HADRONIC PHYSICS AND PHYSICS LIST DESIGN

V. Ivanchenko

CERN & Tomsk State University, Tomsk, Russia

25th Geant4 Collaboration Workshop

23 September 2020

Outline

- Hadronic design update
- Destruction end of job
- New utilities for hadron physics configuration
- Hadronic parameters
- Variation of hadronic cross sections
- Initialization of nuclear level data
- Summary

- Basic goals or recent modification of Physics Lists:
  - *Reduce duplicated code in EM and hadronic configurations*
  - *More transparent configuration of models and cross sections*
  - *Optional addition of b-, c- hadrons*
  - *Cross section variation required for systematics studies*

Motivation

- Number of EM physics constructors – 12
  - *Number of particles – 55*
    - 14 are configured individually
    - 41 are configured in the same way
- Number of hadronic builders – 62
- Number of hadron_inelastic physics – 18
- Amount of duplicate code is huge
  - *If we will implement b- and c- meson/baryon physics in the same style then number of combinations will increase substantially*
- Hadronic physics was not properly destructed end of run

Models, hadronic framework

- All hadronic processes, models, and cross sections are registered and are destructed end of run
  - *BuildPhysicsTable(..) method for G4VCrossSectionDataSet and G4HadronicInteraction*
  - *Components of hadronic framework use G4HadronicInteraction as a base class*
    - G4QuasiElastic, G4ExcitedStringDecay….
- Simplified instantiation of FTF and QGS model
  - *Pre-compound is the default transport model*
  - *G4ExcitedStringDecay by default uses G4LundStringFragmentation*
- Recommendations:
  - *Do not delete internal objects end of run if they are registered*
  - *Do not use particle type at initialisation, instead use PDG code*
    - Elementary cross section G4HadronNucleusXsc is fixed in this respect

Destruction of physics at exit
(initial requirements)

- Why we need full destruction of physics at exit?
  - *Users have trouble using debug tools like valgrind*
  - *Users may have problem in their application code when destruct Geant4*
  - *Developers have problem to identify memory leaks*
- Recommendations:
  - *cross sections, models, and processes should be instantiated via pointers not be part of any other objects*
  - *no private destruction of these objects is allowed*
  - *We should not use G4THREAD_LOCAL data members*
    - Both in hadronics and in Physics Lists constructors
- The most important pending updates:
  - *Simplified instantiation of FTF and QGS model*
    - Builders should not instantiate Lund fragmentation and Participants
      - *This should be done without interface change*
  - *Provide correct destruction of HP and AllHP models and cross sections*

Destruction at exit in 10.7

- Destruction end of job was not working properly
  - *Some data members are thread local*
  - *Some was not deleted end of job*
- Difficulty of destruction of physics is in the fact, that cross section, model, and process classes may be shared between different particles in different ways for different Physics Lists
  - *This is strongly needed to reduce memory and CPU for initialization of physics but make destruction problematic*
- For 10.7beta the most part of physics is destructed due to use of register/deregister mechanism
  - *Processes, models, and cross sections should not be deleted by Physics List classes*
  - *A good part of thread local variables in Physics Lists are removed*
- What is not done: some hadronic builders are thread local

New utilities for hadron physics configuration

- G4HadParticles – returns several lists of particle PDG codes
  - *std::vector<G4int>& G4HadParticles::GetKaons();*
  - *std::vector<G4int>& G4HadParticles::GetHyperons);*
  - *std::vector<G4int>& G4HadParticles::GetAntiHyperons();*
  - *…*
- G4HadProcesses – return pointers to hadronic processes per particle and allows adding extra cross section per particle
  - *G4HadronicProcess* G4HadProcesses::FindInelasticProcess(const G4String& partname);*
  - *G4HadronicProcess* G4HadProcesses::FindInelasticProcess(const G4String& partname);*
  - *G4bool G4HadronicProcesses::AddInelasticCrossSection(const G4ParticleDefinition*, G4VCrossSectionDataSet* my_xs);*
  - *……*
- G4HadronicBuilder – build standard set of models and cross sections for group of particles
  - *G4HadronicBuilder::BuildHyperonsFTFP_BERT();*
  - *G4HadronicBuilder::BuildBCHadronsFTFP_BERT();*
  - *G4HadronicBuilder::BuildHyperonsQGSP_BERT();*

