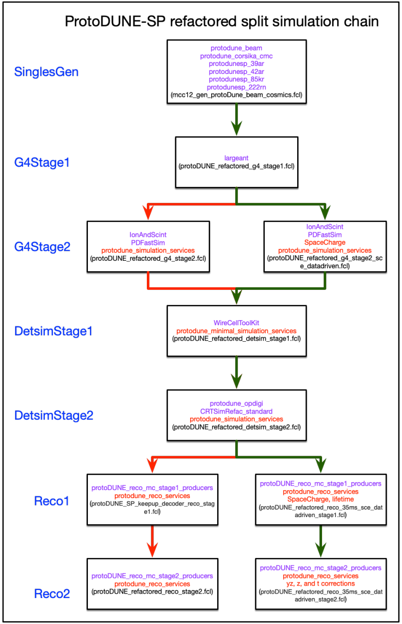
ProtoDUNE Analysis Workflow

Tejin Cai

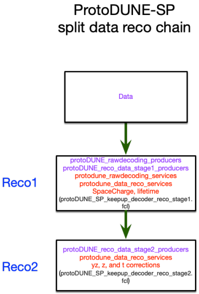
York University

July 22, 2021

Outline



Production Workflow



Production Chain

Simulation:

- SinglesGen: generate beam, cosmic rays, radiological decays
- G4Stage1, G4Stage2: particle propagation in detector volume. Two paths for stage 2– simulate space charge effect or not.
- DetsimStage1/2: Readout simulations
- Reco stages

Data:

- Only reco stages.

Reco stages

Reco1: hit,track, shower reconstruction
Reco2: apply calibration constants and perform particle identification (PID).
Reco2 contains:

- raw,recob,sim objects associated with the ART framework.
- i.e. information on beam, track, shower
- PID metrics, i.e. particle scores from various methods (template fitting, log-likelihood, machine learning methods, etc)

Reco2 contains fully reconstructed event, in ART framework objects. It is then processed by Analyzer Modules to produce standalone root tree humans can read.

From Reco to Anatuple

Anatuples are framework-independent root files produced by Analyzer Modules using the Reco2 files.
Depending on the Analyzer Module used, could have different branches.
An analyzer performs cuts, kinematic reconstructions, physics calculations on these anatuples.

Example of analysis anatuple – a record of reconstructed objects in standalone root format

```
class beamana {
public :
   TTree          *fChain;   //!pointer to the analyzed TTree or TChain
   Int_t           fCurrent; //!current Tree number in a TChain

// Fixed size dimensions of array or collections stored in the TTree if any.

   // Declaration of leaf types
   Int_t           run;
   Int_t           subrun;
   Int_t           event;
   Int_t           MC;
   Int_t           reco_beam_type;
   Double_t        reco_beam_startX;
   Double_t        reco_beam_startY;
   Double_t        reco_beam_startZ;
   Double_t        reco_beam_endX;
   Double_t        reco_beam_endY;
   Double_t        reco_beam_endZ;
   Double_t        reco_beam_len;
   Double_t        reco_beam_alt_len;
   Double_t        reco_beam_calo_startX;
   Double_t        reco_beam_calo_startY;
   Double_t        reco_beam_calo_startZ;
   Double_t        reco_beam_calo_endX;
   Double_t        reco_beam_calo_endY;
   Double_t        reco_beam_calo_endZ;
   vector<double>  *reco_beam_calo_startDirX;
   vector<double>  *reco_beam_calo_startDirY;
   vector<double>  *reco_beam_calo_startDirZ;
   vector<double>  *reco_beam_calo_endDirX;
   vector<double>  *reco_beam_calo_endDirY;
   vector<double>  *reco_beam_calo_endDirZ;
   Double_t        reco_beam_trackDirX;
   Double_t        reco_beam_trackDirY;
   Double_t        reco_beam_trackDirZ;
   Double_t        reco_beam_trackEndDirX;
   Double_t        reco_beam_trackEndDirY;
   Double_t        reco_beam_trackEndDirZ;
   Int_t           reco_beam_vertex_nHits;
   Double_t        reco_beam_vertex_michel_score;
   Int_t           reco_beam_trackID;
   Int_t           n_beam_slices;
   Int_t           n_beam_particles;
   vector<int>     *beam_track_IDs;
   vector<double>  *beam_particle_scores;
   vector<double>  *reco_beam_dQdX_SCE;
   vector<double>  *reco_beam_EField_SCE;
```

Ultimate goal: make cross section measurements

- Event selection
  - Form signal/validation/control regions
  - Constrain background though data-driven methods
- Treatment of systematics
  - cross section/event rate modifications through reweighting
  - propagate systematic variations to variable computation and event selection
- Correcting detector resolution through unfolding

What the interactive analysis facility will try to do

- Event selection and optimization.
- Background constraints.
- Efficiency calculation.
- Smearing matrix (mc)
- Unfolding to correct detector resolution.
- Combining outputs into cross section.
- Carry systematic variations along the way.

File Sizes

The sample analysis I have contains only 1 data and 1 mc files for run 5387. Data file:

- Nevents: 32600
- File size: 619M

MC file:

- Nevents: 26330
- File size: 515M

For the current production, run 5387 contains 127441 events $\sim 3.9\times$ the sample.

Including other runs:ProtoDUNE-SP-datasets

- $\sim 1.1 \times 10^6$ events.
- $\sim 32\times$ the data files.
- Would like higher mc statistics than data eventually.

Running Time

The Reco2-Anatuple chain ~ 5 hours with ~ 1300 jobs.

Analysis stage, including full reweighting of systematics ~ 10 hours with ~ 1300 jobs.

Backup

Example analysis and branches used

Direction of showers to the interaction vertex:

```
TVector3 GetShowerVector(const int ii)
{
  const TVector3 vtx(AnaIO::reco_beam_endX, AnaIO::reco_beam_endY, AnaIO::reco_beam_endZ);
  const TVector3 shw((*AnaIO::reco_daughter_allShower_startX)[ii], (*AnaIO::reco_daughter_allShower_startY)[ii], (*AnaIO::reco_daughter_allShower_startZ)[ii]);

  const TVector3 dist=shw-vtx;
  return dist;
}
```

The code snippet came from Xianguo Lu. The showers are used to reconstruct π^0.
The branches used are:

double reco_beam_startX/Y/Z (incoming beam), and
vector<double>*reco_daughter_allShower_startX/Y/Z (reconstructed object, many showers could exist hence the vector).

A lot of information is preserved in the standalone root files to allow versatile event selection.