MEIFENG LIN
BROOKHAVEN NATIONAL LABORATORY

# COMPUTATIONAL SCIENCE INITIATIVE

BROOKHAVEN
NATIONAL LABORATORY

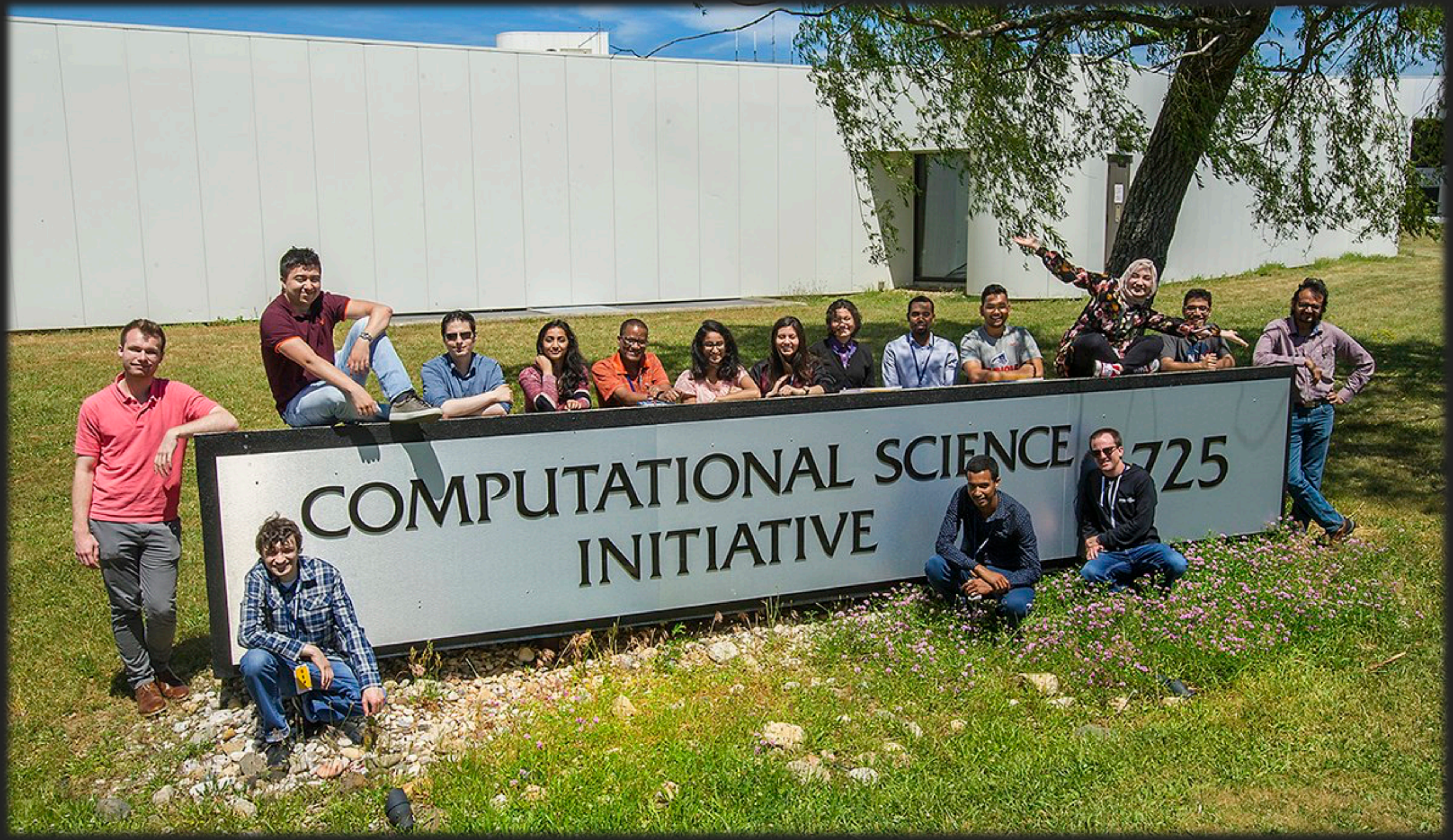
# OUTLINE

- BNL/CSI Overview
- HPC Project Highlights
- Some Comments/Thoughts on Exascale Computing
- Some Side Notes

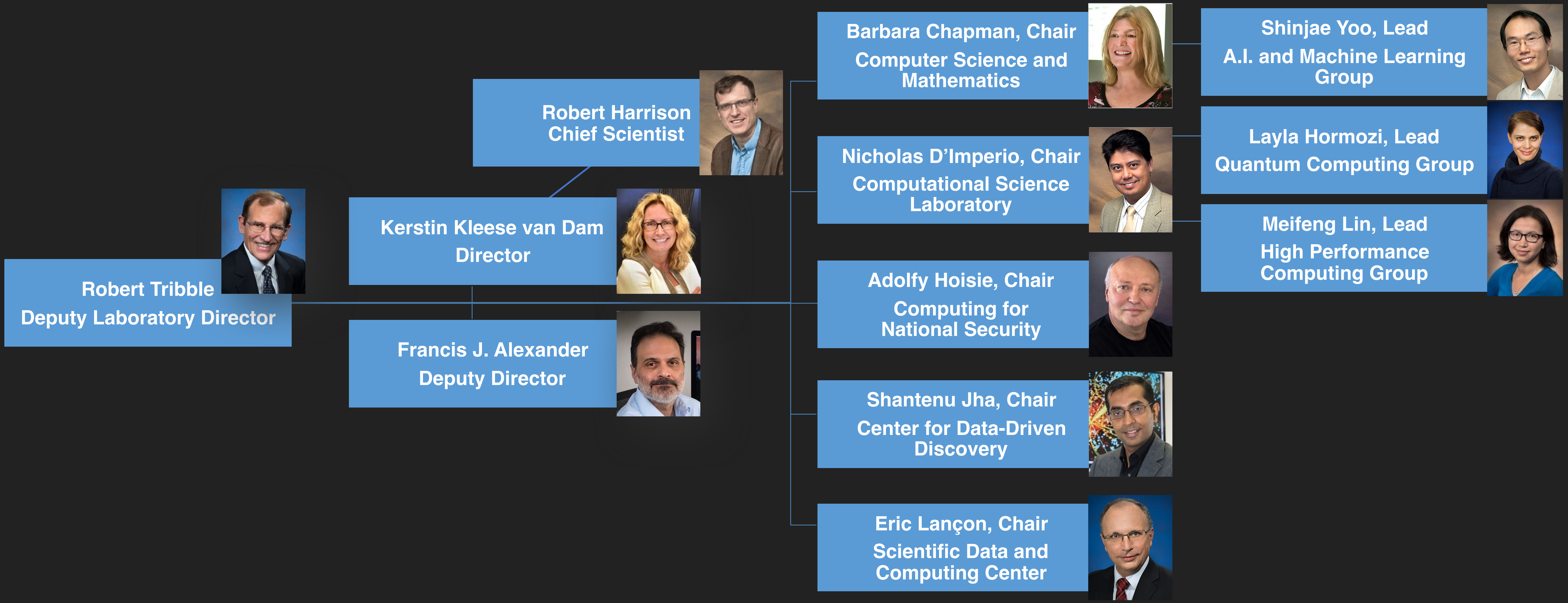
# COMPUTATIONAL SCIENCE INITIATIVE



- Established in 2015
- An umbrella to bring together computing and data expertise across BNL
- Aims to foster interdisciplinary collaborations in domain sciences, computer science, applied math and data analytics.
- **Focus areas:**
    - High performance and novel computing, including quantum computing
    - Data analytics at scale, incl. scalable machine learning, visual analytics, workflow, provenance, etc.
    - State-of-the-art computing and storage facility
- 5 departments, ~50 staff members and growing!



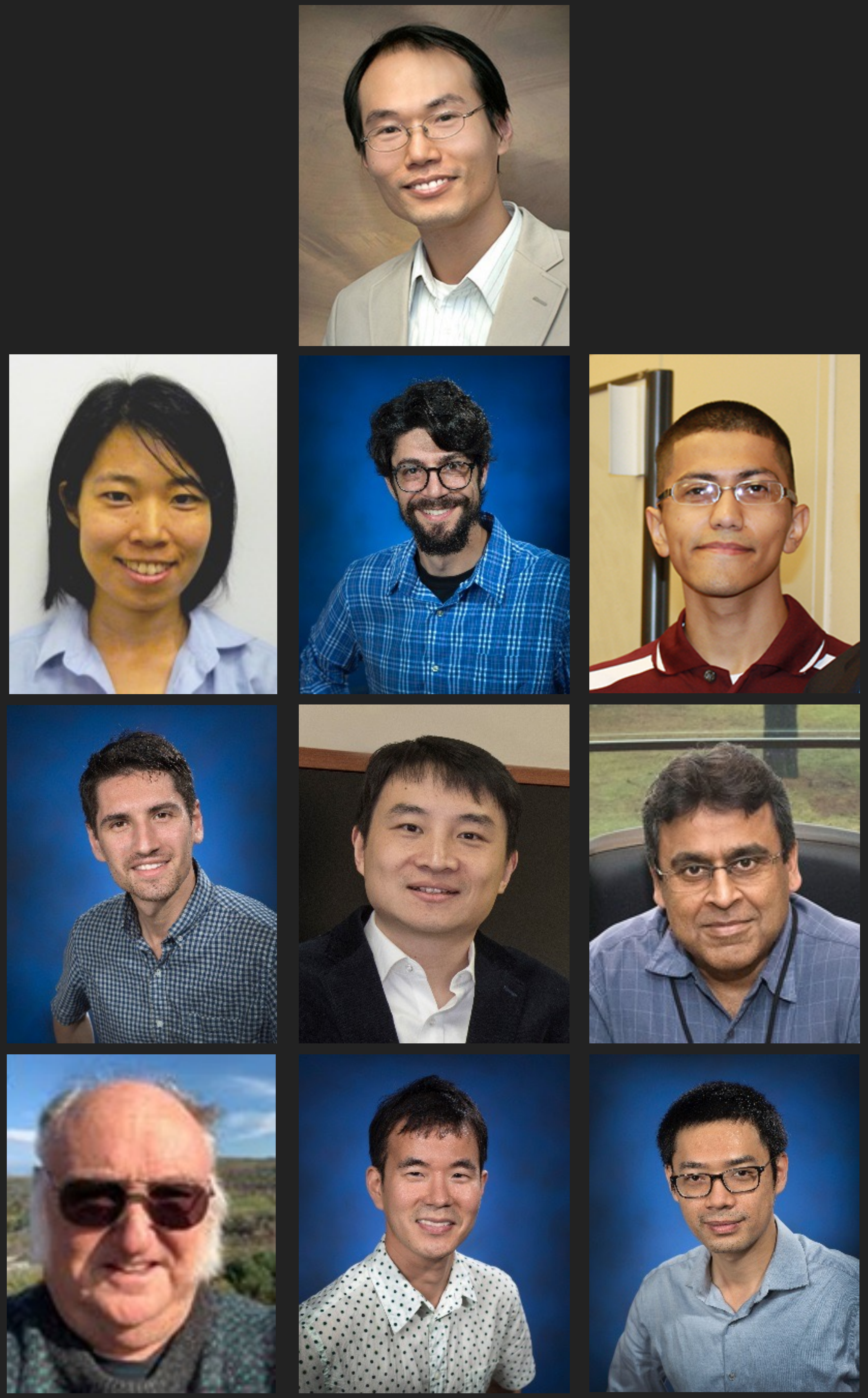

# CSI ORGANIZATIONAL STRUCTURE



# MACHINE LEARNING AND A.I. GROUP



- Group Lead: Shinjae Yoo
- Specific Focus:
  - Real-Time Analysis of Experimental Data (NSLS II, CFN, Cryo-EM, Solar Power)
  - Causal Analysis (Biology, Power Grid)
  - Natural Language Processing for Science
  - Robustness, Explainability, Reproducibility
  - Quantum Machine Learning



