

International HPC Summer School 2022: Performance analysis and optimization

Hands-on:
NPB-MZ-MPI/BT Reference run

VI-HPS Team
Ilya Zhukov – Jülich Supercomputing Centre

Performance analysis steps

- 0.0 Reference preparation for validation
- 1.0 Program instrumentation
 - 1.1 Summary measurement collection
 - 1.2 Summary analysis report examination
- 2.0 Summary experiment scoring
 - 2.1 Summary measurement collection with filtering
- 3.0 Trace measurement with filtering
 - 3.1 Event trace examination & analysis

Performance analysis steps

- 0.0 Reference preparation for validation
- 1.0 Program instrumentation
 - 1.1 Summary measurement collection
 - 1.2 Summary analysis report examination
- 2.0 Summary experiment scoring
 - 2.1 Summary measurement collection with filtering
- 3.0 Trace measurement with filtering
 - 3.1 Event trace examination & analysis

Tutorial exercise objectives

- Familiarize with a typical **workflow**
- 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
 - Investigate reproducibility of measurements: CPU frequency/Turboboost
 - 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
 - ...

Access to Bridges2

```
# Connect to the Bridges2 login node
% ssh -X userid@bridges2.psc.edu
```

- Logging in to Bridges2

```
$PROJECT
/ocean
...

/ocean/projects/tra210016p/zhukov/ihpcss22/tutorial
```

- File systems & directories
 - Use \$PROJECT for the tutorial

Tutorial materials

- More extensive documentation:
 - <https://www.psc.edu/resources/bridges-2/user-guide-2>

Compiling & job submission

- Development environment: GNU compiler with OpenMPI
 - Use OpenMPI compiler wrappers
 - `mpicc`
 - `mpicxx`
 - `mpif77`
- Bridges2 uses the SLURM batch system
 - Jobs submitted from tutorial accounts will need to specify our reservation

```
% sbatch jobscript.sbatch  
% squeue -u $USER  
% scancel <jobid>
```

← Submit job
← View job queue
← Cancel job

Local installation

- VI-HPS tools not yet installed system-wide
 - Enable and load local tool installations
 - Required for each shell session

```
% module use /ocean/projects/tra210016p/zhukov/ihpcss22/modules/  
% module load openmpi/4.1.1-gcc8.3.1 likwid/5.2.0
```

- Copy tutorial sources to a working directory, e.g. \$HOME

```
% cd $PROJECT  
% tar zxvf /ocean/projects/tra210016p/zhukov/ihpcss22/tutorial/NPB3.3-MZ-MPI.tar.gz  
% cd NPB3.3-MZ-MPI
```

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 is ready to “make” one or more of the benchmarks and install them into a (tool-specific) “bin” subdirectory

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
 - Proposed hands-on setup on Bridges2:
 - 2 compute nodes with 2 x AMD EPYC 7742 64-Core Processor
 - 64 cores per CPU, 128 cores per node
 - 256GB RAM each
 - Get more details using likwid-topology
 - 8 MPI processes with 6 OpenMP threads each, not utilizing all cores
 - bt-mz_C.8 should run ~15 seconds on Bridges2

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 Bridges2:      *
*      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_B.30 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**
 - Or use "**make suite**"

System topology

```
% likwid-topology
```

```
-----  
CPU name:      AMD EPYC 7742 64-Core Processor  
CPU type:      AMD K17 (Zen2) architecture  
CPU stepping:  0  
*****
```

```
Hardware Thread Topology  
*****
```

```
Sockets:      2  
Cores per socket: 64  
Threads per core: 1  
-----
```

HWThread	Thread	Core	Die	Socket	Available
0	0	0	0	0	*
1	0	1	0	0	*
...					

- likwid-topology: thread and cache topology on x86 CPUs
- 128 cores available per node
- Hyperthreading deactivated

NPB-MZ-MPI / BT reference execution

```
% cd bin
% cp ../jobscript/bridges2/reference.sbatch.C.8 .
% less reference.sbatch.C.8
#!/bin/bash
#SBATCH -J mzmplibt          # job name
#SBATCH -o ref-C.8-%j.out    # stdout output file
#SBATCH -e ref-C.8-%j.err    # stderr output file
#SBATCH --nodes=2            # requested nodes (28 cores/node)
#SBATCH --ntasks=8           # requested MPI tasks
#SBATCH --cpus-per-task=6     # requested logical CPUs/threads per task
#SBATCH --partition RM       # partition to use
#SBATCH --account=tra210016p # account to charge
#SBATCH --export=ALL         # export env varibales
#SBATCH --time=00:10:00      # max wallclock time (hh:mm:ss)

# setup modules, add tools to PATH
set -x
module load openmpi/4.1.1-gcc8.3.1
export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

# benchmark configuration
export NPB_MZ_BLOAD=0
CLASS=C
PROCS=$SLURM_NTASKS
EXE=./bt-mz_${CLASS}.${PROCS}
SOCKETS=2
TASKS_PER_SOCKET=$(( ( $SLURM_NTASKS / $SLURM_NNODES ) / $SOCKETS ))

mpirun -report-bindings --map-by ppr:$TASKS_PER_SOCKET:socket:pe=$SLURM_CPUS_PER_TASK \
-n $SLURM_NTASKS $EXE
```

- Examine jobscript

NPB-MZ-MPI / BT reference execution (cont)

```
% sbatch reference.sbatch.C.8
% less ref-C.8-<job_id>.out
NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
Number of zones:   16 x   16
Iterations:  200   dt:   0.000100
Number of active processes:    8
Total number of threads:      48 ( 6.0 threads/process)

Time step    1
Time step   20
Time step   40

[...]

Time step  180
Time step  200
Verification Successful

BT-MZ Benchmark Completed.
Time in seconds = 14.25
```

- Copy jobscript and launch as a hybrid MPI+OpenMP application
- Reproducible? CPU frequency constant? Turboboost?

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