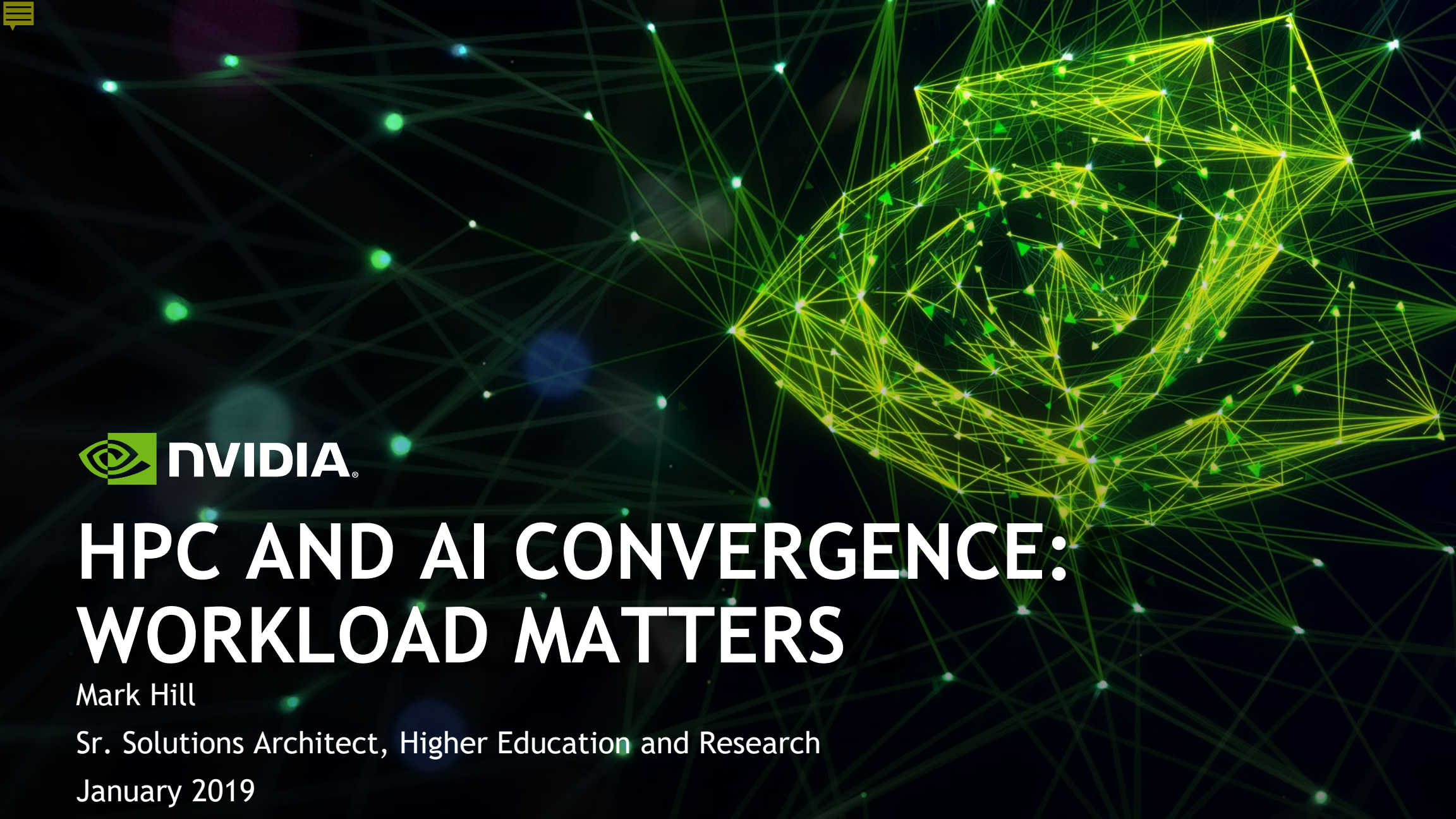


HPC AND AI CONVERGENCE: WORKLOAD MATTERS

Mark Hill

Sr. Solutions Architect, Higher Education and Research

January 2019

HPC AND AI CONVERGENCE

AI - A NEW INSTRUMENT FOR SCIENCE

HPC

> +40 years of algorithms based on first principles theory.

> Proven statistical models for accurate results

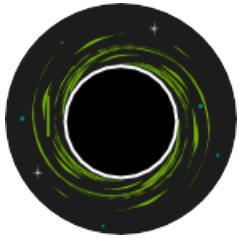
AI

> Neural networks that learn patterns from large data sets.

> Previously unmanageable data sets.

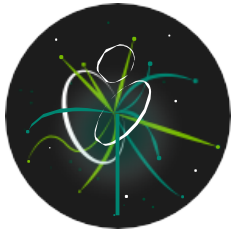
Dramatically Improves Accuracy and Time-to-Solution

Commercially viable fusion energy



Understanding cosmological dark energy and matter

Clinically viable precision medicine

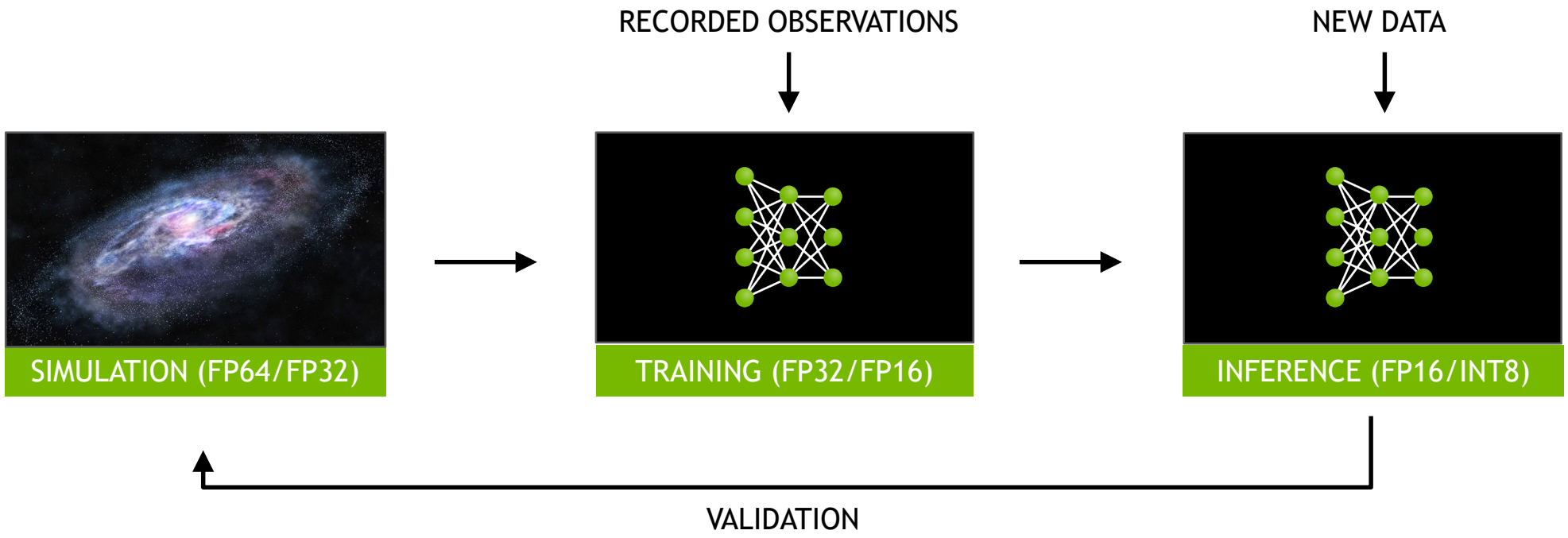


Improvement and validation of the Standard Model of Physics

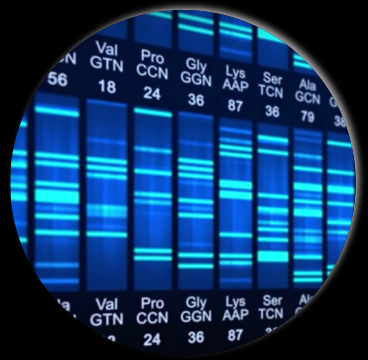
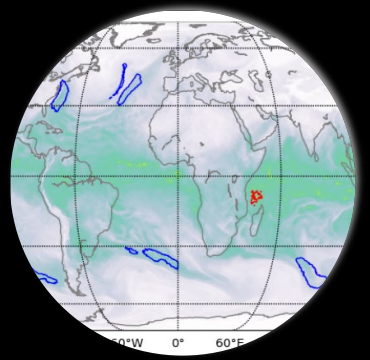
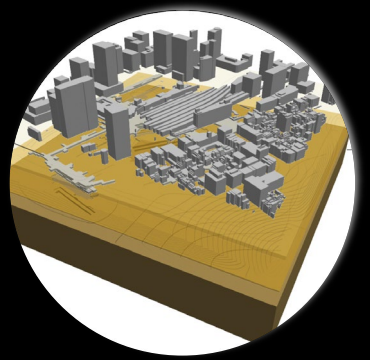
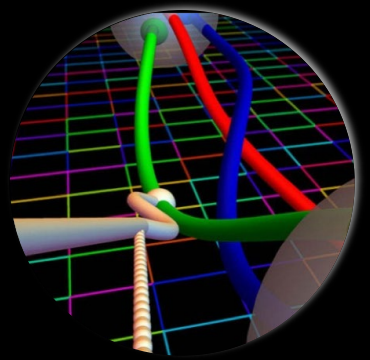
Climate/weather forecasts with ultra-high fidelity

AI - A NEW INSTRUMENT FOR SCIENCE

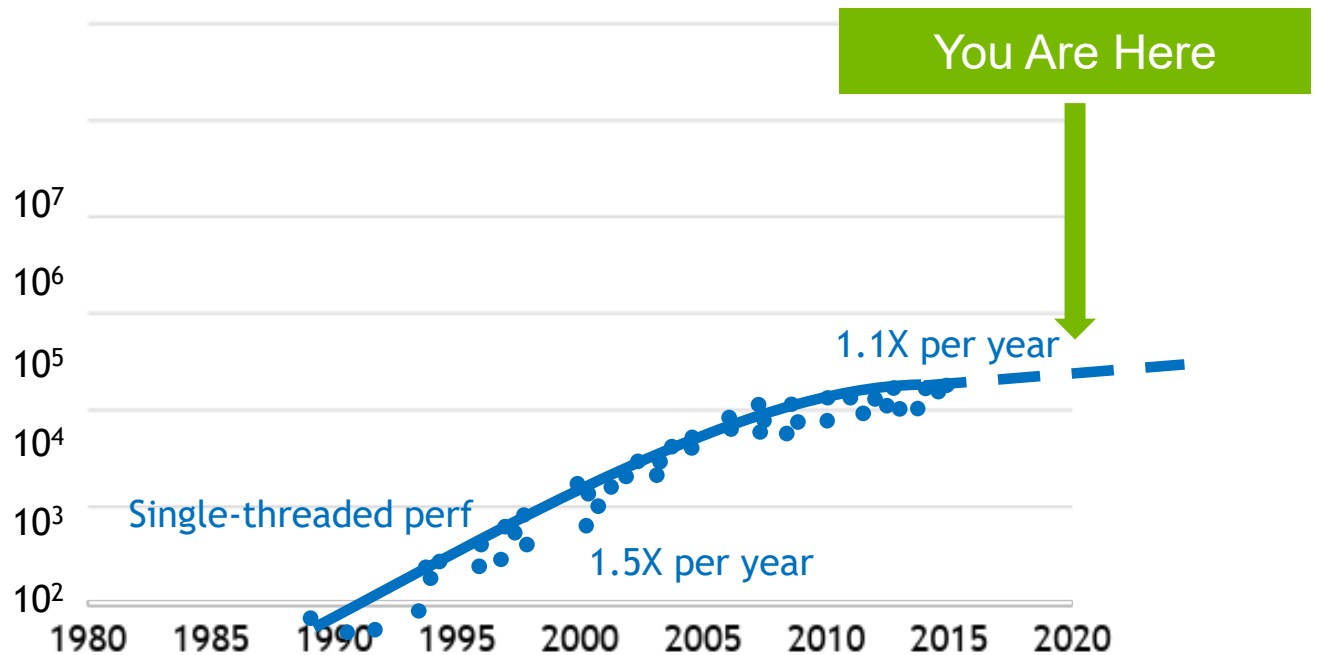
Requires A Unified Mixed Precision Platform



GORDON BELL NOMINEES ILLUSTRATE THE NEW NORMAL

New Algorithms	Scalable HPC*AI	New Algorithms HPC*AI	Scalable HPC*AI	New Algorithms
				
BERKELEY LAB, OAK RIDGE National Laboratory	BERKELEY LAB, NVIDIA	東京大学 THE UNIVERSITY OF TOKYO, OAK RIDGE National Laboratory	OAK RIDGE National Laboratory	BERKELEY LAB, Lawrence Livermore National Laboratory, NVIDIA
Genomics 2.36 ExaOps	Weather 1.15 ExaOps	Seismic 25X	Material Science 300X	Quantum Chromodynamics 15x

