local systems which each consists of a convex combination of linear systems, or Takagi–Sugeno (TS) fuzzy subsystems, respectively. For given local energy functions a combination of local subsystems is to find so that a synchronization of the dynamical behavior of the local systems is provided. The simulation results show a Pareto-optimal solution for every local system can be reached. One condition for existence of such an optimum is that in every local system a stable combination of subsystems and a common range of parameters exist. The approach has been applied both for unrestricted communication between agents (net structure, global price) and a restricted communication (ring structure, local prices). Simulation results show that this approach is successful for combinations of linear subsystems and nonlinear TS fuzzy subsystems, respectively. Future work will be directed to decentralized control design of TS fuzzy systems in the framework of market-based optimization and its enhancement with learning strategies.

REFERENCES

[1] E. Altman, T. Basar, and R. Srikant, "A team-theoretic approach to congestion control," in *Proc. 14th IFAC World Congr.*, Beijing, China, 1999.

[2] J. Ayel, "Supervising conflicts in product management," *Int. J. Comp. Integr. Manufact.*, vol. 8, no. 1, pp. 54–63, 1995.

[3] G. Barrett and S. Lafortune, "Some issues concerning decentralized supervisory control with communication," in *Proc. 38th CDC*, Phoenix, AZ, USA, 1999, pp. 2230–2236.

[4] S. H. Clearwater, *Market-Based Control: A Paradigm for Distributed Resource Allocation*, S. H. Clearwater, Ed, Singapore: World Scientific, 1996.

[5] G. Fleury, J.-Y. Goujoun, M. Gourgand, and P. Lacomme, "Multi-agents approach for manufacturing systems optimization," in *Proc. PAAM 96*, London, U.K., 1996, pp. 225–244.

[6] C. Gerber, C. Russ, and G. Vierke, "On the suitability of market-based mechanisms for telematics applications," in *Proc. 3rd Intern. Conf. Auton. Agents*, Seattle, WA, May 1–5, 1999, p. 409.

[7] T. B. Gold, J. K. Archibald, and R. L. Frost, "A utility approach to multiagent coordination," in *Proc. 2000 IEEE Int. Conf. Robot. Automat.*, San Francisco, CA, 2000, pp. 2052–2057.

[8] O. Guenther, T. Hogg, and B. A. Huberman, "Controls for unstable structures," in *Proc. SPIE*, San Diego, CA, 1997, pp. 754–763.

[9] E. Haddad, "Real-time optimization of distributed load balancing," in *Proc. 2nd Workshop Parallel Distrib. Real-Time Syst. Cancun*, Mexico, Apr. 28–29, 1994, pp. 52–57.

[10] E. W. Large, H. I. Christensen, and R. Bajcsy, "Scaling the dynamic approach to path planning and control: Competition among behavioral constraints," *Int. J. Robot. Res.*, vol. 18, no. 1, pp. 37–58, 1999.

[11] J. Lei and Ü. Özgüner, "Combined decentralized multidestination dynamic routing and real-time traffic light control for congested traffic networks," in *Proc. 28th CDC*, Phoenix, AZ, 1999, pp. 3277–3282.

[12] K. Y. Lian, Ch. S. Chiu, and P. Liu, "Semi-decentralized adaptive fuzzy control for cooperative multirobot systems with H^∞ motion/internal force tracking performance," *IEEE Trans. Syst., Man Cybern. B*, vol. 32, pp. 269–280, June 2002.

[13] R. Murray-Smith and T. A. Johansen, *Multiple Model Approaches to Modeling and Control*. London, U.K.: Taylor and Francis, 1997.

[14] R. Palm, D. Driankov, and H. Hellendoorn, *Model Based Fuzzy Control*. Berlin: Springer-Verlag, 1997.

[15] S. E. Rhoads, *The Economist's View of the World: Government, Markets and Public Policy*. Cambridge, U.K.: Cambridge University Press, 1985, p. 63.

[16] T. Teredesai and V. C. Ramesh, "A multi-agent initiative system for real-time scheduling," in *Proc. SMC'98 Conf.*, San Diego, CA, 1998, pp. 439–444.

[17] H. Voos and L. Litz, "A new approach for optimal control using market-based algorithms," in *Proc. Eur. Contr. Conf. ECC'99*, Karlsruhe, Germany, 1999.

[18] A. Wallace, "Flow control of mobile robots using agents," in *Proc. 29th Int. Symp. Robot.*, Birmingham, U.K., 1998, pp. 273–276.

[19] M. B. Zaremba, Z. A. Banaszak, P. Majdzik, and K. J. Jedrzejek, "Distributed flow control design for repetitive manufacturing processes," in *Proc. 14th IFAC World Congr.*, Beijing, China, 1999.

Design and Characterization of Cellular Automata Based Associative Memory for Pattern Recognition

Niloy Ganguly, Pradipta Maji, Biplab K. Sikdar, and P. Pal Chaudhuri

***Abstract*—This paper reports a cellular automata (CA) based model of associative memory. The model has been evolved around a special class of CA referred to as generalized multiple attractor cellular automata (GMACA). The GMACA based associative memory is designed to address the problem of pattern recognition. Its storage capacity is found to be better than that of Hopfield network. The GMACA are configured with nonlinear CA rules that are evolved through genetic algorithm (GA). Successive generations of GA select the rules at the edge of chaos [1], [2]. The study confirms the potential of GMACA to perform complex computations like pattern recognition at the edge of chaos.**

***Index Terms*—Associative memory, CA, GA, GMACA.**

I. INTRODUCTION

This paper reports a cellular automata (CA) based model of associative memory designed to recognize patterns. Characterization of the model in respect of its pattern recognition capability along with other associated parameters has been reported from extensive study of the model. The storage capacity of the model has been found to be higher than $0.2n$ for a pattern size of n bits.

Pattern recognition is the study as to how the machines can learn to distinguish patterns of interest from their background. The *Associative Memory* model provides an elegant solution to the problem of identifying the closest match to the patterns learnt/stored [3]. The model, as shown in Fig. 1, divides the entire state space into some pivotal points (say) *a, b, c*. The pivots (patterns) are assumed to be learnt by the machine during its training phase. The states close to a pivot are the noisy vectors (patterns) associated with that specific pivotal point. The process of recognition of a pattern with or without noise, amounts to traversing the transient path (Fig. 1) from the given pattern to the closest pivotal point learnt. As a result, the time to recognize a pattern is independent of the number of patterns stored.

Since early 1980's the model of associative memory has attracted considerable interest among the researchers [4], [5]. Both sparsely connected machine (Cellular Automata) and densely connected network (Neural Net) have been explored to design the associative memory model for pattern recognition [4], [6], [7]. The Hopfield's neural net [7]–[9] models a "general content addressable memory," where the state space is categorized into a number of locally stable points referred to as attractors (Fig. 1). An input to the network initiates flow to a particular stable point (pivot). However, the complex structure of neural net with nonlocal interconnections has partially restricted its application for design of high speed low cost pattern recognition machine.

