ハイパフォーマンス
コンピューティング　82−35
（2000．8．5）

Information-propagate problem in neuron-computer

Hanxi ZHU, Tomoo AOYAMA*, Umpei NAGASHIMA**
*The Faculty of Engineering, Miyazaki University
Gakuen Kibanadai-nishi 1-1, Miyazaki 889-2192, Japan
E-mail: aoyama_t@cc.miyazaki-u.ac.jp
**Computational Science, National Institute for Advanced Interdisciplinary Research
1-1-4 Higashi, Tsukuba, Ibaraki 305-8562, Japan
E-mail: umpei@nair.go.jp

We studied a neuron-computer that was constructed of plural kinds of neural networks. The architecture is designed to avoid von Neumann's bottleneck and to evaluate directly fuzzy logical formula that can't represent by logical AND, OR, NOT operators. The computer has two kinds of memories; one is the association memory that requires learning, and another is the stack memory. We publish the stack's construction techniques on use of multi-layer neural networks with feedback loops. The techniques concern with signal-propagations in the networks. In order to suppress accumulation of the error on the signals, we proposed a neural network including step functions; and we examined the properties of a 2W-multi-valued memory.

ニューロンコンピュータの情報伝播問題

朱 含希、青山智夫*、長嶋雲兵**

*宮崎大学工学部電気電子工学科
**通商産業省工業技術院　産業技術融合領域研究所

我々は複数のニューラルネットワークから構成されるニューロンコンピュータを研究してきた。このアーキテクチャはノイマンのボトルネックを避けるために考案されたもので、論理積和否定演算子によって表現不能なファジイ論理式を直接評価できる。ニューロンコンピュータは二種類の記憶を持つ。一つは学習を要する連想記憶、他はスタックメモリである。我々は階層型ニューラルネットワークと帰還ループを使った、このスタックの構成手法を示す。その技術はネットワーク内部の信号伝播に関する。信号値の誤差累積を抑止するため、階段関数を含むニューラルネットワークを提案し、2語多値メモリを評価した。

1. Introduction

Many researchers have studied architectures for non-Neumann's computers because of escaping its bottleneck. To avoid the bottleneck, someone began to develop neuron-based computers. The computers have neurons and their connections that are constructed of the learning. They have information processing facilities, and at the same time, they are like as simplified brains to make inference.

We consider the neuron-computer to be constructed of plural kinds of neural networks [1-5]. The logical construct is similar to a computer. The neural networks correspond to resources in the computer, however, since neural networks have a learning facility, we don't design the networks, but make them in the learning on use of truth tables.

There are logic formulas that can't be represented in the binary-logic operations. They are as for multi-value or fuzzy logics, and are represented by the truth tables. If the neuron-computer is adopted, they can be executed in their networks. Therefore, the neuron-computer simulates more functions than that of computers based on the binary-logic.

2. Normal form of the logical function

2.1 Neural networks for disjunctive normal form

We discuss functions constructed of multi-layer neural networks. One network can express any truth table on binary logic. Therefore, plural networks can represent any computer resources. Any binary-logic formula has disjunctive and conjunctive normal forms. The disjunctive normal form is a logical-OR of minimum product terms. The conjunctive normal form is a logical-AND of maximum sum terms.

We show a neural network expression for the disjunctive normal form in figure 1. The OR/AND parts are replaced by neural networks. Therefore, this is a network constructed of the neural networks.

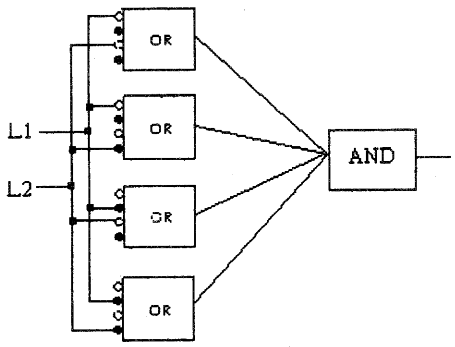

Figure 1. *Disjunctive normal form expressed by a network*

If we found a method to determine the connections between the neural networks by learning data only, any binary-logic unit can be developed by the learning method. A key point of the determination is the differential for input/output values of a neural network. That is already published [6]. For three- layer network, the expression is that.

$$\partial O / \partial X = Vf'Wg'$$

O is output of neural network. X is input data for neural network. V is matrix of the connections between neurons on the 1st and 2nd layers. "f'" is the differential of simulation function for neurons on 2nd layer. W is matrix between 2nd and 3rd layer. "g" is the function on 3rd layer.

2.2 Neural network representations of computer memories

Neural networks have storage function originally. The function requires a learning that is rather long time processing. Such storage isn't suitable for stacking. Stack memories are indispensable in information processing because of work-storages. Information must not be stored in the memories by a learning state; where the memorizing must be a flash. Even if the processor is a neuron-computer, the storages are necessary. We must develop the storages in neural networks.

A stack memory would be constructed by a feedback loop in networks. We developed D-flip-flop, FIFO, and vector register by using multi-layer neural networks with a feedback loop. The methods have published already, where we showed a truth table for a vector register [2,3]. We can use the table as leaning data for BP algorithm. The vector register in super computers can be constructed by neural network only.

