



Profiling with TAU

Le Yan





6/6/2012

LONI Parallel Programming Workshop 2012





Three Steps of Code Development

- Debugging
 - Make sure the code runs and yields correct results
- Profiling
 - Analyze the code to identify performance bottlenecks
- Optimization
 - Make the code run faster and/or consume less resources









Profiling

- Gather performance statistics during execution
 - Inclusive and exclusive time
 - Number of calls
- Reflects performance behavior of program entities
 - Routines
 - Loops
- Implemented through
 - Sampling: OS interrupts or hardware counters
 - Instrumentation: measurement functions











What is TAU

- Tuning and Analysis Utilities
 - Developed at University of Oregon
- Scalable and flexible performance analysis toolkit
 - Performance profiling and tracing utilities
 - Performance data management and data mining
 - Automatic instrumentation through Program Database Toolkit(PDT)
 - Provides an instrumentation API









Availability on LONI and LSU HPC resrouces

- Tezpur and LONI Linux clusters
 +tau-2.19.2-intel-11.1-mvapich-1.1
- Philip
 - +tau-2.19.2-intel-11.1-mpich-1.2.7p1









How to Use

- Add the softenv key to .soft and resoft
- Compile your code with TAU compiler scripts
 - tau_f90.sh for Fortran, tau_cc.sh for C and tau_cxx.sh for C++
 - The code is instrumented automatically
- Execute the generated executable as normal
 Profile data files: profile.x.x.x
- Analyze/visualize the profiling results with paraprof









Paraprof

- Java-based analysis and visualization tool for performance data
- "pprof" is for text based profile display
- Can work with profile data generated by other profiling tools, e.g. MPIP
- Options
 - -f <file type>: specify type of performance data
 - m: perform runtime monitoring
 - --pack <file>: pack profile data into one file











