計算機アーキテクチャ　103－2
設　計　自　動　化　69－2
(1993. 12. 16)

ニューラルネットワークに基づく並列自動配線アルゴリズム

鈴来 響太郎† 花田 彰† 天野 英晴† 武藤 佳恭‡

†慶應義塾大学 理工学部

神奈川県 横浜市 港北区 日吉 3-14-1

‡慶應義塾大学 環境情報学部

神奈川県 藤沢市 遠藤 5322

あらまし

現在までに提案されている並列自動配線アルゴリズムのほとんどは、従来からある迷路法、線分探索法を並列化したものである。このため、細粒度の並列化と高いプロセッサ利用率を同時に実現できず、並列計算機に実装した場合に高い台数効果を得ることが難しい。本研究では、この条件を満たせるようにニューラルネットワークに基づく並列自動配線アルゴリズムを提案し、シーケンシャルマシン上に実装してアルゴリズムの質の評価を行なう。また、並列計算機への実装の方法についても検討する。

和文キーワード　細密配線、並列計算、HVルール、ニューラルネットワーク

A parallel routing algorithm based on neural networks

Kyotaro Suzuki† Akira Hanada† Hideharu Amano† Yoshiyasu Takefuji‡

†Faculty of Science and Technology, Keio University

3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Kanagawa, 223 JAPAN

‡Faculty of Environmental Information Keio University

5322 Endoh, Fujisawa, Kanagawa, 252 JAPAN

Abstract

Since most of proposed parallel routing algorithms are parallelized algorithms of maze running or line search which were developed for sequential machines, efficient parallel processing with high processor utilization is difficult. Here, a parallel routing algorithm based on neural networks which can achieve both high degree of parallelism and utilization ratio is proposed, and the quality of the solution is presented on a sequential machine. The implementation on a parallel machine is also discussed.

英文 key words　detailed routing, parallel computing, HV rule, neural networks

1 はじめに

VLSI チップや PCB の配線レイアウトは、回路数の増加、配線の高密度化に伴い配線に要する時間が増大し、セル配置と共に設計全体のボトルネックとなっている。並列処理による自動配線の高速化はこの問題の解決に対する有力なアプローチの一つであり様々な手法が提案されている [1][2][3][4]。

しかし、自動配線の並列化手法のほとんどは、線分探索法、迷路法という逐次処理を前提として開発されたアルゴリズムを並列化したものであるため、以下のような問題を持っている。

- 線分探索法
  多くの場合、ネットを並列処理の最小単位としており、並列性が不足する場合がある。また、線分探索のフェーズでは探索線分単位で計算することができるが、線分を選択する際に逐次的な処理が必要となる。また、領域単位での並列化を行なう場合、各領域については逐次的に処理されるため、複数のネットを同時に処理しても十分なプロセッサ利用率を得ることができない。

- 迷路法
  格子点を単位として領域単位の並列処理が容易なため並列性が大きい反面、ほとんどの時間プロセッサはアイドルしている。また、複数ネットを同時に処理するためには、膨大なメモリ空間が必要になるため同時処理できるネット数が制約を受ける。

ニューラルネットワークによるアプローチは、最適解に収束する保証がない等の問題があるが、本質的に高い並列性を持っており、探索等では膨大な計算時間を必要とする問題に対して短時間に解を得ることが可能である。しかし、VLSI や PCB を対象とした実際の細密配線の分野では、障害物となるセルや他ネットの端子、via ホール、配線折れ曲がり等多くの現実的な制約を含み、定式化が困難であるためニューラルネットワークの応用例は少ない [5][6]。しかし、筆者らの一人の提案によるモデル [7] では、$2 \times n$ 層の配線層を用いたチャネル配線問題に対し、Hopfield 型ニューラルネットワークを用いて高い解の質を得ることに成功し、並列計算機への実装についても高い台数効果を得ることが出来た [8]。

本論文では、さらにこの方法を一般化し、チャネル配線以外に一般配線、スイッチボックス配線に対応することが可能なニューラルネットワークモデルを提案し、いくつかの例題についての適用例を示す。このモデルは、本質的にニューロン単位での並列処理が可能であり、逐次的な制限も少ないため、並列計算機上に実装した場合にその並列計算機の粒度に関係なく高いプロセッサ利用率と台数効果が期待できる。

