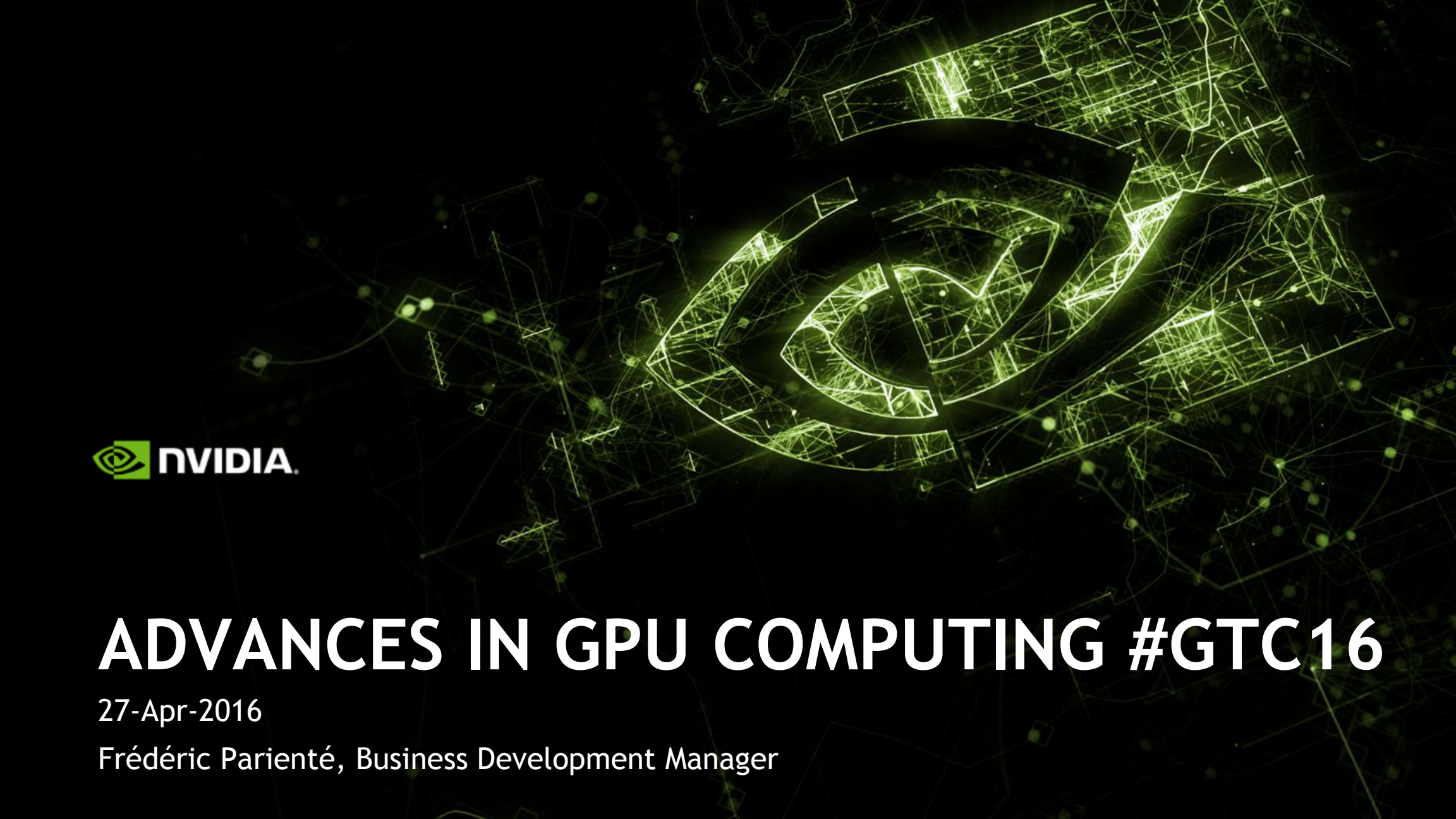


ADVANCES IN GPU COMPUTING #GTC16

27-Apr-2016

Frédéric Parienté, Business Development Manager

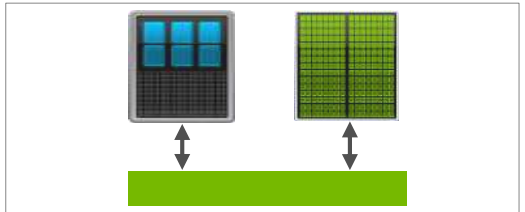
AGENDA

NVIDIA SDK

TESLA P100

NVIDIA DGX-1

NVGRAPH



CUDA 8 & UNIFIED MEMORY

OUR CRAFT

GPU-ACCELERATED COMPUTING

GRAPHICS

HIGH PERFORMANCE COMPUTING







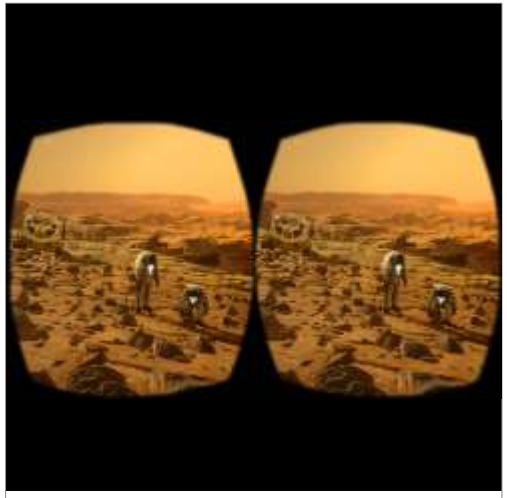




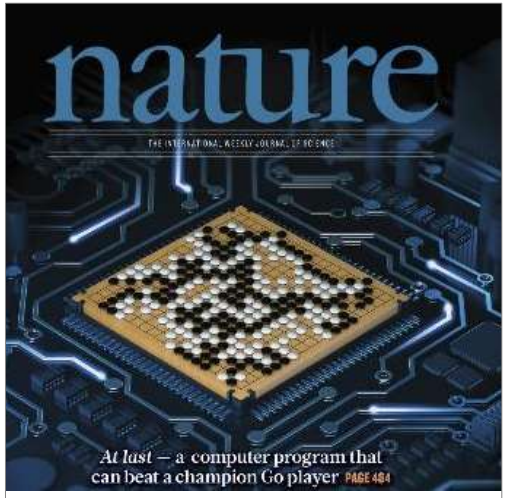



SELECT MARKETS

GAMING



PRO-VIZ



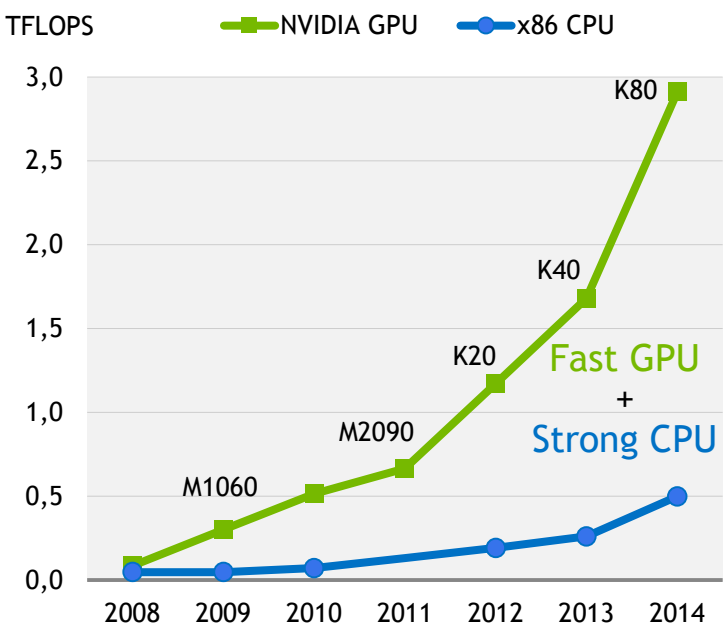
HPC & AI (DATACENTER)

AUTOMOTIVE (EMBEDDED)

TESLA ACCELERATED COMPUTING PLATFORM

Focused on Co-Design from Top to Bottom

Fast GPU Engineered for High Throughput



Productive Programming Model & Tools

NVIDIA CUDA

OpenACC
Directives for Accelerators

Expert Co-Design

- APPLICATION
- MIDDLEWARE
- SYS SW
- LARGE SYSTEMS
- PROCESSOR

Accessibility

阿里云计算 Alibaba Cloud Computing

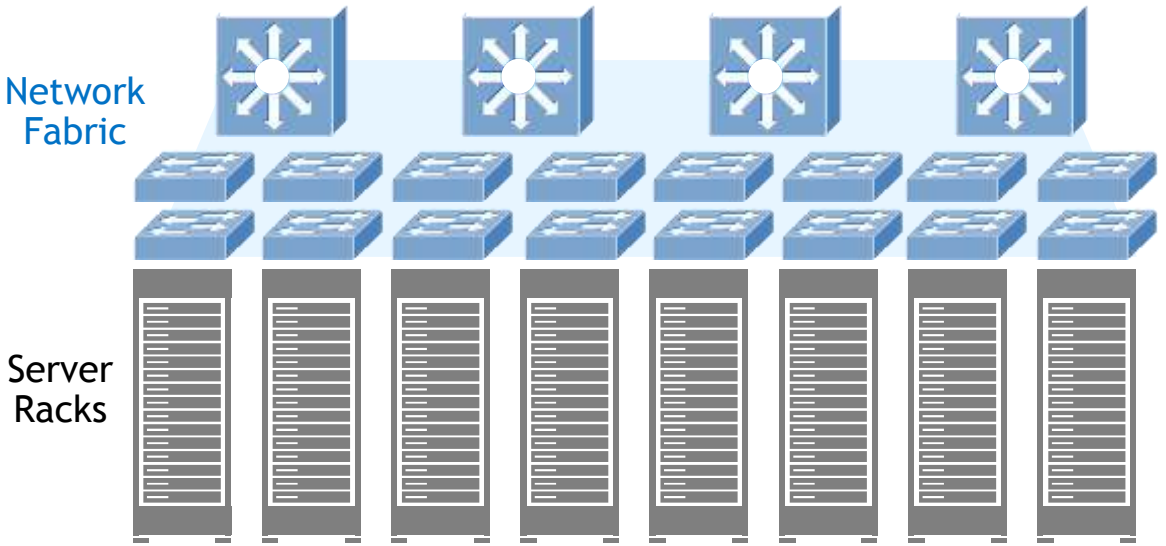
amazon web services · IBM SOFTLAYER

Microsoft Azure

NVIDIA

DATA CENTER TODAY

Well-suited For Transactional Workloads Running on Lots of Nodes



Commodity Computers Interconnected with Vast Network Overhead

THE DREAM

For Important Workloads with Infinite Need for Computing

Few Lightning-Fast Nodes with Performance of Thousands of Commodity Computers

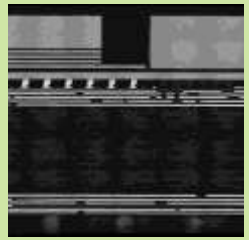
INTRODUCING TESLA P100

New GPU Architecture to Enable the World's Fastest Compute Node

Pascal Architecture

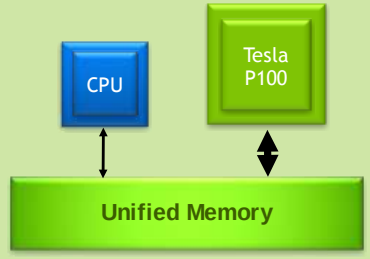
Highest Compute Performance

CoWoS HBM2



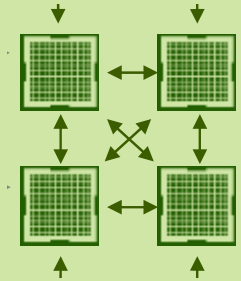
Unifying Compute & Memory in Single Package

