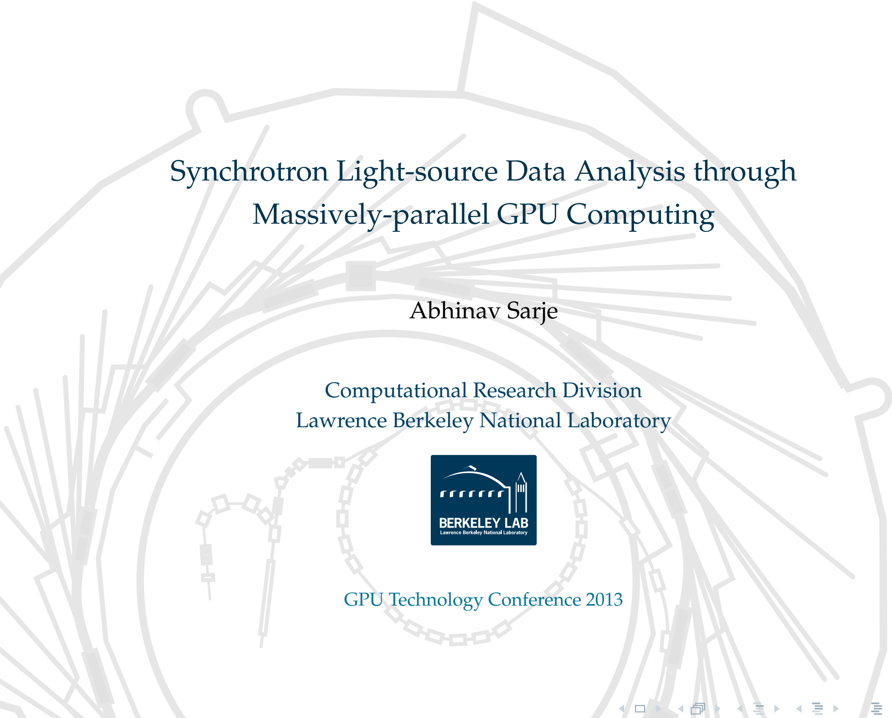


Synchrotron Light-source Data Analysis through Massively-parallel GPU Computing

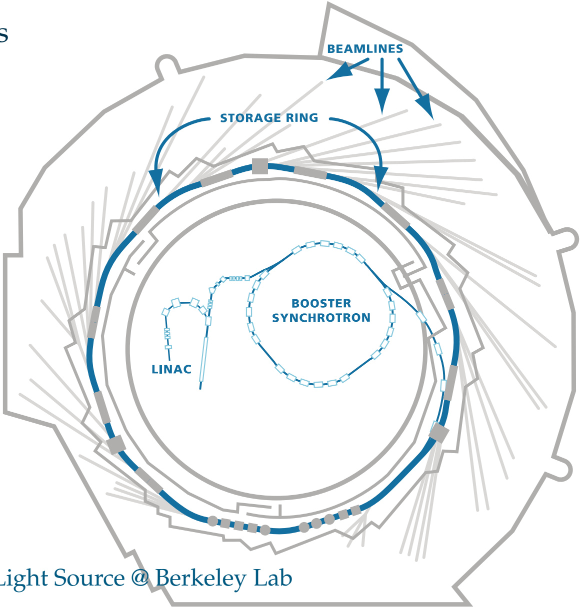
Abhinav Sarje

Computational Research Division
Lawrence Berkeley National Laboratory

GPU Technology Conference 2013

Synchrotron Light-Sources

- Electron accelerator to generate high-intensity electromagnetic radiation.
- Radiation in form of high-intensity beams, used for experiments at beamlines.

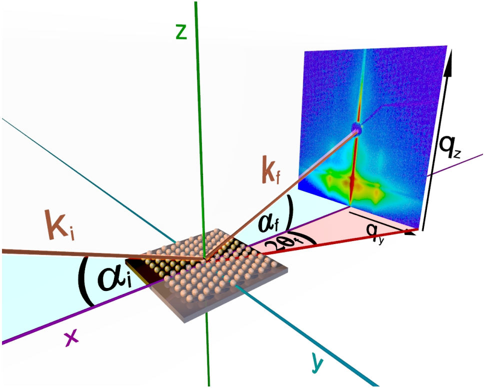


Advanced Light Source @ Berkeley Lab

High-energy X-ray Scattering

- X-ray scattering to measure structural properties of materials, and
- characterize macromolecules and nano-particle systems at micro and nano-scales.
  - probing the electronic structure of matter,
  - semiconductors,
  - 3D-biological imaging,
  - protein crystallography,
  - chemical reaction dynamics,
  - biological process dynamics,
  - optics,
  - .. and so on.
- Broad variety of applications. E.g.:
  - Materials: Design of energy efficient devices like solar cells, high-density storage media
  - Medicine: Design of synthetic enzymes, drugs and bio-membranes.

