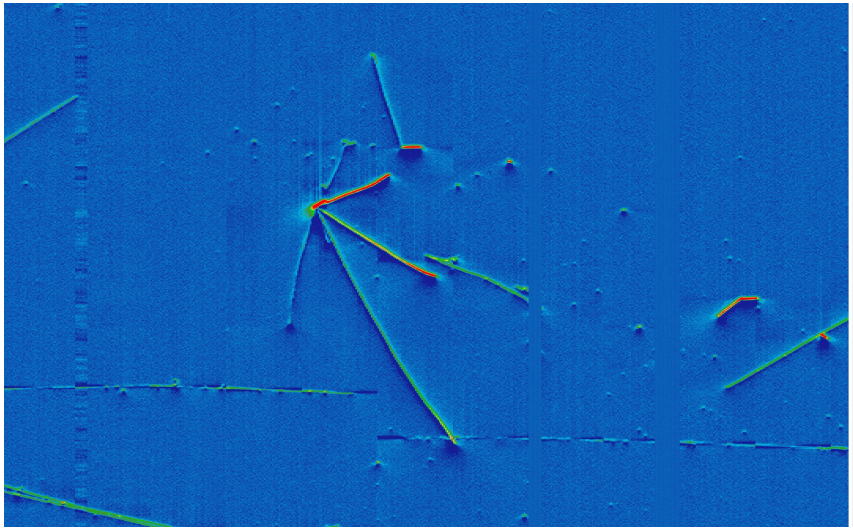
Reconstruction for LArTPC Neutrino Detectors Using Parallel Architectures



Sophie Berkman

for the SciDAC HEP Reconstruction and HEP at HPC groups

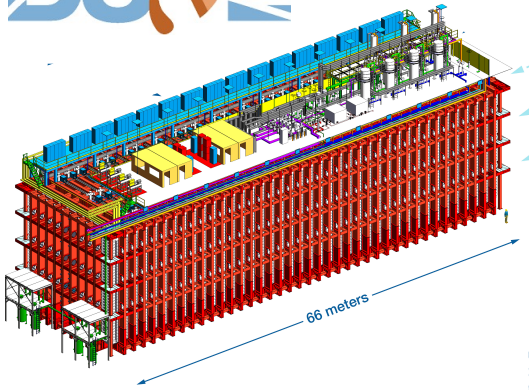
December 1, 2021

SciDAC
Scientific Discovery through Advanced Computing

Fermilab

Neutrino Physics in LArTPCs

- Series of operating, commissioning, and planned liquid argon time projection chambers (LArTPC) detectors in US

