourage allocations by science gateways by allowing more than the 10M SU limit, and exempting them from the standard XSEDE allocations reconciliation process, which would otherwise reduce their gateway award from the recommended amount.
- Use Trial accounts to give users 1,000 core-hours within 24 hours of their request, which allows them to quickly get on *Comet* and assess it for their computing needs.
- Support shared nodes jobs, which are used extensively by high throughput computing applications, and science gateways.
- Allow long running jobs up to 1 week for gateways, with longer times available on request.

- Provide a significant computing capacity of ~2 petaflop/s, configured to serve the 98% of XSEDE jobs (50% of XSEDE core-hours) that use fewer than 1,000 cores.
- An interconnect that provides full bisection bandwidth up to 1,728 cores, coupled with maximum job limits of the same size to reduce idle cycles, and improve throughput for jobs.
- A 7PB raw Lustre-based *Performance Storage* at 200 GB/s bandwidth for both scratch and allocated persistent storage, as well as 6 raw PB of *Durable Storage* for data reliability.
- User-accessible local flash drives on all compute nodes to improve I/O for the user jobs, and alleviate load on the shared, parallel file system.
- Provide the option for fully virtualized software stacks with near zero performance overhead.
- Employ an "on-ramping" service, which results in a fully operational software environment that can be instantiated via the Slurm scheduler using tools developed by the project.
- Support the rapid growth in GPU computing that is being driven in large part by the maturity of 3rd party community codes that are optimized for GPUs.

Following a rigorous acceptance and reliability testing process, SDSC's *Comet* system entered production on May 13, 2015 as an allocated resource through the Extreme Science and Engineering Discovery Environment (XSEDE) [2]. Since then the system has provided nearly 600M core-hours across a wide range of disciplines. Table 1 summarizes the key architectural features of *Comet*.

2. IMPACT

Since entering production roughly 24 months ago, *Comet* has supported 508 research allocations, 876 Startup allocations, and 78 Educational allocations from 393 unique institutions. *Comet* has been quickly adopted by the community, with requests exceeding the available time by a factor of 3, and recommended awards by a factor of 2. **These allocations have resulted in over 678 publications across a wide range of disciplines that have cited the use of *Comet* - a number that does not include the many publication that used gateways running on *Comet*. For example, CIPRES alone cites over 3,000 publications, many of which were done using *Comet*.**

The following are a few examples of the scientific advancements that have been enabled by the unique features of *Comet*, including the Virtual Cluster capability, science gateways, and the GPU nodes.

2.1 Science Higlights

Comet Virtual Clusters used to help confirm the discovery of gravitational waves

In February 2016 the NSF announced that scientists for the first time had detected gravitational waves in the universe as hypothesized by Albert Einstein 100 years ago, opening up a new era of exploration for astronomers and astrophysicists. *Comet* was one of several high-performance computers used by researchers to help confirm that landmark discovery before a formal announcement was made. LIGO researchers consumed almost 630,000 hours of computational time on *Comet*, using the the Virtual Cluster (VC) interface for integration with the Open Science Grid (Fig. 1) [3].

One of the most detailed genomic studies of any ecosystem to date used Comet and SDSC's CIPRES science gateway

Table 1 Comet System Configuration

System Component	Configuration
Intel Haswell Standard Compute Nodes	
Node count	1,944
Clock speed	2.5 GHz
Cores/node	24
DRAM/node	128 GB
SSD memory/node	320 GB
NVIDIA Maxwell K80 GPU Nodes	
Node count	36
CPU cores:GPUs/node	24:4
CPU:GPU DRAM/node	128 GB:40 GB
Large-memory Haswell Nodes	
Node count	4
Clock speed	2.2 GHz
Cores/node	64
DRAM/node	1.5 TB
SSD memory/node	400 GB
Storage Systems	
File systems	Lustre, NFS
Performance Storage	7 PB raw/5 TB usable
H	

