

IMPLEMENTATION OF CELLULAR LEARNING AUTOMATA ON RECONFIGURABLE COMPUTING SYSTEMS

Morteza Saheb Zamani and Farhad Mehdipour and Mohammad Reza Meybodi

Computer Engineering Department
Amirkabir University of Technology
{szamani, mehdipour, meybodi} @ce.aut.ac.ir

Abstract

Reconfigurable computing systems (RCS) use the flexibility of programmable devices and the speed of hardware to implement high performance systems. Implementation of RCS is normally made by means of programmable devices, such as FPGAs. On the other hand, recently, cellular learning automata (CLA) have been proposed as a combination of conventional cellular automaton and learning automaton. Software simulation of CLA has shown it to be successful for solving some hard problems. However, the process on conventional computers is slow. To overcome this problem, we implemented CLA in hardware. In addition, for some applications which necessitate run time changes for parameters, the ability of run-time reconfiguration (RTR) in hardware is a solution. In this paper, the design and implementation of CLA on a reconfigurable system are presented. Experimental results show considerable speed-up gain of RCS version over the software version. Independence on CLA dimensions is another benefit of reconfigurable hardware implementation of CLA. In other words, by increasing the dimensions of CLA, the time needed for running reconfigurable CLA implemented on hardware remains constant.

Keywords: Reconfigurable systems, Cellular learning automata, Hardware accelerator.

1. Introduction

Cellular automaton(CA) was introduced as a model for analyzing complicated systems behavior [1] [13] [14]. It consists of a regular set of cells each of which has several

values. On the other hand, *learning automaton*(LA), introduced in early 1960 can update its actions in a stochastic environment for improving its performance. Each LA uses a learning algorithm and learns how to interact best with the environment. The model of cellular learning automaton (CLA) has been proposed as a combination of cellular and learning automata. CLA contains several number of cells. In each cell, an internal learning automaton determines the status or action of the cell. CLA has been used in some applications such as commerce network modeling and image processing[8] [4] [9].

The rest of the paper is organized as follows: section 2, 3 and 4 briefly explains the cellular automata, learning automata and cellular learning automata, respectively. A noise elimination approach based on cellular learning automata are presented in section 5. Finally in section 6, a reconfigurable hardware version of CLA for noise elimination and experimental results are presented.

2. Cellular Automata

Cellular automata (CA) are mathematical models for systems consisting of a large number of simple identical components with local interactions. The simple components act together to produce complicated patterns of behavior. CA can perform complex computations with high degree of efficiency and robustness. In addition, they can be used to model the behavior of complex systems in the nature.

It is called cellular, because it is made up of cells like points in a lattice or like squares in a checker board and it is called automaton, because it follows a simple digital rule. There are two problems concerning CA: forward problem and inverse problem. In the forward problem, the goal is to determine (predict) properties of a given CA rule and in the inverse problem, a description of some

properties is given and the goal is to find a rule or a set of rules with these properties.

A CA consists of a finite dimensional lattice of sites whose values (internal states) are restricted to a finite set of integers $\Phi = \{0, 1, \dots, k-1\}$. The value of each site at any time instant, n , is a function of the values of the neighboring sites at the previous time instant, $n-1$. Based on the nature of this function, CAs are classified in two main groups, namely deterministic CAs and stochastic CAs. Given a finite set Φ and an integer d , one can consider a d -dimensional lattice Z^d in which every point has a label from the set Φ . Each site, u , in the d -dimensional lattice Z^d is represented by a d -tuple (z_1, z_2, \dots, z_d) . In the deterministic cellular automata (DCA), the value of each site at any time instant, n , is determined by a deterministic mapping of the values of the neighboring sites at the previous time instant, $n-1$. In stochastic cellular automata (SCA), this mapping is stochastic. [1][10] [14]

3. Learning Automata

Learning automata (LA) is an abstract model, which performs certain actions. A probabilistic environment evaluates the action performed by LA and provides an appropriate response for it. LA uses this response and prepares the next reasonable action. Figure 1 shows the relationship between the environment and LA. In a LA, the environment E is defined as a triple, $E = \{\alpha, \beta, c\}$ where $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ is the input set, $\beta = \{\beta_1, \beta_2, \dots, \beta_m\}$ is the output set and $c = \{c_1, c_2, \dots, c_r\}$ is the set of penalty probabilities.

