計算機アーキテクチャ　71－12
(1988. 7. 7)

並列ニューラルネットワークシミュレータ
Parallel Neural Network Simulators

梶原 信樹, 田地野 真次†, 中田 登志之, 松下　智, 小池 誠彦
Nobuki KAJIHARA, Shinji TAJINO†, Toshiyuki NAKATA,
Satoshi MATSUSHITA, Nobuhiko KOIKE

日本電気(株) C&C システム研究所, †日本電気技術情報システム開発(株)
C&C Systems Research Laboratories NEC Corp.,
†NEC Scientific Information System Development Ltd.

あらまし　近年、ニューラルネットワークの研究が盛んでハードウェア化の研究も行われている。本文ではニューラルネットワークの持つ並列性を引き出し大規模なニューラルネットワークの高速なシミュレーションを実現するために現在開発中の２つの並列ニューラルネットワークシミュレータについて述べる。１つは汎用シミュレーションマシンとして開発した分散共有メモリ型のマルチプロセッサシステム上でのシミュレータでネットワーク分割によりニューラルネットワークの持つ並列性を引き出す。もう１つはニューラルネットワークシミュレーションに特化し、ネットワーク分割に加え専用プロセッサによるシミュレーションアルゴリズムのパイプライン化によりアルゴリズムの持つ並列性も引き出す。

Abstract　This paper describes two parallel neural network simulators, CjNet and NeuMan, which were especially designed for high speed simulation of large-scaled neural networks. CjNet was developed on a general purpose multi processor system with distributed and shared memory. CjNet makes use of parallelism inherent in neural networks through sub-network parallelism. NeuMan on the other hand is a special-purpose processor dedicated to neural network simulation. Besides making use of sub-network parallelism, it makes use of parallelism in the simulation algorithm through pipelining.

１．はじめに

ニューラルネットワークを粒度の細かい並列計算のモデルと考え、情報連続、時間連続のニューラルネットワークモデルを提案し，ニューラルネットワーク記述言語，コンパイラ，シミュレータの試作，及びニューラルネットワークによる小規模な情報処理機能（簡単な制御システム，前向き推論システム，時系列パターンの認識，Ｎ＿Ｑｕｅｅｎ，等）の実験を行なっている。ニューラルネットワークのプログラムはリンクの重みとして格納し、シミュレーションにより動作の確認を行なった。シミュレータは従来型の計算機（ＬＩＳＰマシン，ＶＡＸ，ＰＣ９８０１）上にソフトウェアでインプリメントした。ＬＩＳＰマシン上のソフトウェアによるシミュレーションでは９０～４００リンク／秒の処理速度である（但し、ＬＩＳＰマシン上のシミュレータは作成に当たって処理速度はあまり重視していないのでこの数値はもう１桁程度は向上すると思われる）。ＶＡＸ８６５０上に高速性を重視して試作したシミュレータでは約１４０Ｋリンク／秒の速度を実現できた。しかし、より大規模な実験や実際のアプリケーションの開発には処理速度が不十分である。これは逐次型の計算機で並列な計算をシミュレートしているためで、ニューラルネットワークの並列性を引き出せるシミュレータの開発が必要である。

並列処理の研究の一貫として電気系のＣＡＤを対象とした汎用シミュレーションマシンを開発した。このシミュレーションマシンは分散共有メモリ型のマルチプロセッサシステムで、この上にニューラルネットワークシミュレータ（ＣｊＮｅｔ）を開発した。

さらに高速なシミュレーションを行なうためにニューラルネットワークのシミュレーションに特化した専用マシン（ＮｅｕＭａｎ）を開発中である。ＮｅｕＭａｎは複数の専用ＰＥ（プロセッサエレメント）とＰＥ間通信のための多段接続ネットワークで構成される。ネットワーク分割に加えて、専用ＰＥによるシミュレーションアルゴリズムのパイプライン化によりアルゴリズムの持つ並列性も引き出す。

以下ではシミュレータの対象とする時間連続のニューラルネットワークモデル、設計方針、ネットワーク分割による並列シミュレーション方式、汎用シミュレーションマシン上のニューラルネットワークシミュレータＣｊＮｅｔ、及びニューラルネットワーク専用マシンＮｅｕＭａｎについて述べる。