Page Migration Engine



Simple Parallel Programming with Virtually Unlimited Memory

NVLink

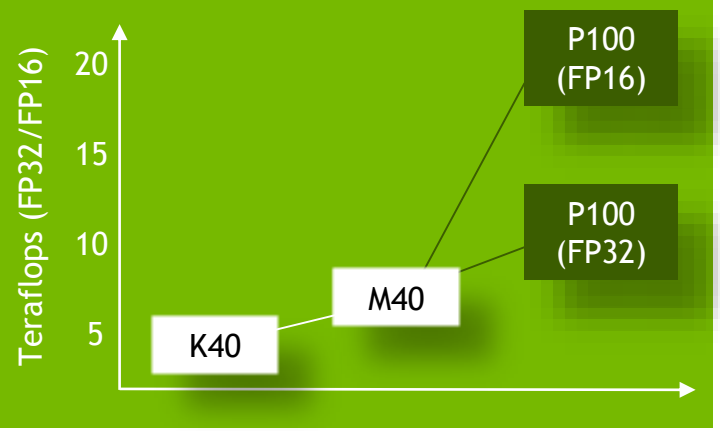


GPU Interconnect for Maximum Scalability

GIANT LEAPS IN EVERYTHING

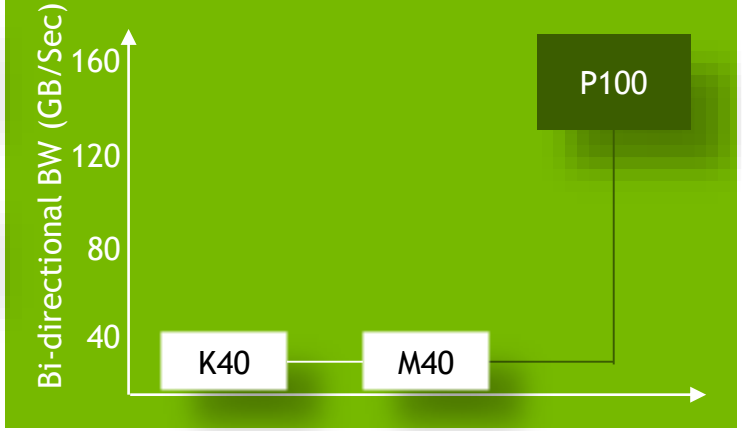
PASCAL ARCHITECTURE

21 Teraflops of FP16 for Deep Learning



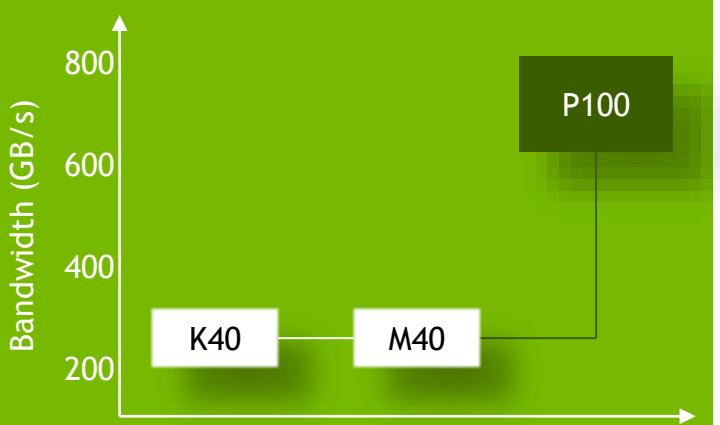
NVLINK

5x GPU-GPU Bandwidth



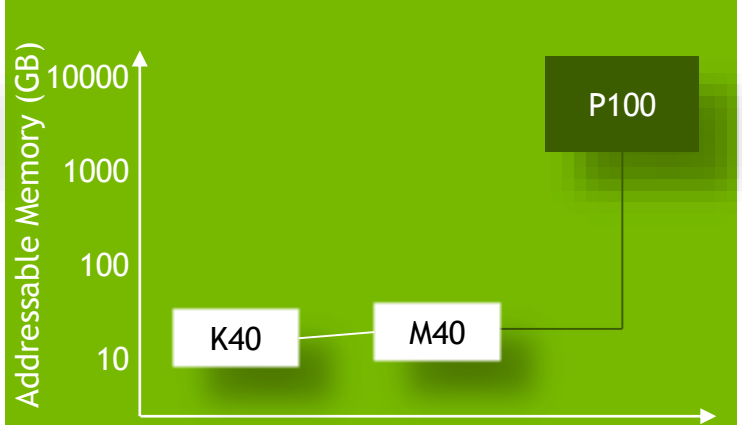
CoWoS HBM2 Stacked Mem

3x Higher for Massive Data Workloads



PAGE MIGRATION ENGINE

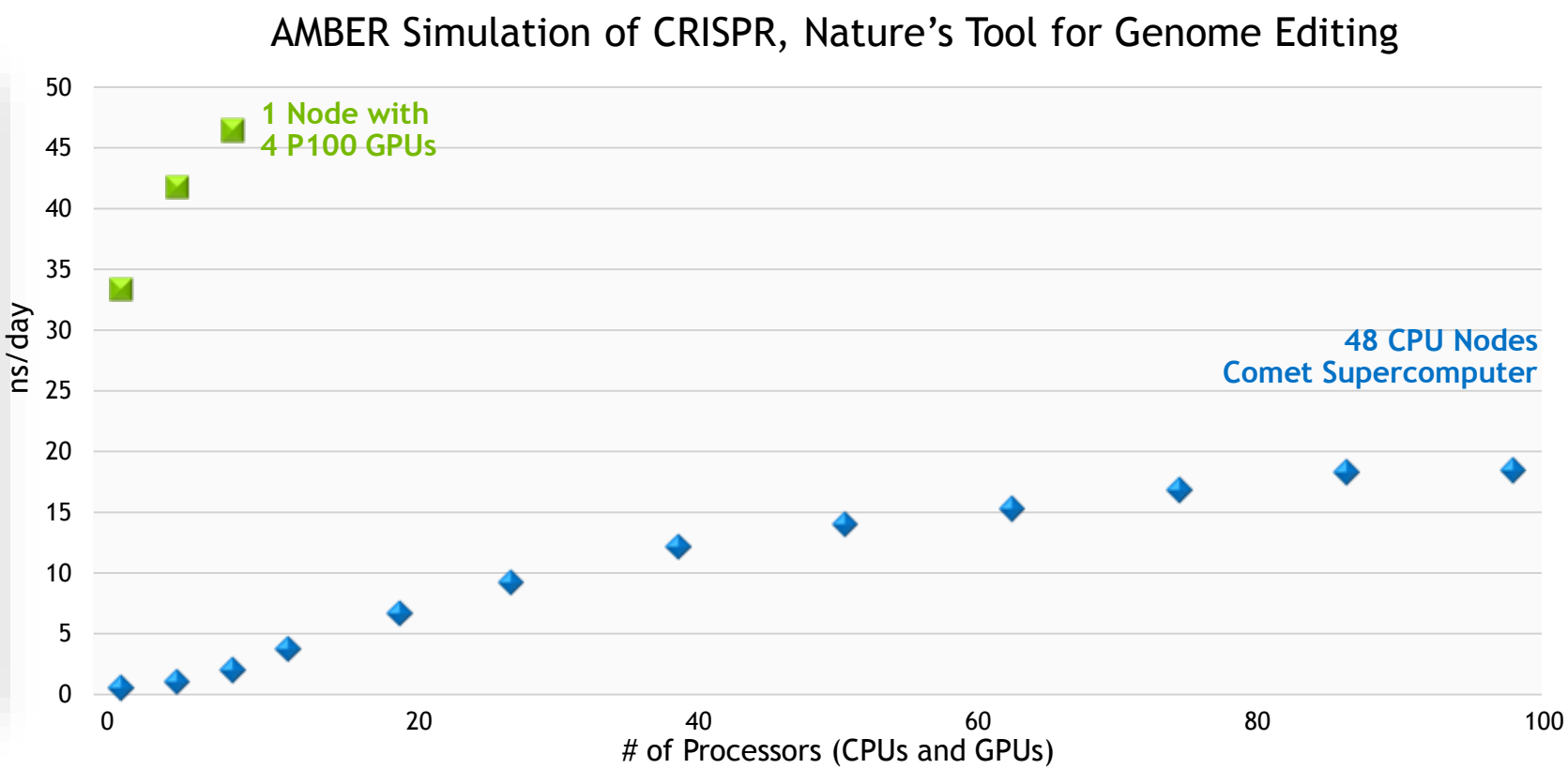
Virtually Unlimited Memory Space



BIG PROBLEMS NEED FAST COMPUTERS

2.5x Faster than the Largest CPU Data Center

"Biotech discovery of the century"
-MIT Technology Review 12/2014

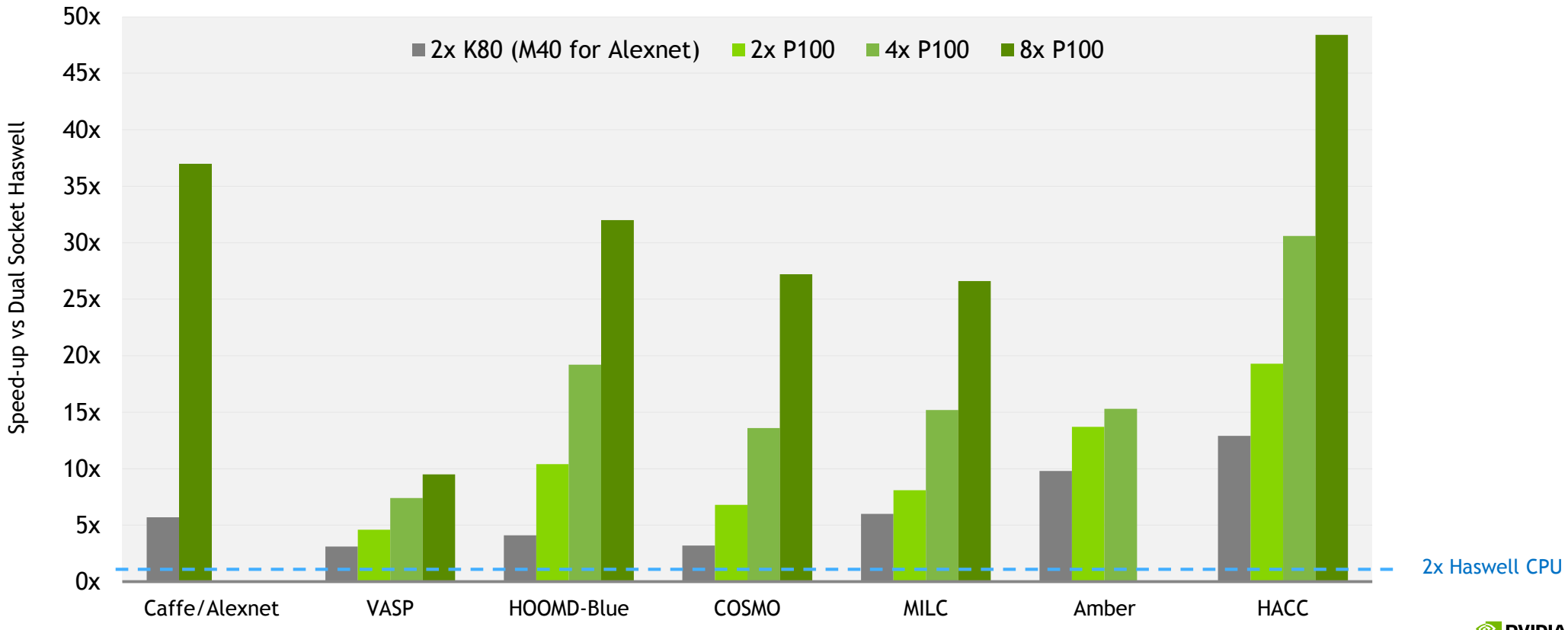


AMBER 16 Pre-release, CRSPR based on PDB ID 5f9r, 336,898 atoms
CPU: Dual Socket Intel E5-2680v3 12 cores, 128 GB DDR4 per node, FDR IB

