ESPM2: 2022 ACM/IEEE 7th International Workshop on Extreme Scale Programming Models and Middleware

Performance Results at Exascale in HPC and ML

Nicholas Malaya, Principal Engineer, AMD

Nicholas Malaya is a Principal Engineer at AMD, where he is AMD's technical lead for exascale application performance. Nick's research interests include HPC, computational fluid dynamics, Bayesian inference, and ML/AI. He received his PhD from the University of Texas. Before that, he double majored in Physics and Mathematics at Georgetown University, where he received the Treado medal. In his copious spare time he enjoys motorcycles, long distance running, wine, and spending time with his wife and children.

Performance Results at Exascale in HPC and ML

Nick Malaya

Extreme Scale Programming Models and Middleware 2022

November 14, 2022

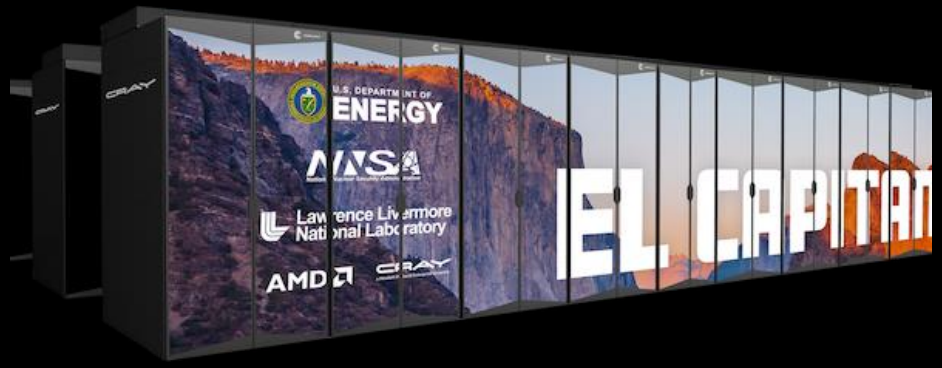
SC22

The Exascale Era is Here

OAK RIDGE NATIONAL LAB
FRONTIER

AMD
EPYC | INSTINCT

LAWRENCE LIVERMORE NATIONAL LAB
EL CAPITAN

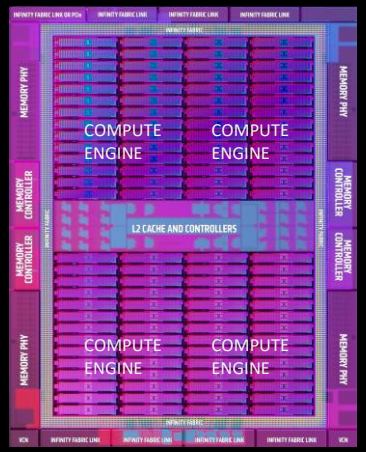


Frontier at-a-glance

- Deployed at Oak Ridge National Laboratory
- HPE Slingshot Interconnect w/ 100 GB/s bandwidth
- 9,408 nodes in 74 HPE Cray EX cabinets
- 1.685 EF of peak double precision compute performance
  - Achieved 1.1 EF in 21 MW of power

AMD INSTINCT™ MI250X ACCELERATOR

TSMC 6NM TECHNOLOGY	110 CU PER GRAPHICS COMPUTE DIE
4 MATRIX CORES PER COMPUTE UNIT	MATRIX CORES ENHANCED FOR HPC
8 INFINITY FABRIC LINKS PER DIE	SPECIAL FP32 OPS FOR DOUBLE THROUGHPUT



FRONTIER NODE AT A GLANCE

- 1x Optimized 3rd Gen AMD EPYC™ CPU (64 core)
- 4x AMD Instinct™ MI250X accelerators
  - Direct Attached to the NIC
- Coherent connectivity
  - Via Infinity Fabric™ interconnect
  - Tightly integrated
  - Unified memory space

