Vol.2014-HPC-143 No.17
2014/3/3

汎用生体モデルの高速な並列シミュレーションのための階層的依存グラフの自動分割

吉川 禎[1]　中村 亜希[2]　置田 真生[1]　安部 武志[3]　浅井 義之[3]　北野 宏明[3]　野村 泰伸[4]　萩原 兼一[1]

概要：我々は汎用生体シミュレータ Flint を改良し，大規模な生体モデルを階層的グラフとして解釈・分割した結果をもとに並列シミュレーションコードを生成する手法を開発している．この手法では冗長計算による通信の削減を基本方針としており，従来の通信時間を辺の重みとしたグラフ分割ではシミュレーションコードの性能が低下しうる．本研究では，冗長計算および通信を実行時間に与える影響という観点から統一して扱い，その影響度を辺の重みとして表す．これによりシミュレーションコードにおける冗長計算および通信の最小化を達成するような分割を目指す．結果として，通信による影響が計算による実行時間の増加の 80 倍程度である環境において，従来の分割方針を用いた場合と比較して通信時間を 28%，総実行時間を 4.4%削減した．

キーワード：汎用生体モデル 生体シミュレーション 冗長計算 重みつきグラフ分割

Partitioning Hierarchical Dependency Graph for Accelerating Parallel Simulation of General Biophysical Models.

Yoshikawa Tadashi[1]　Nakamura Aki[2]　Okita Masao[1]　Abe Takeshi[3]　Asai Yoshiyuki[3]
Kitano Hiroaki[3]　Nomura Taishin[4]　Hagihara Kenichi[1]

Abstract: Flint, a general biophysical simulator, requires a new approach that interprets a huge biophysical model as a hierarchical graph and partitions the graph in the higher coarse-grained layer to generate a parallel simulation code. Since the hierarchical graph has two types of edges, which are respectively translated to a redundant calculation and a communication in generated code, a traditional minimum-cut graph partitioning assuming single edge type may degrade the performance of the generated code. Our proposed method integrates two edge types and weights edges with their bad effect on the execution time of the code, aiming to minimize redundant calculations and communications in the code. As a result, our method achieved to decrease the communications by 28% compared with previous methods.

Keywords: General biophysical model, Biophysical simulation, Graph partition

1. はじめに

生理学の分野において生体機能の解析を目的とした研究が盛んである．その研究のひとつに，生体機能をモデル化し計算機上でシミュレーションする研究がある．シミュレーションの対象となる生体機能は，細胞，組織および臓器など多階層にわたるとともに，特定の機能に限定されない．これら複数の生体機能は相互に作用しあうため，それ

[1] 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University
[2] 大阪大学基礎工学部情報科学科
Department of Information and Computer Sciences, School of Engineering Science, Osaka University
[3] 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
[4] 大阪大学大学院基礎工学研究科
Graduate School of Engineering Science, Osaka University

らを統合的に扱う生体シミュレーションが求められている．

多階層かつ多機能な生体モデルの研究を目的とした開発環境として，PhysioDesigner[1][2] がある．PhysioDesigner は，汎用的な生体機能のモデル化を支援する統合開発環境である．このモデルは XML に基づく記述言語である PHML（Physiological Hierarchy Markup Language）[3][4] により記述される．PHML は汎用的な生体機能を，階層構造を持つ数理モデルとして記述できる．また PhysioDesigner は，PHML により記述された生体モデル（以降，PHML モデル）を入力とするシミュレータとして Flint[5][6] を提供する．

Flint は PHML モデルを対象として，並列シミュレーションを実行する汎用生体シミュレータである．Flint を利用したシミュレーションは，PHML モデルからシミュレーションコード（以降，SC）への変換と生成した SC の実行に分かれる．SC の変換において，Flint は PHML モデルを細粒度の依存グラフとして解釈する．このグラフを部分グラフに分割し PE に割り当てた後，並列に SC を生成する．この細粒度グラフを介して SC を生成する手法を以降では FGM と表す．生成した SC を並列に実行することによりシミュレーション結果を得る．

近年では，生理学の発展に伴い生体モデルが複雑化している．FGM では細粒度グラフの生成が性能ボトルネックになるため，モデルの大規模化により実行時間および主記憶の使用量が増加し，実用的内にシミュレーション結果を得られない．そこで Flint は PHML モデルを階層的な依存グラフとして解釈する手法（以降，HGM）により必要となる主記憶量を削減する．HGM では，高階層のグラフを分割し低階層のグラフに対して最適化を行う．