NVIDIA

HIGHEST ABSOLUTE PERFORMANCE DELIVERED

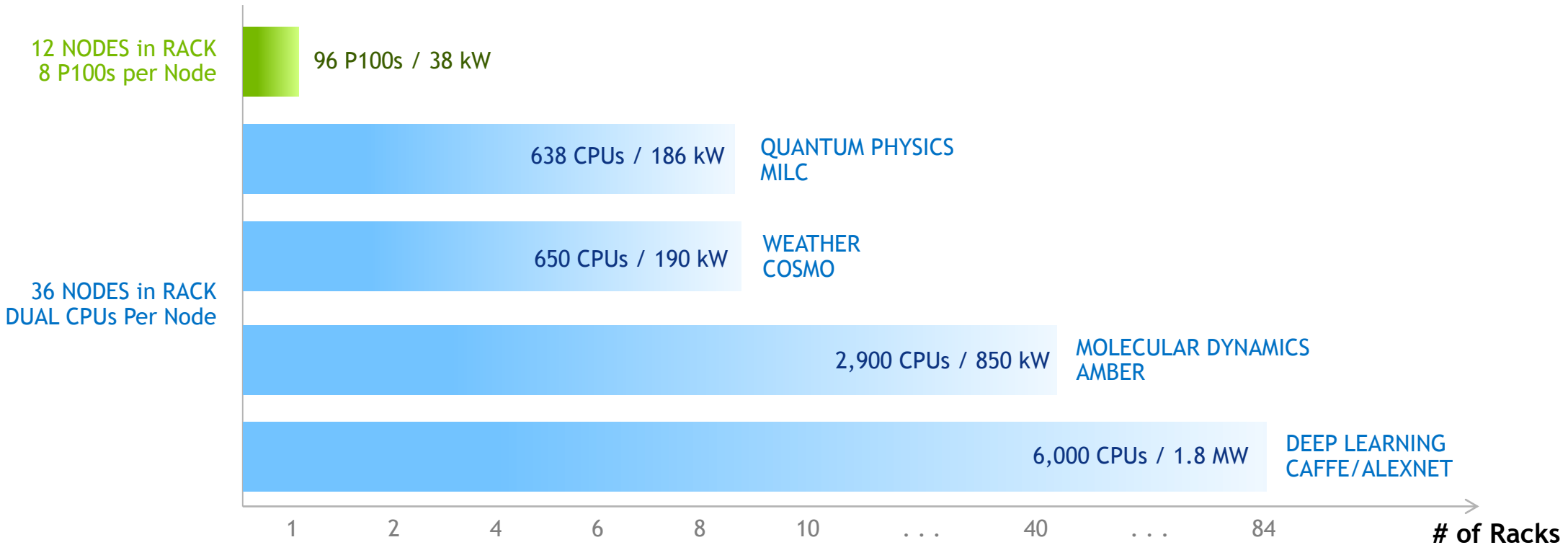
NVLink for Max Scalability, More than 45x Faster with 8x P100



NVIDIA

DATACENTER IN A RACK

1 Rack of Tesla P100 Delivers Performance of 6,000 CPUs



NVIDIA

TESLA P100 ACCELERATOR

	
Compute	5.3 TF DP · 10.6 TF SP · 21.2 TF HP
Memory	HBM2: 720 GB/s · 16 GB
Interconnect	NVLink (up to 8 way) + PCIe Gen3
Programmability	Page Migration Engine Unified Memory
Availability	DGX-1: Order Now Atos, Cray, Dell, HP, IBM: Q1 2017

NVIDIA DGX-1

WORLD'S FIRST DEEP LEARNING SUPERCOMPUTER

170 TFLOPS FP16

8x Tesla P100 16GB

NVLink Hybrid Cube Mesh

Accelerates Major AI Frameworks

Dual Xeon

7 TB SSD Deep Learning Cache

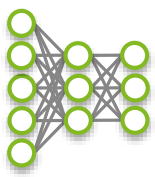
Dual 10GbE, Quad IB 100Gb

3RU – 3200W

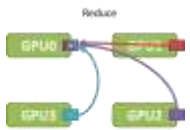
NVIDIA DGX-1 SOFTWARE STACK

Optimized for Deep Learning Performance

Accelerated Deep Learning



cuDNN



NCCL

cuSPARSE

cuBLAS

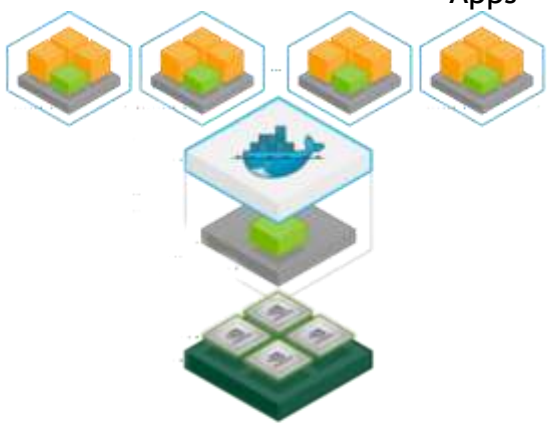
cuFFT

Container Based Applications

Digits

DL Frameworks

GPU Apps



NVIDIA Cloud Management

END-TO-END PRODUCT FAMILY

HYPERSCALE HPC

Tesla M4, M40

Hyperscale deployment for DL training, inference, video & image processing

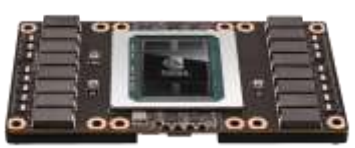
MIXED-APPS HPC

Tesla K80

HPC data centers running mix of CPU and GPU workloads

STRONG-SCALING HPC

Tesla P100



Hyperscale & HPC data centers running apps that scale to multiple GPUs

FULLY INTEGRATED DL SUPERCOMPUTER

DGX-1

For customers who need to get going now with fully integrated solution

NVIDIA SDK

DEEP LEARNING
Deep Learning SDK
High-performance tools and libraries for deep learning

SELF-DRIVING CARS
NVIDIA DriveWorks™
Deep learning, HD mapping and supercomputing solutions, from ADAS to fully autonomous

VIRTUAL REALITY
NVIDIA VRWorks™
A comprehensive SDK for VR headsets, games and professional applications

GAME DEVELOPMENT
NVIDIA GameWorks™
Advanced simulation and rendering technology for game development

ACCELERATED COMPUTING
NVIDIA ComputeWorks™
Everything scientists and engineers need to build GPU-accelerated applications

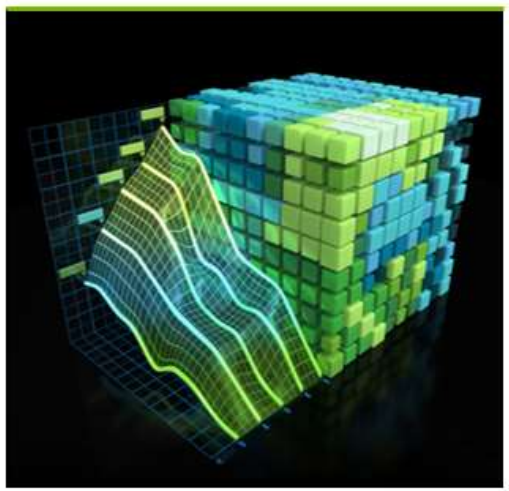
DESIGN & VISUALIZATION
NVIDIA DesignWorks™
Tools and technologies to create professional graphics and advanced rendering applications

AUTONOMOUS MACHINES
NVIDIA JetPack™
Powering breakthroughs in autonomous machines, robotics and embedded computing

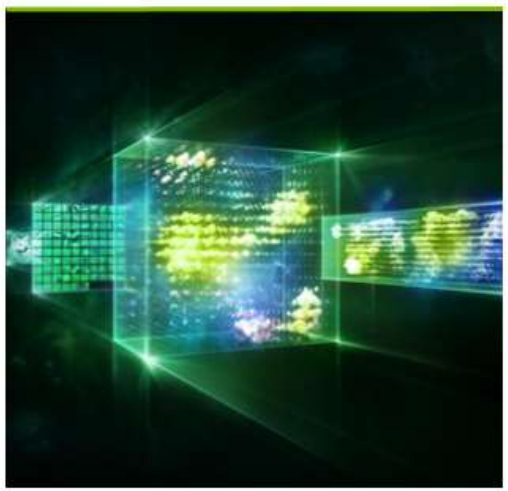
ADDITIONAL RESOURCES
More resources for GPU Developers

NVIDIA SDK: COMPUTEWORKS

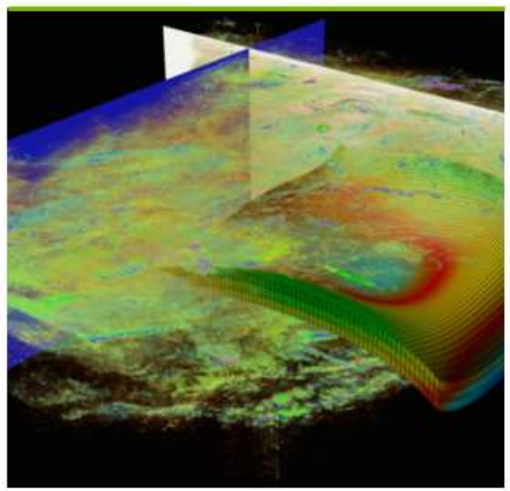
COMPUTEWORKS | GAMEWORKS | VRWORKS | DESIGNWORKS | DRIVEWORKS | JETPACK



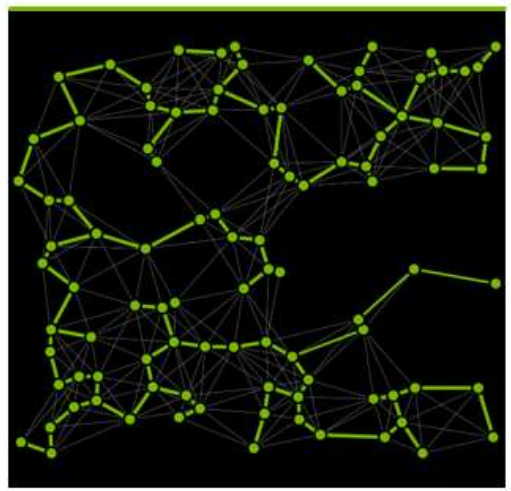
CUDA



cuDNN



IndeX



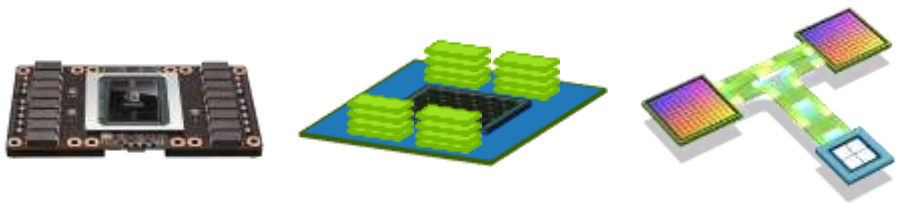
nvGRAPH

And other technologies such as:
AMGx, cuSOLVER, cuSPARSE, OpenACC, NSIGHT, THRUST

