離散時間ニューラルネットワークに基づく 連立1次方程式求解並列処理システム[†]

三浦 真樹*・亀山 充隆*・樋口 龍雄*

Parallel Processing System for Solution of Simultaneous Linear Equations Based on Discrete-Time Neural Network

Masaki MIURA*, Michitaka KAMEYAMA* and Tatsuo HIGUCHI*

This paper presents highly parallel multiprocessing system for solution of simultaneous linear equations. The parallel processing system can be naturally discussed based on the concept of discrete-time neural network. Using an algorithm similar to the steepest descent method the system always gives optimal solution of simultaneous linear equations since the state space has no local minima.

The corresponding multiprocessor system is efficiently constructed using ring interconnection network. The practical performance is evaluated by the implementation to demonstrate its advantage.

Key Words: discrete-time neural network, simultaneous linear equations, optimal solution, highly parallel processing system

1. ま え が き

ディジタル信号処理、ロボット制御などの実時間応用やVLSI設計、流体計算などのコンピュータシミュレーションの分野において、連立1次方程式の高速な求解が要求されている¹⁾。そのためには直列処理方式とは異なった、高並列ハードウェアによる求解が必要となる。従来、並列ハードウェアを構成する場合には既存のアルゴリズムを並列化可能な処理ごとに分割していたが、最初から並列化を考慮していないため処理のスケジューリングや制御が複雑になるなどの問題が生じ実際にハードウェア化した場合にアルゴリズムを効率よく並列化できるとは限らなかった。一方、高並

列ハードウェアによる高速な連立1次方程式の求解を行うためには、多数の均一な処理要素が規則的な動作を繰り返すことで求解を行うようなシステムを構成することが重要となる。

以上のような観点から、本論文では離散時間において多数の処理要素（セル）が結合した離散時間ニューラルネットワークの概念が連立1次方程式求解高並列ハードウェア構成に有用であることを明らかにしている²⁾。

従来、ニューラルネットワークのエネルギー関数と最適化問題の評価関数を対応させることで最適化問題を解きうるものが示されているが^{3), 4)}、セルがシグモイドなどの最終的に0, 1の2状態をとる非線形関数を用いて定義されるため、連立1次方程式を解く場合には問題が多峰性の最適化問題に変換されて最適解が保証されない⁵⁾。また連続時間で定義されているため、離散化には近似が含まれるという問題がある。

これに対し、本論文では最急降下法と類似のアルゴリズムに基づく離散時間ニューラルネットワークを定義すれば最適解を高並列に計算できることに着目した並列処理システムを提案している。セルの機能として線形動作主体の動作アルゴリズムをもつ離散時間ニューラルネットワークを定義することにより連立1次方程式の最適解を保証する並列処理アルゴリズムが自然に導出でき、同時に離散時間システムによる直接的実現を可能としている。特に、エネルギー計算に基づいて収束の加速を行っているが、エネルギー関数を変形することでセル間の並列性、規則性を保っている。本動作アルゴリズムは状態空間の勾配に基づいて極小化を行うという点では通常の最急降下法と結果的に共通点を有しているが、最初から離散時間ニューラルネットワークの概念を利用した高並列システムを前提とし、その処理要素であるセルの動作として記述される

[†] 第28回計測自動制御学会学術講演会で発表(1989・7)

* 東北大学工学部 仙台市青葉区荒巻字青葉

* Faculty of Engineering, Tohoku University, Sendai
(Received April 2, 1990)

(Revised August 22, 1990)

ため並列性が高く、高並列ハードウェアによる実現に適するという特長がある。

この並列処理アルゴリズムに基づき、1個のセルに1個のプロセッサを割り当ててマルチプロセッサシステムを構成する。実際のハードウェアではプロセッサ間の結合網が重要となるが、できるだけ少ない配線量でセルの演算能力を満足するという観点から、リング網によりセル間の結合を行う。また、セルが必要とする演算は加算、乗算、条件分岐など基本的なものであるため、1個のセルもきわめて小規模に実現可能である。実際に試作したマルチプロセッサシステムに対し、連立1次方程式求解の処理時間の評価を行い、高並列ハードウェアによる連立1次方程式の高速求解に有用であることを明らかにする。

