サンプルジョブの実行方法

最終更新: 2025/10/7 メニューに追加

はじめに

RCCS では共通領域に導入したアプリケーションについて、サンプルジョブスクリプトを用意しています。
RCCS で用意したものをそのまま使う場合はもちろん、自身のディレクトリに導入した別バージョンを使う際のテンプレートとしても利用できます。
以下では Gromacs と GAMESS を例にして、サンプルをどのように実行するのかを説明します。

なお、Gaussian については、専用の g16sub/g09sub を用いた方法をまずはご覧ください。
ORCA についても専用の osub コマンドが利用できますので、こちらの利用もご検討ください。(ORCA を利用するには別途事前登録が必要です。)

導入済みアプリケーションのリスト

いくつかの方法があります。

1. ウェブページ上の情報を参照する

パッケージプログラム一覧のページに掲載されている情報で確認できます。

2. module avail コマンドの出力を見る

以下のように出力されます。通常、ジョブとして投入するようなアプリについては以下で赤で示した /apl/modules/apl 内にあります。

```
[user@ccfep3 ~]$ module avail
---------------------- /home/users/qf7/modules/defaults ----------------------
2022  2024  2025

---------------------------- /apl/modules/oneapi ----------------------------
compiler-rt/2022.0.2  intelmpi/2021.5.1     mkl/2022.2.1    tbb/2021.10.0
compiler-rt/2022.2.1  intelmpi/2021.7.1     mkl/2023.0.0    tbb/2021.11
...
(中略)
...
------------------------------ /apl/modules/apl ------------------------------
abcluster/3.0        gromacs/2023.2-CUDA          nwchem/6.8
ABINIT-MP/v1r22      gromacs/2023.4               nwchem/7.0.2
ABINIT-MP/v2r4       gromacs/2023.4-CUDA          nwchem/7.2.2
ABINIT-MP/v2r8       gromacs/2023.5               nwchem/7.2.2-CUDA
amber/20u13          gromacs/2023.5-CUDA          nwchem/7.2.3
amber/22u1           gromacs/2024.2               nwchem/7.2.3-intel
amber/22u4           gromacs/2024.2-CUDA          openbabel/3.1.1
amber/24u1           gromacs/2024.4               openmolcas/v21.10
...
gromacs/2022.6       ntchem/2013.13.0.0/mpi       xtb/6.7.0
gromacs/2022.6-CUDA  ntchem/2013.13.0.0/mpiomp    xtb/6.7.1
gromacs/2023.2       ntchem/2013.13.0.0/serial

Key:
loaded  directory/  auto-loaded  default-version  modulepath
[user@ccfep3 ~]$
```

(module avail コマンドはキーボードの q を押すか、画面の下までスクロールさせると終了します。)

3. /apl を参照する

OS 非標準のアプリやライブラリについては、/apl に導入されています。そちらで直接確認することもできます。
/apl/(アプリケーション名)/(バージョンやリビジョン) のようなディレクトリ構成となっています。

```
[user@ccfep3 /apl]$ ls
ABINIT-MP     autoconf      cudnn      gsl        namd       orca
GRRM          autodock      cudss      hpc-x      nbo        pbs
LigandMPNN    autodock-gpu  cusparselt i-pi       nciplot    plumed
ProteinMPNN   autodock-vina dalton     imolpro    ninja      psi4
RFDiffusionAA bio           dftb+      julia      ntchem     qe
RFdiffusion   boost         dftd4      lammps     nvhpc      reactionplus
RoseTTAFold-AA censo        dirac      libtorch   nwchem     scalapack
abcluster     cmake         eigen      luscus     omegafold  siesta
aiida         colabfold     elsi       lustre-dev oneapi     tcl
alphafold     conda         ffmpeg     magma      openbabel  togl
amber         cp2k          gamess     modules    openblas   turbomole
aocc          crest         gaussian   molden     openmm     vmd
aocl          crystal       genesis    molpro     openmolcas xcrysden
apptainer     cuda          gromacs    mvapich    openmpi    xtb
[user@ccfep3 /apl]$ ls /apl/amber
20u13  22u1  22u4  24u1  24u3
[user@ccfep3 /apl]$ ls /apl/amber/20u13
amber.csh   build          configure  lib64      README     test
amber.sh    cmake          dat        logs       recipe_at  update_amber
AmberTools  CMakeLists.txt doc        Makefile   samples    updateutils
benchmarks  cmake-packaging include   miniconda  share      wget-log
bin         config.h       lib        miniforge  src
[user@ccfep3 /apl]$
```

