汎用生体シミュレータ insilicoSim における 生体モデル内の依存関係に着目したスケジューリング

Focusing on dependencies in biophysical models to improve scheduling in insilicoSim

松井　龍[†]　奥山倫弘[†]　置田真生[†]
安部武志[††]　浅井義之[††]
野村泰伸[†††]　萩原兼一[†]

Ryu Matsui　Tomohiro Okuyama　Masao Okita　Takeshi Abe
Yoshiyuki Asai　Taishin Nomura　Kenichi Hagihara

1. はじめに

近年，生体学を中心とした数理モデル作成の研究が世界中で注目されている.[1)~3)] これらの研究では，生体機能を表現するモデルをコンピュータ上でシミュレーションすることにより，複雑な生体機能のメカニズムの理解を目指している．生体機能は分子，細胞，器官，臓器などの多階層に渡る生体部品の機能を組み合わせることで引き起こされる組織的作用である．

多階層に渡る生体機能を記述する言語として ISML[1)] がある．ISML は，一部の生体機能の記述に特化せず，多様な生体機能のモデルを汎用的に扱うことができる．ISML では，生体機能は複数種の常微分方程式（Ordinary Differential Equations; ODEs）およびそれらが参照する数式で記述される．

*insilico*Sim[4)] は ISML で記述された生体モデル (ISML モデル) を入力とする汎用生体シミュレータである．ISML モデル中の ODEs を連立させた初期値問題を数値的に解くことにより，時間発展型のシミュレーションを行う．大規模な ISML モデルのシミュレーションを高速化するために，*insilico*Sim は MPI を用いた並列実行の機能を備える．並列化の手順は以下の通りである．まず ISML モデルを汎用的なタスクグラフとして解釈する．次に，得られたグラフを分割して，各プロセスに割り当てる．最後にプロセスごとにレベルスケジューリングに基づいて計算の実行順序を決定し，並列に実行する．本論文では，並列版の *insilico*Sim におけるシミュレーションの高速化を対象とする．

ISML モデルを対象としたシミュレーションの高速化の問題点は，ISML モデルの特徴によって並列実行性能が異なることである．*insilico*Sim では，1 つの生体部品に対する接続数が少ない ISML モデル（集中度の低い ISML モデル）について，並列化において線形的な速度向上が達成できている．これに対して，1 つの生体部品に対する接続数が多い ISML モデル（集中度の高い ISML モデル）では，速度向上率が低い．その原因は主に 2 つある．第一に，現在のスケジューリング手法では，スケジュールの同一レベルにおけるプロセス間の負荷が偏り，実行時に遊休時間の長いプロセスが発生するためである．第二に通信データの一括化に伴うオーバヘッドが大きいためである．

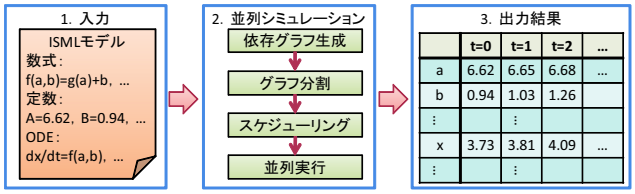


図 1　*insilico*Sim の概要

本研究の目的は，集中度の高い ISML モデルについて，*insilico*Sim による並列実行性能を向上させることである．具体的には，各プロセスの遊休時間を削減するために，クリティカルパスを重視したグラフ分割を導入する．そして実行時の通信データ一括化のオーバヘッドを改善するために，効率的なデータ配置を適用する．

以下，2 章では *insilico*Sim について説明する．次に 3 章で具体的な問題の原因について説明し，4 章で提案手法について詳細に説明する．5 章で提案手法の性能について考察し，最後に 6 章でまとめと今後の課題を述べる．

2. *insilico*Sim

並列版 *insilico*Sim の概要を図 1 に示す．

2.1 入　　力

*insilico*Sim における入力である ISML モデルについて説明する．ISML モデルは，階層的に生体機能を構成するために，細胞，細胞膜，臓器などを表すモジュールを用いて記述される．モジュールは他のモジュールをカプセル化できる．モジュールは他のモジュールとネットワークを構成する場合があり，その関係はモジュール間を有向辺で接続することで定義できる．これらのモジュールのカプセル化および有向辺によるネットワーク構造が，シミュレーションにおけるモジュール間のデー

† 大阪大学大学院情報科学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Information Science and Technology, Osaka University
†† 沖縄科学技術研究基盤整備機構 オープンバイオロジーユニット
Open Biology Unit, Okinawa Institute of Science and Technology
††† 大阪大学大学院基礎工学研究科機能創成専攻
Department of Mechanical Science and Bioengineering, Graduate School of Engineering Science, Osaka University