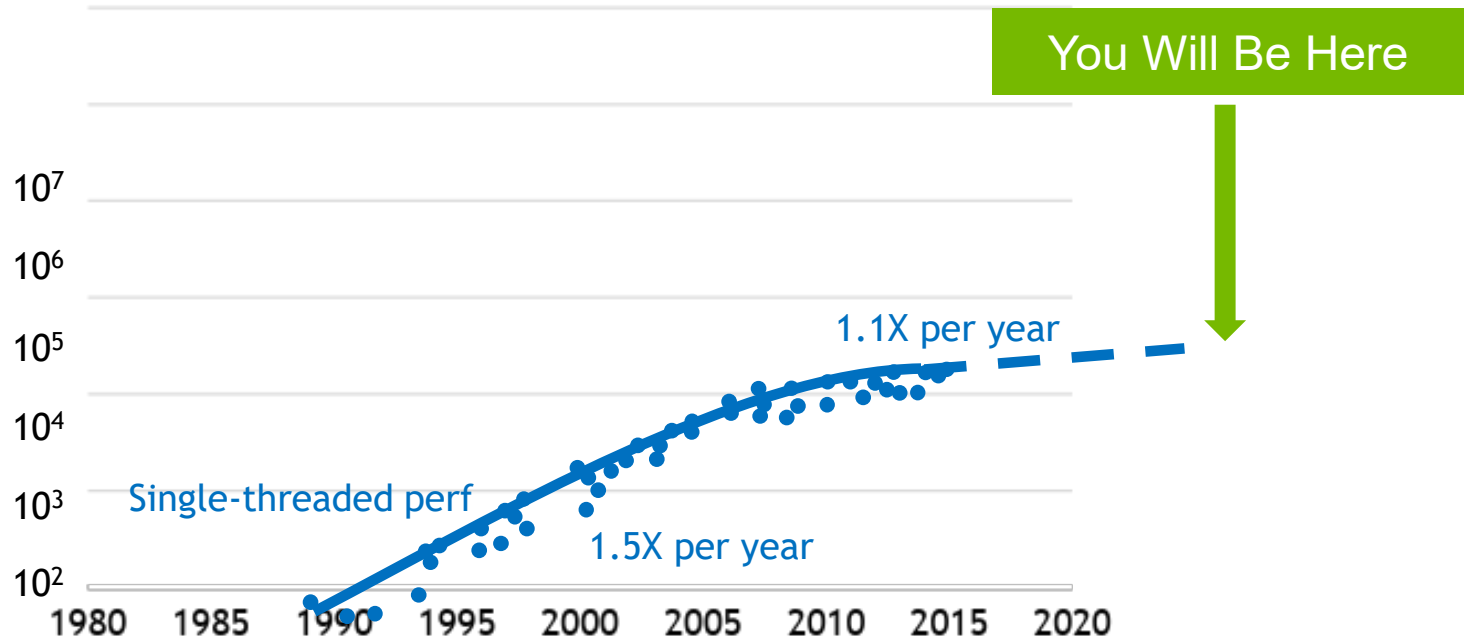
THE CPU IS OUT OF GAS



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

THE CPU IS OUT OF GAS

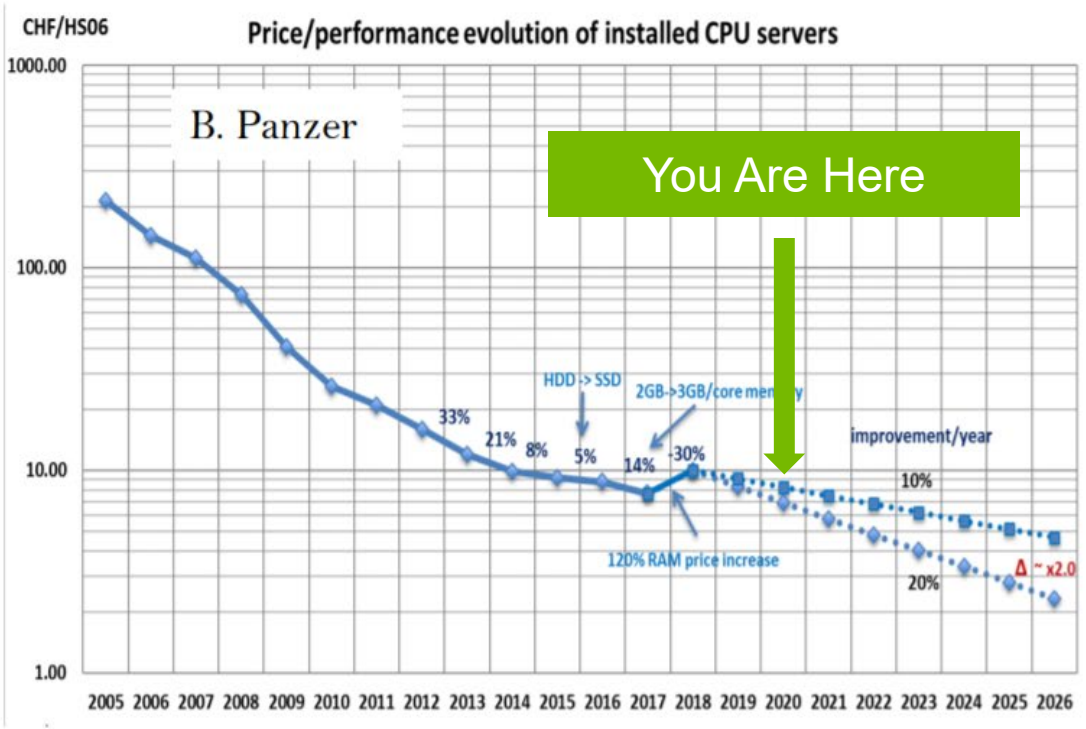
No Really Running on Fumes



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

PRICE/PERFORMANCE ALREADY CONSTRAINED

CPU trends



- CPU evolution is not able to cope with the increasing demand of performance
- Depending on the application, GPUs can provide better performance and energy efficiency

THE NEW HPC WORKLOAD

A Dramatic Shift from the Past

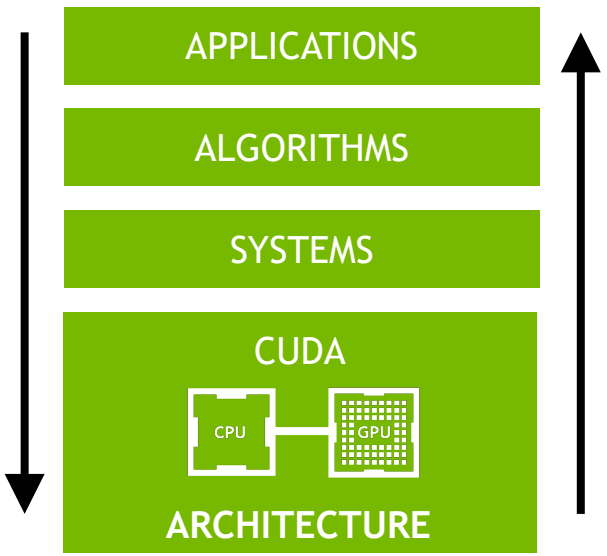
PREVIOUS NORMAL

- Simulated data sources
- Conventional 64 bit full order ab initio simulation
- Conventional reduced order methods
- All jobs are batch

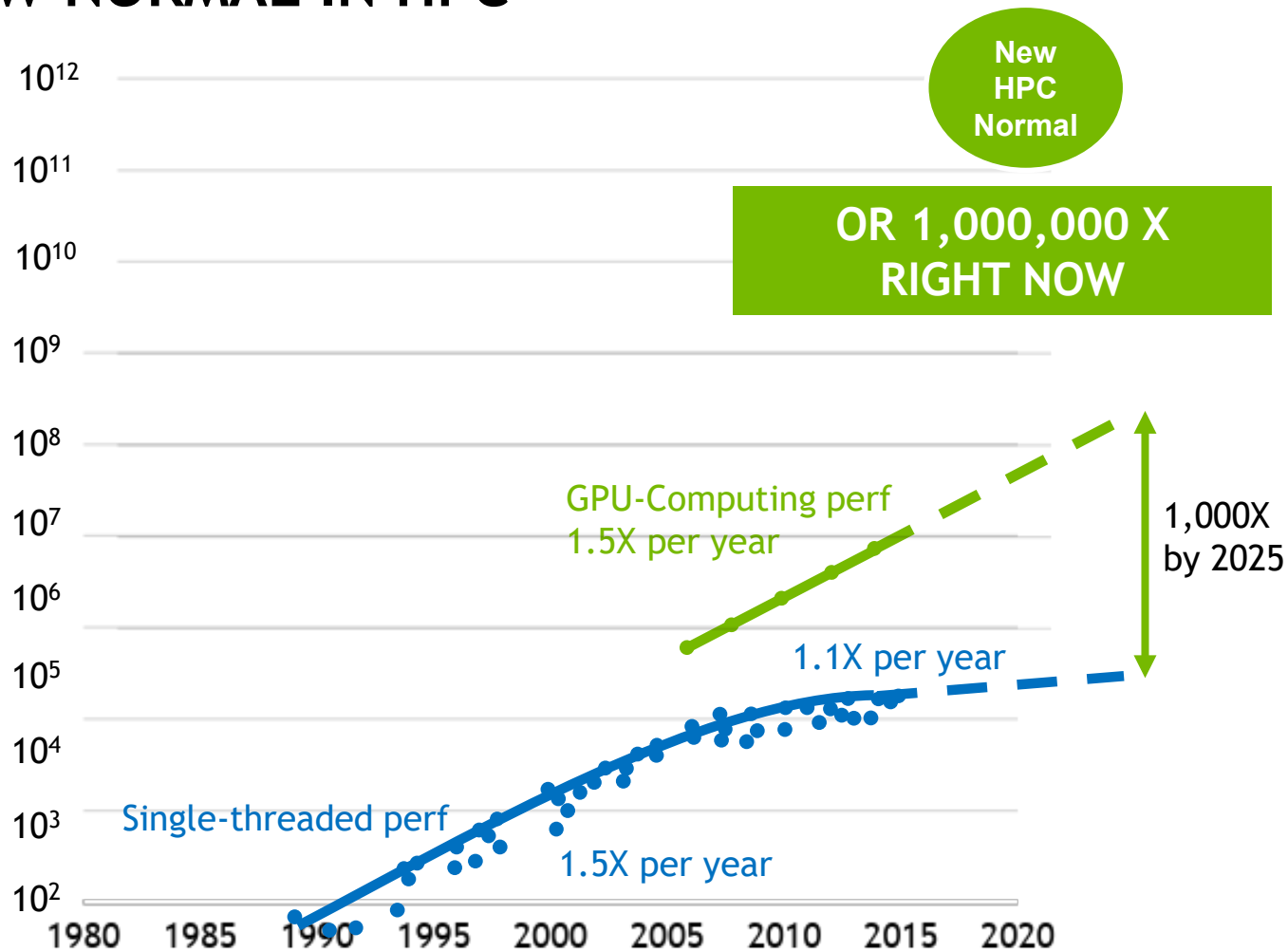
NEW NORMAL

- A mix of experimental and simulated/emulated data sources
- Conventional 64 bit full order ab initio simulation
- Mixed precision algorithms with new methods for ab initio simulation
- Embedded neural nets within full order ab initio simulation
- Conventional reduced order methods
- Deep Neural Net and Machine learning emulation
- Interactive large scale training (model parallel, data parallel and hyperparameter optimization)

NEW ALGORITHMS AND METHODS ARE EMERGING TO DEFINE A NEW NORMAL IN HPC

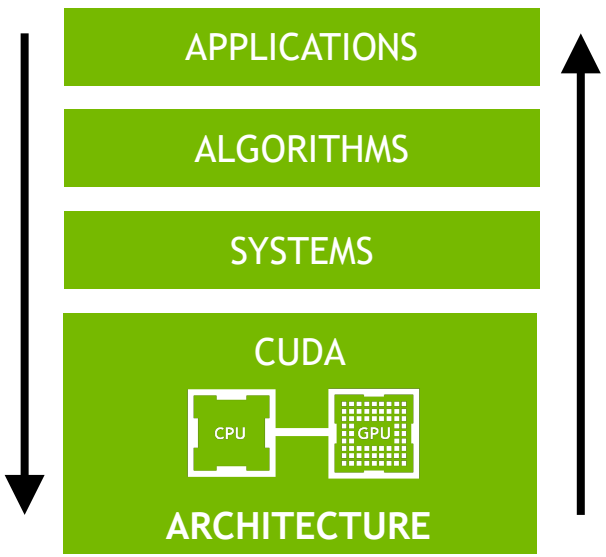


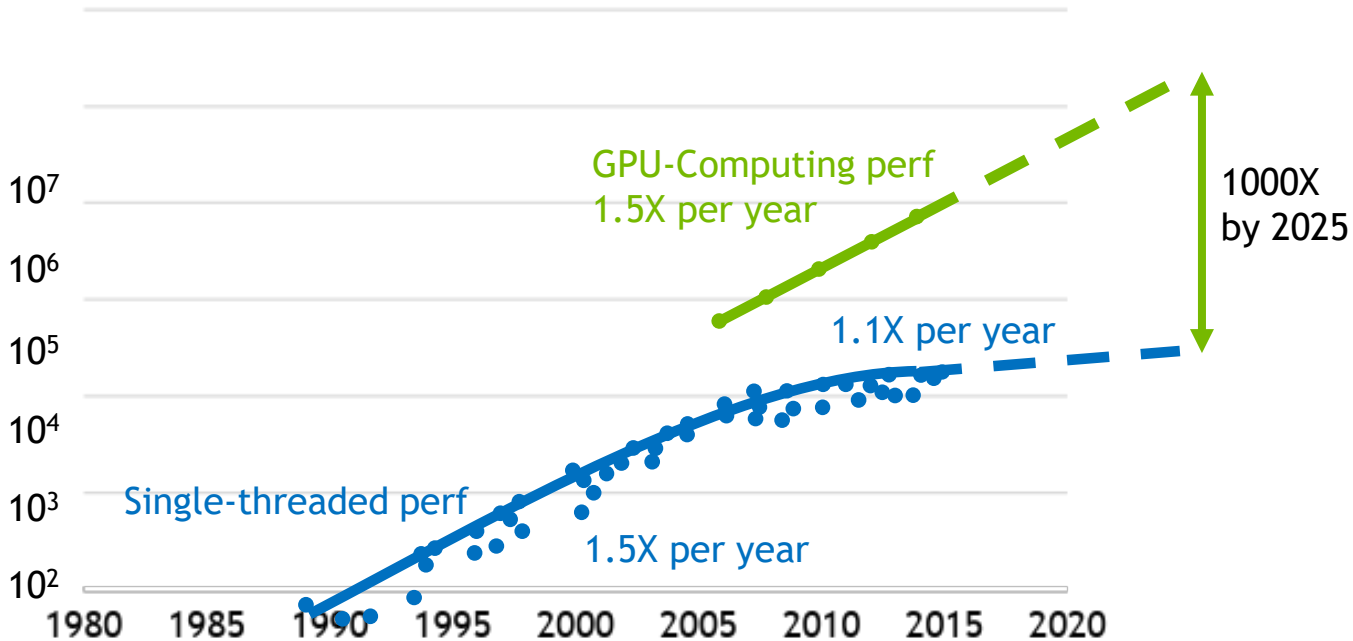
Are You Ready for the NEW NORMAL?



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

AN ACCELERATED SYSTEM CAN MAINTAIN OR EXCEED PREVIOUS PERFORMANCE TRAJECTORY





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

BUT NOT ALL APPS can move from the BLUE line to the GREEN line
And their "mileage" will vary

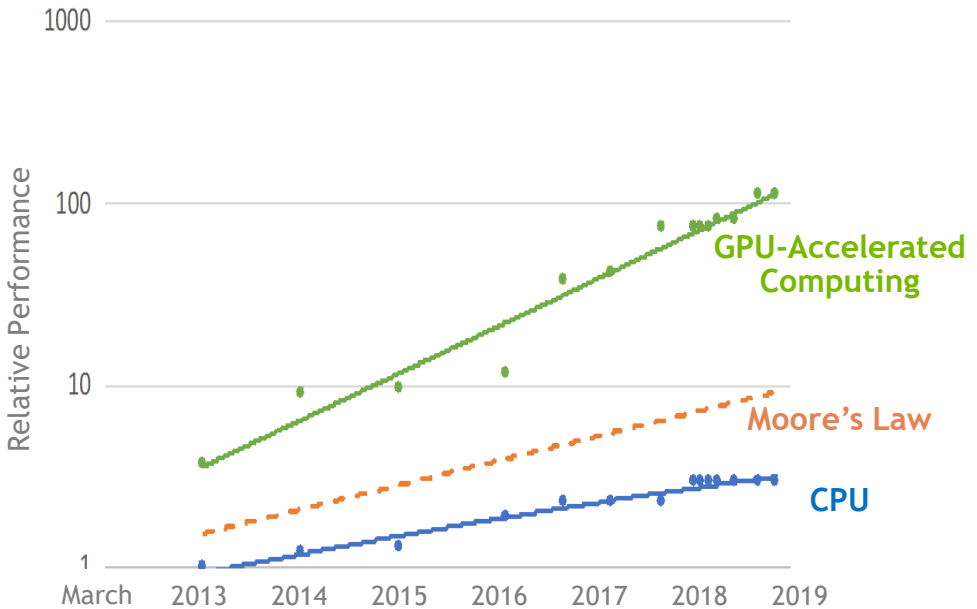
BEYOND MOORE'S LAW

Progress Of Stack In 6 Years

2013

- cuBLAS: 5.0
- cuFFT: 5.0
- cuRAND: 5.0
- cuSPARSE: 5.0
- NPP: 5.0
- Thrust: 1.5.3
- CUDA: 5.0
- Resource Mgr: r304
- Base OS: CentOS 6.2

Accelerated Server With Fermi



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

2019

- cuBLAS: 10.0
- cuFFT: 10.0
- cuRAND: 10.0
- cuSOLVER: 10.0
- cuSPARSE: 10.0
- NPP: 10.0
- Thrust: 1.9.0
- CUDA: 10.0
- Resource Mgr: r384
- Base OS: Ubuntu 16.04

Accelerated Server with Volta

SENSE OF URGENCY??

Dennard Scaling Ended 13 years ago

Single Threaded Performance is Capped at ~ 2.5 GHZ

Moore's Law is Slowing

Transistor Density Increasing at a Slower Rate

The Economics Have Changed

More Transistors per Generation Cost More to Produce, and Don't Run Any Faster

Power Wall - Patterson ~1996

Accelerated System…

What's Your Plan?

