g16subコマンドによるGaussianジョブの投入方法

最終更新: 2024/4/19

はじめに

このページでは g16sub コマンドで RCCS スパコンに Gaussian ジョブを投入する方法を説明します。

- 事前に必要な準備(インプットファイルの転送)
- g16sub でジョブを投入する
- ジョブの状態確認(jobinfo)
- ジョブが終了すると
- formchk の実行

g16sub のオプションやヒントについてはマニュアルの g16sub/g09sub に関するページをご確認ください。

事前に必要な準備

前提条件として、以下の準備が必須です。

- RCCS スパコンログインサーバ(ccfep)にログインできるようにする
- scp, sftp 等で RCCS とのファイル送信をできるようにしておく
- Gaussian のインプットファイルを用意する(テスト用途ならば以下のサンプルインプットでも大丈夫です)

ssh や scp, sftp の設定ができていない場合はクイックスタートガイドのページをご覧ください。

サンプル Gaussian インプットファイル

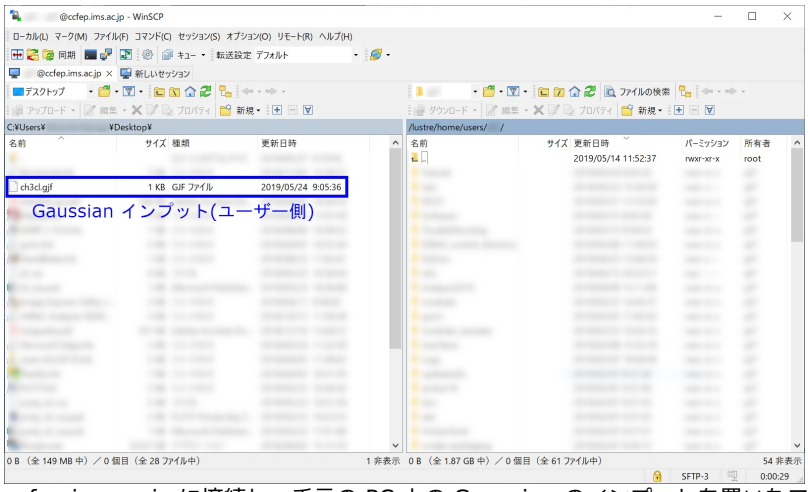
以下のような内容の ch3cl.gjf (.gjf は .com と同じく Gaussian インプットの標準的な拡張子です)が手元にあるとして、これを RCCS に転送し、Gaussian で計算を行います。

```
%chk=ch3cl.chk
# HF/6-31G(d,p) Opt

methyl chloride

0,1
C   -0.000004    1.127470    0.000000
H   -0.511417    1.468491    0.885898
H   -0.511417    1.468491   -0.885898
H    1.022922    1.468527    0.000000
Cl  -0.000004   -0.657078    0.0000
```

%mem, %nprocshared, %cpu の指定は通常 g09sub や g16sub に上書きされます。利用する CPU コア数については g16sub/g09sub の -np オプションで指定してください。メモリについては、jobtype 等に応じた上限値(少しだけ余裕をもたせてあります)が自動的に指定されますので、通常指定する必要はありません。

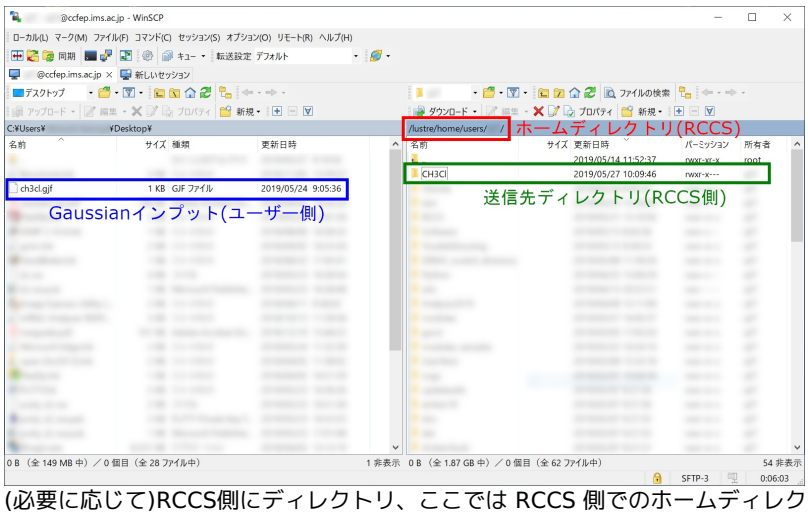
ファイルの転送(1)

