

# 1 软件介绍

Gromacs 是一个开源的、高效的、基于标准 MPI 并行环境设计的大规模分子动力学并行程序，基于 Gromacs 分子力场，由荷兰的 Groningen 大学 Department of Biophysical Chemistry 开发。Gromacs 主要执行生物大分子如蛋白质、核酸、磷脂等物质的计算，也可研究非生物的有机大分子系统，如高分子多聚体等。Gromacs 可免费下载使用，详情参阅：<http://www.gromacs.org/>

## 2 安装环境

Gromacs 版本: GROMACS 2018

操作系统: Ubuntu 16.04

编译器: GCC

cuda: 9.0

fftw: 3.3.5

cmake: 3.5.1

mpi: openmpi-3.0-gnu

## 3 安装步骤

### 3.1 CUDA

到 <https://developer.nvidia.com/cuda-downloads> 下载对应操作系统的 cuda 安装包。下载后执行:

```
sh NVIDIA-Linux-x86_64-384.125.run
```

然后按照相关的提示输入安装路径即可,本文选择默认路径。详细安装步骤可以参考 CUDA 9 安装手册。

环境变量:

```
cat /etc/profile.d/cuda-env.sh
export PATH=/usr/local/cuda-9.0/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-9.0/lib:/usr/local/cuda-9.0/lib64:$LD_LIBRARY_PATH
```

### 3.2 fftw 库

从 FFTW-3.3.5 开始,应该添加参数--enable-avx2,如果 CPU 支持 512 位宽,添加参数--enable-avx512

单精度编译:

```
./configure --prefix=/public/software/mathlib/fftw337-float --enable-float
--enable-avx2 --enable-shared
make
make install
```

双精度编译:

```
./configure --prefix=/public/software/mathlib/fftw337-double --enable-
avx2 --enable-shared
make
make install
```

## 3.3 cmake

Gromacs 2018.1 要求 cmake 要在 3.4.3 版本或者以上。

上传 cmake-3.4.3.tar.gz 安装包到/public/sourcecode 下。

```
tar zxvf cmake-3.4.3.tar.gz
cd cmake-3.4.3
./configure --prefix=/public/software/cmake-3.4.3
make
make install
```

环境变量:

```
cat /public/software/profile.d/cmake-env.sh
export PATH=/public/software/cmake-3.4.3/bin:$PATH
```

## 3.4 mpi

编译器和 openmpi-3.0.0-gnu 安装。

目前 gromacs.2018.1 在 gpu 环境下选择 openmpi-3.0-gnu 可以安装成功, 没有 gpu 环境的时候选择 openmpi-3.0.0-gnu 和 openmpi-3.0.0-intel 都可以。

## 3.5 Gromacs 源码编译

下载 gromacs-2018.3.tar.gz 安装包到 /public/sourcecode 下, 下面可以登录网页:

<http://manual.gromacs.org/documentation/>

目前最新版为 Gromacs 2018.3。然后在此目录下执行如下命令:

```
tar zxvf gromacs-2018.3.tar.gz
cd gromacs-2018.3
mkdir build-gmx-single #此目录名可以随便取
cd build-gmx-single #到此目录下编译
source /public/software/profile.d/cmake-env.sh #选择 cmake
source /public/software/profile.d/openmpi-gnu-env.sh #选择 mpi
```

cmake 编译:

```
cmake ../ \
-DMAKE_INSTALL_PREFIX=/public/software/gromacs-2018.1 \#安装目录
-DGMX_MPI=on \ #开启 mpi
-DGMX_GPU=on \ #这一行和下一行为 gpu 选项, 没有 gpu 的时候去掉这两行。
-DCUDA_TOOLKIT_ROOT_DIR=/usr/local/cuda \
-DCMAKE_C_COMPILER=mpicc \
-DCMAKE_CXX_COMPILER=mpicxx \
-DFFTW_INCLUDE_DIR=/public/software/mathlib/fftw337-float/include \ #选择
3.2 节安装的 fftw337 库
-DFFTW_LIBRARY=/public/software/mathlib/fftw337-float/lib/libfftw3f.so
```

