International Journal of Computational Intelligence and Applications
Vol. 8, No. 3 (2009) 1–28

Imperial College Press
www.icpress.co.uk

CELLULAR LEARNING AUTOMATA BASED DYNAMIC CHANNEL ASSIGNMENT ALGORITHMS

HAMID BEIGY*

Computer Engineering Department
Sharif University of Technology, Tehran, Iran

Institute for Studies in Theoretical Physics and Mathematics (IPM)
School of Computer Science, Tehran, Iran
beigy@sharif.edu

M. R. MEYBODI

Computer Engineering Department
Amirkabir University of Technology, Tehran, Iran

Institute for Studies in Theoretical Physics and Mathematics (IPM)
School of Computer Science, Tehran, Iran
mmeybodi@aut.ac.ir

A solution to channel assignment problem in cellular networks is self-organizing channel assignment algorithm with distributed control. In this paper, we propose three cellular learning automata based dynamic channel assignment algorithms. In the first two algorithms, no information about the status of of channels in the whole network will be used by cells for channel assignment whereas in the third algorithm, the additional information regarding status of channels may be gathered and then used by cells in order to allocate channels. The simulation results show that by using the proposed channel assignment algorithms the micro-cellular network can self-organize itself. The simulation results also show that the additional information used by the third algorithm help the cellular learning automata to find an assignment which results in lower blocking probability of calls for the network.

Keywords:

1. Introduction

With increasing popularity of mobile computing, demand for channels is on the rise. Since the number of channels allocated to the cellular networks is limited, efficient management and sharing of channels among numerous users become an important issue. The limited number of channels means that channels have to be reused as much as possible in order to support the many thousands of simultaneously calls that may arise in any typical mobile communication environment. Since channels are scarce and expensive resources in cellular networks, then efficient assignment

*This research was in part supported by a grant from Institute for Studies in Theoretical Physics and Mathematics (IPM), Tehran, Iran.

of channels to support communication sessions is of vital importance. The problem of assignment of channels to communication sessions is known as the *channel assignment problem* and the algorithm, which assigns channels to communication sessions, is called *channel assignment algorithm*. There are many schemes reported in the literature for assigning channels to communication sessions. These schemes can be divided into a number of different categories depending on the comparison basis. For example, when channel assignment algorithms are compared based on the manner in which co channels are separated, they can be classified as *fixed channel assignment* (FCA), *dynamic channel assignment* (DCA), and *hybrid channel assignment* (HCA) schemes.[1] In FCA schemes, a set of channels are permanently assigned to each cell, which can be reused in another cell, at sufficient distance, such that interference is tolerable.[2] In DCA schemes, there is a global pool of channels from where channels are assigned on demand and the set of channels assigned to a cell varies with time. After a call is completed, the assigned channel is returned to the global pool.[3,4] In HCA schemes, channels are divided into *fixed* and *dynamic* sets.[5] Fixed set contains a number of channels that are assigned to cells as in the FCA schemes. The fixed set of a particular cell are assigned only for calls initiated in that cell. Dynamic set is shared among all users in the network to increase flexibility. When a request for service is received by a base station, if there is a free channel in the fixed set then the base station assigns a channel from the fixed set and if all channels in the fixed set are busy, then a channel is assigned from the dynamic set.

In this paper, we present three cellular learning automata based self-organizing channel assignment algorithms. In order to show the feasibility of the proposed algorithms, computer simulations are conducted. The simulation results show that the cellular network can self-organize the assignment of channels by using the proposed channel assignment algorithms. The proposed algorithms have been compared with two existing methods proposed in Refs. 3 and 4. The simulation results show that the proposed algorithms segregate the channels among cells of the network in such a way that the blocking probability will be in an acceptable range. Even though the blocking probability of some of the proposed algorithms are slightly higher than the blocking probability for channel segregation and reinforcement learning algorithms, but they require smaller number of messages to be exchanged among the cells.

