タイムワープによる並列論理シミュレータのコンパクトな実現手法*

6H－4

松本 幸則 †
mat@tsukuba.rd.sanyo.co.jp
三洋電機(株) 東京情報通信研究所 §

瀧 和男‡
taki@seg.kobe-u.ac.jp
神戸大学工学部¶

1　はじめに

論理シミュレーションは多大な計算時間を要するLSI設計工程であり，高速化が望まれている．この背景から，並列計算機を利用した高速論理シミュレータの研究が盛んになってきた．論理シミュレータの並列化手法としては，Chandy-Misraの方法[3]とともにタイムワープ機構[2]が注目されているが，多くの実現例ではChandy-Misraの方法が用いられている．この主な理由として，タイムワープ機構の組み込みが繁雑と考えられていることがあげられる．本稿では，タイムワープ機構を用いた論理シミュレータ実装の容易化を目的とし，そのコンパクトな実現手法について述べる[1]．

2　タイムワープ機構の概要

時刻印をもつメッセージ通信により，複数のオブジェクトが次々と状態を変えていくモデルを考える．ここで，オブジェクトは時刻順にメッセージを評価する必要があるとする．

タイムワープ機構[2]では，メッセージを時刻順に処理できるであろうという楽観的予測に基づき，各オブジェクトが非同期に動作する．しかし，実際には時刻順にメッセージを処理できない場合が存在する．このような場合，履歴を巻き戻す(ロールバック処理)ことで，そのメッセージが正しい順序で処理できる状態を再現した後，処理のやり直しをおこなう．また，すでに誤って送ったメッセージがある場合，これらを取り消すアンチメッセージを送る．

3　タイムワープの論理シミュレータへの適用

タイムワープ機構を論理シミュレータに適用した場合，ゲート等の素子をオブジェクトに対応付けることが自然である．この場合，メッセージ処理可能なオブジェクトが一つのプロセッサに複数存在しうる．したがって，メッセージ処理順序を適切に制御するスケジューラを各プロセッサに一つづつ配置する必要がある．

また，ロールバック処理のオーバヘッド削減はタイムワープ機構の効率的な動作実現に有効である．例えば，以下に示すアンチメッセージ削減処理は，このオーバヘッド削減のための一つの方法である．

今，オブジェクト間の通信路上において，メッセージの順序が保存される環境を仮定する．この環境でロールバックが発生した時，一連の取り消すべきメッセージに対し，それらの中で最小時刻印を持つメッセージに対応したアンチメッセージ一つのみ(または最小時刻印以下の時刻をもつメッセージ m を新たに出力することが分かっていれば，m のみ)を送信する．受信側では，同一通信路上で受信したメッセージのうち，アンチメッセージ(または m)到着以前に受信し，かつアンチメッセージ(または m)の時刻以上の時刻印を持つメッセージ一連が取り消し対象となる[4, 5]．

*A Compact Implementation Scheme of A Parallel Logic Simulator Based on Time Warp

†Yukinori MATSUMOTO

‡Kazuo TAKI

§Tokyo Information & Communication Research Center, SANYO Electric Co., Ltd.

¶Faculty of Engineering, Kobe University

[1]本研究は筆者らがICOT在籍中におこなったものである．

4　論理シミュレータのコンパクトな実現手法

4.1　素朴に実現した場合

スケジューラも考慮して論理シミュレータを素朴に実現した場合の，スケジューラとオブジェクトの間のメッセージの流れを図1に示す．

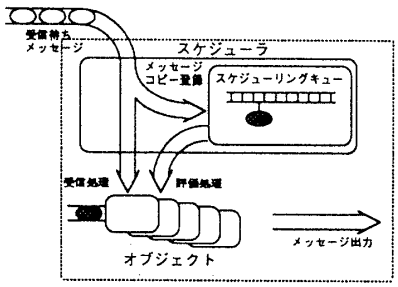

図1: スケジューラとオブジェクト間のメッセージの流れ(素朴な実現方法)

スケジューラは，受信待ちメッセージがあれば，それを受信した後，目的地オブジェクトに送って受信処理をさせる．また，そのコピーをスケジューリングキューに登録する．受信待ちメッセージが無くなって始めて，登録中のメッセージコピーを時刻の小さいものから取り出し，それに対応するゲートに評価処理させる．

