ORNL/TM-2023/3083

Evaluating the Cloud for Capability Class Leadership Workloads

Jack Lange, Thomas Papatheodore, Todd Thomas, Chad Effler, Aaron Haun, Carlos Cunningham, Kyle Fenske, Rafael Ferreira da Silva, Ketan Maheshwari, Junqi Yin, Sajal Dash, Markus Eisenbach, Nick Hagerty, Balint Joo, John Holmen, Matthew Norman, Dan Dietz, Tom Beck, Sarp Oral, Scott Atchley, Phil Roth

September 11, 2023

OAK RIDGE National Laboratory

ORNL IS MANAGED BY UT-BATTELLE LLC FOR THE US DEPARTMENT OF ENERGY

ORNL/TM-2023/3083

National Center for Computational Sciences, Oak Ridge Leadership Computing Facility

Evaluating the Cloud for Capability Class Leadership Workloads

Jack Lange, Thomas Papatheodore, Todd Thomas, Chad Effler, Aaron Haun, Carlos Cunningham, Kyle Fenske, Rafael Ferreira da Silva, Ketan Maheshwari, Junqi Yin, Sajal Dash, Markus Eisenbach, Nick Hagerty, Balint Joo, John Holmen, Matthew Norman, Dan Dietz, Tom Beck, Sarp Oral, Scott Atchley, Phil Roth

September 11, 2023

Prepared by
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, TN 37831-6283
managed by
UT-Battelle LLC
for the
US DEPARTMENT OF ENERGY
under contract DE-AC05-00OR22725

EXECUTIVE SUMMARY

Cloud platforms offer a variety of benefits that are very appealing for a large scale HPC facility with a diverse and dynamic user base and workload set. At the same time, there is cause for concern about transitioning to the cloud. Incorporating cloud resources into existing HPC facilities or even fully transitioning to a cloud deployment poses significant challenges at the technical, organizational, and economic levels. Regardless, based on current trends it is highly likely that cloud platforms will become an integral component of many HPC centers in some form. To gain a better understanding of both the limitations and capabilities of current cloud infrastructures we evaluated the public offerings of the three leading cloud platforms (Amazon Web Services, Microsoft Azure, and Google Cloud Platform) using a selection of representative application workloads from our facility. Our findings show that while current HPC offerings are still nascent, significant progress is being made to address the present shortcomings. At the same time, significant challenges and questions remain about whether HPC cloud offerings will be able to deliver the full range of expected benefits.

ACKNOWLEDGEMENTS

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

CONTENTS

1. INTRODUCTION

Cloud-based environments are becoming increasingly relevant for high performance computing (HPC) workloads. In addition to traditional general-purpose system configurations, cloud vendors are making significant investments in HPC-capable hardware platforms with custom-designed solutions in node, network, and storage architectures. To date, this investment has been driven by the surge in AI/ML workloads that are increasingly relying on system architectures traditionally deployed in the HPC space. As a result, cloud environments are fielding system configurations that compare favorably to those found in current-generation supercomputers. Cloud vendors are steadily producing more and more custom hardware architectures to better align their infrastructure with customer workloads and their ecosystem. The result of this is that many of the advancements that were previously developed by the traditional system vendors are instead being constrained to systems only available in the cloud. Furthermore, cloud environments are offering advanced hardware and software features, such as increased security capabilities (e.g. AWS Nitro [19]) that are currently unavailable in on-premise infrastructures.

In order to gain a better understanding of the capabilities offered by the cloud, we undertook an evaluation of the major cloud environments: Microsoft Azure (Azure), Amazon Web Services (AWS), and Google Cloud Platform (GCP). The goal of this evaluation was to determine the current capabilities of cloud environments, identify any significant issues that might prevent their use in deploying HPC workloads, and gain a better understanding of how the cloud environments operate. Our benchmark for comparison was the Summit supercomputer [25] maintained by the DOE Office of Science's Oak Ridge Leadership Computing Facility. In particular, we sought to determine the degree to which current cloud environments could support leadership scale use cases, where *leadership scale* can be considered as tightly-coupled, GPU-enabled workloads running on 1000s of "fat" (i.e., compute-dense) nodes. Although our evaluation was specifically focused on understanding these leadership scale workloads, we believe this paper provides a current snapshot of the cloud platforms' capabilities at all scales encountered in the HPC community. To further clarify our purpose for this evaluation, we did not want to show what the vendors could do *with sufficient time to prepare*, but rather what they can do *today*. Our intention is not to demonstratively prove whether cloud environments will ever be viable for HPC workloads, but rather to adjust expectations in the community and cut through the hype that is often found in discussions about the cloud. As such, our study was designed to provide very little lead time for the cloud vendors to prepare (1 week), and we intentionally used the publicly available services and interfaces.

Many in the HPC community are aware of the numerous benefits promised by cloud environments in the form of extreme flexibility and simplicity of operating large numbers of systems. As envisioned, the flexibility of the cloud would allow HPC centers to dynamically instantiate large-scale system configurations (consisting of 100s or even 1000s of compute nodes) based on per-application resource requirements. In addition, these resources would be available on demand, enabling the center to operate on a pay-as-you-go model where it would only have to pay for the resources actually used. Such an approach would allow an HPC center to deploy a custom system configuration tailored specifically to a given application, and only have to pay for that system while the application is actively using it. Additionally, it would allow the system configuration to dynamically adapt over time as newer hardware architectures become available, thus enabling the users to track technology advancements in near real time instead of having to wait for the next system procurement cycle. Finally, the cloud promises to empower users with more direct control over their system environments, since they would be able to be define their own dedicated environment that could be dynamically instantiated onto cloud resources as needed. In total, the vision offered by the cloud is one of empowered users being able to deploy workloads onto dynamically configured hardware and software platforms optimized specifically for them without having to purchase, deploy, and operate the systems themselves.

The cloud model has also become appealing to DOE leadership computing facilities for a number of reasons. First, hardware performance gains are rapidly diminishing as vendors are hitting scaling limits in the fabrication processes and underlying physics. As a result, node architectures are starting to move towards more specialized components that support increasingly narrow sets of workload classes. At the same time, workload behaviors are becoming more diverse due to the introduction of AI/ML workloads and algorithmic changes to maintain performance curves. It is entirely possible that it will no longer be economically feasible to procure and deploy a single system architecture that is capable of supporting the entirety of the current and future leadership class application set. Second, the cloud model would potentially allow leadership facilities to be more aggressive in their procurement decisions as they could effectively try out a system configuration on a temporary basis without a long term financial commitment. This flexibility would enable HPC centers to more nimbly navigate the increasingly complex and dynamic HPC landscape and more proactively adapt hardware architectures to their current workload demands.

While the cloud does hold considerable promise to address many of the challenges facing HPC centers in the post exascale era, it is important to separate the realities from the hype. Operating leadership class computing resources is a significant challenge, and one that the cloud environments are only just starting to take on. Additionally, the leadership class HPC use case presents a number of obstacles to the current technical approaches, business models, and economic considerations that the cloud providers have designed their environments around. Adapting to this new usage model will require trade-offs to be made that will diminish (and in some cases effectively negate) the advantages that make the cloud appealing to begin with. As a result of our evaluation, we claim that cloud infrastructures are not *currently* capable of supporting a leadership class HPC use case. This is not to say that the cloud will never be capable of supporting this use case, but rather that there are significant obstacles that will need to be addressed before their use as a leadership class resource could become feasible. While our results are preliminary, and a much more comprehensive evaluation is necessary, we have nevertheless identified a number of significant issues in the technical, organizational, and economic spaces that cloud vendors would need to address in order to achieve viability as a leadership capable platform.

To our knowledge this is the first attempt to evaluate cloud platforms as a leadership scale HPC environment focusing on capability class workloads. Our specific goal was to evaluate the maximum performance and scaling potential that a customer could reasonably expect from a modern cloud environment. Our customer persona was that of an ordinary user accessing the public cloud without any special consideration or dispensation by the vendors. While this approach does not provide an accurate view of the environment a large customer could achieve with a long term contractual obligation, the intention was to develop a baseline for expectations from which such an arrangement could be negotiated. As a side effect of this effort we were also able to provide more direct contributions to the research space, including:

1. The first evaluation of GPU node architectures for leadership class supercomputing workloads.
2. The largest scale evaluation of AWS Graviton CPUs using a tightly coupled parallel workload.
3. The identification of several bugs and functionality gaps in the infrastructures themselves, resulting in both upstream bug fixes and internal guidance for future product roadmaps. This included a bug in

the AWS libfabric codebase, non-optimal configuration defaults across each environment, and recommended improvements to management infrastructures.

