並列離散事象シミュレーション評価環境 VPSE とタイムワープ手法の性質検討

奥村 勝[†,☆] 末吉敏則[††]

シミュレーション対象の持つ潜在的な並列性を活用して処理の高速化を図る PDES では，その対象の特性に応じて実装に関連する手法やアルゴリズムを適切に選択する必要がある．また，PDES の実装が計算機システムの環境やプログラミング手法に強く影響されることから，個々の要素技術についてその適用条件を明確にする必要がある．そこで，PDES の実現に必要な要素技術について，その性能に影響する要因を特定するために，異なる実行状況下での比較調査を支援する並列離散事象シミュレーション評価環境 VPSE の開発を行った．VPSE はソフトウェア的に PDES の実行環境をエミュレートすることによって，任意のプロセッサ台数，通信遅延ならびに計算粒度等のパラメータ化が可能であり，利用者に所望の PDES 実行環境を提供する．本論文では，この VPSE の機能と構成について述べるとともに，VPSE による適用事例としてタイムワープ方式に基づく待ち行列モデルを対象とする性質検討を行った．具体的には，対象モデルの規模と遷移事象による処理効率への影響，対象モデルの時刻印増加率の特性，ならびに事象処理の粒度と通信コストによる性能への影響を定量的に調べ，これら諸要因のタイムワープ方式に対する関連を明らかにした．

VPSE: An Evaluation Environment for Parallel Discrete-Event Simulation and Its Case Study on Time Warp

MASARU OKUMURA[†,☆] and TOSHINORI SUEYOSHI[††]

In Parallel Discrete Event Simulation (PDES), the execution can be speed up by exploiting the potential parallelism of the target simulation. However, choosing the appropriate implementation method and programming algorithms for the target simulation is crucial. Furthermore, because the implementation of PDES is strongly influenced by the target computer system and programming environment, there is the need to verify the influence of the basic technologies comprising PDES in its performance and the need of an environment testbed to verify the status of PDES. We developed the Virtual Parallel Simulation Environment (VPSE) in an attempt to satisfy these needs. VPSE emulates the execution environment and provide a flexible PDES environment to the user. In the software emulation, the user can parameterize the number of processing elements, communication latency and computation granularity. In this paper, we described the organization and functions of VPSE and examined the characteristics of a queueing network based on Time Warp algorithm. We measured the speedup in relation to the number of physical process (PP) and message density, timestamp increment and message density, communication latency and message density. Finally, we discussed the relation of these factors with Time Warp.

1. はじめに

並列計算機や通信システムの性能評価あるいは VLSI 設計における論理検証のように大規模化するシステムに対する有効なシミュレーション手法として，並列離散事象シミュレーション（Parallel Discrete Event Simulation，以下 PDES）がある[1),2)]．PDES は，システム内部の振舞いを離散事象としてモデル化し，これらシミュレーション対象が潜在的に有する並列性を活用して，並列処理によって高速化を図る手法である．PDES の実装に関連する要素技術として対象モデルの分割，スケジューリング，負荷分散，および同期アルゴリズム（仮想時刻管理）があり，PDES の効率的な実行を実現するにはこれら要素技術の適切な選択が不

† 九州工業大学情報工学部知能情報工学科
Department of Artificial Intelligence, Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology
☆ 現在，福岡大学電子計算センター
Presently with Computer Center, Fukuoka University
†† 熊本大学工学部数理情報システム工学科
Department of Computer Science, Faculty of Engineering, Kumamoto University

可欠である．なかでも並列性を効率的に活用する鍵となる仮想時刻管理を中心に研究が行われてきた．

しかしながら，実機を用いて得られたPDESの評価結果は，PDESシステムの実装が実行環境となる計算機システムとそのプログラミング環境に強く依存しているという事実から，限定的な環境下における評価が中心であった．一方，PDESの実行環境そのものも，従来主流であった汎用並列計算機から，最近では分散環境の普及によりネットワーク接続で構成されるワークステーション・クラスタ，あるいは複数CPUを搭載したマルチプロセッサ・ワークステーションへと広がりつつある．

このような実行環境の多様化により，要素技術の適切な選択を行う必要性からも通信性能などの実行環境に依存する要因，計算粒度などの対象アプリーションの特性といった諸条件の差異による要素技術の特性を定量的に評価することの重要性が増しているといえる．

そこで我々は，PDESの核となる要素技術に対して，このような諸条件が実行性能へどのような影響を与えるのかを明らかにするために仮想並列シミュレーション環境（Virtual Parallel Simulation Environment，以下VPSE）の開発を行ってきた[3]．VPSEは，PDESシステムの移植性を高めることで実行環境の差異が実行性能に与える影響を検討するのではなく，並列処理環境自体をソフトウェア的にエミュレートすることで，これらの諸条件の影響の調査を行う．このような実行環境の仮想化により，同一のPDESをパラメータを変更するだけで，異なる状況下で実行することを可能にするとともに，PDESプログラム内部の処理粒度についても任意に指定可能になるため，各処理の影響を比較，検討することが可能となる．

本論文では，並列離散事象シミュレーション評価環境VPSEの特徴と構成を示すとともに，楽観的な仮想時刻管理手法であるタイムワープ方式を対象にVPSEを用いて性質検討を行う．具体的には，評価モデルとして離散事象システムの基本モデルである待ち行列網モデルを利用し，実行環境に依存する通信性能を中心に実装上の要因による実行性能への影響を測定した．

以下，2章ではVPSEの特徴と構成について述べ，3章では，VPSE上へのタイムワープ方式に基づく待ち行列網モデルの実装について述べる．4章では，この待ち行列網モデルを利用してモデルの特性ならびに実行環境の要因に対する性能への影響について調査した．最後に5章でまとめを述べる．

