GENERALIZED MULTIPLE ATTRACTOR CELLULAR AUTOMATA (GMACA) MODEL FOR ASSOCIATIVE MEMORY

NILOY GANGULY
Computer Centre, IISWBM, Calcutta, India 700073
nganguly@hotmail.com

PRADIPTA MAJI*, BIPLAB K. SIKDAR† and P. PAL CHAUDHURI‡
Department of Computer Science and Technology,
Bengal Engineering College (DU), Howrah, India 711103
*pradipta@cs.becs.ac.in
†biplab@cs.becs.ac.in
‡ppc@cs.becs.ac.in

This paper reports an efficient technique of evolving Cellular Automata (CA) as an associative memory model. The evolved CA termed as GMACA (Generalized Multiple Attractor Cellular Automata), acts as a powerful pattern recognizer. Detailed analysis of GMACA rules establishes the fact that the rule subspace of the pattern recognizing CA lies at the edge of chaos — believed to be capable of executing complex computation.

Keywords: Cellular automata; GA; edge of chaos; pattern recognition.

1. Introduction

Pattern Recognition demands automatic identification of objects and images by their shapes, forms, outlines or some other attributes. In conventional approach of pattern recognition the machine compares the given input pattern with each of the stored patterns and identifies the closest match. The search requires the time proportionate to the number of patterns learnt/stored. Obviously, with the growth of the number of stored patterns, the recognition process becomes slow.

The Associative Memory Model provides a solution to the problem where the time to recognize a pattern is independent of the number of patterns learnt/stored. The solution based on this model divides the entire state space into some pivotal points that represent the patterns learnt. The states close to a pivotal point get associated with a specific learnt pattern. Identification of an input pattern (with or without distortion due to noise) amounts to traversing the transient path from the given input pattern to the closest pivotal point. As a result, the process of recognition becomes independent of the number of patterns learnt.

Since early 1980s the model of associative memory has attracted considerable interest among the researchers. Both sparsely connected machine (Cellular
Automata)\(^2\) and densely connected network (Neural Net)\(^5\) have been explored to address the problem of pattern recognition. The seminal work of Hopfield\(^6\) made a breakthrough by modeling a recurrent, asynchronous, neural net as an associative memory system. However, the complex structure of neural net model has partially restricted its application.

Search for alternative model around the simple structure of Cellular Automata (CA) continued.\(^2\);\(^7\);\(^12\) The simple, regular, modular, cascadable local neighborhood structure of Cellular Automata (CA) serves as an excellent sparse network. In this paper, we propose the evolution of CA based associative memory model for pattern recognition. The class of CA termed as Generalized Multiple Attractor CA (GMACA) is employed to model an Associative Memory. The evolved GMACA displays very encouraging results for pattern recognition. The memorizing capacity of GMACA can be found to be better than that of conventional Hopfield Net. Exhaustive experimental results reported in this paper confirm this comparative advantage of CA technology.

The evolved GMACA rule space is next subjected to extensive study. The study categorizes the GMACA rules through in-depth analysis of its local and global parameters. The diverse parameters like \(\chi^8\) and \(Z^{14}\), entropy, mutual information, etc. studied by earlier authors to characterize the CA, are investigated. The GMACA that are found to lie in between order and chaos (defined as the edge of chaos\(^10\)) displays higher memorizing capability.

In order to present the underlying principle of CA based associative memory model, in Sec. 2, we present a CA overview. The GMACA based associative memory and its application for pattern recognition is next outlined in Sec. 3. In Sec. 4, we have detailed the evolution designed with Genetic Algorithm (GA). Appropriate selection of various parameters of GA in order to speed up the convergence process are also covered in Sec. 4. Finally, exhaustive experimental result reporting the memorizing capability of GMACA model is presented in Sec. 5.

2. Cellular Automata Preliminaries

A Cellular Automaton (CA) is a discrete system which evolves in discrete space and time. It consists of a large number of cells, organized in the form of a lattice. Each cell acts as a single processor and communicates only with the neighboring cells. The next state of a cell depends on its own state and the states of its neighboring cells. In this paper we develop our model around three-neighborhood (left neighbor, self and right neighbor) one-dimensional CA and each CA cell having only two states — 0 or 1.

