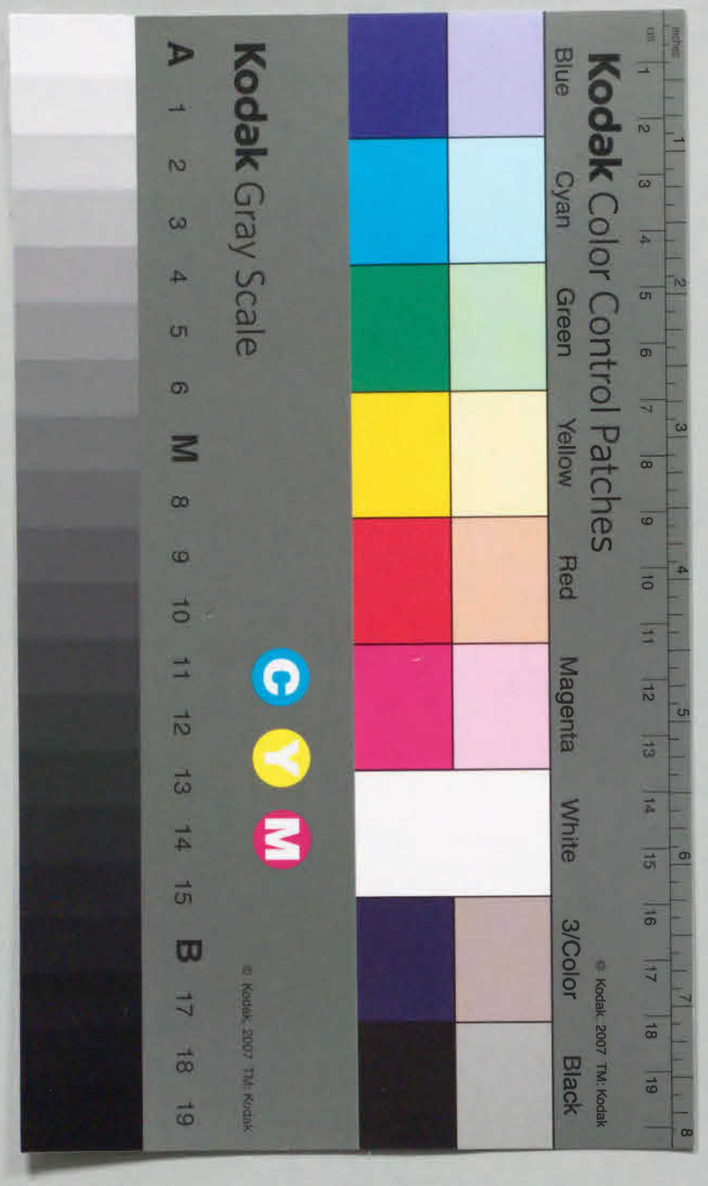


様式6

論　文　目　録

報告番号	甲　工 乙　工　第　42　号 工　修	氏　名	佐野雅彦

学位論文題目	配線問題の並列処理方式に関する研究

論文の目次

参考論文

主論文

題目「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」，
佐野雅彦，高橋義造，情報処理学会論文誌，Vol. 33，No.3，pp. 369-377 (1992).
題目「並列配線問題における並列引き剥し再配線処理の品質改善効果」，
佐野雅彦，高橋義造，情報処理学会論文誌，Vol. 36, No. 2, (1995) (印刷中)

副論文

題目「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比較」，
佐野雅彦，高橋義造，並列処理シンポジウムJSPP'91論文集，pp. 197-204 (1991).
題目「プロセッサ競合方式による並列配線処理～引き剥し処理による品質改善～」，
佐野雅彦，高橋義造，並列処理シンポジウムJSPP'93論文集，pp. 331-338 (1993).
題目「プロセッサ競合方式による並列自動配線～品質改善の試み～」，
佐野雅彦，高橋義造，DAシンポジウム論文集，pp. 181-184 (1993).
題目「並列化を考慮した多端子配線問題の部分引き剥し再配線処理」，
佐野雅彦，高橋義造，DAシンポジウム論文集，pp. 229-234 (1994).

備考

1　論文題目は、用語が英語以外の外国語のときは日本語訳をつけて、外国語、日本語の順に列記すること。
2　参考論文は、論文題目、著者名、公刊の方法及び時期を順に明記すること。
3　参考論文は、博士論文の場合に記載すること。

様式７

論　文　内　容　要　旨

報告番号	（甲）工 乙　工　第　42　号 工　修	氏　名	佐野雅彦
学位論文題目	配線問題の並列処理方式に関する研究		

内容要旨

本論文は，プリント基板やVLSI内部の配線問題の並列処理に関する研究の成果をまとめたものであり，次の６章から構成される．

第１章では，プリント基板やVLSI内部の配線問題の発達の歴史的背景と，本研究を行うに至った直接の背景について述べると共に，本研究の目的と得られた諸成果の概略を述べる．

第２章では，プリント基板やVLSI内部の配線問題とその逐次解法について述べ，歴史的な２つの並列化方式とその問題点について述べる．そして並列計算機のアーキテクチャを概説し，専用並列計算機と汎用並列計算機による並列配線方式と配線処理の現状について述べる．

第３章では，計算機アーキテクチャに対して依存性を抑えるために，アーキテクチャ固有の機能を使用せず，並列計算機が本来持っている基本的な機能（プロセッサ間通信）のみを用いて構成されるプロセッサ競合方式による並列配線処理方式を提案する．これは，計算モデルにマスタ・スレーブモデルを用いて計算機アーキテクチャに対する依存性を抑え，かつ計算粒度の粗い並列処理において高い並列性を得るための方式である．この方式を分散メモリ型並列計算機と共有メモリ型並列計算機においてそれぞれ評価した結果，分散メモリ型では６３台のプロセッサにより約３０倍の高速化を実現し，共有メモリ型では７台のプロセッサにおいて約７倍の高速化を実現した．また，両方式の比較検証を行ったことについて述べる．

第４章では，プロセッサ競合方式で問題であった配線品質に関する解決方法

として，引き剥し再配線処理の反復による経路改善を並列に実行する並列経路改善方式を提案する．これはプロセッサ競合方式を基本モデルに用いて複数の配線経路の引き剥し再配線処理を同時に行うものである．この方法では，配線コストを用いた経路探索法により配線経路間の交差・接触を許容し，ペナルティを用いた評価により配線順序に対する依存性を抑える．本方式をMIMD型並列計算機により評価した結果，配線品質の改善が確認されたことについて述べる．

第５章では，多端子ネットの配線問題において並列経路改善方式を適用し，部分引き剥し再配線のための経路探索法を提案する．これは多端子ネットの経路探索が複数の部分的な経路探索から構成されることに着目し，この部分的な探索単位による並列処理によりプロセッサへの処理の動的割り当て，及びネット内の並列性とネット間の並列性を用いた２段階の高度な並列処理を可能にする．更に，引き剥し再配線による経路改善において，配線経路を部分的に引き剥すことにより経路改善のための再配線回数を削減する．本方式を逐次プログラムで評価した結果，経路改善において探索回数の削減と，配線結果が収束するまでの経路改善回数の削減が確認されたので，これらの詳細と並列処理への適合性，及び動的処理割り当てと並列性抽出の方針についての考察について述べる．

第６章は結論であり，本研究で得られた諸結果を総括的に述べると共に，今後の課題について述べる．

様式9

論文審査の結果の要旨

報告番号	(甲 工) 乙 工 第 42 号 工 修	氏　名	佐野　雅彦
審査委員	主　査　高橋　義造 副　査　赤松　則男 副　査　矢野　米雄		
学位論文題目	配線問題の並列処理方式に関する研究		

審査結果の要旨

プリント配線基板やＶＬＳＩ内部の配線のように，膨大な数の端子を限られた数の平面上で相互に交差しないような配線経路を求める問題は配線問題と呼ばれ，高配線率の配線結果を高速に求めることが重要な課題になっている．高速処理を行うには並列計算機を用いるのが有力な手段であるが，配線問題のように高度に複雑な探索問題を効率よく並列処理することは一般に非常に難しいとされている．本論文はこの問題に関する３つの課題を解決するための研究を行い，実用に耐える並列配線処理方式を開発した結果について述べたものである．

第１の課題は多様な並列計算機の方式からできるだけ独立した，高速な並列配線方式を開発することである．本論文ではマスタースレーブモデルに基づき，多数のスレーブプロセッサが同じ配線領域内で互いに競合しながら独立に配線経路を探索する競合プロセッサ方式を提案し，この並列処理方式が共有メモリ型と分散メモリ型の何れの型の並列計算機でも高い並列処理効率えられることを検証している．

第２の課題は高速性をできるだけ損なわずに高配線率の配線経路がえられる並列配線処理方式を開発することである．本論文では並列に配線される各配線経路ごとにコストを与え，高いコストの配線を引き剥して再配線することを繰り返すことにより全配線のコストを減少させる方法を提案し，その効果を確認している．

第３の課題は多端子ネットを含む配線問題の並列処理方式の開発である．本論文では一本の多端子ネットの配線経路を求める問題を部分配線問題に分割し，これらの部分問題を並列処理することにより，高並列度の並列処理を行う方法を提案する．この方法により，経路改善のための再配線回数と，経路探索回数が減少することが検証された．

以上本研究は，高度に複雑な配線問題を効率よく配線処理するための並列処理方式について研究し，実用に耐える並列配線処理方式を確立し，その効果を検証したものであり，本論文は博士（工学）の学位授与に値するものと判定する．

配線問題の並列処理方式に関する研究

1995年3月

佐　野　雅　彦

配線問題の並列処理方式に関する研究

平成　7年　3月

佐　野　雅　彦

内 容 梗 概

本論文は，プリント基板やVLSI内部の配線問題の並列処理に関する研究の成果をまとめたものであり，次の６章から構成される．

第１章では，プリント基板やVLSI内部の配線問題の発達の歴史的背景と，本研究を行うに至った直接の背景について述べると共に，本研究の目的と得られた諸成果の概略を述べる．

第２章では，プリント基板やVLSI内部の配線問題とその逐次解法について述べ，歴史的な２つの並列化方式とその問題点について述べる．そして並列計算機のアーキテクチャを概説し，専用並列計算機と汎用並列計算機による並列配線方式と配線処理の現状について述べる．

第３章では，計算機アーキテクチャに対して依存性を抑えるために，アーキテクチャ固有の機能を使用せず，並列計算機が本来持っている基本的な機能（プロセッサ間通信）のみを用いて構成されるプロセッサ競合方式による並列配線処理方式を提案する．これは，計算モデルにマスタ・スレーブモデルを用いて計算機アーキテクチャに対する依存性を抑え，かつ計算粒度の粗い並列処理において高い並列性を得るための方式である．この方式を分散メモリ型並列計算機と共有メモリ型並列計算機においてそれぞれ評価した結果，分散メモリ型では６３台のプロセッサにより約３０倍の高速化を実現し，共有メモリ型では７台のプロセッサにおいて約７倍の高速化を実現した．また，両方式の比較検証を行ったことについて述べる．

第４章では，プロセッサ競合方式で問題であった配線品質に関する解決方法として，引き剥し再配線処理の反復による経路改善を並列に実行する並列経路改善方式を提案する．これはプロセッサ競合方式を基本モデルに用いて複数の配線経路の引き剥し再配線処理を同時に行うものである．この方法では，配線コストを用いた経路探索法により配線経路間の交差・接触を許容し，ペナルティを用いた評価により配線順序に対する依存性を抑える．本方式をMIMD型並列計算機により評価した結果，配線品質の改善が確認されたことについて述べる．

第５章では，多端子ネットの配線問題において並列経路改善方式を適用し，部分引き剥し再配線のための経路探索法を提案する．これは多端子ネットの経路探索が複数の部分的な経路探索から構成されることに着目し，この部分的な探索単位による並列処理によりプロセッサへの処理の動的割り当て，及びネット内の並列性とネッ

ト間の並列性を用いた２段階の高度な並列処理を可能にする．更に，引き剥し再配線による経路改善において，配線経路を部分的に引き剥すことにより経路改善のための再配線回数を削減する．本方式を逐次プログラムで評価した結果，経路改善において探索回数の削減と，配線結果が収束するまでの経路改善回数の削減が確認されたので，これらの詳細と並列処理への適合性，及び動的処理割り当てと並列性抽出の方針についての考察について述べる．

第６章は結論であり，本研究で得られた諸結果を総括的に述べると共に，今後の課題について述べる．

目　次

第1章

序論

現在の社会は電子技術及び電子装置によって支えられており，もしこれらが存在しなければ今の社会は存在しないと言っても過言ではない．電子装置の集大成とも言えるコンピュータの発達は我々の生活様式を大きく変化させ，現在ではコンピュータは大なり小なり殆ど電子装置に組み込まれており，装置自身はますます小型化が進んでいる．電子装置の小型化は真空管からトランジスタへの変遷のように電子デバイスの発達の歴史でもあるが，実装技術の発達の歴史でもある．電子部品を実装するためのプリント配線基板（Printed Circuit Board）の出現以前では手配線により電子部品間の結線を行っていたため，結線ミス等の不良品が発生し易く，また作業時間を長く必要としたが，プリント基板の出現により相当量の複雑な結線作業が単純な部品取り付け作業へと変化し，電子装置の小型化と大量生産を可能とした．当時プリント基板上の配線経路の決定は人手によって行われていたが，電子装置の高密度化による小型化とトランジスタに代る新たな電子デバイスである集積回路（IC）の登場により，配線経路の複雑さは急激に増大した．当時のICのトランジスタ数は少なく人手による配線経路の計算が可能であったが，大規模集積回路（LSI）の登場によりトランジスタ数は数千～数万個程度に増大し，もはや人手で配線経路を決めることは非常に困難であった．このため人手に代ってコンピュータによる自動配線処理が研究された結果，数多くの方法が開発され実用化されるようになった．しかし，自動配線処理の発達に加えて材料分野や実装技術の革新などにより，更なる電子装置の小型化と集積回路の高密度化が計算量を増加させ，その処理時間の長さが新たな問題となっている．

増加する処理時間の短縮と，高密度な状態での配線品質の維持は両立が困難であり，特に後者は更に多くの計算力が必要とする．このためにはスーパーコンピュータを用いた方法があるが，今後更に複雑化する配線処理によって必要な計算量を考慮すると，近年急速に発達した並列計算機による並列自動配線処理が有望であるとされる．しかしながら，並列処理は従来の逐次処理方法とは異なるパラダイムが必要になるため，逐次アルゴリズムをそのまま並列化するだけの並列処理では，その計算能力を十分に利用することができず，加えてこれまでのアルゴリズムの多くは並列計算機のアーキテクチャに対する依存性が高いため，他のアーキテクチャを持つ並列計算機上に実装することが難しい点が問題であった．このように，アルゴリズムの難しさと実装の難しさから並列処理アプリケーションプログラムが不足し，これによる利用者少なさが並列処理の研究促進を阻んでいる．これらの問題解決には並列処理のための新しいアルゴリズムが必要であるが，それは並列計算機のアーキテクチャに対する依存性の低いものが必要である．

一般に，プリント基板上の実装部品間の配線経路を求めることやVLSI内部の配線経路を求める問題を総称して配線問題と呼ぶ（これに対して部品配置を決定する問題は配置問題と呼ばれている）．配線問題の最も基本的な解法として，迷路法[1-3]や線分探索法[3-5]などがあり，これらを基本とした多くの逐次アルゴリズムによる経路探索法が開発されている．しかし，逐次アルゴリズムによる経路探索の並列化は並列計算機のアーキテクチャにより実装方法が異なるため，実際にはアーキテクチャに応じて各種の方法が開発されている．

配線問題の並列処理方法は大別して，専用並列計算機で処理するもの（ハードウエアルータ）と，汎用並列計算機で処理するもの（ソフトウエアルータ）とに分けられる．専用並列計算機ではハードウエア化のために複雑な経路探索アルゴリズムの採用は難しいが高速に実行できる利点があり，汎用並列計算機ではソフトウエアにより処理されるので複雑な探索アルゴリズムを使用できる利点がある．ハードウエアルータでは迷路法を用いたL-マシン[6]，PAR[7]，MAPLE-RP[8]などが代表的で，いずれも２次元メッシュ状に配置されたプロセッサによりSIMD的に動作する．汎用並列計算機では，MIMD型並列計算機が用いられる場合が多く，使用する経路探索法に応じて多くの処理方式が提案されている．

MIMD型並列計算機（以下，並列計算機と記す場合，特に指定がない場合はこれ

を指す．）における並列配線処理方式は，使用する経路探索法，計算機アーキテクチャ及び並列処理アルゴリズムにより実装方法が異なるため，ある処理方式を開発しても他のアーキテクチャの並列計算機に実装することが難しく，この実装の難しさは並列計算機の発展の障害となっている．

この問題に対して，本研究では計算機アーキテクチャに対する依存性の低い並列処理方式の開発を目的としている．これは，アーキテクチャに対して依存性の低い計算モデルを基本モデルとして並列処理アルゴリズムを開発することにより，並列計算機への実装性を高め，かつ高い並列処理性能が得られるためである．この方法の特徴は，同じ計算モデルで異なるアーキテクチャによる並列処理が実現できるため，プログラム開発者の負担が減少し生産効率が向上することにある．筆者は構成が単純で理解し易いマスタ・スレーブモデルを並列処理の基本計算モデルとして採用する立場から研究を行った結果，プロセッサ競合モデルを提案し，プロセッサ競合方式による並列配線処理方式を開発した[9-12]．この方式では並列計算機による評価により高度な並列性が得られることが確認されたが，商用の逐次型配線プログラムと比較して配線率で劣る欠点があった．この欠点を解消するため研究を行い，プロセッサ競合方式と引き剥し再配線処理を併用した並列経路改善方式を新たに開発した[13-17]．この方式は複数の配線経路を同時に引き剥し再配線処理する点で従来の並列配線処理方式と異なるものである．また，並列経路改善方式を多端子ネットの配線問題に適用するための経路探索法について研究し，部分引き剥し再配線を用いた経路探索法を提案した[18-20]．本論文ではこれらの詳細について述べる．

本論文の構成は次の通りである．第２章で，配線問題とこれまでの逐次アルゴリズムによる経路探索手法とその特徴について述べる．次に，配線問題の持つ並列性について述べ，従来の経路探索手法から並列化された経路探索手法に対してその特徴と問題点について考察する．また，現在の並列配線処理の現状について述べる．

第３章では，計算機アーキテクチャに対する依存性を抑えるために，アーキテクチャ固有の機能を使用せず，並列計算機が本来持っている基本的な機能（プロセッサ間通信）のみを用いて構成されるプロセッサ競合方式による並列配線処理方式を提案する．これは，計算モデルにマスタ・スレーブモデルを用いて計算機アーキテクチャに対する依存性を抑え，かつ計算粒度の粗い並列処理において高い並列性を得るための方式である．この方式を分散メモリ型並列計算機と共有メモリ型並列計

算機においてそれぞれ評価した結果，分散メモリ型では６３台のプロセッサにより約３０倍の高速化を実現し，共有メモリ型では７台のプロセッサにおいて約７倍の高速化を実現することができた．また両方式の比較検討を行った結果について述べる．

第４章では，プロセッサ競合方式で問題であった配線品質に関する解決方法として，引き剥し再配線処理の反復による経路改善を並列に実行する並列経路改善方式を提案する．これはプロセッサ競合方式を基本モデルに用いて複数の配線経路の引き剥し再配線処理を同時に行うものである．この方法では，配線コストを用いた経路探索法により配線経路間の交差・接触を許容し，ペナルティを用いた評価により配線順序に対する依存性を抑えることができる．本方式をMIMD型並列計算機により評価した結果，配線品質の改善が確認されたことについて述べる．

第５章では，多端子ネットの配線問題において並列経路改善方式を適用し，部分引き剥し再配線のための経路探索法を提案する．これは多端子ネットの経路探索が複数の部分的な経路探索から構成されることに着目し，この部分的な探索単位による並列処理によりプロセッサへの処理の動的割り当て，及びネット内の並列性とネット間の並列性を用いた２段階の高度な並列処理を可能にする．更に，引き剥し再配線による経路改善において，配線経路を部分的に引き剥すことにより経路改善のための再配線回数を削減する．本方式を逐次プログラムで評価した結果，経路改善において探索回数の削減と，配線結果が収束するまでの経路改善回数の削減が確認されたので，これらの詳細と並列処理への適合性，及び動的処理割り当てと並列性抽出の方針についての考察について述べる．

第６章では本論文における結論をのべる．

第２章

配線問題とその並列処理方式

２.１　まえがき

この章ではプリント基板やVLSIの配線問題とその基本的な逐次解法について述べた後，逐次解法を並列化した基本的な方法である配線領域を分割する方法と配線領域を共有して探索を並列処理する方法について述べる．またSIMD型とMIMD型並列計算機方式とその基本構成について述べ，専用並列計算機によるハードウエアルータ及び，これまでに他の研究者により発表された並列配線処理方式について紹介する．

本章では，２.２節に配線問題とその逐次型経路探索法について説明し，２.３節で逐次型アルゴリズムの高速化について述べる．２.４節では並列計算機方式と逐次型経路探索法を並列化した方式について述べる．２.５節では専用計算機を用いて並列配線処理を行うハードウエアルータについて述べる．２.６節では並列配線処理の現状について述べる．

2.2　配線問題と経路探索法

本節では，プリント基板やVLSI等の配線問題とこれまでの発表を含めた開発された経路探索法の代表的な方法について述べる．

2.2.1　配線問題

プリント基板やVLSIの開発における広義の配線問題には，部品を配置するための**配置問題**と配置された部品間を配線するための**配線問題**がある．この配置問題と配線問題の関係は**図2-1**に示すように，配置問題を解いた後に配線問題を解くという依存関係がある．従って設計側が要求する配線結果が得られない場合には配線問題又は配置問題に戻り，変更を加えた後に再試行するサイクルを繰り返す．

配置問題と配線問題をそれぞれ解くために必要な時間を比較すると，配置問題では詳細な配線経路を扱わず，配線容量や部品間の中心距離などを用いた近似モデルが用いられるが，配線問題は配置結果に従って詳細な配線経路を決定するため，配線問題を解く方が遥かに多くの計算時間を必要とする．本研究は，並列処理によりこの配線問題を解く時間の短縮を目的としている．なお，配置／配線問題をコンピュータにより自動処理する問題はそれぞれ自動配置／自動配線問題と呼ばれる

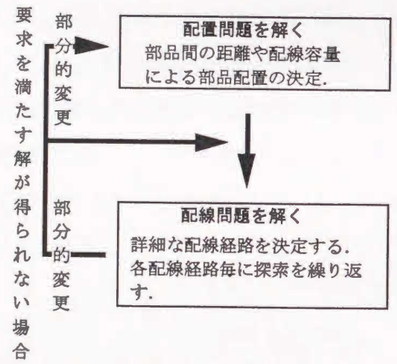
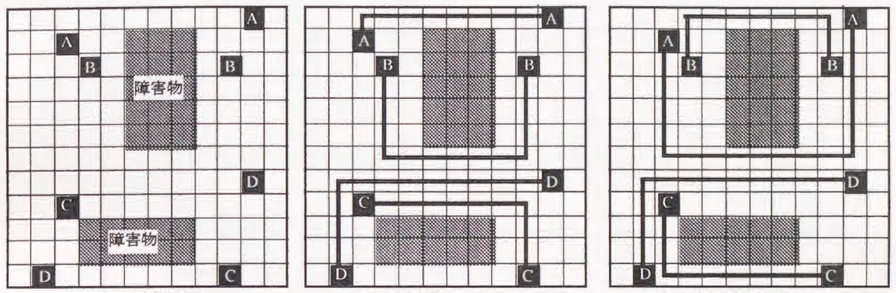


図2-1　配置問題と配線問題の関係

本論文で取り扱う配線問題では詳細配線だけを扱うものとする．従って与えられる配線問題は既に配置問題が解決されたものとなる．**図2-2** (a)に簡単な配線問題を示す．これは与えられた端子対の経路を探索する単層配線問題であり，この配線問題を解くと図(b)に示す配線経路が得られるが，これは一例に過ぎず，配線順序や

配線規則などにより図(c)のように多数の解が存在する．このような配線問題はプリント基板やLSIの内部に見られるが，両者は前提となる条件が異なるため，以下に取り上げるように，それぞれの配線問題の性質に合わせて制限された配線問題として解かれる．なお本論文で言う配線規則とは我々が期待する配線結果を得るための条件を指し，配線方向，配線経路同士の間隔，位置関係など多様なものが存在する．



(a)配線問題　(b)配線結果（その１）　(c)配線結果（その２）

図2-2　簡単な単層配線問題

（1）プリント基板の配線問題

簡単なプリント基板の配線問題は単層の配線領域で解くことができるが，部品数が増加して配線密度が高くなると，単層の配線領域では配線できなくなり，複数の配線層を用いて配線問題を解くことになる（複数の配線層を持つ配線問題は多層配線問題と呼ばれる）．プリント基板の配線問題は部品配置の自由度が高く，前提となる配線条件（配線層数，配線間隔など）により問題が変化するため，最適な配線結果を得るには多大な計算時間を要する．特に最近のプリント基板では異なる種類の部品が混在するため配線規則が複雑になり，計算時間の増加を招く傾向にある．**図2-3**にプリント基板の配線問題の例を示す．これは現在のプリント基板の典型的な配線問題であり，高密度化に不可欠なVLSIと低集積度のICが混在し，配線経路はそれらの部品間を縫うように存在する．

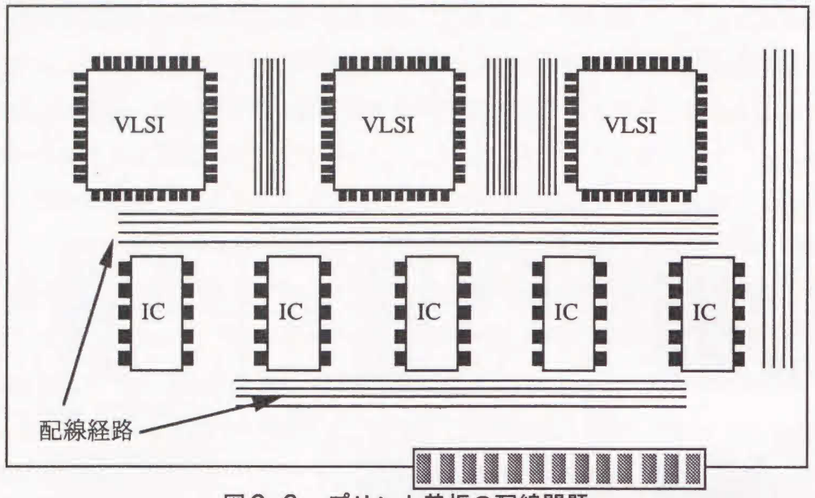


図２-３　プリント基板の配線問題

（２）ICにおける配線問題

VLSIを含むIC（集積回路）では製造方法によりカスタムICとゲートアレイICに分類される．カスタムICはプリント基板の配線問題と同様に配線条件を自由に選択することができる．ゲートアレイは予め作られている機能モジュール（セル）間を配線することにより製造され，開発コストが低く短時間に製造できる利点があるが，カスタムICのように高密度化することは難しい．

チャネル配線問題はICにおける典型的な配線問題であり，１９７１年にHashimotoとStevensによって提案された[21]．これは矩形状の配線領域中に障害物が存在せず，配線すべき端子は全て矩形配線領域の辺上に位置する配線問題であり，主にスタンダードセル方式によるゲートアレイICやカスタムICにおけるブロック間の配線において見られる．この配線問題は図２-４に示す例のように隣り合うセル間に存在する端子群を配線するものである．通常は２層及び３層の配線層を持ち，各層にそれぞれ縦横の線分経路により接続するものである．

このチャネル配線問題を更に制限したものにスイッチボックス問題がある．この配線問題では図２-５に示すように矩形の配線領域の周囲に端子が存在し，矩形領域内部には障害物が存在しない配線問題である[22].

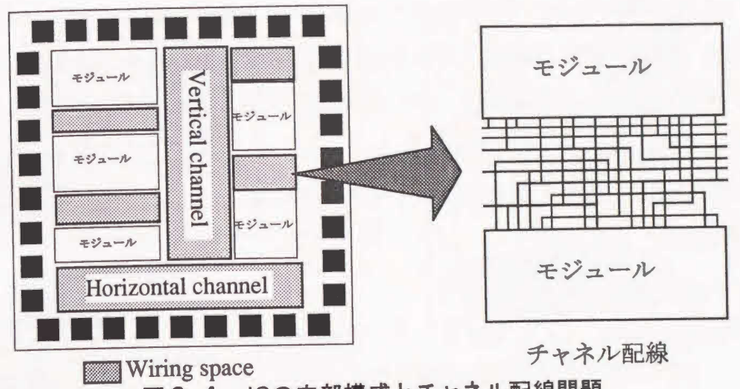


図２-４　ICの内部構成とチャネル配線問題

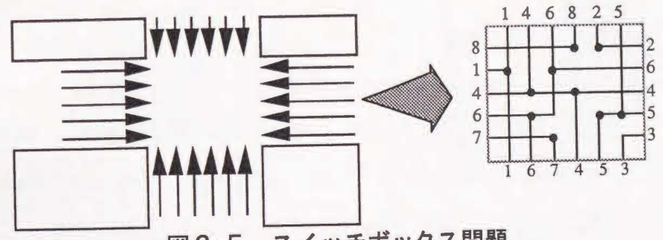


図２-５　スイッチボックス問題

２.２.２　経路探索法

（１）迷路法（Maze algorithm）

自動配線のための最初の経路探索法として提案されたものが迷路法[1-3]である．この方法は以後開発された多くの経路探索法の基本となっており，次に述べる線分探索法と並んで最も一般的な方法の一つである．この方法の最大の特徴は，最短経路が存在すればその発見が保証される点であり，配線格子を用いることで経路探索の手順を簡単にしている．なお以下では元々の迷路法であるLeeアルゴリズム[2]について述べる．

迷路法では図２-６に示すように配線経路幅ｗと配線間隔ｃによりΔ＝ｗ＋ｃによる値を用いて配線領域をΔ間隔に格子状（グリッド）に区切り，配線経路をグリッド内を通過させることで配線経路が常に配線間隔規則を満たすようにしている．従って隣接する他の配線経路との接触や交差の判定が不要となる．端子は配線経路と同様にグリッド内部に配置される．

Leeアルゴリズムでは単層の正方形のグリッドを用いた配線問題を対象としたもの

で，これは波面伝播法（wave propagation method）とも呼ばれるように，水面に石を落したときにできる同心円が拡がるように探索の開始点から周囲に探索が行われる幅優先の探索法である．探索波がゴールに到達すると，ゴールから逆に探索された部分を後戻り（バックトレース）して経路を確定する．**図2-7**に探索のようすを示す．これは2点間の経路を探索する単層の経路探索例である．

＜迷路法のアルゴリズム＞

ステップ1：初期化プロセス

探索開始点（スタート）となるグリッドを選択し，このグリッドのラベルを0にする．そしてこのグリッドの座標とラベルの値をキューに入れる．他のグリッドのラベルは空にする．

ステップ2：ラベル伝播プロセス

キューよりグリッドの座標とラベルの値を取りだし，4近傍にラベルを伝播させる．ラベルが伝播する条件は，取り出したラベルの値をLcurrent，探索する次のグリッドのラベル値をLnextとすると，Lnext=φかLnext＞Lcurrent+1の場合である．ラベル伝播が行われたとき，Lnext=Lcurrent+1としてラベルを更新し，そのグリッドの座標とラベル値をキューに入れる．これをゴールが見つかるまで繰り返す．

ステップ3：バックトレースプロセス

ゴールのグリッドからラベルが1づつ減少するように逆戻りしてスタートへ戻る．この逆戻りした経路が求める配線経路となる．

Leeアルゴリズムの計算量はLをスタートとゴール間の距離とすると$O(L^2)$となる．このため計算量の最悪値は配線領域をN×Nグリッドとすると$O(N^2)$となる．また配線領域をグリッドに分割するために，$O(N^2)$の記憶容量を必要とする．

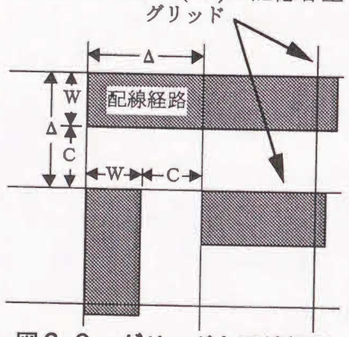
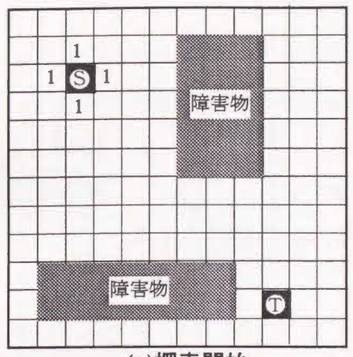
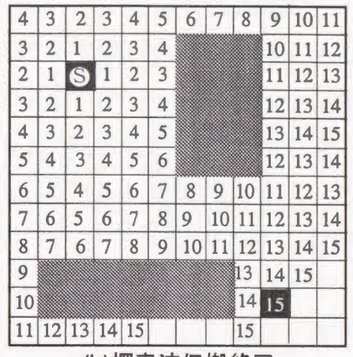


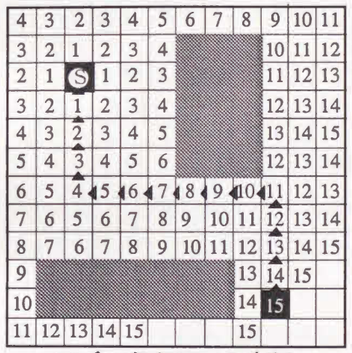
図2-6　グリッドと配線規則



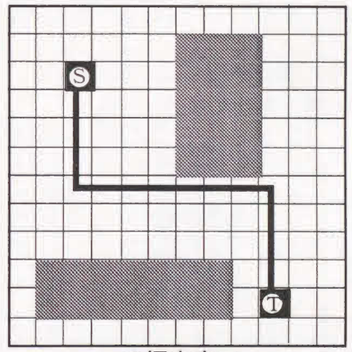
(a)探索開始



(b)探索波伝搬終了



(c)バックトレース実行



(d)探索完了

図2-7　迷路法による経路探索の例

（2）線分探索法（Line-Search Algorithm）

迷路法は最短経路が保証されるが計算に要する記憶容量が配線領域のグリッド数の２乗に比例するため，大規模な配線問題を処理する場合には無理があった．そこでより少ない記憶容量で経路探索できるアルゴリズムとして線分探索法が提案された[3-5]．これは迷路法と同様にグリッド上を探索するが，迷路法とは異なり線分単位で探索し，線分単位でラベル付けを行うことから記憶容量が少なくて済む．またグリッドは実際に存在する必要がなく仮想グリッド上で探索を行うことができるため，記憶容量と実行時間の短縮ができる．しかしながら迷路法とは異なり探索された経路が最短である保証はない．

最初に提案されたMikami - Tabuchiの線分探索法[4]のアルゴリズムは以下の通りである．

＜線分探索法のアルゴリズム＞

ステップ１：初期化

レベル０の試験線を両方の端子より四方に出し，障害物や他のネットを検出するまで直線状に探索する．

ステップ２：探索プロセス

それぞれの端子に属するレベル（i）の試験線を交互に選択し，その試験線に直交するレベル（i+1）の試験線をレベル（i）の試験線の始点（基準点）に近い順から出して直線状に探索する．これをレベル（i）の試験線が無くなるまで繰り返す．レベル（i）の試験線が無くなれば，次のレベルを（i+1）としてそれぞれの試験線が交差するまで行う．

