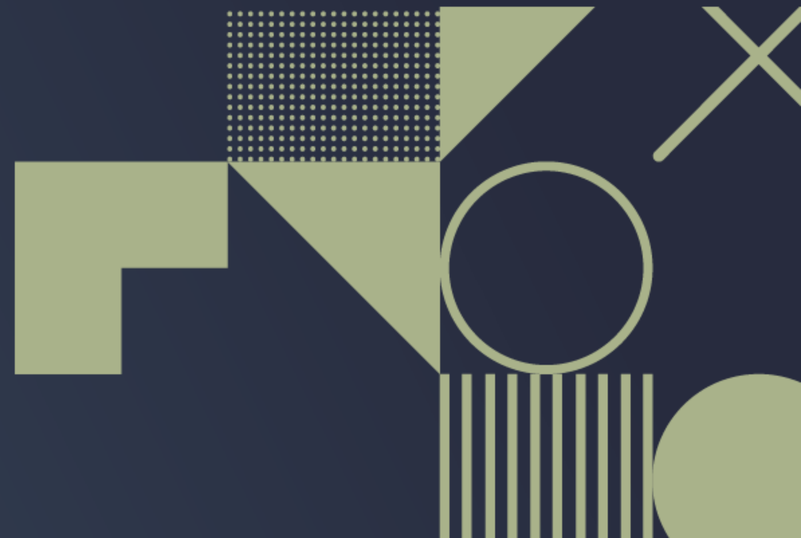
Scaling and Acceleration of Three-dimensional Structure Determination for Single-Particle Imaging Experiments with SpiniFEL

Iris Hsing-Yin Chang[1], Elliott Slaughter[1], Seema Mirchandaney[1],
Jeffrey Donatelli[2], Chun Hong Yoon[1]
[1]SLAC National Accelerator Laboratory, [2]Lawrence Berkeley National Laboratory

The 4th Annual Parallel Applications Workshop,
Alternatives To MPI+X



Outline

- Motivation and Challenges for Data Analytics at the Linac Coherent Light Source (LCLS)
- Background — Single-Particle Imaging (SPI)
- Mathematical Framework — Multitiered Iterative Phasing (M-TIP) Algorithm
- Acceleration and Parallelization — GPU Offloading + Pygion
- Results and Performance Analysis

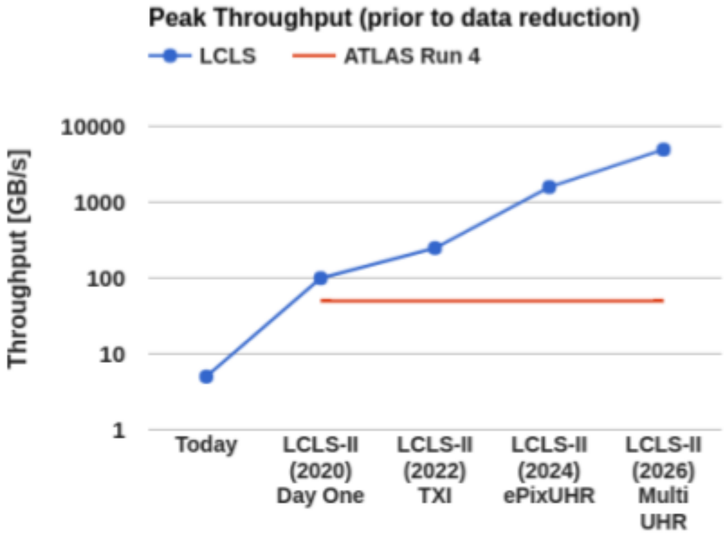
Data Analytics at the Exascale for Free Electron Lasers (ExaFEL)

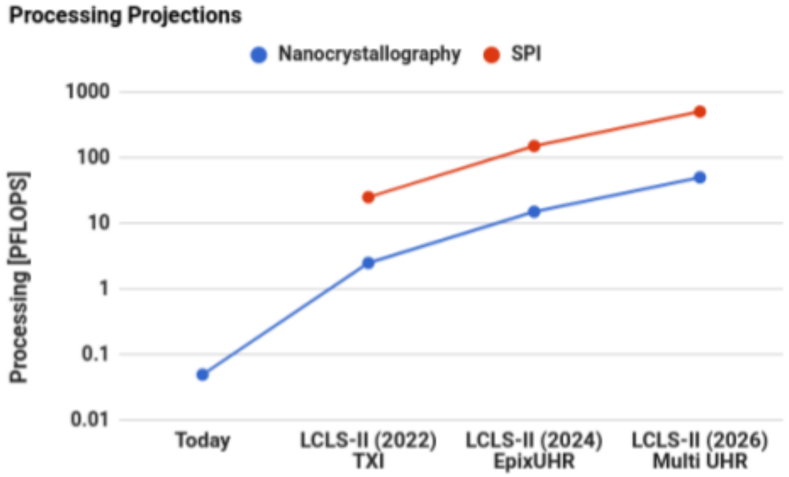
Challenging Characteristics of LCLS Computing

