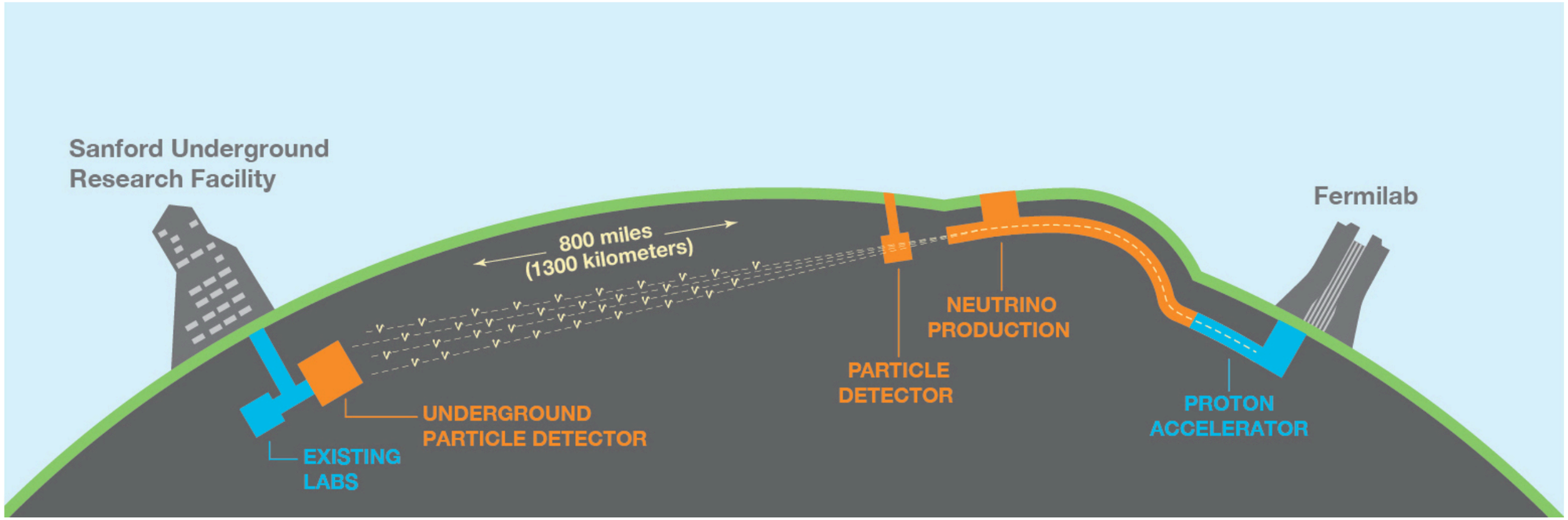
Use of TTree and plans for RNTuple in DUNE

Mike Kirby (BNL) for the DUNE Computing Consortium

6 Nov 2023 - RNTuple Format and Feature Assessment

@BrookhavenLab

Quick reminder about DUNE



neutrino experiment studying neutrino oscillation parameter (mass ordering, matter vs antimatter asymmetry, unitarity), proton decay, supernova neutrinos, and more.

four very large LAr TPC (17 kT) at 4850 ft underground in Lead, SD (Homestake Mine)

near detector onsite at Fermilab being designed (3 sub-detectors, two that move)

two prototypes at CERN - (ProtoDUNE II Horizontal Drift - ProtoDUNE II Vertical Drift)

