Acceptance Testing the Chicoma HPE-Cray EX Supercomputer

Everson[Ψ], Ferrell, Green, Lapid, Magee, Ogas, Seamons, Sly
High Performance Computing Environments
Los Alamos National Laboratory
Los Alamos, USA
{pferrell,jgreen,lapid,dmagee,jogas,calvindseamons,sly}@lanl.gov, kody.everson@trojans.dsu.edu

***Abstract*—Since the installation of MANIAC I in 1952, Los Alamos National Laboratory (LANL) has been at the forefront of addressing global crises using state-of-the-art computational resources to accelerate scientific innovation and discovery. This generation faces a new crisis in the global COVID-19 pandemic that continues to damage economies, health, and wellbeing; LANL is supplying high-performance computing (HPC) resources to contribute to the recovery from the impacts of this virus.**

Each system that LANL's HPC Division installs requires an understanding of the intended workloads for the system, the specifications and expectations of performance and reliability for supporting the science, and a testing plan to ensure that the final installation has met those requirements. Chicoma, named for a mountain peak close to Los Alamos, NM, USA, is a new HPC system at LANL purchased for supporting the Department of Energy's Office of Science Advanced Scientific Computing Research (ASCR) program. It is intended to serve as a platform to supply molecular dynamics simulation computing cycles for epidemiological modeling, bioinformatics, and chromosome/RNA simulations as part of the 2020 Coronavirus Aid, Relief, and Economic Security (CARES) Act. Chicoma is among the earliest installations of the HPE Cray EX supercomputer running the HPE-Cray Programming Environment (CPE) for Shasta Architecture.

Like other cutting-edge HPC systems, Chicoma is being installed in phases- beginning with the testbed (Chicoma) installation, followed by the delivery, installation, and integration of the final system. A fundamental part of the preparation of the system focuses on verifying that it can support the science for which it was purchased. To ensure a supercomputer's viability for real customer workloads, LANL HPC support staff plans, prepares, and conducts extensive testing.

The Programming and Runtime Environments Team (PRETeam) of LANL's High Performance Computing Environments Group, supports HPC environments in production and facilitates the integration and acceptance testing of new systems.

The Chicoma HPE-Cray EX system is evaluated by the PRETeam over its phases of installation to ensure that the resulting product meets the requirements as set forth by the statement of work for the procurement. By generating substantive test results, we empirically demonstrate its ability to perform the intended scientific workloads. The testing plan developed for the system includes functional, performance, and application tests selected to quantify the ability of the system to meet performance, reliability, and precision requirements specifically for supporting the research efforts of the COVID-19 HPC Computing Consortium.

The Pavilion2 HPC Testing Harness automates the execution of tests on HPC systems. Improvements to Pavilion2 are leveraged in the Chicoma test configurations to enable a simplified and reproducible benchmarking and synthetic workload. Developing and harnessing the tests under Pavilion2 ensures controlled consistency of the application's build and execution environments. Pavilion2's test definitions express rules to generate scripts to interface the environment modulefiles, filesystem, scheduler, and system resources to build, schedule, run, collect, and log results for each test. The Splunk data analytics platform and programmed dashboards automatically graph results as they're fed into the system. This simplifies the process of analyzing and deriving statistics among tests of similar configurations, reducing time spent post-processing results.

This paper documents the acceptance testing process executed by LANL HPC Environments Group's PRETeam staff to ensure that the Chicoma system meets the requirements of the funding program and is capable of sustaining a predetermined workload of synthetic and real scientific applications. It describes the Chicoma acceptance test suite, configurations and tuning of the tests under Pavilion2, presents the results of acceptance testing, and concludes with a discussion of the outcomes of the acceptance testing effort on Chicoma and future work.

Keywords—benchmarking, supercomputing, test-harness, performance testing, HPC

[Ψ] Dakota State University Advanced Research Laboratory

This research used resources provided by the Los Alamos National Laboratory Institutional Computing Program, which is supported by the U.S. Department of Energy National Nuclear Security Administration under Contract No. 89233218CNA000001. LA-UR-21-26200

I. The Chicoma Supercomputer

The Chicoma Supercomputer is an HPE-Cray liquid-cooled system residing within the Vendor Liaison Enclave network at LANL. There are currently 512 Slingshot interconnected dual socket AMD EPYC™ compute nodes, each running 128 hardware threads with hyper-threading enabled to facilitate 256 threads per node with 8 NUMA local domains per node [6]. There's 515.4 GB physical memory per node, with L1, L2, and L3 caches. Jobs on the system are scheduled using SchedMD's Slurm Workload Manager, v19.05.5.

A. Processor Information

The Cray Windom motherboard supplies two AMD EPYC™ 7H12 AuthenticAMD Family 23 Model 49 Stepping 0 Processors with a Base Frequency of 2.6 GHz, but supports 3.3GHz with a max boost capability when running in Boost Mode. The tests were conducted with Boost Mode enabled on each compute node of the system.

B. Memory Information

Eight memory channels supply 204.8 GB/s memory per socket DDR4 memory bandwidth. The NUMA configuration of the node is 2 sockets, each supporting 4 local memory domains for a total of 8 NUMA nodes per compute node. Each NUMA domain supplies 63 GB shared memory among 16 cores with 2 Processing Units each.

TABLE 1: CHICOMA SYSTEM SPECIFICATIONS

System Information	
Operating System	Linux4.12.14-197.34_9.1.22-cray_shasta_c x86_64
Model	Cray Inc. Windom
Motherboard	Cray Windom
Memory	257712 MB
BIOS (confirm)	Cray Inc. 0.8.3-SBIOS-1049-vm1
Processor Information	
Name	AMD EPYC™ 7H12
Topology	2 Processors, 128 Cores, 256 Threads
Identifier	AuthenticAMD Family 23 Model 49 Stepping 0
Base Frequency	2.6 GHz
L1 Inst Cache	32.0 KB x 64
L1 Data Cache	32.0 KB x 64
L2 Cache	512 KB x 64
L3 Cache	8.0 MB x 16

C. Cache Information

Each core on the AMD EPYC™ processor hosts 32KB L1 instruction cache, another 32 KB L1 data cache, and has a 512 KB L2 cache. The L3 cache size is 16 MB shared among 4 cores, for a total of 64 MB L3 cache per NUMA node. Thread and memory affinities were considered in the test configurations to ensure optimal computational efficiency of threaded applications.

