Fundamenta Informaticae 78 (2007) 369–396

RBFFCA: A Hybrid Pattern Classifier Using Radial Basis Function and Fuzzy Cellular Automata

Pradipta Maji*
Machine Intelligence Unit
Indian Statistical Institute
203 B. T. Road, Kolkata, 700 108, India
pmaji@isical.ac.in

P Pal Chaudhuri
Cellular Automata Research Laboratory (CARL)
Techno India Campus, EM 4/1, Sector V
Kolkata, 700 091, India
palchau@carltig.res.in

Abstract. A hybrid learning algorithm, termed as RBFFCA, for the solution of classification problems with real valued inputs is proposed. It comprises an integration of the principles of radial basis function (RBF) and fuzzy cellular automata (FCA). The FCA has been evolved through genetic algorithm (GA) formulation to perform pattern classification task. The versatility of the proposed hybrid scheme is illustrated through its application in diverse fields. Simulation results conducted on benchmark database show that the hybrid pattern classifier achieves excellent performance both in terms of classification accuracy and learning efficiency. Extensive experimental results supported with analytical formulation establish the effectiveness of RBFFCA based pattern classifier and prove it as an efficient and cost-effective alternative for the classification problem.

Keywords: Cellular Automata, Fuzzy Cellular Automata, Radial Basis Function, Genetic Algorithm, Pattern Classification.

*Address for corespondence: Machine Intelligence Unit, Indian Statistical Institute, 203 B. T. Road, Kolkata, 700 108, India.

1. Introduction

From the days of Von Neumann [39] to the recent days of S. Wolfram [53], cellular automata (CA) has dominated computing technology for doing computation more efficiently - in terms of speed, cost, power dissipation, information storage, and solution quality. It has been proposed to study the general phenomenological aspects, including communication, computation, construction, growth, reproduction, competition, and evolution [28, 48]. CA also provides an excellent tool for modeling physical phenomena by reducing them to their basic, elemental laws [47, 52].

Interesting computational properties of the CA model has inspired us to investigate new application avenues. Pattern classification is an important and interdisciplinary research area spanning several disciplines such as database systems [46], machine learning [17, 35, 37], intelligent information systems, statistics [19, 26, 44], and expert systems. Many new approaches are being introduced [34, 43], as well as existing ones getting refined [5, 10, 22, 24, 29, 40]. However, search for new and better solutions continues, specifically to classify large volume of dataset generated in the internet-worked society of cyber-age.

In the above scenario, design of CA based model for pattern recognition are reported in [7, 25, 38, 41]. Robust model of CA based associative memory is introduced in [21, 31]. The concept of cellularity embedded in neural network structure has given rise to the popular concept of cellular neural networks [3, 12, 13]. Tzionas et al. [49, 51] proposed a hybrid scheme for multi-valued pattern classification using the parallel architecture that employs a two dimensional CA combined with a single layer perceptron architecture. Tzionas et al. [50] also presented another variation of a CA based pattern classifier based on a nearest neighborhood discriminant. Many concepts from the discipline of biology have been borrowed to build the CA based clustering model. One such model mimics the behavior of ants to gather and sort corpses in a self-organized manner [11, 23]. The fact that the special class of CA referred to as multiple attractor CA (MACA) can act as a natural pattern classifier is pointed in [9]. Chattopadhyay et al. [8] have recently refined this concept to develop CA based pattern classification model. Ganguly et al. [20, 42, 45] further characterized the MACA basins to propose robust models for pattern classification and associative memory. Comparison of performance of MACA based classifier and conventional schemes like decision tree, multi-layer perceptron are investigated by Maji et al. [32, 33].

However, CA based pattern classifier proposed in [8, 20, 32] can handle attributes expressed as binary patterns even though real life applications demand classification of data involving real numbers. For the classifiers designed to handle binary data, an explicit or implicit discretization procedure is applied to cluster the continuous data of real numbers to a set of subintervals. However, most discretization procedures suffer from user's bias in generating the subintervals [14, 27]. Also, since discretization is performed on a finite training set, it is doubtful whether the clustered subintervals encapsulate the real distribution. Thus, some information may be lost in the transformation from continuous domain to finite subintervals and that will invariably degrade the quality of solution.

In this background, design of pattern classifier based on a special class of CA, termed as fuzzy CA (FCA), has been explored in [30] to address the problem of classification of patterns of real valued data. FCA is a conventional CA with fuzzy logic applied as next state function of a cell [1, 6, 18]. Design of tree-structured pattern classifier based on a special class of FCA, termed as fuzzy multiple attractor CA (FMACA), has been proposed in [30]. In general, FMACA based pattern classifier can handle continuous attributes due to their powerful nonlinear processing ability. Therefore we believe that a strong learning paradigm can be attained through FMACA based pattern classifier.

In the current paper, we consolidate and refine the design approach of [30] while integrating the principles of radial basis function (RBF) and fuzzy cellular automata (FCA) for designing an efficient pattern classifier. The major contributions of this paper are summarized below:

1. A hybrid pattern classifier is proposed based on the theory of RBF and FMACA.

2. Two new operators, dependency vector (DV) and derived complement vector (DCV), are introduced for analysis and synthesis of FMACA.

3. The analysis of FMACA based on DV and DCV is next combined with genetic algorithm (GA) to formulate an elegant evolutionary scheme. The genetic operators are implemented in such a way that they help to preserve the structure of FMACA.

4. Extensive experimental results establish that the classification accuracy of the proposed hybrid scheme is comparable while its memory overhead and retrieval time are very lesser compared to conventional classification algorithms.

In order to realize specified objectives, we introduce FCA preliminaries including FMACA fundamentals in Section 2. The concept of DV and DCV is introduced in Section 3. Design of pattern classifier based on RBF and FCA is presented in Section 4. An evolutionary synthesis scheme is next presented in Section 5 employing DV and DCV. Finally, applications of the proposed classifier in image classification, finding splice-junction and protein-coding regions of DNA sequences are reported in Section 6.

2. Fuzzy Cellular Automata

An elementary FCA [6, 18] is a linear array of cells which evolves in time. Each cell of the array assumes a state q_i, a rational value in the interval $[0, 1]$ (fuzzy states) and changes its state according to a local evolution function on its own state and states of its two neighbors. The global evolution results from synchronous application of the local rule to all cells of the array. In a FCA, the conventional Boolean functions are evaluated as follows:

Boolean Function	Operation	FCA Operation
OR	$a + b$	$\min\{1, a + b\}$
AND	ab	$a \cdot b$
NOT	$\overline{a}$	$(1 - a)$

Here a and b are two states having rational values in the interval [0, 1]. The resulting local rule of FCA is a real valued function [6, 18]. In a 3-neighborhood FCA, there are total 256 distinct next state functions (rules). An n-cell FCA is configured with the rule vector $\mathcal{R}$= $< \mathcal{R}_1, \cdots, \mathcal{R}_i, \cdots, \mathcal{R}_n >$ where i^{th} cell is configured with rule $\mathcal{R}_i$; each $\mathcal{R}_i$ being one of the possible 256 rules. Out of 256 rules, there are only 16 rules with OR and NOR logic [30]. A FCA rule involving NOR logic is referred to as complemented FCA rule; otherwise, it is a non-complemented FCA rule.

For an n-cell FCA, an n-tuple rule vector $\mathcal{R}$ with only OR and NOR rules can be represented by an $n \times n$ matrix T and an n-dimensional binary vector F [30]. If $\mathcal{P}_i(t)$ represents the state assignment of

the i^{th} cell of a FCA at t^{th} instant of time, the state of i^{th} cell at $(t+1)^{th}$ instant of time is

$$\mathcal{P}_i(t+1) =\mid F_i - \min\{1, \sum_{j=1}^{n} T_{ij} \cdot \mathcal{P}_j(t)\} \mid \tag{1}$$

where T is an $n \times n$ binary matrix and F is an n-bit binary vector, termed as complement vector (CV).

Example 2.1. For rule vector $< 238, 1, 238, 3 >$, T corresponds to rule vector $< 238, 254, 238, 252 >$. In this example, 2nd and 4th cells employ complemented rules. Hence,

$$T = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad F = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Next subsection introduces FMACA. The classifier proposed in Section 4 is built around this FCA.

Fuzzy Multiple Attractor CA

A FMACA is a special class of FCA that can efficiently model an associative memory to perform pattern classification task [30]. Its state transition behavior consists of multiple components - each component, as noted in Fig.1, is an inverted tree. A node with self loop is referred to as an attractor state. The states in the tree rooted on an attractor form an attractor basin. The states in a basin other than the attractor are referred to as transient states in the sense that a FMACA finally settles down in one of its attractor state after passing through such transient states.

An n-cell FMACA with k-attractor basins can be viewed as a natural classifier [30]. It classifies a given set of patterns into k distinct classes, each class containing the set of states in the attractor basin. The following example illustrates a FMACA based two class classifier.

Example 2.2. Let us have two pattern sets S_1 ={(0.00, 0.00, 0.50), (0.00, 0.25, 0.00), (0.25, 0.25, 0.00), (0.00, 0.50, 0.00), (0.00, 0.00, 0.00), (0.25, 0.00, 0.00), (0.50, 0.00, 0.00), (0.00, 0.00, 0.25), (0.00, 0.00, 0.75), (0.00, 0.50, 0.25)} (Class I) and S_2 ={(0.75, 1.00, 0.00), (1.00, 0.75, 0.50), (1.00, 1.00, 1.00), (0.75, 1.00, 1.00), (1.00, 1.00, 0.75), (1.00, 0.75, 1.00), (0.50, 0.75, 1.00), (1.00, 0.75, 0.75), (0.75, 1.00, 0.75), (0.75, 0.75, 1.00)} (Class II) with three attributes. In order to classify these two pattern sets into two distinct classes - Class I and II respectively, we have to design a FMACA such that the patterns of each class falls in distinct attractor basins of a FMACA.

