Scaling the Summit: Deploying the World's Fastest Supercomputer*

Verónica G. Vergara Larrea, Wayne Joubert, Michael J. Brim, Reuben D. Budiardja, Don Maxwell, Matt Ezell, Christopher Zimmer, Swen Boehm, Wael Elwasif, Sarp Oral, Chris Fuson, Daniel Pelfrey, Oscar Hernandez, Dustin Leverman, Jesse Hanley, Mark Berrill, and Arnold Tharrington

National Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, TN, USA

Abstract. Summit, the latest flagship supercomputer deployed at Oak Ridge Leadership Computing Facility (OLCF), became the number one system in the Top500[17] list in June 2018 and retained its top spot in the November 2018 list. An extensive acceptance test plan was developed to evaluate the unique features introduced in the Summit architecture and system software stack. The acceptance test also includes tests to ensure that the system is reliable, stable, and performant.

Keywords: large-scale system testing · high performance computing.

1 Introduction

The United States Department of Energy's Collaboration of Oak Ridge, Argonne, and Livermore National Laboratories (CORAL) project started in 2012 with the goal to procure and deploy up to three pre-exascale systems by 2018. These systems were designed to provide world-class speed and capability to the computing community, advance the Department of Energy's mission, and achieve a necessary step towards exascale.

Oak Ridge National Laboratory (ORNL) and Lawrence Livermore National Laboratory (LLNL) selected IBM POWER-based systems with NVIDIA accelerators, and Argonne National Laboratory (ANL) selected an Intel-based system.

In June 2018, ORNL's system, Summit [15], became the fastest supercomputer in the world as measured by the Top 500 [17] organization. The Summit system not only achieved the highest high performance Linpack (HPL) number, but also took the top spot in the 10th HPCG Performance list [1], and the number three spot in the Green 500 [9] November 2018 list. Furthermore, five of six SC18 Gordon Bell award finalists used Summit, including the two winners.

Due to the scale and complexity of the Summit system, the Oak Ridge Leadership Computing Facility (OLCF) developed a thorough acceptance test (AT) plan that both verified performance targets of individual hardware as well as determined the system's readiness to support the OLCF's user programs by conducting tests of all major components of the system software stack. The AT plan includes four distinct test elements: hardware test (HW), functionality test (FT), performance test (PT), and stability test (ST).

In this work, we describe the multi-month process of the Summit deployment which includes acceptance test plan design, test development and validation, Summit's acceptance phases, and the description of several issues, lessons learned, and best practices uncovered along the way.

2 System Architecture

2.1 Summit

The Summit supercomputer consists of 4,608 AC922 compute nodes each with two 22-core POWER9 (P9) processors and six NVIDIA Tesla V100 (Volta) GPUs. NVLink 2.0 buses with 50 GB/s peak bandwidth connect each POWER9 to three V100s, and each of the three GPUs to one another (see Fig. 1 for connection details). The V100 GPU peak single (double) precision performance is approximately 14 (7) TF, and peak memory bandwidth is 900 GB/s. Each node contains 512 GB DDR4 main memory, and each GPU has 16 GB HBM2 memory. The nodes are connected to two rails of a Mellanox EDR InfiniBand fat tree interconnect. Each node includes a 1.6 TB NVMe device for use as node-local storage. For shared storage Summit is connected to Alpine, the center-wide GPFS file system.

2.2 File Systems

Two different file systems were used during acceptance test: Alpine Test and Development System (AlpineTDS) and Alpine. AlpineTDS was deployed to provide a sandbox environment for testing software, firmware, and configuration changes before they are applied to the main production file system, Alpine. AlpineTDS consists of an IBM Elastic Storage Server (ESS) GL4, which is the basic building block for Alpine. There are 77 GL4s deployed and configured as a single POSIX namespace in Alpine. AlpineTDS is a standalone system and is configured as a separate POSIX namespace.

Each GL4 has four 106-port disk enclosures, connected to two Network Shared Disk (NSD) I/O servers with POWER9 CPUs. The NSD servers act

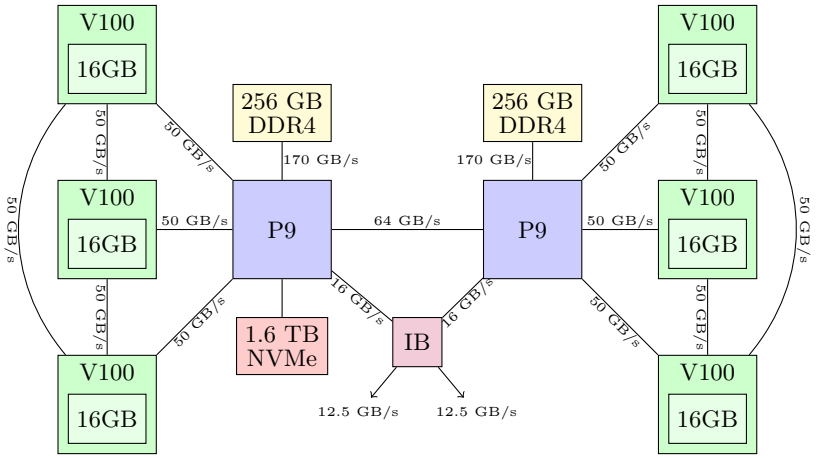

Fig. 1. Summit Node Architecture

as a failover pair. The disk enclosures are connected to NSD servers via 12 Gbps SAS links. Each GL4 has 422 10 TB Near-Line SAS disks. These are organized in two Redundancy Groups (RG), each with 211 disks. The software-based distributed parity RAID services (GPFS Native RAID engine - GNR) running on the NSD servers use an 8+2 RAID 6 scheme built atop of these RGs.

AlpineTDS provides roughly 3 PB usable capacity and performs at around 35 GB/s for large-block sequential read and write I/O operations. Alpine provides 250 PB usable capacity and performs at 2.5 TB/s for large-block sequential reads and writes.

To streamline the Summit acceptance activities, AlpineTDS was deployed and configured first and presented to Summit as a POSIX mountpoint. This allowed more time to properly deploy, configure, test, and validate the Alpine file system without impeding Summit acceptance activities.

3 Acceptance Test

In addition to Summit, the last two supercomputers that the OLCF has successfully deployed, Jaguar [20] and Titan [25], have been ranked number one on the Top500 [17] list. Summit's acceptance test followed the same methodology used for both Titan and Jaguar. The objective for any OLCF acceptance is to develop a set of tests that represent the current OLCF application portfolio and that can adequately simulate the expected production workload.

Summit's acceptance was conducted in two phases: (i) Summit Phase 1 (SP1) included 1,080 compute nodes (60 racks) and was completed in December 2017; and (ii) Summit Phase 2 (SP2) used the full system (256 racks) and was completed in November 2018. Concurrent to both phases, we also executed acceptance of the AlpineTDS and Alpine GPFS file systems.