2．ニューラルネットワークモデル

図１にニューラルネットワークシステムのイメージ図を示す。『環境』はニューラルネットワークの動作する環境で、一般には時間とともに状態を変化させるダイナミックなシステムである。ニューラルネットワークは多数のノードと各ノードを接続する重み付のリンクから構成される。

ニューラルネットワークはダイナミックに変化する環境を認識しながら動作する。このため各ノードは活性度という内部状態を持ち、環境または他のノードの影響を受けながら時間とともに変化する。各ノードの活性度のパターンが現在の状況を表しているといえる。ニューラルネットワークは現在の状況（各ノードの活性度）と環境からの情報に従って行動を決定し、次の状態に変化するダイナミックなシステムである。

ノードにはセンサノード、コンテキストノード、モータノードの３種類のノードがある。センサノードは環境の状態に応じてその活性度を変化させ、モータノードはその活性度に応じて環境に操作を加える。コンテキストノードはセンサノードとモータノードの介在を行い、ここでシステムの短期記憶と情報処理の機能が実現される。

ネットワークの構造には対称，層構造等の特定の構造は仮定しない。自分自身へのフィードバックも含めて任意の構造をとる。理論的な解析や学習には特定の構造を仮定した方が便利であるが、情報処理能力という観点から見ると構造を仮定しない方が有利である。

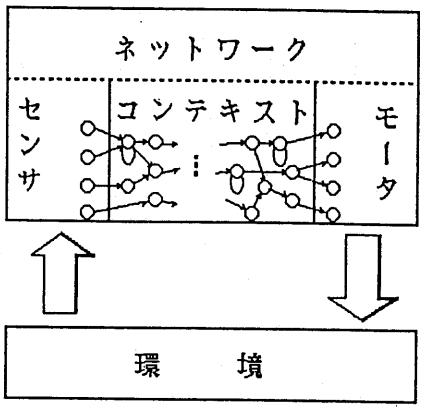

図１．ニューラルネットワークシステム

ダイナミックに変化するパターンを認識するためのニューラルネットワークモデルとして以下に説明する時間連続、情報連続のモデル[1]を提案する。図２は自分自身へのフィードバックを許すニューラルネットワークのモデルである。図２のノードｘの活性度は他のノード$y_1, y_2, \cdots, y_n$及び自分自身の活性度の影響を受けて変化する。$W_{y_i x}$は、ノードy_iからｘへのリンクの重み、W_{xx}はｘからｘ自身への重みである。各ノードの活性度をｘ，y_1，y_2，…，y_nとするとｘは次の微分方程式に従って変化する。

$$\tau \frac{dx}{dt} = \sum W_{y_i x} \cdot \mu(y_i) + W_{xx} \cdot \mu(x) - x \qquad (1)$$

（１）式を動作方程式と呼ぶ。$W_{y_i x} \cdot \mu(y_i)$をノードy_iからノードｘへの投票と呼ぶ。τは時定数でこの値が小さいほど活性度の変化は大きくなる。出力関数μは（２）式、図３に示す様なリミター特性を持つ広義の単調増加関数である。

$$\mu(x)=\begin{cases}0 & ;\ x\leqq 0\\ 2x^2 & ;\ 0<x\leqq 0.5\\ 1-2(1-x)^2 & ;\ 0.5<x\leqq 1\\ 1 & ;\ 1<x\end{cases}\qquad(2)$$

動作方程式（１）はオイラー法によって逐次的に解くことができる。時間Δｔ毎の積分は次の漸化式で実行できる。

$$\begin{aligned}acc\,x &:= \Sigma W_{y_i x}\cdot\mu(y_i)+W_{xx}\cdot\mu(x)\\ \Delta x &:= \Delta t\,[acc\,x-x]/\tau\\ x &:= x+\Delta x\end{aligned}\qquad(3)$$