ステップ３：経路確定プロセス

それぞれの試験線が交差した点から，それぞれの端子の方向にレベルが順次下がるように辿ることにより経路が確定される（バックトレース）．

線分探索法による２点間の単層の経路探索の例を**図２-８**に示す．なお図中の( ) 内の数値は試験線のレベルを示す．

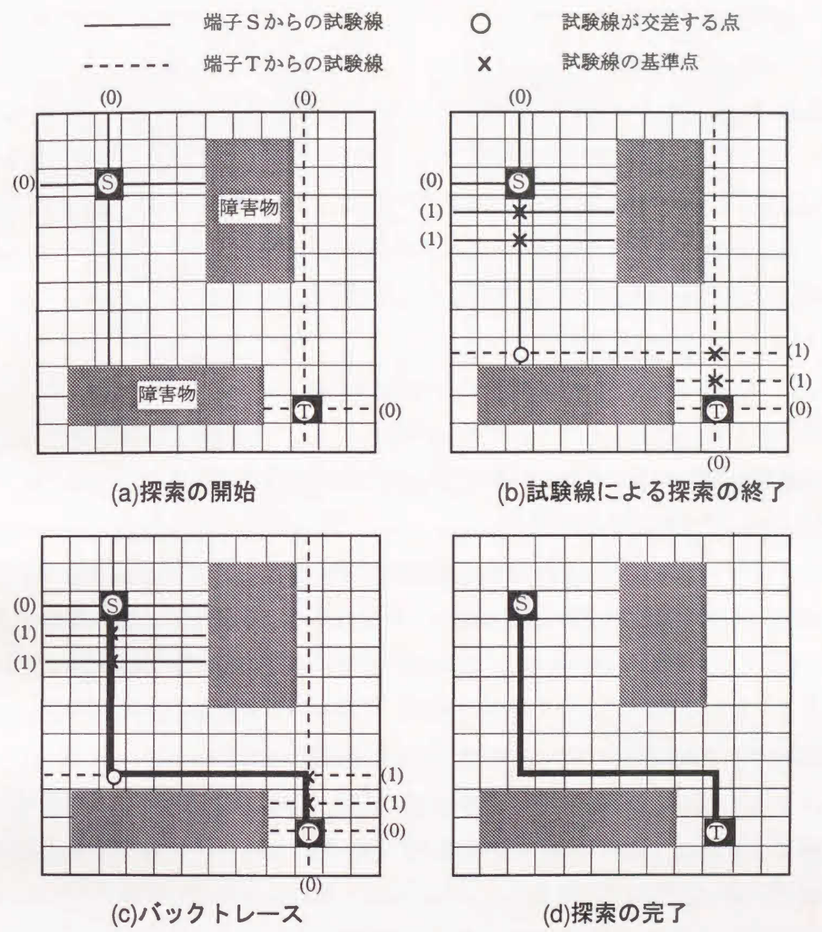


図２-８　線分探索法による経路探索の例

線分探索法の計算量は試験線の本数をｎとするとO(ｎ^2)，記憶容量はO(ｎ)であることが報告されている[3]．このように計算時間は配線領域の面積に依存しないことから，迷路法よりも高速に処理できる．

2.3　経路探索の高速化

2.3.1　高速化の目的

配線処理の高速化要求の背景には，プリント基板や半導体の集積度の向上や設計技術の進歩により配線問題の規模が拡大し，これを処理するための計算時間が増加したことと，配線品質向上のため配線規則が複雑になり問題が複雑化したことにより多くの計算を必要としたことがある．また問題規模の拡大により配線処理に必要な記憶資源が大幅に増加し，より高性能な計算機が必要とされる．

2.3.2　高速化の方法

高速化のためには（1）アルゴリズムの改良，（2）計算機の高性能化，（3）並列計算機の適用の３つの方法が考えられる．以下にこれらについて述べる．

（1）アルゴリズムの改良

アルゴリズムの改良により計算量を削減して高速化ができる．例えば迷路法では$N \times N$のグリッドでは$O(N^2)$の探索処理が必要であるが，Hadlockによれば$O(N)$～$O(N^2)$で済むアルゴリズムがあり[23]，Soukupは迷路法と線分探索法を組み合わせることにより２層配線問題において10倍～50倍の計算時間の短縮を行っている[24]．また線分探索法でも試験線の数を減らして改善した結果が報告されている[25,26]．

このような経路探索アルゴリズムの改善以外に，概略配線を行うことで探索範囲を効率よく制限し，探索時間を短縮する方法がある[27,28]．これは図２-９のように配線領域を粗いグリッドに分割し，各配線経路の経路探索の概略範囲を決定し，詳細配線は概略範囲内で行うものである．探索面積が大幅に削減されるため探索時間の短縮に効果がある．図例は２×２の配線グリッドを１つの概略配線グリッドとして概略探索を行ったものである．

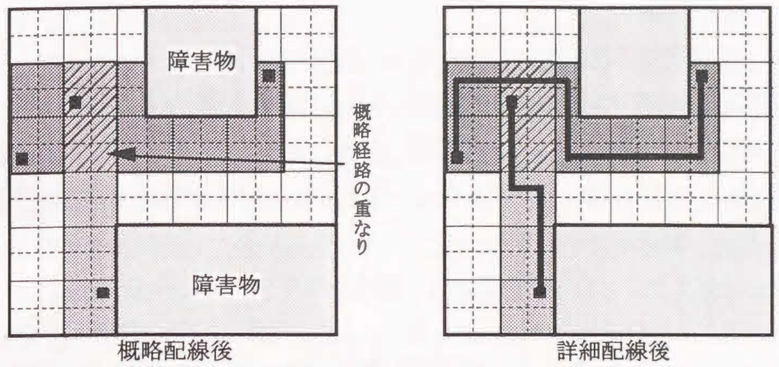


図２-９　概略配線を用いた経路探索の例

（2）計算機の高性能化

逐次処理において処理速度を高速化するもう一つの柱は，高速な計算機を使用して処理時間を短縮することである．これは計算機の性能向上に比例して処理時間の短縮ができるためである．実際，大規模な配線問題ではスーパーコンピュータを用いて処理されることがある．しかしスーパーコンピュータのような逐次計算機の性能向上には限界があり，次に述べる並列計算機の適用が研究されている．

（3）並列計算機の適用

スーパーコンピュータは最も高速な逐次計算機であるが，これを用いても最近ますます複雑化し，大規模になった配線問題を処理するには性能が不足する傾向にある．スーパコンピュータの性能は今後の劇的な向上が難しいので，より高速な処理が可能で，スーパコンピュータより価格性能比が優れている並列計算機の適用が効果的であると考えられる．しかし，並列計算機を効果的に用いるに逐次処理アルゴリズムとは異なった並列処理のためのものが必要である．

並列計算機には汎用並列計算機と専用並列計算機がある．前者を用いる場合は，そのアーキテクチャに適した並列アルゴリズムを開発し，プログラミングする必要があるが，専用計算機を開発できれば配線処理がハードウエアにより高速に処理できるため処理時間が劇的に改善される．特に配線処理中の計算量が最も多い経路探索に適用することができればその効果は大きい．

専用計算機により配線問題を処理する場合，ノイマン型の並列計算機を用いる場合が多いが，非ノイマン型の並列計算機の適用にはアナログ計算機によるものが提案されている[29]．

2.4　並列処理

2.4.1　配線問題の並列性

配線問題の並列性には**ネット内の並列性**と**ネット間の並列性**があるとされる．ネット内の並列性によってネットの経路探索を並列に処理でき，ネット間の並列性により複数ネットの経路探索を並列に処理できる．これらについて以下に説明する．

（1）ネット内の並列性

説明を簡単にするため経路探索法に迷路法を使用し，**図2-10**(a)のように配線領域を分割し，それぞれの領域にプロセッサを割り当てて並列に探索する場合について述べる．**図**(b)まず端子Sのある領域を担当するプロセッサ０が経路探索を初め，探索波を周囲に伝搬させる．**図**(c)探索波が領域の境界に達し，隣接した領域に探索波が進入すると，担当のプロセッサが経路探索を開始する．**図**(d)探索が進行し，プロセッサ１，２，３が並列探索する様子である．**図**(e)プロセッサ１，２は探索を終え．プロセッサ３が探索を継続してゴールを発見し，**図**(f)その後，経路確定を行う．これは１例にすぎないが，他の経路探索方法を使用してもネット内の並列性による並列処理が可能である．

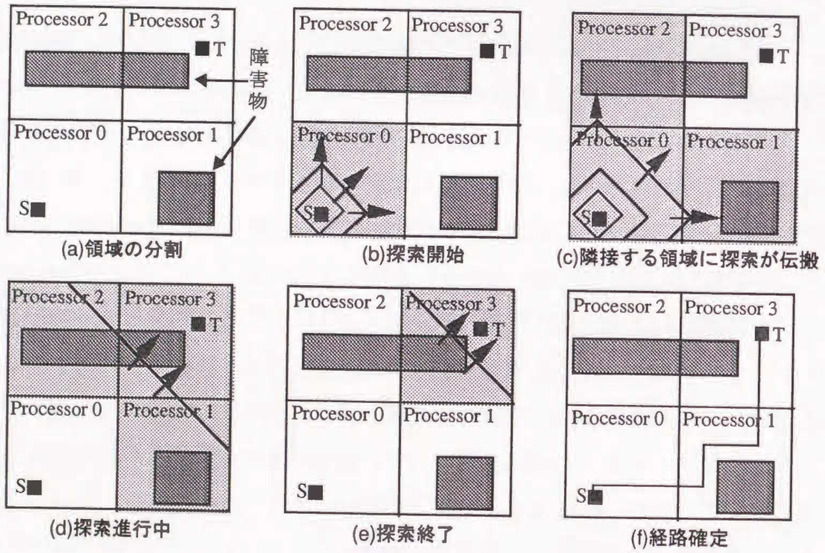


図2-10　ネット内の並列性

（2）ネット間の並列性

ネット間の並列性では互いに依存関係のないネット同士は並列に経路探索を処理できる．例えば４本のネットを配線処理する**図2-11**の場合，**図**(a)に示されるように４本のネットA，B，C，Dをそれぞれ異なるプロセッサに割り当てることで**図**(b)のように並列に経路探索できる．但し，**図**(c)のネットAとネットBのように，ネット同士が干渉する場合に並列経路探索ができないが，一般的に大規模な配線問題ではネット間の並列性は高いものと考えられる[12,30].

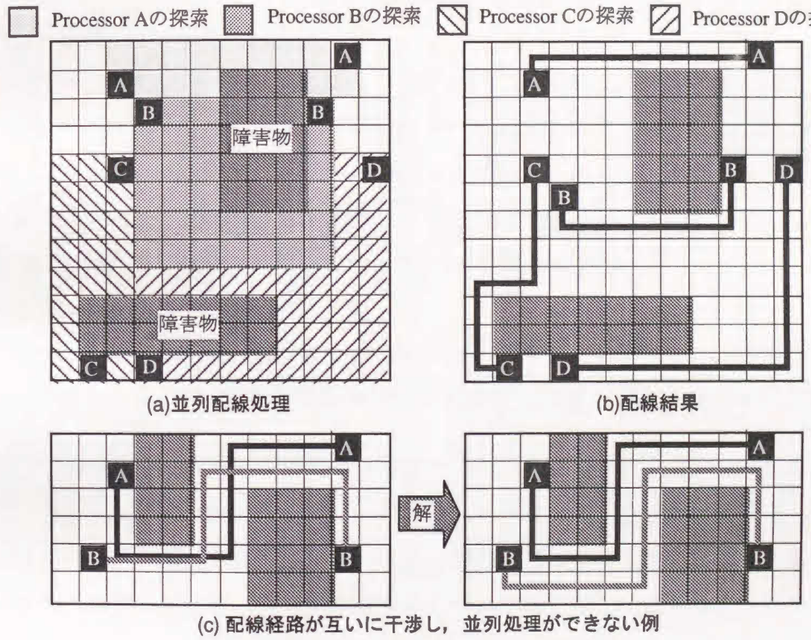


図2-11　ネット間の並列性

2.4.2　並列計算機方式

並列計算機方式はその実行形式により４種類に分類されるが，実際に採用される割合の高いSIMD（Single Instruction Multiple Data）型とMIMD（Multiple Instruction Multiple Data）型並列計算機方式の２方式について概要を述べる．

2.4.2.1　SIMD型並列計算機

SIMD型並列計算機（SIMD型）は複数のプロセッサに対して同じ命令を与え，個別データを処理させる方式で，大量のデータに単純な処理を行う目的に向いており，主にベクトル演算やパイプライン処理に用いられることが多い．図2-12に模式図を示す．

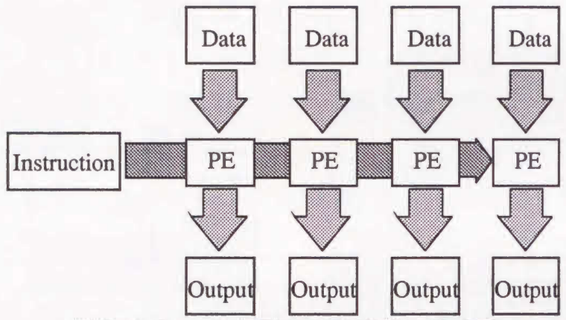


図2-12　SIMD型並列計算機の模式図

PEの構造がMIMD型並列計算機に比べて非常に簡単であるため，配線問題では迷路法による並列探索を主に用いたハードウエアルータに適用されることがある．それらの多くは，２次元メッシュ状に接続されたプロセッサ群を配線グリッドに見立て，各グリッドに１ビット程度のプロセッサを割り当てて，探索波伝播の処理を高速に行う方法が用いられる[6-8].

2.4.2.2　MIMD型並列計算機

MIMD型並列計算機（MIMD型）は各プロセッサ毎に異なる命令を用いて異なるデータを処理する方式である．SIMD型よりも複雑な処理に向いており，より汎用目的に適用することができるが，計算粒度（計算の最小単位）はSIMD型よりも粗い．図2-13(a)に模式図を示す．MIMD型はプロセッサ間通信が高速な密結合型と低速な疎結合型に分類することができ，前者には共有メモリ型並列計算機，後者には分散メモリ型並列計算機がある．また，最近では両者の特徴を取り入れた分散共有メモリ型並列計算機も研究されている．

（1）共有メモリ型並列計算機

共有メモリ型並列計算機は各プロセッサが共通のメモリを持ち，このメモリへのアクセスは共有バスを用いて行うものと通信網を通して行うものがある．図2-13(b)に前者の構成例を示す．共有メモリ型の特徴はプロセッサ間通信の高速性とデータの共有ができる点である．このため逐次型アルゴリズムの並列化が行い易い利点があるが，ハードウエアの都合上，共有メモリに多数のプロセッサを接続することができないので，大規模な並列計算機を構築することは難しい．

（2）分散メモリ型並列計算機

分散メモリ型並列計算機は図2-13(c)に示すように各プロセッサはそれぞれローカルメモリを持っており，プログラムやデータは全てその中に持ち，プロセッサ間通信は相互結合網と呼ばれる通信網で接続される．分散メモリ型は共有メモリ型とは異なりデータの共有が難しいため，データを分散させた並列処理に向いている．また，超並列計算機のような大規模な並列計算機が構成できる．

（3）分散共有メモリ型並列計算機

分散共有メモリ型は，共有メモリを一ヶ所に置かず，計算機全体に適当なまとまりで分散させる方法である．従って図2-13(d)に示すように，共有メモリ型同士を相互結合網で結合した構造を持つため，あるプロセッサが別のクラスタの共有メモリを参照する場合，通信網を経由して行われる．この方法により共有メモリ型に比べて多くのプロセッサ台数を持つことが可能である．

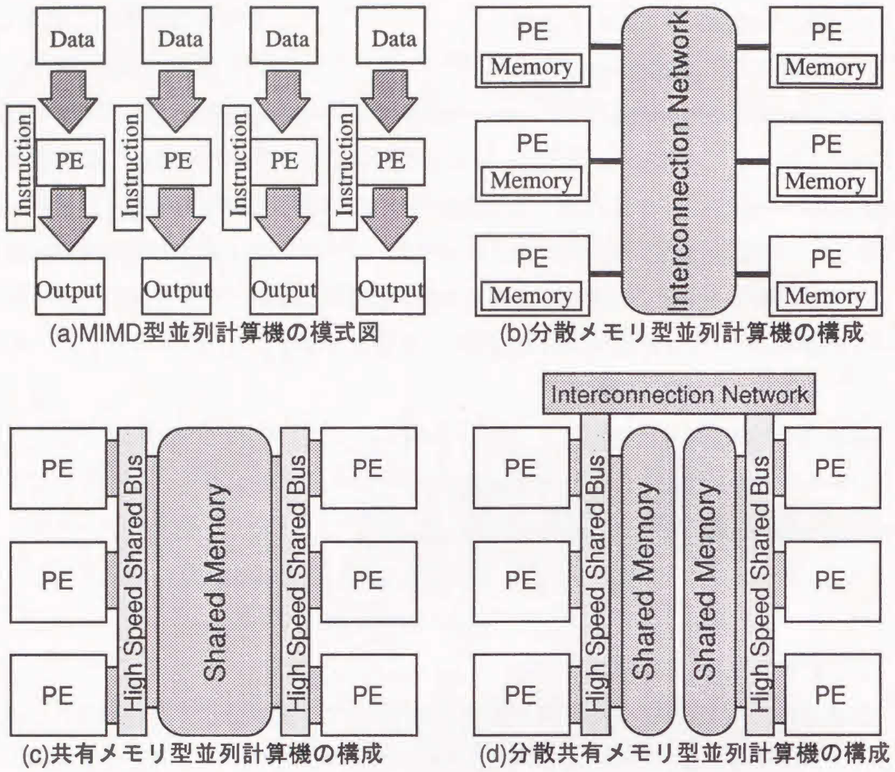


図2-13　MIMD型並列計算機の構成

2.4.3　従来の並列配線方式

配線処理の中でも時間を要するのは経路探索であり，この処理を並列化することで全体の処理時間が短縮できるが，従来の並列化方法は逐次型の経路探索アルゴリズムをそのまま並列化するものである．これらは主に配線領域を分割して並列探索を行う方法と，配線領域を分割せず経路探索を並列化する方法に分かれる．これらについて以下に述べる．

（1）領域分割法

この方法は配線領域を幾つかに分割し，それぞれにプロセッサを割り当てて経路探索を並列処理するものである．領域の分割方法により幾つかの方法があるが，平面的に分割する方法と，階層的に分割する方法がある．前者は図2-9の例に示したように，探索が分割の境界を越えると担当のプロセッサにより探索が継続される．

後者では，図2-14のように最も小さな領域から経路探索を行い，領域を跨る探索は上位のプロセッサが探索する方法である[31]．これらの方法の問題点は，有効に働くプロセッサは経路探索を行っているものだけであり，高いプロセッサ利用率を得ることが難しいことである．

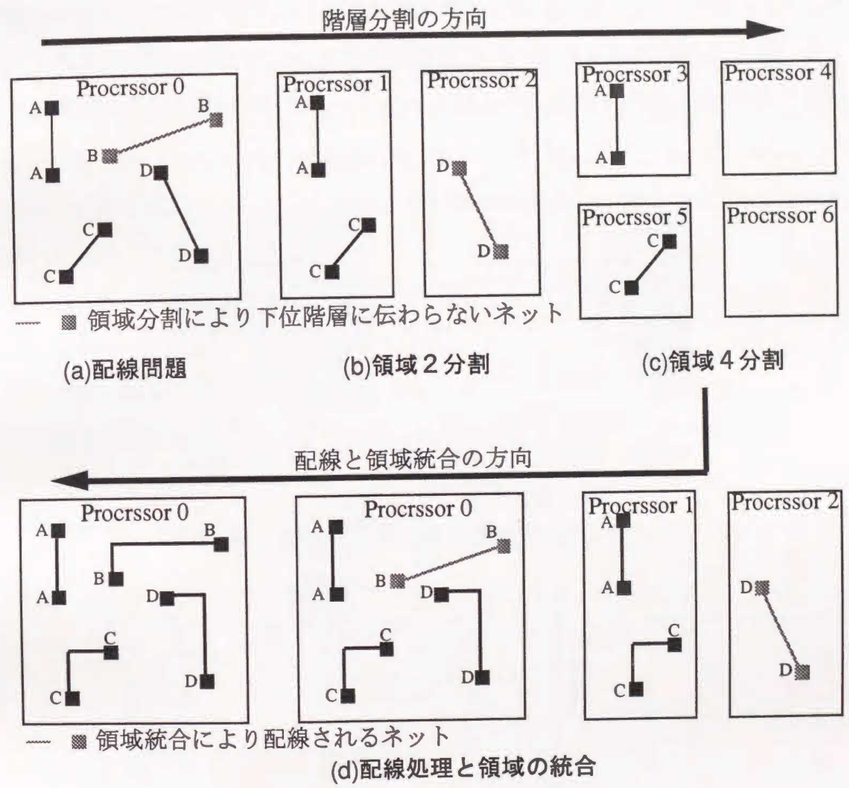
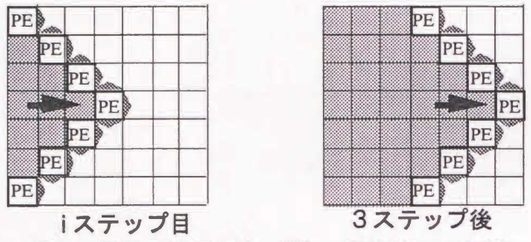


図2-14　階層領域分割法による並列配線処理

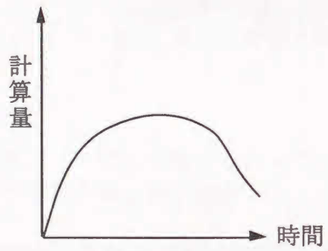
（2）共有探索法

この方法は共有メモリ型並列計算機による処理を前提としており，配線領域を全てのプロセッサで共有し，経路探索を並列処理する方法である．図2-15に迷路法を用いて説明する．逐次型の迷路法では一度に１つのグリッドの探索波の伝播が行われるが，図(a)に示すようにグリッド毎の探索をプロセッサに割り当てることで探索波の伝播が並列処理される．この方法の問題点は経路探索の並列度はネットの長

さや位置等に依存しているため，その計算量は図(b)のように時間的に変化する．仮に最大の計算量に見合う数のプロセッサを用いれば速度向上性能は向上するが，計算量が少ないときには処理を割り当てられないプロセッサがアイドル状態となり，プロセッサ利用率が低下する．逆にプロセッサ数を減らせば得られる速度向上性能が低下する．更にネット毎に必要な計算量が変化するため計算量の最大値を見積もることは難しい．また，プロセッサに割り当てられる計算粒度が細かいため，探索の矛盾を避けるための排他制御のオーバヘッドが大きく，性能向上の障害になる[17].



図(a)配線領域を共有した並列探索の例



図(b)計算量の時間変化

図2-15　共有探索法による並列探索

2.5　ハードウエアルータ

専用計算機により実現されたハードウエアルータについて述べる．ハードウエアルータでは経路探索アルゴリズムがハードウエアで実現され，高速な経路探索が可能である．アルゴリズムの単純さから迷路法を用いたものが多い．

2.5.1　L-マシン[6]

L-マシンは実際にハードウエアが作成されたものではないが，迷路法を完全にハードウエア化することを目指している．図2-16に示すように，2次元メッシュ状に配置されたL-セル，その選択／参照のためのX，Y各軸のエンコーダ／デコーダ，及び制御部から構成されてるSIMD型並列計算機である．L-マシンによる並列処理は次のステップで実行される．

（1）全セルの初期化（ブランク状態に設定）

（2）障害物の設定

（3）始点／終点の設定

（4）探索：セルの4近傍のうちブランク状態のセルがあればラベル付けし，ラベル付けされたセルがゴールであれば探索修了．

（5）バックトレースによる経路確定

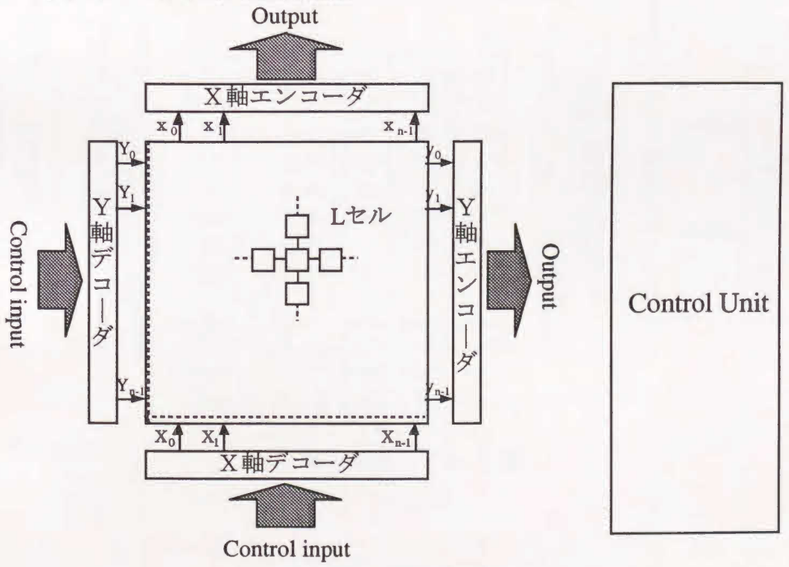


図2-16　L－マシンの構成

2.5.2 NTTのPAR[7]

Parallel Adaptive Routing(PAR)はNTTで開発されたAdaptive Array Processpr (AAP)により実現されたハードウエアルータである．AAPは図2-17に示されるように，アレイユニット，制御部，メモリから構成されるSMID型並列計算機で，アレイユニットは1ビットのプロセッサを256×256の配列で構成され，迷路法を改良して実装することにより多端子ネットの疑似最適な配線経路が得られる特徴を持つ．2.2.2で述べた迷路法を実装した場合では1MIPSの逐次型汎用計算機上でFortranを用いて実装されたものより100倍高速であることが報告されている．

PARのアルゴリズムを図2-18を用いて説明すると，まず(a)探索始点Sから探索する．(b)最初に見つかった端子T_1から探索波を出し，(c)SとT_1を最短で結ぶことができる領域をマークする．(d)この領域全体を探索始点として探索し，端子T_2を見つける．その後(e)バックトレースによりマークした領域から端子T_2までの経路が確立し，経路の分岐点が決定される．(f)それぞれの端子から分岐点までの経路を確定し，経路探索は完了する．このようにして多端子ネットの疑似最適な経路探索が行われる．

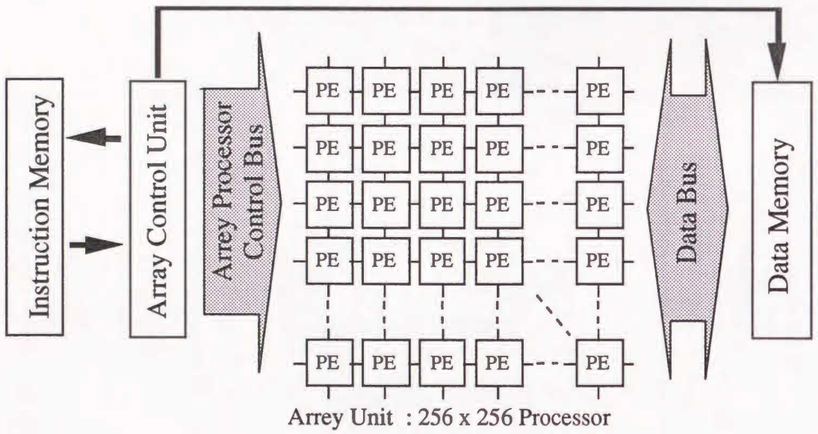


図2-17　AAPの構成

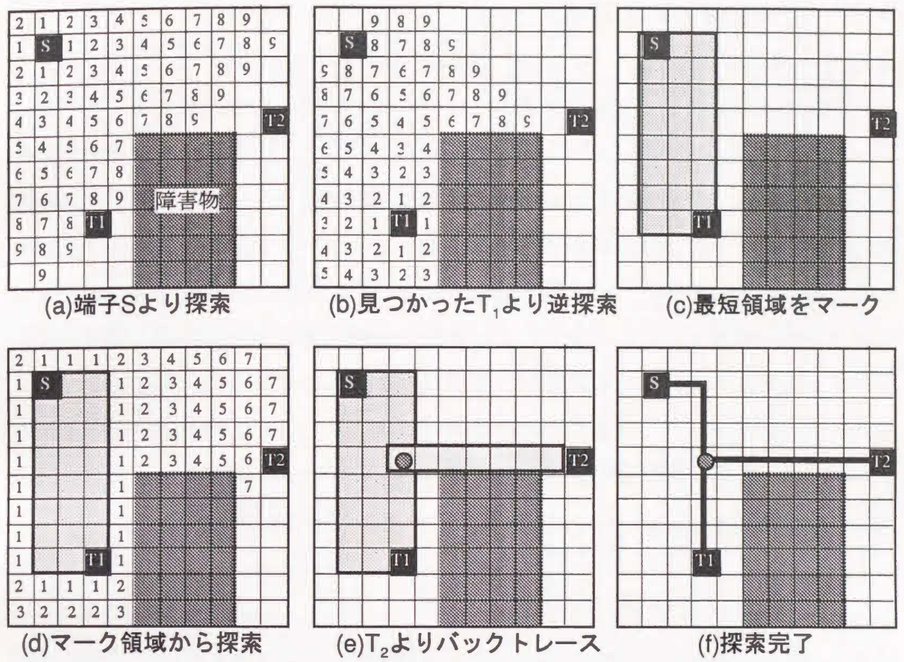


図2-18　PARにおける経路探索の例

2.6　並列配線方式の現状

2.6.1　最近の動向

逐次型経路探索アルゴリズムを並列化しただけでは充分な並列性を得ることが難しいため，最近では複数のネットを並列に経路探索する方法も併用される傾向がある．つまりネット内の並列性による並列探索ではプロセッサの有効利用が難しいため，複数のネットを同時に処理することによりプロセッサ利用率を高め速度向上性を向上させるものである．また別の動きとして並列計算機の豊富な計算力を配線品質向上のために用いるものがある．この背景にはVLSI等の高機能化よる回路規模の増大は実装面積を増加させるが回路の動作速度を低下させるため，よりコンパクトな実装が必要であり，これまでの逐次型配線方式よりも高い配線品質が要求されることがある．前者の例としてPROTON，タイムワープ方式及びプロセッサ競合配線方式，後者の例としてMAPLE-RPについてそれぞれの特徴を以下に述べる．

2.6.2　NECのPROTON[32]

NECにより開発されたPROTONは，ネット内とネット間の並列性の２段階の並列処理を行う方式であり，配線領域分割法と改良線分探索法[26]，及び概略配線による並列性の静的抽出法により実現された．PROTONにおける配線領域の分割は図２-１９(a)に示すようにＸ軸方向とＹ軸方向に矩形に分割することで，探索効率の低下を抑え，プロセッサ間通信を減らす．この方式では概略配線を行った結果を用いてプロセッサへの探索の割り当てを決定する．これを図２-１９(b)～(d)の例を用いて説明する．これは４台のプロセッサ（PE 0～PE 3）を用いて６本のネットを並列処理するもので，配線領域はPE 0，PE 1がＸ軸方向，PE 2，PE 3がＹ軸方向にそれぞれ分割して割り当てるものとし，概略配線は終了しているものとする．

まず図(b)に示す概略配線の結果を用いて，同時に処理できるネットのグループに対して割り当てられるプロセッサグループを図(c)に示すように解析し，このプロセッサグループの処理単位をルーティングユニット（RU）とする．次に図(d)に示すようにRU内部は並列処理し，一つのRUの並列処理が終ると全てのプロセッサで同期を各RUを逐次的に処理する．

PROTONは分散共有メモリ型並列計算機Cenju[33]に実装され，６４台のPEを用いて４３倍の速度向上比を達成されたことが報告されている．

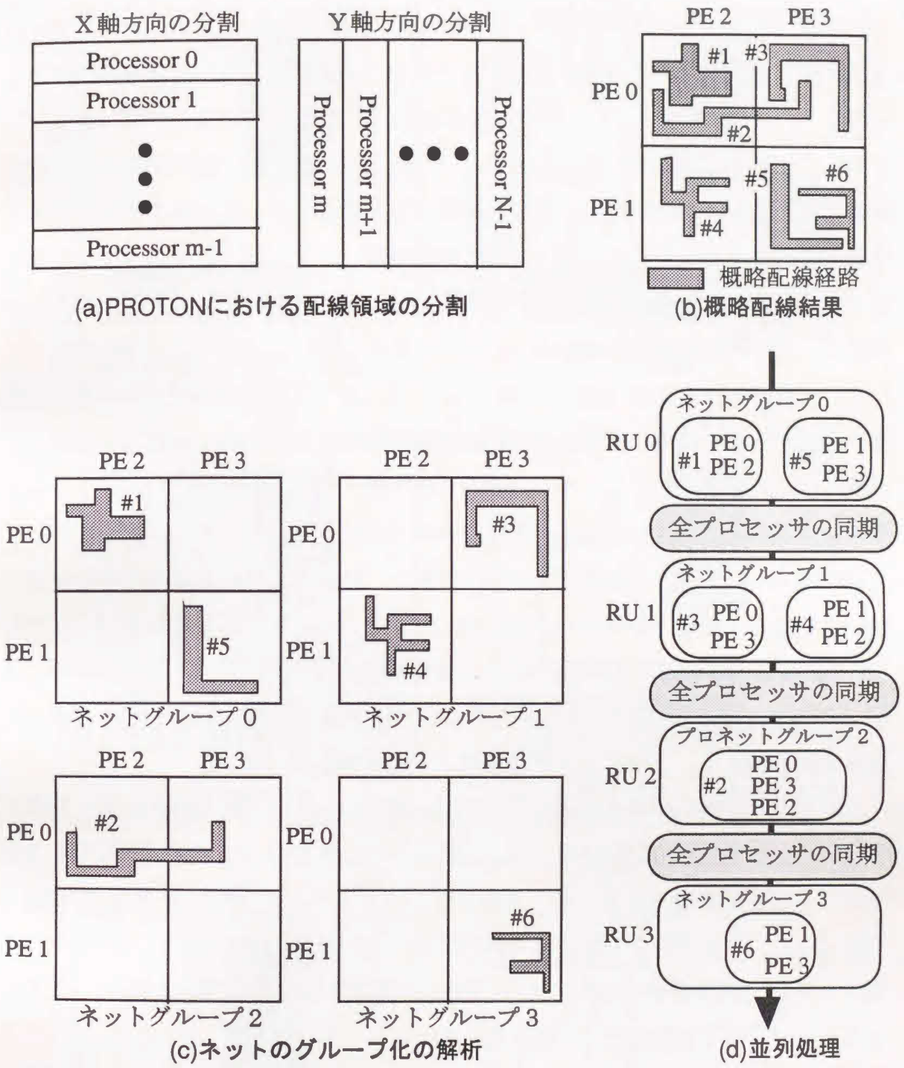


図２-１９　PROTONにおける並列配線方式

2.6.3　ICOTのタイムワープ法による無格子配線システム[34,35]

ICOTにより開発されたタイムワープ法による無格子配線システム（タイムワープ方式）は，並列処理方式に並列オブジェクトモデルを用いた並列配線処理方式[30]を採用し，各オブジェクトにシステム全体で共通な仮想時間によるタイムスタンプを与え，並列処理されるそれぞれのオブジェクトの時間的順序関係を保つことで，逐次配線方式と同じ配線順序より逐次配線方式と同じ配線品質を得る方式である．実際の並列処理ではタイムスタンプ順に処理が進行しないが，この方式では各オブジェクトが持つ処理履歴を正しい時間順序になるまでロールバックした後に再計算を行うことで時間的順序を保証する．

ここで無格子探索法について述べる．2.2.2で述べた迷路法や線分探索法は配線格子を用いるが，無格子探索法[36]では配線可能な領域を矩形に分割して管理し，経路探索は矩形領域のつながりを調べることで行う．このため端子間隔が異なる部品の実装が容易になり配線問題の柔軟性が増す．また迷路法や線分探索法と比較して経路探索の記憶容量が節約できる利点がある．**図2-20**に経路探索の例を示す．(a)前処理として配線可能領域を求めておく．この例では7つの領域に分割されている．(b)端子Sから配線可能領域の検索し矩形領域のリストを得る．探索の結果，領域1->領域4->領域5のリストが得られる．(c)得られたリストから矩形領域内の通過位置を決定し，(d)新たな配線経路により配線領域を分割する．

