

Geant4 Workshop, Hebden Bridge 13-19 September 2007

New Benchmarks for HEP Geant4 CPU performance

J.Apostolakis, G.Cooperman, G.Cosmo,
V.Ivanchenko, I.Mclaren, T.Nikitina, A.Ribon

CERN PH/SFT

Motivation

Geant4 is now a mature software tool, used in production in several high-energy experiments (ATLAS, BaBar, CMS, LHCb, etc.) and other applications (space science, and bio-medical).

It is therefore important to **benchmark and profile** its **CPU performances**, for different applications, in order to **optimise** it.

LHC experiments are already providing interesting feedback on the performance of Geant4, in their very **complex detector geometries** and for several **physics channels** (QCD, top, Higgs, Z' , SUSY, etc.).

In this talk, we focus on the Geant4 activities for monitoring the CPU performances. Some studies were done in the past; now we are doing it more systematically.

Strategy

To monitor and improve the CPU performance of Geant4 we are using two approaches:

- Use a **set of benchmark tests**, each targeted to stress one particular area (e.g. tracking in magnetic field; electromagnetic physics; hadronic physics), to compare the execution times between different versions of Geant4: 5.2.p02, 6.2.p02, 7.1.p01a(baseline), 8.0.p01 -> 9.0.p01. The goal is to understand the source of any significant variation of performance from one version to the next one.
- For the same set of benchmark tests (eventually with reduced statistics), **profile** a given Geant4 version to identify "hot spots" and get hints for possible optimizations.

LHC user studies

Useful performance studies are being made by LHC users, in particular:

- ❑ Ryszard Jurga (CERN OpenLab)
- ❑ Rafi Yaari (CERN visitor)
- ❑ CMS Collaboration (especially Vincenzo Innocente)
- ❑ Fermilab team CMS G4 now (especially Marc Paterno, Marc Fischer and Jim Kowalowski)
- ❑ ATLAS Collaboration (especially Andrea Di Simone and Andrea Dell'Acqua)

CMS and ATLAS are still very much active in these performance studies!

Pure tracking benchmark

Honeycomb calorimeter benchmark

It consists of transporting 10,000 geantinos, along predefined directions, in a honeycomb calorimeter made of two modules, each 26 x 50 tubes

Release	total time	Ratios	
5.2.p02	2.57s	0.84	
6.2.p02	3.05s	1.00	<--- <i>G4Navigator</i> becomes base class
7.0.p01	3.00s	0.98	
7.1.p01a	3.06s	1.00	
8.0.p01	3.07s	1.00	
8.1.p02	3.02s	0.99	
8.2.p01	3.14s	1.03	<--- <i>in G4Navigator</i>
8.3	3.15s	1.03	<i>LocateGlobalPointAndSetup() metod</i>
8.3.p01	3.13s	1.02	becomes virtual
9.0	3.15s	1.03	
9.0.p01	3.14s	1.03	

These changes in *G4Navigator* have been done to accommodate the Tgeo/VMC interface (ALICE requirement)

Tracking in Magnetic Field: only transportation process.

BaBar Tracker

It consists of simulating the BaBar silicon tracker and 40 layers drift chamber, in a 1.5 T constant magnetic field.

Only transportation, no physics. 100 B-Bbar events simulated.

Locally build with static libraries .

With afs version big time variations were measured (5% or more)

Release	sec/event	Ratios	
7.1.p01a	2.05	1.00	
8.0.p01	2.04	1.01	
8.1.p02	2.14	1.04	<--- <i>G4FieldTrack::LoadFromArray</i> not inline
8.2	2.31	1.12	<--- <i>G4Navigator::LocateGlobalPointAndSetup</i>
8.2.p01	2.31	1.12	become virtual
8.3	2.3	1.12	
8.3.p01	2.31	1.12	
9.0	2.26	1.10	<--- <i>G4PropagatorInField</i>
9.0.p01	2.26	1.10	(better initialization of <i>G4FieldTrack</i> array)

The number of steps and calls to fields are almost the same in all cases.