- **High data throughput**
  - High repetition rate (1 MHz) and data throughput (200 GB/s) generated by LCLS-II
- **Fast feedback**
  - Real-time analysis (~minutes or faster) is critical to the user's ability to take informed devision during an LCLS experiment
- **Wide variety of experiments**
  - Speed and flexibility of the **development cycle** is critical

Motivation

- Focus on the scientific questions — lowering the barrier to analysis (serial femtosecond crystallography (SFX), **single-particle imaging (SPI)**, etc)



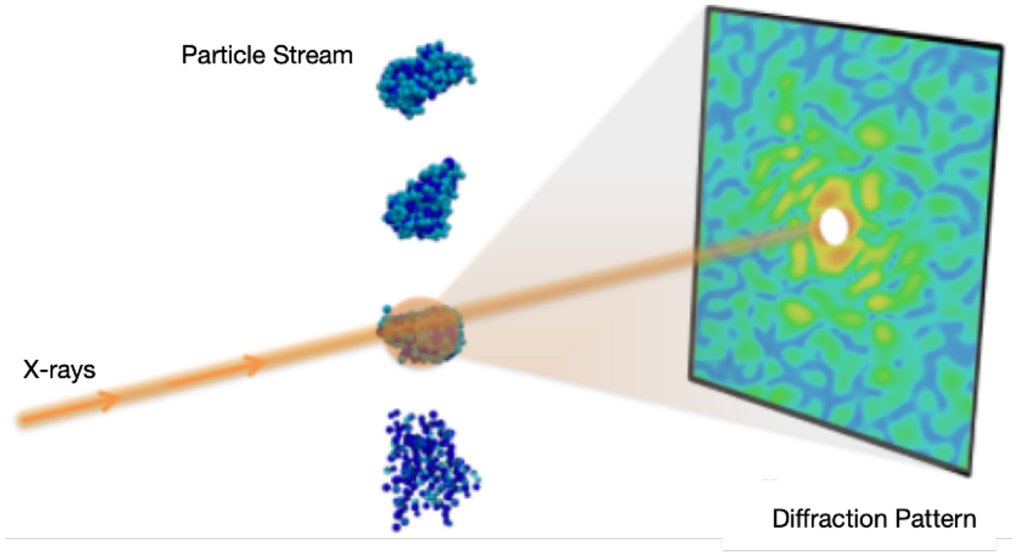


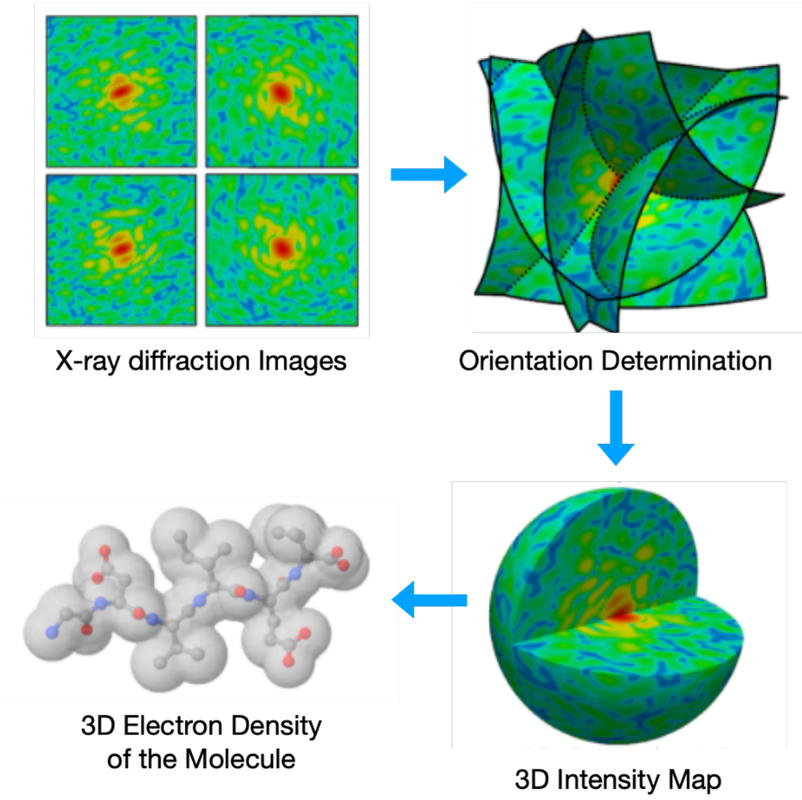
Single-Particle Imaging — A Revolution in X-ray Free Electron Science

Single-Particle Imaging — A Revolution in X-ray Free Electron Science

- **Single-Particle Imaging (SPI)** offers significant advantages over crystallography (e.g. no need to crystallize molecules) or cryo-electron microscopy (cryo-EM) techniques (e.g. ability to make measurements under physiological conditions, access to time domain).

- Ability to scale **SPI** represents a revolution for X-ray free electron lasers.