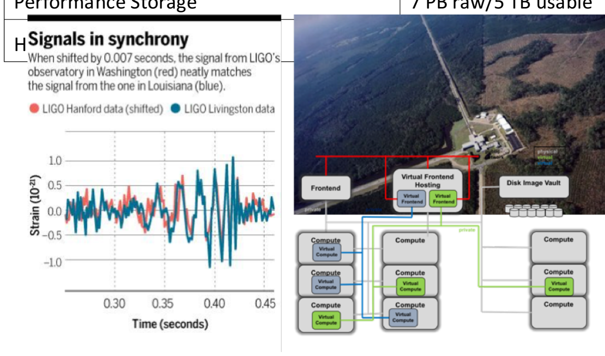

Figure 1. OSG use of Comet VC for gravitational wave confirmation. Image Credit: Caltech/MIT/LIGO Laboratory

This work revealed an underground world of microbial diversity while adding dozens of new branches to the tree of life. Scientists reconstructed the genomes of more than 2,500 microbes from sediment and groundwater samples collected at an aquifer in Colorado. (Fig. 2) [4, 5].

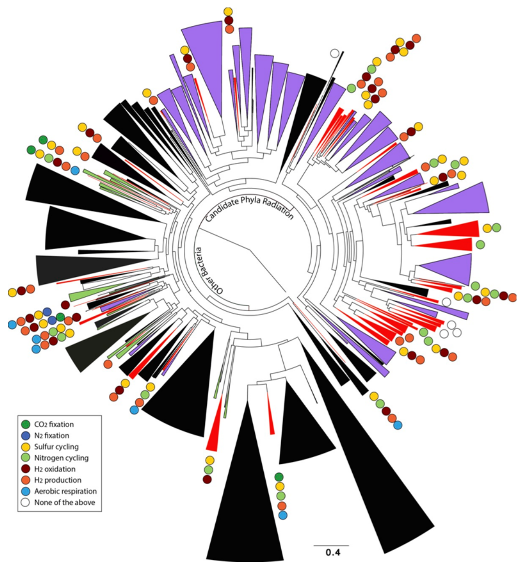

Figure 2 CIPRES gateway assists in understanding microbial diversity. Credit: Banfield Group

Comet's GPU nodes used in help in the search for neutrinos

The IceCube neutrino observatory found the first evidence for astrophysical neutrinos in 2013 and is extending the search to lower energy neutrinos. One of the steps in the IceCube simulation production, photon propagation in the ice, is well suited to run on GPUs. XSEDE ECSS staff, working with IceCube team, have successfully integrated their CVMS-based data analysis workflow into Comet's GPU nodes, which now run this analysis with a speedup of 100x over CPUs (Fig. 3).

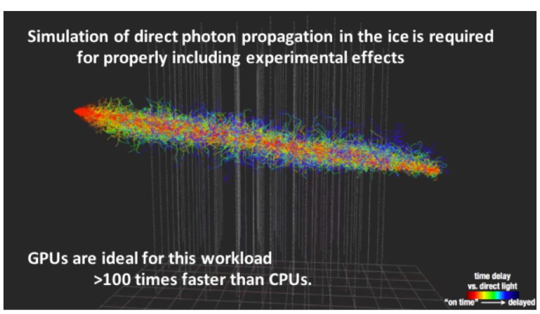

Figure 3 Comet's GPU nodes are accelerating the search for neutrinos. Image credit: Gonzalo Merino, IceCube

3. LESSONS LEARNED

Based on a user survey conduced in 2016, the community has embraced *Comet* for its excellent throughput, user support, mix of CPU, GPU and large memory nodes, and the software environment that supports a wide range of computing modalities. Here we describe some of the key lessons learned during the first two years that have made *Comet* successful and suggest some future work to adapt to the strong demand and more fully understand the usage patterns and trends associated with serving the long tail.

System design, operational, and allocation policies have been effective at supporting the long tail

Comet jobs are capped at 1,728 cores, and upon user request, can be scheduled to run within a single rack. This results in jobs enjoying full bisection interconnect performance. Considering the most recent XSEDE data, this easliy accommodates the SU weighted average job size of 1000 cores considering all XSEDE resources (Fig. 4). By contrast, the average job size on *Comet*, weighted by *Comet* SUs is about 400 cores (Fig 5). The unweighted jobsize average is about 30 core, a reflection of the many smaller jobs that make up gateways, and high throughput computing jobs, which are a substantial portion of the Comet job counts.

