ハイパフォーマンス
コンピューティング 65－5
（1997. 3. 6）

ATMネットワーク並列シミュレータの構築

守屋 充雄　高井 峰生　山城 登久二　成田 誠之助

早稲田大学 理工学部
{moriya, takai, yamasiro, narita}@narita.elec.waseda.ac.jp

ATMネットワークシミュレータをSCA(Synchronous Conservative Algorithm)を用いて並列計算機上で構築し、このシミュレータを効率的に利用するための最適なモデル分割方法について述べる。並列計算機を効率的に利用するためには各PE(Processing Element)の負荷を平均的にし、かつ内部プロセッサ間の通信回数を最小限に押さえる必要がある。本稿ではこのSCAを用いたシミュレータの実行時間に関わる要因を分析し、ATMネットワークモデルの特徴を考えることで、最も並列効果を得ることが可能となるモデル分割方法について考える。これらの特徴を有効に利用することで大きな並列効果を得ることが期待できる。実際にこのモデル分割方法を用い並列計算機Cenju-3上でATMネットワークの並列シミュレーションを行い評価した。

Developing a Parallel ATM Network Simulator

Takao Moriya　Mineo Takai　Tokuji Yamashiro　Seinosuke Narita

School of Science and Engineering, Waseda University
{moriya, takai, yamasiro, narita}@narita.elec.waseda.ac.jp

A new mapping method for the simulation of ATM network models is described. An ATM parallel networks simulator based on the synchronous conservative algorithm(SCA) has been developed on parallel computers. Efficient use of this simulator requires that the computational load be balanced among PE in such a way that the interprocessor communication be minimized. In this paper, we present a new simple mapping method that considers both the factors of SCA execution time and the features of ATM network models. Analyzing these features leads to a optimistic mapping for parallel simulator using SCA. The method is evaluated to be simulated on a Cenju-3. The results show that utilizing the mapping method realizes efficient parallel simulation.

1 はじめに

ATM(Asynchronous Transfer Mode; 非同期通信モード) ネットワークは、次世代の情報伝送基盤として大変期待されている。しかし、その技術はまだ研究段階にあり、実際の情報伝達手段としての地位を築くには未解決の問題も多い。

これら未解決の問題を分析するためにシミュレーションは有効な手段である [1]。

ところが大規模 ATM ネットワークをシステム分析に必要な精細さでモデル化した場合、シミュレーションは膨大な計算量となり通常の逐次型計算機を用いると実用的な時間でシミュレート結果を得ることは不可能となる。そこで ATM ネットワークのシミュレーションを並列計算機上で行うことで実行時間の短縮を図ることが期待できる。

ATM ネットワークのような離散系システムを並列計算機を用いてシミュレーションを行う場合、解決しなければならない問題がいくつかある [2]。中でも次に上げるマッピング問題と仮想時刻同期手法に関する問題はシミュレーション実行時間に大きく依存しており、最適な解法を用いることが重要となる。マッピング問題とはどの様に対象モデルを各 PE に割り振るかを考える問題であり、仮想時刻同期手法とは各プロセッサ間に割り振ったモデルの状態を矛盾なく並列にシミュレーションを行なうために用いる手法である。これら二つの問題は密接な関係を持っており、用いる仮想時刻同期手法に合わせたマッピングを考えることが実行時間短縮のためには重要となる。

本稿では、最も効果的な仮想時刻同期期手法の一つである SCA(保守的同期アルゴリズム)[3] を用いた ATM ネットワーク並列シミュレータの構成について触れ、さらに SCA のシミュレーション時間に関わる原因と ATM ネットワークモデルの特徴を分析して考えた最適なマッピング手法について述べる。また、このマッピング手法を並列シミュレータに適用し評価を行った。

本論は次のような構成になっている。2 章でこのシミュレータの概要を説明し、3 章で SCA と ATM ネットワークの特徴について考え、これに最適なマッピング手法についての考察を行う。4 章でこの手法を用いて行った並列シミュレーション結果を述べ、5 章でまとめる。

