Benchmarking AWS and HPC Services

Next-generation Aerospace modelling and simulation

October 2020

Notices

Contents

Abstract

The aerospace industry has been using Computational Fluid Dynamics (CFD) for decades to create and optimize designs digitally, from the largest passenger planes and fighter jets to gliders and drones. CFD enables them to get to market faster and at a lower development cost, and reducing, for example, expensive wind tunnel testing[1].

By 2050, commercial aircraft will face even more stringent environmental legislation, which makes continuously improving aircraft performance[2] necessary. Additionally, engineers are increasingly using more complex and higher fidelity CFD simulations as well as digital twinning to reduce design times.

Running these simulations require a large computing power. As the simulation complexity increases, the demand of High-Performance Computing (HPC) systems also increases. This trend is set to continue, and aerospace companies are already seeking the next generation of CFD tools and the high-performance computing (HPC) systems to run them.

This whitepaper will guide you through the complexity of CFD in the aerospace sector and how new HPC architectures can help the aerospace industry meet future challenges.

Introduction

In this whitepaper, we look at the HPC offerings available from Amazon Web Services (AWS) and use a full aircraft CFD simulation to compare the performance against a traditional on-premises supercomputer. AWS offers the advantage of a flexible, on-demand infrastructure. Additionally, we will demonstrate how the AWS cloud can match the performance of the on-premises supercomputer and also compete on price. The example performance in this paper will be benchmarked using a 149 million cell CFD mesh running on up to 3,600 CPU cores. We will demonstrate that running CFD on GPUs gives the industry the ability to vastly speed up CFD simulations, a trend that we are likely to see continued as the next generation of GPUs get deployed.

We will start by comparing an on-premises x86 cluster with a range of AWS high-performance infrastructures that include Amazon Intel, NVIDIA, Amazon Elastic Compute Cloud (Amazon EC2) Graviton 2 arm64 based instances[3], and Amazon's high performance FSx for Lustre filesystem[4].

Is the Cloud Ready for HPC?

High-performance computing (HPC) is used in a wide variety of industries for a large number of applications from the simulation of real-world physics to machine learning and data processing. Applications that require a large amount of computing will require HPC capabilities in some manner.

A CFD simulation is central processing unit (CPU) intensive but also requires a large amount of memory bandwidth and fast I/O. To keep the runtime down and distribute the memory requirements, these typically run in parallel over several computers in an HPC cluster. The requirement for HPC has driven companies to purchase dedicated HPC systems that are deployed on-premises, or make use of dedicated HPC centers.

In the last few years, the advances in cloud computing infrastructure, has meant that more and more workloads that previously required on-premises HPC can now run on the cloud[5]. However, there is still the perception that large scale workloads require an on-premises HPC resource. In this paper we explore this and test various AWS HPC performance options for a large scale industrial CFD test case.

The CFD Challenge

To test the AWS infrastructure, we needed a CFD code, and we used the zCFD solver from Zenotech. Zenotech is a computational engineering company based in Bristol, UK, who has extensive experience working in the aerospace industry. As well as expertise in CFD, they offer on-demand cloud HPC via their EPIC product.

zCFD combines the latest CFD numerical methods running on the latest computing hardware. It's designed to perform a fast and scalable simulation. zCFD is fully parallel and can simulate turbulent flow (RANS, URANS, DDES or LES) including automatic scalable wall functions. It is a fully compressible solver with preconditioning for low Mach numbers, and can be run in either standard finite volume mode or with new high order (DG flux reconstruction) capability for efficient scale-resolution. It has solution acceleration via geometric multigrid, dual-time-stepping and polynomial multigrid. Additionally, it will automatically detect and run on NVIDIA GPU accelerators where present and also supports arm64 architectures.

We used the XRF1 Simulation Challenge supplied by Airbus as a test case[6]. XRF1 is a generic non-production aircraft design but represents the kind of simulations that Airbus performs to produce a real aircraft. Airbus has extensive wind tunnel result data for

XRF1 and has released the model to partners to support the verification and validation of CFD codes.

Combining zCFD and XRF1 enabled us to run the benchmarks in a way that replicates runs that would typically be used in an aerospace production design cycle.

To quantify the performance of AWS infrastructure we will be comparing it with a dedicated HPC system. The system is an on-premises Cray CS400 cluster, configured specifically for HPC tasks. It is one of several types of HPC cluster that large companies have on-premise.

This benchmarking exercise was carried out as part of the Aerospace Cloud Services project, part funded by Innovate UK and the Aerospace Technology Institute.

AWS HPC Configuration

To create an HPC cluster in AWS the following services were used:

- Amazon Elastic Compute Cloud (Amazon EC2)
- Elastic Fabric Adapter (EFA)
- Elastic Network Adapter (ENA)

Amazon FSx for Lustre (FSx for Lustre) AWS infrastructure was set up by Zenotech and managed by their EPIC platform. EPIC provides access to a variety of HPC systems, both cloud-based and a range of specialist supercomputing providers.

Amazon EC2

For these tests several instance types were used. The initial scaling tests used the compute optimized Amazon EC2 C5 instances. For the GPU runs, Amazon EC2 P3, and Amazon EC2 G4 instance types were used. Finally, Amazon EC2 C6g Graviton2 arm64-based instance types were used.

Table 1 – Instance Comparison

Instance Type	Chipset	# of cores/vCPU	GPUs	Network
C5.18xlarge	Intel	72 vCPU (36 physical cores)	-	25 Gbit/s ethernet
C5n.18xlarge	Intel	72 vCPU (36 physical cores)	-	100 Gbit/s ethernet with EFA
P3.16xlarge	Nvidia	64 vCPU (36 physical cores)	8 x Nvidia V100	25 Gbit/s ethernet
P3dn.24xlarge	Nvidia	96 vCPU (48 physical cores)	8 x Nvidia V100	100 Gbit/s ethernet with EFA
G4dn.12xlarge	Nvidia	48 vCPU (24 physical cores)	4 x Nvidia T4	50 Gbit/s ethernet
C6g.16xlarge	AWS arm64	64 physical cores	-	25 Gbit/s ethernet

For our tests, hyperthreading was disabled, restricting the instance to just use the physical CPU cores available. All AWS instances were run within *cluster placement groups*[7] to minimize network latency between nodes. The latest release of Amazon Linux 2 was used as the base OS for the compute node Amazon Machine Image (AMI).

