Brain Agent Model using Vector State Machine

Tadashi Ae, Keiichi Sakai, Hiroyuki Araki
Graduate School of Engineering, Hiroshima University
1-4-1 Kagamiyama, Higashi-Hiroshima, 739-8527 Japan
Fax: +81-824-22-7195
Email: ae@aial.hiroshima-u.ac.jp.

Abstract

First, we introduce VM(vector state machine) which is generalized from the structured vector addition system, and, next, propose KR/VM model, where KR is the model for knowledge representation, since the conventional AI(Artificial Intelligence) technique is also important. As a result, we obtain a hybrid model of the AI model and the vector state machine, and it will be a good brain agent model which is widely applicable for the practical problems.

Keywords: Automaton, Agent Model, Transition System, Vector State Machine, Inductive Learning

1 Introduction

The neural network is a typical model for human brain and has been discussed by a lot of researchers. The recurrent network plays an important role of system model for human brain, since the model including the states corresponds to a system including the memory part as well as the processing part. Many problems on the recurrent network, however, remain still unsolved.

We have already proposed a two-level model, i.e., a hybrid system model mixed with the neural network and the automaton[Ae *et al.* 1998]. The model is practically utilized for a kind of applications, but is not general because two layers are exclusively connected, that is, not well mixed.

In this paper we introduce a vector state machine, which is generalized from the structured vector addition system[Ae *et al.* 2001]. The vector addition system (in short, VAS) is proposed by R.Karp et al. as a parallel processing model[Karp *et al.* 1968]. The VAS seems to be far from the neural network, but it is an excellent "macro" model for the brain behavior, especially, for the emotional behavior. The original VAS is weak to represent the control mechanism, and therefore, we have proposed a structured VAS, where the control mechanism plays a role of simulating the dynamical behavior of human emotion, together with the state transition of vectors. The addition, however, is an example of the generalized vector state model, since any operation can be used depending on the purpose.

The conventional AI(Artificial Intelligence) technique is also important. We will first introduce it, and define a hybrid model of the AI model and the vector state machine. As

International Journal of Computing Anticipatory Systems, Volume 12, 2002
Edited by D. M. Dubois, CHAOS, Liège, Belgium, ISSN 1373-5411 ISBN 2-9600262-6-8

a result, we will obtain a brain agent model which is widely applicable for the practical problems.

2 Fundamental Model

The state transition model is generally represented as the following;

$$\delta : Q \times I \rightarrow Q \qquad (1),$$

$$\gamma : Q \times I \rightarrow O \qquad (2),$$

where Q is the set of states, and I and O are the set of inputs and the set of outputs, respectively. For the case that the output is uniquely determined by the state, Equation (2) is replaced by

$$\gamma' : Q \rightarrow O \qquad (3).$$

Moreover, Equation (3) will disappear when the output itself is replaced by the state, and Equation (1) is only used.

The set of states is represented by a vector for the brain-like system. The dimension of vector is large for the neural network model (e.g., the recurrent network), but we use a small-size vector because it is enough powerful to represent the human emotional behavior[Musha 1996](Fig.1). The VAS(Vector Addition System) is an example of Vector State Machines, and its behavior is shown as in Fig.2.

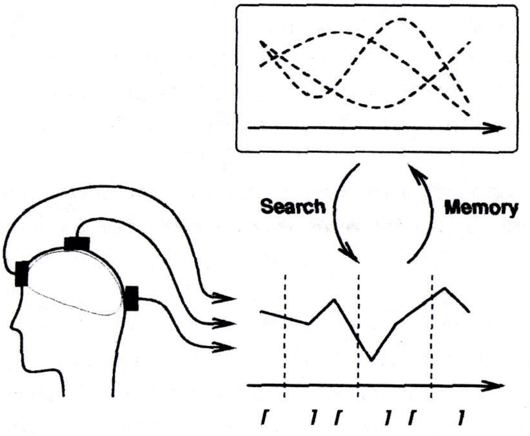


Fig. 1. **A behavior of Vector State Machine.**

The model works in two modes; Learning Mode and Execution Mode.

