# Mobile Sensor Network Deployment Using Cellular Learning Automata Approach

M. Kalantary

Computer Engineering and Information Technology Department,

Islamic Azad University, Qazvin, Iran

mary.kalantari@gmail.com

M. R. Meybodi[2]

Computer Engineering and Information Technology Department,

Amirkabir University of Technology, Tehran, Iran

mmeybodi@aut.ac.ir

*Abstract*— **Deployment problem is how to deploy a number of nodes in the area of the network so that the covered area is maximized. In this paper we consider the problem of self-deployment of a mobile sensor network. Such networks with locomotion capability are able of self-deployment; i.e., starting from some random initial configuration, the nodes in the network can distribute such that the area 'covered' by the network is maximized. This paper describes an irregular cellular learning automata based deployment algorithm of mobile sensor network. The proposed algorithm first clusters the network and then let the base-stations to help the deployment process by controlling the number of nodes in their clusters. This algorithm is designed for real-time online deployment for maximum coverage of the environment. Experimental results are present to evaluate our algorithm.**

*Keywords*—**Wireless sensor network, Coverage, Deployment, Cellular Learning Automata**

## I. INTRODUCTION

Mobile sensor networks are an exciting topic of research for several years already. A mobile sensor network is composed of a distributed collection of nodes, each of which has locomotion capability in addition to sensing, communication and computation [1]. Wireless sensor networks facilitate monitoring and controlling of physical environments from remote locations with better accuracy. They have applications in a variety of fields such as environmental monitoring, military purposes and gathering sensing information in inhospitable locations [2]. Mobile sensor networks usually face two limitations. First of all, the environment is dynamic. The position of other mobile identities and the geographical features of the environment are usually dynamic. This makes offline planning of deployment by searching through the static map of the environment very inefficient and inaccurate. Second, sensor networks are very sensitive to power consumptions because they are usually designed for applications that run for a long time, e.g. a surveillance system. The embedded systems of the mobile nodes also constrain the available power reservation.

One of many important issues in wireless sensor networks is network coverage; which indicate how to deploy or arrange sensor nodes that maximizes network coverage. However, due to random deployment without human involved, e.g. by air-dropping from an aircraft, some locations might not be covered initially are never covered unless moving sensor nodes to maximize network coverage. Once the sensors are deployed on the ground, their data are transmitted back to the base station to provide the necessary situational information [3].

In this paper, we consider the problem of self-deployment of mobile sensor networks. This paper describes an online irregular cellular learning automata based deployment algorithm of mobile sensor network that is interactive with the dynamics of the environment. The proposed algorithm first clusters the network and then let the base-stations to help the deployment process by controlling the number of nodes in their clusters. The proposed controlling mechanism is based on the irregular cellular learning automata. In this method, learning automaton residing in each cell in cooperation with the learning automata residing in its neighboring cells selects for the base-station to repel or attract its members. Based on the feedbacks received from the members, each base-station gradually learns that should attract or repel members using which, the expected number of nodes in each cluster to cover entire area.

The rest of this paper is organized as follows.

Section 2, gives a brief literature overview. The problem statement is given in section 3. Learning automata, cellular learning automata and irregular cellular learning automata will be discussed in section 4. In section 5 the proposed method is presented. Simulation results are given in section 6. Section 7 is the conclusion.

## II. RELATED WORK

The deployment problem that this paper addresses is the blanket coverage problem described by Gage [4]. Gage defines three basic types of coverage: blanket coverage, where the objective is to achieve a static arrangement of nodes that maximizes the total detection area; barrier coverage, where the objective is to minimize the probability of undetected penetration through the barrier; and sweep coverage, which is more-or-less equivalent to a moving barrier. According to this taxonomy, the deployment problem described in this paper is a blanket coverage problem.

There are lots of research work [5], [6], [7], [8], [9] related to the sensor nodes placement in network topology design. Most of them focused on optimizing the location of the sensors in order to maximize their collective coverage. However only a single objective was considered in most of the research papers, other considerations such as energy consumption minimization are also of vital practical

978-1-4244-7286-4/10/$26.00 ©2010 IEEE

importance in the choice of the network deployment. Self-deployment methods using mobile nodes [8], [10] have been proposed to enhance network coverage and to extend the system lifetime via configuration of uniformly distributed node topologies from random node distributions. In [8], the authors present the virtual force algorithm (VFA) as a new approach for sensor deployment to improve the sensor field coverage after an initial random placement of sensor nodes. Howard *et al* used potential field techniques and spread the nodes over the environment by driving them with a virtual "force" [1].