Elastic Network Adapter (ENA)

ENA[8] is a custom network interface optimized to deliver high throughput and packet per second (PPS) performance, and consistently low latencies on EC2 instances. Using ENA, up to 100 Gbps of network bandwidth on certain EC2 instance types can be used.

Elastic Fabric Adapter (EFA)

The Amazon EC2 C5N & P3DN instance type provides EFA[9] network adapters[10]. EFA provides lower and more consistent latency and higher throughput than the TCP/IP transport traditionally used in cloud-based HPC systems. It is optimized to work on the existing AWS network infrastructure, and it can scale depending on application requirements.

The latest releases of Intel MPI and Open MPI are integrated with EFA via libfabric. To make use of the EFA Intel MPI 2019 Update 6 was used, which bundles libfabric 1.9.0 with a dedicated provider for EFA. Newer releases of OpenMPI also support EFA via libfabric.

Amazon FSx for Lustre (FSX for Lustre)

AWS has a number of storage solutions that could be used for an HPC cluster. Common patterns include the export of an EBS volume via NFS, exporting local NVMe SSD storage via NFS, creating your own parallel file system using EC2 instances (for example using BeeGFS, GlusterFS, etc). AWS has also launched the AWS FSx for Lustre service, which provides on demand managed Lustre file systems. Lustre is a parallel file system that provides high speed concurrent access to the dataset and avoids data contention.

The benchmarks were run using the AWS FSx for Lustre configured in the following way:

- Scratch 2 configuration
- 7.2 TiB volume size
- 1.406 GB/s Throughput capacity

The throughput capacity of the FSx volume is dependent on the size of the volume, and so this volume was sized to achieve the same throughput seen during tests on the on-premises cluster[11]. With the AWS cluster configured as described above the following hardware comparison for the first scaling tests is listed in the table below.

Table 2 – Hardware Comparison

	On-premises Cray CS400	**AWS**
CPU	2 x 12 core E5-2600v v4 (Broadwell)	2 x 18 core 1st Gen Intel Xeon SP (Skylake)
Memory	128 GB	144 GB (C5) 192 GB (C5n)
Network	Mellanox FDR Infiniband 56Gbit/s	25 Gbit/s ethernet (C5.18xlarge) 100 Gbit/s ethernet (C5n) EFA (C5n.18xlarge)
Storage	IBM Spectrum Scale	FSx for Lustre (200 MB/sec/TB)
OS	Centos 7	Amazon Linux 2
Scheduler	Slurm	Slurm

Scaling Performance

The results below show how the runtime of the CFD solver scales with the number of CPU cores used. This is a strong scaling test; the same CFD mesh was used but with a varying number of CPU cores. The *ideal* case is where the performance scales linearly to the number of CPUs used, but at higher core counts this becomes difficult as the communication between partitions starts to dominate the calculation time. The speed and bandwidth of the network interconnect also has a considerable impact on strong scaling.

For the scaling tests, Zenotech generated a 149 million cell mesh on the XRF1 geometry. This mesh is much larger than one that would be used in day-to-day industrial CFD, which would be closer to 20 million cells, but is large enough to test the infrastructure at scale.

The tests run with zCFD in three modes:

MPI - The code was run in pure MPI mode, one MPI process per CPU core and one thread per MPI process. This mode is the most network intensive as all processes can potentially communicate.

Hybrid - The code was run in hybrid MPI/OpenMP mode. In this case one MPI process is run per CPU socket and then one OpenMP thread per CPU core on that socket. So, for example a compute node with two 12 core CPUs would run two MPI processes each with 12 OpenMP threads. This mode is more efficient on the network as it reduces the number of components communicating.

GPU - Zenotech's zCFD is capable of offloading the solve to GPUs via CUDA[12]. To do this one MPI process is run per GPU on the system and the code detects the available GPUs and binds one MPI process to each GPU.

The solver was also compiled for arm64 to benchmark the Graviton2[13] arm64-based CPUs.

How did the AWS Cluster Perform?

The chart below illustrates the strong scaling results[14] for the Amazon EC2 c5n.18xlarge instance type. It shows a comparison of the scaling with and without the use of EFA for communications. This is running zCFD in Hybrid mode with two MPI processes on each node and 18 OpenMP threads per MPI process.

