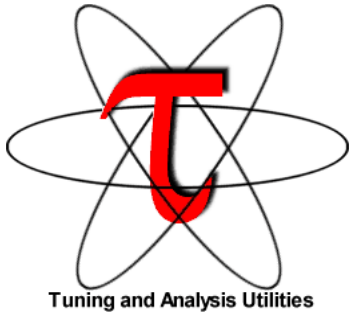
TAU Performance System®



Sameer Shende
sameer@cs.uoregon.edu
University of Oregon
http://tau.uoregon.edu

TAU hands-on exercises

TAU tutorial exercise objectives

- Familiarise with usage of TAU tools
  - complementary tools' capabilities & interoperability
- Prepare to apply tools productively to *your* applications(s)
- Exercise is based on a small portable benchmark code
  - unlikely to have significant optimisation opportunities

- Optional (recommended) exercise extensions
  - analyse performance of alternative configurations
  - investigate effectiveness of system-specific compiler/MPI optimisations and/or placement/binding/affinity capabilities
  - investigate scalability and analyse scalability limiters
  - compare performance on different HPC platforms
  - …

Local Installation (*mn4, CEA*)

- Setup preferred program environment compilers

```
% module load intel/2017.7 impi/2017.4
% source /home/nct00/nct00005/tau.bashrc; getnode;
# If you have previous performance data from Score-P, you may view it with TAU's paraprof
% paraprof profile.cubex &
```

For PerfExplorer:

```
% cp /home/nct00/nct00005/pkg/tar/data.tgz . ; tar zxf data.tgz; cd data
% cat README; cd tau; ./upload.sh; perfexplorer &
  and follow the steps
```

NPB-MZ-MPI Suite

- The NAS Parallel Benchmark suite (MPI+OpenMP version)
  - Available from:

    http://www.nas.nasa.gov/Software/NPB
  - 3 benchmarks in Fortran77
  - Configurable for various sizes & classes
- Move into the NPB3.3-MZ-MPI root directory

```
% ls
bin/     common/  jobscript/  Makefile  README.install   SP-MZ/
BT-MZ/   config/  LU-MZ/      README    README.tutorial  sys/
```

- Subdirectories contain source code for each benchmark
  - plus additional configuration and common code
- The provided distribution has already been configured for the tutorial, such that it's ready to “make” one or more of the benchmarks and install them into a (tool-specific) “bin” subdirectory

Compiler and MPI modules

- Select appropriate compiler / MPI combination (if not done automatically)

```
% module load intel/2017.4 impi/2017.4
```

- Copy tutorial sources to your scratch directory

```
% cd /gpfs/scratch/nct00/$USER
% tar zxvf /home/nct00/nct00004/tutorial/NPB3.3-MZ-MPI.tar.gz
% cd NPB3.3-MZ-MPI
```

NPB-MZ-MPI / BT: config/make.def : Use default compiler!

```
#                SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS.
#
#-----------------------------------------------------------------------

#-----------------------------------------------------------------------
# Configured for generic MPI with GCC compiler
#-----------------------------------------------------------------------
#COMPILER = -homp            # Cray/CCE compiler
COMPILER = -fopenmp          # GNU/GCC compiler
#COMPILER = -qopenmp         # Intel compiler

...
#-----------------------------------------------------------------------
# The Fortran compiler used for MPI programs
#-----------------------------------------------------------------------
# MPIF77 = mpif77 # OpenMPI with Intel compiler
MPIF77 = mpif77
# Alternative variant to perform instrumentation
# MPIF77 = tau_f90.sh
# PREP is a generic preposition macro for instrumentation preparation
#MPIF77 = $(PREP) mpif77 -f77=ifort
#MPIF77 = scorep …
...
```

Default (no instrumentation)

Uncomment MPIF77 and comment Score-P's compiler wrapper to generate uninstrumented binary!