• Learning Mode

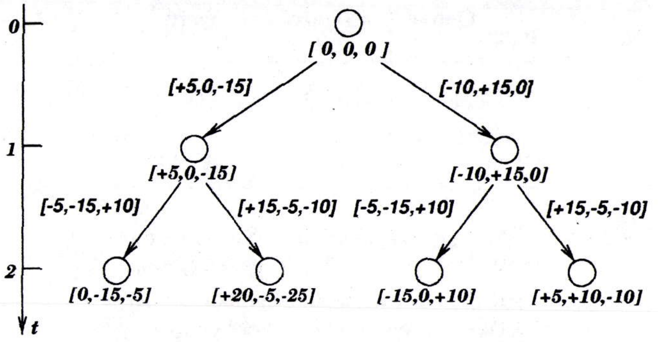


Fig. 2. An Example : VAS (tree representation).

a) Learning for Step Input

$$Q \times I \to Q$$

This is the same as Equation (1) and only adjusting is made.

b) Learning for Sequence Input $I_1 I_2 \dots I_k$
Equation (1) is recursively applied as

$$(\dots((Q \times I_1) \times I_2) \dots \times I_k) \to Q \qquad (4).$$

This corresponds to the automata learning (e.g. MAT [Angluin 1987]) which is introduced to AST Learning, i.e., a hybrid learning[Ae *et al.* 1998], where LVQ (a neural network learning) is also introduced for classification of the input space for transition. Learning for HMM(Hidden Markov Model) is also classified into this type of learning.

• Execution Mode

a) Step Output

$$Q \times I \to O$$

This is the same as Equation (2) and the output is given to the realistic world (e.g., Reinforcement Learning).

b) Sequence Output
Equation (2) is recursively applied as

$$(\dots((Q \times I_1) \times I_2) \dots \times I_k) \to O_1 O_2 \dots O_k \qquad (5).$$

We have the similar one for the case of Equation (3), but omit it.

We denote one step input/output by Symbol, and the sequence input/output by Word. The combination of Symbol or Word yields totally four cases, because of two in the input and two in the output, and therefore, we have Table 1.

Input \ Output	Symbol	Word
Symbol	SISO	SIWO
Word	WISO	WIWO

SISO : Symbol Input / Symbol Output
SIWO : Symbol Input / Word Output
WISO : Word Input / Symbol Output
WIWO : Word Input / Word Output

Table 1. Four Cases depending on Two Modes.

3 Knowledge Representation Model and Vector State Machine

The several models are already developed as the knowledge representation model by AI(Artificial Intelligence) researchers, but the rule model and the frame model are practically used.

Example 1: Rule Model

The knowledge is represented by rules.

Set of Rules : $\{P(X) \to Q(X), R(X) \to S(X)\}$

Set of Instances : $\{X = human, X = bird, X = stone\}$

Predicates P, Q, R and S have the meaning each.

Example 2: Frame Model

The knowledge is represented by frames. An example of frames is shown as in Fig.3.

There exist many applications which are hardly to obtain a knowledge representation, and it is known as the bottleneck problem in AI.

This is true, but we do not need to exclude the AI Model. We will develop a model mixed with AI Model(i.e., Knowledge Representation Model) and Vector State Machine. We denote it by KR/VM model, and the KR/VM model has three versions shown as in Fig.4, where KR and VM mean Knowledge Representation and Vector State Machine, respectively. It is noted that Fig.4 shows the class of machines each. Three versions are Version 1 that KR is larger than VM, Version 2 that VM is larger than KR, and Version 3 that KR and VM are incomparable with each other.

Version 1 shows that VM does not play a role. If we obtain KR, we will use it. This means that the conventional AI technique only is useful.

The realization of Version 2 or Version 3 is important for the KR/VM model, and it is shown as in Fig.5. In Fig.5(a) KR is controlled by VM, and it corresponds to Version 2.

NAME	SMART
AKO (or ISA) Description	
SMART is a kind of CAR. - - - - - -	
Part-Of Description	
Wheel is part of SMART. - - - - - -	
Pointers to Other Frames	

Fig. 3. **An Example of Frame Model.**

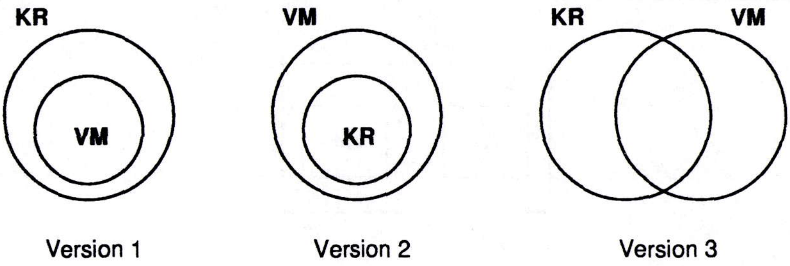


Fig. 4. **Three Hybrid Versions(KR:Knowledge Representation, VM:Vector State Machine).**

This means as follows; First, KR is used to describe a system, but it is enough powerful to do it. Then, VM works so that the description coincides with the system exactly. In Fig.5(b) the interaction of KR and VM is realized by filtering with each other, and it corresponds to version 3.