2. RELATED WORK

Much has been said about HPC in the cloud [18], and numerous studies have been conducted across both industry, academic, and government HPC facilities [10, 9, 23, 14]. Notable among these are the Magellan Project [6, 15, 22] a DOE funded study from Lawrence Berekeley National Laboratory (LBNL), the NASA evaluation of commercial clouds for HPC applications [3], and an ongoing multi-year evaluation by the Department of Defense's HPC Modernization Program (DOD-HPCMP) [1]. While each of these studies evaluated the cloud along similar parameters as in this paper, there are several aspects to our study that are significantly different. First, each of these studies evaluated only the CPU based node configurations offered by the cloud vendors, and did not utilize GPU enabled instance types. Second, the scaling goals for each of these studies was smaller in scope to this evaluation, which was trying to compare directly to a capability class leadership supercomputer. Therefore, we believe that ours is the first study to focus on a leadership class supercomputing use case that attempts a large scale evaluation of GPU based node architectures for tightly coupled scientific applications. As a result of this, we believe our findings present a new set of data points in the understanding of cloud capabilities, performance, and economics.

3. EVALUATION DESIGN

The primary focus of this study is to measure the achievable scale currently available from each of the vendors. While a full evaluation would require a significant undertaking and focus on many dimensions of the cloud platforms' offerings, this effort focused only on a small number of basic metrics to develop a general overview of the potential provided by cloud based HPC architectures. To accomplish this, the study selected 5 applications that were deemed strategically interesting (WfBench, StemDL, Lattice QCD, LSMS, and 3D Cloud model) and deployed them across the three cloud providers. The evaluation plan called for measuring the performance of each application at exponentially increasing scale within the constraints of a fixed budget and a fixed deadline. Due to these constraints, this study should not be considered as an accurate view of the ultimate capabilities of each environment, but instead an initial effort to establish a baseline of expectations when considering the capabilities of the various cloud vendors, as well as to demonstrate the disparities between commodity and HPC resources available in a cloud environment.

Due to the focus of this evaluation effort on scalability of computing resources, the study purposefully did not analyze or evaluate other significant system components that are required in a full HPC environment. Most notably this study *did not* evaluate I/O or storage capabilities of the vendors, and it either selected or configured the application set to minimize or omit any I/O operations during their execution. The study also intentionally *did not* perform extensive tuning or performance optimization of either the applications or the cloud environments themselves due to both budget and time constraints. Therefore, when evaluating the results of this study, it is important to consider that the absolute performance numbers should not be considered fully representative of what is likely achievable in the cloud. Instead, they should be taken as rough estimates to indicate general performance capabilities as well as indications of potentially broader issues that will likely need to be addressed in the future.

Finally, it should be noted that this study was conducted by a team with limited, and in some cases no, prior experience with the environments they were working with. Among the application teams and HPC Center staff members, few had previous experience deploying application codes in cloud environments. The fact that the team was able to set up and evaluate the three environments in the given timespan demonstrates the capabilities of the cloud environments and their in house support staff to enable fast setup and configuration of their respective infrastructures. It must be pointed out that this study would not have been possible without direct support from each vendor in the form of guidance, troubleshooting, and resource capacity allocations.

3.1 EVALUATION PLAN

Our initial evaluation goal was to measure both the performance and cost of each cloud environment as we scaled applications from 16 to 4,096 nodes. However, it quickly became apparent that this was not a realistic goal due to a number of reasons including availability of cloud resources, insufficient budget allocations, and the fact that several of our applications were not designed to scale to those limits. We therefore had to adjust our expectations, and came up with the following evaluation plan incorporating these constraints:

One quarter of the total budget was allocated for each vendor to execute all 5 applications at increasing levels of scale. Each application would start scaling at 16 nodes and increase in powers of two until one of the following conditions was reached:

1. The application reached the 4,096 node scale
2. Sufficient resources could not be provisioned from the cloud environment
3. The budget was exhausted
4. The application reached its scalability ceiling

An additional quarter of the budget was kept in reserve to cover infrastructure costs and debugging/testing activities.

3.2 APPLICATION WORKLOADS

Overview The application workloads chosen for this study include a wide range of science domains and functionalities that span deep learning algorithms, the generation of workflow benchmarks, and large-scale scientific modeling and simulation codes. This section presents brief summaries of the five applications chosen to explore the capabilities of the three cloud HPC providers.

WfBench WfBench [7] generates realistic scientific workflow benchmark specifications that can be translated into benchmark code to be executed with workflow management systems. Specifically, WfBench generates workflow tasks with arbitrary performance characteristics regarding CPU, memory, and I/O usage, and the code also generates realistic task dependency structures. The benchmark's execution proceeds in three phases: (1) Read — incorporate input from disk in a single thread; (2) Compute – this phase is configured with a number of cores, a total amount of CPU work to perform, a total amount of memory work to perform, and the fraction of the computation's instructions that correspond to non-memory operations; and (3) Write — insert output into a file in a single thread. The compute phase starts groups of threads which are pinned to the same CPU core. Within each group, the threads run a memory-intensive executable; the remaining threads run a CPU-intensive executable (compiled C++) that calculates an increasingly precise value of π until the specified total amount of computation has been performed. WfBench leverages the WfChef open source tool [8]. WfChef analyzes the task graphs to produce a "workflow recipe". Here, WfBench is used to evaluate the performance when scaling to compute 100,000 CPU- and memory-intensive tasks. The code is both memory and CPU intensive.

STEMDL State of the art scanning transmission electron microscopes (STEM) produce focused electron beams with atomic dimensions that yield diffraction patterns for nanoscale material volumes. Backing out the local atomic structure of the materials requires compute- and time-intensive analyses (known as convergent beam electron diffraction, CBED). Reconstruction of a material's local electron density with atomic resolution is a longstanding inverse problem lacking a general solution. Under-determination results because the information needed to directly invert the forward model is always missing. This issue is known as the phase problem. STEMDL [17, 24] is a deep learning application based on fully-convolutional dense neural networks (FC-DenseNet) that reconstructs the local electron density from microscopic diffraction images. It is trained on the CBED patterns of over 60,000 solid-state materials (about 0.5 PB of data) and is capable of an atomically-accurate reconstruction of materials. Computationally, it introduces a new technique to overlap communication with computation, and near-linear scaling (93%) on the entire Summit supercomputer is achieved. A peak performance of 2.15 EFLOPS is observed. The primary computational constraints are the network bandwidth for passing gradients in the deep neural network between nodes and the file system performance for random-read processes.

LSMS LSMS [26, 13, 16] is a first principles electronic structure code for calculations in Materials Science and Condensed Matter Physics. LSMS solves the Schrödinger or Dirac equation for the electrons using a Density Functional Theory description with a multiple scattering theory approach. Linear scaling with system size is achieved in the LSMS by using unique properties of the Green's function. The compact nature of the information that needs to be passed between processors and the high efficiency of the dense linear algebra algorithms employed are responsible for the superior performance. The code relies heavily on numerical libraries for linear algebra and displays near-perfect weak scaling. Greater than 90% of all floating point operations are concentrated in the calculation of the scattering path matrix. This is dominated by the matrix inversion of a double precision complex matrix. A recent calculation was performed on a million-atom iron/platinum system on the Frontier exascale supercomputer. A key constraint is the memory bandwidth during the dense linear algebra operations.

LatticeQCD Lattice Quantum Chromodynamics (Lattice QCD or LQCD) is a framework in theoretical physics to calculate the interactions of elementary particles. The computations consist of Monte-Carlo evaluations of Feynman Path-Integrals. Calculations are split into three parts: 1) the generation of ensembles of configurations 2) the computation of the correlation functions and 3) the extraction of the physical information. Here, we focus on 1) since it is the most computationally demanding. The multigrid (MG) portion of the algorithm can produce latency bottlenecks. We will list the average performance of our solvers in GFLOPS. We will list the performance of the multishift conjugate gradients (MCG) algorithm and the MG algorithm for the lightest and heaviest quark masses. Our LatticeQCD benchmarks utilize the Chroma code [12] with GPU acceleration of the necessary linear solvers that utilize the QUDA Library [4, 2] with the multigrid solver implementation described in [5]. To accelerate the non-solver components we rely on the QDP-JIT implementation of QDP++ described in [27]. The code is both memory and network bandwidth bound.

3D-Cloud The 3-D cloud model miniWeatherML application [21, 20] is a software tool to examine atmospheric flows that includes moisture in the forms of vapor, cloud, and precipitation. The application solves the inviscid, stratified, compressible Euler equations that govern atmospheric dynamics on a 3-D Cartesian domain with regular grid spacing. The discretization scheme uses a high-order Finite-Volume method (with third-order accuracy) in space and a Runge-Kutta Ordinary Differential Equation (ODE) solver in time. The domain is decomposed in the two horizontal dimensions, as the vertical dimension contains tighter coupling and loop-carried dependencies that make decomposition difficult. The application uses a stencil-based approach rather than global FFT or sparse linear algebra solves. Further, a strong scaling approach is important for this application (rather than weak scaling) because weather and climate models must simulate with fast throughputs. A result is that parallel overheads often dominate and available, on-node threading for accelerators is smaller than optimal. The algorithms are to a great extent memory bandwidth limited. They are also subject to latency requirements, since one must hop through multiple nodes to transfer data to neighboring MPI tasks with nearest-neighbor communication.

