社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2005－SLDM－118 (12)
2005／1／25

SBML対応細胞シミュレータ環境の構築

岩岡 洋† 長名 保範†† 福島 知紀†† 吉見 真聡†† 舟橋 啓†††
広井 賀子††† 柴田裕一郎†††† 岩永 直樹†††† 北野 宏明††† 天野 英晴†

† 慶應義塾大学理工学部
†† 慶應義塾大学大学院理工学研究科
〒223-8522 横浜市港北区日吉3-14-1

††† 科学技術振興機構 北野共生システムプロジェクト
〒150-0001 東京都渋谷区神宮前6-31-15 マンション31, 6A

†††† 長崎大学工学部情報システム工学科
〒852-8521 長崎市文教町1-14

E-mail: †bio@am.ics.keio.ac.jp

あらまし 近年，細胞内の化学反応機構のシミュレーションに関する研究が盛んである．しかし，大規模なモデルを扱う場合，膨大な計算時間を要するため，FPGAを用いてシミュレーションを高速化する研究が行われている．これまでに開発された多くのシミュレータは，SBMLと呼ばれるモデル記述言語が標準的に利用しており，互換性を保つことが重要である．本研究報告では，SBMLで記述されたモデルをFPGA上での効率的な処理を行うシミュレーション環境のための一部ソフトウェアを実装し，評価を取った．

キーワード SBML, FPGA, 生化学シミュレーション

Design of cellular simulation platform for SBML model

Yow Iwaoka†, Yasunori Osana††, Tomonori Fukushima††, Masato Yoshimi††,
Akira Hunahashi†††, Noriko Hiroi†††, Yuichiro Shibata††††, Naoki Iwanaga††††,
Hiroaki Kitano†††, and Hideharu Amano†

† Faculty of Science and Technology, Keio University
†† Graduate School of Science and Technology, Keio University
Hiyoshi 3-14-1, Yokohama-shi, Kanagawa, 223-8522 JAPAN

††† Kitano Symbiotic Systems Project, ERATO-SORST, Japan Science and Technology Agency
Suite 6A, M31, 6-31-15 Jingumae, Shibuya-ku, Tokyo, 150-0001 JAPAN

†††† Dept. of Computer and Information Sciences, Nagasaki University
1-14 Bunkyo-machi, Nagasaki, 852-8521 JAPAN

E-mail: †bio@am.ics.keio.ac.jp

Abstract Computer simulation in widely used in biochemistry research. However, simulating large-scale model requires a large computational time. To reduce it, we have been developing a FPGA-based high-speed simulator, but previous research mainly forcues on high speed implementation, and standard model for description has not been prepared. SBML model, that is a standard modeling language for biochemical networks, has been popular in many software simulator. In this reseach report, an environment of FPGA-based simulator which can use SBML for modeling is implemented and evaluated.

Key words SBML, FPGA, Biochemical Simulation

1. はじめに

1980 年代後半から，生物学の急速な発展により，生物の遺伝子配列や，細胞内の代謝系などに関する大量の定量的なデータが蓄積されている．また，計算機の処理性能が飛躍的に向上したことから，それらの膨大なデータに対し計算機を用いて処理を行うことで，細胞内の代謝システムを再現しようとする試みが行われるようになった．こういった細胞シミュレータとして，E-Cell [1] や，Virtual Cell [2] などが開発されてきた．

しかしこれらのシミュレータは，計算量が膨大なため，細胞全体を対象にするような大規模なシミュレーションにおいては，計算時間が大きな問題となる．科学計算の分野では，専用計算機を利用することで大幅な性能向上が可能であるが，細胞システムのシミュレーションでは，アルゴリズムの変更，拡張が頻繁に行われ，シミュレーション対象によっては多種類のアルゴリズムを複合して利用することがあるため，専用ハードウェアを設計することは困難であり，コスト面でも現実的ではない．

当研究グループでは，FPGA を用いた高速で，多種類のアルゴリズムが利用可能な細胞シミュレータ ReCSiP(Reconfiguable Cell Simulation Platform) [3] を開発してきたが，今までの研究は主として個々のアルゴリズムの高速化であった．しかし，この種のシステムでは，ユーザが容易にシミュレーション対象のモデリングを行えるようにすることが重要である．そこで，これまでに開発された多くのシミュレータで利用可能なモデル記述言語 SBML を入力インターフェイスとするような，ReCSiP によるシミュレーション環境を構築する．