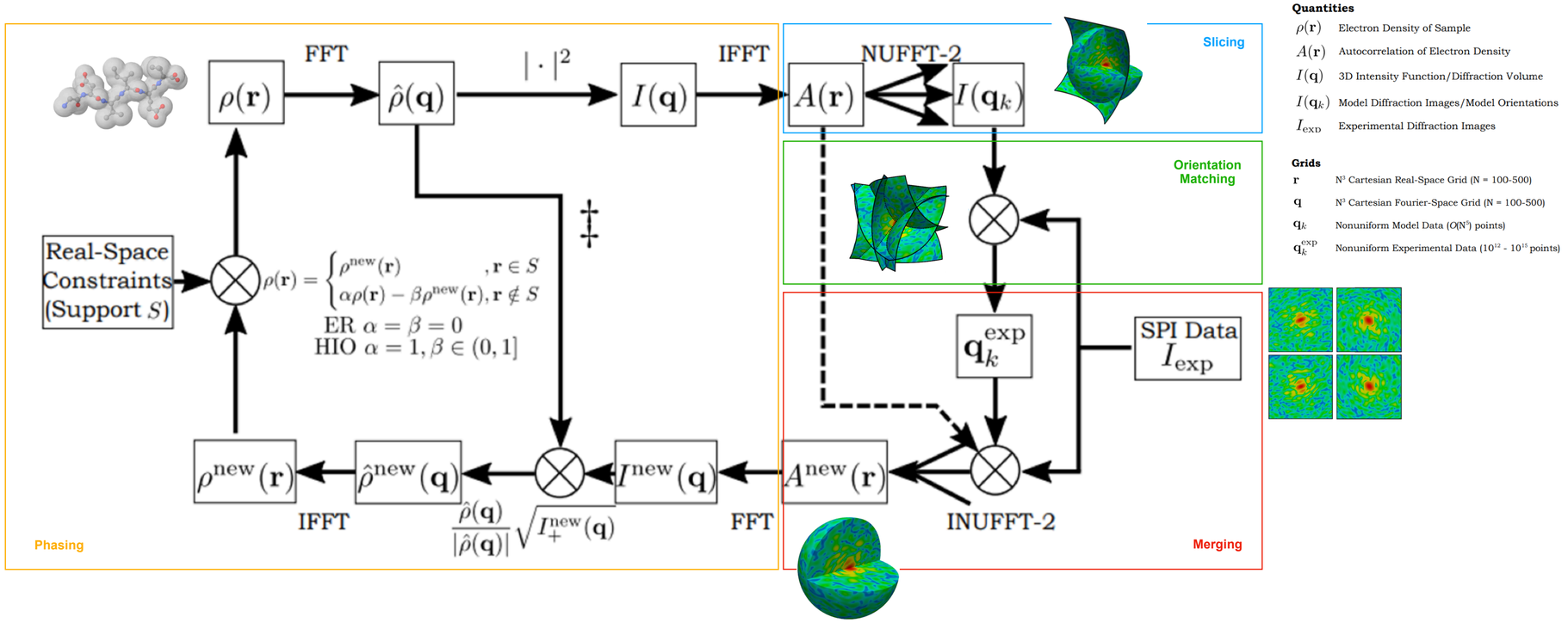


Multitiered Iterative Phasing (MTIP) Cartesian-NUFFT Framework

- A new approach—multitiered iterative phasing (M-TIP) algorithm—is proposed, which **simultaneously determines conformational states, orientations, intensity and phase** from single particle diffraction images.

- **Modular approach** allows modifications to model systematic issues in data and incorporate additional constraints.

- A new Cartesian-nonuniform fast Fourier transform (NUFFT) formulation of M-TIP algorithm is developed, which allows SPI reconstruction to be parallelized.

