TAU Performance System®

CScADS

Jul 25, 2012
Wyatt Spear, Sameer Shende, Allen D. Malony, Scott Biersdorff, Suzanne Millstein

TAU team, University of Oregon

wspear@cs.uoregon.edu

http://tau.uoregon.edu
Introduction
TAU Performance System

- http://tau.uoregon.edu/
- Multi-level performance instrumentation
  - Multi-language automatic source instrumentation
- Flexible and configurable performance measurement
- Widely-ported parallel performance profiling system
  - Computer system architectures and operating systems
  - Different programming languages and compilers
- Support for multiple parallel programming paradigms
  - Multi-threading, message passing, mixed-mode, hybrid
- Integration in complex software, systems, applications
For more information

- **TAU Website:** [http://tau.uoregon.edu](http://tau.uoregon.edu)
  - Software
  - Release notes
  - Documentation

- **TAU LiveDVD:** [http://www.hpclinux.com](http://www.hpclinux.com)
  - Boot up on your laptop or desktop
  - Includes TAU and variety of other packages
  - Include documentation and tutorial slides
What is TAU?

- TAU is a performance evaluation tool
- It supports parallel profiling and tracing
- Profiling shows you how much (total) time was spent in each routine
- Tracing shows you \textit{when} the events take place in each process along a timeline
- TAU uses a package called PDT for automatic instrumentation of the source code
- Profiling and tracing can measure time as well as hardware performance counters from your CPU
- TAU can automatically instrument your source code (routines, loops, I/O, memory, phases, etc.)
- TAU runs on all HPC platforms and it is free (BSD style license)
- TAU has instrumentation, measurement and analysis tools
  - \texttt{paraprof} is TAU’s 3D profile browser
- To use TAU’s automatic source instrumentation, you need to set a couple of environment variables and substitute the name of your compiler with a TAU shell script
Performance Optimization Cycle

- Design experiment
- Collect performance data
- Calculate metrics
- Analyze results
- Visualize results
- Identify bottlenecks and causes
- Tune performance

[Diagram showing the cycle with steps labeled: Instrumentation, Measurement, Analysis, Presentation, Optimization]
TAU Instrumentation Approach

- Supports both direct and indirect performance observation
  - Direct instrumentation of program (system) code (probes)
  - Instrumentation invokes performance measurement
  - Event measurement: performance data, meta-data, context
  - Indirect mode supports sampling based on periodic timer or hardware performance counter overflow based interrupts

- Support for standard program events
  - Routines, classes and templates
  - Statement-level blocks and loops
  - Begin/End events (Interval events)

- Support for user-defined events
  - Begin/End events specified by user
  - Atomic events (e.g., size of memory allocated/freed)
  - Flexible selection of event statistics

- Provides static events and dynamic events
Inclusive and Exclusive Profiles

- Performance with respect to code regions
- Exclusive measurements for region only
- Inclusive measurements includes child regions

```c
int foo()
{
    int a;
    a = a + 1;
    bar();
    a = a + 1;
    return a;
}
```
## Interval Events, Atomic Events in TAU

### Interval Event

- **e.g., routines (start/stop)**

### Atomic Events

- **trigger with value**

---

<table>
<thead>
<tr>
<th>%Time</th>
<th>Exclusive msec</th>
<th>Inclusive total msec</th>
<th>#Call</th>
<th>#Subrs</th>
<th>Inclusive Name</th>
<th>usec/call</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.0</td>
<td>0.187</td>
<td>1.105</td>
<td>1</td>
<td>44</td>
<td>1105659 int main(int. char **)</td>
<td>C</td>
</tr>
<tr>
<td>93.2</td>
<td>1.030</td>
<td>1.030</td>
<td>1</td>
<td>0</td>
<td>1030654 MPI_Init()</td>
<td></td>
</tr>
<tr>
<td>5.9</td>
<td>0.879</td>
<td>65</td>
<td>40</td>
<td>320</td>
<td>1637 void func(int. int)</td>
<td>C</td>
</tr>
<tr>
<td>4.6</td>
<td>51</td>
<td>51</td>
<td>40</td>
<td>0</td>
<td>1277 MPI_Barrier()</td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>13</td>
<td>13</td>
<td>120</td>
<td>0</td>
<td>111 MPI_Recv()</td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>9</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>9328 MPI_Finalize()</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.137</td>
<td>0.137</td>
<td>120</td>
<td>0</td>
<td>1 MPI_Send()</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.086</td>
<td>0.086</td>
<td>40</td>
<td>0</td>
<td>2 MPI_Bcast()</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.002</td>
<td>0.002</td>
<td>1</td>
<td>0</td>
<td>2 MPI_Comm_size()</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.001</td>
<td>0.001</td>
<td>1</td>
<td>0</td>
<td>1 MPI_Comm_rank()</td>
<td></td>
</tr>
</tbody>
</table>

---

**USER EVENTS Profile :NODE 0, CONTEXT 0, THREAD 0**

<table>
<thead>
<tr>
<th>NumSamples</th>
<th>MaxValue</th>
<th>MinValue</th>
<th>MeanValue</th>
<th>Std. Dev.</th>
<th>Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>365</td>
<td>5.138E+04</td>
<td>44.39</td>
<td>3.09E+04</td>
<td>1.234E+04</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>365</td>
<td>5.138E+04</td>
<td>2064</td>
<td>3.115E+04</td>
<td>1.21E+04</td>
<td>Heap Memory Used (KB) : Exit</td>
</tr>
<tr>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>0</td>
<td>Message size for broadcast</td>
</tr>
</tbody>
</table>

---

% setenv TAU_CALLPATH_DEPTH 0
% setenv TAU_TRACK_HEAP 1
Atomic Events, Context Events

<table>
<thead>
<tr>
<th>Time</th>
<th>Exclusive</th>
<th>Inclusive</th>
<th>#Call</th>
<th>#Subs</th>
<th>Inclusive Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>msec</td>
<td>total msec</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100.0</td>
<td>0.253</td>
<td>1.106</td>
<td>1</td>
<td>44</td>
<td>1106701 int main(int, char **) C</td>
</tr>
<tr>
<td>93.2</td>
<td>1.031</td>
<td>1.031</td>
<td>1</td>
<td>0</td>
<td>1031311 MPI_Init()</td>
</tr>
<tr>
<td>6.0</td>
<td>1</td>
<td>66</td>
<td>40</td>
<td>0</td>
<td>1650 void func(int, int) C</td>
</tr>
<tr>
<td>5.7</td>
<td>63</td>
<td>63</td>
<td>40</td>
<td>0</td>
<td>1588 MPI_Barrier()</td>
</tr>
<tr>
<td>0.8</td>
<td>9</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>9119 MPI_Finalize()</td>
</tr>
<tr>
<td>0.1</td>
<td>1</td>
<td>1</td>
<td>120</td>
<td>0</td>
<td>10 MPI_Recv()</td>
</tr>
<tr>
<td>0.0</td>
<td>0.141</td>
<td>0.141</td>
<td>120</td>
<td>0</td>
<td>1 MPI_Send()</td>
</tr>
<tr>
<td>0.0</td>
<td>0.085</td>
<td>0.085</td>
<td>40</td>
<td>0</td>
<td>2 MPI_Bcast()</td>
</tr>
<tr>
<td>0.0</td>
<td>0.001</td>
<td>0.001</td>
<td>1</td>
<td>0</td>
<td>1 MPI_Comm_size()</td>
</tr>
<tr>
<td>0.0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0 MPI_Comm_rank()</td>
</tr>
</tbody>
</table>

USER EVENTS Profile :NODE 0, CONTEXT 0, THREAD 0

<table>
<thead>
<tr>
<th>NumSamples</th>
<th>MaxValue</th>
<th>MinValue</th>
<th>MeanValue</th>
<th>Std. Dev.</th>
<th>Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>5.139E+04</td>
<td>44.39</td>
<td>3.091E+04</td>
<td>1.234E+04</td>
<td>Message size for broadcast: Entry</td>
</tr>
<tr>
<td>40</td>
<td>5.139E+04</td>
<td>3.091E+04</td>
<td>1.234E+04</td>
<td>0</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>40</td>
<td>5.139E+04</td>
<td>3.091E+04</td>
<td>1.234E+04</td>
<td>0</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>1</td>
<td>2067</td>
<td>2067</td>
<td>2067</td>
<td>0</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>1</td>
<td>2066</td>
<td>2066</td>
<td>2066</td>
<td>0</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>1</td>
<td>5.139E+04</td>
<td>5.139E+04</td>
<td>5.139E+04</td>
<td>0.0006905</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>1</td>
<td>57.56</td>
<td>57.56</td>
<td>57.56</td>
<td>0</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>120</td>
<td>5.139E+04</td>
<td>1.138E+04</td>
<td>1.138E+04</td>
<td>1.187E+04</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>120</td>
<td>5.139E+04</td>
<td>1.138E+04</td>
<td>1.138E+04</td>
<td>1.187E+04</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>1</td>
<td>44.39</td>
<td>44.39</td>
<td>44.39</td>
<td>0</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
<tr>
<td>40</td>
<td>5.036E+04</td>
<td>2068</td>
<td>3.011E+04</td>
<td>1.227E+04</td>
<td>Heap Memory Used (KB) : Entry</td>
</tr>
</tbody>
</table>

% setenv TAU_CALLPATH_DEPTH 1
% setenv TAU_TRACK_HEAP 1
Parallel Profile Visualization: ParaProf
Overview of different methods of instrumenting applications
Instrumentation: Events in TAU

• Event types
  – Interval events (begin/end events)
    – measures performance between begin and end
    – metrics monotonically increase
  – Atomic events
    – used to capture performance data state

• Code events
  – Routines, classes, templates
  – Statement-level blocks, loops

• User-defined events
  – Specified by the user

• Abstract mapping events
Instrumentation Techniques

- Events defined by instrumentation access
- Instrumentation levels
  - Source code
  - Object code
  - Runtime system
  - Library code
  - Executable code
  - Operating system
- Different levels provide different information
- Different tools needed for each level
- Levels can have different granularity
Instrumentation Techniques

- Static instrumentation
  - Program instrumented prior to execution

- Dynamic instrumentation
  - Program instrumented at runtime

- Manual and automatic mechanisms

- Tool required for automatic support
  - Source time: preprocessor, translator, compiler
  - Link time: wrapper library, preload
  - Execution time: binary rewrite, dynamic

- Advantages / disadvantages
TAU Performance System Architecture

Instrumentation
- source code
- object code
- library wrapper
- binary code
- virtual machine

Measurement
- event creation and management
  - event identifier
  - entry/exit events
  - atomic events
  - event mapping
  - event control

Profiling
- statistics
- mapping (callpath)
- atomic profiles
- entry/exit profiles
- profile I/O
- sampling profiles

Tracing
- trace buffering
- record creation
- trace I/O
- timestamp generation
- trace filtering

Performance data sources
- timing
- hardware counters
- system counters

OS and runtime system modules
- threading
- interrupts
- runtime system
- ...
Program Database Toolkit (PDT)

- Application / Library
  - C / C++ parser
  - Fortran parser F77/90/95
  - IL analyzer
  - IL analyzer
  - C / C++ IL analyzer
  - Program Database Files

- Program documentation
  - Application component glue
  - C++ / F90/95 interoperability
  - Automatic source instrumentation

- DUCTAPE
  - PDBhtml
  - SILOON
  - CHASM
  - TAU_instr
Automatic Source-Level Instrumentation in TAU using Program Database Toolkit (PDT)
Using TAU with source instrumentation

- TAU supports several measurement options (profiling, tracing, profiling with hardware counters, etc.)
- Each measurement configuration of TAU corresponds to a unique stub makefile and library that is generated when you configure it.
- To instrument source code using PDT:
  - Choose an appropriate TAU stub makefile in <arch>/lib:
    `% module load tau`
    `% export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-papi-mpi-pdt-pgi`
    `% export TAU_OPTIONS='--optVerbose …'` (see tau_compiler.sh -help)
    And use tau_f90.sh, tau_cxx.sh or tau_cc.sh as Fortran, C++ or C compilers:
    `% mpif90 foo.f90`
    changes to
    `% tau_f90.sh foo.f90`

- Execute application and analyze performance data:
  `% pprof` (for text based profile display)
  `% paraprof` (for GUI)
TAU Measurement Configuration

% cd $TAULIBDIR; ls Makefile.*
Makefile.tau-pdt-pgi
Makefile.tau-mpi-pdt-pgi
Makefile.tau-pthread-pdt-pgi
Makefile.tau-papi-mpi-pdt-pgi
Makefile.tau-papi-pthread-pdt-pgi
Makefile.tau-mpi-papi-pdt-pgi
Makefile.tau-mpi-pdt-openmp-opari

• For an MPI+F90 application, you may want to start with:

Makefile.tau-mpi-pdt-pgi
  – Supports MPI instrumentation & PDT for automatic source instrumentation
  – % export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-mpi-pdt-pgi
  – % tau_f90.sh matrix.f90 -o matrix
  – % mpirun -np 256 ./matrix
  – % paraprof
Usage Scenarios: Routine Level Profile

- Goal: What routines account for the most time? How much?