以下の構成は次のようになっている。まず 2章ではこのアルゴリズムで扱う配線問題について説明する。次に 3章-4章で今回提案する並列アルゴリズムを説明する。5-7章では実際にいくつかの問題を解いてアルゴリズムを評価する。そして 8章では現在実装中である並列計算機への実装の方法について述べ、最後に 9章で今後の課題と結論を述べる。

2 配線問題

複数の同電位の端子グループとその間を接続する配線経路を合わせてネットと呼ぶ。3 端子以上の端子については、このアルゴリズムでは、2 端子をペアとしてその間の配線経路を求める。

配線は、配線層に打たれた正方格子点に沿って縦または横方向に行なわれる。縦方向の配線層である V 層と横方向の配線層である H 層の計 2 層の配線層が用いられる（HV ルール）。経路の回折は、格子点上で直角方向に行なわれる。配線層間の接続は、格子点上に via ホールをあけて行なう。各端子や障害物はどちらの配線層にも存在するものとする。従って、異なるネット同士の接触は、他ネットの端子に触れた場合と、同一配線層の上で平行方向に接触した場合に限られ、直角方向に交差している経路は接触したことにはならない（図 1）。

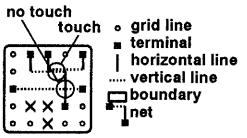

図 1: 配線問題

3 定義

まず、配線層の x 方向、y 方向の格子点の数をそれぞれ $size_x$、$size_y$ とする。配線層上の座標は図 2のように各方向とも 0 から始まる。

各ペアは、ネット識別子番号とネット内識別番号を持っている。これらの識別番号は 0 から始まるものとする。ネット識別番号 i を持つネットの中に含まれるネット内識別番号 j を持つペアを、ネット i のペア j と呼ぶ。

ネット数を $\#net$、ネット i に含まれるペア数を $\#pair_i$ とする。x 方向に配線された線分を x 線分、y 方向に配線された線分を y 線分と呼び、1 つのペアの x 線分の数を

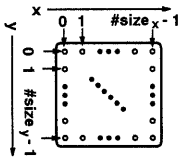

図 2: 座標

$\#line_x$、y 線分の数を$\#line_y$とする。配線は x 方向、y 方向への回折を交互に繰り返していくため、当然$\#line_x$と$\#line_y$の差は 1 以内である。

ネット i のペア j の配線経路について考える。端子の一方を配線経路の始点とし、他方を終点とする。この始点の座標を $(P_{i,j,0,x}, P_{i,j,0,y})$ と表す。

以下では特に断らない限り配線経路は始点からまず x 方向に進んでいくものとする。始点から x 方向に進んだ線分が到達した最初の回折点の x 座標を $P_{i,j,1,x}$と表す。すると、この配線経路の第 1x 線分は、$(P_{i,j,0,x}, P_{i,j,0,y}) - (P_{i,j,1,x}, P_{i,j,0,y})$ で表される。次に $(P_{i,j,1,x}, P_{i,j,0,y})$ から y 方向に進み次の回折点に到達する。同様にこの回折点の y 座標を $\Gamma_{i,j,1,y}$とすると、第 1y 線分は、$(P_{i,j,1,x}, P_{i,j,0,y}) - (P_{i,j,1,x}, P_{i,j,1,y})$ で表される。各線分の両端は、始点により近い方を始端、他方を終端とする。同様にして第 kx 線分は、$(P_{i,j,k-1,x}, P_{i,j,k-1,y}) - (P_{i,j,k,x}, P_{i,j,k-1,y})$、第 ky 線分は、$(P_{i,j,k,x}, P_{i,j,k-1,y}) - (P_{i,j,k,x}, P_{i,j,k,y})$ で表され、終点の座標は $(P_{i,j,\#line_x,x}, P_{i,j,\#line_y,y})$ で表される。