2 ATM ネットワークシミュレータの構成

本シミュレータはイベント処理法式を用いている。イベント処理方式とはモデルを構成する各要素 (以後、コンポーネン) から生起するイベントを生起時刻順に処理していく方式である。それぞれのコンポーネントは決められた処理ルーチンによりイベントの処理を行い、イベントが処理されていくことにより新たにイベントが生起しシミュレーションが進んでいく。

2.1 ATM ネットワークモデル

シミュレート対象モデルである ATM ネットワークモデル [4] は、複数のコンポーネントにより構成される。各コンポーネントの名前と生起する主なイベントは

- ATM Application 「セルの発生」
- B-TE 「セルの受信・接続先コンポーネントの状態把握」
- ATM Switch 「セルの受信・経路判別」
- Physical Link 「セルの受信・接続先コンポーネントの状態把握」

である。なおセルとは上記の各要素内を動き回り実際にデータを蓄え・運ぶものであり、セルが動くことによってシミュレーションが進行する。

シミュレーションの基本的な動きは、セルがある Application から生じ、いくつかの B-TE、Link、Switch を経由して目的の Application に到着するという一連の動作の繰り返しである。

図 1に ATM ネットワークモデルの簡単な例を示す。

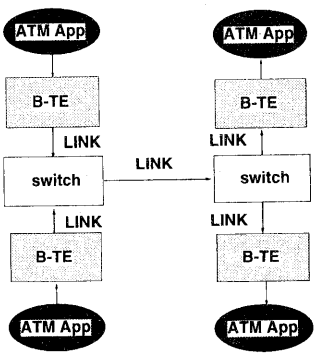

図 1: ATM ネットワークモデルの簡単な例

表 1: ATM ネットワークモデルの特徴

コンポーネント名	遅延時間	イベント処理時間 $[us/call]$
ATM Application	0 or 最小	71.9
B-TE	0 or 最小	106.2
Switch	0 or 最小	107.5
Link	大	69.2

2.2 SCA

並列計算機を用いて離散事象シミュレーションを矛盾なく行うためには仮想時刻同期手法を用いなくてはならない。このシミュレータでは最も速い仮想時刻同期手法の一つである SCA を用いる。次に SCA の仕組みを詳しく見ていく。

SCA の実行は二つの段階からなっている。

第一段階は最小保証時刻を求めることである。現在の仮想時刻を T_c とすると、すべての PE において各 PE が独立して (外部の PE に対して影響を与えないで) イベント処理が行える仮想時刻 $T_c + T_{si}, i = 1, \ldots, n_p$ (但し n_p は PE 数) を求める。この時 T_{si} の最小値 $T_s = min(T_{s1}, \ldots, T_{n_p})$ を求めると、時刻 T_c+T_s の間はすべての PE においてイベント処理しても、外部の PE に影響を与えないことになる。この時刻 $T_c + T_s$ を最小保証時刻と呼ぶ。

第二段階では、イベント処理を行う。第一段階でもとめた 最小保証時刻 $T_c + T_s$ 内においてイベント処理を行い、 $T_c + T_s$ を超えるイベントが発生した時点でバリア同期を行い、他の PE が計算を終了するのを待つ。すべての PE で同期をとり、この際必要ならばイベントの送受信を行い、再び第一段階からこれらの動作を繰り返すことになる。

このアルゴリズムは毎回これら一連の動作を行うので、実装を行うアーキテクチャやシミュレーションモデルのデータ構造によって大きく効率が依存してしまう。これについては次の章で詳しく見ていく。

3 ATM ネットワークシミュレータのマッピング手法

並列シミュレータの実行時間はモデルの形とそれをどの様にマッピングするかによって大きく変ってくる。

本章では、ATM ネットワークモデルの特徴と SCA を用いたシミュレータを実行するのに何に時間がかかっているのかとの分析を行い、SCA を用いた ATM ネットワークモデル並列シミュレータへの最適なマッピング方法について述べる。