## III. Problem Statement

In this section, we describe the network and mobility model used in this study and problem statement is given.

Consider a sensor network $S$ consists of $N$ mobile sensor nodes $s_1$, $s_2$, ..., $s_N$ scattered randomly throughout a rectangular $LxL$ area A. The sensing range of each sensor node is $R_s = .5 \cdot Rc$ (communication range of each node). Sensor nodes are grouped into clusters controlled by a single command node. Sensors are only capable of radio-based short-haul communication and are responsible for probing the environment to detect a target/event. Every cluster has a cluster head node that is stationary and manages sensors in the cluster. We also assume that every nodes aware of their geographical location. This awareness can be achieved using location services such as [11] and does not require a GPS receiver at every node.

In this work, we consider the following simple sensor mobility model. We assume sensors move independently of each other and with coordination among them. The movement of a sensor is characterized by its speed and direction. A sensor randomly chooses a direction $\theta \in [0, 2\pi)$ according to some distribution with probability density function $f_\theta^s(\theta)$. The speed of the sensor, Vs, is randomly chosen from a finite range $v_s \in [0, V_s^{max})$, according to a distribution density function of $f_V^s(v)$ [12].

The coverage configuration problem can be summarizing as follows.

Given N mobile nodes, how should they deploy themselves so that the resulting configuration maximizes the net sensor coverage of the network?

## IV. Cellular Learning Automata

In this section we briefly review cellular automata, learning automata, cellular learning automata and irregular cellular learning automata.

**Cellular Automata:** Cellular automata (CA) are a mathematical model for systems consisting of large number of simple identical components with local interactions. CA is a non-linear dynamical system in which space and time are discrete. It is called cellular because it is made up of cells like points in a lattice or like squares of checker boards, and it is called automata because it follows a simple rule [12]. CA performs complex computations with a high degree of efficiency and robustness. Informally, a d-dimensional CA consists of an infinite d-dimensional lattice of identical cells. Each cell can assume a state from a finite set of states. The state of each cell at any time instant is determined by a rule from states of neighboring cells at the previous time instant.

**Learning Automata:** Learning automata (LA) is an abstract model which randomly selects one action out of its finite set of actions and performs it on a random environment. Environment then evaluates the selected action and responses to the automata with a reinforcement signal. Based on the selected action, and received signal, the automata updates its internal state and selects its next action. Learning automata are classified into fixed-structure stochastic, and variablestructure stochastic. A variable-structure learning automaton is defined by the quadruple $\{\alpha, \beta, p, T\}$ in which $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ represents the action set of the automata, $\beta = \{\beta_1, \beta_2, \ldots, \beta_r\}$ represents the input set, $p = \{p_1, p_2, \ldots, p_r\}$ represents the action probability set, and finally $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ represents the learning algorithm. This automaton operates as follows. Based on the action probability set $p$, automaton randomly selects an action $\alpha_i$, and performs it on the environment. After receiving the environment's reinforcement signal, automaton updates its action probability set based on (1) for favorable responses, and (2) for unfavorable ones.

$$
\begin{aligned}
p_i(n+1) &= p_i(n) + a.(1 - p_i(n)) \\
p_j(n+1) &= p_j(n) - a.p_j(n) \qquad \forall j \quad j \neq i
\end{aligned}
\tag{1}
$$

$$
\begin{aligned}
p_i(n+1) &= (1-b).p_i(n) \\
p_j(n+1) &= \frac{b}{r-1} + (1-b)p_j(n) \qquad \forall j \quad j \neq i
\end{aligned}
\tag{2}
$$

In these equations, $a$ and $b$ are reward and penalty parameters respectively. If $a = b$, learning algorithm is called $L_{R-P}$, if $b << a$, it is called $L_{R \varepsilon P}$, and if $b = 0$, it is called $L_{R-I}$. For more information about learning automata the reader may refer to [13, 14].

**Cellular Learning Automata:** Cellular learning automata, which is a combination of cellular automata and learning automata, is a powerful mathematical model for many decentralized problems and phenomena. A CLA is a CA in which a learning automaton is assigned to every cell. At any instant of time, the action probability vector of the LA resides in a particular cell constitutes the state of that cell. Like CA, there is a rule that the CLA operates under. The local rule of CLA and the actions selected by the neighboring LAs of any particular LA determine the reinforcement signal to that LA. The neighboring LAs of any particular LA constitute the local environment of the cell in which that LA resides. The local environment of a cell is nonstationary because the action probability vectors of the neighboring LAs vary during the evolution of the CLA. The operation of a CLA could be described as follows: At the first step, the internal state of every cell is specified. This initial value may be chosen on the basis of past experience or at random. In the second step, each LA selects one of its actions based on its action probability vector and performs it on the environment. Next, the local rule of CLA determines the reinforcement signal to each LA.