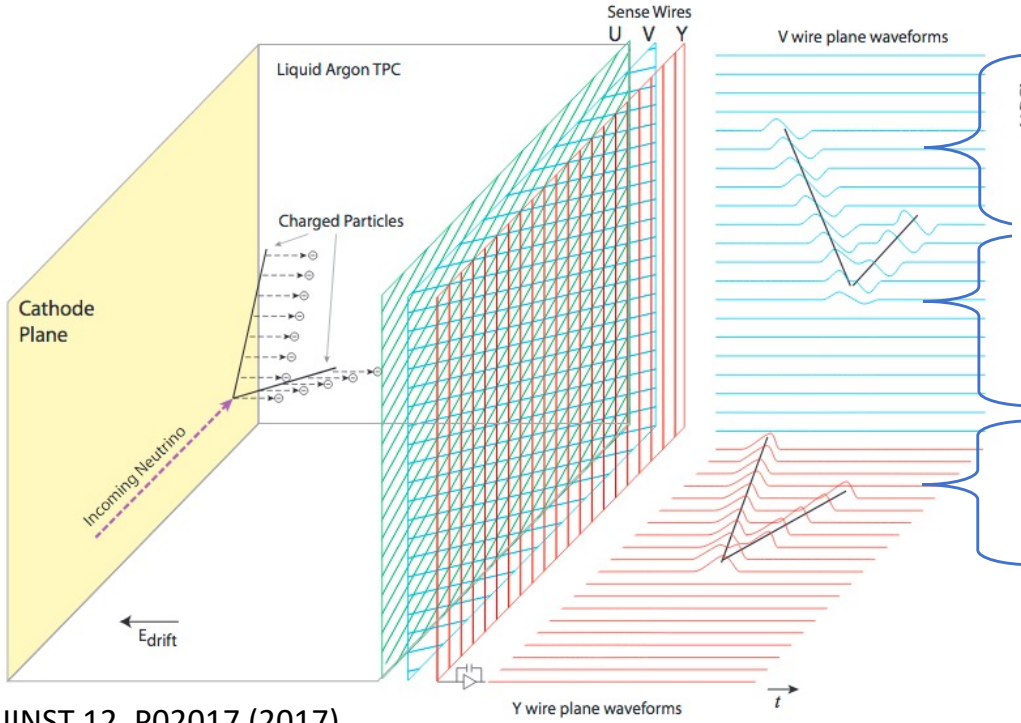


Operation: 2015-2021 2021- Future

- Answer questions such as:
  - Which neutrino is the heaviest, which is the lightest?
  - Do neutrinos and anti-neutrinos oscillate in the same way? Is there CP violation in neutrinos?
  - Can we understand and explain anomalies in neutrino physics?

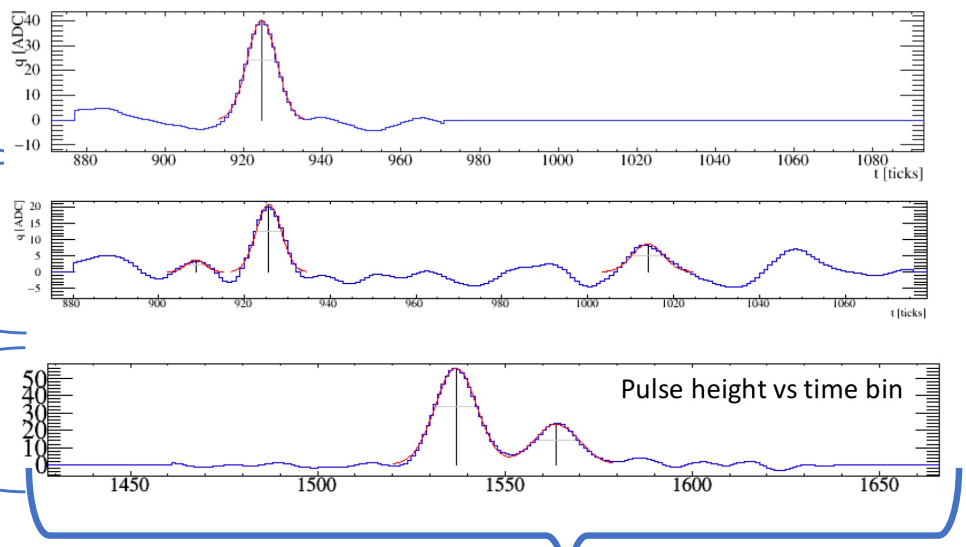
LArTPC Detectors

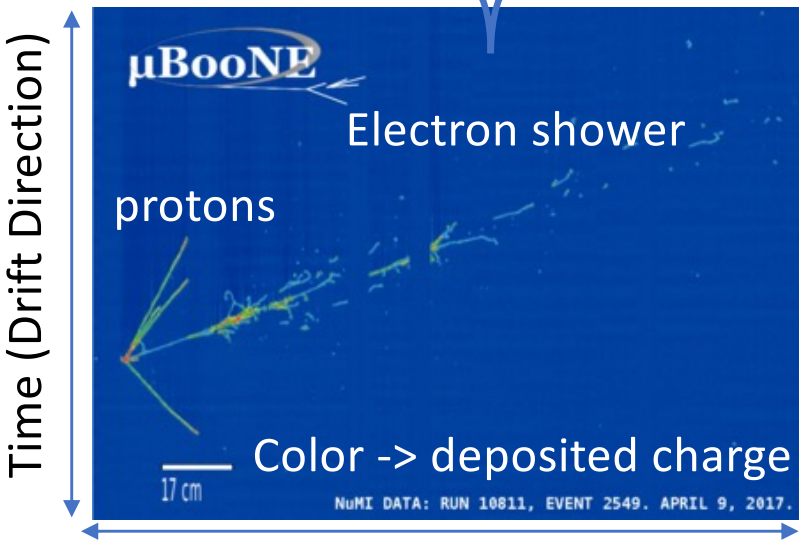
- Detailed images of neutrino interactions