サンプルファイルの場所

各アプリ向けのサンプルファイルは原則として /apl/(アプリ名)/(バージョン)/samples 以下に置いています。

例: gromacs 2024.5 のサンプルを /apl から探す

```
[user@ccfep3 /apl]$ ls /apl
ABINIT-MP     autoconf      cudnn      gsl        namd       orca
GRRM          autodock      cudss      hpc-x      nbo        pbs
LigandMPNN    autodock-gpu  cusparselt i-pi       nciplot    plumed
ProteinMPNN   autodock-vina dalton     imolpro    ninja      psi4
RFDiffusionAA bio           dftb+      julia      ntchem     qe
RFdiffusion   boost         dftd4      lammps     nvhpc      reactionplus
RoseTTAFold-AA censo        dirac      libtorch   nwchem     scalapack
abcluster     cmake         eigen      luscus     omegafold  siesta
aiida         colabfold     elsi       lustre-dev oneapi     tcl
alphafold     conda         ffmpeg     magma      openbabel  togl
amber         cp2k          gamess     modules    openblas   turbomole
aocc          crest         gaussian   molden     openmm     vmd
aocl          crystal       genesis    molpro     openmolcas xcrysden
apptainer     cuda          gromacs    mvapich    openmpi    xtb
[user@ccfep3 /apl]$ ls /apl/gromacs/
2016.5       2021.6       2022.4      2023.2-CUDA  2024.2      2024.5-CUDA
2016.6       2021.6-CUDA  2022.4-CUDA 2023.4       2024.2-CUDA 2025.2
2020.6       2021.7       2022.6      2023.4-CUDA  2024.4      2025.2-CUDA
2021.4       2021.7-CUDA  2022.6-CUDA 2023.5       2024.4-CUDA
2021.4-CUDA  2021.7-mdtest 2023.2      2023.5-CUDA  2024.5
[user@ccfep3 /apl]$ ls /apl/gromacs/2024.5
bin  include  lib64  samples  share
[user@ccfep3 /apl]$ ls /apl/gromacs/2024.5/samples/
conf.gro  grompp.mdp     sample-mpi.sh       sample-threadmpi.sh
cp2k      sample-mpi.csh sample-threadmpi.csh topol.top
```

(-CUDA がついているものは GPU 版です。)

サンプルディレクトリ内のファイルについて

サンプルディレクトリ内でインプットそのものは原則一つです。
ただし、それを実行するためのジョブスクリプトについては複数ある場合があります。
例えば、シェルが違う場合、並列方式が違う場合、設定の読み込み方法が違う場合、GPU利用の有無などが挙げられます。

- 例1: sample.sh => /bin/sh を使うサンプル
- 例2: sample.csh => /bin/csh を使うサンプル
- 例3: sample-gpu.sh => /bin/sh を使い、GPU も利用するサンプル

必要に応じて中身を比較するなどしてください。

例: gamess 2022R2 の場合

実行スクリプトが３つあります(sample.csh, sample-module.sh, sample.sh)。それぞれシェルの種類や設定方法が違いますが、インプットは exam01.inp 一つです。

```
[user@ccfep4 ~]$ ls /apl/gamess/2022R2/samples/
exam01.inp  sample.csh  sample-module.sh  sample.sh
```

例: gromacs 2024.5 の場合

こちらもいくつか sample スクリプトがありますが、インプットファイルは 1 セットだけです。

```
[user@ccfep4 samples]$ ls
conf.gro  grompp.mdp      sample-mpi.sh        sample-threadmpi.sh
cp2k      sample-mpi.csh  sample-threadmpi.csh topol.top
```

- -mpi がついているものは Open MPI (HPC-X)を使った MPI 並列版です(マルチノード対応)
- -threadmpi がついているものは組み込みの thread MPI を使った並列版です(シングルノード用)
- (cp2k ディレクトリは倍精度版 gromacs と CP2K を組み合わせて QM/MM 計算を実行するサンプルです)

サンプルの実行: 一般的な手順

- まず、サンプルのファイルを自分の書き込める領域にコピーします。
- サンプルをコピーしたディレクトリで jsub sample.sh のように実行します。
- (大抵の場合は、sh ./sample.sh のような形でログインノードで実行可能にもなっています。)
  - ただし、CPU 数の指定が jsub 時と変わる場合があります
  - GPU を使うものについては ccfep 上では実行できません。ccfep から ccgpu にログインしてお試しください(ssh ccgpu コマンド)。
  - ccgpu では GPU を 2 枚導入しておりますので、MPI 並列テストも可能です。

例1: gamess 2022R2 の場合

サンプルを ~/gamess2022R2_test で実行する場合です。

```
[user@ccfep4 ~]$ mkdir -p ~/gamess2022R2_test
[user@ccfep4 ~]$ cd ~/gamess2022R2_test
[user@ccfep4 gamess2022R2_test]$ cp /apl/gamess2022R2/samples/* .
[user@ccfep4 gamess2022R2_test]$ ls
exam01.inp  sample-module.sh  sample.csh  sample.sh
[user@ccfep4 gamess2022R2_test]$ jsub sample.sh
4008689.cccms1
```

実行中のジョブの状況は jobinfo -c コマンドで確認できます。

```
[user@ccfep4 gamess2022R2_test]$ jobinfo -c
--------------------------------------------------------------------------------
Queue   Job ID Name            Status CPUs User/Grp       Elaps Node/(Reason)
--------------------------------------------------------------------------------
H       4008689 sample.sh      Run       4  user/---          -- ccc001
--------------------------------------------------------------------------------
[user@ccfep4 gamess2022R2_test]$
```

