

東京大学大学院新領域創成科学研究科
人間環境学専攻

平成26年度

修士論文

クラウドコンピューティングを用いた
効率的な構造解析支援システムの開発

2015年1月提出

指導教員 奥田 洋司 教授

47-136696 井原 遊

目次

第1章	序論	1
1.1	研究背景	1
1.2	意義	2
1.3	目的	2
1.4	本論文の構成	3
第2章	構造解析の手順と手法	5
2.1	緒言	5
2.2	有限要素法	5
2.3	連立一次方程式の解法	7
2.3.1	直接法	7
2.3.2	反復法	8
2.4	プリプロセッシング	9
2.4.1	数値計算メッシュ生成	9
2.4.2	解析の種類	10
2.4.3	ソルバ解法の決定	11
2.4.4	並列解析のための領域分割	11
2.5	ソルバ	12
2.6	ポストプロセッシング	12
2.7	結言	13
第3章	クラウドコンピューティング	15
3.1	緒言	15
3.2	5つの主要な特徴	15
3.2.1	オンデマンド・セルフサービス (On-demand self-service)	15
3.2.2	幅広いネットワークアクセス (Broad network access)	15
3.2.3	リソースの共用 (Resource pooling)	15
3.2.4	スピーディな拡張性 (Rapid elasticity)	16
3.2.5	サービスが計測可能であること (Measured Service)	16
3.3	3つのサービスモデル	16
3.3.1	SaaS(Software as a Service)	16
3.3.2	PaaS(Platform as a Service)	16

3.3.3	IaaS(Infrastructure as a Service)	17
3.4	4つの配置モデル	17
3.4.1	プライベートクラウド	17
3.4.2	コミュニティクラウド	17
3.4.3	パブリッククラウド	17
3.4.4	ハイブリッドクラウド	17
3.5	本研究において採用するクラウドモデル	18
3.6	CAEの分野におけるクラウドサービス	18
3.7	結言	19
第4章	連携するソフトウェア, ライブラリ, API	21
4.1	FrontISTR	21
4.2	REVOCAP_Prepost	21
4.2.1	REVOCAP_Mesh	21
4.3	ADVENTURE_Tetmesh	22
4.4	OpenGL	22
4.5	libpng	22
第5章	三次元モデルの二次元画像による操作	23
5.1	緒言	23
5.2	回転	24
5.3	拡大縮小と平行移動	26
5.4	面の表現	27
5.5	ファイル名による画像の特定	28
5.6	結言	28
第6章	開発したプログラムとライブラリ	29
6.1	緒言	29
6.2	可視化系ライブラリ	29
6.2.1	データの取り扱い	29
6.2.2	モデルの読み込み	29
6.2.3	データの変換と保存	31
6.2.4	境界抽出と表面分割	31
6.2.5	モデルの描画	34
6.2.6	画像の生成	37

6.2.7	描画処理の高速化	39
6.2.8	画像出力の並列化	41
6.3	可視化系ソフトウェア	42
6.3.1	形状データ可視化	42
6.3.2	FEM メッシュデータ可視化	42
6.3.3	FrontISTR 入力ファイル生成・可視化	42
6.3.4	数値計算結果可視化	43
6.4	自動四面体メッシュ連携	43
6.5	ジョブ管理	45
6.6	結言	45
第 7 章	クラウド CAE システムの構築	47
7.1	緒言	47
7.2	サーバの準備	47
7.3	Web サービススクリプト	47
7.4	ローカルバッチジョブシステム	48
7.5	外部計算機	49
7.6	リモートジョブ送信機能	49
7.7	データ整理・管理機能	50
7.8	結言	50
第 8 章	システムの外観と計算の実例	57
8.1	緒言	57
8.2	Gear モデルでの線形弾性静解析	58
8.2.1	トップページ	58
8.2.2	モデルの登録	59
8.2.3	形状データリスト	60
8.2.4	形状データビュー	61
8.2.5	メッシュリスト	62
8.2.6	メッシュビュー	63
8.2.7	境界条件の確認	67
8.2.8	解析モデルビュー	68
8.2.9	解析結果	69
8.3	結言	71

第 9 章 システムの評価	73
9.1 緒言	73
9.2 評価	73
9.3 結言	73
第 10 章 結論	75
10.1 結論	75
10.2 今後の展望	76
謝辞	77
参考文献	81

目次

1.1	Concept of cloud CAE system	3
2.1	Workflow of CAE analysis	5
2.2	Finite element mesh	6
2.3	Algorithm of CG method	9
2.4	Classification of geometric nonlinearities	10
2.5	Solving equations method suitability by form	11
2.6	Domain decomposition	12
4.1	Revocap mesh library	22
5.1	Handling 3D model by 2D image	23
5.2	Rotation of x,y,z -axis	24
5.3	Rotation of θ_1,θ_2	25
5.4	Zoom of model by image	26
5.5	How to set color to selected face	27
5.6	Representation of surface method	27
5.7	Javascript for swapping image and face	28
6.1	Creating instance for handling model data and loading from file	30
6.2	Converting file type	31
6.3	Boundary Extraction and surface operation	33
6.4	Preparing to Model projection	35
6.5	Wireframe	36
6.6	Surface	36
6.7	How to creating image	37
6.8	How to creating image	38
6.9	How to use displaylist	40
6.10	Convering captured image on memory to image file.	41
6.11	Visualizing software	42
6.12	Condition script	43
6.13	Execution mesher script	44
6.14	Job management script	46

7.1	php include	48
7.2	Cron for Job manage script	48
7.3	Remote job handler script	50
7.4	Job creating script for FX10 (1)	51
7.5	Job creating script for FX10 (2)	52
7.6	Job creating script for FX10 (3)	53
7.7	Substitute script for job creating script in realtime computer	54
7.8	Polling script	55
8.1	Gear Model	57
8.2	Toppage of Cloud CAE system	58
8.3	Model upload page	59
8.4	CAD model list page	60
8.5	CAD model view page	61
8.6	Mesh list page	62
8.7	Mesh view page(1)	63
8.8	Mesh view page(2)	64
8.9	Mesh view page(3)	65
8.10	Mesh view and setting condition	66
8.11	Mesh with condition view	67
8.12	Model view	68
8.13	Deformed image of geer model FEM result	69
8.14	Displacement contour of geer model FEM result	70
8.15	Von mises stress contour of geer model FEM result	71

表 目 次

7.1 Spec of CAE cloud server	47
7.2 Spec of calculation server	49

第1章 序論

1.1 研究背景

世界のスーパーコンピュータの上位 500 位を評価付けるプロジェクトである TOP500[1]によると 2014 年 11 月現在の最速のスーパーコンピュータは、33.9PFLOPS である。20 年前、1994 年 11 月においての最速のスーパーコンピュータは 170GFLOPS であったことから、過去 20 年間で概ね 20 万倍になっており、スーパーコンピュータの演算性能は著しく進歩している。そこで、過去において計算時間がボトルネックになっていた数値計算を主体としたシミュレーション研究は、現在においては計算時間が問題になることがなくなった一方で、演算が高速化したために、莫大なデータ処理に費やす時間が顕著になっているという現状がある。

CAE(Computer Aided Engineering, コンピュータ支援工学) は、近年のコンピュータ技術の飛躍的な進歩に従って様々な製品の設計、開発において用いられるようになってきている。それに伴って、様々な CAE ソフトウェアが開発され活用されている。従って、現在はシミュレーションを容易に行うことできる環境が整いつつあり、大規模化・精緻化が進みつつあるが、シミュレーション研究において様々なツール開発、そのツールを活用する技術の習得、多くの分野の知識、結果の処理、パラメータの把握などが必要であり、研究者の負担となっている。また、他人の作ったツールを活用することや、結果を活用することは十分に環境が整備されているとは言い難い。これらの課題をうまく扱えない場合、現状のシミュレーションは、研究・開発業務に高度化、効率化をもたらすとは限らない [2]。

近年では、コンピュータ資源の利用形態として、クラウドコンピューティングが提唱され注目されている。CAE の分野においても様々な形でクラウドサービスが提供されている。一例として、ソフトウェアライセンスと仮想化環境のプラットフォームを提供する「TC クラウド」[3] といったサービスが提供されるようになってきた。このような PaaS 型クラウドサービス以外にも CAE に利用可能なクラウドサービスには、IaaS 型クラウドサービスがあげられる。特に CPU 仮想化技術の進歩によって、「Amazon EC2」[4] や「さくらのクラウド」[5] といった IaaS 型クラウドは普及しつつあり、使用時間による課金である特徴から CAE に用いることで計算機コストを低減させる可能性がある。SaaS 型クラウドサービスとして CAE に活用した例は、ADVENTURE を Web サービス化した研究 [6, 7] がある。この和田らによる研究は、Web 上で CAE のプロセスが実行できるものであるが、サーバ処理が重いことや、対話性が悪いなどの問題点が残された。また、境界条件設定の柔軟性が良くないなどの問題点も残された。また、並列有限要素法構造解析ソフトウェア「FrontISTR」とエンジニアリングデータ管理システム「ASNARO」を用いてクラウドサービス化した研究

[8]では、プリ処理を支援する仕組みがないことや、商用ソフトウェアと連携が必要なことなどの問題点が残されている。

CAEでは、解析の規模によって、高性能なコンピュータを必要とし、その設備の費用は少なくなく、一度にインシヤルコストとしてかかるため、既に活用しているユーザ以外においては導入の決断が難しい。スーパーコンピュータの使用料は年度単位であることや、購入できる時間単位が大きく、使用可能性の検討段階においては導入を躊躇せざるを得ない場合がある。一方で、IaaS型クラウドサービスは使用料の単位が非常に細かいことから、この点において、強力な支援ツールとなりうる可能性を秘めている。

また、市販のCAEソフトウェアはライセンス料が安くなく、導入は容易ではない。一方、オープンソース・ソフトウェアとして提供されているCAEソフトウェアは、機能が不十分である場合や、オープンソースであるがためにコンパイルといった計算機に関する知識を必要とし、その障壁は小さくない。したがって、オープンソース・ソフトウェアを十分に広くCAEに活かすためには何らかの支援が必要とされる現状がある。

1.2 意義

CAEを行う上で、オープンソースのソフトウェアを用いる場合、場合によってはソフトウェアのコンパイルから必要となり、パラメータ設定など多くのソフトウェアの連携、テキストエディタでの編集など、多くの手間がかかっている。このような点から、多くの研究者や技術者は商用ソフトウェアを利用している現状があるが、その場合においても結果データを整理したり、データを管理したりする必要は当然にあり、システムがない場合、このことは研究に直接関係しないところで労力をさくことになる。オープンソース・ソフトウェア、商用ソフトウェアのどちらを使う場合にせよ、スーパーコンピュータで大規模または大規模とまで言わないが並列計算をしようとする、さらに問題は複雑かつ多くなる。まず、最低限、並列計算機特有のノード間並列などの事項についてある程度の知識が必要である。次に、商用ソフトウェアでは、クローズドソース故に計算機環境によっては使えないことがあり、使える場合においても並列数によって極めて高額なライセンス料が発生する。オープンソース・ソフトウェアでは、数値計算を実行する前までに、提供されたコンパイラ特有のエラーや結果の検証などのデバッグ作業に多くの時間を費やすことになる。これらのノウハウは共有されないこともしばしばある。したがって、これらの問題をできる限り解決させ、研究者や技術者からブラックボックス化できるシステムがあれば、より効率的なCAEを行うことができると思われる。

1.3 目的

本研究では、クラウドコンピューティング技術および有限要素法構造解析ソフトウェア「FrontISTR」[9]を利用した、プリポストを含めた統合CAEクラウド支援環境(Fig. 1.1)の構築を行い、CAEの普及を促進することを目的とする。

具体的には、CAD ファイルから一貫して Web 上で CAE の支援を行うことにより、低コストかつ簡単な操作でユーザに提供できるシステムを提案すること、ユーザの作業やデータ整理を自動化・簡略化できる機能を提案したシステムに追加することによって、ユーザの負担やユーザ間の研究連携を促進することを目指す。

その目的を達成するために、既存のプログラムとの連携や、Web 上で展開するための様々な CAE 連携ライブラリを作成する。

これらのことによって、CAE 未利用の現場への新規導入の障壁を減らすこと、CAE を利用してきた現場に負担を軽減した CAE ができるようにすること、ライブラリの活用により CAE のクラウドサービス化研究の基礎になることを目指す。

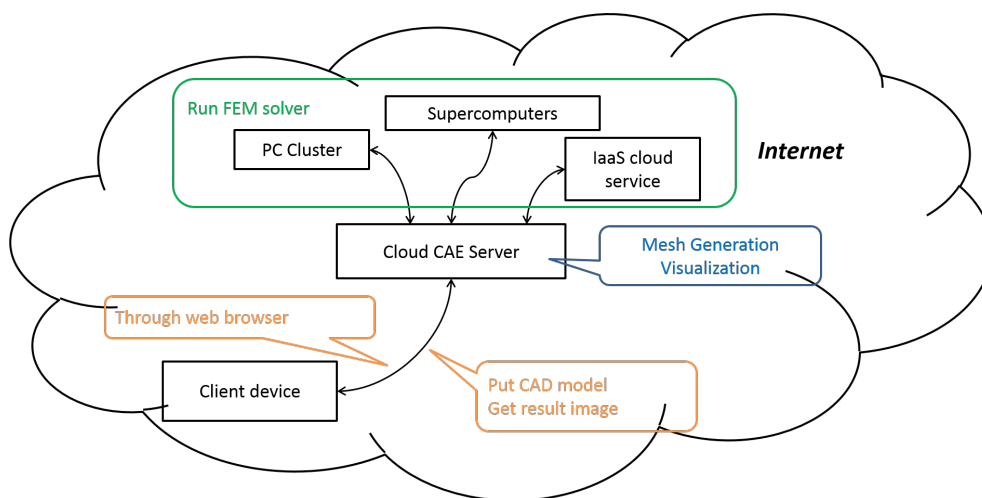


Fig 1.1: Concept of cloud CAE system

1.4 本論文の構成

本研究の目的である、クラウドコンピューティングを活用した効率的な構造解析システムの開発に向けて、本論文は次のように議論を展開する。

第2章において、構造解析の一般的な手法を概観しその手順を整理し説明する。次に、第3章においてクラウドについて説明を行い同時に CAE に活用した研究をレビューする。続いて、第4章において、使用するソフトウェアやライブラリについて説明する。第5章で、Web 上で CAE を行う手法について説明し、6章では、本研究において開発したライブラリやソフトウェアの技術的説明を行う。第7章において、そのライブラリやソフトウェアを組み込んだシステム構成を提案し、説明を行う。第8章において、そのシステムを利用した計算の実例とシステムの外観を紹介し、第9章においてその評価を行い、第10章において、結論と今後の展望を述べる。

第2章 構造解析の手順と手法

2.1 緒言

本章では、構造解析の手順とそれに用いられる手法について述べる。構造解析の手順は大きく分けると、プリプロセッシング、ソルバ、ポストプロセッシングの3つのステップに分けることができる (Fig 2.1)。プリプロセッサは、物理特性を検討しモデルの選択、ソルバで解くための数値計算用のメッシュを作成、境界条件を設定など解析の準備の部分である。ソルバは、数値計算を行う部分であり、この部分が計算負荷の大部分を占める。ポストプロセッサは、ソルバから得られた計算結果をユーザが理解できる形に置き換える部分である。

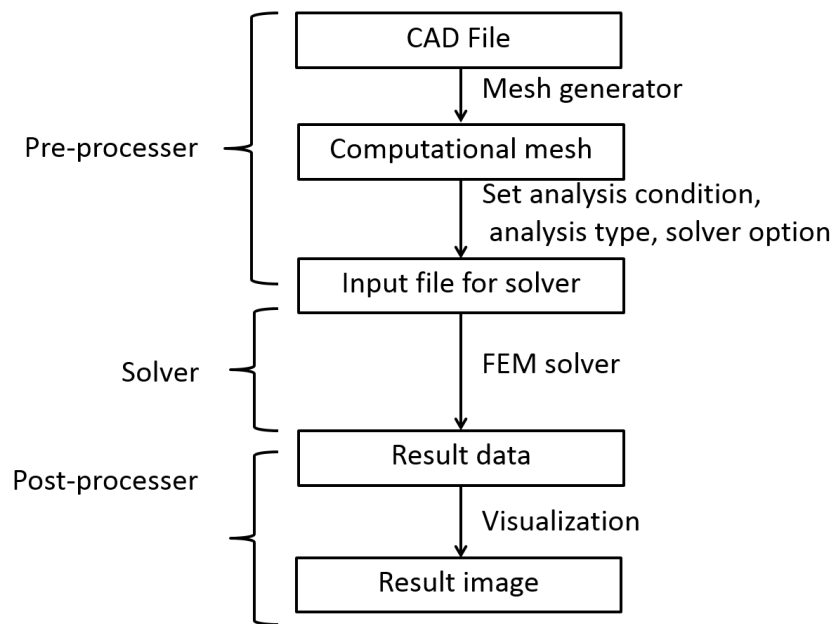


Fig 2.1: Workflow of CAE analysis

2.2 有限要素法

自然界における現象は、連続体として扱われ、偏微分方程式で記述される。連続体問題の偏微分方程式は一般に厳密解を求めることが難しいため、数値計算が行われ近似解として解を求められる。近似解を求めるために、本来は連続な物体を有限個の領域に離散化して表現

する必要がある。有限要素法は、偏微分方程式を近似的に解くための離散化手法の一つであり、産業基盤分野の数値シミュレーションにおいて特に盛んに用いられている。

有限要素法を用いた離散化の手段について述べると、まず解くべき領域を Fig 2.2 のように要素と呼ばれる有限の大きさの小領域に分割する Fig 2.2 では、三角形要素で分割したが、