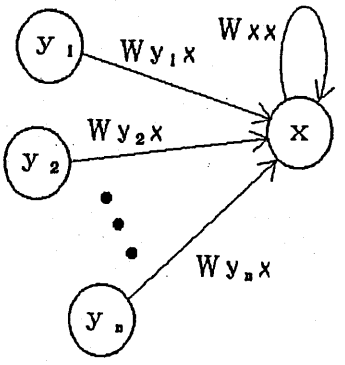

図２．ニューラルネットワークモデル

ニューラルネットワークの動作は各ノード毎に定義される動作方程式（１）の連立方程式で定義される。従ってニューラルネットワーク内の全てのノードに関し（３）式を実行するとニューラルネットワーク全体についてΔｔのシミュレーションができる。これをシミュレーションの１ステップと呼ぶ。シミュレーションの１ステップにはニューラルネットワークの全ノードについて次の３つの処理を行なう必要がある。

［投票フェーズ］
ノードの活性度に出力関数を適用した結果にリンクの重みを乗じた値（投票値）を他のノードに伝える。
［累算フェーズ］
他のノードから送られた投票値を累算する。
［更新フェーズ］
累算結果からノードの活性度を更新する。

この３つの処理は本ニューラルネットワークモデルに限らず他のニューラルネットワークモデルのシミュレーションにも共通する処理である。２つの並列ニューラルネットワークシミュレータは本モデルの高速なシミュレーションを目標とするが、プログラム、マイクロプログラムの変更により他のモデルのシミュレーションも可能である。

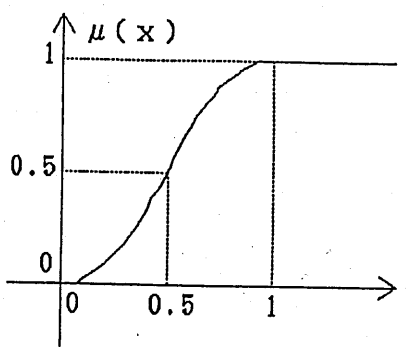

図３．出力関数μ

３．基本設計

ニューラルネットワークは行列によって表現することができる。スーパーコンピュータ，アレイプロセッサは行列演算を高速に行なうことができ、これらの高速プロセッサを用いたニューラルネットワークシミュレータが提案されている。ニューラルネットワークを行列で表現する方法ではニューラルネットワークのノードどうしが殆ど完全結合しているような場合には効率の良い処理ができる。しかしシミュレーションに必要な記憶容量，ハードウェア量はノード数の２乗に比例して増加するのでノード数が数千，数万のニューラルネットワークのシミュレーションは現実的でない。

一般にニューラルネットワークのノードはすべてのノードとリンクを持つわけではない。人の脳でもニューロンの数は数１００億といわれているが１つのニューロンの持つシナプス結合は数千から数万で脳全体のニューロンの数に比べると非常に少ない。実用的な情報処理システムをニューラルネットワークで実現する場合は幾つかの機能モジュール（部分ネットワーク）で全体が構成される。このとき各機能モジュール内のノード間ではリンクの数は多いが、機能モジュール間ではそれ

ほど多くならない。この様なニューラルネットワークを行列で表現した場合、要素のほとんどは零のスパースな行列となりスーパーコンピュータやアレイプロセッサによるアプローチでは無駄な計算，メモリ，ハードウェアが多くなり効率的ではない。

提案する２つのシミュレータはニューラルネットワークの接続情報は１つの行列で表現するのではなくノード毎にそのノードをソースとするリンクの表（ＦａｎＯｕｔ表）を持つことで表現する。

ニューラルネットワークの持つ並列性をいかに引き出すかがシミュレータの性能を左右する。ここではニューラルネットワークの持つ並列性、及びシミュレーションアルゴリズムの持つ並列性に着目しシミュレーションの高速化を図った。

ニューラルネットワークの各ノードは並列に動作することができる。従って、ニューラルネットワークのノード１つに１つの物理的なプロセッサを割り当てると並列性は最も高くなる。しかしノードのシミュレーションに必要な処理をディジタル処理で実行するハードウェアをノード毎に割り当てるのは技術的にもコスト的にも現実的ではない。ニューラルネットワークを複数の部分ニューラルネットワークに分割しこれを複数のプロセッサエレメント(ＰＥ)に割り当てるアプローチを採用する。１つのＰＥ内ではノードは逐次的に処理されるが各々のＰＥに割り当てられた部分ネットワークは並列に処理される。これはＣＡＤに於ける論理シミュレーションで回路分割として実用化されている手法である。

