

# Profiling LArTest with OpenSpeedshop

Soon Yung Jun and Hans Wenzel (PDS)

LArSoft Coordination Meeting

April 25, 2017

## Application: LArTest

- A standalone Geant4 application (developed by H. Wenzel)
  - Cubic (5m $\times$ 5m $\times$ 5m) LAr fiducial volume
  - GDML to assign step limits and sensitive detector to volumes
  - Optical (scintillation) photons produced in sensitive detector
- Computing performance monitoring features
  - Event time
  - Memory (lgProf, statm)
  - Statistics for tracks/steps by the particle type
- Goal for profiling LArTests
  - Monitor Geant4 part of computing performance changes with LAr detectors (energy, particle, physics list)
  - Integrate into the Geant4 computing performance task (<https://g4cpt.fnal.gov>)

## Profiler: OpenSpeedshop (OSS)

- Comprehensive performance analysis for sequential, multithreaded, and MPI applications
- Open source (the Krell institute, <https://openspeedshop.org>) and one of ASCR profiling tools (TAU, HPCToolkit, Jumpshot, ...)
- The base functionality includes
  - Sampling experiment (light-weighted)
  - Support call stack analysis
  - Hardware performance (PAPI) counters
  - Multi-threaded, MPI profiling and tracing
  - Memory function tracing, I/O profiling and tracing, and etc.
- Tested on a variety of Linux clusters and supports parallel hardware architectures (Intel MIC, NVIDIA CUDA) as well as HPC systems (Cray, Blue Gene)

## OSS: Installation and Performance Measurement

- Installation: a typical build (with the version 2.2)
  - ./install-tool --build-krell-root
    - krell-root-prefix \${install\_dir}/krellroot\_v2.2
    - with-openmpi /usr/local/openmpi-1.8.1
  - ./install-tool --build-offline
    - openss-prefix \${install\_dir}/openspeedshop2.2
    - krell-root-prefix \${install\_dir}/krellroot\_v2.2
    - with-openmpi /usr/local/openmpi-1.8.1
- Running an experiment: unmodified binary instrumentation
  - osspcsamp "lArTest lArBox.gdml profile.pi-5GeV" [frequency]
- Performance analysis: GUI example with the output db file
  - openss -f lArTest-pcsamp.openss

# OSS (GUI): Default View and Statistical Panel

The screenshot displays the OpenSpeedShop GUI with several key components:

- Process Control:** Includes buttons for Run, Cont, Pause, Update, and Terminate. A tooltip for the Update button reads: "Update the display with the current information."
- Status:** Shows "Process Loaded: Click on the 'Run' button to begin the experiment."
- Stats Panel:** Contains a toolbar with various icons (I, U, CL, D, S, E, G, H, C, B, TS, OV, SA, LB, CA, CC) and a "View/Display Choice" dropdown menu with options for Functions, Statements, and Linked Objects.
- Executables:** Displays "/home/syjun/g4p/test/openss/cmsExpMT/bin/cmsExpMT Host: cluck.fnal.gov Pids: 1 Threads: 33".
- Functions Report:** A table showing CPU time statistics for various functions. The top function is `tls_get_addr (/lib64/ld-2.12.so)` with 11.216171% of total exclusive CPU time.
- Command Panel:** Shows the prompt `openss>>`.

Red annotations highlight the "Toolbars" and "Top Functions" areas.

% of Total Exclusive CPU Time	Exclusive CPU time	Inclusive CPU time	% of Total Exclusive CPU Time	Function (defining location)
11.216171	105.742855	105.742855	11.216171	<code>tls_get_addr (/lib64/ld-2.12.so)</code>
10.128193	95.485712	95.485712	10.128193	<code>__ieee754_log (/lib64/libm-2.12.so)</code>
6.012668	56.685713	56.685713	6.012668	<code>__ieee754_exp (/lib64/libm-2.12.so)</code>
4.748917	44.771428	62.885713	4.748917	<code>G4HadronCrossSections::CalcScatteringCrossSections</code>
3.279086	30.914285	156.999997	3.279086	<code>G4CrossSectionDataStore::GetCrossSection (/home/s</code>
	20.400000	64.571427	2.163833	<code>G4ElasticHadrNucleusHE::HadrNucDifferCrSec (/hor</code>
	16.028571	47.514285	1.700155	<code>G4Navigator::LocateGlobalPointAndSetup (/home/sy</code>
	15.400000	15.400000	1.633482	<code>cmsExpMagneticField::GetFieldValue (/home/syjun/</code>
	12.571428	62.085713	1.333455	<code>G4hPairProductionModel::ComputeDMicroscopicCro</code>
	11.885714	14.371428	1.260721	<code>G4ProductionCutsTable::ScanAndSetCouple (/home/</code>
other				

