‘Stateless Geant4’ prototype

A.Gheata, W.Pokorski

15.09.2020

Introduction

- hardware trends push for concurrency and vectorization
    - HPC centers
    - GPUs
- general simulation code is probably the one most ‘unfriendly’ for vectorization
    - many branches
        - we don’t have 1000 particles undergoing the same process, but we have 1000 ’if’ statements…
    - stochasticity
- experiments really need faster (and more precise) simulation
    - we need to try whatever we can to efficiently run on modern hardware
- attempts of vectorization like the GeantV prototype (https://arxiv.org/abs/2005.00949) gave an insight what realistically could or could not be achieved
    - can we apply the techniques that gave some speedups in Geant4?

GeantV lesson

- GeantV prototype showed that although the speedup of several factors cannot be achieved, some processing stages could potentially gain from vectorization
  - propagation in magnetic field
  - MSC
  - some physics models
- can we get those speedups in Geant4 for a reasonable price?
  - getting any % without breaking (too much) the backward compatibility would be a success
- can we (do we need to) do anything to make exploitations of those different approaches easier (possible)?

Current status

- Geant4 propagates particles one-by-one (per thread)
- several Geant4 classes (managers, navigator, transportation, etc) hold the state related to the currently propagated particle
- default behavior is to propagate each particle from the first to the last step in one go
  - one can play with ‘stacking actions’ to ‘postpone’ particles, but this can be done only between steps
- Geant4 step contains several ‘stages’ (geometry, physics, transportation, magnetic field propagation) which currently can’t be separated
  - this prevents any fine-grained parallelism

Goal of this R&D

- make Geant4 engine ‘stateless’
    - attach all the ‘state’ to each track
- split Geant4 step in ‘stages’ that can be executed independently
- introduce containers, allowing to group particles for each of the stage
    - each ‘stage’ processes particles waiting and passes them on to the next stage
    - ‘stages’ can run independently and in parallel
        - they ‘consume’ what is in their container and populate other containers
            - they could process the input from the container in vector-like manner

Disclaimer

- no performance measurements
- no memory usage measurements
- no studies of saturating specific containers or blocking between stages
- no detailed physics validation (certainly several bugs overlooked)

- goals:
  - understand the code changes in Geant4 required to introduce such an architecture
  - demonstrate the feasibility by running a simple example

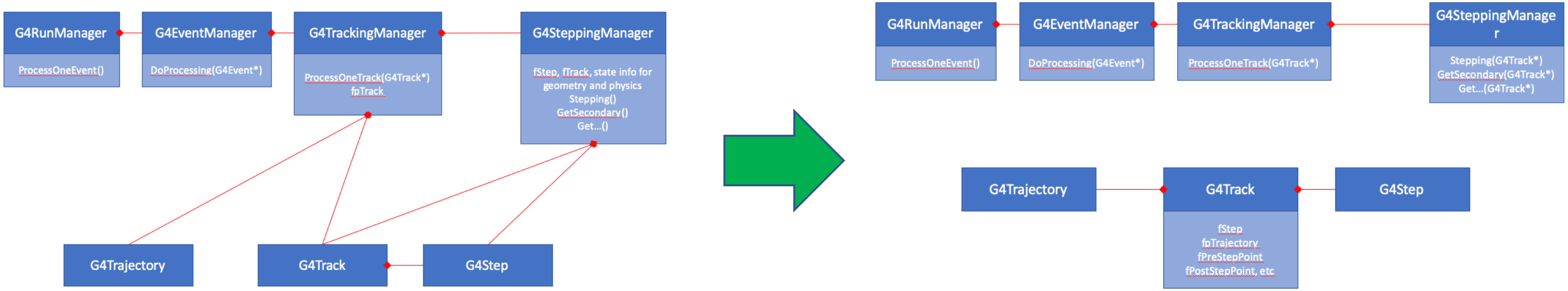
Code changes for ‘statelessness’