シミュレーションの１ステップを構成する３つの処理のうち投票フェーズと、累算フェーズは並列に実行できる。ニューラルネットワーク専用マシンＮｅｕＭａｎでは、各ＰＥは演算ユニットを２つ持ち２つのフェーズをパイプライン的に並列処理できるように専用化し、シミュレーションアルゴリズムの持つ並列性も引き出す。

４．並列シミュレーションアルゴリズム

２つのシミュレータ（ＣｊＮｅｔ，ＮｅｕＭａｎ）は、どちらもニューラルネットワークを複数の部分ニューラルネットワークに分割し複数のＰＥに割当て並列に処理することによりニューラルネットワークの持つ並列性を引き出す。シミュレーション用のデータ構造（ニューラルネットワークの表現法）、シミュレーションアルゴリズムも基本的には同じである。

シミュレーション用のデータ構造を図４に示す。ノードの活性度を保持する活性度表Activity[ｘ]、他のノードからの投票値を累計する累算器表Accumulator[ｘ]、及びノードから出るリンクの情報を保持したＦａｎＯｕｔ表（ノードｘが影響を与えるノードｙのアドレスとノードｙへのリンクの重みＷxyの表）からなる。各ＰＥは担当する部分ネットワークの情報を図４の形式で保持する。

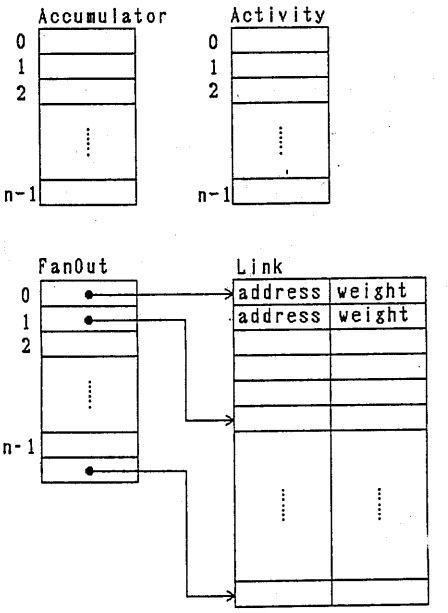

図４．シミュレーション用データ構造

各ＰＥはこのデータ構造を参照して、他のＰＥと同期しながらシミュレーションを行なう。各フェーズでの処理は次のようになる。

```
【投票フェーズ】
FOR x IN PEiが担当する全てのノード DO
  mx := μ(Activity[x]);
  FOR link IN ノードxの全てのFanOutリンク DO
    MSG.addr := link.address;
    MSG.vote := link.weight × mx;
    put(MSG);
sync();
```

【累算フェーズ】

```
FOR MSG in 送られてきた全てのメッセージ DO
  Accumulator[MSG.addr] :=
    Accumulator[MSG.addr] + MSG.vote;
```

【更新フェーズ】

```
FOR x IN PEiが担当する全てのノード DO
  Activity[x] := Activity[x] +
    Δt×{Acuumulator[x] - Activity[x]}／τ;
  Accumulator[x] := 0.0;
```

【記号の説明】

```
          PEi: プロセッサエレメントi
   Activity[x]: ノードの活性度
Accumulator[x]: ノードの累算器
      MSG.addr: メッセージのアドレス部
      MSG.vote: メッセージの投票部
  link.address: リンクのノードアドレス
   link.weight: リンクの重み
        sync(): 他のＰＥとの同期処理
         get(): メッセージの入力
      put(MSG): メッセージの出力
           Δt:  積分の刻み幅
```

投票フェーズでは活性度表とＦａｎＯｕｔ表を参照して投票処理を行なう。ノードの活性度に出力関数μを適用しノードの出力を計算する。この値にリンクの重みを乗じた値（投票値）と投票先のノードのアドレスを投票メッセージにまとめ他のＰＥに送る。担当する全てのノードの投票処理が終了すると他のＰＥと同期のための処理を行なう。

累算フェーズでは、他のＰＥから届いた投票メッセージのアドレス部で指定されるノードの累算器にメッセージの投票部の値を加算する。

投票・累算フェーズでは、同一ＰＥで処理されるノード間のリンクの処理には他のＰＥとの通信は必要ない。異なるＰＥに割り当てられた部分ニューラルネットワークに属するノード間にリンクがある場合はＰＥ間の通信が必要になる。

