Enabling Co-Design of Multi-Layer Exascale Storage Architectures

Award Number: DE-SC0004875

Final Report through 8/31/2015

1 PI Personnel

Christopher D. Carothers, Rensselaer Polytechnic Institute

2 Goals and Objectives

Growing demands for computing power in applications such as energy production, climate analysis, computational chemistry, and bioinformatics have propelled computing systems toward the exascale: systems with 10^{18} floating-point operations per second. These systems, to be designed and constructed over the next decade, will create unprecedented challenges in component counts, power consumption, resource limitations, and system complexity. Data storage and access are an increasingly important and complex component in extreme-scale computing systems, and significant design work is needed to develop successful storage hardware and software architectures at exascale. Co-design of these systems will be necessary to find the best possible design points for exascale systems.

The goal of this work has been to enable the exploration and co-design of exascale storage systems by providing a detailed, accurate, and highly parallel simulation of exascale storage and the surrounding environment. Specifically, this simulation has (1) portrayed realistic application checkpointing and analysis workloads, (2) captured the complexity, scale, and multilayer nature of exascale storage hardware and software, and (3) executed in a timeframe that enables "what if" exploration of design concepts. We developed models of the major hardware and software components in an exascale storage system, as well as the application I/O workloads that drive them. We used our simulation system to investigate critical questions in reliability and concurrency at exascale, helping guide the design of future exascale hardware and software architectures. Additionally, we provided this system to interested vendors and researchers so that others can explore the design space. We validated the capabilities of our simulation environment by configuring the simulation to represent the Argonne Leadership Computing Facility Blue Gene/Q system and comparing simulation results for application I/O patterns to the results of executions of these I/O kernels on the actual system.

3 Technical Progress

3.1 Torus Network Simulation

Torus networks (Figure 1) have been widely employed as the underlying network topology for many supercomputing systems, such as the Blue Gene [Adiga2005] and Cray XT [Bland2009] families. Torus networks offer

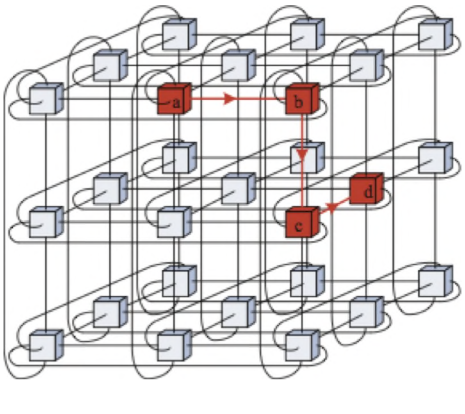


Figure 1: 3-ary 3-cube torus.

low latency for nearest neighbor communication and scalable bisection bandwidth. They also provide an easy physical wiring plan for upgrading a system with additional nodes without having to updating entire core network [Adiga2005].

We constructed a packet-level simulation of torus networks using the Rensselaer's Optimistic Simulation System (ROSS), a parallel, discrete-event simulation framework using Jefferson's Time Warp event scheduling mechanism [Jefferson1985].

The simulation results were validated using Little's law for different torus configurations under varying packet arrival rates. We also conducted comparison tests between the actual Blue Gene torus network and our model using MPI Send()/MPI Recv(), showing reasonable correlation to actual results.

We are able to achieve a near-linear speedup for our torus model. On Blue Gene/L, the peak event-rate on 32K cores is 4.78 G/s. On Blue Gene/P, the best event-rate observed is 12.359G/s on 128K cores. We further demonstrated the ability to model and simulate a million-node and a billion-node torus network on both Blue Gene/L and Blue Gene/P platforms [Liu2012].

Following this initial work, we have focused efforts in further improving the fidelity of the model, focusing on even more accurately modeling the BG/P and BG/Q networks for MPI communication (Figure 2). The mpptest tool is being used to capture performance results for comparison purposes.