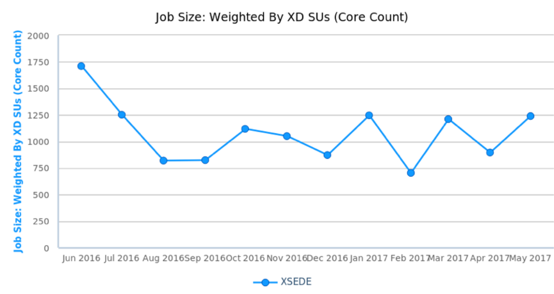

Figure 4 XSEDE Average jobs size, weighted by SU is about 1000 cores (June 1, 2016 - May 31, 2017)

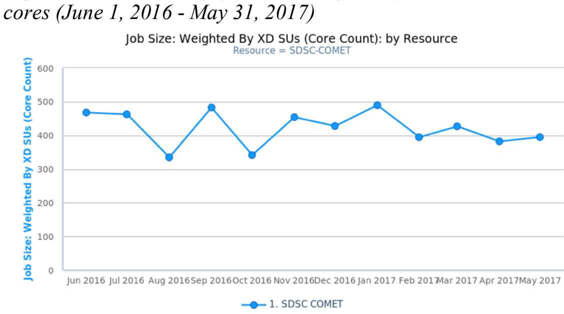

Figure 5 Comet averge jobs size, weighted by Comet SUs is about 400 cores (June 1, 2016 – May 31, 2017)

Supporting modest scale jobs contributes to lower wait times since smaller jobs are easier to schedule, which can help with overall higher utilization. *Comet* is typically running at 90% utilization or higher (Fig. 6).

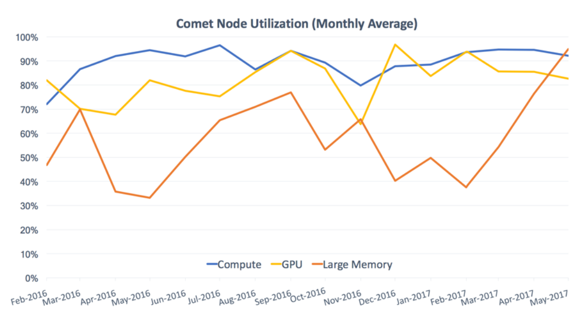

Figure 6 Comet utilization, by node type

The integration of OSG and its support for high throughput computing workloads has added to the already high job counts coming from science gateways, and the many small and modest scale users. Over the past year, *Comet* ran nearly 6M jobs, which is 3x more than any other system in XSEDE. Initially, the high job submission rate caused performance bottlenecks with the Slurm scheduling software in SDSC's custom accounting add-on, which assesses every job submission for expected charges and compares those to the current allocation state to avoid overcharges. Because each allocation's current balance consists of several parts, including current balance known to XSEDE, jobs which have already finished after last synchronization with XSEDE, jobs currently queued and each having a maximum walltime and resources requested, it became an IO-intensive procedure to compute the value for each job, especially when a user submits thousands of those in a short period of time. The SDSC systems team deployed a memchached database for caching these values and reusing them for some period of time with updating as job submissions are processed which allowed an increase in the possible number of job submissions by a factor of thousands.

There is a cautionary tale here about demand for a successful resource like *Comet*. Ultimately, the amount of time made available to the allocations process is where supply meets demand. Recently, we have seen requests for large transfers, partly because of the success of the system, and partly because of the diverse composition of XSEDE resources. As Figure 8 shows, there has been an upward trend in expansion factors, and we feel that *Comet* has reached a point where a more selective review of transfers and a smaller amount of time will be made available to upcoming XRAC to slow this trend. Similar trends can be seen in other XSEDE resources, indicating the ongoing pressure that Service Providers face in meeting demand. Taking action like this is in keeping with the long tail approach of ensuring that *Comet* continues to deliver good throughput.

