NPB3.3-MZ-MPI/BT tutorial example OpenMP+MPI application (generic cluster version)

Brian Wylie
Jülich Supercomputing Centre
b.wylie@fz-juelich.de

March 2012

Tutorial exercise objectives

- Familiarise with usage of VI-HPS tools
  - complementary tools' capabilities & interoperatibilty
- Prepare to apply tools productively to *your* application(s)
- Exercise is based on a small portable benchmark code
  - unlikely to have significant optimization opportunities

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

Workshop systems (NLHPC)

Use “ssh -X account@host” to one of the workshop systems

- @development.dim.uchile.cl or @syntagma.cmm.uchile.cl

then load the desired MPI+compiler base module:

Base module	***openmpi_gcc***	***openmpi_intel*** (recommended)	***intelmpi*** (only syntagma)
Compiler	*GCC*	*Intel*	*Intel*
OpenMP	-fopenmp	-openmp	-openmp
MPI	*Open MPI*	*Open MPI*	*Intel MPI*
C compiler	mpicc	mpicc	mpiicc
C++ compiler	mpicxx	mpicxx	mpiicpc
F90 compiler	mpif90	mpif90	mpiifort

and submit SGE jobscripts with “qsub script.sge”

Local installation (NLHPC)

- Load the UNITE module and then the tool of interest

```
% module use /home/apps/UNITE/master
% module load UNITE
% module avail
cube/3.4.1                           periscope/1.4-openmpi-intel
scalasca/1.4.1-openmpi-gcc44         scalasca/1.4.1-openmpi-intel
scorep/1.0-openmpi-intel             vampir/7.5.0-demo
% module load scalasca/1.4.1-openmpi-intel
```

- Distinct modules for each compiler+MPI library combination
  - MPI libraries typically not binary compatible
  - OpenMP run-time libraries not binary compatible
  - Compiler-based instrumentation is compiler-specific
    - measurement collection, analysis & examination shouldn't be
- Tutorial sources should be copied to your own directory where you can then work with them

```
% cp -r  $UNITE_ROOT/tutorial/NPB3.3-MZ-MPI  $WORK
```

NPB-MZ-MPI suite

- NAS Parallel Benchmark suite (sample MZ-MPI version)
  - Available from `http://www.nas.nasa.gov/Software/NPB`
  - 3 benchmarks (all in Fortran77, using OpenMP+MPI)
  - Configurable for various sizes & classes
- Move into the NPB3.3-MZ-MPI root directory

```
% cd NPB3.3-MZ-MPI; ls
BT-MZ/   LU-MZ/   SP-MZ/
bin/     common/  config/  jobscript/       Makefile          README   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” benchmarks and install them into a (tool-specific) “bin” subdirectory

Building an NPB-MZ-MPI benchmark

- Type “make” for instructions

```
% make
  =========================================
  =      NAS Parallel Benchmarks 3.3      =
  =      MPI+OpenMP Multi-Zone versions   =
  =========================================

  To make a NAS multi-zone benchmark type

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

  To make a set of benchmarks, create the file config/suite.def
  according to the instructions in config/suite.def.template and type

         make suite

 ***************************************************************
 * Custom build configuration is specified in config/make.def  *
 * Suggested tutorial benchmark specification:                 *
 *       make bt-mz CLASS=B NPROCS=4                           *
 ***************************************************************
```

Building the NPB-MZ-MPI/BT benchmark

- Specify the benchmark configuration
  - benchmark name: **bt-mz**, lu-mz, sp-mz
  - the number of MPI processes: NPROCS=**4**
  - the benchmark class (S, W, A, B, C, D, E, F): CLASS=**B**