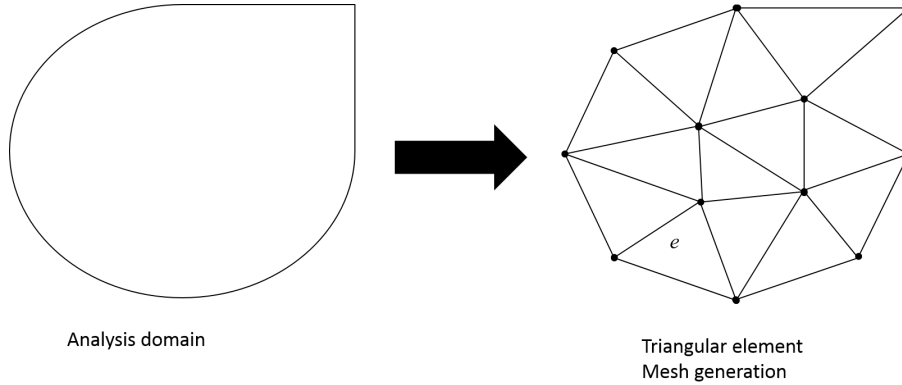


Fig 2.2: Finite element mesh

要素の形状は様々なものがあり、特性に応じて利用されている。また、要素の頂点のことを節点と言う。

ある三角形要素 e に注目すると、要素内の関数 S は、構成する節点の S の値、 S_1, S_2, S_3 を用いて内挿近似する。既知の内挿関数 N_1, N_2, N_3 とすると、 e 内の関数 S は

$$\tilde{S} = N_1(x, y)S_1 + N_2(x, y)S_2 + N_3(x, y)S_3 \quad (2.1)$$

と表すことができる。

解くべき方程式を $F(S) = 0$ に、 \tilde{S} を代入すると、 \tilde{S} は厳密解でないので、

$$R \equiv F(\tilde{S}) \neq 0 \quad (2.2)$$

である。 R を残差と言う。

残差を小さくする条件を考える。積分領域を e として、 w を適当に与えられる重み関数としたとき、重み付き残差法は

$$\int_e w R dx dy = 0 \quad (2.3)$$

となる。領域内の全要素について式 2.3 を求めることによって、全ての節点における未知数についての連立一次方程式が得られる。

各要素について重み関数にガラーキン法を適用し、偏微分方程式に対して、Green-Gauss 定理を適用して弱形式化する。各要素において積分して、要素マトリクスを得て、要素マトリクスを全体マトリクスに足す。

数値解を求めることは連立一次方程式を解くことに帰着される。

2.3 連立一次方程式の解法

有限要素法解析において、最も計算コストの高い処理がこの連立一次方程式を解く部分である。連立一次方程式 $A\mathbf{x} = \mathbf{b}$ の解法を大別すると、直接法と反復法の2種類がある。直接法とは、ガウスの消去法に代表されるように、方程式から変数を消去していくこと、つまり式変形によって解を求める方法である。丸め誤差がないとき、有限回の計算で解を得ることができ、ロバスト性に優れた手法である。反復法は、適当な近似解から出発して、一定の演算を反復することで近似解の精度を高める方法である。

2.3.1 直接法

直接法の最大の利点は、ロバスト性である。反復法では収束しない問題も直接法では一定の演算を行うことで解が得られるため、悪条件問題では必須となっている。直接法における計算量は行列サイズの $\mathcal{O}(n^3)$ であり、必要とする主記憶容量は行列サイズの $\mathcal{O}(n^2)$ である。反復法と比較すると、問題規模の増大に対して計算量、必要な主記憶容量の増加が急激であると言える。計算順序に依存関係があるために、並列化を効率よく行うことが難しい。並列計算に対応したライブラリ MUMPS[10, 11] や、WSMP[12], SuperLU_DIST[13], Pastix[14] などがあるが、反復法のような並列化効率は得られない。そのため、解析規模が数十万自由度程度までの問題ではよく使われるものの、大規模問題では利用が難しい。

直接法の主な例として、変数消去法や Gauss の消去法、LU 分解を利用する方法がある [15]。

LU 分解は、 $A = LU$ を満たす下三角行列 L と上三角行列 U に分解する方法であり、 A を、

$$A = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ * & 1 & 0 & \cdots & 0 \\ * & * & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ * & * & * & \cdots & 1 \end{pmatrix} \begin{pmatrix} * & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ 0 & 0 & * & \cdots & * \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & * \end{pmatrix} \equiv LU \quad (2.4)$$

と分解できるとき、

$$A\mathbf{x} = \mathbf{b} \quad (2.5)$$

に代入し、

$$LU\mathbf{x} = \mathbf{b} \quad (2.6)$$

と書くことができる。ここで、

$$L\mathbf{y} = \mathbf{b} \quad (2.7)$$

とおくと、

$$U\mathbf{x} = \mathbf{y} \quad (2.8)$$

である。

式 2.8 を解き、式 2.7 を解くことで、逆行列を求めることなく解 \mathbf{x} を得ることができる。

LU 分解では、一度分解した LU を保持しておけば、多数の異なる右辺ベクトル \mathbf{b} を持つ問題には 2.8, 式 2.7 を解くだけで解が得られることから、このような問題には極めて有効である。

2.3.2 反復法

反復法では、解くべき連立一次方程式 $A\mathbf{x} = \mathbf{b}$ を、初期値 \mathbf{x}_0 を適当に取り、反復計算を繰り返すことで近似解を得ることを目標とする。

反復解法を大別すると定常反復法と非定常反復法に分けられる。定常反復法の主な例として、Jacobi 法、Gauss-Seidel 法、逐次加速緩和法などが挙げられるが、対角優位であれば収束性が良いが、そうでないときに時間がかかる欠点がある。これらの定常反復法の場合、計算量は反復回数を k とすると $O(kn^2)$ である。ただし、 k が無限大になることもある。

一方、非定常反復法は $\mathbf{x}_{(n)} \rightarrow \mathbf{x}_{(n+1)}$ の漸化式が、 $\mathbf{x}_{(n)}$ に関して非線形な反復解法であり、一般に、定常反復法より収束が早い傾向がある。Krylov 部分空間法は、大規模連立一次方程式に対する強力な解法であることから、有限要素法解析では一般的によく用いられている。Krylov 部分空間とは、係数行列 A と残差ベクトル $\mathbf{r} = \mathbf{b} - A\mathbf{x}$ で作れる Krylov 部分空間である。Krylov 部分空間法の主な例として、共役勾配法 (CG 法: Conjugate gradient method), (CG 法: Conjugate gradient method), 最小残差法 (MINRES 法: Minimum Residual method), それを一般化した一般化最小残差法 (GMRES 法: Minimum Residual method) などがあげられる。これらの Krylov 部分空間法の場合、理論的には $O(n^3)$ の計算量が必要であるが、実用上は、 n よりも十分に少ない反復で目標とする解の精度に達するため、 n が十分に大きいときには $O(n^2)$ と見ることができる。

共役勾配法においては、丸め誤差がないとき、目標とする精度まで有限回の反復で収束する特徴から、大規模解析において良く用いられる。また、主に行列とベクトルの積の演算で計算を進めることから、問題規模の増大に対する計算量とメモリ使用量の増加は穏やかであり、並列計算に対する適合性は高い。共役勾配法のアルゴリズムを Fig 2.3 に示す。

CG 法では、条件数が大きい問題、つまり悪条件 (ill-conditioned) 問題の収束性の改善の手段として前処理 (Preconditioning) がある。前処理には、大きく分けると、係数行列 A を近似する不完全分解、逆行列 A^{-1} を近似する近似逆行列分解がある [16]。前者は、コレスキー分解 [17] がよく知られ、後者は A-直交過程にもとづいて逆行列 A^{-1} を近似分解する近似逆行列 (Approximate Inverse, AINV)[18] が知られている。また、AINV における近似分解を工夫し、CG 法の収束性の安定化をした安定化近似逆行列 (Stabilized-AINV, SAINV)[19] もある。さらに、SAINV から得られた分解因子を用いたロバスト不完全分解 (Robust Incomplete Factorization, RIF)[20] も提案されている。

なお、反復法では、直接法のようなロバスト性はないため、悪条件な行列の問題においては強力な前処理を適応したとしても収束できないこともある。

```

1: let  $\mathbf{x}_0$  be an initial approximation
2:  $k = 0$ 
3:  $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$ 
4:  $\mathbf{p}_0 = \mathbf{r}_0$ 
5: while  $\|\mathbf{r}_{k+1}\| < \epsilon\|\mathbf{b}\|$  do
6:    $\alpha_k = \frac{(\mathbf{r}_k, \mathbf{p}_k)}{(\mathbf{p}_k, A\mathbf{p}_k)}$ 
7:    $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$ 
8:    $\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k A\mathbf{p}_k$ 
9:    $\beta_k = -\frac{(\mathbf{r}_{k+1}, A\mathbf{p}_k)}{(\mathbf{p}_k, A\mathbf{p}_k)}$ 
10:   $\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$ 
11:   $k = k + 1$ 
12: end while

```

Fig 2.3: Algorithm of CG method

2.4 プリプロセッシング

CAEにおけるプリプロセッシングとは、構造解析を行う上で、ソルバの入力ファイルとして必要なデータを揃える工程である。本論文では、形状データがある前提であるので、まず、CADの形状データの出力からをプリプロセッシングとして定義する。CADの形状データの出力には、IGES,STEP形式などの中間ファイルの標準があり、異なるソフトウェア間での利用が可能である。また、近年は3Dプリンタの普及で三角形表面パッチによって構成されるデータ、STL(Standard Triangulated Language, Stereolithographyとも)フォーマットも用いられている。本研究では、これらの標準形式を入力形状データとして用いることにする。

2.4.1 数値計算メッシュ生成

形状データは、三次元の形状のデータであり、これを有限要素法による構造解析をするために、メッシュと呼ばれる小領域に分割する必要がある。三次元四面体要素（以下、四面体要素という）でメッシュ生成するのであれば、多くのオープンソースの四面体メッシャがある[21, 22]。一般的に四面体メッシャでは、入力形状ファイルがIGES,STEP形式であれば、まず表面の抽出を行うことで、三角形表面パッチを生成する。生成した表面パッチを平滑化し、表面メッシュを生成する。生成した表面メッシュを頂点とし、内部に四面体メッシュを生成する流れである。

2.4.2 解析の種類

有限要素法による構造解析で、どのような材料でどのような挙動をするか事前に検討し、計算負荷を考えながら解析の種類を決定しなければならない。

解析の種類は複数の視点から、分けることができる。

まず、解くべき方程式の違いである。大きく分けると、静解析と動解析がある。

剛性マトリクスを K 、減衰マトリクスを C 、質量マトリクスを M 、変位ベクトルを \mathbf{u} 、荷重ベクトルを \mathbf{F} とする。

静解析では、剛性方程式

$$K\mathbf{u} = \mathbf{F} \quad (2.9)$$

を解くことになる。

動解析では、運動方程式

$$M\ddot{\mathbf{u}} + C\dot{\mathbf{u}} + K\mathbf{u} = \mathbf{F} \quad (2.10)$$

を解くことになる。

静解析は慣性力を考慮しない場合に選択し、動解析は慣性力を考慮する場合に選択するが、動解析は静解析と比較して計算負荷が大きいことに留意しなければならない。

運動方程式から、減衰と荷重を取り除くと、

$$M\ddot{\mathbf{u}} + K\mathbf{u} = 0 \quad (2.11)$$

となり、固有値解析の式となる。

次に、幾何学的非線形性の有無がある。これは、扱う構造物の変位の規模によって、荷重・応力、ひずみ・変位の関係を線形と近似できるかという問題である。幾何学的線形の例は、微小変形問題である。幾何学的非線形 (Fig 2.4) の例は、大変形問題である。大変形問題は、ひずみが微小であるかによって、微小ひずみ問題と有限ひずみ問題にわけられる。大変形問題の解法では、Total Lagrange 法、Updated Lagrange 法があり、微小ひずみ問題には Total Lagrange 法が適し、大ひずみ問題には Update Lagrange 法が適している。幾何学的非線形性を考慮した場合、幾何学的線形を考慮する場合と比較して計算負荷が増大することに留意する必要がある。

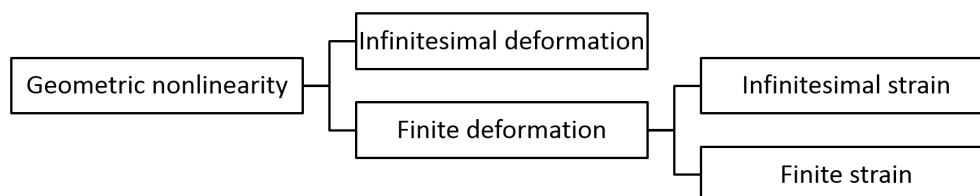


Fig 2.4: Classification of geometric nonlinearities

続いて、材料非線形性の有無がある。構成式は、材料の特性を数学的に表したものであり、その構成式が、線形と近似できるかという問題である。材料線形問題の例は、等方的弾性問題があげられる。材料非線形問題の例は、弾塑性問題 [23]、超弾性問題、粘弾性問題、クリープ問題、超弾性問題があげられる。

最後に、境界非線形性である。接触問題や摩擦を考慮すると境界非線形問題となる。

実問題を厳密に考慮するととなると、様々な非線形性を考慮した複雑な解析になることが多いが、計算負荷や収束性とのトレードオフで決定することになる。

2.4.3 ソルバ解法の決定

解析の種類とメッシュを決定すると、その問題の行列を解くソルバの解法を決定しなければならない。2.3章で、連立一次方程式の解法を説明したように大きく分けて直接法か反復法を選択することになる。どちらを選択することが適するかは、問題に依存する。

まず、重要なのは問題への適合性である。悪条件問題においては反復法を使用すると収束しない場合もある。形状が薄いモデル、長いモデルでは、行列は悪条件になることが多く、反復法の収束性が良くないが、直接法においては Fill-in が発生しにくく効率的に解きやすい。反対に、立方体のようなモデルでは、直接法では Fill-in が発生しやすく解きにくい。行列の条件数は低い、良条件 (well-conditioned) の問題になる傾向がある (Fig 2.5)。次に、解析の規模を考え、大規模な問題を直接法で解く場合、メモリ制約に当たらないかを考えなければならない。

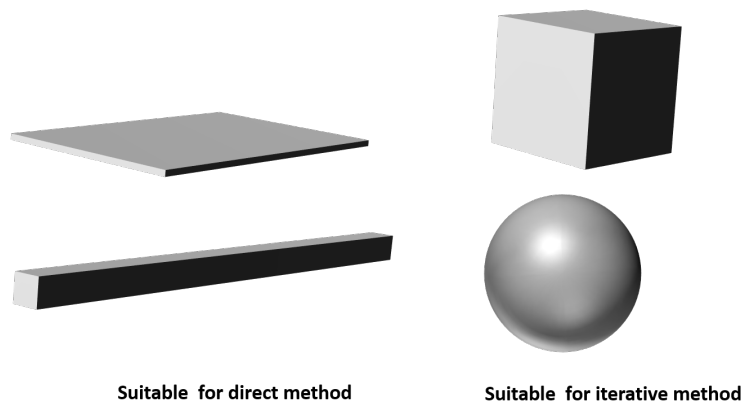


Fig 2.5: Solving equations method suitability by form

2.4.4 並列解析のための領域分割

FrontISTR において並列計算するには、全体メッシュを MPI プロセスと同じ数の部分に分割した分散メッシュの生成が必要である (Fig 2.6)。メッシュをグラフに変換すると、グラフの頂点は節点と対応し、エッジは節点間の接続関係とする。グラフ分割ツール METIS[24]

等を利用し、グラフを分割していく。ここで、分割することによるエッジカットを少なくすることで、並列計算での通信の削減ができるので、並列効率が高まる。

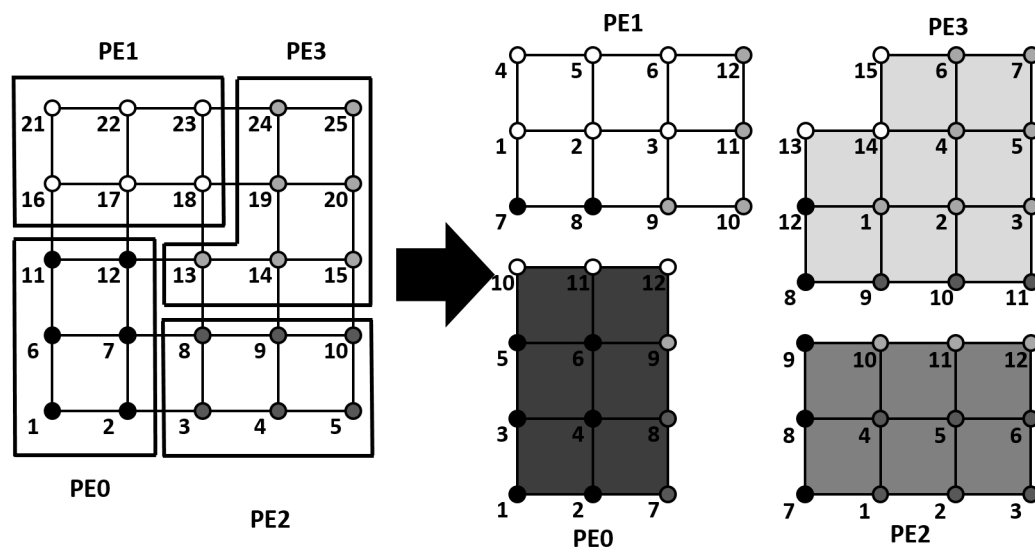


Fig 2.6: Domain decomposition

2.5 ソルバ

作成されたメッシュと解析条件を、数値解析のソルバに入力ファイルとして設定し、実行することで、計算が行われる。一般に高い計算能力を求められる部分は、この部分であり、メッシュファイルと解析条件から生成された剛性行列を解くための演算を行う。CAEに用いられるソルバには、MSCNastran, ANSYS, ABAQUS, AdventureSolid 等様々なソフトウェアがあり提供されている。

2.6 ポストプロセッシング

ソルバの結果として得られた結果ファイルの内容を理解するために可視化処理が行われる。形状を図示し、変位や応力を色彩で表現したり、その変形図を作成したりすることで、その結果を技術者が検討できるものとなる。この処理をポストプロセッシングと言う。CAEに用いられるポストプロセッサには、Femap, Paraview, MicroAVS 等様々なソフトウェアが提供されている。

2.7 結言

本章では，構造解析の手順とそのステップ，構造解析に用いられる有限要素法と連立一次方程式の解法について述べた．

第3章 クラウドコンピューティング

3.1 緒言

クラウドコンピューティングは、様々な運用形態があり、定義も様々で、明確でなくバズワード化しているとの指摘もあるため、以下のように米国国立標準技術研究所 (NIST: National Institute of Standards and Technology)[25] の定義を引きたい。