- Flat profile with wallclock time:
  - Metric: PVIRTUAL_TIME
  - Value: Exclusive
  - Units: seconds

```
9647.318  LEQ_IKSWEPT
4357.213  LEQ_BICGS0T
2669.887  LEQ_MATVECT
1777.752  SOLVE_SPECIES_EQ
1417.986  SOLVE_LIN_EQ
1028.448  PHYSICAL_PROP
  783.402  RRATES
  682.376  LEQ_MSOLVET
  530.858  INIT_AB_M
  463.788  CALC_MASS_FLUX_SPHR
  446.025  INIT_MU_S
  421.747  CALC_RESID_S
  381.363  SOLVE_ENERGY_EQ
  371.199  SOURCE_PHI
  258.829  DRAG_GS
```
Solution: Generating a flat profile with MPI

% module load tau
% export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-mpi-pdt-pgi
% export PATH=$TAUROOTDIR/x86_64/bin:$PATH
% tau_f90.sh matmult.f90 -o matmult
(Or edit Makefile and change F90=tau_f90.sh)

% qsub -I -l nodes=1:ppn=8 -X
% mpirun -np 8 ./matmult
% pprof
% paraprof &
OR
% paraprof --pack app.ppk
  Move the app.ppk file to your desktop.

% paraprof app.ppk
% paraprof &
Automatic Instrumentation

- We now provide compiler wrapper scripts
  - Simply replace ftn with tau_f90.sh
  - Automatically instruments Fortran source code, links with TAU MPI Wrapper libraries.

- Use tau_cc.sh and tau_cxx.sh for C/C++
Compile-Time Environment Variables

- Optional parameters for TAU_OPTIONS: [tau_compiler.sh –help]
  - optVerbose      Turn on verbose debugging messages
  - optCompInst     Use compiler based instrumentation
  - optNoCompInst   Do not revert to compiler instrumentation if source instrumentation fails.
  - optDetectMemoryLeaks  Turn on debugging memory allocations/de-allocations to track leaks
  - optTrackIO      Turn on tracking POSIX IO by linking TAU’s wrapper library
  - optKeepFiles    Does not remove intermediate .pdb and .inst.* files
  - optPreProcess   Preprocess Fortran sources before instrumentation
  - optTauSelectFile=""  Specify selective instrumentation file for tau_instrumentor
  - optLinking=""    Options passed to the linker. Typically
                     $(TAU_MPI_FLIBS) $(TAU_LIBS) $(TAU_CXXLIBS)
  - optCompile=""    Options passed to the compiler. Typically
                     $(TAU_MPI_INCLUDE) $(TAU_INCLUDE) $(TAU_DEFS)
  - optPdtF95Opts=""  Add options for Fortran parser in PDT (f95parse/gfparsre)
  - optPdtF95Reset="" Reset options for Fortran parser in PDT (f95parse/gfparsre)
  - optPdtCxxOpts=""  Options for C parser in PDT (cparsre). Typically
                     $(TAU_MPI_INCLUDE) $(TAU_INCLUDE) $(TAU_DEFS)
  - optPdtCxxOpts=""  Options for C++ parser in PDT (cxxparsre). Typically
                     $(TAU_MPI_INCLUDE) $(TAU_INCLUDE) $(TAU_DEFS)
Compiling Fortran Codes with TAU

• If your Fortran code uses free format in .f files (fixed is default for .f), you may use:
  % export TAU_OPTIONS=''-optPdtF95Opts="-R free" -optVerbose'"

• To use the compiler based instrumentation instead of PDT (source-based):
  % export TAU_OPTIONS=''-optCompInst -optVerbose'"

• If your Fortran code uses C preprocessor directives (#include, #ifdef, #endif):
  % export TAU_OPTIONS=''-optPreProcess -optVerbose -optDetectMemoryLeaks'"

• To use an instrumentation specification file:
  % export TAU_OPTIONS=''-optTauSelectFile=mycmd.tau -optVerbose -optPreProcess'"
  % cat mycmd.tau
  BEGIN_INSTRUMENT_SECTION
  memory file="foo.f90" routine="#"
  # instruments all allocate/deallocate statements in all routines in foo.f90
  loops file="*" routine="#"
  io file="abc.f90" routine="FOO"
  END_INSTRUMENT_SECTION
Usage Scenarios: Compiler-based Instrumentation

- Goal: Easily generate routine level performance data using the compiler instead of PDT for parsing the source code
Use Compiler-Based Instrumentation

% export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-mpi-pdt-pgi
% export TAU_OPTIONS='--optCompInst --optVerbose'
% module load tau
% make F90=tau_f90.sh
(Or edit Makefile and change F90=tau_f90.sh)

% mpirun -np 8 ./a.out
% paraprof --pack app.ppk
  Move the app.ppk file to your desktop.
% paraprof app.ppk
Re-writing Binaries

- Support for both static and dynamic executables
- Specify the list of routines to instrument/exclude from instrumentation
- Specify the TAU measurement library to be injected
- Simplify the usage of TAU:
  - To instrument:
    - `% tau_run a.out –o a.inst`
  - To perform measurements, execute the application:
    - `% mpirun –np 4 ./a.inst`
  - To analyze the data:
    - `% paraprof`
tau_run with NAS PBS

/home/livetau% cd ~/tutorial
/home/livetau/tutorial% # Build an uninstrumented bt NAS Parallel Benchmark
/home/livetau/tutorial% make bt CLASS=W NPROCS=4
/home/livetau/tutorial% cd bin
/home/livetau/tutorial/bin% # Run the instrumented code
/home/livetau/tutorial/bin% mpirun -np 4 ./bt_W.4
/home/livetau/tutorial/bin% # Instrument the executable using TAU with DyninstAPI
/home/livetau/tutorial/bin% tau_run ./bt_W.4 -o ./bt.i
/home/livetau/tutorial/bin% rm -rf profile.* MULT*
/home/livetau/tutorial/bin% mpirun -np 4 ./bt.i
/home/livetau/tutorial/bin% paraprof
/home/livetau/tutorial/bin% # Choose a different TAU configuration
/home/livetau/tutorial/bin% ls $TAU/libTAUsh
libTAUsh-depthlimit-mpi-pdt.so*
libTAUsh-mpi-pdt.so*
libTAUsh-mpi-pdt-upc.so*
libTAUsh-mpi-python-pdt.so*
libTAUsh-papi-mpi-pdt.so*
libTAUsh-papi-mpi-pdt-upc.so*
libTAUsh-papi-mpi-pdt-upc-udp.so*
libTAUsh-papi-mpi-pdt-vampirtrace-trace.so*
libTAUsh-papi-mpi-python-pdt.so*
/home/livetau/tutorial/bin%
/home/livetau/tutorial/bin% tau_run -XrunTAUsh-papi-mpi-pdt-vampirtrace-trace bt_W.4 -o bt.vpt
/home/livetau/tutorial/bin% setenv VT_METRICS PAPI_FP_INS:PAPI_L1_DCM
/home/livetau/tutorial/bin% mpirun -np 4 ./bt.vpt
/home/livetau/tutorial/bin% vampir bt.vpt.otf &
Usage Scenarios: Instrument a Python program

- **Goal:** Generate a flat profile for a Python program
Library wrapping: tau_gen_wrapper

• How to instrument an external library without source?
  – Source may not be available
  – Library may be too cumbersome to build (with instrumentation)

• Build a library wrapper tools
  – Used PDT to parse header files
  – Generate new header files with instrumentation files
  – Three methods to instrument: runtime preloading, linking, redirecting headers

• Application is instrumented

• Add the `–optTauWrapFile=<wrapperdir>/link_options.tau` file to TAU_OPTIONS env var while compiling with tau_cc.sh, etc.

• Wrapped library
  – Redirects references at routine callsite to a wrapper call
  – Wrapper internally calls the original
  – Wrapper has TAU measurement code
HDF5 Library Wrapping

[sameer@zorak]$ tau_gen_wrapper hdf5.h /usr/lib/libhdf5.a -f select.tau

Usage: tau_gen_wrapper <header> <library> [-r|-d|-w (default)] [-g groupname] [-i headerfile] [-c|-c++|-fortran] [-f <instr_req_file> ]

- instruments using runtime preloading (-r), or -Wl,-wrap linker (-w), redirection of header file to redefine the wrapped routine (-d)
- instrumentation specification file (select.tau)
- group (hdf5)
- tau_exec loads libhdf5_wrap.so shared library using –loadlib=<libwrap_pkg.so>
- creates the wrapper/ directory with -opt

NODE 0;CONTEXT 0;THREAD 0:

<table>
<thead>
<tr>
<th>Time</th>
<th>Exclusive</th>
<th>Inclusive</th>
<th>#Call</th>
<th>#Subrs</th>
<th>Inclusive</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>msec</td>
<td>total msec</td>
<td></td>
<td></td>
<td></td>
<td>usec/call</td>
<td></td>
</tr>
<tr>
<td>100.0</td>
<td>0.057</td>
<td>1</td>
<td>1</td>
<td>13</td>
<td>1236</td>
<td>.TAU Application</td>
</tr>
<tr>
<td>70.8</td>
<td>0.875</td>
<td>0.875</td>
<td>1</td>
<td>0</td>
<td>875</td>
<td>hid_t H5Fcreate()</td>
</tr>
<tr>
<td>9.7</td>
<td>0.12</td>
<td>0.12</td>
<td>1</td>
<td>0</td>
<td>120</td>
<td>herr_t H5Fclose()</td>
</tr>
<tr>
<td>6.0</td>
<td>0.074</td>
<td>0.074</td>
<td>1</td>
<td>0</td>
<td>74</td>
<td>hid_t H5Dcreate()</td>
</tr>
<tr>
<td>3.1</td>
<td>0.038</td>
<td>0.038</td>
<td>1</td>
<td>0</td>
<td>38</td>
<td>herr_t H5Dwrite()</td>
</tr>
<tr>
<td>2.6</td>
<td>0.032</td>
<td>0.032</td>
<td>1</td>
<td>0</td>
<td>32</td>
<td>herr_t H5Dclose()</td>
</tr>
<tr>
<td>2.1</td>
<td>0.026</td>
<td>0.026</td>
<td>1</td>
<td>0</td>
<td>26</td>
<td>herr_t H5check_version()</td>
</tr>
<tr>
<td>0.6</td>
<td>0.008</td>
<td>0.008</td>
<td>1</td>
<td>0</td>
<td>8</td>
<td>hid_t H5Screate_simple()</td>
</tr>
<tr>
<td>0.2</td>
<td>0.002</td>
<td>0.002</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>herr_t H5Tset_order()</td>
</tr>
<tr>
<td>0.2</td>
<td>0.002</td>
<td>0.002</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>hid_t H5Tcopy()</td>
</tr>
<tr>
<td>0.1</td>
<td>0.001</td>
<td>0.001</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>herr_t H5Sclose()</td>
</tr>
<tr>
<td>0.1</td>
<td>0.001</td>
<td>0.001</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>herr_t H5open()</td>
</tr>
<tr>
<td>0.0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>herr_t H5Tclose()</td>
</tr>
</tbody>
</table>
Profiling GPGPU Executions