```
% make bt-mz CLASS=B NPROCS=4
cd BT-MZ; make CLASS=B NPROCS=4 VERSION=
gmake: Entering directory 'BT-MZ'
cd ../sys; cc  -o setparams setparams.c -lm
../sys/setparams bt-mz 4 B
mpif77 -c  -O3 -[f]openmp  bt.f
...
mpif77 -c  -O3 -[f]openmp  setup_mpi.f
cd ../common;  mpif77 -c  -O3 -[f]openmp  print_results.f
cd ../common;  mpif77 -c  -O3 -[f]openmp  timers.f
mpif77 -O3 -[f]openmp  -o ../bin/bt-mz_B.4 \
    bt.o make_set.o initialize.o exact_solution.o exact_rhs.o \
    set_constants.o adi.o define.o copy_faces.o rhs.o solve_subs.o \
    x_solve.o y_solve.o z_solve.o add.o error.o verify.o setup_mpi.o \
    ../common/print_results.o ../common/timers.o
Built executable ../bin/bt-mz_B.4
gmake: Leaving directory 'BT-MZ'
```

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

- What does it do?
  - Solves a discretized version of unsteady, compressible Navier-Stokes equations in three spatial dimensions
  - Performs 200 time-steps on a regular 3-dimensional grid using ADI and verifies solution error within acceptable limit
  - Intra-zone computation with OpenMP, inter-zone with MPI
- Implemented in 20 or so Fortran77 source modules

- Runs with any number of MPI processes & OpenMP threads
  - bt-mz_B.4 x4 is reasonable (increase to 4x6 as appropriate)
    - excess processes idle when run with more than compiled
  - bt-mz_B.4 x4 should run in around 30 seconds
    - typically runs more efficiently with more processes than threads
  - CLASS=C does much more work and takes much longer!

NPB-MZ-MPI/BT reference execution

- Set OMP_NUM_THREADS and launch as an MPI application

```
% cd bin;  OMP_NUM_THREADS=4  mpiexec -np 4 ./bt-mz_B.4
 NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
 Number of zones:    8 x   8
 Iterations:  200    dt:   0.000300
 Number of active processes:      4

 Time step    1
 Time step   20
 Time step   40
 Time step   60
 Time step   80
 Time step  100
 Time step  120
 Time step  140
 Time step  160
 Time step  180
 Time step  200
 Verification Successful

 BT-MZ Benchmark Completed.
 Time in seconds = 28.86
```

Hint: copy/edit example batch scripts from jobscript directory:
`% qsub ../jobscript/run.sge`

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

Tutorial exercise steps

- The tutorial steps are similar and repeated for each tool
- Use the provided NPB3.3-MZ-MPI tutorial directory

```
% cd NPB3.3-MZ-MPI; ls
BT-MZ/  LU-MZ/  SP-MZ/
bin/    common/ config/ jobscript/      Makefile        README  sys/
```

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

```
% make clean
% make bt-mz CLASS=B NPROCS=4
Built executable ../bin.$(TOOL)/bt-mz_B.4
```

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

```
% cd bin.$(TOOL)
% export OMP_NUM_THREADS=4 ...
% mpiexec -np 4 ./bt-mz_B.4
```

Hint: check available scripts:

```
% qsub ../jobscript/$TOOL.sge
```

NPB-MZ-MPI/BT build configuration definition

- config/make.def

```
#             SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS
#---------------------------------------------------------------------
# Items in this file may need to be changed for each platform.
…
#OPENMP = -fopenmp  # GCC
OPENMP  = -openmp   # Intel
#---------------------------------------------------------------------
# The Fortran compiler used for hybrid MPI programs
#---------------------------------------------------------------------
MPIF77 = mpif77

# Alternative variants to perform instrumentation
#MPIF77 = marmotf77
#MPIF77 = psc_instrument  mpif77
#MPIF77 = scalasca -instrument  mpif77
#MPIF77 = tau_f90.sh
#MPIF77 = vtf77 -vt:hyb -vt:f77  mpif77
#MPIF77 = scorep --user  mpif77