Finally, each LA updates its action probability vector on the basis of the supplied reinforcement signal and the chosen action. This process continues until the desired result is obtained. A CLA is called synchronous if all LAs are activated at the same time in parallel. A CLA is called asynchronous (ACLA) if at a given time only some LAs are activated independently from each other, rather than all together in parallel. The LAs may be activated in either time-driven or step driven manner. In time-driven ACLA, each cell is assumed to have an internal clock which wakes up the LA associated to that cell while in step-driven ACLA; a cell is selected in fixed or random sequence. CLA has found many applications such as image processing [15-18], rumor diffusion [19], modelling of commerce networks [17], channel assignment in cellular networks [20] and VLSI placement [21], to mention a few. For more information about CLA the reader may refer to [22-24].

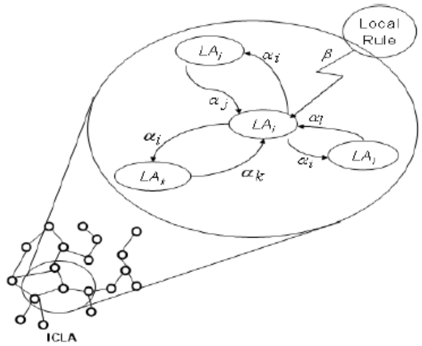

Figure 1. Irregular cellular learning automata

**Irregular Cellular Learning Automata:** An Irregular cellular learning automata (ICLA) (Figure 1) is a cellular learning automata in which the restriction of regular grid structure is removed. This generalization is expected because there are applications such as wireless sensor networks, immune network systems, graphs, etc. that cannot be adequately modelled with regular grids. An ICLA is defined as an undirected graph in which, each vertex represents a cell which is equipped with a learning automaton. Like CLA, there is a rule that the ICLA operates under. The rule of the ICLA and the actions selected by the neighboring LAs of any particular LA determine the reinforcement signal to that LA. The neighboring LAs of any particular LA constitute the local environment of the cell in which that LA resides. The local environment of a cell is non-stationary because the action probability vectors of the neighboring LAs vary during the evolution of the ICLA. The operation of ICLA is identical to the operation of CLA. Like CLA, an ICLA can be synchronous or asynchronous and an asynchronous ICLA can be time-driven or step-driven. ICLA is recently used as a learning model in a clustering algorithm for wireless sensor networks [25].

## V. Proposed Method

This paper proposes an ICLA deployment algorithm to address the issue self-deployment in mobile wireless sensor networks. The proposed method consists of two phases; clustering and controlling. In the clustering phase, a distributed clustering algorithm is used to partition the nodes of the network into a number of disjoint clusters each with a cluster head and a number of members. In the controlling phase, the cluster heads are responsible to control the number of members of their clusters by attract or repel them.

### *A. Clustering phase*

Clustering phase is a preprocessing step to the proposed algorithm and hence any clustering algorithm which considers a network of homogeneous nodes and partitions the network into disjoint clusters can be used in this phase. Specifically we consider HEED (Hybrid Energy-Efficient Distributed clustering), that periodically selects cluster heads according to a hybrid of the node residual energy and a secondary parameter, such as node proximity to its neighbors or node degree [26].

### *B. Controlling phase*

This algorithm operates in rounds. In this method, each cluster head node corresponds to the cell $c_k$ in ICLA. The learning automaton in each cell $c_k$ of ICLA, referred to as $LA_k$, has two actions $\alpha_0$ and $\alpha_1$. Action $\alpha_0$ is "repel its members" and action $\alpha_1$ is "attract its members". The probability of selecting each of these actions is initially set to 0.5.

Each iteration of the algorithm comprises of the following phases.