THE KEY DIFFERENCES

Unaccelerated systems

Have homogenous Nodes, where all nodes are the same

Max throughput with minimum variance regardless of workload

All the apps are modeling/simulation

Most apps are batch

Most modeling/simulation apps are double precision

Accelerated systems

Have heterogenous Nodes, where nodes are tuned to the workload

Max throughput and maximize performance variance based on workload

Wide range of apps: modeling/simulation to AI/DL to data science

A Mix of Batch and Interactive jobs

Precision selected to optimize performance

IT'S GOING TO BE 2023 BEFORE YOU KNOW IT

- Key focus on Optimum Configuration to Maximize Throughput/Cost
  - Maximize Throughput and Minimize Operating Cost within current technology constraints
  - Peak FLOPS and Top 500 have to be considered but not become a distraction
  - Design an optimum configuration based on current and future workload
- Key Points of Emphasis
  - All codes won't likely be accelerated, so understanding the workload is critical
  - An optimum configuration requires a complete stack that includes a robust ecosystem (hardware, software and humans)
  - **A "New HPC" is emerging where convergence of Modeling/Simulation and AI (Data Science) is demonstrating performance gains >>> than Moore's Law**

NVIDIA DGX SUPERPOD

AI LEADERSHIP REQUIRES

AI INFRASTRUCTURE LEADERSHIP

Test Bed for Highest Performance Scale-Up Systems

- 9.4 PF on HPL | ~200 AI PF | #22 on Top500 list
- <2 mins To Train RN-50

Modular & Scalable GPU SuperPOD Architecture

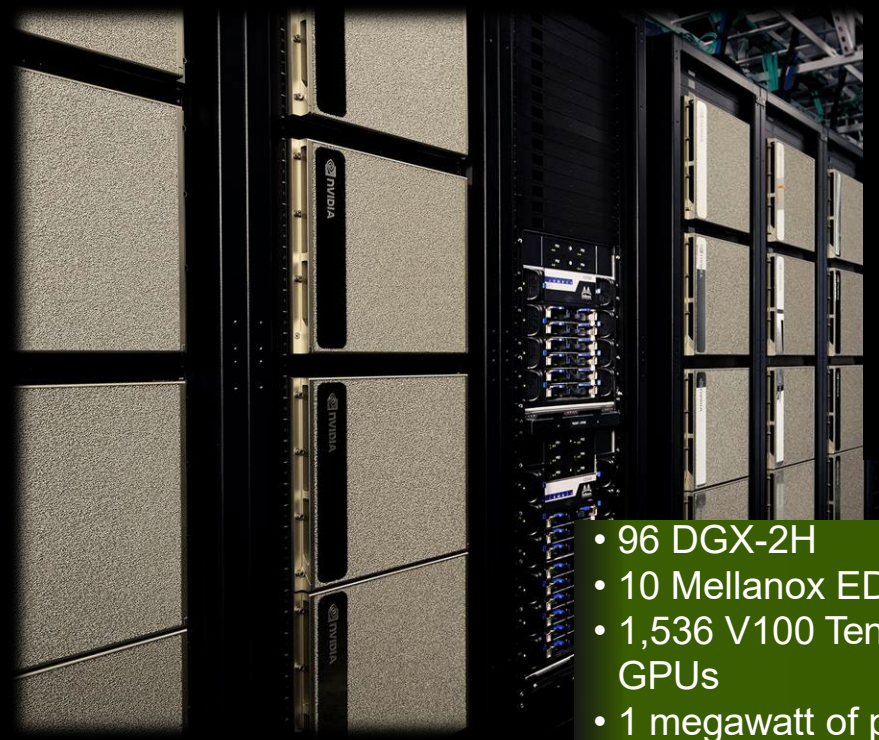
- Built in 3 Weeks
- Optimized For Compute, Networking, Storage & Software

Integrates Fully Optimized Software Stacks

- Freely Available Through NGC

Autonomous Vehicles | Speech AI | Healthcare | Graphics | HPC

https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/dgx-pod/nvidia-dgx-superpod-datasheet.pdf

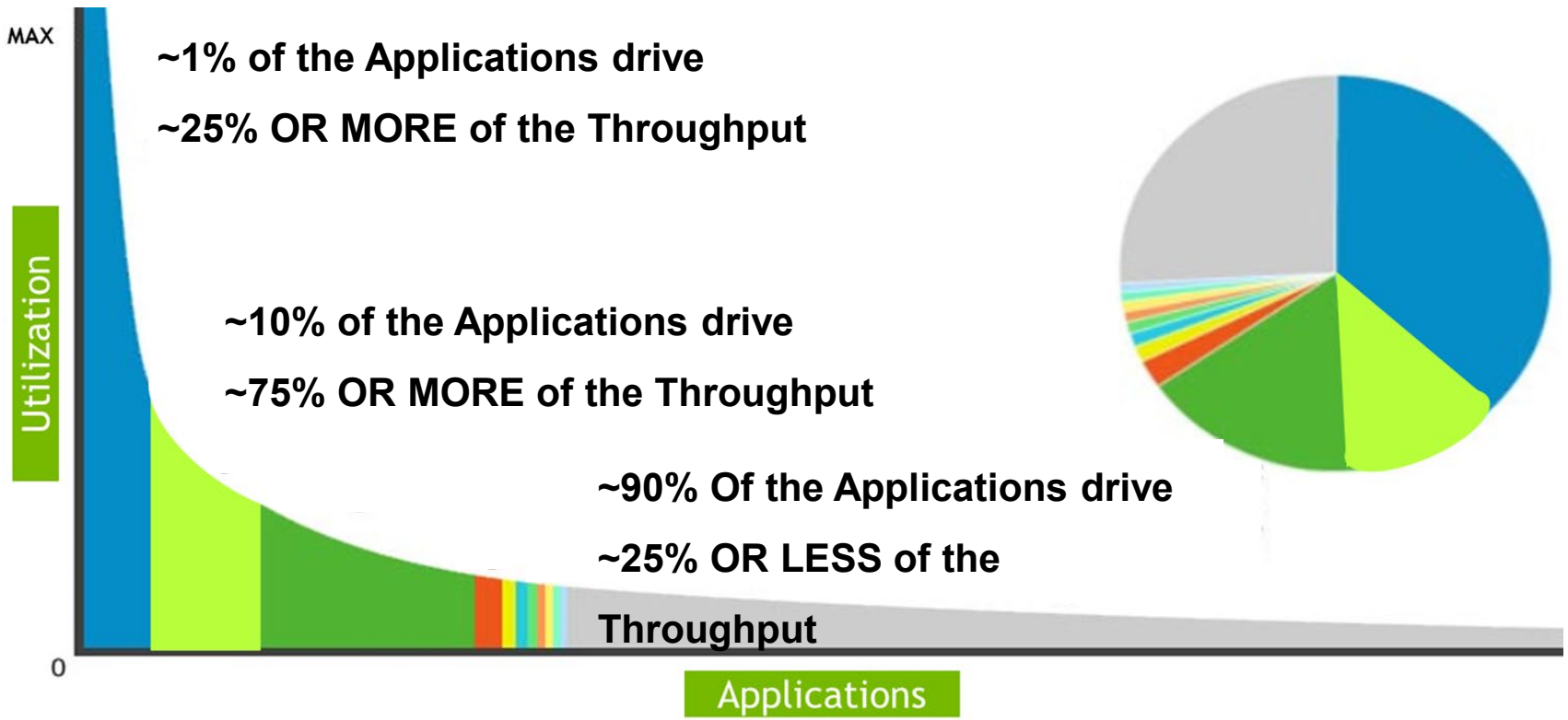


- 96 DGX-2H
- 10 Mellanox EDR IB per node
- 1,536 V100 Tensor Core GPUs
- 1 megawatt of power

WHAT'S YOUR WORKLOAD?

TYPICAL USAGE PROFILE

Apps May Change but the Shape is the Same



Accelerating as few as 5 Key Applications Benefits 100's to 1000's of Users

3 THINGS

Then how do we make it faster?

- The Resource Scheduler can tell you how well your scheduling jobs on your infrastructure

- What apps are running and how much resource do they consume?
  - Xalt
- How efficiently are they using the resources (CPU/GPU)
  - Profile the code
    - {DL prof, nvprof} -> nsight systems 2.1

XALT

- XALT is an open source tool to collect system usage data
  - https://github.com/Fahey-McLay/xalt
- Maintained by the Texas Advanced Computing Center
- Used by TACC, NICS, NCSA, KAUST, …
- Commercial support from Ellexus
- Uses LD_PRELOAD env var to run code before and after main()

NV Online Documentation help https://xalt.readthedocs.io/en/latest/

DATABASE QUERIES

Top 10 Users

```
$ mysql xalt -t -e "SELECT user, syshost AS cluster, COUNT(date) AS count,
ROUND(SUM(num_cores*run_time/3600),6) AS core_hours FROM xalt_run WHERE DATEDIFF(NOW(), date) < 7
GROUP BY user, syshost ORDER BY core_hours DESC LIMIT 10;"
```

user	cluster	count	core_hours
tlee	psg	434	5543.335433
award	psg	366	1003.509619
pspringer	prm	56838	527.143689
chyan	psg	288	136.322844
eweinberg	prm	709	101.652244
xan	prm	33	92.128333
bbadal	psg	467	30.818167
jianxiangm	psg	300	29.292125
mclark	prm	844	21.848478
afroger	prm	301	17.765908

DATABASE QUERIES

Top 20 Executables

```
$ mysql xalt -t -e "SELECT exec_path AS program, COUNT(date) AS count, ROUND(SUM(num_cores*run_time/3600),6) AS core_hours FROM
xalt_run WHERE syshost = 'psg' AND DATEDIFF(NOW(), date) < 7 GROUP BY program ORDER BY core_hours DESC LIMIT 20;"
```

program	count	core_hours
/home/tlee/amber_release/bin/pmemd.cuda_SPFP.MPI	434	5543.335433
/home/award/NAMD/NAMD_2.13-July26_2018/namd2.13-smp-cuda-memopt	43	267.571533
/home/award/NAMD/NAMD_2.13-July26_2018/charmrun	56	228.394044
/home/award/NAMD/NAMD_2.13-July26_2018/namd2.13-multicore-cuda	160	227.582111
/home/award/GROMACS/gromacs-2018.2_installation_gcc73_non-mpi_cuda9288/bin/gmx	60	198.022133
/datasets/chyan/libjpeg-turbo/build/tjbench	278	136.322267
/home/award/NAMD/NAMD_2.13-July26_2018/namd2.13-multicore-cuda-memopt	6	71.763556
/home/jianxiangm/amber16/bin/pmemd.cuda_SPFP.MPI	21	29.257417
/cm/extra/apps/CUDA.linux86-64/9.2.88.1_396.26/bin/cuda-gdb	21	26.496056
/home/award/pdsh/pdsh-2.26_installation_with-ssh_without_rsh/bin/pdsh	21	10.137067
/tmp/ns.exe	16	3.458225
/home/nvydyanathan/Work/fftw2/fftw-2.1.5/mpi/rfftw_mpi_test	40	3.455725
/cm/extra/apps/CUDA.linux86-64/9.1.85_387.26/bin/nvprof	93	3.375119
/home/mknox2/RELION/relion_installation_SP_sm70_stock_kernel/bin/relion_refine_mpi	10	2.989922
/home/robv/apps/namd-2.13-multicore-cuda	10	2.608000
/cm/extra/apps/CUDA.linux86-64/9.2.88.1_396.26/bin/cuda-memcheck	147	2.154222
/home/jacksontu/quda-build/tests/invert_test	73	1.715750
/home/karumugam/workspace/repos/gitlab/LAMMPS-PERFLAB_201807/src/lmp_kokkos_cuda_mpi	28	1.697067
/home/ykang/zhifeng/feedbackLoopCUDA/feedbackLoop	27	1.201483
/home/bbadal/sam	30	1.082194

GPU TRACKING

- XALT 2.2 introduced GPU tracking based on DCGM

  Install DCGM
  ```
  ./configure --with-trackGPU=yes …
  ```

- Be aware of a corner case bug
  - If the user binary uses Google Protobuf and is also linked with DCGM, then the binary will abort due to a Protobuf version mismatch
  - https://github.com/Fahey-McLay/xalt/issues/36

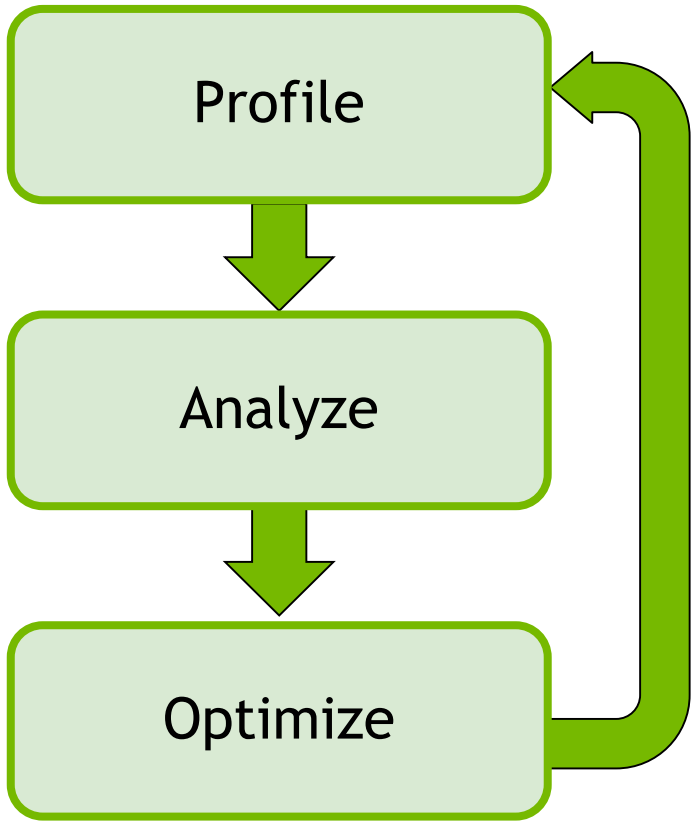
GETTING GOOD DATA TO MAKE GOOD DECISIONS

XALT

- XALT can help you quantitatively understand the application mix running on a system
- XALT answers the question “what’s your workload?”

Nsight Systems Introduction

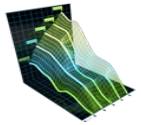
Typical Optimization Workflow



Iterate until desired performance is achieved

Legacy Transition

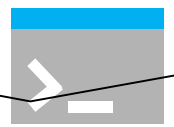
nvprof
Command-line tool

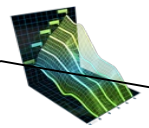


NVIDIA Visual Profiler
Standalone UI

Legacy Transition

Nvprof
Command-line tool





NVIDIA Visual Profiler
Standalone UI

Nsight Systems
Standalone GUI+CLI

- CPU-GPU interactions & triage
- Low overhead capture
- GPU compute & graphics
- Faster GUI + more data

Nsight Compute
Standalone GUI+CLI

- GPU CUDA kernel analysis & debug
- Very high freq GPU perf counters
- Compare results (diff)
- Incredible statistics & customizable

