

# Local Proper Orthogonal Decomposition に対する階層型領域分割並列化

## Hierarchical Domain Decomposition Parallelization for Local Proper Orthogonal Decomposition

新館京平<sup>1)</sup> 森田直樹<sup>2)</sup> 金子栄樹<sup>3)</sup> 三目直登<sup>4)</sup>  
Kyohei Shintate, Naoki Morita, Kaneko Shigeki and Naoto Mitsume

<sup>1)</sup>筑波大学システム情報工学研究群 (〒 305-8573 茨城県つくば市天王台 1-1-1, E-mail: shintate.kyohei.qm@alumni.tsukuba.ac.jp)

<sup>2)</sup>博 (環境) 筑波大学システム情報系 助教 (〒 305-8573 茨城県つくば市天王台 1-1-1, E-mail: nmorita@kz.tsukuba.ac.jp)

<sup>3)</sup>博 (工) 名古屋工業大学大学院工学研究科 助教

(〒 466-0061 愛知県名古屋市中昭和区御器所町, E-mail: kaneko.shigeki@nitech.ac.jp)

<sup>4)</sup>博 (工) 筑波大学システム情報系 助教 (〒 305-8573 茨城県つくば市天王台 1-1-1, E-mail: mitsume@kz.tsukuba.ac.jp)

Model order reduction techniques represent a very effective approach for computationally intensive applications. On the other hand, the efficiency of these systems is constrained by the large number of degrees of freedom due to the complexities of Partial Differential Equations. To tackle these issues, the combination of Reduced Order Model and Domain Decomposition Methods (DDM) has been explored. Although various coupling methods exist, there are few examples implemented in distributed memory parallel computers, which are highly valuable for memory usage. In this study, we propose an overlapping DDM in distributed memory computer, based on generalized hierarchical graph decomposition. We discuss the computing efficiency and parallel computational performance through its application in diffusion equation.

**Key Words** : Parallel Computing, Domain Decomposition, Graph Structure, Reduced Order Modeling

### 1. 序論

Reduced Order Model (ROM) [1] は、数値計算における高次元の未知数を特徴的な構造で捉えて低次元化し、高速に解析する手法である。パラメトリックな偏微分に代表される繰り返し解析を実行する問題に対して有効な手法であり、多彩な工学問題に応用されている。

一方、ROM 解析の計算効率は、偏微分方程式の複雑さに起因する自由度数の多さによって制約されている。この問題に対処するために、領域分割法が活用されてきた。領域分割法とは、解析モデルを幾何学的に分割することで自由度を削減する手法である。有限要素法をはじめとする高精度な解析システムを用いた解析では、大規模問題の計算時間の削減、メモリ容量制限の緩和を目的として、分割領域に並列プロセッサが割り当てられ、分散メモリ型並列計算に活用されている。ROM 解析では、各分割領域で ROM を構築するため、計算複雑性を部分的に回避することができ、マルチフィジックス問題 [2] など局所的に複雑な現象に対して活用されている。加えて、高精度な解析システムとのカップリング手法 [3,4] も提案されており、対象問題に応じて柔軟に解析を行うことが可能になる。しかし、領域分割法を用いた ROM 解析は、高精度な解析システムのように分散メモリ型並列計算環境で実装されている例は少なく、大規模問題への適用に課題がある。Cao [5] らは、領域分割前の大域的な領域で構築された ROM に対して並列計算を行っているが、分割領域毎に構築された ROM に対する並列計算は実現されていない。

このような背景から、本研究では大規模問題への ROM 解析の適用を目的として、階層的なグラフ構造に基づく ROM 解析手法を提案する。提案手法により、分散メモリ型並列計算環境において領域分割型の ROM 解析が実現された。加えて、並列計算を行う領域と ROM を構築する領域が独立に決定されるため、計算環境に依存しない並列計算が可能になった。本研究では、ROM 解析手法として Local Proper Orthogonal Decomposition (Local POD) [3] を採用し、提案手法の計算効率および並列計算性能の評価を行う。

### 2. Local POD

#### (1) Snapshot POD

ROM の構築に用いる基底選択の手法として Snapshot POD [6] を用いる。Snapshot POD は、はじめに高自由度の解析システムで数値計算または実験による測定を行うことで、Snapshot と呼ばれるトレーニングデータを収集し、形成される Snapshot matrix に対して固有直交分解 (POD) を行う手法である。

$n_{\text{snap}}$  本の Snapshot データ  $\mathbf{d}_{(i)} \in \mathbb{R}^n$  ( $i = 1, \dots, n_{\text{snap}}$ ) からなる Snapshot matrix  $\mathbf{X} \in \mathbb{R}^{n \times n_{\text{snap}}}$  を次式に示す。

$$\mathbf{X} = [\mathbf{d}_{(1)}, \dots, \mathbf{d}_{(n_{\text{snap}})}] \quad (1)$$

ここで、 $n$  は Snapshot データの大きさを表し、数値計算における総自由度数に対応する。

続いて、Snapshot matrix  $\mathbf{X}$  の特徴を表現する  $k_{\text{POD}}$  本の基底  $\mathbf{v}_i$  ( $i = 1, \dots, k_{\text{POD}}$ ) を取得することを考える。そ