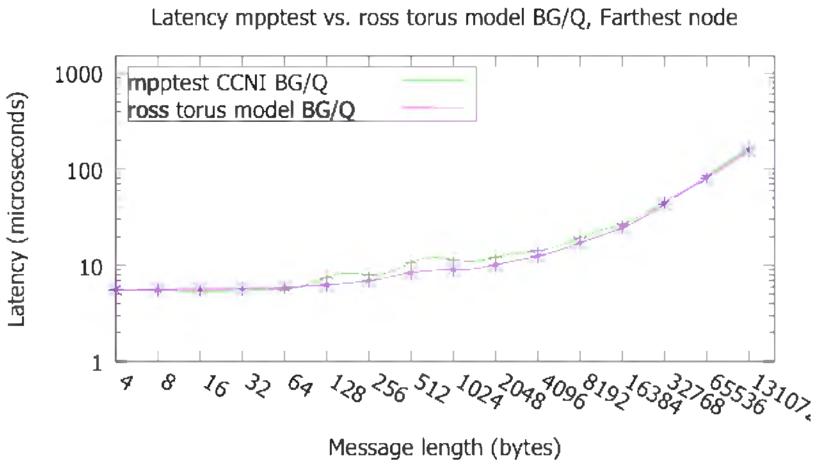


Figure 2: Comparing experimental results to simulation of network latency in BG/Q system. Improvements to our torus model have resulted in highly accurate modeling of communication latency and bandwidth.

In follow on work [Mubarak2014a and Mubarak2014b], we improve both the accuracy of our torus network model as well as add the ability to efficiently simulate collective network operations.

3.2 Dragonfly Network Simulation

The dragonfly network topology (Figure 3), a two-level directly connected network, is another candidate for exascale architectures because of its low diameter and reduced latency. The dragonfly topology lowers the cost of the interconnection network and improves network performance by using high-radix routers, made possible by increasing chip pin counts, to reduce the diameter of the network and limit the number of global channels traversed by packets [Kim2008, Kim2009].

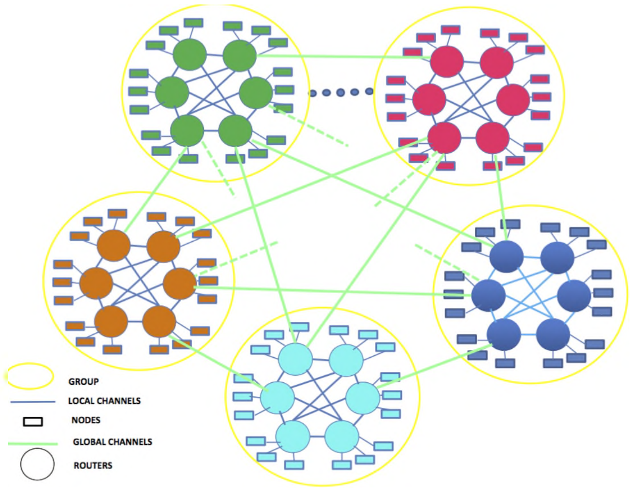


Figure 3: Dragonfly network topology (p=h=3, a=6). Dotted lines show global channels connected to other groups not shown in figure.

In the case of the Dragonfly network, a high fidelity (but less scalable) simulator (BookSim) had been used by Kim et al. to examine the network characteristics for systems with up to 1,024 endpoints and 264 routers. Although BookSim is not appropriate for use in exascale system simulation, it has proven to be a useful validation tool until more Dragonfly based systems are deployed. Figure 4 shows a performance comparison between BookSim and our Dragonfly simulation: even at small scale our simulation executes much more quickly while producing very accurate simulation results.

We evaluated the strong-scaling characteristics of the dragonfly network model on two massively parallel architectures: the Argonne Leadership Computing Facility (ALCF) IBM Blue Gene/P system (Intrepid) and the Computational Center for Nanotechnology Innovations (CCNI) IBM Blue Gene/Q system. We used three problem sizes of the dragonfly model. The first test case has 4,196,352 nodes and 131,000 routers. The second test case has 10 million nodes and 256,000 routers, and the third test case has 50 million nodes and 864,000 routers. Simulations achieve a peak event rate of 1.3 billion events per second, with total committed events of up to 872 billion on the CCNI Blue Gene/Q.

As part of this activity we also evaluated the impact of various ROSS parameters on the rate of simulation on the two platforms, identifying specific configurations that lead to higher performance. These parameters will assist us in making most effective use of the platforms for future simulations.