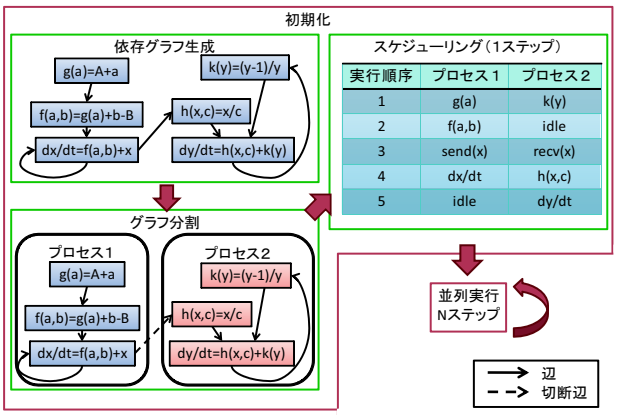


図 2　*insilico*Sim における並列シミュレーションの流れ

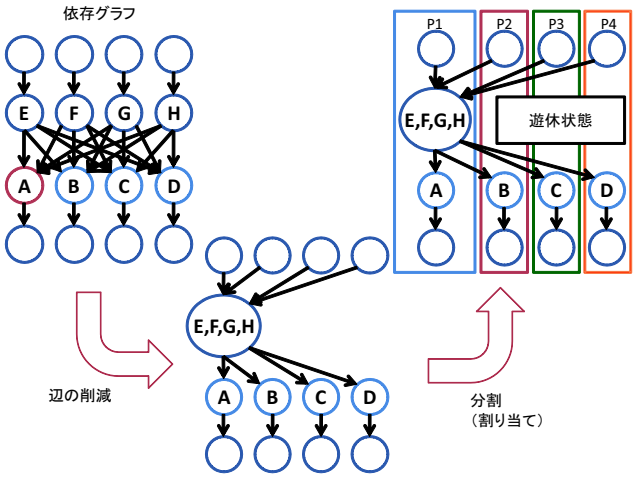


図 3　遊休時間の増加

タ依存を決定する．

モジュールは標準的な関数を表現する数式，自然法則や生体固有の値を表現する定数，および状態を表現する ODEs から成る．モジュール内の式および ODEs の間にもデータ依存が存在する．

ここで，ある式 f が他の式 g にデータ依存が存在するとは，f を計算するために g の計算結果が必要になることを表す．本研究では，シミュレーション中にデータ依存が動的に変化しない ISML モデルを対象とする．

2.2　並列シミュレーション

*insilico*Sim における並列シミュレーションの流れを図 2 に示す．並列シミュレーションは大きく分けて初期化と並列実行の 2 段階である．初期化の実行は 1 回であることに対して，並列実行は N 回繰り返す．なお，N はユーザが指定するステップ数である．

1 つ目の初期化はさらに依存グラフ生成，グラフ分割，スケジューリングの 3 つに分かれる．

まず依存グラフ生成では，ISML モデル内のモジュールが有するデータ依存から依存グラフを生成する．依存グラフ $G = (V, E)$ は式 $f, g \in V$ を頂点，式間の依存関係 $< f, g > \in E$ を辺とする有向グラフである．頂点は ODE または数式を表す．辺 $< f, g >$ は式 g が式 f に依存することを示す．

次にグラフ分割では，依存グラフを分割し，分割したグラフを各プロセスに割り当てる．一般に並列処理では通信処理がボトルネックとなる．そこで *insilico*Sim では，負荷分散および通信量の削減を重視した分割手法 ParMETIS[5] を用いる．この分割手法は，負荷分散を維持しつつ，分断される辺（切断辺）数を最小化するために，multilevel k-way partitioning algorithm[6] に基づいた分割アルゴリズムを使用する．このアルゴリズムでは，重み付き無向グラフ G と分割数 p を入力すると，頂点集合 V の分割 $(V_1, V_2, ..., V_p)$ を出力する．

最後にスケジューリングでは，プロセス毎に割り当てられた依存グラフから式の計算順序を決定する．このスケジューリングには，依存グラフのトポロジカルソートを用いる．

2 つ目の並列実行では，生成したスケジュールの計算順序に従って，ODEs および数式を順に計算する．1 ステップの計算後，時刻を進めて，N ステップ回の計算を繰り返す．数値解法には，オイラー法あるいはルンゲ＝クッタ法を用いる．

2.3　出 力 結 果

シミュレータの出力について説明する．ステップ毎にユーザが指定する任意の変数の値をテキスト形式で出力する．

すべてのステップ分の計算結果を出力をする場合，*insilico*Sim では出力処理時間がボトルネックとなる．例えば，中規模の ISML モデル（出力可能な変数の数が 400）では，総実行時間の 58%を出力処理が占める．ただし，実用上は数時刻分に 1 回の出力で十分である．本論文では，計算処理がボトルネックとなる状況を想定し，出力を省略した場合，つまり総実行時間の 90%以上をシミュレーション時間が占める場合を対象とする．