1.1 生化学シミュレーション

生物細胞内では様々な化学反応が関与し，多数の反応経路を形成している．これらの反応が発生すると，細胞内の分子濃度が変化する．シミュレータでは，細胞の代謝に関わるすべての反応の相互作用を詳細に記述し，個々の反応の発生をシミュレートする．これらの反応経路のモデル化する手法として，従来から微分方程式を用いたものが利用されてきた．

たとえば，以下のような反応を考えた場合，

$$A + B \xrightarrow{k} C \tag{1}$$

ある分子 X の濃度を $[X]$ と表すとすると，この反応の反応速度は，

$$v = \frac{d[C]}{dt} = k[A][B] \tag{2}$$

と表わすことができる．k は反応に固有の反応速度定数である．

各分子濃度の時間変化を微分方程式で記述して，初期値を与え，数値計算を行なうことよって，細胞内の各分子濃度を時系列に沿って求めることが可能である．

1.2 SBML/SBW

SBML(Systems Biology Markup Language) [4] とは XML に基づく生化学モデルの記述言語で，システム生物学の研究におけるコンピュータ上でのモデルを表現するためのものである．これまでに開発された多くのシミュレータは，入力インターフェースとして SBML が利用可能である．また，シミュレータ以外にも，たとえば，CellDesigner [5] といったモデリングツールや，分析用ソフトウェアなどが開発されており，GUI を利用したモデルの作成や分析が可能である．

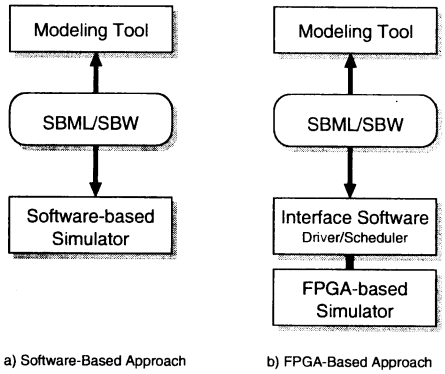

図 1 SBML/SBW 対応シミュレータ

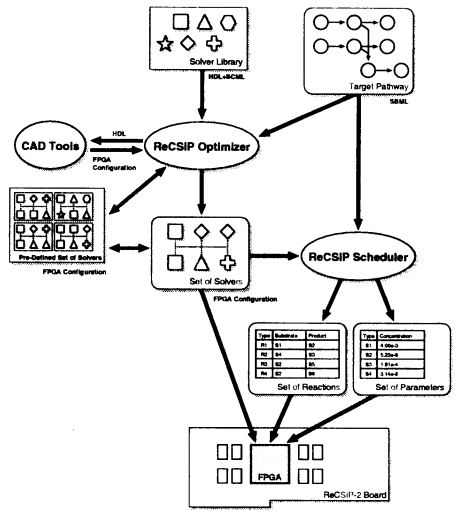

図 2 ReCSiP を用いたシミュレーション

SBW(Systems Biology Workbench) [6] は，オープンソースのアプリケーション統合環境である．独立して開発されているシミュレータや分析用ソフトウェアを簡単に結合するという目的で開発された．SBW と SBW 準拠のソフトウェアを使うことで，ユーザが異なるソフトウェアにシームレスにアクセスすることが可能となる．

2. ReCSiP システム

FPGA を用いたシミュレータ環境 ReCSiP は，図 1 に示すように，入力として SBML/SBW インターフェースを備える．このようなインターフェースを備えることで，ユーザは，ほかのシミュレータや，SBW に準拠したソフトウェアを利用するときと同様の操作で ReCSiP を利用できる．

図 2 に，SBML モデルを ReCSiP を利用してシミュレーションを行うときの様子を示す．このとき，システム内で重要とな

るコンポーネントについて次に詳しく述べる．

2.1 Solver ライブラリ

既に述べたように，シミュレータは，反応の機構によって決定される微分方程式を利用して分子濃度の変化を計算することでシミュレーションを行う．多くの場合，1 つのシミュレーション対象内で，複数種類の反応機構が利用される．そこで様々なモデルに対応させるために，特定の微分方程式が計算可能なハードウェアモジュール “solver” を多種類用意し，ユーザに solver ライブラリとして提供する．

