GeantV status and performance

Andrei Gheata

23rd Geant4 Collaboration Meeting

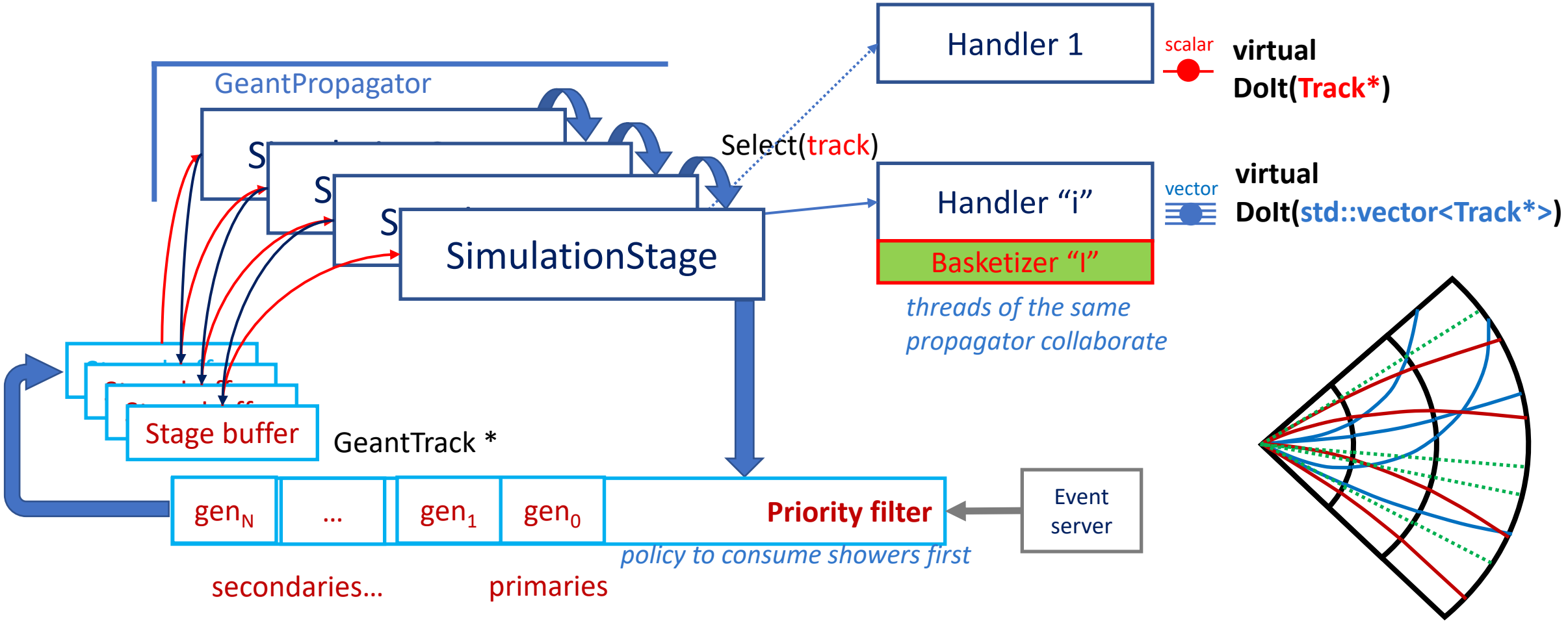
Lund, 27-31 August 2018

Outlook

- Latest vectorization work and features
- Latest results and comparison with Geant4
- Ongoing work and plans
- Next -> how can we use some of this in Geant4