Tracking in Magnetic Field: QGSP_EMV Physics List

BaBar Tracker

Same Geant4 example as in the previous slide, but this time with the QGSP_EMV Physics List. 100 B-Bbar events simulated.

Local build with static libraries.

Release	sec/event	Ratios	
7.1.p01a	3.04	1.00	(QGSP_GN)
8.0.p01	3.78	1.24	} *
8.1.p02	3.85	1.27	
8.2	3.72	1.22	
8.2.p01	3.84	1.26	
8.3	3.91	1.29	
8.3.p01	3.89	1.28	
9.0	3.57	1.17	<--- Code review of <i>Electromagnetic</i>
9.0.p01	3.62	1.19	<i>physics module</i>

- * The variations are due to tuning and adding safety checks to Urban Multiple Scattering model.

Electromagnetic physics

EM-1 : 10 GeV **e-** in matrix 5x5 of PbWO4 crystals (CMS-type);
cut = 0.7 mm, 1000 events.

EM-2 : 10 GeV **e-** in ATLAS barrel type sampling calorimeter;
cut = 0.7 mm, 1000 events.

EM-3 : 10 GeV **e-** in ATLAS barrel type sampling calorimeter;
cut = 0.02 mm, 100 events.

Release	QGSP			QGSP_EMV		
	EM-1	EM-2	EM-3	EM-1	EM-2	EM-3
5.2.p02	1.03	0.99	1.59	All numbers are with CERN afs installation for SLC3 and shared libraries		
6.2.p02	0.89	0.98	0.97			
7.1.p01	1.00	1.00	1.00			
8.0.p01	1.33	2.24	2.26			
8.1.p01	1.37	2.43	2.01	1.06	1.08	1.07
8.2.p01	1.27	2.03	1.73	1.03	1.09	1.06

QGSP in **8.x** is slower than **7.1** by 20-140%

QGSP_EMV in **8.x** is slower than **7.1** by 3-9%

Electromagnetic physics: CPU benchmark SLC4

- Static build on dedicated SLC4 PC, no libraries from afs
- SLC3 to SLC4 migration slightly change ratio between CPU of different tests

	QGSP			QGSP_EMV		
	EM1	EM2	EM3	EM1_EMV	EM2_EMV	EM3_EMV
8.3 SLC4	1.33	2.30	1.84	1.0	1.0	1.0
9.0	1.21	2.05	1.65	0.92	0.93	0.94
9.0ref01	1.17	2.07	1.66	0.91	0.92	0.91

Better CPU performance in 9.0 mainly due to
code review of Electromagnetic physics module

Main physics changes affecting CPU

- ❑ Electromagnetic physics

New model of Multiple Scattering
(not in QGSP_EMV)

- ❑ Hadronic physics

CHIPS capture at rest for negatively charged
hadrons (G4QStoppingPhysics since 8.1)

- ❑ Due to these improvements in physics

more steps and tracks per event are produced

Which slow down the CPU performance

Hadronic physics. Large statistics(1)

π^- 50 GeV on Copper-Scintillator calorimeter (25 layers, Cu (6cm) - Sci (4mm): a simplified version of CMS HCAL); default 0.7 mm production cut, QGSP_EMV, 4000 events
Local installation with static libraries on dedicated computer (SLC4)

Release	^{sec/evt} B=0	B=4T	Ratios		#steps/evt	
7.1.p01a	1.83	2.07	1.00	1.00	99,050	99,190
8.0.p01	2.00	2.20	1.09	1.06	105,290	105,280
8.1.p02	2.12	2.41	1.16	1.16	105,000	105,620
8.2.p01	2.25	2.56	1.23	1.24	107,290	107,500
8.3	2.22	2.50	1.21	1.21	107,000	106,550

e^- 50 GeV

Release	B=0	Ratios	#steps/evt
7.1.p01a	2.994	1.00	172,240
8.0.p01	3.114	1.04	181,380
8.1.p02	3.160	1.06	175,500
8.2.p01	3.042	1.02	175,690
8.3	3.075	1.03	174,680