D. Interconnect Information

Chicoma's compute nodes are interconnected with HPE Slingshot technology designed to support a mix of parallel workloads including simulation, analytics, and AI on heterogeneous architectures [7]. Its backbone supplies high bandwidth with 64 ports operating at 200 Gb/s. The HPE Slingshot high radix 64 port switch couples with the Dragonfly topology to ensure a maximum of 3-hop efficiency among over 250,000 endpoints. Smart switching via Cray's adaptive routing of the Slingshot devices with congestion control capabilities promises low latency and scalability of the network and therefore greater parallel efficiency of data intensive HPC.

E. Filesystem Information

Our initial tests of the Chicoma system were limited by the lack of a parallel file system appliance to handle tests and applications that perform large parallel IO operations. Future tests for this system will include more realistic IO workloads when an appliance is supplied to handle them. Tests to evaluate the IO capabilities of the system include IOR, QCD, and VPIC utilizing checkpointing/restarting running very large problems. The tests were conducted to run, write, and read from a 1 TB NFS appliance, therefore the tests were configured to account for this limitation in the early standup of Chicoma.

F. Programming Environment

The default HPE-Cray Programming Environment (CPE) for the Shasta Architecture supplied by the vendor is used in the compilation, linking, and execution of tests in the acceptance test suite. The intent is to assess the system performance and characteristics based upon the delivered system. The supplied software stack is a vital part of system delivery and requires careful evaluation. We acknowledge that some routines could be optimized if care is taken to build out separate software installations, however, the intent of the acceptance testing is to ensure that the delivered product, e.g., hardware and default software stack, is capable of sustaining a representative workload.

The initial installation of the CPE on Chicoma supplies the Cray Compiler (Clang), GNU Compiler Collection (GCC), and the AMD Optimizing Clang Compiler (AOCC) Cray Programming Environments, each of which are utilized by one or more tests. During our early user period on the system, upon request, we supplied the Intel Compiler (Intel) Cray Programming Environment, however, this wasn't incorporated into our acceptance tests. More extensive CPE testing could be achieved by testing all the components of the Cray Developer Toolkit. Due to time constraints, we intend to explore this more thoroughly in the future.

Additionally, we note that important container mechanisms, e.g., Linux "user" and "mount" namespaces, required by unprivileged user container implementations, like Charliecloud[1], are able to be run in the context of a single-node user job on the system. This is important for some customers who use or wish to explore containerized workflows.

[1] https://hpc.github.io/charliecloud/

We note that the usability of the User Access Instances (UAIs), while functional, is inadequate for conducting tests as there are current limitations preventing access to the compute nodes via a secure shell from the containers. Nor is there a native Slurm environment required for some functional tests to execute. We launch our test harness from user-access nodes (UANs), due to these limitations.

Once we receive the alpha-version of the CPE Container from HPE-Cray, we will start exploring its viability on the Chicoma system with Charliecloud and supply feedback to the vendor to ensure its readiness for our customers wanting the versatility of a containerized application along with the benefit of the Cray PE. Those explorations will include assessments of performance with Pavilion2 containerized tests against bare metal performance and the pliability of the Cray CPE container to reduce the containers' footprint to achieve a minimal streamlined environment of runtime dependencies. This work, however, is outside the scope of the acceptance testing project.

II. Chicoma Acceptance Test Plan

A. Testing Approach

Acceptance testing is an integral part of the stand-up for LANL HPC clusters. The tests used in procurements often consist of micro-benchmarks, standard HPC benchmarks, and applications from the intended users of the cluster. Acceptance testing includes operational, performance, and stability testing. The third consists of system level tests and real application codes running for an extended period of time while measuring both the stability of the hardware and consistency of performance. This phase of testing is beneficial for: 1) deriving baselines for the HPC subsystems; 2) identifying any problems; and 3) measuring the ability of the system to sustain performance over time. Tuning and configuration optimizations to overcome any weaknesses are engineered during this phase of system evaluation so the resulting production system, once integrated and productionized, is hardened to support the demands of simulation science workloads. Once the desired level of performance is confirmed, the system is "accepted" and the cluster moves on to the final phases of system integration, and ultimately, production mode of operation.

The set of benchmarks and applications used during acceptance testing supplies performance baselines that will be used to gauge the continued performance of the system as it transitions from delivery and configuration to integration and production operational modes. Continuous testing with the acceptance test configurations throughout these phases of the system's life-cycle ensures that our production testing procedures identify performance regressions that may occur with changed configurations, upgrades, and hardware failures.

Our selected tests for targeting the performance and functional capabilities of Chicoma take into account the limitations from the early standup phase of the system's life cycle. We target tests that exercised subsystems integral for the computational requirements of the IC Program's target applications, emphasizing proxy applications for molecular dynamics (MD) simulations and analytic workloads. We curated the following set of tests (Tab. 2), for acceptance testing the system, and configured their build, job, and runtime characteristics to ensure adequate evaluation of the system.

