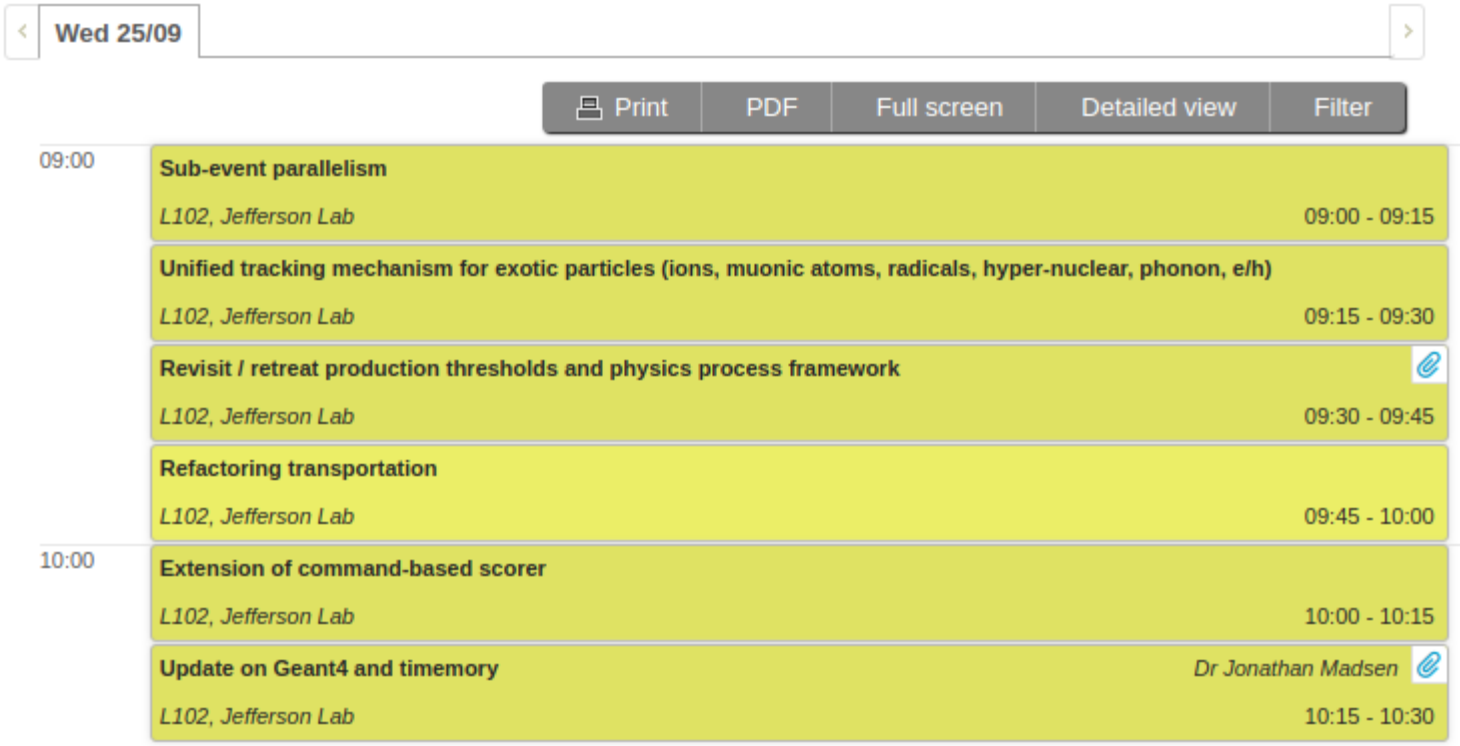
Report of parallel 4B session
Near-term kernel developments

Conveners:
Makoto Asai
Marc Verderi

Session agenda



- Session was setup as a working session, to take advantage of being together to address a series of points.