ただし，HGM では FGM と比較して SC の実行における通信が増加し，通信パターンが複雑化する傾向にある．通信パターンの複雑化による実行性能の低下を避けるため，HGM においては積極的に冗長計算を追加することで通信を削減する．

Eric らは Flint の前身である *insilico*Sim を用いたシミュレーションにおいて，冗長計算の追加によって通信を削減している [5]．この手法では，辺の切断数が最小となるような分割をした後，切断辺に相当する冗長計算を追加する．しかし，依存関係の大きな連鎖を持つ辺を切断した場合には，大量の冗長計算がシミュレーションコードに追加され，実行時間が増大する．

そこで本研究では，冗長計算の削減，通信の削減，および各 PE における計算量の均衡の 3 つを満たすグラフ分割を目標とする．そのために，冗長計算と通信を SC の実行時間に与える影響度を用いて統一的に扱う．冗長計算および通信によって増大する SC 実行時間をグラフにおける辺の重みとし，既存の重み付きグラフの最小切断アルゴリズムを利用することで求める分割結果を得る．

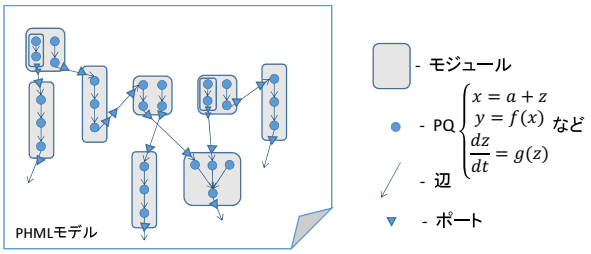

図 1 PHML モデル

2. 汎用生体シミュレータ

2 節では，本研究において対象とする PHML モデルおよびそのシミュレータである Flint について述べる．

2.1 入力：PHML モデル

PHML モデルは多様な生体機能とそのダイナミクスを記述できる [3]．そのうち，Flint の対象は常微分方程式（以降，ODE）で表現された生体機能の時間変化のシミュレーションである．

PHML では，生体モデルをモジュールおよび依存関係の集合として表現する．PHML モデルの主な構成要素を以下に示す．

- モジュール
  モジュール（以降，MD）は一塊の生体部品を表し，ポート，Physical Quantity（以降，PQ）および他の MD を内包する． MD はカプセル化されており，内部の PQ および MD は外部の MD から直接参照できない．
- ポート
  ポートは MD の特徴量を表し，外部の MD から参照できる．その値は，内部の PQ およびポートの転送（あるいは集約）として記述する．
- Physical Quantity
  PQ は生体機能が持つ物理量を表し，その値を定数，変数，代数関数，ODE および条件文を用いて記述する．
- 辺
  辺は PQ およびポート間の参照を表す．全ての辺は有向であり，1 つの辺につき 1 つのデータ依存を意味する．

2.2 Flint

Flint を用いた PHML モデルの並列シミュレーションについて述べる．

2.2.1 利用の流れ

Flint の利用の流れを**図 2** に示す．全体は 2 つの処理：SC s の生成および s の実行に分かれる．以降では，入力モデル p から s の生成および s の実行を，それぞれ $g(p)$ および $e(s)$ と記す．

まず $g(p)$ について述べる．Flint は p を構成する PQ および辺を抽出し，依存グラフ $G = (V, E)$ として解釈する（図 2 a.）．G は PQ を頂点，PQ 間の依存関係 $< v, u > (v, u \in V)$ を辺とする有向グラフである．p を構成する他の要素（MD およびポート）については，直接計算する必要がないため，G に含めない．例えば，PQ v がポート o を介して PQ u を参照する場合には，o を削除し辺 $< v, u >$ を G に追加する．

Flint は並列処理のために G を部分グラフに分割する（図 2 b.）．$e(s)$ において，各 PE が 1 つの部分グラフに属する PQ 群を計算する．したがって，分割における切断辺が s における通信に変換される．一般に並列処理においては通信が性能ボトルネックとなりうる．そこで切断辺が少なくかつ部分グラフの頂点が均衡する分割手法として，multilevel k-way partitioning algorithm[7] を用いる．なお，図 2 a および b の処理は単一 PE 上で逐次処理する．

得られた部分グラフを $G_1, G_2, \ldots, G_n\,(n = |PE|)$ と表す．分割後，各 PE において $G_k\,(1 \leq k \leq |PE|)$ に対するトポロジカルソート [8] を実行し，計算および通信のスケジュールを決定する（図 2 c.）．決定したスケジュールをもとに最適化を施した SC s_k を生成する（図 2 d.）．ODE で表される PQ 値の計算にはオイラー法あるいはルンゲ＝クッタ法を用いる．