These two types are used for realization of the brain agent.

4 Agent Construction using KR/VM Model

The KR/VM model in Fig.5 is regarded as a component to construct a realistic agent. We call it Fundamental Component(in short, FC) including two types of Fig.5.

The procedure for construction of a brain agent is described as follows;

Procedure $AgentComposition(X_i, X_j)$;
Either $X_k := X_i + X_j$ {Serial Connection}
Or $X_k := X_i \parallel X_j$ {Parallel Connection}.

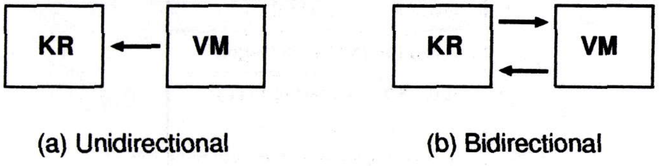


Fig. 5. **Interaction of KR and VM(KR:Knowledge Representation, VM:Vector State Machine).**

This procedure is shown as in Fig.6, where X_i and X_j are initially FCs, but the procedure is recursively applied. As a result, we can realize a realistic brain agent.

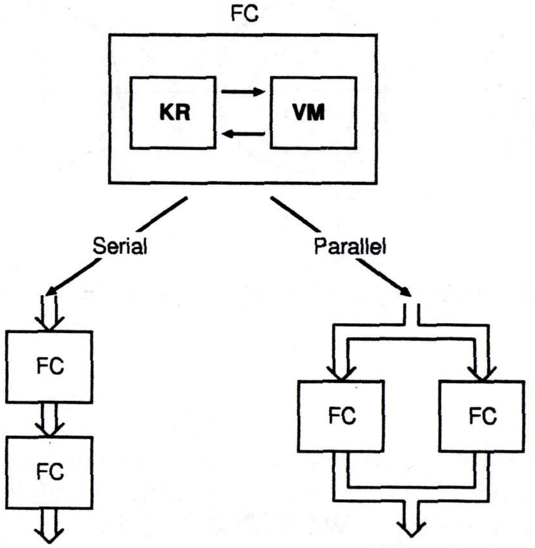
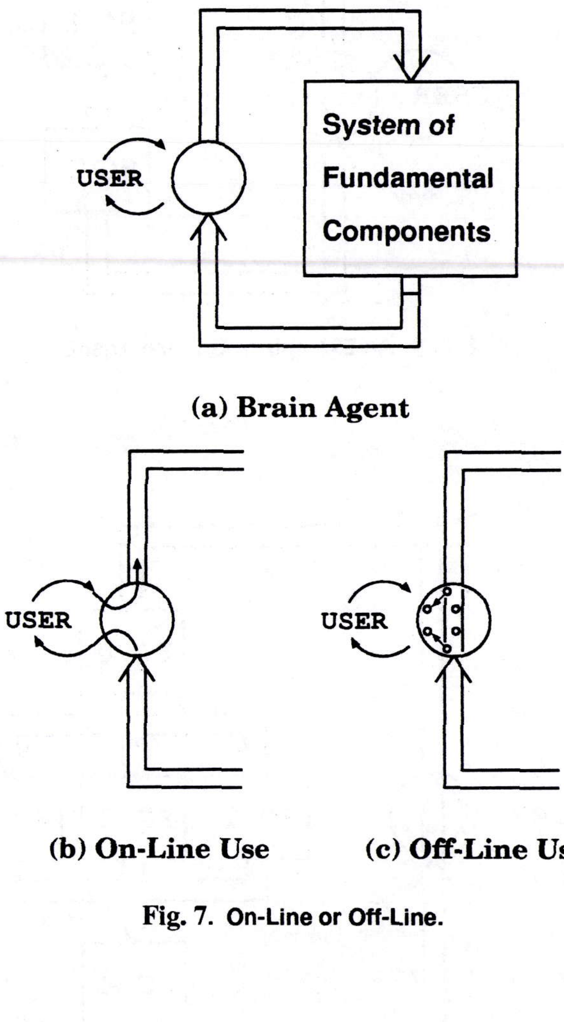


Fig. 6. **Serial/Parallel Connection(FC:Fundamental Component).**

The brain agent is classified into two types (on-line and off-line), which are depending on the frequency of user access as in Fig.7. The on-line agent interacts fundamentally every time with the user as in Fig.7(b), but the off-line agent does not. This classification depends on applications, because the model is apparently the same.

