Toward a Proof of Concept Cloud Framework for Physics Applications on Blue Gene Supercomputers

Patrick Dreher\textsuperscript{1,2}, William Scullin\textsuperscript{3}, Mladen Vouk\textsuperscript{1}

\textsuperscript{(1)}-NC State University, \textsuperscript{(2)} Massachusetts Institute of Technology, \textsuperscript{(3)} Argonne Leadership Computing Facility

E-mail: padreher@ncsu.edu, wscullin@alcf.anl.gov, vouk@csc.ncsu.edu

Abstract. Traditional high performance supercomputers are capable of delivering large sustained state-of-the-art computational resources to physics applications over extended periods of time using batch processing mode operating environments. However, today there is an increasing demand for more complex workflows that involve large fluctuations in the levels of HPC physics computational requirements during the simulations. Some of the workflow components may also require a richer set of operating system features and schedulers than normally found in a batch oriented HPC environment. This paper reports on progress toward a proof of concept design that implements a cloud framework onto BG/P and BG/Q platforms at the Argonne Leadership Computing Facility. The BG/P implementation utilizes the Kittyhawk utility and the BG/Q platform uses an experimental heterogeneous FusedOS operating system environment. Both platforms use the Virtual Computing Laboratory as the cloud computing system embedded within the supercomputer. This proof of concept design allows a cloud to be configured so that it can capitalize on the specialized infrastructure capabilities of a supercomputer and the flexible cloud configurations without resorting to virtualization. Initial testing of the proof of concept system is done using the lattice QCD MILC code. These types of user reconfigurable environments have the potential to deliver experimental schedulers and operating systems within a working HPC environment for physics computations that may be different from the native OS and schedulers on production HPC supercomputers.

1. Introduction

Physics researchers running applications needing state-of-the-art computational hardware and software usually select supercomputer systems to run those applications. Although these systems have the capability to deliver the highest level of CPU power and network connectivity, what gets sacrificed in such a selection is the flexibility of choices for operating systems and user customized software. Over the past several years cloud computing\cite{1} has emerged as an alternate hardware architecture for computation. From a technical and operational perspective, users now have a spectrum of choices in the design and configuration of customized system and application software stacks for various types of physics computations and analysis.

Today cloud computing options have been applied to situations where users have constraints on facility access to computational resources, small prototype computations needing many short calculations with different parameters, and large computations with minimal communications requirements between processors. Some of these types of computations have been successfully implemented in both private or commercial cloud computing systems\cite{2,3}. The overall consensus is that today’s cloud technologies have now advanced to the point where they can
provide end-users with considerable flexibility and reliability to self-provision resources, either explicitly or implicitly, and provide on-demand computational capabilities and services. As a result, companies, academic institutions, organizations and individuals are seriously considering and experimenting with cloud computing as a platform for computation and data analysis.

Despite all of these advances, cloud computing has only had mixed success when attempting to implement physics supercomputing applications onto these types of platforms. Users explicitly requiring high performance computing favor systems that allow them to operate ”close to the metal” with the ability to tune both the hardware and storage in order to optimize computational performance and overall throughput. Early efforts to re-create these HPC capabilities in cloud systems selected the most straightforward option of deploying these physics supercomputing applications onto existing cloud platforms. Although this method did show some promise for codes with minimal inter-processor communications requirements, the more tightly coupled HPC physics applications suffered degraded performance. Alternative approaches were then explored that involved constructing small groups of HPC clouds with more robust uniform hardware architectures and network connections. This design provided ”spill-over provisioning” from the HPC supercomputer to a cloud system when the HPC system became saturated [4]. Although these implementations did provide some overall acceleration, the underlying shortcomings of delivering HPC supercomputer level computational throughput with commodity cloud cluster hardware still remained problematic.

