Early Experiences with Trinity - The First Advanced Technology Platform for the ASC Program

C.T. Vaughan, D.C. Dinge, P.T. Lin, K.H. Pierson, S.D. Hammond, J. Cook,
C.R. Trott, A.M. Agelastos, D.M. Pase, R.E. Benner, M. Rajan and R.J. Hoekstra

Abstract—Trinity, the first in a new generation of Advanced Technology supercomputing systems (ATS) for the Department of Energy's National Nuclear Security Administration, will enter production service in late 2016. We present and discuss initial application performance analysis and scaling results for several key SIERRA production codes - as well as microbenchmarks - that have been ported to Trinity development machines. By means of comparison, we contrast the performance of Trinity to the previous capability and capacity computing resources available to analysts at Sandia. Our findings show that our codes will benefit from a doubling in per-MPI rank performance and a second doubling in terms of MPI rank density per node. The result is that the first phase of Trinity delivers a combined 4X improvement in capability computing to an important class of problem in the NNSA/ASC stockpile stewardship program.

Index Terms—Trinity, Performance, SIERRA, Engineering, HPC, Cray, XC40

1 Introduction

In late 2015 Los Alamos National Laboratory and Sandia National Laboratories took delivery of *Trinity*, the first Advanced Technology System (ATS) deployment by the Department of Energy's National Nuclear Security Administration (NNSA). As the first ATS platform, Trinity, a Cray XC40, marks the start in a new generation of computers for the NNSA laboratories that are more aggressively adopting new processor technologies and system designs to prepare production application codes for future Exascale environments. In the first phase of deployment, approximately 10,000 nodes of 16-core, dual-socket Intel Haswell E-class server processors are being installed, with a second phase, due within 2016, that will feature self-hosted Intel Xeon Phi Knights Landing (KNL) nodes.

As with any new deployment, there is considerable interest from production code groups as to the performance of the new system on 'day one' of its use. For Trinity, there are many changes over the existing computing environment (the Cielo Cray XE6 machine which has been the NNSA labs' capability computing environment since 2011) that are expected to provide significant improvements in application runtime including: (1) a new processor vendor (Intel instead of AMD); (2) a doubling of cores per processor socket; (3) considerably enhanced instruction set architecture (ISA) with wider and more capable SIMD vector units; (4) improvement in memory bandwidth through the use of DDR4 memory; and, (5), a more capable and higher performing network interconnect (Cray Aries instead of Cray Gemini).

In this paper we present initial application performance and scaling results for several key production codes that have recently been ported to Trinity development and testing machines prior to the full machine entering production service, building on preparation activities shown in [1]. These codes, which are part of the SIERRA family of engineering applications that are heavily used by stockpile stewardship programs at Sandia, comprise many millions of lines of source written in C, C++ and Fortran and commonly utilize up to 50 third-party libraries. Porting these codes to new machines is therefore nontrivial due to the complexity associated with ensuring all of the dependencies can be successfully compiled on a new architecture. As a basis for comparison, we also present several performance results from the predecessor production computer Cielo, and a heavily used capacity cluster from Appro (now owned by Cray) called Chama.

The contributions of this work are several:

- **Microbenchmark Comparison** - we present microbenchmark comparisons of performance primitives for a Cray XE6, a commodity InfiniBand cluster and a Cray XC40 for MPI point-to-point operations, reduction collectives and memory bandwidth.
- **SIERRA Application Performance Analysis** - we present performance comparisons of the three important engineering analysis applications from the Sandia SIERRA Solid Mechanics, Structural Dynamics and Aero suite. These codes represent production computing within Sandia's stockpile stewardship program and, as such, heavily exercise hardware capabilities.
- **Analysis and Discussion** - Finally, we present discussion and analysis of our initial experiences and performance results from Trinity. In so doing, we show that many of our applications achieve a 2X in direct runtime-to-runtime comparison versus Cielo (the previous capability production resource), as well as an additional 2X improvement in density. The

result is that on a node-to-node evaluation, our applications are able to utilize a total of 4X performance and density improvement reflecting a significant enhancement to our scientific capability.

The remainder of this paper is laid out as follows: in Section 2 we describe the machines evaluted in this paper including the Phase-I (Haswell) partition of Trinity which is the focus of this work; Section 3 presents microbenchmark results of STREAM (memory bandwidth) and base MPI operations on each of these platforms; an analysis of application behavior on the platforms can be found in Section 4. Finally, Sections 5 and 6 present some thoughts on future activities as we expand our work on Trinity and begin to focus on the Phase-II Knights Landing deployment and then conclude our paper with a summary of the results presented.

