Proposed Initial Data Reduction for protoDUNE/SP

Xin Qian, Brett Viren

January 18, 2017(draft)

Contents

1 Basic Idea

In addition to saving all raw data to tape, this note proposes to write another output data tier which satisfies the following:

- includes data from every trigger,
- is the result of applying just the initial parts of the required production processing chain,
- retains all meaningful waveform information and
- has **a greatly reduced data size**

The rest of this note gives detail on how this basic idea can be accomplished and works out expected data reduction factors under various options and estimates the computing resources needed to run the reduction process so that it keeps up with protoDUNE/SP data production. Organizational issues are omitted.

1.1 Data reduction procedure

The main steps for the proposed **data reduction** processing are, in order:

1. software noise filtering
2. signal processing
3. redigitizing, rebinning and bit packing

The first step is largely to remove any "man made" a.k.a. "excess" noise such as may come from sources of RF that can be picked up by the sensitive cold electronics. This brings the data to a state where, ideally, only the intrinsic noise which is inherent to the cold electronics remains. This noise is well understood *a priori* of detector commissioning.

The second step replaces the waveform shaping due to field and electronics responses with a known unipolar software filter. In particular, deconvolving these response functions transforms the bipolar induction plane ADC waveforms into unipolar ones which give a direct measure of the drifting charge. This produces a waveform which may otherwise be buried under the noise and it enables for all planes an efficient form of "zero suppression", here called "region-of-interest" (ROI) selection, to be applied. The ROI selection algorithm is written so that any segment of waveform outside of any ROI is very unlikely to contain useful information about drifting charge and thus it can be safely discarded. Of course, any special analysis that requires the discarded noise segments can always go back to the full raw data.

Additional reduction can be achieved after recognizing that the nominal output of the signal process represents an oversampling in two dimensions: time and dynamic range. The 2 MHz tick of the initial digitization is faster than several characteristic, physical time scales:

- the time to drift between the wire planes is about 3 μs
- the chosen electronics shaping time is 2 μs
- the spread in collection time across all drift paths near to a given wire, which contributes to the signal shape of nearby wires of all three planes, is 1-2 μs

Issues related to stuck codes and ADC nonlinearity still warrant initial digitization of 2 MHz but after these are corrected by the software noise filtering step, the data may be safely reduced further by removing this oversampling in time. By Nyquist theorem, this can be done by resampling the waveform. However with no extra cost to storage, a slightly more preceise down sampling can be achieved by summing neighboring bins. The requirement on oversampling indicates that 2- or 3-bin summing is safe and 4-bin summing exactly reaches the limit allowed by Nyquist.

The second oversampled dimension is due to the artificially inflated dynamic range that occurs during processing. While the 12 bit ADC meets requirements, the signal processing is performed in 32 bit floating point values. Each resulting value can be "re-digitized" or "truncated" down to a 16 bit integer so that it may conveniently fit into a `short int` while still retaining a larger dynamic range than the original 12 bit ADC count. Two obvious requirements for any re-digitization are:

1. no saturation/truncation of high signals
2. the (re)digitization noise ($1/\sqrt{12}$) is negligible compared to the electronic noise.

The first requirement can be met "by construction" if the redigitization is done to 16 bits, assuming proper scaling and shifts of the floating point values are followed. Whatever extra digitization noise that is introduced at this step is $1/\sqrt{12}$ bit on a 16 bit scale while initial digitization is that on a 12 bit scale. In addition, residual electronics noise is expected to be no less than a full bit on the 12 bit scale (in MicroBooNE final residual noise after filtering is 1-2 bits depending on wire length).

On top of this safety, additional artificial dynamic range can be preserved (on average) by performing a dynamic scaling based on the local extent of the waveforms. For example, scaling by the minimum and maximum values for a given plane of channels in a given trigger.