このように無格子探索法は配線可能領域の集合から経路探索を行うため，並列オブジェクトモデルを用いた処理方式では空き領域の断片に対してオブジェクトを割り当てる方法が有効であると考えられるが，タイムワープ方式では**図2-21**(a)に示すように，初期状態として与えられる配線可能領域にはそれぞれオブジェクトが割り当てて，配線処理の進行従って配線可能領域が分割された場合は図(b)のように同じオブジェクトが管理する方法を用いている．

タイムワープ方式はMIMD型分散メモリ型並列計算機PIM/m[37]に実装され，６４プロセッサを用いて１４倍の速度向上比が得られたことが報告されている．

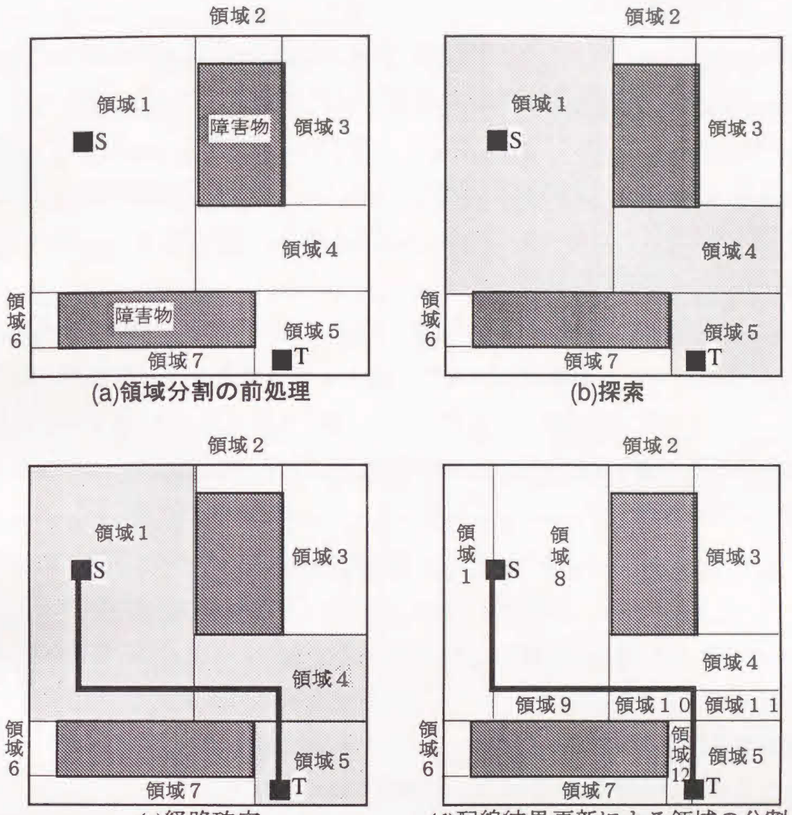


図2-20　無格子探索法による経路探索の例（続き）

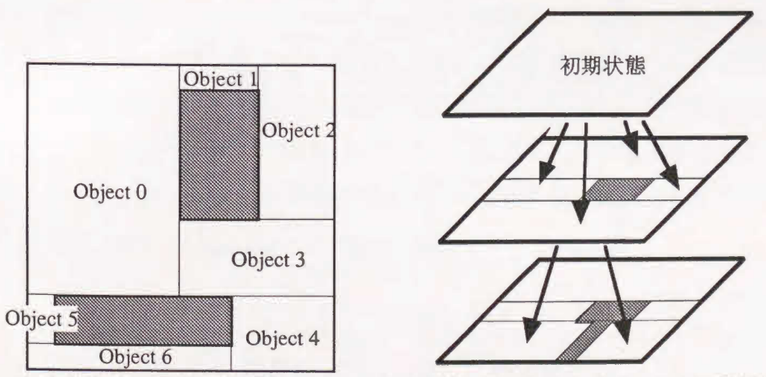


図2-21　オブジェクトの割り当てと配線可能領域の分割

2.6.4　富士通のMAPLE-RP［7］

PROTONやタイムワープ法は汎用並列計算機上に実装された並列配線方式であるが，ここで述べるMAPLE-RP（RP）は富士通により開発されたSIMD型の専用並列計算機を用いたハードウエアルータである．RPでは配線品質の向上のため，従来の逐次計算機ではで解くことが困難な配線アルゴリズムを専用並列計算機により短縮し，実用時間で解くことを可能にする．そして従来の逐次配線方式より高い配線品質が得られることが報告されている．

RPでは，制約緩和迷路法[38]という配線の制約条件をコストで表現した迷路法を用いて，**コスト＝ａ×配線長＋ｂ×ビア数＋ｃ×交差数＋ｄ×接触数**（但し，ａ，ｂ，ｃ，ｄは0以上の係数）であるコスト関数を最小化するように引き剥がし再配線処理を繰り返す．しかし，この探索法は全ての配線経路について引き剥がし再配線処理を繰り返すため莫大な計算量があり，逐次型計算機で解くことは困難である．そのため，河村等はSIMD型専用並列計算機MAPLE-RPを開発（試作機で４０９６プロセッサ）し，処理時間を短縮している．RPは１ビットのプロセッサ３２個を１チップのVLSIに集積し（１００Ｋゲート），５１２個の実装により１６３８４プロセッサを実現している．この構成を図２-２４に示す．

同じ制約緩和迷路法をワークステーションSun Sparc Station 1に実装し比較した結果，１２００倍の性能を得られたことが報告されている．

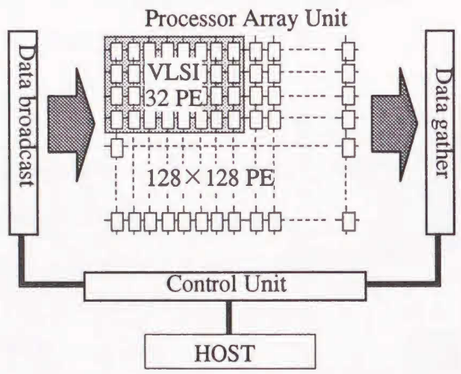


図２-２４　MAPLE-RPの構成

2.6.5　徳島大の競合プロセッサ配線方式[39]

徳島大学により開発された競合プロセッサ配線方式は各プロセッサにそれぞれ異なるネットを割り当てて並列配線させる方式であり，単層配線問題を対象としたものである．これは図２-２５(a)に示すようにマスタスレーブ方式の処理モデルを採用し，スレーブプロセッサはマスタプロセッサから配線処理を割り当てられてられる．そしてスレーブプロセッサ間の通信を削除することにより，疎粒度で不均質な並列処理に於いて高い性能を発揮することが報告されている．この方式では以下に示す処理サイクルを全てのネットに対して並列に繰り返すことで並列配線処理を行う．

ステップ１：配線処理の割り当て
　　マスタはスレーブごとに一本のネットの配線処理を割り当てる．
ステップ２：配線処理
　　スレーブは独立して配線処理する．配線結果はマスタに送られる．
ステップ３：配線結果の評価
　　マスタは先着順に配線結果を評価する．評価の結果
ステップ４：の更新
　　配線結果によりデータベースを更新する．

競合プロセッサ配線方式は徳島大学で開発されたMIMD型並列計算機Coral-68K[40]上に実装された．Coral-68Kの構成を図２-２５(b)に示す．テストデータを用いた評価の結果，６３台のプロセッサを用いて５０倍の性能を得られたことが報告されている．しかし取り扱う配線問題が単層であるため，実際の配線問題を並列処理するには不十分であった．そこで競合プロセッサ配線方式の持つ問題点を解消し，計算機アーキテクチャに対する依存性を抑えるため，第３章で提案する並列配線処理方式の研究を行った．

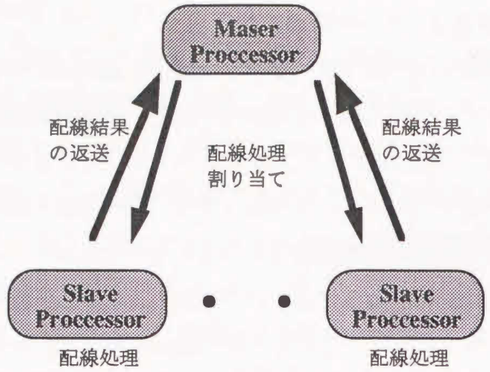


図2-25　マスタ・スレーブモデルよ並列処理

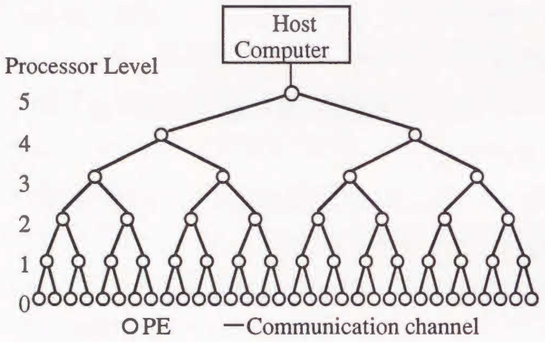


図2-26　Coral-68Kの構成

2.7　まとめ

本章ではプリント基板やVLSIの配線問題と，それらを解くための経路探索法として迷路法と線分探索法について述べた．この２つの方法は現在使用されている多くの経路探索法の原型となっている．

配線問題の複雑化と大規模化による計算量の増加により，より高速に配線できる処理方式が必要とされるが，配線処理の高速化には（１）アルゴリズムの改善，（２）計算機の高性能化，（３）並列計算機の適用，の３つの方法があり，これらの概要について述べた．

並列計算機にはその実行形式やプロセッサの構成方法により各種のアーキテクチャがあるが，それらのうちSIMD型とMIMD型並列計算機アーキテクチャについて概要を述べた．また専用並列計算機を用いたハードウエアルータについてL-マシンとAAPを用いたPAR法における並列配線方式について述べた．

これらの配線方式は逐次配線アルゴリズムをそのまま並列化したものであるため，プロセッサ利用率の点で性能を充分に発揮することができない問題がある．そのため最近の動向はプロセッサ利用率を向上させる工夫が取り入れられており，これらについての他の研究者による研究の現状を紹介した．

本論文ではプロセッサの計算粒度が粗い場合において高並列度を発揮できる処理方式を開発するため，続く第３章でプロセッサ競合方式による並列配線方式を提案する．

第３章

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

3.1　まえがき

並列計算機による並列処理は計算機アーキテクチャに対する依存性が高く，その依存性の高さが並列計算機の応用発展の妨げとなるが，アプリケーション開発をする側ではアーキテクチャが統一されているほうが開発期間と労力を短縮できる．また，現在までに開発されている並列計算機を有効活用する場合，アプリケーション開発側から見て同じ計算モデルであり，高並列性が得られることが理想的である．

そこで本章では，簡単な計算モデルであるマスタ・スレーブモデルを用いたプロセッサ競合方式による並列配線処理方式を提案する[9-12]．プロセッサ競合方式は計算機アーキテクチャに対する依存性の問題を改善する一つの方法であり，計算モデルを簡単にすることで並列処理アルゴリズムを単純にし，計算機アーキテクチャに固有の特殊機能を用いない実装により依存性を抑えるものである．

本章の構成は以下の通りである．３.２節では並列処理の目的と方針について述べ，３.３節，３.４節ではマスタ・スレーブモデルによるプロセッサ競合モデルとその基本アルゴリズム及びプロセッサ競合方式の特徴について述べる．３.５節では分散メモリ型並列計算機Cora1-68Kに実装した例とその評価結果について述べ，これに対して３.６節では共有メモリ型並列計算機TOP-1に実装した例とその評価結果について述べる．更に３.７節では両方式の比較検討を行い，それぞれの計算機方式に対するプロセッサ競合方式の有効性について検証する．

3.2　目的と方針

本節では提案するプロセッサ競合方式による並列配線処理方式の目指す目的とその方針について述べる．

3.2.1　目的

第２章で述べたように，従来の並列配線方式は計算機アーキテクチャに対する依存性が少なからず存在するため，アーキテクチャの異なる並列計算機上への実装は難しい．なぜなら，これまでに開発されている多くの並列計算機は研究目的のため多様なアーキテクチャが存在するが，このアーキテクチャの多様性に対して効果的な並列処理を可能とする計算モデルの研究が不十分であるからである．このため，複数のアーキテクチャ上で動作可能なアプリケーションの作成には時間と労力が必要とされてきた．この負担を減らす方法として，アーキテクチャに依存しない並列OSを開発・実装し，その上でアプリケーションを開発する方法[41,42]があるが，ターゲットとする並列計算機に同じOS又は同機能を持つOSが必要であり，更に複数のアーキテクチャを効率良くサポートする並列OSは未だ研究段階であるため，実用性に問題がある．このため従来の方法と異なるアプローチによる方法が必要であると判断する．

本研究の目的は，MIMD型並列計算機のアーキテクチャに対する依存性が低く，かつ高並列性が得られる並列処理方式を開発し，その有効性を検証することにある．また，すでに述べたように汎用の並列OSは研究段階にあるため，計算機アーキテクチャに低依存の並列処理方式及びその計算モデルを開発することにより，OSレベルに対する依存性を抑えることができる．

3.2.2　基本方針

MIMD型並列計算機には多様なアーキテクチャが存在するため，アーキテクチャ固有の機能を利用した計算モデルは他のアーキテクチャへの実装を難しくする．一方，簡単な計算モデルを提供することにより並列アルゴリズムを単純化し，この単純性によりアーキテクチャ固有の機能を使用せず，依存性を抑えることができる．

MIMD型並列計算機アーキテクチャの基本的な共通点はプロセッサ間が通信で結合されることから，本研究で必要とされる計算モデルはプロセッサ間通信の性質を

十分に反映可能なものが望ましい．特にプロセッサ間通信速度が低い場合（通信コストが高い）でも性能が発揮できる必要があるが，その場合プロセッサ間通信に大きな遅延を伴うために，計算粒度の細かい並列処理は同期待ち時間の増加によりプロセッサの利用率を低下させる．そこで計算粒度が比較的粗い並列処理方式に関して検討することで並列性の低下を防ぐことを基本方針とした．

3.2.3 取り扱う配線問題

取り扱う配線問題はプリント基板の配線問題を対象としており，配線問題を簡単にするために２端子ネットの配線問題を仮定している．実際の配線問題は複数の端子が存在する多端子ネットであるが，近似による問題変換で多端子ネット配線問題を２端子ネット配線問題にすることができる[3][18-20]．但し，近似化された配線問題であるために多端子配線問題と比較して冗長な配線経路が生成されることがあるが，この問題は並列配線処理の速度性能との関連が小さいため，実装する並列配線処理方式では経路探索の簡略化のために２端子の配線問題を用いている．なお，多端子配線問題に関しては第５章に述べる．

3.3 プロセッサ競合方式

提案するプロセッサ競合方式の詳細について述べる．プロセッサ競合モデルの原型は高橋，佐々木の研究[39]によるものである．ここで提案するプロセッサ競合モデルはこのモデルを更に一般化したものであり，計算機アーキテクチャ上に依存しない実装が可能になるように改良したものである．

3.3.1 マスタ・スレーブモデル

並列処理における共通の計算モデルとしてマスタ・スレーブモデルを以下に述べる理由から採用している．

（１）並列処理を行うに当たり，各プロセッサの計算粒度を決定する必要がある．密結合型並列計算機（以下密結合型）を計算モデルの前提とする場合は，プロセッサ間通信の高速性から計算粒度を疎結合型並列計算機（以下疎結合型）より細かくできるため，同期待ち時間が短くなる点で有利であるが[12]，疎結合型による細粒度の並列処理[(1)]は同期待ち時間が増加するため全体性能の低下が問題になる．このため両方式において性能を発揮するには細粒度並列処理よりも中粒度もしくは疎粒度の並列処理が適当である．

（２）疎結合型ではプロセッサ間のデータ交換の時間間隔が密結合型と比較して長くなるため，プロセッサ間でデータの一貫性（コヒーレンシ）が損なわれる問題がある．一貫性を保つにはデータ参照と更新に対してロック，アンロック操作による排他制御方法が有効であるが，そのためにはプロセッサ間通信が必要であり，データの参照が局所的に集中した場合では排他制御による待ち時間の増加により全体性能が低下する．別の方法として，プロセッサに対してデータが重複しないように割り当てる方法があるが，配線問題の並列処理では配線領域や配線経路情報など共有されるデータが多いためデータが重複しないように割り当てることは容易ではない．

一般に，粗粒度の並列処理における頻繁な同期は同期待ち時間の増加を招き，システム全体の性能を低下させるが，データの厳密な一貫性を捨てて部分的な矛盾を許容することにより不用となる同期操作を削除すればこれを防ぐことができる．これにより同期待ち時間を極力減らし，計算粒度が不均等で非同期な並列処理におい

---

(1)等粒度の場合は粒度の大小に拘らず，同期のタイミングが一致するため，同期待ち時間が非常に短くなる．また，パイプライン的に並列処理を実行できるために同期処理の必要性が小さい．

て総合性能の低下を抑えることができる．

（3）処理割り当てなどを含むプロセッサの管理を考える場合，管理を集中的に行うか分散的に行うかにより計算モデルが異なる．集中的に行う場合は単一の管理機能によりプロセッサ管理が行われる．分散で行う場合には管理機能自身も分散する必要があるが，管理機能の分散には負荷分散と機能分散の利点がある．但し処理割り当てに伴うデータ分割とデータの一貫性保持に関する対策が必要である．

計算モデルの単純性から言えば，集中的に管理する方が単純でアプリケーション開発側における計算モデルの理解度が高いが，負荷集中の危険性がある．実際は各プロセッサからのサービス要求の頻度とサービス実行に要する処理時間によって負荷集中の割合は変化するため，必ずしも負荷集中が発生して並列処理性能を低下させるわけでない．そこで計算モデルの単純性を重視して，集中管理による計算モデルにおいて実用上問題のない程度の負荷集中で並列処理可能な方法を検討する．

（4）プロセッサ管理を集中的に行う方針から，計算モデルとして図3-1に示すマスタ・スレーブモデルを用いる．マスタ・スレーブモデルではプロセッサ間通信がマスタとスレーブ間で行われるために，スレーブ間の通信を利用した計算粒度の細かい並列処理には向かず，粒度の粗い並列処理に向いている．計算モデルとしては非常に簡単で特殊な機能を必要としないため，各種アーキテクチャに対して実装が容易である．

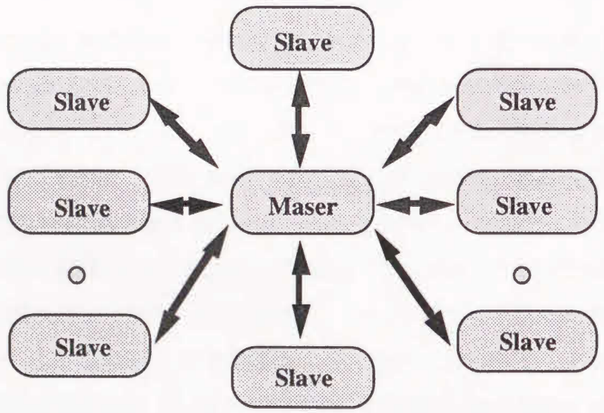


図3-1　マスタ・スレーブモデル

以上のように計算モデルとしてマスタ・スレーブモデルを採用したが，並列処理を行うためにはデータ割り当ての問題，処理分担など様々な考察すべき点がある．以後ではこれらについて述べる．

3.3.2　プロセッサ競合モデル

マスタ・スレーブモデルを用いたプロセッサ競合モデルについて述べる．ここでは取り扱う問題を特定せず一般的な問題の場合の処理アルゴリズムについて述べる．なお，配線問題における処理アルゴリズムは3.4節で述べる．

（1）モデルの構成

プロセッサ競合モデル（以下競合モデル）は前節で述べた理由からマスタ・スレーブモデルを計算モデルに採用している．競合モデルでは図3-2に示すように1台のマスタプロセッサ（以後マスタ）と複数のスレーブプロセッサ（以後スレーブ），データベース，及びプロセッサ間を接続する通信網から構成される．このモデルではMIMD型並列計算機による並列処理を仮定し，並列計算機アーキテクチャに対する依存性を低く抑え，実装容易な処理モデルを実現するために，計算モデルに対して以下に示す仮定を行うことによりモデルを簡単にしている．

・各スレーブとマスタは1対1で通信する．

プロセッサ間通信をマスタとスレーブ間に限定することでモデル上の相互結合網の構造を簡単にする．この通信モデルは各種の相互結合網上に容易に展開できるため，計算機アーキテクチャに対する依存性の抑制に有効である．

・各スレーブは互いに独立して動作する．

粒度の粗いかつ不均等な並列処理を可能にするために，スレーブ間での同期を使用しないことで同期待ちの無駄時間を減らし，スレーブの利用率向上を図る．唯一の同期はマスタと通信を行う場合であるが，マスタとスレーブは1対1に通信するため他のスレーブに与える影響は小さい．

・放送機構を使用しない．

放送機構を使用することは全スレーブに対して同期シグナルを送ることに相当するため，前述の方針に反する．また，並列計算機によっては放送機構を持たないため，依存性の点からも放送機構を使用しない．

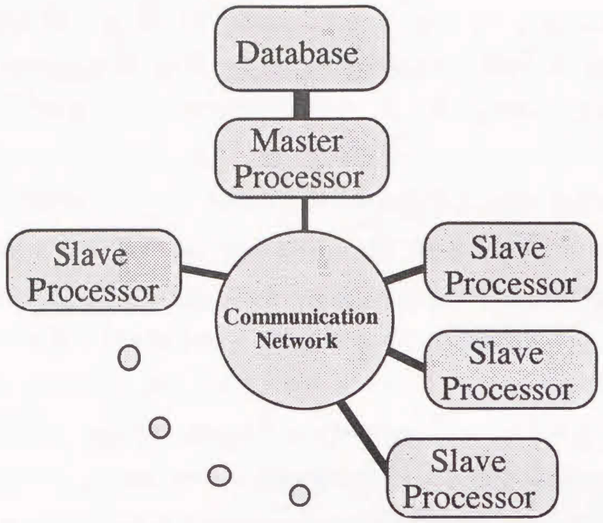


図3-2　プロセッサ競合モデル

次にプロセッサ競合モデルのそれぞれの役割について述べる．

マスタ：

マスタはスレーブへの処理の割り当て，スレーブの処理結果の評価及びデータベースの管理を主体とする．マスタはスレーブからの各種要求を受け取り，これを処理する．

スレーブ：

スレーブはマスタから与えられる処理を実行し，その処理結果はマスタへ送る．スレーブの唯一の通信先はマスタであり，スレーブが割り当てられた処理の実行に必要な各種データはデータベースから得る．

データベース：

データベースは並列処理に必要なデータや処理結果を保存する．データベースは全てのプロセッサより参照可能であるが，データベースの更新はマスタのみが可能とする．これによりデータベース内部の矛盾発生の防止と更新のための排他制御を省くことができる．なお，並列計算機内部でのデータベースの存在場所は任意である．

ネットワーク：

マスタとスレーブが1対1で通信するため通信経路はマスタとスレーブ間にあれば良い．従って任意のネットワークトポロジー上に展開可能である．事実上殆のMIMD型並列計算機がプロセッサ間同士で通信可能であるためネットワーク網に対する依存性は小さい．また，各スレーブは互いに独立して動作するためスレーブ間のプロセッサ間通信が不要になる．

（2）基本アルゴリズム

図3-3に競合モデルにおける並列処理の流れを示す．競合モデルではマスタがスレーブに処理を割り当て，スレーブはデータベースを参照して他のスレーブと独立に処理を実行し，その処理結果をマスタに送る．マスタは受け取った処理結果を評価し，その結果を用いてデータベースを更新した後，次の処理をスレーブに割り当てる．この一連の手順を処理が無くなるまで繰り返す．

このアルゴリズムでは各スレーブが他のスレーブと独立に処理を行うため，割り当てられた処理データが他のスレーブのものと部分的に重複する場合に，スレーブ間の処理結果に食い違いが生じる可能性があるが，マスタがスレーブからの処理結果を先着順に評価し，後から評価される結果に矛盾がある場合に新しい情報を与えて再処理させることでこの問題を解決するようにしている．

（3）競合モデルの特徴

競合モデルの特徴は，マスタ・スレーブモデルに幾つか仮定を行うことにより，計算機アーキテクチャだけでなくOSに対する依存性も低くなるため，様々な並列計算機上に実装可能である．プロセッサ間通信がマスタとスレーブ間だけで行われるためにスレーブ間で同期を取る必要がなく，各スレーブの処理も非同期に実行されるためにプロセッサの利用率が高い．また，マスタ・スレーブモデルは直感的に解り易い計算モデルであるためアルゴリズムの見通しが良くなるなどの利点がある．

3.3.3　ネット割り当て法

配線問題には2.4.1節で述べたネット内の並列性とネット間の並列性が存在する．配線問題の並列処理ではこれらの並列性をいかに利用するかが重要である．競合モデルではマスタ・スレーブモデルを用いることから比較的粗い粒度の処理が好ましい．そこで，計算粒度がネット内の並列処理よりも粗いネット間の並列性が競合モ

デル向きであると判断し，ネット間の並列性を用いた**ネット割り当て法**を用いた．ネット割り当て法では処理の最小単位を一本のネットとして１台または複数台のスレーブに割り当てる方法であり，複数ネットの同時配線が可能である．この方法ではネットの独立性に応じてネット間の並列性が向上するため全体の並列性が向上する．

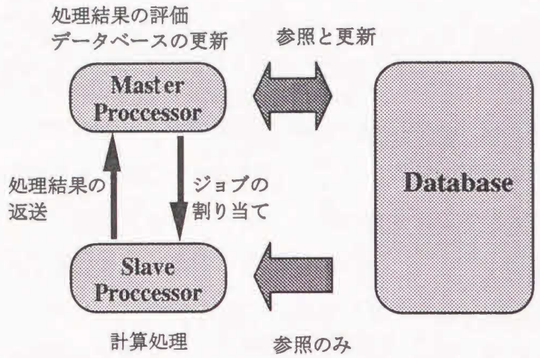


図３-３競合モデルにおける並列処理の流れ

３.４　アルゴリズム

プロセッサ競合方式における並列配線処理方式のアルゴリズムにおける各部の詳細について述べる．

３.４.１　各部の詳細

（１）配線領域の割り当て

各スレーブは与えられる１本のネットデータを受け取るごとに配線処理を行うが，各スレーブの担当する配線領域の範囲の決定が問題になる．つまり，割り当てられたネットの経路探索範囲は配線領域全体に広がる可能性があり，配線領域全体を複数に分割して各スレーブの担当範囲を決定した場合では，一本のネットの経路探索が複数に分割され，隣接する配線領域を担当するスレーブと通信することにより経路探索を行う必要が生じる．これはプロセッサ競合モデルの仮定に反するので，スレーブの処理範囲は配線領域全体とする．このようにしても実際にスレーブが経路探索する範囲は与えられるネットデータに依存することから局所的になり，スレーブ全体で見れば配線領域の参照は配線領域全体に分散するため処理結果が矛盾する割合は小さいと考えられる．

（２）データベースの参照と更新

各スレーブは共通な配線領域上で配線処理を行うので，各時点における全ての配線結果を保存するための共通領域をデータベース中に設け，このデータベースの管理はマスタが行う．従ってスレーブはデータベースを参照しながら配線処理を実行するが更新はできない．

この方法では**図３-４**に示すように，あるスレーブが配線中に別のスレーブの配線結果が新たにデータベースに加えられることがあり，各スレーブの配線結果に矛盾（配線経路の交差や接触）を生じることがある[1]．この矛盾はマスタによって検出され，**（５）**で述べる再配線処理によってスレーブが解消する．

---

[1]この様にして起きる矛盾をスレーブ間の競合による矛盾（以下競合による矛盾）とする．

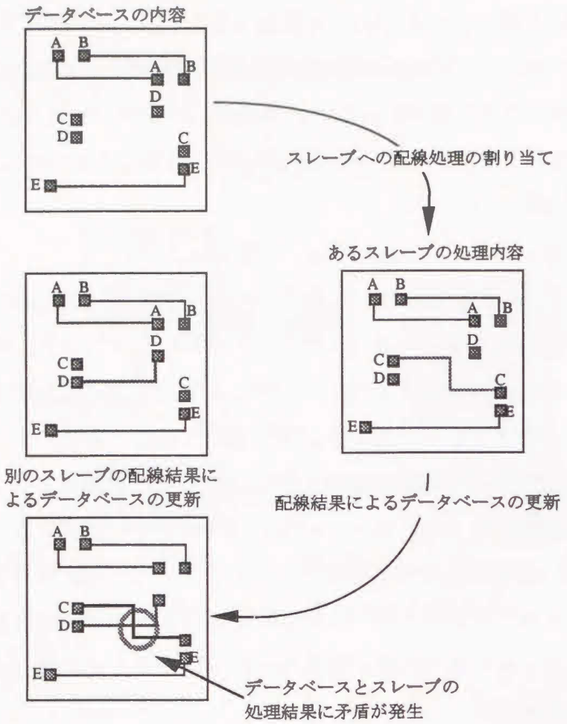


図３-４矛盾発生の例

（３）配線処理

各スレーブは与えられたネットデータをデータベースを参照しながら配線処理（経路探索処理）を実行する．配線処理中に生成される一時的なデータは全て各スレーブでローカルに処理され，データベースを更新しない．

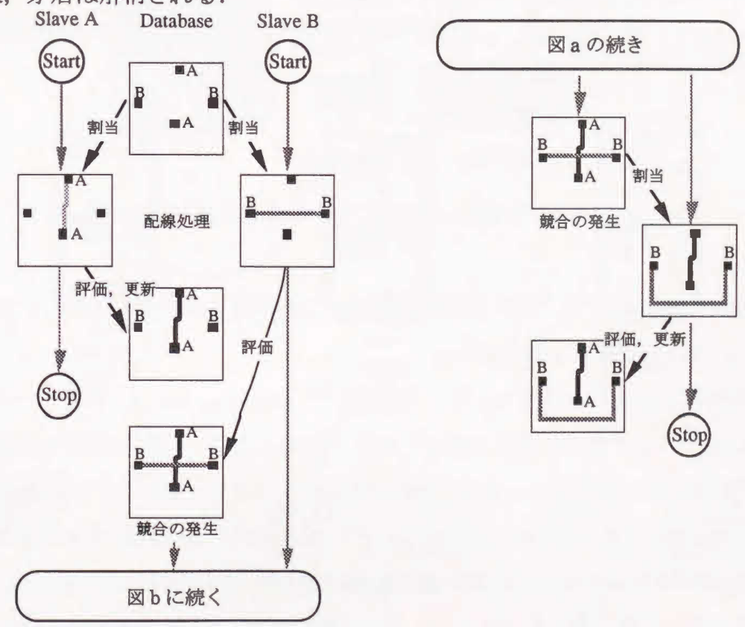
（４）配線経路の評価

配線結果を評価するには幾つかの方法が考えられるが，最も単純な方法として，データベース中の配線領域情報と比較することにより配線結果を評価する方法を用いた．この評価では評価する配線経路の上をトレースすることにより行う．トレースの結果，他の処理済みの配線結果と交差・接触していなければ正しい配線結果とする．

（５）再配線処理

競合による矛盾は次のように解消される．マスタはスレーブからの配線結果を先着順に評価する．評価の結果，矛盾が生じていればその配線経路を処理したスレーブに再配線処理を指示する．再配線処理を指示されたスレーブは再びデータベースを参照しながら配線処理を実行する．この例を図３-５に示す．この例では単層の配線を２台のスレーブA,Bで行った場合である．

図3-5(a)は，２台のスレーブにネットデータが割り当てられ，並列に配線処理を実行する場合である．しかし各スレーブは互いに独立して配線処理するため，互いの経路が無いものと仮定して配線処理する．それぞれのスレーブが配線処理を終了し，マスタが各スレーブは配線結果を評価する段階で，A,Bの順で配線結果が評価されたとすると，スレーブAの評価結果は矛盾はないがスレーブBの評価結果ではスレーブAの配線経路と交差する．図3-5(b)は交差を検出したマスタがスレーブBに対してそのネットの再配線処理を指示する様子であり，スレーブBによる再配線処理の結果，矛盾は解消される．



(a) 配線結果の矛盾の発生　　(b)再配線による矛盾の解消

図３-５　競合による矛盾の発生と再配線処理

3.4.2 アルゴリズム

プロセッサ競合モデルにおける並列配線処理の流れを図3-6を用いて説明する．

（1）マスタは各スレーブに異なる1本のネットを割り当てる．

（2）割り当てられたスレーブはデータベースを参照して配線処理を実行する．

（3）配線結果はマスタに送られる．

（4）マスタは先着順に配線結果を受け取りデータベースを参照して評価する．このとき配線結果がデータベースにある他の既配線経路と交差する場合，このネットを再処理（再配線）するようにスレーブに指示する．そうでなければ既配線経路としてデータベースを更新する．

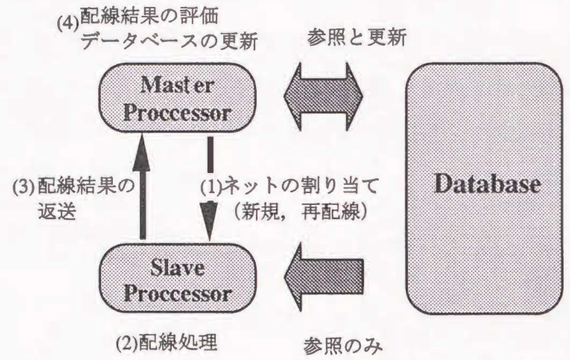
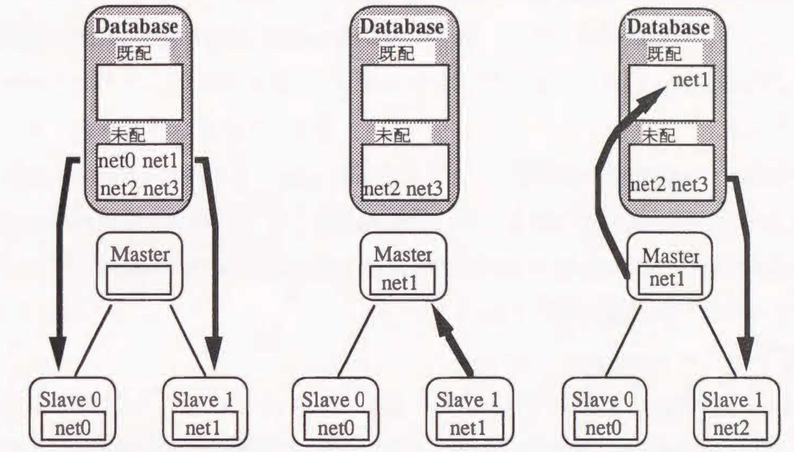


図3-6 プロセッサ競合モデルにおける並列配線処理の流れ

図3-7ではこのアルゴリズムの実行例を示す．この例は4本のネットを2台のスレーブを用いて処理する場合である．

(a)は初期状態であり，この後マスタは全スレーブ（Slave 0, Slave 1）にネットデータ（net 0, net 1）をそれぞれ割り当てる．割り当てられたスレーブは独自に経路探索を行って配線処理を実行する．(b)は配線処理を終えたスレーブがマスタに配線結果を送り，マスタはこれを評価する状態である．(c)は評価により配線結果に矛盾が無ければ，マスタはスレーブに新しい未配線ネットを割り当てた後，データベースを配線結果を用いて更新する様子である，もし配線結果が他の配線経路と矛盾している場合，図(c')に示すようにスレーブに同じネットデータを割り当てて再配線処理させる．図(d)は更に並列処理が進行した状態である．これはスレーブ1におけるnet 1

の配線処理が終わり次のnet 2を配線処理中であり，スレーブ0においてnet 0の配線処理が終了しマスタが配線結果を評価する段階である．図(e)は全ての配線処理が終了した状態である．



