計算機アーキテクチャ 110－22
設 計 自 動 化 73－22
（1995．1．20）

ＯＮＢＡＭ：オブジェクトモデルニューロンによる能動メモリ

阿江　忠、豊崎　剛、福本　光、酒居　敬一

広島大学工学部

オブジェクトモデルニューロンをもとにしたニューラルネットＯＮＢＡＭ（Objective-Neuron-Based Active Memory）を提案し、そのプロトタイプチップをＨＤＬにより設計した。ＯＮＢＡＭは位相をもつ能動メモリであり、位相空間を（リアルではないが）等価的に実現し、その上での内容参照を柔軟に行なうことができるという特徴をもつ。すなわち、ＯＮＢＡＭはメモリベースのＡＩアーキテクチャ実現において重要な役割を果たすものである。

ONBAM:An Objective-Neuron-Based Active Memory

Tadashi Ae, Tsuyoshi Toyosaki, Hikaru Fukumoto, Keiichi Sakai

Faculty of Engineering, Hiroshima University

We propose an objective-neuron-based active memory, ONBAM, and design it by HDL. The ONBAM is an active topological memory, which is a memory with equivalently a topological space of data and a flexibly content-addresable function, which plays an important role of the memory-based AI architecture.

1. Introduction

The memory-based architecture is contrasted with the processor-based architecture, which is known as the von Neumann architecture. The design of conceptual level is really mapped, in the physical level, to the memory in the memory-based architecture, while it is done to the processor in the processor-based architecture, i.e., the conventional processor. In the memory-based architecture the memory must be *active* because it plays a role of several functions. The active AI memory is a kind of functional memories, but it is a new type that works in the topological space, because it is constructed by an extension of neural networks.

The active AI memory is an important part of NFM (New Functional Machine), the memory-based AI machine, but we focus on the memory part in this paper.

2. Active AI Memory

The knowledge acquisition/representation is the main problem in the AI system, and its support tools are also developed. For the problems in the real world, whose model is unknown, we often fail in it. We focus on the neural network that works effectively for knowledge acquisition, even if the model is not clear. In this sense, the neural network is a functional memory[1] as well as an active memory. The neural network, however, has no direct knowledge representation facility, and is not enough powerful as a complete AI system. Therefore, we develop a sufficiently *powerful* active memory that is derived from a *new* neural network, the objective-neuron-based network, where "*powerful*" means that it has the facility of **knowledge representation** and of interface with the conventional AI system. (Such an active AI memory includes two kinds of facilities both knowledge acquisition and knowledge representation.)

3. Knowledge Acquisition using Active AI Memory

The knowledge acquisition on the active AI memory is achieved because of classification facility in the neural network. All input data to the neural network are classified into categories, each of which corresponds to an equivalent class[2] (induced by a relation, i.e., "similarity") on a topological space. In this step, we do not need to specify the neural network.

The behavior for acquiring the knowledge in a neural network is as follows;

Step a) Store

All data are stored, ideally, in the designated topology.

Step b) Search

For an input data the most desirable data (i.e., the data with the nearest distance) is obtained.

When the distance is required on a topological space, it can be given by the hardware[3,4]. It is inefficient to execute the step a), if the step is realized as it is. The step a) can be replaced by a data-read procedure when the designated topology is disregarded, although it is used in the step b). In our realization, the hardware is regarded to have only a virtual topological space instead of the real one, because the space is really required only to find the nearest data to the input data. To explain it, we show a simple example as follows;

Example 1.

Data Set: S = { 1.2, 2.4, 3.1, 4.4 }

a) Store:

S is stored as it is. { practical way, *not* ideal }

b) Search:

Search Data: x = 2.8

Question1: Is there the same data as x in S ?

Answer: No.

Question2: What is the nearest data to x in S ? { " norm " is required here }
Answer: It is " 3.1 ".
Question1 is the case of CAM (Content Addressable Memory), but our proposal can support its extension, i.e., Question 2.

In this example, the scalar values are used for simplicity, although the vector is used for the general case. Moreover, the distance (norm) should be flexibly defined depending on the user problem. The hardware for knowledge acquisition is mainly the circuit to find the nearest data to the input data, which is added to the conventional memory, and is sometimes called the winner-take-all circuit[3]. This reduces also the time complexity as well as the hardware complexity, because *searching* is made without *sorting*.

4. ONBAM : A New Active Memory

In order to add the facility of knowledge representation to the neural network, we introduce newly one or more objective neuron(s) for a category (corresponding to an equivalent class) and the mechanism to obtain the representative(s) automatically on each category after classifying the data space.

For data processing among objects with message passing the objective-neuron-based neural network is constructed by a neural network as in Fig.1.