GeantV multi-particle processing

Both scalar/vector flow are supported



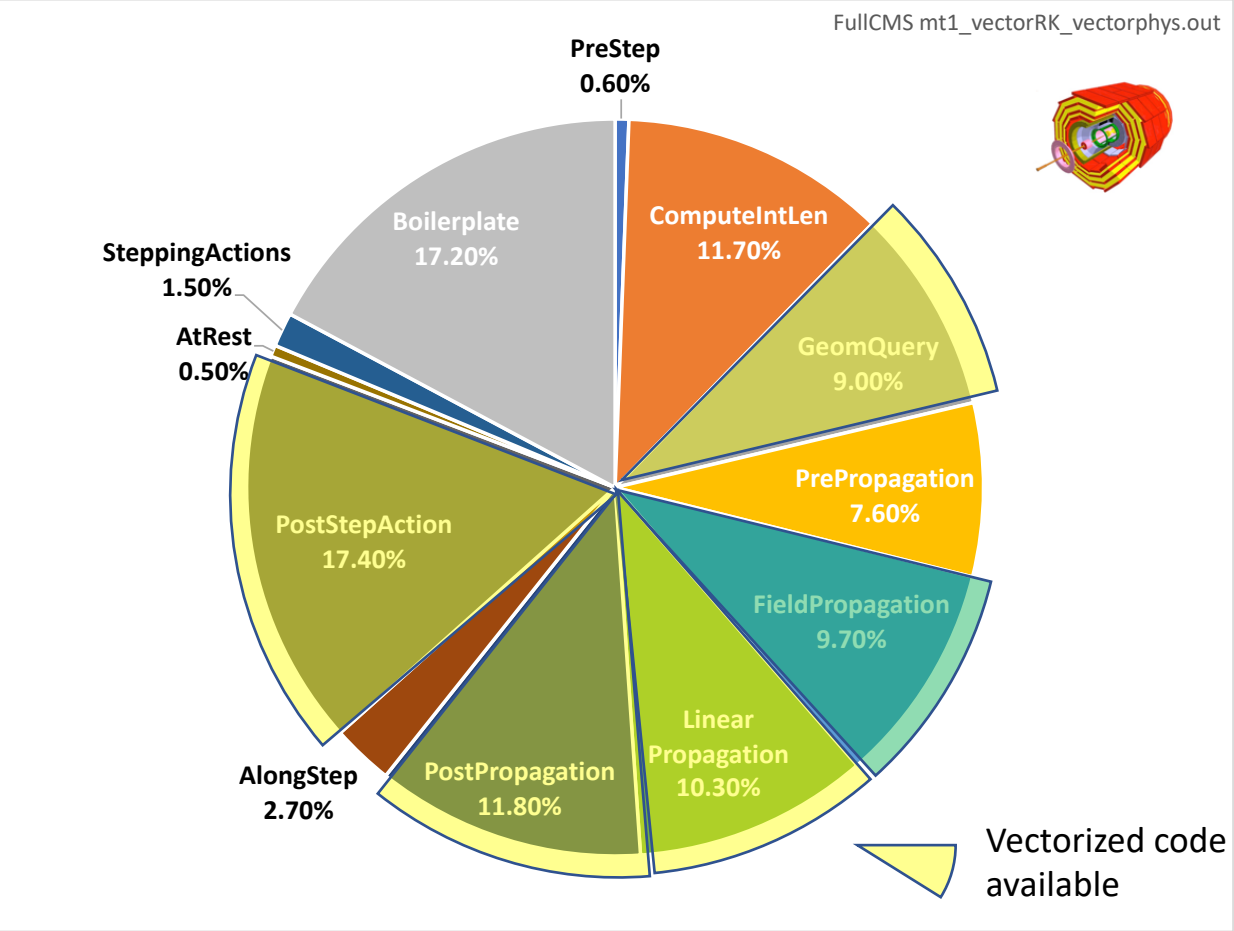
Vectorization work

- Particle regrouping per simulation stage, pushing buffer of tracks to stage handlers
  - Handlers may “basketize” and send tracks to vectorized algorithms
  - Dynamic basketizing: fired/flushed ratio (FFR) > 2
- Currently basketized handlers
  - Geometry query: one handler per volume
  - Magnetic field driver (RK or helix): one handler
  - Post-step physics processes (EM): 12 handlers (1 per model)
  - Post-step continuous process handlers (MSC): 1 handler
- Possible to switch on/off basketizing on per handler basis

