Hands-on exercise: NPB-MZ-MPI / BT

VI-HPS Team

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
  - …

Local UNITE installation

- VI-HPS tools accessible through the UNITE framework
  - Currently installed in a non-default location
    - Need to extend the module search path manually
  - Load the UNITE meta-module

```
% module use /gpfslocal/pub/vihps/UNITE/local
% module load UNITE
% module avail
----- /gpfslocal/pub/vihps/UNITE/modulefiles/tools -----
 [...]
scalasca/2.1a3-intel13-impi(default)
scorep/1.2.3-intel13-impi(default)
scorep/1.2.3-intel13-impi-cuda50
scorep/1.2.3-intel13-impi-cuda55
vampir/8.2
 [...]
----- /gpfslocal/pub/vihps/UNITE/modulefiles/utils -----
maqao/2.0.0
```

Local installation for workshop

- VI-HPS-TW meta-module loads default set of VI-HPS tools (and corresponding compiler+MPI) for workshop

```
% module load VI-HPS-TW
(load) intel version 13.0.1
Using intel compiler to configure intelmpi wrappers...
(load) intelmpi version 4.0.3
(load) papi version 5.3.0_intel13
scorep/1.2.3-intel13-impi loaded
cube4/4.2.3rc1-intel loaded
scalasca/2.1a3-intel13-impi loaded
tau/2.22.2-intelmpi-intel13 loaded
vampir/8.2 loaded
% module list
1) UNITE/1.1                        6) cube4/4.2.3rc1-intel
2) intel/13.0.1                     7) scalasca/2.1a3-intel13-impi
3) intelmpi/4.0.3                   8) tau/2.22.2-intelmpi-intel13
4) papi/5.3.0_intel13               9) vampir/8.2
5) scorep/1.2.3-intel13-impi  10) VI-HPS-TW/intel13
```

Local Installation (cont.)

- Module help provides location of tutorial sources

```
% module help VI-HPS-TW
[...]
VI-HPS-TW: VI-HPS Tuning Workshop environment
for intel/13.0.1 compilers and intelmpi/4.0.3 MPI

VI-HPS tutorial sources: $UNITE_ROOT/tutorial
VI-HPS sample experiments: $UNITE_ROOT/samples
[...]
```

- Copy tutorial sources to your working directory

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

  - (When available, generally advisable to use a parallel filesystem such as $WORK)

- 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

Building an NPB-MZ-MPI Benchmark

- Type “make” for instructions

```
% 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=B NPROCS=4                           *
 ***************************************************************
```

Building an NPB-MZ-MPI 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): CLASS=**B**

```
% make bt-mz CLASS=B NPROCS=4
cd BT-MZ; make CLASS=B NPROCS=4 VERSION=
make: Entering directory 'BT-MZ'
cd ../sys; cc  -o setparams setparams.c
../sys/setparams bt-mz 4 B
mpiifort -c  -O3 -openmp bt.f
 [...]
cd ../common;  mpiifort -c  -O3 -openmp timers.f
mpiifort -O3 -openmp -o ../bin/bt-mz_B.4 \
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
Built executable ../bin/bt-mz_B.4
make: 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
- Implemented in 20 or so Fortran77 source modules

- Uses MPI & OpenMP in combination
  - 4 processes with 4 threads each should be reasonable
  - bt-mz_B.4 should run in around 13 seconds
  - bt-mz_C.4 should take around 3-4x longer

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

```
% cd bin
% cp ../jobscript/mds/run.ll .
% less run.ll
% llsubmit run.ll
% cat NPB_B.<id>
 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
 Total number of threads:     16   (   4.0 threads/process)

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

 BT-MZ Benchmark Completed.
 Time in seconds = 12.74
```

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 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 -n 4 ./bt-mz_B.4
```

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

```
#            SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS
#-----------------------------------------------------------------------
# Items in this file may need to be changed for each platform.
#-----------------------------------------------------------------------
...
#-----------------------------------------------------------------------
# The Fortran compiler used for MPI programs
#-----------------------------------------
MPIF77 = mpiifort -fpp

# Alternative variants to perform instrumentation
#MPIF77 = psc_instrument -u user,mpi,omp -s ${PROGRAM}.sir mpiifort
#MPIF77 = tau_f90.sh
#MPIF77 = scalasca -instrument mpiifort
#MPIF77 = vtf77 -vt:hyb -vt:f77 mpiifort
#MPIF77 = scorep --user mpiifort

# PREP is a generic preposition macro for instrumentation preparation
#MPIF77 = $(PREP) mpiifort

# This links MPI Fortran programs; usually the same as ${MPIF77}
FLINK   = $(MPIF77)
...
```

Default (no instrumentation)

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