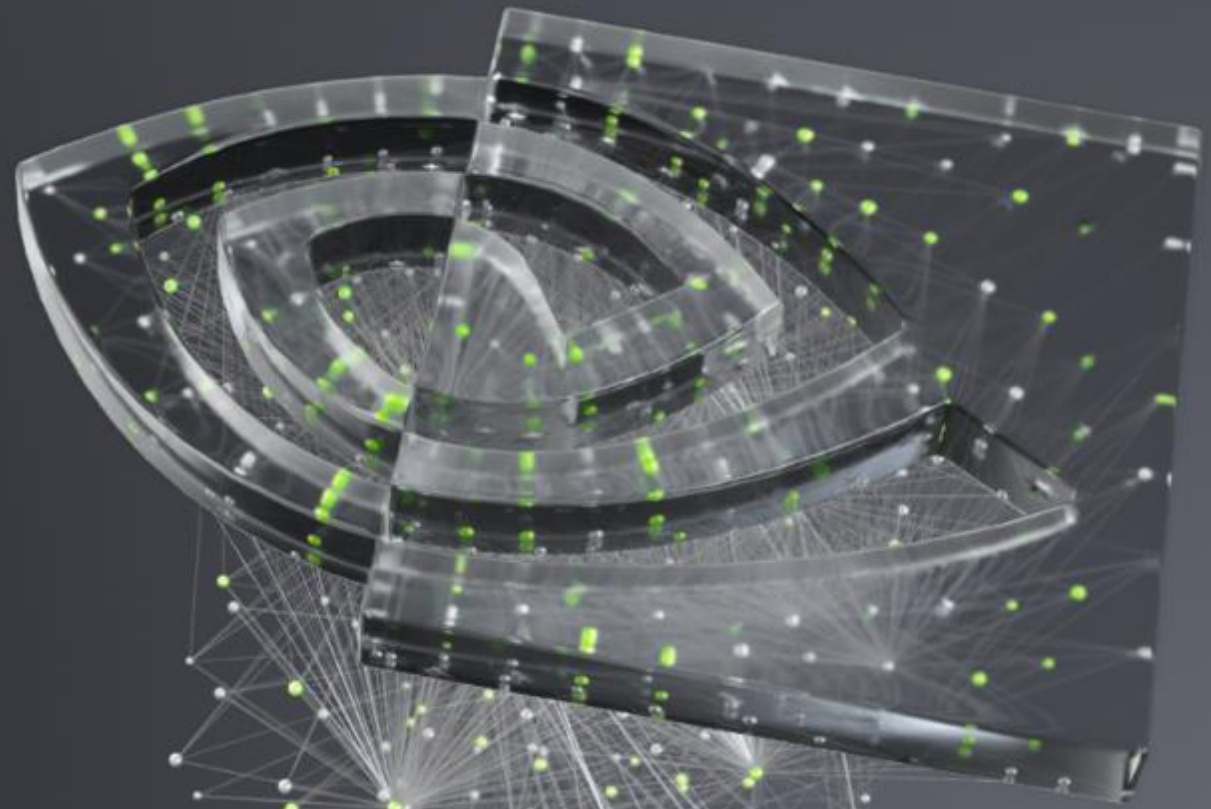
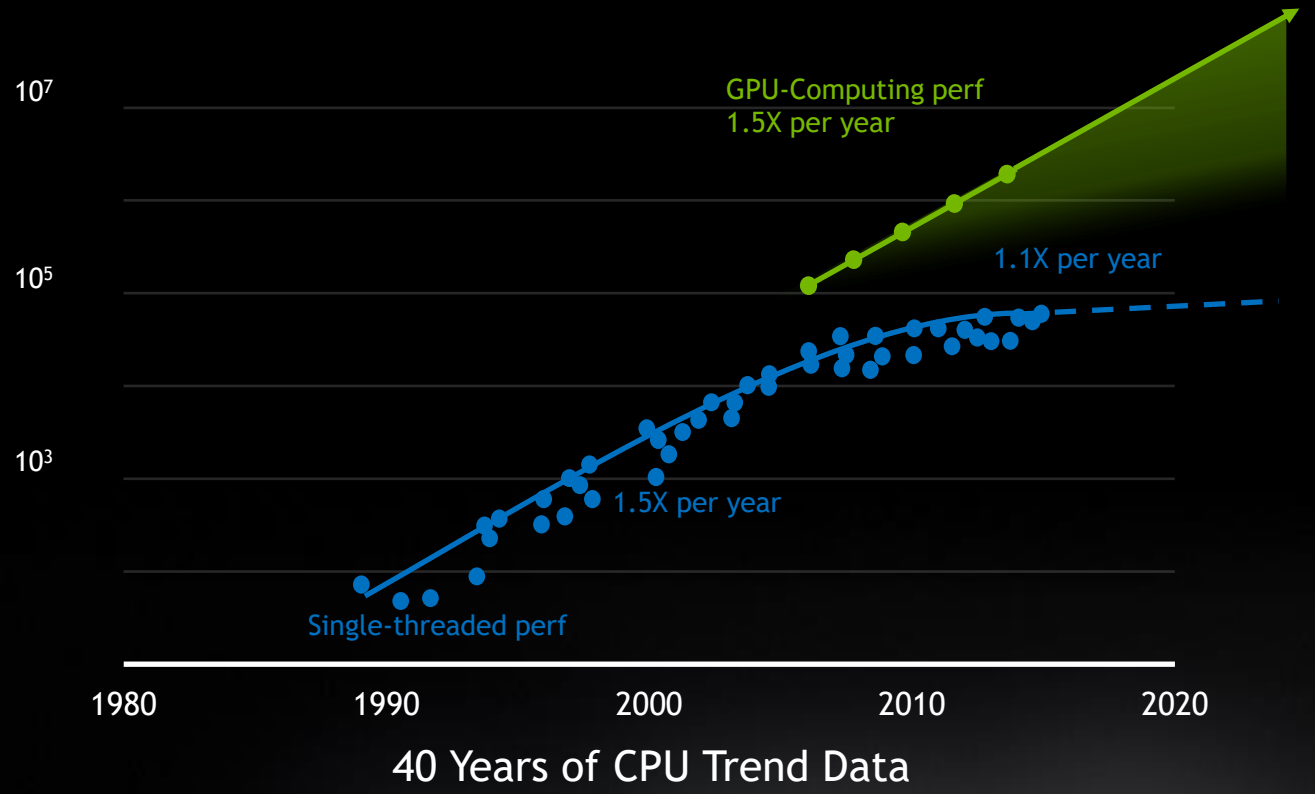
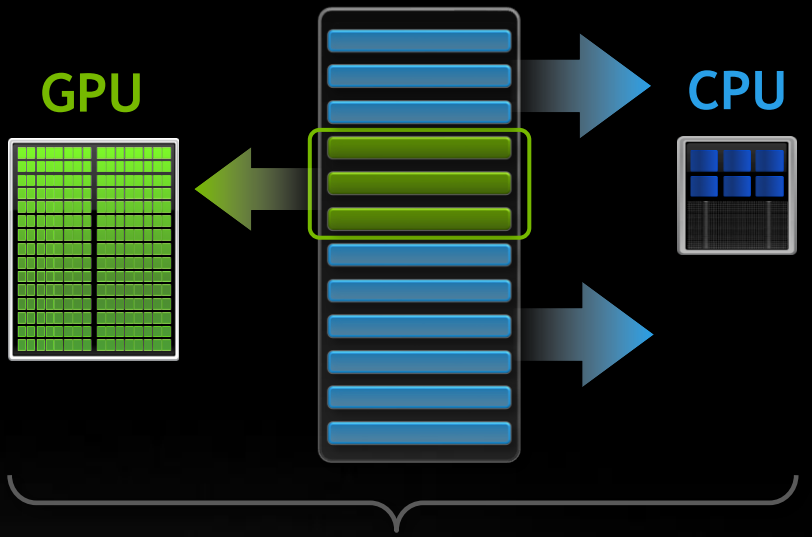


CIRRUS
EFFICIENT USE OF GPUS FOR HPC

Paul Graham | Senior Solutions Architect | NVIDIA

RISE OF GPU COMPUTING



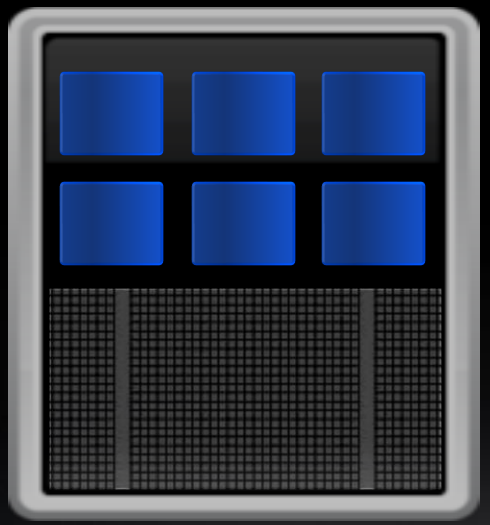


Original data up to the year 2010 collected and plotted by M. Horowitz,
F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten New plot and data collected for 2010-2015 by K. Rupp

CPU IS A LATENCY REDUCING ARCHITECTURE

CPU

Optimized for
Serial Tasks



CPU Strengths

- Very large main memory
- Very fast clock speeds
- Latency optimized via large caches
- Small number of threads can run very quickly

CPU Weaknesses

- Relatively low memory bandwidth
- Cache misses very costly
- Low performance/watt

GPU IS ALL ABOUT HIDING LATENCY

GPU Strengths

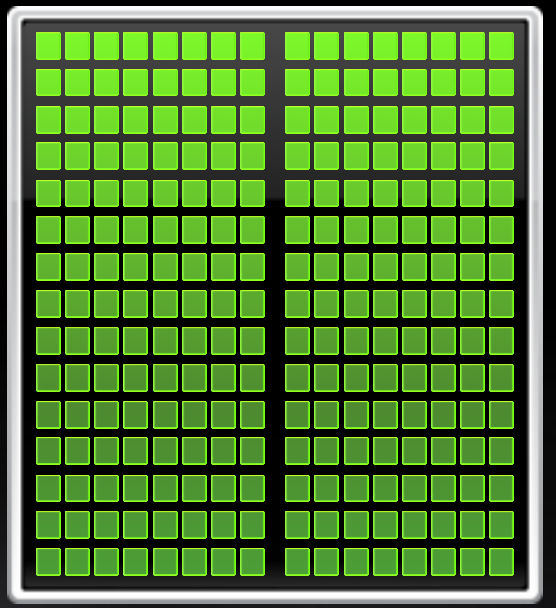
- High bandwidth main memory
- Significantly more compute resources
- Latency tolerant via parallelism
- High throughput
- High performance/watt

GPU Weaknesses

- Relatively low memory capacity
- Low per-thread performance

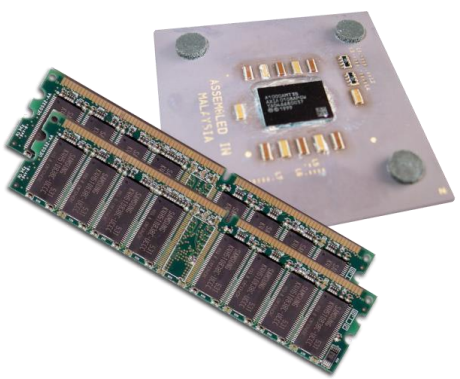
GPU Accelerator

Optimized for Parallel Tasks

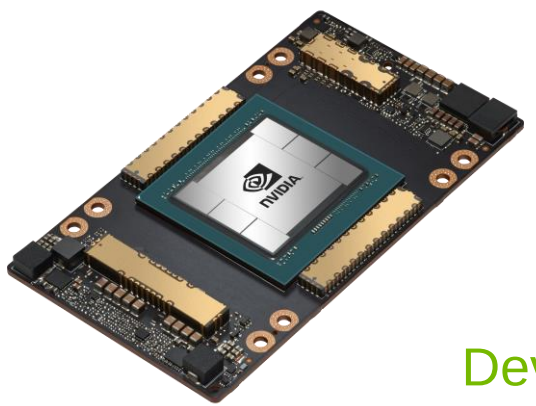


Heterogeneous Computing

- Terminology:
  - *Host* The CPU and its memory (host memory)
  - *Device* The GPU and its memory (device memory)