2 Benchmarking Platforms

In this section of the paper we briefly contrast the supercomputing resources used in our performance studies: (1) the NNSA/ASC Cielo platform; (2) the NNSA/ASC Chama capacity cluster and, finally, (3) the NNSA/ASC Trinity machine. Table 1 provides a summary comparison of basic hardware parameters.

2.1 NNSA/ASC Cielo

Deployed during 2010, the current capability computing platform for the NNSA/ASC Tri-Lab community is Cielo [2], [3], a 1.37 PFLOP/s system using Cray'x XE6 architecture. The machine consists of approximately 8,950 dual-socket, oct-core AMD Magny-Cours compute nodes interconnected by Cray's Gemini network. Each socket is partitioned into two NUMA domains with each housing four cores and 8 GB of system memory (for a node total of 16 cores and 32GB). The Gemini network interconnect [4] used in Cielo, is a three-dimensional torus with some Y-dimension routing being able to take advantage of faster inter-router data movement.

The major design goals for Cielo were the ability to perform several large-scale three-dimensional multiphysics simulations at roughly the 80 - 100TB scale. While job sizes and scale have varied considerably over the life of the machine due to variation in scientific campaigns being granted time on the machines, routine jobs often scale to 32,000 cores or beyond. Once the Phase-I installation and acceptance of Trinity (described later in this paper) is fully completed Cielo will be retired during the summer of 2016.

2.2 NNSA/ASC Chama

Chama [5] is an instantiation of the successful Tri-Lab Capacity Cluster (TLCC-2) procurement in which the NNSA/ASC Tri-Labs collaboratively procure numerous installations of smaller capacity machines in order to support small to medium sized jobs across the stockpile stewardship complex. All machines procured under the TLCC program have identical baseline hardware configurations to reduce cost and improve portability between machines. A TLCC-2 node comprises two sockets of 8-core Intel Sandy Bridge E-class server processors running at 2.60GHz. Standard memory configurations include 32GB of system memory per core although larger configurations are provided on specific (non-standard) nodes or machines (up to 256GB) for specialized functions such as meshing or analysis processing. The TLCC-2 platforms are interconnected by a QLogic QDR-InfiniBand Fat-Tree. Although dual-way hyper-threading is available on Chama, this is regularly disabled for jobs executing on the machine as it rarely provides performance improvements (all runs presented in this paper utilize a single thread per core).

2.3 NNSA/ASC Trinity

The NNSA/ASC Trinity platform [6] is a collaborative supercomputer deployment between the Los Alamos National Laboratory and Sandia National Laboratories under the ACES (Alliance for Computing at Extreme Scale) partnership. The deployment of the machine is split into two phases. The first phase, which is already well under way, will see approximately 9,500 nodes of dual-socket, sixteen-core Haswell E5-server class processors and 128GB of system memory per node installed (see Table 1 for a basic hardware comparison to Chama and Cielo). The second phase, due in 2016, will be one of the first installations in the world to utilize the self-hosted Knights Landing many-core processor by Intel.

For the purposes of this study we have performed much of the benchmarking activities on the smaller, but hardware identical, Mutrino test machine located at Sandia National Laboratories. Mutrino is a single rack of the Trinity Haswell compute blades provided for initial application porting and performance analysis while the main Trinity machine installation is stabilized. As similar to Chama, all the runs performed on Mutrino are running using a single thread context per core (*i.e.* with hyper-threading disabled).

3 Microbenchmark Performance Analysis

Almost all new hardware delivered to the NNSA laboratories first undergoes initial microbenchmark performance analysis to ensure basic hardware functionality. Instinctively, the microbenchmark results act as a bounding-box for the performance our codes are likely to see with memory bandwidth benchmarks being the most likely of these to gain interest. In this short section we provide primitive memory bandwidth and MPI performance information across our benchmark machines as a sentinel for how we may expect our larger applications to perform.