のためには、各 Snapshot データと基底によって張られる空間の誤差が最小になればよく、次式で表される。

$$v_i = \arg \min_{g_i} \sum_{j=1}^{n_{\text{snap}}} \left\| d_{(j)} - \sum_{l=1}^{k_{\text{POD}}} g_l g_l^T d_{(j)} \right\|^2$$

with  $G^T G = I$ ,  $G = [g_1, \dots, g_{k_{\text{POD}}}] \in \mathbb{R}^{n \times k_{\text{POD}}}$  (2)

式 (2) は、Snapshot matrix  $X$  に対して特異値分解を行うことで求められる。

$$X = U \Sigma W^T \quad (3)$$

ここで、 $U \in \mathbb{R}^{n \times r}$  ( $r = \text{rank}(X)$ ) は左特異値行列、 $\Sigma = \text{diag}(\sigma_i)$  ( $i = 1, \dots, r$ ) は特異値行列、 $W \in \mathbb{R}^{n \times r}$  は右特異値行列であり、 $i$  番目の特異値を  $\sigma_i$  で表す。左特異値行列  $U$  の内はじめての  $k_{\text{POD}}$  本を選定し、POD 基底  $V = [v_1, \dots, v_{k_{\text{POD}}}] \in \mathbb{R}^{n \times k_{\text{POD}}}$  を得る。

特異値  $\sigma_i$  は Snapshot データの寄与度を示すため、適当な閾値  $\varepsilon_{\text{POD}}$  を用いて次式を満たすように基底の数  $k_{\text{POD}}$  が決定される。

$$\frac{\sum_{i=1}^{k_{\text{POD}}} \sigma_i}{\sum_{i=1}^n \sigma_i} > 1 - \varepsilon_{\text{POD}} \quad (4)$$

## (2) POD-Galerkin 法

本研究では高制度な解析システムとして有限要素法を採用し、解ベクトルを POD 基底で近似することで ROM を構築する。

支配方程式として次式で表される非定常拡散方程式を用いる。

$$\frac{\partial u}{\partial t} - k \nabla^2 u = f \quad (5)$$

$k$ ,  $u$  はそれぞれ拡散係数、拡散する物理量を表す未知数であり、解析領域を 3 次元の有界領域  $\Omega$  とすると、 $f$  は解析領域  $\Omega$  で発生する物理量を表すソース項である。境界条件は次式で表される。

$$u = u_D \quad \text{on} \quad \Gamma_1 \quad (6)$$

$$\nabla u \cdot \mathbf{n} = q_N \quad \text{on} \quad \Gamma_2 \quad (7)$$

$u_D$  は Dirichlet 境界  $\Gamma_1$  上での関数解  $u$  の値であり、 $\mathbf{n}$  は Neumann 境界  $\Gamma_2$  上の外向き法線ベクトル、 $q_N$  は境界  $\Gamma_2$  で与える物理量を表す関数である。式 (5) を有限要素法により適切に離散化することで、離散化された式 (8) が得られる。

$$A \mathbf{u} = \mathbf{f} \quad (8)$$

ここで、行列  $A \in \mathbb{R}^{n \times n}$  は係数行列であり、 $\mathbf{u}, \mathbf{f} \in \mathbb{R}^n$  はそれぞれ解ベクトル、右辺ベクトルである。 $n$  は数値計算における総自由度数を表し、節点数と節点における自由度の積となる。

次に、POD 基底を用いて ROM を構築する。本研究では時間方向へのパラメータ変化を考え、トレーニングデータとして有限要素法の解ベクトルを取得する。また、有限要素法による離散化の過程で、Neumann 境界条件 (7) は支配方程式への弱形式化によって自動的に満たされるのに対して、Dirichlet 境界条件 (6) は独立に要

請される。そのため、事前に計算した有限要素法の解ベクトルから Dirichlet 境界条件の値を減じた値を  $i$  番目の Snapshot データ  $d_{(i)} \in \mathbb{R}^n$  とし、Snapshot matrix  $X$  を構成する。

解ベクトル  $\mathbf{u}$  は、Snapshot matrix  $X$  から得られた POD 基底  $V$  によって近似されるとする。

$$\mathbf{u} \approx V \tilde{\mathbf{u}} + \mathbf{g} \quad (9)$$

ここで、 $\tilde{\mathbf{u}} \in \mathbb{R}^{k_{\text{POD}}}$  は基底ベクトルの結合係数を表す係数ベクトルであり、 $\mathbf{g} \in \mathbb{R}^n$  は Dirichlet 境界条件を満たし、その他ではいたるところ 0 のベクトルである。式 (9) で表される近似式は Galerkin 法と組み合わせで用いられ、係数行列 (8) は次式に縮約される。

$$V^T A V \tilde{\mathbf{u}} = V^T \mathbf{f} \quad (10)$$

また、縮約された係数行列  $\tilde{A} = V^T A V \in \mathbb{R}^{k_{\text{POD}} \times k_{\text{POD}}}$ 、縮約された右辺ベクトル  $\tilde{\mathbf{f}} = V^T \mathbf{f} \in \mathbb{R}^{k_{\text{POD}}}$  を用いて、式 (10) の両辺を表すと次式となる。