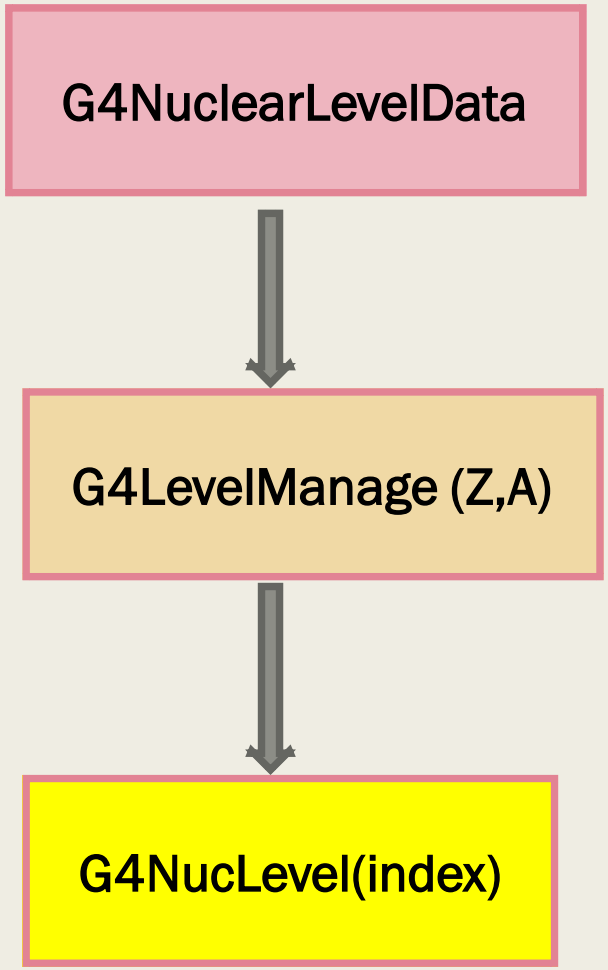
Hadronic parameters

- Utilities in the previous slide are using G4HadronicParameters class
  - *User has a chance to change any parameter between instantiation of the PhysicsList and run initialization*
    - G4State_PreInit
    - Both C++ interface and UI commands (not for all) are available
    - Should we force, at least, a printout on each UI command?
- New parameters:
  - *G4bool EnableBCParticles()*
- Proposed extra parameters:
  - *G4double EnergyThresholdForHeavyParticles()*
    - The default is 1.1 GeV
    - If max energy is below, then no hyperons, anti-ions, b-, c- particle physics
    - We need to check if this bring some advantages to low-energy simulations

Variation of hadronic cross sections

- For study of systematic uncertainty due to simulation we may considered following approach:
  - *For hadronic models we propose to use different Physics Lists*
    - FTFP_BERT -> QGSP_BIC, FTFP_INCLXX, or QBBC
  - *For cross sections we may propose to use a factor to vary cross section value*
    - +- 5-10% would be within Geant4 accuracy
- Cross section factors are defined via G4HadronicParameters class:
  - *G4bool ApplyFactorXS() const; // false by default*
  - *G4double XSFactorNucleonInelastic() const ;*
  - *G4double XSFactorNucleonElastic() const ;*
  - *G4double XSFactorPionInelastic() const ;*
  - *G4double XSFactorPionElastic() const ;*
  - *G4double XSFactorHadronInelastic() const ;*
  - *G4double XSFactorHadronElastic() const ;*
  - *G4double XSFactorEM() const ;*
- User must change the flag and set corresponding factor via C++ interface

