タイムワープ機構の新しい応用

——並列無格子配線——

松本幸則† 瀧 和男††

タイムワープ機構が，並列処理における実行順序制御機構として汎用的に利用できることに着目し，その全く新しい応用として並列LSI配線への適用を試みるとともに，有効性を評価した．タイムワープ機構は，主に並列事象シミュレーションの分野で，事象処理の実行順序制御に用いられている．この機構は，実行順序を守りながらも，積極的な見込み計算により高い並列性を抽出するという性質を持つ．われわれは，この性質が他の多くの問題分野でも有効であると考え，一例として，従来とは全く異なる並列LSI配線問題への適用を試みた．並列LSI配線処理では，タイムワープ機構の導入によって，配線品質に影響する配線順を考慮しながら，かつ，高い並列性が抽出可能になると期待される．われわれは，タイムワープ機構を適用した並列配線プログラムをMIMD型分散メモリマシン上に試作し性能評価を行った．その結果，64プロセッサを用いて19倍以上の台数効果を得た．また，大規模並列計算機上では，従来の並列化手法による配線プログラムよりも効率的に動作することを確認した．

A New Application of the Time Warp Mechanism

——To Parallel Gridless Routing——

Yukinori Matsumoto† and Kazuo Taki††

This paper focuses on a completely new application of the Time Warp (TW) mechanism. A novel approach to parallel LSI routing based on TW is presented. TW has been applied mainly to parallel discrete event simulation. TW can exploit high parallelism in target problems with speculative computation while controlling execution order correctly. We consider that this feature is advantageous to many applications such as parallel LSI routing, where quality of a routing solution depends on a given wiring order whereas high parallelism extraction is needed for high-speed execution. We built an LSI router based on TW and evaluated it on an MIMD machine. Measurements showed that more than 19-fold speedup was attained using 64 processors. Besides, empirical comparison revealed that therouter using TW worked more efficiently than a conventional router on a large-scale machine.

1. はじめに

近年，超高速計算を実現する技術として並列処理が注目されている．並列処理においては，高性能ハードウェアとともに，その性能を十分に引き出す並列プログラミング技術が重要である．並列プログラミングでは，対象問題から高い並列性を抽出することと共に，処理の正当性を保証するために実行順序を正しく制御することが要求される．

数値計算など，均質な計算が大量に発生し，かつ静的に並列性を抽出できる問題においては，上記の要求を共に満たすことは比較的容易である．一方，不均質な計算が動的に発生する問題においては，正しい実行順序を厳密に守ろうとすると並列性が損なわれる場合がある．事象シミュレーションは後者の典型的な例である．

並列事象シミュレーションでは，処理の正当性を保証するために，全順序関係を守る同期的方法[1]や，半順序関係を厳密に守るデータ待ち合わせ機構を用いた方法[16),22)]が提案されている．しかし，これらの手法では，ボトルネックやデッドロックの発生によって，効率的な並列処理が難しい場合がある．一方で，実行順序の誤りを許して処理を進めるタイムワープ機構[8)]が提案され，その有効性が報告されている[2),14)]．

タイムワープ機構は，処理順序の正当性を仮定した見込み計算（speculative computation）を積極的に行

† 三洋電機(株)東京情報通信研究所新情報処理研究室
Advanced Information Processing Laboratory, Tokyo Information & Communication Research Center, SANYO Electric Co., Ltd.
†† 神戸大学工学部情報知能工学科
Department of Computer and Systems Engineering, Kobe University

うことで高い並列性を抽出する．したがって，不当な処理を実行する場合もあるが，誤りが判明した時点でやり直すことにより，結果的には処理の正当性を保証する．われわれは，処理順序を厳密に守ることが並列性低下につながるような問題一般に対して，タイムワープ機構が有効に機能すると考えた．そして，この性質を持つ問題の具体例として LSI 配線問題に着目し，これをタイムワープ機構の全く新しい応用問題として取り上げた．

LSI 配線は LSI 設計の一工程であり，LSI 表面に配置された端子間を接続するための配線径路決定を目的とする．LSI の配線設計は多大な処理時間を必要とするため，高速化が強く望まれている．このような背景から，並列計算機を利用した配線プログラムがいくつか開発されてきた[3)~5),10),20),26)]．しかしながら，タイムワープ機構を適用した例はない．

われわれは，LSI 配線問題に対するタイムワープ機構の有効性実証を目的に，並列推論マシン PIM/m[17),23)]上にタイムワープ機構を用いた配線プログラム（以後タイムワープルータと呼ぶ）を試作した．そして，実際の LSI データを用いて，台数効果の測定や従来の並列化手法との比較評価を行った*．なお，基本配線アルゴリズムとして無格子配線アルゴリズム[7),13),19)]の一つを採用した．このアルゴリズムは，配線幅および配線位置の自由度が高いため，将来的にも有望な方法と考えられる**．

以下，第2章では，タイムワープ機構の概要とその応用分野について述べる．続いて第3章で LSI 無格子配線へのタイムワープ機構の適用方法を具体的に説明し，第4章で実装概要について簡単に述べる．また，第5章で台数効果や従来の並列化手法との比較など評価結果を報告し，第6章で本稿をまとめる．

2. タイムワープ機構とその応用

2.1 タイムワープ機構の概要

タイムワープ機構[8)]は並列システムにおける実行順序制御機構の一つである．タイムワープ機構では，システム全体で共通な仮想時刻（virtual time）の概念を導入し，これによって処理の実行順序を表す．すべての処理は，仮想時刻順序にしたがって並列に実行される．

* 本研究は第五世代コンピュータプロジェクトの一環として行われた．

** もちろんタイムワープ機構は他の配線アルゴリズムにも適用可能である．

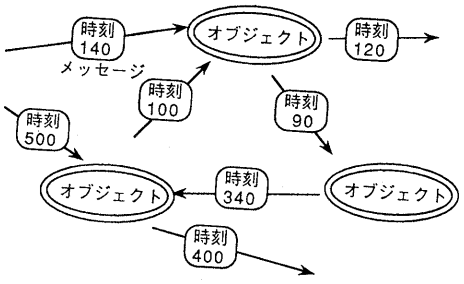


図 1 タイムワープ機構の仮定するモデル
Fig. 1 The model for the Time Warp mechanism.

