

Lisp並列処理マシン-EVLISマシンと ニューロエンジンの結合システム

3W-4

安井 裕, 平山一哉, 工藤幸一, 阿部一裕

(大阪大学・工学部)

1. はじめに

記号処理や人工知能の研究において、新たな可能性を見いだすことを目的として、本研究ではその第一段階として、我々の研究室で開発された Lisp並列処理マシン -EVLISマシン-^{1), 2), 3)} に、データフロープロセッサで構成されたニューロエンジンを接続することにより、“Neuro-Lisp”と名付けた Lisp環境から高速にニューラルコンピューティングが可能なシステムの試作を行った。本研究におけるシステム構成とその設計について述べ、このシステムを用いた実行結果の一例を示す。

2. EVLISマシンとニューロエンジンの結合

現実に存在する問題は様々であり、解法のアルゴリズムのはっきりしている問題もあれば、アルゴリズムを容易に記述できない問題もある。そのためアルゴリズムが明白となる問題に対しては従来のプログラミング言語で記述し、アルゴリズムの不明確な問題に対しては、可能であるならば、ニューラルネットを利用して問題を解決することを考える。

本研究では Lispにニューラルコンピューティングを取り入れ、Lispの関数として容易にニューラルネットを使用できることを目的とする。単にニューラルネットを使いやすくするだけでなく、学習機能の高速処理など、記号処理やAIに新たな可能性を示す意味も持つであろう。

2-1. マシンの構成

本研究では、複数台のニューロエンジンを結合させる基本となる Lispシステムとして、我々の研究室で Lisp並列処理マシンとして既に開発された EVLISマシンを設定した。EVLISマシン・ニューロエンジンの結合システムのハードウェア構成を、図1に示す。EVLISマシンは複数の EVAL-IIプロセッサにより並列処理を行なうことが可能となっており、現在3台の EVAL-IIプロセッサが実装されている。並列 Lispインタプリタ・コンパイラが実現されている。

2-1-1. EVAL-IIプロセッサ

EVAL-IIプロセッサはリスト処理向きのアーキテクチャを持ち、非常に高速にLispプログラムを実行することができる。

2-1-2. I/Oプロセッサとコンソール

I/Oプロセッサは EVLISマシンの入出力処理、及びシステムの管理を行なう。入出力関係では、EVAL-IIプロセッサとの間に介在して、フロッピーディスク、及びコンソールへの入出力を行なう。I/Oプロセッサとコンソールは RS-232Cの通信ケーブルで接続されており、現在 9600bpsの通信速度で交信している。システムの管理においては、各メモリの初期化、及び Lisp実行中のガーベジコレクションを EVAL-IIプロセッサとともに行なう。

2-2. EVLISマシンでのニューラルコンピューティングの実現方法

EVLISマシンに関してニューラルコンピューティングを実現する

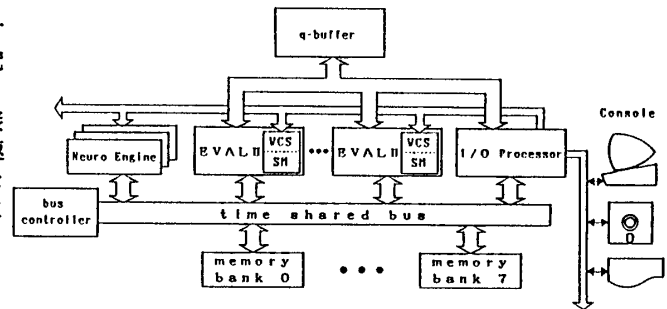


図1. EVLISマシン・ニューロエンジン結合システムのハードウェア構成

際、どのレベルで実現するか問題となる。それには、ニューラルネット・シミュレーションプログラムを Lispで記述することや、Lispの処理系にニューラルネット・シミュレーション関数をつけ加える、あるいは、EVLISマシンの EVAL-IIプロセッサの内部バスを経由してニューロエンジンを取り付ける、I/Oプロセッサにニューロエンジンを接続する、などの実現方法が考えられるが、本研究の目標とするものは内部バスを経由して接続する手段である。しかし、現在の EVLISマシンのシステム構成を大幅に変更することなくニューラルコンピューティングを可能とするため、EVLISマシンの入出力を行なう I/Oプロセッサを経由して外部にニューロエンジンを接続した。そして、ニューロエンジンに関する仕事を行なう際には I/Oプロセッサを介して通信する方法で研究の第一歩を進めることとした。