Budget Limitations

The primary constraint placed on this study was the fixed budget available to pay for the cloud resources. We intentionally decided to use the on-demand pricing model for this study and as a result had to pay substantially higher costs than we otherwise would have. This meant that the initial scaling goal of 4,096 nodes was immediately deemed impossible due to the lack of budget that would be required, since the cost of a study that reached 4,096 node scale would easily exceed $1 million for each cloud environment. Furthermore, even at smaller scales the cost of the compute resources could easily result in dramatic budget overruns. This ended up being a continual source of concern throughout the evaluation due to limited budget controls provided by the cloud environments, the relatively small size of the budget, and the sheer speed at which costs could be incurred if a mistake was made.

In order to maintain control over the budget, we put together an evaluation plan for each application that provided a guide for their scaling runs. This plan was based on runtime estimates collected from the application teams and the hourly rate they would be paying for each compute node. Using this information we were able to calculate rough projections of the costs incurred by each application as they ran at increasing node counts. This allowed us to place a strict limit on the scale each application would be able to reach while still remaining within the budgetary constraints.

The projections were determined by taking the expected runtime of the application plus some additional headroom and then adding in infrastructure initialization times (based on what we had seen during the initial setup phase) to get the total runtime for each evaluation run. The total time was then multiplied by the hourly rate for the instance type the application would be targeting and the total number of node instances used.

$$Cost = (conf\_time + runtime) * scale * rate \tag{1}$$

Due to space constraints, we are unable to provide a full breakdown of our projected costs, but instead in Table 1 provide an example for one application (LSMS) running on one cloud environment (AWS). As can be seen, the budget allocation only allowed a very limited range of scale using the public pricing models. However, as we will explain in Section 4., even these limited scaling goals proved to be overly optimistic relative to the actual availability of HPC nodes in the cloud.

Scale	Conf. time (minutes)	Runtime (minutes)	Rate ($/hour)	Cum. Cost
16	8	30	39.33	$398.54
32	8	30	39.33	$1,195.63
64	8	30	39.33	$2,789.81
128	8	30	39.33	$5,978.16
256	8	30	39.33	$12,354.86
512	8	30	39.33	$25,108.27
1,024	8	30	39.33	$50,615.09
2,048	8	30	39.33	$101,628.72
4,096	8	30	39.33	$203,655.98

Table 1. Cumulative cost breakdown for LSMS running on AWS. The expected runtime is 30 minutes with 8 minutes predicted for node setup time. The projected cost for the full scaling run of LSMS on one cloud environment would exceed the budget for the entire study. Rows in grey are scales we not did attempt due to budget constraints.

3.3 CLOUD ENVIRONMENTS

The focus of our evaluation is on three of the established cloud vendors currently offering HPC-capable environments: Amazon Web Services, Microsoft Azure, and Google Cloud Platform. For all three cloud platforms, we configured a batch-scheduled cluster environment that would be similar to one found on a standard HPC system in a DOE HPC Facility. All three cloud platforms were configured with the same general setup; Slurm for job scheduling and dispatch, a login node as a point of entry into the cluster, and a file system with user areas and a shared folder for installing libraries and packages as needed. On all cloud platforms, the Slurm scheduler was configured to allow for the dynamic spinning up and shutting down of computing resources as jobs were submitted and completed.

Amazon Web Services (AWS) The AWS environment was configured using ParallelCluster, AWS's open-source Python3-based configuration and cluster management tool, which uses CloudFormation templates to set up and deploy HPC environments. For the evaluation of this work, we created a cluster with two Slurm partitions: one CPU-only partition to support the WFbench application and a GPU partition to support the GPU-enabled applications.

The nodes in the GPU partition were equipped with dual-socket 24-core Intel Xeon CPUs and 8 NVIDIA A100-40GB GPUs, which are fully-connected with NVIDIA's NVLink, giving peak uni-directional bandwidths of up to 300 GB/s between GPUs. These nodes were connected with 4 100 Gb/s NICs based on AWS's custom Elastic Fabric Adapter (EFA) architecture. This partition was located in the GovCloud us-west-1 region. The nodes in the CPU partition consisted of a 64-core AWS Graviton2 CPU with access to 512 GB of memory and a single 25 Gb/s ENA NIC. This partition was located in the us-east-1 region. Both of these partitions, as well as the login node, mounted a shared NFS file system, which was configured with a 35 GB root and 3 TB */home* partition. While AWS does offer a LustreFX solution for storage, we opted not to use it because I/O performance was not within the scope of our evaluation.

Microsoft Azure The Azure environment was set up and configured using the Azure Portal and CycleCloud, Microsoft's toolkit for deploying and managing clusters on Azure. The configuration and deployment of the environment was managed by Microsoft support staff. We were provided two clusters, a CPU-only cluster for evaluating WfBench and a GPU-enabled cluster for all the other applications.

The nodes in the GPU cluster consisted of 2 48-core AMD EPYC CPUs, 1.9 TB of DRAM, 8 NVIDIA A100-80GB GPUs, and 4 TB of node-local NVMe storage. The GPUs are fully connected with NVIDIA's NVLink, giving a uni-directional bandwidth of 300 GB/s between GPUs. Each node has 8 200 Gb/s NICs for a total of 200 GB/s of inter-node bandwidth. The GPU cluster was deployed in the East US region. The CPU-only nodes were equipped with a 120-core AMD EPYC CPU, 456 GB of memory, and a single 200 Gb/s Infiniband NIC, and was deployed in the South Central US region. Both clusters mounted a shared NFS file system that was mapped by CycleCloud.

Google Cloud Platform (GCP) The GCP environment was set up using Google's open-source HPC Toolkit, which uses YAML description files that generate Terraform scripts to deploy an HPC cluster. The cluster was deployed in the us-central-1 region. As in the other environments, we used two Slurm partitions to provide CPU-only and GPU-capable node configurations. Storage was provided through a shared fileystem mounted from the login node using NFS.

We originally intended to evaluate GCP's A100 GPU nodes, similar to AWS and Azure, but high demand for these nodes required us to instead evaluate GCP's more-available V100 nodes. So the GPU nodes we evaluated had a single 32-core Intel CPU, 256 GB of memory, 8 NVIDIA V100 GPUs arranged in a ring topology, and a 12.5 GB/s NIC. These nodes were not intended for HPC workloads but they allowed us to evaluate GCP alongside the other cloud platforms. The CPU-only nodes were equipped with a 56-core AMD EPYC processor and 896 GB of memory.

4. CLOUD CAPABILITIES AND LIMITATIONS

This evaluation effort should be viewed as an initial step in further exploring and understanding the capabilities of the cloud environment. The results we collected from the scaling study should serve as a baseline to calibrate expectations of cloud environments instead of an ultimate measure of their capabilities. Indeed, the results of this evaluation should be viewed in the context of the timeframe and manner under which it was conducted, and be understood as not representing the general HPC capabilities of each environment. Nevertheless, while the results from this evaluation effort were mixed, there are some important conclusions and observations that can be derived from the experience.

Currently, the largest barrier to adopting public cloud platforms as a large scale HPC environment is the lack of readily available computing resources. When trying to allocate nodes comparable to those in a leadership class system, we were given explicit scale limits by the vendors which we would not be able to exceed. The scaling limits given to us were:

1. AWS: 32 A100 GPU nodes, and 128 x86 CPU nodes, and 256 Graviton2 CPU nodes
2. Azure: Initially, 32 A100 GPU nodes and 128 CPU nodes, however Azure worked quickly to obtain a larger capacity reservation of 256 GPU nodes, and 512 CPU nodes. In practice, even with a reservation in place, scaling to 128 nodes proved difficult due to capacity constraints.
3. GCP: 32 A100 GPU nodes and 128 x86 CPU nodes. However in practice it was not possible to allocate a single A100 node, and so we instead switched to V100 nodes with the same 32 node limit.

There are several reasons for this lack of available capacity. First, the demand for GPU accelerators is significant due to the rapid advancement of AI/ML workloads and their requirements for large scale training infrastructure. Second, the business model of the cloud does not align to support readily available large scale HPC infrastructures, as every cloud vendor optimizes their profit margins by maintaining minimal spare capacity as a cost saving measure. This means that if a large (i.e. several hundred node) allocation is requested, there are often not enough resources available to satisfy it. There are alternative business models designed to address the needs of customers requiring large scale, however these agreements generally require that customers promise to pay for a set amount of resource utilization (whether it is used or not) in order to maintain the spare capacity to satisfy those customers' requests. Furthermore, most of the prototype agreements we have seen appear to require fairly explicit node architectures, meaning that they would force the customer to commit to a static node architecture for the period of the agreement. So while these arrangements do address the capacity issues we experienced, they appear to come at the cost of placing significant constraints on the flexibility to dynamically re-target to different node configurations based on workload requirements. This diminishes the perceived flexibility of the cloud since it requires that organizations develop capacity requirements and system specifications based on projected workload demands, i.e. a very similar process to what is currently required for an on-prem system procurement. While the cloud will certainly exhibit a greater degree of flexibility over current on-prem systems, that level of flexibility *will almost certainly be less than* that which is often assumed.

