Work Needed for the protoDUNE Data Reduction Process

Brett Viren

Physics Department

BROOKHAVEN NATIONAL LABORATORY

DRA

2017 Mar 16

Data Reduction Overview

- Run the required pre-reco algorithms.
  - Only processes TPC waveform data (nominally).
  - Algs: ADC corrections, noise filtering, signal processing.
  - Consists of $\sim$20% of full-chain CPU time.
- Save a much reduced output file.
  - But, don't delete full, raw input files!
  - Remove oversampling and employ data packing.
  - Essentially, information-lossless for signal.
  - 2.5 PB $\rightarrow$ 6.5 TB (just TPC part).
- Keep up with protoDUNE/SP data taking
  - Requires $\sim$200 nodes / $\sim$4000 cores during running.

DocDB 2089 for details

Data Reduction Process Steps

1. **input unpacking:** with DAQ's raw data access library (DAL)
   - → undo the $\sim 4\times$ compression, 12 → 32 bit: $\sim 10\times$ **inflation**
2. **ADC mitigation:** stuck codes and non-linearity corrections
   - → may improve compression at $\sim$10% level
3. **noise filter:** reduce excess noise (if any)
   - → if pD/SP is **not** noisy, raw compression goes: $4\times \rightarrow 6\times$
4. **signal processing:** waveform deconvolution + signal-ROI
   - → $150\times$ total reduction factor if stop here
5. **rebinning:** exploit oversampling but respect Nyquist
   - → $400\times$ total reduction (for "rebin-3")
6. **output packing:** save in raw, fragment format
   - → relies on ROOT compression

Work needed and ongoing

For each data reduction process step, I try to summarize:

- • What exists or is needed.
- + Who is currently working on parts.
- → Possible areas for new people to get involved.

Unpacking

- Need a way to read and unpack raw data.
- Outer layer handled by `artdaq::Fragment`
- Need raw "Data Access Library" (DAL)
  - unpacks the "fragment payload" array
  - each type of fragment unique (TPC, PDS, TI, CR, maybe BI).
  - must be developed by DAQ "fragment experts".
  - data reduction nominally only unpacks TPC fragments,
  - non-TPC fragments "shunted" as-is to output.
  - RCE and FELIX fragments nominally to be identically packed.
  - needs to handle evolution of fragment schema.

\+ **List of requirements has been developed.**

  - No code development yet.

$\rightarrow$ A volunteer could start the library/repo

  - Needs close coordination with *art*, DAQ and offline experts.
  - Could start with "toy" packing as `v0` schemas.

Comment: the DAL is DAQ responsibility, but as prime "clients" offline/DRA really should participate and directly help out (where appropriate).

ADC Mitigation and Calibration

- The ADCs will need some level of **stuck-code mitigation** and **non-linear calibrations**
- \+ **David Adams**' ongoing studies of initial cold electronics test data taken at BNL.
- \+ All pD/SP electronics will undergo **production CE testing**.
  - Effort already well underway on production test lab.
  - Matt Worcester, Brian Kirby, Justin Hugon, others.
- $\rightarrow$ Effort needed toward an ADC mitigation/calibration.
  - Get involved in the CE production testing.
  - Develop some form of mit/calib parameter database
    - Could be substantial (all ADCs, maybe 4096 numbers per!).
    - If time variance is an issue, may need ongoing updating.
  - Develop code to apply these values to raw data.

Excess Noise Filtering

- No "planned" excess noise for protoDUNE.
- Strategy: assume the worse and prepare software filters for all known types of excess noise in protoDUNE.

→ Need dedicated noise hunters for very first days of data.

→ Effort for using existing software filters:

  \+ MB's noise filter in Wire Cell Toolkit, used via `larwirecell`

  +/→ Needs improvement to make modular (all in one *art* Module).

  \+ 35t's in `dunetpc` (David Adams)

  → Needs "porting" to protoDUNE?

  → Comprehensive/modular job configuration.

→ Expect need for fast deployment of code improvements

  → Need s/w infrastructure expert to assure that noise filter experts can quickly (in hours) update "keep up" data reduction process with improved new noise filter code and configuration.

Signal Processing

+ 2D signal deconvolution pioneered in MicroBooNE exists.
  - 2D needed to remove long range induction effects, although protoDUNE's grid wires reduce the size of this problem.

→ In WC prototype, port to Toolkit (Xin Qian, BV)
  - Required now'ish in any case for MB sig. proc. paper.

→ Validation with *realistic detector response simulation*

+/→ Truth metric based validation (Brook Russell, BV)
  - (details on simulation next)

→ Integrate signal processing into `larwirecell`
  - volunteers welcome, o.w. probably falls to me.

Rebinning and Packing

- Rebinning alg is trivial.
- \+ Validated with simple Gaussian.
- → More validation as a side effect when validating signal processing with *realistic detector response simulation*.
  - Brook, BV, others welcome.
- → Develop "reduced DAL" (BV, or ??)
  - Should closely mimic "raw DAL" as non-TPC data in the reduced file will still be raw fragments.

Improved Detector Response Simulation

- \+ Main improvements over current simulation:
  - Long-range induction effects.
  - Inter-pitch field variations.
  - Drift model and its statistics.
- Required for correct signal shapes, especially important for evaluating signal processing on induction plane channels.
- Needs completion for upcoming MB sig. proc. paper.
- \+ Core functionality exists, application layer a WIP (BV).
- +/→ "Truth metrics" need defining and implementing (Brook/BV).
- +/→ Drift model partly implemented (BV or ??).
- LArSoft integration not needed here, but eventually will happen.

Comment: could provide "simulated real" raw data files if we can get a proper fragment payload **packing** code from DAQ group (ie, the inverse of the "raw DAL" unpackers).

LArSoft Integration

- Data reduction algorithms are needed in many contexts.
- Some implementations are in Wire Cell Toolkit, potentially other libraries.
- Integrate WCT implementations into LArSoft via `larwirecell`
  - only part of LS that depends on WCT.
  - WCT does not depend on LS
- Use a layered approach for code (and data):

Wire Cell Toolkit and other implementations (implementation-specific data structures)
art **Service** or **Tool** (generic data structures)
art **Module** (LArSoft data structures)
`lar` executable (FCL configuration)