2. VPSEの構成

2.1 VPSEの位置付け

現在，DESならびにPDESの支援環境としては，シミュレーション記述言語と呼ばれるモデル記述手法を用いたDES支援システムが多数利用可能である．利用者は，独自言語あるいはC言語の拡張形式として対象モデルを記述することで，所望の対象モデルを効率的にモデル化することができる．さらに，これらのDES支援システムは記述モデルを逐次，あるいは並列に実行するためのシミュレーションエンジンを提供している．しかしながら，PDESの研究への利用を検討すると，利用者がシミュレーションエンジンを自由に変更できないという点で十分であるとはいえない．そこで，PDESの要素技術の研究に適したDES支援システムの要件を考えると，次のような要件を提案できる．

(1) 同期アルゴリズムから独立したモデル記述

シミュレーション対象を限定せず，多数の物理システムを対象として取り扱うためには，シミュレーション記述言語のようなモデル記述能力が必須である．このようなモデル記述機能を持つDES支援システムとしてはMODSIM III[4]，CSIM18[5]等がある．特に対象システムに対する同期アルゴリズムの特性を検討するためには，モデル記述から同期アルゴリズムを分離することが望ましい．

(2) 同期アルゴリズムのモジュール化

PDESでは，その並列処理効率が同期アルゴリズムにより大きく影響を受ける．よって，対象モデルや実行環境の特性に応じて，柔軟に同期アルゴリズムを変更して評価を行えることが望ましく，(1)のモデル記述と同期アルゴリズムを分離するインタフェースを含め，同期アルゴリズム間にモジュール性を持たせることが重要である．このようなモデル記述と同期アルゴリズムの分離・モジュール化を実現しているDES支援システムにはCPSim[6]，MAISIE[7]等がある．

(3) 様々な動作環境の提供

PDESの要素技術について，どのような条件下で効果があるかという検討を行うためには，異なる複数の実行環境での評価が必要となる．これを実現するために2種類のアプローチが考えられる．1つは，PVMやMPIのような機種依存性の少ない通信システムを用いてPDESシステムの移植性を高めることで，複数の計算機システム上で動作させるというアプローチである．もう1つは，実行環境そのものをエミュレートするアプローチである．前者は実計算機システムを用いるこ

とで信頼性の高い評価結果を得ることができる．このようなシステムにはCPSim, MASIE, WARPD[8], Osim[9]がある．しかしその反面，プロセッサ要素数，接続形態等の項目が利用する計算機システムの構成に制約されることになり，利用者が任意の通信状況下での評価を行うには，PDESシステムに付加的な機構を盛り込む必要がある．これに対し，後者のアプローチでは，実行環境そのものをエミュレートすることで，実機の構成に制約されない実行環境下での評価が可能になる．

したがって，DES支援システムとしては上記3要件を満たすことが最適であるが，本研究は前章で述べたように実行環境を含む諸条件に対する要素技術の特性を検討することを研究の目的とするため，上記の3要件のうち特に(3)を実現するDES支援システムとしてVPSEの開発を行った．

2.2 VPSEの概要

VPSEは，並列処理環境上でPDESを実行するPDESシミュレータを単一計算機上でシミュレートするシミュレータ・シミュレータである．プロセッサ要素数，通信形態（トポロジ，通信遅延）などの実行環境に該当する部分をVPSEではソフトウェアによりエミュレートされる仮想プロセッサ（以下，プロセッサ）と通信機構として提供する．また，VPSEでは実計算機システムのプロセッサ上で実行される個々のPDESプログラムに該当する処理をシミュレーション・プリミティブ（以下，プリミティブ）とそのスケジューリング機構として定義し，これらのプリミティブ処理を各プロセッサ上で実行する．

このように実行環境のソフトウェア・エミュレーションを行うことで，利用者は任意の実行環境下におけるシミュレーション対象のマッピング手法や同期アルゴリズムの検討が行える．また，実際に実行されるプログラム処理を利用者が定義したプリミティブのみに制限することで，実機上で生じる他プロセスの影響などを完全に除くことが可能である．VPSEのこのような特徴により，異なる実並列処理環境下でPDESの評価を行う場合に比べて，PDESプログラムの修正を必要とせず，コンパイラ，OS，通信システム等の複数要因による影響といった測定結果を不明瞭にする要因を排除し，特定のパラメータに着目した測定を行うことが可能になる．

同様のシステムとしてはVirtual Timeアルゴリズム[10]に基づく並列シミュレーションの性能評価を行うVTSIM[11]があるが，VPSEはメッセージ交換に基づくシミュレーション対象であれば，任意の分散時刻管理方式を実装できる．以下，VPSEの特徴を要約する．

- メッセージ交換に基づくPDESのエミュレート環境を提供する．
- PDESの実行に必要なプリミティブ処理と同期処理（仮想時刻管理）のみが実行される．
- 任意のプロセッサ台数を選択できる．
- 任意の通信形態（トポロジ，通信遅延）を想定できる．
- 各プリミティブの処理粒度を任意に設定できる．

2.3 VPSEの基本構成

VPSEは図1に示すように3つの部分，(1) 分割処理部，(2) シミュレーション・プリミティブ部，(3) 仮想並列プロセッサ部，から構成される．以下，本節では，これらの3つ部分について具体的に説明する．

2.3.1 分割処理部

PDESでは一般にシミュレーション対象を物理プロセス（Physical Process，以下PP）の集合としてとらえ，これらのPPに対応するシミュレータ上の要素を論理プロセス（Logical Process，以下LP）と呼んでモデル化を行う．さらに，これらのLPは実際の処理を行う並列計算機のプロセッサに割り当てる必要がある．分割処理部では，利用者によって指定されるプロセッサへのLPの分割情報に従い，LPのプロセッサへのマッピングと各LP間の接続構造を作成する．