- strategy
  - remove any members from the Geant4 classes (managers, navigators, etc) that hold the state of the track
    - either introduce them in G4Track directly
    - or introduce ‘state’ classes and add pointers to those from G4Track
  - change the signature of the relevant methods and add
    - either G4Track as argument
    - or one of the new ‘state’ classes as argument

  - in summary: no caching of state and all relevant methods called with ‘G4Track*’ as argument

Managers

- pointers to track and step moved away from managers
  - methods of the managers called now with argument *Track
- moved any state information from managers to track
- all step- and physics-related members moved from stepping manager to track



Navigators

- Removing state from navigator (G4Navigator, G4VoxelNavigator, G4VoxelSafety)
  - Quite some caching and 'state memory' in different methods
- Problem: can't pass track as argument to navigator methods because of circular dependency
- Solution: introduced 'navigator state struct’ (similar idea to G4FieldTrack)
  - Each track owns such an object
  - it gets passed it to navigator methods
- Added state as argument to Navigator methods
  - G4double ComputeStep(…) -> G4double ComputeStep(G4NavigatorState *const state, …)

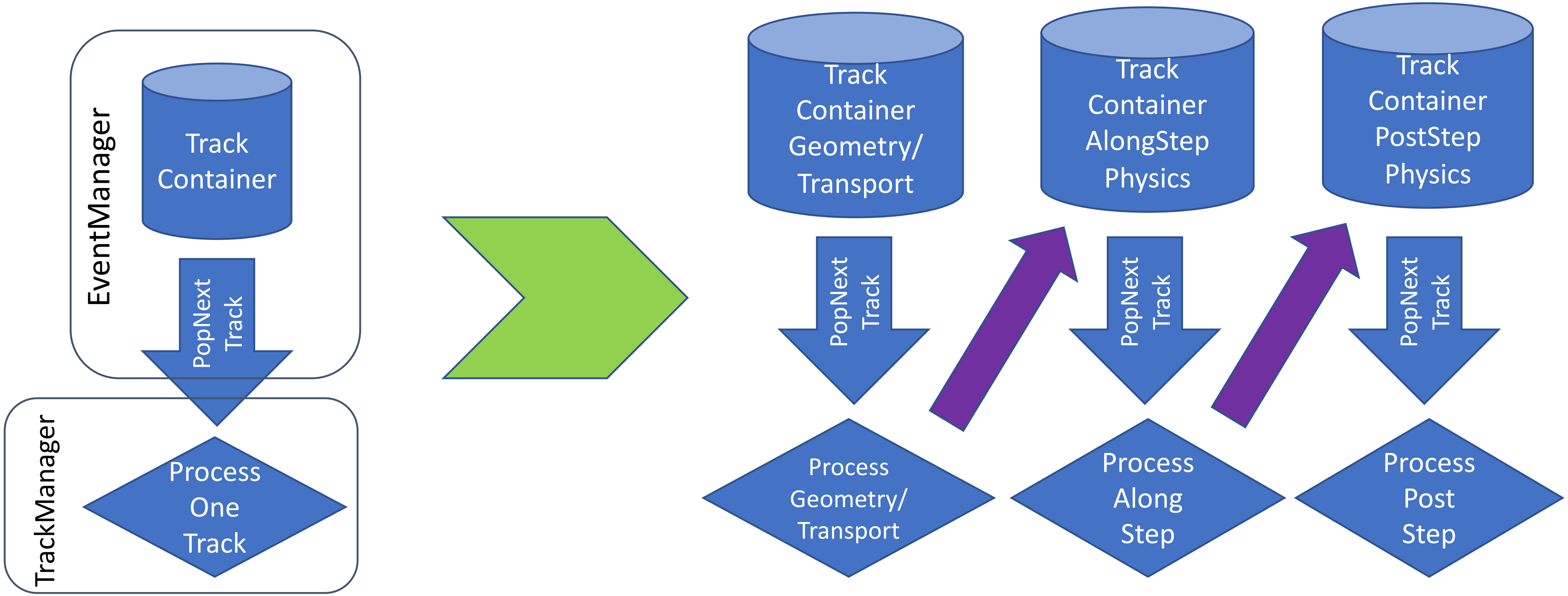
Processes

- similarly for Transportation and other processes
  - removed any ’state caching members’ and introduced
    - G4TransportState.hh
    - G4ELossState.hh
    - etc