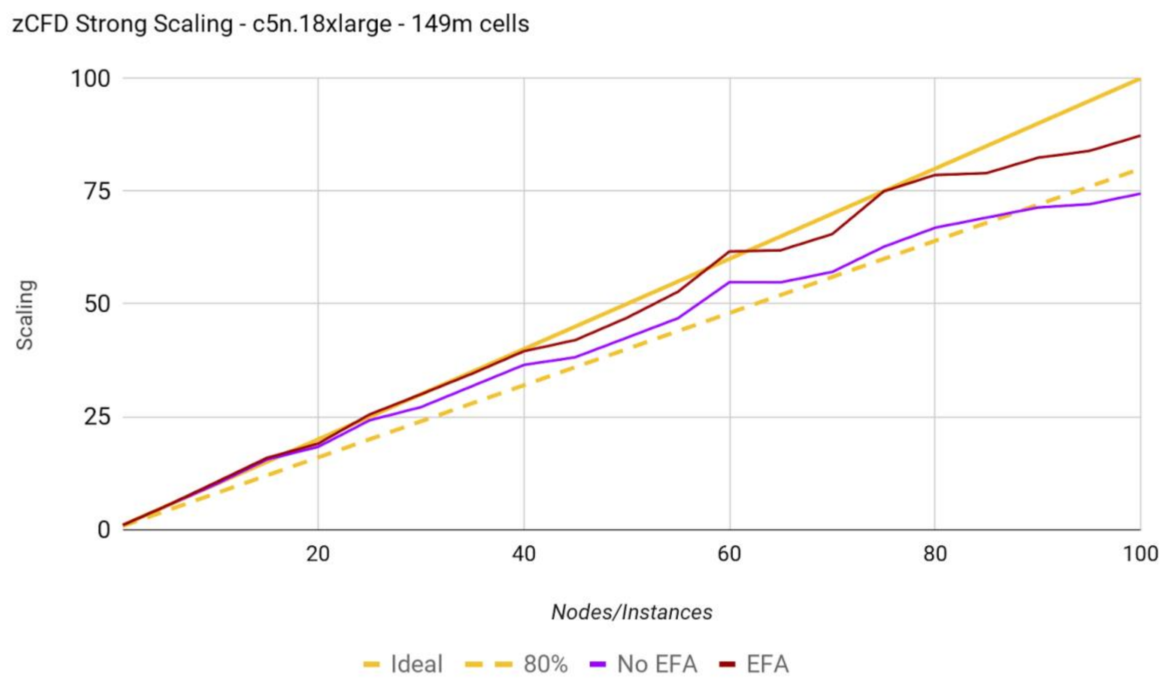

Figure 1 – Scaling Results for the Amazon EC2 c5n.18xlarge instance type

Even without EFA enabled, the scaling is still very strong, with around 75% efficiency at 100 nodes/3600 cores. This is over the standard ENA based ethernet interfaces on those nodes using the libfabric TCP provider. The introduction of EFA improves things further, with near linear scaling up to around 75 nodes and an efficiency of 87% at 100 nodes/3600 cores.

Some applications, such as legacy CFD codes, are not able to run in a hybrid mode, so we also ran the case in full MPI mode; with 36 MPI processes per node and no OpenMP threads. The results for this are shown below.

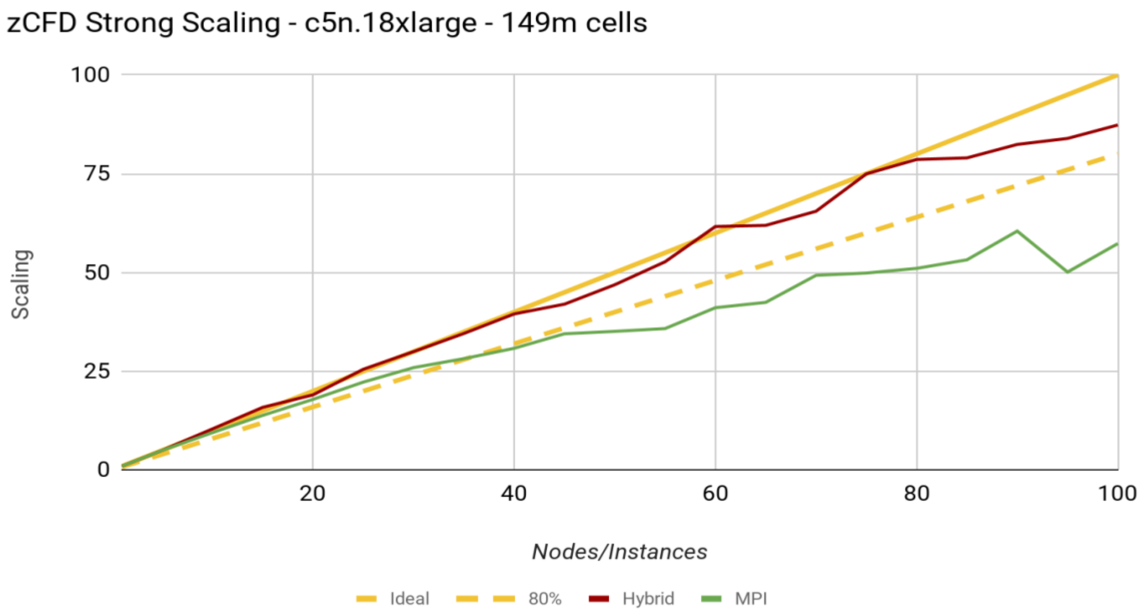

Figure 2 – Hybrid mode results

The scaling for the MPI run drops off faster than the hybrid run. This was expected due to the increased number of communicating processes in the MPI run putting more load on the network. These results mirror those seen on the on-premises cluster.

Comparison to the On-Premises Cluster

The results from the AWS cluster look very promising. Let's review how they compare to the Infiniband connected on-premises cluster. The chart below shows a comparison of a zCFD hybrid run on c5n.18xlarge with EFA enabled and the on-premises cluster.