3.　問題の分析

*insilico*Sim を用いて，集中度が高い ISML モデルをシミュレーションする場合，次の 2 つの問題が発生する．

P1　遊休時間の増加

P2　通信データ一括化のオーバヘッド

本章では，P1，P2 の原因について詳細に説明する．

3.1　P1：遊休時間の増加

遊休時間の増加の原因は ParMETIS によるグラフ分割にある．ParMETIS は，次数の高い頂点に依存する複数の頂点を，1 つの頂点に統合する．これにより辺数を削減し，グラフ分割時の切断辺数が削減できる．ISML モデル内に 1 対全の依存関係がある場合，直前のレベルの全頂点が 1 つの頂点に統合される．プロセスへの割り当ては統合した頂点単位であるため，あるレベルの頂点すべてが 1 プロセスに割り当てられる．このとき残りの全プロセスが遊休状態となる．

グラフ分割の例を図 3 に示す．図 3 では，頂点の大きさが頂点の重みを表しており，頂点 A, B, C, D, E, F, G, H の重みは等しい．また辺数は 24 である．

図 3 の依存グラフにおいて，次数が高い頂点は A, B, C, D, E, F, G, H であり，次数は 5 である．次数が等しい頂点が複数ある場合，頂点の重みを比較し，大きな頂点を選択する．し

かし図 3 では，比較する頂点の重みはすべて等しいので，頂点 A が選択されると仮定する．この場合，頂点 A と依存関係をもつ E, F, G, H を 1 つの頂点に統合することにより，辺数は 12 に削減される．そして，辺数削減後の依存グラフに対して，切断辺数の削減を重視して，グラフを分割し，プロセスに割り当てる．その結果，同一レベルにある頂点の E, F, G, H をプロセス 1 が計算する．したがってプロセス 1 の計算の間，プロセス 2,3,4 の各プロセスは遊休状態となる．

3.2 P2：通信データ一括化のオーバヘッド

通信データ一括化のオーバヘッドの発生は，MPI の通信特性と *insilico*Sim の通信方式の 2 つが原因である．

第一に MPI の通信特性上，メモリ上に不規則に配置された通信データを複数回に分けて通信するより，一括化して通信するほうが効率がよい．

第二に *insilico*Sim の通信では，通信バッファに変数値をコピーする．依存グラフを用いた分割およびスケジューリングでは，どの変数をどのプロセスに通信するかはスケジュール作成まで決定できない．このため，通信データはメモリ上に不規則に配置される．通信の一括化のために，不規則に配置された通信データを通信バッファに集約してから通信する必要がある．

ここで，計算に対して通信量の少ない ISML モデルではコピーのオーバヘッドは無視できるほど小さい．一方，本論文で対象とする集中度が高い ISML モデルでは通信量が多く，全体のうちコピーのオーバヘッドの占める割合が大きい．

4. 提 案 手 法

本章では前章で挙げた P1，P2 を解決するために，それぞれ依存グラフの各レベル内の負荷分散を優先したスケジューリング，メモリ上データの配置変更を提案する．

4.1 レベル内の負荷分散を優先したグラフ分割

P1 の問題を改善するために，依存グラフの各レベル内の負荷を分散して，各プロセスの遊休時間を削減する必要がある．そこで我々は，依存グラフにおけるクリティカルパスの長さ（CP 長）の短縮に着目した．ここで，クリティカルパスは依存グラフ内の開始頂点から終点頂点までのパスの中で，パスを構成する頂点の重みと辺の重みの合計が最大となるパスである．CP 長は，クリティカルパスを構成する頂点の重みと辺の重みの合計値である．

依存グラフ分割において CP 長の短縮は，1 ステップの所要時間の短縮に寄与する．1 ステップ内の総計算時間は一定であるため，1 ステップの所要時間の短縮は各プロセスの遊休時間の削減につながる．

本論文では，CP 長を短縮する手法として MCP アルゴリズム[7] を実装した．このアルゴリズムでは，依存グラフの各頂点に対して，As Late As Possible（ALAP） time に基づいて優先順位を付ける．ALAP time はクリティカルパスに影響しない範囲，つまり CP 長が増加しない範囲で，頂点の計算を最も遅く開始できる時刻である．

分割アルゴリズムの概要を示す．従来のグラフ分割と同様

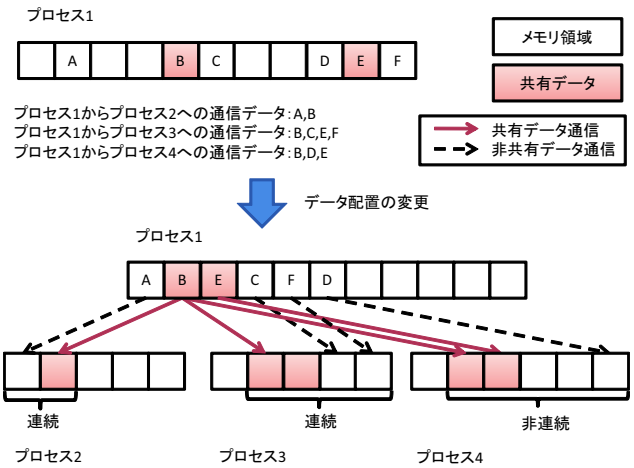


図 4 共有データが存在する場合の通信とデータ配置

に，重み付き有向グラフ G と分割数 p を入力し，頂点集合 V の分割 $(V_1, V_2, ..., V_p)$ およびスケジュールを出力する．ここで，頂点 $n \in V$ の計算時間は $w(n)$ とする．

