

大規模メッシュ作成について ーメッシュ作成の並列処理ー

pmsh:netegnによるメッシュ作成の並列処理

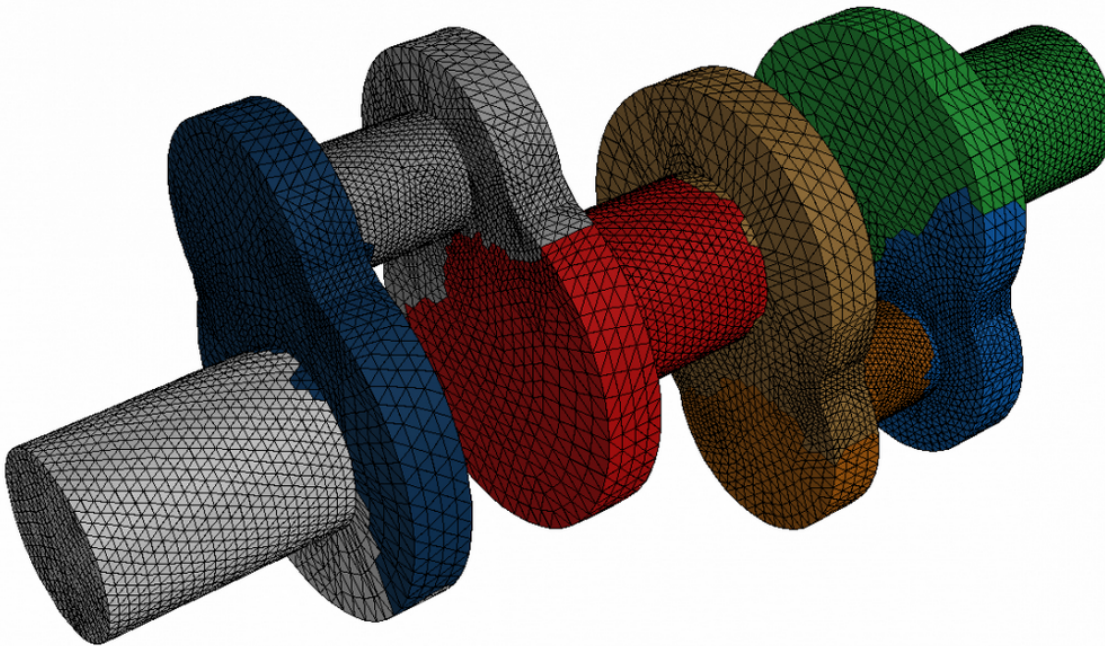
粗いメッシュを作成した後、
メッシュ分割 (parmetis)、
mpirunでnetgenを複数起動してメッシュ作成
最後に、mergeして全体のメッシュが完成。

1. pmshについて

Parallel NETGENの記述がある
「<http://velizarefremov.com/ms-thesis/>」

Parallel Mesh Generator Using NETGEN

June 13, 2014 / velizar / Uncategorized



pmshについては、
「<https://code.google.com/archive/p/pmsh/>」からsourceをdownload

Projects

Search

About


Project

Source

Issues

Wikis

Downloads

 **pmsh**

Parallel Mesh Generation for OpenFOAM with Netgen

Parallel Mesh Generation for OpenFOAM with Netgen

<https://pmsh.googlecode.com/svn/prace.png>

This work was supported by the PRACE-3IP project funded in part by the EUs 7th Framework Programme (FP7/2012-2014) under grant agreement no. 312763.

