Fermilab U.S. DEPARTMENT OF ENERGY | Office of Science

Exploring the Performance of Spark for a Scientific Use Case

Saba Sehrish (ssehrish@fnal.gov), Jim Kowalkowski and Marc Paterno

IEEE International Workshop on High-Performance Big Data Computing (HPBDC)
In conjunction with the 30th IEEE IPDPS 2016
05/27/2016

Scientific Use Case: Neutrino Physics

- The neutrino is an elementary particle which holds no electrical charge, travels at nearly the speed of light, and passes through ordinary matter with virtually no interaction.
  - The mass is so small that it is not detectable with our technology
- Neutrinos are among the most abundant particles in the universe.
  - Every second trillions of neutrinos from the sun pass through your body.
- There are three flavors of neutrino: electron, muon and tau.
  - As a neutrino travels along, it may switch back and forth between the flavors. These flavor "oscillations" confounded physicists for decades.

Neutrino Unknowns

- The NOvA experiment is constructed to answer the following important questions about neutrinos:
  - What are the heaviest and lightest of neutrinos?
  - Can we observe muon neutrinos changing to electron neutrinos?
  - Do neutrinos violate matter/anti-matter symmetry?
- NOvA - NuMI Off-Axis Electron Neutrino Appearance
  - NuMI - Neutrinos from the Main Injector

The NOvA Experiment

- Fermilab's accelerator complex produces the most intense neutrino beam in the world and sends it straight through the earth to northern Minnesota, no tunnel required.
- Moving at close to the speed of light, the neutrinos make the 500-mile journey in less than three milliseconds.
- When a neutrino interacts in the NOvA detector in Minnesota, it creates distinctive particle tracks.
  - Scientists study the tracks to better understand neutrinos

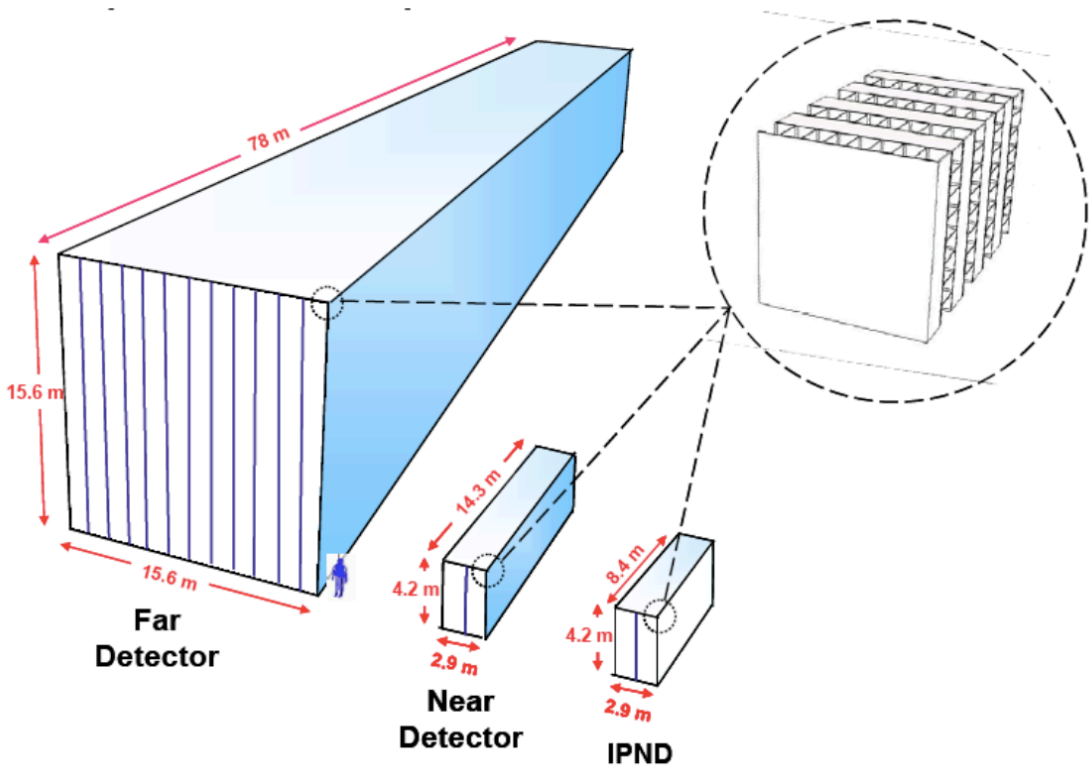
The NOvA Detectors

The NOvA experiment is composed of two liquid scintillator (95% baby oil) detectors,

