NVIDIA Workshop

Brad Palmer, Senior Solutions Architect

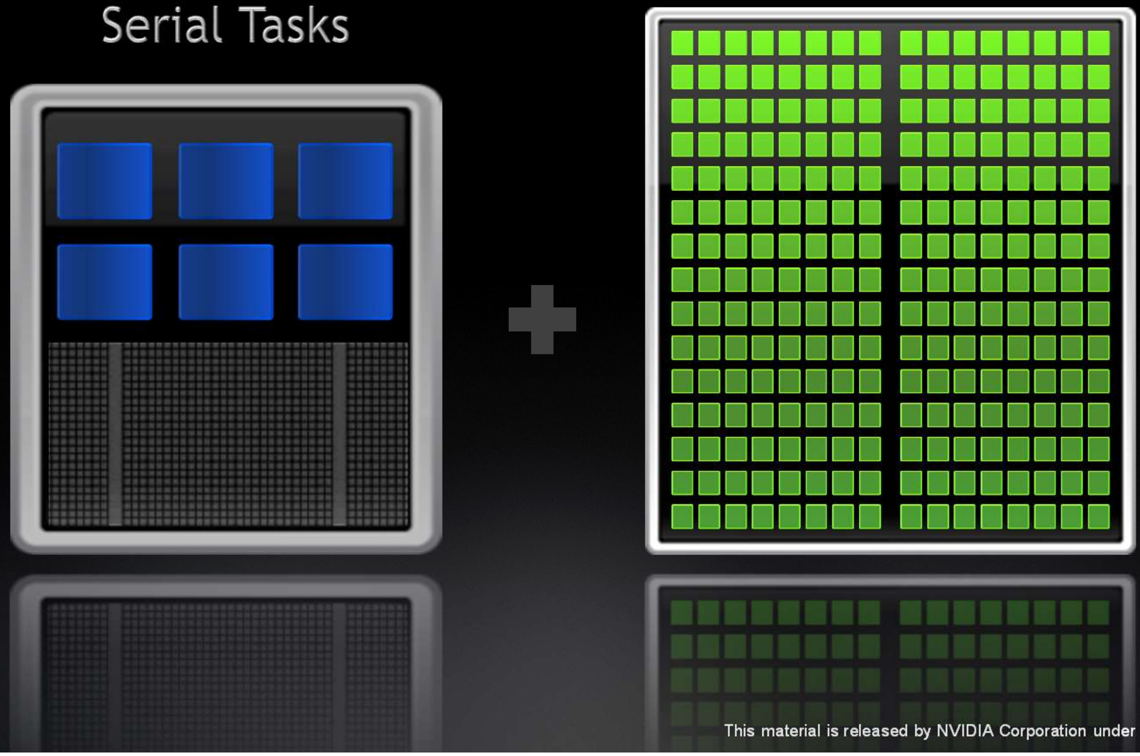
- 5 Ways to Accelerate with GPUs
- Important GPU Features and System Architectures
- Data Center GPUs Overview
- Best Practices for Best Performance
- GPUs in the Public Cloud

First, some GPU basics

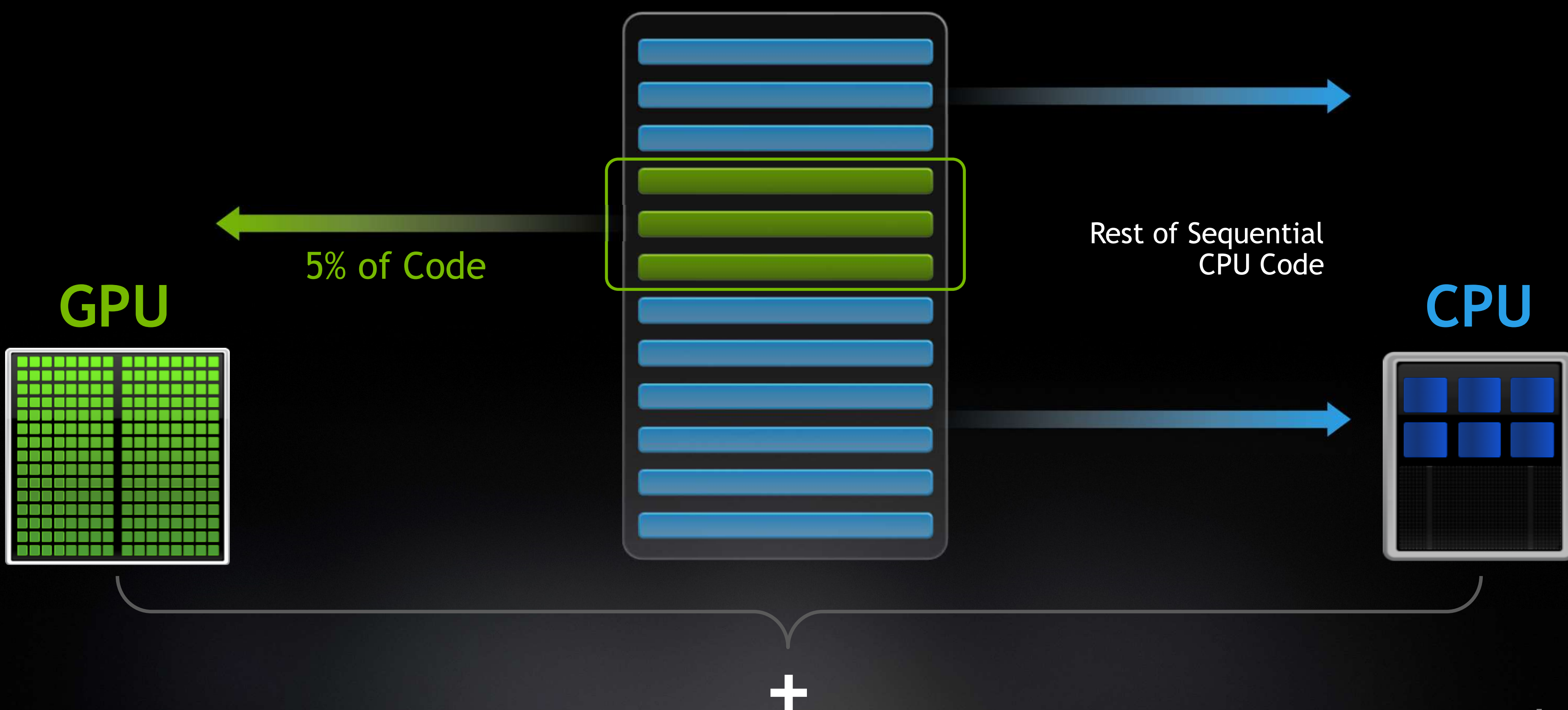
ACCELERATED COMPUTING

CPU
Optimized for Serial Tasks

GPU Accelerator
Optimized for Parallel Tasks

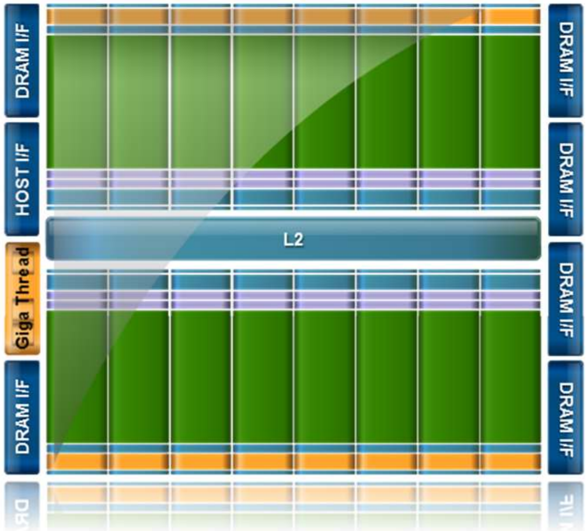


HOW GPU ACCELERATION WORKS



GPU ARCHITECTURE

Two Main Components



Global memory

Analogous to RAM in a CPU server

Accessible by both GPU and CPU

*H100 has **80 GB***

Streaming Multiprocessors (SM)

Perform the actual computation

Each SM has its own: Control units, registers, execution pipelines, caches

*H100 has **114 SMs***

GPU ARCHITECTURE

Streaming Multiprocessor (SM)

Many CUDA Cores per SM

Architecture dependent

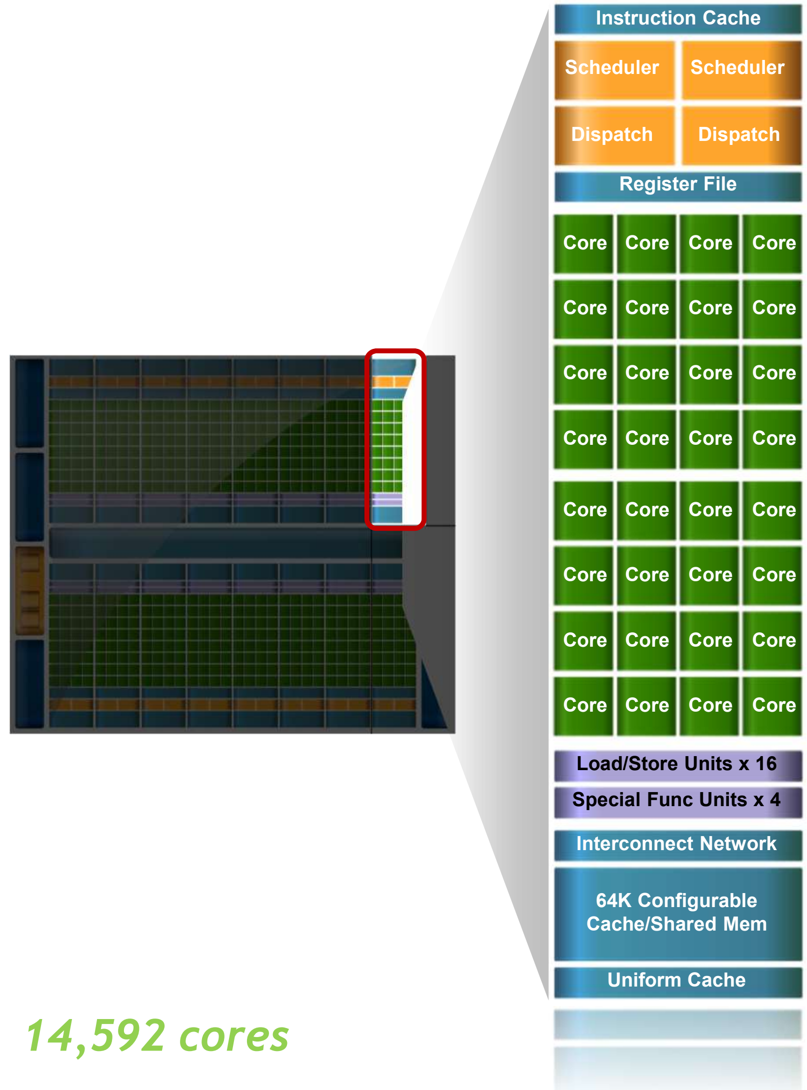
H100 SM has 128 cores

Special-function units

cos/sin/tan, etc.

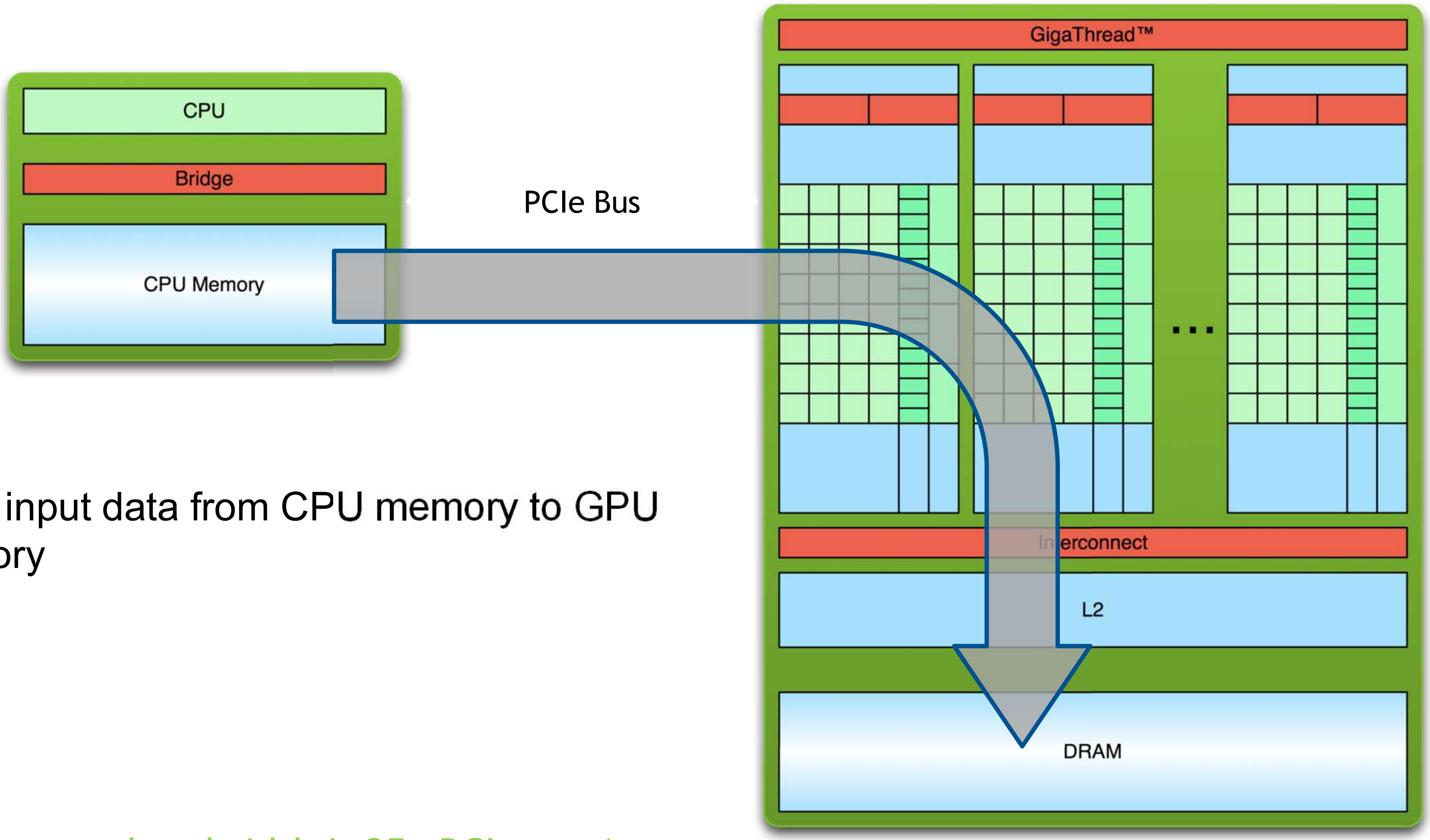
Shared mem + L1 cache

Thousands of 32-bit registers



H100 PCIe has a total of 14,592 cores

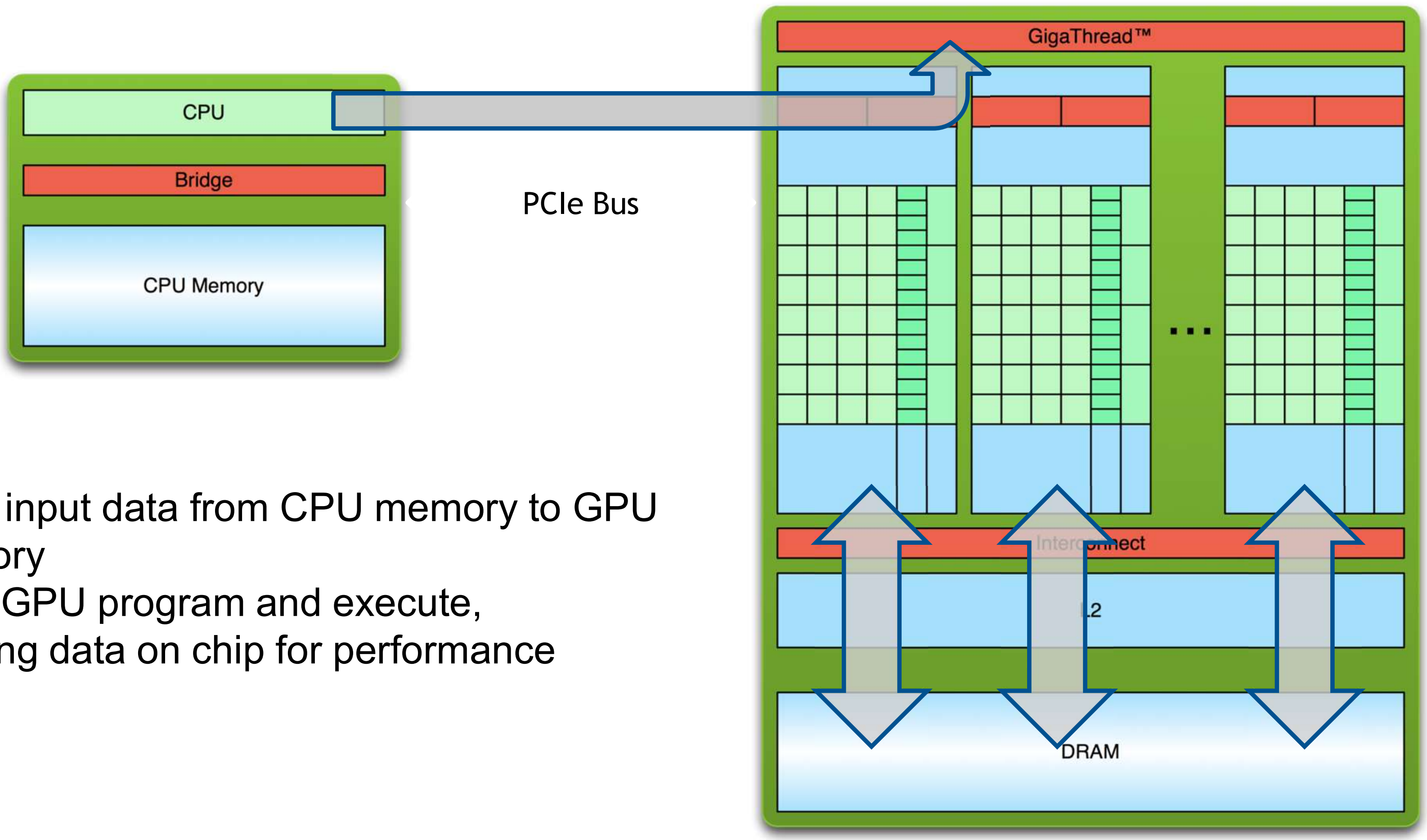
PROCESSING FLOW



1. Copy input data from CPU memory to GPU memory

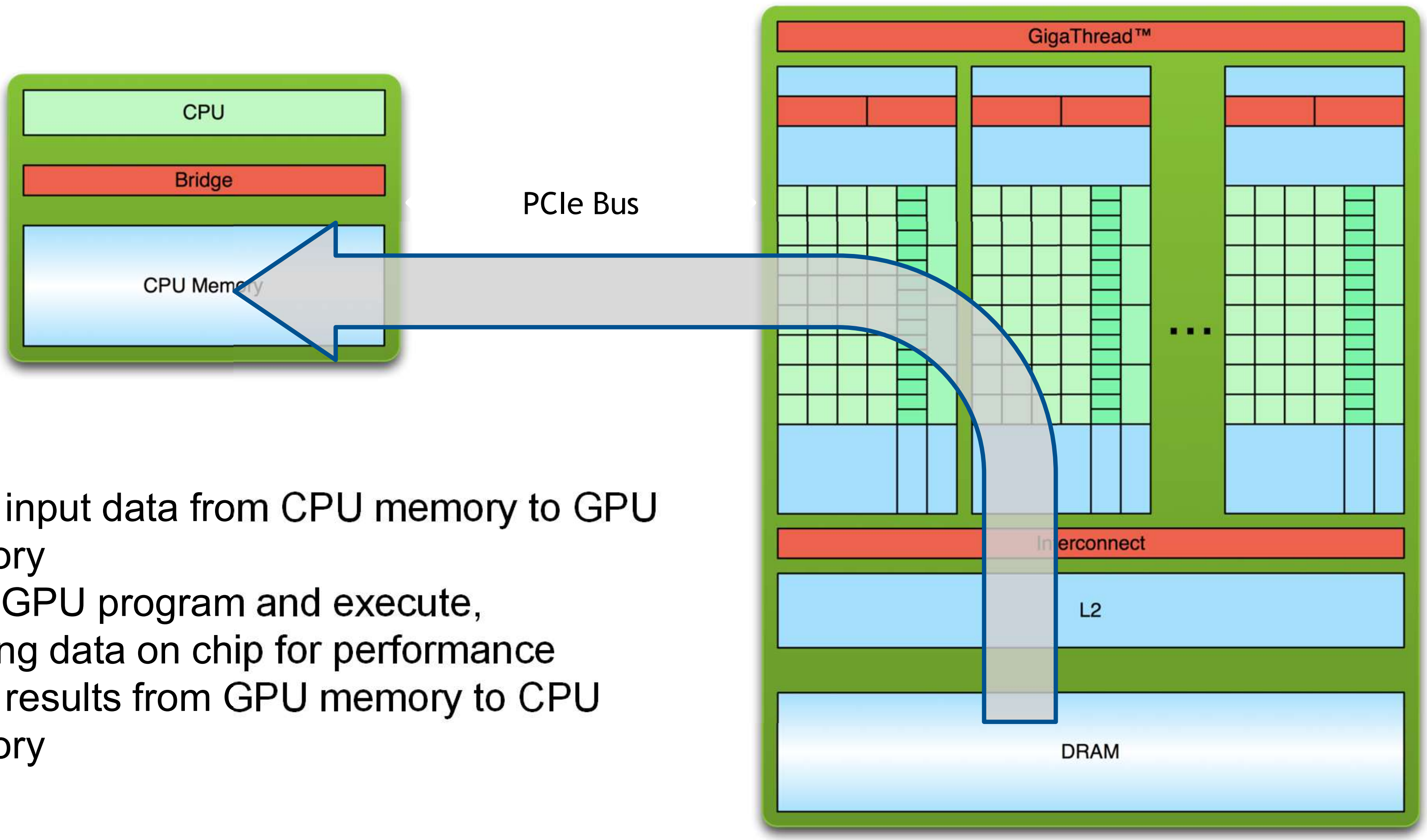
A100 memory bandwidth is 25x PCIe gen4

PROCESSING FLOW



1. Copy input data from CPU memory to GPU memory
2. Load GPU program and execute, caching data on chip for performance

PROCESSING FLOW

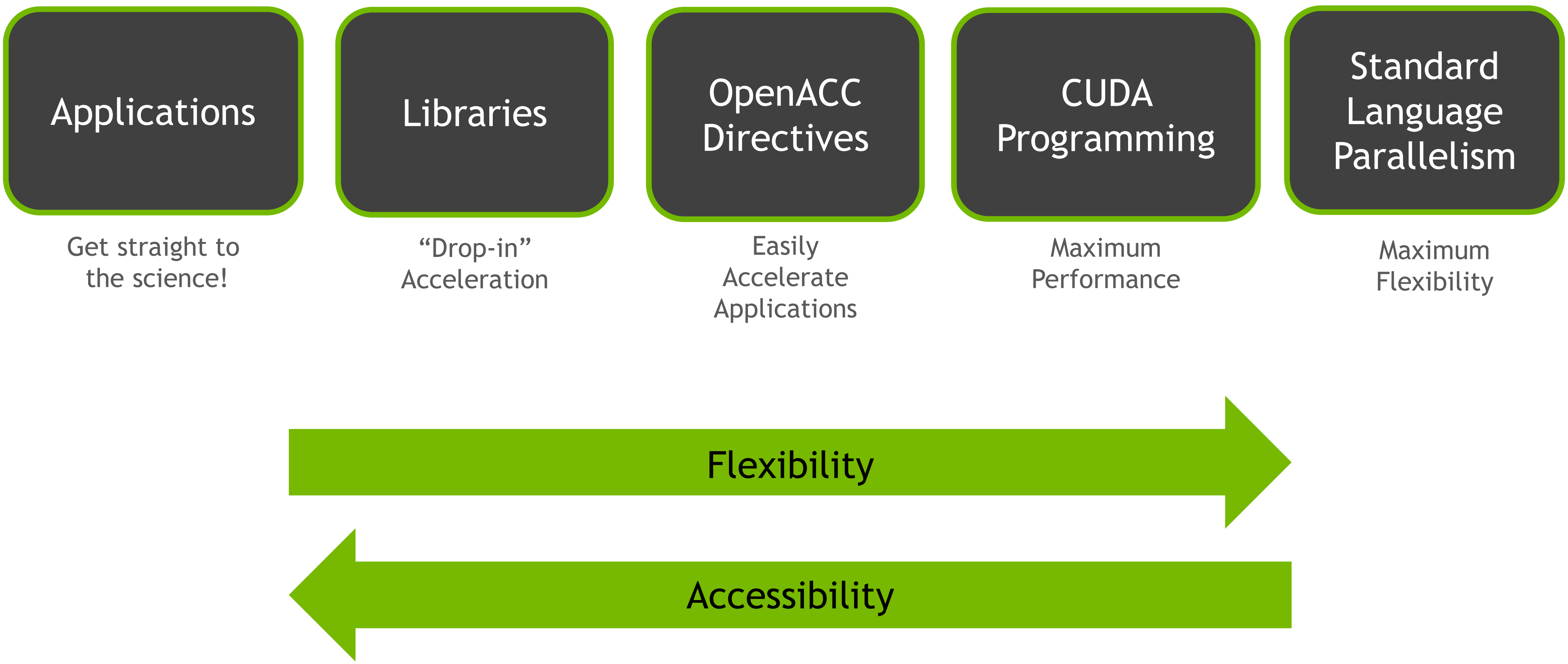


1. Copy input data from CPU memory to GPU memory
2. Load GPU program and execute, caching data on chip for performance
3. Copy results from GPU memory to CPU memory

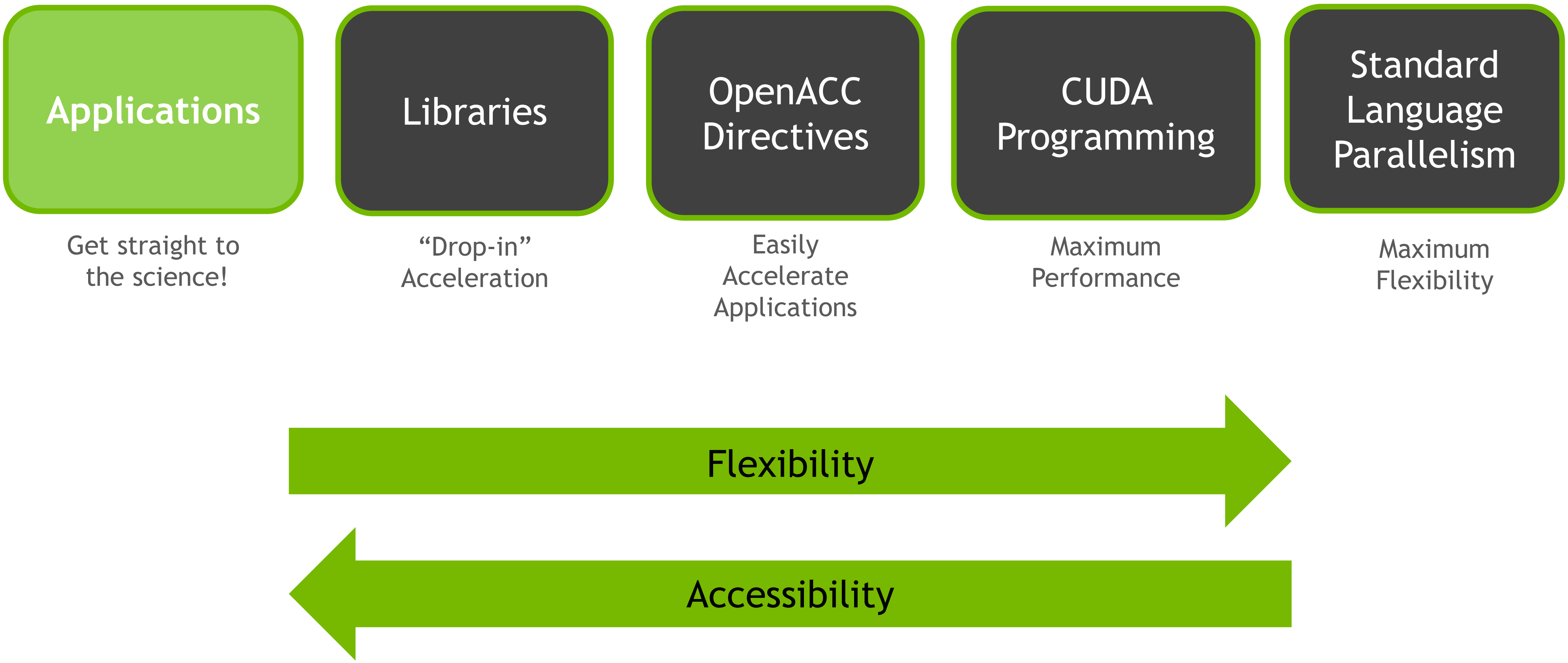
- A parallel computing platform and application programming interface (API) model created by NVIDIA
- Allows software developers and software engineers to use a CUDA-enabled GPUs for general purpose processing
- Backwards compatible
- The name CUDA was originally an acronym for Compute Unified Device Architecture

The Five Ways to Accelerate with GPUs

5 WAYS TO ACCELERATE WITH GPUS



5 WAYS TO ACCELERATE WITH GPUS



THOUSANDS OF GPU-ACCELERATED APPLICATIONS

Transforming Every Industry

ARTIFICIAL INTELLIGENCE
- PyTorch
- MXNet
- TensorFlow

...