High-energy X-ray Scattering



graphic: courtesy of A. Meyer, *www.gisaxs.de*

Examples:

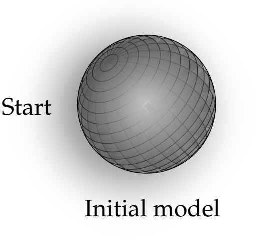
- Small-angle X-ray Scattering (SAXS)
- Grazing Incidence SAXS (GISAXS).

Outline

1. Motivations and the need of HPC.
2. Computational problems in structure prediction.
    - Scattering pattern simulations.
    - Inverse modeling/fitting.
3. GISAXS simulations.
4. HipGISAXS: an HPC solution.
    - Implementation and optimizations.
    - Performance analysis.
5. Conclusions and ending notes.

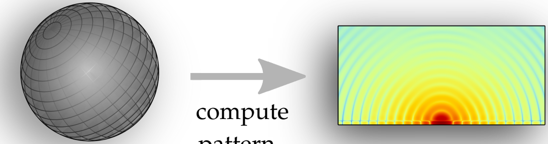
Computational Problems in Structure Prediction

- An example workflow:



Computational Problems in Structure Prediction

- An example workflow:



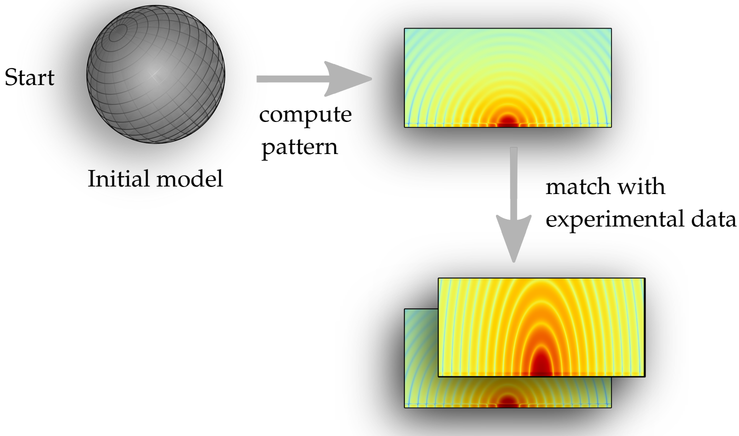
Start

Initial model

compute pattern

Computational Problems in Structure Prediction

- An example workflow:



Start

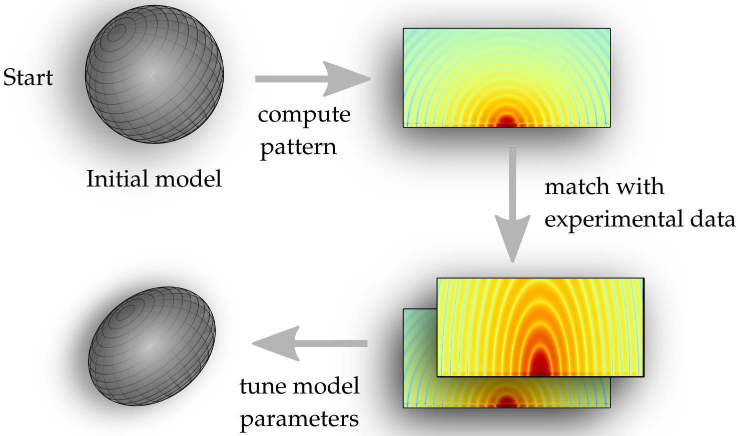
Initial model

compute pattern

match with experimental data

Computational Problems in Structure Prediction

- An example workflow:



