AMBER 11 (PMEMD) with GPUDirect

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:

Amber is the collective name for a suite of programs that allow users to carry out molecular dynamics simulations, particularly on biomolecules. Amber11 is centered around the sander and pmemd simulation program. PMEMD is a version of sander that is optimized for speed, parallel scaling and NVIDIA GPU acceleration.

3. Version Information:

Download Amber 11 (PMEMD) at:

http://ambermd.org/gpus/

Download the list of benchmarks for Amber from:

http://ambermd.org/gpus/AMBER11_GPU_Benchmarks.tar.bz2

4. Prerequisites:

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

4.1 Hardware:

The instructions from this best practice have been tested on the HPC Advisory Council, Dell PowerEdge M610 blade server based cluster.

- AMD Opteron 6172 12-core (Magny-Cours) CPUs
- NVIDIA Tesla C2050 cards
- Mellanox ConnectX-2 QDR InfiniBand Adapters
- Mellnaox QDR InfiniBand Switch

4.2 Software:

- OS: RHEL 5 Update 4
- InfiniBand Software: Mellanox OFED 1.5.1 with GPU Direct Support (version a417)
- MPI: MVAPICH2 version 1.5
- NVIDIA Development Driver for Linux version 256.35
- NVIDIA CUDA Toolkit 3.1 and CUDA SDK 3.1

5. Setting up for GPUDirect Support

Download the Mellanox OFED for Linux with GPUDirect support (Version a417) which contains the bundle that provides support for Graphics Processing Unit (GPU) Direct for Mellanox InfiniBand drivers.

GPU Direct enables sharing pages between Mellanox HCA and the GPU installed on the system, this support aims to save memory copies and CPU utilization, and as a result a better performance.

Perform the following installation for all the compute nodes:

5.1 Kernel Installation

Install the required kernel RPMs, run:

```
# rpm –force -ivh kernel-2* kernel-devel-* kernel-headers-*
```

Modify the boot loader configuration file if needed

```
# /etc/grub.conf
```

Reboot the machine with the new kernel

```
# reboot
```

5.2 MLNX OFED Drivers Installation

Mount the ISO file:

```
# mount -o ro,loop MLNX_OFED_LINUX-Nvidia-1.5.1-rhel5.4.iso /mnt
```

Run the installation script:

```
# /mnt/mlnxofedinstall
```

Restart the driver, run:

```
# /etc/init.d/openibd restart
```

5.3 NVIDIA Driver and Toolkit Installation

Install NVIDIA Development Driver for Linux x86_64, available under:

http://developer.nvidia.com/object/cuda_3_1_downloads.html

For example:

```
# devdriver_3.1_linux_64_256.35.run
```

Follow the installation wizard instructions

To make sure that NVIDIA driver was installed successfully:

Load nvidia driver:

```
# modprobe nvidia
```

Check the nvidia driver for version 256.35:

```
# cat /proc/driver/nvidia/version
```

Install the CUDA Toolkit and SDK

```
# ./cudatoolkit_3.1_linux_64_rhel5.4.run
# ./gpucomputingsdk_3.1_linux.run
```

Compile the SDK samples and verify the GPU device by running deviceQuery on each compute node

```
# /root/NVIDIA_GPU_Computing_SDK/C/bin/linux/release/deviceQuery
```

6. Building AMBER 11 with GPU Support

Before building AMBER11 with GPU Support, make sure that the root directory of the files are located is named "pmemd".

