アルゴリズム駆動ニューロコンピュータAN1 (2) 直交アーキテクチャ

3U-5

阿江忠　山下雅史　相原玲二　新田健一　藤田聡
(広島大学)

1. まえがき

本稿ではアルゴリズム駆動ニューロコンピュータ[1]の一実現形態である直交アーキテクチャについて述べる。また、その具体的な使用方法について、例を示して説明する。

2. 直交アーキテクチャ

AN1 (Algorithm-driven Neurocomputer —Prototype 1 —) では、アルゴリズム駆動アーキテクチャの具体的な形として、図1に示すような**直交アーキテクチャ** (Orthogonal Architecture)[2] をとる。直交アーキテクチャではNMの要素である基本ブロックを

$$\text{行集合}\quad NM_r = \{ F_{1r}, F_{2r}, \cdots, F_{pr} \}$$
$$\text{列集合}\quad NM_c = \{ F_{c1}, F_{c2}, \cdots, F_{cq} \}$$

に二分する (図1)。基本ブロックの大きさ (ニューロン数) は行ブロック q、列ブロック p と表されるが、AN1の場合 p,q とも最大15である†。NMの総ニューロン数は $2pq$ であるから、AN1では最大4500のニューロンを実装できるようになっている。

基本ブロックごとに評価関数 $\phi_{1r}, \cdots, \phi_{pr}, \phi_{c1}, \cdots, \phi_{cq}$ が付随しているので、NMはその値をそのままNPへ渡す。NMからNPへのインターフェースである**合成アレイ** (Synthesis Array, SAと略す)は、評価関数の引渡のほか、行と列のつくる格子点に対応する2つのニューロン間の演算 (出力) とニューロンへのトリガ (入力) の役割をする。すなわち、NPの側から見ると、まず、

$$read\ x_{ij}\ \ s.t.\ \ x_{ij} = x_{ij}^{(r)} \circ x_{ij}^{(c)}$$
{行と列の交点の演算}

(ただし、$x_{ij}^{(r)}$は行ブロックに属する i 行目の基本ブロックの左から j 番目のニューロンの出力であり、$x_{ij}^{(c)}$は列ブロックに属する j 列目の基本ブロックの上から i 番目のニューロンの出力であり、2つのニューロンは交点にある) によりNMからの値 x_{ij} (i=1, $\cdots$, p, j=1, $\cdots$, q) を読む (図2参照)。 $\circ$ は任意の2変数ブール演算 (and,or,...) を表す。むろん、単なる値の読出し

$$read\ x_{ij}^{(r)} \quad \{\text{行読出し}\}$$
$$read\ x_{ij}^{(c)} \quad \{\text{列読出し}\}$$

も独立に行える。次に、書込みはすべて独立に

$$write\ 0\ or\ 1\ to\ x_{ij}^{(r)} \quad \{\text{行書込み}\}$$
$$write\ 0\ or\ 1\ to\ x_{ji}^{(c)} \quad \{\text{列書込み}\}$$

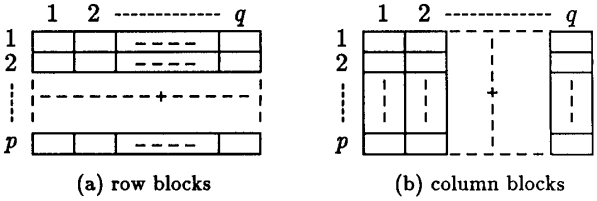
となる。なお、これらはビットごとにすべて並列に行うことができる。

NPは読み込んだ基本ブロックごとの評価関数値と各ニューロンの出力値を入力とし、アルゴリズムにもとづく処理を施したのち、その結果を各ニューロンへのトリガとして書き込む働きをする。アルゴリズムはペトリネット風に表現されることからわかるように、並列プログラムで実現されるので、NPも並列処理機構で実現されるのが望ましい。しかし、AN1ではとりあえずシングルプロセッサ上のソフトウェアで実現される。SAが出力のビット処理を並列に行うことで、SIMD処理を支援することもできる††。