3.1 ATM ネットワークモデルの特徴

一般的な離散事象システムと比較して、ATM ネットワークには幾つかの特徴的な点が存在する。これらの特徴を考慮してマッピングにいかすことが重要である。

ATM ネットワークの特徴として特に次の 2 点があげられる。

1、 遅延時間の偏り

遅延時間とは各コンポーネントが次のコンポーネントに対してイベントの発生を行う場合に次のコンポーネントに対してイベントに付加する時間のことである。各コンポーネントを各 PE にマッピングした時に、外部 PE と接続関係にあるコンポーネントの中で最小の遅延時間が前述した T_{si} となる。

いくつかのコンポーネントの遅延時間はリンクの遅延時間に比べてはるかに短い。(表 1) この原因は物理的に考えると、セルが計算機の中を移動する時間に比べて光ファイバーのようなリンクを通過する時間の方が長い、というネットワークの特徴であると考えられる。

このような遅延時間の短いコンポーネントを異なった PE にわけてマッピングすると大変なオーバーヘッド招くことが予想される。遅延時間が小さいと一同期当りに処理できるイベントの数が減り、同期回数も大きく増えることになる。

2、 イベントの発生の偏り

図 2は単位時間当りの各コンポーネントにおけるイベントの発生率の概略を表している。図から分かるようにイベントの発生率に大きな隔たりがある。

これは物理的に考えるとセルの動きに関係している。例えばセルをある端末からある端末まで送信しようとしたとき、セルは必ず Switch を通ることになる。これをモデルで表すと、各 Application から生起したイベントは間接的に接続された Switch においてイベントを

生起することになり、必然的にSwitchやそれに接続するLinkから生起するイベントの数の方が多いことになる。

以上の理由から各コンポーネントをPEにマッピングを行う場合、コンポーネントの数が等しいからといって、各PEにおいて生起するイベントの数が等しいとは言えず、各PEの負荷分散を均等にするためには各コンポーネントにおけるイベントの発生率を考慮する必要がある。なお、表1に示すように各イベントをPEが処理するのにかかる時間も一定ではなく考慮にいれる必要がある。これについて詳しくは次に取り上げる。

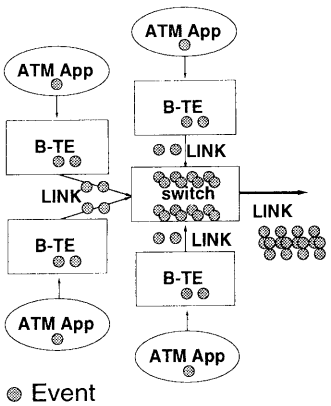

図 2: イベント発生率の概略

3.2 SCAを用いたシミュレーションの実行時間

2.2節からSCAは大きく二つの段階から成り立っていることがわかった。本節ではさらにSCAを用いたシミュレータの実行時間について詳しく見ていく。

SCAを用いたシミュレータの実行時間はつぎの五つの要素から構成される。[5](図3参照)

- C_e; 一つのイベントを処理するのにかかる時間
- C_s; バリア同期を行うのにかかる時間
- C_{ms}; イベントを送信するのにかかる時間
- C_{mr}; イベントを受信するのにかかる時間
- C_w; 同期待ち時間

実際には2.2節からSCAを用いたシミュレータの実行時間は大きく二つの段階にわけられる。一つは同期

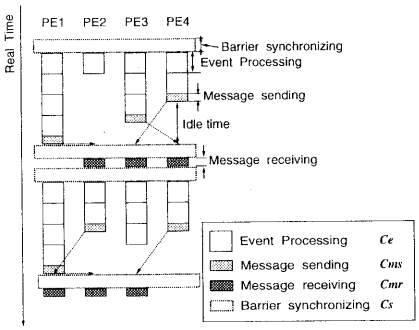

図 3: SCA(保守的同期アルゴリズム)

