Journal of Korea Multimedia Society Vol. 18, No. 11, November 2015(pp. 1281-1288)
http://dx.doi.org/10.9717/kmms.2015.18.11.1281

Generation of Pattern Classifiers Based on Linear Nongroup CA

Un-Sook Choi[†], Sung-Jin Cho[††], Han-Doo Kim[†††]

ABSTRACT

Nongroup Cellular Automata(CA) having two trees in the state transition diagram of a CA is suitable for pattern classifier which divides pattern set into two classes. Maji et al.[1] classified patterns by using multiple attractor cellular automata as a pattern classifier with dependency vector. In this paper we propose a method of generation of a pattern classifier using feature vector which is the extension of dependency vector. In addition, we propose methods for finding nonreachable states in the 0-tree of the state transition diagram of TPMACA corresponding to the given feature vector for the analysis of the state transition behavior of the generated pattern classifier.

Key words: Pattern Classifier, Cellular Automata(CA), Multiple Attractor CA(MACA), Nonreachable state, Feature Vector

1. INTRODUCTION

Pattern recognition has applications in computer vision, radar processing, and text classification. Pattern recognition is to classify input data by classes to distinguish with extracting important characteristics and attributes from data. It is necessary to classify patterns into classes by automated devices for a given pattern set. The ultimate goal of pattern recognition system is to classify the recognized patterns into a member of corresponding pattern classes by detecting and extracting the common features from the given pattern set[2]. The important prerequisites for designing the pattern classifier are high throughput, small storage state space and low cost hardware implementation.

A Cellular Automata(CA) is a discrete dynamical system, which evolves in discrete space and time[3-9]. CA is divided into linear CA and nonlinear CA according to the type of rules applying to the state transition. If the rule of a CA cell employs only XOR logic, then the CA is called a *linear CA*, otherwise it is called a *nonlinear CA*[3]. In addition, CA is divided into group CA and nongroup CA. Linear CA are characterized utilizing matrix algebra. In this paper, we concentrate on linear CA with two-state 3-neighborhood dependency. Also we limit on discussion for null boundary condition only. A CA generating both cyclic and non-cyclic states is a nongroup CA. For a nongroup CA, there are some states that can not be reachable from any other state. These states are referred to as *nonreachable states*. The nongroup CA for which the state transition diagram consists of a set of disjoint components forming tree-like structures rooted at attractors are referred to as *multiple attractor*

※ Corresponding Author : Sung-Jin Cho, Address: (608-737) Dept. of Applied Math., Pukyong National University, 45, Yongso-ro, Nam-Gu, Busan, Korea , TEL : +82-51-629-5527, FAX : +82-51-629-5519, E-mail : sjcho@pknu.ac.kr
Receipt date : July 1, 2015, Revision date : Sep. 17, 2015
Approval date : Sep. 23, 2015

[†] Dept. of Information & Communications Eng., Tongmyong University
(E-mail : choies@tu.ac.kr)
[††] Dept. of Applied Math., Pukyong National University
[†††] Institute of Basic Science and Dept. of Applied Math., Inje University
(E-mail : mathkhd@inje.ac.kr)
※ This Research was supported by the Tongmyong University Research Grants 2013(20013A068)

CA(MACA)[10]. Two predecessor MACA(TPMACA) having two trees is suitable for natural pattern classifier which divides pattern set into two classes.

Maji et al.[1] has designed a pattern classifier that can effectively classify 2-class patterns in a manner that minimizes the amount of memory by synthesizing MACA. They proposed a method for identifying and classifying patterns using a dependency vector. Cho et al. proposed method for synthesizing 90/150 maximum length CA, 90/150 TPMACA with two attractor trees and a two predecessor SACA(TPSACA)[8,11]. And Cho et al.[12] proposed a theoretical method of synthesizing an n-cell TPMACA derived from an $(n-1)$-cell TPMACA $(n \geq 4)$. Moreover, until now there does not exist any method for synthesizing MACA for the feature vectors of the forms $(0 * \cdots * 0)$ or $(* \cdots * 100 \cdots 01 * \cdots *)$. To overcome this problem, we propose a pattern classifier based on a linear nongroup CA that can classify patterns according to a given feature vector.