1. Each cluster head, chooses one of its actions using its action probability vector and send action message to its member.
2. Each member start to move according to mobility model as receive repel action, and start to move in direction of cluster head as receive attract action. Then, each sensor node sends its new location to its new or previous cluster head.
3. Receiving all messages, each cluster head calculates $SUM_k(n)$ as total number of messages and $E_k(n)$ as the cumulative number of active nodes in the cluster k up to the *n*th iteration, i.e.

$$E_k(n) = \sum_{i=1}^{n} SUM_k(n) \tag{3}$$

4. Each cluster head sends these calculations to neighbor cluster heads. After receiving all calculation messages from neighbor cluster heads, each cluster head calculates average number of nodes for neighbor cells:

$$E(S_h) = \frac{\sum_{i=1}^{h} SUM_i(n)}{I \times n \ \times h} \tag{4}$$

Where h is number of neighbor cells, $SUM(S_i)$ is number of nodes for *i*th neighbor in iteration I.

5. Each cluster head calculates $\gamma_1 - E_k(n)$ and $\gamma_2 - E(s_h)$ where $\gamma_1$ and $\gamma_2$ are predefined threshold. If these equations were positive, then the number of cells nodes is less than the desired number, and If these equations were negative, then the number of cells nodes is more than the desired number.

6. $LA_k$ updates its action probability vector based on the selected action and according to the following learning algorithm:

- if cluster head selects repel action and $\gamma_1 - E_k(n) < 0$:

  If $\gamma_2 - E(s_h) < 0$ the selected action is rewarded by (6). Otherwise, is rewarded by (5).
- if cluster head selects repel action and $\gamma_1 - E_k(n) > 0$ then, the selected action is penalized by (7).
- If cluster head selects attract action and $\gamma_1 - E_k(n) < 0$ then, the selected action is penalized by (7).
- If cluster head selects attract action and $\gamma_1 - E_k(n) > 0$:

  If $\gamma_2 - E(s_h) < 0$ the selected action is rewarded by (5). Otherwise, is rewarded by (6).

$$\begin{aligned} p_i(n+1) &= p_i(n) + (\lambda_2 - E(S_h)).(1 - p_i(n)) \\ p_j(n+1) &= p_j(n) - (\lambda_2 - E(S_h)).p_j(n) \qquad \forall j \;\; j \neq i \end{aligned} \tag{5}$$

$$\begin{aligned} p_i(n+1) &= p_i(n) + (\lambda_1 - E(S_k)).(1 - p_i(n)) \\ p_j(n+1) &= p_j(n) - (\lambda_1 - E(S_k)).p_j(n) \qquad \forall j \;\; j \neq i \end{aligned} \tag{6}$$

$$\begin{aligned} p_i(n+1) &= (1 - (-(\lambda_2 - E(S_h))).p_i(n) \\ p_j(n+1) &= (1 - p_j) \cdot (-(\lambda_2 - E(S_h))) + p_j \qquad \forall j \;\; j \neq i \end{aligned} \tag{7}$$

Using the proposed method, each cluster head gradually learns whether to attract or repel its members and algorithm terminates when the coverage reaches the maximum coverage.

## VI. Simulation Study

To evaluate the performance of the proposed method several experiments have been conducted. All simulations have been implemented using GloMoSim simulator.

The simulation environment is a 200(m) x 200(m) area through which 100 sensor nodes are scattered randomly. We assumed that each node carried a laser-range finder and an omni-camera with maximum range at four meters. The maximum speed of the nodes was 0.5m/s. We also measured the power consumption on locomotion and sensor operation in terms of distance travelled and the number of times the sensor operated. We assumed that the each sensor operation consumed 1J and the motors consume 0.5J per meter.

**Experiment 1:** This experiment is designed to find the best values for $\gamma_1$ and $\gamma_2$. We used different values for $\gamma_2$ and ran the simulation for 300s. As it is shown in figure 2, with $\gamma_2 = 0.5$, 0.6 and 0.7, we observed 100% coverage. However, as we show in figure 3, $\gamma_2 = 0.7$ converges faster than other values.

To evaluate the performance of the proposed algorithm for different values of $\gamma_1$, one cell is randomly selected and we fixed the value of $\gamma_2 = 0.7$. As it is shown in figure 4, network coverage is 100% for $\gamma_1 = 0.4$.

Figure 4 shows that these values are adequate for different network sizes.