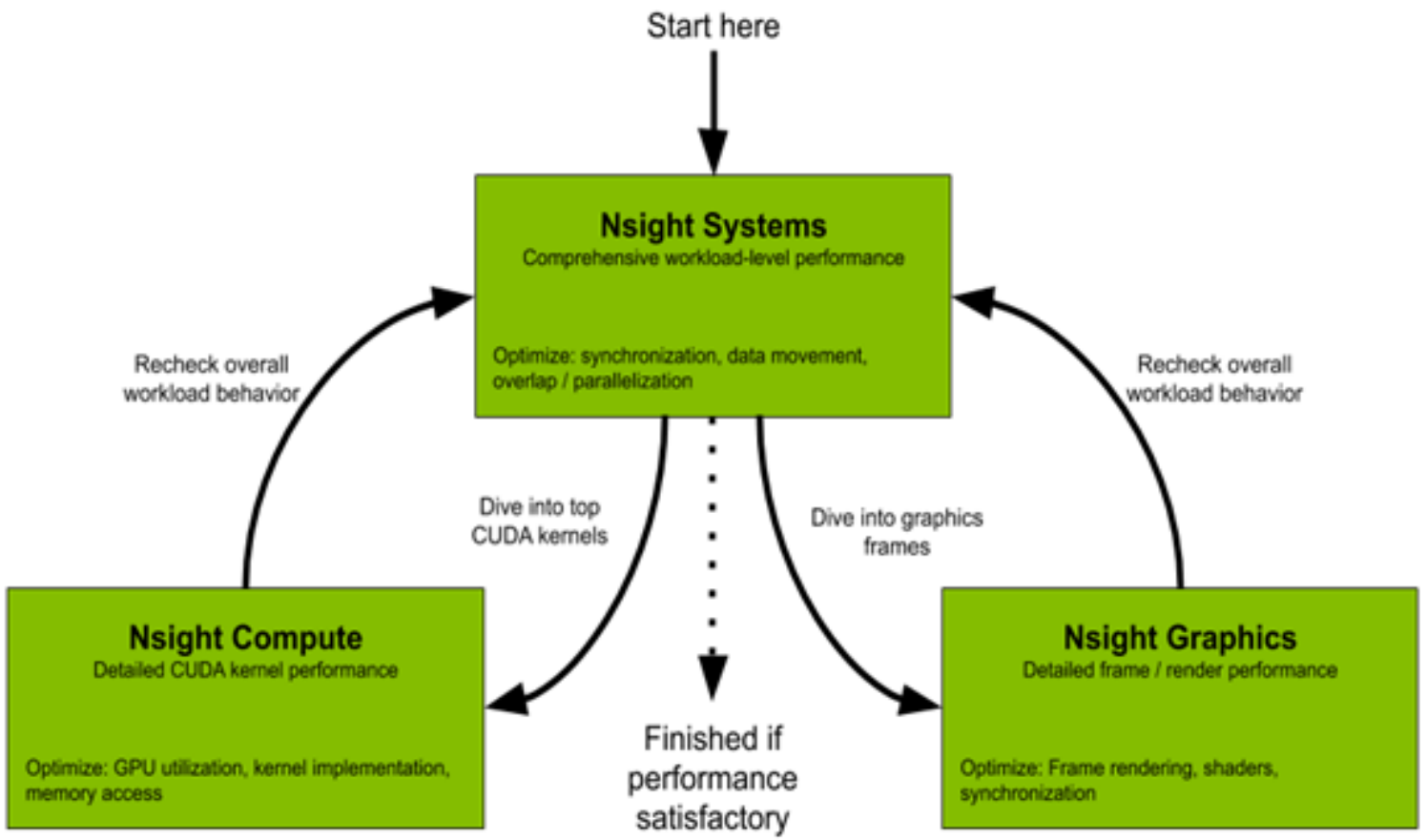
Nsight Product Family

Workflow

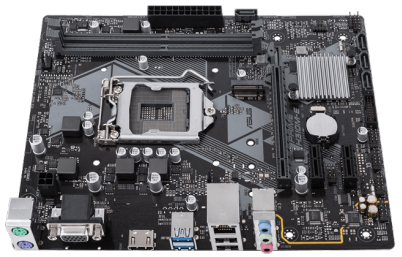
Nsight Systems -
Analyze application algorithm system-wide

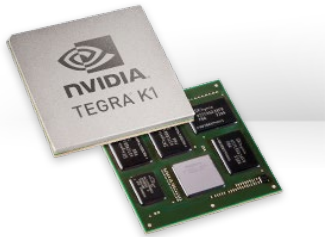
Nsight Compute -
Debug/optimize CUDA kernel

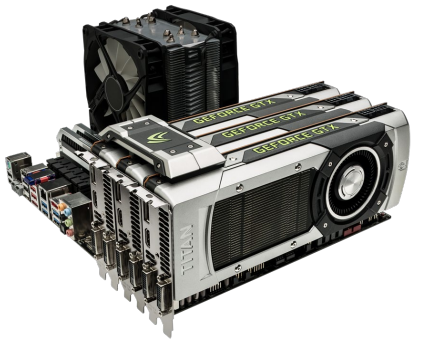
Nsight Graphics -
Debug/optimize graphic workloads



Tuning an Orchestra of Tasks







Lattice Microbes

Overview

- System-wide application algorithm tuning
  - Multi-process tree support
- Locate optimization opportunities
  - Visualize millions of events on a very fast GUI timeline
  - See gaps of unused CPU and GPU time
- Balance your workload across multiple CPUs and GPUs
  - CPU algorithms, utilization, and thread state
  - GPU streams, kernels, memory transfers, etc
- Multi-platform: Linux & Windows, x86-64 & Tegra
  - Mac (host-only)

Timeline Features

- **Compute**
  - **CUDA 9+ API & GPU workload ranges with correlation**
    - **Memcpy/set and UVM transfers**
  - **Libraries: cuBLAS, cuDNN, TensorRT**
  - **OpenACC**
- **Graphics**
  - **Vulkan, OpenGL, Direct3D11 , Direct3D12, DXR, V-sync**
- **OS**
  - **Thread state and CPU utilization**
  - **OS runtime long call trace (>1us, pthread, glibc → mmap, file & IO, ...)**
    - **Call-stack backtraces (>80us)**
  - **ftrace / ETW (page faults, signal, interrupts, …)**
  - **MPI**
- **User annotations API (NVTX)**

Other Key Features

- **Thread call-stack periodic sampling**
  - **Backtraces via hardware LBRs or frame pointers**
  - **Hot functions**
- **Command Line Interface (CLI)**
  - **No host PC required to record**
  - **No need for root access**
  - **Ready to use in cloud environments**
  - **Works in docker containers**
  - **Usable on HPC systems with access limitations**
  - **Interactive mode**

GETTING GOOD DATA TO MAKE GOOD DECISIONS

Nsight

- **Nsight Systems - Analyze application algorithm system-wide**
- **Nsight Compute - Debug/optimize CUDA kernel**
- **Nsight Graphics - Debug/optimize graphics workload**

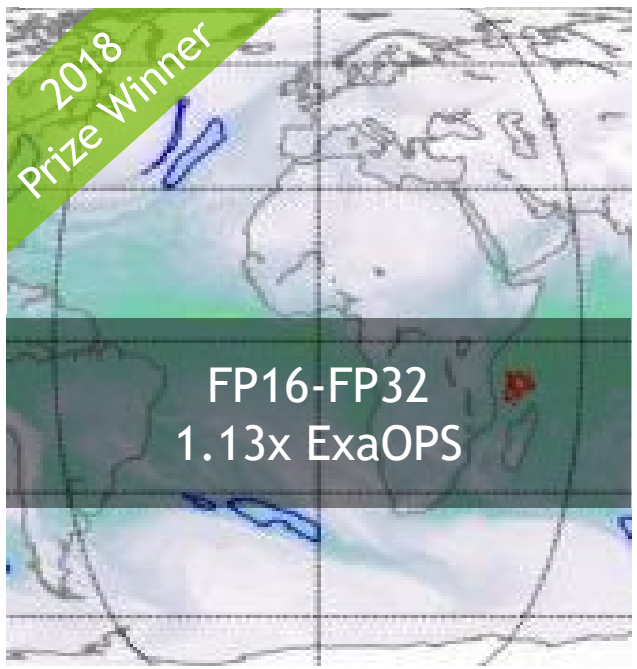
THE IDEAL OUTCOME AT SCALE

The RIGHT number of GPU's at the RIGHT price
The RIGHT ratio of GPU's to CPU cores
The RIGHT mix of GPU capabilities

TENSOR CORES FOR HPC + AI

GORDON BELL FINALISTS ON NVIDIA GPUS

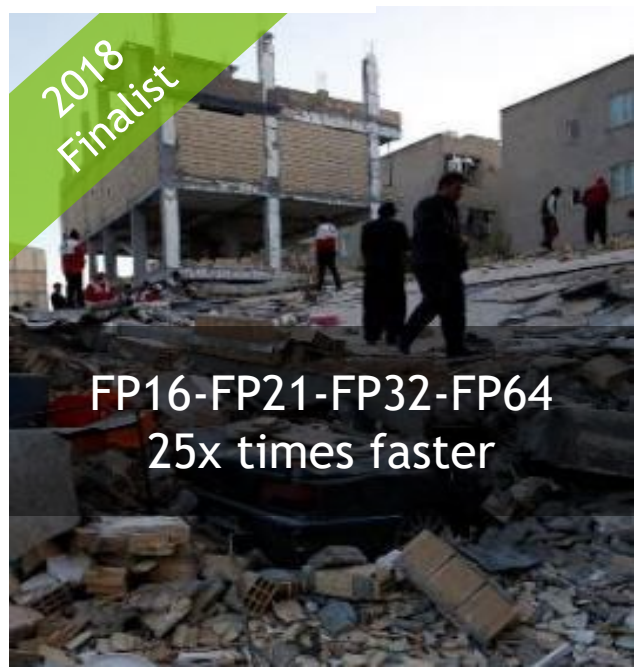
Mixed Precision Critical to Breaking Records



BERKELEY LAB · OAK RIDGE National Laboratory · NVIDIA

Weather

Fastest Deep Learning Algorithm



東京大学 THE UNIVERSITY OF TOKYO · OAK RIDGE National Laboratory

Seismic

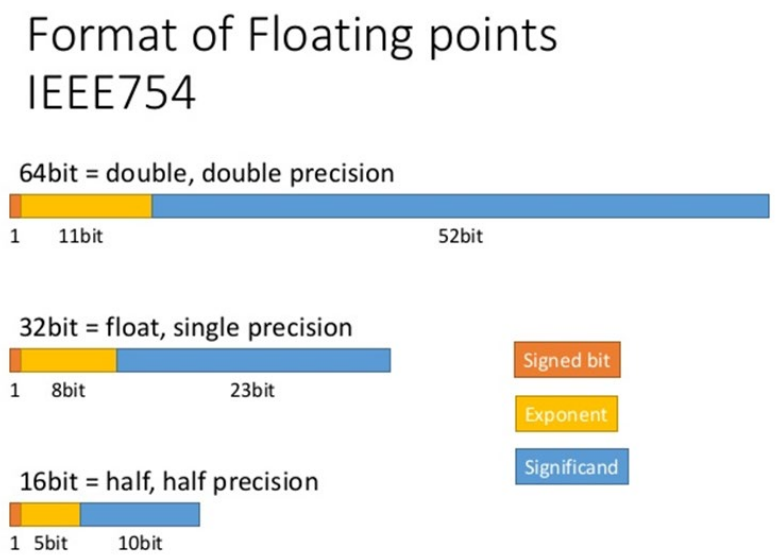
1st Soil & Structure Simulation

HISTORY AND FP64

Double-precision floating point (FP64) has been the de facto standard for doing scientific simulation for several decades.

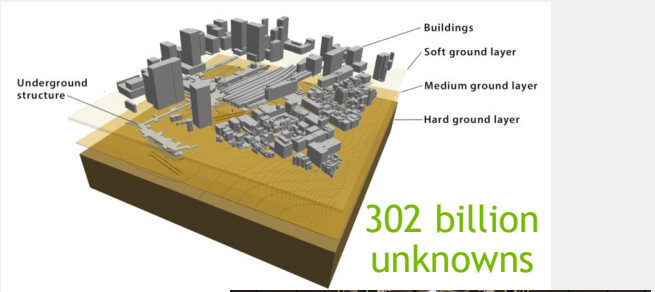
Most numerical methods used in engineering and scientific applications require the extra precision to compute correct answers or even reach an answer.

FP64 also requires more computing resources and runtime to deliver the increased precision levels.



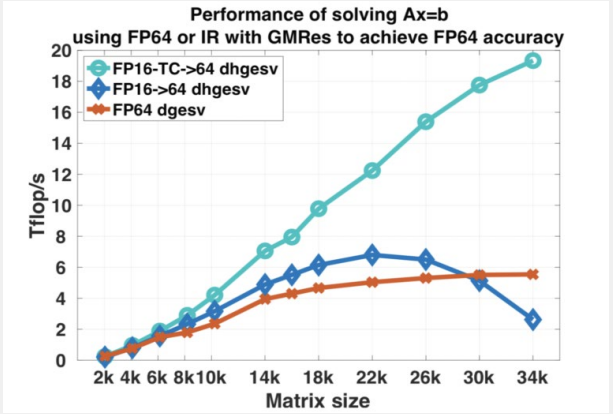
HPC SPEEDUPS 4X TO 25X

FP64 Operations Accelerated by Mixed Precision



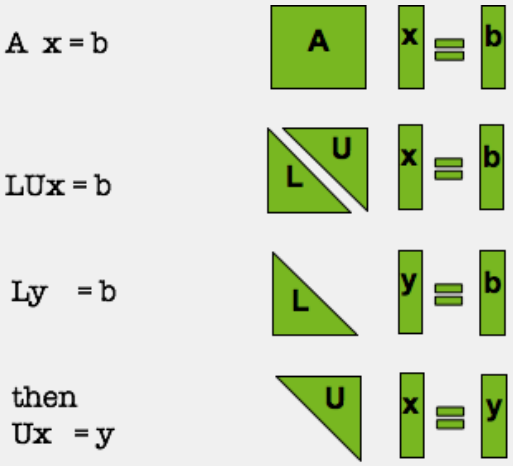
Tokyo Earthquake Simulation - 25X Speedup

2018 Gordon Bell Finalist on SUMMIT Supercomputer



MAGMA – Researchers Achieve 4X Speedup

FP64 operations solved in FP16 Mixed Precision



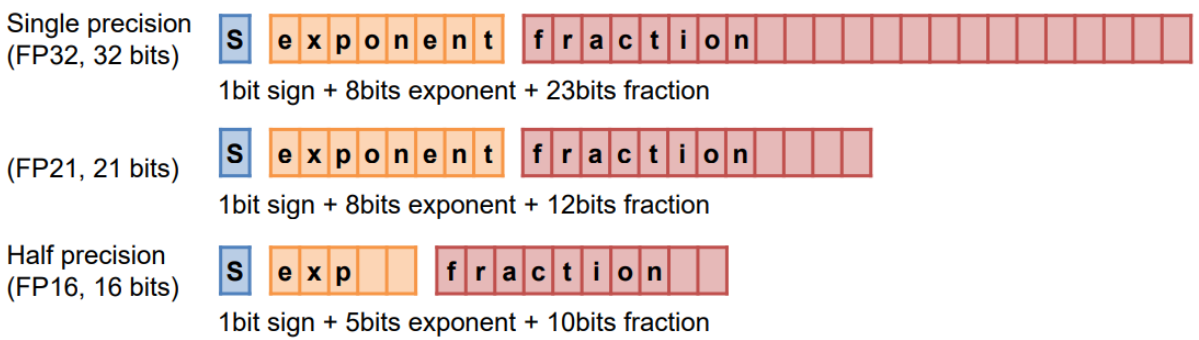
FP64 Dense Linear Algebra Functions

Iteratively Solve in Reduced Precisions

FP21 ?

Introduction of custom data type: FP21

- Most computation in CG loop is memory bound
  - However, exponent of FP16 is too small for use in global vectors
- Use FP21 variables for memory bound computation
  - Only used for storing data (FP21 × 3 are stored into 64bit array)
  - Bit operations used to convert FP21 to FP32 variables for computation



BFLOAT16?

Brain Float 16

bfloat16 has the following format:

- Sign bit: 1 bit
- Exponent width: 8 bits
- Significand precision: 8 bits (7 explicitly stored), as opposed to 24 bits in a classical single-precision floating-point format

The bfloat16 format, being a truncated IEEE 754 single-precision 32-bit float, allows for fast conversion to and from an IEEE 754 single-precision 32-bit float; in conversion to the bfloat16 format, the exponent bits are preserved while the significand field can be reduced by truncation (thus corresponding to round toward 0), ignoring the NaN special case. Preserving the exponent bits maintains the 32-bit float's range of $\approx 10^{-38}$ to $\approx 3 \times 10^{38}$.[10]

The Intel Papers

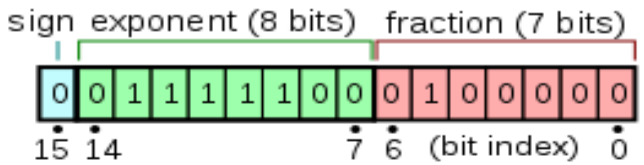
https://arxiv.org/pdf/1905.12322.pdf

https://arxiv.org/pdf/1904.06376.pdf

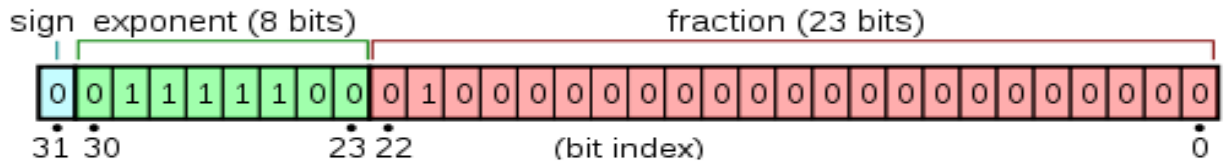
BFLOAT16?