CLIMATE & WEATHER
- Cosmos
- Gales
- WRF

...

COMPUTATIONAL FINANCE
- O-Quant Options Pricing
- MUREX
- MISYS

...

DATA SCIENCE & ANALYTICS
- Anaconda
- H20
- OmniSci

...

FEDERAL DEFENSE & OTHER
- ArcGIS Pro
- EVNI
- SocetGXP
- Cyllance
- FaceControl

...

LIFE SCIENCES
- Amber
- LAMMPS
- GROMACS
- NAMD
- Relion
- VASP

...

MANUFACTURING, CAD, & CAE
- Ansys Fluent
- Abaqus SIMULIA
- AutoCAD
- CST Studio Suite

...

MEDIA & ENTERTAINMENT
- DaVinci Resolve
- Premiere Pro CC
- Redshift Renderer

...

MEDICAL IMAGING
- aidoc
- PowerGrid
- RadiAnt

...

OIL & GAS
- Echelon
- RTM
- SPECFEM3D

...

RETAIL
- Everseen
- Deep North
- Third Eye Labs
- AWM
- Malong
- Clarifai
- Antuit

...

SUPERCOMPUTING & HER
- Chroma
- GTC
- MILC
- QUDA
- XGC

...

For a comprehensive list of all apps, please refer to GPU application catalog: https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/tesla-product-literature/gpu-applications-catalog.pdf

Sample GPU Accelerated Applications

See https://www.nvidia.com/en-us/gpu-accelerated-applications/

- Amber
- GROMACS
- LAMMPS
- NAMD
- Relion
- Chroma
- GTC
- MILC
- SPECFEM3D
- FUN3D

Standard Benchmark speedup on single A100 vs dual CPU

https://developer.nvidia.com/hpc-application-performance

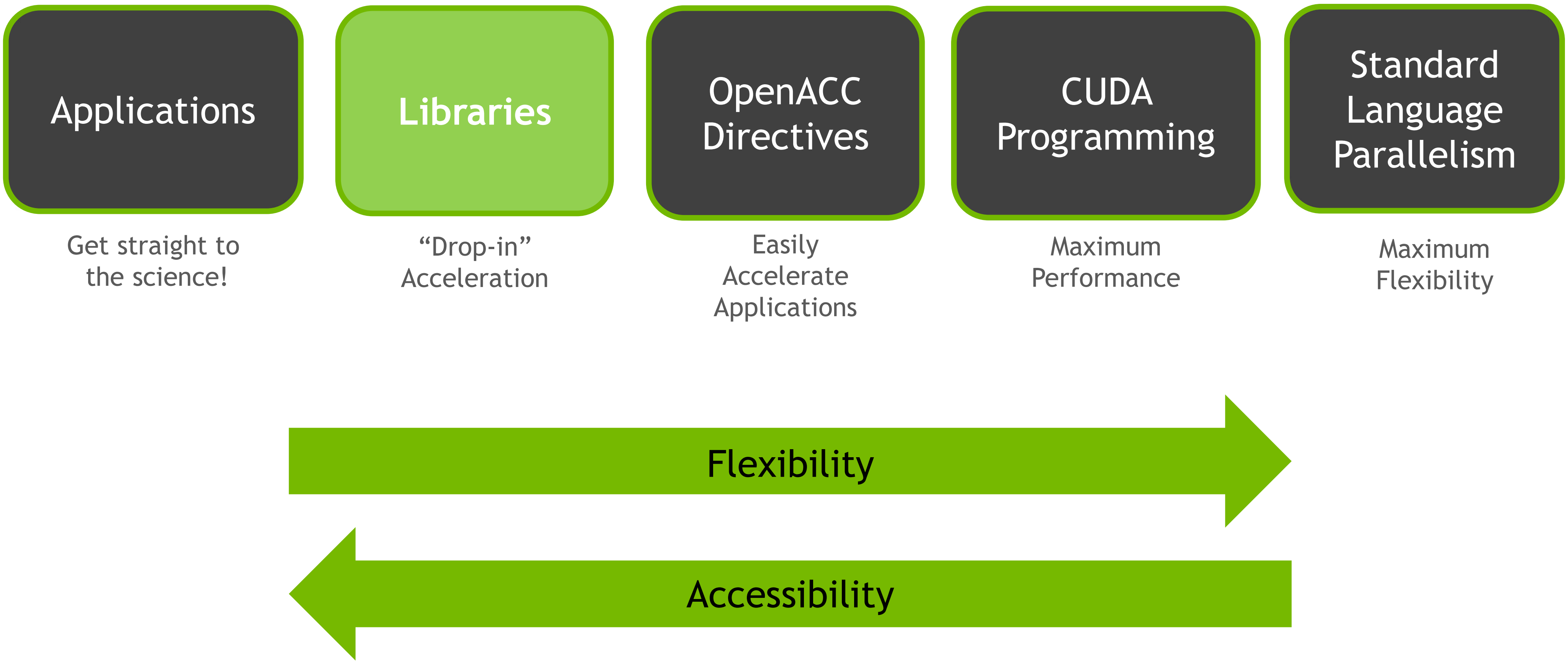
- Amber 13x – 39x
- GROMACS 6x – 9x
- LAMMPS 5x – 18x
- NAMD 6x – 8x
- Relion 4x – 5x
- Chroma 32x
- GTC 14x
- MILC 32x
- SPECFEM3D 29x
- FUN3D 13x

More Sample GPU Accelerated Applications

https://www.nvidia.com/en-us/gpu-accelerated-applications/

- Ansys Fluent
- ArcGIS Pro
- COMSOL
- MATLAB
- Mathematica
- ParaView
- TensorFlow
- PyTorch

5 WAYS TO ACCELERATE WITH GPUS

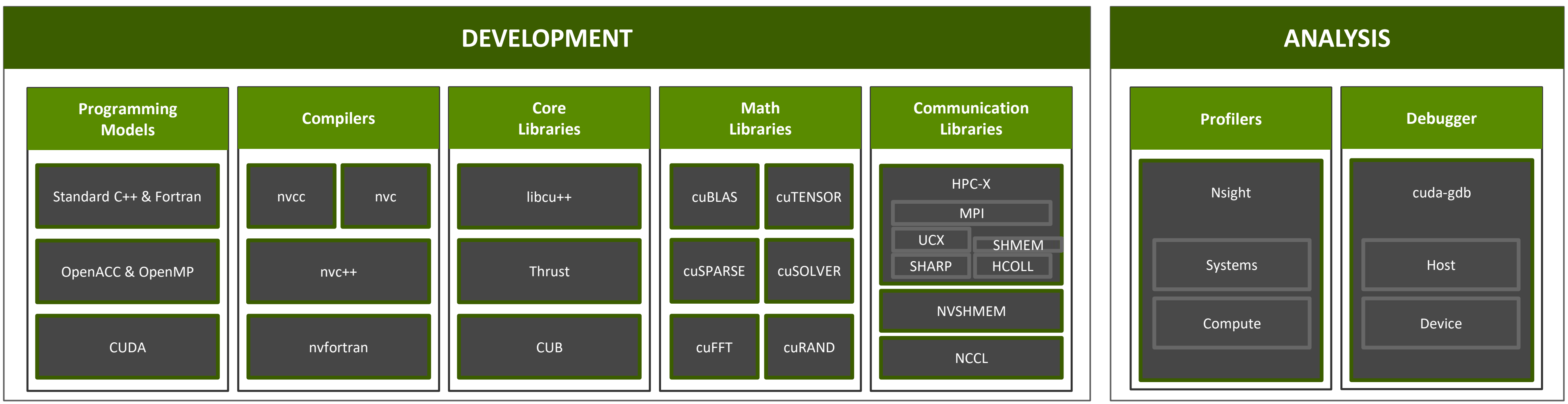


LIBRARIES: EASY, HIGH-QUALITY ACCELERATION

EASE OF USE	Using libraries enables GPU acceleration without in-depth knowledge of GPU programming
"DROP-IN"	Many GPU-accelerated libraries follow standard APIs, thus enabling acceleration with minimal code changes
QUALITY	Libraries offer high-quality implementations of functions encountered in a broad range of applications
PERFORMANCE	NVIDIA libraries are tuned by experts

NVIDIA HPC SDK

Available at developer.nvidia.com/hpc-sdk, on NGC, via Spack, and in the Cloud



Develop for the NVIDIA Platform: GPU, CPU and Interconnect
Libraries | Accelerated C++ and Fortran | Directives | CUDA
7-8 Releases Per Year | Freely Available

3 STEPS TO CUDA-ACCELERATED APPLICATION

Step 1: Substitute library calls with equivalent CUDA library calls

`saxpy ( … )` ► `cublasSaxpy ( … )`

Step 2: Manage data locality

- with CUDA: `cudaMalloc(), cudaMemcpy(), etc.`
- with CUBLAS: `cublasAlloc(), cublasSetVector(), etc.`

Step 3: Rebuild and link the CUDA-accelerated library

```
gcc myobj.o –l cublas
```

SAXPY is "Single-Precision A times X Plus Y"

GPU Accelerated Libraries (some examples)

https://developer.nvidia.com/how-to-cuda-libraries

CUBLAS – an implementation of BLAS (Basic Linear Algebra Subprograms).

CUFFT – a Fast Fourier Transform library with support for the FFTW API.

CURAND – provides facilities that focus on the simple and efficient generation of high-quality pseudorandom and quasi-random numbers.

CUSPARSE – contains a set of basic linear algebra subroutines used for handling sparse matrices.

cuSOLVER – GPU-accelerated dense and sparse direct solvers (LAPACK-like features)

CUDA Math Library – GPU-accelerated standard mathematical function library (Available to any CUDA C or CUDA C++ application simply by adding “#include math.h” in your source code)

Thrust – GPU-accelerated library of C++ parallel algorithms and data structures

nvJPEG – High performance GPU-accelerated library for JPEG decoding

ArrayFire – open source library for matrix, signal, and image processing

MAGMA – linear algebra routines for heterogeneous architectures

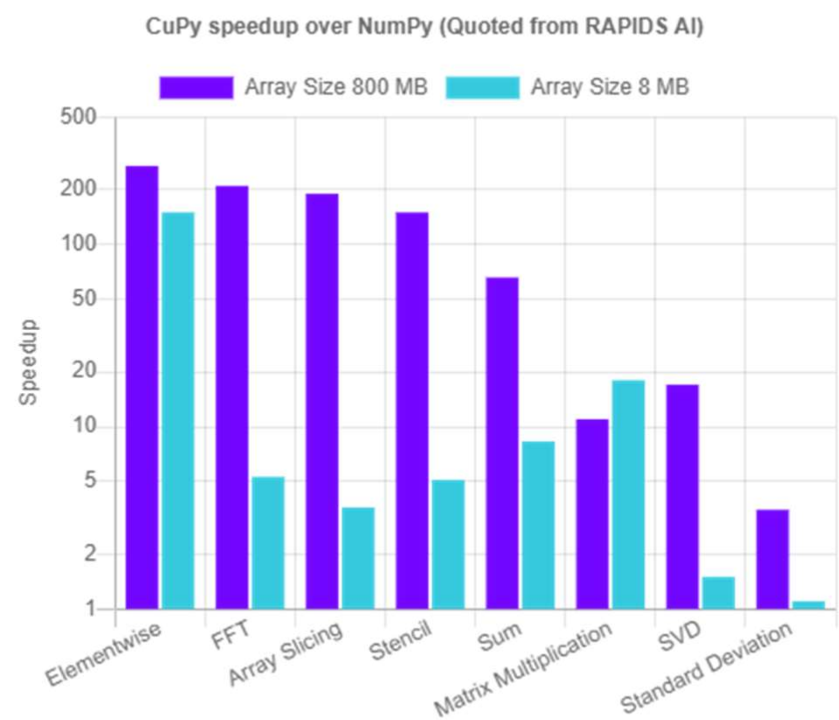
CHOLMOD – functions for sparse direct solvers

https://github.com/nvidia/cudalibrarysamples

CuPy

https://cupy.dev/

- Open-source array library for GPU-accelerated computing
- Interface is highly compatible with NumPy and SciPy
- Can be used as a drop-in replacement in most cases
- Just replace `numpy` and `scipy` with `cupy` and `cupyx.scipy`
- Speeds up some operations more than 100X



RAPIDS

https://rapids.ai/

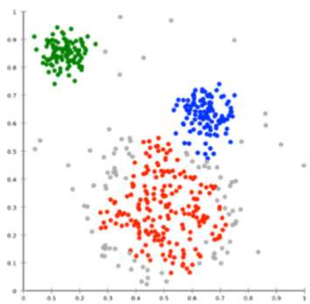
RAPIDS: a suite of open source software libraries and APIs gives you the ability to execute end-to-end data science and analytics pipelines entirely on GPUs. Licensed under Apache 2.0

Popular Libraries:

- **cuDF** – a pandas-like dataframe manipulation library
- **cuML** – GPU versions of algorithms in scikit-learn
- **cuSignal** – signal processing library based on SciPy Signal
- **cuGraph** – Network-X-like accelerated graph analytics library
- **cuSpatial** – GPU-accelerated GIS and spatiotemporal algorithms

ALGORITHMS

GPU-accelerated Scikit-Learn



Classification / Regression

Decision Trees / Random Forests
Linear/Lasso/Ridge/ElasticNet Regression
Logistic Regression
K-Nearest Neighbors
Support Vector Machine Classification and Regression
Naive Bayes

Inference

Random Forest / GBDT Inference (FIL)

Preprocessing

Text vectorization (TF-IDF / Count)
Target Encoding
Cross-validation / splitting

Clustering Decomposition & Dimensionality Reduction

K-Means
DBSCAN
Spectral Clustering
Principal Components
Singular Value Decomposition
UMAP
Spectral Embedding
T-SNE

Time Series

Holt-Winters
Seasonal ARIMA / Auto ARIMA

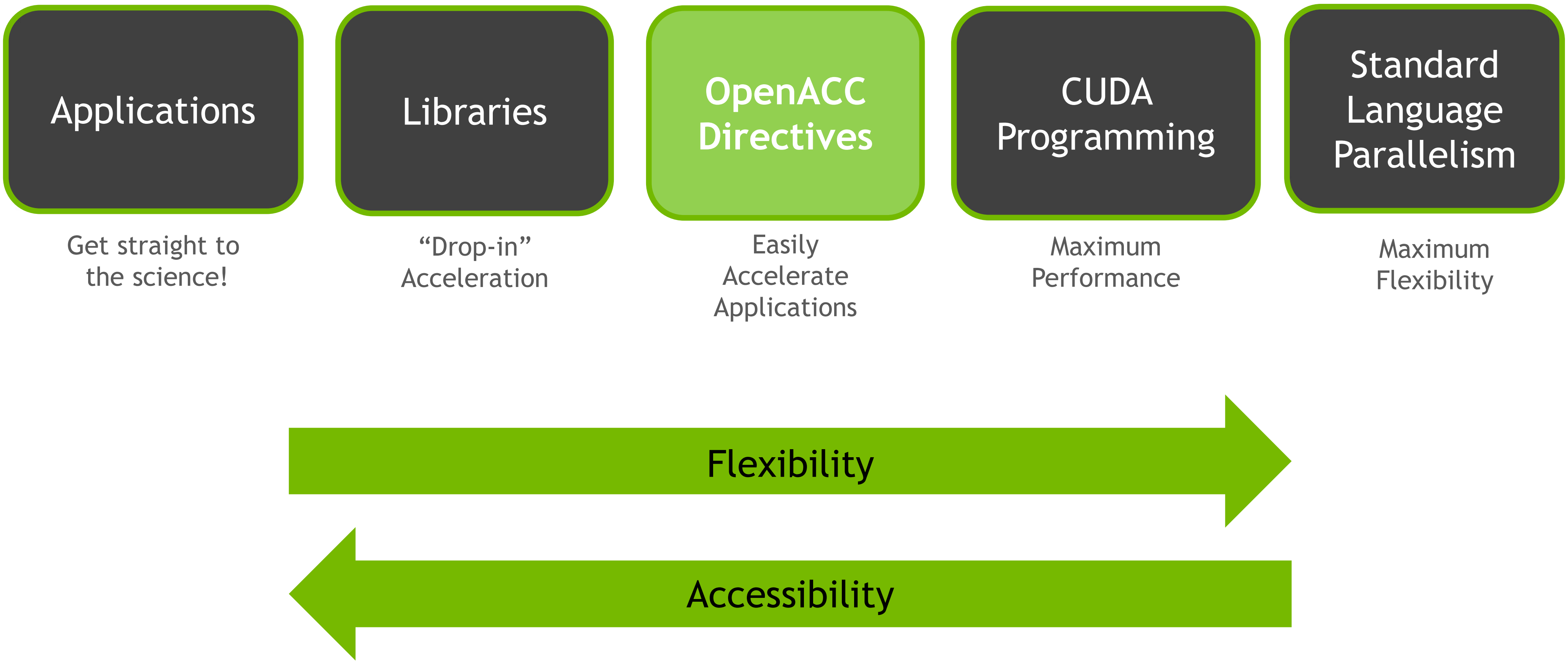
Cross Validation

Hyper-parameter Tuning

More to come!

https://github.com/rapidsai/cuml#supported-algorithms

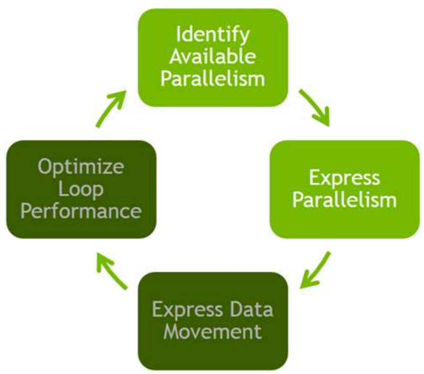
5 WAYS TO ACCELERATE WITH GPUS



OpenACC Directives

https://www.openacc.org/

OpenACC is a user-driven directive-based performance-portable parallel programming model. It is designed for scientists and engineers interested in porting their codes to a wide-variety of heterogeneous HPC hardware platforms and architectures with significantly less programming effort than required with a low-level model.



C

```
#pragma acc directive [clause [,] clause] …]
```

Often followed by a structured code block

Fortran

```
!$acc directive [clause [,] clause] …]
```

Often paired with a matching end directive surrounding a structured code block

```
!$acc end directive
```

- **Simple Compiler hints**
- **Compiler Parallelizes code**
- **Works on many-core GPUs & multicore CPUs**

https://www.gpuhackathons.org/

A VERY SIMPLE EXERCISE: SAXPY

SAXPY in C

```c
void saxpy(int n,
           float a,
           float *x,
           float *restrict y)
{
#pragma acc kernels
  for (int i = 0; i < n; ++i)
    y[i] = a*x[i] + y[i];
}

...
// Perform SAXPY on 1M elements
saxpy(1<<20, 2.0, x, y);
...
```

SAXPY in Fortran

```fortran
subroutine saxpy(n, a, x, y)
  real :: x(:), y(:), a
  integer :: n, i
$!acc kernels
  do i=1,n
    y(i) = a*x(i)+y(i)
  enddo
$!acc end kernels
end subroutine saxpy


...
$ Perform SAXPY on 1M elements
call saxpy(2**20, 2.0, x_d, y_d)
...
```

TOP HPC APPS ADOPTING OPENACC

OpenACC - Performance Portability And Ease of Programming

ANSYS Fluent Gaussian
VASP

3 of Top 10 Apps

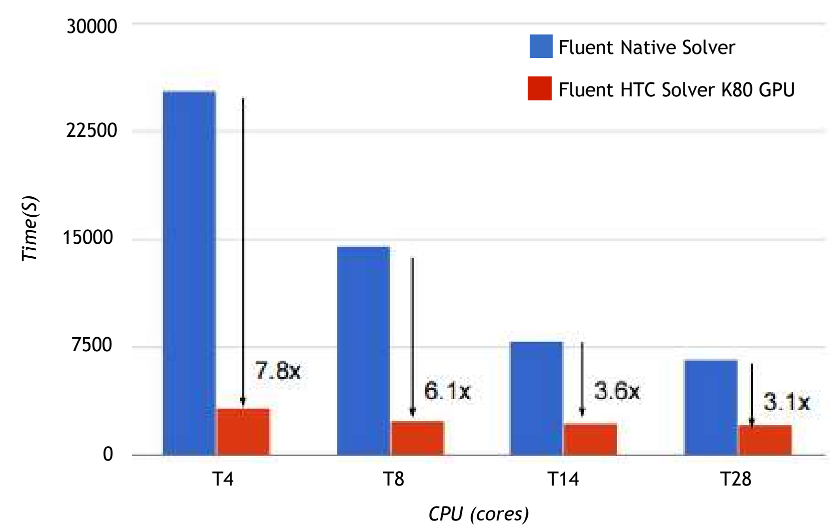
GTC
XGC
ACME
FLASH
LSDalton

5 ORNL CAAR Codes

COSMO
ELEPHANT
RAMSES
ICON
ORB5

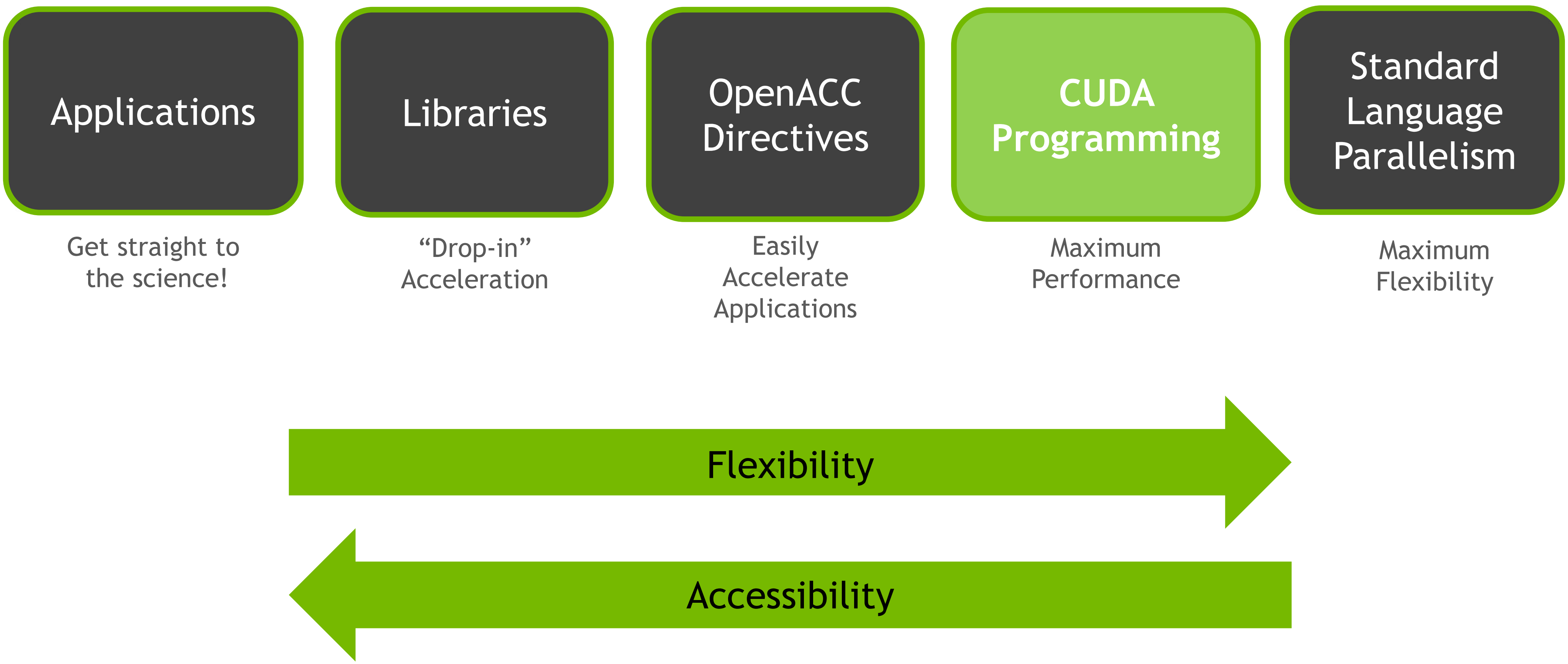
5 CSCS Codes

ANSYS Fluent R18.0 Radiation Solver



CPU: (Haswell EP) Intel(R) Xeon(R) CPU E5-2695 v3 @2.30GHz, 2 sockets, 28 cores
GPU: Tesla K80 12+12 GB, Driver 346.46

5 WAYS TO ACCELERATE WITH GPUS



CUDA Programming (ultimate control)

https://developer.nvidia.com/blog/even-easier-introduction-cuda/

CUDA gives you fine-level control over

- thread execution
- use of GPU memory hierarchy

Tune your code for optimal performance

Scale your parallel execution to multiple GPUs and multiple hosts using NCCL and MPI

CUDA API – C, C++, Fortran, Julia, Python

CUDA aware MPI (OpenMPI, MVAPICH, Spectrum MPI, and more)

CUDA C

```c
void saxpy_serial(int n,
                  float a,
                  float *x,
                  float *y)
{

  for (int i = 0; i < n; ++i)
    y[i] = a*x[i] + y[i];
}

// Perform SAXPY on 1M elements
saxpy_serial(4096*256, 2.0, x, y);
```

```c
__global__
void saxpy_parallel(int n,
                    float a,
                    float *x,
                    float *y)
{
  int i = blockIdx.x*blockDim.x +
          threadIdx.x;
  if (i < n) y[i] = a*x[i] + y[i];
}

// Perform SAXPY on 1M elements
saxpy_parallel<<<4096,256>>>(n,2.0,x,y);
```

http://developer.nvidia.com/cuda-toolkit

RAPID PARALLEL C++ DEVELOPMENT

- Resembles C++ STL
- High-level interface
  - Enhances developer productivity
  - Enables performance portability between GPUs and multicore CPUs
- Flexible
  - CUDA, OpenMP, and TBB backends
  - Extensible and customizable
  - Integrates with existing software
- Open source

```
// generate 32M random numbers on host
thrust::host_vector<int> h_vec(32 << 20);
thrust::generate(h_vec.begin(),
                 h_vec.end(),
                  rand);
// transfer data to device (GPU)
thrust::device_vector<int> d_vec = h_vec;
// sort data on device
thrust::sort(d_vec.begin(), d_vec.end());
// transfer data back to host
thrust::copy(d_vec.begin(),
             d_vec.end(),
             h_vec.begin());
```

http://developer.nvidia.com/thrust or http://thrust.googlecode.com

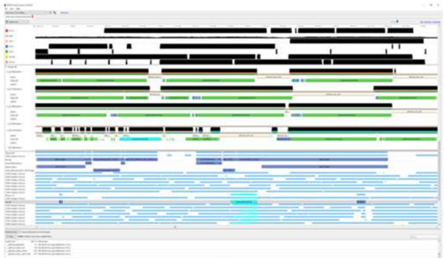
CUDA FORTRAN

- Program GPU using Fortran
  - Key language for HPC
- Simple language extensions
  - Kernel functions
  - Thread / block IDs
  - Device & data management
  - Parallel loop directives
- Familiar syntax
  - Use allocate, deallocate
  - Copy CPU-to-GPU with assignment (=)

http://developer.nvidia.com/cuda-fortran

```
module mymodule contains
  attributes(global) subroutine saxpy(n,a,x,y)
    real :: x(:), y(:), a,
    integer n, i
    attributes(value) :: a, n
    i = threadIdx%x+(blockIdx%x-1)*blockDim%x
    if (i<=n) y(i) = a*x(i) + y(i);
  end subroutine saxpy
end module mymodule

program main
  use cudafor; use mymodule
  real, device :: x_d(2**20), y_d(2**20)
  x_d = 1.0; y_d = 2.0
  call saxpy<<<4096,256>>>(2**20,3.0,x_d,y_d,)
  y = y_d
  write(*,*) 'max error=', maxval(abs(y-5.0))
end program main
```

