Structured Vector Addition System — A Simulated Brain Model for Creative Activity —

Tadashi Ae, Hiroyuki Araki, Keiichi Sakai
Electrical Engineering, Faculty 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

In this paper we introduce an extended vector addition system, i.e., a structured vector addition system. The vector addition system (in short, VAS) is proposed by R.Karp et al. as a parallel processing model, but the VAS seems to be far from the neural network. However 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 propose a structured VAS (in short, SVAS), where the control mechanism plays a role of simulating the dynamical behavior of human emotion, especially, with the state transition of vectors.

We will discuss the inductive learning and the anticipation on SVAS.

Keywords: Automaton, Transition System, Vector Addition System, Inductive Learning, Evolution

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, remains still unsolved.

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

In this paper we introduce an extended vector addition system, i.e., a structured vector addition system. 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 propose a structured VAS, where the control mechanism plays a role of simulating the dynamical behavior of human emotion [Musha 1996], especially with the state transition of vectors.

In the paper we describe the following:
- Fundamental model and the structured VAS (in short, SVAS).
- Inductive learning and anticipation.
- Possibility of artificial creativity.

2 Fundamental Model: Vector Addition System

First we introduce the Vector Addition System (in short, VAS) which is originally defined by Karp et al. [Karp *et al.* 1968].

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

$$M = (\boldsymbol{V}(0), \mathcal{S}), \tag{1}$$

where $\boldsymbol{V}(0)$ is the initial vector and $\mathcal{S}$ is the set of vectors for addition.

The behavior of VAS with finite length is represented as follows;

$$\begin{aligned} &\boldsymbol{V}(0) = \boldsymbol{0}\ (= [0, 0, \cdots, 0]) \\ &\boldsymbol{V}(t+1) = \boldsymbol{V}(t) + \boldsymbol{A}(t) \\ &\left(\begin{array}{l} t = 0, 1, \cdots, n-1 \\ \boldsymbol{A}(t) \in \boldsymbol{S}_t \\ \boldsymbol{S}_t = \{\boldsymbol{a}_{t1}, \boldsymbol{a}_{t2}, \cdots, \boldsymbol{a}_{tq}\} \end{array} \right) \end{aligned} \tag{2}$$

where the selection of vectors for addition at time t is given by the subset $\boldsymbol{S}_t \subset \mathcal{S}$, and $\boldsymbol{S}_0, \boldsymbol{S}_1, \cdots, \boldsymbol{S}_{n-1}$ is defined in M, while we suppose to have a finite times addition. An example behavior is shown as in Fig.1.

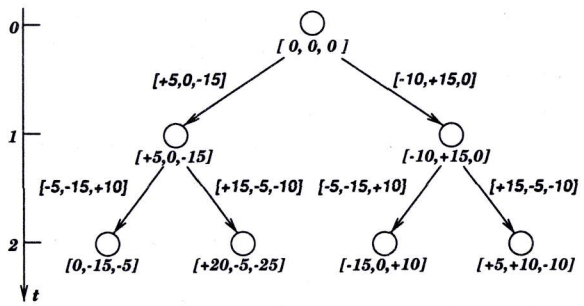

Fig. 1. An Example : VAS (tree representation).

In l-dimensional vector space we have a sequence of state vectors $\mathcal{V}$

$$\mathcal{V} = \boldsymbol{V}(0)\boldsymbol{V}(1)\cdots\boldsymbol{V}(n-1)\boldsymbol{V}(n). \tag{3}$$

We denote by $\boldsymbol{R}(M)$ the set of all sequences of state vectors of length $n+1$, which are generated by M. We denote by $\mathcal{A}$ the sequence of vectors for addition with length n corresponding to $\mathcal{V}$

$$\mathcal{A} = \boldsymbol{A}(0)\boldsymbol{A}(1)\cdots\boldsymbol{A}(n-1). \tag{4}$$

The VAS is originally proposed for a parallel processing model by R.Karp et al, and seems to be far from the neural network, but it is an excellent macro model for the brain behavior, especially, for the emotional behavior [Musha 1996] (see Fig.2). When focusing on the vector corresponding to each sensor attached to the brain surface, the vector sequence represents the behavior of human brain (although it may include unnecessary information). Hereafter we use the VAS as a macro model for simulated brain.