Main Data Window







Main Data Window: Unstacked Bars









Function Data Window: Histogram







Function Data Window: Bar Chart







3D View



6/6/2012





Individual Thread View

File Options Windows Help File Options Windows Help Merri:: Time Yalue: Exclusive File Options Windows Help Std. Dev, node 1 Werri:: Time Yalue: Exclusive Void LAMMPS_NS:PairLjCut::compute(int, int) [tp: MPLAIreduce0 Std. Dev, node 3 Void LAMMPS_NS::Neighbor::halfbin.newfon(LA void LAMMPS_NS::Neighbor::halfbin.newfon(LA void LAMMPS_NS::Neighbor::halfbin.newfon(LA void LAMMPS_NS::FixtWE::final_integrate(int) [tfi 0.624 Imtegrated Statistics Table Show Thread Statistics Table Show Thread Call Graph on Show Thread Call Graph on Show Thread Call Graph on Show Thread Call Bart Relations nod Show User Event Statistics Window on Show Thread Call Graph on Show Thread Call Graph on Show Context Event Window on Show Thread Call Graph on Show Thread Call Graph on Show Thread Call Graph on Show Thread Call Graph on Show Context Event Window on Show Metadata for Thread and Thread to Comparison Window Weil Call Weils No::FixtWE::final_integrate() [tfi: 0.214 ImtPl.Sen:Option::check_distance) [their 0.139 [Void LAMMPS_NS::Velet::force_clear) [tverlet.cp 0.131 WPL_Sen:Comm::exchange0 [(comm.cp) 0.051 Void LAMMPS_NS::Singhbor::bin.atoms() [their cpp) 0.051 Void LAMMPS_NS::Singhbor::bin.atoms() [their cpp) 0.051 Void LAMMPS_NS::Theore::pack.comm() 0.031 MPL_Sen:Comm::exchange0 [(comm.cp) 0.051 Void LAMMPS_NS::Theore::stamp() [(timer.cpp) [4 0.037 MPL_Irecv) 0.035 Void LAMMPS_NS::Theore::stamp() [(timer.cp) [4 0.037 MPL_Irecv) Subsection Void LAMMPS_NS::Medify::initial_integrate() [(timer.cp) [4 0.037 MPL_Irecv) Subsection Void LAMMPS_NS::Modify::initial_integrate() [(timer.cp) [4 0.037 Void LAMMPS_NS::Modify::initial_integrate() [(timer.cp) [4 0.037 Void LAMMPS_NS::Modify::initial_integrate() [(timer.cp) [4	TAIL: ParaProf: /work/wap1/Ch	💶 📧 TAU: ParaProf: n,c,t 3,0,0 - /work/lyan1/ClusterTes 💶 🔸 🗙
Metric: Time Yalue: Exclusive Metric: Time Yalue: Exclusive Std. Dev. Mean node 0 node 1 node 2 node 2 node 2 node 2 node 3 node 3 node 1 node 2 node 3 node 3 n	File Options Windows Help	File Options Windows Help
Std. Dav. vid LAMMPS_NS::PairLjCutt:compute(int, int) [{p: mode 0 node 1 node 2 .051 Show Thread Statistics Text Window mode Show Thread Statistics Table on Show Thread Statistics Table on Show Thread Call Graph on Show Thread Call Path Relations show User Event Statistics Window nod Show User Event Statistics Window on Show User Event Statistics Window on Show User Event Statistics Window nod Show User Event Statistics Window nod Show User Event Statistics Window nod Show User Event Statistics Window nod Show Metadata for Thread Add Thread to Comparison Window vid LAMMPS_NS::FixNVE::final_integrate() [{fix_n 0.227 Int LAMMPS_NS::Statistics Window nod Show Metadata for Thread nod Add Thread to Comparison Window vid LAMMPS_NS::Verlet::iterate(int) [{verlet.cp}] 0.134 WPL_Cant_create() 0.128 vid LAMMPS_NS::Verlet::iterate(int) [{verlet.cp}] 0.051 0.034 vid LAMMPS_NS::Neighbor::bin_atoms0 [(neight 0.034 0.035 vid LAMMPS_NS::Time::stamp() [{timer.cpp}] { 0.035 0.035 vid LAMMPS_NS::Modify::initial_integrate() { 0.025	Metric: Time Value: Exclusive	Metric: Time Value: Exclusive Units: seconds
0.034 Void LAMMPS_NS::Noighbor::bin_atoms() [{neight 0.084 Void LAMMPS_NS::Comm::borders() [{comm.cpp} 0.084 Void LAMMPS_NS::Comm::exchange() [{comm.cpp} 0.058 Void LAMMPS_NS::AtomVecAtomic::unpack_rever 0.046 int LAMMPS_NS::AtomVecAtomic::unpack_rever 0.046 int LAMMPS_NS::Timer::stamp(int) [{timer.cpp} 0.043 Void LAMMPS_NS::Timer::stamp() [{timer.cpp} {4: 0.037 0.035 Void LAMMPS_NS::Integrate::ev_set(int) [{integrate: 0.025 0.035 Void LAMMPS_NS::Modify::initial_integrate(int) [{merate 0.025	Std. Dev. Mean node 0 node 1 node 2 node 3 nod Show Thread Bar Chart nod Show Thread Statistics Text Window nod Show Thread Statistics Table nod Show Thread Statistics Table nod Show Thread Call Graph nod Show User Event Bar Chart nod Show User Event Bar Chart nod Show User Event Bar Chart nod Show User Event Statistics Window nod Show Context Event Window nod Show Metadata for Thread nod Add Thread to Comparison Window	15.622 void LAMMPS_NS::PairLjCut::compute(int, int) [{pair_1 2.267 MPI_Allreduce() 2.051 void LAMMPS_NS::Neighbor::half_bin_newton(LAMMF 1.244 void LAMMPS_NS::Comm::communicate() [{comm.cpi 1.103 void LAMMPS_NS::Comm::reverse_communicate() [{c 0.735 void LAMMPS_NS::FixNVE::initial_integrate(int) [{fix_n 0.624 MPI_Wait() 0.403 void LAMMPS_NS::FixNVE::final_integrate() [{fix_nve.() 0.367 MPI_Init() 0.214 MPI_Recv() 0.207 int LAMMPS_NS::Neighbor::check_distance() [{neighb 0.189 Void LAMMPS_NS::Verlet::iterate(int) [{verlet.cpp} {12 0.17 void LAMMPS_NS::Verlet::force_clear() [{verlet.cpp} {0.134 0.134 MPI_Sendrecv() 0.101 MPI_Sendrecv() 0.101 MPI_Sendrecv() 0.101 MPI_Sendrecv() 0.101 MPI_Sendrecv() 0.101 MPI_Sendrecv()
ENTER FOR COMPLITATION		0.094 void LAMMPS_NS::FixSetForce::post_force(int) [{fix_se 0.084 void LAMMPS_NS::Neighbor::bin_atoms() [{neighbor. 0.084 void LAMMPS_NS::Comm::borders() [{comm.cpp} {6 0.058 void LAMMPS_NS::Comm::exchange() [{comm.cpp} {5 0.051 void LAMMPS_NS::AtomVecAtomic::unpack_reverse() 0.046 int LAMMPS_NS::AtomVecAtomic::pack_comm(int, int 0.043 void LAMMPS_NS::Timer::stamp(int) [{timer.cpp} {52 0.043 void LAMMPS_NS::Timer::stamp() [{timer.cpp} {43,1} 0.037 MPL_recv() 0.035 void LAMMPS_NS::Integrate::ev_set(int) [{integrate.cp 0.025 void LAMMPS_NS::Modify::initial_integrate(int) [{modi
	ENTER FOR COMPUTATION	

