

TreeFoam上から名大スパコン、FOCUSに接続し、 OpenFOAMで計算

TreeFoam:

OpenFOAMをGUI上で操作、設定できるアプリ。

TreeFoam上から、各スパコンに接続し、Job投入も可。

17/10/8

藤井 成樹

1. TreeFoamとは

OpenFOAMの操作性向上を目的として、GUI上で操作できる様にしたものの。11/4月から作成し始め、約6年間、試行錯誤しながら、バージョンアップを繰り返している。

<TreeFoamの特徴>

1. caseがツリー表示される。
2. ツリー上にcase内容 (solver、計算結果フォルダ数、並列数など) が表示される。
3. FreeCAD、SAlome-Meca、ParaViewのランチャを備える。
4. Dictファイルを意識せずに、snappyHexMesh、cfMeshでメッシュが作成できる。
5. gridEditorを使ってOpenFOAMの境界条件が表形式で編集できる。
6. topoSetEditorを使ってcellの抽出、加工が容易にできる。
7. TreeFoamやgridEditorは、マルチタスクに対応している為、それらを複数起動して、お互いにcopy & pasteできる。
8. TreeFoamとgnomeのファイルマネージャ間でファイルのcopy & pasteが可能。
9. binaryファイルの編集が可能。
10. サーバがTreeFoam上にマウントでき、FOCUSや名大スパコンについてはのJob管理用GUIを備える。

TreeFoamのホーム画面

TreeFoam上で
startFrom、stopAtが設定できる

case directory: /home/dexcs/Desktop/CAE
現在の解析case名: cavity

OpenFoam環境: bashrc-FOAM-2.3-DEXCS

startFrom: startTime:0 stopAt: endTime:0.5 controlDict: 編集

Tree表示

solver名表示

solver	BCPn	nR	st	ed
icoFoam	anP	1	0.0	
icoFoam	BnP4	6	0.0	0.5

formatや並列数を表示
anP :ascii、非圧縮、シングル処理
BCP2 :binary、圧縮、2並列

log /home/dexcs/TreeFoam/temp/0_logTreeFoam
TreeFoam ver 2.25-150329+dexcsSwak (0) を起動しました。
OpenFOAM - 2.3.x

合計 38.93 GB, 空き 23.88 GB

ホーム画面上で、OpenFOAMのcase内容が一望できる。
(solver名、圧縮・非圧縮、並列数、計算結果有無 等)

境界条件の設定

ホーム画面上からアイコンをクリックして起動するgridEditorで設定
(各fieldをテキストエディタで編集していた。)

gridEditor画面

ダブルクリックすると、Editorが開く

alfabetical order

	define patch at constant/. (boundary)	U	epsilon	k	nuTilda	
field type		volVectorField;	volScalarField;	volScalarField;	volScalarField;	volSc
dimensions		[0 1 -1 0 0 0];	[0 2 -3 0 0 0];	[0 2 -2 0 0 0];	[0 2 -1 0 0 0];	[0 2 -1
internal Field		uniform (0 0 0);	uniform 14.855;	uniform 0.375;	uniform 0;	unifo
frontAndBack	type empty; inGroups 1(empty);	type empty;	type empty;	type empty;	type empty;	type e
inlet	type patch;	type fixedValue; value uniform (10 0 0);	type fixedValue; value uniform 14.855;	type fixedValue; value uniform 0.375;	type fixedValue; value uniform 0;	type c value
lowerWall	type wall; inGroups 1(wall);	type fixedValue; value uniform (0 0 0);	type epsilonWallFunction; value uniform 14.855;	type kqRWallFunction; value uniform 0.375;	type zeroGradient;	type r value
outlet	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type c value
upperWall	type wall; inGroups 1(wall);	type fixedValue; value uniform (0 0 0);	type epsilonWallFunction; value uniform 14.855;	type kqRWallFunction; value uniform 0.375;	type zeroGradient;	type r value

alfabetical order

alfabetical order

Field名

field type & dimension

internalField

Boundary Field

patch名 patchType

ダブルクリックすると、patch名が変更できる

Excelの様にダブルクリックして、cell内容を直接編集できる。

全fieldの境界条件が一望できる、修正が容易。

2. サーバへの接続 (sshで接続)