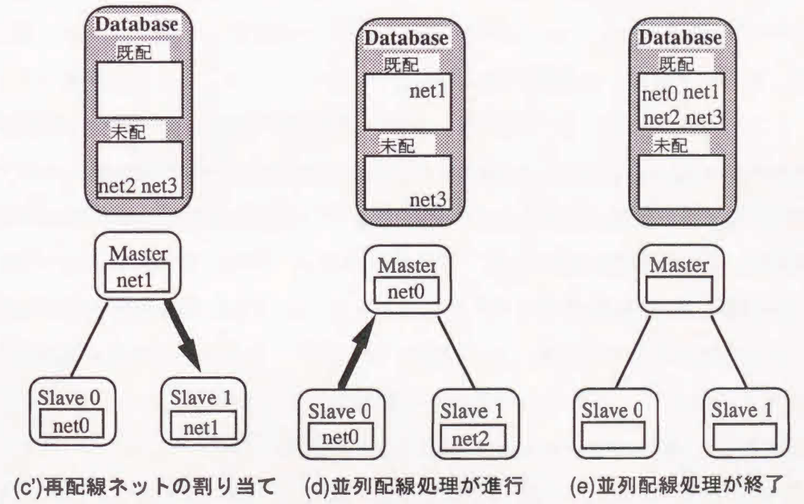


図3-7 アルゴリズムの実行例

3.4.3　マスタ／スレーブの役割

マスタ及びスレーブの役割についてまとめる．**マスタにおける処理**は次の３項目である．

（1）配線結果の評価と登録

各スレーブでの配線結果が正しいかどうか検証し評価する．評価の結果，他の処理済みの配線経路と交差・接触していなければ正しい配線結果としてデータベースに登録される．

（2）複数スレーブの競合の裁定

各スレーブは割り当てられたネットの配線処理終了後，マスタへ配線結果を送り評価を依頼する．しかし他のスレーブでも同様の動作を行うために競合が発生するので，この競合を先着順に裁定する．

（3）データベースの管理

データベース内部には配線問題として与えられたネットデータ，配線結果のネットリスト及び配線領域の情報があり，マスタはこれらのデータを管理する．これらのデータのうち，配線領域の情報はスレーブから参照可能であるが，他のデータはマスタのみ参照更新できる．ネットデータはマスタがスレーブに配線処理を割り当てるときに与えられる．

一方，**スレーブにおける処理**は次の２項目である

（1）マスタより配線すべき１本のネットデータを受け取り，データベースを参照して配線処理を実行する．マスタから与えられるネットデータは未配線ネット（再配線の場合には前回処理したネット）のネットリストが与えられる．基本的にスレーブでは配線／再配線処理の区別せず，同じ配線処理として取り扱われる．

（2）配線終了後，配線結果をマスタに送る．マスタに送る結果は配線処理したネットのネットリストとその他の幾つかの情報のみであり，そのデータ量は少ない．

3.5　分散メモリ型並列計算機による評価

これまではプロセッサ競合方式の一般的特徴について述べてきた．これに対して３.５節及び３.６節では実際に２種類の並列計算機上にプロセッサ競合方式の並列配線プログラムを実装して評価した結果について述べる．本節では分散メモリ型並列計算機への実装とその評価結果について述べる．

3.5.1　分散メモリ型並列計算機

本節で仮定する分散メモリ型並列計算機（以下分散メモリ型）は，２.４.２節で述べたように，各プロセッサはローカルメモリだけを持ち（分散共有メモリ等も認めない），プロセッサ間の情報伝達は相互結合網を通してメッセージ交換により実現されるものとする．

3.5.2　実装

分散メモリ型並列計算機におけるプロセッサ競合方式の並列配線プログラムの実装について各部の詳細とアルゴリズムを以下に述べる．

3.5.2.1　処理方式

分散メモリ型に対する実装のために，３.４節で述べたプロセッサ競合モデルを次のように変更した．

（1）データベースの参照方式

プロセッサ競合モデルの基本計算モデルではデータベースは全てのスレーブから参照可能で，参照は任意の時点で行うことができると仮定している．しかし分散メモリ型では共有メモリを持たないため，スレーブはマスタに対してデータベースの参照要求を出すことにより，マスタが管理するデータベースを参照して配線処理を行う必要がある．

データベースの参照方式には，経路探索中に参照の必要性に応じてデータベースからスレーブにコピーされる**参照時コピー方式**（copy on reference）とネットデータを割り当てるときにコピーする**割り当て時コピー方式**（copy on assignment）がある．参照時コピー方式では，スレーブが経路探索を行う間，逐一データベースを参照する必要があるためマスタへの参照要求の頻度が非常に高くなり，通信コストの高い分散メモリ型ではボトルネックを生じる危険性がある．このため幾つかの参

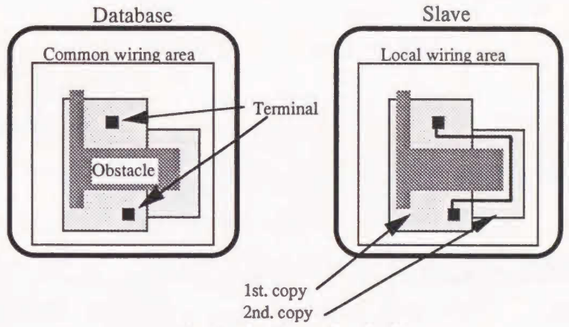
照要求をまとめてマスタに出すか（**要求蓄積方式**），グリッド単位での参照でなく，配線領域を小さな部分領域に分割し，必要に応じて部分領域単位で参照する方法（**部分要求方式**）が考えられる．要求蓄積方式では幾つかの参照要求が溜るまで参照要求が発行されないために，その間の時間待ちが蓄積され，スレーブの利用率を低下させる．一方部分要求方式では，参照に用いなかった残りのグリッド情報はスレーブ側でキャッシングすることにより，データを有効に利用することができるが，参照要求の時間的順序はマスタから送られる参照情報の時間的順序と必ずしも一致しないため，ある時点におけるデータベースの内容とスレーブの参照結果とは一致する保証はない．このため，プロセッサ利用率の点においては部分要求方式が有利であるが，データの参照順序の正確さにおいては蓄積要求方式が有利である．

競合方式ではプロセッサの競合により矛盾が発生するので，データの参照順序は重要ではなく，蓄積要求方式の有効性は低い．一方，発生する矛盾を再配線処理により解消する方法を用いているので，ある程度の矛盾発生は許容できる仕組みを持つため，矛盾発生割合が低ければ部分参照方式は充分有効である．割り当て時コピー方式はこの考えを進めたもので，スレーブにネットデータを割り当てるときにデータベースの配線領域情報も同時にコピーして渡す方法をとる．この場合，予めコピーして渡すので配線処理中はコピーされた配線領域範囲以外の参照を行わない限りスレーブは通信を行う必要がなく，全体の通信回数を大幅に削減できる．従って分散メモリ型への実装には割り当て時コピー方式を用いた．但しこの方式では矛盾の発生による再配線回数が増加し，並列処理効率が低下することが危惧されるが，高橋らの報告[39]では再配線の割合は総配線処理回数に対して高々１０%程度であることが報告されており，過度の性能低下を引き起こすことは無いと判断した．

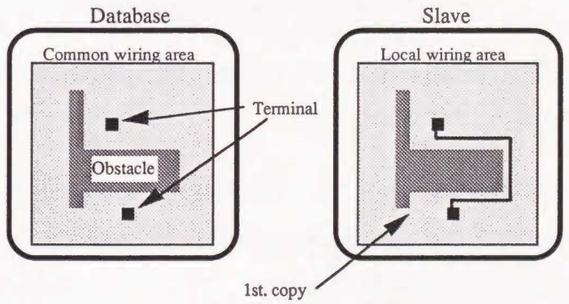
（2）データベースのコピー範囲

（1）割り当て時コピー方式について述べたが，ここではコピーする配線領域の範囲を検討する．コピー方式には部分コピー方式と全体コピー方式の２つがある．**部分コピー方式**は，**図3-8**(a)に示す例のように，配線処理の開始時にマスタがスレーブに割り当てるネットデータから経路探索範囲（図中の塗りつぶされた範囲）を推定してその領域だけをコピーする方式である．もし経路探索がコピーされた領域外に到達した場合，部分要求方式と同様にスレーブはマスタに対して参照要求を出して経路探索を継続する．要求に対してマスタでは新しい参照領域を推定してコピー

する．一方**全体コピー方式**は**図3-8** (b)のように配線領域全体を配線処理開始時にコピーする方式である．この方式は前者と比較して簡単であり配線処理中は通信する必要がないために通信路の負荷が小さく，通信時間も問題サイズに依存するが一定でありコピー回数が１回で済む利点がある．但し配線領域が拡大すると一度にコピーするデータ量が増加するが，今回の実装では小規模の配線問題を対象にしているので全体コピー方式を用いることにした．



（ａ）部分コピー方式



（ｂ）全体コピー方式

図3-8　部分コピー方式と全体コピー方式

3.5.2.2　アルゴリズム

分散メモリ型並列計算機における並列配線アルゴリズムは**図3-9**に示すとおり次の手順で行われる．

（1）マスタは割り当てる一本のネットデータとその時点のデータベース中の配線領域情報のコピーをスレーブに送る．

（2）スレーブはマスタから上記のネットデータとデータベースのコピーを受け取

り，このコピーを参照しながら配線処理を実行する．コピーはスレーブにローカルなものであるため，参照・更新を自由に行う．

（3）スレーブは配線処理結果をマスタに送り，マスタは先着順に受け取る．

（4）マスタはスレーブから受け取った配線処理結果を検証し，データベースを更新する．そして新たなネットデータを割り当てる．もし処理すべきネットデータが存在しない場合，スレーブに対して処理終了のシグナルを送る．シグナルを受け取ったスレーブは全てのスレーブの処理が終了するまでスリープ状態になる．

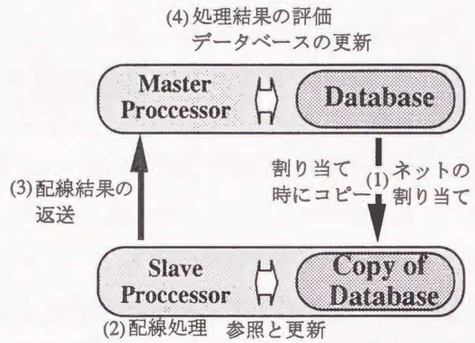


図３-９　分散メモリ型並列計算機における処理の流れ

３.５.３　実装

実装は徳島大学工学部知能情報工学科高橋研究室で1987年に開発されたCoral-68Kを用いた．Coral-68Kは分散メモリ型の実用規模の２進木結合並列計算機であり，MPUにMC68000（10MHz）を採用し，63台のプロセッサエレメント（以下PE）を図３-１０のように２進木状に結合したものである．プロセッサ間通信はDMA通信を用いており2MB／秒の転送速度を持つ．各PEは512KBのローカルメモリを持ち，演算能力は全体で40MIPS，0.3MFLOPSである[40]．以下では，Coral-68Kにおけるプロセッサ競合方式の並列配線プログラムの実装における工夫点について述べる

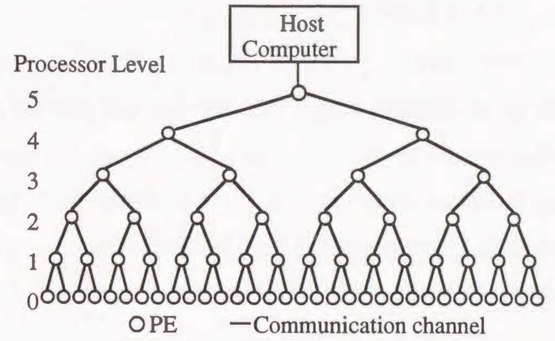


図３-１０　Coral-68Kの構成

（１）プロセッサ間通信

Coral-68Kでは２進木の節にPEが存在し，プロセッサ間通信は最も単純な通信方式である蓄積交換方式（Store and forward method.）により行われる．このため，マスタとスレーブ間で通信を行う場合には通信経路途中にあるPEが中継処理を行う必要がある．この通信の中継処理を割り込みを用いて実現した．これにより通信経路途中にあるPEは配線処理中に下位PEからの割り込みにより中継処理を実行することになる．図３-１１に中継処理の様子を示す．また通信アルゴリズムの骨格を以下に示す．このアルゴリズムは通信部分のみについて記述している．なお下記のアルゴリズムにおける右側の数字は図３-１１における各番号の処理に対応している．

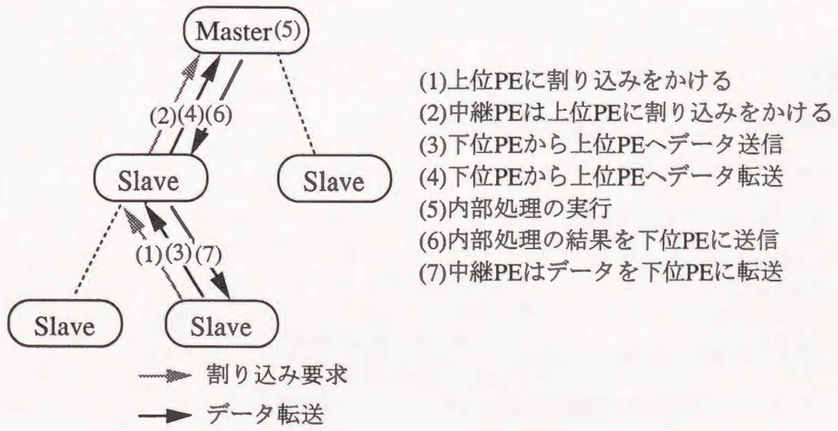


図３-１１　プロセッサ間通信の中継の様子

```
＊＝＝＝　スレーブにおける通信アルゴリズム　＝＝＝
SendReceiveSlave( sbuf , rbuf ) {
        DisableInterrupt(LEFT_DIR ¦ RIGHT_DIR);
        InterruptToUpperProcessor();                    … (1)
        SendToUpperProcessort(sbuf);                    … (3)
        ReceiveFromUpperProcessor(rbuf);                … (7)
        EnableInterrupt(LEFT_DIR ¦ RIGHT_DIR);
}
＊＝＝＝　スレーブにおけるデータ中継アルゴリズム　＝＝＝
＊　下位プロセッサからの割り込みにより呼び出される．
＊＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝
RelayTransmittion(dir, tmpbuf) {
        DisableInterrupt(dir) ;
        InterruptToUpperProcessor();                    … (2)
        ReceiveFromLowerProcessor(dir, tmpbuf) ;        … (3)
        SendToUpperProcessort(tmpbuf);                  … (4)
        ReceiveFromUpperProcessor(tmpbuf);              … (6)
        SendToLowerProcessor(dir, tmpbuf) ;             … (7)
        EnableInterrupt(dir);
}
＊＝＝＝＝　マスタにおける通信アルゴリズム　＝＝＝＝
＊　割り込みによってこの関数は呼び出される．
＊＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝
ReceiveSendMaster(dir, sbuf, rbuf) {
        DisableInterrupt(dir);
        ReceiveFromLowerProcessor(dir, rbuf) ;          … (4)
        Do_other_process(dir, rbuf) ;                   … (5)
        SendToLowerProcessor(dir, sbuf) ;               … (6)
        EnableInterrupt(dir);
}
```


（2）経路探索法

経路探索法として線分探索法[4]を用いた（2.2.2(2)を参照）．線分探索法は基本的にグリッド単位で探索座標を管理しているが迷路探索法のようにグリッドを実在させる必要は無い．しかし，Coral-68Kにおける実装では迷路法のようなグリッドを使用してデータベース中の配線領域を表現し，スレーブにおける経路探索ではコピーされたデータベースの配線領域の情報をそのまま経路探索に使用することにより，データベースの情報から配線領域の情報を変換するための処理を省くことができた．

（3）アルゴリズム

Coral-68Kに実装した競合方式のマスタとスレーブのアルゴリズムのうち並列配線処理に関する部分のみを以下に示す．マスタにおけるMainLoopOfMaster関数はスレーブからの割り込みシグナルによって呼び出され，スレーブからの配線結果を受け取り，これを評価し，新しいネットデータを割り当てた後，データベースを更新する一連の処理を実行する．ここで評価の後に直ちにデータベースを更新せず次に処理するネットデータを割り当てている理由は，なるべく短時間で次の処理データを割り当てることで，スレーブの待ち時間を減らすためである．

```
MainLoopOfMaster(dir) {
        DisableInterrupt(dir) ;
        ReceiveFromLowerProcessor(dir, rbuf) ;
        switch(rbuf.Instruction) {
          case REQUEST_NEW_DATA:
               sbuf.Instruction =  DO_NEW_ROUTING ;
               sbuf.NetData     = Get_newdata_from_database() ;
               break ;
          case REQUEST_EVALUATION:
               if( EvaluateReceivedResult(rbuf) == CONFLICTED ) {
                    sbuf.Instruction =  DO_RETRY_ROUTING ;
                    sbuf.NetData     = rbuf.NetData ;
               }  else {
                    if( All_Netdata_is_done != YES ) {
                         sbuf.Instruction =  DO_NEW_ROUTING ;
                         sbuf.NetData     = Get_newdata_from_database() ;
                    } else {
                         sbuf.Instruction = COMPUTING_DONE ;
```

```
                }
                UpdateDatabase(rbuf) ;
            }
            break ;
        }
        SendToLowerProcessor(dir, sbuf) ;
        EnableInterrupt(dir);
}
```

スレーブではMainLoopOfSlave関数が初期化後に呼ばれる．これはスレーブにおける配線処理の基本ループを構成している．この基本ループではスレーブはマスタにネットデータを要求し（初期化直後の場合）ネットデータと共にデータベース中の配線領域のコピーを受け取った後，経路探索処理を実行する．次に得られた探索経路をマスタに送り，次に処理するネットデータを受け取る．なおアルゴリズムからも判るように，スレーブでは配線処理と再配線処理の区別はされず，同じ処理として取り扱われる．

```
MainLoopOfSlave() {
        sbuf.Instruction = REQUEST_NEW_DATA ;
        rbuf  = EMPTY ;
        while( SendReceiveSlave(sbuf, rbuf) != COMPUTING_DONE ) {
                InitilaizeLocalData() ;
                SearchPath() ;
                WriteResultToBuffer(sbuf);
                sbuf.Instruction = REQUEST_EVALUATION ;
        }
}
```

3.5.4　評価

この節ではCoral-68Kに実装されたプロセッサ競合方式の実験評価について述べる．

3.5.4.1　配線問題

評価に使用した配線問題は次の通りである．

・配線領域を256×256グリッドの２層とする

・第１層を横方向，第２層を縦方向の配線線規則

・ピン間０本

・２端子配線問題

ネットデータは配線領域上のランダムな位置に選んだ２点の始点・終点を端子とする２端子ネットデータを生成し，端子間のマンハッタン距離の平均値がそれぞれ１０，２０となるようなネットをそれぞれ1,０００，２,０００，３,０００本づつ作成した．**図3-１２**に3,０００本の場合のランダムネットのマンハッタン距離の分布を示す．

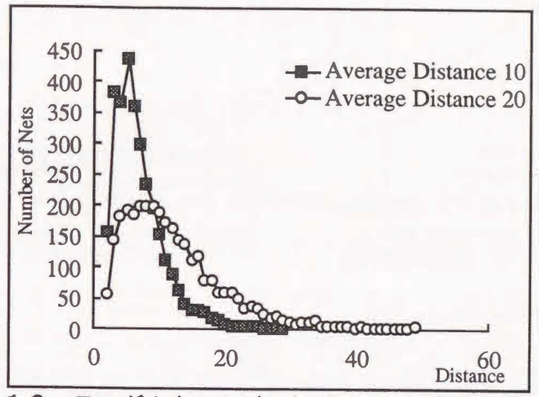


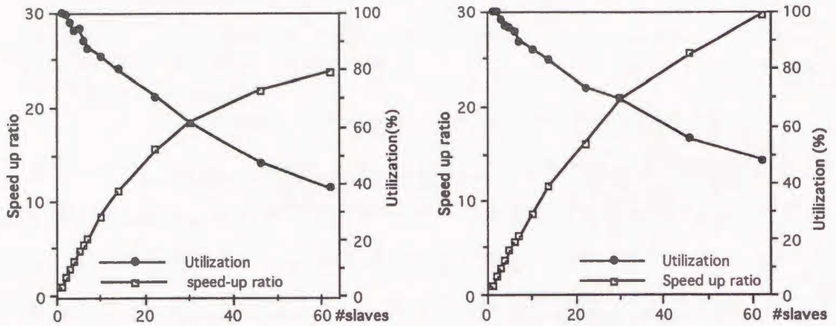
図3-１２　ランダムネットデータのマンハッタン距離の分布

3.5.4.2　実験結果

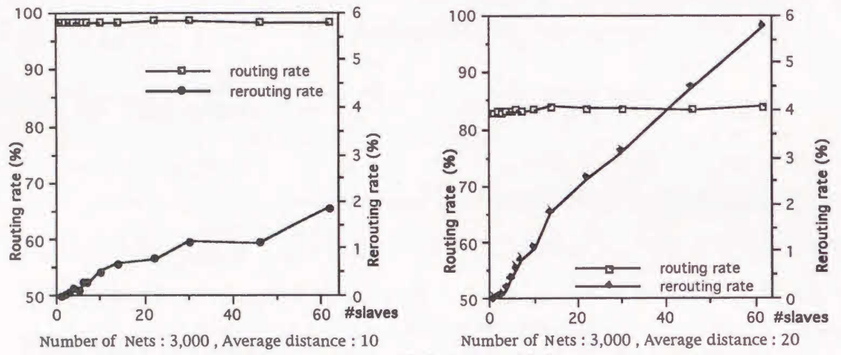
実験は前述のネットデータを使用してスレーブ数を１から最大６２まで変化させて行った．**図3-１３**にネット数３,０００の場合の実験結果を示す．(a)は速度向上比とスレーブプロセッサの利用率，(b)は配線率と再配線率，(c)は５００ネット毎に集計した配線率と再配線率のグラフである．この時の平均配線率は平均距離１０のネットで９８.３%，平均距離２０のネットで８３.５%が得られた．

グラフ(a)に示されるように，平均距離２０のネットデータの場合，スレーブ６２台で約３０倍の速度向上比が得られた．しかしスレーブ数の増加に従って利用率が低下していることが確認され，６２台では約５０%まで低下している．これは台数増加によるプロセッサ間通信の増加が原因である．つまりプロセッサ間通信の増加によりスレーブの通信待ち時間が増加したことで，プロセッサの利用率が低下したためである．次にネットの平均距離の違いに注目すると，平均距離の長いネットの方が速度向上比が高いことが確認された．これは平均距離が長いネットの方が平均配線処理時間が長いため，その分マスタにおける通信の集中が避けられることが理

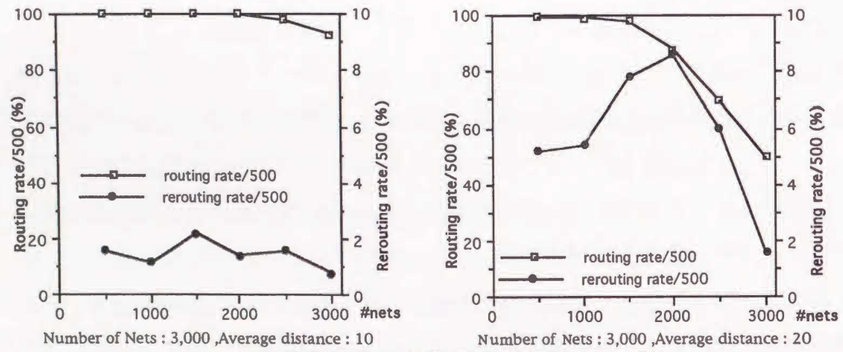
由である．しかしグラフ(b)で見られる様に平均距離が長いと配線経路の競合が発生し易く，再配線率が高くなっている．



(a) 速度向上比とスレーブプロセッサの利用率



(b)配線率と再配線率



(C)５００ネット毎の配線率と再配線率

図3-13　分散メモリ型並列計算機における実験結果

グラフ(c)から配線処理の進行に伴う配線率の変化を見ると，平均距離１０のネットでは配線率は殆ど変化しないが，平均距離２０のネットでは配線処理の後半で配線率が著しく低下している．これは配線処理が進行するに従って配線可能な領域が減少することが原因である．また再配線の平均割合は図(b)より最大でも６％未満であることから矛盾の発生割合は比較的小さいことがわかる．これは高橋らの単層配線問題の研究結果[39]よりも更に小さくなっている．

3.5.4.3　評価

Coral-68Kを用いた実験結果から，以下のことが明らかになった．

（1）プロセッサ競合モデルの有効性

マスタ・スレーブモデルを用いた単純な計算モデルを使用することにより，並列配線処理アルゴリズムは簡単になり，アーキテクチャ固有の機能を使用しない実装を実現した．この方式ではデータベースの参照方法として割り当て時コピー方式と全体コピー方式を用いてプロセッサ間通信の頻度を抑えることにより，ある程度のボトルネックの発生が見られるものの，高い速度向上比が得られた．これらのことから分散メモリ型における競合モデルの有効性は高いことがわかる．

（2）ネット割り当て法の有効性

ネット割り当て法による割り当て戦略は計算粒度が競合モデルに対して適度に粗く，マスタにおけるボトルネックの発生が少ないことから，ネット割り当て法は有効であるといえる．

（3）割り当て時コピー方式と全体コピー方式の有効性

データベース中の共通な配線領域の参照方式として，割り当て時コピー方式とその参照範囲として全体コピー方式を用いた．これら方法では，実装で述べたように，データベースとスレーブにコピーされた内容との間に矛盾を生じることがあるが，実験結果から，配線結果の矛盾の割合はマスタの処理回数（データベースの更新回数と同じ）に対して６％未満と低いため，本方式は競合モデルに対して有効であることがわかる．

（4）プロセッサ台数増加による並列性の低下

プロセッサ台数の増加に従ってプロセッサ利用率（並列性）が低下することが確認された．これはマスタへの負荷集中によるものであると推測される．実際，Coral-68Kにおける実装ではスレーブから配線結果が送られて来るとそれを評価して

再配線の必要性を検査し，次のネットデータを割り当てる処理サイクルを持つため，この間は他の配線結果を送ろうとしているスレーブは待ち状態になる．このためプロセッサ数が増加するとこの待ち時間が増加してプロセッサ利用率を低下させるものと考えられる．

（５）配線品質

処理の進行による配線率の変化では処理済みネットが増加するに従い配線率が低下している．これは実験結果の中で述べたように配線処理が進行するに連れて配線可能な領域が減少するためであるが，一度データベースに登録された配線経路は引き剥がされないことと，後続のネットの配線経路を考慮せずに配線処理を行っているためである．しかし配線終了後の配線率はほぼ一定であり，プロセッサ台数に依存しないことが確認された．

３.５.５　考察

競合モデルを分散メモリ型並列計算機Coral-68Kに実装し評価した結果，確認された競合モデルの問題点，すなわちボトルネックの発生，通信時間，及び配線品質の３点について考察する．

（１）ボトルネックの発生

スレーブ数が少ない場合にはボトルネックは殆ど発生しないが，スレーブ数が増加するに従ってボトルネックが発生している．この原因はマスタへの通信の集中であるが，これはマスタとスレーブの計算粒度のバランスの問題といえる．つまり，マスタの計算粒度がスレーブの計算粒度に対して相対的に十分小さいならば，ボトルネックの発生は割合は更に低下するものと考えられる．

実装されたアルゴリズムではスレーブから配線結果を受けとって新しいデータを割り当てるまでの一連の処理サイクルの間，他のスレーブは待ち状態となる．この原因はマスタはスレーブからデータを受け取った後，そのスレーブに対して再配線処理を指示するかどうかを判断するために配線結果を評価しており，この評価時間がスレーブに対する待ち時間の増加を招くことにある．この問題点は再配線処理待ちのキューを用意してマスタの処理サイクルの順序を次のように変更することで改善が期待できる．

現在の処理サイクル	改善案による処理サイクル
(1)配線結果の受取	(1)配線結果の受取
(2)配線経路の評価	(2)次のネットデータの割り当て
(3)次のネットデータの割り当て	(3)配線経路の評価
(4)データベース更新	(4)データベース更新

このようにすることで，**図３-１４**に示すように配線結果を受け取ってから次のネットデータを割り当てるサイクルから配線経路の評価を取り除くことができるため，スレーブの待ち時間が短縮される．

（２）通信時間

マスタとスレーブ間の通信データには，データベースのコピー，マスタが割り当てるネットデータ，及びスレーブからの配線結果があるが，このうちネットデータと配線結果は０.５ＫＢ程度と小さいが，データベースのコピーは256×256（縦×横）×4bit（層当りの情報量）×2（層数）＝６４ＫＢと大きく，データベースの通信時間が支配的であることがわかる[1]．

データベースのコピーには約３１ミリ秒を必要[2]とするが，マスタは理論的に毎秒約３２回の通信を行うことができ，毎秒３２台のスレーブの処理結果を受け付けることが可能である[3]．言い替えれば，スレーブからの通信頻度がマスタが処理可能なスレーブ数未満であれば，スレーブでの通信待ち時間はほとんど無いことを意味する．実験結果では平均距離が長いネットの方が高い速度向上比を示し，スレーブの計算粒度が粗く通信頻度が低いことを裏付けている．このため性能の向上には（１）で述べた方法の他に通信時間の短縮及び，スレーブでの計算粒度を粗くすることが考えられる．

次の３.６節で述べる共有メモリ型における実装では，データベースを共有メモリ上に配置することでデータベースの通信時間を削減することにより性能改善を試みている．

---

[1]プロセッサ間通信効率は，転送データサイズが大きくなるに連れて向上する傾向があるため通信時間はデータサイズの比に比例するとは一概には言えない．Coral-68Kでは数KB以上の転送サイズでは，通信効率は殆変化はない[38].

[2]Coral-68Kの通信速度が2MB/sであることから，一回の通信時間は６４／２０４８≒３１msとなる．

[3]実際にはマスタの他の処理時間などがあるため更に低下する．

（3）配線品質

実装したアルゴリズムでは，引き剥し再配線処理は考慮されていないため，並列配線処理が進行すると配線可能領域が不足して配線不可能な状態になる．この問題の解決方法として，概略配線を行ってから詳細配線を行う方法や[32][43]，引き剥し再配線処理[44][45]を用いる方法などが考えられるが，この問題は第４章で述べる引き剥し再配線処理方式により解決される．

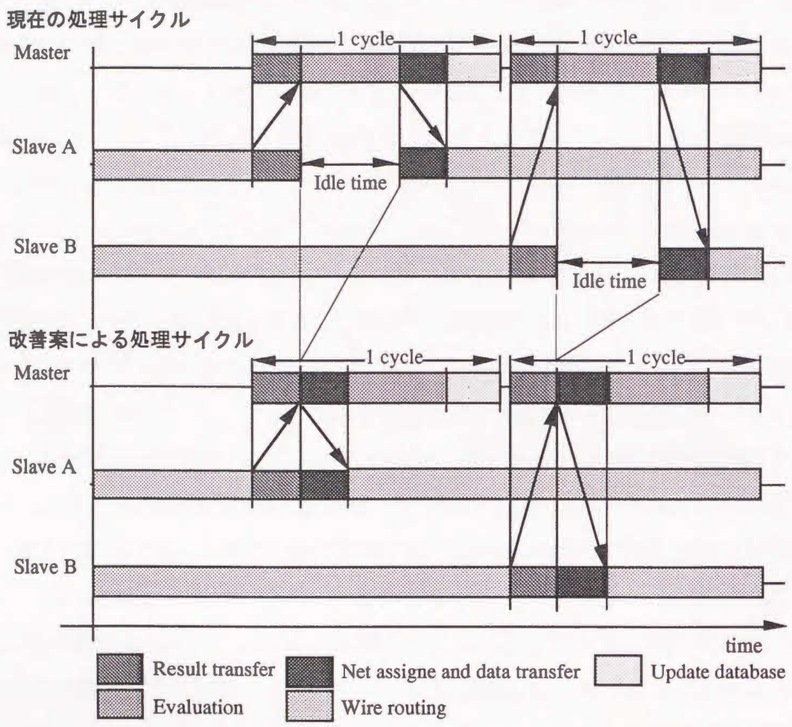


図３-１４　アルゴリズムの変更によるスレーブの待ち時間の改善例

３.６　共有メモリ型並列計算機による評価

３.５節では分散メモリ型並列計算機に対してプロセッサ競合方式の有効性を検証した．この分散メモリ型で用いた実装方法を共有メモリ型に適用することは容易であり，その結果を分散メモリ型の実験結果と比較することはこの並列配線方式の並列計算機方式への不偏性を証明するため極めて有意義である．また競合モデルを共有メモリ型に実装するに当たって，分散メモリ型への実装における問題点の改善と性能向上の新たな試みとして，競合モデルに対して幾つかの変更を加えることで，共有メモリ型並列計算機に対する競合方式の有効性を検証する．以下ではこれらの詳細について述べる．

３.６.１　計算機方式

本節で仮定する共有メモリ型並列計算機は，２.４.２節で述べたように，各プロセッサには共有のメモリを持ち，プロセッサ間通信は共有メモリ上へアクセスすることにより行われる．

３.６.２　共有メモリ型並列計算機におけるプロセッサ競合方式

共有メモリ型並列計算機（以下共有メモリ型）におけるプロセッサ競合方式について各部の詳細と一般的なアルゴリズムについて述べる．

３.６.２.１　実装方針

３.５節の評価結果はプロセッサ競合方式は分散メモリ型アーキテクチャ上で有効であることを示した．一方共有メモリ型アーキテクチャではプロセッサ間通信は共有メモリを介して行われるため，通信時間が分散メモリ型と比較して非常に短い．従って共有メモリ型に分散メモリ型と同様のプログラムの実装を行った場合，マスタのボトルネックが小さくなるため，分散メモリ型の場合よりもより性能が期待できる．なおCoral-68Kによる実験結果ではスレーブ台数が増加するにしたがってマスタに対するボトルネックが発生し，プロセッサ利用率が５０%程度まで低下している．これはデータベース参照のための通信時間が一つの原因であることが確認されている．そこで共有メモリ型における実装ではこの問題点を改善し性能向上を図るため，データベースの参照方式を変更して共有メモリ適した実装に関して幾つかの方法を試みることにした．

3.6.2.2　処理方式

共有メモリ型ではプロセッサ競合モデルに対して次のように変更を行った．

（1）データベースの参照方式

分散メモリ型における実装では，スレーブにおけるデータベースの参照は割り当て時コピー方式を用いることにより通信頻度を抑えてプロセッサ利用率の低下を抑える方法を用いていた．これに対して共有メモリ型では，前述の方針に従って通信時間の大半を占めているデータベースのコピー時間の短縮を図るため，データベースを共有メモリ上に配置する．これによりスレーブは非常に短い時間でデータベースの参照ができるため，データベース参照のための通信時間が大幅に削減され，全体性能が向上する筈である．

（2）データベースの参照範囲

データベースを共有メモリ上に配置したことにより，スレーブからの参照に対するプロセッサ間通信による制約は殆ど解消されるため，スレーブによるデータベース中の配線領域の参照範囲は配線領域全体とした．

（3）データベースの参照と更新

データベースを共有メモリへ配置すればスレーブからデータベースを直接参照することができる．このため，分散メモリ型に用いた割り当て時コピー方式ではなく，参照時コピー方式を用いることができる．また，部分コピー方式を用いることでデータベースとスレーブの処理結果との矛盾を軽減することができる．

共有メモリ型では通信コスト（通信時間）が低いので，計算粒度を細かくしてプロセッサ競合による矛盾を無くすことが可能であるが，先に決めた競合モデルにおける実装の方針を踏襲して分散メモリ型の場合と同じ計算粒度を用いることにする．従ってデータベースの参照は分散メモリ型の場合と同様に自由に行うものとするが，データベースの更新はデータベース内部での矛盾を避けるためにマスタのみが行うものとする．

（4）マスタ機能の変更

一般に，共有メモリ型並列計算機はそのアーキテクチャ上の制限から多数のプロセッサ数を持つことは難しいとされるため，数台から数十台規模の構成が多い．一方分散メモリ型並列計算機では共有メモリ型と比較してプロセッサが豊富にあるため，競合モデル上のマスタに対してプロセッサを１台割り当てていた．しかし配線