The basic difficulty is that general cloud computing systems lack the specialized HPC architectural infrastructure needed to deliver the required high throughput. There are many examples of tightly coupled HPC codes that require state-of-the-art network interconnects to provide maximum computational throughput and minimum latency. For example physics applications that need lattice based infrastructures and communications requirements between nearest and next nearest neighbor sites usually have some of the highest memory bandwidth and network interconnect requirements. Running such applications on standard cloud computing systems generally results in degraded performance. Additional performance degradation is also attributed to a lack of uniformity in the computational hardware.

The motivation for this proof of concept project was to work toward building and provisioning a secure cloud framework tuned to a computational physics code based on a customized system software and application stack not found or able to be installed on current supercomputer systems. That stack would be embedded in a host’s supercomputer hardware architecture in a way that would address many of the shortcomings noted with the current cloud computing implementations. Beyond the novelty of constructing such a system, this type of construct would capitalize on the infrastructure advantages and enhanced capabilities of a supercomputer, allowing the cloud to take advantage of the installed high speed network. This proof of concept cloud framework on a supercomputer would provide the low latency interconnects between processors and allowing for the cloud computing cluster to capitalize on the localized homogeneous and uniform HPC supercomputer architecture.

This type of design may also potentially assist with the issue of data migration and transfer. In many instances data from physics simulations and/or data input from sensors/devices are stored on disk farms at the supercomputer center. The post-processing and analysis of that data usually requires customized software stacks sometimes located at sites remote to the production supercomputer centers. Having the flexibility to change/customize the software stacks and operating systems within an HPC architecture using a cloud environment may alleviate the need to move data from one physical location to another to access the appropriate computational system.

In addition, these ideas offer the potential for designing frameworks that may lead to better hybrid workflows. This work may also allow for experimentation with new schedulers and operating systems within a working HPC environment that may be different from the native
OS and schedulers on the HPC supercomputer itself. Finally, from an economic perspective, the idea of capitalizing on the overall operational cost effectiveness of a supercomputer, such as the IBM Blue Gene architecture, and merging it with the elasticity of a cloud computing architecture may be a very cost effective design for the future.

Section 2 summarizes the key HPC and cloud computing design characteristics and the proof of concept implementation on the BG/P and Blue Gene/Q architectures. Section 3 provides a brief introduction into the properties of the lattice QCD physics code selected to test the cloud framework on the Blue Gene. Section 4 discusses some of the initial testing of physics applications on these systems. Finally section 5 summarizes possible next steps and suggested future trends for applying physics applications to cloud technologies and supercomputers.

2. Proof of Concept Cloud Computing Framework on the BG/P and BG/Q

The idea for the original project was to select a suitable supercomputer in which to embed a cloud computing framework. The IBM Blue Gene/P was selected as the HPC architecture because of its network topologies and excellent hardware infrastructure. In addition, an IBM research group ([5],[6]) developed an open source software utility called Kittyhawk. This software utility was implemented on the IBM Blue Gene/P architecture and showed promise for this project because it incorporated some of the key building blocks needed for embedding a cloud system into the BG/P with the needed access to internal supercomputer networks and communications. The cloud computing system selected for this project was a proven open source production level cloud architecture, called the Virtual Computing Laboratory (VCL) ([10],[11],[12],[13]). The software for VCL is available through the Apache Software Foundation [14]. This proof of concept design was demonstrated at the Department of Energy’s Argonne Leadership Computing Facility (ALCF). Dreher, Vouk and Mathew recently published detailed reports ([8],[9]) documenting a successful demonstration of this proof of concept implementation of a secure cloud framework embedded in a BG/P in such a way that it allowed the VCL cloud system to capitalize on the host’s supercomputer infrastructure. Based on the results from that work, these proof of concept ideas and framework were extended to the new Blue Gene/Q architecture. Dreher, Vouk and Scullin [7] recently reported on the initial the proof of concept cloud framework installation on the BG/Q.

3. Lattice QCD Physics Application

The physics application selected to test this new proof of concept cloud framework on the Blue Gene is a physics code used for lattice Quantum Chromodynamics (LQCD). This application is especially computationally demanding with large memory bandwidth and low latency requirements. LQCD is a non-perturbative implementation of a quantum field theory using the Feynman path integral approach. The method for lattice QCD calculations proceeds analogously as if the quantum field theory was being solved analytically.

