3C－8 並列シミュレーションマシンCenjuにおけるニューラルネットワークシミュレータ

田地野 眞次* 浅野 由裕* 梶原 信樹** 小池誠彦**
(*日本電気技術情報システム開発(株)、**日本電気(株))

1. はじめに

ニューラルネットワーク(NN)シミュレータを並列シミュレーションマシンCenju[1]上に実現し、その評価を行った。本稿では、その成果について報告する。

本シミュレータでは、時間連続、情報連続なニューロンモデルを採用した。NNとしては、自分自身へのフィードバックリンクも含めた任意の構造を持つものを扱える。また、処理速度の高速化を考慮し、NN分割や、プロセッサ間通信量の削減等の工夫を施している。

本シミュレータの評価は、画像処理用にHough変換のアルゴリズムをNNで実現したもの(大規模なNN)を用いて行った。

2. NNモデル

シミュレータで扱うNNシステム(図1)とNNモデル(図2)について説明する。

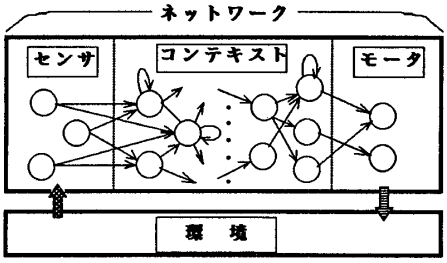


図1 NNシステム

環境は、一般には、時間とともにその内部状態を変化させる動的なシステムである。NNはセンサ、モータノードを介して環境と情報を交換する。コンテキストノードはセンサ、モータノードを介在し、ここで何らかの情報処理が行われる。

動的な環境で動作するためNNも動的なシステムとして定義した。各ノードは活性度という内部状態を持ち、環境(センサノードのみ)または重み付きのリンクを介して他のノードの影響を受けて、その活性度を時間とともに変化させる。つまり、図2においてノードXの活性度は、ノードY_1、Y_2、…、Y_nの影響により変化する。

ノードXの活性度xは式(1)の動作方程式に従って変化する。出力関数μは式(2)で定義され、図3に示す非線形飽和特性を持つ。

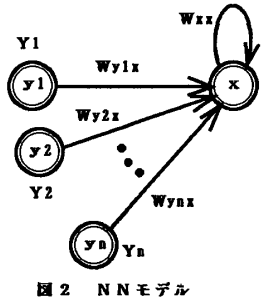


図2 NNモデル

動作方程式:

$$\tau dx/dt = \Sigma Wyix\,\mu(yi) + Wxx\,\mu(x) - x \quad \cdots (1)$$

Wyix:ノードyiからxへのリンクの重み(正負の実数値)
Wxx:ノードxからx自身へのリンクの重み(正負の実数値)
X、Yi:ノード名
x、yi:ノードの活性度
μ():出力関数
τ:時定数

動作方程式(1)はオイラー法によって解き、以下に示す漸化式に従ってΔt毎に計算する。

$$accx := \Sigma Wy_i x \cdot \mu(y_i) + Wxx \cdot \mu(x)$$
$$\Delta x := \Delta t(accx - x)/\tau \quad \cdots (3)$$
$$x := x + \Delta x$$

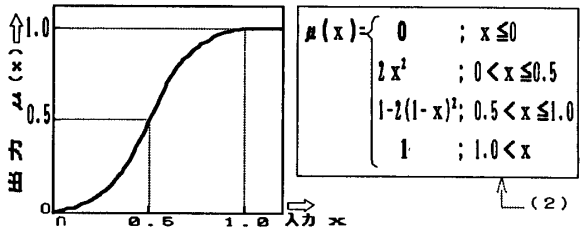


図3 出力関数

3. シミュレータの構成

本シミュレータは、分散共有メモリ型マルチプロセッサシステムCenju上に作成した(図4参照)。

Cenjuは、汎用な高速演算ユニットとローカルなメモリを持つ複数のプロセッサエレメント(PE)から構成されている。各PEは、共有バスで接続され、相互に他のPEのメモリにアクセス可能な構成となっている。

