

Data handling with SAM and *art* at the NO ν A experiment

A Aurisano¹, C Backhouse², G S Davies³, R Illingworth⁴, N Mayer⁵,
M Mengel⁴, A Norman⁴, D Rocco⁶ and J Zirnstein⁶

¹University of Cincinnati, Cincinnati OH, USA

²California Institute of Technology, Pasadena CA, USA

³Indiana University, Bloomington IN, USA

⁴Fermi National Accelerator Laboratory, Batavia IL, USA

⁵Tufts University, Medford MA, USA

⁶University of Minnesota, Minneapolis MN, USA

E-mail: aurisaam@ucmail.uc.edu, bckhouse@caltech.edu, gsdavies@iu.edu,
illingwo@fnal.gov, nathan.mayer@tufts.edu, mengel@fnal.gov, anorman@fnal.gov,
rocco@physics.umn.edu, jan.zirnstein@gmail.com

Abstract. During operations, NO ν A produces between 5,000 and 7,000 raw files per day with peaks in excess of 12,000. These files must be processed in several stages to produce fully calibrated and reconstructed analysis files. In addition, many simulated neutrino interactions must be produced and processed through the same stages as data. To accommodate the large volume of data and Monte Carlo, production must be possible both on the Fermilab grid and on off-site farms, such as the ones accessible through the Open Science Grid. To handle the challenge of cataloging these files and to facilitate their off-line processing, we have adopted the SAM system developed at Fermilab. SAM indexes files according to metadata, keeps track of each file's physical locations, provides dataset management facilities, and facilitates data transfer to off-site grids. To integrate SAM with Fermilab's *art* software framework and the NO ν A production workflow, we have developed methods to embed metadata into our configuration files, *art* files, and standalone ROOT files. A module in the *art* framework propagates the embedded information from configuration files into *art* files, and from input *art* files to output *art* files, allowing us to maintain a complete processing history within our files. Embedding metadata in configuration files also allows configuration files indexed in SAM to be used as inputs to Monte Carlo production jobs. Further, SAM keeps track of the input files used to create each output file. Parentage information enables the construction of self-draining datasets which have become the primary production paradigm used at NO ν A. In this paper we will present an overview of SAM at NO ν A and how it has transformed the file production framework used by the experiment.

1. Introduction

NO ν A [1] is a two-detector, long-baseline neutrino oscillation experiment located 14 mrad off-axis from the NuMI [2] neutrino beam and is designed to measure the oscillation probabilities for $\nu_\mu \rightarrow \nu_e$ and $\bar{\nu}_\mu \rightarrow \bar{\nu}_e$. The probabilities will allow us to study the mass ordering of the three neutrino species as well as constrain the charge-parity violating phase in the leptonic sector. NO ν A can also measure the oscillation probabilities for $\nu_\mu \rightarrow \nu_\mu$ and $\bar{\nu}_\mu \rightarrow \bar{\nu}_\mu$ to improve the precision with which we know $|\Delta m_{32}^2|$ and θ_{23} .



The NO ν A detectors are low Z tracking calorimeters composed of alternating vertical and horizontal planes of liquid scintillator filled PVC cells. Light is transported out of the cells using wave-length shifting fiber loops that are read out by avalanche photodiodes (APDs). The far detector (Fig. 1) has a mass of 14 kton with $\sim 344,000$ 15 m cells while the near detector has a mass of 0.3 kton with $\sim 20,000$ cells. The near detector (Fig. 2) is designed to be functionally equivalent to the far detector to allow for systematic uncertainty cancellation; however, near detector cells are 1/4 the length of far detector cells. To handle the high rate environment of the near detector cavern, the near detector is outfitted with 4x faster electronics than the far detector. As seen in Fig. 3 the far detector is located near Ash River, Minnesota, 810 km from the NuMI target. The near detector is located on the Fermilab campus 1 km from the NuMI target. Both detectors are 14 mrad off-axis to create a 2 GeV narrow-band beam.



Figure 1. The NO ν A far detector is a 14 kton tracking calorimeter composed of $\sim 344,000$ 15 m cells arranged in alternating horizontal and vertical planes. It is located on the surface with a 6 in barite overburden.

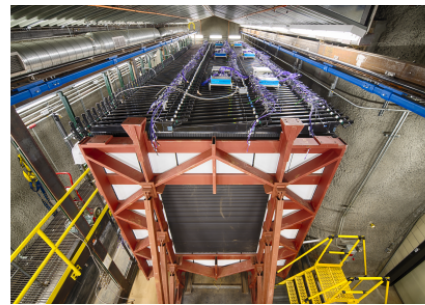


Figure 2. The NO ν A near detector is a 0.3 kton tracking calorimeter designed to be functionally equivalent to the far detector. It is composed of $\sim 20,000$ cells and is outfitted with 4x faster electronics to handle the high rate environment of the near detector cavern.



