異質な並列性の二層化

1L-2

和田 広志, 松山 泰男

茨城大学

1 はじめに

生データからの知識情報処理は，レベルの異なる二つの並列性を内に持っている．これは，広範なPDP型の並列分散処理に共通した特性であり "from raw data to predicates and action"[1][2] あるいは "symbiosis between fast signal processing and logical analysis"[3] と表現することができる．そして，Minskyの "Society of mind"[4] もこれと深く関連している．

本論文では，PDP型の並列分散処理を実行できる並列処理マシンのエミュレータを作成する．その特徴とするところは，"from raw data" あるいは "fast signal processing" に相当する部分には "fine-grained" なデータ並列性を用い，"predicates and action" あるいは "logical analysis" に当たる部分については "coarse-grained" な並列性を用いている点である．そして，上記の二つの異質な並列性を有機的に結合する事により，生データ処理と知識情報処理を一体化した環境を得ている．

さらにベンチマークとして，このシステムを用いてニューラルネットの学習法の一つである，逆伝播アルゴリズム [5] の実行を行う．

2 異質な並列性の二層化

与えられた生データに対する機械学習には，レベルの異なる二つの並列性が内在する．一つは，生データに対して比較的簡素で同種の演算を大量に施すという "fine-grained" なデータ並列性であり，もう一つは，その上層に位置しスケジューリング，制御，論理判断等における "coarse-grained" な手続きレベルの並列性である．

このような二つの異質な並列性を有機的に融合したシステムは，"from raw data ..." あるいは "symbiosis between ..." という考えから，PDP型の並列分散処理において非常に高い利用価値がある．

ここでは，この生データ処理と知識情報処理を一体化した環境を得るために，エミュレータを作成し利用する．全体のシステムには "*NeuroCube*" という通称を用いている．

3 NeuroCube

NeuroCube のエミュレータは，二つの部分の有機的な結合から成っている．一つは大規模データ並列性を扱うSIMD型の疎結合プロセッサ群であり，もう一つはそれらとの通信・制御，アルゴリズムのスケジューリング，抽出された特徴に対する論理判断を行う並列処理機構で，MIMDに相当する機能を有するものである．

An environment conprising two heterogeneous parallelisms.
Hiroshi Wada and Yasuo Matsuyama
Dept. of Computer and Information Sciences, Ibaraki University

NeuroCube の下位層に当たる疎結合プロセッサ群に求められる機能を有し，かつ簡潔なアーキテクチャとしては，ハイパーキューブによるコネクションマシン [6] があり，本エミュレータでも，これに類似したものを用いる．ただし本研究においては，ハイパーキューブマシンはあくまでも二重並列性のための一部分システムである．

一方，上位層には論理プログラミングのパラダイムを有し，かつ並列性を記述できる必要がある．これを満たすものの一つにICOTのKL1があり，これを利用することにする．従って，*NeuroCube* は次のモジュールから構成されている．

1. KL1処理系,
2. ハイパーキューブマシンのエミュレータ,
3. KL1処理系とハイパーキューブマシンのインタフェース.

以下に各部分を説明するが，1については文献 [7] があるので，そちらを参照されたい．

3.1 下位層

NeuroCube の下位層に当たる，2のハイパーキューブエミュレータはC言語で書かれており，これはさらに次の三つから成っている．

i. PE(処理エレメント),

ii. ルータ,

iii. マイクロコントローラ.

3.1.1 PE

一個のPE(処理エレメント) はALU，フラグレジスタ，及びローカルメモリからなる．ここで行われる演算は，全て1bit幅である．具体的な演算は，次のように行われる．ローカルメモリ上の二つの1bitデータ (AとB) 及びフラグレジスタ上の1bit(F) を使い，与えられた真理値表 (メモリ用とフラグ用の二つ) に従って，ローカルメモリ (A) と指定されたフラグ (F') を更新する．

3.1.2 ルータ

各PE間の通信は，iiのルータを介して行われる．一個のルータには，一個或いは数個のPEが接続されている．

一般に $N(=2^n)$ 個のPEを直接結合するには $\binom{n}{2}$ 本の回線を必要とするが，このような結合は当然のことながらコストが高くなりすぎ，またそれに見合うだけのメリットもない．従って，ここでは各ルータをnキューブ状に接続している．この場合結合は $n \cdot 2^{n-1}$ 本の回線ですみ，また形に対象性があるので，通信に当たっては両者間の相対的な位置関係のみを考慮すれば良いという利点がある．

3.1.3 マイクロコントローラ

iii のマイクロコントローラは，上位層との通信及び命令のバンド幅変換器として機能する．ここでの振る舞いは，外部から読み込んだマイクロコードにより規定される．従って，汎用性の高い命令群のみに限らず，より個々の問題に適した命令群を用いる事も可能である．