- all those ‘state’ objects are attached to G4Track

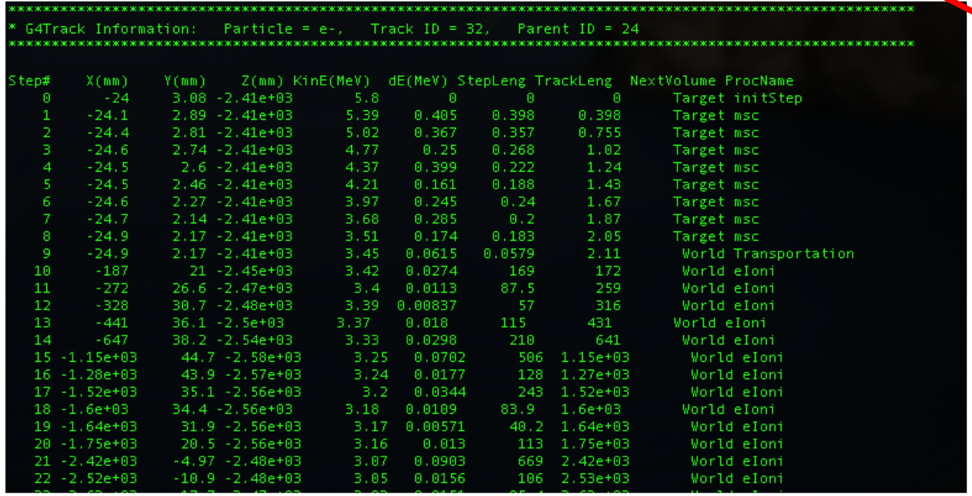
Breaking ‘steps’ into stages and introducing containers

- stepping split in stages (along step geometry, physics, post step, etc)
- Introduced containers of particles for each stage
- processing is finished when all containers are empty