tentative DUNE future timeline

Spring 2024 - operations of ProtoDUNE VD/HD

Summer 2024 - operations of ProtoDUNE HD/VD

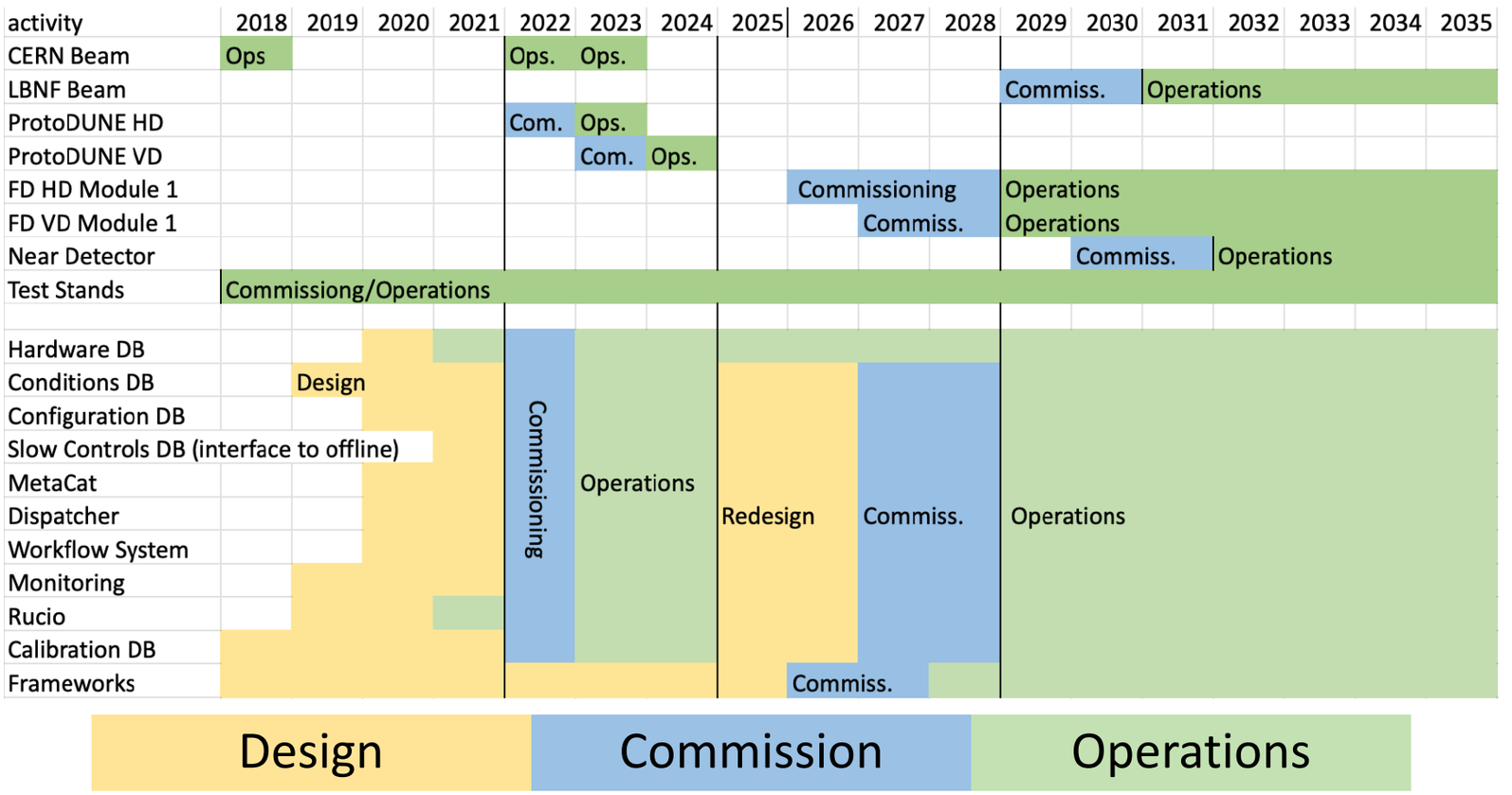
2024 - DUNE computing operations at scale with PD II data

FD HD Module 1

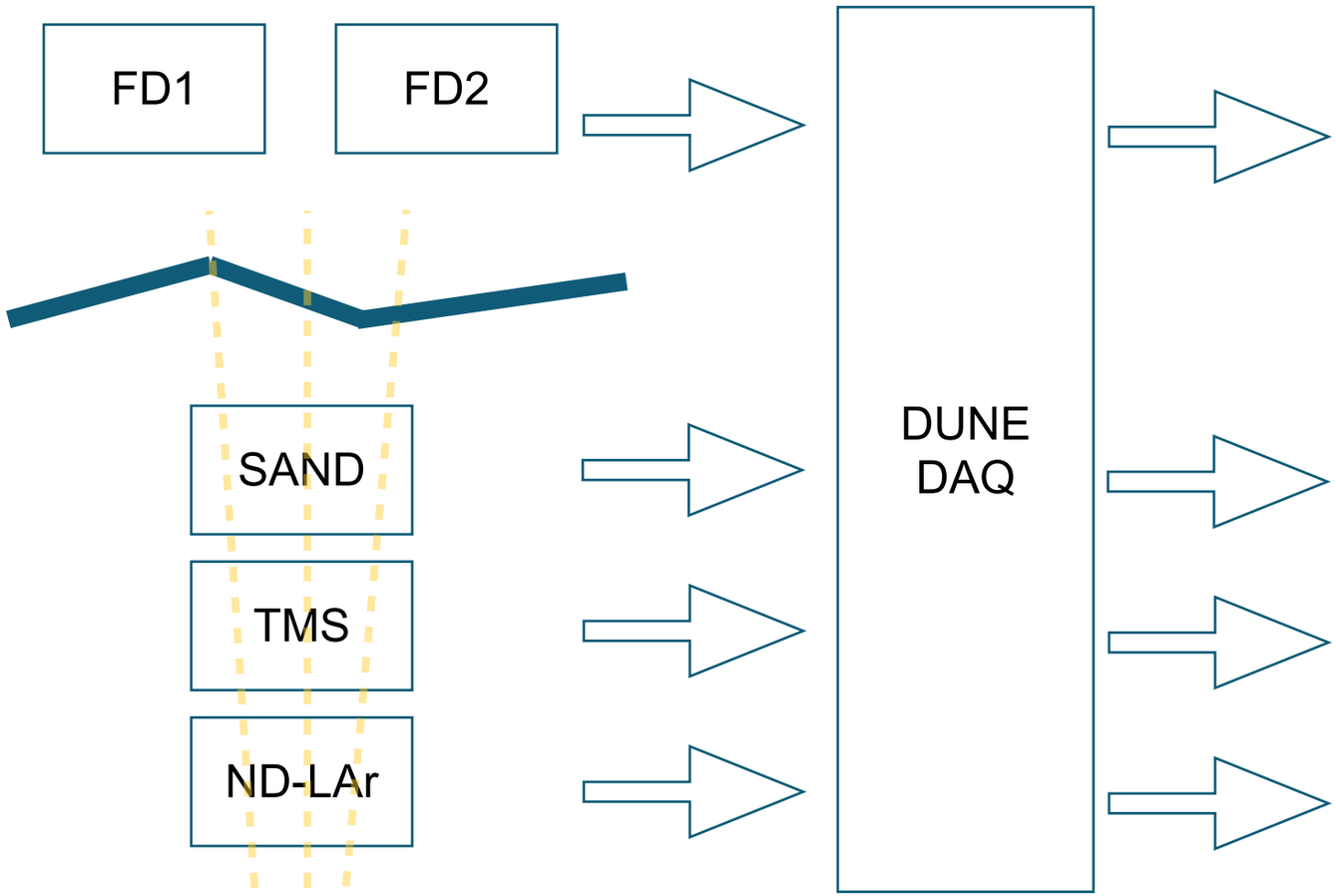
- 2027 construction
- 2028 commissioning
- 2029 physics
- FD VD Module - 1 year offset