Building an NPB-MZ-MPI Benchmark

```
% make
  ========================================
  =      NAS PARALLEL BENCHMARKS 3.3     =
  =      MPI+OpenMP Multi-Zone Versions  =
  =      F77                             =
  ========================================

  To make a NAS multi-zone benchmark type

        make <benchmark-name> CLASS=<class> NPROCS=<nprocs>

  where <benchmark-name> is "bt-mz", "lu-mz", or "sp-mz"
        <class>          is "S", "W", "A" through "F"
        <nprocs>         is number of processes

  [...]

 ***********************************************************
 * Custom build configuration is specified in config/make.def  *
 * Suggested tutorial exercise configuration for HPC systems:  *
 *       make bt-mz CLASS=C NPROCS=8                           *
 ***********************************************************
```

- Type "make" for instructions

Building an NPB-MZ-MPI Benchmark

```
% make bt-mz CLASS=C NPROCS=8
make[1]: Entering directory `BT-MZ'
make[2]: Entering directory `sys'
cc  -o setparams setparams.c -lm
make[2]: Leaving directory `sys'
../sys/setparams bt-mz 8 C
make[2]: Entering directory `../BT-MZ'
mpif77  -c  -O3 -fopenmp       bt.f
[…]
mpif77  -c  -O3 -fopenmp       mpi_setup.f
cd ../common;  mpif77  -c  -O3 -fopenmp       print_results.f
cd ../common;  mpif77  -c  -O3 -fopenmp       timers.f
mpif77  -O3 -fopenmp   -o ../bin/bt-mz_C.8 bt.o
 initialize.o exact_solution.o exact_rhs.o set_constants.o adi.o
 rhs.o zone_setup.o x_solve.o y_solve.o  exch_qbc.o solve_subs.o
 z_solve.o add.o error.o verify.o mpi_setup.o ../common/print_results.o
 ../common/timers.o
make[2]: Leaving directory `BT-MZ'
Built executable ../bin/bt-mz_C.8
make[1]: Leaving directory `BT-MZ'
```

- Specify the benchmark configuration
  - benchmark name: **bt-mz**, lu-mz, sp-mz
  - the number of MPI processes: NPROCS=**8**
  - the benchmark class (S, W, A, B, C, D, E): CLASS=**C**

Shortcut: `% make suite`

NPB-MZ-MPI / BT (Block Tridiagonal Solver)

- What does it do?
  - Solves a discretized version of the unsteady, compressible Navier-Stokes equations in three spatial dimensions
  - Performs 200 time-steps on a regular 3-dimensional grid
- Implemented in 20 or so Fortran77 source modules

- Uses MPI & OpenMP in combination
  - 8 processes each with 12 threads should be reasonable for 2 compute nodes of mn4
  - bt-mz_B.8 should run in less than 4 seconds
  - bt-mz_C.8 should run in around 15 seconds

NPB-MZ-MPI / BT with TAU (tau_exec –T ompt)

```
% cd bin
% cp ../jobscript/mn4/tau_ompt.sbatch .
% sbatch./tau_ompt.sbatch
% cat mzmpibt.o<job_id>
 NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
 Number of zones:   16 x 16
 Iterations:  200   dt:   0.000300
 Number of active processes:     8
 Total number of threads:       64 (  8.0 threads/process)

 Time step    1
 Time step   20
  [...]
 Time step  180
 Time step  200
 Verification Successful

 BT-MZ Benchmark Completed.
 Time in seconds = 17.34
% paraprof &
% paraprof --pack bt.ppk
<Copy file over to desktop using scp>
% paraprof bt.ppk  &
```

- Copy jobscript and launch as a hybrid MPI+OpenMP application

tau_exec launches un-instrumented binaries!

NPB-MZ-MPI / BT with TAU's Sampling (tau_exec –T ompt –ebs)

```
% cd bin
% cp ../jobscript/mn4/tau_ompt_ebs.slurm .
% sbatch ./tau_ompt_ebs.slurm
% cat mzmpibt.o<job_id>
 NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
 Number of zones:   16 x 16
 Iterations:  200   dt:   0.000300
 Number of active processes:     8
 Total number of threads:       64 (  8.0 threads/process)

 Time step    1
 Time step   20
  [...]
 Time step  180
 Time step  200
 Verification Successful

 BT-MZ Benchmark Completed.
 Time in seconds = 17.34
% paraprof &
% paraprof --pack bt_ebs.ppk
<Copy file over to desktop using scp>
% paraprof bt_ebs &
```

- Copy jobscript and launch as a hybrid MPI+OpenMP application

tau_exec launches un-instrumented binaries!