The AT plan includes four distinct test elements: hardware test (HW), functionality test (FT), performance test (PT), and stability test (ST). SP1 included the following elements: HW, FT, and 3-day ST. SP2 included the following elements: HW, FT, PT, and 14-day ST. AlpineTDS was used for SP1 and for the FT of SP2. Alpine was used for PT and ST of SP2.

Each AT element has specific entry and exit criteria that must be successfully met before the next element can begin. At the entry to each AT element, all compute nodes that will be used in testing must be online and available. In addition, during an AT element, the system software must remain unchanged.

Throughout acceptance, any issue or test failure encountered is documented and a bug is filed with the vendor. The severity of each issue is classified following the definitions shown in Table 1. If a Severity 1 or Severity 2 issue is opened during an AT element, the element cannot be concluded until a fix or workaround is identified.

Table 1. Defect Severity Classification

Severity Level	Priority	Description
Severity 1	Highest	Complete failure of the system, a subsystem, or a unit within the system. Application results that fail to meet correctness criteria.
Severity 2	High	Service is partially interrupted or impaired with significant impact to the AT. Any Sev 1 problems that can be circumvented.
Severity 3	Medium	A problem that impacts specific tests but that does not interfere with the AT. Node failures that cause a running job to terminate abnormally.
Severity 4	Low	A problem that has little or no impact on users and can be bypassed easily.

Hardware Test (HW). The HW consists of complete hardware diagnostics executed by the vendor to ensure that individual parts meet the manufacturer's specifications. The results from HW are provided to ORNL staff for review and archival. The HW element includes diagnostics designed to measure the performance of each POWER9 CPU and each V100 GPU on the system. As part of HW, the vendor also executes the High Performance Linpack (HPL) benchmark [29].

The HW also verifies that telemetry data from various components is available. It is critical to be able to monitor power, temperature, and utilization of the system in real-time in order to automatically adjust different support systems (e.g., water temperature, chillers). For that reason, during HW we also verify that telemetry data can be collected with reasonable performance.

The HW also includes system administration tasks commonly needed in production. First, the full system is rebooted twice to ensure that it can be put back into production in a reasonable amount of time. Then, a collection of multi-node jobs, each running a simple MPI application, is started until the entire system is occupied. A node failure is then simulated, and the test is considered successful

if the failure only impacts the job that was allocated on that node. Finally, the HW also includes thermal protection and emergency power off (EPO) tests.

Functionality Test (FT). The FT demonstrates that basic hardware and software functionality meet essential requirements. The FT includes tests to evaluate the functionality of the application launcher, the scheduler and resource manager, advanced network features, burst buffer, compilers, math and I/O libraries, MPI implementation, and tools. To accomplish this goal a set of benchmarks, mini-applications (miniapps), and real-world applications is used. These were selected to ensure high coverage of features commonly used by scientific application developers. Table 2 summarizes the codes used during the FT phase and each code's test objectives.

Table 2. FT benchmarks, miniapps, and applications

Test	Description
ALCF MPI Benchmarks	MPI bandwidth and latency
AMG[1]	Parallel algebraic multigrid solver for linear systems
CAM-SE[1]	Atmospheric climate modeling code
CUDA & GPU Direct tests	CUDA, CUDA Fortran, CUDA MPS, and GPU Direct
Chroma	QCD application
E3SM (formerly ACME)	Energy Exascale Earth System Model
GTC[4]	Gyrokinetic 3D particle-in-cell application
HACC[2,4]	Extreme-scale cosmological simulation application
Intel MPI Benchmarks	MPI bandwidth and latency
LAMMPS	Molecular dynamics application
LSMS[2,3]	Locally Self-consistent Multiple Scattering application
LULESH[1]	Shock hydrodynamics miniapp
MCB[1]	Monte Carlo benchmark
Minisweep	Radiation transport miniapp with OpenMP 3.1 and CUDA support
NAMD[1,3,4]	Molecular dynamics application
NUCCOR kernels	Nuclear physics miniapp; DLA operations using: LAPACK, OpenBLAS, ESSL; programming models: OpenMP 3.1, OpenMP 4.5, OpenACC
Nekbone[2]	Simulates Nek5000
NVLink Tests	CPU↔GPU, GPU↔GPU bandwidth
NWCHEM[4]	Computational chemistry application
OpenMP 3.1 verification and validation	OpenMP 3.1 specification
OpenMP 4.5 verification and validation	OpenMP 4.5 specification
OpenACC verification and validation	OpenACC specification
Profugus	Radiation transport miniapp; proxy application for Shift
QBOX[2]	First-principles molecular dynamics application
QMCPACK[1,4]	Quantum Monte Carlo simulation code
ScaLAPACK tests	Parallel dense linear algebra (DLA) operations
SNAP[1]	Proxy application for PARTISN
SPEC OMP2012	OpenMP 3.1 functionality and performance
SPEC ACCEL ACC suite	OpenACC 1.0 functionality and performance
SPEC ACCEL OMP suite	OpenMP 4.5 functionality and performance
STRIDE	Stress test for the memory subsystem
STREAM & GPU-STREAM	Measures memory bandwidth
UMT[1]	Radiation transport mimniapp
XRayTrace	Ray propagation miniapp; uses: C++11 threads, OpenMP, OpenACC, CUDA

During FT, each code is compiled with each applicable compiler. Then, a single job for each unique test is submitted. Once each test has completed successfully at least once, the entire set of tests is launched continuously for a period of at least 8 hours. During that period, any job failure is investigated and classified. The test phase is considered complete if there are no job failures, or in the event that there are failures, if the root cause for each failure has been identified and a remediation or a fix exists.

Performance Test (PT). The PT demonstrates that the system hardware and software meet performance and scalability requirements of the CORAL Benchmarks suite [3] defined in the contract with the vendor.

During PT, each test is executed in isolation to minimize disruptions or activity on the system that could negatively impact performance. For that reason, only a subset of test codes are used for this AT element. Individual performance metrics are collected for each test and later used to measure runtime variability under a realistic workload during the ST.

In addition, the results obtained for the CORAL Benchmarks from the vendor at HW are verified to ensure that the reported metrics are reproducible, and the scalability requirements in the contract were met. Table 8 summarizes the figures of merit obtained for each individual benchmark.

Stability Test (ST). The ST demonstrates stability across a mix of simulated code development activity and production simulations. The OLCF test harness [31] is used to fill the entire system with test jobs that vary in terms of number of nodes used and runtime. The executable for each test is built on the system right before its job is submitted, which mirrors normal user behavior. The mixed workload includes benchmarks, miniapps, and real scientific applications selected from the OLCF portfolio.