最後に $e(s)$ について述べる．各 s_k を 1 つの PE に割りあて，並列に実行する．s_k にはオイラー法（ルンゲ＝クッタ法）の 1 タイムステップの計算内容が記述されている．これをユーザが指定するステップ数反復することでシミュレーション結果を得る．

2.2.2 ステップ内依存およびステップ間依存

オイラー法およびルンゲ＝クッタ法を用いて計算する場合，辺 $< y, x >$ が表すデータ依存は，ステップ内依存とステップ間依存の 2 種類に分類できる．この依存の種類は，y の値の計算式が ODE であるか否かによって決定する．

y が ODE でない場合，同一のタイムステップ t_q の一連の計算において，y を計算した後に x を計算する依存関係が生じる．このような依存をステップ内依存と呼び，ステップ内依存を表す辺の集合を E_f と表す．

一方で y が ODE の場合，x および y の間に t_q 内における依存は存在しない．この理由は，オイラー法の計算方法による．例えば，x の計算式が ODE $dx/dt = f(y)$ である場合，これは

$$\begin{cases} t_q = t_{q-1} + \Delta t \\ x_q = x_{q-1} + \Delta t \cdot f\,(y_{q-1}) \end{cases}$$

と計算する．t_q における x の計算には，t_{q-1} における y の値を用いる．ルンゲ＝クッタ法についても同様である．このような依存をステップ間依存と呼び，ステップ間依存を表す辺の集合を E_o と表す．

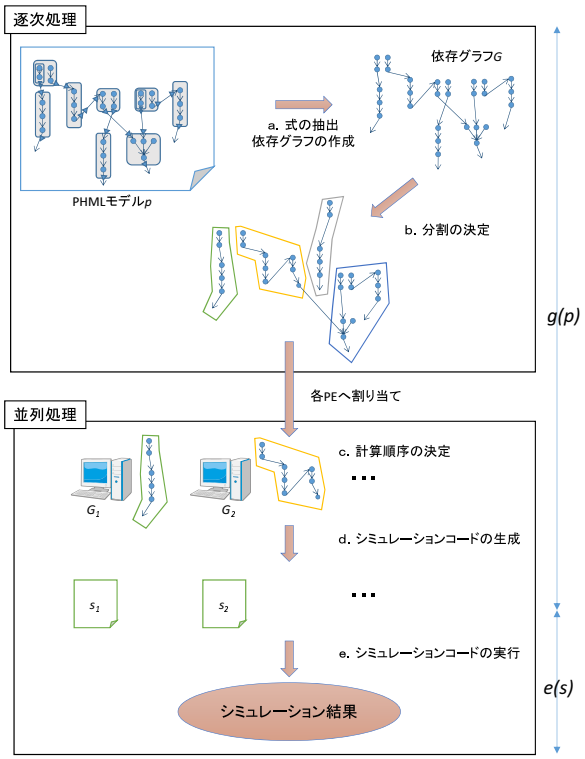

図 2 Flint におけるシミュレーション

データ依存の種類によって，$e(s)$ における通信タイミングの制約が異なる．$< y, x > \in E_f$ が切断された場合，y の計算後から同一タイムステップ内の x の計算前までに y の値を通信する必要がある（ステップ内通信と表す）．一方，$< y, x > \in E_o$ が切断された場合，次のタイムステップの開始までに y の値を通信すれば十分である（ステップ間通信と表す）．ステップ間通信はタイムステップの最後に一括化するなどの最適化が可能となる．

2.2.3 FGM の特徴

FGM は，PHML モデルの MD 構造を無視し PQ のみに着目した単純なグラフとしてモデルを解釈することで，以下の 5 つの利点を備える．

(i) PHML 以外の生体モデル記述言語を入力可能
(ii) 汎用的な並列処理
(iii) s のサイズ縮小
(iv) $e(s)$ における通信の効率化
(v) $e(s)$ における計算の効率化

Flint は PHML に加え，SBML[9] および PHML と SBML の混合モデルのシミュレーションが可能である (i)．また，モデルの MD 構造（多数の小 MD からなるモデル，単一の大 MD からなるモデル等）によらず並列度の高い s を生成できる (ii)．さらに，s に対して行・命令単位の最適化を施している [6](iii)(iv)(v)．