2.3.2 シミュレーション・プリミティブ部

シミュレーション・プリミティブ部は一般に並列計算機の各プロセッサ要素上で実行されるPDESプログラムに相当する部分であり，VPSEの利用者はプリミティブとしてシミュレーションに必要な事象評価やメッセージ送受信等の処理を行う機能を実装する．VPSEにおいては，このプリミティブが後述の仮想並

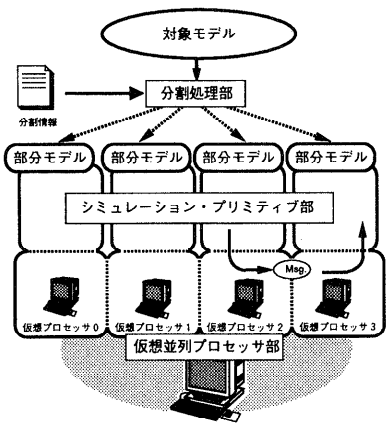


図1 VPSEの構成
Fig. 1 The organization of VPSE.

列プロセッサ部のプロセッサ上で実行される基本処理単位となる．また，各プリミティブの実行順序を管理するための事象リストの操作や仮想時刻の制御に必要となる同期機構（仮想時刻管理機構）は，プリミティブのスケジューリング機能として組み込む．なお，同一仮想時刻印のメッセージに対する処理順序のようなタイミング依存の部分についても，プリミティブ・スケジュールの一部として利用者が設定可能であり，その順序決定規則に再現性がある場合，実行結果の再現性も保障される．

一方，シミュレーション処理の評価に要する総実行ステップ数を算出するために，各プリミティブには実行時間に相当する値として，実行ステップ数を任意に設定できる．これにより，利用者は処理粒度をパラメータとして変化させてシミュレーション実行を試みることが可能になる．また，通常のPDESでは，各種事象処理の統計や計測のためのルーチンが，PDESの実行結果に影響を与える恐れがあるが，VPSEではこの実行ステップによって経過時間を測定するため，さまざまな項目についての情報を実行結果への影響を与えることなく取得できる．なお，現段階ではモデル記述ならびに同期アルゴリズムについては，専用の記述言語ではなく，C言語によるプログラム記述として実装している．

2.3.3 仮想並列プロセッサ部

仮想並列プロセッサ部は，単一計算機上に実装するVPSE上にメッセージ・パッシングに基づく仮想的な並列処理環境を提供する．具体的には，複数台のプロセッサ上で前述のプリミティブをラウンドロビン方式で処理することで並列実行をエミュレートしている．プリミティブの実行にともなって発生するメッセージ通信も仮想並列プロセッサ部で処理する．さらに，利用者はプロセッサ間の接続形態（トポロジ）や通信遅延などを指定することにより，任意の通信形態上でのシミュレーション実行が行える．現在，VPSEでは接続形態として，(1) バス接続，(2) 2次元メッシュ網，(3) 2次元トーラス網の3種類のトポロジを実装しており，プロセッサ間のホップ数に応じた通信遅延が反映される．なお，通信経路におけるメッセージ流量に応じた通信遅延の反映については，現時点では考慮していない．

仮想並列プロセッサ部のプリミティブ処理の具体例を図**2**を用いて説明する．まず，実行ステップ $T=\alpha$ においてプロセッサ1上でプリミティブAが実行される．ここでプリミティブAの実行ステップ数は1であるため，この時点でプリミティブAは終了する．プリミティブAによって生成されたプロセッサ2宛のメッセージ（Msg.）の受信処理は指定されている通信遅延（3 stcp）に基づき，プロセッサ2の実行ステップ $T=\alpha+3$ にスケジューリングされる．同様に，各実行ステップごとにプロセッサにスケジューリングされているプリミティブが実行される．プリミティブB（3 step）のように複数ステップを要するプリミティブは，その間該当プロセッサを占有することになる．また，実行ステップ $T=\alpha+3$ でプロセッサBにスケジューリングされたメッセージは，$T=\alpha+3$ の時点でプリミティブBが実行中であるため，受信処理はプリミティブBが終了まで延期され，実行ステップ $T=\alpha+5$ で処理されることになる．

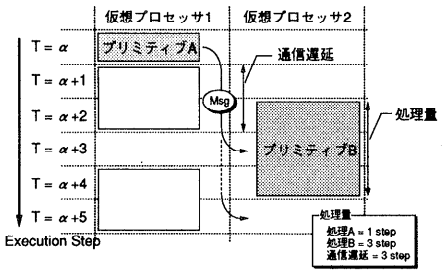


図2 仮想並列プロセッサ部の動作
Fig. 2 Behavior of the virtual parallel processor unit.

3. シミュレーション対象モデル

本章では，評価対象であるタイムワープ方式の仮想時刻管理方式とシミュレーション対象となる待ち行列網モデルの詳細，ならびにタイムワープ方式に基づきVPSEへ実装するためのプリミティブとスケジューリングの定義について述べる．

3.1 タイムワープ方式

DESシステムではシミュレーション対象をLPとしてモデル化し，LP間での相互作用は，その系の時刻（仮想時刻）において受け手に影響を与えるべき仮想時刻を時刻印として付加したメッセージをLP間でやりとりすることによって実現する．さらに，高速化を図るPDESでは，同一仮想時刻の事象あるいは互いに因果関係のない事象を並行に処理することを許すが，LPごとに異なる仮想時刻が進行する恐れが生じる．よって，あるLPが受け取るメッセージの順序が時刻印順であるとは限らず，対象全体での事象処理の正当性を保証するためにメッセージの処理順序を正しく管理する必要がある．

このような同期アルゴリズム（仮想時刻管理方式）の1つにJerffersonによって提案されたVirtual Time[10]に基づくタイムワープ方式がある．楽観的な手法に分