The rest of the paper is organized as follows. A brief review of channel assignment algorithms is given in Sec. 2. In Sec. 3, cellular learning automata is presented. Sections 4 and 5 present proposed algorithms and numerical results, respectively, and Sec. 6 concludes the paper.

2. Channel Assignment Algorithms

In cellular mobile networks, as shown in Fig. 1, the total service area of the network is partitioned into smaller areas, called *cells*, each with a *base station* located at its center. A number of base stations are linked to a *mobile switching center* through

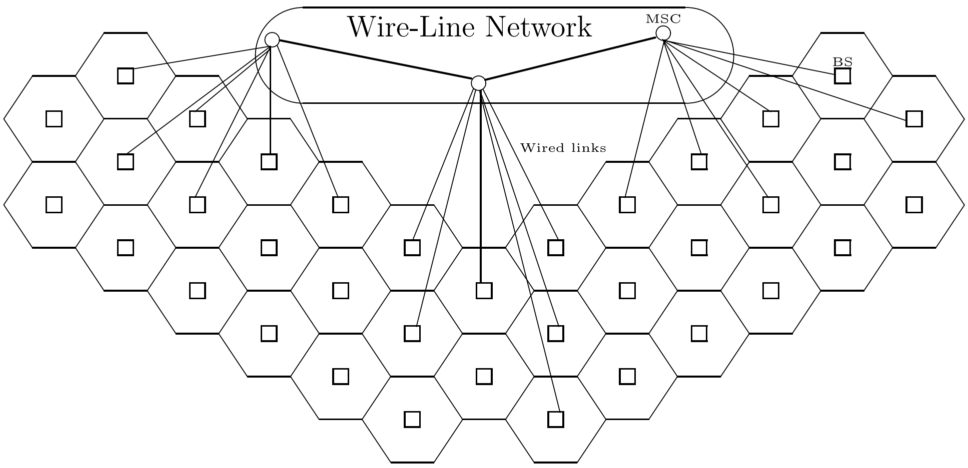

Fig. 1. Model of cellular mobile networks.

wire-line links. The mobile switching centers acts as a gateway of the mobile network to the existing wire-line networks such as *public switched telephone networks*. Mobile users can communicate with other users only by employing the wireless communication via the base station of the cell which they currently occupy. The wireless communication between a mobile user and a base station is done by using a channel. Channels are assigned to mobile users within each cell by the base station of that cell. If a particular channel is used concurrently by more than one communication session in the same cell or in the neighboring cells, the signal of communicating units will interfere with other signals. Such an interference is called *cochannel interference*. However, the same wireless channel can be used to support communicating sessions in geographically separated cells such that their signals do not interfere with each other. Such a reusability of channels is possible due to the loss of the transmitter power. This reusability is known as *frequency reuse* and corresponding cells are called *cochannel cells*. The set of all such cells which use the same channel are referred to as *cochannel sets* and the minimum distance at which channels can be reused with acceptable interference is called *cochannel reuse distance*. The set of all neighboring cells that are in the co channel interference range of each other forms a *cluster*. Thus, at any time, a channel can be used to support at most one communication session in a cluster. There are two other sources for interference: the first is, two different communication session in the same cell or neighboring cells using two adjacent channels, this interference is called *adjacent channel interference*, and the second is, two communication sessions in the same cell using different channels with some range, this interference is called *co-site interference*.

Channels are assigned using a channel assignment algorithm to the mobile users within each cell by the base station and cannot be reused within its cluster.

The channel assignment algorithms can be divided into a number of different categories depending on the comparison basis. For example, when channel assignment algorithms are compared based on the manner in which co channels are separated, they can be classified as *fixed channel assignment* (FCA), *dynamic channel assignment* (DCA), and *hybrid channel assignment* (HCA) schemes.[1] In the rest of this section we review some channel assignment algorithms proposed in the literature.