或者使用下面的命令直接运行, DGMX\_BUILD\_OWN\_FFTW=ON 会自动下载并编译 FFTW, 注意更改 cuda 的安装路径:

```
cmake .. -DGMX_MPI=on -DGMX_GPU=on -DCUDA_TOOLKIT_ROOT_DIR=/usr/local/cuda
-DCMAKE_C_COMPILER=mpicc -DCMAKE_CXX_COMPILER=mpicxx -
DGMX_BUILD_OWN_FFTW=ON -DREGRESSIONTEST_DOWNLOAD=ON
```

make 编译:

```
make -j N #N 为可用的 cpu 进程数。
make install
```

安装成功后会在/public/software/gromacs-2018.1/bin 下生成可执行文件, 包括 gmx\_mpi 等。

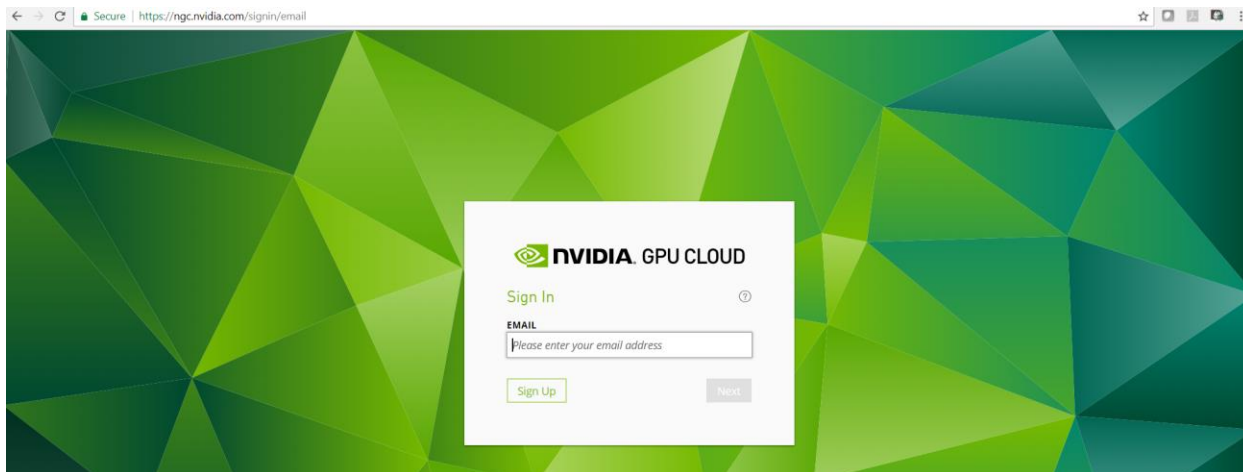
创建环境变量:

```
vi /public/software/profile.d/gromacs-2018.3-env.sh
GMX_INSTALL_DIR=/public/software/gromacs-2018.3
export PATH=${GMX_INSTALL_DIR}/bin:$PATH
export LD_LIBRARY_PATH=${GMX_INSTALL_DIR}/lib:$LD_LIBRARY_PATH
```

## 3.6 Gromacs 基于 NGC Docker 安装

在 NVIDIA GPU Cloud (NGC) 中, 包含 Gromacs 的 docker 镜像文件, 可以直接下载, 导入 Linux docker 环境就可以使用。

注册 NGC



填写注册信息：

**NAME**  
Please enter your full name

**ROLE**  
Please select your job role

**COMPANY**  
Please enter your company or organization

**INDUSTRY**  
Please select your industry

**EMAIL**  
Please enter your email address

**COUNTRY**  
Please select your country

I agree to the [NVIDIA ACCOUNT TERMS OF USE](#)

By obtaining any third party software containers through NGC, I agree that NVIDIA will share my registration information with such third party software providers, who may use my information as permitted by their privacy policies.

