(2017/04/12)

FOCUS スパコン

従量利用アプリケーション
利用の手引き

計算科学振興財団

目次

1. 従量利用アプリケーションの実行方法 3
1.1. Gaussian の実行 3
1.1.1. 環境の設定 3
1.1.2. ジョブ投入スクリプトの作成 4
1.1.3. ジョブ投入スクリプトの実行 5
1.2. MIZUHO/ BioStation の実行 6
1.2.1. 環境の設定 6
1.2.2. ジョブ投入スクリプトの作成 6
1.2.3. ジョブ投入スクリプトの実行 6

1. 従量利用アプリケーションの実行方法

FOCUS スパコンにて、ライセンスが必要ないくつかのアプリケーションを利用できます。個々のユーザーがライセンスを準備する必要はありません。また、計算資源利用料金とは別にそれぞれのアプリケーション利用料がかかります。アプリケーションの利用にはシステム上に登録する必要がありますので、OKBiz「問合せ（技術質問）」よりご連絡ください。

1.1. Gaussian の実行

以下ではフロントエンドサーバにログインし、SLURM を利用したジョブ投入スクリプトとして Gaussian の計算を実行する方法を説明します。

1.1.1. 環境の設定

利用したいバージョンの Gaussian の環境を設定します。
適切な環境設定ファイルを読み込んでください。下記の通り設定をジョブ投入スクリプトに記述してください。
（シェルの環境設定ファイル $HOME/.bashrc に記述することも可能です）
まず、利用する Gaussian のトップディレクトリを g09root あるいは g16root として読み込みます。
利用するバージョンに応じて下記のいずれかの行を指定してください。

```
export g09root=/home1/share/g09               # デフォルトの Gaussian（現在は g09d01s のリンク）
export g09root=/home1/share/g09c01             # Gaussian 09 Rev. C01
export g09root=/home1/share/g09d01s            # G09 Rev. D01 ソースコード版 pgi コンパイル
export g09root=/home1/share/g09d01s_intel      # G09 Rev. D01 ソースコード版 intel コンパイル：全システム版（SSE4.2）
export g09root=/home1/share/g09d01s_intel_DE # G09 Rev. D01 ソースコード版 intel コンパイル：DEFH システム版（AVX）

export g09root=/home1/share/g09e01intelsse4  # G09 Rev. E01 ソースコード版 intel コンパイル：全システム版（SSE4.2）
export g09root=/home1/share/g09e01intelavx   # G09 Rev. E01 ソースコード版 intel コンパイル：DEFH システム版（AVX）
export g09root=/home1/share/g09e01pgi        # G09 Rev. E01 ソースコード版 pgi コンパイル

export g09root=/home1/share/g09e01sse4       # G09 Rev. E01 バイナリ版 全システム版（SSE4.2 w/LINDA）
export g09root=/home1/share/g09e01avx        # G09 Rev. E01 バイナリ版 DEFH システム版（AVX w/LINDA）
```

```
export g16root=/home1/share/g16a03s_pgi      # G16 Rev. A03 ソースコード版 pgi コンパイル【導入済み】
export g16root=/home1/share/g16a03s_intel_sse
                          # G16 Rev. A03 ソースコード版 intel コンパイル：全システム版（SSE4.2）【5 月導入予定】
export g16root=/home1/share/g16a03s_intel_avx
                          # G16 Rev. A03 ソースコード版 intel コンパイル：DEFH システム版（AVX）【5 月導入予定】
export g16root=/home1/share/g16a03s_intel_avx2
                          # G16 Rev. A03 ソースコード版 intel コンパイル：FH システム版（AVX2）【5 月導入予定】

export g16root=/home1/share/g16a03_legacy    # G16 Rev. A03 バイナリ版 全システム版（legacy w/LINDA）
export g16root=/home1/share/g16a03_sse4      # G16 Rev. A03 バイナリ版 全システム版（SSE4.2 w/LINDA）
export g16root=/home1/share/g16a03_avx       # G16 Rev. A03 バイナリ版 DEFH システム版（AVX w/LINDA）
export g16root=/home1/share/g16a03_avx2      # G16 Rev. A03 バイナリ版 FH システム版（AVX2 w/LINDA）
```

利用する Gaussian の設定後、プロファイルを読み込みます。

```
source $g09root/g09/bsd/g09.profile
```

```
source $g16root/g16/bsd/g16.profile
```

なお、インテルコンパイラ版を利用の場合は、インテルコンパイラの環境も設定する必要があります。

```
module load PrgEnv-intel
```

1.1.2. ジョブ投入スクリプトの作成

サンプルスクリプトを参考にして、ジョブ投入スクリプトを作成します。
また、/home1/share/g09 にジョブ投入スクリプト例（逐次計算、ノード内並列、ノード間並列(Linda)）を置いておりますので参考にしてください。