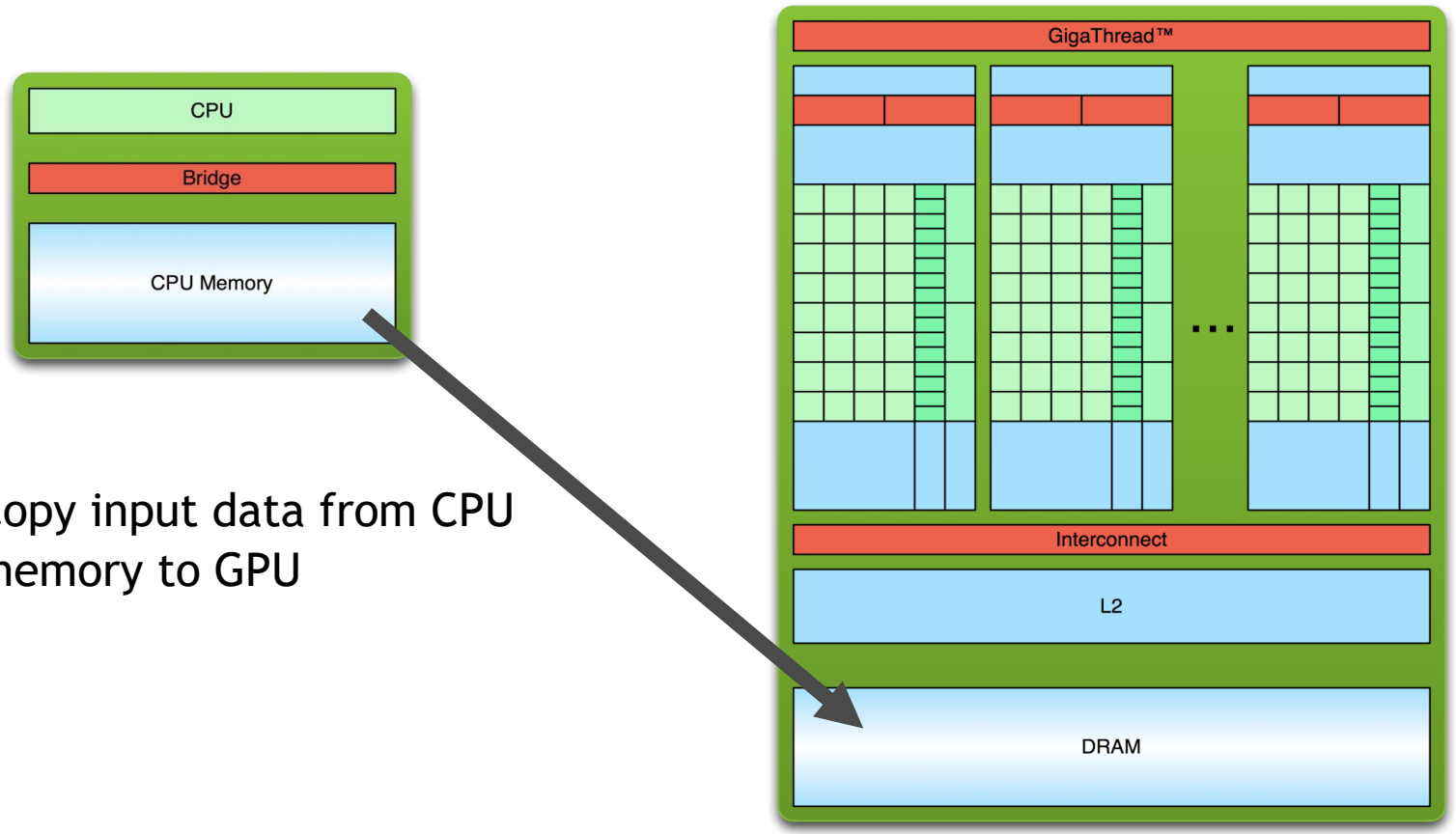


Host



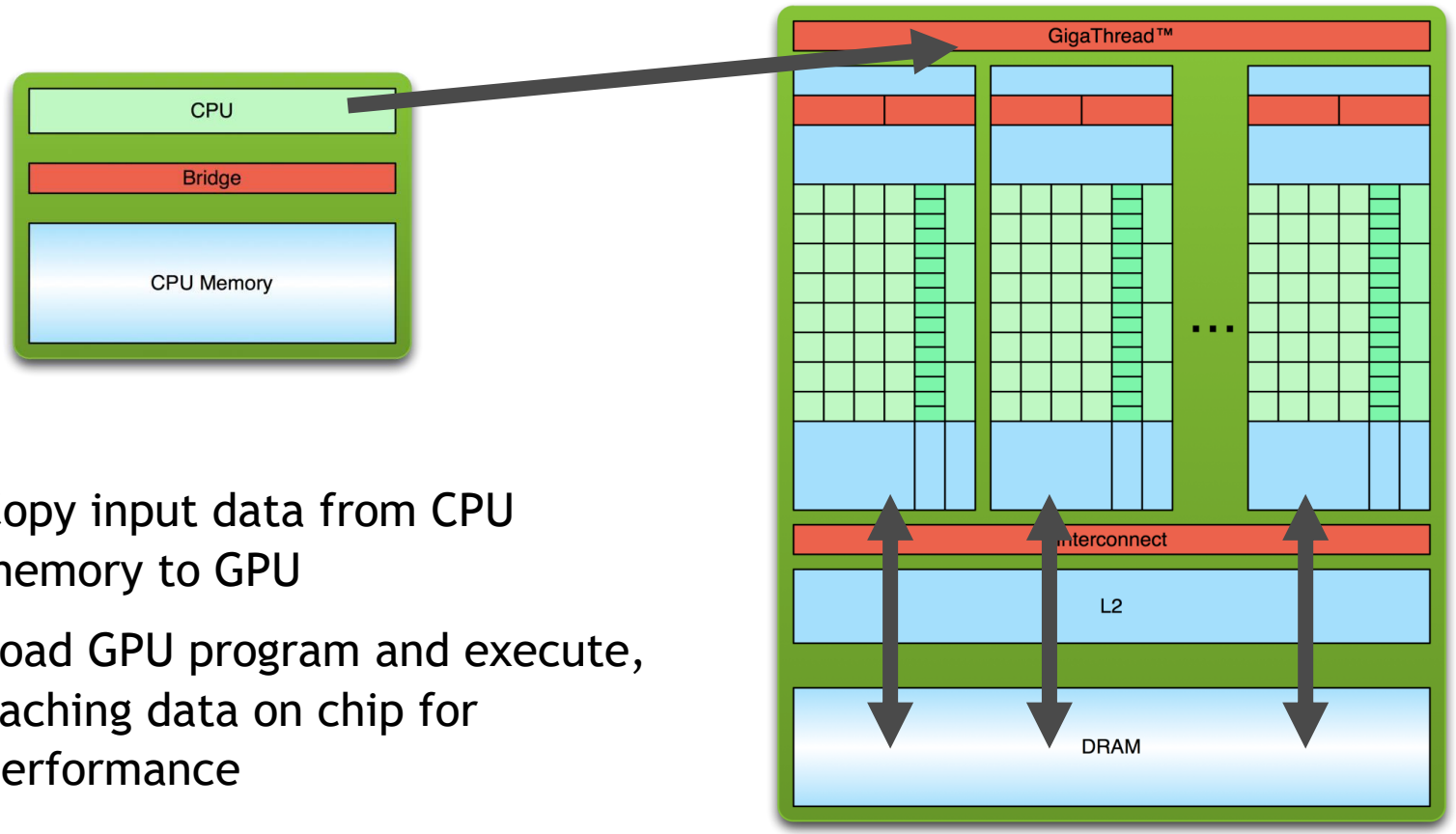
Device

SIMPLE PROCESSING FLOW



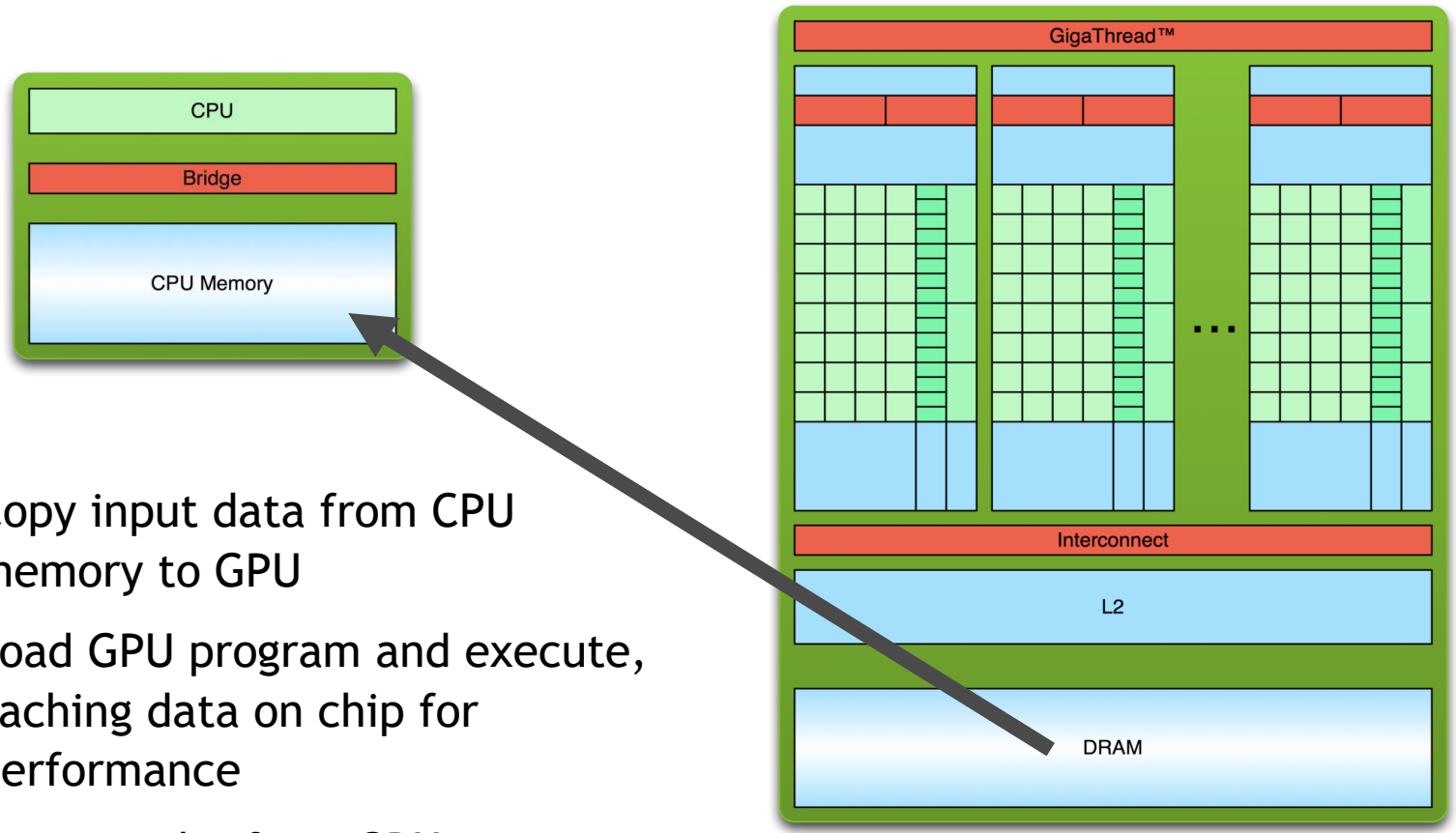
1. Copy input data from CPU memory to GPU

