Geant4 Computing Performance Task with OpenISpeedshop

Soon Yung Jun, Krzysztof Genser, Philippe Canal (Fermilab) 21st Geant4 Collaboration Meeting, Ferrara, Italy Sept. 12 - 16, 2016

Geant4 Computing Performance (Review 1/4)

• Progressive improvements over the last five years



(more information at https://g4cpt.fnal.gov/g4p/prplots/cpu_by_version.html)

Geant4 Computing Performance (Review 2/4)

• Changes in linked objects (libraries): from 9.5 to 10.2



Library	9.5	10.2	9.5/10.2
libG4processes	971348	577240	1.68
libG4geometry	619304	509816	1.21
libm.so	356072	125249	2.90
libG4tracking	148078	140128	1.06

Also many changes in hot functions from 9.5 to 10.2

Geant4 Computing Performance (Review 3/4)

• The current hot spot: G4PhysicsVector::Value (~6%)

eno current project> - Intel VTune Amplifier (on mic.fnal.gov)	
🚹 🖄 🖙 🕨 🗃 🜗 😅 🕜 Welcome 🛛 hotspot 🗙	=
Basic Hotspots Hotspots by CPU Usage viewpoint (<u>change</u>) ③	Intel VTune Amplifier XE 2016
💿 🖽 Collection Log \varTheta Analysis Target 🛕 Analysis Type 🕅 Summary 🔗 Bottom-up 🚱 Caller/Callee 😽 Top-down T	Tree 📴 Platform
Grouping: Function / Call Stack) i. q x
	CPU Time 🗸 🔷
Function / Call Stack	Effective Time by Utilization
	Idle Poor Ok Ideal Over
▼G4PhysicsVector::Value	3105.414s
\land G4NeutronCaptureXS::GetElementCrossSection \leftarrow G4CrossSectionDataStore::GetCrossSection \leftarrow G4CrossSectionData	776.165s
\Bbbarrow G4VEmProcess::PostStepGetPhysicalInteractionLength \leftarrow G4SteppingManager::DefinePhysicalStepLength \leftarrow G4StepLength \leftarrow G4StepPingManager::DefinePhysicalStepLength \leftarrow G4StepLength \leftarrow	601.601s 700 /
\land G4NeutronElasticXS::GetElementCrossSection \leftarrow G4CrossSectionDataStore::GetCrossSection \leftarrow G4CrossSectionDataS	585.379s
\ltimes G4NeutronInelasticXS::GetElementCrossSection \leftarrow G4CrossSectionDataStore::GetCrossSection \leftarrow G4CrossSectionDataStore::GetCrossSectionDataStore::GetCrossSection \leftarrow G4CrossSectionDataStore::GetCrossSectionDataStore::GetCrossSection \leftarrow G4CrossSectionDataStore::GetCrossSectionDataStore::GetCrossSection \leftarrow G4CrossSectionDataStore::GetCrossSectionDataStore::GetCrossSection \leftarrow G4CrossSectionDataStore::GetCrossSectionDataStore:::GetCrossSectionDataStore::GetCrossSectionDataStore:::GetCrossSectionDataStore:::GetCrossSectionDataStore:::GetCrossSectionDataStore:::GetCrossSectionDataStore:::GetCrossSectionDataStore::::GetCrossSectionDataStore::::GetCrossSectionDataStore::::GetCrossSectionDataStore::::GetCro	314.8105
\ltimes G4VEnergyLossProcess::PostStepGetPhysicalInteractionLength \leftarrow G4SteppingManager::DefinePhysicalStepLength \leftarrow G4StepLength \leftarrow	234.9825
\ltimes G4VEnergyLossProcess::AlongStepGetPhysicalInteractionLength \leftarrow G4SteppingManager::DefinePhysicalStepLength \leftarrow G4StepLength \leftarrow G4StepPingManager::DefinePhysicalStepLength \leftarrow G4StepLength \leftarrow G4StepLength \leftarrow G4StepLength \leftarrow G4StepLength \leftarrow G4StepLength \leftarrow G4S	144.718s
\land G4VEnergyLossProcess::AlongStepDoIt \leftarrow G4SteppingManager::InvokeAlongStepDoItProcs \leftarrow G4SteppingManager::Step	114.2985 Ton / Caller
\ltimes G4UrbanMscModel::ComputeTruePathLengthLimit \leftarrow G4VMultipleScattering::AlongStepGetPhysicalInteractionLength \leftarrow	88.475s IOP - Cances
\Bbbarrow G4UrbanMscModel::SampleScattering \leftarrow G4VMultipleScattering::AlongStepDoIt \leftarrow G4SteppingManager::InvokeAlongStep	56.554s
▷ \circ G4UPiNuclearCrossSection::In	
G4UrbanMscModel::Compute(G4NeutronCaptureXS::GetFlerr	nentCrossSection
K G4WentzelVIModel::Compute	
© G4UrbanMscModel::SampleCo	
G4VEnergyLossProcess::Post! G4VEmProcess::PostStepGetP	nysicalinteraction
尽 G4SeltzerBergerModel::Samp	,
© G4LossTableBuilder::BuildRang CANoutron Flactic VCCot Flame	ntCrossCostion (
▷ G4ComponentSAIDTotalXS::G/ 04NeulronEldSLICAS::0elElenne	$nucrosssection \leftarrow u$
尽 G4NeutronInelasticXS::IsoCro	
G4ComponentSAIDTotalXS::G G4 NeutronInelasticXSGetFlen	nentCrossSection
♦ G4EmCorrections::BarkasCorr	lenceross section a
\Bbbarrow G4WentzelVIModel::ComputeGeomPathLength \leftarrow G4WentzelVIModel::ComputeTruePathLengthLimit \leftarrow G4VMultipleSca	1.300s
${\tt \ \ } G4KokoulinMuonNuclearXS:: GetElementCrossSection \leftarrow G4CrossSectionDataStore:: GetCrossSection \leftarrow G4CrossSection \leftarrow G4Cros$	0.530s
▷ \sigma G4EmCorrections::ShellCorrection	0.160s
Selected 1 row(s):	2663.291s
EUTER 🕥 100.0% 🔍 🛛 Any Processi a) Thread Any Thread	er functions
Any House (Any House) (Any House) (Any House) (Any House)	