The associative memory model around the simple structure of cellular automata has been explored by a number of researchers [6], [10], [11]. Most of the CA based designs concentrated around uniform *CA* [6], [10], [11] with same rule applied to each of the *CA* cells. This structure has restricted the *CA* based model to evolve as a general purpose

Manuscript received May 13, 2002; revised October 11, 2002. This paper was recommended by Associate Editor V. Govindaraju.

N. Ganguly is with the Computer Centre, IISWBM, Calcutta, West Bengal 700073, India (e-mail: n_ganguly@hotmail.com).

P. Maji, B. K. Sikdar, and P. P. Chaudhuri are with the Department of Computer Science and Technology, Bengal Engineering College, Calcutta, West Bengal 711103, India (e-mail: pradipta@cs.becs.ac.in; biplab@cs.becs.ac.in; ppc@cs.becs.ac.in).

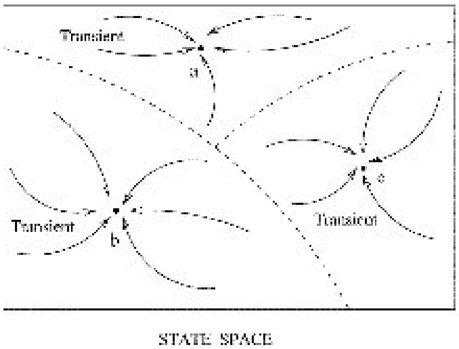


Fig. 1. Model of associative memory with three pivotal points.

pattern recognizer [6]. Further, estimation of memory capacity of the *CA* in relation to its architectural parameters such as number of cells (n), rules of the *CA* cells, etc has not been explored [12].

In the above background, this paper proposes an efficient *CA* model of associative memory for pattern recognition designed with nonlinear rules for *CA* cells. This class of hybrid *CA* is referred to as *generalized multiple attractor CA (GMACA)* [13]. It is the generalization of multiple attractor cellular automata (MACA) that employs only *additive* rules with *XOR/XNOR* logic [14].

We make use of genetic algorithm (GA) to arrive at the desired *GMACA* configurations. The nonlinear rule space of *GMACA* has been investigated. Diverse parameters λ, Z, entropy, G-density etc., proposed by the researchers [1], [15]–[17] to study *CA* behavior, are computed to characterize the *GMACA* evolved for pattern recognition. The results derived from the study confirm the following facts:

i) the *GMACA* lies in between order and chaos defined as the *edge of chaos* [1];
ii) the complex computation like pattern recognition occurs only at the *edge of chaos*;
iii) the memorizing capacity of *GMACA* is more than 20% of its lattice size and is better than Hopfield network.

The design of *GMACA* based associative memory and its application for pattern recognition are outlined in Section IV preceded by the design specification noted in Section III. The characterization of the model in respect of different parameters is reported in Section V. A brief introduction to *Cellular Automata* follows.

II. Cellular Automata

Cellular automata (CA) are the simple model of spatially extended decentralized systems made up of a number of cells [18]. Each cell of a *CA* is in a specific state which changes over time depending on the states of its neighbors. In this paper, we will concentrate on three-neighborhood (left, self, and right) one dimensional *CA*, each *CA* cell having two states—0 or 1. In a two state three-neighborhood *CA*, there can be a total of 2^{2^3}, i.e, 256 distinct next state functions referred to as the *rule* of *CA* cell [19]. If the same rule is applied to all the cells, then the *CA* is a *uniform CA*, else it is a *hybrid CA*. The rule tables for two such rules, 90 and 150, are illustrated as

Neighborhood:	111	110	101	100	011	010	001	000	*Rule*
NextState:	0	1	0	1	1	0	1	0	90
NextState:	1	0	0	1	0	1	1	0	150.

The first row lists the possible combinations of present states of the neighbors (left, self and right) of the ith cell at time t referred to as $q_i(t)$. The next two rows list the next states of ith cell at $(t+1)$ and denoted as $q_i(t+1)$. Decimal equivalent of the 8 next state values (90/150) is known as the rule of *CA* cell i. A nonlinear *CA* employs all possible 256 rules, while additive *CA* [14] employs only additive rules

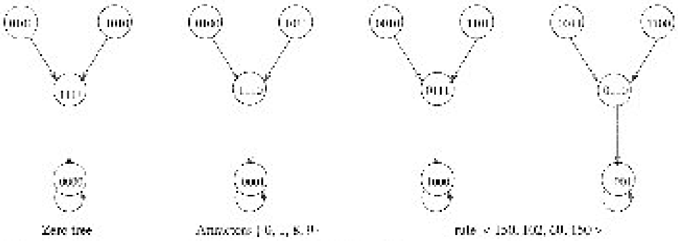


Fig. 2. State space of a four-cell MACA with attractors—0,1,8,9.

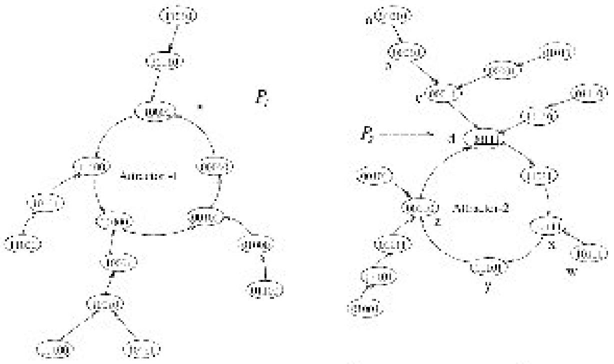


Fig. 3. State space of a five cell GMACA $\langle 89, 39, 87, 115, 91\rangle$.

with *XOR/XNOR* logic. The example additive *CA* of Fig. 2 with rule vector $\langle 150, 102, 60, 150\rangle$ is a *multiple attractor CA (MACA)*.

The entire state space of an *MACA* are divided into disjoint trees rooted at some attractor cycles. The length of an attractor cycle of the example *MACA* of Fig. 2 is 1. An inverted tree is also called attractor basin. States other than the cyclic state of an attractor are referred to as transient states. A transient state when loaded as a seed for the *CA* reaches the attractor cycle after some time steps. The maximum number of time steps needed for any state to reach the attractor cycle is called the depth or transient length of *MACA*.

This research work extends the concept of *MACA* to *Generalized MACA (GMACA)* with the following characteristics.

1) It employs nonlinear rules.
2) Its attractor cycle length may be ≥ 1.
3) Its tree structure, unlike *MACA*, is not uniform.

State space of an example *GMACA* is shown in Fig. 3.

Based on different dynamical behavior, Wolfram [19] reported a specific class (class IV) of *CA* displaying complex patterns of localized structure (attractor) with long transients. Wolfram predicted that class IV *CA* are capable of doing nontrivial computations.

The term *edge of chaos* is the critical point of a system, where a small change can push the system into chaotic state or lock the system into a fixed behavior. The logical and complex computations are likely to occur at edge of chaos [1]. Packard highlighted that the state transition behavior of class IV *CA* exhibits the property of a system at *edge of chaos* [15]. Further, Langton [1] defined the range of a parameter (λ) to identify the *CA* rules that perform complex computations.

The above observations motivate us to explore *GMACA* based associative memory model for pattern recognition. Design guidelines next follows.