一方，FGM の欠点は大規模な生体モデルに適さない点である．まず，$g(p)$ においては G の生成が性能ボトルネックとなる．G の生成は逐次処理であるため，PQ 数の増大

に従って $g(p)$ の並列化効率が低下し，1PE に必要な最大主記憶量が増大する．例えば，約 2,000 万個の PQ を含む心筋細胞モデルに対しては，G の生成時間が $g(p)$ の全処理時間の約 20%を占め，64 GB 以上の主記憶を必要とする．

さらに，$g(p)$ および $e(s)$ において通信パターンの複雑化が問題となる．大規模な生体モデルを高い並列度でシミュレーションする場合，大量の切断辺によって通信回数が増大し通信パターンが複雑化する．特に一括化できないステップ内通信が原因となり，$g(p)$ における通信のスケジューリングに要する時間が増大し，$e(s)$ における通信オーバヘッドも増大する．

3. HGM

HGM は $g(p)$ において p を二階層のグラフとして解釈し，高階層における分割と低階層における最適化を組み合わせる手法である．

3.1 HGM の概要

HGM を用いた Flint の利用の流れを**図 3** に示す．HGM における SC s' の生成処理を $g'(p)$ と表す．

まず，生体モデル p を粒度の大きい高階層のグラフ $G^H = (V^H, E^H)$ として解釈する（図 3 a）．G^H は MD 木を頂点とし，MD 木間の依存関係 $< m_i, m_j > \left(m_i, m_j \in V^H\right)$ を辺とする有向グラフである．MD 木を頂点とする理由は，MD のカプセル化を利用して高階層グラフの辺数を削減するためである．入れ子になった MD 群は木構造を構成し，一般的な PHML モデルにおいて多くの依存関係は木内で閉じている．E^H は，PHML モデル内のポート間の依存関係のうち，MD 木を跨ぐ依存に相当する．

次に，G^H を部分グラフ G^H_k $(1 \leq k \leq |PE|)$ に分割し，各 PE へ割り当てる（図 3 b）．各 PE は割り当てられた G^H_k から PQ および辺を抽出し，PQ を頂点とする低階層グラフ $G^L_k = (V^L_k, E^L_k)$ を作成する（図 3 c）．G^L_k の構造は FGM における依存グラフと同じである．以降は FGM と同様の処理を G^L_k に適用し，SC s'_k を生成する（図 3 d）．

HGM の特長は，逐次処理において生成する G^H の規模が G と比較して小さい点にある．p の大規模化に伴うコード生成処理の性能ボトルネックを軽減し，大規模な p のシミュレーション結果を実用的な時間内に得ることを目指す．しかし一方で，HGM の採用に伴う課題も生じる．

3.2 HGM における課題

HGM を採用することで，FGM が備える利点（25 節）の一部が失われる．

まず (i) について，HGM は PHML の MD 構造に依存するため，PHML モデル以外の入力に対応できない．そこで HGM と FGM を併用し，大規模な PHML モデルに対しては HGM を採用し，小規模なモデルあるいは SBML（混合）モデル対しては FGM を採用する．SBML（混合）モデルも MD に似た構造を持つが，その利用は今後の課題である．

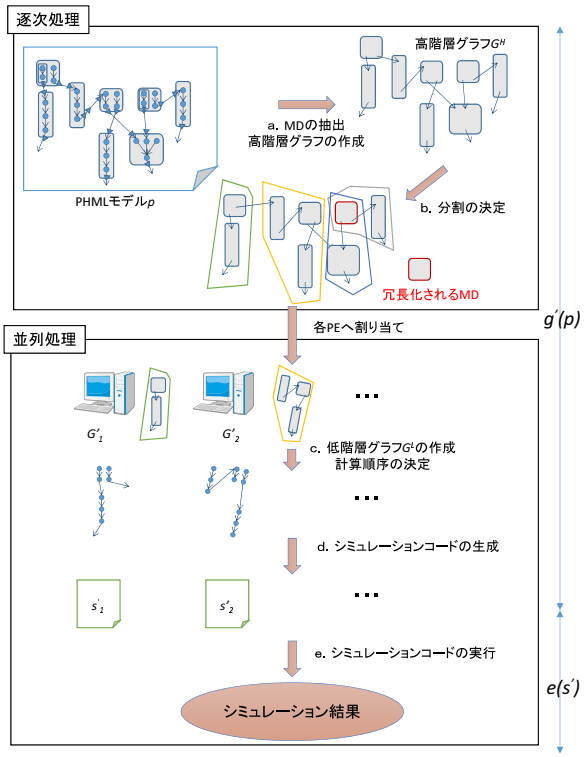

図 3 HGM によるシミュレーション

次に (ii) について，HGM を採用する場合に高い並列度を得るためには，PHML モデルが十分な数の MD を含む必要がある．ただし，本研究の対象とする大規模な PHML モデルは生体部品の数が数万個以上となり，対応する MD 数も同等以上となる．したがって大規模なモデルに対しては実用上問題とならないと予想される．