Geant4 Computing Performance (Review 4/4)

10.2 cycle: CPU/mem under control while improving physics



Geant Multi-threading: good scalability and memory reduction



5

Current Profiling Tools and Primary Metrics

Tools

- FAST (CPU Profiling)
- IgProf (Memory profiling)
- OpenISpeepshop (Geant4 MT profiling)
- Metrics (Observables)
 - Timing
 - Event
 - Function (exclusive and inclusive)
 - Library
 - Memory
 - Footprint (live, total)
 - statm (top)
 - Multi-threaded applications
 - Event throughput and memory reduction

Challenges and New Requirements

- Flat profile
 - No obviously reducible hot spots or bottlenecks
 - Need more specific metrics
- Measurement uncertainty
 - Fixed rate for sampling and call path collections (100Hz)
 - Different optimization paths when recompiling Geant4 libs ?
 - Rerun reference for each release (4 nodes x 32 jobs)
- Limited extensibility
 - Not all tools are capable for profiling multi-threaded applications
 - Not capable for profiling codes on new hardware architectures
- Identify the better profiling tool(s) for G4CPT
 - Pick a right (integrated) tool to meet all new requirements
 - Surveyed several tools and selected OpenISpeedshop

Introduction to OpenISpeedshop

- Open source (the Krell institute, <u>https://openspeedshop.org/</u>)
- Sampling Experiments (Light weighted)
- Support for Call Stack Analysis
- Hardware Performance Counters
- Memory Function Tracing
- MPI Profiling and Tracing
- I/O Profiling and Tracing
- Floating Point Exception Analysis
- POSIX Thread Function Tracing (Multithreading-capable)
- Flexible analysis options (GUI, command line, online)
- Support Intel MIC architecture and NVIDIA CUDA
- Well documented and supported

Sampling Experiments in OSS

- pcsamp (periodic sampling the program counters)
 - Low overhead overview of time distribution
- usertime (call path profiling)
 - Inclusive and exclusive timing data
 - Call paths, caller and callee relationships
- hwcsamp (periodic sampling hardware counters)
 - Profile of hardware counter events (PAPI events)
- mem (memory tracing)
 - Call paths for memory related function call events
 - Aggregate and individual rank, thread, or processing timings
- pthreads (POSIX thread tracing)
- io (I/O tracing)
- Many other useful experiments

Measurement Overheads and Output Size

 pcsamp: exclusive time - insensitive to sampling frequency (default 100Hz, set to 100000 Hz for G4CPT)