$$\tilde{A} \tilde{\mathbf{u}} = \tilde{\mathbf{f}} \quad (11)$$

## (3) Overlapping 型領域分割

領域分割法は、分割領域が重ならない non-overlapping 型領域分割と、分割領域が重なる overlapping 型領域分割に分けられる。本研究では、数値解析手法により離散化して得られる連立一次方程式の解法に注目し、それらを数値解析手法によらず統一的に扱うことが可能な overlapping 型領域分割法を採用する。並列計算に必要な演算に関する定式化は、3.2 節にて詳述する。

はじめに、以下の定義のもと節点集合  $X$  を  $N$  個の節点集合  $X^{(i)}$  ( $i = 1, \dots, N$ ) に重複なく分割する。

$$X^{(i)} \cap X^{(j)} = \emptyset \quad (i \neq j) \quad (12)$$

$$\bigcup_{i=1}^N X^{(i)} = X \quad (13)$$

式 (12), (13) により分割したとき、各領域に属する節点集合を内部節点集合とし、領域  $i$  に属する内部節点数を  $n^{(i)}$  で表す。オーバーラッピング領域を加味した節点集合  $\bar{X}^{(i)}$  は次式で与えられる。

$$\bar{X}^{(i)} = X^{(i)} \cup X_{\text{ovl}}^{(i)} \quad (14)$$

ここで、 $X_{\text{ovl}}^{(i)}$  は領域  $i$  が隣接する節点からなる集合のうち、隣接する領域に属する節点に限定した集合である。なお、節点  $a$  に定義される形状関数  $N_a(x)$  に対して、形状関数の積  $N_a(x)N_b(x)$  が非零となる座標  $x$  が存在するとき、節点  $a$  は節点  $b$  と隣接するとみなす。本研究では、METIS [7] による計算負荷の均一化および通信量の最小化に基づく領域分割法を採用するが、後述する提案手法はこれに限らず任意の overlapping 型領域分割法へ適用可能である。

次に、領域分割前の節点集合  $X$  に対応する係数行列  $A \in \mathbb{R}^{n \times n}$  を、領域  $i, j$  の内部節点集合  $X^{(i)}, X^{(j)}$  に対応する行列  $A^{(i, j)} \in \mathbb{R}^{n^{(i)} \times n^{(j)}}$  を用いて次式で表す。

$$A = \begin{bmatrix} A^{(1, 1)} & \dots & A^{(1, N)} \\ \vdots & \ddots & \vdots \\ A^{(N, 1)} & \dots & A^{(N, N)} \end{bmatrix} \quad (15)$$

ここで、 $\mathbf{A}^{(i,j)}$  は領域  $i$  が領域  $j$  から受ける作用を表す行列であり、領域  $i$  が領域  $j$  と隣接するとき非零なブロック行列成分を持つ。また、領域分割前の節点集合  $X$  に対応し、各節点上に値が定義されたベクトル  $\mathbf{x} \in \mathbb{R}^n$  は、領域  $i$  の内部節点集合  $X^{(i)}$  に対応するベクトル  $\mathbf{x}^{(i)} \in \mathbb{R}^{n^{(i)}}$  を用いて次式で表される。

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}^{(1)} \\ \vdots \\ \mathbf{x}^{(N)} \end{bmatrix} \quad (16)$$

#### (4) Local POD の定式化

Baiges [3] らによって提案された Local POD は、各分割領域で POD 基底を取得する手法である。分割領域毎に基底を取得するため、領域分割せず基底を取得した場合と比較して、より少ない計算コストで基底が取得される。また、ROM 解析は式 (9) より、基底によって張られる部分空間から解が決定されるため、対象問題を高精度に表現する基底を選定することが重要である。Local POD では、各分割領域で一定の基底本数を取得したとき、領域分割数に応じて解析領域全体での基底本数が増加する。そのため、「分割領域で局所的に取得された POD 基底全体」が張る部分空間は、「分割前の領域で大域的に取得された POD 基底」が張る部分空間よりも大きい、または等く、領域分割数に応じて高精度な解析を可能にする。

はじめに、各分割領域の内部節点  $n^{(i)}$  ( $i = 1, \dots, N$ ) に対応する Snapshot Matrix  $\mathbf{X}^{(i)} \in \mathbb{R}^{n^{(i)} \times n_{\text{snap}}}$  に対して、特異値分解を行う。

$$\mathbf{X}^{(i)} = \mathbf{U}^{(i)} \mathbf{\Sigma}^{(i)} \mathbf{W}^{(i)\top} \quad (17)$$

式 (4) より、左特異値行列  $\mathbf{U}^{(i)} \in \mathbb{R}^{n^{(i)} \times r}$  ( $r = \text{rank } \mathbf{X}^{(i)}$ ) のうち、はじめの  $k_{\text{POD}}^{(i)}$  本を選定し、POD 基底  $\mathbf{V}^{(i)} \in \mathbb{R}^{n^{(i)} \times k_{\text{POD}}^{(i)}}$  とする。各領域で独立に基底が取得されるため、全体領域での POD 基底  $\mathbf{V}$  は次式で与えられる。