Sub-event parallelism, Unified tracking mechanism …, Refactoring transportation

- **Sub-event parallelism**
  - Author : Makoto
  - Solution mostly designed
  - However no manpower in the short term to provide an implementation
- **Unified tracking mechanism for exotic particles (ions, muonic atoms, radicals, hyper-nuclear, phonon, e/h)**
  - Author : Makoto
  - Mechanism provided, relies on a GenericXXX concept:
    - GenericXXX messages the appropriate particle definition
  - Generic Ions and Generic Muonic Atoms have been implemented
  - More species will be provided once their physics will be available
- **Refactoring transportation**
  - Authors : Makoto, John
  - Plan to start with a neutral and a charged flavor of transportations
    - Neutral version produced by taking away code from the charged one
  - Model approach can be considered too
    - One transportation with several models (neutral, charged, gravitational,…)
    - Instead of several dedicated transportations
  - Logic to call to field code will have to be revised:
    - Field logic very general (too general ? Could be gravitational field), always called

REVIEW OF PRODUCTION THRESHOLDS (DRAFT OF PLENARY)

Parallel 4B

Marc Verderi
LLR/Ecole polytechnique
Jefferson Lab Collaboration Meeting
September 2019

Review of productions thresholds

- Parallel session offers the opportunity to discuss the content of the presentation and of the proposal
- Pointed one misunderstanding about “apply cuts” mechanism
  - Corrected during coffee break for plenary presentation ;)
- Came essentially to an agreement on the proposal:
  - Of giving full responsibility of cuts to processes
  - And offloading kernel classes from the cut business
- Discussion continued shortly during plenary
  - with the conclusion to further continue this study
  - And come with a solid / well documented proposal

Extending command-based scorer

SLAC

- Current command-based scorer works as following.
  1. Construct a parallel world with mesh geometry and register a dedicated G4ParallelWorldProcess to ProcessManager of all particle types
  2. Instantiate thread-local G4MultiFunctionalDetector and register requested PrimitiveScoere(s)
  3. Set thread-local G4MultiFunctionalDetector to logical volume of the mesh cell
  4. HitMap of each PrimitiveScoere is summed up after each event, and merged after the event loop
- Idea is to provide the same command-based scoring mechanism for volumes defined in mass world.
  - Steps 2. and 4. can be reused. The only addition is letting thread-local G4MultiFunctionalDetector to a logical volume in the mass world
- Also, we can utilize the G4Analysis package to create histograms, plots, n-tuples on the fly without writing any single line of C++ code.

Command examples

- Construct geometry via reading GDML or ordinary detector construction, etc.
- Command samples

  /score/create/realWorldLogVol *<logVolName>*

  /score/quantity/energyDeposit *<primitiveScorerName> <unit>*

  /score/close

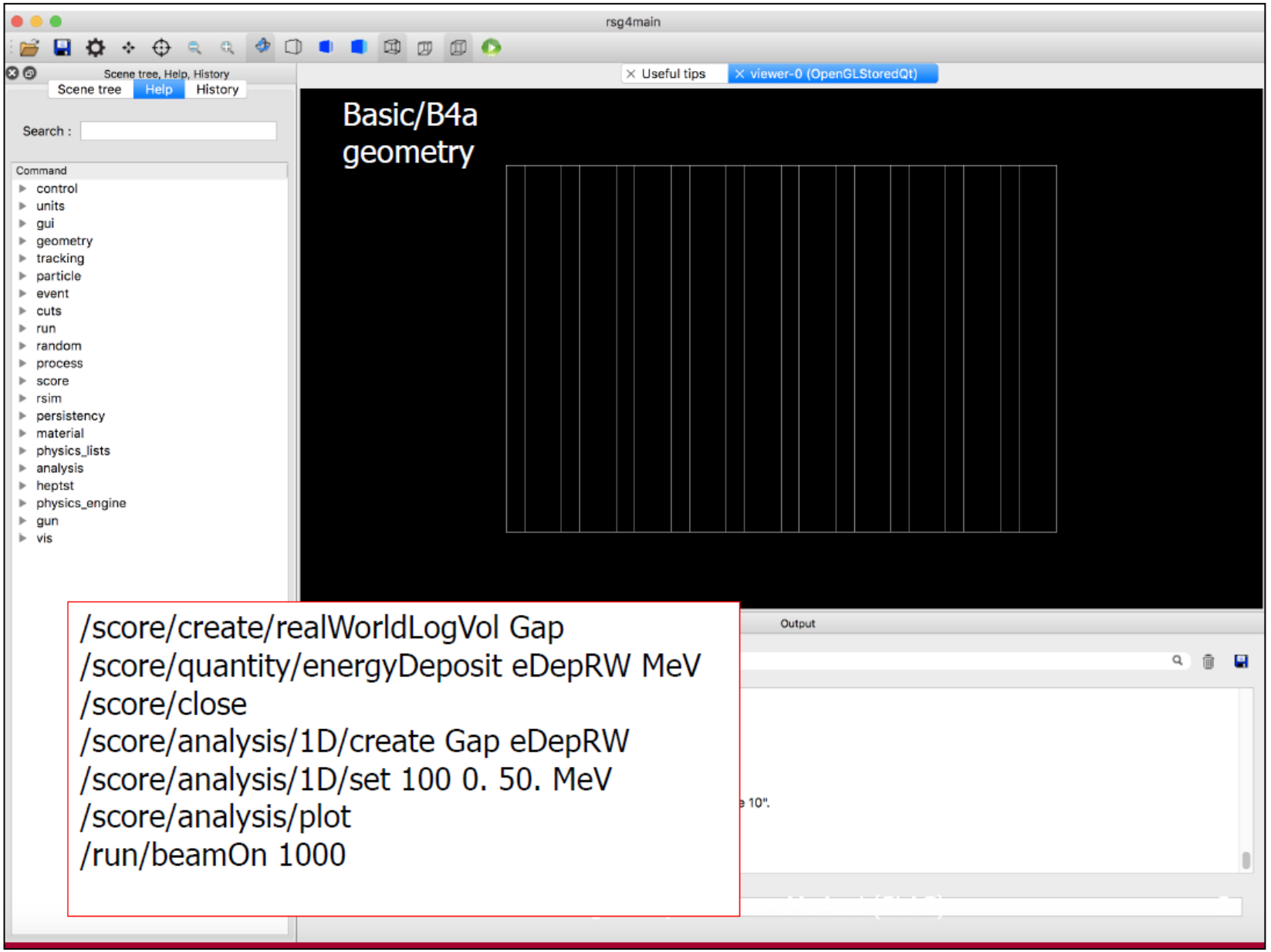
  /score/analysis/1D/create *<logVolName> <primitiveScorerName> <copyNo>*

