Amber20 ユーザーマニュアル

目次

1 Amber20 について

Amber20 はカリフォルニア大学等のグループが開発する分子動力学シミュレーションプログラムのパッケージです。

Amber20 を使用するには開発元との間でライセンス契約が必要です。契約の内容は以下 URL をご参照下さい。

http://ambermd.org/LicenseAmber20.pdf

Amber20 のパッケージは ambermd.org のサイトからダウンロードして入手します。パッケージの入手方法等については契約時に開発元より E メール等で案内が届きますので、そちらをご参照下さい。

Amber の最新版である Amber20 のインストールをする際、AmberTools のパッケージが必要です。そこで、2021 年 4 月にリリースされた AmberTools21 も合わせてインストールをしてあります。AmberTools21 は ambermd.org のサイトの下記 URL でユーザー情報を登録することで無償使用することができます。

http://ambermd.org/GetAmber.php#ambertools

本マニュアルは Amber20 のインストール概要とジョブの実行方法等をご案内します。Amber20 の詳細については Amber20 の公式サイト（ http://ambermd.org/index.php ）をご覧下さい。

2 Amber20 インストール概要

Amber20 と AmberTools21 は以下ディレクトリにインストールをしています。

パッケージ	ディレクトリ
Amber20	/usr/local/Amber20
AmberTools21	/usr/local/Amber20/AmberTools

Amber20 と AmberTools21 はソースコードで配布されています。Amber20 のインストールの際は以下の開発環境を使用してビルドを行っています。

そのため、本文章にて解説している Amber20 の動作は、以下の開発環境の設定が有効になっている必要があります。

パッケージ	ディレクトリ
Intel C/C++/Fortran Classic Compiler Version 2021.2.0	/opt/intel/oneapi/compiler/2021.2.0
Intel MPI Version 2021.2.0	/opt/intel/oneapi/mpi/2021.2.0
CUDA 11.4[1]	/usr/local/cuda-11.4

開発陣の作成したソース付属のテストをシリアル、MPI 並列で実行し、公式テストの動作に問題が無い事を確認しました。

Amber20 の環境設定は Environment Module と呼ばれる環境設定ユーティリティ（module コマンド）を用いて行います。Amber20 用のモジュール定義ファイル（Environment Module 用設定ファイル）は、Amber20 と同時にインストールされております。動作確認の際に使用したテスト、およびその実行結果については、/opt/hpcs/app_doc 以下に PDF 文章がありますので、別途文章を確認ください[2]。

Environment Module を用いた Amber20 の設定方法については、4.1 節を参照ください。

[1] GPU 版のみ

[2] CPU 版：Amber20p12-AmberTools21p12-InteloneAPI202120-IMPI202120-20220228.pdf
GPU 版: Amber20p12-AmberTools21p12-InteloneAPI202120-IMPI202120-CUDA114-20220405.pdf

3 Amber20 コマンド

Amber20 のコマンドについては Amber20 公式サイトの以下 URL にマニュアルがありますので、詳しくはそちらをご覧下さい。

http://ambermd.org/doc12/Amber21.pdf

以下は主に使用するコマンドと、その説明について Amber20 公式サイトより抜粋したものです。

・**sander**

(part of AmberTools) is the basic energy minimizer and molecular dynamics program. This program relaxes the structure by iteratively moving the atoms down the energy gradient until a sufficiently low average gradient is obtained. The molecular dynamics portion generates configurations of the system by integrating Newtonian equations of motion. MD will sample more configurational space than minimization, and will allow the structure to cross over small potential energy barriers. Configurations may be saved at regular intervals during the simulation for later analysis, and basic free energy calculations using thermodynamic integration may be performed. More elaborate conformational searching and modeling MD studies can also be carried out using the sander module. This allows a variety of constraints to be added to the basic force field, and has been designed especially for the types of calculations involved in NMR, Xray or cryo-EM structure refinement.

・**pmemd**

(part of Amber) is a version of sander that is optimized for speed and for parallel scaling; the pmemd.cuda variant runs on GPUs. The name stands for "Particle Mesh Ewald Molecular Dynamics," but this code can now also carry out generalized Born simulations. The input and output have only a few changes from sander.

・**nab**

Stands for Nucleic Acid Builder. NAB is really a compiler that provides a convenient molecular programming language loosely based on C.

・**xleap**

A script that calls xaLeap with specific setup command-line arguments.

・**antechamber**

antechamber is the main program to develop force fields for small organic molecules (e.g., drugs, modified amino acids) using a version of the general Amber force field (GAFF). These can be used directly in LEaP, or can serve as a starting point for further parameter development.

