Multithreaded LArSoft Services

Michael Wang with Erica Snider and Paul Russo

06 Nov 2018

Introduction

- As of version 3, the art framework has multithreaded processing capability:
  - Specifically: the ability to run multiple events within the same SubRun in parallel
- Unfortunately, LArSoft was not designed from the start with thread-safe operation in mind
- We (Erica Snider, Paul Russo, and I) have recently, therefore, begun looking into refactoring the LArSoft services with the goal of making them thread-safe.

Thread-safety in art v3 services

- In art v3, the approach to making many of the services thread-safe was to simply “drop in” a “RecursiveMutexSentry” into the member functions:

```
void
  FileCatalogMetadata::setMetadataFromInput(collection_type const& mdFromInput)
  {
    RecursiveMutexSentry sentry{mutex_, __func__};
    if (mdToInherit_.empty()) {
      return;
    }
    if (!imd_) {
      imd_ = make_unique<InheritedMetadata>(mdToInherit_, mdFromInput);
    } else {
        .
        .
        .
```

- This permits only one thread at a time to have access to the member functions, ensuring thread safety.

Some challenges in LArSoft

- Unfortunately, for most LArSoft services, except perhaps the most trivial ones, we cannot simply take the approach shown in the previous slide of simply dropping in the sentry.
- Take a few examples:
  - DetectorClocksService: the private member “fClocks” needs to be updated across event boundaries. Separate clocks need to be maintained when processing concurrent events.
  - LArFFT: the public member fFFT (a ROOT TFFTRealComplex object) also needs to be updated across events; in fact, even across wires within a single event. So some restructuring needs to be done for perform deconvolution of wire signals across multiple wires in parallel.
- Another issue is that although multi-threaded art v3 exists, LArSoft has not been built against art v3. So there is currently no testing environment for multithreaded LArSoft development work.

Facing the challenges and moving forward

- To address the previous issue, with Paul's help, we have set up a minimal standalone testing environment that has access to the ART services, and in which we can launch multiple threads. This will allow us to begin developing and testing prototype multithreaded LArSoft services.
- Based on feedback from Gianluca P., we will start with the LArFFT service which is a good candidate on which to focus our efforts in developing an initial thread-safe prototype. The task of deconvoluting the wire signals from the electronics response, etc., which relies on this service, will immediately reap the benefits of multithreading capability.