Which type of neural networks is appropriate for such an object-oriented model?
The objective neuron is represented as follows;

Objective Neuron for Knowledge Representation:

Data: $X_i = (x_{i1}, x_{i2}, ... , x_{in})$
{ n-dimensional data for category i }
Method:
Method L:
Modify X_i by Rule { Learning };
Method I/O:
Output for Input { I/O Response };

This objective neuron (in short, ON) differs from the multilayerd perceptron (in short, MLP) as in Example 2. The ON is just an object model, while the MLP can be also represented as an object model.

Example 2.

For input set S with four categories A,B,C and D,
i) MLP neural network.
Neuron 1 divides S into { A, B }and { C, D }.
Neuron 2 does S into { A, C } and { B, D }.
Neuron 3 divides S into { A }, { B }, { C } and { D }, after receiving both results of Neurons 1 and 2. (Neurons 1 and 2 are in the internal (hidden) layer, and Neuron 3 is in the output layer.)
ii) ON neural network.
Neuron 1 extracts { A } from S.
Neuron 2 does { B } from S.
Neuron 3 does { C } from S.
Neuron 4 does { D } from S.
Neuron 5 is used for the output, where no function is needed. Therefore, the fundamental ON neural network is two-layered, although it can be also expanded to three-

layered.

When an object corresponds to a category, the numbers of neurons in the internal layer (in other words, the hidden layer) for n inputs is given as, for k categories,

log k (the minimum case) for MLP, and
k for ON,

respectively.

This comes from that the MLP neural network represents the internal state (corresponding to each category) in distributed form (by log n neurons at the minimum), while the ON does it by one neuron to one category form. If more than one neurons correspond to a category, we regard a neuron as a representative of its sub-category.

As is easily understood, the neural network of ONs requires a large number of neurons when k (number of categories) increases. For the memory-based architecture[5], however, the number of ONs does not interfere realization, since we can expect the nano-meter IC technology to yield the huge scale systems.

Example 3.

Number of Input : n=65,536
Number of Category : k=256
Number of Neurons in Internal Layer:
256 (ON neural network), 8 (minimum, MLP neural network)

One of the important features using ONs is that learning procedure becomes easier than in the MLP. As is in Fig.1, L (Learning stage) provides a parameter modification to each neuron, where the convergence time of learning depends on the learning algorithm. The error-back-propagation algorithm is popular in the MLP neural network, but the convergence time of leaning increases for the large-scale problem. To avoid this, we adopt the learning algorithm for the winner-take-all-type neural network (such as Kohonen net and Grossberg net). This type of learning algorithm fits the ON neural network. In this case, the objective neuron and its state is represented as in Fig.2.

Another important feature is that the objective neuron can be used as an object in the conventional object-oriented programming. This means that the objective neuron plays a role of *bridge* between the neural network and the conventional programming in object level.

5. Knowledge Representation using ONBAM

The ONBAM is already shown as an object style, which corresponds to a *frame* in the conventional knowledge representation. The model of objective neuron is used also in the learning stage. The objective neuron after learning is shown as follows;

Objective Neuron after Learning:
Data: $X_i = (x_{i1}, x_{i2}, \ldots\ldots , x_{in})$
{ n-dimensional data for category i }
Method:
Method L{ Learning is **not in use** };
Method I/O:
Output for Input { I/O Response }
$Y_i = a_{i1}d_{i1} + a_{i2}d_{i2} \ldots\ldots + a_{in}d_{in}$,
where $d_{ij} = | x_{ij} - x'_{ij} |$ and
$X'_i = (x'_{i1}, x'_{i2}, \ldots\ldots , x'_{in})$
{ X'_i is an input for response }.

This means that the objective neuron has only an I/O function in this stage. To explain why this objective neuron is equivalent to a knowledge representation, we give a simple example as follows;

Example 4.

Suppose that an ON neural network be two inputs, x_1 and x_2, and four categories, A, B, C and D. For instance, we have

If x_1 is high and x_2 is high, **then** Category A ;
If x_1 is high and x_2 is low, **then** Category B ;
If x_1 is low and x_2 is high, **then** Category C ;
If x_1 is low and x_2 is low , **then** Category D .

An objective neuron plays a role of each rule as above.

This should be contrasted with the MLP neuron as follows;

If x_1 is high **then** Categories A and B
else Categories C and D ;
If x_2 is high **then** Categories A and C
else Categories B and D ;

It is quite natural, because the number of neurons in the internal layer is that of rules in the ON neural network, where it is reduced in the MLP neural network. This also means the difference between the neural-net-based inference and the fuzzy logic inference, but we omit the details here.