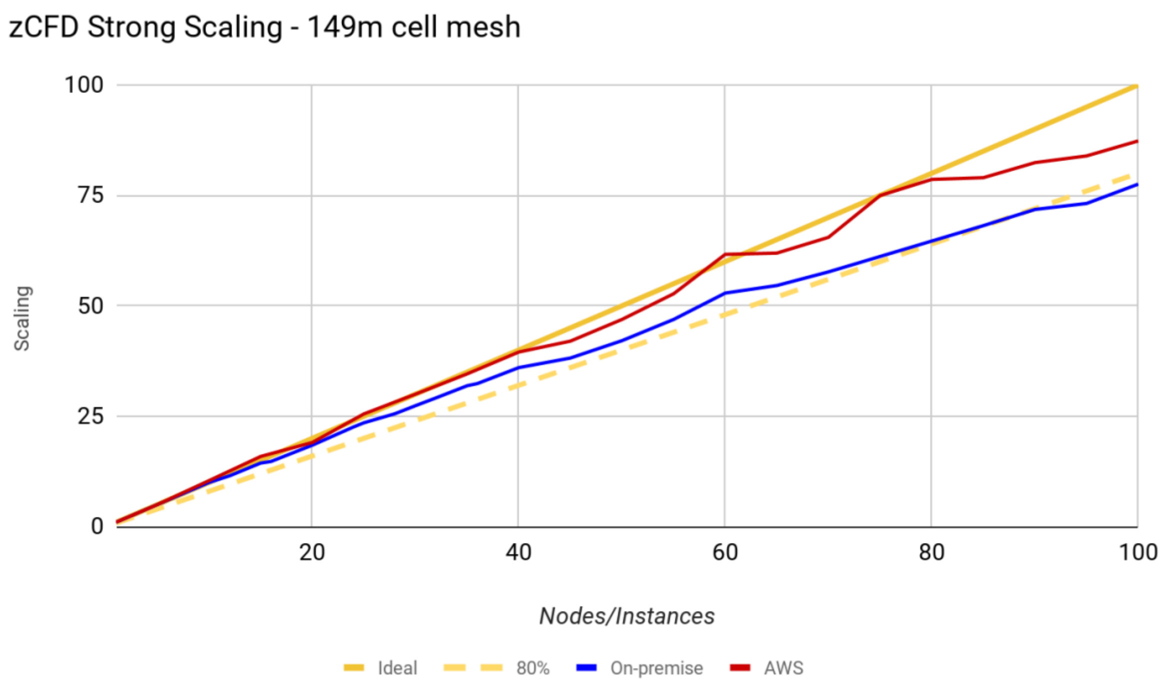

Figure 3 - Comparison of a zCFD hybrid run on c5n.18xlarge with EFA enabled

As you can see the scaling on the AWS cluster is closer to Ideal than the on-premises cluster. Even at 100 nodes we see 78% parallel efficiency, when compared to a single node, for the on-premises cluster versus 87% on the AWS cluster.

So parallel scaling on the AWS cluster looks good, but it's also useful to make a direct performance comparison. If we look at cycle time per compute node/instance the AWS cluster is around 1.5x faster than the on-premises cluster. If we run the job until the solver converges (in this case around 15,000 cycles) we can calculate a *time to solution*. The graph below shows the time to solution at different node counts, where a lower time to solution is better.

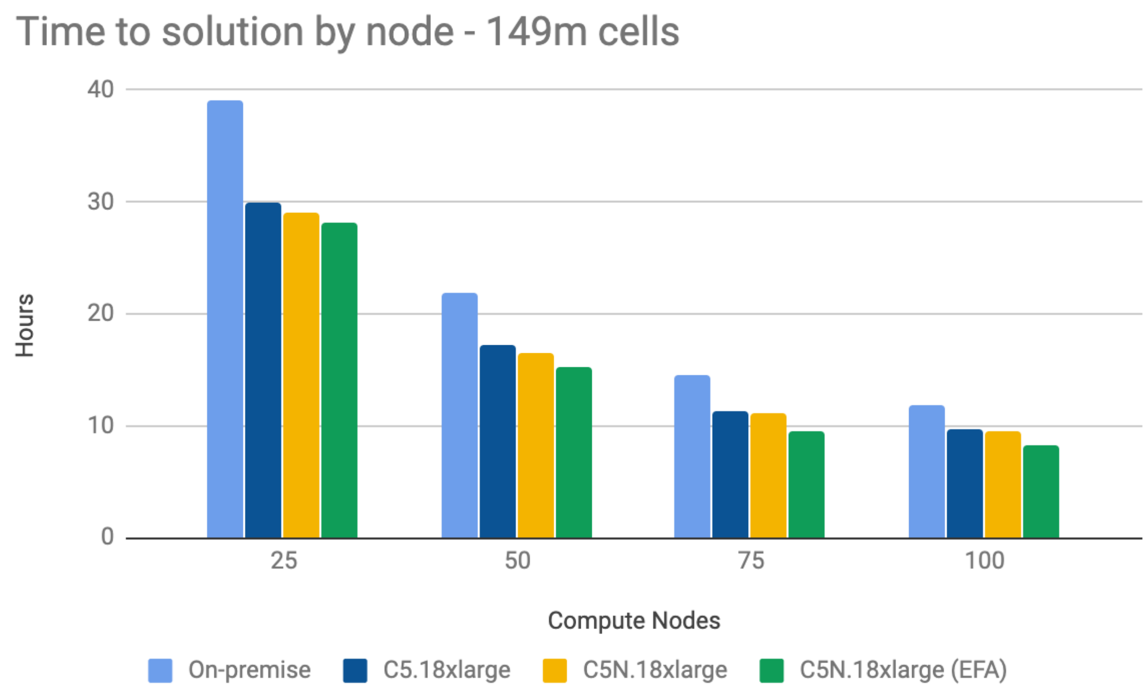

Figure 4 – Time to solution node counts

From the chart above, we can see the AWS instances are giving a quicker time to solution for the same node counts as the on-premises cluster. This is to be expected as the AWS compute nodes have 36 Skylake cores versus the 24 Broadwell cores on the on-premises cluster compute nodes. However, the scaling performance demonstrates that this advantage is still present at the higher node counts where one might expect the impact of the interconnect network to give the on-premises cluster an advantage.

What was the Impact of FSx for Lustre?

The scaling figures above are the time taken for the solver loop of the CFD run, removing the impact of time taken doing file system I/O, for example, reading the mesh, logging and writing the results. To assess the impact of Amazon FSx for Lustre we also ran the tests using an NVMe exported via NFS based solution and compared the overall elapsed time for the run (to 20 cycles). The chart below shows the comparison between the on-premises cluster with IBM Spectrum Scale storage, Amazon FSx for Lustre and the NFS solution.