# 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)
- pthreads (POSIX thread tracing)
- mem (memory tracing)
  - call paths for memory related function call events
  - aggregate and individual rank, thread, or processing timings
- io (I/O tracing)
- Many other useful experiments

## OSS: Measurement Overheads and Output Size

- pcsamp: exclusive time - insensitive to sampling frequency (default 100Hz)

Frequency	Time(sec)	OverHead(%)	DB size(MB)
base :	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): similar overhead for hwcsamp

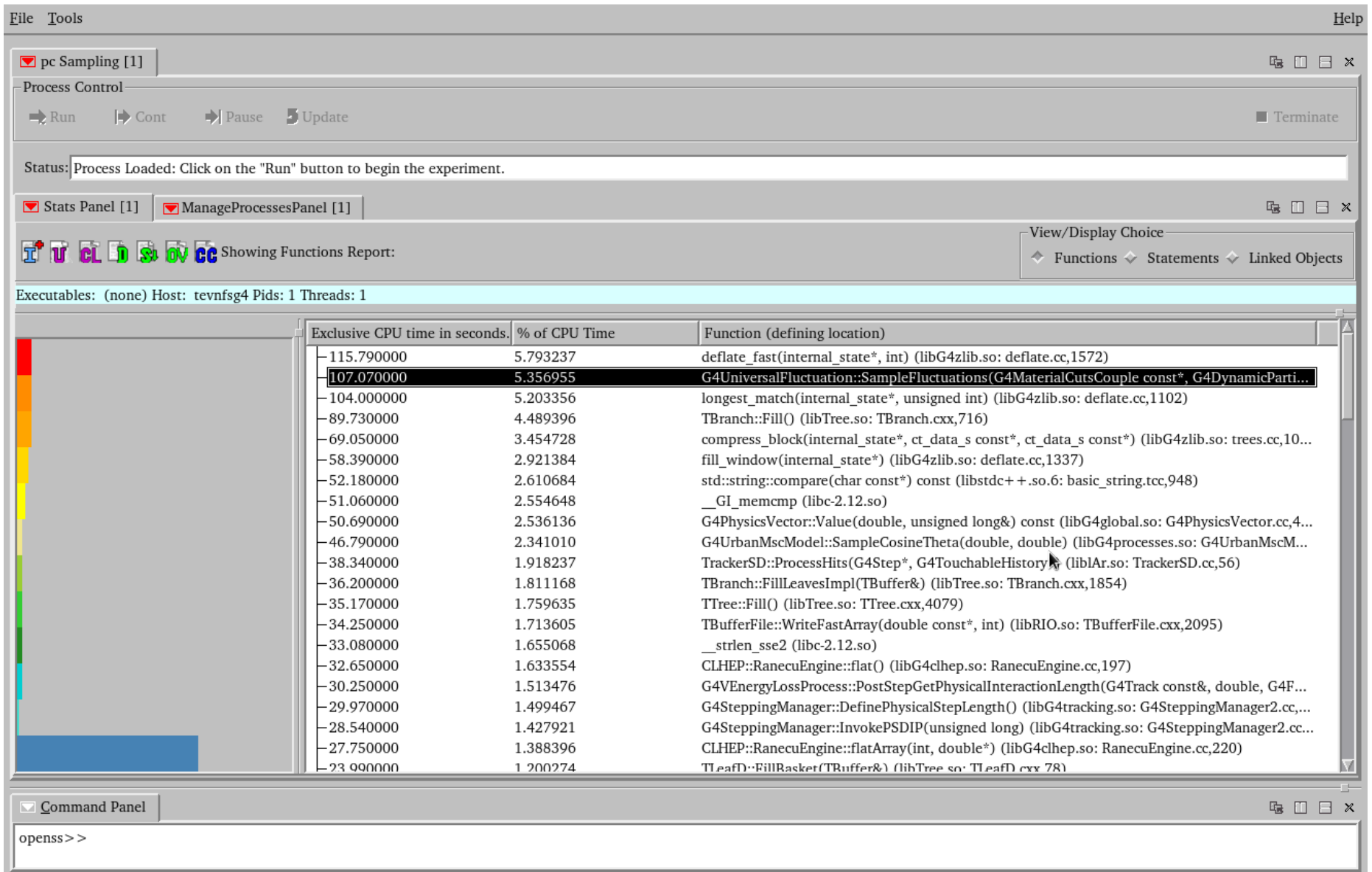
Frequency	Time(sec)	OverHead(%)	DB size(MB)
base :	52.80	-	
35 Hz:	53.89	2.06	1.087488
50 Hz:	54.33	2.90	1.430528
100 Hz:	56.21	6.46	2.355200
200 Hz	60.25	14.11	4.208640
1000 Hz:	92.84	75.83	18.725888