OpenFOAM is an open source computational fluid dynamics (CFD) package with a large user base from many areas of engineering and science. An enablement tool called PMSH is developed to generate multi-billion element unstructured tetrahedral meshes for OpenFOAM. PMSH is developed as a wrapper code around the popular open source sequential Netgen mesh generator. Parallelization of the mesh generation process is carried out in five main stages: (i) generation of a coarse volume mesh (ii) partitioning of the coarse mesh to get sub-meshes, each of which is processed by a processor (iii) extraction and refinement of coarse surface mesh to produce fine surface sub-meshes (iv) re-meshing of each fine surface sub-mesh to get a final fine mesh (v) matching of partition boundary vertices followed by global vertex numbering. Test results obtained on an SGI Altix ICE X system with 8192 cores and 14 TB of total memory confirm that our approach does indeed enable us to generate multi-billion element meshes in a scalable way.

pmsh-netgen installation tutorial

:

Project Information

The project was created on Nov 7, 2013.

- License: [GNU GPL v2](#)
- 2 stars
- svn-based source control

Labels:

Academic

OpenFOAM用だが、Netgenをベースとしている為、以下のformatが扱える

import:

Neutral format

Surface mesh format

Universal format

TET format

Pro/ENGINEER neutral format

export:

Neutral format

Surface mesh format

Abaqus format

FLUENT format

FEAP format

Elmer format

Gmsh format

netgenは、「netgen-5.1.targz」をダウンロードする。

最新版は、netgen-6.2だが、6.2にはバグがあり、複数の領域をもつメッシュがきれいに分割できない。

この為、メッシュ作成は、5.1を使う。

コンパイルは、「<https://code.google.com/archive/p/pmesh/>」に従って行うが、configのオプションは、以下に変更した。(ubuntu-16.04)

```
./configure --with-tclconfig=/usr/lib/tcl8.5 --with-tkconfig=/usr/lib/tk8.5 --with-tclinclude=/usr/include/tcl8.5 --with-tkinclude=/usr/include/tcl8.5 -prefix=$INSTALLDIR
```

pmshは、stlファイルから、直接粗いメッシュをきり、直ぐに並列で細かくメッシュを切って行ってしまうので、粗いメッシュの確認ができない。

予め、粗いメッシュを準備して、これをpmshに渡して細かくメッシュをきる様にsourceを修正。

1. 実行状況

pmshで細かくメッシュを切った結果、

- 1) メッシュ形状が粗いメッシュのまま
→ 元々のstl形状に表面をfitさせる事を追加
- 2) 境界条件を設定する為のgroup化が設定できない
→ 境界条件を設定する形状のstlを準備して設定できる
又は、粗いメッシュで設定されているgroupを引き継ぐ
様に追加。

の問題があった為、pmsh実行後、後処理（paraMesh.py）を追加。

<実行方法>

並列数 細分化レベル

```
$ paraMesh.py -np 4 -nl 2 -pre premesh.vol -nd top.stl bottom.stl -sf side.stl  
-f sideFace.stl
```

fit形状 粗いメッシュ 節点group 面group

(groupを引き継ぐ場合)

```
$ unv2vol.py hinge.unv  
$ paraMesh.py -np 4 -nl 3 -pre hinge.vol -f hinge.stl
```

----- paraMesh.py の使い方 -----

<使い方>

paraMesh.py -np <nProc> -nl <numLevel> -pre <premesh> [option <parameter>]

-np <nProcs> 並列数を設定
-nl <numLevel> メッシュ再分割化のレベル
-pre <premesh> vol形式の粗いメッシュfileを指定

[option]

-maxh <maxSize> メッシュを細かく切り直す時の内部の最大メッシュサイズを指定
 default値は、「1000」
-nd <stl> stlファイルの形状のnodeGroupを作成する。（複数指定可）
-sf <stl> stlファイルの形状のsurfaceGroupを作成する（複数指定可）
-f <stl> 節点の位置をstlファイルの形状の位置に移動（複数指定可）
-h ヘルプ
--np -npと同じ
--numlevel -nlと同じ
--premesh -preと同じ
--maxh -maxhと同じ
--ndGrps -ndと同じ
--sfGrps -sfと同じ
--fit -fと同じ
--help -hと同じ