The FMACA of Fig.1 is able to classify the patterns into distinct attractor basins where Class I (S_1) is represented by one set of attractor basins with attractors {(0.00 0.00 0.00), (0.25 0.25 0.00), (0.50 0.50 0.00) and (0.75 0.75 0.00)} in Fig.1 while Class II (S_2) is represented by the remaining basin with attractor {(1.00 1.00 0.00)}. When the FMACA is loaded with an input pattern say $\mathcal{P}$ =(0.75 0.75 0.50) and is allowed to run in autonomous mode, it travels through a number of transient states and ultimately reaches an attractor state (1.0 1.0 0.0) - the attractor representing Class II.

To identify the class of an input pattern $\mathcal{P}$, the FMACA is initialized with $\mathcal{P}$ and operated for d time steps where d is depth of the FMACA. In maximum of d time steps, the FMACA reaches the attractor

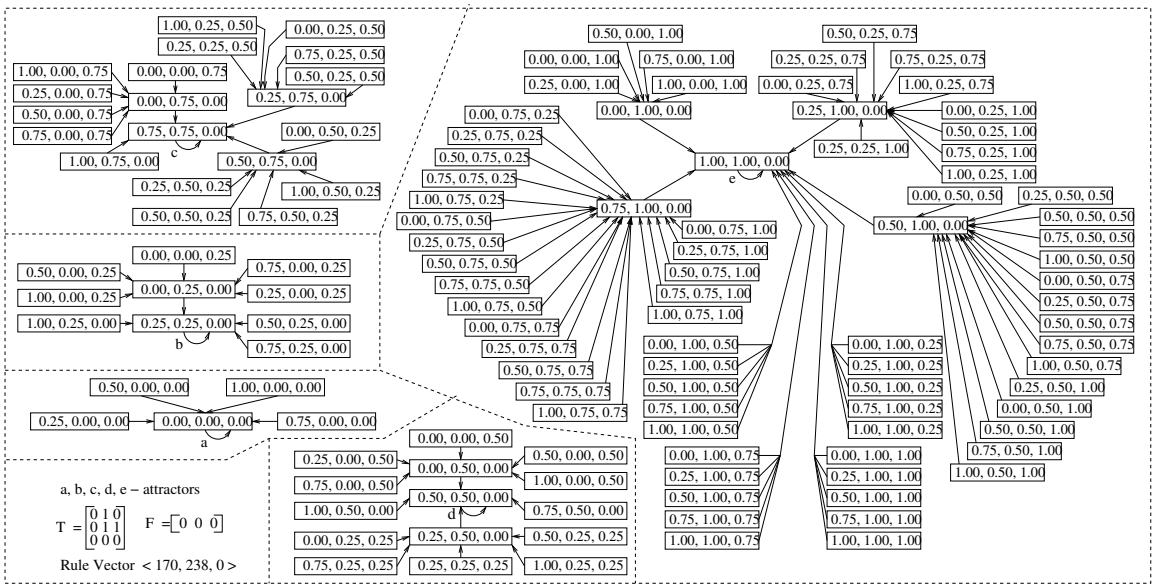

Figure 1. State space of a 3-cell 5-state FMACA divided into five attractor basins

state. The state of the pivot cell (PC) of attractor points to the memory location that stores the class information of the input pattern $\mathcal{P}$. The concept of PC has been formalized in Definition 3.2 of Section 3.1 subsequent to characterization of FMACA attractor basins. For the example FMACA of Fig.1, 2nd cell is the PC.

However, an n-cell FMACA can be characterized by its $n \times n$ T matrix and its complement vector F. So, generation of an attractor state $\mathcal{P}(t+d)$ from any state $\mathcal{P}(t)$ involves O(n^3) complexity. Consequently, the identification of the class in which any state $\mathcal{P}$ belongs, involves O(n^3) complexity. In order to ensure the scalability of present day's classification on very large datasets, linear complexity of algorithm is highly desirable. This motivates us to undertake new characterization of FMACA with the help of some linear operators other than T and F. Next section presents the characterization of FMACA.

3. Characterization of FMACA

In this section, we characterize attractor basins of FMACA. The analysis reported in this section bring down the complexity from O(n^3) to O(n) to identify the basin and consequently class of a pattern.

3.1. Non-complemented FMACA

The characterization of non-complemented FMACA proceeds under the following conjecture that is derived out of extensive experimentation.

Conjecture 1. If the number of attractor basins (k) of a FMACA is equal to $\mathcal{K}^m$ where $\mathcal{K}$ is the number of fuzzy states and $m = 1, 2, \cdots, n$, there exist m dependency relations among all vectors of each basin.

Example 3.1. Fig.1 is used to illustrate the above concept. Consider an n-dimensional (=3) vector space with $\mathcal{K}$ (=5) fuzzy states, - that is, 0.00, 0.25, 0.50, 0.75 and 1.00. Then, total possible vectors in the vector space is $\mathcal{K}^n$ (= 5^3 = 125). The n-dimensional vector space is divided into 5 basins (k) - Basin I, II, III, IV and V. That is, $k = \mathcal{K}$ and $m = 1$. If the vectors of any basin is conceived as a system of equations with three variables (x_1, x_2, x_3), then

$$\min\{1, x_2 + x_3\} = \begin{cases} 0.00, & \text{for Basin I} \\ 0.25, & \text{for Basin II} \\ 0.50, & \text{for Basin III} \\ 0.75, & \text{for Basin IV} \\ 1.00, & \text{for Basin V} \end{cases}$$

Let a pattern $\mathcal{P} = (1.00\ 0.50\ 0.00) \in$ Basin III. In this case, $\min\{1, x_2 + x_3\} = \min\{1, 0.50\} = 0.50$.

In the above context, we next introduce the term dependency vector (DV).

Definition 3.1. The dependency vector (DV) represents each individual dependency relation satisfied by all the vectors in each attractor basin. The DV for the illustrative Example 3.1 is $< 011 >$. The bits in the DV represents the variable in the sequence $< x_1x_2x_3 >$. The 1's in the DV specify the dependent variables. In Example 3.1, x_2 and x_3 are dependent variables. The OR of the corresponding variables in all the vectors of an attractor basin is equal to one of the fuzzy states. Here, OR (addition) implies that $(a + b) = \min\{1, (a + b)\}$.

Thus, in an n-dimensional vector space with $\mathcal{K}$-fuzzy states, a FMACA having k-attractor basins can be characterized by m number of DVs if $k = \mathcal{K}^m$, where $m = 1, 2, \cdots, n$. For ease of subsequent discussions, we next introduce a few terminologies.

- $\mathcal{K}$ represents the number of fuzzy states.
- q_i represents a fuzzy state, $q_i \in [0, 1]$. That is, $q_i = \frac{i}{\mathcal{K}-1}$, where $i = 0, 1, \cdots, \mathcal{K} - 1$. For example, if $\mathcal{K} = 4$, a cell of a FMACA can assume a state q_i out of 4 fuzzy states - $q_0 = 0.00, q_1 = 0.33, q_2 = 0.67$ and $q_3 = 1.0$.
- V_{q_i} represents a basin in which sum (OR) of the dependent variables of any vector $v \in V_{q_i}$ is q_i.
- $\tilde{w}$ represents the number of dependent variables of a DV - that is, the number of 1's of DV is $\tilde{w}$.

Valid Dependency Vector

A 3-neighborhood FMACA whose next state depends on itself, its left and right neighbors, cannot produce all the variations of DV. The structure of DVs generated by FMACA is next elaborated with illustrative example.

Axiom 1. In case of a 3-neighborhood FMACA, a DV contains 1's in successive positions. That is, a 3-neighborhood FMACA can generate a DV with a running sequence of 1's like $<000 \cdots 11111 \cdots 0000>$.

Example 3.2. Some examples of DVs which can be generated by a 3-neighborhood FMACA are $<$ $001111000>$, $<1111000>$, $<001111>$, $<001000>$ and $<11111>$.

Consider an n-cell $\mathcal{K}$-attractor basins FMACA with a DV of the form $<00\cdots 1111\cdots 1111\cdots 000>$. The DV contains runs of 1's from i^{th} to j^{th} positions. The region i^{th} to j^{th} cell is defined as the dependent region (DR) of that FMACA, while i^{th} and j^{th} cells are termed as first cell (FC) and last cell (LC) of DR respectively. For ease of subsequent discussions, the i^{th} and j^{th} positions are referred to as first cell position (FCP) and last cell position (LCP) respectively. Consequently, k^{th} bit of DV is

$$DV_k = \begin{cases} 1, & \text{if FCP} \leq k \leq \text{LCP} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Let q_m represents a fuzzy state where $q_m = \frac{m}{\mathcal{K}-1}$ and $m = 0, 1, \cdots, \mathcal{K}-1$. For an attractor basin V_{q_m}, any vector $v \in V_{q_m}$ must satisfy the relation

$$\min\{1, \sum_{k=i}^{j} x_k\} = q_m \tag{3}$$

In this context, we next introduce pivot cell of an attractor.

Definition 3.2. The pivot cell (PC) of an attractor of a basin is in between FCP and LCP of the DR. If q_m represents state of the PC of attractor of basin V_{q_m}, then

$$q_m = \text{state of PC} = \min\{1, \sum_{k=i}^{j} x_k\}$$

Example 3.3. In Example 3.1 (Fig.1), $DV =< 011 >$, $\tilde{w}$=2. In this case, FCP=2, LCP=3 and PCP=2.

The pivot cell position (PCP) of an attractor may be in between FCP and LCP of the DR. The T matrix corresponding to a DV depends on PCP. The state of PC represents an attractor basin uniquely. It yields the address of memory that stores basin information.

The characterization based on DV establishes that a non-complemented rule vector $\mathcal{R}=<\mathcal{R}_1, \cdots, \mathcal{R}_i, \cdots, \mathcal{R}_j, \cdots, \mathcal{R}_n>$ can generate a DV in which successive 1's are placed in between i^{th} to j^{th} positions. The rule vector $\mathcal{R}$ depends on the number of dependent variables $\tilde{w}$ of DV and PCP of the DR. This leads to three specific cases of rule vectors.