In a two-state three-neighborhood CA, there can be a total of \(2^3\) i.e. 256 distinct next state functions of a cell referred to as the rule of CA cell.\(^13\) The following is an illustration for two such rules, 90 and 150:
Neighborhood: 111 110 101 100 011 010 001 000

(i) NextState: 0 1 0 1 1 0 1 0 90
(ii) NextState: 1 0 0 1 0 1 1 0 150.

The first row lists the possible combinations of present states (left, self and right) for a three-neighborhood CA cell at time \( t \). The next two rows list the following states for the \( i \)th cell at time instant \( (t+1) \) — the decimal equivalent of the 8 bit binary number is referred to as Rule Number. The number of 1’s in a rule is quantified by Langton’s parameter \( \lambda^8 \) — it is equal to the number of 1’s in the binary rule number divided by 8. For example, \( \lambda \) value of rules 90 and 150 is 0.5.

The concept of *Multiple Attractor Cellular Automata* (MACA) has been introduced in the book. Its state transition behavior consists of multiple components — each component, as noted in Fig. 1, is an inverted tree. Each node with a self-loop in a component is referred to as an *Attractor*. The nodes with binary numbers 0000, 0001, 1000 and 1001 are the attractors in the MACA of Fig. 1 with rule vector \langle 150, 102, 60, 150 \rangle — i.e. rule 150 is applied on left most cell, followed by rule 102 on next one and so on. The set of nodes in a component with attractor \( \alpha \) is referred to as \( \alpha \)-basin. Even though such an MACA displays interesting characteristics, it employs only 14 XOR/XNOR rules out of 256 rules. It seems unlikely that pattern recognition can be done effectively with such a CA referred to as *Additive* CA. Hence, the present research work explores a more general class of MACA referred to as *Generalized Multiple Attractor* CA (GMACA) designed with nonlinear rules implementing AND/OR logic. A GMACA is a hybrid CA (different rules applied to different cells) and can model an associative memory to perform pattern recognition task. Figure 2 illustrates the state space of a four-cell hybrid GMACA with rule vector \langle 98, 236, 226, 107 \rangle. The state space of this CA is divided into two attractor basins. The states not covered by the attractor cycles are viewed as *Transient States* in the sense that a CA finally settles down in one of its attractor cycles after passing through such transient states.

![State space of a four-cell CA divided into four attractor basins.](image-url)
The state transition behavior of a GMACA (Fig. 2) roughly corresponds to that of an MACA with the following differences:

— Unlike single cycle attractor of an MACA (Fig. 1), an attractor in a GMACA refers to a cycle with single/multiple nodes. The two components in Fig. 2 have attractors of cycle lengths 3 and 4.
— All the inverted trees rooted on attractor node are identical in an MACA built with additive CA. By contrast, such trees are not identical in a GMACA designed with nonlinear rules.

In the next section, we have concentrated on evolving hybrid nonlinear GMACA and its attractive features to memorize the large number of patterns.

3. GMACA for Pattern Recognition

The pattern recognizer is trained to memorize some specific pattern set $P = \{P_1, \ldots, P_i, \ldots P_k\}$ and it can recognize an incoming pattern $P_i$ even if the pattern is corrupted with limited noise. When the new pattern $P_i$ is input to the system, the pattern recognizer identifies it as $P_i$, where $P_i$ is the closest match to $P_i$. If the entire pattern is viewed as a binary string, then the hamming distance between $P_i$ and $P_i$ is the least among all $P_i$’s. The hamming distance between $P_i$ and $P_i$ is the measure of noise.