SIMPLE PROCESSING FLOW



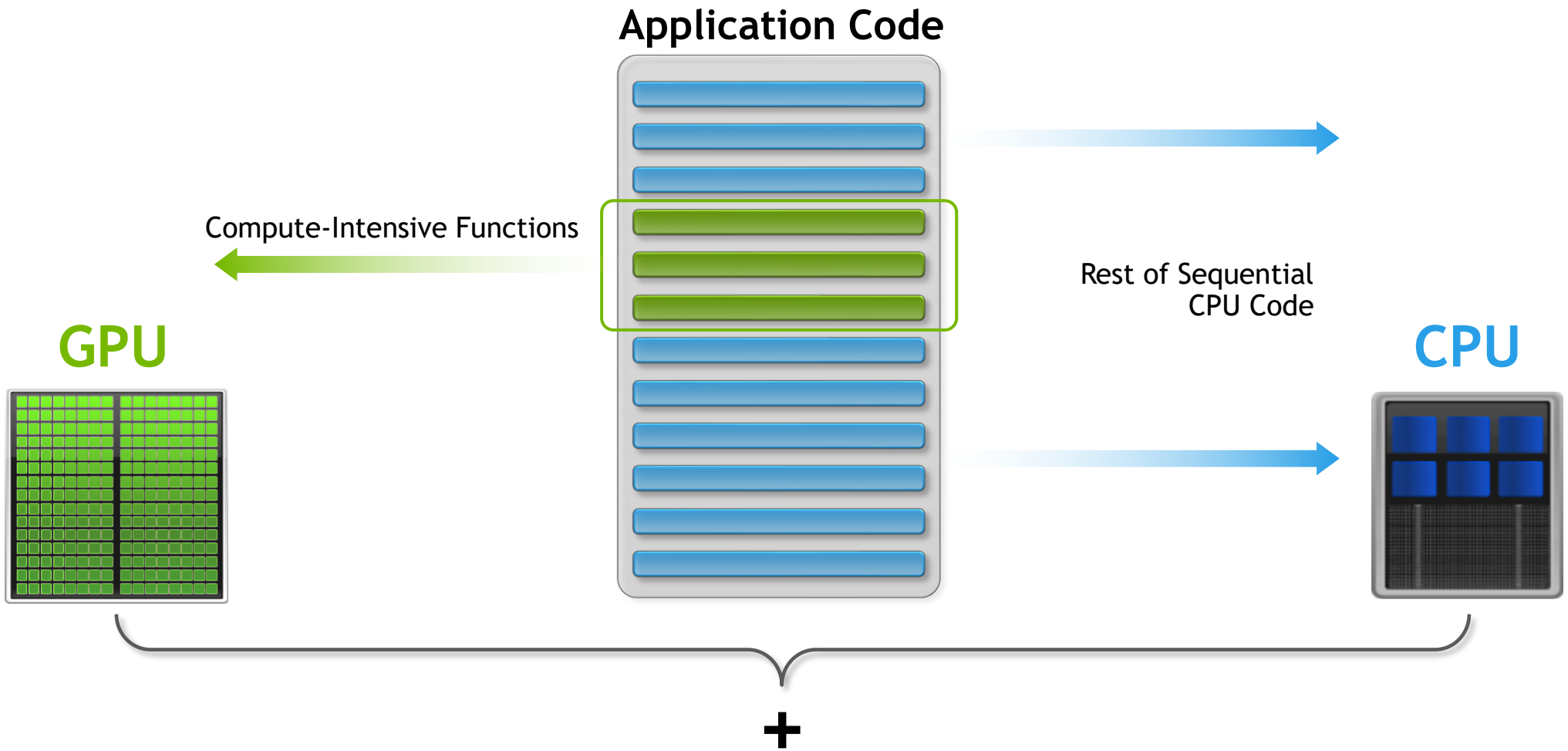
1. Copy input data from CPU memory to GPU
2. Load GPU program and execute, caching data on chip for performance

SIMPLE PROCESSING FLOW



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

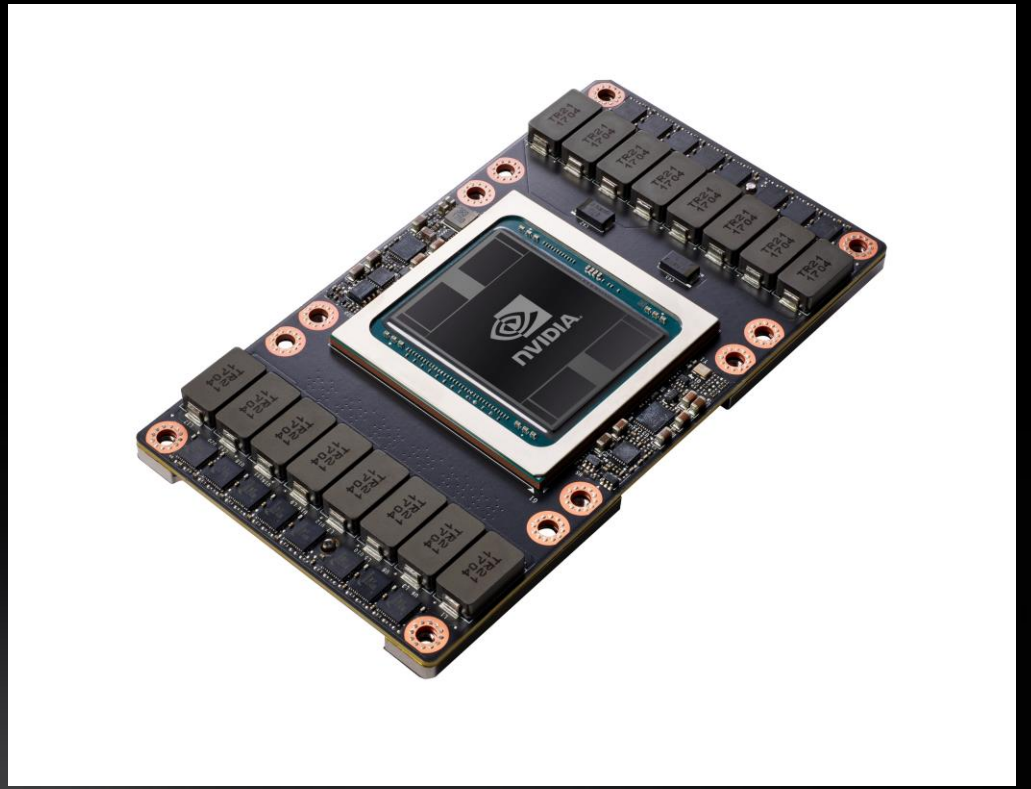
SMALL CHANGES, BIG SPEED-UP



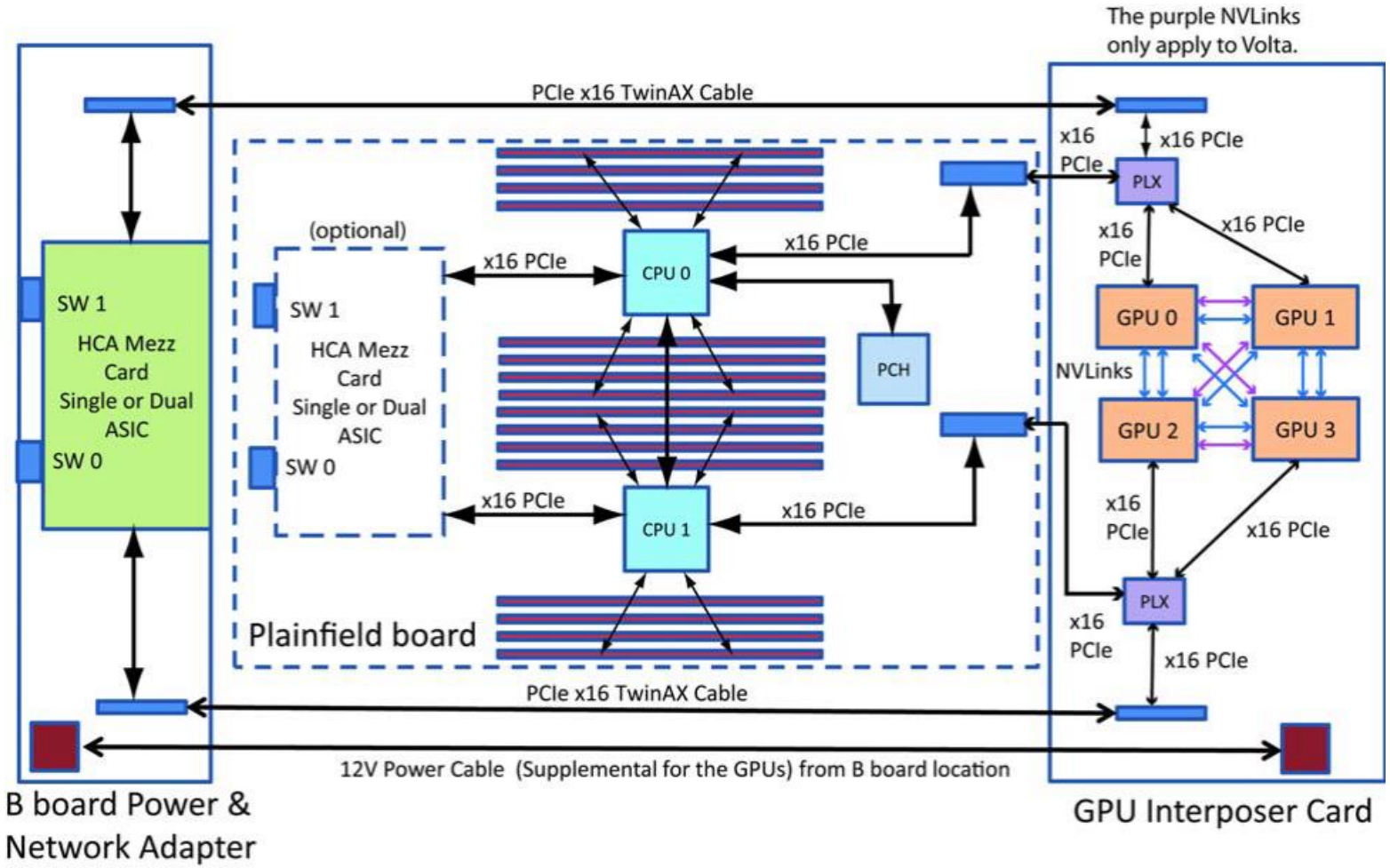
HARDWARE

V100 SXM2 IN CIRRUS

- 7.8 TFLOPS of double precision floating-point (FP64) performance
- 15.7 TFLOPS of single precision (FP32) performance
- 125 Tensor TFLOPS
- 16GB Memory
  - 900 GB/sec peak bandwidth
- 6MB L2 cache