1. If DV contains single 1 at the i^{th} position ($\tilde{w}$=1),
   - $\mathcal{R}_i = 204$; and
   - $\mathcal{R}_1 = \cdots = \mathcal{R}_{i-1} = \mathcal{R}_{i+1} = \cdots = \mathcal{R}_n = 0$.
2. If $\tilde{w} \geq 2$, then
   (a) for PCP=FCP=i
   - $\mathcal{R}_i = 238$;
   - $\mathcal{R}_{i+1} = \mathcal{R}_{i+2} = \cdots = \mathcal{R}_{j-1} = 170$; and
   - $\mathcal{R}_1 = \cdots = \mathcal{R}_{i-1} = \mathcal{R}_j = \mathcal{R}_{j+1} = \cdots = \mathcal{R}_n = 0$.

(b) for PCP=LCP=j

- $\mathcal{R}_j = 252$;
- $\mathcal{R}_{i+1} = \mathcal{R}_{i+2} = \cdots = \mathcal{R}_{j-1} = 240$; and
- $\mathcal{R}_1 = \cdots = \mathcal{R}_{i-1} = \mathcal{R}_i = \mathcal{R}_{j+1} = \cdots = \mathcal{R}_n = 0$.

3. If $\tilde{w} \geq 3$, then for PCP=x

- $\mathcal{R}_x = 254$ where $i < x < j$;
- $\mathcal{R}_{i+1} = \mathcal{R}_{i+2} = \cdots \mathcal{R}_{x-1} = 240$;
- $\mathcal{R}_{x+1} = \cdots = \mathcal{R}_{j-1} = 170$; and
- $\mathcal{R}_1 = \cdots = \mathcal{R}_i = \mathcal{R}_j = \cdots = \mathcal{R}_n = 0$.

To achieve wide variations in state transition behavior, we next characterize complemented FMACA.

3.2. Complemented FMACA

This subsection analyzes the state transition behavior of complemented FMACA with reference to its non-complemented counterpart. While the number of attractor basins and the number of state vectors in each basin of a complemented FMACA is same as that of non-complemented counterpart, there is a movement of vectors from one basin to another. That is, the structure of state space in terms of attractor basins is identical while the states covered by the basins differ as illustrated below.

Example 3.4. Fig.2 illustrates an example where some patterns are moved from one basin to another. Here, F has been changed from $[0\ 0\ 0]$ to $[0\ 1\ 1]$, while T remains same.

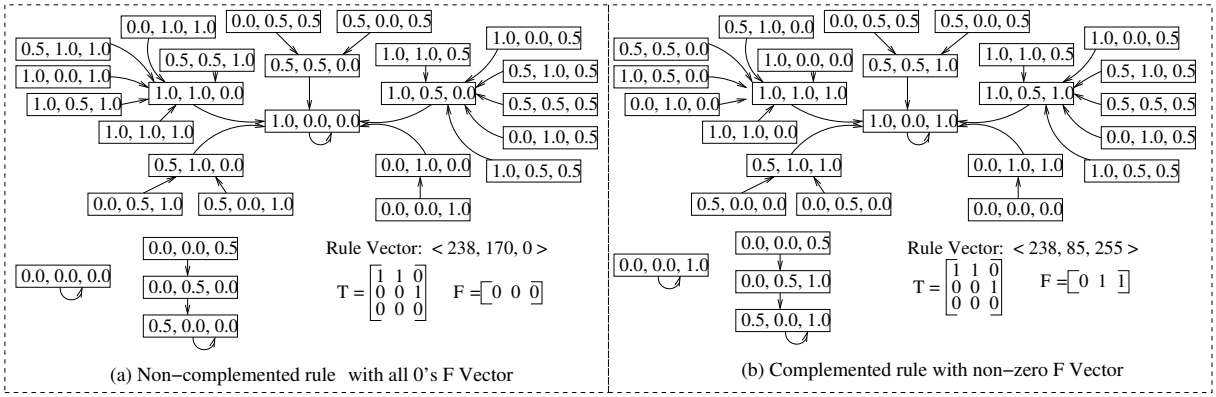

Figure 2. Modification of state transition behavior

Fig.2(a) represents state transition diagram of a 3-cell 3-state FMACA configured with non- complemented $\mathcal{R} = < 238, 170, 0 >$. In this case, DV is $< 111 >$, - that is, DR is from 1st to 3rd positions.

Thus, any vector $v \in V_{q_m}$ must satisfy Relation 3 where $q_m = 0.0, 0.5, 1.0$. That is,

$$\min\{1, \sum_{k=1}^{3} x_k\} = \begin{cases} 0.0, & \text{for Basin I } (V_{0.0}) \\ 0.5, & \text{for Basin II } (V_{0.5}) \\ 1.0, & \text{for Basin III } (V_{1.0}) \end{cases}$$

The following discussion analyzes the effect of n-dimensional complement vector (CV) F on the state transition diagram of an n-cell FMACA. Analysis is based on the DR of a DV with a running sequence of 1's from FCP to LCP. Each 1 in the sequence refers to dependency of the corresponding cell and consequently a dependent variable in the complemented FMACA while PCP refers to the pivot cell position. The characterization is based on PCP.

Conjecture 2. For a non-complemented rule vector $\mathcal{R} = <\mathcal{R}_1, \cdots, \mathcal{R}_i, \cdots, \mathcal{R}_j, \cdots, \mathcal{R}_n>$, if $\mathcal{R}_i$ to $\mathcal{R}_j$ is the DR - that is, the DV contains runs of 1's from i^{th} to j^{th} positions, then there exists a complemented rule vector $\acute{\mathcal{R}}$ where

- $\mathcal{R}_{\rm PC}$ (rule for pivot cell) remains same as that of non-complemented FMACA;
- in the DR, the number of complemented rules in each side of the PC is even; and
- any rule $\mathcal{R}_1$ to $\mathcal{R}_{i-1}$ and $\mathcal{R}_{j+1}$ to $\mathcal{R}_n$ may be complemented,

then, the structure of state space in terms of number and size of attractor basins of complemented FMACA remain identical to its non-complemented counterpart.

If complement (NOR) rules are applied on cells out side the DR (1 to $(i-1)$ and $(j+1)$ to n), Relation 3 remains same for each basin. But, if NOR rules are applied on the cells of DR (i to j), the number of NOR rules in each side of PCP must be even. In this case, if NOR rules are applied pairwise on $(\acute{i}_1, \acute{j}_1), (\acute{i}_2, \acute{j}_2), (\acute{i}_3, \acute{j}_3), \cdots$ and $(i_1, j_1), (i_2, j_2), (i_3, j_3), \cdots$ cell positions where

$$1 \le i \le \acute{i}_1 < \acute{j}_1 < \cdots < \text{PCP} < i_1 < j_1 < \cdots \le j \le n$$

then, $v \in V_{q_m}$ satisfies the relation

$$\min\{1, A + B\} = q_m \tag{4}$$

$$A = \{\sum_{k=i}^{\acute{i}_1 - 1} x_k + \sum_{k=\acute{i}_1}^{\acute{j}_1 - 1} (1 - x_k) + \sum_{k=\acute{j}_1}^{\acute{i}_2 - 1} x_k + \sum_{k=\acute{i}_2}^{\acute{j}_2 - 1} (1 - x_k) + \cdots + \sum_{k=\acute{j}_3}^{\text{PC}} x_k\}$$

$$B = \{\sum_{k=\text{PC}+1}^{i_1} x_k + \sum_{k=i_1+1}^{j_1} (1 - x_k) + \sum_{k=j_1+1}^{i_2} x_k + \sum_{k=i_2+1}^{j_2} (1 - x_k) + \cdots + \sum_{k=j_3+1}^{j} x_k\}$$

Thus, the number of attractor basins and number of vectors in each basin remain same. Only some state vectors will move from one attractor basin to another.

In the above context, we next introduce a parameter, termed as derived complement vector (DCV), which is derived from both DV and CV. To identify an attractor basin in O(n) time complexity, the DCV is employed. The details of this identification process is reported in Section 3.3.

Definition 3.3. If a DV contains runs of 1's from i^{th} to j^{th} positions and F contains 1 at $(\acute{i}_1, \acute{j}_1), (\acute{i}_2, \acute{j}_2), \cdots$ and $(i_1, j_1), (i_2, j_2), \cdots$ positions, then, k^{th} bit of DCV is

$$DCV_k = \begin{cases} 1, & k = \acute{i}_1, (\acute{i}_1 + 1), \cdots, (\acute{j}_1 - 1), \acute{i}_2, \cdots, (\acute{j}_2 - 1), \cdots, \\ & (i_1 + 1), (i_1 + 2), \cdots, j_1, (i_2 + 1), \cdots, j_2, \cdots, \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where DCV is termed as derived complement vector.

Thus, from Relation 4, any vector $v \in V_{q_m}$ of an attractor basin (V_{q_m}) must satisfy the relation

$$\min\{1, \sum_{k=i}^{j} \mid DCV_k - x_k \mid\} = q_m \quad (6)$$

Fig.2 is used to illustrate the effect of DCV on state space covered by a complemented FMACA.

Example 3.5. Fig.2(a) represents state transition diagram of a 3-cell 3-state FMACA configured with non-complemented rule vector $\mathcal{R} =< 238, 170, 0 >$. Here, PCP=FCP=1. As per Conjecture 2, the only possible complemented rule vector of $\mathcal{R}$ is $\acute{\mathcal{R}} =< 238, 85, 255 >$ where

- $\mathcal{R}_1 = \mathcal{R}_{PC} = 238$ remains same; and
- complemented rules in the DR are rule 85 and 255.