ST is executed continuously for a predetermined length of time. For Summit Phase 1, a 72-hour ST was executed. For Summit Phase 2, which included the full system, ST was required to be executed for 336-hours without any Severity 1 or Severity 2 failures. ST requires the following criteria:

- **Pass rate:** 95% of jobs executed complete successfully.
- **Correctness:** 100% of jobs that complete must produce correct results.
- **Availability:** 95% of resources must be available at least 90% of the time.

4 Alpine Acceptance

A large-scale file system deployment and acceptance is a complex effort and requires careful resource planning to manage the internal (file system specific)

[1] CORAL Throughput benchmark.
[2] CORAL Scalable benchmark.
[3] CORAL benchmark and upstream versions used.
[4] CAAR version was not used as it was in development.

and external dependencies (network and clients). Alpine is no exception considering the sheer number of components: 32,494 disks, 154 NSD servers, and 40 InfiniBand top-of-rack (TOR) and 10 Ethernet switches.

To simplify the acceptance process the deployment and acceptance team divided the work into manageable subgroups, where the same acceptance tests can be run in parallel in a repetitive manner over multiple subgroups. Each subgroup consisted of eight GL4s arranged in four racks. A series of basic functionality tests were conducted on each subgroup. These included configuring and verifying RGs and building individual and standalone POSIX namespaces on each GL4. Once each namespace was verified for correctness, the performance of each GL4 was tested and verified for sequential and random write and read I/O patterns followed by a metadata test.

Individual performance tests on each GL4 allowed the team to quickly build a performance profile and establish a baseline for comparing each unit's performance. Often, lower performance of a storage unit is a clear indication of a fault within the unit, and it is common for a storage system to have multiple faults at initial deployment. After identifying and clearing out faults on units, the team scaled up the namespace on each subgroup independently. This was the team's first scale up experience with IBM Spectrum Scale technology. Performance tests were again conducted on each subgroup to make sure there were no problems at each scaling level.

After enough subgroups were scaled up and verified for performance, a larger scale namespace was built consisting of half of the available storage hardware. Again, performance was measured and verified to ensure that the scaling results were within expected parameters. Finally, after all subgroups were cleared of problems, a full-scale namespace was built on all Alpine hardware. Correctness was tested and verified. Using roughly 600 Summit compute nodes, the first full-scale performance test was conducted, using sequential and random I/O workloads for write and read operations with 32MB and 16MB I/O sizes. The metadata performance with 32KB write operations and file creation performance on a single shared directory were also tested according to the acceptance test plan.

Lastly, a 2-week stability test was conducted to make sure Alpine was able to stay operational under load and ride out any errors without a downtime. This test overlapped with Summit's ST. During this period, a known I/O pattern was issued from a limited number of Summit compute nodes while the remaining nodes generated a regular production I/O workload against Alpine during execution of acceptance codes.

Table 3 shows the results of Alpine acceptance testing. As shown in Table 3, the file I/O performance at scale was satisfactory. Improvements to metadata performance are expected in late 2019.

Table 3. Summary of Alpine at-scale acceptance performance testing.

Acceptance Test Description	Results
Sequential IOR write/read for 20 min.	2.476 TB/s write (mean) 2.723 TB/s read (mean)
Random IOR write/read for 20 min.	2.381 TB/s write (mean) 3.072 TB/s read (mean)
32KiB creates for 20 min. using IOR	607.007K creates/s (mean)
Run mdtest for 20 sec. in single shared directory	25.465K creates (mean)
Run IOR on a single node single thread for 20 min.	10.126 GB/s write (mean) 6.404 GB/s read (mean)
Run IOR on a single node multi thread for 20 min.	13.708 GB/s write (mean) 14.415 GB/s read (mean)

5 Summit Phase 2 Acceptance

Summit Phase 2 acceptance (SP2) included all acceptance test elements: HW, FT, PT, and a 336-hour ST. In this section, we describe the each element and present a summary of the results obtained.

5.1 Hardware Test (HW)

The vendor completed the delivery and installation of all 4,608 compute nodes on August 4, 2018. On August 30, 2018 the IBM HPC Software Stack (HPC SW) that was targeted for acceptance was released. The HPC SW included the full stack as shown in Table 4.

Table 4. IBM HPC Software Stack Evolution

Feature	Product	Aug. 2018	Dec. 2018	Production	Vendor
Batch Scheduler	Spectrum LSF	10.1.0.6	10.1.0.6[5]	10.1.0.7[4]	IBM
Batch Scheduler	Job Step Manager	10.2.0.7	10.2.0.10	10.2.0.11	IBM
MPI Library	Spectrum MPI	10.2.0.7	10.2.0.10	10.2.0.11	IBM
Math Libraries	ESSL	6.1.0.1	6.1.0.2	6.1.0.2	IBM
Compilers	XL C/C++	16.1.0	16.1.1-1	16.1.1-2	IBM
	XL Fortran	16.1.0	16.1.1-1	16.1.1-2	IBM
	PGI	18.7	18.10-1	19.1	PGI
	clang[6]	5.X	5.X	5.X	IBM
	GCC	4.8.5	4.8.5	4.8.5	RedHat
CUDA support	CUDA Toolkit	9.2.148.1-1	9.2.148.1-1	9.2.148.1-1	NVIDIA
	CUDA Driver	396.47	396.64	396.64	NVIDIA
Parallel File System	Spectrum Scale (GPFS)	5.3.1 eFix 14	5.0.1-2 efix 7	5.0.1-2 efix 8	IBM

5.2 Functionality Test (FT)

There are two types of tests in FT: ones that must be executed successfully on every single compute node, and those that must be executed successfully at

[5] Patched version.
[6] IBM branch.

least once. The set of "every node" tests includes GPU specific tests to evaluate correct functionality of NVLink, GPU_Direct, CUDA libraries, MPI benchmark tests, and memory bandwidth tests for both CPU and GPU memory.

Once the "every node" tests were completed and we verified that the hardware was healthy, we continued to execute the full set of FT tests.

GPU tests: To verify that both the accelerator hardware (V100s) and the accelerator software (CUDA Driver and CUDA Toolkit) were functioning correctly, we used tests derived from samples included in the CUDA Toolkit 9.0 release. The CUDA tests executed are shown in Table 5.

Table 5. Tests used from samples provided with CUDA Toolkit 9.0.