2025-2027 - use this time for development addressing unique DUNE Challenges

Long Term Computing Project Schedule



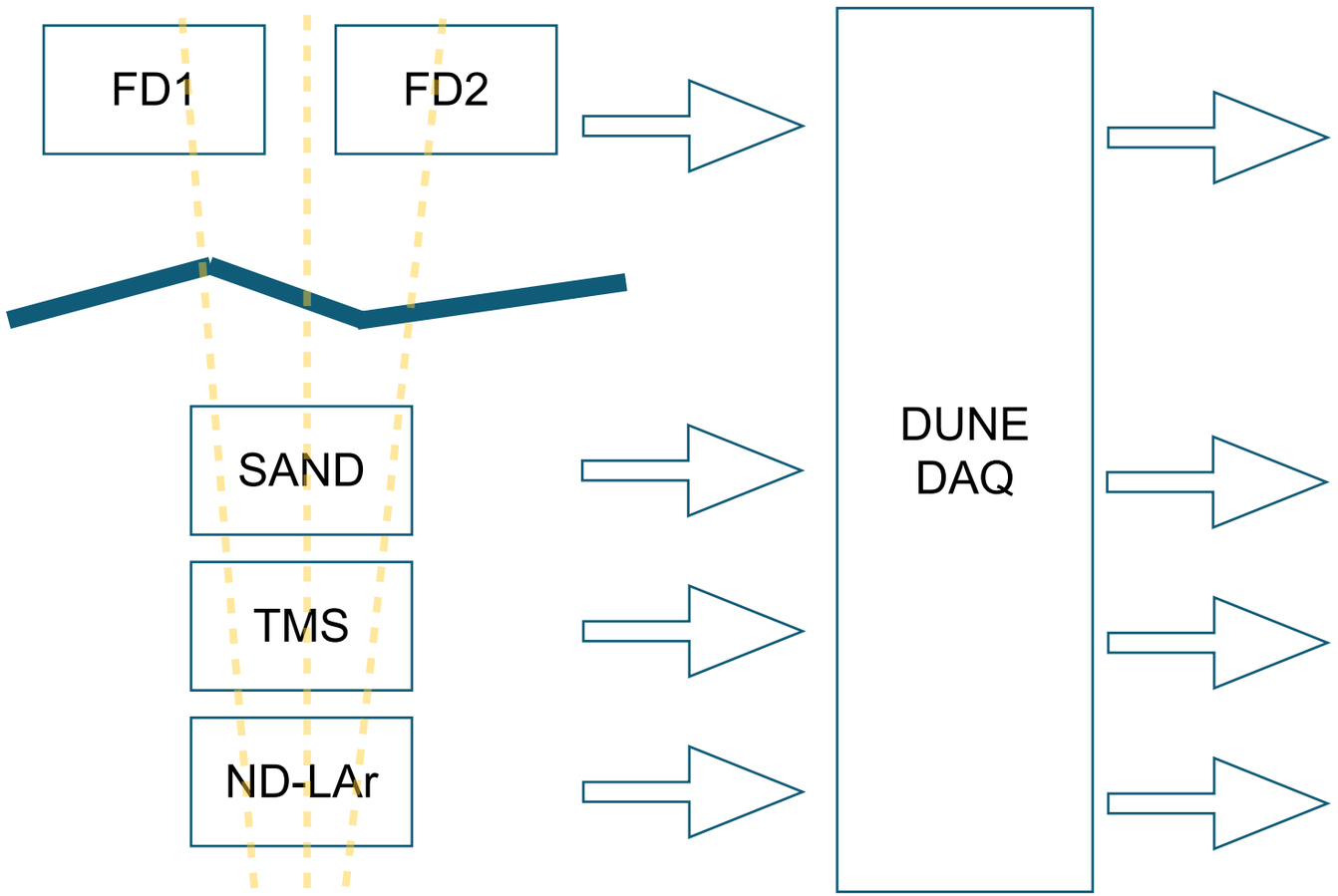
DUNE Raw Data format



- Output in HDF5 format
- format continues to evolve
- online and offline working together to evaluate and coordinate
- in general dataset for each source and a mapping from source to detector component
- adapts to many different detector configurations and dynamic data taking (readout time and components)