COMPUTE DEVELOPER TOOLS



Nsight Systems

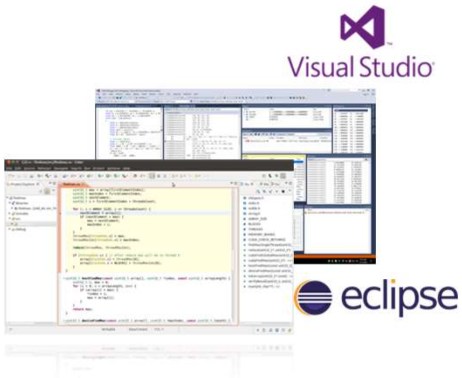
System-wide application algorithm tuning

Nsight Compute

CUDA Kernel Profiling and Debugging

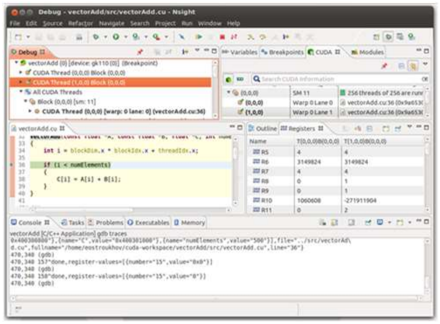
Nsight Graphics

Graphics Shader Profiling and Debugging



IDE Plugins

Nsight Eclipse Edition/Visual Studio (Editor, Debugger)



cuda-gdb

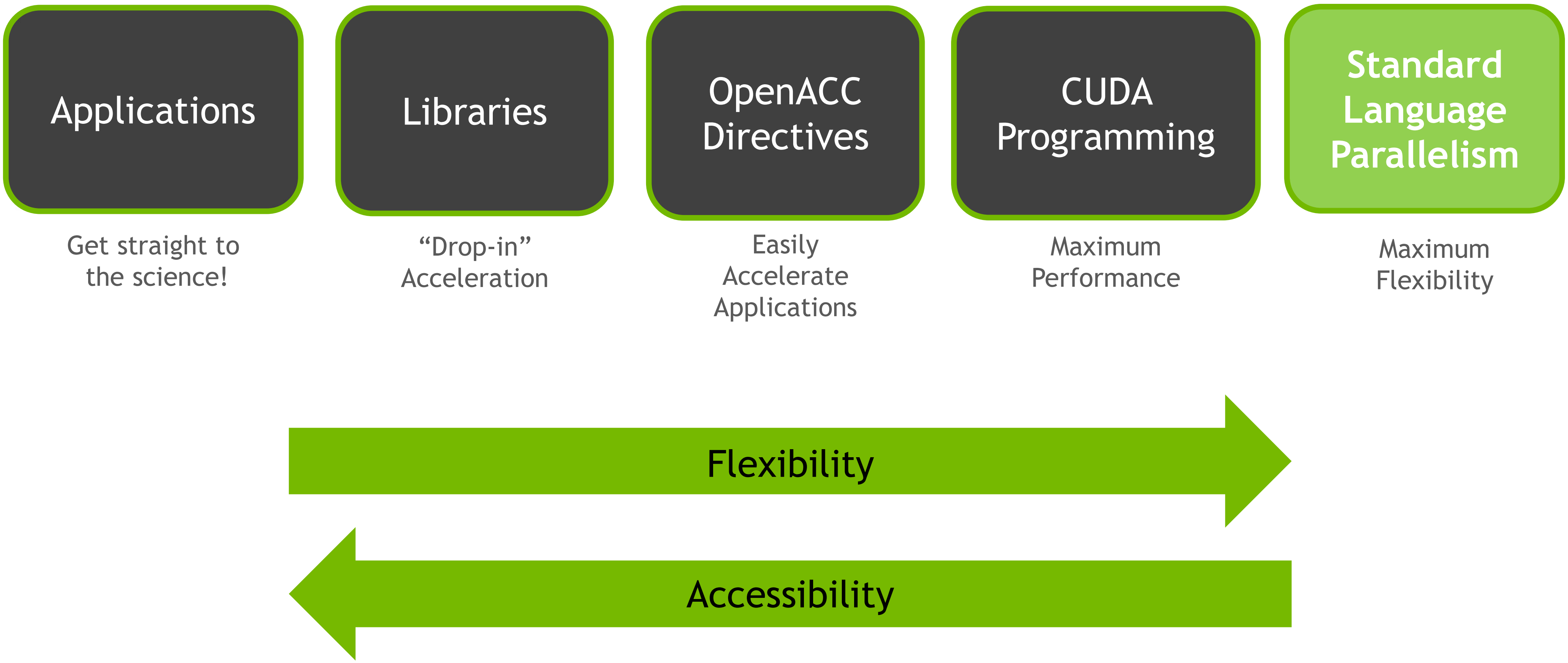
CUDA Kernel Debugging

```
//Out-of-bounds Array Access

__global__ void oobAccess(int* in, int* out)
{
    int bid = blockIdx.x;
    int tid = threadIdx.x;

    if (bid == 4)
    {
        out[tid] = in[dMem[tid]];
    }
}

int main()
{
    ...
    // Array of 8 elements, where element 4 causes the OOB
    std::array<int, Size> hMem = {0, 1, 2, 10, 4, 5, 6, 7};
    cudaMemcpy(d_mem, hMem.data(), size, cudaMemcpyHostToDevice);

    oobAccess<<<10, Size>>>(d_in, d_out);
    cudaDeviceSynchronize();
    ...

$ /usr/local/cuda-11.0/Sanitizer/compute-sanitizer --destroy-on-device-error kernel --show-backtrace no basic
========= COMPUTE-SANITIZER
Device: Tesla T4
========= Invalid __global__ read of size 4 bytes
=========     at 0x480 in /tmp/CUDA11.0/ComputeSanitizer/Tests/Memcheck/basic/basic.cu:40:oobAccess(int*,int*)
=========     by thread (3,0,0) in block (4,0,0)
=========     Address 0x7f551f200028 is out of bounds
```

Compute Sanitizer

Memory, Race Checking

5 WAYS TO ACCELERATE WITH GPUS



STANDARD LANGUAGE PROGRAMMING

ACCELERATED STANDARD LANGUAGES

ISO C++

```
std::transform(par, x, x+n, y,
  y,[=](float x, float y){
      return y + a*x;
  }
);
```

```
matrix_product(par, mA, mB,
mC);
```

ISO Fortran

```
do concurrent (i = 1:n)
   y(i) = y(i) + a*x(i)
enddo
```

```
C = matmul(A, B)
```

Python

```
import cunumeric as np
…
def saxpy(a, x, y):
    y[:] += a*x
```

```
c = np.matmul(a, b)
```

PLATFORM SPECIALIZATION

CUDA

```
__global__
void saxpy(int n, float a,
            float *x, float *y) {
  int i = blockIdx.x*blockDim.x +
          threadIdx.x;
  if (i < n) y[i] += a*x[i];
}

int main(void) {
  ...
  cudaMemcpy(d_x, x, ...);
  cudaMemcpy(d_y, y, ...);

  saxpy<<<(N+255)/256,256>>>(...);

  cudaMemcpy(y, d_y, ...);
```

https://developer.nvidia.com/blog/accelerating-standard-c-with-gpus-using-stdpar/
https://developer.nvidia.com/blog/accelerating-fortran-do-concurrent-with-gpus-and-the-nvidia-hpc-sdk/
https://developer.nvidia.com/cunumeric

HPC PROGRAMMING IN ISO C++

ISO is the place for portable concurrency and parallelism

Preview support coming to NVC++

C++17

Parallel Algorithms

- In NVC++
- Parallel and vector concurrency

Forward Progress Guarantees

- Extend the C++ execution model for accelerators

Memory Model Clarifications

- Extend the C++ memory model for accelerators

C++20

Scalable Synchronization Library

- Express thread synchronization that is portable and scalable across CPUs and accelerators
- In libcu++:
  - `std::atomic<T>`
  - `std::barrier`
  - `std::counting_semaphore`
  - `std::atomic<T>::wait/notify_*`
  - `std::atomic_ref<T>`

C++23 and Beyond

Executors / Senders-Recievers

- Simplify launching and managing parallel work across CPUs and accelerators

`std::mdspan/mdarray`

- HPC-oriented multi-dimensional array abstractions.

Range-Based Parallel Algorithms

- Improved multi-dimensional loops

Linear Algebra

- C++ standard algorithms API to linear algebra
- Maps to vendor optimized BLAS libraries

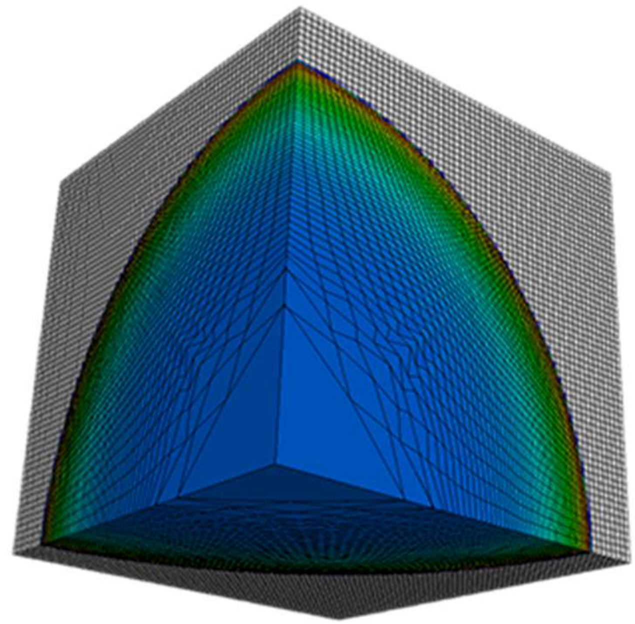
Extended Floating Point Types

- First-class support for formats new and old: `std::float16_t/float64_t`

C++17 PARALLEL ALGORITHMS

Lulesh Hydrodynamics Mini-app

- ~9000 lines of C++
- Parallel versions in MPI, OpenMP, OpenACC, CUDA, RAJA, Kokkos, ISO C++…
- Designed to stress compiler vectorization, parallel overheads, on-node parallelism



codesign.llnl.gov/lulesh

```
static inline
void CalcHydroConstraintForElems(Domain &domain, Index_t length,
    Index_t *regElemlist, Real_t dvovmax, Real_t& dthydro)
{
#if _OPENMP
  const Index_t threads = omp_get_max_threads();
  Index_t hydro_elem_per_thread[threads];
  Real_t dthydro_per_thread[threads];
#else
  Index_t threads = 1;
  Index_t hydro_elem_per_thread[1];
  Real_t dthydro_per_thread[1];
#endif
#pragma omp parallel firstprivate(length, dvovmax)
  {
    Real_t dthydro_tmp = dthydro ;
    Index_t hydro_elem = -1 ;
#if _OPENMP
    Index_t thread_num = omp_get_thread_num();
#else
    Index_t thread_num = 0;
#endif
#pragma omp for
    for (Index_t i = 0 ; i < length ; ++i) {
      Index_t indx = regElemlist[i] ;

      if (domain.vdov(indx) != Real_t(0.)) {
        Real_t dtdvov = dvovmax / (FABS(domain.vdov(indx))+Real_t(1.e-20)) ;

        if ( dthydro_tmp > dtdvov ) {
          dthydro_tmp = dtdvov ;
          hydro_elem = indx ;
        }
      }
    }
    dthydro_per_thread[thread_num] = dthydro_tmp ;
    hydro_elem_per_thread[thread_num] = hydro_elem ;
  }
  for (Index_t i = 1; i < threads; ++i) {
    if(dthydro_per_thread[i] < dthydro_per_thread[0]) {
      dthydro_per_thread[0] = dthydro_per_thread[i];
      hydro_elem_per_thread[0] = hydro_elem_per_thread[i];
    }
  }
  if (hydro_elem_per_thread[0] != -1) {
    dthydro = dthydro_per_thread[0] ;
  }
  return ;
}
```

C++ with OpenMP

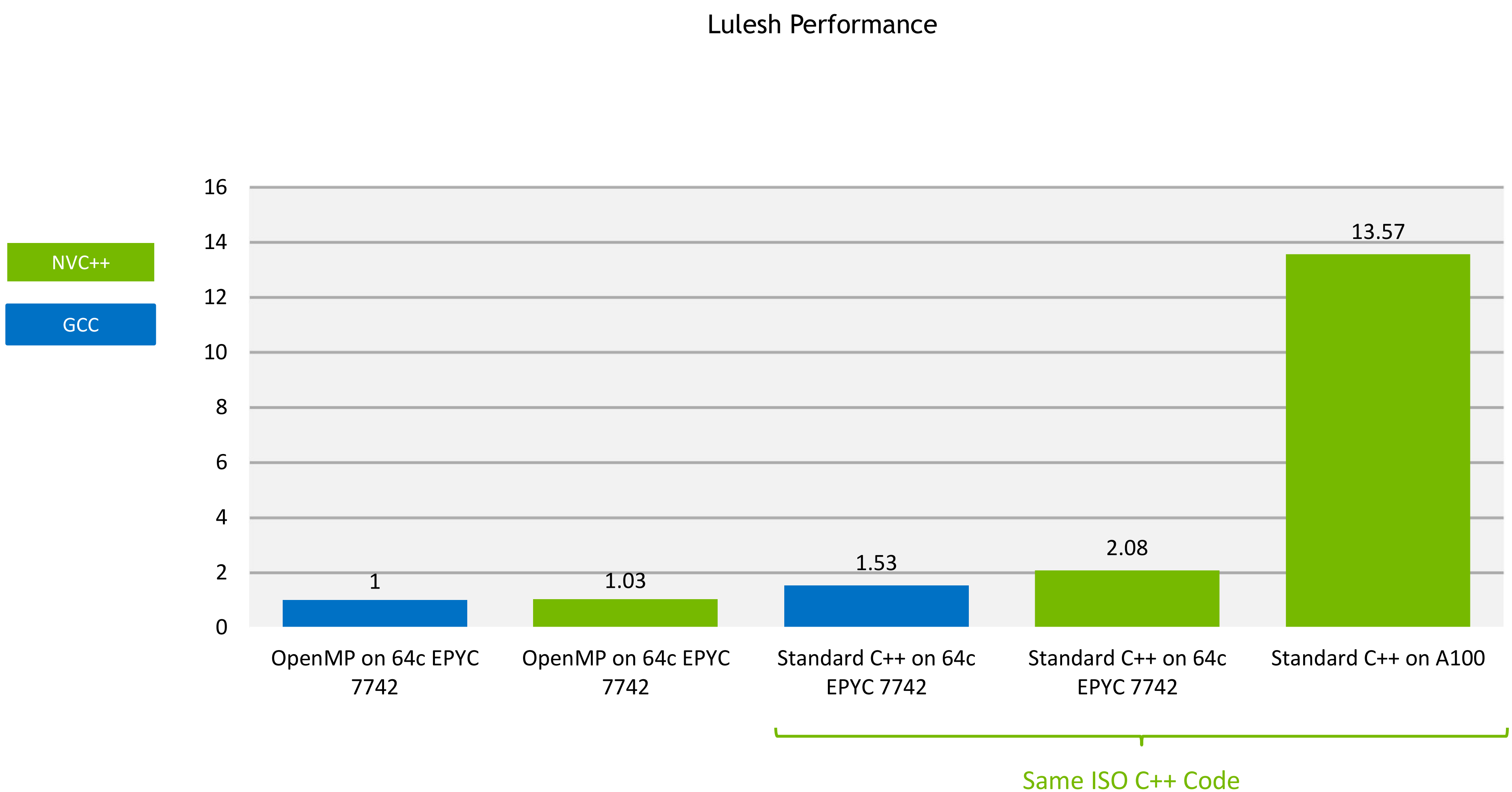
STANDARD C++

- Composable, compact and elegant
- Easy to read and maintain
- ISO Standard
- Portable – nvc++, g++, icpc, MSVC, …

```
static inline
void CalcHydroConstraintForElems(Domain &domain, Index_t length,
    Index_t *regElemlist, Real_t dvovmax, Real_t &dthydro)
{
  dthydro = std::transform_reduce(
    std::execution::par, counting_iterator(0), counting_iterator(length),
    dthydro, [](Real_t a, Real_t b) { return a < b ? a : b; },
    [=, &domain](Index_t i)
  {
      Index_t indx = regElemlist[i];
      if (domain.vdov(indx) == Real_t(0.0)) {
        return std::numeric_limits<Real_t>::max();
      } else {
        return dvovmax / (std::abs(domain.vdov(indx)) + Real_t(1.e-20));
      }
  });
}
```

Standard C++

C++ STANDARD PARALLELISM



ACCELERATED STANDARD LANGUAGES

Parallel performance for wherever your code runs

ISO C++

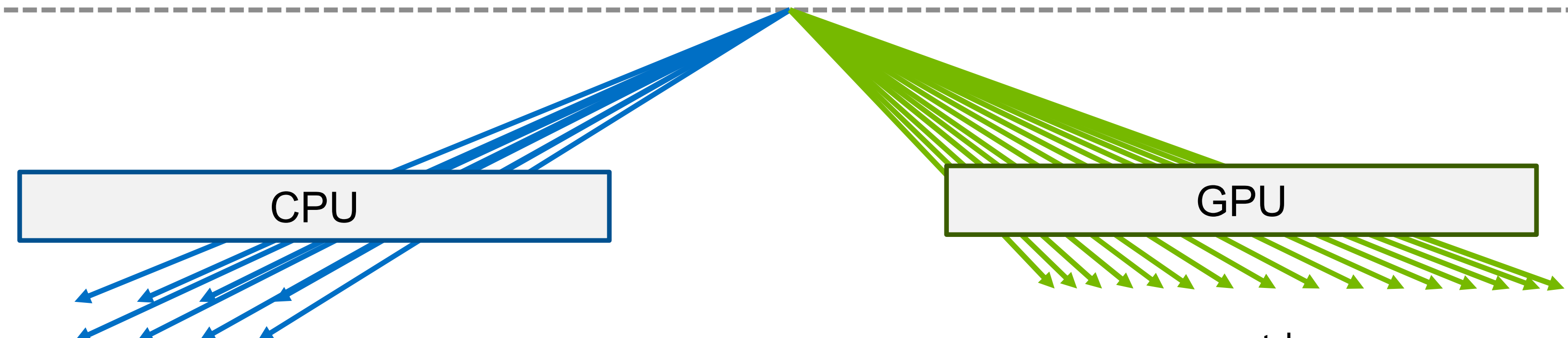
```
std::transform(par, x, x+n, y,
   y,[=](float x, float y){
      return y + a*x;
    }
);
```

ISO Fortran

```
do concurrent (i = 1:n)
   y(i) = y(i) + a*x(i)
enddo
```

Python

```
import cunumeric as np
…
def saxpy(a, x, y):
    y[:] += a*x
```



nvc++ -stdpar=multicore
nvfortran -stdpar=multicore
legate -cpus 16 saxpy.py

nvc++ -stdpar=gpu
nvfortran -stdpar=gpu
legate -gpus 1 saxpy.py

5 WAYS TO ACCELERATE WITH GPUS

Applications	Libraries	OpenACC Directives	CUDA Programming	Standard Language Parallelism
Get straight to the science!	"Drop-in" Acceleration	Easily Accelerate Applications	Maximum Performance	Maximum Flexibility

Flexibility →

← Accessibility

NVIDIA

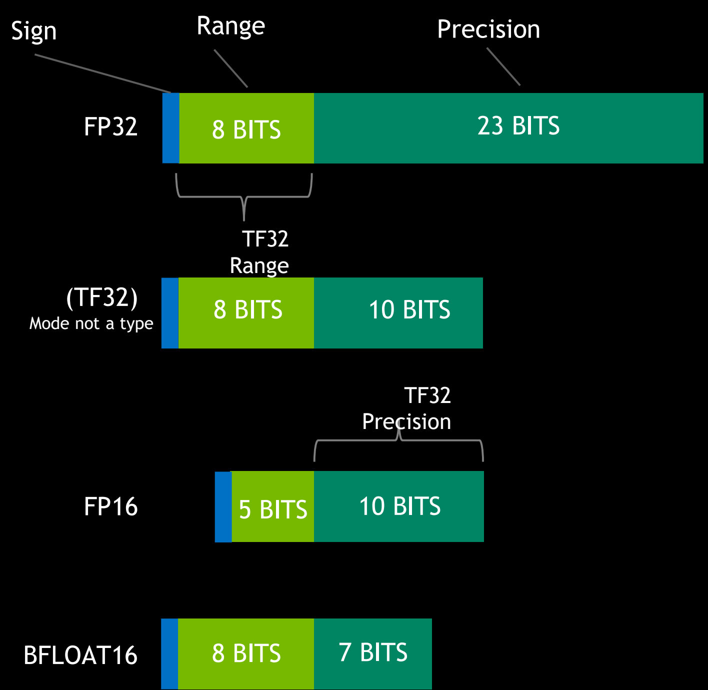
Important GPU Features and System Architecture

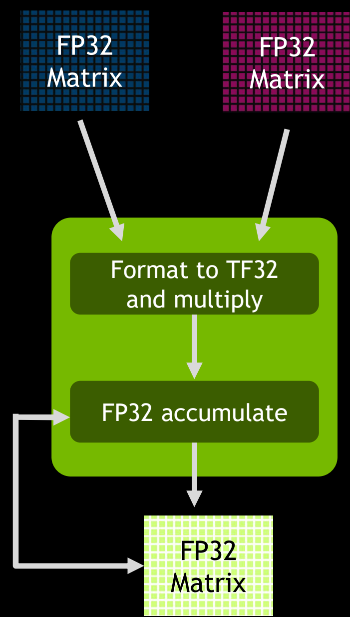
Tensor Cores and Mixed Precision

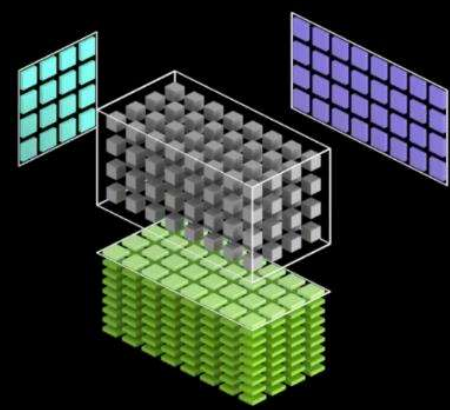
Tensor Cores are **programmable matrix-multiply-and-accumulate units**
cuBLAS uses Tensor Cores to speed up GEMM computations
Tensor Cores enable mixed-precision computing, dynamically adapting calculations to accelerate throughput while preserving accuracy

	Hopper	Ampere	Turing	Volta
Supported Tensor Core precisions	FP64, TF32, bfloat16, FP16, FP8, INT8	FP64, TF32, bfloat16, FP16, INT8, INT4, INT1	FP16, INT8, INT4, INT1	FP16
Supported CUDA® Core precisions	FP64, FP32, FP16, bfloat16, INT8	FP64, FP32, FP16, bfloat16, INT8	FP64, FP32, FP16, INT8	FP64, FP32, FP16, INT8

TF32 TENSOR CORES

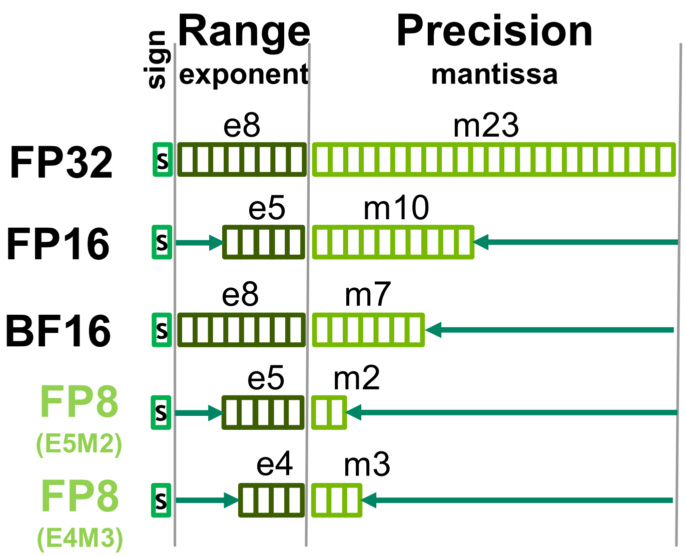


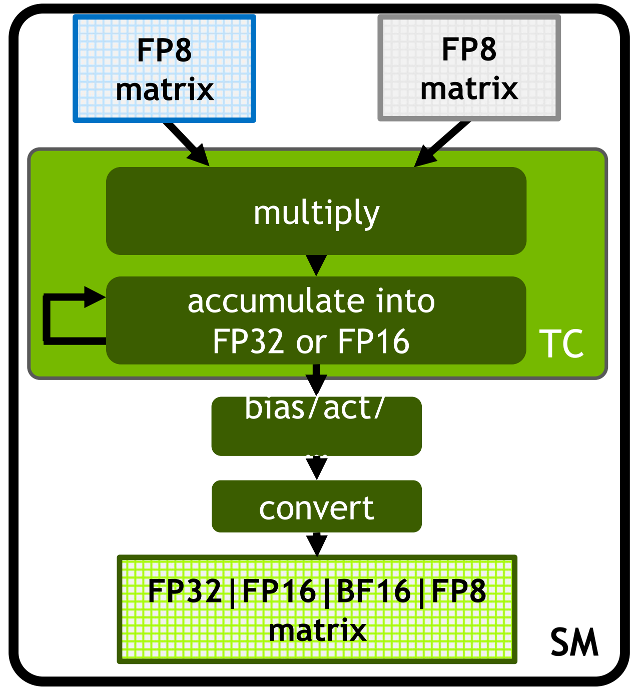




- Range of FP32 and Precision of FP16
- Input in FP32 and Accumulation in FP32
- No Code Change Speed-up for Training
    - Up to 8x more throughput compared to FP32 on A100
    - Up to 10x compared to FP32 on V100

INSIDE 8-BIT FLOATING POINT (FP8)





Allocate 1 bit to either range or precision

Support for multiple accumulator and output types

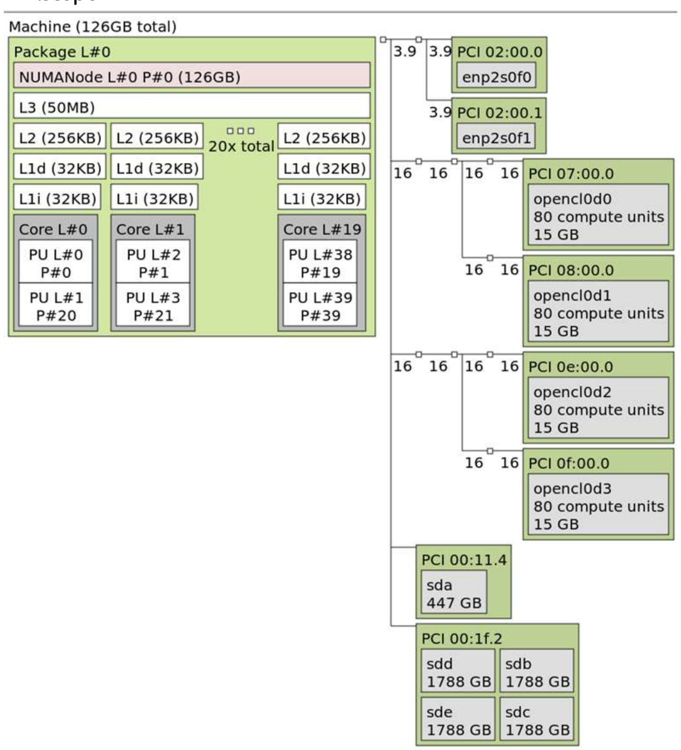
2x throughput & half footprint of FP16/BF16