Hadronic physics. Large statistics(2)

π^- 50 GeV on Copper-Scintillator calorimeter (25 layers, Cu (6cm) - Sci (4mm): a simplified version of CMS HCAL); default 0.7 mm production cut, QGSP_EMV, 4000 events

Run in the **same conditions** as on previous slide but few months later

Release	sec/evt		Ratios		#steps/evt	
	B=0	B=4T				
8.3.p01	2.31	2.62	1.00	1.00	105,440	106,290
9.0.	2.14	2.45	0.93	0.94	106,670	106,240
9.1.p01	2.19	2.50	0.95	0.95	106,300	105,620

Release	sec/evt		Ratios		#steps/evt	
	B=0					
8.3.p01	3.210		1.00		174,640	
9.0.	2.959		0.92		174,270	
9.1.p01	3.029		0.94		174,290	
8.3	3.175		0.99		174,680	
8.3(05.2007)	3.075		0.96		174,680	

What we have learned

- ❑ It's vital to monitor systematically the Geant4 CPU performance
 - ❑ Profiling and code review very helpful for improvements in CPU performance
 - ❑ afs version with shared libraries gives too big fluctuations (5% or even more)
 - ❑ 3-4% difference was found when re-monitoring the same locally installed version after few months
- Can be due to System upgrades, afs, not single user
- ❑ In future, the best would be to use one dedicated machine with local installation of different versions and total control on the system

Observations

- ❑ CPU performance optimization of Geant4 has been and is an important consideration.
- ❑ LHC experiments are providing us with CPU timing (and profiling) information for their real-life applications (complex detector + physics events).
- ❑ Our G4 benchmarks are based on a set of simple setups, dedicated to stress individual components. We are going to extend the coverage of these tests, including a real complex detector geometry (e.g. CMS) imported via GDML.
- ❑ We are planning to monitor systematically the Geant4 CPU performance at each reference tag, as an extension of our acceptance suite.

Conclusions

- ❑ Identified 'jumps' in CPU time of Geant4 8.x versions :

In Pure Tracking : G4Navigator becomes virtual
(ALICE requirement)

In Electromagnetic physics: New Multiple Scattering
(Not in QGSP_EMV)

In Hadronic physics : extra tracks due to
G4QStoppingPhysics

- ❑ Improvements in 9.0 especially due to
CODE REVIEW in Electromagnetic physics

Added materials:

- More CPU benchmarks from Review07
- Profiling Geant4

CPU comparisons of Physics Lists (1/4)

π^- 50 GeV on Copper-Scintillator calorimeter (25 layers, Cu (6cm) - Sci (4mm): a simplified version of CMS HCAL); 500 events.
Geant4 8.2.p01, B=0.

1 km production threshold, and kill neutrons (StackingAction)

Physics Lists	sec/evt	Ratios	#Steps/evt
LHEP	0.08	1.00	2,590
QGSP_EMV	0.36	4.31	2,290
FTFP	0.34	4.15	2,690
QGSP	0.39	4.69	2,700
QGSC	0.43	5.26	2,560
QGSP_BIC	0.78	9.38	2,850
QGSP_BERT	0.48	5.86	3,040
QGSP_BERT_HP	0.52	6.27	3,830

CPU comparisons of Physics Lists (2/4)

π^- 50 GeV on Copper-Scintillator calorimeter (25 layers, Cu (6cm) - Sci (4mm): a simplified version of CMS HCAL); 500 events.
Geant4 8.2.p01, B=0.

1 km production threshold.