A GMACA having its state space distributed into disjoint basins (Fig. 2) with transient and attractor states, models an associative memory. The entire state space built around some pivotal points can be viewed as the state space generated by a GMACA with its attractors and transient states. Memorizing the set of pivotal patterns...
points \{P_1, \ldots, P_i, \ldots, P_k\} is equivalent to the design of a GMACA with the pivotal points as the states in attractor cycle. Any other transient point \(\hat{P}_i\), in close vicinity of a pivotal point \(P_i\), can be considered as a pattern with some noise. The correct output \(P_i\) can be produced in time proportionate to the state traversal of the GMACA.

A GMACA designed to perform pattern recognition tasks should maintain the following two relations:

**R1:** Each attractor basin of the GMACA should contain one and only one pattern \((P_i)\) to be learnt in its attractor cycle; for the rest of the paper, the corresponding basin is referred to as \(P_i\)-basin.

**R2:** The hamming distance of each state \(S_i \in P_i\)-basin with \(P_i\) is lesser than the hamming distance of \(S_i\) with any other \(P_i\)'s.

Figure 2 illustrates a four-cell GMACA that maintains both R1 and R2. It learns two patterns, \(P_1 = 0000\) and \(P_2 = 1111\). The state \(\hat{P} = 1110\) has the hamming distances 3 and 1 with \(P_1\) and \(P_2\) respectively. If \(\hat{P}\) is given as the input to be recognized, then the recognizer designed with the GMACA of Fig. 2 is loaded with \(\hat{P} = 1110\). The GMACA returns the desired pattern \(P_2\) after two time steps.

The next section reports the GA based searching scheme implemented to evolve the GMACA tuned for a set of patterns to be recognized. The search is made to arrive at a rule vector of a GMACA that satisfies both the relations **R1** and **R2**. All possible combinations of hybrid CA rules are the candidates for initial population. Since the search space is exponentially large, we employ and appropriately tune the Genetic Algorithm to arrive at the desired GMACA.

4. Design of Genetic Algorithm for Evolution of GMACA

The aim of this design is to evolve the GMACA (rule vector) that can perform pattern recognition task. This section describes the GA based solution to evolve the GMACA rules with the desired functionality.

Genetic Algorithms (GA) are stochastic search and optimization algorithms which are inspired by the mechanics of natural selection and genetics. The effectiveness of genetic algorithms can depend crucially on how they are carried out. In order to apply a genetic algorithm effectively, two types of decisions must be made. The first is the choice of representation — solutions to the problem must be represented as strings and genetic search operators (e.g. crossover, selection, mutation). Good choices of problem representation and parameter values can lead to genetic algorithms that are extremely effective; poor choices can render the method little better than random search.

The solutions of the search space, encoded in string-like structures, are referred to as chromosomes. The chromosomes (population) undergo evolution (reproduction) for a number of generations. The reproductive plan essentially consists of tuning the three genetic operators, namely, selection, crossover and mutation.
In our model, the GA starts with an initial population (IP) of 50 chromosomes (CA rules). Each CA rule evolves over several generations under selection (elitist model), crossover and mutation. The evolution process is controlled by the fitness function $F(C_r)$.

4.1. **Fitness function**

The fitness $F(C_r)$ of a particular chromosome $C_r$ (CA rule) in a population is determined by the hamming distance between the attractor $P_i$, evolved for a state from the run (generation), and the desired attractor $P_i$. The chromosome is run with 300 randomly chosen initial configurations (ICs) and fitness of CA is determined by averaging the fitness for each individual IC.

Let us assume that, a chromosome $C_r$ is run for the maximum number of iterations ($L_{max}$) for an IC and it reaches a state $P_i$. If $P_i \notin$ any attractor cycle, that is, it is still a transient state, then the fitness value of $C_r$ is considered as zero. On the other hand, if $P_i \in$ an attractor cycle containing $P_i$, then the fitness of $C_r$ is $\frac{n-|P_i-P_i|}{n}$, where $n$ is the length of the CA rule (chromosome). But, if any attractor other than $P_i$ exists in that attractor cycle, then fitness of $C_r$ is also considered as zero. Therefore, fitness function

$$F(C_r) = \frac{1}{k} \sum_{i=1}^{k} \frac{n-|P_i-P_i|}{n}$$

where $k$ is the number of random ICs.