1.2 Other data opportunities

This proposal specifically targets reducing the data from TPC wire channels (*ie* as read out and injected into artDAQ by the RCE and FELIX systems). Nominally, we propose to simply copy all other types of raw data fragments to the output file. However, we (as a collaboration) should think about what additional useful operations we should perform on SSP, Cosmic Tracker, Trigger Info and Beam Info fragments at the same time as this data reduction step.

In particular, this process may be ideal for merging in the Beam Info stream which arives out-of-band and with some some delay from the primary stream built by artDAQ from all the other data sources. This feature will require substantially more complex code, and possibly special support at the level of batch system and data delivery. If this feature is to be included in this process we would like to request assistance from fellow collaborators to implement it.

A second option that can be explored is to save the output of this process into multiple streams by selecting on trigger information. The most obvious idea is to save in-spill and out-of-spill triggers to different file sets. As described below, if the highly reductive options are used then multiple, overlapping streams can easily be tolerated even though some duplication of data would result. So, in addition to these two streams, there may be additional useful streams to define for example based on beam information (eg, beam particle ID). The reduction process should be ready to support such additional features but their details are beyond this proposal and we invite the collaboration to consider what might be suitable.

2 Outline of process

The high-level processing pipeline making up the data reduction process is shown in Figure 1. The two main components (noise filter and signal processing) been successfully implemented already for MicroBooNE and we expect they can be readily adapted for protoDUNE/SP. The ellipses indicate data files, the diamonds are algorithms and the boxes represent intermediate data in memory. Some details on each component in the above graph are given in the following subsections.

2.1 DAQ raw file

The DAQ raw file is written by artDAQ and using its nomenclature, the file contains some number of *events* each of which is composed of some number of *fragments* from a common detector trigger. The data in a fragment is in the form of a contiguous array of integers. The array is composed of two blocks: an artDAQ

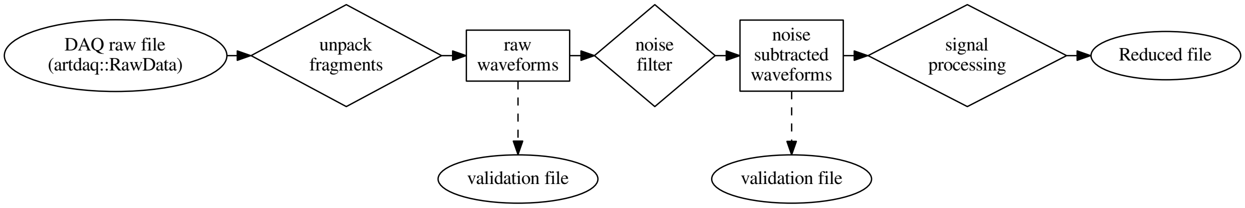

Figure 1: Outline of the data reduction process

header block and a payload block which is created by the Board Reader associated with a number of wire channels (in the case of RCE and FELIX APAs, etc for other fragments). The data is saved to a ROOT file in a schema that is based on a wrapper classes provided by artDAQ.

2.2 Unpack fragments

The two block types of the fragment array must be unpacked offline, each with its own dedicated code. The issues that pertain to this step are described here. First, the artDAQ software is nominally required for unpacking the header block part of the fragment array.

The unpacking of the payload block is expected to be more complex and more variable over time. As the DAQ makes necessary changes, new unpacking code must be applied. The unpacking of RCE channels and FELIX channels is expected to be different. After unpacking, the payload block must also be decompressed, at least for RCE data. FELIX fragments may also be compressed and if so the compression methods and parameters will almost certainly differ. The differences between RCE and FELIX will be "erased" at this point so that downstream code sees a uniform data interface across all APAs.

2.3 Raw waveforms

The unpacked data is inflated into memory. It is expected that some care is needed to manage this memory. The full 6 APA readout will be 230 MB for 12 bit samples. It must necessarily be inflated into 2 bytes for initial input and 4 byte samples for all actual processing. This requires 300 MB and 600 MB, respectively, for all 6 APAs.