Brain Float 16

The bits are laid out as follows:

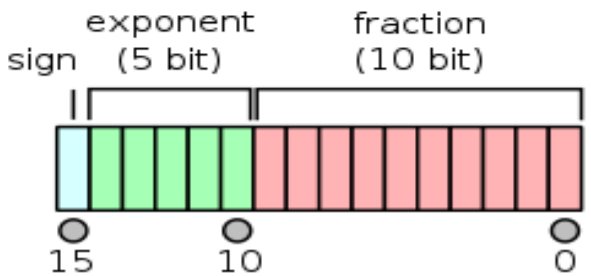


Compare to an IEEE 754 single-precision 32-bit float:



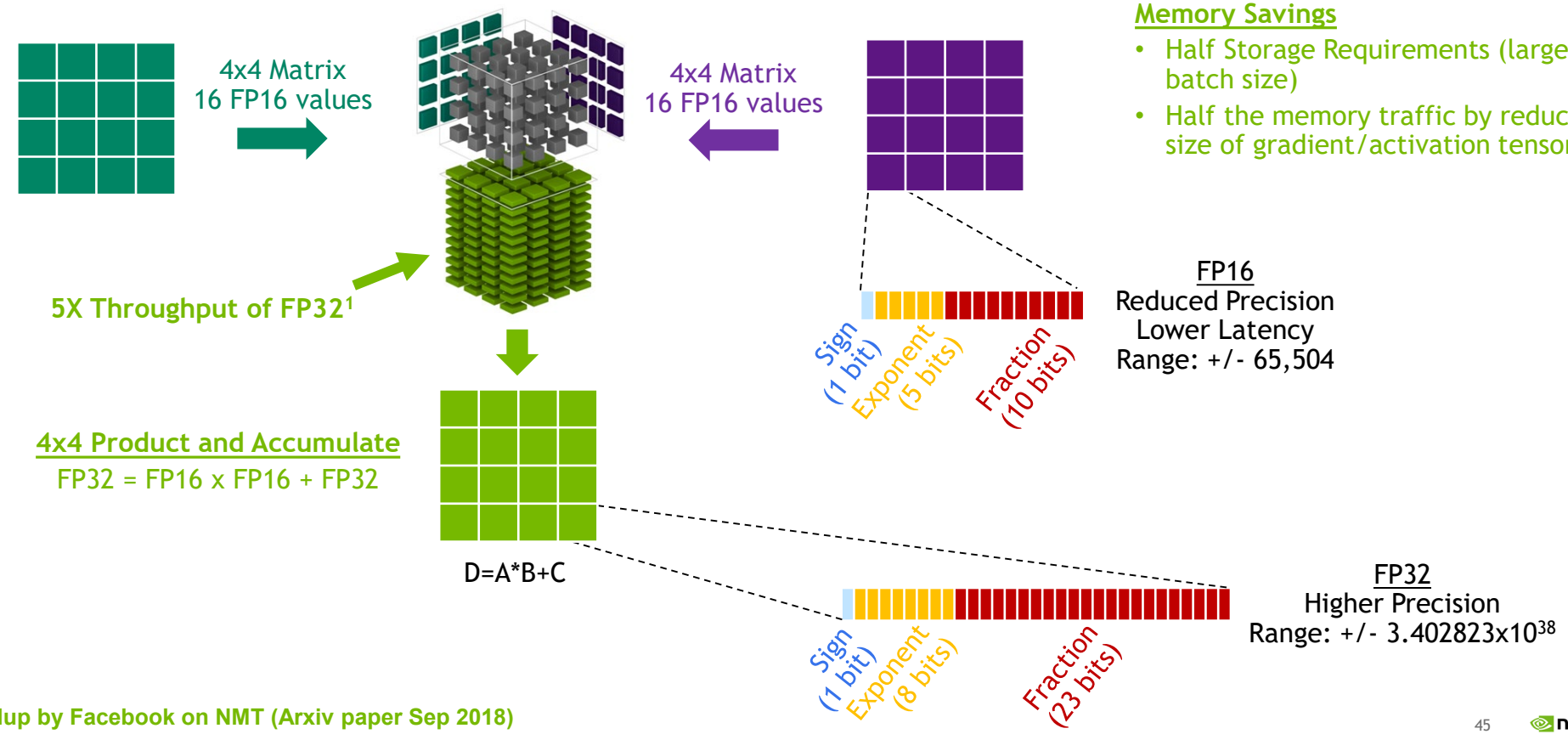
= 0.15625

And contrast with an IEEE half-precision 16-bit float:



TENSOR CORES BUILT FOR AI AND HPC

Mixed Precision Accelerator – Delivering Up To 5X Throughput of FP32[1]



Memory Savings

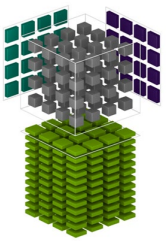
- Half Storage Requirements (larger batch size)
- Half the memory traffic by reducing size of gradient/activation tensors

[1]Fastest Tensor Core Speedup by Facebook on NMT (Arxiv paper Sep 2018)
https://arxiv.org/pdf/1806.00187.pdf

MASSIVE PARALLEL ACCELERATION

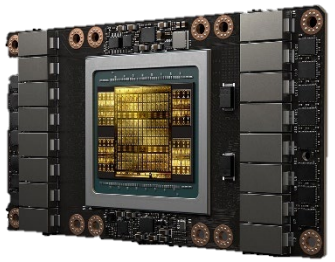
Volta and Turing GPUs

Tensor Core



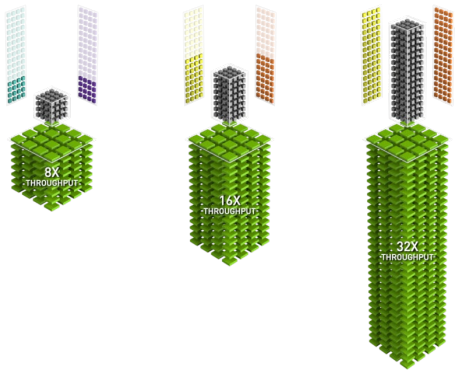
FP32/FP16

640 Tensor Cores



5120 CUDA Cores

Volta V100 GPU



FP32/FP16/INT8/INT4

320 Tensor Cores

2560 CUDA Cores

Turing T4 GPU

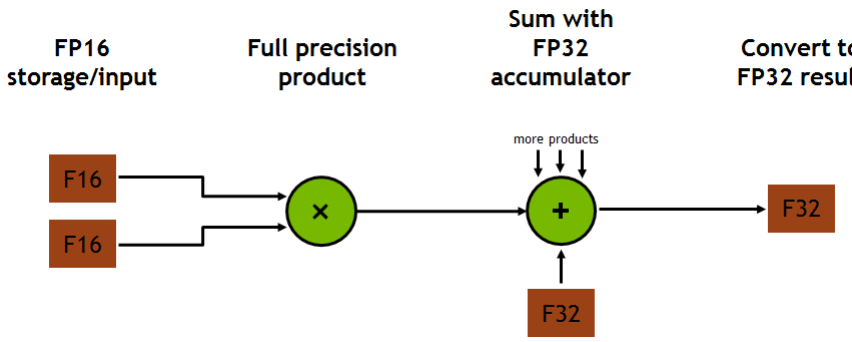
VOLTA

Tensor Cores

Each Tensor Core performs 64 floating point FMA mixed-precision operations per

8 Tensor Cores in an SM perform a total of 1024 floating point operations per clock.

Tensor Cores operate on FP16 input data with FP32 accumulation. The FP16 multiply results in a full precision result that is accumulated in FP32 operations with the other products in a given dot product for a 4x4x4 matrix multiply

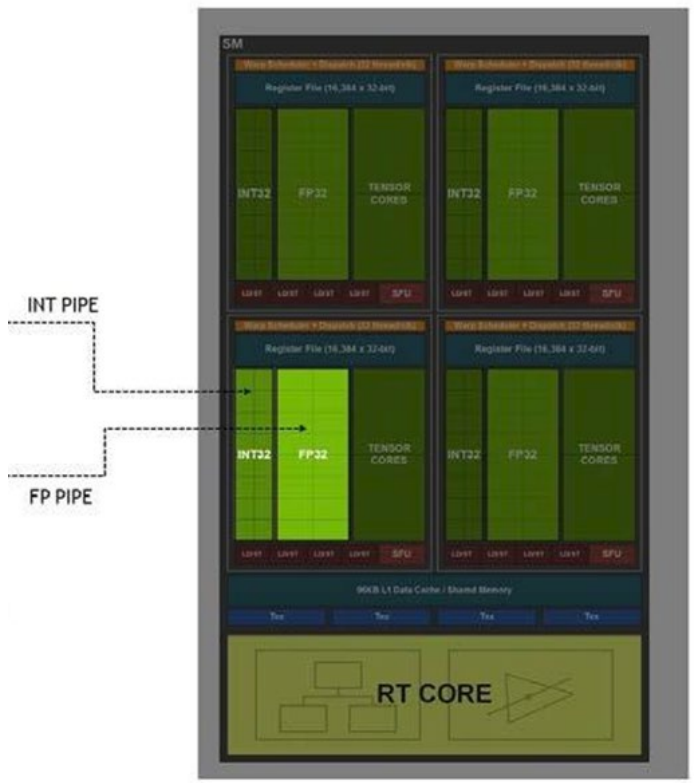


TURING

Tensor Cores

The TU102 GPU also features 144 FP64 units (two per SM), which are not depicted in this diagram. The FP64 TFLOP rate is 1/32nd the TFLOP rate of FP32 operations. The small number of FP64 hardware units are included to ensure any programs with FP64 code operates correctly.

The Turing SM supports Concurrent Execution of Floating Point and Integer Instructions



TENSOR CORES FOR HPC

Dig Deeper

https://devblogs.nvidia.com/tensor-cores-mixed-precision-scientific-computing/

Dongarra at SC'18 and Slides (Magma)

Harnessing Tensor Cores FP16 Arithmetic to Accelerate Linear Solvers and HPC Scientific Applications

Programming Tensor Cores using NVIDIA's CUDA accelerated-computing API

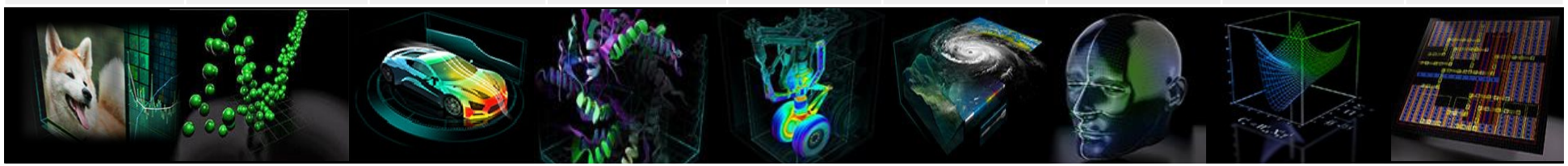
Tensor Cores in the Volta architecture

https://devblogs.nvidia.com/nvidia-turing-architecture-in-depth/

NVIDIA CUDA-X UPDATES

SOFTWARE TO DELIVER ACCELERATION FOR HPC & AI APPS; 500+ NEW UPDATES

Machine Learning & Deep Learning	Computational Physics & Chemistry	Computational Fluid Dynamics	Life Sciences & Bioinformatics	Structural Mechanics	Weather & Climate	Geoscience, Seismology & Imaging	Numerical Analytics	Electronic Design Automation



600+ Apps

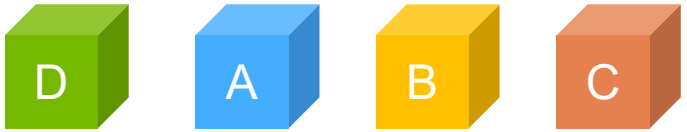
Linear Algebra | Parallel Algorithms | Signal Processing | Deep Learning | Machine Learning | Visualization

CUDA-X HPC & AI

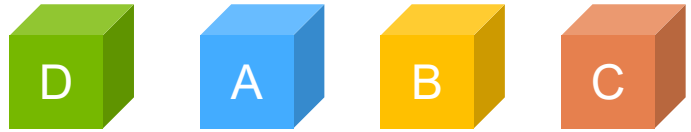
40+ GPU Acceleration Libraries

CUDA

Desktop Development	Data Center	Supercomputers	GPU-Accelerated Cloud



cuTENSOR



A New High Performance CUDA Library for Tensor Primitives

- Tensor Contractions
- Elementwise Operations
- Mixed Precision
- Coming Soon
  - Tensor Reductions
  - Out-of-core Contractions
  - Tensor Decompositions
- Pre-release version available
  http://developer.nvidia.com/c

```
cutensorStatus_t cutensorCreateTensorDescriptor ( cutensorTensorDescriptor_t* desc,
                                                  unsigned int numModes,
                                                  const int64_t *extent,
                                                  const int64_t *stride,
                                                  cudaDataType_t dataType,
                                                  cutensorOperator_t unaryOp );


cutensorStatus_t cutensorContraction (cuTensorHandle_t handle,
      const void* alpha, const void *A, const cutensorTensorDescriptor *descA, const int modeA[],
                         const void *B, const cutensorTensorDescriptor *descB, const int modeB[],
      const void* beta,  const void *C, const cutensorTensorDescriptor *descC, const int modeC[],
                               void *D, const cutensorTensorDescriptor *descD, const int modeD[],
      cutensorOperator_t opOut, cudaDataType_t typeCompute, cutensorAlgo_t algo,
      void* workspace, size_t workspaceSize, cudaStream_t stream );


cutensorStatus_t cutensorElementwiseTrinary ( cuTensorHandle_t handle,
      const void* alpha, const void *A, const cutensorTensorDescripto r *descA, const int modeA[],
      const void* beta,  const void *B, const cutensorTensorDescriptor *descB, const int modeB[],
      const void* beta,  const void *C, const cutensorTensorDescriptor *descC, const int modeC[],
                               void *D, const cutensorTensorDescriptor *descD, const int modeD[],
      cutensorOperator_t opAB, cutensorOperator_t opABC, cudaDataType_t typeCompute, cudaStream_t stream );
```

cuSPARSE

New Improved Sparse BLAS APIs

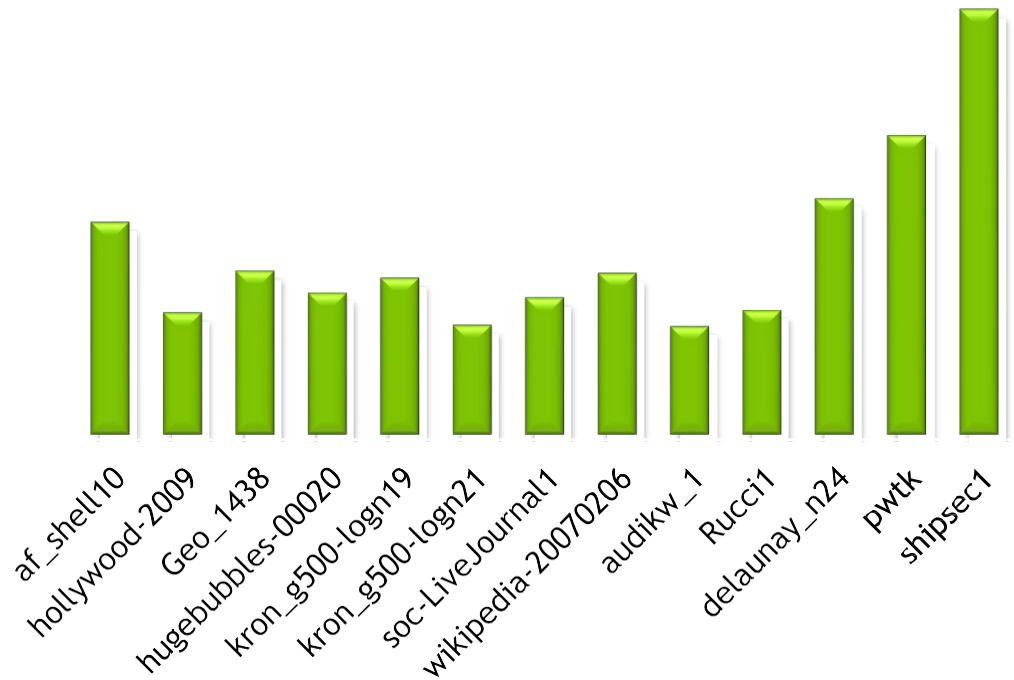
Introduced generic APIs with improved performance