## Preliminary Performance Experiments with LArTest

- LArTest configuration
  - Beam: 5 GeV pi-
  - Step limit: 0.01 cm
  - Physics list: FTFP\_BERT + Standard EM
  - 1000 events
- osspcsamp (100 Hz)
  - I/O (digitization) ON
  - Analysis ON
- ossusertime and osshwcsamp (35 Hz)
  - I/O (digitization) OFF
  - Analysis OFF



# osspcsamp LArTest: Functions (Exclusive CPU Time)



- I/O (zlib/root functions) are leading top functions (> 30%)

# osspcsamp LArTest: Statements (Line Numbers)

- Select statement level granularity
- List line numbers in program that took most of time

File Tools Help

pc Sampling [1]

Process Control

Run Cont Pause Update Terminate

Status: Process Loaded: Click on the "Run" button to begin the experiment.

Stats Panel [1] ManageProcessesPanel [1]

Showing Statements Report... View/Display Choice: Functions Statements Linked Objects

Executables: (none) Host: tevnfsg4 Pids: 1 Threads: 1

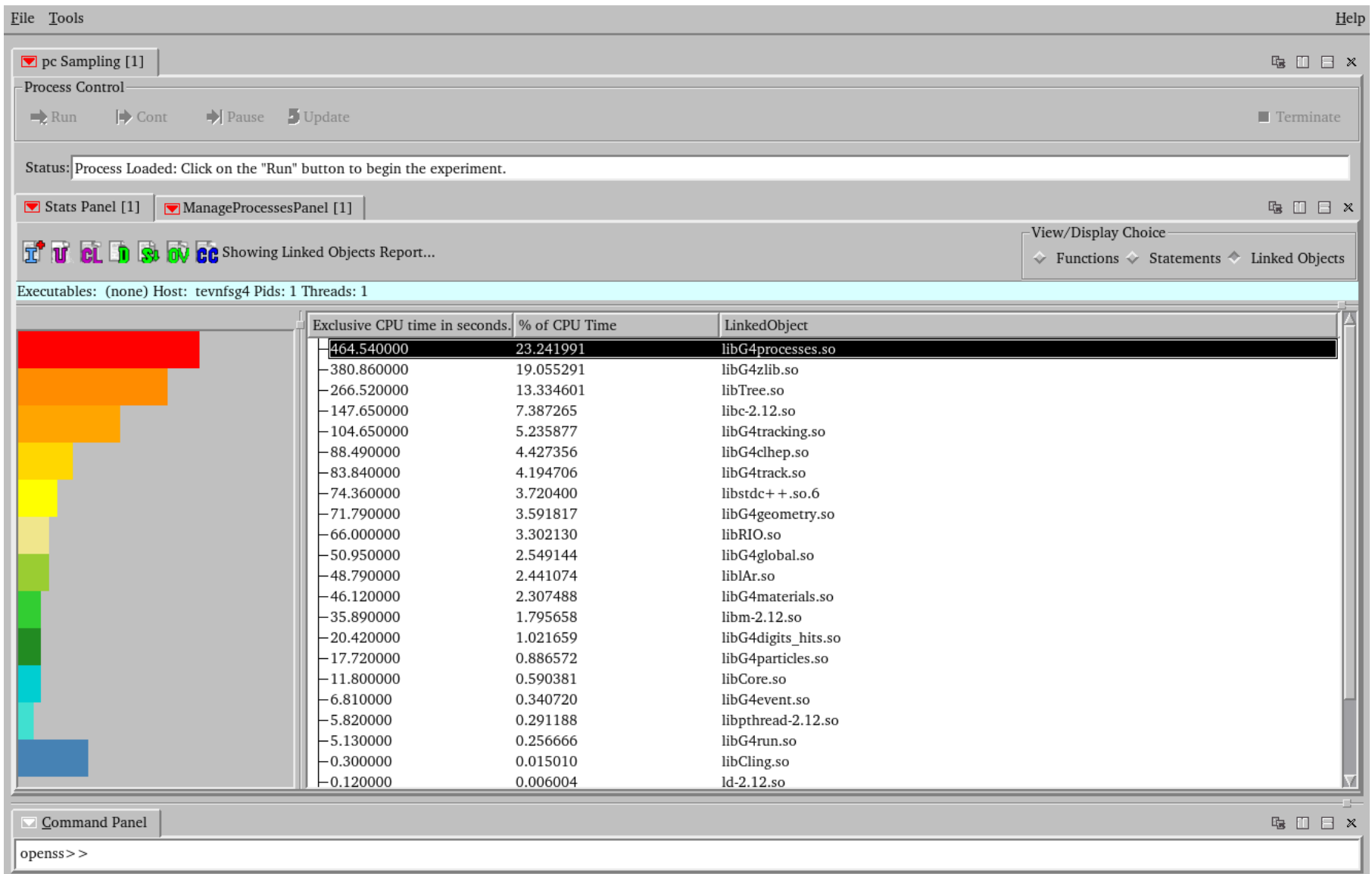
Exclusive CPU time in seconds.	% of CPU Time	Statement Location (Line Number)
31.210000	1.741902	deflate.cc(1601)
28.910000	1.613533	trees.cc(1035)
24.740000	1.380796	TBranch.cxx(742)
23.290000	1.299868	deflate.cc(1595)
19.310000	1.077735	TBufferFile.cxx(2101)
19.080000	1.064899	TBranch.cxx(1859)
18.590000	1.037551	deflate.cc(1236)
17.520000	0.977831	deflate.cc(1649)
17.080000	0.953274	deflate.cc(1195)
16.270000	0.908066	TObjArray.h(91)
15.620000	0.871788	TLeaf.cxx(276)
14.100000	0.786953	char_traits.h(263)
<b>14.010000</b>	<b>0.781930</b>	<b>G4UniversalFluctuation.cc(317)</b>
13.400000	0.747885	deflate.cc(1381)
13.170000	0.735048	basic_string.tcc(956)
13.150000	0.733932	basic_string.tcc(959)
12.830000	0.716072	deflate.cc(1388)
12.530000	0.699328	deflate.cc(1389)
12.330000	0.688166	TrackerSD.cc(87)
11.670000	0.651329	deflate.cc(1380)
10.940000	0.610586	deflate.cc(1240)
10.870000	0.606680	adler32.cc(106)