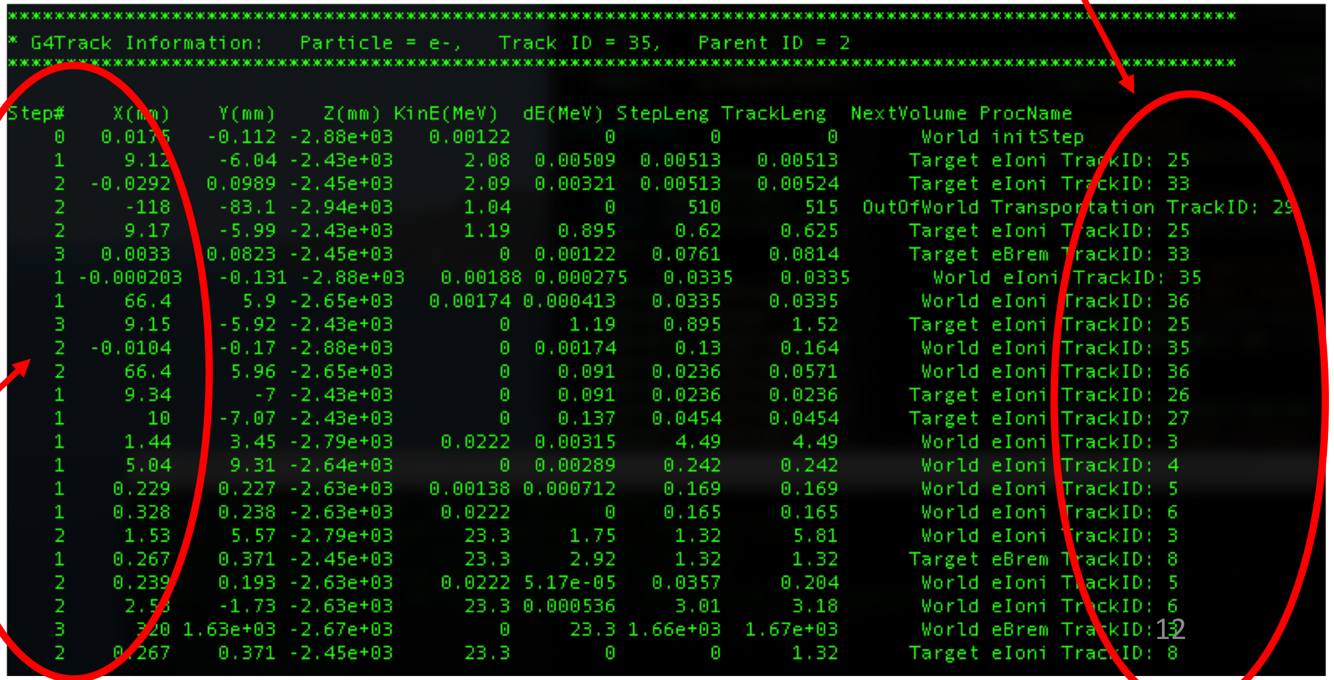
Tests

- running exampleB2a for the purpose of test
  - no detailed validation, just making sure it compiles, doesn’t crash and ‘seems to run ok’
- able to run 1000s of events with steps split into stages and several particles transported in the same time



```
****************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 32,   Parent ID = 24
****************************************************************************************

Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0      -24     3.08 -2.41e+03       5.8        0        0         0      Target initStep
    1    -24.1     2.89 -2.41e+03      5.39    0.405    0.398     0.398      Target msc
    2    -24.4     2.81 -2.41e+03      5.02    0.367    0.357     0.755      Target msc
    3    -24.6     2.74 -2.41e+03      4.77     0.25    0.268      1.02      Target msc
    4    -24.5      2.6 -2.41e+03      4.37    0.399    0.222      1.24      Target msc
    5    -24.5     2.46 -2.41e+03      4.21    0.161    0.188      1.43      Target msc
    6    -24.6     2.27 -2.41e+03      3.97    0.245     0.24      1.67      Target msc
    7    -24.7     2.14 -2.41e+03      3.68    0.285      0.2      1.87      Target msc
    8    -24.9     2.17 -2.41e+03      3.51    0.174    0.183      2.05      Target msc
    9    -24.9     2.17 -2.41e+03      3.45   0.0615   0.0579      2.11       World Transportation
   10     -187       21 -2.45e+03      3.42   0.0274      169       172       World eIoni
   11     -272     26.6 -2.47e+03       3.4   0.0113     87.5       259       World eIoni
   12     -328     30.7 -2.48e+03      3.39  0.00837       57       316       World eIoni
   13     -441     36.1 -2.5e+03     3.37    0.018      115       431      World eIoni
   14     -647     38.2 -2.54e+03      3.33   0.0298      210       641       World eIoni
   15 -1.15e+03     44.7 -2.58e+03      3.25   0.0702      506  1.15e+03       World eIoni
   16 -1.28e+03     43.9 -2.57e+03      3.24   0.0177      128  1.27e+03       World eIoni
   17 -1.52e+03     35.1 -2.56e+03       3.2   0.0344      243  1.52e+03       World eIoni
   18 -1.6e+03     34.4 -2.56e+03      3.18   0.0109     83.9  1.6e+03       World eIoni
   19 -1.64e+03     31.9 -2.56e+03      3.17  0.00571     40.2  1.64e+03       World eIoni
   20 -1.75e+03     20.5 -2.56e+03      3.16    0.013      113  1.75e+03       World eIoni
   21 -2.42e+03    -4.97 -2.48e+03      3.07   0.0903      669  2.42e+03       World eIoni
   22 -2.52e+03    -10.9 -2.48e+03      3.05   0.0156      106  2.53e+03       World eIoni
```