Scaling to Multiple GPUs
GPU to GPU communication

GPU TOPOLOGY

nvidia-smi topo -m

lstopo



	GPU0	GPU1	GPU2	GPU3	CPU Affinity	NUMA Affinity
GPU0	X	PIX	PHB	PHB	0-39	N/A
GPU1	PIX	X	PHB	PHB	0-39	N/A
GPU2	PHB	PHB	X	PIX	0-39	N/A
GPU3	PHB	PHB	PIX	X	0-39	N/A

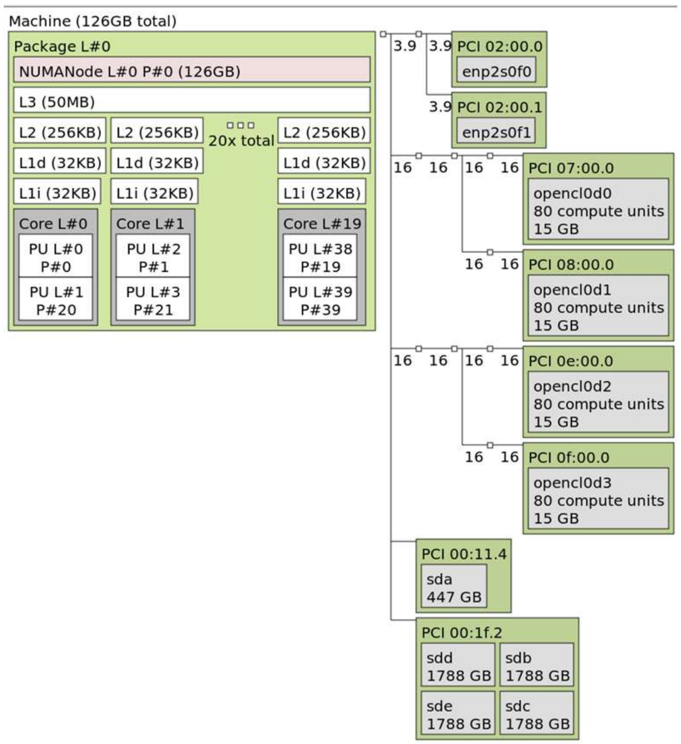
Legend:

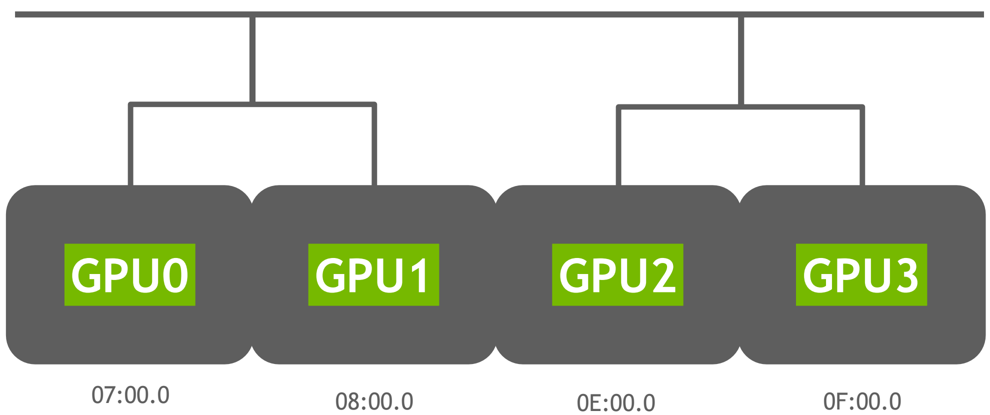
X = Self
SYS = Connection traversing PCIe as well as the SMP interconnect between NUMA nodes (e.g., QPI/UPI)
NODE = Connection traversing PCIe as well as the interconnect between PCIe Host Bridges within a NUMA node
PHB = Connection traversing PCIe as well as a PCIe Host Bridge (typically the CPU)
PXB = Connection traversing multiple PCIe bridges (without traversing the PCIe Host Bridge)
PIX = Connection traversing at most a single PCIe bridge
NV# = Connection traversing a bonded set of # NVLinks

WHY DOES GPU TOPOLOGY MATTER?

nvidia-smi topo -m

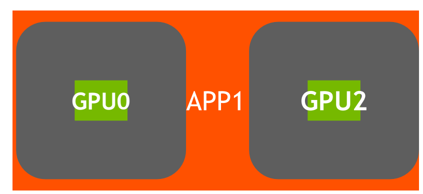
lstopo





- No penalty for single GPU applications
- Latency impact on multi-GPU applications

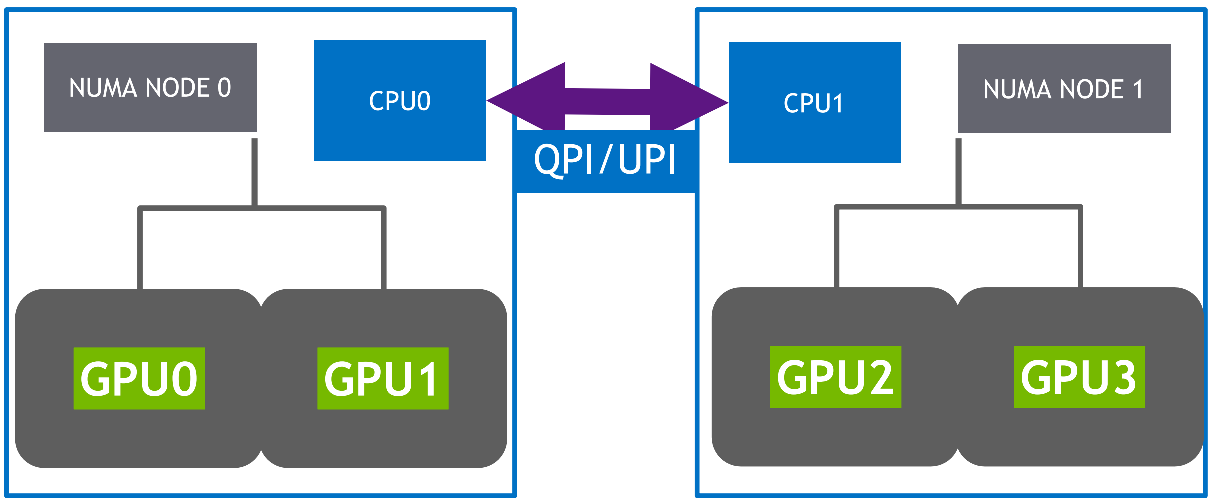
Faster



Slower

WHY DOES GPU TOPOLOGY MATTER?

nvidia-smi topo -m



What can go wrong?

GPU2 assigned to a VM on CPU0

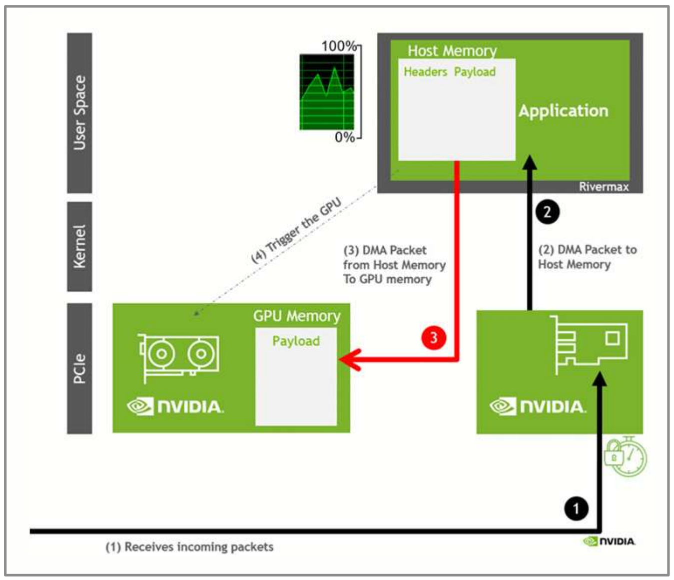
P2P, h2d, d2d and d2h bandwidth inconsistencies if the devices are not set

Tinker with

```
$export CUDA_VISIBLE_DEVICES=0,1
$export CUDA_VISIBLE_DEVICES=0,2
```

NVIDIA GPUDirect RDMA

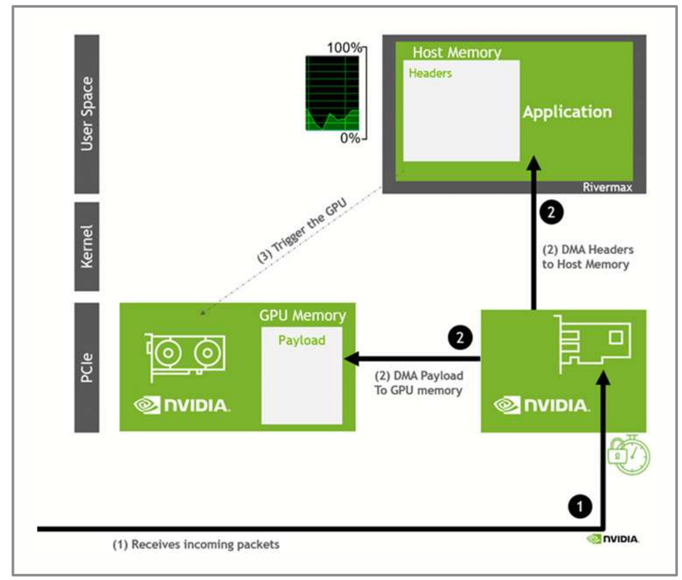
10X Higher Performance



No GPUDirect

Network Handled by CPU and CPU-Memory

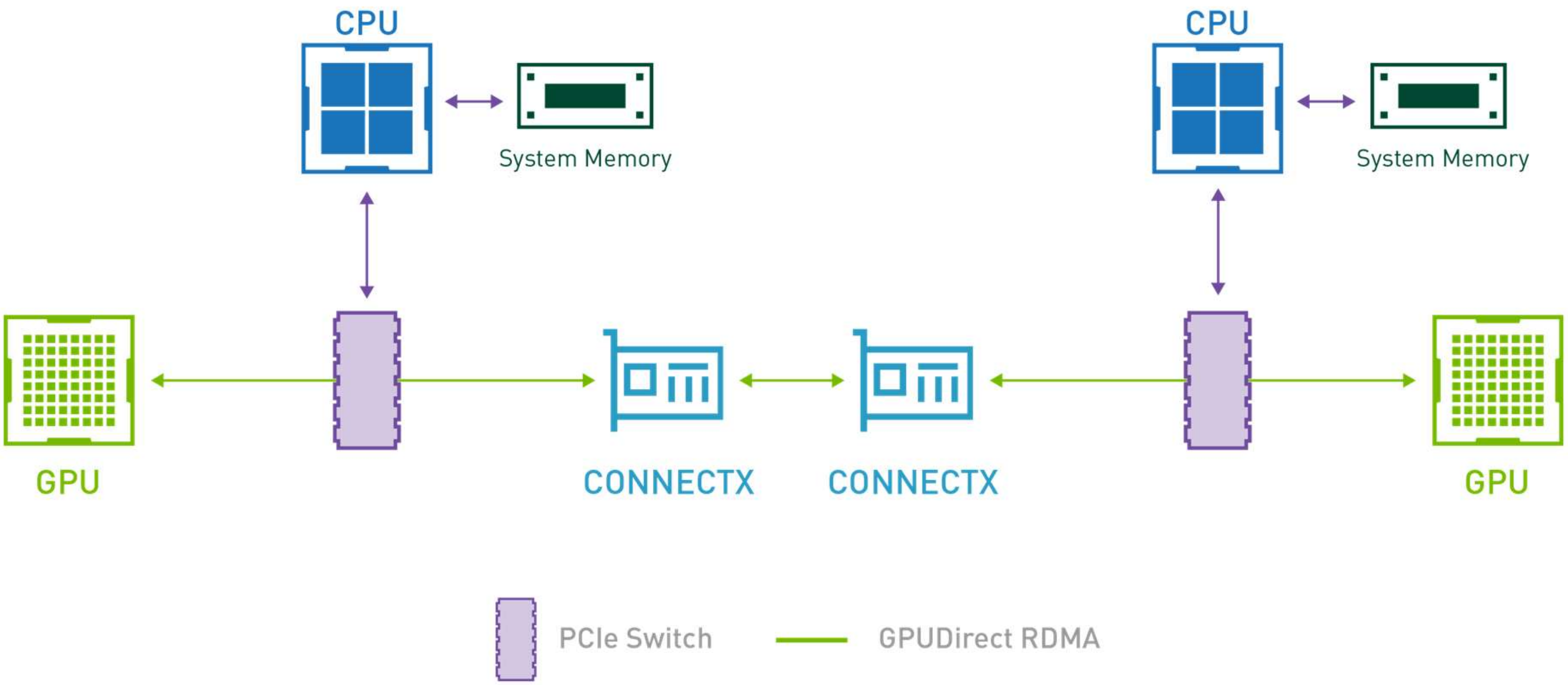
2	Full Copy Operations	0
2	PCIe Transactions	1
↓	GPU Utilization	↑
↑	CPU Usage	↓
↑	Latency	↓



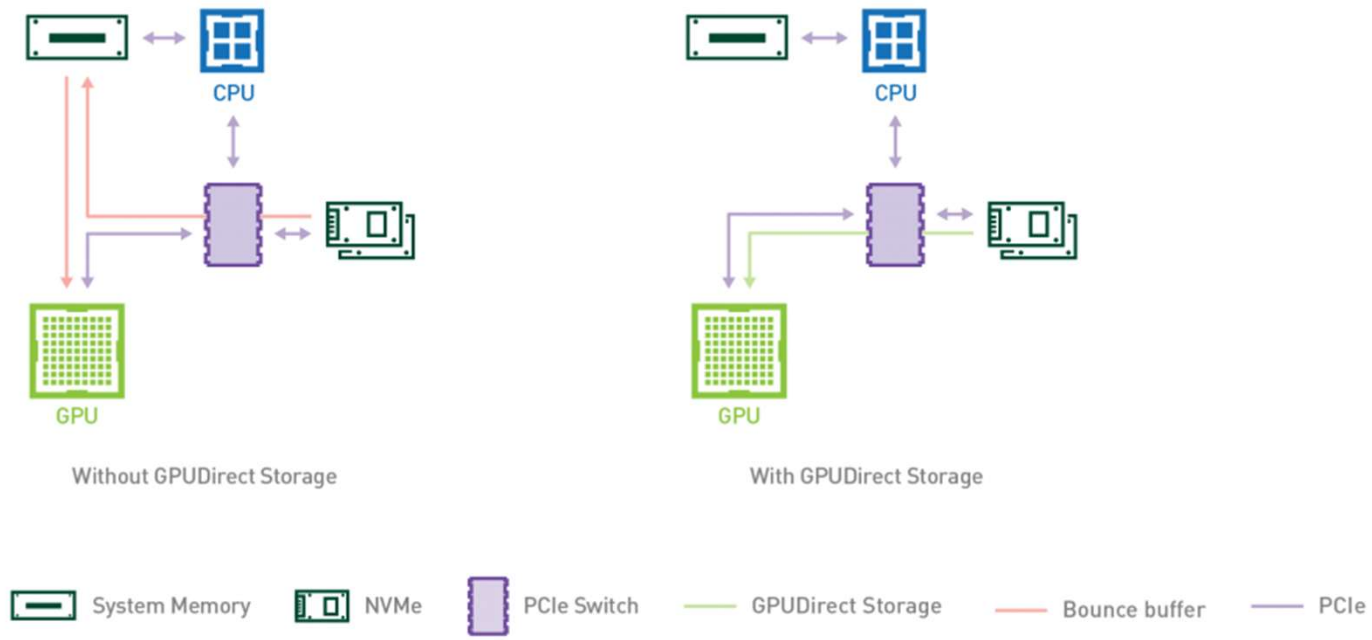
GPUDirect

Network Goes Directly to GPU Memory

NVIDIA GPUDirect RDMA



NVIDIA GPUDirect Storage



Faster GPU to GPU communication

NVLINK

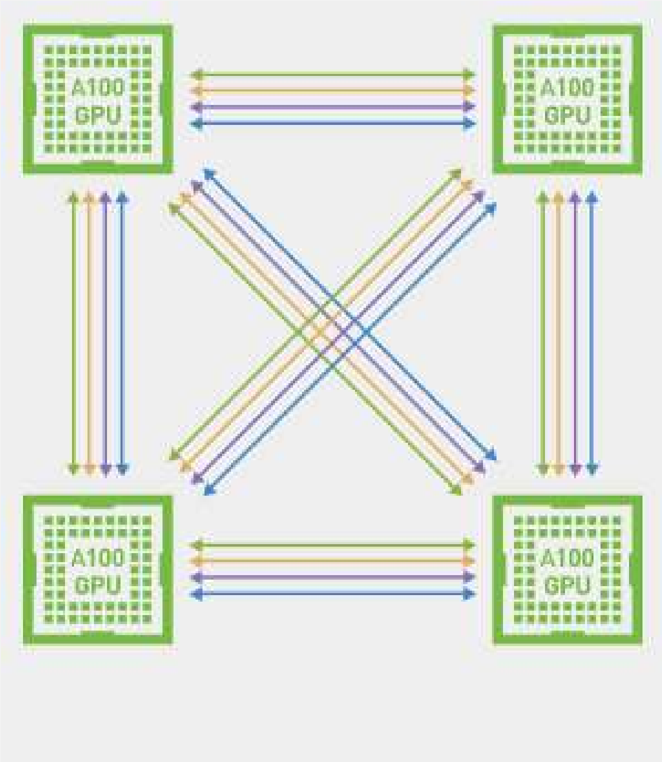
2-way all-to-all connection
25 GB/s per link in each direction
12 links per H100
4 links per bridge
600 GB/s GPU-to-GPU

NVIDIA A100 PCIe with NVLink GPU-to-GPU connection

H100 80GB PCIE
With NVLINK bridges

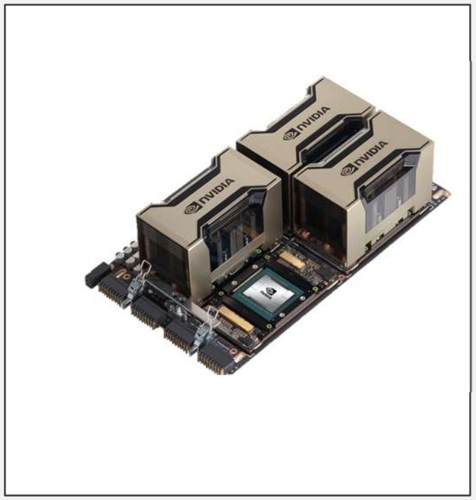
NVLINK

4-way all-to-all connection
25 GB/s per link in each direction
18 links per H100
900 GB/s GPU-to-GPU



NVIDIA A100 with NVLink GPU-to-GPU connections

HGX H100 4-GPU

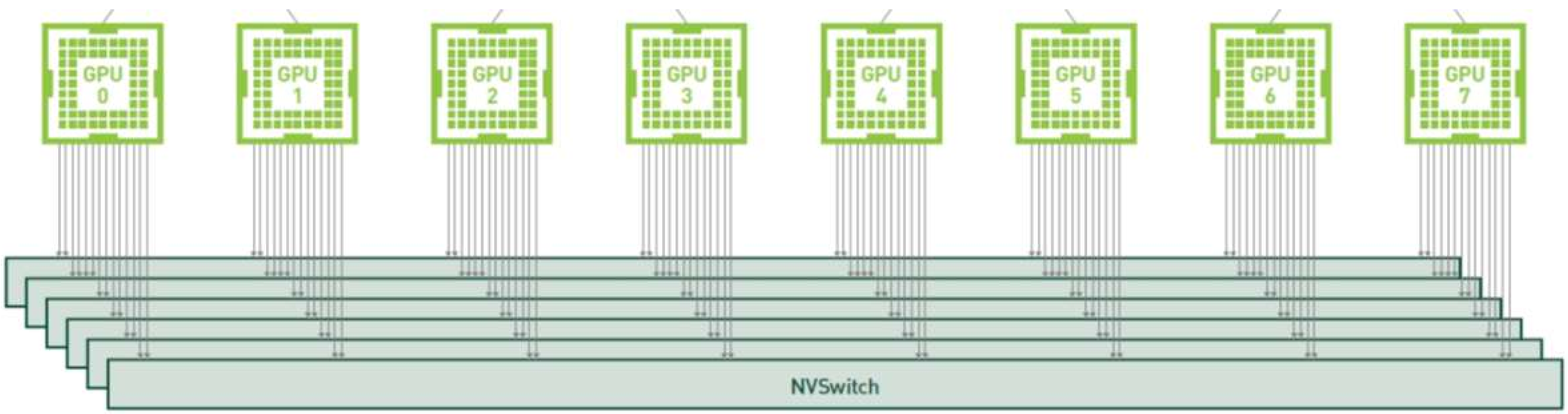


Scale-Up – Mixed AI & HPC

4 H100s, Fully Connected w/ shared NVLinks

NVSWITCH

8-way all-to-all connection
25 GB/s per link in each direction
18 links per H100
900 GB/s GPU-to-GPU



Data Center GPUs

NVIDIA HOPPER & ADA LOVELACE DATA CENTER GPUS

NVIDIA H100

Highest Perf Compute
AI, HPC, Data Processing

Fastest Compute, FP64
Up to 7 MIG instances

H100
350W & 700W | 80G
2-slot FHFL | NVLINK

NVIDIA L40S

AI & High Performance Graphics Visual Computing

AI and Graphics Performance

L40S
350W | 48GB
2-slot FHFL

NVIDIA L40

Highest Perf Graphics
Visual Computing

Fastest RT Graphics
Largest render models

L40
300W | 48GB
2-slot FHFL

NVIDIA L4

Mainstream Graphics &
Video with AI

4K Cloud Gaming
Graphics, Video with AI

L4
72W | 24GB
1-slot FHFL

Compute ← → *Graphics*

NVIDIA H100 SXM5

Unprecedented Performance, Scalability, and Security for Every Data Center

HIGHEST AI AND HPC PERFORMANCE

4PF FP8 (6X)| 2PF FP16 (3X)| 1PF TF32 (3X)| 60TF FP64 (3X)
3TB/s (1.5X), 80GB HBM3 memory

TRANSFORMER MODEL OPTIMIZATIONS

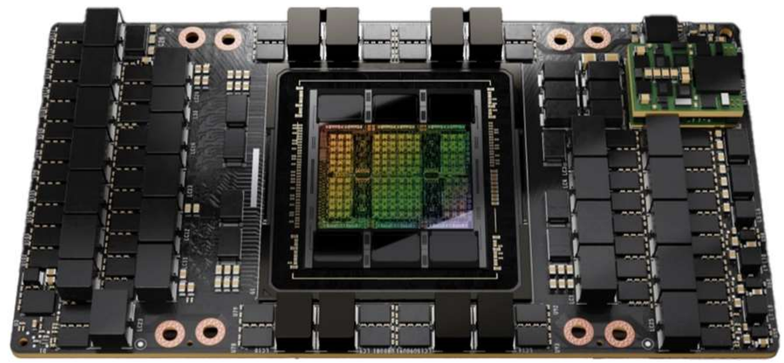
6X faster on largest transformer models

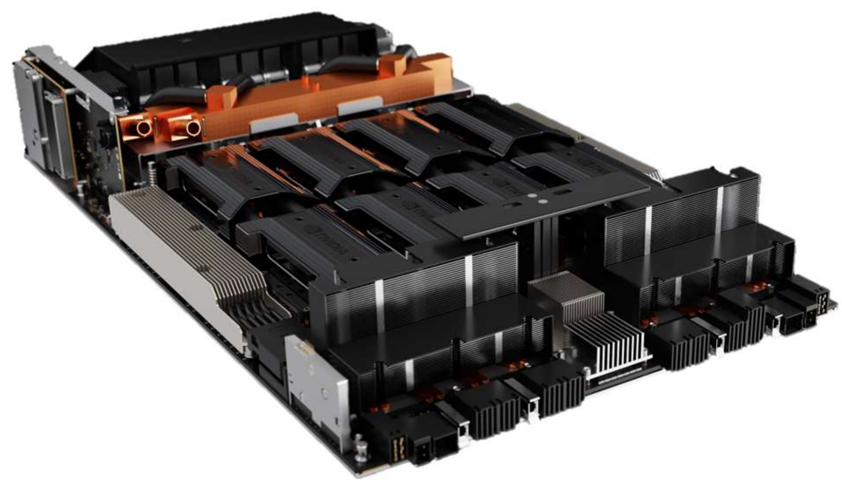
HIGHEST UTILIZATION EFFICIENCY AND SECURITY

7 Fully isolated & secured instances, guaranteed QoS
2nd Gen MIG | Confidential Computing

FASTEST, SCALABLE INTERCONNECT

900 GB/s GPU-2-GPU connectivity (1.5X)
up to 256 GPUs with NVLink Switch | 128GB/s PCI Gen5





FP8, FP16, TF32 performance include sparsity. X-factor compared to A100

NVIDIA H100 NVL

Supercharge Real-Time Large Language Model Inference

Mainstream

Deploy Everywhere

PCIe-Based

GPT3-175B Inference

12X

More Throughput vs HGX A100

NVLinked HBM3

188GB

Super GPU

	H100 PCIe	H100 NVL	
FP16 Tensor Core	1,513 TFLOPS*	3,958 TFLOPS*	2.6X
FP8 Tensor Core	3,026 TFLOPS*	7,916 TFLOPS*	2.6X
GPU Memory	80GB HBM2e	188GB HBM3	2.4X
GPU Memory Bandwidth	2TB/s	7.6TB/s	3.8X
Interconnect	NVLink 600GB/s PCIe Gen5 128GB/s	NVLink Bridge 600GB/s PCIe Gen5 128GB/s	

LLM Inference: GPT3-175B 700 ms | x8 H100 NVL FP8 | HGX A100 FP16 | Iso-power 20MW Data Center.

