Dawn of the Exascale Computing Era

Douglas B. Kothe
Director, US Department of Energy Exascale Computing Project
Associate Laboratory Director, Computing and Computational Sciences
Oak Ridge National Laboratory

MultiCore WorldX
February 14, 2023
Wellington, NZ

Exascale Applications: potential outcomes and impact

Will be far-reaching for decades to come

- Predictive microstructural evolution of novel **chemicals and materials for energy** applications.
- Robust and selective **design of catalysts** an order of magnitude more efficient at temperatures hundreds of degrees lower.
- Accelerate the widespread adoption of additive manufacturing by enabling the **routine fabrication of qualifiable metal alloy parts**.
- Design **next-generation quantum materials** from first principles with predictive accuracy.
- Predict **properties of light nuclei** with less than 1% uncertainty from first principles.
- **Harden wind plant design** and layout against energy loss susceptibility, allowing higher penetration of wind energy.
- Demonstrate commercial-scale transformational energy technologies that **curb fossil fuel plant CO2 emission** by 2030.
- Accelerate the **design and commercialization of small and micronuclear reactors**.
- Provide the foundational underpinnings for a **'whole device' modelling capability for magnetically confined fusion plasmas** useful in the design and operation of ITER and future fusion reactors.

Exascale Applications: potential outcomes and impact

Will be far-reaching for decades to come

- Address **fundamental science questions** such as the **origin of elements** in the universe, the **behavior of matter at extreme densities**, the source of **gravity waves**; and demystify key unknowns in the dynamics of the universe (**dark matter, dark energy and inflation**).
- Reduce the current major uncertainties in **earthquake hazard and risk assessments** to ensure the safest and most cost-effective seismic designs.
- Reliably guide safe long-term **consequential decisions about carbon storage and sequestration**.
- Forecast, with confidence, **water resource availability, food supply changes and severe weather probabilities in our complex earth system environment**.
- **Optimize power grid planning and secure operation** with very high reliability within narrow operating voltage and frequency ranges.
- Develop treatment **strategies and pre-clinical cancer drug response models** and mechanisms for RAS/RAF-driven cancers.
- Discover, through **metagenomics analysis**, knowledge useful for environment remediation and the manufacture of novel chemicals and medicines.
- Dramatically **cut the cost and size of advanced particle accelerators** for various applications impacting our lives, from sterilizing food of toxic waste, implanting ions in semiconductors, developing new drugs or treating cancer.

DOE HPC Roadmap to Exascale Systems

FY 2012	FY 2016	FY 2018	FY 2021	FY 2022	FY 2023

decommissioned

ORNL
Cray/AMD/NVIDIA

ANL
IBM BG/Q

ANL
Cray/Intel KNL

LBNL
Cray/Intel Xeon/KNL

LLNL
IBM BG/Q

LANL/SNL
Cray/Intel Xeon/KNL

ORNL
IBM/NVIDIA

LLNL
IBM/NVIDIA

ORNL
HPE/AMD

Exascale Systems

ANL
HPE/AMD/NVIDIA

ANL
Intel/HPE

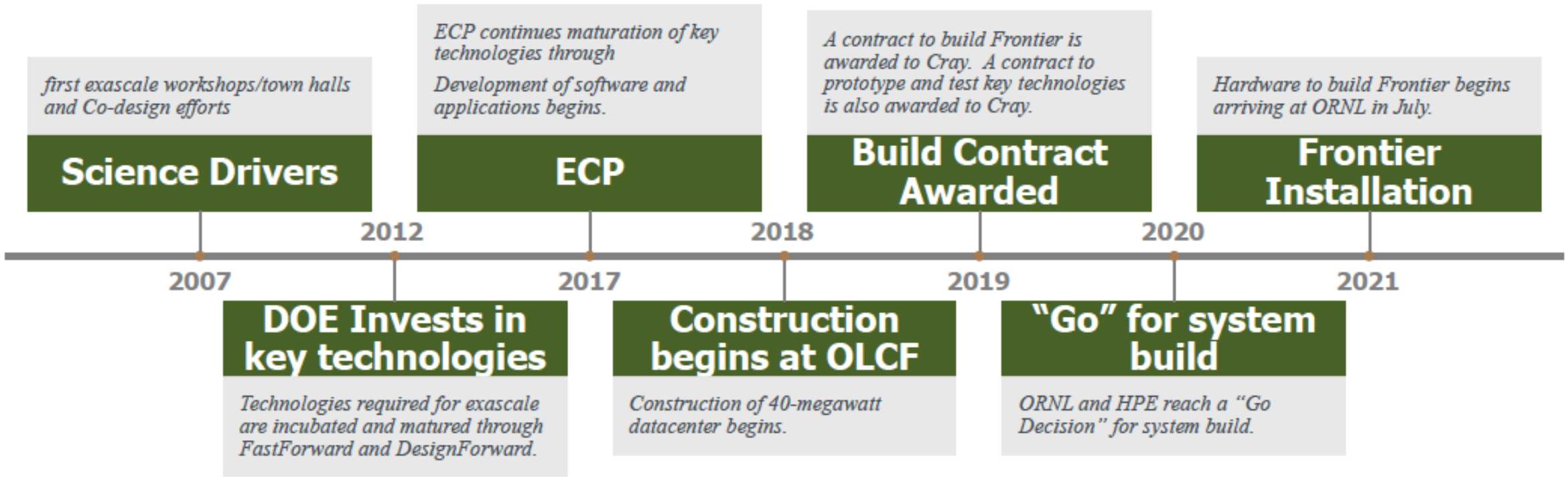
LBNL
HPE/AMD/NVIDIA

LANL/SNL
HPE/Intel

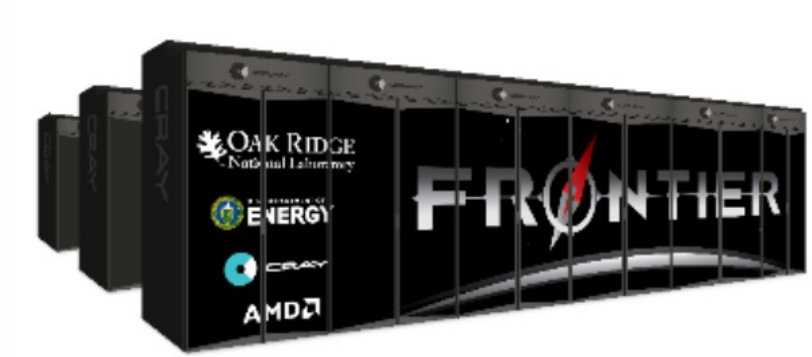
LLNL
HPE/AMD

Underlined: GPUs

Decadal effort to deliver U. S. Exascale systems led to Frontier

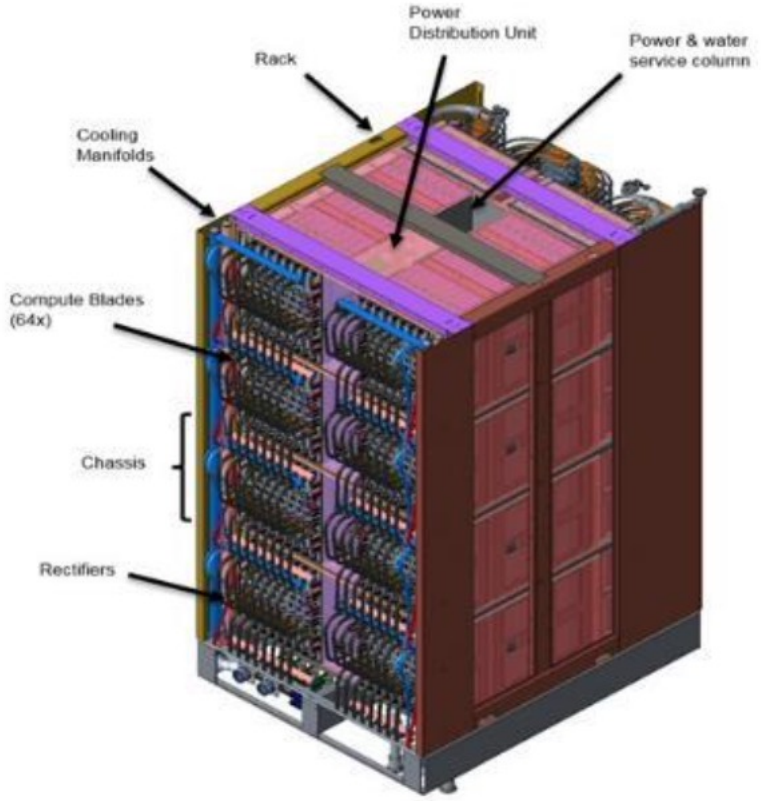


Frontier overview



System
- 2 EF peak DP Flops
- 74 compute racks
- 29 MW power consumption
- 9408 nodes
- 9.2 PB memory (4.6 PB HBM, 4.6 PB DDR4)
- Cray Slingshot network with dragonfly topology
- 37 PB node local storage
- 716 PB center-wide storage
- 4000 ft^2 footprint

Olympus rack
- 128 AMD nodes
- 8000 lbs
- Supports 400 kW



AMD node
- 1 AMD "Trento" CPU
- 4 AMD MI250X GPUs
- 512 GiB DDR4 memory on CPU
- 512 GiB HBM2e total per node (128 GiB HBM per GPU)
- Coherent memory across the node
- 4 TB NVM
- GPUs & CPU fully connected with AMD Infinity Fabric
- 4 Cassini NICs, 100 GB/s network BW

Compute blade
- 2 AMD nodes

All water cooled, even DIMMS and NICs

Frontier Node

All GPUs and CPU are fully connected on node and have coherent shared memory

Custom AMD EPYC CPU (64 core)

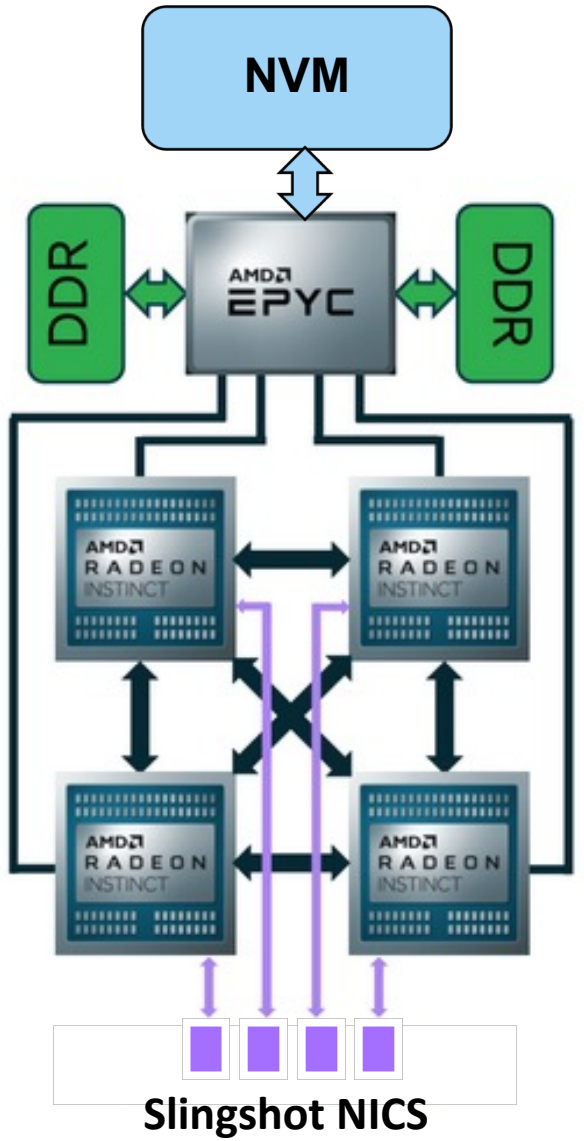
- Supports Infinity Fabric
- Adds PCIe links for on node NVM (4 TB)
- 512 GB of DDR4 memory (1/2 TB per node)

Four AMD MI250X GPUs

- Announced by AMD November 8 2021
- 128 GB of HBM2e each (1/2 TB per node)
- 3.2 TB/s memory bandwidth

Each GPU is connected to a Slingshot NIC

- Eliminates GPU-CPU link bottleneck seen in Titan and Summit
- 1 GPU or CPU can use all NICS together



Frontier multi-tier storage system is designed to excel at Data Science and AI for Scientific Discovery

	Capacity	Performance	
Multi-tier I/O Subsystem		**Read**	**Write**
37 PB Node Local Storage		65.9 TB/s	62.1 TB/s
		11 Billion IOPS	
11 PB Performance tier		9.4 TB/s	9.4 TB/s
695 PB Capacity tier		5.2 TB/s	4.4 TB/s
10 PB Metadata		2M Transactions per sec	

Two 2TB SSD NVM per node Local Storage (Flash)

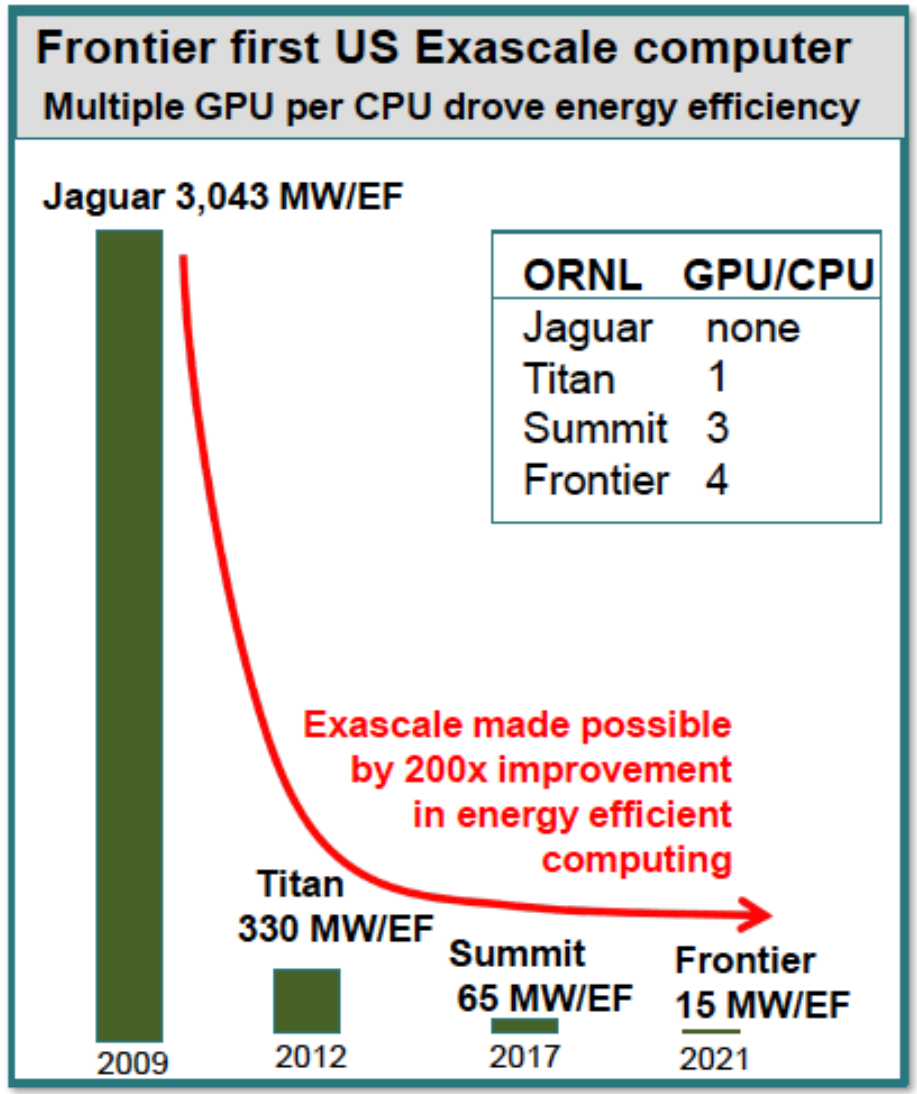
Gazelle SSD Storage board (Performance Tier and Metadata)

Moose HDD Storage board (Capacity Tier)

Energy Efficient Computing – Frontier achieves 14.5 MW per EF

Since 2009 the biggest concern with reaching Exascale has been energy consumption

- **ORNL pioneered GPU use in supercomputing** beginning in 2012 with Titan thru today with Frontier. Significant part of energy efficiency improvements.
- **ASCR [Fast, Design, Path] Forward vendor investments** in energy efficiency (2012-2020) further reduced the power consumption of computing chips (CPUs and GPUs)..
- **200x reduction in energy per FLOPS** from Jaguar to Frontier at ORNL
- ORNL achieves additional energy savings from using warm water cooling in Frontier (32 C).
  ORNL Data Center PUE= 1.03



The Exascale Computing Project (ECP) enables US revolutions in technology development; scientific discovery; healthcare; energy, economic, and national security

ECP Mission

Develop exascale-ready applications and solutions that address currently intractable problems of strategic importance and national interest.

Create and deploy an expanded and vertically integrated software stack on DOE HPC exascale and pre-exascale systems, defining the enduring US exascale ecosystem.

Deliver **US HPC vendor technology advances and deploy ECP products** to DOE HPC pre-exascale and exascale systems.

ECP Vision

Deliver **exascale simulation and data science innovations and solutions to national problems** that enhance US economic competitiveness, change our quality of life, and strengthen our national security.