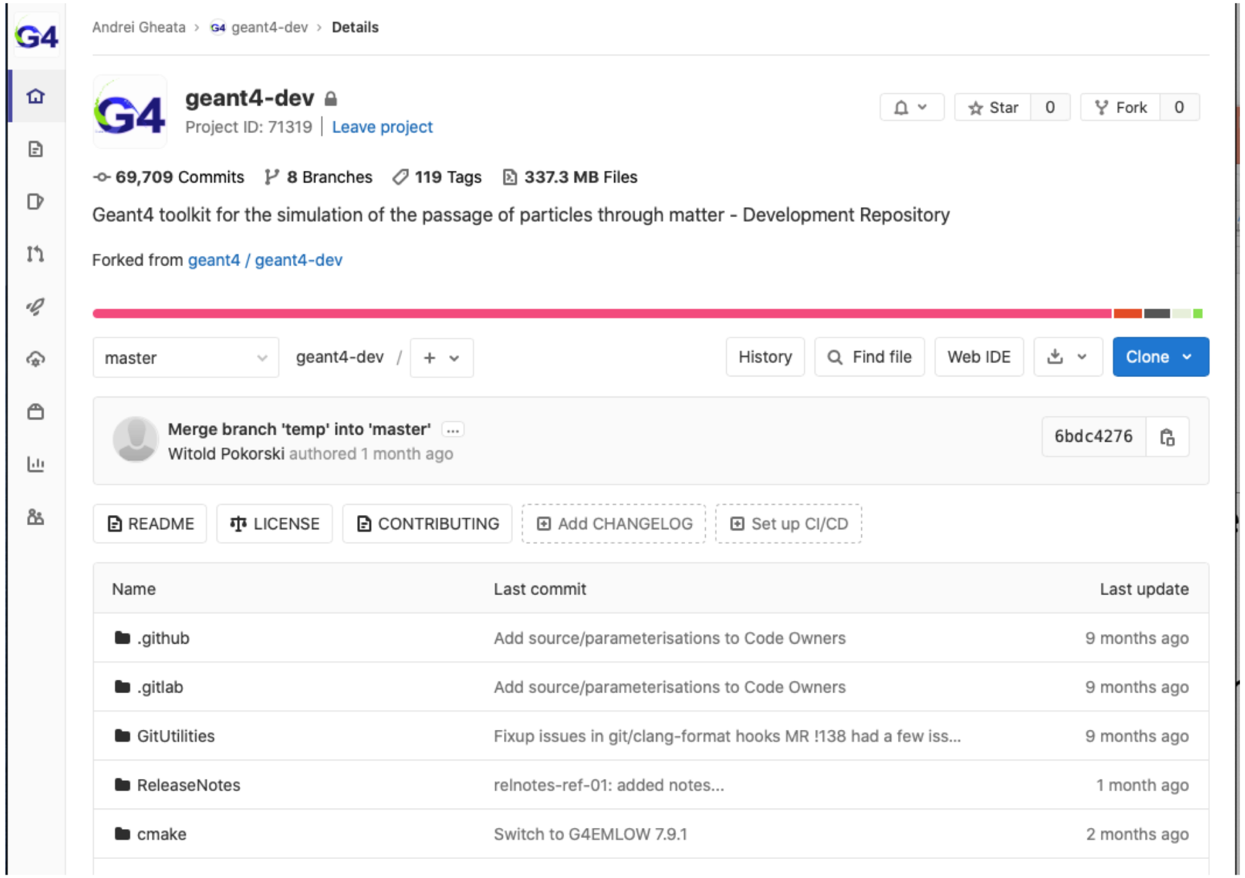
```
****************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 35,   Parent ID = 2
****************************************************************************************

Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0   0.0175   -0.112 -2.88e+03   0.00122        0        0         0       World initStep
    1     9.12    -6.04 -2.43e+03      2.08  0.00509  0.00513   0.00513      Target eIoni TrackID: 25
    2  -0.0292   0.0989 -2.45e+03      2.09  0.00321  0.00513   0.00524      Target eIoni TrackID: 33
    2     -118    -83.1 -2.94e+03      1.04        0      510       515  OutOfWorld Transportation TrackID: 29
    2     9.17    -5.99 -2.43e+03      1.19    0.895     0.62     0.625      Target eIoni TrackID: 25
    3   0.0033   0.0823 -2.45e+03         0  0.00122   0.0761    0.0814      Target eBrem TrackID: 33
    1 -0.000203   -0.131 -2.88e+03   0.00188 0.000275   0.0335    0.0335       World eIoni TrackID: 35
    1     66.4      5.9 -2.65e+03   0.00174 0.000413   0.0335    0.0335       World eIoni TrackID: 36
    3     9.15    -5.92 -2.43e+03         0     1.19    0.895      1.52      Target eIoni TrackID: 25
    2  -0.0104    -0.17 -2.88e+03         0  0.00174     0.13     0.164       World eIoni TrackID: 35
    2     66.4     5.96 -2.65e+03         0    0.091   0.0236    0.0571       World eIoni TrackID: 36
    1     9.34       -7 -2.43e+03         0    0.091   0.0236    0.0236      Target eIoni TrackID: 26
    1       10    -7.07 -2.43e+03         0    0.137   0.0454    0.0454      Target eIoni TrackID: 27
    1     1.44     3.45 -2.79e+03    0.0222  0.00315     4.49      4.49       World eIoni TrackID: 3
    1     5.04     9.31 -2.64e+03         0  0.00289    0.242     0.242       World eIoni TrackID: 4
    1    0.229    0.227 -2.63e+03   0.00138 0.000712    0.169     0.169       World eIoni TrackID: 5
    1    0.328    0.238 -2.63e+03    0.0222        0    0.165     0.165       World eIoni TrackID: 6
    2     1.53     5.57 -2.79e+03      23.3     1.75     1.32      5.81       World eIoni TrackID: 3
    1    0.267    0.371 -2.45e+03      23.3     2.92     1.32      1.32      Target eBrem TrackID: 8
    2    0.239    0.193 -2.63e+03    0.0222 5.17e-05   0.0357     0.204       World eIoni TrackID: 5
    2     2.53    -1.73 -2.63e+03      23.3 0.000536     3.01      3.18       World eIoni TrackID: 6
    3      320 1.63e+03 -2.67e+03         0     23.3 1.66e+03  1.67e+03       World eBrem TrackID:
    2    0.267    0.371 -2.45e+03      23.3        0        0      1.32      Target eIoni TrackID: 8
```