III. Design Specification of GMACA Model for Pattern Recognition

A pattern recognizer is trained to get familiarized with specific pattern set $\{\mathcal{P}_1,\ldots,\mathcal{P}_i,\ldots \mathcal{P}_k\}$ so that it can recognize patterns with or without noise. If a new pattern $\acute{\mathcal{P}}_r$ is input to the system, the pattern recognizer identifies it as $\mathcal{P}_i$, where $\acute{\mathcal{P}}_i$ is the closest match to $\mathcal{P}_i$. The hamming distance between $\acute{\mathcal{P}}_i$ and $\mathcal{P}_i$ ($HD(\acute{\mathcal{P}}_i, \mathcal{P}_i)$) is the measure of noise.

If a *GMACA* has to function as a pattern recognizer, it has to learn/store the given pattern set $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_i, \ldots \mathcal{P}_k\}$. While the *GMACA* is run for some time steps with $\acute{\mathcal{P}}_i$ as an input, it returns $\mathcal{P}_i$. Hence, $\acute{\mathcal{P}}_i$ is a transient state close to $\mathcal{P}_i$. Therefore, the design guidelines for *GMACA* can be specified as follows.

R1: Each attractor cycle of the *GMACA* should contain only one pattern ($\mathcal{P}_i \in \mathcal{P}$) to be learnt.

R2: The Hamming Distance ($HD(\acute{\mathcal{P}}_i, \mathcal{P}_i)$) between each state $\acute{\mathcal{P}}_i \subset \mathcal{P}_i$-basin and $\mathcal{P}_i$ is lesser than the $HD(\acute{\mathcal{P}}_i, \mathcal{P}_j)$, where $\mathcal{P}_j \subset \mathcal{P}, \forall j = 1, 2, \ldots k. \& j \neq i$.

A *CA* which satisfies both **R1** & **R2** is the desired *GMACA* for pattern recognition. The 5-cell *GMACA* of Fig. 3 can learn two patterns, $\mathcal{P}_1 = 10000$ and $\mathcal{P}_2 = 00111$. It maintains both R_1 and R_2. A state $\acute{\mathcal{P}} = 11010$ has the hamming distances 2 and 3 from $\mathcal{P}_1$ & P_2, respectively. If $\acute{\mathcal{P}}$ is to be recognized, the recognizer must return $\mathcal{P}_1$. The *GMACA* of Fig. 3 if loaded with $\acute{\mathcal{P}} = 11010$, it returns the desired pattern $\mathcal{P}_1 = 10000$ after two time steps.

The search to arrive at a rule vector of *GMACA*, satisfying **R1** and **R2**, from all possible combinations of hybrid *CA* rules is of exponential complexity. So we fall back on *genetic algorithm (GA)* to arrive at the desired *GMACA* with pattern recognition capability.

IV. Evolution of GMACA

The aim of this evolution scheme is to arrive at the *GMACA* (rule vector) that can perform pattern recognition task. The rule vector of a *GMACA* is viewed as the chromosome for the current *GM* formulation. The following subsection reports some novel techniques of enhancing the fitness of initial population for *GA* to ensure fast convergence of evolution process.

A. Selection of Initial Population (IP)

Three elegant schemes for the selection of *IP* are proposed next.

IP From λ_{gr} Region: In this scheme the *IP* is constructed from the study of *CA* rule configurations. The rules are popularly characterized by Langton's parameter λ [1]. The number of 1's in a rule is quantified by λ and defined as $\lambda =$ (number of 1's in the rule number)/8. For example, λ value of Rule 90 (01 011 010) is 0.5 [Fig. 4(a)]. The present research work explores hybrid *CA* for which we introduce a parameter called λ_{av}, the average value of λ for all the cells in a *CA*. The λ_{av} for the hybrid *CA* of Fig. 4(b) is 0.425.

It has been observed that the *GMACA* rules, acting as efficient pattern recognizers, lie within a specific range of λ_{av} value (λ_{gr} region). The *IP* from that region ensures fast convergence of *GA*. The following hypothesis characterizes the λ_{gr}.

Hypothesis 1: λ_{gr} settles around 0.46 and 0.54 that are roughly equidistant from 0.5.

The analytical foundation of the hypothesis is reported next followed by experimental validation.

Analytical foundation: In a *GMACA*, the state $\acute{\mathcal{P}}_i$ while traversing toward $\mathcal{P}_i$ assumes new state $\acute{\mathcal{P}}_i(t)$ at time step t. During traversal, it is not always true that there is a continuous decrease in hamming distances. In Fig. 3, $\acute{\mathcal{P}}_i(t) = u = 10111$ is associated with pivot $d = 00111$ (*Attractor-2*). The hamming distance between u and d is 1. After one time step, $\acute{\mathcal{P}}_i(t+1) = x = 11111$, where the hamming distance $HD(x, d) = 2$. During its movement toward d the seed u exhibits an oscillation in terms of hamming distance, e.g., 1, 2, 3, 1. The *CA* rules having an equal balance of 0's & 1's in its next state function can ensure this phenomena. In otherwords, a *CA* having λ_{av} value around 0.5 is the better candidate for *IP*.

	Cell 1	2	3	4	5
rule	90	90	90	90	90

$\lambda_{av} = \lambda = 0.5$

a) Uniform CA

rule	100	130	150	170	180

$\lambda_{av} = 0.425$

b) Hybrid CA

Fig. 4. λ parameter values of uniform and hybrid CA.

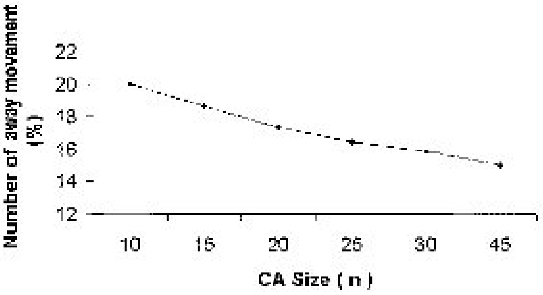


Fig. 5. Away movements for different values of n.

This fact is also validated through following experimentations done on the set of evolved pattern recognizers (*GMACA*).

a) We randomly generate 100 seeds and observe the movements of patterns toward a pivot $\mathcal{P}_i$ in a *GMACA*. Let assume $d' = HD(\acute{\mathcal{P}}_i(t), \mathcal{P}_i)$ at time step "t," while $d'' = HD(\acute{\mathcal{P}}_i(t+1), \mathcal{P}_i)$. If $d' < d''$, then it refers to as "*away movement*" of pattern from the pivot. The *away movements* of patterns for different values of n (pattern size) are shown in Fig. 5. For each n, we have taken 10 different *GMACA* and observed movements of 100 patterns for each *GMACA*. Out of 100 × 10 runs for an n, the number of cases (%) in which we encounter away movement is noted in Fig. 5.