Feature	Samples Test
CUDA	UnifiedMemoryStreams, UnifiedMemoryStreams_GPU_Direct, asyncAPI, batchCUBLAS, concurrentKernels, conjugateGradient, cppIntegration, cudaOpenMP, inlinePTX_nvrtc, radixSortThrust, simpleCUFFT_MGPU, simpleHyperQ, simpleIPC, simpleMPI_MPS, simpleMultiCopy, simpleP2P
GPU Direct	Collective, Pingpong, Stencil
NVLink	p2pBandwidthLatencyTest, simpleP2P, bandwidthTest

Some of the samples were modified to use Multi-Process Service (MPS), CUDA Fortran, GPU Direct, and managed memory. The GPU Direct tests uncovered an issue that prevented direct device-device communication due to the use of cgroups to delineate resource sets. Without this capability, applications cannot take advantage of the fast NVLink interconnect. To work around this issue, the `-step_cgroup n` option had to be used when submitting a batch job to disable job step cgroups. Then, the `CUDA_VISIBLE_DEVICES` environment variable was needed to assign GPUs to specific tasks in the job. A fix for this issue is expected in the upcoming software stack release.

The NVLink tests measure data transfer bandwidth and latency for GPU-to-GPU and GPU-to-CPU memory copies, which we refer to as "device-device" and "host-device" respectively. Both forms of memory copies utilize the NVLink 2.0 buses that connect the V100 GPUs and POWER9 CPU, and cross-socket data copies additionally traverse the CPU X-Bus. To fully evaluate node functionality, a variety of GPU memory copy operation modes were used, including with/without peer-to-peer (P2P) for device-device copies and with/without unified memory (UVA) for host-device copies. The programs `p2pBandwidthLatencyTest` and `simpleP2P` were used for device-device tests, and `bandwidthTest` for host-device tests. The host-device test was modified to support unified memory using the `cudaMallocManaged()` memory allocation API. Although these tests were primarily functionality oriented, they were also useful for understanding relative performance of the various memory copy modes. As expected, direct copies using CUDA were faster than using UVA, and for device-device copies, P2P reduced latency and provided a substantial performance boost. The performance baselines for the tests were 46 GB/s per direction for device-device P2P copies,

and 240 GB/s for host-device CUDA copies involving all six GPUs. The tests were also useful for identifying misbehaving hardware, such as slow GPUs/HBM, misconfigured P2P, and occasionally stray processes leftover from previous jobs that were not properly removed.

MPI tests: To evaluate the Spectrum MPI implementation we used the Intel MPI Benchmarks (IMB) [10] to verify correct functionality and measure the performance provided by this new implementation. In addition, we used an in-house developed benchmark called kickstart to measure application launch time.

IMB test results identified performance issues that caused early timeouts, particularly at scale when using a high number of processes per node (PPN). Upgrades to the stack allowed successful resolution of several performance issues.

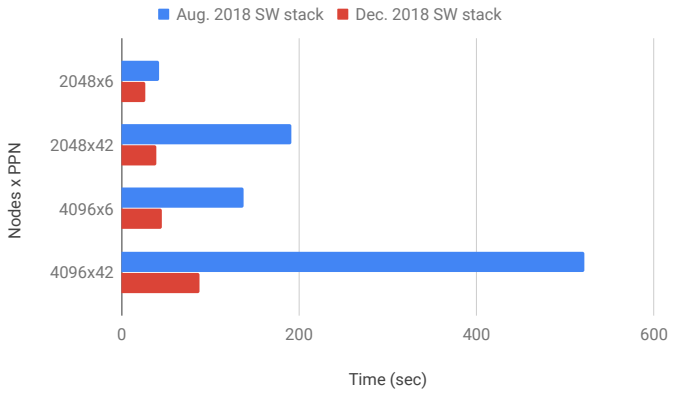

Fig. 2. Application launch times on Summit.

Fig. 2 shows a comparison of the results obtained from running kickstart on Summit using the initial acceptance stack versus the next release. Initially, the job step launcher did not allow full system jobs to launch at a reasonable time. The Dec. 2018 SW stack resolved these performance issues.

In addition, with the Aug. 2018 SW stack, we observed that job steps could not be launched when using 168 PPNs (the maximum value possible that utilizes all hardware threads). The root cause was tracked down to a bug in PMIx which caused the job step to fail.

Memory bandwidth tests: The stream [14] and gpu-stream [13] functionality tests measure memory bandwidth for four basic computational kernels with streaming vector access patterns: copy, scale, sum, and triad. All four patterns exercise both streaming loads and stores. These tests ensure that each node

meets our specified bandwidth targets: 266 GB/s for CPU memory (all patterns), 765 GB/s for GPU copy/scale, and 810 GB/s for GPU sum/triad. For CPU memory bandwith gathered using stream, the test used a single process per node with 42 OpenMP threads (i.e., one thread per CPU core across both sockets). GPU memory bandwith was measured by the CUDA implementation of gpu-stream, which ran with one process per GPU.

InfiniBand Network tests: Acceptance of the InfiniBand network helped validate the proposed performance of the network and identify several issues resulting in the replacement of equipment and software updates. Most notably, we used MPIGraph [11] to test the aggregate system bandwidth when Adaptive Routing is enabled. MPIGraph uses a single rank per node measuring send and recieve bandwidth in a ring communication pattern. The ring monotonically increases the rank distance after each phase until all ranks have communicated with every other rank. The output from MPIGraph shows bandwdith measured from/to all nodes. Using tools to visualize this bandwidth over successive runs it became clear that particular senders and receivers had distinct performance bottlenecks imposing upon their measurements. Using the exact locations of the nodes participating, we identified that a particular switch was underperforming, leading to the replacement of this switch. Subsequent measurements showed improved performance and the elimination of the performance degradations. Our results show that Summit's network can achieve a bisection bandwidth of 95 TiB/s. The latency experiments showed a latency of 1.3μs for 0B messages, 3.54μs for 4KB messages, and 12.67μs for 64KB messages.

Another area of improvement identified in network acceptance was in the measurement of hardware accelerated collectives. Using collective benchmark suites including the OSU Microbenchmarks and the ALCF MPI Benchmark suite, we validated that the hardware accelerated collectives met performance targets. Our testing identified several issues including performance degradation and application crashes associated with Spectrum MPI and the Mellanox HCOLL library. The Spectrum MPI implementation contains its own software collective library that must be bypassed in order to use SHARP collectives. In one particular case we found that using 32 or more PPN at large scale resulted in consistent application segmentation faults. Working with our vendors we have mostly remedied these functionality and performance related issues.

Ethernet Network tests: During Summit's installation, we started noticing intermittent network performance issues. As the number of nodes increased, the problem became worse. Testing confirmed that traffic going from the compute nodes (1 GigE) to the management hosts (10/40 GigE) worked as expected. However, traffic from the management hosts to the compute nodes would intermittently hang and timeout. Packet captures confirmed that data from the management nodes was not reaching the compute nodes.

The Mellanox SX 1024 switches that were originally installed only had 4.6MB buffers. The 10GigE ports on these switches are allocated a maximum of 64KB [33].