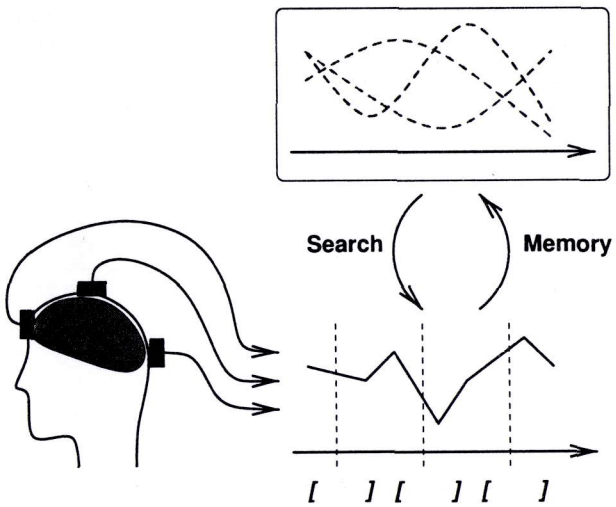

Fig. 2. A Simulated Model of Human Brain (The vector sequence represents the behavior of human brain).

For training of VAS M, M needs to get an appropriate sequence $\mathcal{V}$ when a sequence of state vectors is given, since the sequence search is a fundamental problem for any state transition system (e.g., Markov Model). The formal definition for sequence search is given as follows;
Let $\mathcal{X}$ be

$$\mathcal{X} = \boldsymbol{X}_0\boldsymbol{X}_1\cdots\boldsymbol{X}_m, \tag{5}$$

i.e., an ordered sequence, where $\boldsymbol{X}_0 = \boldsymbol{V}(0)$ and $1 \leq m \leq n$. The sequence search problem is;
Obtain

$$\boldsymbol{X}_i = \boldsymbol{V}(n_i), \quad \exists n_i \; for \; each \; i \tag{6}$$

for a given ordered sequence. An example is shown as in Fig.3. This problem is easily shown to be NP-complete, since the case of $m = 1$ corresponds to the knapsack problem (an well-known NP-complete problem). (Note that the general problem of $m = 1$ is the reachability problem and that its complexity is known to be intractable as same as the case of Petri Net [Mayr 1984].)

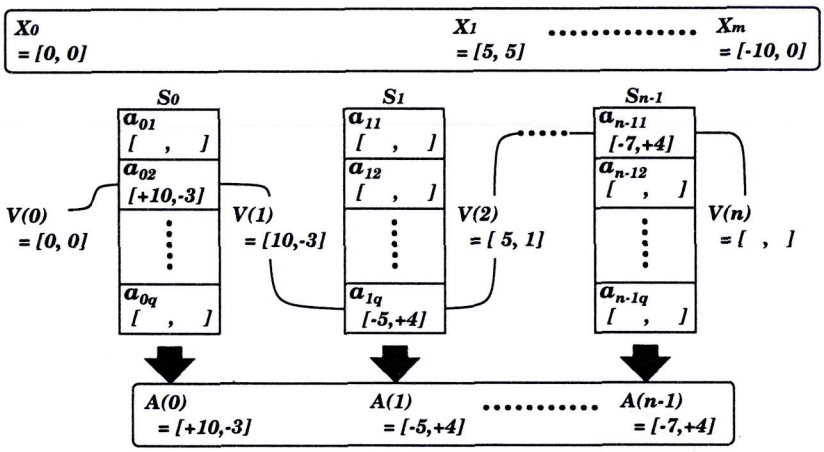

Fig. 3. A Search Problem for Sequences on VAS.

From the viewpoint of practice we introduce an approximated solution for sequence search [Umeda *et al.* 2000]. When introducing the precision δ_i as

$$\delta_i = \frac{|\boldsymbol{V}(n_i) - \boldsymbol{X}_i|}{a_{max}} \quad (for \; \exists n_i), \tag{7}$$
$$\left(\begin{array}{l} 0 < n_1 < n_2 < \cdots < n_m = n \\ a_{max} = \max\limits_{t,k,u} |a_{tku}| \\ \boldsymbol{a}_{tk} = [a_{tk1}, a_{tk2}, \cdots, a_{tkl}] \end{array} \right)$$

and then, equality = in Eq.6 is replaced by $\cong$. Therefore, for sequence search we can apply an algorithm which is already known to be applicable (such as Dynamic Programming, Branch & Bound, Genetic Algorithm etc.).

3 Structured Vector Addition System

Although the VAS can represent a macro behavior of simulated brain, it is not yet useful to simulate the behavior of brain since the control mechanism is not explicitly represented. Therefore, we extend the VAS to the Structured VAS (in short, SVAS) in this paper.