2-1. 設定ファイルの作成

接続の為の設定ファイルを作成する。

ssh設定ファイル (~/.ssh/config)

```
#名大用-----  
Host cx400Nagoya  
  HostName    cx.cc.nagoya-u.ac.jp  
  User        z00000a  
  Port        22  
  IdentityFile ~/.ssh/cx400Nagoya_id_rsa  
  
Host fx100Nagoya  
  HostName    fx.cc.nagoya-u.ac.jp  
  User        z00000a  
  Port        22  
  IdentityFile ~/.ssh/cx400Nagoya_id_rsa
```

TreeFoam設定ファイル (~/.TreeFoamUser/data/sshfs_data)

```
#---- 名大cx400 ----  
Host 名大cx400  
  HostName    cx400Nagoya  
  HostDir     /home/usr3/z00000a  
  MountPoint  
  setEnviron  
    . ~/0F231cxTerminal  
    cd ~  
    echo -n > ~/setEnviron  
  
#---- 名大fx100 ----  
Host 名大fx100  
  HostName    fx100Nagoya  
  HostDir     /home/usr3/z00000a  
  MountPoint  
  setEnviron  
    :
```

login時の環境設定内容

設定ファイルを作成することで、サーバへの接続が可能になる。

2-2. login時の環境設定

login時の環境設定ができる様に、サーバ側の「`~/.bash_profile`」の最後に「`. ~/setEnviron`」を追加しておき、login時に`setEnviron`を実行する様に設定しておく。

`setEnviron`の内容は、TreeFoamが随時書き換えサーバに転送する。

設定した環境設定内容
(`setEnviron`の内容)

```
. ~/0F231cxTerminal  
cd ~  
echo -n > ~/setEnviron
```

OpenFOAM-2.3.1用の環境設定
サーバ内に作成しておく

login時の移動先directory
移動先は、TreeFoam側で
書き換える

`setEnviron`の内容をクリア
他の方法でloginした時に
実行しない様にする為

この設定により、TreeFoam上からloginした時に、
OpenFOAMの環境設定が終了
指定したdirectoryに移動
した状態でlogin端末が起動する。

2-3. TreeFoamServerの設定

通信負荷を軽減する為に、server側で処理した結果を受け取る形式をとっており、この処理の為に「TreeFoam/TreeFoamServer」をフォルダ毎サーバ側の\$HOME直下にコピーする。

local側から何回もsshでコマンドを送出して得られる結果を受け取るよりも、1回のコマンド送付で結果が得られる様に、サーバ側にオリジナルのコマンド群をTreeFoamServer内に準備している。