In this paper we propose two algorithms for generations of TPMACA and MACA corresponding to the given feature vectors. In addition, we propose methods for finding nonreachable states in the 0-tree of the state transition diagram of TPMACA corresponding to the given feature vector for the analysis of the state transition behavior of the generated pattern classifier.

2. PRELIMINARIES AND RELATED WORKS

CA has a simple, regular, modular and cascadable structure with logical neighborhood interconnection. The simple structure of CA with logical interconnections are ideally suited for hardware implementation[9]. The next state of the cell at time $(t+1)$ is affected by its state and the states of its neighbors (left and right neighbors) at time t. The next state s_i^{t+1} of the ith CA cell is specified by f_i as

Table 1. Linear rules and state transition functions

Linear rules	State transition functions
60	$s_i^{t+1} = s_{i-1}^t \oplus s_i^t$
102	$s_i^{t+1} = s_i^t \oplus s_{i+1}^t$
150	$s_i^{t+1} = s_{i-1}^t \oplus s_i^t \oplus s_{i+1}^t$
170	$s_i^{t+1} = s_{i+1}^t$
204	$s_i^{t+1} = s_i^t$
240	$s_i^{t+1} = s_{i-1}^t$

$$s_i^{t+1} = f_i(s_{i-1}^t, s_i^t, s_{i+1}^t) \tag{2.1}$$

where s_i^t represents the state of the ith cell at the tth instant of time and f_i is the next state function and referred to as the *rule* of CA. If the rule of a CA cell involves only XOR logic, then it is called a *linear rule*. A CA with all the cells having linear rules is called a *linear CA*. Table 1 shows linear rules which are used in this paper. The *state transition matrix* T of an n-cell linear CA can be represented as the following tridiagonal matrix which is called the state transition matrix of the CA[8]. For the state transition matrix T the next state Y of X is $Y = TX$, where X is the present state of the CA.

$$T = \begin{pmatrix} d_1 & a_{1,2} & 0 & \cdots & 0 & 0 \\ a_{2,1} & d_2 & a_{2,3} & \cdots & 0 & 0 \\ 0 & a_{3,2} & d_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{n,n-1} & d_n \end{pmatrix} \tag{2.2}$$

An MACA with m attractors can be viewed as a pattern classifier. It classifies a given set of patterns into m distinct classes[13,14]. The following lemma shows properties of MACA.

Lemma 2.1.[10] For an n-cell d-depth MACA with characteristic polynomial $x^d(x+1)^{n-d}$ and minimal polynomial $x^d(x+1)$,

(i) the number of attractors is 2^{n-d}.

(ii) the number of states in each attractor tree is the same and equal to 2^d.

Remark A. Let $\mathbb{C}$ be a TPMACA whose char-

acteristic polynomial and minimal polynomial are the same as $x^{n-1}(x+1)$. Then from Lemma 2.1 the number of attractors of the state transition diagram of $\mathbb{C}$ is 2 and the number of states in each attractor tree is the same and equal to 2^{n-1}. Also the depth of each tree is $n-1$.

We want to use feature vectors in order to classify n-bit patterns. Let P be a test pattern set and let P be divided into two classes S_1 and S_2 where $S_1 \cap S_2 = \varnothing$. Let V be a vector satisfying $V \cdot x = 0$ and $V \cdot y \neq 0$ for all $x \in S_1$ and $y \in S_2$. This V is called a *feature vector*. Fig. 1 shows that the set of all 4-bit patterns is divided into two classes by V, where $V = (1001)$.

Lemma 2.2.[12,13] Let T_M be the state transition matrix of an n-cell TPMACA whose characteristic polynomial and minimal polynomial are the same as $x^{n-1}(x+1)$ corresponding to a feature vector V of length n. Then the following hold.

(i) $rank(T_M) = rank(T_M \oplus I) = n-1$, where I is the $n \times n$ identity matrix.