# NEW QUANTUM COMPUTING GROUP

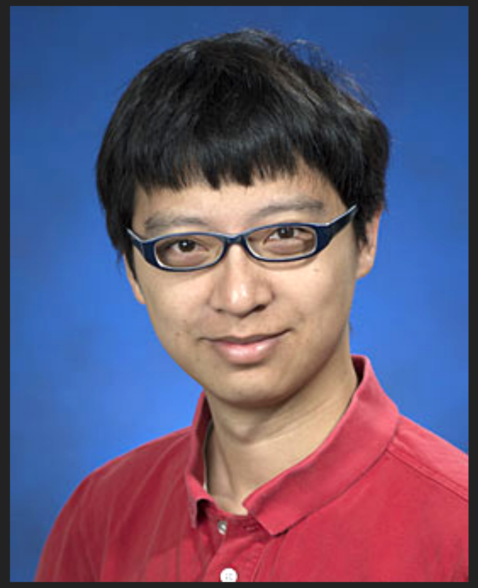


- Group Lead: Layla Hormozi
- Specific Focus:
  - Quantum Networking
  - Connecting Quantum Networking and Quantum Computing
  - Optimized Quantum Algorithm Development for Nuclear, High Energy, and Condensed Matter Physics
  - Quantum Error Characterization and Correction





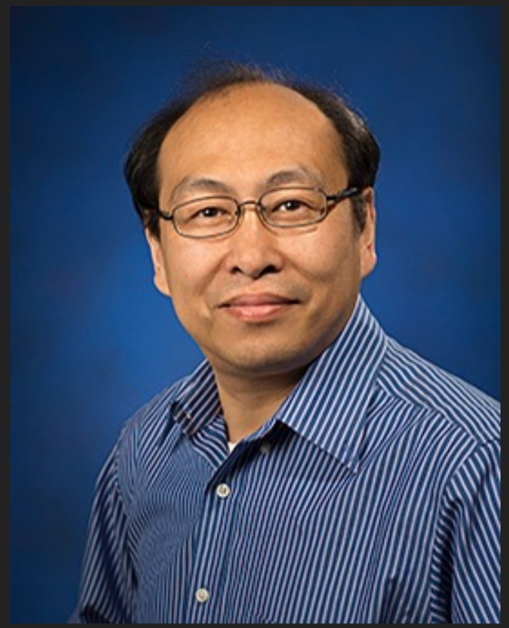


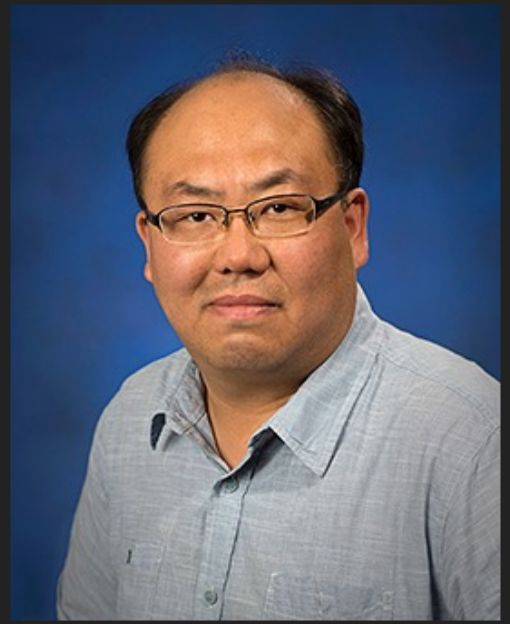


# HIGH PERFORMANCE COMPUTING GROUP



- Group Lead: Meifeng Lin
- The High Performance Computing Group at CSI help the scientists get their codes to run on modern computing architectures
- Research domains range from materials science, quantum chemistry, high energy and nuclear physics, climate science, etc.
- Making use of state-of-the-art software tools and hardware architectures:
    - Performance profiling, analysis and modeling
    - MPI, OpenMP, OpenACC, …
    - CUDA, HIP, SyCL, …
    - Performance portable frameworks





# HPC PROJECT HIGHLIGHTS

# HPC for NSLS II: X-ray Ptychographical Image Reconstruction via Distributed & GPU Computing


**Leo Fang** CSI | Zhihua Dong CSI | Xiaojing Huang NSLS-II | Hanfei Yan NSLS-II | Sungsoo Ha CSI | Wei Xu CSI | Yong Chu NSLS-II | Stuart Campbell NSLS-II | Meifeng Lin CSI


References:


1. Dong *et al.*, NYSDS 2018 (arXiv:1808.10375)
2. Fang *et al.*, "*Accelerated Computing for X-ray Ptychography at NSLS-II*", book chapter in "*Handbook on Big Data and Machine Learning in the Physical Sciences*"
3. Fang *et al.*, in preparation


Work supported by BNL LDRD #17-029, and in part by DOE BES

BROOKHAVEN NATIONAL LABORATORY

# REAL-TIME ANALYSIS AND STEERING OF EXPERIMENTS

- Facility users at CFN and NSLS II typically have limited time allocated.
- Getting the right setup of the experiments often takes trial-and-error.
- Brighter light sources mean faster data rates and larger data volumes.
  - Analyzing data could take a long time
  - Affects the number of experiments users can do
- Need to improve *in-situ* data analysis tools
  - Speed - HPC
  - Usability - intuitive GUI
  - Maintainability - high-level programming abstractions

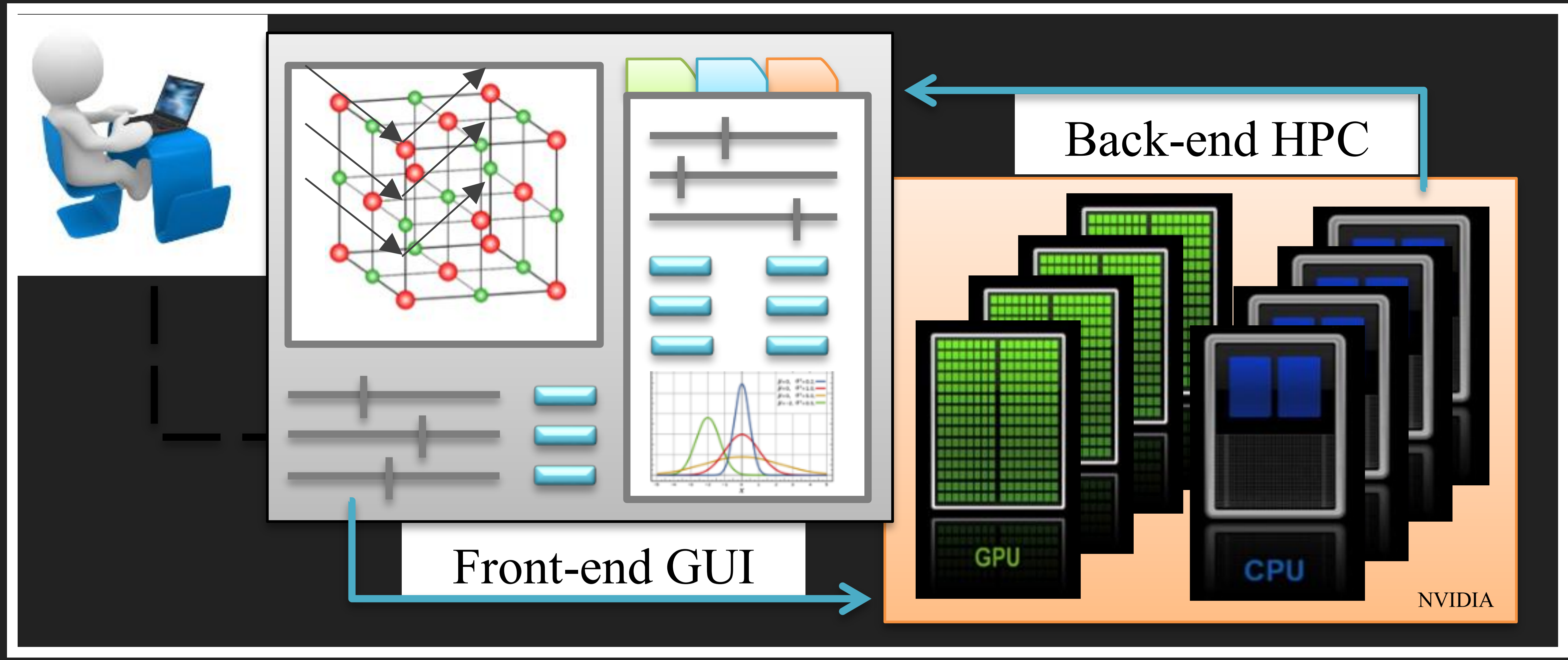
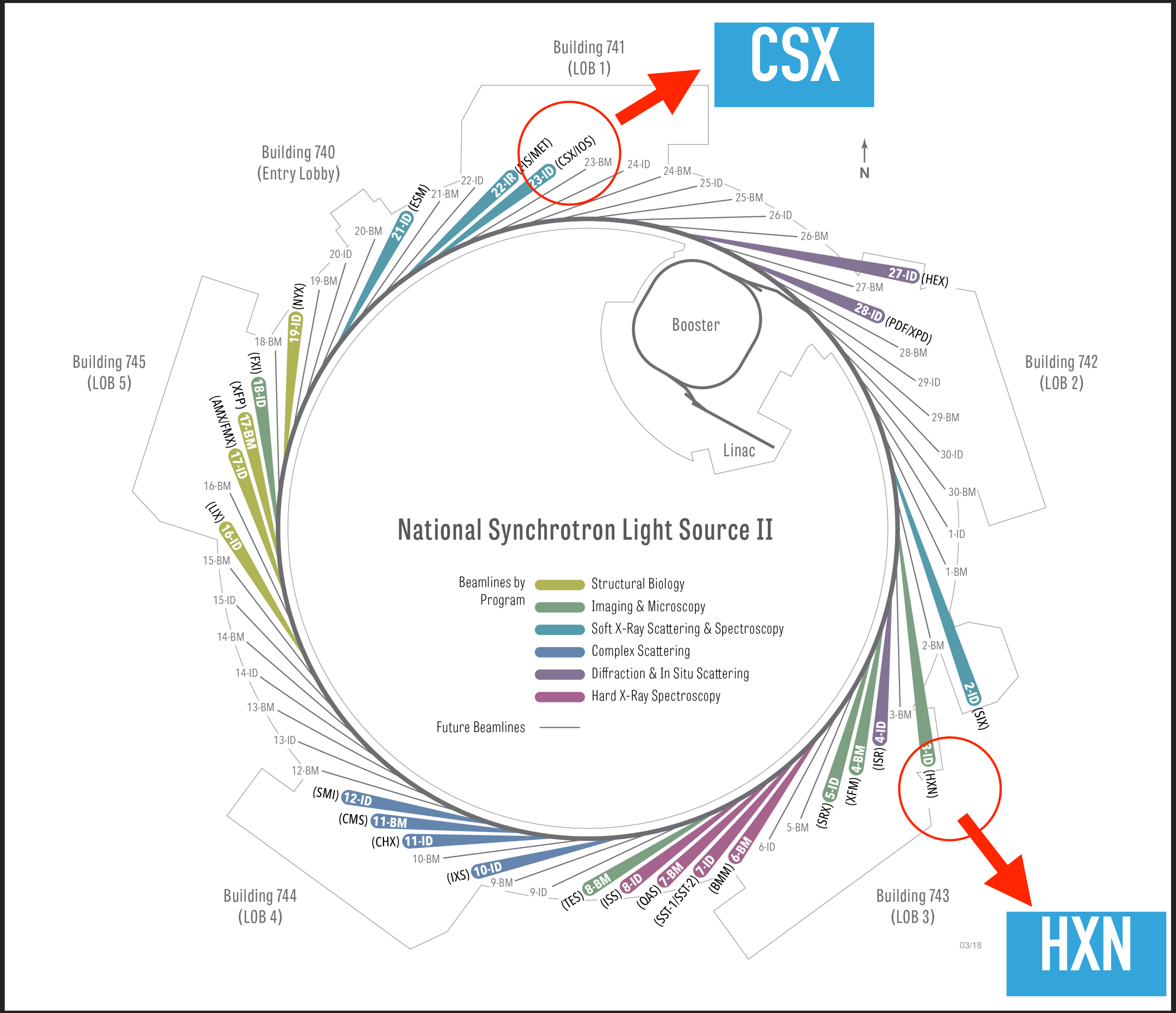