INTRODUCING CUDA 8

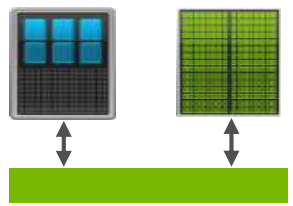
Pascal Support

New Architecture, Stacked Memory, NVLINK



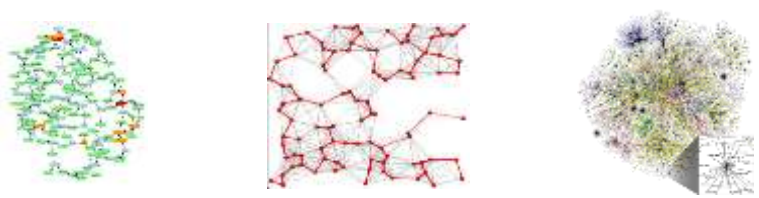
Unified Memory

Simple Parallel Programming with large virtual memory



Libraries

nvGRAPH – library for accelerating graph analytics apps
FP16 computation to boost Deep Learning workloads



Developer Tools

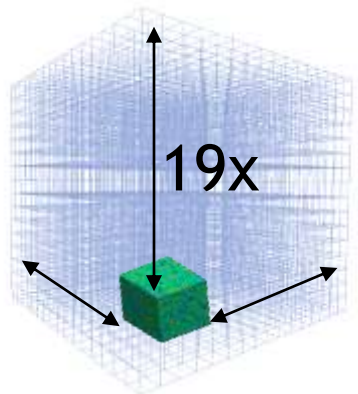
Critical Path Analysis to speed overall app tuning
OpenACC profiling to optimize directive performance
Single GPU debugging on Pascal

OUT-OF-THE-BOX PERFORMANCE ON PASCAL

P100/CUDA8 speedup over K80/CUDA7.5

3.5x

VASP



HPGMG with AMR

1.5x

MILC

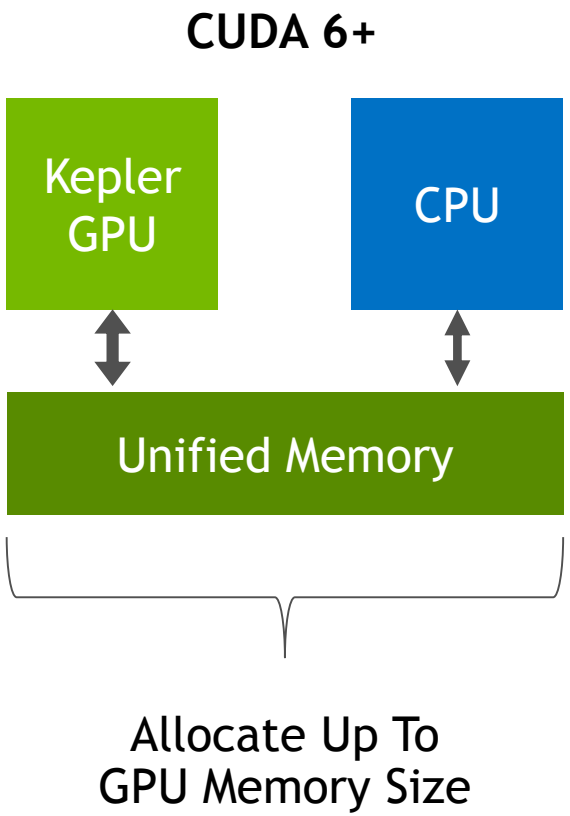
Larger Simulations &
More Accurate Results

developer.nvidia.com/cuda-toolkit

UNIFIED MEMORY

UNIFIED MEMORY

Dramatically Lower Developer Effort



Simpler Programming & Memory Model

- Single allocation, single pointer, accessible anywhere
- Eliminate need for *explicit copy*
- Greatly simplifies code porting

Performance Through Data Locality

- Migrate data to accessing processor
- Guarantee global coherence
- Still allows explicit hand tuning

SIMPLIFIED MEMORY MANAGEMENT CODE

CPU Code

```
void sortfile(FILE *fp, int N) {
  char *data;
  data = (char *)malloc(N);

  fread(data, 1, N, fp);

  qsort(data, N, 1, compare);


  use_data(data);

  free(data);
}
```

CUDA 6 Code with Unified Memory

```
void sortfile(FILE *fp, int N) {
  char *data;
  cudaMallocManaged(&data, N);

  fread(data, 1, N, fp);

  qsort<<<...>>>(data,N,1,compare);
  cudaDeviceSynchronize();

  use_data(data);

  cudaFree(data);
}
```

GREAT PERFORMANCE WITH UNIFIED MEMORY

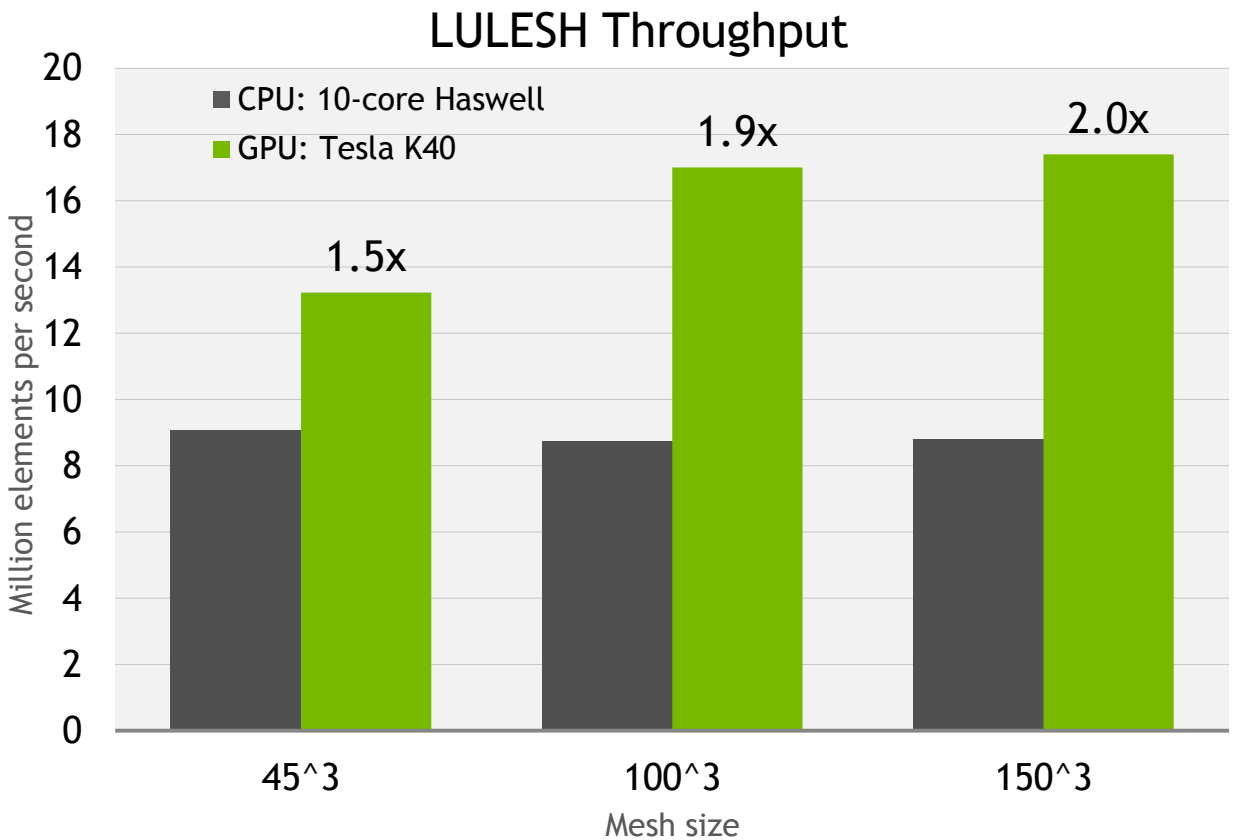
RAJA: Portable C++ Framework for parallel-for style programming

RAJA uses Unified Memory for heterogeneous array allocations

Parallel forall loops run on device

"Excellent performance considering this is a "generic" version of LULESH with no architecture-specific tuning."

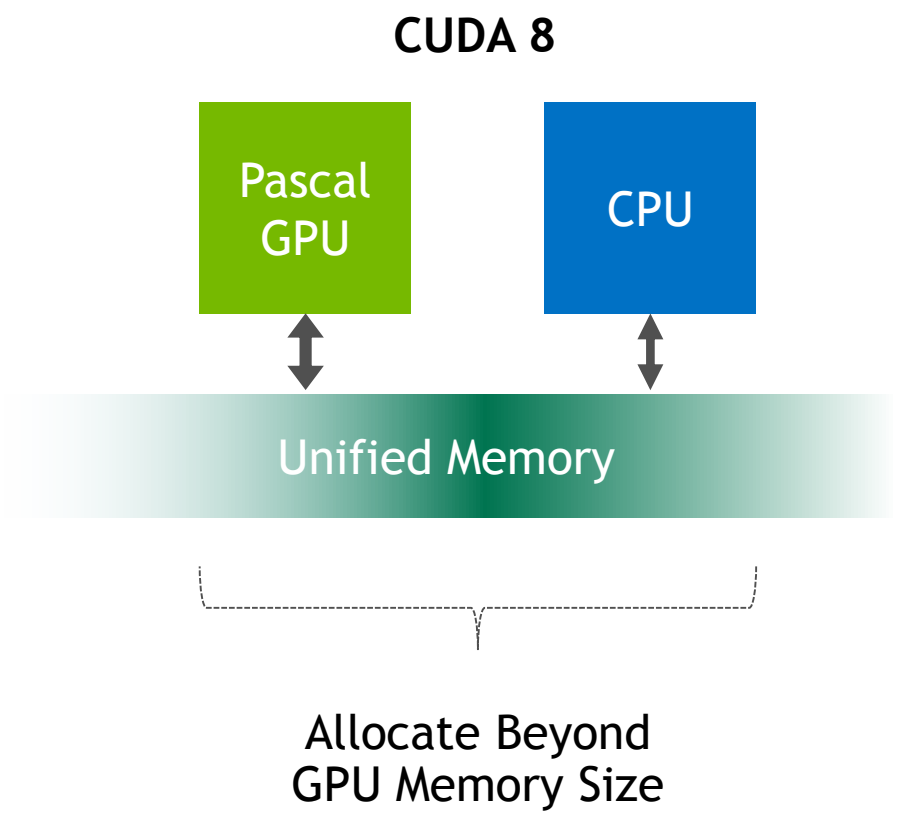
-Jeff Keasler, LLNL



GPU: NVIDIA Tesla K40, CPU: Intel Haswell E5-2650 v3 @ 2.30GHz, single socket 10-core