<使用例>

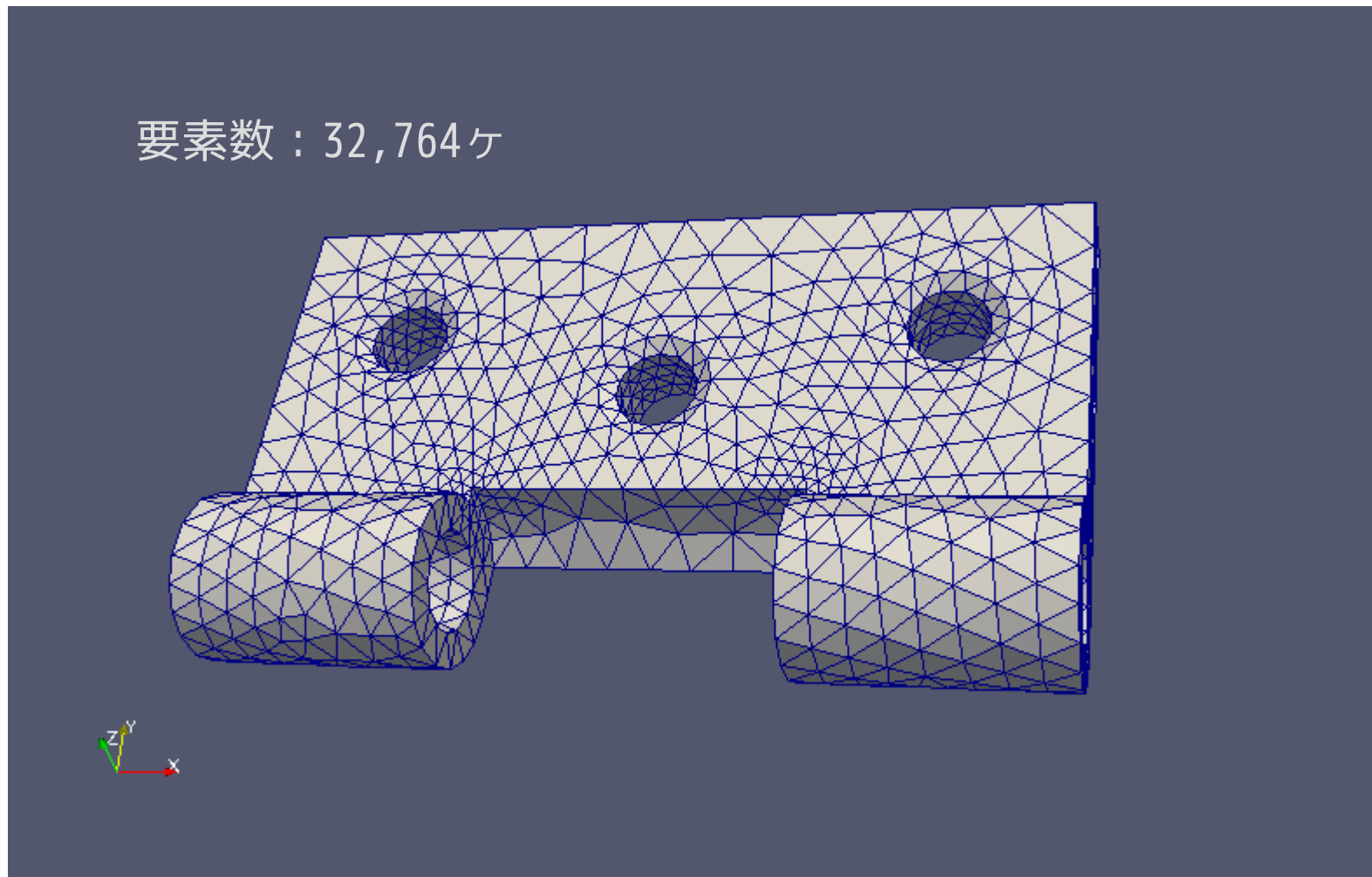
paraMesh.py -np 4 -nl 2 -pre premesh.vol -f sideFace.stl

paraMesh.py -np 4 -nl 2 -pre premesh.vol -nd topFace.stl botomFace.stl -sf sideFace.stl
-f sideFace.stl