クラウドコンピューティングは、共用の構成可能なコンピューティングリソースに、どこからでも、簡便に、必要に応じて、ネットワーク経由でアクセスすることを可能とするモデルであり、少ない手続きで速やかに提供されるものである。

ここで、コンピューティングリソースとは、ネットワーク、サーバ群、ストレージ、アプリケーション、及びそれらによるサービスを意味している。このようなクラウドモデルは、5つの主要な特徴、3つのサービスモデル、4つの配置モデルから構成される。

3.2 5つの主要な特徴

3.2.1 オンデマンド・セルフサービス (On-demand self-service)

ユーザは、サービスプロバイダと人的な介在なしに、必要に応じて自動的に、サーバの稼働時間、ネットワーク、ストレージ等のコンピューティングリソースを設定できる。

3.2.2 幅広いネットワークアクセス (Broad network access)

携帯電話、タブレット、ラップトップ、ワークステーションなど、シンクライアントかシッククライアントか問わず、機能や形態の異なるクライアントから標準的な仕組みでネットワークを通じて利用することができる。

3.2.3 リソースの共用 (Resource pooling)

クラウドサービスプロバイダのコンピューティングリソース (ストレージ、演算処理能力、メモリ、ネットワーク帯域など) は集積され、多数のユーザにいつでも提供可能なように共

有モデルで提供される。物理的・仮想的リソースは、ユーザの需要に応じて、動的に割り当てられる。物理的な所在に制約されないので、ユーザは一般的にその正確な所在地を知ったり、コントロールしたりできないが、プロバイダによっては抽象的なレベル（例えば、国や州、データセンターなど）で特定できることもある。

3.2.4 スピーディな拡張性 (Rapid elasticity)

クラウドサービスのコンピューティングリソースは、迅速に提供されユーザの需要に対して即座にスケールアウト（拡張）、スケールイン（縮小）できる。ユーザから見ると、利用できるリソースは制限がないように見え、いつでも購入できる。

3.2.5 サービスが計測可能であること (Measured Service)

クラウドシステムでは、リソースの利用状況を計測できる機能を持ち、その機能を用いてリソースの使用を最適化できる。リソースの計測は、サービスタイプ別（ストレージ、処理能力、帯域、実利用中のユーザアカウント数）に行われリソースの利用状況はサービスプロバイダ、ユーザ双方に報告される。

3.3 3つのサービスモデル

3.3.1 SaaS(Software as a Service)

SaaS 型クラウドは、ユーザはサービスプロバイダの提供するアプリケーションを扱うことができるサービスである。ユーザが Web ブラウザやメールといったインターフェイスを通じてアクセスする。ユーザは、サービスを提供しているサーバのネットワークやオペレーションシステムやプログラムなどを管理することができない仕組みである。Web メールや動画配信サービスなど多くの Web サービスはこれに当てはまるだろう。

3.3.2 PaaS(Platform as a Service)

PaaS 型クラウドは、ユーザはサービスプロバイダがサポートしているプログラミング言語やライブラリ、サービス及びツールなどが利用してアプリケーションを実装することができる。SaaS 型と同様にユーザがコンピュータの管理権はないが、そのアプリケーションについての設定などは管理権を持つことができる。

共用の Web 配信用レンタルサーバ、スーパーコンピュータのアカウントを例にあげる。

3.3.3 IaaS(Infrastructure as a Service)

IaaS型クラウドは、3つのサービスモデルの中で最も自由度が高い。ユーザは、クラウド上のプロセッサ、ストレージ、ネットワークなどのリソースを任意に配置でき、任意のオペレーションシステム、プログラムを実行させることができるが、計算機の物理的資源を管理することはできない。特定のネットワークコンポーネント機器の管理権も持つ場合がある。

Amazon EC2[4]のようにユーザが、サーバ機器を用意することなくサーバ機器をレンタルしたかのようなサービスのことである。

3.4 4つの配置モデル

3.4.1 プライベートクラウド

プライベートクラウドは、企業や団体などの単一組織によって運営されているプラットフォームである。このモデルの場合、所有、管理、運用はその組織が担うか、第三者に委託され、設置場所としては、組織の施設内または外部である。

3.4.2 コミュニティクラウド

コミュニティクラウドは共通の目的を持つ、複数組織からなる利用者の共同体の専用として提供される。このモデルの場合、所有、管理、運用はその複数組織のいずれかが担うか共同運営され、または第三者に委託され、設置場所としては、組織の施設内または外部である。

3.4.3 パブリッククラウド

パブリッククラウドは、一般ユーザを対象とした配置モデルである。所有、管理、運用は企業や学術機関、政府など様々で、存在場所はクラウドサービスプロバイダの施設内である。

3.4.4 ハイブリッドクラウド

プライベートクラウド・コミュニティクラウド・パブリッククラウドのうち2つ以上から構成されるプラットフォームである。各クラウドサービスは独立して存在しているものの、標準化された技術で相互接続されており、データとアプリケーションの相互運用、移転可能性などがある。

3.5 本研究において採用するクラウドモデル

本研究においては、「FrontISTR」[9]、「REVOCAP_prepost」[26]を用い、作成したライブラリを組み合わせアプリケーション化し、ユーザ管理を含めて実現し、グループ間での共用もできるハイブリッド型で提供を行う。Web ベースで提供することを目指しているため、SaaS 型のサービスモデルとして提供する。

3.6 CAE の分野におけるクラウドサービス

CAE 分野にも様々なクラウドサービスが提供されるようになってきている。一例として、ソフトウェアライセンスと仮想化計算機環境のプラットフォームを提供する「TC クラウド」[3]といったサービスが提供されるようになってきている。このサービスは、PaaS 型クラウドサービスであると考えられる。解析ソフトウェアのライセンスと計算時間が借りるサービスと考えると理解しやすい。確かに、CAE 利用者が HPC 環境を揃えて、ソフトウェアライセンスを購入することに比べれば利便性の向上になるとは言える。しかし、このようなシステムでは、CAE とは言いつつも計算機とライセンスが主体であり、CAE のプリポスト機能については、リモートデスクトップによって実現していることからわかるように、CAE 全体を直接クラウドサービス化したとは言えず、クラウドとしての特徴を十分に活かして切れていない。

クラウドコンピューティングという言葉が登場する前の研究であるが、SaaS 型クラウドサービスの定義に当てはまるサービスとして CAE を提供した例は、ADVENTURE[22]を Web サービス化した研究 [6, 7] がある。これら和田らによる研究は、Web 上で CAE のプロセスが実行できるなど進歩的なものであったが、当時の低いコンピュータスペックもあり、サーバ処理の重さや、対話性が不十分さなどの問題点が残された。また、境界条件設定の柔軟性が良くないなどの問題点もあり、クラウドサービスとして CAE に活かすというには十分にではなかった。これらのことから、このサービスは作成されたものの十分に発展・活用されなかった。

他にも、FrontISTR[9]と ASNARO[27]を用いてクラウドサービス化した研究 [8]では、オープンソース・ソフトウェアである FrontISTR を用いたことでソルバ部分ではコストの掛からない CAE であったが、ASNARO は商用ソフトウェアであり、経済性のメリットは十分に生かせなかった。また、ASNARO のソフトウェアは Windows コンピュータにしか対応しておらず、クラウドコンピューティングの定義における「標準的な仕組みでネットワークから利用可能性」を必ずしも満足しているとは言えない。クラウドサービスに多くある特徴として「従量課金モデル」があげられるが、この研究において用いられた ASNARO はソフトウェアの「売り切りモデル」であり、クラウドサービスらしさに欠けるところがあり、普及に至るに難しい点が残る。また、この研究では、プリプロセッサがサービス化されていなかったことから、機能的に不十分なところも残り、クラウドコンピューティングのメリットを十分に活かすことができなかった。

3.7 結言

本章では，クラウドコンピューティングの定義について整理し，その内容について説明した．同時に CAE 分野に広がるクラウドサービス利用を調査し，その問題点について指摘した．

第4章 連携するソフトウェア, ライブラリ, API

4.1 FrontISTR

FrontISTR[9] は東京大学生産技術研究所革新的シミュレーション研究センターが開発した有限要素法構造解析ソルバプログラムである。

解析機能として, 弾性静解析・動解析, 材料・幾何学的非線形静解析・動解析, 接触静解析・動解析, モーダル応答解析を含む固有値解析及び熱伝導解析を行うことができる。MPI を用いた大規模並列解析に対応している。

利用できる要素として, ソリッド要素, 梁要素, シェル要素などが実装されている。ソリッド要素は, 2 節点トラス要素, 4 節点四面体要素, 10 節点四面体二次要素, 6 節点五面体要素, 15 節点五面体二次要素, 8 節点六面体要素, 20 節点六面体要素が実装されている。シェル要素, 3 節点三次元一次要素, 6 節点三次元二次要素, 4 節点三次元一次要素, 9 節点三次元二次要素が実装されている。

4.2 REVOCAP_Prepost

REVOCAP_Prepost[26] は, 東京大学生産技術研究所革新的シミュレーション研究センターが開発したプリプロセッサ, ポストプロセッサシステムである。

ソルバプログラムである FrontISTR で解析するための, メッシュの作成, 解析条件の設定をグラフィカルインターフェース上で行うことができる。

本研究では, REVOCAP_Prepost の内部に利用されているライブラリである Revocap_Mesh を利用してメッシュファイルの操作を行っている。

4.2.1 REVOCAP_Mesh

REVOCAP_Prepost で用いるメッシュ操作部をライブラリ化したものであり, 内部的に幾何処理用ライブラリ, 行列処理用ライブラリ, メッシュ処理用ライブラリ, メッシュ表示用ライブラリ, メッシュ生成用ライブラリ, メッシュ入出力用ライブラリ, CAD 形状処理用ライブラリから構成 4.1 されている。

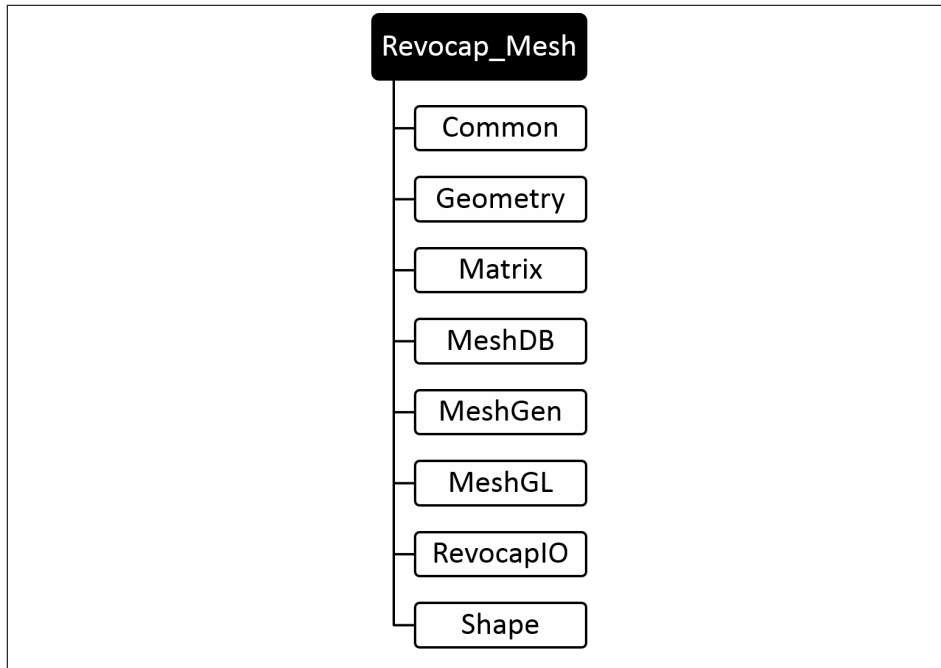


Fig 4.1: Revocap mesh library

4.3 ADVENTURE_Tetmesh

ADVENTURE_Tetmesh[22] は、東京大学生産技術研究所革新的シミュレーション研究センターが開発した四面体メッシュャである。三角形表面パッチデータから四面体メッシュを自動生成する。節点の密度制御などが可能で、二次要素への変換も可能である。本研究では、ADVENTURE_Tetmesh をメッシュャとして利用している。

4.4 OpenGL

OpenGL[28] は、シリコングラフィックスによって開発されたグラフィックスハードウェアのアプリケーションプログラミングインタフェースである。本研究においては、REVO-CAP_Mesh において読み込んだメッシュの表示や回転など 3D モデルの表示に利用している。

4.5 libpng

libpng[29] は、画像データの形式である PNG のエンコード、デコードを行うライブラリである。本研究においては、生成画像の変換に利用している。

第5章 三次元モデルの二次元画像による操作

5.1 緒言

三次元モデルを表示、操作するためにはモデルの情報を読み込み、描画を行うが、モデルサイズの増加に伴い計算処理の負荷が大きくなる。また、Web上での三次元モデル操作は実装においてオーバヘッドが大きく難がある。

デスクトップコンピュータでプリポストを行うとき、モデルサイズが大きくなってくると、読み込みに数分、モデルの操作がスムーズでなくなるの弊害が発生し、対話性に難が出てくることや、メモリ不足による強制終了のリスクが高まる。

そこで、モデルサイズに影響されずに三次元モデルを表示や操作するために、Fig 5.1のように複数枚の画像を事前にサーバで生成し、ユーザ操作に応じてサーバから転送する方法を用いた。この手法では、事前にサーバ側で画像を生成する負荷はあるものの、性能の良いGPUを採用することである程度カバーし、一千万節点規模になっても問題なくできることを確認している。

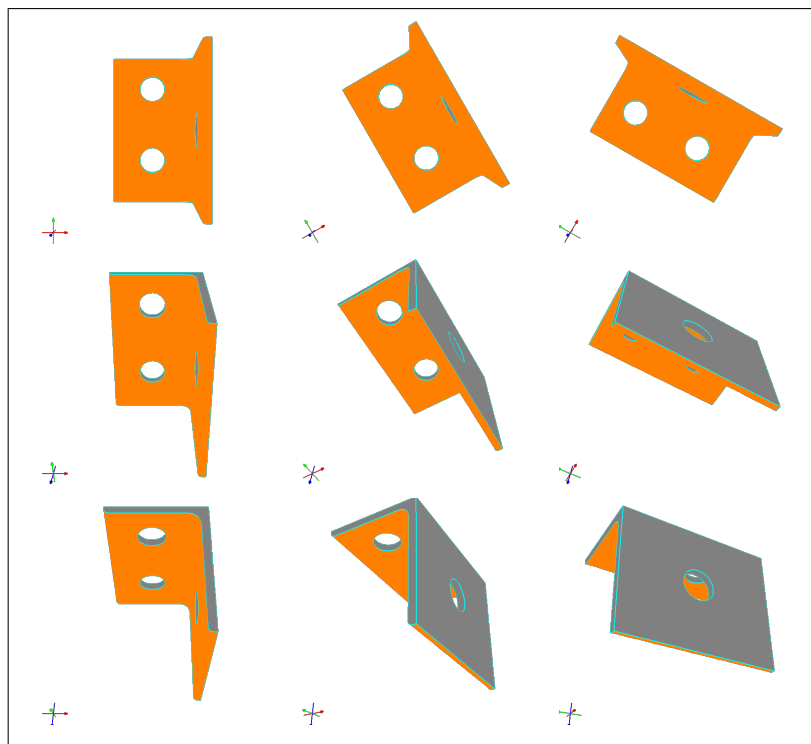


Fig 5.1: Handling 3D model by 2D image

5.2 回転

はじめに、モデルの xyz 軸それぞれを θ_1 度ごとに回転させた画像を生成する方法があげられる。モデル全体を見渡した時、見えない部分や見えにくい部分があるが、単純で生成枚数も少ない方法である (Fig 5.2)。 $\theta_1 = 30$ とすると、ひとつのモデルの可視化に $3 \times (360/30) = 45$ 枚の画像が生成されることになる (Fig 5.3)。

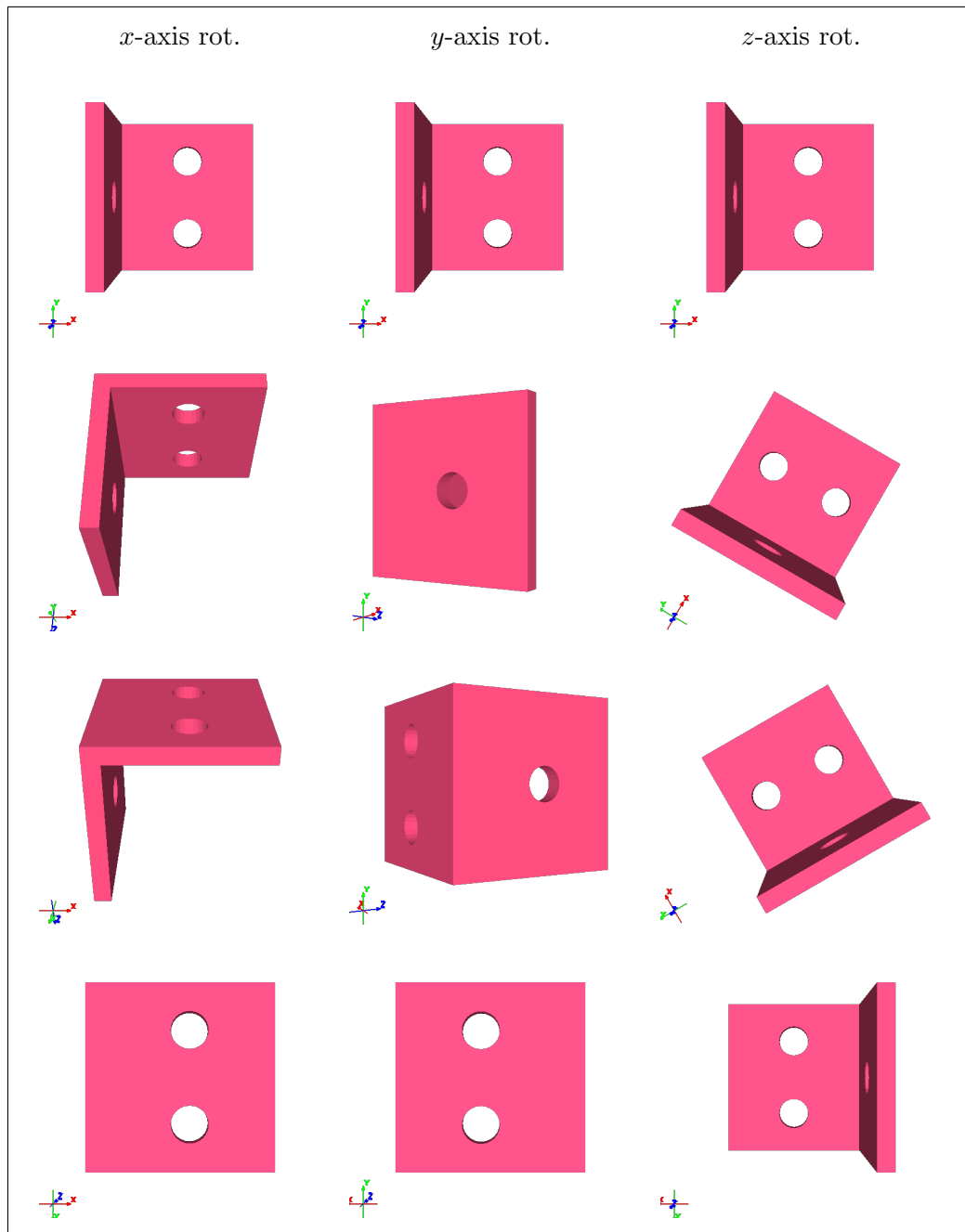


Fig 5.2: Rotation of x,y,z -axis

次に、モデルの z 軸を θ_1 度ごとに回転させた画像を生成し、その投影面の y' 軸でさらに θ_2 度ごとに回転させた画像を生成する方法である。 $\theta_1 = 30, \theta_2 = 30$ とすると、ひとつのモデルの可視化に $(360/30) \times (360/30) = 144$ 枚の画像が生成されることになる (Fig 5.3).