- A 14,000 ton Far Detector on the surface at Ash River
- A ~300 ton Near Detector (~100m underground) at Fermilab, 1 km from source

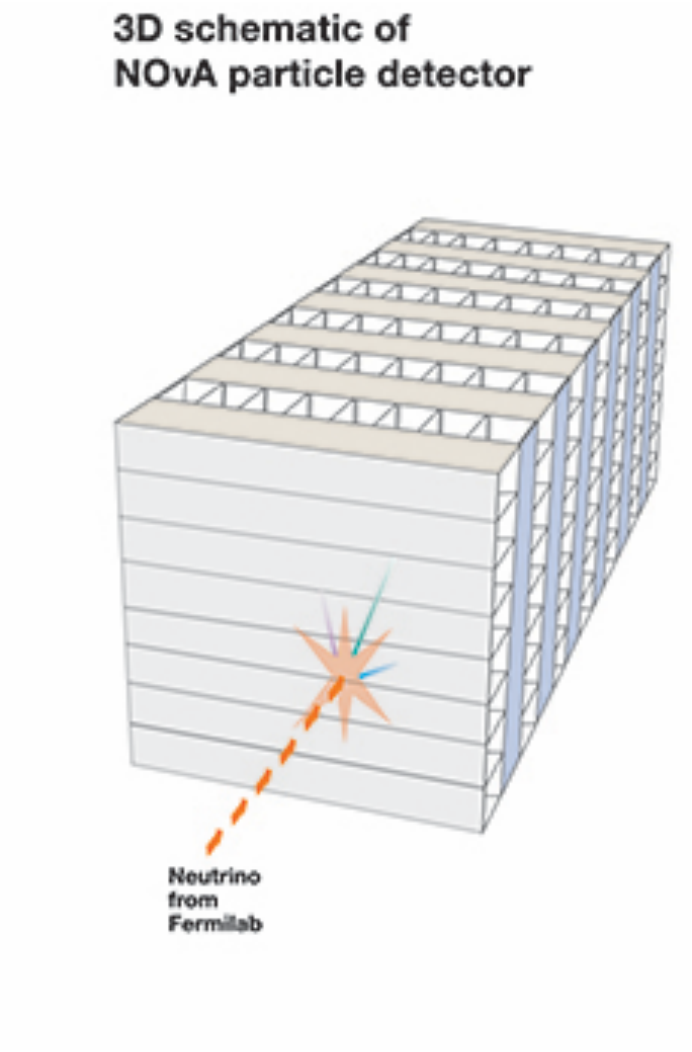
The NOvA detectors are constructed from planes of PVC modules alternating between vertical and horizontal orientations.

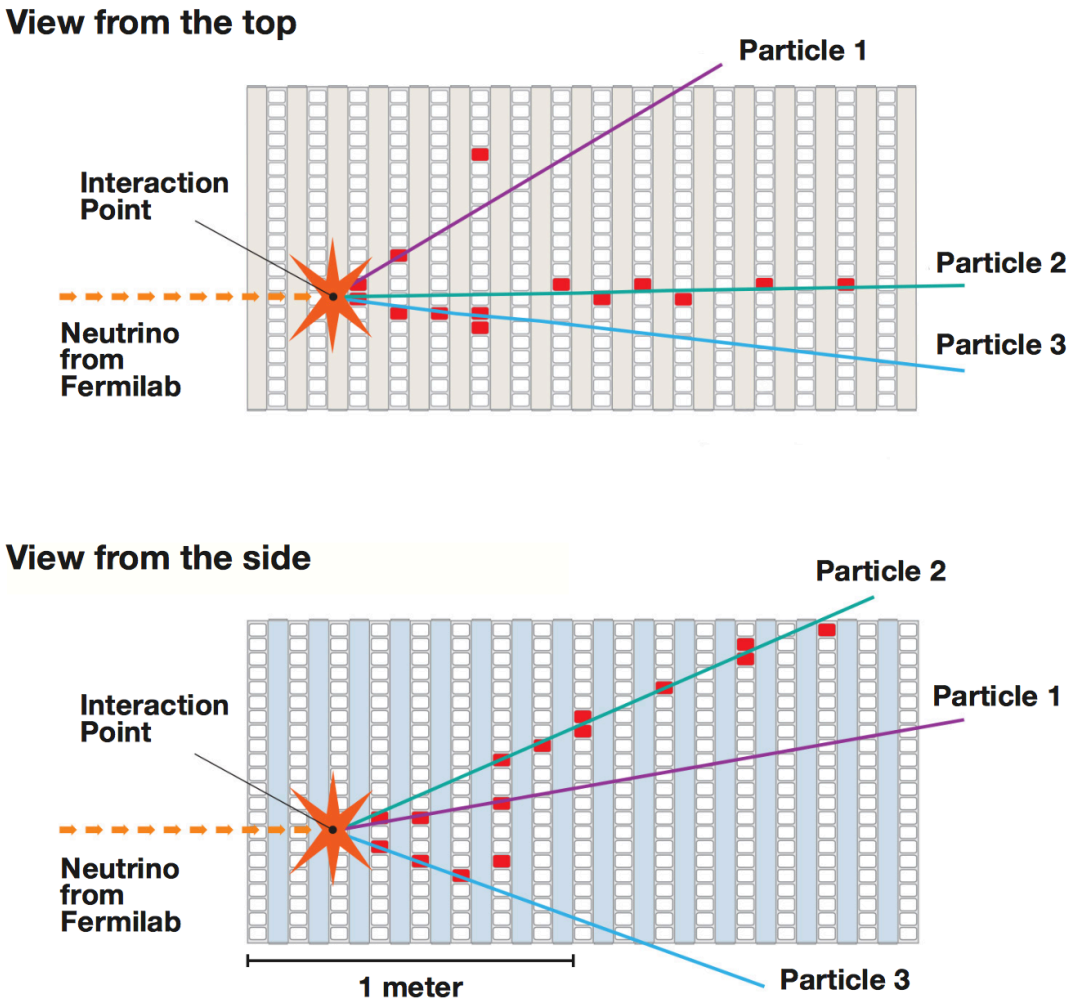
- they form about 1000 planes of 50ft stacked tubes, each about 3x5cm