MOTIVATION

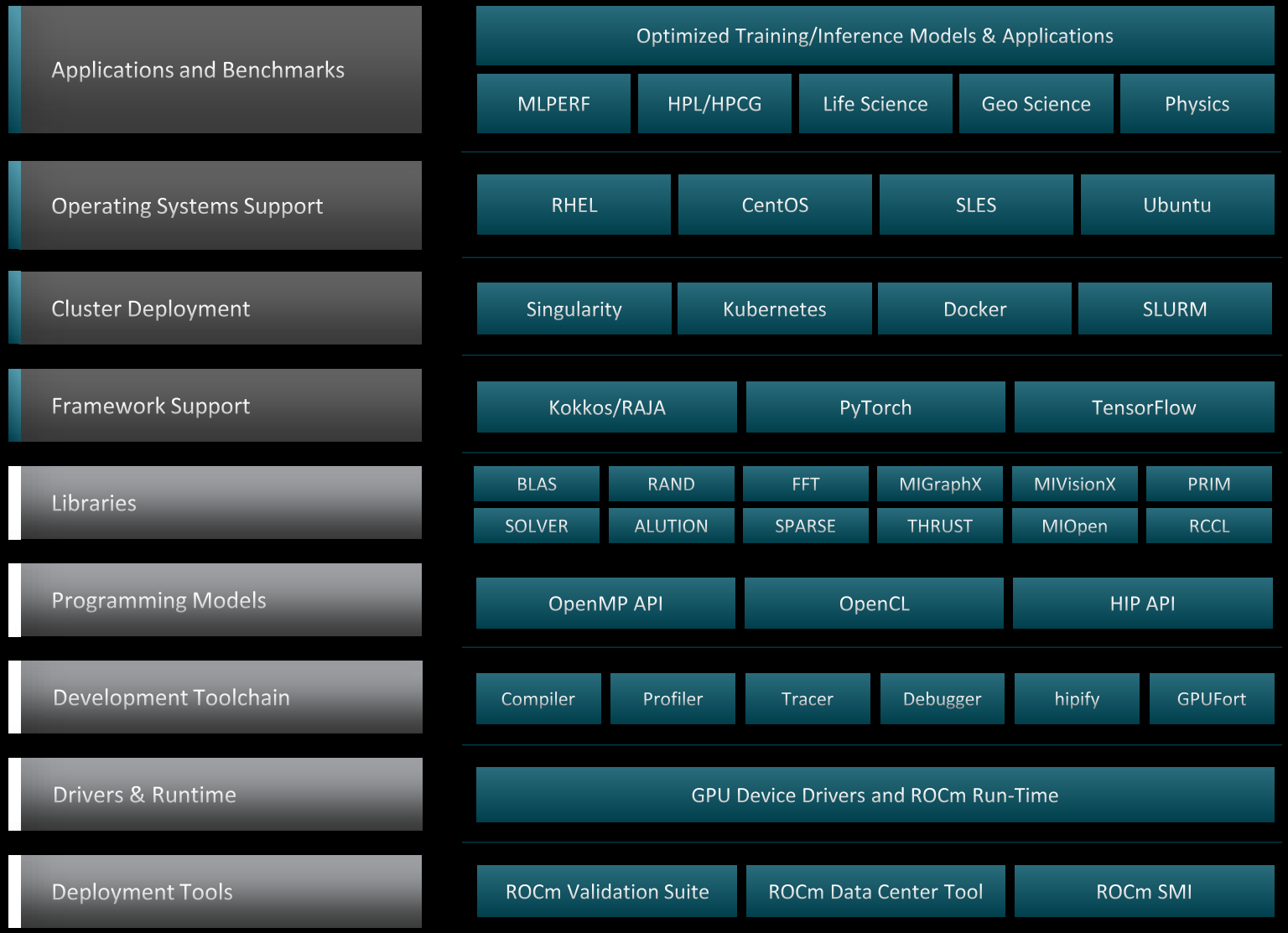
- We have an Exascale computer
  - What are we able to stay today ?
  - What are the implications on extreme programming models and future designs?

- Agenda:
  - Software and programming models on Frontier
  - Coherent programming and easing ‘time to acceleration’
  - Application Performance and Scalability

OPEN SOFTWARE PLATFORM FOR GPU COMPUTE

AMD ROCm

- Full Stack Support for HPC and ML
- Open Source for Collaboration with Partners and 3rd parties



