

# High-Performance Computing for the Age of AI and Cloud Computing



**nvidia**

# High-Performance Computing for the Age of AI and Cloud Computing

<b>The NVIDIA Framework for HPC</b>	<b>2</b>
<b>HPC - New Challenges, Same Goals</b>	<b>4</b>
High-performance Computing Objectives	4
Maximize Insight	4
Maximize Efficiency of Expensive Resources	5
What are the best performance indicators?	6
Application Throughput vs. Job Performance	6
Inefficiencies of Parallelism	6
Measuring Performance Indicators	9
<b>Trends in High-performance Computing</b>	<b>11</b>
Emphasis on Energy Efficiency	11
Reducing Carbon Emissions	12
Water Cooling and Immersion Cooling	13
ARM64 for HPC	15
GPU Power Tuning	16
Direct Current Powered Data Centers	17
Data Driven Discovery	18
The Rise of Dedicated Silicon Acceleration Technologies	19
Pushing HPC to Data Sources (Edge)	20
What is quantum computing?	22
Rise of the Metaverse and Digital Twins	23
<b>High-Performance Computing Key Enablers</b>	<b>25</b>
HPC Application Support and Breadth	25
HPC and AI Convergence	27
Speeding Simulations	27
Equation Enhanced Machine Learning	31
Transforming HPC	33
Coherent System and Accelerator Memory	34
Advances in Computer Networks	36
Network Bandwidth for Scaling	36
Interconnects for Multi-GPU and Multi-Node Acceleration	38
Cloud-Native Supercomputing	42
SmartNIC Offload	44

Converged Accelerator and Network Interface Card	45
HPC Developer Ecosystem	46
Vibrant Community	46
Standard, Open, and Portable Parallel Programming Models	47
Cloud, VMs, and Containers	52
Conclusion	55

## HPC - New Challenges, Same Goals

Whether it is exploring the origins of life on our planet, creating schema to drive a car safely and automatically, or detecting nefarious behavior in financial transactions - [High-Performance Computing](#) (HPC) is used to generate and authenticate new insights.

Because these insights are often novel, they are usually quite valuable. It stands to reason that the primary goal for HPC systems is to maximize the rate at which they produce valuable intellectual property (IP) while minimizing costs.

Whether you are a data center manager, a budding HPC developer, or a seasoned researcher you will find a wealth of information throughout this eBook regardless of your background. You will not only discover what drives HPC throughput, but also gain a deeper understanding of the mechanisms and environments that envelope the current atmosphere of HPC, learn from industry examples that highlight this ever-evolving discipline, and more.

For those wondering what future HPC initiatives will look like at NVIDIA, check back periodically to discover additional chapters, examples, use cases, and linked blogs interlaced throughout these pages.

### High-performance Computing Objectives

#### *Maximize Insight*

Most of the time in HPC, intellectual property is generated by completing workload tasks, which are sometimes called “jobs.” So, to optimize the value of the HPC system, the simplest quantity to maximize is throughput.

However, in most multi-user, multi-disciplinary HPC centers and laboratories - a workload task is not a singular, uniquely defined entity. There is a spectrum of task diversity that tends to correlate with the type of HPC environment being characterized.

Academic systems - referring to centralized HPC environments similar to [TACC Frontera and Wisteria at the University of Tokyo](#) - tend to serve the broadest set of disciplines. This is generally due to the diversity of research and departmental staff who use HPC as a primary investigative and exploratory tool in their respective fields.

Commercial HPC centers tend to be much more narrowly focused on where their HPC is used to define, certify, and support products. This is true for areas as diverse as integrated circuit

design ([SSC-21 at Samsung](#)), [consumer products](#), [aerospace \(TOKI-SORA at Japan Aerospace eXploration Agency\)](#), [energy \(HPC5 at ENI S.p.A\)](#), and automotive - but each of these systems are used to address a range of disciplines.

For example, consumer product HPC systems can simulate fluid flow, structural integrity, thermodynamics, various chemical processes, manufacturing tolerances for quality and cost, and product lifespan as only a portion of their total workload. Performing simulation across such a wide range of disciplines requires the HPC center to support many dozens of applications at a minimum.

Clearly, maximizing insights or throughput demands maximum speed from a wide array of HPC applications.

### *Maximize Efficiency of Expensive Resources*

We have posited that HPC systems are IP generators but need to recognize that they require a number of inputs to perform that valuable function. As machines, they require energy, maintenance, software, and cooling - but they also require information. The information must be arranged in a way acceptable to the HPC application, but also must be structured in such a way that human beings may understand and validate both the inputs and the application outputs.

Therefore, to optimize the efficiency of an HPC system, such factors and their associated costs need to be considered. The list below in order of cost, from most to least, suggests areas where attention should be paid for maximum benefit. Depending on the business, market, and technologies involved, the order of the following list may adjust slightly.

- Researcher / engineer / scientist time
- Software license time - some HPC software is commercial
- Instrument time - collider, microscope, sequencer, telescope, and so on
- Compute resource time

## What are the best performance indicators?

The best performance indicators are those that consider application throughput and job performance while counteracting inefficiencies that arise from parallelism.

### *Application Throughput vs. Job Performance*

For a given set of HPC applications and environment, there is another choice that is often left to users, which can significantly impact efficiency of an HPC system.

That is choosing how throughput and individual job performance are balanced.

Sometimes, this choice is easy. Some applications can only run single-threaded - this is still common in electronic design and genomics - and thus the scaling inefficiencies of parallelism do not come into play.

Because HPC centers are in the business of generating insight - often production deadlines, business goals, publication dates, and other outside effects require the HPC center to sacrifice their maximum throughput to meet the demands of these other factors. This is in efforts to prioritize and execute workloads at higher speeds and lower efficiencies by running them using multi-node parallelism, sometimes referred to as distributed memory parallel.

Most, but not all, of the widely used HPC applications can be run in this multi-node parallel fashion - and all of them are subject to [Amdahl's law](#).

### *Inefficiencies of Parallelism*

To the newly initiated, it may be a surprise to learn that parallel computing usually pays a penalty in efficiency. Gene Amdahl, the inventor of Amdahl's law, codified the mathematics to show how efficiency drops as a task is scaled up in parallelism. Figure 1 uses his equation.

*Note that Amdahl's law itself is an idealization, and real HPC simulation performance does not correlate perfectly due to violations of simplifying assumptions. The charts and discussion here are meant to illustrate the principles for approaching HPC.*

Figure 1. Speedup Factor Growth for Varying Parallel Fractions

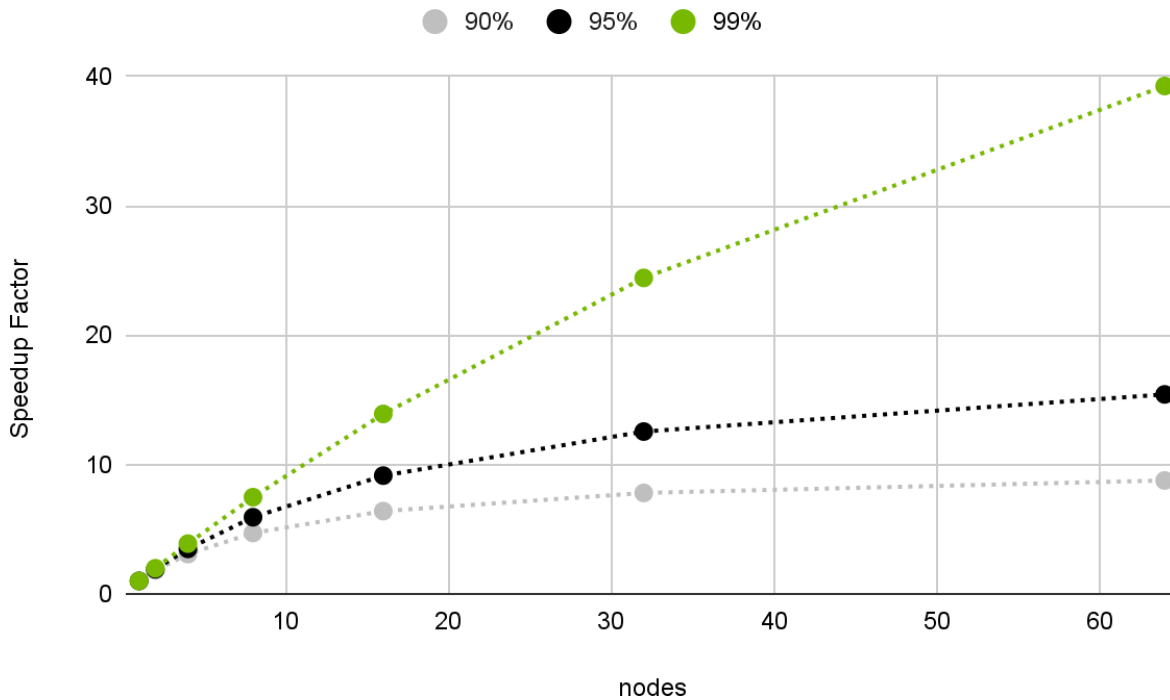


Figure 1 shows the growth of the Speedup Factor by using more compute resources for applications with varying parallel fractions.

To interpret Figure 1, let us review some definitions:

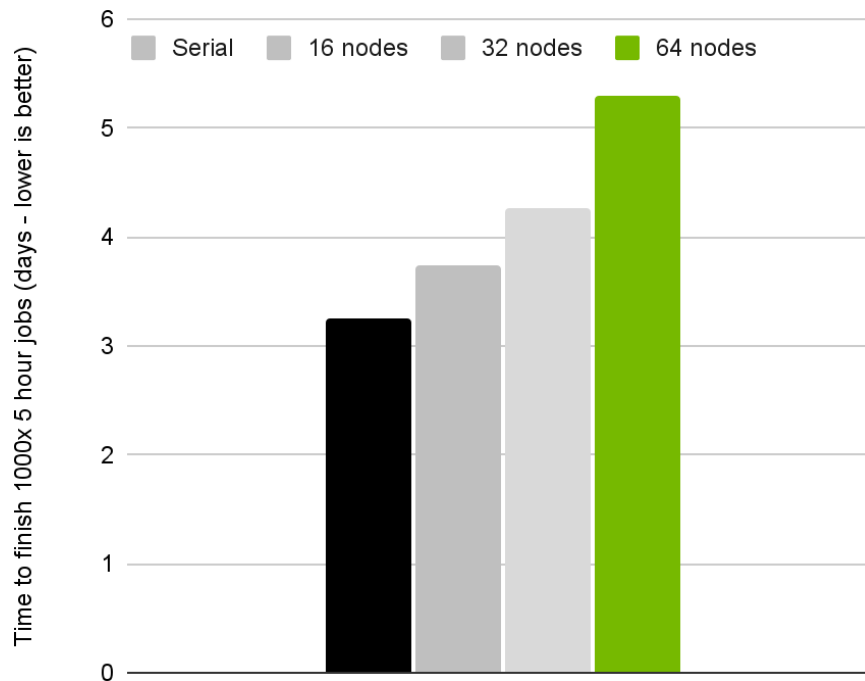
- *Speedup Factor*: the factor, which the runtime for one node is divided by, with bigger divisions being better.
- *Parallel Fraction*: the fraction of the task, in this case a simulation, which can be parallelized

In the above, the first thing that is apparent is that the larger the parallel fraction, the greater the speedup factor. This is true up to the limit of the task being 100% parallelizable. However, it should be noted that most tasks cannot reach 100%.

The chart also shows how achieving significant amounts of speedup at large node counts is dependent on getting the parallel fraction of a simulation as close to 100% as possible. This is the crucial efficiency compromise. If you examine the chart for a task, which is 95% parallel, you will note that there is nearly zero benefit (in terms of incremental speedup factor) going from 32 to 64 nodes. So, the additional 32 nodes being used provide almost no benefit in reducing the runtime.

This, of course, has implications in the quest for maximizing throughput. To make the discussion concrete - consider a simulation, which runs for 5 hours on one node and is 99% parallelized. Further consider a situation where a 64 node HPC environment must process 1,000 of these simulations as quickly as possible. The question is: How many nodes should each simulation run on to finish the 1,000 jobs quickest?

Figure 2. Time to complete 1000 5-hour jobs, using different amounts of parallelism



*Serial* means “using one node.”

Perhaps it is a surprise, but the fastest way to process all the simulations is to run each one on a single node, and run 64 of them simultaneously. So then, why would anyone ever run simulations in parallel? There are several reasons. Some are listed below.

The simplest explanation is that often, providing the user with some intermediate results before the entire 1,000 task workload is complete is valuable. In that case, it may be more efficient to optimize for the most expensive resource - a researcher’s time. At 16 nodes per job, the full 1,000 simulation workload takes about 12 hours longer to complete, but each simulation instead of running for 5 hours, completes in 21 minutes.

If the result of a single simulation is interesting and useful to the researcher - or more likely, the researcher has some post processing to perform on the simulation result, then it is very likely that optimizing the time to provide insight will include some multi-node parallelism.

Situations which demand use of multi-node parallelism are:



1. Large simulations that require large amounts of memory and cannot run on a single node often require parallel programming. Because parallelism often involves some type of domain decomposition, the amount of memory required per node is cut roughly by the number of nodes participating in the parallel task.
2. Similar to the previous case, sometimes users want to run simulations, which would take an unacceptable amount of time to run on a single node. In this case, they are forced to accept the inefficiencies of parallelism
3. There are cases where small amounts of parallelism scale slightly better than what is considered ideal. This means doubling the compute resources results in less than half the runtime and is often described as [super-linear scaling](#).

## Measuring Performance Indicators

Most people who have experience in HPC immediately think of the [Top500](#) and the associated HP-Linpack benchmark used to rank supercomputers for the last several decades. FP64 matrix operations are still a reasonable analog for many traditional 3D physics simulations commonplace in supercomputing centers worldwide.

However, the nature of HPC is changing. Data intensive workloads, mixed and reduced precision algorithms, and AI and machine learning (ML) are augmenting the way HPC is done to make the next leap forward in predictive speed and accuracy.

This section discusses some of the newer tools available for quantitative comparisons, which focus on some of the emerging methods and precisions not covered by HP-Linpack. We expect one or more of these measurements will become more commonly used to inform buying decisions for the next generation of supercomputers.

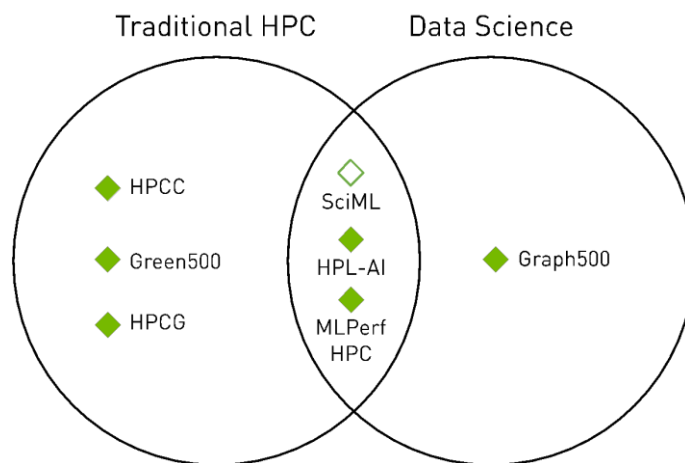
[Graph 500](#) - A focused data intensive workload meant to highlight the differences between data processing and 3D physics simulations. Both types of processing are important to HPC, yet data-focused workloads tend to get short shrift in the HPC community. Graph algorithms are a core part of many data intensive workloads and this benchmark is meant to balance the procurement comparison methods used during an HPC proposal process.

[Green 500](#) - A branch of the top500, which uses the HP-Linpack score submitted for the top500 divided by the peak watt usage during the run to provide a combined metric of FLOPS per Watt.