ネット i のペア j について$\#line_x = \#line_y = 2$ の場合の例を図 3に示す。

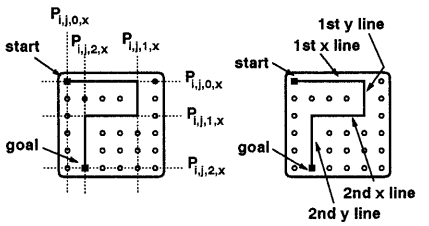

図 3: 配線成分の表現

4 アルゴリズム

4.1 ニューラルネットワークモデル

このアルゴリズムは、疑似的なニューラルネットワークを用いる。このネットワークは、ニューロン間が線形結合でないが、次のように Hopfield 型ニューラルネットワークと共通した部分を多く持っている。

- ネットワーク間は完全結合である。
- 各ニューロンは入力値 U と出力値 V を持っている。
- 出力値 V は 0 と 1 の 2 値を持っていて、1 の状態の時発火しているという。
- 入力値の更新に、エネルギー関数を全ニューロンの出力で微分した動作関数を用いる。

各ニューロンは U と V の更新を交互に繰り返していくことで動作する。

4.2 ニューロンによる配線経路の表現

ネット i のペア j の第 kl線分（l : x または y）の終端の l座標 $P_{i,j,k,l}$を決定するために使われるニューロンは、N_{ijklm}で表される。m は、l方向の座標に対応しているため、$0, 1, \ldots, size_l - 1$ の値をとる。

図 3のネットを例に説明する。始点、終点を表現するための、$P_{i,j,0,x}$,$P_{i,j,0,y}$,$P_{i,j,2,x}$,$P_{i,j,2,y}$は固定されているので、残りの $P_{i,j,1,x}$, $P_{i,j,1,y}$を求めればよい。$size_x = size_y = 6$ であるから、$P_{i,j,1,x}$を決定するために N_{ij1xm}, $m = 0, 1, \cdots, 5$ が必要になり、$P_{i,j,1,y}$を決定するために N_{ij1ym}, $m = 0, 1, \cdots, 5$ が必要となる。

ニューロン N_{ijklm}の入力値と出力値をそれぞれ U_{ijklm}, V_{ijklm}で表す。(i, j, k, l) の組が等しいニューロンを 1 つのグループとし、その中で発火している（V の値が 1 である）ニューロンの m の値が第 kl 線分の終端の l方向の座標を表している。1 つのグループの中で発火しているニューロンの数が 0 個であったり、複数である場合は座標を決定できないため、常に 1 個だけが発火するように V の式を定める。

$$P_{i,j,k,l} = M \quad (\text{ただし } V_{i,j,k,l,M} = 1)$$

$$V_{i,j,k,l,m} = \begin{cases} 1, & U_{ijklm} = max\{U_{ijkln}\}, \\ & n = 0, 1, \cdots, size_l - 1 \\ 0, & \text{その他の場合} \end{cases}$$

始点と終点の座標はあらかじめ決まっているので、この部分に対応しているニューロンのグループでは常に同じニューロンが発火している。

図 4は図 3を表現している各ニューロンの出力の状態を示したものである。黒いマスが発火した状態を表しており、白いマスは発火していない状態を表している。$V_{ij1x4} = 1$ なので $P_{i,j,1,x} = 4$、$V_{ij1y2} = 1$ なので $P_{i,j,1,y} = 2$ になる。

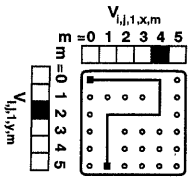

図 4: ニューロンによる経路の表現

4.3 動作関数

入力値 U の単位時間あたりの変化量を決める動作関数を求める。引数は、全ての出力値 V である。

4.3.1 規則違反のカウント

V_{ijkxm}が変化した場合、すなわち $P_{i,j,k,x}$が変化した場合、3 つの連続する線分 $(P_{i,j,k-1,x}, P_{i,j,k-1,y}) - (P_{i,j,k,x}, P_{i,j,k-1,y}) - (P_{i,j,k,x}, P_{i,j,k,y}) - (P_{i,j,k+1,x}, P_{i,j,k,y})$ が影響を受ける（図 5）。また、V_{ijkym}が変化した場合、すなわち $P_{i,j,k,y}$ が変化した場合も同様に $(P_{i,j,k,x}, P_{i,j,k-1,y}) - (P_{i,j,k,x}, P_{i,j,k,y}) - (P_{i,j,k+1,x}, P_{i,j,k,y}) - (P_{i,j,k+1,x}, P_{i,j,k+1,y})$ が影響を受ける。従って、各格子点の座標を決定する時にはそれらの 3 つの線分上に存在する障害物、他ネットの端子、他ネットの配線経路を考慮する必要がある。