3.1 STREAM

The STREAM benchmark [7] has become the *de-facto* assessment of memory sub-system performance as seen from executing code. In total, four benchmark kernels are executed that represent primitives found in basic scientific algorithm sequences. In general, the benchmarked Triad values are of most interest to scientific codes since the vast majority of calculation requires two operands and then the write back of the result to memory. Although STREAM-Triad represents somewhat of an upper bound versus more complex codes (as described later in this paper), we continue

TABLE 1
Architecture Parameters for Cielo, Chama and Trinity

System	Cielo	Chama	Trinity (Phase-I)
Total Nodes	8,894	1,232	9,408
Total Cores	142,304	19,712	301,056
Processor	AMD Magny-Cours	Intel Sandy Bridge	Intel Haswell
Processor ISA	SSE4a	AVX	AVX-2
Clock Speed (GHz)	2.40	2.60	2.30
Cores/Socket	8 (2x4)	8	16
Cores/Node	16	16	32
Peak Node (GFLOP/s)	153.6	332.8	1,177.6
Memory	DDR3-1333	DDR3-1600	DDR4-2133
Memory Channels/Socket	4	4	4
Interconnect	Cray Gemini	QLogic QDR-InfiniBand	Cray Aries
Interconnect Topology	3D-Torus	Fat-Tree	DragonFly

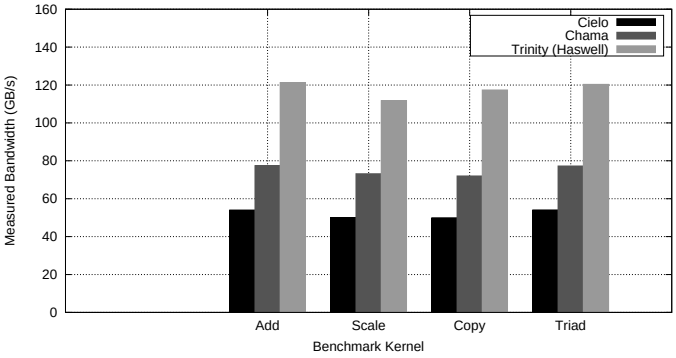

Fig. 1. Benchmarked STREAM Memory Bandwidth on Cielo, Chama and Trinity Compute Nodes

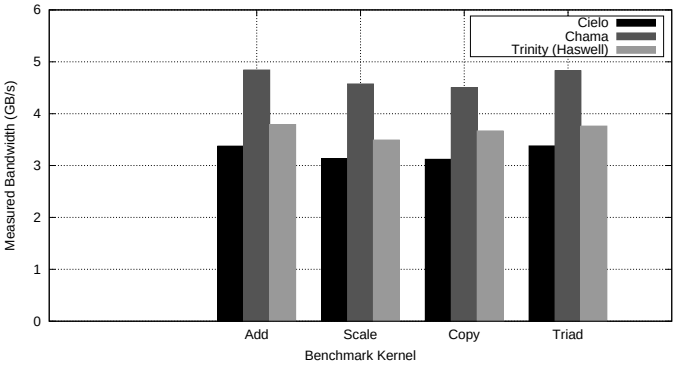

Fig. 2. Benchmarked STREAM Memory Bandwidth Per Core on Cielo, Chama and Trinity Compute Nodes

to use benchmark values from STREAM as a crude measure of how likely future systems are to meet the goals of the NNSA/ASC application portfolio.

In Figure 1 we present benchmarked values for the various kernels of STREAM running on the three platforms. We note that the Trinity platform exceeds the expected performance increase from memory clock speed increases (1333MHz in Cielo to 2133MHz in Trinity should give a 1.6X improvement but benchmarked values are greater than 2X). Since a significant number of applications within the NNSA/ASC program heavily track memory bandwidth this provides the first insight into the expected improvement that our wider application portfolio may expect to see on the new platform.

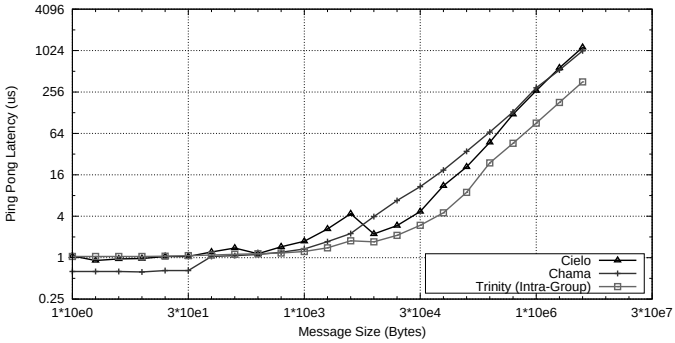

Fig. 3. Benchmarked MPI Ping Pong Message Latency on Cielo, Chama and Trinity

When considered from the per-core perspective however, memory bandwidth has actually decreased when migrating application codes from the Chama TLCC-2 platform to Trinity. Figure 2 shows the benchmarked STREAM values now divided by the core count on each compute node. We note that there is only a small improvement for Trinity when moving from Cielo and an approximate 20% decrease for Trinity over Chama. For application codes that utilize a fixed number of MPI ranks due to meshing or code constraints this may indicate that performance on the new platform will likely be similar if the code in question is, indeed, purely memory bandwidth constrained.