最後に (iii) について，HGM では FGM と比較して $e(s')$ における通信回数が増大する傾向が強い．これは，高階層における $e \in E^H$ の切断が，低階層においては複数の辺切断に相当するためである．特に $e \in E^H_f$ の場合は，大規模並列環境における実行で問題となるステップ内通信の通信パターンが複雑化し，$g'(p)$ および $e(s')$ の処理時間増大の原因となる．

3.3 冗長計算によるステップ内通信の削除

通信パターンの複雑化を避けるため，HGM では冗長計算の追加によるステップ内通信の削減を基本方針とする．**??**において述べた通り現在の Flint も冗長計算の機能を備える．しかし，それが通信コストの大きい環境における効率化オプションであることに対し，HGM では積極的に冗長計算を利用する点で異なる．

冗長計算の追加例を**図 4** に示す．HGM における G^H の分割の際，$e =< m_i, m_j > (e \in E^H_f)$ が切断される場合に

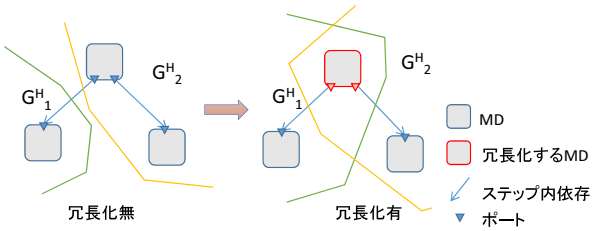

図 4 MD の冗長化

は m_i および m_i がステップ内依存によって接続する MD の集合を冗長化し複数の部分グラフの頂点に含める．$e(s)$ においては e に対応する通信が不要となる．ただし，冗長計算により $e(s)$ の計算時間は増大する．

4. 提案手法

本節では，$e(s')$ の実行時間 $T_{e(s')}$ を短縮するための G^H の分割手法を提案する．

4.1 問題設定

HGM では冗長計算を必須とするため，辺 e の表す依存の種類によって，e の切断が $e(s')$ の実行時間に与える影響が異なる．したがって，FGM と同様に単純に切断辺の少ない分割手法を用いても $T_{e(s')}$ が短縮するとは限らない．$T_{e(s')}$ を短縮するためには，s' が次の 3 つの条件：(1) 冗長計算の量が少ない，(2) 通信回数および通信量が少ない，(3) PE ごとの計算負荷が均衡している，を満たすことが望ましい．これらを G^H の分割に置き換えると，それぞれ，(1') $|C_f^H|$ が最小である，(2') $|C_o^H|$ が最小である，(3') $|V_k^H|$ が $1 \leq k \leq |PE|$ で均衡する，とみなせる．ここで C_f^H および C_o^H はそれぞれステップ内依存を表す切断辺の集合およびステップ間依存を表す切断辺の集合を表す．

一般に，グラフ分割における最小切断（(2') に相当）および頂点の均衡（(3') に相当）の両立は NP 困難である．さらに (1') も (2') および (3') と相互に関係があるため，全ての条件を満たす G^H の分割は難しい．

4.2 提案する分割手法

本研究では C_f^H および C_o^H の解釈を同一の観点で統一し，相違を辺の重みとして表すことで，G^H の分割に既存の重み付きグラフ分割アルゴリズムの適用を可能にする．切断辺に与える重みを調整することで，$T_{e(s')}$ を短縮する G^H の分割を得る．

G^H の分割における切断辺は，その依存の種類に関わらず，$T_{e(s')}$ を増大させる．この増大時間を切断コストと表す．$e_f \in C_f^H$ の場合，冗長計算を追加された PE の計算時間が増大する．切断コスト $t_r(e_f)$ は追加した冗長計算の量に比例する．$e_o \in C_o^H$ の場合，通信に参加する PE の通信時間が増大する．PHML モデルでは 1 つのデータ依存