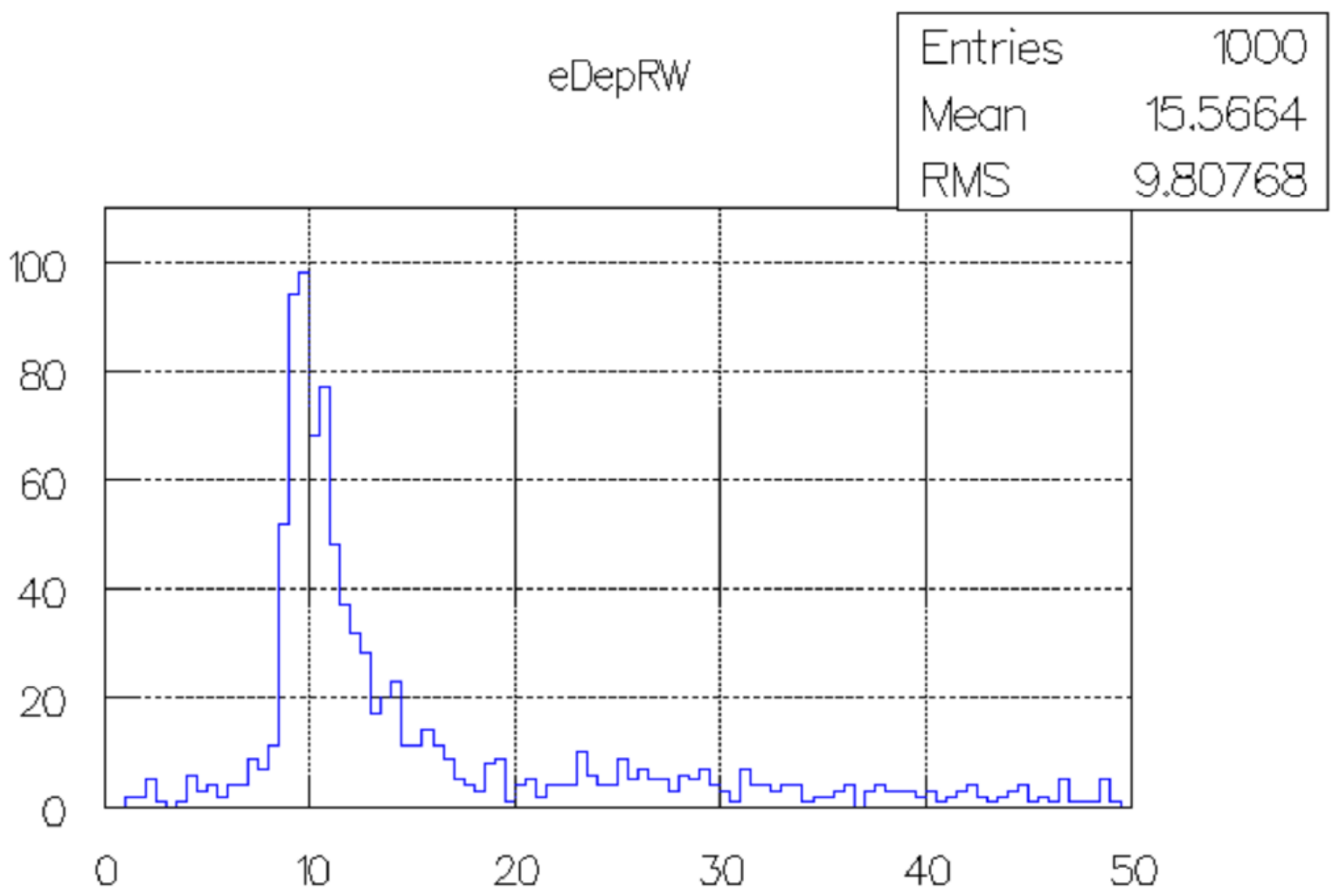
  /score/analysis/1D/set *<nBin> <low> <high> <unit>*