First we extend the domain of vector space in VAS from integer to real number. For the set of real numbers, $\boldsymbol{R}$, we describe the l-dimensional vector by $\boldsymbol{V} = [v_1, v_2, \cdots, v_l](v_u \in \boldsymbol{R},\ u = 1, 2, \cdots, l)$. The SVAS M_s is defined as

$$M_s = (\boldsymbol{V}(0), \mathcal{S}, \mathcal{Q}), \tag{8}$$

where $\boldsymbol{V}(0)$ is the initial vector, $\mathcal{S}$ is the set of vectors for addition, and $\mathcal{Q}$ is the finite set of special states given as the following;

$$\mathcal{Q} = \{q_o\} \cup \{q_1, q_2, \cdots, q_k\}, where\ q_0 = [0, 0, \cdots, 0]\ and\ q_i\ (i = 1, 2, \cdots, k).$$

The behavior of SVAS is represented in a similar fashion to the behavior of VAS as follows;

$$\begin{aligned} &\boldsymbol{V}(0) = \boldsymbol{0}\ (= [0, 0, \cdots, 0]) \\ &\boldsymbol{V}(t+1) = f(\boldsymbol{V}(t), \boldsymbol{V}(t+1), \boldsymbol{A}(t)) \\ &\quad where\ \boldsymbol{V}(t+1) = \boldsymbol{V}(t) + \boldsymbol{A}(t), if\ \mathcal{P}(\boldsymbol{V}(t), \boldsymbol{V}(t+1))\ is\ false; \\ &\qquad\quad \boldsymbol{V}(t+1)\ is\ in\ \mathcal{Q}, \qquad if\ \mathcal{P}(\boldsymbol{V}(t), \boldsymbol{V}(t+1))\ is\ true. \end{aligned} \tag{9}$$

Eq.9 means that the transition depends on the state of time $t+1$ as well as the state of time t, and that the next state is controlled by the predicate $\mathcal{P}$ and the special states $\mathcal{Q}$. The other definitions follow the case of VAS and their description is omitted here.

Predicate $\mathcal{P}$ can be defined by the system each, but three types in Table 1 will be used practically.

Table 1. Several Types of Predicates (V_r is a reference vector and $|\boldsymbol{V}(t+1) - \boldsymbol{V}(t)|$ means the norm between two vectors, where r is a reference value.).

Type	A	B	C
$\mathcal{P}$=true	$\boldsymbol{V}(t) = V_r$ or $\boldsymbol{V}(t) \cong V_r$	$\boldsymbol{V}(t+1) - \boldsymbol{V}(t) > V_r$ or $\lvert\boldsymbol{V}(t+1) - \boldsymbol{V}(t)\rvert > r$	mixed A with B

The intuitive behavior of SVAS is shown as in Fig.4, where the beginning of state transition will jump to one of the designated points (defined by the special states) when Predicate $\mathcal{P}$ becomes true, and the jump will occur as many as it is required.

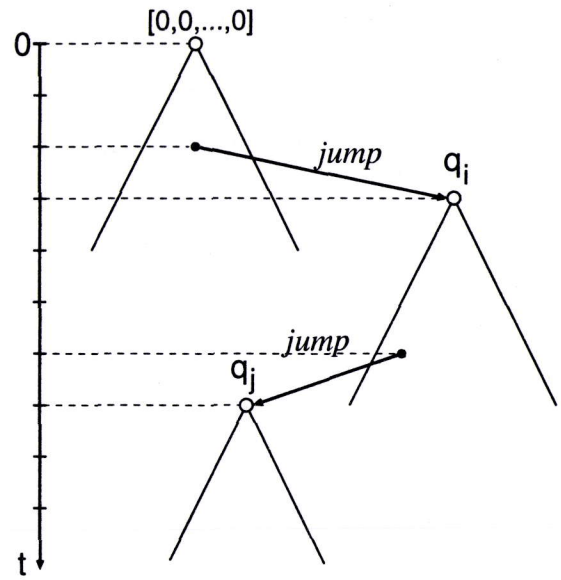

Fig. 4. Behavior of SVAS (Jump to a designated state may occur.).

The class of SVAS is represented as in Table 2. For the discussion of computing class we exclude the restriction of length n of sequence (in Eq. 4), although the number of vectors for addition is finitely restricted. The SVAS extended from VAS is more powerful than Turing Machine (which is similar to the mutually-connected network using real number elements [Hayashihara *et al.* 1990]).