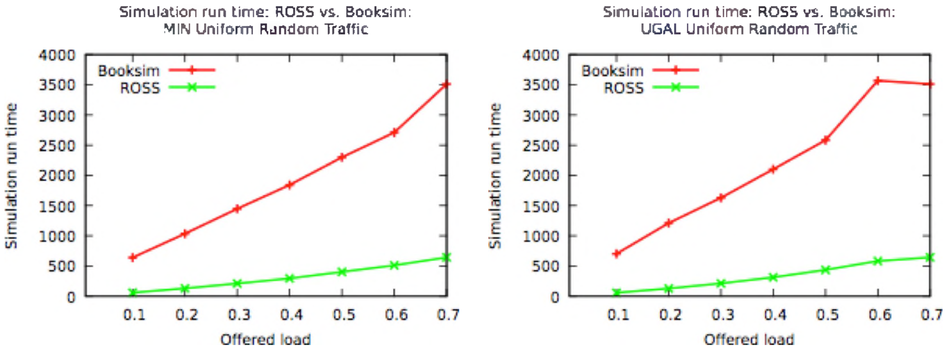


Figure 4: Simulation run time of ROSS Dragonfly model as compared to BookSim for random traffic using different routing algorithms (MIN vs. UGAL).

3.3 Blue Gene/P I/O System Simulation

While the system interconnect is a critical component to model, the other hardware and especially software components are also critical pieces. In [Liu2011] we present the design of and results from simulations of the Intrepid Blue Gene/P system at Argonne National Laboratory.

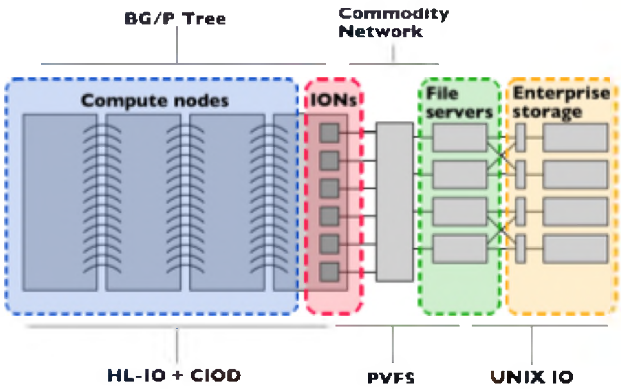
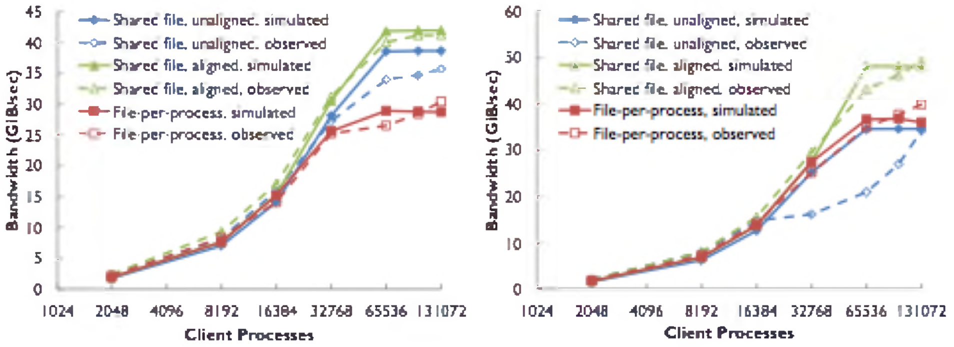


Figure 5: CODES models for the ALCF computing environment, including hardware and software.

We abstracted the common features of each Blue Gene/P hardware component into CN, ION, file server, and DDN models. Our software models approximate the interfaces, protocols, and interactions of the software components deployed in the ALCF computing environment. At the application layer, our models provide a POSIX-like I/O interface. Our application-level models translate application I/O requests into CIOD client requests using a series of CN and ION events. These CN and ION events reflect the interaction between the CIOD clients and servers. The CIOD server receives the CIOD client requests and generates a series of ION and storage server hardware requests that approximate the interaction of the CIOD server and the PVFS file system. The PVFS file system then generates a series of storage server and DDN events that approximate the interactions between the storage server and the DDN storage devices. The accuracy of these models with respect to the protocols used in the software implementations contributes strongly to the accuracy of our results.