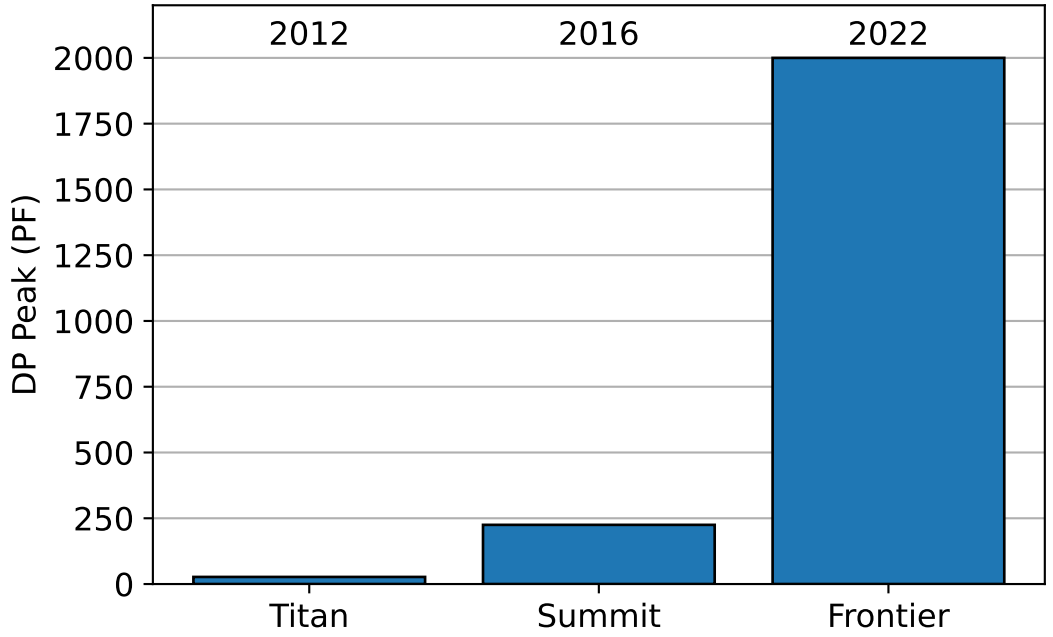
- Funded by DOE Office of Science, Advanced Scientific Computing Research (ASCR) and DOE National Nuclear Security Administration (NNSA)
- 7-year project – $1.8B
- 6 lead labs: ORNL, ANL, LBNL, LLNL, SNL, LANL
- More than 80 research teams
  - >1000 researchers
  - Drawn heavily from 17 DOE labs plus national universities and US companies (100+ each)

Each HPC system has served a vital role for ECP Teams

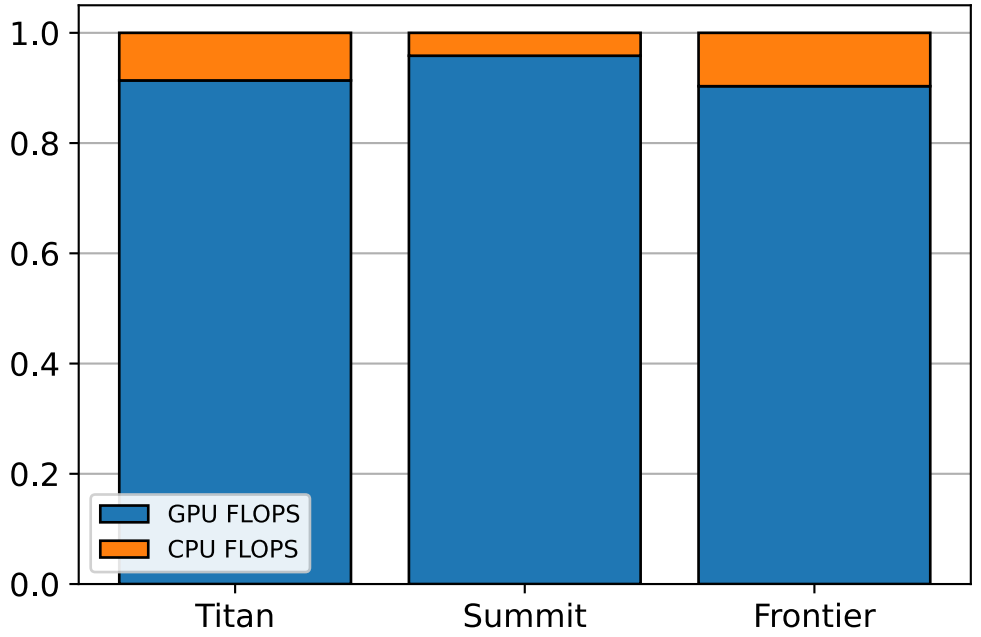
From benchmarking to development to now demonstration of key performance parameters (KPPs)

	Benchmark system for many ECP AD and ST teams	Multi-GPU system for scaling, algorithm & model dev, S/W design	Target system for KPP threshold demonstrations
System	**Titan (2012) Cray**	**Summit (2017) IBM**	**Frontier (2021) HPE**
Peak	27 PF	200 PF	> 1.5 EF
# nodes	18,688	4,608	9,408
Node	1 AMD Opteron CPU 1 NVIDIA Kepler GPU	2 IBM POWER9™ CPUs 6 NVIDIA Volta GPUs	1 AMD EPYC CPU 4 AMD Radeon Instinct GPUs
Memory		2.4 PB DDR4 + 0.4 HBM + 7.4 PB On-node storage	4.6 PB DDR4 + 4.6 PB HBM2e + 37 PB On-node storage, 66 TB/s Read 62 TB/s Write
On-node interconnect	PCI Gen2 No coherence across the node	NVIDIA NVLINK Coherent memory across the node	AMD Infinity Fabric Coherent memory across the node
System Interconnect	Cray Gemini network 6.4 GB/s	Mellanox Dual-port EDR IB 25 GB/s	Four-port Slingshot network 100 GB/s
Topology	3D Torus	Non-blocking Fat Tree	Dragonfly
Storage	32 PB, 1 TB/s, Lustre Filesystem	250 PB, 2.5 TB/s, IBM Spectrum Scale™ with GPFS™	695 PB HDD+11 PB Flash Performance Tier, 9.4 TB/s and 10 PB Metadata Flash
Power	9 MW	13 MW	29 MW

Performance on current and next-gen HPC architectures requires effective use of GPUs



Peak performance



FLOPS by device

ECP Application Portfolio: 24 First-Movers of Strategic Importance to DOE

Starting Point

- **24 applications** and **6 co-design** projects
  - Including **78 separate codes**
  - Representing over **10 million lines of code**
  - Many supporting large user communities
  - Covering broad range of mission critical S&E domains
  - Mostly all MPI or MPI+OpenMP on CPUs
- Each envisioned innovative S&E enabled by 100X increase in computing power
- Path to harnessing 100-fold improvement initially unknown likely to have disruptive impact on software unlike anything in last 30 years

Current status

- All applications have, with their own unique development plans, made tremendous progress in model and algorithm development and software architecture redesign / refactor. Most applications have integrated and adopted software abstraction layers or co-designed motif-based components and frameworks to ensure efficient and portable GPU implementations.
- Many application have already realized >50X increase in science work rate performance on the Summit system at ORNL since starting ECP development activities in 2016

→ **Massive software investments**

ECP Applications

Targeting specific challenge problems that emanate from key DOE program stakeholder strategies

Domain*	Base Challenge Problem	Risks and Challenges
Wind Energy	2x2 5 MW turbine array in 3x3x1 km^3 domain	Linear solvers; structured / unstructured overset meshes
Nuclear Energy	Small Modular Reactor with complete in-vessel coolant loop	Coupled CFD + Monte Carlo neutronics; MC on GPUs
Fossil Energy	Burn fossil fuels cleanly with CLRs	AMR + EB + DEM + multiphase incompressible CFD
Combustion	Reactivity controlled compression ignition	AMR + EB + CFD + LES/DNS + reactive chemistry
Accelerator Design	TeV-class 10^{2-3} times cheaper & smaller	AMR on Maxwell's equations + FFT linear solvers + PIC
Magnetic Fusion	Coupled gyrokinetics for ITER in H-mode	Coupled continuum delta-F + stochastic full-F gyrokinetics
Nuclear Physics: QCD	Use correct light quark masses for first principles light nuclei properties	Critical slowing down; strong scaling performance of MG-preconditioned Krylov solvers
Chemistry: GAMESS	Heterogeneous catalysis: MSN reactions	HF + MP2 + coupled cluster (CC) + fragmentation methods
Chemistry: NWChemEx	Catalytic conversion of biomass	CCSD(T) + energy gradients
Extreme Materials	Microstructure evolution in nuclear matls	AMD via replica dynamics; OTF quantum-based potentials
Additive Manufacturing	Born-qualified 3D printed metal alloys	Coupled micro + meso + continuum; linear solvers

*Required to demonstrate a capability *and* performance metric
*Required to demonstrate a capability metric

ECP Applications

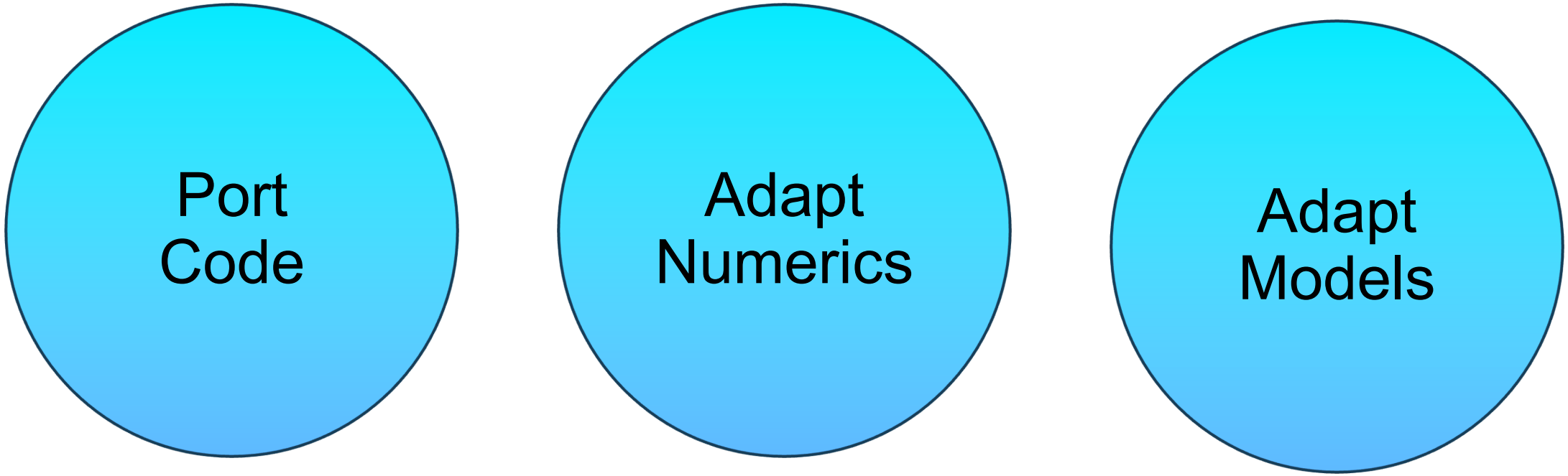
Targeting specific challenge problems that emanate from key DOE program stakeholder strategies

Domain*	Challenge Problem	Computational Hurdles
Quantum Materials	Predict & control matls @ quantum level	Parallel on-node perf of Markov-chain Monte Carlo; OpenMP
Astrophysics	Supernovae explosions, neutron star mergers	AMR + nucleosynthesis + GR + neutrino transport
Cosmology	Extract “dark sector” physics from upcoming cosmological surveys	AMR or particles (PIC & SPH); subgrid model accuracy; in-situ data analytics
Earthquakes	Regional hazard and risk assessment	Seismic wave propagation coupled to structural mechanics
Geoscience	Well-scale fracture propagation in wellbore cement due to attack of CO_2-saturated fluid	Coupled AMR flow + transport + reactions to Lagrangian mechanics and fracture
Earth System	Assess regional impacts of climate change on the water cycle @ 5 SYPD	Viability of Multiscale Modeling Framework (MMF) approach for cloud-resolving model; GPU port of radiation and ocean
Power Grid	Large-scale planning under uncertainty; underfrequency response	Parallel nonlinear optimization based on discrete algebraic equations; multi-period optimization
Cancer Research	Scalable machine learning for predictive preclinical models and targeted therapy	Increasing accelerator utilization for model search; exploiting reduced/mixed precision; resolving data management or communication bottlenecks
Metagenomics	Discover and characterize microbial communities through genomic and proteomic analysis	Graph algorithms, distributed hashing, matrix operations and other discrete algorithms
FEL Light Source	Protein and molecular structure determination using streaming light source data	Parallel structure determination for ray tracing and single-particle imaging

*Required to demonstrate a capability *and* performance metric

*Required to demonstrate a capability metric

Efficiently utilizing GPUs goes far beyond typical code porting

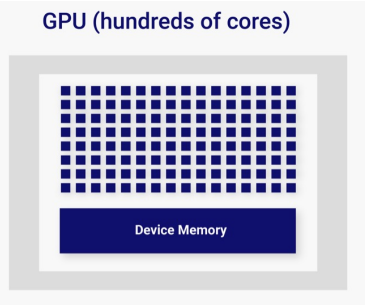


- Rewrite, profile, and optimize
- Memory coalescing
- Loop ordering
- Kernel flattening

- Reduced synchronization
- Reduced precision
- Communication avoiding

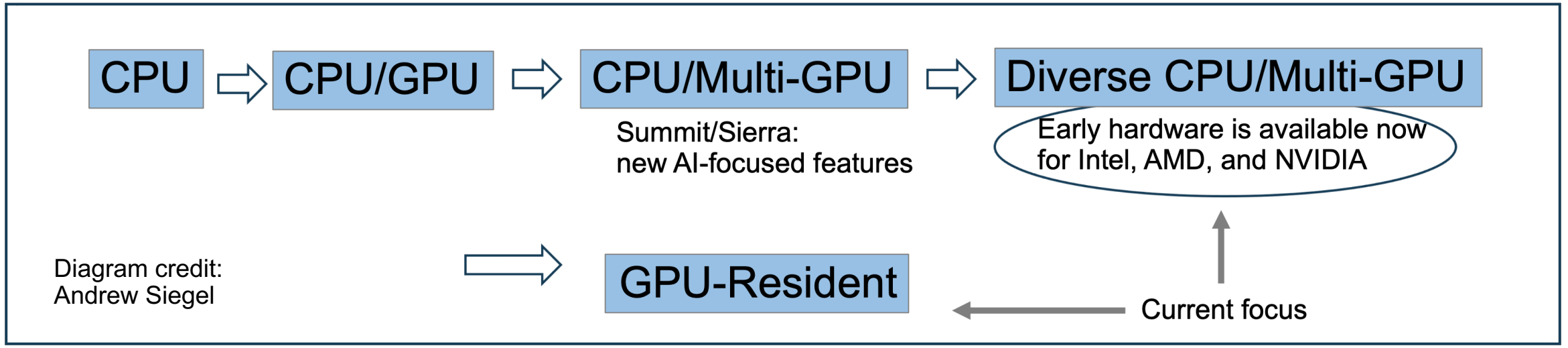
- Mathematical representation
- “On the fly” recomputing vs. lookup tables
- Prioritization of new physical models

Heterogeneous accelerated-node computing



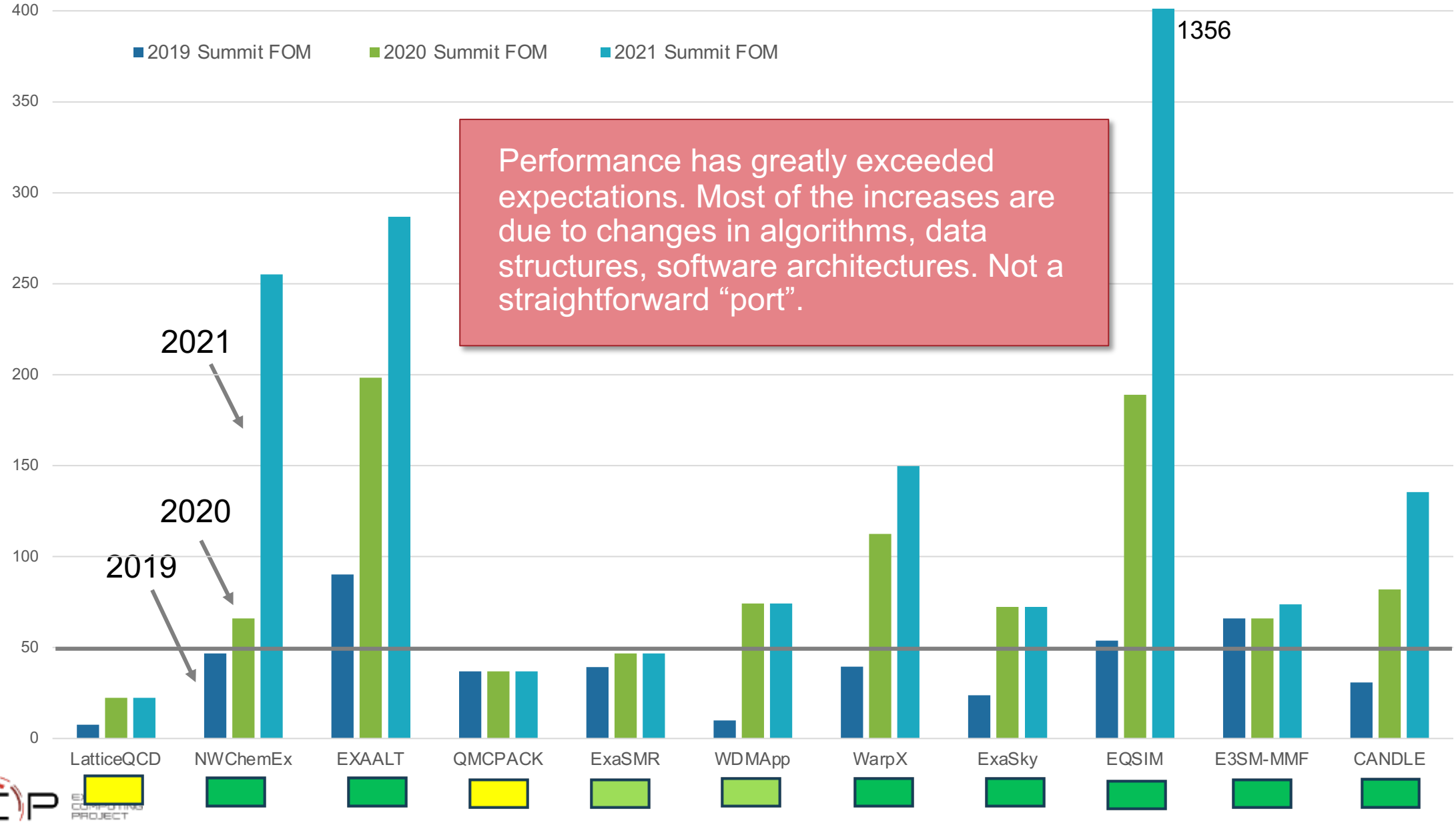
Accelerated node computing: Designing, implementing, delivering, & deploying agile software that effectively exploits heterogeneous node hardware

- Execute on the largest systems … AND on today and tomorrow's laptops, desktops, clusters, …
- We view *accelerators* as any compute hardware specifically designed to accelerate certain mathematical operations (typically with floating point numbers) that are typical outcomes of popular and commonly used algorithms. We often use the term GPUs synonymously with accelerators.

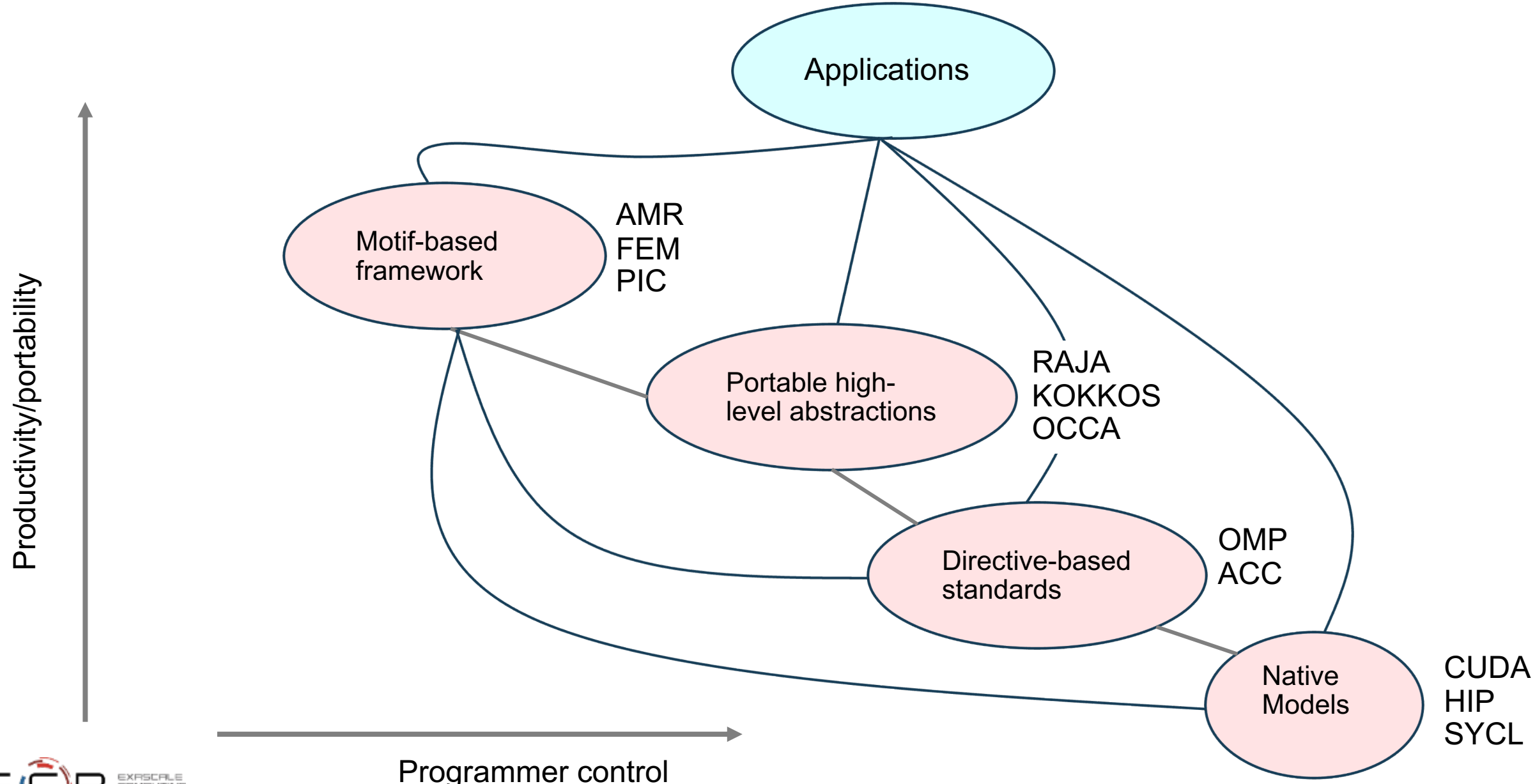


Ref: A Gentle Introduction to GPU Programming, Michele Rosso and Andrew Myers, May 2021

Summit Performance for Selected ECP KPP-1 Applications



Programming model choice balances risk/control with productivity



Application Motifs* (what's the app footprint?)