類されるタイムワープ方式では，投機的にメッセージの評価処理を進める．メッセージの授受により処理順序に矛盾が発生した場合は，ロールバックと呼ばれる矛盾解消処理により処理順序の正当性を保証する．また，誤った仮想時刻のメッセージを送信していた場合は，アンチ・メッセージと呼ばれる取消し要求を送出し，誤った事象の取消し処理を行う．このようにLP間で緩い同期を採用しているため，シミュレーション対象の持つ並列性を有効に引き出せる反面，分散共有メモリ型計算機やメッセージ交換を持つ計算機システム上での利用では，通信処理，処理対象の分割・マッピング，負荷変動などの影響を受けやすいという欠点がある．

そこで，本論文ではメッセージ交換方式に基づく環境上にタイムワープ方式を実現した場合に，その実行環境の基盤となる通信処理やモデルの特性が実行効率にどのような影響を及ぼすかを検討する．本論文で取り扱うタイムワープ方式は，Lazy cancellation[12]やlazy re-evaluation[13]のような処理方式の発展系との比較も考慮して，矛盾処理を発見した際にただちに取消し処理を開始する，最も単純なaggressive cancellationを採用した．なお，通信路上でのメッセージ順序の保存性についてはメッセージ順序が保存されることを前提としており，メッセージの到着順序の逆転は生じないものとしている．また，この前提に基づくメッセージの一括取消し手法[2]については本実装では採用していない．

3.2 VPSEにおける待ち行列のモデル化

3.2.1 評価対象となる待ち行列モデル

VPSEを用いたシミュレーション対象として，本論文では$N \times N$個のサービス施設（以下，サーバ）が2次元トーラス状に結合された待ち行列網モデルを想定する．各サーバは東西南北の4方向に入出力ポートを持つものとする（**図3**参照）．各サーバは仮想時刻を刻む個別の時計を，また，客（以下，メッセージ）はサーバへの到着時刻を表す時刻印をそれぞれ持つものとする．各サーバは，入力ポートからメッセージを受信し，メッセージを到着順にキューイングし，FCFS（First Come First Service）でメッセージ処理を行う．また，サーバは1つのメッセージ処理後，等確率（0.25）で4方向の出力ポートのいずれかに対して新たなメッセージを送信するものとする．このため，サーバの処理によってシステム内のメッセージ数が減少することはない．

各サーバの初期状態としては，D個のメッセージが各サーバにキューイングされているものとする．したがって，対象システム内の総メッセージ数は，$D \times (N \times N)$個となる．以後，この各サーバの初期メッセージ数をメッセージ密度（Message Density）Dと呼ぶ．なお，サーバが1つのメッセージの処理に要する時間T_αは，サーバごとに固定時間とする．

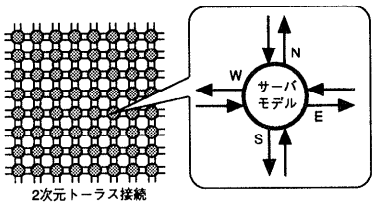


図3 待ち行列システム
Fig. 3 The queueing network system.

3.3 タイムワープ方式の実装

本節では，VPSEへのタイムワープ方式に基づいた待ち行列網モデルの実装を行ううえで必要なPP（サーバ）のLPへのマッピングとLPへの分割問題，そして一般のPDESプログラムに相当するプリミティブとそのスケジューリング機構の定義について説明を行う．

3.3.1 LPのマッピングと分割

PDESでは，実装上の問題として1つのPPを1つのLPに割り当てて実現する手法と複数のPPを1つのLPで実現する手法がある．前者は，自然な並列化であるが，各LPごとで並列処理を実現するための機構を必要とするため，オーバヘッドが問題となる．このためVPSEでは後者の手法を採用した．また，LPのプロセッサへの割当てについても同様の問題が生じるが，先の割当てにより並列度がLP数によって制約されることから，プロセッサに対しては1対1でLPを割り当てることとした．

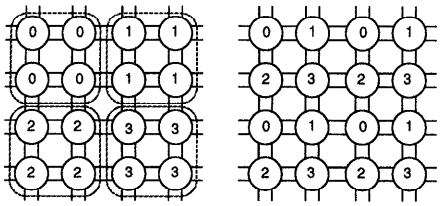
また，対象PP（サーバ）のLPへの割当て問題も，負荷分散や通信コストによる影響の面からも考慮すべき点である．そこで，分割手法として**図4**に示す，ブロック分割とモジュロ分割の2種類の分割手法を定義する．図中の○印内の数字は，サーバが割り当てられるプロセッサ番号を意味する．

3.3.2 シミュレーション・プリミティブ

待ち行列網モデルをVPSE上にタイムワープ方式で実装するために，以下の3種のプリミティブを定義する．各プリミティブの内容を以下に示す．

Server_Primitive

待ち行列網のサーバ処理を実現するプリミティブ．3.3.3項で述べるスケジューリングに従い，キュー上のメッセージをFCFSで処理し，新たな時刻印を付加したメッセージをいずれかの出力ポートに



(a) ブロック分割 (b) モジュロ分割

図4 待ち行列網の分割手法
Fig. 4 Partitioning method of the queueing network.