粗いメッシュの形状（netgenでメッシュ作成）

```
$ netgen -geofile=hinge.stl -fine -meshfile=hinge.vol -batchmode
```

要素数：32,764ヶ



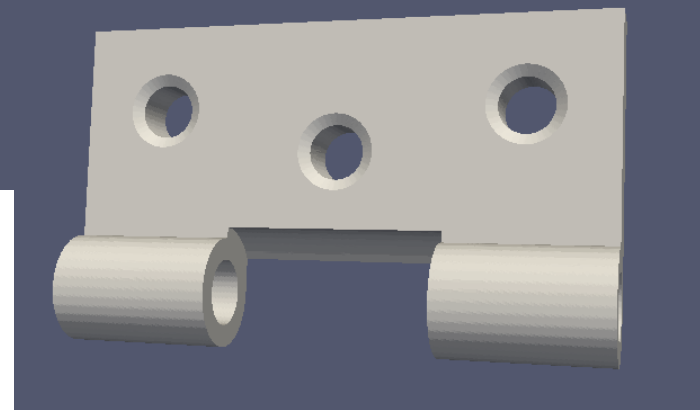
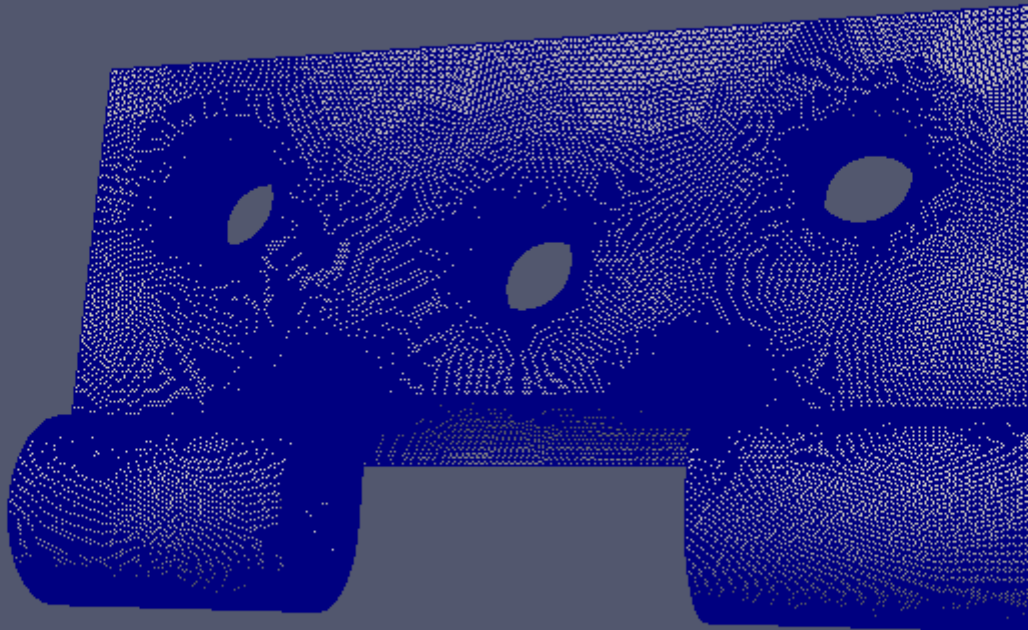
このメッシュを並列で細かく切り直す