(ii) $rank(T_M^{n-1}) = 1$ and the nonzero row of T_M^{n-1} is V.

(iii) The number of 1's of the diagonal elements in T_M is odd.

Remark B. Let $\mathbb{C}$ be an n-cell CA and let T_M be the state transition matrix of $\mathbb{C}$ satisfying the conditions (i) and (ii) in Lemma 2.2. Then $\mathbb{C}$ becomes a TPMACA with two trees. In fact (i) and (ii) in Lemma 2.2 are necessary and sufficient condition for $\mathbb{C}$ to be TPMACA with two trees.

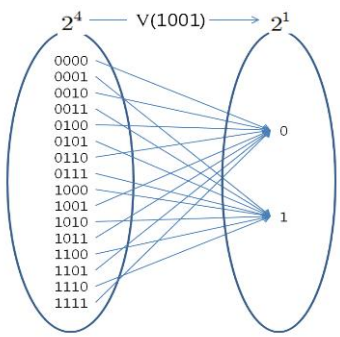

Fig. 1. Pattern classification for V = (1001).

The following special forms of vectors V_{S_1} and V_{S_2} can not synthesize TPMACA[12,13].

$$V_{S_1} = (0 v_2 v_3 \cdots v_{n-1} 0) \tag{2.3}$$

$$V_{S_2} = (v_1 \cdots v_i 0 \cdots 0 v_{i+k+1} \cdots v_n) \tag{2.4}$$

(where $v_i = 1, v_{i+k+1} = 1$ and $k \geq 2$)

3. ALGORITHMS FOR GENERATION OF A LINEAR NONGROUP CA CORRESPONDING TO FEATURE VECTORS

3.1 Generation of TPMACA corresponding to feature vectors

In this subsection we propose a method of synthesizing an n-cell TPMACA whose characteristic polynomial and minimal polynomial are the same as $x^{n-1}(x+1)$ by decomposing the given feature vector into the basic forms in Table 2. In order to synthesize the TPMACA corresponding to the given feature vector V, let $(10\cdots0)$, $(0\cdots01)$, $(1\cdots1)$ and (101) be the basic forms of V. For example if $V = (00111101)$, then V consists of the basic forms (001), (1111) and (101). Table 2 shows rules of TPMACA corresponding to basic forms of V. The $(i+1,i)$ entry of the state transition matrix $T_M = (t_{ij})$ of an n-cell TPMACA corresponding to the basic forms of type (i), type (iii) or type (v) is $t_{i+1,i} = 1$. Also the $(i,i+1)$ entry of T_M corresponding to the basic forms of type (ii), type (iv)

Table 2. Rules of TPMACA corresponding to feature vector V

Type	Basic forms of V	Rules of TPMACA
(i)	$(10\cdots0)$	$60, 240, \cdots, 240$
(ii)	$(0\cdots01)$	$170, \cdots, 170, 102$
(iii)	$(1\cdots1)$	$240, \cdots, 240, 60$
(iv)	$(1\cdots1)$	$102, 170, \cdots, 170$
(v)	(101)	$150, 60, 150$
(vi)	(101)	$150, 102, 150$

or type (vi) is $t_{i,i+1}=1$. Therefore, the rules used to synthesize TPMACA will be synthesized by using rules of type (i), type (iii) or type (v), or synthesized by using rules of type (ii), type (iv) or type (vi).

All the feature vectors can be decomposed into basic forms in Table 2. For example, if $V=(00111101011)$, then V is decomposed into $\mathbf{v_1}=(001)$, $\mathbf{v_2}=(1111)$, $\mathbf{v_3}=(101)$, $\mathbf{v_4}=(101)$ and $\mathbf{v_5}=(11)$. Hereafter we represent V as $V=[\mathbf{v_1}\mathbf{v_2}\mathbf{v_3}\mathbf{v_4}\mathbf{v_5}]$.

