

# A Few Comments

John Freeman, 3/8/16

# Boundary Conditions

- We've discovered that, given an uncompressed spill, the RAM needed to process it with an art module is twice the size of the spill
  - Haven't proved this, but almost certainly even worse when the spill is compressed
  - The upshot? You won't be able to process everything in RAM on a grid node if you have a spill >1 GB
  - Unclear how much heavy swap space usage would slow things down, but it's probably not pretty
- Coincidentally, TBuffer, used to pass artdaq::Fragment objects between eventbuilders and aggregators, can't hold more than 1 GB – again, a Root issue
  - <https://root.cern.ch/phpBB3/viewtopic.php?t=9469>
- Other?

```
//-----
void EventBuilder::loadDigits_(LariatFragment * & LArIATFragment)
{
    if (fTreeIndex != fNumberInputEvents) {
        artdaq::Fragments * fragments = getFragments(fFragmentsBranch, fTreeIndex++);

        if ((*fragments).size() > 1)
            throw cet::exception("EventBuilder") << "artdaq::Fragment vector contains more than one fragment.";

        artdaq::Fragment frag = fragments->at(0);
        const char * bytePtr = reinterpret_cast<const char *> (&*frag.dataBegin());
        LArIATFragment = new LariatFragment((char *) bytePtr, frag.dataSize() * sizeof(artdaq::RawDataType));

        // get SpillTrailer
        LariatFragment::SpillTrailer const& spillTrailer = LArIATFragment->spillTrailer;

        // get timestamp from SpillTrailer, cast as uint64_t
        fTimestamp = (static_cast<std::uint64_t> (spillTrailer.timeStamp));
    }
}
}
```

- Shouldn't be too difficult to handle spills split into separate events
- We could modify EventBuilder::loadDigits\_, called on each event, to “sit on” events and stitch together the spill, then passing it to LariatFragment constructor