- GPGPU compilers (e.g., CAPS hmpp and PGI) can now automatically generate GPGPU code using manual annotation of loop-level constructs and routines (hmpp).
- The loops (and routines for HMPP) are transferred automatically to the GPGPU.
- TAU intercepts the runtime library routines and examines the arguments.
- Shows events as seen from the host.
- Profiles and traces GPGPU execution.
Host (CPU) - GPU Scenarios

- **Single GPU**

- **Multi-stream**

- **Multi-CPU, Multi-GPU**
Host-GPU Measurement – Callback Method

- GPU driver libraries provide callbacks for certain routines and captures measurements
- Measurement tool registers the callbacks and processes performance data
- Application code is not modified
Method Support and Implementation

- Synchronous method
  - Place instrumentation appropriately around GPU calls (kernel launch, library routine, …)
  - Wrap (synchronous) library with performance tool

- Event queue method
  - Utilize CUDA and OpenCL event support
  - Again, need instrumentation to create and insert events in the streams with kernel launch and process events
  - Can be implemented with driver library wrapping

- Callback method
  - Utilize language-level callback support in OpenCL
  - Utilize NVIDIA CUDA Performance Tool Interface (CUPTI)
  - Need to appropriately register callbacks
GPU Performance Measurement Tools

- Support the Host-GPU performance perspective
- Provide integration with existing measurement system to facilitate tool use
- Utilize support in GPU driver library and device
- Tools
  - TAU performance system
  - Vampir
  - PAPI
  - NVIDIA CUPTI
GPU Performance Tool Interoperability

- CUDA
- OpenCL
- TAU
- PAPI
- VampirTrace
- ParaProf
- Vampir

CUPTI

parallel profile

tau2otf

parallel trace

Event queue
Callback
NVIDIA CUPTI

• NVIDIA is developing CUPTI to enable the creation of profiling and tracing tools

• Callback API
  – Interject tool code at the entry and exist to each CUDA runtime and driver API call

• Counter API
  – Query, configure, start, stop, and read the counters on CUDA-enabled devices

• CUPTI is delivered as a dynamic library

• CUPTI is released with CUDA 4.0
TAU for Heterogeneous Measurement

• Multiple performance perspectives

• Integrate Host-GPU support in TAU measurement framework
  – Enable use of each measurement approach
  – Include use of PAPI and CUPTI
  – Provide profiling and tracing support

• Tutorial
  – Use TAU library wrapping of libraries
  – Use `tau_exec` to work with binaries
    % ./a.out  (uninstrumented)
    % tau_exec –T serial –cuda ./a.out
    % paraprof
SHOC FFT Profile with Callsite Info

- TAU is able to associate callsite context information with kernel launch so that different kernel calls can be distinguished.

Each kernel (ifft1D_512, fft1D_512 and chk1D_512) is broken down by call-site, either during the single precession or double precession step.
Example: CUDA Linpack

- TAU traces with Jumpshot visualization
Example: NAMD with CUPTI
Profiling PGI Accelerator Primitives

- PGI compiler allows users to annotate source code to identify loops that should be accelerated.

- When a program is compiled with TAU, its measurement library intercepts the PGI runtime library layer to measure time spent in the runtime library routines and data transfers.

- TAU also captures the arguments:
  - array data dimensions and sizes, strides, upload and download times, variable names, source file names, row and column information, and routines.
Custom profiling
Selective Instrumentation File

• Specify a list of routines to exclude or include (case sensitive)

• # is a wildcard in a routine name. It cannot appear in the first column.

BEGIN_EXCLUDE_LIST
Foo
Bar
D#EMM
END_EXCLUDE_LIST

• Specify a list of routines to include for instrumentation

BEGIN_INCLUDE_LIST
int main(int, char **)
F1
F3
END_INCLUDE_LIST

• Specify either an include list or an exclude list!
Selective Instrumentation File

- Optionally specify a list of files to exclude or include (case sensitive)
- * and ? may be used as wildcard characters in a file name
  ```
  BEGIN_FILE_EXCLUDE_LIST
  f*.f90
  Foo?.cpp
  END_FILE_EXCLUDE_LIST
  ```
- Specify a list of routines to include for instrumentation
  ```
  BEGIN_FILE_INCLUDE_LIST
  main.cpp
  foo.f90
  END_FILE_INCLUDE_LIST
  ```
Selective Instrumentation File

- User instrumentation commands are placed in INSTRUMENT section
- ? and * used as wildcard characters for file name, # for routine name
- \ as escape character for quotes
- Routine entry/exit, arbitrary code insertion
- Outer-loop level instrumentation

```plaintext
BEGIN_INSTRUMENT_SECTION
loops file="foo.f90" routine="matrix#"
memory file="foo.f90" routine="#"
io routine="matrix#"
[static/dynamic] phase routine="MULTIPLY"
dynamic [phase/timer] name="foo" file="foo.cpp" line=22 to line=35
file="foo.f90" line = 123 code = " print *, " Inside foo\"
exit routine = "int foo()" code = "cout <<"exiting foo"<<endl;"
END_INSTRUMENT_SECTION
```
Instrumentation Specification

% tau_instrumentor
For selective instrumentation, use -f option
% tau_instrumentor foo.pdb foo.cpp -o foo.inst.cpp -f selective.dat
% cat selective.dat
# Selective instrumentation: Specify an exclude/include list of routines/files.
BEGIN_EXCLUDE_LIST
void quicksort(int *, int, int)
void sort_5elements(int *)
void interchange(int *, int *)
END_EXCLUDE_LIST

BEGIN_FILE_INCLUDE_LIST
Main.cpp
Foo?.c
*.C
END_FILE_INCLUDE_LIST
# Instruments routines in Main.cpp, Foo?.c and *.C files only
# Use BEGIN_[FILE]_INCLUDE_LIST with END_[FILE]_INCLUDE_LIST
Usage Scenarios: Loop Level Instrumentation

- Goal: What loops account for the most time? How much?
- Flat profile with wallclock time with loop instrumentation:

Metric: GET_TIME_OF_DAY
Value: Exclusive
Units: microseconds

Loop: MULTIPLY_MATRICES [{matmult.f90} {31,9}-{36,14}]
MPI_Recv()
81095 MAIN
49569 MPI_Bcast()
45669 Loop: MAIN [{matmult.f90} {86,9}-{106,14}]
12412 MPI_Send()
8959 Loop: INITIALIZE [{matmult.f90} {17,9}-{21,14}]
8953 Loop: INITIALIZE [{matmult.f90} {10,9}-{14,14}]
5609.2 MPI_Finalize()
2932.667 MULTIPLY_MATRICES
2577.667 Loop: MAIN [{matmult.f90} {117,9}-{128,14}]
2091.8 MPI_Barrier()
1875.667 Loop: MAIN [{matmult.f90} {112,9}-{115,14}]
1833 Loop: MAIN [{matmult.f90} {71,9}-{74,14}]
107 Loop: MAIN [{matmult.f90} {77,9}-{84,14}]
30 INITIALIZE
14.25 MPI_Comm_rank()
1 MPI_Comm_size()
Solution: Generating a loop level profile

```bash
% export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-mpi-pdt-pgi
% export TAU_OPTIONS='--optTauSelectFile=select.tau --optVerbose'
% cat select.tau
BEGIN_INSTRUMENT_SECTION
loops routine="#"
END_INSTRUMENT_SECTION

% export PATH=$TAUROOTDIR/x86_64/bin:$PATH
% make F90=tau_f90.sh
(Or edit Makefile and change F90=tau_f90.sh)
% mpirun -np 8 ./a.out
% paraprof --pack app.ppk
  Move the app.ppk file to your desktop.

% paraprof app.ppk
```
ParaProf’s Source Browser: Loop Level Instrumentation

FILE OPTIONS WINDOWS HELP

Name: Loop: TRANSPORTיאות.COMPUTE SPECIES::DIFFLUX [mixavg_transport_mpp.f90] (630.5s - 656.19s)
Metric Name: PAPI.LL DCM
Value: Exclusive
Units: counts

5.0701E9, 5.0692E9, 5.0704E9, 5.0698E9, 5.0711E9, 5.0692E9

8.36336.1 |  std. dev. mean
n=10,000, n=10,000

...
Manually Instrumenting a C code

```c
#include <TAU.h>

int foo(int x) {
    TAU_START("foo");
    for (i = 0; i < x; i++) { // do work
    }
    TAU_STOP("foo");
}

int main(int argc, char **argv) {
    TAU_INIT(&argc, &argv);
    TAU_START("main");
    TAU_PROFILE_SET_NODE(rank);
    ...
    TAU_STOP("main");
}

% gcc -I<taudir>/include foo.c -o foo -L<taudir>/<arch>/lib -lTAU
% ./a.out
% pprof; paraprof

NOTE: Replace TAU_START("foo") with call TAU_START('foo')
      in Fortran. See <taudir>/include/TAU.h for full API.
```
Throttling effect of frequently called small routines
Optimization of Program Instrumentation

- Need to eliminate instrumentation in frequently executing lightweight routines
- Throttling of events at runtime (default in tau-2.17.2+):
  
  ```
  % export TAU_THROTTLE=1
  ```
  Turns off instrumentation in routines that execute over 100000 times (TAU_THROTTLE_NUMCALLS) and take less than 10 microseconds of inclusive time per call (TAU_THROTTLE_PERCALL). Use TAU_THROTTLE=0 to disable.