Pillars of Exascale Software

OpenMP

Established Leader in Compiler Directives

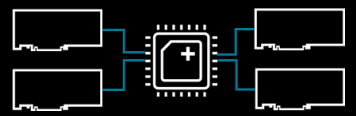
HIP

Open & Portable GPU Programming

TensorFlow PYTORCH

ML Frameworks

100% Open, Portable Software on Community Standards



Unified CPU + GPU Tools

Programming in CUDA & HIP

CUDA VECTOR ADD

```
__global__ void add(int n, double *x, double *y)
{
  int index = blockIdx.x * blockDim.x + threadIdx.x;
  int stride = blockDim.x * gridDim.x;
  for (int i = index; i < n; i += stride)
    {
    y[i] = x[i] + y[i];
    }
}
```

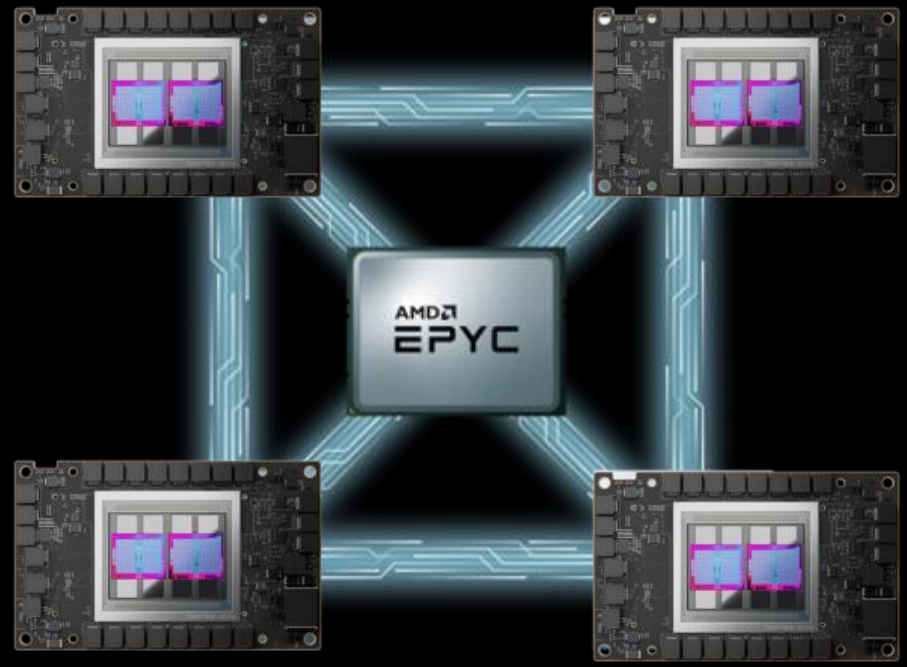
HIP VECTOR ADD

```
__global__ void add(int n, double *x, double *y)
{
  int index = blockIdx.x * blockDim.x + threadIdx.x;
  int stride = blockDim.x * gridDim.x;
  for (int i = index; i < n; i += stride)
    {
    y[i] = x[i] + y[i];
    }
}
```

KERNELS ARE SYNTACTICALLY IDENTICAL

COHERENT PROGRAMMING

- 1x Optimized 3rd Gen AMD EPYC™ CPU (64 core)
- 4x AMD Instinct™ MI250X accelerators
  - Direct Attached to the NIC
- Coherent connectivity
  - Via Infinity Fabric™ interconnect
  - Tightly integrated
  - Unified memory space



COHERENT PROGRAMMING MODEL: SIMPLICITY

CPU CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);


for (int i=0; i<M; i++) //initialize
   in_h[i] = …;

cpu_func(in_d, out_d, M);


for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

GPU CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);
hipMalloc(&in_d, Msize);
hipMalloc(&out_d, Msize);

for (int i=0; i<M; i++) //initialize
   in_h[i] = …;
hipMemcpy(in_d,in_h,Msize);
gpu_func<< >>(in_d, out_d, M);
hipDeviceSynchronize();
hipMemcpy(out_h,out_d,Msize);

for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

COHERENT CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);

for (int i=0; i<M; i++) //initialize
   in_h[i] = …;

gpu_func<< >>(in_h, out_h, M);
hipDeviceSynchronize();

for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

- GPU memory allocation on Device
- Explicit memory management between CPU & GPU
- Synchronization Barrier