CUDA 8: UNIFIED MEMORY

Large datasets, simple programming, High Performance



Enable Large Data Models	Oversubscribe GPU memory Allocate up to system memory size
Simpler Data Accesss	CPU/GPU Data coherence Unified memory atomic operations
Tune Unified Memory Performance	Usage hints via cudaMemAdvise API Explicit prefetching API

UNIFIED MEMORY EXAMPLE

On-Demand Paging

```
__global__
void setValue(int *ptr, int index, int val)
{
  ptr[index] = val;
}


void foo(int size) {
  char *data;
  cudaMallocManaged(&data, size);

  memset(data, 0, size);

  setValue<<<...>>>(data, size/2, 5);
  cudaDeviceSynchronize();

  useData(data);

  cudaFree(data);
}
```

Unified Memory allocation

Access all values on CPU

Access one value on GPU

HOW UNIFIED MEMORY WORKS IN CUDA 6

Servicing CPU page faults

GPU Code

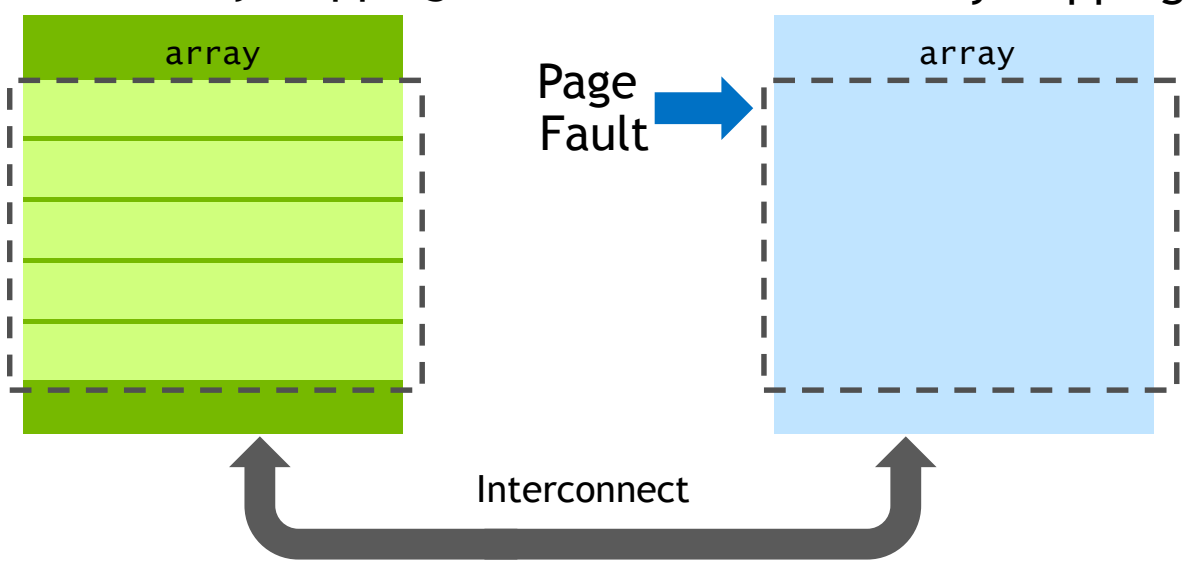
```
__global__
void setValue(char *ptr, int index, char val)
{
  ptr[index] = val;
}
```

CPU Code

```
cudaMallocManaged(&array, size);

memset(array, size);

setValue<<<...>>>(array, size/2, 5);
```

GPU Memory Mapping

CPU Memory Mapping



HOW UNIFIED MEMORY WORKS ON PASCAL

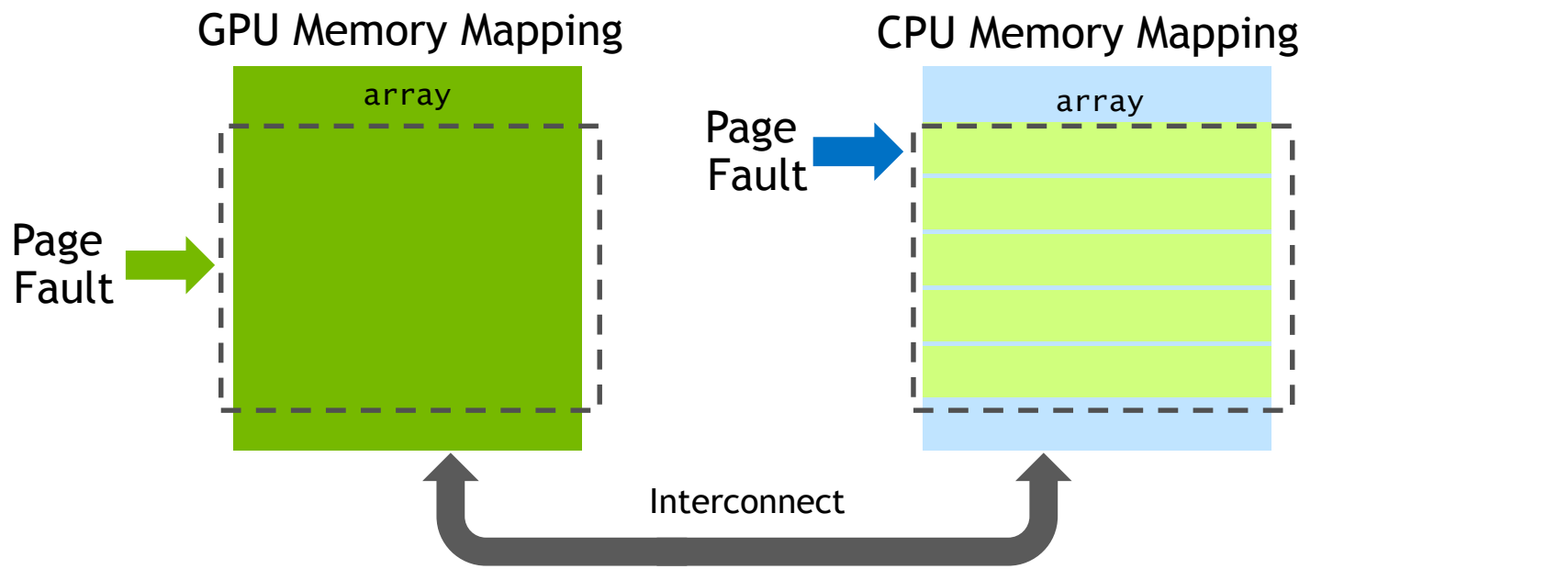
Servicing CPU *and* GPU Page Faults

GPU Code

```
__global__
Void setValue(char *ptr, int index, char val)
{
  ptr[index] = val;
}
```

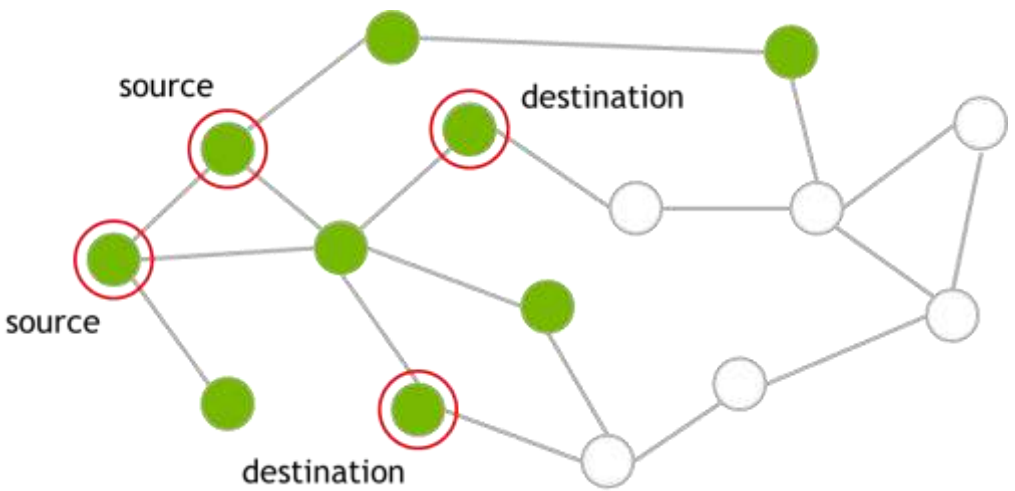
CPU Code

```
cudaMallocManaged(&array, size);

memset(array, size);

setValue<<<...>>>(array, size/2, 5);
```

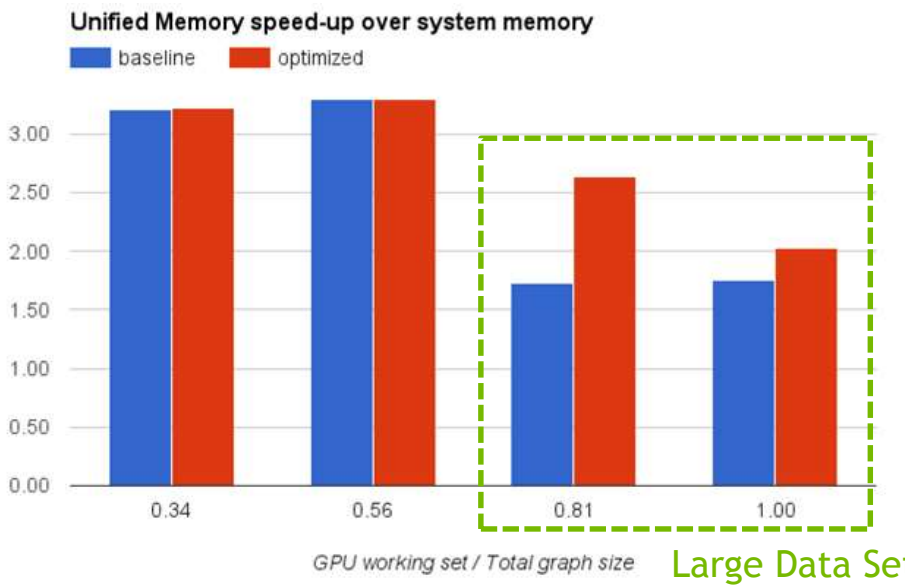


USE CASE: ON-DEMAND PAGING

Graph Algorithms



Performance over GPU directly accessing host memory (zero-copy)



Baseline: migrate on first touch
Optimized: best placement in memory

UNIFIED MEMORY ON PASCAL

GPU memory oversubscription

```
void foo() {
  // Assume GPU has 16 GB memory
  // Allocate 32 GB
  char *data;
  size_t size = 32*1024*1024*1024;
  cudaMallocManaged(&data, size);
}
```

32 GB allocation

Pascal supports allocations where only a subset of pages reside on GPU. Pages can be migrated to the GPU when “hot”.