3.2 MPI Point-to-Point Operations

Figures 3 and 4 present benchmarked MPI ping-pong latency and bandwidth respectively. For these measurements we used the Intel MPI Benchmark (IMB). We note that in all cases the performance improvement of the Cray Aries interconnect is evident with close to 4X higher performance at large message sizes while still providing lower latency for very short messages (those less than 32 bytes). The application portfolio at Sandia communicates a very wide range of message sizes depending on the executing algorithm or particular class of physics which is being executed. Within a single application the mixture of physics and numerical algorithms means that it is not uncommon to have message sizes range from a single double to tens and hundreds of kilobytes. At the larger end, hydrodynamics codes within

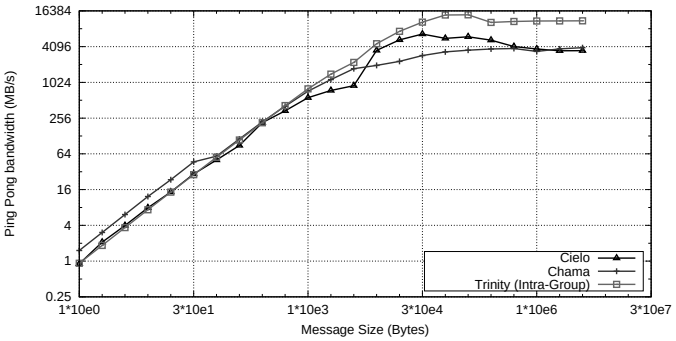

Fig. 4. Benchmarked MPI Ping Pong Bandwidth on Cielo, Chama and Trinity

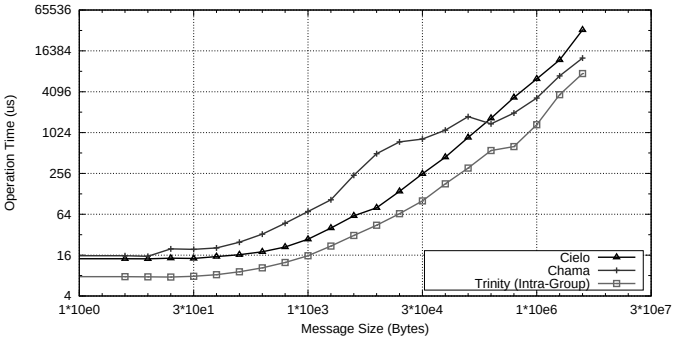

Fig. 5. Benchmarked MPI 256-Rank Allreduce Operations on Cielo, Chama and Trinity

the application suite can exchange message sizes into the megabytes. We present results for Trinity only within a single DragonFly group so far which is likely to have lower latency than cross-group communication. When larger installations permit we will revise the benchmarked values in a subsequent publication. However, for the purposes of this study, this limited focus provides insight into the potential performance improvements that may be seen in codes with frequent communication.

3.3 MPI Collective Operations

As with many scientific codes, the SIERRA application suite from Sandia makes frequent use of parallel reductions. In Figure 5 we show benchmarked results for this class of operation, again taken from the IMB benchmark. At all scales reduction on the Aries interconnect out performs Cielo and Chama, in some cases provided a performance improvement of up to 4X.

4 Application and Benchmark Codes

SIERRA [8] is an object-oriented computational framework and application suite that is currently in production use and development at Sandia National Laboratories. The SIERRA framework (which is shared across our engineering application suite) is a set of software services and parallel data structures upon which many mechanics applications can be written. The main goal is to bring together distributed mesh management, field management (i.e., the variables), and mechanics algorithm support services to facilitate rapid code development and code reuse. The motivation in pursuing a shared framework is that the design, debugging, profiling and porting of modern engineering applications is particularly challenging. By sharing these components, considerable effort and separation of concerns can be focused where the need is greatest. Although the shared SIERRA components and applications are predominantly written in C++, a number of compute intensive numerical and material routines are written in Fortran and C.

Complex performant solvers are an important component of almost all production codes at Sandia. In a similar manner to the SIERRA framework (which extracts important shared kernels for engineering routines), we utilize the Trilinos solver library [9] which provides node- and thread-scalable solvers for production use. For this study we do not utilize the threading capability as these are still actively under development.