結果の評価要求がスレーブから発生しない限りマスタプロセッサは処理する仕事がない．そこで利用されていないマスタプロセッサの計算資源を有効に活用するため，マスタ機能の変更を以下のように検討する．

マスタの機能にはスレーブの配線結果の評価とデータベースの更新があり，これらはスレーブからの配線結果を受けてから実行される．しかし，配線処理を行ったスレーブ自身が配線結果の評価とデータベースの更新を行うことにすれば，マスタの処理をスレーブに代行させることができ，競合モデルにおけるマスタ専用にプロセッサを割り当てる必要がない．更にネットデータの割り当てと配線結果の回収のためのプロセッサ間通信も省くことができる．しかしデータベースの更新は矛盾を避けるため同時に一台のプロセッサのみに限定する必要がある．そこで，全てのプロセッサに対してマスタの機能を処理するマスタモードと，スレーブの機能を処理するスレーブモードを設ける．スレーブモードにおける機能は競合モデルにおけるスレーブと同じであるが，スレーブモードからマスタモードに移行する場合に排他制御を行うことでマスタモードにあるプロセッサが常に１台になるようにする．このマスタモードとスレーブモードの関係を図３-１５に示す．

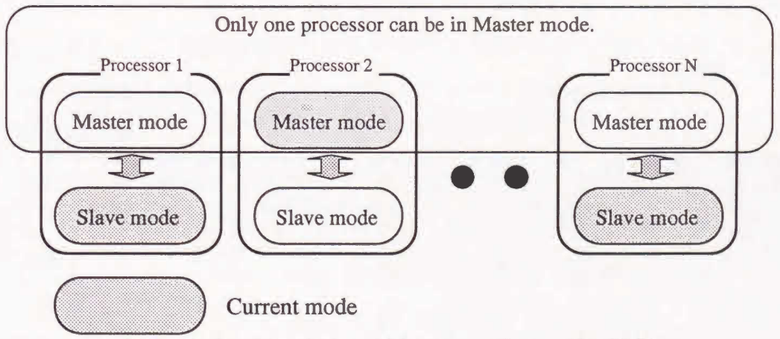


図３-１５　マスタモードとスレーブモードの関係

3.6.2.3　アルゴリズム

共有メモリ型における並列配線アルゴリズムは図３-１６に示す通り全てのプロセッサが次のような処理を行う（但し処理するネットデータは既に割り当て済みとする）．

（1）スレーブモードでは共有メモリ上に配置されたデータベースを参照しながら

配線処理をする．

（２）配線処理終了後，マスタモードの権限を獲得する．

（３）マスタモードに移行し，配線結果を評価する．評価の結果，再配線が必要な場合は評価したネットデータを次に処理するネットデータとする．配線結果が正当な場合はデータベースを更新する．

（４）マスタモードの権限を放棄しスレーブモードに移行し（１）へ戻る．

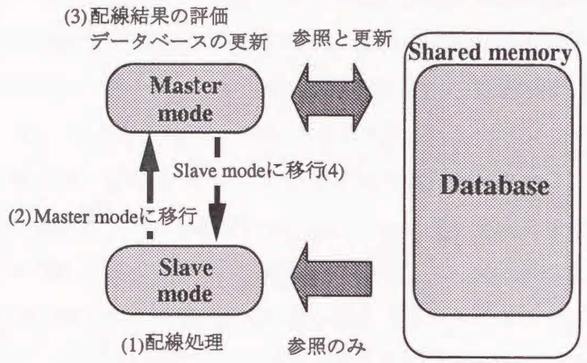


図３-１６　共有メモリ型における処理の流れ

3.6.3　実装

このアルゴリズムの実装には日本IBM（株）東京基礎研究所で開発された共有メモリ型並列計算機TOP-1[46]を使用させて頂いた．この計算機は図３-１７に示すように，MPUとしてi80386を用いた１０台のPEが共有バスによって接続されており，６４ビットの共有バス幅と二重化インタリーブ方式によって８５ＭＢ／秒の実効転送能力を持つ．各PEは１２８ＫＢのスヌープキャッシュを持ち主記憶は３２ＭＢ（最大１２８ＭＢ）である．全体性能は平均約３０MIPS，最大３０MFLOPSである．１０台のPEのうち，システムの監視に１台，OS（UNIX）に１台割り当てられており，またUNIX上の他のプロセスのために１台割り当てられているため，実際にアプリケーションに使用できるPEは７～８台程度である．

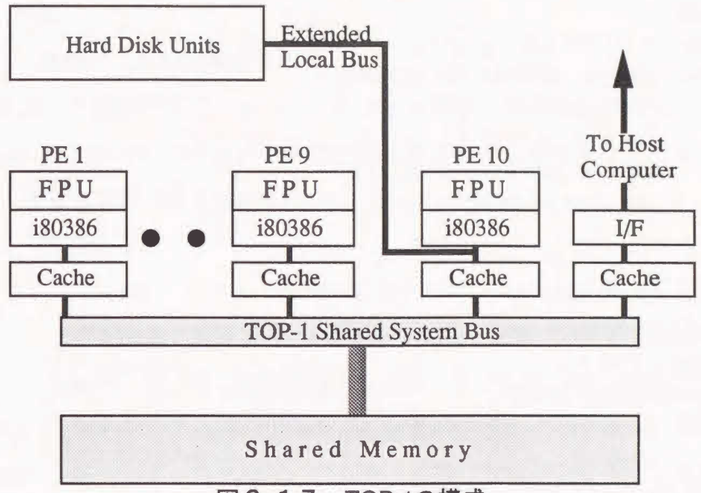


図３-１７　TOP-1の構成

以下ではTOP-1における競合方式の実装に関する工夫点について述べる．

（１）データベースの参照方式と参照範囲

方針で述べたように，データベースを共有メモリ上に配置することによりデータベースの参照コストが大幅に低下し，データベースをまとめてコピーする必要がないので，データベースの参照を部分領域単位で行う部分コピー方式を用いた．また，データベースの参照範囲は分散メモリ型の場合と同じ配線領域全体とした．

（２）経路探索法

この実装での経路探索法にはCoral-68Kの場合と同じ線分探索法を用いた．Coral-68Kでは，コピーしたデータベースをそのまま経路探索に使用できるようなデータ構造を採用していたが，今回の実装では経路探索における障害物の有無のみをデータベースで参照し，スレーブでの更新を伴う操作はローカルに確保した配線領域上で行う方法に変更した．

（２）アルゴリズム

TOP-1における実装に関して，アルゴリズムを以下に述べる．

MainLoopではマスタモードとスレーブモード間の移行を制御し，Do_master_modeでは競合モデルにおけるマスタの機能，Do_slave_modeでは競合モデルにおけるスレーブの機能をそれぞれ実行する．GetRightOfMasterはマスタの権利を得る関数であるが，これは，セマフォを用いた排他制御により実現され，権利が得られるまで待つ．

```
MainLoop(pid) {
        operation = REQUEST_DATA ;
        while(status != COMPUTING_DONE) {
                GetRightOfMaster(pid) ;
                Do_mastaer_mode(key, pid, status, databuf, rbuf ) ;
                ReleaseRightOfMaster(pid, key) ;
                Do_slave_mode(key, pid, status, databuf, rbuf );
        }
}

Do_master_mode(key, pid, status, databuf, rbuf) {
        switch( status ) {
        case REQUEST_DATA:
                databuf = Get_newdata_from_database();
                break ;
        case REQUEST_EVALUATION:
                if( EvaluateWiredResult( rbuf ) == CONFLICTED ) {
                        status = DO_RETRY_ROUTING ;
                        databuf = Get_samedata_from_database();
                        break ;
                } else {
                        if( All_Netdata_is_done != YES ) {
                                status = DO_NEW_ROUTING ;
                                databuf = Get_newdata_from_database();
                        } else {
                                status = COMPUTING_DONE ;
                        }
                UpDataDatabase( rbuf ) ;
                break ;
        }
}

Do_slave_mode( key, pid, status, databuf, rbuf ) {
        if( status != COMPUTING_DONE ) {
                InitializeLocalData() ;
                SearchPath() ;
                WriteResultToBuffer( rbuf ) ;
                status = REQUEST_EVALUATION ;
        }
}
```


3.6.4 評価

3.6.4.1 実験データと実験条件

実験に使用した配線問題とネットデータは分散メモリ型で使用したものと同じものを用いる．なおTOP-1ではユーザが使用できるプロセッサ台数は７，８台程度であるため，評価は７台を最大とする．

3.6.4.2 実験結果

図３-１８にネット数３,０００の場合の結果を示す．(a)は速度向上比とプロセッサの利用率，(b)は配線率と再配線率，(c)は５００ネット毎に集計した配線率と再配線率のグラフである．この時，平均距離１０のネットで９８.３%，平均距離２０のネットで８４.９%の平均配線率が得られた．評価の結果から共有メモリ型の評価の場合とほぼ同じ結果が得られており，プロセッサ競合方式は共有メモリ型並列計算機に対しても有効であることが確認された．またマスタに対するボトルネックは殆ど発生しておらず，線形な速度向上比が得られた．再配線率が最大で１%未満であることから，共有メモリを用いたデータベースの参照時コピー方式の有効性が高いことが確認された．

3.6.4.3 評価

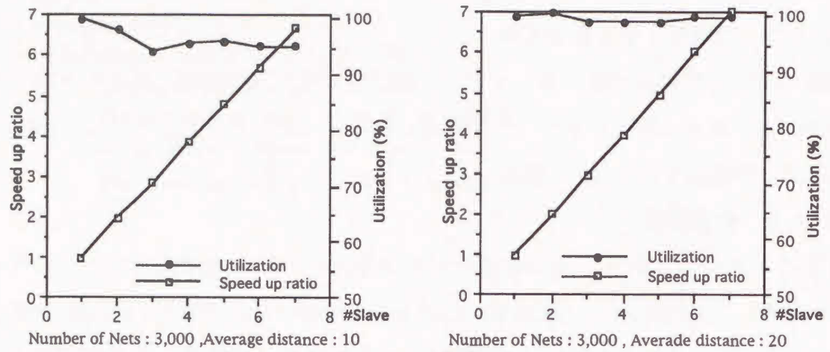
TOP-1を用いた実験結果から以下のことが明らかになった．

（１）データベースの参照時コピー方式の有効性

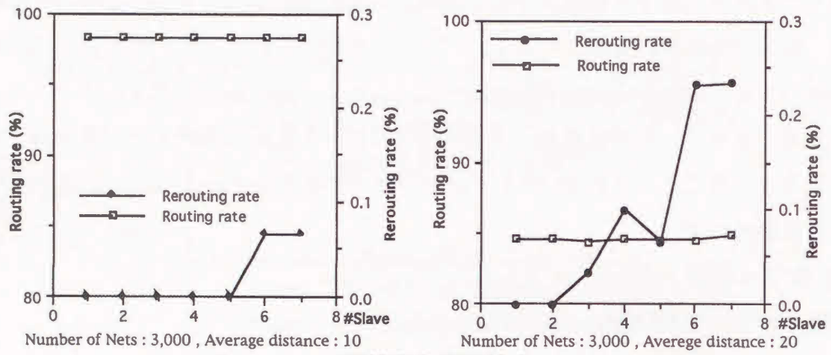
共有メモリ上に配置したデータベースに対する参照のオーバーヘッドはプロセッサ利用率から判断して極めて小さく，参照時コピー方式による共有バスへの負担は小さいといえる．また同じスレーブ数における矛盾発生割合は分散メモリ型の６%に対して１%未満と小さく，データベースからコピーされて実際に参照されるまでの時間が短い参照時コピー方式の有効性が確認された．また分散メモリ型と比較してデータベースの参照時間が大幅に短縮されており，全体性能が向上していることからも有効性が確認できる．

（２）マスタプロセッサ削除の有効性

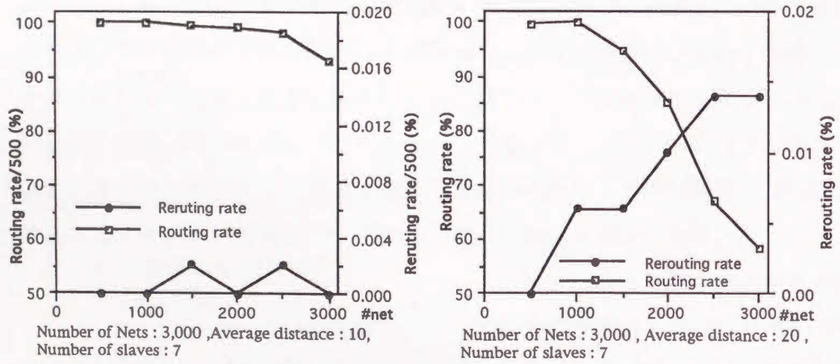
マスタプロセッサを削除したことにより，プロセッサ利用率の向上が得られた．これは，プロセッサの有効利用とデータ転送の削除による他のプロセッサの待ち時間の減少によるものである．このことからマスタプロセッサ削除の有効性が確認された．



(a)速度向上比とスレーブプロセッサの利用率



(b)配線率と再配線率



(c)５００ネット毎の配線率と再配線率

図3-１８　共有メモリ型並列計算機における実験結果

（3）共有メモリの効果

共有メモリを用いた競合方式の実装は高い並列性を発揮し，競合方式の有効性が確認された．また競合による矛盾解消のための再配線処理割合が小さいことからも共有メモリの効果が確認された．

（4）配線品質

共有メモリ型における実験結果では分散メモリ型の場合と同様の傾向であり，処理済みネットの増加に従い配線率が低下していることが確認された．全体の配線率も分散メモリ型の場合と同様であり，プロセッサ数によって殆ど変化しないことが確認された．つまり配線品質はプロセッサ数と関連性が少ないといえる．

3.6.5　考察

共有メモリ型並列計算機におけるプロセッサ競合方式の実装では，性能改善のための幾つかの変更により高い並列性が得られることを確認した．分散メモリ型と同じような実装は共有メモリを用いた通信コストの低下によりその分の全体性能の向上が得られるが，今回の実装では分散メモリ型において確認された問題点の幾つかを解消し性能改善を試みるために，スレーブのデータベース参照方式の変更とマスタ機能の分割を行ったものである．実験結果からは高い並列性が得られることがわかり，性能改善の試みは成功したと言える．

3.7　両計算機方式における評価結果の比較

3.6節及び3.7節で述べた分散メモリ型並列計算機と共有メモリ型並列計算機におけるそれぞれの実験結果を比較し，それらの実装方式に関して考察する．

（1）比較の妥当性

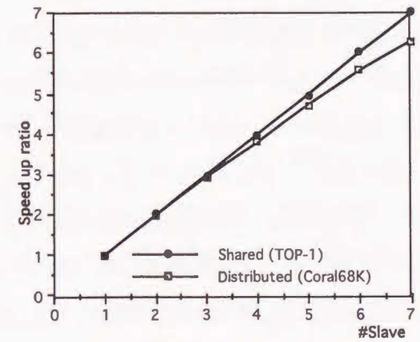
実装に用いた２台の並列計算機の通信性能について考察する．表3-1には，それぞれのPEの性能[1]と通信性能を比較している．通信速度に関してはCoral-68Kの２MB／秒に対してTOP-1では８５MB／秒であるから単純比較して約１：４２の速度比となる．一方，それぞれの通信経路に接続されているプロセッサ数が異なることを考慮すると通信経路に接続されるプロセッサ性能当たりの通信速度（通信速度／MIPS）は，Coral-68Kでは表に示すように約２.５MB／秒／MIPS，TOP-1では約２.８MB／秒／MIPS[2]となり，似通っていることが分かる．このため，それぞれの計算機の絶対性能は異なるが相対的数値で比較する場合は妥当であるといえる．従って，分散メモリ型で用いた実装を共有メモリ型で実装するとほぼ同じ傾向の処理結果が得られることの裏付けとなる．

表３-１計算及び通信性能の比較

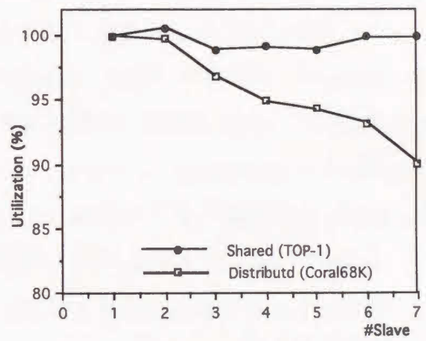
計算機名	PEの性能（MIPS）	通信経路当たりの性能			
		通信速度（B/sec）	経路当たりのPE数	PE当たりの通信性能（B/sec/台）	PE性能当たりの通信性能（B/sec/MIPS）
Coral-68K	0.8	2M	1	2M	2.5M
TOP-1	3	85M	10	8.5M	2.8M

（2）実験結果の比較

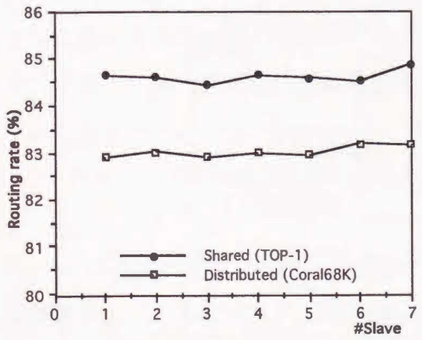
公平な比較のため同じプロセッサ数の場合である１～７の範囲のスレーブ台数で比較する．速度向上比では図3-１９に示すように共有メモリ型の方が若干優れていることが確認された．これは共有メモリの使用によるデータ通信の削除と配線結果の評価待ち時間の減少が理由である．再配線率では分散メモリ型の方が共有メモリ型に比べて３倍程度高くなっていが，これはデータベースの参照方式の違い（分散メモリ型では割り当て時コピー方式，共有メモリ型では参照時コピー方式）によるものである．しかし配線率では殆ど差が見られず，更にプロセッサ台数の変化による配線品質への影響も小さいことから参照方式の違いによる配線品質への影響は小さいといえる．



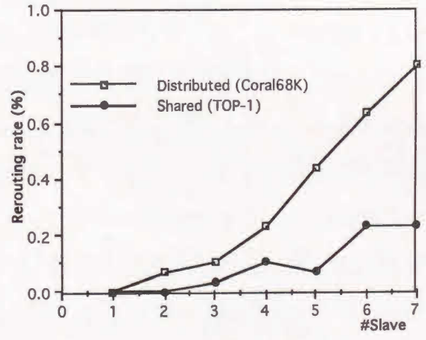
(a) プロセッサ利用率



(b)速度向上比



(c)配線率



(d)再配線率

図３-１９　両計算機方式の実験結果の比較

（3）両実装に用いたそれぞれの機能の効果

（a）割り当て時コピー方式と参照時コピー方式

データベースの参照方式には分散メモリ型と共有メモリ型ではそれぞれ，割り当て時コピー方式と参照時コピー方式を用いた．この方式の違いによる

[1]MPU性能の基準にはMIPS（Million Instructions Per Second.：秒当たりの命令実行数×10^6）値を用いる．

[2]Coral-68Kでは，point-to-point通信を全二重で行っており，一つの通信経路当たりの接続PE数は１であるから，通信速度をMIPS値で正規化すると，MC68000-10MHz≒0.8MIPSなので，2MB/秒（転送速度）÷0.8（MIPS値）≒2.5 MB/秒/MIPSとなる．一方，TOP-1では共有バス上に全てのプロセッサが接続されているため通信経路当たりのMIPS値の合計は30MIPSとなり，85MB/秒 ÷30 MIPS ≒2.8 MB/秒/MIPSとなる．

実験結果の相違は主に再配線率の差となって表れているが，全体の配線率への影響は殆ど見られない．このためデータベースのコピーから参照されるまでの時間差が配線品質に与える影響は小さいといえる．別の見方をすれば，プロセッサ数に対してネット数が十分に多い場合には，データベースと配線結果の矛盾頻度は深刻な問題になるほど高くならないといえる．

（b）部分コピー方式と全体コピー方式

各スレーブがデータベースを参照できる範囲に関して部分コピー方式と全体コピー方式があり，分散メモリ型では通信効率を重視して全体コピー方式を，共有メモリ型では共有メモリによる通信コストの低さを考慮して部分コピー方式をそれぞれ採用した．両方式の違いはコピーされたデータが実際に参照されるまでの時間と通信効率及び通信量である．前者は（a）で述べた理由と同様に再配線率に影響し，後者は並列処理の処理効率に影響を与える．**3.5.4.3 評価**における全体コピー方式の考察から，配線領域の全体が必ずしも参照されるわけでなく，大規模な配線問題における適切な領域サイズの部分コピー方式を用いれば，全体コピー方式よりも部分コピー方式が有効であると考える．

（4）比較のまとめ

以上の結果から，共有メモリ型並列計算機の方が分散メモリ型並列計算機よりも優れているが，共有メモリ型ではプロセッサ数の拡張性が分散型に比べて低く，多数のプロセッサを持った並列計算機を構築することが困難であることを考慮すると，分散メモリ型並列計算におけるプロセッサ競合方式の有効性は高いものといえる

（5）商用のルータとの比較

評価に使用したランダムデータを用いてプロセッサ競合方式と商用のルータ（SUN3上のルータCADNETIX）と比較した結果，**表3-2**に示すように，計算時間は本方式の方が極めて高速で約１８倍短縮されているが，配線率では商用の配線プログラムのほうが１０%以上良い結果を示し，本方式は配線品質の点で不十分であることが確認された．また，**表3-3**に実際のプリント基板のネットデータ[1]を用い

[1]実データはイビデン（株）より提供して頂いたものであり，４層配線基板でピン間２本の配線規則で作成されたものである．今回の評価では本来多端子ネットデータであったものを２端子ネットに変換したものを使用した．

た比較でも同様の結果が得られた．以下ではこの理由について考察する．

競合方式ではスレーブの処理結果の先着順評価により配線順序が一意に定まらず，割り当て順序と評価順序は一致しない．しかしこれまでの経路探索法では配線順序が確定的であることを前提としているため，配線順序が異なる場合に配線品質を低下させるが，実験で使用した線分探索法は配線順序に対する依存性の高いものであった．また競合方式では速度向上に重点を置いていたため，品質向上のための重要な手法の一つである引き剥し再配線処理を使用していなかったことも原因であった．そこで競合方式における引き剥しの効果を検証するため，簡単な引き剥がし再配線の実験を行った．**表3-4**にこの実験結果を示す．これはCoral-68Kを用いて７５０本のランダムネットデータを用いて実験した結果である．この実験ではマスタが評価した配線結果に交差・接触が確認された場合，交差・接触に関係したネット全てを引き剥して再配線するものであるが，実験の結果，殆ど改善が見られなかった．これは交差・接触するネットだけを引き剥して再配線しても，新たな迂回経路が得られない限り配線できず，同じ処理を繰り返すことが原因であった．このため，引き剥し再配線するには周囲の配線経路も考慮に入れた方法を考える必要があろう．

表3-2　Coral-68Kと商用ルータとの比較
（実験データは3,000，平均距離20を使用）

	Routing rate	Computing time
Coral-68K	83.5 %	170 sec (62slaves)
Sun 3/260 CADNETIX	95.5 %	3,042 sec

表3-3　実際のプリント基板のネットデータを用いた比較

Net		Coral 68K Our program	Sun 3/260 CADNETIX
2460 real net of ave. dist. 47	Computing time Routing rate	1,300 sec 85.6 %	25,200 sec 98 %

表3-4　引き剥しを用いた実験結果
（Coral-68K，ネット数750本，平均40，PE数62）

	配線率(%)	再配線率(%)	速度向上比(倍)
引き剥し無し	68.93	105.33	38.4
引き剥しあり	68.13	138.27	32.4

3.8 考察

共有メモリ型と分散メモリ型並列計算機に対してプロセッサ競合方式を実装し評価した結果，両方式共にその有効性が確認された．共有メモリ型と分散メモリ型はMIMD型並列計算機アーキテクチャを２分する方式であるため，両方式における有効性が確認されたことは，プロセッサ競合方式による並列配線処理方式が計算機アーキテクチャに対する依存性が基本的に低いことを示しているものと見ることができる．この点に関しては本研究の目的が達成されたことになる．

両計算機方式における共通な問題には配線品質が劣ることが挙げられる．提案した並列配線処理アルゴリズムでは並列処理による処理速度の向上に主眼を置いており，配線品質に対する考慮は十分されていなかった．その結果，配線処理の進行に伴う配線率の低下や商用ルータとの性能格差の原因となった．

表３-３に示すように競合の発生した配線経路を引き剥して再配線するだけでは配線品質の改善が見られなかったように，単純な引き剥し再配線方式では配線品質の改善は難しいと推測される．このため，高度な引き剥し再配線方式について研究する必要がある．この研究については第４章に述べる．

3.9 まとめ

本章では，ネット間の並列性に着目したネット割り当て法とマスタ・スレーブモデルを用いたプロセッサ競合方式おける並列配線処理方式を提案した．この方式は配線処理時間の大幅な短縮を可能にし，かつその処理方式は分散メモリ型と共有メモリ型並列計算機の両計算機方式において有効であり，MIMD型並列計算機のアーキテクチャに対する依存性の低さを示した．また両計算機方式における評価を比較した場合，共有メモリ型並列計算機の方が再配線率の低さ，マスタプロセッサにおける負荷の軽さの点から有利であることが確認されたが，分散メモリ型並列計算機の使用可能なプロセッサ数の多さ，特に超並列計算機による並列配線処理を考慮すると，分散メモリ型並列計算機に対するプロセッサ競合方式の有効性は高いということがいえる．

プロセッサ競合方式において改善すべき点は配線品質の向上である．配線品質の向上には引き剥し再配線が一般的な方法であるが，単純に引き剥し処理を用いたのでは品質改善が望めない．これは引き剥しを行っても再配線するための配線領域が存在しない限り配線不可能なためであった．次の第４章で述べる並列経路改善方式はプロセッサ競合方式における品質改善を目的としており，配線順序に低依存の経路探索法と引き剥し再配線処理の組み合せにより，配線品質の改善を実現している．

第４章

並列配線問題における並列引き剥し再配線処理の品質改善効果

４.１　まえがき

第３章ではプロセッサ競合方式による並列配線処理方式を提案した．これはMIMD型並列計算機アーキテクチャに対する依存性を抑えて各種並列計算機への実装を容易にし，かつ高い処理効率を得ることを目的するものであった．この目的は分散メモリ型及び共有メモリ型並列計算機に実装・評価により達成されたことを確認したが，商用のルータと比較して配線品質面で劣ることが明らかになり，これを改善することが課題であった．

この課題を解決するために，引き剥し再配線処理を用いた品質改善法に注目して研究を行った結果，本章で提案する並列経路改善方式を開発した[13-17]．本方式は第３章で述べたプロセッサ競合モデルを並列処理方式の基本とし，引き剥し再配線の反復処理による並列配線処理を行うものである．本方式は複数の配線経路を同時に引き剥して再配線することにより複数の経路改善を並列処理する点で，従来の並列配線処理方式と異なっており，またプロセッサ競合モデルとして実装されるため，計算機アーキテクチャに対する低依存性の少ないのが特徴である．

この方式では品質改善のため経路探索法に配線コストを用いた迷路法[38]を採用し，配線経路をコストで表現することで引き剥し再配線による経路改善を可能にした．この経路探索法は配線経路の交差や接触を許容し，交差や接触のない配線経路が存在しない場合でも準最適な配線経路が得られる点で第３章で用いた線分探索法と異なる．

続く４.２節では第３章で述べたプロセッサ競合方式による並列配線方式の問題点について考察した上で，本研究の目的と方針について述べる．４.３節，４.４節，４.５節では本章で提案する並列経路改善法式の基本概念と各部の概要，本方式のアルゴリズム，及び経路探索法について述べる．４.６節，及び４.７節では分散メモリ型並列計算機Coral-68Kに実装し評価した結果について述べ，４.８節でまとめる．

4.2 目的と方針

現在の並列配線方式の現状と本研究の目的について述べた後，第３章で述べたプロセッサ競合方式の改善点ついて考察し，本研究の方針について述べる．

4.2.1 並列配線方式の現状と研究目的

多くの並列配線方式では逐次配線方式のアルゴリズムをそのまま並列化し，計算処理を高速化することを目標にしている[30-32][34,35]．このため，ネット内の並列性とネット間の並列性の２段階の並列性を用いた配線処理方式が研究されている[30][32][34,35]．しかし，並列配線処理において計算速度より配線品質を優先させる場合，これまでの逐次方式よりも高い配線品質で，少なくとも逐次方式よりも短い時間で処理できることが望ましい．逐次方式と同じ配線手法に従う並列配線方式では配線順序が同じであるため，逐次方式と同じ配線結果を得るには充分であるが，それ以上の配線品質を求めるには並列配線方式に適合する配線方式を検討すべきである．このため配線方式には逐次方式では実用的時間で計算不可能とされて採用されなかった発見的手法や，全域探索手法を取り上げることが考えられる．これに要する膨大な計算時間を実用的な時間に短縮するために超並列処理を利用するのである．このような考えにより配線品質を向上させた並列配線処理方式として専用並列計算機MAPLE-RP（以下，RP）がある[8]．これは配線コストを制約条件に用いた経路探索アルゴリズム[38]をルータと呼ばれる専用並列処理装置を用いて並列実行するもので，その配線品質は一般的な逐次方式より優れていることが報告されている．

このような並列計算機の利用法はこれまでの処理時間の短縮を目的とするものとは異なるもので，解の品質改善を目的とする超並列処理なしには実現不可能な新しい利用法であり今後の発展が期待される分野であろう．

本章で述べる研究の目的は，逐次方式よりも高い配線品質を得ることができるMIMD型並列計算機のための並列配線方式を開発することであり，先に述べた超並列処理を基本とするものである．また実装には第３章のプロセッサ競合方式を用いることにより，計算機アーキテクチャに対する依存度が低い特徴を継承しつつ，配線品質の改善を図るものである．

4.2.2 プロセッサ競合方式の改善課題

プロセッサ競合方式による並列配線方式では並列処理効率の点で有効性が確認されたが配線品質の点で劣ることが指摘された．この理由として次の２点が挙げられる．第一点は競合により発生する矛盾の解決を再配線処理だけで行ったことである．このため，配線済みネットが多くなるにつれて再配線のための領域が少なくなり配線率が低下する（3.7節参照）．更に経路探索に他のネットの状態や配線領域の混雑具合などの情報を用いた経路探索を使用しないため，配線経路が特定領域に集中するため配線率が低下する．第二点は，競合方式では配線順序がプロセッサ間の競合により一意的に定まらないため，逐次配線処理を前提とした経路探索法と配線経路の評価方法が不適当なことである[17]．すなわち第３章で用いた経路探索法には配線順序の依存性があり，競合方式ではマスタが割り当てる配線順序と配線結果の評価順序が異なるために配線順序を操作して配線品質を改善することはできない．これに対して逐次配線処理では配線順序を調節することで配線品質の向上を図ることができる．例えば図4-1の例のように逐次配線処理では図(b)に示すようにネットA，B，C，D，E，Fの順に配線することにより正しく配線が行われるが，競合方式では図(c)のようにC，Eが先に配線されてしまうと，Dの配線を正しく行うことができなくなる．

この問題を解決するためには幾つかの方法が考えられるが，最も基本的なものは引き剥し再配線処理である．これは既配線経路を引き剥した後の新しい配線領域上で再配線する方法である[44,45]．但し3.7節で行った実験結果から，単純な引き剥し再配線処理は効果がなく，改善すべき配線経路とその周辺経路も考慮した引き剥し再配線処理を行う必要があることがわかっている．

4.2.3 方針

（1）基本となる計算モデルには第３章で有効性が確認されたプロセッサ競合モデルを用い，ネット割り当て法を用いることでネット単位での並列処理を行うことにする．

（2）改善課題の考察により配線順序に対する依存性を解消し，他のネットを考慮した経路探索を行うことが必要であるが，交差・接触が許容される場合には配線順序に対する依存性は少ないと考えられる．図4-1(b)の例では配線経路の交差・接触を認めないために配線順序が決定されてしまうが，交差・接触を認めれば配線順序

は任意でよい（最終的に交差・接触は解消する必要がある）．

（3）配線経路に交差・接触を許容し，これを最終的に解消するために配線コスト用いた経路探索法を用いる．すなわち交差・接触をコストで表現し，交差・接触のない最適な経路が存在しない場合でも，交差・接触を許容した準最適な配線経路を生成することが可能である．

（4）経路探索における配線品質向上のため，経路探索法には第３章で使用した線分探索法ではなく，最適な配線経路が得られる迷路法を用いる．しかし配線順序に対する依存性は線分探索法と同様にあるので，迷路法を拡張した新しいアルゴリズムを用いることにする．また，配線領域における配線経路の混雑度を経路探索に反映させるため，配線領域に重みを設定することにした．

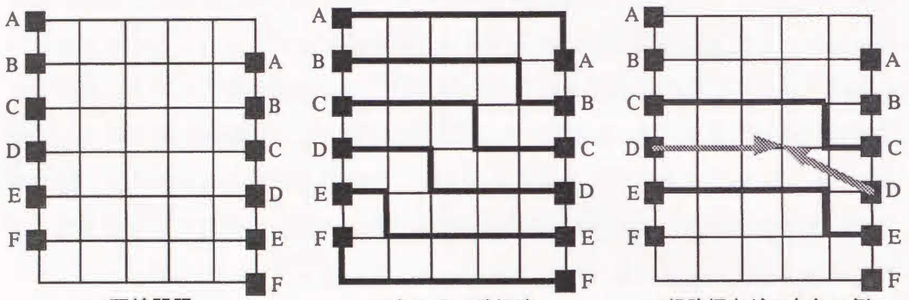


図４-１　配線順序に依存した配線問題

４.３　並列経路改善法

プロセッサ競合方式で得られた経験から，配線順序に対する依存性の小さい経路探索と評価方法による引き剥し再配線処理の反復が，汎用並列計算機のための配線方式として有効であると判断し，新しい並列経路改善方式を提案する．この方式はプロセッサ競合方式とネット割り当て法を用いた並列配線方式に，配線品質改善のための経路改善機構を取り入れており，複数の既配線経路の引き剥し再配線の反復を並列処理することにより並列経路改善を行うものである．また，配線経路に交差・接触を許容することで競合方式により発生する矛盾を経路改善処理により解消することができる．

並列配線方式を配線順序によって分類すると，**同順方式**と**異順方式**の２つの方式に分かれる．同順方式は従来の逐次配線方式と同じ配線順序で並列配線処理するものであり，逐次方式と同じ配線結果と配線品質を得ることができる．多くの並列配線方式はこれに属し，PROTON[32]，タイムワープによる並列無格子配線システム[34,35]（以下タイムワープ方式）等がこれに当たる．これに対して異順方式では与えられる配線問題に依存する配線順序と異なる配線順序で処理するので，高品質な配線結果を得るために引き剥し再配線の反復処理を必要とするので長時間の計算が必要である．富士通のRP[8]がその代表例である．本方式では異順方式を取ることにした．

４.３.１　並列処理方法

配線問題の並列処理方法にはネット内の並列性を利用した配線経路の並列探索と，ネット間の並列性を利用した複数ネットの並列経路探索がある．しかし前者は後者に比べてプロセッサの利用率を向上させることが難しく，これを補うには並列経路探索を行う方法が効果的であることが報告されている[30][32][34]．本方式では以下の理由にから並列処理方法として並列経路探索を採用した．

並列処理による処理時間短縮を図るにはプロセッサ利用率を向上させることが重要である．配線問題では並列探索と並列経路探索により処理時間の短縮ができるが，並列探索では経路探索中に必要な計算量が変化するので，仮に最大の計算量に見合う数のプロセッサを用いれば速度向上性能は向上するが，計算量が少ないときには処理を割り当てられないプロセッサがアイドル状態となり，プロセッサ利用率が低

