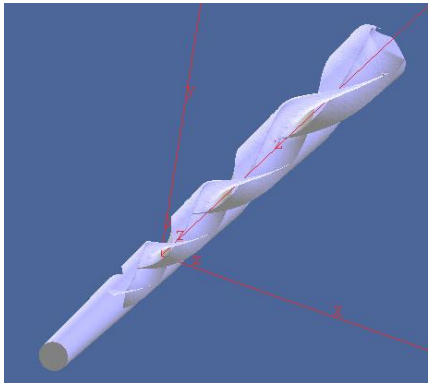


drill

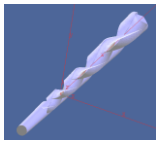
2013年11月1日

第9回FrontISTR研究会

目次

- ねらい
- 入力データの確認
  - 計算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）　（←16プロセッサ以上で実施）
  - 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

drill



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
  - 1,700,262節点、9,895,566要素、四面体1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

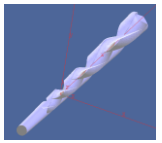
- ログインサーバ上のファイル
  - ~/fistr/drill_****/1（逐次計算用）
  - ~/fistr/drill_****/n （n並列計算用, n=2,4,…,128）
  - FrontISTRの入力データ、実行シェルスクリプト
  - パーティショナの入力データ、実行シェルスクリプト
  - 静解析・固有値解析は解析制御ファイル(cntファイル)が異なるが
    メッシュファイルは共通

****
static 線形静解析
eigen 固有値解析

以後のスライドでは
drill_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

drill



線形静解析(1/3)

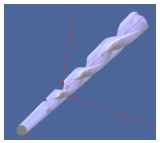
- 逐次解析

- ログインサーバ上のファイル ~/fistr/drill_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)
```

drill



job_fistr.sh　FrontISTRの実行スクリプト

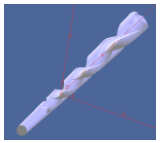
逐次計算

```
#!/bin/bash
#PJM -N "fistr1_1"              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 -L "elapse=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

~/bin/fistr1.serial             実行モジュール名(逐次版)
mv *.log ./log/                 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

drill



線形静解析(2/3)

- 並列解析
- 計算サーバ上のファイル ~/fistr/hinge_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

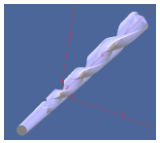
- パーティショナの実行

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

- 計算実行

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

drill



job_part.sh　パーティショナの実行スクリプト

逐次計算

```
#!/bin/bash
#PJM -N  “part_16”
#PJM -L “rscgrp=large”
#PJM -L “node=1”
#PJM -L “elapse=6:00:00”
#PJM -j

~/bin/hecmw_part1
mv *.log ./log/
```

ジョブの名前

リソースグループ名（※）

使用ノード数

計算経過時間

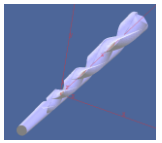
ジョブの標準エラー出力を標準出力へ出力

実行モジュール名

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

drill



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"
#PJM -L "rscgrp=large"
#PJM -L "node=1"
#PJM --mpi "proc=16"
#PJM -L "elapse=12:00:00"
#PJM -j

mpiexec ~/bin/fistr1
```

ジョブの名前

リソースグループ名（※）

使用ノード数

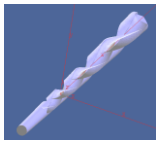
MPIによる並列計算のプロセス数

計算経過時間

ジョブの標準エラー出力を標準出力へ出力

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

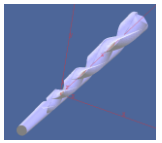
drill



線形静解析(3/3)

- 結果の確認

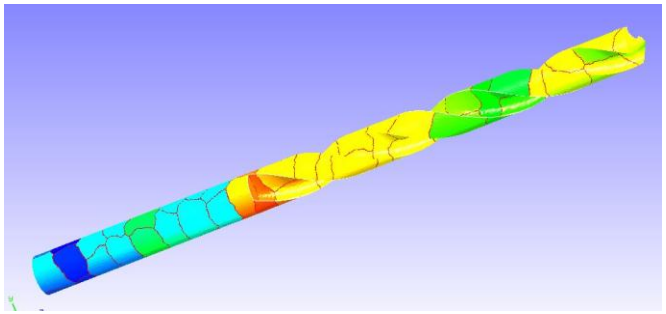
- REVOCAPの利用
  - WinSCPを用いて、端末PCに以下のファイルを転送
    - ~/drill_static/FistrModel.cnt
    - ~/drill_static/FistrModel.msh　　全体領域のメッシュ
    - ~/drill_static/n/result/FistrModel.res.XX.X　　全体又は部分領域の計算結果
    - ~/drill_static/n/result/FistrModel.**.inp　　UCDフォーマットの結果ファイル
    - ~/drill_static/n/mesh_dist /subdomains.inp　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



並列性能の評価

- 並列化率（α）、加速率（s）、並列化効率（ε）

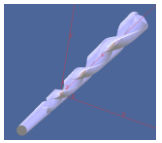
- プロセッサ数を変えて計算時間を測定する　32, 64, 128, …



- アムダールの法則を用いて、並列化率・加速率・並列化効率を推定する

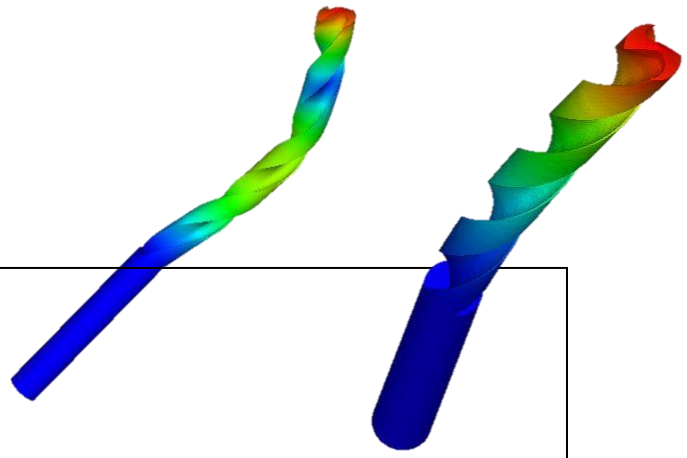
（注意）加速率、FLOPS値（対ピーク性能）、計算時間　についての評価を混同してはいけない。

drill



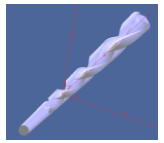
固有値解析

- 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/drill_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
   :
   :
```



drill