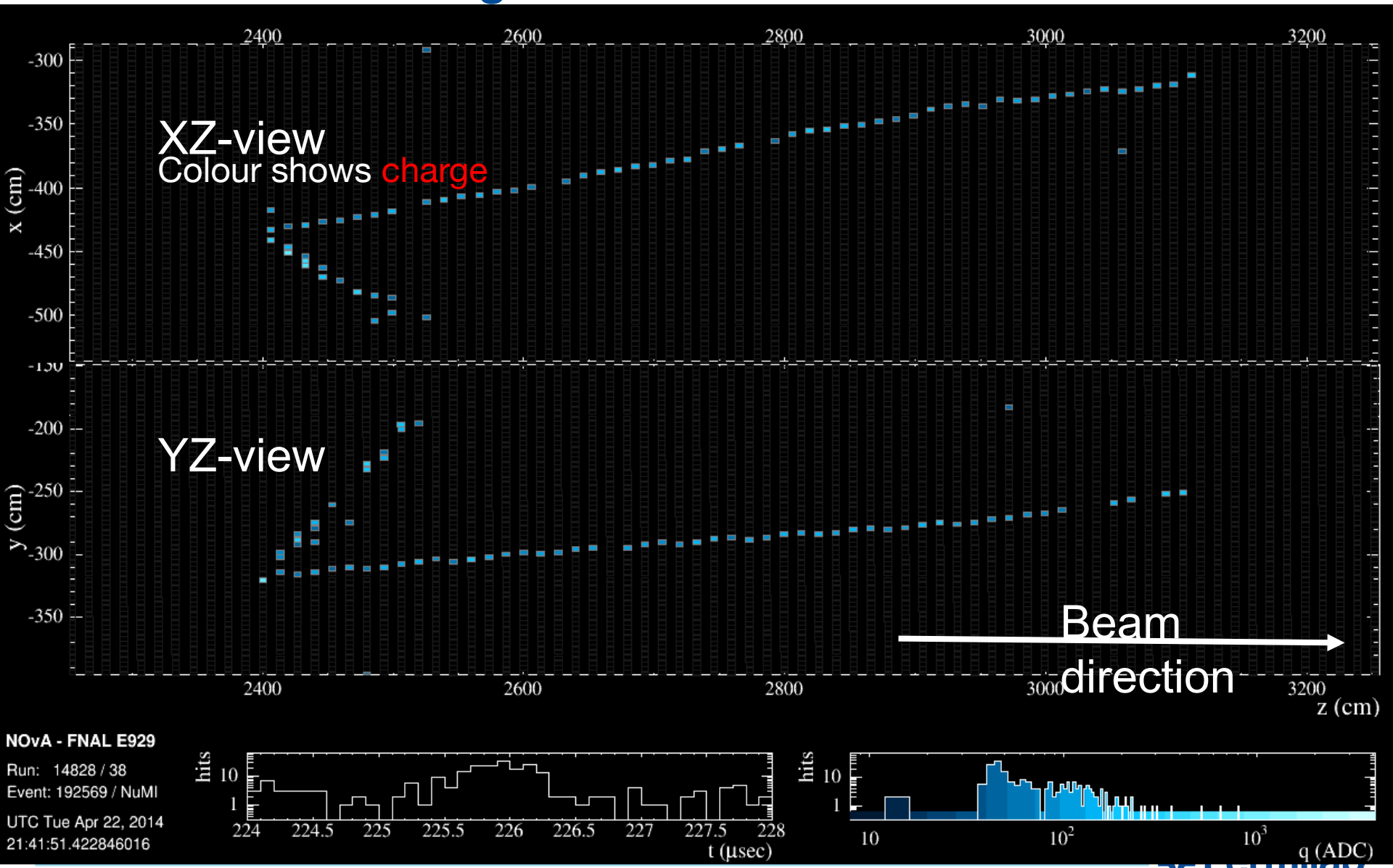
Fermilab

Neutrino interactions recorded by NOvA

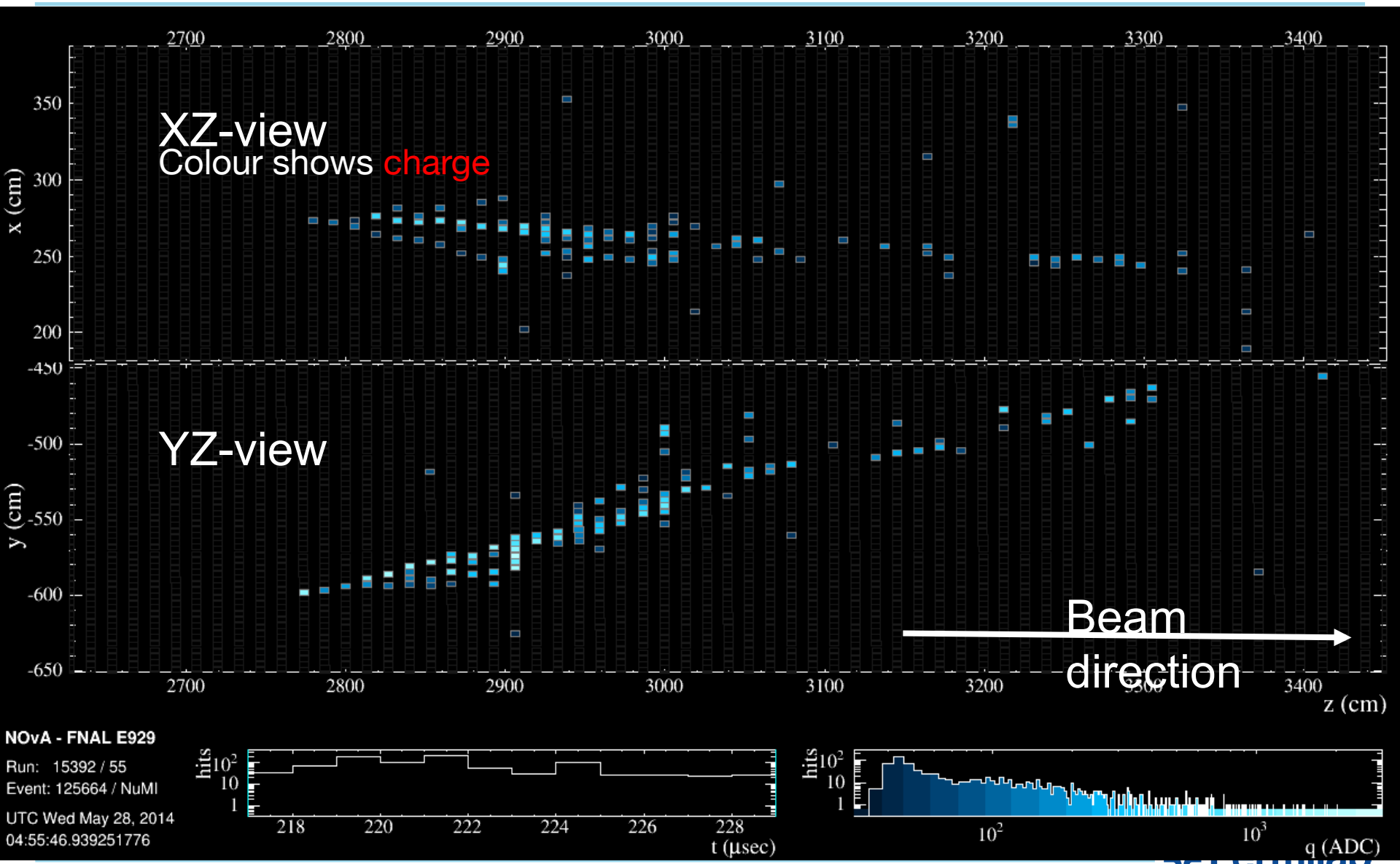




Muon-neutrino Charged-Current Candidate



Electron-neutrino Charged-Current Candidate



Physics problem

- Classify types of interactions based on patterns found in the detector:
  - Is it a muon or electron neutrino?
  - Is it a charged current or neutral current?

Library Event Matching Algorithm

- Classify a detector event by comparing its cell energy pattern to a library of 77M simulated events cell energy patterns, choosing 10K that are “most similar”