下する．逆に半分程度の計算量に見合うプロセッサを用いるとすれば，プロセッサ利用率は向上するが速度向上性能は低下する．一方並列経路探索の場合には，配線問題全体の計算量は並列探索よりも遥かに大きいため，多数のプロセッサを利用してもプロセッサ利用率の低下する割合は並列探索に比べて小さいと考えられる．またプロセッサ競合モデルは比較的計算粒度が粗い処理に向いていおり，ネット単位で並列処理する方が効果的であると判断した．このため本方式では複数経路の並列経路探索のみを使用するが，配線経路の並列探索を使用すれば，更に性能を改善できる方式である．

4.3.2 引き剥し再配線

本方式では異順方式による並列配線処理と既配線経路の引き剥し再配線による経路改善処理を用いるが，引き剥すネットをマスタが評価・選択し，スレーブでネットを再配線するようにしている．これは次の理由によるものである．

（1）配線順序に依存しない評価　競合方式ではネット割り当て法を用いるので，各スレーブの処理時間の違いによって並列性が低下するのを防ぐことができる．しかし，配線順序はプロセッサの競合によりランダムになり，順序の依存性の低い選択方法と選択のための評価方法が必要となる．その方法として，配線の要求を満たすために必要な条件，すなわち配線規則をコストやペナルティとして表現し，それを用いて配線経路を評価する方法が考えられる．配線結果を配線規則に対するコストの合計値で表現すれば，配線順序に依存しない評価を行うことが可能となる．

（2）並列性の維持　競合方式では配線結果の矛盾が配線品質を低下させることがあるが，矛盾を経路改善により解消するためにスレーブ間での同期及びそのための通信が不要になり並列性を損なわないという利点がある．このため高いスレーブ利用率が期待できる．なお一般的な方式ではこの種の矛盾を起こさないアルゴリズムを使用している．例えば同順方式のPROTONでは処理範囲を概略配線により分割しその範囲内で逐次処理することにより配線経路の矛盾を防ぐようにしている[32]．

（3）経路改善の分散処理　経路改善の反復処理には反復回数に比例した計算量が必要になるが，ネット割り当て法を用いれば計算量を各スレーブに分散させることができる．また配線に必要な経路探索も並列化できるため，階層的な並列性を利用

[1]することができる．

（4）マスタ処理の簡素化　配線結果の評価はスレーブで処理された配線結果とマスタが持つ最新のデータベースの内容を比較することにより行われる．これはスレーブの処理量に比べて充分に小さく，簡素化することができる．もしマスタに負荷が集中する場合でもマスタ処理を幾つかの機能に分割して並列処理することにより処理時間の増加を抑えることができる[2]．

4.3.3 経路改善法

経路改善法は組み合わせ最適化問題で用いられる逐次改善法に類似している．組み合わせ最適化問題ではパラメタの組み合せを変更して評価を反復することにより最適解の探索を行う．経路改善法ではパラメタの変更は配線経路の変更に相当し，配線経路の評価は配線率や配線長などの配線品質で評価され，経路の変更と評価の繰り返しにより配線経路全体を改善する．この方法では様々な配線規則が関数で表現され，この関数の値を改善するように配線経路が改善される．経路の変更には経路の引き剥して再配線する他に，配線経路を部分的に移動させることもある．

逐次経路改善では改善可能な部分を一度に一ヶ所だけ改善するのに対して，並列経路改善では複数の改善可能な部分を並列に改善する．逐次経路改善はそれぞれの経路改善が逐次的に行われるため，予め改善順序を決定できることから品質改善の戦略決定が行い易い利点があるが，逐次処理のため多大の計算時間を必要とする．これに対して並列経路改善では改善順序の決定が前者より複雑になるものの，２段階の並列性を用いた高度な並列処理が可能になり，計算時間を大幅に短縮することができる．

4.3.4 並列経路改善

競合方式による並列配線では複数のネットを同時に処理できるため，並列経路改善方式でもこの方法を用いる．例えば**図4-2**の１層配線問題の例では，複数の改善

[1]幾つかの並列配線方式で階層的な並列性が用いられており，その有効性が確認されている．本章で取り上げたPROTON，タイムワープ方式もその例である．

[2]マスタでは，配線経路の評価・データベースの更新，引き剥がすネットの選択・引き剥がし・割り当て，及び，次のネット選択のための後処理の３つの機能から構成され，これらを分割して並列処理することによりマスタの負荷を分散して軽くすることができる．

可能な経路（ハッチングされた領域に含まれる部分経路を指す）をネット割り当て法によって並列に処理できるため，経路改善に要する全体の計算時間を問題の並列性に応じて短縮できる．

経路改善の対象は全ての既配線ネットである．なぜなら，経路間の交差などにより改善を必要とする経路が必ずしも改善可能ではなく，その周囲の経路を変更した後に改めて改善可能になる場合があるからである．改善可能なネット[1]が改善されて全体の配線品質が向上すると同時に，その局所的な改善が周囲の配線経路に影響し，配線結果に多様性を与えるため，局所解に陥りにくく，また陥ったときの脱出を容易にする効果が期待できる．

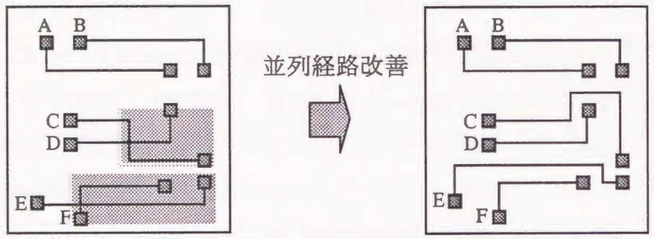


図４-２並列経路改善の例

[1]経路改善は配線経路長やビア数だけでなく他のネットとの関係も含まれる．

4.4　アルゴリズム

4.4.1　アルゴリズムの概要

本章で提案する並列処理アルゴリズム処理手順は次の通りである．図４-３にその流れを示す．

（１）マスタはスレーブにネットデータを割り当てる．

（２）スレーブはデータベースを参照しながら配線処理を行う．

（３）スレーブは配線結果をマスタに送る．

（４）マスタはスレーブの配線結果を評価し，次に未配線ネットを優先して割り当てる．もし未配線ネットが無い場合は既配線ネットから一つ引き剥して割り当てた後，データベースを更新する．

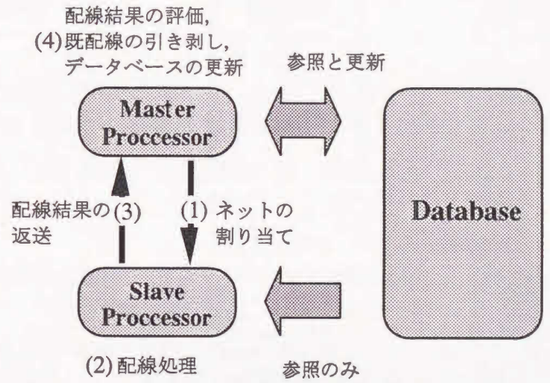


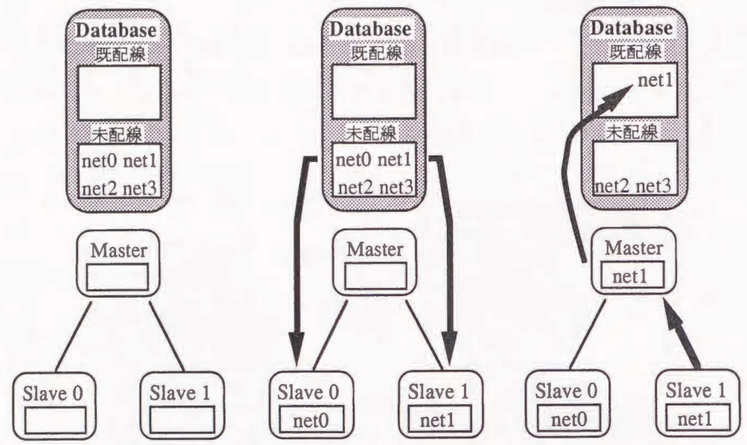
図４-３　並列経路改善方式の処理手順の流れ

次に，図４-４を用いて並列経路改善アルゴリズムの実行例を説明する．この例では２台のスレーブ（Slave0,1）で４本のネットを並列配線する．データベース内部では未配線と既配線に区別されており，図中では上段が既配線，下段が未配線を表す．また，経路探索に必要な経路情報はデータベース内部にあり，ネットの割り当て時にスレーブにコピーされる．

(a)の初期状態ではnet０～net３の全てのネットが未配線状態にあり，既配線ネットは存在しない．まず，処理を開始したスレーブはネットの割り当て要求をマスタに出す．(b) マスタは先着順に要求を受け，各々に異なる未配線ネットを割り当てる．

(c)次に，割り当てられた各スレーブは配線処理を開始する．各スレーブは配線処理

が終ると配線結果をマスタに送り，マスタはスレーブの配線結果を先着順に評価する．スレーブ１の配線処理がスレーブ０よりも先に終ったものとすると，スレーブ１はマスタに配線結果を送り，マスタはその結果を評価し，配線済みネットとしてデータベースを更新する．(d)そして次に処理するネットをスレーブ１に割り当てる．このとき，未配線ネットを優先し，(e) 未配線ネットが存在しない場合は既配線ネットから選択して割り当てる．スレーブ０の場合もスレーブ１と同様である．



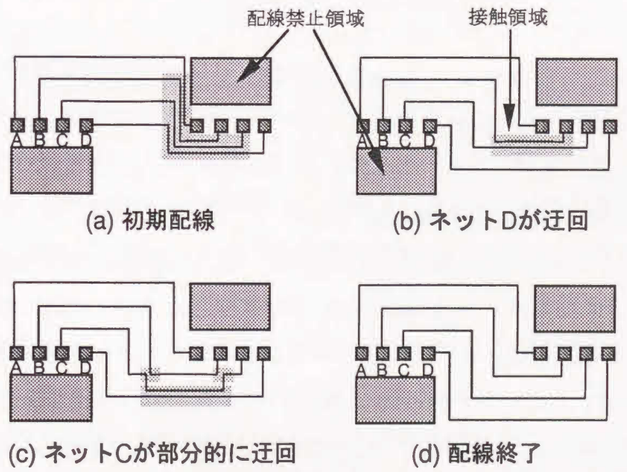
(a) 初期状態　(b)未配線ネットの割り当て　(c)配線結果の評価とデータベースの更新

(d)未配線ネットの割り当て　(e)既配線ネットから選択し引きはがした後に割り当て

図４-４　並列経路改善方式のアルゴリズムの実行例

以後，スレーブとマスタによる引き剥し再配線処理の反復により，並列経路改善が行われる．このようにして全てのネットが既配線となり，他のネットとの交差・衝突などの矛盾が完全に解消されたとき，一つの解が得られたことになる．

図４-５ に簡単な１層の配線問題による並列処理例を示す．これは４本のネットA,B,C,Dの並列処理を４台のスレーブで処理するものである．まずこれらのネットをスレーブに割り当てて並列処理を開始する．最初はデータベースに登録された配線経路が無いので各スレーブは最短距離で経路を生成する．この結果，(a)に示すように全てのネットが網線の部分で接触する．次に引き剥し再配線による経路改善の反復処理が始まり，各経路は並列に改善され，(b)では各ネットの接触部分が小さくなっている．これはネットDの再配線によりネットCとの間に空き領域が生じ，ネットB,Cはその領域を使用することで経路改善されたためである．さらに経路改善を行うと，ネットCは後述する同じ経路の繰り返しに対して与えられるペナルティにより(c)に示すような経路が生成される．その後ネットDが再配線されると，これを迂回する経路が生成され，新たな配線領域ができるため，ネットCも再配線により(d)に示す経路になり，経路改善が終了する．



(a) 初期配線　(b) ネットDが迂回

(c) ネットCが部分的に迂回　(d) 配線終了

- 配線モデルは１層の配線問題．
- 図中の配線経路の接触は，経路間の最低間隔の条件に違反するか，または配線経路上を他の配線経路が交差することを表す．

図４-５　簡単な例による並列配線処理例

4.4.2　経路探索アルゴリズム

本方式における経路探索アルゴリズムには，他の経路と交差・接触のない経路が存在しない場合でもそれらを許した準最適経路が生成できることと，配線順序に対する依存性が小さいことが要求される．そこで配線コストに基づく迷路法を用いることにした[38]．このアルゴリズムでは経路探索に伴う制約をコストで表現し，配線領域に対して配線コストによるラベル付けを行うことにより，始点から終点までの配線コストが最小になる経路を探索することができる．

4.4.2.1　配線コスト

今回使用するコストには５つの要因を考慮して次のような係数を用いて配線コストを定義する．

- **交差**(C)　：配線経路同士の交差に対するコスト係数．
- **接触**(T)　：配線経路同士の接触に対するコスト係数．
- **折れ曲がり**(B)　：配線経路の折れ曲がりに対するコスト係数．
- **ビア**(V)　：異層間を接続するビアに対するコスト係数．
- **配線長**(L)　：配線経路の長さに対するコスト係数．各層毎に縦方向と横方向にそれぞれ設定し，X-Yルールを実現．

配線コスト＝Ｃ＊交差数＋Ｔ＊接触数＋Ｂ＊折れ曲がり数
＋Ｖ＊ビア数＋Ｌ＊配線長

4.4.2.2　既配線経路上の経路探索

配線コストにより既配線経路上の経路探索が可能であるが，単純な迷路法で使用される配線格子だけを用いた経路探索は既配線経路との交差・接触の検出及びその区別が難しく，配線コストを正しく計算することが難しい．また既配線結果リストと比較して検出することができるが，経路同士の比較回数が多く探索時間が増加する欠点がある．仮にこのまま並列処理を行えば，全てのプロセッサに既配線経路を持たせる必要があるため，記憶容量や配線経路の転送に要する通信時間の増加の点で不利である．本手法では既配線経路との比較を避け，配線格子を分割することにより配線格子の情報だけで正しく交差・接触が検出ができる方法[47]を採用した．本手法では既配線経路のある配線格子を４つに分割し，その分割された小格子を通

常の配線格子と見なして経路探索することにより，正しい検出を可能にする．**図４-６**に例を示す．図中の黒色の矢印は通常の探索を示し，斜線でハッチングされた矢印は接触，残りは交差をそれぞれ示す．この図から分かるように既配線経路上の探索は必ず交差か接触になる．

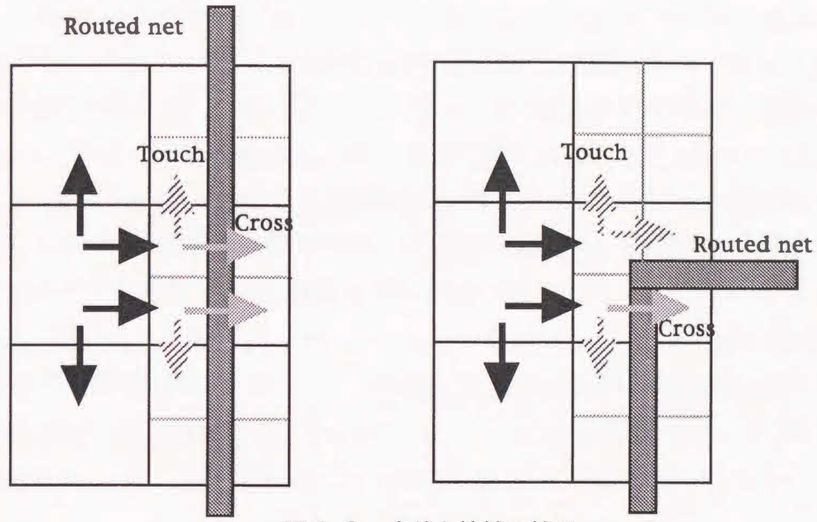


図４-６　交差と接触の検出

4.4.2.3　計算量の削減

迷路法における探索の終了は最初のゴールの発見であり，その計算量は探索点からゴールまでの距離をdとすると$O(d)^2$であるが，本手法では配線コストの最小の経路を見つけるために全域探索を行うため，計算量が迷路法と比較して著しく増加する[38]．そこで本手法で用いた計算量を削減するための探索の打ち切り方法ついて以下に述べる．

（１）探索波の消滅　探索波は探索するグリッドのコストより探索波の持つコストが低い場合に進行し，それ以外は消滅する．

（２）上限コストの設定　探索の上限コストを設けて探索を打ち切る．上限コストには最初に発見されたゴールの探索コストを用いる．ゴールの探索コストが速く求まるほど，残りの探索波は上限コストにより打ち切られる可能性が高い．

（３）探索範囲の制限　ゴールの探索コストを速く求めるために，探索範囲を段階

的に拡大することで枝刈の効果を高め，探索範囲の拡大を防ぐ．

（4）探索コストの見積り　探索波が上限コストに達していなくても，その探索波からゴールまでのマンハッタン距離を用いた探索コストの見積りにより，そのコストが上限コストを越えている場合に探索を打ち切る．

4.4.2.4　アルゴリズム

次にステップ(i)における探索手順を以下に示す．図4-7に探索波進行の様子を示す．

（1）探索波キューより探索波を取り出す．全ての探索波が無くなると探索は終了し，バックトレースにより経路を確定．

（2）探索波に隣接するグリッドが探索可能か調べる．

（3）探索可能なグリッドに探索波が進行する場合の探索コストを計算する．

（4）進行したグリッドがコストを持たない場合，または，既に探索されているが計算したコストよりも高ければグリッドを更新する．更新された場合，このグリッドの情報を探索波キューに入れる．そうでなければその探索波は破棄される．

（5）ステップを(i+1)として（1）へ．

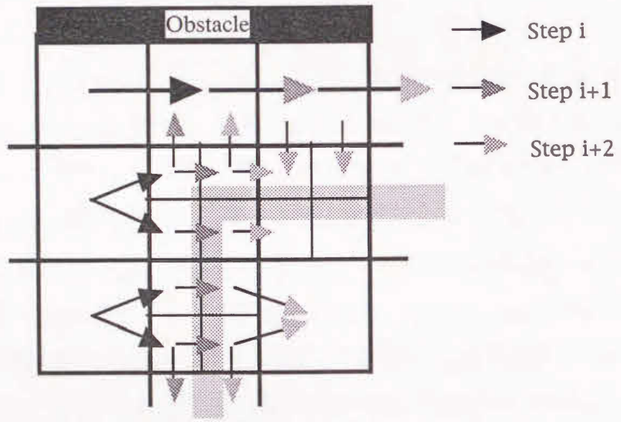


図4-7　探索波の進行の様子

4.4.3　経路の評価

マスタはスレーブの配線経路をデータベースと照合し，競合による配線経路の交差の有無などを検出・評価する．その結果によって配線経路と配線領域にペナルティ

を与え局所的最適状態を回避し，混雑度を緩和する．また，ここでは並列経路改善を効果的に実行するために引き剥す配線経路の選択を行う．

配線経路に対するペナルティ（P）は配線経路の交差・接触の数xの関数$f_p(x)$と前回配線したときの経路との類似度Sの積とする．すなわち，$P= f_p(x) * S$である．これは経路探索に使用する配線コストとは異なるものである．局所的最適状態を回避するため，スレーブの経路探索では各ネットの処理回数に応じて配線コストの係数を変化させるようにする．これにより各ネットの探索コストは探索の度に変化するため，同じネットが繰り返して選択された場合でも異なる探索結果が得られる．このため局所状態が回避できると予想される．しかしネット全体を評価する場合には公平性に欠けるため，マスタでは異なる評価方法が必要であり，このためマスタではペナルティPを用いて配線経路を評価する．

Sの値は［0.5～1.5］とする．ペナルティは関数$f_p(x)$とSの積で表され，S＝0ときペナルティの値が0になるが，次に述べる引き剥がし戦略に支障を来すため，本来の類似度に偏差を与えて用いることにした．このため類似度Sは，Lpを前回の配線経路の長さ，Lnを今回配線した配線経路の長さ，Lsを同じ経路の長さとすると，

$$S = \frac{Ls}{Lp + Ln - Ls} + 0.5$$ とした．

配線領域に与えられるペナルティはその配線領域の混雑度に応じて設定される．その結果，経路探索においてより混雑度の低い領域が選ばれることになり混雑度の緩和に役立つ．

4.4.4　引き剥す経路の選択

引き剥すべき配線経路を選択するには，経路の状況に拘らず順番に選択し引き剥し再配線する方法[8,38]やランダムに選択する方法，あるいは評価結果を基に確率的に選択するなどが考えられる．これらのうち，配線順序に依存するものを除いた最も基本的な選択方法[1]である次の2つの選択方法を用いて評価を行う．

・ランダム選択法　既配線経路数をnとすると，0からn-1までの整数の乱数値を発生させ，この数値を既配線経路の番号とする方法である．配線経路の状態などは一切考慮されない．

---

[1]競合方式では引き剥し再配線の反復とプロセッサ間の競合により配線順序は一意に定まらない．このため配線順序に依存する選択方法では性能を発揮することが難しい．

・**重み付き選択法**　ペナルティPに比例して選択確率が高くなるルーレット戦略[48]を用いており，i番目の配線経路のペナルティP(i)とその合計値をPs＝ΣP(i)の比で選択確率Ps(i)= P(i) / Psを求め，これに比例した確率で選択する方法である．実際には，区間［0，Ps）の一様乱数rを発生させ，ΣP(i-1)≦ r ＜ΣP(i)を満たすiを求めることで選択される（但しi＝0のとき，ΣP(i-1)＝0とする）．図4-8にその模式図を示す．この例では3番目のネットが選択される．

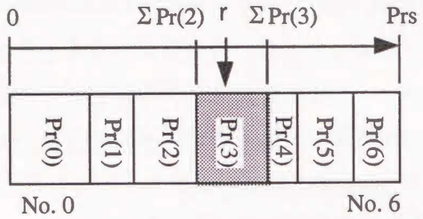


図4-8　選択の模式図

4.5　並列経路改善方式の特徴

本方式の特徴を他の方式と比較しながら述べる．比較対象にはPROTON，タイムワープ方式及びRPを取り上げた．表1にこれらの並列処理法方式に関する特徴を示す．

表1　各並列処理法方式の特徴

	並列処理の種類	ネット間の並列性の抽出	配線順序方式	矛盾解消方法
本方式	複数経路	詳細配線時	異順方式	継続，経路改善
タイムワープ方式	複数経路，経路探索	詳細配線時	同順方式	ロールバック，再計算
RP	経路探索	―――	異順方式	継続，経路改善
PROTON	複数経路，経路探索	詳細配線前	同順方式	―――

注）表中の複数経路，経路探索はそれぞれ複数経路の並列処理，経路探索の並列処理を意味する．

・**並列性**　本方式では並列処理の方法にネット間の並列性を用いた複数経路の並列処理を用いている．他の方式ではネット内の並列性を用いた経路探索の並列処理を行っているが本方式では採用していない．これは，プロセッサ競合方式では経路探索の並列性よりもネット間の並列性の方が効果的に利用できると判断したからである．また，本方式と同様な経路改善を行うRPではSIMD型のハードウエアルータにより高速化を行っているが，ハードウエアルータは経路探索の並列化を目的としたものなので，ネット間の並列性に対応することができない．

・**並列性の抽出方法**　並列性の抽出は，PROTONでは詳細配線前に概略配線を行い，予めプロセッサへ処理を割り当てる静的方式であるのに対し，本方式を含む他の3つの方式では詳細配線時に動的に処理を割り当てる方式である．静的方式では並列性やその計算量を見積れる利点があるが，引き剥し再配線を行う場合には並列性を再抽出する必要がある．PROTONでは経路間の矛盾を避けるため概略配線経路同士が重なる場合は逐次処理するようになっているが，実際には図4-9に示すように，概略配線経路が重なっていても並列処理可能な場合がある．動的方式では全体の並列度や計算量を把握することは難しいが，その時々の配線状況により並列性が抽出されるため図4-8に示した並列処理も可能である．また，並列性抽出のための前処理が不要になる利点がある．

もし本方式に静的方式を用いるとすると，引き剥し再配線の度に並列性再抽出の

ために前処理の計算を行う必要があり，特に大規模な配線問題ではネット数が著しく増加するので前処理に要する時間が無視できないものとなる．従って本方式では動的方式を用いることにした．

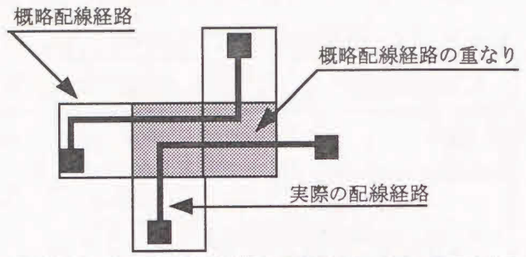


図4-9　配線可能範囲が重なる場合の並列処理

・矛盾の解消方法　タイムワープ方式では配線情報を矛盾のあった時点まで履歴を遡って矛盾が生じる前の状態に復元（ロールバック）してから再計算するが，本方式では配線情報を継続使用して経路改善を行うため，ロールバックが不要となり通信回数を減らすことができる．矛盾がなく余分な再探索の計算量を削減する意味ではロールバック方式が有効であるが，本方式では配線経路の矛盾を許容しており，発生する矛盾を直ちに解消する必要がなく，ロールバック処理の必要性は少ない．

引き剥しのない同順方式では配線順序決定方法が配線品質に強く影響するが，異順方式ではその影響は少ない．そして最適化問題として見た場合の異順方式は同順方式より解空間の探索範囲が広いため，同順方式よりも高い配線品質が期待できる．同じ異順方式であるRPでは逐次反復処理により経路を改善する方法を用いており，その配線順序は逐次配線方式で用いられるものとは異なるが，逐次的に処理されておりその順序関係は保証される．一方，本方式では配線順序関係は一意的ではなく，配線品質の低下を防ぐために配線順序に対する依存性の低い経路探索アルゴリズムと評価方法を用いている．またプロセッサ間の競合により得られる処理結果の多様性を利用して局所解からの脱出を容易にしている．

4.6　実装・評価

4.6.1　配線問題と実験条件

実験には第３章で用いたMIMD型並列計算機Coral-68Kを使用した．この実験で取り扱った配線問題と実験の条件は次の通りである．

・グリッドサイズ64×64の二層配線問題[1]．
・配線規則にはＸ－Ｙルールを適用．
・ＩＣの形に似せて生成した端子群から２端子間をランダムに結ぶ90本のネットを生成し，それらのネットから構成される多端子ネットの理想配線長が最短になるように最小展張木（Minimum Spanning Tree）を用いて端子間の接続を再構成した２端子ネットを使用．使用した２種類のネットデータの諸元を**表３**に示す．
・引き剥すネットの選択方法には，ランダム選択法と重み付き選択法を使用．
・経路探索の並列化は行わない．
・配線終了条件は配線率100%になること．

表３　ネットデータの諸元

項目	データ１	データ２
ネット数	９０	１２１
端子数	１８０	２４２
理想総配線長	３１２３	３２９０
理想平均長	３４.７	２７.２
分散	３２３	２６４
最長経路長	９８	９２
最短経路長	６	２

4.6.2　実験結果

実験結果を**図4-１０**に示す．グラフ上の各点は２０回の実行結果の平均値である．なお配線品質の項目は配線率，総配線長，総ビア数とするが，配線率は100%を前提としているので，総配線長と総ビア数についてのみ比較する．各グラフは，(a)スレーブ数を変化させた時の引き剥し処理回数と処理時間，(b) 全体の速度向上比とネット一本当たりの平均処理時間の速度向上比，(c)総配線長，(d) 総ビア数，(e) スレーブ

[1]Coral-68Kでは実装メモリの制約から256×256の二層配線問題を評価できなかった．そのため，より大規模な配線問題を評価するには別の並列計算機上で行う必要があり，今後の課題の１つである．

数８のときの交差・接触数[1]を示す.

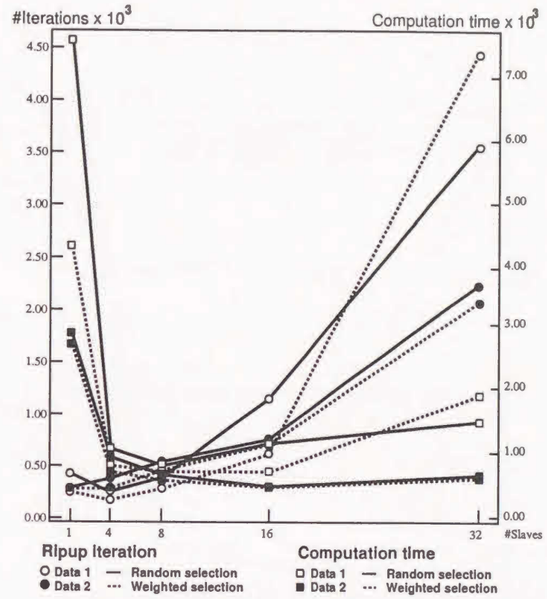
図(b) からデータ１はスレーブ数が８の時に最も処理時間が短縮されており，この時の速度向上比は６.２倍，データ２ではスレーブ数１６の時に６倍となっている．ネット当りの速度向上比はスレーブ数に比例しており，マスタにおけるボトルネックが殆ど発生していないことが判る．これはプロセッサ競合方式においてマスタのプロセスがスレーブのプロセスに比べて充分に小さく，スレーブの利用率が高いことを示している．一方，全体の速度向上比はデータ１ではスレーブ数８，データ２ではスレーブ数１６をそれぞれ境に低下しているが，この理由はスレーブ数の増加がもたらす競合による矛盾発生数の増加であると考えられる．これは図4-10(a)の引き剥し回数とスレーブ数の関係から，引き剥し回数の増加は競合によるものであると判断できる．以上の結果から，ネット当りの処理速度については充分な台数効果が得られているが，並列配線処理全体の速度向率ではある台数を越えると減少することが判る.

次に配線品質を見ると，総配線長はデータ１，２ともにスレーブ数の増加に対して１%程度しか改善されていない．これは総配線長が総理想配線長の約１.１倍程度と改善の余地が小さいためである．一方総ビア数はデータ１，２とも最大で１割改善されている．データ１ではスレーブ数の増加に従って改善率も増えるが，データ２ではスレーブ数８の時に最も改善されていることがわかる.

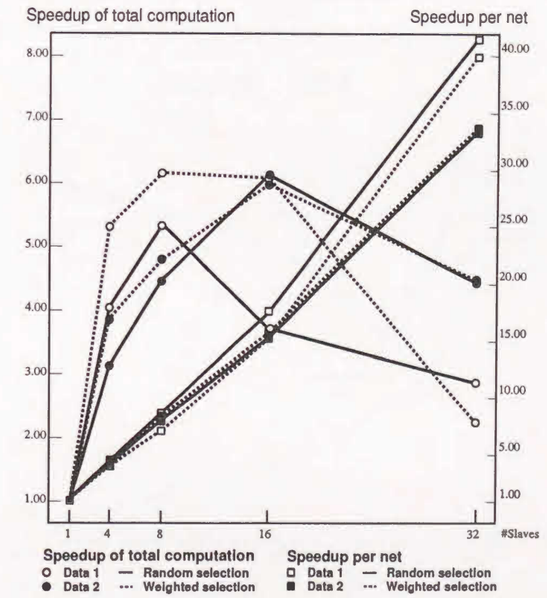
選択方法による違いを見ると，データ１では重み付き選択方法が総配線長，総ビア数共に効果的である．一方データ２では総配線長に関しては効果が見られるが，総ビア数に関しては効果が小さい．また選択法の違いによる全体の並列性に関しては，図4-10(b)より，台数の少ない場合から速度向上比が最高になる点までは重み付き選択法，それ以降はランダム選択法が効果的であることが確認された．また図4-10(d)では並列処理の前半の３００回迄は早い速度で交差・接触数が減少するが，後半では遅くなっていることがわかる.

---

[1]グラフの縦軸であるIterationは，引き剥し再配線処理回数を表しており，マスタによる引き剥し，スレーブによる再配線，マスタによる配線経路の評価を１Iterationとしている.