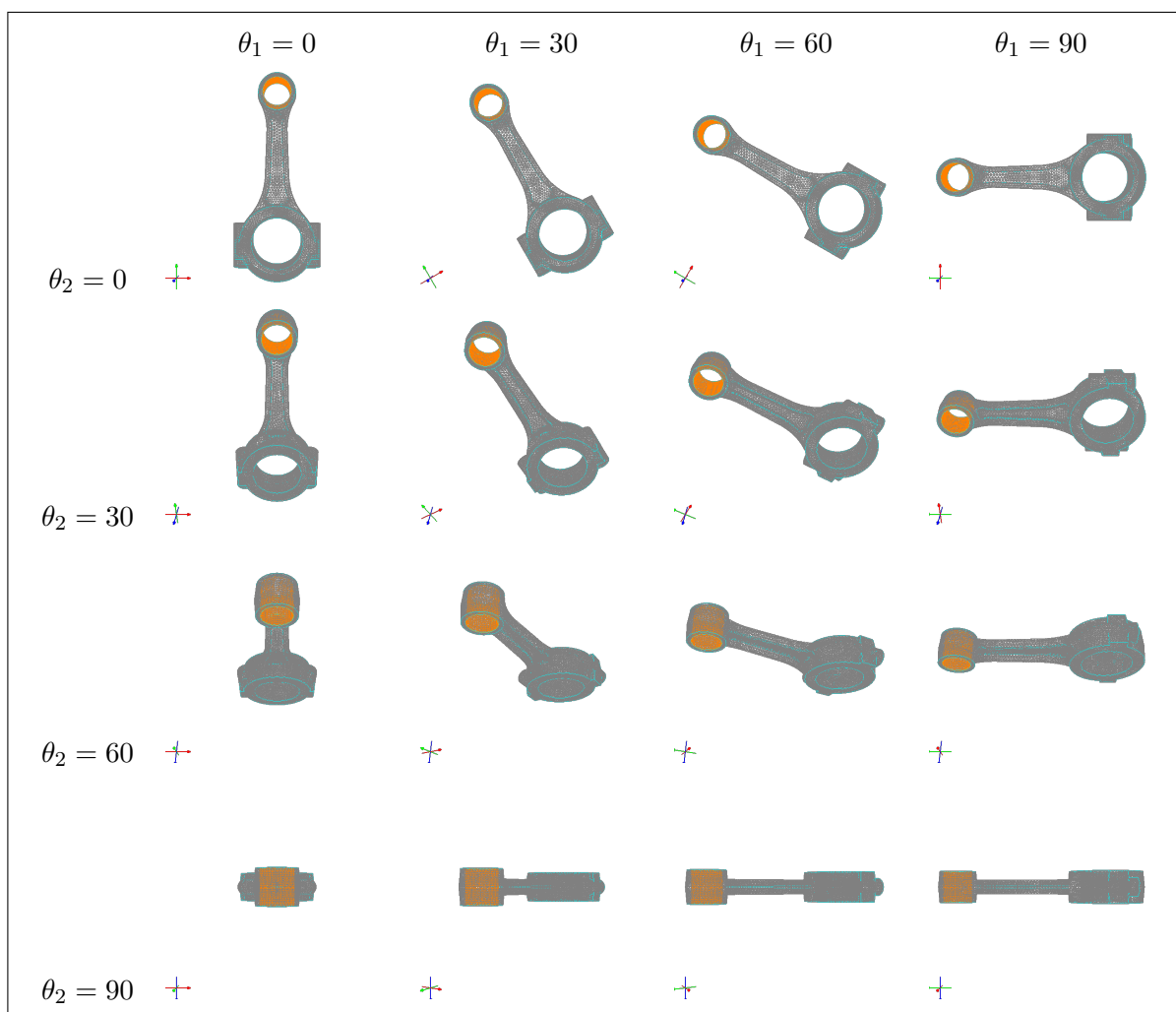


Fig 5.3: Rotation of θ_1, θ_2

生成した画像のファイル名に θ_1, θ_2 の生成順番号 T_1, T_2 を入れておくことで、ブラウザから表示している画像の回転後のファイル名がわかるので、ユーザのボタン操作で画像の URL を差し替えて画像を入れ替える。この機能は Ajax で実現し、具体的には Javascript を用いている。

5.3 拡大縮小と平行移動

予め生成しておく画像は、表示させる描画領域よりも解像度の高いものを用意しておく。表示させた画像は、描画領域に対して縮小画像が表示されていることになるため、この画像の拡大率を操作することでモデルの拡大縮小を表現できる。また、拡大した画像の表示部分をずらすことで、上下左右に移動できる。このようにして平行移動を表現する。この機能は Ajax で実現し、具体的には Javascript を用いて HTML の imgbox を操作して行っている。

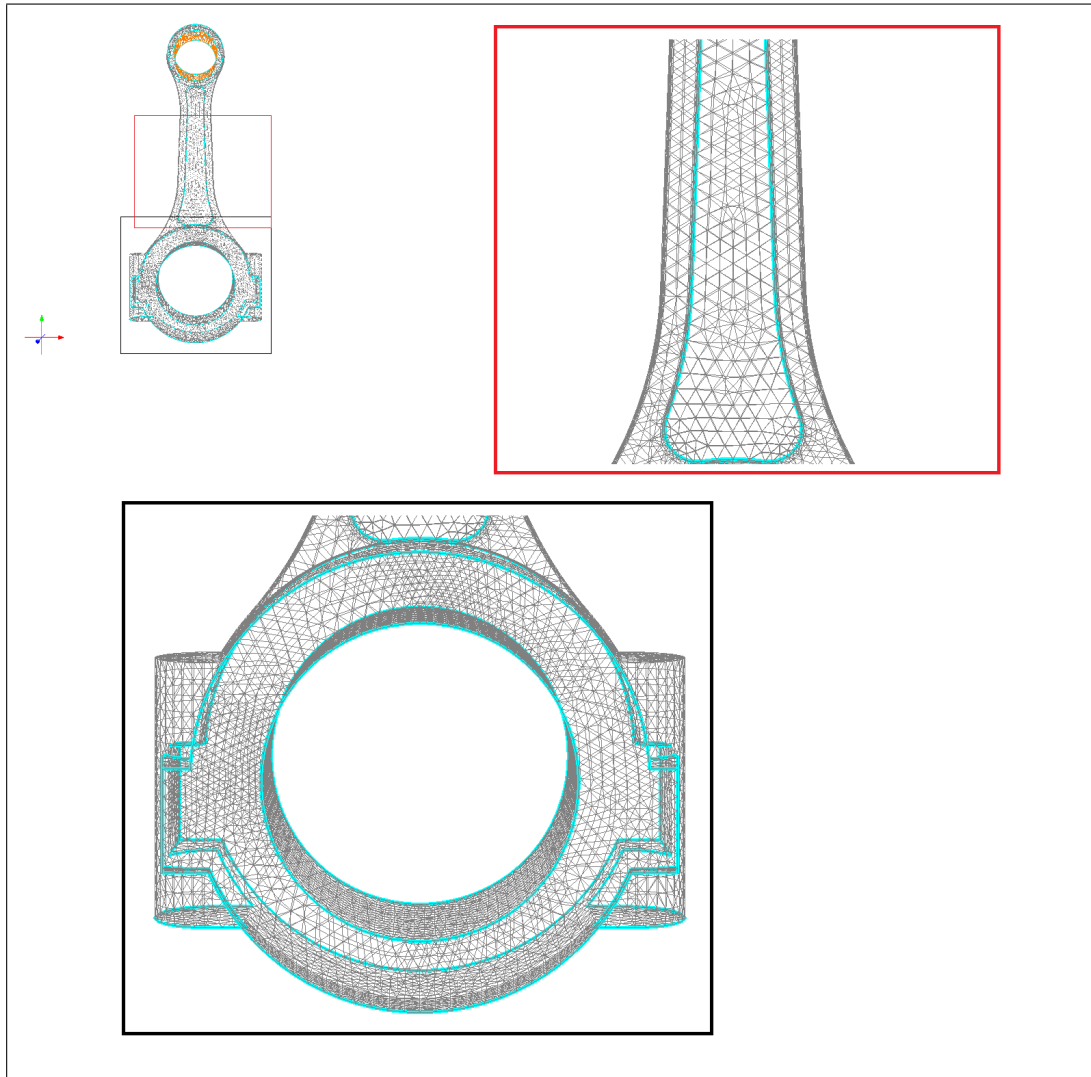


Fig 5.4: Zoom of model by image

5.4 面の表現

REVOCAP_Meshにおいて、面分割閾値により表面を内部的に分割しておく。OpenGLにて表面を描画する際に、面ごとに色を付け (Fig 5.5), それぞれの面にのみ色のついたモデル画像 (Fig 5.6) を用意する。面の切り替えは読み込む画像のファイル名の一部の入れ替えで表現できる。

```
for(j=0; i<=FACE_MAX; i++){
  if (j == TARGET_FACE){
    glColor3f(1.0,0.5,0.0);
  }else{
    glColor3f(0.5,0.5,0.5);
  }
  DRAWFACEFUNC(j);
}
```

Fig 5.5: How to set color to selected face

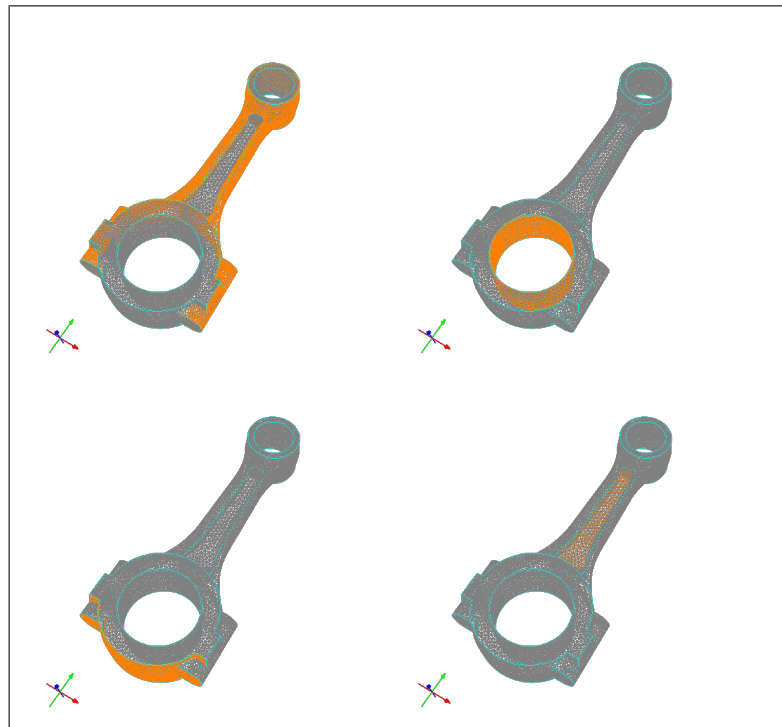


Fig 5.6: Representation of surface method

5.5 ファイル名による画像の特定

ファイル名を"FACE_NAME.T₁.T₂.png"のようにしておくことで、ファイル名だけで画像を特定できることになる。

したがって読み込む画像のファイル名を切り替えることで、回転、面の表現を切り替えることができる。簡単のため、ここでは面の切替とモデル回転を2変数で表すコード例を示す(Fig 5.7)。

```
<SCRIPT type="text/javascript">
<!--
  var face = "DEFAULT"; //set default value
  var rot  = "00";      //set default value
  function ChangeImg(A){
    rot = A;
    document.window.src = "./" + face + "-" + rot + ".png";
  }
  function ChangeFace(A){
    face = A;
    document.window.src = "./" + face + "-" + rot + ".png";
  }

  //-->
</SCRIPT>
<IMG src="" name="window" width=480>
<INPUT TYPE="button" onClick="ChangeImg('00')" VALUE='ChangeImage0'>
<INPUT TYPE="button" onClick="ChangeImg('01')" VALUE='ChangeImage1'>
<INPUT TYPE="button" onClick="ChangeFace('FN0')" VALUE='ChangeFace0'>
<INPUT TYPE="button" onClick="ChangeFace('FN1')" VALUE='ChangeFace1'>
```

Fig 5.7: Javascript for swapping image and face

5.6 結言

本章では、二次元画像のみを用いて三次元モデルの操作をできる手法を述べた。この手法を用いることで三次元モデルを直接ハンドリングすることなく二次元画像をベースに操作ができる。

第6章 開発したプログラムとライブラリ

6.1 緒言

本研究において、4章で挙げるソフトウェア間の連携を図るため REVOCAP_prepost を基にした新しいプリプロセッサを作成するための基盤として、ライブラリを開発した。さらに、自動化されていなかった処理を自動化するプログラム、処理の簡略化するためのプログラムを開発した。最後に、これらのプログラムの実行を制御するバッチシステムを作成した。本章では、このプログラムの動作とその実装について述べる。

6.2 可視化系ライブラリ

5章の手法により画像群を作成するプログラムをライブラリ化する。このプログラムでは、可視化対象のデータを読み込み、画像によって三次元モデルを表現する。モデルに関するデータの保持や操作については、REVOCAP_Mesh ライブラリを利用しているので、REVOCAP_Mesh の上層で動くライブラリという位置づけである。以下にその取り扱いについて述べる。

6.2.1 データの取り扱い

メッシュ処理用ライブラリ (MeshDB) を用いて、モデルデータを保持するインスタンスを生成する (Fig 6.1)。

REVOCAP_Mesh では形状データ、メッシュデータ、解析結果ファイルについて、大きな違いなく同様の手続きで読み込むことや操作することができる。従って、特にデータ種別で分けずに述べることとし、ここで生成したインスタンスについてはメッシュインスタンスと呼ぶこととする。

6.2.2 モデルの読み込み

まず、データの入出力を開始する前に、メッシュインスタンスのメンバ関数である begin-Model を実行する (Fig 6.1)。

その後、メッシュ入出力用ライブラリ (RevocapIO) を利用してデータの読み込みを行う。例えば、FrontISTR の msh ファイルであれば、RevocapIO ライブラリの HecmwIO クラス

```

kmb::MeshDB *mesh= new kmb::MeshDB;
mesh->beginModel();

//Alter IO module by file type
kmb::HecmwIO io;
io.loadFromFile(filename, mesh);

mesh->endModel();

```

Fig 6.1: Creating instance for handling model data and loading from file

を利用し、そのメンバ関数である loadFromFile を実行することで、メッシュインスタンスにデータを読み込むことができる。FrontISTR の res ファイルであれば、メンバ関数である loadFromResFile を実行することで、メッシュインスタンスにデータを読み込むことができる。

同様に、ADVENTURE_Tetmesh の msh ファイルであれば、RevocapIO ライブラリの TetMeshMIO クラスを利用し、そのメンバ関数である loadFromFile を実行することで、メッシュインスタンスにデータを読み込むことができる。

MicroAVS の UCD ファイルであれば、RevocapIO ライブラリの MicroAVSIO クラスを利用し、そのメンバ関数である loadFromFile を実行することで、メッシュインスタンスにデータを読み込むことができる。

メッシュファイルに限らずに同様の処理でデータを読み込むことができ、STL ファイルであれば、RevocapIO ライブラリの STLIO クラスのメンバ関数 loadFromFile を実行する。

ただし、IGES や STEP といった CAD ファイルの場合は、表面パッチを作成する処理が追加される。まず、CAD 形状処理用ライブラリ (Shape) を用いて、CAD データを保持するインスタンスを生成する。その後、Shape ライブラリの CADFileIO クラスを利用し、そのメンバ関数である readIGES や readSTEP でシェイプインスタンスに読み込む。続いて、Shape ライブラリの PatchGenerator クラスを利用し、そのメンバ関数 setIncremental などを実行し、パラメータを設定した後に、execute を実行し表面パッチをメッシュインスタンスに読み込む。

最後にメッシュインスタンスのメンバ関数である endModel を実行することでモデルの読み込みが完了する。

6.2.3 データの変換と保存

読み込みと同様に，メッシュ入出力用ライブラリ (RevocapIO) を利用することでデータの変換や保存ができる (Fig 6.2). FrontISTR の msh ファイルで保存する場合であれば，RevocapIO ライブラリの HecmwIO クラスを利用し，そのメンバ関数である saveToFile を実行することで，メッシュインスタンスのデータを保存することができる。

```
kmb::MeshDB *mesh= new kmb::MeshDB;
mesh->beginModel();

// Alter IO module by file type
kmb::TetMeshMIO loadio;
loadio.loadFromFile(LOADFILENAME, mesh);

mesh->endModel();

// Model Operation
...

// Alter IO module by file type
kmb::HecmwIO saveio;
saveio.saveToFile(SAVEFILENAME,mesh)
```

Fig 6.2: Converting file type

6.2.4 境界抽出と表面分割

プリプロセッシングを進める上メッシュ表面を抽出し，表面分割を行う必要がある。

メッシュインスタンスの getBodyCount 関数を呼び，領域数を取得する。この領域について，1つずつ以下にあげるそれぞれの処理を行う。

getBodyName 関数を呼び名称が設定されているか確認し，設定されていない場合は，setBodyName 関数で何らかの名前を設定しておく。

updateBoundingBox 関数を呼び，BoundingBox を更新する。BoundingBox とは，モデルの表示領域のことである。

境界抽出は，メッシュ処理用ライブラリ (MeshDB) の BoundaryExtractor クラスを利用し，setMesh 関数でメッシュインスタンスを指定し，appendBody 関数で領域を指定し，getBodyBoundaryFace 関数でその領域の境界を抽出する。その後 clear 関数を実行し境界条件を終

了する.