また，このマイクロコントローラを複数個使用することで，ハイパーキューブマシンをさらにハイパーキューブ結合したものとすることができる．たとえばルータが 2^n 個，マイクロコントローラが 2^m 個の場合，(n−m) キューブマシンを m キューブ結合したマシンとなる．ただし，マイクロコントローラの個数が変化した場合，上位層との通信路の数も変化してしまう．しかし，このことを上位層に意識させることは好ましくない．この不都合は，次のインタフェースにより吸収される．

3.2 インタフェース

上下位層間のインタフェースは，二重並列性の結合を司るという意味において *NeuroCube* の核心部分である．この部分の主要な働きは次の二つである．

i. 上位層で生成される複数のプロセス間における，ハイパーキューブマシンへのアクセスの競合を吸収する．

ii. 下位層における通信路数の変化を内部的に吸収する．

i により，各プロセス間の排他制御 (あるいは待ち合わせ) を行う必要がなくなる．これは次のように実現している．プロセス側ではハイパーキューブマシンへの命令にユニークなタグを付けてコマンドストリームに流す．インタフェース側では，そのタグにより各プロセスを認識しながら下位層への命令列のスケジューリングを行う．このとき，下位層からの返答を要する場合には，その返答に先ほどのタグをつけて上位層に送り返す．従って，返答を待っているプロセスは下から上がってくるストリーム中から，自分と同じタグを持つデータを拾い上げれば良い．

一方 ii については，上位層からのタグに従ってハイパーキューブマシンのスケジューリングを行うことで達成できる．この為に，インタフェースは常に下位層とのストリーム数やロードバランスを把握している必要がある．

このインタフェースにおいて最も考慮を要するのは，どこまでインテリジェントにするかという点である．例えば，下位層のタスクの切り替えによるオーバヘッドをどこまで減少させるかという事や，ハイパーキューブマシン内のデータ転送のオーバヘッドをどこまで考慮するのかという事などである．

現在のところ，この点についてはさらに考察を必要としている．

4 NeuroCube 上でのニューラルネットの学習

NeuroCube の目的は，プログラミングの効率化，すなわち二重並列性を互いに協力させて一つの問題を解かせる事ができる環境を用意する事である．そこで，実例としてニューラルネットの一学習法である逆伝播アルゴリズム (back-propagation algorithm)[5] を実行してみた．

この問題を KL1 のみで解く場合には，メモリの消費が激しいなどいくつかの問題が生じる [8]．この場合の大きな問題点は，"from raw data" あるいは "fast signal processing" に相当する部分にあり，この点を *NeuroCube* の下位層に位置させる事で二重並列性の分離による効果を調べる．

5 おわりに

生データにたいして最終的には知識情報処理を行いたい場合，レベルの異なる並列処理が存在するという二重並列化の考えのもとに，この両者が協力して一つの問題を解くことができる環境のエミュレータ "*NeuroCube*" を作成し，実際にニューラルネットの学習を行った．

計算自体の高速化に対する評価は，どの部分をハードウェア化するか，インタフェースにどの程度のインテリジェントを持たせるかなどのストラテジに依存する．しかしながら，下位層部分及び上下位層間の通信チャネルが高速化に最も寄与する事は明かである．

パターン情報処理を例にとるまでもなく，PDP 型の問題は今後さらに多様な展開を生み出しうる分野である．この *NeuroCube* は単なる専用機ではなく，この種の問題に十分適合し高い利用環境を提示するものである．

謝辞

PDSS(KL1) の使用を許可していただいた ICOT に感謝します．また，エミュレータ作成に当たって快く相談に応じてくれた茨城大学の黒澤 泰君に感謝します．

参考文献

[1] Matsuyama, Y. :Vector Quantization with Optimized Grouping and Parallel Distributed Processing, Neural Networks, Vol. 1,Suppl. 1, 1988.

[2] Matsuyama, Y. : Neuro-algorithms for Data Compression with New Generation Computing, Proc. IJCNN'89, 1989.

[3] Hopfield, J. : Neural Networks:Algorithms and Microhardware, Lecture Note 7, IJCNN'89, IEEE, 1989.

[4] Minsky, M. : The society of mind, Simon and Schuster, 1988.

[5] Rumelhart, D.E. 他 : Learning Internal Representations by Error Propagation, Parallel Distributed Processing, The MIT Press, 1986.

[6] Hillis, W.D. :The Connection Machine, The MIT Press, 1985.

[7] ICOT 第 4 研究室 : PDSS 使用手引き, 第 1.60 版, 1988.

[8] 奥村 晃 : GHC による実験的ネットワークプログラミング, ICOT, 1988.