The knowledge representation procedure using the ONBAM is as follows;

1) ***Sampling Procedure***:

First, a small number of data are chosen from the original data set S. These sampling number is supposed to be 0.1 - 10 % depending on the problem. Essentially, this procedure is made by *random* choice.

2) ***Learning-without-Teacher Procedure***:

Suppose that x and C be a sampled input and the stored representatives, respectively.

a) { Searching the nearest data }
Compare x and all $r \in C$, and obtain the minimum distance d between x and r.

b) { Learning Algorithm }
If d is smaller than a parameter e,
then call Kohonen's **LVQ** algorithm[6],
regarding x as a member of Category including r (nearest to x),
else x is stored as a representative of a **new** Category.

Steps a) and b) are repeated while input data exist.

3) ***Knowledge Representation Procedure***:

When the learning procedure stops, each Category (or Sub-Category) has the most appropriate representative. This means that we obtain an equivalent knowledge representation instead of the rule base knowledge. It is not a direct knowledge representation, but is enough effective in object level. Note that the representative of each Category is not a real data, because the leaning procedure has changed its value from the initial one.

The learning procedure includes both learning in Kohonen Net and that in Grossberg Net, and is realized by the hardware in the ONBAM. We can use the learning-with-teacher proceduer (which is equivalent to the rule-based knowledge) together with the learning-without-teacher procedure described as the above.

6. Design of ONBAM

We have designed a prototype of ONBAM (ONBAM-p) as follows;

ONBAM-p.
Input: 32bits
Output: Nearest Data (Distance, Address), where the real value of each data is stored in the attached memory and referred using its address.
State: Arbitrary (e.g., 32 or 64bits for fixed point case), because the ONBAM chip has only the address pointer.

Note that the learning procedure is executed by the conventional processor and that the total ONBAM system is represented as in Fig.3.

The design of ONBAM-p was completed (using VHDL and Verilog), and is now in fabrication stage, where the single chip will include 12 objective neurons.

7. Remarks

The original idea of objective neuron was proposed as a hardware neuron realizing the procedural scheme of Kohonen Net[3], and the design including its performance evaluation are verified by SPICE simulation[4], where a 32bit processing unit is used as a hardware neuron. In this paper we extend the original one to the leaning-without-teacher type, although the ONBAM-p chip is a little simplified from the original objective neuron because the prototype chip is realized by a gate-array with 18K gates/chip. In future, the original version will be realized as a custom design, after the ONBAM-p chip is carefully investigated.

References

[1] T.Ae;"Neural Network and Functional Memories", Journal of IPSJ, vol.32, no.12, pp.1301-1309, 1991 (in Japanese).

[2] T.Ae, Y.Chikamatsu, K.Kawakami;"Topological Neural Net", Proc.WCNN'93(2), pp.151-154, Portland, 1993.

[3] T.Ae, R.Aibara and K.Kioi;"Design of Self-Organizing Chips for Semantic Applications", Proc.3rd Intern. Workshop on VLSI for NNs and AI, Oxford, 1991, and *VLSI for Neural Networks and Artificial Intelligence* (eds.J.D.Delgado-Frias et al.), Plenum Press, pp.109-117, 1994.

[4] T.Ando, K.Kioi, R.Aibara and T.Ae;"A Chip Design of Generalized Winner-Take-All Circuit", Proc.IJCNN(2), pp.1971-1974, Nagoya, 1993.

[5] T.Ae, K.Sakai and T.Toyosaki;"New Functional Machine NFM - Memory-Based AI Architecture - ", IPSJ SIG Workshop on Computer Architecture, Nara, 1994 (in Japanese).

[6] T.Kohonen, *Self-Organization and Associative Memory (3rd Ed.)*, Springer-Verlag, 1989.

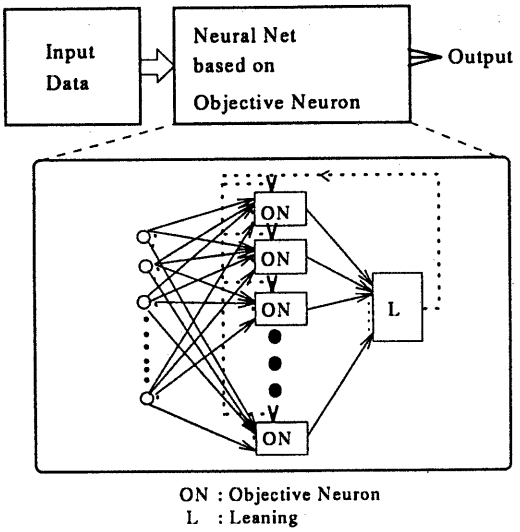
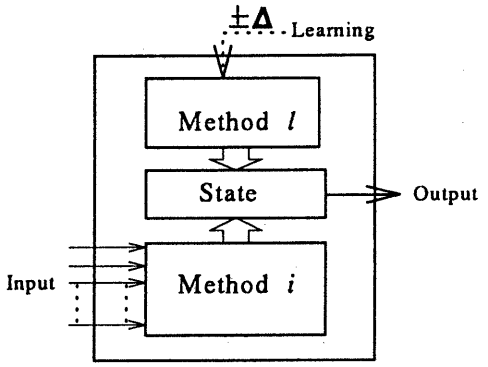
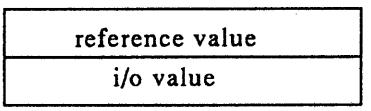