(1) 各プロセスの遊休時間を保持するリストを $I_k = [i_0](1 \leq k \leq p)$ とする．$i_0.start = 0, i_0.end = \infty$ と定める．

(2) すべての頂点 $n \in V$ について ALAP time $alap(n)$ を計算する．

(3) 各頂点の ALAP time に基づいて頂点を昇順にソートする．

(4) (3) で得られたリストが空になるまで以下を繰り返す．

(a) (3) で得られたリストの先頭要素 n について，それぞれのプロセス P_k で以下を実行する．

(i) リスト I_k を先頭から順に走査し，$i.end - max(i.start, alap(n)) \geq w(n)$ を満たす $i \in I_k$ を選択する．

(ii) $T_{first} = max(i.start, alap(n))$ とする．T_{first} はプロセス P_k 内で，n を最も早く実行できる時間である．

(b) 全プロセスの中で最小の T_{first} を保持するプロセス P_m に頂点 n を割り当てる．

(c) n をリストから削除する．

(d) I_m において，$i.end = T_{first}$ と更新する．新たに i' を作成し，$i'.start = T_{first} + w(n), i'.end = i.end$ として I_m へ挿入する．この操作により，頂点割り当て後のプロセスの遊休時間を更新する．

4.2 メモリ上のデータの配置変更

P2 の問題点を改善するために，メモリコピーを削減する必要がある．そこで我々は，本研究で扱うシミュレーションが反復型シミュレーションであることに着目した．ステップが変化しても毎回同じ計算・通信パターンを繰り返すため，通信バッファへのコピー時に参照するメモリ領域も毎回同じである．ゆえに，通信データが連続領域となるよう初期化時に予め配置を変更すれば，通信バッファを介さずに通信データを一括通信できる．

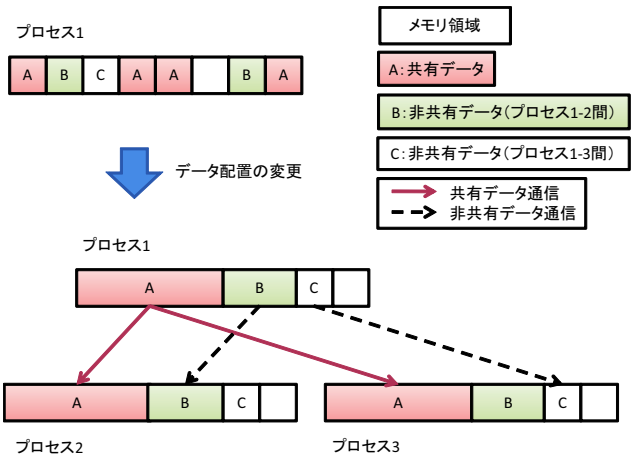


図 5 共有データおよび非共有データの通信とデータ配置

通信データの配置を変更する上での問題は，全ての通信データを連続領域に配置できないことである．その理由は，複数のプロセスに通信するべきデータ（共有データ）が存在するためである．例えばあるプロセスが共有データを保持する場合を図 4 に示す．図 4 中のブロックはメモリ領域を表し，ブロック内の A,B,C,D,E は変数を表す．プロセス 1 が A,B,C,D,E,F の値を保持する．B,E は共有データであり，プロセス 2 には A,B，プロセス 3 には B,C,E,F，プロセス 4 には B,D,E の値を通信する．このとき，プロセス 1-2 間とプロセス 1-3 間で通信データが連続領域となるように配置を変更すると，プロセス 1-4 間では通信データの配置は非連続となる．

この問題を解決するために，通信データを共有データと非共有データに分けて通信する．共有データと非共有データの配置変更について，図 5 に示す．図 5 が示すように共有データは，連続領域となるように並び替え，すべてのプロセスに通信する．これに対して，非共有データは，通信するプロセス毎で連続領域となるように並び替え，1 つのプロセスに対して通信する．共有データ，非共有データの通信は，MPI のノンブロッキング通信を用いて実装する．

共有データ通信では，スケジュールの最初に全共有データを全プロセスに通信する．*insilico*Sim では，冗長計算を用いて数式の値を送信する通信を消去できる．[8] ゆえに，共有されうるデータは ODE に限られる．また数値計算上，数式の計算では，1 ステップ前の ODE の値を用いる．このため，ステップの開始時に，必要な共有データの値はすべて確定している．そこで，通信回数の削減と通信データの集約のために，スケジュールの最初に共有データを通信できる．

