CPU performance studies in Geant4

J.Apostolakis, G.Cooperman, G.Cosmo,
V.Ivanchenko, 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, 8.0.p01, 8.1.p02, and 8.2.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 optimisations.

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 the Fermilab group, and Vincenzo Innocente)
- ATLAS Collaboration (especially Andrea Di Simone and Andrea Dell'Acqua)

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

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
$H \rightarrow llll$	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

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.

	QGSP			QGSP_EMV		
Release	**EM-1**	**EM-2**	**EM-3**	**EM-1**	**EM-2**	**EM-3**
5.2.p02	1.03	0.99	1.59			
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
8.2.ref02	1.29	2.14	1.79	1.03	1.08	1.06
8.2.ref03	1.28	2.08	1.78	1.04	1.04	1.05

Hadronic physics

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, 500 events.

Release	Physics List	sec/evt B=0	sec/evt B=4T	Ratios	
5.2.p02	QGSP_GN	1.96	2.22	1.01	1.03
6.2.p02	QGSP_GN	1.86	2.06	0.96	0.96
7.1.p01a	QGSP_GN	1.95	2.15	1.00	1.00
8.0.p01	QGSP_EMV	2.12	2.39	1.09	1.11
8.1.p02	QGSP_EMV	2.27	2.48	1.17	1.16
8.2.p01	QGSP_EMV	2.36	2.68	1.21	1.25
8.2.ref02	QGSP_EMV	2.35	2.64	1.20	1.23
8.2.ref03	QGSP_EMV	2.30	2.61	1.18	1.22
8.0.p01	QGSP	2.31	2.67	1.19	1.24
8.1.p02	QGSP	2.54	2.85	1.30	1.33
8.2.p01	QGSP	2.56	2.96	1.31	1.38
8.2.ref02	QGSP	2.53	2.91	1.30	1.36
8.2.ref03	QGSP	2.49	2.83	1.28	1.32

Some observations

From the comparisons between different Geant4 versions we can make some observations:

- QGSP in 8.x is slower than in 7.1 by 20-140%
- QGSP_EMV is 3-25% slower than QGSP 7.1

ATLAS and CMS compared the CPU performance of Geant4 7.1.p01a versus 8.x, for several physics events, with the following findings:

- QGSP in 8.x is 30-120% slower than in 7.1
- QGSP_EMV is as fast as QGSP 7.1, or faster

  It could be due to different production thresholds coupled with geometrical thicknesses...

Profiling tools

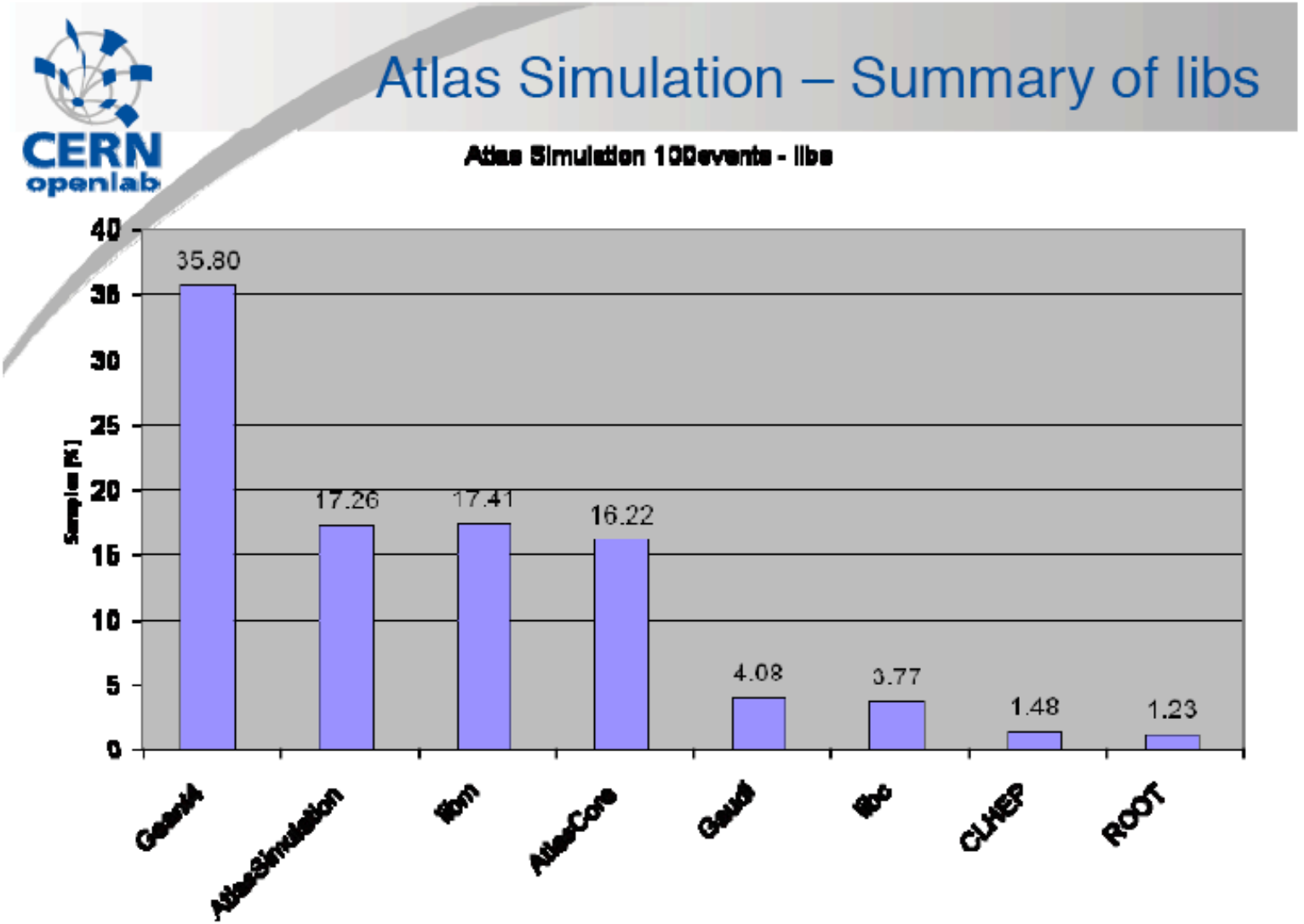
In general, it is a good idea to use different profiling tools, each having its added value.