- SpVV - Sparse Vector Dense Vector Multiplication
- SpMV – Sparse Matrix Dense Vector Multiplication
- SpMM – Sparse Matrix Dense Matrix Multiplication

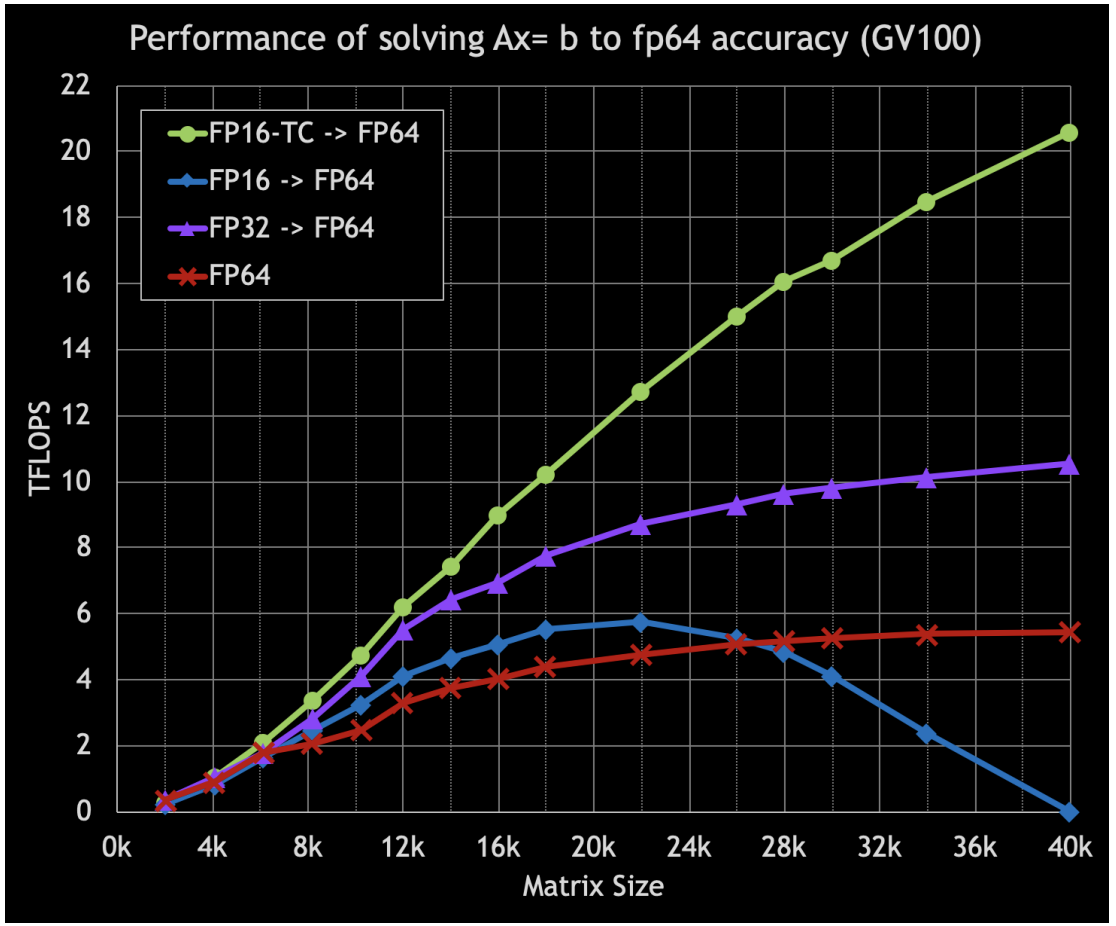
Coming Soon

- SpGEMM – Sparse Matrix Sparse Matrix Multiplication

```
cusparseStatus_t
cusparseSpMM(cusparseHandle_                handle,
t                                           transA,
             cusparseOperation_t            transB,
             cusparseOperation_t            alpha,
             const void*                    matA,
             const cusparseSpMatDescr_t     matB,
             const void*                    beta,
             cusparseDenseMatDescr_t        matC,
             cudaDataType                   computeType,
             cusparseSpMMAlg_t              alg,
             void*                          externalBuffer)
```



TENSOR CORE-ACCELERATED ITERATIVE REFINEMENT SOLVERS



Productization Plans

LU Solver	• ~August 2019 • Real & Complex FP32 & FP64
Cholesky Solver	• ~October-November 2019 • Real & Complex FP32 & FP64
QR Solver	• ~October-November 2019 • Real & Complex FP 32 & FP64

AUTOMATIC MIXED PRECISION

MIXED PRECISION TRAINING AND INFERENCE

Utilizing Volta and Turing Tensor Cores

Tensor core math accelerates matrix-multiply-and-add operations – 8x higher throughput

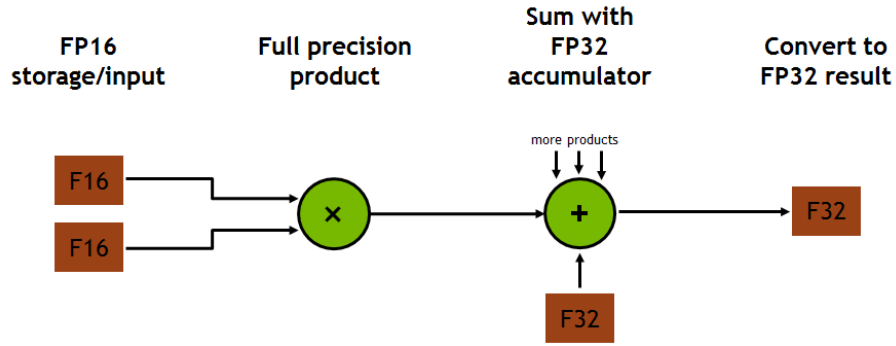
Computes many dot-products in parallel

Reduce layer memory bandwidth pressure

Reduce memory consumption

- Activation and gradient tensor sizes halved
- Enable larger minibatch or input sizes

Configurable accumulation and output for FP32 or FP16



Mixed Precision - Same hyperparameters and learning rate schedule as FP32
ILSVRC12 classification top-1 accuracy.
(Sharan Narang, Paulius Micikevicius *et al.*, "Mixed Precision Training", ICLR 2018)

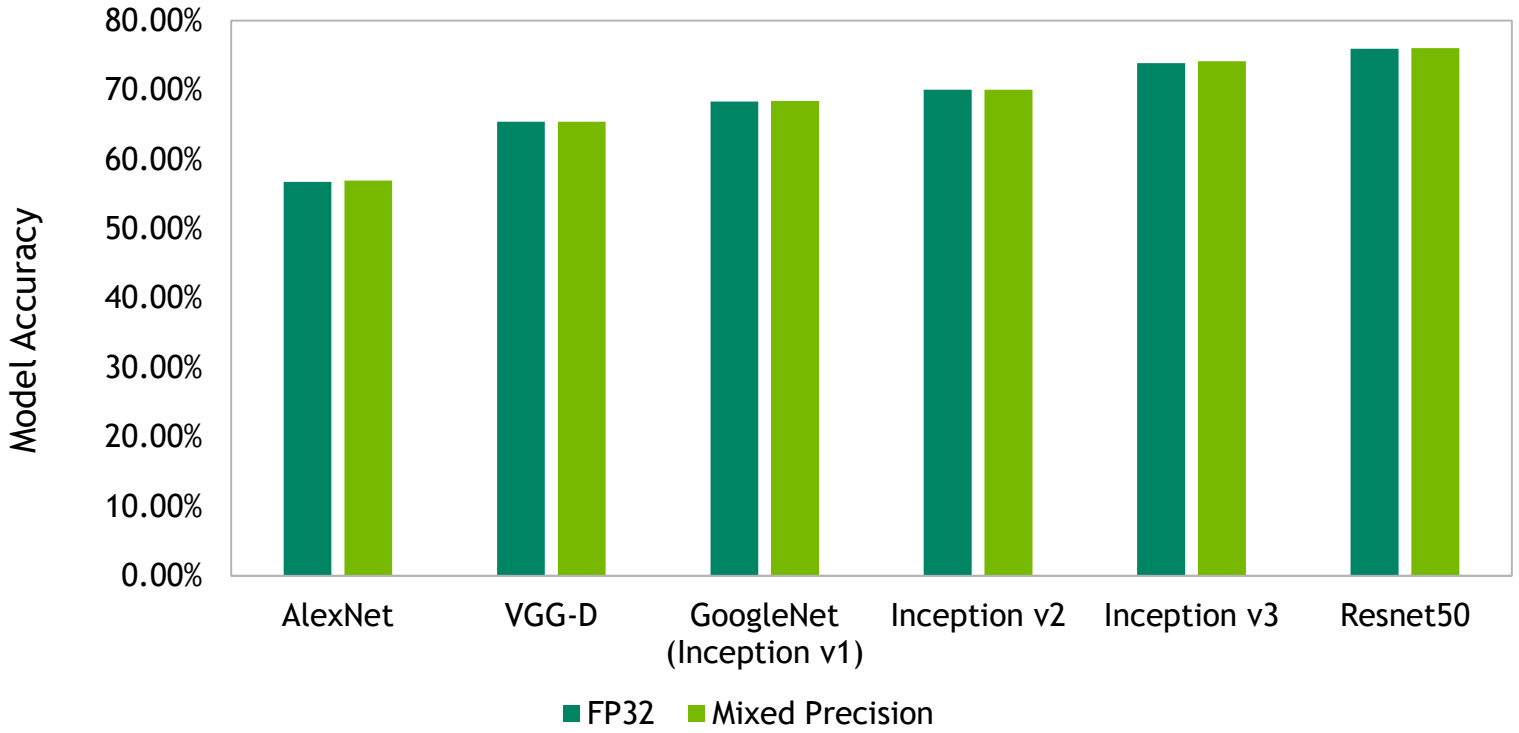
MIXED PRECISION CONSIDERATIONS

When to choose single or half precision math?

	FP16	FP32
Updating weights • Suggested approach: use FP32, keep master in FP32 and make copy in FP16 for fwd/bwd passes	W !>> lr * dW Convnets	W >> lr * dW Recurrent
Activations	ReLU, Sigmoid, Scale, Add… Dynamic range is limited	|f(x)| >> |x| Exp, Square, Log… Normalization or loss layer
Reductions • L1 + L2 norm, softmax • May depend on framework input/output type support	Not suggested	Suggested Avoid overflow, op memory limited Probability layers
Convolutions, MatMul • FP16 input/output storage • Keep MatMul M,N,K multiples of 8	Inference in FP16	Training in FP32

MIXED PRECISION MAINTAINS ACCURACY

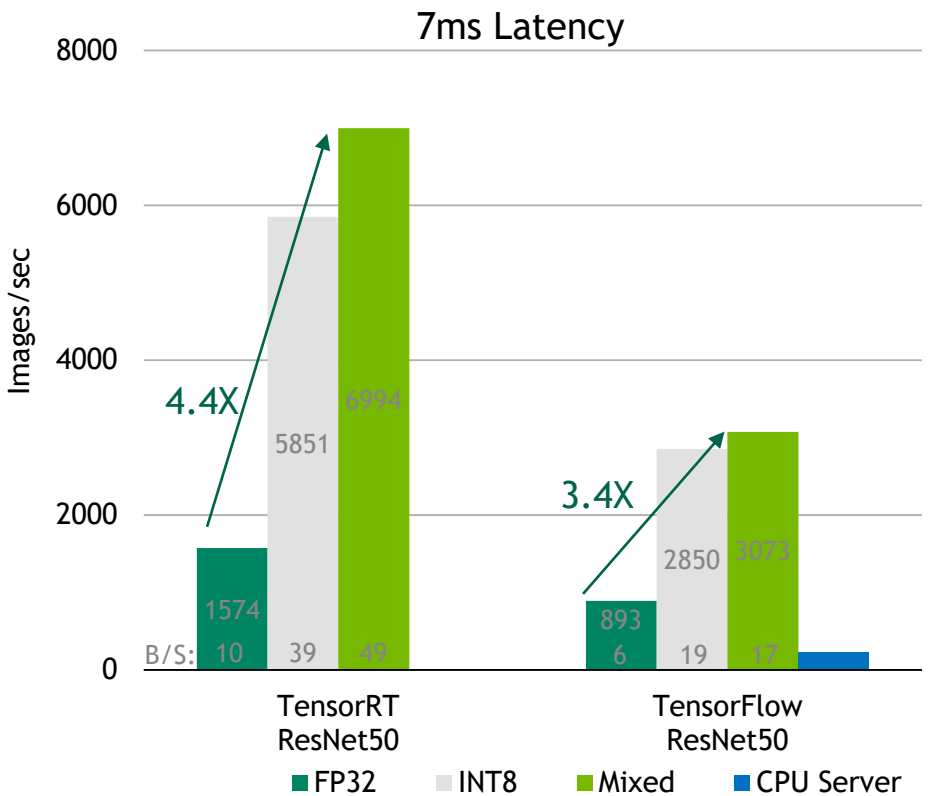
Benefit From Higher Throughput Without Compromise

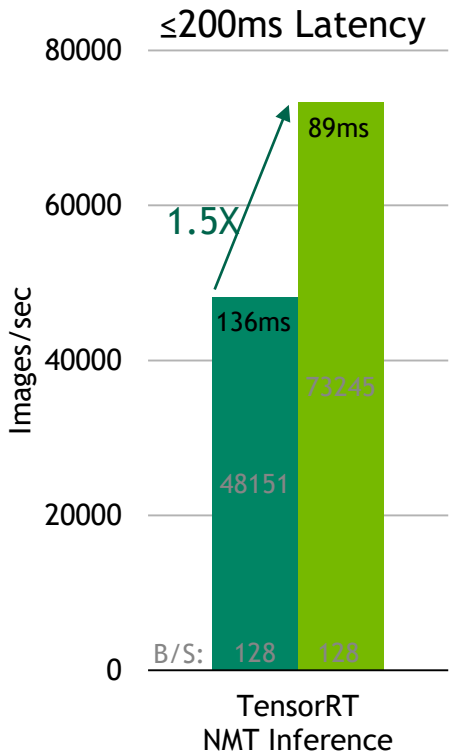


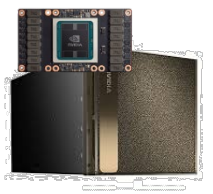
Mixed Precision - Same hyperparameters and learning rate schedule as FP32
ILSVRC12 classification top-1 accuracy.
(Sharan Narang, Paulius Micikevicius *et al.*, "Mixed Precision Training", ICLR 2018)