  - Compare the pattern of energy (hit) deposited in the cells of one event with the pattern in another event.
- The “most similar” metric is motivated by an electrostatic analogy: energy comparison for two systems of point charges laid on top of each other

Algorithm 1: Algorithm to calculate electrostatic energy, this function accepts an event and a template, each event and template has the following information per hit: cells (c), planes (p) and energy (e), and the number of hits (N).

1 function calcEEnergy (A, B);

Input : Event A and template B;
N= the number of hits in the event to be classified;
p_i, c_i, e_i= plane and cell coordinates and energy of hit i in the event to be classified; $i \in N-1$;
N_k= the number of hits in the template k;
$p_{k,i}, c_{k,i}, e_{k,i}$= plane and cell coordinates and energy of hit i in the template k; $i \in N_k - 1$;
Output: The metric value for measuring the closeness of template k to the event.

2 $\beta = 0.5$;

3 T =function of the distance (motivated by the electrostatic analogy) between the hits; see [4] for details;

4 $E_{AB} = \sum_{i=0}^{N-1} \sum_{j=0}^{N_k-1} {e_i}^{\beta} {e_{k,j}}^{\beta} T(p_i - p_{k,j}, c_i - c_{k,j})$;

5 return E_{AB};

$$E = \frac{1}{2}E_{AA} + \frac{1}{2}E_{BB} - E_{AB}.$$

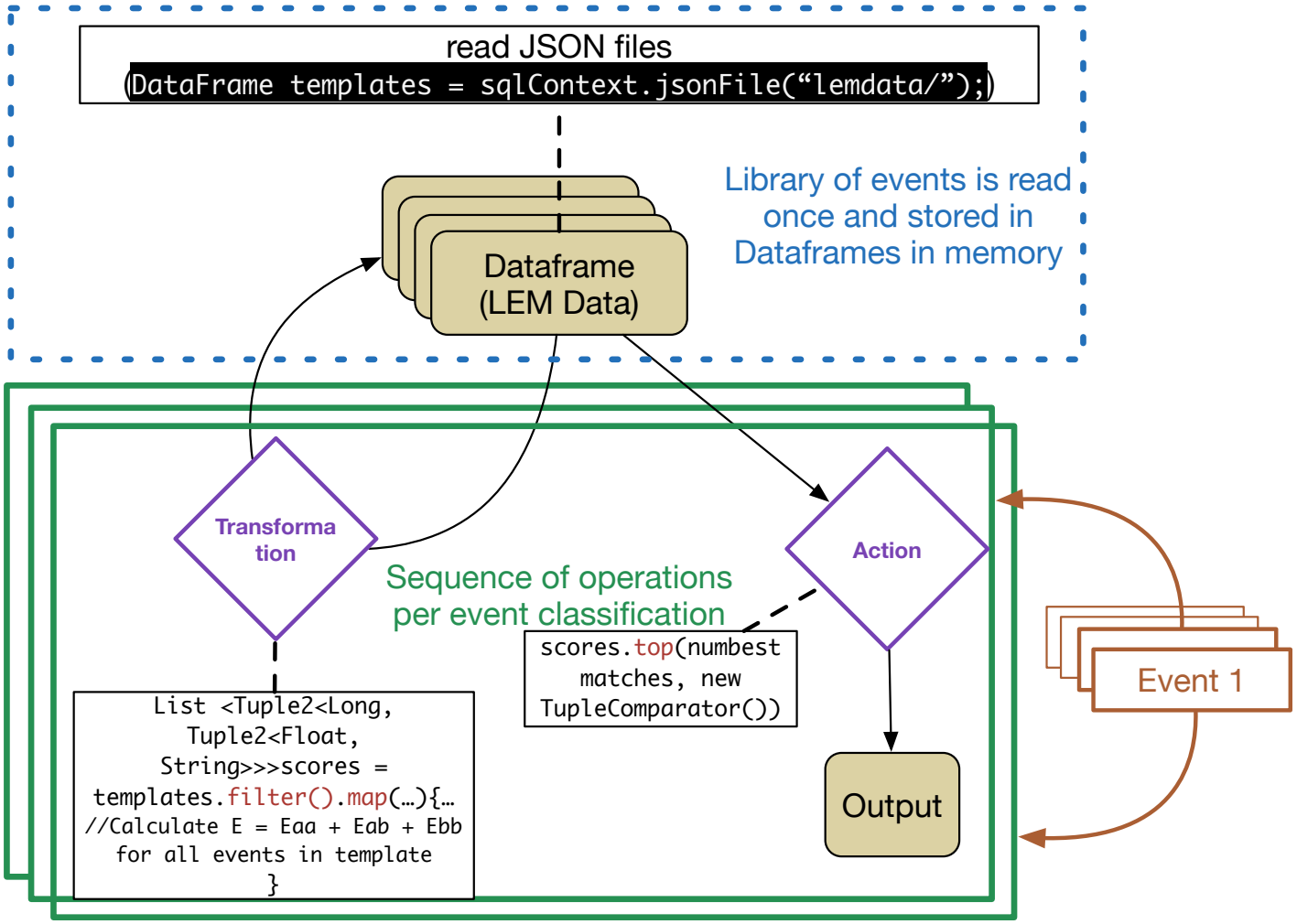
Is Spark a good technology for this problem?