When β has two members, $\beta_1 = 1$ is considered as penalty and $\beta_2 = 0$ is considered as reward. c_i is a probability of unfavorable result of action α_i . The environment of LA is classified into stationary and variable structure. In a stationary environment, c_i values do not change during the learning process which in a variable environment, they can change [5] [6] [7] [11].

4. Cellular Learning Automaton

Recently, Cellular learning automata (CLA) has been introduced as a combination of conventional learning and cellular automata [12][8]. CLA is a model for systems consisting of simple elements. In this model, the behavior of each element is modified based on its experience and the behavior of neighboring cells. CLA has a regular structure of cells, containing one or more LAs. As in CAs, a local rule determines reward or penalty for performing an action.

The purpose of reward or penalty of the cells is synchronous updating of CLA to reach a certain goal.

CLA is defined as $\langle L, V, Q, \Omega, \Phi \rangle$ in which $L = \{l_1, l_2, \dots, l_n\}$ is the set of cells in a Cartesian network. $Q = \{q_1, q_2, \dots, q_k\}$ is the set of actions belonging to a cell in an automaton. $\Omega = \{x: L \rightarrow Q\} = Q^N$ is the state space and Φ is a rule used in the CLA. $V = \{v_i, i \in L\}$ is a set of cells in the neighborhood of a cell i if it has the two following properties:

- a) $i \notin v_i \quad \forall i \in L$
- b) $i \in v_j \quad \text{iff } j \in v_i \quad \forall i, j \in L$

Two common kinds of neighborhood are von Neumann and Moor (see Figure 2). The CLA shown in Figure 3 represents the general functionality of a CLA. In a CLA, Each cell first chooses one of the actions from the actions set. This action is selected randomly or based on the previous experience. The reward or penalty is given in response to this action selection, and then, the internal automaton of the cell is updated. Updating all of the automata is performed synchronously. After updating, each automaton chooses another action. The process of action selection and rewarding or penalizing continues until the CLA become stable or certain criteria are met. The CLA shown in Figure 3, uses von Neumann neighborhood. The happy cells have received reward and the sad ones, have been penalized in the previous stage.

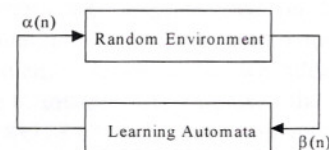
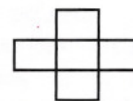


Figure 1. Relationship between LA and environment



Neighborhood Von Neumann



Moor neighborhood

Figure 2. Two kinds of neighborhood

5. Application of CLA to the Elimination of Noise in Images

CLA has been used to solve some hard problems such as image processing problems [4]. In this paper, the application of CLA to the elimination of noise in binary (black and white) images is represented. Both hardware and software implementations were investigated. Results show considerable speedup of hardware implementation as compared with software implementation. We used stationary structure learning automata ($L_{2N,2}$) with two

actions. Each cell performs two actions for black pixels and white pixels in the image. The rule used for CA is majority rule. According to this rule, if the number of neighbors of a cell and the cell itself performing an identical action, is greater than a threshold value (e.g. half of the number of cells in neighborhood), then this cell will be rewarded; otherwise it will be penalized. Some parameters, such as the neighborhood type and threshold value, can be changed during the learning process. The final status of the cells in the CLA represents the color of respective pixels in the image.

In the hardware implementation, an optimized version of a stationary environment and a majority rule-based CA were implemented. The optimization was accomplished for area as the primary objective and the performance as the secondary objective.