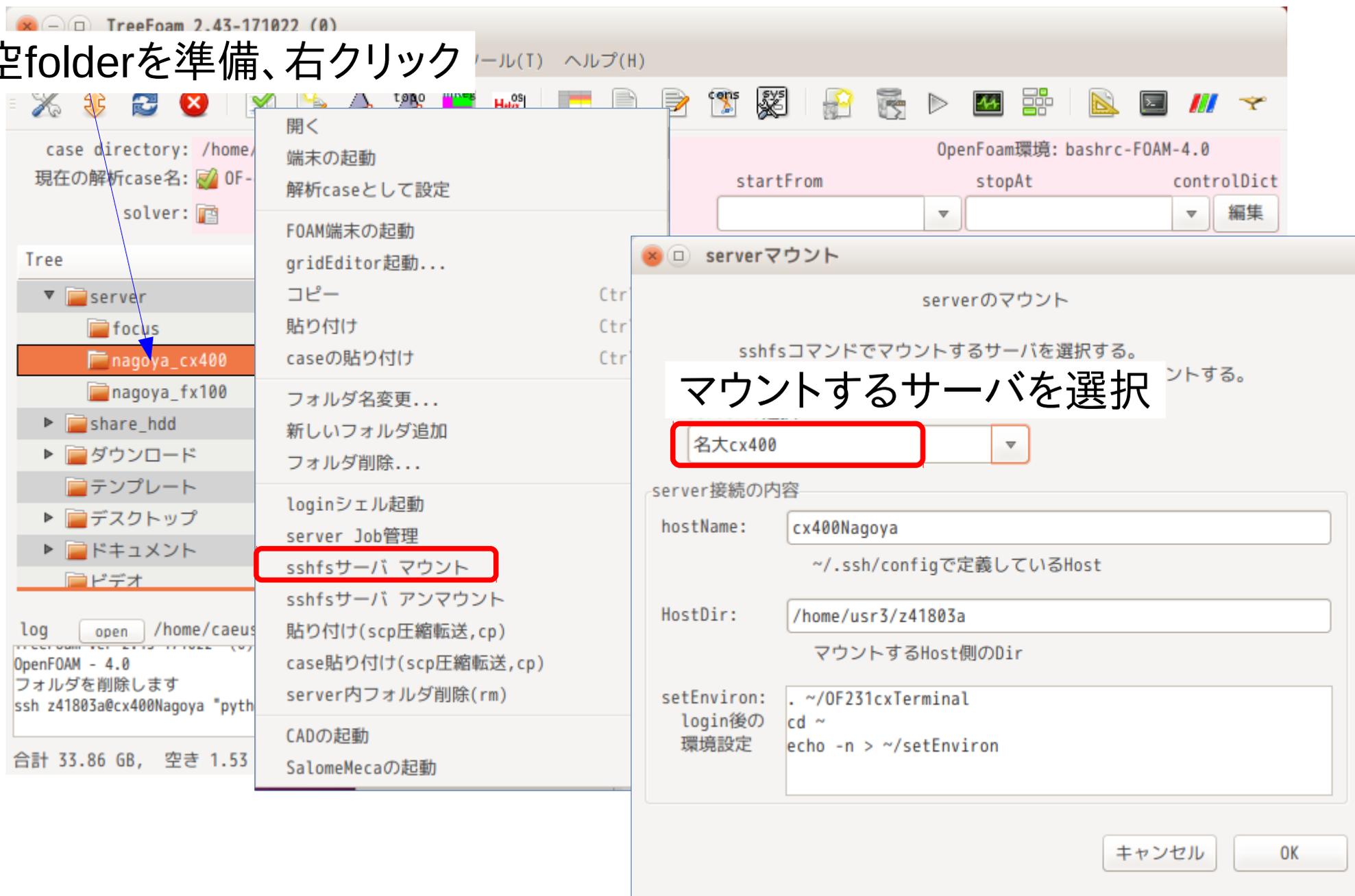
このオリジナルのコマンド群は、pythonスクリプトの為に、サーバ側にpythonがインストールされている必要がある。(FOCUS、名大スパコンともpythonがインストールされている。)

以上の設定で、外部サーバに接続できる状態になる。

3. サーバをlocal側にマウント

サーバに接続する。

空folderを準備、右クリック



TreeFoam_2.43-171022 (0)

ファイル(F) case作成変更(M) 編集(E) 計算(C) ツール(T) ヘルプ(H)

case directory: /home/caeuser/CAE/CAE-FOAM OpenFoam環境: bashrc-FOAM-4.0

現在の解析case名: OF-4.0 startFrom stopAt controlDict

solver:

Tree

- server
 - focus
 - nagoya_cx400**
 - OpenFOAM
 - cx
 - OF2.3
 - cavity
 - cavity_para-1
 - pitzDaily
 - pitzDaily_large
 - icoFoam anP 6 0.0 0.5
 - icoFoam anP2 6 0.0 0.5
 - pisofFoam anP 51 0.0 0.05
 - nisoFoam anP128 1 0.0