2.1. *Fixed channel assignment algorithms*

The simplest channel assignment is the fixed assignment in which a set of channels are permanently allocated to each cell, which can be reused in another cell, at sufficient distance, such that cochannel interference is tolerable. The simplest FCA strategy is allocating the same number of channels to every cell. This strategy is called *uniform channel allocation* and is useful when load has uniform distribution. In this case the probability of call blocking in a cell is the same as the call blocking probability of the cellular network.[1] The uniform channel allocation has poor channel utilization, because the traffic in cellular systems is not uniform and has spatial fluctuations. In this case it is appropriate to allocate channels to cells to match the demands. This strategy is called *nonuniform channel allocation* strategy. Fixed channel assignment problem can be defined using T-coloring problem, which is an *NP-Complete* problem.[2] Hence, channel assignment problem is classified as a NP-Complete problem, which means as the size of the problem increases, the time required to solve the channel assignment problem does not increase in a polynomial manner – but in an exponential manner. In order to find the solution for this problem, several algorithms such as frequency-exhaustive,[6] requirement-exhaustive,[6] and randomized[7] algorithms are proposed. Since fixed channel assignment problem is a NP-Complete problem, many heuristic techniques such as *tabu search*,[8] *neural networks*,[9–11] and *genetic algorithms*[12,13] have been devised for solving the channel assignment problem.

2.2. *Dynamic channel assignment algorithms*

Fixed channel assignment schemes offer negligible computation overhead (channel acquisition time) and zero communication overhead, and the details of the problem are always completely specified in advance. In the real cellular networks, the underlying set of base stations, constraints, and the number of channels are fixed but the demands must be thought of as a sequence of random variables. In these cases, FCA schemes are not able to attain high channel efficiency. This observation was motivated by some researchers to introduce DCA schemes in which there is a global pool of channels from where channels are assigned on demand. After a call is completed, the channel is returned to the global pool. In DCA schemes, a channel could be used in any cell provided that interference constraints are satisfied. An obvious dynamic channel assignment scheme is to consider recalculating the entire assignment every time the demand vector changes. This is known as a *maximum* *packing* strategy.[14] Maximum packing strategy for dynamic channel assignment in cellular networks is a greedy channel assignment algorithm which specifies that a new call attempt is admitted whenever there is some way of rearranging channels so that every call can be carried; otherwise the call is blocked. Note that there can be other dynamic channel assignment algorithms that permit reassignments but they are different from maximum packing algorithms. Maximum packing algorithms have natural appeal due to better performance in low and moderate traffic and the ability to automatically cope with changing traffic patterns. Although the practical usage of maximum packing is limited because of a possible rearrangement of calls which are in progress done on the global basis. One may think that maximum packing algorithms are optimal and give a tight bound on system performance achievable by any dynamic channel assignment algorithm. Though this is true in some cellular systems such as symmetrical cellular systems with equal loads, but in general it may be true only up to a particular load.[15] Fixed and maximum packing strategies are two extremes, but there is clearly room for a wide range in between. Dynamic channel assignment schemes can be divided into *centralized* and *distributed* schemes based on used control strategy. In centralized DCA schemes, the channels are assigned by a central controller on the basis of a cell's status information while in distributed DCA schemes, each base station allocates channels for its local users. Thus distributed schemes are more attractive for implementation in cellular systems, due to the simplicity of the assignment algorithm in each base station. Below, we briefly describe one centralized and one distributed dynamic channel assignment schemes, which will be compared with the algorithms proposed in this paper.

Singh and Bertsekas have formulated the dynamic channel assignment using dynamic programming and reinforcement learning is used for solving it.[4] The objective of this formulation is to assign channels to calls which results in minimization of the expected number of blocked calls over an infinite horizon. In their formulation, state transitions occur when a channel becomes free due to call departures or when a call arrives at a given cell or when there is handoff. In this approach, at each instant, the state of system consists of two components: the list of occupied and unoccupied channels at each cell, which is referred to as configuration of cellular system and the list of events (call arrival, call termination, or handoff) that causes the state transition. When a channel becomes free, due to the call departure in a particular cell, channels at a given cell is rearranged. The aim of this rearrangement is to create a more favorable channel packing pattern among neighboring cells.