並列で細かく切り直した結果

(細分化レベル：3) 元の要素長さを $2^3 = 8$ 分割する

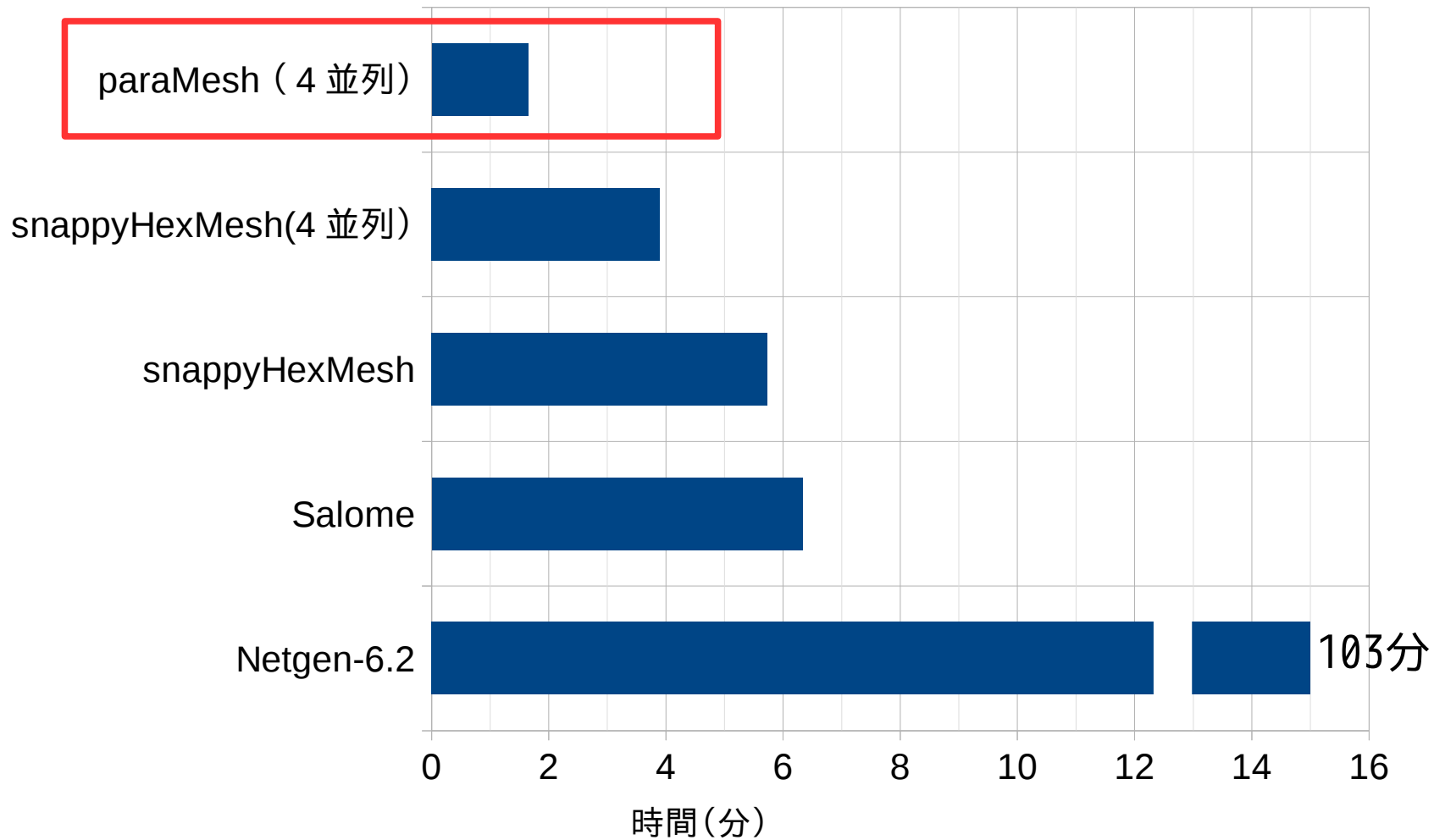
```
$ paraMesh.py -np 4 -nl 3 -pre hinge.vol -f hinge.stl
```

要素数：925,807ヶ



2. メッシュ作成時間の比較

hinge.stlを使って100万要素メッシュを作成するまでの時間



paraMesh (pmsh-netgen) が最も早い。
並列数を増やす事で、さらに短縮が可能。