Freque	ency	Time(sec)	OverHead(%)	DB size(MB)
bas	se :	52.20	-	
50	Hz:	52.27	0.13	0.376832
100	Hz:	52.62	0.80	0.486400
200	Hz	52.36	0.31	0.607232
500	Hz	52.98	1.49	0.811008
1000	Hz:	52.65	0.86	0.971776
10000	Hz:	52.76	1.07	1.012736

 usertime: inclusive time and call paths – large overhead (default 35, set to 100Hz): similar overhead for hwcsamp

Frequency	Time(sec)	OverHead(%)	DB size(MB)
base :	52.80	-	
35 Hz:	53.89	2.06	1.087488
50 Hz:	54.33	2.90	1.430528
100 Hz:	56.21	6.46	2.355200
200 Hz	60.25	14.11	4.208640
1000 Hz:	92.84	75.83	18.725888

Migration Plan and Status:

- Replace SimpleProfiler (FAST) with OpenISpeedshop (OSS)
- All information provided by old tools are reproduced with OSS and results have been posted since 10.2.r06

(i) ▲ https://g4cpt.fnal.gov								
🛅 Most Visited ▼	芦 Fermilab 🔻	📁 Linux Distros 🔻	Geant4 LXR	🔆 CMSSW LXR	Fermilab Redmine	🕞 Geant4	📮 Cern Mai	1 📴 F
Geant4 Profiling and Benchmarking								
<u>Geant4 CPU P</u>	erformance	by Version						
1) The <u>Cur</u>	1) The <u>Current</u> profiling activity is a part of <u>Geant4 Computing Performance Task</u>							
2) Profiling	g Results							
Migration to	Open She	epshop (will be	the default	profiler fro	m 10.3)			
Profiled on the Wilson cluster using AMD 6128HE Opteron 2GHz and Intel Xeon X5650 (2.67GHz)								
Geant4 Ve	rsion	Application		Performan	ce	Summa	ary	
10.2.r07 (O	SS)	SimplifiedCalo	Simple Pro	ofiler <u>Me</u> i	mory Profiler	<u>CPU</u>	MEM	
10.2.r06 (O	SS)	SimplifiedCalo	Simple Pro	ofiler <u>Me</u> i	mory Profiler	CPU I	MEM	
*: recompiled & rerun at the date								

Profiled on the Wilson cluster using AMD 6128HE Opteron 2GHz & gcc 4.9.2

Geant4 Version	Application	Performance Summa			mary
10.2.r07	SimplifiedCalo	Simple Profiler	Memory Profiler	<u>CPU</u>	MEM
10.2.r06 (rerun, 08/19)	SimplifiedCalo	Simple Profiler	Memory Profiler	<u>CPU</u>	MEM
10.2.r06 (10.3-BETA)	SimplifiedCalo	Simple Profiler	Memory Profiler	<u>CPU</u>	MEM
10.2.r05	SimplifiedCalo	Simple Profiler	Memory Profiler	<u>CPU</u>	MEM
10.2.r04 (May 28)	SimplifiedCalo	Simple Profiler	Memory Profiler	<u>CPU</u>	MEM
10.2.r04	SimplifiedCalo	Simple Profiler	Memory Profiler	<u>CPU</u>	MEM

Results: Average CPU Time

 Triple-cross-checks (2 sets x 128 AMD nodes, 1set x 48 Intel nodes) - example: H →ZZ (SimplifiedCalo)