When we denote by SVAS-I a subclass of SVAS whose vector domain is restricted to integer, SVAS-I becomes equivalent to Turing Machine. The proof is given by using a multiple counter automaton which can be easily simulated by SVAS-I, and vice versa.

Table 2. Typical Class of SVAS and VAS.

	Class Name	Computing Power
SVAS	General Type	> Turing Machine
SVAS-I	Vector Domain : $\boldsymbol{Z}$	Turing Machine
VAS-R	Vector Domain : $\boldsymbol{R}$	unknown
VAS	Vector Domain : $\boldsymbol{Z}$	Petri Net

4 Inductive Learning and Anticipation

Learning is divided into two levels, i.e., the fundamental level and the system level. Learning mode in the fundamental level is shown as in Fig.5, where the SVAS is a learner.

The mode has three phases as follows;

(1) Teacher sends Sample to Learner.

(2) Self Learning: Learner repeats learning procedure.

(3) Learner sends Hypothesis to Teacher.

In this case the learner must belong to a learnable class, since the SVAS is generally too hard to be made learn. For instance, a learnable class of SVAS is a subclass of SVAS which is equivalent to the class of deterministic one-counter automata.

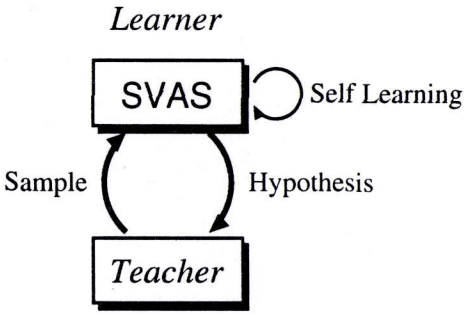

Fig. 5. Learning Model.

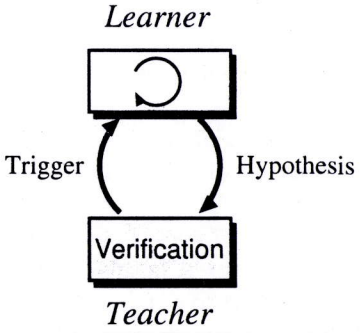

Fig. 6. Induction Cycle (Łearning on System Level).

Within learning mode, (2) Self Learning corresponds to a deductive learning, that is, a learning which is made by a procedural learning algorithm. The algorithm is supposed to terminate within a polynomial time.

The main behavior of learning mode in Fig.5 consists of a loop between Teacher and Learner. For a hypothesis from Learner the teacher must verify whether the hypothesis is true or not. The loop between Teacher and Learner will be repeated until the verification is completed, and then, we focus on learning on the system level, where the SVAS (Learner) is supposed to have finished the fundamental level. Learning on the system level is shown as in Fig.6.

In general we denote by Fig.7 the inductive learning, where it is called the induction cycle and the hypothesis is represented by word (i.e., sequence).

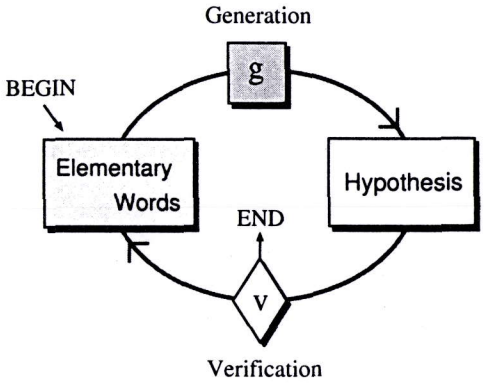

Fig. 7. Induction Cycle.

The anticipation in language corresponds to a creation of "New Words".

Creation Procedure of "New Words" :

0) Setting Initial Pool;

repeat 1) Pool Re-Setting; {Elementary words revised}

2) Generation Procedure;

3) Obtain Hypothesis; {Candidates of New Words}

4) Evaluation using Verification Procedure;

until "New Words" are born .

One of the most important problems is how to construct Teacher, i.e., Verification Procedure. In general it is impossible to obtain a perfect teacher for creative activity, and therefore, first we will have a practically obtainable teacher (an well-trained SVAS in Fig.5), and next, will develop it by a gradually improvable method which we call anticipation or evolution.

Hereafter we will discuss the possibility of birth of New Words by using language theory [Ae *et al.* 2000].

