



conrod

2013年11月1日
第9回FrontISTR研究会

目次

- ▶ ねらい
- ▶ 入力データの確認
 - ▶ ログインサーバ上のファイル
 - ▶ REVOCAPの利用
- ▶ 線形静解析
 - ▶ 逐次解析
 - ▶ 並列解析（領域分割を含む）
 - ▶ 結果の確認
- ▶ 固有値解析
 - ▶ 逐次解析（←時間がかかるのでハンズオンでは実施しません）
 - ▶ 並列解析（領域分割を含む）
 - ▶ 結果の確認



ねらい・入力データの確認

■ 比較的小規模な例題を用いて一連の計算手順を体験する

- 94,047節点、56,115要素、四面体2次要素

ファイルの中味に関する説明は、スライド「はじめてのFrontISTR」を参照のこと

■ ログインサーバ上のファイル

- ~/fistr/conrod_****/1 (逐次計算用)
- ~/fistr/conrod_****/n (n並列計算用, n=2, 4, ..., 128)
- FrontISTRの入力データ、実行シェルスクリプト
- パーティショナの入力データ、実行シェルスクリプト
- 静解析・固有値解析は解析制御ファイル(cntファイル)が異なるがメッシュファイルは共通

static 線形静解析
eigen 固有値解析
thermal 熱伝導解析

以後のスライドでは
conrod_staticを例として説明

■ REVOCAPの利用

- WinSCPを用いて、端末PCに FistrModel.mshとFistrModel.cntを転送
- REVOCAPを用いて、モデル形状、メッシュ、境界条件等を確認



線形静解析 (1/3)

■ 逐次解析

■ ログインサーバ上のファイル ~/fistr/conrod_static/1

- FrontISTRの入力データ hecmw_ctrl.dat, FistrModel.msh, FistrModel.cnt
- FrontISTRの実行シェルスクリプト job_fistr.sh

■ 計算実行

```
cd ~/fistr/conrod_static/1  
pjsub job_fistr.sh      (./do_fistr.shでも可)
```



job_fistr.sh FrontISTRの実行スクリプト

逐次計算

<code>#!/bin/bash</code>	
<code>#PJM -N "fistr1_1"</code>	ジョブの名前
<code>#PJM -L "rscgrp=large"</code>	リソースグループ名(※)
<code>#PJM -L "node=1"</code>	使用ノード数
<code>#PJM -L "elapsed=12:00:00"</code>	計算経過時間
<code>#PJM -j</code>	ジョブの標準エラー出力を標準出力へ出力
<code>~/bin/fistr1.serial</code>	実行モジュール名(逐次版)
<code>mv *.log ./log/</code>	出力ログをフォルダに移動
<code>...</code>	

(※) large: 最大84ノード使用可能, small: 最大12ノード使用可能

線形静解析 (2/3)

■ 並列解析

■ 計算サーバ上のファイル ~/fistr/conrod_static/16

- パーティショナ関係 hecmw_part_ctrl.dat ←分割数(DOMAIN=n)を指定
- パーティショナの実行シェルスクリプト job_part.sh
- FrontISTRの入力データ hecmw_ctrl.dat, FistrModel.msh.0-15, FistrModel.cnt
- FrontISTRの実行シェルスクリプト job_fistr.sh

■ パーティショナの実行

```
cd ~/fistr/conrod_static/16  
pjsub run_part.sh    (./do_part.shでも可)
```

■ 計算実行

```
cd ~/fistr/conrod_static/16  
pjsub run_fistr.sh   (./do_fistr.shでも可)
```



job_part.sh パーティショナの実行スクリプト

逐次計算

<code>#!/bin/bash</code>	
<code>#PJM -N "part_16"</code>	ジョブの名前
<code>#PJM -L "rscgrp=large"</code>	リソースグループ名(※)
<code>#PJM -L "node=1"</code>	使用ノード数
<code>#PJM -L "elapsed=6:00:00"</code>	計算経過時間
<code>#PJM -j</code>	ジョブの標準エラー出力を標準出力へ出力
<code>~/bin/hecmw_part1</code>	実行モジュール名
<code>mv *.log ./log/</code>	出力ログをフォルダに移動

(※) large: 最大84ノード使用可能, small: 最大12ノード使用可能

job_fistr.sh FrontISTRの実行スクリプト

並列計算 PJM (ジョブ管理システム) の利用例

```
#!/bin/bash
#PJM -N "fistr1_16"           ジョブの名前
#PJM -L "rscgrp=large"       リソースグループ名(※)
#PJM -L "node=1"            使用ノード数
#PJM --mpi "proc=16"        MPIによる並列計算のプロセス数
#PJM -L "elapsed=12:00:00"  計算経過時間
#PJM -j                      ジョブの標準エラー出力を標準出力へ出力

mpiexec ~/bin/fistr1
```

(※) large: 最大84ノード使用可能, small: 最大12ノード使用可能

線形静解析 (3/3)

■ 結果の確認

■ REVOCAPの利用

■ WinSCPを用いて、端末PCに以下のファイルを転送

- `~/conrod_static/FistrModel.cnt`
- `~/conrod_static/FistrModel.msh` 全体領域のメッシュ
- `~/conrod_static/n/result/FistrModel.res.XX.X` 全体又は部分領域の計算結果
- `~/conrod_static/n/result/FistrModel.**.inp` UCDフォーマットの結果ファイル
- `~/conrod_static/n/mesh_dist /subdomains.inp` 領域分割図

■ 変形図

■ 応力コンター図

■ 領域分割図

■ 断面表示

■ 部分領域のみの表示

■ 変形アニメーション



固有値解析

- 逐次解析、並列解析 基本的に線形静解析と同じ手順
- ログインサーバ上のファイル `~/fistr/conrod_eigen/n/`
- 分散メッシュはシンボリックリンクで線形静解析と共有
→パーティショニングは線形静解析で行った分割数については不要



FistrModel.cnt

```
:  
:  
!SOLUTION, TYPE = EIGEN      ←固有値解析を指定  
!EIGEN  
3, 1e-007, 60      ←固有値を3個求める  
:  
:
```



熱伝導解析

- 逐次解析、並列解析 基本的に線形静解析と同じ手順
- ログインサーバ上のファイル ~/fistr/conrod_thermal/n/
- メッシュファイルの形式が異なるので、パーティショニングは個別に必要

FistrModel.cnt

```
:  
:  
!SOLUTION, TYPE=HEAT ←熱伝導解析を指定  
!HEAT  
:  
:
```