- Selective instrumentation file to filter events
  
  ```
  % tau_instrumentor [options] -f <file>  OR
  % export TAU_OPTIONS='--optTauSelectFile=tau.txt'
  ```

- Compensation of local instrumentation overhead
  
  ```
  % export TAU_COMPENSATE=1  (in tau-2.19.2+)
  ```
ParaProf: Creating Selective Instrumentation File
Choosing Rules for Excluding Routines

TAU: ParaProf: Selective Instrumentation File Generator

Output File: /mnt/epsilon/Users/sameer/rs/taudata/frontier/select.tau

- Exclude Throttled Routines
- Exclude Lightweight Routines

Lightweight Routine Exclusion Rules

Microseconds per call: 10
Number of calls: 100000

Excluded Routines

bool debugging(const char *) C
double DGam_star(double, double, double, double, double, double) C
double Mfluxsqr(double, double, double, double, double, double, double, double, double, double, double, double) C
double compute_max(double, double *, double, double *, int) C
double compute_min(double, double *, double, double *, int) C
Observing I/O and Memory Activity
Library interposition/wrapping: tau_exec

• TAU provides a wealth of options to measure the performance of an application
• Need to simplify TAU usage to easily evaluate performance properties, including I/O, memory, and communication
• Designed a new tool (tau_exec) that leverages runtime instrumentation by pre-loading measurement libraries
• Works on dynamic executables (default under Linux)
• Substitutes I/O, MPI, and memory allocation/deallocation routines with instrumented calls
  – Interval events (e.g., time spent in write())
  – Atomic events (e.g., how much memory was allocated)
• Measure I/O and memory usage
Issues

• Heap memory usage reported by the mallinfo() call is not 64-bit clean.
  – 32 bit counters in Linux roll over when > 4GB memory is used
  – We keep track of heap memory usage in 64 bit counters inside TAU

• Compensation of perturbation introduced by tool
  – Only show what application uses
  – Create guards for TAU calls to not track I/O and memory allocations/de-allocations performed inside TAU

• Provide broad POSIX I/O and memory coverage
TAU Execution Command (tau_exec)

- **Uninstrumented execution**
  - `% mpirun -np 256 ./a.out`

- **Track MPI performance**
  - `% mpirun -np 256 tau_exec ./a.out`

- **Track I/O and MPI performance (MPI enabled by default)**
  - `% mpirun -np 256 tau_exec -io ./a.out`

- **Track memory operations**
  - `% setenv TAU_TRACK_MEMORYLeaks 1`
  - `% mpirun -np 256 tau_exec -memory ./a.out`

- **Track I/O performance and memory operations**
  - `% mpirun -np 256 tau_exec -io -memory ./a.out`

- **Track GPGPU operations**
  - `% mpirun -np 256 tau_exec -cuda ./a.out`

- **Collect events from callstack sampling**
  - `% mpirun -np 256 tau_exec -ebs ./a.out`
## I/O Calls Supported

<table>
<thead>
<tr>
<th>Unbuffered I/O</th>
<th>Buffered I/O</th>
<th>Communication</th>
<th>Control</th>
<th>Asynchronous I/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>open</td>
<td>fopen</td>
<td>socket</td>
<td>fcntl</td>
<td>aio_read</td>
</tr>
<tr>
<td>open64</td>
<td>fopen64</td>
<td>pipe</td>
<td>rewind</td>
<td>aio_write</td>
</tr>
<tr>
<td>close</td>
<td>fdopen</td>
<td>socketpair</td>
<td>lseek</td>
<td>aio_suspend</td>
</tr>
<tr>
<td>read</td>
<td>freopen</td>
<td>bind</td>
<td>lseek64</td>
<td>aio_cancel</td>
</tr>
<tr>
<td>write</td>
<td>fclose</td>
<td>accept</td>
<td>fseek</td>
<td>aio_return</td>
</tr>
<tr>
<td>readv</td>
<td>fprintf</td>
<td>connect</td>
<td>dup</td>
<td>lio_listio</td>
</tr>
<tr>
<td>writev</td>
<td>fscanf</td>
<td>recv</td>
<td>dup2</td>
<td></td>
</tr>
<tr>
<td>creat</td>
<td>fwrite</td>
<td>send</td>
<td>mkstep</td>
<td></td>
</tr>
<tr>
<td>creat64</td>
<td>fread</td>
<td>sendto</td>
<td>tmpfile</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>recvfrom</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pclose</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Tracking I/O in Each File

### TAU: ParaProf: Context Events for thread n, c, t, 1, 0, 0 - IOR_mana_iotreads_posix.ppp

<table>
<thead>
<tr>
<th>Name</th>
<th>Total</th>
<th>NumSamples</th>
<th>MinValue</th>
<th>MaxValue</th>
<th>MeanValue</th>
<th>StdDev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bytes Read</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bytes Read &lt;file=/opt/ompi/tm/intel/1.4/etc/ompi-mca-params.conf&gt;</td>
<td>28,024</td>
<td>32</td>
<td>8,192</td>
<td>4</td>
<td>625.75</td>
<td>2,014.699</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/opt/ompi/tm/intel/1.4/share/ompi/mhl-btl-openib.txt&gt;</td>
<td>8,192</td>
<td>1</td>
<td>8,192</td>
<td>8,192</td>
<td>8,192</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/opt/ompi/tm/intel/1.4/share/ompi/mca-btl-openib-device-params.ini&gt;</td>
<td>8,727</td>
<td>2</td>
<td>8,192</td>
<td>535</td>
<td>4,363.5</td>
<td>3,828.5</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/class/infiniband/mthca0/node_type&gt;</td>
<td>8</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/class/infiniband/mthca0/ports/1/gids/0&gt;</td>
<td>41</td>
<td>1</td>
<td>41</td>
<td>41</td>
<td>41</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/class/infiniband/verbs/abi_version&gt;</td>
<td>8</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/class/infiniband/verbs/overlbs/0/abi_version&gt;</td>
<td>8</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/class/infiniband/verbs/overlbs/0/device/device&gt;</td>
<td>24</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/class/infiniband/verbs/overlbs/0/vendor&gt;</td>
<td>24</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/class/infiniband/verbs/overlbs/0/ibdev&gt;</td>
<td>64</td>
<td>1</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu0/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu0/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu1/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu1/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu2/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu2/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu3/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu3/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu4/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu4/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu5/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu5/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu6/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu6/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu7/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu7/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu8/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu8/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu9/topology/core_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu9/topology/physical_package_id&gt;</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu0(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu0(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu1(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu1(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu2(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu2(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu3(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu3(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu4(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu4(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu5(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu5(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu6(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu6(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu7(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu7(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu8(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu8(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu9(topology/core_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/sys/devices/system/cpu/cpu9(topology/physical_package_id)</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Read &lt;file=/pipe&gt;</td>
<td>2,932,118</td>
<td>32</td>
<td>1,170,286</td>
<td>0.001</td>
<td>91,629</td>
<td>265,282</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/opt/ompi/tm/intel/1.4/etc/ompi-mca-params.conf&gt;</td>
<td>352,444</td>
<td>1</td>
<td>352,444</td>
<td>352,444</td>
<td>352,444</td>
<td>0</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/opt/ompi/tm/intel/1.4/share/ompi/mhl-btl-openib.txt&gt;</td>
<td>1,170,286</td>
<td>2</td>
<td>1,170,286</td>
<td>1,170,286</td>
<td>1,170,286</td>
<td>0</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/opt/ompi/tm/intel/1.4/share/ompi/mca-btl-openib-device-params.ini&gt;</td>
<td>1,291,5</td>
<td>2</td>
<td>1,024</td>
<td>267.5</td>
<td>6457.5</td>
<td>378.25</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/sys/class/infiniband/mthca0/node_type&gt;</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/sys/class/infiniband/mthca0/ports/1/gids/0&gt;</td>
<td>0.304</td>
<td>1</td>
<td>0.304</td>
<td>0.304</td>
<td>0.304</td>
<td>0</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/sys/class/infiniband/verbs/abi_version&gt;</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/sys/class/infiniband/verbs/overlbs/0/abi_version&gt;</td>
<td>8</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/sys/class/infiniband/verbs/overlbs/0/device/device&gt;</td>
<td>16</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>5,133</td>
<td>1,886</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/sys/class/infiniband/verbs/overlbs/0/device/vendor&gt;</td>
<td>20</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>6,667</td>
<td>1,886</td>
</tr>
<tr>
<td>READ Bandwidth (MB/s) &lt;file=/sys/class/infiniband/verbs/overlbs/0/ibdev&gt;</td>
<td>32</td>
<td>1</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>0</td>
</tr>
</tbody>
</table>
Time Spent in POSIX I/O write()

![Graph showing time spent in POSIX I/O write()](image-url)
Volume of I/O by File, Memory

<table>
<thead>
<tr>
<th>Name</th>
<th>Total</th>
<th>MeanValue</th>
<th>NumSamples</th>
<th>MinValue</th>
<th>MaxValue</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAU application</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>read()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fopen64()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fclose()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OurMain()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>malloc size</td>
<td>25,235</td>
<td>1,097.174</td>
<td>23</td>
<td>11</td>
<td>12,032</td>
<td>2,851.143</td>
</tr>
<tr>
<td>free size</td>
<td>22,707</td>
<td>1,746.592</td>
<td>13</td>
<td>11</td>
<td>12,032</td>
<td>3,660.642</td>
</tr>
<tr>
<td>OurMain [twrapper.py[3]]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>read()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>malloc size</td>
<td>3,877</td>
<td>323.083</td>
<td>12</td>
<td>32</td>
<td>981</td>
<td>252.72</td>
</tr>
<tr>
<td>free size</td>
<td>1,536</td>
<td>219.429</td>
<td>7</td>
<td>32</td>
<td>464</td>
<td>148.122</td>
</tr>
<tr>
<td>fopen64()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fclose()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;module&gt; [( obe.py[8])</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>writeRestartData [(samarcinterface.py[145])</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>samarcWriteRestartData</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>write()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE Bandwidth (MB/s) &lt;file=&quot;samarc/restore.00002/nodes.00004/proc.00001&quot;&gt;</td>
<td>74.565</td>
<td>117</td>
<td>0</td>
<td>2,156,889</td>
<td>246,386</td>
<td></td>
</tr>
<tr>
<td>WRITE Bandwidth (MB/s) &lt;file=&quot;samarc/restore.00001/nodes.00004/proc.00001&quot;&gt;</td>
<td>77.594</td>
<td>117</td>
<td>0</td>
<td>1,941.2</td>
<td>228,366</td>
<td></td>
</tr>
<tr>
<td>WRITE Bandwidth (MB/s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bytes Written &lt;file=&quot;samarc/restore.00002/nodes.00004/proc.00001&quot;&gt;</td>
<td>2,097,552</td>
<td>17,927,795</td>
<td>117</td>
<td>1</td>
<td>1,048,576</td>
<td>133,362,946</td>
</tr>
<tr>
<td>Bytes Written &lt;file=&quot;samarc/restore.00001/nodes.00004/proc.00001&quot;&gt;</td>
<td>2,097,552</td>
<td>17,927,795</td>
<td>117</td>
<td>1</td>
<td>1,048,576</td>
<td>133,362,946</td>
</tr>
<tr>
<td>Bytes Written</td>
<td>4,195,104</td>
<td>17,927,795</td>
<td>234</td>
<td>1</td>
<td>1,048,576</td>
<td>133,362,946</td>
</tr>
<tr>
<td>open64()</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Bytes Written

User Event Window: samarc_obe_4p_lomem_cp.ppk
Name: Bytes Written
Value Type: Mean Value

19580.623  5842.297
3560.416

Std. Dev.
Mean
n, c, t 0, 0, 0
n, c, t 0, 0, 2
n, c, t 1, 0, 0
n, c, t 1, 0, 2
n, c, t 2, 0, 0
n, c, t 2, 0, 2
n, c, t 3, 0, 0
n, c, t 3, 0, 2

User Event Window: samarc_obe_4p_lomem_cp.ppk
Name: Bytes Written
Value Type: Number of Samples

1552  451.922
266.5

Std. Dev.
Mean
n, c, t 0, 0, 0
n, c, t 0, 0, 2
n, c, t 1, 0, 0
n, c, t 1, 0, 2
n, c, t 2, 0, 0
n, c, t 2, 0, 2
n, c, t 3, 0, 0
n, c, t 3, 0, 2

User Event Window: samarc_obe_4p_lomem_cp.ppk
Name: Bytes Written
Value Type: Max Value

1048576  494302.524
349526.667

Std. Dev.
Mean
n, c, t 0, 0, 0
n, c, t 0, 0, 2
n, c, t 1, 0, 0
n, c, t 1, 0, 2
n, c, t 2, 0, 0
n, c, t 2, 0, 2
n, c, t 3, 0, 0
n, c, t 3, 0, 2
# Memory Leaks in MPI