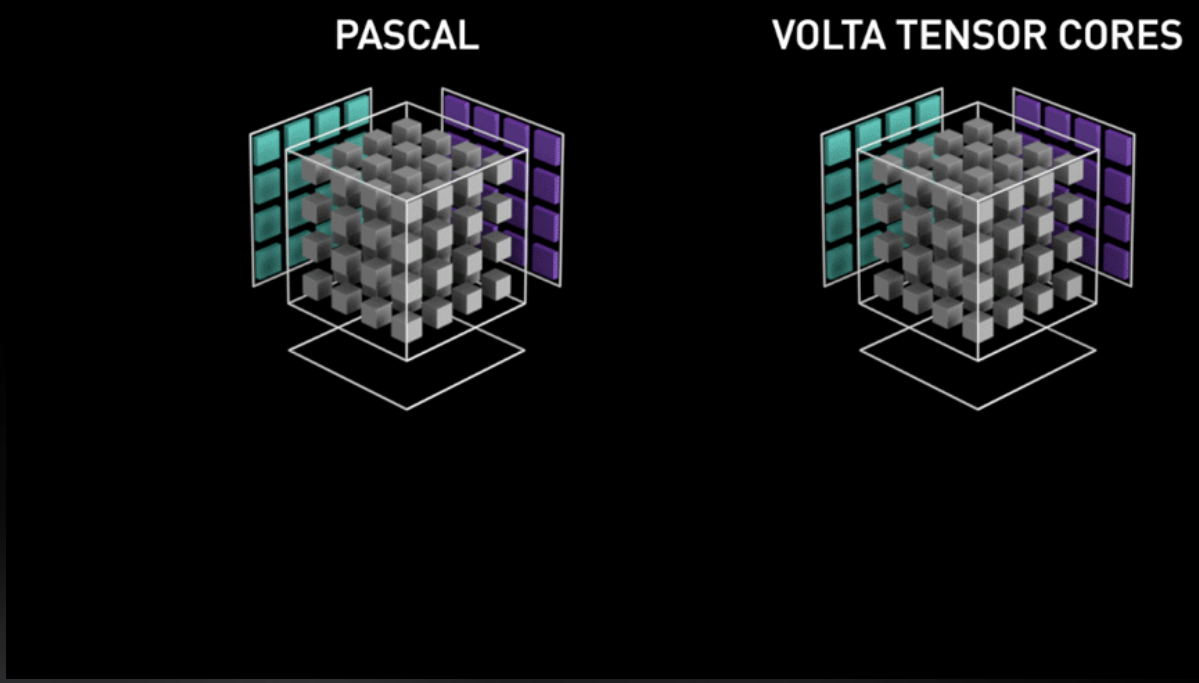
HPE PLAINFIELD NODE



TENSOR CORES

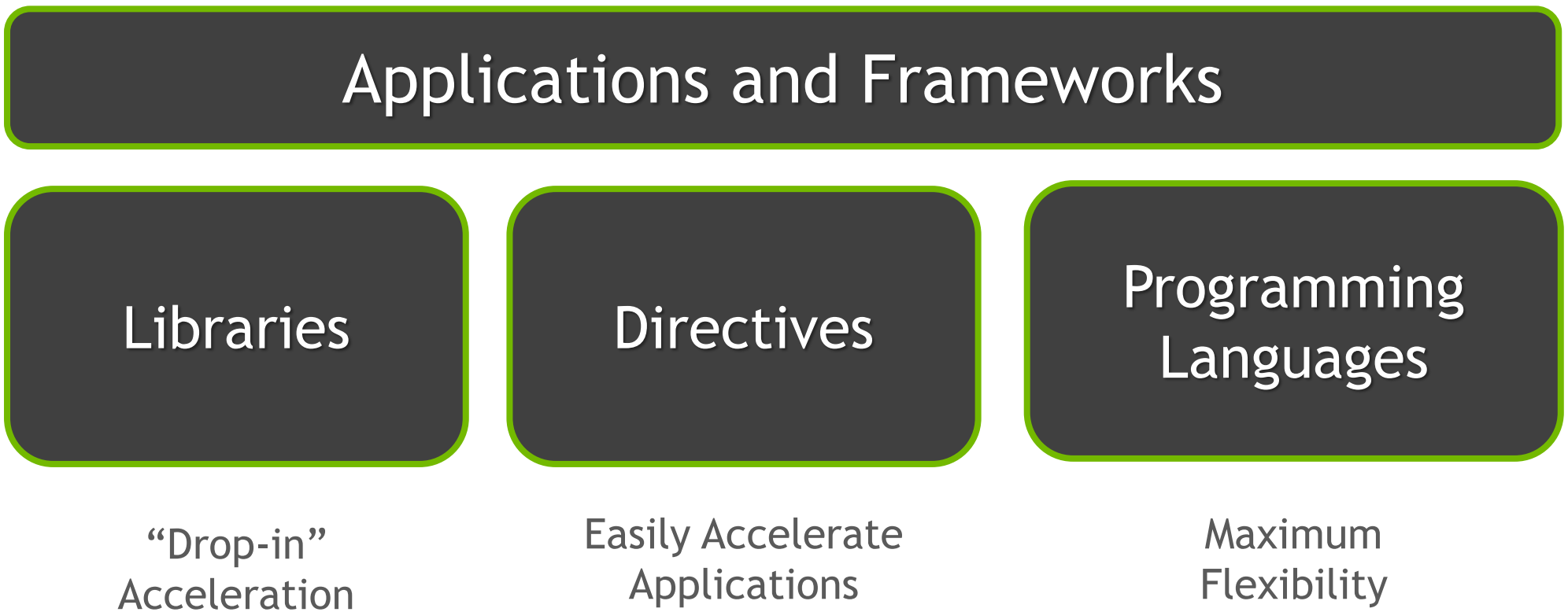
Hardware for Matrix Multiply and Accumulate operations

- Introduced in the V100
- Perform several MMA calcs per clock cycle
  - FP32 in, FP32 out (accumulate)
  - FP16 multiply
- Turing added int8, int4 calculations
- Ampere
  - Full FP64 MMA
  - Bfloat16, Tensor Float 32



PROGRAMMING

WAYS TO ACCELERATION



GPU ACCELERATED APPS AND FRAMEWORKS

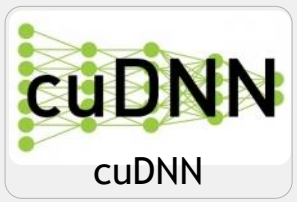
- All major DL frameworks – PyTorch, TensorFlow etc
- Top 15 most used HPC apps
- Apps in a huge range of fields
- Over 1000 apps in total
  Catalogue: link
- Domain-specific frameworks – robotics, vis, healthcare etc

- GROMACS, VASP, LAMMPS, RELION, QE, NAMD, SPECFEM3D ...
  - https://developer.nvidia.com/hpc-application-performance
- NRF for V100x4 – GROMACS 10-14, LAMMPS 9-44, SPECFEM3D 53

GPU ACCELERATED LIBRARIES

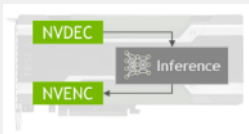
“Drop-in” Acceleration for Your Applications

DEEP LEARNING



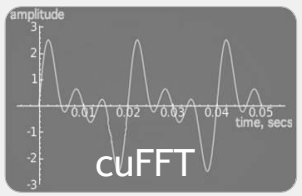
cuDNN


TensorRT



DeepStream SDK

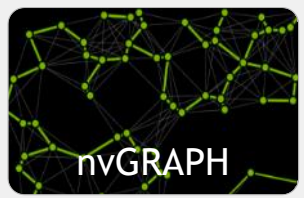
SIGNAL, IMAGE & VIDEO

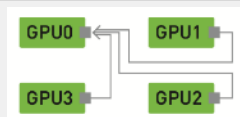


LINEAR ALGEBRA


cuSPARSE

PARALLEL ALGORITHMS





NCCL

More libraries: https://developer.nvidia.com/gpu-accelerated-libraries

OpenACC is a directives-based programming approach to **parallel computing** designed for **performance** and **portability** on CPUs and GPUs for HPC.

Add Simple Compiler Directive

```
main()
 {
  <serial code>
  #pragma acc kernels
  {
    <parallel code>
  }
}
```

GPU PROGRAMMING LANGUAGES

Numerical analytics	MATLAB, Mathematica, LabVIEW
Fortran	CUDA Fortran, OpenACC, ISO Fortran
C, C++	CUDA C++, OpenACC, ISO C++
Python	CUDA Python, PyCUDA
C#	Altimesh Hybridizer, Alea GPU

GPU PROGRAMMING IN 2021 AND BEYOND

Math Libraries | Standard Languages | Directives | CUDA

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

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

GPU Accelerated
C++ and Fortran

```
#pragma acc data copy(x,y)
{

...

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

...

}
```

Incremental Performance
Optimization with Directives

```
__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, ...);
```

Maximize GPU Performance with
CUDA C++/Fortran

GPU Accelerated Libraries

```cpp
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

PARALLEL C++

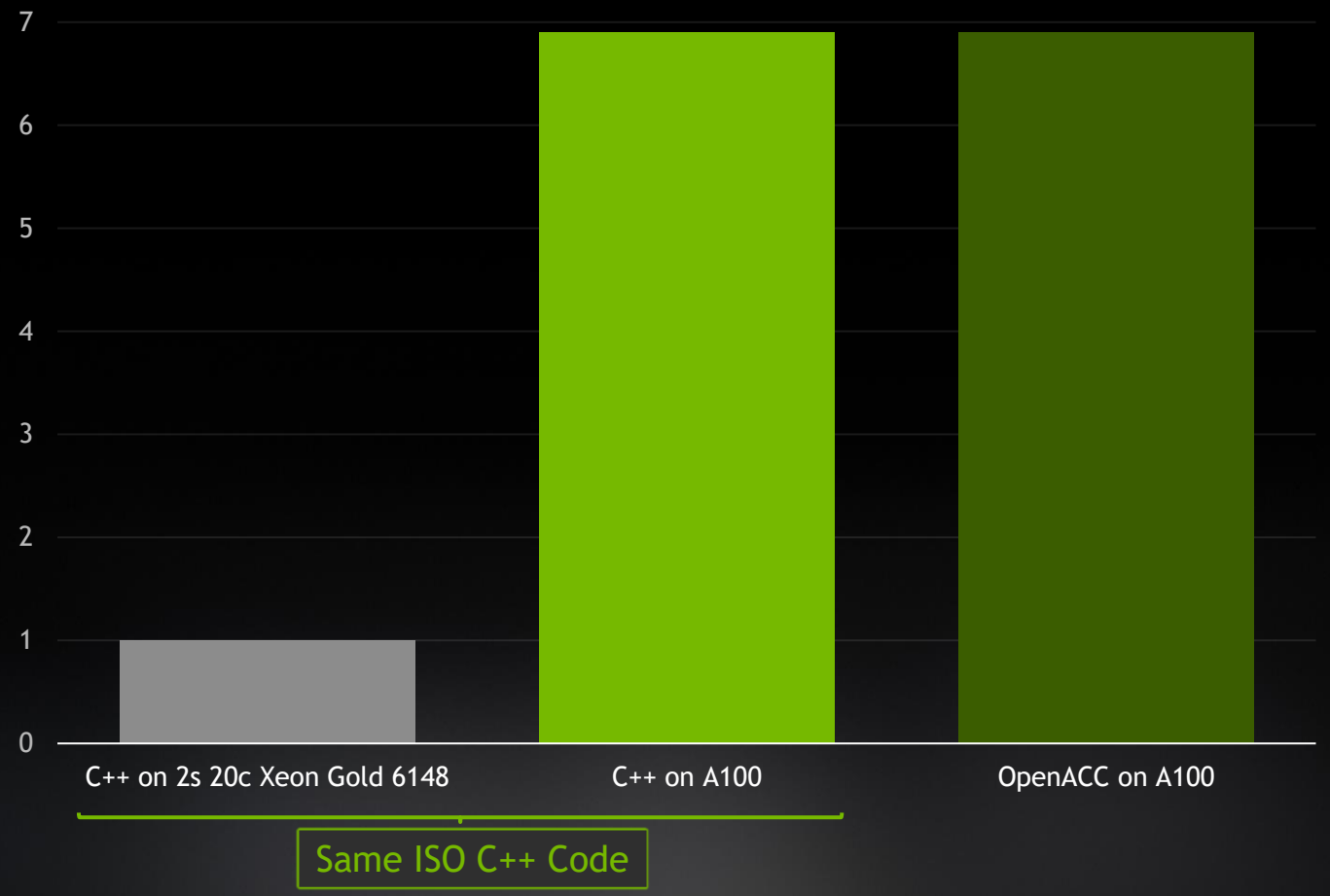
- Composable, compact and elegant
- Easy to read and maintain
- ISO Standard
- Portable – nvc++, g++, icpc, MSVC, …