オブジェクトでは一つのメッセージに対し，二つの処理すなわちメッセージ受信処理と評価処理を，異なるタイミングでおこなう必要がある．それぞれの処理の概略は以下の通りである．メッセージ受信処理では，受信メッセージを受信キューに入れる．ここでは，受信メッセージを即座に評価しない．また，アンチメッセージ受信時，および遅れメッセージ到着時には，メッセージの取消処理やロールバック処理をおこなう．一方，メッセージ評価処理では，スケジューラからの評価指示に対し，受信キューの中に対応するメッセージがあれば[2]，それを取り出して評価する．

4.2　コンパクトな実現手法

未評価メッセージは全てスケジューラのみが管理し，オブジェクトでは，必要な処理全てをメッセージ評価時点におこなう手法を示す．この場合のメッセージの流れを図2に示す．

スケジューラは，受信待ちメッセージがあれば受信し，スケジューリングキューに登録する．受信待ちメッセージがなくなった時点で，登録中のメッセージのうち最小時刻のものを対応ゲートに送信し，評価処理させる．

本手法では，オブジェクトにおける未評価メッセージ操作が不要であるため，オブジェクトにおける処理記述が簡潔になるとともに，メッセージの流れも極めて単純になる．ただし，アンチメッセージがスケジューラに到着した時には，登録中のメッセージのうち取り消されるべきものをすべて見つけて無効化す

[2]評価対象メッセージが受信キューに無い場合も存在する．これは，受信キューのメッセージが取り消された場合でも，スケジューラ内のコピーは消去しないためである．コピーの消去処理は高コストである．

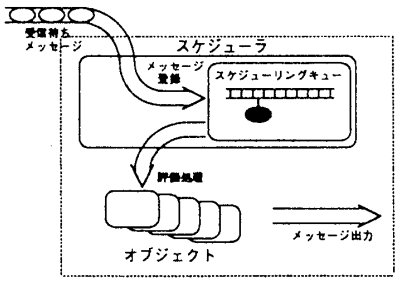

図 2: スケジューラとオブジェクト間のメッセージの流れ (コンパクトな実現方法)

初期化処理
$Lvt := 0$
$Count_{out} := 0$
$Count_{in}(i = 0, \text{入力線数}) := 0$
メッセージ評価時の処理
$l :=$ 評価対象メッセージの通信路 ID
IF 評価対象メッセージのシリアル番号 $> Count_{in}(l)$
THEN
  $OutMsg := \phi$
  $Count_{in}(l) :=$ メッセージのシリアル番号
  IF 評価対象メッセージの時刻印 $\leq Lvt$
  THEN 履歴巻き戻し処理†
    IF アンチメッセージ送信必要
    THEN $OutMsg :=$ アンチメッセージ
  ENDIF
  メッセージ評価‡
  $Lvt :=$ 評価メッセージの時刻印
  IF 出力値が変化
  THEN $OutMsg :=$ 出力メッセージ
  ENDIF
  IF $OutMsg \neq \phi$
  THEN $OutMsg$ の番号を $Count_{out}$ とし，出力
    $Count_{out} := Count_{out} + 1$
  ENDIF
ENDIF

† 1: 評価メッセージと同じ通信路上で受信したメッセージのうち，評価メッセージ以上の時刻印を持つものを消去，2: 評価メッセージと異なる通信路上で受信したメッセージのうち，評価メッセージより大きい時刻印を持つものをスケジューラに再登録，の処理.
‡ 評価対象がアンチメッセージであれば，出力値は無変化とする.