2. 離散時間ニューラルネットワーク

セル i は時刻 k に状態として実数値 $V_i(k)$ をとり、他のセルと自分の状態を用いて1ステップ後に状態 $V_i(k+1)$ に遷移するものとする。Fig. 1 のようにセル i, j 間は結合係数 W_{ij} により結合され、セル i はバイアス入力 B_i をもつ。ここで離散時間ニューラルネットワークに対して時刻 k のエネルギー $E(k)$ をつぎのように定義する。

$$E(k) = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N W_{ij} V_i(k) V_j(k) - \sum_{i=1}^N B_i V_i(k) \quad (1)$$

一方、連立1次方程式 $\mathbf{Ax} = \mathbf{y}$ に対し、その評価関数である誤差関数 e を

$$e = \frac{1}{2} |\mathbf{Ax} - \mathbf{y}|^2 = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \left(\sum_{k=1}^N a_{ki} a_{kj} \right) x_i x_j - \sum_{i=1}^N \left(\sum_{k=1}^N a_{ki} y_k \right) x_i + \frac{1}{2} \sum_{i=1}^N y_i^2 \quad (2)$$

Fig. 1 Discrete-time neural network

と定める。 e は、 $\mathbf{x}$ が解ならば0、それ以外なら正の値をとるから極小値は1個である。ここで(1)式のエネルギー関数と(2)式の誤差関数を対応させれば、連立1次方程式求解をエネルギー極小化として離散時間ニューラルネットワークで直接取り扱うことができる。そのため(1)式と(2)式の変数間の対応をつぎのように決定する。

$$x_i = V_i(k) \quad (3)$$

これにより、連立1次方程式は、結合係数 W_{ij} とバイアス入力 B_i の形で離散時間ニューラルネットワーク上に、つぎのように与えられる。

$$W_{ij} = - \sum_{k=1}^N a_{ki} a_{kj}, \quad B_i = \sum_{k=1}^N a_{ki} y_k \quad (4)$$

このとき、(2)式の誤差関数の性質よりエネルギー極小化の操作で最小点を得ることができ、連立1次方程式の最適解が保証される。

したがってこのときエネルギーを極小化するようなセルの動作アルゴリズムを与えれば、連立1次方程式の求解を直接的に高並列ハードウェアによって実現できる。

3. セルの動作アルゴリズム

3.1 基本アルゴリズム

時刻 k に、あるセル m のみが状態遷移して $V_m(k)$ から $V_m(k+1)$ になったとする。このとき、エネルギーの変化量を $\Delta E(k)$ とすると

$$\begin{aligned} \Delta E(k) &= E(k+1) - E(k) \\ &= - \left\{ \sum_{j=1}^N W_{mj} V_j(k) + B_m \right\} \{V_m(k+1) - V_m(k)\} - \frac{1}{2} W_{mm} \{V_m(k+1) - V_m(k)\}^2 \end{aligned} \quad (5)$$

ここで、 a_m を定数として

$$V_m(k+1) - V_m(k) = \alpha_m \left\{ \sum_{j=1}^N W_{mj} V_j(k) + B_m \right\} \quad (6)$$

とおけば

$$\begin{aligned} \Delta E(k) &= - \left(1 + \frac{1}{2} W_{mm} \alpha_m \right) \alpha_m \\ &\quad \cdot \left\{ \sum_{j=1}^N W_{mj} V_j(k) + B_m \right\}^2 \end{aligned} \quad (7)$$

が得られる。 $\Delta E(k) < 0$ ならば1ステップの動作で必ずエネルギーが低下するから(7)式において

$$- \left(1 + \frac{1}{2} W_{mm} \alpha_m \right) \alpha_m < 0 \quad (8)$$

であればよい。したがって、(6)式より各セルが

$$V_m(k+1) = V_m(k) + \alpha_m \left\{ \sum_{j=1}^N W_{mj} V_j(k) + B_m \right\} \quad (9)$$

$$\left(0 < \alpha_m < -\frac{2}{W_{mm}} \right)$$

と状態遷移すればエネルギーの極小化が行われる。結果的にこれは、最急降下法と同様なものになっているが、セルの並列動作に基づくアルゴリズムを導出する上での基礎となる。すなわち、上述の議論では1ステップに1個のセルのみが状態遷移を行うと仮定しているが、全セルが同時に(9)式によって動作してもエネルギーの極小化が行われることが以下のように示される。

まず、2個のセルからなる2次元状態空間における例をFig. 2に示す。 P_0 は初期値、 P_1 と P_2 はおのどちらか一方のセルのみが状態遷移した場合を示している。エネルギー等高線とその内部は凸集合であるから、 P_0 、 P_1 、 P_2 を頂点とする空間はエネルギー等高線内部に入る。すなわち、その空間内部は P_0 よりエネルギーの低い状態となる。これを N 次元に拡張すれば、エネルギーの極小化が示せる。