In *channel segregation* scheme, each base station selects a channel with an acceptable cochannel interference by scanning all channels.[3] The scanning order is formed independently for each cell on the basis of a channel selection priority vector p. Each base station keeps the current value of channel selection priority $p(i)$ for channel i and selects a channel with highest priority value. In order to determine cochannel interference, the received power level of the selected channel is measured. If the measured power level is above (below) of a threshold value, the channel is determined to be busy (idle). If the selected channel is idle, the base station starts communication using that channel and its priority is increased. If the channel is busy, the priority of the channel is decreased and the next highest priority channel is tried. If all channels are busy, the call is blocked. In the channel segregation scheme, the value of the priority vector is updated in the following manner.

$$p(i) = \begin{cases} \dfrac{p(i)N(i)+1}{N(i)+1} & \text{if channel } i \text{ is idle,} \\ \dfrac{p(i)N(i)}{N(i)+1} & \text{if channel } i \text{ is busy,} \end{cases} \tag{1}$$

where $N(i)$ is the number of times channel i is selected. The simulation results show that the channel segregation strategy uses channels efficiently and decreases the number of intra-cell handoffs, that is, the reassignment of channels to avoid interference. It also decreases the load of switching system as well as quality of degradation during a handoff period.

2.3. *Hybrid channel assignment algorithms*

In hybrid channel assignment algorithms, channels are divided into *fixed* and *dynamic* sets.[5] A fixed set contains a number of channels that are assigned to cells as in the FCA schemes. The fixed channels of a particular cell are assigned only for calls initiated in that cell. Dynamic set of channels is shared between all users in the network to increase flexibility. When a request for service is received by a base station, if there is a free channel in the fixed set then the base station assigns a channel from the fixed set and if all channels in the fixed set are busy, then a channel is assigned from the dynamic set. Several hybrid channel assignment algorithms such as channel assignment with borrowing and reassignment,[16] borrowing with channel ordering,[17] directed retry,[18] borrowing with directional channel locking,[19] ordered channel assignment scheme with rearrangement (ODCA),[20] sharing with bias,[21] load balancing with selective borrowing[22] and distributed load balancing with selective borrowing scheme,[23–25] to mention a few have been proposed in the literature.

The interested readers may refer to[1,26] for more details on channel assignment algorithm.

3. Cellular Learning Automata

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

3.1. *Cellular automata*

Cellular automata (CA) are mathematical models for systems consisting of large numbers of simple identical components with local interactions. CA are non-linear dynamical systems in which space and time are discrete. They are called *cellular*, because they are made up of cell-like points in the lattice (or like squares of checker boards) and they are called *automata*, because they follow a simple rule.[27] The simple components act together to produce complicated patterns of behavior. They are specially suitable for modeling natural systems that can be described as massive collections of simple objects interacting locally with each other.[28,29] The cells update their states synchronously on discrete steps according to a local rule. The new state of each cell depends on the previous states of a set of cells, including the cell itself, and constitutes its neighborhood.[30] The state of all cells in the lattice are described by a configuration, which can be described as the state of the whole lattice. The rule and the initial configuration of the CA specifies the evolution of CA and tell how each configuration is changed in one step.

3.2. *Learning automata*

Learning in the learning automata have been studied using the paradigm of an automaton operating in an unknown random environment. In a simple form, the automaton has a finite set of actions to choose from and at each stage, its choice (action) depends upon its action probability vector ($\underline{p}$). For each action chosen by the automaton, the environment gives a reinforcement signal with a fixed unknown probability distribution. The automaton then updates its action probability vector depending upon the reinforcement signal at that stage, and evolves to the some final desired behavior. A class of learning automata, called *variable structure learning automata*, is represented by triple $\langle \beta, \alpha, T \rangle$, where $\beta = \{0, 1\}$ is a set of inputs and $\beta = 1$ means the chosen action is penalized while $\beta = 0$ means the chosen action is rewarded, $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ is a set of actions, and T is a learning algorithm.[31] The learning algorithm is a recurrence relation and is used to modify the action probability vector $\underline{p}$. Let α_i be the action chosen at time k as a sample realization from probability distribution $\underline{p}(k)$. In linear reward-ϵpenalty algorithm ($L_{R-\epsilon P}$), the action probability vector is updated according to the following rule.