なお，本来は全共有データが全プロセスに必要ではない．例えば図 4 のように，共有データ B はすべてのプロセスで共有される一方，共有データ E はプロセス 3，プロセス 4 で共有される．ただし，提案手法では，連続領域上のデータを一括送信することを優先し，全プロセスに全共有データを送信する方法を採用する．

データの配置を変更する場合，全プロセスへの通信を追加するため，プロセス数を p とすると，総通信回数が $p(p-1)$ 回増加する．また全共有データの全プロセスへの通信により，本来必要でない通信データも通信するため，通信量が増加する．

ここで，通信データの配置変更が有効となるのは，1 対全の依存関係を有する ISML モデルのように，集中度が高い ISML モデルに適用した場合である．1 対全の依存関係を有する ISML モデルのシミュレーションでは，特定の ODE を全プロセスで共有する通信が元々存在する．このため，手法の適用によって増加する通信量は小さい．またこの ISML モデルは通信量が多いため，共有データの通信による通信時間の増加と比較して一括化オーバヘッドの削減の効果が大きい．

なお，各プロセスに対する通信回数を高々1 回にするため，共有データを複製する手法が考えられる．この場合，共有データの計算時に複製した共有データすべてに値を書き込む必要がある．その結果，共有データの計算時に，毎回書き込みオーバヘッドが発生する．これに対して，提案手法では，あるプロセス上で通信相手に関係なくデータ配置が 1 通りとなり，共有データを複製しないため，書き込みオーバヘッドが発生しない．

表 1 性能評価に用いる ISML モデル

ISML モデル	Wang[9]	Luo-Rudy[10]
数式の数	1200	3298
ODE の数	400	800
接続形態	全対全	直線状
集中度 (最大)	99	2

5. 評価実験

本章では以下の 3 点について提案手法を評価した．

- 速度向上率
- 遊休時間の削減
- 追伸データ一括化オーバヘッドの削減

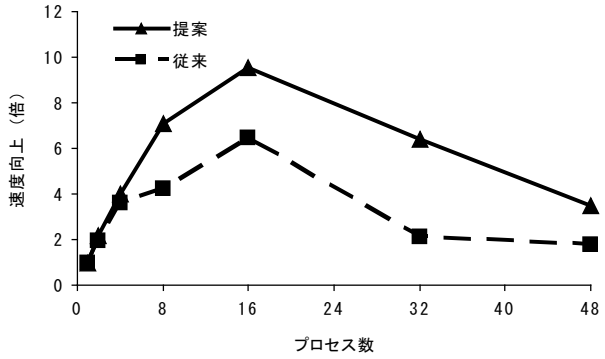
実験環境の CPU は 12-Core AMD Opteron 6174 2.2 GHz × 4，メモリは 256GB である．OS は Ubuntu 11.04，開発環境は gcc 4.5.2，OpenMPI 1.4.3，ParMETIS 3.1 である．

表 1 に実験に用いた ISML モデルを示す．Wang は文献[9] の神経細胞モデルに基づいた ISML モデルである．また Luo-Rudy は，文献[10] の心筋細胞モデルを直線状に 100 個接続した ISML モデルである．

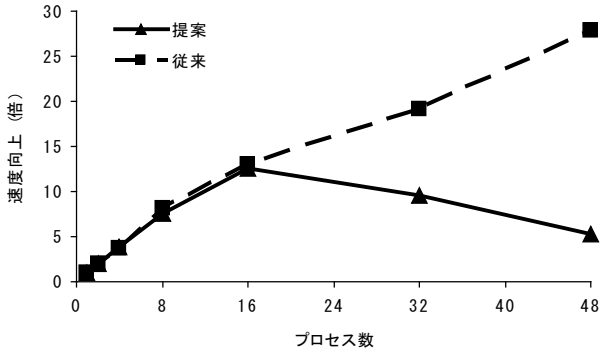
5.1 速度向上率

図 6 に，表 1 の ISML モデルをそれぞれ 100,000 ステップ（モデル内時間 1 秒，刻み幅 0.01 ミリ秒）実行するときの速度向上率を示す．また図 7 は，そのときの実行時間の内訳を示す．ここで通信はデータの転送，通信待ち，一括化オーバヘッドの時間から成る．

まず図 6(a) から，Wang モデルでは，すべてのプロセス数において，従来手法よりも高い速度向上を達成する．また 16 プロセス時，速度向上は最大 9.5 倍となる．一方図 6(b) より Luo-Rudy モデルでは，プロセス数が 16 を超えると速度向上率が低下する．また提案手法は 16 プロセス時，速度向上は最大 12 倍となる．