4.1 SIERRA/SM

SIERRA/SM (*Adagio*) [10] is a Lagrangian nonlinear finite element program for use in analyzing the deformation of solids. It is designed to be MPI parallel and is built upon the SIERRA finite-element framework, employing the Sandia ACME library [11] for contact search algorithms as well as novel native contact algorithms for speed and accuracy. Although the code can scale to large core counts, typical production problems routinely execute at a few hundred to a few thousand cores. Multiple runs of Adagio are often executed with small input deviations in an uncertainty quantification regime to provide more accurate problem analysis.

Adagio provides the following solid mechanics analysis capabilities: implicit quasistatics, explicit transient dynamics, and implicit transient dynamics. In quasistatic simulations, Adagio assumes inertial affects are small where material point velocities are retained but time rates of velocities are neglected. In explicit transient dynamics simulations, mass, acceleration, and inertial forces are taken in account. Explicit transient dynamics does not require a nonlinear solve however is restricted by the Courant time step typically limiting the time-scales to microseconds. Lastly, implicit transient dynamics requires a nonlinear solve to advance in time, has an unconditionally stable time step, and can be applied to longer duration time scales. Sources of nonlinearities include nonlinear stress-strain relations, large displacements, large rotations, large strains, and frictional-frictionless contact mechanics. Quasistatic equilibrium is found using a nonlinear solution strategy, which includes nonlinear conjugate gradients. In this work, we focus on explicit transient dynamics simulations with contact.

Figure 6 shows two variations of the TLC benchmark problem, the first (labeled "TLC") is run at 16-ranks and represents very small problem analysis that is performed by analysts during prospective studies. The refined problem ("TLC-Refined") is a larger run at 128-ranks which is a small, but representative, single run within a UQ set. For the smaller problem Trinity is 46% faster than Cielo and 8.98% faster than the Chama capacity cluster. For the larger refined problem, Trinity is 41.2% faster than Cielo and 5.7% faster the Chama.

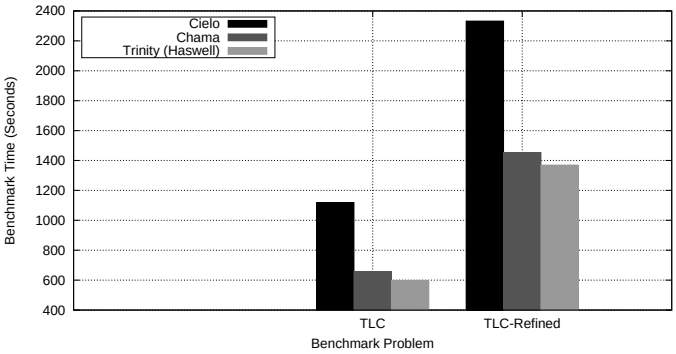

Fig. 6. Benchmarked Adagio Problems on Cielo, Chama and Trinity

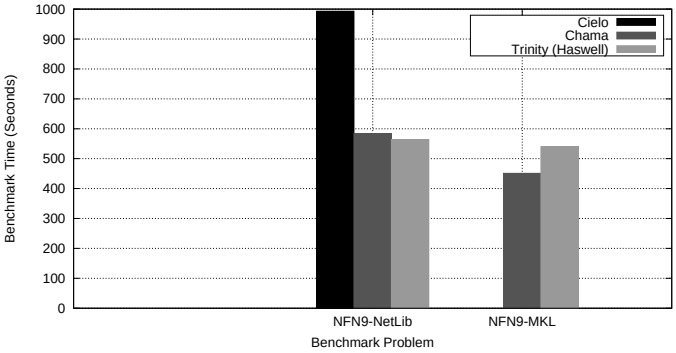

Fig. 7. Benchmarked Salinas Problems on Cielo, Chama and Trinity

Adagio spends a considerable amount of execution time in MPI operations (varying between 20% up to 70% of total execution time when profiled depending on the MPI rank being measured and the degree of dynamic behavior in the problem being studied). This is particularly acute for problems with complex interactions in the ACME contact routines which are exercised in the TLC problem decks. A very large proportion (up to 46%) of the total MPI time is seen in MPI reductions which mostly occur between 8 or 2048 bytes. The significant improvement in the Aries interconnect with respect to MPI performance, particularly small MPI reductions (Figure 5) provides a strong improvement in the performance of this benchmark problem.

4.2 SIERRA/SD

SIERRA/SD (*Salinas*) [11], [12], is an MPI-based massively parallel implicit structural mechanics/dynamics application aimed at providing a scalable computational workhorse for extremely complex finite element (FE) stress, vibration, and transient dynamics models with tens or hundreds of millions of degrees of freedom (dofs).

