システムソフトウェアと
オペレーティング・システム 78－22
（1998．5．8）

並列離散事象シミュレーションにおける適応的同期プロトコルの評価

山口 佳紀 本多 弘樹 弓場 敏嗣

電気通信大学大学院情報システム学研究科

並列離散事象シミュレーションの実行速度向上のためには、適切な同期プロトコルの選択が不可欠となっている。本論文では、事象処理が連続的に成功し続けている状態が保たれるように、過去の事象処理情報を用いて適応的に次の事象を処理するか否かを定める適応的プロトコルを提案する。また、実機における有効性の範囲を確認するために2次元トーラス接続された閉鎖的待ち行列システムをシミュレーション対象として同期プロトコルの評価を行う。

Evaluation of an adaptive synchronization protocol for parallel discrete event simulation

Yoshiki YAMAGUCHI, Hiroki HONDA, and Toshitsugu YUBA
Graduate School of Information Systems
The University of Electro-Communications
E-mail: {yamaguchi, honda, yuba}@yuba.is.uec.ac.jp

For high-speed simulation, it is needed to determine appropriate synchronization protocols for parallel distributed event simulation. This paper proposes a new adaptive synchronization protocol. Utilizing adaptive information, this protocol works so that events are processed as safely as possible. We evaluate effectiveness of this protocol on a closed queuing system model with two dimensional mesh topology on a parallel computer.

1 はじめに

高速ネットワークや高性能計算機を技術基盤として高度分散システム環境を実現するとき、離散事象シミュレーション技術を用いた設計段階におけるシステムの解析・評価が設計が不可欠になっている。近年の急速な技術の進歩に伴い、高速実行可能な離散事象のシミュレーション技術の確立が重要となってきている。この要求を満たすため、並列計算機や分散システム上で実行される並列離散事象シミュレーション (Parallel Discrete Event Simulation: PDES) に関する研究 [3] が行われている。その中で特に、シミュレーション同期プロトコルは、逐次で行うシミュレーションと同じシミュレーション結果をPDESで得るために必要となる。

PDESでは、事象の発生時刻をタイムスタンプとしてメッセージに事象内容とともに付加し、メッセージ交換を行うことによって複数の要素プロセッサに割り当てられた論理プロセス間で事象処理を並列に行う。各論理プロセスにおいて、常に最小のタイムスタンプをもつメッセージの事象から処理を行わないと、事象間の因果関係を破る因果律エラーが生じる。そのため各論理プロセス間での事象処理の適切な同期を保証するプロトコルが必要となる。

同期プロトコルは、保守的プロトコル、楽観的プロトコルに大きく分類できる [3]。それぞれの同期プロトコルには適したシミュレーション対象モデルがあるため、シミュレーション対象モデルに応じた同期プロトコルの選択がシミュレーション高速実行のために必要となる。しかし、シミュレーション対象モデルのもつ性質やシミュレーション実行時の振舞いは、時間とともに動的に変化することが多いため、適切な選択は難しい。そこで、

その性質や振舞いをパラメータ化し、動的に同期プロトコルの機能を変化させる適応的プロトコルについての研究が近年注目をあびている [3][8]。

本稿では、楽観的プロトコルをベースに楽観処理が成功し続けるような機構をもつ適応的プロトコルの評価を行った。以降、2節で適応的プロトコルについて述べ、3節で、提案する適応的プロトコルについて説明する。待ち行列網をシミュレーション対象としたシミュレーション環境および適応的プロトコルを用いたその評価について4、5節で述べ、6節で提案プロトコルの有効性の検討と今後の課題を述べる。

2 適応的プロトコル

シミュレーション高速化のために望まれる同期プロトコルは、シミュレーション実行状態に応じて、保守的プロトコル、楽観的プロトコルのどちらとしても振舞えることである。この性質をもつためには過去の情報より適応的に、保守的プロトコルに楽観性を加えたり、逆に楽観的プロトコルの楽観性を制限するなどの機能が必要となる。以下に現在までに提案されている、いくつかの適応的プロトコルを示す。

事象処理毎に、論理プロセス間の入出力リンクの情報をもとに実時間のブロッキングを行う局所適応的プロトコル [4] が提案されている。ブロッキングのタイミングは、次に処理すべき事象を含むメッセージのタイムスタンプが、到着した入力リンクのタイムスタンプの平均的な増加範囲内かどうか判断し、増加範囲外だった場合のみ事象処理が待たされる。