Preliminary profiling/benchmarking

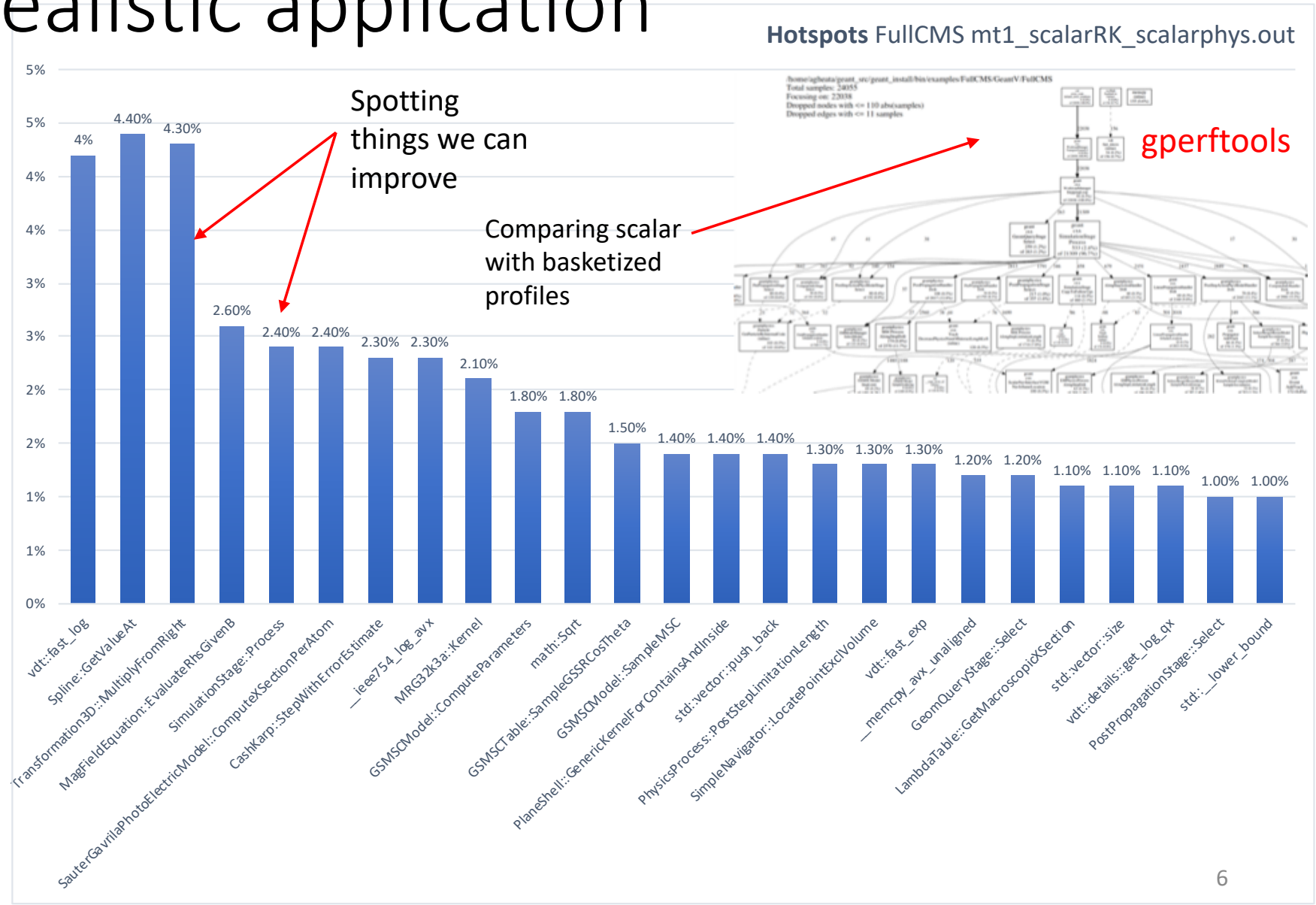
- New folders examples/benchmarks/[example]
  - For now only for TestEm3 (simple calo mockup) and FullCMS
    - TestEm3: 100 events / 1 primary each, 100 GeV electrons
    - FullCMS: 10 events / 10 primaries each, 100 GeV electrons
  - Running GeantV and Geant4 with similar settings
- Several configurations for GeantV
  - Magnetic field: ON/OFF, RK/helix
  - Basketizing ON/OFF per component
- Thorough benchmarking against equivalent Geant4 application ongoing
  - Study basketizing efficiency - baseline for future optimizations