いま複数のオブジェクトがメッセージ通信を行うことによって，次々と状態を変えていくモデルを仮定する（**図1**）．ここで，メッセージはタイムスタンプを持ち，各オブジェクトはタイムスタンプ順にメッセージを処理しなければならないという制約を与える．

タイムワープ機構では，メッセージはタイムスタンプ順にオブジェクトへ到着するだろうという楽観的な仮定をおく．メッセージがタイムスタンプ順に到着している限り，その処理は正しいものとして処理を進める（見込み計算）．

しかし，実際には誤った順序でメッセージが到着する場合が存在する．このような状況に備え，オブジェクトは送受信メッセージと状態の履歴を保存しておく．そして，メッセージ到着順序の矛盾を発見したところで履歴を巻き戻し（ロールバック），処理のやりなおしをする．さらに，その時点で誤って送信したメッセージがあれば，それらのメッセージを取り消す役割を持つアンチメッセージを送信する．これによって，処理結果の正当性が保証される．

なお，メモリ管理・終了検出などのため，時々大域的な時刻（GVT）を求める必要がある．詳細は文献8）に解説されている．

2.2 タイムワープ機構の応用分野

タイムワープ機構は，並列事象シミュレーションにおける事象（event）処理順序の管理機構として用いられている[6)]．また，タイムワープ機構の基盤となる楽観的な考え方は，分散データベースにおける同時実行制御（concurrency control）にも応用されている[12)]．

並列事象シミュレーションは，タイムワープ機構の仮定するモデルによってきわめて自然に表現できる．ここでは，シミュレーション対象がオブジェクトに，シミュレーション時刻が仮想時刻に対応する．シミュ

レーションは，オブジェクトに発生する事象をメッセージを用いて伝達してゆくことにより進む．なお，別の事象処理順序の管理機構としては，メッセージ待ち合わせによる方法[16),22)]も提案されている．しかし，この方法ではデッドロックの問題があり，実際の応用問題においては，タイムワープ機構の優位性が報告されている[14)]．

一方，分散データベースにおける同時実行制御とは，同時刻にトランザクションが発生した場合に，データの一貫性（consistency）を保証するための制御である．ここでは，データ（タプル）がオブジェクトに，また，トランザクション発生時刻が仮想時刻に対応する．楽観的な手法では，同じデータへの書き込み/読み出し操作は同時には発生しないだろうという予測に基づいて処理を行う．データの一貫性を破壊する処理が発生した時点で，トランザクションのやり直しを行い，結果としてデータの一貫性を保つ*．他の同時実行制御手法としては，データのロッキングを用いた方法などがあるが，楽観的な手法にくらべ，並列性が制限されるという問題が指摘されている[12)]．

これらの応用例においてタイムワープ機構や楽観的手法の適用が成功した理由は，対象問題が以下の性質をもつためと考えられる．

a）処理の発生がきわめて動的（予測困難）である．
b）また，並列環境において処理の正当性を保証するには高いコストがかかる．
c）しかし，実際には多くの処理が並列実行可能である．

すなわち，並列事象シミュレーションや分散データベースにおいては，コンパイル時に並列性を抽出するような静的な並列化手法では高い並列効果を得ることが難しい（性質a）．また，正当な処理のみを動的に抽出し実行する手法（例えば，データ待ち合わせやデータのロッキングなど）を用いても，デッドロックや並列性制限の問題が発生する（性質b）．しかし，複数の処理が同時実行できるだろうという楽観的な予測はほとんど当たる（性質c）．

したがって，逆の見方をすれば，対象問題が上記の性質を持つ場合，タイムワープ機構を適用することによって効率的な並列処理が行える可能性があると推測できる*．このことから，われわれは，タイムワープ機構がさらに多くの問題に適用できると期待している．

3. タイムワープを用いた並列配線処理

3.1 タイムワーム機構適用の背景

一般に，LSI 配線における処理時間の大部分は径路探索処理に費やされる．したがって，多くの並列 LSI 配線プログラムでは，各ネット（配線されるべき端子対）の径路探索処理を並列化（ネット内並列性の利用）している[10)]．また，さらに高い並列性抽出を目的とした複数ネットの同時配線処理（ネット間並列性の利用）も試みられている[4),5),20),26)]．

複数ネットの同時配線を行った場合，逐次配線に比べて結線率が低下するという配線品質劣化の問題が指摘されている[4)]．この理由として，あらかじめ与えられた配線順序が守られないことが考えられる．なぜなら，一般に配線順序は，配線品質への影響を考慮してノウハウに基づき注意深く決定されているからである**．

配線品質の劣化なしに複数ネットの同時配線を行う例として，PROTON[26)]の方法がある．これは，前処理によって概略配線領域が重ならないネットを静的抽出し，これらに関して同時配線処理を行うものである***(図2)．

これに対し，われわれは，タイムワープ機構の適用によって動的に並列性が抽出できる点に着目した．すなわち，タイムワープ機構の特徴である見込み計算によって，並列実行可能な配線処理を実行時に抽出できると考えた．この手法では，上記の静的手法が抽出する並列性に加え，以下の並列性も抽出可能になる．

---

* 楽観的手法では，やり直しとなったトランザクションの仮想時刻は新たなものに変更するため，必ずしもトランザクション発生時刻順に処理が行われるわけではない．したがって，厳密にはタイムワープ機構の適用例とはいいにくい．なお，文献9)では，タイムワープ機構をデータベースシステムに厳密に適用する手法について述べられている．この手法では，トランザクション発生時刻順に忠実にしたがって処理を進める．