This limit was generating a large amount of microbursts. Network microbursts Microbursts are difficult to detect and can result in dropped packets when buffers are exhausted. These can occur with network oversubscription, speed mismatches, unicast and broadcast flooding. These Ethernet switches did not have adequate buffers to handle microbursts and would frequently drop packets. Ultimately, they were replaced with switches that had larger buffers.

System software tests: Given that several components of the IBM HPC SW stack were brand new and developed specifically for the CORAL systems, close coordination and testing with IBM was needed to stabilize the products.

The batch workload manager and job launcher work together to allocate and limit compute resources and are key components to system management and usability. IBM's Spectrum Load Sharing Facility (LSF) provides workload management. The LSF development team worked closely with the OLCF to test and modify the existing LSF product to ensure functionality requirements were met. For example, as part of the OLCF leadership computing mandate, batch jobs requesting large portions of the system's compute resources are prioritized above jobs requesting smaller portions of the system's compute resources. The LSF development team was able to provide a method that, when combined with a submission script maintained by the OLCF, allows batch job limits and priorities to be placed on a batch job based on its requested amount of nodes. To help control batch queue throughput, the center limits the number of jobs each project member can have in a running state as well as an eligible to run state. Batch jobs are considered to be in a running state when LSF has allocated the requested compute resources. Batch jobs considered to be in an eligible to run state have not been allocated, but are eligible for allocation as resources become available. LSF developers created a limit named `ELIGIBLE_PEND_JOBS` that enforces a limit on the number of batch jobs eligible for execution. The center utilizes the new limit to control the number of eligible batch jobs each user may have per project at any given time. Similar functionality to limit the number of simultaneous running batch jobs already existed in LSF through the `JOBS` limit. Through testing, the center discovered that queued jobs over the set limit were not considered for allocation, but still gained priority for queue wait time. The center worked with LSF developers to add a configurable limit, `INELIGIBLE`, to the LSF product. When set, the flag prevents queued batch jobs over the `JOBS` limit from gaining wait priority.

The application launcher used on Summit is IBM's Job Step Manager (JSM). JSM provides the `jsrun` command which is the only mechanism provided to launch job steps on the compute nodes. During AT, we identified the need to specify specific and often irregular task and thread layouts on the compute nodes. In response to our feedback, JSM developers provided new functionality via the Extended Resource Format (ERF) [5] file that allows users to create a custom mapping file to specify task and thread placements. OLCF worked closely with developers to help design the map file's structure and functionality.

Compiler tests: Several compiler tool chains are available on Summit and were evaluated as part of acceptance: IBM XL C/C++ and Fortran, PGI, GNU, and clang. Testing for the various compilers focused on support for directives-based programming models for both CPU (OpenMP) and GPU offloading (OpenACC and OpenMP 4.5). OpenACC offloading was evaluated using the OpenACC 2.5 test suite [12, 24]. Support for OpenMP on the CPU was tested using the OpenMP 3.1 test suite [28, 32] while OpenMP GPU offloading was tested using a development snapshot of the OpenMP 4.5 offloading testing suite [30, 23]. Table 6 shows compiler support and test results for the various compilers available on Summit.

Table 6. Compiler directives test results

Compiler	OMP-CPU		OMP-Offloading		OpenACC	
	Support	Results	Support	Results	Support	Results
PGI-C/C++	Y	95.93%	N/A	-	Y	99.4%
PGI-Fortran	Y	88.54%	N/A	-	Y	
XL-C/C++	Y	94.31%	Y	86.00%	N/A	-
XL-Fortran	Y	86.46%	Y	N/A	N/A	-
Clang-C/C++	Y	96.58%	Y	94.00%	N/A	-
GNU-C/C++	Y	95.93%	N/A	-	N/A	-
GNU-Fortran	Y	88.54%	N/A	-	N/A	-

Testing used GA versions of the PGI and XL compilers at the time of testing. The Clang compiler is based on Clang version 3.8 with patches that have since been mostly ported to the mainline Clang public repository. For the GNU compiler, we used version 6.3. We tested CPU OpenMP directives using varying numbers of threads. The numbers reported in Table 6 represent results using 8 OpenMP threads. It should be noted that even though the XL Fortran compiler supports OpenMP offloading, no Fortran version of the test suite was available at the time, so this capability was not exercised. Also, OpenACC offloading in the GNU compiler is available in more recent versions than that used for acceptance testing. We have deployed a version of GCC with OpenACC offloading on Summit (using the `openacc-gcc-8` development branch [8]).

The Standard Performance Evaluation Corporation (SPEC) releases a variety of standardized benchmarks that are widely used to evaluate the performance of computer systems. For Summit acceptance two SPEC benchmark suites, SPEC OMP2012 and SPEC ACCEL, were used to evaluate the different compilers. SPEC OMP2012 measures the performance of OpenMP 3.1 applications using fourteen benchmark applications. The SPEC ACCEL suite measures the performance of accelerator-based systems using a set of computationally intensive applications, and supports OpenCL, OpenACC and OpenMP 4.5.

For the evaluation, "base" runs were produced following SPEC rules. All benchmarks were built using the compiler versions listed in Table 4 using common optimization flags. The benchmarks were run with both the test and train problem sizes, and three iterations of the reference problem sizes. All metrics presented in this section are measured estimates (i.e., SPEC scores normalized

to a baseline reference system measurement), and higher scores indicate better performance. Table 7 summarizes the results for the benchmarks. It is notable that the XL and PGI compilers are both successfully compiling the Spec OMP suite and their respective scores are close. The Spec ACCEL suite is passing compilation for the PGI compiler. The XL compiler failed compilation of one benchmark (551.ppalm). The Table includes data for a run of the benchmark in early February 2018. This shows a) the progress the compiler teams have made and b) the overall performance improvement of the software stack. It is notable, that the ACCEL benchmark shows significant variance between PGI and XL versions of the suite.

Table 7. Measured estimates for SPEC OMP and ACCEL using PGI and XL.