The model was validated against results from a prior study of this Blue Gene/P system [Lang2009]. Figure 6 demonstrates our ability to accurately model a variety of workloads and validate them using the IOR synthetic benchmark.

All the simulations, in this case, ran on an SMP system with a configuration of 8 cores (Intel Xeon x5430, 2.67 GHz) and 32 GiB memory. The largest test case with 128K client processes (represented as LPs in the simulation) finished within a couple of minutes, showing that our tools are capable of simulating interesting storage system designs while using modest resources.



(a) write

(b) read

Figure 6: Comparison of simulated and observed IOR performance. Discrepancies in read performance appear to be due to contention in the external GigE network that was not modeled in our simulations.

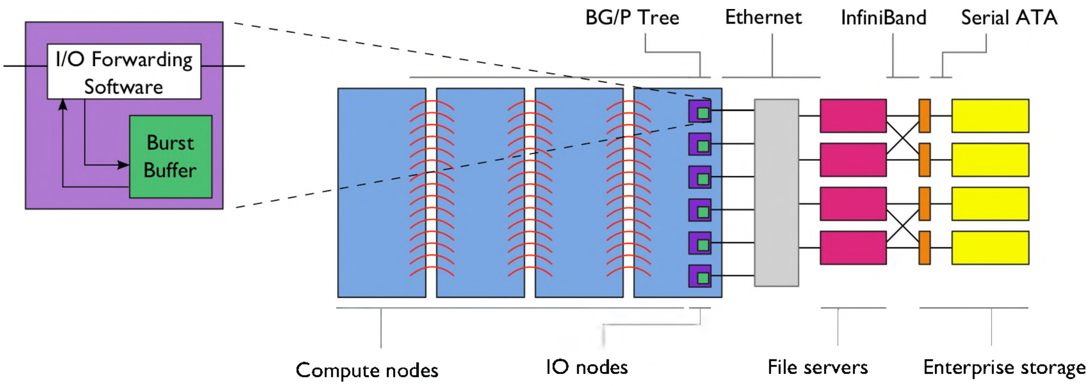


Figure 7: Overview of Blue Gene/P computing environment, showing proposed tier of burst buffers integrated into existing I/O nodes.

3.4 Burst Buffer Simulation

The largest-scale high-performance (HPC) systems are stretching parallel file systems to their limits in terms of aggregate bandwidth and numbers of clients. To further sustain the scalability of these file systems, researchers and HPC storage architects are exploring various storage system designs. One proposed storage system design integrates a tier of solid-state burst buffers into the storage system to absorb application I/O requests.

Building on the prior BG/P simulation, we integrated a model of a solid-state storage device into the I/O nodes of the BG/P system [Liu2012]. We then examined application I/O patterns on an existing large-scale HPC system to identify common burst patterns, and developed a mechanism for reproducing I/O patterns for simulation. We identified four write-intensive science applications with significant “bursts”. We discovered examples of production applications that generated as much as 67 TiB of data in a single execution. Two of the top four applications (Turbulence1 and AstroPhysics) illustrate the classic HPC I/O behavior in which data is written in several bursts throughout the job execution, each followed by a significant period of idle time for the I/O system. The PlasmaPhysics application diverged somewhat in that it produced only two bursts of significant write activity; the first burst was followed by an extended idle period, while the second burst occurred at the end of execution. The Turbulence2 application exhibited a series of rapid bursts that occurred nearly back-to-back at the end of execution.

A number of studies were performed, starting with simple synthetic benchmarks and culminating in a study of multi-application behavior. One of the observations we made from the multi-application experiment is that burst buffers accelerate the application perceived throughput under mixed I/O workloads. A modest, 400 GiB burst buffer per ION was large enough to buffer the data requests generated by all three workloads.

Additionally, decreasing the size of the storage system by half while using burst buffers had no noticeable impact on the mixed I/O workloads performance – a much less capable external I/O system would be just as effective. For today’s systems this means that storage system costs could likely be significantly reduced—fewer file servers, racks of storage, and external switch ports are needed. For systems in the 2020 time frame, burst buffers are likely to be a mandatory component if peak I/O rates are to be attained.

