# Geant4 performance aspects in ALICE Run3

Sandro Wenzel (CERN), Ivana Hrivnacova (IJClab IN2P3/CNRS)
For the ALICE Collaboration

28th Geant4 Collaboration Meeting, Sapporo,
26 September 2023

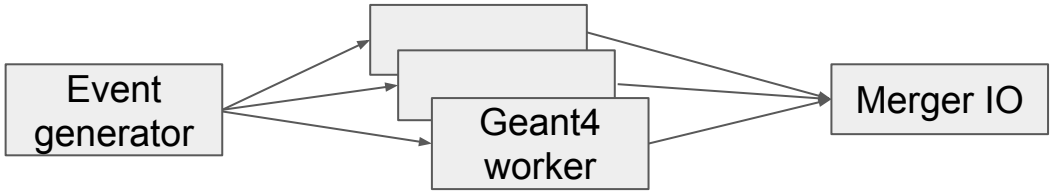
# Setting the scene

1. ALICE is using Virtual Monte Carlo (VMC) ecosystem for detector simulation
    - User code based on TGeo + VMC API (scoring); No direct coding against Geant4
    - Geant4 can be used as one transport backend (next to Geant3 and FLUKA)
2. Geant4 v11.0.4 is the default simulation engine since LHC Run3
    - Previously, Geant3 was used
    - These slides reports first benchmarks from Run3 production system
    - Good baseline but not yet fully optimized (focus on other parts of software). There is room for improvement.
3. Geant4 is using the TGeo navigator
    - Translation between TGeo geometry structure to G4 necessary (in-memory tables)
    - Initialization overhead + some small additional runtime cost
    - Currently no VecGeom usage possible since TGeo not yet able to use VecGeom

# Main Geant4 configuration choices

- NystromRK4 stepper
- Physics lists:
  - FTFP_BERT_EMV+optical
  - FTFP_BERT_EMV+optical+biasing (INCLXX physics in ITS region)

# Architectural overview of ALICE detector sim




Services operate in parallel and asynchronously

- The o2-sim detector simulator is an executable that spawns various sub-processes as microservices that communicate via messaging
  - Event generator
  - Geant transport workers
  - Data merger for simulation output (final ROOT IO)
- Parallel Geant worker processes collaborate on simulation at the event level
  - Workers transport **sub-events** (or event chunks)
  - Processes are **memory shared due to late fork** (similar to multi-threading mode)
  - Parallelism works seamlessly also for FLUKA and Geant3

# Recent profile of default Geant4 (no ZDC) - most important functions

| | | |
|---|---|---|
| 4.49% | libgeant4vmc.so | TG4CachedMagneticField::GetFieldValue |
| 4.20% | libO2Field.so | o2::math_utils::Chebyshev3D::Eval |
| 4.10% | libG4geometry.so | G4NystromRK4::Stepper |
| 2.72% | libG4geometry.so | field_utils::relativeError2 |
| 2.43% | ld-linux.so.1 | _dl_tlsdesc_dynamic |
| 2.33% | libGeom.so.6.28.04 | TGeoSubtraction::Contains |
| 1.65% | libGeom.so.6.28.04 | TGeoUnion::Contains |
| 1.60% | libm.so.6 | __sincos |
| 1.53% | libGeom.so.6.28.04 | TGeoNavigator::Safety |
| 1.52% | libGeom.so.6.28.04 | TGeoVoxelFinder::GetNextCandidates |
| 1.39% | libBase.so.18.4.9 | FairMCApplication::Stepping |
| 1.38% | libm.so.6 | __atan2_finite@GLIBC_2.17 |
| 1.37% | libG4tracking.so | G4SteppingManager::DefinePhysicalStepLength |
| 1.37% | libm.so.6 | pow@@GLIBC_2.29 |
| 1.26% | libGeom.so.6.28.04 | TMath::BinarySearch<double> |
| 1.20% | libGeom.so.6.28.04 | TGeoTranslation::MasterToLocal |

understandable:

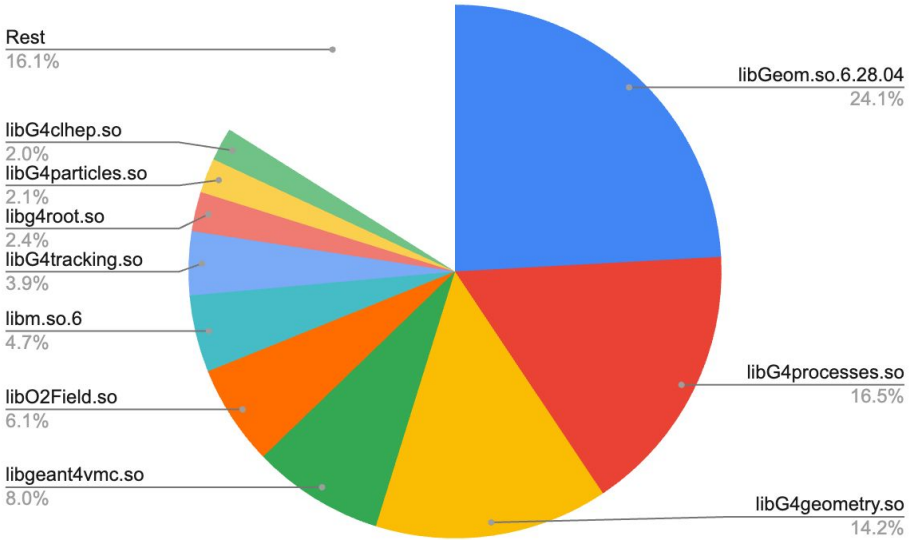
- Field calls (cached + call to ALICE field) on top
- Then stepping and some geometry shapes
- Pow + sincos math functions

unexpected:

- Overhead from thread-local storage (MT mode of Geant4)

"Perf record on 200 pp or 20 PbPb events on Ubuntu 22.04 (ARM)" .. PbPb events are similar

# Contributions by category




- TGeo geom calls most significant (24%)
  - Would clearly benefit from VecGeom replacement
- Field calc totals to ~12%
  - libO2Field + cached field in libgeant4vmc
- Stepper (libG4geometry) 14.2%
- Physics processes (16.5%)
- “Rest” is sum of tiny contributions

We should benefit most from algorithmic improvements or tunings in magnetic field access and geometry

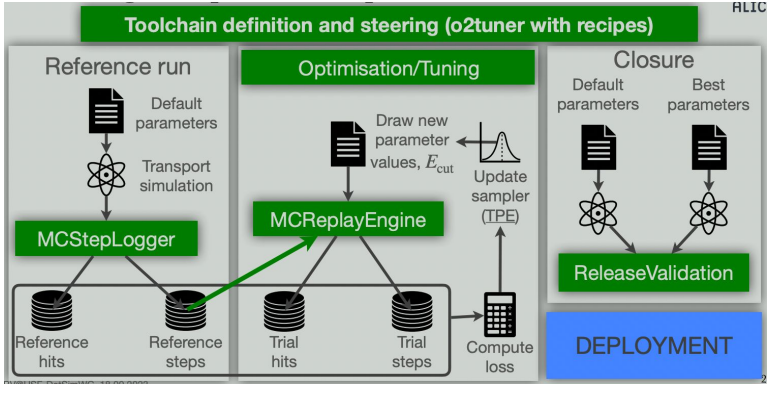
We would benefit from putting into production templated Stepper code (no virtual functions to field, equations, etc.) in Geant4 (old R&D project)

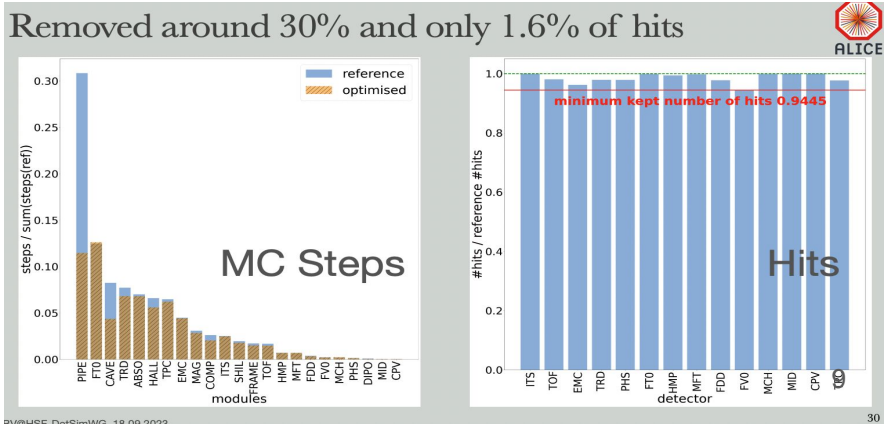
# Advantage of non-MT Geant4 build

- So far we compiled Geant4 with MT mode enabled, although we are not using the MT feature
- Benchmarks here reveal that there is a ~2% cost associated to handling thread-local storage
- Switch off MT mode will be advantageous