OVER 4X SPEED UP IN V100 SERVER INFERENCE WITH MIXED PRECISION

Mixed Precision Across Diverse AI Use Cases With Accuracy Unchanged







Scale-up Server
1x V100 GPU

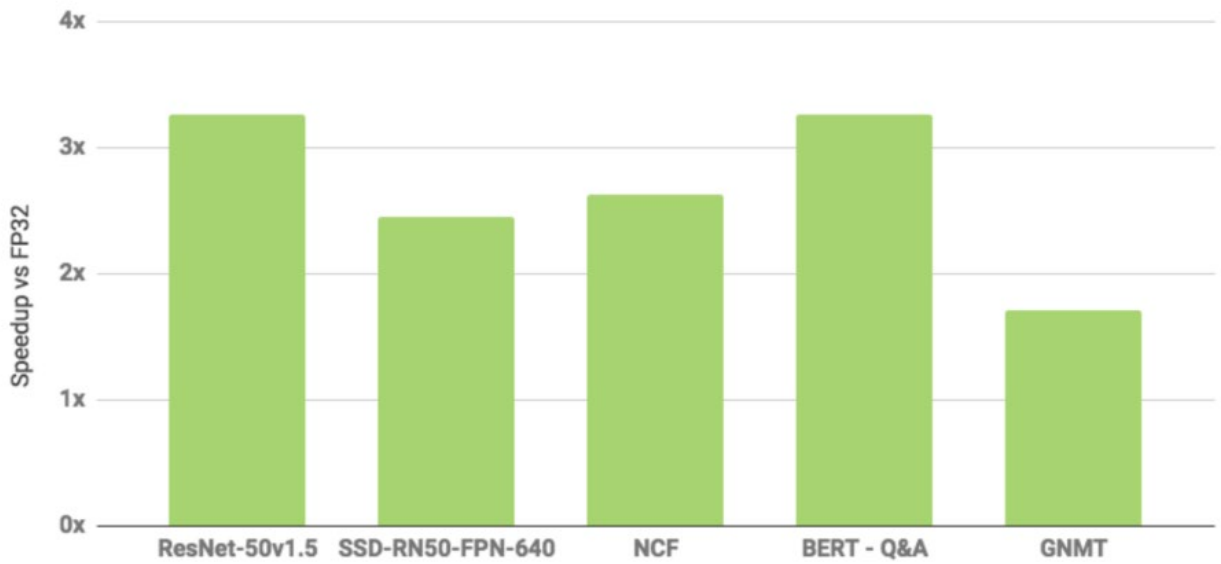
Higher Throughput

Lower TCO

CPU Comparison: Gold6140, OpenVINO, BS:1, Latency 4ms, 227 images/sec
NVIDIA Server: DGX-1 | GPU: 1x V100-SXM2-16GB | CPU: E5-2698v4
TensorRT: NMT Inference: 19.01_py3(FP32) 18.11_py3(Mixed), Dataset: WMT16 English-German | ResNet50: 18.12_py3(FP32,INT8), 19.01_py3(Mixed), Dataset: ImageNet2012
TensorFlow: ResNet50: 19.01_py3 Hybrid TRT(FP32), 18.11_py3 Hybrid TRT(INT8, Mixed), Dataset: ImageNet2012

AUTOMATIC MIXED PRECISION IN TENSORFLOW

Upto 3X Speedup



TensorFlow Medium Post: Automatic Mixed Precision in TensorFlow for Faster AI Training on NVIDIA GPUs

All models can be found at:
https://github.com/NVIDIA/DeepLearningExamples/tree/master/TensorFlow, except for ssd-rn50-fpn-640, which is here: https://github.com/tensorflow/models/tree/master/research/object_detection
All performance collected on 1xV100-16GB, except bert-squadqa on 1xV100-32GB.
Speedup is the ratio of time to train for a fixed number of epochs in single-precision and Automatic Mixed Precision. Number of epochs for each model was matching the literature or common practice (it was also confirmed that both training sessions achieved the same model accuracy).
Batch sizes measured as follows. rn50 (v1.5): 128 for FP32, 256 for AMP+XLA; ssd-rn50-fpn-640: 8 for FP32, 16 for AMP+XLA; ncf: 1M for FP32 and AMP+XLA; bert-squadqa: 4 for FP32, 10 for AMP+XLA; gnmt: 128 for FP32, 192 for AMP.

ENABLE AMP

Supported in common frameworks

TensorFlow

```
export TF_ENABLE_AUTO_MIXED_PRECISION=1
# or
os.environ['TF_ENABLE_AUTO_MIXED_PRECISION'] = '1'

opt =
tf.train.experimental.enable_mixed_precision_graph_re
write(opt)
```

PyTorch

```
model, optimizer = amp.initialize(model, optimizer,
opt_level="O1")

with amp.scale_loss(loss, optimizer) as scaled_loss:
  scaled_loss.backward()
```

https://developer.nvidia.com/automatic-mixed-precision

NVIDIA END-TO-END SOFTWARE STACK

Deep Learning Streamlined From Conception to Production at Scale

NVIDIA Optimized Models Across Multiple Frameworks


Computer Vision


Speech

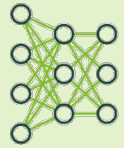

Recommender


GANs

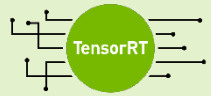
mxnet PYTORCH TensorFlow

...

Train

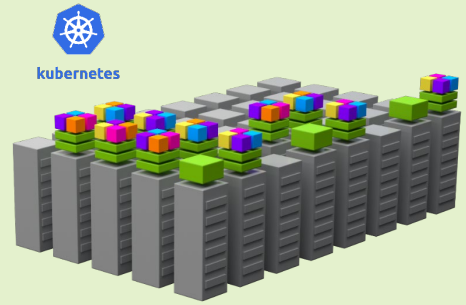


Inference



TensorRT

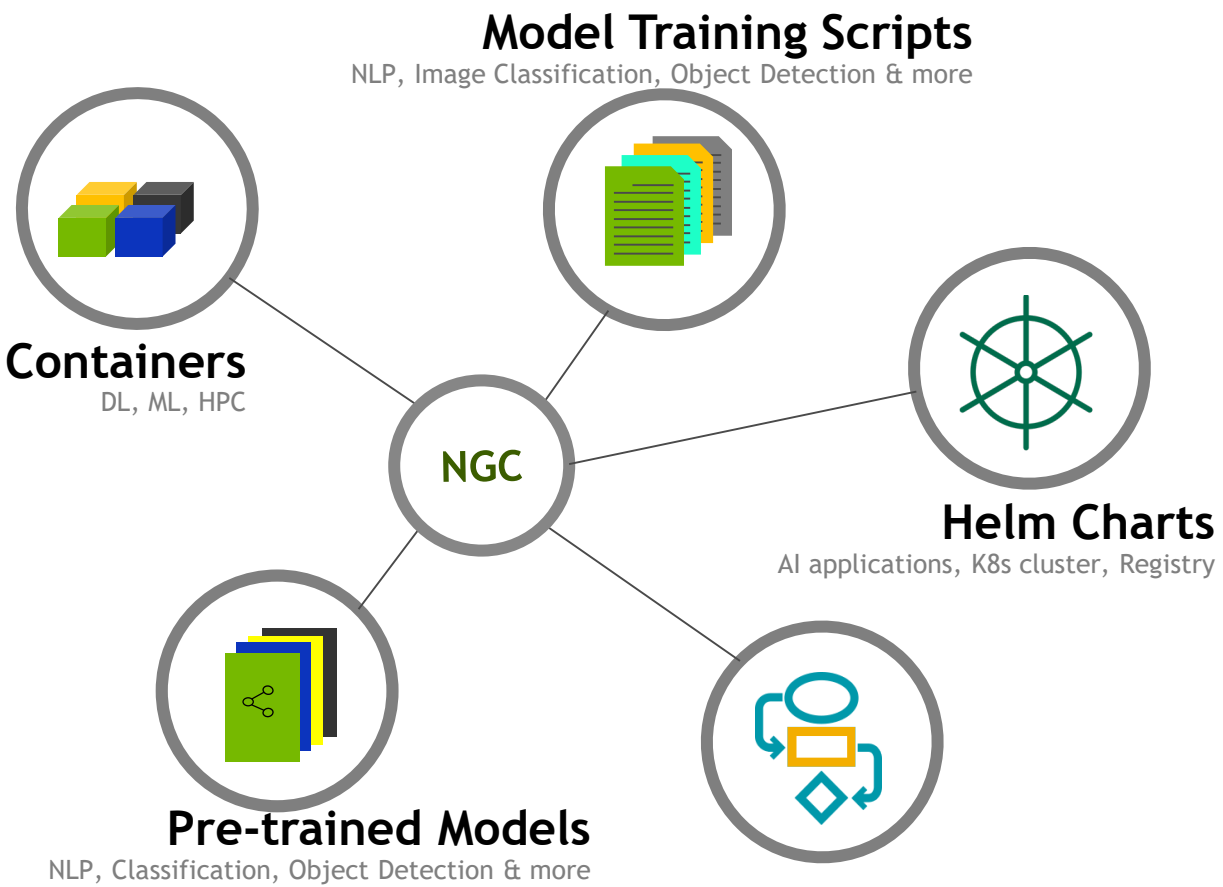
Scale



TRT Inference Server

NGC: GPU-OPTIMIZED SOFTWARE HUB

Simplifying DL, ML and HPC Workflows



Simplify Deployments

Innovate Faster

Deploy Anywhere

ANNOUNCING SUPPORT FOR ARM

ENERGY-EFFICIENT SUPERCOMPUTING

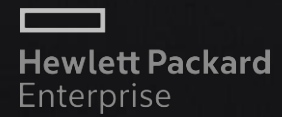
NVIDIA GPU Accelerated Computing Platform On ARM

Optimized CUDA-X HPC & AI Software Stack

CUDA, Development Tools and Compilers

Available End of 2019

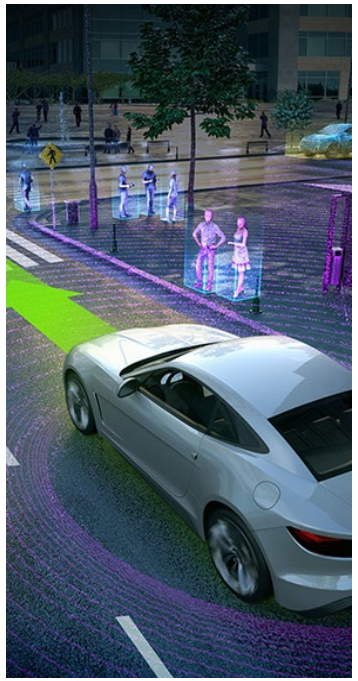
&




NVIDIA

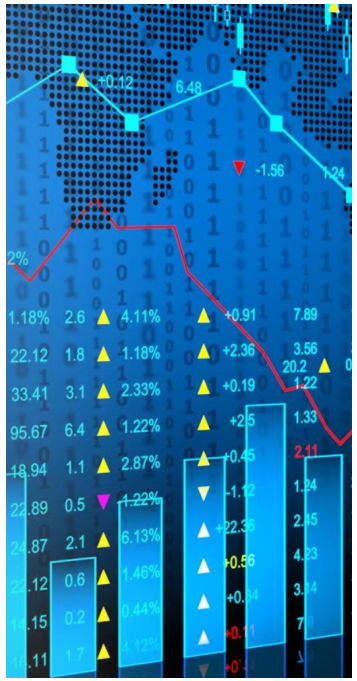

NVIDIA
NGC

GRAND CHALLENGES REQUIRE MASSIVE COMPUTING

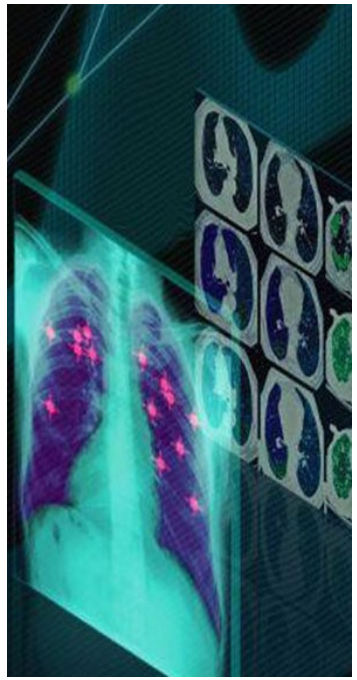


AUTONOMOUS DRIVING

ASTROPHYSICS



FRAUD DETECTION



MEDICAL IMAGING

NUCLEAR ENERGY

TELECOM

DIFFERENT ROLES. SAME GOALS.

Driving Productivity and Faster Time-to-Solutions

Data Scientists and Researchers

Eliminate mundane tasks, focus on science and research

Developers

Speed up development with existing building blocks

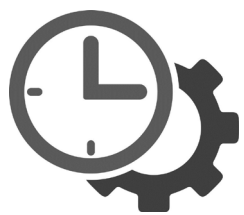
DevOps

Consistent & faster develop-to-production cycle

Sysadmins

Deliver appropriate deployment environments

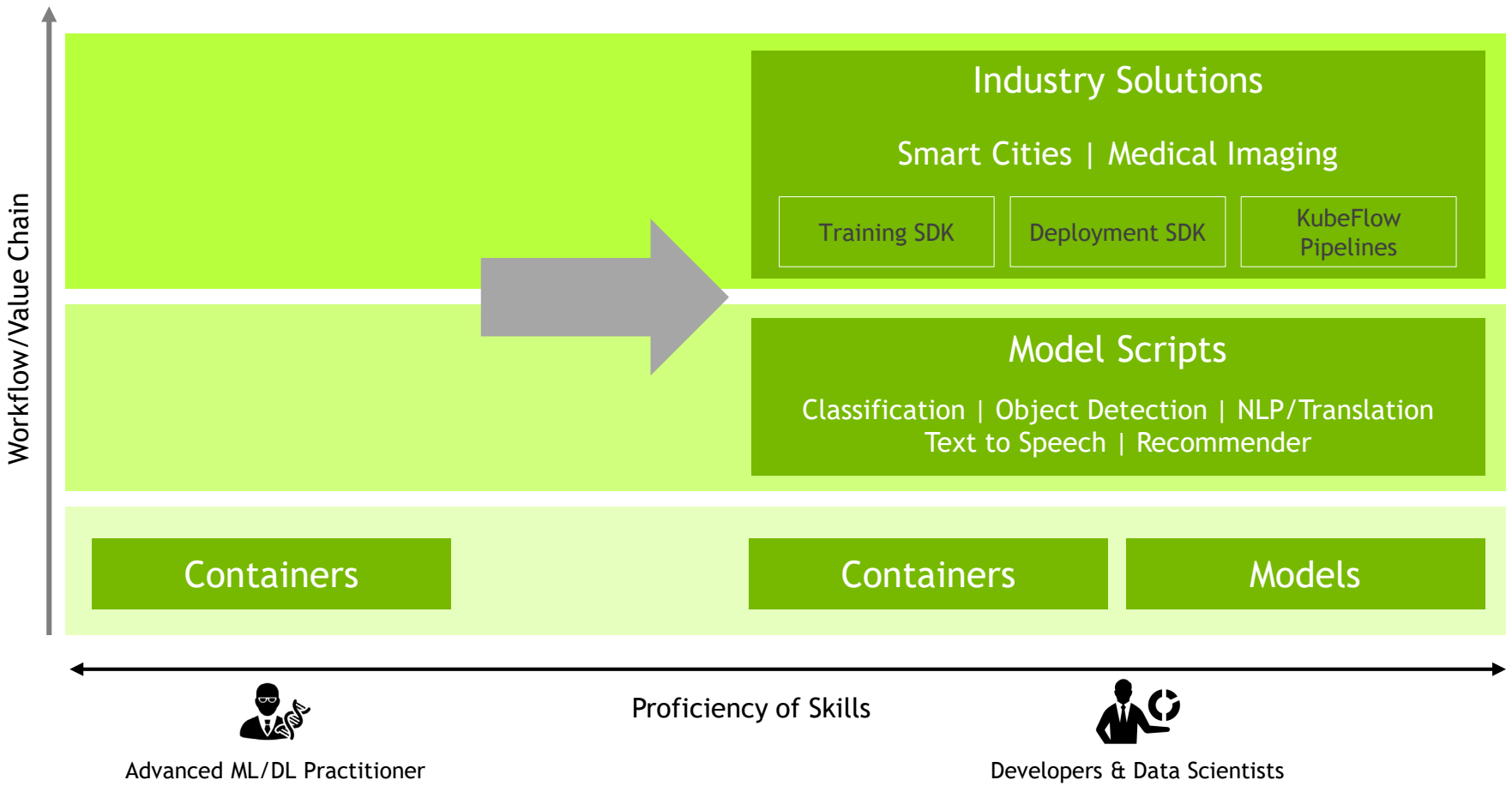
CHALLENGES UTILIZING AI & HPC SOFTWARE

EXPERTISE	INSTALLATION	OPTIMIZATION	PRODUCTIVITY	MAINTAINENCE
				
Building AI-centric solutions requires expertise	Complex, time consuming, and error-prone	Requires expertise to optimize framework performance	Users limited to older features and lower performance	IT can’t keep up with frequent software upgrades