COHERENT PROGRAMMING MODEL: PERFORMANCE

COHERENT CODE

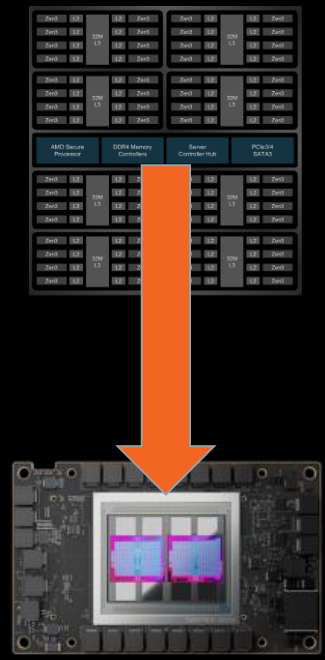
```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);



for (int i=0; i<M; i++) //initialize
   in_h[i] = …;

gpu_func<< >>(in_d, out_d, M);
hipDeviceSynchronize();


for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```



Operation	MI250X (MCM)
Coherent access over Infinity Fabric	56 GB/s

- ~~GPU memory allocation on Device~~
- ~~Explicit memory management between CPU & GPU~~
- Synchronization Barrier

OPENMP TARGET OFFLOAD

COHERENT CODE

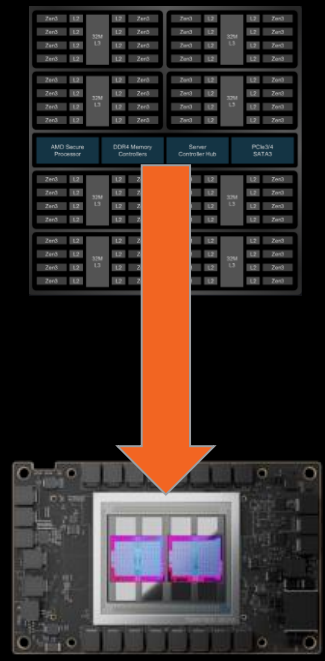
```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);



for (int i=0; i<M; i++) //initialize
   in_h[i] = …;

#pragma omp requires unified_shared_memory
#pragma omp target
{
…
}


for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```



- Runtime knows it can omit copies and map clauses
- (Implicit) Synchronization Barrier

GPU-AWARE MPI

- Frontier directly attaches AMD Instinct™ MI250x Accelerator to the Slingshot Network
  - Minimize the role of the CPU in the control path
  - Lowest latency for network message passing is from GPU HBM memory