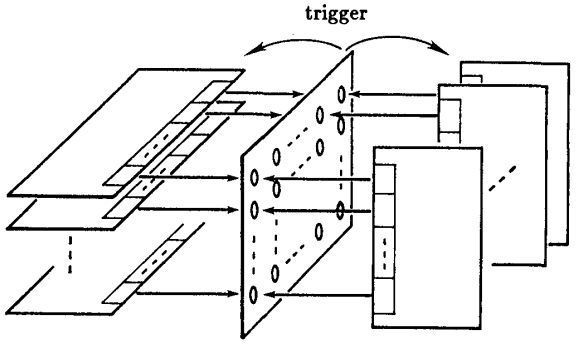
3. ニューラルメモリの設定法

アルゴリズム駆動ニューロコンピュータAN1は、NMをベースにしたアーキテクチャである点がノイマン型アーキテクチャと最も異なる点である。NMにはユーザが開発する基本ブロック F を付加することは可能とするが、基本的なものについては初めから用意しておくことも必要である。

NMの要素である F は、メモリと称しているが、アドレッシングが陽でないという意味で、レジスタとみることができる。通常のレジスタと異なるのは、「ビットベクトルどうしの間に状態遷移が存在し、不安定なベクトルは安定ベクトルへサイクルタイムの間に遷移する」と



(a) row blocks　(b) column blocks



p row blocks　synthesis array　q column blocks
(c) Orthogonal Architecture

図1　直交アーキテクチャ

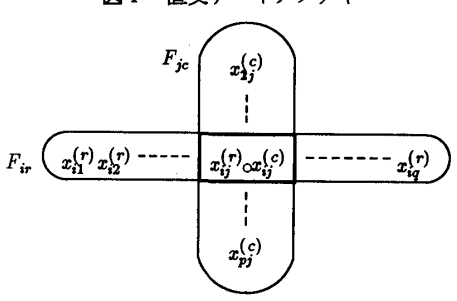


図2　交点の演算 $x_{ij}=x_{ij}^{(r)} \circ x_{ij}^{(c)}$

†) 本来は、目的にもよるが、少なくとも $10^2 \sim 10^3$ くらいが望ましい。

†† AN1はまだこの点は改良の余地が残されている。

AN1:Algorithm-driven Neurocomputer
(2)Orthogonal architecture
Tadashi AE,Masafumi YAMASHITA,Reiji AIBARA,
Ken-ichi NITTA,Satoshi FUJITA
Hiroshima University.

いう点である。したがって、アルゴリズム駆動アーキテクチャをニューロコンピュータの特徴を生かして実現するには、この状態遷移を起すような F の中から、有用なものを用いるのが適当である。

一般に、n ビットからなる通常のレジスタは 2^n 個の安定ベクトルをもつが、これらのベクトル間の内部遷移は存在しない（外部トリガでのみ遷移する）。ところで、F をHニューラルネットワークで構成した場合、ニューロン数が n ならば、安定ベクトルの個数の最大値は ${}_nC_k$ であることが古くから知られている[3]。このときのベクトルが重み一定となることに着目し、著者は、しきい値のみを変化させて、任意の一定の重みの安定ベクトルの集合をもつようにできることを示した[4]。この結果を、表1に掲げておく。表1において、${}_nC_i$ 個の安定ベクトルをもつとき、n ビットベクトル中 i 個の 0（すなわち、$n-i$ 個の 1）をもつ重み一定ベクトルが安定ベクトル集合になることを意味している。n=4 の場合を図3に示す。安定ベクトル以外は不安定であるが、状態遷移は束グラフで与えられ、連結している。重み一定符号は、これまでの計算機構成においても採用されてきた符号の一つであり、融合性の点では都合がよい。したがって、AN1では、

システム側からは、F として、k-out-of-n 符号を安定状態としてもつ ${}_nC_k$ 安定回路を供給する

という方針を取っている。アルゴリズム駆動アーキテクチャの特徴であるビルディングブロック構成ゆえ、**NM** は複数の F をもつ。F はニューロン数 m からなるから、**NM**は *k-out-of-m* 符号を一つのフィールドとした図4のような**非常に長いレジスタ** (Very Long Register) とみることもできる。なお、一つのフィールドは、${}_mC_k$ 個の有効ベクトル（安定ベクトル）を有するが、状態遷移は付随する制御プロセッサにより行われるし、遷移に必要な指示（例えば、テーブル）は制御プロセッサのもつ私有メモリに格納されるのが適当である。

表1　しきい値と安定ベクトルの個数の関係

Threshold θ	Number of State Vectors
$n-1<\theta$	${}_nC_0=1$
$n-1<\theta<n-1$	${}_nC_1=n$
⋮	⋮
$n-i-1<\theta<n-i$	${}_nC_i$ $(max.\ {}_nC_{\lfloor n/2 \rfloor})$
⋮	⋮
$0<\theta<1$	${}_nC_{n-1}=n$