3.5 I/O Workload Modeling

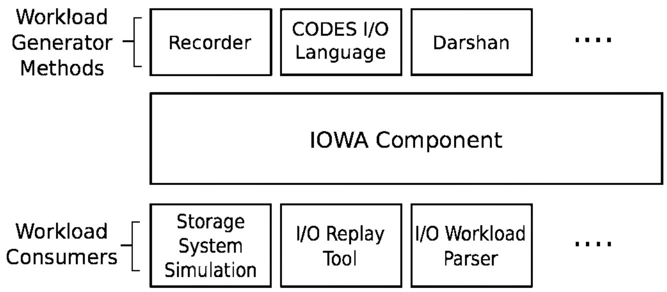


Figure 1: Interaction of IOWA with different workload generators and consumers (e.g., CODES simulations).

Accurate analysis of HPC storage system designs is contingent on the use of I/O workloads that are truly representative of expected use. Generally, I/O analyses are bound to specific workload modeling techniques such as synthetic benchmarks or trace replay mechanisms, however, despite the fact that no single workload modeling technique is appropriate for all use cases. We have designed *IOWA* (Figure 1), a novel I/O workload abstraction that allows arbitrary workload consumer components to obtain I/O workloads from a range of diverse input sources. Using IOWA, researchers can choose specific I/O workload generators based on the resources they have available and the type of evaluation they wish to perform. As part of this research, we also designed three distinct workload generation methods, based on I/O traces, synthetic I/O kernels, and I/O characterizations. We analyzed each of these workload generation techniques in the context of storage system simulation models as well as production storage systems. We found that each generator mechanism offers varying levels of accuracy, flexibility, and breadth of use that should be considered before performing I/O analyses. We also developed a set of best practices for HPC I/O workload modeling based on challenges that we encountered while performing our evaluation.

This work was published in [Snyder2015].

3.6 Preparing CODES for External Use

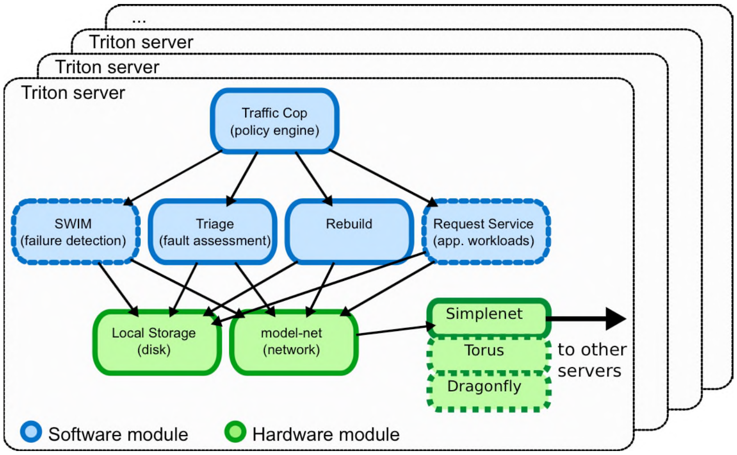


Figure 8: Example of multi-component storage model using CODES infrastructure.

One of the goals of this project is to provide the CODES models to the larger research community as a tool for accelerating understanding of exascale storage systems.

An important step in realizing this goal is the development of modularization techniques that allow rapid prototyping and integration of new models. This is not a capability provided by the underlying ROSS framework. Thus far, modular components for local storage (disk) and network have been developed, and the models described earlier in this document are being ported to operate in this new mode (Figure 8).

Additionally, we have developed functionality for capturing simulation results at runtime using data buffering and collective data aggregation, allowing large output to be quickly stored for subsequent data analysis.

Finally, we have contributed a number of enhancements and bug fixes back to the ROSS development team to enhance that framework and ease configuration and build by new users.

3.7 Impact

Impact thus far has been on three fronts. First, in terms of simulation in the context of exascale, we are pushing the boundaries of what is possible using the Time Warp approach, showing that very high fidelity models can be executed quickly using current terascale and petascale resources [Barnes2013]. These successes have been in part responsible for others using event-driven simulation as a tool for understading future exascale systems (e.g., the Intel Fast Forward team simulating gossip protocols in this manner, as we understand it, using ROSS).