4.1 TECHNICAL ISSUES

Semantic Gap The other large issue we discovered during this exercise was the existence of a semantic gap between a customer's HPC environment and the underlying cloud infrastructure. This gap exists because the HPC environment is an abstraction layer over the underlying cloud infrastructure. This semantic gap introduces a number of issues that currently impact the suitability of the cloud for large scale HPC deployments. The main problem is that there is an opaque barrier between the vendor's infrastructure and the user's environment, so neither the vendor nor the user has visibility into what is happening on the other side. This introduced a number of challenges such as the inability to effectively manage the budget across the collection of users, the inability of both the vendor and our evaluation team members to independently debug performance issues, and a general lack of control over the resources an application was deployed on. While some of these issues could likely be addressed by the vendors, the fundamental friction will likely remain and be a constant source of problems and challenges moving forward. How much of a fundamental limitation this semantic gap poses for HPC environments in the cloud is an open question.

Resource Allocation The resource allocation process in particular proved to be a constant source of problems during the evaluation. While the issues were varied, they mostly boiled down to the semantic gap caused by the abstraction of the actual hardware nodes behind an allocation interface that hid many of the physical details from the user environment. Due to the elastic nature of cloud resources, nodes are dynamically allocated only in response to jobs scheduled, meaning that the actual nodes provisioned for the HPC environment often experience a high rate of churn as they are returned when there are no pending jobs available to schedule. All of the budget excursions we experienced during the evaluation were the direct result of this. Examples of these issues include (a) incremental allocations, (b) allocating unhealthy nodes, (c) misconfigurations hidden from the users, and (d) long node startup times. Incremental allocations happen when queued jobs request more nodes than can be satisfied immediately by the infrastructure. When this occurs the job scheduler will request the full number of nodes, but the underlying node allocation interface will only allocate as many nodes as are currently available. In the case where there are not enough available nodes, the allocator will still return a subset of the allocation request to the job scheduler which will then hold those nodes in an idle state while it waits for the rest of the allocation to succeed. However, because the held nodes were successfully allocated, they will begin incurring charges with no feedback to the user that this is happening. It is also possible for the allocator to return unhealthy nodes that have performance or correctness problems. Because nodes are dynamically allocated and released, it is not possible to track these unhealthy nodes across allocations, and as a result the users must perform (and pay for) their own node health checks on every node returned from the allocator. We also experienced an issue with node misconfigurations where allocated nodes reported a hardware configuration that did not match what was expected by the job scheduler. This resulted in an infinite allocation/deallocation loop as nodes were repeatedly allocated and immediately discarded because of a memory size mismatch. This was not a minor problem either, since the configuration check was performed after the node had started incurring charges and the entire process was completely hidden from the users who simply saw their jobs stall in a configuring state on the queue. Finally, we noticed that node initialization often took a considerable amount of time (often exceeding several minutes) to fully install and configure the packages needed. We expect that these times would likely be substantially longer if we had been running a more realistic environment with significantly more software packages on the node. Again it must be noted that the install and configuration process is done after the node has been allocated, so the time is charged to the customer. Overall we believe that the cumulative cost of these issues resulted in $\sim \$50,000$ spent on idle nodes, most of which was ultimately reimbursed by the vendors.

Budget Controls Another source of difficulty during the evaluation was the lack of budget controls that could provide effective limits at a per user or per application granularity. This again is due to the semantic gap, since the billing infrastructure and budgetary controls are implemented at the underlying cloud infrastructure level and has no visibility into the HPC environment running on top. As a result, all of the budget controls had no knowledge of our users or their individual jobs; the controls only saw node allocation requests without any additional context. The coarse grained protections we could enable only operated at the cluster level. This meant that we were protected from a single user exceeding the total project budget, but were unable to prevent a single user from exhausting the budget and starving the other users on the system.

Performance Problems Another issue we encountered was the inability for us to independently debug and troubleshoot performance problems. While a significant part of this was due to our unfamiliarity with the environment, there was an inherent limitation we faced in lack of visibility into the underlying infrastructure. Experience with operating large scale HPC resources has demonstrated to us the necessity in being able to monitor and inspect the underlying system behavior in order to locate and mitigate performance problems. However, in the cloud this level of visibility is not available due to the abstraction layers between the customer and the underlying infrastructure. As a result we found that debugging performance issues required a collaborative effort between our evaluation team and vendor support staff. While this is not unique to cloud environments, the degree of separation between the underlying infrastructure and the user environment did pose a significant hurdle.

Network Architectures As described in Section 3.3, the node level architectures across the different cloud environments are roughly similar to each other as well as being comparable to the node architectures currently being deployed on DOE leadership class supercomputers. There is, however, significant divergence in the network architectures and performance across the various vendors and DOE systems. In addition, network architectures are a significant point of focus for the cloud vendors, and something they clearly see as a source of competitive advantage. The most serious constraint is the scalability limit of modern cloud network topologies, which do not currently scale past ~1000 endpoints. Considering that many leadership workloads regularly scale into the thousands of nodes, this is a significant limitation. In addition the node injection bandwidth offered by each vendor varied by a significant margin (50GB/s for AWS, 100 GB/s for GCP, and 200 GB/s for Azure) and each network utilizes different underlying protocols and architectures, much of which is proprietary. Finally, it should be noted that only Azure fully supported GPUDirect enabled MPI in its production environment. We were able to get access to a pre-release version of GPUDirect enabled MPI from AWS, though we were only able to successfully enable it for one of our applications (which required the identification and fixing of a bug in the AWS libfabric library). While an issue for this evaluation, we expect that GPUDirect support will be available from all the cloud vendors in the near term, and mention its lack only to indicate the level of maturity of the cloud HPC environments at this point in time.

Application Environment Regarding the programming environment, our goal was to present the application teams with an environment that was similar to what they are used to on other (DOE, NSF, etc.) HPC systems: *ssh* to a login node, use module environments to manage software packages, and schedule and launch jobs using a familiar tool (e.g., Slurm). Generally speaking, that *is* what we gave them. However, there were difficulties providing some of these components. For example, it was surprising to realize that a working GPU-aware MPI (nevermind intra- and inter-node GPU direct) was not available for 2 of the 3 cloud platforms (Azure being the exception). To be fair, on GCP, "not available" simply meant we had to build it ourselves, although GCP does not have support for GPU direct RDMA under GPU-aware MPI (i.e., GPU buffers are staged through host buffers under the hood for inter-node MPI communication). On the other hand, AWS does support both intra- and inter-node GPU direct for GPU-aware MPI, but AWS support staff had to work with us to get it running due to their EFA network interface. That said, this is a "soft issue" that can be worked out over time. But it does highlight the fact that some components that we find crucial to running our tight-coupled applications is not well-tested on some of the cloud platforms.

4.2 ORGANIZATIONAL ISSUES

Beyond the technical issues we have identified there are also deeper concerns that must be considered. Supporting a leadership class HPC environment requires a deeply rooted strategic commitment inside an organization. Undertaking the deployment of a leadership class supercomputer requires substantial investments and unique engineering solutions. To do so successfully requires not just the commitment to engineer such a system, but a strategic understanding of how to convert those solutions into products suitable for the broader market. While we have no doubt that the cloud vendors are more than capable of deploying such a system today, we are less confident that they would be able to do so in a way that is sustainable in their business model. While we are in no place to comment on the internal strategic directions of the cloud vendors, we can make two observations that we believe to be somewhat indicative.

Resource Availability

Throughout the evaluation we were continually faced with resource capacity issues in the various cloud environments, and the support staff from each vendor were actively engaged in trying to allocate more capacity for our effort. While much of this effort was happening behind the scenes, we were able to make general observations about how much organizational support was available to support an evaluation of HPC capabilities. We observed a wide range of responses between the vendors, with the expected results shown in how much capacity each vendor was able to procure for our effort.

Operations Staff In addition, the impact of cloud adoption on customer organizations must also be considered. Based on our experiences during the study, we are confident in saying that the need for local operations and support staff will not disappear and will likely remain constant. This is due to the fact that while cloud platforms do alleviate some of the operations overhead they add new overheads in their place, particularly in the area of cost monitoring. In addition, cloud environments still require the customer to handle a significant amount of administrative duties, which our application users were both unprepared for and resistant to taking on themselves. Based on our experience, we believe that the majority of HPC users will prefer to view the cloud in the same way they currently view large scale supercomputing environments, i.e. a large facility which manages the system on their behalf and is available to offer direct support in the form of system maintenance and debugging.

