

**Fermilab** 



### Geant4

RENCI

Geant4 is a toolkit for the simulation of particles passing through and interacting with matter. Its areas of application include high energy, nuclear and accelerator physics, as well as studies in space science, material science, medicine and biology, and also in security and industrial applications. In high energy physics, the Geant4 toolkit fulfills a critical need for the simulation of detectors at the LHC and at other existing and future experiments and facilities.



Geant4 is maintained and further developed by an international collaboration, which consists of more than 100 physicists and computer scientists. U.S. involvement in the development of Geant4 has been substantial since its early stages, and has increased with time. Several key Geant4 functionalities, including core framework, hadronic physics and visualization, are lead by the SLAC team, while major contributions in the key areas of hadronic physics and computing performance are made by the Fermilab team.

| \varTheta 🔿 🔿 hpcviewer: cmsEx  | pMT                   |                 |          |     |
|---|-----------------------|-----------------|----------|-----|
|   |                       |                 |          | ' 🗆 |
| 😪 Calling Context View 💫 Callers View 📅 Flat View                               |                       |                 |          | ' 🗆 |
|   |                       |                 |          |     |
|   | BABL TOT CVC-Curr (I) | DADI TOT CVC-C  | ···· (E' |     |
| Scope   | PAPI_TOT_CYC:Sum (I)  | PAPI_TOT_CTC:SL | Im (E, ♥ |     |
| Experiment Aggregate Metrics  | 1.73e+12 100          | 1.73e+12        | 100 %    |     |
| P_leee/54_log   | 1.51e+11 8.83         | 1.51e+11        | 8.8%     |     |
| CAElasticHadzNuclausHEvHadzNucDifferCrSec(int_int_double)                       | 1.180+11 0.83         | 5 1.18e+11      | 6.88     |     |
| C4Navigatory: a cateClobalPointAndSetun(CLHEP::Hen3Vector const&                | 1.680+11 9.73         | 5.73e+10        | 3.38     |     |
| inee754_stan2   | 2 000+10 2 20         | 4.29e+10        | 2.3%     |     |
| C4PhysicsVector::Value(double_unsigned long&) const                             | 5.90e+10 2.3          | 3.90e+10        | 2.3%     |     |
| C4CrossSectionDataStore::CetCrossSection(C4DynamicParticle const                | 1 77e+11 10 29        | 2 94e+10        | 1 7%     |     |
| C4SteppingManager::Stepping()   | 1 42e+12 82 19        | 2.540+10        | 1.5%     |     |
| C4SteppingManager::DefinePhysicalStepLength()                                   | 6.00e+11 34.8         | 2.41e+10        | 1.4%     |     |
| C4VoxelNavigation::ComputeStep(C1HEP::Hen3Vector const& C1HEP                   | 1.27e+11 7.39         | 2.39e+10        | 1.4%     |     |
| G4Navigator::ComputeStep(CLHEP::Hep3Vector.const&, CLHEP::Hep3                  | 1.85e+11 10.75        | 2.25e+10        | 1.3%     |     |
| ► G4hPairProductionModel::ComputeDMicroscopicCrossSection(double                | 8.72e+10 5.19         | 2.14e+10        | 1.2%     |     |
| ► G4HadronCrossSections::CalcScatteringCrossSections(G4DynamicPar               | 2.06e+10 1.2          | 1.91e+10        | 1.1%     |     |
| G4ParticleChange::CheckIt(G4Track const&)                                       | 1.78e+10 1.0          | 1.78e+10        | 1.0%     |     |
| CLHEP::RanecuEngine::flat()   | 1.72e+10 1.0          | 1.72e+10        | 1.0%     |     |
| ► G4CrossSectionDataStore::GetCrossSection(G4DynamicParticle const <sup>2</sup> | 1.94e+11 11.3         | 1.67e+10        | 1.0%     |     |
| ► G4PhotoNuclearCrossSection::GetIsoCrossSection(G4DynamicParticle              | 1.74e+10 1.03         | 1.64e+10        | 1.0%     |     |
| ►G4BGGNucleonInelasticXS::CoulombFactor(double, int)                            | 1.55e+10 0.99         | 1.53e+10        | 0.9%     |     |
| ►G4Transportation::PostStepDolt(G4Track const&, G4Step const&)                  | 1.57e+11 9.15         | 1.43e+10        | 0.8%     |     |
| ▶ sincos  | 1.42e+10 0.8          | 1.42e+10        | 0.8%     |     |
| G4VEmProcess::PostStepGetPhysicalInteractionLength(G4Track const                | 4.35e+10 2.5          | 1.34e+10        | 0.8%     |     |
| ►G4CrossSectionDataStore::GetIsoCrossSection(G4DynamicParticle con              | 1.04e+11 6.0          | 1.34e+10        | 0.8%     | Ă   |
| G4SteppingManager::InvokeAlongStepDoltProcs()                                   | 1.47e+11 8.5          | 1.31e+10        | 0.8%     | v   |
| CONTRACT AND A THE PARTY OF AND A THE CONTRACT                                  |                       |                 |          |     |

HPCToolkit screenshot showing the most expensive procedures in cmsExpMT (GEANT4 10.0.beta, GCC 4.6.3). Note that the IEEE transcendentals are called from many sites each. he other routines have few llers.