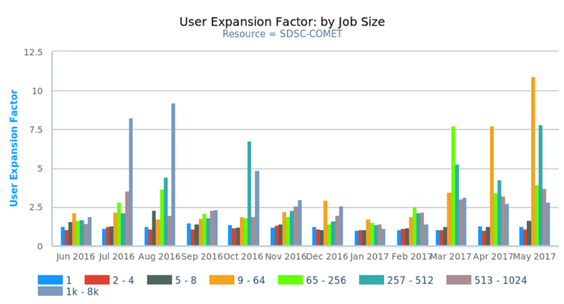

Figure 7 Comet expansion factors, by job size

Science gateways are effective at broadening impact

Comet was designed to favor gateways. This is manifested in the job size limits, allocation policies that favor gateway PIs, ancilliary node types to support gateway front-ends, and auxiliary storage that separates some gateway I/O from the parallel file system. In January of 2017, driven largely by gateways, *Comet* reached a milestone of serving nearly 18,000 unique users (Table 2). Allocations data and usage data collected from gateway PIs, shows that **science gateways serve approximately 500 users for every 1M SUs, whereas this drops by two orders of magnitude to 6 users/1M SU for traditional research allocations**. Importantly, science gateways ease traditional support load since users are submitting jobs via a web interface developed by the community rather than via the command line, and therefore frontline use support comes from the gateway team. The growth of gateways has been remarkable. The I-TASSER gateway [6] was recently awarded over 7M SU on *Comet*, and in a short few months, has provided access to over 8,000 unique users via *Comet.* As of December

Table 2 Comet User Counts as of Jan. 2017

User Type	Counts
Traditional login	2,604
CIPRES	6,310
I-TASSER	8,015
All other gateways	951
Total unique user counts	**17,880**

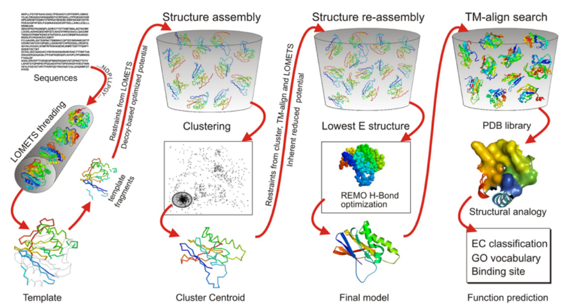

Figure 8 I-TASSER Workflow

Table 2 Science Gateways running on Comet

Gateway	Area of Science
Cipres; UC San Diego	Systematic and Population biology
Neuroscience Gateway; UC San Diego	Neuroscience
SEAgrid; IU	Chemistry, engineering
I-TASSER; U Michigan	Biochemistry, molecular structure and function
Ultrascan3; UT Hlth Sci Ctr Houston	Biophysics
TAS; SUNY Buffalo	XDMoD jobs
waterhub; Purdue U	Earth sciences
SciGaP; Indiana U	Gateway development
ChemCompute; Sonoma State U	Chemistry
COSMIC2; UC San Diego	Biochemistry, molecular structure and function
dREG; Cornell U	Veterinary medicine, gene expressions
UCI Social Science Gateway; UC Irvine	Anthropology
vdjserver; UT Southwestern Med Ctr	Molecular Biosciences

2016, 77% of all active users ran through science gateways. Clearly, this is a major shift in how computational science is done is here to stay, and the rise in science gateways can be see across XSEDE and beyond.

Comet now runs 12 gateways from a wide range of disciplines (Table 3, [7]) with many more in the pipline. Moving forward, we expect additional opportunities to extend *Comet's* role in supporting gateways through interactions with the recently announced Science Gateway Community Institute [8], which will provide support, consulting services, and best practices for those current and future gateway developers.