現在，SBML Level1 [7] において標準で定義されている関数群を計算するための solver ライブラリを整備中である．

2.2 オプティマイザ

生化学シミュレーションは，対象によって性質が大きく異なるため，あらかじめすべてのモデルに対応した回路を FPGA 上に配置しておくのは不可能である．そこでオプティマイザは，SBML で書かれたモデルを読み込み，solver ライブラリから必要な solver を選別し，それらを FPGA 上に配置する．このとき，FPGA に面積的な余裕がある場合は，同種の solver を複数配置することで，並列計算による高速化を図る．

2.3 スケジューラ

シミュレーション対象内の分子は，複数の反応に関与している．それらの反応が同種の反応機構なら，1 つの solver のみで計算を行えるが，機構が異なる場合は，複数の solver で分子濃度データが利用される．この場合，データを solver 間で共有する必要があり，solver 間での通信が発生する．そこで，スケジューラは，必要に応じて，計算前の各 solver へのデータの配布や，計算後のデータの回収などの，データ転送のスケジューリングを行う．

スケジューラについては，後の章で詳細を述べる．

2.4 ReCSiP-2 ボード

ReCSiP-2 ボードは，FPGA とメモリから構成される PCI ボードである．ホスト CPU と FPGA 上のハードウェアでそれぞれ処理を分担することによって，効率の良い処理が可能である．FPGA 上では多数の演算器による並列計算を行なうことができ，クラスタに比べ安価であり，通信処理によるオーバーヘッドは存在しない．

2.4.1 構　　成

ReCSiP-2 ボードのシステム構成と写真を図 3 および図 4 に示す．FPGA は，Xilinx 社の Virtex-II Pro(XC2VP70) が搭載されている．その周辺に，QDR-SRAM，DDR-SRAM，物理乱数ジェネレータチップを実装している．PC/WS の PCI インターフェイスとして QuickLogic 社の QuickPCI(QL5064) を実装しており，64bit/66MHz の PCI バスに接続することが可能である．

ボードはホストからはメモリデバイスとして見えるようになっており，ボード上のメモリや，FPGA 上に実装した任意のレジスタをメモリ空間上に割り当てることが可能である．PCI バスと，ボード上のローカルバスのクロックは分離されており，ハードウェア設計者の負担も軽減されるように配慮されている．

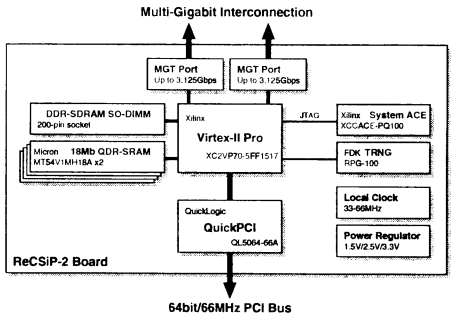

図 3　ReCSiP-2 ボードの構成

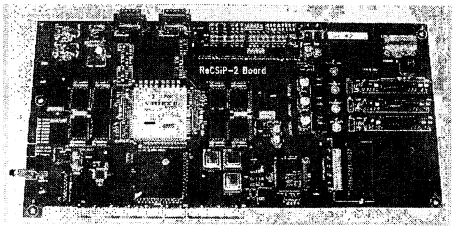

図 4　ReCSiP-2 ボードの写真

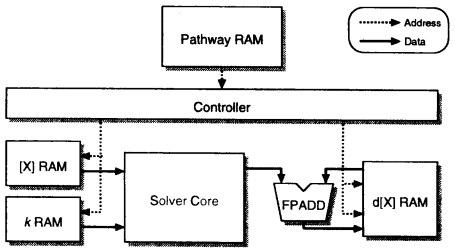

図 5　Solver 内部構造

3. シミュレータ構造

3.1 Solver

Solver は，特定の反応速度式を解いて，タイムステップあたりの分子濃度の変化を計算するモジュールであり，FPGA でシミュレーションを行う上で基本となるモジュールである．Solver を FPGA 上に多数配置することによって，並列計算が可能となり，計算時間を削減することができる．