すべての論理プロセスと入出力リンクの中に存在するメッセージのタイムスタンプの最小値である大域仮想時刻 (Global Virtual Time: GVT) は、シミュレーション終了処理やメモリ管理のために必要となる。常に GVT に近い時間でシミュレーションを行うと因果律エラーの発生が少なくなる。この GVT の計算用に専用のハードウェアを用意し、論理プロセス毎に GVT からの局所仮想時刻の差に比例した実時間だけ事象処理の実行待ちをする NPSI プロトコル [7] が提案されている。

また、メモリ使用量の観点から適応的に事象処理を行う適応的メモリ管理プロトコル [2] も提案されている。因果律エラー時のロールバック処理のオーバヘッドを減らすことだけでなく、GVT 計算のオーバヘッド、GVT より小さな仮想時刻の状態の削除なども考慮し、使用可能メモリ量から事象処理実行の制御を行う。

これら従来の適応的プロトコルにおいては、因果律エラーを引き起こした事象の情報も適応情報として用いている。また、1回の因果律エラーの発生によってその後のシミュレーション状態が変化し、因果律エラー発生以前の情報があまり役にたたない状況になる可能性がある。

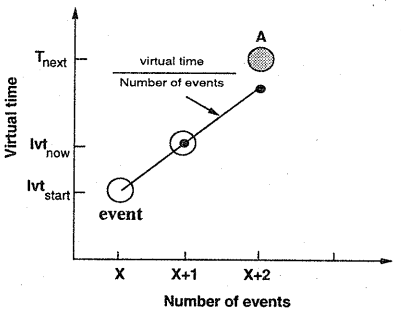
3 サクシード法

楽観的プロトコルにおけるシミュレーション性能に影響を及ぼす大きな要因は、ロールバック処理のコストである。ロールバック処理のコストは、1回のロールバック処理における実時間のオーバヘッドにロールバック処理の発生頻度を掛け合わせた、実時間量である。1回のロールバック処理のオーバヘッドを小さくするために、様々な研究が行われている [3]。例えば、ロールバック処理において同じ結果をもたらす可能性のある事象処理の再実行をわざと遅らせることなどが行われている。

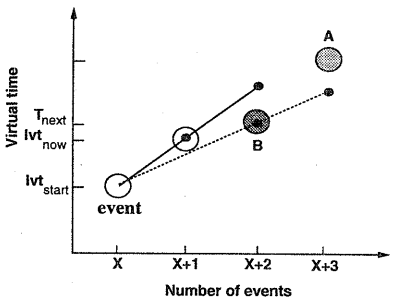
シミュレーション対象モデルがもつ並列性を活かすために楽観処理を行ったものが、結果的にロールバックの原因となることが多い。そうした場合、楽観処理を行う回数を制限することでロールバックの発生頻度を減らすことができる。

本論文で提案する適応的プロトコルは、楽観的プロトコルのタイムワープ法 [5] をベースに過去のシミュレーション実行状態を用いて楽観処理を行うか否かを判断する。それによって、ロールバックの発生頻度を減らし、PDES の実行時間の高速化を狙う。

図1に示すように、次に処理対象となる事象を楽観処理するか否かを、ロールバックした後から現在までの仮想時刻の増加値と連続してロールバックなしで事象処理を行えた数を基準に判断する。



(a) realtime = t



(b) realtime = t' (>t)

図1　事象処理開始の判断

図1(a)において、実時間 t で次の処理対象である事象Aを処理するかどうかの判断を行う。実時間 $t'(>t)$ において、事象の開始仮想時刻が事象Aよりも小さな値をもつ事象Bが到着し、因果律エラーを起こさず事象処理が行うことができる(図1(b))。

具体的には、ロールバック処理終了直後の論理プロセスの局所仮想時刻 (Local Virtual Time: LVT) を lvt_{start} 、ロールバックを起こさずにシミュレーションを続けることができている時の現在のLVTを lvt_{now}、lvt_{start}から lvt_{now}までに処理した事象数を n とする。処理対象とする事象のタイムスタンプを T_{next}とすると、式(1)を満たすときのみ事象処理を行い、それ以外のときは事象処理開始を延期する。

$$T_{next} \leq \frac{lvt_{now} - lvt_{start}}{n} + lvt_{now} \qquad (1)$$

これは、ロールバックを生じさせずに事象処理を行っている傾向が、式(1)の条件が満たされる場合、保たれるであろうという予測に基づいている。連続して成功することを目指す意味で、このプロトコルをサクシード法と呼ぶこととする。