・**cpptraj**

cpptraj is the main trajectory analysis utility (written in C++) for carrying out superpositions, extractions of coordinates, calculation of bond/angle/dihedral values, atomic positional fluctuations, correlation functions, analysis of hydrogen bonds, etc.

・**MMPBSA.py**

MMPBSA.py is a python script that automates energy analysis of snapshots from a molecular dynamics simulation using ideas generated from continuum solvent models. (There is also an older perl script, called mm_pbsa.pl, that has similar functionality.)

4 Amber20 実行例

4.1 Amber20 の環境設定方法

Amber20 用モジュール定義ファイルは、以下の場所に置かれています。

/home/.common/modulefiles/oneAPI/2021.2.0/amber20/AmberTools21p12

環境設定方法を以下に記します。

(1) 使用できるモジュールの一覧表示：module avail

module avail コマンドは module load コマンドで使用できるモジュールの一覧を表示します。
具体的な実行例を次に示します。

```
$ module␣avail
---------- /home/.common/modulefiles/oneAPI/2021.2.0 ------------
amber20/AmberTools21p12
-------------- /opt/intel/oneapi/modulefiles-HPCS ---------------
advisor/2021.2.0      compiler-rt32/2021.2.0  debugger/10.1.1
…(後略)…
```

赤文字で示した通り、amber20/AmberTools21p12 というモジュールが使用可能になっています。

(2) モジュールの有効化/無効化：module load / module unload

モジュールを有効にするコマンドは module load です。実行方法は、コマンドプロンプトにおいて、"module load モジュール名" です。
Amber20 用モジュールを有効化する場合、具体的なコマンドは次となります。

```
# module␣load␣AmberTools21p12
```

逆に、有効化したモジュールを無効化するコマンドは module unload です。実行方法は、コマンドプロンプトにおいて、"module unload モジュール名" です。

有効化した Amber20 用モジュールを無効化する場合、具体的なコマンドは次となります。

```
# module␣unload␣AmberTools21p12
```

module unload は、有効化した Amber20 用モジュールによる環境設定が、別の作業時に悪影響を及ぼす際に使用します。有効化した Amber20 用モジュールによる環境設定が、特に悪影響を及ぼしていない場合には、無理に使用する必要はありません。

正常に Amber20 用モジュールを読み込めたどうかについては、以下で説明する module list コマンドで確認できます。

(3) Environment Module で有効になっているモジュールの表示：module list

現在 Environment Module で有効になっているモジュールの一覧を表示するコマンドは module list です。具体的なコマンドを次に示します。

```
# module␣list
```

Amber20 の設定が有効になっている場合、以下のように、Amber20 の項目が表示されます。

```
# module␣list
Currently Loaded Modulefiles:
 1) tbb/2021.2.0          4) compiler/2021.2.0   7) amber20/AmberTools21p12
 2) debugger/10.1.1       5) mkl/2021.2.0
 3) compiler-rt/2021.2.0  6) mpi/2021.2.0
```

赤文字で示した通り、amber20/AmberTools21p12 が有効になっています。

(4) ユーザーシェルログイン時に Amber20 用モジュールを自動的に有効化する方法

お客様環境において、ユーザーログイン時に、自動的に Amber20 用モジュールの有効化を行いたい場合、ユーザーのシェル環境設定ファイルに `module load` コマンドを追記する必要があります。具体的な修正内容は以下のようになります。

- ☞ `bash` をお使いの場合
  ホームディレクトリの `.bashrc` の最終行に以下の追記を行います。

- ☞ `tcsh` をお使いの場合
  ホームディレクトリの `.cshrc` の最終行に以下の追記を行います。

```
# -------------- 任意のコメント ----------------------
# module␣load␣AmberTools21p12
```

Amber20 用モジュールの有効化が他のアプリケーションに悪影響を及ぼすようなケースも存在いたしますので、ログイン時のシェル環境における自動有効化については、有効時の影響に十分注意した上で行うようお願いいたします。

4.2 シリアル(1CPU コア)の実行例

以降、Amber20 の機能のうち、sander、pmemd のバイナリ（コマンド）について、実行方法を説明します。

Amber20 のマニュアルに記載されている sander 実行時のオプションは以下の通りです。
これらのオプションは pmemd でも有効となります。

```
sander [-help] [-O] [-A] -i mdin -o mdout -p prmtop -c inpcrd -r restrt
-ref refc -mtmd mtmd -x mdcrd -y inptraj -v mdvel -frc mdfrc -e mden
-inf mdinfo -radii radii -cpin cpin -cpout cpout -cprestrt cprestrt
-cein cein -ceout ceout -cerestrt cerestrt -evbin evbin -suffix suffix
```