Start

Initial model

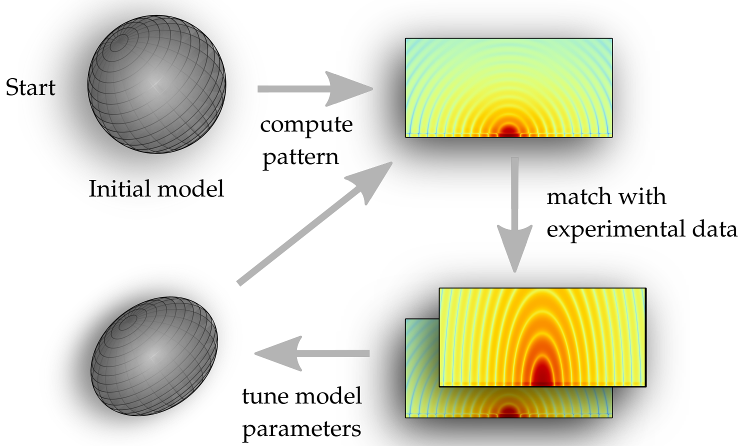
compute
pattern

match with
experimental data

tune model
parameters

Computational Problems in Structure Prediction

- An example workflow:



Start

Initial model

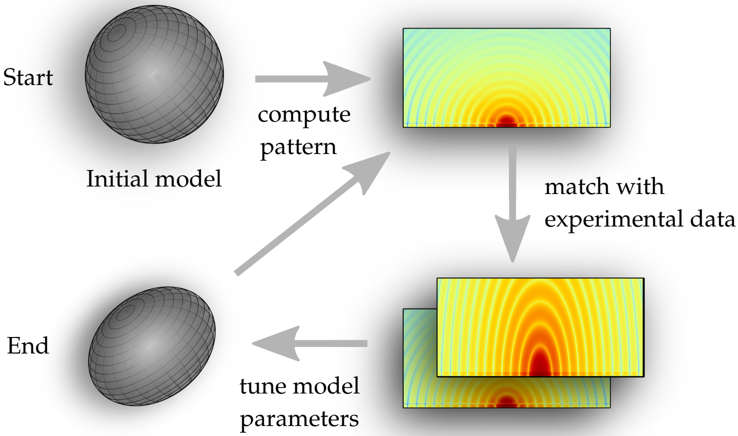
compute pattern

match with experimental data

tune model parameters

Computational Problems in Structure Prediction

- An example workflow:



Start

Initial model

compute
pattern

match with
experimental data

tune model
parameters

End

Need for High-Performance Computing

Mismatch in rates of data generation and data processing:

- High measurement rates of current state-of-the-art light sources.
- Inefficient utilization of facilities due to mismatch.
- *Example*: 100 MB raw data per second. Up to 12 TB per week.

Need for High-Performance Computing

High Computational and Accuracy Requirements:

- Errors are proportional to resolutions of various computational discretization.
- Higher resolutions require greater computational power.
- *Example*: $O(10^7)$ to $O(10^{14})$ kernel computations for one experiment. $O(10^2)$ experiments per material sample.

Need for High-Performance Computing

Science Gap:

- Beam-line scientists lack access to fast algorithms and codes.
- In-house developed codes, limited in compute capabilities and performance.
- Also, they are slow – wait for days and weeks to obtain results.

Fortunately ...

- Involved computations have high degree of parallelism.
- Largely independent computations.
- Perfect for "GPUization".

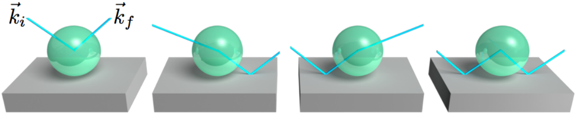
Forward Simulations: Computing Scattered Light Intensities

Given:

1. a sample structure model, and
2. experimental configuration,

simulate experiments and generate scattering patterns.

Based on *Distorted Wave Born Approximation* (DWBA) theory.



Forward Simulations: Computing Scattered Light Intensities

Q-grid: a 3D region grid in inverse space where scattered light intensities are to be computed.

Intensity: is computed at each Q-grid point $\vec{q}$.
At a point $\vec{q}$, it is proportional to square of the sum of *Form Factors* at $\vec{q}$, due to all structures in the sample:

$$I(\vec{q}) \quad \propto \quad \left| \sum_{s=1}^{S} F(\vec{q}) \right|^2$$

Forward Simulations: Computing Form Factors

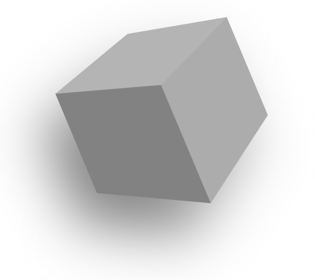
- Form Factor at $\vec{q}$ is computed as an integral over shape surface.

$$F(\vec{q}) = -\frac{i}{|q|^2} \int_{S(\vec{r})} e^{i\mathbf{q_r}\cdot\vec{r}} q_n(\vec{r}) d^2\vec{r}$$

- Approximated as a discretized surface (triangulated surface) summation:

$$F(\vec{q}) \approx -\frac{i}{|q|^2} \sum_{k=1}^{t} e^{i\vec{q}\cdot\vec{r}_k} q_{n,k} \sigma_k$$

- Complex number computations.
- Analytically for simple shapes.