Image Credit: Wei Xu

# NSLS II



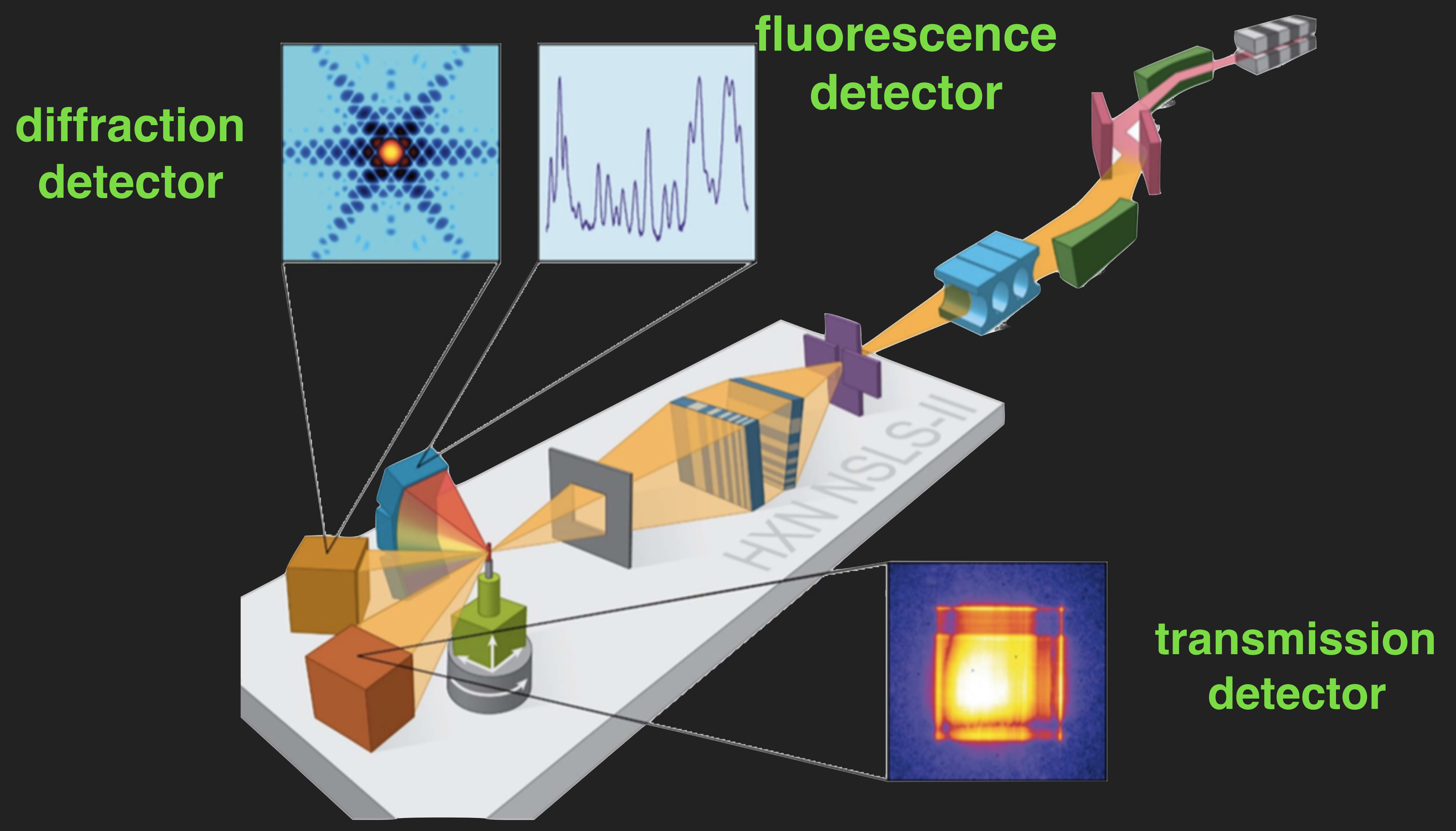
- NSLS II - National Synchrotron Light Source II (there was an NSLS at BNL - now CSI)
- State-of-the-art, medium-energy (3-billion-electron-volt, or GeV) electron storage ring that produces x-rays up to 10,000 times brighter than the NSLS
- First light: 2014
- 28 beam lines in operation; 1 under development
- HXN - Hard X-ray Nanoprobe
- CSX - Coherent Soft X-ray Scattering



# DATA AND COMPUTATION CHALLENGES



- Ptychography reconstruction at HXN
  - Typically O(10,000) - O(100,000) scan images
  - ~200x200 pixels (in floating points) per image
  - Data size of <u>input</u> images:  O(1GB) to O(10 GB)
- Memory requirements for the DM algorithm (including temporary buffers):
  - Single-mode: >4x of input size
  - Multi-mode: >10x of input size
  - **Need multiple GPUs for sufficient memory**
- Difference map iterative algorithm: O(100) iterations
- **Serial Python code:** typically takes hours, and sometimes days (e.g., multislice reconstruction), to complete one ptychography reconstruction.




Yan et al., Nano Futures 2, 011001 (2018)

# NSLS-II PTYCHOGRAPHY SOFTWARE – CURRENT STATUS



- Fully Python-based (numpy + scipy + …) software stack
- ➡ for easy integration with NSLS-II control, data acquisition & analysis environment (databroker, bluesky, ophyd, etc)
- CPU version: **mpi4py** + numpy
- GPU version: **mpi4py + cupy + numba**
- Computationally intensive functions rewritten in CUDA C and/or Numba
- Graphical user interface (GUI) provided
- Already deployed in production at HXN & CSX beamlines

# USING CUPY WITH NUMBA



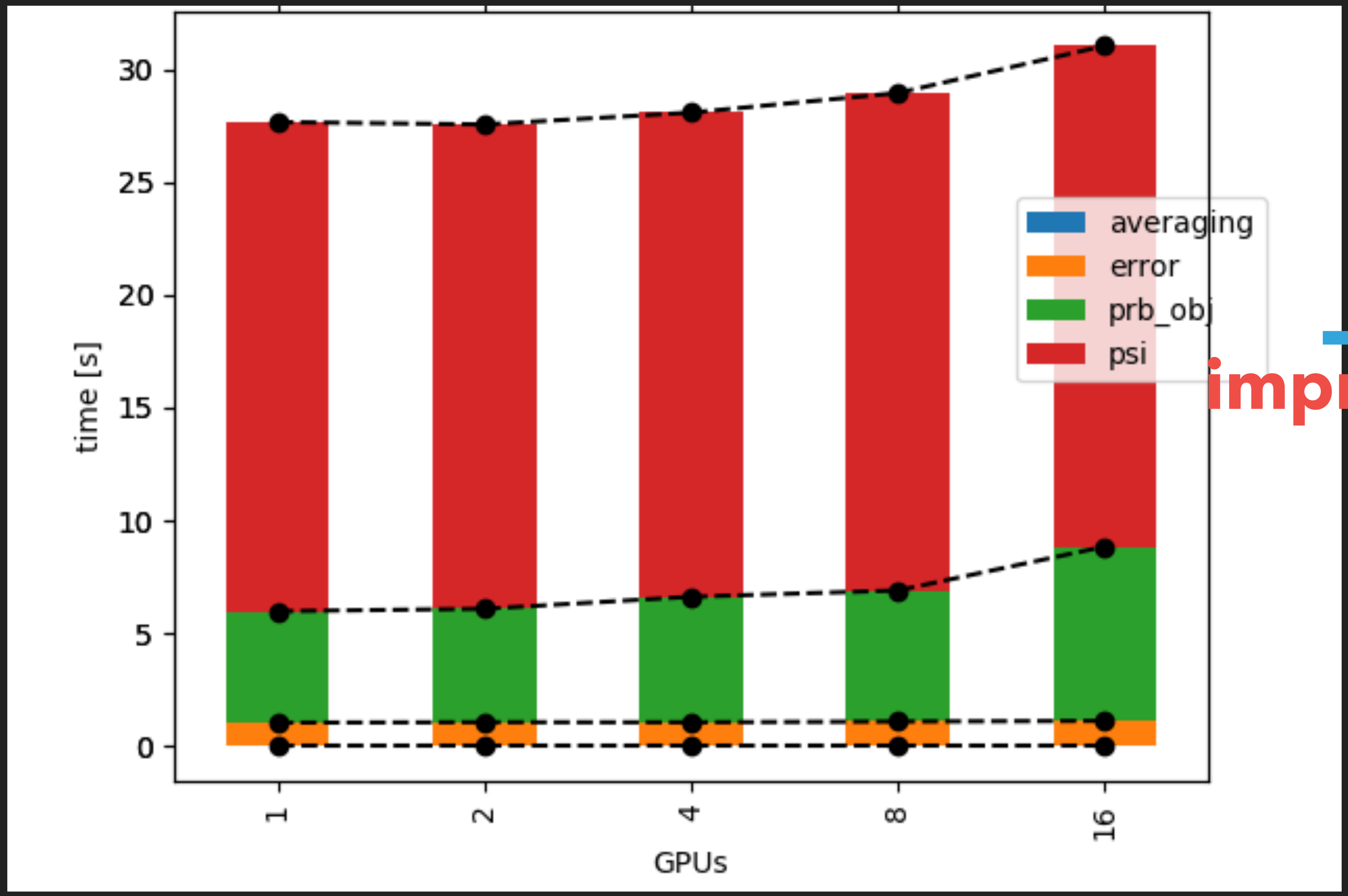
- Use CuPy to create and manage GPU arrays
- Use numba to JIT compile CUDA kernels - no need to write raw CUDA C kernels

```
@cuda.jit()
def accumulate_obj(prb_norm_d, obj_upd_d, prb_sqr_d, prb_conj_d, product_d,
point_info_l, start, batch):
    x, y, z = cuda.grid(3)
    x_max = int32(product_d.shape[-2])
    y_max = int32(product_d.shape[-1])
    if x < x_max and y < y_max and z < batch:
        x_start = point_info_l[start+z, 0]
        y_start = point_info_l[start+z, 2]
        temp = prb_conj_d[x, y] * product_d[start+z, 0, 0, x, y]
        cuda.atomic.add(prb_norm_d, (x_start+x, y_start+y), prb_sqr_d[x, y])
        cuda.atomic.add(obj_upd_d.real, (x_start+x, y_start+y), temp.real)
        cuda.atomic.add(obj_upd_d.imag, (x_start+x, y_start+y), temp.imag)
```