NPB-MZ-MPI / BT with TAU's Sampling (tau_exec –T pthread –ebs)

```
% cd bin
% cp ../jobscript/mn4/tau_pthread_ebs.slurm .
% sbatch ./tau_pthread_ebs.slurm
% cat mzmpibt.o<job_id>
 NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
 Number of zones:   16 x 16
 Iterations:  200   dt:   0.000300
 Number of active processes:     8
 Total number of threads:       64 (  8.0 threads/process)

 Time step    1
 Time step   20
  [...]
 Time step  180
 Time step  200
 Verification Successful

 BT-MZ Benchmark Completed.
 Time in seconds = 17.34
% paraprof &
% paraprof --pack bt_ebs.ppk
<Copy file over to desktop using scp>
% paraprof bt_ebs &
```

- Copy jobscript and launch as a hybrid MPI+OpenMP application
- Using Pthread instead of OMPT for tracking threads!

tau_exec launches un-instrumented binaries!

TAU's Source Instrumentation

TAU Source Instrumentation

- Edit config/make.def to adjust build configuration
  - Uncomment specification of compiler/linker: MPIF77 = tau_f90.sh
- Make clean and build new tool-specific executable

```
% make clean
% make bt-mz CLASS=C NPROCS=8
Built executable ../bin.tau/bt-mz_C.8
```

- Change to the directory containing the new executable before running it with the desired tool configuration

```
% cd bin.tau
% cp ../jobscript/mn4/run.msub .
% sbatch reference.slurm
```

NPB-MZ-MPI / BT: config/make.def

```
#                SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS.
#
#---------------------------------------------------------------------

#---------------------------------------------------------------------
# Configured for generic MPI with GCC compiler
#---------------------------------------------------------------------
#OPENMP  = -fopenmp      # GCC compiler
OPENMP = -openmp         # Intel compiler

...
#---------------------------------------------------------------------
# The Fortran compiler used for MPI programs
#---------------------------------------------------------------------
# MPIF77 = mpif90 # OpenMPI with Intel compiler

# Alternative variant to perform instrumentation
MPIF77 = tau_f90.sh

# PREP is a generic preposition macro for instrumentation preparation
#MPIF77 = $(PREP) mpif77 -f77=ifort
#MPIF77 = scorep …
...
```

Default (no instrumentation)

Uncomment TAU's compiler wrapper to do source instrumentation with TAU
Comment out Score-P wrapper

Building an NPB-MZ-MPI Benchmark with TAU's Opari configuration

```
% export TAU_MAKEFILE=$TAU/Makefile.tau-icpc-papi-mpi-pdt-openmp-opari
% make suite
make[1]: Entering directory `BT-MZ'
make[2]: Entering directory `sys'
cc   -o setparams setparams.c -lm
make[2]: Leaving directory `sys'
../sys/setparams bt-mz 8 C
make[2]: Entering directory `../BT-MZ'
tau_f90.sh -c  -O3 -g -openmp        bt.f
[…]
tau_f90.sh -c  -O3 -g -openmp        mpi_setup.f
cd ../common;  mpiifort -c  -O3 -g -openmp        print_results.f
cd ../common;  mpiifort -c  -O3 -g -openmp        timers.f
tau_f90.sh -O3 -g -openmp   -o ../bin.tau/bt-mz_C.8 bt.o
 initialize.o exact_solution.o exact_rhs.o set_constants.o adi.o
 rhs.o zone_setup.o x_solve.o y_solve.o  exch_qbc.o solve_subs.o
 z_solve.o add.o error.o verify.o mpi_setup.o ../common/print_results.o
 ../common/timers.o
make[2]: Leaving directory `BT-MZ'
Built executable ../bin.tau/bt-mz_C.8
make[1]: Leaving directory `BT-MZ'
```

- Specify the benchmark configuration
  - benchmark name: **bt-mz**, lu-mz, sp-mz
  - the number of MPI processes: NPROCS=**8**
  - the benchmark class (S, W, A, B, C, D, E): CLASS=**C**

Shortcut: `% make suite`

NPB-MZ-MPI / BT with TAU

```
% cd bin.tau
% cp ../jobscript/mn4/tau.sbatch .
% sbatch tau.sbatch
% cat mzmpibt.o<job_id>
 NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
 Number of zones:   16 x 16
 Iterations:  200   dt:   0.000300
 Number of active processes:     8
 Total number of threads:       64 (  8.0 threads/process)

 Time step    1
 Time step   20
  [...]
 Time step  180
 Time step  200
 Verification Successful

 BT-MZ Benchmark Completed.
 Time in seconds = 22.34
% paraprof &
% paraprof --pack bt.ppk
<Copy file over to desktop using scp>
% paraprof bt.ppk  &          OR    % getnode; paraprof &
```