Send me NVIDIA GPU Cloud updates and enterprise news

Cancel Clear Sign Up

## 登录 NGC 系统

注册 NGC 后，登录 NGC 系统，就可以看到下图中所有的 HPC Apps 的 docker image 镜像。

Repositories

- nvcr.io
  - caffe
  - caffe2
  - cntk
  - cuda
  - digits
  - mxnet
  - pytorch
  - tensorflow
  - tensorflowrt
  - theano
  - torch
- hpc
  - candle
  - gamess
  - gromacs**
  - lammps
  - lattice-microbes
  - namd
  - relicon
  - vmd
- nvcr.io-hpcvis

hpc/gromacs

```
docker pull nvcr.io/hpc/gromacs:2016.4
```

```
nvcr.io-hpcvis run -ti --rm -v $(pwd):/results nvcr.io/hpc/gromacs:2016.4 "/workspace/run_cont_adh.sh"
```

Note you could also point the CLI command to your local directory instead and run your own scripts (xxx.sh for example). The script below starts the gromacs container and runs the xxx.sh script from your results directory.

```
nvcr.io-hpcvis run -ti --rm -v $(pwd):/results -it --rm nvcr.io/hpc/gromacs:2016.4 "/results/xxx.sh"
```

### 1.2 Running GROMACS Interactively

In this example, we are running the adh\_cubic model benchmark again while inside the /workspace directory in the container. Running interactively is useful for making multiple GROMACS containers run within the same OS image. To run the GROMACS container interactively, issue the following command which starts the container and also mounts your current directory to /results so it is available inside the container. (see the -v options on the

TAG	SIZE	USER	LAST MODIFIED	PULL
2016.4	1.15 GB		November 14, 2017	

## Docker image 下载

下载 image 镜像之前，先要获取 API key:

Registry Get API Key

**Documentation**  
How to use NGC containers on supported platforms >

Repositories

- nvcr.io
  - caffe
  - caffe2

hpc/gromacs

```
docker pull nvcr.io/hpc/gromacs:2016.4
```

点击 Get API Key，即可获得:

## API

## API Information

Your API Key authenticates your use of NGC service when using NGC CLI or the Docker client. Anyone with this API Key has access to all services, actions, and resources on your behalf.

Click Generate API Key to create your own API Key. If you have forgotten or lost your API Key, you can come back to this page to create a new one at any time.

## Usage

Use your API key to log in to the NGC registry as follows.

Docker™ [↗](#)

For the username, enter 'soauthtoken' exactly as shown. It is a special authentication token for all users.

```
$ docker login nvcr.io
Username: soauthtoken
Password: <Your Key>
```

点击“Generate API Key”，并点击弹出对话框中的“Confirm”，系统会生成一个 API Key 作为 nvcr.io 的登录密码，并复制该 Password，用于登录：

## API

## API Information

Your API Key authenticates your use of NGC service when using NGC CLI or the Docker client. Anyone with this API Key has access to all services, actions, and resources on your behalf.

Click Generate API Key to create your own API Key. If you have forgotten or lost your API Key, you can come back to this page to create a new one at any time.

## Usage

Use your API key to log in to the NGC registry as follows.

Docker™ [↗](#)

For the username, enter 'soauthtoken' exactly as shown. It is a special authentication token for all users.

```
$ docker login nvcr.io
Username: soauthtoken
Password: NDg0YzJ0ajIyMjRrdWxzczNzZXB2dG40cWE6MTQ0ZTUxZTUzZGEwNC00ZTgwLWE1NjItODgyNDkyOTJiNzUz
```