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}^{(1)} & & 0 \\ & \ddots & \\ 0 & & \mathbf{V}^{(N)} \end{bmatrix} \quad (18)$$

次に、分割された領域に対応する係数ベクトル  $\tilde{\mathbf{u}}^{(i)} \in \mathbb{R}^{k_{\text{POD}}^{(i)}}$  からなる係数ベクトル  $\tilde{\mathbf{u}}$  を用いて、式 (9) で解ベクトルを近似する。

$$\tilde{\mathbf{u}} = \begin{bmatrix} \tilde{\mathbf{u}}^{(1)} \\ \vdots \\ \tilde{\mathbf{u}}^{(N)} \end{bmatrix} \quad (19)$$

Galerkin 法と合わせて支配方程式 (5) を離散化すると、縮約された方程式  $\mathbf{V}^\top \mathbf{A} \mathbf{V} \tilde{\mathbf{u}} = \mathbf{V}^\top \mathbf{f}$  (10) を得る。

このとき、縮約された係数行列  $\tilde{\mathbf{A}} = \mathbf{V}^\top \mathbf{A} \mathbf{V}$  は、次式で表される。

$$\tilde{\mathbf{A}} = \begin{bmatrix} \tilde{\mathbf{A}}^{(1,1)} & \dots & \tilde{\mathbf{A}}^{(1,N)} \\ \vdots & \ddots & \vdots \\ \tilde{\mathbf{A}}^{(N,1)} & \dots & \tilde{\mathbf{A}}^{(N,N)} \end{bmatrix} \quad (20)$$

ここで、縮約された行列の各ブロック行列  $\tilde{\mathbf{A}}^{(i,j)} \in \mathbb{R}^{k_{\text{POD}}^{(i)} \times k_{\text{POD}}^{(j)}}$  は、対応する節点の POD 基底との積で表され、以下のように計算される。

$$\tilde{\mathbf{A}}^{(i,j)} = \mathbf{V}^{(i)\top} \mathbf{A}^{(i,j)} \mathbf{V}^{(j)} \quad (21)$$

領域  $i$  と領域  $j$  が隣接するとき、行列ベクトル積  $\mathbf{V}^{(i)\top} \mathbf{A}^{(i,j)} \mathbf{V}^{(j)}$  が非零になるため、式 (20) は式 (15) と同様なブロック行列の非零構造を持つ。

縮約された右辺ベクトル  $\tilde{\mathbf{f}} = \mathbf{V}^\top \mathbf{f}$  は、対応する領域に属する POD 基底との積で計算される。

$$\tilde{\mathbf{f}} = \begin{bmatrix} \tilde{\mathbf{f}}^{(1)} \\ \vdots \\ \tilde{\mathbf{f}}^{(N)} \end{bmatrix} = \begin{bmatrix} \mathbf{V}^{(1)\top} \mathbf{f}^{(1)} \\ \vdots \\ \mathbf{V}^{(N)\top} \mathbf{f}^{(N)} \end{bmatrix} \quad (22)$$

### 3. Local POD に対する階層型並列化手法

本研究では、階層型グラフ分解に基づいた Local POD の並列計算アルゴリズムを提案する。

はじめに、本研究で扱うグラフについて述べる。グラフ  $G$  は、ノード集合  $X = \{x_1, \dots, x_n\}$  とノードを結ぶエッジ集合  $E = \{e_{a,b} \mid x_a, x_b \in X\}$  を用いて、 $G = (X, E)$  と定義されるデータ構造である。ここで、 $e_{a,b}$  は2つのノード  $x_a, x_b \in X$  が隣接していることを表す。また、 $n$  はノード数  $|X|$  である。本論文で定義されるグラフはセルフエッジ  $e_{a,a}$  を含むものとし、向きのない無向グラフとして扱う。

#### (1) 計算点グラフとメタグラフ

計算点グラフとは、計算点間の相互作用を表現するグラフ構造であり、計算点上に値を定義して得られる連立一次方程式の非零構造に対応する。例えば、有限要素メッシュの節点を計算点とする計算点グラフは、有限要素法により離散化して得られる連立一次方程式の非零構造に対応する。

グラフ構造を用いることで、ある隣接関係を持つ任意のノード集合に対して領域分割が定義できる。有限要素メッシュの節点を計算点とする計算点グラフを考えたとき、「有限要素メッシュに対する領域分割」が、「計算点グラフに対するグラフ分割」として一般化される。すなわち、ある隣接関係を持つ任意のノード集合  $X$  に対して、式 (12), (13) により領域分割し内部節点集合に対応するノード集合  $X^{(i)}$  ( $i = 1, \dots, N$ ) を得た後、式 (14) によってオーバーラップ領域  $X_{\text{ovl}}^{(i)}$  を含むノード集合  $\tilde{X}^{(i)}$  が定義される。また、ノード集合  $\tilde{X}^{(i)}$  から構成されるエッジ集合を  $\tilde{E}^{(i)}$  としたとき、部分グラフ  $G^{(i)} = (\tilde{X}^{(i)}, \tilde{E}^{(i)})$  が得られる。以降、計算点グラフに対するグラフ分割は、部分グラフ  $G^{(i)}$  が定義されるグラフ分割とする。