5 Possibility of Artificial Creativity

1) Birth of New Words

Suppose that the elementary sets of words are $L(T_1), L(T_2), \cdots, L(T_p)$, where $L(T_i)$ $(i = 1, 2, \cdots, p)$ is the set of words that T_i (SVAS) accepts and each set has been trained. The generation procedures are supposed to be intersection and shuffling [Ae *et al.* 2000]. Both cases (intersection and shuffling) can produce the new words, since both operations are not closed in the language which corresponds to a learnable class of SVAS. The new words are expected to be beyond the context-free language, although the elementary language is supposed to be at most the context-free language.

2) Evolution

Suppose that we obtain several new words after the induction cycle in Fig.7 was repeated. If the i-th stage is terminated, then the stage goes into the $i+1$-th stage, where $i = 1, 2, \cdots$. Fig.8 shows an evolutional step of induction cycles. How to determine the end of each stage? This is important and the most difficult problem. The control mechanism as in Fig.9 is required to determine whether or not the cycle goes from the i-th stage to the $i+1$-th stage. As a result we represent the evolutional step checker as in Fig.10.

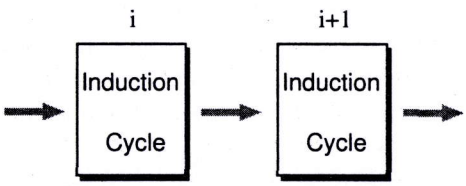

Fig. 8. Evolutional Step of Induction Cycles.

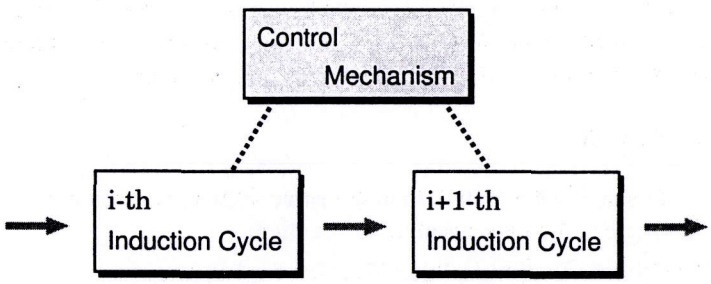

Fig. 9. Control Mechanism for Evolution on Induction Cycles.

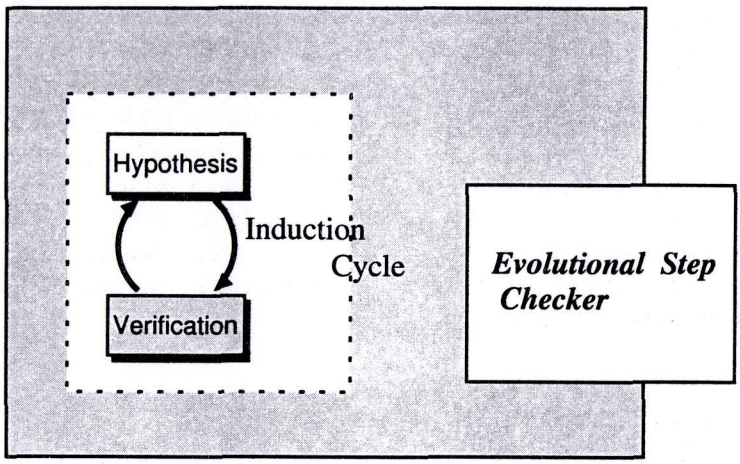

Fig. 10. Evolutional Step Checker.

3) Evolutional Step Checker

On the evolutional step, the $i+1$-th stage must be higher than the i-th stage, and the evolutional step in Fig.8 is from i-th to i+1-th, that is, one-directional. Then, the direction of interconnection between the i-th stage and the $i+1$-th stage is also one-directional. The present cannot look at the future in the world of determinism. The automata theory, however, includes non-determinism, which is also important in the class of pushdown automata. If the non-determinism is allowed in the i-th stage ($L(T_i)$), the interconnection becomes equivalent to be two-directional, because the automaton of $L(T_i)$ can guess the behavior of $L(T_{i+1})$ and the automaton of $L(T_{i+1})$ can verify its guess [Ae 1977]. On the evolution step, the i-th stage guess the behavior of higher ($i+1$-th) stage, and its report is sent to the i+1-th stage. The i+1-th stage will judge whether the report is really evolutional or not. The i+1-th stage may be a teacher for the i-th stage, and the i-th stage cannot ask the equivalence question which is essentially concerned with evolution.