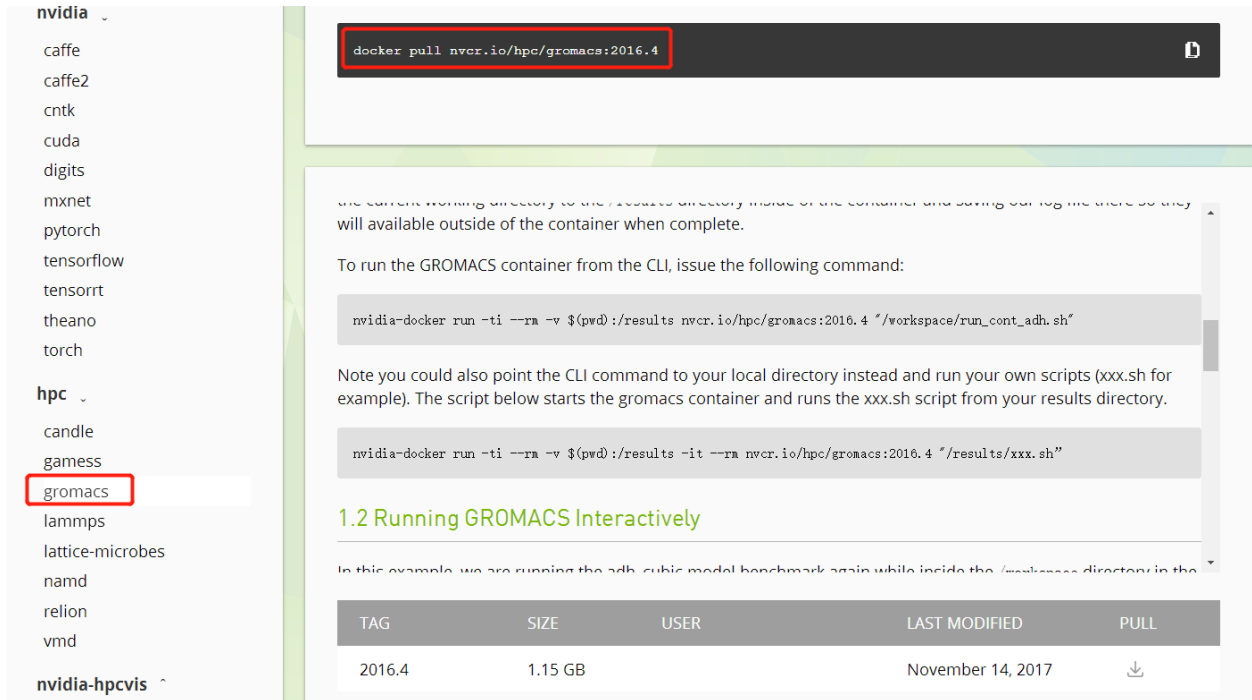
API Key generated successfully. This is the only time your API Key will be displayed. Keep your API Key secret. Do not share it or store it in a place where others can see or copy it.

```
API Key: NDg0YzJ0ajIyMjRrdWxzczNzZXB2dG40cWE6MTQ0ZTUxZTUzZGEwNC00ZTgwLWE1NjItODgyNDkyOTJiNzUz
```

在 Linux 客户端登录方式如下，作者是在 DGX-1 平台下，Ubuntu 16.04 的系统演示的，如下：

```
dgxsa@dgx1:~$ docker login nvcr.io
Username ($oauthtoken): $oauthtoken
Password:
Login Succeeded
dgxsa@dgx1:~$ █
```

登录成功以后，就可以进行 Gromacs 容器下载，如下：



The screenshot shows the NVIDIA Container Toolkit website. On the left sidebar, the 'gromacs' link is highlighted with a red box. The main content area displays the command `docker pull nvcr.io/hpc/gromacs:2016.4` in a terminal window. Below this, there are instructions on how to run the GROMACS container from the CLI, including a command: `nvidia-docker run -ti --rm -v $(pwd):/results nvcr.io/hpc/gromacs:2016.4 "/workspace/run_cont_adh.sh"`. A note mentions that you can also point the CLI command to your local directory. Another command is shown: `nvidia-docker run -ti --rm -v $(pwd):/results -it --rm nvcr.io/hpc/gromacs:2016.4 "/results/xxx.sh"`. The section is titled '1.2 Running GROMACS Interactively'. Below the text, there is a table with columns: TAG, SIZE, USER, LAST MODIFIED, and PULL. The table contains one row for tag '2016.4' with a size of '1.15 GB' and a last modified date of 'November 14, 2017'.