# RESULTS: WEAK SCALING*

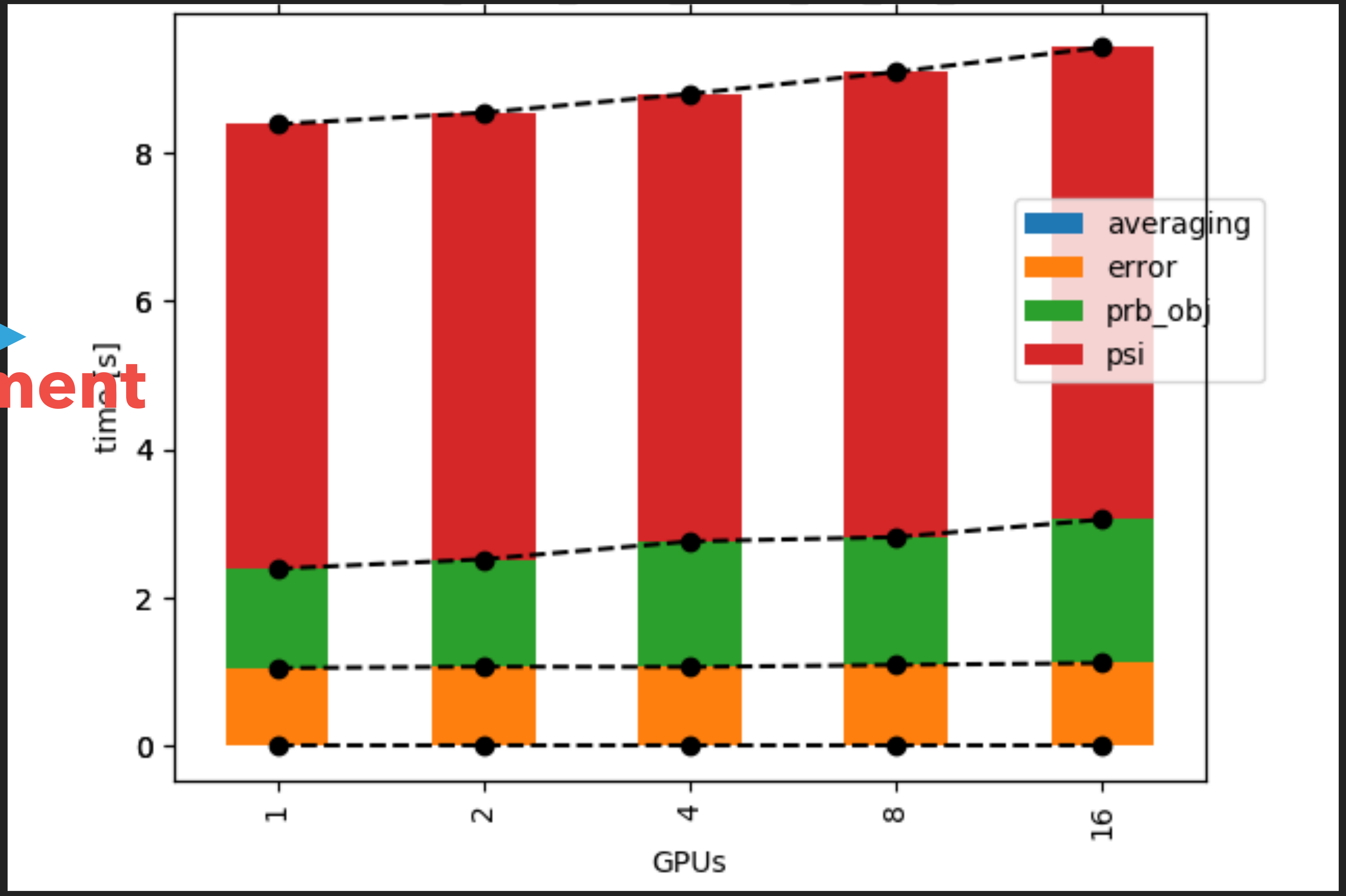


pure CuPy implementation




3X improvement

CuPy arrays + Numba kernels




Pure CuPy is suitable for quick prototyping
performance is reasonable but still much slower than CUDA/numba

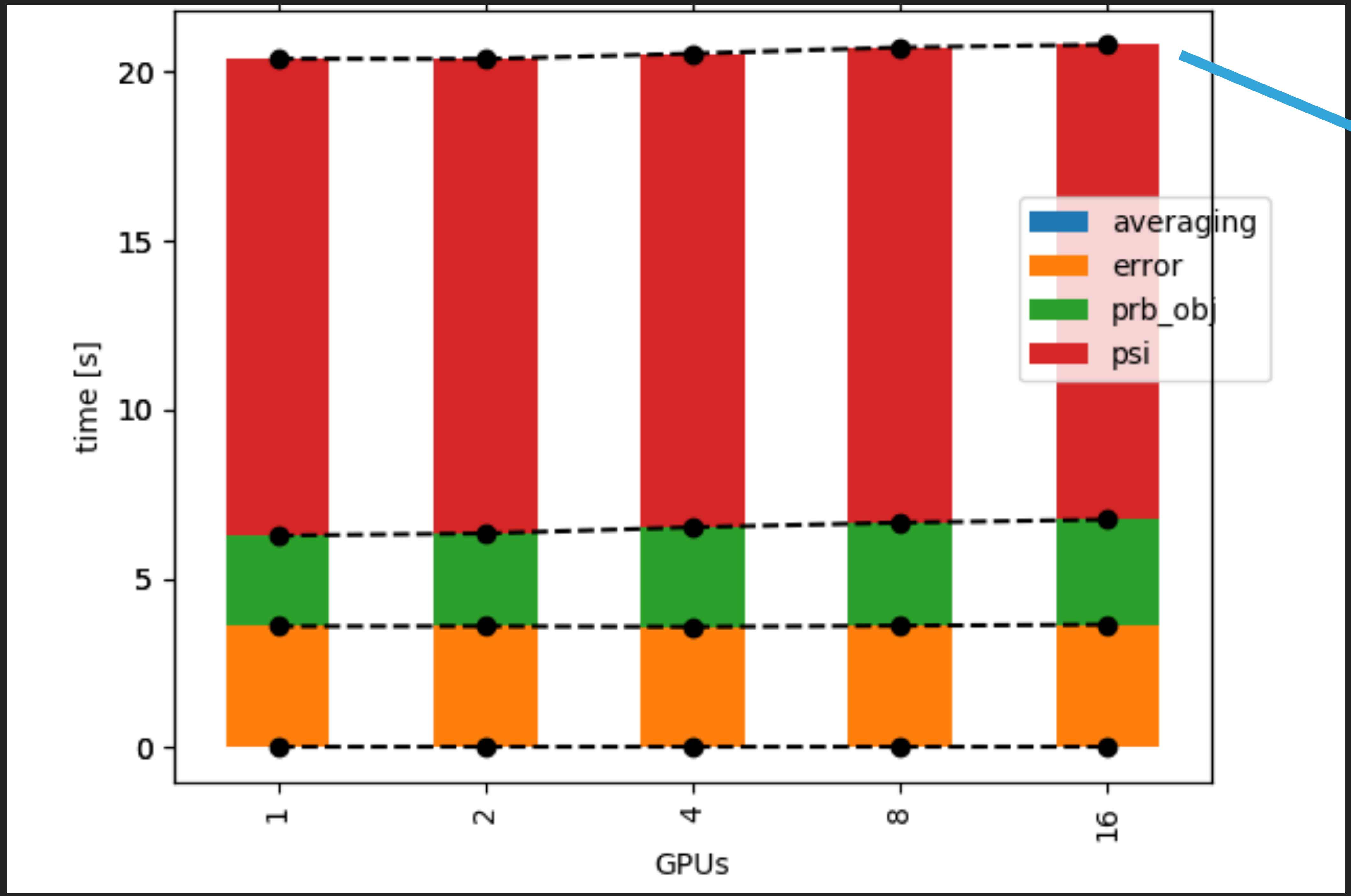
* tested on single DGX-2 with single precision + **no mode** + CuPy v6.1.0 + Open MPI 4.0.1 + NCCL v2.4.2-1
* Test data size: 5000 images per GPU (each image 200x200 pixels)