2.4 Noise filter

The nature of the noise filter will depend almost entirely on what we discover during commissioning. Based on 35t and MicroBooNE experience the largest, contiguous unit of data that must be processed at one time is only 48 channels. This is determined by the span of the coherent noise in the previous detectors. If this (or a similar size) holds true for protoDUNE/SP then it gives us the option to pipeline the unpacking and noise filtering at a rather fine granularity and avoid reading in all 6 APAs at once. More likely, data from APA "plane" (given wrapping) would be in the pipeline at any time.

2.5 Noise subtracted waveforms

These are necessarily 4 byte `float` samples in memory so represent as much as 600 MB if all APAs are processed together or as little as 25 MB if one "plane" at a time is processed.

2.6 Signal processing

The signal processing centers around performing a deconvolution of the waveforms with a model of the detector response. In order to account for long range induction effects this deconvolution must span both time and space (wires). For MicroBooNE each plane can be treated independently. If this holds true for protoDUNE/SP it means the minimum unit of data is again a "plane". Keeping in mind the wrapping, this requires one quarter of the APA's channels or about only 25 MB in memory at once.

Contiguous regions of each waveform can now be selected such that collectively they are likely to contain all signals induced by the drifting electrons. In addition, the nature of the signal processing on the induction wires necessarily enhances low frequency noise which can mimic signal peaks. These regions must, in any case, be left for downstream reconstruction to deal with. In face of that reality, the ROI selection accepts some additional pure-noise regions in order to maximize the efficiency to keep all signal.

Note, the third stage of resampling, rebinning and packing are fairly trivial operations and are thus, conceptually, absorbed into the signal processing component in the graph above.

2.7 Reduced file

The final output holds just the selected regions of signal waveform undergoes one of the various rebinning, resampling and packing options explored below. This file would be archived to tape and made available to the collaboration using standardard SAM and F-FTS infrastruture tools. It would become the input for the remaining production processing chain.

2.8 Validation files

The intermediate data products can be converted to instances of the appropriate LArSoft classes in order to allow for validation of the associated process step and to facilitate that step to be used in some other context without requiring the full reduction process. These data products can also be saved to file as may be required but are not expected to be written during official production running as each alone could produce **more** data than the input raw file (shown below), or undergo only moderate compression.

3 Expected Reduction

Generally speaking, the efficacy of any reduction and subsequent lossless compression of the data depends on the amount of noise in the input waveforms. The estimates given here are made by applying MicroBooNE noise filtering and signal processing to a small sample of nine "typical" events. It is expected that protoDUNE/SP will have a noise level which is no higher than MicroBooNE's so this sample should provide reliable under-estimates of the expected reduction factors given the different options.

It is important to note that these estimates are based only on the TPC data. Data from other sources are not included and how they are processed in this data reduction step will affect the actual, total data reduction factor. Given the very large reduction factor that can be achieved for the TPC data, the other types of data in the raw data stream, if merely copied, may present a significant fraction of the overall output data volume. The need for and possible implementation of a data reduction scheme for these data should be investigated.

3.1 Options

As described above, this reduction procedure attacks on various fronts. This allows for many potential options for what to write out. Due to employing lossless compression and how it responds to information representation, it is also critical to investigate data types and data packing. The options are categorized into a five dimensional tuple:

1. tier: the major processing phase producing values in memory,

2. scaling: subsequent linear scaling of those values, if any
3. type: the data type holding one sample after scaling,
4. packing: the form of packing of samples into elments of an array and
5. compression: the applied level of ROOT compression, if any.

Each of these five categories are described below and a nomenclature is defined and used in Section 4.

3.1.1 Tiers

The data type in memory for the ADC tier is conceptually a 12 bit integer, although it may be temporarily stored in a larger `short int` or `int`. All remaining tiers are 32 bit `float` in memory.