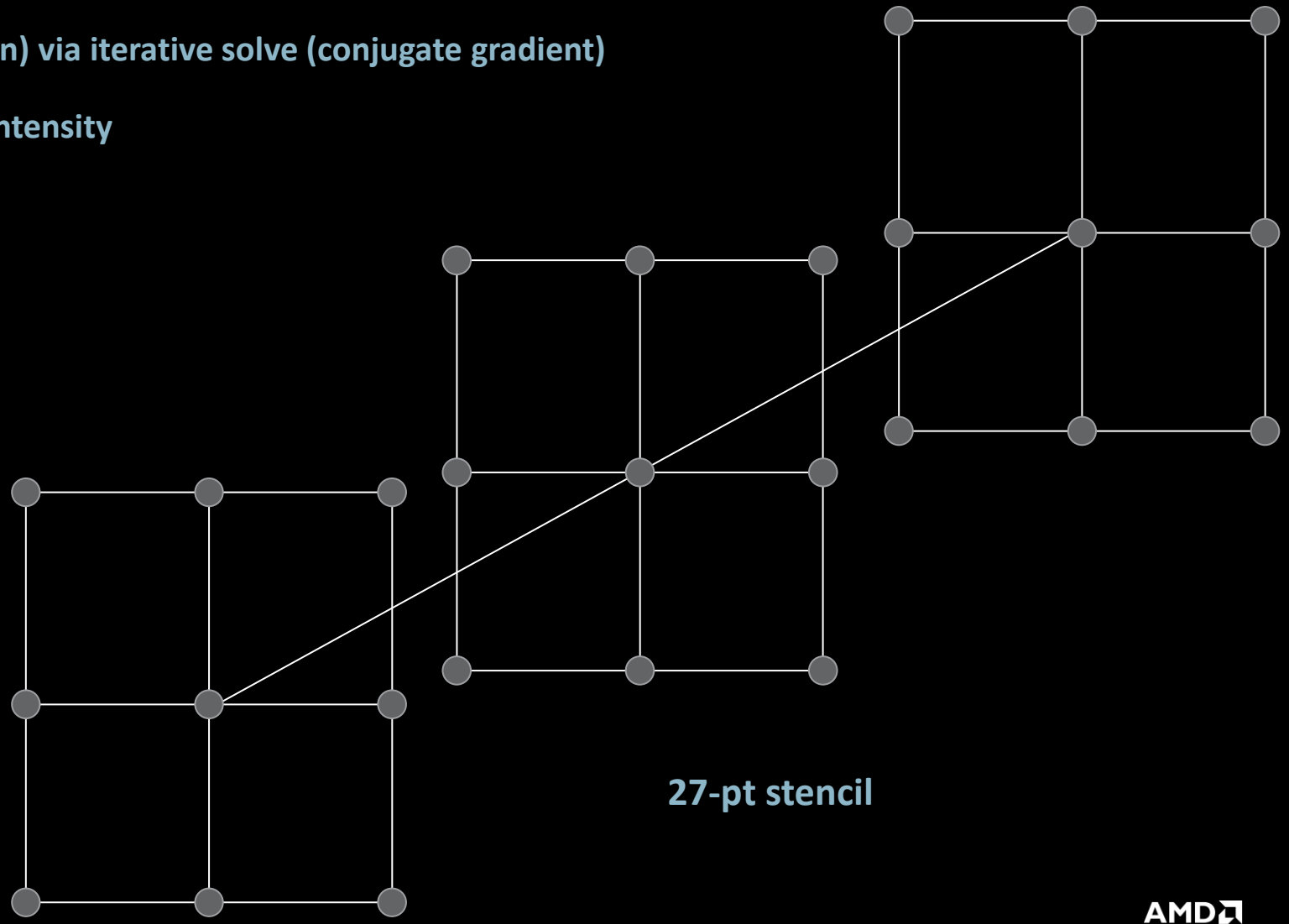
- For example,

```
hipMalloc(&device_buffer, Msize);
init_data<< >>(device_buffer, Msize);  // launch GPU kernel
hipDeviceSynchronize();
MPI_Sendrecv(&device_buffer, Msize, MPI_FLOAT, … , MPI_COMM_WORLD, MPI_STATUS_IGNORE);
```