Command Panel

```
opensp>>
```

# osspcsamp LArTest: Linked Objects

- The library in which the associated function is located (aggregated by shared objects)



# ossusertime LArTest: Call Path (Functions)

- Function calls observed anywhere in the stack
- The inclusive time taken by the function and all its callees

Executables: lArTest Host: tevnfsg4 Pids: 1 Threads: 1

Exclusive CPU time in seconds.	Inclusive CPU time in seconds.	% of Total Exclusive CPU Time	Function (defining location)
107.028569	154.971425	10.149836	G4UniversalFluctuation::SampleFluctuations(G4MaterialCutsCouple
<b>51.542856</b>	<b>51.542856</b>	<b>4.887962</b>	<b>G4PhysicsVector::Value(double, unsigned long&amp;) const (libG4global</b>
46.428570	62.942856	4.402959	G4UrbanMscModel::SampleCosineTheta(double, double) (libG4proc
30.142857	316.314279	2.858536	G4SteppingManager::DefinePhysicalStepLength() (libG4tracking.so:
28.571428	28.571428	2.709513	CLHEP::RanecuEngine::flat() (libG4clhep.so: RanecuEngine.cc,197)
27.257142	27.257142	2.584875	CLHEP::RanecuEngine::flatArray(int, double*) (libG4clhep.so: Ranec
27.085714	40.428571	2.568618	G4VEnergyLossProcess::PostStepGetPhysicalInteractionLength(G4Tr
26.514285	316.999994	2.514428	G4SteppingManager::InvokePSDIP(unsigned long) (libG4tracking.so
24.685714	24.685714	2.341019	__GI_memcmp (libc-2.12.so)
20.685714	35.428571	1.961687	G4Navigator::ComputeStep(CLHEP::Hep3Vector const&, CLHEP::Hep
20.085714	20.085714	1.904788	__ieee754_log (libm-2.12.so)
17.800000	17.800000	1.688027	G4ParticleChange::CheckIt(G4Track const&) (libG4track.so: G4Partic
16.000000	1040.485693	1.517327	G4SteppingManager::Stepping() (libG4tracking.so: G4SteppingMana
15.800000	15.800000	1.498361	__strlen_sse2 (libc-2.12.so)
15.257143	26.742857	1.446880	std::_Rb_tree<G4String, std::pair<G4String const, double>, std::S
13.542857	13.542857	1.284309	__int_malloc (libc-2.12.so)
13.514285	13.514285	1.281600	CLHEP::RandGaussQ::transformQuick(double) (libG4clhep.so: Rand
13.257143	26.799999	1.257214	__libc_malloc (libc-2.12.so)