Algorithmic methods that capture a common pattern of computation and communication

1. **Dense Linear Algebra**
   - Dense matrices or vectors (e.g., BLAS Level 1/2/3)
2. **Sparse Linear Algebra**
   - Many zeros, usually stored in compressed matrices to access nonzero values (e.g., Krylov solvers)
3. **Spectral Methods**
   - Frequency domain, combining multiply-add with specific patterns of data permutation with all-to-all for some stages (e.g., 3D FFT)
4. **N-Body Methods (Particles)**
   - Interaction between many discrete points, with variations being particle-particle or hierarchical particle methods (e.g., PIC, SPH, PME)
5. **Structured Grids**
   - Regular grid with points on a grid conceptually updated together with high spatial locality (e.g., FDM-based PDE solvers)
6. **Unstructured Grids**
   - Irregular grid with data locations determined by app and connectivity to neighboring points provided (e.g., FEM-based PDE solvers)
7. **Monte Carlo**
   - Calculations depend upon statistical results of repeated random trials
8. **Combinational Logic**
   - Simple operations on large amounts of data, often exploiting bit-level parallelism (e.g., Cyclic Redundancy Codes or RSA encryption)
9. **Graph Traversal**
   - Traversing objects and examining their characteristics, e.g., for searches, often with indirect table lookups and little computation
10. **Graphical Models**
    - Graphs representing random variables as nodes and dependencies as edges (e.g., Bayesian networks, Hidden Markov Models)
11. **Finite State Machines**
    - Interconnected set of states (e.g., for parsing); often decomposed into multiple simultaneously active state machines that can act in parallel
12. **Dynamic Programming**
    - Computes solutions by solving simpler overlapping subproblems, e.g., for optimization solutions derived from optimal subproblem results
13. **Backtrack and Branch-and-Bound**
    - Solving search and global optimization problems for intractably large spaces where regions of the search space with no interesting solutions are ruled out. Use the divide and conquer principle: subdivide the search space into smaller subregions ("branching"), and bounds are found on solutions contained in each subregion under consideration

**The Landscape of Parallel Computing Research: A View from Berkeley, Technical Report No. UCB/EECS-2006-183 (Dec 2006).*

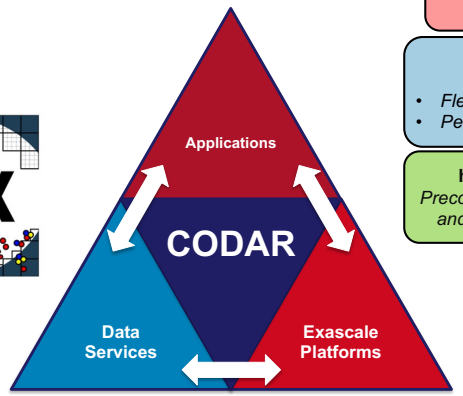
ECP Co-Design Centers for key computational motifs

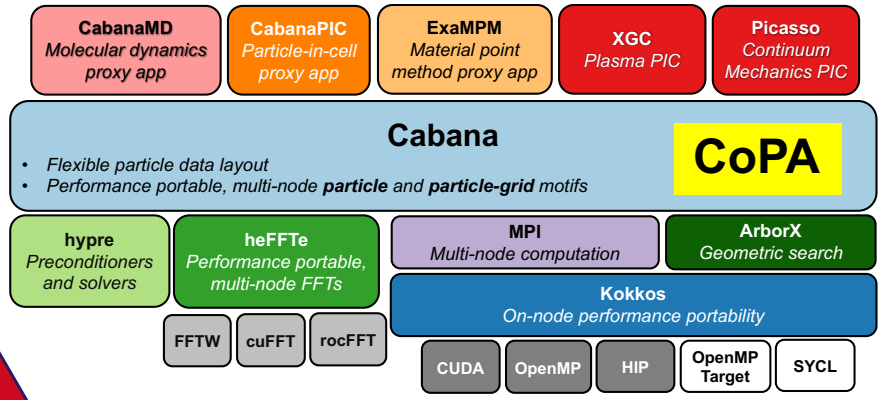
Project	PI Name, Inst	Short Description/Objective
CODAR	Ian Foster, ANL	Understand the constraints, mappings, and configuration choices between applications, data analysis and reduction, and exascale platforms
AMReX	John Bell, LBNL	Build framework to support development of block-structured adaptive mesh refinement algorithms for solving systems of partial differential equations on exascale architectures
CEED	Tzanio Kolev, LLNL	Develop next-generation discretization software and algorithms that will enable finite element applications to run efficiently on future hardware
CoPA	Susan Mniszewski, LANL	Create co-designed numerical recipes and performance-portable libraries for particle-based methods
ExaGraph	Mahantesh Halappanavar, PNNL	Develop methods and techniques for efficient implementation of key combinatorial (graph) algorithms
ExaLearn	Frank Alexander, BNL	Deliver state-of-the-art machine learning and deep learning software at the intersection of applications, learning methods, and exascale platforms

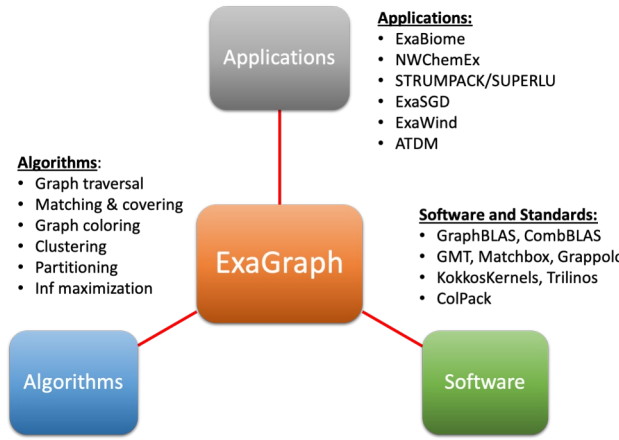
CEED
EXASCALE DISCRETIZATIONS

AMReX

ECP EXASCALE COMPUTING PROJECT

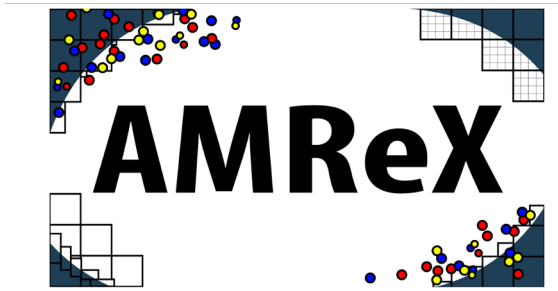




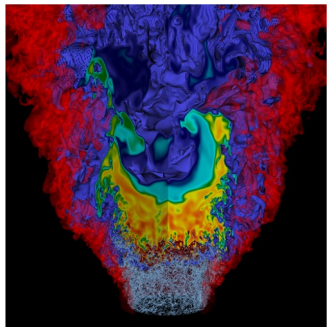


AMReX provides portability to ECP applications through multiple low-level implementations

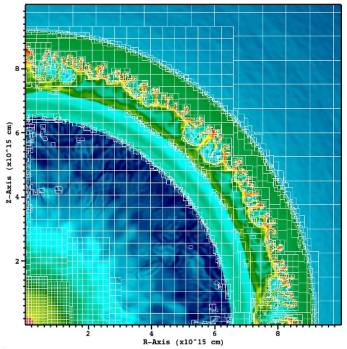
Principal motif: structured mesh, patch-based adaptive mesh refinement



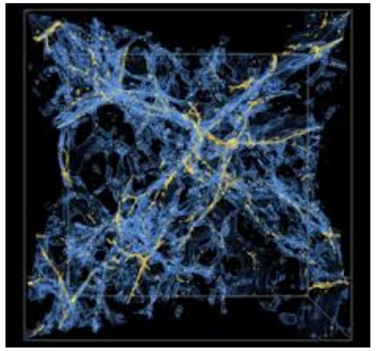
Combustion-PELE (PeleC and PeleLM)



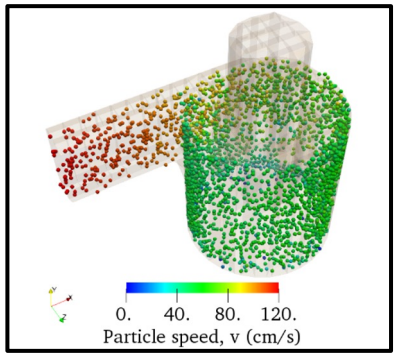
ExaStar (Castro)



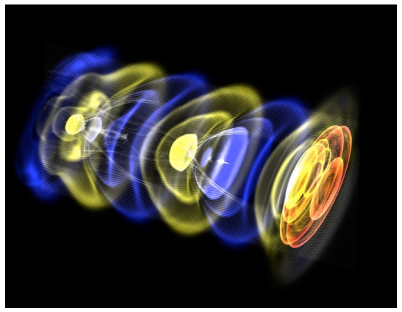
ExaSky (Nyx)



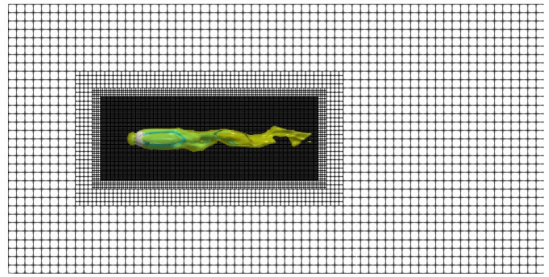
MFIX-Exa



WarpX



ExaWind (AMR-Wind)



AMReX

MPI | **OpenMP** | **OpenACC**

CUDA | **HIP** | **DPC++**

https://amrex-codes.github.io/

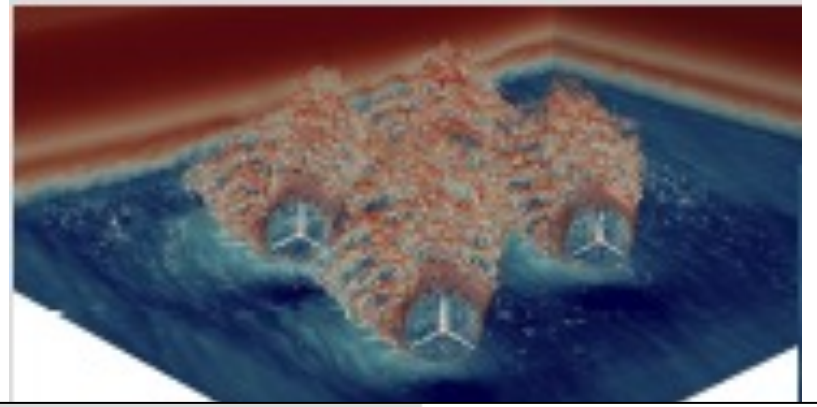
Then (2016) and Now (2023): AMReX

Adaptive Refinement of Patch-based Structured Meshes

AMReX Then		AMReX Now
Mix of C++11 (data structures, high-level control flow) and Fortran (low-level numerical operations)	**Source code**	Source code: pure C++17 with extensive use of template metaprogramming
MPI + OpenMP only	**Hybrid Parallelism**	MPI + X , where X is one of OpenMP (CPUs) or CUDA, HIP or SYCL (NVIDIA, AMD, or Intel GPUs)
Support for redistribution and particle-mesh, array-of-structs only	**Particles**	Both array-of-struct and struct-of-array data, halo exchange + neighbor lists for particle-particle collisions
None	**Complex Geometry**	Support for embedded boundaries via cut-cell approach
Native multi-level geometric multigrid	**Linear Solvers**	Same + EB-aware options, interfaces to hypre and PETSc
GNUMake only	**Installation**	CMake + GNUMake for compilation from source One step installation with Spack
Native plotfiles	**IO**	Native plotfiles + HDF5, support for compression with SZ and ZFP, Asynchronous IO
VisIt, yt, Paraview	**Visualization**	Same + support for in-situ analysis and visualization with ALPINE, SENSEI
Manual runs of test suite Limited documentation Informal code reviews for critical changes	**Development policies/practices**	Extensive test coverage with continuous integration Extensive online documentation and tutorials Formal code reviews for all changes
Applications could run at full-scale on Edison, Cori KNL	**Performance**	**AMReX applications can run efficiently at full-scale on Perlmutter, Fugaku, Summit, and Frontier.**

Then (2016) and Now (2023): ExaWind

Predictive physics-based simulation of wind plants



Then (2016)

Approach: Create computational fluid/structure dynamics (CFD and CSD) codes for Reynolds-averaged Navier-Stokes (RANS)/large-eddy simulations (LES) where wind turbine geometry and blade boundary layers are resolved and include moving meshes, fluid-structure interaction, and atmospheric turbulence

Starting-Point Codes:

Nalu: https://github.com/nalucfd/

- Unstructured-grid, incompressible-flow CFD
- LES turbulence model
- C/C++
- Built on Trilinos STK, Tpetra/Belos/MueLu solvers, and Kokkos
- Mesh rotation achieved through a sliding-mesh interface

OpenFAST: https://github.com/openfast/

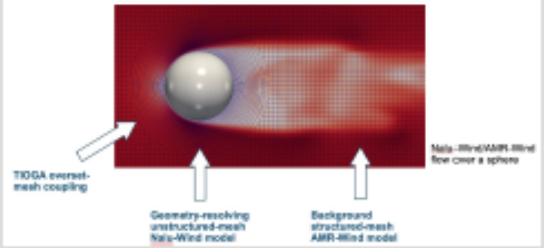
- Whole-turbine simulation code (structural dynamics, control)
- Fortran90

Challenges:

- Target problem requires resolving spatial scales going from blade boundary layers (e.g., 10^{-5} m) to the wind farm domain (e.g., 10^{3} m), i.e., at least eight orders of magnitude
- Finite volumes with extreme aspect ratios (e.g., 10,000), which are necessary for hybrid-RANS/LES, were a serious challenge linear-system solvers
- Time-integration scheme required impractically small time-step sizes (e.g., 10^{-6} s) for production simulations
- Sliding-mesh approach presented mesh-creation challenges and no clear pathway for yaw motions

Now (2023)

Shift in Approach: Added AMR-Wind as a background solver and made Nalu-Wind the near-body solver; coupling via overset meshes



Proof-of-concept simulation of flow over a sphere using the hybrid Nalu-Wind/AMR-Wind solver.

Primary Application Codes:

Nalu-Wind

- https://github.com/exawind/nalu-wind
- Wind-specific offshoot from Nalu; primarily used for near-body flows
- *hypre* is primary linear-system-solver package
- Hybrid-RANS/LES with time integrator that enables practical time step sizes
- Overset meshes (via TIOGA, https://github.com/jsitaraman/tioga) is primary method for moving meshes
- Performant on NVIDIA GPUs; Advanced Micro Devices, Inc. (AMD) GPUs are in progress

AMR-Wind

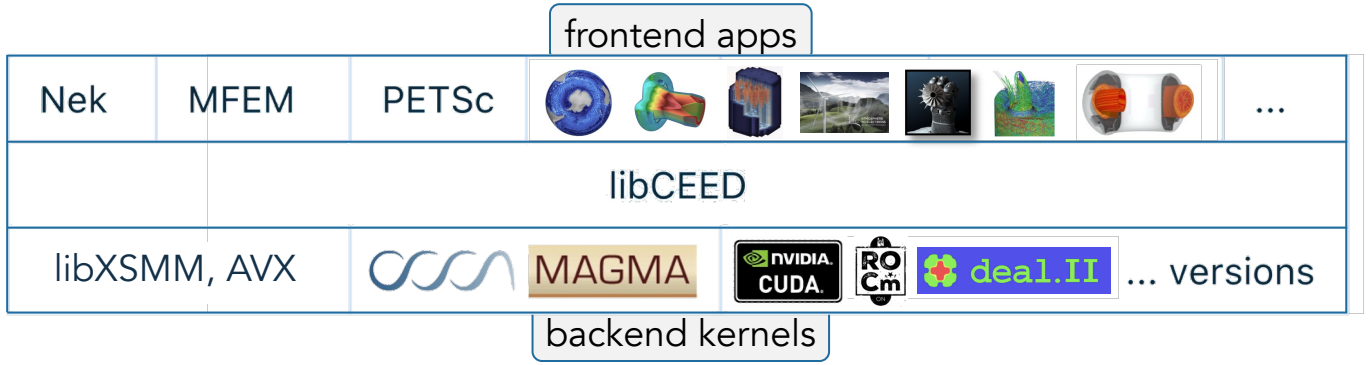
- https://github.com/exawind/amr-wind
- Structured-grid adaptive mesh refinement (AMR) CFD code; background solver
- C++ and built on the AMReX library
- Performant on NVIDIA and AMD GPUs

OpenFAST

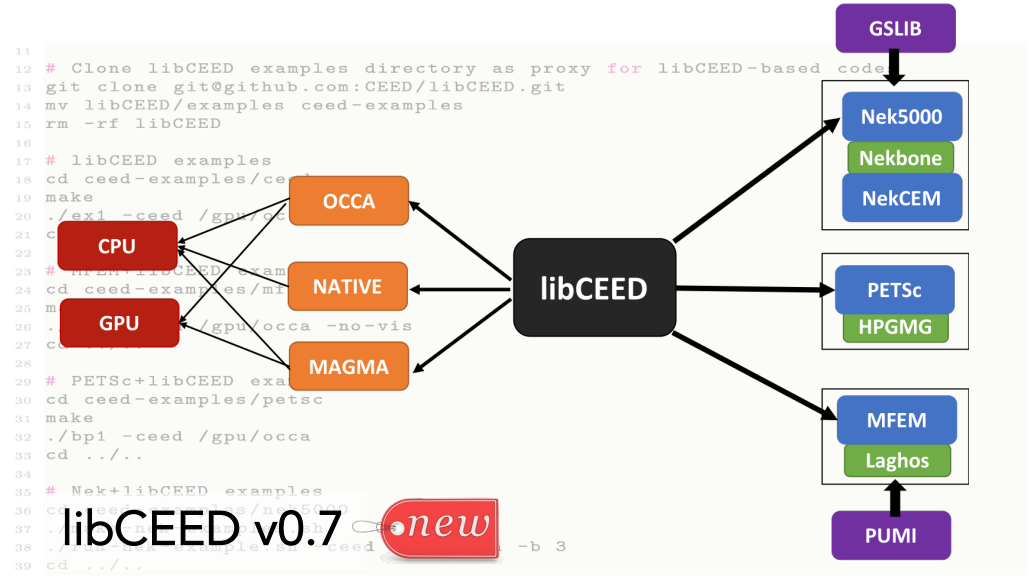
- No pathway to support parallelization or GPUs
- Starting new FY23 WETO project to create replacement: OpenTurbine https://github.com/exawind/openturbine