2-3. ニューロエンジン

現時点のシステムでは、本研究の外部のニューロエンジンとして、データフロープロセッサ(μPD7281)を4個搭載した日本電気製の ImPPボード⁴⁾(イメージ・パイプライン・プロセッサボード)を1台のニューロエンジンとして使用している。

後述するように、ハード的な大幅な改造を加えることなく実験を進めるため、EVLISマシンのコンソールとしての PC-9801を使用し、ImPPボードを実装している。そのため、ImPPボード上でニューラルコンピューティングを行なうにあたり、ソフトウェアとして“道真”⁴⁾のソースファイルを活用し、その一部を改変し利用している。そして、ニューラルコンピューティングの操作もコンソールへの通信回線を用いることができる。

3. システムの設計と実現方法

3-1. 結合方式

2-3.で述べたように、ニューラルコンピューティングはコンソールとして接続された PC-9801上の ImPPボードで行なう。そして ImPPボード上のメモリへのプログラム及びデータの入出力に関しては、ImPPボードに付属のライブラリ関数⁵⁾、及びサンプルプログラムを使用し、また、コンソールとしての機能をもたすため、通信機能をサポートするプログラムを作成・使用した。

3-2. ニューラル関数

ニューラルコンピューティングを行なうための関数群をニューラル関数と名付けており、その例を次に示す。

netinit 機能: ネットワークを指定した構造に設定し, 各ニューロンのウエイト, しきい値を乱数で初期化する. また ImPPボード上にシグモイド関数テーブルを作成する.

saveweight 機能: ImPPボード上の, ウエイト・しきい値テーブルをファイルとしてセーブする.

loadweight 機能: ウエイト・しきい値ファイルを読みとり, ImPPボード上にテーブルを作成する.

learn 機能: 入力信号と教師信号を与えて, 学習を行なう.

neuro 機能: 入力ボタンを与えて, ニューラルコンピューティングを実行する.

3-3. EVLISマシン上での実現

3-3-1. ニューロエンジンとの通信

本研究において, 本来ならば, EVLISマシンのメインプロセッサである EVAL-IIプロセッサにつながるバスに, 直接ニューロエンジンを接続するのが, 速度的には最も効果的であるが, EVLISマシンのシステムへの実装には多大の時間を必要とするため, とりあえずニューロエンジンとして用いる ImPPボードを, EVLISマシンのコンソールとして接続した PC-9801に実装し, ニューラルコンピューティングはコンソール内の機能として持たせることにした. 図2に, 本研究で使用したシステム構成の概念図を示す.

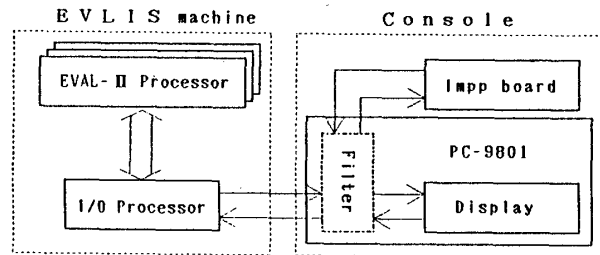


図2. 実験に使用したシステム構成の概念図

3-3-2. コンソール側での処理

コンソール側では, EVLISマシンから受け取った情報が, コンソールへの要求か, ニューラルコンピューティングの要求かを判断し, ニューラルコンピューティングの要求であれば, ImPPボードを用いて適当な処理を行ない, その結果を EVLISマシンに送る.

これらの機能を実現するため, ImPPボードの制御機能を持つ通信プログラムを作成した. このプログラムは通信を受け持つ部分と, ImPPボードの制御を受け持つ部分の2つのモジュールより成り立っている. ImPPボードの操作部分, 及び ImPPボード上でニューラルコンピューティングを行なうデータフロープログラムは, "道真"の一部を活用した.