4.2. **Selection**

Selection is the mechanism which increases the mean fitness of a population by choosing more fit individuals with a higher probability than unfit ones. This operator selects chromosomes in the population for reproduction. The fitter the chromosome, the more times it is likely to be selected to reproduce. In selection policy, principle of the survival of the fittest determines the chromosomes that should be propagated to the subsequent generations. In the elitist model of the GA, the best set of chromosomes are retained for the next generation. In our experiment, we follow the elitist model. The number of chromosomes to be retained for the next generation can be the same over generations or it can vary with generations. We have experimented with both the techniques, in order to determine which selection procedure enhances the convergence rate. It is seen that selection depending on generation converges fast. A detail description of the experiment with both the techniques is reported below.

**Fixed Selection.** In each generation, some fixed number of chromosomes is selected depending upon some fitness criterion. For the current version we have selected the value as 10.
**Selection Depending Upon Generation Number.** In each generation the number of chromosomes is selected depending upon generation number. Number of chromosomes selected in a particular generation is

\[ S = a + b \cdot \exp(-kG_n) \]  

where \( G_n \) is the generation number and \( a, b \) and \( k \) are constants. From extensive experimentation, we have set the values of \( a, b \) and \( k \) as 5, 40 and 0.125. The value of \( S \) decreases over generation, thus the scheme gives more weightage to crossover and mutation for higher generations.

The better convergence rate of the second technique is illustrated through Fig. 3. The graph of Fig. 3 is a representative case. It compares the convergence rate with respect to the mean fitness as well as best fitness of the above two selection schemes for CA of size 15. *It has been observed that the dynamic selection scheme based on generation numbers outperforms the first one. Hence, our GA employs this scheme.*

![Graph showing the comparison of best and mean fitness between fixed and selection depending upon generation](image)

**4.3. Crossover**

This operator randomly chooses a locus and exchanges the subsequences before and after that locus between two chromosomes to create two offspring. The crossover operator helps to explore the search space by virtue of providing means to generate new solutions out of the old ones. The probability of selecting a pair of chromosomes to be employed for crossover depends upon the fitness of chromosomes.
Moreover, a majority of chromosomes for the next generation are produced through
the crossover process. Here, we have experimented with two different techniques of
crossover to speed up the rate of convergence. However, there is no major difference
in performance between the two techniques.

**Single Point Crossover.** In case of single point crossover, one crossover point
is selected randomly. The rule vectors from the beginning of a chromosome to the
crossover point is copied from one parent, the rest is copied from the second parent.

**Double Point Crossover.** In this case, two crossover points are selected randomly
and the rule vectors in between two chromosomes are copied from one parent, while
the rest is copied from the other parent.

Experiments show that in both cases the mean fitness and the best fitness of
a particular generation are more or less same. Also, the number of generations re-
quired to converge the GA is same for both cases. Figure 4 illustrates a typical case,
and it also represents the rate of increase of mean and best fitness of a population
using single point and double point crossover. The population is taken to evolve
CA of size 15.

*It has been observed that the double point crossover does not improve the re-
results obtained with single point crossover. Hence, in order to reduce computational
overhead we employ single point crossover in our GA.*

![Graph showing the comparison of best and mean fitness between single point and double point crossover (CA of size 15).](image-url)
4.4. Mutation

This operator randomly flips some of the bits in a chromosome. The mutation, applied to perturb one or more solutions, ensures that the search space explored is not closed under crossover. The probability of mutation on a given chromosome is kept very low. In our problem, 10% of the NP (next population) are generated out of mutations of the elite rules. Here we also have experimented with two standard techniques of mutation.

**Single Point Mutation.** In case of single point mutation, one rule has been randomly selected. The rule is then converted into a binary string. A single bit is selected and then the bit is inverted.