TABLE 2: CHICOMA ACCEPTANCE TEST SUITE

Test Names	Description
DGEMM	DGEMM is a FORTRAN dense matrix multiplication solver routine from the Linear Algebra PACKage (LAPACK)
ExaMiniMD	A proxy application for MD particle codes implemented to utilize various parallel models
GROMACS	An MD package designed for simulations of proteins, lipids, and nucleic acids running a COVID-19 research problem from LANL
hello_mpi	MPI Hello world MPI-C functional test
HPCG	The sparse matrix calculations performed in the benchmark measure the limitations of the memory subsystem and interconnect
HPL	The test plan includes running three types of High Performance Linpack (HPL) workloads: Single Node, 8 Node Linpack, and Full system
MILC7	The MILC Code is a body of high performance research software written in C for doing SU(3) lattice gauge theory
Quicksilver	Quicksilver is an MPI/MPI-OMP proxy application included in the CTS Benchmarks Suite that solves a simplified dynamic Monte Carlo particle transport problem
Stream	Tests the performance of the memory subsystem by performing four simple vector kernel calculations
SystemConfidence	SystemConfidence is a benchmark that measures statistical variation of network latencies
VPIC	VPIC is a simulation code for modeling kinetic plasmas on one, two, or three dimensions. It employs a second-order, explicit, leapfrog algorithm to update charged particle positions and velocities in order to solve relativistic kinetic equations
LULESH	Solves a simple Sedov blast problem with analytic answers

B. Test integration and Functional Testing

The Cray Shasta Management interface system is a brand-new product-line that is undergoing rapid vendor development concurrent with standup and acceptance efforts, so the partnership between customer and vendor in this procurement is unique. Multiple HPC centers partner in the Compass organization with HPE-Cray to deliver a production-worthy product with a brand-new interface, software stack, and hardware. Along with the vendor, site engineers are integrating lacking components in the Shasta System whereas the PE teams are engaging to evolve the software delivery mechanism to better support customer needs and container runtime requirements.

The effort to tailor the test parameters and build configurations for a new architecture such as Chicoma naturally couples with basic evaluations of the usability of the CPE due to the unique circumstances surrounding the state of the Shasta system software. Several weeks of test integration and trial runs helped to pinpoint obvious weaknesses and shortcomings, which then were resolved by the integration team and vendor during

this phase of testing. The test integration and early functional testing took around 6 weeks to complete in preparation for conducting acceptance testing.

The Shasta 1.4.1 upgrade is complete on Chicoma, and presents more integration effort to stabilize the system for its return to early users. So, whereas the acceptance runs prove valuable to ensure the product delivery meets the contractual requirements, re-running the acceptance test suite is underway. Using the initial baselines collected for the tests will ensure that performance, functionality and reliability is, at minimum, sustained from Shasta 1.2 to 1.4.1. Preliminary testing uncovered instability in the high-speed network (HSN), therefore, results from a ready system are not available at the time of this writing.

C. The Pavilion2 HPC Test Harness

Pavilion2[2] is a Python3 system designed to accommodate various testing requirements of HPC data centers. It is maintained by LANL's High Performance Computing Environments Group and is open-sourced for community contributions. Pavilion2 supplies a framework for creating sophisticated YAML configurations to automate the workflow of running jobs on HPC systems. Using test definitions and host specific configurations for Chicoma, the execution of the tests on the system are greatly simplified for ensuring repeatability of the tests, results collection and analysis. Pavilion parses output results of every test and logs in a json file format, which then is able to be processed by a number of analysis utilities.

D. Splunk Pavilion Results Analysis

LANL's High Performance Computing data centers are monitored by a vast monitoring infrastructure, collecting data and logs from every possible component to ensure optimal visibility and alerting on events that could lead to trouble in HPC environments. The data centers are staffed 24/7 by an operations team trained to intercept potential issues and alert subspecialty teams when detected. The monitoring infrastructure is supported by a distributed Splunk instance on every network, gathering large temporal data for analysis. The Pavilion2 test harness generates and logs test data that are indexed in Splunk, enabling charting, reporting, and baselining systems. The results presented are products of the Pavilion2 Dashboard developed in Splunk.

III. Test Results

A. DGEMM

DGEMM is a FORTRAN dense matrix multiplication solver routine from the Linear Algebra PACKage (LAPACK) designed to measure the sustained floating-point computational rate of a single node.

Our DGEMM test iterates over problem sizes 2500, 5000, 10000, 20000, and 40000 MB, utilizing 128 OpenMP threads per node. This test is conducted in full system sized batch jobs, using *srun* to launch backgrounded applications in a loop, and wait for their completion.

The test currently utilizes the BLAS routines supplied by Cray's LibSci library. There is a limitation (we suspect in the configuration/build of BLAS with max threads) preventing the threads to span the test across both sockets on the node, and as such, the results reflect less than optimal efficiency. Needless to say, the test is beneficial in identifying nodes exhibiting drastic performance deviation from the average and median baseline measurements (Fig. 2.), with a problem size of 40,000 MB.

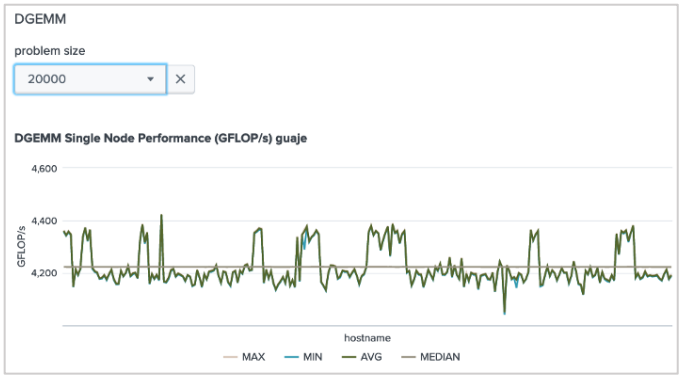

Fig. 1. DGEMM Performance (problem size 20000)

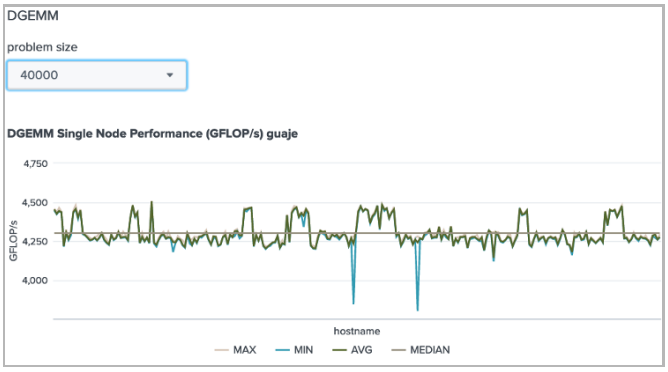

Fig. 2. DGEMM Performance (problem size 40000)

B. HPL

High Performance Linpack [5] solves a (random) dense linear system in double precision (64 bit) arithmetic on distributed-memory computers. The test plan includes running three types of HPL workloads: Single Node, 8 Node Linpacks, and Full system. This test is beneficial as it can demonstrate the floating-point operational potential of a system, reported in the results as GFLOP/s, and be used as a diagnostic to identify underperforming nodes on an HPC system.