4.3 ECONOMIC ISSUES

Finally, economic realities are something that must be considered when evaluating the cloud for leadership scale HPC. It is well known that the cloud is a lucrative business model that generates substantial economic returns, and there is a question as to whether a business case can be made for leadership class HPC in the cloud. We have significant concerns that a cost competitive leadership scale HPC environment is simply not profitable enough to sustain the business interests of the cloud providers. To demonstrate this point we can provide a comparison of the cost per node hour of comparative node architectures in the cloud versus an on-prem leadership class system installed into a pre-existing datacenter facility. While our cost analysis will not be exact it should be close enough to demonstrate the issue. For the cloud costs we will assume a node hour cost of ~ $40/hour, which will represent the cost of only the compute nodes (i.e. we will assume storage is free). We will compare that to a total cost of ownership (TCO) for a hypothetical on-prem exascale system as specified in the CORAL-2 RFP [11] and using general rules of thumb for calculating procurement costs. These costs can be broken down across both capital and operating expenditures. These include:

1. Total system purchase price of $600MM (we will assume this includes 10,000 compute nodes and a 700PB storage system)
2. Direct and indirect costs (rule-of-thumb 10% of system cost)
3. Facility upgrades (rule-of-thumb 10% of system cost)
4. Estimated power costs (rule-of-thumb $1MM/MW/year). Average CORAL-2 predictions were 40MW at peak which we will assume to be constant, so $40MM/year.

If we amortize the CapEx costs across the projected five-year lifespan of the system, the annual cost of the hypothetical system including power comes to $174MM/year. If we then assume a 90% utilization rate of the full system, this gives us a node hour rate of $2.35/hour or 5.88% the cost of a comparable cloud node. Again this includes the cost of 700PB of storage in hypothetical system, and assumes that cloud storage would be completely free. Based on our internal estimates we project that under current public pricing, the cost of cloud compute resources is closer to 17X more expensive than an on-prem leadership class system based on the public pricing information. While each cloud vendor has been very clear that significant discounts are available and price is highly negotiable, we have not yet seen any thing close to a discount rate of 95%. This raises a significant question as to whether a price competitive large scale HPC infrastructure is economically attractive (or even feasible) for the cloud vendors.

5. APPLICATION RESULTS

In this section, we will show the results from the five applications running on each cloud platform.

5.1 WFBENCH

For this evaluation, we generated a 100,000-task workflow benchmark that mimics the task graph structure of a seismology application. It is characterized by a fan-out-fan-in structure with CPU-intensive tasks that calculate a total of 50,000,000 operations. For each cloud platform, we performed a strong scaling evaluation using 2^n nodes, where $n \in [4, 8]$. Figure 1 shows the total runtime (a) and speedup (b) observed from these scaling tests. It is important to note that each cloud platform provided a different CPU node architecture so the runtime results should not be compared directly. However, because we executed exactly the same workflow benchmark on each platform, we can still draw conclusions from the speedup trends. In Figure 1b, the baseline execution (i.e., speedup = 1.0) is defined by the execution of the benchmark workflow on 16 nodes of a specific platform.

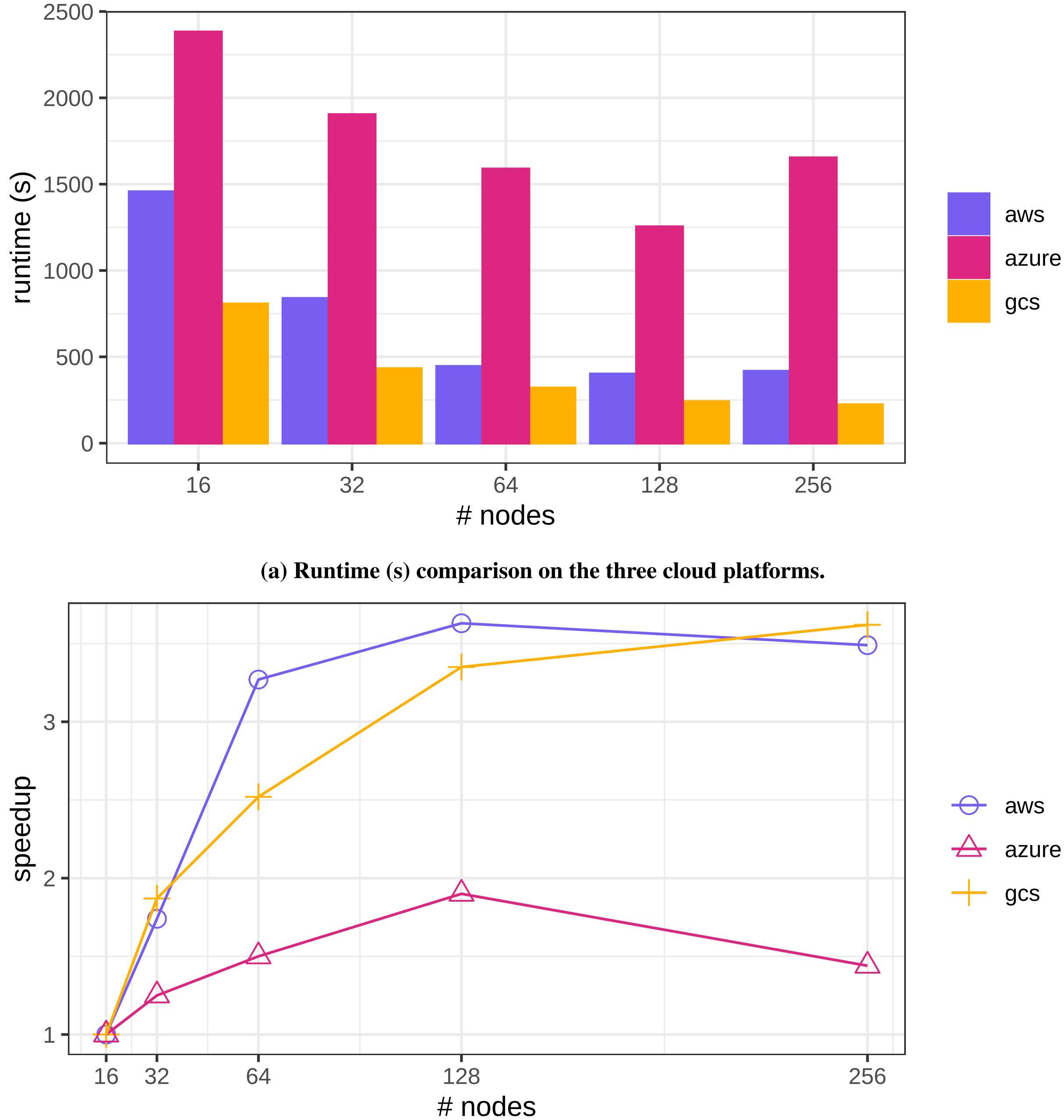
Overall, WfBench scales reasonably up to 128 nodes for all cloud platforms, but turns over beyond that for AWS and Azure - but it is unclear what specifically was responsible for the turnover. One thing to note is that the workflow tasks used in WfBench also perform memory operations (using the Linux command $stress - ng$) so memory contention became an issue as the node count increased. To circumvent the issue, we limited the maximum number of cores used per node. Table 2 shows the total number of CPU cores used in each of the runs as well as the number of cores available per node in parentheses. The lower performance on the Azure platform seemed to be related to I/O operations. Although we essentially turned off I/O, we were still creating 0 kB files and these operations showed delays during our runs on Azure nodes.

Total # of CPU Cores Used (# CPUs/Node)			
# Nodes	AWS	Azure	GCP
16	1024 (64)	1920 (120)	896 (56)
32	1792 (56)	3840 (120)	1792 (56)
64	3584 (56)	3840 (60)	2048 (32)
128	7168 (56)	3072 (24)	2048 (16)
256	8192 (32)	4096 (16)	2048 (8)

Table 2. Total number of CPU cores used for each WfBench run. The numbers in parentheses show the number of CPU cores used per node.

5.2 STEMDL

For this evaluation, we used the StemDL benchmark to perform a scaling study on each cloud platform. Figure 2 shows the results of our tests as well as those obtained on OLCF's Summit. On AWS, we were able to secure up to 32 nodes (256 GPUs) for our scaling runs. As shown in the figure, at this node count, we were about 2X faster than on Summit, which makes sense since the A100's FP16 performance is 2.5X that of the V100, but the scaling efficiency was only about 73%. This is likely due to difficulties in properly using the full potential of the network. On Azure, we were able to run up to 64 nodes (512 GPUs) and



(a) Runtime (s) comparison on the three cloud platforms.

(b) Speedup trends on the three cloud platforms.

Figure 1. Strong scaling of 100,000-task CPU-intensive WfBench seismology workflow benchmark.