That is, $F =< 011 >$. As per Definition 3.3, $DCV =< 001 >$. Fig.2(b) represents the state transition behavior of a 3-cell FMACA with rule vector $\acute{\mathcal{R}} =< 238, 85, 255 >$. It has identical number of attractor basins, with each basin having number of states identical to that of non-complemented FMACA shown in Fig.2(a). In this case, any vector $v \in V_{q_m}$ must satisfy Relation 6. That is,

$$\min\{1, \sum_{k=1}^{2} x_k + (1 - x_3)\} = \begin{cases} 0.0, & \text{for } V_{0.0} \\ 0.5, & \text{for } V_{0.5} \\ 1.0, & \text{for } V_{1.0} \end{cases}$$

The characterization of complemented FMACA based on DCV establishes the fact that the state space of a non-complemented FMACA can be altered while keeping number of attractor basins and number of states in each basin identical. This can be achieved by designing complemented FMACA as per the above formulation.

Valid Derived Complement Vector

In case of a 3-neighborhood FMACA, if a DV contains successive 1's in between i^{th} to j^{th} cells, then the PCP may be anywhere in between i^{th} to j^{th} positions marked as FCP and LCP respectively. As per Conjecture 2 and Definition 3.3, the valid DCV depends on the PCP.

1. If PCP=FCP, then the DCV is given by

$$DCV_k \in \begin{cases} \{0, 1\} & \text{if } (\text{FCP} + 1) < k \leq \text{LCP} \\ \{0\} & \text{otherwise} \end{cases} \quad (7)$$

2. If PCP=LCP, then

$$DCV_k \in \begin{cases} \{0,1\} & \text{if FCP} \leq k < (\text{LCP}-1) \\ \{0\} & \text{otherwise} \end{cases} \tag{8}$$

3. If FCP $<$ PCP $<$ LCP, then

$$DCV_k \in \begin{cases} \{0,1\} & \text{if FCP} \leq k < (\text{PCP}-1) \text{ and } (\text{PCP}+1) < k \leq \text{LCP} \\ \{0\} & \text{otherwise} \end{cases} \tag{9}$$

Example 3.6. Examples of DCVs which can be generated by a 3-neighborhood complemented FMACA with $DV =< 0011111100 >$ are $< 0000101000 >$, $< 0000110100 >$, $< 0010010000 >$, and $< 0010000100 >$.

The following formulation provides the application of DV and DCV to identify basin of FMACA.

3.3. Identification of Attractor Basins

As per Section 3.1, for non-complemented FMACA, if DV is an n-bit dependency vector of an n-dimensional vector space and $\mathcal{P}$ is a pattern belonging to V_{q_m}, then

$$q_m = \text{state of PC} = \min\{1, \sum_{i=1}^{n} DV_i \cdot \mathcal{P}_i\} \tag{10}$$

As per Section 3.2, for complemented FMACA, if DV and DCV represent dependency vector and derived complement vector respectively, and $\mathcal{P}$ belonging to V_{q_m}, then

$$q_m = \text{state of PC} = \min\{1, \sum_{i=1}^{n} \mid DCV_i - DV_i \cdot \mathcal{P}_i \mid\} \tag{11}$$

In Section 2, when the FMACA (T and F) is loaded with $\mathcal{P}$, it travels through a number of states equal to depth d of the FMACA and ultimately reaches an attractor $\acute{\mathcal{P}}$, i.e.,

$$\mathcal{P}(1) =\mid F - T \cdot \mathcal{P}(0) \mid; \quad \mathcal{P}(2) =\mid F - T \cdot \mathcal{P}(1) \mid; \quad \acute{\mathcal{P}} = \mathcal{P}(d) =\mid F - T \cdot \mathcal{P}(d-1) \mid$$

where T is an $n \times n$ matrix, F is an n-dimensional CV, and $d \simeq n$. The state q_m of PC of attractor of the basin where $\mathcal{P}$ belongs is identified accordingly which represents the basin V_{q_m}. The complexity of this algorithm is O(n^3), n being the size of the patterns. Whereas in the proposed scheme, the state of PC of attractor of the basin where $\mathcal{P}$ belongs is given by

$$q_m = \min\{1, \sum_{i=1}^{n} \mid DCV_i - DV_i \cdot \mathcal{P}_i \mid\} \tag{12}$$

So, the complexity of this approach is O(n). Thus, to identify attractor basin V_{q_m} of a pattern $\mathcal{P}$, DV and DCV can be employed rather than T and F which reduce complexity from O(n^3) to O(n).

Next we present the design of a hybrid pattern classifier based on the principles of radial basis function (RBF) and FMACA to classify a given set of patterns into K classes.

4. RBFFCA: A Hybrid Pattern Classifier

The RBFFCA is a hybrid pattern classifier based on the theory of RBF and FCA. It consists of three layers with entirely different roles. The input layer is made up of source nodes (sensory units) that connect the system to its environment. The second layer, the only hidden layer, applies a nonlinear transformation from the input space to hidden space. The output layer is quasi-linear, supplying the response of the system to the activation pattern (signal) applied to the input layer.

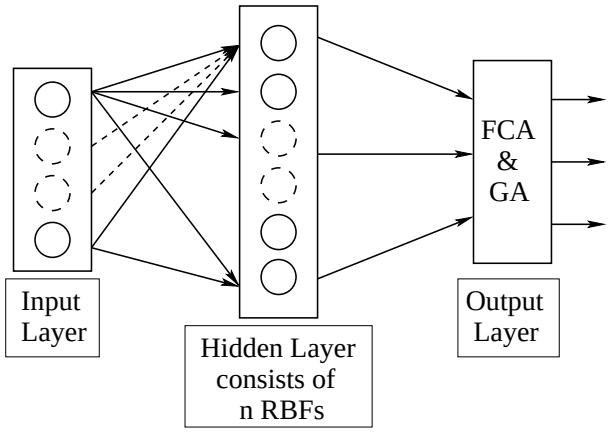

Figure 3. Architecture of RBFFCA based hybrid pattern classifier

Fig.3 represents the architecture of RBFFCA. It consists of three layers - input, hidden, and output layers denoted as x_i ($i = 1, 2, \cdots, n_0$), y_j ($j = 1, 2, \cdots, n$), and o_k ($k = 1, 2, \cdots, K$) respectively. The first layer is composed of input (source) nodes whose number is equal to the dimension n_0 of the input vector x. The second layer is a hidden layer, composed of nonlinear units that are connected directly to all of the nodes in the input layer. There is one hidden unit for each data point x^l, $l = 1, 2, \cdots, N$, where N is the size of training samples. The activation functions of the individual hidden units are defined by

$$G(x, t_j) = \exp(-\frac{||x - t_j||^2}{2\sigma_j^2}) \tag{13}$$

where $G(x, t_j)$ is the multivariate gaussian function, t_j and σ_j denote the center and width of the function, and $j = 1, 2, \cdots, n$ where $n < N$. The output layer consists of K units, K being the number of classes. This layer is fully connected to the hidden layer by FCA. The FCA provides an appropriate mappings of patterns of hidden layer into output layer. The desired FCA is evolved through GA.

4.1. Learning Phase of RBFFCA

The hybrid learning process consists of two stages:

1. Self-organized learning stage: The purpose of this stage is to estimate appropriate locations for the centers of the radial basis functions in the hidden layer.

2. Supervised learning stage: It completes the design by finding the desired FCA for the output layer.

4.1.1. Design of RBF

For the self-organized learning process, k-means clustering algorithm is used [15]. The k-means algorithm places the centers of the radial basis functions in only those regions of the input space where significant data are present. Let n denotes the number of radial basis functions. The determination of a suitable value for n may require extensive experimentation. Let $\{t_j(s)\}_{j=1}^n$ denote the centers of the radial basis functions at iteration s of the algorithm. Then, the k-means clustering algorithm proceeds as follows:

Algorithm 1. k-means clustering

1. Choose random values for initial centers $t_j(0)$ in such a way that these initial values be different.
2. Draw a sample vector x from the input space with a certain probability. The vector x is input into the algorithm at iteration s.
3. Let $j(x)$ denotes the index of the best-matching (winning) center for input vector x. Find $j(x)$ at iteration s by using the minimum-distance euclidean criterion:
$$j(x) = \arg\ \min_k ||x(s) - t_j(s)|| \tag{14}$$
where $t_j(s)$ is the center of the j^{th} radial basis function at iteration s and $j = 1, 2, \cdots, n$.
4. Adjust the centers of the radial basis functions, using the update rule:
$$t_j(s+1) = \begin{cases} t_j(s) + \eta[x(s) - t_j(s)] & j = j(x) \\ t_j(s) & \text{otherwise} \end{cases} \tag{15}$$
where η is a learning rate parameter that lies in the range $0 < \eta < 1$.
5. Increment s by 1, go back to Step 2, and continue the procedure until no noticeable changes are observed in the centers t_j.

Having identified the individual centers $\{t_j\}_{j=1}^n$ of the gaussian radial basis functions and their widths $\{\sigma_j\}_{j=1}^n$ using the k-means algorithm, the next and final stage of the hybrid learning process is to find out desired fuzzy cellular automata (FCA) for the output layer.

4.1.2. Design of FCA

A special class of FCA, termed as FMACA, is used to design the proposed pattern classifier. Suppose, we want to design a FMACA based classifier for a training set $S = \{S_1, \cdots, S_i, \cdots, S_K\}$ partitioned into K classes, where S_i represents the set of elements of class i. First, a FMACA with k-attractor basins is generated, where $k \geq K$. The elements of the training set S get distributed into k-attractor basins. Distribution of the training set S in k-attractor basins is then evaluated. Let, $\acute{S}$ be the set of elements in i^{th} ($i = 1, 2, \cdots, k$) attractor basin. Then label i^{th}-basin as j^{th} class, if the number of elements of j^{th} class in $\acute{S}$ is maximum. For ease of subsequent discussions, we introduce following terminologies.

- K denotes number of classes in the training set S.
- k denotes number of attractor basins of a FMACA in which the dataset S is to be distributed.
- N_{ij} represents the number of elements of class j covered by i^{th} attractor basin, where $i = 1, 2, \cdots, k$ and $j = 1, 2, \cdots, K$.
- M_i indicates the distribution of class elements in the i^{th} attractor basin.