投票フェーズと累算フェーズが終了すると更新フェーズを実行する。更新フェーズでは活性度表と累算器表を参照して、担当する全てのノードについて活性度の更新処理を行う。

５．汎用シミュレーションマシン上のシミュレータ

電気系のＣＡＤ用に開発した汎用シミュレーションマシンのシステム構成を図５に示す。本シミュレーションマシンは複数のＰＥをＶＭＥｂｕｓ準拠のバスで結合した分散共有メモリ型のシステムである。各ＰＥは汎用のマイクロプロセッサシステムで３２ビットマイクロプロセッサ（６８０２０）と浮動小数点演算コプロセッサ（６８８８２）、浮動小数点アクセラレータ（ＷＴＬ１１６７）を持つ。ＰＥは４Ｍバイトのローカルメモリを持つ。このメモリは２ポートで他のＰＥからもアクセスできる。ローカルメモリには、ローカルなアドレスとグローバルなアドレスが割り当てられている。ＰＥは自分自身のローカルメモリはローカルアドレス、グローバルアドレスのどちらでもアクセスできる。他のＰＥのローカルメモリはグローバルなアドレスによってアクセスできる。

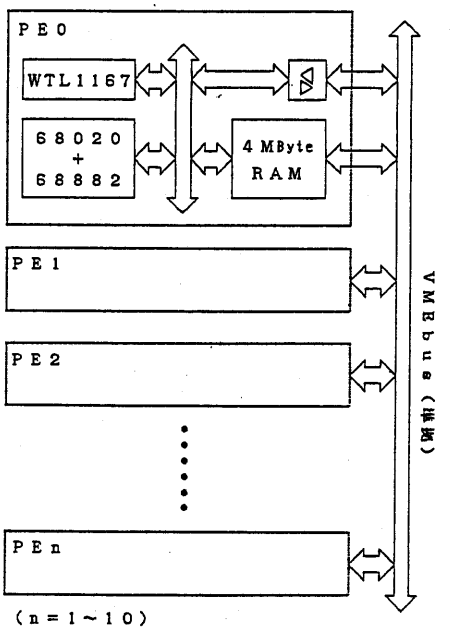

図５．汎用シミュレーションマシンシステム構成

このシステムは、汎用性が高くこの上でいくつかのアプリケーションを開発している。アプリケーションの１つとしてニューラルネットワークの持つ並列性を引出せるニューラルネットワークシミュレータ（ＣｊＮｅｔ）を開発した。

ＣｊＮｅｔでは、４節の並列アルゴリズムをプログラムによって実現した。

ＣｊＮｅｔはＰＥ間通信はバスを介して行なう。異なるＰＥに割り当てられた部分ニューラルネットワークに属するノード間のリンクの処理は次のようにバスのトラフィックを軽減するようにしている。

ある部分ニューラルネットワーク（Ｎ１とする，ＰＥ１が担当）の複数のノードｙ１，．．ｙｎから別の部分ニューラルネットワーク（Ｎ２，ＰＥ２が担当）のあるノードＸへリンクがある場合、各リンク毎にＰＥ間通信を行なう必要はない。ノードｙ１，．．ｙｎからノードｘへの投票値の累算はＰＥ２ではなくＰＥ１が行い、累算結果を投票値としてＰＥ１に送ることにより、ＰＥ間通信をｎ回から１回に減らすことができる。

ＣｊＮｅｔの性能評価はまだ行なっていないが、ＰＥ単体の性能は約１００Ｋリンク／秒で、ＶＡＸ８６５０の０．７倍、次節で説明するニューラルネットワーク専用マシンＮｅｕＭａｎのＰＥの０．１倍の処理速度である。

