Amber20 update 12

ウェブページ

http://ambermd.org/

バージョン

Amber20 update 12, AmberTools 21 update 11

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.1-12 & AmberTools20 update.1-15 & AmberTools21 update.1-11; スクリプト内で取得)
- patch-nmrat-gpu.cpp
    - http://archive.ambermd.org/202110/0206.html 参照。公式アップデートではパッチ確認できず

```
--- src/pmemd/src/cuda/gpu.cpp.org      2022-01-06 16:02:15.915217989 +0900
+++ src/pmemd/src/cuda/gpu.cpp  2022-01-06 16:02:26.857121731 +0900
@@ -2849,7 +2849,7 @@
     }
     // torsions, resttype = 3
     else if (resttype[i] == 3) {
-      if (nmrat[i][0] >= 0 && nmrat[i][1] >= 0 && nmrat[i][2] >= 0 && nmrat[3] >= 0) {
+      if (nmrat[i][0] >= 0 && nmrat[i][1] >= 0 && nmrat[i][2] >= 0 && nmrat[i][3] >= 0) {
         torsions++;
       }
       else {
```

- patch-cpptraj-configure
    - PNetCDF を強制的に off にするためのパッチ

```
--- AmberTools/src/cpptraj/configure.org        2022-01-13 22:56:46.000000000 +0900
+++ AmberTools/src/cpptraj/configure    2022-01-13 22:57:10.000000000 +0900
@@ -1771,8 +1771,6 @@
   if [ "${LIB_STAT[$LPARANC]}" != 'off' -a $USE_MPI -eq 0 ] ; then
     WrnMsg "Parallel NetCDF enabled but MPI not specified. Assuming '-mpi'."
     USE_MPI=1
-  elif [ $USE_MPI -ne 0 -a "${LIB_STAT[$LPARANC]}" = 'off' ] ; then
-    LIB_STAT[$LPARANC]='enabled'
   fi
   # If we are using the bundled ARPACK then we will need C++/Fortran linking.
   if [ "${LIB_STAT[$LARPACK]}" = 'bundled' ] ; then
```

- patch-configure_python
    - miniconda のかわりに miniforge を使うためのパッチ

```
--- AmberTools/src/configure_python.org 2022-01-12 15:46:09.042775250 +0900
+++ AmberTools/src/configure_python     2022-01-12 15:48:09.177986821 +0900
@@ -107,8 +107,7 @@

 echo "Downloading the latest Miniconda distribution"
 if [ $mac -eq 1 ]; then
-    curl -L -# https://repo.continuum.io/miniconda/Miniconda${version}-${MINICONDA_VERSION}-MacOSX-x86_64.sh > \
-              miniconda.sh
+    :
 else
```

```
     # Try to figure out if the machine builds 32- or 64-bit binaries,
     # respectively.
@@ -145,23 +144,23 @@
             ;;
     esac
     if [ $bit -eq 32 ]; then
-        wget https://repo.continuum.io/miniconda/Miniconda${version}-${MINICONDA_VERSION}-Linux-x86.sh \
-            -O miniconda.sh
+        exit 0
     else
-        wget https://repo.continuum.io/miniconda/Miniconda${version}-${MINICONDA_VERSION}-Linux-x86_64.sh \
-            -O miniconda.sh
+        wget https://github.com/conda-forge/miniforge/releases/${MINICONDA_VERSION}/download/Miniforge${version}-Linux-x86_64.sh \
+            -O miniforge.sh
     fi
 fi

-if [ -d "$prefix/miniconda" ]; then
-    echo "Deleting existing miniconda at $prefix/miniconda"
-    /bin/rm -fr "$prefix/miniconda"
+if [ -d "$prefix/miniforge" ]; then
+    echo "Deleting existing miniforge at $prefix/miniforge"
+    /bin/rm -fr "$prefix/miniforge"
 fi

 echo "Installing Miniconda Python."
-bash miniconda.sh -b -p "$prefix/miniconda" || error "Installing miniconda failed"
+bash miniforge.sh -b -p "$prefix/miniforge" || error "Installing miniconda failed"
+ln -s ./miniforge ./miniconda

-export PATH="$prefix/miniconda/bin":"$PATH"
+export PATH="$prefix/miniforge/bin":"$PATH"
 echo "Updating and installing required and optional packages..."

 $prefix/miniconda/bin/python -m pip install pip --upgrade
@@ -172,7 +171,7 @@
 # Use pip to install matplotlib so we don't have to pull in the entire Qt
 # dependency. And cache inside the Miniconda directory, since we don't want to
 # be writing outside $AMBERHOME unless specifically requested to
-$prefix/miniconda/bin/python -m pip --cache-dir=$prefix/miniconda/pkgs \
+$prefix/miniconda/bin/python -m pip --cache-dir=$prefix/miniforge/pkgs \
     install matplotlib || install_matplotlib='yes'

 if [ ! -z "$install_matplotlib" ]; then
@@ -183,22 +182,22 @@
 mkdir -p $prefix/lib
 cwd=`pwd`
 cd "$prefix/bin"
-ln -sf ../miniconda/bin/python amber.python || error "Linking Amber's Miniconda Python"
-ln -sf ../miniconda/bin/conda amber.conda || error "Linking Amber's Miniconda conda"
-ln -sf ../miniconda/bin/ipython amber.ipython || error "Linking Amber's Miniconda ipython"
-ln -sf ../miniconda/bin/jupyter amber.jupyter || error "Linking Amber's Miniconda jupyter"
-ln -sf ../miniconda/bin/pip amber.pip || error "Linking Amber's Miniconda pip"
+ln -sf ../miniforge/bin/python amber.python || error "Linking Amber's Miniconda Python"
+ln -sf ../miniforge/bin/conda amber.conda || error "Linking Amber's Miniconda conda"
+ln -sf ../miniforge/bin/ipython amber.ipython || error "Linking Amber's Miniconda ipython"
+ln -sf ../miniforge/bin/jupyter amber.jupyter || error "Linking Amber's Miniconda jupyter"
+ln -sf ../miniforge/bin/pip amber.pip || error "Linking Amber's Miniconda pip"
 cd "$prefix/lib"
-for dir in ../miniconda/lib/tcl*; do
+for dir in ../miniforge/lib/tcl*; do
     ln -sf "$dir" || error "Linking TCL library folder $dir"
 done
-for dir in ../miniconda/lib/tk*; do
+for dir in ../miniforge/lib/tk*; do
```

```
     ln -sf "$dir" || error "Linking TK library folder $dir"
 done
 cd $cwd
 echo ""
-echo "Done. Miniconda installed in $prefix/miniconda"
+echo "Done. Miniforge installed in $prefix/miniforge"

-/bin/rm -f miniconda.sh
+/bin/rm -f miniforge.sh

-$prefix/miniconda/bin/conda clean --all --yes
+$prefix/miniforge/bin/conda clean --all --yes
```