- Requires: HPE Cray MPICH

THE HPCG BENCHMARK

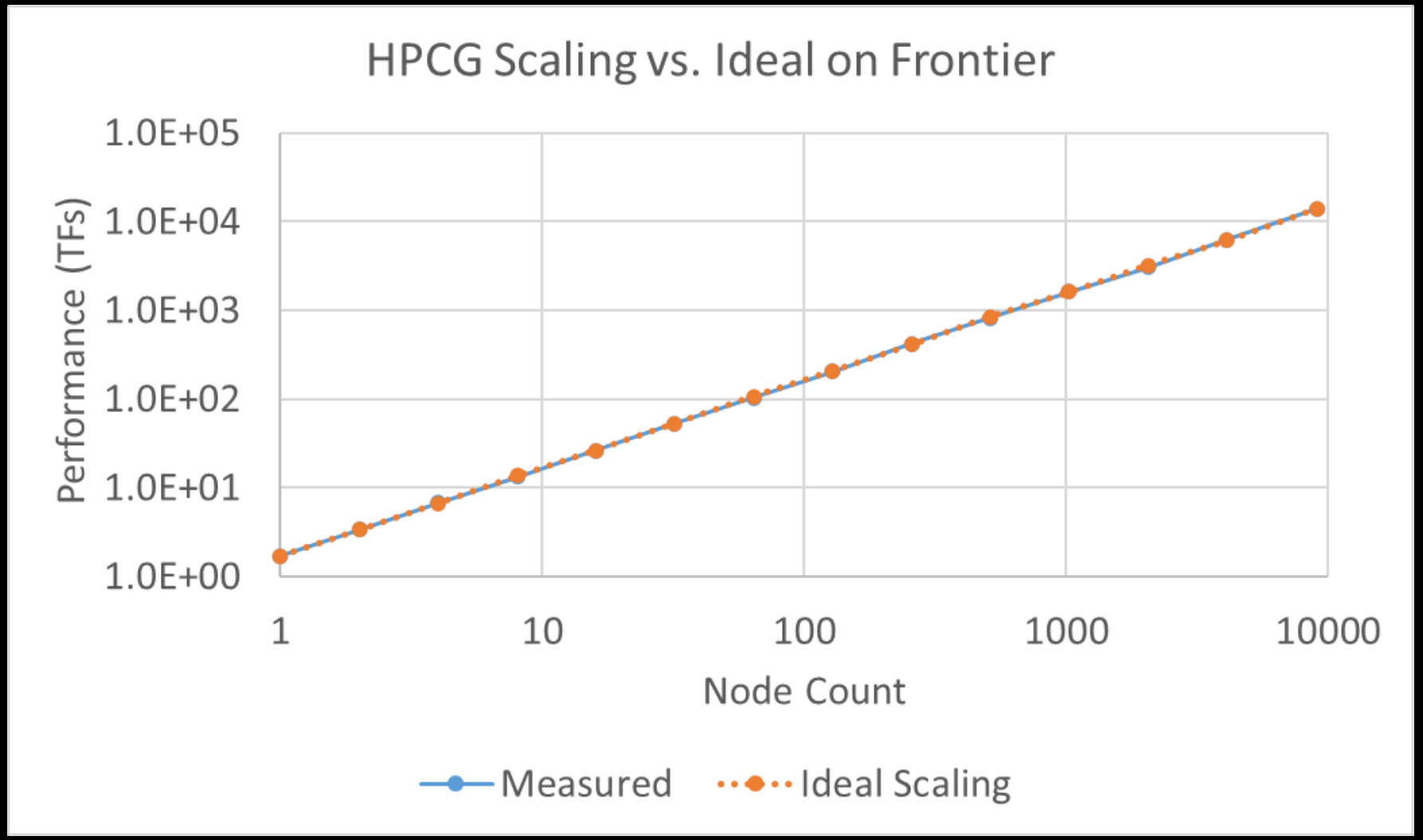
- High Performance Conjugate Gradient (HPCG)
  - Solves the 3d Poisson equation (heat diffusion) via iterative solve (conjugate gradient)
  - Sparse linear solver (SpMV), low arithmetic intensity
  - Broadly representative of many HPC codes
    - Memory bandwidth scaling (SpMV, Gauss-Seidel)
    - MPI collectives (all-reduce)
    - MPI sendrecvs (halo exchange)
  - Weak scaling benchmark



27-pt stencil

HPCG SCALING ON FRONTIER

- HPCG
  - Weak scaling benchmark
  - Nearly ideal scalability
  - From 1 -> 9000+ nodes
  - Implemented via MPI
- HPE Cray MPI
  - Slingshot interconnect
  - GPU-aware MPI
- Why does it scale well?
  - Low-latency
  - Low tail-latency