Forward Simulations: Computing Form Factors

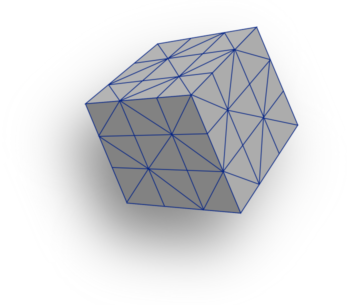
- Form Factor at $\vec{q}$ is computed as an integral over shape surface.

$$F(\vec{q}) = -\frac{i}{|q|^2} \int_{S(\vec{r})} e^{i\mathbf{q}_{\mathbf{r}}\cdot\vec{r}} q_n(\vec{r}) d^2\vec{r}$$

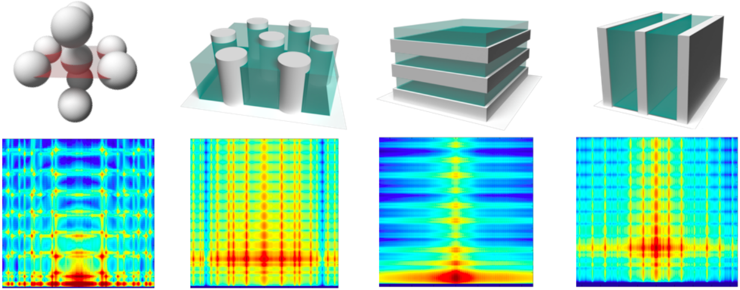
- Approximated as a discretized surface (triangulated surface) summation:

$$F(\vec{q}) \approx -\frac{i}{|q|^2} \sum_{k=1}^{t} e^{i\vec{q}\cdot\vec{r}_k} q_{n,k} \sigma_k$$

- Complex number computations.
- Analytically for simple shapes.

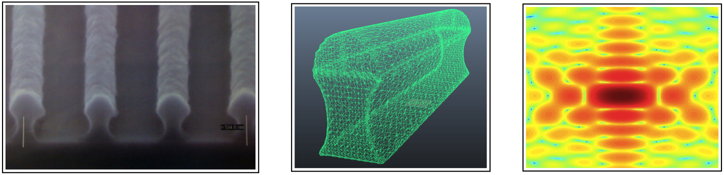


Forward Simulations: Analytic Computation Examples

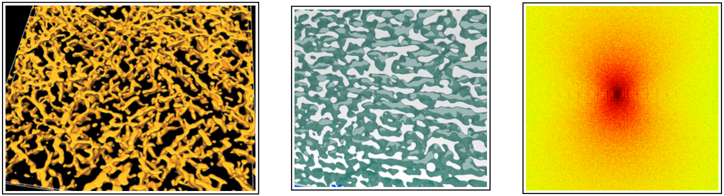


Forward Simulations: Numeric Computation Examples

Rectangular Grating with Undercut:



Organic Photovoltaics (OPV) Tomography:



Real Sample Model Scattering Pattern

HipGISAXS: A High-Performance GISAXS Simulation Code

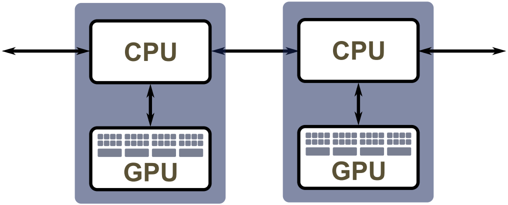
- Solves many limitations of previous codes.
- Implements new flexible algorithms to handle
  - any complex morphology,
  - multi-layered structures, and
  - all sample rotation directions and beam angles.
- Implements parallelization methods:
  - Deliver high-performance on massively parallel state-of-the-art clusters of GPUs and multi-core CPUs.
  - Bring computational time down to just seconds and minutes.
- Written in C++ with MPI, OpenMP and NVIDIA CUDA.
- Flexible and modularized code for future extensions.