次に、上述する計算点グラフに対してグラフ分割を行ったとき、定義される領域の隣接関係について述べる。領域は分割されたグラフ  $G$  の内部領域  $X^{(i)}$  に対してノード  $\tilde{x}_i$  を対応させた関係を持ち、このようなデータ構造  $\tilde{G} = (\tilde{X}, \tilde{E})$  をメタグラフと定義する。メタグラフの概略図を図 1 の左側に示す。ここで、ノード集合  $\tilde{X} = \{\tilde{x}_1, \dots, \tilde{x}_N\}$  は分割された領域を表す集合であり、エッジ集合  $\tilde{E} = \{\tilde{e}_{i,j} \mid \tilde{x}_i, \tilde{x}_j \in \tilde{X}\}$  は領域間の隣接関係を

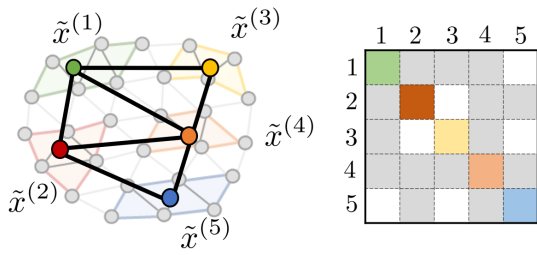


図-1 Relationships between metagraph and block matrix.

表す. このようなグラフは, 領域間を跨ぐエッジを判別することで構築され, 次式が成り立つとき領域  $i$  は領域  $j$  と隣接するとみなし, メタグラフのエッジ  $\tilde{e}_{i,j}$  で表現する.

$$X^{(i)} \cap X_{\text{ovl}}^{(j)} \neq \emptyset \quad (23)$$

続いて, 計算点グラフおよびメタグラフに基づいて, グラフ分割前後のベクトル, 行列の対応関係を明示する. これらの対応関係は, 3.2 節にて述べるグラフ構造に基づいた並列計算の定式化に利用する. ノード集合  $X$  の要素  $x_a$  にグローバル番号  $i_G$  を紐付け, 部分グラフ  $G^{(i)}$  の要素  $x_a$  に領域  $i$  のローカル番号  $i_L$  を紐付ける. 分割前後の係数行列  $A$  と  $A^{(i,j)}$  およびベクトル  $\mathbf{x}$  と  $\mathbf{x}^{(i)}$  の関係は, グローバル番号  $i_G$  上に定義されたベクトル要素を領域  $i$  のローカル番号  $i_L$  へ対応させる行列  $\mathbf{P}^{(i)} \in \mathbb{R}^{|X| \times |X^{(i)}|}$  を用いて, 次式で表される.

$$\mathbf{A} = \sum_{i=1}^N \sum_{j=1}^N \mathbf{P}^{(i)} \mathbf{A}^{(i,j)} \mathbf{P}^{(j)T} \quad (24)$$

$$\mathbf{x} = \sum_{i=1}^N \mathbf{P}^{(i)} \mathbf{x}^{(i)} \quad (25)$$

ここで, 領域  $i$  と  $j$  が隣接しているとき, 領域  $i$  と  $j$  の相互作用を表すブロック行列  $\mathbf{A}^{(i,j)}$  は非零となる. メタグラフは領域間の隣接関係を表すため, 図 1 に示すように行列  $\mathbf{A}$  のブロック行列  $\mathbf{A}^{(i,j)}$  に関する非零構造が一意に定まる. 領域  $i$  のローカル番号  $i_L$  をグローバル番号  $i_G$  に変換する関数  $i_G = \text{id}_x(i, i_L)$  を考えると,  $\mathbf{P}^{(i)}$  の行列成分は次式となる.

$$P_{i_G i_L}^{(i)} = \begin{cases} 1, & i_G = \text{id}_x(i, i_L) \\ 0, & \text{otherwise} \end{cases} \quad (26)$$

各計算点が担当する自由度が  $d$  の場合, 非零成分には  $\mathbf{I} \in \mathbb{R}^{d \times d}$  を持つブロック行列構造となる.

## (2) グラフ構造に基づいた並列計算

本研究では, 並列計算機の利用を前提として, 反復法を採用する. 反復法の並列計算は, そのアルゴリズムから, ベクトル和, ベクトル内積, 行列ベクトル積の並列計算を実現すればよい.