  /score/analysis/plot

  /run/beamOn 1000

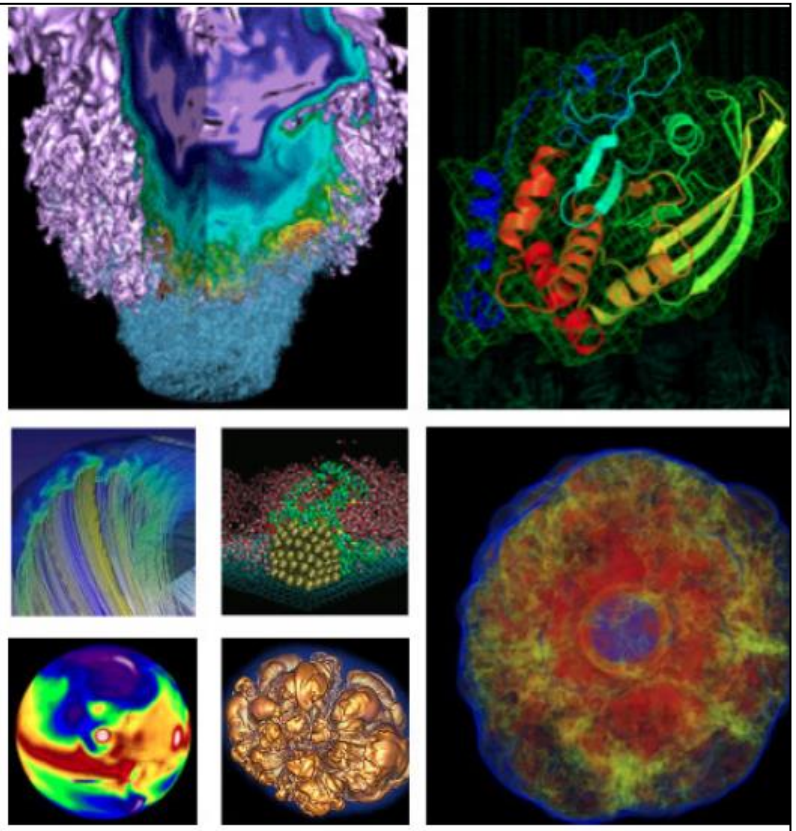
Note: command names are still tentative


rsg4main
Scene tree, Help, History
Scene tree
Help
History
Search :
Command
control
units
gui
geometry
tracking
particle
event
cuts
run
random
process
score
rsim
persistency
material
physics_lists
analysis
heptst
physics_engine
gun
vis
Useful tips
viewer-0 (OpenGLStoredQt)
Basic/B4a
geometry
Output
/score/create/realWorldLogVol Gap
/score/quantity/energyDeposit eDepRW MeV
/score/close
/score/analysis/1D/create Gap eDepRW
/score/analysis/1D/set 100 0. 50. MeV
/score/analysis/plot
/run/beamOn 1000


eDepRW
Entries 1000
Mean 15.5664
RMS 9.80768
100
80
60
40
20
0
0
10
20
30
40
50

Geant4 + timemory

Updates since Lund



National Energy Research Scientific Computing Center

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

Jonathan R. Madsen

✉ jrmadsen@lbl.gov

National Energy Research Scientific Computing Center
Lawrence Berkeley National Laboratory