The idea is correct in logical world, certainly. However, we must pay very careful attention to propagation of information. In large logic unit, information of neurons propagates through many connection paths among neurons. If the information were transferred as analogue value, propagation of exact information is very difficult. We found the difficulty in feedback loops of the stack memory. We observed that the information was diminishing quickly in the stack. We are sure that information maintenance logic is necessary in neuron systems. In this system, since the information corresponds with the amplitude of output on neurons; the maintenance logic must keep the amplitude.

3. Simulations of stacking memories

3.1 Binary memory

When a resource is constructed by a truth table, the important point is to minimize size of the table. A practical method is introducing of a restriction; that is, all variables have one value at that instant. Invalid or uncertain signal is regarded as a value. The restriction suppresses the increasing of combinations for variables having plural values. On the other hand, the number of the logic value is added by one. That is, nth logic is represented by (n+1)th logic [5]. The restriction is not so heavy load on the neuro-computing. We show a truth table for a memory in table 1.

Table 1. *Truth table for a memory (binary, 2W)*

WE	ADR	Din	M0(in)	M1(in)	Out	M0(out)	M1(out)
1	0	{0,1}	{0,1}	{0,1}	*	Din	M1(in)
	1	{0,1}	{0,1}	{0,1}	*	M0(in)	Din
0	0	*	{0,1}	{0,1}	M0(in)	M0(in)	M1(in)
	1	*	{0,1}	{0,1}	M1(in)	M0(in)	M1(in)

In the table 1, "*" is invalid value, however, it must be defined in calculation of neural network. We rewrote the table as following.

Table 2. *Learning data for a neural network*

Input data					*teaching data*		
WE	ADR	Din	M0(in)	M1(in)	Out	M0(out)	M1(out)
1	0.5	{0.5,1}	{0.5,1}	{0.5,1}	0	Din	M1(in)
	1	{0.5,1}	{0.5,1}	{0.5,1}	0	M0(in)	Din
0.5	0.5	0	{0.5,1}	{0.5,1}	M0(in)	M0(in)	M1(in)
	1	0	{0.5,1}	{0.5,1}	M1(in)	M0(in)	M1(in)

We made a neural network memorize table 2, where the numbers of neurons in the network were 6, 8, and 3. The number of iterations is 10,001, and the maximum error is 0.025 at the end of iterations. After the learning, the network functions as a two-words-memory. The simulation is listed in figure 2.

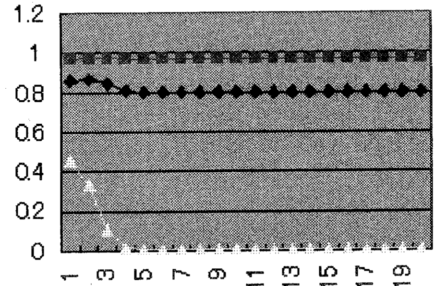

Figure 2. *A simulated result of 2W-memory*

Cells in the memory are initialized by 0.5, and at first, a value "1" is written in M0, that is a cell's name. M1-cell is not written. After the setting, the M0 is read by 20 times.

Upper line is the values for M0, middle line is outputs, and lower line is values of M1-cell. The outputs are converged on near 0.8, that is recognized as "1" in case of the binary. ("1" is between 0.75 and 1.0, and "0" is 0.75 to 0.25, and invalid is 0.25 to 0.0.) Since the allowance amplitude is 0.25 on binary logic, memorized information is kept. However, it must be emphasized that even if memorized value is "1", an output from the memory is 0.8. The value in M1-cell, that is not operated, becomes invalid soon. We believe that the memory is unstable.

3.2 Multi-value memory

On the neuron-computer, multi-value logics can be introduced easily. Therefore, in order to improve mounting density on circuits, we try to adopt multi-value logic actively. We believe that the multi-value must be the four. Because, the following relation is defined for not-operator "~": {~0,~a,~b,~1}={1,b,a,0}. The four-value logic is similar to the binary than the three. We examined a memory of the four-value logic. The amplitude is five kinds those are {0,1/4,2/4,3/4,1}, where "0" is the invalid.

At first, values "3/4" and "2/4" are written in M0 and M1-cells, and a fetch-sequence is repeated by four times. The sequence is to read M0-cell by 10 times and to do M1 by same times. The test is for stability check of memorized information, and the result is listed in figure 3.