式 (24), (25) について,  $i$  番目の分割領域を表すインデックス  $i$  に並列プロセスを対応させ, 並列計算に必要な演算を定義する. 例として, 行列ベクトル積  $\mathbf{y} = \mathbf{A}\mathbf{x}$  について述べる. 領域間の連結情報はメタグラフ  $\tilde{G}$  によって記述されたため, 領域  $i$  から領域  $j$  へ与える作用

を表す行列  $\mathbf{A}^{(i,j)}$  は, メタグラフ  $\tilde{G}$  の隣接関係を参照することで定まる.  $\text{nbhd}(x_i, G)$  を, グラフ  $G = (X, E)$  に関してノード  $x_i \in X$  と隣接するノードをすべて返す関数としたとき, 次式で表される.

$$\begin{aligned} \mathbf{y} &= \sum_{i=1}^N \mathbf{P}^{(i)} \mathbf{y}^{(i)} \\ &= \sum_{i=1}^N \mathbf{P}^{(i)} \left\{ \mathbf{A}^{(i,i)} \mathbf{x}^{(i)} + \sum_{\tilde{x}_j \in \text{nbhd}(\tilde{x}_i, \tilde{G})} \mathbf{A}^{(i,j)} \mathbf{x}^{(j)} \right\} \end{aligned} \quad (27)$$

式 (27) より, 内部領域  $X^{(i)}$  に関する計算結果  $\mathbf{y}^{(i)}$  のみ更新されるため, 隣接領域に属するベクトル  $\mathbf{x}^{(j)}$  を MPI 通信によって取得する必要がある.

## (3) Local POD に対する階層型並列計算アルゴリズム

本研究における並列化の対象は Local POD であり, 計算過程は低次元化された連立一次方程式 (10) の構築およびその連立一次方程式の求解 (線形ソルバ) に分けられる. 提案手法でははじめに, 線形ソルバを並列計算することを目的として, 階層的なグラフ構造を導入する. 次に, 低次元化された連立一次方程式 (10) の構築に対して並列計算を行うため, 階層的なグラフ構造を参照しメッシュ構造を再構築する.

提案する階層的なグラフ構造は, 「有限要素メッシュの節点を計算点とする計算点グラフに対するグラフ分割」と, 「POD 計算領域を計算点とする計算点グラフに対するグラフ分割」の 2 階層のグラフ分割から構成されている. なお POD 計算領域は, 分割された領域のうち POD を実行する 1 領域であり, 並列プロセスを割り当てる領域とは独立に決定される. 図 2 の上部に示すように, 節点を計算点とする計算点グラフより得られるメタグラフのノードに対して POD 計算領域を割り当て, POD 計算領域を計算点とする計算点グラフより得られるメタグラフのノードに対して並列プロセスを割り当てることで, 領域分割型並列計算を行う. ここで, 節点を計算点とする計算点グラフは, 有限要素法により離散化して得られる係数行列の非零構造に対応する. POD 計算領域を計算点とする計算点グラフは, Local POD の連立一次方程式の非零構造 (20) を表現する. このとき, 節点を計算点とした計算点グラフに対するグラフ分割で得られたメタグラフのノードに POD 計算領域を代表させ, メタグラフを POD 計算領域を計算点とする新たな計算点グラフとして定める.

上述の階層的なグラフ構造に基づき, Local POD の並列計算を実現する. 線形ソルバに対する並列計算は, POD 計算領域を計算点とする計算点グラフに対してグラフ分割を適用して得られる分割グラフおよびメタグラフに基づき, 3.2 節に沿って並列計算が実行される. ここで, 各計算点の自由度は各 POD 計算領域において取得される POD 基底本数である. 式 (10) で表される低次元化された連立一次方程式の構築に対する並列計算は, 並列プロセスが担当する節点に基づいて行う. 並列プロセスに割り当てる領域は, 図 2 の下部に示すように, 各ノードに POD 計算領域が紐づく計算点グラフ