# PREP is a generic preposition macro for instrumentation preparation
#MPIF77 = $(PREP)  mpif77
...
```

Set flag according to compiler

Default (no instrumentation)

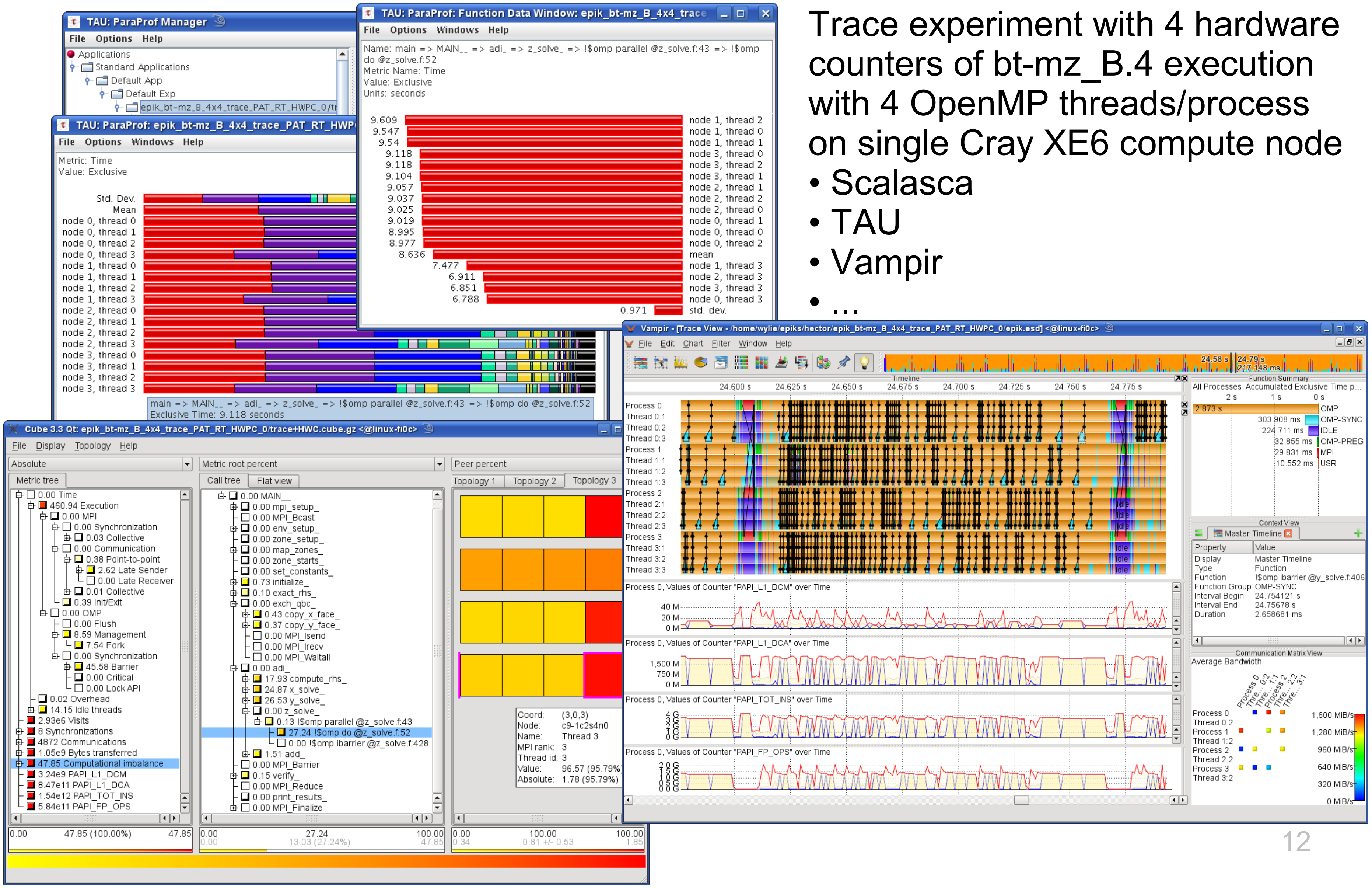
Hint: uncomment one of these alternative compiler wrappers to perform instrumentation ...

… or this for generic variant

NPB-MZ-MPI/BT analyses with VI-HPS tools

Trace experiment with 4 hardware counters of bt-mz_B.4 execution with 4 OpenMP threads/process on single Cray XE6 compute node

- Scalasca
- TAU
- Vampir
- ...



TAU: ParaProf Manager

File Options Help

Applications
Standard Applications
Default App
Default Exp
epik_bt-mz_B_4x4_trace_PAT_RT_HWPC_0/tr

TAU: ParaProf: Function Data Window: epik_bt-mz_B_4x4_trace

File Options Windows Help

Name: main => MAIN__ => adi_ => z_solve_ => !$omp parallel @z_solve.f:43 => !$omp do @z_solve.f:52
Metric Name: Time
Value: Exclusive
Units: seconds

Value	
9.609	node 1, thread 2
9.547	node 1, thread 0
9.54	node 1, thread 1
9.118	node 3, thread 0
9.118	node 3, thread 2
9.104	node 3, thread 1
9.057	node 2, thread 1
9.037	node 2, thread 2
9.025	node 2, thread 0
9.019	node 0, thread 1
8.995	node 0, thread 0
8.977	node 0, thread 2
8.636	mean
7.477	node 1, thread 3
6.911	node 2, thread 3
6.851	node 3, thread 3
6.788	node 0, thread 3
0.971	std. dev.

TAU: ParaProf: epik_bt-mz_B_4x4_trace_PAT_RT_HWP

File Options Windows Help

Metric: Time
Value: Exclusive

Std. Dev.
Mean
node 0, thread 0
node 0, thread 1
node 0, thread 2
node 0, thread 3
node 1, thread 0
node 1, thread 1
node 1, thread 2
node 1, thread 3
node 2, thread 0
node 2, thread 1
node 2, thread 2
node 2, thread 3
node 3, thread 0
node 3, thread 1
node 3, thread 2
node 3, thread 3

main => MAIN__ => adi_ => z_solve_ => !$omp parallel @z_solve.f:43 => !$omp do @z_solve.f:52
Exclusive Time: 9.118 seconds

Cube 3.3 Qt: epik_bt-mz_B_4x4_trace_PAT_RT_HWPC_0/trace+HWC.cube.gz <@linux-fi0c>

File Display Topology Help

Absolute | Metric root percent | Peer percent

Metric tree | Call tree | Flat view | Topology 1 | Topology 2 | Topology 3

Metric tree:

```
0.00 Time
  460.94 Execution
    0.00 MPI
      0.00 Synchronization
        0.03 Collective
      0.00 Communication
        0.38 Point-to-point
          2.62 Late Sender
          0.00 Late Receiver
        0.01 Collective
      0.39 Init/Exit
    0.00 OMP
      0.00 Flush
      8.59 Management
        7.54 Fork
      0.00 Synchronization
        45.58 Barrier
        0.00 Critical
        0.00 Lock API
  0.02 Overhead
  14.15 Idle threads
2.93e6 Visits
8 Synchronizations
4872 Communications
1.05e9 Bytes transferred
47.85 Computational imbalance
3.24e9 PAPI_L1_DCM
8.47e11 PAPI_L1_DCA
1.54e12 PAPI_TOT_INS
5.84e11 PAPI_FP_OPS
```

Call tree:

```
0.00 MAIN__
  0.00 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  0.73 initialize_
  0.10 exact_rhs_
  0.00 exch_qbc_
    0.43 copy_x_face_
    0.37 copy_y_face_
    0.00 MPI_Isend
    0.00 MPI_Irecv
    0.00 MPI_Waitall
  0.00 adi_
    17.93 compute_rhs_
    24.87 x_solve_
    26.53 y_solve_
    0.00 z_solve_
      0.13 !$omp parallel @z_solve.f:43
        27.24 !$omp do @z_solve.f:52
        0.00 !$omp ibarrier @z_solve.f:428
  1.51 add_
  0.00 MPI_Barrier
  0.15 verify_
  0.00 MPI_Reduce
  0.00 print_results_
  0.00 MPI_Finalize
```

Coord:	(3,0,3)
Node:	c9-1c2s4n0
Name:	Thread 3
MPI rank:	3
Thread id:	3
Value:	96.57 (95.79%
Absolute:	1.78 (95.79%)

0.00 47.85 (100.00%) 47.85 | 0.00 27.24 13.03 (27.24%) 100.00 47.85 | 0.00 100.00 0.81 +/- 0.53 100.00 1.85

Vampir - [Trace View - /home/wylie/epiks/hector/epik_bt-mz_B_4x4_trace_PAT_RT_HWPC_0/epik.esd] <@linux-fi0c>

File Edit Chart Filter Window Help

Timeline: 24.600 s 24.625 s 24.650 s 24.675 s 24.700 s 24.725 s 24.750 s 24.775 s

Process 0, Thread 0:1, Thread 0:2, Thread 0:3, Process 1, Thread 1:1, Thread 1:2, Thread 1:3, Process 2, Thread 2:1, Thread 2:2, Thread 2:3, Process 3, Thread 3:1, Thread 3:2, Thread 3:3

Process 0, Values of Counter "PAPI_L1_DCM" over Time

Process 0, Values of Counter "PAPI_L1_DCA" over Time

Process 0, Values of Counter "PAPI_TOT_INS" over Time

Process 0, Values of Counter "PAPI_FP_OPS" over Time

Function Summary — All Processes, Accumulated Exclusive Time p...

2.873 s	OMP
303.908 ms	OMP-SYNC
224.711 ms	IDLE
32.855 ms	OMP-PREG
29.831 ms	MPI
10.552 ms	USR

Context View — Master Timeline

Property	Value
Display	Master Timeline
Type	Function
Function	!$omp ibarrier @y_solve.f:406
Function Group	OMP-SYNC
Interval Begin	24.754121 s
Interval End	24.75678 s
Duration	2.658681 ms

Communication Matrix View — Average Bandwidth

Local installation (UNITE)

- Load the UNITE module and then the tool of interest