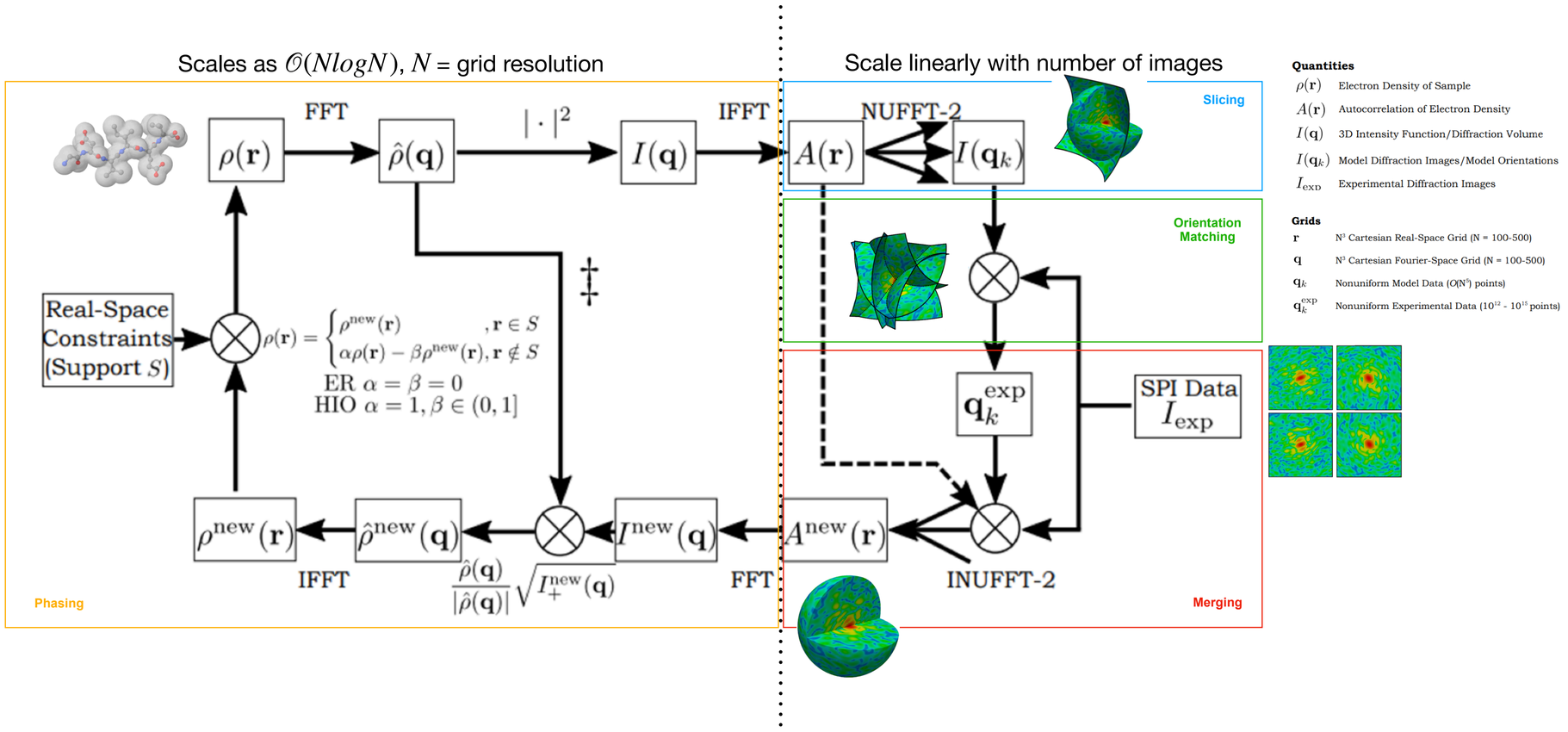


Multitiered Iterative Phasing (MTIP) Cartesian-NUFFT Framework

- Repeat **(1) Merging, (2) Phasing, (3) Slicing,** and **(4) Orientation Matching** until convergence.

- **Slicing + Orientation Matching** is made scalable over multiple nodes by initializing M model images at startup and distributing N data images, scales as $\mathcal{O}(NM) \Longrightarrow$ embarrassingly parallel.

- **Phasing** does not require access to data images, and only requires computing model values on a Cartesian grid, scales as $\mathcal{O}(NlogN)$.



Merging — Nonuniform Fourier Transform Inversion

Problem Formulation

- Linear, discrete ill-posed problems of the form

$\min_x \|Ax - b\|_2 \iff A^*Ax = A^*b$ (normal equations)

where

$A \in C^{n\times m}$ is the NUFFT operator (uniform to non-uniform Fourier transform),

$A^* \in C^{m\times n}$ its adjoint (non-uniform to uniform Fourier transform),

$x \in R^m$ is a vector representing the autocorrelation on a uniform grid,

$b \in R^n$ is a vector of intensity data on a non-uniform grid.

- The incompleteness of the data in combination with the high levels of noise lead to instabilities and nonuniqueness of the solution $\Longrightarrow$ use **Tikhonov regularization** to determine a solution that approximates the exact solution.

- Minimize the perturbation δx

$\min_x \|(Ax - b)\|_2 + \lambda_r\|\delta x\|_2 \iff$ solving $A^*Ax + \lambda_r\delta x = A^*b \iff A^*Ax + \lambda_r(x - x_0) = A^*b$

$\iff A^*Ax + \lambda_r x = A^*b + \lambda_r x_0$ for x,

where

λ_r the tuning parameters for the penalty term, x_0 the initial guess of the autocorrelation.

Regularization Parameter Selection

- Repeat the **conjugate gradient (CG)** linear solve over several ranks, where each rank attempts the solve with a different set of hyper parameters.
- Select the best solution using **L-curve criterion** — optimal trade-off between the size of a regularized solution and its fit to the given data.

Minimizing Reconstruction Times — Exploiting Toeplitz Structure

- NUFFT inversion can be reformatted to exploit the Toeplitz structure of matrices evaluated every iteration, but it requires larger oversampling than what is strictly required by NUFFTs.