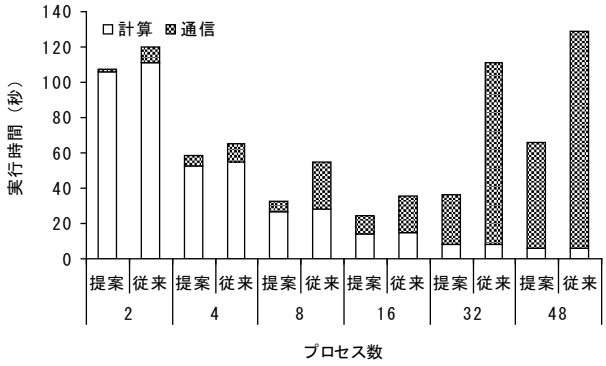


(a) Wang

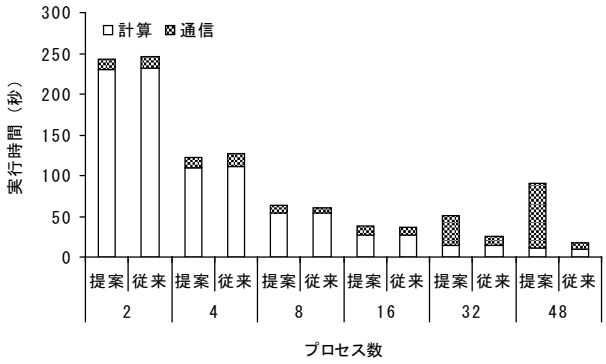


(b) Luo-Rudy

図 6 速度向上率



(a) Wang



(b) Luo-Rudy

図 7 実行時間内訳

次に図 7(a)，7(b) より，Wang モデルにおける提案手法の速度向上の原因は，通信時間の削減であり，Luo-Rudy モデルにおける提案手法の速度低下の原因は通信時間の増加である．

5.2 プロセスの遊休時間

図 8 に，通信時間の内訳を示す．図 8 は，従来手法のグラフ分割手法のみを変更した場合の計測データである．ここで，一括化は通信データ一括化のオーバヘッドである．MPI を用いた並列実行では，通信により同期が発生する．この同期までに要する計算量がプロセス間で不均衡であるならば，計算量の小さいプロセスは通信待ち，つまり遊休状態となる．したがって通信待ち時間を評価することにより，プロセスの遊休時間を評価できる．

図 8(a) から，Wang モデルでは，プロセス数が 2,8,32,48 のとき，提案手法は従来手法と比較して通信待ち時間を削減する．32 プロセス時，最大 84%まで通信待ち時間を削減する．提案手法では，プロセス数が 32 のとき，従来手法と比較して転送時間が 9.5 倍となる．

図が示すように，プロセス数が 32 以上の場合，転送時間の増加よりも通信待ち時間の削減が大きく，提案手法の効果が高い．

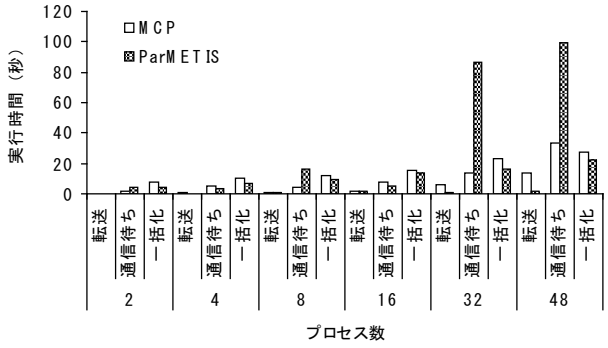
通信待ち時間はプロセス間の計算量の不均衡により発生する．図 9 に，Wang モデルに対して，ParMETIS と MCP をそれぞれ適用した場合の各プロセスの式の分散状況を示す（プロセス数=32）．図 9(a) より，ParMETIS による分割では，プロセス数が 32 のとき，1 つのプロセスに計算が集中し，全体の 26%の式がプロセス 11 で計算される．この計算量の不均衡のために，通信待ち時間が増加したと考えられる．これに対して図 9(b) より，MCP による分割では，プロセス数が 32 のとき，1 つのプロセスは全体の 2.4〜4.4%の式を計算する．図 9(a) と比較して，計算量の不均衡が小さい．したがって，ParMETIS と比較して，通信待ち時間を削減できる．

ただし，プロセス数が 4,16 の場合，提案手法を適用しても通信待ち時間を削減できない．この理由は，ParMETIS を用いた分割においても負荷を均等に分散できたためである．分割数によって ParMETIS の負荷分散が異なる原因の究明は今後の課題である．

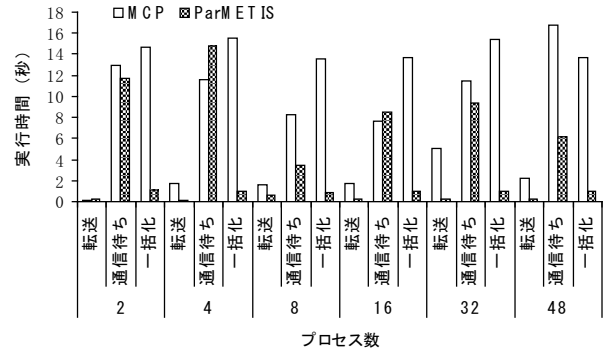
一方図 8(b) より Luo-Rudy モデルでは，提案手法はプロセス数が 4,16 のとき，従来手法と比較して通信待ち時間を削減する．また提案手法はプロセス数が 4,8,16,32,48 のとき従来手法と比較して，転送時間が増加する．