Solver は図 5 に示すように，計算を行う core と，分子濃度データなどを格納しておくメモリと，コントローラから構成される．これらのコンポーネントについて以下に詳しく述べる．

3.1.1 Solver core

Solver core は，実際に計算を行うモジュールであり，内部には複数の演算器が配置されている．分子濃度や反応速度定数を入力として受け取り，分子濃度の差分を出力とする．core はパイプライン動作が可能であり，そのパイプラインピッチは，多くの場合，数クロックである．

3.1.2 Solver 制御機構

core には，パイプラインピッチ毎にデータの入出力を制御する機構が必要である．この制御機構は図 5 に示すように，以下

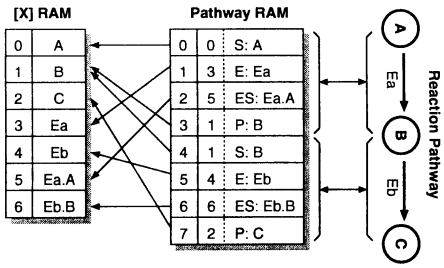

図 6　Pathway RAM による反応経路のプログラム

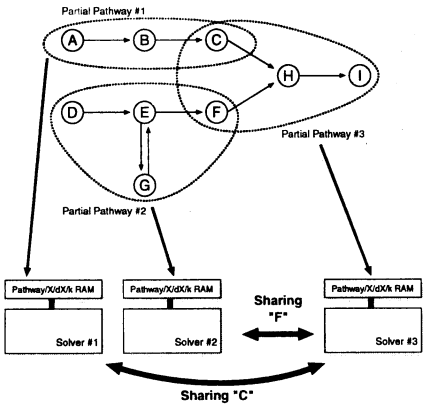

図 7　大きな反応系の分割

の 4 つのメモリを持っている．

- 各分子の濃度を保持する $[X]$ RAM
- 各反応の反応速度定数を保持する k RAM
- 分子の濃度差分を保持する $d[X]$ RAM
- 反応経路をプログラムする Pathway RAM

Pathway RAM には，図 6 に示すように，反応経路を表す $[X]$ RAM へのポインタが順に格納されている．これに従って $[X]$ RAM を読みだし，core に送ることで一連の反応を順に処理する．この図は簡略化されたもので，実際には，Pathway RAM の 1 ワードには，$[X]$ RAM へのポインタ，k RAM へのポインタと，反応系記述の終わりを示す制御ビットが含まれている．

一連の反応の処理が終わると，$d[X]$ RAM の内容を $[X]$ RAM の同じ番地に加算して，次のタイムステップの処理を開始する．アドレス管理の簡略化のため，$[X]$ RAM と、$d[X]$ RAM には，同じ番地には同じ分子に関するデータを格納するようになっている．

3.2 Solver 間通信機構

既に述べたように，一つのシミュレーション対象内で多種の反応モデルが利用されることが一般的であり，また並列計算による高速化のためにも，シミュレーションを行なう際に FPGA 上に複数の solver を配置させる必要がある．その際，図 7 のように，対象となる反応系をいくつかの小さな反応系の集合に分割して，分割された反応系をひとつずつ solver に割り当てて処理を行うことになり，solver 間で濃度データを共有する必