WinSCP での例を示します。



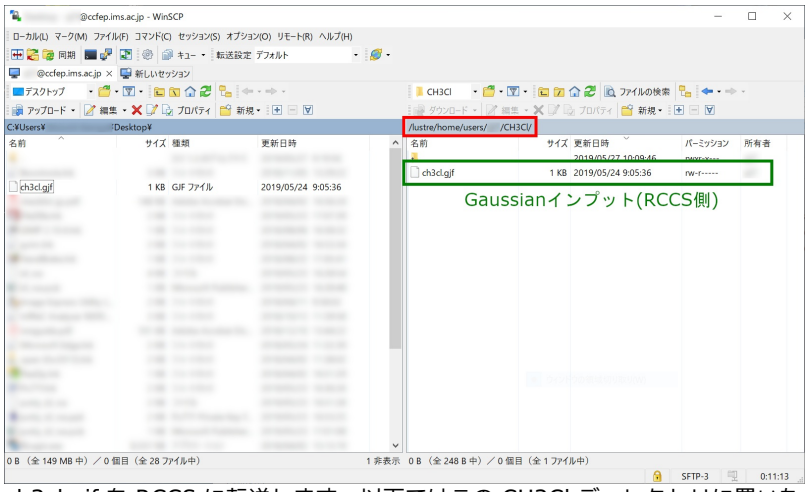
ccfep.ims.ac.jp に接続し、手元の PC 上の Gaussian のインプットを置いたフォルダに移動します。
(注意: RCCS システム上では /lustre/home/users/(ユーザー名) と /home/users/(ユーザー名) は同一の場所を指します)

ファイルの転送(2)



(必要に応じて)RCCS側にディレクトリ、ここでは RCCS 側でのホームディレクトリ直下の CH3CI (/home/users/(ユーザー名)/CH3CI)、を作成します。

ファイルの転送(3)



ch3cl.gjf を RCCS に転送します。以下ではこの CH3CI ディレクトリに置いた ch3cl.gjf というインプットを実行します。
別の場所に置いた場合には以下の記述を適宜読み替えて対応ください。

g16sub でジョブを投入する

Gaussian ジョブを投入するため、ssh (PuTTY 等)で ccfep.ims.ac.jp にログインします。
ログインができたら、CH3CI ディレクトリに cd コマンドで移動し、ls コマンドで先ほど転送したファイルがそこにあることを確認します。(ディレクトリの場所などは適宜読み換えてください。)

```
Last login: Fri Jan 27 11:24:15 2023 from ***.***.***.***
[user@ccfep3 ~]$ ls CH3Cl/
ch3cl.gjf
[user@ccfep3 ~]$ cd CH3Cl/
[user@ccfep3 CH3Cl]$ ls
ch3cl.gjf
[user@ccfep3 CH3Cl]$
```

そして、g16sub コマンドでジョブを投入します。

```
[user@ccfep3 CH3Cl]$ g16sub ch3cl.gjf
```

このようにオプション指定無しで g16sub コマンドを実行した場合、8 コアを利用します。
%mem, %nprocshared, %cpu の指定は通常 g09sub や g16sub に上書きされる点に注意してください。
計算の制限時間(walltime)は 72 時間となっています。時間内に終わらなかった場合は強制的に終了となります。

