Amber20

ウェブページ

http://ambermd.org/

バージョン

Amber20 + AmberTools20-up3

ビルド環境

- Intel Parallel Studio 2017 Update8（MPI only)
- GCC 7.3.1 (devtoolset-7)
- CUDA 10.1.243

ビルドに必要なファイル

- Amber20.tar.bz2
- AmberTools20.tar.bz2
- (AmberTools20 update.1-3; 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-up0"
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/10.1

export AMBERHOME=${INSTALL_DIR}
export CUDA_HOME="/local/apl/lx/cuda-10.1"

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
# GPU tests will be done elsewhere
# ccgpup cannot access external network, ccfep doesn't have GPGPUs

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

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

echo "[CPU parallel edition]"
LANG=C ./configure --no-updates -mpi gnu
make -j${PARALLEL} install
export DO_PARALLEL="mpirun -np 2"
make test.parallel
export DO_PARALLEL="mpirun -np 4"
cd test && make test.parallel.4proc

cd $AMBERHOME
make clean && chmod 700 src
```

パフォーマンスに関するメモ

GPU

- P100(jobtype=gpup) では AMBER20-up0(gcc7 + CUDA-10.1) よりも AMBER18-bf16 (gcc4.8 + CUDA-9.1) の方がわずかに速いようです。(2-3 % 程度)
    - AMBER20-up0 を CUDA-9.1 でビルドすると CUDA-10.1 のバージョンよりも明確に遅くなります
    - GPU 版については gcc のバージョンの違いで大きく速度が変わるケースはこれまでのところはありません。
    - 公式メーリングリストの情報でも同様の事象が確認されています
- 対照的に、V100 (jobtype=gpuv) では現時点でも AMBER20-up0 の方が少し速そうです。(5 % 程度)
- センターでは確認できていませんが、Turing 世代の GPU では以前のバージョンに比べて 15% 程度の速度低下が報告されています。(公式メーリングリストの情報)

明示的に V100 を使う場合、Amber20 の新機能を使いたい場合だけは、Amber20-up0 を選択するべきかもしれません。
しかし、そうでないのなら、現状(2020/6 時点, Amber20-up0)では Amber18-bf16 を選択すべきかもしれません。
(なお、Amber20 については今後のパッチリリースによってパフォーマンスが大きく向上する可能性があります。)

CPU

- DHFR 系でテスト
  - CPU 版の pmemd.MPI については intel の方が明確に速い。intel17 と intel19 の間に明確な差は見出せなかった。
  - gcc 系ではバージョン 6-8 では大きな違いは見られず。一方で 4.8 は 6-8 より明らかに少し遅い。(5 は未検証)

メモ

- MPI+OpenMP バージョンの CPU 版 pmemd についてはスキップ
- update_amber の際に system 標準の python を使うとエラーとなる(何かのパッケージが足りない or 古い?)ため、amber の miniconda のものを拝借
  - システム標準の python3 ならば動作
  - anaconda2-2019Jul の python でも動作する
- gcc8 を使うと CUDA 版の GB 計算のテストが失敗する。1 ステップ目の EGB の時点でおかしい。
  - gcc6, gcc7 では発生せず。2 ステップ目以降のダイナミクスに影響がありそうな気配。
- GPU 版の GAMD 計算のテストでエラーが発生するケースがある。(シリアル、並列の両方)
  - ログに GAMD = ********** のような行が出力されたり、値が不審に 0.0 だったりする
  - これまでに把握できた限りでは、gcc のバージョンには依存しない模様
  - 確実に発生するわけでは無い。起こらない場合もありそう。また、ダイナミクスには影響無いようで、他のエネルギー等は正常に見える
    - cpu 側と gpu 側の出力用バッファの同期が甘い？もしくは reduction の処理に問題がある？良くわからない。
- インテルコンパイラー(17u8, 19u5 で検証)では test/dhfr の bussi のテスト(ntt=11; Bussi thermostat)で数値エラーが出る。2 ステップでの運動エネルギーの値が明確にずれる。
  - 熱浴の挙動が純粋におかしい可能性があるため、intel コンパイラは回避することにした。
- gcc7+mkl17 と intel17(+mkl)+gcc7 では追加で nab のテストで失敗する。数値エラー。(intel17 の mkl の影響? gcc + intel19 の mkl では未検証)
  - intel19(+mkl)+gcc8 では nab のテストは成功するが、他で一部 Program error 等が見られたため、今回は回避した。
  - gcc7 で mkl 使わない場合は問題無し。ひとまず mkl は外す。