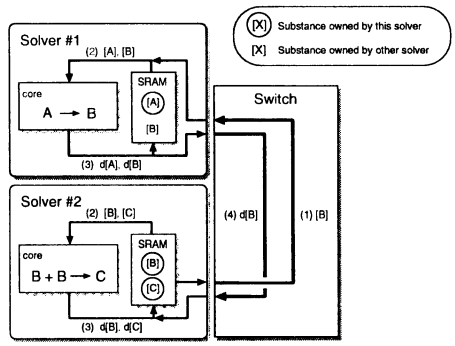

図 8　solver 間通信によるデータの流れ

要がある．Solver 間でのデータの共有は，以下のような方法で行う．

（1） いずれかの solver にオリジナルのデータ (つまり $[X]$) があり，必要に応じて共有する solver に最新のデータが配布される

（2） 配布された値を用いて計算した $d[X]$ は，計算が終ったらオリジナルデータをもつ solver に転送する

（3） 共有データのオリジナルを保持する solver は，転送されてきた $d[X]$ を集計して、次のタイムステップにおける $[X]$ に反映する

通信機構を利用した場合のデータ転送手順を簡略化したものを図 8 に示す．シミュレーションを行うために最低限必要な通信メカニズムは，上記の通り前半の処理における $[X]$ RAM 間でのデータ転送と，後半の処理における $d[X]$ RAM 間でのデータ転送であり，これは solver の外部からそれぞれのメモリを read/write することにより実現できる。

4. 実　　装

Solver が複数配置され，データが共有される場合，前章で述べたような手順に従って，データ転送を行う必要がある．そこで，SBML モデルの読み込みから，データ転送のスケジューリングの最適化までを自動化するためのソフトウェア (スケジューラ) の実装を行った．

4.1 反応の読み込み・分類

SBML モデルの読み込み部分は，libSBML [8] という SBML パーサライブラリを利用し作成した．

シミュレーション対象内には多種多数の反応があるので，それらの反応を計算手法によって分類し，それらが計算可能な solver に割り付ける必要がある．SBML では反応ごとに計算式を指定するので，それによって分類を行う．

4.2 反応の分割

同じ種類の solver を複数利用することができるので，そのような場合には，反応計算の割り付け先 solver を決定するために，同種の反応群をさらに分割する必要がある．このとき，solver 間で共有しなければならないデータを少なくすれば，solver 間におけるデータ転送数を削減でき，計算を効率良く行うことが可能である．そこで以下のようなアルゴリズムを実装した．

（1） 反応の個数分の反応群を用意し，各群に反応を一つずつ割り当てる

（2） 各反応群が計算時に利用するデータを割り出し，同じデータを利用する割合が高い反応群同士を結合し，新たな反応群を生成する

（3） 反応群が FPGA 上に配置する solver 数以下になるまで (2) の繰り返す

反応群同士を結合する際，一定数以上の反応を持つ反応群の生成を行わないようすることで，各 solver へ割り当てられる反応数がほぼ均等になるようにした．

4.3 オリジナルデータの配置

分子濃度データは，その整合性を保つため，各々にオリジナルデータを所有する solver を決めなければならない．そのデータを所有している solver は，すぐに計算を開始することができるが，所有していない solver はデータが配布されるまで計算を開始することができない．このとき，solver 内に計算を開始できる他の反応が存在する場合は，そちらを先に行い，残りの反応もデータが届き次第，計算を開始させることで，無駄な時間を減らすことが可能である．そこで，そのデータを計算に利用する反応の数が最も多い solver にその分子濃度のオリジナルデータ所有させるようにした．

4.4 オリジナルデータの配布

データの格納には FPGA 上のデュアルポートメモリを利用しており，core で計算を行っているときでも，他 solver とのデータ送受信が可能である．よってすべての反応計算に必要なデータが揃っていない場合であっても，揃っている反応から順次，計算を開始できる．そのデータを必要としない反応の数が多ければ，データの到着が遅れても solver core のパイプラインを空けることなく処理ができる．ここで，分子濃度のデータ X の到着が遅れても，パイプラインに空きが発生しない限界時間 slack(X) は，転送によるオーバーヘッドも考慮して，

$$\begin{aligned}\text{slack}(X) = \{&(\text{全反応数}) - (X \text{ を利用する反応数})\} \\ &\times (\text{core パイプラインピッチ}) \\ &- (\text{スイッチの転送オーバーヘッド})\end{aligned}$$

のように表わすことができる．そこで，すべての分子濃度データの slack を求め，その値が小さい順にデータの配布を行う．

またスイッチは，クロスバーで solver 間をつないでおり，送受信する solver が異なれば，同時に転送を行うことができるので，可能な限り多重転送を行う．

4.5 反応の計算・差分の回収

反応の計算は，必要なデータが揃ったものから，core への入力が空きしだい開始する．

他の solver がオリジナルデータを所有するものに関しては，計算が終了し，転送用ポートが空きしだい，計算した差分をオリジナルデータを所有する solver に転送する．この際，配布と同様，可能な限り多重転送を行う．