BROOKHAVEN NATIONAL LABORATORY

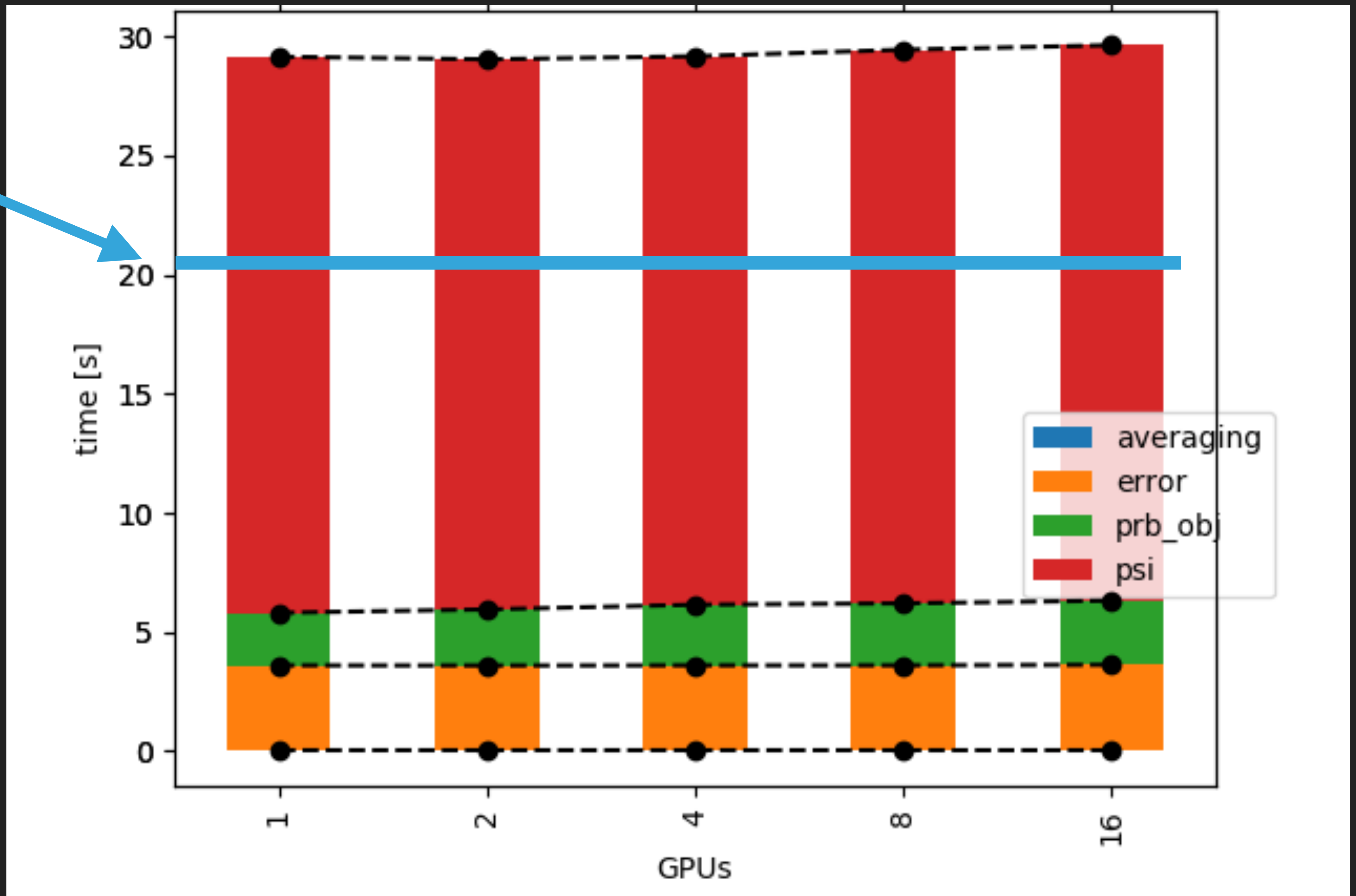
# RESULTS: WEAK SCALING



## CuPy arrays + CUDA kernels




## CuPy arrays + Numba kernels




CuPy + Numba is enough for further performance boost
(~50% slower than CUDA C)

* tested on single DGX-2 with single precision + **5 modes** + CuPy v6.1.0 + Open MPI 4.0.1 + NCCL v2.4.2-1
* Test data size: 5000 images per GPU (each image 200x200 pixels)

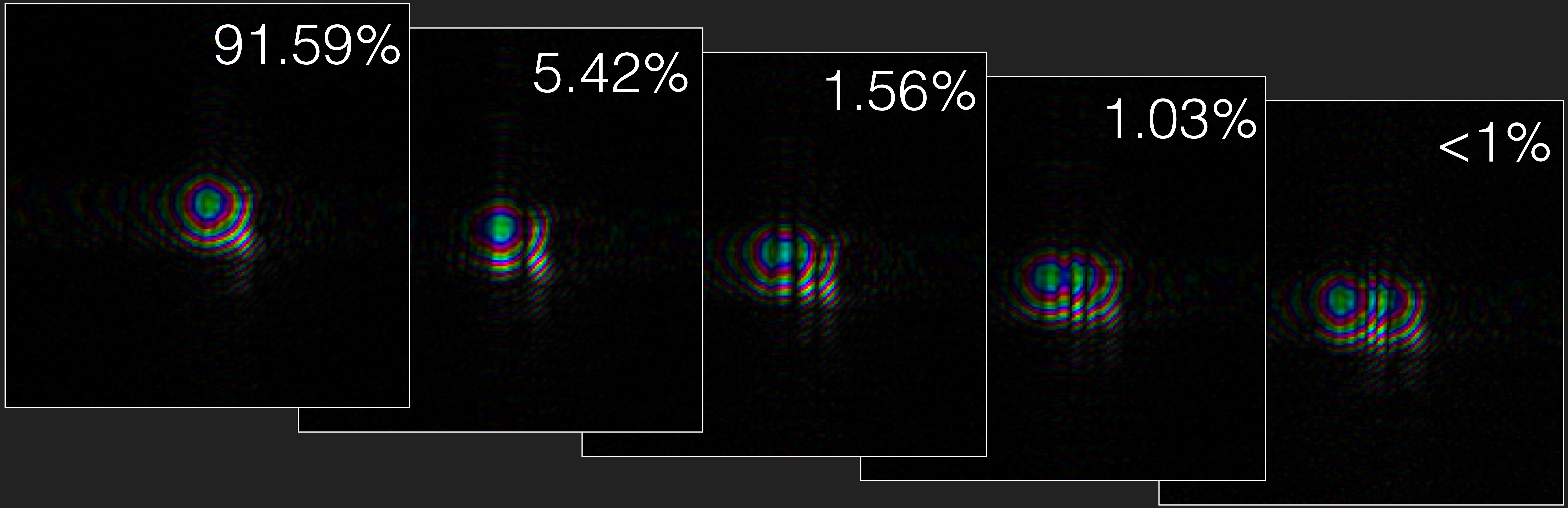
BROOKHAVEN NATIONAL LABORATORY

# RESULTS: GALLERY

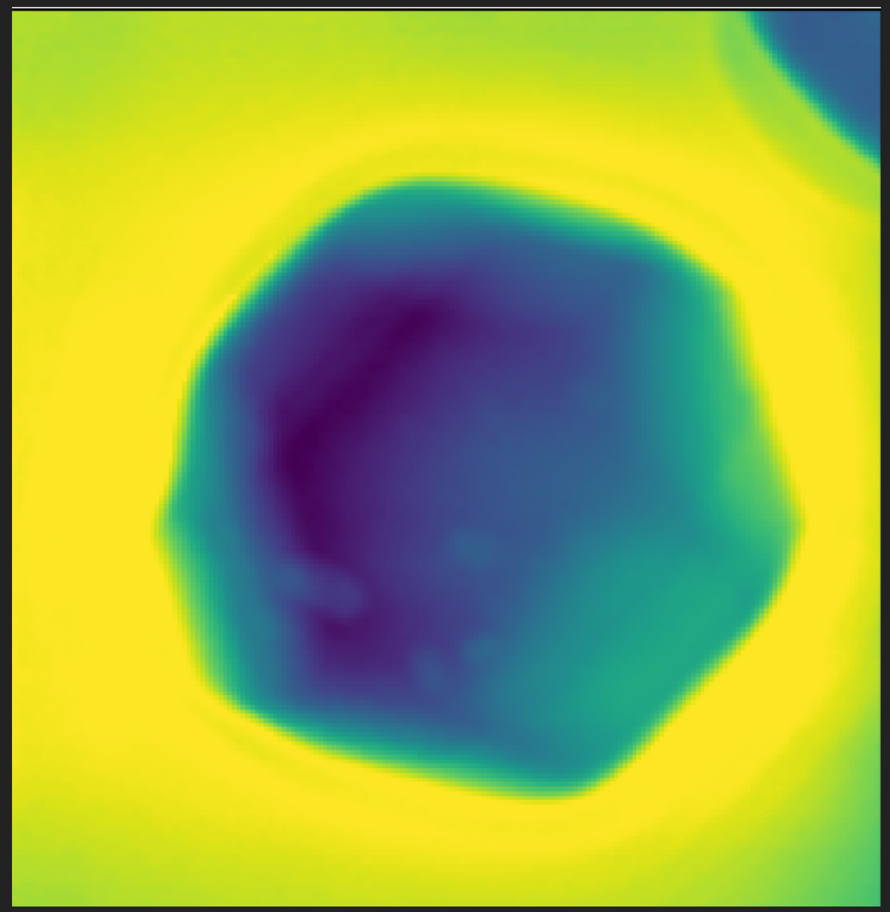


showcase: gold nano-crystal with multi-mode

Test machine: xf03id-srv5@HXN, Intel Xeon CPU E5-2630 v4 @2.20GHz, 256GB RAM, 4 NVIDIA Tesla V100 GPUs. 50 iterations used.