<table>
<thead>
<tr>
<th>Event Description</th>
<th>Total</th>
<th>Mean Value</th>
<th>Num Samples</th>
<th>Max Value</th>
<th>Min Value</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>malloc size</td>
<td>5,013,902</td>
<td>65,972,395</td>
<td>76</td>
<td>5,000,000</td>
<td>569,732,815</td>
<td>0</td>
</tr>
<tr>
<td>MEMORY LEAK!</td>
<td>5,000,264</td>
<td>500,026.4</td>
<td>10</td>
<td>5,000,000</td>
<td>1,499,991.2</td>
<td>0</td>
</tr>
</tbody>
</table>
Generating event traces
Tracing Analysis and Visualization

<table>
<thead>
<tr>
<th></th>
<th>master</th>
<th>worker</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>master</td>
<td>worker</td>
</tr>
<tr>
<td>2</td>
<td>master</td>
<td>worker</td>
</tr>
<tr>
<td>3</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>58</td>
<td>A</td>
<td>ENTER</td>
</tr>
<tr>
<td>60</td>
<td>B</td>
<td>ENTER</td>
</tr>
<tr>
<td>62</td>
<td>A</td>
<td>SEND</td>
</tr>
<tr>
<td>64</td>
<td>A</td>
<td>EXIT</td>
</tr>
<tr>
<td>68</td>
<td>B</td>
<td>RECV</td>
</tr>
<tr>
<td>69</td>
<td>B</td>
<td>EXIT</td>
</tr>
</tbody>
</table>

![Diagram of trace analysis and visualization]
Profiling / Tracing Comparison

- Profiling
  - ☑ Finite, bounded performance data size
  - ☑ Applicable to both direct and indirect methods
  - ☑ Loses time dimension (not entirely)
  - ☹ Lacks ability to fully describe process interaction

- Tracing
  - ☑ Temporal and spatial dimension to performance data
  - ☑ Capture parallel dynamics and process interaction
  - ☑ Some inconsistencies with indirect methods
  - ☹ Unbounded performance data size (large)
  - ☹ Complex event buffering and clock synchronization
Trace Formats

• Different tools produce different formats
  – Differ by event types supported
  – Differ by ASCII and binary representations
    – Vampir Trace Format (VTF)
    – KOJAK/Scalasca (EPILOG)
    – Jumpshot (SLOG-2)
    – Paraver

• Open Trace Format (OTF)
  – Supports interoperation between tracing tools
Generate a Trace File

% export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-mpi-pdt-pgi
% export PATH=$TAUROOTDIR/x86_64/bin:$PATH
% make F90=tau_f90.sh
(Or edit Makefile and change F90=tau_f90.sh)
% qsub -I -l nodes=1:ppn=8 -X
% export TAU_TRACE=1
% mpirun -np 8 ./a.out
% tau_treemerge.pl
(merges binary traces to create tau.trc and tau.edf files)
JUMPSHOT:
% tau2slog2 tau.trc tau.edf -o app.slog2
% jumpshot app.slog2
OR
VAMPIR:
% tau2otf tau.trc tau.edf app.otf -n 4 -z
(4 streams, compressed output trace)
% vampir app.otf
OR
PARAVER:
% tau_convert -paraver tau.trc tau.edf app.prv
% paraver app.prv
Jumpshot

- Developed at Argonne National Laboratory as part of the MPICH project
  - Also works with other MPI implementations
  - Jumpshot is bundled with the TAU package
- Java-based tracefile visualization tool for postmortem performance analysis of MPI programs
- Latest version is Jumpshot-4 for SLOG-2 format
  - Scalable level of detail support
  - Timeline and histogram views
  - Scrolling and zooming
  - Search/scan facility
Jumpshot
ParaVer [http://www.bsc.es/paraver]
VNG Process Timeline with PAPI Counters
Vampir Counter Timeline Showing I/O BW
Running the application, generation of performance data
# Environment Variables in TAU

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAU_TRACE</td>
<td>0</td>
<td>Setting to 1 turns on tracing</td>
</tr>
<tr>
<td>TAU_CALLPATH</td>
<td>0</td>
<td>Setting to 1 turns on callpath profiling</td>
</tr>
<tr>
<td>TAU_TRACK_MEMORY_LEAKS</td>
<td>0</td>
<td>Setting to 1 turns on leak detection (for use with tau_exec –memory)</td>
</tr>
<tr>
<td>TAU_TRACK_HEAP or TAU_TRACK_HEADROOM</td>
<td>0</td>
<td>Setting to 1 turns on tracking heap memory/headroom at routine entry &amp; exit using context events (e.g., Heap at Entry: main=&gt;foo=&gt;bar)</td>
</tr>
<tr>
<td>TAU_CALLPATH_DEPTH</td>
<td>2</td>
<td>Specifies depth of callpath. Setting to 0 generates no callpath or routine information, setting to 1 generates flat profile and context events have just parent information (e.g., Heap Entry: foo)</td>
</tr>
<tr>
<td>TAU_SAMPLING</td>
<td>1</td>
<td>Generates sample based profile</td>
</tr>
<tr>
<td>TAU_COMM_MATRIX</td>
<td>0</td>
<td>Setting to 1 generates communication matrix display using context events</td>
</tr>
<tr>
<td>TAU_THROTTLE</td>
<td>1</td>
<td>Setting to 0 turns off throttling. Enabled by default to remove instrumentation in lightweight routines that are called frequently</td>
</tr>
<tr>
<td>TAU_THROTTLE_NUMCALLS</td>
<td>100000</td>
<td>Specifies the number of calls before testing for throttling</td>
</tr>
<tr>
<td>TAU_THROTTLE_PERCALL</td>
<td>10</td>
<td>Specifies value in microseconds. Throttle a routine if it is called over 100000 times and takes less than 10 usec of inclusive time per call</td>
</tr>
<tr>
<td>TAU_COMPENSATE</td>
<td>0</td>
<td>Setting to 1 enables runtime compensation of instrumentation overhead</td>
</tr>
<tr>
<td>TAU_PROFILE_FORMAT</td>
<td>Profile</td>
<td>Setting to “merged” generates a single file. “snapshot” generates xml format</td>
</tr>
<tr>
<td>TAU_METRICS</td>
<td>TIME</td>
<td>Setting to a comma separated list generates other metrics. (e.g., TIME:linuxtimers:PAPI_FP_OPS:PAPI_NATIVE_&lt;event&gt;)</td>
</tr>
</tbody>
</table>
Generate a Callpath Profile

% export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-mpi-pdt
% export PATH=$TAUROOTDIR/x86_64/bin:$PATH
% make F90=tau_f90.sh
(Or edit Makefile and change F90=tau_f90.sh)

% qsub -I -l nodes=1:ppn=8 -X
% export TAU_CALLPATH=1
% export TAU_CALLPATH_DEPTH=100

% mpirun -np 8 ./a.out
% paraprof --pack app.ppk
    Move the app.ppk file to your desktop.
% paraprof app.ppk
(Windows -> Thread -> Call Graph)
Usage Scenarios: Generating Callpath Profile

- Callpath profile for a given callpath depth:
Callpath Profile

- Generates program callgraph
Communication Matrix

% export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-mpi-pdt
% export PATH=$TAUROOTDIR/x86_64/bin:$PATH
% make F90=tau_f90.sh
(Or edit Makefile and change F90=tau_f90.sh)

% qsub -I -l nodes=1:ppn=8 -X
% export TAU_COMM_MATRIX=1
% mpirun -np 8 ./a.out (setting the environment variables)

% paraprof
(Windows -> Communication Matrix)
ParaProf: Communication Matrix Display
Analyzing performance data with ParaProf, PerfExplorer
TAU Performance System Architecture

Profile Data Management (PerfDMF)
- profile translators
- Metadata (XML)
- profile database

Profile Analysis (ParaProf)

Profile Data Mining (PerfExplorer)

Trace Data Management
- trace translators
- trace storage

Trace Visualizers
- Vampir
- JumpShot
- Paraver

Trace Analyzers
- Expert
- ProfileGen
- Vampir Server

Analysis
PerfDMF: Performance Data Mgmt. Framework
ParaProf Main Window

% paraprof matmult.ppk

Click left mouse button

Click right mouse button
ParaProf: Manager Window

ParaProf Manager

File  Options  Help

Applications

- Standard Applications
  - Default App
    - Default Exp
      - 16pAIX200iter/s3d/taudata/rs/sameer/Users/

- Runtime Applications

- DB Applications

- AORS2D

- Basic run-time profiling for Socorro

- Heap memory management for Socorro

- hydroshock

- MFX

- S3D

- AIX

  - 16pAIX10iter/s3d/taudata/rs/sameer/Users/

  - 16pAIX200iter/s3d/taudata/rs/sameer/Users/

  - Time

  - 16pAIXcall200iter/s3d/taudata/rs/sameer/Users/

  - Time

Field  Value

Name  16pAIXcall200iter/s3d/taudata/rs/sameer/Users/
Application ID  8
Experiment ID  16
Trial ID  34
time
problem_definition  nx_g=400, ny_g=400, npx=1, npy=4, npz=1
node_count  16
contexts_per_node  1
threads_per_context  1
userdata
i_time_end=200, i_time_save=200, TAU_CALLPATH_DEPTH=2

Load Trial

Trial Type  Tau profiles

Select Directory  /Users/sameer/rs/taudata/s3d

Cancel  Ok
Comparing Effects of Multi-Core Processors

- Metric: PAPI_RES_STL
  Value: Exclusive
  Units: counts

- C:\iter.350x350.4096pes.sn.loops.BARRIER.ppk - Mean
- C:\iter.350x350.2048pes.dc.loops.BARRIER.ppk - Mean

AORSA2D
- magnetized plasma simulation
- Automatic loop level instrumentation
- Blue is single node
- Red is dual core
- Cray XT3 (4K cores)
ParaProf: Mflops Sorted by Exclusive Time