送出する．その際，送出先のサーバが他のプロセッサに割り当てられたLPに属する場合，送出されたメッセージは，2.3.3項で述べたように適切な通信遅延時間を経た後，該当LPに受信処理される．同一プロセッサに属するサーバへのメッセージ通信は，ただちに該当サーバ上へ登録される．また，ロールバックに備えてメッセージ評価ごとに履歴を保存する機能を有する．

Recv_Primitive

他のLPから到着したメッセージの受信処理を行うプリミティブ．後述のスケジューリングに従い，受信したメッセージを該当サーバのキュー上に登録する．

Rollback_Primitive

処理対象のメッセージがアンチ・メッセージの場合に実行されるプリミティブ．削除対象のメッセージが未処理でキュー上に存在する場合は単に削除対象メッセージを削除する．一方，削除対象のメッセージがすでに処理されてしまっている場合は，矛盾発生時刻の直前までLPの履歴情報を巻き戻し，履歴情報に基づいてLPの状態を復元する．また，誤って送出したメッセージに対してはアンチ・メッセージを送出する．

3.3.3 スケジューリング

次に，タイムワープ方式における各プロセッサ上でのプリミティブのスケジューリングを**図5**に示す．各プロセッサは，スケジューリングに従い，最早生起仮想時刻のメッセージを *Server_Primitive* により処理する．ただし，プロセッサ間では非同期的にこれらの処理が実行されるため，処理順序に矛盾が発生する可能性があり，この場合は *Rollback_Primitive* による矛盾解消処理，あるいはアンチ・メッセージの送信処理を行う．なお，図5には示していないが，本モデル化では同一時刻印を持つメッセージが複数ポートから到着した場合は，ポートごとに設けた優先度に従って処理を実行するものとした．

また，ロールバックに備えて，*Server_Primitive* 実行ごとに履歴情報を保存するが，シミュレーションの進行にともないメモリ使用量が増加するため，定期的に大域的な同期処理を行い，*GVT* (*GlobalVirtualTime*) と呼ばれる仮想時間の下限を求め，*GVT* 以前の不要な履歴情報を破棄する．

```
vtime = start_time /* vtime : 仮想時刻 */
while (vtime <= end_time){
  if(not_empty(recv_list)){
     /* Insert msg to event_list */
     Execute Recv_Primitive
  }
  if(GVT Interval elapsed){
     Calculate GVT
     Barrier Sync
     vtime = GVT /* GVT:Global Virtual Time */
     Garbage Collection /* 履歴情報の破棄 */
  }
  first_event = min_time_event(event_list)
  if(first_event scheduled for anti){
     Execute Rollback_Primitive for Anti
  } else if(first_event scheduled for rollback){
     Execute Rollback_Primitive for RollBack
  } else {
     /* Execute first_event */
     Execute Server_Primitive
  }
}
```

図5 タイムワープ方式に基づくプリミティブのスケジューリング
Fig. 5 The scheduling of the primitives based on Time Warp.

4. 性質検討

本章では，VPSEを用いて実装した待ち行列網モデルに対して，実行環境に相当する要因ならびにタイムワープ方式を実装するうえで考慮すべき項目について，並列処理効率との関係を検討する．

4.1 測定条件

本節では，本章の測定に用いた各種の測定条件を定義する．本節以降，特に明記しない限り，これらの設定を用いて測定を行った．

まず，想定する実行環境としては，プロセッサがバス接続で接続された構成とし，いずれのプロセッサ間の通信も固定された通信遅延 $T_{latency}$ で行えるものとした．$T_{latency}$ については，待ち行列網モデルの主な処理である *Server_Primitive* に要する実行ステップを基準として相対的に定義する．$T_{latency}$ が $\times 10$ であれば，プロセッサ間での1つのメッセージ通信には *Server_Primitive* 10回分の実行ステップ数を要することを意味する．

処理時間算出の基準となる各プリミティブの実行

表1 プリミティブの実行ステップ
Table 1 Execution steps of primitives.

プリミティブ名	実行ステップ数
Server_Primitive	100
Recv_Primitive	1
Rollback_Primitive	100 × 履歴の深さ

ステップ数については，**表1** に示すように設定する．*Server_Primitive* のステップ数を最小ステップである1ではなく，100ステップとすることで，事象処理に対して $T_{latency}$ の粒度を ×0.01 まで想定可能にしている．また，*Recv_Primitive* のコストについては，$T_{latency}$ の差に影響を与えることを防ぐため，十分に小さな値（1ステップ）とした．さらに，タイムワープ方式において性能低下の一因となる *Rollback_Primitive* については，実行時の動的な実行コストの増減を考慮して，巻き戻す履歴の深さに比例するコストを設定した．

また，通信処理の影響を顕著にするために，LPの分割手法としては3.3.1項で述べたモジュロ分割を用いた．なお，*GVT* 処理は各プロセッサが，1,000メッセージ評価ごとに実行するのとし，シミュレーションの終了条件については，*GVT* が100,000 uvt（unit virtual time）を超えるまでとした．

4.2 PP数と事象密度による影響

まず，いくつかの測定を行う前にパラメータの変更に対する基準となる特性を把握するために，シミュレーション対象の並列性の推移に対する性能変化を測定する．対象中の並列性を変化させるためにサーバ（PP）数を 8 × 8 (= 64 個)，16 × 16 (= 256 個)，32 × 32 (= 1024 個) の3種類に，遷移する事象数を決定するメッセージ密度 D を 1，4，16，32，64 と変化させて測定を行った．また，各サーバのサービス時間 T_α を 320 uvt 固定と [0,640] uvt に均一に分布する2種類で測定した．プロセッサ数64台，$T_{latency}$ を ×1 と設定した場合の並列度の変化にともなう速度向上率を**図6** に示す．ここで，速度向上率は同じVPSE上に実装した逐次処理方式による待ち行列網シミュレーションの実行結果に対する速度向上比である．

この結果から，各プロセッサ要素あたりのLP数の増加（図中の (a)→(b)→(c)，(A)→(B)→(C)）が性能に大きな影響を与えていることが確認できる．

また，メッセージ密度 D の増加，すなわち対象モデル中の事象数の増加が性能向上の一因であることが図6中 (a)，(B) 等からも確認できる．事象数の増加による効果については，図6中 (B) についてのメッセージ密度 D と事象の取消し回数の比（処理前/処理後）の推移を**表2** に示す．

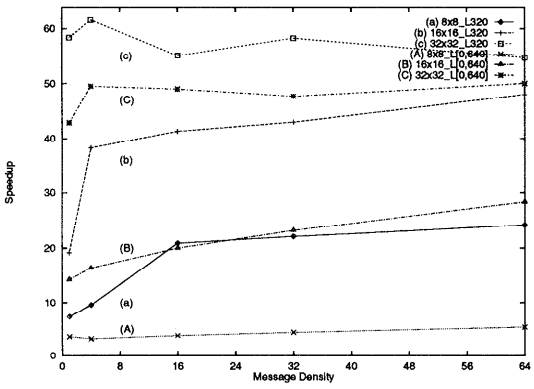


図6 PP数に対する速度向上率
Fig. 6 Speedup with the number of PPs and message density.