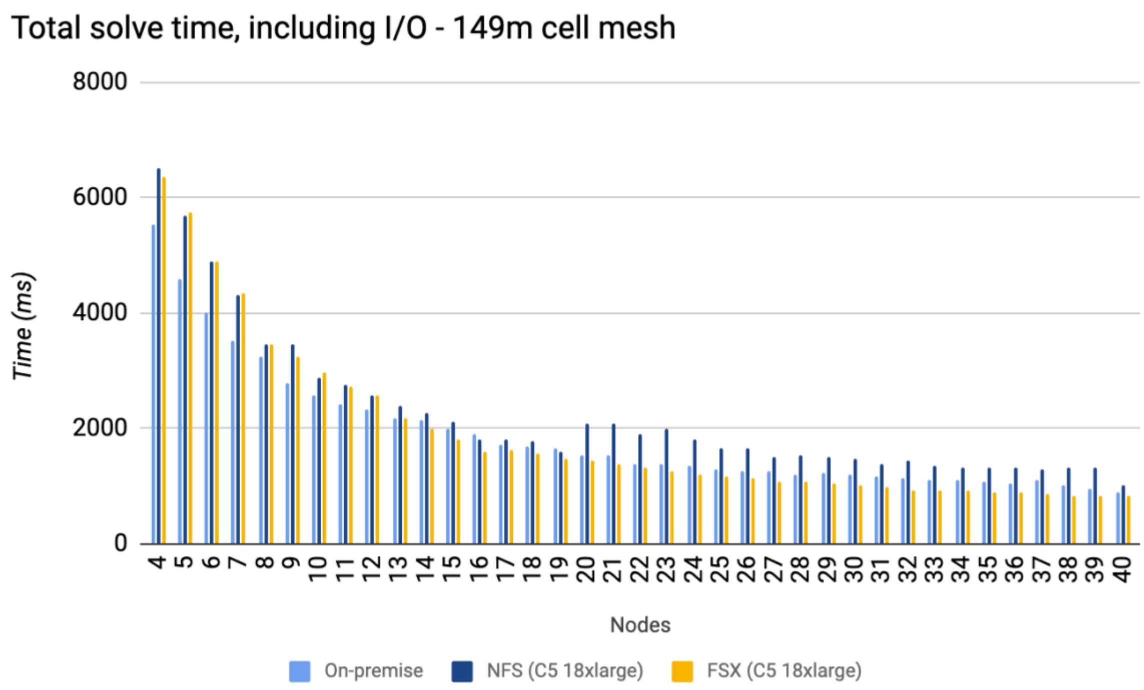

Figure 5 - Comparison between the on-premises cluster with IBM Spectrum Scale storage

The above chart shows that the Amazon FSx for Lustre based solution scales well with the increased node count and at the higher node counts actually outperformed the Spectrum Scale solution.

GPUs

The Amazon EC2 Accelerated Computing instances offer access to hardware accelerators and coprocessors that have the potential to offload work from the CPU and speed up your workload. Among these instances several NVIDIA GPU types are available. If an application can make use of the GPUs it can potentially access a large amount of computing power per node. For example, a single V100 can perform nearly 8 TFLOPs of double precision performance and 15 TFLOPs of single precision performance.

zCFD is capable of making use of CUDA enabled GPUs, it looks for these devices on startup and will use them if they are available. We can run the same benchmarks on GPU nodes and see how they perform in relation to both the Cray resources and the Amazon EC2 C5 instances.

The AWS setup is as described above but the AMI was updated to include the latest NVidia drivers and CUDA 10.2. The benchmarks were run on the following instance types.

Table 3 – Benchmark instance types

Instance	GPU Type	GPU Count	GPU Memory (GB per GPU)
P3.16xlarge	NVIDIA V100	8	16
P3dn.24xlarge	NVIDIA V100	8	32
G4dn.12xlarge	NVIDIA T4	4	16

Single Node Performance

The first test was a single node performance comparison using the 17 million cell mesh. For these tests all available cores or GPUs on the node were used. The results below show the cycle time of the CFD solver, lower is better.

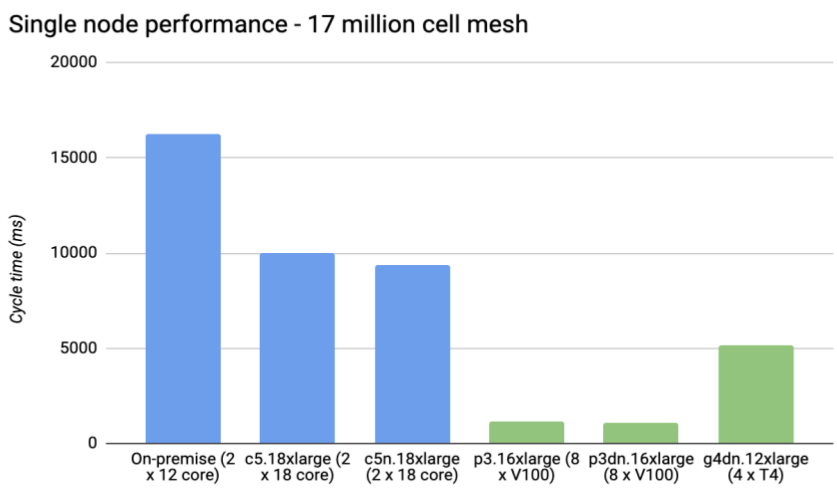

Figure 6 - Cycle time of the CFD solver

The single node performance on the P3 instances shows the speed-up as a possible consequence of using the GPUs. If we extrapolate the time to run the simulation to 15,000 cycles, the on-premises node would take 67.6 hours, while a single P3 node would take 4.6 hours.

Strong Scaling on GPU

The chart below shows the strong scaling using the 149 million cell mesh on the GPU instances. The speed-up show is relative to 8 GPUs.