具体的な実行例は以下のようになります。

sander の実行例

```
$ sander␣-O␣-i␣mdin␣-o␣mdout␣-p␣prmtop␣-c␣inpcrd␣-r␣restrt
```

pmemd の実行例

```
$ pmemd␣-O␣-i␣mdin␣-o␣mdout␣-p␣prmtop␣-c␣inpcrd␣-r␣restrt
```

上記の実行例でのオプションの意味は以下となります。

- -O 出力ファイルを全て上書き
- -i sander のオプションを記述した入力ファイル名を指定。実行例では "mdin"
- -o 出力ファイル名を指定。実行例では、"mdout"
- -p パラメーター／トポロジーのインプットファイル名を指定。実行例では "prmtop"
- -c 初期構造のインプットファイル名を指定。実行例では "inpcrd"
- -r 座標、速度等の最終計算結果の出力ファイル名を指定。実行例では "restrt"

sander、pmemd 以外のコマンドの使用方法や、本文章において解説を行っていない sander、pmemd の実行オプションに関する詳細情報は、Amber20 の公式マニュアルを参照ください。

4.3 MPI 並列の実行例

実行ファイルに .MPI の拡張子があるものは MPI での並列実行が可能です。-np 以降に並列数を指定します。以下は 16CPU コアでの例です。MPI 並列での実行時のオプションはシリアルと同様です。

以下の実行例内の環境変数 AMBERHOME は、Amber20 用モジュール amber20/AmberTools21p12 を有効化（module load）することによって設定されます。

sander.MPI の実行例

```
$ mpirun␣-np␣16␣$AMBERHOME/bin/sander.MPI␣-O␣-i␣mdin␣-o␣mdout␣-p␣prmtop
␣-c␣inpcrd -r␣restrt
```

pmemd.MPI の実行例

```
$ mpirun␣-np␣16␣$AMBERHOME/bin/pmemd.MPI␣-O␣-i␣mdin␣-o␣mdout␣-p␣prmtop␣
-c␣inpcrd␣-r␣restrt
```

※ mpirun 以降の実行ファイルは絶対パスで指定する必要があります。実行ファイルがあるディレクトリにパスが通っていても同様です。

4.4 GPU 版バイナリの実行例（ GPU 版バイナリ同梱時のみ ）

Amber20 には、GPU を計算コアとして使用することができる機能が存在します。
この GPU で計算を行うバイナリは CPU 版バイナリとは異なる名前の、別バイナリとなります。
また、GPU バイナリは計算できる条件（対応している計算）が存在しています。そのため、CPU 版バイナリで実行できるすべてのインプットが計算できるわけではありません。詳細については公式マニュアルを参照ください。

Amber20 の GPU 版バイナリは、数値精度が異なる 2 種類のバイナリが存在します。
以下の記述は Amber20 公式マニュアルからの抜粋となります。

・ SPFP - (Default) Use a combination of single precision for calculation and fixed precision for accumulation. This approach is believed to provide the optimum tradeoff between accuracy and peformance and hence at the time of release is the default model invoked when using the executable pmemd.cuda.

・ DPFP - Use double precision (and double precision equivalent fixed precision) for the entire calculation. This provides for careful regression testing against the CPU code. It makes no additional approximations above and beyond the CPU implementation and would be the model of choice if performance was not a consideration. On v2.0 NVIDIA hardware (e.g. M2090) the performance is approximately half that of the SPFP model while on v3.0 NVIDIA hardware (e.g. K10) the performance is substantially less than the SPFP model.

GPU 版バイナリの具体的ファイル名は、以下の通りです。

`pmemd.cuda`、`pmemd.cuda_SPFP`： Serial 版 SPFP 精度 pmemd バイナリ。
（この２つのバイナリは全く同一となります）
`pmemd.cuda_DPFP`：Serial 版 DPFP 精度 pmemd バイナリ。

`pmemd.cuda_SPFP.MPI`：Parallel 版 SPFP 精度 pmemd バイナリ。
`pmemd.cuda_DPFP.MPI`：Parallel 版 DPFP 精度 pmemd バイナリ。

上記のバイナリは、4.2 節、4.3 節で解説した CPU 版バイナリ実行例のそれぞれのバイナリ名（ Serial 版：pmemd、Parallel 版:pmemd.MPI ）を、該当するバイナリ名に変更することで実行できます。