Better metrics are needed to characterize science gateway usage

With the rapid growth in the gateway community, comes a need to better understand the underlying usage patterns, and user characteristics. XDMoD has become an important tool for NSF and Service Providers to report and review usage. However, while aggregate gateway job statistics are now part of XDMoD, it is not possible to obtain detailed user statistics for those jobs. Rather all jobs for a given gateway appear as running under the account of the gateway user. Thus, it is necessary to work directly with gateway PIs to get the detailed statistics. The authors are working with the XDMoD development team to help define requirements and do testing of new features that will implement detailed gateway accounting. This will be a welcome addition that will help better support the growth of gateways, and aid in the design of future gateway-friendly systems.

XSEDE's Extended Collaborative Support Services (ECSS) project has been helpful in assisting Startup and science gateway developers

XSEDE's ECSS program is an important program for helping users make efficient use of XSEDE's resources. This is even more important for a system like *Comet*, which is strives to bring in users that may be relatively new to HPC. There have been 38 ECSS projects on *Comet*, 15 of which are associated with gateways, and 23 with Startup allocations. These is an important adjunct to standard user support since it pairs programming and HPC expertise with groups that otherwise lack the resources to move applications from their research to national scale cyberinfrastructure. Table 3 gives a representative same of some of these projects.

Trial accounts have been effective at recruiting new users and transitioning them to regular XSEDE allocations

The *Comet* Trial account was implemented as a first-of-its-kind in XSEDE to encourage new users of HPC to explore this system by lowering the barrier to sytem access. The process is a simple one of creating a portal account and submitting a request to XSEDE. Apart from meeting the basic qualification of any PI, there is no additional information needed. **Since inception, 566 trial accounts have been created, with 236 of these now going on to either Startup or Research allocations**. Users have commented on how useful these have been in getting their research plan underway by letting them quickly assess the usefulness of *Comet*.

Virtual Cluster integration with Open Science Grid (OSG) has expanded Comet's role in high throughput computing, and led to growth in virtual cluster adoption on Comet

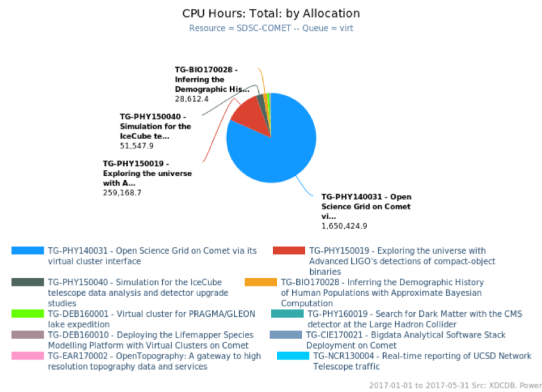

Figure 9 Open Science Grid support for multiple allocations via Comet's Virtual Cluster interface

The OSG virtual cluster interface is now fully integrated with *Comet*. Any XSEDE user with an allocation and an OSG account, is able utilize the OSG interface. Virtual cluster allocations appear in XDMoD under the *virt* queue. Currently, there are ten XSEDE allocations running on *Comet* capable of utilizing virtual clusters. Five of them are via the OSG virtual cluster interface (Fig. 9). Another 2 virtual clusters are in on-ramping process to use virtual clusters. OSGs virtual clusters serve also as one of the use cases to apply virtual clusters.

Table 4 Representative Sample of Comet ECSS projects

Institution	ECSS Project
Valparaiso University	Image Analysis of Rural Photography
UIUC	What is Entrepreneurship?: Multiple Perspectives in Higher Education
University of Alaska, Fairbanks	Hydrological model development and application for a refined understanding of Arctic hydrology
Northeastern U.	Laser-based Structural Sensing and Damage Assessment
Arizona State	Simulating climate-environment-human interactions in coastal South Africa
University of Utah	Insight into biomolecular structure, dynamics, interactions and energetics from simulation: Finding
The Citadel	Finding Real Differences Between Semi-supervised Learning Algorithms Using PJ2
Syracuse University	Exploring the universe with Advanced LIGO's detections of compact-object binaries
LSU	Simulocean Gateway
University of Arizona	Implementation of Sol in OpenTopography
Bay Mills CC	Tribal College and University Distributed Advanced Manufacturing Project
Purdue University	WaterHUB for Hydrologic Modeling and Education
U. Wisconsin-Madison	Simulation for the IceCube telescope data analysis and detector upgrade studies
UCSD	The MP-Complete Science Gateway