ADC the unprocessed ADC samples

SUB noise subtracted samples

SIG result of detector response deconvolution with zeros outside of ROIs

ROI runs of zeros removed from `SIG`

RBn Sum N neighboring bins (rebin) of `SIG` followed by zero removal

The dependencies between the data tiers are shown in Figure 2. The option exists to terminate the processing, saving only the final tier to file. As diagrammed in in Figure 3, a data tier in memory may undergo a number of transformations before it is written to file. The details of the transformation depends on what options are selected. Trade-offs between features of the data and file size can be considered based on the results in Section 4.

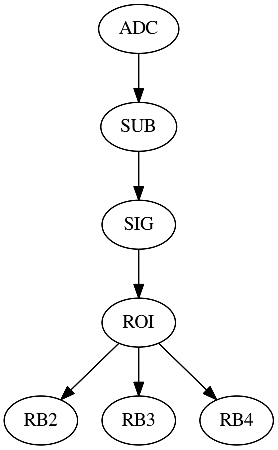

Figure 2: Enumeration and dependency of the major possible data tiers in the reduction.

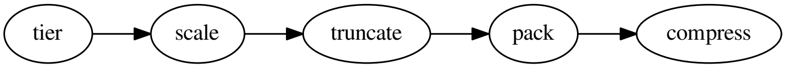

Figure 3: Each data tier is potentially transformed from memory to disk through scaling, truncation, packing and compression. Details of each phase depends on options.

3.1.2 Scaling

After a particular data tier is produced, a linear scaling may be applied to its values. The type of scaling is marked by appending one of the following letters:

n no scaling is applied

s static scaling

d dynamic scaling

Static scaling is uses predetermined scaling parameters (eg, output of SUB is already in the same units of 12 bit ADC and needs only a known baseline shift). Dynamic scaling uses the minimum and maximum of the sample values to exactly scale the data to fill a target range. Here, the target range is chosen to match the intended truncation precision (see next section). Regardless of the type of scaling, the parameters used must be saved into the output data to allow for restoration during later reading of the file. The results presented here neglect this as they contribute a negligible amount of data.

3.1.3 Truncation

Next, the scaled value can be cast or truncated into a data type other than what is used in memory. The method that is employed is marked with one of the following letters to the name in the nomenclature:

f 32 bit `float`

c 12 bit integer

s 16 bit `short int`

i 32 bit `int`

As indicated above, a proper scaling is chosen in all cases so that truncation does not clip the waveform.

3.1.4 Packing

The samples are then collected into a ROOT `TArrayI` or `TArrayF` container just prior to saving to file after having possible scaling and truncation. The different possible truncation data types may be packed into array elements in a number of ways. These three packing methods are considered:

p1 native packing, regardless of truncation data type, each sample is stored in one element of the correspondingly typed ROOT array

p2 two 12 (**c**) or 16 bit (**s**) sample values are packed across one 32 bit element of a `TArrayI`

p3 two 12 bit (**c**) samples are packed exactly into 3 bytes of elements in a `TArrayI`

In this last case, the packing progresses through the array and any final bits are padded with zero to assure 4-byte alignment.

3.1.5 Compression

Finally, data inside `TArrayI` or `TArrayF` are saved to ROOT files. ROOT provides an implicit compression feature. The compression method and compression level used are in control of the user. Here we will use the default compression method or ROOT6 and mark the level with `zN`. An `N` of 0 indicates no compression. Up to a level of 9 is possible. The higher number indicates better compression at the cost of more CPU.

4 Results

Detailed results are presented here for each data tier and applicable scaling, truncation, packing and compression. The final part of this section summarizes some of the most important results.

The absolute data sizes are presented and also normalized to a "canonical" data size (referred to as `csize` in the tables below). This is the size of the data in its uncompressed and fully packed ADC bit stream. It is this size which is used in all the data volume and throughput estimates for protoDUNE.

In the following sections the results are presented as tables showing the meaningful options for each data tier. Each table has the following columns:

format a short name following the nomenclature for the shaping, truncating, packing and compression

size the absolute size of the data on file in MB

factor the reduction factor expressed as the ratio of the canonical size to the file size. A **factor which is greater than unity means an overall reduction** while less than unity means an inflation.

time the time required to save the data to a RAM disk. It is dominated by ROOT's lossless compression time. This time should be roughly doubled to estimate the per-trigger save time for a protoDUNE 6 APA trigger.