The starting point for the LQCD approach is the partition function in Euclidean space-time

\[ Z = \int D\mathbf{A}_\mu \, D\psi \, D\bar{\psi} \, e^{-S} \]  

(1)

where \( S \) is the QCD action

\[ S = \int d^4x \mathcal{L} \]  

(2)

and the Lagrange density \( \mathcal{L} \) represents the continuum Euclidean QCD. The expanded equations for the Lagrange density are

\[ \mathcal{L} = \sum_r \sum_i \bar{\psi}_{ri} i \mathbf{D} \psi_{ri} - \frac{1}{4} F_{\mu\nu} F^{\mu\nu} \]  

(3)
For fermions coupled to the strong nuclear force through gluon interactions in a particular
gauge group the Dirac operator is
\[ D = \partial - igA^a Tr \]

and the field strength tensor \( F^a_{\mu\nu} \) is
\[ F^a_{\mu\nu} = \partial_\mu A^a_\nu - \partial_\nu A^a_\mu + gf^{abc}(A_\mu)_b(A_\nu)_c \]

The matrices \( T^a_r \) represent the generators in the representation \( r \) of the gauge group to which
the fermions are assigned. The index \( a \) runs over the generators of the gauge group, \( i \) runs over
all \( n(r) \) fermion flavors belonging to the representation \( r \). The \( f^{abc} \) are the structure constants
of the gauge group. The fermions are represented by Grassmann variables \( \psi \) and \( \bar{\psi} \). These can
be integrated exactly with the result
\[ Z = \int DA_\mu \det M e^{\int d^4x \left( -\frac{1}{4} F_{\mu\nu} F^{\mu\nu} \right)} \]

After integration over the fermions, the fermionic contribution is now contained in the highly
non-local term \( \det M \) where \( M \) is the Dirac operator. The partition function at this point
is an integral over only background gauge configurations that can be written as \( S = S_{\text{gauge}} + S_{\text{quarks}} = \int d^4x \left( \frac{1}{4} F_{\mu\nu} F^{\mu\nu} \right) - \sum_i \log(\text{Det} M_i) \) where the sum is over the quark flavors,
distinguished by the value of the bare quark mass.

One of the objectives of the computations in lattice QCD is to obtain values for experimentally
observed quantities by calculating expectation values of operators
\[ \langle O \rangle = \frac{1}{Z} \int DA_\mu O e^{-S} \]

where \( O \) is any given combination of operators expressed in terms of time-ordered products of
 gauge and quark fields.

At present, the only means of carrying out such non-perturbative QCD calculations is
through large scale physics first principles numerical simulations within the framework of lattice
gauge theory (lattice QCD). These HPC simulations calculate expectation values of operators
representing quarks and gluons that are believed to approximate the physical observables in the
laboratory. Examples of such calculations include efforts to calculate the mass spectrum of the
elementary particles (such as the proton) from these theoretical physics first principles.

The method summarized above is the only known technique for solving this type of relativistic
quantum field theory. For readers interested in a complete in-depth discussion of the physics
and computational methods of Lattice Quantum Chromodynamics the book by DeGrand and
DeTar [15] is an excellent reference.

4. Testing the Proof of Concept Cloud Framework on The Blue Gene Using
Lattice QCD
The first tests performed on this design were basic ”hello world” codes. The basic ”hello world”
tests utilizing MPI functions were successful with the BG/P design using Kittyhawk and the
BG/Q design using the FusedOS operating system described in [7]. The more interesting and
complex tests involve installing production physics application codes using these proof of concept
design based on the FusedOS and VCL cloud computing system installed on the BG/Q.