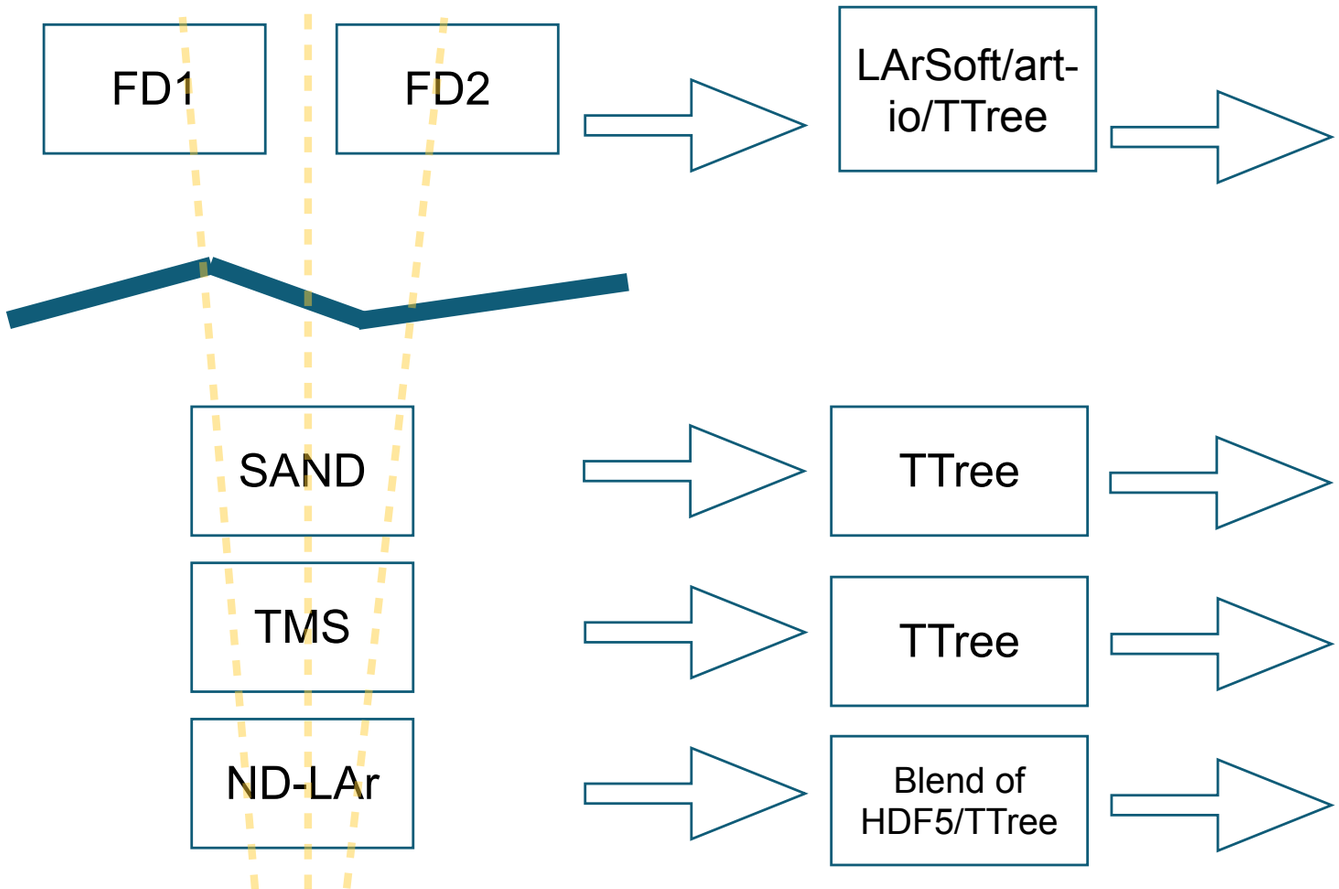
Not to scale

DUNE Raw Data format



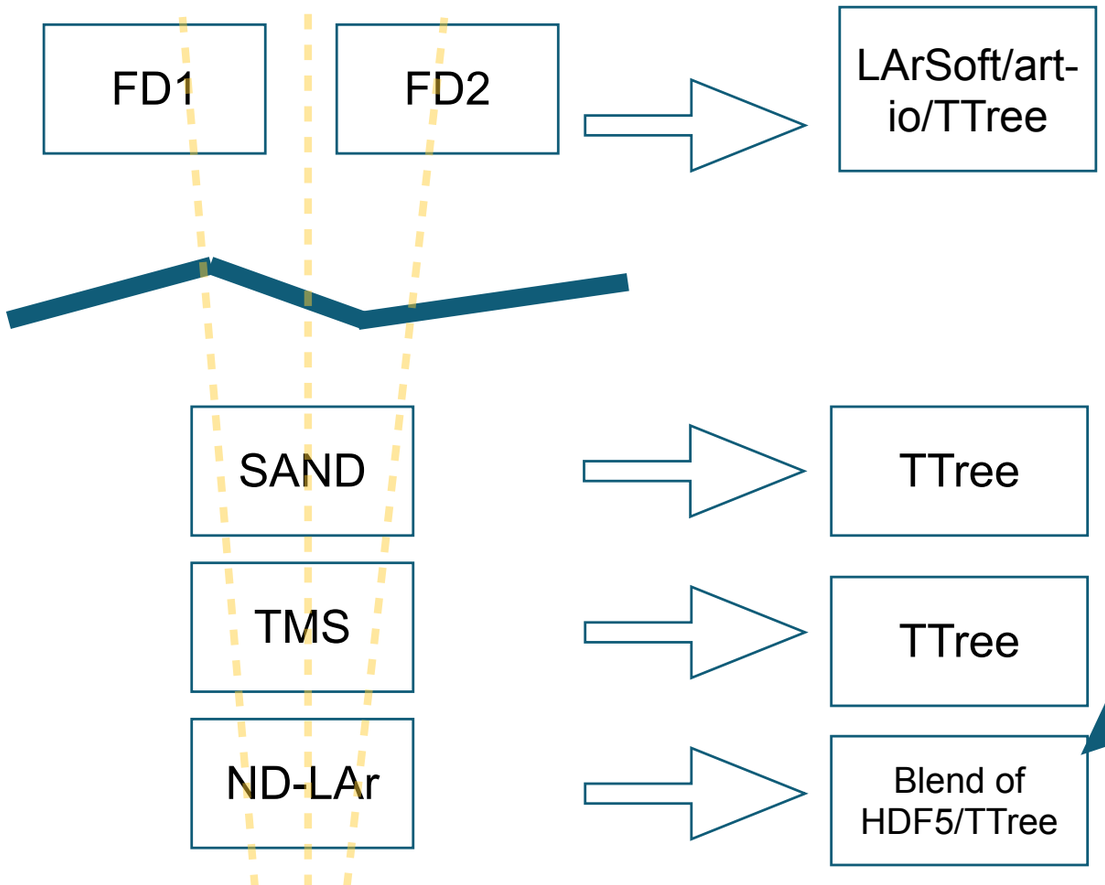
- Output in HDF5 format
- format continues to evolve
- online and offline working together to evaluate and coordinate
- in general dataset for each source and a mapping from source to detector component