 $prof_{100}_{events_{plot.png:}}$







OSS pcsamp (program counter sampling) Experiment

• OSS GUI example: Function view \rightarrow g4cpt web plot/table

<u>F</u> ile <u>T</u> ools						g4.10.2.r06 SimplifiedCalo		_
					AMD	OpteronProcessor61	28HE	
💌 pc Sampling [1]		G4PhysicsVector::Value					• • •	
-Process Control		G4Mag_UsualEqRhs::EvaluateRhsGivenB CLHEP::MTwistEngine::tlat			00000 •			
Run 🌔 C	ic.	G4SteppingManager::DetinePhysicalStepLength G4PropagatorInField::ComputeStep						
		G4SteppingManager::Stepping	-	ю				
Status: Drocess Load	G4	4Transportation::AlongStepGetPhysicalInteractionLength G4UniversalFluctuation::SampleFluctuations	••• •	0				
Status. Process Load	1	G4ParticleChange::CheckIt G4C basicalP K4::DurabStarms	:					
V Stats Panel [1]		G4VEmProcess::PostStepGetPhysicalInteractionLength						
	Í.	G4Navigator::Compute Step G4SteppingManager::Invoke PSDIP						
11 U CL 🗈 🛛	G4VEne	rgyLossProcess::PostStepGetPhysicalInteractionLength sincos	-					
Executables: Simplif	Name	short			mname		max.leaf.frac.r	nedian
Laccaubies: binipin								
	9	G4PhysicsVector::Value			AMDOptero	nProcessor6128HE	0.00	6152750
	4	G4Mag_UsualEqRhs::EvaluateRhsGivenB			AMDOptero	nProcessor6128HE	0.03	3266805
	1	CLHEP::MTwistEngine::flat			AMDOptero	nProcessor6128HE	0.02	2831920
	11	G4SteppingManager::DefinePhysicalStepLength AMDOpte		AMDOptero	nProcessor6128HE	0.02	2491585	
	10	G4PropagatorInField::ComputeStep			AMDOptero	nProcessor6128HE	0.02	2167700
	13	G4SteppingManager::Stepping AMDOpteronProcessor6128		nProcessor6128HE	0.02	2000040		
	15	G4Transportation::AlongStepGetPhysicalInteractionLength AMDOpteronProc		nProcessor6128HE	0.0	1962175		
	16	G4UniversalFluctuation::SampleFluctuations AMDOpteronProcessor612		nProcessor6128HE	0.0	1911510		
	8	G4ParticleChange::CheckIt			AMDOptero	nProcessor6128HE	0.0	1836950
Command Panel	3	G4ClassicalRK4::DumbStepper			AMDOptero	nProcessor6128HE	0.0	1812575

Other views: linked object (libraries), function paths and so on

Comparison between FAST and OSS: Top Ten Functions

• Exclusive time fraction (10.2.r06, $H \rightarrow ZZ$, SimplifiedCalo)

FAST (100 Hz)

OSS (10 KHz)

short	.frac.median	frac.median	short
G4PhysicsVector::Value	0.06152750	0.061318050	G4PhysicsVector::Value
G4Mag_UsualEqRhs::EvaluateRhsGivenB	0.03266805	0.032771450	G4Mag_UsualEqRhs::EvaluateRhsGivenB
CLHEP::MTwistEngine::flat	0.02831920	0.027021450	CLHEP::MTwistEngine::flat
G4SteppingManager::DefinePhysicalStepLengt	0.02491585	0.025940300	G4SteppingManager::DefinePhysicalStepLength
G4PropagatorInField::ComputeStep	0.02167700	0.024555300	G4PropagatorInField::ComputeStep
G4SteppingManager::Stepping	0.02000040	0.021670850	G4Transportation::AlongStepGetPhysicalInteract
G4Transportation::AlongStepGetPhysicalIntera	c 0.01962175	0.019675500	G4Navigator::ComputeStep
G4UniversalFluctuation::SampleFluctuations	0.01911510	0.018696050	G4UniversalFluctuation::SampleFluctuations
G4ParticleChange::CheckIt	0.01836950	0.018662900	G4ClassicalRK4::DumbStepper
G4ClassicalRK4::DumbStepper	0.01812575	0.018078150	G4SteppingManager::Stepping

 Also consistent profiling results for exclusive timing (call paths) and linked objects

OpenISpeedshop: Experiments with Hardware Counters

- Periodic sampling hardware counters (hwcsamp)
- Supports both derived and non-derived PAPI presets
- A list of some possible hardware counter combinations

For Xeon processors:	
PAPI_FP_INS, PAPI_LD_INS, PAPI_SR_INS	Load store info, memory bandwidth needs
PAPI_L1_DCM, PAPI_L1_TCA	L1 cache hit/miss ratios
PAPI_L2_DCM, PAPI_L2_TCA	L2 cache hit/miss ratios
LAST_LEVEL_CACHE_MISSES,	L3 cache info
LAST_LEVEL_CACHE_REFERENCES	
MEM_UNCORE_RETIRED:REMOTE_DRAM,	Local/nonlocal memory access
MEM_UNCORE_RETIRED:LOCAL_DRAM	
For Opteron processors:	
PAPI_FAD_INS, PAPI_FML_INS	Floating point add multiply
PAPI_FDV_INS, PAPI_FSQ_INS	Square root and divisions
PAPI_FP_OPS, PAPI_VEC_INS	Floating point and vector instructions
READ_REQUEST_TO_L3_CACHE:ALL_CORES,	L3 cache
L3_CACHE_MISSES:ALL_CORES	