実行すると、以下のような情報が表示されます。

- H(ap)で始まる緑の部分は投入したキューの基本情報です。コア数あたりのメモリ量等が表示されます。
- 青の部分にはジョブに関係したファイル等の名前が表示されます。
- 一番最後、赤で示した数字がジョブ ID と呼ばれるユニークな ID です。投入に失敗した場合はこの数字が表示されません。

```
[user@ccfep3 CH3Cl]$ g16sub ch3cl.gjf
QUEUE detail
-----------------------------------------------------------------------------
QUEUE(MACH)  Jobtype  MaxMem     DefMem     TimLim     DefCPUs(Min-Max)
-----------------------------------------------------------------------------
   H(  ap)          1.8GB      1.2GB      72:00:00   8(1-128)
-----------------------------------------------------------------------------
JOB detail
=========================================================================
 MOL name(s)    : ch3cl
 INP file(s)    : ch3cl.gjf.ap
 OUT file(s)    : ch3cl.out
 Current dir    : /lustre/home/users/***/CH3Cl
 SCRATCH dir    : /lwork/users/${USER}/${PBS_JOBID}/gaussian

 QUEUE          : H
 Memory         : 9.6GB
 Time limit     : 72:00:00
 Job script     : /lustre/home/users/***/CH3Cl/H-1571896.sh
 Input modified : y
=========================================================================

/usr/local/bin/jsub –q H /lustre/home/users/***/CH3Cl/H-1571896.sh

4008669.ccpbs1
[user@ccfep3 CH3Cl]$
```

ジョブの状態確認(jobinfo)

投入したジョブの状態は jobinfo -c コマンドで確認できます。
(投入した直後の場合は表示されないこともあります。その場合は少しお待ち下さい。)

```
[user@ccfep3 CH3Cl]$ jobinfo –c
-----------------------------------------------------------------------------
Queue   Job ID Name           Status CPUs User/Grp       Elaps Node/(Reason)
-----------------------------------------------------------------------------
H       4008669 H-1571896.sh   Run       8  user/---  00:00:00 ccc001
-----------------------------------------------------------------------------
[user@ccfep3 CH3Cl]$
```

- ジョブID(2カラム目): g16sub 実行時の最後に表示されるものと同じジョブの固有 ID です
- ジョブのステータス(4カラム目): Run ならば実行中で、Queue ならば他のジョブの終了を待っている状態です

- 経過時間と実行中のノード名(一番右のカラム): 時間は実行時間(Run の場合)もしくはこれまでに待った時間(Queue の場合)です。
    - Run の場合は右端のカラムに実行中のノード名が表示されます。
    - Queue の場合は実行待ちになっている理由が表示されます。
        - (cpu) 空き CPU が足りないために待っています
        - (long) 次回メンテまでにジョブが終わらないため、実行できない状態です
        - (other) 投入直後はこの表示になることがあります。

一旦ジョブが投入されてしまえば、SSH接続を切断してもジョブは残ったままになります。

ジョブが終了すると

ジョブが終了すると、実行したディレクトリには以下のようなファイルが残っているはずです。
この内、H_ で始まるファイルと .ap で終わるファイルについては g16sub が作成したもので、正常に終了した場合には気にする必要はありません。

```
$ ls
H-1571896.sh          H-1571896.sh.o4008669  ch3cl.gjf     ch3cl.out
H-1571896.sh.e4008669  ch3cl.chk             ch3cl.gjf.ap
$
```

出力ファイル(ch3cl.out)についてはジョブの実行中も less や tail コマンドで内容を確認できます。
今回のインプットのように %chk でチェックポイントファイルを指定していれば、それもここに作成されます。

formchk の実行

チェックポイントファイル(.chk)を formchk コマンドでテキスト形式(.fchk)に変換したい場合、まず以下のようなコマンドで設定を読み込む必要があります。

ログインシェルが /bin/bash もしくは /bin/zsh の場合:

```
$ source /apl/gaussian/16c02/g16/bsd/g16.profile
```

ログインシェルが /bin/csh の場合:

```
$ source /apl/gaussian/16c02/g16/bsd/g16.login
```

上記コマンドを実行すれば、一見なにも起こりませんが内部設定が変更され、formchk コマンドが使えるようになります。
(PYTHONPATH: Undefined variable. と出力された場合も読み込みは完了しています。以下のように formchk も実行可能です)

```
$ formchk ch3cl.chk
 Read checkpoint file ch3cl.chk type Unk
 Write formatted file ch3cl.fchk
 FChkPn:  Coordinates translated and rotated
 FChkPn:  Coordintesmatch /B/ after translation and rotation
$
```

もし、自身のログインシェルがどちらかわからない場合は以下のように echo $SHELL コマンドを実行してください。

```
$ echo $SHELL
```