表2 事象処理の取消し回数の比
Table 2 Ratio of cancelled processed events and cancelled unprocess event.

Messege Density	1	4	16	32	64
Ratio	0.56	0.67	0.76	0.92	1.33

また，これらの測定結果を比較検討するうえで，パラメータとなるPP数，メッセージ密度およびサービス時間 T_α の変化に対する待ち行列網の性質変化を考慮する必要がある．本測定においては，各々固有のサービス時間 T_α を持つPPが，等確率で隣接するPPにメッセージを送出することを前提にしている．よって，PP数の変化は対象問題の規模を変化させるが，その性質自身には影響を及ぼさない．また，同様にメッセージ密度の変化も事象遷移の傾向（サーバにおけるメッセージの処理待ち時間など）には影響を与えるものの，対象問題の性質そのものには影響を与えない．これに対して，PPの T_α は待ち行列網の性質に影響を与える要因である．そこで，T_α の変化に対する事象の発生傾向を把握するために，対象内の事象遷移傾向を示す指標として，各PPにおける平均メッセージ送出間隔に着目する．**図7** は，サーバ数が 8 × 8 の待ち行列網において T_α をそれぞれ，10，40，160，320，640 uvt に固定する場合と [0,10]，[0,40]，[0,160]，[0,320]，[0,640] uvt の範囲内に均一に分布する10通りの場合に対してメッセージ密度とメッセージ送出間隔との関係を示したものである．図7から，サービス時間の変化に対して，各々の待ち行列網では異なる平均送出間隔を示すものの，メッセージ密度の変化に対してはいずれの T_α においても類似した傾向を持つことが確認できる．したがって，T_α も待ち行列網の性質に影響するものの，事象遷移に対しては一

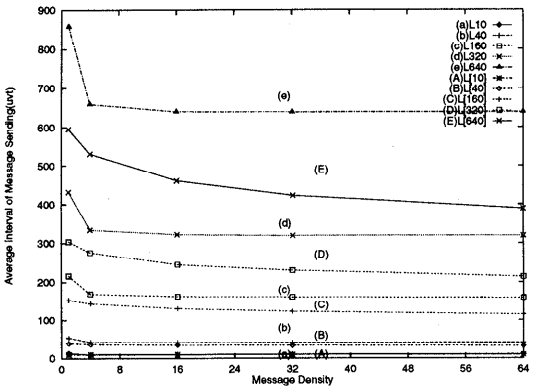


図7 PPの平均メッセージ送出間隔
Fig. 7 Average interval of messeage sending on PP.

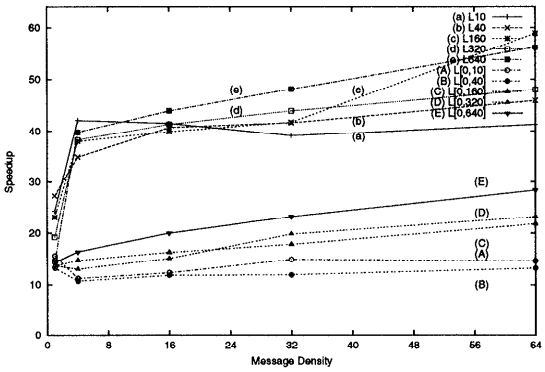


図8 時刻印の増加に対する速度向上率
Fig. 8 Speedup with timestamp increment and messeage density.

定の性質を持つものと見なせる．

本節以降，サーバ数 16 × 16，プロセッサ数 64 台の場合について，測定を行うことにする．

4.3 時刻印の増加幅による影響

PDES の処理性能に影響を与える対象モデルの特性の 1 つに時刻印増加幅（*Lookahead*）がある．*Lookahead* とは，仮想時刻 T の LP が時刻 $T+L$ まで新たな事象を送出しないことを送出先 LP に対して保証できる場合の時間幅 L であり，本論文中ではサーバのサービス時間 T_α がこれに相当する．この *Lookahead* よる影響を把握するために，全サーバのサービス時間 T_α を 10，40，160，320，640 uvt の 5 種類について固定して測定を行った．さらに，T_α が指定された範囲内に均一に分布する場合として，[0,10]，[0,40]，[0,160]，[0,320]，[0,640] uvt の 5 種類についても測定を行った．なお，$T_{latency}$ は ×1 である．

図8 に示す測定結果から，タイムワープ方式においては対象モデル中の PP が固定された T_α を持つ状況（図中 (a)，(b)，(c)，(d)，(e)）よりも，それぞれの PP の T_α が異なる状況（図中 (A)，(B)，(C)，(D)，(E)）が，処理性能を大きく低下させていることが確認できる．

これは対象とする PP が複数の他 PP からのメッセージにより活性化されるため，送信元 PP の *Lookahead* にばらつきがある場合，処理順序の逆転が容易に発生するためと推測できる．また，対象モデルの *Lookahead* がシミュレーション対象全体でほぼ一定であることに加えて，1 プロセッサに対して十分な PP 数が割り当てられないと高い並列性を抽出できないことが *Lookahead* に対する測定と前節の図 6 から確認できる．