V_{ijklm}の変化によって影響を受ける 3 つの線分上にある障害物の数を$\#OBST_{ijklm}$、他ネットの端子の数を$\#TERM_{ijklm}$、他ネットの x 線分の数を$\#LINE_{ijklm}$とする。

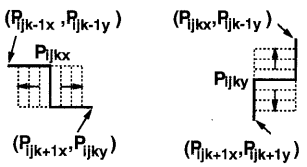

図 5: 座標の変化による影響

4.3.2 ループの via 数の抑制

電気的な信頼性を維持するためだけではなく、配線率を向上させるためにも配線長および via 数の抑制は重要である。

図 6のように $P_{i,j,k,l}$が $P_{i,j,k,l-1} \leq P_{i,j,k,l} \leq P_{i,j,k,l+1}$の範囲にない場合、方向 lについてループが存在している。ループを抑制することで配線長および via 数が抑制できる。N_{ijklm}の動作関数に与えるループ抑制項 $loop_{ijklm}$は次のようになる。

$$loop_{ijklm} = \begin{cases} min\{P_{ijkl-1}, P_{ijkl+1}\} - m, \\ \quad \text{if } m < min\{P_{ijkl-1}, P_{ijkl+1}\} \\ m - max\{P_{ijkl-1}, P_{ijkl+1}\}, \\ \quad \text{if } m > max\{P_{ijkl-1}, P_{ijkl+1}\} \\ 0, \\ \quad \text{その他の場合} \end{cases}$$

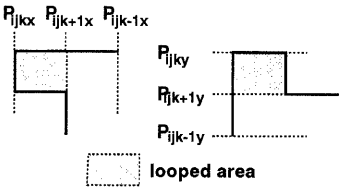

図 6: 配線経路のループ

動作関数の要素である $loop_{ijklm}$の式で V 以外の変数 $P_{i,j,k,l}$が使われているが、$P_{i,j,k,l}$は、V のみから求められる値である。

$$P_{i,j,k,l} = max\{V_{ijklm}\}, m = 0, 1, \ldots, size_l - 1$$

4.3.3 動作関数

以上から N_{ijklm}に与える抑制項は求められるが、このままでは入力値は単調減少なので、P_{ijkl}に対応するニューロンのグループの入力値が全て負になった場合、そのグループの全ての入力値に一定の値を加える必要がある。

$$hill_{ijkl} = \begin{cases} 1, & U_{ijklm} < 0, m = 0, 1, \cdots, size_l - 1 \\ 0, & \text{その他の場合} \end{cases}$$

とすると、動作方程式は次のようになる。

$$\frac{dU_{ijklm}(t)}{dt} = -A \cdot \#OBST_{ijklm}$$

$$-B \cdot \#TERM_{ijklm} - C \cdot \#LINE_{ijklm}$$

$$-D \cdot loop_{ijklm} + E \cdot hill_{ijklm}$$

$$(A, B, C, D, E : \text{正の定数})$$

$hill$関数の他にも、入力値の差が大きくなり過ぎることを防ぐために上限 U_{max}と下限 U_{min}を設定する。U の初期値はその間の値をとる。また、更新された U が U_{min}を下回っていれば U_{min}にセットする。

4.4 アルゴリズムの流れ

アルゴリズムの流れは以下のようになる。

1. 初期化
   U,V をランダムな値に初期化し、t=0 にする。
2. U の更新
3. 収束判定
   全ての $U_{ijkl}P_{ijkl}$ の動作関数について定数 A,B,C を係数に持つ項の値が 0 ならば step 6 に移る。
4. V の更新
5. t の更新
   t に 1 を加え、$t < t_{max}$ ならば step 2 に戻る。
6. 終了

5 評価に用いた問題

5.1 problem 1

一般的な配線問題である（図7)。グリッドサイズは8×8、ネット数は4で各ネットは1つずつしかペアを持たない。

配線は始点から x 方向に出発し、x 線分、y 線分共に 2 個ずつ持つものとした。

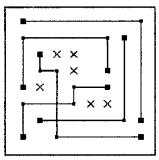

図 7: problem 1

5.2 problem 2

チャネル配線問題と呼ばれる（図8)。ゲートアレイやビルディングブロックのようなセミカスタムチップの細密配線に使われる問題で、上端、下端をセルに挟まれた長方形の配線領域（チャネル）上で上端と下端にのみ存在する端子列の間の配線経路を求める問題である。障害物は存在しない。