An example of On-Line Brain Agent is shown as in Fig.8, which is used for real-time application. An example of Off-Line Brain Agent is shown as in Fig.9, which is used for supporting a human creative activity.



(a) Brain Agent

(b) On-Line Use **(c) Off-Line Use**

Fig. 7. On-Line or Off-Line.

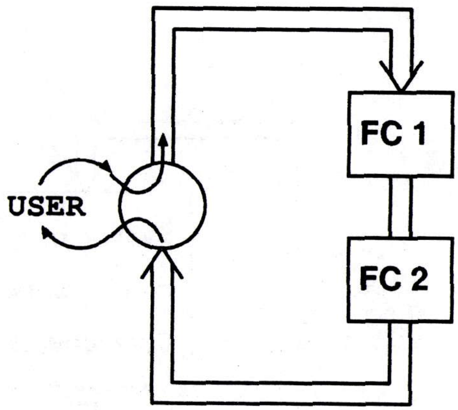


Fig. 8. **An Example of On-Line Agent.**

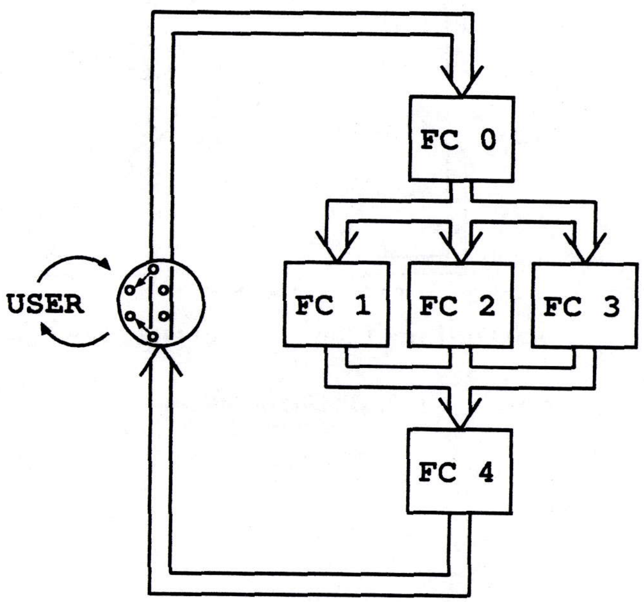


Fig. 9. **An Example of Off-Line Agent.**

5 Learning for KR/VM Model

The learning for KR/VM Model is represented as follows;

Inputs: Set of Sequences $\{X_1, X_2, \ldots, X_k\}$.
$X_i(i = 1, 2, \ldots, n)$ is given as $X_{i1}X_{i2}\ldots X_{in}$, where $X_{ij}(j = 1, 2, \ldots, n)$ is a tuple of symbol and vector, e.g., $(\alpha, (v_1, v_2, v_3, v_4))$.

Learning: From the set of sequences, $\{X_1, X_2, \ldots, X_k\}$, the types of state sequences are obtained after learning procedure.

type 1: $Q_{11}, Q_{12}, \ldots, Q_{1q}$
type 2: $Q_{21}, Q_{22}, \ldots, Q_{2q}$
....
type p: $Q_{p1}, Q_{p2}, \ldots, Q_{pq}$

State $Q_{ij}(i = 1, 2, \ldots, p, j = 1, 2, \ldots, q)$ is defined on VM (Vector Machine), e.g., SVAS(Structured Vector Addition System, see the appendix). If VM has no predicate, the learning will be the same as in HMM (Hidden Markov Model), but it must be a structured leaning for the general case.

The learning procedure is represented as in Fig.10, since the inductive learning consists of the induction cycle. The evolutional step between two induction cycles in Fig.11 is important to recognize whether the system in really improved or not. The problem of evolutional steps, however, has an infinite hierarchy as in Fig.12.

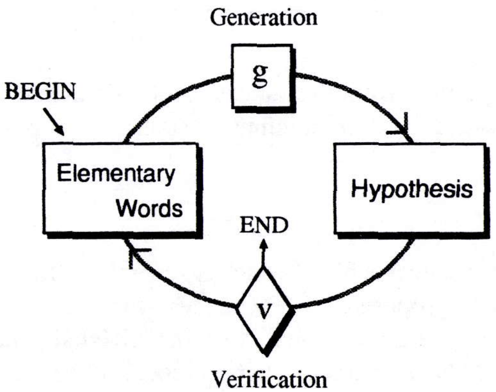


Fig. 10. Induction Cycle.