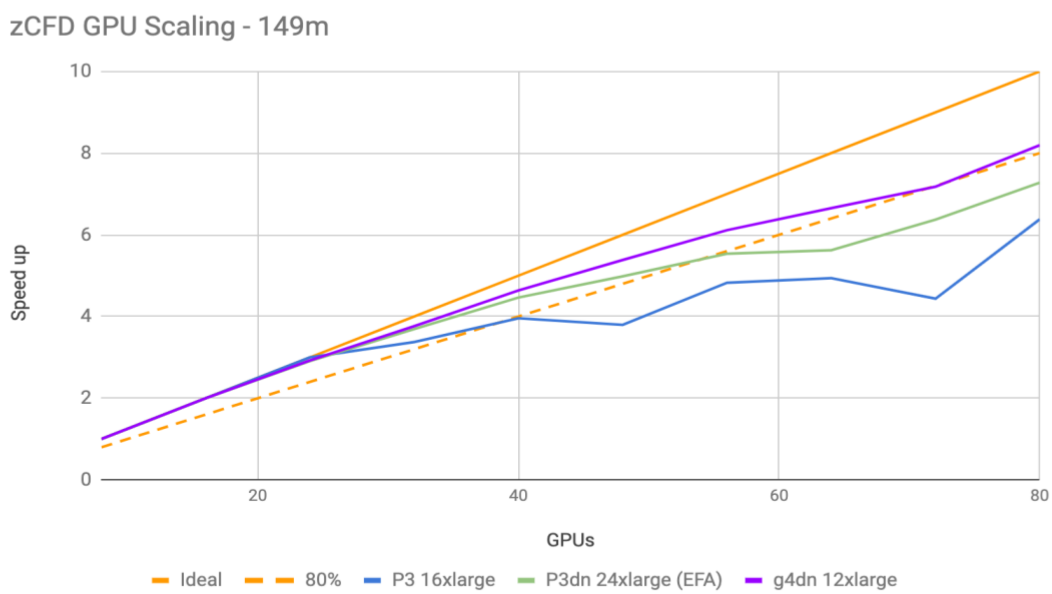

Figure 7 - Scaling using the 149 million cell mesh on the GPU instances

The improvement in scaling between the P3 and the P3DN is down to the low latency communications over EFA. The G4 instance type is showing even better scaling, in this case there are a lower number of less powerful GPUs per node and so the volume and rate of communications per node is significantly less, reducing the load on the instance network adapters.

In addition, NVIDIA has recently released the A100 GPUs and we would expect to see a performance jump beyond that shown on the V100s in these benchmarks. AWS has recently announced[15] they will provide these A100 GPUs, in the future blogs will look at the performance on these instances.

Using Graviton2

AWS Graviton2[16] processors are custom designed by AWS using 64-bit arm64 Neoverse cores to deliver the best price performance for your cloud workloads running in Amazon EC2.

The performance tests were repeated on the c6g.16xlarge instance type, which has 64 Graviton2 cores. To run on the EC2 C6g instances, zCFD was compiled for arm64 and switched to using OpenMPI 4.0.8 rather than Intel MPI. The case was run in hybrid mode with 16 MPI processes, each with 4 OpenMP threads.

Single Node Performance

The results for single node performance comparison using the 17 million cell mesh are shown below. All 64 cores on the instance were used for this test.

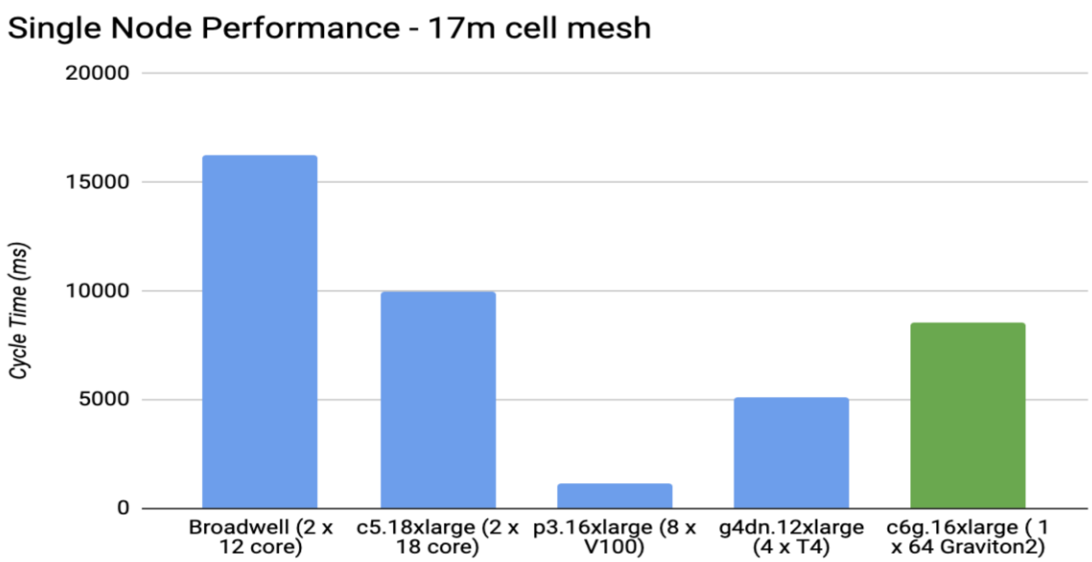

Figure 8 – Single node performance

The chart shows that a single EC2 C6g node is providing better performance than a c5.18xlarge node. This is interesting when we consider the price performance discussed below.

Strong Scaling on Graviton2

The chart below shows the strong scaling results for the 149 million cell mesh, it was run on up to 55 instances (3520 Graviton2 cores).