CEED provides multiple back-ends, including through its OCCA portability layer

Principal motif: unstructured mesh finite element discretization



- ✓ API between *frontend apps* and *backend kernels*
- ✓ *Efficient operator description* (not global matrix)
- ✓ Clients use any backend as a run-time option
- ✓ Backend can be added as plugins without recompiling
- ✓ Backends compete for best performance, latency vs throughput, optimize for order/device, use JIT, …

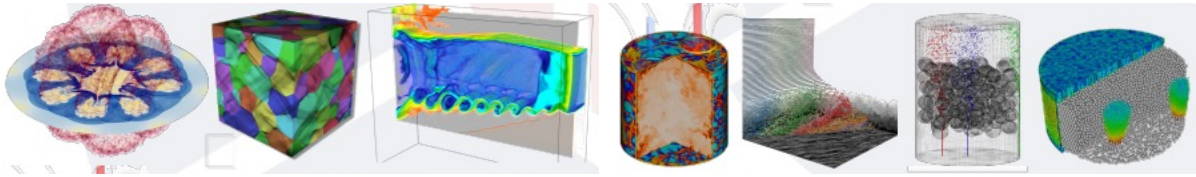


- ✓ Extensible backends
- **CPU**: reference, vectorized, libXSMM
- **CUDA** using NVRTC cuda-gen
- **OCCA** (JIT): CPU, OpenMP, OpenCL, CUDA
- **MAGMA**

https://ceed.exascaleproject.org/

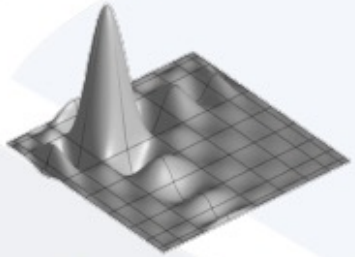
Then (2016) and Now (2023): CEED

Center for Efficient Exascale Discretizations

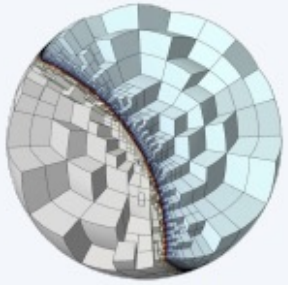


Then

- **PDE-based simulations on unstructured grids**
- **High-order and spectral finite elements**
  - ✓ *any order space on any order mesh*
  - ✓ *curved meshes,*
  - ✓ *unstructured AMR*
  - ✓ *matrix-free methods*
  - ✓ *optimized low-order support*



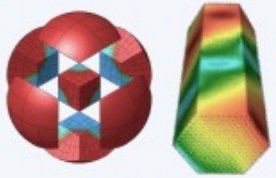
10th order basis function

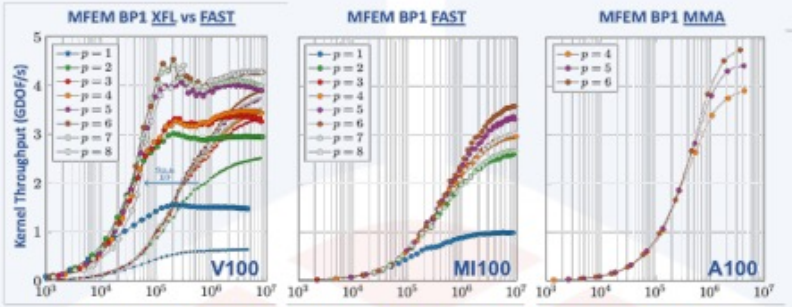


non-conforming AMR, 2nd order mesh

Now *new*

- **CEED discretization libraries**
  - ✓ **High-Level API:** *Nek & MFEM projects*
  - ✓ Nek5000/NekRS: nek5000.mcs.anl.gov
  - ✓ MFEM: mfem.org

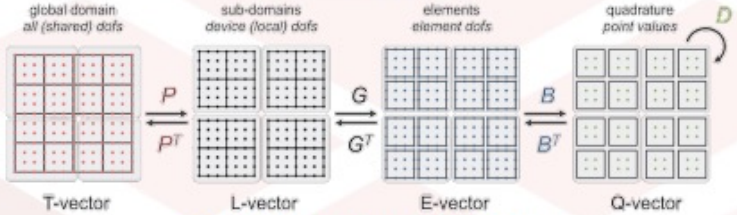




New MFEM GPU kernels: (1) have better strong scaling, (2) perform on NVIDIA + AMD GPUs, and (3) can utilize tensor cores

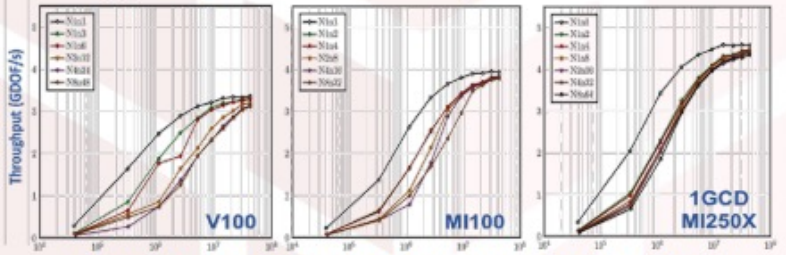
- **libCEED** github.com/CEED/libceed
  - ✓ **Low-Level API:** *new library for efficient operator evaluation*
  - ✓ *state-of-the-art CPU and GPU kernel performance*

$$A = P^T G^T B^T D B G P$$



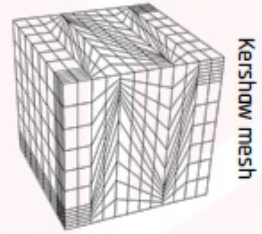
Finite element operator decomposition

- **Miniapps:** Laghos, libParanumal, hipBone



hipBone performance for order 7: 1 MI250X GCD = 1.2 MI100 = 1.3 V100

- **Benchmarks**
  - ✓ *bake-off problems:* **BP1-BP6**
  - ✓ *solver BPs:* **BPS3, BPS5**
  - ✓ *high-order community benchmarks*



- **High-order software ecosystem**
  - ✓ *high-order meshing, optimization*
  - ✓ *high-order visualization*
  - ✓ *performance portability, GPUs*
  - ✓ *scalable "matrix-free" solvers*

RAJA CEED MAGMA PETSc hypre VTK-m Ascent PUMI

- **More information and downloads**
  - ✓ *CEED project website:* ceed.exascaleproject.org
  - ✓ *CEED code repositories:* github.com/CEED

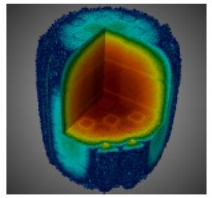
Then (2016) and Now (2023): ExaSMR

Resolved coupled neutronics+thermal hydraulics phenomena in nuclear reactor cores

MC Neutronics Then

- Minimal GPU support
- Fixed material temperatures
- Single statepoint (limited isotopic depletion)
- Performance: 10^7 particles/second

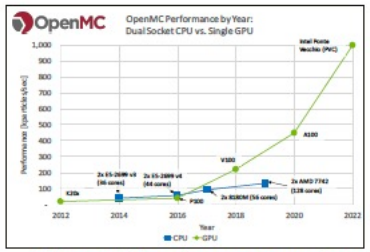
SMR core geometry



Total reaction rate in SMR core

MC Neutronics Now

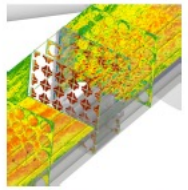
- Support for Nvidia, AMD, and Intel GPUs using HIP and OpenMP target offload
- On-the-fly Doppler broadening
- Integrated isotopic depletion capability
- Performance: >10^9 particles/second



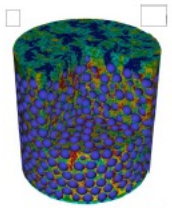
CPU vs. GPU performance over time

CFD Then

- Nek5000: CPU only (experimental OpenACC support)
- Single fuel assembly simulations
- Max problem size: 30 million elements, 10 billion DOF
- Performance: 3x10^9 DOF/second



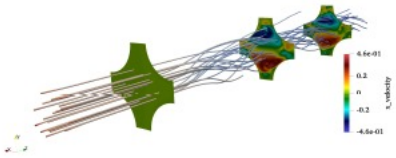
Coolant flow through mixing vane



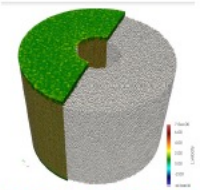
Simulation of flow through 1000 pebbles

CFD Now

- NekRS: Efficient execution on Nvidia, AMD, and Intel GPUs using OCCA
- Full SMR core with effect of heat exchanger
- Improved solver/preconditioning capabilities
- State-of-the-art mixing vane modeling
- Max problem size: 1 billion elements, 350 billion DOF
- Performance: ~5x10^{11} DOF/second



Fluid streamlines downstream of mixing vane



NekRS simulation of FHR pebble bed reactor (350k pebbles)

CoPA: Cabana particle library is built on a Kokkos portability layer

Principal motif: particles

CabanaMD
Molecular dynamics proxy app

CabanaPIC
Particle-in-cell proxy app

ExaMPM
Material point method proxy app

XGC
Plasma PIC

Picasso
Continuum Mechanics PIC

Cabana

- *Flexible particle data layout*
- *Performance portable, multi-node **particle** and **particle-grid** motifs*

hypre
Preconditioners and solvers

heFFTe
Performance portable, multi-node FFTs

FFTW | **cuFFT** | **rocFFT**

MPI
Multi-node computation

ArborX
Geometric search

Kokkos
On-node performance portability

CUDA | **OpenMP** | **HIP** | **OpenMP Target** | **SYCL**

Then (2016) and Now (2023): CoPA

Addressing the challenges for particle-based applications to run on exascale architectures

Cabana: A Co-Designed HPC Library for Particle Applications

https://github.com/ECP-CoPA/Cabana

Lead: Sam Reeve (ORNL), Co-lead: Stuart Slattery (ORNL)

*Developers: Christoph Junghans (LANL), Damien Lebrun-Grandie (ORNL), Austin Isner (ORNL), Kwitae Chong (ORNL), Shane Fogerty (LANL), Aaron Scheinberg (PPPL-consultant), Guangye Chen (LANL), **Yuxing Qiu (UCLA)**, **Yu Fang (UCLA)**, **Stephan Schulz (Jülich)**, Jim Glosli (LLNL), Evan Weinberg (NVIDIA)*

Collaborators: Stan Moore (SNL), Lee Ricketson (LLNL), Steve Rangel (ANL), Adrian Pope (ANL), Mark Stock (HPE)

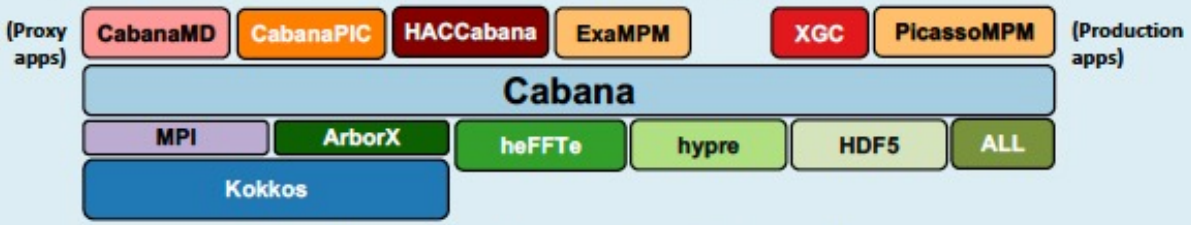
How we started

- Each particle application defined and implemented separate particle data structures, algorithms, and communication, even with some significant overlap between domains: **Cabana did not exist.**
- Each partner application had different strategies for the coming exascale and performance portability (direct vendor backends for **HACC** and Kokkos for **LAMMPS**), but some strategies were unsustainable (multiple sets of conflicting and complex dependencies for **XGC**). Finally, the **PicassoMPM** application did not exist.

Where we are now

Cabana is a full-featured particle library as an extension of Kokkos

- Particle data structures, particle algorithms, and multi-node particle communication
- Structured grids, grid algorithms, multi-node grid communication, and particle-grid interpolation
- Particle algorithms, load balancing, and I/O through optional third-party libraries



Tier-1 application partner integrations

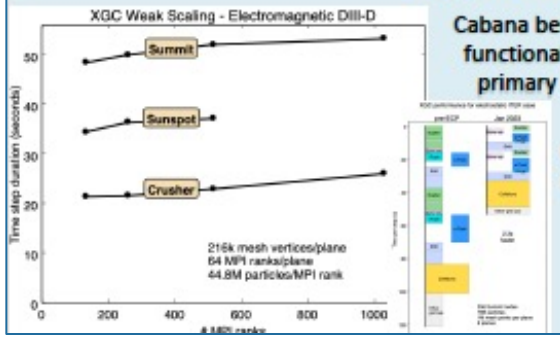
Cabana provides benefits across many use cases, exemplified by our app partners:

- **XGC:** Direct use of Cabana for migration to performance portability with plans for further algorithm adoption
- **PicassoMPM:** Full use of Cabana for development of a brand new particle-grid application
- **HACC:** Proxy app for rapid exploration of new algorithms and designs alongside production codes (**HACCabana**)
- **LAMMPS:** Comparison and sharing of algorithms and Kokkos performance strategies

Additional impact

- PIC algorithm development using Cabana for rapid prototyping (**CabanaPIC**)
- Sharing of algorithm and performance strategies with the **AMReX** adaptive mesh refinement library
- New non-ECP applications: **CabanaPD** (ORNL LDRD peridynamics), **Hyperion** (LANL LDRD multi-physics hybrid PIC), **MRMD** (Max Planck multi-resolution MD), **PUMI-PIC** (RPI PIC), **Beatnik** (UNM PSAAP Z-model)

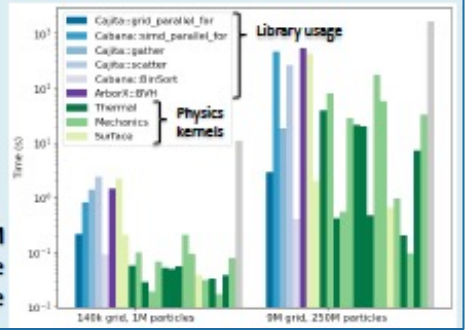
Application performance



Cabana benchmarks across all functionality and runs on all primary Kokkos backends

Left: **XGC** weak scaling & speedup

Right: **PicassoMPM** Crusher single node GPU performance



Then (2016) and Now (2023): ExaAM

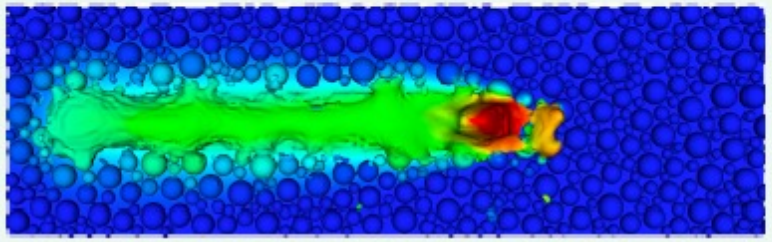
Simulated additive manufacturing at the fidelity of the microstructure

Before ExaAM…

- Overall workflow simply didn't exist.
- Some components didn't exist.
- Very few could have run on GPU's.
- Part-scale melt pool simulation intractable

Pre-ExaAM Computational Components

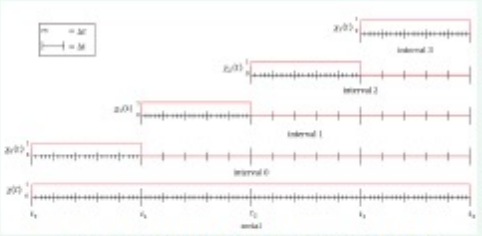
Exascale Challenge Problem Role	Code	Physics	Major limitations
Part-scale Thermo-mechanics	Diablo (LLNL)	Solid mechanics, Heat & mass transport, Contact	Under-resolved process model, no process-aware material properties
Melt pool physics	Truchas (LANL) additiveFOAM (ORNL)	Heat transfer, Phase change, Fluid flow, Species diffusion	Implicit-only integration. No time parallel capability. Limited heat source options.
Powder-resolved melt pool physics	ALE3D (LLNL)	Free-surface flow, Heat transfer, Phase change	Not accessible to community (license, export control). Simulations took weeks per millimeter of laser track. CPU only.
Grain-scale microstructure	ExaCA (ORNL/LLNL)	Cellular automata	Not fully developed, serial CPU only.
Post-solidification microstructure	MEUMAPPS-SS (ORNL)	Phase-field, Solid-solid phase transformation	CPU only
Micromechanical properties	Abaqus user material	Polycrystal plasticity	Not scalable, restrictive licensing, CPU only
Sub-grain scale microstructure	AMPE (LLNL/ORNL)	Phase-field, solidification	Needed additional physics, CPU only
	Tusas (LANL)	Phase-field, solidification	Needed additional physics



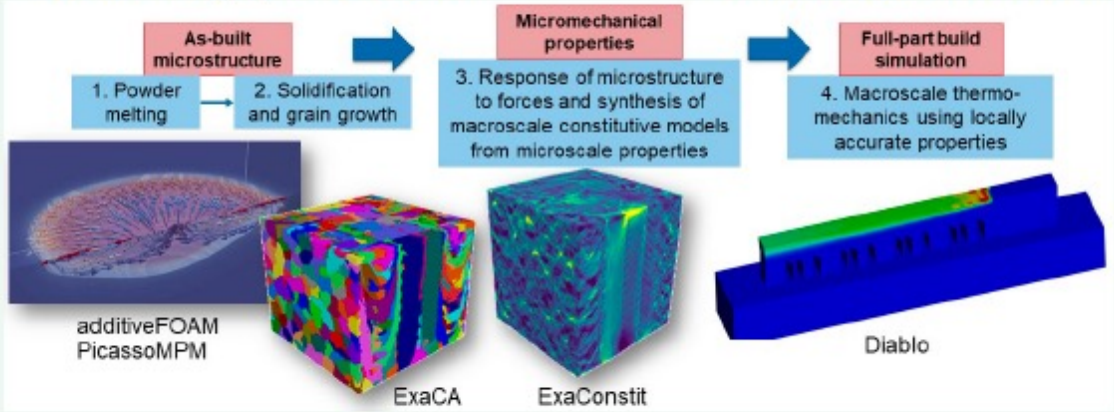
- ALE3D – high fidelity, powder resolved melt pool
- No microstructure
- >100k CPU weeks/mm