SPEC OMP			SPEC ACCEL					
	PGI	XL	PGI			XL		
Benchmark	**18.7**	**16.1.0**	Benchmark	**18.1***	**18.7**	Benchmark	**13.1.7***	**16.1.0**
350.md	4.65	3.82	303.ostencil	6.67	12.4	503.postencil	4.106	10.7
351.bwaves	7.25	0.69	304.olbm	10.2	12.7	504.polbm	4.53	5.80
352.nab	2.61	2.64	314.omriq	8.9	31.3	514.pomriq	4.65	17.4
357.bt331	6.82	8.09	350.md	13.6	25.7	550.pmd	CE	1.49
358.botsalgn	3.26	2.54	351.palm	2.84	3.06	551.ppalm	CE	CE
359.botsspar	1.41	1.32	352.ep	7.76	11.1	552.pep	0.856	1.29
360.ilbdc	3.15	3.43	353.clvrleaf	7.98	12.0	553.pclvrleaf	CE	18.3
362.fma3d	3.49	3.48	354.cg	7.9	12.9	554.pcg	2.35	7.13
363.swim	7.45	7.57	355.seismic	5.29	13.1	555.pseismic	2.11	6.07
367.imagick	5.42	4.40	356.sp	6.74	11.7	556.psp	CE	22.1
370.mgrid	5.11	5.51	357.csp	RE	13.5	557.pcsp	RE	8.82
371.applu	8.77	7.77	359.miniGhost	6.85	9.79	559.pmniGhost	CE	2.64
372.smithwa	6.22	3.00	360.ilbdc	6.5	11.3	560.pilbdc	1.08	21.0
376.kdtree	1.61	1.94	363.swim	4.1	5.87	563.pswim	CE	3.54
			370.bt	RE	16.1	570.pbt	RE	6.70

*results from February 2018
CE Compile error
RE Runtime Error

Math Libraries: Two test codes were used to evaluate vendor and third party libraries of critical importance to applications at the OLCF (for further details see [19]). First, the ScaLAPACK test was used to validate functionality and correctness of the ScaLAPACK library for a variety of compiler choices and underlying BLAS dense linear algebra libraries. Second, the nuccor_kernels test evaluates a large number of combinations of compilers (XL, PGI, GNU, LLVM), parallelism models (OpenMP, OpenACC, CUDA), and BLAS libraries (ESSL variants, MAGMA, LAPACK, CUBLAS). The main focus of these tests was functionality and correctness; performance was primarily evaluated by other tests such as the CORAL applications.

I/O Libraries: Although I/O libraries are implicitly tested using applications and miniapps, reports of possible performance issues from CAAR development

teams prompted us to include two I/O-specific miniapps in the AT. FLASH I/O [7] recreates the primary data structure of the full application FLASH [6] and measures only the I/O portion. Chimera I/O mimics the I/O subroutines of Chimera [21] by using parallel HDF5 to collectively write output files from multiple processes, with each process writing the sub-domain it owns. Using both miniapps, we identified very slow performance for parallel HDF5 with a single-shared file. A subsequent Spectrum MPI update included a newer version of ROMIO that, coupled with necessary tuning hints, improved parallel HDF5 performance significantly.

OLCF Application tests: After reviewing the set of applications that were actively used on Titan, we selected applications that represented a diverse set of algorithmic patterns, programming models, programming languages, and math and I/O libraries. The application set used in [19] was augmented to provide full coverage for these requirements (see Table 2).

Debugger and Profiler tests: Arm DDT is the primary parallel debugger used in production at OLCF. The FT criteria for DDT includes starting the debugger at 20% of full-system scale within five minutes and performing basic debugging operations, such as setting breakpoints at various source locations and inspecting local variables. For this test we used DDT's `offline` capability that permits non-interactive debugging in batch jobs, with output captured in a log file. GenASiS [22] was used as the target application, using several test cases and varied PPN. Early test attempts failed to start DDT consistently at scale within a reasonable amount of time, due to what appeared to be a hang. Subsequent updates to the software stack resolved this problem. The DDT test also uncovered an issue with CUDA debugging which caused breaking at kernel launches to take many seconds per kernel. As DDT in offline mode automatically breaks at all kernel launches by default, applications with many kernel launches appeared non-responsive. We disabled this behavior while the issue is being resolved.

The nvprof tests exercise the profiler in a variety of operational modes, using both single-host and multi-host applications. We verified the ability of nvprof to generate traces, profiles, and analysis metrics for regular CUDA programs, as well as applications using OpenMP and OpenACC directives. Additionally, we evaluated support for profiling MPS and MPI applications. By engaging NVIDIA, we identified unsupported modes that were not properly documented (i.e., limitations for "application replay" mode), as well as a bug that produced incorrect profiling results when running multiple nvprof instances concurrently on a node. A fix for this bug is included in the CUDA 10.1 release.

5.3 Performance Test (PT)

To ensure that the system was able to meet contractual performance from the CORAL Benchmarks as well as deliver adequate performance for real-world applications, several tests were executed in isolation. The subset of tests included

all the CORAL Scalable Science and Throughput Benchmarks and the applications shown in Table 2 selected from the OLCF portfolio.

As part of PT, the scalable and throughput CORAL benchmarks were executed to measure their individual figures of merit (FOMs). For the scalable benchmarks, each code was executed on a quiet system at near full system scale. Each throughput benchmark requires 192 nodes. In order to measure the throughput FOMs, 22 job steps of the same benchmark were executed simultaneously to fill up the system. Table 8 summarizes the results obtained when executed on a quiet system.

Table 8. CORAL Scalable and Throughput benchmarks FOMs obtained on Summit.

CORAL Benchmark		RFP Baseline FOM	Measured FOM	Speedup
Scalable	LSMS	3.39E+00	3.01E+01	8.88
	QBOX	5.31E+09	3.65E+10	6.87
	HACC	1.06E+09	1.01E+10	9.53
	Nekbone	1.58E+09	1.10E+10	6.96
Throughput	CAM-SE	4.44E-01	1.12E+00	2.52
	UMT	2.58E+11	5.87E+11	2.28
	AMG	4.50E+10	3.15E+11	7.00
	MCB	3.23E+10	4.56E+11	14.12
	QMCPACK	2.29E+05	1.86E+06	8.12
	NAMD	1.55E+00	8.01E+00	5.17
	LULESH	1.12E+07	2.89E+08	25.80
	SNAP	2.21E+02	9.76E+02	4.42

NAMD is a classical molecular dynamics application and part of the CORAL Throughput benchmarks. NAMD is both compute and communication intensive with random memory access. For acceptance testing, multiple copies of 192-node NAMD jobs were run concurrently to fill up the system. Each run was required to meet the contractual performance target.

We encountered several issues with NAMD during the acceptance period. Because it is communication and memory intensive, NAMD's performance is sensitive to process placement on the node. We also identified a bug in CHARM++'s PAMI back-end which is used on Summit[2]. This bug caused a race condition manifested by occasional hangs on application exit. When the scheduler eventually killed the job due to its time limit, stray processes were often left on the node that degraded the node's performance until they were cleaned up. NAMD was able to meet its performance target following the resolution of these issues.

While verifying the CORAL benchmarks, we observed Nekbone test performance using 4,560 nodes was 48-71% slower than expected. A single node screen was developed to narrow down the cause, which helped identify specific nodes in this degraded state. The root cause was found to be a firmware bug that was leaving the PCIe bus improperly trained [18], resulting in degraded bandwidth. This issue was addressed in the December 2018 HPC SW stack update.