2 Pavilion2 is a complete redesign of Pavilion by the HPC PRETeam at LANL
https://github.com/hpc/pavilion2

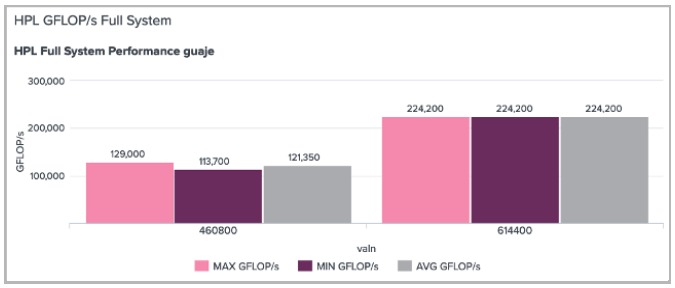

Fig. 3. HPL Full-system performance (256 nodes)

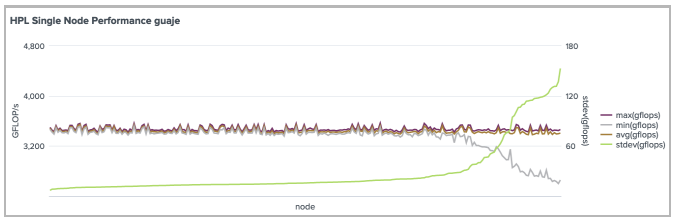

Fig. 4. HPL single-node performance (256 nodes)

C. HPCG

The High-Performance Conjugate Gradient (HPCG) [4] benchmark was developed to serve as an alternative to HPL to better represent a real application, exercising data access patterns common to parallel applications. The sparse matrix calculations performed in the benchmark measure the limitations of the memory subsystem and interconnect.

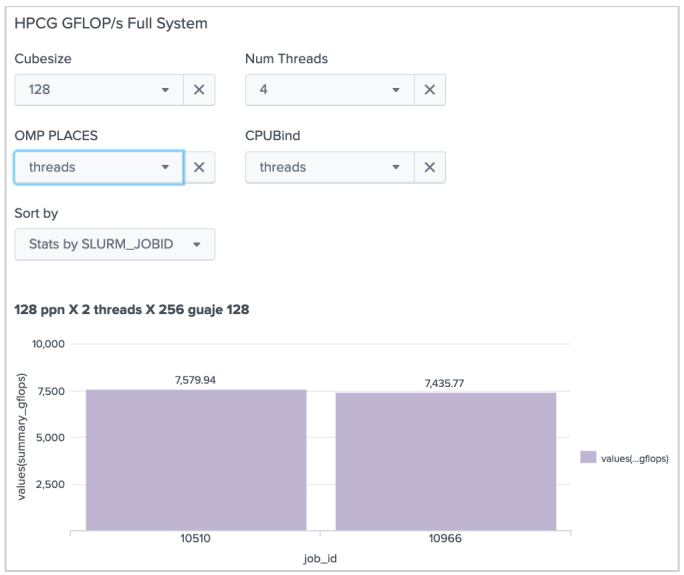

Fig. 5. HPCG Performance Dashboard

D. STREAM

Stream is a simple synthetic benchmark that tests the performance of the memory subsystem by executing four simple vector kernel calculations and reporting the memory rate achieved with each [13]. Stream is tuned to be appropriately sized for the system upon which it's running. A simple calculation in our tests ensures the problem size is appropriate for the memory resources on Chicoma.

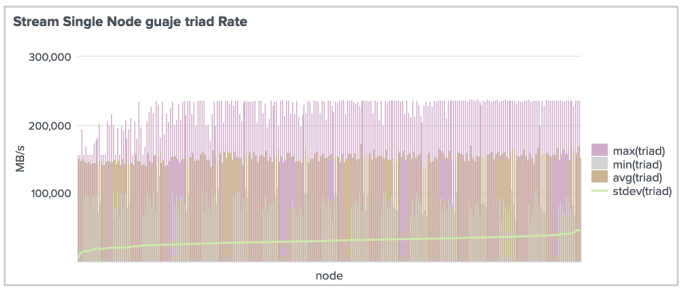

Fig. 6. STREAM Single Node Triad Performance (Array Size 99115652)

E. IMB

Intel MPI Benchmarks (IMB) [14] conducts scaling studies of the MPI-1 specifications utilizing the Cray Message Passing Toolkit (MPT). These routines are common in most HPC applications and serve as good indicators of the inter- and intra-nodal communication performance of a supercomputer.