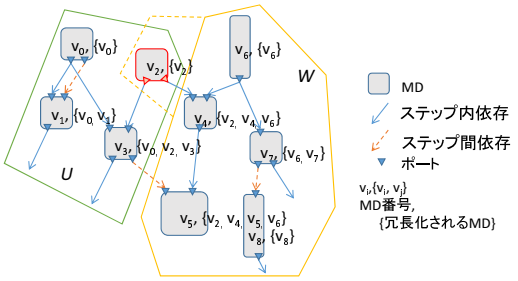

図 5 分割および辺の重み付けの例

は 1 つの変数参照を表すので，切断コスト $t_c(e_f)$ は待ち時間を無視するとほぼ定数となる．冗長計算および通信が全 PE に均等に分散すると仮定すると，全切断辺に依存する $T_{e(s')}$ の増大量，すなわち並列化に伴う $e(s')$ のオーバヘッド O は次の式 (1) で予測できる．

$$O = \frac{1}{|PE|}\left(\sum_{e \in C_f^H} t_r(e) + \sum_{e \in C_o^H} t_c(e)\right) \quad (1)$$

O は辺の切断コストの総和に比例する．したがって，辺の切断コストをその辺の重みとみなし，重み付きグラフの最小切断問題を解くことで，$T_{e(s')}$ を短縮する分割結果を得られる．最小切断問題を解く際には，FGM と同様に multilevel k-way partitioning algorithm を実装したライブラリ ParMETIS[10] を用いる．

以上から，提案手法では G^H に対して頂点の重み $w(v)$ および辺の重み $w(e)$ を次のように与える．

$$w(v) \equiv PQ(v)$$

$$w(e) \equiv \begin{cases} c_{ca} \sum_{u \in RD(e)} PQ(u) & (e \in C_f^H) \\ c_{co} & (e \in C_o^H) \end{cases}$$

ここで $PQ(v)$ は MD v を根とする MD 木が内包する PQ 数を表し，$RD(e)$ は辺 e の切断によって冗長化される MD の集合を表す．計算および通信のコストを表す定数 c_{ca} および c_{co} は環境によって異なるため，実験的に決定した値を用いる．

重み付けの例を**図 5** に示す．各頂点（MD）は，自身を参照する辺が切断された際に追加される冗長計算の重みを予め保持する．$V = \{v_0, v_1, \ldots v_8\}$ を頂点とするグラフが $\{U, W\}$ に分割されるとする．$U = \{v_0, v_1, \ldots v_3\}$，$W = \{v_4, v_5, \ldots v_8\}$ とすると，$e_1 =< v_2, v_4 > (e \in E_f^H)$ が切断され v_2 が W に冗長化される．したがって，$w(e_1) = c_{ca} \cdot PQ(v_2)$ となる．また，$e_2 =< v_4, v_5 > (e \in E_f^H)$ が切断される場合 v_2, v_4, v_5 が冗長化されるため，その重みは $w(e_2) = c_{ca}(PQ(v_2) + PQ(v_4) + PQ(v_5))$ となる．

なお，$T_{e(s')}$ を最小化するためには，冗長計算も含めた計算負荷を均等にする必要がある．しかし，提案手法では冗長計算を除いた場合の負荷分散しか考慮しておらず，この対応は今後の課題である．

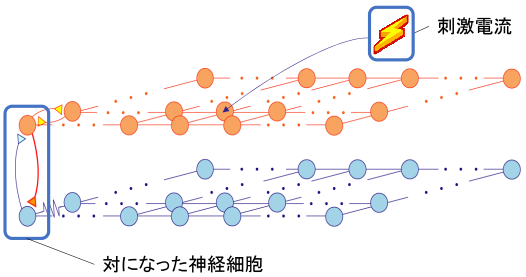

図 6 NN モデルの概略図

5. 適用実験

本節では以下の観点から提案手法を評価する．

- 分割したグラフの評価
- グラフ分割に要する時間
- $e(s')$ の実行性能

実験には 40 コアの共有メモリ計算機 1 台（Intel Xeon E7-4850 2.0 GHz 10-core × 4，主記憶 256GB）を用いる．OS は Ubuntu 12.04 を利用している．

実験は神経細胞ネットワークモデル（MD 数：約 1 万，PQ 数：約 264 万，以降 NN モデル）を対象とする．NN モデルは**図 6** のように，対になった神経細胞が格子状に接続したモデルである．モデルにおいては一対の神経細胞の組が MD 木として記述されており，各対が高階層グラフにおいて頂点として解釈される．細胞対同士の接続は ODE の参照であり，細胞への刺激電流のみ関数の参照による接続である．