The problem for the interconnection of pushdown automata remains still open for the case of the determinism, i.e., the problem whether or not the one-direction is properly less strong than two-direction. In this paper, however, we propose a practical way for evolutional step without discussing such a cumbersome problem.

An Evolutional Step Checker:

Imagine $i+1$-th stage Teacher, and then, construct i-th stage Teacher $\cap$ $i+1$-th stage Teacher, where $\cap$ includes a semantic operation.

Set this intersection as Evolutional Step Checker for i-th stage.

The evolutional step checker includes an ambiguity as in Fig.11, because the $i+1$-th stage teacher is constructed by imagination (in practice, including a random procedure). The $i+1$-th stage teacher, however, will be a properly distinct teacher from the i-th stage one, although both have a common property (i.e., the intersection between two is not empty).

As a result we have an infinite hierarchy of checker class as in Fig.12. This is quite natural for evolution, since the evolutional steps will continue infinitely.

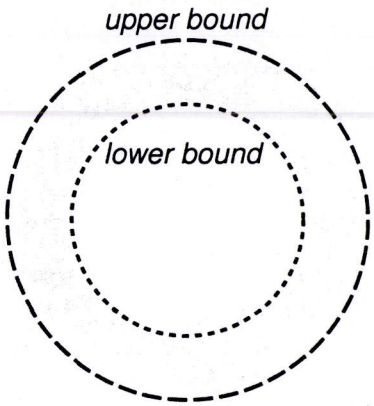

Fig. 11. Ambiguity of Checker.

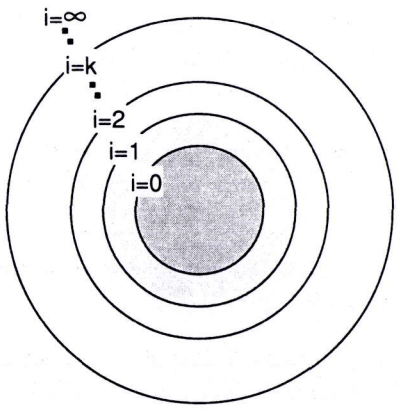

Fig. 12. Infinite Hierarchy of Checker Class.

6 Conclusion

In this paper we have introduced an extended vector addition system, i.e., a structured vector addition system (SVAS). The original VAS is weak to represent the control mechanism, and therefore, the SVAS defining the control mechanism explicitly can represent directly the dynamical behavior of human emotion with the state transition of vectors.

We have discussed the inductive learning and the anticipation on SVAS. The feature of automaton-based system is that the procedural technique is available even for the inductive learning. The technique is usually restricted by the discrete mathematics, and often encounters undecidability [Dubois 2000], and therefore, the SVAS is extended to a machine on real number (which is similar to a mutually-connected neural network [Hayashihara *et al.* 1990]). However, the theoretical problems still remain open.

References

[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).

[Mayr 1984] E.W.Mayr, "An Algorithm for the General Petri Net Reachability Problem", SIAM Journal of Computing, Vol.13, No.3, pp.441-460 (1984).

[Umeda *et al.* 2000] K.Umeda, H.Araki, T.Ae, "Search for Sequences on Vector Addition System", Tech. Report IEICE of Japan, SS99-62 (January 2000, in Japanese).

[Hayashihara *et al.* 1990] K.Hayashihara, M.Yamashita, T.Ae, "On Machine Power of Neural Networks Depending on Continuity or Discreteness of Sigmoidal Function", Trans. IEICE of Japan, Vol.J73-D-II, No.8, pp.1220-1226 (August 1990, in Japanese).

[Ae *et al.* 2000] T.Ae, H.Araki, K.Sakai, "Automaton-Based Anticipatory System", International Journal of Computing Anticipatory Systems, edited by Daniel M. Dubois, Vol.6, pp.67-74 (2000).

[Ae 1977] T.Ae, "Direct or Cascade Product of Pushdown Automata", J. Computer and System Sciences, Vol.14, pp.257-263 (1977).

[Dubois 2000] D.M.Dubois, "Review of Incursive, Hyperincursive and Anticipatory Systems- Foundation of Anticipation in Electromagnetism", Computing Anticipatory Systems: CASYS'99-Third International Conference, edited by Daniel M. Dubois, American Institute of Physics, Conference Proceedings 517, pp.3-30, New York (2000).