表面分割は, SurfaceOperation クラスを利用し, setMesh 関数でメッシュインスタンスを指定し, divideFaceGroupWithRidge 関数で角度を指定し, その領域表面での表面分割を完了する. ここでは, 表面分割数が戻り値として戻る.

以上の処理を領域ごとに行うことで, 境界抽出と表面分割を行うことが出来る (Fig 6.3).

```

kmb::MeshDB *mesh= new kmb::MeshDB;
std::map <std::string,kmb::bodyIdType>  faceIds;

// Model Load
...

int body = mesh->getBodyCount();
for (int i=0;i<body;i++){
    int dim = mesh->getDimension(i);
    char* bodyname = (char*)mesh->getBodyName(i);
    if (strcmp(bodyname,"")==0){
        sprintf(bodyname,"Body%llu",i);
        mesh->setBodyName(i,bodyname);
    }

    //Update Bounding Box
    mesh->updateBoundingBox(i);
    sprintf(wholeFaceName,"%s_",bodyname);

    //Boundary extraction
    kmb::BoundaryExtractor bc;
    bc.setMesh(mesh);
    bc.appendBody(i);
    kmb::bodyIdType BI;
    BI = bc.getBoundaryFace(i,wholeFaceName);
    bc.clear();

    //SurfaceOperation
    kmb::SurfaceOperation div;
    div.setMesh(mesh);
    int faces;
    faces = div.divideFaceGroupWithRidge(wholeFaceName,DEGREE,faceIds);
}

//Continue
.....

```

Fig 6.3: Boundary Extraction and surface operation

6.2.5 モデルの描画

まず、OpenGL ライブラリの `glClear` 関数を最初に実行し、カラーバッファ、デプスバッファ、アキュムレーションバッファ、ステンシルバッファをクリアする。

次に、投影行列の設定を行う。メッシュインスタンスの `getBoundingBox` 関数で `BoundingBox` 構造体を取得し、そのメンバの `Diameter` を取得する。これが投影されるべき描画領域である。 `glMatrixMode(GL_PROJECTION)` 命令を実行し投影行列に切り替える。 `glLoadIdentity()` 命令を実行し投影行列を単位行列にする。透視法射影の場合は、 `glFrustum` 関数を実行し、正射影の場合は、 `glOrtho` を実行する。

続いて、モデルビュー行列の設定を行う。 `gluLookAt` 関数で視点を設定する。回転を `glRotatef` 関数で指定する。メッシュインスタンスの `getBoundingBox` 関数で `BoundingBox` 構造体を取得し、そのメンバの `centerX`, `centerY`, `centerZ` を取得し、 `glTranslatef` でモデル中心を原点に移動させる。メッシュモデルの表示のときは、 `glDisable(GL_LIGHTING)` 命令で、光源の使用をやめる。形状モデルの表示のときは、 `glEnable(GL_LIGHTING)` 命令で、光源の使用を行う。 `glPolygonMode` 関数で、ワイヤースタイル表示、サーフェス表示の切り替えを行う。 `glPolygonMode(GL_FRONT_AND_BACK, GL_LINE)` 命令では、表面だけでなく裏面も描画することを表し、 `GL_LINE` でワイヤースタイル (Fig 6.5), `GL_FILL` でサーフェス (Fig. 6.6) 表示となる。

ここまでで、モデル描画のための行列を設定した (Fig 6.4)。

ここから、表面についてイテレータを回し、 `glColor3f` 関数で色を指定し、メッシュ表示用ライブラリ (MeshGL) の `drawFaceGroup` 関数で表面を描画する。

モデル回転は `glRotatef` 関数の回転角を変えて何度もこの処理を繰り返すことで作成している。

```

// Clear buffer
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT
        | GL_ACCUM_BUFFER_BIT | GL_STENCIL_BUFFER_BIT);

// Initialize matrix
glMatrixMode(GL_TEXTURE);
glLoadIdentity();
glMatrixMode(GL_PROJECTION);
glLoadIdentity();

//Orthographic projection
glOrtho(-diameter*0.5,+diameter*0.5,-diameter*0.5,
        +diameter*0.5, diameter, diameter*3);

//Model View Matrix
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(0, 0, 0+diameter*2, 0, 0, 0, 0.0, 1.0, 0.0);
glPushMatrix();

//Rotation
glRotatef( theta_alpha, 1.0,0.0,0.0);
glRotatef( theta_beta, 0.0,0.0,1.0);

//Centering
glTranslatef(-center.x,-center.y,-center.z);

//Disable Lightning and Enable Z-buffer
glDisable(GL_LIGHTING);
glEnable(GL_DEPTH_TEST);

//Wireframe
glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);

//Draw Model
...

```

Fig 6.4: Preparing to Model projection

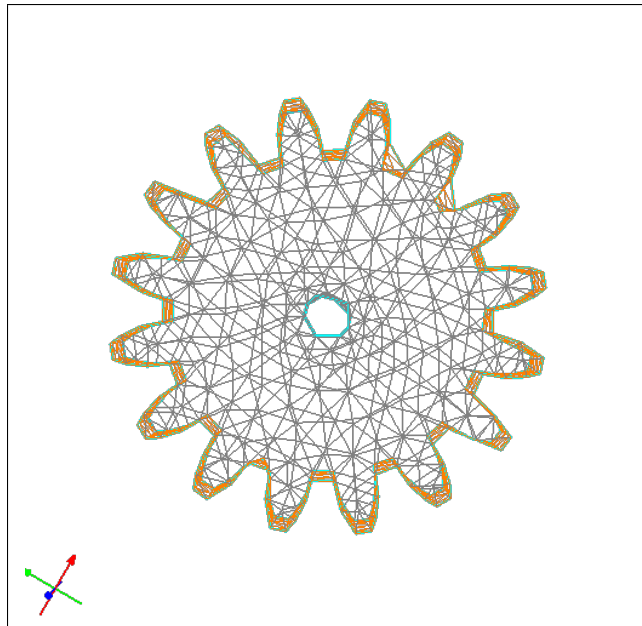


Fig 6.5: Wireframe

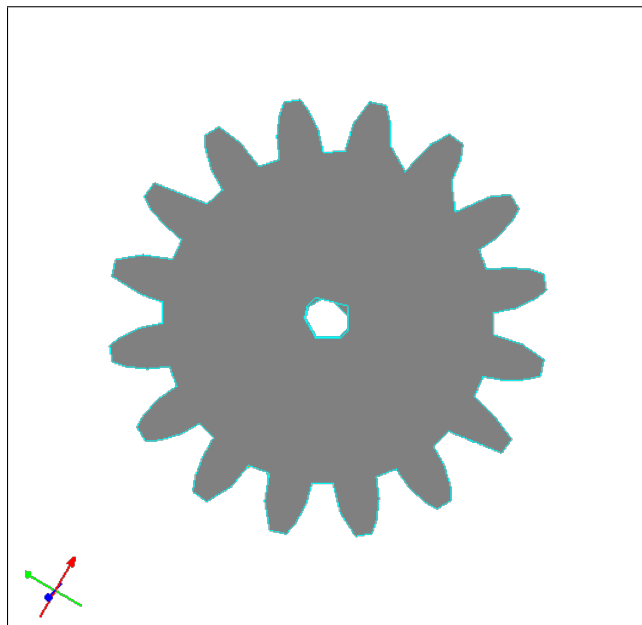


Fig 6.6: Surface

6.2.6 画像の生成

前章で描画した画像はフレームバッファから、OpenGL の `glReadPixels` 命令 (Fig 6.8) を利用し、メインメモリにコピーする。OpenGL の処理がアイドルになった時に実行される関数を格納する `glutIdleFunc` から呼び出すことで、順次画像が回転され、複数枚の画像が作成される。画像とその回転角情報は、構造体で保持し、STL(Standard Template Library) の両端キューに、末尾に要素を追加していくことで、保存する。メモリ使用量があるしきい値一定以上となると、両端キューの先頭から要素を取り出し、画像を変換してファイルに書き出す。

```
void Capture(GLubyte* pptr, int WIDTH, int HEIGHT){  
  
    glReadBuffer(GL_BACK);  
    glPixelStorei(GL_PACK_ALIGNMENT , 4);  
    glReadPixels(0, 0, WIDTH , HEIGHT, GL_RGBA, GL_UNSIGNED_BYTE, pptr);  
  
}
```

Fig 6.7: How to creating image

```

void idle(void){

    Image_list p; //Structure for created image list.
    ....
    p.img = (GLubyte*)malloc((WIDTH)*(HEIGHT)*4*(sizeof(GLubyte)));
    capture(pptr, WIDTH, HEIGHT);

    // change parameter.
    ....

    glutPostRedisplay();

}
void display(){

    //OpenGL code
    ...

    glutSwapBuffers();

}

int main(int argc, char* argv[]){

    ...
    glutDisplayFunc( display );
    glutIdleFunc( idle );

    glutMainLoop();
}

```

Fig 6.8: How to creating image

6.2.7 描画処理の高速化

本ライブラリは、多くの OpenGL 命令を実行する。Displaylist は、glNewList～glEndList 関数で間に挟まれた一連の OpenGL 命令をコンパイルしメモリ上に配置し、glCallList 関数でそのコンパイル済みの OpenGL 命令を呼び出す機能である (Fig 6.9)。

特に OpenGL の処理が集中する関数は、MeshGL ライブラリの drawFaceGroup 関数といった実際にモデルを描画する命令であり、事前に DisplayList を作成しておくことで、Mesh ライブラリ側での同じ命令を省略することができ、MeshGL のインスタンスのメンバ関数を直接実行するよりも高速化が可能である。モデルを回転させ同じ対象物の描画処理は、glCallList に置き換えて呼び出すことで処理の軽量化に成功した。これにより、OpenGL によるモデル描画の処理時間は無視できるレベルにまで高速化された。

```

GLuint DL;
void idle(void){

    glutPostRedisplay();

}
void display(){

    glRotatef( ... );
    glTranslatef( ... );
    glCallList(DL);
    glutSwapBuffers();

}

int main(int argc, char* argv[])

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGBA | GLUT_DEPTH);
    glutInitWindowSize( ... );
    glutCreateWindow( ... );
    glutDisplayFunc( display );
    glutIdleFunc( idle );

    DL = glGenLists(1);
    glNewList(DL, GL_COMPILE);
        glutSolidTeapot(1);
    glEndList();
    glutMainLoop();

}

```

Fig 6.9: How to use displaylist

6.2.8 画像出力の並列化

メインメモリにコピーされた画像は、4バイトビットマップである。この画像を、libpngを利用してPNG(Portable Network Graphics)画像に変換する。PNG形式は透過をサポートしており、軽量であることからWeb上での画像表示に適している。

画像出力は、メモリコピーした画像を保存している構造体を両端キューについて順番にPNGへの変換命令実行しファイルに書き出す処理から構成される (Fig 6.10)。

表面分割や境界抽出などのREVOCAP_Mesh側の処理を除けば、この画像出力の処理が最も時間のかかる処理である。従って、この画像の出力はOpenMPを用いて並列化した。

メモリ上の画像を変換する処理だけであり、依存関係もないので並列効率は非常に高い。

```
void image_flush( std::deque<Image_list> &image ){  
  
    #pragma omp parallel  
    {  
        #pragma omp for schedule(guided)  
        for (int i=0; i < image.size(); i++) {  
            Image_list *p = &image[i];  
            WritePNG( ... );  
            free(p->img);  
        }  
    }  
    image.clear();  
  
}
```

Fig 6.10: Converting captured image on memory to image file.

6.3 可視化系ソフトウェア

6.2で作成したライブラリを用いて以下のソフトウェアを実装した。ファイルの読み込みや境界抽出などはライブラリに任せたが、実際にプログラムとしてシステムから実行されるものを4つに分けて用意した (Fig 6.11)。

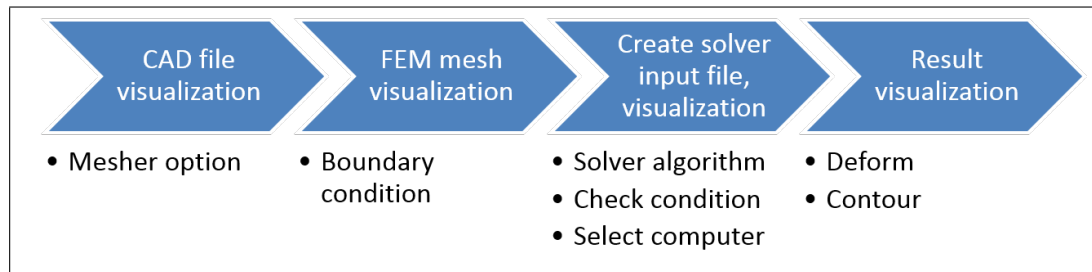


Fig 6.11: Visualizing software

6.3.1 形状データ可視化

形状データ可視化プログラムは、CADモデルなどの形状の内容を図示して確認するために使用する。CADモデルを読み込み、三次元モデルを表現できる画像の組み合わせで表現する。このプログラムは形状を読み込むだけであるので、負荷は比較的軽い。

6.3.2 FEMメッシュデータ可視化

FEMメッシュを図示して確認するプログラムである。6.4章により作成された四面体要素のみならず、六面体要素でも利用可能であり、FrontISTRでサポートする多くの要素に対応している。最も重要な可視化プログラムであり、可視化系としての画像生成枚数は最も多くなる。REVOCAP_Meshライブラリで、境界抽出と表面分割処理が含まれ、並列化されていないため時間がかかる。

ここでは、表面分割したそれぞれの面に色付けしたものを用意し、その画像を組み合わせで3Dモデルを表現している。さらにワイヤーフレームとサーフェスの切り替えのためにどちらの画像も生成している。他の可視化系と比較すると、負荷は比較的重い。ただし、このプログラムの中で一番時間がかかる処理はPNG画像に圧縮する処理でかつ並列化できていることから、サーバスペック強化で対処できる。

6.3.3 FrontISTR 入力ファイル生成・可視化

ユーザの操作により境界条件が設定され、FrontISTR[9]で読み込むことのできる形式に変換される。しきい角により表面を分割する処理が含まれ、並列化できていないため可視化

系としては処理が多いプログラムの1つである。境界条件設定スクリプトをシステムで解釈し、それにしたがって解析条件を設定する。この境界条件設定スクリプトは、「面の名前, 境界条件の種類, 境界条件設定名, 値1, 値2, …」形式のカンマ区切りテキスト形式 (Fig 6.12) であり, メッシュをスクリプトとともに保存することで解析モデル用のファイルの重複を減らすことができる。

```
Steel,Material,ALL,7860,0.29,2.06e+11  
BND0,BOUNDARY,Body0_0,0,0,0  
CLO,CLOAD,Body0_0,0,0,1  
DLO,DLOAD,Body0_0,1
```

Fig 6.12: Condition script

6.3.4 数値計算結果可視化

ソルバの計算結果を利用して, ユーザが理解できる図を作成するプログラムである。

境界抽出などの必要がなく, 処理自体は比較的軽い。変形強調図, 変位コンター図, ミーゼス応力コンター図などを自動的に作成し, 画像の組み合わせとして出力する。

6.4 自動四面体メッシュ連携

CAD モデルを自動四面体メッシュにより三次元四面体要素に変換する。まず CAD モデルを 6.3 章で作成した可視化系ソフトウェアで形式の変換を行い Adventure.tetmesh[22] で読み込める形式に変換する。その後, 節点密度制御などの設定したオプションとともに Adventure.tetmesh を実行するジョブスクリプトを作成する (Fig 6.13)。

```

#!/bin/sh
cd WORKDIR
php jobstatus_change.php $1 Started
php jobstatus_change.php $1 'Smoothing the surface patches'
advtmesh9p FILENAME
php jobstatus_change.php $1 'Generating the tetrahedral mesh'
advtmesh9m FILENAMEc

# if Generate quadratic tetrahedral mesh
php jobstatus_change.php $1 'Generating quadratic tetrahedral mesh'
advtmesh9s FILENAMEc

if [ -e FILENAMEcs.msh ]; then
php jobstatus_change.php $1 'Finished' 'Success'
php mesh_regist.php include_mesher_FILENAME.php
else
php jobstatus_change.php $1 'Finished' 'Failed'
fi

```

Fig 6.13: Execution mesher script

6.5 ジョブ管理

本研究に用いるプログラムの実行はジョブ管理プログラム (Fig 6.14) によって管理される。cronD で毎分呼び出す形では、SQL データベースを参照し、待ちキューに入っているジョブがないかを確認し、そのキューがある場合は、LoadAverage を確認して実行するような仕組みである。同時にリモートジョブを送信した場合、ポーリングして結果を確認する役割も同時に持つ。