NUFFT Method

Nonuniform discrete Fourier transform (NUDFT) $Ax[k] = \sum_m x[m] e^{-2\pi i m \cdot q_k}$

Solve normal equations from Fourier data $b \Longleftrightarrow A^*Ax = A^*b$

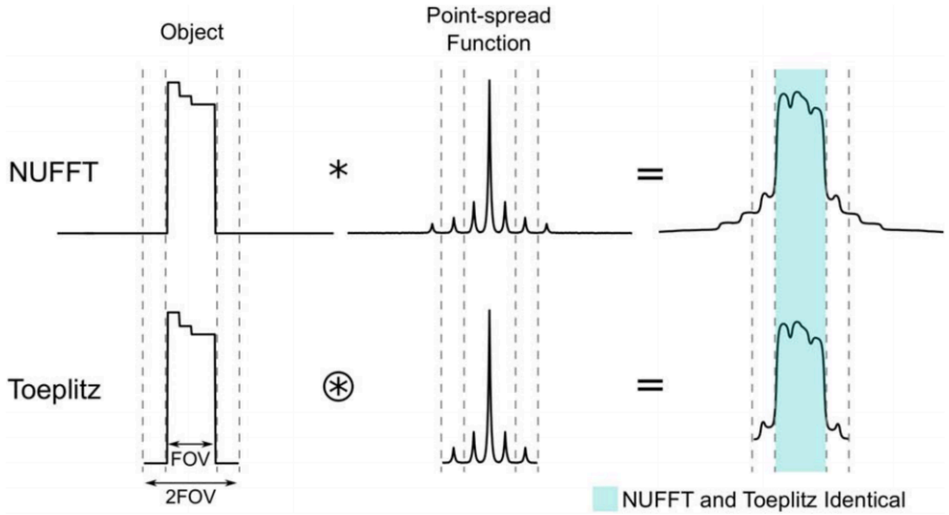
Toeplitz Method

(left hand)

$$A^*Ax[n] = \sum_k \sum_m x[m] e^{-2\pi i m \cdot q_k} e^{2\pi i n \cdot q_k} = \sum_m x[m] \left(\sum_k e^{2\pi i (n-m) \cdot q_k} \right) = \sum_m x[m] Q[n-m] = (x \circledast Q)[n]$$

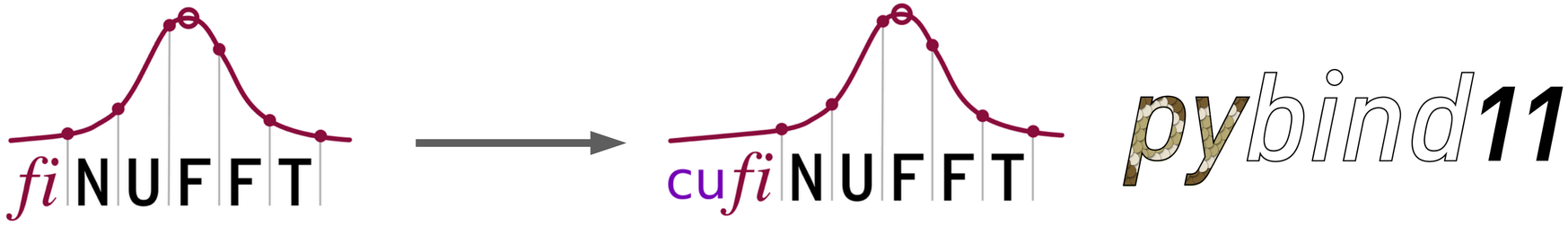
where $Q[n] = \sum_k e^{2\pi i n \cdot q_k}$

