## Amber20 update 9

### ウェブページ

http://ambermd.org/

### バージョン

Amber20 update 9, AmberTools 20 update 15

### ビルド環境

- Intel Parallel Studio 2017 Update8（MPI only)
- GCC 7.3.1 (devtoolset-7)
- CUDA 11.1 Update 1

### ビルドに必要なファイル

- Amber20.tar.bz2
- AmberTools20.tar.bz2
- (Amber20 update.1-9 & AmberTools20 update.1-15; スクリプト内で取得)

### ビルド手順

```
#/bin/sh

VERSION=20
TOOLSVERSION=20

INSTALL_DIR="/local/apl/lx/amber20-up9"
TARBALL_DIR="/home/users/${USER}/Software/AMBER/20"

PARALLEL=12

#------------------------------------------------------------------------
module purge
module load intel_parallelstudio/2017update8
module load scl/devtoolset-7
module load cuda/11.1

export AMBERHOME=${INSTALL_DIR}
export CUDA_HOME="/local/apl/lx/cuda-11.1"

export LANG=C
export LC_ALL=C

# install directory has to be prepared before running this script
if [ ! -d ${AMBERHOME} ]; then
  echo "Create ${AMBERHOME} before running this script."
  exit 1
fi

# the install directory must be empty
if [ "$(ls -A ${AMBERHOME})" ]; then
  echo "Target directory ${AMBERHOME} not empty"
  exit 2
fi

ulimit -s unlimited

# prep files
cd ${AMBERHOME}
bunzip2 -c ${TARBALL_DIR}/Amber${VERSION}.tar.bz2 | tar xf -
bunzip2 -c ${TARBALL_DIR}/AmberTools${TOOLSVERSION}.tar.bz2 | tar xf -
```

```
mv amber${VERSION}_src/* .
rmdir amber${VERSION}_src

# install python first. otherwise, update_amber failed to connect ambermd.org
./AmberTools/src/configure_python
AMBER_PYTHON=$AMBERHOME/bin/amber.python

# apply patches and update AmberTools
echo y | $AMBER_PYTHON ./update_amber --upgrade
$AMBER_PYTHON ./update_amber --update

echo "[GPU serial edition (two versions)]"
LANG=C ./configure --no-updates -cuda gnu
make -j${PARALLEL} install && make clean

echo "[GPU parallel edition (two versions)]"
LANG=C ./configure --no-updates -mpi -cuda gnu
make -j${PARALLEL} install && make clean

echo "[CPU serial edition]"
LANG=C ./configure --no-updates gnu
make -j${PARALLEL} install && make clean

echo "[CPU openmp edition]"
LANG=C ./configure --no-updates -openmp gnu
make -j${PARALLEL} install && make clean

echo "[CPU parallel edition]"
LANG=C ./configure --no-updates -mpi gnu
make -j${PARALLEL} install && make clean

# run tests
. ${AMBERHOME}/amber.sh
cd ${AMBERHOME}

# parallel tests first
export DO_PARALLEL="mpirun -np 2"

make test.parallel && make clean.test
make test.cuda_parallel && make clean.test # DPFP
cd test; ./test_amber_cuda_parallel.sh SPFP; make clean; cd ../

export DO_PARALLEL="mpirun -np 4"
cd test; make test.parallel.4proc; make clean; cd ../

unset DO_PARALLEL

# openmp tests
make test.openmp && make clean.test

# serial tests
make test.serial && make clean.test
make test.cuda_serial && make clean.test # DPFP
cd test; ./test_amber_cuda_serial.sh SPFP; make clean; cd ../

cd ${AMBERHOME}
chmod 700 src
```

## テスト

- テストは軽微な数値エラーは見られるものの特に問題は無し
- (前回(update0)と異なり、今回のバージョンでは CUDA 版の GAMD では問題は発生していない。修正に感謝！)

## メモ

- ccgpup でビルド
  - ccgpup から外部サイトへの http アクセスが許可されたため、update の download からテストまで全て ccgpup で実行しています。
- 前回(update0)と比べ、速度面での明確な変化は見られず。(V100 でわずかに(< 1%)速度が落ちているかもしれない)
- P100 では相変わらず amber18 の方が速度が出ている。
- configure 版の python3 については未検証