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)

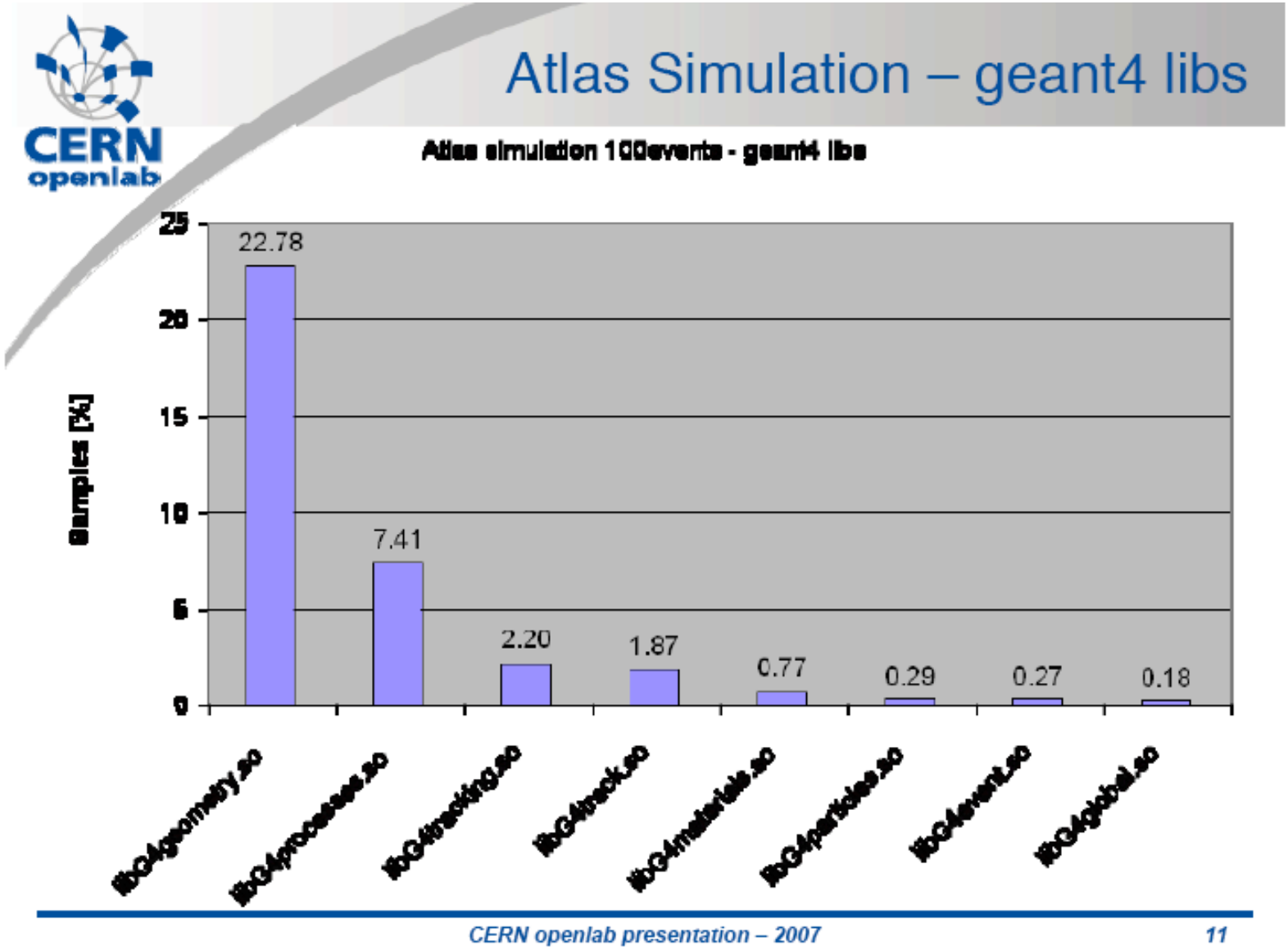
Pfmon (1/3)

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)

CERN openlab

# 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%	gbmagl_: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 ≈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(...)

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.

Conclusions

- CPU performance optimisation of Geant4 is now 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.