(right hand) $(A^*b)[n] = \sum_k b_k e^{2\pi i n \cdot q_k}$

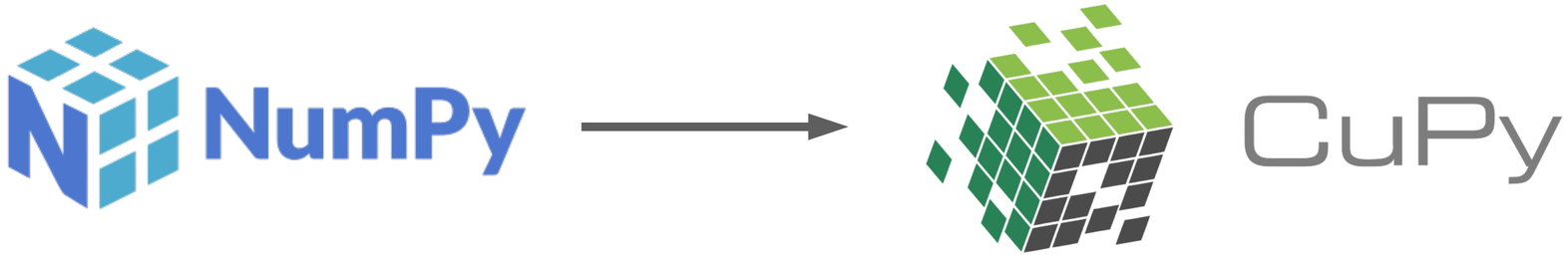


Offloading Compute Intensive Components to GPUs

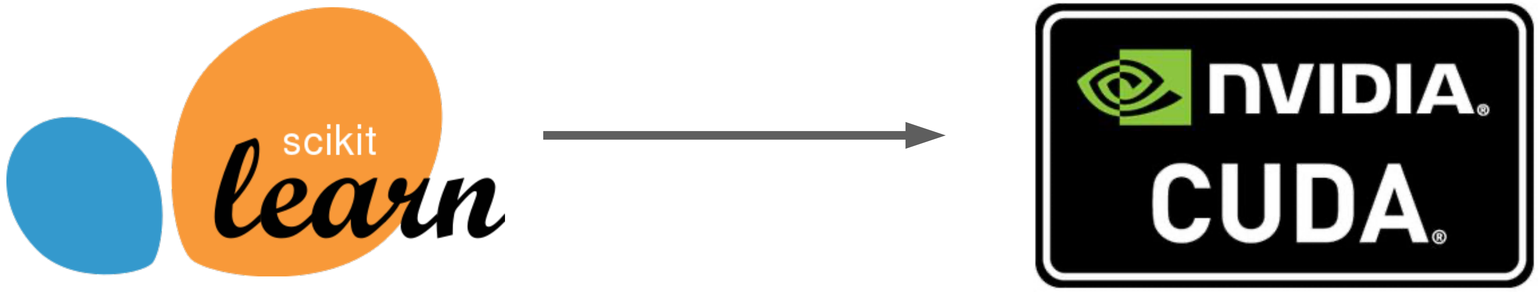
- NUFFTs in **merging** and **slicing** operations are offloaded to GPUs using cuFINUFF — an implementation of the 2- and 3-dimensional NUFFT of types 1 and 2, in single and double precision, based on the CPU code FINUFFT.



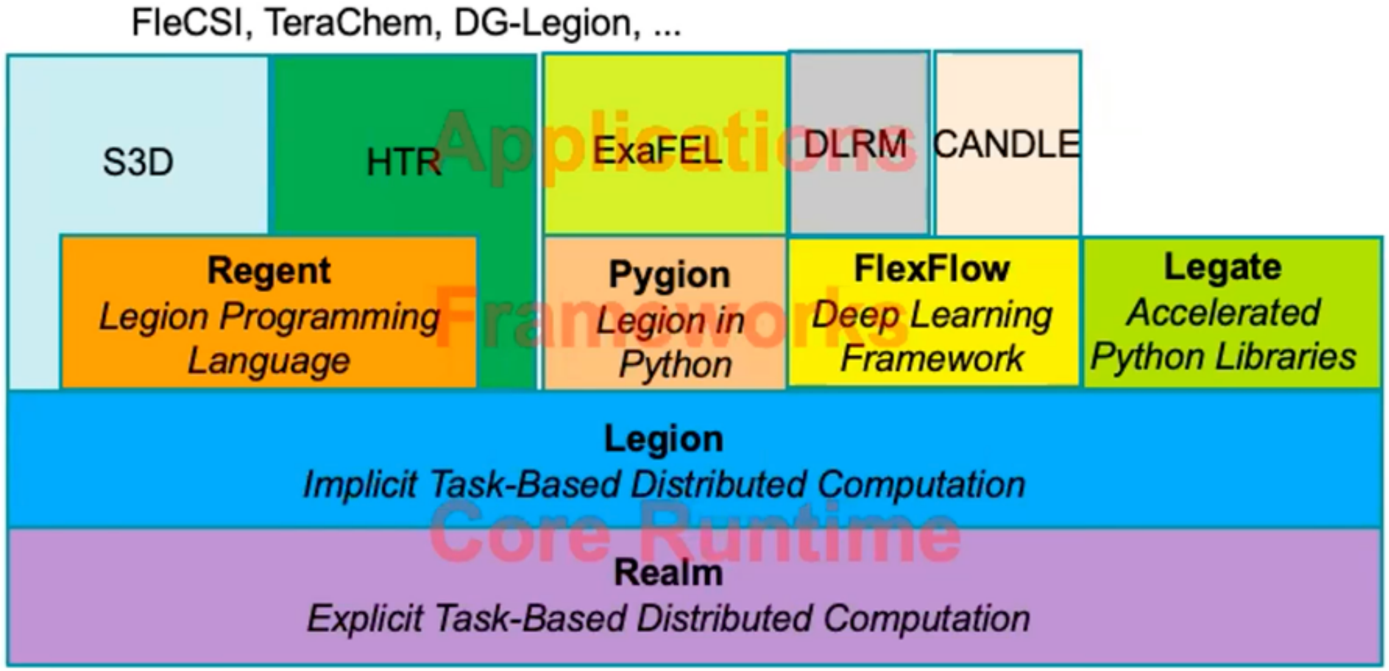
- FFTs in **phasing** module is offloaded to GPU using CuPy, a drop-in replacement of NumPy.