TAG	SIZE	USER	LAST MODIFIED	PULL
2016.4	1.15 GB		November 14, 2017	↓

将命令“`docker pull nvcr.io/hpc/gromacs:2016.4`”复制到 Linux terminal:

```
dgxsa@dgx1:~$ docker login nvcr.io
Username ($oauth token): $oauth token
Password:
Login Succeeded
dgxsa@dgx1:~$ docker pull nvcr.io/hpc/gromacs:2016.4
2016.4: Pulling from hpc/gromacs
```

### 启动 docker 并运行 Gromacs

使用 `nvidia-docker` 命令查看 Gromacs 容器镜像信息。

```

dgxsa@dgx1:~$ nvidia-docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
qp_lammps_180403    v1.0         95f9899acc84     4 days ago      5.5GB
qp_caffe2_mpi_ofed v1.0         f36e5f65e417     6 days ago      7.39GB
qp_caffe2_mpi       v1.0         050f87845aab     6 days ago      6.59GB
nvcr.io/nvidia/tensorrt 18.03-py2   2d6903917375     5 weeks ago     4.33GB
nvcr.io/nvidia/digits 18.02       4f7dc6bf79a6     7 weeks ago     5.45GB
nvcr.io/nvidia/tensorflow 18.02-py3  57ae51ee8b74     7 weeks ago     2.91GB
nvcr.io/nvidia/tensorflow 18.02-py2  5729fa7f740d     7 weeks ago     2.91GB
nvcr.io/nvidia/pytorch 18.02-py3  0fa1c8e0011b     7 weeks ago     5.76GB
nvcr.io/nvidia/caffe   18.02-py2  fc16e9d37dff     7 weeks ago     3.29GB
nvcr.io/nvidia/caffe2  18.02-py3  13aa25b41ad1     8 weeks ago     2.87GB
nvcr.io/nvidia/caffe2  18.02-py2  bfe2278714cd     8 weeks ago     2.86GB
nvcr.io/nvidia/theano  18.02       77ac9f9b866d     8 weeks ago     3.86GB
nvcr.io/nvidia/mxnet   18.02-py3  9731cb1dbf8b     8 weeks ago     2.81GB
nvcr.io/nvidia/cntk    18.02-py3  d1cb490099fd     8 weeks ago     6.4GB
nvcr.io/nvidia/tensorrt 18.02-py2  23f146705293     8 weeks ago     4.31GB
nvcr.io/nvidia/caffe  18.01-py2  c81ff7e540db     3 months ago    3.25GB
nvcr.io/nvidia/mxnet   18.01-py2  ffe7de2f34c3     3 months ago    2.64GB
nvcr.io/nvidia/tensorflow 18.01-py2  377b46c75bfc     3 months ago    2.88GB
nvcr.io/nvidia/pytorch 18.01-py3  27f7f6895e25     3 months ago    4.73GB
nvcr.io/nvidia/caffe2  18.01-py2  fa0881b1b245     3 months ago    2.82GB
nvcr.io/hpc/namd       2.12-171025 9e6c17154075     4 months ago    2.91GB
nvd1.githost.io:4678/dgx/tensorrt 17.12-stage fd28d0eaae4b     4 months ago    4.08GB
nvcr.io/nvidia/tensorflow 17.12       19afd620fc8e     4 months ago    2.88GB
nvd1.githost.io:4678/dgx/cuda 9.0-cudnn7-devel-ubuntu16.04 634be617d3ed     4 months ago    1.75GB
nvcr.io/hpc/gromacs    2016.4       d5711c31ecde     4 months ago    2.86GB
nvcr.io/hpc/lammps     patch230ct2017 d8ae9ee6bf06     4 months ago    3.48GB
nvcr.io/nvidia/pytorch 17.11       fb6782fcfd54     4 months ago    4.76GB

```

启动 docker 镜像:

```

dgxsa@dgx1:~$ nvidia-docker run --name MyGromacs -v
/home/dgxsa/chengyi/share:/data -it nvcr.io/hpc/gromacs:2016.4 /bin/bash

