Hands-on: CooLMUC-3 Intel KNL partition
NPB-MZ-MPI / bt-mz_C.32

Ilya Zhukov
Jülich Supercomputing Centre

Tutorial exercise objectives

- Familiarise with usage of VI-HPS 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
  - …

Compiler and MPI modules (CooLMUC-3)

- Ensure that desired compiler and MPI modules (toolchain) are loaded first

```
% module list
Currently Loaded Modulefiles:
 1) admin/1.0      3) intel/17.0    5) mpi.intel/2017
 2) tempdir/1.0    4) mkl/2017      6) lrz/default
```

Alternatively switch compilers (gcc) and/or MPI (ompi) …

- Set-up and load the required VI-HPS tools modules (when needed)

```
% source /home/hpc/a2c06/lu23voh/load-vihps
```

- Copy tutorial sources to your $SCRATCH_LEGACY directory

```
% cd $SCRATCH_LEGACY
% cp /home/hpc/a2c06/lu23voh/tutorial/NPB3.3-MZ-MPI.tar.gz .
% tar xvf 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
  - but config/make.def may first need to be adjusted to specify appropriate compiler flags

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

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

#---------------------------------------------------------------------------
# Configured for generic MPI with compiler-specific OpenMP flags
#---------------------------------------------------------------------------
#COMPFLAGS = -fopenmp -march=knl -mtune=knl # GNU/GCC compiler
#COMPFLAGS = -openmp -xMIC-AVX512           # Intel14/15 compiler
COMPFLAGS = -qopenmp -xMIC-AVX512           # Intel compiler

...
#---------------------------------------------------------------------------
# The Fortran compiler used for MPI programs
#---------------------------------------------------------------------------
MPIF77 = mpif77

# Alternative variant to perform instrumentation
#MPIF77 = scorep --user  mpif77

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

Uncomment flags specification according to current compiler

Default (no instrumentation)

Hint: uncomment a compiler wrapper to do instrumentation

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=32                           *
 ***************************************************************
```

- Type "make" for instructions

Building an NPB-MZ-MPI Benchmark

```
% make bt-mz CLASS=C NPROCS=32
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 32 C
make[2]: Entering directory `../BT-MZ'
mpif77  -c  -O3 -qopenmp       bt.f
                                    […]
mpif77  -c  -O3 -qopenmp       mpi_setup.f
cd ../common;  mpif77  -c  -O3 -qopenmp        print_results.f
cd ../common;  mpif77  -c  -O3 -qopenmp        timers.f
mpif77  -O3 -qopenmp   -o ../bin/bt-mz_C.32 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.32
make[1]: Leaving directory `BT-MZ'
```

- Specify the benchmark configuration
  - benchmark name: **bt-mz**, lu-mz, sp-mz
  - the number of MPI processes: NPROCS=**32**
  - 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
  - 32 processes each with 4 threads should be reasonable for 2 compute nodes of CooLMUC-3
  - bt-mz_B.32 should run in less than 8 seconds
  - bt-mz_C.32 should run in around 17 seconds

NPB-MZ-MPI / BT Reference Execution

```
% cd bin
% cp ../jobscript/coolmuc3/reference.sbatch .

% cat reference.sbatch
 #!/bin/bash
 #SBATCH -J npb_btmz                    # Job name
 #SBATCH -o npb_btmz.o%j                # Name of stdout output file(%j expands to jobId)
 #SBATCH -e npb_btmz.e%j                # Name of stderr output file(%j expands to jobId)
 #SBATCH --get-user-env                 # Copy environment
 #SBATCH --clusters=mpp3                # KNL cluster
 #SBATCH --nodes=2                      # Total number of nodes requested
 #SBATCH -n 32                          # Total number of mpi tasks requested
 #SBATCH -t 00:05:00                    # Run time (hh:mm:ss) - 5 minutes
 #SBATCH --constraint=cache,quad        # Request partition in cache-quadrant mode
 #SBATCH --reservation=TuningWorkshop   # Reservation

 source /etc/profile.d/modules.sh

 # benchmark configuration
 export OMP_NUM_THREADS=4
 export NPB_MZ_BLOAD=0
 PROCS=32
 CLASS=C
 EXE=./bt-mz_$CLASS.$PROCS

 # run the application
 mpiexec $EXE
```

- Copy and examine jobscript

NPB-MZ-MPI / BT Reference Execution

```
% sbatch reference.sbatch

% cat npb_btmz.o<job_id>
 NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
 Number of zones:  16 x  16
 Iterations: 200    dt:   0.000100
 Number of active processes:    32
 Total number of threads:    128   (  4.0 threads/process)

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

 BT-MZ Benchmark Completed.
 Iterations      =                   200
 Time in seconds =                 16.32
```

- Launch jobscript and examine output

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

Tutorial Exercise Steps

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

```
% make clean
% make bt-mz CLASS=C NPROCS=32
Built executable ../bin.scorep/bt-mz_C.32
```

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

```
% cd bin.scorep
% cp ../jobscript/coolmuc3/scorep.sbatch .
% sbatch scorep.sbatch
```