- Euclidean distance calculation, k-Nearest Neighbor (kNN) and sorting in **orientation matching** are accelerated using approximately 800 lines of low-level CUDA code.



Scaling and Parallelization with Legion



Pygion: Flexible, Scalable Task-Based Parallelism with Python

- *Tasks* → functions marked for deferred execution, somewhere on the system
  - Parameters pickled for transfer

- *Privileges* → explicit the rights of the tasks on the regions

- *Regions* → representation of the data
  - Defined shape
  - *Fields,* exposed as NumPy arrays of different types
  - Underlying data only moved if needed
  - Can be *partitioned* into *subregions*

- *Layout* → memory access pattern

Code Example: Regions and Partitions

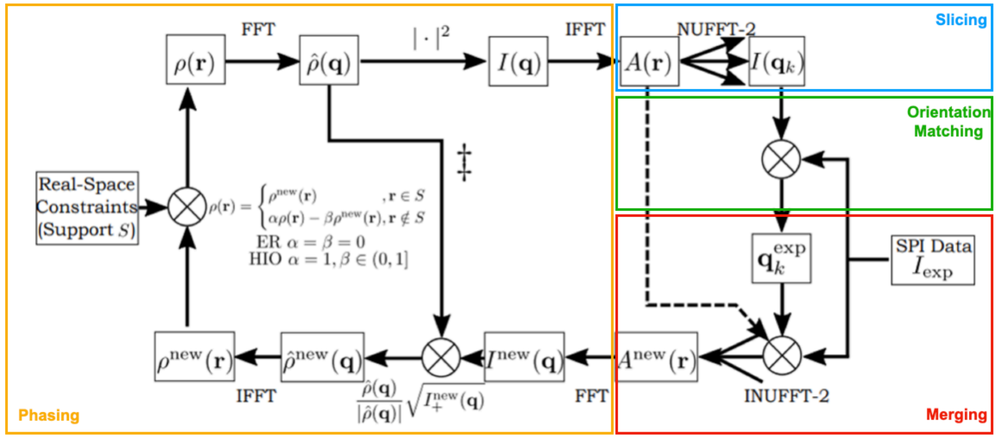
```
def create_distributed_region(N_images_per_proc, fields_dict, sec_shape):
  N_procs = Tunable.select(Tunable.GLOBAL_PYS).get()
  N_images = N_procs * N_images_per_proc
  shape_total = (N_images,) + sec_shape
  shape_local = (N_images_per_proc,) + sec_shape
  region = Region(shape_total, fields_dict)
  region_p = Partition.restrict(region, [N_procs], N_images_per_proc *
  np.eye(len(shape_total), 1), shape_local)
  return region, region_p
```

Pygion: Flexible, Scalable Task-Based Parallelism with Python

- Task-based models are a promising direction
  - Sequential semantics, parallelism discovered automatically
  - Recent advances makes scaling more user-friendly

Code Example: Workflow

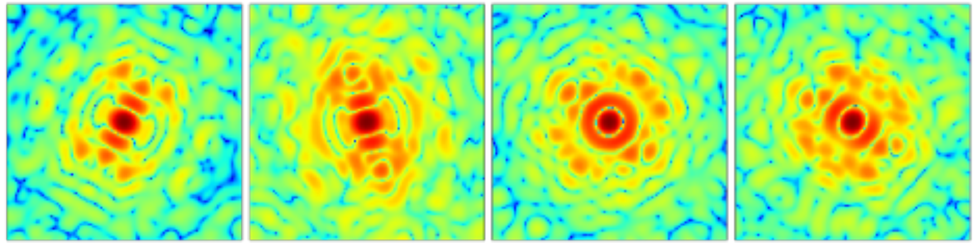
```
while not converged:
  index_launch([N_procs], solve_autocorrelation, slices[ID],
  orientations[ID], phased)
  index_launch([N_procs], slice, autocorrelation[ID], orientations[ID])
  index_launch([N_procs], match_orientations, orientations[ID])
  phase(solved, phased)
  converged = check_convergence(phased)
```



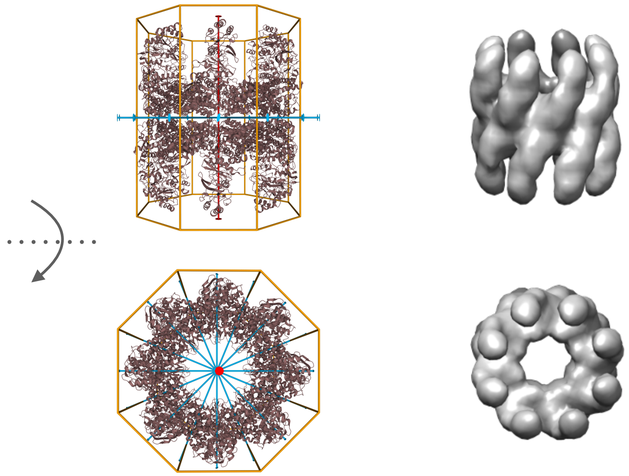
Results

- The experiments were conducted on Summit. Each Summit node consists of 2 POWER9 CPUs, 6 Nvidia V100 GPUs and 512 GB of memory. Nodes are connected via EDR InfiniBand, and GPUs via NVLink.