The Kittyhawk software worked well on the BG/P and the utility was able to be successfully
integrated with the VCL system. The BG/Q architecture differed sufficiently from the BG/P
and so a different approach of using an experimental heterogeneous operating system (FusedOS)
was selected for testing. Some tests of the FusedOS environment have already been conducted by other groups. The authors in [19] tested the FusedOS using the LAMMPS code. LAMMPS is an HPC benchmark that simulates nearest neighbor interactions by constructing "particle-like" models of systems such as molecular dynamics. The specific tests within LAMMPS employed a three dimensional Lennard-Jones melt benchmark to test the code under the FusedOS heterogeneous environment. The detailed results are contained within their paper. The authors extend their comments regarding FusedOS in [20] suggesting that the FusedOS approach can accommodate existing codes already running in a compute node kernel (CNK) environment on a BG/Q without modifications.

Although the testing of the simple "hello world" codes utilizing MPI functions has been successful under FusedOS on blocks built of up to 128 nodes and I/O nodes, the more realistic lattice QCD physics application code has encountered difficulties. There were several solid production open source lattice QCD codes available from which to choose because theoretical physicists in the high energy and nuclear physics communities have banded together in various collaborations to write the necessary computer codes to do these numerical simulations on the fastest supercomputers. One of these lattice QCD groups that has developed QCD codes is the MIMD Lattice Computation (MILC) Collaboration [16]. This collaborations has written a fully open source implementation of lattice QCD that has been optimized in production for both the IBM BG/P and BG/Q supercomputers. Our project utilized a small test code within MILC (ks-spectrum) on a 4 node BG/Q to analyze the particle physics spectrum based on some Lattice QCD generated configurations.

When the MILC code from the MIMD Lattice Computation collaboration was implemented under FusedOS, several issues were encountered. The two primary issues noted involve handling of alignment issues and network geometry. When using the PowerPC A2 CPU’s QPX vector unit, all addresses should be aligned to the nearest 32-byte boundary. If the alignment of an address is off for any reason, an exception is generated. The handling of this exception is software dependent. Under the CNK operating system, these alignment exceptions are trapped and handled, though a performance penalty is incurred each time. For this reason, CNK uses a user-tunable environment variable, $\textit{BG}_M\textit{AXALIGNEXP}$, to allow the user to set a maximum number of alignment exceptions to handle before forcing the application to terminate and core dump. Under FusedOS, the handling of these alignment errors is not handled by the operating system and application failure is assured.

It was determined that the design of the Blue Gene/Q hardware and FusedOS software have presented obstacles to the successful execution of production codes such as MILC, though it is believed that these observed difficulties are not insurmountable. It is suspected that most of those issues likely trace back to alignment exceptions. A second issue that arose was an incompatibility between the geometry of the available block and the MILC binary. While the MILC binary had no restrictions on geometry, it had been suggested that the smallest sub-block under FusedOS should be at least eight contiguous nodes in five dimensions. When attempted runs failed, crashing both application and operating system, we found it difficult to debug, relying on logging and hardware bugging interfaces in an attempt to de-construct the origin.

5. Observations and Next Steps

The continued development of these heterogeneous cloud frameworks within a supercomputer host environment still offer potential interesting alternatives for embedding cloud framework with supercomputer host architectures to develop hybrid workflows, schedulers and potential HPC cloud computing environments. In the short term, we continue to work with developers and will continue to try to target MILC as a sample application as it has been successfully run at all node counts in a Linux environment and under CNK. Over the longer term, continued research into advanced heterogeneous OS environments such as FusedOS may play a role toward
the development of exascale level capabilities for physics computations and may provide cost effective options for combining the HPC capabilities of supercomputers with the provisioning flexibility and elasticity of cloud computing.

6. Acknowledgements
The authors would like to thank Dr. Yoonho Park from IBM, Dr. Eric Van Hensbergen from ARM and Marius Hillenbrand from Karlsruhe Institute of Technology for helpful conversations on FusedOS. This work is supported in part through NSF grant 0910767, 1318564, 1330553, the U.S. Army Research Office (ARO) grant W911NF-08-1-0105 managed by the NCSU Science of Security Initiative and the Science of Security Lablet, by the IBM Share University Research and Fellowships program funding, and the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357. One of us (Patrick Dreher) gratefully acknowledges support with an IBM Faculty award.