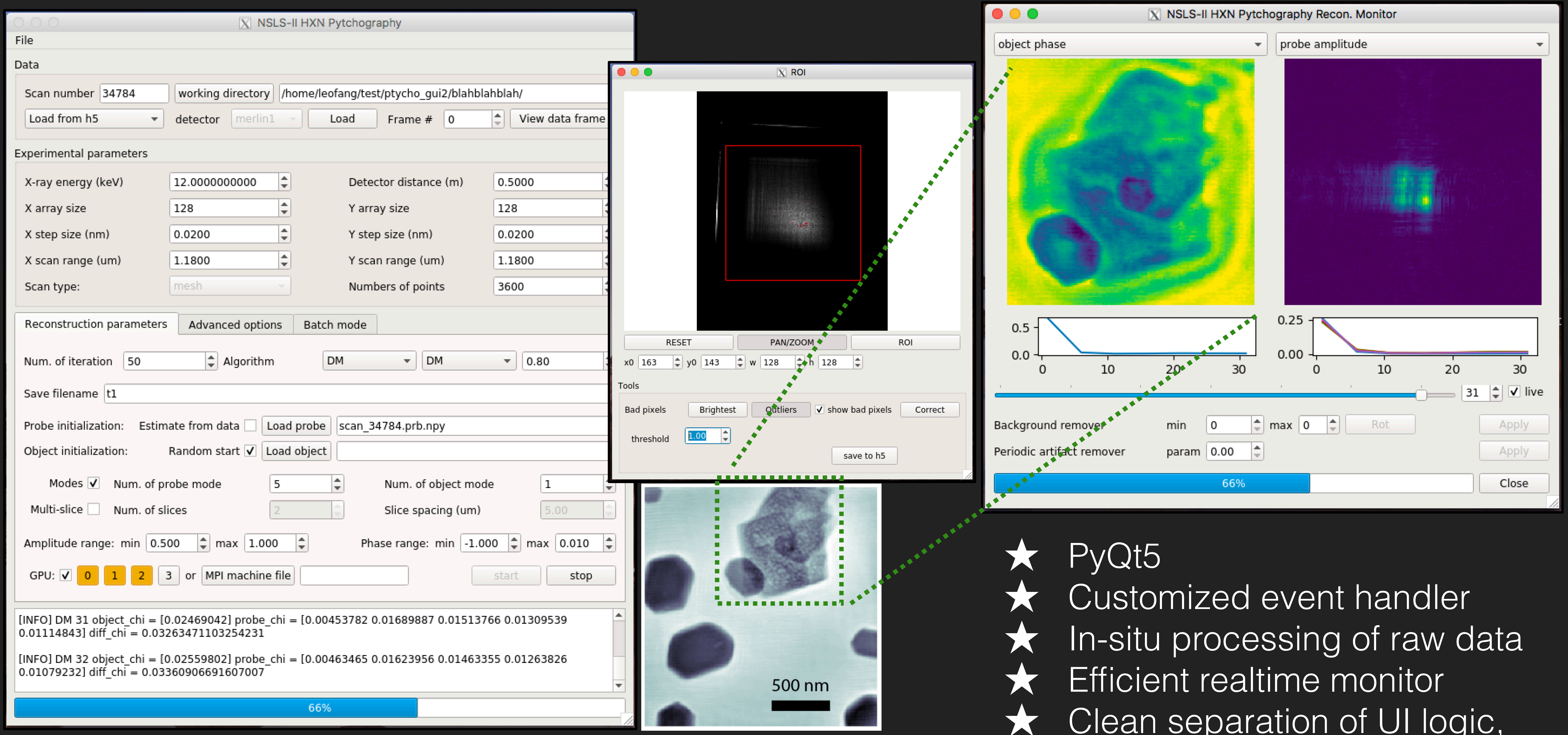

obj. phase



Serial CPU code: 8.8 hr —**>1000x speedup!**→ 4 V100 GPUs: 25.69s

# GRAPHICAL USER INTERFACE






- ★ PyQt5
- ★ Customized event handler
- ★ In-situ processing of raw data
- ★ Efficient realtime monitor
- ★ Clean separation of UI logic, implementation & computation

# HPC for LHC: Accelerating ATLAS Fast Calorimeter Simulations on GPUs








**Zhihua Dong**
**BNL**

Tadej Novak
Jozef Stefan Institute

Kwangmin Yu
BNL

Ahmed Hasib
U. of Edinburgh

Charles Leggett
LBNL

Doug  Benjamin
ANL

Heather Gray
LBNL

Meifeng Lin
BNL

Work supported by DOE HEP via HEP Center for Computational Excellence (CCE)

BROOKHAVEN
NATIONAL LABORATORY

# HL-LHC: SOFTWARE AND COMPUTE CHALLENGES



- Upgrade planned for **High-Luminosity (HL) LHC** in 2026
  - ~10x luminosity of the original LHC design value
  - ~5x increase in event size
  - ~10x increase in event rate

→ **50x data**

- Currently none of ATLAS production software uses compute accelerators.
- "Business as usual" may not be able to meet the compute demands of HL-LHC.
- Need to be able to utilize HPC systems as well as traditional HTC/cloud
- Current and future HPC systems increasingly feature (different kinds of) compute accelerators
- **Portability across different architectures is essential!**

TOP500 The List.

PRESENTED BY

| | SYSTEM | SPECS | SITE |
|---|---|---|---|
| 1 | Summit | IBM POWER9 (22C, 3.07GHz), NVIDIA Volta GV100 (80C), Dual-Rail Mellanox EDR Infiniband | DOE/SC/ORNL |
| 2 | Sierra | IBM POWER9 (22C, 3.1GHz), NVIDIA Tesla V100 (80C), Dual-Rail Mellanox EDR Infiniband | DOE/NNSA/LLNL |
| 3 | Sunway TaihuLight | Shenwei SW26010 (260C, 1.45 GHz) Custom Interconnect | NSCC in Wuxi |
| 4 | Tianhe-2A (Milkyway-2A) | Intel Ivy Bridge (12C, 2.2 GHz) & TH Express-2, Matrix-2000 | NSCC Guangzhou |
| 5 | Frontera | Dell C6420, Xeon Platinum 8280 28C 2.7GHz, Mellanox InfiniBand HDR | TACC/U of Texas |

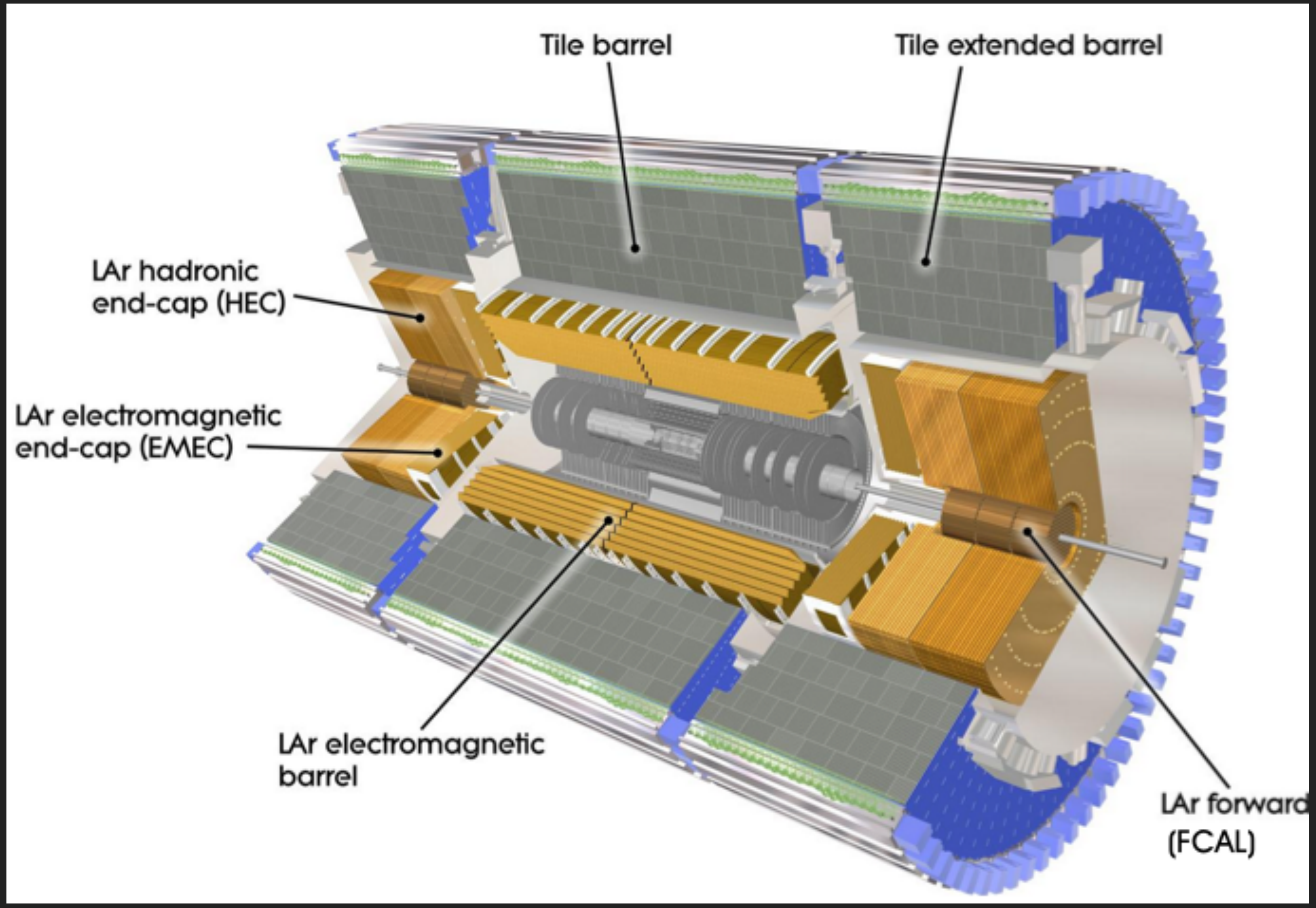
www.top500.org


Upcoming US exascale systems: Auroa (ALCF) and Frontier (OLCF)

# ATLAS FAST CALORIMETER SIMULATION



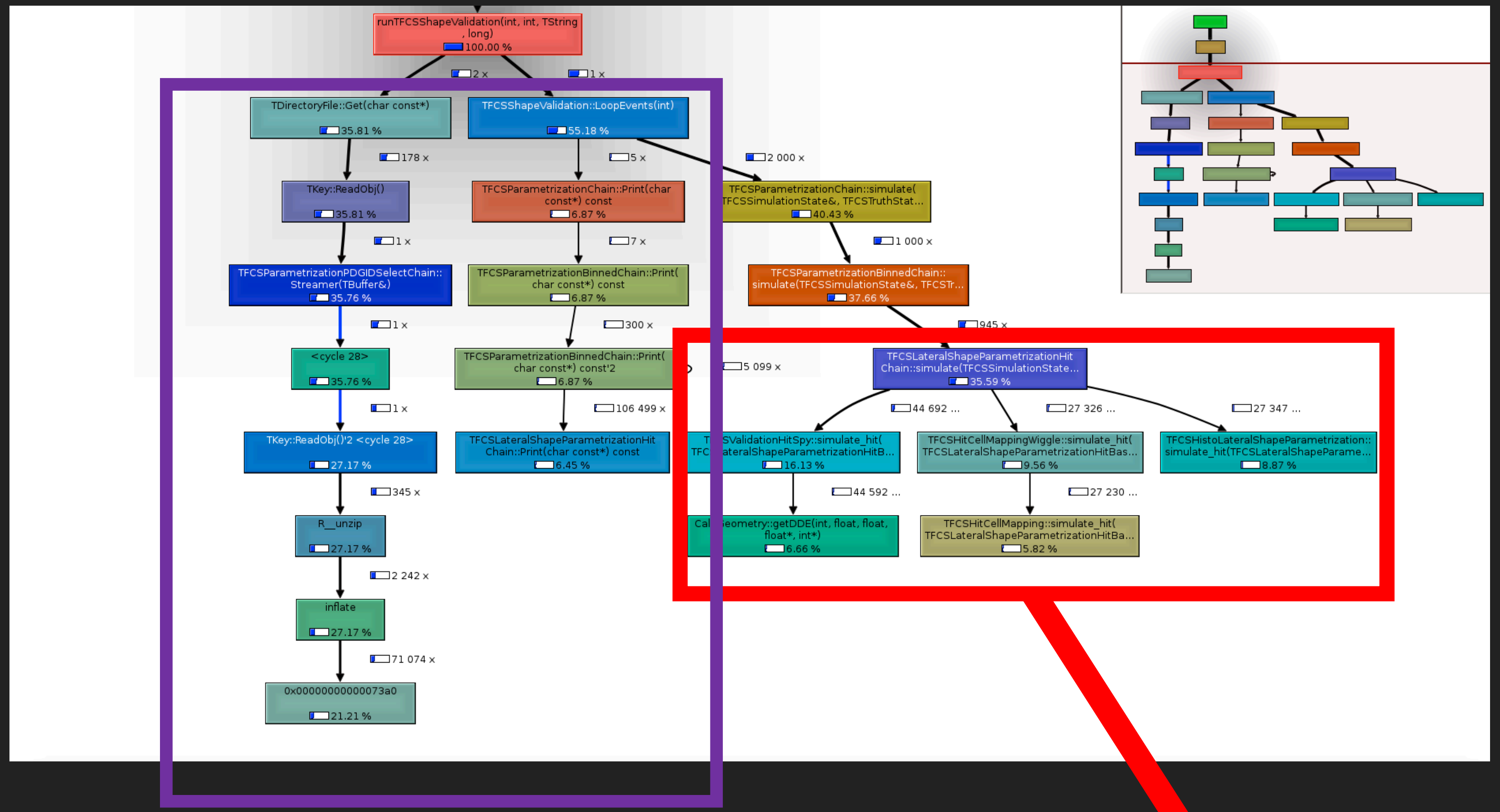
- Calorimeter simulation measures the energy depositions of O(1000) particles after each collision.
- Full detailed simulation uses Geant4, which is very slow
- Fast calorimeter simulation uses parametrization of the calorimeter: less accurate but much faster than Geant4 [T. Yamanaka (ATLAS) 2011]
- FastCaloSim (FCS): a relatively self-contained code for fast ATLAS calorimeter simulation
  - Good candidate for proof-of-concept GPU/ portability studies