**Double Point Mutation.** In case of double point mutation, two rules are randomly selected and each rule is mutated using the same principle of single point mutation. It is seen that the convergence rate of the GA is much better in case of single point mutation than double point. Figure 5, similar to Figs. 3 and 4, shows the mean and best fitness evolved over generations in both cases for a representative CA of size 15. Thus we decide in favor of single point mutation.

From the exhaustive experimentations of various parameters, we have observed that the mean fitness as well as best fitness of a particular generation are much better in case of selection based on generation number and single point crossover and

![Graph showing the comparison of best and mean fitness between single point and double point mutation.](image)
Our design of the Generic Algorithm is thus guided by this formalization.

The evolution process is carried out to obtain GMACA of different sizes. Side by side, for each size the number of patterns to be learnt by the GMACA is progressively increased during experimentation.

Table 1 demonstrates the pattern recognition capability of a GMACA based associative memory. Column 2 of Table 1 depicts the maximum number of patterns that an \( n \)-cell GMACA can memorize. The results of Hopfield Net are provided in Column 3 for the sake of comparison. The experimentation clearly indicates that the GMACA have much higher storage capacity in comparison to Hopfield Net.

In this context, it is worthwhile to study and characterize the rule space of GMACA that displays encouraging results for pattern recognition. The next section characterizes the GMACA. Summing up, our design of the GA is thus guided by the principle of (a) selection depending upon generation number (b) single point crossover and (c) single point mutation.

5. Characterization of GMACA Rule Space

In this section, we make an extensive study on the evolved GMACA rule space. The study is directed towards the categorization of GMACA rules implementing the complex task of pattern recognition. The study has been performed

(i) by exploring CA Rule Table, and
(ii) by exploring the dynamical behavior of the Space-Temporal Patterns generated by the CA set.

All the study points to the fact that the rule space of GMACA lies on the edge of chaos — a space which is believed to be supporting complex computational tasks.

5.1. Exploring CA rule table

The CA rule space can be characterized by evaluating the parameters — \( \lambda \) and \( \gamma \). Different values of these parameters correspond to different CA dynamics.
Table 2. Parameters characterizing rule space.

<table>
<thead>
<tr>
<th>CA (n)</th>
<th>$\lambda_{av} &gt; 0.5$</th>
<th>$\lambda_{av} &lt; 0.5$</th>
<th>$Z$ Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Std. Dev.</td>
<td>Mean</td>
</tr>
<tr>
<td>10</td>
<td>0.537</td>
<td>0.020</td>
<td>0.472</td>
</tr>
<tr>
<td>15</td>
<td>0.524</td>
<td>0.021</td>
<td>0.458</td>
</tr>
<tr>
<td>20</td>
<td>0.512</td>
<td>0.005</td>
<td>0.487</td>
</tr>
<tr>
<td>25</td>
<td>0.518</td>
<td>0.007</td>
<td>0.486</td>
</tr>
<tr>
<td>30</td>
<td>0.509</td>
<td>0.005</td>
<td>0.465</td>
</tr>
<tr>
<td>35</td>
<td>0.511</td>
<td>0.021</td>
<td>0.481</td>
</tr>
<tr>
<td>40</td>
<td>0.523</td>
<td>0.011</td>
<td>0.476</td>
</tr>
<tr>
<td>45</td>
<td>0.521</td>
<td>0.013</td>
<td>0.489</td>
</tr>
</tbody>
</table>

Characterization of the uniform CA state transition, based on the value of $\lambda$, is reported in Ref. 8. The number of 1’s in a rule is quantified by the Langton’s parameter $\lambda$. As the $\lambda$ value is shifted from 0 to 0.5, the average behavior of the CA passes through the transitions: homogeneous $\rightarrow$ periodic $\rightarrow$ complex $\rightarrow$ chaotic. The CA behaves in the reverse order when the $\lambda$ value is incremented from 0.5 to 1. The range of $\lambda$ for which the rule exhibits complex function is referred to as $\lambda_c$. All complex computations are likely to occur near $\lambda_c$. The region is referred to as the edge of chaos. The subsequent discussions introduce the associated parameters for the hybrid CA we have studied.