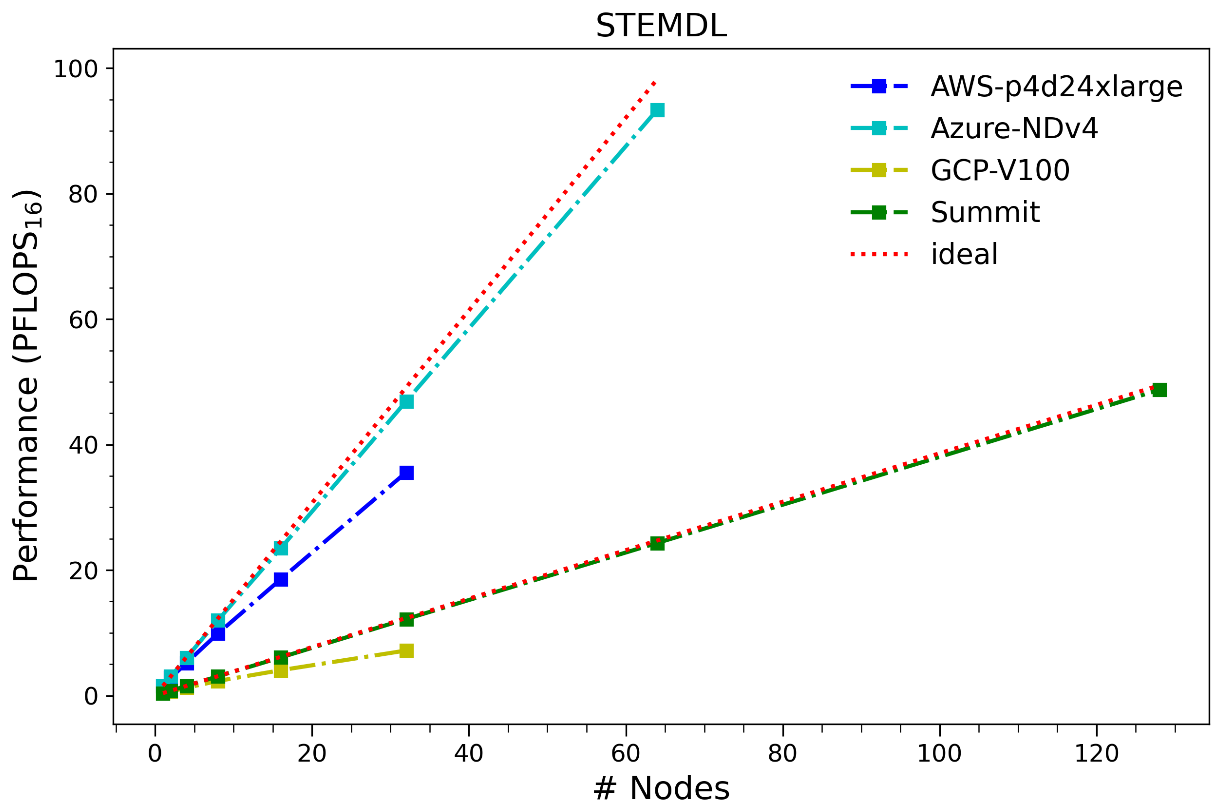


Figure 2. Scaling results for the StemDL benchmark on the cloud platforms and OLCF's Summit

found a scaling efficiency of about 95% - much more inline with our Summit results. GCP could not provide A100 GPUs so we opted to use their [8X] V100 nodes, which unsurprisingly gave lower performance than the other cloud vendors. These GCP nodes were not intended for HPC workloads (GPUs arranged in ring topology, very low inter-node bandwidth), which resulted in about 63% scaling efficiency at 32 nodes.

5.3 LSMS

The LSMS test problem used for this evaluation is a bcc Fe crystal with 48 Fe atoms per node and lmax=3. Each MPI rank targets 1 GPU, so for the cloud platforms (all with 8 GPUs per node), this means there are 8 MPI ranks per node, with 6 Fe atoms per GPU. On Summit, there are only 6 GPUs per node, so there are 6 MPI ranks per node, with 8 Fe atoms for each GPU to calculate. Figure 3 shows the weak scaling results on the three cloud platforms as well as on Summit for comparison. These tests do not use GPU-aware MPI on any platforms.

LSMS is known to scale well on many GPU-accelerated node architectures so it is not surprising to see it scale well on AWS. The AWS results are about 2.2X faster than on Summit, which makes sense given the difference in FP64 performance between the V100 and A100 GPUs, the fact each GPU on the cloud platforms needs to calculate 6 atoms instead of 8, and the improved intra- and inter-node bandwidth on AWS relative to Summit. LSMS also scaled well on the Azure platform for similar reasons as on AWS.

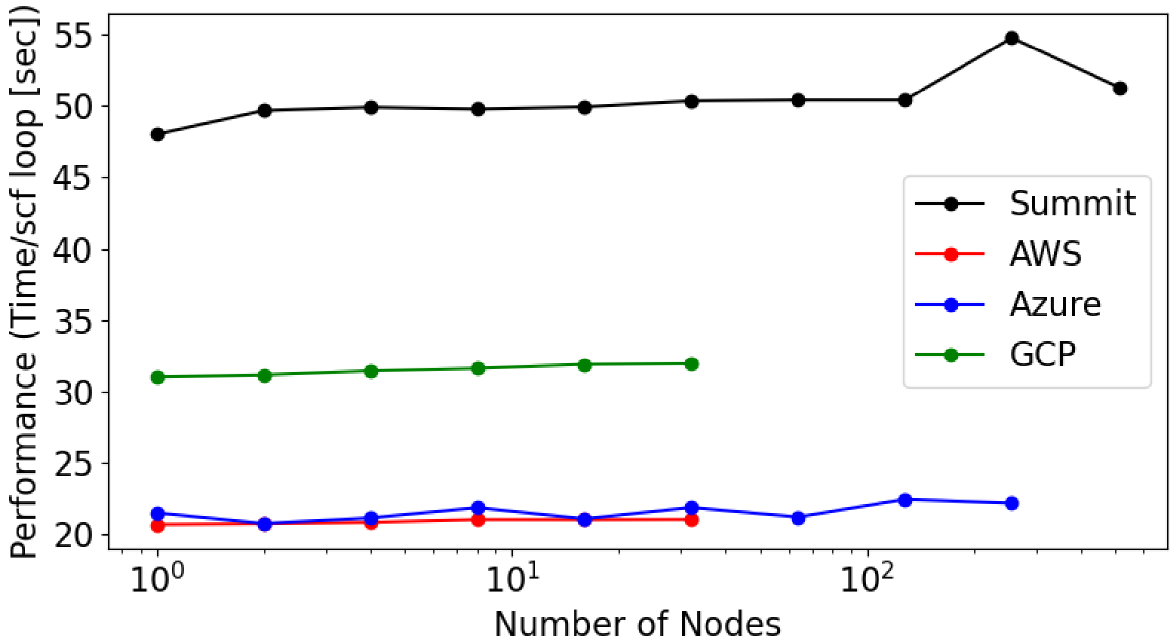


Figure 3. Weak scaling results for LSMS test on the cloud platforms and Summit.

However, we were able to secure more compute nodes on Azure, allowing us to scale out to 256 nodes (or 2048 GPUs). While LSMS did show good scaling on GCP as well, it did not perform as well as the other cloud platforms due to using [8x] V100 GPUs instead of A100s. GCP's improved performance over Summit is likely due to calculating only 6 atoms per GPU on GCP as opposed to 8 atoms per node on Summit. Similar to AWS, only 32 nodes were available for our tests on GCP.

5.4 LQCD

The LQCD benchmark used in this evaluation is a gauge generation on a lattice with volume $64^3 \times 128$ lattice sites. We time 2 hybrid Monte Carlo updates and take as our figure of merit the time for the second trajectory. The first trajectory includes some auto-tuning work which makes the second trajectory a cleaner benchmark. Figure 4 shows the strong scaling results of the benchmark on two of the cloud platforms as well as other similar systems. For our scaling runs, we were able to secure up to 32 nodes on AWS (red line in the figure) and 64 nodes on Azure (blue line). The GPU nodes on these platforms both have 8 A100 GPUs with the same intra-node bandwidth, but the Azure nodes have 4X the inter-node bandwidth, which might account for some of the difference in performance. We were unfortunately unable to run successfully on the GCP platform. All tests shown in the figure used GPU-aware MPI.