September 25, 2019

- Last year, presented introduction of timemory into Geant4
  - Lightweight "in-situ" package for automating performance analysis
  - Designed to be very easy to use
  - Used macro `TIMEMORY_AUTO_TIMER("")` macro at beginning of certain high-level functions in Geant4
  - When not compiled with timemory support ⇒ empty macro
- Capabilities were limited: wall/user/sys/cpu/cpu-util timing + peak/page memory in 3 possible bundles (timing, memory, timing and memory)

- Library essentially underwent a complete re-write using CRTP
- The capabilities have increased **significantly**
  - HW counters, Roofline, gperftools, `rusage`, many more...
  - Still uses `TIMEMORY_AUTO_TIMER("")` but more available
  - Provides fully capabilities from Caliper and TAU
- Library is now header-only for all the previous capabilities
- Measurements can be arbitrarily assembled into any combination
- Always-on bundles (tuple), run-time optional bundles (list), and pair of those bundles (hybrid)
- GitHub: github.com/NERSC/timemory
- Documentation: timemory.readthedocs.io

Listing 1: Sample wall-clock Text Output

```
> [cxx] cmsExpMT                                                    :  330.661 sec real,   1 laps
> [cxx] |_BeamOn@'G4RunManager.cc':267                             :  319.992 sec real,   2 laps
> [cxx]   |_RunInitialization@'G4RunManagerKernel.cc':660           :  167.113 sec real,   2 laps
> [cxx]     |_BuildPhysicsTables@'G4RunManagerKernel.cc':791       :   97.510 sec real,   1 laps
> [cxx]   |_RunInitialization@'G4RunManager.cc':319                 :    0.000 sec real,   2 laps
> [cxx]   |_DoEventLoop@'G4RunManager.cc':376                       :    1.242 sec real,   2 laps
> [cxx]     |_InitializeEventLoop@'G4MTRunManager.cc':259           :    1.242 sec real,   2 laps
> [cxx]       |_DoWork@'G4WorkerRunManager.cc':608                  : 2103.221 sec real,  12 laps
> [cxx]         |_BeamOn@'G4RunManager.cc':267                      : 1819.951 sec real,  24 laps
> [cxx]           |_RunInitialization@'G4WorkerRunManager.cc':138   :    0.322 sec real,  24 laps
> [cxx]             |_RunInitialization@'G4RunManagerKernel.cc':660 :    0.278 sec real,  23 laps
> [cxx]               |_BuildPhysicsTables@'G4RunManagerKernel.cc':791 :  0.213 sec real,  10 laps
> [cxx]             |_RunInitialization@'G4RunManagerKernel.cc':660 :    0.025 sec real,   1 laps
> [cxx]               |_BuildPhysicsTables@'G4RunManagerKernel.cc':791 :  0.024 sec real,   1 laps
> [cxx]           |_DoEventLoop@'G4WorkerRunManager.cc':221         : 1692.912 sec real,  22 laps
> [cxx]             |_ProcessOneEvent@'G4WorkerRunManager.cc':266   : 1692.886 sec real, 122 laps
> [cxx]               |_GenerateEvent@'G4WorkerRunManager.cc':279   :    0.034 sec real, 121 laps
> [cxx]             |_TerminateOneEvent@'G4RunManager.cc':423       :    0.002 sec real, 100 laps
> [cxx]           |_DoEventLoop@'G4WorkerRunManager.cc':221         :    0.000 sec real,   1 laps
> [cxx]             |_ProcessOneEvent@'G4WorkerRunManager.cc':266   :    0.000 sec real,   1 laps
> [cxx]               |_GenerateEvent@'G4WorkerRunManager.cc':279   :    0.000 sec real,   1 laps
```