Suppose that V is an n-bit vector and V can be decomposed into k number of $\mathbf{v_i}(i=1,2,\cdots,k)$. And if the length of each $\mathbf{v_i}$ is n_i, then the following holds.

$$n=\left(\sum_{i=1}^{k} n_i\right)-k+1 \tag{3.1}$$

If $\mathbb{C}$ is an n-cell TPMACA corresponding to $V=(v_1v_2\cdots v_n)(v_i\in\{0,1\})$, then we can synthesize the pattern classifier using the following algorithm. The following algorithm shows a synthesis algorithm of a TPMACA $\mathbb{C}$ corresponding to $V=(v_1v_2\cdots v_n)$.

Algorithm 1 SynthesisOfTPMACA

Step 1. If $V=(1v_2v_3\cdots v_{n-1}0)$, then we synthesize by using type (iii) and type (v), and finally by using type (i).

Step 2. If $V=(0v_2v_3\cdots v_{n-1}1)$, then we synthesize by using type (ii) firstly, and then synthesize by using type (iv) and type (vi).

Step 3. If $V=(1v_2v_3\cdots v_{n-1}1)$, then we synthesize by using type (iii) and type (v), or by using type (iv) and type (vi).

In case of $V=(0v_2v_3\cdots v_{n-1}0)$, there does not exist an n-cell TPMACA corresponding to V. But, in the subsection 3.2, we can show that there exists an n-cell MACA corresponding to V by concatenating TPMACA which are obtained by Algorithm 1.

The following shows important conditions that must be taken into account when synthesizing TPMACA using the basic forms.

Theorem 3.1. Let $\mathbb{C}$ be an n-cell TPMACA corresponding to the feature vector $V=[\mathbf{v_1}\mathbf{v_2}\cdots\mathbf{v_k}]$ $(k\le n)$. Then the following hold.

(1) If $\mathbb{C}$ is synthesized by using type (i), type (iii) and type (v), then the last rule of the $\mathbf{v_i}$ is changed by the first rule of $\mathbf{v_{i+1}}$.

(2) If $\mathbb{C}$ is synthesized by using type (ii), type (iv) and type (vi), then the first rule of the $\mathbf{v_{i+1}}$ is changed by the last rule of $\mathbf{v_i}$.

Proof (1) $[\mathbf{v_i}\mathbf{v_{i+1}}]$ can be decomposed into type (v, iii), type (v, i), type (v, v), type (iii, i) and type (iii, v), where type (i), type (iii) and type (v) are in Table 2. There are five possible cases to consider. Firstly we show (1) for the case of type (v, iii). In this case $\mathbf{v_i}=(101)$ and $\mathbf{v_{i+1}}=(11\cdots1)$. The rule vector of 3-cell CA corresponding to $\mathbf{v_i}$ is <150, 60, 150>. Also the rule vector of n_{i+1}-cell CA corresponding to $\mathbf{v_{i+1}}$ is <240, ⋯, 240, 60>. If <150, 60, **240**, ⋯, 240, 60> is the rule vector of $(2+n_{i+1})$-cell CA $\mathbb{C}$, then the state transition matrix T of $\mathbb{C}$ is $T=\begin{pmatrix}1&1&0&0&\cdots&0&0\\1&1&0&0&\cdots&0&0\\0&1&0&0&\cdots&0&0\\0&0&1&0&\cdots&0&0\\\vdots&\vdots&\vdots&\vdots&\ddots&\vdots&\vdots\\0&0&0&0&\cdots&1&1\end{pmatrix}$. The number of 1's of the diagonal elements in T is 3 and $rank(T)=n-1$, and $rank(T\oplus I)=n-1$. Thus by Remark B, $\mathbb{C}$ is a $(2+n_{i+1})$-cell TPMACA. Let $T^i=(t_{jk}^{(i)})(i=2,\cdots,n_{i+1})$. Then

$$t_{jk}^{(i)}=\begin{cases}1, & j=i+1,k=1,2\\1, & j=i+k,k=2,\cdots,n-i\\1, & j=n,k=n-i+1,\cdots,n\\0, & o/w\end{cases} \tag{3.2}$$