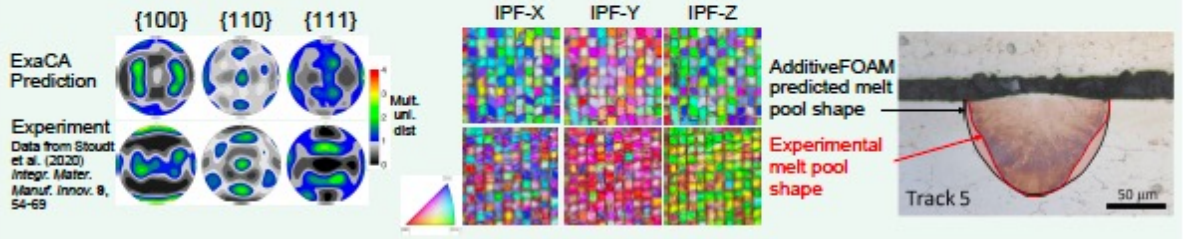
With ExaAM…

- Integrated workflow
- GPU: ExaCA, ExaConstit, PicassoMPM
- Parallel-in-time capability
- Experimental validation
- Uncertainty quantification



Parallel-in-time melt pool simulation technique



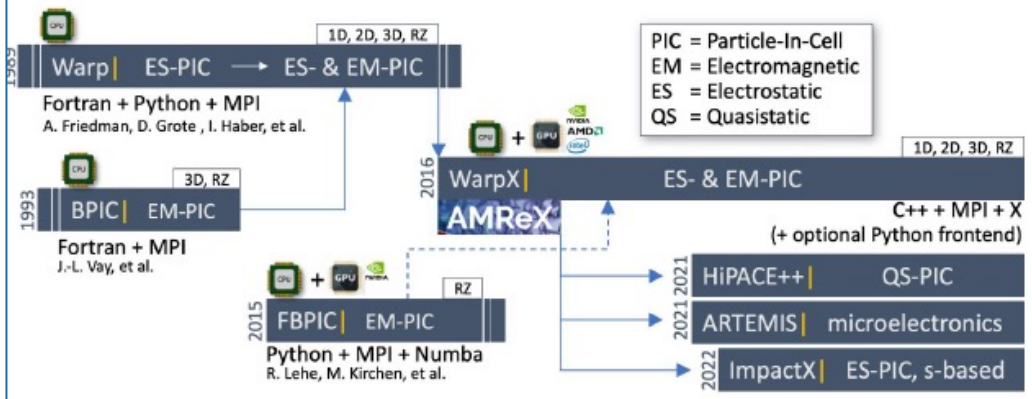


Challenge Problem Experimental Comparison

Then (2016) and Now (2023): WarpX

Modeling of charged particle beams and accelerators, lab & astro plasmas, fusion devices

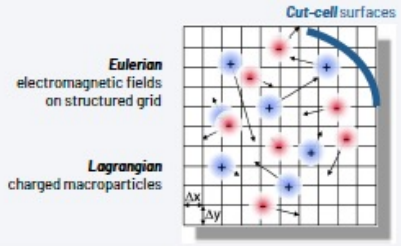
WarpX & Spinoffs History & Roadmap



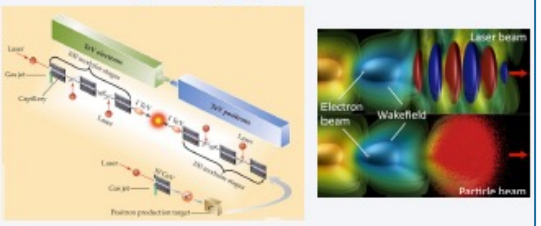
Overview of Warp/WarpX

Warp and WarpX are multiphysics codes/frameworks for the modeling of charged particle beams and accelerators, lab & astro plasmas, fusion devices & more.

Codes are constructed around the Particle-In-Cell (PIC) algorithm:



Challenge ECP problem: the modeling of chains of plasma-based particle accelerators for future high-energy physics colliders



Warp as of 2016		WarpX as of 2023
large set of advanced, novel algorithms	**Algorithms**	Warp advanced algos + new algorithms introduced during ECP
50% Fortran + 50% Python (including programmable frontend) had grown to large >1M lines of codes w/ varying programming styles	**Source code**	Source code: C++17 & optional Python programmable frontend compact thanks to C++ templating
CPUs, MPI-parallel	**Supported hardware**	CPUs, 3 flavors of GPUs, MPI-parallel
limited	**Performance optimization**	extensive
limited support, independently	**Load balancing & AMR**	combined native support
compilation from source some support for binaries	**Installation**	standard (CMake) compilation from source one step with Spack/Conda/PyPI, multi-platform
small team (2+) of computational physicists + individual contributions over several decades	**Development team**	tightly integrated team of computational physicists + applied mathematicians + computer scientists + software engineers
manual runs of test suite partial online documentation, outdated in part informal code reviews for critical changes	**Development policies/practices**	extensive test coverage with continuous integration extensive online documentation formal code reviews for all changes
could perform 3-D modeling of single plasma accelerator stage at moderate resolution	**ECP science case**	**can perform 3-D modeling of chain of tens of plasma accelerator stages at twice the resolution in each direction**

WarpX's "Then and Now" is compelling . . . as it is for every team

Each ECP team's articulation of this reality will help with adoption, sustainability, evolution

Warp (as of 2016)	WarpX (as of 2022)
Runs on CPUs	Runs on CPUs & 3 vendors of GPUs
~ 50% Fortran + 50% Python	100% C++ + optional Python frontend
Many advanced algorithms & physics	More & better algorithms & physics
Good scaling to ~6000 CPU nodes	Good scaling to ~150000 CPU nodes, 8000 GPU nodes
No dynamic load balancing	Efficient load balancing
"Home-made", brittle Mesh refinement capability	Mesh refinement based on robust AMReX library
Scaling of I/Os was a bottleneck	Good scaling of I/Os with ADIOS/HDF5
Installation required compilation	Easy installation with Spack, Conda, …
Manual tests ensured correctness	~200 physics benchmarks run automatically on every code commit
Modeling of one plasma accelerator stage at moderate resolution	Modeling of 10+ plasma accelerator stages at high resolution

Figure-of-Merit over time

Date	Code	Machine	N_c/Node	Nodes	FOM
3/19	Warp	Cori	0.4e7	6 625	2.2e10
3/19	WarpX	Cori	0.4e7	6 625	1.0e11
6/19	WarpX	Summit	2.8e7	1 000	7.8e11
9/19	WarpX	Summit	2.3e7	2 560	6.8e11
1/20	WarpX	Summit	2.3e7	2 560	1.0e12
2/20	WarpX	Summit	2.5e7	4 263	1.2e12
6/20	WarpX	Summit	2.0e7	4 263	1.4e12
7/20	WarpX	Summit	2.0e8	4 263	2.5e12
3/21	WarpX	Summit	2.0e8	4 263	2.9e12
6/21	WarpX	Summit	2.0e8	4 263	2.7e12
7/21	WarpX	Perlmutter	2.7e8	960	1.1e12
12/21	WarpX	Summit	2.0e8	4 263	3.3e12
4/22	WarpX	Perlmutter	4.0e8	928	1.0e12
4/22	WarpX	Perlmutter†	4.0e8	928	1.4e12
4/22	WarpX	Summit	2.0e8	4 263	3.4e12
4/22	WarpX	Fugaku†	3.1e6	98 304	8.1e12
6/22	WarpX	Perlmutter	4.4e8	1 088	1.0e12
7/22	WarpX	Fugaku	3.1e6	98 304	2.2e12
7/22	WarpX	Fugaku†	3.1e6	152 064	9.3e12
7/22	WarpX	Frontier	8.1e8	8 576	1.1e13

500x

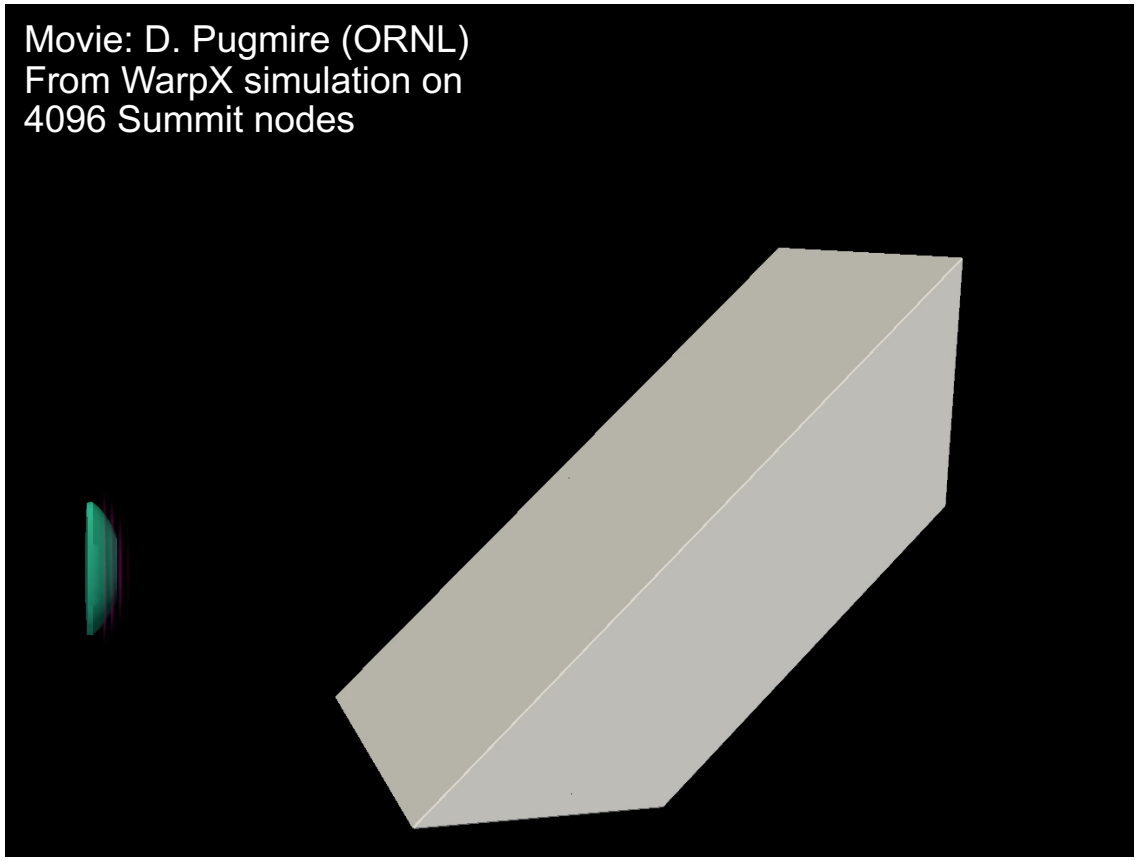
Computational power increase:

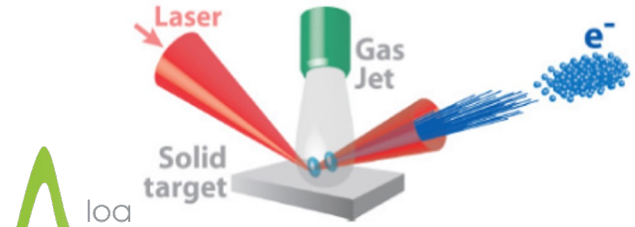
- 500x: Warp (2016) ➔ WarpX (2022)

WarpX team: Gordon Bell Award Winner at SC22!!

April-July 2022: WarpX on **world's largest HPCs**

L. Fedeli, A. Huebl et al., SC'22, 2022





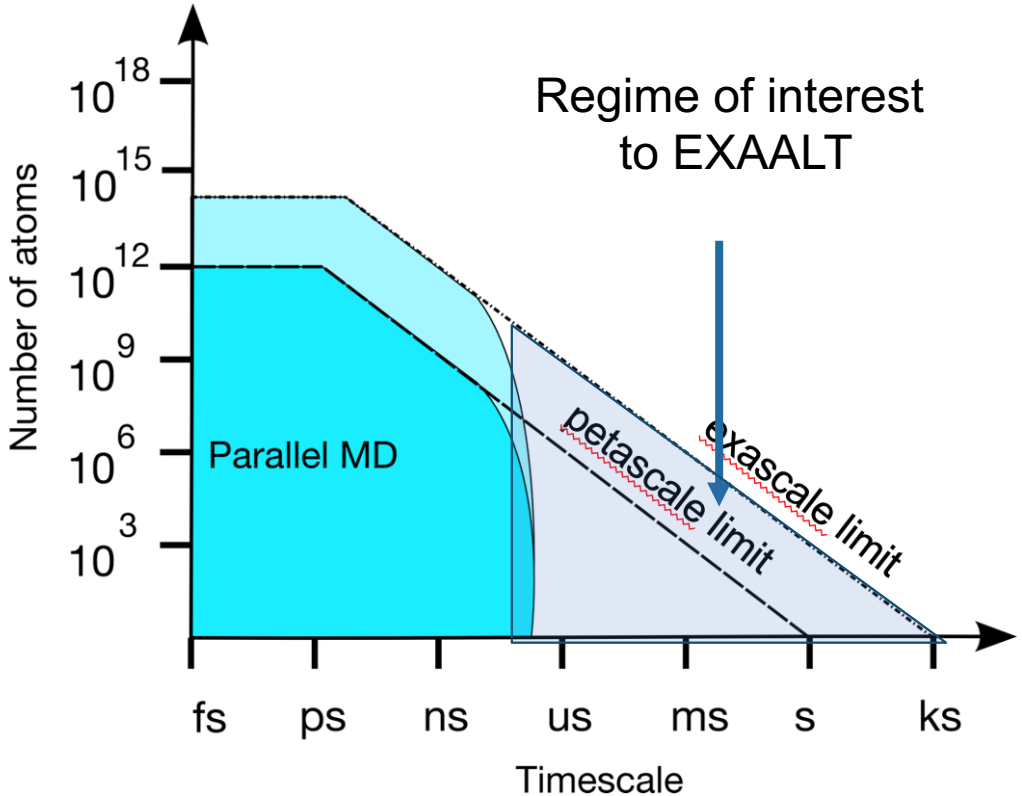
Novel hybrid solid-gas target concept

Our simulations demonstrated that the new concept leads to unprecedented beam quality using a PW-class laserI, and are supporting experiments at LOA (Ecole Polytechnique, France) to validate the new concept.

Success story of a multidisciplinary, international multi-institutional team!

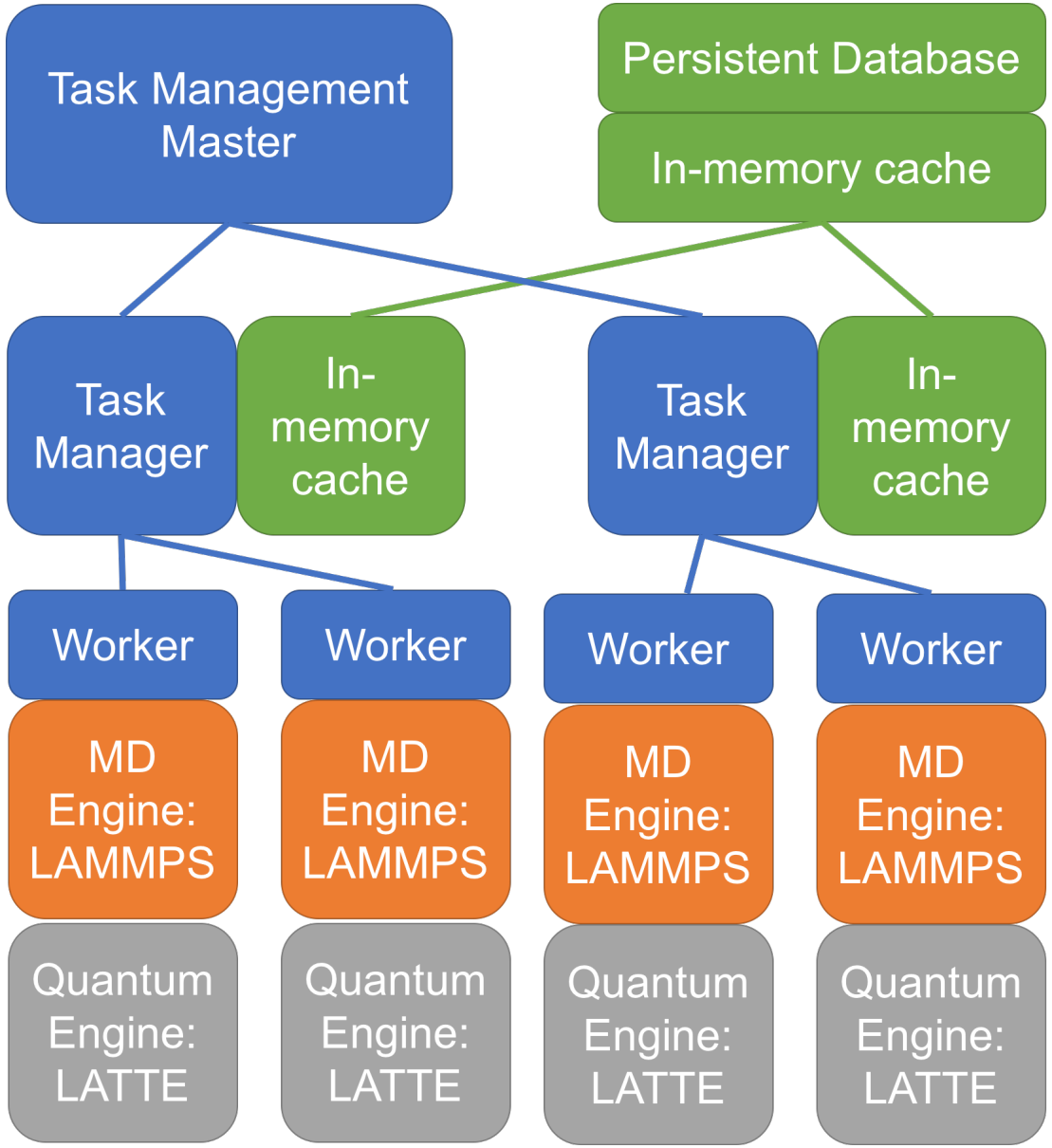
ECP's EXAALT Application: Methods

- Long times accessed with Accelerated MD methods (Voter *et al.*)
  - Parallel Trajectory Splicing (Perez *et al.*)
  - TAMMBER (Swinburne *et al.*)
- Parallelizes in the *time domain* using replica-based techniques
- Dynamically accurate to arbitrary precision (Lelievre *et al.*)
- Intermediate size/time regime through combined domain+replica decomposition (Synchronous sub-lattice, Amar *et al.*)



ECP's EXAALT Application: Computational Capability

- AMD methods implemented through custom-made task and data management system
- Fully asynchronous execution: no blocking/all-to-all communications
- Can be used to implement a variety of complex workflows:
  - Kinetic model construction
  - Machine-learning potentials

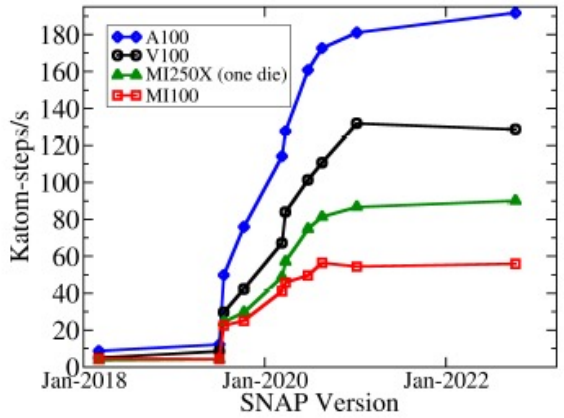


Then (2016) and Now (2023): EXAALT

Integrated MD simulation environment to access as much Accuracy/Length/Time simulation space as possible

The Evolution of EXAALT: Then and Now

Key Kernel Performance



The computational performance of EXAALT is dominated by the calculation of atomic energies and force.