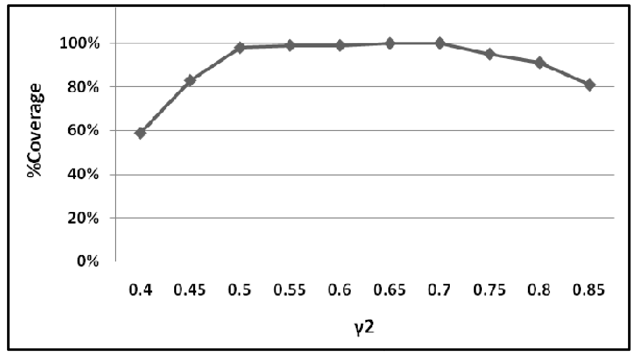


Figure 2. Simulation results for different $\gamma_2$

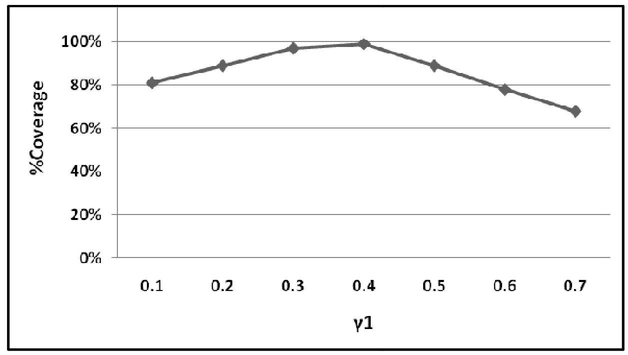


Figure 3. Coverage time for different $\gamma_2$

**Experiment 2:** In this experiment, the proposed algorithm is compared with Genetic Algorithm and Potential Field in terms of time to reach 100% coverage. Figure 5 shows that proposed algorithm is faster than others, because ICLA is easy to implement and has few parameters to adjust.

## VII. Conclusions

We have presented a self-deployment algorithm for mobile sensor networks that is designed to maximize the coverage. We have proposed irregular cellular learning automata based algorithm that is distributed and scalable, doesn't require a prior map of nodes and adapts to changes in the environment and the network itself. The proposed algorithm first clusters the network and then let the base-stations to help the deployment process by controlling the number of nodes in their clusters. The simulation results show that this algorithm outperforms the other methods.

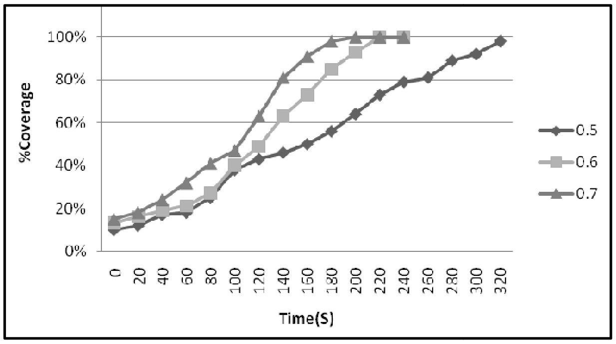


Figure 4. Simulation results for different $\gamma_1$

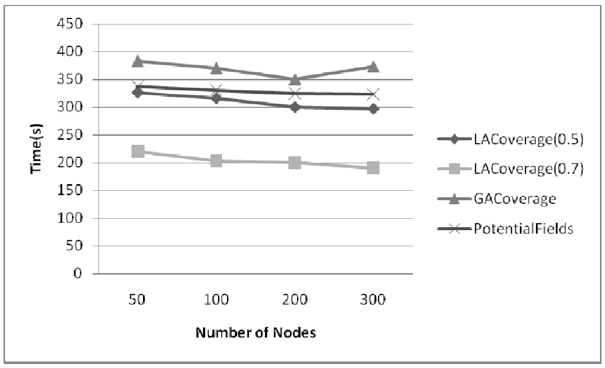


Figure 5. Time to maximum coverage

## REFERENCES


[1] A. Howard, M. Mataric, " Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem", In Proceedings of the 6th International Symposium on Distributed Autonomous Robotics Systems (DARS02) Fukuoka, Japan, June 25-27, 2002

[2] M. Ilyas, I. Mahgoub, "Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems", CRC Press, London, Washington, D.C., 2005.

[3] W. Xiaoling, et al, " Energy-efficient Deployment of Mobile Sensor Networks by PSO",

[4] D Gage, “Command control for many-robot systems,” *Unmanned Systems Magazine*, 1992, vol. 10, no 4, pp 28-34.