6/6/2012

LONI Parallel Programming Workshop 2012





Comparing Multiple Threads

		ParaProf: Comparison Window 📃 🚽		
TAU: ParaProf: /work/lyan1/C	File Options Windows Help			
File Options Windows Help Metric: Time Value: Exclusive	Metric: Time Value: Exclusive Units: seconds	🔲 tau/lammps-09jan09, 📕 tau/lammps-09jan09,)/tautest/ClusterTest/lyan1/work/ - node 2)/tautest/ClusterTest/lyan1/work/ - node 3	
Std. Dev. Nean Nean Node 0	15.622 (366.584%)	4.261	void LAMMPS_NS::PairLJCut::compute(int, int) [{pair_lj_cut.cpp} (68	3, 1}-{1
node 1 node 2 node 3 node 3 nod Show Thread Bar Chart nod Show Thread Statistics Text Window	15.346	0.624 (4.063%) 🗧 🛚	MPI_Wait()	
		3.391 2.267 (66.864%)	MPI_Allreduce()	
nod Show Thread Statistics Table		0.54 📘 2.051 (379.868%) 💳	void LAMMPS_NS::Neighbor::half_bin_newton(LAMMPS_NS::NeighLi	st *) [{r
nod Show Thread Call Path Relations		0.33 🛽 1.244 (377.394%) 📕 V	void LAMMPS_NS::Comm::communicate() [{comm.cpp} {397,1}-{44	43,1}]
node Show Context Event Window node Show Metadata for Thread node Add Thread to Comparison Window		0.363 🚦 1.103 (303.314%) 📕 V	void LAMMPS_NS::Comm::reverse_communicate() [{comm.cpp} {45	50,1}-{·
		0.198 🛿 0.735 (370.682%) 📕 V	void LAMMPS_NS::FixNVE::initial_integrate(int) [{fix_nve.cpp} {63,1	.}-{103,
		0.115 0.403 (348.977%) 📱 V	void LAMMPS_NS::FixNVE::final_integrate() [{fix_nve.cpp} {107,1}-	{140,1]
		0.37 🚦 0.367 (99.193%) 🚪 N	MPI_Init()	
		0.178 0.214 (120.127%) N	MPI_Recv()	
		0.075 0.207 (274.588%) ⁱⁱ	int LAMMPS_NS::Neighbor::check_distance() [{neighbor.cpp} {915,	1}-{93
		0.189 [0.189 (100.003%) 🛘 N	MPI_Cart_create()	
LSU	4	0.186	wid LAMMPS_NS-Verlet-iterate/int) //verlet.com//128_1_/222_1\	, –
ENTER FOR COMPUTATION				Lorayene

6/6/2012

LONI Parallel Programming Workshop 2012





Callpath Profile







Options for TAU Compiler Scripts

- Display available options with "tau_xxx.sh help"
- Options
 - optVerbose: display verbose debugging information
 - optKeepFiles: keep intermediate files (instrumented source files)
 - -optDetectMemory: trace malloc/free calls









Keeping Intermediate Files (1)

```
[lyan1@poseidon2 single_file]$ 11
total 16
-rwxr-xr-x 1 lyan1 loniadmin 2163 Apr 17 09:23 mat_trans_alt.f90
-rw-r--r-- 1 lyan1 loniadmin 10300 Apr 17 09:50 mat_trans_alt.o
[lyan1@poseidon2 single_file]$ tau_f90.sh -optKeepFiles mat_trans_alt.f90
...
[lyan1@poseidon2 single_file]$ 11
total 1032
-rwxr-xr-x 1 lyan1 loniadmin 1578296 Apr 17 10:18 a.out
-rwxr-xr-x 1 lyan1 loniadmin 2163 Apr 17 09:23 mat_trans_alt.f90
-rw-r--r-- 1 lyan1 loniadmin 2493 Apr 17 10:18 mat_trans_alt.inst.f90
-rw-r--r-- 1 lyan1 loniadmin 10300 Apr 17 10:18 mat_trans_alt.o
-rw-r--r-- 1 lyan1 loniadmin 2019 Apr 17 10:18 mat_trans_alt.pdb
```