$$p_j(k+1) = \begin{cases} p_j(k) + a \times [1 - p_j(k)] & \text{if } i = j \\ p_j(k) - a \times p_j(k) & \text{if } i \neq j \end{cases} \tag{2}$$

when $\beta(k) = 0$ and

$$p_j(k+1) = \begin{cases} p_j(k) \times (1 - b) & \text{if } i = j \\ \dfrac{b}{r-1} + p_j(k)(1 - b) & \text{if } i \neq j, \end{cases} \tag{3}$$

when $\beta(k) = 1$. Parameters $0 < b \ll a < 1$ represent *step lengths* and r is the number of actions for LA. Parameter $a(b)$ determines the amount of increase (decreases) of the action probabilities. The algorithm is called *linear reward penalty* (L_{R-P}) if $a = b$ and called *linear reward inaction* (L_{R-I}) if $b = 0$. Nearly all research in the area of learning automata deal with automata having a fixed action set.

In some applications, such as CPU job scheduling, learning automata with changing number of actions is needed. In such automata, the set of available actions at every instant need only be a subset of the complete set of actions and could change instantly. It has been shown that L_{R-I} algorithm with changing number of actions is both *absolutely expedient* and $\epsilon-$*optimal*.[32]

Learning automata have been used successfully in many applications such as telephone and data network routing,[33] solving NP-Complete problems,[34] capacity assignment,[35] neural network engineering[36,37] and cellular networks,[38–42] to mention a few.

3.3. *Cellular learning automata*

Cellular learning automata (CLA) is a mathematical model for dynamical complex systems that consists of a large number of simple components and introduced in Ref. 43. The simple components, which have learning capability, act together to produce complicated behavioral patterns. A CLA is a CA in which a learning automaton is assigned to every cell. The learning automaton residing in a particular cell determines its state(action) on the basis of its action probability vector. The CLA has a local rule which collectively concurs with actions selected by the neighboring learning automata of any particular learning automaton, and determines the reinforcement signal to the learning automaton, and residing in a cell. The neighboring LAs of any particular LA constitute the local environment of that cell. The local environment of a cell is nonstationary because the action probability vectors of the neighboring LAs vary during evolution of the CLA.[44] The operation of the cellular learning automata can be described as follows: At first step, the internal state of cells are specified. The state of each cell is determined on the basis of the action probability vectors of the learning automaton residing in that cell. The initial value of this state may be chosen on the basis of past experience or at random. In the second step, the rule of the cellular learning automata determines the reinforcement signal input to the learning automaton residing in the cell. Finally, each learning automaton updates its action probability vector on the basis of the supplied reinforcement signal and the action chosen by the cell. This process continues until the desired result is obtained.

The CLA can be classified into two groups: *synchronous* and *asynchronous* CLA.[45] In synchronous CLA, all cells are synchronized with a global clock and executed at the same time. The operation of the synchronous CLA take place as the following iterations. At iteration k, each LA chooses an action.[46] Then all LAs receive a reinforcement signal, which is produced by the application of the local rule. The higher value of the reinforcement signal means that the chosen action of LA will receive higher reward. In asynchronous CLA (ACLA), LAs in different cells are activated asynchronously. The operation of ACLA takes place as the following iterations. At iteration k, the activated LAs choose one of their actions. The activated automata use their current actions to execute the rule (computing the reinforcement signal). The actions of neighboring cells of an activated cell are their most recently selected actions. Finally, activated LAs update their action probability vectors and the process repeats.