JINST 12, P02017 (2017)

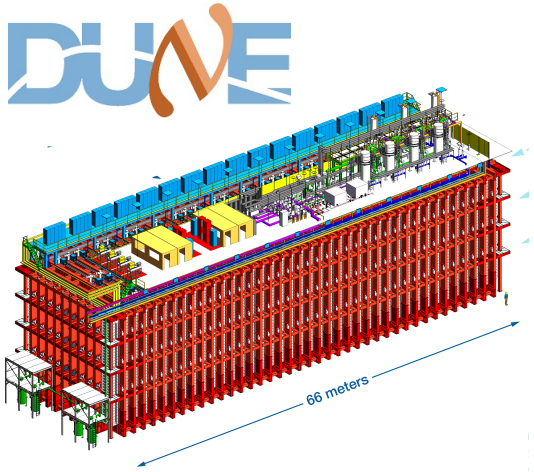
Example Wire Pulses





Future Liquid Argon TPCs

- Detectors are growing bigger, and more neutrinos are expected



1x wires → 5x wires → 100x wires

1x mass → 5x mass → 500x mass

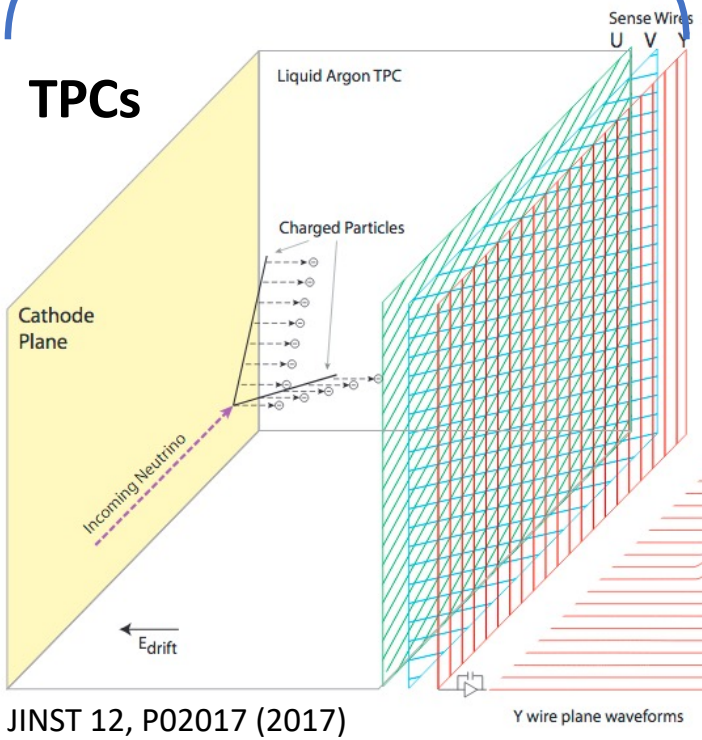
- Challenges ahead to process the data from these experiments efficiently to meet physics goals
  - Already a limiting factor, must actively confront

Parallelizing Event Reconstruction

Cryostats

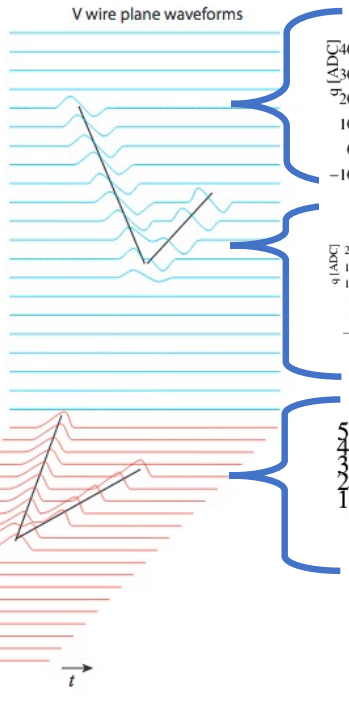
- Given computing trends, parallelism is necessary to gain speed increases
- Typically parallelize at event level
- Can also take advantage of parallelism within events
  - Use computing resources to their full capacity