5.5 3D CLOUD MODEL

The configuration of miniWeatherML chosen for this study uses simple Kessler microphysics that evolve three forms of water: vapor, cloud, and precipitation. Three problem sizes are used in each job: (A) $1024 \times 1024 \times 100$ cells run for 900 model seconds; (B) $2048 \times 2048 \times 100$ cells run for 250 model

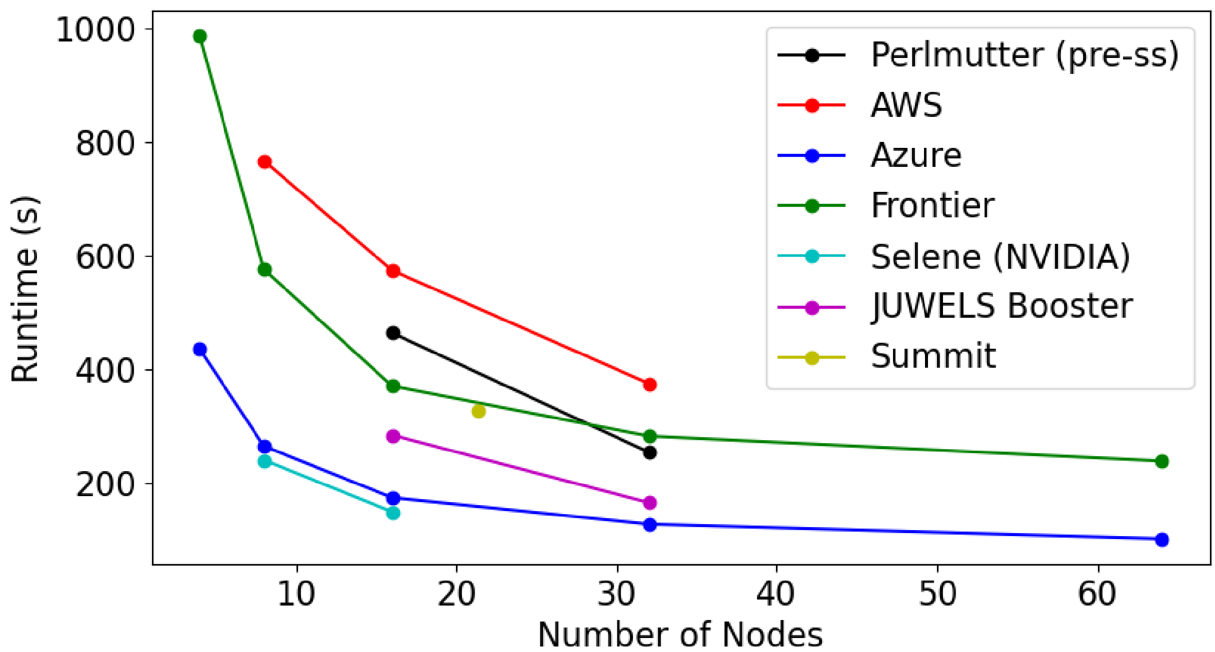


Figure 4. LQCD strong scaling results on the cloud vendors and other systems.

seconds; (C) $4096 \times 4096 \times 100$ cells run for 100 model seconds. Figure 5 shows the strong scaling results obtained for the (A) and (C) configurations of the mini-app on the cloud vendors, Summit, and Frontier.

Before further considering the results, we note that AWS and GCP were unable to provide a GPU-aware MPI implementation during our time on their systems, so we had to stage MPI data transfers through host buffers instead of sending directly from GPU-to-GPU on their clusters. These two vendors were also only able to provide up to 32 nodes for our study, which is why we do not have data for these vendors at higher node counts. Unsurprisingly, GCP shows the lowest performance since their nodes used V100 GPUs (instead of A100), their MPI implementation was not GPU-aware, their intra-node GPU topology (on these specific nodes) was arranged as a ring, and their inter-node bandwidth was relatively low. On the other hand, on Azure, we found runtimes comparable with Summit for the smaller problem size (A) and runtimes that outperformed Summit when more on-node computation was available (B). Because Azure was able to provide more compute nodes, we were also able to observe scaling trends comparable with the OLCF systems out to 256 nodes. Although the node architecture itself is very similar on AWS and Azure, the performance on AWS was lower than Azure, likely due to the lack of a working GPU-aware MPI and a quarter of the inter-node bandwidth.

6. OBSERVATIONS AND CONCLUSIONS

What is clear from our evaluation effort is that the cloud was not originally designed to be an HPC capable environment. Both current and future cloud HPC capabilities are, and will be, implemented as abstraction layers on top of a system architecture that is fundamentally different from what we are used to thinking of as an HPC system. The fundamental mismatch comes from how the cloud vendors conceptualize their infrastructure. The DOE labs have long understood that an HPC system needs to be considered as a single

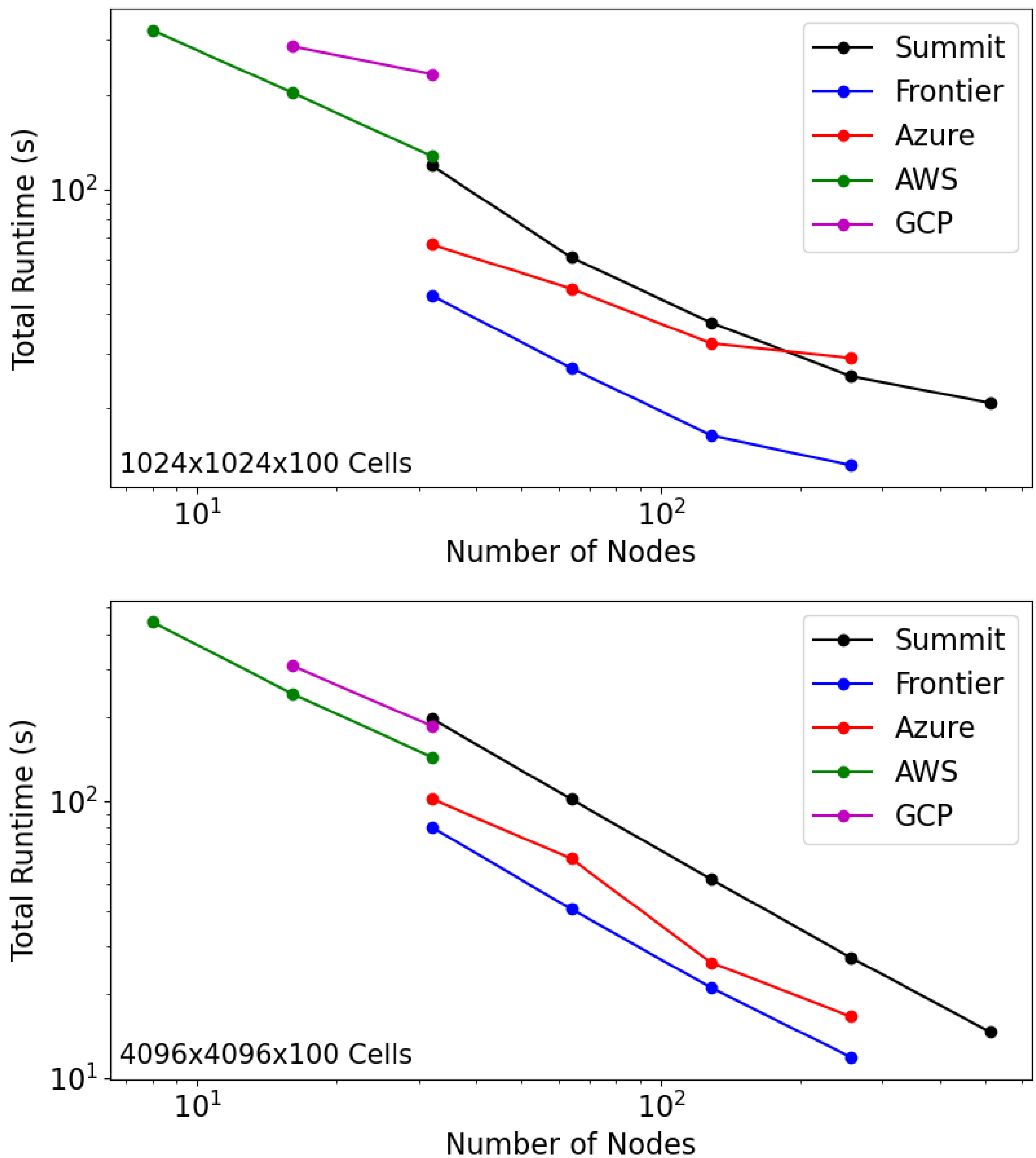


Figure 5. 3D Cloud Model strong scaling results on the cloud vendors, Summit, and Frontier.

holistic unit that is more than merely the sum of its parts. When designing and operating such a system it is the integration of the components that matter, not merely the number of components present. In contrast, the cloud environments have historically been designed as a distributed set of interchangeable resources, that are managed as independent units that can be loosely organized into a larger system architecture. Therefore, while the cloud vendors are more than capable of aggregating the most recent and advanced node architectures into the same physical location, they currently lack the ability to fully integrate those components into a single leadership scale system configuration; a shortcoming that becomes apparent at large and even medium scale. Whether this limitation can be overcome by the vendors in the future is very much an open question, and one we cannot answer at this time. However, we can say that modern cloud capabilities are most likely unable to effectively support moderate to large scale HPC environments as the DOE is used to thinking of them.