Luo-Rudy モデルは切断辺数の削減を重視した分割により，通信待ち時間，転送時間が削減される．この ISML モデルは直線状の接続，つまり細胞間の依存が少ない．このため，ParMETIS の分割では，同一レベルでの負荷の偏りが発生せず，通信待ち時間が Wang モデルのように増加しない．

しかし MCP の分割では，ALAP time の小さい順にプロセスに割り当てる．このとき，頂点の割り当てにおいて，頂点

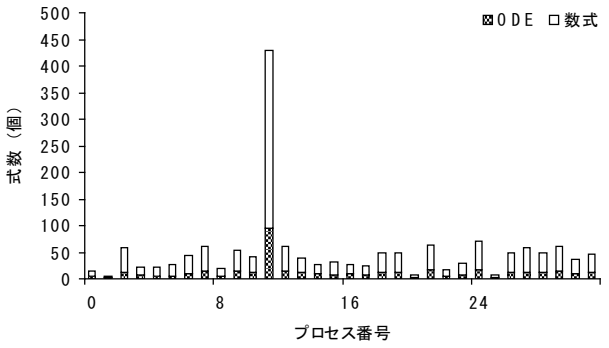


(a) Wang

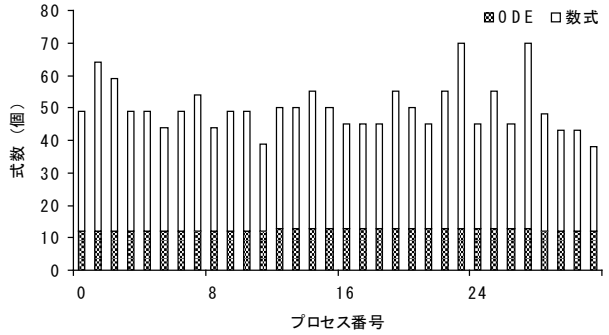


(b) Luo-Rudy

図 8 グラフ分割手法を変更した場合の通信待ち時間



(a) ParMETIS



(b) MCP

図 9 グラフ分割手法を変更した場合の式の負荷分散（プロセス数=32）

間の依存関係に基づいた切断辺数の削減よりも各プロセスの遊休時間の削減を優先する．このため，転送時間および通信待ち時間が増加すると考えている．したがって Luo-Rudy モデルでは提案手法の効果が低い．

5.3 通信データの一括化オーバヘッド

図 10 に，一括化オーバヘッドの計測結果を示す．図 10 は，従来手法にデータ配置の変更のみを適用した場合の計測データである．

図 10(a) から，Wang モデルでは，提案手法はすべてのプロセス数において，従来手法と比較して一括化オーバヘッドおよび通信待ち時間を削減する．どのプロセス数においても，98〜99%まで一括化オーバヘッドを削減する．提案手法は，プロセス数が 4,8,16,32,48 のとき，従来手法と比較して転送時間が増加する．

このように Wang モデルのような集中度の高い ISML モデルでは，データの配置変更は有効である．しかし，プロセス数の増加に伴い，転送時間の増加量が増大している．このため，より大規模な環境では，転送時間の増加がデータの配置変更の削減効果を上回る可能性がある．

一方図 10(b) より，Luo-Rudy モデルでは，提案手法はすべてのプロセスにおいて，従来手法と比較して一括化オーバヘッドを削減する．一括化オーバヘッドは 92〜93%に削減される．提案手法は，すべてのプロセス数において，従来手法と比較して，転送時間が増加する．また 4,8,16,32,48 プロセスのとき，通信待ち時間が増加する．

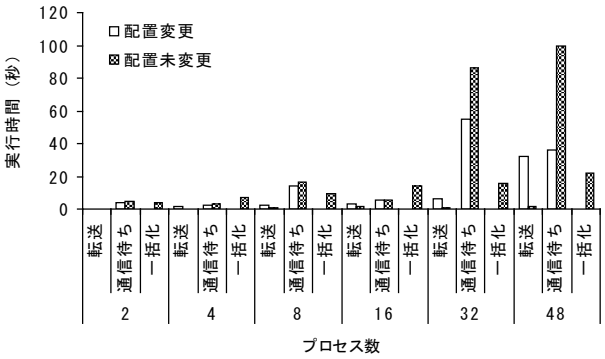
Luo-Rudy モデルは集中度が低いため，各共有データを必要とするプロセス数が少ない．ゆえに共有データを全プロセスで共有する通信を追加すると，通信量と通信回数がともに増加する．その結果，転送時間，通信待ち時間が増加し，データの配置変更による一括化オーバヘッドの削減効果を上回る．このため，Luo-Rudy モデルでは，データの配置変更は有効でない．