H100 IN VOLUME SERVERS

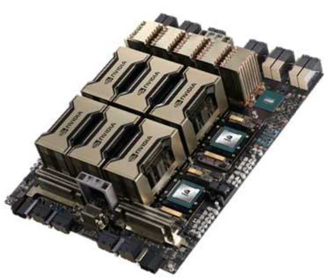
SPECIALIZED GPU SERVERS
ULTIMATE COMPUTE

Ultimate Performance and Scaling

Fastest Time to Solution

Multi-GPU, Multi Node Scaling

Supercomputing HPC+AI



HGX H100 8-GPU

HGX H100 4-GPU

4 -16x H100 PCIE

MAINSTREAM SERVERS
HIGHEST COMPUTE

Wide selection of standard 2U servers

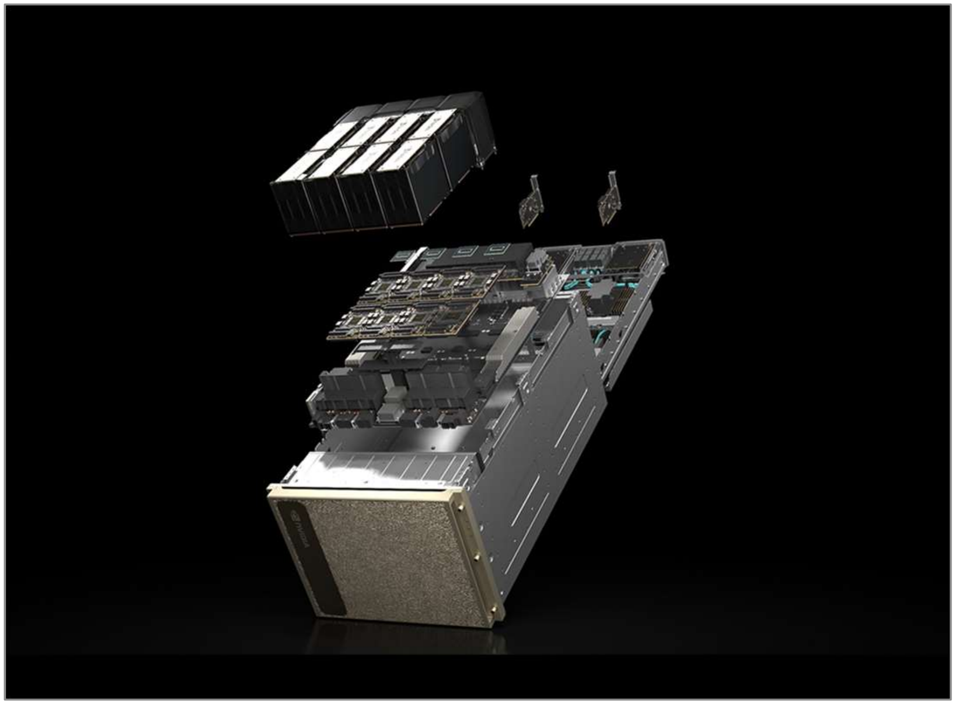
Flexibility, modularity, and ease of deployment

H100 80GB PCIE

H100 NVL

NVIDIA DGX H100: The Proven Choice for Enterprise AI

The gold standard for AI infrastructure



NVIDIA DGX H100

The world's first AI system with the NVIDIA H100 Tensor Core GPU

- 8x NVIDIA H100 GPUs With 640 Gigabytes of Total GPU Memory
  - 18x NVIDIA NVLink connections per GPU, 900 gigabytes per second of bidirectional GPU-to-GPU bandwidth
  - 24 TB/s memory bandwidth
- 4x NVIDIA NVSwitches
  - 7.2 terabytes per second of bidirectional GPU-to-GPU bandwidth, 1.5X more than previous generation
- 10x NVIDIA ConnectX-7 400 Gigabits-Per-Second Network Interface
  - 1 terabyte per second of peak bidirectional network bandwidth
- Dual 56-core 4th Gen Intel® Xeon® Scalable Processors and 2 TB System Memory
  - Powerful CPUs and massive system memory for the most intensive AI jobs
- 30 Terabytes NVMe SSD
  - High speed storage for maximum performance
- 32 petaFLOPS AI performance

Introducing NVIDIA L40S

Unparalleled AI and Graphics Performance for the Data Center

New Ada Architecture Features
- New Streaming Multiprocessor
- 4th-Gen Tensor Cores
- 3rd-Gen RT Cores
- 91.6 teraFLOPS FP32

Gen-AI, LLM Training, & Inference
- Transformer Engine - FP8
- 1.5 petaFLOPS Tensor Performance*
- Large L2 Cache

3D Graphics & Rendering
- 212 teraFLOPS RT Core Performance
- DLSS 3.0, AI Frame Generation
- Shader Execution Reordering

Media Acceleration
- 3 Encode & Decode Engines
- 4 JPEG Decoders
- AV1 Encode & Decode Support

*Peak teraFLOPS, sparsity enabled

NVIDIA L40S - Fine Tune in Hours, Train Small Models in Days

Reserve HGX H100 Capacity for Large Scale Foundational Model Training

Fine Tuning Time to Train

Model Parameters	# of GPUs	L40S	HGX H100
Llama 2-7B SFT (1B tokens)	8	5.5 hours	1.9 hours
Llama 2-13B SFT (1B tokens)	16	5.2 hours	1.8 hours
Llama-33B SFT (1B tokens)	32	6 hours	2 hours
Llama 2-70B SFT (1B tokens)	64	8.2 hours	2.3 hours
GPT 3-175B SFT (1B tokens)	128	9.3 hours	2.5 hours

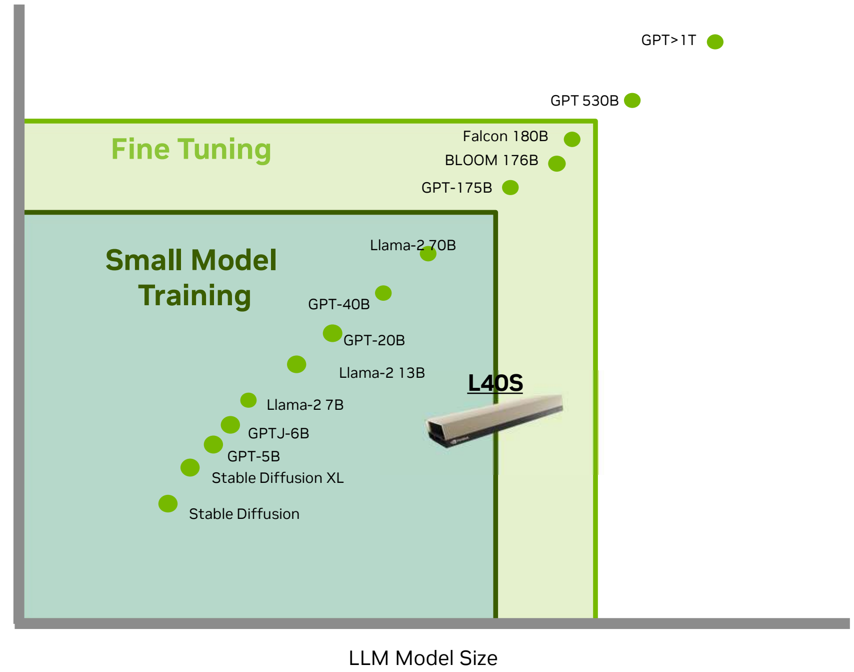
Small Model Training Time to Train

Model Parameters	# of GPUs	L40S	HGX H100
Llama 2-7B (100B tokens)	64	2.9 days	1 day
Llama 2-13B (100B tokens)	128	2.6 days	1 day
Llama 2-70B (1T tokens)	1024	19.8 days	6 days

Foundation Model Training Time to Train

Model Parameters	# of GPUs	L40S	HGX H100
GPT 3-530B (10T tokens)	16,000	-	28 days

Popular LLM Models



L40S system: PCIE 2-4-3 or 2-8-5 GPU system with 200Gbps IB NIC / GPU
H100 system: HGX H100 8-GPU with 400Gbps IB NIC/ GPU

Preliminary performance projections, subject to change
1. Fine Tuning Llama2-7B/13B/33B/70B SFT GBS=64/128/128/128, SL=4096, FP8.
2. Fine-Tuning GPT-175B SFT; GBS=128, SL=4096, FP8.
3. Small LLM Training Llama 2-7B/13B/70B, GBS=512/512/2048, SL=4096, FP8.

NVIDIA L40

Accelerated graphics, AI, and compute performance

Specifications

- Up to 90.5 TFLOPs Single Precision (FP32) Performance
- Up to 724 TFLOPs Tensor Operation Performance*
- Up to 209 TFLOPs Rendering Performance
- 48GB GDDR6 GPU Memory with ECC
- 4 DisplayPort 1.4 Display Outputs
- 3 Encode / 3 Decode Engines
  - Including AV1 Encode & Decode
  - 4 JPEG Decode Engines
- 300W, Dual Slot, FHFL

Data Center Ready

- NVIDIA vGPU Support
- Secure Boot with Root of Trust
- NEBS Level 3 Ready
- Passive Cooling
- In and Out of Band Management
- Lifetime 24/7 Reliability

* Using FP8 data format with structural sparsity enabled.

NVIDIA

NVIDIA L4

Universal Accelerator for Efficient Video, AI, and Graphics

AI Video

120X

More Performance

Graphics

4X

Faster Graphics with 3rd Gen RT Cores

Generative AI

2.5X

Better Performance with 4th Generation Tensor Core

Single Slot, Low Profile
Fits Any Server

Measured Performance:
AI Video: 8x L4 vs 2S Intel 8380 CPU server performance comparison : end-to-end video pipeline with CV-CUDA pre-post processing, decode, inference (SegFormer), encode, TRT 8.6 vs CPU only pipeline using OpenCV 4.7
Graphics: Real-time Rendering: NVIDIA Omniverse performance for real-time rendering at 1080p and 4K with DLSS 3
Generative AI: L4 vs T4: image generation performance, 512x512 Stable Diffusion, FP16

Data Center GPU Comparison

	H100			L40S	L40
Design	**Highest Perf AI, Big NLP, HPC, DA**			**Highest Perf Universal**	**Powerful Graphics + AI**
Form Factor	SXM5	x16 PCIe Gen5 2 Slot FHFL 3 NVlink Bridge	X16 PCIe Gen5 Dual 2 Slot FHFL using 3 NVLink Bridges	x16 PCIe Gen4 2 Slot FHFL	x16 PCIe Gen4 2 Slot FHFL
Max Power	700W	350W	2x 400W	350W	300W
FP64 TC | FP32 TFLOPS[2]	67 | 67	51 | 51	134 | 134	NA | 91.6	NA | 90.5
TF32 TC | FP16 TC TFLOPS[2]	989 | 1979	756 | 1513	1979 | 3958	366 | 733	181 | 362
FP8 TC | INT8 TC TFLOPS/TOPS[2]	3958 | 3958	3026 | 3026	7916 | 7916	1466 | 1466	724 | 724
GPU Memory	80GB HBM3	80GB HBM2e	188GB HBM3	48GB GDDR6	
Multi-Instance GPU (MIG)	Up to 7		UP to 14	-	
Media Acceleration	7 JPEG Decoder 7 Video Decoder		14 JPED Decoder 14 Video Decoder	3 Video Encoder 3 Video Decoder 4 JPEG Decoder	
Ray Tracing	-		-	Yes	
Transformer Engine	Yes		Yes	Yes	
DPX Instructions	Yes		Yes	-	
Graphics	For in-situ visualization (no NVIDIA vPC or RTX vWS)			Top-of-Line	
vGPU	Yes			Yes	
Hardware Root of Trust	Internal and External			Internal	
Confidential Computing	Yes			-	
NVIDIA AI Enterprise	Add-on	Included	Add-on	Add-on	

Other GPUs

NVIDIA RTX in Every Form Factor

Solutions to Do Your Best Work Anywhere

Desktop

RTX 6000 Ada Generation (48GB)
RTX 5000 Ada Generation (32GB)
RTX 4500 Ada Generation (24GB)
RTX 4000 Ada Generation (20GB)
RTX 4000 SFF Ada Generation (20GB)

Laptop

RTX 5000 Ada Laptop GPU (16GB)
RTX 4000 Ada Laptop GPU (12GB)
RTX 3500 Ada Laptop GPU (12GB)
RTX 3000 Ada Laptop GPU (8GB)
RTX 2000 Ada Laptop GPU (8GB)

Data Center

NVIDIA L40S (48GB)
NVIDIA L40 (48GB)
NVIDIA L4 (24GB)

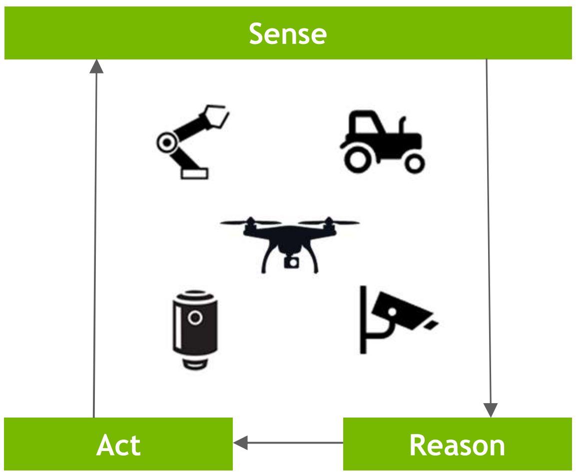
NVIDIA

NVIDIA JETSON

Software-Defined AI Platform

AI at the Edge

Sensor Fusion & Compute Performance

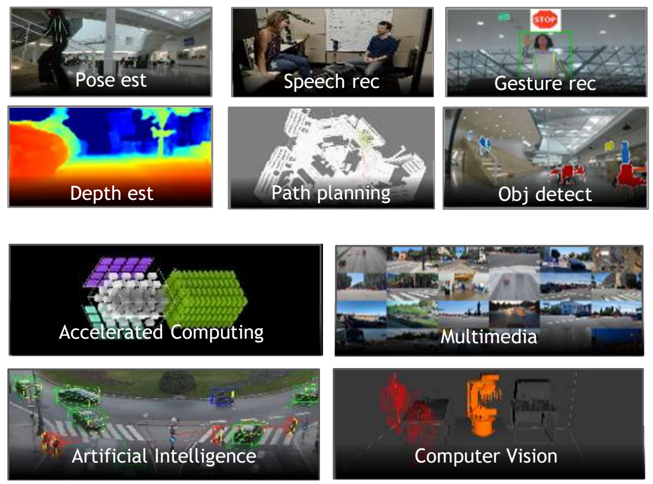


JETSON COMPUTER

Autonomous Machines: The Future of AI | NVIDIA

SOFTWARE DEFINED

SDK, Design Tools, Libs, GEMs



Jetpack SDK · CUDA · TensorRT · Triton · ONNX · ROS

Jetson Software | NVIDIA Developer

ECOSYSTEM

Expertise, Time to Market

Jetson Ecosystem | NVIDIA Developer

H200

Announcing NVIDIA HGX H200

The World’s Leading AI Computing Platform

Highest Performance for AI and HPC

8-way or 4-way H200 GPUs

Up to 32 PetaFLOPs FP8

Up to 1.1TB High Bandwidth Memory

Fastest, Scalable Interconnect

4th Gen NVLINK with 2X faster All-Reduce communications

3.6 TB/s bisection bandwidth

Fully Compatible with Partner H100 Systems

Supported by Leading Major OEMs and CSPs

Coming to Leading OEM and CSP Partners

Starting Q2 2024

ASRock Rack, ASUS, aws, CoreWeave, DELL Technologies, EVIDEN

GIGABYTE, Google Cloud, Hewlett Packard Enterprise, Lambda, Lenovo, Microsoft Azure

ORACLE CLOUD Infrastructure, QCT, SUPERMICRO, VULTR, wistron, wiwynn

Announcing NVIDIA H200 Tensor Core GPUs

Supercharging the Highest Performing Generative AI and HPC Platforms

Memory
141GB
HBM3e

Memory Bandwidth
4.8 TB/s
HBM3e

Llama 2 70B Inference
1.9X
Performance vs H100

GPT-3 175B Inference
1.4X
Performance vs H100

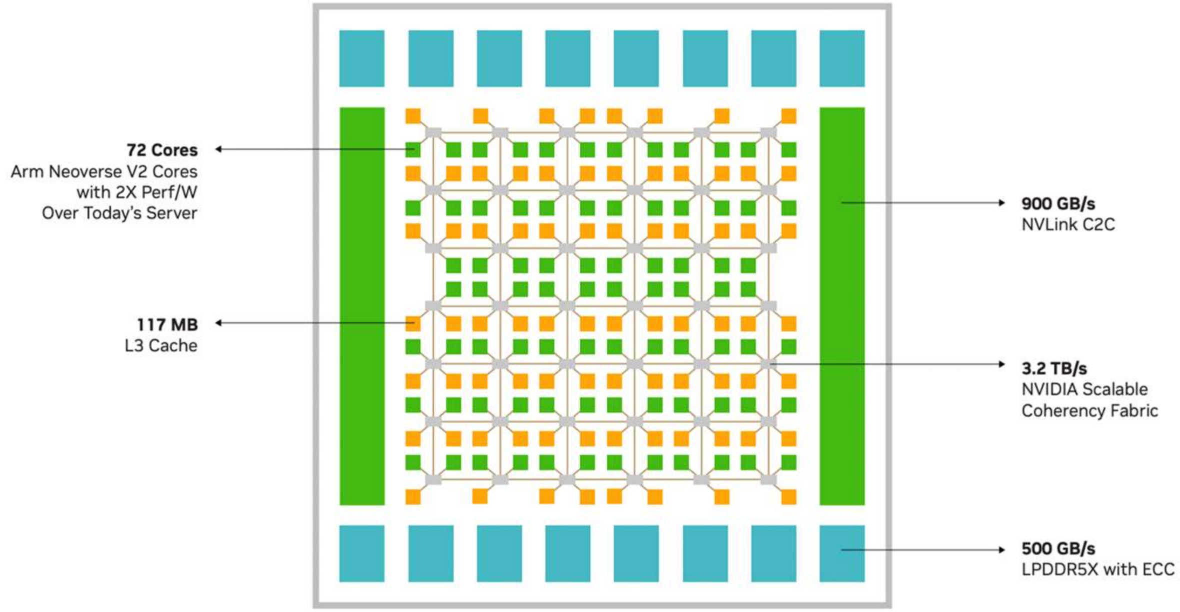
MILC HPC Simulation
110X
Performance vs x86 CPUs

Grace CPU

GRACE IS A COMPUTE & DATA MOVEMENT ARCHITECTURE

NVIDIA Scalable Coherency Fabric and distributed cache design

- 3,225.6 GB/s Bi-section BW
- 117MB of L3 cache
- Scalable to 72+ cores per die
- Local caching of remote die memory
- Supports up to 4-die coherency over Coherent NVLINK
- Background data movement via Cache Switch Network



Example possible fabric topology for illustrative purposes

GRACE HOPPER SUPERCHIP

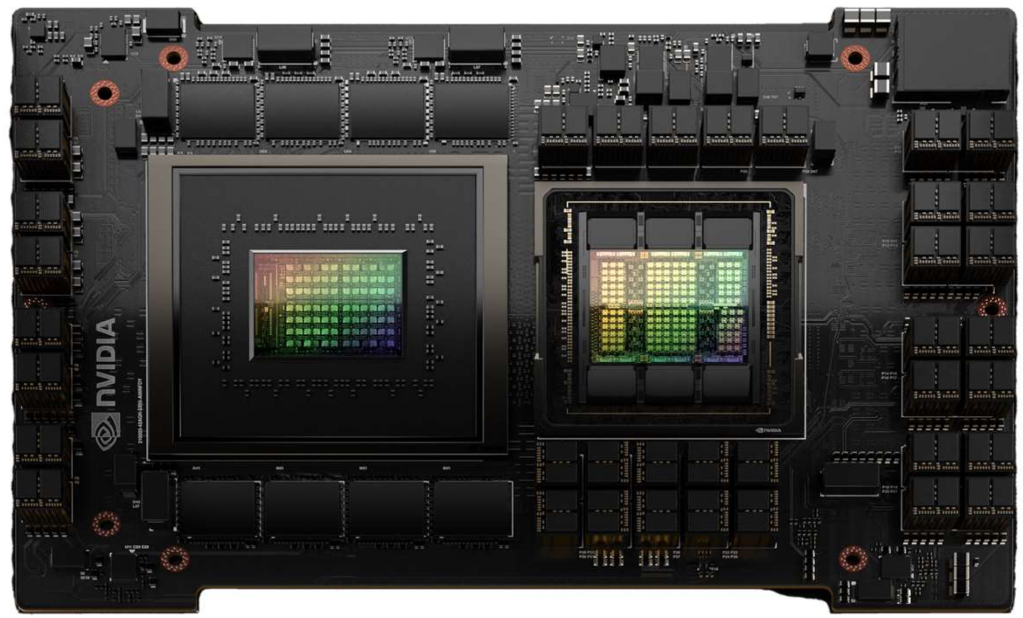
CPU+GPU Designed for Giant Scale AI and HPC

600GB Memory GPU for Giant Models

New 900 GB/s Coherent Interface

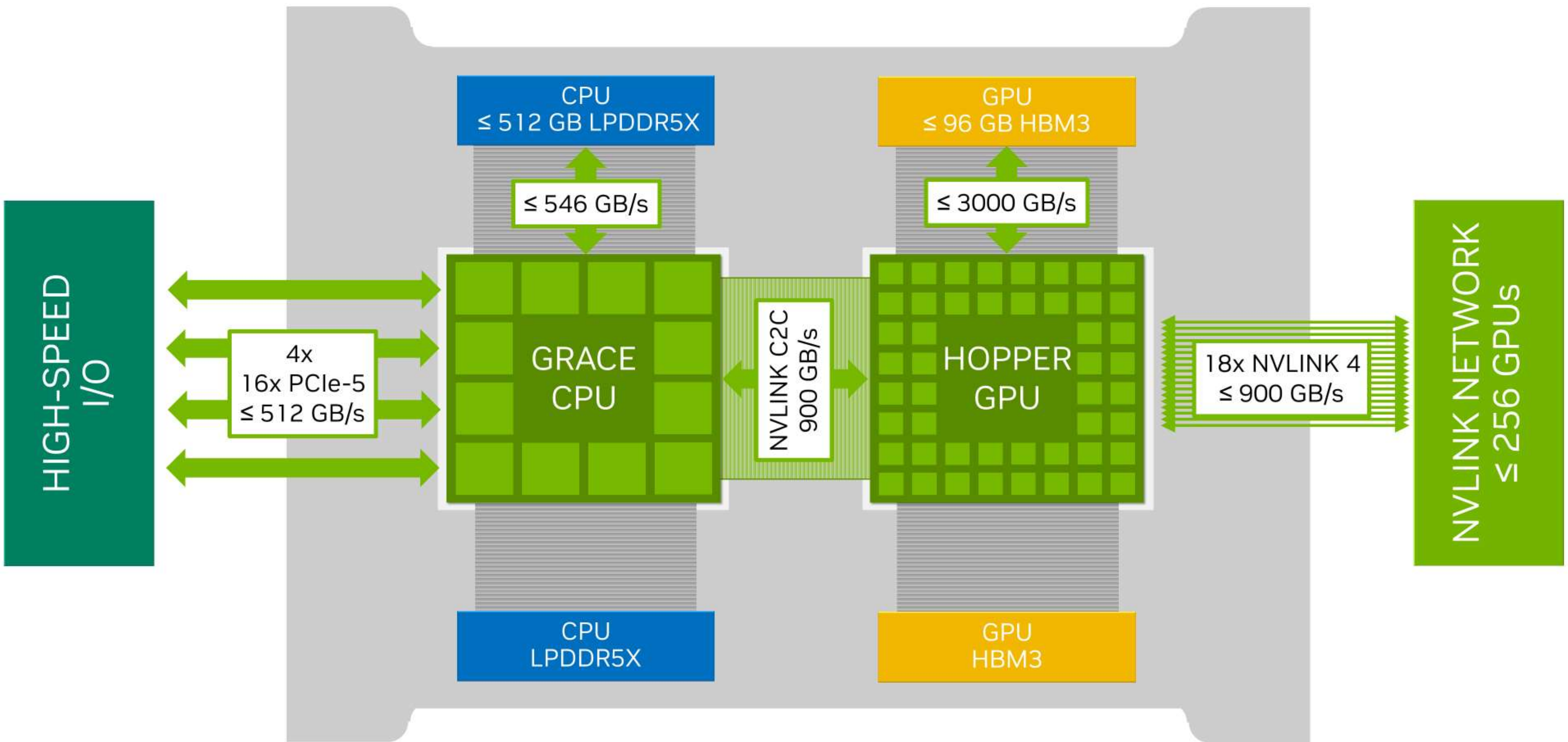
30X Higher System Memory B/W to GPU In A Server

Runs Nvidia Computing Stacks



Grace Hopper Superchip Platform

Speeds and Feeds



All standard Linux Memory Management APIs can be used for both CPUs and GPUs

Grace-Hopper Memory Model

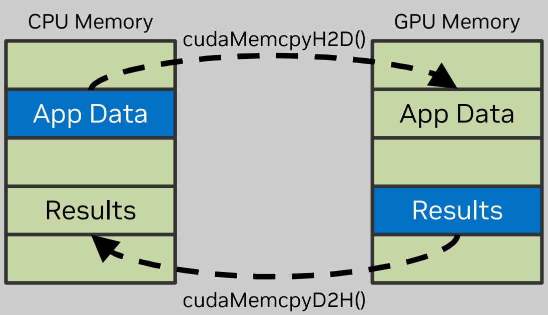
Full CUDA support with additional Grace memory extensions

Explicit Copy

Application explicitly moves data between CPU & GPU as needed

PCIE: ~60 GB/s PCIE transfers (H2D/D2H)

Grace: Faster transfers; up to 450 GB/s C2C transfers

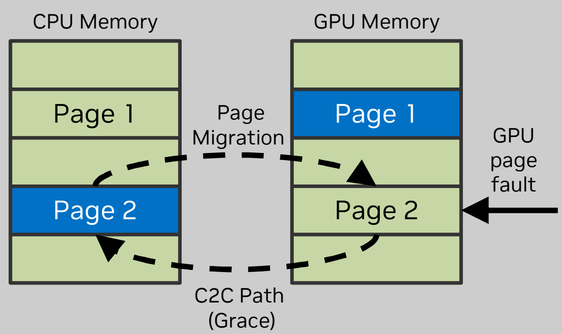


Managed Memory

CPU and GPU can access memory on-demand and data migrated locally for higher BW access

PCIE: Requires migration to GPU

Grace: Migrations not required and faster migrations when they happen

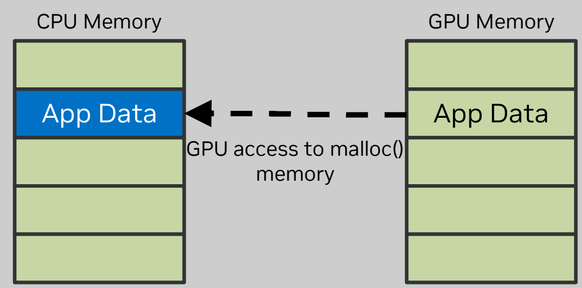


System Allocated

GPU can access memory allocated from malloc(), mmap(), etc.