By (3.2) $T^{n_{i+1}}=\begin{pmatrix}0&0&0&\cdots&0\\\vdots&\vdots&\vdots&\ddots&\vdots\\0&0&0&\cdots&0\\1&1&0&\cdots&0\\0&1&1&\cdots&1\end{pmatrix}$

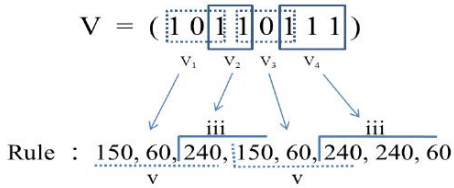

Fig. 2. The synthesis of TPMACA.

and so $T^{n_{i+1}+1}=\begin{pmatrix}0&0&0&\cdots&0\\ \vdots&\vdots&\vdots&\ddots&\vdots\\ 0&0&0&\cdots&0\\ 0&0&0&\cdots&0\\ 1&0&1&\cdots&1\end{pmatrix}$.

Therefore by Lemma 2.2(ii), $\mathbb{C}$ is the TPMACA corresponding to $V=(101\cdots1)$. The proof of (1) for the cases of type (v, i), type (v, v), type (iii, i) and type (iii, v) is similar to the proof for type (v, iii). (2) It can be proved by the similar method as (1).

Example 3.2. Let $V=(10110111)=[\mathbf{v_1v_2v_3v_4}]$, where $\mathbf{v_1}=\mathbf{v_3}=(101)$, $\mathbf{v_2}=(11)$ and $\mathbf{v_4}=(111)$. Then the synthesized CA is an 8-cell TPMACA with the rule vector <150, 60, 240, 150, 60, 240, 240, 60>. Fig. 2 shows the synthesis of TPMACA with the rule vector <150, 60, 240, 150, 60, 240, 240, 60>.

Table 3 shows the synthesis of TPMACA corresponding to 10-bit V. The fifth V in Table 3 can also be synthesized with type (vi), type (vi), type (vi) and type (iv) in Table 2. In this case the rule vector of the TPMACA is <150, 102, 150, 102, 150, 102, 150, 170, 170, 170>.

The following theorem shows a method for finding a nonreachable state in the 0-tree of TPMACA synthesized by Algorithm 1 and Theorem 3.1.

Theorem 3.3. Let $V_1=(1v_2v_3\cdots v_{n-1}0)$ (resp. $V_2=(0v_2v_3\cdots v_{n-1}1)$ and $V_3=(1v_2v_3\cdots v_{n-1}1)$) be the n-bit feature vector. Then the following state N_S is a nonreachable state of the 0-tree in the state transition diagram of the n-cell TPMACA $\mathbb{C}$ obtained by Algorithm 1 and Theorem 3.1.

$$N_{S_1}=\begin{cases}(110\cdots0), & v_2=1\\ (1010\cdots0), & v_2=0\end{cases} \tag{3.3}$$

$$\left(resp.\ N_{S_2}=\begin{cases}(0\cdots0011), & v_{n-1}=1\\ (1010\cdots0), & v_{n-1}=0\end{cases}\text{ and }N_{S_3}=\begin{cases}N_{s_1}, & t_{i+1,i}=1\\ N_{S_2}, & t_{i,i+1}=1\end{cases}\right) \tag{3.4}$$

where t_{ij} is the (i,j) entry of the state transition matrix $T_M=(t_{ij})$ of $\mathbb{C}$.

Proof. Let $V_1=(1v_2v_3\cdots v_{n-1}0)$ and $v_2=1$. Then $N_S=(110\cdots0)$ and $V=(11v_3\cdots v_{n-1}0)$. Since $V\cdot N_S=0$, N_S is a state of the 0-tree in the state transition diagram of $\mathbb{C}$. Let T_M be the state transition matrix of $\mathbb{C}$. Then $T_M^{n-1}N_S^t=\mathbf{0}$ because the depth of the 0-tree is $n-1$. To show that N_S is a nonreachable state we must show that $T_M^{n-2}N_S^t\neq\mathbf{0}$. Since $V=(11v_3\cdots v_{n-1}0)$, V is one of the forms $(111*\cdots*0)$, $(1101*\cdots*0)$ or $(1100\cdots0)$. Thus by Theorem 3.1 and Table 2 the first column of T_M and second column of T_M is $(010\cdots0)^t$ and $(0*10\cdots0)^t$ respectively. If c_i is the ith column of T_M^{n-2}, then $T_M^{n-2}N_S^t=c_1+c_2$. On the other hand, $\mathbb{C}$ is synthesized by Algorithm 1 with type (i), type (iii) and type (v) in Table 2. Thus