5.1 グラフ分割結果

4.1 節の（1’）～（3’）をメトリクスとして，グラフ分割を評価する．また，FGM において用いられていた分割方針により高階層グラフを分割した場合と比較する．以降ではこの手法を従来手法と記載する．従来手法とは，ステップ内通信およびステップ間通信を同じ通信とみなし，通信回数の最小化を優先する分割方針である．

まず（1’）および（2’）を評価する．ステップ内依存を表す辺の切断数を**表 1** に，ステップ間依存を表す辺の切断数を**表 2** に示す．従来の手法では，グラフとして解釈する際に辺の重みに差がない．そのため切断辺の数を最小にできる限り，辺は無作為に切断される．それゆえ，追加される冗長計算のコストにばらつきがあると考えられる．

一方，提案手法では冗長計算と通信コストを同時に最小化することを目指している．そのため通信コストが増大しない範囲においては冗長計算のコストが小さい辺から切断される．通信コストと冗長計算のコストの和を最小化するため，通信コストとあわせて評価する．

今回実験した環境においては，通信 1 回の時間が計算 1 回の時間に対し約 80 倍であることを確認している．この

表 1 ステップ内依存の切断

分割数	2	4	8	16	32
提案する分割方針	2	4	6	6	6
従来の分割方針	0	6	4	6	8

表 2 ステップ間依存の切断数

分割数	2	4	8	16	32
提案する分割方針	488	1,028	1,778	2,926	4,462
従来の分割方針	608	9,58	1,868	3,038	4,564

表 3 追加される冗長計算のコスト

分割数	2	4	8	16	32
提案する分割方針	2	2	6	6	6
コスト	14	14	42	42	42
従来の分割方針	0	4	0	4	4
コスト	0	28	0	28	28

表 4 NN モデルにおける標準偏差

分割数	2	4	8	16	32
σ（MD）	3.5e-1	1.3e-1	2.2e-1	1.7e+1	3.9e-2
σ（PQ）	1.2e+2	3.0e+3	4.7e+3	5.5e+3	1.6e+3

比率と**表 3** に示した追加される冗長計算のコストを元に，実行時間増加量を見積もり評価する．1 回の通信が 1 回の計算の 80 倍の重みになるようギャップを設定する．これにより，4.2 節において示した $w(e)$ を比較する．結果，分割数が 4 の時を除き $w(e)$ は提案する手法を用いた際に小さくなっている．

最後に（3’）を評価する．各 V_k^H の重みの均衡（$V_k^H \subseteq V^H$）について，分割時の各 PE における MD 数，PQ 数の標準偏差 σ を指標として評価する．標準偏差を**表 4** に示す．

$\sigma(MD)$ は高々 $1.7e+1$ であり，総数の約 0.1%である．また，G^L 間の負荷分散について考えると，$\sigma(PQ)$ は，高々 $5.5e+3$ であり総数の約 0.2%である．したがって高階層グラフの分割時の負荷分散は実現できている．

また，負荷分散は冗長計算の追加によっても変化する．本研究においては追加される冗長計算の負荷分散を実現できていない．しかし，NN モデルにおいては一部の MD 木を除くとステップ内依存が存在しないため，冗長計算の総量が全体の計算量と比べ小さい（表 1）．結果として冗長計算の負荷分散が実現できないことによる計算時間の増加が問題にならない．

5.2 HGM における分割のオーバヘッド

HGM における G^h を分割するオーバヘッドについて評価する．SC の生成時間および分割の時間を**表 5** に示す．

分割に要する時間は生成時間のうち高々 8%であることから，HGM における性能ボトルネックにはならない．

表 5 SC の生成時間および分割時間（秒）

分割数	2	4	8	16	32
生成時間	219.0	125.0	83.0	65.0	55.0
分割時間	3.8	3.9	4.3	4.3	4.5

表 6 SC の実行時間（秒）

	計算時間	通信時間	通信＋計算
提案する分割方針	227.8	34.1	261.9
従来の分割方針	226.8	47.1	273.9

5.3 SC の実行性能

生成した SC の実行時間について比較する．

提案手法による分割を利用した場合の SC と従来の分割方針による分割を利用した場合の SC を比較する．それぞれ分割数 32，ステップ幅 $2\mu sec$，ステップ長 $0.4sec$ として実験している．実行結果を**表 6** に示す．提案手法を用いることで通信時間を 28%削減した．一方で計算時間は増加したものの，通信時間の削減と比べ増加量は小さい．結果として全体で 4.4%の実行時間削減となった．これは，提案手法により，冗長計算の削減と通信の削減を両立した結果であると考えられる．