* 残念ながら，現在のところタイムワープ機構適用が有効かどうかについての定量的な判断基準は与えることができない．これは今後の課題である．

** 最近は，配線径路が配線順に影響されにくいアルゴリズムも提案されてきている[3),10)]．しかし，配線順の影響を完全に排除することは容易ではないと考えられる．なお，与えられた配線順序どおりに配線しても未結線が発生する場合には，引き剝がし・再配線処理によって完全結線を目指す方法[18)]が多く用いられている．

*** 詳細配線処理での探索処理は概略配線領域内に限定されるため，配線順序が配線径路に影響を与えないことが保証される．なお，PROTON では，プロセッサマッピングも考慮して並列配線するネットを決定している．すなわち，概略配線領域が直接重なっていなくても，配線領域を担当するプロセッサが同じものについては並列配線しない．

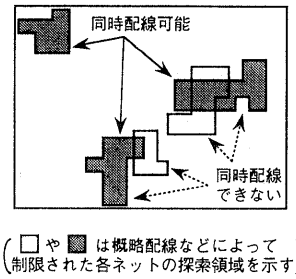


図 2 静的抽出可能なネット間並列性
Fig. 2 Inter-net parallelism statically extracted.

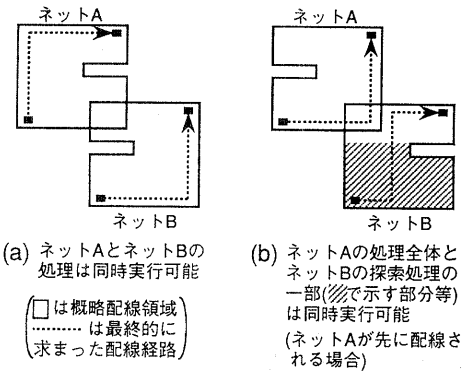


図 3 動的抽出可能なネット間並列性
Fig. 3 Inter-net parallelism dynamically extracted.

- 概略配線領域が重なるネット間でも，結果的にはおのおのの配線径路が互いに影響しない場合がある．このような場合の並列配線処理を可能にする(**図 3**(a)).
- 配線径路の一部が互いに影響し合うネットにおいても，おのおのの探索処理の一部についてみれば，並列に実行できるものがある．これらの並列処理（ネット単位より小さい粒度）を可能にする(図 3(b)*).

したがって，LSI 配線処理にタイムワープ機構を適用することで，従来手法よりさらに高い並列性が抽出できると期待される．

3.2 タイムワープ機構適用の基本指針

タイムワープ機構を組み込んだプログラムを設計する場合，タイムワープ機構の仮定するモデル（図 1）に適合するように与えられた問題をモデル化する必要がある．このモデル化に際しては，以下の 2 点が本質的な作業となる．

- 並行オブジェクトモデルによる問題表現
- 仮想時刻を用いた処理順序表現

第一点目は，問題から高い並行性（concurrency）を抽出することを目的とする．並行オブジェクトモデル[4),24)] (concurrent objects model) とは，能動的に動作する複数のオブジェクトが，メッセージを交換しながら問題を解決してゆくモデルである．並行オブジェクトを用いたモデル化によって，与えられた問題をきわめて多数の部分問題に分割する．したがって，この手法は高い並列性を抽出しようとする場合に有効なモデル化手法といえる．また，並行オブジェクトモデルは，タイムワープ機構が仮定する問題モデルとの親和性も高い．

第二点目は，処理の正当性保証を目的とする．処理の正当性を保証するためには，メッセージ処理の依存関係が守られる必要がある．タイムワープ機構では，大域的に意味を持つ仮想時刻をメッセージに割り当て，この順序を守ってメッセージ処理することにより，処理全体の正当性を保証する*．なお，実際にはオブジェクトごとに半順序関係さえ守ればよい（全順序関係を厳密に守る必要はない）ので，高い並列性が得られる．すなわち，異なるオブジェクトでは，異なる時刻のメッセージを同時に処理できる．

3.3 タイムワープルータの設計

3.2 節で示した指針に沿って行ったモデル化の内容を具体的に説明する．なお，基本アルゴリズムとしては，無格子配線アルゴリズムの一つである文献19)の手法を採用し，これを最短径路を得るように変更している．また，簡単のため，2 層配線および 2 端子ネットのみを扱う配線モデルを対象とした．各層ではそれぞれ縦配線，横配線のみを行う．

3.3.1 並行オブジェクトモデルによる問題表現

タイムワープルータでは，**図 4** に示すように，配線処理開始時の配線領域および端子を矩形で表現し，それぞれをオブジェクトとみなす．配線領域に対応するオブジェクトは，その“状態”として，オブジェクトの領域内における配線可能な領域に関する情報をも

* この図は概念図であることに注意されたい．実際に並列実行可能な探索領域は，配線アルゴリズムやデータ構造に依存するため，この図に示す範囲とは異なる場合がある．

* なお，仮想時刻のような全順序情報を持たない単純な並行オブジェクトモデルでは，処理の正当性保証のために，各オブジェクトでメッセージの送信時期を制御する必要がある．通常，局所情報のみを用いた制御（例えばメッセージ待ち合わせなど）が行われる．この場合，デッドロックが発生しないよう注意深く制御手順を設計する必要がある．

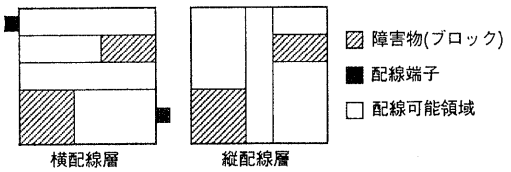


図 4 矩形表現された端子・配線可能領域
Fig. 4 Terminals and free areas represented by rectangles.