References
[1] Mell, P., and Grance, T. (2011) The NIST Definition of Cloud Computing 800-145 (Gaithersburg: National Institute of Standards and Technology)
[2] UberCloud Experiment, https://www.theubercloud.com/hpcexperiment/
[3] Amazon Web Services, http://aws.amazon.com
[4] – http://www.anl.gov/Mediacenter/News/2009/news090402.html
[5] Appavoo, J., U. V., Rosenberg, B., DaSilva, D., and Moreira, J. (2009). Kittyhawk: Enabling Cooperation and Competition in a Global Shared Computational System. IBM Journal of Research and Development, vol 53 no 4 41-15.
[6] http://kittyhawk.bu.edu/kittyhawk/Kittyhawk.html
[7] Dreher P., Scullin W., Vouk M., (2014) Toward a Proof of Concept Implementation of a Cloud Infrastructure on the Blue Gene/Q, International Journal of Grid and High Performance Computing (submitted)
[8] Dreher, P., and Mathew, G. (2014) Toward Implementation of a Software Defined Cloud on a Supercomputer IEEE International Conference on Cloud Engineering (IC2E) IEEE Computer Society Conference Publishing Services, 10.1109/IC2E.2014.57, 580-585
[9] G. Mathew, P. Dreher, M. Vouk, “Integrating VCL into a Re-Configurable Cloud Computing Layer on an IBM BG/P Supercomputer”, (2014) 2nd International IBM Cloud Academy Conference, Atlanta, GA
[10] Vouk, M. (2008) Cloud Computing - Issues, Research and Implementations Journal of Computing and Information Technology, vol 16 no 4 235-246
[11] Vouk, M., Rindos, A., Averitt, S., Bass, J., Bugaev, M., Peeler, A., and Valenzisi, V. (2009). Using VCL Technology to Implement Distributed Reconfigurable Data Centers and Computational Services for Educational Institutions. IBM Journal of Research and Development, vol 53 no 4 1-18
[12] Dreher, P., Vouk, M., Sills, E., and Averitt, S. (2009). Evidence for a Cost Effective Cloud Computing Implementation Based Upon the NC State Virtual Computing Laboratory Model Advances in Parallel Computing; High Speed and Large Scale Scientific Computing 18 236-250. Amsterdam: IOS Press
[13] Schaffer, H., Averitt, S., Hoit, M., Peeler, A., Sills, E., and Vouk, M. (2009). NCSUs Virtual Computing Laboratory: A Cloud Computing Solution. IEEE Computer Society vol 42 no 7 94-97
[14] Apache VCL, http://vcl.apache.org, alsohttp://vcl.ncsu.edu
[15] DeGrand, T. and DeTar, C. (2006) Lattice methods for Quantum Chromodynamics (London: World Scientific)
[16] http://physics.indiana.edu/ sg/milc.html
[17] Blue Gene/Q: By Co-design, (2013). Computer Science - Research and development, vol 28, no 2-3, 127-135
[18] IBM Research, 2014, FusedOS, GitHub repository, https://github.com/ibm-research/fusedos
[19] Park, Y., Van Hensbergen, E., Hillenbrand, M., Inglett, T., Rosenberg, B., Ryu, K., Wisniewski, R., (2012). FusedOS: Fusing LWK Performance with FWK Functionality in a Heterogeneous Environment, Proceedings of the 2012 IEEE 24th International Symposium on Computer Architecture and High Performance Computing, 8, 211-218
[20] Park, Y., Hensbergen, E.V., Hillenbrand, M., Inglett, T., Rosenberg, B.S., Ryu, K.D., Wisniewski, R.W.: (2012) Poster: FusedOS: A Hybrid Approach to Exascale Operating Systems High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion: 1417