Top-level profile of basketized GeantV simulation of CMS

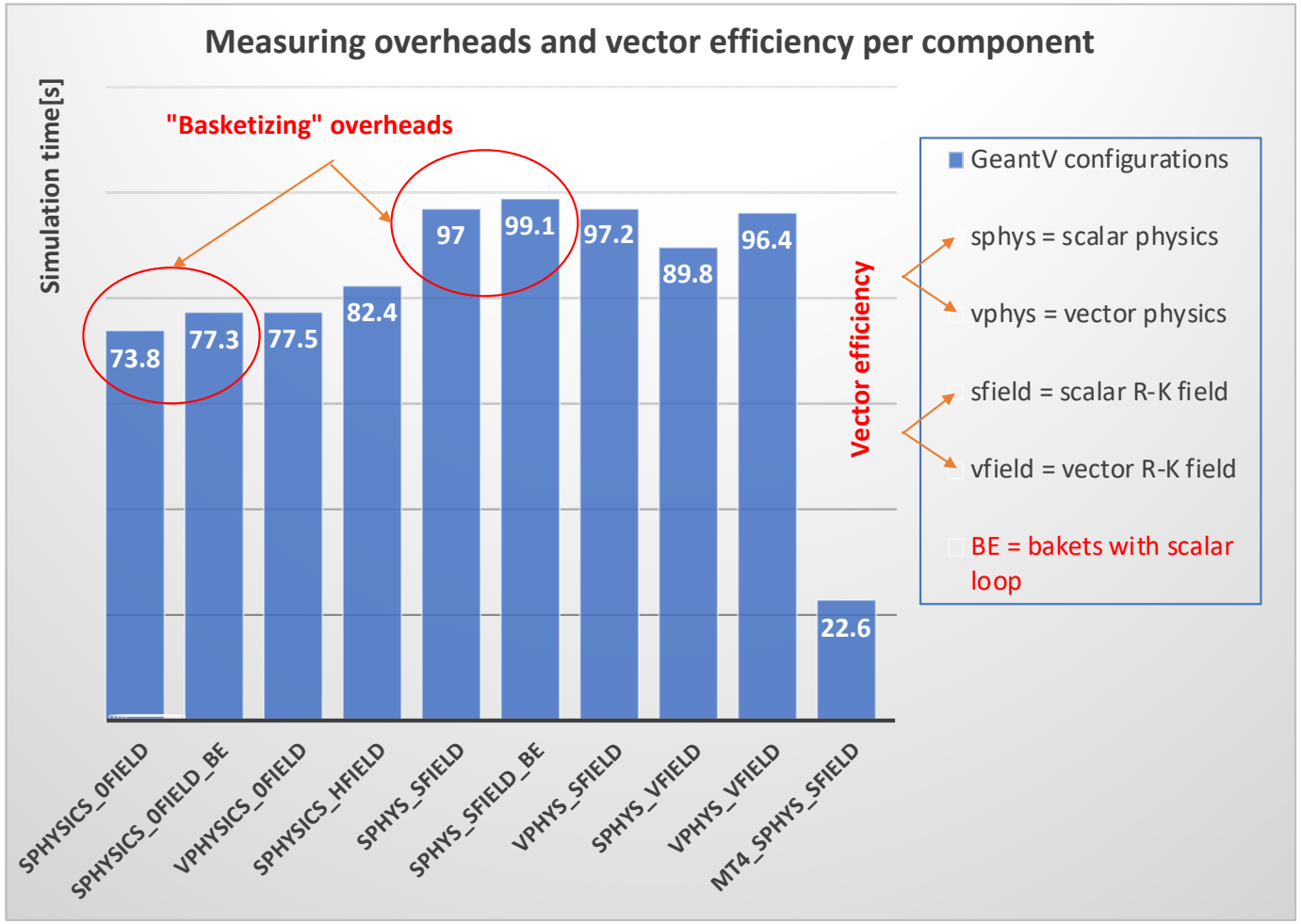


Profiling a realistic application

- Understanding hotspots
- Understanding basketization overheads
- Working on reducing scalar bottlenecks

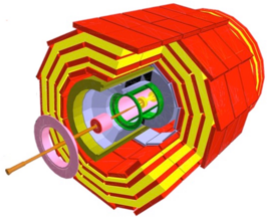


Benchmarking configurations



- Magnetic field ON/OFF, propagator type
  - Understand behavior of different component implementations
- Basketizing ON/OFF (Physics, field, geometry in future)
  - Understand vectorization efficiency
- Basketize but perform scalar loops
  - Understand framework bottlenecks for basketizing
- MT simulation
  - Understand scalability, memory footprint
- Corresponding profiles generated
- Putting in place systematic performance monitoring