<table>
<thead>
<tr>
<th>Metric: PAPI_FP_INS / LINUX_TIMERS</th>
<th>Value: Exclusive</th>
<th>Units: Derived metric shown in microseconds format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loop: CHEMKIN_M:REACTION_RATE_VEC</td>
<td>[chemkin_m.pp.f90]</td>
<td>457.3</td>
</tr>
<tr>
<td>Loop: TRANSPORT_M:COMPUTECOEFFICIENTS</td>
<td>[mixavg_transport_m.pp.f90]</td>
<td>492.5</td>
</tr>
<tr>
<td>MPI Wait()</td>
<td></td>
<td>0.014</td>
</tr>
<tr>
<td>Loop: TRANSPORT_M:COMPUTESPECIESDIFFFLUX</td>
<td>[mixavg_transport_m.pp.f90]</td>
<td>630.5</td>
</tr>
<tr>
<td>MPI_Isend()</td>
<td></td>
<td>0.021</td>
</tr>
<tr>
<td>Loop: RHSSF</td>
<td>[rhssf.pp.f90]</td>
<td>209.3</td>
</tr>
<tr>
<td>Loop: INTEGRATE</td>
<td>[integrate_erlk.pp.f90]</td>
<td>73.3</td>
</tr>
<tr>
<td>MPI_Comm_split()</td>
<td></td>
<td>0.06</td>
</tr>
<tr>
<td>MPI_Barrier()</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>RHSSF</td>
<td></td>
<td>148.608</td>
</tr>
<tr>
<td>Loop: TRANSPORT_M:COMPUTEHEATFLUX</td>
<td>[mixavg_transport_m.pp.f90]</td>
<td>782.5</td>
</tr>
<tr>
<td>THERMCHM_M:CALC_TEMP</td>
<td>[thermchem_m.pp.f90]</td>
<td>175.5</td>
</tr>
<tr>
<td>RHSSF</td>
<td>[rhssf.pp.f90]</td>
<td>537.3</td>
</tr>
<tr>
<td>RHSSF</td>
<td>[rhssf.pp.f90]</td>
<td>545.3</td>
</tr>
<tr>
<td>RHSSF</td>
<td>[rhssf.pp.f90]</td>
<td>515.3</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_X_CALC</td>
<td>[derivative_x.pp.f90]</td>
<td>432.10</td>
</tr>
<tr>
<td>DERIVATIVE_Y_COMM</td>
<td></td>
<td>2.177</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_Y_CALC</td>
<td>[derivative_y.pp.f90]</td>
<td>431.0</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_Z_CALC</td>
<td>[derivative_z.pp.f90]</td>
<td>435.10</td>
</tr>
<tr>
<td>VARIABLES_M:GET_MASS_FRAC</td>
<td>[variables_m.pp.f90]</td>
<td>96.3</td>
</tr>
<tr>
<td>Loop: THERMCHM_M:CALC_INV_AVG_MOL_WT</td>
<td>[thermchem_m.pp.f90]</td>
<td>127.5</td>
</tr>
<tr>
<td>MPI_Irecv()</td>
<td></td>
<td>0.002</td>
</tr>
<tr>
<td>DERIVATIVE_Y_COMM</td>
<td></td>
<td>3.778</td>
</tr>
<tr>
<td>Loop: THERMCHM_M:CALC_SPECENTH_ALLPTS</td>
<td>[thermchem_m.pp.f90]</td>
<td>506.3</td>
</tr>
<tr>
<td>INTEGRATE</td>
<td>1.294</td>
<td></td>
</tr>
<tr>
<td>S3D</td>
<td></td>
<td>7.111</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_X_CALC</td>
<td>[derivative_x.pp.f90]</td>
<td>490.13</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_Y_CALC</td>
<td>[derivative_y.pp.f90]</td>
<td>461.13</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_Y_CALC</td>
<td>[derivative_y.pp.f90]</td>
<td>488.13</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_Y_CALC</td>
<td>[derivative_y.pp.f90]</td>
<td>459.13</td>
</tr>
<tr>
<td>Loop: TRANSPORT_M:COMPUTESTRESSTENSOR</td>
<td>[mixavg_transport_m.pp.f90]</td>
<td>711.5</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_Z_CALC</td>
<td>[derivative_z.pp.f90]</td>
<td>464.13</td>
</tr>
<tr>
<td>TRANSPORT_M:COMPUTESPESIESDIFFFLUX</td>
<td></td>
<td>3.328</td>
</tr>
<tr>
<td>Loop: DERIVATIVE_Z_CALC</td>
<td>[derivative_z.pp.f90]</td>
<td>493.13</td>
</tr>
<tr>
<td>RK_M::CONTROLLER</td>
<td></td>
<td>84.649</td>
</tr>
<tr>
<td>DERIVATIVE_X_CALC</td>
<td>30.583</td>
<td></td>
</tr>
<tr>
<td>DERIVATIVE_Y_CALC</td>
<td>33.374</td>
<td></td>
</tr>
<tr>
<td>DERIVATIVE_Z_CALC</td>
<td>33.405</td>
<td></td>
</tr>
<tr>
<td>THERMCHM_M::CALC_INV_AVG_MOL_WT</td>
<td>61.032</td>
<td></td>
</tr>
<tr>
<td>GET_VELOCITY_VEC</td>
<td>95.032</td>
<td></td>
</tr>
<tr>
<td>Loop: TRANSPORT_M:COMPUTECOEFFICIENTS</td>
<td>[mixavg_transport_m.pp.f90]</td>
<td>527.5</td>
</tr>
<tr>
<td>Loop: FILTER::FILTER</td>
<td>[filter_m.pp.f90]</td>
<td>1232.5</td>
</tr>
<tr>
<td>THERMCHM_M::CALC_TEMP</td>
<td></td>
<td>196.839</td>
</tr>
<tr>
<td>TRANSPORT_M::COMPUTESCALARADGRADIENT</td>
<td></td>
<td>293.594</td>
</tr>
<tr>
<td>COMPUTE::CALCULER</td>
<td></td>
<td>141.127</td>
</tr>
<tr>
<td>COMPUTE::CALCULER</td>
<td></td>
<td>24.465</td>
</tr>
<tr>
<td>Loop: INTEGRATE</td>
<td>[integrate.erlk.pp.f90]</td>
<td>108.3</td>
</tr>
<tr>
<td>VARIABLES_M::GET_MASS_FRAC</td>
<td></td>
<td>0.245</td>
</tr>
</tbody>
</table>
Scalable Visualization: ParaProf (128k cores)
Scatter Plot: ParaProf (128k cores)
ParaProf: Topology View: MPI_Send on BG/P
ParaProf: Topology View: BG/Q Core Layout
ParaProf – Callgraph Zoomed (Flash)
<table>
<thead>
<tr>
<th>Name</th>
<th>Inclusive Time</th>
<th>Exclusive Time</th>
<th>Calls</th>
<th>Child Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSI</td>
<td>5,223.564</td>
<td>0.098</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>SPECMOD::INIT_SPEC_VARS</td>
<td>0.26</td>
<td>0.26</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>MPI_Init()</td>
<td>0.056</td>
<td>0.054</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>GSISUB</td>
<td>5,223.094</td>
<td>0.012</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>RADINFO::RADINFO_READ</td>
<td>0.103</td>
<td>0.101</td>
<td>1</td>
<td>1,196</td>
</tr>
<tr>
<td>PCPINFO::PCPINFO_READ</td>
<td>0.042</td>
<td>0.042</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>GL350I</td>
<td>5,212.171</td>
<td>0.024</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>MPI_Finalize()</td>
<td>1.004</td>
<td>1.004</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>OBS_PARA</td>
<td>3.635</td>
<td>0.181</td>
<td>1</td>
<td>56</td>
</tr>
<tr>
<td>JFUNC::CREATE_JFUNC</td>
<td>0.142</td>
<td>0.142</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>GUESS/Grids::create_ges_bias_grids</td>
<td>0.059</td>
<td>0.059</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>READ_GUESS</td>
<td>1,406.412</td>
<td>0.023</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>READ_OBS</td>
<td>3,770.188</td>
<td>0.016</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>MPI_Allreduce()</td>
<td>3,725.802</td>
<td>3,725.802</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>READ_BUFRTOV</td>
<td>44.369</td>
<td>0.254</td>
<td>1</td>
<td>871,535</td>
</tr>
<tr>
<td>SATTHIN::MAKEGVALS</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>W3FS2I</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>BINARY_FILE_UTILITE::OPEN_BINARY_FILE</td>
<td>0.025</td>
<td>0.012</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>INITIALIZE::INITIALIZE_RTM</td>
<td>0.099</td>
<td>0.001</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>GUESS/Grids::create_sfc_grids</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>M_FVANAGRID::ALLGETLIST_</td>
<td>30.582</td>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>ERROR_HANDLER::DISPLAY_MESSAGE</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>JFUNC::SET_POINTER</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>OZINFO::OZINFO_READ</td>
<td>0.016</td>
<td>0.016</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>DETER_SUBDOMAIN</td>
<td>0.008</td>
<td>0.008</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>GRIDMOD::CREATE_MAPPING</td>
<td>0.005</td>
<td>0.005</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>INIT_COMMVARs</td>
<td>0.004</td>
<td>0.004</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>M_FVANAGRID::ALLGETLIST_</td>
<td>10.711</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>GRIDMOD::CREATE_GRID_VARS</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
ParaProf - Callpath Thread Relations Window

<table>
<thead>
<tr>
<th>Parent Routine</th>
<th>0.023</th>
<th>0.023</th>
<th>2/430</th>
<th>COMPUTE_DERIVED [55]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.003</td>
<td>0.003</td>
<td>1/430</td>
<td>INSTALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>0.033</td>
<td>0.033</td>
<td>104/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>0.33</td>
<td>0.33</td>
<td>104/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>1.639</td>
<td>1.639</td>
<td>1/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>3725.802</td>
<td>3725.802</td>
<td>3/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>214.294</td>
<td>214.294</td>
<td>6/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>20.069</td>
<td>20.069</td>
<td>208/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>196.148</td>
<td>196.148</td>
<td>430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Routine</th>
<th>2.6E-4</th>
<th>30.582</th>
<th>1/430</th>
<th>GLBSQ1 [93]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.007</td>
<td>0.036</td>
<td>1/430</td>
<td>GLSUS [1690]</td>
</tr>
<tr>
<td></td>
<td>2.7E-4</td>
<td>10.711</td>
<td>1/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>31.273</td>
<td>1347.037</td>
<td>3/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>0.412</td>
<td>0.412</td>
<td>1/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>70.198</td>
<td>1406.389</td>
<td>4/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>0.952</td>
<td>0.952</td>
<td>3/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
<tr>
<td></td>
<td>86.937</td>
<td>95.933</td>
<td>3/430</td>
<td>INFALLMOD::INSTALL [1708]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Children</th>
<th>196.148</th>
<th>15</th>
<th>INFALLMOD::INSTALL [1708]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.2E-5</td>
<td>6.2E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>4.6E-5</td>
<td>4.6E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>3.949</td>
<td>3.949</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>0.017</td>
<td>0.017</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>2.0E-4</td>
<td>2.0E-4</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>8.0E-5</td>
<td>8.0E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>5.7E-5</td>
<td>5.7E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>4.9E-5</td>
<td>4.9E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>0.042</td>
<td>0.042</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>0.082</td>
<td>0.816</td>
<td>3/3</td>
</tr>
<tr>
<td></td>
<td>1.4E-4</td>
<td>3.1E-4</td>
<td>3/3</td>
</tr>
<tr>
<td></td>
<td>4.2E-5</td>
<td>4.2E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>8.0E-5</td>
<td>8.0E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>0.169</td>
<td>0.169</td>
<td>3/3</td>
</tr>
<tr>
<td></td>
<td>3.3E-4</td>
<td>3.3E-4</td>
<td>3/3</td>
</tr>
<tr>
<td></td>
<td>9.1E-5</td>
<td>9.1E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>2.2E-4</td>
<td>2.2E-4</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>6.6E-5</td>
<td>6.6E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>5.8E-5</td>
<td>5.8E-5</td>
<td>1/1</td>
</tr>
<tr>
<td></td>
<td>0.003</td>
<td>0.003</td>
<td>1/430</td>
</tr>
<tr>
<td></td>
<td>0.017</td>
<td>0.017</td>
<td>68/116</td>
</tr>
<tr>
<td></td>
<td>0.004</td>
<td>0.004</td>
<td>297/409</td>
</tr>
</tbody>
</table>
ParaProf – Histogram View (Miranda)

**MPI_Alltoall()**

- **8k processors**
- **16k processors**

**MPI_Barrier()**
Using Performance Database (PerfDMF)

- **Configure PerfDMF (Done by each user)**
  
  ```
  % perfdfm_configure --create-default
  ```
  - Choose derby, PostgreSQL, MySQL, Oracle or DB2
  - Hostname
  - Username
  - Password
  - Say yes to downloading required drivers (we are not allowed to distribute these)
  - Stores parameters in your ~/.ParaProf/perfdmf.cfg file

- **Configure PerfExplorer (Done by each user)**
  
  ```
  % perfexplorer_configure
  ```

- **Execute PerfExplorer**
  
  ```
  % perfexplorer
  ```
PerfDMF and the TAU Portal

- Development of the TAU portal
  - Common repository for collaborative data sharing
  - Profile uploading, downloading, user management
  - Paraprof, PerfExplorer can be launched from the portal using Java Web Start (no TAU installation required)

- Portal URL
  http://tau.nic.uoregon.edu
Performance Data Mining (PerfExplorer)

- Performance knowledge discovery framework
  - Data mining analysis applied to parallel performance data
    - comparative, clustering, correlation, dimension reduction, ...
  - Use the existing TAU infrastructure
    - TAU performance profiles, PerfDMF
  - Client-server based system architecture