- adapts to many different detector configurations and dynamic data taking (readout time and components)

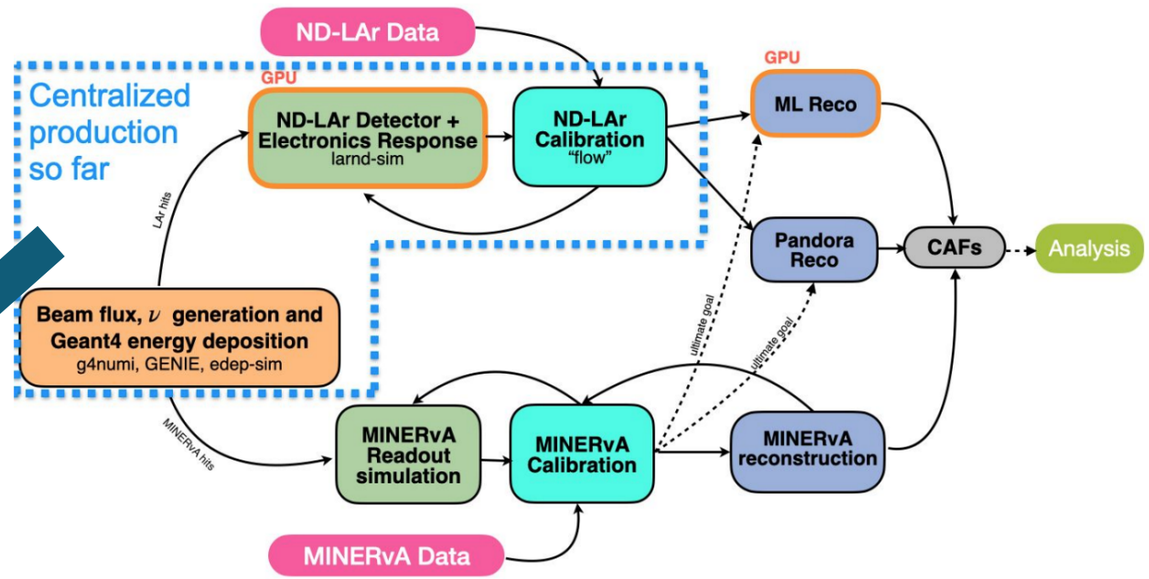
DUNE Simulation/reconstruction



- Variety of different formats
  - there is not yet a unifying EDM for DUNE
  - all based on ROOT and TTrees
- Use extensive feature set of TTree in the persisting data to files
- Ongoing work to study performance and coordinate EDM going forward