6.6 結言

本章で作成したライブラリとソフトウェアを利用してプリポストを含めたクラウドサービスに用いていくことにする。

```

<?php

//db module
require_once("db.php");
//get loadaverage
$la = sys_getloadavg();
//get system cpu number
$core = exec("getconf _NPROCESSORS_ONLN");

if ($la[0]>$core*2){
exit;
}

//load from db
$stmt = $mysqli->prepare("SELECT id.... FROM job
    where status = 'Pending' order by id asc");
$stmt->execute();
$stmt->bind_result($id,$script.....);
while ($stmt->fetch()) {

    //change status
    $sttus = "Running";
    $stmt2 = $mysqli2->prepare("UPDATE job SET phase = ? WHERE id = ?");
    $stmt2->bind_param('si', $sttus, $id );
    $stmt2->execute();
    $stmt2->close();
    //Run script
    shell_exec("$script $id");

}
$stmt->close();
?>

```

Fig 6.14: Job management script

第7章 クラウドCAEシステムの構築

7.1 緒言

本研究では、WebブラウザだけでCAE全体を行うことのできるクラウドサービスの作成を目的としている。この目的を達成するため、ハードウェアを用意して、4章で挙げたソフトウェアと6章で作成したソフトウェア・ライブラリを組み合わせることでシステムを構成した。

7.2 サーバの準備

Webサービス化するために、まず作成したソフトウェアを実行するコンピュータを用意した (Table 7.1)。

このコンピュータにオペレーションシステムとして Ubuntu 14.04.1 LTS をインストールを行った。さらに、HTTPサーバデーモンである apache-2.4.7、データベースサーバである mysql-5.5、スクリプト言語の php-5.5.9 をインストールした。

Table 7.1: Spec of CAE cloud server

CPU	Intel(R) Celeron(R) CPU G1820 @ 2.70GHz
GPU	Radeon R9 290 (Hawaii PRO)
Memory	DDR3 SDRAM PC3-17000 2GBx4
Hard Disk	HP 250GB SATA disk VB0250E AVER

7.3 Webサービススクリプト

Webサービス化するために、作成したライブラリソフトウェアをHTTP上から呼び出せるphpのスクリプトプログラムを作成した。外観を構成する部分と内部的にプログラムを実行させる部分から構成されている。phpのincludeを活用することで、ヘッダーとフッターを共通ファイルとした (Fig 7.1)。cssを用いてデザイン性も高めている。

描画処理が必要なHTTPリクエストが到着した時点で、6章で作成した描画プログラムを内部的にバックグラウンドで実行させる。

この描画プログラムは、扱うファイルのファイルサイズ・節点数などデータベースに登録されている情報から実行時間を簡易的に予測し、閾値を超える場合はバッチジョブ機能で実行し、超えない場合はリアルタイム処理で実行しプリプロセッシングに用いる画像を生成させる。この閾値は経験的に設定を行い、調整できるようにしてある。

```
<?php include "./common/header.php" ?>
Content
<?php include "./common/footer.php" ?>
```

Fig 7.1: php include

7.4 ローカルバッチジョブシステム

負荷の大きなプロセスによって、本システムの負荷が異常となり、Webサービスの提供が不能とならないことを目的として、数秒で終わらないと見積もられるプロセスはバッチジョブ機能により実行する。この機能は、`cron`を用いて20秒ごとにphpスクリプトを呼ぶことで実現されている (Fig 7.2)。`cron`では1分より細かい設定ができないため、`sleep`コマンド (`sleep 20;./run;`のように)を組み合わせ毎分20秒、40秒での実行を実現した。

バッチジョブ機能を構成するphpスクリプトは実行されるとデータベースにアクセスし、実行待ちジョブの有無を確認し、実行待ちジョブがある場合はLoad averageとCPUコア数から実行できる余裕があるかを確認して実際に実行する機能を持つ。

システムデーモンのためにプロセス数を制限でき、本システムではシステム向けに1コア分を保留する設定としたので、2コアしかない本システムでは並列にジョブの実行は行わないことになる。

```
* * * * * EXECUSER php run_job.php
* * * * * EXECUSER sleep 20;php run_job.php
* * * * * EXECUSER sleep 40;php run_job.php
```

Fig 7.2: Cron for Job manage script

7.5 外部計算機

東京大学情報基盤センターのFX10 スーパーコンピュータシステム (Oakleaf-FX) [30], 東京大学大学院新領域創成科学研究科人間環境学専攻奥田研究室の TC クラスタ [31], Amazon EC2[4] を外部計算機として利用した (Table 7.2).

Table 7.2: Spec of calculation server

	GFLOPS per node	Node
EC2 (C3.large)	22.4	1
EC2 (C3.8xlarge)	358.4	1
TC-Cluster	85.1	12
Oakleaf-FX(FX10)	236.5	4800

7.6 リモートジョブ送信機能

外部計算機へファイルを転送し, バッチジョブを作成する仕組みを用意した (Fig 7.3).

この機能では, アーキテクチャの違いの吸収, 計算ジョブの送信の仕方などを, 外部計算機で「リモートジョブ作成スクリプト」 (Fig 7.4, Fig 7.5, Fig 7.6, Fig 7.7) を実行する処理と置き換え, クラウド CAE サーバから見て, ブラックボックス化する.

ここで, 定義した「リモートジョブ作成スクリプト」の入出力形式に従うように「リモートジョブ作成スクリプト」を作成することで, 本研究で触れなかった計算機においても同様に利用できる. この仕組みの前提は, SSH によってシェル実行する計算機であること, SCP によりファイルが転送できることであり, 一般的にスーパーコンピュータでも IaaS 型のクラウドサービスでも通常利用できる方法である.

ファイル転送と「リモートジョブ作成スクリプト」を実行するためのスクリプト (Fig 7.3) によって, ファイルを転送し, 「リモートジョブ作成スクリプト」を実行し, その「リモートジョブ作成スクリプト」が必要に応じてリソースグループや経過時間を設定し, 外部計算機にジョブを登録する.

ジョブシステムを使わずに, リアルタイム処理するための計算機の場合は, 「リモートジョブ作成スクリプト」形式に合わせたスクリプトでリアルタイム処理を実行する (Fig 7.7).

つまり, この「リモートジョブ作成スクリプト」によって計算機の違い, 例えば, ジョブシステムの違いや, アーキテクチャの違いを吸収する仕組みである. クラウド CAE システムからは並列数を指定してこの「リモートジョブ作成スクリプト」を実行する操作 (`~/bin/cistr_job.sh NumPE`) を行うのみであり, パーティショニングや数値計算のジョブを投入したり, ファイルを転送したりという処理は, このスクリプトを実行して待機するという処理に置き換えることができる.

以上の手続きで、外部計算機に自動的に解析ジョブが送信、または解析プログラムが実行され、その結果が出力されたかどうか定期的に cron によるポーリングが行われる (Fig 7.2).

内部計算機 (クラウド CAE サーバ) で計算する場合においても、同様のリアルタイム型の「リモートジョブ作成スクリプト」形式 (Fig 7.7) で実行でき、localhost に SSH で繋ぐことを通して外部計算機と同様に取り扱うことができる。

具体的には、このスクリプトを利用した場合 (Fig 7.4, Fig 7.5, Fig 7.6, Fig 7.7), touch によってリモート計算機に「finished」ファイルが作成されていることを確認した場合、ソルバの計算が終了したことになる。

定期的なポーリングによりリモートジョブが終了していることを確認した場合には、結果ファイルの存在を確認し、存在しない場合はエラーとして処理し、存在する場合は、ローカルに結果ファイルを SSH で転送し、そのファイルを元にポストプロセッサが実行される (Fig 7.8).

この機能はクラウド CAE システム側のローカルバッチジョブシステム経由で実行される。

```
cd WORKDIR
php jobstatus_change.php $1 'Transferring to Calc. Server'
ssh -i PRIVKEY USER@SERV 'mkdir -p ~/WORKDIR'
scp -i PRIVKEY hecmw_*.dat model.cnt mesh.msh USER@SERV:~/WORKDIR
ssh -i PRIVKEY USER@SERV 'cd ~/WORKDIR; ~/bin/cistr_job.sh NumPE'
php jobstatus_change.php $1 'Remote Job Submitted' "NULL" POLLINGSCRIPT.sh
```

Fig 7.3: Remote job handler script

7.7 データ整理・管理機能

本システムでは、システム上で作成した全てのファイルを自動的に保存しデータベースに登録する。そのファイルの作成元となったデータとの関連性についてもデータベースのユニークな番号で同時に登録される。したがって、容易にデータの関連性が利用者に示されることになり、解析結果の整理に役立つようになっている。

7.8 結言

本章ではクラウド CAE システムについて、作成したプログラムや連携するプログラムを合わせて実装を行い、システムの機能について概要を述べた。

```

#!/bin/sh

PROCS=$1
if [ $PROCS -gt 0 ] 2> /dev/null
then
echo "${PROCS}PE OK"
else
echo "Not N"
exit
fi

DAYOFWEEK='date +%w'
HOUR='expr \date +%H\' + 0'

# 16 procs per node
NODE='expr $PROCS / 16 + \( \( $PROCS % 16 \) \> 0 \)'

# set elapse time, set resource group
RSCGROUP_PARTITIONER=short
ELAPSE_PARTITIONER=06:00:00

```

Fig 7.4: Job creating script for FX10 (1)

```

if [ "$NODE" -lt 12 ]
then
RSCGROUP_SOLVER=short
ELAPSE_SOLVER=06:00:00
elif [ "$NODE" -lt 216 ]
then
RSCGROUP_SOLVER=small
ELAPSE_SOLVER=48:00:00
elif [ "$NODE" -lt 372 ]
then
RSCGROUP_SOLVER=medium
ELAPSE_SOLVER=48:00:00
elif [ "$NODE" -lt 480 ]
then
RSCGROUP_SOLVER=large
ELAPSE_SOLVER=48:00:00
elif [ "$NODE" -lt 1440 ]
then
RSCGROUP_SOLVER=x-large
ELAPSE_SOLVER=24:00:00
else
exit;
fi

#create bash script for solver
cat > ./solve.sh << EOT
#!/bin/bash
#PJM -N "fistr1"
#PJM -L "rscgrp=$RSCGROUP_SOLVER"
#PJM -L "node=$NODE"
#PJM --mpi "proc=$PROCS"
#PJM -L "elapse=$ELAPSE_SOLVER"
export OMP_NUM_THREADS=1
export PARALLEL=1
rm -f finished
mpiexec ~/bin/fistr1 2>&1 | tee fistr1.log
touch finished
EOT

```

Fig 7.5: Job creating script for FX10 (2)

```

#create bash script for partitioner

cat > ./part.sh << EOT
#!/bin/bash
#PJM -N "hecmw_part1"
#PJM -L "rscgrp=$RSCGROUP_PARTITIONER"
#PJM -L "node=1"
#PJM --mpi "proc=1"
#PJM -L "elapse=$ELAPSE_PARTITIONER"

export OMP_NUM_THREADS=1
export PARALLEL=1

sed -i -e "s/DOMAIN=[1-9][0-9]*/DOMAIN=$PROCS/g" hecmw_part_ctrl.dat
mpiexec ~/bin/hecmw_part1 2>&1 | tee part1.log

EOT

chmod +x ./part.sh
chmod +x ./solve.sh

#send job
RES='pjsub --step ./part.sh'
JID='echo $RES|awk '{print $6}' | \
sed -e "s/^\([1-9][0-9]*\)_[0-9]\+/\1/g"'
pjsub --step --sparam "jid=$JID" ./solve.sh

```

Fig 7.6: Job creating script for FX10 (3)

```
#!/bin/bash

PROCS=$1

if [ $PROCS -gt 0 ] 2> /dev/null
then
echo "${PROCS}PE OK"
else
echo "Not N"
exit
fi

rm -f finished
sed -i -e "s/DOMAIN=[1-9][0-9]*/DOMAIN=$PROCS/g" hecmw_part_ctrl.dat
mpirun -np 1 -host HOSTNAME ~/bin/hecmw_part1 2>&1 | tee part1.log
mpirun -np $PROCS -host HOSTNAME ~/bin/fistr1 2>&1 | tee fistr1.log
touch finished
```

Fig 7.7: Substitute script for job creating script in realtime computer


```

cd WORKDIR
if ssh -i PRIVKEY -p 22 USER@SERV '[ -e WORKDIR/finished ]'; then
  php jobstatus_change.php $1 'Transferring from Calc. Server' NULL
  scp -i PRIVKEY -P 22 USER@SERV:~/workdir/vis_out/*.inp ./

  if [ -e mesh_psf.0001.inp ]; then
    export DISPLAY=:0.0
    xset -dpms; xset s off
    php jobstatus_change.php $1 Visualing
    post
    jobstatus_change.php $1 'Finished' 'Success'
  else
    jobstatus_change.php $1 'Finished' 'Failed'
  fi

else

fi

```

Fig 7.8: Polling script

第8章 システムの外観と計算の実例

8.1 緒言

本章では，作成したクラウド CAE システムを用いて実際に CAE を行った場合の操作例を示す。

本章で利用する例題は，16 歯のギアのモデル (Fig 8.1) の歯部の節点変位を拘束し，ギアの上面から面荷重を与える，線形弾性静解析である。

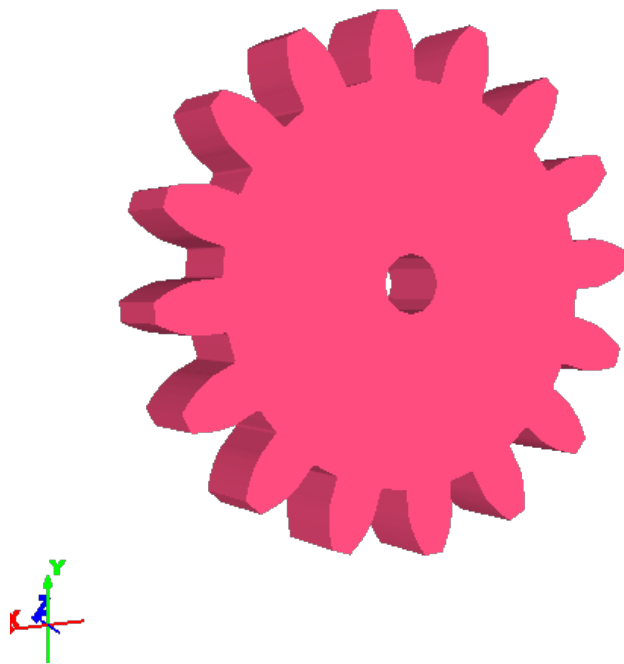


Fig 8.1: Gear Model

8.2 Gear モデルでの線形弾性静解析

8.2.1 トップページ

初めてログインしようとする場合であれば先にユーザ登録をしておく必要がある。まず、トップページからログインを行う (Fig 8.2)。それぞれのアカウントで各個人のデータは管理・整理される。



Fig 8.2: Toppage of Cloud CAE system

8.2.2 モデルの登録

形状データファイル、メッシュファイルの登録をサポートしている。アップロードファイルにモデル登録ページから行うことができる (Fig 8.3)。登録するデータに名前を付け、説明を追加することができる。

ようこそ, TEST 様
»東京大学大学院 奥田研究室
»革新的シミュレーション研究センター

Cloud ISTR α Cistr Cloud service for FrontISTR/Revocap

ホームページ ジョブ情報 モデル登録 形状データ メッシュ 解析モデル 解析結果 ユーザ情報 ログアウト

モデル登録

形状データファイル (CADデータ)

ファイル 選択されていません
データの種類 STL(Stereolithography) ▼
モデル名
モデル説明
他ユーザーに公開

メッシュファイル

ファイル 選択されていません
データの種類 HECMW ▼
モデル名
モデル説明
他ユーザーに公開

»東京大学大学院 奥田研究室
»革新的シミュレーション研究センター
»Copyright © , All Rights Reserved.

Fig 8.3: Model upload page

8.2.3 形状データリスト

アップロードしたモデルは形状データリストに登録される (Fig 8.4). 入力したモデルの名称, 説明とデータサイズ, 登録日時など一覧となって表示される.