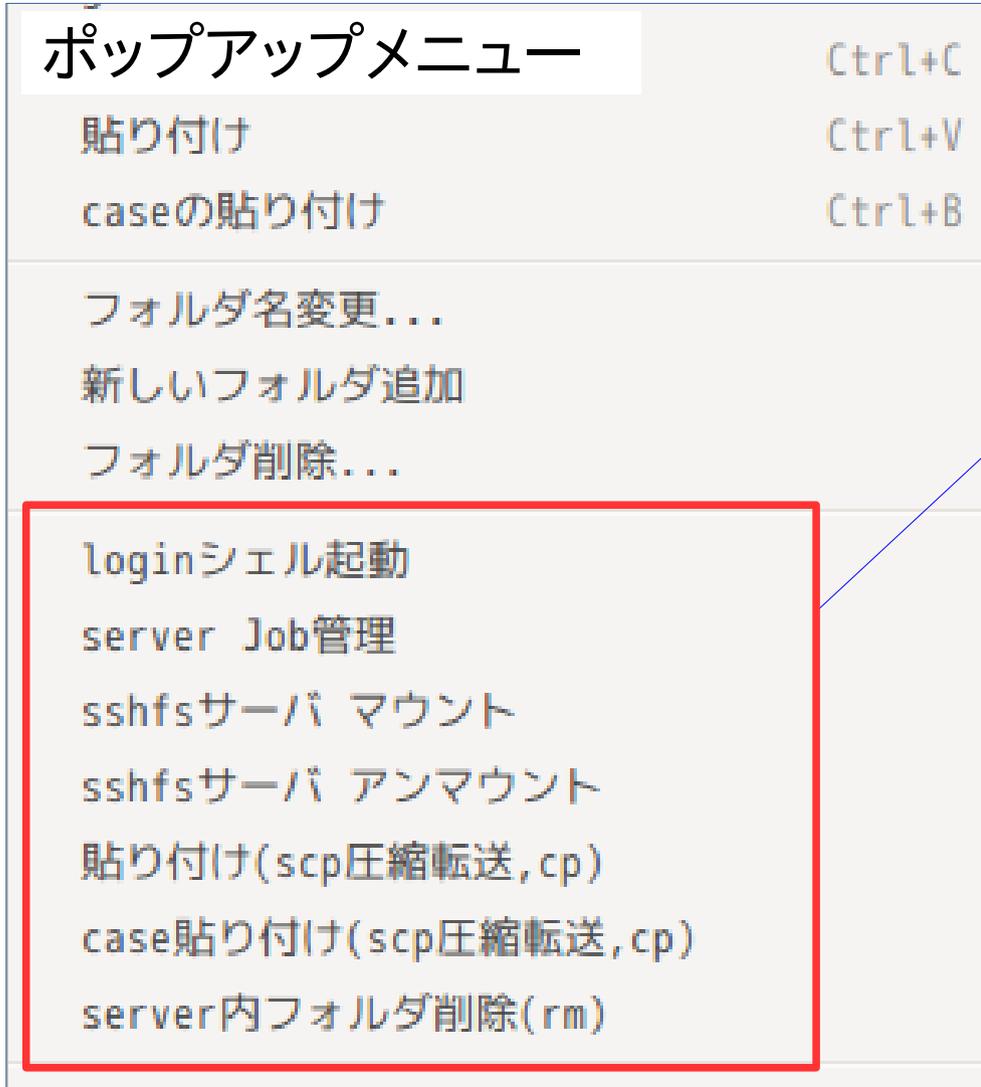
log /home/caeuser/.TreeFoamUser/temp/0_logTreeFoam

```
ssh z41803a@cx400Nagoya "python ~/TreeFoamServer/bin/server_getUntilSelSolveFolders.py /home/usr3/z41803a/./ /home/usr3/z41803a/./"
ssh z41803a@cx400Nagoya "python ~/TreeFoamServer/bin/server_getUntilSelSolveFolders.py /home/usr3/z41803a/./ /home/usr3/z41803a/./"
```