を一度行うことによって生じる時間C_{t1}であり、もう一つは各イベントを処理するのに必要な時間C_{t2}である。上の五つの要素を用いて表すと、

$$C_{t1} = C_s + C_{mr} + C_{ms} + C_w \quad (1)$$

$$C_{t2} = C_e * N_{per\_sync} \quad (2)$$

で表される。ここでN_{per_sync}は同期一回当りに発生するイベントの数である。上の二式は同期一回当りにかかる時間なので、SCAを用いたシミュレーションを実行したときにかかる時間T_{total}は次式によって表される。

$$T_{total} = (C_{t1} + C_{t2}) * N_{sync}, \quad (3)$$

ここでN_{sync}はシミュレーション中における全同期回数である。

今回シミュレータを実装した並列計算機にCenju-3を用いたため、並列計算用Cライブラリmini-MPIを用いてCenju-3上でこのシミュレータを実装したときのバリア同期時間, イベント処理時間 , メッセージ送受信にかかる時間 の値の平均値を表2に示す。なお、C_eはあるネットワークモデルを実行したときに生起したすべてのイベントを処理するときにかかった時間の平均値であり、C_{ms}, C_{mr}はfloat型のデータを最大5000個送受信するのにかかった時間の平均値である。

これらの値の特徴として、同期一回当りにかかる時間C_{t1}がイベントを一つ処理するのにかかる時間C_eに比べて大きいということが言える。本稿ではこれらの値をもとに話を進め、表2に示されるような値を持つアーキテクチャ上で特に効果をあげれるようなマッピング方法を次節で考える。

表 2: SCA 各処理における平均時間

処理名		時間 $[us/call]$
バリア同期	C_s	694
イベント処理	C_e	79
メッセージ受信	C_{mr}	38
メッセージ送信	C_{ms}	56

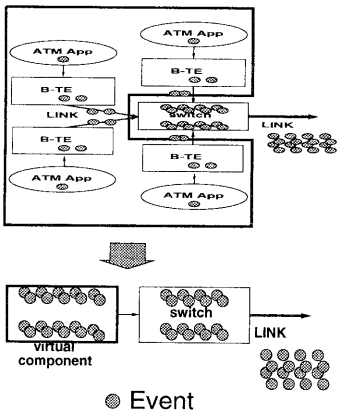

図 4: 仮想的なコンポーネントの生成

3.3 マッピング方法

これまでに見てきたこのシミュレータの特徴を考慮して、このATMネットワークシミュレータに最適なマッピング方法について考える。

マッピングは二つの段階によって構成される。

1. 3.1節の問題を解決するために図4のように遅延時間が小さく、かつ一つのコンポーネントに流れ込むコンポーネント群を一つにまとめて仮想的なコンポーネントを作る。

この作業により全体のコンポーネントの数も少なくなり、分割する場合の組合わせの数を減らすことにもなる。また各コンポーネントから単位時間当りに発生するイベント数の偏りが改善される。

2. 次に1.でまとめた仮想的なコンポーネントを含めコンポーネントを各PEに割り振る。

モデルをN個のPEに割り振るとすると、モデルをN等分しその時の実行時間を3.2節で考察した T_{total} を用いて予測する。これを何通りか繰り返し、最も短い予測実行時間をになったものをマッピングに用いる。

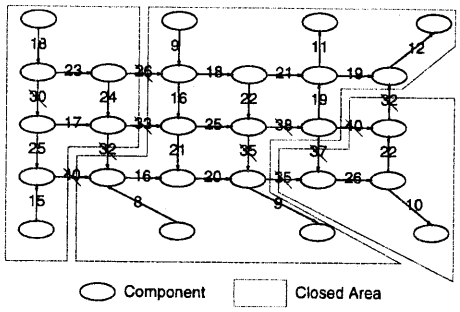

図 5: モデル分割の例

図5にこの分割方法の例を示す。図中の数字は各コンポーネント間に生じる遅延時間である。

次にこのアルゴリズムについて図5を用いながら詳しく見ていく。各PE間の遅延時間が大きいほど同期回数が少なくなるので、遅延時間の大きい値からカットしていき、ある閉じた領域 (closed area) ができるまで繰り返す。N等分するのであれば、これを繰り返しN個の閉じた領域を作る。(図では3個の閉じた領域)

閉じた領域がN個できると (3)T_{total} を利用して予測実行時間 T_total() を調べる。(下参照)

さらに閉じた領域をN個以上作り、N個以上の閉じた領域を組み合わせてN個の閉じた領域に組み合わせなおし予測実行時間をを求める。何回かこれを繰り返し、最も予測実行時間が短いものをマッピングに利用する。