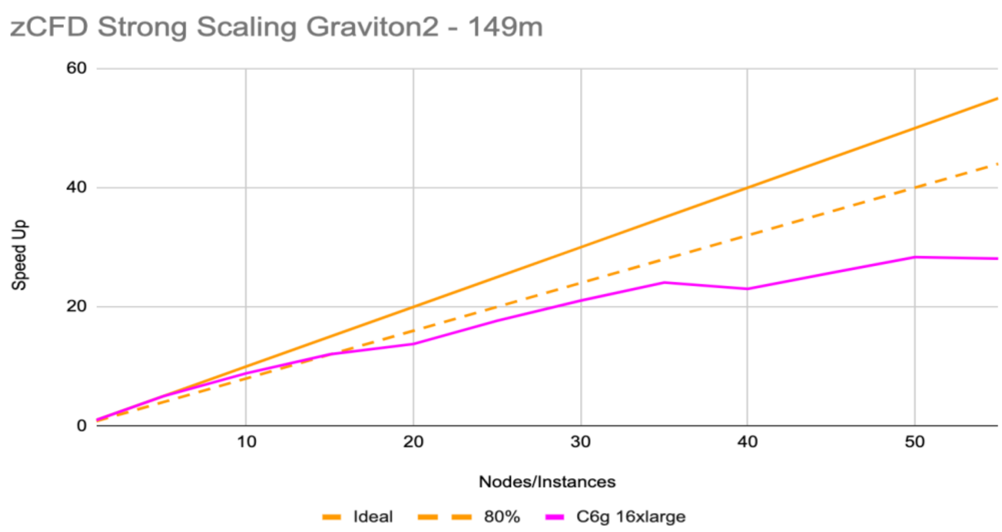

Figure 9 – Scaling results for 149 million cell mesh

From the results shown, the scaling isn't quite as good as we saw on the EC2 C5 instance types, this is partly due to the change in communications pattern, 4 MPI processes per node vs. 2 MPI processes. Improvements to the scaling would be expected with some tuning to the MPI/OpenMP behavior within zCFD.

Price Performance

Performance is an important factor for CFD, but price is also significant. The number of CFD runs that need to be performed as part of the design cycle is large and so price/performance for CFD on HPC becomes critical.

AWS offers several pricing models[17]. For the purposes of this study, we considered standard on-demand pricing, reserved instance (RI) pricing and the spot price of the instances when the benchmarks were carried out. For the pricing of the on-premises cluster we have used a price of approximately **$0.05 per core/hour**, which from our experience is a fair representation of the cost of using an on-demand specialist HPC system.

The relative price performance shown below is the cycle time of the run divided by the resource cost, normalized relative to the on-premises system. A value above 1 represents better price/performance than the on-premises cluster.

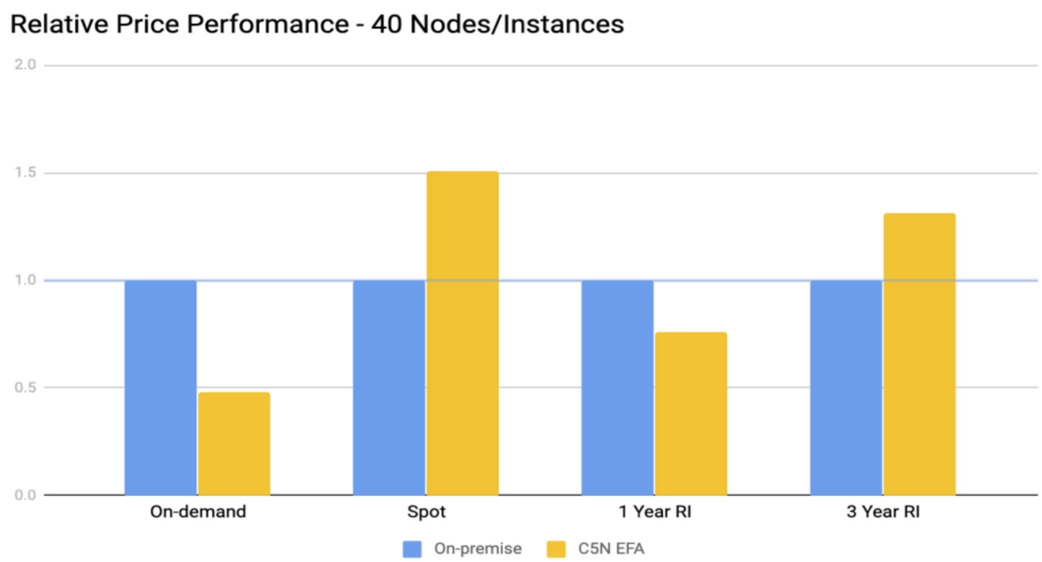

Figure 10 – Price performance

The on-demand pricing places the AWS solution at a lower price/performance than the on-premises cluster. However, when we use reserved instance or spot pricing the AWS solution is more cost effective than the on-premises cluster.

The drawback of the spot pricing model is that your instances may be reclaimed during the CFD run. Application level checkpointing, to the FSx file system, can help in this case - allowing you to restart after an interruption but for a large, long running CFD run spot may not be appropriate.

Another approach is to look at the Reserved Instance pricing. This is ideal for the situation where you can forecast your HPC demand for 1 or 3 years and access reduced pricing by reserving a number of instances on those timescales. 3-year reserved instance pricing brings discounts close to the spot pricing but the safety of knowing your instances will not be reclaimed. The 3-year pricing is also a familiar model to companies that tend to refresh in-house hardware on similar timescales. Mixing on-premise, reserved instance and spot pricing will often provide a company with the best mix of flexibility and cost savings.

Below is the same price/performance metric for the GPU and Graviton2 instances. You can see that the increased performance of the GPU options balances out the higher instances prices and brings the price performance in line with on-premises cluster even at on-demand pricing, in fact the G4 instances are providing significantly better price performance.