A Kokkos implementation was available pre-ECP. Totally rewritten using different loop structure, memory access patterns, etc.

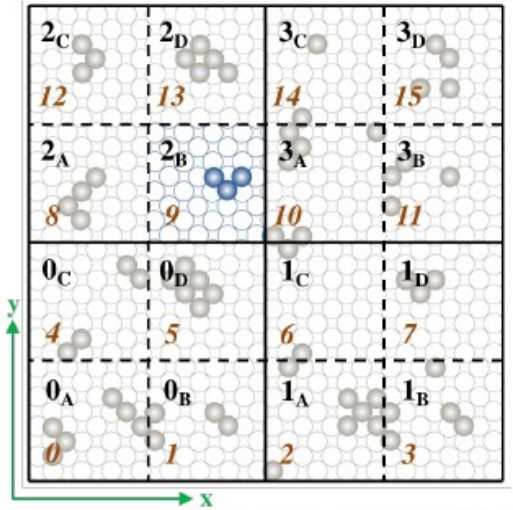
25x performance improvement over baseline implementation coming from **code improvements alone.**

Projected FOM at scale on Frontier:
756x speedup vs full Mira

Long-timescale methodologies

The baseline code implemented the original Parallel Trajectory Splicing (ParSplice) algorithm.

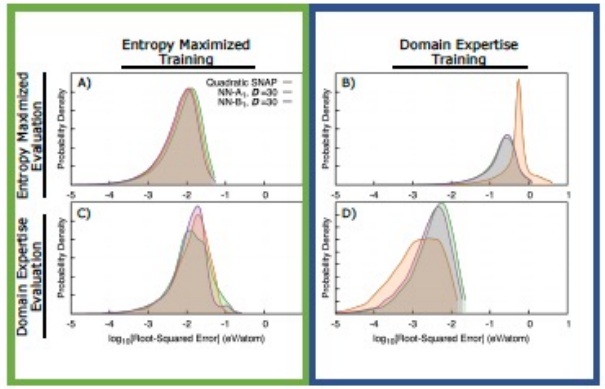
As part of ECP, we developed a Sub-Lattice implementation that greatly improves the size-scaling by introducing an additional level of domain decomposition, benefiting from the locality of transitions.



Numerous other methodological improvements for long-time dynamics:

- Improved Extended Lagrangian algorithm for fast, SCF-free, dynamics in reactive systems
- Improved speculation procedure for ParSplice
- Demonstrations of the advantages of dynamic resources allocation in ParSplice

ML for high-accuracy MD simulations



Now:

- Globally accurate models
- Training process automated using EXAALT framework

Then:

- Locally accurate models
- Limited Transferability
- Labor intensive training process

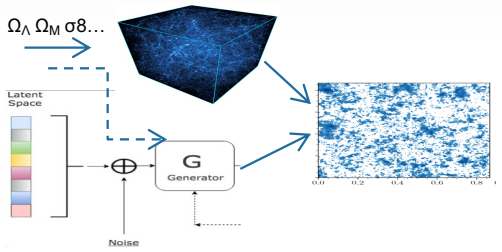
Numerous other methodological improvements for high-accuracy simulations:

- Spin-polarized, DFTB+U, electronic structure in LATTE
- Orbital-free charge-equilibration models coupled with ML potentials
- UQ for ML potentials
- Integrated ML potential development environment

ECP's ExaLearn Co-Design Center: Application Pillars

Surrogates

- ML-created models
- Faster and/or higher fidelity models
- Generative networks
- Using ML to replace complicated physics
- Cosmology



Control

- ML-controlled experiments
- Efficient exploration of complex space
- Reinforcement Learning
- Use RL agent to control light source experiments
- Temperature control for Block Co-Polymer (BCP) experiments

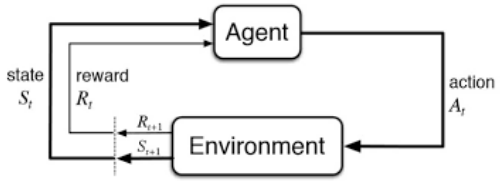
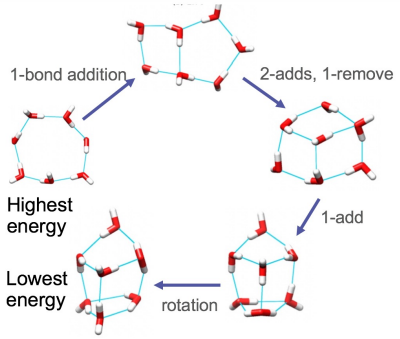


Image courtesy Sutton, Barto, Reinforcement Learning 2017

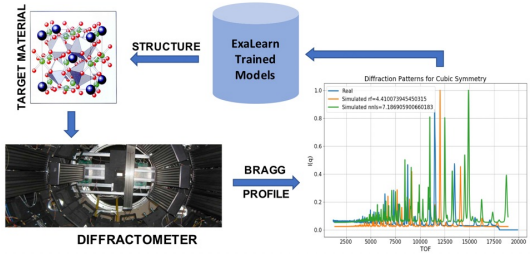
Design

- ML-created physical structures
- Optimized proposal for desired behavior of structure within complex design space
- Graph-Convnets
- Use Graph-CNN to propose new structures that respect chemistry
- Molecular Design



Inverse

- ML projection from observation to original form
- Back-out complex input structure from observed data
- Regression models
- Predicting crystal structure from light source imaging
- Material structure from neutron scattering



EXASCALE COMPUTING PROJECT

PI: Frank Alexander (BNL)

Then (2016) and Now (2023): ExaLearn

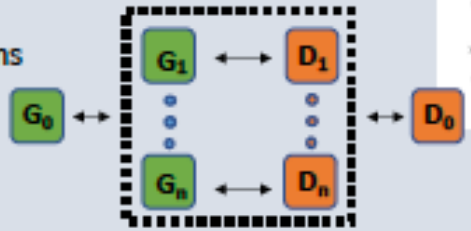
Machine learning for design, control, inverse problems, surrogates

SURROGATES (for Cosmology)

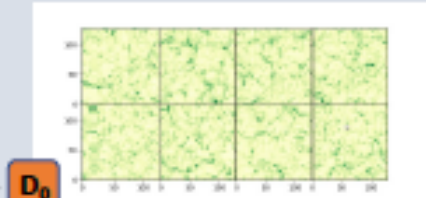
Challenges and Importance: Many DOE simulation efforts could benefit from having realistic surrogate models in place of computationally expensive simulations. These can be used to quickly flesh out parameter space, help with real-time decision making and experimental design, and determine the best areas to perform additional simulations. We are targeting large-scale structure simulations of the universe. As the field is well developed, the scale can easily be ramped up to an exascale ML challenge, and the field is robust enough to explore systematics at the sub-percent level.

THEN

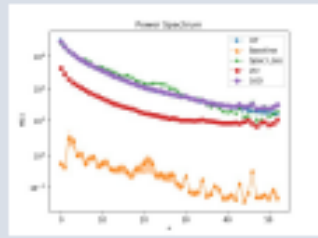
Before ExaLearn, no one had attempted to directly create 3D surrogates of n-body cosmological simulations at any scale. Several groups had worked on 2D slices, and there were some nascent efforts that used a multiscale approach (Perraudin et al., 2019) for GANs where 1283 simulations were down-sampled to $32^3 \times 4$. Visually, they looked good. Yet, statistically, they produced surrogates lacking the quality needed in cosmological analyses. Training for these efforts was limited to single GPUs with 16 GB of RAM.



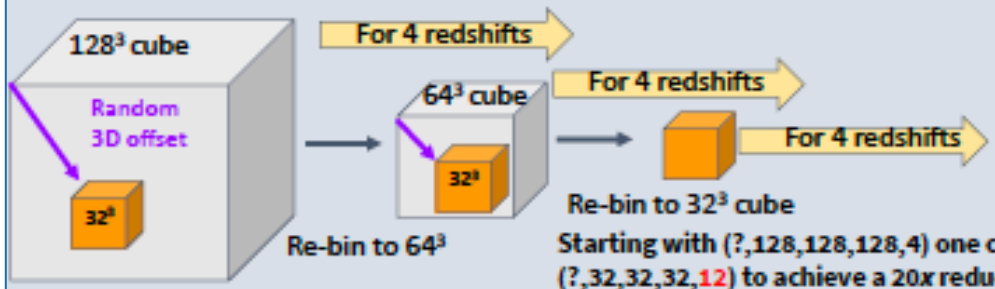
Our MultiGAN approach.



Top Right: 1282 slices of the training data (above) vs. surrogates (below).



Bottom Right: Experimental results comparing physics-informed spectral loss vs. MultiGAN. Using 16 discriminators exceeds the performance of using spectral loss.



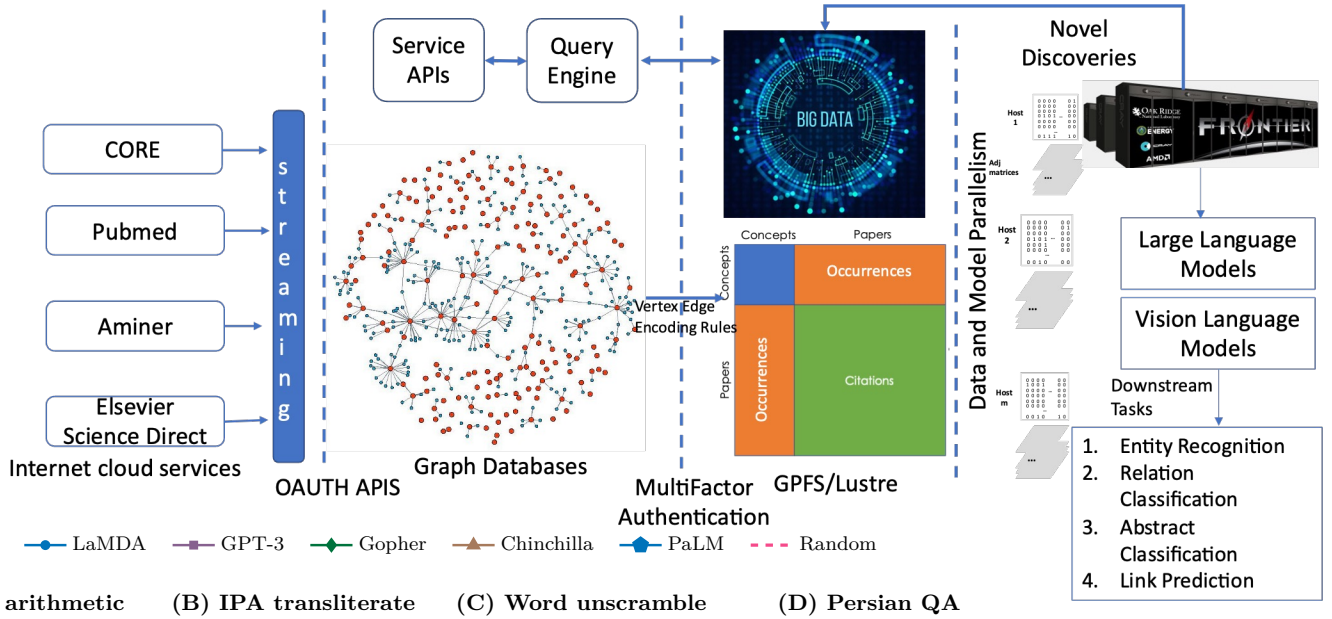
Starting with (?,128,128,128,4) one combines three scales: $32^3 + 64^3 + 128^3$ as: (?,32,32,32,12) to achieve a 20*x* reduction in data size to fit on one GPU.

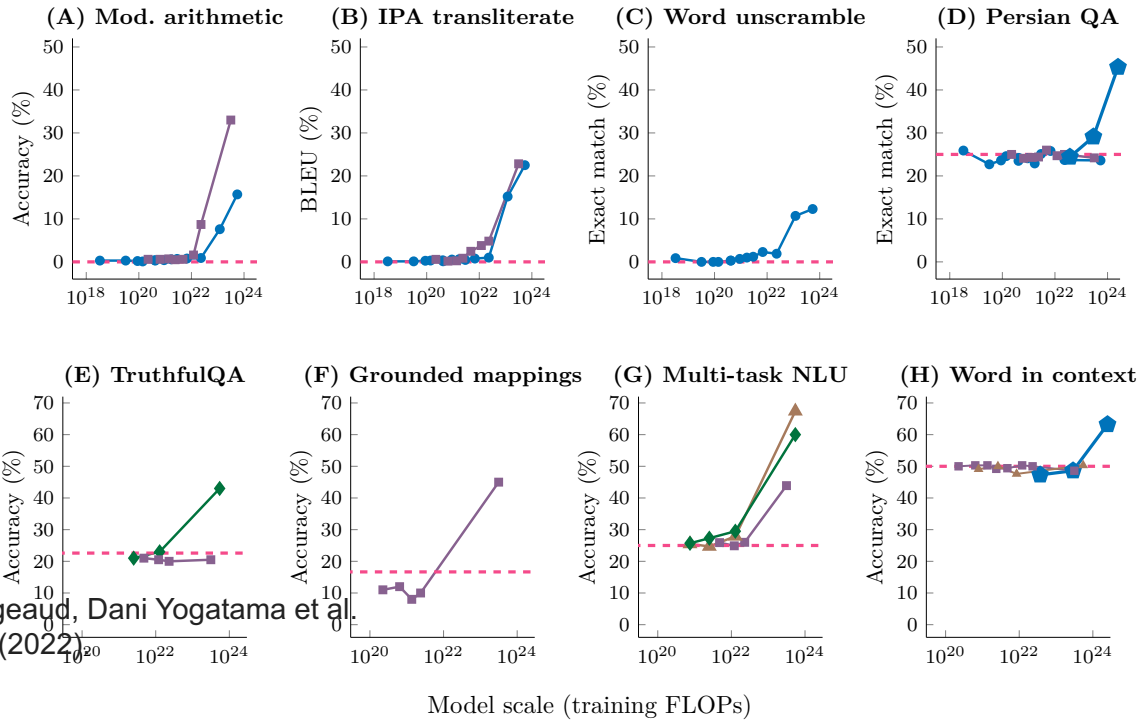
NOW

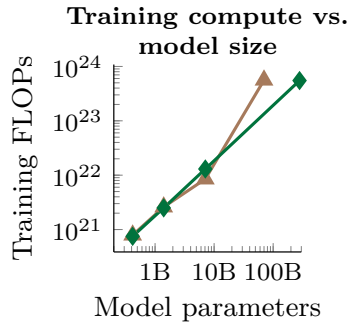
Using the LBANN code and training on ExaSky Nyx simulations, we solved three challenges posed by using GANs for these cosmological surrogates. By employing a multi-discriminator, multi-generator approach and using LBANN's inherent model and data parallelism, we were able to: 1) mitigate the unstable dynamics, oscillatory behaviors, scalability, convergence, and mode collapse issues GANs often face in scientific applications; 2) employ the full machine to train the GANs on Lassen, Perlmutter, and Crusher; and 3) achieve high-quality statistical surrogates for our n-body cosmological simulations.

Can Frontier train the largest AI models (>10^{14} parameters)?

- We are in the quest of demonstrating the HPC needs for training real world scientific AI problems – specifically scientific text and images.
- Pre-train large language models (LLM) such as GPT-3, BLOOM, PALM, LaMDA, Gopher and Vision Language models on scientific texts like Pubmed, Aminer, MAG and materials related publication texts
- Frontier
  - We believe we train up to 150 Trillion FP32 Parameter model in Frontier. This is approximately ~300X bigger than the largest PaLM model with 540B parameters.
  - Training some of these off the shelf large language models could at least take 12 days on Frontier at HPL parallel performance efficiency







Wei, Jason, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama et al. "Emergent abilities of large language models." *arXiv preprint arXiv:2206.07682* (2022).

BIG for Science?

BEYOND THE IMITATION GAME: QUANTIFYING AND EXTRAPOLATING THE CAPABILITIES OF LANGUAGE MODELS

Q: What movie does this emoji describe? 👧🐟🐠🐡

2m:	i'm a fan of the same name, but i'm not sure if it's a good idea
16m:	the movie is a movie about a man who is a man who is a man …
53m:	the emoji movie 🐟🐠🐡
125m:	it's a movie about a girl who is a little girl
244m:	the emoji movie
422m:	the emoji movie
1b:	the emoji movie
2b:	the emoji movie
4b:	the emoji for a baby with a fish in its mouth
8b:	the emoji movie
27b:	the emoji is a fish
128b:	finding nemo

arXiv:2206.04615v1 [cs.CL] 9 Jun 2022

AI for science
What comes after exascale

- Over 1,300 scientists participated in 4 town halls during the summer/fall of 2019
- Research opportunities in AI
  - Biology, chemistry, materials,
  - Climate, physics, energy, cosmology
  - Mathematics and foundations
  - Data life cycle
  - Software infrastructure
  - Hardware for AI
  - Integration with scientific facilities
- Modeled after the Exascale Series in 2007
- ASCAC subcommittee report Sept. 2020

Leadership AI aimed at mission needs

Scientific discovery, user facilities, energy research, environment and national security

Leverages relevant DOE assets

- Exascale class computing

- Exascale class data infrastructure
- Large-scale Experimental Facilities
- Large-scale Scientific Simulation Capabilities

- Interdisciplinary teams

AI for Advanced Properties Inference and Inverse Design

Energy Storage
Proteins, Polymers

AI and Robotics for Autonomous Discovery

Materials, Chemistry, Biology
Light-Sources, Neutrons, ..

AI Based Surrogates for HPC

Climate Ensembles
Effective Zettascale on Exa

AI for Programming and Software Engineering

Code Translation, Optimization
Quantum Compilation, QAlgs

AI for Prediction and Control of Complex Engineered Systems

Accelerators, Buildings, Cities
Reactors, Power Grid, Networks

Foundation AI for Scientific Knowledge

Hypothesis Formation, Math
Theory and Modeling Synthesis

Then (2016) and Now (2023): ExaSGD

Optimization for the modern electric power grid

State of the Art in Power Grid Optimization: Then

ExaGO: Did not exist
HiOp: Built for specialized structural engineering systems (dense systems only on CPU)

Algorithms: Optimization for power grid dominated by approximations because of compute platform constraints

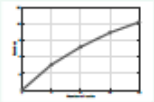
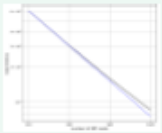
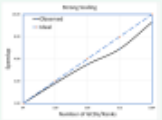
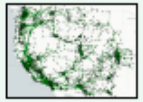
- DC approximations used even though power grid is AC
- Security constraints applied "after the fact"
- Contingencies and scenarios limited to most likely or most worrisome (leaving huge "blind spot" for operations)
- Rapid prototyping of new methods for new platforms constrained by legacy tools

Penetration of Renewables: Renewables introduce instability into system that make old approximations worse

- Grid systems respond to demand which is coupled with uncertain weather
- Inability to sample large range of possible weather futures means we are vulnerable to extreme weather events (TX ice storm 2021, polar vortex 2022, etc...)