シミュレータは、単一のマスタPE上で、環境(図1参照)とセンサ、モータノード間の状態交換処理を行う。また、複数のスレーブPEにコンテキストノードの処理を分散させ、並列処理可能な構成になっている。

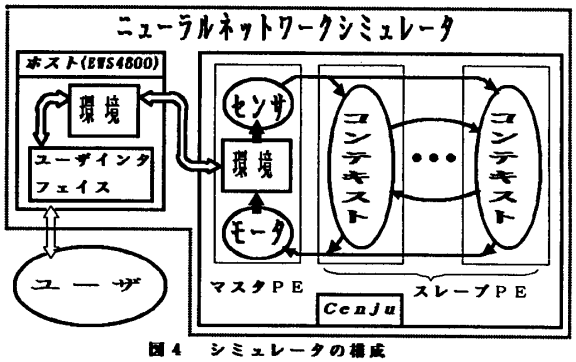


図4 シミュレータの構成

4. シミュレーションの実現

(1) 基本動作

ノードの活性度は、式(3)で示した漸化式に従い時間Δtを1単位として変化する。漸化式の処理は①投票、②累算、③更新の3つのフェーズ[2]をNNの全ノードに対して行う。それぞれの処理内容を以下に示す。

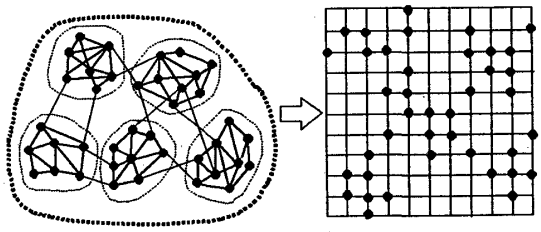
①ノードの活性度に出力関数を適応した結果にリンクの重みを乗じた値(投票値)を他のノードに伝える処理。
②他のノードから送られた投票値を累算する。
③累算結果からノードの活性度を更新する。

これらの処理は、各プロセッサで同期を取りながら並列実行される。

Parallel Neural Network Simulator of Cenju
Shinji TAJINO*, Yoshihiro ASANO*, Nobuki KAJIHARA**, Nobuhiko KOIKE**
*NEC Scientific Information System Development Ltd., **NEC Corporation

（2）データ構造

一般に大規模なＮＮでは、それを行列表現するとランダムでスパースな行列（構造化したＮＮ）となり（図5）、計算及びメモリ使用効率が悪くなる可能性が高い。



（ａ）大規模なＮＮ　　（ｂ）ランダムスパース行列

図５　構造化ＮＮ

そこで、本シミュレータでは、ノードから出ている有効なリンク（重みが非零）のみを対象として処理が行えるよう、各ノード毎に FanOut表 を作成した（図6）。

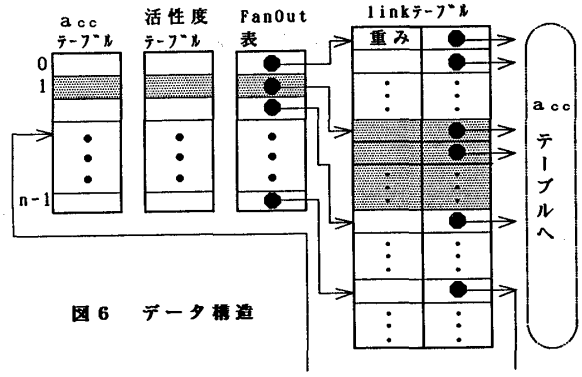


図６　データ構造

5．高速化手法

（1）ＮＮ分割

マルチプロセッサシステム上で情報を高速に処理するためには、負荷が各ＰＥに均等に分散していることが望ましい。従って、本シミュレータでは、ＮＮの性質を検討し、各ＰＥの負荷が均等になるように大規模なＮＮをＰＥ数の部分ＮＮに分割する。そして、それらを各ＰＥに割り当てている。