Success in onramping Virtual Cluster projects requires access to expertise in the allocated project team

Comet's approach to virtualization is to deliver to the allocated project staff a near bare metal environment allowing a high degree of reuse and customization of the software environment. Consequently, the project staff managing the virtual cluster must possess the necessary systems administration expertise. However, we have found that this is not always the case and therefore more time is required by the comet team to bootstrap these projects. This suggests that additional education beyond those originally envisioned is needed. On the other hand the Indiana University team has taught hundreds of students to set up complex virtual cluster environments as part of their core big data class curriculum [9] resulting in a number of DevOps deployable virtual cluster templates that can be deployed by the non expert [6] and [7]. Tutorials at XSEDE 16, and PEARC'17 and a presentation at NIST [7] supported dissemination. Interfaces to the comet virtual clusters are integrated in a convenient to use tool called cloudmesh client [8] that not only allows us to create virtual clusters on comet but also strives to do so on AWS, Azure, Openstack, and even container based cluster management software.

A study on Virtual Cluster jobs shows good response time to VC requests

With more VCs utilizing the system, we conducted an initial assessment of the virtual clusters. As a VC is integrated as a first class job, each VC node(s) starting/powering on action is equivalent to a Slurm job on the *Comet* queuing system. Hence, VC shares the same resources as the HPC jobs do. To better understand the VC jobs characteristics we studied the waittime, which is defined as the duration between a VC admin requested to start VC node(s) to when the node(s) is started and ready to use. In the queuing system this is translated to the duration between job submission time and job starting time. Figure 11 shows the VC jobs waittime in their submitted order. Figure 12 shows the empirical cumulative distribution of the waitime. Figure 10 shows that most often users only need to wait minutes, while in resource limited times wait times increase to about an hour. Concretely, Figure 11 shows that 3 quarters of the jobs waited less than 3.7 minutes, and 90% of them waited less than 19.6 minutes. Statistics also shows that 67% jobs waited less than 1 minute to get started. The analyses are based on the data we have been collecting more recently (about a month worth of data, ~700 VC jobs). We will continue monitoring this to help provide better services to VC users.

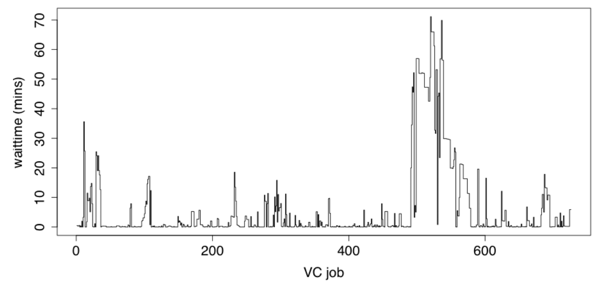

Figure 10. VC jobs wait time (minutes) in submitted order

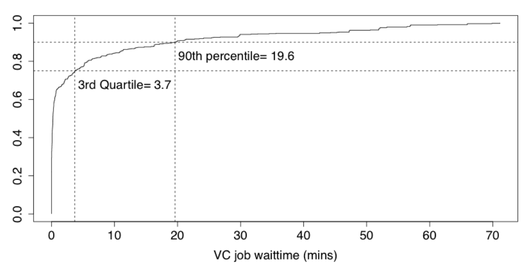

Figure 11. Empirical Cumulative Distribution of VC jobs waittime (minutes)

GPU computing has reached a tipping point

Comet was deployed with 36 GPU nodes, each with 4 Kepler K80 GPUs. At the time the system was proposed, it was unclear what the demand would be for GPUs. Soon after the system was put into production, the GPU resource quickly became fully utilized (Fig. 6). This rapid rise in interest was due in large part to the availability of GPU-enabled community codes, like Amber, NAMD, and others, which provided an easy onramp for users who were themselves unfamiliar with GPU programming. We are also seeing a lot of interest in machine learning, and increasingly are supporting users who run Tensorflow, Torch and other applications for this emerging areas. Following an thorough review of usage, including the *Comet* annual survey, a supplemental proposal for 36 additional nodes each with 4 NVIDIA Pascal P100 GPU was submitted and subsequently awarded. The nodes are currently in build, and expected to enter production on July 1, 2017.