(i) N 次元空間における任意の点を P_0 とする。

(ii) エネルギーの極小点が1個であるから、 P_0 とエネルギーの等しい空間とその内部は凸集合をなす。

(iii) P_0 に対し、 $\alpha_m = -2/W_{mm}$ として各次元方向に(9)式を適用して得られる点を $P_1, P_2, \dots, P_N$ とすると、それらは P_0 と同じエネルギーをとる(・(7)式より $\alpha_m = -2/W_{mm}$ のとき $E(k) = 0$)。

(iv) P_0 および $P_1, P_2, \dots, P_N$ を頂点とする空間内部は P_0 とエネルギーの等しい空間内部に含まれる。すなわち P_0 よりエネルギーの低い空間になる。

(v) したがって P_0 から $P_1, P_2, \dots, P_N$ までの、各次元方向の変化量をそれぞれ適当に選べば、 P_0 よりもエネルギーの低い点を得られる。□

以上より、セルの基本アルゴリズムとして次式を得る。

Fig. 2 Two-dimensional state space

$$V_i(k+1) = V_i(k) + \alpha_i \left\{ \sum_{j=1}^N W_{ij} V_j(k) + B_i \right\} \quad (10)$$

$$\left(0 < \alpha_i < -\frac{2}{NW_{ii}} \right)$$

3.2 動作アルゴリズムの高速化

(10)式で示されるセルの動作アルゴリズムでは、与えられた連立1次方程式によっては解への収束が極端に遅くなることがある。ここでは、このような場合にも高速な収束を行うため加速の項を加える。しかし、単に加速の項を加えるだけではゆきすぎが生じ、かえって収束が遅くなる場合が多い。したがって常にエネルギーの計算を行い、状態遷移の結果 $\Delta E(k) \geq 0$ となったら一度加速を停止することにより強制的に $\Delta E(k) < 0$ として収束の高速化を図っている。これが収束することはつぎの(i)から(iii)のように示される。

(i) 時刻 k における状態を用いて状態 $V_i(k+1)$ を計算する。

(ii) 時刻 $(k-1)$ で計算されるエネルギー変化量が $\Delta E(k) < 0$ ならば加速項を増大させる。

(iii) $\Delta E(k) \geq 0$ ならば、加速を停止し、状態を $V_i(k)$ に戻す。□

すなわち、常に

$$\Delta E(k+1) \leq 0$$

が成り立ち、エネルギーの低下しない状態遷移が連続して起きることはない。したがってエネルギーの極小化が行われる。

この方法において、状態遷移に要するエネルギー $E(k)$ を定義通りに計算すると $O(N^2)$ の乗算が必要となり1ステップの状態遷移に要する時間が大幅に増加してしまう。また、ハードウェア化を考えた場合エネルギーの計算をある1個のセルに割り当てることになるが、1個のセルのみ計算量が大きく異なると高並列ハードウェアとしての実現が困難となる。したがってエネルギー計算の計算量を小さくする必要がある。

まずエネルギー関数をつぎのように変形する。

$$\begin{aligned} E(k) &= -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N W_{ij} V_i(k) V_j(k) \\ &\quad - \sum_{i=1}^N B_i V_i(k) \\ &= -\frac{1}{2} \sum_{i=1}^N \left\{ \left\{ \sum_{j=1}^N W_{ij} V_j(k) + B_i \right\} V_i(k) \right\} \end{aligned} \quad (11)$$

ここで

$$S_i(k) = \sum_{j=1}^N W_{ij} V_j(k) + B_i \quad (12)$$

とおくと

Fig. 3 Algorithm for reduction of computing time

Fig. 4 Simulation using the algorithm of Fig. 3

$$E(k) = -\frac{1}{2} \sum_{i=1}^N \{S_i(k) + B_i\} V_i(k) \quad (13)$$

となり N 回の乗算でエネルギーが計算できる。 $S_i(k)$ は、各セルが状態遷移をするときに計算されているから、単にセル間の通信で得ることができる。

このアルゴリズムを Fig. 3 に示す。 $v_i(k)$ は加速を表す項、 β は $v_i(k)$ の変化量を定めるパラメータであり、ここでは $\beta=0.9$ である。エネルギー計算、セルの状態遷移、ともに乗算回数が N 回であり同規模のハードウェアで並列に実行可能である。したがって均一なセルの結合により実現可能となり、高並列ハードウェアに適した構成となっている。実際に連立 1 次方程式を解く場合の離散時間ニューラルネットワークの動作シミュレーションを 8 元連立 1 次方程式を例として行った結果を Fig. 4 に示す。