TPCs

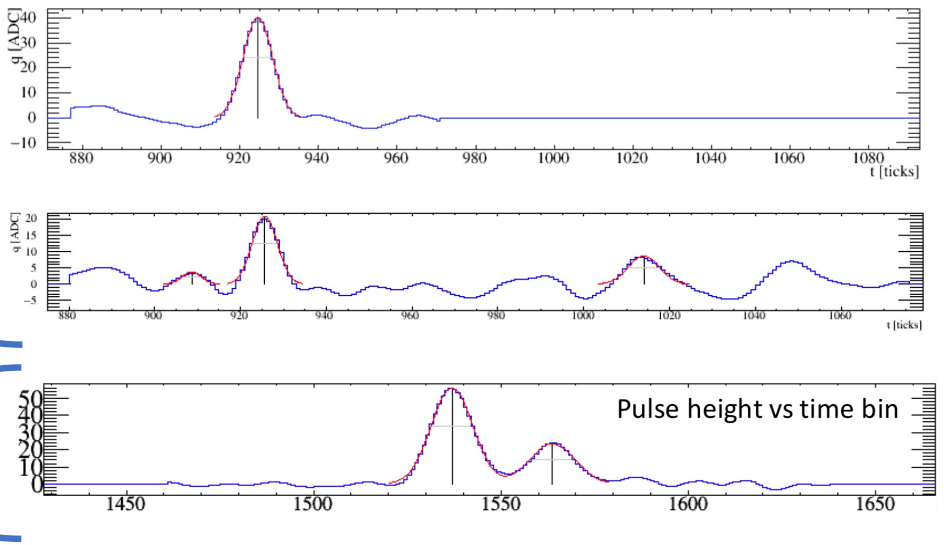


JINST 12, P02017 (2017)