It is observed that in 15% to 20% cases, the *away movement* occurs during the traversal of a pattern to the pivot. This validates the analysis that the hamming distance of $\acute{\mathcal{P}}_i$ from $\mathcal{P}_i$ undergoes oscillations in its movement toward $\mathcal{P}_i$.

b) Let us assume that during the traversal from seed $\acute{\mathcal{P}}_i$ to $\mathcal{P}_i$ the away movement first occurs at time step "t" and continues till $(t+m)$. The number of bits that flip away from the pivot $\mathcal{P}_i$ during this *away movement* is therefore $M = [HD(\acute{\mathcal{P}}_i(t+m), \mathcal{P}_i) - HD(\acute{\mathcal{P}}_i(t), \mathcal{P}_i)]$. The maximum value of M for all such durations $(t, t+m)$ (for traversal of $\acute{\mathcal{P}}_i$) is the "Magnitude of *away movement*" for $\acute{\mathcal{P}}_i$. Magnitude of "*away movement*" for a *GMACA* is computed as the average of away movements of a large number of seeds. Fig. 6 depicts magnitude of away movements for different values of n.

It can be observed from Fig. 6 that the away movement is limited to 20% of n—that is, for $n = 30$, as many as 5 b get flipped during the traversal from a seed to the nearest pivot. This fact demands—*GMACA* rules should have an equal balance of 1s and 0s.

c) To identify the range of λ_{av} value (λ_{gr}), the experiment has been conducted for $n = 10$ to 30. For each n, 15 different sets of patterns, to be learnt, are selected randomly. The number of patterns in a set is taken as $0.15n$,—the maximum number of n-bit patterns that a Hopfield network can recognize. The *GA* starts with the *IP* of 50 *CA* (chromosome) chosen at random and then undergoes evolution till 100% fit rules (*GMACA*) are obtained. Table I displays the mean and standard deviation of λ_{av} of the

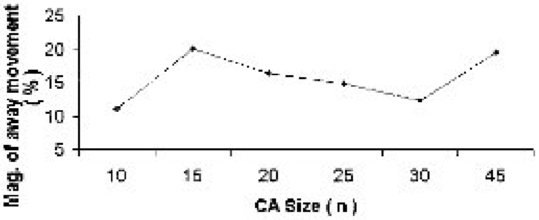
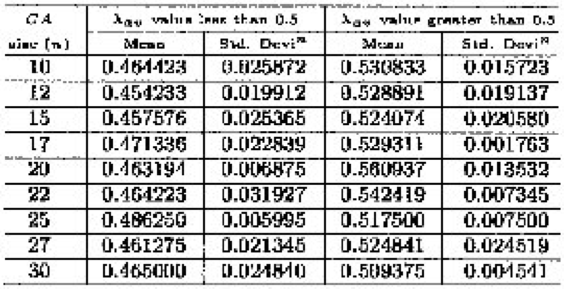


Fig. 6. Maximum away movement of seeds.

TABLE I
MEAN AND STANDARD DEVIATIONS OF λ_{av} VALUE OF GMACA



CA	λ_{av} value less than 0.5		λ_{av} value greater than 0.5	
size (n)	Mean	Std. Devi^n	Mean	Std. Devi^n
10	0.464423	0.025872	0.530833	0.015723
12	0.454233	0.019912	0.528891	0.019137
15	0.457576	0.025365	0.524074	0.020580
17	0.471336	0.022839	0.529311	0.001763
20	0.463194	0.006875	0.560937	0.013532
22	0.464223	0.031927	0.542419	0.007345
25	0.486250	0.005995	0.517500	0.007500
27	0.461275	0.021345	0.524841	0.024519
30	0.465000	0.024840	0.509375	0.004541

evolved *GMACA* rules for an n. *Column 3* indicates the mean of λ_{av} values below 0.5 while for λ_{av} above 0.5, the mean is noted in *Column 5*. The results of Table I report that the λ_{av} of desired *GMACA* are clustered around in the areas that are roughly equidistant (0.04) from 0.5 and, therefore, λ_{av} can be chosen as 0.46± or 0.54±.

IP From Graph Resolution Algorithm: This scheme to construct *IP* employs a reverse engineering technique. It constructs a set of *GMACA* rules for the patterns $P_1, P_2, \ldots, P_K$ to be learnt. The basic concept of mismatch pair algorithm proposed by Myer [20] is employed to construct the three-neighborhood *GMACA* considering patterns to be learnt as the members of different attractor cycles of a *GMACA*. The following steps illustrate the scheme.

Step 1) Construct basin for each P_i, pattern to be learnt, assuming P_i as the attractor state (single cycle) and each reachable state having "p" number of predecessors. The set of states $\{\acute{P}_i\}$ which follow **R2** of Section III are taken randomly as the predecessor nodes of P_i. Fig. 7 displays k arbitrary basins for the patterns $P_1, P_2, \ldots, P_k$ to be learnt assuming $p = 3$. $\acute{P}_{j(i)}$ is the jth noisy pattern of P_i, where $i = 1, 2, \ldots, k$ and $j = 1, 2, \ldots,$.

Step 2) Generate state transition table from the basins [20] created in *Step 1*. Fig. 8 illustrates a set of representative patterns of a complete state transition table.

Step 3) Generate rule vector of *GMACA* from the state transition table. The identification of rule for the ith cell of *GMACA* is done on the basis of the $(i-1)$th, ith and $(i+1)$th columns of state transition table. Let us consider the identification of rule for the second cell of *GMACA* from Fig. 8. States for the eight present state configurations of first, second, and third columns of Fig. 8 are

Neighborhood: 111 110 101 100 011 010 001 000
(i) NextState: 0 x 0 x 1 x 0 0.

x represents don't care. Randomly replacing don't cares by 0/1, we arrive at the rule. Therefore, the rule for the 2nd cell is 01 001 000—that is, 72.

The collision in the state transition table—that is, both 0 and 1 appear as the next state for a present state configuration, is resolved heuristically. In Fig. 8, we have shown collision for the fifth cell. The occurrence of "0" in the next state of 011 is $n_0 - 1$ while number of occurrence of "1" is $n_1 = 3$. In resolving the collision

a) we randomly select either 0 or 1 if $n_0 \simeq n_1$;
b) if $n_0 \gg n_1$ the next state is taken as 0 while for $n_1 \gg n_0$ it is 1.

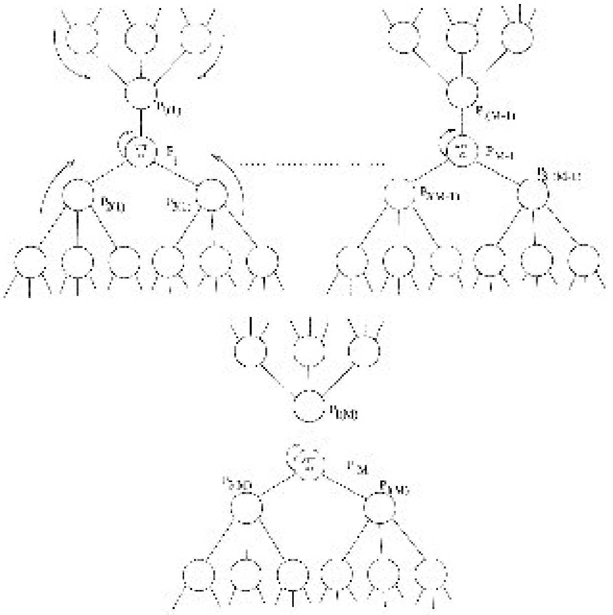
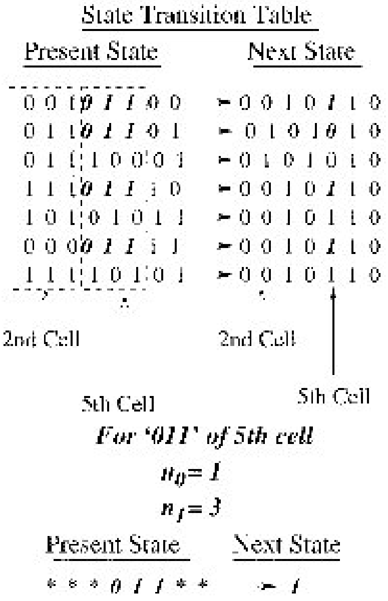


Fig. 7. Empirical basins created by graph resolution algorithm.