最後に，HGM により主記憶の使用量を削減できているか評価する．32PE を用いて並列に SC を生成した際の主記憶の使用量を調べた結果，FGM においては分割を担当する PE が最大 4.3GB の主記憶を使用している．FGM において，SC 生成において細粒度のグラフを生成する際に 3.0GB の主記憶を使用する．一方で HGM では分割後に細粒度のグラフを生成するため，細粒度のグラフ生成時に使用する主記憶の量を削減できる．結果として，各 PE における主記憶の使用量を 1.4GB まで削減した．

6. まとめと今後の課題

本研究は，大規模な生体モデルを対象とするシミュレーション手法である HGM を対象とする．HGM においては，冗長計算の削減，通信コストの削減，および各 PE の計算量の均衡を満たす分割が望ましい．これらは相互に関係があるため，全ての条件を満たす解を求めることは難しい．そこで，冗長計算の計算量および通信コストを実行時間に与える影響という観点から統一する手法を提案する．辺の重みを適切に設定し，既存の最小切断アルゴリズムを用いることで望む分割結果を得る．

提案手法を適用した結果を，従来の通信最小化を優先した分割結果と比較した．分割数が 4 の場合を除き，提案手法を用いることで通信を削減し計算が増加する分割結果を得る．実際に SC を実行したところグラフ分割の結果通り，通信時間を 28%削減し，全体の実行時間を約 4.4%削減した．

今後の課題として SBML モデルに対する HGM の適用，冗長計算を含めた計算の負荷分散の実現，および通信時間と計算時間の重みの自動決定がある．

謝辞 本研究の一部は科学研究費補助金（基盤研究（B）23300007，若手研究（B）23700036，新学術領域研究（研究領域提案型）25136711）および文部科学省「医・工・情報連携によるハイブリッド医工学産学連携拠点整備事業」の支援による．

参考文献

[1] PhysioDesigner.org: About PhysioDesigner, http://www.physiodesigner.org/.
[2] Asai, Y., Abe, T., Okita, M., Okuyama, T., Yoshioka, N., Yokoyama, S., Nagaku, M., Hagihara, K. and Kitano, H.: Multilevel Modeling of Physiological Systems and Simulation Platform: PhysioDesigner, Flint and Flint K3 Service, *Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on*, pp. 215–219 (2012).
[3] PhysioDesigner.org: About Physiological Hierarchy ML (PHML), http://physiodesigner.org/phml/index.html.
[4] 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, *Journal of Physiological Sciences*, Vol. 58, No. 7, pp. 447–458 (2008).
[5] Heien, E., Okita, M., Asai, Y., Nomura, T. and Hagihara, K.: insilicoSim: an Extendable Engine for Parallel Heterogeneous Biophysical Simulations, *Proceedings 3rd International Conference on Simulation Tools and Techniques* (2010).
[6] Okuyama, T., Okita, M., Abe, T., Asai, Y., Kitano, H., Nomura, T. and Hagihara, K.: Accelerating ODE-based Simulation of General and Heterogeneous Biophysical Models using a GPU, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 99, No. PrePrints (2013).
[7] Karypis, G. and Kumar, V.: Multilevel Algorithms for Multi-constraint Graph Partitioning, *Proceedings of the 1998 ACM/IEEE Conference on Supercomputing*, Supercomputing '98, pp. 1–13 (1998).
[8] Reynolds, R. G.: An introduction to cultural algorithms, *Proceedings of the third annual conference on evolutionary programming*, World Scientific, pp. 131–139 (1994).
[9] Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., Arkin, A. P., Bornstein, B. J., Bray, D., Cornish-Bowden, A., Cuellar, A. A., Dronov, S., Gilles, E. D., Ginkel, M., Gor, V., Goryanin, I. I., Hedley, W. J., Hodgman, T. C., Hofmeyr, J.-H. S., Hunter, P. J., Juty, N. S., Kasberger, J. L., Kremling, A., Kummer, U., Novère, N. L., Loew, L. M., Lucio, D., Mendes, P., Minch, E., Mjolsness, E. D., Nakayama, Y., Nelson, M. R., Nielsen, P. F., Sakurada, T., Schaff, J. C., Shapiro, B. E., Shimizu, T. S., Spence, H. D., Stelling, J., Takahashi, K., Tomita, M., Wagner, J. M., Wang, J. and the rest of the SBML Forum: 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).
[10] Karypis, G. and Kumar, V.: A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs, *SIAM J. Sci. Comput.*, Vol. 20, No. 1, pp. 359–392 (1998).