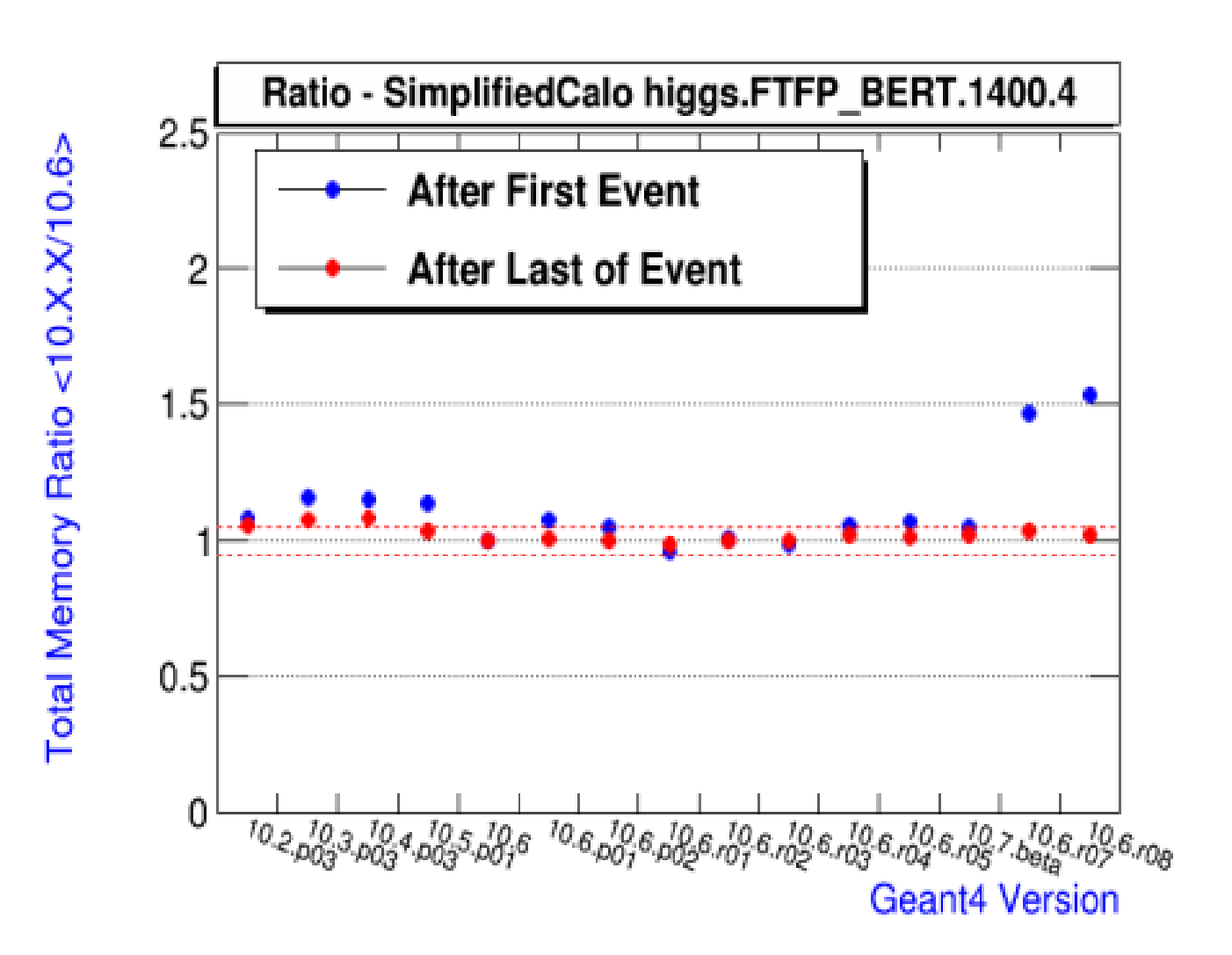
Nuclear level data



- Since 10.3 we have nuclear level data handled by G4NuclearLevelData class
  - *Static singleton shared between all threads*
  - *In 10.5 we had only lazy initialization per isotope*
  - *In 10.6 two possibilities*
    - On demand initialization of all needed isotopes before the run Z < Zmax
    - lazy initialization of the data per isotope badly interacts
  - *In 10.7 Z < Zmax initialization will be default*
- The memory used:
  - *Full data (all levels are uploaded) takes 56 MB*
  - *Data without e- internal conversion coefficients 8 MB*

Memory profile (FNAL group)

https://g4cpt.fnal.gov/g4p/summary/mem_SimplifiedCalo_higgs.html



Summary

- Described developments together with development of Alberto for decay channels of b-, c- hadrons complete our plans to improve Physics List configurations for 2020
  - *We can offer c- and b- hadron physics*
  - *We can offer a method of Geant4 physics variation for study of systematics*
    - It is available for more popular Physics List (FTFP_BERT…, QGSP_…, QBBC)
    - Should be propagated to other hadron inelastic constructors
  - *Bug fix and tuning of the approach are not excluded*
- Destruction of all EM and hadronic physics end of run is achieved
- Optimization of the initialization of nuclear level data structures is not yet finalized
  - *By default we upload only data for Z<Zmax*
  - *It is possible to upload all data begin of run*
  - *Recently Makoto proposed convert ASCII data files into binary compressed file(s)*

Plans for the next release

- Makoto pointed out that having data structure with many small files makes problems for HPCs:
  - *Lazy initialization is difficult at this architecture*
  - *Reading of many small files at initialization is also a problem*
- Proposed solution:
  - *Produce one big binary file from these ASCII files*
  - *G4NuclearLevelData::DumpData(const G4String& file)*
  - *Added extra Boolean parameters ReadASCII*
  - *Create 2 binary files*
    - One for HEP - no internal e- conversion
    - Second - full data
  - *May be implemented for the next release*
- Physics_list/builder sub-directory may be cleaned up
  - *Removing all thread local variables*