```

### docker run Options

- -i -t or -it : 交互式, 连接到一个"tty"
- --name : 给容器命名
- -v /home/dgxsa/chengyi/share:/data : 将 host 主机的/home/dgxsa/chengyi/myshare 存储目录映射到容器的 data 目录。

```

dgxsa@dgx1:~$ nvidia-docker run --name MyGromacs -v /home/dgxsa/chengyi/share:/data -it nvcr.io/hpc/gromacs:2016.4 /bin/bash
=====
== GROMACS ==
=====

Container image Copyright (c) 2017, NVIDIA CORPORATION. All rights reserved.

This container contains several software products that have described in
academic publications. Below is a list of the software products include
in this container.

GROMACS      --

Various files include modifications (c) NVIDIA CORPORATION. All rights reserved.
NVIDIA modifications are covered by the license terms that apply to the underlying project or file.

root@408eaf27213c:/workspace#

```

启动容器以后, 就可以像在一台 Linux 服务器上操作了, 里面已经配置好了所有运行环境, 如果需要安装其他软件, 可以使用命令: apt-get install xxxx 进行安装。

如果想退出容器, 可以使用命令: Ctrl+D

如果想删除容器, 可以使用命令: nvidia-docker rm 408eaf27213c ; 408eaf27213c 为 docker-ID

如果想退出容器登录界面, 但保持容器后台运行, 可以使用命令: Ctrl+P, 然后 Ctrl+Q



## 4 标准算例

### 4.1 准备算例

使用普通用户运行：

```
su - test
mkdir gmx-test
cd gmx-test
wget http://www.gromacs.org/@api/deki/files/128/=gromacs-gpubench-
dhfr.tar.gz #或者上传算例包 gromacs-gpubench-dhfr.tar.gz 到此
tar zxvf gromacs-gpubench-dhfr.tar.gz
cd dhfr
source /public/software/profile.d/openmpi-gnu-env.sh
source /public/software/profile.d/gromacs-2018.3-env.sh #加载环境变量
```

OpenMPI 的环境变量配置文件：

```
MPI_HOME=/home/chengyi/software/openmpi-3.1
export PATH=${MPI_HOME}/bin:$PATH
export LD_LIBRARY_PATH=${MPI_HOME}/lib:$LD_LIBRARY_PATH
export C_INCLUDE_PATH=${MPI_HOME}/include:$C_INCLUDE_PATH
export MANPATH=${MPI_HOME}/share/man:$MANPATH
```

gromacs-2018.3-env.sh 的配置文件：

```
GMX_HOME=/mnt/gromacs/gromacs-2018.3/buildgpu
export PATH=${GMX_HOME}/bin:$PATH
export LD_LIBRARY_PATH=${GMX_HOME}/lib:$LD_LIBRARY_PATH
```

### 4.2 手动运行算例

运行命令如下：

```
cd dhfr/GPU/dhfr-solv-PME.bench/
gmx_mpi grompp -f gpu-PME.mdp
mpirun --allow-run-as-root -np 8 gmx_mpi mdrun -ntomp 5 -nb gpu -nsteps
1000 -s topol.tpr
或：
mpirun --allow-run-as-root -np 2 gmx_mpi mdrun -ntomp 5 -gpu_id 0,1 -
nsteps 1000 -s topol.tpr
```

ntomp:表示每个 cpu 进程启动的 OpenMP 线程数

gpu\_id:表示选择的 gpu, 本例为 gpu\_id 后面 0, 1 表示使用 GPU ID 为 0,1 的 GPU; 若使用所有 gpu, 使用-nb gpu。