これらの動作を関数的に表したものを次ページに示す。(図6参照)

予測実行時間 *T_total()*

N個の閉じた領域の中で最もイベントの発生率が高い閉じた領域に着目して予測実行時間の計算を行う。この閉じた領域においては $C_w = 0$ と見なすことができ、T_{total} を用いて予測実行時間 T_total() は次のように表される。ただし、各コンポーネントにおける単位時間当りのイベントの発生数は分かっているものとする。

$$T\_total() = (C_s + C_{mr} + C_{ms} + C_e * N_{per\_sync}) * N_{sync}$$

ここで、同期回数 N_{sync} は (全シミュレーション時間)/(カットした遅延時間の最も小さい値) であり、N_{per_sync} は一同期当りに発生するイベントの数である。C_s, C_{mr}, C_{ms}, C_e の各値はアーキテクチャにより異なるが、今回は表2の値を用いた。

```
// N_delay  ; 遅延時間の数
// T_delay_i; 遅延時間
//            (i = 0,1,..,N_delay)
//          = {最大値,....,最小値}
// T_total  ; シミュレーション予測実行時間
// T_expect ; ある一定時間
do{
  for (i = 0 ; i < N_delay ; i++){
    cut off the line of T_delay_i;
    if(a closed area is made){
      if(the number of closed areas <= N)
        break;
      else
        add up closed areas to N;
        break;
    }
  }
  Calculate T_total();
}while(T_total > T_expect)
```

図 6: 分割アルゴリズム

4 実行結果

性能評価のため、実際に前述の方法を用いたマッピングを行い、分散メモリ型並列計算機 Cenju-3 上で実際にシミュレーションを行った。8 台の PE を用いてランダムにマッピングしたものと今回提案したマッピング手法とを比較した。なお、今回用いたランダムマッピングとは、乱数的にコンポーネントを各 PE に割り振る方法である。

モデルはコンポーネント数 200 の ATM ネットワークモデルを利用した。

表 4にこれらの結果を示す。

今回提案したマッピング手法における実行結果は 1 PE を用いたときの 2.58 倍速くなった。

また、ランダムマッピングを用いた場合非常に悪い結果となっており、同期回数がパフォーマンスに大きく影響していることがわかる。

表 3: 実行時間の比較

マッピングの種類	実行時間 [*sec*]	同期回数
ランダムマッピング	3124.16	4035793
提案したマッピング	265.97	122101
1-PE	687.56	-

5 まとめ

SCA を用いた ATM ネットワークシミュレータを構築し、このシミュレータのためのマッピング手法について考えた。並列効果は現れたが台数効果に比例して効果があらわれたという訳にはいかなかった。ただ、SCA を用いたシミュレーションは同期回数により大きく性能が左右され、モデルの形によってはどんなに最適ににマッピングできても遅延時間が小さくしかとれなく必然的に同期回数も増え、並列効果があまりできない場合も考えられる。

今後は最適化アルゴリズムを用いたマッピング等他のマッピング手法との比較や新しい仮想時刻同期手法の導入も考えていきたい。また今回は Cenju-3 上でのシミュレータについての考察であったけれども、他の並列計算機上では同期時間やプロセッサの速度が異なるといった面があり異なったアーキテクチャ上での評価も検討していく。

参考文献

[1] Canadian universities (Calgary, Alberta and Saskatchewan) The Telesim Project. http://www.wnet.ca/telesim.

[2] R.M.Fujimoto.: Parallel Discrete Event Simulation. Communications of ACM, vol.33, No.10. October 1990

[3] Nicol.D.M.: The Cost of Conservative Synchronization in Parallel Discrete Event Simulations. Journal of the ACM 1993

[4] N.Golmie, A.Koenig, D.Su.: The NIST ATM Network Simulator:User Manual. NIST Internal Report, 1995

[5] M.Takai, T.Yamasiro, S.Narita: Performance Estimation of Parallel Discrete Event Simulation Using Synchronous Conservative Algorithm. *Trans. of IEICE*, Vol. J80-D-I, No. 3, March 1997