Table 3. Synthesis of TPMACA corresponding to 10-bit V

V	Decomposition of V_i	Basic forms (type)	Rules of TPMACA
1010111101	(101)(101)(1111)(101)	v,v,iii,v	150,60,150,60,240,240,240,150,60,150
1110110111	(111)(101)(11)(101)(111)	iii,v,iii,v,iii	240,240,150,60,240,150,60,240,240,60
0110111011	(01)(11)(101)(111)(101)(11)	ii,iv,vi,iv,vi,iv	170,102,170,102,150,170,170,102,150,170
0000110101	(00001)(11)(101)(101)	ii,iv,vi,vi	170,170,170,170,102,170,102,150,102,150
1010101111	(101)(101)(101)(1111)	v,v,v,iii	150,60,150,60,150,60,240,240,240,60

we can get the (j,k) entry $t_{jk}^{(i)}$ of T_M^i $(i \le n-2)$ as the following :

$$t_{jk}^{(i)} = \begin{cases} * & , \ j < i+k \\ 1 & , \ j = i+k \\ 0 & , \ i+k < j \le n \end{cases} \quad (3.5)$$

Therefore $c_1 = (0 * \cdots 10)^t$ and $c_2 = (0 * \cdots * 1)^t$. So $c_1 + c_2 \neq 0$. Hence $T_M^{n-2} N_S^t \neq 0$ and thus N_S is a nonreachable state of the 0-tree of $\mathbb{C}$. We can show that remaining states N_S are all nonreachable states by the similar method.

Corollary 3.4. Let V be the n-bit feature vector $V = (10 \cdots 0)$ (resp. $V = (0 \cdots 01)$). Then $N_S = (0 \cdots 01)$ (resp. $N_S = (010 \cdots 0)$) is a nonreachable state of the 0-tree in the state transition diagram of the n-cell TPMACA $\mathbb{C}$ obtained by Algorithm 1.

3.2 Generation of MACA corresponding to feature vectors

Now consider the case there is no TPMACA for a given feature vector V_S of the form in (2.3) and (2.4). So in this subsection we propose a method for synthesizing an n-cell MACA for such feature vectors. Let $V_S = (V_1 \| V_2 \| \cdots \| V_k)$ be the concatenation of $V_i (i = 1,2,\cdots,k)$. Here V_i is the feature vector of a TPMACA $\mathbb{C}_i (i = 1,2,\cdots,k)$ generated by Algorithm 1. And let T_{M_i} be the state transition matrix of $\mathbb{C}_i$ whose characteristic polynomial and minimal polynomial are the same as $x^{d_i}(x+1)$, where the length of V_i is $n_i = d_i + 1$. Let

$$T = \begin{pmatrix} T_{M_1} & O & \cdots & O \\ O & T_{M_2} & \cdots & O \\ \vdots & \vdots & \ddots & \vdots \\ O & O & \cdots & T_{M_k} \end{pmatrix} \quad (3.6)$$

and let $\mathbb{C}$ be the n-cell MACA whose state transition matrix is T, where $n = n_1 + n_2 + \cdots + n_k$. Then this $\mathbb{C}$ is the MACA corresponding to V_S. The characteristic polynomial and the minimal polynomial of T is $x^{n-k}(x+1)^k$ and $x^d(x+1)$ respectively, where $d = \max(d_1, d_2, \cdots, d_k)$. Also the number of trees in the state transition diagram of $\mathbb{C}$ is 2^k.