・ノード内並列用サンプルスクリプト：g09sample.sh

```
#!/bin/bash
#SBATCH -p d009m                              # キュー名
#SBATCH -n 20                                 # 最大プロセス数
#SBATCH -J Gaussian_SHARED                    # ジョブ名
#SBATCH -e Gaussian_SHARED.e%J                # 標準エラー出力、%J はジョブ ID に置換
#SBATCH -o Gaussian_SHARED.o%J                # 標準出力、%J はジョブ ID に置換

#g09d01s Intel Compiler version
module load PrgEnv-intel                      # Intel コンパイラー環境変数設定
export g09root=/home1/share/g09d01s_intel     # Gaussian ディレクトリの指定
                                    (例：G09 Rev. D01 ソースコード版　intel コンパイル：全システム版（SSE4.2）

#Please use g09d01s, if there is a problem in g09d01s_intel.
#export g09root=/home1/share/g09d01s

source $g09root/g09/bsd/g09.profile           # Gaissian 環境変数設定

#Input Data
INPUT=test0139                                # 入力ファイル名の指定

export GAUSS_SCRDIR=/work/                    # スクラッチディレクトリを指定
g09 ${INPUT}.com                              # Gaussian を実行
```

1.1.2.1. 環境変数「GAUSS_SCRDIR」の指定（/work）

環境変数「GAUSS_SCRDIR」では Gaussian の作業ファイル（※）が作成されるスクラッチディレクトリを指定します。/work 配下を指定してください。

※「Gau-プロセス ID.拡張子」というファイルで、ジョブが終われば自動的に消されます。

環境変数「GAUSS_SCRDIR」の指定を行わない場合は、Gaussian 環境設定ファイルにて指定されているディレクトリ（ホームディレクトリ内 tmp）がスクラッチディレクトリとして設定されます。

演算ノードのローカルディレクトリ（/work）や、分散ファイルシステム（/home2）は、比較的高速に書込みを行いますが、それに対し各アカウントのホームディレクトリ（/home1）は書込みが遅く、他の利用者も含めて同時に多数の Gaussian ジョブが流れると、この書き込み速度をそれらのジョブで分け合うことになり、書込み速度が低下します。このことから /work や /home2 の使用を推奨します。

表 1.1.2.1 ディレクトリ別の書込み速度

システム	ディレクトリ	書込み速度	容量(※)	備考
A,B,C システム	/work	75MB/s	400GB 未満	環境変数「GAUSS_SCRDIR」推奨
D,E システム	/work	D:300MB/s E:200MB/s	D：6TB 未満 E：2TB 未満	環境変数「GAUSS_SCRDIR」推奨
F,H システム	/work	F:130MB/s H:440MB/s	F：870GB 未満 H：470GB 未満	環境変数「GAUSS_SCRDIR」推奨
NAS ストレージ	/home1/gxxx	100MB/s	課題あたり 200GB	xxx は課題名
分散ファイルシステム	/home2/gxxx	1GB/s (プロセスあたり)	別途契約が必要	xxx は課題名

1.1.3. ジョブ投入スクリプトの実行

sbatch コマンドを使ってジョブ投入スクリプト（例：g09sample.sh）を実行します。

```
$ sbatch g09sample.sh
```

1.2. MIZUHO/BioStation の実行

以下ではフロントエンドサーバにログインし、SLURM を利用したジョブ投入スクリプトとして MIZUHO/ABINIT-MP3.0 の計算を実行する方法を説明します。

1.2.1. 環境の設定

module コマンドにより MIZUHO/BioStation の環境を設定します。

```
$ module load MIZUHO_ABINIT-MP3.0_FOCUS
```

1.2.2. ジョブ投入スクリプトの作成

サンプルスクリプトを参考にして、ジョブ投入スクリプトを作成します。
また、/home1/share/MIZUHO_ABINIT-MP3.0_FOCUS/sample にジョブ投入スクリプト例を置いておりますので参考にしてください。

・サンプルスクリプト：run.sh

```
#!/bin/bash
#SBATCH -p d009m                                    # キュー名
#SBATCH -n 20                                       # 最大プロセス数
#SBATCH -J mizuho_abinit-mp                         # ジョブ名
#SBATCH -e mizuho_abinit-mp.e%J                     # 標準エラー出力、%J はジョブ ID に置換
#SBATCH -o mizuho_abinit-mp.o%J                     # 標準出力、%J はジョブ ID に置換

module load MIZUHO_ABINIT-MP3.0_FOCUS               # MIZUHO_ABINIT-MP 利用環境変数設定

BASENAME=gly5-HF                                    # 入力ファイル名
mkinp.tcl < $BASENAME.ajf > $BASENAME.ajf.tmp
mpirun -np ${SLURM_NTASKS} abinitmp $BASENAME.ajf.tmp >& $BASENAME.log
```

1.2.3. ジョブ投入スクリプトの実行

sbatch コマンドを使ってジョブ投入スクリプト（例：run.sh）を実行します。

```
$ sbatch run.sh
```