- Copy jobscript and launch as a hybrid MPI+OpenMP application

Hint: save the benchmark output (or note the run time) to be able to refer to it later

TAU's Analysis Tools: PerfExplorer

Using TAUdb

- Configure TAUdb (Done by each user)

  % taudb_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/taudb.cfg file
- Configure PerfExplorer (Done by each user)

  % perfexplorer_configure
- Execute PerfExplorer

  % perfexplorer

Local Installation (*mn4, BSC*)

- Setup preferred program environment compilers

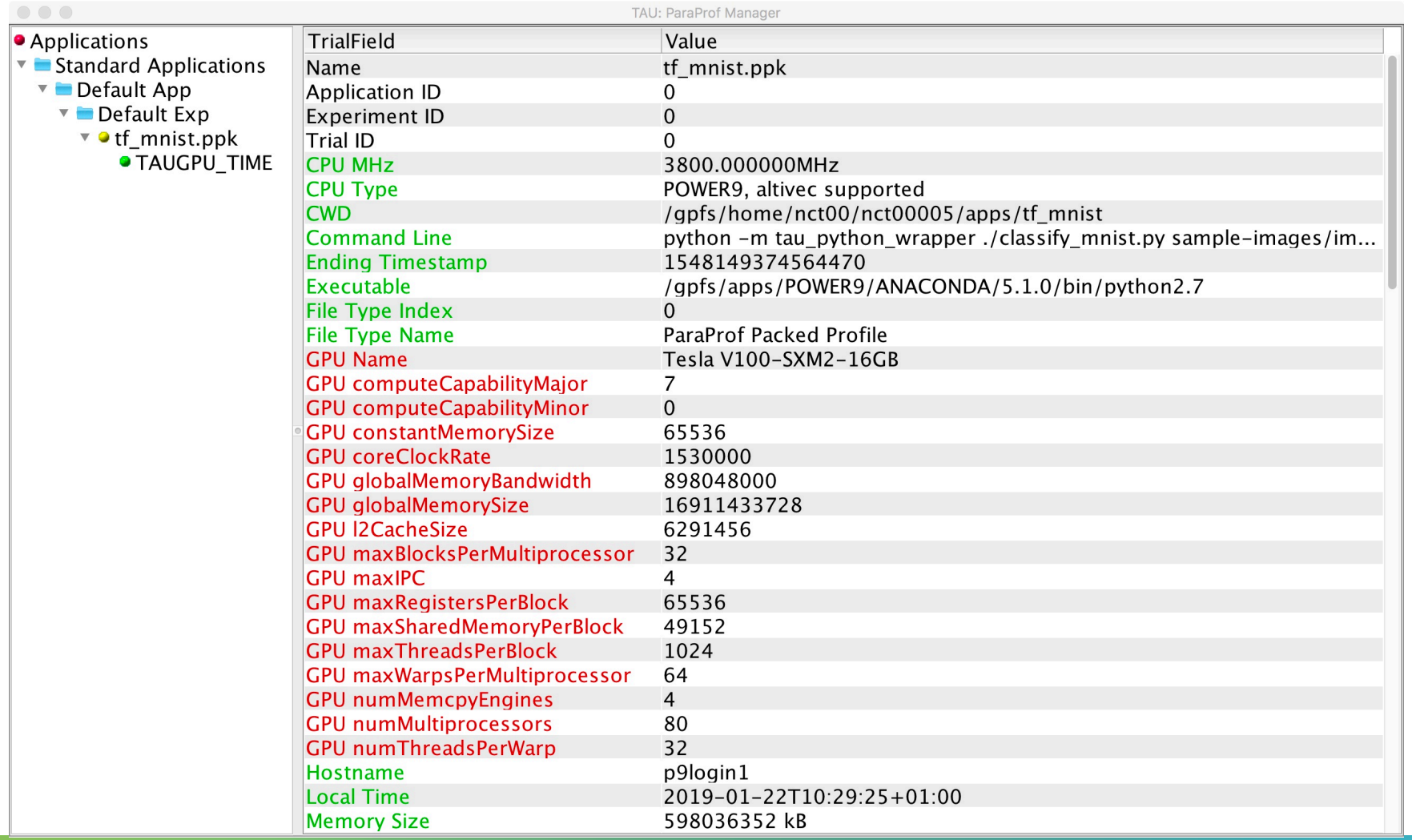
```
% module load intel/2017.4 impi/2017.4
% source /home/nct00/nct00005/tau.bashrc
% cp /home/nct00/nct00005/pkgs/tar/data.tgz . ; tar zxf data.tgz
% cd data
% cat README
  and follow the steps
% cd tau
% ./upload.sh
% getnode
% perfexplorer
```