- Goal is to to classify 100 detector events per second
  - In the worst case this equates to 7.7B similarity metric calculations per second using the 77M event library!
  - NOvA is thinking about increasing the library size to 1B events to improve the accuracy
- Spark has attractive features
  - In-memory large-scale distributed processing
  - Uses distributed file system such as HDFS, which supports automatic data distribution across computing resources
  - Language supports the operations needed to implement the algorithm
  - Good for similar repeated analysis performed on the same large data sets

Implementation in Spark

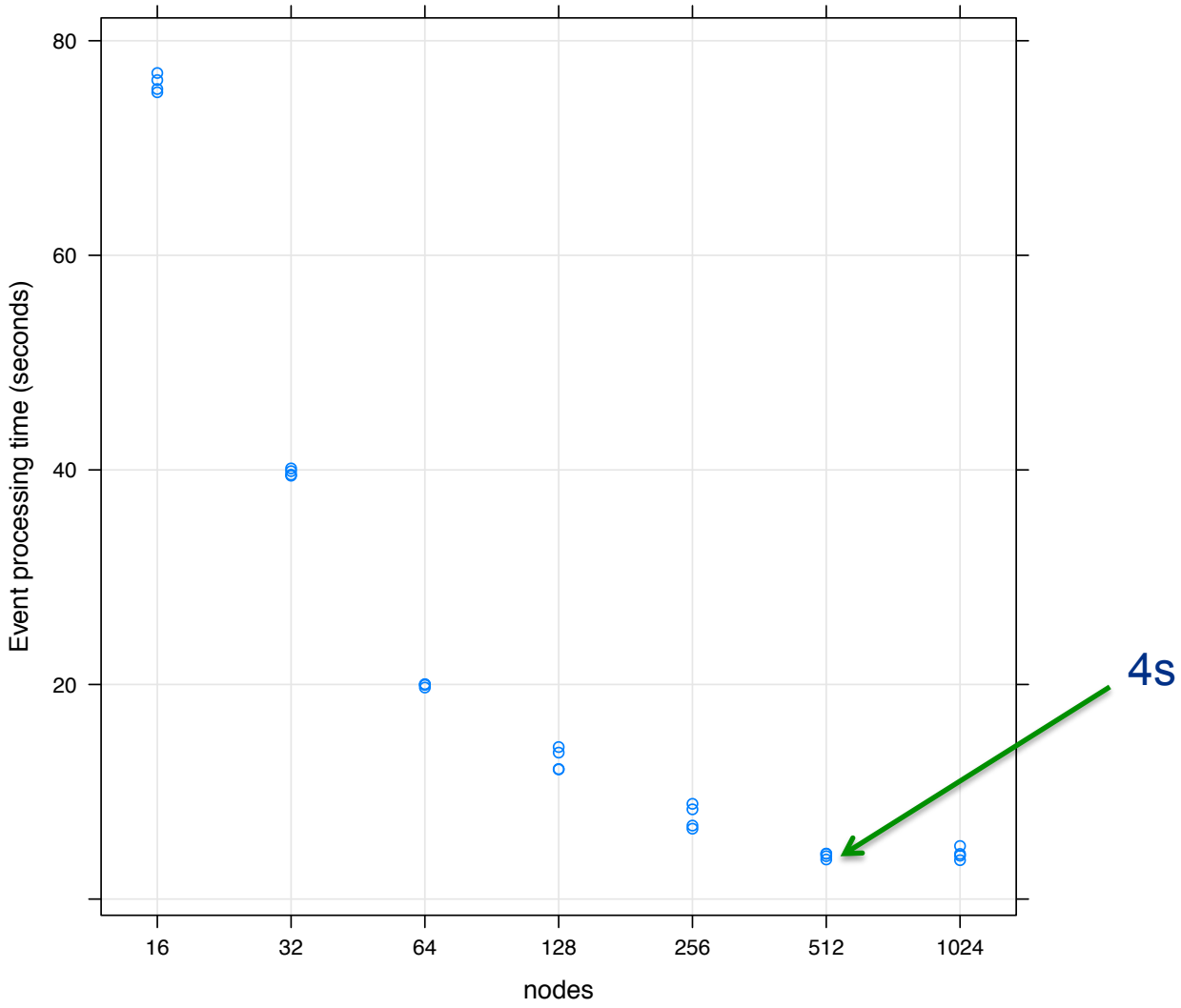
- Used Spark's DataFrame API (Java)
- Input data - used JSON format (read once)
- Transformation – create a new data set from an existing one
  - `filter`
    - Return a new dataset formed by selecting those elements of the source on which *func* returns true.
  - `map`
    - Return a new distributed dataset formed by passing each element of the source through a function *func*.
- Action – return a value to the driver program after running a computation on the dataset
  - `top`
    - Return the first *n* elements of the RDD using either their natural order or a custom comparator.