For Salinas we run a single problem (named "NFN9") at 120 MPI ranks. This problem is a smaller version of much larger analyses that will be performed when the full Trinity comes online to the ASC program. In Figure 7 we show benchmarked results using an internal version of the NetLib BLAS and LAPACK routines [13], [14] as well as benchmark times when the Intel Math Kernel Library (MKL) is used – note that for this problem the 'sequential' (serial) version of MKL is used. In the NetLib case, Trinity compute nodes are 43.4% faster than Cielo and 3.6% faster than the Chama cluster; when MKL is utilized (we omit these results on Cielo because the processor is provided by AMD and therefore the library may only have limited optimization on this platform), Mutrino is 19.7% slower than Chama at the same MPI rank count. We are currently investigating this issue through further analysis.

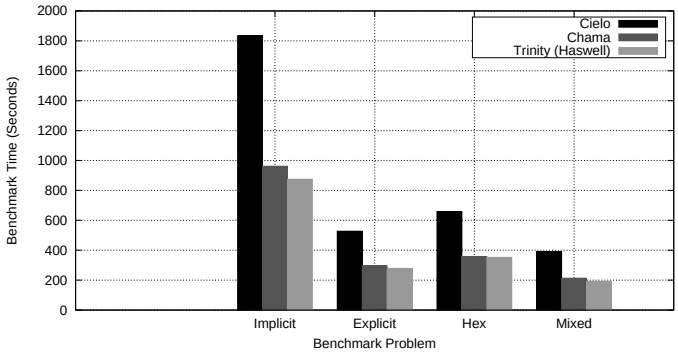

Fig. 8. Benchmarked SIERRA-Aero Problems in Cielo, Chama and Trinity

Salinas spends roughly 40% of its execution time in MPI operations when profiled, although the vast majority of this time (up to 90%) is occupied in waiting for MPI operations during the main solve phases (of this, roughly half is in reductions and half split between call to wait and blocking point-to-point calls). The frequent use of MPI reductions helps to explain some of the performance improvement over Cielo.

The focus on NetLib and MKL reflects our profile-based analysis of differences between execution time on Cielo, Chama and Trinity. When performing a block-solve, Salinas makes frequent calls to `DGEMM` and `DGEMV` to perform matrix-matrix and matrix-vector operations over small, dense sub-blocks in the parent matrix. The profile analysis of these routines shows that execution time on Chama provides almost all of the performance improvement over the Trinity compute nodes.

4.3 SIERRA/Aero

SIERRA/Aero is a compressible fluid dynamics application for turbulent flow simulations [15].

The Aero performance results from Cielo, Chama and Trinity are shown in Figure 8. We show several performance relevant benchmark cases – implict and explict solve cases (that execute at 128 MPI ranks) and Hex and Mixed meshes executing at 512 MPI ranks. Because the Aero application scales very well on modern systems the latter results gain more interest with coding teams and analysts. The Trinity simulations employed one MPI process per Haswell core. Because a Trinity compute node has twice as many cores as the other two platforms, only half as many compute nodes were needed. For the smaller implict and explicit test cases, Trinity is 52.3% and 47.2% faster than Cielo and 9.0% and 5.57% faster than Chama, respectively. In the case of the Hex and Mixed test problems, Trinity is 46.54% and 50.76% faster than Cielo and 1.12% and 9.8% faster than Chama, respectively.

For these four simulations, Aero spent between 10% and 55% of its execution time in MPI operations, the vast

majority of which is spent in small reduction or MPI wait operations. This helps to explain some of the performance difference between Cielo, and the limited improvement with Chama. The vast majority of collective operations are performed at 16 bytes or less which is a particular strong point for the newer Aries interconnect when compared with the previous Gemini network.

4.4 Analysis and Discussion

On a node-for-node basis, Trinity compute resources are twice as dense as both Cielo and Chama (32 processor cores versus 16). Because our application meshes are typically resolved as a fixed number of MPI ranks, our benchmarking evaluation has utilized half as many Trinity compute resources as equivalent runs on Cielo and Chama. When compared machine-to-machine, node density is of interest to the ASC program, and particularly codes in the SIERRA suite which operate under Uncertainty Quantification regimes (where multiple runs are executed in an ensemble to improve prediction accuracy). A 1X performance baseline on Trinity compared to Cielo or Chama in our results, therefore really represents a 2X increase in our capability to run more uncertain quantification ensembles, or put differently, our ability to utilize half of the machine to perform the same workload. Given that the memory bandwidth and networking resources of the machine are less than 2X, the performance improvements on the Trinity test machines represent a significant improvement in the capability available to the production computing users at Sandia.