4.1 ADC tier

This data tier consists of the unprocessed 12 bit ADC values written to disk. The data never undergoes inflation to a `float` and so the meaningful options include different packing schemes and compression levels. The uncompressed reductions factors represent merely the size of 12 vs 16 vs 32 bits per sample.

format	size (MB)	factor	time (sec)
csize	118.8	1.0	
sip1z0	316.8	0.37	0.37
sip1z1	92.4	1.29	3.12
sip1z5	80.0	1.48	8.82
sip1z9	68.2	1.74	310.30
ssp2z0	158.4	0.75	0.19
ssp2z1	46.3	2.56	1.66
ssp2z5	39.9	2.98	4.38
ssp2z9	34.3	3.47	134.02
scp3z0	118.8	1.00	0.14
scp3z1	36.6	3.25	1.29
scp3z5	28.1	4.23	3.41
scp3z9	21.7	5.47	128.94

At the strongest ROOT compression and the tightest packing, the reproduction of the compression factor of 5 which has been informally reported by MicroBooNE is reached. ProtoDUNE assumes a factor of 4 is reached which can be achieved with moderate and much faster compression. ProtoDUNE noise level is expected to be no higher than MicroBooNE so even at moderate compression we can continue to treat the compression factor of 4 as a "conservative" underestimate. Of course, actual compression will not be of the general purpose type used in ROOT so may be far better. The results of the SUB tier (Section 4.2) give an estimate that can be interpreted to be what we can expect if protoDUNE has no man-made noise.

One final observation about this data tier is that the highest ROOT compression level takes an amount of time comparable to what the processing itself requires. The details are in Section 5 and there we estimate 45 seconds / APA / trigger for processing, excluding ROOT compression. As will be shown in the following results sections, the compression time becomes negligible in the tiers which have actual data reduction.

4.2 SUB tier

The SUB data tier consists of 32 bit floats after the man-made noise has been filtered.

format	size (MB)	factor	time (sec)
csize	118.8	1.0	
nfp1z0	316.8	0.37	0.33
nfp1z1	268.8	0.44	8.06
nfp1z5	267.7	0.44	9.17
nfp1z9	266.9	0.45	14.66
dsp2z0	158.4	0.75	0.16
dsp2z1	73.8	1.61	2.24
dsp2z5	68.7	1.73	8.21
dsp2z9	60.9	1.95	113.46
sip1z0	316.8	0.37	0.38
sip1z1	66.2	1.80	2.73
sip1z5	51.6	2.30	7.32
sip1z9	42.3	2.81	330.62
ssp2z0	158.4	0.75	0.19
ssp2z1	32.8	3.62	1.33
ssp2z5	25.7	4.62	3.77
ssp2z9	21.3	5.58	144.46
scp3z0	118.8	1.00	0.13
scp3z1	27.0	4.40	1.06
scp3z5	19.7	6.04	2.81
scp3z9	14.9	7.97	107.17

Filtering involves FFT/invFFT and other operations that tent do "spread" the original 12 bits of information across many more floating point codes. This can be seen by the inflated reduction factor (factor less than unity) for the compressed `nfp1zN` formats.

This tier roughly corresponds to a "perfect" detector w.r.t. man-made noise. If we redigitize (truncate) this tier then we can get a somewhat realistic estimate for what protoDUNE if it manages to only contain suffer intrinsic noise. Even in the simple `short int` packing, the compression factor of 4 is achieved and a compression factor as good as 8 can be achieved at the highest ROOT compression.

4.3 SIG tier

The SIG data tier consists of 32 bit `float` samples with formed by deconvolving the detector response from the SUB tier. As part of the signal processing stage, the ROIs are identified and all samples which are outside of all ROIs is set identically to 0.0, but otherwise retained in the waveform.