Fails on Kepler/Maxwell

GPU OVERSUBSCRIPTION

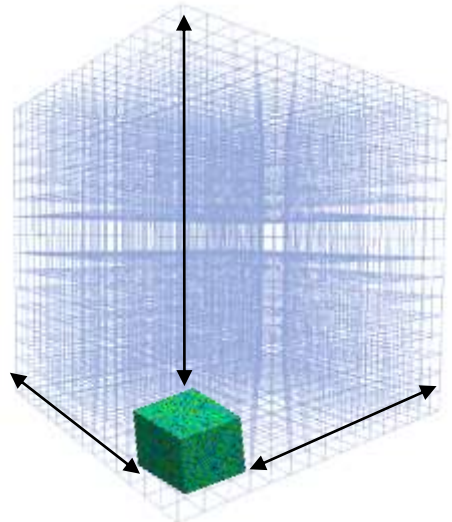
Now possible with Pascal

Many domains would benefit from GPU memory oversubscription:

Combustion – many species to solve for

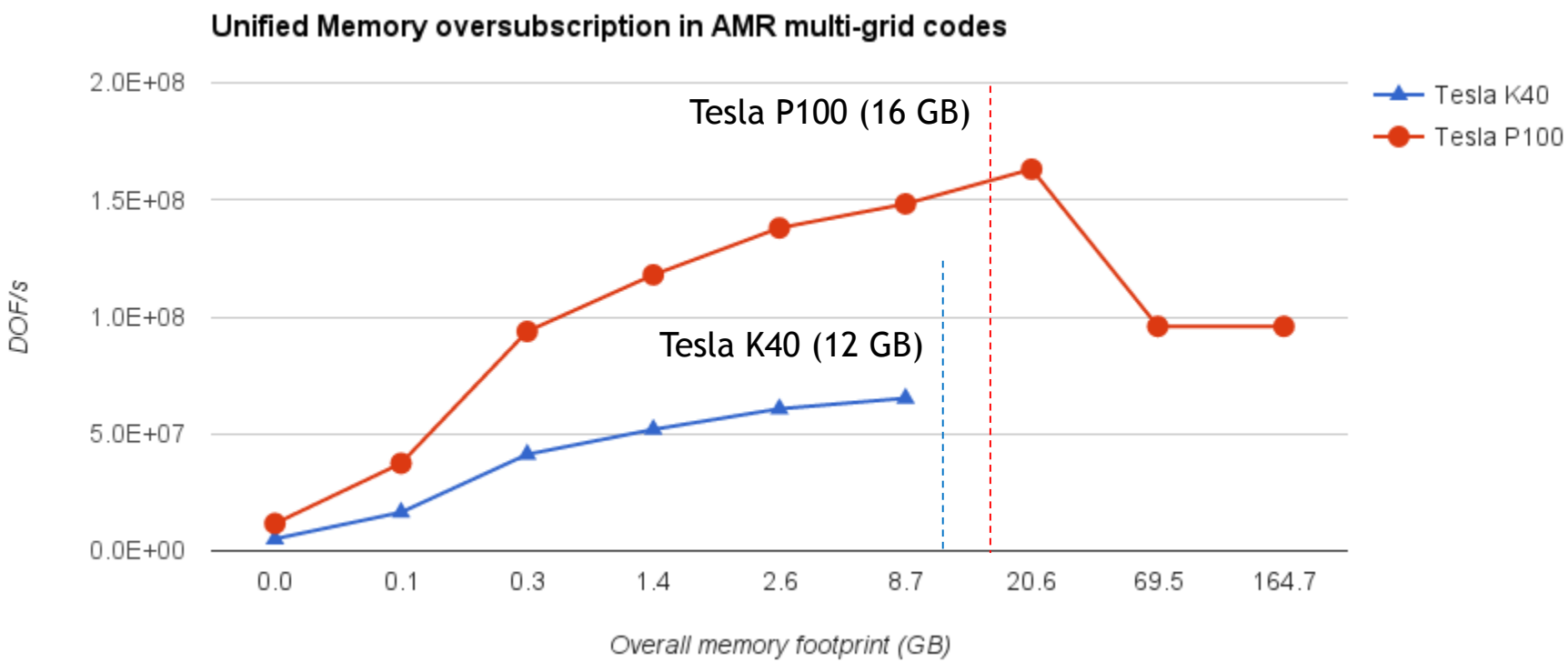
Quantum chemistry – larger systems

Ray tracing - larger scenes to render



GPU OVERSUBSCRIPTION

HPGMG: high-performance multi-grid



*Tesla P100 performance is very early modelling results

UNIFIED MEMORY ON PASCAL

Concurrent CPU/GPU access to managed memory

```
__global__ void mykernel(char *data) {
  data[1] = 'g';
}

void foo() {
  char *data;
  cudaMallocManaged(&data, 2);

  mykernel<<<...>>>(data);
  // no synchronize here
  data[0] = 'c';

  cudaFree(data);
}
```

OK on Pascal: just a page fault

Concurrent CPU access to 'data' on previous GPUs caused a fatal segmentation fault

UNIFIED MEMORY ON PASCAL

System-Wide Atomics

```
__global__ void mykernel(int *addr) {
  atomicAdd(addr, 10);
}

void foo() {
  int *addr;
  cudaMallocManaged(addr, 4);
  *addr = 0;

  mykernel<<<...>>>(addr);
  __sync_fetch_and_add(addr, 10);
}
```

Pascal enables system-wide atomics

- Direct support of atomics over NVLink
- Software-assisted over PCIe

System-wide atomics not available on Kepler / Maxwell

PERFORMANCE TUNING ON PASCAL

Explicit Memory Hints and Prefetching

Advise runtime on known memory access behaviors with `cudaMemAdvise()`

- `cudaMemAdviseSetReadMostly`: Specify read duplication
- `cudaMemAdviseSetPreferredLocation`: suggest best location
- `cudaMemAdviseSetAccessedBy`: initialize a mapping

Explicit prefetching with `cudaMemPrefetchAsync(ptr, length, destDevice, stream)`

- Unified Memory alternative to cudaMemcpyAsync
- Asynchronous operation that follows CUDA stream semantics

To Learn More:
S6216 “The Future of Unified Memory” by Nikolay Sakharnykh
at http://on-demand.gputechconf.com/

FUTURE: UNIFIED SYSTEM ALLOCATOR

Allocate unified memory using standard malloc

CUDA 8 Code with System Allocator

```
void sortfile(FILE *fp, int N) {
  char *data;

// Allocate memory using any standard allocator
  data = (char *) malloc(N * sizeof(char));

  fread(data, 1, N, fp);

  qsort<<<...>>>(data,N,1,compare);

  use_data(data);

// Free the allocated memory
  free(data);
}
```

Removes CUDA specific allocator restrictions

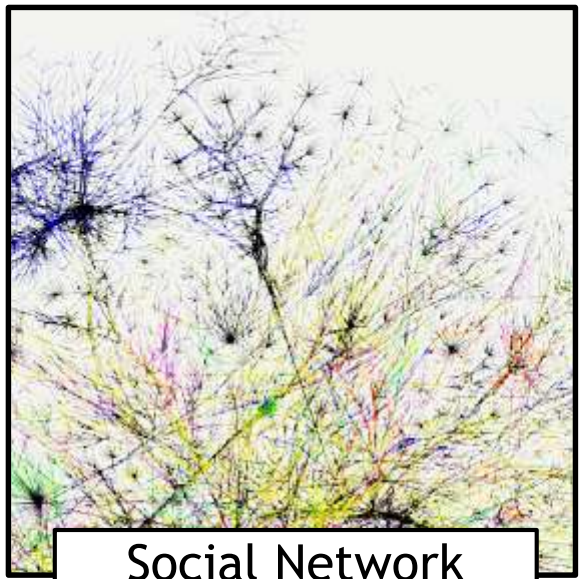
Data movement is transparently handled

Requires operating system support

GRAPH ANALYTICS

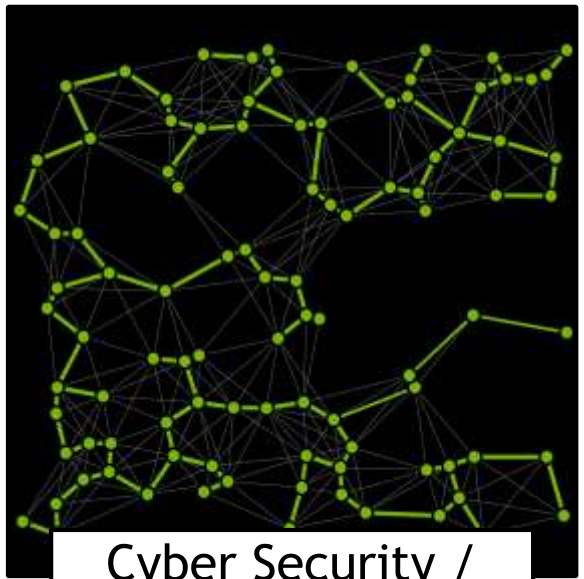
GRAPH ANALYTICS

Insight from Connections in Big Data

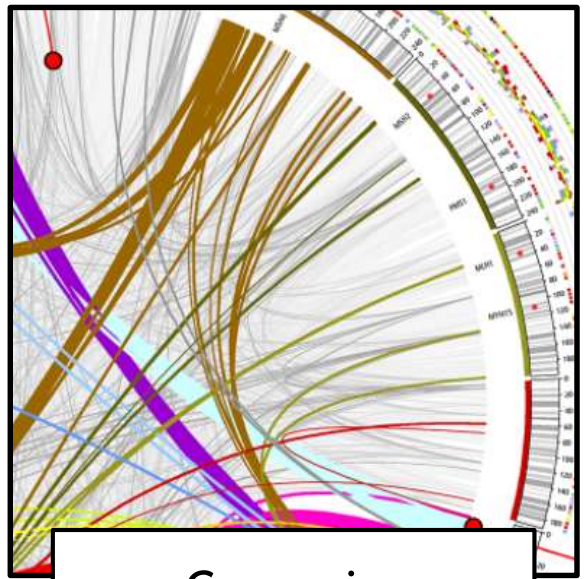


Wikimedia Commons

Social Network Analysis



Cyber Security / Network Analytics



Circos.ca

Genomics

… and much more: Parallel Computing, Recommender Systems, Fraud Detection, Voice Recognition, Text Understanding, Search

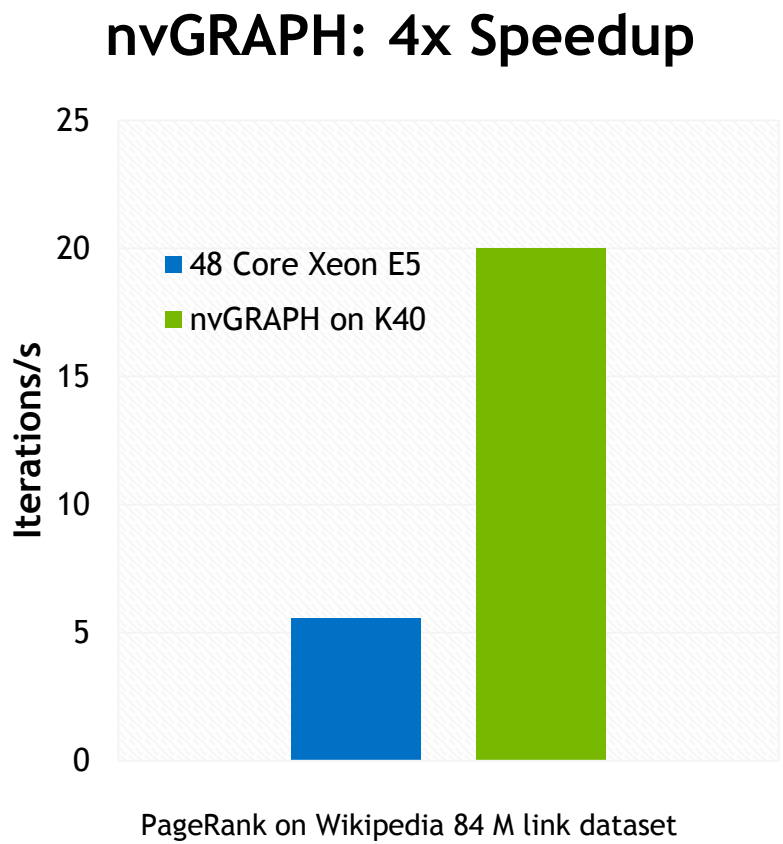
nvGRAPH

Accelerated Graph Analytics

Process graphs with up to 2.5 Billion edges on a single GPU (24GB M40)