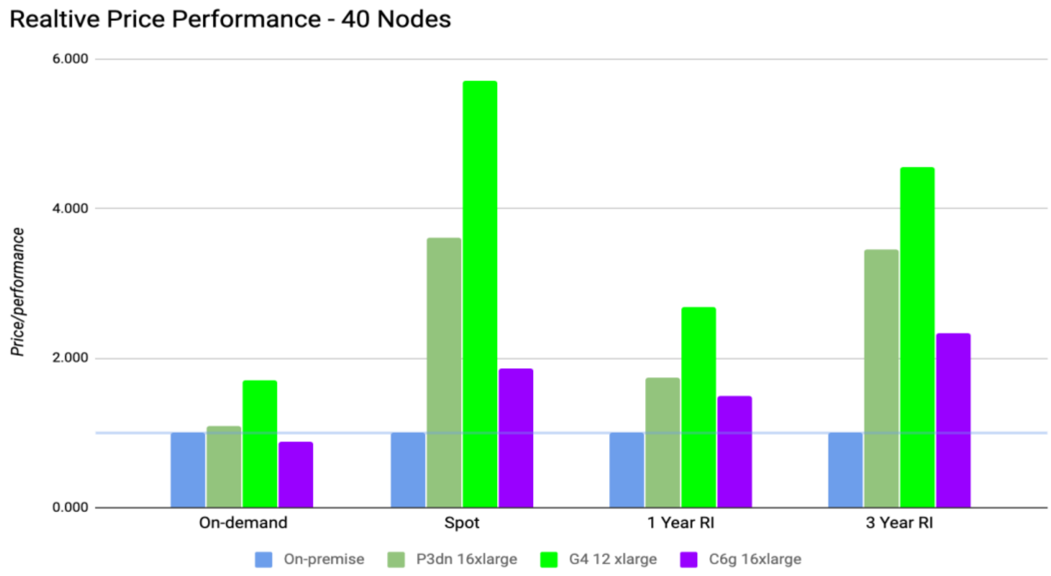

Figure 11 – Price / Performance metric

The Graviton2 price performance is also close to the on-premises and higher than the EC2 C5n price/performance. This makes it a good option for those codes that can't easily take advantage of GPUs but may be able to target the arm64 architecture.

Considerations

One of the biggest changes that comes with using the Cloud for HPC is the ability to be flexible with the size of the infrastructure. It enables you to do things that are not easy with an in-house cluster such as *design of experiments* and increase model accuracy or fidelity.

If turnaround time is a priority you can match your HPC cluster size to your demand, meaning users no longer have to queue. HPC clusters can be spun up in a matter of minutes and then used for a single job, a specific project or even dedicated to a single user. For example, Zenotech was able to spin up all the infrastructure required to run the 100 node benchmarks in less than 10 minutes, and then when the runs completed they post-processed the data, stored the results on Amazon Simple Storage Service

(Amazon S3), then terminated all of the infrastructure, with no on-going costs associated with the HPC cluster.

Conclusion

We ran the benchmarks on 3,600 CPU cores using a 149 million cell aircraft mesh and demonstrated that the AWS based infrastructure is capable of running a large aircraft CFD simulation at scale at a level of performance that exceeds an on-premises HPC cluster. Combined with the on-demand Lustre file system offered by FSx for Lustre gives organizations that require HPC a flexible cloud-based option to complement, or replace on-premises HPC.

We also demonstrated that running CFD on GPUs gives the industry the ability to vastly speed up CFD simulations, a trend that will continue as next generation of GPUs gets deployed. The advent of arm64 processors is a very interesting development and we have demonstrated that they can be used for an HPC workload, and offer a price/performance advantage over the x86 equivalent. Cloud HPC offers the flexibility to use this heterogeneous mix of hardware without expensive reconfiguration or capital purchases.

Contributors

Contributors to this document include:

- Mike Turner, CTO, Zenotech
- Neil Ashton, Principal Solution Architect CFD, AWS
- Gilles Tourpe, Senior Go-to-Market Specialist HPC

Further Reading

- https://zenotech.com/epic-elastic-private-interactive-cloud/
- https://aws.amazon.com/hpc/cfd/
- https://www.hpcworkshops.com/

Document Revisions

Date	Description
October 2020	First publication

Notes

1 http://journals.cambridge.org/abstract_S000192401500010X

2 https://www.cambridge.org/core/journals/aeronautical-journal/article/role-of-cfd-in-aerodynamics-offdesign/266F5861A59DB86B48A36313FA5A68D5

3 https://aws.amazon.com/ec2/instance-types/p3/ https://aws.amazon.com/ec2/instance-types/g4/

4 https://aws.amazon.com/ec2/instance-types/c6/

5 https://aws.amazon.com/fr/hpc/cfd/

6 https://link.springer.com/article/10.1007/s13272-015-0179-7

7 https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html

8 Enabling enhanced networking with the Elastic Network Adapter (ENA) on Linux instances - Amazon Elastic Compute Cloud

9 https://aws.amazon.com/fr/hpc/efa/

10 https://aws.amazon.com/blogs/aws/now-available-elastic-fabric-adapter-efa-for-tightly-coupled-hpc-workloads/

11 Amazon FSx for Lustre FAQs Page

[12] https://blogs.nvidia.com/blog/2012/09/10/what-is-cuda-2/

[13] https://aws.amazon.com/ec2/graviton/

[14] https://www.kth.se/blogs/pdc/2018/11/scalability-strong-and-weak-scaling/#:~:text=Strong%20scaling%20concerns%20the%20speedup,is%20governed%20by%20Gustafson's%20law.

[15] https://aws.amazon.com/blogs/machine-learning/aws-to-offer-nvidia-a100-tensor-core-gpu-based-amazon-ec2-instances/

[16] https://aws.amazon.com/ec2/graviton/

[17] https://aws.amazon.com/ec2/pricing/