4.6 オリジナルデータの更新

Solver core では，現在のタイムステップにおける各分子の濃度差分を計算する．よって，現在の濃度を求めるには，前回までの分子濃度と，計算で求めた差分を加算しなければならない．各 solver で，オリジナルを所持しているデータに関しては，反応計算の終了直後から，その solver で求めた差分の加算処理を行う．それと同時に，他 solver で計算された差分が回収されてくるので，それらのデータも加算処理を行う．

5. 評　　価

今回実装した ReCSiP の内部スケジューラの評価として，スケジューリングの最適化を行ったときの効果について評価した．

現在，ReCSiP システムは開発中であり，全てのモジュールを利用することができない．そのため，評価を取るのに十分な規模の生化学モデルの入手が困難である．そこで，乱数を用いてモデルを生成し，評価を取った．

以下の項目について，最適化した場合としなかった場合の性能の比較を行う．

（1） 反応の分割

（2） 分子濃度のオリジナルデータを所有する solver の決定

（3） 分子濃度のオリジナルデータの配布

（4） 計算した分子濃度の差分の回収

この際，これらの項目において最適化を行わなかった場合は，以下のようなスケジューリングを行った．

（1） 各々の反応は，均等に solver に割り当てられる．この際，反応が利用する分子濃度データの共有は考慮しない

（2） すべての分子濃度のオリジナルデータは，ただひとつの solver が所有する

（3） すべての solver に必要なデータの配布が終了してから，各 solver での反応の計算を開始する．配布の際，多重転送は行わない

（4） すべての solver で反応の計算が終了してから，データの回収を開始する．この際，多重転送は行わない

今回は，solver の種類と配置数の変化が，これらの最適化にどのような影響を与えるかを調べるため，表 1 に示した 3 つのモデルを生成した．評価に際しては，実際に計算を行える solver は必要ない．全ての solver について，パイプラインピッチが 3 クロック，パイプラインの深さが 20 クロックと，比較的小規模なものを想定して評価を行った．

表 1　評価で用いたモデル

モデル	A	B	C
反応の種類	1	3	9
反応数 (種類 x 個数)	900 (1x900)	900 (3x300)	900 (9x100)
配置した solver (種類 x 配置数)	9 (1x9)	9 (3x3)	9 (9x1)
分子の種類	100	100	100

最適化項目 (1) と (2) は，solver 間での共有する分子濃度データの数に影響する．(1) と (2) を最適化した場合としなかった場合の，solver 間でのデータの転送数を表 2 に示す．

(1) は 1 種類の solver が複数配置されているときに，効率的な分割を行うためのもので，分割する個数が多いほど最適化の

表 2 Solver 間でのデータの転送数

最適化項目	モデル A	モデル B	モデル C
(1) と (2)	414	508	790
(2) のみ	465	557	790
(1) のみ	795	768	798
最適化なし	809	785	798

効果が現れすい．各種類の solver の配置数が減ると，最適化の効果が小さくなっていくのがわかる．モデル C においては，分割する必要がないので，最適化をしない場合と等価である．

また，(1) のみ最適化を行った場合，効率的な分割を行ったにも関わらず，オリジナルデータは一つの solver にまとめて配置されてしまうため，効果は薄い．

次に，各項目の組み合わせによる，サイクル当りのクロック数の変化を表 3 に示す．1 サイクルのクロック数というのは，データの配布が開始されてから，計算で得られた差分の加算処理が全て終了するまでにかかるクロック数である．これは，シミュレーションを 1 タイムステップ時間進めるのに要する時間と等価である．

表 3 最適化の結果

	1 サイクルのクロック数			core パイプライン利用率		
最適化項目	A	B	C	A	B	C
全項目	527	507	539	70%	75%	77%
(1),(2),(4)	659	651	688	53%	59%	60%
(1),(2),(3)	768	811	1021	48%	47%	40%
(1),(2)	900	955	1189	41%	40%	37%
最適化なし	1231	1207	1220	33%	34%	34%