- Technology integration
  - Java API and toolkit for portability
  - PerfDMF
  - R-project/Omegahat, Octave/Matlab statistical analysis
  - WEKA data mining package
  - JFreeChart for visualization, vector output (EPS, SVG)
PerfExplorer - Cluster Analysis

• Performance data represented as vectors - each dimension is the cumulative time for an event

• $k$-means: $k$ random centers are selected and instances are grouped with the "closest" (Euclidean) center

• New centers are calculated and the process repeated until stabilization or max iterations

• Dimension reduction necessary for meaningful results

• Virtual topology, summaries constructed
PerfExplorer - Cluster Analysis (sPPM)
PerfExplorer - Correlation Analysis (Flash)

- Describes strength and direction of a linear relationship between two variables (events) in the data.
-0.995 indicates strong, negative relationship

As CALC_CUT_BLOCK_CONTRIBUTIONS() increases in execution time, MPI_Barrier() decreases
PerfExplorer - Comparative Analysis

• Relative speedup, efficiency
  – total runtime, by event, one event, by phase

• Breakdown of total runtime

• Group fraction of total runtime

• Correlating events to total runtime

• Timesteps per second

• Performance Evaluation Research Center (PERC)
  – PERC tools study (led by ORNL, Pat Worley)
  – In-depth performance analysis of select applications
  – Evaluation performance analysis requirements
  – Test tool functionality and ease of use
PerfExplorer - Interface

Select experiments and trials of interest

Data organized in application, experiment, trial structure (will allow arbitrary in future)

Experiment metadata
PerfExplorer - Interface

Select analysis
PerfExplorer - Relative Efficiency Plots
PerfExplorer - Relative Efficiency by Routine
PerfExplorer - Relative Speedup
PerfExplorer - Timesteps Per Second
Usage Scenarios: Evaluate Scalability

- **Goal:** How does my application scale? What bottlenecks occur at what core counts?
- Load profiles in PerfDMF database and examine with PerfExplorer

![Graph showing relative speedup for S3D (Jaguar, ORNL): Harness Scaling Study GET_TIME_OF_DAY](image)
Usage Scenarios: Evaluate Scalability
Performance Regression Testing

FACETS Bassi Regression: 32 Procs (events above 2%)

- int main(int, char **)
- std::vector< double, std::allocator< double > >
- FcCoreCellUpdate...
- void FcTmCoreFluxCalc::computeFluxes()
- MPI_Recv()
- double FcDataAssimilator::getValue(const std::string&, const...)
- MPI_Init()
- FcHdf5Tmp< DATATYPE >::writeDataSet
- void FcDataAssimilatorUfiles::parseUfiles(const std::vector<...)
- void FcUpdaterComponent::dumpToFile(const std::string&, const...)
- other
Evaluate Scalability using PerfExplorer Charts

% export TAU_MAKEFILE=$TAU_ROOT
  /lib/Makefile.tau-mpi-pdt
% export PATH=$TAUROOTDIR/x86_64/bin:$PATH
% make F90=tau_f90.sh
(Or edit Makefile and change F90=tau_f90.sh)
% mpirun -np 1 ./a.out
% paraprof --pack 1p.ppk
% mpirun -np 2 ./a.out ...
% paraprof --pack 2p.ppk ... and so on.

On your client:
% perfdfm_configure --create-default
(Chooses derby, blank user/passwd, yes to save passwd, defaults)
% perfexplorer_configure
(Yes to load schema, defaults)
% paraprof
(load each trial: DB -> Add Trial -> Type (Paraprof Packed Profile) -> OK) OR use
  perfdfm_loadtrial
Then,
% perfexplorer
(Select experiment, Menu: Charts -> Speedup)
PAPI hardware counters
Hardware Counters

Hardware performance counters available on most modern microprocessors can provide insight into:

1. Whole program timing
2. Cache behaviors
3. Branch behaviors
4. Memory and resource access patterns
5. Pipeline stalls
6. Floating point efficiency
7. Instructions per cycle

Hardware counter information can be obtained with:

1. Subroutine or basic block resolution
2. Process or thread attribution
What’s PAPI?

• Open Source software from U. Tennessee, Knoxville
• http://icl.cs.utk.edu/papi
• Middleware to provide a consistent programming interface for the performance counter hardware found in most major microprocessors.
• Countable events are defined in two ways:
  – Platform-neutral preset events
  – Platform-dependent native events
• Presets can be derived from multiple native events
• All events are referenced by name and collected in EventSets
PAPI Utilities: papi_avail

$ utils/papi_avail -h
Usage: utils/papi_avail [options]
Options:

General command options:
   -a, --avail   Display only available preset events
   -d, --detail  Display detailed information about all preset events
   -e EVENTNAME  Display detail information about specified preset or native event
   -h, --help    Print this help message

Event filtering options:
   --br          Display branch related PAPI preset events
   --cache       Display cache related PAPI preset events
   --cnd         Display conditional PAPI preset events
   --fp          Display Floating Point related PAPI preset events
   --ins         Display instruction related PAPI preset events
   --idl        Display Stalled or Idle PAPI preset events
   --l1          Display level 1 cache related PAPI preset events
   --l2          Display level 2 cache related PAPI preset events
   --l3          Display level 3 cache related PAPI preset events
   --mem         Display memory related PAPI preset events
   --msc        Display miscellaneous PAPI preset events
   --tlb        Display Translation Lookaside Buffer PAPI preset events

This program provides information about PAPI preset and native events.
PAPI preset event filters can be combined in a logical OR.
$ utils/papi_avail
Available events and hardware information.
--------------------------------------------------------------------------------

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAPI Version</td>
<td>4.0.0.0</td>
</tr>
<tr>
<td>Vendor string and code</td>
<td>GenuineIntel (1)</td>
</tr>
<tr>
<td>Model string and code</td>
<td>Intel Core i7 (21)</td>
</tr>
<tr>
<td>CPU Revision</td>
<td>5.000000</td>
</tr>
<tr>
<td>CPUID Info</td>
<td>Family: 6 Model: 26 Stepping: 5</td>
</tr>
<tr>
<td>CPU Megahertz</td>
<td>2926.000000</td>
</tr>
<tr>
<td>CPU Clock Megahertz</td>
<td>2926</td>
</tr>
<tr>
<td>Hdw Threads per core</td>
<td>1</td>
</tr>
<tr>
<td>Cores per Socket</td>
<td>4</td>
</tr>
<tr>
<td>NUMA Nodes</td>
<td>2</td>
</tr>
<tr>
<td>CPU's per Node</td>
<td>4</td>
</tr>
<tr>
<td>Total CPU's</td>
<td>8</td>
</tr>
<tr>
<td>Number Hardware Counters</td>
<td>7</td>
</tr>
<tr>
<td>Max Multiplex Counters</td>
<td>32</td>
</tr>
</tbody>
</table>
--------------------------------------------------------------------------------

The following correspond to fields in the PAPI_event_info_t structure.

[MORE...]

[122]
The following correspond to fields in the PAPI_event_info_t structure.

<table>
<thead>
<tr>
<th>Name</th>
<th>Code</th>
<th>Avail</th>
<th>Deriv</th>
<th>Description (Note)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAPI_L1_DCM</td>
<td>0x80000000</td>
<td>No</td>
<td>No</td>
<td>Level 1 data cache misses</td>
</tr>
<tr>
<td>PAPI_L1_ICM</td>
<td>0x80000001</td>
<td>Yes</td>
<td>No</td>
<td>Level 1 instruction cache misses</td>
</tr>
<tr>
<td>PAPI_L2_DCM</td>
<td>0x80000002</td>
<td>Yes</td>
<td>Yes</td>
<td>Level 2 data cache misses</td>
</tr>
<tr>
<td>PAPI_VEC_SP</td>
<td>0x80000069</td>
<td>Yes</td>
<td>No</td>
<td>Single precision vector/SIMD instructions</td>
</tr>
<tr>
<td>PAPI_VEC_DP</td>
<td>0x8000006a</td>
<td>Yes</td>
<td>No</td>
<td>Double precision vector/SIMD instructions</td>
</tr>
</tbody>
</table>

Of 107 possible events, 34 are available, of which 9 are derived.

avail.c       PASSED
$ utils/papi_avail -e PAPI_FP_OPS

The following correspond to fields in the PAPI_event_info_t structure.

Event name: PAPI_FP_OPS
Event Code: 0x80000066
Number of Native Events: 2
Short Description: |FP operations|
Long Description: |Floating point operations|
Developer's Notes: ||
Derived Type: |DERIVED_ADD|
Postfix Processing String: ||
Native Code[0]: 0x4000801b |FP_COMP_OPS_EXE:SSE_SINGLE_PRECISION|
Number of Register Values: 2
Register[ 0]: 0x0000000f |Event Selector|
Register[ 1]: 0x00004010 |Event Code|
Native Event Description: |Floating point computational micro-ops, masks:SSE* FP single precision Uops|

Native Code[1]: 0x4000081b |FP_COMP_OPS_EXE:SSE_DOUBLE_PRECISION|
Number of Register Values: 2
Register[ 0]: 0x0000000f |Event Selector|
Register[ 1]: 0x00008010 |Event Code|
Native Event Description: |Floating point computational micro-ops, masks:SSE* FP double precision Uops|
PAPI Utilities: `papi_native_avail`

UNIX> `utils/papi_native_avail`
Available native events and hardware information.
--------------------------------------------------------------------------------

[...]

<table>
<thead>
<tr>
<th>Event Code</th>
<th>Symbol</th>
<th>Long Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x40000010</td>
<td>BR_INST_EXEC</td>
<td>Branch instructions executed</td>
</tr>
<tr>
<td>0x40000011</td>
<td>BR_INST_RETIRED</td>
<td>Retired branch instructions</td>
</tr>
<tr>
<td>40000410</td>
<td>:ANY</td>
<td>Branch instructions executed</td>
</tr>
<tr>
<td>40000810</td>
<td>:COND</td>
<td>Conditional branch instructions executed</td>
</tr>
<tr>
<td>40001010</td>
<td>:DIRECT</td>
<td>Unconditional branches executed</td>
</tr>
<tr>
<td>40002010</td>
<td>:DIRECT_NEAR_CALL</td>
<td>Unconditional call branches executed</td>
</tr>
<tr>
<td>40004010</td>
<td>:INDIRECT_NEAR_CALL</td>
<td>Indirect call branches executed</td>
</tr>
<tr>
<td>40008010</td>
<td>:INDIRECT_NON_CALL</td>
<td>Indirect non call branches executed</td>
</tr>
<tr>
<td>40010010</td>
<td>:NEAR_CALLS</td>
<td>Call branches executed</td>
</tr>
<tr>
<td>40020010</td>
<td>:NON_CALLS</td>
<td>All non call branches executed</td>
</tr>
<tr>
<td>40040010</td>
<td>:RETURN_NEAR</td>
<td>Indirect return branches executed</td>
</tr>
<tr>
<td>40080010</td>
<td>:TAKEN</td>
<td>Taken branches executed</td>
</tr>
<tr>
<td>0x40000011</td>
<td>BR_INST_RETIRE</td>
<td>Retired branch instructions</td>
</tr>
<tr>
<td>40000411</td>
<td>:ALL_BRANCHES</td>
<td>Retired branch instructions (Precise Event)</td>
</tr>
<tr>
<td>40000811</td>
<td>:CONDITIONAL</td>
<td>Retired conditional branch instructions (Precise Event)</td>
</tr>
<tr>
<td>40010011</td>
<td>:NEAR_CALL</td>
<td>Retired near call instructions (Precise Event)</td>
</tr>
</tbody>
</table>

