A brief overview of the Geant4 Tutorials @JLab (March 2024)

Maxim Potekhin
Brookhaven National Laboratory

ePIC S&C Meeting
04/10/2024

About these slides

- The Geant4 event at JLab included a very large amount of material that is impossible to cover in any amount of detail in less than a few hours (or days)
- Included here are some pointers to a few topics that caught my attention, and a general outline of the progression of the presentations and hands-on sessions
- All combined, this was a very good and comprehensive starting point for anyone wishing to enhance or acquire their knowledge of Geant4
- The materials used in these tutorials are not committed to any central repository or a document management system. As per Geant4 tradition, informed by experience, they remain attached to the Indico agenda of the event.

General info

- Time: March 25–29, 2024. Venue: JLab CEBAF Center
- https://indico.jlab.org/event/828/
- An in-person event, well attended: ~60 participants with various research backgrounds (e.g. radiochemistry, nuclear)
- A complete introduction to Geant4, progressing from the basics to fairly advanced topics during the week; a fully functional installation of Geant4 v11.2 by the participants was a prerequisite to the course
- Daily “hands-on” sessions, well prepared and well documented

The hands-on materials: the progression

- Starting point: the well known “B1” example with simple geometry which illustrates the basics of volume definition and placement
- Finishing point: derived from “B5”. *“This example implements a double-arm spectrometer with wire chambers, hodoscopes and calorimeters. Event simulation and collection are enabled, as well as event display and analysis.”*
- …as such, a fairly realistic representation of a fixed-target experiment

Presentations: some points of interest

- Physics, and Physics List selection
- Stack management (three stacks)
- Multithreading
  - Sub-event parallelization (work in progress)
- Scoring and probes
- The “analysis”
- Not relevant to HEP/NP, but worth a mention
  - Geant4-DNA, a project initiated by the European Space Agency to study radiation damage at the molecular level
  - Tomography and radiotherapy applications

Physics Lists

- "Reference Physics Lists" are maintained, used by major experiments, extensively validated
- Still, there is not guarantee a particular list will work very well for a particular application, so it's up to the user to perform their own validation
- Naming conventions are used to help navigate the collection of the physics lists
- And, there is the *G4VUserPhysicsList* — does what the name suggests; the user must implement the two pure virtual methods *ConstructParticle()* and *ConstructProcess()* and other elements – NB you would have to construct the transport part as well, set cuts etc

Stacks in Geant4

- "Urgent", "Waiting" and "*PostponeToNextEvent*"
- A track is popped up only from Urgent stack
- Once Urgent stack becomes empty, all tracks in Waiting stack are transferred to the Urgent stack
- The stack API allows the developer to manage allocation of tracks to stacks
- One of the goals of this design: better control of resources e.g. if specific tracks of interest don't make it, the event simulation can be cut short, or other measures taken

Multithreading (event-level)

- *"CPU frequency had come to a plateau in 2005. Number of transistors you can buy per $ is still growing. Hence, many cores."*
- But, memory size per core is shrinking
- Multithreaded event-level parallelism mode: since Geant4 v10.0 (Dec 2013)
  - Taking the advantage of independence of events, many cores (threads) process events in parallel (event-level parallelism)
  - Geometry/cross-section tables are shared over threads
- Task-based event-level parallelism mode: since Geant4 v11.0 (Dec 2021)
  - Decoupling task (event loop) from thread — more flexible load-balancing

Multithreading (sub-event level)

- ETA: December 2024, work in progress
- Split an event into sub-events and task them separately
- Sub-event :
  - Group of tracks of selected kinds or getting into a particular detector component
- **The goal is to utilize heterogeneous hardware architectures**
  - Recent studies showed that each GPU process should have strictly limited scope to achieve the desired performance

Scoring and probes

- Geant doesn't preserve any data produced at runtime by default
- The canonical way to form and preserve the data is to create a "sensitive detector"
- …but this can be not so easy and/or an overkill in some use cases
- Geant provides built-in and customized scorers
  - e.g. *energyDeposit*, *nOfTracks* etc for the built-in scorers
  - Can be managed from the Geant command interface
- Can be linked to a *"scoring mesh"* created by the user or a logical volume
- *Filters* can be added to scorers i.e. scoring can be conditional
- The user can create *probes* which are "virtual cubes" and apply the scoring logic there

”Analysis”

- Somewhat of a misnomer since its a limited scope (but useful) toolkit for organizing and persisting the data produced by Geant applications, not really an analysis tool
- Allows the developer to create and fill histograms and Ntuples, and save those in a few available formats (ROOT, XML, CSV, HDF5)
- Comment: hdf5 can be a very efficient way to interface Python-based software, note that using it requires HDF5 libraries installation and Geant4 build with -DGEANT4_USE_HDF5=ON
- Users access the ***g4analysis*** tools via the G4AnalysisManager, a singleton class that comprises all of the functionality for data manipulation and persistence

Summary

- Geant remains in the state of active development, with ongoing effort to further improve existing functionality and add new capabilities, both in terms of physics and utilization of computing resources
- The tutorials at JLab generated materials that can be quite useful for study and reference. They do not replace the Geant4 documentation but serve as a navigation tool for exploring it