The diversity of i^{th} attractor basin of an n-cell FMACA is given by

$$M_i = \frac{\max\{N_{ij}\}}{\sum_{j=1}^{K} N_{ij}} \tag{16}$$

The i^{th} ($i = 1, 2, \cdots, k$) attractor basin indicates the class j ($j = 1, 2, \cdots, K$) for which N_{ij} is maximum. The classification accuracy F of the FMACA is determined by the percentage of patterns which are correctly classified into different attractor basins. That is,

$$F = \frac{1}{k}\sum_{i=1}^{k} M_i = \frac{1}{k}\sum_{i=1}^{k} \frac{\max\{N_{ij}\}}{\sum_{j=1}^{K} N_{ij}} \tag{17}$$

To determine the overall best distribution, classification can be done by a simple application of the FMACA to the training set. The complexity lies in determining the best distribution for each basin. The desired FMACA is evolved through GA. Details of GA formulation are reported in Section 5.

4.2. Identification Phase

Let, x be an input pattern with dimension n_0 whose class is to be identified by RBFFCA based hybrid pattern classifier. The input pattern $x(x_1, x_2, \cdots, x_{n_0})$ is first transformed into n-dimensional pattern $y(y_1, y_2, \cdots, y_n)$ where

$$y_j = \exp(-\frac{||x - t_j||^2}{2\sigma_j^2}) \tag{18}$$

and t_j and σ_j represent the center and width of the j^{th} RBF respectively and $j = 1, 2, \cdots, n$. Next, y is loaded with the DV and DCV of FMACA which return q_m, the state of the attractor's pivot cell (PC) of an attractor basin where y belongs. Combining Equations 12 and 18, we obtain

$$q_m = \min\{1, \sum_{j=1}^{n} \mid DCV_j - DV_j \cdot \exp(-\frac{||x - t_j||^2}{2\sigma_j^2}) \mid\} \tag{19}$$

The value of q_m yields the address of the memory that stores the class information of input pattern x.

5. Synthesis of FMACA: A GA Formulation

The basic structure of GA revolves around the concept of evolving successive solutions according to their fitness. The fitness function for FMACA is elaborated next.

5.1. Fitness Function

The fitness F of a particular FMACA in a population is determined by the percentage of patterns which are correctly classified into different attractor basins. That is,

$$F = \frac{1}{k} \sum_{i=1}^{k} \frac{\max\{N_{ij}\}}{\sum_{j=1}^{K} N_{ij}} \tag{20}$$

where k and K represent the number of attractor basins and the number of classes in the training set; and N_{ij} depicts the number of patterns of class j covered by i^{th} attractor basin. Thus, F represents the capability of the evolved FMACA for classifying the given input pattern set into separate set of basins.

For the purpose of evolution, each solution (the solution is a FMACA) has to be encoded in bit string format (chromosome). Three major functions - random generation of initial population (IP), crossover and mutation, as developed in the current GA formulation, are next discussed.

5.2. Chromosome

The scheme to generate n-cell FMACA employs a chromosome consisting of:

1. a symbol string of numerical digits consists of m number of DVs each corresponding to an $n_i \times n_i$ T matrix, where $n_1 + n_2 + \cdots n_m = n$; and
2. a symbol string of numerical digits consists of m number of DCVs representing n-dimensional complement vector F.

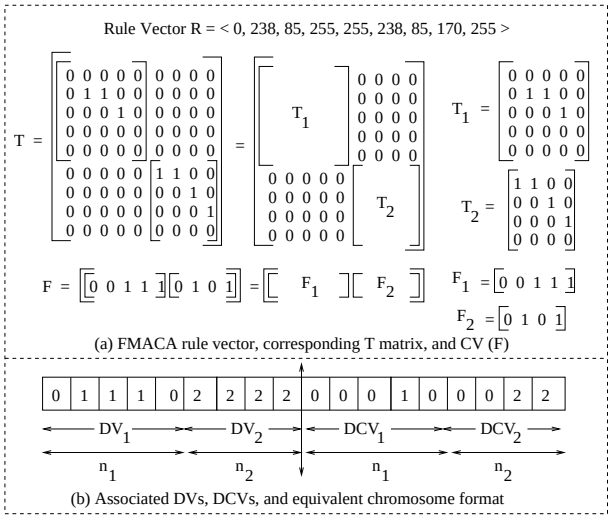

Figure 4. An example chromosome for GA formulation

So, the length of the chromosome is $2n$ where n is the number of cells in a FMACA. In essence, the m DVs/DCVs are concatenated together to form the final DV/DCV. The final DV/DCV represents the

multiple dependency of the variables if there exists. The DV/DCV of a FMACA with $\mathcal{K}^m$ attractor basins can be derived from the m number of DVs/DCVs of the FMACA with $\mathcal{K}$ attractor basins. A final DV/DCV has the constituent DVs/DCVs placed in non-overlapping positions. If two DVs are $DV_1 =< 01110 >$ and $DV_2 =< 1111 >$, then the corresponding final DV is given by $DV = [011102222]$. Similarly, if two DCVs are $DCV_1 =< 00010 >$ and $DCV_2 =< 0011 >$, then the corresponding final DCV is given by $DCV = [000100022]$. In general, the i^{th} dependency relation in a DV/DCV is represented with i ($i = 1, 2, \cdots, m$) inserted in place of corresponding dependent variables.

Fig.4 represents a 18-bit chromosome corresponding to the rule vector <0, 238, 85, 255, 255, 238, 85, 170, 255>. While Fig.4(a) represents the T matrix and the F; Fig.4(b) represents the equivalent chromosome format of FMACA (DV_is and DCV_is).

5.3. Random Generation of Initial Population

To form the initial population, it must be ensured that each solution randomly generated is an n-cell FMACA with $k = \mathcal{K}^m$ attractor basins where k and $\mathcal{K}$ represent the number of attractors of FMACA and the number of fuzzy states respectively. The chromosomes are randomly synthesized according to the following steps.

1. Randomly partition n into m number of integers such that $n_1 + n_2 + \cdots + n_m = n$.
2. For each n_i, randomly generate a valid DV (DV_i) and a valid DCV (DCV_i).
3. Synthesize n-bit DV through concatenation of m number of DV_is for first part.
4. Synthesize n-bit DCV through concatenation of m number of DCV_is for second part.
5. Synthesize a chromosome through concatenation of first and second part.

Fig.4(b) represents a randomly generated 18 bit chromosome that refers to the T matrix and the complement vector F shown in Fig.4(a). The 9×9 T matrix is obtained from two matrices (T_1 and T_2) of length 5 and 4 respectively by block diagonal form (BDF). The 9-bit F is produced through the concatenation of two CVs (F_1 and F_2) of length 5 and 4 respectively.

5.4. Crossover Algorithm

Fig.5 illustrates the crossover process employed in the GA evolution. Two chromosomes (FMACA_1 and FMACA_2) are shown in Fig.5. The single crossover point is selected randomly which is 8 in this case. The first 8 symbols of first part are taken from FMACA_1, while the rest 2 symbols are taken from FMACA_2 to form the new DV for FMACA_3. Similarly, in the second part, first 8 symbols of FMACA_1 and rest 2 symbols of FMACA_2 are merged together to form the n-dimensional DCV DCV_3 (Fig.5).

The resultant chromosome (FMACA_3), as explained below, violates the chromosome format in the 7th, 8th and 9th positions of first part (encircled in Fig.5). In Fig.5, the DV is [1110022322] where 2 and 3 are interleaved, which is invalid. The resultant valid chromosome after local recoding of symbols is shown in Fig.5 (FMACA_4).

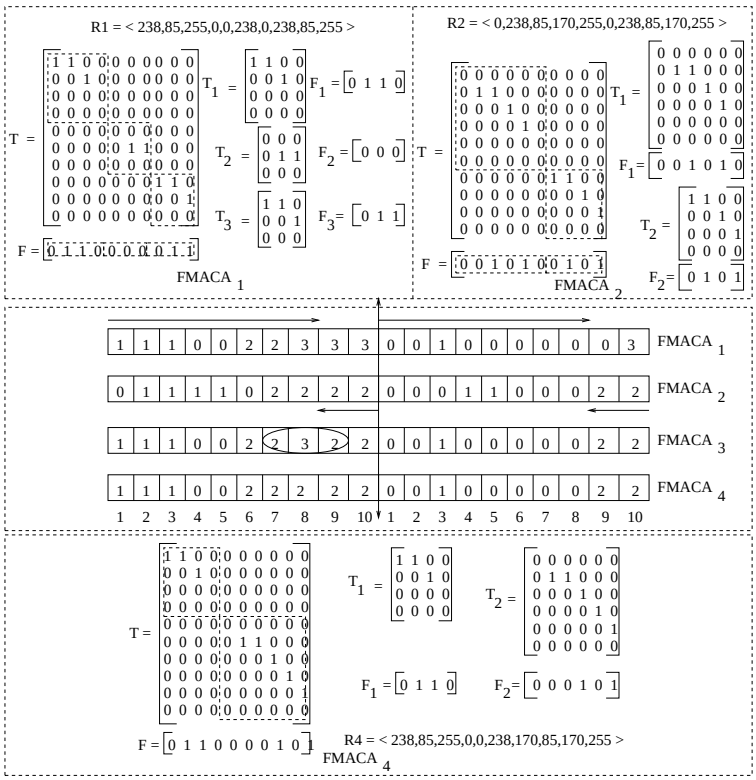

Figure 5. An illustration of crossover technique

5.5. Mutation Algorithm

As per the analysis reported in Section 3, if in a DV, the value of $\tilde{w}$ (number of 1's in a DV) is greater than 2, there exists a valid F and DCV. In mutation, we first randomly select a chromosome. If the i^{th} DV DV_i of the selected chromosome contains successive 1's from i_i (FC) to j_i (LC) cell positions and $\tilde{w}_i$ (of DV_i) ≥ 2, then we randomly select a mutation point x and flip the x^{th} position, where

$$(\text{FCP}+1) < x \leq \text{LCP} \text{ if PCP} = \text{FCP}; \ (\text{FCP}+1) \leq x < (\text{LCP}-1) \text{ if PCP} = \text{LCP};$$
$$(\text{FCP}+1) \leq x < (\text{PCP}-1) \text{ or } (\text{PCP}+1) < x \leq \text{LCP} \text{ if FCP} < \text{PCP} < \text{FCP}$$

So, in this case, the T matrix and the DV remain same, only F and DCV has been changed.