Profiling MNIST dataset with IBM PowerAI on CTE-POWER with TAU

- AI program to recognize digits

```
% ssh –Y plogin1.bsc.es
% source /home/nct00/nct00005/tau.bashrc
% cp /home/nct00/nct00005/apps/tf_mnist.tgz . ; tar zxf tf_mninst.tgz; cd tf_mnist
% module load powerAI
% source /opt/DL/tensorflow/bin/tensorflow-activate
% cat r
python ./classify_mnist.py sample-images/img_8.jpg
% cat rt
/bin/rm -rf prof*
export TAU_CALLPATH_DEPTH=100
tau_python -T cupti,serial -cupti -ebs ./classify_mnist.py sample-images/img_8.jpg
% ./r
% ./rt
% paraprof
```

Mnist PowerAI example



TrialField	Value
Name	tf_mnist.ppk
Application ID	0
Experiment ID	0
Trial ID	0
CPU MHz	3800.000000MHz
CPU Type	POWER9, altivec supported
CWD	/gpfs/home/nct00/nct00005/apps/tf_mnist
Command Line	python -m tau_python_wrapper ./classify_mnist.py sample-images/im...
Ending Timestamp	1548149374564470
Executable	/gpfs/apps/POWER9/ANACONDA/5.1.0/bin/python2.7
File Type Index	0
File Type Name	ParaProf Packed Profile
GPU Name	Tesla V100-SXM2-16GB
GPU computeCapabilityMajor	7
GPU computeCapabilityMinor	0
GPU constantMemorySize	65536
GPU coreClockRate	1530000
GPU globalMemoryBandwidth	898048000
GPU globalMemorySize	16911433728
GPU l2CacheSize	6291456
GPU maxBlocksPerMultiprocessor	32
GPU maxIPC	4
GPU maxRegistersPerBlock	65536
GPU maxSharedMemoryPerBlock	49152
GPU maxThreadsPerBlock	1024
GPU maxWarpsPerMultiprocessor	64
GPU numMemcpyEngines	4
GPU numMultiprocessors	80
GPU numThreadsPerWarp	32
Hostname	p9login1
Local Time	2019-01-22T10:29:25+01:00
Memory Size	598036352 kB

