LArSoft: simulation and reconstruction for Liquid Argon TPC

Gianluca Petrillo, on behalf of LArSoft project

Liquid Argon TPC R&D Workshop
Fermilab, July 8th , 2014

What is LArSoft

LArSoft:

A software suite for simulation, reconstruction and analysis of Liquid Argon TPC data

- knows the geometry of the detector
- drives event generation (GENIE: ν beam interactions; CRY: cosmic rays; "particle gun")
- contains detector geometry and runs detector simulation based on GEANT 4
- simulates the digitization of the readout
- applies reconstruction algorithms to identify clusters of energy deposit, tracks, particle showers and interaction vertices

art

LArSoft is based on art framework:

- simplified branch of CMS software, in use by many "intensity frontier" Fermilab experiments (NOνA, muon $g-2$, Mu2e)
- event data model: immutable "products" stored in ROOT files
- provides I/O of products and tracking of their history
- provides an event loop where user code ("modules") is run
- ... and a lot more

art team has been attentive and supportive to our needs:

- information can be obtained via mailing list, art-users@fnal.gov
- issues can be opened in the tracker
- LArSoft has a seat in the weekly art "stakeholders' meeting"

Understanding art

Understanding of how art works helps creating more efficient code: the art workbook, being updated, is a precious source.

Who is LArSoft

- LArSoft is actively contributed by ArgoNeuT, LArIAT, MicroBooNE, LBNE, with the recent addition of LAr1-ND.
- LArSoft is the product of the collaboration between these experiments and Fermilab
- facilitates the convergence of expertise and ideas into one place, benefiting all the experiments
- experiment members, LArSoft and art experts meet every week in two meetings:

  Partners' meeting: general management, reports and requests and updates on the current activities

  Librarians' meeting: updates on software algorithms and new features, discussions and decisions of major changes to the code
- mailing list larsoft@fnal.gov provides a quicker response

The organization is not crystallized, and we learn week by week.

LArSoft structure

- LArSoft is hosted in GIT repositories:
  - 10 repositories contain general software, e.g.
    - general "services" (geometry and material database, file transfer service, timing coordination...)
    - utilities (event generation, Geant4...)
    - reconstruction algorithms (hit, cluster, track, shower, vertex...)
    - a Pandora framework box
  - experiment-specific repositories (uboonecode, lbnecode... lar1ndcode?), e.g.
    - digitization code
    - configuration files
- compilation of code across GIT repositories is performed by cmake and handled by the MultiRepository Build (mrb)
- each repository has a test area and some libraries: EventGenerator, LArG4, HitFinder, VertexFinder...

LArSoft contributors

The contributors creating LArSoft content are the users themselves! LArSoft is a community effort with shared goals and responsibilities.

- users can push directly in the develop branch
- new code is expected to come with tests and documentation
- we envision the figure of librarians:
  - ⇒ nominated by their experiment
  - choose one or more libraries, small(ish) parts of LArSoft code
  - expert/informed of that library
  - communicate status/news of the library across experiments
  - verify existence of running tests & documentation in the new code
  - oversee validation and debug of the code

  They provide monitoring of LArSoft code base at a fine granularity level that couldn't be provided by a single person.

Work is still in progress to define the librarian role and provide them with information, tools and instructions to fulfill these tasks.

LArSoft activities

LArSoft keeps a list of feature and bug fix requests at:

https://cdcvs.fnal.gov/redmine/projects/larsoft/issues

#	Tracker	Status	Priority	Subject	Assignee	Updated
6510	Bug	Assigned	Normal	uboonecode zsh compatibility and relocatability	Herbert Greenlee	06/24/2014 10:43 am
6509	Support	Feedback	Normal	xrootd support in larsoft root	Lynn Garren	07/07/2014 11:05 am
6486	Bug	New	Normal	No complaint when you don't register a module		06/17/2014 01:28 pm
6452	Support	Accepted	Normal	Large memory use by fuzzyCluster	Gianluca Petrillo	07/02/2014 06:41 pm
6432	Support	Resolved	High	Merging TimeService into the develop branch	Gianluca Petrillo	06/28/2014 03:09 pm
6401	Bug	Assigned	Normal	Review the data product classes structure	Gianluca Petrillo	06/24/2014 11:08 am
6394	Bug	Assigned	Normal	Verify that association query objects are created outside tight loops	Gianluca Petrillo	06/24/2014 11:09 am
6363	Feature	New	Normal	Geometry method to find the slope (dTime/dWire) in a plane given the plane number and slopes in the other two planes		05/28/2014 03:08 pm
6362	Feature	Assigned	Normal	Pre-calculate trig functions for wire angles	Gianluca Petrillo	07/08/2014 11:21 am
6355	Milestone	New	Normal	Deliver successfully to the Microboone Simulation Challenge		05/27/2014 03:24 pm
6298	Support	Resolved	Normal	New root behavior of TH1 with nbins <= 0	Herbert Greenlee	05/19/2014 01:12 pm
6191	Support	Resolved	Normal	Requst to patch genie v2.8.0		05/12/2014 10:38 am
6066	Milestone	Resolved	Normal	Workshop - LArSoft test/validation/algorithm workshop	Erica Snider	07/08/2014 11:28 am
6065	Milestone	New	Normal	Continuous Integration - Beta release of test/validation system based on Build Service and CI	Patrick Gartung	07/08/2014 11:27 am

Beside supporting and maintaining the software and the other users, the principal activities of the LArSoft project are currently:

- release management and software deploy
- creation of a Continuous Integration and testing infrastructure
- optimization of resource usage
- revision of the software and data architecture
- drawing up of an article about LArSoft, to be submitted to NIM

Release management and software deploy

Lynn Garren is the release manager for `LArSoft` repositories:

- produces a weekly release from the content in `develop` branch
- integrates underlying software updates (e.g., `art`, `nutools`)
- tests the release, fixes it, hammers and rubs it until it shines
- makes sure the binary packages (compiled for SLF5 and SLF6, in profiling and debugging modes) are delivered into the common grid area (`/grid/fermiapp/products/larsoft`)
- last release: `v02_02_00` (July 1st)

The experiments are expected to have release managers for the experiment packages:

ArgoNeuT Tingjun Yang

LBNE David Adams[1]

MicroBooNE Eric Church

[1] *waiting for signature*

Continuous Integration Service and testing (I)

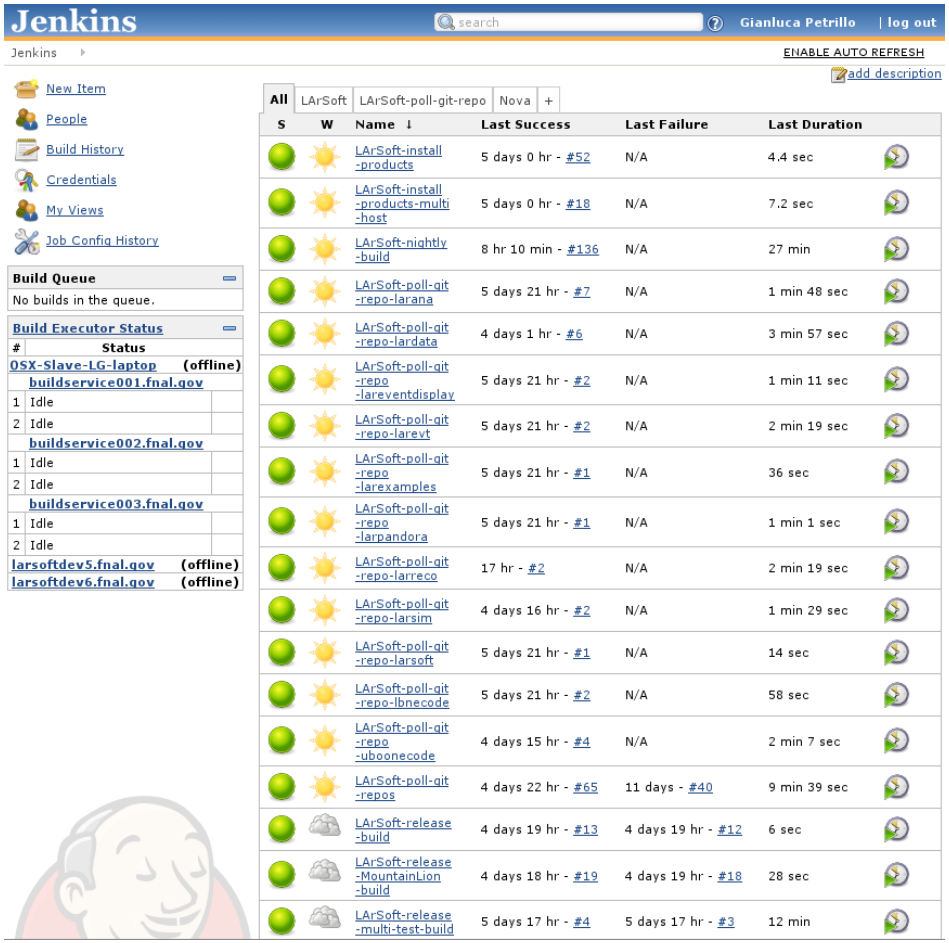
Integration:

the combination of many independent changes into one software suite

Following the model of many experiments, we need a system to:

- constantly monitor the quality of the code in the repositories
- automatically perform tests on every “push” into repositories
- perform thorough tests every night and on a release
- be able to communicate to humans when action is needed
- maintain a history of the past events
- allow any developer to easily add tests

Continuous Integration Service and testing (II)



The Jenkins web interface at

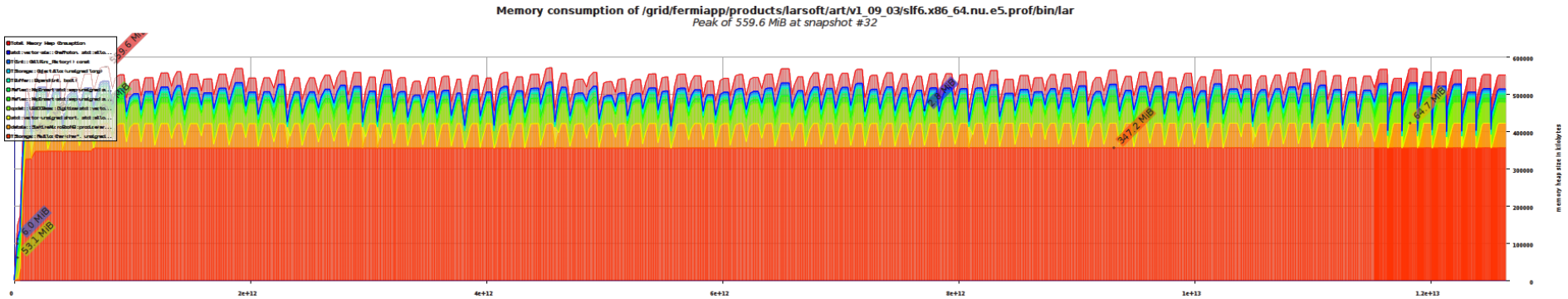
http://buildmaster.fnal.gov:8080

- a set of requirements is being finalized
- the tool of choice is Jenkins (used also by CMS)
- it allows automated builds, triggered by changes in the repositories or by explicit requests
- we have a server and a few “slave” nodes!
- Marc Mengel and Partick Gartung are leading the effort, including Mark Dykstra (student) and Brett Viren

Optimization of resources (I)

Resources for mass simulation and for data reconstruction are limited:

- experiments need to “opportunistically” grasp others’ resources
- memory in most remote execution nodes is limited (often, to 2 GB)
- `LArSoft` has one person assigned specifically to the task (Gianluca)
- improvements have been possible thanks to the joint effort with authors and with experts
- optimization starts with the localization of the problem... the solution is sometimes a 10-minute business
- some performance analysis tools are available to all the developers
- instructions will be expanded (current ones are not useful enough)



Optimization of resources (II)

A few highlights of recent achievements:

- modifications to geometry and its readout allowed saving $\mathcal{O}$ (600 MB) for LBNE full detector simulation
- restructuring MicroBooNE AnalysisTree saved $\mathcal{O}$ (1 GB)
- introduction of data compression in ADC counts (both lossy and lossless) made code a lot faster *and* decreased memory usage *and* output file size

Some of the current topics:

- all-round optimization for the coming MicroBooNE Monte Carlo Challenge 5 (this month)
- speeding up GEANT simulation

Task force for data structure revision and planning

Code and data structures play a fundamental role:

- their design has a severe impact on software performance
- as an experiment matures, new information is needed that data structures must host, and new ideas are produced that the code must accommodate, while some become obsolete

A task force is being created, to revisit the data structures according to the current and expected needs, and to address possible code structure issues. For example:

→ how to optimally merge LBNE channel disambiguation in the current algorithms?