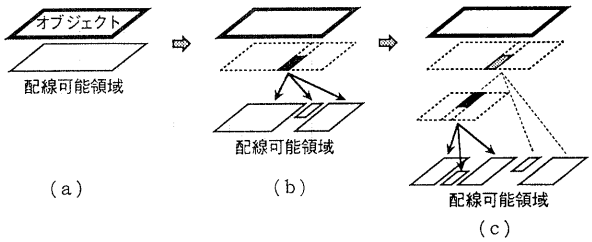


図 5 オブジェクトの状態変化
Fig. 5 State transition of an object.

つ．状態の初期値は，オブジェクト全体に対応した領域であり，**図 5**（a）に示すように一つの矩形で表される．配線処理が進行するにつれ，配線可能領域内に新しい配線径路が設定されれば，これを新たな障害物とみなし，状態はいくつかの小矩形によって表されるようになる*（図 5（b），（c））．

一つのネットに関する処理は探索処理とデータ更新処理の二つに分けられる．探索処理では，端子間の径路を与える配線領域オブジェクトの連なりを見つける．データ更新処理では，発見された配線領域オブジェクトの連なりに対し，実際の径路を割り当てるとともに，それを新たな障害物とみなすようデータ（オブジェクトの状態）を更新する．いずれの処理もオブジェクト間のメッセージ通信によって進んでゆく．以下，おのおのの処理について説明する．

●探索処理

探索処理は，スタート端子**オブジェクトから，それに隣接する配線領域オブジェクトに *search* メッセージを送信することによって開始する．

search メッセージを受信した配線領域オブジェクトは，状態を示す小矩形それぞれについて，すでに探索済みかどうかを調べる．探索済みの場合は何もしない（探索の枝刈り）．未探索の場合は，その小矩形と交差する別層のオブジェクト（および，もしあれば隣接するターゲット端子オブジェクト）に対して，探索処理が継続可能かどうかを調べ，可能な場合は，それらに *search* メッセージを送信する．以上のようにして *search* メッセージはオブジェクト間を伝播してゆく（**図 6**）．

探索処理は，*search* メッセージがターゲット端子オブジェクトに到着する（径路が存在するとき）か，またはシステム内に，そのネットに関する *search* メッセージが存在しなくなった場合（径路が存在しないとき）に終了する．ターゲット端子オブジェクトは，*search* メッセージを受信した場合，その直後に，全オブジェクトに対して *terminate* メッセージを放送する．*terminate* メッセージを受信したオブジェクトは，以後，そのネットに関する *search* メッセージの処理を行わない．これによって無駄な *search* メッセージの伝播が抑制される．

なお，各 *search* メッセージにコスト評価値 f（**図 7** に示すように決定される）を付加し，f の順に *search* メッセージが処理されるように（例えば優先度付き待ち行列を用いて）制御すれば，A* アルゴリズムにしたがった効率的な最短径路探索処理[11),27)]が実現される．タイムワープルータでは，3.3.2 項に示すように，タイムワープ機構の導入によってこの制御を実現している．

●データ更新処理

探索処理が終了し径路が発見されていれば，データ

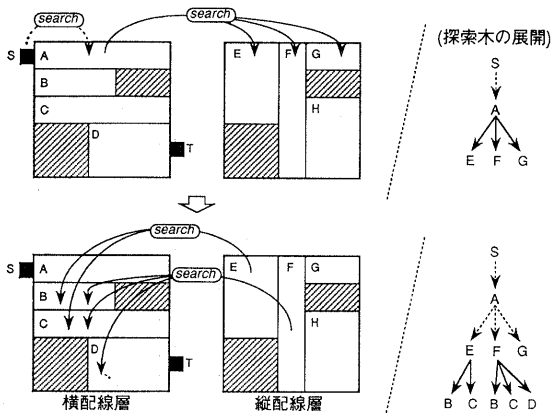


図 6 探索処理の進行
Fig. 6 Snapshots in the search phase.

* 分割された小矩形をそれぞれ新たなオブジェクトとみなす方法も考えられるが，本稿では，オブジェクトの状態の一部を示すものとして扱う．

** 一つのネットを構成する端子はあらかじめスタート端子，ターゲット端子の区別がされているとする．

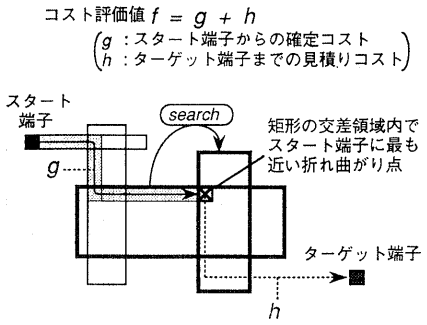


図 7 コスト評価値の例
Fig. 7 An example of f value.

更新処理に移る．この処理は，探索処理で発見された配線径路を逆に辿るように，ターゲット端子オブジェクトから *traceback* メッセージを伝達していくことで進行する．

traceback メッセージを受信した配線領域オブジェクトは，図5で示した状態変化を起こす．データ更新処理は *traceback* メッセージがスタート端子オブジェクトに到達した時点で終了する．

3.3.2 仮想時刻を用いた処理順序表現

タイムワープルータでは，以下に示す順序で配線処理が進むべきである．

- 一つのネットの探索処理については，コスト評価値の小さい順に *search* メッセージの処理が行われる（A* アルゴリズムにしたがった最短径路探索処理）．
- 異なるネットについては，あらかじめ与えられた配線順序が小さいネットから順に配線処理が行われる（配線順序を守った配線処理）．

このため，タイムワープルータでは，配線順序Oとコスト評価値fの値の組 (O, f) で時刻を表し，各メッセージにタイムスタンプとして付加した*．

任意の二つのメッセージの持つタイムスタンプを $TS_1=(O_1, f_1)$，$TS_2=(O_2, f_2)$ とすると，これらの大小は，以下に示す規則によって比較する．

IF $O_1<O_2$ THEN $TS_1<TS_2$
ELSE IF $O_1=O_2$ and $f_1<f_2$ THEN $TS_1<TS_2$
ELSE IF $O_1=O_2$ and $f_1=f_2$ THEN $TS_1=TS_2$
ELSE $TS_1>TS_2$