[...]
PAPI Utilities: *papi_native_avail*

```
UNIX> utils/papi_native_avail -e DATA_CACHE_REFILLS
Available native events and hardware information.
--------------------------------------------------------------------------------
[...]
--------------------------------------------------------------------------------
The following correspond to fields in the PAPI_event_info_t structure.

Event name: DATA_CACHE_REFILLS
Event Code: 0x4000000b
Number of Register Values: 2
Description: |Data Cache Refills from L2 or System|
    Register[ 0]: 0x0000000f |Event Selector|
    Register[ 1]: 0x00000042 |Event Code|

Unit Masks:

    Mask Info: |:SYSTEM|Refill from System|
    Register[ 0]: 0x0000000f |Event Selector|
    Register[ 1]: 0x000000142 |Event Code|

    Mask Info: |:L2_SHARED|Shared-state line from L2|
    Register[ 0]: 0x0000000f |Event Selector|
    Register[ 1]: 0x000000242 |Event Code|

    Mask Info: |:L2_EXCLUSIVE|Exclusive-state line from L2|
    Register[ 0]: 0x0000000f |Event Selector|
    Register[ 1]: 0x000000442 |Event Code|
```
PAPI Utilities: *papi_event_chooser*

```bash
$ utils/papi_event_chooser PRESET PAPI_FP_OPS
Event Chooser: Available events which can be added with given events.
--------------------------------------------------------------------------------

<table>
<thead>
<tr>
<th>Name</th>
<th>Code</th>
<th>Deriv</th>
<th>Description (Note)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAPI_L1_DCM</td>
<td>0x80000000</td>
<td>No</td>
<td>Level 1 data cache misses</td>
</tr>
<tr>
<td>PAPI_L1_ICM</td>
<td>0x80000001</td>
<td>No</td>
<td>Level 1 instruction cache misses</td>
</tr>
<tr>
<td>PAPI_L2_ICM</td>
<td>0x80000003</td>
<td>No</td>
<td>Level 2 instruction cache misses</td>
</tr>
<tr>
<td>PAPI_L1_DCA</td>
<td>0x80000040</td>
<td>No</td>
<td>Level 1 data cache accesses</td>
</tr>
<tr>
<td>PAPI_L2_DCR</td>
<td>0x80000044</td>
<td>No</td>
<td>Level 2 data cache reads</td>
</tr>
<tr>
<td>PAPI_L2_DCW</td>
<td>0x80000047</td>
<td>No</td>
<td>Level 2 data cache writes</td>
</tr>
<tr>
<td>PAPI_L1_ICA</td>
<td>0x8000004c</td>
<td>No</td>
<td>Level 1 instruction cache accesses</td>
</tr>
<tr>
<td>PAPI_L2_ICA</td>
<td>0x8000004d</td>
<td>No</td>
<td>Level 2 instruction cache accesses</td>
</tr>
<tr>
<td>PAPI_L2_TCA</td>
<td>0x80000059</td>
<td>No</td>
<td>Level 2 total cache accesses</td>
</tr>
<tr>
<td>PAPI_L2_TCW</td>
<td>0x8000005f</td>
<td>No</td>
<td>Level 2 total cache writes</td>
</tr>
<tr>
<td>PAPI_FML_INS</td>
<td>0x80000061</td>
<td>No</td>
<td>Floating point multiply instructions</td>
</tr>
<tr>
<td>PAPI_FDV_INS</td>
<td>0x80000063</td>
<td>No</td>
<td>Floating point divide instructions</td>
</tr>
</tbody>
</table>
--------------------------------------------------------------------------------

Total events reported: 34
```

event_chooser.c                          PASSED
PAPI Utilities: `papi_event_chooser`

```
$ utils/papi_event_chooser PRESET PAPI_FP_OPS PAPI_L1_DCM
Event Chooser: Available events which can be added with given events.

[...

<table>
<thead>
<tr>
<th>Name</th>
<th>Code</th>
<th>Deriv</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAPI_TOT_INS</td>
<td>0x80000032</td>
<td>No</td>
<td>Instructions completed</td>
</tr>
<tr>
<td>PAPI_TOT_CYC</td>
<td>0x8000003b</td>
<td>No</td>
<td>Total cycles</td>
</tr>
</tbody>
</table>

Total events reported: 2

event_chooser.c                          PASSED
```
$ utils/papi_event_chooser NATIVE RESOURCESTALLS:LD ST X87_OPS RETIRED
INSTRUCTIONS RETIRED

[...]

UNHALTED_CORE_CYCLES  0x40000000
|count core clock cycles whenever the clock signal on the specific core is running (not
halted). Alias to event CPU_CLK_UNHALTED:CORE_P|
|Register Value[0]: 0x20003       Event Selector|
|Register Value[1]: 0x3c          Event Code|

UNHALTED_REFERENCE_CYCLES       0x40000002
|Unhalted reference cycles. Alias to event CPU_CLK_UNHALTED:REF|
|Register Value[0]: 0x40000       Event Selector|
|Register Value[1]: 0x13c         Event Code|

CPU_CLK_UNHALTED        0x40000028
|Core cycles when core is not halted|
|Register Value[0]: 0x60000       Event Selector|
|Register Value[1]: 0x3c          Event Code|

Total events reported: 3

event_chooser.c                          PASSED
Usage Scenarios: Calculate mflops in Loops

- **Goal:** What MFlops am I getting in all loops?
- **Flat profile with PAPI_FP_INS/OPS and time with loop instrumentation:**

Metric: PAPI_FP_INS / GET_TIME_OF_DAY
Value: Exclusive
Units: Derived metric shown in microseconds format
Generate a PAPI profile with 2 or more counters

% export TAU_MAKEFILE=$TAULIBDIR/Makefile.tau-papi-mpi-pdt-pgi
% export TAU_OPTIONS='--optTauSelectFile=select.tau --optVerbose'
% cat select.tau

BEGIN_INSTRUMENT_SECTION

loops routine="#"

END_INSTRUMENT_SECTION

% export PATH=$TAUROOTDIR/x86_64/bin:$PATH
% make F90=tau_f90.sh
(Or edit Makefile and change F90=tau_f90.sh)
% qsub -I -l nodes=1:ppn=8 -X
% export TAU_METRICS=TIME:PAPI_FP_INS:PAPI_L1_DCM
% mpirun -np 8 ./a.out
% paraprof --pack app.ppk

Move the app.ppk file to your desktop.
% paraprof app.ppk

Choose Options -> Show Derived Panel -> "PAPI_FP_INS", click "/", "TIME", click "Apply" choose.
# Derived Metrics in ParaProf

![ParaProf Manager](image)

## Metrics Table

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial ID</td>
<td>0</td>
</tr>
<tr>
<td>CPU Cores</td>
<td>6</td>
</tr>
<tr>
<td>CPU MHz</td>
<td>2500.093</td>
</tr>
<tr>
<td>CPU Type</td>
<td>Six-Core AMD Opteron(tm) Processor 8435</td>
</tr>
<tr>
<td>CPU Vendor</td>
<td>AuthenticAMD</td>
</tr>
<tr>
<td>CKID</td>
<td>/root/1990</td>
</tr>
<tr>
<td>Cache Size</td>
<td>512 KB</td>
</tr>
<tr>
<td>Executable</td>
<td>/root/1990/ring_i</td>
</tr>
<tr>
<td>File Name</td>
<td>Tau profiles</td>
</tr>
<tr>
<td>Hostname</td>
<td>b2e45s.cerning.com</td>
</tr>
<tr>
<td>Local Time</td>
<td>2010-04-28T02:27:58-04:00</td>
</tr>
<tr>
<td>MPI Processor</td>
<td>b2e45s.cerning.com</td>
</tr>
<tr>
<td>Memory Size</td>
<td>66006532 MB</td>
</tr>
<tr>
<td>Node Name</td>
<td>b2e45s.cerning.com</td>
</tr>
<tr>
<td>OS Machine</td>
<td>x86_64</td>
</tr>
<tr>
<td>OS Name</td>
<td>Linux</td>
</tr>
<tr>
<td>OS Release</td>
<td>2.6.18-128.el5_perfor</td>
</tr>
<tr>
<td>OS Version</td>
<td>#1 SMP Mon Jun 29 12:32:22 PDT 2009</td>
</tr>
<tr>
<td>Starting Time</td>
<td>1271572078780</td>
</tr>
<tr>
<td>TAU Architecture</td>
<td>x86_64</td>
</tr>
<tr>
<td>TAU Configuration</td>
<td>./home/...</td>
</tr>
<tr>
<td>TAU Makefile</td>
<td>./home/corning/apps/paraprof/tau-2.19.1/x86_64/i6864 invade emacs</td>
</tr>
<tr>
<td>TAU Version</td>
<td>tau-2.19.1</td>
</tr>
</tbody>
</table>

## Expression

```plaintext
PAPI_L1_DCM
```

![Expression](image)
TAU IDE Integration
TAU IDE Integration

• High performance software development environments
  – Tools may be complicated to use
  – Interfaces and mechanisms differ between platforms / OS

• Integrated development environments
  – Consistent development environment
  – Numerous enhancements to development process
  – Standard in industrial software development

• Integrated performance analysis
  – Tools limited to single platform or programming language
  – Rarely compatible with 3rd party analysis tools
  – Little or no support for parallel projects
TAU and Eclipse

- Provide an interface for configuring TAU’s automatic instrumentation within Eclipse’s build system
- Manage runtime configuration settings and environment variables for execution of TAU instrumented programs

C/C++/Fortran Project in Eclipse → Add or modify an Eclipse build configuration w/ TAU

TAU instrumented libraries → Compilation/linking with TAU libraries

Temporary copy of instrumented code → Program execution

Performance data → Program output
TAU and Eclipse

PerfDMF
Choosing PAPI Counters with TAU in Eclipse

% /soft/perftools/tau/eclipse/eclipse
Configuring Job Submission for Remote Execution on BGQ
Job and System Display for BGQ Resource Manager
Hands-on training with sample codes
Lab Instructions

Get \texttt{workshop.tar.gz} using:

\texttt{\% wget http://tau.uoregon.edu/workshop.tar.gz}

Or

\texttt{\% tar zxf workshop.tar.gz}

And follow the instructions in the README file.

For LiveDVD, see \texttt{~/workshop-point/README} and follow.
Lab Instructions

To profile a code using TAU:

1. Change the compiler name to tau_cxx.sh, tau_f90.sh, tau_cc.sh:
   F90 = tau_f90.sh

2. Choose TAU stub makefile
   % module load tau
   % export TAU_MAKEFILE=
   $TAULIBDIR/Makefile.tau-[options]

3. If stub makefile has \texttt{-papi} in its name, set the
   \texttt{TAU\_METRICS} environment variable:
   % export
   \texttt{TAU\_METRICS=TIME:PAPI\_L2\_DCM:PAPI\_TOT\_CYC}...

4. Run:
   %qsub \texttt{-I \-l \texttt{nodes=1:ppn=8 \texttt{-X; mpirun \-np 8 \texttt{/a.out}}

5. Build and run workshop examples, then run
   \texttt{pprof/paraprof}
Support Acknowledgements

- Department of Energy (DOE)
  - Office of Science contracts
  - SciDAC contracts, LBL
  - LLNL-LANL-SNL ASC/NNSA contract

- Department of Defense (DoD)
  - PETTT, HPTi

- National Science Foundation (NSF)
  - POINT, SI-2

- University of Oregon
  - Dr. A. Malony, W. Spear, Dr. Lee, S. Biersdorff, S. Millstein, N. Chaimov

- University of Tennessee, Knoxville
  - Dr. Shirley Moore

- T.U. Dresden, GWT
  - Dr. Wolfgang Nagel and Dr. Andreas Knupfer

- Research Centre Juelich
  - Dr. Bernd Mohr, Dr. Felix Wolf