6/6/2012







Keeping Intermediate Files (2)

```
[lyan1@poseidon2 single file]$ cat mat trans alt.inst.f90
          ! Matrix dimension
          data ndim /16, 12/
          character(len=*), parameter :: FMT1="(12(1x,i4))"
          character(len=*), parameter :: FMT2="(16(1x,i4))"
              integer profiler(2) / 0, 0 /
              save profiler
              call TAU PROFILE INIT()
              call TAU PROFILE TIMER (profiler, '
          æ
             &MATRIXTRANS ALT1 [{mat trans alt.f90} {1,1}-{90,28}]')
            call TAU PROFILE START(profiler)
              call mpi init(ierr)
          call mpi comm size(mpi comm world, nprocs, ierr)
          call mpi comm rank(mpi comm world, myrank, ierr)
CENTER FOR COMPUTATION
   & TECHNOLOGY
```







Notes for Fortran Programmers

- Use include `mpif.h' instead of use mpi
- If free format is used with .f files, use the `optPdtF950pts=-R free' option
- If more than one module files are used, use the `-optPdtGnuFortranParser' option
- If C preprocessor directive are used, use the `optPreProcess' option









TAU Environment Variables

- TAU provides many environment variables
 - TAU_MAKEFILE
 - TAU_THROTTLE
 - TAU_OPTIONS
 - PROFILEDIR
 - TRACEDIR





6/6/2012





TAU_MAKEFILE

- Different TAU makefiles corresponds to different configurations
- The default is icpc-mpi-pdt-openmp-opari
- There are quite a few others

[lyanl@philip1 lib]\$ ls Makefile.tau-intel-11.1-mpich-1.2.7p1-* Makefile.tau-intel-11.1-mpich-1.2.7p1-callpath-icpc-mpi-compensate-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-callpath-icpc-mpi-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-depthlimit-icpc-mpi-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-mpi-compensate-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-mpi-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-mpi-pdt-openmp-opari Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-mpi-pdt-openmp-trace Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-pdt-openmp Makefile.tau-intel-11.1-mpich-1.2.7p1-icpc-pdt-openmp











TAU_CALLPATH

- Enables callpath profiling
 - Recored callpath for each event
 - Need to set TAU_MAKEFILE to one of those with callpath in their names
- TAU_CALLPATH_DEPTH
 - Level to which callpath is recorded
 - Default is 2
 - Overhead increases with the depth of callpath









Other Environment Variables

• TAU_THROTTLE

- Enable event throttling
- Purpose: reduce profiling overhead
- If a function executes more than \$TAU_THROTTLE_NUMCALLS times and has an inclusive time per call of less than TAU_THROTTLE_PERCALLS microseconds, then profiling of that function will be disabled after the threshold is reached
- PROFILEDIR
 - Controls where the profile files are written to (the default is current directory)
- TAU_OPTIONS
 - Override the default instrumentation options











Selective Profiling (1)

- Instruct TAU
 - Which part(s) of the code to profile
 - How they are profiled
- -optTauSelectFile=<file>
 - The select profiling file specifies files, functions and sections that will be included or excluded in the profiling
 - Wildcards can be used





6/6/2012





Selective Profiling (2)

[lyan1@poseidon2 src]\$ echo \$TAU_OPTIONS
-optVerbose -optTauSelectFile=/work/lyan1/ClusterTest/tautest/lammpsmpi-only/src/select.tau

[lyan1@poseidon2 src]\$ cat select.tau

BEGIN_INCLUDE_LIST

MPI#	
mpi#	
Mpi#	

END_INCLUDE_LIST





6/6/2012

LONI Parallel Programming Workshop 2012





Tracing (1)

- Recording of information about events during execution
 - Entering/exiting code region (function, loop, block...)
 - Thread/process interactions (send/receive message...)
- Save information in event record
 - Timestamp
 - CPU identifier
 - Event type and event-specific information
- Event trace is a time-sequenced stream of event records









Tracing (2)

- Pick the correct makefile using TAU_MAKEFILE (those with "trace" in the file name")
- Compile with TAU compiler scripts and run the program
- Use external utilities to analyze the trace files
 - JUMPSHOT
 - VAMPIR
- Be careful: trace files can grow very big!









Not Covered

- Database management
- Phase based profiles
- Track memory and I/O
- Instrumentation API