For the hybrid CA domain we introduce the parameter $\lambda_{av}$, the average of $\lambda$ values for different CA cells. In Table 2, we provide the $\lambda_{av}$ values for different GMACA rules, evolved for different $n$. Column 2 indicates the resulted mean of $\lambda_{av}$ above 0.5 while Column 4 indicates mean value below 0.5. Corresponding standard deviations are noted in the Columns 3 and 5 respectively. Table 2 illustrates that the $\lambda_{av}$ of the evolved GMACA are clustered around in the areas that are roughly equidistance from 0.5 (from 0.433 to 0.494 and from 0.503 to 0.557) which corresponds to the region at edge of chaos.

From Table 2, it is seen that CA with rules having close to an equal number of 0’s and 1’s tend to be much better pattern recognizers than CA with rules having a relatively high degree of homogeneity. In CA with more homogeneous rules, the states of too many cells freeze early in the dynamics. By frozen cell, we mean cell whose state does not change from sometime onwards in the CA dynamics. The tendency of freezing increases as the rule becomes more and more homogeneous. Hence, the CA formed with such rules does not acquire the capacity of accommodating noise.

The value of $Z$ varies from 0 to 1. The $Z$ value close to 1 indicates chaotic behavior of the CA, while $Z = 0$ indicates order. Any intermediate value of $Z$ identifies complex CA rules. The $Z$ parameters of evolved GMACA rules for different values of $n$ are also reported in Table 2. Column 6 depicts the mean value of $Z$ and Column 7 contains the standard deviation of $Z$ parameter among the evolved GMACA rules. The $Z$ parameter values for the evolved GMACA are found intermediate between 0 and 1. It indicates, the GMACA rules are complex rules.
Table 3. Space temporal study to categorize CA rule space.

<table>
<thead>
<tr>
<th>CA Size (n)</th>
<th>Entropy Mean</th>
<th>Std. Deviation</th>
<th>Mutual Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.813</td>
<td>0.024</td>
<td>0.729</td>
</tr>
<tr>
<td>15</td>
<td>0.921</td>
<td>0.033</td>
<td>0.740</td>
</tr>
<tr>
<td>20</td>
<td>0.859</td>
<td>0.037</td>
<td>0.638</td>
</tr>
<tr>
<td>25</td>
<td>0.903</td>
<td>0.021</td>
<td>0.829</td>
</tr>
<tr>
<td>30</td>
<td>0.867</td>
<td>0.030</td>
<td>0.688</td>
</tr>
<tr>
<td>35</td>
<td>0.878</td>
<td>0.031</td>
<td>0.724</td>
</tr>
<tr>
<td>40</td>
<td>0.857</td>
<td>0.052</td>
<td>0.713</td>
</tr>
<tr>
<td>45</td>
<td>0.839</td>
<td>0.021</td>
<td>0.683</td>
</tr>
</tbody>
</table>

5.2. Space temporal study

Dynamical behavior of space-time patterns generated by the GMACA is another way to characterize the CA rule space. The macroscopic measurements of CA dynamics like entropy, mutual information are used to classify the CA rules.

Entropy is the measure of randomness of a system. The maximum entropy (close to 1) of a system signifies chaotic behavior. The values shown in the Columns 2 and 3 of Table 3 are the mean and standard deviation of entropy, computed for the evolved GMACA rules. The high entropy and low standard deviation indicate that the evolved GMACA, performing pattern recognition task, are at the edge of chaos.

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 $P_1$ and $P_2$ is 1. Whereas, the mutual information between two statistically independent patterns is 0. The ordered CA rules do not create spatial structures, in effect generate pattern set with low mutual information. On the other hand, the complex CA rules create highly correlated structures. To measure the mutual information, we select patterns generated from the evolved GMACA rule, separated by the window of size 6. The mutual information value corresponding to an evolved GMACA rule is noted in Column 4 of Table 3. All the values in Column 4 are found to be very high. So, the high value of mutual information between patterns generated by GMACA ensures that the GMACA capable of performing pattern recognition tasks are at the edge of chaos.