5.4 Stability Test (ST)

In order to simulate a realistic workload on the full system, we used the OLCF Test Harness [31]. The harness submits the full set of tests developed for acceptance as individual jobs in the batch queue. The code or application used for a given test is built before submission to simulate code development activities. The harness records whether a given test instance builds successfully, is submitted to the batch queue successfully, and is executed successfully.

The ST started on Oct. 25, 2018 and was executed continuously for 2 weeks (336 hours). Over 29,000 individual tests were executed out of which 97.77% completed successfully. Fig. 3 shows a distribution of the types of failures encountered during ST. Intermittent performance failures were observed in NAMD, Chroma, RayTrace, CAM-SE, nvlink, and stream tests. For some of the applications, the performance failures exceeded the acceptable runtime variability criteria. In addition, 339 build failures occurred because the temporary license used with the IBM XL compiler expired. The IBM applications team determined that the two failures marked as unknown at the conclusion of ST were caused by a race condition in LULESH. In addition to job failures, all ST hardware failures were closely monitored. As shown in Fig. 4, we encountered a hardware failure that terminated a job every 28 hours on average. For this reason, we set the maximum allowed walltime on Summit to be 24 hours.

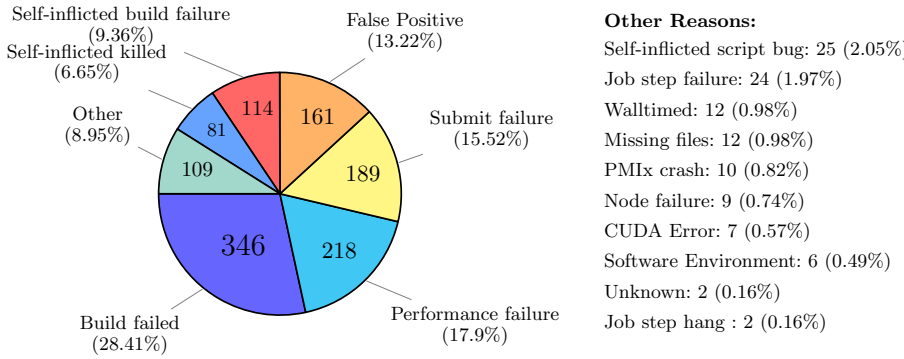

Fig. 3. Failure distribution for SP2 ST (336-hour period).

6 Lessons Learned

With each successive leadership class system the OLCF has launched, the number of tests in the acceptance test suite has continued to rise. This is inevitable insofar as node and system complexity and heterogeneity are continually increasing, requiring more testing.

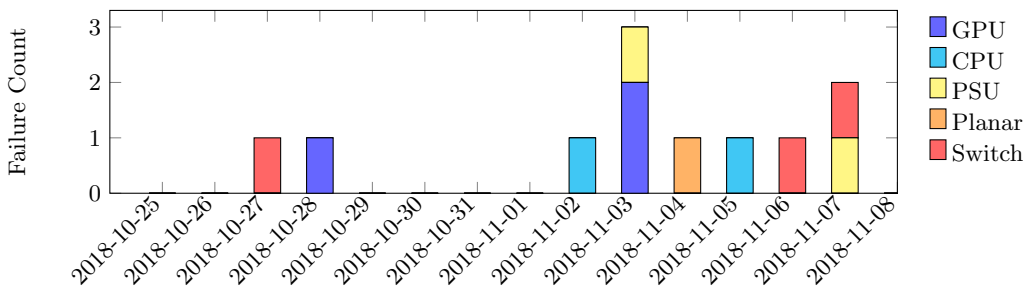

Fig. 4. Number of catastrophic job failures due to a CPU, GPU, power supply unit (PSU), motherboard (Planar), or switch issue for SP2 ST (336-hour period).

Deploying Summit involved switching to a different vendor for the OLCF's top computing system as well as using brand new components for parts of the system software stack, e.g., IBM's JSM, CSM, and Spectrum MPI. Close interactions with the vendor starting early in the project were key to the successful deployment of features required by OLCF users.

When selecting tests for a new system, it is essential to include codes that check for exact bit-for-bit correctness. Two of such tests, Minisweep and CoMet, allowed us to identify and replace two defective GPUs. Both tests were incorporated to the "every node" tests set for this reason.

The results of AT show that in some cases, when the system is fully loaded, runtime variability can exceed the desired threshold. This was alleviated by dedicating one core on each POWER9 processor to system services. While isolating additional cores per socket could further reduce variability and increase I/O bandwidth, this will result in reduced throughput for applications. To effectively use the GPUs, applications require as many cores as available.

Testing revealed that releasing the GPFS daemon from the isolated cores can provide up to ∼22 GB/s of I/O bandwidth. In cases where applications are I/O limited, it can be beneficial to use the locally-developed `maximizegpfs` LSF option.

Benchmarks and applications are both necessary to provide full coverage for acceptance testing. One is not a substitute for the other. In the case of parallel HDF5 and MPI-IO libraries, although the file system acceptance used benchmarks that utilized these libraries under the hood, performance issues were not discovered until we ran an application test. This is due to the fact that benchmarks' access patterns are not equivalent to those of real applications. The former is likely written to get the best performance the system is capable of providing. The latter is written to fulfill the need of the application.

When designing a network, the nature of all of the applications, and directions of traffic flows and their speeds must be well understood and inform the hardware selection. Vendors should include detailed information on buffer sizes and how buffers are allocated to ports in their data sheets. We should stress test any new

network gear for congested paths, large amounts of broadcast traffic, and mixed speeds. This should be done throughout the system's deployment. We should validate network monitoring features in any new gear to verify if it is capable of alerting and monitoring microbursts, latency, buffer usage, quality of service and any dropped data.

7 Science on Summit

All thirteen Center for Accelerated Application Readiness (CAAR) applications selected for Summit successfully achieved their performance targets.

The Summit Early Science Program generated tremendous interest, with 65 letters of intent, 48 full proposals and 33 early access awards. Notably, 12 of the intent letters featured a machine learning component, reflecting high interest in using Summit's machine learning capabilities for scientific discovery.

Of the five 2018 ACM Gordon Bell finalists using Summit, all projects used mixed precision, four projects used the Volta GPU Tensor Cores, and four projects used some form of machine learning. In particular, the CoMet computational genomics application achieved 2.36 ExaOps of mixed precision performance out of the peak achievable Summit performance of 3.2 ExaOps. This was the world's first application to break the ExaOp barrier, with real-world applications such as finding genetic causes of diseases like opioid addiction [26]. The Tensor Cores were also successfully used to develop a half-precision version of the DeepLabv3+ neural network used to detect extreme weather patterns. DeepLabv3+ was able to achieve 1.13 ExaOps of peak performance [27] on Summit.

Summit was put in production in January 2019, and currently supports 30 projects for the DOE Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program [4], as well as other projects delivering new science outcomes.