ここで問題となるのは、時間が経過すると必ず式(1)の条件が成立するとは限らないことである。(図1(a)の状況が長く続く場合も考えられる。)そのため、ある実時間のタイムアウトを設け、タイムアウト後は楽観的に次の事象処理を行う。タイムアウトの時間は、予測が外れた場合、シミュレーション実行速度を遅くしてしまう可能性があるので適切な値を与える必要がある。

図2にサクシード法の処理手順を疑似コードで示す。図中のWait_lockは事象を延期するかどうかのフラグである。

```
Initialize(ENDTIME, NO_PROC_EVENT, TIMEOUT);
Wait_lock = off;
/* メイン */
while(gvt <= ENDTIME){
  Check_next_message(timestamp(next), n);
  if(Wait_lock == off){
    /* 因果律エラー発生か? */
    if(timestamp(next) < LVT or
        next == antimessage){
      Rollback_process; /* ロールバック処理 */
      Generate_anti_message;
      lvt_start = Rollbacked_LVT;
      n = 1;
    }else{
      /* 事象処理開始の判断 */
      Ideal_time = (lvt_now - lvt_start) / n
                 + lvt_now;
      if(timestamp(next) > Ideal_time){
        Wait_lock = on;
        Wait_timer_start;
      }else{
        lvt_now = timestamp(next);
        Normal_event_process;
        n += NO_PROC_EVENT;
      }
    }
  }else{
    /* 事象処理を延期している間に条件を満たすか? */
    if(timestamp(next) <= Ideal_time){
      Wait_lock = off;
    }else if(Wait_timer >= TIMEOUT){
      Wait_lock = off;
    }
  }
  Send_message;
  Calculate_gvt;
}
```

図2　サクシード法

4 待ち行列モデルのシミュレーション

サクシード法の有効性を評価するために、楽観的プロトコルのタイムワープ法との比較を行う。

シミュレーション対象として、4×4の16個のサーバで構成された2次元トーラス接続の閉鎖的待ち行列システムを用い、各サーバを1つの論理プロセスに割り当て、モデル化した（図3）。

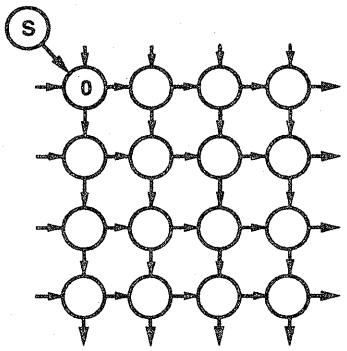


図3 2次元トーラス接続の閉鎖的待ち行列システムモデル

このモデルは、4×4の16個のサーバに客が到着し、あるサービス時間を経た後、確率0.5で選択された2つの出力枝のどちらか一方から出力される閉鎖的待ち行列システムをモデル化したものである。モデル化に際して、各サーバをそれぞれを1つの論理プロセスに割り当て、客の到着などの事象をその事象開始時刻とともにメッセージに付加する。サービス時間は事象処理時間に対応している。

このようなトーラス接続の閉鎖的待ち行列システムモデルを用いたシミュレーションは同期プロトコルの評価が行いやすいということから各種行われている [4][6]。

初期条件として一定数のメッセージが図3のプロセスSからプロセス0にに到着することとする。シミュレーションは、メッセージに含まれている事象処理終了後、新たな事象を含んだメッセージを1つ生起する。生成されたメッセージは、出力リンクの一方から隣接するプロセスに出力する。以後、GVTが終了時刻になるまで同様のことを繰り返す。

シミュレーション実行中に、このモデルに存在しているメッセージの総量は初期条件で与えられているプロセスSからプロセス0に送出されたメッセージ量と等しく、この量を全論理プロセス数(16)で割った平均的に1つのプロセスがもつメッセージ量をメッセージ密度と定義する。