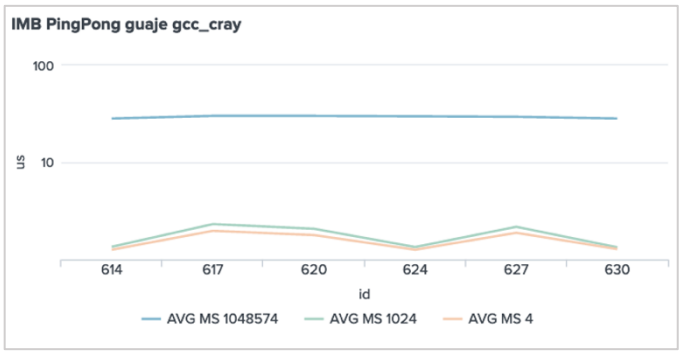

Fig. 7. IMB PingPong Performance

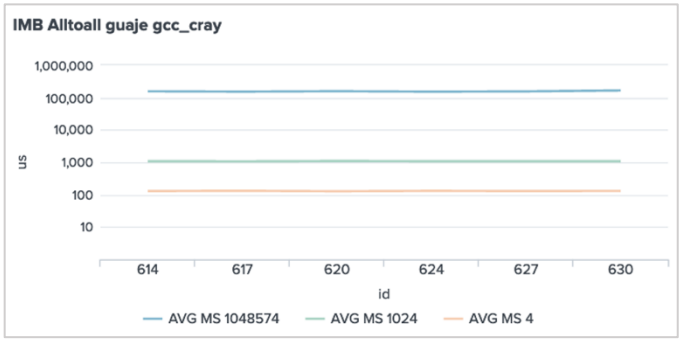

Fig. 8. IMB AlltoAll Performance

F. Quicksilver

Quicksilver [3] is an MPI/MPI-OMP proxy application included in the CTS Benchmarks Suite that solves a simplified dynamic Monte Carlo particle transport problem. Its performance is bound by poor vectorization potential, latency bound table look-ups and a heavily branching or divergent code path.

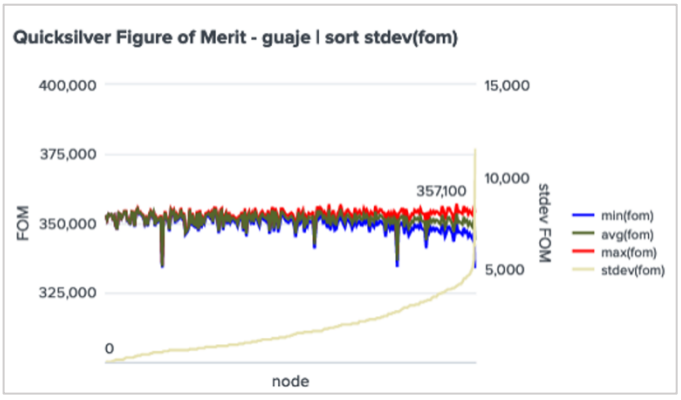

Fig. 9. Quicksilver Single Node Performance

G. ExaMiniMD

ExaMiniMD is a proxy application for Molecular Dynamics (MD) particle codes implemented to utilize various parallel models. The implementation we chose for Chicoma's acceptance suite uses the Kokkos programming model and the supplied input deck. The test plan includes scaling the test to use 8, 16, 32, and 64 tasks per node, keeping the number of Kokkos threads-per-task consistent for the 16, 32, and 64 tasks versions. Important data to collect include total time for each of the 256,000 particles to calculate force, create neighbor lists, and communicate.

Further enhancements to this test would be to explore improved inputs generated from the LAMMPS Molecular Dynamics Simulator.