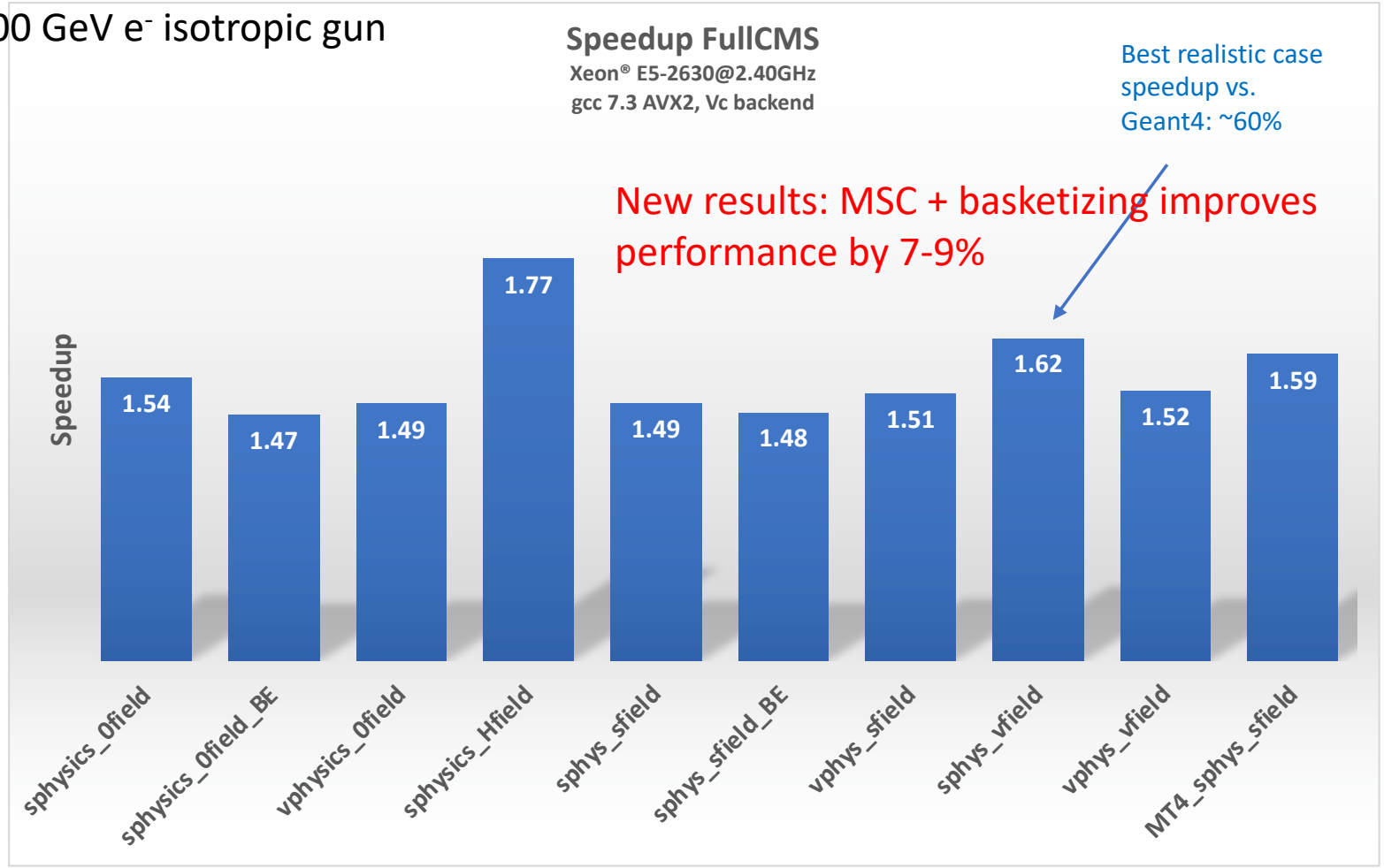
Current generic performance benchmark



CMS geometry, 100 GeV e^- isotropic gun

Geant4 10.4 Rev2

- Generic benchmark with standard EM processes for e^+, e^-, γ, field ON/OFF
- Full geometry imported from GDML, including production cuts
- Scoring:
  - mean energy deposit
  - mean charged and neutral step lengths
  - mean number of steps
  - mean number of secondary particles