To increase the efficiency of pattern classifier we need to minimize d and k. Especially to minimize d it is necessary for V_i to have similar lengths.

The following algorithm is a synthesis algorithm of linear MACA which is the pattern classifier corresponding to $V_S = (V_1 \| V_2 \| \cdots \| V_k)$.

Algorithm 2 Synthesis Of Linear MACA Pattern Classifier

Step 1. Decompose the given feature vector V_S into the minimum number of V_i's with similar sizes.

Step 2. Synthesize TPMACA $\mathbb{C}_i$ corresponding to V_i by Algorithm 1.

Step 3. Concatenate $\mathbb{C}_i$'s $(i = 1,2,\cdots,k)$.

Step 4. Change the jth rule and the $(j+1)$th rule by using Table 4.

Table 4 shows change of the state transition rules occurring in the synthesis process of linear MACA.

Example 3.5. Let $V_S = (0111010)$. Then $V_S = (W_1 \| W_2) = (01 \| 11010)$ or $V_S = (V_1 \| V_2) = (011 \| 1010)$. The minimal polynomial of the pattern classifier synthesized using $V_S = (W_1 \| W_2) = (01 \| 11010)$ is $x^4(x+1)$. But the minimal polynomial of the pattern classifier synthesized using $V_S = (V_1 \| V_2) = (011 \| 1010)$ is $x^3(x+1)$. Therefore, it is more effective to decompose V_S into $V_S = (V_1 \| V_2)$ rather than $V_S = (W_1 \| W_2)$. Thus by Algorithm 1, $\mathbb{C}_1$ and $\mathbb{C}_2$ are <170,102,170> and <150,60,60,240> respectively. Therefore, the linear MACA $\mathbb{C}$ corresponding to V_S is $\mathbb{C} = <\mathbb{C}_1 \| \mathbb{C}_2> = <170,102,0,102,60,60,240>$ by Algorithm 2 and Table 4.

Fig. 3 shows the pattern classifier corresponding

Table 4. Change of the state transition rules occurring in the synthesis process of linear MACA

The jth rule (The last rule of V_i)		The $(j+1)$th rule (The first rule of V_{i+1})	
Present rule	Next rule	Present rule	Next rule
240	24	240	0
102	204	102	102
60	60	60	204
170	0	170	170
150	60	150	102

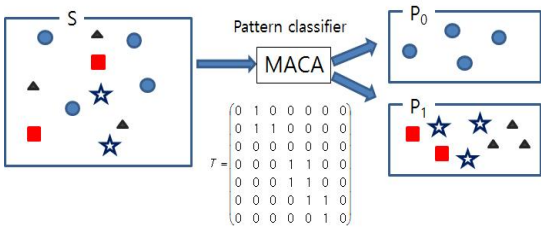

Fig. 3. Pattern classifier corresponding to the feature vector $V_S=(0111010)$.

to feature vector $V_S=(0111010)$. In Fig. 3, $S=\{0,1,2,\cdots,126,127\}$ and $V_S=(0111010)$. By using Algorithm 1 and Algorithm 2 we obtain T. Then we can classify S into P_0 and P_1 with T. Here $P_0=\{0,1,4,5,10,\cdots,117,122,127\}$ and $P_1=S\setminus P_0=\{2,3,6,7,\cdots,125,126\}$.

In Example 3.5 $N_{S_1}=(011)$(resp. $N_{S_2}=(1010)$) is the nonreachable state of $\mathbb{C}_1$(resp. $\mathbb{C}_2$) by Theorem 3.3. Thus the nonreachable states of $\mathbb{C}$ can be obtained by linear combination of $(N_{S_1}\|\mathbf{0})$ or $(\mathbf{0}\|N_{S_2})$.

4. CONCLUSION

In this paper we proposed a pattern classifier based on a linear nongroup CA that can classify patterns according to a given feature vector and two algorithms for generations of TPMACA and MACA corresponding to the given feature vectors. In addition, we found methods for finding nonreachable states in the 0-tree of the state transition diagram of TPMACA corresponding to the given feature vector for the analysis of the state transition behavior of the generated pattern classifier.