Credit: ATLAS

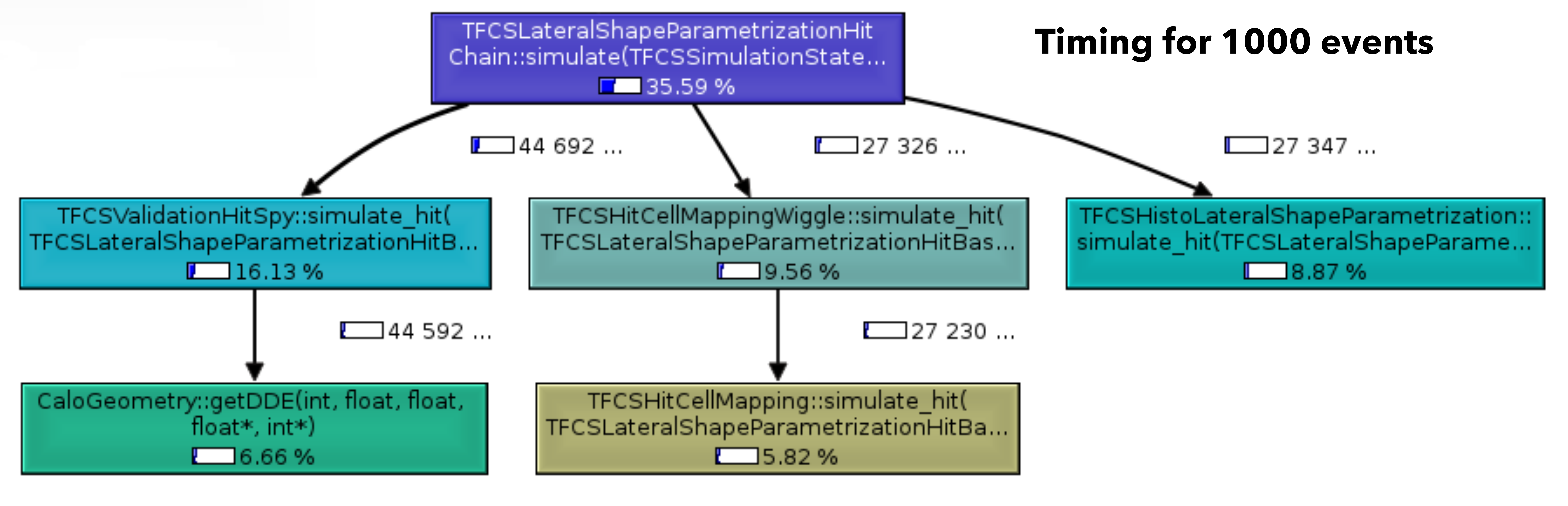
# PERFORMANCE PROFILING





I/O routines

- ➢ **TFCSLateralShapeParametrizationHitChain::simulate()** is the **most significant** routine except I/O (~30%).
- ➢ **TFCSLateralShapeParametrizationHitChain::simulate()** The running time **scales with the number of events.**
- ➢ **TFCSLateralShapeParametrizationHitChain::simulate()** is our target to parallelize/port to GPUs.




**Timing for 1000 events**

# GPU PORTING



- Initial strategy: CUDA
  - to identify feasibility and challenges with GPU porting
- Data structure modification from CPU to GPU:
  - Implemented new GPU CaloGeomory structure and supporting Classes
  - Simpler, no ROOT Dependence
  - CaloGeometry data can be loaded once and be reused: ~25MB
- Multi-stage CUDA kernels to generate histograms
  - Blockwise atomic update with shared memory
  - Followed by reduction across all blocks
- To get # of hits in the calo cells
  - only ~200 cells get hit out of 20,000 cells - trial run to narrow down the hit cells
  - Reduces memory requirement, and load imbalance

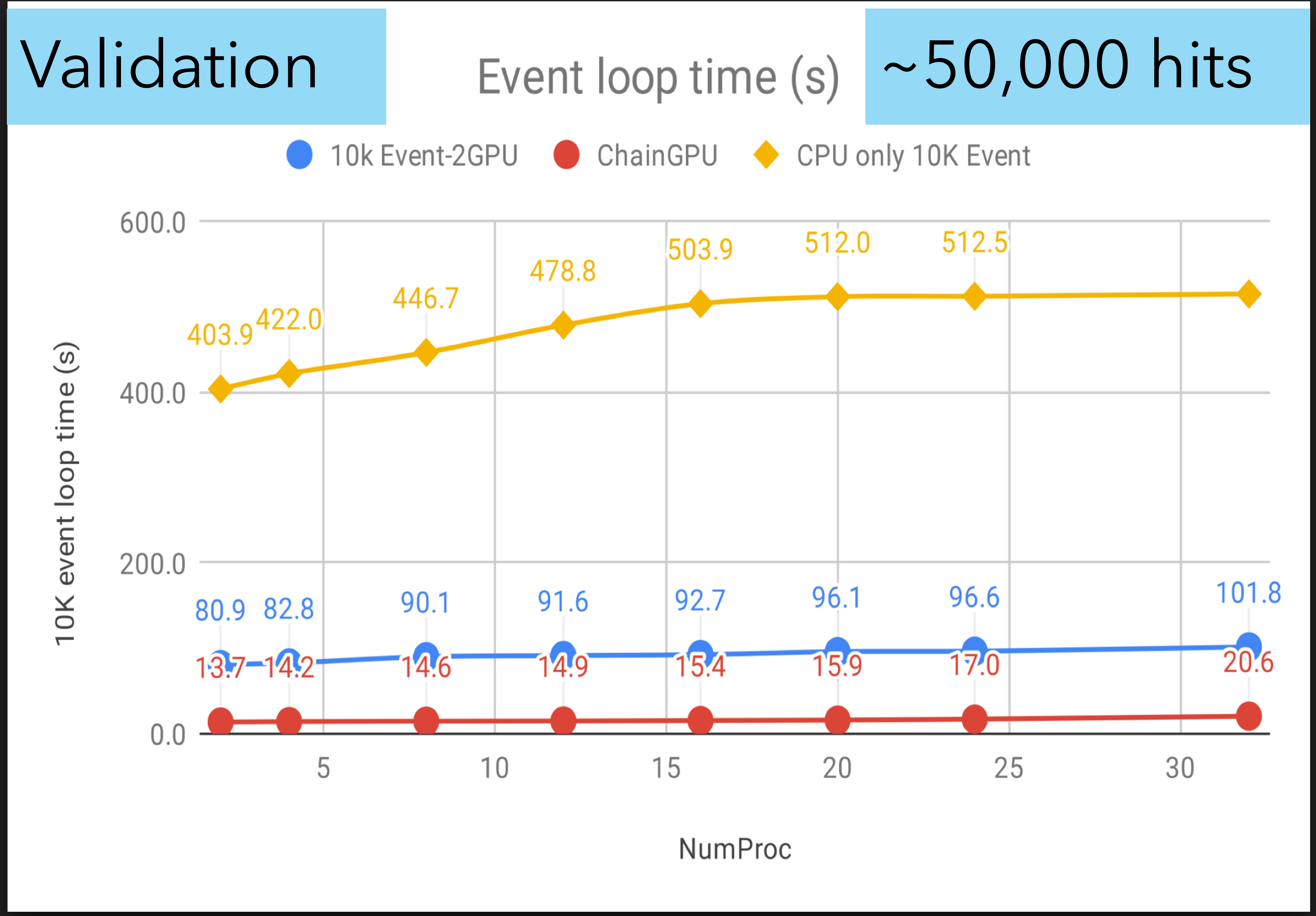
# CPU VS GPU PERFORMANCE COMPARISON



- Validation against GEANT4 most time consuming (~50K hits)
- CPU: "embarrassingly parallel" - different processes simulate different events
- GPU: Use CUDA-MPS to share 2 P100 GPUs on BNL Institutional Cluster*
- ~5X gain with 50K hits compared to CPU only runs (32 parallel processes).
- Actual production runs have fewer hits - less compute
  - Less performance gain: 2-3X vs. CPU