[HPC Challenge \(HPCC\)](#) - A suite of synthetic benchmarks which include the HP LINPACK, DGEMM, STREAM PTRANS, RandomAccess, FFT, and communications latency and bandwidth based on  $b_{eff}$ .

**HPCG** - High-Performance Conjugate Gradients (HPCG) is designed to exercise computational and data access patterns that more closely match a different and broad set of important HPC applications. HPCG is also designed to give incentive to computer system designers to invest in capabilities that will have an impact on the collective performance of these applications. Alternative operations included in HPCG include: sparse matrix-vector multiplication, vector updates, global dot products, and so on.

Figure 3. Evolution of HPC Benchmarks towards Data Science and AI/ML



**HPL-AI** - Is a synthetic benchmark similar to HPL, which does matrix math at 64-bit floating-point precision. However, the AI version does matrix math at lower levels of precision, recognizing that machine learning models often take advantage of lower precision math for training (fp32, 16, 8, and even int8) and achieve higher throughput without sacrificing accuracy.

**MLPerf HPC** - ML Commons is another consortium approach to defining groups of benchmarks focused on machine learning. Measurements for training and inference are separated, and a third, HPC-focused benchmark - which is machine learning training for HPC use cases - is defined and measured for both maximum performance - called strong scaling, and throughput - called weak scaling. See [NVIDIA's 2021 results](#) and [how they were achieved](#).

**SciML** - A showcase of examples for scientific technical computing making use of machine learning. This is currently a developing collection, indicated by the open diamond in the figure above, but can be used for benchmarking.

# Trends in High-performance Computing

## Emphasis on Energy Efficiency

Several factors are contributing to an awareness of the environmental impact computing has, the primary one being growth rate. Our lives are ever more wrapped up in a continuous stream of information that is collected, stored, culled, processed, selected, and presented to us every day. All this information distillation is driven by billions of processing and storage units scattered around the internet. And, those processing units are multiplying at an exponential rate in an effort to keep pace with our appetite for information.

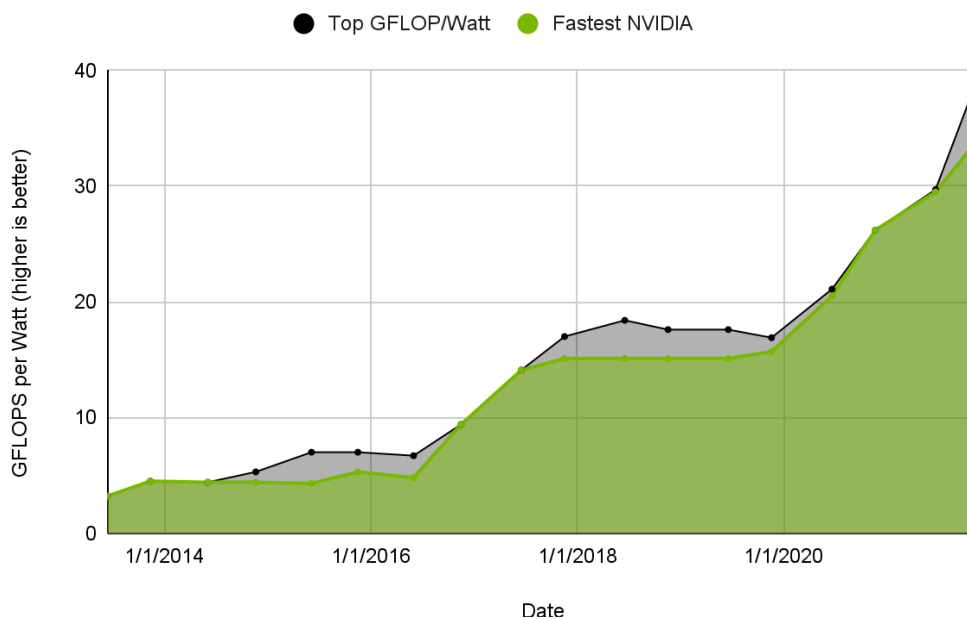
Though HPC is a small percentage of the total computing consumed by the world, it is used as a crucible to develop and prove technologies that get commoditized and eventually used by the most modest devices.

Recognition of this fact drove leaders in 2005 to launch a benchmark that tracks performance and power together - called the Green500 List. This list is now the principal form of recognition for leaders in HPC who made an effort to maximize the power efficiency of the HPC environments they manage.

The power consumed by HPC systems can be directly linked to CO<sub>2</sub> emissions in most locations that do not use carbon free energy systems such as nuclear, geothermal, wind, and solar. Due to the increased visibility and concern regarding climate change, leaders in computing have committed to aggressive schedules to get their enterprises to carbon neutrality. For most, this mission goes well beyond, but still includes, the data centers they manage.

Over the last decade, NVIDIA has spurred significant developments in efficiency by making GPU accelerated computing a standard throughout HPC. Figure 4 shows the increases in efficiency over time, and the track record NVIDIA has for being at or near the forefront of this important metric.

Figure 4. Recent 8-year History of the Green500 List vis a vis Overlay of Top NVIDIA GPU System Score



At GTC Fall 2021 - NVIDIA further committed to a major investment in understanding global climate change by pledging to build a new supercomputer dubbed “Earth-2.” NVIDIA President and CEO, Jensen Huang himself has [blogged about it](#). Earth-2 will use meter scale resolution simulations to inform a digital twin of the Earth’s climate. This scale will finally capture cloud formation physics, a crucial part to modeling reflected and absorbed sunlight as well as accurate long-term predictions.

## Reducing Carbon Emissions

Like many statements about humankind’s impact on the global climate, quantitative statistics vary. [One study from 2018](#) claimed that worldwide data centers consume 200 terawatt-hours of electricity, nearly 1% of the global demand, and contributed 0.3% of CO<sub>2</sub> emissions for that year. However, arguments have been made for and against how that calculation was made.

These arguments raise a few questions. Should it include the CO<sub>2</sub> associated with manufacturing the computers, storage, HVAC, and other components of the data center? Does it accurately account for the method of power generation in the local region or use a global average?

[Another study](#) on the environmental footprint of data centers focuses on the US, primarily because one third of all data centers in the world are located there. In that study, it was estimated that 0.5% of the total greenhouse gas emissions were attributable to data centers.

Yet another study alarmingly estimates that data centers are emitting roughly the same amount of CO<sub>2</sub> as the global airline industry.

The primary reason to focus on the impact of data centers, regardless of the studies above, is the fact that data center growth, led by the hyperscalars, is exponential. This growth far surpasses the rate at which computing is getting more efficient and, the only way to minimize the impact of these electricity gobbling sites, is to provide them with only clean renewable energy.

### *Water Cooling and Immersion Cooling*

The push for use of only clean renewable energy by data centers is an important one - getting there sooner by making data centers more efficient with their consumption is one approach to help us get there. A measure of data center energy efficiency is Power Usage Effectiveness (PUE). It is not perfect, but it attempts to quantify, in a fraction, the total energy coming into a data center divided by the power being supplied to the servers in that data center.

The goal is to push a data center PUE as close to 1.00 as possible so that all energy consumed is used to drive computing, and not for other parasitic needs such as cooling, air movement, water pumping, AC to DC conversion, and so on.

The vast majority of data centers today use cooled air to keep their equipment running within operating temperatures. This standard has led to development of several standard technologies and build practices that contribute to expense and inefficiency of data centers including:

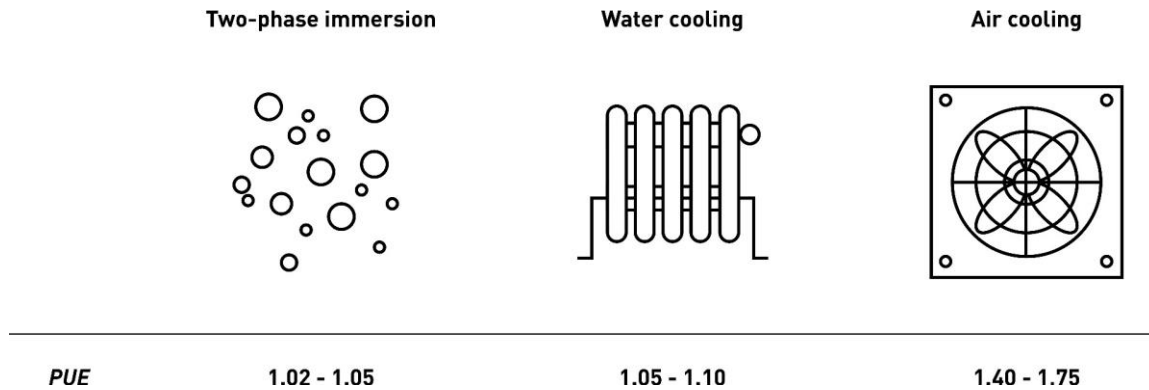
- Raised floors, which must withstand heavy rack weights
- Movable floor tiles
- In-data center Computer Room Air Conditioner (CRAC) units
- HVAC units to circulate air within the data center and under floors
- Noise and vibration conditions considered unhealthy for human beings

Unfortunately, the thermodynamics of air cooling make this an unavoidable and inefficient proposition - primarily because of the heat conductance and heat capacity of air.

Some forward-looking data centers have explored using direct water cooling to the power-hungry parts of a server, such as the CPUs, GPUs, memory, and networking. This is much more efficient than air cooling, but still requires pumps, chillers, and a significant amount of plumbing.

The most innovative data centers are exploring [immersion cooling](#) in which the entire server is immersed in some kind of heat conductive liquid that is electrically insulated. This is extremely advantageous due to the fact that thermal properties of a liquid are 1,000x better in terms of heat transfer and capacitance. Figure 5 shows the differences in PUE for data centers based on these technologies.

Figure 5. Comparison of data center cooling technologies PUE



A special case of immersion cooling, dubbed two-phase immersion cooling, uses special liquids, which boil at 45-55 degrees Celsius (113 - 131 degrees Fahrenheit). This type of immersion cooling is even more efficient because it takes advantage of the heat of vaporization of the cooling fluid. All the waste heat from server components goes to boiling the fluid. The vapor is then cycled to an ambient temperature condenser where it turns back to liquid and is returned to the cooling tub.

#### Advantages:

- Dramatic drop in data center PUE
- No longer requires raised flooring and is cheaper to build
- Standard immersive cooling does not require CRAC systems and can use simpler and cheaper heat exchangers
- The more exotic two-phase immersion cooling does not require heat exchangers at all
- Dramatic reduction or elimination of water usage

#### Disadvantages:

- Warranty and hardware availability certified for immersion
- Leaks, spills, and mess
- Weight
- Data center server density due to tubs of coolant that lay horizontally on the floor

Microsoft, for example, has demonstrated substantial energy reduction in cooling costs by submerging server racks in specially designed fluids. They also have been exploring [submerging small data centers in watertight containers](#) to the bottom of the sea.

## *ARM64 for HPC*

Since June of 2020, the fastest supercomputer in the world, according to the Top500, has been Fugaku, a megalith in the supercomputing world made from greater than 7.5 million cores and powered by 30 MegaWatts. However, the scale is not the only thing that makes Fugaku stand apart. It is powered by ARM64 processors, proprietary CPUs called the A64FX, and is made by Fujitsu.

At the time the A64FX was introduced in late 2019, it captured the top spot on the Green500 List, just a hair lower than 17 GFLOPs/Watt. Since then, semiconductor dies have shrunk, libraries have gotten more efficient at using processor hardware so that 17 (see Figure 5 above) is not a great score anymore. However, with all the new activities around the [Neowise core](#) from several private companies, and the introduction of accelerated computing for ARM64, you can expect great leaps forward in efficiency as this new HPC system architecture comes online.

One measure of an HPC platform being taken seriously by the community is the existence of important and common HPC applications being created for that platform. By that measure, [ARM64 is already being taken seriously as ports](#) for GAMESS, GROMACS, LAMMPS, OpenFOAM, WRF, NAMD, and MILC exist at the time this is being written. Other applications are nascent but may have development and testing versions being worked without community announcements.

A downstream measure of consideration, if applications exist for the platform, is if the platform is being selected by world leading supercomputing centers. ARM64 gets a checkmark there too, as NVIDIA announced their [Grace processor](#) at GTC Spring 2021 and also announced a [supercomputer win at Swiss National Computing Center](#) for the platform with HPE.

However, NVIDIA is not the only company publicly talking about an ARM64-based processor targeted at HPC. Since regional HPC technology capabilities are global economic and sovereignty bulwarks, the private company SiPearl is launching [a European consortium called the European Processor Initiative](#). SiPearl's vision for the EPI project is to design their own high-performance and low-power microprocessor for European supercomputers along with surrounding HPC technologies and applications. Further, Amazon Web Services have their Graviton instances, which are ARM64 based, currently on their second generation. Expectations are that a third-generation Graviton will focus on price-performance conscious HPC cases in the cloud.

## *GPU Power Tuning*

Both Intel and AMD offer CPU stepping technologies, which allow the administrator to step up and step down the CPU frequency at various granularities. This helps to maximize performance for what is running, or control the job performance per watt of energy expended. Of course, doing this at HPC center scale, for a wide array of HPC applications, all automatically, is a very

challenging and time consuming process to measure, codify, setup, and maintain. Not to mention, one which would be repeated for each installation every 3-5 years. So, for the most part, only the most environmentally minded HPC centers attempt it.

Similar control is available on NVIDIA GPUs as well. In fact, just recently the [University of Cambridge in the UK installed a DGX-A100 system](#) where they used this frequency control technology to de-clock their A100's to save 35-40% on power consumption while losing only 10-11% on performance.

Generally speaking, this sounds very compelling when optimizing an HPC app for GFLOPS per Watt and hitting a high score on the Green500 List. However, to be of real benefit, the total power consumed by a task needs to be optimized, not the rate of consumption. Meaning, reducing the power consumption rate of a task by 50% is great, but if the task takes 75% longer to run, then the total power consumed is greater than running at full speed and power.

Once again, the only way to know what approach is best is to do the benchmarking measurements of applications at various GPU frequencies, measuring the performance (run time) and the power (consumption rate). Armed with that data, optimization is entirely within grasp.

For those interested, this kind of control is available through the [NVIDIA System Management Interface](#) (SMI) as shown in the code snippets below:

```
-lgc --lock-gpu-clocks= Specifies <minGpuClock,maxGpuClock> clocks as  
a  
pair (e.g. 1500,1500) that defines the range  
of desired locked GPU clock speed in MHz.  
Setting this will supercede application clocks  
and take effect regardless if an app is running.  
Input can also be a singular desired clock value  
(e.g. <GpuClockValue>).
```

For example:

```
# nvidia-smi -pm 1  
# nvidia-smi -i 0 -pl 250  
# nvidia-smi -i 0 -lgc 1090,1355  
DISPLAY=:0 XAUTHORITY=/run/user/129/gdm/Xauthority  
nvidia-settings \  
-a [gpu:0]/GPUFanControlState=1 \  
-a [fan:0]/GPUPowerMizerMode=1 \  
-a [fan:1]/GPUPowerMizerMode=1 \  
-a [gpu:0]/GPUPowerMizerMode=1 \  
-a [fan:0]/GPUFanControlState=1 \  
-a [fan:1]/GPUFanControlState=1
```



```
-a [gpu:0]/GPUGraphicsClockOffsetAllPerformanceLevels=75
```

## *Direct Current Powered Data Centers*

More than a hundred years ago, Thomas Edison and Nikola Tesla were engaged in an intellectual and business war for the soul of electrification of the United States. In the end, alternating current (AC) won out because of the efficiencies of distribution for large amounts of power. This conclusion is not expected to change because, other than solar, power generation is not accomplished within 500 meters of where most people live or work.