Wire Planes



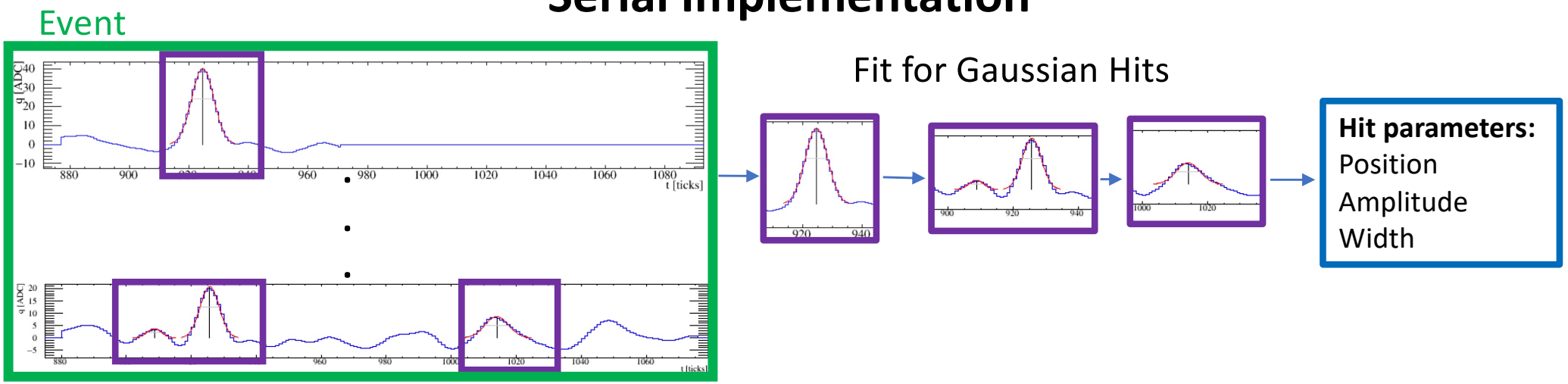
Wires



Gaus Hit Finder Optimization

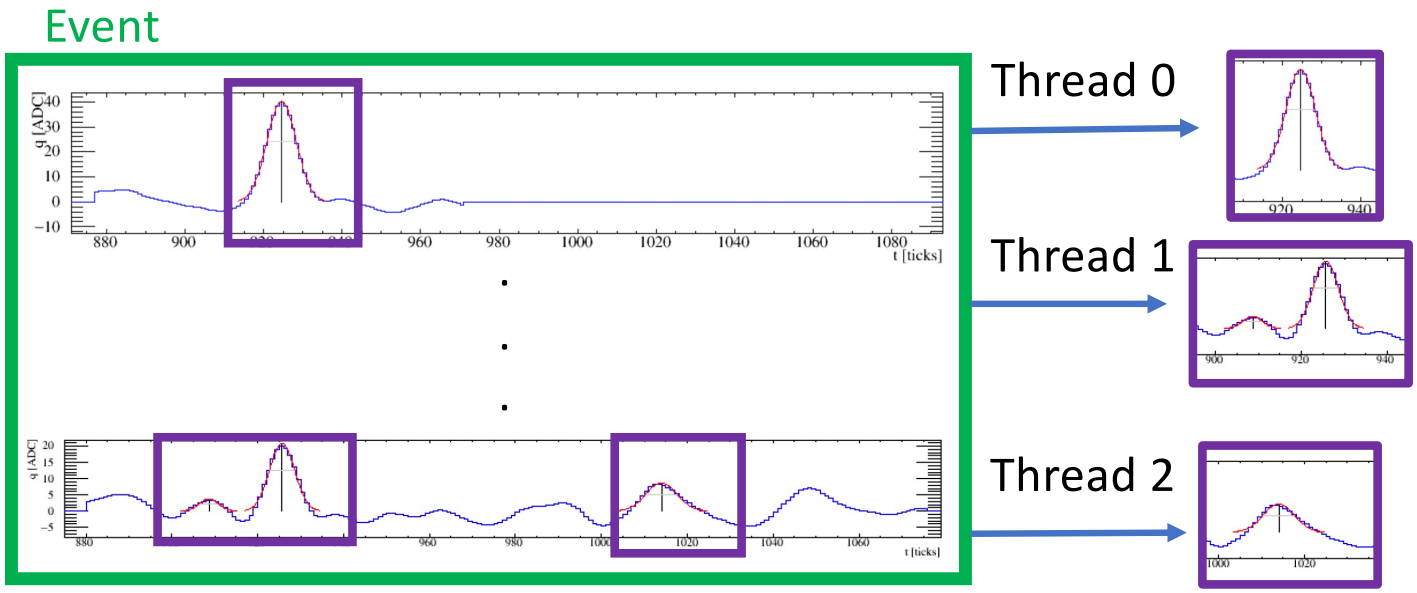
- **Algorithm:** Charged particles produce pulses on wires. Identify parameters associated with pulses, or “hits” (position, amplitude, width).
- **Suited for parallel implementation:** Wires are independent; can be processed independently
- **Potential for impact**: Up to 30% of reconstruction time, depending on the experiment
- Implement dedicated Levenberg-Marquardt algorithm for multi-Gaussian fitting instead of ROOT/Minuit: ~8x faster than ROOT version
- Developments performed within a standalone application