* CPU: Intel Xeon "Broadwell" 32 cores per node
* GPU: 2 NVIDIA P100 per node

Validation ~50,000 hits



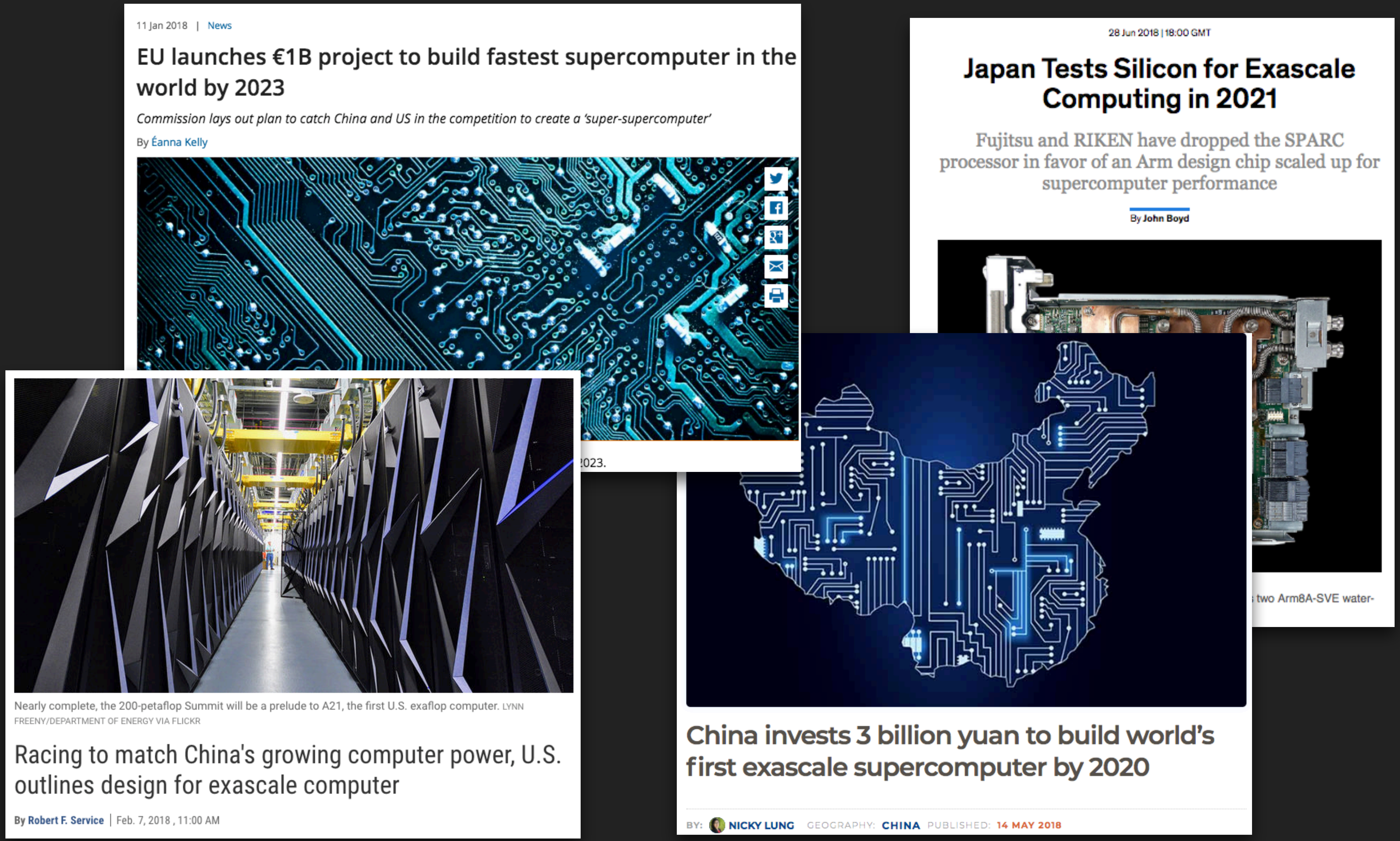

Simulations ~5,000 hits

| #MPI Processes | Particle | Energy | Min Eta | CPU (s) /10K event / process | GPU (s) / 10K event / process |
|---|---|---|---|---|---|
| 1 | Electron | 65536 | 2.2 | **18.8** | **6.0** |
| 32 | Electron | 65536 | 2.2 | **24.0** | **7.1** |

# EXASCALE COMPUTING

# THE RACE TO EXASCALE

- China, EU, Japan and US are all developing exascale supercomputers.



# DIVERSE EXASCALE ARCHITECTURES IN US ALONE



## Relevant US DOE Pre-Exascale and Exascale Systems for ECP

**Pre-Exascale Systems**

**Future Exascale Systems**

| 2012 | 2016 | 2018 | 2020 | 2021–2023 |
|---|---|---|---|---|
| TITAN – ORNL Cray/AMD/NVIDIA | CORI – LBNL Cray/Intel | SUMMIT – ORNL IBM/NVIDIA | PERLMUTTER – LBNL Cray/AMD/NVIDIA | FRONTIER – ORNL Cray/AMD |
| MIRA – ANL IBM BG/Q | THETA – ANL Intel/Cray | | | Aurora – ANL Intel/Cray |
| SEQUOIA – LLNL IBM BG/Q | TRINITY – LANL/SNL Cray/Intel | SIERRA – LLNL IBM/NVIDIA | | CROSSROADS – LANL/SNL TBD; EL CAPITAN – LLNL Cray |

ECP EXASCALE COMPUTING PROJECT

Doug Kothe @ 2019 NSF Blueprint workshop

# TWO MAJOR PARALLEL PROCESSING PARADIGMS (SINGLE-NODE)



- **SIMD - Single Instruction Multiple Data**
  - Intel Xeon Phi (AVX512): Cori/NERSC, Theta/ALCF
  - Intel Xeon "Skylake" (AVX512): Frontera/TACC
  - New! ARM SVE (Scalable Vector Extensions), supporting 128-bit to 2048-bit vector units: Post-K/Japan, new system at SBU
- **SIMT - Single Instruction Multiple Threads**
  - GPGPUs - NVIDIA, AMD, Intel New!
- Can we have the same data format/layout/programming model for both?

# PORTABILITY CONSIDERATIONS



**Given the diversity of current and upcoming HPC architectures, we may need to design our software with following considerations:**

- Performance Portability
  - How much tradeoff do you want to make between performance and portability?
  - Is it possible to design your software to be portable and at the same time reasonably performant?
- Programming Models
  - What programming models do you want to use c.f. performance portability?
  - OpenMP, OpenACC, OpenCL, CUDA, HIP, SyCL, OneAPI, Kokkos, etc.
- Programming Languages
  - Parallelism has increasing become part of the language itself, e.g. pSTL in C++.
- Data Layout
  - Is there a “one-size-fits-all” data layout for the diverse architectures?

# CSI ECP PROJECTS



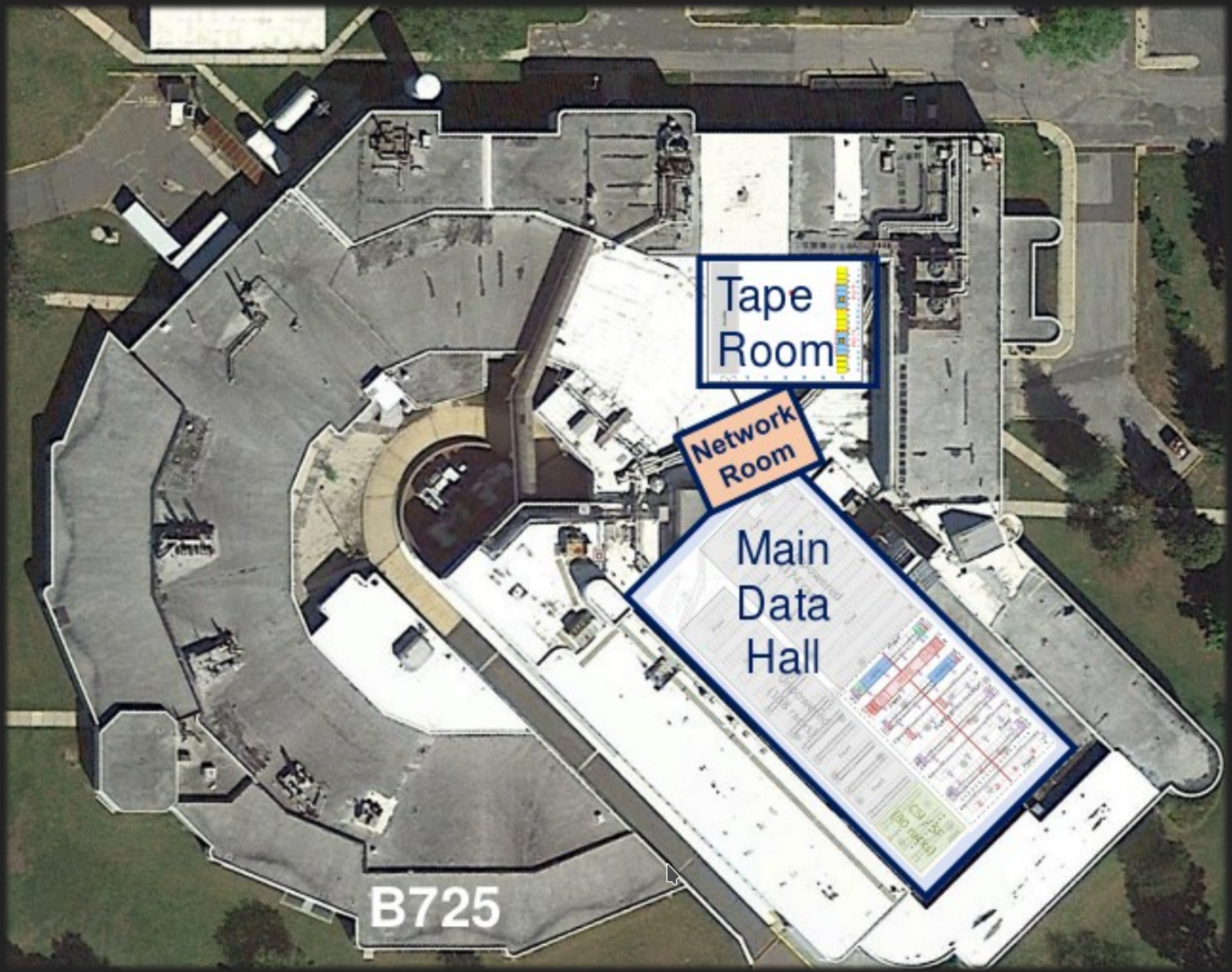
- Application Development
  - Lattice QCD - algorithms, performance portability, workflows
  - NWChemEX - newly-designed C++-based library (from Fortran-based NWChem)
- Software Technologies
  - SOLLVE (Scaling OpenMP LLVM Compiler towards Exascale) - OpenMP standard, LLVM compiler infrastructure
- Codesign Centers
  - CODAR - Center of Data Analysis and Reduction
  - ExaLearn - Machine Learning software for Exascale applications

# SIDE NOTES

# NEW DATA CENTER CONSTRUCTION UNDER WAY



- Repurposed NSLS Light Source building
- "Tier III" Class data center*
  - Redundant infrastructure
  - Concurrently maintainable
  - Completely self sufficient in emergencies
- New data center occupancy timeline
  - ATLAS areas ready before CY2021 - to coincide with LHC Run 3 start
  - Other areas become ready for occupancy throughout CY2021






* using the "Tier" classification defined by the Uptime Institute

Slide Credit: Shigeki Masawa, Imran Latif, Alexandr Zaytsev

# TRAINING EVENTS



- CSI regularly holds hands-on training events
- Hands-on training events (hackathons) give scientists access to expert guidance on modern HPC architectures and programming tools.
- Great way to jumpstart incorporating a new programming tool/model in your code
- Planned this year:
  - GPU Hackathon, August 17-21, 2020
  - OpenMP Hackathon, dates TBD
  - ML/AI Tutorials, dates TBD