That said, there were strong points where the cloud providers were able to shine during our evaluation. One example is the significant flexibility in the hardware that can be targeted. Indeed, at several points during this evaluation, we were able to re-target the underlying hardware infrastructure extremely quickly. As described previously, we were able to switch both GPU architectures (NVIDIA A100s to V100s) and CPU architectures (x86 to ARM based Graviton2's) quite rapidly. Moreover, these changes did not need to be permanent, meaning that we could "try out" a new architecture without fully committing to it. This is a very attractive characteristic for the DOE labs as well as any organization that provides HPC resources to a user community; e.g., testing new architectures, upgrading to new hardware, and allowing users to target architectures that are most efficient for their workloads. Another point to make is that some of our application teams were able to run on up to 256 nodes. While this is not the leadership scale that the DOE labs are charged with delivering, and although this scale was limited to Azure during our evaluation, it shows what scales are possible for tightly-coupled, GPU-enabled applications in the cloud right now. There are certainly many universities and other organizations that run clusters of this size.

In conclusion, HPC in the cloud is currently in a nascent stage and must address numerous challenges before becoming a viable platform for leadership scale HPC. We believe many of the technical issues are solvable, and most will be resolved as the platforms mature. However, the fundamental economics pose a significant challenge, and it is unclear whether the cloud platforms will be able to deliver leadership scale HPC environments that are cost competitive with on-prem system procurements.

Bibliography

[1] Report to Congress on Use of Commercial Cloud to Support the High-Performance Computing Needs of the Department of Defense. Senate Report 116-48, page 115, accompanying S. 1790, the National Defense Authorization Act for Fiscal Year 2020.

[2] Babich, R., Clark, M. A., and Joó, B. Parallelizing the QUDA Library for Multi-GPU Calculations in Lattice Quantum Chromodynamics. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis* (Washington, DC, USA, 2010), SC '10, IEEE Computer Society, pp. 1–11.

[3] Chang, Y.-T. S., Hood, R. T., Jin, H., Heistand, S., Cheung, S., Djomehri, M. J., Jost, G., and Kokron, D. Evaluating the Suitability of Commercial Clouds for NASA's High Performance Computing Applications: A Trade Study. Tech. Rep. NAS-2018-01, NASA, May 2018.

[4] Clark, M., Babich, R., Barros, K., Brower, R., and Rebbi, C. Solving Lattice QCD systems of equations using mixed precision solvers on GPUs. *Comput.Phys.Commun. 181* (2010), 1517–1528.

[5] Clark, M. A., Joó, B., Strelchenko, A., Cheng, M., Gambhir, A., and Brower, R. C. Accelerating lattice qcd multigrid on gpus using fine-grained parallelization. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (2016), SC '16, IEEE Press.

[6] Coghlan, S., and Yelick, K. The magellan final report on cloud computing.

[7] Coleman, T., Cansanova, H., Maheshwari, K., Pottier, L., Wilkinson, S. R., Wozniak, J., Suter, F., Shankar, M., and Ferreira da Silva, R. WfBench: Automated Generation of Scientific Workflow Benchmarks. In *2022 IEEE/ACM International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)* (2022), pp. 100–111.

[8] Coleman, T., Casanova, H., and Ferreira da Silva, R. Wfchef: Automated generation of accurate scientific workflow generators. In *17th IEEE eScience Conference* (2021), pp. 159–168.

[9] De Sensi, D., De Matteis, T., Taranov, K., Di Girolamo, S., Rahn, T., and Hoefler, T. Noise in the clouds: Influence of network performance variability on application scalability. *Proc. ACM Meas. Anal. Comput. Syst. 6*, 3 (dec 2022).

[10] Deelman, E., Singh, G., Livny, M., Berriman, B., and Good, J. the cost of doing science on the cloud: The montage example. In *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*.

[11] DOE. CORAL-2 Request for Proposal (RFP). https://procurement.ornl.gov/rfp/CORAL2/01_CORAL-2_RFP%20LetterRev8.pdf, 2018. [Online; accessed 19-March-2023].

[12] Edwards, Robert G. and Joó, Bálint. The Chroma software system for lattice QCD. *Nucl.Phys.Proc.Suppl. 140* (2005), 832.

[13] Eisenbach, M., Larkin, J., Lutjens, J., Rennich, S., and Rogers, J. H. GPU acceleration of the locally selfconsistent multiple scattering code for first principles calculation of the ground state and statistical physics of materials. *Computer Physics Communications 211* (2017), 2–7.

[14] Gupta, A., Kalé, L. V., Milojicic, D. S., Faraboschi, P., Kaufmann, R., March, V., Gioachin, F., Suen, C. H., and Lee, B.-S. The who, what, why and how of high performance computing applications in the cloud. In *Proceedings of the 5th IEEE International Conference on Cloud Computing Technology and Science* (2013), CloudCom '13.

[15] Jackson, K. R., Ramakrishnan, L., Muriki, K., Canon, S., Cholia, S., Shalf, J., Wasserman, H. J., and Wright, N. J. Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud. In *2010 IEEE Second International Conference on Cloud Computing Technology and Science* (2010), pp. 159–168.

[16] Karabin, M., Mondal, W. R., Östlin, A., Ho, W.-G. D., Dobrosavljevic, V., Tam, K.-M., Terletska, H., Chioncel, L., Wang, Y., and Eisenbach, M. Ab initio approaches to high-entropy alloys: a comparison of cpa, sqs, and supercell methods. *Journal of Materials Science 57*, 23 (2022), 10677–10690.

[17] Laanait, N., Romero, J., Yin, J., Young, M. T., Treichler, S., Starchenko, V., Borisevich, A. Y., Sergeev, A., and Matheson, M. A. Exascale deep learning for scientific inverse problems. *ArXiv abs/1909.11150* (2019).

[18] Matsuoka, S., Domke, J., Wahib, M., Drozd, A., and Hoefler, T. Myths and legends in high-performance computing, 2023.

[19] AWS Nitro System. https://aws.amazon.com/ec2/nitro/.

[20] Norman, M., Lyngaas, I., Bagusetty, A., and Berrill, M. Portable c++ code that can look and feel like fortran code with yet another kernel launcher (yakl). *International Journal of Parallel Programming* (2022), 1–22.

[21] Norman, M. R. A high-order weno-limited finite-volume algorithm for atmospheric flow using the ader-differential transform time discretization. *Quarterly Journal of the Royal Meteorological Society 147*, 736 (2021), 1661–1690.

[22] Ramakrishnan, L., Zbiegel, P. T., Campbell, S., Bradshaw, R., Canon, R. S., Coghlan, S., Sakrejda, I., Desai, N., Declerck, T., and Liu, A. Magellan: Experiences from a Science Cloud. In *Proceedings of the 2nd International Workshop on Scientific Cloud Computing* (New York, NY, USA, 2011), ScienceCloud '11, Association for Computing Machinery, p. 49–58.

[23] Roloff, E., Carreño, E., Valverde-Sánchez, J., Diener, M., Da Silva Serpa, M., Houzeaux, G., Schnorr, L., Maillard, N., Gaspary, L., and Navaux, P. Performance evaluation of multiple cloud data centers allocations for hpc. In *High Performance Computing - 3rd Latin American Conference,*

CARLA 2016, Revised Selected Papers (Germany, 2017), C. Barrios Hernandez, I. Gitler, and J. Klapp, Eds., Communications in Computer and Information Science, Springer, pp. 18–32.

[24] Romero, J., Yin, J., Laanait, N., Xie, B., Young, M., Treichler, S., Starchenko, V., Borisevich, A. Y., Sergeev, A., and Matheson, M. A. Accelerating collective communication in data parallel training across deep learning frameworks. In *Symposium on Networked Systems Design and Implementation* (2022).

[25] Vazhkudai, S. S., de Supinski, B. R., Bland, A. S., Geist, A., Sexton, J., Kahle, J., Zimmer, C. J., Atchley, S., Oral, S., Maxwell, D. E., Larrea, V. G. V., Bertsch, A., Goldstone, R., Joubert, W., Chambreau, C., Appelhans, D., Blackmore, R., Casses, B., Chochia, G., Davison, G., Ezell, M. A., Gooding, T., Gonsiorowski, E., Grinberg, L., Hanson, B., Hartner, B., Karlin, I., Leininger, M. L., Leverman, D., Marroquin, C., Moody, A., Ohmacht, M., Pankajakshan, R., Pizzano, F., Rogers, J. H., Rosenburg, B., Schmidt, D., Shankar, M., Wang, F., Watson, P., Walkup, B., Weems, L. D., and Yin, J. The Design, Deployment, and Evaluation of the CORAL Pre-Exascale Systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis* (2019).

[26] Wang, Y., Stocks, G. M., Shelton, W. A., Nicholson, D. M. C., Temmerman, W. M., and Szotek, Z. Order-N multiple scattering approach to electronic structure calculations. *Phys. Rev. Lett. 75* (1995), 2867.

[27] Winter, F. T., Clark, M. A., Edwards, R. G., and Joó, B. A framework for lattice qcd calculations on gpus. In *Proceedings of the 2014 IEEE 28th International Parallel and Distributed Processing Symposium* (Washington, DC, USA, 2014), IPDPS '14, IEEE Computer Society, pp. 1073–1082.