Serial Implementation

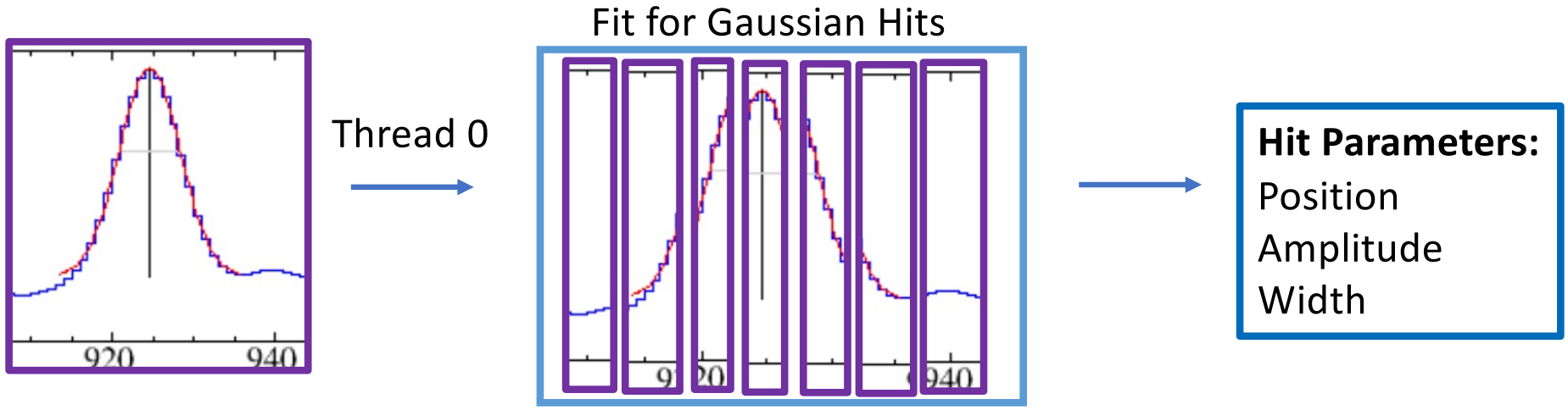


Gaus Hit Finder Parallelization

Multi-Threaded: process independent data regions on separate threads

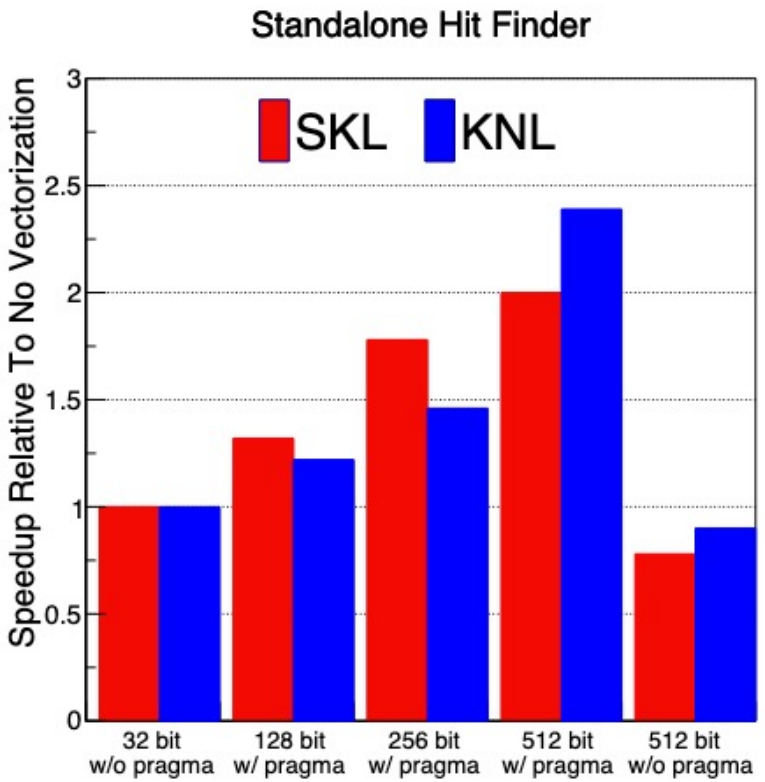


Vectorization: perform same operation in parallel on different data bins



Vectorization of Stand-Alone GausHitFinder

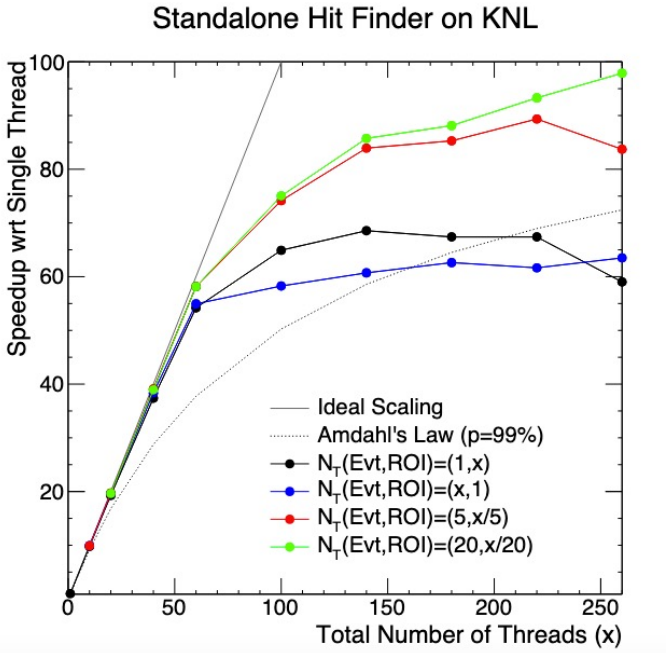
- Vectorization challenges:
  - Minimization difficult because fits converge in different numbers of iterations
  - Cannot fit multiple hits at the same time
  - Vectorize most time consuming loops, over waveform data in each hit, but this is not all of the code
- Vectorization strategies:
  - Compiler vectorization: use avx512
  - Explicit vectorization on the most time consuming loops based on profiling