5 既知の問題点

5.1 テスト Run.rxsgld 8 並列時の挙動について

２章で述べた通り、Amber20 の正常動作の確認は、開発陣が用意し、公式マニュアルに記載されている公式テスト（ `make test.serial` 等 ）を用いて行います。このテストは、並列数、GPU 使用の有無等、テスト条件に応じてテスト内容が異なりますが、このうち、CPU 8 並列テスト実行時に、テスト “`rxsgld_4rep`”が、いつまでも計算が終わらない状況になる、という異常挙動が発生する事が確認されております。
この問題について調査を行ったところ、Amber18 の公式テスト結果の解説ページ、
http://ambermd.org/pmwiki/pmwiki.php/Main/Amber18Test にて、以下の記述があることが判明しました。

Amber MPI 8 thread*

239 file comparisons passed

6 file comparisons failed

*1 tests experienced an error

The tests actually halt at the rxsgld_4rep - pmemd.MPI segfaults and then hangs, forcing you to use Ctrl-C. Note sander.MPI does not have this issue. Had to run tests with rxsgld_4rep commented out.

記載されている状況は、今回の Amber20 での結果と同じです。これに対する開発者のコメントとして、" test 実行箇所をコメントアウトして対応" という、問題解決ではなく、テスト自体を回避する方法での対策が提示されています。

今回、Amber20 を検証するにあたって、公式の Mailing List 等を調べましたが、（Amber20 への）アップデートにおいて、この `rxsgld_4rep` について修正された、という記述が一切見つかりませんでした。そのため、Amber18 と同様の問題が発生してしまっていると推測しています。

以上の状況から、`rxsgld_4rep` の 8 並列時の異常挙動については、開発者も認識している既知の問題と判断し、問題ないと判断いたしました。

5.2 GPU 版テスト結果について

GPU 版バイナリについても、Amber20 の公式テストを実行しておりますが、前節で解説したテスト “ rxsgld_4rep ”同様に、テスト“ Run.neb_explicit ” において GPU4 並列実行時に異常挙動が発生しエラーで終了する事が判明しております。

このテストですが、Error 発生時にログファイルで以下のような出力がされております。

“ gpu_neighbor_list_setup :: Small box detected, with <= 2 cells in one or more dimensions. The current GPU code has been deemed unsafe for these situations. Please alter the cutoff to increase the number of hash cells, make use of the CPU code, or (if absolutely necessary) run pmemd.cuda with the -AllowSmallBox flag. This behavior will be corrected in a forthcoming patch.”

これは、計算途中にセルの大きさが閾値より下回ったために計算を終了させる、
という内容であるため、バイナリの動作自体は正常で、計算結果が閾値に引っかかってしまったために発生していると判断されます。

弊社の知見では、このような症状の場合、バイナリが正常で、テストインプットに問題がある場合が多いと認識しております。テストインプットに問題がある場合、具体的なケースとしては、

- そもそもテストインプットに誤りがあり、正常に計算できない場合
- アプリの VersionUp による仕様変更等のため、そもそもテストに適さなくなった場合
- テストインプットの計算結果が非常に環境に敏感で、テスト環境のハードウェア構成やビルド時の環境（Compiler、CompilerOption 等）の差異で正誤が変化してしまう場合

のように分類されます。

本テストについてエラーの原因を調べるため、まず公式のメーリングリスト等において、
調査を行いましたが、該当インプットについてはほとんど情報が無く、詳細な情報が得られませんでした。

そこで、GPU 版 Amber20 について、Compiler 変更、Compiler Option の修正、MPI の変更といったソフトウェア開発環境の変更を試みましたが、残念ながら改善が見られませんでした。2022/4 時点において、情報の不足も伴い、弊社の検証環境において正常にテストを PASS するバイナリを作成する開発環境、CompilerOption の組み合わせは見つかっておりません。

以上の経緯により、現時点では GPU 版 Amber20 において、これ以上の改善は難しいと判断し、出荷版として採用しております。

付録 A

A.1 HPC システムズ お問い合わせ先


弊社ホームページ　http://www.hpc.co.jp/support_index.html

サポート案内やお問い合わせの多い内容など様々な情報を掲載しております。
是非ご活用ください。

HPC システムズ株式会社
〒108-0022　東京都港区海岸 3-9-15 LOOP-X 8 階

HPC 事業部


【営業】03-5446-5531　【サポート】03-5446-5532

お電話によるサポート受付は祝日、弊社指定休日を除く月曜日から金曜日の 9:30～17:30 とさせて頂きます。


【FAX】03-5446-5550


【電子メール】hpcs_support@hpc.co.jp