Fig.6 represents an example of mutation technique on a randomly selected chromosome (FMACA_1). A single mutation point (5th position) is randomly selected from the second part of the chromosome (FMACA_1). The FMACA_2 is the mutated version of FMACA_1.

5.6. Selection

Selection is done by the roulette wheel method. The probabilities are calculated on the basis of ranking of the individuals in terms of the fitness function, instead of the fitness function itself. Elitism is incorporated in the selection process to prevent oscillation of the fitness function with generation. The fitness of

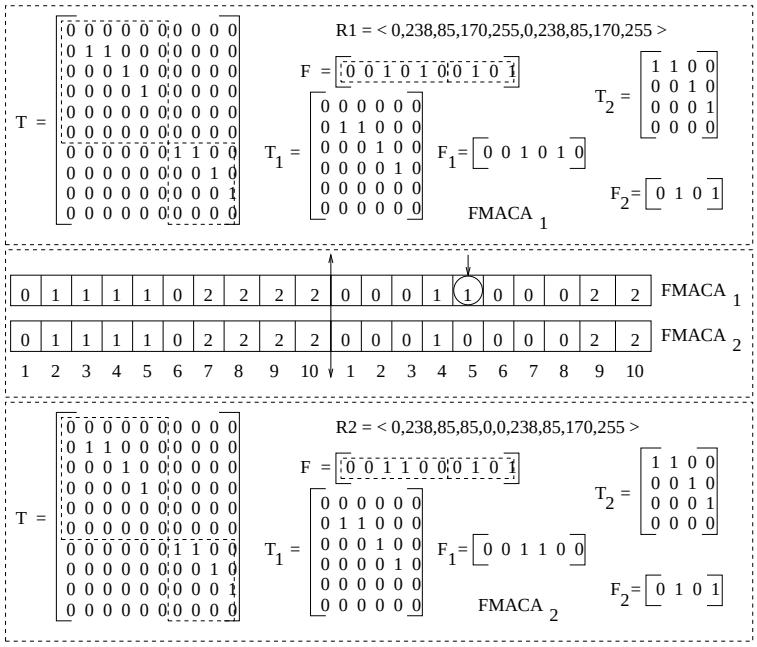

Figure 6. An illustration of mutation technique

the best individual of a new generation is compared with that of the current generation. If the later has a higher value the corresponding individual replaces a randomly selected individual in the new population.

The experimental results reported next confirm that the GA evolution provides the desired direction to arrive at the best FMACA to perform the pattern classification task.

6. Performance Analysis

This section presents the application of RBFFCA based pattern classifier in different applications. The major metric for evaluating classifier performance is classification accuracy. We also report retrieval time and memory overhead required to store RBFFCA based classifier as secondary metrics for performance evaluation. In all of our experiments, the GA parameters used are as follows:

Maximum generation :	100
Population size :	100
Chromosome length :	2n where n is the number of RBFs
Probability of crossover :	0.8
Probability of mutation :	0.001

The parameters are held constant across all runs. Unbiased initial population is generated randomly spreading over entire variable space in consideration. All the experiments are performed in SUN with Solaris 5.6, 350 MHz clock.

To analyze the performance of RBFFCA, the experimentation has been done in two parts.

1. In the first part, we have used the synthetic dataset proposed in [2].

2. In the second part, we perform the experiment on some of the STATLOG datasets [36] available from http://www.ics.uci.edu/∼mlearn and different DNA dataset proposed by Fickett et.al [16].

6.1. Synthetic Database

An often used benchmark in classification is STATLOG [36]. However, due to lack of a classification benchmark containing large datasets, we use the synthetic database proposed in [2] for our experiments. Each record in this synthetic database consists of nine attributes. Ten classification functions were also proposed in [2] to produce databases with distributions with varying complexities. In this subsection, we present the results of RBFFCA for these functions. All these functions divide the database into two classes. Attributes values are randomly generated.

Experimental Results

Table 1 presents the classification accuracy of RBFFCA for each of the ten functions proposed in [2]. It also shows a comparison with radial basis function neural network (RBFN). In all cases, 50 percent of the samples are used as training set and the remaining samples are used as test set. Ten such independent runs are performed and the mean value and standard deviation of the classification accuracy, computed over them, are presented in Table 1. The parameters used in this experiments are as follows:

Number of hidden nodes for RBFFCA :	50
Number of hidden nodes for RBFN :	50
Number of fuzzy states for RBFFCA :	101
The value of m for RBFFCA :	1

The results corresponding to memory overhead and retrieval time of RBFFCA and RBFN are as follows:

Memory overhead of RBFFCA :	24.02 KB
Memory overhead of RBFN :	32.19 KB
Retrieval time of RBFFCA for N = 10000 :	291 ms
Retrieval time of RBFN for N = 10000 :	327 ms
Retrieval time of RBFFCA for N = 100000 :	382 ms
Retrieval time of RBFN for N = 100000 :	617 ms

Table 1. Classification Accuracy of RBFFCA and RBFN on Synthetic dataset

Fn No	Size of Dataset	RBFFCA				RBFN			
		Training		Testing		Training		Testing	
		Mean	Sd. Dv.	Mean	Sd. Dv.	Mean	Sd. Dv.	Mean	Sd. Dv.
1	10000	83.2	0.25	81.7	0.09	84.1	0.33	79.6	0.12
	100000	83.0	0.12	80.2	0.04	83.8	0.13	78.2	0.30
2	10000	81.7	0.10	78.3	0.31	80.6	0.15	75.3	0.21
	100000	82.1	0.07	77.8	0.11	81.9	0.04	76.6	0.04
3	10000	80.9	0.12	78.5	0.20	82.5	0.04	72.1	0.17
	100000	82.3	0.01	80.1	0.01	83.1	0.16	79.8	0.02
4	10000	85.4	0.13	81.9	0.06	84.8	0.01	78.3	0.31
	100000	83.7	0.20	82.0	0.30	85.2	0.06	81.6	0.07
5	10000	81.6	0.31	80.2	0.11	80.4	0.31	76.7	0.12
	100000	81.1	0.11	78.9	0.02	79.3	0.11	78.2	0.03
6	10000	84.0	0.08	82.4	0.05	82.8	0.08	81.9	0.01
	100000	84.5	0.17	81.8	0.15	82.0	0.02	79.0	0.22
7	10000	83.9	0.04	81.6	0.17	81.6	0.12	81.1	0.10
	100000	83.6	0.02	80.3	0.19	82.4	0.03	78.5	0.02
8	10000	82.8	0.21	78.4	0.11	83.6	0.09	73.2	0.01
	100000	81.5	0.02	77.7	0.01	83.7	0.10	74.1	0.27
9	10000	84.1	0.03	83.8	0.00	79.5	0.20	79.7	0.14
	100000	85.4	0.09	83.2	0.16	86.1	0.11	75.3	0.02
10	10000	82.6	0.11	79.1	0.03	82.3	0.02	74.4	0.01
	100000	82.8	0.20	78.6	0.08	82.1	0.22	78.3	0.23

The time required to generate RBFFCA and RBFN for a given dataset are as follows:

Learning time of RBFFCA for N = 10000 :	1357 ms
Learning time of RBFN for N = 10000 :	872 ms
Learning time of RBFFCA for N = 100000 :	2689 ms
Learning time of RBFN for N = 100000 :	1207 ms

All the results reported above and in Table 1 clearly establish the fact that RBFFCA performs better than RBFN with lesser memory overhead and retrieval time irrespective of the functions and size of the dataset. Though the generation time of RBFFCA is greater than that of RBFN, this is not an important design consideration. Because usually the classifier is generated once and used over and over again. So, the higher classification accuracy with lesser retrieval time and memory overhead mainly establish the significance of RBFFCA. Next the application of RBFFCA in satellite image classification is presented.

6.2. Satellite Image Classification

The original Landsat data for this database is generated from data purchased from NASA by the Australian Centre for Remote Sensing, and used for research at the University of New SouthWales. The sample database is generated taking a small section (82 rows and 100 columns) from the original data [36]. The data are divided into train and test set with 4435 examples in train set and 2000 in test set.

6.2.1. Experimental Results

The results on dataset reported earlier are presented in Table 2 in respect of classification accuracy. Column I of Table 2 represents the number of hidden units (that is, the number of gaussian radial basis functions) while Columns II and III depict the number of fuzzy states and the value of m. Columns IV and V show the mean value and standard deviation of the classification accuracy of both training and test dataset respectively. The classification accuracy of training and testing confirm that the evolved RBFFCA based hybrid pattern classifier can generalize the satellite database irrespective of the number of hidden units, number of fuzzy states and the value of m.

Table 2. Classification Accuracy of RBFFCA on SatImage dataset

Hidden Units	Value of $\mathcal{K}$	Value of m	Training (%)		Testing (%)	
			Mean	Sd. Dv.	Mean	Sd. Dv.
25	11	1	78.4	0.11	74.9	0.06
		2	79.1	0.17	76.3	0.12
	101	1	83.2	0.16	80.1	0.02
		2	81.9	0.07	79.4	0.17
50	11	1	90.4	0.14	**88.1**	0.01
		2	88.6	0.23	87.0	0.07
	101	1	89.6	0.13	84.1	0.21
		2	89.9	0.08	85.8	0.13
75	11	1	90.1	0.21	85.1	0.01
		2	91.5	0.18	82.3	0.20
	101	1	89.4	0.31	81.9	0.16
		2	87.0	0.42	82.8	0.33

6.2.2. Comparison of Different Algorithms

Table 3 compares the performance of RBFFCA with that of conventional classification algorithms like RBFN [24, 29], MLP (multilayer perceptron) [24, 29], C4.5 [40], FCATree (fuzzy cellular automata tree) [30], CATree (cellular automata tree) [33] in terms of classification accuracy, memory overhead, and retrieval time.