**Overall Analysis** 

•The general result is that when compiled correctly, cmsExpMT with geant4\_mt\_proto.9.5.p01 has no significant computational hot spots and cache usage is

- efficient.
- "Hot spot" is being used in the sense of "a small section of code that makes a large contribution to the cost of execution".

•Due to its object-oriented design, GEANT4's costs are diffused across a broad set of classes and deep call chains. There are hot functional areas, however, such as computing the physical interaction lengths as part of the process of stepping along tracks.



Depth of the "hottest" call chain. 2<sup>nd</sup> column is the inclusive cost summed across all threads. **Choice of compiler** GEANT4 is usually compiled using GCC 8-core Intel Xeon 5462 2.8GHz 16 GB RAM. run\_cmsExp with 10,000 events

gcc 4.7.3 : 1440.99 seconds Intel 13.0.1 : 1272.56 seconds

Alternative choices of compiler can yield significant performance advantages

An autotuning exercise exploring compilation flags may be productive.

Geant4

# **HEP-ASCR R&D Effort Towards Geant4 Re-engineering**

Boyana Norris, Paul Ruth, Robert Fowler, Pedro Diniz, Robert Lucas, Makoto Asai, Andrea Dotti, Richard Mount, Philippe Canal, Daniel Elvira, Soon Yung Jun, James B. Kowalkowski, Marc F. Paterno, Robert M. Roser

High energy physics research as we know it today would not be possible without simulations. The massive production of event samples similar to those expected in the real experiment is an integral part of the process to design, build, operate highly complex accelerators and detectors and analyze the physics results. Thus Geant4-based simulation is currently the largest consumer of LHC compute cycles. In recent years, space and medicine have become significant user domains, with applications ranging from instrument and detector response verification to radiation dose and shielding optimization and analysis of biological effects. To address to such an increasing demand of high statistic simulations, the Geant4 Collaboration is releasing a new version of Geant4 (Geant4 version 10.0) in December 2013, which enables the use of multi-core CPUs and coprocessors in multi-threaded mode.



### **Performance Analysis**



HPCToolkit screenshot illustrating the deep call chains in the integrator.

### **CPU performance analysis of Geant4 and** Geant4MT

- Effects of different compilers and compiler options
- Callpath profiling of a CMS experiment benchmark (execution time, memory performance)

#### Initial conclusions

- Deep call chains in integrator do not allow local optimizations (including compiler optimizations)
- Bad CPU and memory utilization caused by operating on a single particle at a time in functions at the bottom of deep call paths

| 🕻 nwp profiling 🦳 🕻 cmsExp run_eGamma.g  | 4 🕻 *stepper n=1 (adaptive step control of 4th order RK).nwp 🛿 🗖 🗖                                 | 🔲 Properties 🗔 Detail Graphs 🕱 🛛 🗖        |  |  |
|--|--|---|--|--|
| 🖦 📪 🗞 T 🕂 Ə, Ə, 🗶 🔣                      |  | Max: 14.954 ms                            |  |  |
|  | s 0.5 s 1 s  | 8.551 ms - Avg: 6.403 ms<br>Min: 1.189 ms |  |  |
| Process 17949                            |  | Duration                                  |  |  |
| Thread 1197238048                        |  | 22.004 MB - Max: 22.004 MB                |  |  |
| Runtime API                              |  |   |  |  |
| Driver API                               |  | Avg: 7.587 MB                             |  |  |
| Profiling Overhead                       |  | Size                                      |  |  |
| [0] lesia C2070     Context 1 (CUDA)     |  |   |  |  |
| MomCov (HtoD)                            |  | Max: 5.64 GB/s                            |  |  |
|  |  | Avg: 3.9 GB/s                             |  |  |
|  |  | 1 2.51 GB/S - Min: 2.3 GB/s               |  |  |
| T 100.0% [10] rk4 kernel(GPFieldMap*, G) | Track*, unsigned long)   |   |  |  |
| 2  |  | 1   |  |  |
| 🖬 Analysis 🕱 🗔 Details 📮 Console         |  | - 8                                       |  |  |
| Scope                                    | Results  |   |  |  |
| Analyze Entire Application               | Low Compute Utilization [ 1/0 001 ms / 2 882 s = 5 2% ]  |   |  |  |
|  | The multiprocessors of one or more CPUs are mostly idle  | More                                      |  |  |
|  |  | <u></u>                                   |  |  |
| Stages                                   | Low Memcpy/Compute Overlap [ 0 ns / 49.483 ms = 0% ]   |   |  |  |
| 🖻 Reset All 🛛 🖳 Analyze All              | The percentage of time when memcpy is being performed in parallel with compute is low. <u>More</u> |   |  |  |
| Timeline                                 | Low Memcpy Throughput [ 2.36 GB/s avg, for memcpys accounting for 75.9% of all memcpy time ]       |   |  |  |
| Timeine                                  | <sup>a</sup> The memory copies are not fully using the available host to device bandwidth.         | More                                      |  |  |
| Multiprocessor 🥑                         | Low Memcpy Overlap [ 0 ns / 20.488 ms = 0% ]   |   |  |  |
|  | The percentage of time when two memory copies are being performed in parallel is low.              | More                                      |  |  |
| Kernel Memory 🛛 😵                        |  |   |  |  |
| Kernel Instruction                       |  |   |  |  |