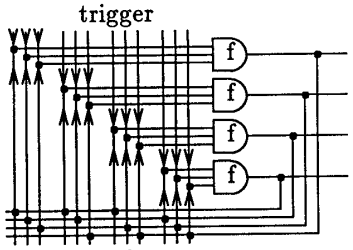


図3　${}_nC_k$ 安定回路の例

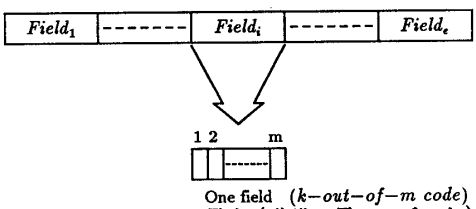


図4　NMの別の見方（非常に長いレジスタ）

4. AN1の応用例

直交アーキテクチャの特徴を生かしたAN1の応用方法について、あるスケジューリング問題を例として説明する。

以下のものが与えられる。

T：タスク集合、すなわち $T=\{\tau_1,\cdots,\tau_p\}$

$l(\tau_i)$：タスク τ_i $(i=1,\cdots,p)$ の長さ

$d(\tau_i)$：タスク τ_i $(i=1,\cdots,p)$ のデッドライン

ここでのスケジューリング問題とは、すべてのデッドラインを満足しながら T に対する r 個の資源スケジュール σ を見つけるという問題である。ただし、各タスクは単位時間ごとに分割することが可能であり (preemptive)、$\sigma(\tau_i)=(t_1,\cdots,t_{l(\tau_i)})$ と与えられる。ここで、t_j は単位時間に分割された j 番目のタスク開始時刻であり $t_j<t_{j+1}$ $(j=1,\cdots,l(\tau_i)-1)$ かつ $t_{l(\tau_i)}+1\le d(\tau_i)$ を満たす。

上記の問題を解くために、まず、i 番目の行ブロックを ${}_nC_{n-l(\tau_i)}$ 安定回路として設定し、さらに左から $d(\tau_i)$ 個を除くすべてのニューロンに、常に 0 を書込む。すなわち、

$$write\ \ 0\ to\ x_{ij}^{(r)} \quad (j>d(\tau_i))$$

とする。一方、列ブロックはすべて ${}_nC_{n-r}$ 安定回路に設定する。（図5）

行ブロックにおいて、1 であるニューロンのうち左から j 番目のものの位置を t_j と表わせば、$(t_1,t_2,\cdots,t_{l(\tau_i)})$ が求めるスケジュール $\sigma(\tau_i)$ となるには、すべての i,j $(1\le i\le p,\ 1\le j\le q)$ について

$$x_{ij}^{(r)}\le x_{ij}^{(c)}$$

でなければならない。そこで**NP**は

$$read\ \ x_{ij}=(\neg x_{ij}^{(r)})\vee x_{ij}^{(c)}$$

を実行し、その情報を基に**NM**に対してトリガ入力を行う。すべての i,j について $x_{ij}=1$ となればその時の $(t_1,t_2,\cdots,t_{l(\tau_i)})$ が求めるスケジュールとなり、

$$read\ \ x_{ij}^{(r)}$$

により読み出すことができる。

さらに、各タスクが分割できない (non-preemptive) 場合についても、上記で求めた結果を漸近解として真の解を求めることが可能である。

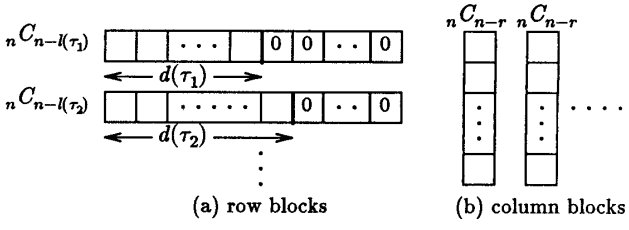


図5　各ブロックの設定例

文献

(1) 阿江他; "アルゴリズム駆動ニューロコンピュータAN1（1）基本構想", 情処全大（昭64-03）.

(2) T.Ae and R.Aibara; "A neural network for 3-D VLSI accelerater", Proc. International Workshop on VLSI for Artificial Intelligence, Oxford (July 1988).

(3) 大場; "相互に結ばれたしきい素子群の2,3の性質", 信学論(C), 51-C,7,pp.332 (昭43-07).

(4) T.Ae, H.Nagami and N.Yoshida; "On multistable Transistor circuits using threshold logic operation", Int.J.Electronics, **36**,6,pp.849-856 (1974).