Column II of Table 3 represents the classification accuracy while Columns III and IV depict the memory overhead and retrieval time of different algorithms. The experimental results of Table 3 clearly

Table 3. Comparison of Different Classification Algorithms

Different Algorithms	Classification Accuracy (%)	Memory Overhead (KB)	Retrieval Time (ms)
RBFFCA	88.1	9.16	484
RBFN	87.9	10.17	502
MLP	86.2	11.33	716
C4.5	85.2	709.72	2255
FCATree	81.9	189.62	2419
CATree	87.6	222.74	4943

establish the fact that the classification accuracy of RBFFCA is higher while its memory overhead and retrieval time are lesser compared to that of different classification algorithms.

Next subsection presents the application of RBFFCA for finding splice-junction in DNA sequences.

6.3. Identification of Splice-Junction in DNA Sequence

In bioinformatics, one of the major task is the recognition of certain DNA subsequences important in the expression of genes. Basically, a DNA sequence is a string over alphabet D=$\{$A,C,G,T$\}$. DNA contains the information by which a cell constructs protein molecules. The cellular expression of protein proceeds by the creation of a 'message' ribonucleic acid (mRNA) copy from the DNA template. This mRNA is then translated into a protein. One of the most unexpected findings in molecular biology is that large pieces of the mRNA are removed before it is translated further [4]. The utilized sequences are known as exons while the removed sequences are known as introns, or intervening sequences. The points at which DNA is removed are known as splice-junctions. The splice-junction problem is to determine into which of the following three categories a specified location in a DNA sequence falls: (1) exon/intron borders, referred to as donors (2) intron/exon borders, referred to as acceptors and (3) neither.

6.3.1. Description of Dataset

The dataset used in this problem is a processed version of the Irvine Primate splice-junction database [36]. Each of the 3186 examples in the database consists of a window of 60 nucleotides, each represented by one of four symbolic values ($\{$A,C,G,T$\}$) and the classification of the middle point in the window as one of intron-exon boundary, or neither of these. Processing involved the removal of a small number of examples(4), conversion of the original 60 symbolic attributes to 180 binary attributes and the conversion of symbolic class labels to numeric labels. The training set of 2000 is chosen randomly from the dataset and the remaining 1186 examples are used as the test set.

6.3.2. Experimental Results

The experimental results on DNA dataset reported earlier are presented in Table 4 and subsequent discussions analyze the results in respect of classification accuracy.

Column I of Table 4 represents the number of hidden units while Columns II and III depict the number of fuzzy states and the value of m. Columns IV and V show the mean value and standard deviation of the classification accuracy of both training and test dataset respectively. The classification accuracy of training and testing confirm that the evolved RBFFCA can generalize the DNA database irrespective of the number of hidden units, number of fuzzy states and the value of m.

Table 4. Classification Accuracy of RBFFCA on DNA dataset

Hidden Units	Value of $\mathcal{K}$	Value of m	Training (%)		Testing (%)	
			Mean	Sd. Dv.	Mean	Sd. Dv.
25	11	1	88.2	0.23	81.9	0.28
		2	89.3	0.18	79.4	0.31
	101	1	84.8	0.31	77.2	0.10
		2	86.4	0.02	87.6	0.27
50	11	1	93.8	0.05	88.1	0.23
		2	93.9	0.09	90.2	0.07
	101	1	96.1	0.17	**92.9**	0.03
		2	96.4	0.12	91.3	0.11
75	11	1	95.7	0.15	89.8	0.20
		2	95.2	0.32	88.1	0.30
	101	1	91.6	0.11	86.5	0.17
		2	92.7	0.22	89.2	0.34

6.3.3. Comparison of Different Algorithms

Table 5 compares the performance of RBFFCA with RBFN, MLP, C4.5, FCATree, CATree, etc. in terms of accuracy, memory overhead, and retrieval time.

Column II of Table 5 represents the classification accuracy while Columns III and IV depict the memory overhead and retrieval time of different algorithms. The experimental results of Table 5 clearly establish the fact that the classification accuracy of RBFFCA is comparable to different algorithms while its memory overhead and retrieval time are lesser compared to that of different algorithms.

6.4. Identification of Protein Coding Region in DNA

This subsection presents the application of RBFFCA for finding protein-coding (exon) regions in anonymous sequences of DNA. The idea of the new method is to use the framework of RBF and FCA. The new method is evaluated for few sequences and an analysis regarding the accuracy is also presented.

6.4.1. Description of Dataset

The data used for this study are the human DNA data collected by Fickett and Tung [16]. The benchmark human data includes three different datasets. For the first dataset, non-overlapping human DNA

Table 5. Comparison of Different Classification Algorithms

Different Algorithms	Classification Accuracy (%)	Memory Overhead (KB)	Retrieval Time (ms)
RBFFCA	92.9	28.71	298
RBFN	93.9	34.19	876
MLP	91.4	37.22	1184
C4.5	93.3	1067.96	655
FCATree	88.1	51.22	491
CATree	86.4	50.86	494

sequences of length 54 have been extracted from all human sequences, with shorter pieces at the ends discarded. Every sequence is labeled according to whether it is entirely coding, entirely non-coding, or mixed, and the mixed sequences (i.e., overlapping the exon-intron boundaries) are discarded. The dataset also includes the reverse complement of every sequence. This means that one-half of the data is guaranteed to be from the non-sense strand of the DNA, which makes the problem of identifying coding regions somewhat harder. For the current study, we have used the same division into training and test data as in the benchmark study [16]. The training set is used exclusively to construct the RBFFCA based pattern classifier, and the classifier is then used to classify the test set. In addition to the 54-base dataset, we have used datasets containing 108 and 162 bases. No information about reading frames is used in this study. We have tried to solve the problem of finding coding regions in DNA about which nothing is known. Every window is either all-coding or all-non-coding, but the reading frame of each window is unknown. This choice of window lengths and experimental method follows that used by Fickett and Tung [16], and the problem here is what they defined as protein coding region.

6.4.2. Experimental Results

The experimental results on Fickett and Tung dataset are presented in Table 6. Column I of Table 6 represents the number of hidden units while Columns II and III depict the number of fuzzy states and the value of m. Columns IV-VI show the classification accuracy of both training and test dataset on Human DNA 54bp, 108bp, and 162bp respectively. The classification accuracy of training and testing confirm that the RBFFCA can generalize the Fickett and Tung dataset irrespective of DNA sequence length, number of hidden units, number of fuzzy states and value of m.

6.4.3. Comparison of Different Algorithms

Table 7 compares the classification accuracy of RBFFCA with that of conventional algorithms namely OC1, Position Asymmetry, Fourier, Hexamer, Dicodon Usage, etc. Columns II-IV of Table 7 represent the classification accuracy of different algorithms on Human DNA 54bp, 108bp, and 162bp respectively. The experimental results of Table 7 clearly establish the fact that the classification accuracy of RBFFCA is greater than that of different algorithms irrespective of its sequence length.

Table 6. Classification Accuracy of RBFFCA on Fickett and Tung dataset

Hidden Units	Value of $\mathcal{K}$	Value of m	Human 54bp		Human 108bp		Human 162bp	
			Training	Testing	Training	Testing	Training	Testing
50	11	1	78.4	74.8	79.2	75.8	80.6	74.9
		2	77.0	74.1	80.6	77.4	78.3	73.3
	101	1	81.6	77.5	81.5	76.3	83.4	78.6
		2	82.1	76.3	83.3	78.0	85.7	81.8
75	11	1	84.8	79.0	83.9	79.2	86.2	83.2
		2	83.7	78.7	85.1	74.8	86.8	84.1
	101	1	86.1	**83.2**	87.9	**84.1**	88.2	**84.6**
		2	87.8	81.5	88.5	83.6	89.1	84.3
100	11	1	85.3	82.6	83.2	82.9	85.5	82.8
		2	83.6	79.3	80.8	77.2	84.8	80.9
	101	1	81.2	76.4	84.2	80.7	86.9	83.4
		2	84.3	79.9	80.6	73.5	85.8	80.5

Table 7. Classification Accuracy of Different Algorithms

Algorithms	54bp	108bp	162bp
RBFFCA	83.2	84.1	84.6
OC1	73.9	83.7	84.2
Position Asymmetry	70.7	77.6	81.7
Fourier	69.5	77.4	82.0
Hexamer	69.8	71.4	73.8
Dicodon Usage	69.8	71.2	73.7

7. Conclusion

In this paper, we propose a hybrid learning algorithm based on the principles of radial basis function (RBF) and fuzzy cellular automata (FCA) for pattern classification of real valued data. Two new operators are introduced to characterize a special class of FCA, termed as fuzzy multiple attractor CA (FMACA). Application of such operators brings down the complexity to identify the class of an input pattern from O(n^3) to O(n). An efficient formulation of genetic algorithm (GA) has been proposed for evolution of desired FMACA to perform pattern classification task. The genetic operators are implemented in such a way that they help to preserve the structure of FMACA.

The aforesaid model is used in different pattern classification problems and the results are compared with some of the related techniques on the basis of some quantitative performance indices. Extensive experimental results show that the proposed hybrid learning algorithm has high classification accuracy with less memory overhead and computation time. The investigation, besides having significance in cellular automata (CA) research, has potential in soft computing research and for application to large scale problems involving data mining, bioinformatics, etc.

Acknowledgment

The authors would like to thank anonymous referees for providing helpful comments and valuable criticisms on the original version of the manuscript which have greatly improved the presentation of paper.

References

[1] Adamatzky, A.: Hierarchy of Fuzzy Cellular Automata, *Fuzzy Sets and Systems*, **62**, 1994, 167–174.