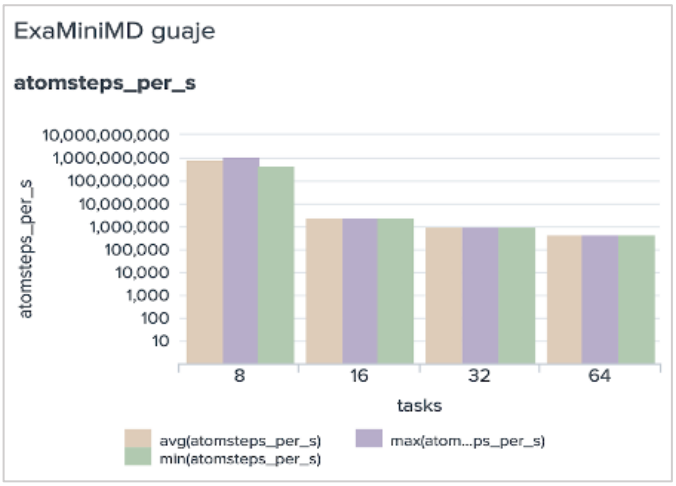

Fig. 10. ExaMiniMD AtomSteps/s

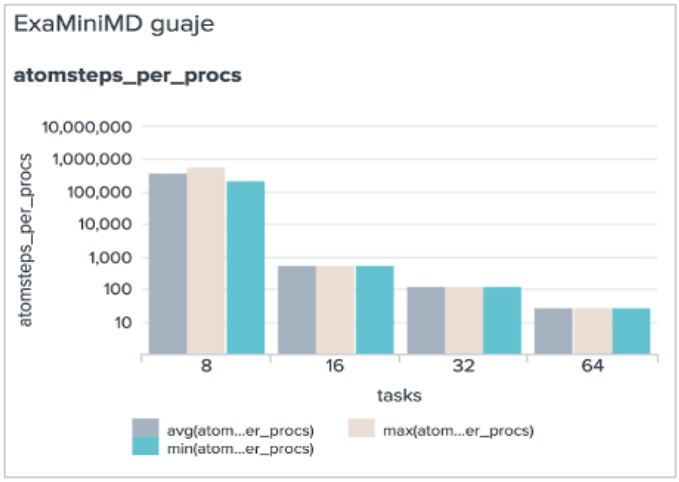

Fig. 11. ExaMiniMD AtomSteps/process

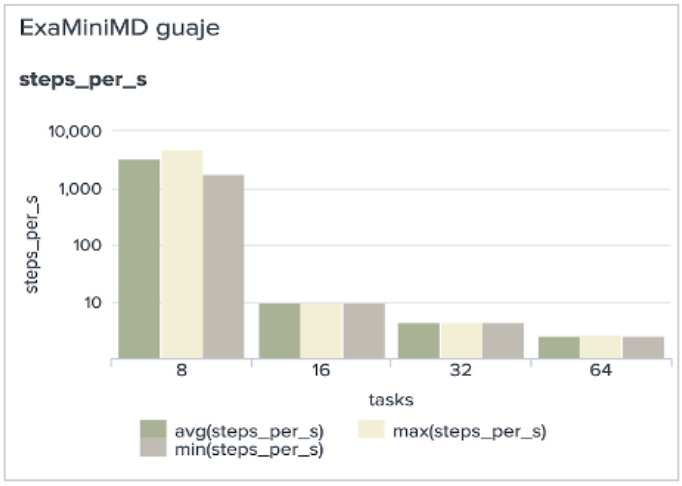

Fig. 12. ExaMiniMD Steps/second

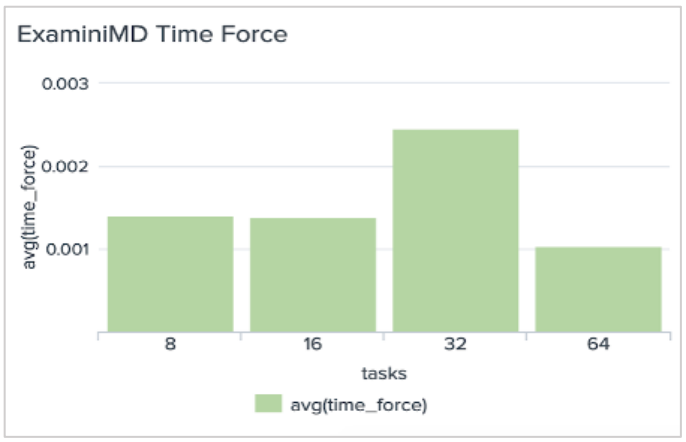

Fig. 13. ExaMiniMD AVG Time Force

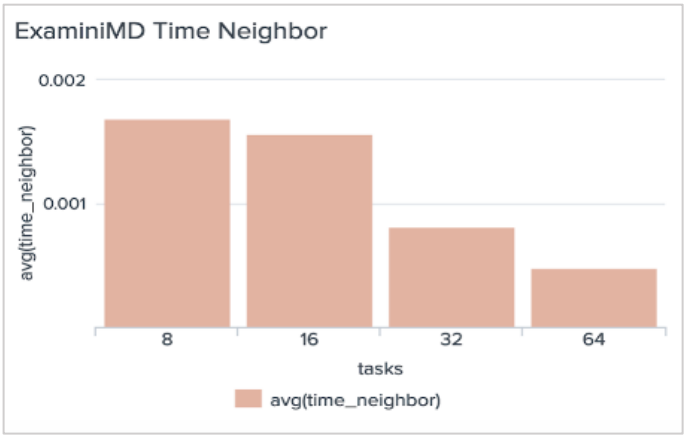

Fig. 14. ExaMiniMD AVG Time Neighbor

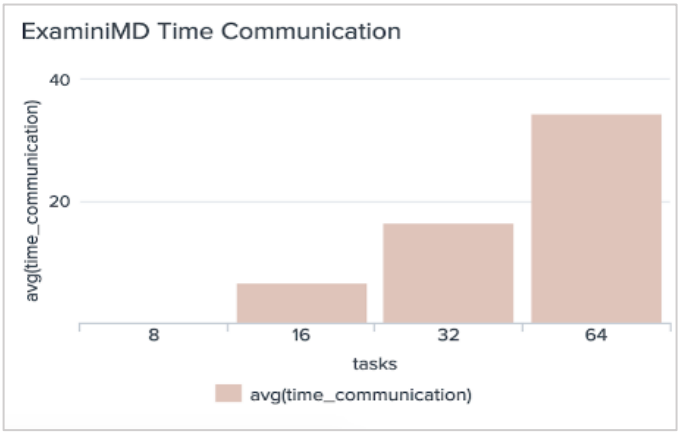

Fig. 15. ExaMiniMD AVG Time Communication

H. Hello MPI

Hello MPI is a simple MPI-C code that evaluates the ability to compile, link, and run an MPI-C code against all available compilers and MPI libraries on the system. It's a functional test that swiftly identifies broken MPI installations or environment module files on a supercomputer. It supplies a pass/fail result viewable in Splunk as well as Pavilion2's command line results interface, so we are able to detect when problems arise.

I. VPIC (Lyin-Sequoia Problem)

Vector Particle-In-Cell[3] application is a simulation code for modeling kinetic plasmas on one, two, or three dimensions. It employs a second-order, explicit, leapfrog algorithm to update charged particle positions and velocities in order to solve relativistic kinetic equations. The input deck, a modified version of lyin_sequoia[4], executes the problem that Lawrence Livermore National Laboratory used to evaluate their Sequoia system's potential to model the interaction of realistic fast-ignition-scale lasers with dense plasmas in three dimensions with sufficient speed to explore a large parameter space and optimize the design for ignition [9].

We ran this problem in a series of configurations that could be compared to both XC40 Haswell (HSW) and Knight's Landing (KNL) partitions. Thus, the upper limit boundary of test parameters was determined by the lowest common number of available resources, which in this case was set by the XC40 HSW compute nodes. The problem was run multiple times on 256 nodes at 1, 2, 4, 8, 16, and 32 processes-per-node. We intentionally avoided hyperthreading due to the large difference in available ranks between hardware configurations; we wanted to compare the simulation completion time of each system against a physical number of cores.

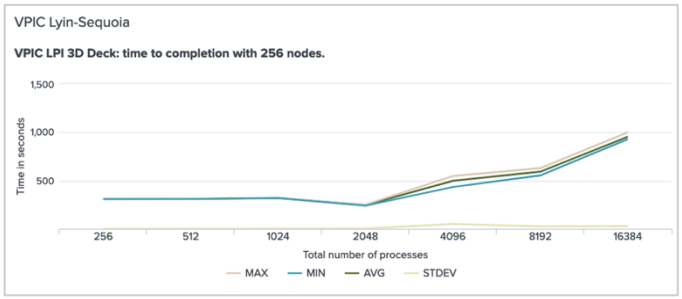

Fig. 16. VPIC Runtime by Processor Count

J. Livermore Unstructured Lagrangian Explicit Shock Hydrodynamics (LULESH)

LULESH [8] is a highly simplified application, hard-coded to only solve a simple Sedov blast problem with analytic answers – but represents the numerical algorithms, data motion, and programming style typical in scientific C or C++ based applications.