8 Conclusions

Deploying a system of Summit's scale requires close collaboration between the vendor and the center. OLCF and IBM worked together to identify and fix issues encountered during Summit's installation and acceptance testing. Thanks to this collaboration, we were able to provide direct feedback to IBM development teams in the early stages of deployment.

The OLCF designs a thorough AT plan to help determine if the system is ready for production workloads. As presented here, several issues that impacted functionality and performance of applications were addressed. The issues found and lessons learned from executing the AT have been translated into documentation and examples [15, 16]. This information will be helpful for users starting to run on Summit.

Summit, with 4,608 compute nodes and 27,648 V100 GPUs, is currently ranked first in the Top500 [17] and the HPCG list [1]. Its top-three ranking in the

Green500 list [9] also reflects its energy efficiency. In its short lifetime, Summit has already proven to be a prolific scientific instrument. Projects working on climate analytics [27] and bioinformatics [26] have leveraged Summit's mixed-precision Tensor Cores to break the ExaOp barrier.

Acknowledgement

The authors would like to thank the entire IBM CORAL team for their efforts towards a successful deployment and acceptance of the Summit system.

This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725.

References

1. 10th HPCG Performance List, https://www.hpcg-benchmark.org/custom/index.html?lid=154&slid=298
2. Charm++ Bug # 1988, https://charm.cs.uiuc.edu/redmine/issues/1988
3. CORAL Benchmark Codes, https://asc.llnl.gov/CORAL-benchmarks
4. DOE Leadership Computing: INCITE, http://www.doeleadershipcomputing.org/
5. Extended resource file format, https://www.ibm.com/support/knowledgecenter/en/SSWRJV_10.1.0/jsm/10.2/base/erf_format.html
6. Flash center for comptational sciecne, http://flash.uchicago.edu/site/flashcode/
7. Flash i/o benchmark routine – parallel hdf 5, http://www.ucolick.org/ zingale/flash_benchmark_io/
8. GCC8 openacc development branch. https://github.com/gcc-mirror/gcc/tree/openacc-gcc-8-branch
9. Green 500, https://www.top500.org/green500
10. Intel MPI Benchmarks User Guide, https://software.intel.com/en-us/imb-user-guide
11. mpiGraph, https://github.com/LLNL/mpiGraph
12. OpenACC Test Suite. https://github.com/OpenACCUserGroup/OpenACCV-V
13. STREAM, for lots of devices written in many programming models, https://github.com/UoB-HPC/BabelStream
14. STREAM: Sustainable Memory Bandwidth in High Performance Computers, https://www.cs.virginia.edu/stream
15. Summit: Scale new heights. discover new solutions., https://www.olcf.ornl.gov/summit/
16. Summit System User Guide: Known Issues, https://www.olcf.ornl.gov/for-users/system-user-guides/summit/summit-user-guide/#known-issues
17. Top 500, https://top500.org
18. TS001460803: "slow nodes" observed with nekbone tests
19. Experiences Evaluating Functionality and Performance of IBM POWER8+ Systems (October 2017)
20. Bland, A.S., Joubert, W., Kendall, R.A., Kothe, D.B., Rogers, J.H., Shipman, G.M.: Jaguar: The world's most powerful computer system–an update. Cray Users Group (2010)

21. Bruenn, S.W., Blondin, J.M., Hix, W.R., Lentz, E.J., Messer, O.E.B., Mezzacappa, A., Endeve, E., Harris, J.A., Marronetti, P., Budiardja, R.D., Chertkow, M.A., Lee, C.T.: Chimera: A massively parallel code for core-collapse supernova simulation. arXiv e-prints arXiv:1809.05608 (Sep 2018)
22. Cardall, C.Y., Budiardja, R.D.: Genasis basics: Object-oriented utilitarian functionality for large-scale physics simulations (version 2). Computer Physics Communications **214**, 247 – 248 (2017). https://doi.org/https://doi.org/10.1016/j.cpc.2016.12.019, http://www.sciencedirect.com/science/article/pii/S0010465517300097
23. Diaz, J.M., Pophale, S., Hernandez, O., Bernholdt, D.E., Chandrasekaran, S.: Openmp 4.5 validation and verification suite for device offload. In: de Supinski, B.R., Valero-Lara, P., Martorell, X., Mateo Bellido, S., Labarta, J. (eds.) Evolving OpenMP for Evolving Architectures. pp. 82–95. Springer International Publishing, Cham (2018)
24. Friedline, K., Chandrasekaran, S., Lopez, M.G., Hernandez, O.: Openacc 2.5 validation testsuite targeting multiple architectures. In: Kunkel, J.M., Yokota, R., Taufer, M., Shalf, J. (eds.) High Performance Computing. pp. 557–575. Springer International Publishing, Cham (2017)
25. Joubert, W., Archibald, R.K., Berrill, M.A., Brown, W.M., Eisenbach, M., Grout, R., Larkin, J., Levesque, J., Messer, B., Norman, M.R., et al.: Accelerated application development: The ORNL Titan experience. Computers and Electrical Engineering **46** (May 2015). https://doi.org/10.1016/j.compeleceng.2015.04.008
26. Joubert, W., Weighill, D., Kainer, D., Climer, S., Justice, A., Fagnan, K., Jacobson, D.: Attacking the opioid epidemic: Determining the epistatic and pleiotropic genetic architectures for chronic pain and opioid addiction. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis. pp. 57:1–57:14. SC '18, IEEE Press, Piscataway, NJ, USA (2018), http://dl.acm.org/citation.cfm?id=3291656.3291732
27. Kurth, T., Treichler, S., Romero, J., Mudigonda, M., Luehr, N., Phillips, E., Mahesh, A., Matheson, M., Deslippe, J., Fatica, M., et al.: Exascale deep learning for climate analytics. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis. p. 51. IEEE Press (2018)
28. OpenMP Validation and Verification Suite: Openmp validation suite. https://github.com/sunitachandra/omp-validation
29. Petitet, A., Whaley, R.C., Dongarra, J., Cleary, A.: High performance linpack (2018), https://www.netlib.org/benchmark/hpl
30. Pophale, S., Diaz, J.M., Hernandez, O., Bernholdt, D., Chandrasekaran, S.: OpenMP 4.5 Validation and Verification Suite for Device Offload. https://crpl.cis.udel.edu/ompvvsollve/
31. Tharrington, A.N.: Nccs regression test harness, version 00 (9 2015), https://www.osti.gov//servlets/purl/1232564
32. Wang, C., Chandrasekaran, S., Chapman, B.: An openmp 3.1 validation testsuite. In: Chapman, B.M., Massaioli, F., S., M.M., Rorro, M. (eds.) OpenMP in a Heterogeneous World. pp. 237–249. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
33. Warner, J.: Mellanox switches, https://people.ucsc.edu/ warner/Bufs/mellanox