The sig tier is the first tier where the the nominal 12 bit ADC scale is abandoned and is instead in units of number of electrons. It is also the first real candidate for a final output. For this and subsequent tiers only two scaling/truncating/packing schemes are considered. Either a full 32 bit native float is saved or it is dynamically scaled and truncated to a 16 bit `short int`. Due to the destructive (of noise) nature of setting all but ROIs to 0.0 we being to describe the `factor` in terms of overall reduction which includes compression.

format	size (MB)	factor	time (sec)
csize	118.8	1.0	
nfp1z0	316.8	0.37	0.27
nfp1z1	5.1	23.41	0.98
nfp1z5	3.9	30.74	1.63
nfp1z9	3.8	31.25	4.36
dsp2z0	158.4	0.75	0.14
dsp2z1	2.4	49.23	0.46
dsp2z5	1.5	77.71	0.82
dsp2z9	1.5	79.49	2.53

Even in full `float` form, the effect of compression on the long runs of zeros is seen and strong total reduction factor of 20-30 is achieved. Generous truncation to 16 bit `short int` gives roughly twice the reduction and then further more due to compression working on fewer bit codes. The time to write out now becomes negligible.

4.4 ROI tier

The ROI tier takes SIG and drops zero samples. As mentioned above, for these simple tests, no bookkeeping numbers are put into the output data stream. It is of course required to record the starting tick and channel information to allow unpacking. This small additional data is neglected for purposes of estimating size.

format	size (MB)	factor	time (sec)
csize	118.8	1.0	
nfp1z0	3.0	39.42	0.00
nfp1z1	2.7	44.18	0.09
nfp1z5	2.7	44.02	0.12
nfp1z9	2.7	44.02	0.14
dsp2z0	1.5	78.84	0.00
dsp2z1	0.9	135.65	0.02
dsp2z5	0.8	141.22	0.08
dsp2z9	0.8	149.46	0.55

As good as the zeros are for the compression factor, its clearly better to remove them.

4.5 Rebinned tiers

The final tier, as described above, involves exploiting the oversampling in time by summing neighboring bins. It's worth repeating that RB4 just exactly meets the oversampling requirement and so is not necessarily recommended. RB2 and RB3 is safely not destroying any information.

RB2, 2-bin rebinning:

format	size (MB)	factor	time (sec)
csize	118.8	1.0	
nfp1z0	1.6	75.87	0.00
nfp1z1	1.4	83.75	0.05
nfp1z5	1.4	83.72	0.06
nfp1z9	1.4	83.72	0.07
dsp2z0	0.8	151.72	0.00
dsp2z1	0.5	261.38	0.01
dsp2z5	0.4	271.87	0.04
dsp2z9	0.4	287.05	0.29

RB3, 3-bin rebinning:

format	size (MB)	factor	time (sec)
csize	118.8	1.0	
nfp1z0	1.1	109.78	0.00
nfp1z1	1.0	121.18	0.03
nfp1z5	1.0	121.17	0.04
nfp1z9	1.0	121.16	0.05
dsp2z0	0.5	219.51	0.00
dsp2z1	0.3	378.94	0.01
dsp2z5	0.3	394.04	0.03
dsp2z9	0.3	415.55	0.20

RB4, 4-bin rebinning:

format	size (MB)	factor	time (sec)
csize	118.8	1.0	
nfp1z0	0.8	141.45	0.00
nfp1z1	0.8	154.99	0.02
nfp1z5	0.8	154.97	0.03
nfp1z9	0.8	154.96	0.03
dsp2z0	0.4	282.82	0.00
dsp2z1	0.2	489.54	0.01
dsp2z5	0.2	508.92	0.02
dsp2z9	0.2	536.01	0.15

At its most dramatic, while still staying with the safe RB3, an overall reduction factor of 400x can be achieved. This would allow the entire reduced protoDUNE data (2.5 PB raw) to reside on a modest workstation disk of 6.25 TB.