Computational Problem

Input: 3 arrays, q_x, q_y, q_z of lengths n_x, n_y, n_z, resp., representing a Q-grid of resolution $n = n_x \times n_y \times n_z$, and Numeric: An array defining the triangulated shape surface as a set of t triangles.

Output: A 3-D matrix M of size $n_x \times n_y \times n_z$, where each $M(i, j, k) = F(q_i, q_j, q_k) = F(\vec{q}_{i,j,k})$.

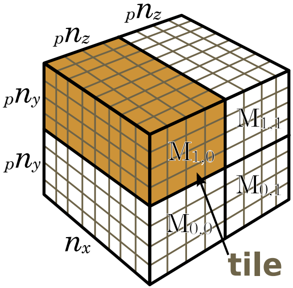
Environment: p node cluster of GPUs/multi-core CPUs.



Computation Decomposition Hierarchy: Tiling

1. Across Multiple Nodes/Processes: *Tiling*

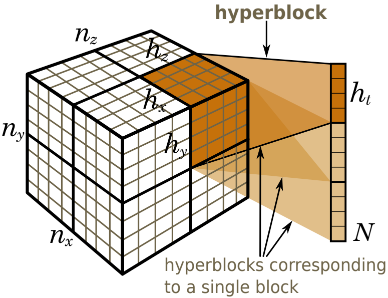
- Partition M along y and z dimensions into grid of $P = P_y \times P_z$ *tiles*.
- x dimension is typically small.
- Tile $M_{i,j}$ is assigned to node $P_{i,j}$.
- Tile data is distributed to respective nodes using MPI.



Computation Decomposition Hierarchy: Blocking

2. Handle Memory Limitations: *Blocking*

- Data may not fit in device memory.
- Partition local tile along x, y, and z into *blocks* of size $h_x \times h_y \times h_z$.
- Partition triangle array into *segments* of size h_t.
- Represent combinations of blocks and segments as 4D *hyperblocks*.



- Process one hyperblock at a time on device.
- Hyperblocks result in partial sums. All partial sums for a block are reduced on host.

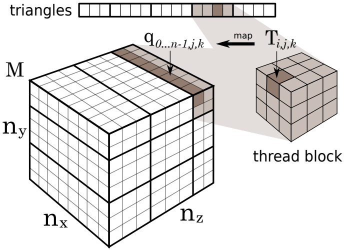
Computation Decomposition Hierarchy: Threading

3. Within device: *threading*

Phase 1 Local Computations.

- Partition along y, z and t into *thread blocks*.
- Compute over a triangle at all grid-points $\vec{q}$ in x dimension:

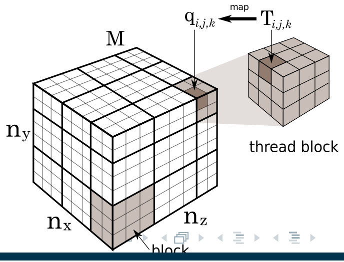
$$F_t(\vec{q}) = e^{i\vec{q}\cdot\vec{r}} s_t$$



Phase 2 Reduction.

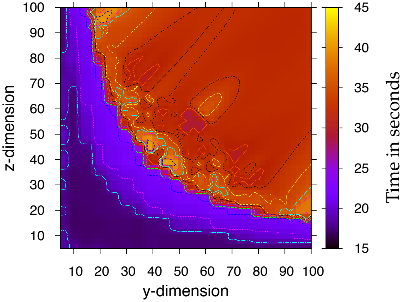
- Partition along x, y and z into *thread blocks*.
- Reduce all F_t at a grid-point $\vec{q}$:

$$F(\vec{q}) \approx \sum_{t=1}^{h_t} F_t(\vec{q})$$

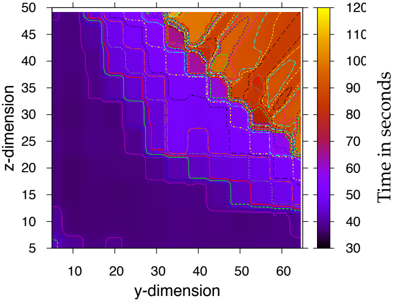


Tuning Hyperblock Size $h_x \times h_y \times h_z \times h_t$

- Crucial for high performance.
- Small size = low parallelism + large number of data transfers.
- Large size = transfer of large amounts of data.
- Find a good balance, explore the search space.
- Example heat maps of runtimes with varying h_y and h_z (4M q-points.)