Software Environment: Code was run "on laptops" so no push for HPC

- Code doesn't natively run on accelerators, so "good enough" on low end systems was accepted

FY20	Illinois		SCOPF 	 Scaling on Summit @ ORNL for Illinois network with 100 contingencies
FY21	Texas		Stochastic SCOPF 	 Strong scaling efficiency on Summit @ ORNL with 10 wind scenarios and 1000 contingencies
FY22	Western Interconnect		Stochastic SCOPF 	 Strong scaling efficiency on Crusher @ ORNL with 10 wind scenarios and 1000 contingencies
FY23	Western Interconnect		Stochastic SCOPF 	

State of the Art in Power Grid Optimization: Now

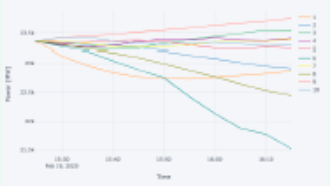
ExaGO: multiple stable full stack software releases
HiOp: General purpose, portable optimization engine (includes mixed dense/sparse and sparse solvers)

Algorithms: New implementations take advantage of increased memory, compute power on a single rank making it possible to capture important realism

- High fidelity AC physics included—**more accurate model leads to efficiency gains and better situational awareness**
- Security constraints applied inside optimization loop means compliance with regulations is built into the solution– **optimal solutions are chosen with security baked in**
- Accelerator based implementation and HPC engine allow for vastly larger number of contingencies and scenarios to be explored—**better awareness leads to better national readiness**
- Portability built into ExaSGD code enables rapid prototyping of new methods and platforms—**flexibility ensures ExaGO is innovative and impactful to industry**

Penetration of Renewables: better representation of weather effects improves grid management especially as renewables penetrate deeper

- Stochastic variability is enabled by Exascale computing which allows for **sampling of a large number of possible weather scenarios** which influence generation in different ways
- Accounting for more possible renewable generation profiles leads to **more stable operation**

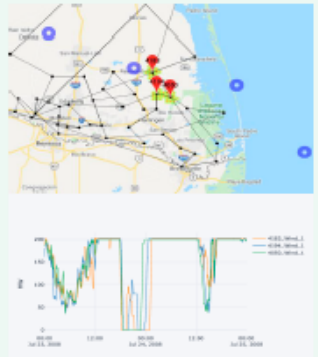


Wind power variability
10 wind power scenarios aggregated over the ACTIVSg70k test system enabled by ExaGO. We highlight the scenario with a significant drop in wind generation (~2GW in 40 min which is far outside changes expected from conventional inertia-based generation), and remark that AC power flow physics will enable planners to better understand the effects of significant variability in renewable generation on large power systems.

Software Environment:

- Optimization, linear solver and domain model code all running on accelerators enables rapid time to solution for power grid models– **enables calculations the size of Western Interconnect and larger on single node systems**
- Portable software stack running on Exascale systems enables optimization over thousands of weather scenarios and for highly complex contingencies (i.e. loss of multiple power grid elements simultaneously)—**enables evaluation of complex damage "what if's" like hurricane or cyber attack**

Voltage and Reactive Power Event (AC Effects Matter!)
Voltage and reactive power event on ACTIVSg2000 test system caused by broken transmission lines during a hurricane strike.



Hurricane Dolly and Renewables
Path of the eye of Hurricane Dolly (blue), and the effect on power generation from 3 wind farms. Timeseries generation data used as input for AC optimal power flow computations.

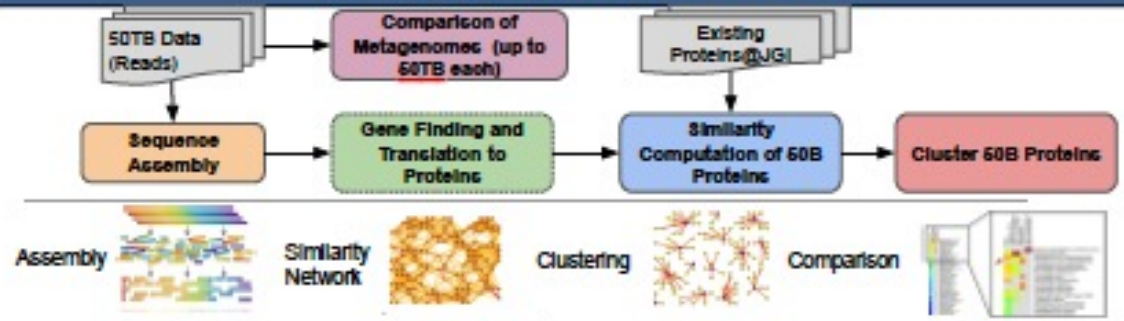
Then (2016) and Now (2023): ExaBiome

Microbiome analysis

Microbiomes are Critical to Energy and the Environment

- **Microbes:** single cell organisms, such as bacteria and viruses
- **Microbiomes:** communities of 1000s of microbial species, less than 1% individually culturable in a lab (and thus sequenced)
- **Metagenomics:** genome sequencing on these communities (growing exponentially)

Exabiome Exascale Challenge



Distributed Implementations

- MetaHipMer
- diBELLA/ELBA
- HipMCL
- PASTIS
- GOTTCHA2
- MerBench

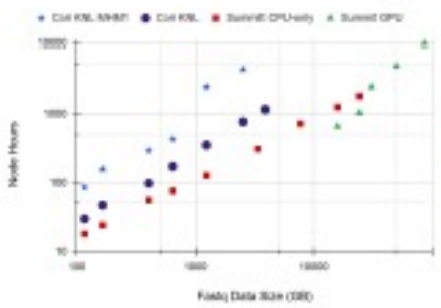
Parallelism Motifs

- Hash tables
- Sorting
- Graph traversal
- Generalized N-body
- Sparse matrices
- Sequence alignment

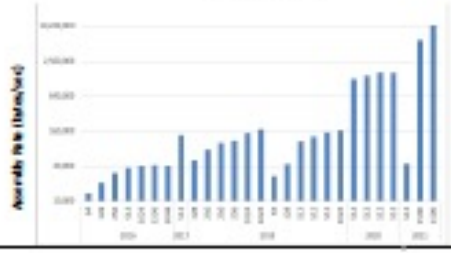
Large-Scale Metagenome Assembly with MetaHipMer

MetaHipMer "then" (at the start of ECP)

MetaHipMer had just been written as an modification of the single genome assembler, HipMer. It was written in a combination of UPC and MPI. It was an unstable early stage code, and the quality of the assemblies was not yet equivalent to that produced by existing metagenome assemblers. It could run at scale, but one of the stages in the pipeline could only run on a single node, and for large assemblies, this required a high-memory node. The earliest large scale assembly of 2.6TB on 512 Cori KNL nodes in March 2018 took 1800 node hours, and larger runs were not possible because of the memory constraints of single nodes.



Scaling with input data size. Multiple datasets are represented and dataset composition affects assembly time. The speedup from MHM1 to MHM2 is due to the transition to UPC++ and algorithmic improvements.



Assembly rate in bytes per second over the course of the project. On the same number of nodes, the speedup from 2016 to 2021 is over 250x due to algorithmic improvements, use of GPUs and UPC++

MetaHipMer "now"

- Production quality code used extensively by the JGI for client assemblies (only assembler able to do terabase-scale metagenome assembly).
- Code rewritten entirely in UPC++, which would not have been possible without the ECP Pagoda project.
- Many algorithmic improvements, e.g. implementation of a new scaffolding algorithm for metagenomes that scales and replace previous bottleneck single node stage.
- Improved data locality, e.g. minimizers (similar to locality sensitive hashing) combined with reordering of input data reduced communication volumes by 5x.
- Support for GPUs in several stages, requiring new algorithms.
- A big surprise was up to 7x speedup for some stages on GPUs - we initially believed that the asynchronous, random access nature of the code would make it hard to exploit GPUs.
- Large impact: went from a low quality assembly of perhaps 1TB to a high quality assembly of over 30TB (and our target is 50TB). Code performance improved by a couple of orders of magnitude over the course of the project

Future Directions and Challenges

- Port more of the assembly pipeline to GPUs for increased speedups.
- Enable the assembly of long-read datasets with MetaHipMer
- Modifications for single genome assembly (porting features from the old HipMer code base, which is still used for single genome assemblies)
- Algorithmic improvements through machine learning, e.g. to determine how to best traverse the contig graph in scaffolding

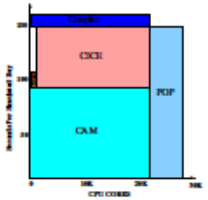
Then (2016) and Now (2023): Energy Exascale Earth System Model

Cloud-resolving Climate Modeling of the Earth's Water Cycle

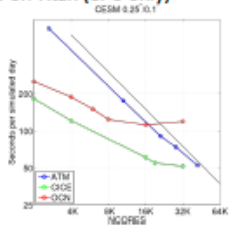
Then

Baseline model (non-MMF)

- E3SM v0 = CESM 1.2 (branch point of the E3SM project)
- High resolution configuration: 25 km atmosphere, 10 km ocean
- GPU acceleration: None
- Hydrostatic (no nonhydrostatic capability)
- 25 km model running at 1.5 SYPD on Titan (CPU only)



Coupled model performance



Strong scaling of atmosphere, ocean and sea ice

Performance at Cloud Resolving resolution

- Performance of E3SM v0 with the atmosphere running at 3 km (cloud resolving) resolution, using all of Titan
- E3SM had never run at 3 km resolution and so performance was estimated based on 25 km atmosphere
- Performance extrapolated to all of Titan, assuming perfect weak scaling, 20% coupler overhead, ocean concurrent with other components:
  - Max(atm_time+ice_time, ocn_time) * 1.2
- Figure of Merit (FOM) = 0.11 Simulated Years Per Day on all of Titan

MMF Cloud Resolving Capability

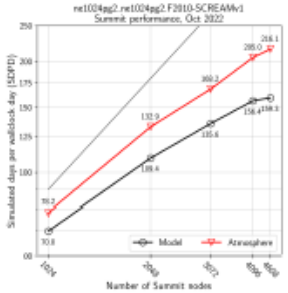
- Promising research results using MMF in CESM
- Not integrated into E3SM

THEN: Figure of Merit (FOM): 0.011 Simulated Years Per Day

Now

Baseline model (non-MMF)

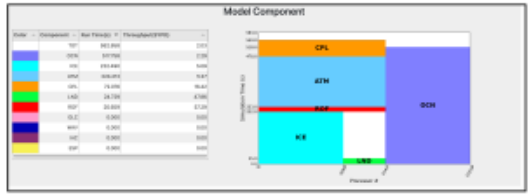
- SCREAM: E3SM's 3 km cloud resolving atmosphere model
- Rewritten from scratch, led by E3SM with many contributions from ECP E3SM-MMF project
- Nonhydrostatic dycore with HEVI-IMEX
- Atmosphere with prescribed SST simulations running on all of Summit (obtaining 0.43 SYPD) on 4600 nodes.

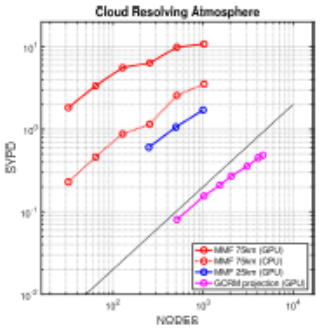


- E3SM-MMF "AMIP" simulations.
- CPU node vs 6 GPUs:
  - w/ 3D CRM: excellent GPU speedup (>20x) and scaling
  - w/ 2D CRM: 9x CRM speedup
- Baseline GCRM projection
  - Based on dycore GPU performance
- E3SM-MMF significantly faster and more efficient than GCRM approach on GPUs

E3SM-MMF Fully coupled model running on Summit

- MMF fully integrated into E3SM with many science and algorithmic improvements, and dramatically improved I/O performance via SCORPIO + ADIOS
- KPP Challenge problem running on Summit
- Weather resolving atmosphere (25 km) coupled with cloud resolving convection and turbulence (1 km)
- Coupled to the MPAS Ocean/Ice components running on the 18to6 km (Eddy Resolving) mesh
- Running at 2.03 SYPD





Node comparison: 2xP9 vs 6xV100

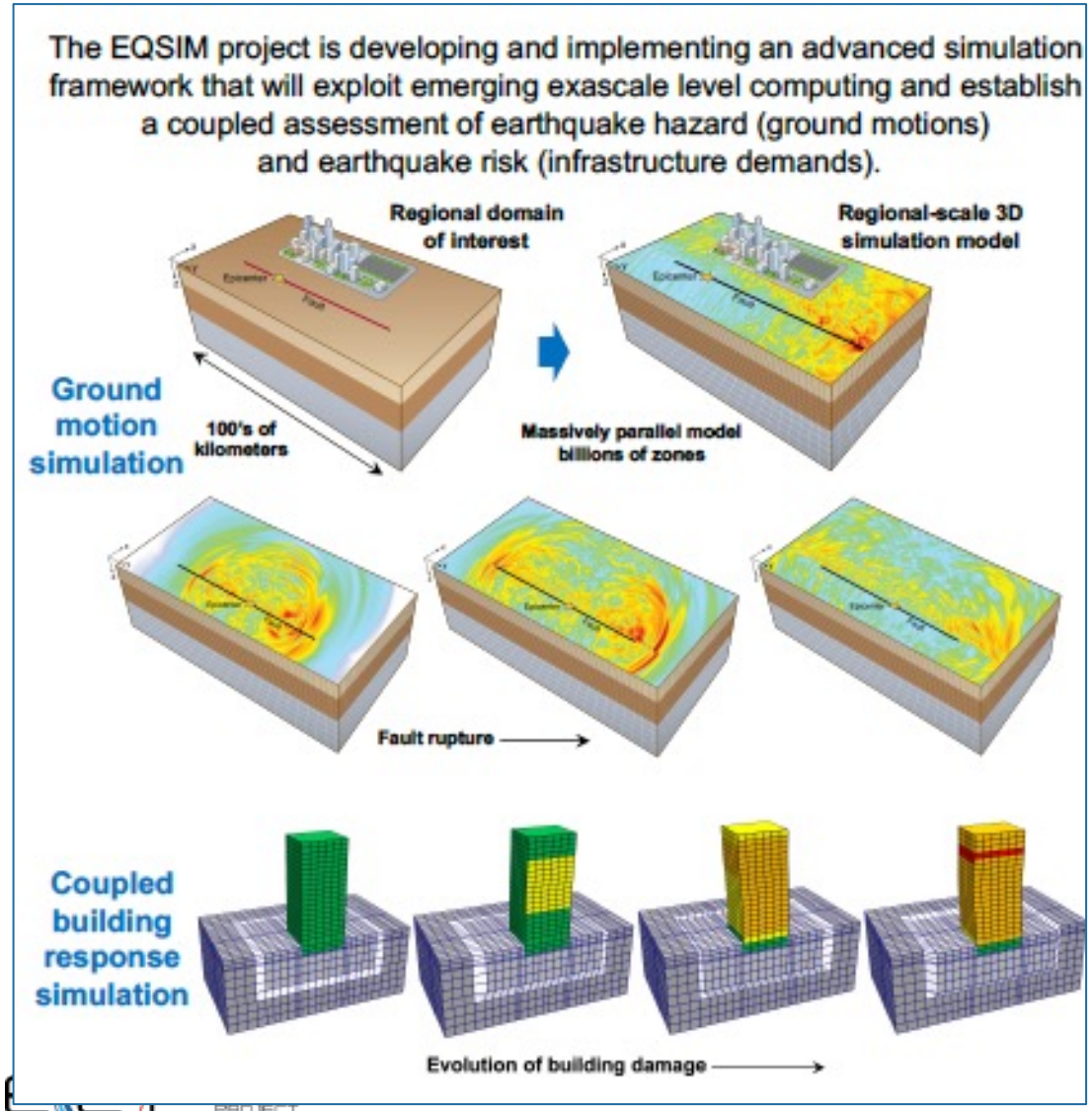
- Strong scaling of E3SM-MMF atmosphere component vs baseline model on Summit
- Red curves: 75 km benchmark problem, GPU vs CPU: Good GPU acceleration out to 15 GCRMS per GPU)
- Bule: 25 km KPP challenge problem running on GPUs – should scale to all of Summit
- Purple: Baseline model running on GPUs
- MMF approach achieves many aspects of a cloud resolving model and is far more efficient than the full cloud resolving baseline approach

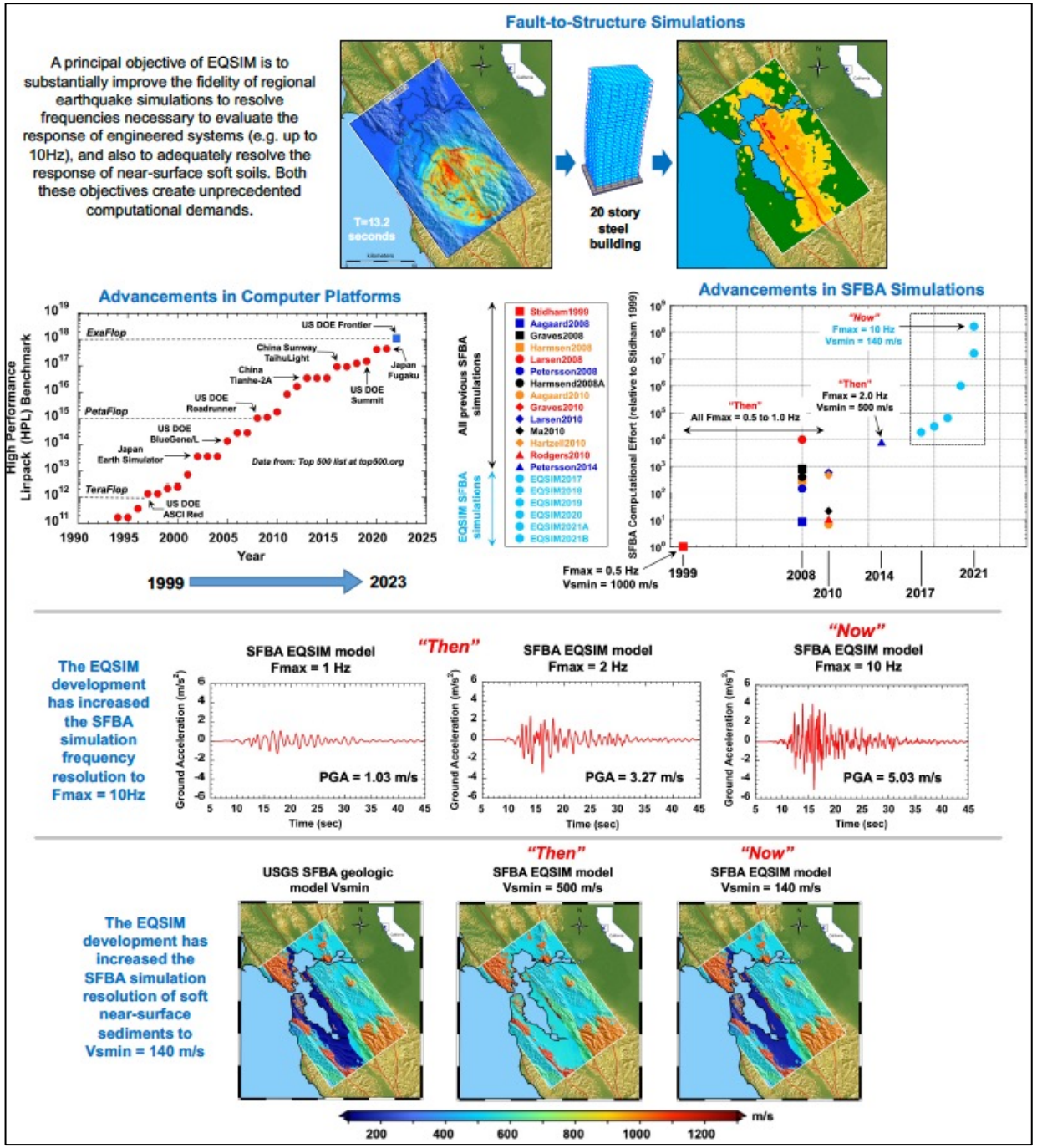
Figure of Merit (FOM): 2.0 SYPD on Summit (181x FOM improvement)