4.6 Summary of Reduction Results

The main conclusions from the detailed results are:

- The nominal ADC compression factor of 4 used for the current protoDUNE/SP data volume estimates can be readily achieved if the detector is no nosier than MicroBooNE and if full 12 bit packing and moderate compression level is selected. In this mode, the compression time is negligible.
- If protoDUNE has no man-made noise (SUB tier) then a raw data compression ratio of 6 can be achieved with negligible compression time. At the expense of about one third more compression time then a compression factor of 8 can be achieved.

- The power of signal processing (including ROI selection) alone gives reduction factors of 30-45
- Coupled with a redigitization (truncation) to 16 bits, which still remains an artificially high dynamic range compared to original ADC counts, gives a reduction factor of 150 for the ROI tier.
- Reclaiming the oversampling in time allows for a factor of 400 reduction in the safe RB3 case.
- If we can convince ourselves that rebinning by four is acceptable then the reduction factor is more than 500.

5 Required Computing

If the reduction process does not employ finer grained pipelining then, from the discussion above, it may require a minimum of 1.2 GB at its peak. This is the bare data size just for the waveforms and occurs when both "noise subtracted" and "sigma" waveforms are in memory at the same time during the "signal processing" stage. If the pipeline operates on a per-APA basis then this number is only 200 MB. Overhead from LArSoft may add an additional 1 to 2 GB.

Based on experience running similar pipeline on MicroBooNE data we estimate that it will take 45 seconds to process one APA worth of data from one trigger. This estimate takes into account some recent speed improvements in the noise filtering, scaling from MicroBooNE to protoDUNE/SP APA channel count, and assuming modern CPUs are two times faster than are used by the particular workstation used for the benchmarking. To keep up with the peak data rate (25 Hz in spill) requires 25 Hz × 6 APA × 45 seconds/APA = 6750 CPU cores. To keep up with the average (10 Hz) trigger rate amortized over the cycle requires 2700 cores.

Typical Xeon CPUs have 10 cores per CPU and nodes typically come with 2 CPUs. This may roughly translate to needing a minimum of 2700 cores / 20 cores/node = 135 nodes running 20 jobs each to keep up with the average data rate. Taking the generous 2 GB LArSoft overhead and worse case 1.2 GB data peak leads to requiring a minimum of 3.2 GB × 20 = 64 GB RAM per node.

This RAM requirement is somewhat higher than comfortable for yesterday's commodity nodes. There are some mitigating circumstances that may reduce this but they need investigation:

- a generouse overhead for LArSoft is assumed. Testing may show this can be reduced. It may be possible to build a "pure art" level application that requires less RAM. It is also possible to build a stand-alone application which does not requires art.
- if the pipeline runs on a per-APA basis then the RAM usage is dominated by just the LArSoft overhead and the requirement drops to 48 GB.
- the above assumes all cores of the nodes are kept busy. Of course if the jobs are spread among additional nodes, the per-node RAM requirement drops.

Caveat: the estimation above is without any contingency to cover for processing fluctuations, peaks or errors in estimation. In particular, better understanding of what memory overhead to assume for the job is a driver. It also does not include any estimate for processing non-TPC data. For now, we roughly double the number of nodes, assume a one-APA pipeline and recommend a minimum of **200 nodes each with 20 cores and 48 GB RAM** devoted to *keep up* data reduction.

6 Summary

An initial data reduction which includes noise filtering, signal/response deconvolution, a region of interest selection and a rebinning and packing can provide a safe reduction factor greater than 400. With the bulk of the activity of the test data files due to cosmic ray muons, it is worth noting that the eventual DUNE far detector will enjoy a much larger reduction factor. Around 4000 cores will be required to run this process such that it keeps up with the data taking. This will provide the collaboration with a tidy data set containing all useful signal information almost immediately after the raw data is taken.

7 Acknowledgment

We acknowledge useful discussions and input on this idea from David Adams, Tom Junk, Andrew Norman, Maxim Potekhin, Dorota Stefan, Robert Sulej, Tingjun Yang, Elizabeth Worcester and Chao Zhang.