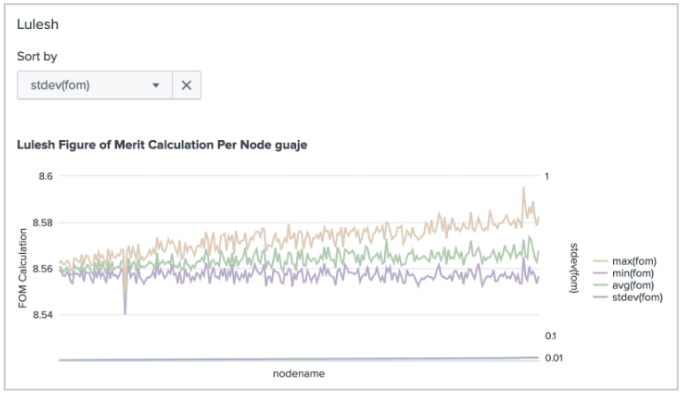

Fig. 17. Lulesh FOM/node

K. MILC7

The MILC [2] code is a body of high performance research software written in C for doing SU(3) lattice gauge theory on high performance computers as well as single-processor workstations. A wide variety of applications are included. The problem we run in our test suite is the basic improved Kogut Susskind molecular dynamics code.

3 https://github.com/lanl/vpic

4 The original problem used over one-million time-steps; we significantly reduced the size of the problem to conduct tests in the span of hours vs. weeks.

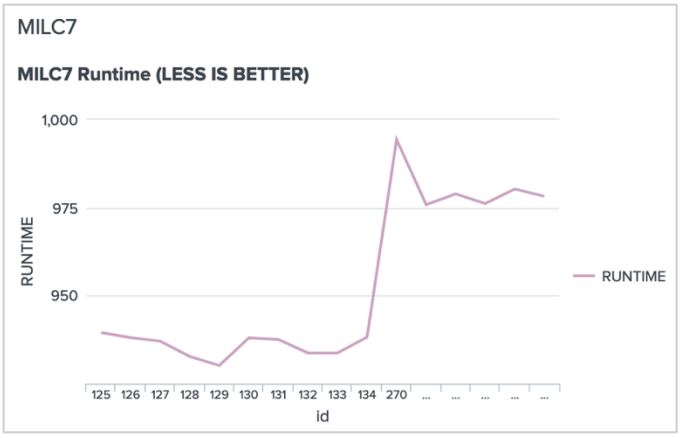

Fig. 18. MILC7 Runtime by testid

L. GROMACS

GROMACS [1] is primarily designed to simulate the behavior of biochemical molecules such as proteins, lipids, and nucleic acids that have a lot of complicated bonded interactions. However, since GROMACS is extremely fast at calculating the nonbonded interactions (that usually dominate simulations), many groups are also using it for research on non-biological systems, e.g., polymers.

The problem we run in our suite is one supplied by a COVID-19 researcher at Los Alamos National Laboratory [10].

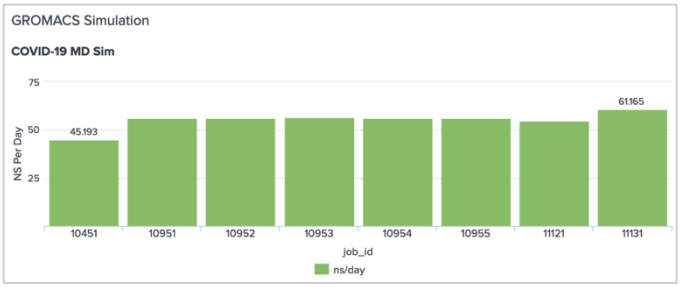

Fig. 19. GROMACS COVID-19 Research Problem NS/day

M. SystemConfidence

SystemConfidence (Josh Lothian, 2012) was used to measure interconnect latency on Chicoma. Full system SystemConfidence runs flood the interconnect and measures the latency of pairwise communication on the system.

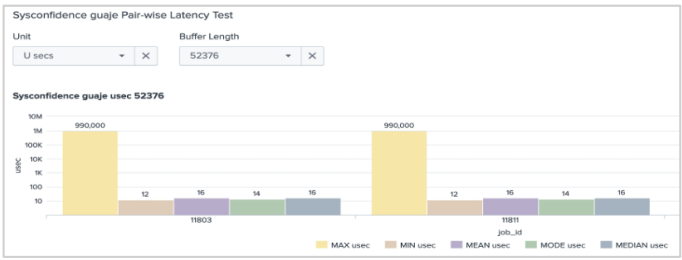

Fig. 20. SystemConfidence Pairwise Latency

IV. Outcomes, Conclusions, and Future Work

A. Outcomes

The procurement, installation, configuration, integration, and acceptance testing of supercomputers is a highly coordinated effort that is only achieved with the expertise of various teams. Centers are required to ensure that the system adheres to the requirements as agreed to within the statement of work between the vendor and site. Furthermore, in initial assessment, we are presented with the technical challenge of building, tuning, and executing applications on an unfamiliar system- one that requires debugging and adjustments to facilitate real usage. The process of establishing, configuring, automating, and executing a fully functional test suite on a new system feeds into the staff knowledge, subsequently building in-house expertise in order to better assist real users once the machine is moved to a production state. This test integration effort works out gaps in configuration, bugs in system configurations and software, and ensures the viability of the sub-systems' functionality before running the more advanced phases of testing for acceptance.