6. おわりに

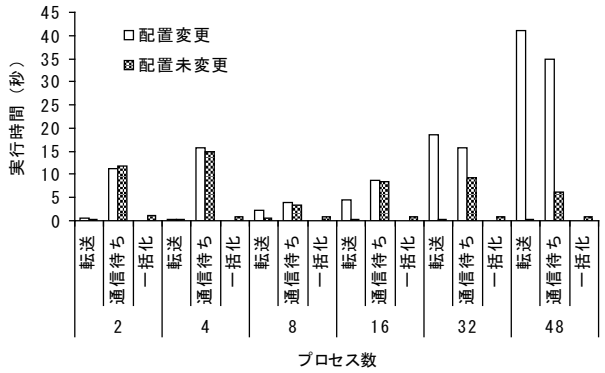
本論文では，*insilico*Sim を対象に集中度が高い ISML モデルに対するグラフ分割手法の改善，及びデータの配置変更について提案した．クリティカルパスを短縮するグラフ分割手法を用いることで，各プロセスの遊休時間を削減できることを示した．また通信データ量が多い場合に，性能低下の原因となる通信データ一括化時のオーバヘッドを削減した．具体的には，初期化時に通信データをメモリ上で連続領域となるように並べ替えることで，毎ステップ発生していたメモリコピーを削除した．

結果，集中度の高い Wang モデルに提案手法を適用した場合，CPU の単一プロセスで実行する場合に比べ，最大 9.5 倍の速度向上を達成した．

今後は通信待ち時間をさらに削減できるグラフ分割手法の考案に取り組む．また共有データの転送時間を削減する工夫に取り組む．さらに ISML モデルの特徴に応じて，グラフ分割手法を自動的に切り替える手法の研究に取り組む．



(a) Wang



(b) Luo-Rudy

図 10 データ配置を変更した場合の通信一括化オーバヘッド

謝辞 本研究の一部は以下の支援による．

- 科学研究費補助金（基盤研究 (B)23300007）
- 科学研究費補助金（若手 (B)23700036）
- グローバル COE プロジェクト *in silico* medicine

参 考 文 献

1) Asai, Y., Suzuki, Y., Kido, Y., Oka, H., Heien, E., Nakanishi, M., Urai, T., Hagihara, K., Kurachi, Y. and Nomura, T.: Specifications of *insilicoML 1.0: A Multilevel Biophysical Model Description Language,* The Journal of Physiological Sciences*, Vol. 58, No. 7, pp. 447–458 (2008).*

2) Hucka, M. et al.: The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models, Bioinformatics*, Vol.19, No.4, pp.524–531 (2003).*

3) Cuellar, A. A., Lloyd, C. M., Nielsen, P. F., Bullivant, D. P., Nickerson, D. P. and Hunter, P. J.: An Overview of CellML 1.1, a Biological Model Description Language, SIMULATION*, Vol.79, No.12, pp.740–747 (2003).*

4) Heien, E.M., Okita, M., Asai, Y., Nomura, T. and Hagihara, K.: insilicoSim: an extendable engine for parallel heterogeneous biophysical simulations, Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, SIMUTools '10, ICST, Brussels, Belgium, Belgium, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), pp.78:1–78:10 (2010).*

5) Karypis, G. and Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM Journal on Scientific Computing*, Vol. 20, pp. 359–392 (1998).*

6) Schloegel, K., Karypis, G. and Kumar, V.: Parallel Multilevel Algorithms for Multi-constraint Graph Partitioning, Euro-Par 2000 Parallel Processing *(Bode, A., Ludwig, T., Karl, W. and Wismuller, R., eds.), Lecture Notes in Computer Science, Vol. 1900, Springer Berlin / Heidelberg, pp. 296–310 (2000). 10.1007/3-540-44520-X_39.*

7) Wu, M. Y. and Gajski, D.D.: Hypertool: A Programming Aid for Message-Passing Systems, IEEE Trans. Parallel Distrib. Syst.*, Vol.1, pp.330–343 (1990).*

8) Heien, E.M., Asai, Y., Nomura, T. and Hagihara, K.: Optimization Techniques for Parallel Biophysical Simulations Generated by insilicoIDE, IPSJ Online Transactions*, Vol.2, pp.149–161 (2009).*

9) Wang, X.-J. and Buzsaki, G.: Gamma Oscillation by Synaptic Inhibition in a Hippocampal Interneuronal Network Model, The Journal of Neuroscience*, Vol.16, No.20, pp.6402–6413 (1996).*

10) Luo, C. and Rudy, Y.: A model of the ventricular cardiac action potential. Depolarization, repolarization, and their interaction, Circulation Research*, Vol. 68, No.6, pp.1501–1526 (1991).*