DISCUSSION POINTS ON GENERIC BIASING

Generic Processes & Material Session

Marc Verderi, LLR, Ecole polyechnique
Geant4 Catania Collaboration Meeting
7 – 11 October 2024

Overview

- G4XxxxGeneralProcess & Generic Biasing
  - Xxxx = Gamma or Neutron
- Geometry Importance Biasing & Generic Biasing
- Physics Process Biasing & Final State Case

G4XxxxGeneralProcess & Generic Biasing

Xxxx = Gamma or Neutron

- G4XxxxGeneralProcess improves CPU performances @ no cost on physics
  - G4GammaGeneralProcess now used as default (in all std EM phys. lists ?)
  - G4NeutronGeneralProcess pending, as clashing with GB in examples
- To bias a physics process, GB wraps it, to possibly substitute, on user's demand, interaction law and/or final state generation with a biased version
  - Changing the interaction law can be made generically
  - Changing the final state generation is an other story (see after)
- There is (a priori) no show-stopper for GB to use G4XxxxGeneralProcess
- But we should discuss if:
  - We consider G4XxxxGeneralProcess to be biased as a whole
    - In what case, we "just" need to create new GB classes (in examples), adapted to general processes
    - I would advocate then for options to create, eg, FTFP_BERT with/without general processes
      - To avoid an inflation of number of physics lists
      - And to make clear that the physics content is the same
  - We wish GB to control G4XxxxGeneralProcess sub-processes
    - In what case we need to define a G4VGeneralProcess interface
      - To give access to sub-processes
      - And to avoid dependencies of processes/biasing onto other processes domain

Geometry Importance Biasing & Generic Biasing

- Geometry Importance Biasing was the first biasing technique offered in Geant4
- "Importance values" I_{volume} are assigned volumes and used for
  - splitting (if $I_{next\ volume} > I_{volume}$) or
  - Russian-roulette-killing (if $I_{next\ volume} < I_{volume}$)
  - the tracks that reach cell/volume boundaries
- In short :
  - Importance values are defined in cells
  - These importance values are associated to a particle type
  - A dedicated process use these importance values to apply the biasing
- Long pending discussion on how to "unify/merge" GIP and GB
  - Note : GB03 provides a flexible geometry importance based example, independent of the Geometry Importance Biasing design
- Main idea:
  - Allow usage of (user) existing Geometry Importance Biasing setups in GB with small/minimal changes
- Initial idea was to provide an example for this
  - But appears as a too naïve approach
  - Because Geometry Importance Biasing classes carry quite interplay between the importance values, the process and the particle type handlings
  - In short "just taking the importance values setup" is not possible
  - Need to separate things in the Geometry Importance Biasing classes
    - G4VIStore with a singleton derived class G4IStore : to store the importance values of the "cells"
    - G4GeometrySampler : to associate a geometry (mass or parallel) to a particle type (by name)
- So, some work is needed to make this possible

Physics Process Biasing & Final State Case

- GB provides hooks to bias physics processes
  - PostStep biasing
    - Biasing of interaction probability
    - Biasing of final state
  - Same for Along
  - To be done for AtRest
- Biasing of interaction probability
  - Generic approach is possible
    - As underneath law is the classical exponential one
  - Formalism applies to both neutral and charged particles
    - In short, giving $\sigma(\ell)$ or $p(\ell)$ or $P_{NI}(\ell)$ is enough to define the interaction law (each of these can be transformed in the others)
    - And biasing simply consists in replacing the analog version of these by a biased one.
  - Biasing of charged particles interaction law can be made based on an "à la Woodcock" approach
- Biasing of final state is difficult
  - Because there is no generic "final state" class
  - Would require a generic differential cross-section class
  - But some popular techniques (eg: DXTRAN) only requires 1 → 1 differential cross-section (elastic)
    - Having at least a generic solution for low multiplicities would help