以上の時刻の定義に基づいて，各オブジェクトは次のように動作する．

通常のメッセージ受信時

IF 履歴中の最大タイムスタンプ
＞受信メッセージのタイムスタンプ
THEN ロールバックにより受信メッセージが正しい順序で受信された状況を再現
ENDIF
受信メッセージの種類に応じ，3.3.1 項に示した処理を実行

アンチメッセージ受信時

IF 取消対象のメッセージが評価済み
THEN ロールバックにより取消対象メッセージ評価前の状況を再現
ENDIF
未評価の状態にある取消対象メッセージとアンチメッセージを共に消去

なお，実際のプログラムでは，ロールバック発生時でもメッセージによってはやり直し不要なものがあることを利用し*，ロールバック処理の効率化をはかっている．すなわち，巻き戻し時刻以上の処理において，やり直さなければ結果が誤りとなってしまうもののみ選択的に取消処理（履歴消去やアンチメッセージ送信）を行う．

4. 実　　装

4.1 実行環境

タイムワープルータは，並行論理型言語 KL 1[25]で記述され，PIM/m[17),23)]上に実装された配線プログラムである．KL 1 は，並行オブジェクトモデルに基づくプログラムを，きわめて自然に記述できる言語である．PIM/m は MIMD 型計算機で，要素プロセッサが2次元メッシュ状のネットワークで結合されている．最大で，要素プロセッサ 256 台による構成が可能である．メモリはすべて分散管理されており，他の要素プロセッサへのデータアクセスコストすなわちプロセッサ間通信コストは高いが，台数拡張性に富む．

4.2 スケジューリング

タイムワープ機構におけるロールバック処理はコス

* *terminate* および *traceback* メッセージのタイムスタンプは，ターゲット端子が受信した *search* メッセージ（*terminate* および *traceback* メッセージ送信の原因）のタイムスタンプ値に等しくした．

* たとえば，配線領域オブジェクトに *search* メッセージが遅れて到着しても，より時刻の大きい処理済みの *search* メッセージについては，処理やり直しの必要はない（最悪の場合でも余分な処理となるだけ）．これに対し，ターゲット端子に *search* メッセージが遅れて到着した場合は，より時刻の大きい処理済みの *search* メッセージについては，処理をやり直さなければならない（さもなければ，最短径路が保証されない）．

トが高いと考えられる．したがって，タイムワープルータを効率的に動作させるためには，ロールバック発生頻度をできるだけ小さく抑えることが重要である．適切なメッセージスケジューリングを行うことはロールバック頻度低減の観点から有効であると考えられる．

タイムワープルータでは，二つのスケジューリング機構を組み込んでいる．一つはプロセッサごとに配置した局所メッセージスケジューラである．これは，プロセッサ内において，最小タイムスタンプの未処理メッセージから処理されるような制御を行う*．もう一つは大域的なスケジューリングを行う時間窓[2),15),21)]である．これは，仮想時間軸上へ擬似的な窓（時間窓）を設定することにより，処理を行うメッセージの時刻に対して一時的な上限値を与えるものである．

4.3 負荷分散

負荷分散はサイクリック割り付けの方法を採用した．すなわち，各オブジェクトに与えたシリアル番号**について総プロセッサ数の剰余を求め，これに対応するプロセッサに割り当てた．この方法は，負荷の均一化および並列性の抽出という観点からは良い方法であるが，プロセッサ間通信頻度は高くなる．なお，オブジェクトはプロセッサに割り当てられた後，別のプロセッサには移動しないもの（静的負荷分散）とした．

5. 評　価

64プロセッサ構成のPIM/m上で実験を行い，台数効果（1プロセッサ使用時に比べた速度向上），および従来の並列化手法に対する優位性について評価した．

実験には2種類のデータ（**表1**）を用いた．両データとも実際のLSIデータであり，二つの配線層を用いる．これらはもともと配線層上に格子を仮定したものであるが，実験では全ネットの配線幅を1とみなし，無格子配線アルゴリズムを適用している．また，多端子ネットはすべて2端子ネットに分解し，それぞれ配線長に関する制約を与えた．

なお，従来の並列化手法との比較評価を目的に，A*アルゴリズムによる探索処理を並列化した配線プログラム（以下では並列A*ルータと呼ぶ）を作成した．並列A*ルータの基本配線アルゴリズムおよび実行モデルは，3.3.1項で述べたタイムワープルータのものと同じであり，配線結果も一致するが，以下の点でタイムワープルータと異なる．

- *search* メッセージ処理順序の制御を，各プロセッサに分散された優先度つき待ち行列[27)]によって行う．
- 最短経路保証のため，全プロセッサ中に処理待ちの *search* メッセージがなくなったことを検出した後，探索処理からデータ更新処理に移る*．
- 前処理によって同時実行可能が判明したネットについてのみ並列配線する**．

計測した各ルータの配線処理時間を**表2**に示す．また，1プロセッサ使用時のタイムワープルータの処理時間を基準とした相対処理時間の逆数，すなわち相対

表1 実験データ
Table 1 Testing data.

データ名	データ1	データ2
格子規模	106×262	322×389
ネット数	136	85
ブロック数†（縦配線層）	528	501
（横配線層）	528	440
配線領域オブジェクト数	1,274	1,286
端子配置の特徴	分散的	集中的
ネットの端子間マンハッタン距離平均（単位：格子数）	42.4	183.9
提供元	日立製作所	NTT

† 障害物

表2 実行時間（単位：秒）
Table 2 Execution time (second).

