LArSoft data product revision

Status report

Gianluca Petrillo

Fermi National Accelerator Laboratory

LArSoft Coordinators' Meeting, October 11th , 2016

Overview

The tasks of LArSoft data product revision, phase II:

1. adoption of a recommended data structure for physics vectors and for containers *(this talk)*
2. definition of a data augmentation protocol
3. development of "view" tools to facilitate the use of data products
4. reorganisation of existing data products: *need to define the targets*

Recommended data structures

I want to get to a physics vector class recommendation:

- to be used in data products to reduce memory overhead
- to be used in algorithm to gain speed and use less memory

While the driving motivation is data products, whenever a conversion is needed at the data product/algorithm boundary, resources are wasted.

The following two requirements:

1. support for Minkowski metric ("Lorentz vector")
2. support for both single and double precision (for memory saving)

reduce the list of candidates to the following:

- `ROOT::Math::LorentzVector` and friends[1]

(the previously considered CLHEP fails the second requirement)

[1] This is *not* the traditional `TLorentzVector`.

Demonstrators

To better assess the impact of the different choices, I plan to develop two demonstrators:

1. how hard is to use the candidate library
2. what is the impact on data products

Demonstrator: usability of candidate library

The plan for the usability demonstrator:

- identify an algorithm that extensively uses vectors
  I have chosen Projection Matching track maker (PMA), that makes quite the use of 2D and 3D vectors
- run a profile to compare timing (nothing dramatic expected)
- compare the overall memory usage (nothing dramatic expected)

Demonstrator: data product

The plan for the data product demonstrator:

- identify a small set of data products to be modified
  A good candidate might be `recob::Track`*, due to the previous study. That will probably also drag other associated classes like vertices etc.*
- develop a pattern that can minimise the disruption of existing code
- check resource usage (disk, CPU, memory) as needed

I still have to work out a minimal disruption pattern...

- vectors are invasive in LArSoft code, the changes will be vast
- a one-step change is not feasible
- need a pattern that
  - allows staged update
  - encourages authors to update

Status: PMA demonstrator

- I have completed the update of Projection Matching Track Maker code to use ROOT GenVector
- it took me about 40 hours (I used a slow, safe approach)
- performed profiling (on OSX, which does not have igprof)

PMA (100 events)	`TVector`'s	GenVector
CPU time (*art*)	3886 s	3596 s
CPU time (iprofiler)	3594 s	3433 s
memory (RSS)	362.3 MB	364.5 MB

The difference is 5–10% (depending on the metrics we want to trust). 125 s are directly from `Tvector`'s `TObject`ness.

I have learned some things about ROOT GenVector...

ROOT GenVector: lessons learned [1/2]

GenVector is a library in ROOT:

- the feeling is not completely dissimilar from `TVector2/3`
- GenVector distinguishes two different concepts:

  position ("point"): subject to translations

  displacement ("vector"): transparent to translations

  The latter is pretty much equivalent to `TVector`'s
- the two vectors have different addition operator behaviours:

left op.	*right op.*	*left + right*	*left – right*
point	point	*n/a*	vector
vector	point	*n/a*	*n/a*
point	vector	point	point
vector	vector	vector	vector

This makes sense, but design requires some thinking.

ROOT GenVector: lessons learned [2/2]

continued...

- the algorithm to compute centroid of points is less straightforward than I would have liked (goodbye `(a + b) * 0.5`)
- no `operator[]` is provided: component increment and dynamically choosing on which coordinate to act become more complicate
- likewise, no conversion to array/pointer of **`float`**/**`double`**
- dot product is available only as member function (no `v1*v2`)
- there are minor notation changes (e.g. `Mag()` $\rightarrow$ `R()`)

There are a lot of interesting features, like storing the vectors in different coordinate systems etc. But these are not game-changing. The lack of conversion to pointer *can* become an issue.

Summary

- vector replacement investigation is ongoing
- I expect it to be completed by the next coordination meeting
  - need to concentrate on the data product side now
- next is to identify data products to test the augmentation strategy:
  - `recob::Cluster` might be split in two
  - `recob::Track` is the perfect candidate, but it needs basic design (which I hope to be bootstrapped by FNAL Reconstruction group)

Backup

The following slides from Coordination meeting on August 30th , 2016 describe the other parts of the project.

Augmented data!

Two trends:

- larger data structures consume more memory
  - you end up loading lots of data even when you don't use it
- larger data structures consume more time
  - a producer is forced to fill all data even when it does not apply
- fragmented data structures are harder to use
  - need synchronisation between different data products

One solution:

- large data is fragmented is smaller storage units
- data is accessed via a unifying interface ("façade")

This will require some assumptions, that need to be agreed upon.
Overhead must be carefully assessed (*ideally, there should be none*).

Data as a view

LArSoft still lacks classes exposing different components of the reconstruction as a unity; for example:

- a track that contains its vertices, clusters and hits
- a particle set that includes showers and tracks
- an interaction, with vertices and particles (`PFParticle` tries)

Technically,

- might be implemented as an extension of the façade interface
- might benefit from Saba Sehrish's work on association navigation

But, more important than technical implementation:

> *Interfaces should talk the language of physics.*
>
> — Robert Kutschke

Reorganisation of data products

And, last and first:

- it's time to reconsider the choices from two years ago:
  - which data products did not age well?
  - what was left behind to be completed?
- the purpose, meaning and content of single data products should be discussed:
  - the track object is as bad as it was two years ago
  - `recob::Shower` needs a soul
  - `recob::Cluster` might be split
  - `recob::Vertex` needs uncertainty
  - and particle ID (MVA),
  - truth information (*nutools*)...