合計 985115.62 GB, 空き 844012.73 GB

サーバがマウントされたので
空folder内にサーバ側のdirectoryが確認できる。

TreeFoam内にサーバマウントしている為、違和感なく操作できる。
login端末、Job管理、ファイル転送がGUI上で可能になる。

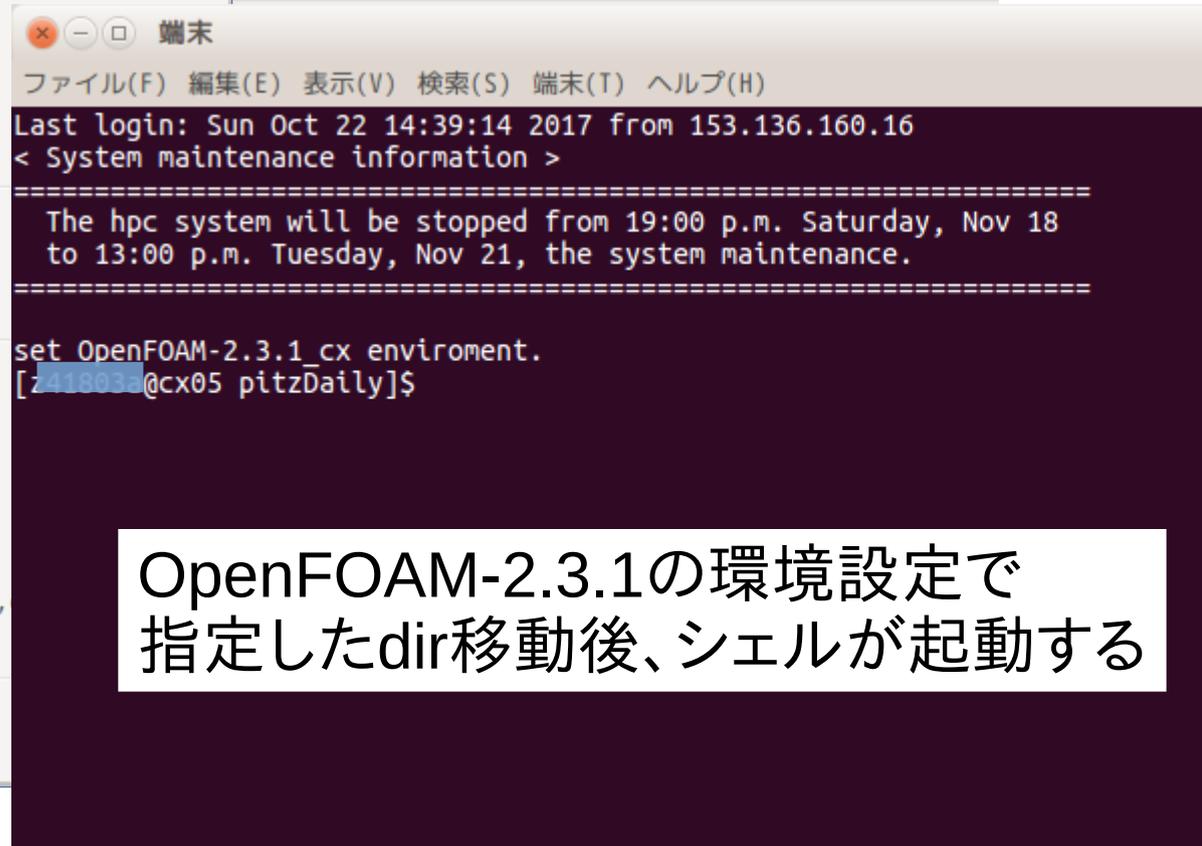


対サーバのメニュー

- ・loginシェル起動
- ・server Job管理
- ・sshfsサーバ マウント
- ・sshfsサーバ アンマウント
- ・貼り付け(scp圧縮転送, cp)
- ・case貼り付け(scp圧縮転送, cp)
- ・server内フォルダ削除(rm)

local-server間の「貼り付け」のデータ転送は、圧縮転送するのでnormal転送に比べて4~5倍早くなる。
server内「貼り付け」はcpコマンド、server内「削除」はrmコマンドを送出する為、処理は早い

5. サーバにloginする



フォルダを選択
右クリック

OpenFOAM-2.3.1の環境設定で
指定したdir移動後、シェルが起動する

6. サーバにJobを投入

<CX400の場合>

名大CX400 Job管理

currDir: ...r/nagoya_cx400/cx/0F2.3/pitzDaily_large_copy5_para24

Job投入

Jobファイルを選択、作成して、Jobを投入する。
pjsubオプション（キュー名、プロセス数等）は、右画面上で、再設定できる。

Job File名: runPara 参照... Job編集

キュー: cx-small 制限時間: 5.0 hrs 設定

core数/vnode: 24 vnode数: 1

設定file編集 Job投入実行...