Accelerate a wide range of applications:

PageRank	Single Source Shortest Path	Single Source Widest Path
Search	Robotic Path Planning	IP Routing
Recommendation Engines	Power Network Planning	Chip Design / EDA
Social Ad Placement	Logistics & Supply Chain Planning	Traffic sensitive routing



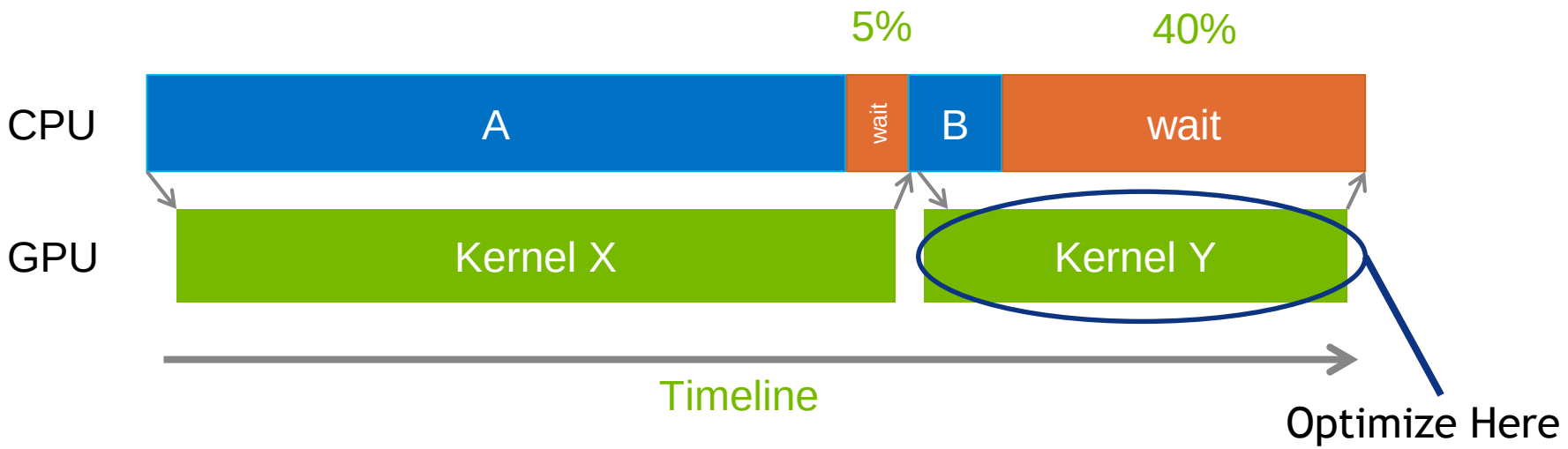
developer.nvidia.com/nvgraph

NVIDIA

ENHANCED PROFILING

DEPENDENCY ANALYSIS

Easily Find the Critical Kernel To Optimize



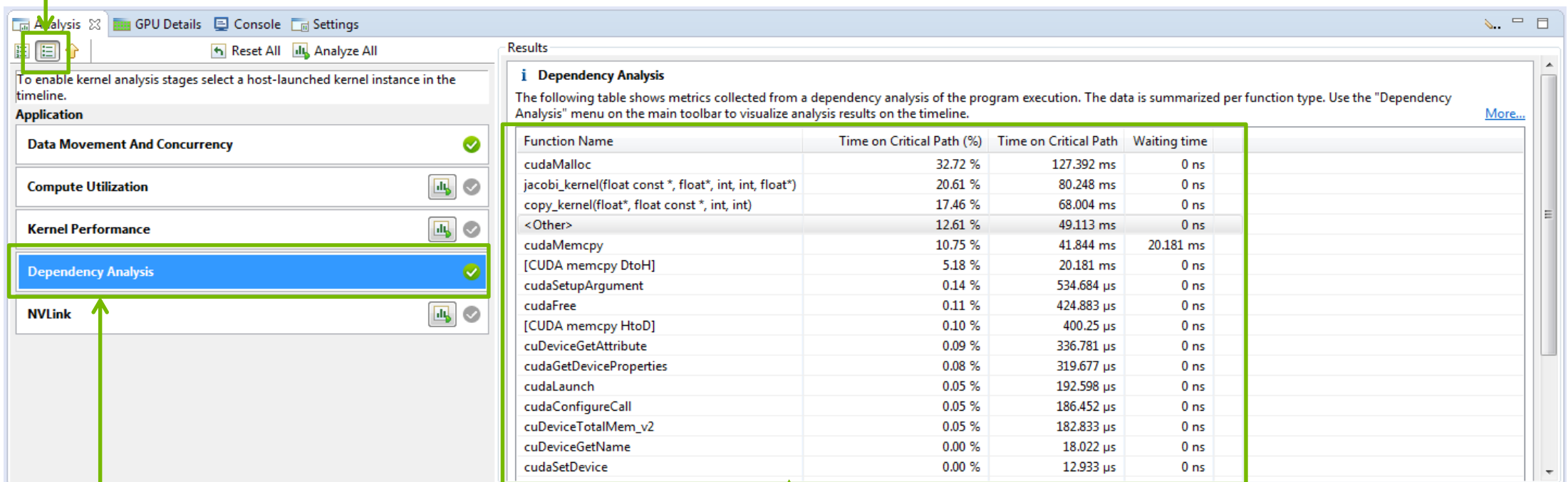
The longest running kernel is not always the most critical optimization target