```cpp
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));
      }
  });
}
```

Parallel C++17

LULESH PERFORMANCE

Speedup – Higher is Better



HPC PROGRAMMING IN ISO FORTRAN

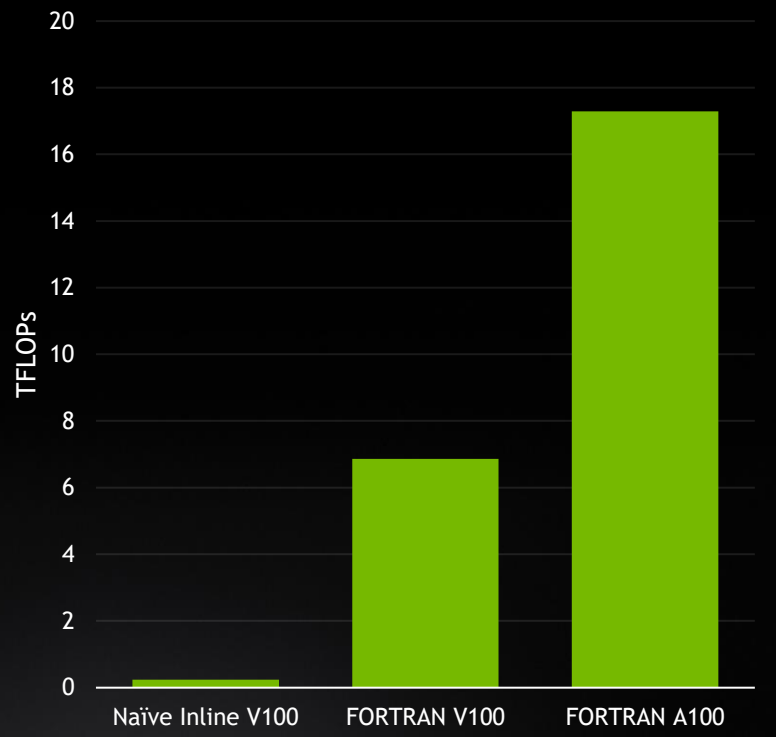
NVFORTRAN Accelerates Fortran Intrinsics with cuTENSOR Backend

```
real(8), dimension(ni,nk) :: a
real(8), dimension(nk,nj) :: b
real(8), dimension(ni,nj) :: c, d

...

!$acc enter data copyin(a,b,c) create(d)

do nt = 1, ntimes
    !$acc kernels
    do j = 1, nj
        do i = 1, ni
            d(i,j) = c(i,j)
            do k = 1, nk
                d(i,j) = d(i,j) + a(i,k) * b(k,j)
            end do
        end do
    end do
    !$acc end kernels
end do

!$acc exit data copyout(d)
```

Inline FP64 matrix multiply

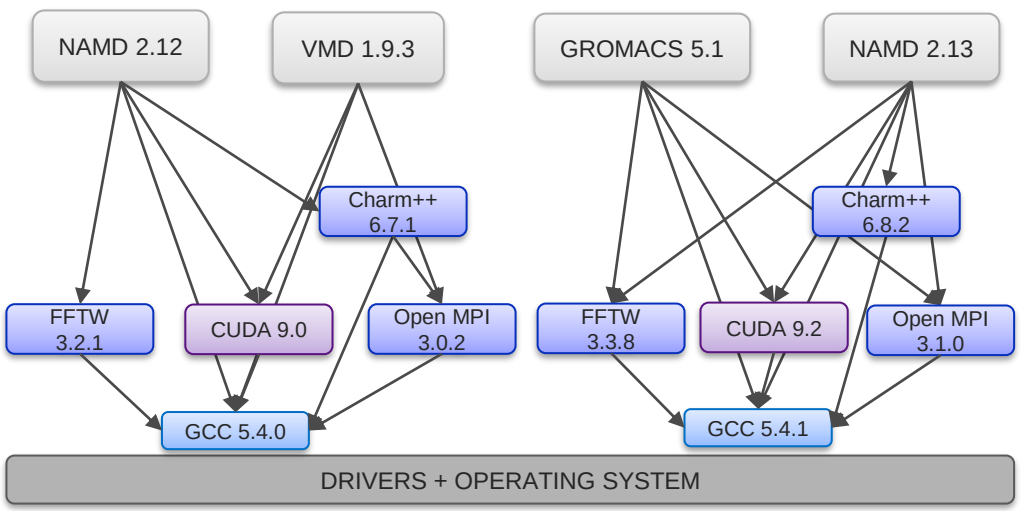
```
real(8), dimension(ni,nk) :: a
real(8), dimension(nk,nj) :: b
real(8), dimension(ni,nj) :: c, d

...

!$acc enter data copyin(a,b,c) create(d)

...

!$acc host_data use_device(a,b,c,d)
do nt = 1, ntimes
    d = c + matmul(a,b)
end do
!$acc end host_data

...