プロセッサ数	データ1		データ2	
	タイムワープ	並列A*	タイムワープ	並列A*
1	82.06	72.53	257.02	202.31
2	46.60	44.42	136.19	117.66
4	33.03	28.98	78.92	72.13
8	17.53	19.19	49.28	46.22
16	11.29	12.84	34.37	34.95
32	5.98	9.91	21.53	25.15
64	4.25	11.83	15.59	22.37

* この制御により，1プロセッサで配線処理を実行した場合，探索処理は厳密なA*アルゴリズムにしたがい，ロールバックは発生しない．

** 配線処理前の回路データコンパイル時に決定．

* タイムワープルータでは，ターゲット端子オブジェクトが *search* メッセージを受信すれば，これが最短経路を与えるという楽観的仮定に基づいて即座にデータ更新処理に移る．

** 今回の実験では概略配線を行っていないため，3.1節で述べたPROTONの方法によるネット間並列性の静的抽出はできない．そのかわり，配線長制約を利用して探索領域を限定し，探索領域が重ならないネットをグループ化する前処理によって，ネット間並列性を抽出した．

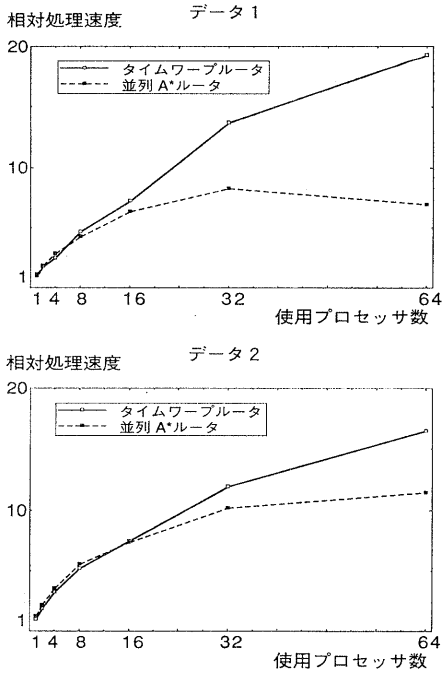


図 8 相対処理速度の比較
Fig. 8 Relative processing speed.

処理速度を図8に示す.

5.1 台数効果

タイムワープルータの台数効果は，64プロセッサ使用時にデータ1で約19.3倍，データ2で約16.5倍であった（図8）．この値は，実験データの規模が小さいことを考慮すれば良好な値といえる*.

台数効果に影響する要因を明確にするため，処理時間の内訳や発生したメッセージ数の内訳を計測した．結果を図9および図10に示し，以下に考察を加える．

プロセッサ稼働率

プロセッサ稼働率を，全処理時間に対するプロセッサ稼働時間の割合とする．これを計測したところ，データ1で約55%，データ2で約38%と低い値であった（図9）．したがって，プロセッサ稼働率が台数効果に大きく影響していることがわかる．さらに，個々のプロセッサの稼働率を計測したところ，データ1では34%から99%の間に，データ2では15%から78%の間に大きくばらついていた．これは，各

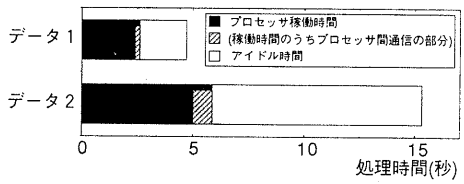


図 9 処理時間の内訳（プロセッサ平均，64プロセッサ使用時）
Fig. 9 Analysis of execution time (average, using 64 processors).

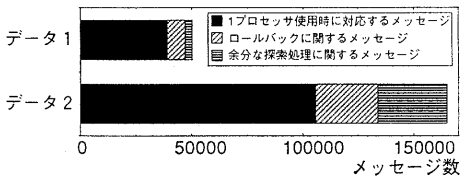


図 10 発生メッセージの内訳（64プロセッサ使用時）
Fig. 10 Analysis of the number of messages generated (using 64 processors).

プロセッサの負荷が不均等であることを示している．負荷の不均等はプロセッサ稼働率を低下させる大きな要因である*.

プロセッサ間通信時間

サイクリック割り付けによる負荷分散では，プロセッサ数をNとすると，発生メッセージのほぼ$(N-1)/N$がプロセッサ渡りのメッセージになると考えられる．したがって，プロセッサ間通信頻度は高い．しかし，メッセージあたりのプロセッサ間通信時間が，メッセージあたりに発生するプロセッサ内での処理時間に比べて小さい（測定した結果，1/5〜1/6であった）ため，実際にプロセッサ間通信時間が処理全体に占める割合は小さい（図9）．以上から，プロセッサ間通信時間が台数効果に与える影響は小さいといえる．

余分なメッセージの発生

複数プロセッサ使用時のタイムワープルータでは，1プロセッサ使用時に比べ発生メッセージ数が増加する．これは，複数プロセッサ使用時には，a)ロールバック処理によってアンチメッセージが発生し，さらに，これによっていくつかのメッセージが取り消され

* 実際，同じデータを用いた実験例[5]（ただし基本配線アルゴリズムは異なる）では11〜16倍であった．しかし，大規模データを用いた例（文献20)では24〜30倍，文献26)では17〜43倍）に比べると低い．一般に，データ規模が大きいほど，内在する並列性が高い傾向にある[4].

* データ2の場合，時間窓も稼働率を低下させる要因になっている．一般に時間窓の大きさが小さいほど，稼働率は低下する傾向にある．データ2では平均ネット長が長く，しかも端子が集中的に配置されているため，ネット間の並列性は小さい．この場合，過大な時間窓を与えるとロールバックが多発し，かえって速度が低下すると懸念されたため，窓の大きさを小さめにした．なお，データ1では十分に大きい時間窓を与えている．

ること，b）1プロセッサ使用時（逐次A*アルゴリズムに一致）では探索しないような領域に関しても探索処理を行うこと，の二つの理由による．64プロセッサ使用時の発生メッセージ数を計測したところ，a）によるメッセージはデータ1，2ともに全体の17%前後，b）によるものはデータ1で全体の6%，データ2で全体の18%*であった（図10）．したがって，これら余分なメッセージの発生が台数効果に与える影響は無視できない．しかし，プロセッサ稼働率の影響に比べると小さい．