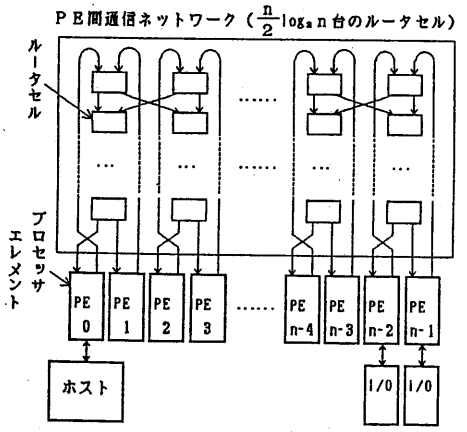

図６．ＮｅｕＭａｎシステム構成

６．ニューラルネットワーク専用マシン

ニューラルネットワーク専用マシンＮｅｕＭａｎのシステム構成、ＰＥのブロック図を図６，図７に示す。ＮｅｕＭａｎは複数の専用ＰＥとＰＥ間でメッセージを転送するためのＰＥ間通信ネットワークで構成される。

ＮｅｕＭａｎはホストのバックエンドプロセッサとして動作し、シミュレーション用のニューラルネットワークのデータはＰＥ間ネットワークを介してホストから各ＰＥにロードされる。シミュレーションの制御、シミュレーション時の各ノードの活性度のモニタ、環境とのインタフェースもホストが行なう。また各ＰＥはＩ／Ｏポートを持っておりホストを介さずに環境とインタフェースを行なうことも可能である。

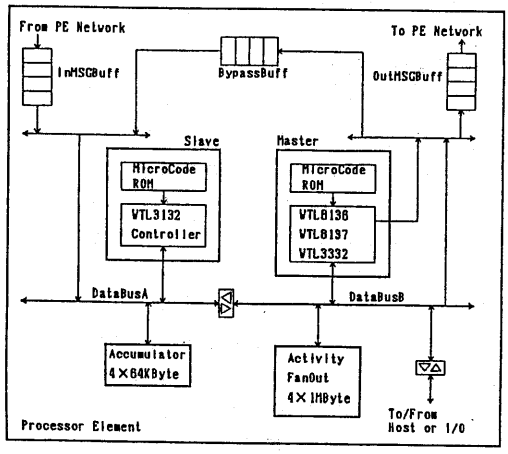

図７．プロセッサエレメント（ＰＥ）

ＰＥは投票フェーズと更新フェーズをパイプライン的に並列に実行するためにマスタとスレーブの２つの処理ユニットを持つ。マスタ，スレーブはともにマイクロプログラムで制御されマイクロコードの変更によりいろいろなニューラルネットワークモデルや学習にも対応できる。マスタは３２ビット浮動小数点プロセッサ，整数プロセッサ，シーケンサで構成されＰＥ内のすべての資源にアクセス可能で投票フェーズと更新フェーズを実行する。スレーブは３２ビット浮動小数点プロセッサと単純なコントローラで構成され、マスタが投票フェーズを実行中にこれと並列に累算フェーズを行なう。ニューラルネットワークのシミュレーションの大きな部分を占める投票フェーズと累算フェーズをマスタ、スレーブでパイプライン的に並列に処理することにより高速なシミュレーションを実現する。投票，累算フェーズ終了後の更新フェーズはマスタが実行する。

投票フェーズで参照する活性度表、ＦａｎＯｕｔ表と累算フェーズで参照する累算器表は各々の

演算ユニットから独立にアクセスすることができる。更新フェーズでは累算器表と活性度表を参照する必要がある。マスタはスレーブを停止した状態で累算器表、活性度表をアクセスする。

ＰＥは他のＰＥと投票メッセージの送受信用にＰＥ間ネットワークの入出力ポートを持つ。また、２つのユニット間には投票メッセージ用のバイパスがあり、同一ＰＥ内のノード間の投票メッセージはこのバイパスを使いＰＥ間ネットワークのトラフィックを軽減する。

活性度表・ＦａｎＯｕｔ表の容量は４Ｍバイト，累算器表の容量は２５６Ｋバイトで、１つのＰＥでは最大６４Ｋノード，４４０Ｋリンクのニューラルネットワークのシミュレーションが可能で、ＰＥあたり約１Ｍリンク／秒の処理速度を実現する。

シミュレーション時のデータ転送，ホストによるシミュレーションの制御，ＰＥの同期，ＰＥへのニューラルネットワークデータのロードはＰＥ間通信ネットワークを介してメッセージベースで行なう。