!$acc exit data copyout(d)
```

MATMUL FP64 matrix multiply

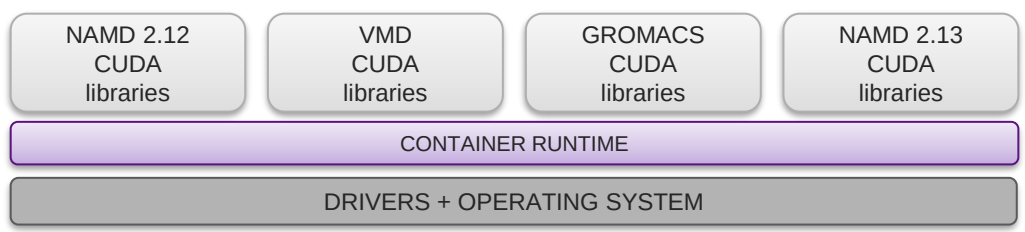


USING GPUS EFFICIENTLY

TYPICAL HPC LIBRARIES & BARE METAL



BARE METAL



CONTAINERS

WHY CONTAINERS?

- Answers the question: “How to get software to run reliably when moved from one computing environment to another”
  - Run anywhere OS kernel supports it (Mobility of Compute)
  - Greatly reduces time-consuming and error-prone bare metal application installations when moving from system to system
- Greatly improves reproducibility (key for HPC)
- Consistent Environment
- Flexibility
- Simplify deployment of software, particularly GPU-accelerated software
- Share, collaborate, and test applications across different environments
- Equivalent performance to baremetal

NGC: GPU-OPTIMIZED SOFTWARE HUB

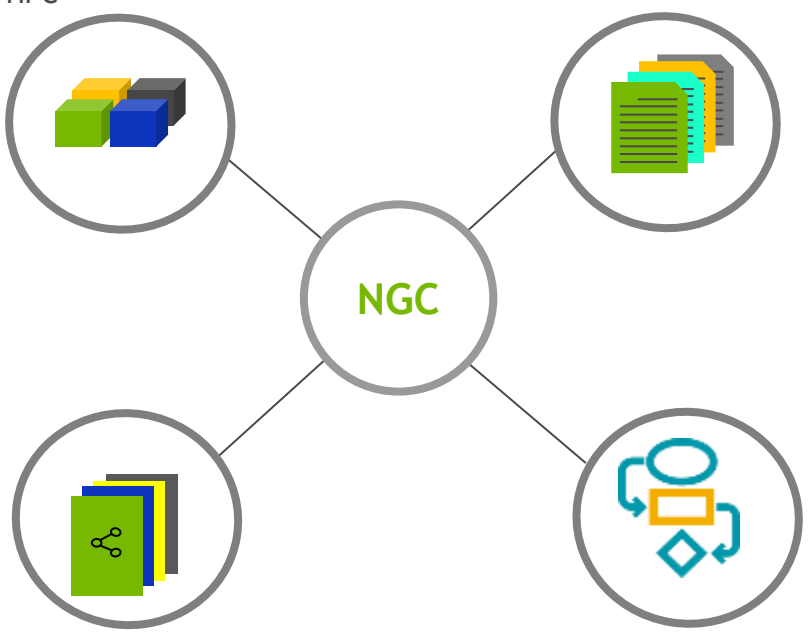
Simplifying DL, ML and HPC Workflows

50+ Containers

DL, ML, HPC

15+ Model Training Scripts

NLP, Image Classification, Object Detection and more



60 Pre-trained Models

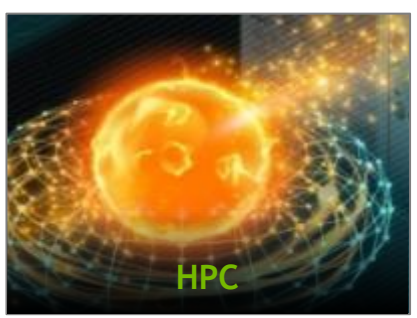
NLP, Image Classification, Object Detection and more

Workflows

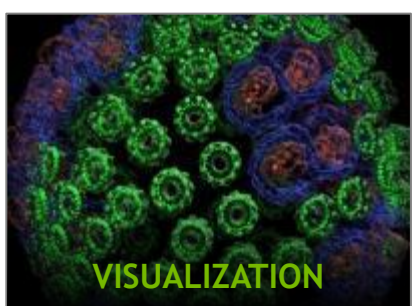
Medical Imaging, Intelligent Video Analytics

TensorFlow | PyTorch | more

RAPIDS | H2O | more



NAMD | GROMACS | more

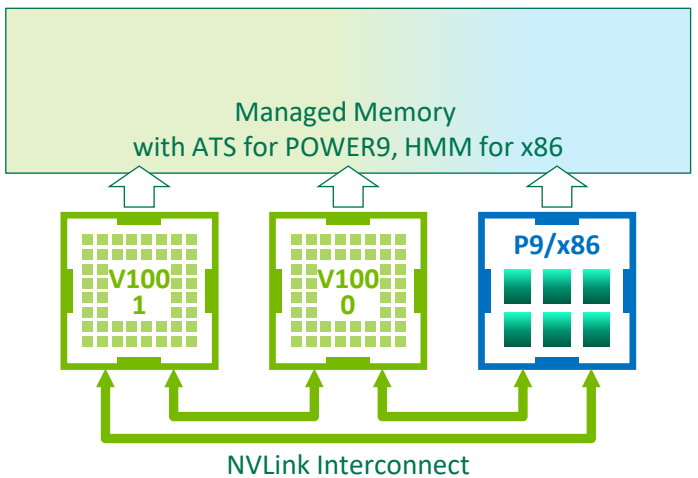


ParaView | IndeX | more

NVIDIA

UNIFIED MEMORY

Access all memory in the node



ALLOCATION

Automatic access to all system memory: malloc, statics, globals, stack, file system

ACCESS

All data accessible concurrently from any processor, anytime

Atomic operations resolved directly over NVLink

VOLTA MULTI-PROCESS SERVICE

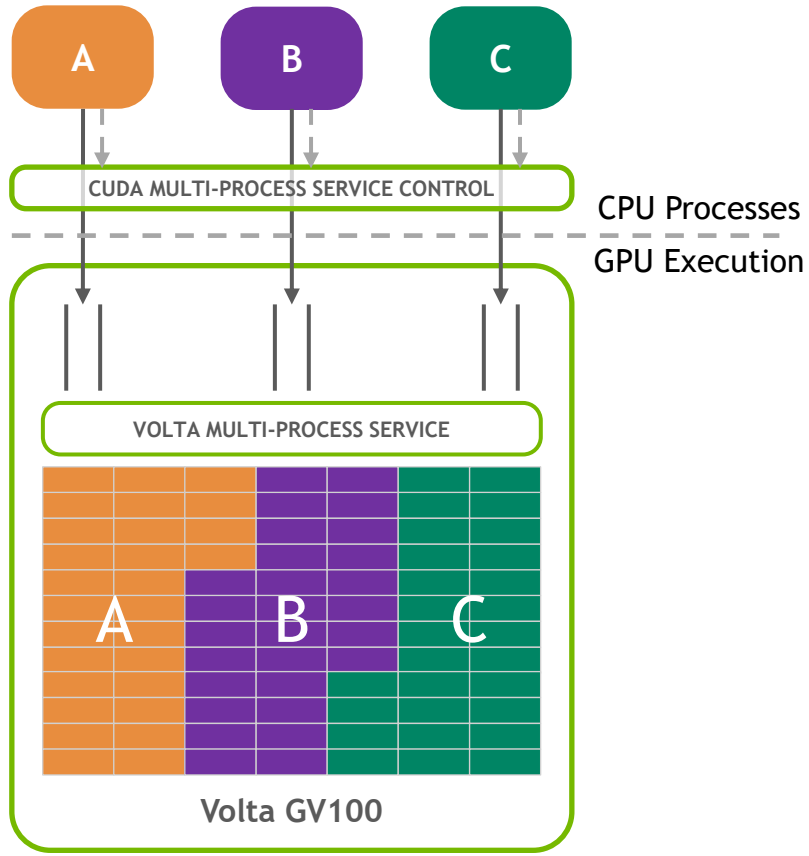
Flexible balance MPI ranks/GPU

Volta MPS Enhancements:

- Reduced launch latency
- Improved launch throughput
- Improved quality of service with scheduler partitioning
  - More reliable performance
- 3x more clients than Pascal
- https://docs.nvidia.com/deploy/mps/index.html

Hardware Accelerated Work Submission

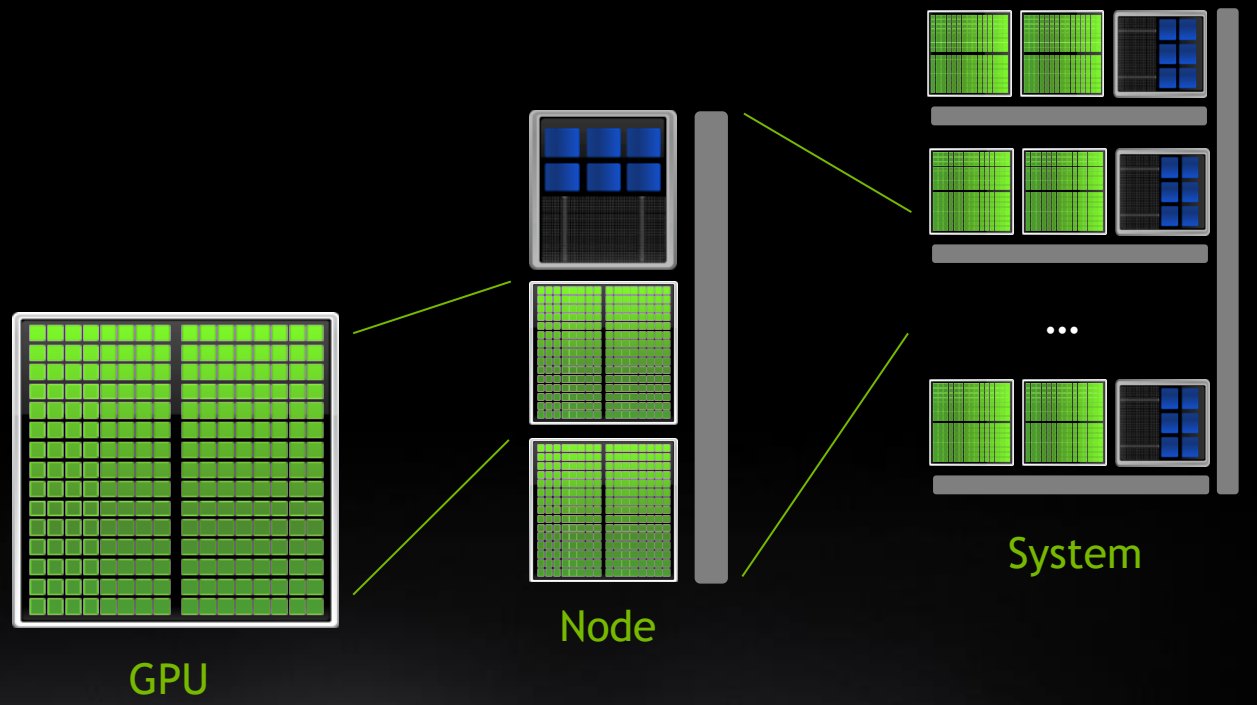
Hardware Isolation



OUTSIDE OF THE GPU

Accelerating at all scales

- PCIe3/4
- NVLink
- NVSwitch
- GPUDirect
  - Peer to peer
  - RDMA
  - Storage
- DPU - Bluefield



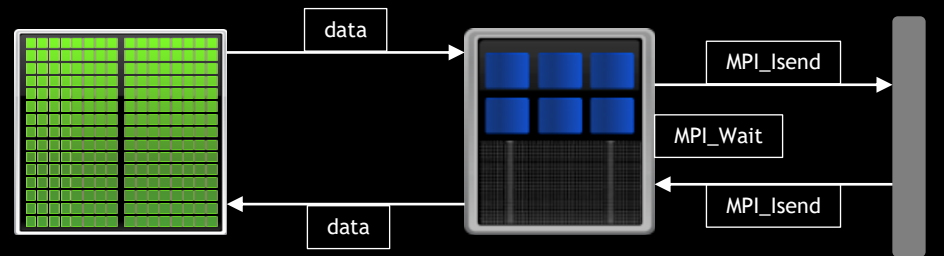
INTRODUCING NVSHMEM

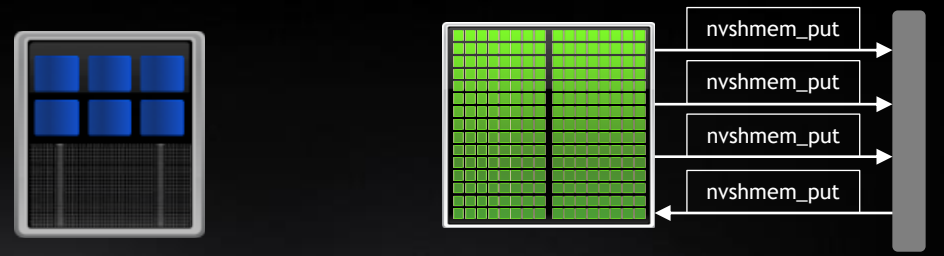
GPU Optimized OpenSHMEM

- Initiate from CPU or GPU
- Initiate from within CUDA kernel
- Issue onto a CUDA stream
- Interoperable with MPI & OpenSHMEM

Pre-release Impact

- LBANN, Kokkos/CGSolve, QUDA





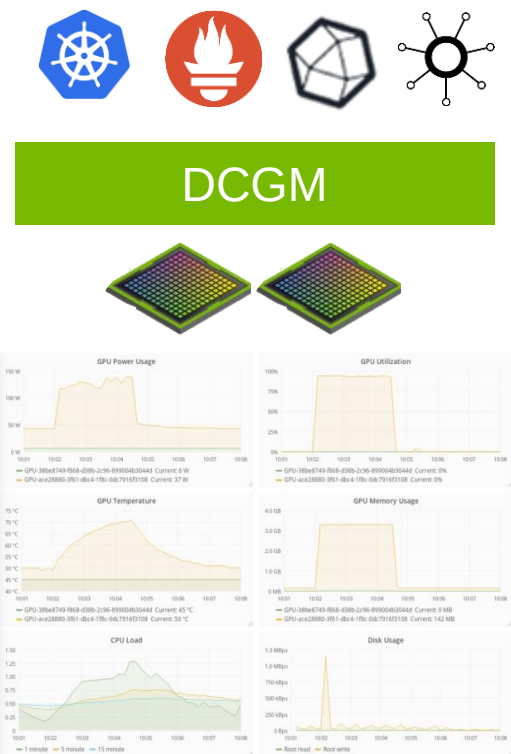
DEVELOPER TOOLS

TOOLS

Investigating and Monitoring Performance

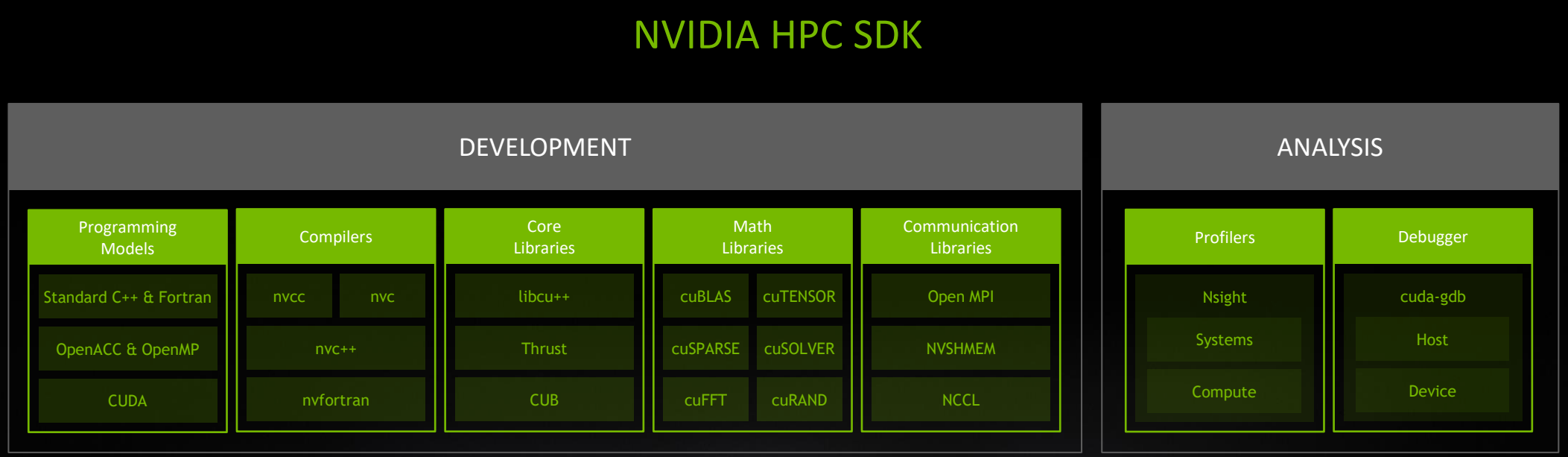
- Standalone Performance Tools