```
% module load UNITE
% module avail
kcachegrind/1.4       marmot/2.4.0           paraver/4.0
periscope/1.4b        scalasca/1.3.3         tau/2.20.3
vampir/7.3.0          vampirserver/2.3.0     vampirtrace/5.11
% module load scalasca/1.3.3
```

- Often distinct modules for each MPI/compiler combination
  - MPI libraries typically not binary compatible
  - OpenMP run-time libraries not binary compatible
  - Compiler-based instrumentation is compiler-specific
    - measurement collection, analysis & examination shouldn't be
- Tutorial sources should be copied to your own directory where you can then work with them

```
% cp -r ~hpclab01/tutorial/NPB3.3-MZ-MPI $WORK
```

Workshop systems (hardware)

System	***juropa***	***cluster-beta***	***jugene***
Domain	fz-juelich.de	rz.rwth-aachen.de	fz-juelich.de
Vendor	Sun/Bull	Bull	IBM
Type	Constellation	Bullx	BlueGene/P
Network	Infiniband	Infiniband	BlueGene/P
Processors	Xeon X5570	Xeon X5675 / X7550	PowerPC 450
Frequency	2930 MHz	3060 / 2000 MHz	850 MHz
Compute nodes	3288	1358 / 346	2304
Chips per node	2	2 / 4	32
Cores per chip	4	6 / 8	4
Threads per core	2	1 / 1	1
Memory per node	24 GB	24-96 / 64-256 GB	2 GB

Workshop systems (software environment)

System	***juropa***	***cluster-beta***	***jugene***
domain	fz-juelich.de	rz.rwth-aachen.de	fz-juelich.de
Filesystem	*Lustre*	*Lustre*	*GPFS*
Parallel filesys	$WORK	/hpcwork/$USER	$WORK
Compiler	*Intel*	*Intel*	*IBM XL*
OpenMP	-openmp	-openmp	-qsmp=omp
MPI	*ParaStation*	*IntelMPI / OpenMPI*	*BG MPICH2*
C compiler	mpicc	mpiicc / mpicc	mpixlc_r
C++ compiler	mpicxx	mpiicpc / mpicxx	mpixlcxx_r
F90 compiler	mpif90	mpiifort / mpif90	mpixlf90_r
Queue	*Moab/PBS*	*LSF*	*LoadLeveler*
job submit	msub job	bsub < job	llsubmit job

Cray peculiarities

- Our tutorials are most often done with the Linux Live DVD so they use generic names
  - MPI compiler: `mpicc, mpicxx, mpif77, mpif90`
  - MPI launcher: `mpiexec -np …`
  - OpenMP flag: `-fopenmp`
- If your system is different, then you need to adapt compiler/launch commands accordingly
- Cray systems are certainly different
  - MPI compiler: `cc, CC, ftn`
  - MPI launcher: `aprun -n …`
  - OpenMP flag: `-fopenmp / -homp / -openmp / -mp` (for GNU, CCE, Intel & PGI/Pathscale, respectively)

Local installation (CINECA 'plx')

- Load your desired PrgEnv, then the Scalasca module

```
% module load profile/advanced
% module avail scalasca
scalasca/1.4.1--IntelMPI-4.0--co-2011.2.137--binary
scalasca/1.4.1--openmpi-1.4.4--gnu-4.5.2
scalasca/1.4.1--openmpi-1.4.4--intel--co-2011.6.233--binary
% module load scalasca/1.4.1--openmpi-1.4.4--intel--co-2011.6.233
```

- Distinct modules for each MPI+compiler environment
  - OpenMPI & Intel MPI, GNU/GCC & Intel compilers
    - MPI libraries typically not binary compatible
    - OpenMP run-time libraries not binary compatible
    - Automatic instrumentation by compiler is compiler-specific
- Tutorial sources should be copied to your own directory where you can then work on them

