Compiling Environment Practical

Presenter: Ilias Katsardis

Obtain HPL

HPL is free to use we can download it from http://www.netlib.org/benchmark/hpl/hpl-2.2.tar.gz
For your ease the software can be found and downloaded locally in ECMWF.

- `cd $SCRATCH`
- `mkdir hpltest`
- `cd hpltest`
- `cp ~trx/compenv_practical/hpl-2.2.tar.gz .`
- `tar -zxvf hpl-2.2.tar.gz`
- `cd hpl-2.2/`

Understand the Compilation Procedure

Usually on the top dir we can follow a certain trend of filenames left there from the developers to help us build the software even if no other instructions are provided. To see what files exist on the top dir we do a simple file listing:

- `ls -l`

```
crayik@cca-login3:/scratch/external/cray/crayik/hpltest/hpl-2.2> ls
BUGS       HISTORY  include  Makefile  Make.top  README  src      TODO    www
COPYRIGHT  hpl      INSTALL  makes     man       setup   testing  TUNING
```

In most case like this one we can see some "key" files like: README, Makefile, INSTALL. Usually we start from README.

Cray Compilation

- `cp setup/Make.Linux_Intel64 .`
- `vi Make.Linux_Intel64`

TOPdir = $(HOME)/hpl	TOPdir = /scratch/external/cray/crayik/hpltest/hpl-2.2
LAdir = $(MKLROOT)	LAdir =/opt/intel/composer_xe_2013.5.192/mkl
LAinc = $(LAdir)/mkl/include	LAinc = $(LAdir)/include

LAlib = -L$(LAdir)/mkl/lib/intel64 \ -Wl,--start-group \ $(LAdir)/lib/intel64/libmkl_intel_lp64.a \ $(LAdir)/lib/intel64/libmkl_intel_thread.a \ $(LAdir)/lib/intel64/libmkl_core.a \ -Wl,--end-group -lpthread -ldl	LAlib = -L$(LAdir)/lib/intel64/ $(LAdir)/lib/intel64/libmkl_scalapack_lp64.a \ -Wl,--start-group -lmkl_cdft_core \ -lmkl_intel_lp64 -lmkl_core -lmkl_sequential \ -lmkl_blacs_intelmpi_lp64 \ -Wl,--end-group -static
CC = mpiicc	CC = cc
CCNOOPT = $(HPL_DEFS)	CCNOOPT = $(HPL_DEFS) -hlist=m -O3 -static
OMP_DEFS = -openmp	OMP_DEFS =
CCFLAGS = $(HPL_DEFS) -O3 -w -ansi-alias -i-static -z noexecstack -z relro -z now -nocompchk -Wall	CCFLAGS = $(HPL_DEFS) -O3 -hlist=m -static
LINKFLAGS = $(CCFLAGS) $(OMP_DEFS) -mt_mpi	LINKFLAGS = $(CCFLAGS) $(OMP_DEFS)

Then compile by running:

```
• make arch=Linux_Intel64
```

Intel Compilation

```
• make arch=Linux_Intel64 clean
• prgenvswitchto intel
• make arch=Linux_Intel64
```

GNU Compilation

```
• prgenvswitchto gnu
• cp ~trx/compenv_practical/Make.Linux_Intel64gnu .
• vi Linux_Intel64gnu
    ➢ TOPdir       = /scratch/external/cray/crayik/hpl/hpl-2.2
• make arch=Linux_Intel64gnu
```

Run HPL

Now that we have compiled our binary all we need to do is run it! Keep in mind that with a single ECMWF Node the Rpeak is 1209 TFlops.

```
• qsub -I -q np -l EC_total_tasks=36
• cd XX/bin/Linux_Intel64
• cp ~trx/compenv_practical/HPL.dat .
```

- `aprun -n 36 ./xhpl`

```
➢   ================================================================================
➢   T/V                N    NB     P     Q               Time                 Gflops
➢   --------------------------------------------------------------------------------
➢   WR00L2L2       37056   192     4     9              38.82              8.738e+02
➢   HPL_pdgesv() start time Sun Jan 22 20:01:45 2017
➢
➢   HPL_pdgesv() end time   Sun Jan 22 20:02:23 2017
➢
➢   --VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV-
➢   Max aggregated wall time rfact . . . :               0.39
➢   + Max aggregated wall time pfact . . :               0.19
➢   + Max aggregated wall time mxswp . . :               0.16
➢   Max aggregated wall time update  . . :              35.28
➢   + Max aggregated wall time laswp . . :               2.46
➢   Max aggregated wall time up tr sv  . :               0.05
➢   --------------------------------------------------------------------------------
➢   ||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)=        0.0027774 ...... PASSED
➢   ================================================================================
➢
➢   Finished      1 tests with the following results:
➢                 1 tests completed and passed residual checks,
➢                 0 tests completed and failed residual checks,
➢                 0 tests skipped because of illegal input values.
➢   --------------------------------------------------------------------------------
➢
➢   End of Tests.
➢   ================================================================================
```