Here we refer to the decidability problems on KR/VM model. In general, the equivalence problem (whether or not a KR is equivalent to a VM) is undecidable. Similarly, the inclusion problem also becomes undecidable. Such a problem becomes decidable only for the case of nearly finite-state models.

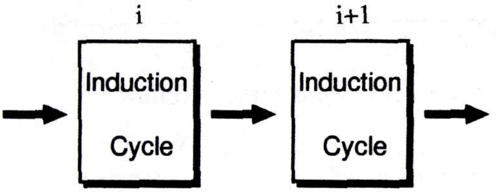


Fig. 11. **Evolutional Step of Induction Cycles.**

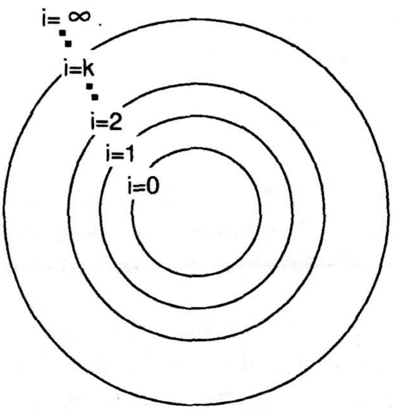


Fig. 12. **Infinite Hierarchy of Evolutional Steps.**

The essential problem comes from that an extension of finite systems requires an infinite number of finite systems to cover an infinite system.

6 Conclusion

We introduced VM(vector state machine), which is generalized from the structured vector addition system, and, next, proposed KR/VM model, where KR is the model for knowledge representation, since the conventional AI(Artificial Intelligence) technique is also important. As a result, we have obtained a hybrid model of the AI model and the vector state machine, and it will be a good brain agent model which is widely applicable for the practical problems.

References

[Ae *et al.* 1998] T.Ae, H.Araki, K.Sakai, "Structured Brain Computer and its Learning", Computing Anticipatory Systems: CASYS'98-Second International Conference, edited by Daniel M. Dubois, American Institute of Physics, Conference Proceedings 465, pp.111-120, Woodbury, New York (1998).

[Ae *et al.* 2001] T.Ae, H.Araki, K.Sakai, "Structured Vector Addition System — A Simulated Brain Model for Creativity Activity —", International Journal of Computing Anticipatory Systems (to appear, 2001).

[Karp *et al.* 1968] R.Karp, R.Miller, "Parallel Program Schemata", J. Computer and System Sciences, Vol.3, pp.147-195 (1968).

[Musha 1996] T.Musha, "Measuring Mind", Nikkei-Science, pp.20-29, Tokyo (April 1996, in Japanese).

[Angluin 1987] D.Angluin, "Learning Regular Sets from Queries and Counterexamples", Information and Computation, Vol.75, pp.87-106 (1987).

Appendix

1. Fundamental Model: Vector Addition System

The Vector Addition System (in short, VAS) is originally defined by Karp et al..

For the set of integers, Z, we describe the l-dimensional vector by $V = [v_1, v_2, \ldots, v_l]$ $(v_u \in Z, u = 1, 2, \ldots, l)$. The VAS M is defined as

$$M = (V(0), S).$$

where $V(0)$ is the initial vector and S is the set of vectors for addition. The state transition in VAS is represented as

$$(Q(t), i) \rightarrow Q(t+1), \text{ where } Q(t+1) = Q(t) + i.$$

2. Structured Vector Addition System

Although the VAS can represent a macro behavior of simulated brain, it is not useful to simulate the behavior of brain, since the control mechanism is not explicitly represented. Therefore, we extend the VAS to a structured VAS.

The Structured Vector Addition System (in short, SVAS) is defined as follows;

$S = \{s_0\} \cup \{s_1, s_2, \ldots, s_k\}$, where $s_0 = [0, 0, \ldots, 0]$ and $s_i (i = 1, 2, \ldots, k)$. S is the set of special state vectors.

$(Q(t), i) \rightarrow Q(t+1)$, where $Q(t+1) = Q(t) + i$,
if $P((Q(t), Q(t+1))$ is false; (the same as VAS)
$Q(t+1)$ is in S,
if $P((Q(t), Q(t+1))$ is true. (the same as Logic Programming)

Predicate P can be defined in the system.