Command Panel  
opensp>>

# ossusertime LArTest: Hot Call Path

- Relationship between caller and callee
- The paths through the application that take the most time

The screenshot shows the 'User Time [1]' application interface. The 'Process Control' section has a 'Run' button highlighted by a red dashed line. Below it, the 'Stats Panel [1]' shows 'Showing Hot Callpath Report: Executables: lArTest Host: tevnfsg4 Pids: 1 Threads: 1'. The main area displays a table with the following data:

Exclusive CPU time in seconds.	Inclusive CPU time in seconds.	% of Total Exclusive CPU Time	Call Stack Function (defining location)
			@ 517 in monitor_main (libmonitor.so.0.0.0: main.c,492)
			@ 160 in main (lArTest: lArTest.cc,39)
			@ 524 in G4UImanager::ApplyCommand(char const*) (libG4intercoms.so: G4UImanager.cc,...)
			@ 231 in G4UIcommand::DoIt(G4String) (libG4intercoms.so: G4UIcommand.cc,124)
			@ 378 in G4UIcontrolMessenger::SetNewValue(G4UIcommand*, G4String) (libG4intercoms....)
			@ 297 in G4UImanager::ExecuteMacroFile(char const*) (libG4intercoms.so: G4UImanager.cc...)
			@ 216 in G4UIbatch::SessionStart() (libG4intercoms.so: G4UIbatch.cc,198)
			@ 171 in G4UIbatch::ExecCommand(G4String const&) (libG4intercoms.so: G4UIbatch.cc,169)
			@ 524 in G4UImanager::ApplyCommand(char const*) (libG4intercoms.so: G4UImanager.cc,...)
			@ 231 in G4UIcommand::DoIt(G4String) (libG4intercoms.so: G4UIcommand.cc,124)
			@ 384 in G4RunMessenger::SetNewValue(G4UIcommand*, G4String) (libG4run.so: G4Run...)
			@ 273 in G4RunManager::BeamOn(int, char const*, int) (libG4run.so: G4RunManager.cc,263)
			@ 367 in G4RunManager::DoEventLoop(int, char const*, int) (libG4run.so: G4RunManager.c...)
			@ 399 in G4RunManager::ProcessOneEvent(int) (libG4run.so: G4RunManager.cc,397)
			@ 184 in G4EventManager::DoProcessing(G4Event*) (libG4event.so: G4EventManager.cc,99)
			@ 126 in G4TrackingManager::ProcessOneTrack(G4Track*) (libG4tracking.so: G4TrackingMa...)
			@ 191 in G4SteppingManager::Stepping() (libG4tracking.so: G4SteppingManager.cc,118)
			@ 417 in G4SteppingManager::InvokeAlongStepDoItProcs() (libG4tracking.so: G4SteppingM...)
			@ 1424 in G4VEnergyLossProcess::AlongStepDoIt(G4Track const&, G4Step const&) (libG4pr...)
15.142857	1.436042		@ 317 in G4UniversalFluctuation::SampleFluctuations(G4MaterialCutsCouple const*, G4Dy...

The 'Command Panel' at the bottom shows 'openss >>'.

# ossusertime LArTest: Hot Call (Source)

- Exclusive time on highlighted lines that indicate relatively high CPU times

File Tools Help

User Time [1]

Process Control

Run Cont Pause Update Terminate

Status: Process Loaded: Click on the "Run" button to begin the experiment.