## (cmake 版テスト時のメモ)

- (実際に導入したのは上記の configure を使ったものです。cmake 版は下記 pbsa_cuda_cg テストのエラーのため、利用を見合わせました。)
- miniconda を python3 にすると初期チェックは通るものの、ビルドの途中で python2 が呼び出されてエラーとなったため python2 のみでテスト
- configure 時とは少しビルド方式も違うように見える
  - configure で cuda 有効の場合、sander API は無効化されるが、cmake 利用時には無効化されていないように見える
  - 他にも違いがあるかもしれない
- cmake でビルドした時のみ test_at_cuda の pbsa_cuda_cg のテストで問題。(configure を使ったものでは問題起こらず)
  - EPB の値が大きくずれていて、Iterations required の表示が 0 だったり、Convergence achieved の数字も明らかに不自然。(下記参照; bc5, bc10 についても同様のエラー)
  - pbsa_cuda_cg 以外のテストについては全て問題無し。
  - gcc7+cuda-11.1, gcc7-cuda10.1, gcc6+cuda-9.1 で試したが全て同様の結果(gcc-8 やインテルコンパイラでは未検証)
  - (複雑なので一見しただけでは問題の原因わからず。)
    - 少なくとも上記の sander API は関係無い(cmake 時に -DBUILD_SANDER_API=FALSE 付加しても結果変わらず)
    - -DBLA_VENDOR=Generic でも変わらず。(何もつけない場合は openblas)
    - cuda 関連ライブラリの選択ミスが発生している？

```
possible FAILURE: (ignored)  check mdout.1a93_B.p22.min_bc2.dif
/local/apl/lx/amber20-up9/AmberTools/test/pbsa_cuda_cg
929c929
<    Iterations required       :   220
>    Iterations required       :    0
931c931
<    Norm of the residual vector:     0.1317175924778
>    Norm of the residual vector:     0.
932c932
<    Convergence achieved      : 9.6113021224978365E-5
>    Convergence achieved      : 1.1278324955777715E-44
933c933
<  Total surface charge     -1.9750
>  Total surface charge     -2.0000
934c934
<  Reaction field energy   -982.8577
>  Reaction field energy   -805.5799
936c936
<  Etot   =    -2949.8108  EKtot   =       0.  EPtot     =       0.
>  Etot   =    -2772.5330  EKtot   =       0.  EPtot     =       0.
936c936
<  Etot   =    -2949.8108  EKtot   =       0.  EPtot     =       0.
>  Etot   =    -2772.5330  EKtot   =       0.  EPtot     =       0.
939c939
<  EELEC  =    -1786.2464  EPB    =    -982.8577  RESTRAINT  =       0.
>  EELEC  =    -1786.2464  EPB    =    -805.5799  RESTRAINT  =       0.
945c945
<  Etot   =    -2949.8108  EKtot   =       0.  EPtot     =       0.
>  Etot   =    -2772.5330  EKtot   =       0.  EPtot     =       0.
948c948
<  EELEC  =    -1786.2464  EPB    =    -982.8577  RESTRAINT  =       0.
>  EELEC  =    -1786.2464  EPB    =    -805.5799  RESTRAINT  =       0.
### Maximum absolute error in matching lines = 2.20e+02 at line 929 field 4
### Maximum relative error in matching lines = 8.52e+39 at line 932 field 4
```

GPU シリアル版用の cmake のフラグ:
(AMBER_PYTHON は miniconda でインストールした python を指します(cmake 前に導入), CUDA_HOME は /local/apl/lx/cuda-(バージョン))

```
cmake .. \
    -DCOMPILER=GNU \
    -DMPI=FALSE \
    -DCUDA=TRUE \
    -DINSTALL_TESTS=FALSE \
    -DDOWNLOAD_MINICONDA=FALSE \
    -DPYTHON_EXECUTABLE=${AMBER_PYTHON} \
    -DCUDA_TOOLKIT_ROOT_DIR=${CUDA_HOME} \
    -DCHECK_UPDATES=FALSE
```