Physics Lists	sec/evt	Ratios	#Steps/evt	#neutronSteps/evt
LHEP	0.25	1.00	8,650	2,570
QGSP_EMV	0.51	2.03	10,370	4,410
FTFP	0.54	2.16	11,490	4,470
QGSP	0.52	2.08	11,120	4,280
QGSC	0.66	2.62	11,300	3,160
QGSP_BIC	2.12	8.43	39,330	15,890
QGSP_BERT	2.62	10.43	65,980	32,690
QGSP_BERT_HP	13.70	54.61	104,200	41,500

CPU comparisons of Physics Lists (3/4)

π^- 50 GeV on Copper-Scintillator calorimeter (25 layers, Cu (6cm) - Sci (4mm): a simplified version of CMS HCAL); 500 events.

Geant4 8.2.p01, B=0.

Default production threshold (0.7 mm).

Physics Lists	sec/evt	Ratios	#Steps/evt
LHEP	1.98	1.00	99,220
QGSP_EMV	2.29	1.16	107,780
FTFP	2.47	1.24	112,440
QGSP	2.49	1.26	113,550
QGSC	2.61	1.32	114,680
QGSP_BIC	4.21	2.12	146,340
QGSP_BERT	4.65	2.35	172,690
QGSP_BERT_HP	15.60	7.88	209,650

CPU comparisons of Physics Lists (4/4)

- From the 1st table (1km + killN) one sees the intrinsic CPU time of the hadronic models.
- From the 2nd table (1km) one sees the combined CPU effect of the hadronic models + tracking the created particles, in particular the neutrons.
- From the 3rd table you can see the overall difference between the various Physics Lists, when all the effects are included.
- It appears that the extra time of Cascade models (Bertini and Binary) is due to extra particles produced and, to a lesser degree, to model computation cost.

Full CMS Detector: Timing Performance

Electromagnetic and Hadron calorimeter

2000 single pion events

100 GeV pions generated separately

in the barrel ($|\eta| \approx 0.3$) and the endcap ($|\eta| \approx 2.1$) detectors with in a small ϕ window

Geant Version	Physics List	Barrel	Endcap
4.7.1.p01a	QGSP	8.32 sec/event	7.44 sec/event
4.8.1.p01	QGSP	12.37 sec/event	10.19 sec/event
4.8.1.p01	QGSP_EMV	8.56 sec/event	7.29 sec/event

Full Atlas Detector: Timing Performance

Using msc from G4.7

Range cut 1mm

CPU time per event (kSI2K)	G4.7	G4.8	G4.8 1mm	G4.8 msc71
Susy	896	2020	1690	
$Z \rightarrow ee$	890	1916	1573	850
$Z \rightarrow \mu\mu$	713	1369	1202	642
$Z \rightarrow \tau\tau$	750	1428	1254	744
$H \rightarrow ll ll$	862	1788	1430	884
Jets	686	1442	1365	701

G4.8 with old msc needs about the same time as G4.7
New multi-scattering leads to about x2 time issue