Second, we have produced the first quantitative (if simulated) results showing the potential for the burst buffer approach in HPC systems. An important result of our work wasn't just that burst buffers are plausible, but that they actually *reduce* the bandwidth needed from the external I/O system. This is a critical result.

Third, we are developing infrastructure that can be more broadly applied. Figure 8 shows an example of using CODES components in the context of future storage architecture simulations being pursued under external funding. Colleagues at UCSC and elsewhere are similarly interested in deploying CODES in their own work.

4 Deliverables

4.1 Publications

[Liu2011] N. Liu, C. Carothers, J. Cope, P. Carns, R. Ross, A. Crume, and C. Maltzahn. Modeling a leadership-scale storage system. In Proceedings of the 9th International Conference on Parallel Processing and Applied Mathematics 2011 (PPAM 2011), October 2011.

[Liu2012] N. Liu, C. Carothers, J. Cope, P. Carns, and R. Ross. Model and simulation of exascale communication networks. Journal of Simulation, March 2012.

[Liu2012] N. Liu, J. Cope, P. Carns, C. Carothers, R. Ross, G. Grider, A. Crume, and C. Maltzahn. On the role of burst buffers in leadership-class storage systems. In Proceedings of the 2012 IEEE Conference on Massive Data Storage, Pacific Grove, CA, April 2012.

[Mubarak2012] M. Mubarak, C. D. Carothers, R. B. Ross, and P. Carns. Modeling a million-node dragonfly network using massively parallel discrete event simulation. In 3rd International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS12) held as part of SC12, November 2012.

[Barnes2013] P. D. Barnes, C. D. Carothers, D. R. Jefferson and J. M. Lapre. Warp Speed: Executing Time Warp on 1,966,080 Cores. Proceedings of the ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (PADS). May, 2013. Montreal, CA.

[Mubarak2014a] M. Mubarak, C. D. Carothers, R. B. Ross, and P. Carns, "A case study in using massively parallel simulation for extreme-scale torus network codesign," in *Proc. of the 2nd ACM SIGSIM/PADS Conf. on Principles of Advanced Discrete Simulation*, 2014, pp. 27–38.

[Mubarak2014b] M. Mubarak, C. D. Carothers, R. B. Ross, and P. Carns, "Using massively parallel simulation for MPI collective communication modeling in extreme-scale networks," in *Proc. of the 2014 Winter Simulation Conf.*, 2014, pp. 3107–3118.

[Snyder2015] S. Snyder, P. Carns, R. Latham, M. Mubarak, R. Ross, C. D. Carothers, B. .B Huong. V. T. Luu and S. B. Prabhat "Techniques for Modeling Large-Scale HPC I/O Workloads", In Proceedings of the 5th International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS 2015) as part of Supercomputing (SC'15). Austin, TX, November 2015.

5 References

[Adiga2005] Adiga N R, Blumrich M A, Chen D, Coteus P, Gara A, Giampapa M E, Heidelberger P, Singh S, Steinmacher- Burow B D, Takken T, Tsao M and Vranas P (2005). Blue Gene/L torus interconnection network. *IBM J. RES. & DEV.* 49: 265–276.

[Bland2009] Bland A S, Kendall R A, Kothe D B, Rogers J H and Ship- man G M (2009). Jaguar: The world's most powerful computer. *Compute the Future, CUG 2009 Proceedings.* Atlanta, Geogia.

[Jefferson1985] Jefferson D R (1985). Virtual time. *ACM Trans. Program. Lang. Syst.* 7: 404–425.

[Kim2008] J. Kim, W. Dally, S. Scott, and A. D., "Technology-driven, highly- scalable dragonfly topology," ACM SIGARCH Computer Architecture News, vol. 36, no. 3, pp. 77–88, 2008.

[Kim2009] J. Kim, W. Dally, S. Scott, and Abts, "Cost-efficient dragonfly topology for large-scale systems," Micro, IEEE, vol. 29, no. 1, pp. 33–40, 2009.

[Lang2009] S. Lang, P. Carns, R. Latham, R. Ross, K. Harms, and W. Allcock. I/O perfor- mance challenges at leadership scale. In Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis, page 40. ACM, 2009.