The issue comes from the fact that the information age and its key enabling technology - the transistor - operate on direct current (DC). As such, every digital circuit must have a rectifier (AC to DC power supply) upstream to provide the needed type of power. Consequently, virtually every data center component from a network switch to a mainframe has an embedded, and often redundant, power supply performing the conversion. Unfortunately, physics dictates that energy conversion is not perfectly efficient and will generate waste in the form of heat in the process. Naturally, the more times power is converted, the larger the losses will be. This waste heat inside every computer is added to the other heat sources and must be mitigated by the air-cooled data center HVAC and CRAC units.

It is possible however, to convert AC high voltage and current to DC at the rack or row level - and distribute DC power directly to each server, switch, and storage unit. This technique has some laudable benefits:

- Takes the rectifiers (power supplies) out of servers, which remove heat, noise sources, and restrictions on the airflow for cooling other server components
- Removes a conversion step from DC to AC when UPS backup is in operation
- Corrects an omission in the PUE definition (some power is lost in the AC to DC conversion)
- Eliminates myriad redundant power supplies and makes centralized AC to DC converters highly reliable

In the quest for higher PUE and optimized GigaFLOPS per Watt, reimaging power distribution may be another tool worth exploring.

## **Data Driven Discovery**

The world around us is complex and with the digital device explosion, we are increasingly flooded with data of every conceivable type. The same digital revolution is changing the nature of how science, or the practice of scientific data collection, is done.

To back up a bit, understanding the world at the fundamental physical law level relies on the construction of empirical models. These models describe complex physical systems that enable scientists to predict behaviors and answer “what-if” questions. Such questions become

experiments, which seek to confirm, or deny, the validity of the empirical model. This is what we call the scientific method.

Today, human beings attempt to hypothesize and understand new physics based on years of subject matter experience, intuition, and a sense of mathematical elegance. Generally, two complementary methods have emerged - the theoreticians who work from a purely symbolic framework and the experimentalists who gather observations in harmony or discord with theory. The give and take between these two advance our understanding of the world around us incrementally.

However, a new opportunity is emerging with the practical application of machine learning techniques. With more data than ever becoming available through improved sensing, ubiquitous high-bandwidth networking, and free compendiums of data, the opportunity exists to build models to speed scientific discovery through an automated inductive reasoning approach.

In physics, the language of the universe is written in Partial Differential Equations (PDE). Electro-magnetism, gravitation, fluid-dynamics, geo-mechanics, relativity, and many others all use PDEs to describe the phenomena they attempt to capture. The PDEs we use today were developed by a handful of thought leaders in each field. But now, with the explosion of data, there is a new method to discover new descriptions of universal mechanics. Verifying this method has promise, and fluid mechanics, a field which has been studied for a long time, was used as a [test case](#) to see if the method could derive the governing PDEs purely from observations. In fact, [California Institute of Technology has a center](#) dedicated to leveraging data to derive new insights with several interesting projects.

In another example, geoscientists are putting this method to work in an effort to build a [predictive model of the Earth](#). This is not only from a climate standpoint, but from a full biome point of view, to minimize the impact of natural and human caused disasters such as earthquakes, tsunamis, and hurricanes.

## The Rise of Dedicated Silicon Acceleration Technologies

Accelerated computing can deliver many improvements in performance and energy efficiency compared to CPU-only approaches, and many HPC workloads are well-suited to take advantage of the capabilities of accelerators.

There are many types of accelerators such as GPUs, FPGAs, and ASICs, that differently balance flexibility, performance, and energy efficiency. Gains along one dimension can potentially come at the expense of another.

HPC applications are diverse. They require any HPC-focused accelerator to be highly programmable and able to deliver significant speed-ups across a wide range of operation types

while ensuring that the platform does not constrain users. At the same time, given the increased emphasis on energy efficiency in HPC, acceleration technologies must maximize the benefit (performance) for the power they consume.

Building the right accelerator for HPC requires the optimal balance of performance, energy efficiency, programmability, and flexibility, which currently has led to the GPU to emerge as the accelerator of choice in HPC.

Modern GPUs, such as [NVIDIA's latest H100](#), accelerate both vector and matrix operations spanning a range of numerical formats, from double-precision floating point all the way to 8-bit integer and floating-point formats. Proof of value is in adoption statistics. NVIDIA GPUs accelerate over 700 HPC applications and span a wide range of domains, including engineering, healthcare and life sciences, astrophysics, and high energy physics (See Figure 7 for specific highlights).

Another notable acceleration technology for HPC is wide vector units incorporated as part of general-purpose CPUs. This approach is employed, for instance, by Fujitsu's A64FX, the Arm-compatible processor at the heart of the Fugaku supercomputer.

Additionally, AI is increasingly playing an important role at the forefront of scientific research and discovery, which is accelerating the convergence of HPC and AI. GPUs are the most popular accelerators for AI workloads, and modern GPUs, such as the [NVIDIA A100](#) and H100, incorporate acceleration for the matrix math operations that are core to both AI training and inference.

The convergence of HPC and AI has also led to some interest in the application of specialized AI accelerators to HPC workloads. For example, Lawrence Livermore National Laboratory, has announced projects with AI accelerator startups such as [Cerebras Systems](#) as well as [SambaNova Systems](#). [Graphcore](#), another vendor of specialized AI accelerators, has also discussed using its products to advance portions of HPC workloads that take advantage of machine learning.

## Pushing HPC to Data Sources (Edge)

First, let us define what we mean by “edge” as in, “edge computing.” Edge computing physically distributes computation engines to locations adjacent to sources of data.

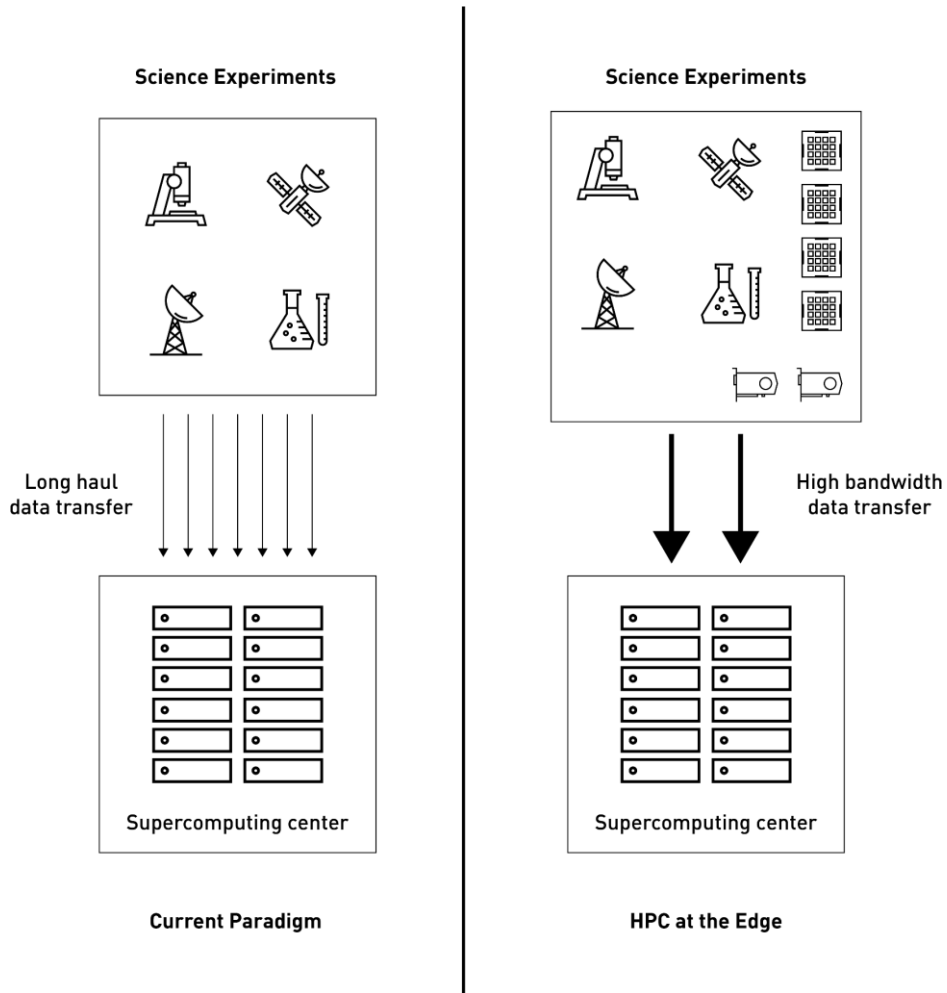
Advantages:

1. Preprocessing at the data source can provide a filtering effect where some or most data does not need to be transferred or processed further
2. Latency for responses to data is reduced leading to greater productivity and better results

3. Enables scalability to handle growth in the data volumes by processing it at the source
4. Platforms are individually more cost effective because they allow for granular scaling of resource investments to match data growth instead of upgrading central facilities

**Edge computing** enables integration of experimentation and theory accelerating the new paradigm for scientific experimentation. Simulation predictions may inform new experiments and improve observations while the new observed data may correct the models rapidly to improve the simulation models. In some cases, simulations optimize data acquisition in a closed loop by directing digitally driven experimental apparatus. Data gathered can then be fed back into simulation surrogates to adjust and acquire more data to maximize accuracy or minimize uncertainty.

Figure 6. Current And Future Paradigms for Experimental Data Processing and Edge HPC



However, most cutting-edge science experiments require computationally, data intensive, application-capable infrastructure to process near real time even when using surrogates. This is

where the need for high-performance computing arises. Supercomputing is incorporating HPC at the edge as leading research installations are undertaking large instrument upgrades with their exascale computing upgrades. Developments in deep learning methods and edge processing are anticipated to help make maximum use of upgrades and computational challenges at the [Advanced Photon Source Upgrade \(APS-U\), the European Synchrotron Research Facility Extremely Brilliant Source \(ESRF-EBS\) and PETRA-III](#). In addition, this is recognized by world leading science experiments such as the Large Hadron Collider, the Allen Telescope Array, the Square Kilometer Array, and several others.

More traditional supercomputing workflows that serially ingest data transferred from instruments are being re-factored using mashups of continuous data stream processing, machine learning training and inference tools, and visualization at the instrument. These re-imagined workflows are right sized for current scientific needs, while simultaneously preparing for instrument upgrades and the associated jump in data ingest rates and processing requirements.

## What is quantum computing?

[Quantum computing](#) leverages the quantum mechanical phenomena of superposition, interference, and entanglement to perform calculations. Quantum computers are believed to be capable of solving some problems faster than classical (non-quantum) computers.

There are a few quantum algorithms that are known to have better scaling behavior on a quantum computer over the best-known classical algorithm. This includes Shor's algorithm for finding the prime factors of an integer, and Grover's algorithm for searching an unstructured data set. Quantum computers are also believed to be better suited than classical computers for simulating large quantum systems, which would have valuable applications in many fields, including drug development and energy. There are a wide variety of other proposed applications of quantum computers, including optimization and machine learning problems. Where and when we should expect quantum advantage is a subject of intense active research.

There has been great progress in the past 20 years in the development of several different physical implementations of a quantum computer, and today we even have quantum computers that can outperform classical supercomputers on a couple of very specific tasks. However, these tasks are abstract and esoteric, and a quantum computer that can outperform classical computers on a task with any practical application is very likely still a long way off.

One major issue is scale. Today's quantum computers are generally in the range of 5-100 qubits, whereas solving Shor's algorithm or simulating a useful molecule will require thousands to millions of qubits. Another major issue is fidelity. Due to the sensitive and analog nature of quantum computers, they are inherently error-prone, and errors in each logical operation accumulate through the duration of a quantum circuit. The error rates of today's quantum

computers limit them to running only a few logical operations with consistency before accumulated errors lead to a random result. This is impractical as applications with known algorithmic speedups require thousands to millions of operations to be run.

The intense research to understand how quantum algorithms perform compared to the best classical algorithms, combined with the limited power of today's quantum computers, means that powerful simulation tools are essential. Researchers today are using [simulators to develop, test, and debug new quantum algorithms](#) at a scale and performance far beyond what is possible on today's quantum computers.

Simulating quantum computers with classical computers requires tremendous computational power, but as many of the subtasks are in parallel, it is a natural fit for GPUs. The ability to effectively simulate quantum computers is expected to lead to new insights and breakthroughs in our understanding of how to get the most out of the quantum computers of the future. NVIDIA offers a [cuQuantum library](#) to assist and accelerate the next generation of simulators.

It is also widely accepted that all valuable applications of quantum computers will be in a hybrid computing model, where a quantum computer performs some tasks working in tandem with classical computers. Today, leadership computing facilities are looking at how to integrate future quantum computers with other HPC resources as part of a hybrid data center. Such an integrated system will require a unified programming model and software toolchain, enabling domain scientists to program seamlessly and integrate quantum computers into existing workflows in a transparent and performant manner.

## Rise of the Metaverse and Digital Twins

The term “digital twin” has been taken and used by analysts and media alike, all with varying definitions. To some, a digital twin is just another name for a digital model of an object and some of its properties. It could be a bicycle going over a bump or it could be a tritium atom subjected to the conditions of a [tokamak](#). Generally then, these models are subjected to physics simulations which predict behavior. The challenge here though is that such a digital twin:

- Does not respond in real time
- Models only certain physics
- Does not ingest and digitally represent data from sensors

However, some maintain that [second-generation digital twins](#) combine operational data from connected assets with model-based physics, based on numerical simulation.

We agree but believe the path to the second-generation digital twin will require a major step forward in predicting a model response to stimuli. Numerical methods for solving approximations to physics PDEs, even running on accelerated hardware, simply aren't fast enough to produce results in real time, nor do we expect they will achieve this feat anytime soon.

To get there, we need something that can estimate the physics in the digital twin without resorting to running a numerical simulation. [By injecting AI technology into the backend of a digital twin](#), there are multiple methods by which the speed required can be achieved while maintaining enough accuracy for the digital twin to be valuable.

The simulators and simulation data associated with archived analyses and previous designs all can be leveraged as training material for an AI model, which in turn can be used to mimic the simulator output and drive the digital twin in real time. NVIDIA and SIEMENS Energy have demonstrated a similar [digital twin approach using AI to model a combined cycle power plant](#).

A subsequent section entitled; "[Equation Enhanced Machine Learning](#)," describes a newer technique by which an AI learns by using physics equations as constraints during the training process. By doing this, as the AI model matures, the predictions it makes obey the physical laws used during training.

NVIDIA announced to the world the intention to build a full [Earth digital twin](#) for detailed modeling of climate in hopes of accurately predicting weather in 30-60 years, and also model mitigation strategies. For climate models to render clouds well, the mesh of the Earth's atmosphere needs to be on the 1-meter scale. The current rate of computing acceleration predicts that it will be another 40 years before we can run a model at such a resolution with traditional numerical models. So again, we appeal to the need for AI to make the digital twin useful.

NVIDIA is offering another tool, [Modulus](#), to assist in building data and physics-based machine learning models, which we believe will be the key technology to making the "Earth-2" digital twin a near term reality.

# High-Performance Computing Key Enablers

## HPC Application Support and Breadth

The NVIDIA platform enables generations of novel insight on many levels:

- Simulating intelligence with AI frameworks.
- Simulating microscale and macroscale physics-based on first-principles algorithms.
- Simulating visual representations of natural phenomena with computer graphics.

Today, there are over 2,500 [GPU-accelerated applications](#) across a broad spectrum of scientific domains such as material science, quantum chemistry, seismic, weather, and climate. And now, with NVIDIA Quantum [InfiniBand technology](#), the number of applications which leverage our low latency, high-bandwidth interconnect - which is crucial for many HPC simulators - is even higher.