PCIE: Access possible with explicit call to cudaHostRegister() at PCIe speeds

Grace: cudaHostRegister() not needed; access at NVLink C2C speeds



NVIDIA Grace CPU Superchip

2X Performance at the Same Power for the Modern Data Center

High Performance Power Efficient Cores

144 flagship Arm Neoverse V2 Cores with
SVE2 4x128b SIMD per core

Fast On-Chip Fabric

3.2 TB/s of bi-section bandwidth connects
CPU cores, NVLink-C2C, memory, and system IO

High-Bandwidth Low-Power Memory

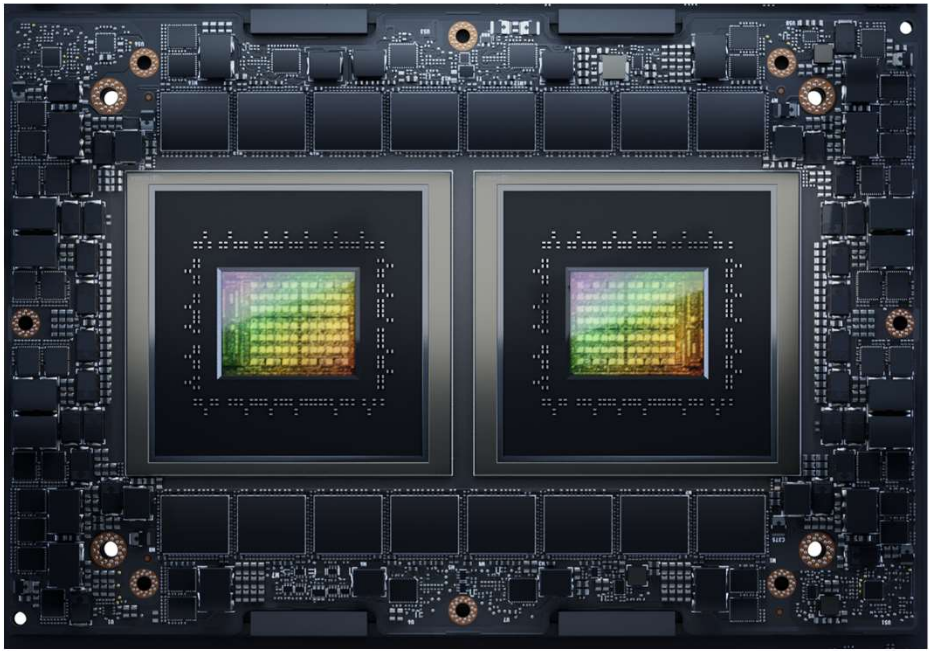
Up to 960GB of data center enhanced LPDDR5X Memory that
delivers up to1TB/s of memory bandwidth

Fast and Flexible CPU IO

Up to 8x PCIe Gen5 x16 interface. PCIe Gen 5 up to 128GB/s
2X more bandwidth compared to PCIe Gen 4

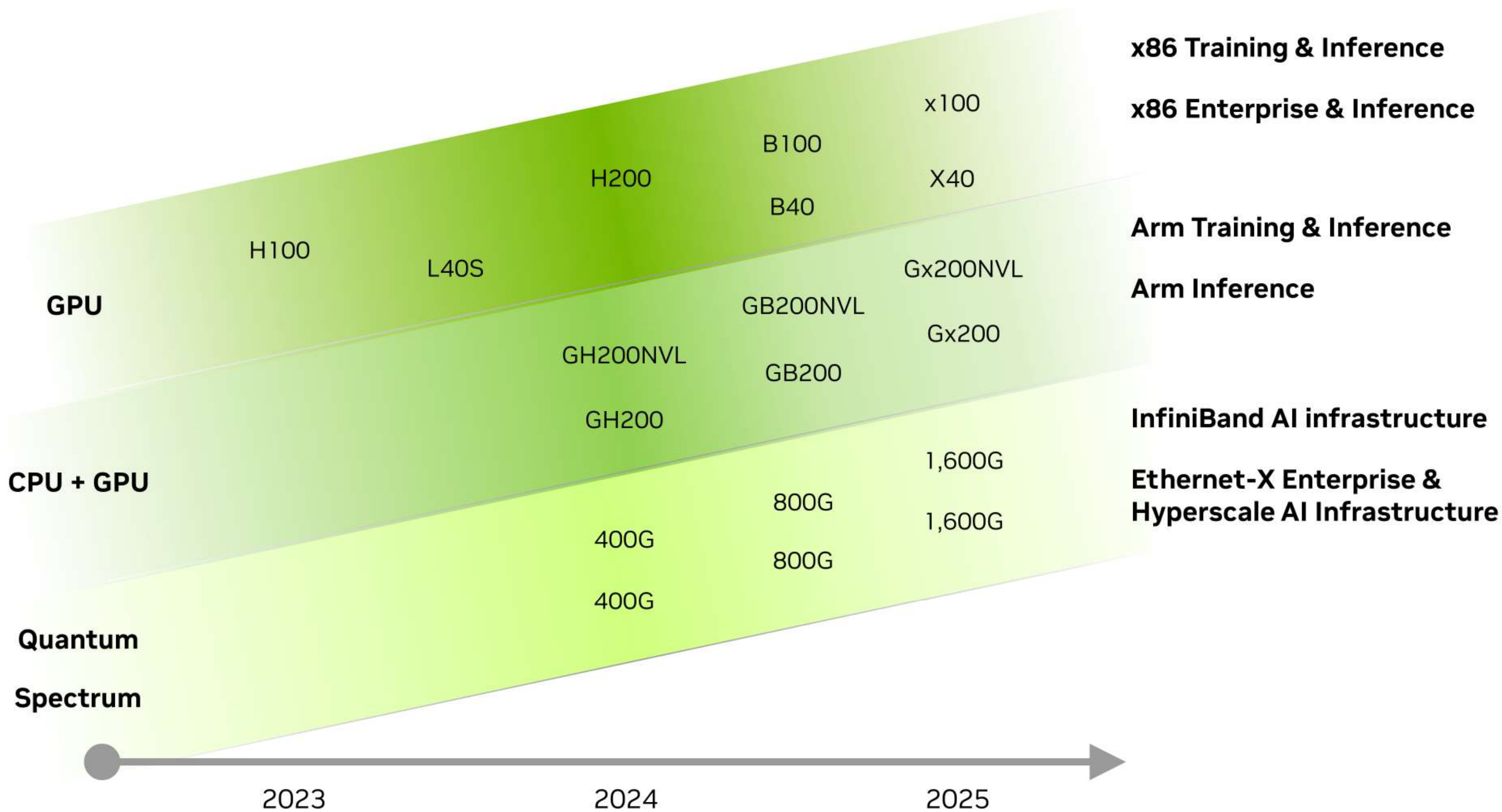
Full NVIDIA Software Stack

AI, Omniverse



NVIDIA AI - One Architecture | Train and Deploy Everywhere

One –Year Rhythm



NVIDIA

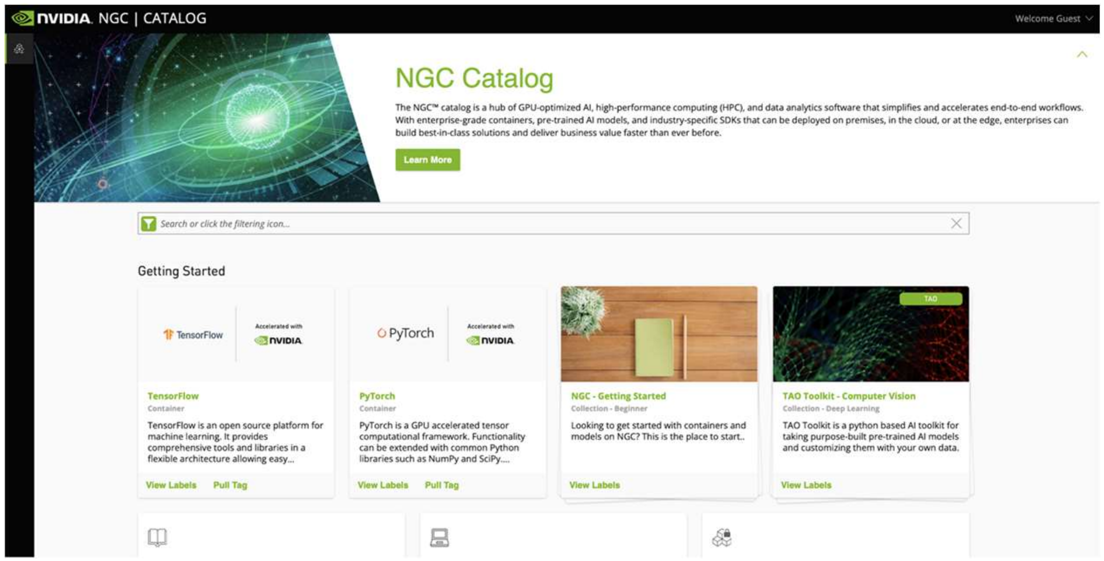
Best Practices for Best Performance

BUILD FASTER WITH NVIDIA CONTAINERS

https://ngc.nvidia.com

219 Containers

690 Models



PERFORMANCE OPTIMIZED

Scalable

Updated Monthly

Better performance on the same system

DEPLOY ANYWHERE

Docker | cri-o | containerd | Singularity

Bare metal, VMs, Kubernetes

Multi-cloud, on-prem, hybrid, edge

ENTERPRISE READY SOFTWARE

Container scanning reports for CVEs, malware

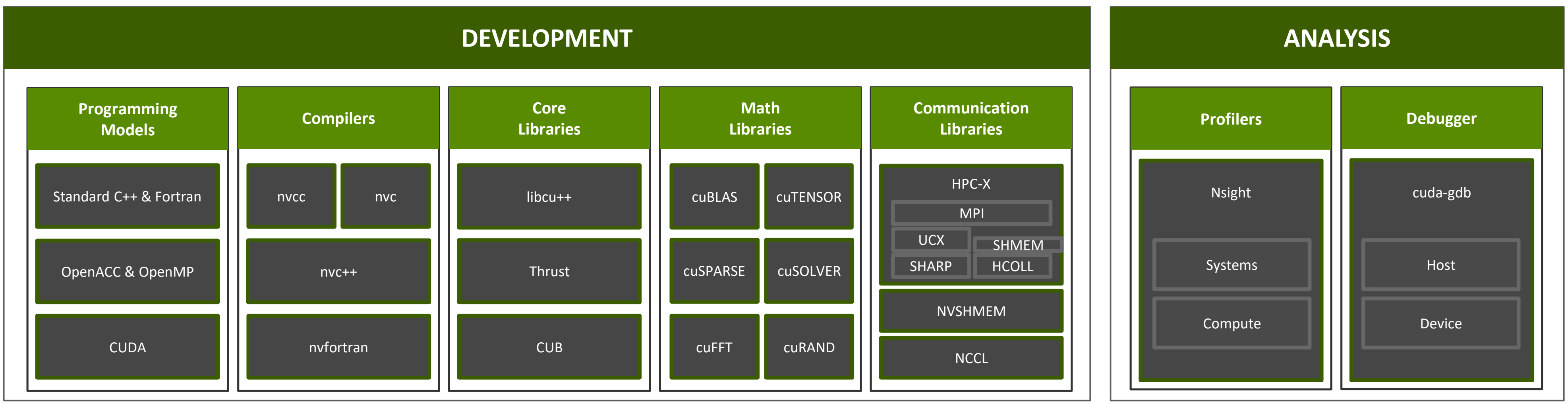
Tested for reliability

Backed by Enterprise support

NVIDIA HPC SDK

Available at developer.nvidia.com/hpc-sdk, on NGC, via Spack, and in the Cloud

Performance   Portability   Productivity



Develop for the NVIDIA Platform: GPU, CPU and Interconnect
Libraries | Accelerated C++ and Fortran | Directives | CUDA
X86_64 | Arm | OpenPOWER
7-8 Releases Per Year | Freely Available

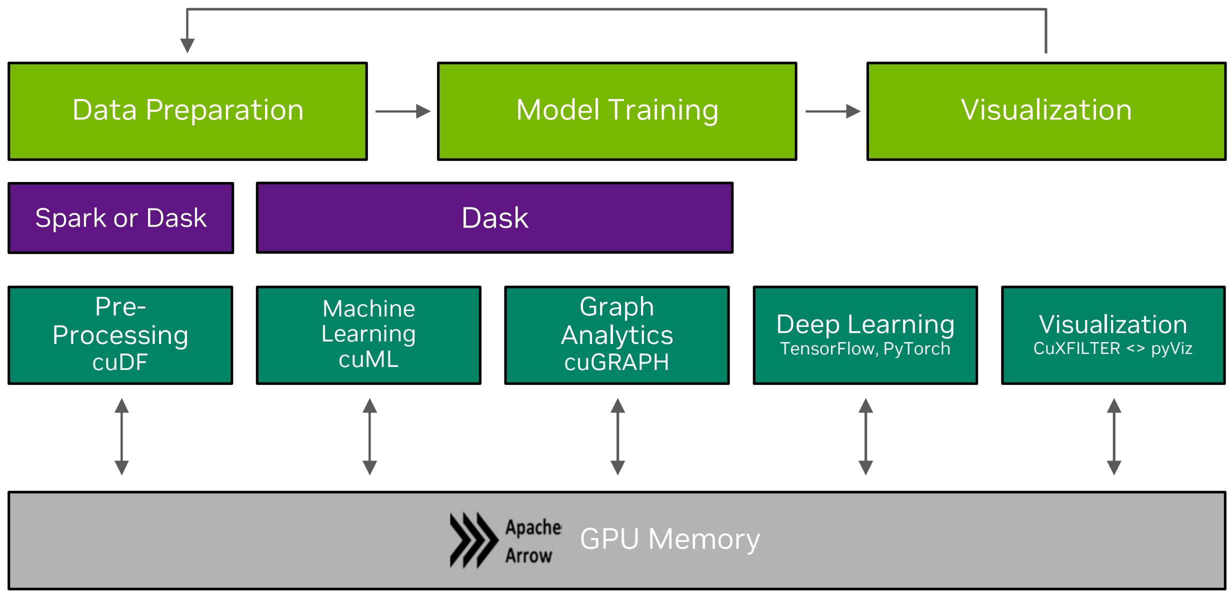
RAPIDS Accelerates Popular Data Science Tools

Delivering enterprise-grade data science solutions

The RAPIDS suite of open-source software libraries gives you the freedom to execute end-to-end data science and analytics pipelines entirely on GPUs.

RAPIDS utilizes **NVIDIA CUDA** primitives for low-level compute optimization and exposes GPU parallelism and high-bandwidth memory speed through user-friendly interfaces like Apache Spark or Dask.

With Spark or Dask, RAPIDS can scale out to multi-node, multi-GPU cluster to power through big data processes.

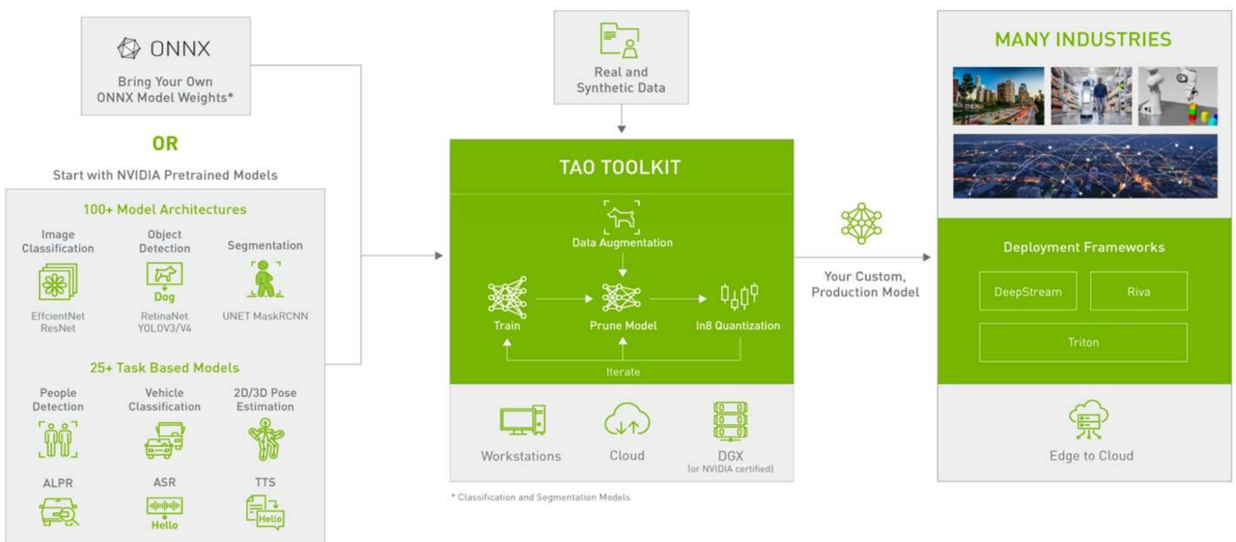


RAPIDS puts the power of GPUs in the hands of all Data Scientists

NVIDIA TAO Toolkit

Create custom, production-ready AI models in hours rather than months

1. Bring your own model weights or choose from NVIDIA's library of model architectures or task-based models
2. Quickly train, adapt, and optimize models with your real or synthetic data
3. Integrate your customized models into your application and deploy



TRAIN EASILY

Fine tune NVIDIA pretrained models with fraction of the data

CUSTOMIZE FASTER

Built on TensorFlow and PyTorch that abstracts away the AI framework complexity

OPTIMIZE FOR DEPLOYMENT

Optimize for inference and integrate with Riva or DeepStream

SUPPORTED BY EXPERTS*

Supported by NVIDIA experts to help resolve issues from development to deployment

* Requires NVIDIA AI ENTERPRISE SUBSCRIPTION. Learn more here: https://www.nvidia.com/en-us/data-center/products/ai-enterprise/

NVIDIA TensorRT

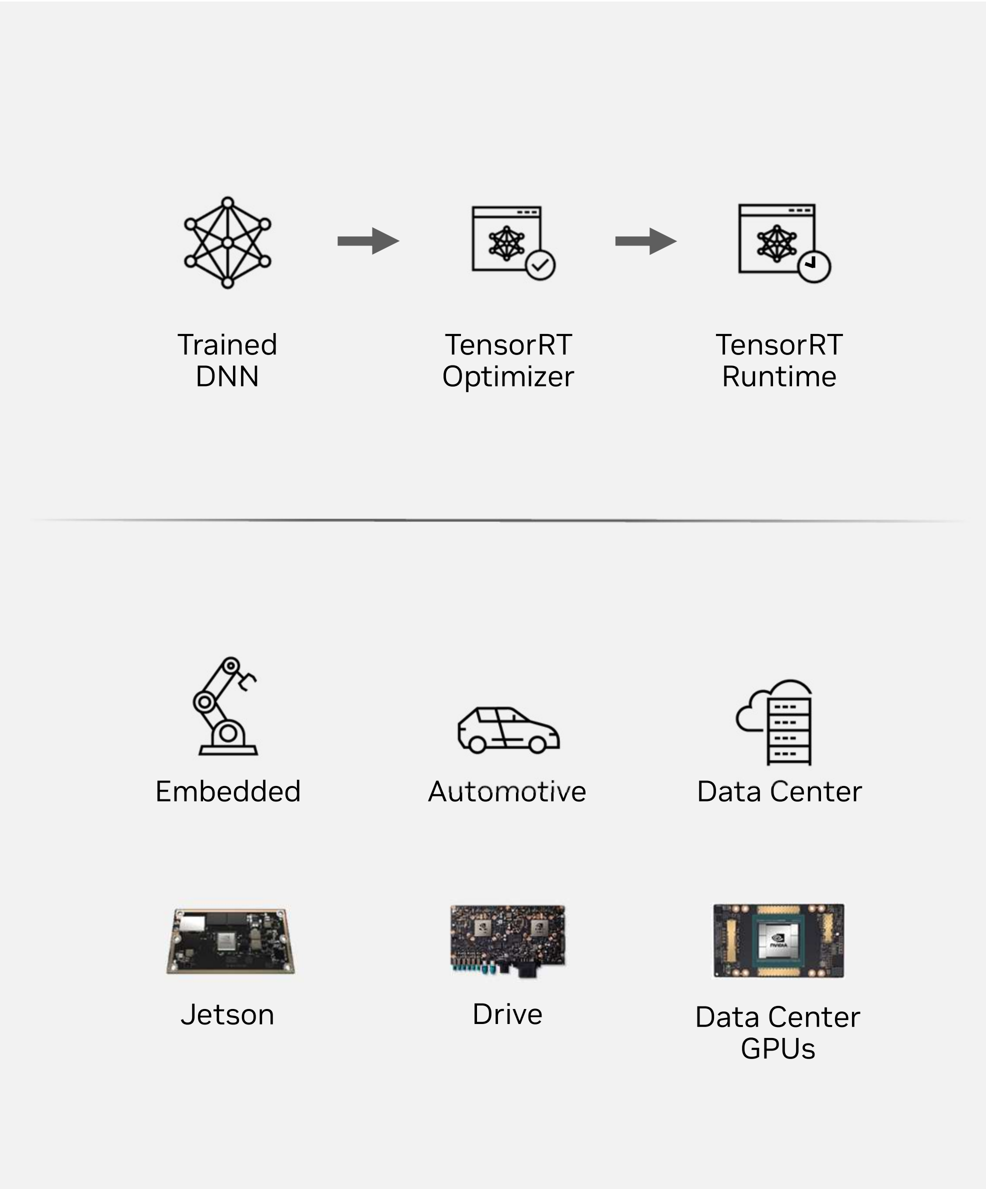
SDK for high-performance deep learning inference

Optimize & deploy all networks, including CNNs, RNNs, and Transformers.

Maximize throughput for latency-critical apps with compiler and runtime.

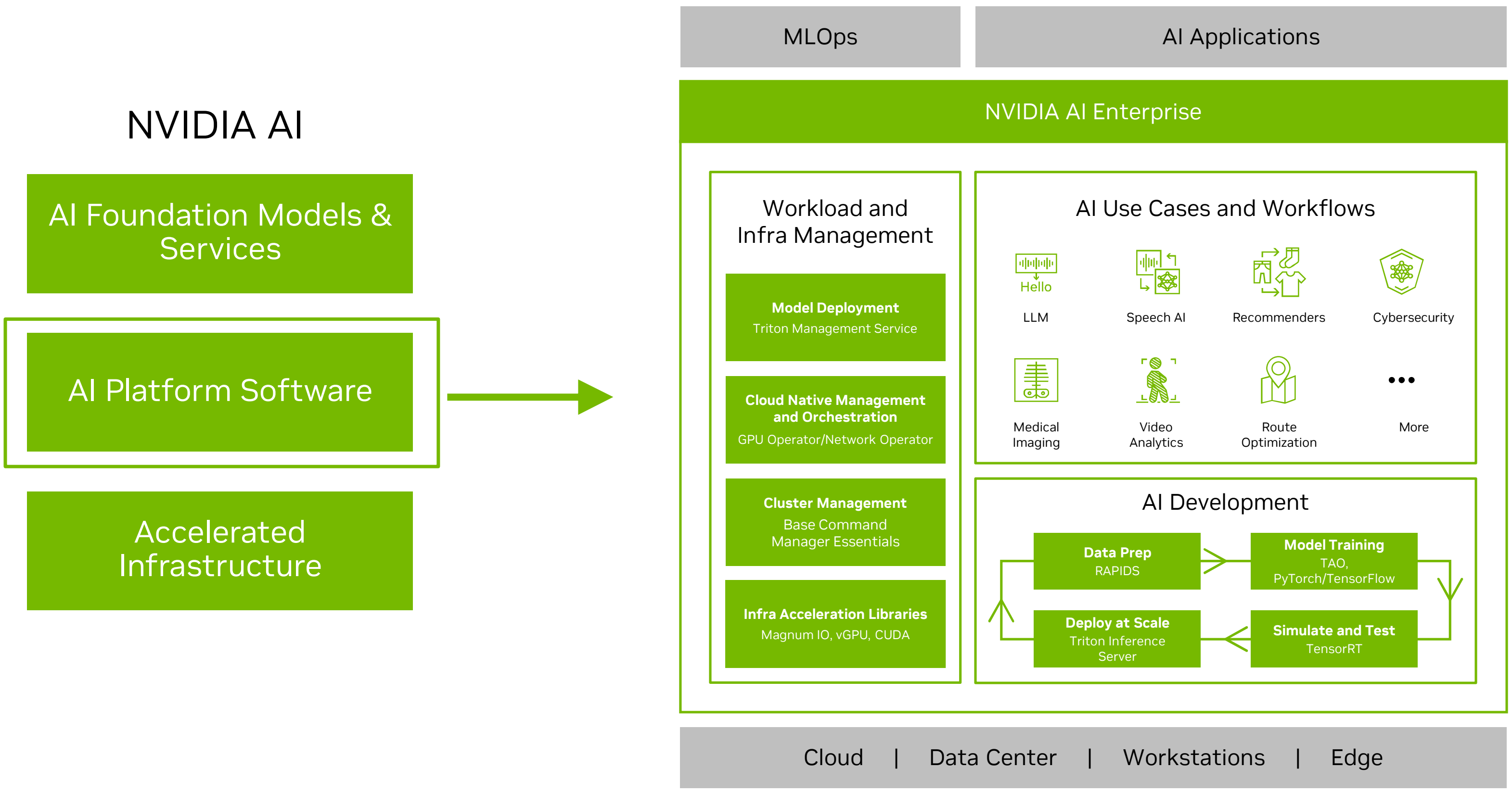
1. Reduced mixed precision: FP32, TF32, FP16, and INT8.
2. Layer and tensor fusion: Optimizes use of GPU memory & bandwidth.
3. Kernel auto-tuning: Select best data layer & algorithm on target GPU.
4. Dynamic tensor memory: Deploy memory-efficient models.
5. Multi-stream execution: Scalable design to process multiple streams.
6. Time fusion: Optimizes RNN over time steps.

https://developer.nvidia.com/tensorrt



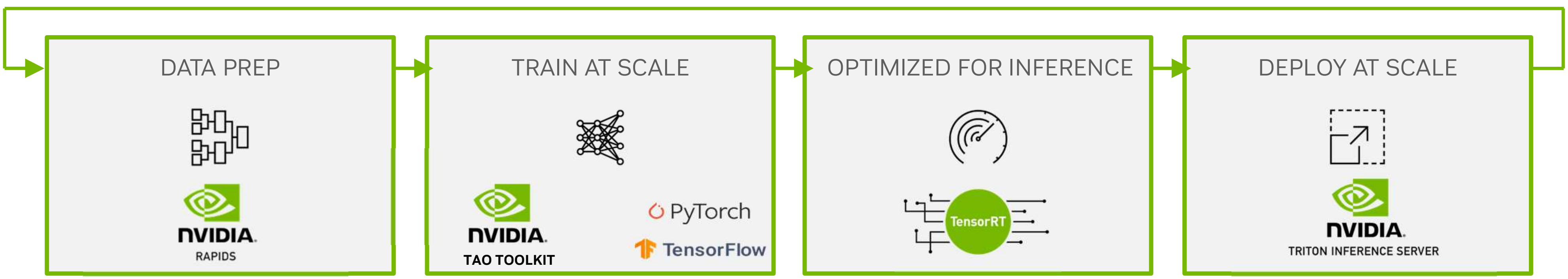
NVIDIA AI Enterprise

End to end AI software



NVIDIA End-to-End AI Software Suite

Deep Learning Streamlined From Conception to Production at Scale



- Reduces data science processes from hours to seconds
- 70x faster performance than similar CPU configuration
- 20x more cost-effective than similar CPU configuration

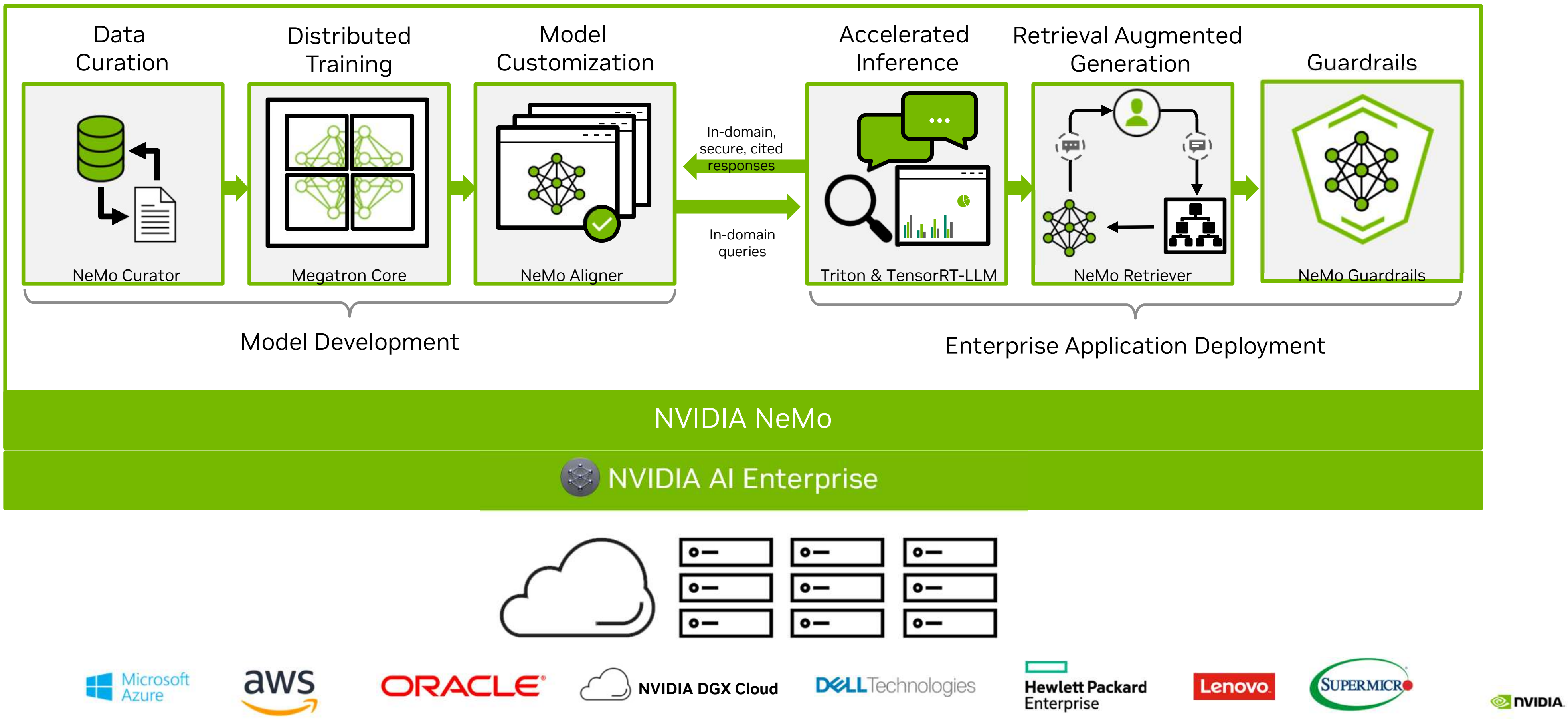
- Train, Adapt, Optimize Models in hours vs. months
- Open-source ML frameworks optimized for GPU
- Integrated w/ NVIDIA RAPIDS to simplify development

- Maximize throughput for latency-critical apps w/ compiler & runtime
- Optimize every network (CNNs, RNNs, & Transformers)
- Optimizes use of GPU memory bandwidth

- Fast & scalable AI to applications
- Diverse query types – real time, offline batch, ensembles
- Up to 266x performance increase over CPU-only
- Triton with FIL backend delivers best inference performance for tree-based models on GPUs

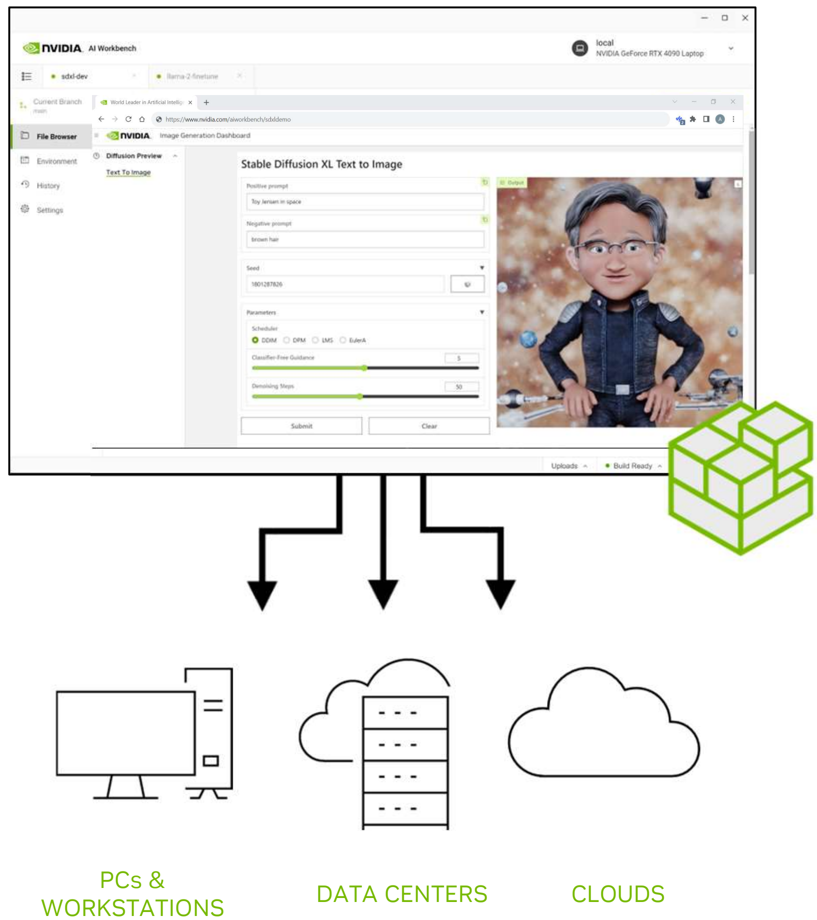
Building Generative AI Applications for the Enterprise

Build, customize and deploy generative AI models with NVIDIA NeMo



NVIDIA AI Workbench

Enables anyone with access to a GPU to be a generative AI creator

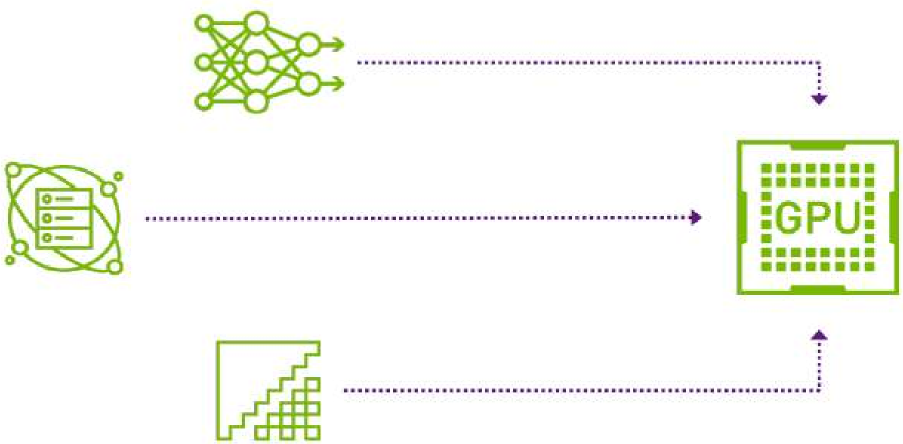


- Create projects for tuning and deployment of generative AI and LLMs
- Move projects between PCs and workstations, data centers, public clouds, and NVIDIA DGX Cloud
- Easily start with pre-built project examples

Multi-Process Service (MPS)

WHY SHARE GPUS?