ビルド手順

```
#!/bin/sh

VERSION=20
TOOLSVERSION=20 # will be upgraded to 21

INSTALL_DIR="/local/apl/lx/amber20-up12"
TARBALL_DIR="/home/users/${USER}/Software/AMBER/20"

PATCH0=${TARBALL_DIR}/patch-nmrat-gpu.cpp
PATCH1=${TARBALL_DIR}/patch-cpptraj-configure
PATCHX=${TARBALL_DIR}/patch-configure_python

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

patch -p0 < $PATCHX
```

```
# install python first. otherwise, update_amber failed to connect ambermd.org
./AmberTools/src/configure_python -v 3
AMBER_PYTHON=$AMBERHOME/bin/amber.python

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

# patch
# see http://archive.ambermd.org/202110/0206.html
patch -p0 < $PATCH0
# ad hoc something for pnetcdf
patch -p0 < $PATCH1

# reaxff-puremd is openmp only (tentatively)
# quick is tentatively not enabled; libstdc++ related issue?
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 -mkl -reaxff-puremd-openmp -openmp gnu
make -j${PARALLEL} install && make clean

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

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

# ad hoc work-around
# https://github.com/Amber-MD/pdb4amber/issues/85#issuecomment-672812778
cd ${AMBERHOME}/lib/python*/site-packages
if [ -d ParmEd*/parmed ]; then
  ln -s ParmEd*/parmed ./parmed
fi
if [ -d pdb4amber*/pdb4amber ]; then
  ln -s pdb4amber*/pdb4amber ./pdb4amber
fi
if [ -d pytraj*/pytraj ]; then
  ln -s pytraj*/pytraj ./pytraj
fi

# parallel tests first
cd ${AMBERHOME}
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

- 結果は /local/apl/lx/amber20-up12/logs 以下にあります
- ほとんどは軽微な数値エラー
- test_at_serial において parmed 関連のエラーが２つ出ている。parmed の python パッケージの読み込みに失敗していることが原因？

メモ

- cmake でビルドした場合、pbsa_cuda_cg テストがエラーになるため今回も configure を選択
- miniconda のかわりに miniforge を利用
- Quick については実行時にライブラリ(おそらくは libstdc++)関連のエラーが出たため今回は回避
    - 詳細な検証は行っていません
- reaxff-puremd は試験的に OpenMP 版で有効に
- Python のファイル配置がどうもおかしいため一部手動で修正
    - https://github.com/Amber-MD/pdb4amber/issues/85#issuecomment-672812778
    - amber.python で import parmed や import pdb4amber が正常に実行できることまでは確認しています。
- nmrat の修正パッチは公式からは提供されていないものの、明らかに間違っていると思われるためパッチを適用
    - テスト結果に特筆すべきような影響は確認できていません
- gcc8 や gcc10 を使うと一部テストが通らなくなるため、今回も gcc7 を利用(amber20 系では共通の事象)
- GPU を使わない openmp 版と MPI 版では -mkl は問題無いようなので今回は有効に
- cpptraj の PNetCDF は強制的に無効化している。テスト結果には悪影響は出ていないように見える。
    - 上記パッチを適用しない場合は cpptraj.MPI 等のビルドに失敗してしまう
- (2022/6/1 更新) bin/parmed と bin/pdb4amber が正常に動いていなかったため更新。
    - 中身を以前のバージョンと同じ形に手動で書き換え。とりあえず動作はするようになった。