*Table 1. HPC Applications by Discipline  
Note: Green text denotes GPU-accelerated applications*







Fluid Mechanics	Physics	Media and Entertainment	Structural Mechanics
<ul style="list-style-type: none"> <li>• <b>Fluent</b></li> <li>• <b>FUN3D</b></li> <li>• <b>openFOAM</b></li> <li>• <b>PowerFLOW</b></li> <li>• <b>StarCCM+</b></li> </ul>	<ul style="list-style-type: none"> <li>• <b>BQCD</b></li> <li>• <b>Chroma</b></li> <li>• <b>CPS</b></li> <li>• <b>GENE</b></li> <li>• <b>MILC</b></li> <li>• <b>XGC</b></li> </ul>	<ul style="list-style-type: none"> <li>• <b>3ds Max</b></li> <li>• <b>Arnold</b></li> <li>• <b>Blender</b></li> <li>• <b>Maya</b></li> <li>• <b>Renderman</b></li> </ul>	<ul style="list-style-type: none"> <li>• <b>ABAQUS</b></li> <li>• <b>Ansys Mech.</b></li> <li>• <b>CTS</b></li> <li>• <b>LSDyna</b></li> <li>• <b>NX Nastran</b></li> </ul>
Seismology	Electronics Design	Climate and Weather	Cosmology
<ul style="list-style-type: none"> <li>• <b>DELFI</b></li> <li>• <b>ECLIPSE</b></li> <li>• <b>Nexus</b></li> <li>• <b>SpecFEM3D</b></li> <li>• <b>tNavigator</b></li> </ul>	<ul style="list-style-type: none"> <li>• <b>Allegro</b></li> <li>• <b>Clarity</b></li> <li>• <b>Icepack</b></li> <li>• <b>NCSim</b></li> <li>• <b>VCS</b></li> </ul>	<ul style="list-style-type: none"> <li>• <b>GRAF</b></li> <li>• <b>ICON</b></li> <li>• <b>IFS</b></li> <li>• <b>MPAS</b></li> <li>• <b>WRF</b></li> </ul>	<ul style="list-style-type: none"> <li>• <b>CASTRO</b></li> <li>• <b>Cholla</b></li> <li>• <b>Gamer</b></li> <li>• <b>HACC</b></li> <li>• <b>RAMSES</b></li> </ul>



Financial Services	AI/ML/DL Data Science	Genomics and Proteomics	Quantum Chemistry
<ul style="list-style-type: none"> <li>• Algorithmics</li> <li>• Calypso</li> <li>• MACS</li> <li>• MOSES</li> <li>• Pathwise</li> <li>• TowersWatson</li> </ul>	<ul style="list-style-type: none"> <li>• MXNet</li> <li>• PyTorch</li> <li>• TensorFLOW</li> </ul>	<ul style="list-style-type: none"> <li>• GLIDE</li> <li>• GPUBlast</li> <li>• HMMER</li> <li>• NCBI Blast</li> <li>• openFOLD</li> <li>• RosettaFOLD</li> </ul>	<ul style="list-style-type: none"> <li>• CP2K</li> <li>• GAMESS</li> <li>• Gaussian</li> <li>• VASP</li> <li>• NWChem</li> <li>• NWChemEX</li> </ul>
Molecular Dynamics		Computational Chemistry	
<ul style="list-style-type: none"> <li>• AMBER</li> <li>• CHARMM</li> <li>• FEP+</li> <li>• GROMACS</li> <li>• LAMMPS</li> <li>• NAMD</li> <li>• OpenMM</li> </ul>		<ul style="list-style-type: none"> <li>• DeepMDKit</li> </ul>	

Simulations and simulated environments are rapidly becoming the focus for revolutionizing entire industries. Table 2 provides just a few examples of simulation environments available from NVIDIA.

Table 2. Industry-wise NVIDIA Simulation Tools and Environments

 Transportation	 Healthcare	 Commerce	 Manufacturing	 Robotics	 Quantum Computing
<a href="#">NVIDIA DRIVE</a>	<a href="#">NVIDIA CLARA</a>	<a href="#">NVIDIA MERLIN</a>	<a href="#">NVIDIA OMNIVERSE</a>	<a href="#">NVIDIA ISAAC</a>	<a href="#">NVIDIA cuQUANTUM</a>
End-to-end solutions for autonomous vehicles.	An application framework optimized for healthcare and life sciences developers.	An open-source framework for building high-performing recommender systems at scale.	A scalable, multi-GPU real-time reference development platform for 3D simulation and design collaboration.	An accelerated platform for robotics and AI.	An SDK of libraries and tools for accelerating quantum computing workflows

## HPC and AI Convergence

From early human use of stone tools to instruments so sensitive they can measure ripples in space-time, technology can be modeled as a complex ladder, where one invention enables and gives rise to the next.

Sometimes seemingly disparate innovations must be combined to take a new step forward. Today, data science and AI are fusing with traditional HPC technologies in an effort to discover solutions for areas as diverse as automotive safety to interest rate prediction.

Now that AI techniques are rapidly maturing, there is a new way to produce results, which has the potential to be both faster and potentially more accurate than previous methods. Simultaneously, there are several ways in which this powerful technology can be applied, all with the goal of providing the user with results based on their input, as soon as possible with as much accuracy as possible.

### Speeding Simulations

The process of building an AI model, specifically training, is a compute-dominated operation. Thankfully, optimizing compute speed is almost in the name of high-performance computing. Unfortunately, for CPUs, that optimization has all but played out in the form of Moore's Law and no longer seems to hold.

Figure 7. Moore's Law Obsoleted in the Context of GPU Computing, AI, and ML

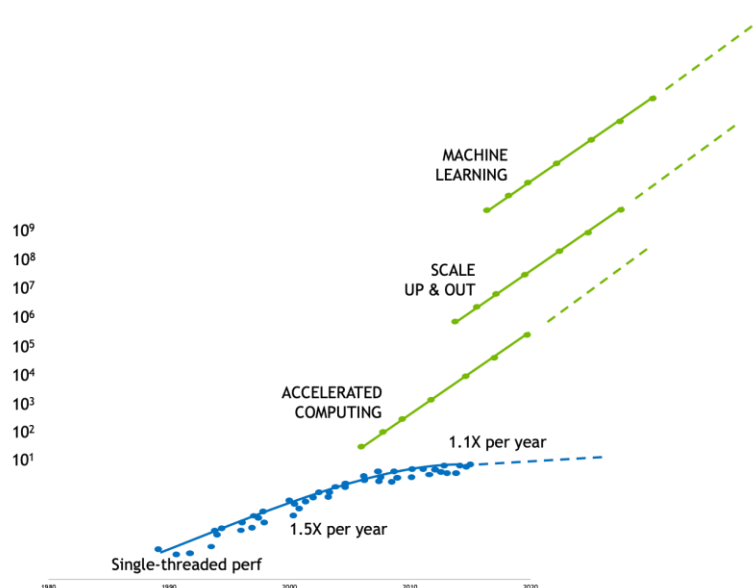


Figure 7 illustrates how the track of CPUs, as they relate to Moore's Law, is nearly obsolete compared to the superior slope of GPU-accelerated computing technology including accelerated

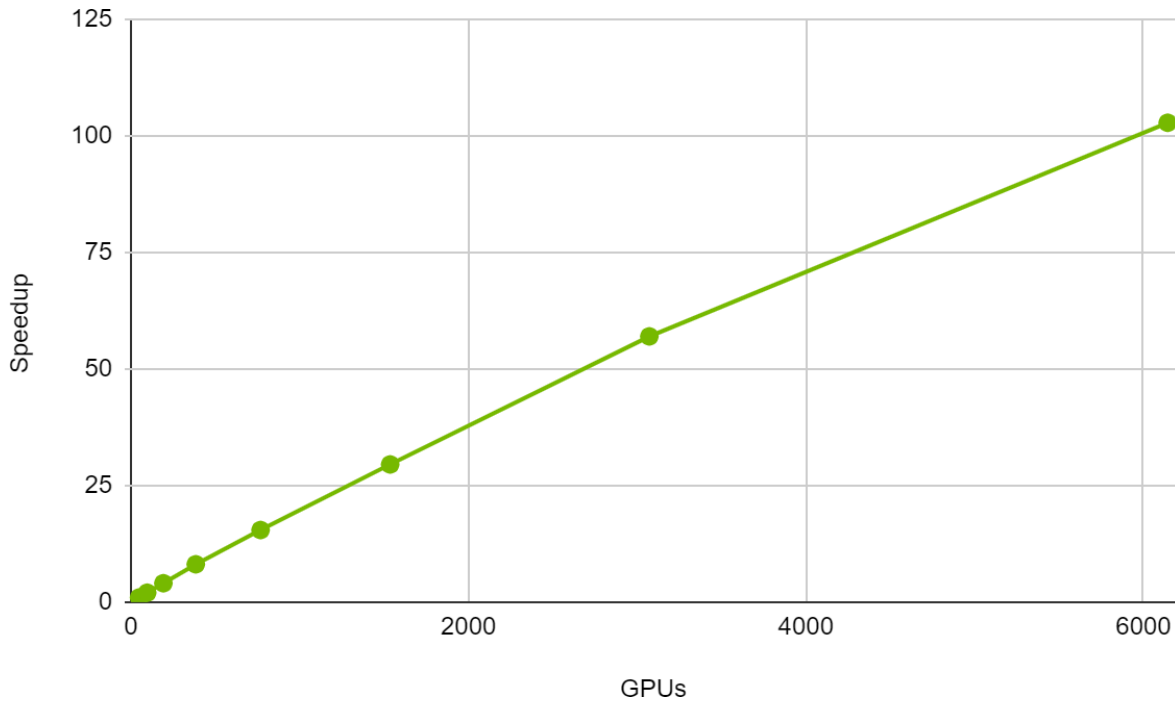
computing, AI, and ML. CPUs, which are governed by Moore's Law, are at a disadvantage compared to GPU-accelerated computing technology, which is on a much steeper growth curve with more capability.

In the past two years, the combination of AI and HPC has produced breakthroughs which would not have been possible using accelerated computing alone. In one example, in 2020 a team of researchers from the US and China combined a machine learning model called "DeepMD-kit" with a molecular dynamics simulation tool called LAMMPS to [simulate 100 million atoms simultaneously](#). The machine learning model calculates the interatomic forces, which in turn drive the dynamics of the system more than 1,000 times faster than standard methods can do and thereby allows the simulated system to be an order of magnitude larger in size.

Shown in Figure 8, [Huerta](#) and associates built and trained an AI model to characterize the signal, which comes from a pair of orbiting black holes as they spiral into each other and cause space-time itself to ring like a bell.

This model was trained with time series data that describes the gravitational wave signals generated in the last few moments before the black holes merge and go silent. The connection to HPC here is that this training data was not obtained by scientific measurement - it was simulation-generated data, which mimicked what an experiment would measure.

Generating this training data required a large astrophysics simulation model and was scaled on an exceptionally large HPC system. This significantly reduced the wall clock time to generate the data and thereby train the model, resulting in an impressive reduction from 24 days on four GPUs to 1.2 hours on 1,536 GPUs. Figure 8 illustrates the scaling of machine learning models' training data generations using traditional HPC.

*Figure 8. Training Data Speedup Using Traditional HPC*

**Source:** Adapted from data and graph presented in [Convergence of artificial intelligence and high performance computing on NSF-supported cyberinfrastructure | Journal of Big Data | Full Text \(springeropen.com\)](#).

This method of training a machine learning model to do the work of a simulation is called a “surrogate model” and is one of several different methods for applying AI to traditional HPC. The two examples above show how HPC and AI come together to make individual simulations run faster. In figure 9 below, this is the purple end of the spectrum.

Figure 9. AI Applied to HPC

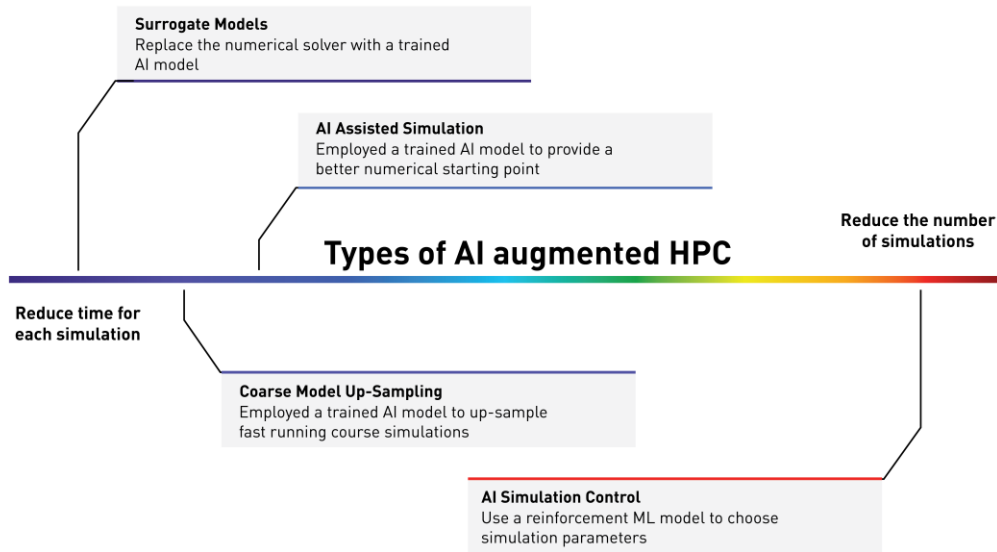


Figure 9 shows a spectrum of possible ways to apply AI in HPC. Most methods focus on acceleration of a single simulation and, when aggregated, provide better user service, which yields faster time to results and higher throughput overall for the HPC environment. AI can be applied to traditional HPC workflows in a spectrum of ways. Some accelerate individual simulations, others minimize the number of simulations.

The red side of the spectrum is an area, which focuses more on efficiency than simulation acceleration. Efficiency should be considered when HPC users are faced with the challenge of multi-parameter optimization. This occurs when they have a method to judge if a particular set of parameters to be better or worse from the results of a simulation, but need to make hundreds or thousands of simulations to explore and optimize the values of these parameters in an allowable design space. Sometimes these problems are referred to as “design of experiments” (DOE), where an experiment in this case is equivalent to a simulation.

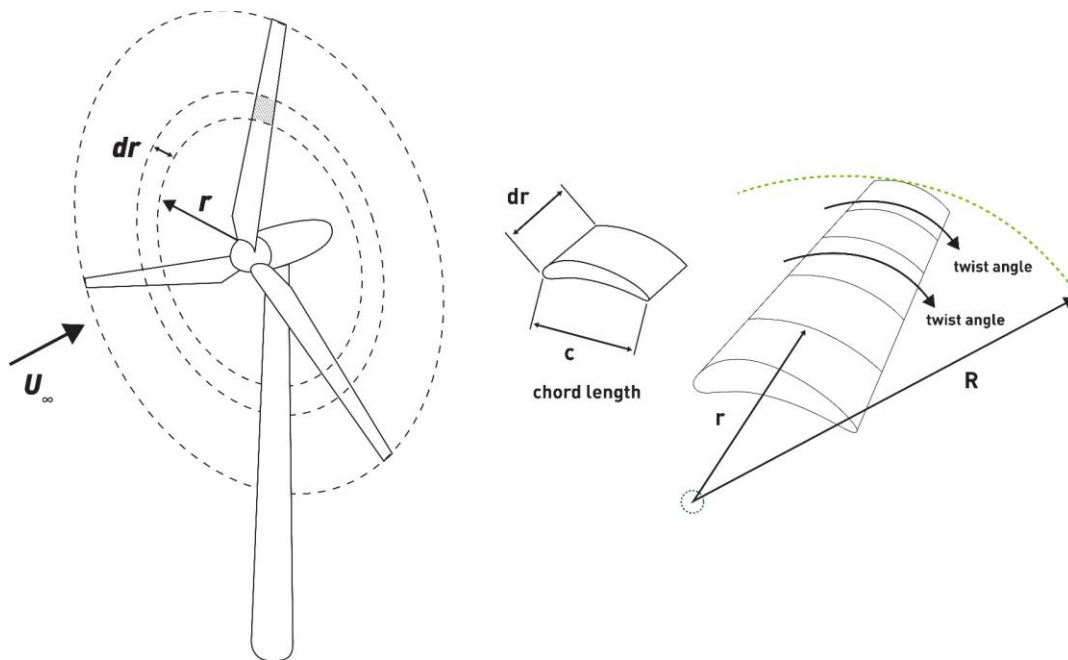
DOE is very common in commercial HPC where simulation is used as a product design tool. Industry examples include aerospace, drug development, automotive, electronic design, and more. To make this kind of application of AI to HPC more concrete, consider the following example:

Imagine an engineer designing a wind turbine. There are many parameters as shown in Figure 10 below, which describes the turbine from the number of blades, the diameter, the airfoil, twist,

root chord, tip chord, taper of each blade, to name only a few. Generic parameterization of a wind turbine is an example of design space exploration - another opportunity for AI in HPC.