本例表示：启动 8 个 cpu 进程，每个 cpu 进程启动 5 个线程。

运行 nvidia-smi 会看到启动了八个 gpu 进程。

testverlet:是这个算例的需要。

-maxh: 最大允许的计算时间

-g: 输出的日志文件。

更多的参数可以运行 gmx\_mpi mdrun -h 查阅。

Gromacs 运行 shell 脚本如下, 黄色部分是需要修改的地方。

```
#!/bin/bash

#export CUDA_VISIBLE_DEVICES=0,1,2,3
DIR_ROOT=/mnt/software/gromacs2018
source $DIR_ROOT/bin/GMXRC

IDIR=/mnt/test/water-cut1.0_GMX50_bare/1536
IFILE=$IDIR/topol.tpr

ODIR=/tmp
LDIR=$ODIR/run/logs
LFILE_ROOT=1536
WDIR=$ODIR/run/job

mkdir -p $WDIR
mkdir -p $LDIR

cd $WDIR

NSTEPS=10000
#NGPUS=`nvidia-smi | grep Tesla | wc -l`
NGPUS=4

LASTCORE=$(lscpu | grep On-line | cut -f3 -d"-")
THREADSPER=$(lscpu | grep Thread | cut -f2 -d":")
export OMP_NUM_THREADS=$(( ( (LASTCORE+1)/NGPUS) / THREADSPER) ))

LOGFILE=${LDIR}/${LFILE_ROOT}.${NGPUS}.log

# initialize if needed
if [ ! -f ${IFILE} ] ; then
  cd ${IDIR}
  gmx_mpi grompp -f pme.mdp
fi

mpirun --allow-run-as-root -bind-to none -np ${NGPUS} gmx_mpi mdrun -g
${LOGFILE} -pin on -rethway -v -noconfout -nsteps ${NSTEPS} -s ${IFILE}
```

## 4.3 PBS 脚本范例

```
su - test
cd gmx-test/dhfr/GPU/dhfr-solv-PME.bench/
    建立提交脚本 gromacs-gpu-single.pbs
```

```
#!/bin/bash
#PBS -N gromacs-gpu-single
#PBS -q gpu
#PBS -l nodes=node-with-gpu:ppn=4
#PBS -l walltime=100:00:00

source /public/software/profile.d/openmpi-gnu-env.sh
source /public/software/profile.d/gromacs-2018.1-env.sh

cd $PBS_O_WORKDIR
NP=`cat $PBS_NODEFILE|wc -l`
NN=`cat $PBS_NODEFILE|uniq -c|wc -l` #节点数
PPN=`cat $PBS_NODEFILE|uniq -c|sed -n '1,1p'|awk '{print $1}'`#每个节点的进
程数, 本例中 PPN=4
GPUID=`printf %.${PPN}d 0` #本例中 GPUID=0000
mpirun -np $NP -hostfile $PBS_NODEFILE gmx_mpi mdrun -multi $NN -nb gpu -g
md.log -nsteps 1000 -s topol.tpr
    提交作业, 并查看是否正确运行。
```

```
qsub gromacs-gpu-single.pbs
qstat
```

## 4.4 Slurm 脚本范例

```
su - test
cd gmx-test/dhfr/GPU/dhfr-solv-PME.bench/
    建立提交脚本 batch.gromacs-gpu-single
```

```
#!/bin/bash
#SBATCH --job-name=gromacs_test
#SBATCH --nodes=1 # number of Nodes
#SBATCH --tasks-per-node=32 # number of MPI processes per node
#SBATCH --gres=gpu:2
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=8
#SBATCH --mem-per-cpu=4000 # memory limit per CPU (megabytes)
#SBATCH --time=24:00:00 # time limit (D-HH:MM:ss)

module load gcc/6.4.0 openmpi/2.1.1 gromacs/2018.3
export OMP_NUM_THREADS="${SLURM_CPUS_PER_TASK:-1}"

srun gmx_mpi mdrun -deffnm md -maxh 24
    提交作业, 并查看是否正确运行。
```

## 5 Benchmark 测试

测试算例 water-cut1.0\_GMX50\_bare/1536, 共 1.536M 个水分子, 4.608M 个原子, NSTEPS=10000。

NVIDIA DGX-1 服务器配置:

2\*Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz

8\*Tesla V100-SXM2

512GB DDR4 Memory

CPU 进程数*每进程的线程数+GPU 进程数	Performance(ns/day)	Walltime(s)	加速比
2 × 5+0	0.703	1229.269	5.13
2 × 5+2	3.550	243.452	
4 × 5+0	1.139	758.762	3.92
4 × 5+4	4.469	193.392	
8 × 5+0	2.322	372.106	3.51
8 × 5+8	8.147	106.076	

GPU 的加速效果分析: 相同的 CPU 计算核数时的性能相比, GPU 可以可以显著提升计算性能。