No optimisation yet of range cuts

Profiling tools

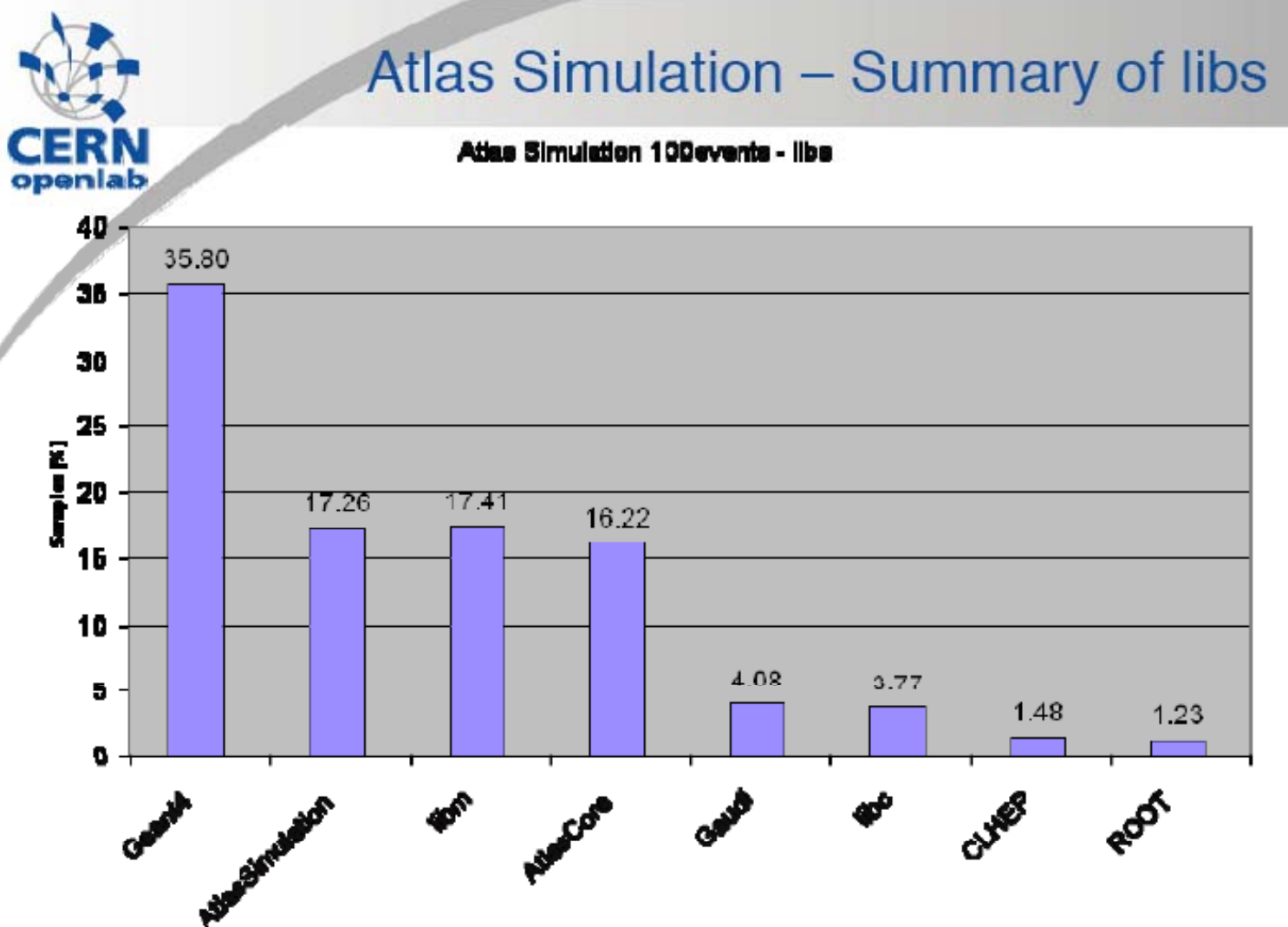
In general, it is a good idea to use different profiling tools, each having its added value.

These are the tools we are using:

- ❑ **gprof** : this is the classic tool; needs static libraries; a bit cumbersome to look at the results...
- ❑ **callgrind** : nice graphical results; information on cache hits and misses; the code runs 50 times slower...
- ❑ **pfmon/perfmon2** : new powerful tools that we start using, with the help of CERN OpenLab (R.Jurga, S.Jarp)