分散記憶型並列計算機Cenju-3上に、C言語とMPIライブラリを用いて、タイムワープ法およびサクシード法を実装した。

論理プロセスの要素プロセッサへの割り当ては1対1とし、16台の要素プロセッサを使用した。

5 プロトコルの有効性評価

サクシード法の有効性評価を以下の項目について行った。

- メッセージ密度とロールバック発生頻度の関係。
- メッセージ密度とロールバック距離の関係。
- メッセージ密度とシミュレーション実行時間の関係。

事象処理の実行を延期することによってもたらされる効果として、因果律エラーとそれに伴うロールバックの発生頻度の減少が考えられる。メッセージ密度が増加すると、待ち行列モデル内の総メッセージ数も増加するため、因果律エラーの発生状況も変わる。そこでメッセージ密度に対する、タイムワープ法およびサクシード法を用いた時のロールバックの発生頻度を調べる。また、ロールバック処理のコストは巻き戻す事象の数に比例することから、1回のロールバックによって生じる、事象の巻き戻し数（ロールバック距離）も調べる。

ロールバックの発生頻度を減らすことで、シミュレーション実行時間を短くすることができると考えられるので、メッセージ密度に対するシミュレーション実行時間を調べる。

シミュレーションは、事象処理を平均40、分散10（単位は仮想時間）のアーラン分布のサービス時間かかるものとし、GVTが終了時刻100,000をこえるまで行った。なお、メモリ領域の有効利用を可能とするGVT計算処理によるシミュレーション実行時間に与える影響を少なくするために、GVT計算は各論理プロセスのLVTが終了時刻近くになった時のみ行った。[1]

タイムワープ法とサクシード法の各論理プロセスあたりのロールバックの平均発生頻度と、その発生によって巻き戻される事象の数（平均ロールバック距離）を図5、図6に示す。

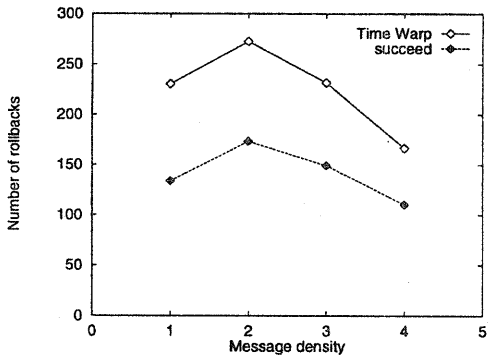


図5　平均ロールバック発生頻度の比較

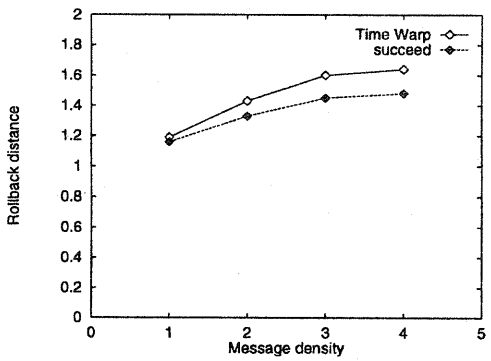


図6　平均ロールバック距離の比較

[1]トーラス接続で事象処理の内容が各論理プロセスでほぼ等しい性質があり、LVTの時間進行の著しい差がないことから可能である。

図5より、サクシード法がタイムワープ法に比べて平均ロールバック発生頻度を最大43%(MD=1)削減できていることがわかる。また、図6で示すように、平均ロールバック距離もタイムワープ法に比べ短くすることができている。

図7にタイムワープ法、サクシード法のメッセージ密度に対する実行時間を示す。

最大7.7%(MD=1)とわずかであるが、サクシード法はタイムワープ法に比べて実行時間が短くなっている。

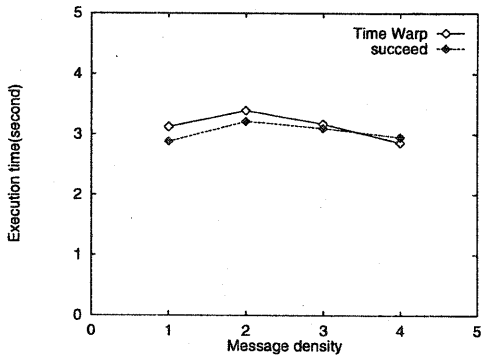


図7　実行時間の比較

本シミュレーション対象モデルでは、メッセージ密度を大きくすると（5以上）メッセージがプロセス0に集中してしまう傾向が見られる。その結果、全体の実行時間はプロセス0の処理時間に依存してしまうことが確認された。そこで、各プロセスが初期条件としてメッセージ密度と同じ数のメッセージをもった状態から開始する別の対象モデルにおいてシミュレーションを行った。ロールバックの発生頻度、ロールバック距離、実行時間の結果は、メッセージ密度が4まで先のシミュレーションモデルと同じ傾向を示した。しかし、メッセージ密度が5を越えるとシミュレーション実行時間はタイムワープ法に比べて短くならなかった。また、ロールバック処理によって生じる、取消メッセージの受信数がメッセージ密度が大きくなると減る現象が見られた。