    - #pragma omp simd, #pragma ivdep
    - Mostly function value computations and derivatives across data bins
- Speed increases: 2-2.3 times faster than with no vectorization

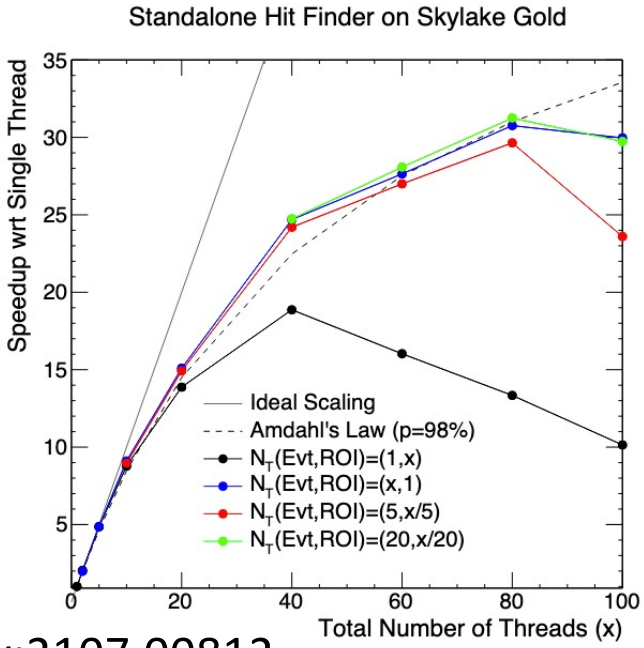


arXiv:2107.00812

Multi-Threading of Stand-Alone GausHitFinder

- Multi-threading using OpenMP
  1. Parallel for loop over events
  2. Parallel region with OMP for + critical (to synchronize output) over regions of interest on the wires
  - Fastest with “dynamic” thread scheduling
- Multi-threading challenges:
  - Algorithm has a relatively small amount of work
  - Thread overhead and imbalance may limit speed up
- Speed increases with multi-threading:
  - KNL: up to 100 times faster
  - Skylake: up to 30 times faster
  - Parallel fractions based on Amdahl’s law: 98-99%