[2] Agrawal, R., Ghosh, S., Imielinski, T., Iyer, B., Swami, A.: An Interval Classifier for Database Mining Applications, *VLDB92, Vancouver, British Columbia, Canada*, 1992, 560–573.

[3] Austin, J., Kennedy, J., Buckle, S., Moulds, A., Pack, R.: The Cellular Neural Network Associative Processor, C-NNAP, *IEEE Monograph on Associative Computers*, 1997.

[4] Breathnach, R. J., Mandel, J. L., Chambon, P.: Ovalbumin gene is split in chicken DNA, *Nature*, **270**, 1977, 314–319.

[5] Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J.: Classification and Regression Trees, *Wadsworth, Belmont*, 1984.

[6] Cattaneo, G., Flocchini, P., Mauri, G., Santoro, N.: Cellular Automata in Fuzzy Backgrounds, *Physica D*, **105**, 1997, 105–120.

[7] Chady, M., Poli, R.: Evolution of Cellular-automaton-based Associative Memories, *Technical Report no. CSRP-97-15*, May 1997.

[8] Chattopadhyay, S., Adhikari, S., Sengupta, S., Pal, M.: Highly Regular, Modular, and Cascadable Design of Cellular Automata-Based Pattern Classifier, *IEEE Transaction on VLSI Systems*, **8**(6), December 2000, 724–735.

[9] Chaudhuri, P. P., Chowdhury, D. R., Nandi, S., Chatterjee, S.: Additive Cellular Automata, Theory and Applications, VOL. 1, *IEEE Computer Society Press, Los Alamitos, California*, **ISBN-0-8186-7717-1**, 1997.

[10] Cheeseman, P., Stutz, J.: Bayesian Classification (AutoClass): Theory and Results, *In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smith and R. Uthurusamy, editors, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press*, 1996, 153–180.

[11] Chopard, B., Droz, M.: *Cellular Automata Modelling of Physical Systems*, Cambridge University Press, 1998.

[12] Chua, L. O., Yang, L.: Cellular Neural Networks : Application, *IEEE Transaction on Circuits and Systems*, **35**(10), 1988, 1273–1290.

[13] Chua, L. O., Yang, L.: Cellular Neural Networks : Theory, *IEEE Transaction on Circuits and Systems*, **35**(10), 1988, 1257–1272.

[14] Dougherty, J., Kohavi, R., Sahami, M.: Supervised and Unsupervised Discretization of Continuous Features, *Proceedings of the 12th International Conference on Machine Learning, San Francisco, CA: Morgan Kaufmann*, 1995, 194–202.

[15] Duda, R. O., Hart, P. E., Stork, D. G.: *Pattern Classification*, Wiley-Interscience, 2000, ISBN 0471056693.

[16] Fickett, J., Tung, C. S.: Assessment of protein coding measures, *Nucleic Acids Res.*, **20**, 1992, 6441–6450.

[17] Fisher, D.: Improving Inference Through Conceptual Clustering, *In Proceedings 1987 AAAI Conference, Seattle, Washington*, 1987.

[18] Flocchini, P., Geurts, F., Mingarelli, A., Santoro, N.: Convergence and Aperiodicity in Fuzzy Cellular Automata: Revisiting Rule 90, *Physica D*, **142**, 2000, 20–28.

[19] Fotheringham, S., P. Rogerson, E.: Spatial Analysis and GIS, *Taylor and Francis*, 1994.

[20] Ganguly, N., Maji, P., Dhar, S., Sikdar, B. K., Chaudhuri, P. P.: Evolving Cellular Automata as Pattern Classifier, *Proceedings of Fifth International Conference on Cellular Automata for Research and Industry, ACRI 2002, Switzerland, Lecture Notes in Computer Science, Springer*, **2493**, October 2002, 56–68.

[21] Ganguly, N., Maji, P., Sikdar, B. K., Chaudhuri, P. P.: Design and Characterization of Cellular Automata Based Associative Memory for Pattern Recognition, *IEEE Transaction on System, Man and Cybernetics, Part B*, **34**(1), February 2004, 672–678.

[22] Goldberg, D. E.: Genetic Algorithms in Search, Optimizations and Machine Learning, *Morgan Kaufmann*, 1989.

[23] Gordon, D. M.: The Development of Organization in An Ant Colony, *American Scientist*, **83**, Jan-Feb 1995, 50–57.

[24] Hertz, J., Krogh, A., Palmer, R. G.: Introduction to the Theory of Neural Computation, *Santa Fe Institute Studies in the Sciences of Complexity, Addison Wesley*, 1991.

[25] Jen, E.: Invariant Strings and Pattern Recognizing properties of 1D CA, *Journal of Statistical Physics*, **43**, 1986.

[26] Kaufmann, L., Rousseeuw, P. J.: Finding Groups in Data: An Introduction to Cluster Analysis, *John Wiley and Sons*, 1990.

[27] Kohavi, R., Sahami, M.: Error-based and Entropy-based Discretization of Continuous Features, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR: AAAI Press*, 1996, 114–119.

[28] Langton, C. G.: Self-reproduction in cellular automata, *Physica D*, **10**, 1984, 135–144.

[29] Lippmann, R.: An Introduction to Computing with Neural Nets, *IEEE ASSP Magazine*, **4(22)**, April 1987.

[30] Maji, P., Chaudhuri, P. P.: Fuzzy Cellular Automata for Modeling Pattern Classifier, *IEICE Transactions on Information and Systems*, **E88-D**(4), April 2005, 691–702.

[31] Maji, P., Ganguly, N., Chaudhuri, P. P.: Error Correcting Capability of Cellular Automata Based Associative Memory, *IEEE Transaction on System, Man and Cybernetics, Part A*, **33**(4), July 2003, 466–480.

[32] Maji, P., Shaw, C., Ganguly, N., Sikdar, B. K., Chaudhuri, P. P.: Theory and Application of Cellular Automata for Pattern Classification, *Fundamenta Informaticae*, **58**(3-4), December 2003, 321–354.

[33] Maji, P., Sikdar, B. K., Chaudhuri, P. P.: Cellular Automata Evolution for Pattern Classification, *Proceedings of 6th International Conference on Cellular Automata for Research and Industry, ACRI 2004, Amsterdam, The Netherlands, Lecture Notes in Computer Science, Springer*, **3305**, October 2004, 660–669.

[34] Mehta, M., Agrawal, R., Rissanen, J.: SLIQ: A Fast Scalable Classifier for Data Mining, *Proceedings of International Conference on Extending Database Technology, Avignon, France*, 1996.

[35] Michalski, R. S., Carbonnel, J. M., Mitchell, T. M.: Machine Learning: An Artificial Intelligence Approach, *Morgan Kaufmann, Los Atlos, CA*, 1983.

[36] Michie, D., Spiegelhalter, D. J., Taylor, C. C.: Machine Learning, Neural and Statistical Classification, *Ellis Horwood*, 1994.

[37] Mitchell, T. M.: Generalization as Search, *In Artificial Intelligence*, **18**, 1982, 203–226.

[38] Morita, K., Ueno, S.: Parallel Generation and Parsing of Array Languages Using Reversible Cellular Automata, *International Journal of Pattern Recognition and Artificial Intelligence*, **8**, 1994, 543–561.

[39] Neumann, J. V.: The Theory of Self-Reproducing Automata, *A. W. Burks, Ed. University of Illinois Press, Urbana and London*, 1966.

[40] Quinlan, J. R.: C4.5: Programs for Machine Learning, *Morgan Kaufmann, CA*, 1993.

[41] Raghavan, R.: Cellular Automata In Pattern Recognition, *Information Science*, **70**, 1993, 145–177.

[42] Saha, S., Maji, P., Ganguly, N., Sikdar, B. K., Chaudhuri, P. P.: Evolution of Cellular Automata Based Pattern classifier and recognizer, *IEEE Conference on Systems, Man and Cyberbnatics*, 1, October 2002.

[43] Shafer, J., Agrawal, R., Mehta, M.: SPRINT: A Scalable Parallel Classifier for Data Mining, *In 22nd VLDB Conference*, 1996.

[44] Shaw, G., Wheeler, D.: Statistical Techniques in Geographical Analysis, *London, David Fulton*, 1994.

[45] Sikdar, B. K., Ganguly, N., Majumder, P., Chaudhuri, P. P.: Design of Multiple Attractor $GF(2^p)$ Cellular Automata for Diagnosis of VLSI Circuits, *Proceedings of* 14^{th} *International Conference on VLSI Design, India*, January 2001, 454–459.

[46] Stonebraker, M.: Readings in Database Systems, *2ed, Morgan Kaufmann*, 1993.

[47] Toffoli, T.: Reversible computing, *In J. W. De Bakker and J. Van Leeuwen, editors, Automata, Languages and Programming*, 1980, 632–644.

[48] Toffoli, T., Margolus, N.: Cellular Automata Machines, *The MIT Press, Cambridge, Massachusetts*, 1987.

[49] Tzionas, P., Tsalides, P., Thanailakis, A.: Design and VLSI implementation of a Pattern Classifier using pseudo 2D Cellular Automata, *IEE Proc. G*, **139**(6), December 1992, 661–668.

[50] Tzionas, P., Tsalides, P., Thanailakis, A.: A new Cellular Automaton-based nearest neighbor pattern classifier and its VLSI implementation, *IEEE Transaction on VLSI Implementation*, **2**(3), 1994, 343–353.

[51] Tzionas, P., Tsalides, P., Thanailakis, A.: A Cellular Neural Network Predicting the behaviour of a Complex System modelled as a Cellular Automaton, *Fifteenth Annual International Symposium on Forecasting ISF95*, Toronto, CANADA, June 4-7 1995.

[52] Vichniac, G.: Simulating physics with cellular automata, *Physica D*, **10**, 1984, 96–115.

[53] Wolfram, S.: *A New Kind of Science*, Wolfram Media, Inc, 2002.