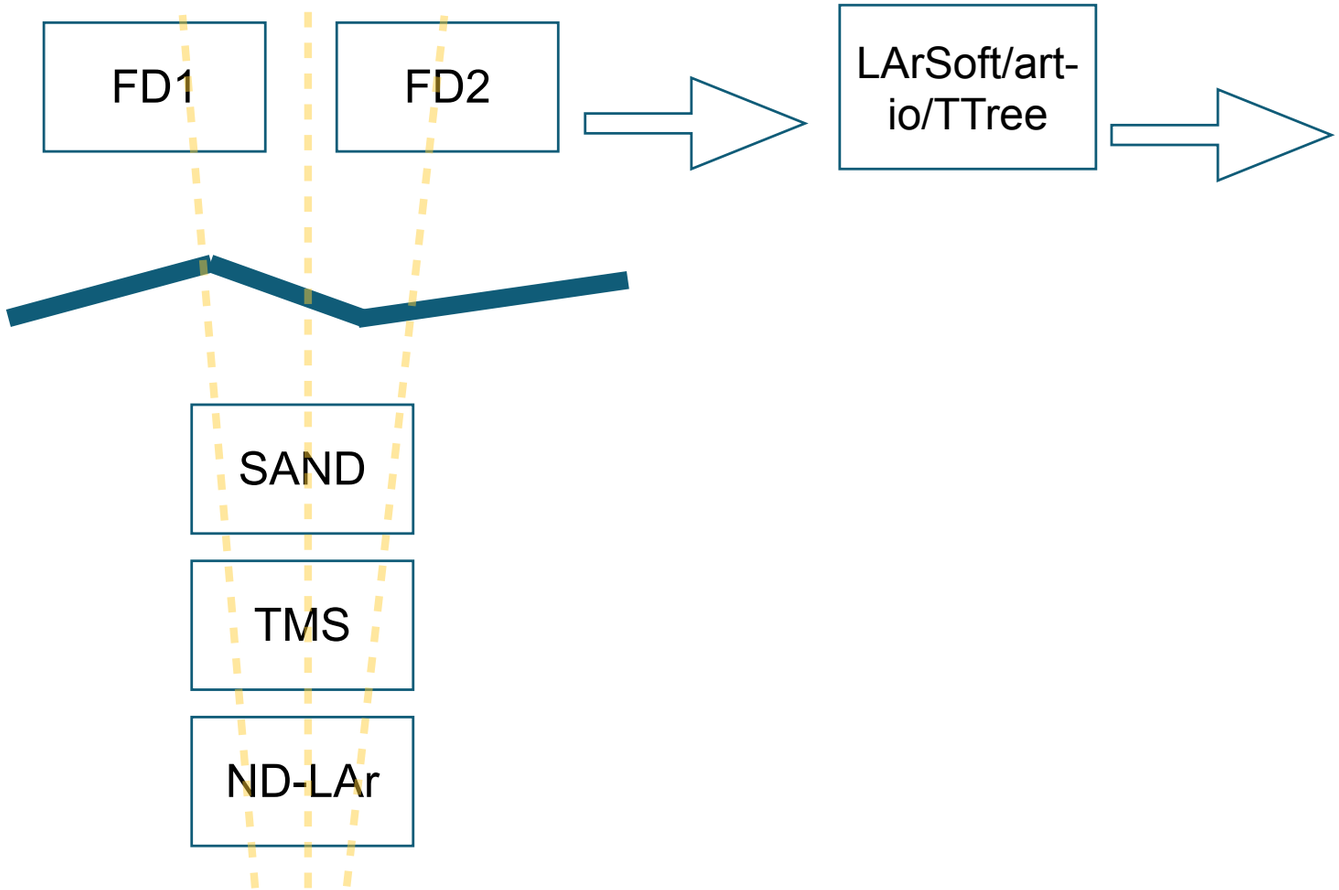
DUNE Simulation/reconstruction





Brookhaven National Laboratory

DUNE Simulation/reconstruction



DUNE takes advantage of the features in the art-root-io system
persist std::vector, std::map, std::set and our own classes
if std::set and std::map were de-supported, could work around this, but not ideal

from art data product policy “raw pointers and references are forbidden”

as well, DUNE does not persists pointers to polymorphic types - but a full survey has not been done and is on the list of tasks

“An annoyance is that it is pretty much impossible to generate a dictionary for an enumeration without auto-parsing header files. It'd be wonderful if that is no longer a problem with RNtuple.”

art::assns -- art utility for persistable pointers (really just counts inside the persisted vectors) feature of an association - can have some auxiliary data along with the pair <int>, essentially another data product. would be easily enough coded into an RNTuple?