（2）プロセッサ間通信量の削減

ノード間のリンクが異なるＰＥ間にまたがる場合、リンク元のノードが存在するＰＥ内に、リンク先のノードに対応する仮想ノードを設ける。そして、一旦その仮想ノードへ投票、累算処理を行い、ＰＥ内の全てのノードが投票、累算フェーズを終了した時点で、他ＰＥにある実ノードへ投票、累算処理を行う（図7）。

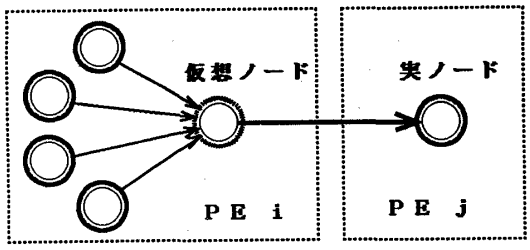


図７　ＰＥ間通信削減の概念図

6．動作評価

（1）評価用ＮＮ

本シミュレータは、構造化している大規模なＮＮを対象とするため、画像処理用にＨｏｕｇｈ変換のアルゴリズムをＮＮで実現し、これを用いてシミュレータの評価を行った。この評価用ＮＮ（以降Ｈ＿ＮＮと呼ぶ）は、ノード数に対するリンク数の割合が全結合型のそれに比べて非常に小さい。

Ｈ＿ＮＮでは、入力画像（直線画像）の角度検出を行う。そのＮＮ構造は、①画像空間、②エッジ空間、③直線のパラメータ（角度、法線）空間、④角度空間の４層構造で、それぞれの空間にノード群が対応している。そして、①に画像データが入力されると、①→②のリンクでエッジ検出、②→③でＨｏｕｇｈ変換、③→④で射影等の処理が行われ、④に結果として角度が出力される。

Ｈ＿ＮＮは、５６９６ノード、９３７９６リンクによって構成されている。このＨ＿ＮＮのリンク数は５６９６ノードを全結合したものの０．２９％程度（リンク利用率）である。

（2）評価結果

本シミュレータの性能として、ＰＥ数と、それに対応するシミュレーションの処理速度を計測し、それらの関係を評価した。図８に計測結果を示す。

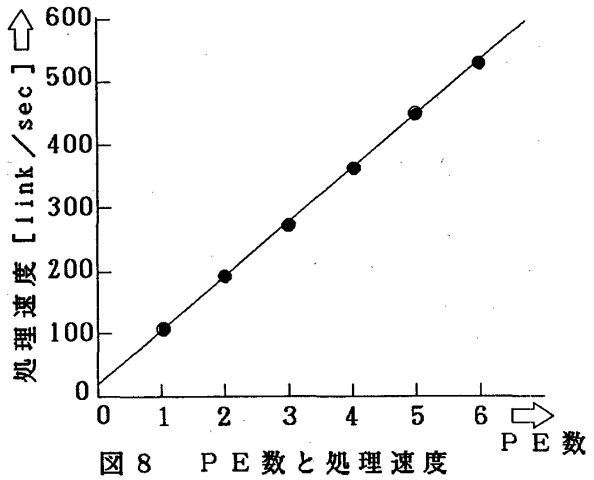


図８　ＰＥ数と処理速度

図８より、リンク利用率０．２９％のＨ＿ＮＮでのシミュレーションは、５ＰＥで１ＰＥの約４．２倍の処理速度が得られることがわかる。従って、本シミュレータは、構造化した（低リンク利用率な）ＮＮに対して、その並列動作特性を効率良く引き出すことが可能であることが確認できた。

7．おわりに

分散共有メモリ型マルチプロセッサ上に構築したＮＮシミュレータは、構造化している大規模なＮＮに対し有効であることが確認できた。

今後はＮＮのプログラミング環境として記述性の高いＮＮ記述言語が重要と考え、その開発を予定している。

参考文献

［１］松下ほか、：“並列シミュレーションマシン”、情報処理学会第３６回全国大会予稿集（3N-1）

［２］梶原ほか、：“並列ニューラルネットワークシミュレータ”、情報処理学会研究報告（88-ARC-71-12）