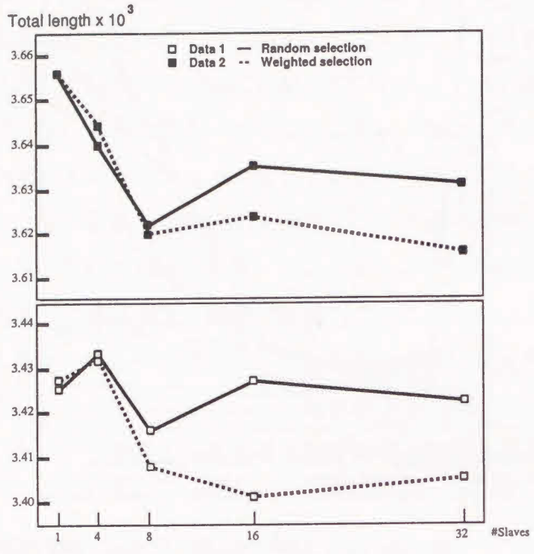


（a）引き剥し回数及び総処理時間とスレーブ数の関係

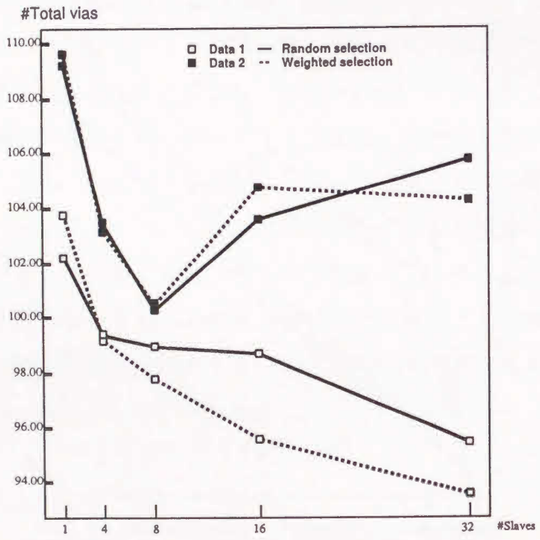


（b）全体及びネット当たりの速度向上比とスレーブ数の関係

図４-１０　Coral-68Kによる実験結果

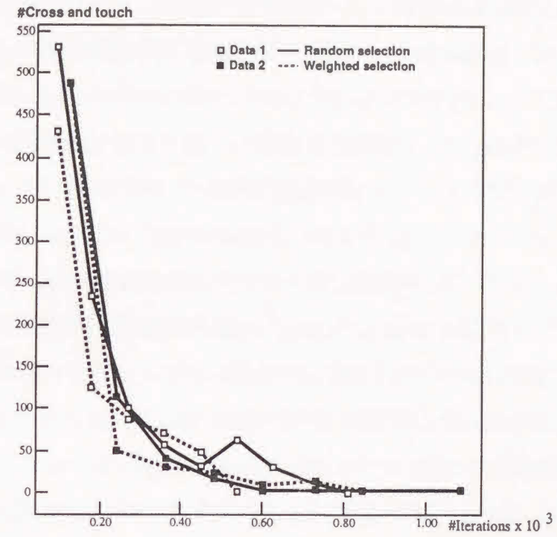


(c) 総配線長とスレーブ数の関係



(d) 総ビア数とスレーブ数の関係

図4-10　Coral-68Kによる実験結果



(e) 交差・接触数と繰り返し回数の関係（8スレーブ）

図4-10　Coral-68Kによる実験結果

4.7 考察

実験結果から並列経路改善方式の有効性と問題点について考察する．

（1）配線品質　配線品質には配線率，配線長，ビア数などがあるが実験では100%の配線率を前提としているので，総配線長と総ビア数について考察する．**図4-10**の結果では総配線長(c)が１%程度改善され，総ビア数(d)は６%～１０%改善されている．この理由を考察する．まず総配線長の改善割合が小さいのは総配線長と理想総配線長の比が小さく，配線問題の改善の余地が小さいことを示している．これは使用したネットデータが一様乱数によって作成されており，ネットが配線領域全体に分布しているためと考えられる．次に総ビア数ではデータ１ではスレーブ数の増加に従ってより改善されている．これは例えばスレーブ１台の場合を考えると，その配線処理は逐次的になり探索できる経路の組み合せは少ないが，台数が増加するにつれて競合による効果も含め探索できる組み合わせが増加し，より良い解の発見に結び付いたものと判断できる．一方データ２ではスレーブ数が８の場合には前述の理由が当てはまるが，スレーブ数が多い場合には品質改善効果の低下が観測され，問題によってはスレーブ数の増加による矛盾割合の増加により品質改善効果の低下が起きると推測される．

本方式では複数ネットの並列改善処理を採用しているが，スレーブ１台の時[1]に比べて配線品質が低下していないことから，配線経路の並列探索のみの場合と比較しても品質面では遜色無いことが判る．

（2）並列性　ネット一本当りの並列性は充分であることから各スレーブの利用率は高く，スレーブの並列性維持及び経路改善の分散処理の有効性が確認された．一方スレーブ数の増加による矛盾発生割合の増加は配線結果の収束性に影響を与え，処理回数を増加させる結果となった．また処理回数の増加による全体の並列性の低下が確認された．そこで全体の並列性についてスレーブ数を固定した場合と変化させた場合について考察する．

スレーブ数を固定した場合，**図4-10**(e)で示すように並列処理前半（約３００回まで）では交差・接触数（矛盾数）が多く，選択される確率も高いため経路改善に関するスレーブの利用率は高いが，改善の進行による交差・接触数の減少はスレーブの矛盾解消に寄与する実質的な利用率を低下させる．これは配線問題の並列性が配線処理の進行に従って低下することを意味する．この場合並列性の低下を別の方法で補う必要がある．例えば実質的な並列性の低下により必要とするスレーブ数が減少するため，結果として余ったスレーブを他の処理に割り当てることで全体としての並列性を維持する方法が考えられる．例えば余剰スレーブを用いたネットの並列性による経路探索の並列処理や，配線問題を多端子ネットに拡張してその部分経路の探索を並列処理するなどの方法がある．

スレーブ数が増加した場合は矛盾の発生割合が台数増加に従うため，処理の終盤に矛盾改善のための反復処理を多く必要とする結果となり，実質的な並列性が低下する．そこで必要に応じてスレーブ数を変化させるか，スレーブに割り当てる場合の割り当て戦略を改良することにより矛盾発生割合を抑制することが必要である．

（3）ネット選択方法の影響　配線品質では重み付き選択法が配線長，ビア数共に効果を上げており，ランダム選択法に比べて有効性が高いといえる．一方並列処理ではスレーブ数が少ない場合に重み付き選択法，多い場合にランダム選択法が良い結果を示している．今回の実験では二つの選択方法しか評価していないが，実際には数多くの方法が考えられるので，配線状況に応じて異なる選択方法を組み合わせる方法が有効であろう．

（4）最適プロセッサ台数　現在のアルゴリズムで，データ１では８台が最適なスレーブ台数となっており，総ネット数に対する最適プロセッサ数の割合が全ネット９０本に対して９%程度に相当する．データ２ではネット数１２１本に対して，速度向上の点で１６台（１３%），配線品質では８台（７%）の場合が最適な台数となっている．この割合は配線問題の性質によって左右されるが，１０,０００本規模の配線問題において，仮に５%の最適台数が得られるとすると５００台のスレーブが使用できる．現在利用できる並列計算機で使用できるプロセッサ台数を考慮すると，これは実用上充分な拡張性である．つまり総ネット数に対する最適プロセッサの割合が低くても配線問題の規模が大きい場合，本方式は充分な拡張性を持つことができる．

（5）競合方式との比較　第３章で述べた競合方式と本方式を比較した場合，**図3-13**(b)に示す配線率の変化からも明らかなように，競合方式では配線経路の迂回路が無くなると再配線を行っても配線できなくなる．しかし本方式では初期配線が終っ

[1]スレーブが１台の場合，本方式は逐次経路改善になるため，RPで用いられる配線経路の並列探索のみを行った場合の配線品質に相当する．

た段階では配線率が低いが反復処理により改善されるため，迂回路が存在しない場合でも改善処理が停止することはない．さらに，生成された経路が周囲の経路に影響を与えるために混雑度の緩和に貢献し全体として品質改善につながる．実際両方式をネットデータ１を使用して比較すると，競合方式では９０％の配線率に対して本方式では１００％の配線率であった．また競合方式では，配線結果が衝突した時のみ再配線が行われるため，ある時点における再配線率は競合の発生率と同等である．これに対して本方式では再配線率は使用するスレーブ数に依存している．例えばデータ１において９０本のネットで８台のスレーブで処理した場合，８／９０≒８.９で約９％の再配線率となる．つまりスレーブ数を変化させることにより再配線率を変化させ，競合発生割合を変化させることが期待できる．その場合処理中の最適台数を決定する方法（例えば配線経路の競合の計測など）により，常に問題に適したスレーブ数で処理を行うことが可能になる．

以上のように，改善すべき点が残されてはいるが本方式のスレーブ数増加による配線品質の向上の特徴は，汎用並列計算機，特に超並列計算機において探索できる組み合わせ数の多さと競合方式の汎用並列計算機に対する依存性の少ないことを考慮すると，本方式の有効性は高いと言える．

４.８ まとめ

本研究の目的は汎用並列計算機に対して依存性が低く，効率の良い並列アルゴリズムを開発することである．そして配線問題に対するプロセッサ競合方式の経験から，逐次処理方式（同順方式）より高い配線品質を得るには異順方式による引き剥し再配線処理の反復が効果的であると判断し，プロセッサ競合方式による並列経路改善方式を提案した．同方式はプロセッサ競合方式の並列処理方式と配線品質向上のための並列経路改善機構により，複数ネットの引き剥がし再配線の並列処理を行う．競合方式による引き剥し再配線処理はこれまでの並列配線方式と異なり，配線順序が一意的でないため，配線順序に対する依存性を押さえるための配線コストを用いた経路探索アルゴリズムと，ペナルティを用いたネットの評価及び引き剥しのための選択方法を用いた．

本方式を第３章で述べたプロセッサ競合方式における改善課題であった配線品質の向上に適用した結果，配線品質の向上とその台数効果を確認した．またネット当たりの並列性は充分あることが確認され，マスタとスレーブの処理のバランスは適当であることが確認された．しかし全体の並列性はネット当たりの並列性と比較して不充分であった．これは経路改善の進行による配線問題の並列性の低下とプロセッサ数の増加による配線経路間の矛盾発生割合の増加が原因であるが，考察の結果，プロセッサ数の動的変化と余剰プロセッサの効果的な割り当て，及びスレーブへのネットの割り当て戦略の改良によりこの問題は改善可能であると考えられる．

ランダム選択法と重み付き選択法を比較した結果，ランダム選択法はプロセッサ数が多い場合の並列処理において効果的であり，重みを付加した選択方法は配線品質面で効果があるが，最終的には複数の選択方法を組み合わせたものが有効であると思われる．またスレーブにおける経路探索方法にも複数の方法が考えられ，改善の余地が残されている．

今回の評価ではスレーブの処理量がマスタの処理量と比較して充分に大きく，スレーブ数が３０台程度と多くなかったため，マスタにおけるボトルネックは殆ど観察されなかった．そのため最適なスレーブ台数では殆ど問題無いと言える．

第５章

部分引き剥し再配線法による並列処理のための多端子ネットの経路探索法

５.１　まえがき

第４章で述べた並列経路改善方式はプロセッサ競合方式において課題であった配線品質を向上させたが，取り扱う配線問題は２端子ネットであり，多端子ネットの配線問題は２端子ネットに分割して配線する必要があった．そしてその配線結果は配線経路の途中での分岐を許さないために冗長的になり，配線品質向上の障害になっている．そこで本章では２端子ネットの配線問題を多端子ネットの配線問題に拡張し，その並列配線方式について研究した結果，多端子ネットの並列処理を考慮した部分引き剥し再配線法による経路探索法を提案する．この手法は，多端子ネットの経路探索が複数の部分経路探索で構成されることに注目し，多端子ネットの引き剥し再配線をする場合に，複数ネットの配線を並列に処理するとともに部分経路探索のプロセスを動的にプロセッサに割り当てることにより細粒度の並列処理を可能にするものである．本手法では部分経路探索の結果得られる複数の経路を仮経路とすることで部分経路間での探索継続に必要な情報量を削減するができる．本手法の逐次アルゴリズムをWS（ワークステーション）上で評価した結果，経路探索回数をネット全体を引き剥し再配線する方法と比較して約半分に抑えられることが確認された．

本章では５.２節で本研究の目的と方針について述べ，５.３節では多端子配線問題と並列経路改善法への拡張に対する問題点とその解決方針を明確にする．５.４節では経路探索の具体的な方針とそのアルゴリズムについて述べ，５.５節では提案する

方式を逐次処理方式と比較した結果について考察する．５.６節，５.７節では本方式の並列配線処理に対する本方式の適合性と，並列性の抽出方法について述べる．

5.2　研究目的と方針

本章における研究目的とその方針について，まず背景を述べた後，本研究の目的と方針について述べる．

5.2.1　背景

近年の電子装置の高密度化によりプリント基板やＶＬＳＩの配線問題は益々複雑化し，それに必要な計算時間を現実時間の範囲に抑えるために問題を効率よく解くための各種の方法が研究されている．特に並列計算機による並列配線処理は，従来の逐次計算機で処理されてきた配線問題を高速に処理する最も効果的な方法として注目されているが，高並列度の処理を行うには従来の逐次アルゴリズムとは異なるアルゴリズムの開発が必要である．また，これまでに発表された並列配線処理アルゴリズムは並列計算機のアーキテクチャに対する依存性が高いため，特定の並列計算機でないと期待する性能を得られないものが多かった．これに対して，本研究では第３章で述べたようにプロセッサ競合方式による並列配線方式によりアーキテクチャに対する依存性を抑え，第４章に述べたように同方式の配線品質面を改善した並列経路改善方式を提案した．この方式では複数ネットの経路改善を同時に処理することで計算時間が短縮されるが，取り扱う配線問題を２端子ネットに限定したため，多端子ネットが含まれる実データを処理するには２端子ネットに分割する必要があり，冗長な経路が生成され，配線品質向上の妨げとなる欠点があった．そこで多端子ネットを配線する並列配線方式を研究し，前述の欠点を解消できるアルゴリズムを開発する必要があった．

5.2.2　研究目的

本研究は並列経路改善方式による多端子ネットの配線問題の並列処理を目的とし，前述の欠点を解消するために並列経路改善方式を多端子ネットの配線問題に拡張してその検証を目標とする．そのため，まず多端子ネットの並列探索が可能な経路探索法を開発し，その有効性を検証を行うことにする．これは問題拡張をする上で最も基本となるものである．

5.2.3　基本方針

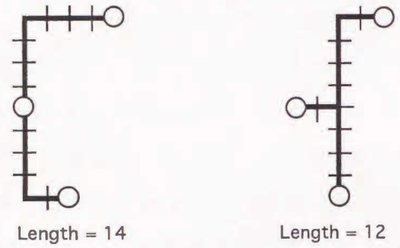
多端子ネットの経路探索が複数の部分的な経路探索から構成されることに注目し，部分経路探索を割り当て単位とする並列経路探索法により，多端子ネットを一度に経路探索する方法と比較して計算粒度を細かくし，並列性の向上を図る．また問題拡張に伴う計算量の増加を抑えるための配線経路の部分引き剥し再配線方式について検討する．多端子ネットの経路探索は２端子ネットよりも多くの探索を行うため計算量が増加するが，ネットを構成する部分経路のうち必要な部分のみ改善することにより計算量の削減を図り，経路改善において配線経路を部分的に引き剥すことにより再配線回数を削減することができる．また部分経路の探索結果から得られる情報を削減することにより，経路探索継続に係る通信量の増加を押えることができる．

5.3　多端子ネットの配線処理

多端子ネットの配線問題の並列処理を行うために，基本的な配線問題と逐次経路探索による近似解法ならびに並列処理方式と並列経路探索の方針について検討する．

5.3.1　簡単な多端子ネットの配線問題

多端子ネットの配線問題は図5-1の例のように，複数の端子間を配線するものであり，単純に最小展張木（Minimum Spanning Tree）を用いて2端子ネットに分解して配線すると，図(a)のようになる．しかし図(b)のように配線経路に分岐点を設けた木構造にすれば経路長が短縮される．このような木構造を求める問題は，Rectiliner Steiner Tree（ＲＳＴ）問題として知られているが，経路探索を必要とする配線問題では最適解を求めることは難しく，一般的には近似解が用いられる[7][49,50].



(a) 2端子ネットによる配線 (b) 分岐点による経路長の短縮

図5-1　簡単な多端子ネットの配線問題

5.3.2　逐次経路探索法と問題点

逐次探索による近似解の例を図5-2を用いて説明する[7]．(1)まず代表となる端子t_0を選択し探索を行う．(2)そして最初に見つかった端子t_1までの最短経路が通過できる領域R_0をマークし，次の端子を探索する．(3)次に見つかった端子までの最短経路が通過できる領域R_1をマークし，R_0，R_1の交点をj_0とし，t_0，t_1，j_0を最短で通過する経路r_0を確定する．(4)そして確定された経路r_0と領域R_1から次の探索を行う．この操作を繰り返すことで(5)に示すようなRSTの近似解が得られる．但しこの方法では探索途中結果を保存する必要があるため，ネット内の並列性による並列探索は，通信量が多くなり分散メモリ型並列計算機へ実装するのは難しい．

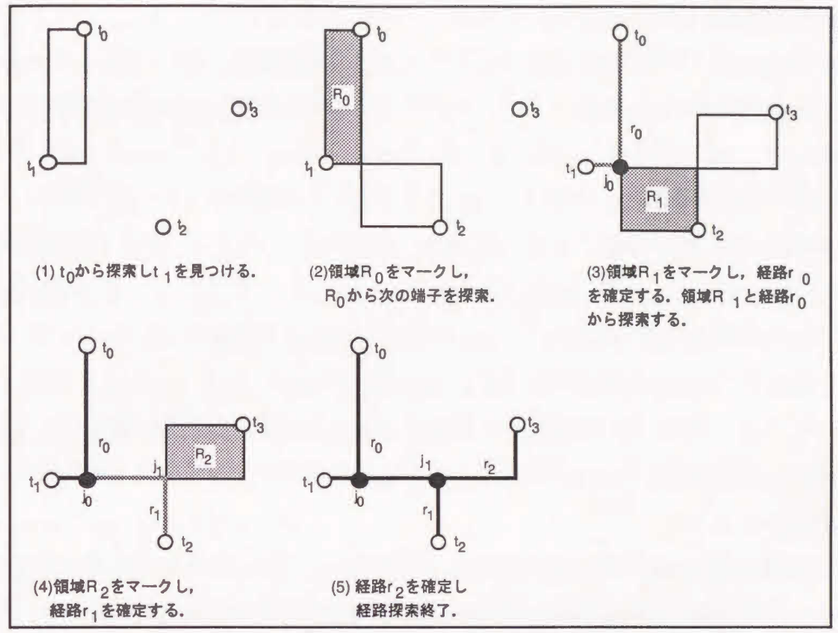


図5-2　逐次探索による近似解法の例

5.3.3　並列処理方式

並列経路改善方式を多端子ネットの配線問題に適用する場合の問題点について考察する．多端子ネットの経路探索には複数の部分的な経路探索を行う必要があり，ネット当たりの探索面積が2端子ネットの配線問題と比べて増加する．このためネット割り当て法によるネット単位の配線処理を行うと，探索面積が増加するため並列処理の計算粒度が粗くなる．探索面積の増加に伴ってデータベースを広範囲に参照する必要があるため，参照のための通信頻度が増加し，データベースへのアクセスの集中を招きボトルネックを生じることになる．またデータベースの一貫性保持の点からデータベースのアクセスに排他制御を行う必要があるが，そのために他のプロセッサからのデータベースへのアクセスを大幅に制限し，プロセッサ利用率を低下させる原因となる．このため配線処理全体の性能を低下させることになる．また並列経路改善処理の終盤で改善すべき配線経路数が少なくなると，残された経路改善数が減少し，有効な経路改善数に対してプロセッサが過剰になる．しかし処理割り当てはネット単位で行われるため，有効な改善処理を割り当てられないプロセッ

サが無駄になる．

これらの問題点を解決するには各プロセッサの処理粒度を小さくすることが必要である．粒度を細かくしてデータベースのアクセス時間を短くすれば，同期待ち時間が短くなり，並列性の低下が抑えられる．但し，逆にプロセッサ間通信によるオーバヘッドが増加するため，両者のバランスを取ることが重要となる．多端子ネットの経路探索の処理粒度を細かくする方法は，並列処理方式や実装方法及び経路探索手法などに左右され，一概に決定するのは難しい．しかし本方式は並列計算機方式の基本的な特徴のみで構成されており，計算機方式に対する依存性が低いため，各種並列処理方式に対する実装性を考慮しなくてもよい．

以上の考察の結果，次の理由から多端子ネットの経路探索の処理粒度を小さくするために，経路探索をネット単位でなく部分経路毎の細かい探索単位で並列処理することにした．

（1）多端子ネットを経路改善する場合，そのネットの全配線経路が改善対象になるとは限らない．このため，必要な部分経路だけ改善を行うことができれば経路改善に必要な計算量を削減でき，平均的な計算粒度を小さくできる．

計算粒度を小さくする点では共有メモリを用いた実装が有効である[1]とされるが，本研究で採用する計算モデルは並列計算機方式に対する依存性を抑えるために共有メモリを用いておらず，同様の並列処理を行うには通信頻度の増加による性能低下を我慢しなければならない．しかし部分経路単位で並列探索を行う方法では計算粒度は共有メモリを使用する場合より粗くなるが通信頻度を削減することができ，プロセッサ間の結合密度が疎である計算モデルに向いている．

（2）多端子ネットの経路探索では部分的な経路探索を段階的に行うため，それぞれの探索を逐次的に行う必要があり並列処理の障害となるが，複数のネットを並列に処理することにより各ネットの探索の逐次性を保存したまま並列処理ができる．更に，ある部分経路の探索結果が他の部分経路の探索に影響しない場合にも部分経路探索を並列に行うことができる．このような依存関係の小さな配線経路が予測ができれば，部分経路探索を並列に処理することができる．

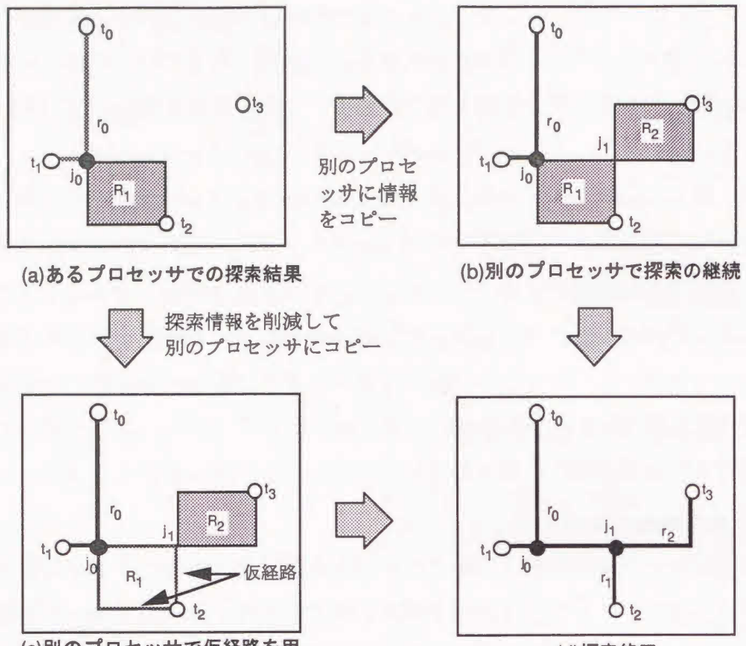
---

[1]プロセッサ間通信が短縮でき，その分計算粒度を小さくすることができる．これは第３章の分散メモリ型と共有メモリ型の実験結果の比較から確認される．

以上のことから，各プロセッサに対する処理割り当て戦略には２つの方式が考えられる．一つは，あるプロセッサに対してネットを割り当てると，そのネットの全ての部分経路探索を終えるまで割り当てを固定し，部分経路探索を終える毎にデータベースを更新する方式（静的割り当て）であり．他は，部分経路探索ごとに割り当てるプロセッサを特定せず状況に応じて変化させる方式（動的割り当て）である．前者では割り当てられたプロセッサだけがそのネットの部分経路探索を行うため，部分経路探索は逐次処理となり，そのネットの並列探索が可能な場合を有効に活かすことができないが，後者では部分経路探索単位で割り当てを変更できるため，この問題点を解消できる．本方式では割り当てアルゴリズムは多少複雑になるが，より高い並列性が得られる動的割り当て方式を採用する．

５.３.４　並列経路探索の方針

部分経路単位でプロセッサへの割り当てできる経路探索法では，前述のように，配線領域の全情報を保存しつつ探索することは通信量増加の点から難しい．例えば**図５-３**に示すように，ある部分経路の探索後に別のプロセッサに探索が継続される場合を考える．配線処理を別のプロセッサが継続するには**図(a)**に示すように，探索途中の領域R_1と既に確定された部分経路r_0の情報が必要である．そして，探索を引き継いだプロセッサでは部分経路と探索領域の情報を用いて探索を継続する**図(b)**．この場合には部分経路の情報量は少ないが配線領域の情報はR_1の面積に比例するため，この面積が広がると通信量が増加しボトルネックを生じる．従って探索継続のための通信量の増加を抑える必要がある．ここで通信量削減方法について考察すると，探索継続に必要な配線領域の情報は全体でなく，一部分のみでよいことが解る．すなわち，**図(a)**中のR_1の領域全体の情報を必要とせず，**図(c)**に示すようにR_1上の幾つかの配線経路の候補（以下，仮経路）があれば**図(d)**のように探索が継続できる．これにより継続のための情報量が削減され，通信量の増加を抑えられる．



(a)あるプロセッサでの探索結果　(b)別のプロセッサで探索の継続

(c)別のプロセッサで仮経路を用いた探索の継続　(d)探索終了

図5-3　異なるプロセッサ間での経路探索の継続

5.4　アルゴリズム

前述の方針による並列配線処理を考慮した多端子ネットの部分引き剥し再配線処理における経路探索手法と引き剥し再配線戦略について述べる．

5.4.1　経路探索

多端子ネットの経路探索を複数の部分経路の探索に分割して行う経路探索法について述べる．これは5.3.4節で述べた方針から，部分経路の探索結果を仮経路の集合として取り扱うことにより探索結果の情報を少なくし，探索継続のための通信量を削減するためである．

部分経路探索は，設定された探索開始位置から探索し，発見したゴールに対して得られる複数の部分経路のうち，有効性が高いと判断される複数の経路を仮経路とする．以前に探索された仮経路又は既に配線された部分経路を探索ゴールとする仮経路の中で最も望ましい仮経路を新しい部分経路とする．同じ探索ゴールを持つ複数の仮経路が存在する場合には，部分経路の配線コストの増加が最小となる仮経路を選択して新しい部分経路とし，他の仮経路は削除する．

このアルゴリズムは次の通りである．

（1）初期探索開始点を選択．

ネットの最小展張木（Minimum Spanning Tree）により隣接する端子との平均距離が最短となる端子を初期探索開始点とする．

（2）探索ゴールを設定し部分経路探索を実行．

探索開始点を含む部分経路以外の同ネットの全ての端子，部分経路及び仮経路をゴールとして部分経路探索を実行し，複数のゴールに対する配線経路を得る．

（3）探索結果から仮経路を決定．

得られた複数の経路から無駄な経路を除いた残りの有効性の高いものを仮経路とする．

（4）仮経路の中で確定できる経路を検査．

仮経路が確定可能かどうか検査し，複数の確定可能な経路が同じゴールを持つ場合には，新たに増加する部分経路の配線コストが最小となる仮経路を確定して他を削除する．確定できない仮経路は次回以降の探索ゴールとして

用いられる．

（5）次の探索開始点を選択し２へ．

得られた部分経路と仮経路から最も近い未探索端子を次の探索の開始点として選択し２へ戻る．未探索端子がなくなり，ネット内の全ての端子が配線済みであればこのネットの探索は終了する．

図５-４に経路探索の例を示す．この例は，処理対象の多端子ネットを逐次的に部分経路探索をして配線処理した場合である．

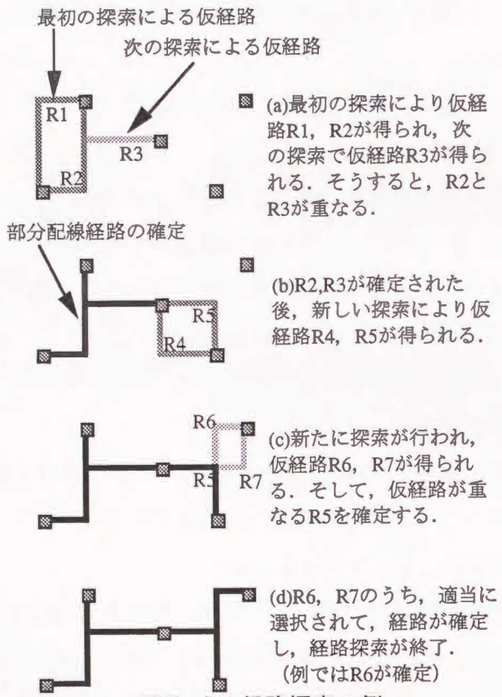


図５-４　経路探索の例

本方式における部分経路探索アルゴリズムでは，第４章で用いたものと同じように，他の経路との交差・接触のない経路が存在しない場合でもそれらを許した準最適経路が生成でき，経路探索に伴う制約をコストで表現することで，始点から終点までの配線コストが最小になる経路の探索が可能である．使用した配線コストの係数は，**交差**(C)，**接触**(T)，**折れ曲がり**(B)，**ビア**(V)，**配線長**(L)の５種類であり，配線コストは以下の式で定義される（詳細は４.４.２節：経路探索アルゴリズムを参照）．

配線コスト＝C＊交差数＋T＊接触数＋B＊折れ曲がり数
＋V＊ビア数＋L＊配線長

５.４.２　木構造表現

本研究では部分引き剥し再配線を容易にするために，配線経路を仮想端子を導入した木構造で表現する方法を用いた．仮想端子とは実際の端子ではないが，ベンド点，ビア及び分岐点を端子と見なすためのものである．本研究で用いる配線方法はＸＹルールを用いるため，図５-５に示すように，配線経路は仮想端子または端子からなる木の節点とそれらを結ぶ線分経路を表す枝から構成される．各端子（仮想端子を含む）と線分経路はそれぞれリストを構成することで検索を容易にする．また，線分経路を表す枝にはその線分経路の評価によるペナルティを与え，引き剥がす部分経路を選択するために使用する．これは第４章で述べたペナルティと同様であるが，先ではネット毎に計算したが今回は線分経路単位で求める点が異なる．従って線分経路に対するペナルティ（Ｐ）は，線分経路の交差・接触数の合計数ｘに対する関数$f_p(x)$とその線分に対応した前回配線したときの線分経路との類似度Ｓを用いて，$P = f_p(x) * S$で与える．この類似度Ｓも第４章で使用したものと同様であるが，Ｌｐを前回配線された線分経路の長さ，Ｌｎを今回配線した線分経路の長さ，Ｌｓを経路の重なる部分の長さとすると，$S = \frac{Ls}{Lp + Ln - Ls} + 0.5$とした．

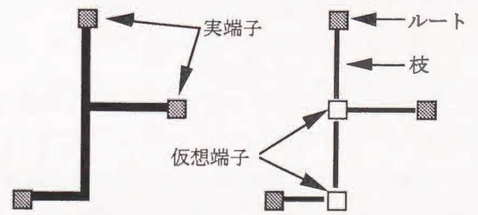


図５-５　配線経路の木構造表現

5.4.3　引き剥し線略

本方式では部分的に引き剥すネットを経路の線分単位とし．引き剥す経路の選択は，各部分経路の線分経路ごとに評価されたペナルティ値を用いたルーレット戦略[48]により，ペナルティ値に比例した確率で選択される．ルーレット戦略による選択は一度の引き剥し再配線処理の中で一回だけ行われる．このとき，引き剥した線分経路の前後では冗長な経路となる場合があるが，最初に選択された線分経路に引き続いて再帰的に検査しながらこのような冗長な経路を引き剥す．ここで再帰的な検査により選択される線分経路の選択範囲について説明する．まず選択された経路の端が端子であれば選択は終了する．経路の端が仮想端子の場合には仮想端子に接続される他の端子の状態によって選択が左右される．これを**図5-6**を用いて説明すると，図(a)に示すように仮想端子がベンド点である場合は全て選択され取り除かれる．図(b)，及び図(c)は共に経路の分岐点における選択であり，それぞれ，仮想端子（ベンド点）が残る場合と取り除かれる場合である．

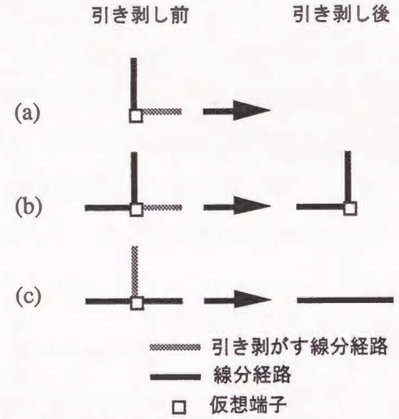


図5-6　仮想端子周りの引き剥しの組み合わせの例

引き剥しの例を**図5-7**の単層配線問題を用いて説明する．図(a)に示すように線分経路aが引き剥しの対象として選択されるとすると，線分経路aとその両端に接続される別の線分経路との構造を調べ，引き剥しの対象になるかどうかを調べる．端子T1は仮想端子であるが，**図5-6**より仮想端子T1は引き剥しの対象になる．端子T2は実端子であるためこれ以上引き剥せず，線分経路dは引き剥しの対象にならない．図(b)は仮想端子T1の引き剥しにより次の検査対象となる線分経路b，cについて検査する場合で，同様に再帰的な検査と選択による引き剥しの結果，図(c)に示す結果が得られる．

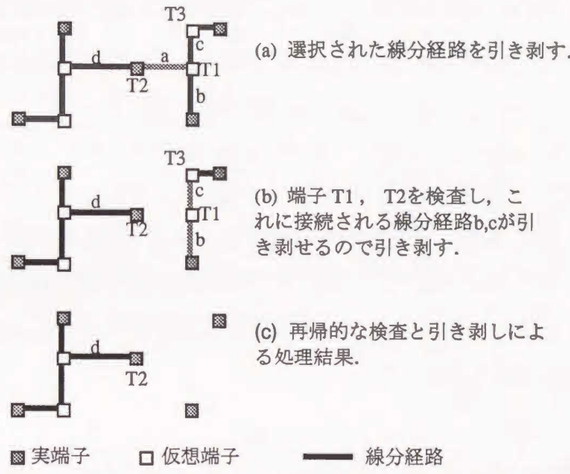


図5-7　簡単な配線問題による引き剥し例

5.4.4　再配線戦略

選択された線分経路は配線領域から引き剥され，該当する多端子ネットは複数の部分経路（部分木）に分割される．ここでは木構造表現を用いた部分経路間の再配線戦略について述べる．分割された部分木の統合にはその部分木間の経路を探索し，部分経路を再構成することで新たなネットを生成する．この部分経路間を結ぶ2つの部分経路間で最適な経路を求めるには，片方の部分木全体を探索のスタートとし，他方の部分木全体をゴールとする方法[3]が考えられるが，探索範囲が拡がりすぎて計算量が増加する．そこで計算量を削減するため次のように近似化する方法を採用した．すなわち，配線経路の分割により対応する木構造表現も部分木に分割されるが，それぞれ異なる部分木に属する部分経路同士（木構造の枝同士）のマンハッタン距離が最小になる部分経路の組み合せを選択し，規模が小さい部分経路をスタート，規模の大きい部分経路全体をゴールとする探索を行うことで探索範囲の拡大することを防ぐことにする．

5.5 実装評価

本章で提案する部分引き剥し再配線による経路探索手法を逐次アルゴリズムをWSに実装し計算時間と配線品質を評価した．以下では実装と用いた配線問題，実験結果及びその評価についてそれぞれ述べる．

5.5.1 実装

提案手法の逐次アルゴリズムをWSに実装し評価した．WSにはSun SPARC Station 2(27MIPS)を使用した．適用した配線問題は64×64グリッドの２層のものであり，配線規則としてXYルールを適用した．また，本方式の部分引き剥し再配線法（以下部分引き剥し法）と比較するため，ネット全体を引き剥がす全引き剥し再配線法（以下全引き剥し法）も同様に実装した．

5.5.2 配線問題

使用した多端子ネットデータは，いずれもランダムな位置に生成した端子間をランダムに選択して生成したものである．このネットデータの諸元を**表5-1**に示す．

表5-1 ネットデータの諸元

データ名	端子数	ネット数	平均端子数
Data-1	170	51	3.33
Data-2	128	38	3.37
Data-3	191	41	4.66
Data-4	160	41	3.90
Data-5	179	29	6.17

5.5.3 実験・評価

実験結果を**表5-2**，**表5-3**に各々示す．この結果は各ネットに対して行った１０回の試行の平均値である．**表5-2**の平均端子数はネット１本当たり端子数を表し，平均探索回数は引き剥し再配線におけるネット当たりの平均探索回数，平均探索時間と平均探索面積はそれぞれ一回の部分経路探索における探索時間，探索面積の平均である．**表5-3**の各項目は全引き剥し法に対する部分引き剥し法の比である．以下ではこの実験から得られた引き剥し再配線による経路改善の効果について述べる．

表5-2 逐次処理による実験結果

		全引き剥し再配線法			部分引き剥し再配線法		
データ名	平均端子数	平均探索回数	平均探索時間	平均探索面積	平均探索回数	平均探索時間	平均探索面積
Data-1	3.33	2.37	1.48	31240.52	1.67	1.85	47373.72
Data-2	3.37	2.54	1.18	30740.64	1.71	1.90	44938.55
Data-3	4.66	3.74	1.08	22308.30	1.78	1.90	45694.74
Data-4	3.90	2.96	1.00	20156.77	1.44	1.65	42053.38

表5-3 部分引き剥し法と全引き剥し法の項目比

データ名	平均端子数	探索回数比	探索時間比	探索面積比	配線長比	ビア数比	ベンド数比	探索回数比×探索時間比
Data-1	3.33	0.70	1.26	1.52	98.49%	95.57%	96.70%	0.94
Data-2	3.37	0.67	1.60	1.46	99.26%	97.32%	94.00%	0.97
Data-3	4.66	0.48	1.76	2.05	100.62%	102.25%	97.09%	1.03
Data-4	3.90	0.49	1.66	2.09	100.82%	100.79%	96.97%	1.02
Data-5	6.17	0.49	1.83	1.35	100.56%	105.59%	96.07%	1.06

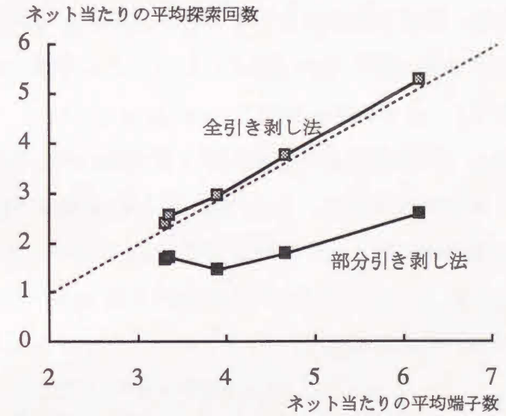


図5-8 ネット当たりの平均端子数と平均探索回数の関係

表5-2の平均探索回数から，部分引き剥し法は全引き剥し法と比較して約半分の探索回数となっていることがわかる．**図5-8**はネット当たりの平均端子数を横軸に，平均探索回数を縦軸にグラフ化したものである．全引き剥し法では平均端子数の増加に比例して平均探索回数が増加しており，平均探索回数≒平均端子数－１の関係となっている．これは，多端子ネットをMSTを用いて２端子ネットに分割し，配線処理した場合の必要探索回数（グラフ中に点線で示した）とほぼ同じである．一方，部分引き剥し法では平均端子数が小さい場合は部分引き剥し再配線による探索回数削減効果は小さく，平均端子数が増加するとその効果は大きくなるが，約半分の比率で飽和している．この理由として次のことが挙げられる．すなわち，平均端子数が少ない場合にはネットを構成する部分経路が少なく，引き剥しによりネットの殆

どの部分経路が引き剥され，全引き剥し法に近くなり，差が小さくなる．平均端子数が増加すると部分経路の平均数が増加し，部分引き剥しの効果が大きくなると考えられる．また，**表5-3**に示す探索回数比が約０.５より小さくならない理由は，経路探索回数がネット当たりの引き剥される部分経路数に依存し，ネット当たりの平均的な引き剥し数が部分経路数の半分程度であるものと考えられる．

次に配線時間であるが，平均探索時間では部分引き剥し法が全引き剥し法よりも長いが，処理時間全体では部分引き剥し法が全引き剥し法よりも短いことがわかる．これは，部分引き剥し法の方が少ない引き剥がし回数で収束することを意味する．探索時間の増加原因は部分経路探索における探索面積が増加したためと考えられる．今回用いた経路探索法は迷路法を基本としているため探索面積が増加すると経路探索時間が著しく増加するが，**表5-3**の探索面積比を見ると，部分引き剥し法の方がより広範囲の探索を行うことが確認される．

最後に配線品質では，配線長はどのネットデータにおいても差は小さく，配線長が長くなった場合でも１%未満である．ビア数は平均端子数の増加と相関して増加しているが，ベンド数が全体的３－６%減少している．これらのことから，部分引き剥し法では全引き剥し法に比べてビア数が増加する傾向があるが，配線長には殆ど差がなく配線長への影響は少ない．

5.5.4　考察

実験結果から得られた事実から，経路探索時間と配線品質について考察する．

・経路探索時間

全引き剥し法と比較して探索回数の削減と全体の処理回数の削減効果が得られた．しかし，部分引き剥し法では部分経路探索における探索面積が増加し，**表5-3**の探索時間比×探索回数比は殆改善されていない．探索時間の増加は探索面積の増加であり，この原因には以下のことが考えられる．

（1）本方式では経路探索に配線コストを用いた迷路法を使用しているが，計算量削減のため，発見されたゴールの配線コストを用いて探索の枝刈を行っており，配線コストの低い経路が存在する場合は探索面積は狭い．しかし，再配線は他の配線経路による交差や接触が主な原因であり，配線領域は密な状態であると考えられるため，配線コストが高くなり探索面積が拡大する．

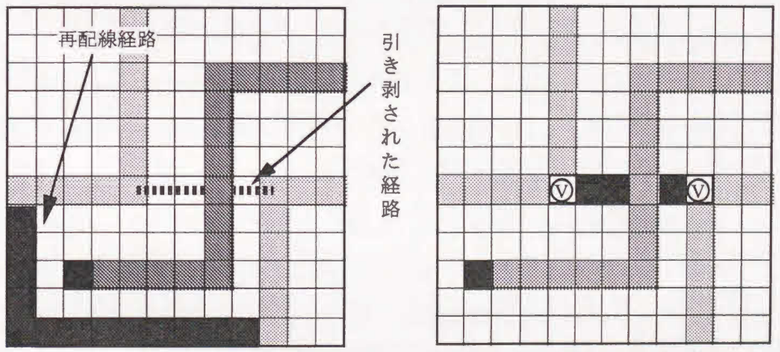
（2）再配線時の探索点は探索を行う部分木間の最近傍点のうち規模の小さい方の部分木を探索開始点とするため，探索開始点が引き剥された部分経路の端と同じになることがある．このため，交差・接触を原因とする引き剥しはその引き剥された部分経路上に他のネットの部分経路が存在するため，同じ点からの最近傍点までの探索コストが上昇し，広範囲の探索を必要とする．

（3）部分引き剥し法では全引き剥し法と比べて引き剥し回数が少なく，引き剥される部分経路も交差・接触数の多いものから高い確率で引き剥されるため，再配線時の経路探索において（1）で述べた理由により探索面積が拡大する．

・配線品質

実験結果では配線長は殆ど変化していないが，ビア数が増加し配線経路のベンド数が減少している．この原因は次のように考えられる．

配線経路が交差する別の経路を同じ配線層で迂回するよりもビアを通して迂回する方が再配線のコストが小さい．このことはビアに対する配線コストの設定に関連があると推測される．つまり**図5-9**に示すように，同層での迂回コストがビアによる迂回コストよりも小さい場合に**図**(a)に示すように迂回経路が生成されるため，配線長とベンド数が増加する．この逆の場合には**図**(b)のように迂回経路が生成されるため，配線長は変化せずビア数が増加することになる．



(a)迂回コストが小さい場合.
（ベンド数と配線長が増加）

(b)ビアによる迂回コストが小さい場合.
（ビアにより，配線長は変化せず）

図5-9　迂回経路の違い

5.6 並列処理への適合性

部分引き剥し再配線を用いた多端子ネットの配線処理方法についての実験結果をもとに並列処理への適合性について考察する.