The CLA can also be classified into *closed CLA* and *open CLA*.[47] In the former, the action of each cell in the next stage of its evolution only depends on the actions of its local environment (actions of its neighboring LAs) while in the latter, the action of each cell in next stage of its evolution not only depends on the local environment but it also depends on the external environments. In *open cellular learning automata* (OCLA), two types of environments are considered: global environment and exclusive environment. Each OCLA has one global environment that influences all cells and an exclusive environment for each particular cell. The operation of OCLA takes place as iterations of the following steps. At iteration k, each LA chooses one of its actions. The actions of all LAs are applied to their corresponding local environments (neighboring LAs) as well as global environments and their corresponding exclusive environments. Then all LAs receive their reinforcement signal, which is a combination of the responses from local, global and exclusive environments. These responses are combined using the local rule. Finally, all LAs update their action probability vectors based on the received reinforcement signal. Note that the local environment for each LA is nonstationary while global and exclusive environments may be stationary or nonstationary.

The CLA have been used in many applications such as image processing,[43] rumor diffusion,[48–52] modeling of commerce networks,[53] channel assignment in cellular networks,[54] VLSI Placement,[55] and sensor networks,[56] to mention a few.

4. The Proposed Dynamic Channel Assignment Algorithms

In this section, we propose three dynamic channel assignment algorithms based on asynchronous cellular learning automata. In the first two algorithms, no information about the status of channels in the whole network will be used by the cells for the channel assignment purpose whereas in the third algorithm we allow additional information regarding status of channels to be gathered and used by the cell in order to allocate channels. The additional information helps the cellular learning automata to find an assignment which results in lower blocking probability for the network.

4.1. *CLA based dynamic channel assignment Algorithm I*

In this section the first dynamic channel assignment algorithm based on CLA will be proposed. In this algorithm, a network with n cells and m channels is modeled with an ACLA with n cells, where cell v is equipped with m two-actions LA of L_{R-I} type. In each cell v, the kth LA specifies that the kth channel is to be used in this cell or not. The action set of LAs is equal to $\{0, 1\}$, where 1 means the corresponding channel is selected as a candidate channel for the assignment to the incoming call while 0 means that the corresponding channel is not selected. The operation of this algorithm can be described as follows: when a call arrives at cell u, all LAs of this cell are scanned using a sweeping strategy until an interference free channel is found or all channels are scanned. The sweeping strategy orders the LAs of a cell for scanning. The sweeping strategies used for this algorithm are: fixed sweeping, maximum usage sweeping, minimum usage sweeping, and random sweeping. Let $I_u = (i_1, i_2, \ldots, i_m)$ be the scanning order of learning automata of cell u specified by the sweeping strategy. If an interference-free channel is found, the incoming call is accepted, a channel is assigned to it, and then the selected action of the corresponding LA is rewarded; otherwise the call will be blocked. The AssignChannel(s) shown in Procedure 1 is executed by a cell upon the receipt of a call to assign a channel using a strategy s.

In what follows, we study how the proposed algorithm is mapped to an ACLA with multiple LAs in each cell. The activation probability vector of ACLA is obtained by taking expectation from product of an n-dimensional vector $\underline{\pi}_1$ and an $n \times nm$-dimensional matrix $\underline{\pi}_2$. Vector $\underline{\pi}_1$ is called *cell activation vector* and determines when a given cell is activated. It is apparent that when a call arrives to a cell i, it will be activated, i.e., $\pi_1(i) = 1$. Thus $E[\pi_1(i)] = \frac{1}{\lambda_i}$ for cells $i = 1, 2, \ldots, n$, where λ_i is the call arrival rate for cell i. Matrix $\underline{\pi}_2$ is called learning automata activation matrix and determines when an LA in an activated cell is triggered. Elements $\pi(i, j)$

Procedure 1. The channel assignment Procedure I executed by each base station.