Flow of operations and data in Spark



Fermilab

Results from Cori (Spark 1.5)



Comments about Spark

- Adding a new column to the spark DataFrame from a different DataFrame is not supported
  - Our data was read in to two different DataFrames
  - Performance of *join* operation is extremely slow
- It is hard to tune a Spark system; Cori was far better than ours
  - How many tasks per node?
  - How much memory per node?
  - How many physical disks per node?
- Interactive environment is good for rapid development
  - pyspark or scala-shell or sparkR
- Rapid Evolution of this product
  - More than 4 versions since we started developing
  - Introduction of DataFrame interface, which helped to improve the expression of the problem we are solving

Alternative implementation in MPI

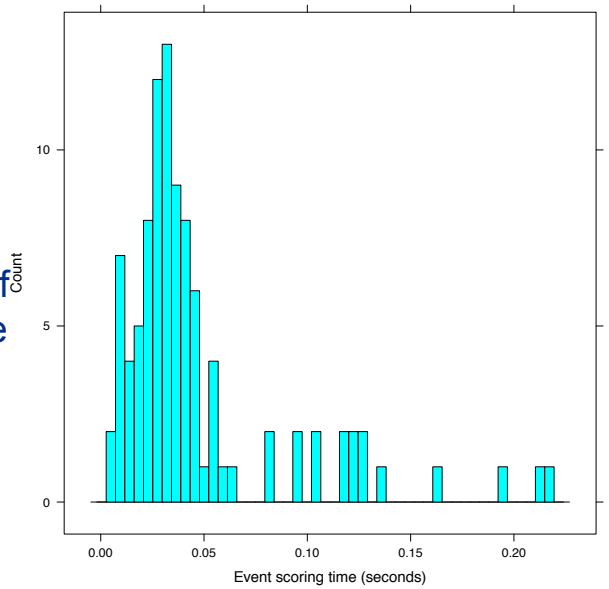
- Input data
  - data structures similar to the Spark JSON schema (read once)
  - Binary format, much faster to read: critical for development
- Data distribution and task assignment
  - Fixed by file size
- Computations
  - Armadillo (dot) and the C++ STL (std::partial_sort)
- Stages
  - `MPI_Bcast` to hand out an event to be classified
  - Filter the input data sets based on the number of hits in the event to be classified
  - Scoring and sorting to find best 10K matches
  - `MPI_reduce` to collect the best 10K across all the nodes