3.3 収束性に関するシミュレーション

本動作アルゴリズムについて、従来の解法アルゴリズムの一つである JOR (Jacobi Overrelaxation) アルゴリズムと比較しながら収束性についてシミュレー

ションを行う。JOR アルゴリズムは、従来よく用いられる SOR (Successive Overrelaxation) アルゴリズムの基礎であり、次式で表わされる。

$$x_i(k+1) = (1-\gamma)x_i(k) - \frac{\gamma}{a_{ii}} \left\{ \sum_{j=1}^N a_{ij}x_j(k) - b_j \right\} \quad (14)$$

直列処理で実現される場合には SOR アルゴリズムのほうが収束が速いが、並列性の点では JOR アルゴリズムのほうが有利である。

セルの動作アルゴリズムは、基本的には線形の反復過程であるから、収束性の尺度として行列 αW のスペクトル半径 $\rho(\alpha W)$ を用いることができ、これは次式のように与えられる。

$$\rho(\alpha W) = \max_i |\lambda_i(\alpha W)| \quad (15)$$

ここで α は対角要素が α_{ii} の対角行列であり、 $\lambda_i(\alpha W)$ は αW の i 番目の固有値を表す。本動作アルゴリズムが収束するとき明らかに

$$0 < \rho(\alpha W) < 1 \quad (16)$$

であり、1 に近いほど収束しにくい問題であることを表わしている。

一般に $\rho(\alpha W)$ の計算には固有値の計算が必要であるが、 αW のマトリクスノルム $\|\alpha W\|$ に対し

$$\rho(\alpha W) \leq \|\alpha W\| = \max_i \left| \sum_{j=1}^N (\alpha W)_{ij} \right| \quad (17)$$

が成り立ち、これは容易に計算できるので収束性の尺度として近似的に $\|\alpha W\|$ を用い

$$0 < \|\alpha W\| < 1 \quad (18)$$

となるように連立 1 次方程式の A を設定する。このとき収束までに要する本動作アルゴリズムと JOR アルゴリズムの反復回数のシミュレーション結果を Fig. 5 に示す。本動作アルゴリズムは強制的な加速の制御を行うため反復回数にばらつきがあるが、収束しにくい問題になるほど本動作アルゴリズムのほうが高速に

Fig. 5 Convergence in the proposed algorithm and JOR algorithm

なる。これは、状態空間の性質から、収束しにくい問題になるほど JOR アルゴリズムは1ステップの状態遷移の変化量が小さくなるのに対して、本動作アルゴリズムでは加速を行うことで1ステップの変化量を次第に増加させているからである。

4. マルチプロセッサシステムによる実現

4.1 リング網によるセルの結合

実際のハードウェア構成として、1個のセルに1個のマイクロプロセッサを割り当ててマルチプロセッサシステムの構成を行う。離散時間ニューラルネットワークの定義では、セル間が完全網で結合されていると仮定しているが、実際のハードウェアでは配線量のオーダーが $O(N^2)$ と膨大になるため実現が困難である。また、各セルを1個のマイクロプロセッサで構成する場合、セルは1回に1個のデータしか処理できないため、結合網はプロセッサ1回の基本演算(加算, 乗算)につき1個のデータの割合でデータ転送できれば十分である。したがって、できるだけ少ない配線量でセルの演算能力を満足し、さらに配線の局所性、通信の容易性、システムの拡張性などを実現するという観点から、リング網に基づくマルチプロセッサシステムが適切である。このとき、データ転送を Fig. 6 のように行えば N 回でセル間の通信が終了する。

4.2 マルチプロセッサシステムの構成

1個のセルは、以下の部分からなる。

CPU: 動作アルゴリズムの実行と通信の制御。

メモリー: 結合係数 W_{ij} とバイアス入力 B_i の保存、演算ルーチンおよび各種ワークエリア。

I/O ポート: セル間の通信。

原理実験であり絶対的処理速度は問題としないことから、CPU には8ビットの Z80 を使用し、多精度の演算などはソフトウェアで行う。また、通信方式は I/O ポートを介したハンドシェイク方式として、通信