以上から，低いプロセッサ稼働率が台数効果に最も大きく影響していること，および，この低稼働率は負荷の不均等に起因していることがわかった．サイクリック割り付けによる負荷分散手法は，一般に，データ規模が十分大きい場合には負荷がほぼ均等化される手法である．しかしながら，今回の実験で使用したデータは小規模であり，これが負荷の不均等につながったと考えられる．小規模データに対応できる負荷分散手法としては，あらかじめ各配線領域オブジェクトの負荷を見積もり**，これを考慮してプロセッサ割り付けを行う方法などが考えられる．

5.2 タイムワープルータと並列A*ルータの比較

使用プロセッサ数が少ない場合は，いずれのデータについても並列A*ルータのほうが高速であった（図8）．タイムワープルータでは履歴保存のオーバヘッドがあり，これが処理速度の差として現れたと考えられる．しかしながら，使用プロセッサ数が増えるにつれ，タイムワープルータのほうが高速になる．この理由としては，以下の2点が考えられる．

- 並列A*ルータで利用できるるネット間並列性は，静的抽出可能な箇所に限られているため，使用プロセッサ数が多い場合，全プロセッサを稼働させるだけの十分な並列性を得にくい．これに対し，タイムワープルータでは，可能な限りのネット間並列性を動的抽出する．
- 並列A*ルータでは，並列配線可能なネット群ごとに探索終了検出処理が必要である．これは大域的な同期処理の一つであり，プロセッサ数が増えるにつれコストが高くなる．

データ1を並列A*ルータによって配線した場合，32プロセッサ使用時に比べ，64プロセッサ使用時のほうが処理速度が低下しているが，これは，探索終了検出コストの影響と考えられる．

結局，タイムワープルータの最高処理速度は並列A*ルータの最高値に比べ，データ1で約2.3倍，データ2で約1.4倍であった*．また，スケーラビリティについても，タイムワープルータのほうが良好であった．以上から，タイムワープ機構のもつ動的な並列性抽出機能は，並列LSI配線処理において有効であるといえる．

6. おわりに

タイムワープ機構が，並列処理における実行順序制御機構として汎用的に利用できることに着目し，その全く新しい応用として並列LSI配線への適用を試みるとともに，有効性を評価した．

われわれは，実際にタイムワープ機構を用いた配線プログラムを分散メモリ計算機であるPIM/m上に試作し，台数効果，および従来の並列化手法に対する性能向上という二つの観点から性能評価を行った．台数効果については，64プロセッサを用いて最高で19倍以上という値を得た．これは，実験データ規模が小さいことを考慮すると良好な値といえる．また，従来の手法との比較については，大規模並列計算機ではタイムワープ機構を用いた配線プログラムのほうが高速であり，スケーラビリティも良好であるという結果を得た．今回の実験から，タイムワープ機構は並列LSI配線処理において有効に機能することがわかった．

今後の課題としては，負荷分散手法の改良のほか，概略配線の導入，多端子ネットへの対応や実行時のインクリメンタルな配線順変更機能（不適切な配線順へ対処）の追加により，システムとしての完成度を向上させることがあげられる．また，タイムワープ機構の応用分野拡大という点からは，タイムワープ機構の有効性についての判断基準を明確にするとともに，さらに多くの問題にタイムワープ機構を適用し，その有効性を実証することが必要であろう．

謝辞 本研究を行うにあたり，多くの有益な助言を

* 余分な探索メッセージ数は配線長制約に依存し，配線長制約が緩いほど多く発生する．

** 例えば，概略配線処理などにより各ネットの探索領域があらかじめ限定できる場合，配線領域オブジェクトが，多くのネットの探索領域を含むほど，その負荷が大きいと見積もられる．

* 並列A*ルータの処理時間にはネット間並列性を抽出する前処理（現プログラムでは，この部分は並列化されていない）の時間を含めていない．タイムワープルータでは，あらゆる並列性を動的に抽出するため，前処理は本質的に不要である．したがって，前処理時間を考慮すれば，タイムワープルータと並列A*ルータとの処理速度差はさらに大きくなる．

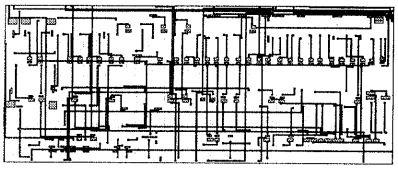
頂いた ICOT 第2研究室 伊達 博氏（現 日立製作所）および ICOT PIC-WG 委員諸氏に深謝します．また，LSI データを提供頂いた日立製作所中央研究所，および NTT LSI 研究所の方々に深謝します．