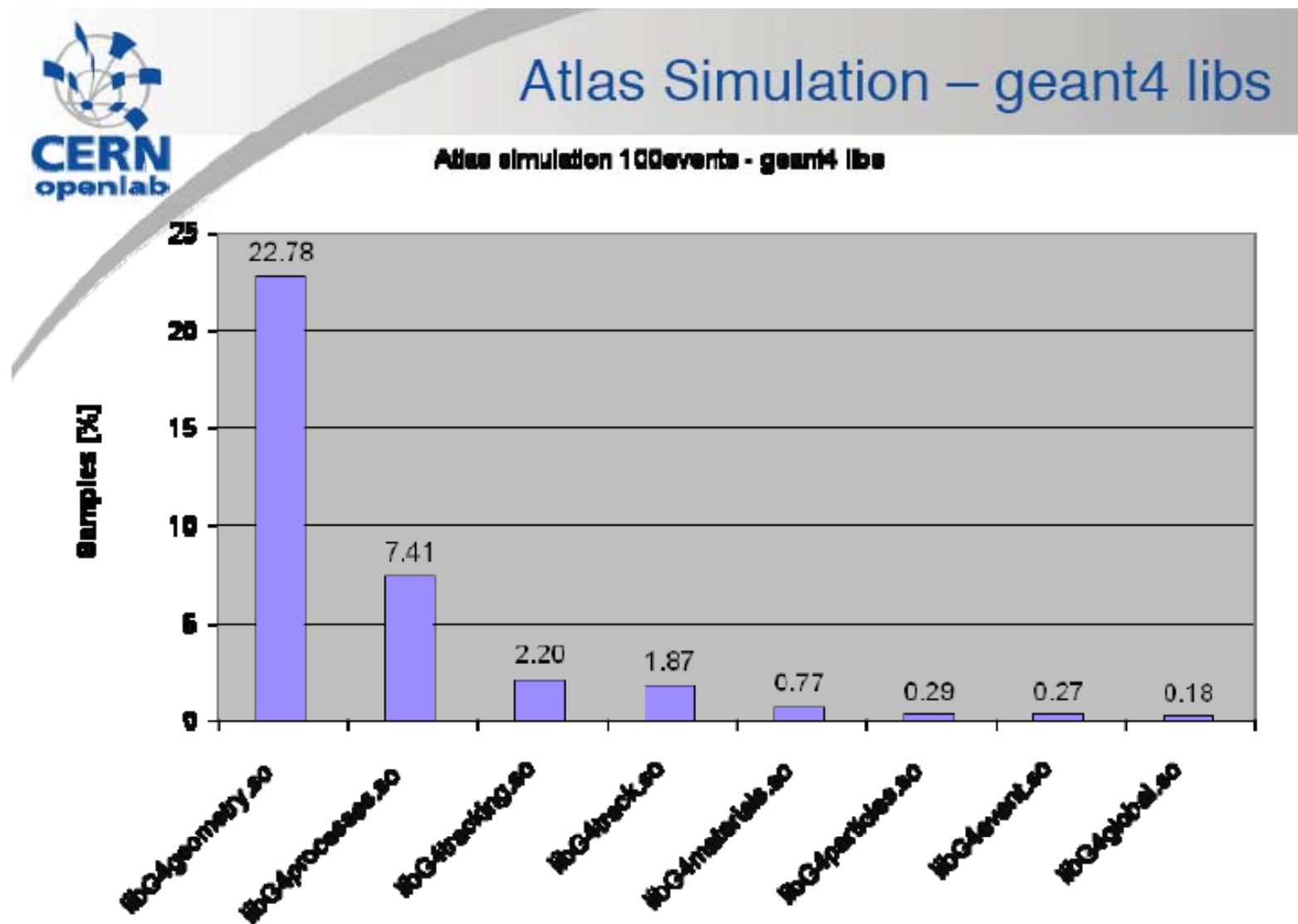
Pfmon (1/3)

Ryszard Jurga, Geant4 Technical Forum Jan 2007



Pfmon (2/3)

Ryszard Jurga, Geant4 Technical Forum Jan 2007



Pfmon (3/3)