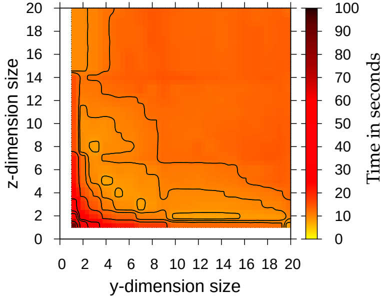
$t = 2,292$



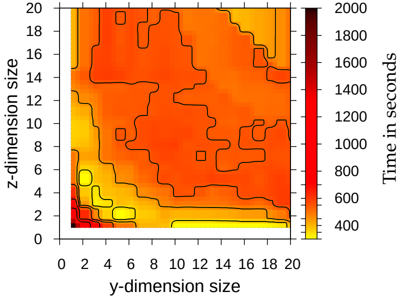
$t = 91,753$

Tuning Thread Block Sizes

- Also crucial for high performance.
- Small size = not enough threads in warps, or small number of warps.
- Large size = small number of thread blocks (less parallelism).
- Find a balanced size, explore search space.
- Example runtime heat maps with varying thread block sizes.



$t = 2,292$



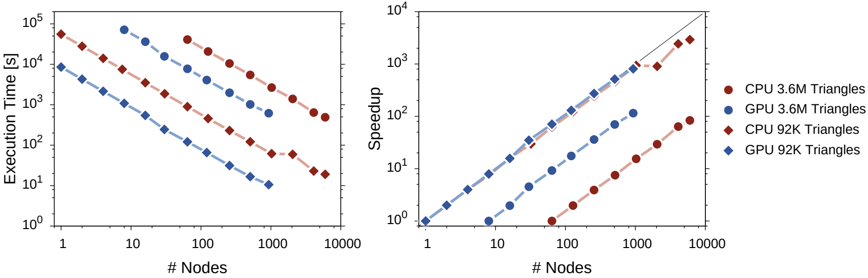
$t = 91,753$

HipGISAXS Performance

- **GPU Cluster:** *"TitanDev"*. Up to 930 nodes.
  - NVIDIA Tesla X2050 Fermi GPUs,
  - 6 GB device memory,
  - 1.15 GHz CUDA core clock,
  - AMD Opteron Interlagos 16 core CPU,
  - 32 GB main memory,
  - Gemini interconnects.
- Single precision complex number computations.

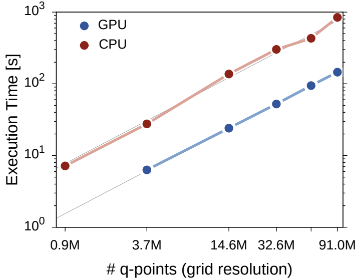
Strong Scaling with Number of Nodes

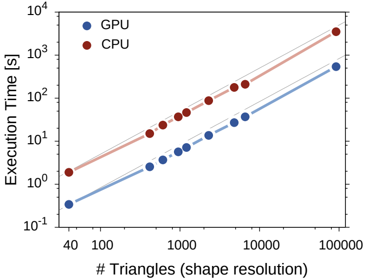
- GPU cluster = 1 to 930 nodes.
  - One MPI process per node. 16 OpenMP threads on host.
- CPU cluster = 1 to 6,000 nodes (24 to 144,000 cores).
  - Four MPI processes per node. 6 OpenMP threads per MPI process.
- *Q*-grid size = 91M *q*-points.
- Expected scaling = linear, observed = linear.



Scaling with Input Sizes n & t

- Q-grid resolution, n = 0.9M to 91M q-points (left).
- Shape resolution, t = 40 to 91K triangles (right).
- Number of nodes used = 4.
- Expected = linear, observed = linear.





Observations & Comparisons

Comparison	**GPU (930 Nodes)**	**CPU (6,000 Nodes)**
Single node speedup (wrt sequential code)	125×	20×
Performance ratio	1	6.25
Cluster speedup (relative to single node)	900× (96%)	5400× (90%)
Throughput (billion *q*-points per second)	999.98	941.07
Code base size ratio (LOC)	1.45	1
Development time person-hours ratio	4	1

End Notes

- Synchrotron light-source data analysis computations are well suited for GPUs due to high degree of parallelism.
- Developed high-performance GISAXS forward simulation code on GPU clusters:
- Perform of much larger samples ($O(10^6)$ triangles) and with higher resolutions ($O(10^8)$ q-points) than previously feasible.
- Brought down computational time from days and weeks to minutes and seconds.
- Ongoing work ...

Acknowledgments

- This work is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
- Used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.
- Used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

Thank you!