4. 使用例と実行結果

4-1. 使用例

本研究で, EVLISマシンに新たに追加したシステムの機能の確認のため, ニューラル関数の使用例として, いくつかの入力信号と教師信号の組を与えておけば, 自動的にそれぞれの組に対して学習を進行させ, この操作を最初に設定された程度にネットワークが収束するまで繰り返すプログラム **autolearn** を作成した.

このプログラムの実行例として, 入力層のニューロンの数4, 出力層のニューロンの数16とし, 入力を4桁の二進数とみなし, そのデコードされた数に対応する番号のニューロンが反応する, ニューラルネットデコーダを様々なネットワーク構成について学習させた例についてその実行時間を示す.

4-2. 実行結果

実行結果として, 4-1.の実行例において, 4種のネットワーク構成について, 学習終了までに要した時間を表1に示す.

表1. 実行時間例

	ネットワーク構成	EVAL-II	ImPP	合計
①	4-20-16	2.23	4.95	7.18
②	4-50-16	3.20	14.18	17.18
③	4-20-20-20-16	4.37	29.01	33.01
④	4-50-50-50-16	5.15	68.46	73.61

表中の単位は, 秒である. ①, ②の構成は3層であり, 中間層のニューロンの個数は20, 50である. ③, ④は5層でありそれぞれの中間層のニューロンの個数は20, 50である. 各時間にはネットワークの初期化, 及び EVLISマシンとニューロエンジン間の通信時間は含まない.

4-3. 考察

この使用例は, 各入力信号と教師信号の組に対する学習回数など, 人間が経験的に定めていたパラメータを意識することなく, Lispのプログラムによって, 系の収束を見計らいながら自動的に学習を進行させることが可能であることの一例を示している. その他, 学習の順序などによりネットワークの形成に大きな差異が生じてくる場合での, より収束が早く効率のよい自動学習アルゴリズム等の諸問題における動的なパラメータの調整等を有機的に行なうことなどが期待できるものと考ええる.

現在のシステムでは本来コンソールのみに使用している通信回線を流用して接続したいわば仮のシステムであるが, 機能を獲得・評価するという立場では, 現システムの通信に要する時間は本質ではないと考えている.

5. まとめ

以上のように, Lispから簡単に高速なニューラルコンピューティングが可能な Neuro-Lispの環境を実現するシステムが作成できた. これにより, アルゴリズムのはっきりしている記号処理の問題に対しては, リスト処理を高速に行える Lispマシンが処理し, アルゴリズムの不明確な問題に対しては, Lisp側からニューロエンジン (あるいはデータフロープロセッサとしても) を調整し, 協調的な関係において処理を進行させるといったことが可能になる.

現在のシステムでは, EVAL-IIプロセッサ群とニューロエンジンは, 疎な結合状態にあるといえる. これをより密な結合状態にするためには, 図1に示したように, EVAL-IIプロセッサの内部バス等を介してニューロエンジンを直接結合させることが理想的である.

本研究の現状ではまだその緒についたばかりであるが, 並列 Lispマシンにおける Lispの世界からニューラルコンピューティングが可能なシステムが相当の規模で実用できる段階に達すれば, 様々な応用が考えられる. その暁には, 記号処理と, ニューラルコンピューティング (あるいはデータフロープロセッサとしても) の利点を有効に組み合わせることにより, さらに複雑な処理が可能となるであろうと期待する.

【参考文献】

- (1) 安井 裕 他: LISPでの並列処理における動特性とEVLISマシンの構成, 情報処理学会, 記号処理研究会資料10-4, 1979.
- (2) 前川 博俊 他: 高速LISPマシンとリスト処理プロセッサ EVAL II, 情報処理学会論文誌, Vol. 24, No. 5, pp. 683-695, 1983.
- (3) H. Yasui, et al.: Design of the shared memory system for multi-processor LISP machines and its implementation on the EVLIS machine, US/Japan workshop on parallel LISP, 1989.
- (4) 『Neuro-07』 ユーザズマニュアル, 日本電気市場開発(株), 1988.
- (5) ImPPボード用Cライブラリ [Ver. 2.0] リファレンスガイド, 日本電気(株), 1988.