In addition, there are a host of constraints, which limit the design choices from maximum tip speed, cost, material availability and structural limits, to tower moments. Given a statistical model of wind direction and speed by altitude, with the help of a simulator, the engineer can calculate the torque generated by the turbine at the hub, which determines the power output.

Figure 10. Wind Turbine - An Opportunity for AI in HPC



Often, the engineer's task is to choose optimal values of the parameters, which describe the turbine to maximize the power output within a margin of safety while minimizing costs. To make the choice, they would run hundreds of simulations to explore the design space and approximate the best choice. AI can be used to minimize the number of simulations to reach an optimal design set without ever being involved in the simulations themselves.

This method is showing up in many locations, from [Astra Zeneca's automated rapid drug discovery system](#) to a fully [3D printed AI designed car](#).

### Equation Enhanced Machine Learning

Most machine learning models, such as Deep Neural Networks (DNNs) or Convolutional Neural Networks (CNNs), take their primary input in the form of specific examples assembled from structured data. These examples must be curated and labeled before being fed to the AI model during the training process. The process can be slow and labor intensive because training this

way often requires hundreds of thousands, sometimes several million, examples to train a model to usable accuracy.

In some cases, these kinds of methods produce trained models, which work well for their training sets, but when confronted with new data, can produce results, which do not conform to generalized rules.

A classic example of this problem is creating an [AI model for fluid flow](#), where the model, given some geometry, produces a prediction of velocities, directions, and pressures within the flow volume. Though streamlines and pressure may appear correct at the qualitative level, when basic principles like conservation of mass or energy in the flow field are investigated for the AI produced solution - the prediction is found to violate these simple laws of physics and therefore cannot be trusted.

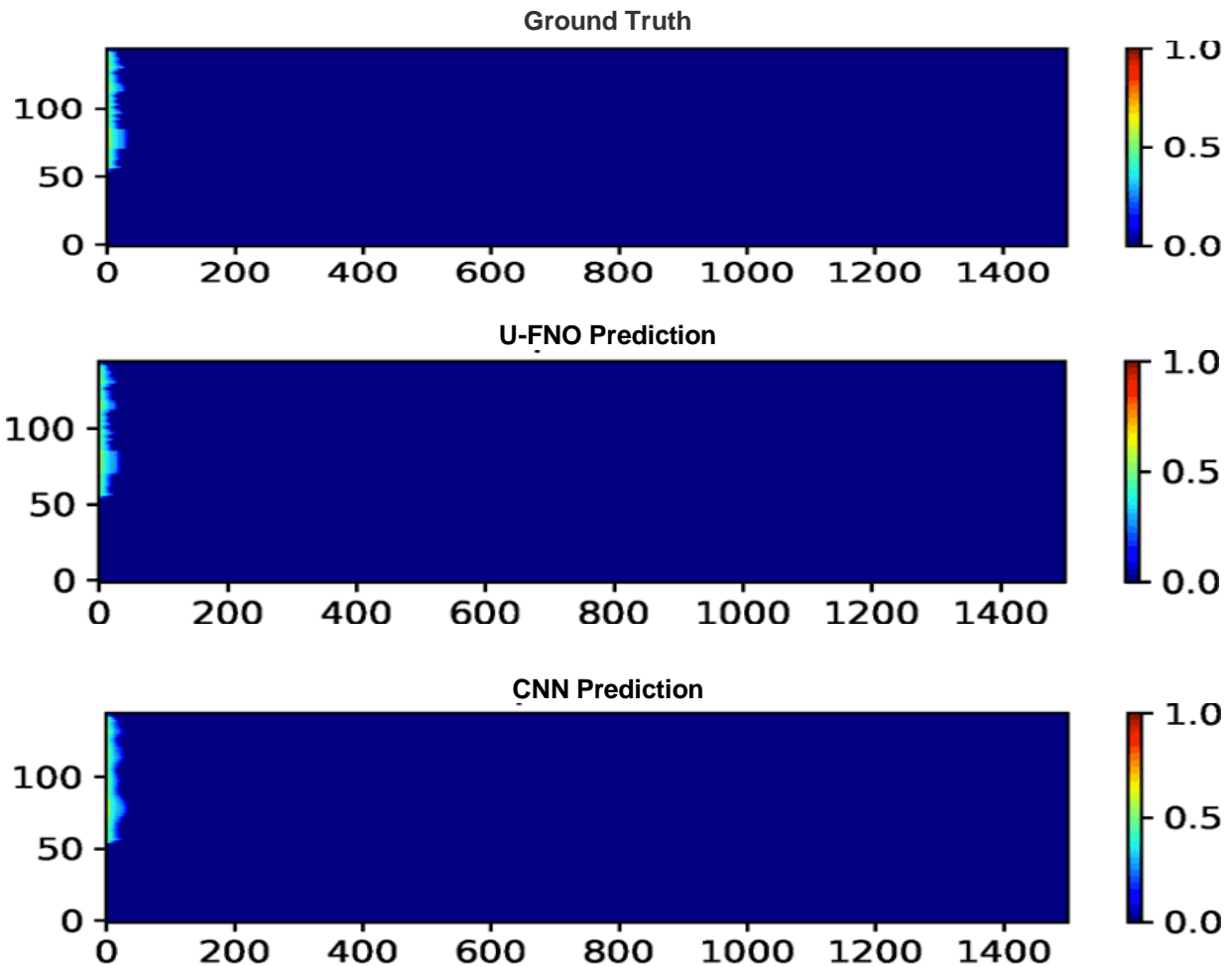
Ingenious researchers, such as Paris Perdikaris, Assistant Professor, University of Pennsylvania, have [suggested an alternate method](#), which uses equations - in this case, physics equations, conservations of mass and momentum - in the loss function during training. There are several examples of this kind of approach:

- Physics Informed Neural Networks (PINNs)
- Fourier Neural Operator or Unet modified Fourier Neural Operator (FNO / U-FNO)

In the case of the U-Net modified FNO, Gege Wen, Ph.D. Candidate, Stanford University, and collaborators have [shown this accuracy difference for a case of CO<sub>2</sub> saturation over time](#) for an Underground Carbon Sequestration case. In this study, they compared their model against both a traditional PDE solver approach as well as a convolutional neural network.

Figure 11 illustrates the differences in the predicted saturation of carbon dioxide over a period of 30 years for the reference model (PDE based numerical solution), Unet-modified FNO, and a convolutional neural network. Watch the [animated version of Figure 11 here](#).

Figure 11. Convolutional Neural Network Prediction Compared to the Ground Truth and Unet-modified FNO Prediction



## Transforming HPC

A [transformer model](#) is a neural network that learns context and thus, meaning, by tracking relationships in sequential data like the words in this sentence, or basepairs in a DNA sequence. First described in [a 2017 paper](#) from Google, transformers are among the newest and one of the most powerful classes of models invented to date.

Transformers are different from other neural networks in that they explicitly search for and learn about proximate and distant influential relationships. Because of this, they can consume unlabeled data. Other machine learning algorithms rely on slow to build and curate labeled data, and often their accuracy is dependent upon the amount of training data available. This makes transformers capable of consuming much larger datasets with less human involvement. As



such, they have become something of a favorite among researchers. In a quick survey of ArXiv, in the last two years, transformers were mentioned in the abstracts of 70% of all the papers cataloged in science and engineering.

New applications and use cases for transformers abound. Disciplines such as computer aided drug discovery, climate modeling, and protein folding all use some form of transformer models. Table 2 contains a list of links to some transformer models by discipline.

*Table 2. Transformer Models by Discipline*

Discipline	Product
Computer aided drug discovery (CADD)	<a href="#">MegaMolBart</a>
Climate Modeling	<a href="#">FourCastNet</a>
Protein Folding	<a href="#">AlphaFold2</a>
	<a href="#">RoseTTAFold</a>
	<a href="#">OpenFold2</a>

## Coherent System and Accelerator Memory

Memory coherence is a desirable condition in which corresponding memory locations for each processing element in a multi-element processor have full memory speed access to all pools of memory.

The challenge is that extremely fast accelerators need equivalently fast access to data to run at full speed, which is why we see high-bandwidth memory (HBM) being put directly on GPUs. Of course, a large enough problem will always be larger than the accelerator memory and force the accelerator to fetch data from main CPU memory. Sometimes, if the computation is heavy enough, this data storing and fetching can be hidden while computation happens in a “just in time” sort of computing pipeline. Sadly, this is not always possible. In many cases, accelerator computation falls idle while waiting for input/output (I/O).

The natural solution is to think about an architecture where an accelerator has full speed access to main memory at 10-20x the footprint of leading accelerator memory sizes. This kind of architecture where an accelerator is a peer of a CPU is called “coherent”.

Several standards have emerged:

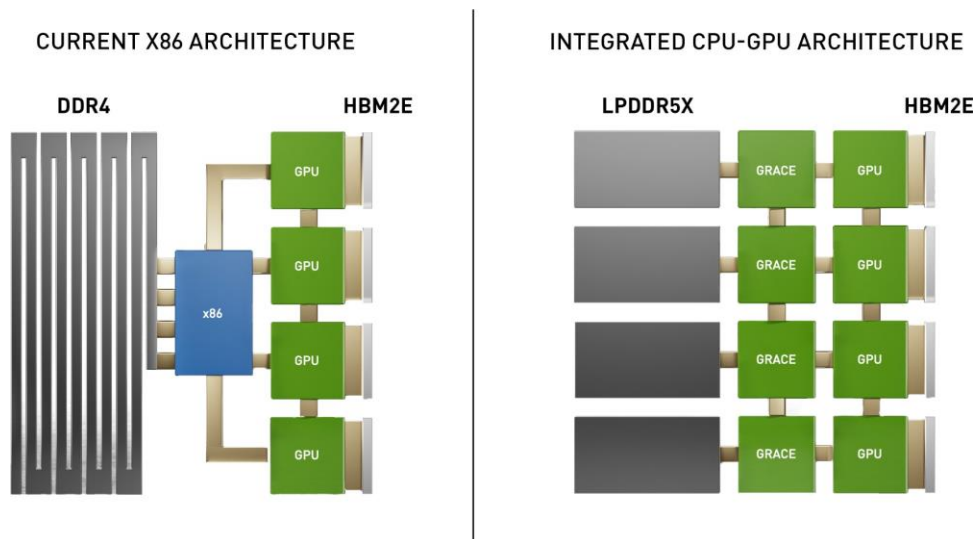
- [openCAPI - Coherent Accelerator Processor Interface](#)
- [Cache Coherence Interconnect for Accelerators \(CCIX\)](#)
- [Heterogeneous System Architecture Foundation](#)

In March of 2021, [NVIDIA announced an Arm based architecture called “Grace”](#), which would be memory coherent along with the [first supercomputer ALPS](#) to be built with the first shipments of the architecture. AMD similarly launched their [memory coherent architecture Trento](#) to be used by both Oak Ridge National Laboratory (ORNL) Frontier and Lumi. You can view more information about the [porting platform NVIDIA is providing](#) to select clients.

These new architectures and systems will be an exciting step forward in performance and capability, which we expect will unlock advances throughout all areas of investigation where HPC systems are memory bandwidth bound. Further, with the infusion of machine learning technologies into HPC applications, and the associated requirements to train ML models before they are useful, we expect technologies like memory coherence to create new opportunities there as well.

Figure 12 shows a comparison of the current x86 paradigm architecture and the coherent Grace architecture from NVIDIA.

*Figure 12. A Comparison of the Current x86 Architecture and NVIDIA’s Coherent Grace Architecture*



## Advances in Computer Networks

### *Network Bandwidth for Scaling*

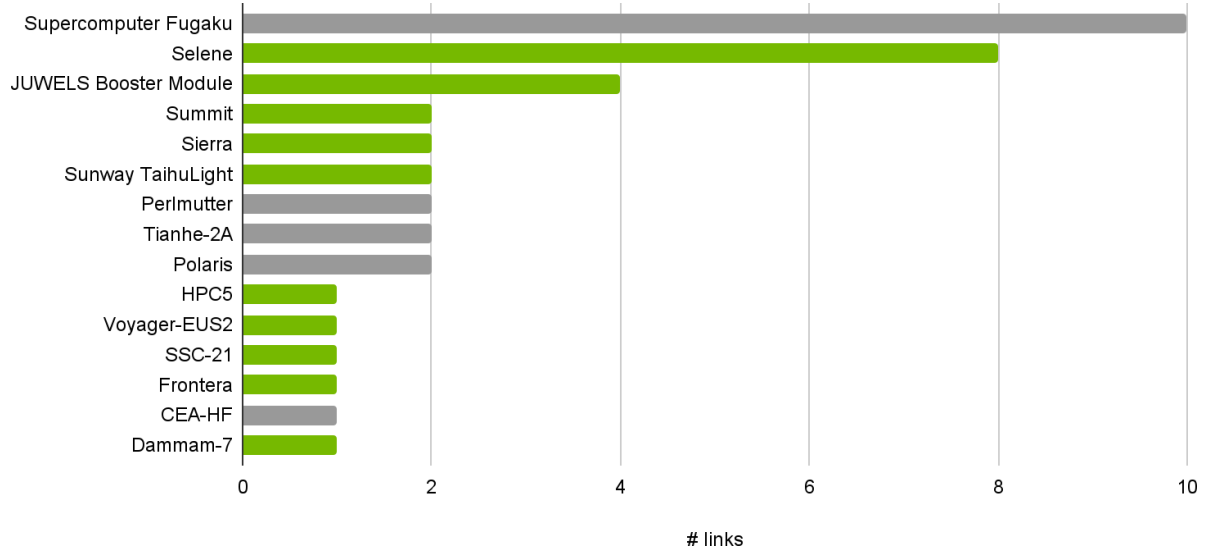
Common practice in HPC system design is to use a single high-performance network, usually used for both message passing and filesystem data flow. Some of the top 10 from the 2021 Top500 use multiple network interface adapters per server node such as the Summit Supercomputer which has 2x Enhanced Data Rate (EDR) 100 GB/s InfiniBand, and the [NVIDIA Selene](#) which has 8x High Dynamic Range (HDR) 200Gb/s InfiniBand. Often these leadership systems have other signaling and management networks used for provisioning, monitoring, and other low bandwidth but useful activities. Such management networks are beyond the scope of this eBook since the focus is on networks carrying data being processed.

Comparison of the per-node bandwidth for the top 15 supercomputers (on the Top500) is difficult because vendors do not have to share the bandwidth per link, nor the number of links per node. Although, some share this information voluntarily.

As data-hungry machine and deep learning training becomes one of the predominant workloads running on HPC systems, emphasis on communication and data movement to scale training performance becomes more important. The paradigm shift toward accelerated computing has carried with it an additional requirement for about 200 GB/s of bandwidth per accelerator. As faster accelerators become available, the bandwidth required to keep them fed with data will grow and the number of parallel connections needed will grow proportionally. That being said, interconnect bandwidths are on a steady growth rate too, currently keeping the number of parallel connections from generation to generation of accelerators in check.