Figure 3. The far detector is located near Ash River, Minnesota, 810 km from the NuMI target, and the near detector is located on the Fermilab campus, 1 km from the NuMI target. Both detectors are 14 mrad off-axis to create a 2 GeV narrow-band beam.

NO ν A produces a large volume of data, both in number of files and total file size. This is primarily due to the high cosmic ray flux seen at the far detector as exemplified in Fig. 4. To assure that we understand our cosmic ray background and for calibration purposes, we take regular zero bias triggers. The large number of cosmic rays captured in each 550 μ s trigger window results in 5000-7000 raw files produced per day with peaks in excess of 12,000 files. These raw files must be processed through several stages to produce fully calibrated and reconstructed analysis files. In addition, large samples of simulated neutrino interactions must be produced and processed through the same stages as data. In this environment it is impossible to maintain the paradigm where all files live in central NFS storage, and jobs are run on the local grid using explicit file lists. It is necessary to move to using both local and off-site grids for Monte Carlo production and to multiple storage modes (central NFS storage, large cache disks, and tape storage).

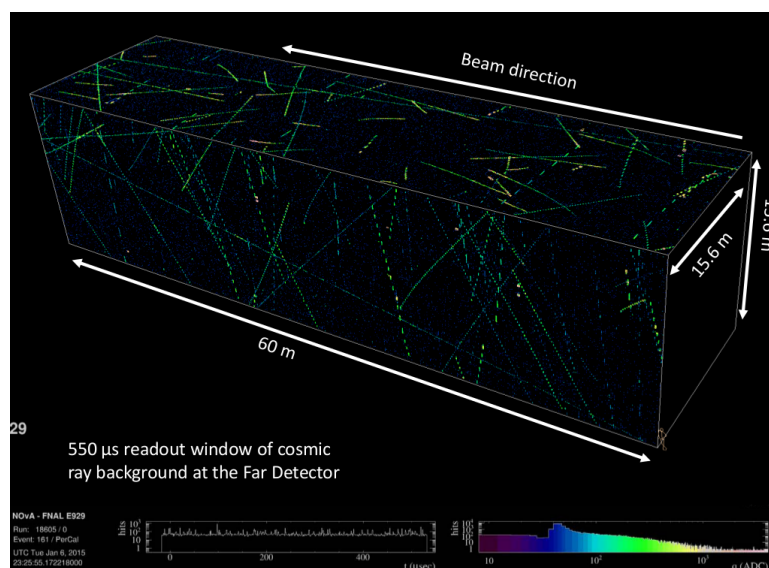


Figure 4. A typical 550 μ s trigger window at the far detector. Since the far detector is on the surface, it sees ~ 140 kHz of cosmic rays. The large cosmic ray rate, combined with the large number of channels leads to very large datasets, even though neutrinos interact rarely.

To make this new paradigm possible and as transparent to the end user as possible, we have migrated to the SAM system. SAM (Sequential Access via Metadata) [3] is a data handling system that was originally developed at Fermilab and was used by the D0 and CDF experiments. It was recently modernized to use a lightweight interface. SAM provides a number of helpful services, including indexing files according to experiment-defined metadata and keeping track of file locations. Files can be accessed by defining *datasets*, which are constructed in SAM through metadata queries. At job run time, SAM creates a *project*, defined as the list of all files satisfying the metadata query stored in the dataset from which the project was created. Many worker nodes can connect to a project and request files, which are served by SAM as *URL* links. Since each file can have multiple locations registered with SAM, SAM transparently chooses the appropriate one (for instance, choosing a file from central NFS storage rather than tape). This allows SAM to efficiently serve files by distributing easy to access ones first while simultaneously staging more difficult to reach ones.

Once the worker node receives an access *URL* from SAM, it can use IFDH tools to fetch them. IFDH (Intensity Frontier Data Handling) [4] is a set of tools developed at Fermilab which

facilitates the transfer of data from storage elements to compute nodes and back. This allows the end user to run jobs on off-site grids without needing to know the details of where the data is coming from or what the correct transfer protocol is. IFDH tools can be used as command-line programs or from within *art* using the IFDH service. The *art* framework [5] is developed and maintained at Fermilab and serves as a common framework for Intensity Frontier experiments. Since the NO ν A framework is built on *art*, the IFDH service facilitates running NO ν A jobs in a mode where they directly connect to a SAM project and all the file requests and file transfers are hidden inside the event loop.

2. Metadata Management

SAM indexes files based on metadata, so generating metadata and ensuring its correctness is paramount. To ensure that metadata cannot be separated from the files with which it is associated, we embed metadata inside our files. The *art* framework provides a metadata service which takes a vector of string key/value pairs and writes them to a SQLite database embedded within the output file. During final analysis file processing, the metadata module can also dump the metadata it contains as a C++ map so that it can be included inside non-*art* ROOT files.

The core component that manages the flow of metadata through files in various processing stages is a custom NO ν A *art* module. The metadata module is in charge of collecting metadata from several sources, merging the metadata, and then forwarding it to the metadata service to be embedded in the output files. The metadata module collects metadata from several static sources: metadata parameters can be specified on the command-line, in environment variables, or in the parameter set for the metadata module inside the job's configuration file. These sources are the entry point for all metadata concerning Monte Carlo generation jobs.