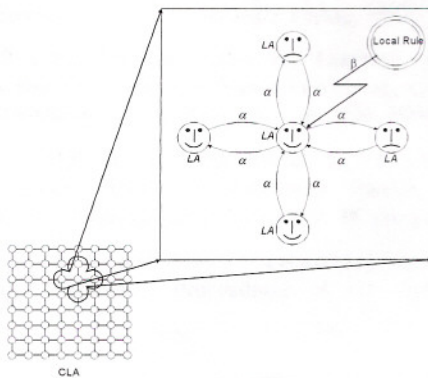


Figure 3. Cellular Learning Automaton

6. The Reconfigurable Hardware Version and the Experimental Results

Reconfigurable computing systems (RCS) use the flexibility of programmable devices and the speed of hardware to implement high performance systems. Applications with inherent parallelism; local connections between components, integer arithmetic computations and simple decisions are appropriate for implementation on RCS and CLA is one of these applications [3][2].

On the other hand, CLA has been successful in image processing problems. In addition, for some applications, there are parameters concerning cellular or learning automata, which necessitate run time changes in the hardware architecture. In such cases, the ability of run-time reconfiguration (RTR) is a solution.

RCSs are normally implemented in programmable devices, such as FPGAs(Field Programmable Gate Arrays). To be able to change the configuration of the device during its operation, RAM-based FPGAs must be used.

A modified algorithm of noise reduction in black and white images uses variable threshold value and variant neighborhood of cells during learning process of CLA. We used a hardware reconfigurable version for the implementation of this kind of CLA. In this implementation, some configurations were provided as before they are used during learning process of CLA. Optimized versions of configurations for loading on FPGA were generated.

One of the problems in the hardware implementation of a CLA on an RCS is the large number of I/O pin required for representing each status of CLA cells. This is , each cell should have one output, to show that the corresponding pixel is white or black. The CLA we used for noise elimination had at least 128x128 cells in a two dimensional array. Therefore, the number of CLA outputs exceeded the number of available I/O pins of conventional FPGA devices. We designed the CLA cells in such a way that the outputs are saved in internal bits and the outputs are needed to be observed, a shifting process is performed by setting the observe input of cell(Figure 4). By clocking the CLA when observe input is enable, the output of the cells will be observable. For an $N \times N$ CLA, N clock periods are required to observe all outputs. This approach increases area overhead of the design but reduces the required I/O pins.

The algorithm was implemented on VIRTEX from Xilinx. The running time was measured for software and hardware implementations. First, we added at most 30 percent noise to images. Then we used the reconfigurable version of 128x128 CLA for the elimination of this noise. Experiments showed that the hardware version had considerable speedup(at least 3000 times) compared to software implementation(see Table 1). Another important advantage of reconfigurable hardware CLA is that due to the inherent parallelism of CLA, increasing the dimensions of CLA does not change the time needed for running it. In the other words, the running time is independent of the size of CLA. Figure 5 shows the noise reduction from the image which has the noise.

Table 1. Software and Hardware running time comparisons

Image No.	Software running time(sec)	Hardware running time(μsec)	speedup
1	0.8	210	3809
2	0.75	196	3826
3	0.87	210	4142
4	0.8	182	4395
5	0.06	14	4285
6	0.74	154	4805
7	0.13	42	3095