- To test our framework, a noise-free synthetic dataset of 500,000 SPI images (128 x128 pixels, edge resolution = 12 Å) from a lidless Mm-cpn in the open state (PDB ID: 3IYF) is generated.

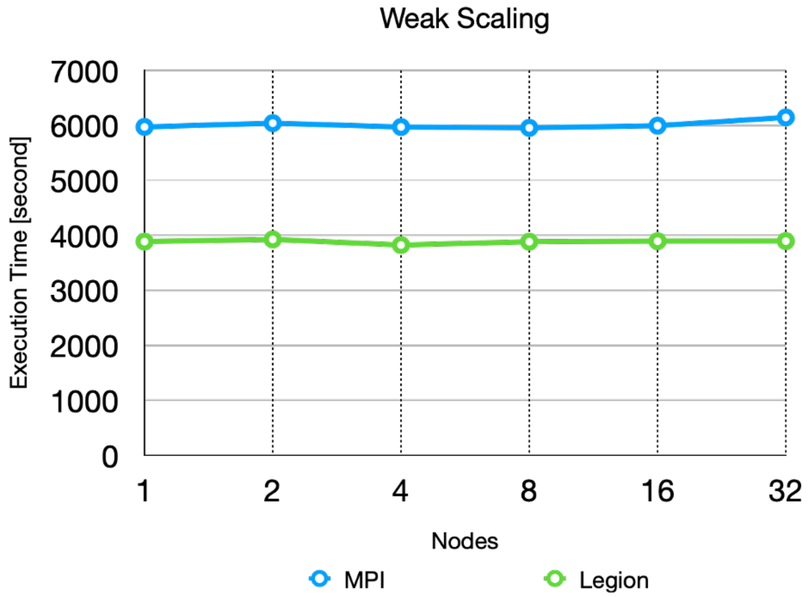


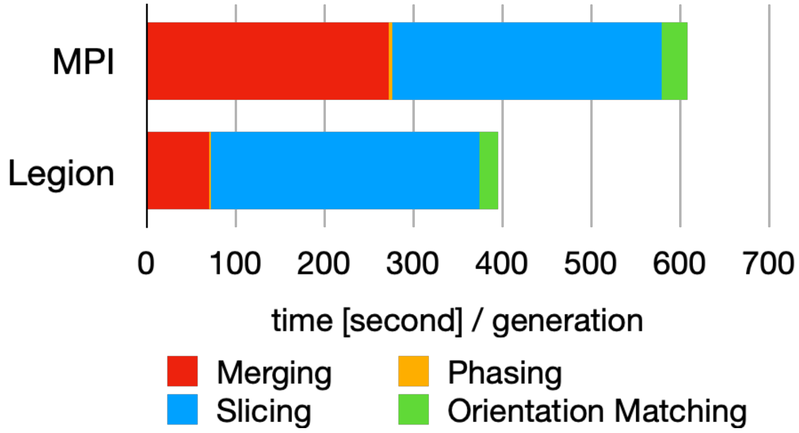
- Implementing M-TIP algorithm using Legion task-based programming model provides useful 3D density map from large datasets in slightly over an hour.



Results

- **Weak scaling** — increase the number of nodes from 1 (1,000 images processed) to 32 (for a total of 32,000 images processed) on Summit, with 42 CPUs and 6 GPUs per node.
- Both the MPI and Legion versions of SpiniFEL perform well with increasing numbers of nodes as indicated by constant scaling.
- Compared to an equivalent MPI implementation, Pygion gives an overall **1.5X** speedup or more.





Conclusion

- We presented improvements to SPI algorithms that enable potential scaling to large node counts.
- These algorithms have been implemented in SpiniFEL, an accelerated and distributed implementation that leverages the Legion tasked-based programming model.
- Compared to an equivalent MPI implementation, Pygion is easier to scale out of the box as it manages load-balancing of tasks across cores, shared memory (between distinct Python processes on a node) and provides high level parallelization constructs.

Thank You ☺

Authors

- Iris Hsing-Yin Chang (SLAC)
- Elliott Slaughter (SLAC)
- Seema Mirchandaney (SLAC)
- Jeffrey Donatelli (LBNL)
- Chun Hong Yoon (SLAC)

Acknowledgements

- This research was supported by the Exascale Computing Project (ECP), Project Number: 17-SC-20-SC, a collaborative effort of two DOE organizations—the Office of Science and the National Nuclear Security Administration—responsible for the planning and preparation of a capable exascale ecosystem—including software, applications, hardware, advanced system engineering, and early testbed platforms—to support the nation's exascale computing imperative.
- Use of the Linac Coherent Light Source (LCLS), SLAC National Accelerator Laboratory, is supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Contract No. DE-AC02-76SF00515.
- This work was performed on Summit at OLCF.