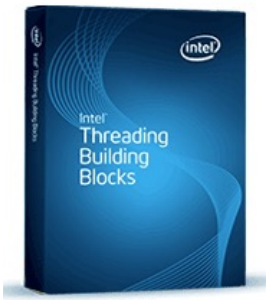




arXiv:2107.00812

Integration into Experiment Code

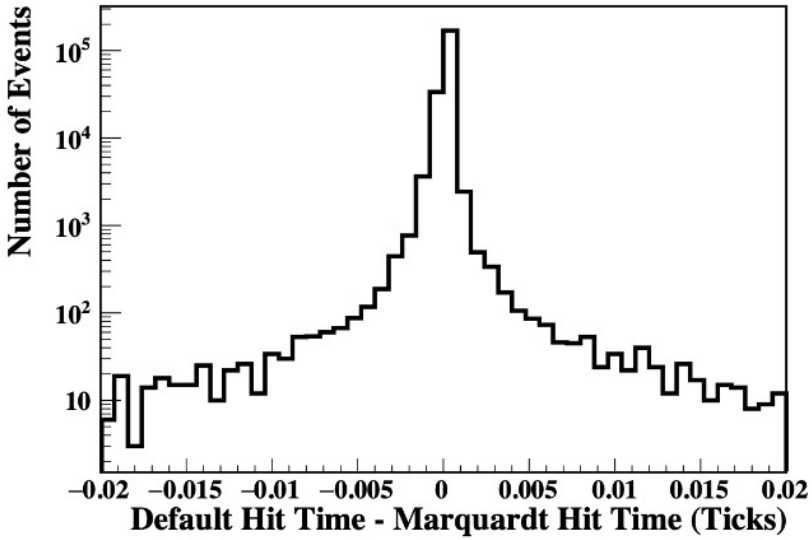
- Common software framework used by all Fermilab LArTPC neutrino experiments.
- Integrate parallelized algorithm:
  1. Add algorithm plug-in: 7-12 times faster
  2. Multi-threading over wires and regions of interest:
     - TBB parallel for loops, focus on parallelism within events as event parallelism is part of the framework
     - TBB concurrent vectors used to save output in thread-safe way
     - SKL: 17 times faster with 95% parallel fraction. (97% parallel fraction within events in stand alone code)
  3. Vectorization:
     - Use spack to custom compile fitting algorithm with icc+avx512 while not changing other code compilation
     - SKL: 2 times faster, similar to stand alone



arXiv:2107.00812

Results In Experiment Framework

- Validation on samples from experiments with different noise levels, on simulation and data
- Negligible difference in physics output for new algorithm:
  - 98% of reconstructed hit times within 0.02 waveform bin units of original, and similar results for other hit parameters
  - No loss in efficiency for identifying hits
- This hit finder is now used by the Icarus and ProtoDUNE experiments



arXiv:2107.00812

HPC Workflow

- Developing workflow to run part of Icarus production on Theta HPC at ALCF:
  - Enable more parallelism than available in typical grid production environment: use improvements to their full capacity
  - Efficiently include AI reconstruction algorithms, such as graph neural networks (ACAT Poster 730)
- Strategy:
  - Local build of LArSoft with **spack** on Theta:
    - Custom compiler options for specific pieces of code: ie. avx512+icc for optimal vectorization speed increases with GausHitFinder
  - **HDF5-HEPNOS** to organize the data
  - **DIY** to distribute the data across nodes
- First spack build on Theta complete
- Profiling code to look for more opportunities to introduce parallelism using tools and expertise from hit finder optimization

Argonne NATIONAL LABORATORY

Refs: 1, 2

HEPnOS-Dataloader
Project ID: 2282

Refs: 1, 2

MATHEMATICS AND COMPUTER SCIENCE DIVISION

DIY: Do-it-Yourself Analysis DIY

Refs: 1, 2, 3

Conclusions

- LArTPC reconstruction algorithms can be parallelized to more efficiently use computing resources and enable physics of interest
- GausHitFinder algorithm:
  - **Optimized:** Up to 200 times faster in stand alone version with vectorization and multi-threading
  - **Made available to experiments:** initially O(10x) faster in LArSoft with similar thread scaling and vectorization performance as in the stand alone version
  - **Enable use on HPC:** work ongoing to run production for Icarus at the Theta ALCF HPC to take full advantage of parallelization speed increases
- Use this work to introduce additional algorithm parallelism in LArTPC reconstruction and make it available to experiments through processing at HPCs

Paper submitted to JINST (arXiv:2107.00812)