混雑していなければ、しばらく待てばジョブが終了し、出力結果が得られます。

```
[user@ccfep4 gamess2022R2_test]$ ls ~/gamess2022R2_test
exam01.dat  exam01.log  sample-module.sh  sample.sh.e4008689
```

```
exam01.inp  sample.csh  sample.sh        sample.sh.o4008689
[user@ccfep4 gamess2022R2_test]$
```

参考: sample.sh の中身について(青文字は説明文で、本来のファイルに含まれない部分です)

```
#!/bin/sh
#PBS -l select=1:ncpus=4:mpiprocs=4:ompthreads=1   # <= 4 コアのジョブ(1 vnode)
#PBS -l walltime=24:00:00   # <= 制限時間は 24 時間

if [ ! -z "${PBS_O_WORKDIR}" ]; then
  cd ${PBS_O_WORKDIR}  # <= ジョブを投入したディレクトリへ移動する。(jsub での投入時に必須の操作)
  NCPUS=$(wc -l < ${PBS_NODEFILE})
else
  NCPUS=4  # <= ここの二行は jsub を介さないで実行する時向けの設定
  export OMP_NUM_THREADS=1
fi

module -s purge
module -s load intelmpi/2021.7.1      #  <= 実行に必要な環境設定。アプリによって大きく変わります。
module -s load compiler-rt/2022.2.1

# processes per node; equal to mpiprocs value
PPN=4  # <= PPN = process per node, 一番上で定義されている mpiprocs の値と同じにする。

VERSION=2022R2
RUNGMS=/apl/gamess/${VERSION}/rungms
INPUT=exam01.inp

${RUNGMS} ${INPUT:r} 00 $NCPUS $PPN >& ${INPUT%.*}.log # <= 実際に実行するところ
```

例2: gromacs 2024.5 の場合

サンプルを ~/gromacs2024.5_test で実行する場合です。

```
[user@ccfep4 ~]$ mkdir -p ~/gromacs2024.5_test
[user@ccfep4 ~]$ cd ~/gromacs2024.5_test
[user@ccfep4 gromacs2024.5_test]$ cp /apl/gromacs/2024.5/samples/* .
[user@ccfep4 gromacs2024.5_test]$ ls
conf.gro  grompp.mdp      sample-mpi.sh        sample-threadmpi.sh
cp2k      sample-mpi.csh  sample-threadmpi.csh topol.top
[user@ccfep4 gromacs2024.5_test]$ jsub sample-mpi.sh
4008695.ccpbs1
```

実行中のジョブの状況は jobinfo -c コマンドで確認できます。

```
[user@ccfep3 gromacs2024.5_test]$ jobinfo -c
--------------------------------------------------------------------------------
Queue   Job ID Name            Status CPUs User/Grp       Elaps Node/(Reason)
--------------------------------------------------------------------------------
H       4008695 sample-mpi.sh  Run       6  user/---          -- ccc001
--------------------------------------------------------------------------------
```

混雑していなければ、しばらく待てばジョブが終了し、出力結果が得られます。

```
[user@ccfep4 gromacs2024.5_test]$ ls ~/gromacs2024.5_test
conf.gro      grompp.out       sample-mpi.sh           state.cpt
confout.gro  md.logmdout.mdp  sample-mpi.sh.e4008695  topol.top
cp2k          mdout.mdp        sample-mpi.sh.o4008695  topol.tpr
ener.edr      mdrun.out        sample-threadmpi.csh
grompp.mdp   sample-mpi.csh   sample-threadmpi.sh
[user@ccfep4 gromacs2024.5_test]$
```

参考: sample-mpi.sh の中身(青文字は説明文で、本来のファイルに含まれない部分です)

```
#!/bin/sh
#PBS -l select=1:ncpus=6:mpiprocs=6:ompthreads=1 # <= 6 コア(6 MPI ジョブ)
#PBS -l walltime=00:30:00  # <= 制限時間 30 分

if [ ! -z "${PBS_O_WORKDIR}" ]; then
 cd "${PBS_O_WORKDIR}" # <= ジョブを投入したディレクトリへ移動する。(jsub での投入時に必須の操作)
 NPROCS=$(wc -l < "${PBS_NODEFILE}")
else
 # when jsub is NOT used # <= jsub を介さないで実行する時向けの設定
 NPROCS=6
 export OMP_NUM_THREADS=1
fi

module -s purge
module -s load gromacs/2024.5 # <= 環境変数などの設定は module ファイルから読み込みます

##############################################################################

N_MPI=$NPROCS
N_OMP=$OMP_NUM_THREADS

gmx grompp -f grompp.mdp >& grompp.out
mpirun -n ${N_MPI} gmx_mpi mdrun -ntomp ${N_OMP} -s topol >& mdrun.out
```

ジョブスクリプトの書き方に関する参考情報

こちらのページに select 文のサンプルや簡単な説明があります。