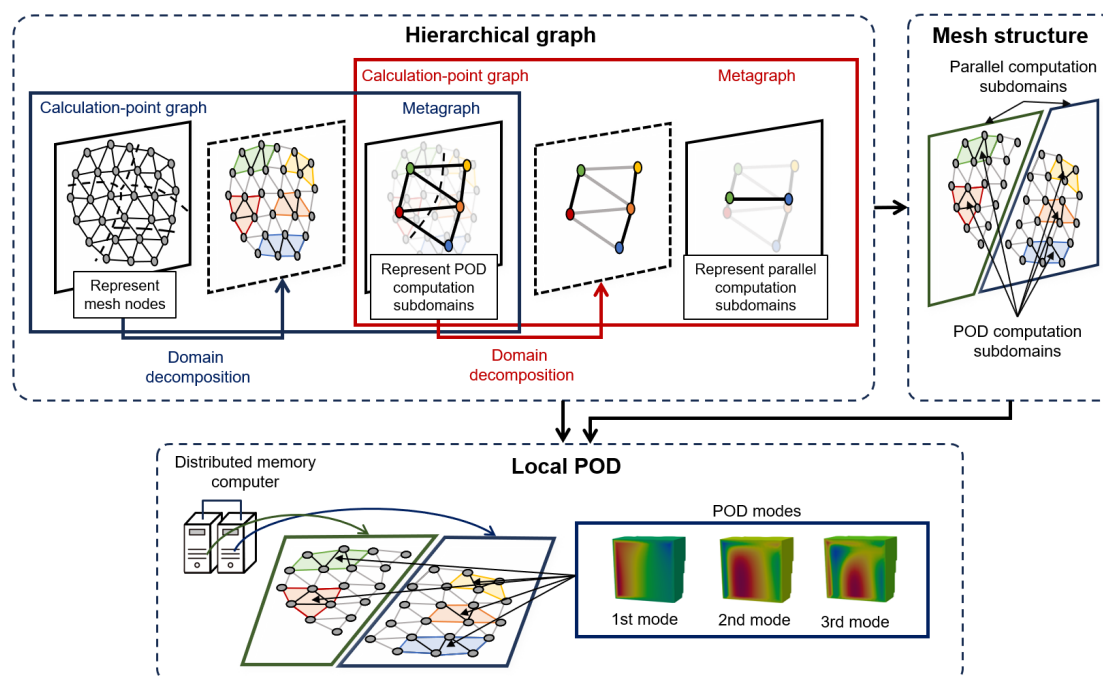


図-2 Overview of proposed method.

およびその分割グラフを参照し、複数の POD 計算領域をグルーピングすることで構築する。

#### 4. 数値実験

##### (1) 精度検証

提案手法について、計算精度の観点から検証を行った。

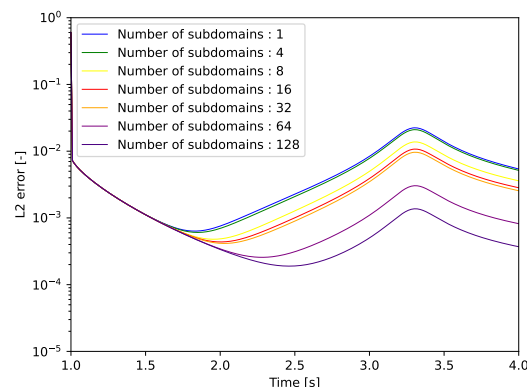
支配方程式として非定常拡散方程式 (5) を用いた。拡散係数  $k$  は 1, タイムステップ  $\Delta t$  は 0.001, ソース項  $f$  は 0 とする。境界条件として Neumann 境界条件は与えず、境界  $\Gamma$  全面に Dirichlet 境界条件 (28) を課した。

$$u = \sin(0.25x) \sin(0.5y) \sin z \sin t \quad \text{on } \Gamma \quad (28)$$

解析領域  $\Omega$  は 3 次元立方体領域  $[0, 5]^3$  とし、要素がすべて立方体となるようにメッシュを構築する。このとき、自由度は 132,651 とした。基底関数には 1 次の Lagrange 補間関数を用い、Gauss 求積によって要素毎の積分を計算した。行列計算には Monolithic linear solver based on domain decomposition (MONOLIS) [8] を使用した。連立一次方程式の求解には CG (conjugate gradient) 法を用い、収束判定閾値は  $1.0 \times 10^{-9}$ , 最大反復回数を 10,000 とし、前処理には対角スケーリング前処理を用いた。誤差評価には、 $L^2$  ノルムによる相対誤差式 (29) を用いて、ROM による数値解  $u_{\text{ROM}}$  と、有限要素法による数値解  $u_{\text{FEM}}$  との比較を行った。各領域で取得する POD 基底本数は 10 本で一定とし、領域分割数は 1, 4, 8, 16, 32, 64, 128 とした。

$$\varepsilon_{\text{ROM}} = \frac{\sqrt{\int_{\Omega} |u_{\text{ROM}} - u_{\text{FEM}}|^2 d\Omega}}{\sqrt{\int_{\Omega} |u_{\text{FEM}}|^2 d\Omega}} \quad (29)$$

有限要素法により 100 タイムステップ (1s) の解析を行い、101 タイムステップから 400 タイムステップまで

図-3 Relative  $L^2$  error norm of Local POD in each number of subdomains.

Local POD を用いて解析したとき、相対誤差 (29) の時刻歴を図 3 に示す。図 3 より、領域分割数が増加するほど計算精度が上がる事が確認され、実装の妥当性が示された。

##### (2) 計算効率の検証

提案手法である階層型並列計算アルゴリズムの並列計算性能の評価を行った。本検証で使用した分散メモリ型計算機は大阪大学のスーパーコンピュータシステム SQUID [9] である。計算機諸元を表 1 に記す。誤差評価式、その他の解析条件は 4.1 節と同一とし、自由度を 10,077,696 とした。並列計算領域内の POD 計算領域数を 2, 4, 8, 16 とした 4 ケースに対し、並列数 (= 並列計算領域数) を 32, 64, 128, 256, 512 と変化させた場合の並列計算性能評価を行った。このとき、メモリバンド幅律速の影響を考慮し、1 ノードあたりの最大ブ



表-1 Specification of SQUID super computer

Component	Configuration of nodes
CPU	Intel® Xeon® Platinum 8368 (Icelake / 2.4 GHz 38 cores) × 2
Memory	256 GB
Peak performance	8,871 PFLOPS
The number of nodes	1,520 nodes