The metadata module also dynamically generates some metadata by keeping track of the number of events seen, which runs and sub-runs the events have belonged to, and the file names of input *art* files. Recording the input file names in the parentage metadata field is particularly important for automating production tasks, as we will describe below. Metadata is also preserved from the input *art* files the job receives. The metadata module uses the IFDH service to query SAM every time a new files is opened. This ensures that the metadata from the previous processing stage is preserved in each subsequent stage.

Each file type (configuration files with metadata in parameter sets, *art* files with metadata in embedded SQLite databases, and non-*art* ROOT files with metadata embedded as C++ maps) have an associated script that understands how to extract the embedded metadata and use it to declare the file to SAM.

3. Monte Carlo Production

In addition to facilitating metadata transfer into *art* files, embedding metadata in configuration files has the side effect of allowing them to be indexed in SAM. This allows us to create special configuration file datasets for use in Monte Carlo production where the configuration files are treated like input files. In this case, the configuration files serve a similar role to raw file in data, so no special handling is necessary to start a Monte Carlo generation job. These files are fully indexed in SAM and saved to dCache storage, so it is easy to search for and download a few representative files to understand how a set of Monte Carlo files was generated. The metadata module treats the job configuration file as an input file, and records the configuration file name in the parentage information of the output file. This allows us to locate, download, and examine the configuration of a Monte Carlo file corresponding to any output simulated file. Finally, careful construction of metadata provides effective versioning for Monte Carlo runs, which makes it easy to validate on new Monte Carlo features.

4. Draining Datasets

During routine processing of raw files, it is common that some files for a given day will be delayed in being transferred from the detector to be processed. Similarly, processing occasionally fails for a few files due to transient technical problems on the grid. It is crucial that any production system be able to automatically process late files (“top-up”) or retry failed jobs without having to re-run over all successfully processed files.

Tracking parentage metadata in SAM allows us to solve these problems by defining *draining datasets*, which are standard SAM datasets with a metadata query that contains a clause based on parentage information. For instance, a draining dataset used to process raw files from our proprietary raw format into an *art*-compatible format would include a clause like: `data_tier raw and not isparentof: (data_tier artdaq)`. This would select all files in the raw data tier which do not have a descendant in the artdaq data tier, where data tier refers to the type of file resulting from a particular stage of processing. As new raw files arrive from the detector, this dataset expands; as artdaq files are successfully produced, the dataset shrinks.

This functionality makes it simple to automate routine processing; however, it can also simplify pulsed Monte Carlo generation by making it possible to construct an automatic, asynchronous processing pipeline. At the start of a generation run, one can define draining datasets for the initial generation as well as all subsequent processing stages. At regular intervals, a fraction of the total number of jobs can be submitted for each processing stage, and at each submission, files will automatically propagate through this processing pipeline as successfully processed files are removed from the dataset of a previous stage and enter the dataset of the next stage. The total data throughput can be managed by tuning the number of generation jobs dispatched with each submission to reduce system strain.

5. Summary

Early in the life cycle of the NO ν A experiment, it was possible to store all files on central NFS storage and run jobs on the local grid using explicit file lists. As the scale of raw data and Monte Carlo production has grown, this paradigm became increasingly untenable. In order to move to system with a mixture of central NFS storage, cache disks, and tape, and to utilize both local and off-site computing resources, NO ν A has migrated to using tools based around the SAM data handling system. The move to SAM, IFDH, and native and custom *art* facilities have lead to the creation of several new production schemes including using datasets of configuration files as inputs to Monte Carlo generation jobs and using draining datasets with parentage information to facilitate automatic bookkeeping of files through many stages of processing. These new techniques have made it possible to automate daily processing of new raw files as well as manage large, pulsed, Monte Carlo production runs, enhancements that will be essential in enabling NO ν A’s timely production of analysis results and fulfillment of its extensive physics program.

Acknowledgements

The author acknowledges support for this research was carried out by the Fermilab scientific and technical staff. Fermilab is operated by Fermi Research Alliance, LLC under contract No. De-AC02-07CH11359 with the United States Department of Energy.

References

- [1] Ayres D *et al.* (NO ν A Collaboration) 2007 The NO ν A Technical Design Report Tech. Rep. FERMILAB-DESIGN-2007-01 Fermilab
- [2] Anderson K *et al.* 1998 The NuMI Facility Technical Design Report Tech. Rep. FERMILAB-DESIGN-1998-01 Fermilab
- [3] Illingworth R 2014 *J. Phys.: Conf. Series* **513** 032045
- [4] Lyon A and Mengel M 2014 *J. Phys.: Conf. Series* **513** 032068
- [5] Green C, Kowalkowski J, Paterno M, Fischler M, Garren L and Lu Q 2012 *J. Phys.: Conf. Series* **396** 022020