Stats Panel [1] ManageProcessesPanel [1] Source Panel [1]

Exclusive CPU time in seconds: /g4/g4p/build/g4.10.3.p01/geant4.10.3.p01/source/processes/electromagnetic/standard/src/G4UniversalFluctuation.cc

Exclusive CPU time in seconds	Source Code
0.057143	304 }
0.485714	305
0.742857	306 G4double w2 = alfa*e0;
0.171429	307 G4double w = (tmax-w2)/tmax;
0.514286	308 if(w > 0.0) {
>> 15.142857	309 const G4int nb = G4Poisson(p3);
	310 if(nb > 0) {
	311 if(nb > sizearray) {
	312 sizearray = nb;
	313 delete [] rmdmarray;
	314 rmdmarray = new G4double[nb];
	315 }
	316 rmdmEngineF->flatArray(nb, rmdmarray);
	317 for (G4int k=0; k<nb; ++k) { loss += w2/(1.-w*rmdmarray[k]); }
	318 }
	319 }
0.742857	320 if(emean > 0.0) { SampleGauss(rmdmEngineF, emean, sig2e, loss); }
0.657143	321 }
0.485714	322 losstot += loss;
	323 }
	324 //G4cout << "Vavilov: " << losstot << " Nstep= " << nstep << G4endl;
	325
	326 return losstot;

Command Panel

openss>>

# OpenSpeedshop: Experiments with Hardware Counters

- Periodic sampling hardware counters (hwcsamp)
- Supports both derived and non-derived PAPI presets
  - Ex.: Wilson intel12 nodes: 58 available events, 14 are derived
- A list of some possible hardware counter combinations

For <b>Xeon</b> 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, LAST_LEVEL_CACHE_REFERENCES	L3 cache info
MEM_UNCORE_RETIRED:REMOTE_DRAM, MEM_UNCORE_RETIRED:LOCAL_DRAM	Local/nonlocal memory access
For <b>Opteron</b> 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_MISSES:ALL_CORES	L3 cache

# osshwcsamp LArTest: Hardware Counter Sampling

- 'papi\_avail -a' will show available papi events on a system
- Metrics for INS, FLOPS, memory and resource patterns, ...

The screenshot shows the HWCSamp Panel interface. A red box highlights the text "PAPI hwc" in the Process Control section. A red dashed arrow points from this box to a table of PAPI metrics. The table has columns for Exclusive CPU time, % of CPU time, papi\_tot\_cyc, papi\_tot\_ins, papi\_fp\_ops, and Function (defining location). The first row of data is circled in red.

Exclusive CPU time in seconds.	% of CPU time	papi_tot_cyc	papi_tot_ins	papi_fp_ops	Function (defining location)
103.844443	9.880537	238072285047	103028210581	16893047741	G4UniversalFluctuation::SampleFluctuations(G4MaterialCutsCoupl
55.111111	5.243683	126318620429	102749596516	8946035679	G4PhysicsVector::Value(double, unsigned long&) const (libG4glol
46.555555	4.429644	106775342756	86543030788	7919089722	G4UrbanMscModel::SampleCosineTheta(double, double) (libG4p
31.444444	2.991860	72048169097	58654838134	5042279306	G4SteppingManager::DefinePhysicalStepLength() (libG4tracking.s
28.955555	2.755048	66430726872	54158585775	4639678425	G4VEnergyLossProcess::PostStepGetPhysicalInteractionLength(G4
28.488889	2.710646	65348816099	53202548312	4609499319	CLHEP::RanecuEngine::flat() (libG4clhep.so: RanecuEngine.cc,197
27.955555	2.659901	64028514071	52208146862	4480292193	CLHEP::RanecuEngine::flatArray(int, double*) (libG4clhep.so: Ran
26.622222	2.533037	61110837179	49772120291	4327842442	G4SteppingManager::InvokePSDIP(unsigned long) (libG4tracking.
24.444444	2.325827	56020565509	45584419721	3972397216	__GI_memcmp (libc-2.12.so)
19.733333	1.877577	45257897510	36798636053	3148575513	__ieee754_log (libm-2.12.so)
19.355555	1.841632	44392061033	36175379636	3100280785	G4Navigator::ComputeStep(CLHEP::Hep3Vector const&, CLHEP::H
16.400000	1.560419	37599864927	30652840846	2624429457	G4ParticleChange::CheckIt(G4Track const&) (libG4track.so: G4Pa
15.644444	1.488529	35836916031	29148246541	2526679670	__strlen_sse2 (libc-2.12.so)
15.466667	1.471614	35422081311	28829396851	2495986647	G4SteppingManager::Stepping() (libG4tracking.so: G4SteppingMa



# 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.

- LArTest (Overall): 5 GeV pi- (Intel Xeon X5650@2.67GHz)
  - IPC = 0.79 (relatively small)
  - FMO = 0.32