fig 1: Objective-Neuron-Based Neural Network

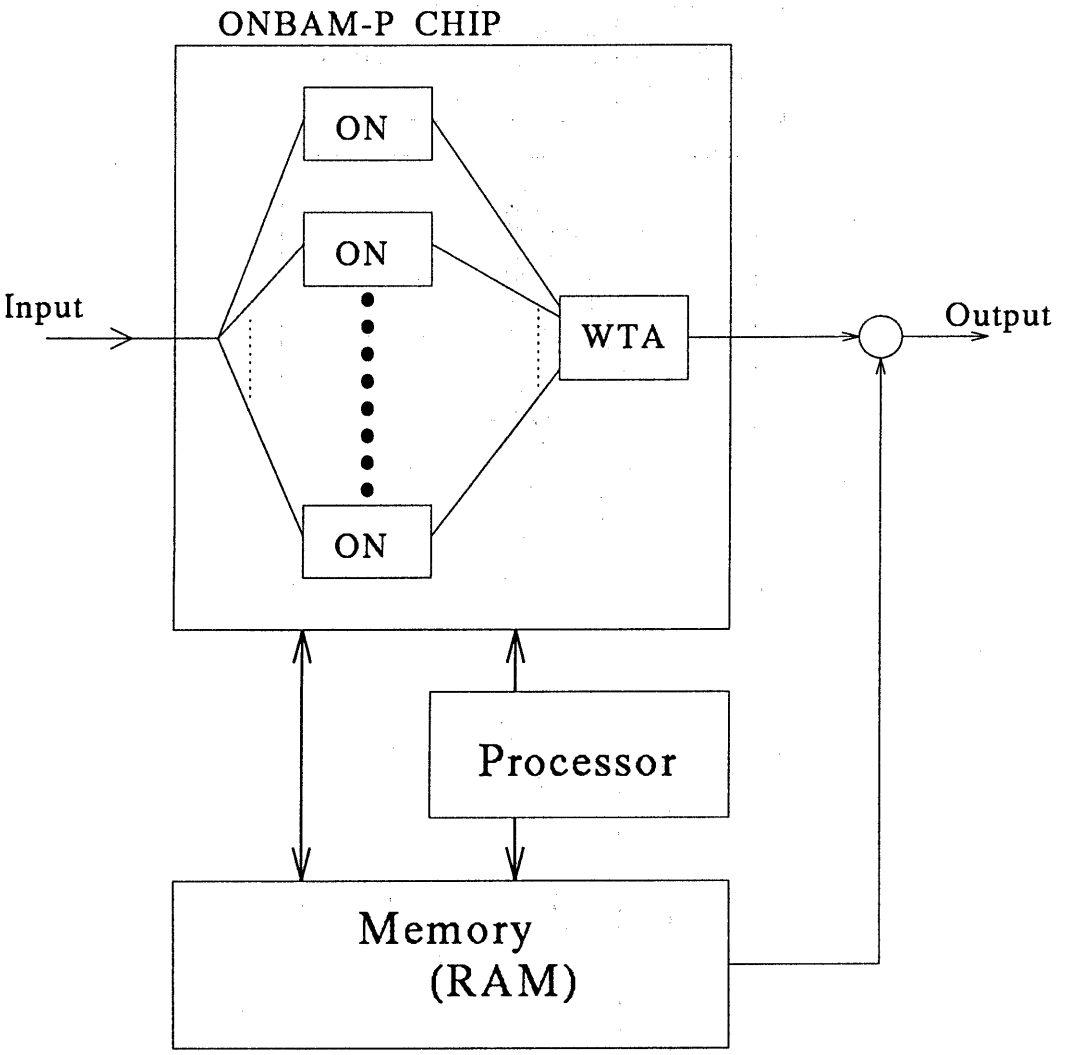


(a) Objective Neuron



(b) state

fig 2: Objective Neuron



ON :Objective Neuron

WTA :Winner Take All Circuit

ONBAM System

fig 3: ONBAM System