4.4 ロールバック・コストによる影響

タイムワープ方式では事象処理の処理順序に矛盾が

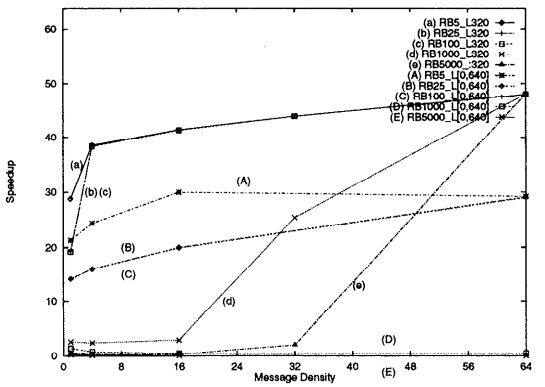


図9 ロールバックコストに対する速度向上率
Fig. 9 Speedup with rollback cost and message density.

発生した際，誤った事象処理を取り消すためにロールバック処理を必要とする．このロールバック・コストは対象モデルに依存するため，事象処理のコストを上回るケースも想定できる．そこで，ロールバック・コストすなわち *Rollback_Primitive* の実行ステップ数を *Server_Primitive* の実行ステップ数の ×1/20，×1/4，×1，×10，×50 と変化させて測定を行った．T_α は 320 uvt 固定と [0,640] uvt の 2 種類について，通信遅延 $T_{latency}$ を ×1 とした場合の測定結果を**図9** に示す．

図 9 より，ロールバック・コストが PP の事象処理よりも大きい場合（図中 (d)，(e)，(D)，(E)）は性能の著しい低下を招くのに対し，ロールバック・コストの改善を想定した場合（図中 (a)，(b)，(A)，(B)）における性能向上はわずかであることが観測できる．ただし，T_α がばらつくような状況では，ロールバック・コストの増加は致命的であり，メッセージ密度の増加による性能の大きな改善は認められない．

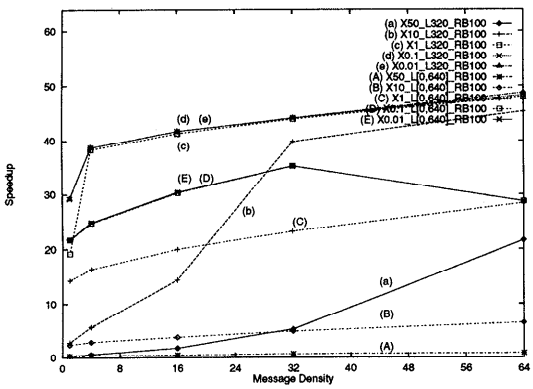


図10 通信遅延に対する速度向上率
Fig. 10 Speedup with communication latency and message density.

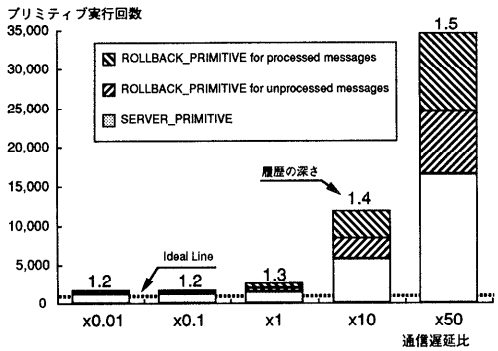


図11 通信遅延に対するプリミティブ実行回数
Fig. 11 Number of primitive executions against communication latency.

4.5 通信遅延による影響

通常の並列処理と同様に，PDESにおいても計算処理と通信処理の比率が並列処理効率を大きく左右する．特にメッセージ交換に基づくPDESでは，その通信遅延時間 $T_{latency}$ がLP間の仮想時刻の伝達に直接影響するため，検討すべき重要な項目である．ここでは，事象処理の中心となる *Server_Primitive* の処理粒度に対して，PE間の通信遅延である $T_{latency}$ を変化させて測定を行う．*Server_Primitive* の実行ステップ数を基準に，$T_{latency}$ を ×0.01，×0.1，×1，×10，×50 と変化させて測定した．T_{α} を320 uvt固定と[0,640] uvtに設定した場合の測定結果を**図10**に示す．

いずれの場合も，$T_{latency}$ が事象処理の粒度よりも大きくなると性能が大幅に低下し（図中(a)，(b)，(A)，(B)），事象処理よりも小さくなると若干の改善が見られる（図中(d)，(e)，(D)，(E)）．この原因を検討するために，各プリミティブの実行回数に注目する．**図11**に T_{α} を320 uvt固定，メッセージ密度 $D=1$ とした

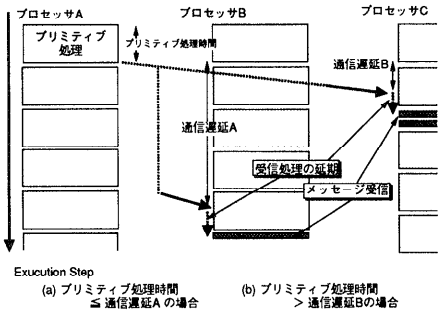


図12 プリミティブにおける受信処理
Fig. 12 Recive processing of primitives.

表3 事象処理の実行効率
Table 3 Execution efficiency of event processing.