## Other useful OSS Features

- Flexible analysis options (GUI, command line, online)
- Export report data in different formats (text, cvs, chart)
- Multi-threading capability
- Compare two experiments (osscompare): examples
  - two releases
  - two experiments with the different numbers of threads
- Call path analysis based on DB
- Experiments for parallel codes (MPI tracing)

## Summary and Plan

- Developed a standalone Geant4 application (LArtest) with a LAr fiducial volume and added functionalities for computing performance measurement and analysis
- Profiled LArTest with OpenSpeedshop (and IgProf)
- Extend the test with other geometry descriptions
  - protodune.gdml (v3 from Tom Junk)
  - GDML extension for material properties
- Monitor Geant4 part of computing performance changes for LAr-based detectors by
  - Beam energy
  - Particle type
  - Physics list
  - Geant4 (reference) release

# LArTest: IgProf (TOTAL MEM)

Mozilla Firefox

Geant4/... Mail - syj... GeantV ... Simulati... Geant4/... http...html Intel® M... https.../self x Meeting ... MKL ER... About b...

https://g4cpt.fnal.gov/cgi-bin/igprof-navigator/lArTest/IgProf\_pi-.FTFP\_BERT.5\_MEM\_TOTAL\_END/st

Most Visited Fermilab Linux Distros Geant4 LXR CMSSW LXR Fermilab Redmine Geant4 Cern Mail FermiPoint Workday

## IgProf\_pi-.FTFP\_BERT.5\_MEM\_TOTAL\_END - lArTest, igprof-navigator

Counter: MEM\_TOTAL

[Back to profiles index](#)

Counter: MEM\_TOTAL, first 100

Sorted by self cost

[Sort by cumulative cost](#)

Rank	% total	Counts		Calls		Paths		Symbol name	
		to / from this	Total	to / from this	Total	Including child / parent	Total		
20	91.96	34,069,068,357	869,743,283	0	869,743,283	869,743,283	12,785	12,785	std::basic_string<char, std::char_traits<char>, std::allocator<char>>:: Rep:: S create(unsigned long, unsigned long, std::allocator<char>)
33	7.25	2,686,965,840	50,115						deflateInit2
55	0.19	70,819,840	8,645						llvm::raw_ostream::SetBuffered()
60	0.12	45,809,937	210						TBuffer::TBuffer(TBuffer::EMode, int)
41	0.08	30,122,056	190						TKey::TKey(TObject const*, char const*, int, TDirectory*)
95	0.07	26,610,929	16						TStorage::ReAllocChar(char*, unsigned long, unsigned long)
87	0.04	13,487,544	80,283						G4InuclCollider::collide(G4InuclParticle*, G4InuclParticle*, G4CollisionOutput&)
129	0.03	12,453,440	311,336						std::pair<std:: Rb tree iterator<unsigned long>, bool> std:: Rb tree<unsigned long, unsigned long, std:: Identity<unsigned long>, std::le
159	0.02	8,700,314	5,834						llvm::SmallVectorBase::grow_pod(void*, unsigned long, unsigned long)
183	0.02	6,534,144	2,070						llvm::DenseMap<clang::IdentifierInfo*, llvm::SmallVector<clang::Decl*, 2u>, llvm::DenseMapInfo<clang::IdentifierInfo*> >::grow(unsigned i
222	0.02	5,788,928	1,523						std::vector<double, std::allocator<double>>:: M fill_insert( gnu_cxx:: normal_iterator<double*, std::vector<double, std::allocator<doubl
269	0.01	4,321,280	250						clang::Decl::operator new(unsigned long, clang::ASTContext const&, unsigned int, unsigned long)
273	0.01	4,125,032	4,650						std::vector<double, std::allocator<double>>::reserve(unsigned long)
277	0.01	4,023,395	405						llvm::BumpPtrAllocatorImpl<llvm::MallocAllocator, 4096ul, 4096ul>::Allocate(unsigned long, unsigned long)
293	0.01	3,404,000	370						G4Fancy3DNucleus::G4Fancy3DNucleus()
299	0.01	3,237,888	3,162						llvm::DenseMap<unsigned long, unsigned int, llvm::DenseMapInfo<unsigned long>>::grow(unsigned int)
306	0.01	3,000,184	1						std::vector<clang::Decl*, std::allocator<clang::Decl*>>::resize(unsigned long)

Back to summary

# osshwcsamp LArTest: Libraries

The screenshot shows the HWCSamp Panel interface. At the top, there are menu items: File, Tools, and Help. Below the menu is a 'Process Control' section with buttons for Run, Cont, Pause, Update, and Terminate. The status bar indicates 'Process Loaded: Click on the "Run" button to begin the experiment.' Below this is a 'Stats Panel [1]' and 'ManageProcessesPanel [1]' section. A toolbar contains icons for various actions, and a 'View/Display Choice' dropdown is set to 'Linked Objects'. The main area displays 'Executables: (none) Host: tevnfs4 Pids: 1 Threads: 1'. A table lists linked objects with their performance metrics. A vertical bar on the left of the table shows a color-coded distribution of the data.