メッセージは可変語長（１語１６ビット）でヘッダ（１語）と本体（０～１２７語）から成り、ヘッダにはメッセージのタイプ，送り先のＰＥのアドレス，メッセージ長の情報を含む。

ニューラルネットワークシミュレーション時の各ノード間の投票値の転送は投票メッセージで行なう。投票メッセージは３語構成で、投票の送り先のノードを識別するための情報と、投票値を含む。

ＰＥ間通信ネットワークは複数のルータセル（２×２の交換器）からなる多段接続ネットワークである。ｎ台のＰＥを接続するためには[ｎ／２]×$\log_2 n$台のルータセルが必要である。図８にルータセルのブロック図を示す。ルータセルは、２つの入力ポートＡ，Ｂと２つの出力ポートＸ，Ｙを持つ２×２の交換器で内部に４本のメッセージバッファを持つ。各ポートの幅は１６ビットである。メッセージはそのアドレスに従って所定のバッファに入れられる。どちらのバッファに入れるかは前段のルータセルの出力ポートが制御する。

ルータセルはマルチプレクサへの入力となる２つのバッファが両方とも空の時以外は稼働し続けることができる。シミュレーション時に各ＰＥが１Ｍリンク／秒の速度で動作するためにはルータセルは２つの入力ポートの各々から平均１Ｍメッセージ／秒の投票メッセージを受取り、２つの出力ポートのそれぞれへ同様のメッセージ密度で送り出さなければならない。本ルータセルは１つの入力ポートの平均メッセージ密度が１．５Ｍメッセージ／秒まではバッファが溢れることなく（溢れる確率は1.5×10^{-8}）交換できる。

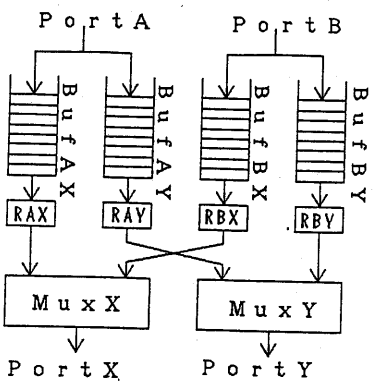

図８．ルータセル

７．おわりに

分散共有メモリ型のマルチマイクロプロセッサ上のニューラルネットワークシミュレータＣｊＮｅｔ、及びニューラルネットワークシミュレーションに専用化したニューラルネットワーク専用マシンＮｅｕＭａｎについて述べた。両シミュレータともにネットワーク分割によりニューラルネットワークの持つ並列性を引き出す。さらにＮｅｕＭａｎではＰＥとＰＥ間通信ネットワークを専用化することによりシミュレーションアルゴリズムの持つ並列性も引き出す。

今後は、ＣｊＮｅｔ上に大規模なニューラルネットワーク応用システム（画像処理システム）を構築する。ＮｅｕＭａｎは、ＰＥ４～８台構成のシステムを試作しアーキテクチャの妥当性を確認する予定である。また、シミュレータと密結合したニューラルネットワークプログラミング環境を開発する。ニューラルネットワークプログラミング環境にはニューラルネットワーク記述言語、コンパイラ、ネットワーク分割ローダ、モニタが必要で、特に記述性の高いニューラルネットワーク記述言語とネットワーク分割ローダは重要である。

ネットワーク分割によるニューラルネットワー

クシミュレータが効率良くシミュレーションを実行するためには部分ニューラルネットワーク間のリンクがなるべく少なくなる様にニューラルネットワークを部分ニューラルネットワークに分割しなければならない。ニューラルネットワークプログラムはニューラルネットワーク記述言語を使って機能毎にモジュール化して記述される。ニューラルネットワークの分割にはこのモジュール化の情報を利用することができる。

8．参考文献

［1］梶原，辻；”ニューラルネットワーク推論システム”；情報処理学会知識工学と人工知能研究会資料４５－１０（１９８６）

［2］梶原，中田，小池；”専用並列マシンＭＡＮ－ＹＯにおけるニューラルネットワークシミュレータ”；情報処理学会第３５回全国大会予稿集（１９８７）

［3］梶原，中田，松下，小池；”ニューラルネットワークシミュレーションマシン：ＮｅｕＭａｎ”；情報処理学会「コンピュータアーキテクチャ」シンポジウム（１９８７－５）