参考文献

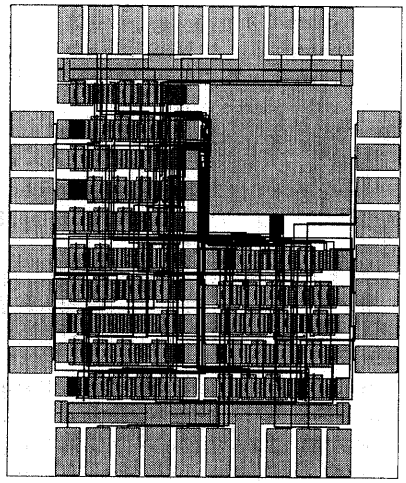
1) Agrawal, P.: Concurrency and Communication on Hardware Simulators, *IEEE Trans. Computer-Aided Design*, Vol. CAD-5, No. 4, pp. 617-623 (1986).
2) Briner, J. V. et al.: Parallel Mixed-level Simulation using Virtual Time, *CAD Accelerators*, Ambler, A. P. et al. (eds.), pp. 273-285, North-Holland (1991).
3) Brouwer, R. J. and Banerjee, P.: PHIGURE: A Parallel Hierarchical Global Router, *Proc. 27th DA Conf.*, pp. 650-653 (1990).
4) 伊達 博，大嶽能久，瀧 和男：並列オブジェクトモデルに基づく LSI 配線プログラム，情報処理学会論文誌，Vol. 33, No. 3, pp. 378-394 (1992).
5) Data, H. et al.: LSI-CAD programs on Parallel Inference Machine, *Proc. Int. Conf. Fifth Generation Comp. Syst.*, pp. 237-247 (1992).
6) Fujimoto, R. M.: Parallel Discrete Event Simulation, *Commun. ACM*, Vol. 33, No. 10, pp. 30-53 (1990).
7) Heyns, W. et al.: A Line-Expansion Algorithm for the General Routing Problem with a Guaranteed Solution, *Proc. 17th DA Conf.*, pp. 243-249 (1980).
8) Jefferson, D. R.: Virtual Time, *ACM Trans. Prog. Lang. Syst.*, Vol. 7, No. 3, pp. 404-425 (1985).
9) Jefferson, D. R. and Motro, A.: The Time Warp Mechanism for Database Concurrency Control, *U.S.C. Tech. Rep.*, Dept. of Computer Science, Univ. of Southern California (1983).
10) Kawamura, K. et al.: Touch and Cross Router, *Proc. Int. Conf. Computer-Aided Design '90*, pp. 56-59 (1990).
11) Kumar, V. et al.: Parallel Best-First Search of State-Space Graphs: A Summary of Results, *Proc. AAAI '88*, pp. 122-127 (1988).
12) Kung, H. T. and Robinson, J. T.: On Optimistic Methods for Concurrency Control, *ACM Trans. Database Syst.*, Vol. 6, No. 2, pp. 213-226 (1981).
13) Margarino, A. et al.: A Tile-Expansion Router, *IEEE Trans. Computer-Aided Design*, Vol. CAD-6, No. 4, pp. 507-517 (1987).
14) 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. 387-395 (1992).
15) Matsumoto, Y. and Taki, K.: Adaptive Time-Ceiling for Efficient Parallel Discrete Event Simulation, *Proc. Western Multiconf. Comput. Simulation—Object-Oriented Simulation Conference—*, pp. 101-106 (1993).
16) Misra, J.: Distributed Discrete-Event Simulation, *ACM Comput. Surv.*, Vol. 18, No. 1, pp. 39-64 (1986).
17) Nakashima, H. et al.: Archtecture and Implementation of PIM/m, *Proc. Int. Conf. Fifth Generation Comp. Syst.*, pp. 425-435 (1992).
18) Rosenberg, E.: A New Iterative Supply/Demand Router with Rip-up Capability for Printed Circuit Boards, *Proc. 24th DA Conf.*, pp. 721-726 (1987).
19) 佐藤政生，大附辰夫：グリッドレス・ルータ—格子を用いない二層配線径路探索手法—，電子通信学会論文誌，Vol. J 69-D, No. 5, pp. 808-811 (1986).
20) 佐野雅彦，高橋義造：分散メモリ型と共有メモリ型マルチプロセッサによる並列配線処理の性能評価，情報処理学会論文誌，Vol. 33, No. 3, pp. 369-377 (1992).
21) Sokol, L. M. et al.: MTW: A Stategy for Scheduling Discrete Simulation Events for Concurrent Execution, *SCS Multiconf. Distributed Simulation*, pp. 34-42 (1988).
22) Soulé, L. and Gupta, A.: Analysis of Parallelism and Deadlock in Distributed-Time Logic Simulation, *Stanford Univ. Technical Report*, CSL-TR-89-378 (1989).
23) Taki, K.: Parallel Inference Machine PIM, *Proc. Int. Conf. Fifth Generation Comp. Syst.*, pp. 50-72 (1992).
24) 瀧 和男，市吉伸行：マルチ PSI における並列処理とその評価—小粒度高並列オブジェクトモデルに基づくパラダイムについて—，電子情報通信学会論文誌，Vol. J 75-D-I, No. 8, pp. 723-739 (1992).
25) Ueda, K. and Chikayama, T.: Design of the Kernel Language for the Parallel Inference Machine, *The Comput. Journal*, Vol. 33, No. 6, pp. 494-500 (1990).
26) Yamauchi, T. T. et al.: PROTON: A Parallel Detailed Router on an MIMD Parallel Machine, *Proc. Int. Conf. Computer-Aided Design '91*, pp. 340-343 (1991).
27) 和田久美子，市吉伸行：マルチ PSI 上の最短経路時間の実現と評価，情報処理学会計算機アーキテクチャ研資，89-79, pp. 83-91 (1989).

付　　録

タイムワープルータの配線結果を**図11**に示す.



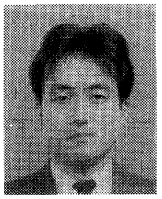
データ1



データ2

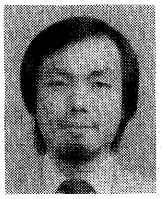
図11 配線結果
Fig. 11 Routing results.

(平成5年9月16日受付)
(平成6年2月17日採録)



松本　幸則（正会員）

昭和37年生．昭和60年京都大学工学部電子工学科卒業．同年三洋電機(株)入社．筑波研究所にてFA用画像処理検査装置の開発に従事．平成元年(財)新世代コンピュータ技術開発機構に出向．並列応用ソフトウェアの研究開発に従事．現在三洋電機(株)東京情報通信研究所に所属．電子情報通信学会会員．



瀧　　和男（正会員）

昭和27年生．昭和51年神戸大学工学部電子工学科卒業．昭和54年同大学院修士課程システム工学修了．工学博士．同年(株)日立製作所入社．昭和57年(財)新世代コンピュータ技術開発機構に出向．逐次型および並列型推論マシンと並列応用プログラムの研究開発に従事．平成2年同機構第1研究室室長．平成4年9月より神戸大学工学部情報知能工学科助教授．並列マシンのアーキテクチャ，並列プログラミング，LSI，CAD等に興味を持つ．電子情報通信学会，IEEE，ソフトウェア科学会各会員．