It is optional to run the "configure" script to generate the config.h file. To run the configure script, you need to provide arguments. For example, this is one of the ways for generate the config.h file with public fft implementation, don't use netCDF bintraj facility on the Linux, AMD Opteron or later, x86_64 platform.

```
% ./configure linux64_opteron gfortran mpi  pubfft nobintraj
```

Make sure that the config.h file has the right set of flags that enables GPU CUDA support, and the directory of MVAPICH2 is installed. Here is a sample of the config.h being used. The highlighted lines indicate changes have been made:

```
MATH_DEFINES =
MATH_LIBS =
FFT_DEFINES = -DPUBFFT
FFT_INCLUDE =
FFT_LIBS =
MPI_HOME = /usr/mpi/gcc/mvapich2-1.5
MPI_DEFINES = -DMPI
MPI_INCLUDE = -I$(MPI_HOME)/include
MPI_LIBDIR = $(MPI_HOME)/lib
MPI_LIBS = -L$(MPI_LIBDIR) -lmpich -lmtl_common -lpthread
DIRFRC_DEFINES = -DDIRFRC_EFS -DDIRFRC_NOVEC –DCUDA
CPP = /lib/cpp
CPPFLAGS = -traditional -P -DMPI -DGVERBOSE
F90_DEFINES = -DFFTLOADBAL_2PROC
F90 = mpif90
MODULE_SUFFIX = mod
F90FLAGS = -c -DMPI –DCUDA
F90_OPT_DBG = -g -traceback
F90_OPT_LO =  -O0
F90_OPT_MED = -O2
F90_OPT_HI =  -O3
F90_OPT_DFLT =  $(F90_OPT_HI)
CC = mpicc
CFLAGS = -DMPI
NVCC = nvcc
CU_FLAGS = -use_fast_math -gencode arch=compute_20,code=sm_20 –ptxas-options="-v" –DMPI
CU_INCLUDES = -I/usr/local/cuda/include $(MPI_INCLUDE)
CU_LIBS = -L/usr/local/cuda/lib64 -L. $(MPI_LIBS)
CU_LOADLIBS = -lcufft -lcudpp64 -lcudart
LOAD = mpif90
LOADFLAGS = -lcurl
LOADLIBS = -Wl
```

8. Running PMEMD

Start MVAPICH2 nodal daemons on all the nodes specified in a hostfile

```
% mpdboot -f <hostfile> -n 8
```

Since MVAPICH2 uses RDMA-Write (RPUT) for doing communications over InfiniBand, therefore GPUDirect will be used automatically by MVAPICH2.

Change to the benchmark directory and run the pmemd executable:

```
% mpirun -np 8 $PMEMD_ROOT/src/pmemd -O -i mdin -c inpcrd -l logfile2
```

To disable GPU Direct, one can force MVAPICH2 to use R3 for doing Send/Recv communications, also disable the MPI one-sided communications for RDMA.

```
% export MV2_RNDV_PROTOCOL=R3
% export MV2_USE_RDMA_ONE_SIDED=0
```

9. Verifying Activities of GPUDirect

One way to verify the code is taking advantage of the GPUDirect feature is by looking at the ib_core parameters for GPU Direct. The activity counters are located in this directory:

```
$ ls -al /sys/module/ib_core/parameters/*
-rw-r--r-- 1 root root 4096 Jul 27 18:15 /sys/module/ib_core/parameters/gpu_direct_enable
```

```
-rw-r--r-- 1 root root 4096 Jul 27 18:15 /sys/module/ib_core/parameters/gpu_direct_fail
-rw-r--r-- 1 root root 4096 Jul 27 18:15 /sys/module/ib_core/parameters/gpu_direct_pages
-rw-r--r-- 1 root root 4096 Jul 27 18:15 /sys/module/ib_core/parameters/gpu_direct_shares
```

The following command could be used to monitor any GPUDirect activities on a node in a 5-second interval.

```
$ watch –d –n5 cat /sys/module/ib_core/parameters/*
```

Also check 'dmesg' and /var/log/messages on any information about GPU messages from the CUDA runtime library.

```
parm:           gpu_direct_enable:Enable GPU Direct [default 1] (int)
parm:           gpu_direct_shares:GPU Direct Calls Number [default 0] (int)
parm:           gpu_direct_pages:GPU Direct Shared Pages Number [default 0] (int)
parm:           gpu_direct_fail:GPU Direct Failures Number [default 0] (int)
```

HPC ADVISORY COUNCIL
NETWORK OF EXPERTISE

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