A recent study done by Johns Hopkins University [and Amazon Web Services \(AWS\)](#) showcases this phenomenon as the findings show that distributed training can benefit from high network bandwidth and the scaling factor can improve to nearly 100% if the network is fully utilized.

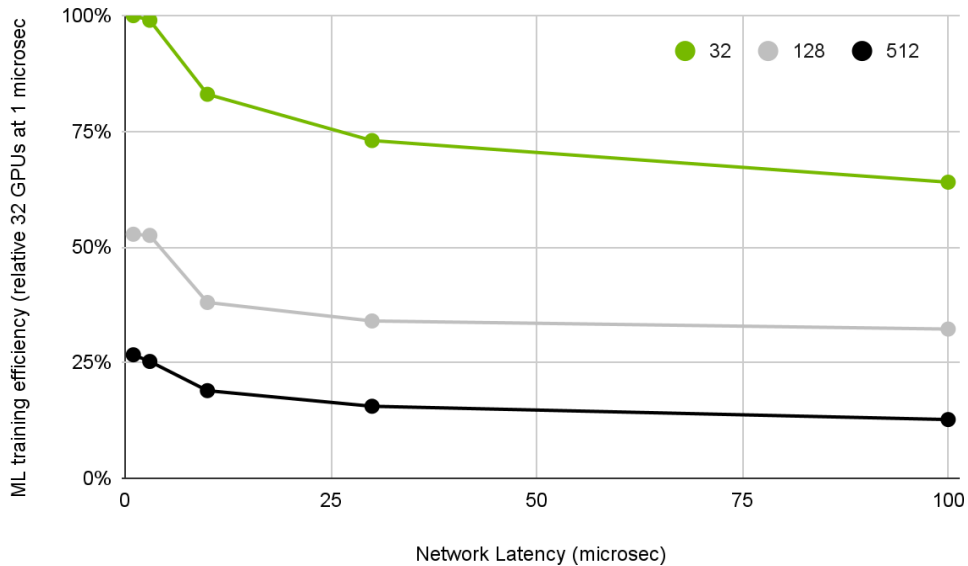
Figure 13 shows some of the top 20 of the Top500 supercomputers sorted by the number of network connections per node. The number of network links per node by supercomputer shown are not in rank order. Green represents NVIDIA InfiniBand networking and gray depicts other network types.

*Figure 13. Bar Graph of Interconnect Links for the Top Fifteen Supercomputers*

Recently, Massachusetts Institute of Technology (MIT) did a [study on the impact of network performance](#), both latency and bandwidth, on ML training performance for transformers, ResNet50, and Megatron. Not surprisingly, they showed that for most models, as the network bandwidth per GPU grew, the training time scaled when the models were trained to the equivalent level of accuracy.

Further, they looked at network latency and showed its impact on ML training scaling efficiency as the network latency grows. The interesting result is that the scaling is impacted by latency almost 2x more severely at larger GPU counts than it did for smaller GPU counts. Figure 14 illustrates ML training scaling efficiency as a function of network latency for different GPU counts.

Figure 14. ML Training Scaling Efficiency as a Function of Network Latency



### Interconnects for Multi-GPU and Multi-Node Acceleration

Large language and deep learning recommender models, and the neural networks they use, are expanding in size and complexity with larger datasets to deliver the next level of insights. As a result, computing and memory capacity requirements are growing exponentially to train these networks in useful timeframes. Multi-GPU implementations have demonstrated almost linear scaling of performance to address this challenge. And a key architectural enabler to scale performance of these multi-GPU systems is resilient, flexible, high-bandwidth inter-GPU communications.

[NVIDIA NVLink](#) provides a solution to connect GPUs together in a point-to-point network and it has elevated the performance of eight GPU servers. NVLink technology addresses cases when models exceed the on-chip GPU memory size and enables the interconnection of GPUs with high speed I/O, 900 GB/s, at 7x the bandwidth of Peripheral Component Interconnect Express (PCIe) Gen5. See a method to improve HPC application performance up to 75% using [NVLink and NVtags](#).

One NVLink (see Figure 15) supports 18 NVLink ports per GPU, each operating at 25 GB/s. A full NVLink point-to-point connection takes all 18 ports to connect two GPUs at 900 GB/s.

*Figure 15. NVLink with 18 Ports*

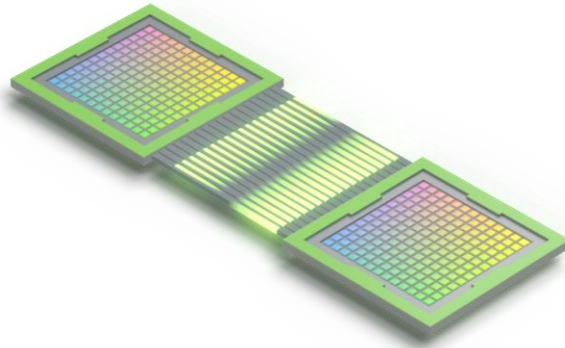
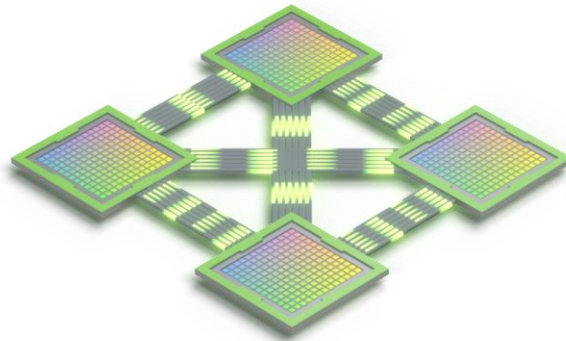


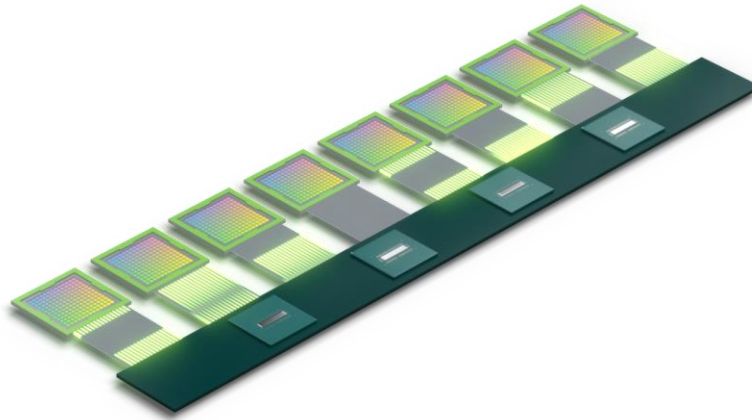
Figure 16 is a representation of a four-way connection utilizing six ports per GPU to each of the other three GPUs providing 300 GB/s between any two GPUs.

*Figure 16. NVLink Configuration For Four Interconnected GPUs*



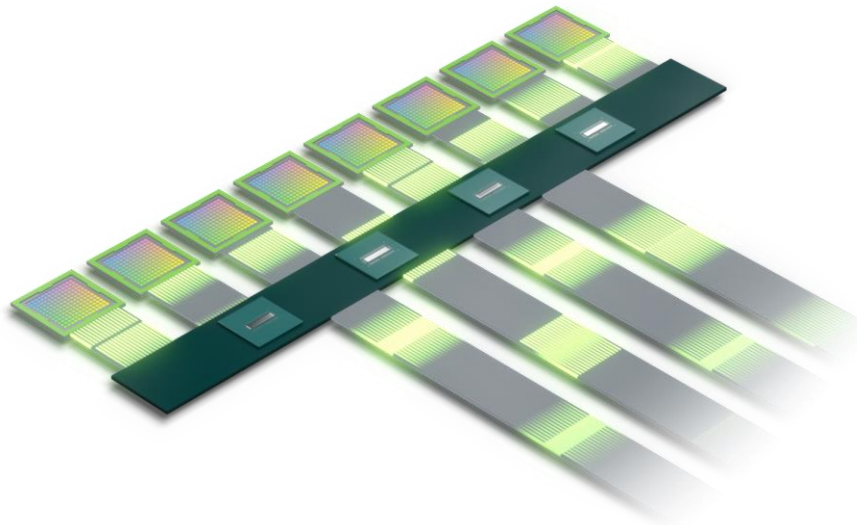
However, with all-to-all communications, where every GPU needs to communicate with each other without any sacrifice in GPU-to-GPU NVLink bandwidth, in order to support 900GB/s, an advanced switching solution such as NVSwitch is needed to bring GPU server performance higher. Each NVSwitch, shown in Figure 17, provides 64 NVLink ports to network every GPU at 900 GB/s.

*Figure 17. Eight GPUs Interconnected Using Four NVSwitch Devices*



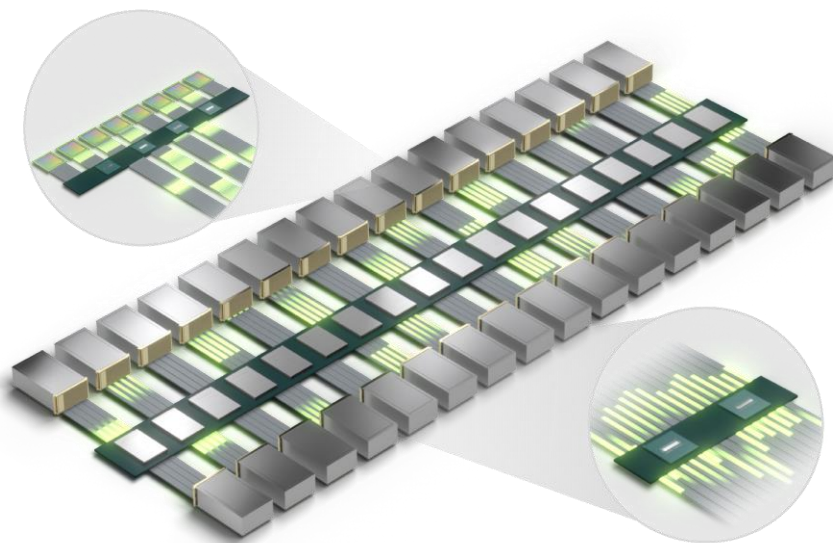
NVSwitch addresses this challenge and provides a fully NVLink-connected system with all GPUs delivering the full 900 GB/s of bi-directional connectivity. Server architectures from eight to 16 GPUs with NVSwitch can address 640 GB to 1,280 GB with fast GPU memory access. Figure 18 shows how fourth generation NVLink technology adds external connectivity to extend high-performance connectivity between nodes.

*Figure 18. Fourth Generation NVLink and NVSwitch Configuration*



With NVSwitch and NVLink, even larger GPU clusters may be created expanding the addressable memory footprint. Figure 19 shows 32 servers across 18 NVLink Switches with 256 GPUs in one NVLink domain boasting 20 TB total GPU memory.

Figure 19. 32-server Configuration with 18 NVLink Switches



The NVLink Switch System adds a second tier of NVSwitch-based connections to create a fat-tree network topology. One layer of NVSwitches is housed within each server node and a second layer lies in external rack switches, NVLink Switches, to connect all the GPUs for all-to-all communications.

With the external NVLink Switch and NVLink Switch System, NVLink interconnections can extend beyond single server physical boundaries to larger [NVIDIA DGX PODs](#). In this setup, all GPUs are fully interconnected in one NVLink domain. Further optimizations for collective operations have been introduced in the third generation NVSwitch by accelerating multicast and performing in-network reductions with [Scalable Hierarchical Aggregation and Reduction Protocol](#) (SHARP).

The result of expanding the GPU I/O interconnect speed and addressable GPU memory is partially responsible for the expected 6X greater performance for 3D FFT and 9X faster training throughput for a “Mixture of Experts” machine learning technique. Note however, the performance gains are attributable both to the shift from A100 to H100 but also take advantage of the greater GPU interconnect speeds going from HDR 200 Gb/s InfiniBand to both NDR 400 Gb/s InfiniBand and NVLink Switch System.

To optimize parallel efficiency with efficient communications and minimize coordination overhead for high-bandwidth networks and with large numbers of GPU nodes, you may want to consult this post, [Scaling Scientific Computing with NVSHMEM](#).



## *Cloud-Native Supercomputing*

Extracting the highest possible performance from supercomputing systems while achieving efficient utilization has traditionally been incompatible with the secured, multi-tenant architecture of modern cloud computing. A [cloud-native supercomputing](#) platform provides the best of both worlds for the first time, [combining peak performance and cluster efficiency with a modern zero-trust model for security isolation and multi-tenancy](#).

With AI and HPC becoming primary compute environments for wide commercial use, supercomputers now need to serve a broad population of users and to host a more diverse software ecosystem, delivering non-stop services dynamically. New supercomputers must be architected to deliver bare-metal performance in a multi-tenancy environment. The design of a supercomputer focuses on its most important mission: maximum performance and lowest overhead.

The goal of the cloud-native supercomputer architecture is to maintain these performance characteristics while meeting cloud services requirements, such as: least-privilege security policies and isolation, data protection, and instant, on-demand AI, and HPC services. The development of the cloud-native supercomputer architecture is based on open community development, including commercial companies, academic organizations, and government agencies.

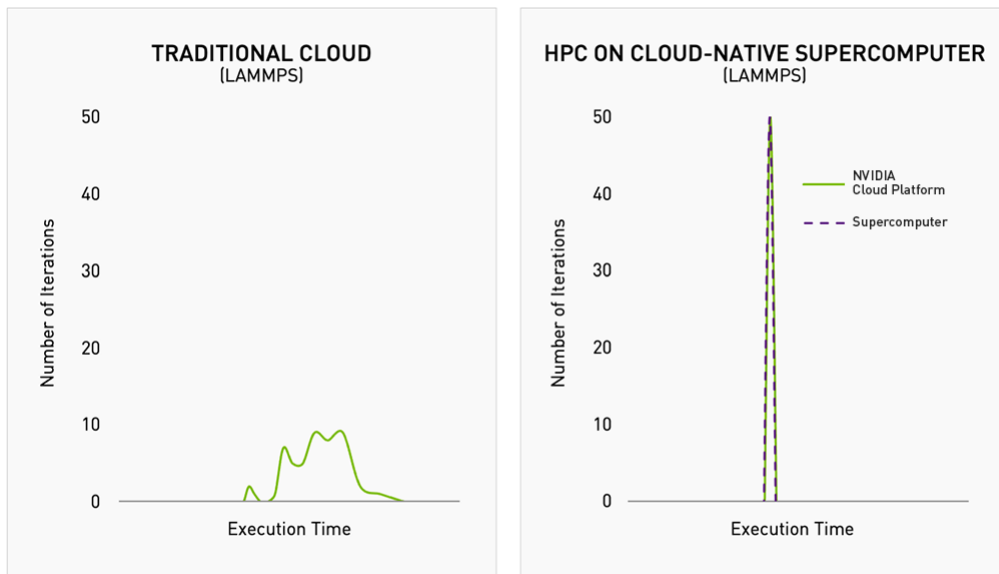
Multi-tenant supercomputing involves many user applications running on shared infrastructure, potentially overlapping use of physical servers, storage, networks, and the I/O traffic patterns that these applications generate. To share the computational resources of supercomputer data centers, NVIDIA introduced the Cloud-Native Supercomputing architecture, which combines bare metal performance, multi-tenancy, and performance isolation for supercomputing. This architecture is part of the NVIDIA Quantum-2 InfiniBand platform, which includes the NVIDIA Quantum-2 switch family, BlueField-3 DPUs, and ConnectX-7 network adapters.

NVIDIA Quantum-2 InfiniBand manages network congestion when it is detected and exercises controls to reduce it at the source. But, with multi-tenancy, user applications may be unaware of indiscriminate interference with neighboring application traffic, therefore isolation is required to deliver expected performance levels. With the Quantum-2 InfiniBand switching system, proactive monitoring and congestion management delivers the needed traffic isolation. This nearly eliminates performance jitter, and ensures the expected predictive performance as if the application is being run on a dedicated system.