上端と下端に位置し、その x 座標が等しい端子のペアについては無条件に両端を結ぶ y 線分をひく（図 9.(a)）ものとし、その他の始点と終点の x 座標が異なる端子のペアの経路は、始点から y 方向に出発し、x 線分を 1 個と y 線分を 2 個持たせる（図 9.(b))。

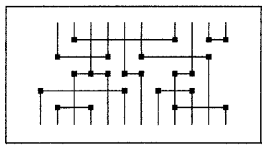

図 8: problem 2

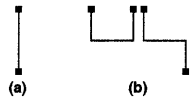

図 9: チャネル配線のパタン

5.3 problem 3

チャネル配線の後に残された、スイッチボックスと呼ばれるチャネル領域の交差点にあたる、セルと接することのない長方形の領域の配線問題である（図 10)。4 辺上にのみ存在する端子列の間の配線経路をを行なう。障害物は存在しない。

ここで扱っている問題は Burstein's difficult switchbox problem[9] と呼ばれ、辺上の白い四角で表された未配線のネットを配線できる HV ルールのアルゴリズムは知られていない。

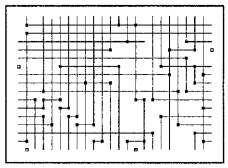

図 10: problem 3

problem 2 と同様に完全に向かい合う位置に始点、終点がある端子のペアについては、無条件に始点と終点の間を 1 本の線分で結ぶ（図 11.(a)）。始点と終点が隣接する辺上にある場合は x 線分と y 線分を 2 個ずつ使って経路を作る（図 11.(c)）。その他のペアは 3 本の線分で結ぶ（図 11.(b)）。

6 実験結果

以下ではニューロンの各値の更新回数の上限を 300 回 ($t < 300$) としてデータをとった。

表は初期値を変えて 100 回の試行を行なった際のニューラルネットワークの収束率と成功した時の収束までの平均

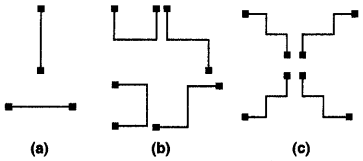

図 11: チャネル配線のパタン

更新回数である。

表 1: 収束率と平均更新回数

問題	収束率 (%)	平均更新回数
problem 1	79	91.8
problem 2	74	82.1
problem 3	8	109.9

また、problem 1 と problem 2 の更新回数の分布をそれぞれ図12と図13に示してある。problem 3 の収束回数は、23, 34, 62, 83, 122, 152, 161, 242 であった。

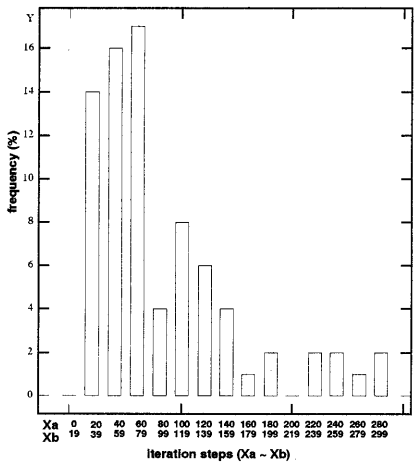

図 12: problem 1 の収束回数の分布

7 評価

図12、図13の更新回数の分布には明らかな偏りがある。共に収束する場合の 80%は、更新回数が 139 回以内である。また、最大更新回数を 300 回に設定していることを考えると、平均更新回数は 3 問ともあまり大きな違いはない。より多く種類、多くのサイズの問題を解かなければ分からないが、ニューロンの更新回数は、問題にあまり依存しないのではないかと考えられる。途中の経過を見ても、どの問題でもほとんどの場合最初の 30 回位の更新で規則違反を犯しているペアの数は数個にまで減っている。残された数個のペアの規則違反を解消するために残りのステップは用いられている。