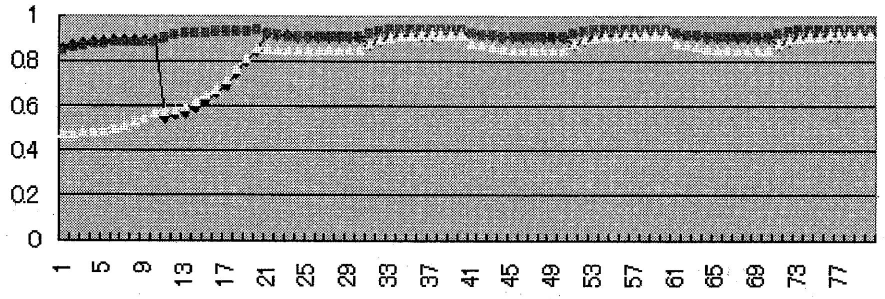

Figure 3. *Simulation result for 4-value logic memory: Fetch tests for 1/3 and 2/3 values*

In the figure 3, the values in the neural network are 1, 0.75, 0.5, 0.25, and 0. The logic value 2/3 is 0.75, and 1/3 is 0.5. At initial states, 2/3 and 1/3 are memorized certainly, however with the progress of the time, the value of 2/3 is increased and exceeds the limit of 0.875(=0.75+0.25/2=1-0.25/2). It is similar in case of 1/3. Since the allowance is narrow on four-value logic, memorized information is not kept. The memory cannot be used.

4. Neuron layer with step-function

Since processing in the neural networks is analogous, accumulation of the error is inevitable. Therefore, compensation units are necessary in neural networks; it must operate while the error is small. We considered AD/DA converter units at first. However, since the units enlarge the logic size of neuron-computer.

The neuron functions on second or third layer are sigmoid functions usually, but the functions on first layer are null (or through) function; and information on the layer is not processed. We wish to introduce step functions in the layer, that are defined as,

$f(x)=j*d$,

$B+j*d<x<B+(j+1)*d$,

Where,

$j=\{0,1,2,....n-2\}$, $d=1/(n-1)$, and $B=d/2$.

We examined the four-value memory introduced the step functions at M0 and M1-cell's neurons on the first layer. The examination is listed in figure 4.

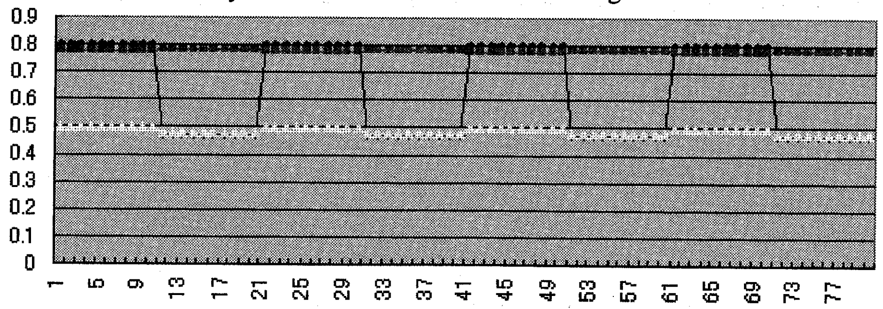

Figure 4. *Simulation result for 4-value logic memory introduced the step-functions: Fetch tests for 1/3 and 2/3 values*

The effect of introducing the step-functions is remarkable. The information for 2/3 and 1/3 values is kept completely. We are sure that multi-value memories get practical usage.

5. Conclusion

Stacks are indispensable in information processing. Even if the processor is a neuron-computer, they are necessary. We developed a stack memory that was constructed by neural networks for neuron-computers. The stack memory is constructed by introducing a feedback loop.

We developed two-words four-valued memory by using a multi-layer neural network and back propagation algorithm. Since the neural network converts any vector into another; if a truth table is defined, and even if the table is multi-valued one; any computer resource is constructed. The idea is correct in logical world; however, it doesn't assure propagation of information. In large logic unit, information of neurons propagates through many connection paths among neurons. If the information is transferred as an analogue value, exact propagation is difficult. We showed the difficulty on the stack memory, and did that information maintenance logic was necessary in neuron systems. We introduced new three-layer neural network where neuron functions on first layer were represented by step functions.

We examined the four-value memory introduced the step functions, and the proposed technique gives reasonable results.

Acknowledgement

The research was financially supported by the Sasagawa Scientific Research Grant from the Japan Science Society.

References

[1] . Tomoo AOYAMA, "Large scale multi layer neural networks", 99-HPC-76(IPSJ SIG Notes,1999.5.14).

[2] Hanxi ZHU, Kenji KOBATA, Housei UEDA, and Tomoo AOYAMA, "Functional memories constructed of neural network", 99-ARC-134(IPSJ SIG Notes,1999.8.3).

[3] Hanxi ZHU, Tomoo AOYAMA, and Ikuo YOSHIHARA, "Functional Memories constructed of neural networks", Proceedings of 14th KACC, E-210-213, (1999.10.14).

[4] Tomoo AOYAMA, Hanxi ZHU, Tomoki TESHIMA, and Ikuo YOSHIHARA, "Simulations of construction learning as for neuron-computer resources", Proc. of 5th. Int. Symp. on AROB, pp.825-828,(2000.1.26).

[5] Hanxi ZHU and Tomoo Aoyama, "Memory functions in loops including multi-layer neural-network:FIFO and Vector register", Proc. of 4th HPC-ASIA 2000 (Beijing; 2000.5.14--17).

[6] Tomoo AOYAMA, Hanxi ZHU, and Ikuo YOSHIHARA, "Forecasting of the chaos by iterations including multi-layer neural-network", Proc. of IJCNN'2000 (Como, Italy; 2000.7.24--27).