OSS hwcsamp (hardware counters) Experiment

 hwcsamp example: TOT_CYS, TOT_INC, FP_OPS, LD_INS (10.2.r06, H→ZZ, SimplifiedCalo)

	Open Spe	edShop			
<u>F</u> ile <u>T</u> ools					<u>H</u> elp
HWCSamp Panel [1] Process Control					
Run 🕨 Cont 🏓 Pause 5	Update				Terminate
Status: Process Loaded: Click on the "Run" but	ton to begin the exper	iment.			
▼ Stats Panel [1] ▼ ManageProcessesPar	el [1]				⋐ 🛛 🗠 ×
T C D S O S C Showin	g Functions Report:		View/Display	y Choice ———— s 👽 Statements	✓ Linked Objects
Executables: SimplifiedCalo Host: tev0213 Pic	s: 1 Threads: 1				
Exclusive CPU tim % of C	PU I papi_tot_cyc	papi_tot_ins	papi_fp_ops	papi_ld_ins	Function (definir
-713.510000 6.1758	91 1892759526833	1399990082267	237726850429	535181336480	G4PhysicsVector:
-392.690000 3.3989	86 1041789723689	771310626385	131433882645	298371573078	G4Mag_UsualEq1
-261.910000 2.2670	01 694781090619	513525568386	86968803745	194786334389	G4SteppingMana
-240.310000 2.0800	39 637528252419	471776525267	80373538044	182490443412	G4PropagatorInF
-215.940000 1.8691	01 572810451364	423747412392	71659104631	160213157867	CLHEP::MTwistE
-192.720000 1.6681	16 511273859112	378760623923	64785290428	146873535016	G4UniversalFluct
-189.660000 1.6416	30 503111037709	371886581901	62798162971	140104450036	G4SteppingMana
-170.970000 1.4798	56 453574126404	335574209764	57211872732	129991487682	G4VEmProcess::P
-169.150000 1.4641	03 448694858464	331741971326	56054570509	125331873452	G4Navigator::Coi
-167.910000 1.4533	70 445373769045	328823366321	55290084112	122476615702	ieee754_log (li
□ □ 167.180000 1.4470	51 443519111592	328299790621	55971167622	127018300123	G4ClassicalRK4:: M

Code Performance by Hardware Counter Metrics

• Derivatives: examples

Hardware Counter Metrics Derivatives	Performance
IPC (Instruction/Cycle)	Large values suggest good balance with minimal stalls.
FPC (FLOPS/Cycle)	Large values for floating point intensive codes suggests efficient CPU utilization
FMO (FLOPS/Memory Ops)	Good data locality, Computational Intensity
LPC (Loads/Cycle)	Useful for calculating FMO, may indicate good stride through arrays.
SPC (Stores/Cycle)	Useful for calculating FMO, may indicate good stride through arrays.

- Example: Higgs \rightarrow ZZ (10.2.r06) with SimplifiedCalo
 - IPC = 0.74 (relatively small)
 - FMO = 0.30

More OSS Post Analysis

- Profiling information at the function statement (line) level

 Will provide the top-100 list from 10.3
- Multi-threading capability (see the next talk)
- Compare two experiments (osscompare): examples to use
 - two releases
 - Two experiments with the different numbers of threads
- Call path analysis based on DB

Work Plan

- Migration to OpenIspeedshop (partially done)
 - Have been tested with recent releases since 10.3.beta
 - Reproduced all performance data that we used to measure
 - More information are ready to be added
- Current set of experiments
 - pcsamp (exclusive time)
 - usersamp (inclusive time and call path)
 - hwcsamp (PAPI hardware counters)
- Extension
 - mem (memory tracing)
 - pthreads (POSIX thread tracing)
 - io (I/O tracking)

Summary

- Reviewed Geant4 computing performance for recent releases
- Challenges and new requirement in G4CPT have been addressed
- OpenIspeedshop is selected as the default profiler for the future computing performance task
- Migration is done and computing performance profiling and benchmark are ready to be extended

Backup Slides

Geant4 Computing Performance: Hot Functions

Changes in hot functions from 9.5 to 10.2







Results: OSS linked Objects

• OSS GUI example: Linked Objects view \rightarrow g4cpt web plot