最適化を行ったことで，全く行わないときと比較して，1 サイクルにかかるクロック数を最大 58% 削減できた．

全項目の最適化を行ったときと，行わなかったときのパフォーマンスは，どのモデルもほぼ同じであるが，(3) 又は (4) の最適化を行わなかった場合に，モデル C のパフォーマンスの低下が顕著である．モデル C は，表 3 に示したとおり，他のモデルに比べて solver 間転送の数が多いので，多重転送などを行わないことによる，オーバーヘッドが表面化したと考えられる．

最適化の効果は (3) より，(4) の方が大きいことがわかる．オリジナルデータの配布は，同じデータを複数の solver に転送するため，多重転送を行わない場合でも，その影響は少ないと考えられる．しかし，計算結果の差分データの回収は，個々のデータが異なる．さらに，今回評価に用いたモデルでは，どの solver も計算時間を同じに設定してあり，全ての solver にほぼ均等に反応が割り振られるので，反応計算の終了時間はほぼ同じである．そのため，多重転送を行わない場合，計算が終了したにもかかわらず転送ができない，といった情況が頻発しているものと考えられる．

これらの評価結果から，データの配布や回収，特に，多重転送を行うことで，パフォーマンスが大幅に向上することがわかった．

6. ま と め

本研究報告では，FPGA を用いて細胞内代謝系のシミュレータである ReCSiP の構造について述べ，シミュレータユーザが容易に ReCSiP システムを利用できる環境のための，SBML 入力インターフェースと内部スケジューラの実装し，評価を行った．実装したスケジューラによって内部のデータ転送手順の最適化を行うことで，計算時間を 58% 削減できた．

7. 今後の課題

今回は，SBML 入力インターフェース部を実装したが，シミュレーション結果の出力は標準的なフォーマットではない．また，オプティマイザの実装をまだ行っていないため，FPGA への solver 配置は手作業で行う必要がある．ユーザが利用しやすいシミュレータ環境を提供するために，今後，これらのソフトウェアの開発を行う予定である．

また，今回実装したスケジューラでは，生化学モデルを用いた性能検証を詳細には行っていないため，今後，検証を行い，実際のモデルで効率的な最適化手法を検討する必要がある．

謝 辞

この研究は文部科学省の平成 16 年度科学技術振興調整費による「システム生物学者育成プログラム」の一環として行われたものです．

文 献

[1] M. Tomita, et al.: "E-cell: software environment for whole-cell simulation", Bioinformatics, **15**, 1, pp. 72–84 (1999).

[2] J. Schaff, et al.: "A general computational framework for modeling cellular struc ture and function", Biophysical Journal, **73**, pp. 1135–1146 (1997).

[3] Y. Osana, T. Fukushima, M. Yoshimi and H. Amano: "An fpga-based acceleration method for metabolic simulation", IEICE Trans. on Information and Systems, **E87-D**, 8, pp. 2029–2037 (2004).

[4] M. Hucka, A. Finney, B. Bornstein, S. Keating, B. Shapiro, J. Matthews, B. Kovitz, M. Schilstra, A. Funahashi, J. Doyle and H. Kitano: "Evolving a lingua franca and associated software infrastructure for computational systems biology: The systems biology markup language (sbml) project", IEE Systems Biology, **1**, 1, pp. 41–53 (2004).

[5] A. Funahashi, N. Tanimura, M. Morohashi and H. Kitano: "Celldesigner: a process diagram editor for gene-regulatory and biochemical networks", BIOSILICO, **1**, 5, pp. 159–162 (2003).

[6] H. Sauro, M. Hucka, A. Finny, C. Wellock, H. Bolouri, J. Doyle and H. Kitano: "Next generation simulation tools: The systems biology workbench and biospice integration", Omics, **7**, 4, pp. 355–572 (2003).

[7] M. Hucka, A. Finney, H. Sauro and H. Bolouri: "Systems Biology Markup Language (SBML) Level 1: Structures and Facilities for Basic Model Definitions", Systems Biology Workbench Development Group, ERATO Kitano Symbiotic Systems Projec, MC 107-81 California Institute of Technology, Pasadena, CA 91125, USA, version 2 edition (2003).

[8] B. Bornstein, B. Kovitz, S. Keating, M. Hucka and A. Finney: "libsbml: A software library for the systems biology markup langauge", ICBS (2004).