Further challenges are presented with the integration of a cutting-edge system, such as the HPE-Cray Ex system running Shasta with early hardware and redesigned system-software in tow. Ushering vendor-supplied solutions still under early development into a production environment is an on-going and open-ended effort that continues to be refined and hardened along the way. The Chicoma procurement aims to supply a COVID-19 research platform, but the additional benefit to the laboratory is the systems teams' development of an in-depth familiarity and a strong vendor-HPC Center feedback loop. Due to the collaborative partnership among LANL and HPE-Cray engineers, there are continued improvements being made to the new HPE-Cray Shasta system stack. The experience gained by the teams will carry over for future procurements and facilitate improved support for anticipated HPE-Cray systems for the DOE NNSA laboratories.

B. Conclusions

The Chicoma system testing revealed consistency in performance in the tests selected, and remarkable stability of the system to sustain full-scale, mock workloads comparable to that which our researchers will be running on the machine. Identified problems were uncovered by the tests as we evaluated the system and prepared for the final acceptance. These findings aided in the discovery and remedies to issues surrounding the network, hardware, and sub-system software components. Ultimately, we were able to conduct 48+ hours of non-interrupted runtime with the Chicoma acceptance testing suite, comprising synthetic, proxy and real applications, proving that the system would be stable, performant, and could reliably conduct the molecular dynamics computational workload.

C. Future Work

Though the initial testing has been completed, and the system formally accepted, continued effort ensues by the Systems Engineering Team at LANL to fully integrate the system into production. Barriers to this next phase have been identified, and as those obstacles are removed, the system will continue to undergo updates and adjustments, while sustaining real workloads by researchers. Ultimately, the system will reside in a production network, but until it can be transitioned there, we anticipate continued updates and adjustments that will require more testing to ensure the system maintains stability, reliability, and performance expectations. The knowledge gained from the efforts of the teams working on Chicoma will feed into future acceptance testing and systems engineering activities as we acquire and integrate future systems into production. The support cycle of a supercomputer entails continued testing throughout its useful life, and the baselines and tests that were used in its acceptance will continue to be used to measure performance against initial measured baselines to identify hardware or software regressions or degradations.

ACKNOWLEDGMENT

Our team would like to acknowledge the hard work of the System Engineering team at LANL who've worked tirelessly integrating this system and preparing it to host workloads. The HPE-Cray Shasta innovation presented a unique challenge to the system engineering effort, but opened opportunities for better vendor-customer collaboration. Positive outcomes of the early implementation of the Shasta software on a site installation include forged partnerships among the vendor and customers to deliver a management infrastructure that would replace what was already well known. Such technical collaboration ensures better communication and more rapid problem solving. The sheer fact that we were able to run a suite of tests against this system is heavily due to their innovation and perseverance in piecing together incomplete beta system software and making it work.

[1] M. J. Abraham, T. Murtola, R. Schulz, S. Páll, J. C. Smith, B. Hess, E. Lindahl, GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers, SoftwareX, Volumes 1–2, 19-25. 2015.

[2] G. Bauer, S. Gottlieb and T. Hoefler, "Performance modeling and comparative analysis of the milc lattice qcd application su3 rmd", *2012 12th IEEE/ACM International Symposium on Cluster Cloud and Grid Computing (ccgrid 2012)*, pp. 652-659, 2012.

[3] "Co-design at lawrence livermore national lab: Quicksilver", *Lawrence Livermore National Laboratory (LLNL)*, 07 2017, [online] Available: https://codesign.llnl.gov/quicksilver.php.

[4] J. Dongarra, M.A. Heroux and P. Luszczek, "High-performance conjugate-gradient benchmark: A new metric for ranking high-performance computing systems," *The International Journal of High Performance Computing Applications*, Vol. 30(1) 3–10. 2016.

[5] J. Dongarra. The linpack benchmark: An explanation. In 1st International Conference on Supercomputing, pages 456–474, June 1987.

[6] HPE Cray EX Supercomputer QuickSpecs, https://h20195.www2.hpe.com/v2/GetDocument.aspx?docname=A00094635ENW

[7] HPE The Next Generation of Exascale Computing, https://www.hpe.com/us/en/compute/hpc/supercomputing/Cray-exascale-supercomputer.html

[8] I. Karlin, J. Keasler, R. Neely, LULESH 2.0 Updates and Changes, Lawrence Livermore National Laboratory. http://codesign.llnl.gov/lulesh

[9] D.S. Lemons, D. Winske, W. Daughton, B. Albright, Small-angle Coulomb collision model for particle-in-cell simulations, *Journal of Computational Physics*, Volume 228, Issue 5, Pages 1391-1403. 2009.

[10] "Los Alamos National Laboratory brings next-generation HPC to the fight against COVID-19," Los Alamos National Laboratory, Los Alamos, NM, USA, October 2020. [Online]. Available: https://www.lanl.gov/discover/news-release-archive/2020/October/1020-hpc-to-fight-against-covid19.php. Accessed June 30, 2020.

[11] Martinasso, M, Gila, M, Sawyer, W, Sarmiento, R, Peretti-Pezzi, G, Karakasis, V. Cray programming environments within containers on Cray XC systems. *Concurrency Computat Pract Exper*. 2020; 32:e5543. https://doi.org/10.1002/cpe.5543

[12] R.A. Mansbach, S. Chakraborty, K. Nguyen, D. C. Montefiori, B. Korber, S. Gnanakaran, The SARS-CoV-2 Spike variant D614G favors an open conformational state, *Science Advances.* Apr 2021.

[13] J. D. McCalpin, "STREAM Benchmark." Available: http://www.cs.virginia.edu/stream.

[14] S. Saini, R. Ciotti, B. T. N. Gunney, T. E. Spelce, A. E. Koniges, D. Dossa, P. A. Adamidis, R. Rabenseifner, S. R. Tiyyagura, M. Mller, and R. Fatoohi. Performance evaluation of supercomputers using hpcc and imb benchmarks. In 20th International Parallel and Distributed Processing Symposium (IPDPS 2006), Proceedings, 25-29 April 2006, Rhodes Island, Greece. IEEE, 2006.