DEPENDENCY ANALYSIS

Visual Profiler

Unguided Analysis

Generating critical path

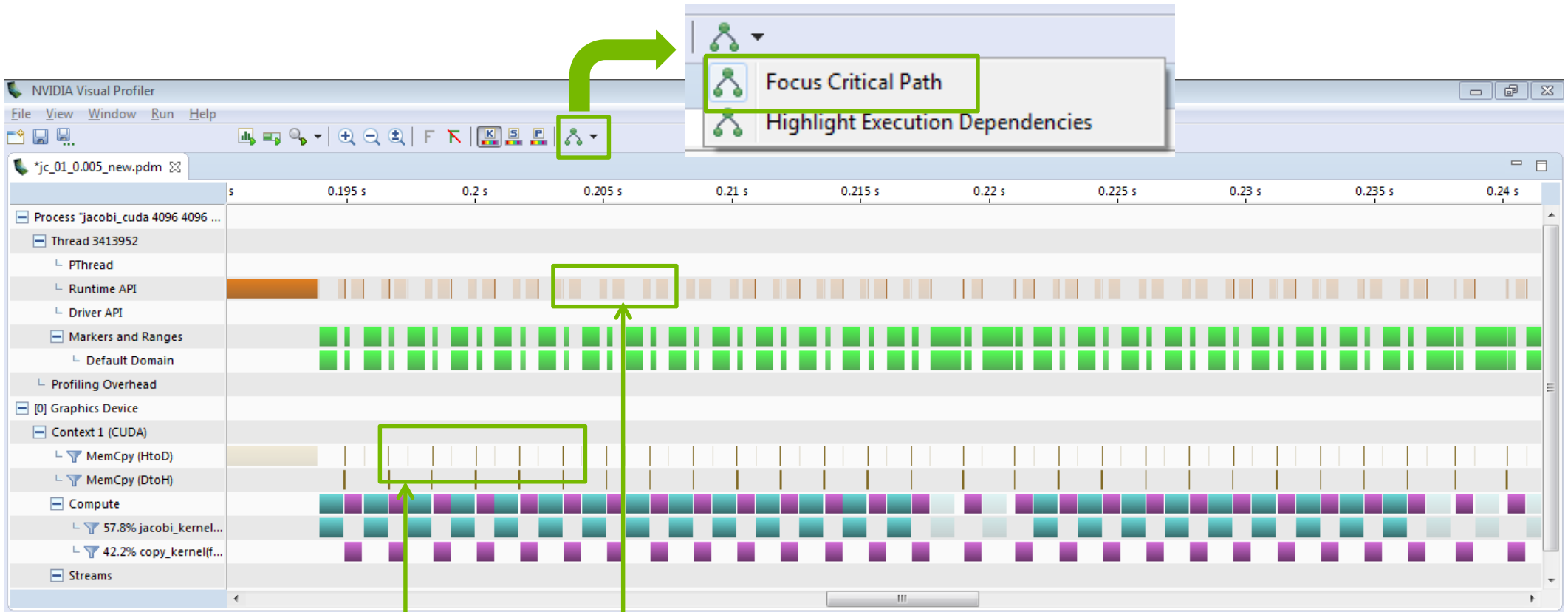


Dependency Analysis

Functions on critical path

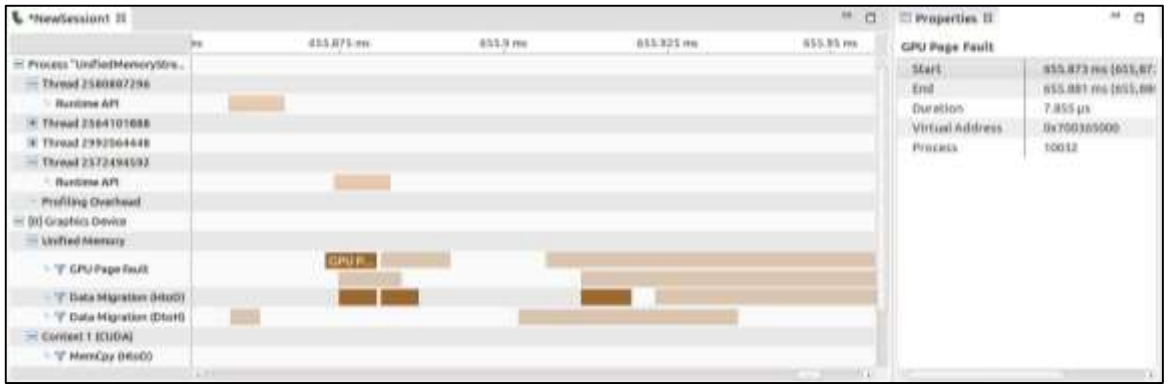
DEPENDENCY ANALYSIS

Visual Profiler

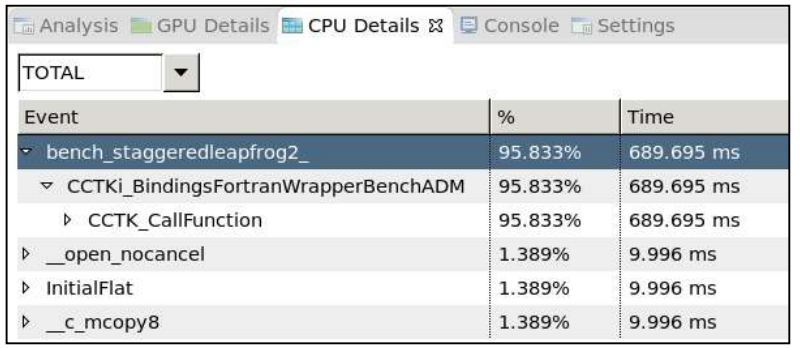


APIs, GPU activities not in critical path are greyed out

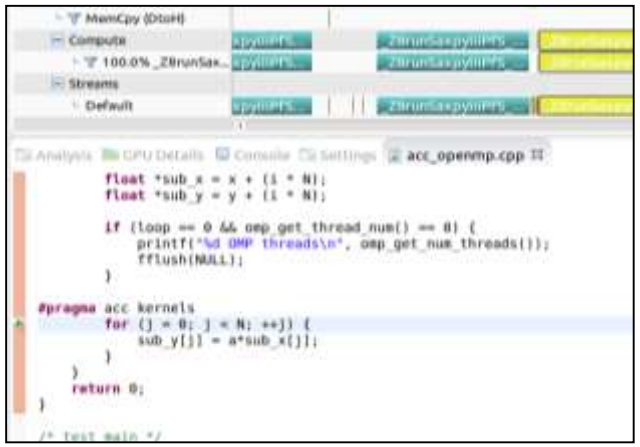
MORE CUDA 8 PROFILER FEATURES



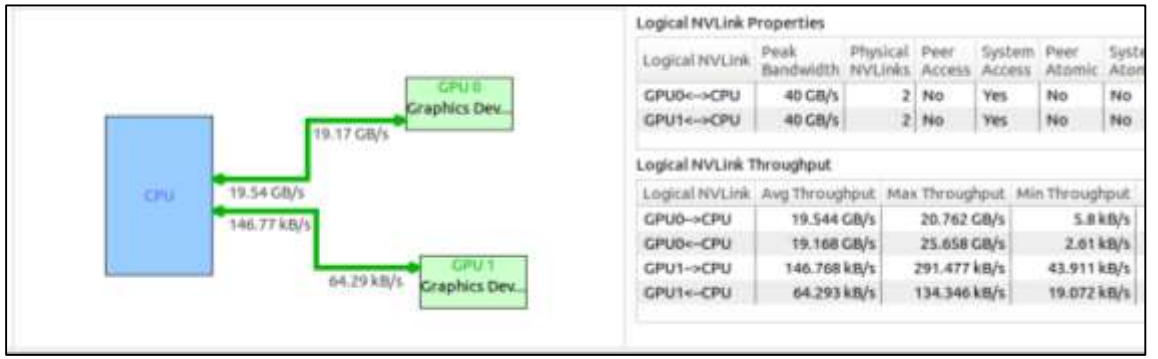
Unified Memory Profiling



CPU Profiling



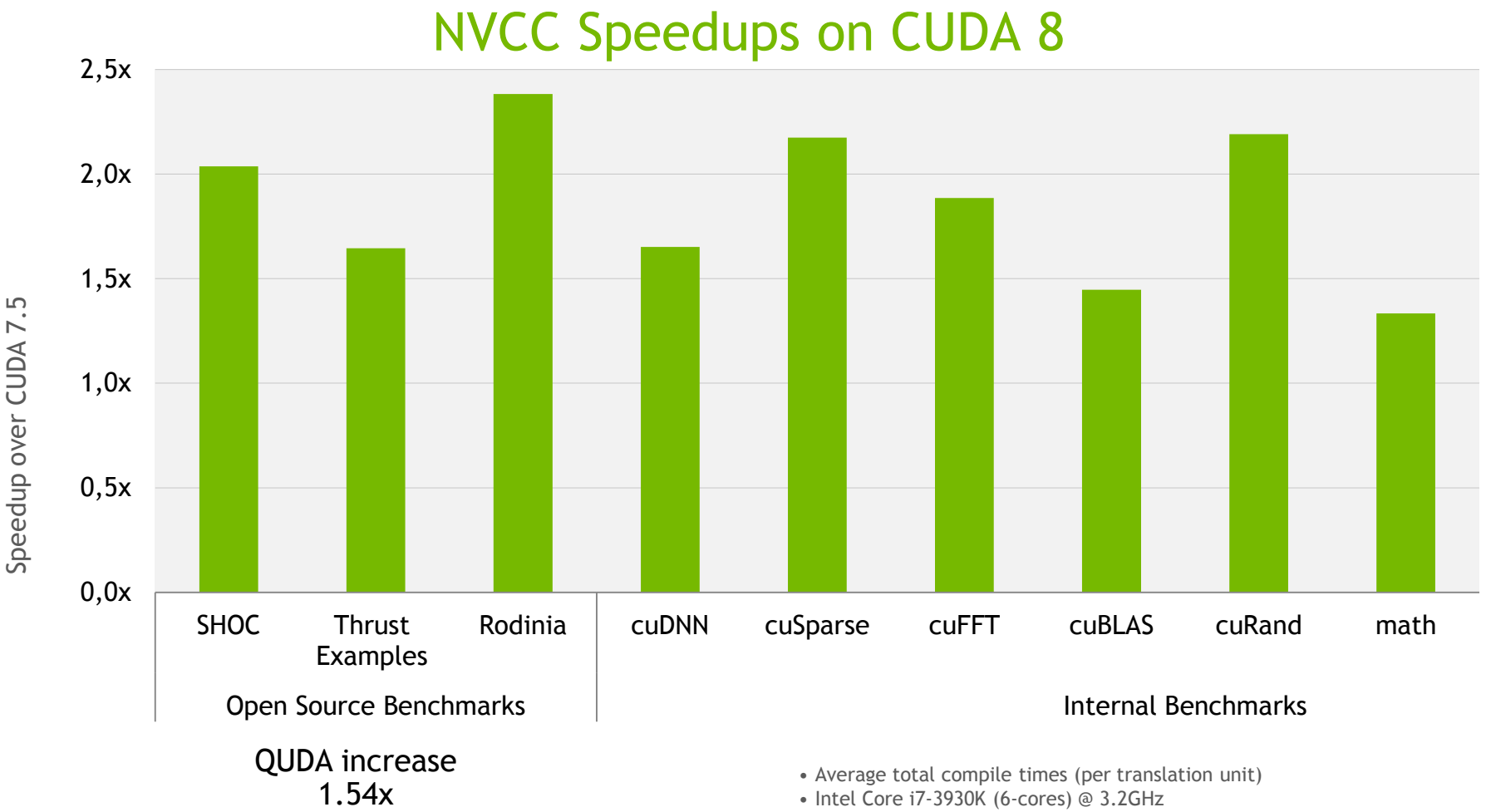
OpenACC Profiling



NVLink Topology and Bandwidth profiling

COMPILER IMPROVEMENTS

2X FASTER COMPILE TIME ON CUDA 8



Performance may vary based on OS and software versions, and motherboard configuration

- Average total compile times (per translation unit)
- Intel Core i7-3930K (6-cores) @ 3.2GHz
- CentOS x86_64 Linux release 7.1.1503 (Core) with GCC 4.8.3 20140911
- GPU target architecture sm_52

HETEROGENEOUS C++ LAMBDA

Combined CPU/GPU lambda functions

```
__global__ template <typename F, typename T>
void apply(F function, T *ptr) {
  *ptr = function(ptr);
}

int main(void) {
  float *x;
  cudaMallocManaged(&x, 2);

  auto square =
    [=] __host__ __device__ (float x) { return x*x; };

  apply<<<1, 1>>>(square, &x[0]);

  ptr[1] = square(&x[1]);

  cudaFree(x);
}
```

Call lambda from device code

__host__ __device__ lambda

Pass lambda to CUDA kernel

… or call it from host code

Experimental feature in CUDA 8.
`nvcc --expt-extended-lambda`

HETEROGENEOUS C++ LAMBDA

Usage with Thrust

```
void saxpy(float *x, float *y, float a, int N) {
    using namespace thrust;
    auto r = counting_iterator(0);

    auto lambda = [=] __host__ __device__ (int i) {
      y[i] = a * x[i] + y[i];
    };

    if(N > gpuThreshold)
      for_each(device, r, r+N, lambda);
    else
      for_each(host, r, r+N, lambda);
}
```

__host__ __device__ lambda

Use lambda in thrust::for_each on host or device

Experimental feature in CUDA 8.
`nvcc --expt-extended-lambda`

OPENACC

More Science, Less Programming

```
main()
{
  <serial code>
  #pragma acc kernels
  //automatically runs on
GPU
  {
    <parallel code>
  }
}
```

SIMPLE	Minimum efforts Small code modifications
POWERFUL	Up to 10x faster application performance
PORTABLE	Optimize once, run on GPUs and CPUs

FREE FOR ACADEMIA

PERFORMANCE PORTABILITY FOR EXASCALE

Optimize Once, Run Everywhere with OpenACC

2015	2016	2017
		ARM CPU
	OpenPOWER CPU	OpenPOWER CPU
	x86 Xeon Phi	x86 Xeon Phi
x86 CPU	x86 CPU	x86 CPU
AMD GPU	AMD GPU	AMD GPU
NVIDIA GPU	NVIDIA GPU	NVIDIA GPU

PGI Roadmaps are subject to change without notice.

NVIDIA

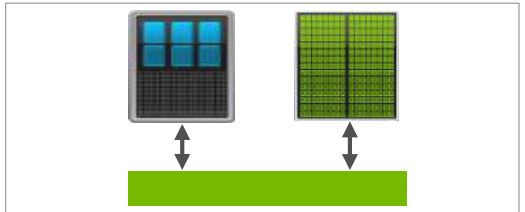
SUMMARY

NVIDIA SDK

TESLA P100

NVIDIA DGX-1

NVGRAPH

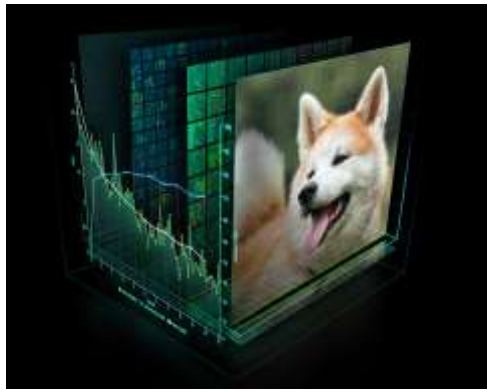


CUDA 8 & UNIFIED MEMORY

GPU TECHNOLOGY CONFERENCE

Sep 28-29, 2016 | Amsterdam
www.gputechconf.eu #GTC16

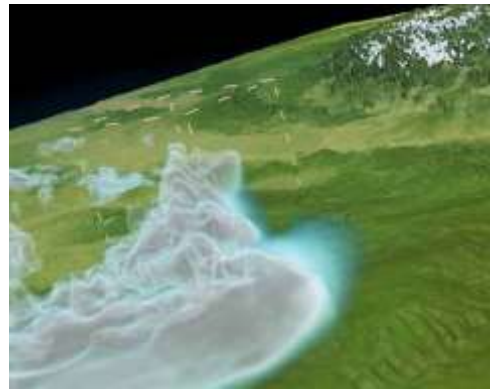
EUROPE'S BRIGHTEST MINDS & BEST IDEAS



DEEP LEARNING &
ARTIFICIAL INTELLIGENCE

SELF-DRIVING CARS

VIRTUAL REALITY &
AUGMENTED REALITY



SUPERCOMPUTING & HPC

GTC Europe is a two-day conference designed to expose the innovative ways developers, businesses and academics are using parallel computing to transform our world.

2 Days | 800 Attendees | 50+ Exhibitors | 50+ Speakers | 15+ Tracks | 15+ Workshops | 1-to-1 Meetings