AMR Best Practices for Intel® Cluster Ready

HPC ADVISORY COUNCIL
NETWORK OF EXPERTISE
BEST PRACTICES

1. Introduction

The following best practices document is provided as courtesy of the HPC Advisory Council.

2. Application Description:

Adaptive Mesh Refinement (AMR) is actually a collection of three applications for solving a wide variety of problems that benefit from grids with adaptive, inhomogeneous spatial resolution. AMR is the product of the Center for Computational Sciences and Engineering at Lawrence Berkeley National Laboratory. This particular benchmark makes use of the HyperClaw application for solving a gas dynamic problem. It is written primarily in C++.

3. Version Information:

Version of this date is used: 2009 May 11.

4. Prerequisites:

The instructions from this best practice have been tested with the following configuration:

4.1 Hardware:

- Dell PowerEdge M610 38-node cluster
- Intel Xeon X5670 CPUs @ 2.93 MHz
- Memory: 24GB per node @ 1333MHz
- Mellanox ConnectX-2 QDR InfiniBand Adapters
- Mellanox QDR InfiniBand Switch

4.2 Software:

- Intel® Cluster Ready running RHEL 5.5
- Application: AMR
- Compilers: Intel compilers, GNU compilers
- MPI: Intel MPI 4, MVAPICH2 1.5.1p1, Open MPI 1.5.1, Platform MPI 8.0.1
- Benchmark workload: Standard

5. Building AMR

Changes for GNU 4.1.2 Compilers

app/amr/HCAII/GNUmakefile

PROFILE = TRUE

COMP = GNU

FCOMP = GNU

DEBUG = FALSE

USE_XT3 = FALSE

CXXOPTF += -O3 -ffast-math -ftree-vectorize -ftree-loop-linear -funroll-loops

FOPTF += -O3 -ffast-math -ftree-vectorize -ftree-loop-linear -funroll-loops

fOPTF += -O3 -ffast-math -ftree-vectorize -ftree-loop-linear -funroll-loops

app/amr/mk/ Make.rules

Add –lgfortran

$(PRELINK) $(CXX) $(CPPFLAGS) $(CXXFLAGS) $(LDFLAGS) \

-o $@ $(objForExecs) $(libraries) -lgfortran

Changes for Intel Compilers

App/amr/HCAII/GNUmakefile

COMP = Intel

FCOMP = Intel

DEBUG = FALSE

USE_MPI = TRUE

USE_XT3 = FALSE

App/amr/mk/Make.defs

ICC_VERSION := 11.1

ICC_MAJOR_VERSION := 11

ICC_MINOR_VERSION := 1

Changes for MVAPICH2

app/amr/HCAII/GNUmakefile

USE_MPI = TRUE

CXX := mpicxx

FC := mpif77

fC := mpif77

app/amr/mk/Make.mpi

ifndef MPI_HOME

MPI_HOME=/lustre/application/mvapich2-1.5.1p1-gnu

endif

Changes for MPI: Platform MPI

App/amr/mk/Make.defs

```
CXX := mpiCC
FC  := mpif77
fC  := mpif77
```

Make.mpi

```
ifndef MPI_HOME
  MPI_HOME=/opt/platform_mpi
endif
ifneq ($(WHICHLINUX), JACQUARD)
  ifneq ($(WHICHLINUX), COLUMBIA)
    BL_MPI_LIBS += -lmpi
  endif
endif
ifeq ($(WHICHLINUX), GENERICLINUX)
  LIBRARY_LOCATIONS += $(MPI_HOME)/lib
  INCLUDE_LOCATIONS += $(MPI_HOME)/include
endif
```

Changes for MPI: Open MPI

App/amr/mk/Make.defs

```
CXX := mpicxx
FC  := mpif77
fC  := mpif77
```

Make.mpi

```
ifndef MPI_HOME
  MPI_HOME=/usr/mpi/gcc/openmpi-1.4.2/
endif
ifneq ($(WHICHLINUX), JACQUARD)
  ifneq ($(WHICHLINUX), COLUMBIA)
    #BL_MPI_LIBS += -lmpich
    BL_MPI_LIBS += -lmpi
  endif
endif
ifeq ($(WHICHLINUX), GENERICLINUX)
  LIBRARY_LOCATIONS += $(MPI_HOME)/lib
  INCLUDE_LOCATIONS += $(MPI_HOME)/include
endif
```

6. Running AMR

Setting up the dataset

```
% dataset=standard
% cd ${HPCMP_DIR}/ded/amr/$dataset
% mkdir test
% cp ${HPCMP_DIR}/ded/amr/standard/inp/inputs test
% cp ${HPCMP_DIR}/ded/amr/standard/inp/probin test
% DATA=${HPCMP_DIR}/ded/amr/standard/inp/inputs
```

For executable compiled using Intel compilers:

```
% EXE=hc3d.Linux.Intel.Intel.MPI.ex
```

For executable compiled using GNU compilers:

```
% EXE=hc3d.Linux.GNU.GNU.MPI.ex
```

Running with Intel MPI and MVAPICH2

```
% mpdboot -r ssh -f ~/mpd.hosts -n 38
% mpiexec -np 456 -IB $EXE $DATA
%mpdallexit
```

Running with Open MPI

```
%mpirun -np 456 -mca btl self,sm,openib -hostfile ~/hostfile -mca mpi_paffinity_alone 1 $EXE $DATA
```

Running with Platform MPI

```
% mpirun -np 456 -IBV -cpu_bind -prot -hostfile ~/hostfile $EXE $DATA
```

HPC ADVISORY COUNCIL
NETWORK OF EXPERTISE

350 Oakmead Pkwy, Sunnyvale, CA 94085
Tel: 408-970-3400 • Fax: 408-970-3403
www.hpcadvisorycouncil.com