Non-traditional Use Cases

- There is no restrictions on nesting ⇒ start and stop in any order
- This enables timemory to be used within Geant4 in *very* non-traditional performance analysis contexts
  - For the Geant ECP Project, attached timemory to all `G4Track` instances ⇒ post-processed JSON to analyze per-particles processing time w.r.t. number of instances

```
1 G4Track::G4Track(G4DynamicParticle* dynParticle,
2                  G4double aValueTime,
3                  const G4ThreeVector& aValuePosition)
4 {
5     auto pname = dynParticle->GetDefinition()->GetParticleName();
6     trackTimer = new G4AutoTimer(pname, 0);
7 }
```

Non-traditional Use Cases

```
> [cxx]          |_e-          :    0.000 sec real,  1 laps, depth 38
> [cxx]        |_gamma         :    0.000 sec real,  2 laps, depth 37
> [cxx]        |_e-            :    0.000 sec real,  2 laps, depth 37 (exclusive:  30.3%)
> [cxx]          |_e-          :    0.000 sec real,  1 laps, depth 38 (exclusive:  28.4%)
...
> [cxx]            |_e+        :    0.000 sec real,  1 laps, depth 47
> [cxx]            |_gamma     :    0.000 sec real,  1 laps, depth 47 (exclusive:  23.2%)
> [cxx]              |_gamma   :    0.000 sec real,  1 laps, depth 48
> [cxx]              |_e-      :    0.000 sec real,  1 laps, depth 48
> [cxx]            |_e-        :
> [cxx]              |_e-      :
> [cxx]        |_proton        :
> [cxx]          |_neutron     :
> [cxx]            |_gamma     :
> [cxx]              |_gamma   :
> [cxx]              |_e-      :
> [cxx]            |_e-        :
> [cxx]              |_e-      :
> [cxx]          |_proton      :
> [cxx]            |_Al27      :
> [cxx]              |_gamma   :
> [cxx]              |_e-      :
> [cxx]              |_gamma   :
> [cxx]              |_e-      :
> [cxx]    |_neutron           :
> [cxx]      |_gamma           :
> [cxx]        |_neutron       :
> [cxx]          |_W183        :
> [cxx]      |_e-              :
> [cxx]        |_e-            :
> [cxx]    |_proton            :
> [cxx]      |_Al27            :
```


Non-traditional Use Cases

PARTICLE	WALL	CPU	PERC_WALL	PERC_CPU	AGG_WALL	AGG_CPU	LAPS
e-	5092082.068	4523577.476	59.901 %	59.140 %	59.901 %	59.140 %	24999623
gamma	1453282.218	1394334.528	17.096 %	18.229 %	76.996 %	77.369 %	10511498
neutron	470668.394	413414.379	5.537 %	5.405 %	82.533 %	82.774 %	656765
proton	458564.377	400630.414	5.394 %	5.238 %	87.927 %	88.012 %	509989
pi+	279696.042	236695.434	3.290 %	3.094 %	91.218 %	91.106 %	8407
pi-	231377.604	210396.925	2.722 %	2.751 %	93.939 %	93.857 %	9533
anti_proton	164314.328	149861.520	1.933 %	1.959 %	95.872 %	95.816 %	745
pi0	125214.507	111033.410	1.473 %	1.452 %	97.345 %	97.268 %	8280
e+	53113.975	52151.025	0.625 %	0.682 %	97.970 %	97.949 %	935320
O16	38601.293	36932.746	0.454 %	0.483 %	98.424 %	98.432 %	207421
Al27	17994.699	14117.837	0.212 %	0.185 %	98.636 %	98.617 %	61435
C12	11972.456	12024.187	0.141 %	0.157 %	98.777 %	98.774 %	65480
Cu63	11202.182	10357.312	0.132 %	0.135 %	98.908 %	98.909 %	85695
eta	10178.519	10093.700	0.120 %	0.132 %	99.028 %	99.041 %	438
Fe56	10167.581	7785.565	0.120 %	0.102 %	99.148 %	99.143 %	49170
kaon+	5225.957	5197.097	0.061 %	0.068 %	99.209 %	99.211 %	472
kaon0L	5001.100	4928.734	0.059 %	0.064 %	99.268 %	99.276 %	366
mu+	4949.642	4260.876	0.058 %	0.056 %	99.326 %	99.331 %	3332
mu-	4887.122	4202.955	0.057 %	0.055 %	99.384 %	99.386 %	646
kaon0S	4704.240	3966.241	0.055 %	0.052 %	99.439 %	99.438 %	346