State Transition Table

Present State Next State

0 0 1 0 1 1 0 0 → 0 0 1 0 1 1 0
0 1 1 0 1 1 0 1 → 0 1 0 1 0 1 0
0 1 1 1 0 0 0 1 → 0 1 0 1 0 1 0
1 1 1 0 1 1 1 0 → 0 0 1 0 1 1 0
1 0 1 0 1 0 1 1 → 0 0 1 0 1 1 0
0 0 0 0 1 1 1 1 → 0 0 1 0 1 1 0
1 1 1 1 0 1 0 1 → 0 0 1 0 1 1 0

For '011' of 5th cell

$n_0 = 1$

$n_1 = 3$

Present State Next State

* * * 0 1 1 * * → 1

Fig. 8. Example of state transition table.

For the example design of Fig. 8, the next state for 011 is 1. Different sets of *CA* rules are derived by constructing approximate state transition diagrams for the attractor set $P_1, P_2, \ldots, P_k$. These rules are the member of *IP*.

IP From Mixed Rules: The *IP* from mixed rules comprises of the

i) chromosomes from λ_{av} region;
ii) chromosomes produced through graph resolution algorithm;
iii) randomly generated chromosomes;
iv) chromosomes that are produced from the concatenation of fit *CA* of smaller sizes.

The next two subsections report the principles employed for fast convergence of the *GA* evolution.

TABLE II
COMPARISON OF MEAN FITNESS AND NUMBER OF GENERATION REQUIRED TO CONVERGE WITH RANDOM AND PRESELECTED RULES

CA	Random (I)			λ_{pr} region (II)			Graph resolution(III)			Mixed rule (IV)			Execution
size	No of gen^n	Mean Fitness	Std. Dev^n	No of gen^n	Mean Fitness	Std. Dev^n	No of gen^n	Mean Fitness	Std. Dev^n	No of gen^n	Mean Fitness	Std. Dev^n	time (sec)
10	166	65.36	7.395	141	67.16	5.249	118	72.86	3.863	128	73.12	7.819	18.183
12	160	64.25	5.889	198	66.19	5.176	132	72.71	3.983	96	73.09	7.516	19.438
15	156	62.28	5.611	253	65.05	5.463	148	72.67	5.611	66	74.81	9.114	21.378
17	328	61.36	6.113	312	63.22	5.714	268	71.89	3.274	94	73.12	8.102	23.061
20	512	61.16	5.828	413	61.78	6.926	364	70.66	5.493	172	72.84	7.436	24.867
22	634	61.81	4.119	521	61.08	4.761	368	71.88	4.106	198	73.21	8.662	27.823
25	721	60.44	4.045	690	61.15	3.746	376	73.46	4.128	210	74.38	8.037	30.344
27	785	60.19	6.719	688	61.84	4.992	412	74.29	3.982	235	73.89	7.159	34.621
30	852	59.33	5.712	718	62.10	5.139	468	75.03	4.106	280	76.88	8.196	37.268
32	*	59.68	6.119	715	61.89	4.329	486	73.67	4.942	312	74.63	9.613	39.976
35	*	59.75	4.981	785	60.62	4.107	525	67.96	4.359	340	70.44	7.824	42.792
37	*	59.67	5.723	809	60.12	5.006	537	67.46	3.746	362	70.87	7.114	45.217
40	*	59.71	5.182	826	59.81	5.628	575	67.21	4.109	410	71.28	8.129	48.882
42	*	58.81	6.091	872	59.81	4.984	599	67.82	3.917	428	70.98	7.197	53.617
45	*	58.48	5.661	935	58.50	5.647	612	66.78	4.008	485	71.17	8.004	57.792

B. Fitness Function

The fitness $\mathcal{F}(C_r)$ of a particular chromosome C_r (that is, *CA* rule) in a population is determined by the hamming distance between a state $\acute{\mathcal{P}}_j$, generated by C_r and the desired attractor state $\mathcal{P}_j (j = 1, 2, \ldots k)$, the learnt pattern. A chromosome C_r is run with 300 randomly chosen initial seeds and the fitness of C_r is determined by averaging the fitness computed for the seeds.

Let us assume that, the chromosome C_r is run for a maximum iteration $\mathcal{L}_{max}$ with a seed and reaches to a state $\acute{\mathcal{P}}_j$. If $\acute{\mathcal{P}}_j$ is not the member of any attractor cycle, then the fitness value of C_r is considered as zero. On the other hand, if $\acute{\mathcal{P}}_j \in$ an attractor cycle containing $\mathcal{P}_j$, then the fitness of C_r is $(n- \mid \mathcal{P}_j - \acute{\mathcal{P}}_j \mid)/n$, where $\mid \mathcal{P}_j - \acute{\mathcal{P}}_j \mid$ is the hamming distance between $\mathcal{P}_j$ & $\acute{\mathcal{P}}_j$ and n is the maximum possible hamming distance between an input and the learnt pattern. Therefore, $\mathcal{F}(C_r)$ can be defined as

$$\mathcal{F}(C_r) = \frac{1}{q} \sum_{j=1}^{q} \frac{n- \mid \mathcal{P}_j - \acute{\mathcal{P}}_j \mid}{n} \tag{1}$$

where q is the number of random seeds.

C. Selection, Crossover and Mutation

From in-depth study of *GA* evolutions, we have set the associated parameters to derive next population (*NP*) out of present population (*PP*). The population size at each generation is set to 50. Out of which 35 chromosomes of *NP* are formed from single point crossover of *PP*. The five chromosomes of *NP* are formed from single-point mutation of the best 10 chromosomes of *PP*. We follow the elitist model and carry forward ten best solutions to the next generation. The detailed experimental results of next subsection validate the *GA* framework we have set to arrive at the desired solution.

D. Experimental Results

This section reports the performance of proposed *GMACA* based pattern recognition scheme in terms of cost and quality of the design. While the convergence rate of *GA* evolution gives the measure of computation cost, the quality of the design can be gauged from the memorizing capacity of *GMACA*.