$T_{latency}$	×0.01	×0.1	×1	×10	×50
Efficiency	79%	68%	49%	12.5%	4.3%

際のプリミティブ実行回数の平均回数ならびに1ロールバックあたりの巻き戻した履歴の深さの平均値を示す．なお，図11中の点線は，プロセッサ数64台時の理想的な *Server_Primitive* の実行回数である．

まず，$T_{latency}$ が事象処理の粒度よりも大きくなる場合については，取消し回数ならびに *Server_Primitive* の実行回数が急増している．これは通信遅延が事象処理粒度よりも大きくなる場合，**図12**(a)に示すようにメッセージが送信されてから送信先プロセッサで受信されるまでに数プリミティブ分の時間を要することから，アンチ・メッセージを含む仮想時刻の伝達に遅れが生じ，後に取り消すことになる不要な事象処理を引き起こすためと推察できる．この点については，図11と同一条件における総 *Server_Primitive* の実行回数に占める有効実行回数（ロールバックされなかった回数）の割合を示す，**表3**の値からも裏付けられる．

一方，$T_{latency}$ が事象処理の粒度よりも小さくなる場合，プリミティブにおけるメッセージの受信処理は図12(b)に示すように行われる．すなわち，プリミティブ実行中に到着したメッセージは，少なくとも1プリミティブ終了までは受信処理が遅延させられるものの，$T_{latency}$ が事象処理の粒度よりも大きい場合に比べ，速やかに受信処理されるため，表3にもみられるように不要な取消し処理や事象処理を抑制する効果がある．ただし，$T_{latency}$ 減少の効果は受信処理時に吸収されやすいことから，×0.1 と ×0.01 の差による効果が現れていないことが，図10中の(d)，(e)，(D)，(E)からも確認できる．

また，4.4節で行ったようなロールバック・コストの削減による効果は，図11中の ▧ と ▨ の部分の実行時

間を減少させるが，▨で示される *Server_Primitive* の回数を上回る部分が依然として性能低下の要因として残ることになる．

5. おわりに

本論文では，仮想的な並列処理環境上でPDESの実行をシミュレートする並列離散事象シミュレーション評価環境VPSEの特徴と構成について述べるとともに，タイムワープ手法について待ち行列網モデルを対象とする性質検討を行った．この性質検討から，aggressive cancellationを用いたタイムワープ方式では，(1) 各プロセッサあたりのPP数の増加が高い並列処理効率を実現する一因であり，さらに対象中の事象遷移数の増加は，不要な事象の発生を未然におさえる効果がある，(2) PP間における *Lookahead* のばらつきが顕著に性能低下を引き起こす，(3) 事象処理の粒度に対する通信遅延の増大は，ロールバックの慢性的な発生を引き起こす一因である，ことをシミュレーション実験により明らかにした．

今後は，シミュレーション対象の特性を含めて，どのような要因が処理効率に効果的であるか，あるいは性能低下の原因となるかをVPSEを活用して検討を行う予定である．

謝辞 本研究を行うに際し，PDESに関して日頃からご討論いただく本学知能情報工学科の江島孝幸氏ならびに情報処理処理機構講座の諸氏に感謝します．

参考文献

1) Reinhardt, S.K., Hill, M.D., Larus, J.R., Lebeck, A.R., Lewis, J.C., Wood, D.A.: The Wisconsin Wind Tunnel: Virtual Prototyping Parallel Computers, *Proc. ACM SIGMETRICS Conference* (1993).
2) 松本幸則，瀧　和男：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.387–396 (1992).
3) 奥村　勝，末吉敏則：並列離散事象シミュレーション評価環境：VPSE, 信学技法，CPSY97-58, pp.55–62 (1997).
4) http://www.caciasl.com/modsim.html, CACI Products Company
5) http://www.mesquite.com/, Mesquite Software Inc.
6) http://www.wdn.com/bti-sim/, BoyanTech, Inc.
7) Bagrodia, R.L., Liao, W.T.: A Language for the Design of Efficient Discrete-Event Simulation, *IEEE Trans. Softw. Eng.*, Vol.20, No.4, pp.225–238 (1994).
8) Martin, D.E., McBrayer, T.J. and Wilsey, P.A.: warpd: A Time Warp Simulation Kernel for Analysis and Application Development, *29th Hawaii Intl. Conf. on System Sciences (HICSS-29)*, Vol.1, pp.383–386 (1996).
9) 阿部一裕ほか：超並列オブジェクト指向シミュレーション環境Osim, *Proc. JSSST Workshop on Object Oriented Computing*, S3–2, 研究会資料シリーズNo.2 (1996).
10) Jefferson, D.R.: Virtual Time, *ACM Trans. Programming Languages and Systems*, Vol.7, No.3, pp.404–425 (1985).
11) 島津伸行，工藤知宏：Virtual Timeアルゴリズムに基づく分散離散事象シミュレーションの性質検討，信学技法，CPSY95–54, pp.33 38 (1995).
12) Gafni, A.: Rollback Mechanisms for Optimistic Distributed Simulation Systems, *Proc. SCS Multiconference on Distributed Simulation*, Vol.19, pp.61–67 (1988).
13) Jefferson, D.R.: Virtual Time II: the cancelback protocol for storage management in Time Warp, *Proc. 9th Annual ACM Symposium on Principles of Distributed Computing*, pp.75–90 (1990).

(平成9年11月4日受付)
(平成10年4月3日採録)

奥村　勝（正会員）
1970年生．1993年九州工業大学情報工学部知能情報工学科卒業．1995年同大学大学院情報工学研究科情報科学専攻博士前期課程修了．同大学院博士後期課程を経て，1998年4月より福岡大学電子計算センター講師．並列処理，計算機アーキテクチャの研究に従事．

末吉　敏則（正会員）
1953年生．1976年九州大学工学部情報工学科卒業．1978年同大学大学院工学研究科情報工学専攻修士課程修了．同年九州大学工学部情報工学科助手．同大学院総合理工学研究科助教授，九州工業大学情報工学部知能情報工学科助教授を経て．1997年より熊本大学工学部数理情報システム工学科教授．工学博士．計算機アーキテクチャ，システムソフトウェア，計算機ネットワーク，VLSIシステム設計などの研究に従事．著書「並列処理マシン」(共著，オーム社)．