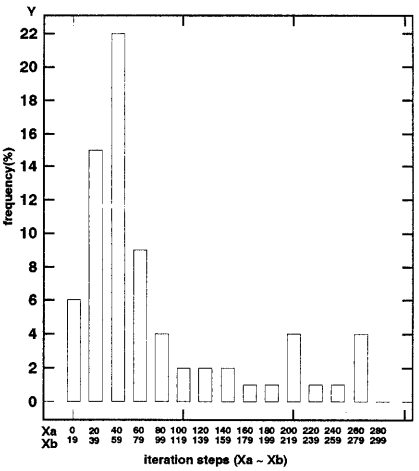

図 13: problem 2 の収束回数の分布

また、どの問題も収束率が低い。ほとんどの場合、同じ数個のペアの規則違反が最後まで解消できない場合が多い。より適切な項を動作関数に与えることによってモデルの改良をする必要がある。

チャネル問題とスイッチボックス問題では、異なるネットの端子が向かい合って配置されている場合に制約を受けてしまう。図14を例に説明する。ネット C の x 線分はネット B の x 線分より下に位置しなければならない。また、ネット B の x 線分はネット A の x 線分より下に位置しなければならない。しかし、動作関数にはこの制約条件を満たすような要素が含まれていない。最も収束率が低かった problem 3 では、図10の左下の部分で隣接する複数のネットが関連しあってこの制約を作り出している。現在の動作関数では、このように配線密度が高く制約条件が複雑な部分では、制約条件を犯すような位置関係に陥ってしまう。この問題に対処するために、動作関数にこれらの制約を守る要素を加える、最初から制約条件を犯すような場所に対応するニューロンを用意しない等の方法が考えられる。これらの方法を用いて相関する制約条件を持つネット群の経路の変化を少なくすると、同じ部分を通っている関係のないネットの配線経路が別の場所に逃げて収束率を上げることができる。また、最初からニューロンを用意しない場合

は、計算量が減るためにアルゴリズムの高速化にもつながる。その他に、現在は端子が完全に向かい合う位置にあるペアは無条件にその間を1本の線分で結んでいるがこのペアの自由度を上げることも考えられる。

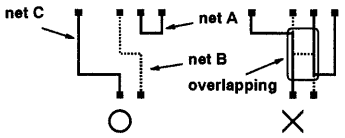

図 14: 制約条件

8 並列計算機への実装方法

8.1 ターゲットマシン: ATTEMPT-0

このアルゴリズムは、当大学で開発された並列計算機テストベッド ATTEMPT-0 [10] (図 15) 上に実装する予定である。10 台の PU が IEEE Futurebus[11] を通して共有メモリと結合されているが、そのうちの 2PU は共有メモリとして使用されるため最大使用 PU は 8 台である。各 PU は、CPU/FPU (68030/68882)、共有メモリに比べより高速なローカルメモリ、共有メモリへのライトスルースヌープキャッシュから構成されている。また、同期と高速なマルチキャストを行なうためのシンクロナイザメモリが用意されている。

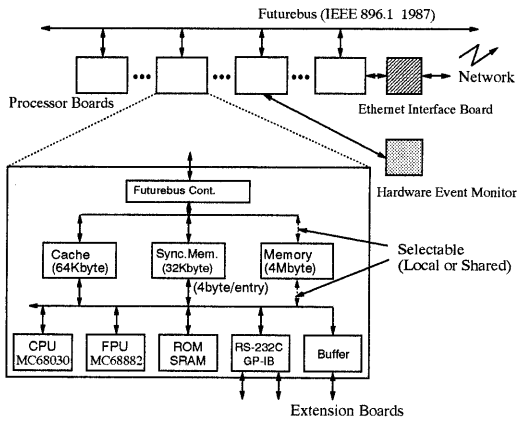

図 15: ATTEMPT-0

8.2 並列処理の単位

アルゴリズムは、基本的には1ニューロン単位での細粒度並列処理が可能であるが、以下の理由により同一グループに属するニューロンの入力値、出力値の更新をまとめて1つの並列処理単位とし、同じ PU 上で処理する。