Convergence Rate: The performance of different *IP* selection schemes in respect of *GA* convergence rate and the initial mean fitness with standard deviation is noted in Table II. The experiment has been done on Compaq server in Linux environment. The execution times, reported in the last column of Table II, clearly establish that the execution time increases linearly with n (pattern size). The entries "*" in *Column 2* of Table II signify that the *GA* with random *IP* does not converge within 1000 generations. The convergence of *GA* with random and preselected *IP* is also shown in Fig. 9. It can be observed that the rate of convergence for mixed rule is better than that with other *IP*. The convergence rates of *GA* with different classes of *IP* vary for two reasons:

i) the initial fitness of the population are different for different *IP*. For preselected *IP* it is better than the random *IP*;
ii) the more subtle reason is that the preselected *IP* can identify the right "schema."

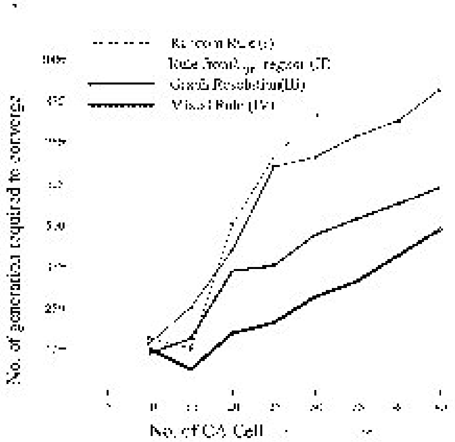


Fig. 9. GA convergence for different IP.

A schema corresponds to a chromosome. It is represented by a template of ones, zeros and asterisks, the asterisks are the wild cards. An example schema is $\langle * * * * 010 * * \rangle$. In the process of *GA* evolution at any instant of time the chromosomes of present population (*PP*) differ with their counter part in the next population (*NA*). The objective of selection process in *GA* is to gradually bias the schemas whose fitness is above average. The preselected rules enhances the convergence of *GA* because it includes schemas which help the genetic algorithm to climb through the correct path. The following experiment has been conducted to establish this fact.

For experimentation, we have started *GA* evolution for each *IP* selection scheme from a fixed level of fitness F (in Fig. 10, it is marked with dotted line). Subsequently, the number of generations needed to converge is noted for each case. Fig. 10 depicts that the better performance has been observed for the preselected *IP*. For example, shown in Fig. 10, the number of generations required to converge from fitness level F are 686, 531, 369, and 260 for the random, λ_{pr}, graph resolution and mixed *IP*, respectively. For each *IP* selection scheme, the 50 chromosomes (CR_F) of a population that reached the fitness

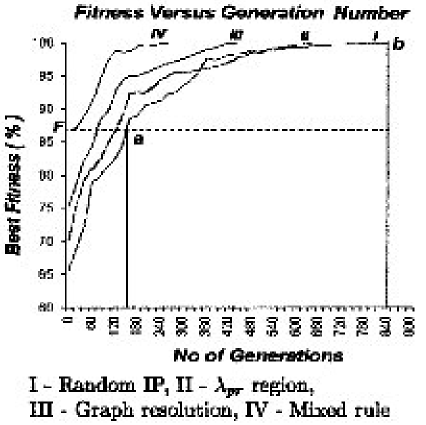


Fig. 10. Graph showing the number of generations required in different initial populations for a particular CA cell ($n = 25$).

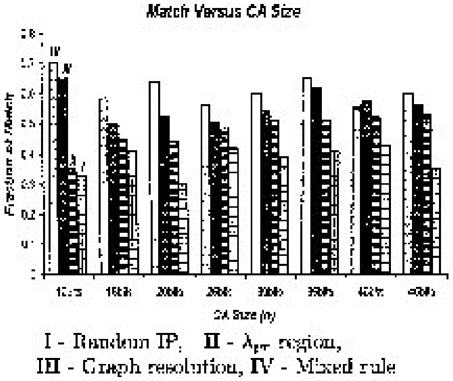


Fig. 11. Matching in different IP selection schemes.

level F are taken. Each such chromosome is compared bit by bit with 10 best chromosomes (CR_b) of final population to compute the *Fraction* of match defined as the number of bits found same between a pair (CR_F, CR_b) divided by n, n is the size of a chromosome.

For mixed *IP* the fraction of match is as high as 0.7 and its average value > 0.6 (Fig. 11). This validates higher convergence rate of *GA*.

Memorizing Capacity of GMACA: The experiments to evolve pattern recognizable n-cell *GMACA* for different values of n are carried out. For each n, 15 different sets of patterns to be trained are selected randomly. The number of patterns to be learnt by the *CA* is progressively increased. Table III demonstrates the pattern recognition capability of *CA* based design. *Column 2* of Table III depicts the maximum number of patterns that an n-cell *CA* can memorize—that is, the number of patterns for which the *GA* has obtained 100% fit rules. The results of conventional Hopfield Net on the same data set are provided in *Column 3* for the sake of comparison. Hopfield net, as reported in [8], memorizes $0.15n$ pattern of n-bit. **The experimental results clearly establish that the *GMACA* have much higher capacity to learn patterns in comparison to Hopfield Net.**

The rule space covered by the *GMACA*, capable of performing pattern recognition, has gone through extensive study. The diverse parameters proposed over the years to characterize the *CA* behavior are used to identify the class in which the *GMACA* belongs. The significance of the parameters and the results of the studies on *GMACA* are reported next.

V. *GMACA* CHARACTERIZATION

The *GMACA* has been characterized in respect of the parameters proposed in [1], [10], [16], [17], [19], [21] to study *CA* behavior. Discussion on each of the parameters follows.

TABLE III
PERFORMANCE OF THE *CA* BASED PATTERN RECOGNIZER

CA size (n)	CA based Pattern Recognizer	Conventional Hopfield Net
10	4	2
12	4	2
15	4	2
17	4	3
20	5	3
22	5	3
25	6	4
27	6	4
30	7	5
32	7	5
35	8	5
37	8	6
40	9	6
42	9	6
45	10	7

TABLE IV
SPACE TEMPORAL STUDY TO CATEGORIZE *CA* RULE SPACE

GMACA size (n)	Entropy		Mutual Information
	Mean	Std. Devi^n	
10	0.840966	0.072076	0.969
12	0.832549	0.062185	0.957
15	0.820147	0.064446	0.940
17	0.832854	0.071821	0.942
20	0.845909	0.057592	0.978
22	0.841167	0.042171	0.907
25	0.834503	0.065129	0.929
27	0.829847	0.038596	0.917
30	0.839001	0.070335	0.928
32	0.832786	0.052841	0.943
35	0.838102	0.071002	0.964
37	0.841829	0.030996	0.941
40	0.848010	0.061029	0.953
42	0.832871	0.054291	0.948
45	0.839116	0.0510321	0.933

A. Space Temporal Study

Dynamical behaviors of space-time patterns generated by the *CA* provide a guideline to characterize the *CA* rule space [1], [10], [19]. The macroscopic measurements of *CA* dynamics like *entropy,mutual information* are studied to classify the *GMACA* rules.

1) **Entropy** is the measure of randomness [22]. The maximum *entropy* (close to 1) [15] of a system signifies *chaotic behavior*, whereas low *entropy* indicates *ordered behavior*. In case of *complex* system, mean *entropy* is close to the critical value 0.84 with high variance [1].