The screenshot displays the 'CAD形状データリスト' (CAD Model List) page. At the top, there is a navigation menu with items: メインページ, ジョブ情報, モデル登録, 形状データ, メッシュ, 解析モデル, 解析結果, ユーザ情報, and ログアウト. The page title is 'CAD形状データリスト'. Below the title is a table listing uploaded models:

ID	モデル名	種類	説明	サイズ	日時	図	操作
8	????	STL		332.91 KB	2014-12-09 04:58:08		使用 削除
7	glass	STL		59.50 KB	2014-11-20 21:43:09		使用 削除
6	scaby	STL		2.98 MB	2014-11-19 23:50:36		使用 削除
5	geer	STL		1.84 MB	2014-11-19 22:34:32		使用 削除

Below the table is a form for uploading a new model:

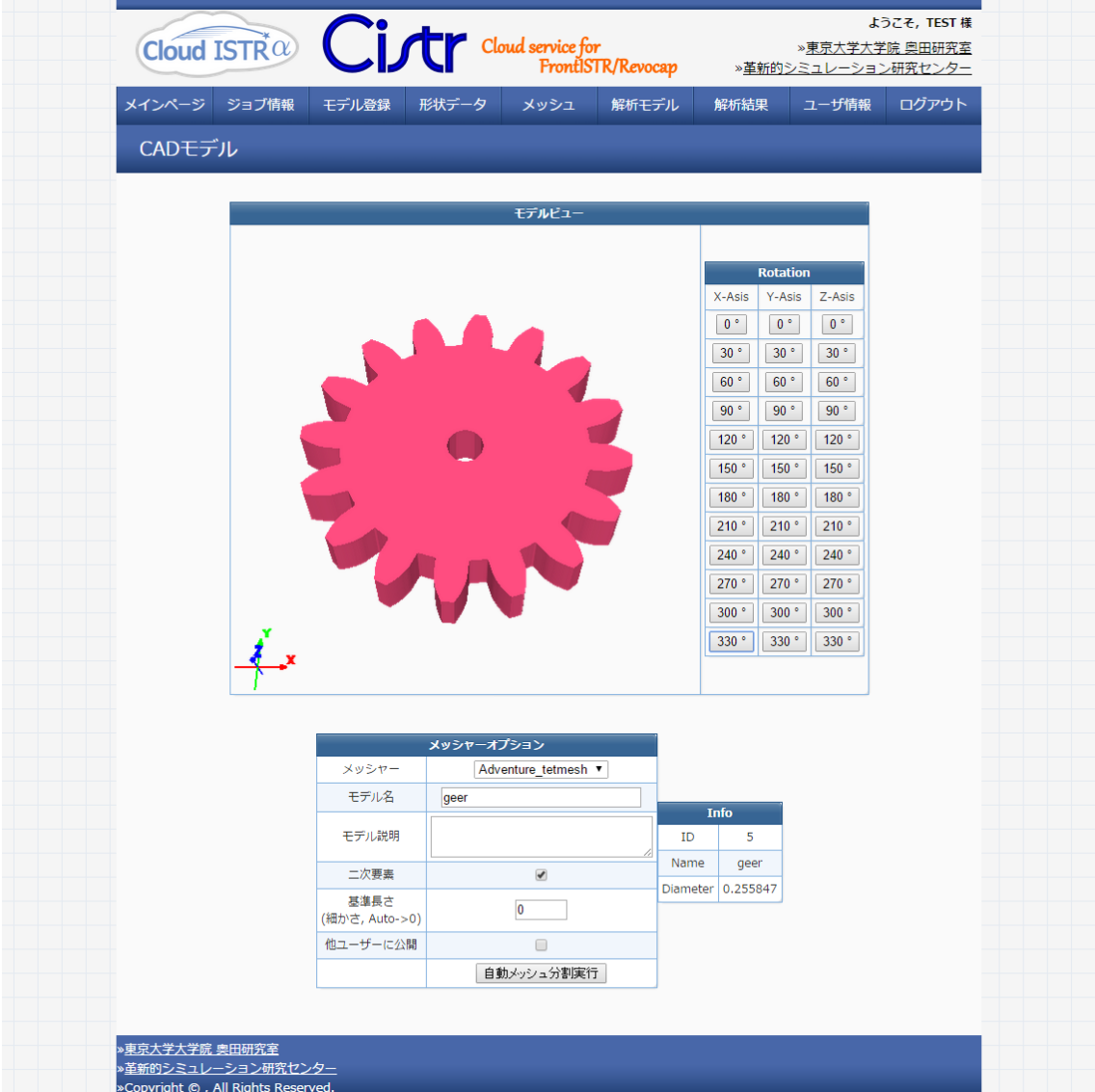
- ファイル: ファイルを選択 (選択されていません)
- データの種類: STL(Stereolithography) ▼
- モデル名: [Text Input Field]
- モデル説明: [Text Area]
- 他ユーザーに公開:
- アップロード

At the bottom of the page, there is a footer with the following text: 東京大学大学院 奥田研究室, 革新的シミュレーション研究センター, Copyright © , All Rights Reserved.

Fig 8.4: CAD model list page

8.2.4 形状データビュー

使いたいモデルをクリックすると、CADモデルの図が表示される (Fig 8.5)。この画面において、モデルを回転させ、外観を掴むことができる。このページから四面体メッシュのパラメータ、オプションを必要に応じて設定を行い、メッシュ生成を実行するジョブを生成することができる。メッシュによるメッシュ生成について、生成されるデータに名前を付け、説明を追加することができ節点密度の設定や二次要素の使用の有無などを記録しておくことができる。



The screenshot displays the 'CADモデル' (CAD Model) view page. At the top, there is a navigation bar with links: 'ホームページ', 'ジョブ情報', 'モデル登録', '形状データ', 'メッシュ', '解析モデル', '解析結果', 'ユーザ情報', and 'ログアウト'. The main content area is titled 'モデルビュー' (Model View) and features a 3D rendering of a pink gear. To the right of the gear is a 'Rotation' control panel with a grid of buttons for X-Asis, Y-Asis, and Z-Asis, ranging from 0° to 330° in 30° increments. Below the gear is a 'メッシュャーオプション' (Mesher Options) panel with the following fields:

メッシュャーオプション	
メッシュャー	Adventure_tetmesh
モデル名	geer
モデル説明	
二次要素	<input checked="" type="checkbox"/>
基準長さ (縮かさ, Auto->0)	0
他ユーザーに公開	<input type="checkbox"/>
自動メッシュ分割実行	

To the right of the options panel is an 'Info' table:

Info	
ID	5
Name	geer
Diameter	0.255847

At the bottom left, there is a footer with the text: '東京大学大学院 奥田研究室', '革新的シミュレーション研究センター', and 'Copyright ©, All Rights Reserved.'

Fig 8.5: CAD model view page

8.2.5 メッシュリスト

四面体メッシュによる FEM メッシュの生成が完了するとメッシュリストに登録される。モデルの名称、説明とデータサイズ、登録日時など一覧となって表示される (Fig 8.6)。ここで、使いたいメッシュをクリックし、表面分割角度を入力することで、メッシュビューを表示し、プロセスを進めることができる。表面分割角度は、続くプロセスでの面分割に影響する、

The screenshot shows the Cistr web interface. At the top, there is a navigation menu with items: メインページ, ジョブ情報, モデル登録, 形状データ, メッシュ, 解析モデル, 解析結果, ユーザ情報, ログアウト. Below the menu is a header for 'CAD形状データリスト'. The main content area features a table with the following data:

ID	モデル名	種類	説明	サイズ	日時	図	操作
8	????	STL		332.91 KB	2014-12-09 04:58:08		使用 削除
7	glass	STL		59.50 KB	2014-11-20 21:43:09		使用 削除
6	scaby	STL		2.98 MB	2014-11-19 23:50:36		使用 削除
5	geer	STL		1.84 MB	2014-11-19 22:34:32		使用 削除

Below the table is a form for uploading a new model. It includes fields for 'ファイル' (File), 'データの種類' (Data type) set to 'STL(Stereolithography)', 'モデル名' (Model name), and 'モデル説明' (Model description). There is also a checkbox for '他ユーザーに公開' (Public to other users) and an 'アップロード' (Upload) button.

At the bottom of the page, there is a footer with the following text: 東京大学大学院 奥田研究室, 革新的シミュレーション研究センター, Copyright ©, All Rights Reserved.

Fig 8.6: Mesh list page

8.2.6 メッシュビュー

メッシュリストにおいて使いたいメッシュをクリックすると、メッシュの図が生成される (Fig 8.7)。メッシュ図は、使用したメッシュのサイズにおいてリアルタイム処理、バッチジョブ処理が振り分けられる。リアルタイム処理が選択された場合は、完了後ブラウザは自動的に画面遷移を行い、自動的に表示される。メッシュサイズが大きい場合であれば、バッチジョブとして生成され、ジョブリストからメッシュビューの生成を確認する。

The screenshot shows the Cistr web application interface. At the top, there is a navigation menu with items: メインページ, ジョブ情報, モデル登録, 形状データ, メッシュ, 解析モデル, 解析結果, ユーザ情報, ログアウト. Below the menu is a header with the Cistr logo and the text "Cloud service for FrontISTR/Revocap".

The main content area is titled "メッシュ" (Mesh) and contains a "モデルビュー" (Model View) window. This window displays a 3D mesh of a gear. To the right of the model view are controls for "Drawing" (Wireframe/Surface), "Face" (Body0_0 to Body0_3), and "Rotation" (Init, Rev.).

Below the model view is a "境界条件設定" (Boundary Condition Setting) panel. It includes a table for setting boundary conditions and material properties.

境界条件設定	
解析種類	線形弾性静解析
Material 材料物性	(Preset) Steel
	Density: 7860
	Poisson's ratio: 0.29
	Young's modulus: 2.06e+11
BOUNDARY 節点拘束・強制変位	BND0 <input checked="" type="checkbox"/> x 0 <input checked="" type="checkbox"/> y 0 <input checked="" type="checkbox"/> z 0
CLOAD 集中荷重	CL0 x 0 y 0 z 0
DLOAD 面荷重	DL0 0
	反映 クリア

At the bottom right, there is an "Info" panel showing details for the selected mesh:

Info	
ID	22
Name	gear
Node	83544
Elem	52493

At the bottom of the page, there is a footer with the text: "東京大学大学院 奥田研究室 革新的シミュレーション研究センター Copyright © , All Rights Reserved."

Fig 8.7: Mesh view page(1)

面の切り替え並びにワイヤフレーム／サーフェスの切り替えを行った例である (Fig 8.8). 面の切り替えはリストからの選択形式で行われる. リストの選択を行うと瞬時に画面の反映が行われる. ワイヤフレーム表示とサーフェス表示の切り替えも同様である.

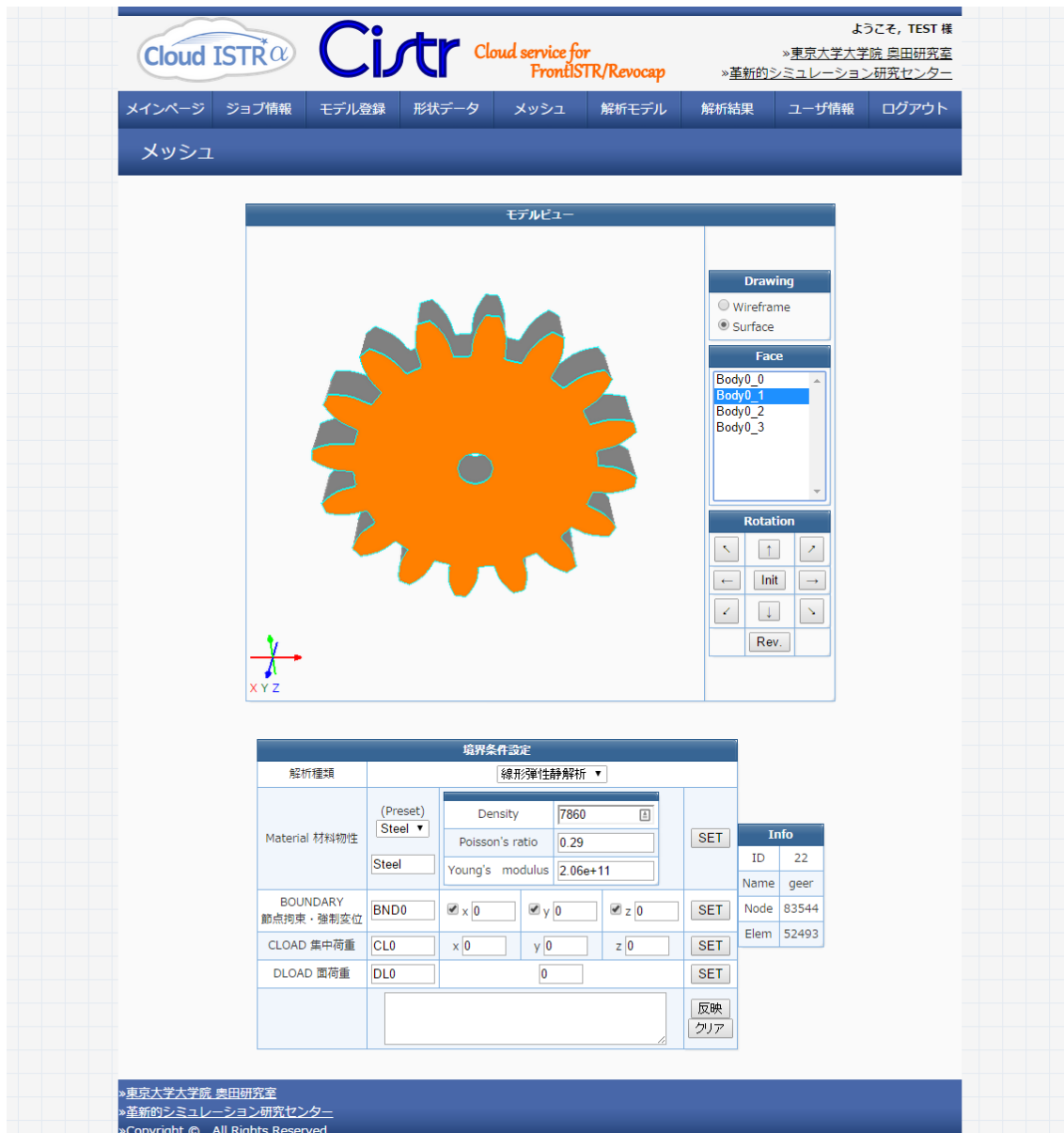


Fig 8.8: Mesh view page(2)

面や、表示手法に関わらず、すべての場合においてユーザ任意に回転させることができる (Fig 8.9).



Fig 8.9: Mesh view page(3)

材料物性は解析モデル全体に設定され、複数の材料物性を与える機能はサポートされていない。節点強制変位は、各面についてそれぞれ与えることができる。集中荷重は、面に属する全ての節点においてベクトル値で与えることができる。面内に属する一部の節点にのみ集中荷重を与える機能はサポートされていない。面荷重は、面に属する節点に直交する力としてスカラ値で与えることができる。これらの境界条件を設定すると画面下部のテキストエリアに表示される (Fig 8.10)。画面下部のテキストエリアに表示される境界条件は、テキストであり、別のモデルに使うことができる。



Fig 8.10: Mesh view and setting condition

8.2.7 境界条件の確認

境界条件の反映を行うと設定した境界条件を図にすることができる (Fig 8.11).. 条件図は、使用したメッシュのサイズにおいてリアルタイム処理、バッチジョブ処理が振り分けられる。リアルタイム処理が選択された場合は、完了後ブラウザは自動的に画面遷移を行い、自動的に表示される。メッシュサイズが大きい場合であれば、バッチジョブとして生成され、ジョブリストから条件図の生成を確認する。

節点強制変位は、茶色の丸型のマークが節点に表示される。集中荷重の場合は節点に対して緑色のベクトルが表示される。面荷重では青色の四角形マークが表示される。

目的のモデルの生成を確認すると、解析モデルとして登録を行うことができる。登録するデータに名前を付け、説明を追加することができる。

The screenshot displays the Cistr web application interface. At the top, there is a navigation menu with items: メインページ, ジョブ情報, モデル登録, 形状データ, メッシュ, 解析モデル, 解析結果, ユーザ情報, and ログアウト. The main content area is titled 'モデルビュー' (Model View) and features a 3D visualization of a gear mesh. To the right of the mesh is a 'Rotation' control panel with a table of rotation angles for X, Y, and Z axes.

Rotation		
X-Asis	Y-Asis	Z-Asis
0 °	0 °	0 °
30 °	30 °	30 °
60 °	60 °	60 °
90 °	90 °	90 °
120 °	120 °	120 °
150 °	150 °	150 °
180 °	180 °	180 °
210 °	210 °	210 °
240 °	240 °	240 °
270 °	270 °	270 °
300 °	300 °	300 °
330 °	330 °	330 °

Below the model view is a '解析モデルの登録' (Register Analysis Model) form with the following fields:

- モデル名 (Model Name): geer
- モデル説明 (Model Description): [Empty text area]
- 他ユーザーに公開 (Public to other users): [Checked checkbox]
- 解析モデルの登録 (Register Analysis Model) button

At the bottom of the page, there is a footer with the text: 東京大学大学院 奥田研究室, 革新的シミュレーション研究センター, Copyright © , All Rights Reserved.

Fig 8.11: Mesh with condition view

8.2.8 解析モデルビュー

登録した解析モデルは解析モデルリストに登録され、再使用ができる。入力した解析モデルの名称、説明とデータサイズ、登録日時など一覧となって表示される。解析モデルリストより解析モデルを選択し、解析モデルビューを生成する。解析モデルビューは境界条件図と同じものである。ここで、実際に計算を行う計算サーバを選択し、並列数を指定する (Fig 8.12)。この例では、ギアの歯の部分固定し、上面から面荷重を与えている。ソルバ解法と、反復法であれば前処理を選択して実行することで外部の計算機にジョブが送信され、自動的に解析が行われる。

The screenshot shows the Cistr web interface for the 'Model View' of a gear mesh. The interface includes a navigation menu at the top, a 3D model of the gear mesh, a rotation table, and solver settings.

Rotation Table:

X-Axis	Y-Axis	Z-Axis
0 °	0 °	0 °
30 °	30 °	30 °
60 °	60 °	60 °
90 °	90 °	90 °
120 °	120 °	120 °
150 °	150 °	150 °
180 °	180 °	180 °
210 °	210 °	210 °
240 °	240 °	240 °
270 °	270 °	270 °
300 °	300 °	300 °
330 °	330 °	330 °

Solver Settings:

解析	
解析サーバ	Oakleaf-fx (16core/node)
並列数	16
ソルバ設定	
ソルバ解法	反復法(CG)
ソルバ前処理	不完全コレスキー分解 IC(0)
反復回数	20000
打ち切り残差	1e-06
解析実行	

Info Table:

ID	9
Mesh_ID	23
Name	geer
Description	

※反復法は、並列性能が良く大規模問題に向きますが、悪条件問題では収束性が良くありません。
 ※直接法は、問題規模に応じてメモリ・計算時間が大きくなり、並列性能が良くありません。

©東京大学大学院 奥田研究室
 革新的シミュレーション研究センター
 Copyright ©, All Rights Reserved.