6 議論

適応的な事象処理開始の延期を行うことで、タイムワープ法に比べて、ロールバックの発生頻度を削減することができた。しかし、シミュレーション実行時間では、メッセージ密度が小さな場合にわずかな速度向上が見られただけであった。これは、事象処理開始の延期による実時間でのオーバヘッドが大きくなっていることが考えられる。現在のプロトコルではタイムアウト時間を静的に決めているので、1回のロールバック処理のオーバヘッドが小さいと、事象処理延期のためのオーバヘッドの方が実行時間に影響を及ぼす。実際、今回評価したモデルでは、平均ロールバック距離も2程度と小さく、1回のロールバック処理のコストが実行時間にあまり影響を与えない。

ロールバック距離が大きくなる状況がメッセージ密度を増加させたとき、現れた。しかしメッセージ密度の増加により、入力リンク内での因果律エラーを生じさせる可能性のあるメッセージが取消メッセージによって消去される確率が増えたため、逆にロールバックの発生頻度が減っていた。その結果、この状況において、ロールバック処理のコストが実行時間に与える影響は少なく、事象処理延期のためのオーバヘッドが顕著になり、サクシード法の有効性が現れなかったと考えられる。こうした現象は、通信速度や事象処理実行速度のバランスによっても生じる。

以上の結果から、過去の事象処理の情報から適応的に事象処理延期の判断するとともに、タイムアウト時間の調整もシミュレーション実行の実機環境に応じて、適応的にすることが必要である。具体的には、事象処理延期の際、楽観的に処理しても正しかった状況が続いた場合、タイムアウト時間を短くすることが考えられる。

また、プロトコルの優劣がシミュレーション対象モデルに依存するケース（潜在的に因果律エラーが発生する状況が少ないなど）も考えられる。今後はプロトコルの改良とともにロールバックの巻き戻しの距離が長く、ロールバックのコストが大きいと考えられている個人通信サービスの利用環境のシミュレーション [1] などに、本提案プロトコルを実装し、他の適応的プロトコルとの評価を行う予定である。

謝辞　並列計算機 Cenju-3 の利用環境をご提供頂いた、NEC C & C 研究所並列処理センタに感謝致します。本研究は郵政省電気通信フロンティア研究の一環として（財）テレコム先端技術研究支援センターからの受託研究として行った。

参考文献

[1] Carothers, C. D., et al., "Distributed simulation of large scale PCS networks," Proc. the 2nd International Conference on Modeling Analysis, and Simulation of Computer and Telecommunication Systems(MASCOTS'94), pp.2-11, Jan. 1994.

[2] Das, S. and R. M. Fujimoto, "An Adaptive Memory Management Protocol for Time Warp Parallel Simulation," Proc. the 1994 ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems, pp.201-210, May 1994.

[3] Ferscha, A., "Parallel and distributed simulation of discrete event systems," in A. Y. H. Zomaya (ed.), *Parallel and Distributed Computing Handbook*, Chapter 35. McGraw-Hill, 1995.

[4] Hamnes, D. O. and A. Tripathi, "Evaluation of a Local Adaptive Protocol for Distributed Discrete Event Simulation," Proc. the 1994 International Conference on Parallel Processing, Vol.3, pp.127-134, Aug. 1994.

[5] Jefferson, D. R., "Virtual Time," ACM Transactions on Programming Languages and Systems, Vol. 7, No. 3, pp.404-425, July 1985.

[6] Prakash, A. and R. Subramanian, "An Efficient Optimistic Distributed Simulation Scheme based on Conditional Knowledge," 6th Workshop on Parallel and Distributed Simulation(PADS92), Vol. 24, No. 3, pp.85-94, Jan. 1992.

[7] Srinivasan, S. and P. F. Reynolds, "NPSI Adaptive Synchronization Algorithms for PDES," Proc. the 1995 Winter Simulation Conference, pp.658–665, Dec. 1995.

[8] 山口 佳紀, 大澤 範高, 弓場 敏嗣, "実時間履歴情報を用いた並列離散事象シミュレーション," 産業システム情報化研究会, 電気学会, IIS-97-8, pp.41–46, Jan. 1997.