   To measure the **entropy**, we select a moving window of 10 time steps ($w = 10$). The system has been run for 10 000 time steps from a random initial state. The mean *entropy* and the standard deviation from the mean have been computed [17]. For each *GMACA* the procedure is repeated for 15 times with different random initial states. The values shown in the *Columns 2* and *3* of Table IV are the mean and standard deviation of *entropy*, computed for the patterns evolved through *GMACA*. The mean value of *entropy*, shown in Table IV, for the *GMACA* clusters around 0.84 with high standard deviation. This fact points *GMACA* as the complex system.

2) **Mutual information** measures the correlation between patterns generated at a fixed time interval. If a pattern P_1 is the copy of P_2, then *mutual information* between P and P_2 is 1. The *mutual information* between two statistically independent patterns is 0. Both the ordered and chaotic *CA* rules do not create spatial structures and in effect generate pattern set with low *mutual informa-*

tion. On the other hand, the complex *CA* rules create highly correlated structures producing maximum *mutual information* [15].

To measure the *mutual information* among the patterns generated from *GMACA*, we follow the method proposed in [15]. We select patterns separated by a particular window of size 6. The *mutual information* corresponding to the evolved *GMACA* rules, noted in *Column 4* of Table IV, are found to be close to 1.

B. Z-Parameter

The *CA* rule space can be categorized by evaluating Z parameter [17]. The Z parameter provides an alternative to track the *CA* behavior very closely. It besides counting the fraction of 1's in the rule table also takes account of the allocation of 1's in the rule table for different categories of neighborhood relations. The details are reported in [21]. The Z parameter varies from 0 to 1. Z value close to 1 indicates chaotic behavior of the *CA*, while $Z = 0$ means order. The intermediate value of Z identifies the complex *CA* rules [21].

The Z parameters for the evolved *GMACA* rules of different size (n) are reported in Table V. *Column 2* depicts the mean value of Z while *Column 3* reports the standard deviation. The values, shown in the table, are in between 0 and 1.

C. Characterization of Attractor Basin

The global parameters such as *G-density, Maximum in-degree, In-degree frequency histogram* and *Transient length*, associated with the attractor basin topology, can be used to characterize *GMACA* rule space. These parameters point to the degree of convergence of dynamical flow of attractor basin.

1) **G-density** defines the density of garden-of-eden states, the states without pre-image (predecessor). For example, in Fig. 3, state "a" is the garden-of-eden state and pre-image of state "b."

2) **Maximum in-degree** is the maximum number of immediate pre-images of a state. In Fig. 3, *in-degree* of the state "d" is 3.

   Very high *G-density* (close to 2^n) and significant frequency of high *in-degree* (close to 2^n) of a *CA* imply short and dense trees, which correspond to ordered rules. While the *G-density* close to 1 implies low convergence, the long sparse trees with branching points having *in-degree* close to 1 implies chaos. The complex rules fall in between these two extremes [21].

   The *columns 2* and *3* of Table VI report the *G-density* and the *Maximum in-degrees* of the evolved *GMACA*. The proper balance between *G-density* and *Maximum in-degree* for a *GMACA* indicates *GMACA* rules as complex.

3) **In-degree frequency histogram** of a basin of attraction can be constructed as—horizontal axis representing the value of in-degree and vertical axis as the frequency of in-degree. It gives accurate measure of attractor basin topology and its convergence. The shapes of histogram indicate different *CA* dynamics. For a complex rule, the histogram follows power law distribution [17].

   Fig. 12 displays a typical In-degree frequency histogram for 15-cell *GMACA* that recognizes four patterns. The histogram exhibits power law distribution. The similar distribution is observed for all the evolved *GMACA*.

4) **Transient length** is the time required to reach an attractor cycle. In Fig. 3, *transient length* for "a" is 4. At *chaos*, the transient lengths of the patterns are close to 2^n, where n is the number of *CA* cells; whereas it is very short if order is maintained [21].The figures in *Column 4* of Table VI depict the *transient lengths* of the evolved *GMACA* rules. The observed transient lengths are quite long but much shorter than 2^n.

The above observations confirm beyond doubt that *GMACA* **rules are complex and lies at the edge of chaos.**

TABLE V
VALUES OF Z PARAMETERS OF *GMACA*

GMACA size (n)	Z parameter	
	Mean	Std. Devi^n
10	0.618	0.033
12	0.636	0.021
15	0.642	0.037
17	0.617	0.031
20	0.621	0.012
22	0.641	0.007
25	0.622	0.009
27	0.639	0.012
30	0.610	0.029
32	0.634	0.015
35	0.627	0.022
37	0.642	0.016
40	0.633	0.018
42	0.627	0.004
45	0.639	0.033

TABLE VI
PARAMETERS OF ATTRACTOR BASIN TOPOLOGY

GMACA size (n)	*G* Density	Maximum In-degree	Transient Length
10	70.425	16	213
12	73.687	22	728
15	78.223	39	916
17	78.873	102	1274
20	80.864	174	1559
22	81.642	263	2036
25	83.281	483	3669
27	83.011	512	4128
30	83.342	724	5138
32	84.112	898	5662
35	85.127	1012	7210
37	85.343	2136	7865
40	86.261	4152	9120
42	87.369	6211	9532
45	88.418	7122	11020

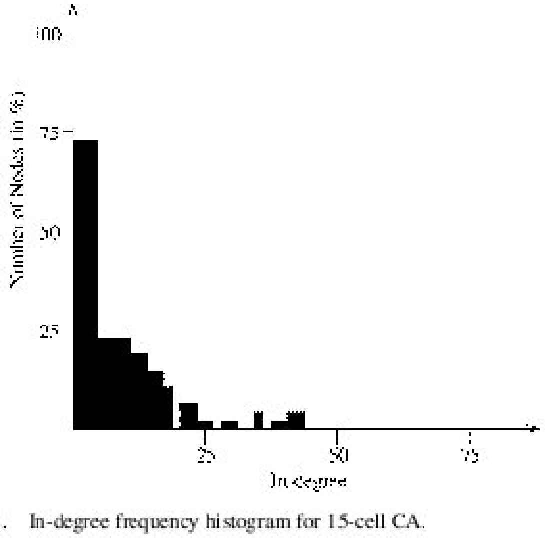


Fig. 12. In-degree frequency histogram for 15-cell CA.

VI. CONCLUSION

This paper presents a comprehensive overview of the potential of cellular automata (*CA*) to act as an associative memory model. The potential has been explored by using genetic algorithm (*GA*) to evolve the desired model of *CA* referred to as *GMACA*. Prudent selection of initial population ensures fast convergence of *GA*. The *GMACA* has been found to bear the properties of class IV *CA* that can perform complex computations. The memorizing capacity of *GMACA* is found to be higher than that of Hopfield Network.

References

[1] C. G. Langton, "Computation at the edge of chaos: Phase transition and emergent computation," *Phys.D*, vol. 42, 1990.

[2] M. Mitchell, *Introduction to Genetic Algorithm*, 5th ed. Cambridge, MA: MIT Press, 1999.