ParaProf context event window

TAU: ParaProf: Context Events for: node 0, thread 2 - tf_mnist.ppk

Name	Total	NumSamp...	MaxValue	MinValue	MeanValue	Std. Dev.
Allocatable Blocks Per SM given Registers used (Blocks)	77	7	32	2	11	10.268
Allocatable Blocks Per SM given Shared Memory usage (Blocks)	6,145	7	1,024	1	877.857	357.975
Allocatable Blocks per SM given Thread count (Blocks)	70	7	16	2	10	6.928
Block Size	3,584	7	1,024	128	512	443.405
GPU Occupancy (Warps)	388	7	64	4	55.429	20.996
Local Memory (bytes per thread)	0	7	0	0	0	0
Local Registers (per thread)	215	7	86	16	30.714	23.15
Shared Dynamic Memory (bytes)	0	7	0	0	0	0
Shared Static Memory (bytes)	32,768	7	32,768	0	4,681.143	11,466.411
▾ void Eigen::internal::EigenMetaKernel<Eigen::TensorEvaluator<Eigen::TensorAssignOp<E						
Allocatable Blocks Per SM given Registers used (Blocks)	2	1	2	2	2	0
Allocatable Blocks Per SM given Shared Memory usage (Blocks)	1,024	1	1,024	1,024	1,024	0
Allocatable Blocks per SM given Thread count (Blocks)	2	1	2	2	2	0
Block Size	1,024	1	1,024	1,024	1,024	0
GPU Occupancy (Warps)	64	1	64	64	64	0
Local Memory (bytes per thread)	0	1	0	0	0	0
Local Registers (per thread)	21	1	21	21	21	0
Shared Dynamic Memory (bytes)	0	1	0	0	0	0
Shared Static Memory (bytes)	0	1	0	0	0	0
▸ void Eigen::internal::EigenMetaKernel<Eigen::TensorEvaluator<Eigen::TensorAssignOp<E						
▸ void Eigen::internal::EigenMetaKernel<Eigen::TensorEvaluator<Eigen::TensorAssignOp<E						
▸ void tensorflow::(anonymous namespace)::GenerateNormalizedProb<float>(float const*,						
▸ void tensorflow::functor::RowReduceKernel<cub::TransformInputIterator<float, tensorflo						
▸ void tensorflow::functor::RowReduceKernel<float const*, float*, cub::Max>(float const*, f						
▾ volta_sgemm_32x32_sliced1x4_nn						
Allocatable Blocks Per SM given Registers used (Blocks)	5	1	5	5	5	0
Allocatable Blocks Per SM given Shared Memory usage (Blocks)	1	1	1	1	1	0
Allocatable Blocks per SM given Thread count (Blocks)	16	1	16	16	16	0
Block Size	128	1	128	128	128	0
GPU Occupancy (Warps)	4	1	4	4	4	0
Local Memory (bytes per thread)	0	1	0	0	0	0
Local Registers (per thread)	86	1	86	86	86	0
Shared Dynamic Memory (bytes)	0	1	0	0	0	0
Shared Static Memory (bytes)	32,768	1	32,768	32,768	32,768	0

ParaProf thread statistics window for Mnist using IBM PowerAI

TAU: ParaProf: Statistics for: node 0, thread 2 - tf_mnist.ppk

Name	Exclusive...	Inclusive ...	Calls	Child ...
.TAU application	358,389	358,421	1	7
volta_sgemm_32x32_sliced1x4_nn	20	20	1	0
void Eigen::internal::EigenMetaKernel<Eigen::TensorEvaluator<Eigen::TensorAssignOp<Eigen::TensorMap<Eigen::Ten	2	2	1	0
void Eigen::internal::EigenMetaKernel<Eigen::TensorEvaluator<Eigen::TensorAssignOp<Eigen::TensorMap<Eigen::Ten	2	2	1	0
void Eigen::internal::EigenMetaKernel<Eigen::TensorEvaluator<Eigen::TensorAssignOp<Eigen::TensorMap<Eigen::Ten	2	2	1	0
void tensorflow::(anonymous namespace)::GenerateNormalizedProb<float>(float const*, float const*, float const*, flo	2	2	1	0
void tensorflow::functor::RowReduceKernel<cub::TransformInputIterator<float, tensorflow::(anonymous namespace):	2	2	1	0
void tensorflow::functor::RowReduceKernel<float const*, float*, cub::Max>(float const*, float*, int, int, cub::Max, std:	2	2	1	0

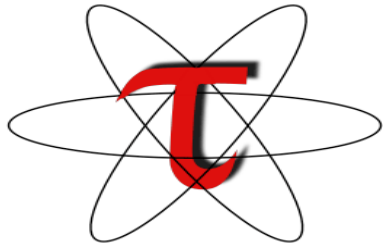
Acknowledgement

This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration) responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering, and early testbed platforms, in support of the nation's exascale computing imperative.

Support Acknowledgments

- US Department of Energy (DOE)
  - Office of Science contracts, ECP
  - SciDAC, LBL contracts
  - LLNL-LANL-SNL ASC/NNSA contract
  - Battelle, PNNL contract
  - ANL, ORNL contract
- Department of Defense (DoD)
  - PETTT, HPCMP
- National Science Foundation (NSF)
  - Glassbox, SI-2
- NASA
- CEA
- IBM
- Partners:
  - University of Oregon
  - ParaTools, Inc., ParaTools, SAS
  - The Ohio State University
  - University of Tennessee, Knoxville
  - T.U. Dresden, GWT
  - Juelich Supercomputing Center

Download TAU from U. Oregon



http://tau.uoregon.edu

http://www.hpclinux.com [LiveDVD, OVA]

http://e4s.io [Containers for Extreme-Scale Scientific Software Stack]

Free download, open source, BSD license