[5] Damien B. Jourdan, Olivier L. de Weck: Layout optimization for a wireless sensor network using a multi-objective genetic algorithm. IEEE 59th Vehicular Technology Conference (VTC 2004-Spring), Vol.5 (2004) 2466-2470

[6] K. Chakrabarty, S. S. Iyengar, H. Qi and E. Cho: Grid coverage for surveillance and target location in distributed sensor networks. IEEE transactions on computers, Vol.51 (2002)1448-1453

[7] A. Howard, M.J. Mataric and G. S. Sukhatme: Mobile sensor network deployment using potential fields: a distributed, scalable solution to the area coverage problem. Proc. Int. Conf. on distributed Autonomous Robotic Systems (2002) 299-308

[8] Y. Zou and K. Chakrabarty: Sensor deployment and target localization based on virtual forces. Proc. IEEE Infocom Conference, Vol. 2 (2003) 1293-1303

[9] B Archana Sekhar, B. S. Manoj and C. Siva Ram Murthy: Dynamic Coverage Maintenance Algorithms for Sensor Networks with Limited Mobility. Proc. PerCom (2005) 51-60

[10] Nojeong Heo and Pramod K. Varshney: Energy-Efficient Deployment of Intelligent Mobile Sensor Networks. IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems And Humans, Vol. 35, No. 1 (2005) 78 - 92.

[11] S. Tilak, V. Kolar, N. B. Abu-Ghazaleh and K. Kang, “Dynamic localization protocols for mobile sensor networks,” in: Proceedings of the IEEE International Workshop on Strategies for Energy Efficiency in Ad-hoc and Sensor Networks, April 2005.

[12] B. Liu, et al , " Mobility Improves Coverage of Sensor Networks", MobiHoc’05, May 25–27, 2005, UrbanaChampaign,Illinois, USA.

[13] M. A. L. Thathachar, P. S. Sastry, "Varieties of Learning Automata: An Overview", IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 32, No. 6, pp. 711-722, 2002.

[14] K. S. Narendra, M. A. L. Thathachar, "Learning Automata: An Introduction", Prentice-Hall Inc, 1989.

[15] M. R. Kharazmi, M. R. Meybodi, "Image Segmentation Using Cellular Learning Automata", *in Proc. of 10th Iranian Conf. on Electrical Engineering, ICEE-96,* Tabriz, Iran, May 2001.

[16] M. R. Kharazmi, M. R. Meybodi, "Image Restoration Using Cellular Learning Automata", *in Proc. of 2nd Iranian Conf. on Machine Vision, Image Processing and Applications,* Tehran, Iran, pp. 261-270, 2003.

[17] M. R. Meybodi, H. Beygi, M. Taherkhani, "Cellular Learning Automata and its Applications", *Journal of Science Tech.*, Sharif (Sharif University of Technology, Tehran, Iran), pp. 54-77, 2004.

[18] M. R. Meybodi, M. R. Khojaste, "Application of Cellular Learning Automata in Modeling of Commerce Networks", *in Proc. of 6th Annual Intl. Computer Society of Iran Computer Conf., CSICC-2001,* Isfehan, Iran, pp. 284-295, February 2001.

[19] M. R. Meybodi, M. Taherkhani, "Application of Cellular Learning Automata in Modeling of Rumor Diffusion", *in Proc. of 9th Conf. on Electrical Engineering,* Power and Water Institute of Technology, Tehran, Iran, pp. 102-110, May 2001.

[20] H. Beigy, M. R. Meybodi, "A Self-Organizing Channel Assignment Algorithm: A Cellular Learning Automata Approach", *Springer-Verlag Lecture Notes in Computer Science,* Vol. 2690, pp. 119-126, 2003.

[21] H. Beigy, M. R. Meybodi, "Call Admission in Cellular Networks: A Learning Automata Approach", *Springer-Verlag Lecture Notes in Computer Science,* Vol. 2510, pp. 450-457, 2002.

[22] H. Beigy, M. R. Meybodi, "A Mathematical Framework for Cellular Learning Automata", *Advances in Complex Systems,* Vol. 7, Nos. 3 & 4, pp. 295-319, September & December 2004.

[23] H. Beigy, M. R. Meybodi, "Open Synchronous Cellular Learning Automata", *Advances in Complex Systems,* Vol. 10, No. 4, pp. 1-30, December 2007.

[24] H. Beigy, M. R. Meybodi, "Asynchronous Cellular Learning Automata", *Automatica,* Vol. 44, No. 5, *pp.* 1350-1357, May 2008.

[25] M. Esnaashari and M. R. Meybodi, "A Cellular Learning Automata based Clustering Algorithm for Wireless Sensor Networks", *Sensor Letters,* Vol. 6, No. 5, pp. 723-735, December 2008.

[26] O. Younis and S.Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," in: Proceedings of the IEEE INFOCOM, Vol. 1, pp. 629-640, March 2004.