Some small investigations into FD and ProtoDUNE data structures

Networks of raw pointers

- RNTuple supports std::unique_ptr<T>, but it doesn't support shared pointers nor (networks of) raw pointers. We'd like to understand to what extent DUNE stores network of pointers in TTree?

- From protoDUNE SP simulated file up to detector simulation:
  - TBranch does not store any “std::unique_ptr” or “std::shared_ptr”

- And, from Far Detector simulatied file up to hit reconstruction:
  - TBranch does not store any “std::unique_ptr” or “std::shared_ptr”

- But those are only 2 files investigated. Need further investigation.

Dynamic polymorphism

- Allows for storing, e.g., a collection of derived classes in a branch of `std::vector<std::unique_ptr<BaseClass>`. In RNTuple, the on-disk data will only store the base class part, i.e. there is object slicing. Does DUNE use polymorphism?
- Potential cases may include next bullet points. Needs more investigation there…
- Summary from EDepSim::PersistencyManager for geant4 is dumped as a tree. EDepSim::PersistencyManager seems to be a derived class. How objects are stored in a TTree for EDepSim could be investigated only if I had a file at my hand.
- Same is with EDepSim::UserPrimaryGeneratorAction used to generate particles which is tracked by the G4 simulation.
- It appears EDepSim uses many derived classes but it is not clear at this point how they appear in a TTree. Need further look at this point.

art::Assn

- DUNE also stores some complex objects like art association.
- Can RNTuple handle art::Assn ? Perhaps need investigation in future.

```
*.....................................................................*
*Br  393 :simb::MCParticlesimb::MCTruthsim::GeneratedParticleInfoart::Assns_largeant__G4.obj : *
*        | art::Assns<simb::MCTruth,simb::MCParticle,sim:             *
*        | :GeneratedParticleInfo>                                    *
*Entries :       10 : Total  Size=  460781171 bytes  File Size  =   12699832 *
*Baskets :       10 : Basket Size=      16384 bytes  Compression=  36.28     *
*.....................................................................*
*.....................................................................*
*Br  150 :recob::Hitrecob::SpacePointvoidart::Assns_hitfd__Reco1.obj : art:  *
*        | :Assns<recob::Hit,recob::SpacePoint,void>                  *
*Entries :      100 : Total  Size=   35154481 bytes  File Size  =    2091624 *
*Baskets :      100 : Basket Size=      16384 bytes  Compression=  16.81     *
*.....................................................................*
*.....................................................................*
*Br  123 :simb::MCFluxsimb::MCTruthvoidart::Assns_generator__GenieGen.obj :  *
*        | art::Assns<simb::MCTruth,simb::MCFlux,void>                *
*Entries :      100 : Total  Size=      11661 bytes  File Size  =        574 *
*Baskets :        1 : Basket Size=      16384 bytes  Compression=  19.06     *
*.....................................................................*
```

- art::assns -- art utility for persistable pointers (really just counts inside the persisted vectors) feature of an association - can have some auxiliary data along with the pair <int>, essentially another data product. would be easily enough coded into an RNTuple?

std::set, std::map

- These are in principle supported (sets merged, maps are almost there), but they make a data model inherently slow.

- DUNE extensively uses std::map. There are instances of std::set <int> found in those files. Is RNTuple going to support them in future?

```
*.....................................................................*
*Br  332 :simb::MCTruths_ar39__SinglesGen.obj.fGenInfo.generatorConfig :     *
*        | unordered_map<string,string> generatorConfig[simb:               *
*        | :MCTruths_ar39__SinglesGen.obj_]                                *
*Entries :       10 : Total  Size=       1164 bytes  File Size  =        197 *
*Baskets :        1 : Basket Size=      16384 bytes  Compression=   1.71     *
*.....................................................................*
```

- The comment says it's a free-form field people can put anything they want in:
- std::unordered_map<std::string, std::string> generatorConfig; ///< free-form field that can be used to keep track of generator configuration (e.g. GENIE tune)