図 3: オブジェクトにおけるメッセージ評価時の処理

る処理が必要になる．これは，未評価メッセージは全てスケジューラに管理されるためである．この処理を素朴に実現した場合，スケジューラに登録中のメッセージの中から無効化対象を見つけ出し，消去する処理が必要であり，記述量，オーバヘッドともに大きい．

しかしながら，以下の方法によってメッセージ評価時に無効化処理がおこなえる．スケジューラは，同一時刻のメッセージに関してはスタック (LIFO) を用いて登録をおこなう．また，メッセージ送信時にオブジェクト毎に，昇順にシリアル番号付けをおこなう．評価時には，評価対象メッセージのシリアル番号と，同じ通信路上のメッセージのうち既に評価済みのものの最大番号を比較することで，評価 / 無視の判断をおこなう．

この判断処理，および第 3 節で述べたアンチメッセージ削減処理を組み込んだ場合の，オブジェクトにおけるメッセージ評価処理内容を図 3に示す．また，図 4にメッセージ無効化処理の例を示す (図 4では，理解容易のため時刻・シリアル番号に具体値を入れている．)．今，オブジェクト O' から O への通信路 l 上を，X, Y, Z, A の順にメッセージが送られた場合を考える．ここで X, Y, Z は通常のメッセージであり，A は Y, Z に対するアンチメッセージとする．この時，メッセージ m のシリアル番号を $N(m)$ と表記すると，$N(X) < N(Y) < N(Z) < N(A)$ である．X, Y, Z, A は一旦全てスケジューラに登録されたのち，X, A, Y, Z の順に評価される[3]．A 以降に送信されたメッセージがない時点で X の評価処理をおこなう場合を考えると，X 評価処理前には $Count_{in}(l) < N(X)$ である．したがって X は正しく評価される．一方 A, Y, Z それぞれの評価時点では，$Count_{in}(l) = N(X) < N(A)$，$Count_{in}(l) = N(A) > N(Y)$，$Count_{in}(l) = N(A) < N(Z)$ となる．したがって Y, Z は無効とみなされる．

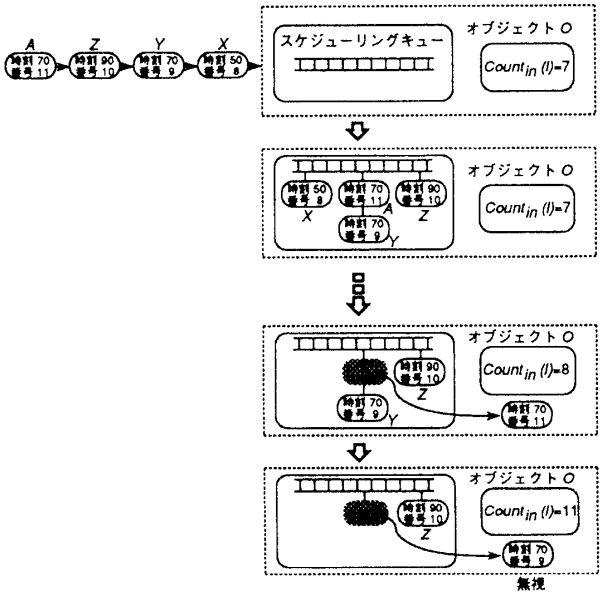

図 4: メッセージ無効化の例

5 おわりに

タイムワープ機構による論理シミュレータをコンパクトに実現する手法を提案した．本手法は，全ての未評価メッセージをスケジューラのみで管理し，オブジェクトでは受信メッセージを即座に評価する単純な方法であるため，コンパクトな記述が可能となる．また，未評価メッセージの取消処理についても，メッセージ評価時にシリアル番号比較をおこなうのみでよいため，記述量，オーバヘッドともに小さい．さらに，ロールバックオーバヘッドを削減する工夫もコンパクトに組み込むことができている．

今後の課題として，番号オーバフロー時の対処がある．

参考文献

[1] Fujimoto, R. M.: "Parallel Discrete Event Simulation," *Communications of the ACM*, Vol.33, No.10, pp.30–53, 1990.

[2] Jefferson, D.R.: "Virtual Time," *ACM Transactions on Programming Languages and Systems*, Vol.7, No.3, pp. 404–425, 1985.

[3] Misra, J.: "Distributed Discrete-Event Simulation," *ACM Computing Surveys*, Vol.18, No.1, pp.39–64, 1986.

[4] 福井：バーチャルタイムアルゴリズムの改良, 情報処理学会論文誌, Vol.30,No.12, pp.1547–1554, 1989.

[5] 松本, 瀧：バーチャルタイムによる並列論理シミュレーション, 情報処理学会論文誌, Vol. 33, No. 3, pp. 387–395, 1992.

[3] 同一時刻のメッセージはスタックを使って管理されるため，Y より先に A を評価する.