References

- [1] Adamatzky, A., "Identification of Cellular Automata", Taylor & Francis Ltd., 1994.
- [2] K. Compton, S. Hauck, "Reconfigurable Computing: A Survey of Systems and Software", ACM Computing Surveys(CSUR), vol. 34, Issue 2, June 2002.
- [3] S. Hauck, A. Agrawal, "Software Technologies for Reconfigurable Systems", Northwestern University Department Of Electrical and Computer Engineering, Technical report, 1996.
- [4] Kharazmi, M. R. and Meybodi, M. R., "Application of Cellular Learning Automata to Image Segmentation", Proceeding of Tenth Conference on Electrical Engineering (10th ICEE), University of Tabriz, Vol. 1, pp. 298-306, 2001.
- [5] Lakshmivarahan, S., "Learning Algorithms: Theory and Applications", New York, Springer Verlag, 1981.
- [6] Mars, P., Chen, J.R. and Nambir, R., "Learning Algorithms: Theory and Applications in Signal Processing, Control and Communications", CRC Press, Inc., pp. 5-24, 1996.
- [7] Meybodi, M.R. and Lakshmivarahan, S., " ϵ - Optimality of a General Class of Absorbing Barrier Learning Algorithms", Information Sciences, Vol. 28, pp. 1-20, 1982.
- [8] Meybodi, M. R. and Beygi, H.Taherkhani, M., "Cellular Learning Automata", Proceedings of 6th Annual CSI Computer Conference, Computer Engineering Department, University of Isfahan, pp. 153-156, 20-22 Feb 2001.
- [9] Meybodi, M. R. and Khojasteh, M. R., "Cellular Learning Automata as a Model for Commerce Networks", Proceedings of 6th Annual CSI Computer Conference, CE Department, University of Isfahan, pp. 284 -295, 20-22 Feb. 2001.
- [10] Mitchell, M., "Computation in Cellular Automata: A Selected Review", Technical Report, Santa Fe Institute, Santa Fe, U.S.A., 1996.
- [11] Narendra, K.S. and Thathachar, M.A.L., "Learning Automata: An Introduction", Prentice Hall, Inc., 1989.
- [12] Taherkhani, M., "Proposing and Studying of Cellular Learning Automata as a Tool for Modeling Systems", M.Sc. Thesis, Computer Eng. Dept., Amirkabir University of Technology, Tehran, Iran, 2000.
- [13] Toffoli, T. and Margolus, N., "Cellular Automata Machines", the MIT Press, 1987.
- [14] Wolfram, S., "Statistical Mechanics of Cellular Automata", Review of Modern Physics, 55, pp. 601-644, 1983.

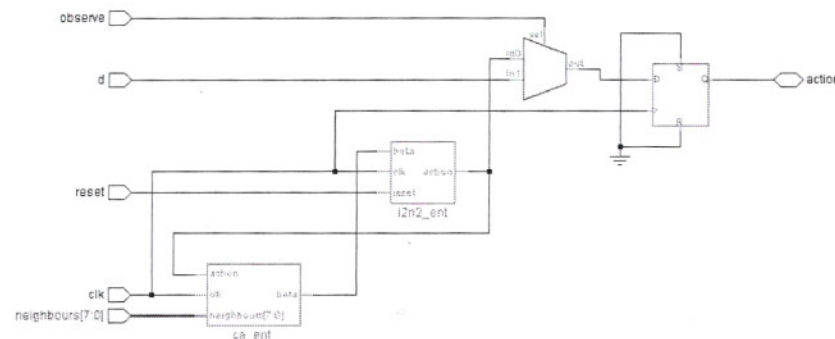


Figure 4. An implemented Cell of CLA

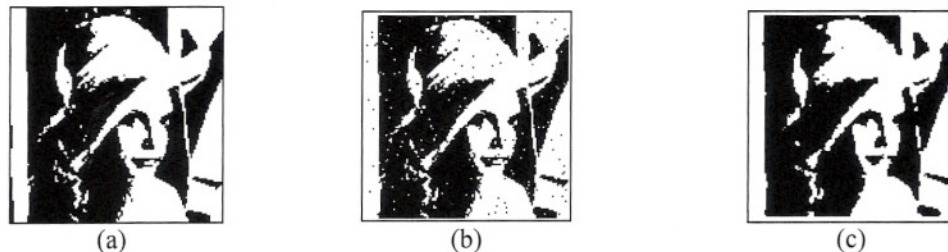


Figure 5. Elimination of noise in Black & White image (a) primary image (b)noisy image (c) noise eliminated image