5 Future Work

In the past two years Sandia has embarked on a path to gradually update and modify existing production engineering codes to utilize shared solver libraries in the form of the Trilinos framework [9] and the use of Kokkos C++ parallelism/data-structure abstractions [16] to portably harness manycore processors such as KNL. While this activity is still very much in the early stages, initial ports of the production applications described in this paper are now well underway. We expect to have preliminary performance information available on early KNL test systems that are to be delivered by Cray under the Trinity Phase-II project. Future work will utilize the experiences and profiling approaches described in this paper to assess areas of strength and weakness on the new Knights Landing platform. We look forward to future Cray User Group meetings where we will be able to attend and describe in detail much of the porting work which has been completed, for both the SIERRA applications discussed here and additional, diverse applications for which porting is underway, such as the Xyce and Charon parallel circuit simulators [17], [18].

6 Conclusions

The deployment of the Trinity supercomputer marks a new approach to capability computing within the NNSA. As the first Advanced Technology System, Trinity will provide a dual role of production capability computing cycles, while, at the same time, providing advanced, future-looking hardware features that allow our application and library code teams to prepare for later generations of Exascale-class machines. Trinity Phase-I, which is based on Haswell, reflects a commitment to continue to provide strong production-capabilities to our code teams, while the later addition, in Phase-II, of self-hosted many-core Knights Landing processors, will provide a significant number of novel hardware features for experimentation, code porting, and additional performance gains.

In this study we have compared the performance that our code teams can expect of the Trinity Phase-I partition with *existing* codes – note that our commitment to continue to optimize our applications and libraries for both Haswell and Knights Landing means we can expect future performance to be strong still. When compared with the previous capability computing resource, Cielo, application performance is 2X on a rank-for-rank basis. The additional doubling of compute node density means that on a node-for-node basis an additional performance improvement of 2X is found over Cielo and Chama. Combined, therefore, our applications can expect aggregate performance improvement on the full machine of 4X as we will now be able to run more jobs, and run each job faster.

Our analysis shows that the performance improvements are multi-faceted – they come from higher memory bandwidth over previous resources as well as a more capable and higher performing interconnect in the form of the Aries DragonFly.

In looking towards the arrival of Knights Landing processors, our application and library teams are focusing on the addition of threading to our important kernels and on ensuring higher levels of vectorization. Many of these benefits will also apply to the Haswell partition benchmarked in this paper.

As a new computing resource for the stockpile stewardship program, Trinity shows great promise. We will simultaneously be able to provide important production capabilities to our application teams while gaining experience with future technologies on the Knights Landing partition. These options are an important part of our planning and commitment to ensure best-of-class engineering analysis to the NNSA program both for near-term and longer-term deliverables.

Acknowledgments

Installing, configuring and testing a new machine deployment the size and complexity of Trinity is a difficult task. We are grateful for the support of our colleagues both at Sandia and Los Alamos who perform these tasks with incredible diligence and professionalism, particularly to Ann Gentile who leads the environment and test system bring up for Sandia. As always, the support and expertise of Cray engineers has been outstanding and we look forward to a deep collaboration with both Phase-I and Phase-II of the Trinity deployments. We are particularly grateful to John Levesque, Mike Davis and - new to our team - Gene Wagenbreth from Cray for their help on our systems. Finally, we also acknowledge considerable input and help from our Center of Excellence colleague from Intel, Ron Green, who has provided deep insight into compiler and environment issues on our machines. Much of our work builds on inputs

from these individuals and we are very pleased to be able to acknowledge their help in this paper.

Preparation of problem input decks and advice on code issues/configuration comes from the various developers of the SIERRA application suite. In particular, support from Mike Tupek, Nate Crane, Mark Merewether, Travis Fisher, Jon Clausen, Clark Dohrmann and Mike Glass has made our activities possible. We thank them for their support and patience with our (many) questions.

Trinity is the first advanced technology computing system ('ATS') operated by the National Nuclear Security Administration (NNSA) for the purposes of ensuring the safety and performance of the United States nuclear deterrent. Access to Trinity and the associated test systems is provided by the NNSA's ASC program to Lawrence Livermore, Los Alamos and Sandia National Laboratories.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

REFERENCES

[1] C. Vaughan, M. Rajan, D. Dinge, C. Dohrman, M. Glass, K. Franko, K. Pierson, and M. Tupek, "Preparation of Codes for Trinity," in *Cray Users' Group (CUG) 2015*, 2015.