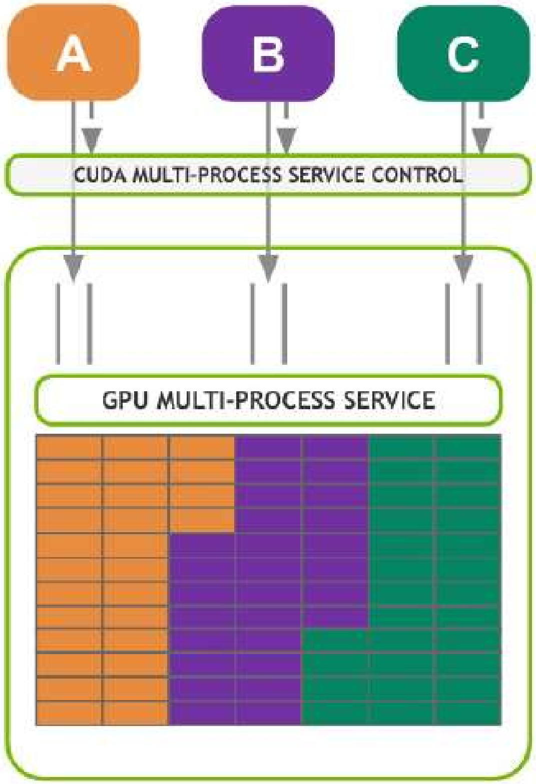
- In deployment environments, individual workloads don't always saturate the GPU's capacity
  - Low-batch inference
  - CI/CD of GPU-based applications
  - Visualization workloads (cloud rendering, CDI, professional workstations)
- Workloads can be bursty *or* generate persistent (but low-level) background effort



CUDA MPS allows multiple processes to share a given GPU instance

Doesn't the GPU do this anyway?

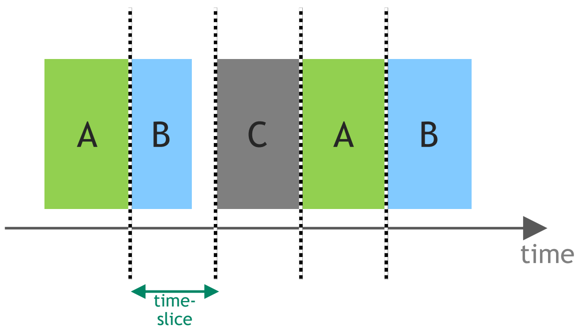
Yes, with Time-Sliced Context Switching



EXECUTION SCHEDULING & MANAGEMENT

Pre-emptive scheduling

Processes share GPU through time-slicing
Scheduling managed by system

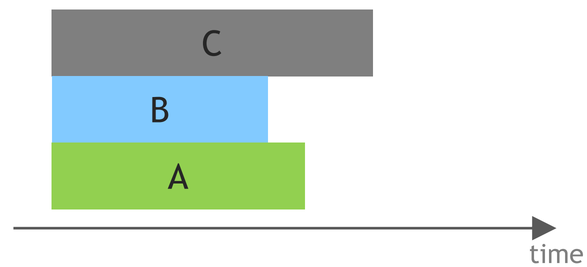



```
$ nvidia-smi compute-policy
  --set-timeslice={default, short, medium,
long}
```

Time-slice configurable via **nvidia-smi**

Concurrent scheduling

Processes run on GPU simultaneously
User creates & manages scheduling streams

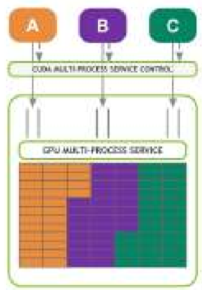



```
cudaStreamCreateWithPriority(pStream, flags, priority);
cudaDeviceGetStreamPriorityRange(leastPriority, greatestPriority);
```

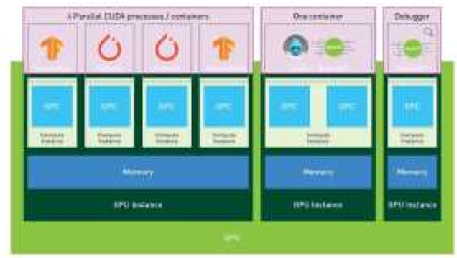
CUDA 11.0 adds a new **stream priority** level

GPU SHARING METHODS

Single-Process Concurrency



Multi-Process Concurrency



Hardware Segmentation

GPU Virtualization

Application Level

CUDA Streams, multi-threading, etc.

Process Level

CUDA Multi-Process Service

Hardware Level

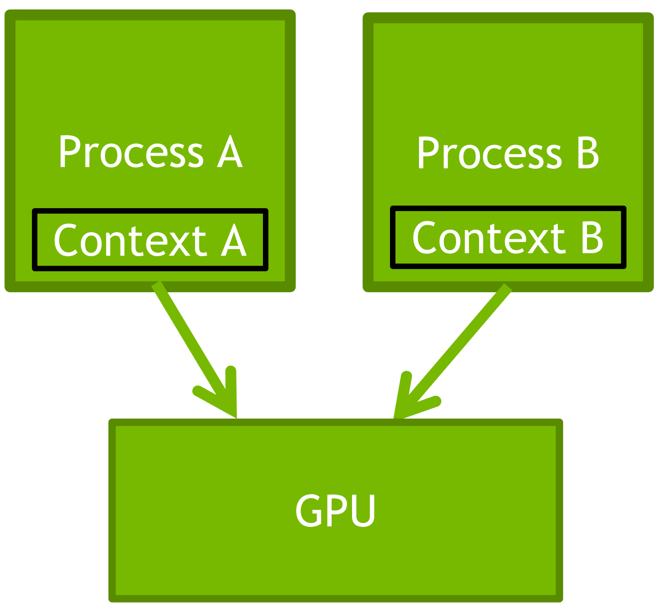
Multi-Instance GPU

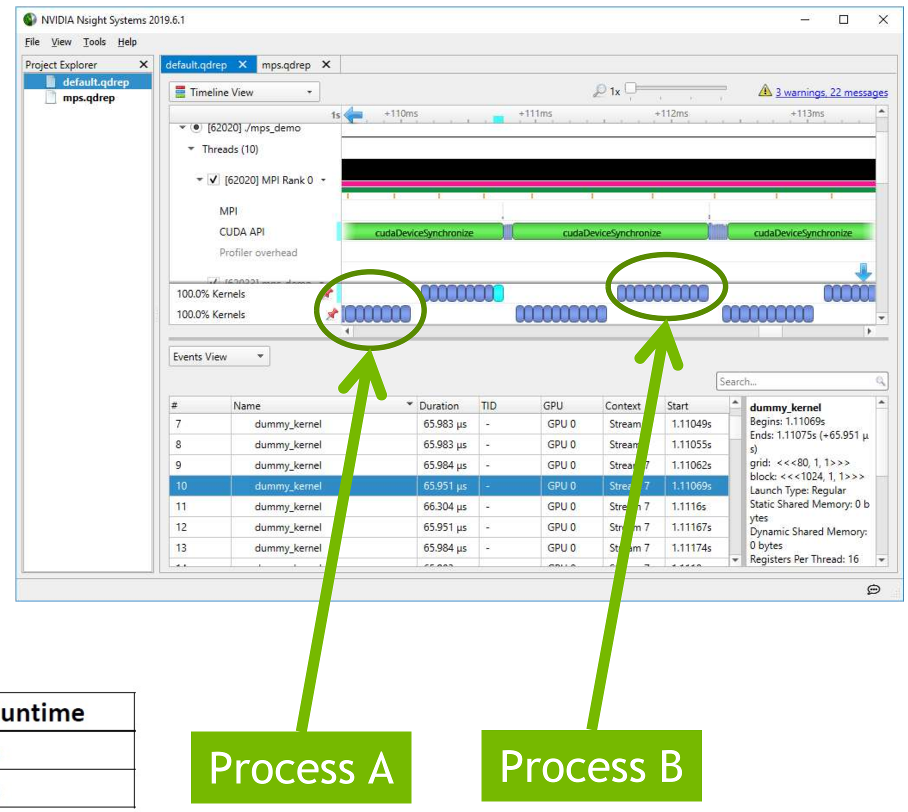
System Level

Virtualization, vGPU

PROCESSES SHARING GPU WITHOUT MPS

No Overlap

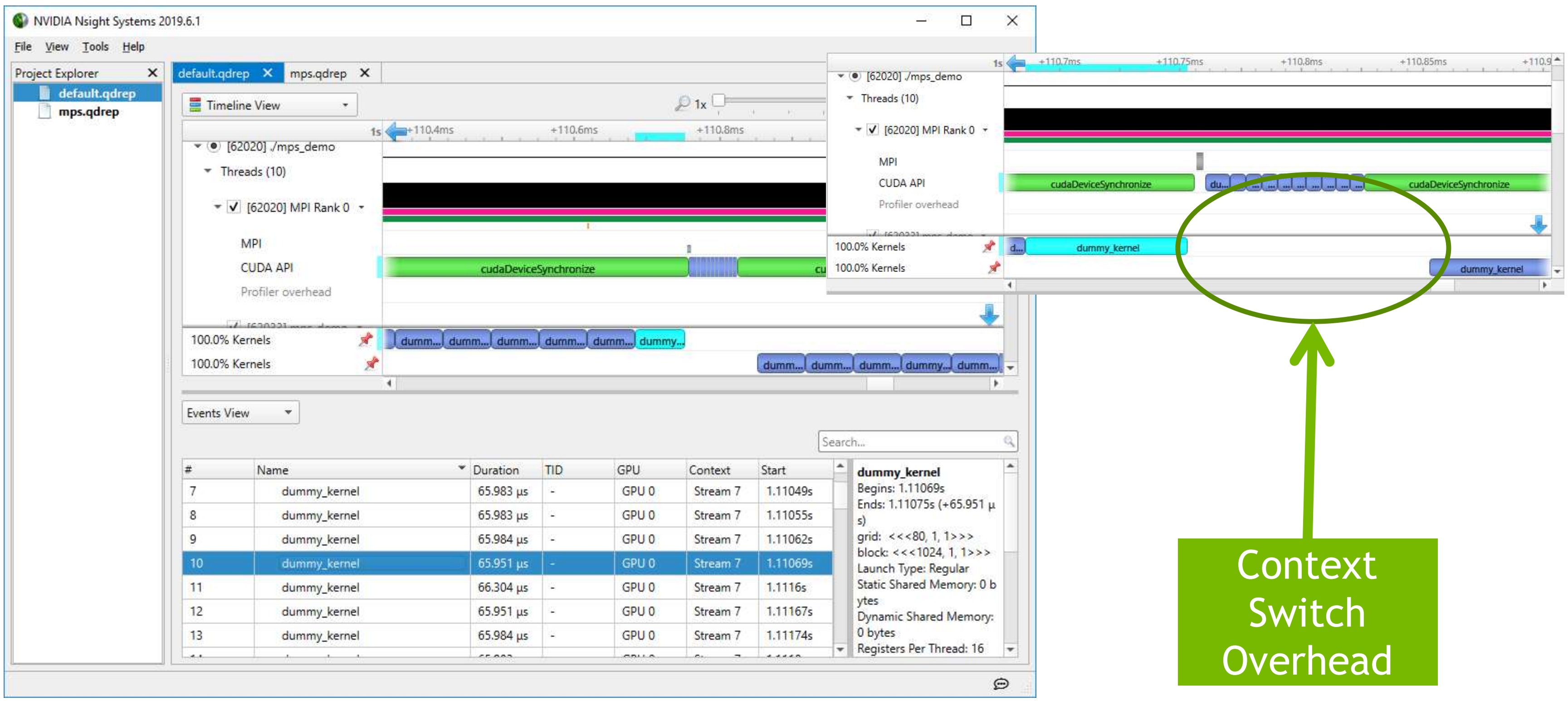




Process	Isolated Runtime	Concurrent Runtime
A	5 ms	10 ms
B	5 ms	10 ms

PROCESSES SHARING GPU WITHOUT MPS

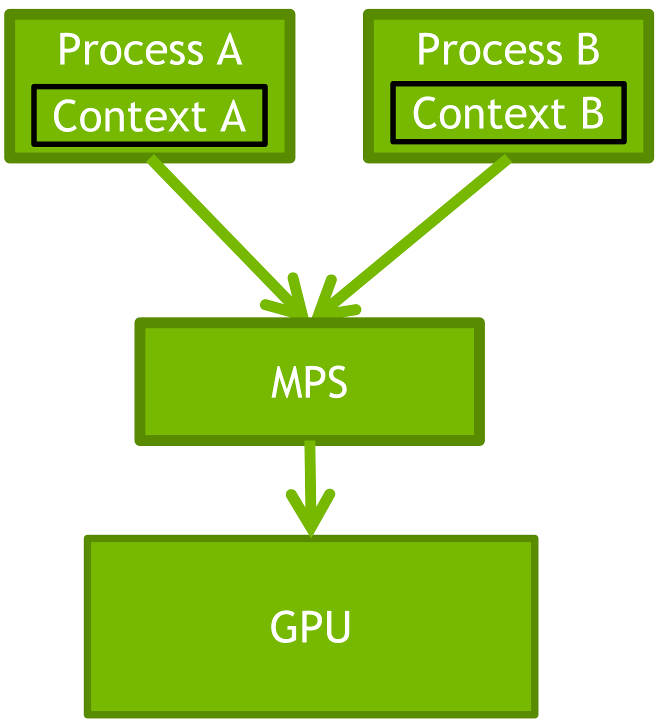
Additional small overhead arising from pre-emptive context switch

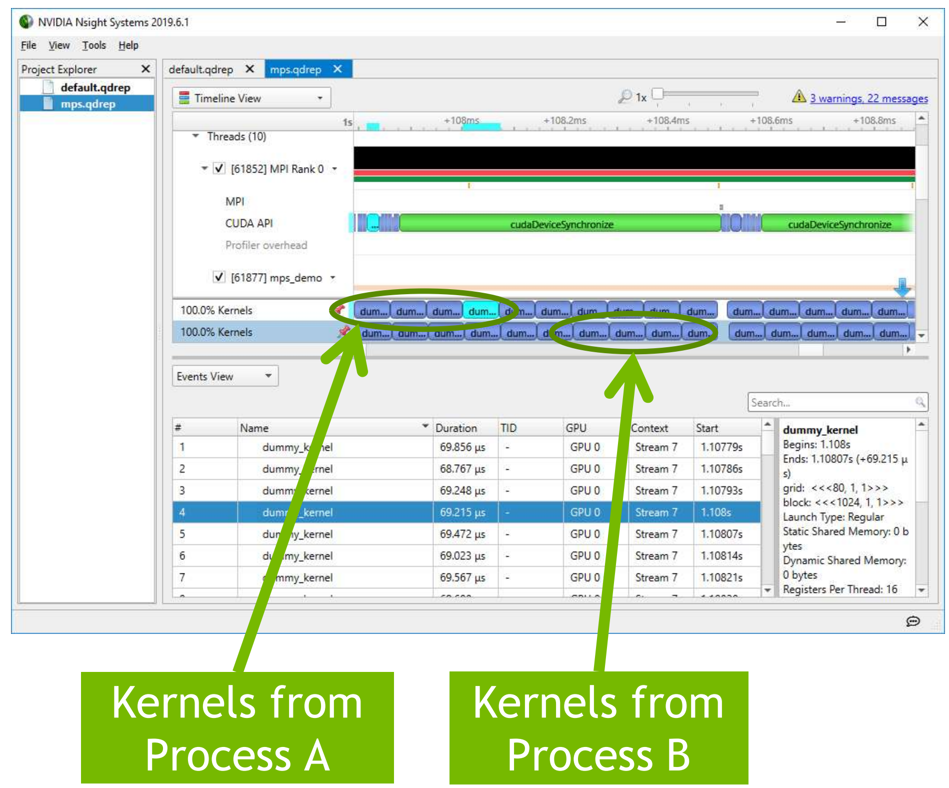


NVIDIA

PROCESSES SHARING GPU WITH MPS

Maximum Overlap

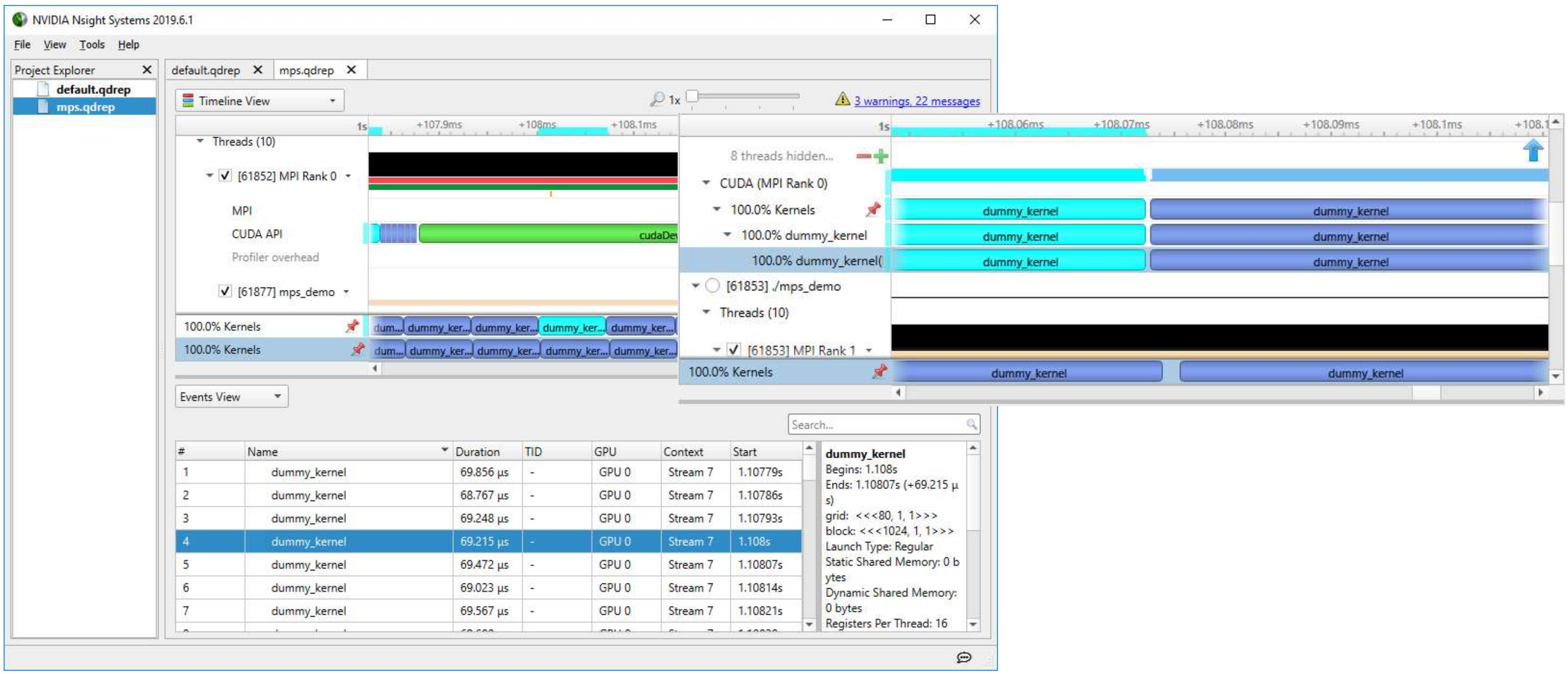




NVIDIA

PROCESSES SHARING GPU WITH MPS

No Context Switch Overhead



USING MPS

No application modifications necessary

Not limited to MPI applications

MPS control daemon spawns MPS server upon CUDA application startup

CUDA tools (debugger & profiler) are MPS-aware

```
#Manually

nvidia-smi -c EXCLUSIVE_PROCESS

nvidia-cuda-mps-control -d
```

Compute modes

- **PROHIBITED** (cannot set device)
- **EXCLUSIVE_PROCESS** (single shared device)
- **DEFAULT** (per-process device)

Recommended to use EXCLUSIVE_PROCESS mode to ensure that only a single MPS server is using the GPU

EXECUTION RESOURCE PROVISIONING WITH MPS

Using MPS, applications can assign fractions of a GPU to each process

$ setenv CUDA_MPS_ACTIVE_THREAD_PERCENTAGE=*percentage*

- Environment variable: configures maximum fraction of a GPU available to an MPS-attached process
- Guarantees a process will use at most *percentage* execution resources (SMs)
- Over-provisioning is permitted: sum across all MPS processes may exceed 100%
- Provisions only **execution** resources (SMs) – does not provision memory bandwidth or capacity
- Before CUDA 11.2, all processes be set to the same percentage
- Since CUDA 11.2, percentage may be different for each process

Full details at: https://docs.nvidia.com/deploy/mps/index.html#topic_5_2_5

GPU PROVISIONING WITH MPS

Using MPS, applications can assign fractions of a GPU to each process

A=33%, B=33%, C=33%

Fractional Provisioning

Process C could use more, but is limited to just 33% of execution resources

Process B is guaranteed space if needed

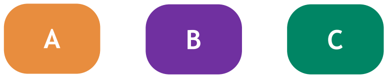
A=33%, B=33%, C=100%

Using Oversubscription

Process B is not using all of its allocation

Process C may grow to fill available space

Additional B work may have to wait for resources



← 3 concurrent MPS processes

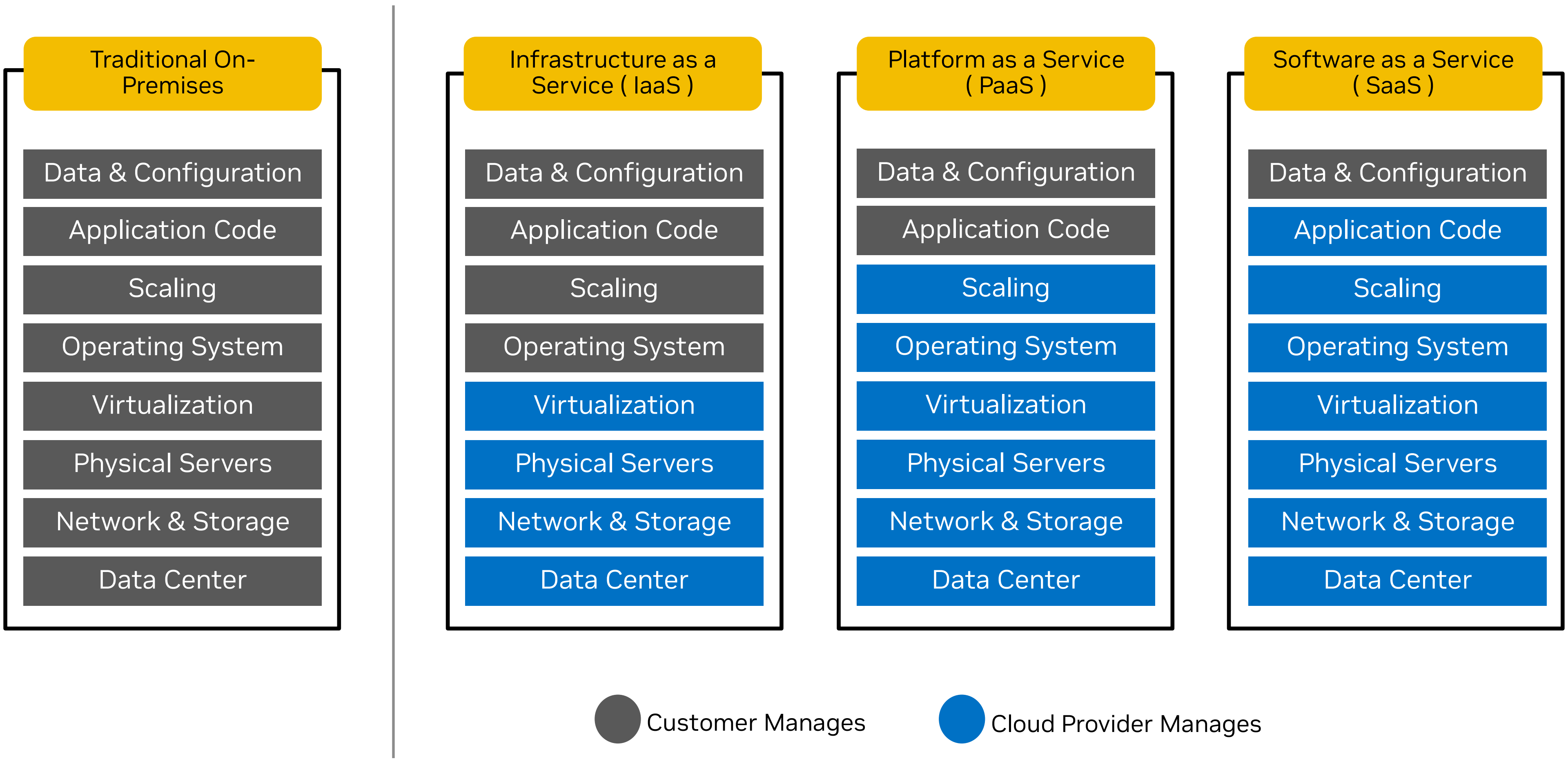
Best Practices for Highest Performance

Summary

- **Use optimized containers from NGC**
- **Use the optimized HPC SDK**
- **Take advantage of Mixed Precision**
- **Exploit sharing with MPS**
- **For multi-GPU workloads, use systems that support NVLink and GPUDirect**