Exclusive CPU time in seconds.	% of CPU Time	papi_tot_cyc	papi_tot_ins	papi_fp_ops	LinkedObject
-440.733329	41.934665	1010287183293	821706415939	71803047505	libG4processes.so
-97.644443	9.290623	223904126558	182246151472	15765064204	libG4tracking.so
-87.755555	8.349720	201052941467	163616287094	14202717251	libc-2.12.so
-82.333333	7.833809	188831766186	153712252843	13281540337	libG4track.so
-82.288888	7.829580	188649205711	153651342384	13215331827	libG4clhep.so
-63.311110	6.023893	145149224810	118154061380	10186308437	libG4geometry.so
-55.244444	5.256370	126625827775	102999244344	8968084781	libG4global.so
-46.977777	4.469817	107757313701	87669000278	7607755114	libG4materials.so
-36.044444	3.429538	82662840100	67114777784	5779714190	libm-2.12.so
-21.066666	2.004440	48308013013	39324161980	3399637208	libstdc++ .so.6
-16.044444	1.526588	36736152662	29844703869	2506031234	libG4particles.so
-10.066667	0.957818	23087210326	18774573855	1620048675	libG4digits_hits.so
-9.355555	0.890158	21488929168	17490775483	1515857703	libCore.so
-1.288889	0.122635	2957297314	2390532624	213586199	libG4event.so

Command Panel: opens>>

# osshwcsamp LArTest: Statements

The screenshot displays the HWCSamp Panel [1] interface. At the top, there are menu items: File, Tools, and Help. Below the menu is a toolbar with icons for Run, Cont, Pause, Update, and Terminate. The Status bar shows: Process Loaded: Click on the "Run" button to begin the experiment.

Below the status bar are two tabs: Stats Panel [1] and ManageProcessesPanel [1]. The Stats Panel [1] is active and shows a "View/Display Choice" dropdown menu with options: Functions, Statements (selected), and Linked Objects. Below the menu is a row of icons: I, U, CL, D, S, OV, SA, CC. The text "Showing Statements Report..." is displayed next to these icons.

Below the icons is a status bar: Executables: (none) Host: tevnfsg4 Pids: 1 Threads: 1.

The main area of the Stats Panel [1] contains a table with the following data:

Exclusive CPU time in seconds.	% of CPU Time	papi_tot_cyc	papi_tot_ins	papi_fp_ops	Statement Location (Line Number)
15.111111	1.649885	34611786726	28187673395	2445890514	G4UniversalFluctuation.cc(317)
9.888889	1.079704	22676791998	18494009992	1608176380	G4Poisson.hh(63)
9.711111	1.060294	22290972068	18071814460	1678450925	G4Log.hh(254)
8.844444	0.965668	20303972163	16539044037	1434925802	G4Poisson.hh(66)
7.644444	0.834648	17521817881	14227972292	1262648901	G4PhysicsVector.icc(119)
7.355555	0.803106	16869481223	13718189990	1196499843	atomicity.h(49)
7.333333	0.800679	16782831104	13697747477	1173550842	RanecuEngine.cc(240)
7.111111	0.776416	16296292862	13151885588	1225198724	G4Exp.hh(214)
6.866667	0.749727	15720896408	12800752699	1118891479	G4Poisson.hh(67)
6.511111	0.710906	14924702164	12061008215	1113240789	G4Exp.hh(217)
6.222222	0.679364	14274224923	11606734658	1022860921	G4PhysicsVector.icc(121)
6.200000	0.676938	14212372784	11562137332	1000407442	RanecuEngine.cc(216)
6.111111	0.667233	14007181041	11414649027	983279481	G4PhysicsVector.cc(501)
6.044444	0.659954	13856415014	11303890010	976015607	ThreeVector.icc(142)

At the bottom of the interface is a Command Panel with a text input field containing "openss >>".