5.6.1 計算粒度

本方式では多端子ネットを単一のプロセッサに割り当てた場合よりも計算粒度を細かくし,プロセッサに動的に処理割り当てを可能にするので,プロセッサ利用率の向上を図ることができる.一方,評価結果からは引き剥し再配線処理の経路改善処理の平均時間比(探索回数比×探索時間比:表5-3参照)が逐次処理の場合と同等であることがわかった.しかし部分引き剥し法では経路探索単位でプロセッサへの処理割り当てを自由に変更することができ,ネット単位で割り当てる全引き剥し法と比較すると計算粒度が細かい.このためネット当たりの探索回数が減り,更に多くのネットを並列処理することができるため,プロセッサ利用率の向上が期待できる.このことから本手法の計算粒度は適当であると言える.

5.6.2 通信量

部分引き剥し法では部分経路探索単位でプロセッサに処理を割り当てることから,割り当ての度に配線情報の受渡しが必要なため,通信量が増加し並列性能の低下を招く恐れがある.しかし第4章の実験結果より,配線領域の全体コピー方式を用いているにも拘らずマスタにおけるボトルネックがほとんどないことがわかる.従って通信量が増加した場合でもデータベースの参照方式に部分コピー方式を用いることにより性能低下を防げるものと判断される.一方,全引き剥し法では割り当てられたネットの探索終了まで通信を行う必要がないので通信量が少ない利点があるが,データベースの更新頻度が低下するため,配線結果とデータベースとの矛盾の増加により,引き剥し再配線処理回数が増加し,全体の収束性を低下させる.仮に通信回数を増やし,データベース更新間隔を短くして矛盾の発生を抑えたとしても,プロセッサに割り当てられる処理単位が部分引き剥し法よりも大きいため,処理量よりもプロセッサ数が多い場合や配線処理の終盤で残りの処理量が少なくなった場合にプロセッサ利用率が大幅に低下することになる.

以上のことから,本手法による通信量増加は,データベースの参照方式の改善な

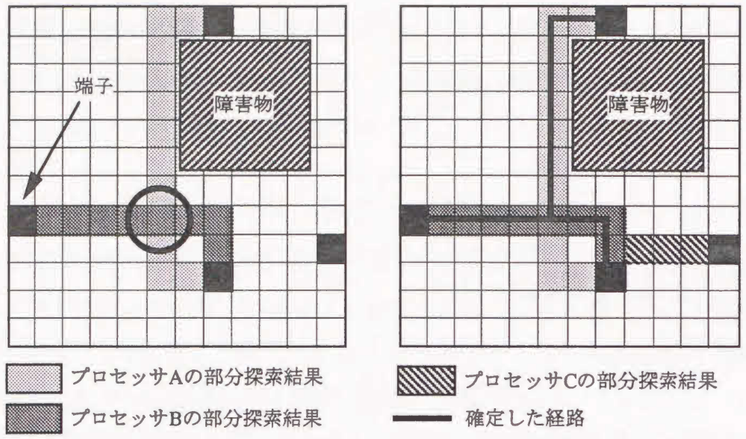
どにより並列性に対する影響は小さくでき,全引き剥し法よりも有効であると考えることができる.

5.6.3 処理割り当て戦略

多端子ネットデータの部分経路探索は一般的には逐次的に行われるが,特定の場合に並列処理をすることができる.この様な場合には並列処理可能な部分経路探索を複数のスレーブに割り当てることにより,プロセッサ競合方式で処理されるネット間の並列性に加えてネット内の並列性による2段階の並列処理ができる[30-32][34,35](処理割り当てを決定するための並列性の抽出戦略は次の5.7節で述べる).この場合,並列経路改善法の課題である改善可能な経路数が少ない場合のプロセッサ利用率の低下を避けることができる.2段階の並列性の利用の割合を変化させることでプロセッサ競合の割合を変化させることができ,競合による多様度の変化を用いた局所最適解の状態からの脱出を助ける効果が期待できる.

5.6.4 部分経路探索間の矛盾解消

並列処理による部分経路探索結果の間に矛盾が生じた場合の解消方法について考える.単純にマスタが矛盾を解消するとすれば,この為の負荷が新たに増加する.プロセッサ競合方式ではマスタの負荷が軽い方が並列性能が向上するため,この方法は不適当である.そこで別のスレーブに配線結果を評価させて矛盾解消するか,部分経路探索を継続するスレーブが矛盾解消する方法が考えられる.同ネット内の矛盾解消にはデータベースを参照する必要はなく,継続した配線情報と確定された部分経路を用いて矛盾は解消されるため,マスタに対して新たな負荷をかけない利点がある.この方法によりスレーブの処理手順は複雑になるものの,並列性能の低下をさせずに矛盾解消ができる.図5-10に示す矛盾解消の例では,プロセッサA,Bがそれぞれ並列に部分経路探索を行い,プロセッサCが経路探索を継続する.図(a)は部分経路が矛盾(交差)した状態であり,図(b)はプロセッサCに探索が継続された場合で,プロセッサCは部分探索結果の矛盾の検出し,その解消と探索の継続を行うものである.



(a)部分探索結果間の矛盾の発生　(b)別のスレーブによる矛盾解消と経路探索の継続

図5-10　部分探索結果間の矛盾の解消

5.6.5　並列処理の適合性に対する結論

通信に関する幾つかの課題があるものの，第４章で述べた並列経路改善方式と同程度の計算粒度であり，処理の動的割り当てにより，並列経路改善方式の課題である利用率が改善されるため，全体では同程度の性能が期待できる．そして多端子ネットの配線により配線品質が改善されることから，本方式を並列経路改善方式に実装した場合，多端子配線問題に対して有効に働くものと考えられる．

5.7　並列性抽出戦略

本手法により並列処理を行う場合の並列性の抽出戦略について述べる．

5.7.1　部分経路探索の分割

部分経路探索には一般に逐次性があるが，状況によっては並列に探索できる場合がある（ネット内の並列性）．図5-11の例を用いて説明する．(a)まず探索開始点を選び部分経路探索を実行する．この探索で得られた仮経路は配線領域の左半分であり，右半分の端子との関連性は低い．このため(b)のように右側の部分経路探索を別のプロセッサで並列に行うことができる．(c)はプロセッサAの処理が終了した状態で，(d)は全ての処理が終った状態である．この様な組み合せを効率良く抽出する方法として，複数の端子をグループ化し，グループ内部では逐次，独立するグループ間では並列，そして従属関係にあるグループ間では逐次に処理する方法がある[18].

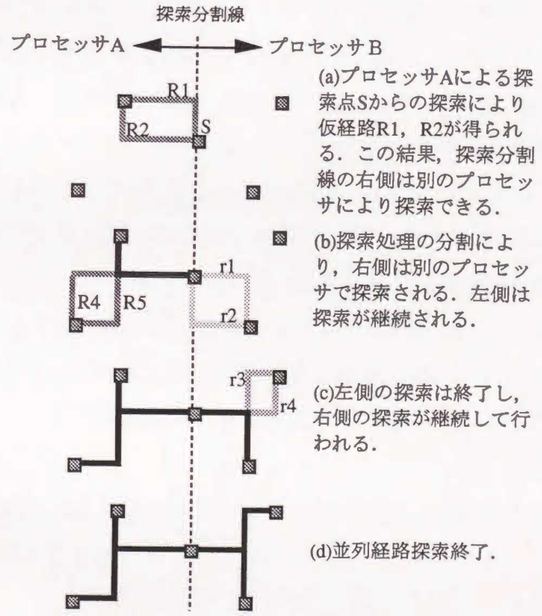


図5-11　経路探索の分割の様子

5.7.2　端子のグループ化

端子のグループ化には２つの方法があり，一つは配線を行う前にグループ化する方法（静的抽出），他は部分経路探索の実行中に動的にグループ化する方法（動的抽出）である．両方法の特徴について以下に述べる．

静的抽出法は著者らが1993年に提案[18]したが，この方法では他の配線経路が考慮されていないことが欠点であった．しかし本手法では引き剥し再配線処理を前提とするため，前回探索した経路情報を用いてグループ化することにより他の配線経路を考慮したグループ化が可能になる．探索前にグループ化するため，ネット全体を配線する場合に後述の動的抽出方法よりも高い並列性が得られることがある．

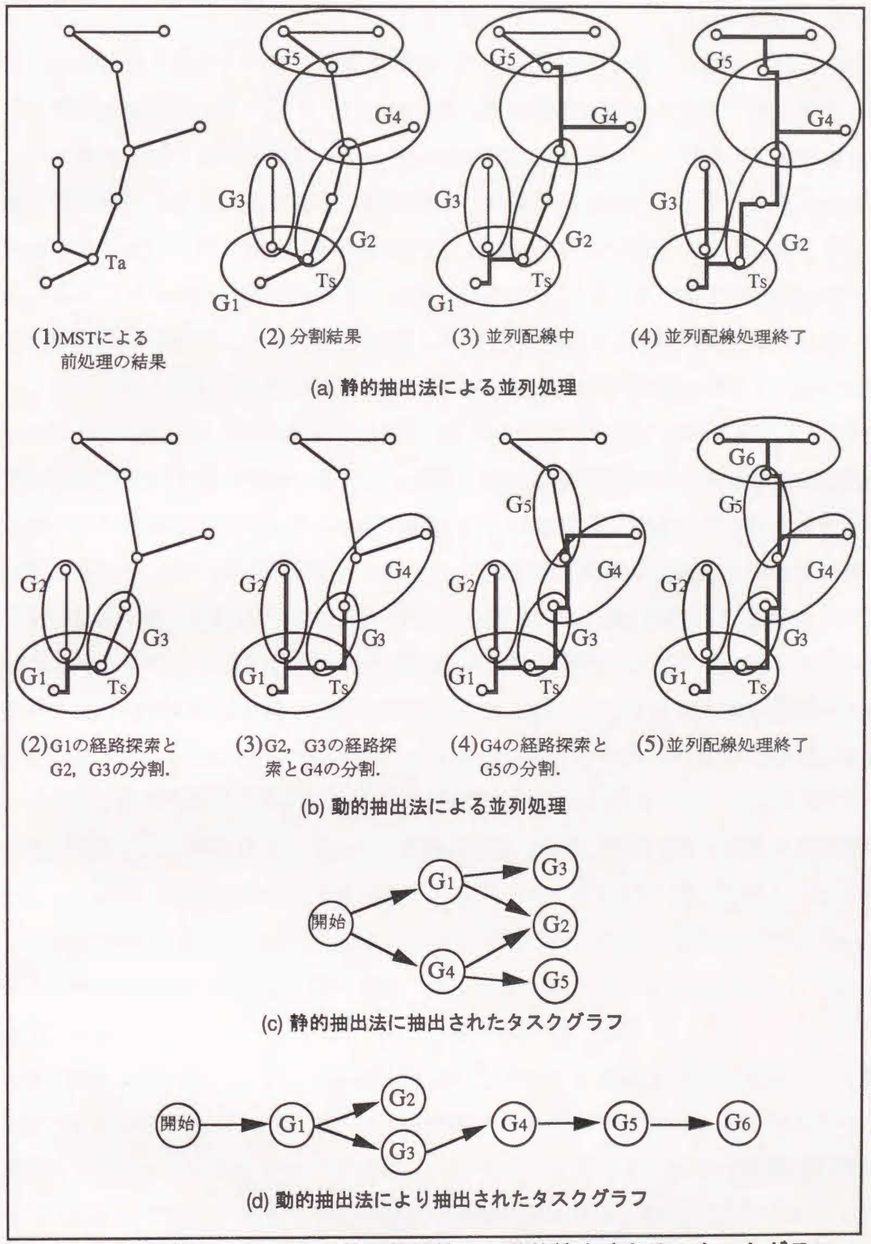
動的抽出法は各部分経路探索の後に並列性が抽出されるため，他の配線経路を反映した抽出ができる．静的抽出法よりも高い並列性[(1)]が得られる可能性は低いものの，部分的に引き剥された経路の再配線では逐次性が高く，また処理するネット数が多い場合には複数のネットが並列処理されるため，問題にはならない．

図5-12に両方法のグループ化の例を示す．図(a)は静的抽出の場合で段階に並列性が抽出される例である．(1)端子T_sを分割開始点として分割した結果，(2)のようにグループ化される．隣接するグループ同士では並列処理することが難しいため，離れたグループ同士で並列に処理行うとすれば，そのタスクグラフは図(c)となる．(3)これに従ってまずグループG1とG4を並列に配線する．(4)残りのグループG2，G3，G5を並列に配線する．次に動的抽出の場合を図(b)に示す．(2)端子T_sを探索と分割の開始点として部分経路探索を行う．その結果グループG2とG3が並列処理できることを発見し，これらの処理を別のプロセッサに割り当てる．(3)グループG2とG3は並列に配線され，グループG3の探索後，新たにグループG4を発見する．(4)，(5)同様にしてグループG4，G5，G6を逐次に配線処理する．この場合のタスクグラフは図(d)となる．

この例からわかるように，静的抽出は全体の計算量の見通しができるため初期配線に効果的であり．これに対して動的抽出では先に述べたように部分引き剥がし再配線に効果的である．

---

(1)これはネット内の並列性のことであり，ネット間の並列性は両方法とも高い．



(1)MSTによる前処理の結果　(2) 分割結果　(3) 並列配線中　(4) 並列配線処理終了

(a) 静的抽出法による並列処理

(2)G1の経路探索とG2，G3の分割.　(3)G2，G3の経路探索とG4の分割.　(4)G4の経路探索とG5の分割.　(5)並列配線処理終了

(b) 動的抽出法による並列処理

(c) 静的抽出法に抽出されたタスクグラフ

(d) 動的抽出法により抽出されたタスクグラフ

図5-12　ネット内の並列性の静的抽出と動的抽出法とそのタスクグラフ

5.8　まとめ

本章では，マスタ・スレーブ方式のプロセッサ競合モデル上で引き剥し再配線により経路改善を反復する並列経路改善方式において，多端子ネットを並列処理するための問題点を検討し，配線経路を部分的に引き剥し再配線する方法を提案した．これは，多端子ネットの経路探索を部分経路探索に分割することにより計算粒度をネット全体を引き剥す方法よりも細かくし，かつ部分引き剥し再配線により経路探索回数を削減することを目的としたものである．また多端子ネットの経路探索において部分経路探索の継続に必要な配線情報の削減を図るため，仮経路を用いた経路探索法と，部分配線経路の管理を容易にするための木構造表現を提案した．

提案した経路探索法を評価するためにWS上で行った逐次処理による実験結果では，探索回数は削減されたものの探索時間が増加しており，探索時間比×探索回数比では全引き剥がし法と大きな差は無い．しかし，本方法の計算粒度は全引き剥し法よりも細かくなり，部分引き剥しによるネット当たりの探索回数が削減されるので，より多くのネットの並列処理が可能になり，プロセッサ利用率を向上できる利点を持つ．このため高並列度の配線処理方式として有効な方法であると結論される．また，２段階の並列性を有効に活用するための処理割り当てと並列性抽出方法の戦略について考察した．

今後の課題は，この提案方法を並列計算機に実装して大規模な配線問題に適用し，その効果を実証することである．

第６章

結論

本論文では自動配線問題の並列処理方式に関する研究として，マスタ・スレーブモデルを基本計算モデルに用いた並列配線処理方式について述べてきた．これは，並列計算機のアーキテクチャに対する依存性を抑えて実装性を高め，かつ高い並列性の実現を目指したものである．この研究の背景には，配線問題は計算量の多さからアルゴリズム面での高速化および並列処理による高速化の試みが従来より行われてきたが，今後ますます電子回路の小型化と高速化により配線問題の複雑度は増加する一方であるため，従来の逐次アルゴリズムとは異なる並列処理方式が必要とされていたことがある．

これに関して第２章では，基本的な配線問題と代表的な逐次アルゴリズムによる解法およびその問題点について述べるとともに，逐次アルゴリズムをそのまま並列化した並列処理方式の特徴と問題点について考察した．更に，現在研究されている並列配線処理方式について，PROTON，タイムワープ方式，及びRPに関してその処理方式を述べた．

第３章ではマスタ・スレーブモデルを用いたプロセッサ競合方式を提案し，分散メモリ型と共有メモリ型の異なるアーキテクチャの並列計算機による評価について述べた．プロセッサ競合方式は計算機アーキテクチャに対する依存性を抑えた高並列な処理方式であり，計算モデルが単純で理解し易いため，配線問題だけでなく他の並列処理への応用ができる特徴がある．

２つの計算機アーキテクチャによる評価の結果，共に有効性が確認されたが，共有メモリ型の方がより高い並列性を発揮した．しかし共有メモリ型は分散メモリ型

よりもプロセッサ数のスケーラビリティの点で劣り，大規模な並列処理の実現が困難であることから，今後要求される超並列処理には分散メモリ型の方が有効であることについて述べた．

第４章では，プロセッサ競合方式の改善課題である配線品質の向上のため，配線経路を引き剥して再配線する並列経路改善方式を提案した．この方式はプロセッサ競合方式を基本処理モデルとして，複数の既配線経路を同時に引き剥し再配線処理し，複数の経路改善を並列処理するものである．また，配線コストを用いた迷路法と配線経路に対するペナルティの採用により，配線経路の交差・接触を許容した配線順序の影響を抑えた経路探索法を述べた．そして分散メモリ型並列計算機Coral-68Kによる評価実験により，プロセッサ数の増加に従った配線品質の改善効果が確認され，その理由ついて考察し，並列経路改善方式の有効性を述べた．

第４章で述べた並列経路改善方式では配線問題として２端子ネットを用いており，多端子ネットの配線処理は２端子ネットに問題変換して行う必要がある．しかし，この方法は配線品質向上の障害となるため，第５章では多端子ネットを並列配線処理するための経路探索法を提案した．これは多端子ネットの経路探索が複数の部分経路探索から構成されることに注目し，部分経路単位の探索によりプロセッサへの動的処理割り当てとプロセッサ利用率の向上を可能にし，配線経路を部分的に引き剥す方法により経路改善における再探索回数を削減するものである．WSによる逐次プログラムの評価により，経路改善における探索回数の削減と配線結果が収束するまでの経路改善回数の削減効果を確認した．また，並列処理への適合性について考察した結果，この方法は並列処理においても有効であり，プロセッサ競合方式で用いるネット間の並列性とネット内の並列性による２段階の並列処理が可能であることが確認され，その処理割り当てと並列性抽出方針について述べた．

今後は大規模な配線問題の並列処理に関して，配線品質を向上しつつ逐次配線処理方式よりも高速な配線処理を可能にする方式を研究し，超並列処理による自動配線処理方式について取り組みたいと思う．

謝辞

本研究の全過程を通じ，直接御指導，御鞭撻を頂いた徳島大学工学部知能情報工学科高橋義造教授に深く感謝致します．

本研究にあたり，御指導頂いた徳島大学工学部知能情報工学科赤松則男教授，矢野米雄教授に感謝致します．

本研究において共有メモリ型並列計算機TOP-1の使用に当たってお世話になった日本IBM東京基礎研究所の鈴木則久所長をはじめ，清水茂則氏，穂積元一氏，大庭信之氏，中谷登志男氏，山内長承氏，中田武男氏の方々に感謝いたします．また，実データの提供と商用ルータによる処理を実行して頂いたイビデン（株）の足立和正氏，清水賢治氏に深く感謝致します．

学会発表の際に御懇切な討論を頂いた，NECの中田登志之氏，山内宗氏，ICOTの瀧和男氏（現在 神戸大学），松本幸則氏（現在 三洋電気），九州大学安浦寛人教授，その他の方々に深く感謝致します．

本研究を通じて，御指導頂いた徳島大学工学部知能情報工学科青江順一教授，仁木登助教授に感謝致します．また，熱心な御討論を頂いた徳島大学工学部知能情報工学科のジュリオ・タノマル助手，井上富夫技官，同科高橋研究室の大学院生岡圭司氏をはじめとする研究室の方々に深く感謝致します．

参 考 文 献

[1] E. F. Moore, "The Shortest Path through a Maze," *Annals of the Harvard Computation Laboratory*, vol. 30, Pt. II, pp.185-292, 1959.

[2] C. Y. Lee, "An Algorithm for Path Connections and its Application," *IRE Trans. on Electronic Computers*, vol. EC-10, pp. 346-365, 1961.

[3] T. Otsuki, :LAYOUT DESIGN AND VERIFICATION, *Elsevier Science Publishers B. V. (North Holland)*, Ch. 3 "Maze-running and Line-search Algorithms," pp. 99-131, 1986.

[4] K. Mikami and K. Tabuchi,"A Computer Program for Optimal Routing of Printed Circuit Connectors," *IFIPS Proc.*, vol. H47, pp. 1475-1478, 1968.

[5] D. W. Hightower, "A Solution to Line-Routing Problem on the Continuous Plane," *Proc. 6th Design Automation Workshop*, pp. 1-24, 1969.

[6] M. A. Breuer and K. Shamsa, " A Hardware Router," *J. Digital Syst.*, no. 4, pp. 393-408, 1981.

[7] T. Watanabe, H. Kitazawa, Y. Sugiyama,"A Parallel Adaptable Routing Algorithm and its Implementation on a Two-Dimensional Array Processor," *IEEE Trans. on Comput er Aided Design*, vol. 6, no. 2, 1987.

[8] K. Kawamura, M. Ishii, H. Shiraishi, "MASSIVELY PARALLEL ROUTING SYSTEM", *Denshi Tokyo*, no. 31, pp. 90-95, 1992.

[9] Y. Takahashi. and M. Sano, "Parallel computing with a number of competing processors," *Pallalel Computing '91*, D. J. Evans et al, Eds. Amsterdom: Elsevier Science Publishers B.V. (North Holland), pp. 339-346, 1991.

[10] 佐野，高橋：プロセッサ競合方式による並列配線処理，情報処理学会第４３回全国大会，分冊６，pp. 243-244 (1991).

[11] 佐野，高橋：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比較，並列処理シンポジウムJSPP'91論文集，pp. 197-204 (1991).

[12] 佐野，高橋：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).

[13] 佐野，岡，高橋：プロセッサ競合方式による並列自動配線～ランダム引き剥し法～，並列／分散／協調処理に関するサマーワークショップSWoPP'92，数値解析-42，pp. 33-40 (1992).

[14] 佐野，高橋：プロセッサ競合方式による並列配線処理～引き剥し処理による品質改善～，並列処理シンポジウムJSPP'93論文集，pp. 331-338 (1993).

[15] 佐野，高橋：プロセッサ競合方式による並列自動配線～品質改善の試み～，DAシンポジウム論文集，pp. 181-184 (1993).

[16] 佐野，高橋：重み拡散を用いた並列配線処理，情報処理学会第４７回全国大会，分冊６，pp. 147-148 (1993).

[17] 佐野，高橋：並列配線問題における並列引き剥し再配線処理の品質改善効果，情報処理学会論文誌，Vol. 36, No. 2, (1995) (印刷中）.

[18] 佐野，高橋：Multi-pin-netの引き剥しを考慮した並列配線処理，SWoPP'93，アルゴリズム-34，pp. 17-34 (1993).

[19] 佐野，高橋：並列化を考慮した多端子配線問題の部分引き剥し再配線処理，DAシンポジウム論文集，pp. 229-234 (1994).

[20] 佐野，高橋：部分引き剥し再配線法による並列処理のための多端子ネットの経路探索法，信学論，（投稿中）.

[21] A. Hashimoto, J. Stevens, "Wire Routing by Optimizing Channel Assignment within Large Apertures," *Proc. 8th Design Automation Workshop*, pp. 155-169, 1971.

[22] M. Burstein, "CHANNEL ROUTINGLAYOUT," DESIGN AND VERIFICATION Elsevier Science Publishers B. V. (North Holland), Ch. 4, pp. 133-167, 1986.

[23] F. O. Hadlock, "A Shortest Path Algorithm for Grid Graphs," *Networks*, vol. 7, pp. 323-334, 1977.

[24] J. Soukup, "Fast Maze Router," *Proc. 15th Design Automation Conf.*, pp. 100-102, 1978.

[25] W. Heyns, W. Sansen and H.Bake, "A Line-Expansion Algorithm for the General Routing Problem with a Guaranteed Solution," *Proc. 17th Design Automation Conf.*, pp. 243-249, 1980.

[26] K. Suzuki, T. Otsuki and M. Sato, "A Gridless Router: Software and Hardware Implementations," *VLSI '87*, pp. 121-131, 1987.

[27] Young-Long Lin, Yu-Chin Hsu and Fur-Shing Tsai,"Hybrid Routing," *IEEE Trans. on CAD*, vol. 9, no. 2, pp. 151-157, 1990.

[28] C. Chang and M. Sarrafzadeh, "Global Routing Based on Steiner Min-Max Tress," *IEEE Trans. on CAD*, vol. 9, no. 12, pp. 1318-1325, 1990.

[29] 程，田中：局所電流比較法による並列的な自動配線，信学論，Vol. J75-A，No. 9，pp.1476-1486(1992).

[30]伊達，大嶽，瀧：並列オブジェクトに基づくLSI配線プログラム，情報処理学会論文誌，Vol.33，No.3，pp. 378-386 (1992).

[31] Y. Takahashi, T. Suzue and S. Kashimoto, "Parallel Maze-Running and Line Search Algorithm for LSI CAD on Binary-Tree Multiprocessor," *Proc. World Conf. on Information Processing/Communication*, Seoul, pp. 128-136 (1989).

[32] 山内，中田 石塚, 西口, 小池：MIMD 型並列計算機上のLSIルーター -PROTON-, 情報処理学会論文誌, Vol.34, No.4, pp. 699-707 (1993).

[33] T. Nakata et al.,"Cenju: A Multiprocessor System for Modular Circuit Simulation," *Computing Systems in Engineering*, vol. 1, no. 1, pp. 101-109, 1990.

[34] 松本，瀧：タイムワープによる並列無格子配線システム，並列処理シンポジウム'９３，pp. 323-329 (1993).

[35] 松本，瀧：タイムワープ機構の新しい応用　ー並列無格子配線ー，情報処理学会論文誌，Vol.35，No.4，pp. 666-676 (1992).

[36] A. Marugarino et al., "A Tile-Expansion Router," *IEEE Trans. on CAD*, vol. 6, no. 4, pp. 507-517, 1987.

[37] H. Nakashima et al., "Archtecture and Implementation of PIM/m," *Proc. Int. Symp. on Fifth Generation Comp. Sys.*, pp.425-435, 1992.

[38] Kawamura, K. et al.: Touch and Cross Router, *Proc. Int. Conf. Computer-Aided Design*, pp. 56-59, 1990.

[39] Y. Takahashi and S. Sasaki," Parallel Automated Wire Routing With a Number of Competing Processors," *Proc. ACM International Conf. on Supercomputing*, pp.310-317 (1990).

[40] 高橋，遠藤，松尾，槌谷：二進木結合並列計算機Coral68Kの開発とその評価，情報処理学会論文誌，Vol.30，No.1，pp.46-57(1989).

[41] K. Taki, "The Parallel Software Research and Development Tool: Multi-PSI System," *Programming of Future Generatioj Computers*, pp.411-426, North-Holland, 1988.

[42] K. Ueda and T. Chikayama,"Design of Kernel Languae for the Parallel Inference Machine," *Comput. J.*, vol. 33, no. 6, pp. 494-500, 1990.

[43] 大村，若林，宮尾，吉田：VLSIレイアウト設計における概略配線を同時に決定する階層化詳細フロアプランニング手法，信学論，Vol. J74-A，No. 6(1991)

[44] E. Rosenberg, "A New Iterative Supply/Demand Router with Rip-up Cacability for Printed Circuit Boards," *Proc. 24th Designe Automation Conf.*, pp.721-726(1987)

[45] M. Raith et al., "A New Hypergraph Based Rip-Up and Reroute Strategy," *Proc. 28th ACM/IEEE Design Automation Conf.*, pp.54-59, 1991.

[46] 清水，大庭，森脇，中田，小原：高性能マルチプロセッサ・ワークステーションTOP-1，並列シンポジウム JSPP'89, pp. 155-162 (1990).

[47] 木村，白石，姫野，花木，草桶：迷路法配線アルゴリズムのグリッドフリー化手法，信学論，Vol. J74-A，No. 8(1991)

[48] D. E. Goldberg, "Genetic Algorithms in Search", Optimization and Machine Learning, Addison-Wesley Publishing Company, Inc., 1989.

[49] Ho J.-M., Vijayan G. and Wong C. K. ,"New Algorithms for the Rectilinear Steiner Tree Problem," *IEEE Trans. Computer Aided Design*, vol. 9, no. 2, pp.185-193, 1990.

[50] J. P. Cohoon , D. S. Richards and J. S. Salowe, "An Optimal Steiner Tree Algorithm for a Net Whose Terminals Lie on the Perimeter of a Rectangle", *IEEE Trans. Comput.er Aided Design*, vol. 9, no. 4, pp.398-407, 1990.

関 連 論 文

第３章に関する論文

・Y. Takahashi. and M. Sano, "Parallel computing with a number of competing processors, *Pallalel Computing '91*, D. J. Evans et al, Eds. Amsterdom: Elsevier Science Publishers B.V. (North Holland), pp. 339-346, 1991.

・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比較，並列処理シンポジウムJSPP'91論文集，pp. 197-204 (1991).

・佐野雅彦，高橋義造：プロセッサ競合方式による並列配線処理，情報処理学会第４３回全国大会，分冊６，pp. 243-244 (1991).

・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).

第４章に関する論文

・佐野雅彦，岡圭司，高橋義造：プロセッサ競合方式による並列自動配線～ランダム引き剥し法～，並列／分散／協調処理に関するサマーワークショップ SWoPP'92，数値解析-42，pp. 33-40 (1992).

・佐野雅彦，高橋義造：ランダム引き剥し法を用いた並列配線処理，情報処理学会第４５回全国大会，分冊６，pp. 151-152 (1992).

・佐野雅彦，高橋義造：プロセッサ競合方式による並列配線処理～引き剥し処理による品質改善～，並列処理シンポジウムJSPP'93論文集，pp. 331-338 (1993).

・佐野雅彦，高橋義造：プロセッサ競合方式による並列自動配線～品質改善の試み～，DAシンポジウム論文集，pp. 181-184 (1993).

・佐野雅彦，高橋義造：重み拡散を用いた並列配線処理，情報処理学会第４７回全国大会，分冊６，pp. 147-148 (1993).

・佐野雅彦，高橋義造：並列配線問題における並列引き剥し再配線処理の品質改善効果，情報処理学会論文誌，Vol. 36, No. 2, (1995) (印刷中)

第５章に関する論文

・佐野雅彦，高橋義造：Multi-pin-netの引き剥しを考慮した並列配線処理，SWoPP'93，アルゴリズム-34，pp. 17-34 (1993).

・佐野雅彦，高橋義造：並列化を考慮した多端子配線問題の部分引き剥し再配線処理，DAシンポジウム論文集，pp. 229-234 (1994).

・佐野雅彦，高橋義造：部分引き剥し再配線法による並列処理のための多端子ネットの経路探索法，信学論，（投稿中）.

配線問題の並列処理方式に関する研究

発　　行　1995年3月
著　　者　佐野雅彦
住　　所　〒779-31
徳島市国府町佐野塚字出口95-1
印刷・製本　株式会社 イシダ測機
徳島市中徳島町2-82
TEL (0886) 25－0720