NGC – SIMPLIFYING AI & HPC WORKFLOWS

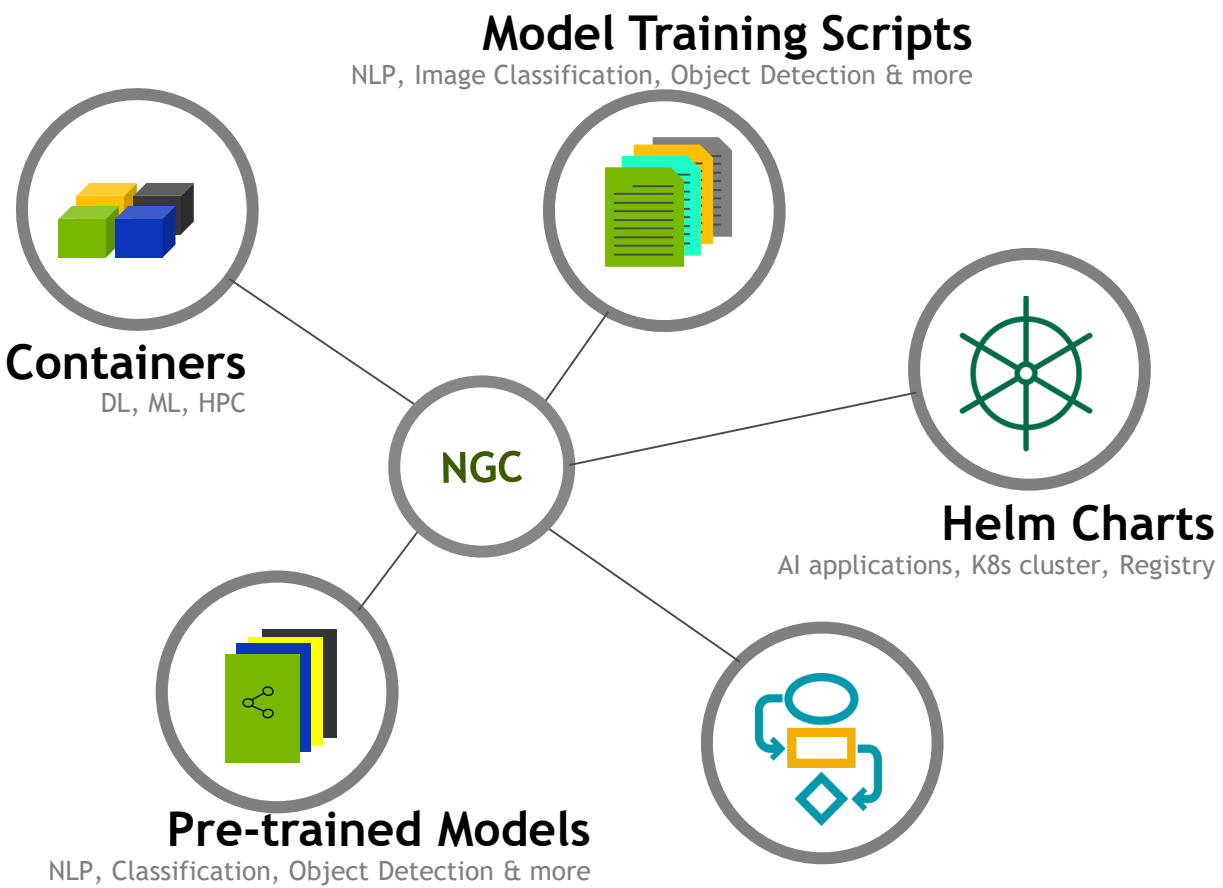
EMBEDDING EXPERTISE	FASTER DEPLOYMENTS	OPTIMIZED SOFTWARE	HIGHER PRODUCTIVITY	ZERO MAINTENANCE
				
Deliver greater value, faster	Eliminates installations. Simply Pull & Run the app	Key DL frameworks updated monthly for perf optimization	Better Insights and faster time-to-solution	Empowers users to deploy the latest versions with IT support

NGC: APPLICATIONS TO END-TO-END SOLUTIONS



NGC: GPU-OPTIMIZED SOFTWARE HUB

Simplifying DL, ML and HPC Workflows



Simplify Deployments

Innovate Faster

Deploy Anywhere

CONTAINERS

CONTAINERS: SIMPLIFYING WORKFLOWS

WHY CONTAINERS

Simplifies Deployments

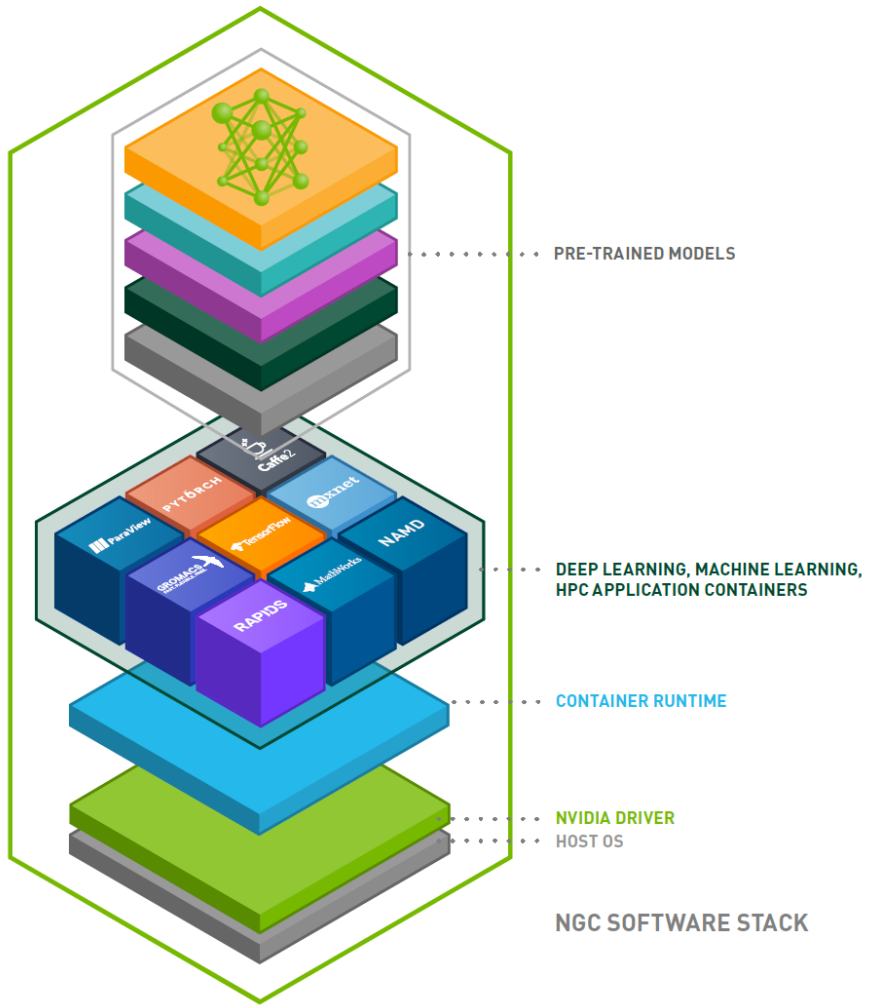
- Eliminates complex, time-consuming builds and installs

Get started in minutes

- Simply Pull & Run the app

Portable

- Deploy across various environments, from test to production with minimal changes



NGC CONTAINERS: ACCELERATING WORKFLOWS

WHY CONTAINERS

Simplifies Deployments

- Eliminates complex, time-consuming builds and installs

Get started in minutes

- Simply Pull & Run the app

Portable

- Deploy across various environments, from test to production with minimal changes

WHY NGC CONTAINERS

Optimized for Performance

- Monthly DL container releases offer latest features and superior performance on NVIDIA GPUs

Scalable Performance

- Supports multi-GPU & multi-node systems for scale-up & scale-out environments

Designed for Enterprise & HPC environments

- Supports Docker & Singularity runtimes

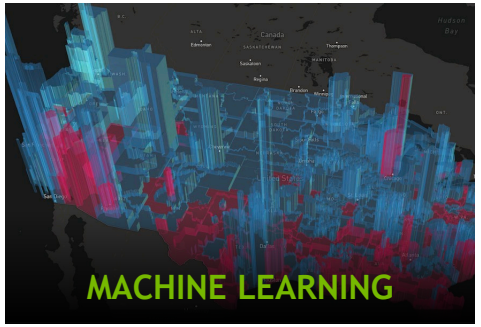
Run Anywhere

- Pascal/Volta/Turing-powered NVIDIA DGX, PCs, workstations, and servers
- From Core to the Edge
- On-Prem to Hybrid to Cloud

GPU-OPTIMIZED SOFTWARE CONTAINERS

Over 50 Containers on NGC

TensorFlow | PyTorch | more

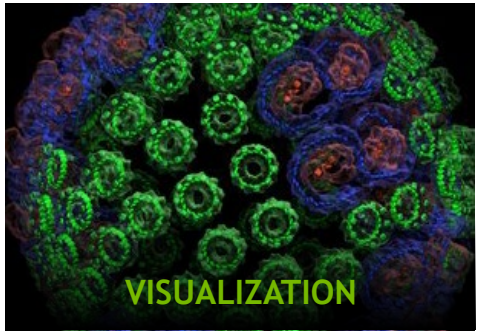


RAPIDS | H2O | more

TensorRT | DeepStream | more

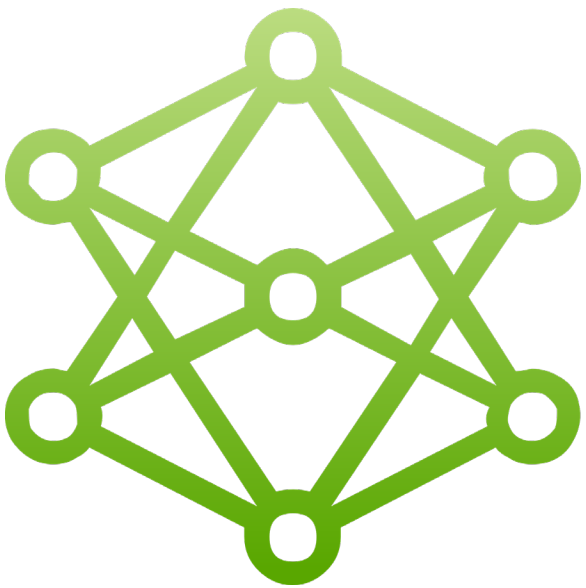
NAMD | GROMACS | more

Parabricks



ParaView | IndeX | more

THE NGC MODEL REGISTRY



Repository of Popular AI Models

- Starting point to retrain, prototype or benchmark against your own models
- Use As-Is or easily customize
- Private hosted registry for NGC Enterprise accounts to upload, share and version

DOMAIN SPECIFIC | INFERENCE-READY

PRE-TRAINED MODELS

- Domain specific for video analytics and medical imaging
- Use transfer learning and your own data to quickly create accurate AI
- Available models: Organ & tumor segmentation, x-ray classification, classification and object detection for video analytics

TENSORRT MODELS

- Ready for inference with Tensor Cores
- Precision: INT8, FP16, FP32
- Optimized for multiple GPU architectures
- Available Models: ResNet50, VGG16, InceptionV1, Mobilenet

LEARN | BUILD | OPTIMIZE | DEPLOY

MODEL SCRIPTS

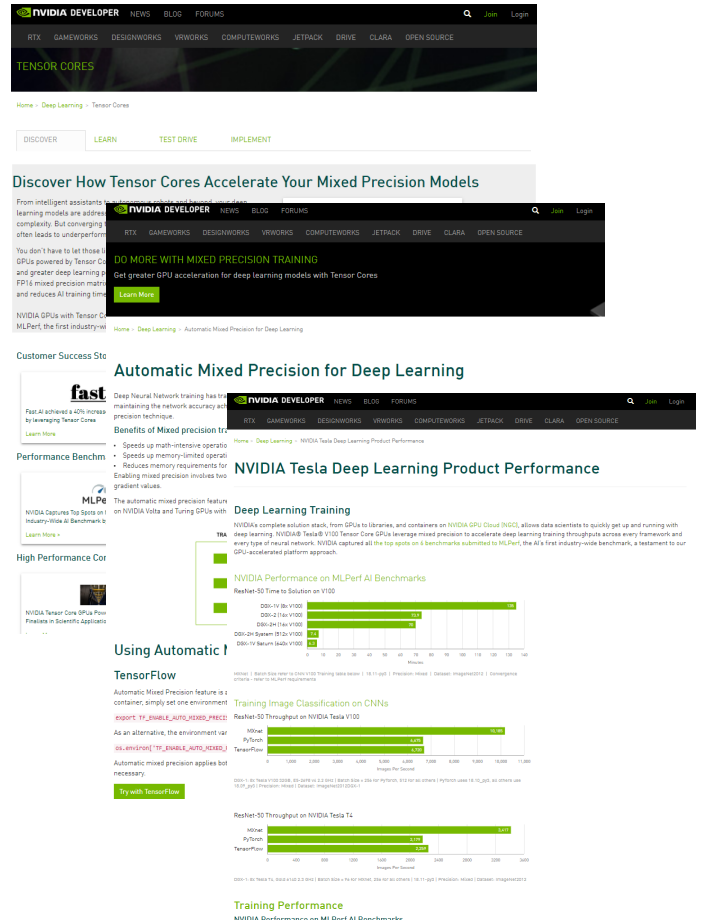
- Best practices for training models
- Faster Performance with Optimized Libraries and Tensor Cores
- State-of-the-Art Accuracy

Image Recognition	ResNet-50
Natural Language Processing	Bert, Transformer
Object Detection	SSD
Recommendation	Neural Collaborative Filtering
Segmentation	Mask R-CNN, UNET-Industrial
Speech	Tacotron, WaveGlow
Translation	GNMT

IMPORTANT WEB LINKS

Share and Promote

- Tensor Cores for Developers (Developer Product Page)
- Automatic Mixed Precision in DL frameworks for Developers (Developer Product Page)
- Deep Learning Examples on Developer Zone (Developer page for tensor core examples)
- Mixed Precision Developer Guide (Developer guide)
- NVIDIA Tesla Deep Learning Product Performance (Get the latest Tesla DL performance info)
- NVIDIA NGC Model Scripts (Tensor Core optimized examples on NGC)



TECHNICAL RESOURCES

Learn, Share and Promote

- Video: Tensor Cores in a Nutshell
- Webinar: Automate Mixed Precision in PyTorch
- DevBlog: ML Perf
- DevBlog: Video Series: Mixed Precision training techniques using Tensor Cores for Deep Learning
- DevBlog: Using Tensor Cores for Mixed Precision Scientific Computing
- DevBlog: New Optimizations to Accelerate Deep Learning Training on NVIDIA GPUs
- DevBlog: Tools for easy mixed precision in PyTorch
- DevBlog: Automatic Mixed Precision in TensorFlow for Faster AI Training on NVIDIA GPUs (TF Medium)
- DevBlog: Automatic Mixed Precision for NVIDIA Tensor Core Architecture in TensorFlow
- DevBlog: GTC on-demand: Real world examples with mixed precision training
- **DevBlog**: Nsight https://developer.nvidia.com/tools-overview

GTC SESSION RECORDINGS 2019

Recommended GTC On-Demand Talks

Overview:

- Mixed precision training with Deep Neural Networks
- Text-to-speech Overview of latest research using Tacotron and Waveglow

PyTorch:

- Automatic Mixed Precision in PyTorch
- Taking advantage of mixed precision to accelerate training in PyTorch

TensorFlow:

- New automated mixed-precision tools for TensorFlow training
- Automatic mixed precision tools for TensorFlow Training

MXNet:

- MXNet Computer Vision and NLP Models Accelerated by Tensor Cores
- MXNet Computer Vision and Natural Language Processing Models Accelerated with NVIDIA Tensor Cores


NVIDIA