ロセス数は16とした。なお、「各並列計算領域内のPOD計算領域数」は設定上のPOD計算領域数 $N_{\text{POD}}$ /並列計算領域数 $N_{\text{par}}$ の値であり、実際にMETISで分割された値と同じになるとは限らない。

並列計算性能の評価手法として、ROM効率 (ROM efficiency)  $R_{N_{\text{par}}}$  を指標として用いた。ROM効率はROM解析の時間が通常の有限要素解析の時間よりも $R_{N_{\text{par}}}$ 倍高速かを表し、次式で表される。

$$R_{N_{\text{par}}} = \frac{T_{N_{\text{par}}}^{\text{FEM}}}{T_{N_{\text{par}}}^{\text{ROM}}} \quad (30)$$

ここで、 $T_{N_{\text{par}}}^{\text{FEM}}$ 、 $T_{N_{\text{par}}}^{\text{ROM}}$  をそれぞれ、並列数 $N_{\text{par}}$ での有限要素法を用いた計算時間、縮約モデルを用いた計算時間とする。提案手法のメッシュ構造は複数のPOD計算領域をグルーピングして構築するため、有限要素解析の計算効率の観点から、最適な領域分割とは異なる。そこで、比較対象の有限要素解析は、節点を計算点とした計算点グラフをMETISにより $N_{\text{par}}$ 個に分割し、それらに並列プロセスを割当て並列計算を行った。

101 タイムステップから200 タイムステップの計100 タイムステップを測定対象とし、各タイムステップにおける全体計算時間の平均に対するROM効率の測定結果を図4に示す。図4より、いずれの並列プロセス数においてもROM効率が維持されていることから、提案手法が大規模問題の並列計算に有効であることが示された。また、各並列計算領域内のPOD計算領域数が2の場合は、ROM効率が低下している。これは、メッシュを再構成する最小単位がPOD計算領域であるため、並列プロセスに割り当てる領域の偏りが大きく、並列計算の同期時間が長いことに起因すると考える。一方で、その他のケースではばらつきがあるものの、高いROM効率を示している。提案手法は、並列プロセスに割り当てるPOD計算領域が可変であり、対象問題に応じて最適な領域数を提案可能である。今後は、各POD計算領域での基底本数を可変にした場合において、適切な分割領域を決定する負荷分散を検討内容とする。

## 5. 結論

本研究では、分散メモリ型並列計算環境における階層型グラフ分解に基づいたLocal POD計算アルゴリズムを提案した。提案手法により、Local PODの並列計算が実現され、POD計算領域と並列計算領域を独立に定義可能となった。数値実験の結果より、領域分割数の増加に伴い計算精度が向上することが確認された。また並列計算の結果より、大規模問題においても高い並列計算性能を持つことが示された。

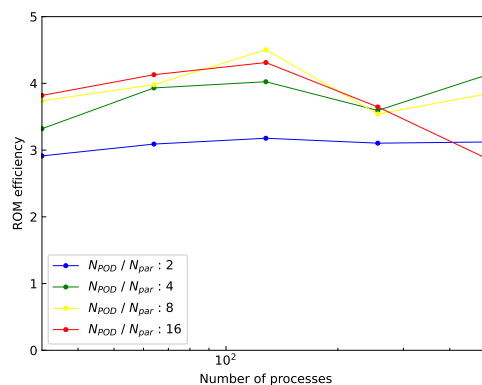


図-4 ROM efficiency of total computation time in each parallel number.

謝辞: 本研究は、JST 創発的研究支援事業 JPMJFR215S および JSPS 科研費 23K24857, 23H00475 の支援を受けたものである。ここに記して謝意を表する。

## 参考文献

- [1] K. Lu, Y. Jin, Y. Chen, Y. Yang, L. Hou, Z. Zhang, Z. Li, and C. Fu. Review for order reduction based on proper orthogonal decomposition and outlooks of applications in mechanical systems. *Mechanical Systems and Signal Processing*, 123:264–297, 2019.
- [2] N. Discacciati and J. Hesthaven. Localized model order reduction and domain decomposition methods for coupled heterogeneous systems. *International Journal for Numerical Methods in Engineering*, 124(18):3964–3996, 2023.
- [3] J. Baiges, R. Codina, and S. Idelsohn. A domain decomposition strategy for reduced order models. Application to the incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 267:23–42, 2013.
- [4] A. Castro, P. Kuberry, I. Tezaur, and P. Bochev. A Novel Partitioned Approach for Reduced Order Model-Finite Element Model (ROM-FEM) and ROM-ROM Coupling. *Earth and Space* 2022, 2023.
- [5] C. Cao, C. Nie, S. Pan, J. Cai, and K. Qu. A constrained reduced-order method for fast prediction of steady hypersonic flows. *Aerospace Science and Technology*, 91:679–690, 2019.
- [6] L. Sirovich. Turbulence and the dynamics of coherent structures. I - Coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987.
- [7] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *Siam Journal on Scientific Computing*, 20(1):359–392, 1999.
- [8] monolis, monolithic linear solver based on domain decomposition. 2024 年 3 月 22 日閲覧.
- [9] Cybermedia center blog archive squid, osaka university. 2024 年 3 月 22 日閲覧.