  - Nsight Systems system-wide application algorithm tuning
  - Nsight Compute Debug/optimize specific CUDA kernels
  - Nsight Graphics Debug/optimize specific graphics
- IDE plugins
  - Nsight Eclipse Edition/Visual Studio editor, debugger, some perf analysis

- Resource monitoring and admin



NVIDIA HPC SDK

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



Develop for the NVIDIA HPC Platform: GPU, CPU and Interconnect

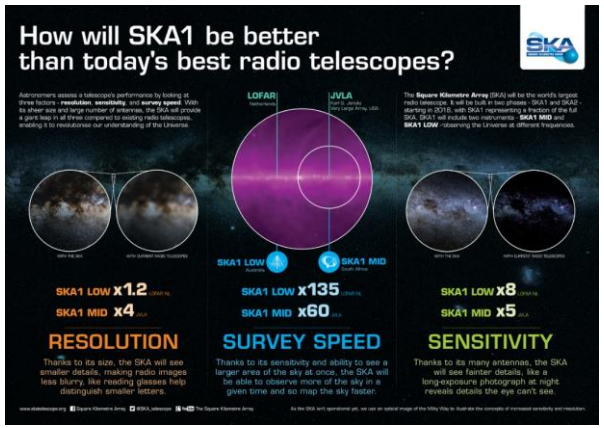
HPC Libraries | GPU Accelerated C++ and Fortran | Directives | CUDA

7-8 Releases Per Year | Freely Available

HPC & AI

HPC CHALLENGE 1

Fully Integrate New Experiments into the HPC Workflow

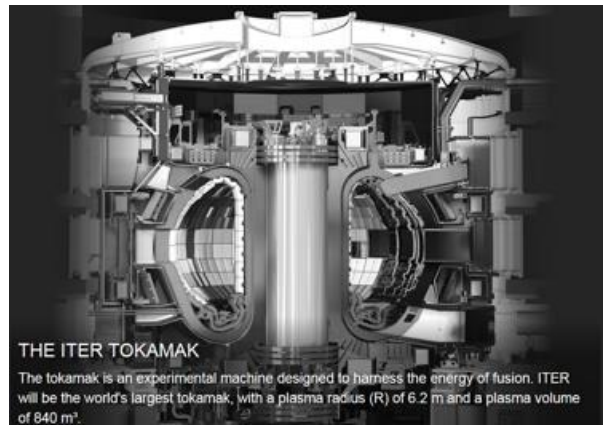


1EB/Day

SKA1 Square Kilometre Array radio telescope will generate more than an Exabyte of data every day.

10X

The CERN large Hadron collider's High Luminosity upgrade will result in a 10X increase in data volume.



30X

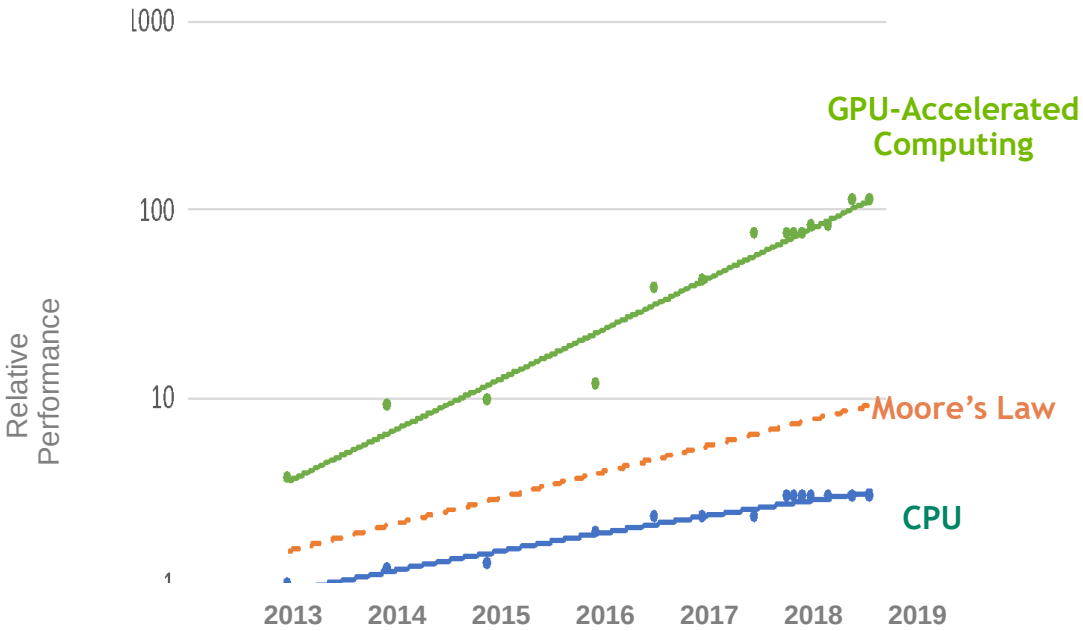
The 500 MW ITER fusion experiment will provide a 30X increase in output power over the largest previous experiment.

HPC CHALLENGE 2

CONVENTIONAL HPC BEYOND MOORE'S LAW

2014

- APPLICATIONS
- TOOLS
- COMPILERS
- ALGORITHMS
- LIBRARIES
- CUDA



Measured performance of Amber, CHROMA, GTC, LAMMPS, MILC, NAMD, Quantum Espresso, SPECFEM3D

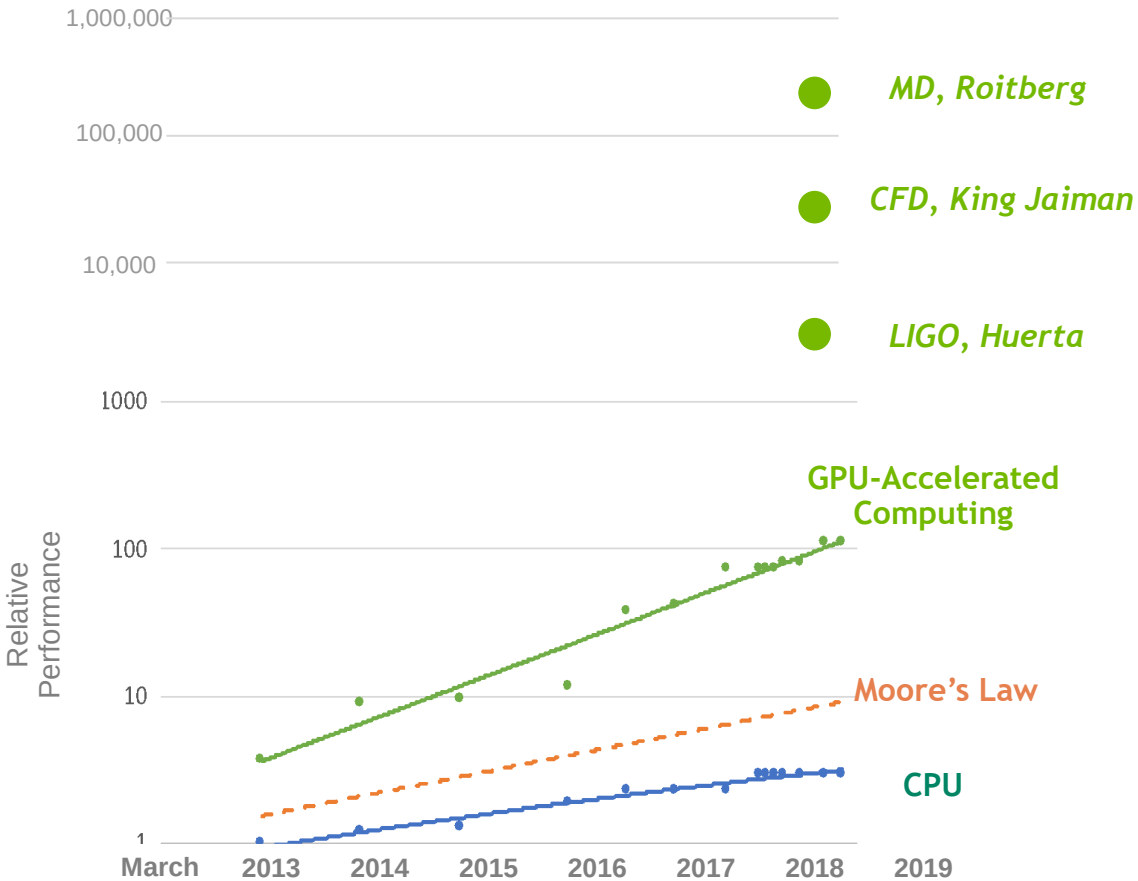
2019

- WORKFLOW TOOLS
- CONTAINERS
- APPLICATIONS
- TOOLS
- COMPILERS
- ALGORITHMS
- LIBRARIES
- CUDA

CONVERGED HPC*AI CHANGES THE GAME

2014

- APPLICATIONS
- TOOLS
- COMPILERS
- ALGORITHMS
- LIBRARIES
- CUDA



Measured performance of Amber, CHROMA, GTC, LAMMPS, MILC, NAMD, Quantum Espresso, SPECFEM3D

2019

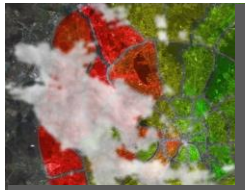
- WORKFLOW TOOLS
- CONTAINERS
- APPLICATIONS
- TOOLS
- COMPILERS
- ALGORITHMS
- LIBRARIES
- CUDA

CONVERGED HPC*AI TAXONOMY

How AI Algorithms are Being Applied in the HPC Workflow

Modelling and Simulation

Ab Initio Algorithm Enhancement

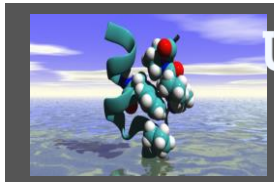


UCI University of California, Irvine

COLUMBIA UNIVERSITY

20X Faster Time to Solution

Reduced Order Model Replacement

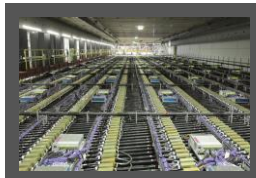


UF UNIVERSITY of FLORIDA

300,000X Faster
Time to solution

Experiment Data Processing

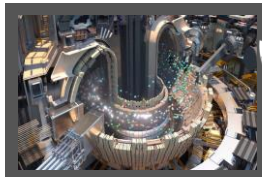
Detection Pipeline



U.S. DEPARTMENT OF ENERGY

49% More Accurate Detection

Real-Time Control



PRINCETON UNIVERSITY

iter

95% prediction accuracy with
5% False positives

RESOURCES

DEVELOPER ENGAGEMENT PLATFORMS

Information, downloads, special programs, code samples, and bug submission	developer.nvidia.com
Containers for cloud and workstation environments	ngc.nvidia.com
Insights & help from other developers and NVIDIA technical staff	devtalk.nvidia.com
Technical documentation	docs.nvidia.com