# Optimizing MC steps (“process cuts”)

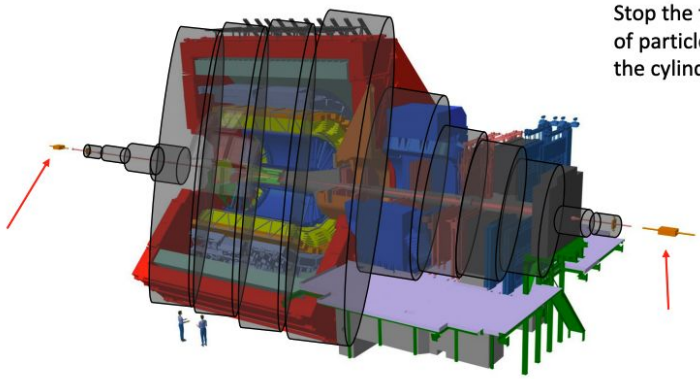
- Started a campaign / project to reduce number of “useless” MC steps : **Avoid work**
  - Tracks dying immediately
  - Tracks leaving no hits
  - etc.
- To this end, an optimization framework developed which automatically tunes “production cuts” under the constraint to keep hit output constant
  - See recent talk at CHEP or HSF meeting
- **So far achieved 30% reduction** in total steps by tuning electromagnetic production thresholds our PIPE geometry
- **To be extended** to more material parameters and **to be put into production**



# Optimizing MC steps (“geometry region killer”)

- Similar project on idea to absorb/kill tracks early once they exit a certain core part of the detector
- Recent summer student project to find optimal arrangement of “track-killing regions”
  - Or in other words a tightly fitting transport region
- Use of same auto-optimization framework to find optimal size of “bounding” cylinders
- Preliminary potential for another 10% reduction in steps



# Updates in Geant4 VMC Cuts

- Motivation
  - Computation of range cuts from VMC (energy) significantly slowed down total initialisation time of Geant4 simulation
  - Idea: dump computed values in a file and then load them from the file to avoid calculation each time
- Fixes
  - With dumping the calculated region data in the file, a few problems in the current procedure were spotted and fixed
  - The regions are constructed within a loop over logical volume store, new region is created per each material and is assigned to all volumes that have this material.
  - Fixed redundant recalculating of the ranges for the same materials in the loop
    - Significantly improves computation time: total time of “pythia8pp” run with 1 event : 95s -> 35 s
  - Plus some more less important fixes
    - Improved match of the calculated cut and the VMC cut

# Updates in Geant4 VMC Cuts - 2

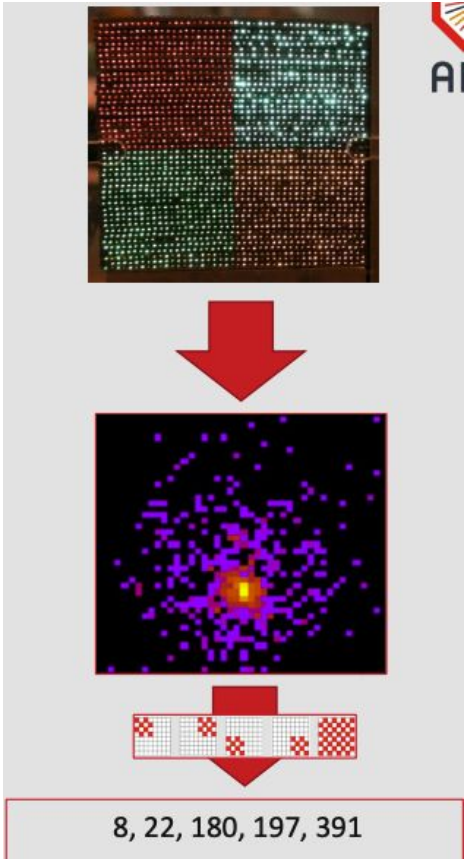
- Improved the “check” option the can be used to detect the regions (materials) where the calculated energy cuts differ within a given “tolerance”
  - This option can be used to tune the optimal number of iterations used in the computation of the ranges, that can be set by user via /mcRegions/setPrecision value
  - Previously used default value 2, could be increased to 5 that gives better match; higher value does not give better result in performance
- Improved match of the calculated cut and the VMC cut:
  - The number of inconsistencies between cut from ranges energy cuts (tolerance 0.01) found: 169 (old precision=2) => 19 new (new precision=2)

# More ideas that are on the list

- Magnetic field seems expensive for us, so tuning access to it is a natural idea
  - Avoid magnetic field calls whenever possible
    - For instance, check that all materials outside of field are marked as “non-field”
  - Play with parameters for caching etc.
- Stop tracks based on more deeply learned ML criteria
  - kill particles early based on particle properties (location, direction, material, …)
- VecGeom integration into TGeo or use of native Geant4 geometry

# FastSim of ZDC

- ALICE is investigating also the use of ML techniques to replace full simulation in the ZDC (zero degree calorimeter)
- When ZDC is switched on, the Geant4 transport time approx doubles - triples due to showering
- Working on ML models that **avoid any transport to ZDC at all**
  - “Predict ZDC output = 2D image directly **based on primary particle properties”**
- Good recent progress with GAN models but more work towards production + validation needed
- Technique would allow to include ZDC without additional cost




ZDC has 2D arrangement of optical fiber tubes

Impeding particles induce photon showers which are essentially 2D images with color == number of photons

The idea is to generate these images with ML tools

Link to recent status update