[3] J. Buhmann, R. Divko, and K. Schulter, "Associative memory with high information content," *Phys. Rev. A*, vol. 39, pp. 2689–2692, 1989.

[4] G. A. Carpenter, "Neural network models for pattern recognition and associative memory," *Neural Networks*, vol. 2, no. 4, pp. 243–257, 1989.

[5] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*. Sante Fe, NM: Addison-Wesley, 1991.

[6] M. Chady and R. Poli, "Evolution of cellular-automaton-based associative memories," CSRP, Tech. Rep. CSRP-97-15, 1997.

[7] J. J. Hopfield, "Neural networks and physical system with emergent collective computational abilities," in *Proc. Nat. Acad. Sci.*, vol. 79, 1982, pp. 2554–2558.

[8] ——, "Pattern recognition computation using action potential timings for stimulus representations," *Nature*, vol. 376, pp. 33–36, 1995.

[9] L. Wang, "Oscillatory and chaotic dynamics in neural networks under varying operating conditions," *IEEE Trans. Neural Networks*, vol. 7, pp. 1382–1388, Nov. 1996.

[10] M. Mitchell, P. T. Hraber, and J. P. Crutchfield, "Revisiting the edge of chaos: Evolving cellular automata to perform computations," *Complex Syst.*, vol. 7, pp. 89–130, 1993.

[11] P. Tzionas, Ph. Tsalidas, and A. Thanailakis, "A new cellular automata based nearest neighbor pattern classifier and its *VLSI* implementation," *IEEE Trans. VLSI Syst.*, vol. 2, pp. 343–353, June 1994.

[12] E. Jen, "Invariant strings and pattern recognizing properties of 1D CA," *J. Stat. Phys.*, vol. 43, 1986.

[13] N. Ganguly, A. Das, P. Maji, B. K. Sikdar, and P. P. Chaudhuri, "Evolving cellular automata based associative memory for pattern recognition," in *Proc. Int. Conf. High Performance Comput.*, Hyderabad, India, 2001.

[14] P. P. Chaudhuri, D. R. Chowdhury, S. Nandi, and S. Chatterjee, *Additive Cellular Automata, Theory and Applications*. Los Alamitos, CA: IEEE Computer Society Press, 1997, vol. 1.

[15] W. Li, N. H. Packard, and C. G. Langton, "Transition phenomena in cellular automata rule space," *PhysicaD*, vol. 45, 1990.

[16] A. Wuensche, "Complexity in One-D Cellular Automata," Tech. Rep., Santa Fe Institute, Working Paper 94-04-025, 2002.

[17] ——, "Classifying Cellular Automata Automatically," Tech. Rep., Santa Fe Institute, 2002.

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

[19] S. Wolfram, *Theory and Application of Cellular Automata*. New York: World Scientific, 1986.

[20] J. E. Myers, "Random boolean networks—Three recent results," Private Communication, 2002.

[21] A. Wuensche, "Complexity in 1-d cellular automata; basins of attraction and the Z parameter," Tech. Rep., Univ. of Sussex, Sussex, U.K., Cognitive Science Research Paper 321, 1994.

[22] P. A. Dufort and C. J. Lumsden, "The complexity and entropy of Turing machines," in *Proc Workshop Phys. Comput.*, 1994, pp. 227–232.

An Iterative Solution to Dynamic Output Stabilization and Comments on "Dynamic Output Feedback Controller Design for Fuzzy Systems"

Min-Long Lin and Ji-Chang Lo

***Abstract*—In this note, we will show that the output feedback controller gains K in the paper [1] is only an approximated solution $K = QP^{-1}\tilde{C}_0^{-}$, with the dagger denoting Moore–Penrose inverse of the matrix $\tilde{C}_0$. Consequently $K \neq K^{\ddagger}$ and therefore it may not satisfy the linear matrix inequality (LMI) constraints in the aforementioned paper. Instead, an iterative LMI approach is suggested to solve the dynamic output stabilization problem for the fuzzy systems.**

***Index Terms*—Iterative linear matrix inequality (ILMI), linear matrix inequality (LMI), Takagi–Sugeno fuzzy model.**

I. Main Result

In the formulation of the main results we admit the fuzzy system formulation and adopt the same matrix notations as those in [1].

For brevity, the closed-loop system with dynamic output feedback controller incorporated is

$$\dot{x}_d(t) = A_d(t)x_d(t) \tag{1}$$

$$y(t) = C_0 x(t) \tag{2}$$

where

$$\begin{aligned} x_d &= \begin{bmatrix} x^T & x_c^T \end{bmatrix}^T \\ A_d(\mu) &= \begin{bmatrix} A_0 - \triangle A(\mu) & 0 \\ 0 & 0 \end{bmatrix} \\ &\quad + \begin{bmatrix} B_0 + \triangle B(\mu) & 0 \\ 0 & I \end{bmatrix} K \begin{bmatrix} C_0 & 0 \\ 0 & I \end{bmatrix} \\ K &= \begin{bmatrix} D_c & C_c \\ B_c & A_c \end{bmatrix} \end{aligned}$$

and

$$E_A E_A^T > \triangle A(\mu)\triangle A^T(\mu) \tag{3}$$

$$E_B E_B^T \geq \triangle B(\mu)\triangle B^T(\mu). \tag{4}$$

We emphasize that there are many papers dealing with quadratic uncertainty bounds. Interested readers could refer to Riccati-based methods [2]–[8], linear matrix inequality (LMI)-based methods [9]–[16], iterative linear matrix inequality (ILMI)-based methods [17]–[19] and references therein for details.

Given the setup, the paper [1] proceeds to solve the fuzzy stabilization problem by defining a new variable $Q = K\tilde{C}_0P$ such that the Q, if it exists, can be obtained via an LMI feasibility problem. The inconsistency arises when one tries to solve for K given Q. What was obtained in the paper for the controller gains was actually $K^{\ddagger}$ and usually $K \neq K^{\ddagger}$ unless $\tilde{C}_0$ is invertible. This inconsistency implies that the gain obtained, $K^{\ddagger} - QP^{-1}\tilde{C}_0^{-} - QP^{-1}\tilde{C}_0^T(\tilde{C}_0\tilde{C}_0^T)^{-1}$, may not satisfy the LMI constraint and therefore, no conclusion is drawn for the fuzzy stabilization problem via dynamic output feedback. To see this, let $Q^{\ddagger} = K^{\ddagger}\tilde{C}_0P$. Since $K \neq K^{\ddagger}$, to stabilize the fuzzy systems, it requires that the time derivative of a Lyapunov function must be $\dot{V}(Q^{\ddagger}) < 0$. Unfortunately, this is not guaranteed. In fact, the setting of a dynamic output feedback stabilization in the paper is a generalized static output feedback stabilization problem, which is a nonconvex

Manuscript received July 18, 2001; revised September 12, 2002. This work was supported in part by the National Science Council of Taiwan, R.O.C., under Grant NSC 90-2213-E-008-039. This paper was recommended by Associate Editor C. P. Neuman.

The authors are with the Department of Mechanical Engineering, National Central University, Jung-Li, Taiwan, R.O.C.