Fig. 6 Data transfer scheme between cells

Fig. 7 Multiprocessor system using ring network

Photo. 1 Implemented multiprocessor system

の制御やセルの増設を容易にしている。

以上に基づいたマルチプロセッサシステムの構成を Fig. 7 に、実際に8個のセルを結合したものを Photo. 1 に示す。ホストのパーソナルコンピュータもリング網に組み入れ、システムの動作中の状態のモニタや制御を容易にしている。

このようにマルチプロセッサシステムを構成することにより、配線および通信は隣接セル間のみに局所化され、拡張する場合も単にセルを増設すればよい。また、その規則性から専用プロセッサ化も容易である。

4.3 動作確認と処理速度の評価

試作マルチプロセッサシステムの動作を確認するため、実際に連立1次方程式の求解を行う。データ表現は32ビット固定小数点表現とし、比較のため結合係数行列 W は、シミュレーションのときと同じものを与える。そのときの収束までの反復回数を Fig. 8 に示す。傾向としてはシミュレーションと一致するが、反復回数自体はやや増加している。これは、データ表現が固定小数点表現であるため浮動小数点表現に比べてダイナミックレンジが小さく、計算途中での量子化誤差が大きいためと考えられる。

つぎに処理速度に関する評価を行う。一般の N 次連立1次方程式を解く場合、本マルチプロセッサシステムのプロセッサ数は $(N+1)$ 個となる。その場合、隣接セルと1回通信を行うのに要する時間を t_c 、1回

Fig. 8 Simulation using the multiprocessor system

の積和演算に要する時間を t_i , 1回の加算, 乗算に要する時間をそれぞれ t_a , t_m とし

$$t_i = N(t_a + t_m) \quad (19)$$

とすると単一プロセッサシステムとの処理速度比 r_N は

$$r_N = \frac{(N+1)t_i}{(N+1)t_c + t_i} = \frac{N}{\left(1 + \frac{1}{N}\right)\frac{t_c}{t_a + t_m} + 1} \quad (20)$$

となる。ここで t_c , t_a , t_m は N によらず一定であるから, プロセッサ数 N にほぼ比例して処理速度比が向上することが示される。実際の測定の結果, 本システムについては $t_i = 89 \text{ ms}$, $t_c = 11.2 \text{ ms}$ である。これより, 単一プロセッサシステムによる処理速度比 r_8 はつぎのように計算される。

$$r_8 = \frac{89 \times 9}{101 + 89} = 4.22 \quad (21)$$

理想的な処理速度比 $r_8 = 9$ になっていないが, これは本マルチプロセッサシステムにおいて, セル間の通信を I/O ポートを介してソフトウェアで処理しているため $t_c > t_a + t_m$ となることが原因である。したがって, セル間の通信をハードウェアによって処理し, $t_c < t_a + t_m$ とすれば容易に理想的な処理速度比に近づけることが可能である。

5. む す び

本論文では, 連立1次方程式の求解を行うための離散時間ニューラルネットワークの概念に基づく高並列ハードウェアを提案した。しかも, セル動作アルゴリズムを線形要素を主体として構成することで, 最適解の保証が与えられる。

本並列処理アルゴリズムに基づき, リング網を用いたマルチプロセッサシステムの構成を提案した。最初から離散時間ニューラルネットワークにおけるセルの並列動作を前提として動作アルゴリズムを導出しているため, 並列性, 規則性の高いハードウェアが構成可能であることを明らかにした。今後は, 試作したマルチプロセッサシステムと, 他の解法に基づく並列処理ハードウェアとの比較を, 特に高速性, ハードウェア規模などの観点から行っていきたい。また, 本離散時間ニューラルネットワークの概念を利用した, より広い最適化問題への拡張や学習可能性についても考察する価値がある。

参 考 文 献

- 1) 羽根邦夫: スーパーコンピュータとデバイスシミュレーション, 情報処理, **28**-11, 1460/1468 (1987)
- 2) 三浦, 亀山, 樋口: 離散時間ニューラルネットワークに基づく連立一次方程式求解並列処理システム, 電子情報通信学会技術研究報告, **89**-312, ICD-147, 23/30 (1989)
- 3) J.J. Hopfield: Neural Network and Physical Systems with Emergent Collective Computational Abilities, Proc. Natl. Acad. Sci. U.S.A., **79**, 2554/2558 (1982)
- 4) J.J. Hopfield: Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons, Proc. Natl. Acad. Sci. U.S.A., **81**, 3088/3092 (1984)
- 5) M. Takeda and J.W. Goodman: Neural Network for Computation: Number Representation and Programming Complexity, Appl. Opt., **25**-18, 3033/3046 (1986)