MANCHESTER 1824

The University of Manchester

Geant4 ASSOCIATES INTERNATIONAL

Experts in Radiation Simulation

Parallellising Geant4

John Allison
Manchester University
and
Geant4 Associates International Ltd

Geant4

- Open source radiation simulation toolkit
- Can handle very complex geometries
  - Millions of volumes
  - Isotopic composition of materials
- Wealth of physics
  - Electromagnetic physics down to a few eV
  - Choice of physics models
  - Cell chemistry (biological effects)
- and tracks particles and ions through geometry description
  - interactions, decay, multiple scattering, etc.

History

- 1994: CERN-KEK initiative in object oriented design
  - GEANT3 (Fortran) unmaintainable; frozen
  - Particle tracking: a natural candidate for object oriented programming
    - Particles, tracks, volumes, materials: all can be objects
    - Different processes share common interface—similarly differently shaped volumes—so one can write code that tracks a particle of unknown type through a volume of unknown shape experiencing a list of possible processes of unknown type. Amazing!!
    - Each particle has a list of possible processes
    - Track *this* particle through *this* volume selecting a *particular* process with the smallest "distance to happening"
    - "Distance to happening" has an exponential distribution according to material and cross-sections or, in the case of the transportation process, is the distance to the next geometrical boundary
    - Energy loss and multiple scattering along the step
- 1994-1999: CERN R&D project RD44
  - C++ chosen
- 1999: First public release.
- Roughly 100 physicists (and a few computer scientists) contributing all or part of their time
  - 27 FTEs at the last count
- 2012: Latest release: version 9.6
- 2013: Multi-threaded version 10 planned

Toolkit

- A set of libraries
  - Download and install (CMake) from geant4.cern.ch
- User writes C++ code to define:
  - Detector
  - Particles and processes
  - Primary particle moment vectors
- Optionally:
  - Scoring
  - Recording
  - Track stacking
  - Drawing

```
...
G4RunManager* runManager = new G4RunManager;
// Three mandatory user actions...
// Geometry of detector and apparatus
runManager->SetUserInitialization(new MyDetectorConstruction);
// Set of physics processes for each desired particle
runManager->SetUserInitialization(new MyPhysicsList);
// Generate particle momentum vectors to be tracked each event
runManager->SetUserAction(new MyPrimaryGeneratorAction);
// ...plus optional user actions, such as scoring, recording at
// end of event or run, then either
// a) start interactive session (type "/run/beamOn 100")
// b) read script of such commands, or (c)
runManager->BeamOn(100);  // 100 events
...
```

Applications

- Distributed as a toolkit
  - User must build his/her own application
  - Every particle and nuclear physics experiment has one
- But public applications are available
  - ESA: MULASSIS in the web-based SPENVIS environment
    - Users can define simple geometries and get doses
  - Medicine: GATE, GAMOS, TOPAS

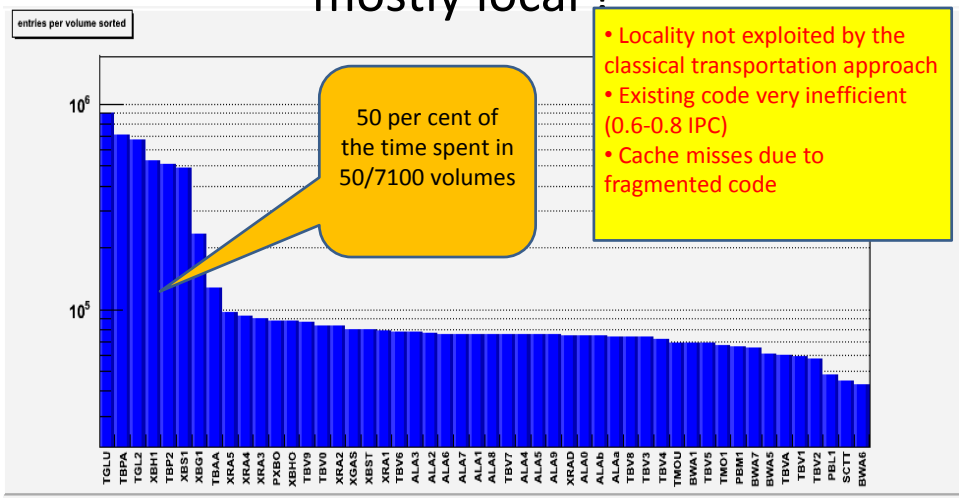
Parallellisation

- Application-level parallellisation
  - de facto practice
    - Each job gets a different random number seed
    - Output files concatenated afterwards
  - Billions of events on tens of thousands of CPUs
    - Particle physics's computing GRID
- Multi-threading
  - Prototype conversions of Geant4 9.4, 9.5 and (soon) 9.6
    - Use pthreads
    - Threads share geometry
    - 25 MB per thread (compared to 2GB per application!)
    - Good scaling up to number of available cores—see poster
    - 18% overhead compared to non-threaded version (was 30% on poster)
  - Geant4 version 10 will be multi-threaded
    - Non-threaded version will still be available by compile-time switch
    - Workshop 10th-15th January 2013
    - May use the std::thread class and associated classes (std::mutex, std::lock, etc.) in the C++11 (2011) standard
      - Not yet available on all platforms

Parallellisation (2)

- Vectorisation
  - Not a good paradigm for random tracking of particles
  - Different tracks have totally different outcomes in terms of number of steps, contribution to score, etc.
  - One may start with a vector of tracks but the outcomes not easily vectored
  - A study involving *track grouping* and vectorisation *within* a volume
  - Track grouping is an overhead
- Graphics Processing Units (GPUs)
  - Intrinsically massively parallel
  - Not easily harnessed
  - Some investigations in progress supported by NVIDIA

Simple observation: HEP transport is mostly local !



ATLAS volumes sorted by transport time. The same behavior is observed for most HEP geometries.

Other issues

- Reproducibility, i.e., the ability to recompute an event exactly
  - Needed for:
    - debugging
    - investigating rare events
  - Requires reproducibility of random number sequence
  - A problem if the random number generator is shared in a multi-threading environment
  - Would require sophisticated pre-allocation of random number seeds for each event and thread-local random number generators

Summary

- Geant4 is compute-intensive
- It needs to capitalise on computing innovations (hardware and software)
- Application-level parallellism works well
  - 2 GB per application when memory is getting cheaper is not much of a problem
- Multi-threading in multi-core environments is an obvious way to go
  - A single job can exploit multi-core technology
  - Useful for quick looks (e.g., ESA's MULASSIS or even for developers on multi-core laptops or servers)
  - Hand crafted prototypes MT versions of 9.4 and 9.5 have been released
  - A final hand crafted MT version of 9.6 will soon be released
  - Main development branch version is imminent
  - Beta release: June 2013
  - Full release: November 2013
- Watch this space!!