REFERENCE

[1] P. Maji, C. Shaw, N. Ganguly, B. Sikdar, and P.P. Chaudhuri, "Theory and Application of Cellular Automata for Pattern Classification," *Fundamenta Informaticae*, IOS Press, Vol. 58, No. 3-4, pp. 321- 354, 2003.

[2] R.D. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification,* Wiley-Interscience, USA, 2001.

[3] J.V. Neumann, *The Theory of Self-reproducing Automata*, University of Illinois Press, Urban and London, 1966.

[4] S.J. Cho, U.S. Choi, and H.D. Kim, "Analysis of Complemented CA Derived from a Linear TPMACA," *Computers and Mathematics with Applications*, Vol. 45, No. 4-5, pp. 680-698, 2003.

[5] P. Dasgupta, P.P Chattopadhyay, P.P. Chaudhuri, and I. Sengupta, "Cellular Automata-based Recursive Pseudoexhaustive Test Pattern Generator," *IEEE Transaction on Computers*, Vol. 50, No. 2, pp. 177-185, 2001.

[6] A.K. Das and P.P. Chaudhuri, "Vector Space Theoretic Analysis of Additive Cellular Automata and Its Application for Pseudo-exhaustive Test Pattern Generation," *IEEE Transaction on Computers*, Vol. 42, No. 2, pp. 177-185, 2001.

[7] S. Bandini, L. Vanneschi, A. Wuensche, and A.B. Shehata, "Cellular Automata Pattern Recognition and Rule Evolution through a Neuro-genetic approach," *Journal of Cellular Automata*, Vol. 4, No. 3, pp. 171-181, 2009.

[8] S.J. Cho, U.S. Choi, H.D. Kim, Y.H. Hwang, J.G. Kim, and S.H. Heo, "New Synthesis of One-dimensional 90/150 Linear Hybrid Group Cellular Automata," *IEEE Transaction on Computer-Aided Design of Integrated Circuits Systems*, Vol. 26, No. 9, pp. 1720-

1724, 2007.

[9] J.S. Lee, H.H. Cho, K.H. Rhee, "Cellular Automata and it Applications," *Journal of Korea Multimedia Society*, Vol. 6, No. 4, pp. 610-619, 2003.

[10] P.P. Chaudhuri, D.R. Chowdhury, S. Nandy, and C. Chattopadhyay, *Additive Cellular Automata Theory and Applications,* IEEE Computer Society Press, California, 1997.

[11] S.J. Cho, U.S. Choi, H.D. Kim, Y.H. Hwang, and J.G. Kim, "Analysis of 90/150 Two Predecessor Nongroup Cellular Automata," *Lecture Notes in Computer Science,* Vol. 5191, pp. 128-135, 2008.

[12] S.J. Cho, H.D. Kim, U.S. Choi, S.T. Kim, J.G. Kim, S.H. Kwon, et al., "Generation of TPMACA for Pattern Classification," *Lecture Notes in Computer Science,* Vol. 8751, pp. 408-416, 2014.

[13] N. Ganguly, *Cellular Automata Evolution : Theory and Applications in Pattern Recognition and Classification*, Doctor's Thesis of CST Department BECDU India, 2003.

[14] J. Ponkaew, S. Wongthanavasu, and C. Lursinsap, "A Nonlinear Classifier using an Evolution of Cellular Automata," *Proceeding of International Symposium on Intelligent Signal Processing and Communications Systems,* pp. 1-5, 2011.

Un-Sook Choi

received the M.S. degree and the Ph.D. degree at Pukyong National University. She is currently a professor at Tongmyong University since 2006. Her research interests include cryptography, cellular automata and its applications.

Sung-Jin Cho

received the M.S. degree and the Ph.D. degree at Korea University. He is currently a professor at Pukyong National University since 1988. His research interests include cryptography, cellular automata and its applications.

Han-Doo Kim

received the M.S. degree and the Ph.D. degree at Korea University. He is currently a professor at Inje University since 1989. His research interests include finite field theory, discrete mathematics and cellular automata.