```
% cp -r  ~bwylie00/tutorial/NPB3.3-MZ-MPI  $CINECA_SCRATCH
```

Local installation (CSC 'louhi')

- Load your desired PrgEnv, then the Scalasca module

```
% module avail scalasca
...
scalasca/1.3.3-cray                   scalasca/1.3.3-gnu
scalasca/1.3.3-intel                  scalasca/1.3.3-pgi
% module load PrgEnv-gnu
% module load scalasca/1.3.3-gnu
```

- Distinct modules for each Cray XT PrgEnv
  - Cray/CCE, GNU/GCC, Intel, Pathscale, PGI compilers
    - Automatic instrumentation by compiler is compiler-specific
      - measurement collection, analysis & examination shouldn't be
- Tutorial sources should be copied to your own directory where you can then work on them

```
% cp -r ~trng01/tutorial/NPB3.3-MZ-MPI $WRKDIR
```

Local installation (CSCS 'palu')

- Load your desired PrgEnv, then the Scalasca module

```
% module load PrgEnv-intel
% module avail scalasca
scalasca/1.3.2
% module load scalasca
```

- Distinct Scalasca modules for each Cray XT PrgEnv
  - Cray/CCE, GNU/GCC, Intel, Pathscale, PGI compilers
  - Scalasca module loaded based on current PrgEnv
  - Scalasca instrumentation by compiler is compiler-specific
    - measurement collection, analysis & examination shouldn't be
- Tutorial sources are located under `~scalasca/tutorial/`

```
% ls ~scalasca/tutorial
NPB3.3-MPI/    NPB3.3-MZ-MPI/    jacobi/
```

- ... and should be copied to your own directory to work on

Local installation (EPCC 'hector2b')

- Load your desired PrgEnv, then the Scalasca module

```
% module avail scalasca
...
scalasca/1.3.3-cray                  scalasca/1.3.3-gnu
scalasca/1.3.3-pathscale             scalasca/1.3.3-pgi(default)
% module load PrgEnv-gnu
% module load scalasca/1.3.3-gnu
```

- Distinct modules for each Cray XT PrgEnv
  - Cray/CCE, GNU/GCC, Intel, Pathscale, PGI compilers
    - Automatic instrumentation by compiler is compiler-specific
      - measurement collection, analysis & examination shouldn't be
- Tutorial sources should be copied to your own directory where you can then work on them

```
% cp -r /usr/local/packages/scalasca/tutorial/NPB3.3-MZ-MPI $HOME
```

Local installation (HLRS 'hermit')

- Load your desired PrgEnv, then the VI-HPS tool module

```
% module load PrgEnv-gnu
% module avail performance
performance/periscope/1.3.2-gnu     performance/scalasca/1.3.3-gnu
performance/tau/2.20.1-gnu          performance/vampirtrace-5.10.1-gnu
performance/vampirserver/2.3.0      performance/vampir/7.3.0
% module load scalasca
```

- Distinct modules for each Cray XT PrgEnv
  - Cray/CCE, GNU/GCC, Intel, Pathscale, PGI compilers
    - Automatic instrumentation by compiler is compiler-specific
      - measurement collection, analysis & examination shouldn't be
- Tutorial sources should be copied to your own directory where you can then work on them

```
% cp -r /sw/hermit/hlrs/performance/tutorial/NPB3.3-MZ-MPI $HOME
```

Local installation (IMAG/CIMENT 'gofree')

VI-HPS

- Load your desired PrgEnv, then the Scalasca module

```
% module avail scalasca
scalasca/1.4.1-openmpi-1.4.4_gcc-4.6.2
scalasca/1.4-openmpi-1.4.4_gcc-4.6.2
scalasca/1.4.1-openmpi-1.4.4_intel-12.0
% module load gcc/4.6.2      openmpi/1.4.4_gcc-4.6.2
% module load scalasca/1.4.1-openmpi-1.4.4_gcc-4.6.2
```

- Distinct modules for each MPI+compiler environment
  - OpenMPI & Intel MPI, GNU/GCC & Intel compilers
    - MPI libraries typically not binary compatible
    - OpenMP run-time libraries not binary compatible
    - Automatic instrumentation by compiler is compiler-specific
- Tutorial sources should be copied to your own directory where you can then work on them

```
% cp -r  ~wylie/tutorial/NPB3.3-MZ-MPI  $WRKDIR
```