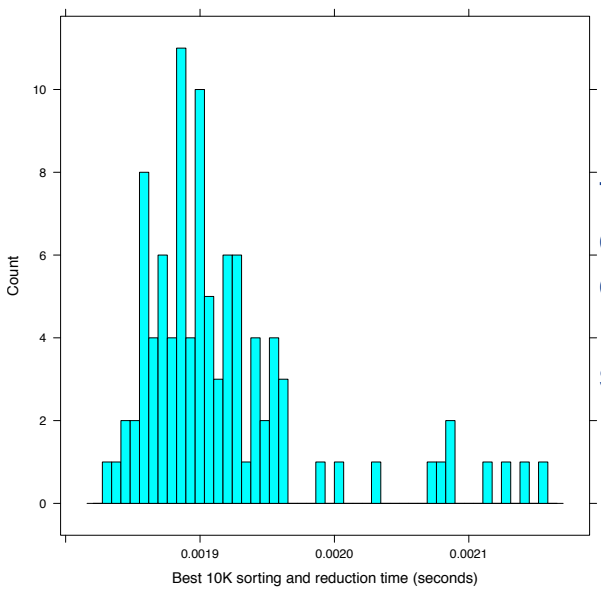
Flow of operations and data in the MPI implementation

Fermilab

Results on Cori

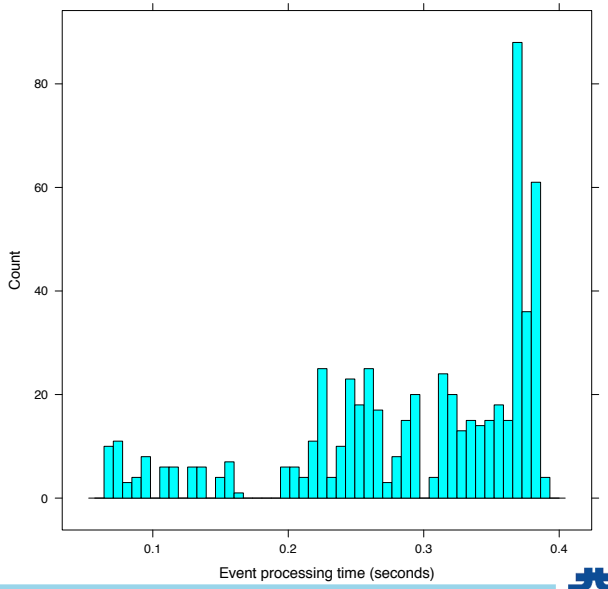
The distribution of score times for the sample of 100 events to be classified.





The distribution of time taken to obtain the best 10 thousand scores.

- Classified 100 events
- 200 cores were used by specifying 7 nodes with 32 cores
- No event took more than 0.4 seconds to classify
- The un-tuned MPI implementation using 7 nodes is 10 times faster than the Spark implementation using 512 nodes



The distribution of the total time to classify each event

Fermilab

Comparison of the two implementations

- Orchestration
  - Data placement and task assignment abstracted from user in the Spark implementation as compared with MPI
- Scaling
  - Good scaling observed in the Spark implementation without any tuning. Good scaling is possible with the MPI implementation but with a lot of work.
- Application tuning
  - Hard to isolate slow performing tasks and stages in Spark due to lazy evaluation as compared with well-defined steps in MPI
- Wrapped types
  - Use of boxed primitives have performance cost for the Spark implementation as compared with the use of primitives in the MPI implementation

Comparison of the two implementations

- Vectorized linear algebra library
  - Unavailability of the the high performance linear algebra library in the Spark implementation contributed to the slow performance of repetitive numerical computations as compared with the MPI implementation, which used Armadillo.

References

1. A Fermilab Today article
   - Nine weird facts about neutrinos by Tia Miceli, Fermilab (http://www.fnal.gov/pub/today/archive/archive_2014/today14-11-06.html)
2. Public presentations on Fermilab NOvA
   - The Status of NOvA by Gavin Davies, Indiana University (http://www-nova.fnal.gov/presentations.html)
3. Fermilab NOvA webpage
   1. http://www-nova.fnal.gov
   2. http://www-nova.fnal.gov/NOvA_FactSheet.pdf
4. Library Event Matching event classification algorithm for electron neutrino interactions in the NOvA detector
   - http://arxiv.org/abs/1501.00968
5. Spark at NERSC
   - http://www.nersc.gov/users/data-analytics/data-analytics/spark-distributed-analytic-framework/
6. Armadillo – C++ linear algebra library
   - http://arma.sourceforge.net