NVIDIA in the Cloud

Cloud Consumption Models



H100 in the Cloud

Access NVIDIA through our Cloud Partners

H100 in the Cloud

See https://cloud-gpus.com/

CoreWeave

Cirrascale CLOUD SERVICES

Lambda

Broad Portfolio of NVIDIA GPUs for AI Workloads in the Cloud

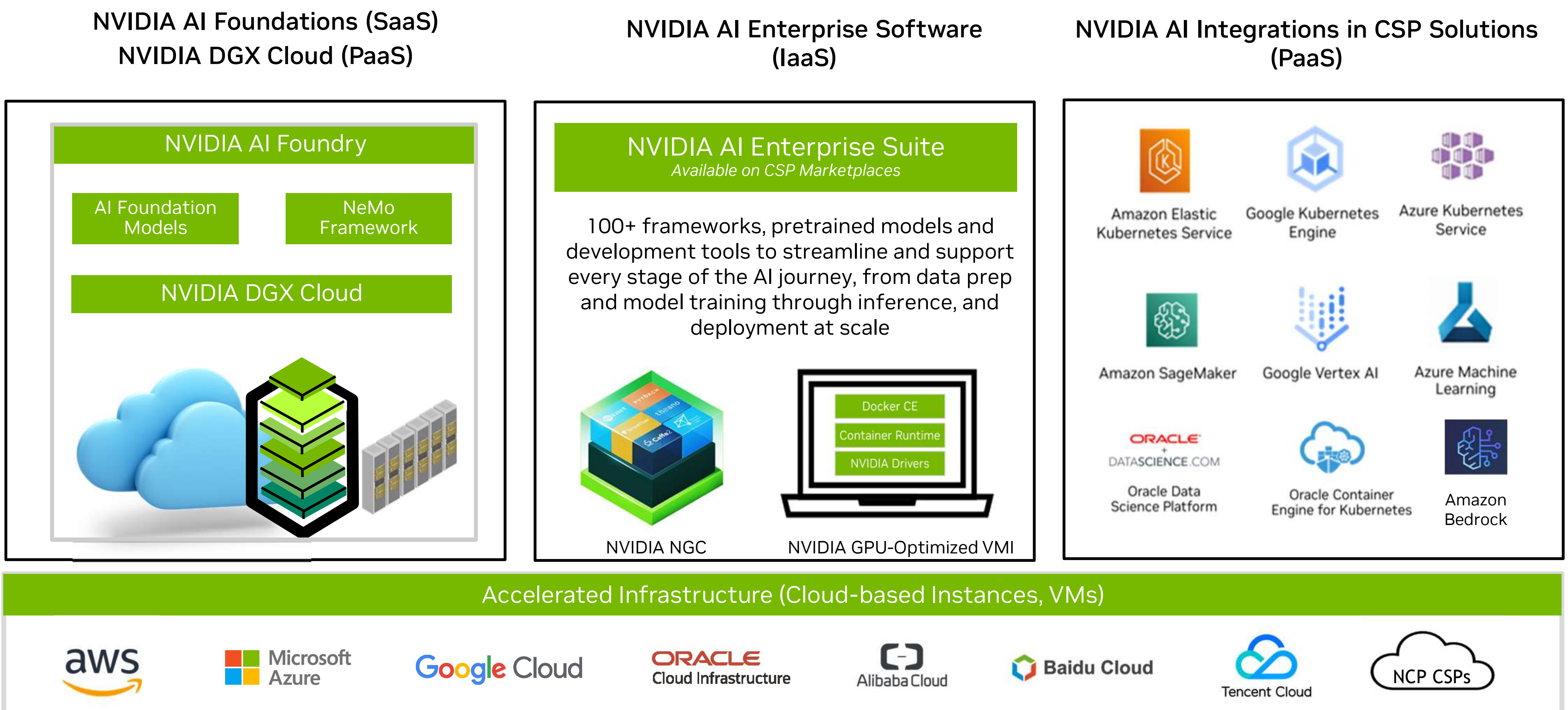
NVIDIA's Latest Platforms Globally Available for Enterprises Using the Cloud

	Ampere			Hopper			Ada Lovelace	
	A100 40GB	A100 80GB	A10	H100	H200*	GH200*	L4*	L40S*
Workloads	AI Training Inference, HPC	AI Training Inference, HPC	Graphics, Gaming, AI Inference	AI Training (LLMs), AI Inference (LLMs) HPC	AI Training (LLMs), AI Inference (LLMs) HPC	AI Training (LLMs), AI Inference (LLMs) HPC	Generative AI, AI-powered Video, Graphics	Generative AI, AI-powered Video, Graphics
AWS	●	●	●	●	●	●	●	●
Microsoft Azure	●	●	●	●	●			
Google Cloud	●	●		●	●		●	
Oracle Cloud	●	●	●	●	●	●		●
Alibaba Cloud	●	●	●					
Tencent Cloud	●		●					
Baidu Cloud	●							
CoreWeave	●	●		●	●	●		●
Cirrascale	●	●		●				
Vultr Cloud		●		●	●	●		●
Paperspace	●	●		●				
Lambda Labs	●	●	●	●	●	●		

* Several CSP H200, GH200, L4, and L40S instances are pre-announcements/private preview/EA/GA. Please refer to each individual CSP's product pages for further details.

Consuming the NVIDIA AI Platform in the Cloud

Multiple Entry Points offering Customers Choice and Flexibility to Build and Deploy AI



NVIDIA

RB0

NVIDIA AI Enterprise Platform in the Public Cloud

Ubiquitous End-to-end Open Platform for Production AI for Everyone, Everywhere
Available on CSP Marketplaces

AI Solution Workflows, Frameworks, and Pretrained Models

Medical Imaging | Speech AI | Customer Service | Recommenders | Physics ML | Communications | Video Analytics | Logistics | Conversational AI | Robotics | Autonomous Vehicles | Cybersecurity

AI and Data Science Development and Deployment Tools

Cloud Native Management and Orchestration

Infrastructure Optimization

Accelerated Infrastructure spanning the Public Cloud

aws | Microsoft Azure | Google Cloud | ORACLE Cloud Infrastructure | Alibaba Cloud | Baidu Cloud | Tencent Cloud | NCP CSPs

Slide 116

RB0 Updated. Added a note up top regarding marketplace availability, and changed "NPN" to "NCP"

Rohil Bhargava, 2023-12-07T05:51:40.929

Leverage NVIDIA AI in CSP Managed On-Prem and Edge Solutions

Extending the Public Cloud Services to On-Prem, Edge or Disconnected Environments

Amazon Web Services	Microsoft Azure	Google Cloud	Oracle Cloud Infrastructure
AWS Outposts Extend AWS services on-prem with support for NVIDIA GPUs for applications with data locality and low-latency requirements **AWS Wavelength** Brings AWS services to edge of 5G networks with NVIDIA GPUs, for ultra-low latency applications **AWS Panorama** Edge appliance with support for NVIDIA Jetson AGX Xavier to bring computer-vision to on-premise cameras	**Azure Stack Hub** Support for NVIDIA GPUs and Networking in Azure Stack Hub integrated systems to deploy AI anywhere **Azure IoT Edge** Support for NVIDIA DeepStream, NVIDIA Fleet Command and NVIDIA GPUs for real-time AI applications at the edge **Azure Stack HCI** Accelerate AI workloads across a hybrid infrastructure with support for NVIDIA GPUs and Networking in Azure Stack HCI managed clusters **Azure Percept** Build and Deploy Edge AI solutions using NVIDIA DeepStream on Azure Stack HCI powered by NVIDIA GPUs	**Anthos on BareMetal, VMware** Create and manage Kubernetes clusters with NVIDIA GPUs on existing infrastructure with VMware or BareMetal Servers **Google Distributed Cloud Hosted** Accelerate sensitive workloads requiring digital sovereignty in a fully-managed on-prem infrastructure with NVIDIA GPUs **Google Distributed Cloud Edge** Accelerate mission-critical AI use-cases with NVIDIA GPUs at Google Edge, Telco Edge or Customer Edge locations **GDC Edge Appliance** Accelerate data processing, analytics & processing with NVIDIA GPUs at remote edge locations	**Oracle Dedicated Region Cloud** Support for NVIDIA GPUs in fully managed on-prem infrastructure applications with low-latency, data residency requirements **Oracle Roving Edge Infrastructure** Deploy AI at remote edge locations with OCI Services, NVIDIA Software and NVIDIA GPUs

Broad Integrations across Cloud Solution Stack (PaaS)

Choose the Level of Abstraction You Need | Accelerate End-to-End Workflows | Reduce Operational Costs

Amazon Web Services	Microsoft Azure	Google Cloud	Oracle Cloud Infrastructure
Amazon Elastic Kubernetes Service Automatically provision, manage and scale K8s clusters with NVIDIA GPU-powered EC2 Instances	**Azure VM Scale Sets** Create, manage and scale up to thousands of NVIDIA GPU-powered VMs and NGC containers	**Google Kubernetes Engine (GKE)** Automatically create, manage and scale K8s clusters with NVIDIA GPUs. Support for GPU-sharing capabilities and NVIDIA NGC containers	**Oracle Data Science Platform** Support for NVIDIA GPUs and RAPIDS to accelerate end-to-end data science and analytics pipelines
Amazon SageMaker Accelerate each step of the end-to-end ML workflow with support for NVIDIA GPUs and NVIDIA NGC software	**Azure CycleCloud** Support to create, manage and orchestrate NVIDIA GPU-based HPC clusters at any scale. Support to deploy NVIDIA NGC containers	**Vertex AI** Access latest NVIDIA AI software like Triton, Merlin, MONAI within a unified MLOps platform to build, deploy and scale ML models in production	**OCI AI Services** Deploy pre-trained ML models or customize them with NVIDIA GPUs on OCI for vision, speech, forecasting and anomaly detection
Amazon ECS Deploy, manage and scale containerized applications on NVIDIA GPU-powered instances including NVIDIA NGC containers	**Azure Kubernetes Service (AKS)** Support to automatically provision, manage and scale K8s clusters with NVIDIA GPUs and NVIDIA NGC containers	**Dataflow** Leverage NVIDIA TensorRT and GPUs to accelerate inference within end-to-end pipelines on streaming data;	**Oracle Kubernetes Engine (OKE)** Automatically provision, manage and scale K8s clusters with NVIDIA GPUs on OCI. Support to deploy NVIDIA NGC containers
Amazon EMR Accelerate large-scale distributed data science pipelines with NVIDIA GPU instances and NVIDIA RAPIDS Accelerator for Apache Spark	**Azure Machine Learning** Leverage NVIDIA GPUs and NVIDIA AI software to accelerate end-to-end ML development and deployments	**Dataproc** Leverage RAPIDS Accelerator for Apache Spark to accelerate Spark SQL/DF based data pipelines with no code changes	

NVIDIA DGX Cloud

AI Training-as-a-Service Platform for the Era of Generative AI

Traditional AI development

DIY tools + open source

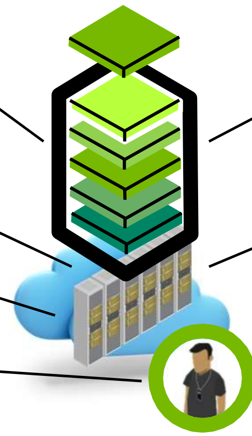
Inconsistent access to multi-node scale across regions

Community forums **"sweat equity"**

Escalating costs, add-on fees for reserved instances, storage, etc.

Traditional AI clouds

NVIDIA DGX Cloud



NVIDIA AI Enterprise
Base Command Platform

DGX Cloud Instance
(8) A100 80GB GPUs
with multi-node scale

10TB storage / instance
10TB egress

NVIDIA AI Expertise

One price
with no surprises

Microsoft Azure | ORACLE CLOUD Infrastructure | Google Cloud

Organizations doubling-down on generative AI need:

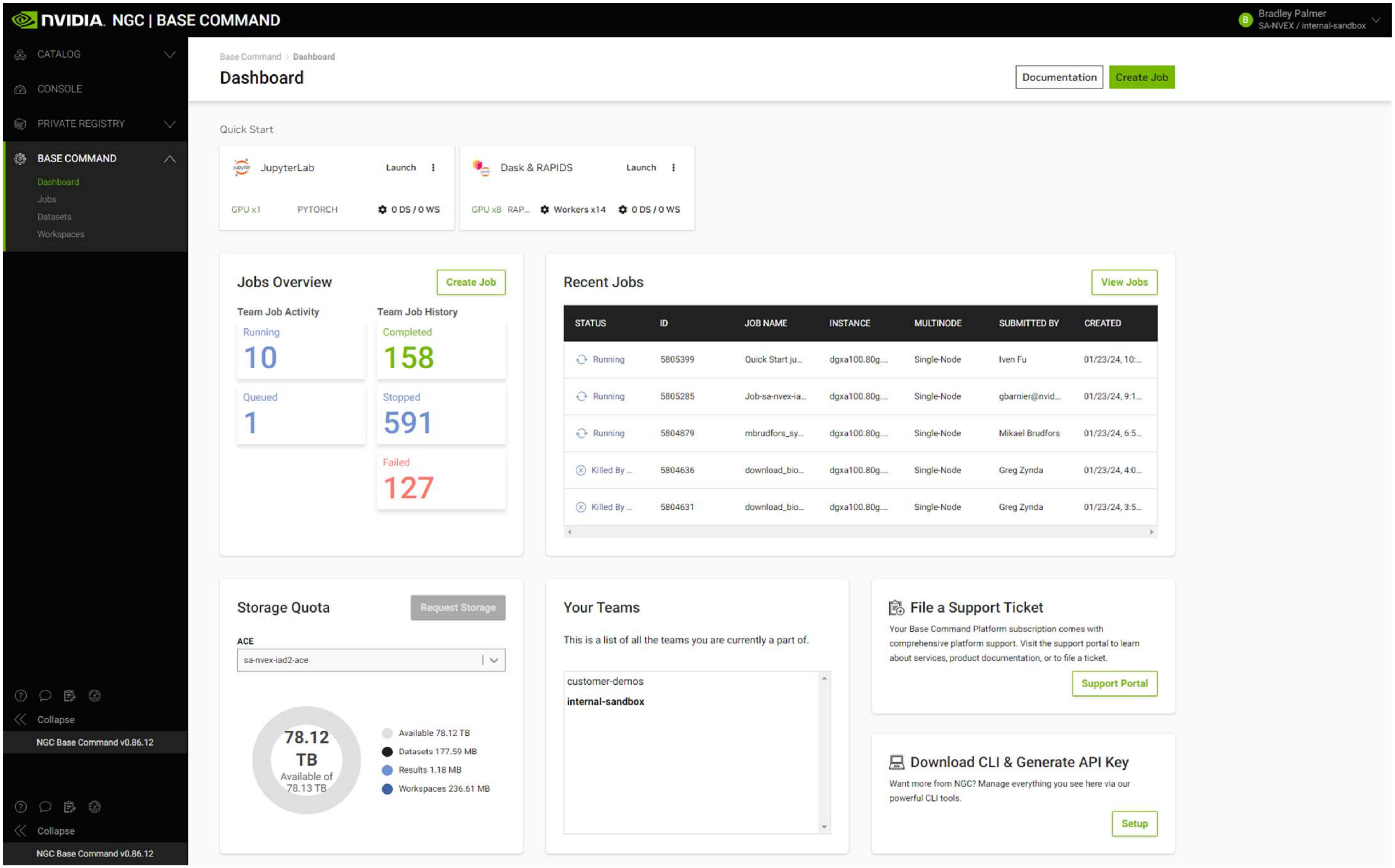
Software that unleashes developer productivity

Multi-node training performance + easy scale

Access AI practitioners who help maximize performance

Predictable pricing that includes everything

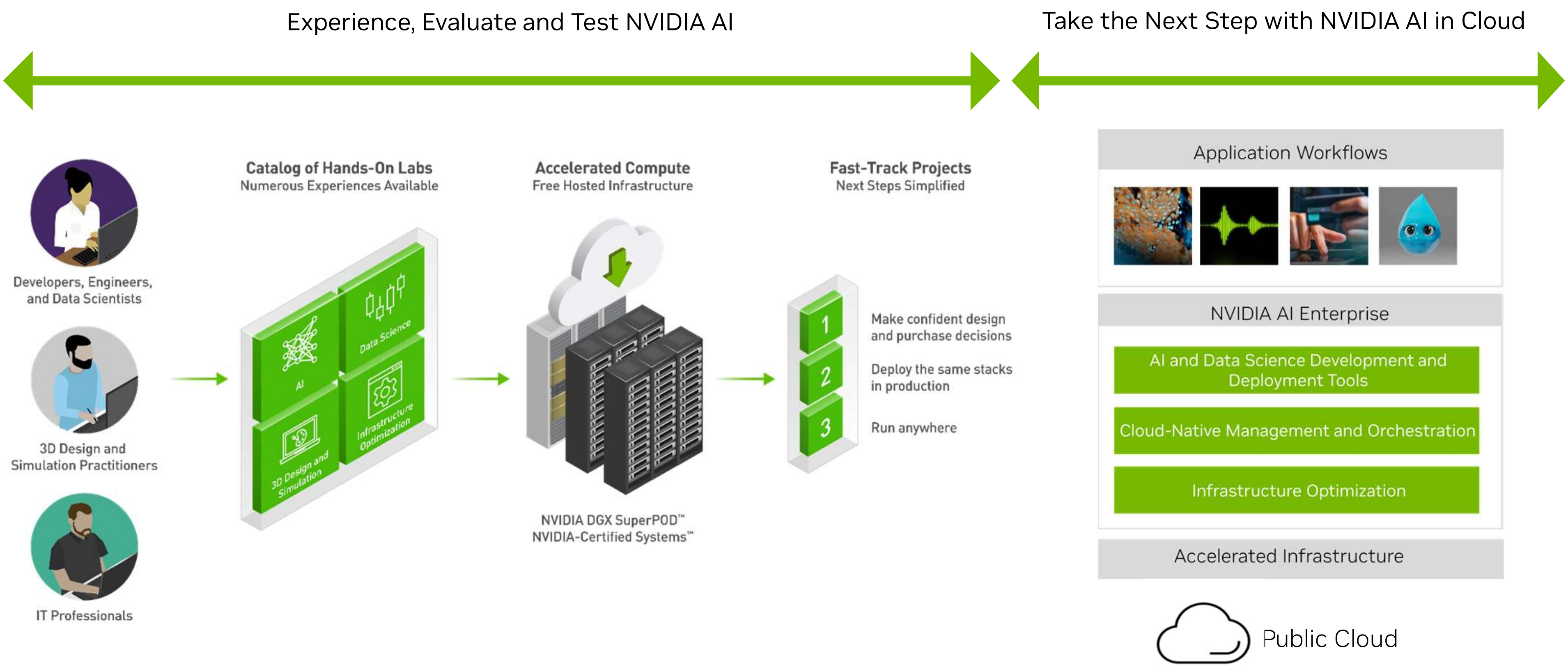
Hosted in **leading clouds**



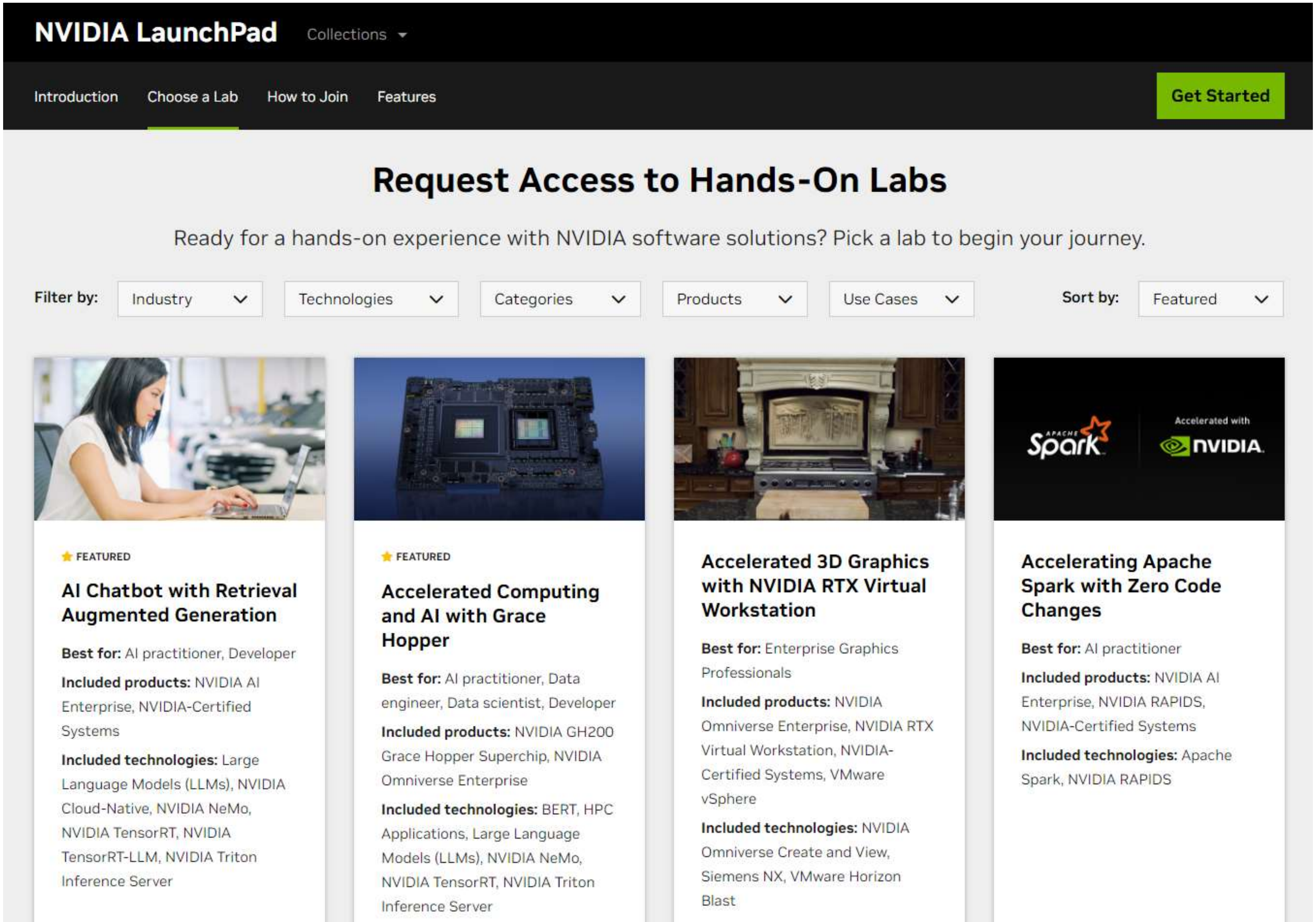
NVIDIA. NGC | BASE COMMAND
Bradley Palmer
SA-NVEX / internal-sandbox
CATALOG
CONSOLE
PRIVATE REGISTRY
BASE COMMAND
Dashboard
Jobs
Datasets
Workspaces
Collapse
NGC Base Command v0.86.12
Base Command > Dashboard
Dashboard
Documentation
Create Job
Quick Start
JupyterLab
Launch
GPU x1
PYTORCH
0 DS / 0 WS
Dask & RAPIDS
Launch
GPU x8
RAP...
Workers x14
0 DS / 0 WS
Jobs Overview
Create Job
Team Job Activity
Team Job History
Running
10
Queued
1
Completed
158
Stopped
591
Failed
127
Recent Jobs
View Jobs
STATUS | ID | JOB NAME | INSTANCE | MULTINODE | SUBMITTED BY | CREATED
Running | 5805399 | Quick Start ju... | dgxa100.80g.... | Single-Node | Iven Fu | 01/23/24, 10:...
Running | 5805285 | Job-sa-nvex-ia... | dgxa100.80g.... | Single-Node | gbarnier@nvid... | 01/23/24, 9:1...
Running | 5804879 | mbrudfors_sy... | dgxa100.80g.... | Single-Node | Mikael Brudfors | 01/23/24, 6:5...
Killed By ... | 5804636 | download_bio... | dgxa100.80g.... | Single-Node | Greg Zynda | 01/23/24, 4:0...
Killed By ... | 5804631 | download_bio... | dgxa100.80g.... | Single-Node | Greg Zynda | 01/23/24, 3:5...
Storage Quota
Request Storage
ACE
sa-nvex-iad2-ace
78.12 TB
Available of 78.13 TB
Available 78.12 TB
Datasets 177.59 MB
Results 1.18 MB
Workspaces 236.61 MB
Your Teams
This is a list of all the teams you are currently a part of.
customer-demos
internal-sandbox
File a Support Ticket
Your Base Command Platform subscription comes with comprehensive platform support. Visit the support portal to learn about services, product documentation, or to file a ticket.
Support Portal
Download CLI & Generate API Key
Want more from NGC? Manage everything you see here via our powerful CLI tools.
Setup
Collapse
NGC Base Command v0.86.12

NVIDIA LaunchPad

Instantly experience end-to-end workflows for AI, data science, 3D design collaboration, and more



https://www.nvidia.com/en-us/launchpad/


NVIDIA LaunchPad
Collections
Introduction
Choose a Lab
How to Join
Features
Get Started
Request Access to Hands-On Labs
Ready for a hands-on experience with NVIDIA software solutions? Pick a lab to begin your journey.
Filter by:
Industry
Technologies
Categories
Products
Use Cases
Sort by:
Featured
Spark
Accelerated with
NVIDIA
FEATURED
AI Chatbot with Retrieval Augmented Generation
Best for: AI practitioner, Developer
Included products: NVIDIA AI Enterprise, NVIDIA-Certified Systems
Included technologies: Large Language Models (LLMs), NVIDIA Cloud-Native, NVIDIA NeMo, NVIDIA TensorRT, NVIDIA TensorRT-LLM, NVIDIA Triton Inference Server
FEATURED
Accelerated Computing and AI with Grace Hopper
Best for: AI practitioner, Data engineer, Data scientist, Developer
Included products: NVIDIA GH200 Grace Hopper Superchip, NVIDIA Omniverse Enterprise
Included technologies: BERT, HPC Applications, Large Language Models (LLMs), NVIDIA NeMo, NVIDIA TensorRT, NVIDIA Triton Inference Server
Accelerated 3D Graphics with NVIDIA RTX Virtual Workstation
Best for: Enterprise Graphics Professionals
Included products: NVIDIA Omniverse Enterprise, NVIDIA RTX Virtual Workstation, NVIDIA-Certified Systems, VMware vSphere
Included technologies: NVIDIA Omniverse Create and View, Siemens NX, VMware Horizon Blast
Accelerating Apache Spark with Zero Code Changes
Best for: AI practitioner
Included products: NVIDIA AI Enterprise, NVIDIA RAPIDS, NVIDIA-Certified Systems
Included technologies: Apache Spark, NVIDIA RAPIDS

Resource

Resources

Where to find all things NVIDIA

- Open a developer account — https://developer.nvidia.com/
- Find documentation — https://docs.nvidia.com/
- Open software and tutorials — https://github.com/NVIDIA
- Replay GTC sessions — https://www.nvidia.com/en-us/on-demand/
- Free and fee-based training — https://www.nvidia.com/en-us/training/
- NGC for containers, trained models, workflows — https://ngc.nvidia.com/
- Try out new technology with LaunchPad — https://www.nvidia.com/en-us/launchpad/
- Register for GTC (March 17-21) — https://www.nvidia.com/gtc/

NVIDIA