Multiple runs of LAMMPS in a traditional cloud environment experiences I/O delays resulting in varied execution times. Congestion control and isolation deliver deterministic, consistent

execution times over a narrow range as when running the application in isolation on a dedicated supercomputer system.

*Figure 20. Execution Times for Traditional Cloud Compared to HPC on Cloud-native Supercomputer*



Scientific computing and AI applications make extensive use of MPI collective operations like `All-Reduce` and `All-to-All`. SHARP for In-Network Computing was first introduced with NVIDIA Quantum InfiniBand switches to offload these collective operations from the host CPU to the switch network. This eliminates the need to send data multiple times between endpoints and decreases the amount of data traversing the network as aggregation nodes are reached. It dramatically reduces the MPI operations time.

The NVIDIA Quantum-2 InfiniBand switch provides 32X the AI engines over the prior Quantum switch to support simultaneous use of SHARP for multi-tenant scalability of large data aggregation through the cloud data center network. With support for unlimited small message reductions and multiple large message reductions flows per switch, multiple tenants running applications across a shared system can leverage doubling performance of complex MPI collective operations, such as `All-Reduce`, and consistent, lower latency regardless of cluster size when using SHARP.

[Watch the In-Network Computing with NVIDIA SHARP Video.](#)

Another core element that enables cloud-native supercomputing is the data processing unit (DPU) - the NVIDIA BlueField. As a fully integrated data center-on-a-chip platform, BlueField offloads and manages data center infrastructure instead of the host processor, enabling security and orchestration of the supercomputer.

### *SmartNIC Offload*

Technologies running in cloud service providers (CSPs) are making their way toward the data centers of the future. They have the potential to maximize server investment by offloading low value tasks, such as hypervisors and I/O, while increasing security by maintaining a hardware level barrier between execution instances. Performance, security, and storage could all be significantly enhanced by employing smart interface network cards (smartNICs). Although the largest hurdle for this deployment is that commercial SmartNIC products are mostly notional today.

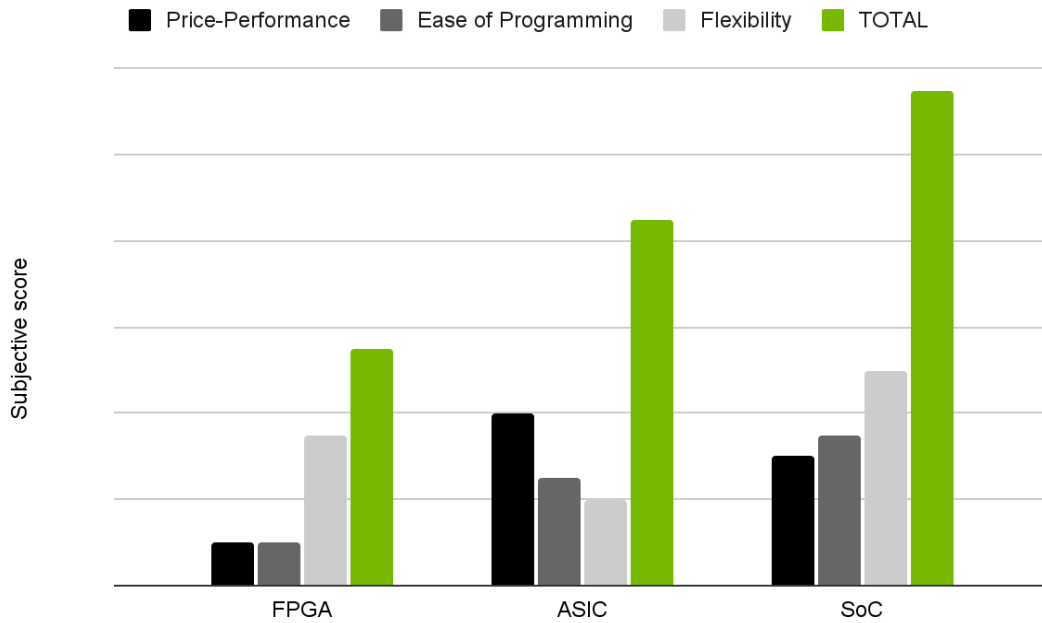
That being said, AWS and their [NITRO](#) technology has pioneered widespread data center usage of service offload, starting with their proprietary hypervisor and growing from there. Since then, industry has taken notice of the success of the concept and several hardware vendors have announced plans to offer server-side offload processors at the network interface edge. This means data passing into and out of the server can be shunted through some kind of processor such as ASIC, FPGA, or SoC. Well known vendors such as Broadcom, Xilinx, Intel, and NVIDIA, along with newcomers Pensando and Fungible, have all announced plans and products in this area.

What does all of this have to do with HPC? That is a good question.

As of this time, an application used in HPC that would run faster on a system equipped with some kind of SmartNIC than it would otherwise is still theoretical. However, there is no disputing that security or I/O enhancements will become use cases.

Taking a step back, Figure 21 shows an evaluation of SmartNIC processor types from three points of view: price-performance, programmability, and flexibility. This is a qualitative and subjective evaluation, but there are reasons we have taken this point of view. See this post about [choosing the best SmartNic](#) for details.

Figure 21. Smart NIC Technology Comparison



## Converged Accelerator and Network Interface Card

What is a converged accelerator and NIC?

The GPU accelerates AI, ML, and HPC applications, but the I/O into and out of the GPU can become the leading bottleneck to overall application performance. Combining the advanced capabilities of a smart network interface card (NIC) with GPU acceleration in a single converged card utilizing the latest PCIe interconnect, creates a balanced architecture by design - a converged accelerator NIC.

Note that PCIe is found in x86 standard and HPC servers because of the wide support in third party add-in options for I/O and other functions. In systems where multiple GPUs are desired, a converged accelerator card enforces the optimal 1:1 ratio of GPU-to-NIC and avoids contention on the server's own x86 PCIe bus, so the performance scales linearly with additional devices. This eliminates data bottlenecks for data going through the CPU host and ensures low and predictable latency for GPU-to-GPU communications and GPU-to-remote NVMe storage using

GPUDirect RDMA and GPUDirect Storage technology (components of the NVIDIA Magnum I/O data center I/O acceleration software).

This converged design delivers optimal performance for GPU-powered I/O intensive workloads such as AI training. A prime example of this architecture is the NVIDIA H100 GPU and the ConnectX-7 SmartNIC with integrated PCIe Gen5 switch. Some of the benefits of this approach are the ability to leverage mainstream PCIe Gen4 or even Gen3 servers to achieve performance levels only possible with purpose-built high-end host systems and using a single card saves on power, space, and PCIe device slots. Magnum I/O software acceleration libraries such as Unified Communication - X Framework, UCX, and NVIDIA Collective Communications Library, NCCL, automatically use the best performing datapath for data transfer to GPUs so existing multi-node accelerated applications can benefit from high performance and scalability without any modification.

## HPC Developer Ecosystem

### *Vibrant Community*

When evaluating HPC technologies, one of the best indicators of long-term success is an active user community doing similar work and an engaged vendor. Common technologies like the x86 architecture have huge communities of users, countless tools, and therefore require less special attention from a prospective vendor offering a solution based on that architecture. Conversely, custom technologies with small communities require significant client-specific investment to achieve similar success.

The [NVIDIA HPC community](#) is vibrant, active, and growing. The success and productivity of developers, researchers, scientists, and engineers leveraging NVIDIA technologies—hardware, software, systems, and platforms—across a diverse set of application domains and industries is key to creating scientific breakthroughs.

This success is founded on understanding the HPC community's unique requirements and then meeting those needs through a broad range of developer support services with varying levels of technical engagement, depth, and business models. Table 3 contains a list of NVIDIA's broad range of support resources for community engagement and learning.

*Table 3. NVIDIA Supported Resources for Community Engagement and Learning*

Complimentary Services and Tools	
NVIDIA Developer Forum	Developer Forums
	Bug Reporting and Tracking
	Pre-trained AI Models
	Enhancement Requests Compiler and Libraries
On Demand Education Content	NVIDIA GTC, YouTube
NVIDIA Documentation Center	docs.nvidia.com
GPU Hackathons and Bootcamps	gpuhackathons.org
Open Source Content	github.com/NVIDIA
Code Samples	github.com/nvidia/cuda-samples
Professional Services	
Training Courses	Deep Learning Institute
Professional Support	Container Support
	Compliers
Consulting	AI Service Delivery Partners

NVIDIA provides a wealth of resources to the NVIDIA community, which include training opportunities, technical documentation, software development kits (SDKs), code samples, networking opportunities, peer and expert support, and much more.

### *Standard, Open, and Portable Parallel Programming Models*

There are a plethora of software development tools and programming models available in the HPC industry, with varying support across target hardware platforms. Options are available from

independent software vendors (ISVs), HW/SW platform providers, open source, and more. While getting their applications functioning properly and performing satisfactorily, developers also have to concern themselves with pros and cons of closed versus open software stacks, as well as the challenges of productivity and portability.

HPC environments are now offering innovative but often complex architectures, mixing CPUs, with accelerators, and now, even DPUs. These systems require sophisticated software solutions to provide support for aligning development to take advantage of the best-in-class features of the different hardware elements. Users need to be able to mix these system elements easily, and efficiently program their solution whether it is a homogeneous CPU-only system, a heterogeneous CPU/GPU, or CPU/GPU/DPU configuration.

A coder needs to balance optimizing performance and streamlining development, while also avoiding lock-in. Juggling à la carte software solutions from various vendors to address different components of a system is possible, but likely not very productive. Software engineers should spend their time adding value to their systems, not stringing together software stack components. A user may want to vary device components over time, so they cannot get restricted by a vendor-only tool suite.

It is likely that one single programming model will not address all the requirements of the modern coder. Ideally users should employ a development environment that supports open standards, and the hybrid integration of multiple appropriate models to achieve all goals. When it comes to identifying the optimal accelerated computing programming model for your HPC application, [NVIDIA offers a comprehensive software development kit](#) (SDK) that provides users the freedom to choose the right tool for the job.

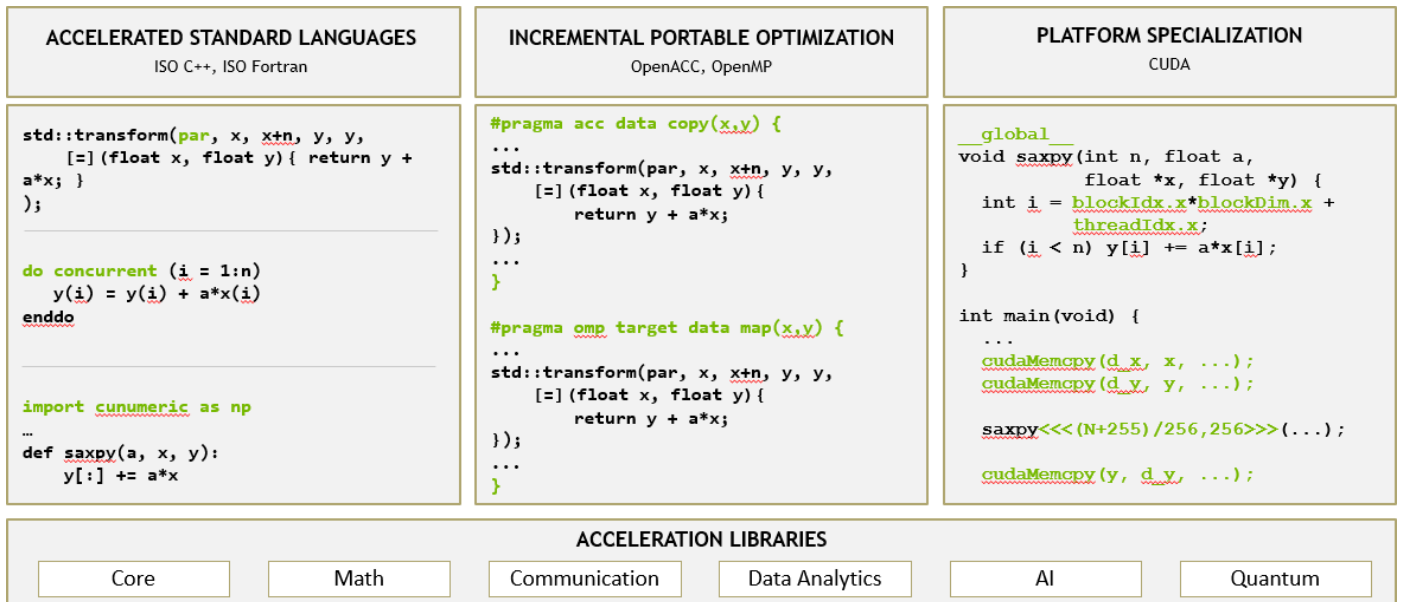
This software stack supports programming of the entire NVIDIA platform, including the CPU, the GPU, and the network. The goal is to enable users with flexible options across the development spectrum to get the absolute best out of the application, whether they are willing to make platform specialized decisions in their development, leverage standard languages, or some type of hybrid coding method in between.

Figure 23 illustrates the four major programming model types available from the NVIDIA HPC SDK:

- Accelerated standard languages,
- Incremental portable optimization,
- Platform specialization, and
- Accelerated libraries

The range of programming options covers platform specialization and directives-based options, to standard approaches and performance optimized libraries.

Figure 23. Major Programming Model Types Available from the NVIDIA HPC SDK



[NVIDIA brings parallelism and acceleration](#) to the developer who is not willing or able to make the investment in platform specialization, and the SDK makes that process seamless, with very high developer velocity and productivity.

Many HPC developers are domain scientists, and their number one challenge is expediting their time to gain scientific insights faster. Getting to a result in computational science has two components, one is your runtime performance, the other is developer velocity. The goal is to make that process seamless and automatic as possible and suggest two recommended strategies to achieve this.

Runtime performance can be achieved through NVIDIA performance optimized Libraries: Core, Math, Communication for the network, data analytics, AI and Deep Learning, and even libraries for Python and quantum computing circuit simulation. All libraries are specifically tailored for these use cases.

A huge advantage in using libraries is that you can fix the API and architecture, and expert NVIDIA staff then optimize libraries for drop-in ease-of-use, platform specialized performance, and support that transcends HW generations. Performance optimized libraries deliver a great balance of user productivity and application performance.

Developer velocity can be achieved by using pre-validated libraries. The programmer quickly and directly enhances the HPC application with the performance benefits of GPU acceleration. Libraries enable the developer to leverage the hardware capabilities without having to be an expert in all the implementation details. For example, the optimized libraries make it easy to take advantage of powerful GPU features like Tensor Cores, enhanced L2 cache, or shared memory.



However, there are myriad use cases just within scientific HPC. When specialized libraries are not available, the next thing to consider is using standard, specialized, and directives-based programming approaches.