Performance profile of the GPU implementation of the 4<sup>th</sup>-order Runge-Kutta electromagnetic field integrator.

#### **GPU performance analysis and tuning of the RK4 integrator**

• Potential for exploiting greater concurrency through multiple streams and better overlap of memory transfers and computation • Work in progress to generate and autotune portions of the kernel implementations

### **Geant4 version 10**

| G4 1   | 0.0.beta<br>now)                                       | G4 10.0 (Dec. 2013)  | >     | G4 10 series<br>(2014~) |
|--|--|--|-------|-------------------------|
| <ul> <li>API</li> <li>Exan</li> <li>mign</li> <li>Furtl</li> <li>First</li> <li>optin</li> </ul> | re-design<br>nple<br>ation<br>her testing<br>mizations | <ul> <li>Production ready</li> <li>Public release</li> </ul> | •     | Further<br>refinements  |
| ,<br>  |  | Preliminar   | Y SCa | alability tes           |

| optimizations |  |
|---------------|--|
|               |  |
|               |  |
|               | Preliminary scalability test of C<br>pre-beta with full CMS detecto<br>core Intel Xeon CPU and one In<br>coprocessor (60 core).<br>From left to right, CPU, CPU in<br>coprocessor, and coprocessor |
| 250           |  |

| 1) | Perfo |
|----|-------|
|    | typic |
| 2) | Reor  |
|    | use o |
| 3) | Prote |

4 version 10-

ometry on 8-

el Xeon Phi

yper-thread,

hyper-thread.

| 4) | Stu | С |
|----|-----|---|
|    |     |   |

## Loops and

### Data Structure Reorganization

Hotspot analysis (HPCToolkit) revealed: • Extensive use of Transcendental

- Functions
- No obvious hot spots
- Irregular threading overhead

### Focus on Application Understanding

• Minor Performance Improvements due to manual tuning (1%) that has been already incorporated in version beta 10.0.

#### **Future Investigations**

• Exploit Materials collision Memoization • Change stacking strategy to exploit coarse- and fine-grain vectorization

#### Memory Hierarchy

•In general, the instruction cache miss rates are found to be reasonable and do not constitute a bottleneck. There are a few sections of the code that exhibit significantly higher rates, but these routines represent a miniscule part of the total time. •Data cache miss rates are, in general, low enough to not constitute a hot spot. •The "Cross Sections" and "Isotope" classes have loops that do table lookups with higher miss rates. In aggregate, these routines make a non-negligible contribution to execution time.





#### • Geant4 HEP applications

- event-level parallelism
- highly sequential
- memory intensive

#### CUDA/OpenCL/OpenAcc





### **HEP-ASCR R&D Effort**

The process of re-engineering Geant4 has to be started, targeting on recently emerging new computing hardware such as many-core coprocessors and GPUs. Using these new architectures efficiently requires to develop to leverage massive parallelization, complex memory hierarchy and deep

vectorization capability.

FNAL, SLAC, UNC, USC and ANL joined forces to launch an R&D effort to investigate the possible evolution of the software infrastructure and numerical algorithms of the Geant4 toolkit to utilize these emerging technologies.



ormance analysis of the current Geant4 toolkit and its al applications,

ganization of loops in the algorithms to make better of vectorization

otyping GPU-based code

lying automated code transformation for GPUs.

### **Prototype Running On GPU**

- <u>GPU-Vector</u>
- track-level parallelism
- vectorized track dispatche
- Dual Warp Issue

| Data Transfer                |  |         |         |      |
|------------------------------|--|---------|---------|------|
| Data Transfer Speed [Gb/sec] | Tesla M2070 (Device) and AMD Opteron(tm) 6138 (Host)<br>4<br>4<br>4<br>4<br>4<br>4<br>4<br>4 |         |         |      |
| e⁻ <                         | <<<32,128>>>   | CPU(ms) | GPU(ms) | Gain |
| Bremsstralung 2099 104       |  | 104     | 20      |      |
| lon                          | Ionization 558 25  |         | 22      |      |
| Mu                           | Itiple Scattering  | 1034    | 185     | 6    |
| Electron Kernel              |  | 751     | 61      | 12   |

- coalesced memory access
- work balance (CPU-GPU)