ノードの状況確認

pjstat --rsc --l... run

Jobファイル

```
1 #!/bin/bash
2 #PJM -N runPara
3 #PJM -L "rscgrp=cx-small"
4 #PJM -L "vnode=1"
5 #PJM -L "vnode-core=24"
6 #PJM -L "elapsed=5:0:0"
7 #PJM -o solve.log
8
9 source /center/local/apl/cx/OpenFOAM/OpenFOAM-2.3.1/etc/bashrc
10 export CMP_NUM_THREADS=24
11 mpirun -np 24 pisofoam -parallel
12 |
```

ACCEPT QUEUED S
0 0
s 0 0

Job実行状況の確認

投入Job読込・表示

pjstat

JOB_ID	JOB_NAME
--------	----------

folder開く login端末起動 閉じる

Job管理スクリプト
この部分はGUIが作成

Jobの投入がGUI上で容易に行える

名大FX100 Job管理

currDir: ...nagoya_fx100/fx/OF2.3/pitzDaily_large_fx_4x6x5:torus

Job投入

Jobファイルを選択、作成して、Jobを投入する。
pjsubオプション（キュー名、プロセス数等）は、右画面上で、再設定できる。

Job File名: runPara [参照..] [Job編集]

キュー: fx-large [設定] 制限時間: 3.0 hr [設定]

node数: 4x6x5:torus [設定file編集] [Job投入実行...]

ノードの状況確認: pjstat --rsc --limit [run]

```
1 #!/bin/bash
2 #PJM -N runPara
3 #PJM -L "rscgrp=fx-large"
4 #PJM -L "node=4x6x5:torus"
5 #PJM -L "elapsed=3:0:0"
6 #PJM -o solve.log
7
8 source /center/local/apl/fx/OpenFOAM/bashrc_for_USER
9 mpiexec pisofoam -parallel
10 |
```

Job実行状況の概観

投入Job読込: [pjstat]

JOB_ID

folder開

FX100の場合は、OpenFOAMで使用する事を前提に node割当をトーラス、メッシュモードに限定している。

7. Jobの実行結果

使用するcaseは、OpenFOAM-2.3内のtutorialsの
pitzDaily LES
を使用。モデル形状を変更し、100万要素に変更。

項目	CX400		FX100	FOCUS
並列数	48	24	120	120
node数	2	1	4x6x5:torus	Asystem, 7
executionTime	5286	5139	3198	2334
clockTime	10214	5145	3209	2519
待ち時間	4928	6	11	185

CX400:node間通信が遅く、全体的に時間がかかる。

node内で収まる並列数(max28)で計算するのが得策

FX100:node間通信が早い。大規模並列に適す。

FOCUS:計算時間は最も早いですが、node間通信がやや遅いので、
大規模並列では、ロス時間が大きくなる可能性あり。

小規模: CX400、 中規模: FOCUS、 大規模: FX100

8. まとめ

接続に必要な設定ファイル、login時の環境設定、通信負荷軽減の為にTreeFoamServerを配置する事で、TreeFoam上からサーバへのアクセスやJob管理等がGUI上で可能になる。

- ・login端末の起動
指定したdirectoryに移動し、OpenFOAMの環境でloginシェルが起動
- ・Job管理
Job投入、Job実行状況の確認(log表示、残渣表示)
- ・ファイルやcaseの転送
folderを指定して圧縮転送するので、通常転送より4~5倍程早い
- ・サーバ内folder操作
コピー、削除について、cp、rmコマンドを送出する

設定ファイル作成と環境設定、TreeFoamServerを配置すれば、名大スパコン、FOCUS以外でもサーバへsshでTreeFoam上からアクセスが可能。(Job管理用GUIを除く。)

詳細は、「TreeFoam操作マニュアル」を参照。