- ATTEMPT-0 は、10PU が IEEE Future Bus によって結合されている。そのうちの 2PU は共有メモリとして使用されるため、PU は最大で 8PU までしか使用できない。
- $\#OBST, \#TERM, \#LINE, loop$ を求める計算で共通する過程が多い。
- ニューロンの更新には同一グループのニューロンの U の値を参照しなければならないが、同じ PU 上で処理すると、U の値を他プロセッサとやりとりする必要がない。

8.3 メモリ配置

並列プログラムを高速に実行するために、通信の局所性を利用し PU 間通信を極力抑えなければならない。

- $\#OBST, \#TERM, \#LINE, loop$
  これらの値は各ニューロンに固有であるとともに、他のニューロンから参照されることはない。各ニューロンに対応する PU のローカルメモリに配置される。
- 入力値 U
  U は V の更新の際に同一グループのニューロンの中だけで参照され、それ以外のニューロンに参照されることはない。このため、並列処理単位の中で完全にローカルである。従って U の値はローカルメモリに配置する。
- 出力値 V
  全ての V の値は U の動作関数を計算する際に全てのニューロンから参照されるため、もしもローカルメモリに配置すると頻繁に PU 間でデータの授受を行なわなければならない。従って V は共有メモリ上に配置する。
- 時間 t
  t は全てのニューロンに共有されるデータであるが、共有メモリの使用を少なくするために、ローカルメモリに配置して各 PU が更新する。

8.4 並列処理の流れ

PU 間の通信、同期が必要になるのは以下の箇所である。

- 全ての出力値 V の初期化が完了したのを確かめるために、$t = 0$ 時の U の更新を始める前に全 PU で同期をとる。
- 収束判定は、各 PU が自分の受け持ったニューロンについてローカルに行ない、PU 間通信でグローバルな収束判定を行なう。
- t = 1 以降の U の更新の際に更新される以前の V の値が参照されないように、V の更新後全 PU で同期をとる。

9 結論

ニューラルネットワークに基づく細密配線アルゴリズムを提案した。従来よりも逐次的制約が少なくより並列処理に適したアルゴリズムであると思われるが、収束率が大変悪くさらにモデルを改良する必要がある。その他の今後の課題として、

- 異なるサイズを持つ問題によるアルゴリズムの検証
- 並列計算機上での速度向上の測定

が挙げられる。

参考文献

[1] H. Date, Y. Matsumoto, K. Kimura, K.Taki, H. Kato, M. Hoshi, "LSI-CAD Programs on Parallel Inference Machine," Proceedings of the International Conference on Fifth Generation Computer Systems, pp.237-247, 1992.

[2] T. Yamauchi, A. Ishizuka, T. Nakata, N. Nishiguchi, N. Koike, "PROTON: A Parallel Detailed Router on an MIMD Parallel Machine," International Conference on Computer-Aided Design, pp.340-343, 1991.

[3] 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.

[4] T. Watanabe, H. Kitazawa, Y. Sugiyama, "A Parallel Adaptable Routing Alogrithm and its Implementation on a Two-Dimensional Array Processor," IEEE Trans. Computer-Aided Design, pp.243-249, vol. CAD-6, no. 2, March 1987.

[5] Pao-Hsu Shih and Wu-Shung Feng, "A Neural Network Approach to Channel Routing," IEEE International Symposium on Circuit and System, pp.1593-1596, 1991.

[6] R. Fujii, M. F. Tenorio, and H. Zhu, "Use of Neural Nets in Channel Routing," IEEE INNS Joint Conference On Neural Networks, vol.1, pp.321-325, 1989.

[7] Y.Takefuji, "Neural Network Parallel Computing," Kluwer Academic Publishers, Jan. 1992.

[8] K. Suzuki, H. Amano, Y. Takefuji, "Neural Network Parallel Computing for Channel Routing Algorithms," International Conference on Automation, Robotics, and Computer Vision, pp.INV-7.3.1-INV-7.3.5, Sept. 1992.

[9] M. Burstein and R. Pelavin, "Hierarchical wire routing," IEEE Trans. Computer-Aided Design, vol. 2, no. 4, pp. 2345-244, Oct. 1983.

[10] H.Amano, T.Terasawa and T.Kudoh, "Cache with synchronization mechanism," Proc. of IFIP Congress89, pp.1001-1006, August, 1989.

[11] "IEEE Standard Backplane Bus Specification for MultiprocessorArchitectures: Futurebus," IEEE , Jun. 1988.