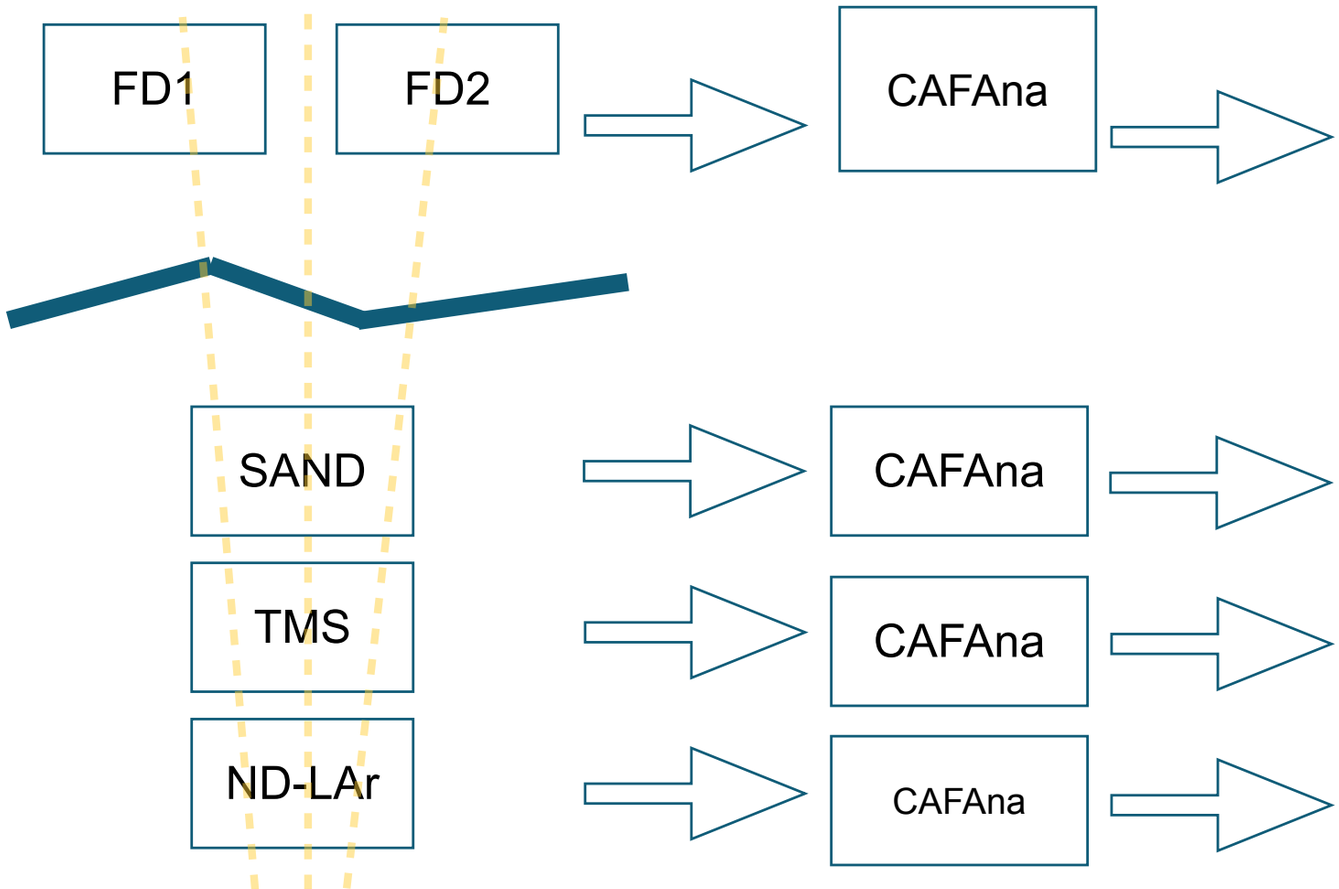
- std::set and std::map are used throughout our software to necessitate the flow of code in terms of physics requirements. ND Python uses an equivalent of STL container, hash-based set

Large Object Size

- Large events, occasionally worth of 1.7GB/10 = 170 MB, are found in TTree.
- Performance may be different for large objects. How does RNTuple behave with large events stored in a tree?

```
*.....................................................................*
*Br  279 :simb::MCParticles_largeant__G4.obj.ftrajectory.ftrajectory :      *
*        | vector<pair<TLorentzVector,TLorentzVector> > ftrajectory[simb:  *
*        | :MCParticles_largeant__G4.obj_]                               *
*Entries :       10 : Total  Size= 1772826741 bytes  File Size  =  669242335 *
*Baskets :       10 : Basket Size=      16384 bytes  Compression=   2.65    *
*.....................................................................*
*.....................................................................*
*Br   78 :sim::SimEnergyDeposits_IonAndScint__G4.obj : vector<sim:          *
*        | :SimEnergyDeposit>                                            *
*Entries :      100 : Total  Size=  586864989 bytes  File Size  =  294298266 *
*Baskets :      100 : Basket Size=      16384 bytes  Compression=   1.99    *
*.....................................................................*
```

DUNE Analysis ntuples



The one place where DUNE has the greatest alignment

flat TTrees utilized for oscillations analyses

likely most straightforward transition to RNTuple…

…but those are likely famous last words since complex data products have started to appear

Brookhaven National Laboratory

Summary

- DUNE is very much interested in working with the ROOT and RNTuple team to understand how DUNE can collaborate, learn from, and give feedback about RNTuple
- ongoing investigations in performance on multiple fronts for data format (trigger record size, extended time readouts, analysis formats, etc)
- have a nice window of opportunity between now and physics data in 2029 to map out that transition
- Huge thanks to Barnali Chowdhury, Tom Junk, Kyle Knoepfel, Peter van Gemmeren, and Heidi Schellman for most of the content of this talk (apologies for my delivery)