Long Tail Metrics Summary

Table 5 summarizes key metrics presenented in this report, along with a few others not discussed. Taken together this illustrates how *Comet* is serving the long tail of science.

Table 5 Comet Long Tail metrics (since production, May 13, 2015)

Metric	Value (approximate)
# Allocations	1,500
# Institutions	400
# Users	18,000
# of publications	700
# of jobs run (total) # of jobs run (shared-node)	6.9 M 5M
# XD SUs consumed (total) # XD SUs consumed (shared-node)	3,700 M 210 M
# Users/1M SU – Gateway allocations	500
# Users/1M SUs - Research allocations	6

4. CONCLUSIONS

Comet has become a workhorse system for a large fraction of computational research provided through XSEDE. Its unique architecture and operational policies that target the long tail of research provide a blueprint for how such systems can be designed and supported. *Comet* lowers barriers to computing through the use of Trial accounts, allocation policies that favor modest scale computing and gateways, and through system management policies that ensure responsiveness and throughput. With the growth of these new computing modalities and approaches we believe that systems like *Comet* should continue to be an essential element of the NSF's computing portfolio.

5. ACKNOWLEDGEMENTS

Comet is supported by NSF grant: ACI #1341698 Gateways to Discovery: Cyberinfrastructure for the Long Tail of Science. It is provided as a resource to the Extreme Science and Engineering Discovery Environment (XEDE, NSF ACI-1053575), which provide allocation, ECSS and other critical services.

6. REFERENCES

[1] *Gateways to Discovery: Cyberinfrastructure for the Long Tail of Science*, R., Moore, C. Baru, D. Baxter, G. Fox, A. Majumdar, P. Papadopoulos, W. Pfeiffer, R. Sinkovits, S. Strande, M. Tatineni, R. Wagner, N. Wilkins-Diehr, M. Norman. In Proceedings of XSEDE14.

[2] XSEDE: Accelerating Scientific Discovery. J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. R. Scott, N. Wilkins-Diehr, Computing in Science & Engineering, vol.16, no. 5, pp. 62-74, Sept.-Oct. 2014, doi:10.1109/MCSE.2014.80

[3] http://www.sdsc.edu/News%20Items/PR20160225_ligo.html

[4] http://www.sdsc.edu/News%20Items/PR20161102_bacteria.html

[5] http://www.nature.com/articles/nmicrobiol201648

[6] http://zhanglab.ccmb.med.umich.edu/I-TASSER/

[7] https://www.xsede.org/web/guest/gateways-listing

[8] http://sciencegateways.org

[9] G. von Laszewski, F. Wang, B. Abdul-Wahid, H. Lee, G. C. Fox and W. Chang, Cloudmesh in support of the NIST Big Data Architecture Framework, Indiana University, Apr. 2017, Bloomingtion IN 47408, https://laszewski.github.io/papers/vonLaszewski-nist.pdf

[10] G. von Laszewski, W. Chang, F. Wang, B. Abdhul Wahid, G. C. Fox, P. Thakkar, A. Zuniga-Alvarado, R. Whetsel, NIST Big Data Interoperability Framework: Volume 8, Interfaces National Institute of Standards, June, 2017, Special Publication (NIST SP), 1500-8, Gaithersburg, MD 20899, https://laszewski.github.io.papers/NIST.SP.1500-8-draft.pdf

[11] G. von Laszewski, F. Wang, B. Abdhul-Wahid, cloudmesh client, https://github.com/cloudmesh/client

[12] https://cloudmesh.github.io/classes