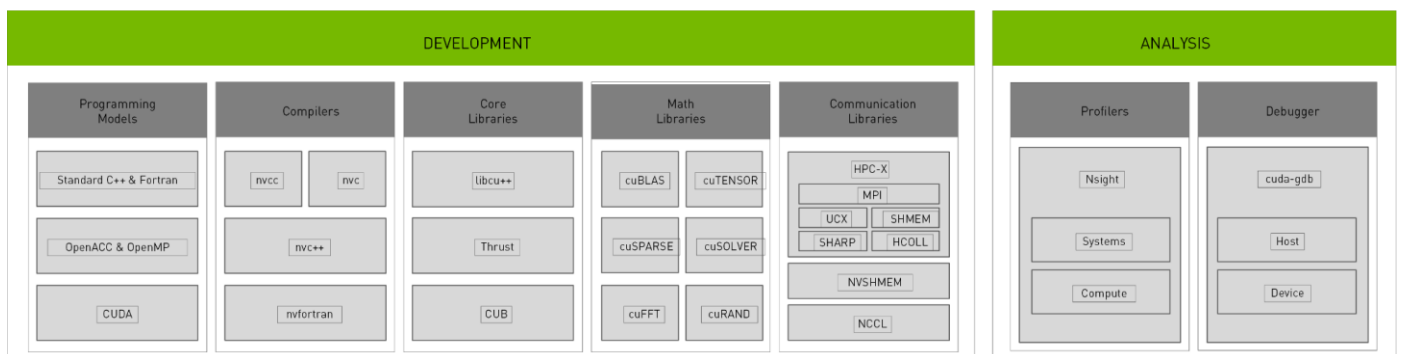
NVIDIA delivers ISO standard, C/C++, and Fortran support, even non-ISO Python as well, supplying automatic GPU-acceleration. These can automatically parallelize execution across multi-core CPUs. The beauty of new parallel construct support in standard languages is that your code can now be more compact and elegant, as well as easy to read and maintain.

Following the ISO model also provides the developer with a robust state of portability. Unlike current competitive software stacks, today's existing NVIDIA compilers enable the running of standard language application codes on CPUs, GPUs, or heterogeneous systems of both. Learn more with deeper dive reads into our post on the advantages of using Standard Languages, as well as our specific posts on [C++](#), and [Fortran](#) programming support.

We recognize that innovative advances in hardware features and capabilities can precede support in standardized language approaches, and in these cases, the developer has the option to do incremental portable optimization through directives-based programming methods, leveraging methods like OpenACC or OpenMP.

For those leading-edge users wanting to take immediate and full advantage of the new advances in hardware features and their performance enhancements, developers can always use the power of specialized [CUDA programming](#) for NVIDIA GPUs and CPU targets. As illustrated in Figure 24, the NVIDIA HPC-SDK boasts several components that are divided according to the two stages: Development and Analysis.

Figure 24. NVIDIA HPC-SDK Support For Development and Analysis



This HPC developer environment is a single, comprehensive collection of the compilers and tools to program the NVIDIA platform, including CPU, GPU, and network. It is available in a variety of ways depending on your preferences. This includes, as a traditional installer from the NVIDIA [Developer Zone web portal](#), as a container off of our NGC site, available through Spack, and accessible off of all the major CSP marketplaces, including AWS, Azure, and others.

This SDK supports all the major programming models previously described: standard C/C++ or Fortran, Python, directive-based programming like OpenACC and OpenMP, and the specialized approach for ultimate performance optimization through CUDA. That support is provided through a suite of compilers in the SDK:

- nvcc: NVIDIA CUDA compiler
- nvc/nvc++: NVIDIA C/C++ compiler
- nvfortran: NVIDIA Fortran compiler

These compilers not only provide automatic GPU acceleration, but they also deliver automatic parallelism for multi-core CPUs, including x86, OpenPOWER, and Arm server- with parallel vectorized high-performance code support- architectures.

The HPC SDK supplies a wide and varied range of [GPU-accelerated libraries](#), including:

- Core libraries: [Thrust](#), [CUB](#), and [libc++](#)
- Math libraries: [cuBLAS](#), [cuTENSOR](#), [cuSPARSE](#), [cuSOLVER](#), [cuFFT](#), and [cuRAND](#)
- Communication libraries: Magnum IO (HPC-X, [NVSHMEM](#), and [NCCL](#))

And of course, the HPC software development kit includes much needed analysis tools for debugging and profiling support, including:

- Debuggers: [CUDA-GDB](#)
- Profilers: [Nsight Systems](#) and [Nsight Compute](#)

These analysis tools help you fix and validate your code functionality, and then provide insights into bottlenecks in your application execution, enabling performance optimization and tuning.

As mentioned earlier, the [NVIDIA HPC SDK is free to download](#), and updates numerous times per year.

For an on-demand video describing the HPC SDK, including a variety of benchmark speed-up examples, please join our developer program and visit the [NVIDIA On-Demand](#) portal, [A Deep Dive into the Latest HPC Software](#).

The presenter steps through a number of different options for programming models such as C/C++, Fortran, directives, and others. They then demonstrate the productivity of leveraging standard language constructs, and the ultimate performance advantages of utilizing hybrid GPU systems over CPU-only platforms: retargeting architectures without having to make a single code change.

The video includes a Lulesh C/C++ code example accelerated 13x, and an even more dramatic MAIA code running 7x faster through programming methods - and almost 60x quicker if you factor in a new NVIDIA A100 GPU in the platform. By comparison, Fortran (Do Concurrent programming loop constructs) speed-up examples include computational chemistry codes NWChem and GAMESS.

## Cloud, VMs, and Containers

Over the timespan of decades, HPC tends to go through cycles of technology diversification and consolidation. Consolidation happens when a new disruptive technology enters the market and becomes a new standard. Often this new disruptive technology is available from only one or a small number of vendors.

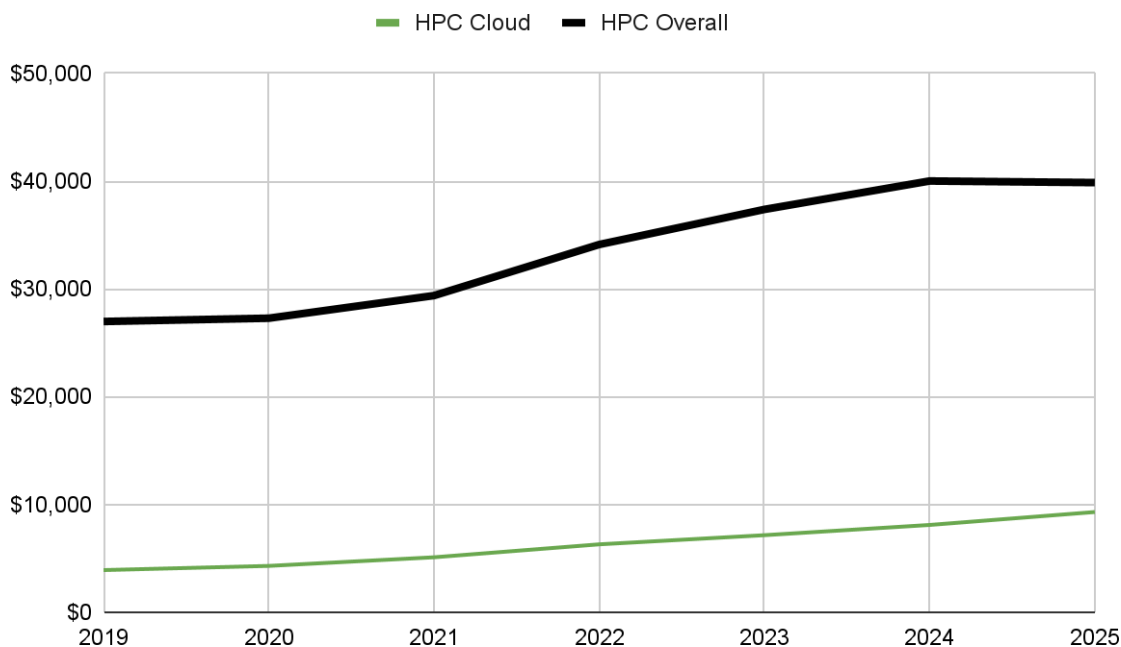
Diversification happens when a new, high potential paradigm comes into play, but a technical standard has not emerged. Many vendors jump into the new paradigm offering solutions leveraging it, and then a process of attrition occurs, as the crucible of HPC business burns away all but the most efficient and valuable offerings.

In the early days, the cloud was not taken seriously for most cases of HPC. Networks were based on Ethernet, storage was based on Common Internet File Systems or Network File Systems, and cores were being divided into hyperthreads. The tone of skepticism was set largely by a [paper from NASA](#), which concluded that cloud computing technology was simply insufficiently designed and far too expensive per GFLOP to consider for HPC workloads.

Ten years and several hardware and software generations later, the facilities and capabilities of several cloud servers are undoubtedly up to the task of high-performance throughput computing (HPTC) and some are shooting at fine grained, highly sensitive, and parallel workloads. An example of this is the 2021 November Top500 list where a virtual supercomputer on [Microsoft Azure reached the top 10](#) fastest machines in the world. To the dismay of some of the HPC community, this supercomputer consisted solely of Hyper-V based virtual machines!

Further confirmation that cloud-based supercomputing is being taken seriously was provided by [Hyperion in their Supercomputing 2021 briefing](#). This briefing predicts that 25% of HPC will run in the cloud by 2025.

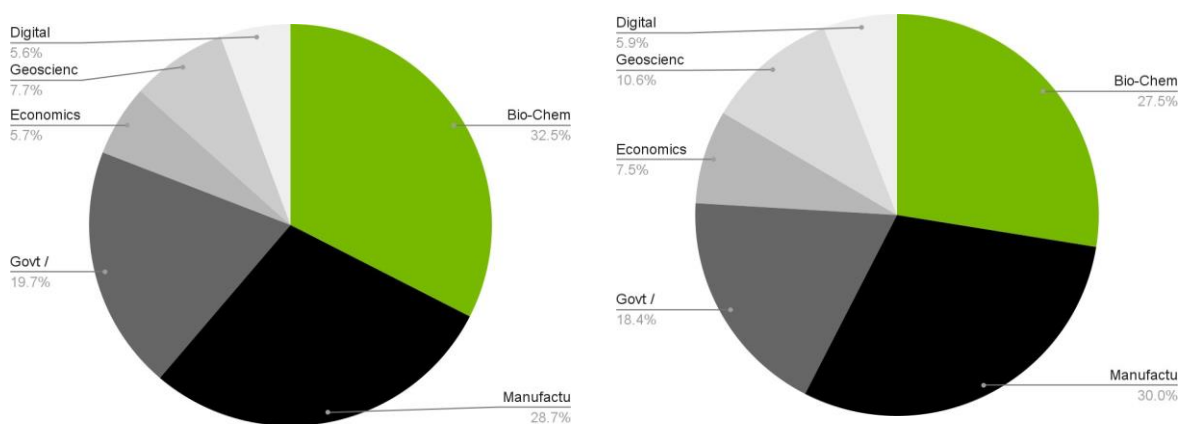
Figure 25. Hyperion: HPC Cloud and HPC Overall Growth



Hyperion predicts steady growth in cloud usage for HPC to reach 25% of the total by 2025.

For the skeptics, the next reasonable question would be: Who is using HPC in the cloud now?

Figure 26. Hyperion HPC by Vertical Market



Once again, Hyperion characterized that for us as illustrated by the pie chart in Figure 26. Currently, two thirds of HPC cloud usage is being used by manufacturing, for efforts such as aerospace, automotive, [pharmaceutical discovery and research](#), computational chemistry, materials science, and more. This commercial entity's leadership in taking HPC to the cloud is interesting, primarily because one of the long-standing objections to using cloud computing was

security. Corporations were uncomfortable using their intellectual property generating activities done on HPC outside their physical premises. The statistics (Figure 26) show that the business case for doing HPC in the cloud has turned that tide.

A cliché among proponents of HPC in the cloud is “Cloud HPC is for people with more money than time.” This statement concedes that running in the cloud is more expensive than using local HPC resources, if they exist and are available. Further, it makes a loose reference to a concept, sometimes referred to as “bursting,” where a business case often makes good sense to use cloud for HPC. In cases where deadlines loom, or additional short-term capacity is desperately needed, the near-instant availability of cloud computing resources can be the best answer for a business.

Another topic which tends to polarize HPC audiences is containerization. Similar to virtual machines, containers are the de facto standard for AI and ML, along with several enterprise and back-office workloads. However, because of severe security concerns around the original container technology - Docker, and the concern about putting a software layer between an application and the hardware, the supercomputing centers of the world split on the use of containers.

Containers offer some significant advantages and arguably, some disadvantages, compared to running on bare metal from user and management perspectives. Table 4 lists the advantages and disadvantages of using containers in HPC applications.

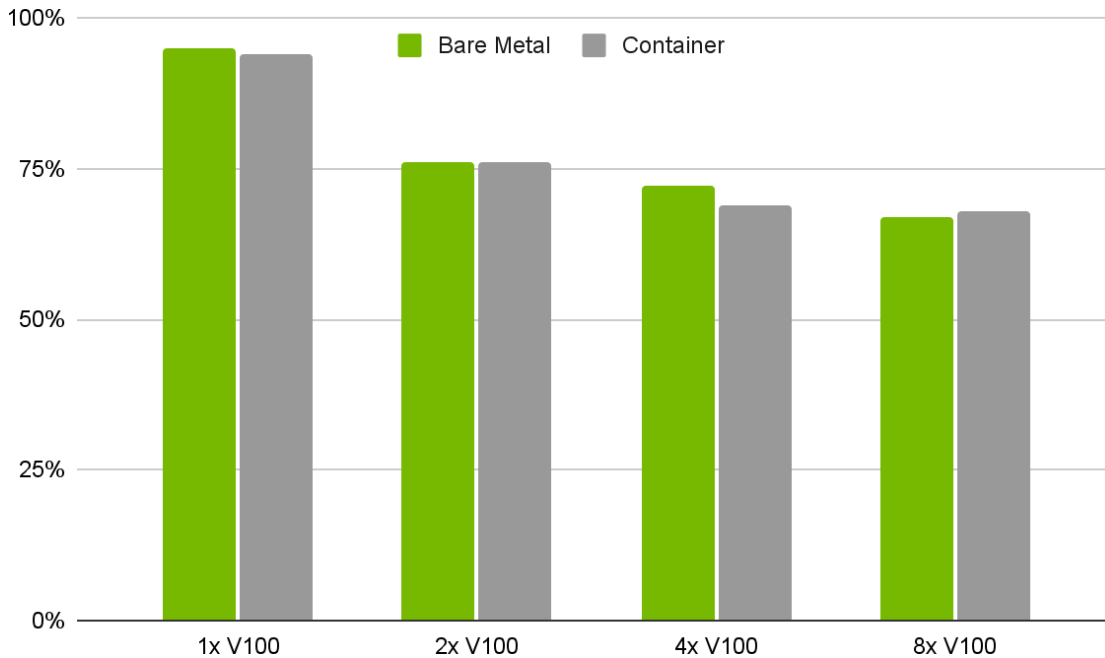
*Table 4. Advantages and Disadvantages of Containers in HPC Applications*

Advantages	Disadvantages
Task isolation	<a href="#">Challenging to build for scientists</a>
App version flexibility	Challenging to customize for scientists
System migration speed	Large footprint (size)
Result reproducibility	security

Thanks to [Singularity](#), the security and multi-node MPI issues with Docker were solved early in the transition to containers, but performance concerns persist. Though NVIDIA technology works on bare metal, virtual machines, and from containers - we are [proponents of container technology](#). Also, from sections above, we believe HPC and AI/ML are coming together and need to interoperate. Since containers are the standard for AI/ML, facilitating that merger will be sped by HPC support of containers.

Figure 27 shows a chart of a common HPC application running on the same server. In one case, running on bare metal, and the other running in containers. Note the lack of performance difference between the two.

Figure 27. Bare Metal Compared to Containers for the MILC HPC Application



## Conclusion

We hope that you found the contents of this eBook helpful to understanding considerations for HPC system design and learning about the current and upcoming technology trends, which will shape HPC systems in the near future. Further, we hope the additional discussion of application support and optimizing HPC application performance, grounds the takeaways of this missive in the understanding that HPC is a means, not an end.

Each of the areas mentioned were only the top points to what should be considered an ever evolving and growing science. From quantum computing to deploying AI with supercomputers, the need for HPC is insatiable and the process to enhance the performance achieved by individual systems is dynamic. As our HPC research and SDKs expand, check the NVIDIA [HPC homepage](#) to periodically review updates that may have been added.