Ryszard Jurga, Geant4 Technical Forum Jan 2007



Atlas Simulation@Woodcrest (Core 2 Duo)

```
# counts %self %cum function name:file
Samples: 62400359
3853150 6.17% 6.17% LArWheelCalculator::DistanceToTheNeutralFibre() const:libGeoSpecialShapes.so
3363708 5.39% 11.57% __ieee754_atan2:libm-2.3.4.so
2898193 4.64% 16.21% G4PolyconeSide::DistanceAway():libG4geometry.so
2724866 4.37% 20.58% cos:libm-2.3.4.so
2647220 4.24% 24.82% sin:libm-2.3.4.so
2170627 3.48% 28.30% bfelix_:libG4Field.so
1957847 3.14% 31.44% LArWheelCalculator::parameterized_slant_angle() const:libGeoSpecialShapes.so
1019438 1.63% 33.07% G4PolyconeSide::PointOnCone():libG4geometry.so
1012955 1.62% 34.69% G4PolyconeSide::Inside():libG4geometry.so
871197 1.40% 36.09% G4IntersectingCone::LineHitsCone1():libG4geometry.so
835073 1.34% 37.43% __ieee754_log:libm-2.3.4.so
799517 1.28% 38.71% G4Mag_UsualEqRhs::EvaluateRhsGivenB() const:libG4geometry.so
774924 1.24% 39.95% AtlasField::FieldValue()const:libG4Field.so
719954 1.15% 41.10% CLHEP::Hep3Vector::operator=():libAtlasSealCLHEP.so
653149 1.05% 42.15% gbmag_:libG4Field.so
599985 0.96% 43.11% bprepa_:libG4Field.so
579772 0.93% 44.04% CLHEP::Hep3Vector::y() const:libGeoModelKernel.so
534460 0.86% 44.90% G4PolyconeSide::Intersect():libG4geometry.so
523761 0.84% 45.74% G4ClassicalRK4::DumbStepper():libG4geometry.so
515473 0.83% 46.56% CLHEP::Hep3Vector::x() const:libGeoModelKernel.so
512346 0.82% 47.38% __libc_malloc:libc-2.3.4.so
480003 0.77% 48.15% G4SandiaTable::GetSandiaCofPerAtom():libG4materials.so
473077 0.76% 48.91% free:libc-2.3.4.so
470060 0.75% 49.66% CLHEP::Hep3Vector::z() const:libGeoModelKernel.so
466184 0.75% 50.41% CLHEP::Hep3Vector::Hep3Vector():libGeoModelSvcLib.so
426264 0.68% 51.09% CLHEP::Hep3Vector::operator*=(double):libGeoModelKernel.so
411540 0.66% 51.75% _init:libGeo2G4.so
389988 0.62% 52.38% G4VoxelNavigation::ComputeStep():libG4geometry.so
```

Some profiling results

- ❑ From a first look of the *gprof* profiling for our simplified calorimeters we see that by proper inlining the following methods we can gain $\approx 5\%$:
 - *G4Track::GetVelocity*
 - *G4PhysicsVector::GetValue*
- ❑ But from the full CMS application these methods contribute less than 1% (QGSP_EMV, G4 8.2.p01)

Leaf	Branch	Name
3.1%	3.1%	G4Mag_UsualEqRhs::EvaluateRhsGivenB(...)
2.6%	10.0%	G4ClassicalRK4::DumbStepper(...)
2.3%	6.3%	sim::Field::GetFieldValue(...)
2.2%	3.1%	G4PolyconeSide::DistanceAway(...)
...		malloc , __libc_free , R_Inflate_codes , atan2 , __isnan
1.3%	1.3%	CLHEP::HepJamesRandom::flat()
1.1%	8.5%	G4VoxelNavigation::ComputeStep(...)
1.0%	45.1%	G4SteppingManager::DefinePhysicalStepLength()
1.0%	6.2%	G4Navigator::LocateGlobalPointAndSetup(...)