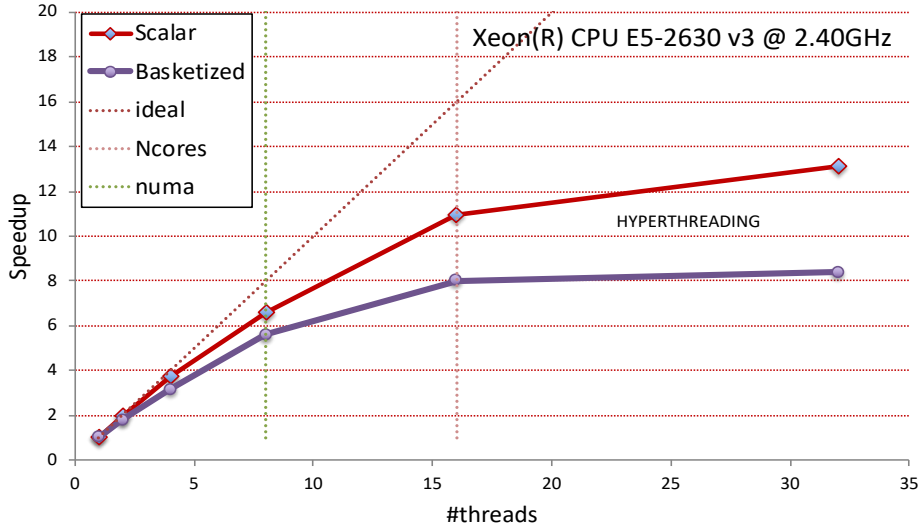


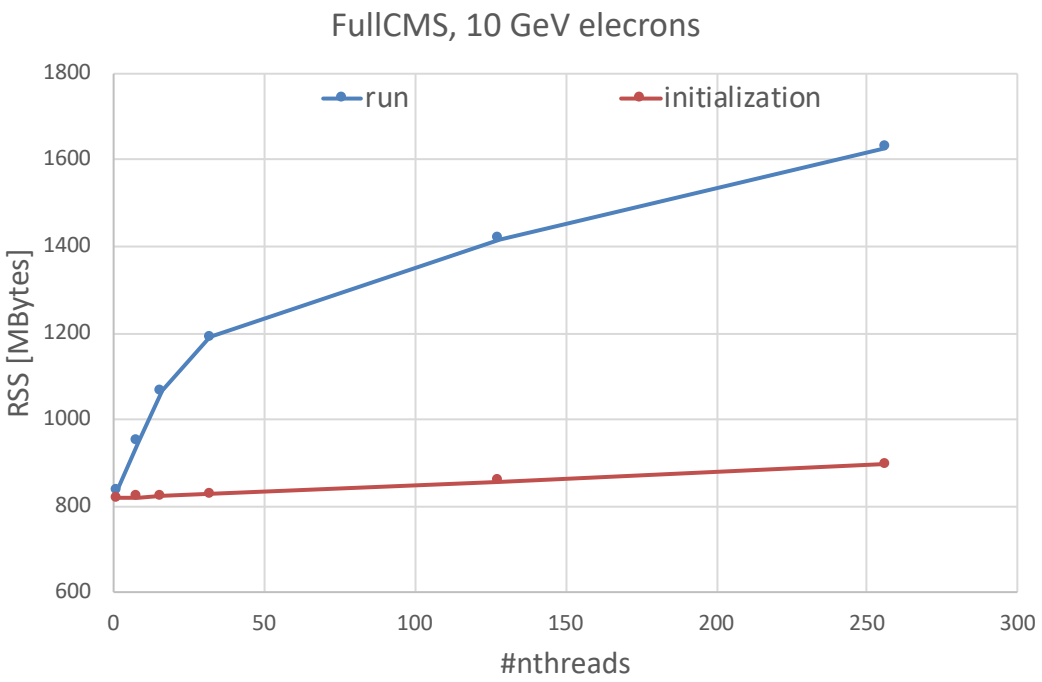
MT mode

- Currently: fine grained parallelism
  - Tracks imported concurrently from an event server
  - Handlers/basketizers per propagator, shared by many threads
    - Typically one propagator per NUMA node, having as many workers as HW threads
- Passive mode: thread-safe re-entrant main transport method
  - User creating and passing as input an “event set”
  - Current policy: same thread filling the event set will return after transport
    - Changing to a user notification via callable for better efficiency
- MT mode + basketizing
  - Possible, but still not very efficient (~30% basketization achieved)
  - Considering a less fine grained track-level parallelism, where tracks from same event are handled by the same thread
  - Basketizers currently shared by many threads, but for the most efficient cases (field, MSC) population from a single worker is enough

Current MT performance

- Tracks being exchanged between threads introduce Amdahl overheads
- Basketized MT version not yet tuned: loss in vector efficiency
- Evolution: some basketizers per thread, or even events fully owned by a thread





- Memory footprint higher than G4 (~3x)
- Size of data structures just after initialization is the major contributor
- Rather good behavior with increasing #threads

Ongoing work

- Reducing basketization overheads
  - Light (per thread) basketizing
  - No tracks exchanged among threads
- Geometry optimizations
  - Specialized volume navigators (reducing scalar bottlenecks by de-virtualizing + caching)
- Physics optimizations + new features
  - Tuning usage of sampling methods depending on model and energy/material
  - Use of floating point instead of double case by case
- Aiming for a factor of ~2x for the full CMS benchmark for the beta tag end 2018 compared to Geant4
  - Out of which ~1/3 from basket vectorization, the rest from improved locality, less virtual function calls, …
  - Detailed Geant4 profiling will give a better picture of where gains are coming from