[2] D. Doerfler, M. Vigil, S. Dosanjh, and J. Morrison, "The Cielo Petascale Capability Supercomputer," in *Cray User Group Meeting, Fairbanks, Alaska*, 2011.

[3] D. W. Doerfler and M. B. Vigil, "The Cielo Petascale Capability Supercomputer: Providing Large-Scale Computing for Stockpile Stewardship," Los Alamos National Laboratory, NM, USA, Tech. Rep. LA-UR-13-21712, 2013.

[4] R. Alverson, D. Roweth, and L. Kaplan, "The Gemini System Interconnect," in *2010 18th IEEE Symposium on High Performance Interconnects*. IEEE, 2010, pp. 83–87.

[5] M. Rajan, D. Doerfler, P. T. Lin, S. D. Hammond, R. F. Barrett, and C. T. Vaughan, "Unprecedented Scalability and Performance of the New NNSA Tri-Lab Linux Capacity Cluster 2," in *International Workshop in Performance Modeling and Benchmarking of High Performance Computer Systems (PMBS12)*. IEEE, 2012, pp. 417–425.

[6] D. Doerfler, "Trinity: Next-Generation Supercomputer for the ASC Program," Sandia National Laboratories (SNL-NM), Albuquerque, NM (United States), Tech. Rep. SAND 2014-2739C, 2014.

[7] J. D. McCalpin, "Memory Bandwidth and Machine Balance in Current High Performance Computers," *IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter*, pp. 19–25, December 1995.

[8] J. R. Stewart and H. C. Edwards, "The SIERRA Framework for Developing Advanced Parallel Mechanics Applications," in *Large-Scale PDE-Constrained Optimization*. Springer, 2003, pp. 301–315.

[9] M. A. Heroux, R. A. Bartlett, V. E. Howle, R. J. Hoekstra, J. J. Hu, T. G. Kolda, R. B. Lehoucq, K. R. Long, R. P. Pawlowski, E. T. Phipps *et al.*, "An Overview of the Trilinos Project," *ACM Transactions on Mathematical Software (TOMS)*, vol. 31, no. 3, pp. 397–423.

[10] J. A. Mitchell, A. S. Gullerud, W. M. Scherzinger, R. Koteras, and V. L. Porter, "Adagio: Non-Linear Quasi-Static Structural Response using the SIERRA Framework," 2001.

[11] M. Bhardwaj, K. Pierson, G. Reese, T. Walsh, D. Day, K. Alvin, J. Peery, C. Farhat, and M. Lesoinne, "Salinas: A Scalable Software for High-Performance Structural and Solid Mechanics Simulations," in *Proceedings of the 2002 ACM/IEEE Conference on Supercomputing*. IEEE Computer Society Press, 2002, pp. 1–19.

[12] M. K. Bhardwaj, G. M. Reese, B. Driessen, K. F. Alvin, and D. M. Day, "Salinas – An Implicit Finite Element Structural Dynamics Code developed for Massively Parallel Platforms," Sandia National Labs., Albuquerque, NM (US), Tech. Rep., 2000.

[13] S. Browne, J. Dongarra, E. Grosse, and T. Rowan, "The Netlib Mathematical Software Repository," *D-Lib Magazine, Sep*, 1995.

[14] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*. SIAM, 1999, vol. 9.

[15] K. J. Franko, T. C. Fisher, P. Lin, and S. W. Bova, "CFD for Next Generation Hardware: Experiences with Proxy Applications," in *Proceedings of the 22nd AIAA Computational Fluid Dynamics Conference, June 22–26, 2015, Dallas, TX*, AIAA 2015–3053.

[16] H. C. Edwards, D. Sunderland, V. Porter, C. Amsler, and S. Mish, "Manycore Performance-Portability: Kokkos Multidimensional Array Library," *Scientific Programming*, vol. 20, no. 2, pp. 89–114, 2012.

[17] S. Hutchinson, E. Keiter, R. Hoekstra, A. WATERS, T. Russo, R. Schells, S. Wix, and C. Bogdan, "THE Xyce Parallel Electronic Simulation - An Overview," *Parallel Computing*, p. 165, 2000.

[18] G. L. Hennigan, D. Fixel, J. P. Castro, and P. Lin, "Charon Parallel Semiconductor Device Simulator," Sandia National Laboratories, Tech. Rep. SAND2010-3905, 2008.