The studies performed in this section point that the GMACA evolved for pattern recognition, belong to the same class and are to be found at the edge of chaos.

6. Conclusions

This paper has established the Cellular Automata as a powerful machine in designing the pattern recognition tool. The pattern recognizer is designed with the hybrid GMACA based associative memory. It is proved experimentally that the storage capacity of GMACA based associative memory is far better than that of Hopfield Net and can train more number of patterns. A new parameter $\lambda_{av}$ extends
the Langton’s observation in the domain of hybrid CA. The consistent values of different parameters of the GMACA rule space ensure that the GMACA rules which can perform complex computation of pattern recognition are most likely to have occurred at the edge of chaos.

References

Niloy Ganguly received the B.Tech. (Hons) degree in computer science from IIT, Kharagpur, in 1992 and Masters in computer science and engineering from Jadavpur University, India, in 1995. He was in the Faculty of Computer Science and Engineering in Haldia Institute of Technology, India from 1997 to 1998 and R. E. College, Durgapur, India, from 1998 to 1999. Presently, he is a Lecturer in the MBA Department, Indian Institute of Social Welfare and Business Management, Calcutta, India.

His research interests include cellular automata, pattern recognition, data mining, soft computing, artificial intelligence, digital system design and test.

Biplab K. Sikdar received the B.Sc. (Hons) degree in physics from Presidency College, Calcutta University, in 1985 and B.Tech. and M.Tech. degrees in computer science and engineering from Calcutta University, India, in 1988 and 1990, respectively. He was in the Faculty of Computer Science and Engineering in North Eastern Regional Institute of Science and Technology, India from 1991 to 1992 and at the University of North Bengal, India, from 1992 to 1997. Presently, he is a Lecturer in the Department of Computer Science and Technology, Bengal Engineering College (DU), West Bengal, India.

His research interests include digital system design and test, design for testability, built-in self-test and VLSI algorithms. He has been working on the development of hierarchical cellular automata for testing VLSI circuits.

Pradipta Maji received the B.Sc. (Hons) degree in physics and M.Sc. degree in electronics science from Jadavpur University, India, in 1998 and 2000, respectively. Currently, he is a Research Scholar in the Department of Computer Science and Technology, Bengal Engineering College (DU), West Bengal, India.

His research interests include cellular automata, pattern recognition, data mining, soft computing and artificial intelligence.
P. Pal Chaudhuri graduated in 1963, and was associated with IBM World Trade Corporation in various capacities until 1975. Subsequently, he switched over to academia and started his career at Indian Institute of Technology (IIT), Kharagpur. He took the leading role to initiate the Computer Science and Engineering program at IIT Kharagpur in the late 1970s. During the period of 1986 to 1988, he was the Head/Chairman of the IIT Computer Science and Engineering Department. In 1991 and 1992, he was a Visiting Professor at the University of Illinois, Urbana-Champaign, for one semester and in the subsequent semester he worked for Cadence Design Systems (India) as a Technical Advisor. During the early 1980s, he initiated a full scale research thrust and established an excellent research base at IIT Kharagpur in the field of VLSI design and testing. Since late 1980s, he has pioneered the study of the homogeneous structure of Cellular Automata (CA) and developed matrix algebraic tools in GF(2) to completely characterize the autonomous structure of CA machines. He subsequently applied this theory to develop a wide variety of applications in diverse fields. He has published more than 100 research papers in international journals and conference proceedings. He authored the books *Additive Cellular Automata Theory and Applications, Volume 1* (IEEE, 1997) and *Computer Organization and Design* (Prentice Hall), a textbook for undergraduate and graduate level courses. In November 1996, he was invited by Intel Corporation, USA, as a Visiting Faculty. He worked at Intel Research Labs, Portland, OR, until the end of 1997. Upon his return, he joined his Alma Mater, Bengal Engineering College.

His research group there has been developing the theory of GF(2^p) CA and its applications in a wide variety of fields such as data compression, data security, pattern recognition, simulation of physical systems, VLSI design, and test, etc.