KPP Challenge problem: Need to achieve 2.6 SYPD on Frontier

Then (2016) and Now (2023): EQSIM

End-to-end simulation of earthquake phenomena





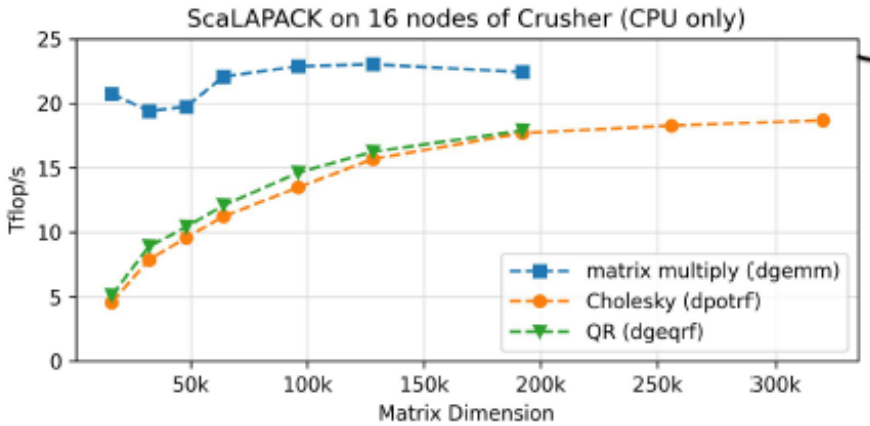
Then (2016) and Now (2023): CLOVER

Preparing linear algebra and FFT for exascale

THEN: ScaLAPACK

First released in 1995, ScaLAPACK is a Fortran 77 library providing dense linear algebra routines for distributed memory machines. While very successful, ScaLAPACK has many limitations in a modern environment:

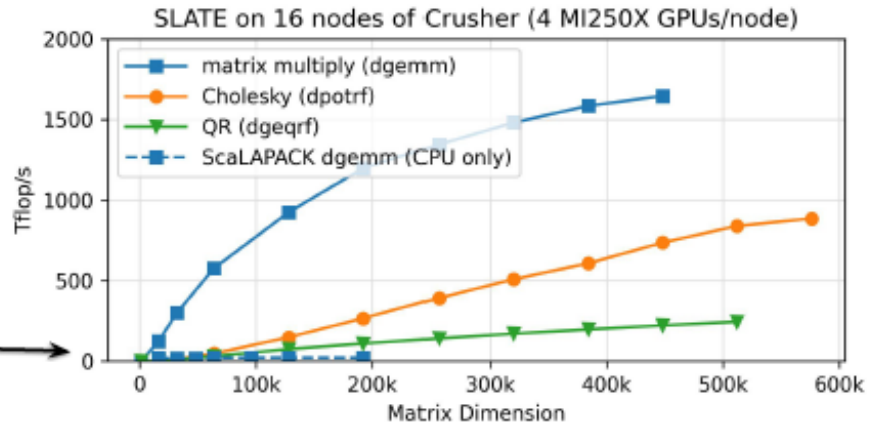
- Cumbersome interfaces with numerous arguments; no C/C++ bindings.
- No multi-threaded CPU execution — typically 1 MPI rank per core.
- No GPU acceleration.
- No overlap of computation and communication (e.g., no lookahead).
- No communication avoiding routines.



NOW: SLATE

SLATE is a modern C++ library providing common linear algebra routines for distributed, GPU-accelerated machines.

- Covers ScaLAPACK functionality including BLAS (matrix multiplication, triangular solves), linear system solvers, least squares solvers, eigenvalue and singular value decompositions.
- BLAS++ and LAPACK++ portability layer across GPU architectures (CUDA, ROCm, oneAPI).
- OpenMP tasking to overlap communication and computation.
- Adds new algorithms including mixed-precision solvers and communication-avoiding algorithms: CAQR, CALU, 2-stage eigenvalue and SVD.



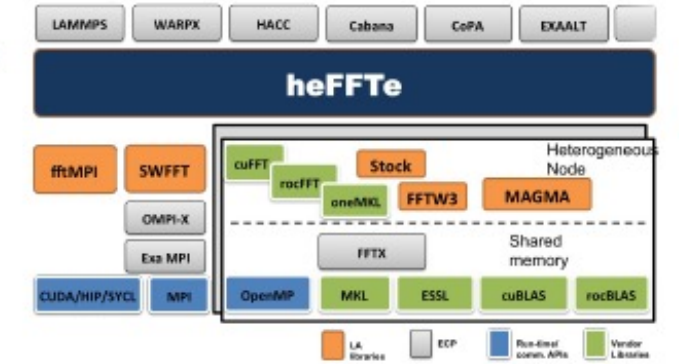
Then (2016) and Now (2023): CLOVER

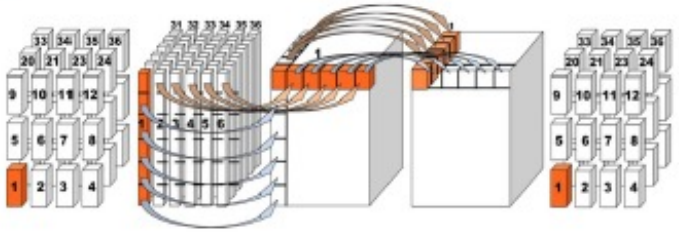
Preparing linear algebra and FFT for exascale

heFFTe

THEN

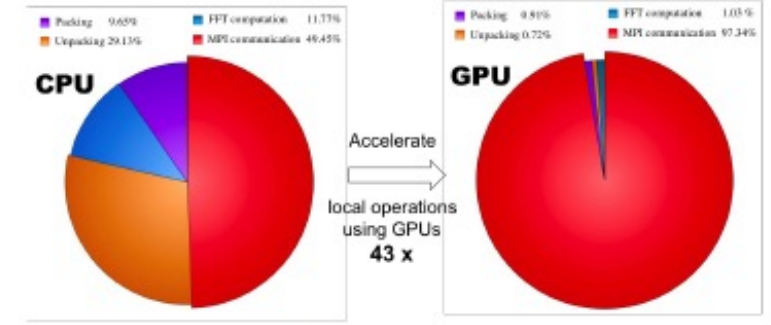
- The fast Fourier transform (FFT) is used in many domain applications - more than a dozen ECP applications use FFTs in their codes;
- State-of-the-art libraries like FFTW were no longer actively developed for emerging platforms;
- No GPU support for distributed multi-dimensional FFTs at the time;
- Some ECP application constructed their own FFTs directly in applications, e.g., fftMPI for LAMMPS and SWFFT for HACC;
- Features and application-specific needs were not supported uniformly;
- The goal was to leverage the existing FFT capabilities and build a sustainable FFT library for Exascale.



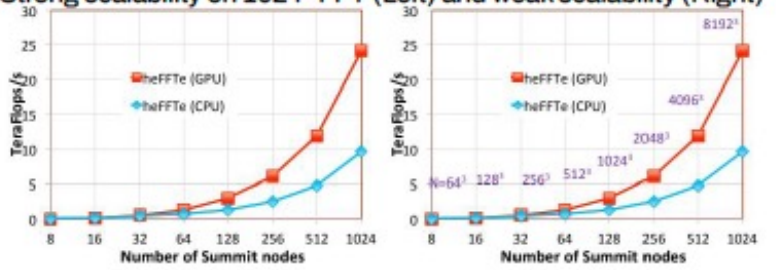


NOW

- GPUs (e.g., V100 on Summit) accelerate local FFT computations more than 40 x
- heFFTe supports multiple backends for Nvidia GPUs, AMD GPUs, Intel GPUs and multicore CPUs;
- Novel features such as Batched 2-D and 3-D FFTs
- Support FFT convolution, sine, and cosine transforms;
- Support for real and complex FFTs, multiple precisions and approximate FFT;
- Very good strong and weak scalability (Figure on right);
- FFT benchmark for MPI collectives and other FFT libraries.



Strong scalability on 1024^3 FFT (Left) and weak scalability (Right)



Then (2016) and Now (2023): CLOVER

Preparing linear algebra and FFT for exascale

Ginkgo

THEN

MAGMA-sparse as experimental code for the development of sparse linear algebra for NVIDIA GPUs serves as starting point and reference for the development of Ginkgo.

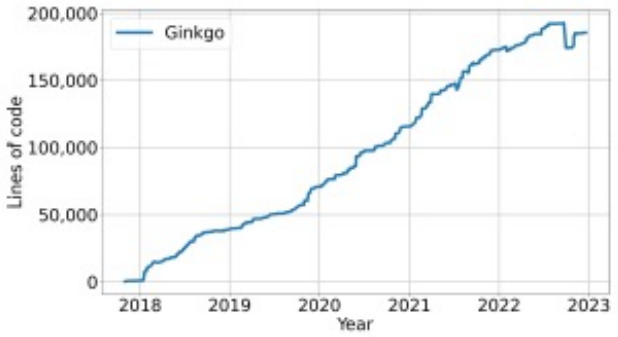
Before the first line of code hits the repository, a complete year is spent with whiteboard discussions on the design.

Ginkgo's development embraces:

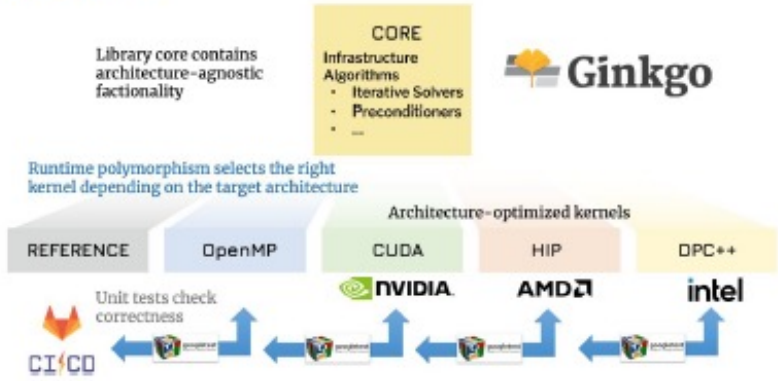
- Platform Portability
- Performance
- xSDK Community Policies
- Modern C++
- CI/CD and unit testing
- Open source & permissive licensing
- Rapid integration of new algorithms

MAGMA SPARSE

ROUTINES	BiCG, BiCGSTAB, Block-Asynchronous Jacobi, CG, CGS, GMRES, IDR, Iterative refinement, LOBPCG, LSQR, QMR, TFQMR
PRECONDITIONERS	ILU / IC, Jacobi, ParILU, ParILUT, Block Jacobi, ISAI
KERNELS	SpMV, SpMM
DATA FORMATS	CSR, ELL, SELL-P, CSR5, HYB



NOW

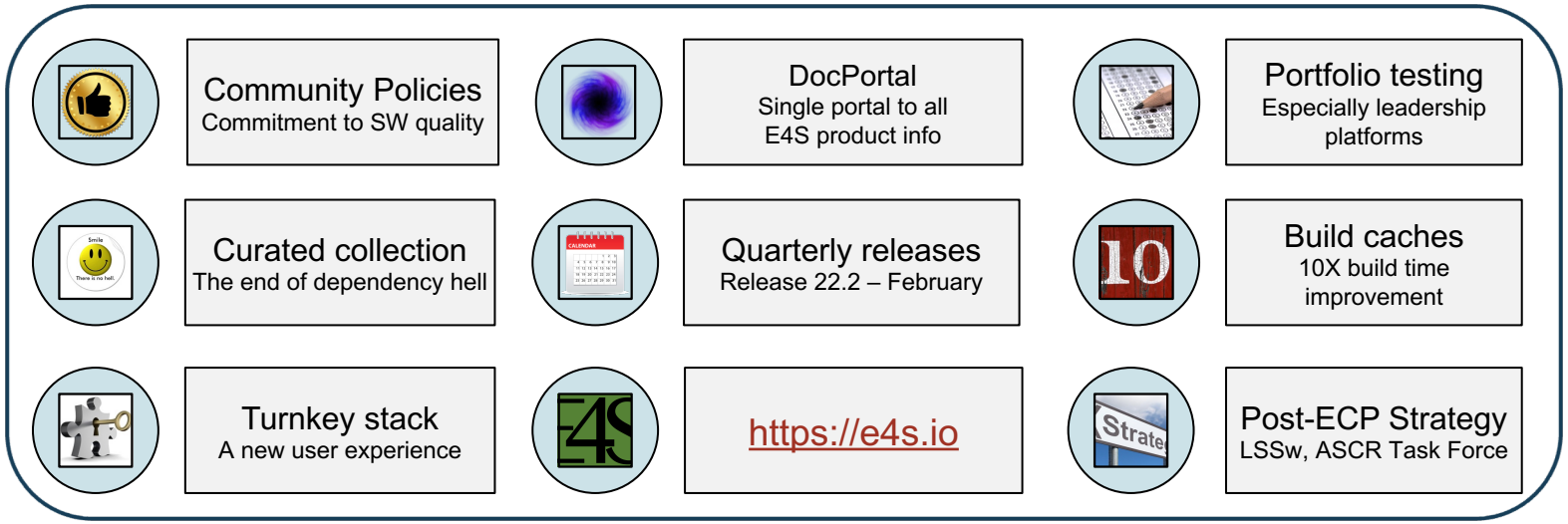


Production-ready modern C++ linear algebra library for single-node and multi-node execution with native support for GPU architectures from AMD, Intel, and NVIDIA.

	Functionality	OMP	CUDA	HIP	DPC++
Basic	SpMV	✓	✓	✓	✓
	SpMM	✓	✓	✓	✓
	SpGeMM	✓	✓	✓	✓
Krylov solvers	BiCG	✓	✓	✓	✓
	BiCGSTAB	✓	✓	✓	✓
	CG	✓	✓	✓	✓
	CGS	✓	✓	✓	✓
	GMRES	✓	✓	✓	✓
	IDR	✓	✓	✓	✓
Preconditioners	(Block-)Jacobi	✓	✓	✓	✓
	ILU/IC		✓	✓	✓
	Parallel ILU/IC	✓	✓	✓	✓
	Parallel ILUT/ICT	✓	✓	✓	✓
	Sparse Approximate Inverse	✓	✓	✓	✓
Batched	Batched BiCGSTAB	✓	✓	✓	
	Batched CG	✓	✓	✓	
	Batched GMRES	✓	✓	✓	
	Batched ILU	✓	✓	✓	
	Batched ISAI	✓	✓	✓	
	Batched Jacobi	✓	✓	✓	
AMG	AMG preconditioner	✓	✓	✓	✓
	AMG solver	✓	✓	✓	✓
	Parallel Graph Match	✓	✓	✓	✓
Sparse direct	Symbolic Cholesky	✓	✓	✓	✓
	Numeric Cholesky	UNDER DEVELOPMENT			
	Symbolic LU	✓	✓	✓	✓
	Numeric LU	✓	✓	✓	
	Sparse TRSV	✓	✓	✓	
Utilities	On-Device Matrix Assembly	✓	✓	✓	✓
	MC64/RCM reordering	✓			
	Wrapping user data		✓		
	Logging		✓		
	PAPI counters		✓		

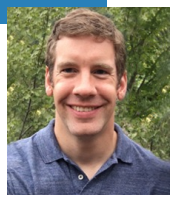
ST's Extreme-scale Scientific Software Stack (E4S) is a key ECP product to sustain and evolve

- E4S: HPC software ecosystem – a curated software portfolio
- A **Spack-based** distribution of software tested for interoperability and portability to multiple architectures
- Available from **source**, **containers**, **cloud, binary caches**
- Leverages and enhances SDK interoperability thrust
- Not a commercial product – an open resource for all
- Growing functionality: November 2022: E4S 22.11 – 100+ full release products



Spack

https://spack.io



Spack lead: Todd Gamblin (LLNL)

E4S

https://e4s.io

E4S lead: Sameer Shende (U Oregon)

Also includes other products, e.g.,
AI: PyTorch, TensorFlow, Horovod
Co-Design: AMReX, Cabana, MFEM

ECP: Key Takeaways

- The Exascale Computing Project (ECP) is not *just* about developing and demonstrating the ability of new and enhanced DOE mission critical applications to tackle currently unsolvable problems of National interest . . . but we also are **building and deploying a new Extreme Scale Scientific Software Stack** (E4S – e4s.io) that greatly lowers the barrier to adoption of new technologies and to porting on advanced hardware. We are building a scientific software ecosystem for decades to come that is present and supports scientific computing from laptops to desktops to clusters to leadership systems

- The fundamental tenant of ECP is not about building boutique applications and a software ecosystem that can only execute on the Nation's largest systems, but it is about ***accelerated node computing*, namely designing, implementing, delivering, and deploying advanced agile software that effectively exploits heterogeneous node hardware on today and tomorrow's laptops and desktops**

- We view ***accelerators* as any compute hardware specifically designed to accelerate certain mathematical operations** (typically with floating point numbers) that are typical outcomes of popular and commonly used algorithms. We often use the term GPUs synonymously with accelerators.

- Compute hardware, from laptop to the largest systems in the world (e.g., ORNL's Summit system), are made up of *accelerated nodes*. **Accelerated-node computing is *here to stay***
  - Accelerators today: GPUs Tomorrow: better GPUs or FPGAs or other ASICs? Near future: quantum?

- **ECP's first-mover applications & E4S software stack are available for testing** (even on laptops) and have greatly demystified and **lowered the barrier to productive utilization** of heterogeneous accelerated-node hardware.

Retrospective

- The US Department of Energy (DOE) has been a leader in High Performance Computing and ”invented” it for the purposes of “design predictability” 80 years ago. **Lots of lessons learned and ROI evidence to share**. ☺
- **Development and application of advanced, predictive modeling and simulation (M&S) – both computational and data science – has long been a mainstay and critical crosscutting technology for the DOE** and its National Laboratories (17 of them!) in achieving its mission goals in science, technology, and national security. This has **never been more vibrant and foundational than today**.
- **Accelerated compute performance (FLOPS, memory, memory B/W, etc.) and enhanced physical models, numerical algorithms, and software architecture enabled by this performance directly correlate with more predictive M&S tools, technologies, outcomes, impact**. This does not come without difficulties, challenges, pain, and perseverance: from GF to TF to PF to EF. We celebrate these milestones - **each one comes with ”tipping points” that are disruptive for app and software stack development yet accompanied by (often unanticipated) high ROI**
- The **EF “exascale era” (>10^{18} floating operations / sec) is upon us, and many institutions and agencies have been preparing and investing** for this milestone for over a decade: DOE included!
- DOE’s Exascale Computing Initiative (ECI), of which the Exascale Computing Project (ECP) is a part, was initiated almost six years ago and is poised and **ready to demonstrate the tremendous “science return” of this technology**

Questions?

kothe@ornl.gov, https://www.exascaleproject.org/contact-us/

For more info

- Alexander F. et al. *Exascale Applications: Skin in the Game,* Phil. Trans. R. Soc. A 378: 20190056 (2020) (http://dx.doi.org/10.1098/rsta.2019.0056).
- Douglas Kothe, Stephen Lee, and Irene Qualters, *Exascale Computing in the United States,* Computing in Science and Engineering 21(1), 17-29 (2019).