each line gets printed when one of the particles, completes it’s ‘travel’ through all the containers

Repository

- fork of the geant4-dev
  - https://gitlab.cern.ch/agheata/geant4-dev
- developments on top of geant4-dev master



Andrei Gheata › geant4-dev › Details

geant4-dev

Project ID: 71319 | Leave project

69,709 Commits 8 Branches 119 Tags 337.3 MB Files

Geant4 toolkit for the simulation of the passage of particles through matter - Development Repository

Forked from geant4 / geant4-dev

Merge branch 'temp' into 'master'
Witold Pokorski authored 1 month ago
6bdc4276

Name	Last commit	Last update
.github	Add source/parameterisations to Code Owners	9 months ago
.gitlab	Add source/parameterisations to Code Owners	9 months ago
GitUtilities	Fixup issues in git/clang-format hooks MR !138 had a few iss...	9 months ago
ReleaseNotes	relnotes-ref-01: added notes...	1 month ago
cmake	Switch to G4EMLOW 7.9.1	2 months ago

Conclusions

- making Geant4 engine ‘stateless’ and splitting the step into stages requires substantial, but feasible changes
  - backwards compatibly could be, to a very large extend, preserved
    - only more advanced users’ code might need modifications
- we could start from introducing changes in G4Navigator, which were anyway planned
  - ‘Separate safety computation and state from navigator’ – on Geometry Work Plan for 2020
- once the G4Navigator is stateless, we could re-evaluate other required changes and try to perform some performance comparisons
  - smooth path to architecture modification allowing to further experiment with vectorization and parallelism
    - it would add quite some flexibility for further potential improvements related to track/step-level parallelism