→ are we making the best use of `art` association objects?

→ do we need a 3D cluster class? ?

→ should we drop information about single photons?

`art` experts will be deeply involved in the discussion.

Future plans

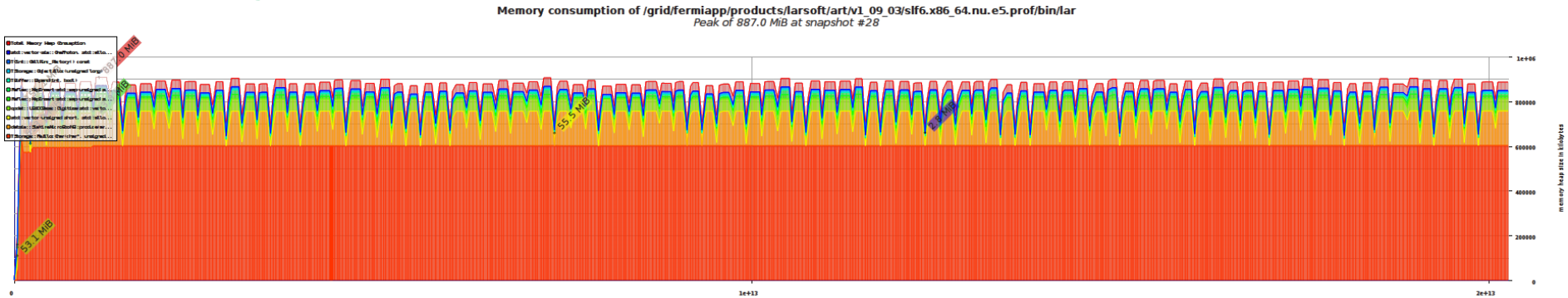
- develop algorithm to be ready for data to flow in and for new experiments to join
- activities: continuous integration, `LArSoft` architecture review
- improve the performance of the code
- identify the release managers for the experiments still missing them
- keep improving the communication between the project partners
- reorganize the documentation, especially the `LArSoft` wiki page

Thank you for your attention!

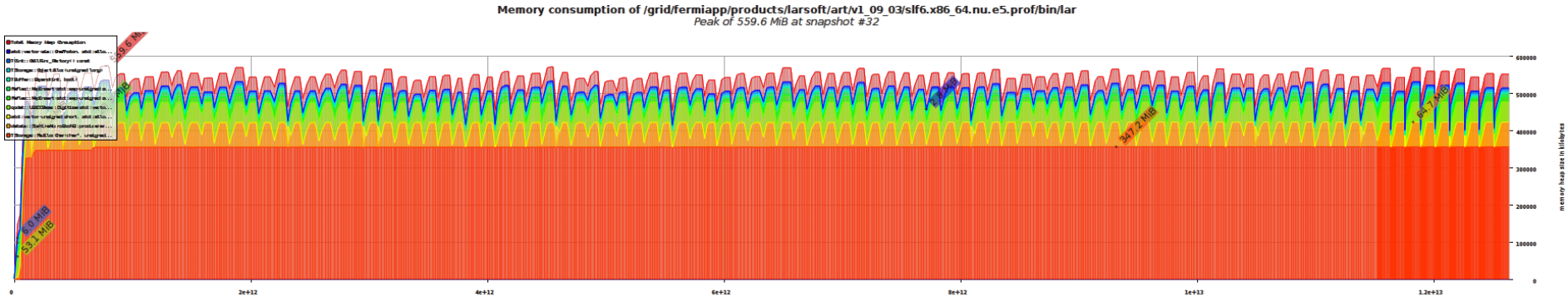
Out of curiosity: that plot over there

Memory profile of a 100-event ν_e + cosmic rays run in MicroBooNE:

Detector digitization: `standard_detsim_uboone.fcl`



The same, plus lossless compression



ROOT output: 590 $\rightarrow$ 350 MB; `raw::RawDigit` data 150 $\rightarrow$ 64 MB

$\Rightarrow$ saved $\approx$ 300 MB of memory and ran 20% faster.