ML MOTIVATION

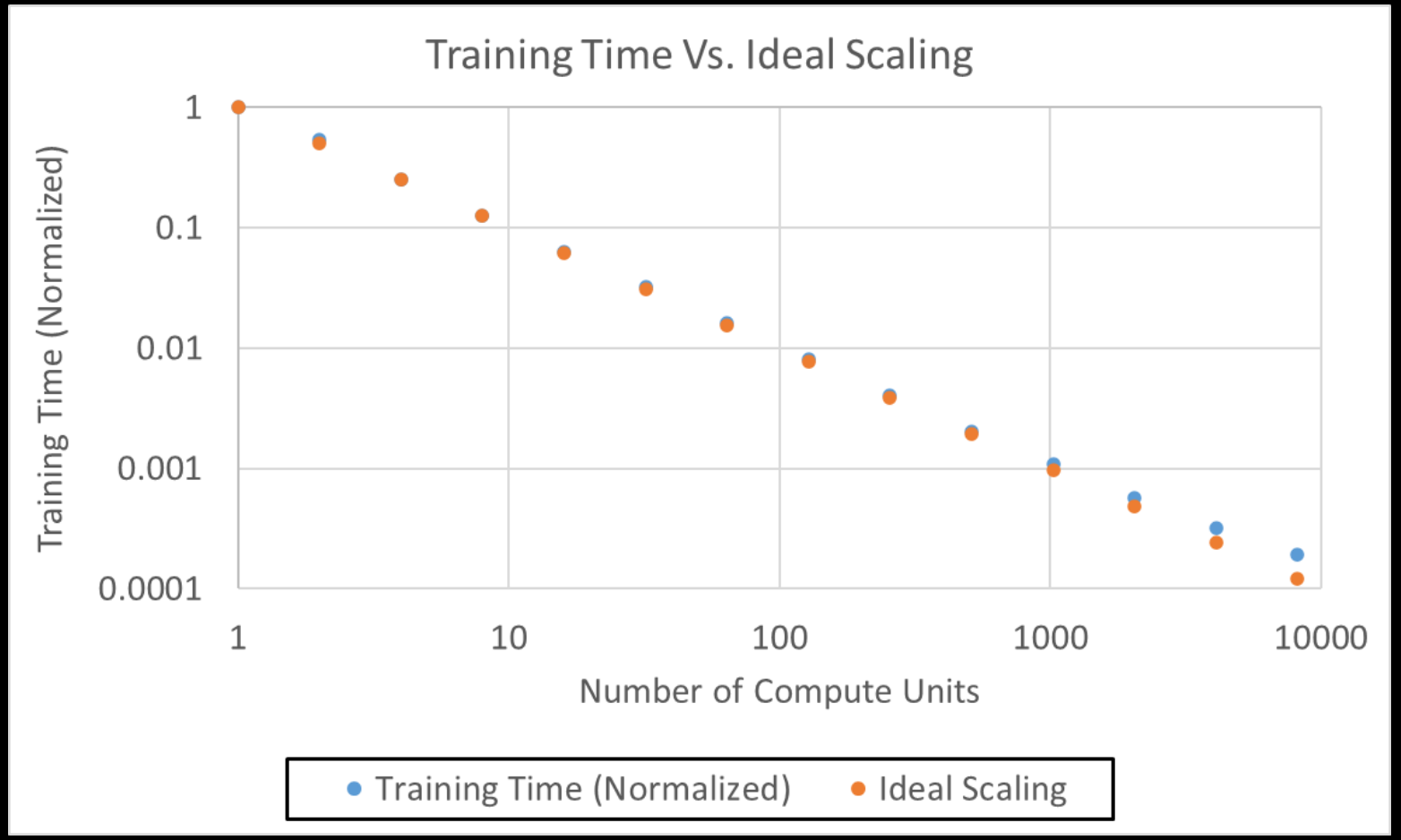
- LLMs (large language models) are useful in a wide range of tasks, in some cases matching human capabilities
- Deep learning models are also being explored for scientific discovery, such as cancer research
- Training LLMs is a computationally intensive task:
  - LaMDA (137 Billion parameters): trained in 51 days on 1024 TPUs
  - GPT-3 (175 Billion parameters): trained in 34 days on 1024 A100s
  - Number of parameters in the models is increasing at an exponential rate: Turing NLG was 17B in 2020

- 37,632 Frontier GPUs / 1024 = 36.75x the compute resources
  - Perfect strong scaling: 51 days of training would become less than two days
  - This would be a disruptive change in capabilities for rapidly training and deploying ML/AI models

ML SCALING ON FRONTIER

- Transformer Network
  - LLM, BERT-Large
  - Network Bandwidth hungry
  - Collective operations
- RCCL
  - ROCm Collective Communication Library
  - Portability: API equivalent to NVIDIA NCCL
  - Leverages HPE libfabric plugin

    NOT MPI standard compliant
- Nearly ideal scalability
  - Strong scaling (data + model parallelism)



Exascale systems are premier instruments for machine learning and AI research

CONCLUSIONS

- Exascale computing is a paradigm shift in *capability*, not in programming models
  - Traditional techniques of scientific computing (CUDA/HIP, OpenMP, MPI, etc.) are effective tools
  - "MPI + X" is the most viable internode programming model and is a success story, particularly, "GPU-Aware MPI + X"
- The largest barrier to effectively leveraging exascale systems today is *intranode programming*
  - *i.e.,* GPU-readiness
  - Leveraging coherent programming can reduce the 'time to acceleration'
  - Performance portability is not a fool's errand!
- Exascale computers are an excellent platform for machine learning development
  - Scale-out and strong scaling of state-of-the-art models could have significant impact
  - The methods and approaches of scientific computing have an important role to play in machine learning discovery

DISCLAIMER

AMD, the AMD Arrow logo, EPYC™, Instinct™ and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED ‘AS IS.” AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.


AMD
together we advance_