Fig 8.12: Model view

8.2.9 解析結果

システムは、解析の完了を一定時間ごとに確認し、その完了を確認すると解析結果をシステムに転送し、図を作成する。

この解析結果は、有効なものである場合、名前を付け、説明を追加し登録を行うことができる。登録を行った解析結果は、再度呼び出すことができる。

変形強調図 (Fig 8.13) の例である。歯部固定で、上面に面荷重を与えたので中心部に向かって凹む形状になったことが図から読み取れる。



Fig 8.13: Deformed image of gear model FEM result

変位コンター図 (Fig 8.13) を切り替えて、結果図を見ることができる。外周部から中心部に行くにしたがって変位の絶対値が大きくなることが図から読み取れる。変形強調図から読み取った内容と同じことが理解できる。

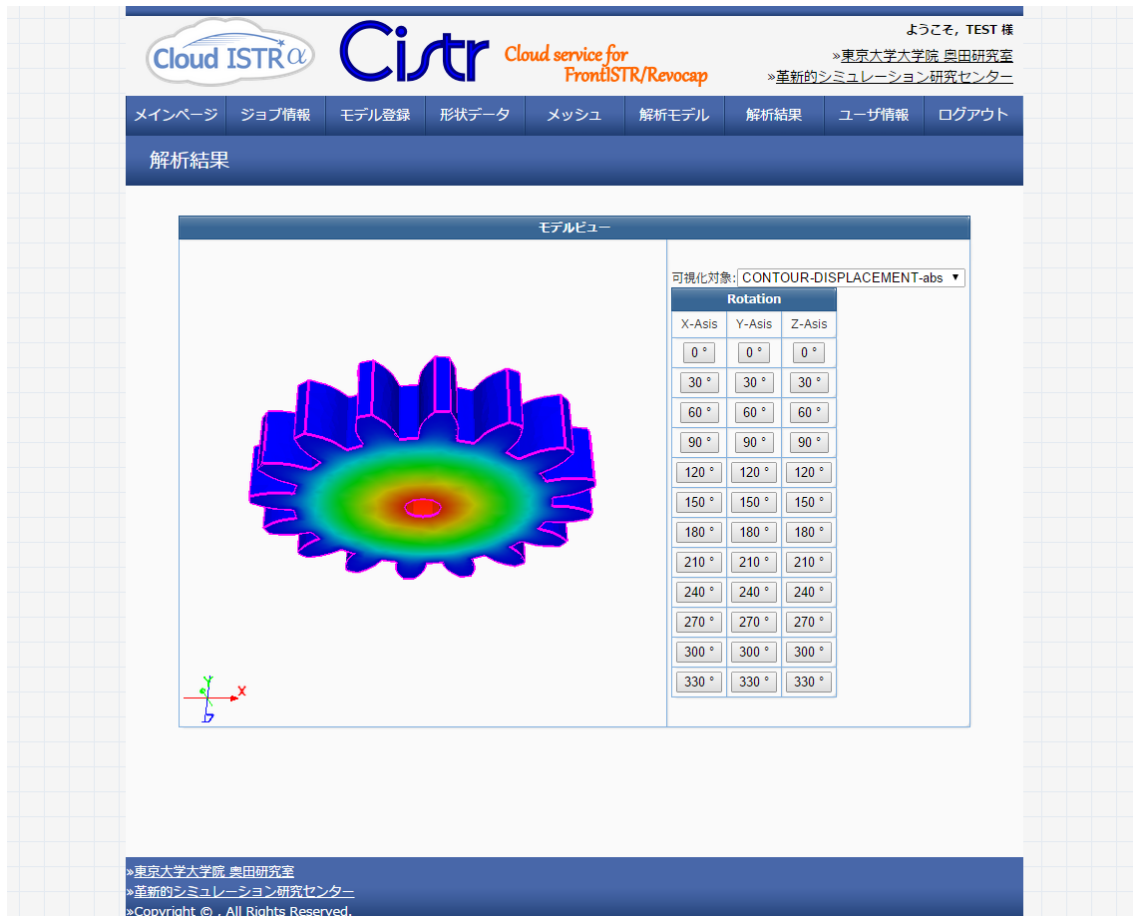


Fig 8.14: Displacement contour of gear model FEM result

ミーゼス応力コンター図 (Fig 8.13) についても切り替えて、結果図を見ることができる。ミーゼス応力は外周部の歯の付け根に集中していることが図から読み取れる。

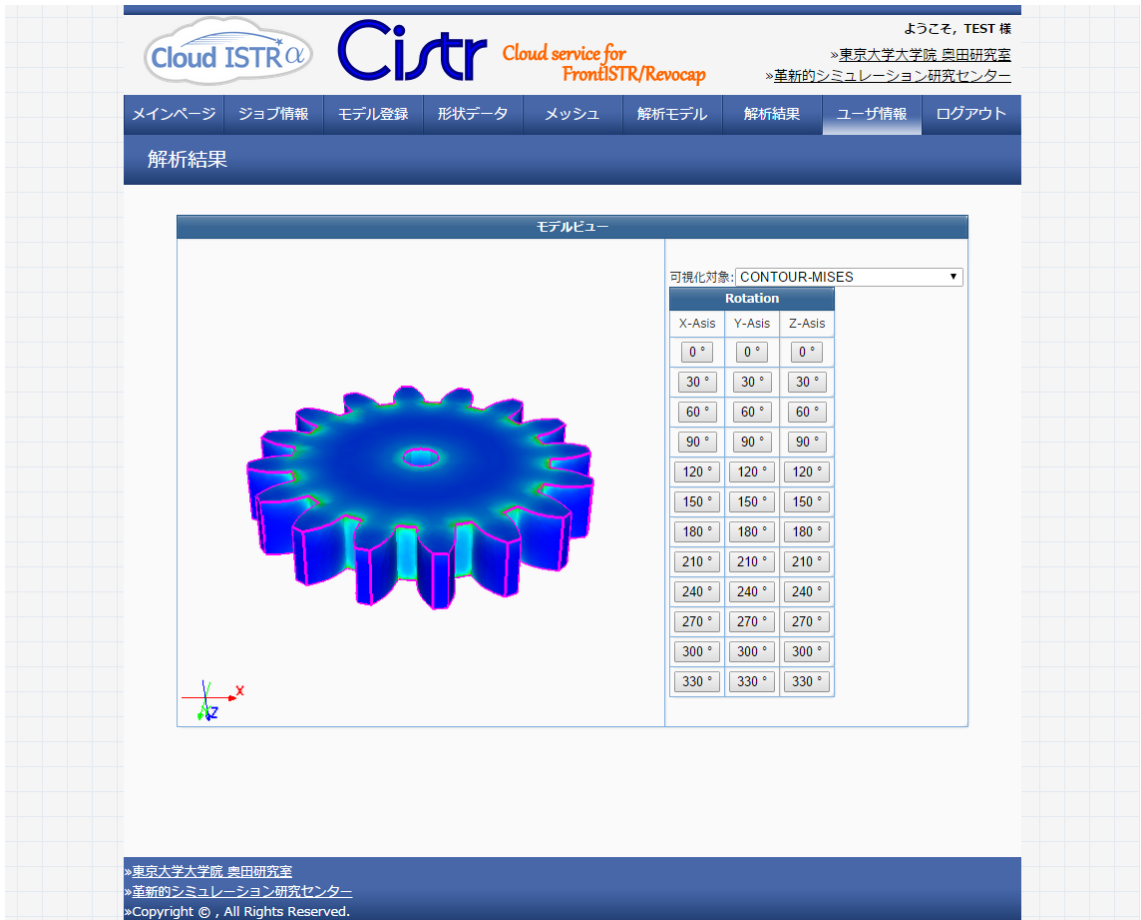


Fig 8.15: Von mises stress contour of gear model FEM result

8.3 結言

本章では、実際のシステムの利用イメージを示し、操作の説明を行った。

第9章 システムの評価

9.1 緒言

作成したクラウド CAE システムを評価するために、FrontISTR ユーザである学生 4 人、研究者 4 人の計 8 名に自由記述方式で評価を頂いた。この 8 名は、十分に CAE に精通しているユーザである。

9.2 評価

概ね、シンプルさについての肯定的な意見が多かった。問題点としてセキュリティ性の向上、計算時間見積もり機能の実装、例外が起きた時の処理、インタフェイスなどの改善の要望を受けた。具体的には以下のような意見があった。

- モデル回転など自由度が低いことが逆に便利である。ちょうどの角度回転できることは便利。
- パラメータスタディするには便利そうだが条件設定は柔軟化してほしい。
- 条件設定を簡潔に面の名前などで設定するのは便利かもしれない。データハンドリング性を上げていくべき。ある節点の結果だけ抽出できればいいかもしれない。
- 同種の研究ではモデルの 3D 処理の高度化を目指すことが多いが、あえて画像と 2D に落とす発想は面白い視点。
- メッシュファイルだけをアップというのはシンプルで良い。
- いくつかの機能をモジュール化していけば広まるのではないか。
- 非常に操作性が良かった。
- プリポストが貧弱なので確認程度にしか使えないかもしれないが、それでも一通りのことができるのは大きなメリット。

評価を行ったユーザは CAE に十分な知識と経験を持つユーザであり、機能が少ないことに対する低評価は当然であるが、シンプルさへの評価が高いことは本研究の目的設定は適当であったと考えられる。

9.3 結言

本章では、頂いた評価についてまとめ、目的設定の妥当性を検討した。

第10章 結論

10.1 結論

本研究では、CAEの一連のプロセスを支援するライブラリ、プログラムを作成し、そのライブラリ、プログラムを組み合わせ、クラウドサービスを作成した。以下に各章の内容を簡潔にまとめる。

第2章では、CAEにおいて行うステップについて述べ、その詳細について述べた。次に、CAEに用いられる有限要素法について述べ、計算においてその核となる連立一次方程式の解法について述べた。

第3章では、クラウドコンピューティングの定義について述べ、複数の視点からの分類について述べた。次に、CAEにクラウドを用いた研究やサービスについて、レビューを行い、本研究の位置づけを述べた。第4章では、本研究で作成するシステムと連携するソフトウェアについて述べた。第5章では、Web上に三次元モデルを展開する手法について述べた。その手法において、三次元モデルの操作の表現について説明をした。第6章では、第5章の手法を、汎用的な一般的なライブラリ、プログラムとして作成を行い、その内容について述べた。第7章では、第6章で作成したプログラム類を元に、Web上で展開するためのスクリプト類を作成し、その内容について述べた。第8章では、第7章で作成したシステムで実際の操作を図として実例を述べ、CAEの一連のプロセスを行った例を示した。第9章では、本研究で作成したシステムについて評価を行い、その内容について述べた。

本研究では、様々なプログラムを用いたが、ソフトウェアはすべてオープンソース・ソフトウェアまたは本研究で作成したソフトウェアだけで構成されており、低コストにCAEができる環境が整った。シェルでの操作、コマンドラインを使うこと無くなったことで、CAEの導入検討を検討しているユーザへのアプローチになっていると考える。また、CAE利用者向けには、

REVOCAP_Prepostではメモリ不足で実行できなかった、数百万節点～数千万節点の問題において、全く大規模の問題を操作していることを意識することなくユーザとしては作業を行うことが出来た。

システムの操作もグラフィカルで簡単なものであるので、CAEを初めて利用する人にも利用しやすくなったことは重要である。本システムでは、一切の作業でデータが保存されていくため、データの滅失や散逸も防げるようになったことも大きい。

ユーザ評価を実施したものの、サンプル数が足りていなく属性も偏っていることから定量的な評価が出来ていない。

10.2 今後の展望

現状で本システムは、線形静解析にのみ対応しているが、解析種類の充実は必須である。少なくとも FrontISTR で整備されている解析機能は、使うことができるようにする必要がある。また境界条件設定についてもクライアント型のプリプロセッサに劣る部分が多くあり、機能を改善する必要がある。特に集中荷重の与え方について、面内の節点をマウスのドラッグなどの操作で選択できるシステムを実装することで実用性の向上を図ることができる。

モデルの視点操作については、さらに直感にあう操作に改善できる可能性があり、マウスのドラッグ操作などを適用することで利便性の向上を図ることができると考える。さらに、機能のモジュール化を進め、ユーザへの提供を行うことで他の研究の効率化に貢献したいと考える。

謝辞

本研究を行うにあたっては、多くの方々にご指導とご協力を賜りました。この場を借り、厚く御礼申し上げます。

まず、指導教官である東京大学大学院新領域創成科学研究科人間環境学専攻 教授 奥田洋司先生に深く感謝致します。修士課程で初めてこの分野に入った時から丁寧なご指導を賜りました。研究の方向性の決定や、躓いたときに極めて的確なアドバイスを頂きました。東京大学大学院新領域創成科学研究科人間環境学専攻 講師 橋本学先生には、日頃よりお世話になり、的確なご指導を頂きましたこと感謝致しております。特任研究員の北山健氏には、指導教官とは違った立場として多くの助言をして頂きましたことを感謝申し上げます。

株式会社キャトルアイ・サイエンス代表取締役上島豊氏には、本研究の方向性を決定していく最中に様々なご提案やご助言を頂きましたことを心より感謝申し上げます。

FrontISTR 研究会を通し、様々な機能の提案や操作性改善などユーザ評価にご協力を頂きました独立行政法人海洋研究開発機構 小川道夫氏、公益財団法人鉄道総合技術研究所 林雅江氏、アドバンスソフト株式会社 徳永健一氏 末光啓二様 袁熙氏には深く感謝申し上げます。

研究室の先輩として、当分野の基本的なことからプログラミングまで様々な技術的なアドバイスやご指導を頂きました合同会社 PExProCS 代表社員 後藤和哉氏に深く感謝致します。

研究室の日々の生活や、システム開発のユーザ評価など特にお世話になりました東京大学大学院新領域創成科学研究科 稲垣和久氏、森田直樹氏、東京大学工学部生野達大氏に感謝申し上げます。また、研究室のメンバーとして、日々お世話になりました Olav Aanes Fagerlund 氏、三橋祐太氏、相良光志氏、芳野修一氏にも併せて感謝申し上げます。

そして奥田研究室の渡辺夏実様にも様々な日常の事務的なことを始め大変お世話になりました。

最後に、著者を見守り、励まして頂きました家族や友人にも心より感謝しております。

参考文献

- [1] TOP500.org. <http://top500.org/>. 2015 年 1 月 1 日閲覧.
- [2] 奥田洋司. マルチパラメータサーベイ型シミュレーションを支えるシステム化技術に関する研究. 学際大規模情報基盤共同利用・共同研究拠点 平成 24 年度共同研究最終報告書, 2013.
- [3] HPC クラウドサービス FUJITSU Technical Computing Solution TC クラウド. <http://jp.fujitsu.com/solutions/hpc/tcccloud/>. 2015 年 1 月 1 日閲覧.
- [4] Amazon EC2. <http://aws.amazon.com/jp/ec2/>. 2015 年 1 月 1 日閲覧.
- [5] さくらのクラウド. <http://cloud.sakura.ad.jp/>. 2015 年 1 月 1 日閲覧.
- [6] 和田義孝. ウェブ・ベース CAE システム:CASOW の開発. 計算力学講演会講演論文集, Vol. 16, pp. 997–998, 2003.
- [7] 和田義孝, 力武俊介, 菊池正紀. FEM を用いた Web ベース CAE システムにおける任意形状解析. 計算力学講演会講演論文集, Vol. 18, pp. 111–112, 2005.
- [8] 奥田洋司, 早田浩平, 橋本学, 上島豊. クラウド CAE システムを用いた効率的な有限要素モデリング. 計算力学講演会講演論文集, Vol. 18, , 2013.
- [9] オープンソース構造解析システム FrontISTR. <http://www.multi.k.u-tokyo.ac.jp/FrontISTR/>. 2015 年 1 月 1 日閲覧.
- [10] MUMPS : a parallel sparse direct solver. <http://mumps.enseeiht.fr/>. 2015 年 1 月 1 日閲覧.
- [11] Patrick R Amestoy, Abdou Guermouche, Jean-YvesL ’ Excellent, Stéphane Pralet. Hybrid scheduling for the parallel solution of linear systems. *Parallel computing*, Vol. 32, No. 2, pp. 136–156, 2006.
- [12] Anshul Gupta, Seid Koric, and Thomas George. Sparse matrix factorization on massively parallel computers. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, p. 1. ACM, 2009.
- [13] Xiaoye S Li and James W Demmel. Superlu_dist: A scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Transactions on Mathematical Software (TOMS)*, Vol. 29, No. 2, pp. 110–140, 2003.

- [14] Pascal Hénon, Pierre Ramet, and Jean Roman. Pastix: a high-performance parallel direct solver for sparse symmetric positive definite systems. *Parallel Computing*, Vol. 28, No. 2, pp. 301–321, 2002.
- [15] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct Methods for Sparse Matrices (Monographs on Numerical Analysis)*. Oxford University Press, 8 1989.
- [16] 藤野清次. CG 法の最近の前処理のロバスト性と効率化について: 閾値によるドロップリングと対角緩和処理. 数理解析研究所講究録, Vol. 1362, pp. 13–21, 2004.
- [17] J ANDVANDERVORST Meijerink and Henk A van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix. *Mathematics of computation*, Vol. 31, No. 137, pp. 148–162, 1977.
- [18] Michele Benzi, Carl D Meyer, and Miroslav Tůma. A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM Journal on Scientific Computing*, Vol. 17, No. 5, pp. 1135–1149, 1996.
- [19] Michele Benzi, Jane K Cullum, and Miroslav Tůma. Robust approximate inverse preconditioning for the conjugate gradient method. *SIAM Journal on Scientific Computing*, Vol. 22, No. 4, pp. 1318–1332, 2000.
- [20] Michele Benzi and Miroslav Tůma. A robust incomplete factorization preconditioner for positive definite matrices. *Numerical Linear Algebra with Applications*, Vol. 10, No. 5-6, pp. 385–400, 2003.
- [21] Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. <http://geuz.org/gmsh/>. 2015 年 1 月 1 日閲覧.
- [22] ADVENTURE PROJECT. <http://adventure.sys.t.u-tokyo.ac.jp/jp/>. 2015 年 1 月 1 日閲覧.
- [23] 寺田賢二郎 (編). 非線形有限要素法-弾塑性解析の理論と実践. 森北出版, 2012.
- [24] METIS - Serial Graph Partitioning and Fill-reducing Matrix Ordering. <http://glaros.dtc.umn.edu/gkhome/metis/metis/overview>. 2015 年 1 月 1 日閲覧.
- [25] Peter Mell and Tim Grance. The NIST definition of cloud computing. 2011.
- [26] 大規模アセンブリ構造・マルチ力学対応プレポスト REVOCAP PrePost. http://www.ciss.iis.u-tokyo.ac.jp/project/rss/software/06_info.html. 2015 年 1 月 1 日閲覧.
- [27] エンジニアリングデータ管理システム 「ASNARO」. <http://www.i4s.co.jp/asnar/asnaroinfo.html>. 2015 年 1 月 1 日閲覧.

- [28] OpenGL. <https://www.opengl.org/>. 2015年1月1日閲覧.
- [29] libpng. <http://www.libpng.org/pub/png/libpng.html>. 2015年1月1日閲覧.
- [30] FX10 スーパーコンピュータシステム (OakleafFX, Oakbridge-FX) . <http://www.cc.u-tokyo.ac.jp/system/fx10/>. 2015年1月1日閲覧.
- [31] PC クラスタ TC. <http://www.multi.k.u-tokyo.ac.jp/computers.php>. 2015年1月1日閲覧.