Deep Learning Institute: workshops & self-paced courses	courses.nvidia.com
In depth technical how to blogs	devblogs.nvidia.com
Developer focused news and articles	news.developer.nvidia.com
Webinars	nvidia.com/webinar-portal
GTC on-demand content	https://www.nvidia.com/on-demand/

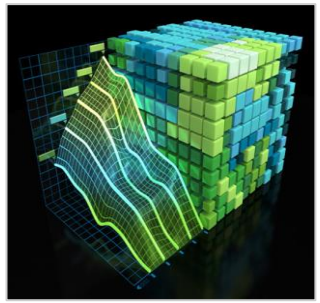
DEEP LEARNING INSTITUTE (DLI)

Hands-on, self-paced and instructor-led training in deep learning and accelerated computing

Request onsite instructor-led workshops at your organization:
www.nvidia.com/requestdli

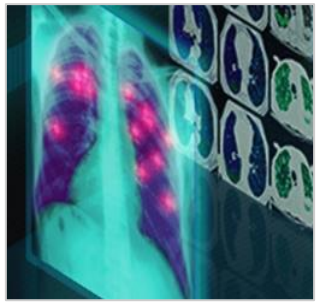
Take self-paced courses online:
www.nvidia.com/dlilabs

Download the course catalog, view upcoming workshops, and learn about the University Ambassador Program:
www.nvidia.com/dli

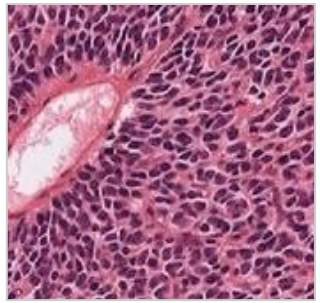


Accel. Computing Fundamentals

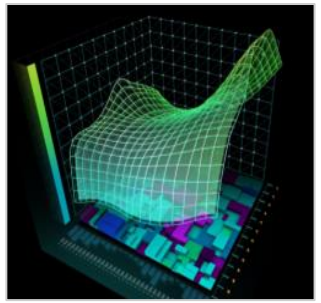
Autonomous Vehicles



Medical Image Analysis



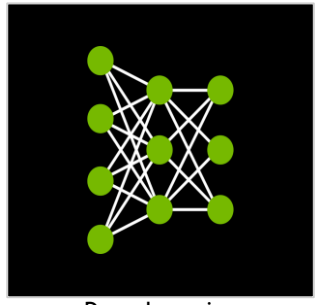
Genomics



Finance

Digital Content Creation

Game Development



Deep Learning Fundamentals

More industry-specific training coming soon…

RESOURCES AVAILABLE TO ACADEMICS

Developer Teaching Kits: which include free access to online training for students but they have to be requested by a lecturer/professor.

Academic Workshops:
The NVIDIA website lists free academic workshops that our Ambassadors are giving around the world that you can go and attend

Bootcamps:
~ 2 day tailored training events, typically for a target group e.g. OpenACC, AI for Science

Hackathons:
In-depth events with access to NV *devtech*

NVIDIA GTC 21

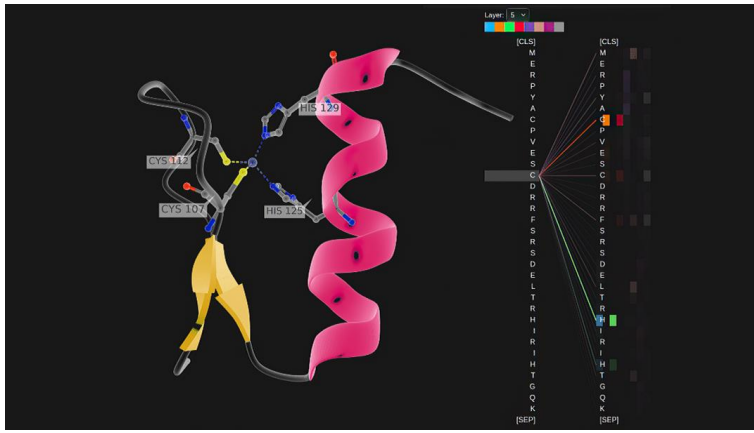
THE CONFERENCE FOR AI INNOVATORS, TECHNOLOGISTS, AND CREATIVES

Join us at GTC Fall 2021 on Nov 8 – 11 for the latest in AI, HPC, healthcare, game development, networking, and more.

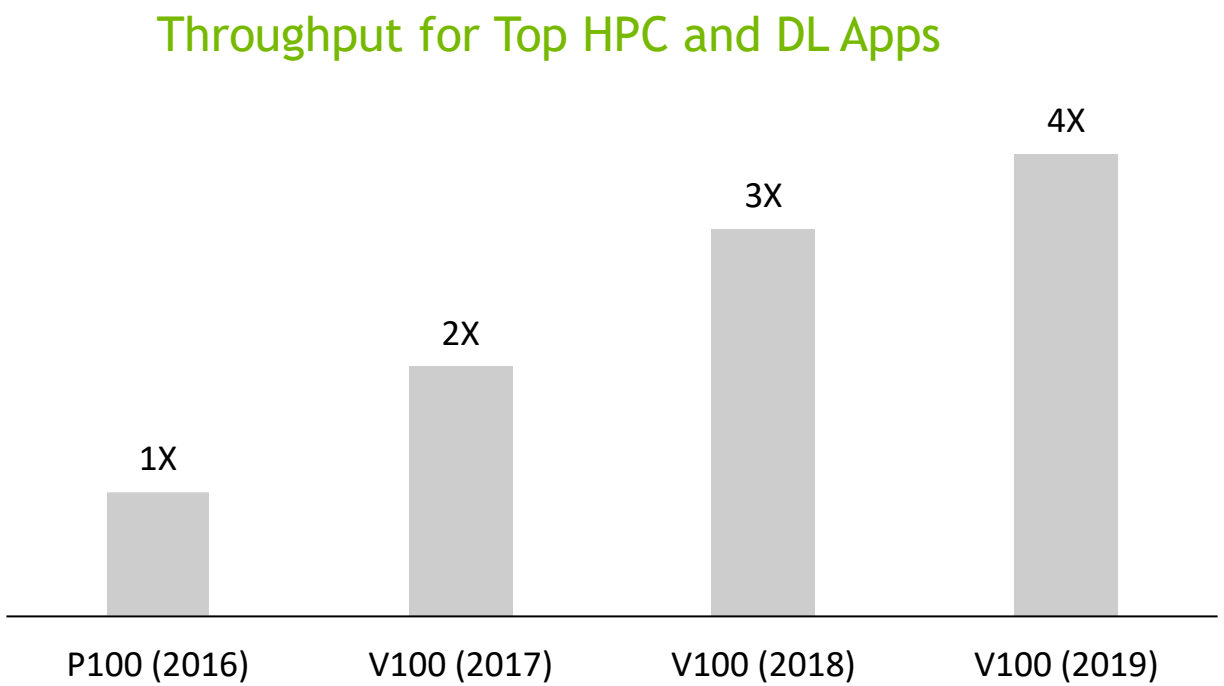
NVIDIA's GTC brings together a global community of developers, researchers, engineers, and innovators to experience global innovation and collaboration.

Don't miss out on the exclusive GTC keynote by Jensen Huang on **Nov 9**, available to everyone.

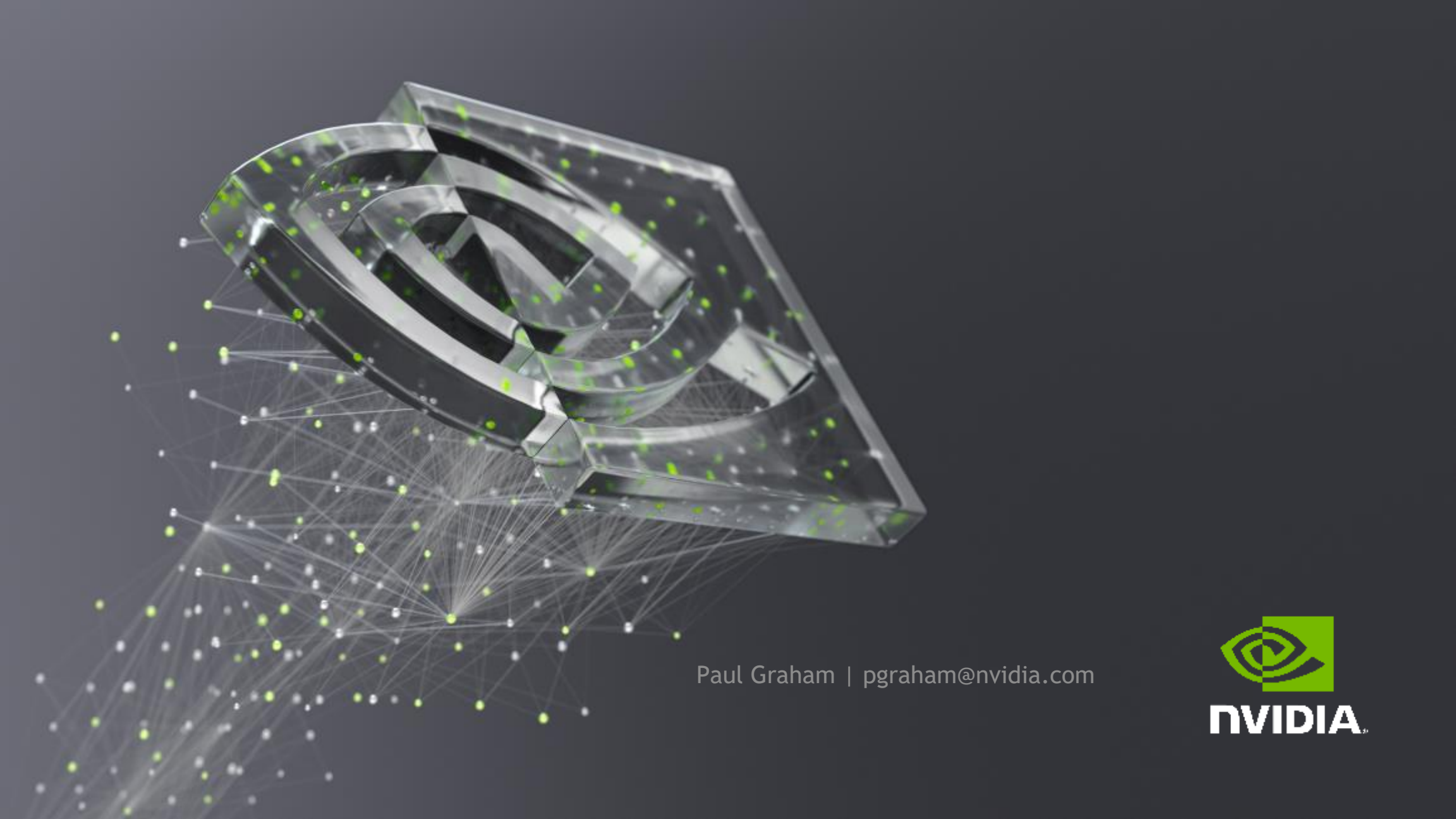
Visit https://www.nvidia.com/gtc to learn more and register for free



KEEP APPS, LIBRARIES AND FRAMEWORKS UP TO DATE



Geometric mean of application speedups vs. P100: Benchmark application: Amber [PME-Cellulose_NVE], Chroma [szscl21_24_128], GROMACS [ADH Dodec], MILC [Apex Medium], NAMD [stmv_nve_cuda], PyTorch (BERT-Large Fine Tuner], Quantum Espresso [AUSURF112-jR]; Random Forest FP32 [make_blobs (160000 x 64 : 10)], TensorFlow [ResNet-50], VASP 6 [Si Huge] | GPU node with dual-socket CPUs with 4x NVIDIA P100, V100, or A100 GPUs.



Paul Graham | pgraham@nvidia.com

NVIDIA