procedure AssignChannel(s)
    order LAs using the given sweeping strategy s and put in list L.
    set $k \leftarrow 1$
    set $found \leftarrow false$
    while $k \leq m$ **and not** found **do**
        LA A_{L_k} chooses one of its actions, where A_{L_k} is the k^{th} LA in list L.
        if selected action is 1 **then**
            if selected channel doesn't interfere with channels used in neighboring cells **then**
                assign the channel and reward the action α_1 of A_{L_k}
                set $found \leftarrow true$
            end if
        end if
        Set $k \leftarrow k + 1$
    end while
    if not $found$ **then**
        block the incoming call
    end if
end procedure

becomes 1 when the kth LA in cell i is activated, where $k = j - (i-1)m$. Thus $E[\pi_2(i,j)|\pi_1(i) = 1]$ equals to the probability of triggering the kth LA in cell i (for $k = j - (i-1)m$) given a call arrives at cell i. Vector $\underline{\pi}_1$ is determined by the call arrival rate while matrix $\underline{\pi}_2$ is obtained from sweeping strategies, which some of them are described below.

Fixed sweep strategy: This strategy scans channels of a typical cell i one by one in an increasing order of their indices, i.e., $I_i = (1, 2, \ldots, m)$. Suppose that a call arrives at cell i (for $i = 1, \ldots, n$), then $\pi_1(i) = 1$ and the LAs are triggered using matrix $\underline{\pi}_2$, which is recomputed every time an LA is triggered. The re-computation of matrix $\underline{\pi}_2$ is done in the following way. At the first step $\pi_2(i, (i-1)m+1)$ is set to 1, i.e., the first LA is activated. Then the remaining elements of $\underline{\pi}_2$ are computed according to the following rule:

$$\pi_2(i,j) = \begin{cases} 1 & \text{if } \pi_1(i) = 1 \text{ and } \pi_2(i, j-1) = 1 \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

for $j = (i-1)m + 2, \ldots, im$. In other words, in this strategy, LAs in cell i are triggered sequentially in increasing order of their indices until a channel is found for the assignment or all channels are examined.

Maximum usage strategy: In this strategy, the set of LAs in cell i are triggered in decreasing order of their usage of their corresponding channels until a non-interfering channel is found. If no channel can be found, then the incoming call will be blocked. In other words, in this strategy, LA A_k is triggered in the kth stage of the activation of cell i if u_k^i is the kth largest element in usage vector $\underline{u}^i = \{u_1^i, u_2^i, \ldots, u_m^i\}$, where u_k^i is the number of times that channel k is assigned to calls in cell i.

Minimum usage strategy: In this strategy, the set of LAs in cell i are triggered in increasing order of their usage of their corresponding channels until a non-interfering channel is found. If no channel can be found, then the incoming call will be blocked. In other words, in this strategy, LA A_k is triggered in the kth stage of the activation of cell i if u_k^i is the kth smallest element in usage vector $\underline{u}^i = \{u_1^i, u_2^i, \ldots, u_m^i\}$, where u_k^i is the number of times that channel k is assigned to calls in cell i.

Random sweep strategy: In this strategy, the set of LAs in the cell are triggered in random order. First a random sequence of indices are generated randomly and then the set of LAs are triggered according to this generated order.

4.2. *CLA based dynamic channel assignment Algorithm II*

In this section we propose the second CLA based dynamic channel assignment algorithm. In this algorithm, we consider a cellular network with n cells and m full duplex and interference free channels. In order to assign channels to calls dynamically, we use an ACLA with n cells and one $L_{R-\epsilon P}$ LA in each cell. The neighborhood and the local rule for ACLA are the same as those given for Algorithm I.

Procedure 2. The channel assignment Procedure II executed by each base station using non-retry strategy.

```
procedure AssignChannel
    LA chooses one of its actions.
    if selected channel doesn't interfere with channels used in neighboring cells then
        assign the channel and reward the selected action of LA
    else
        block the incoming call and penalize reward the selected action of LA
    end if
end procedure
```

Each LA has action set $\underline{\alpha} = \{1, 2, \ldots, m\}$, where action j corresponds to channel j. In this algorithm, when the LA residing in cell i chooses action j, it means that channel j is a candidate channel for assigning to the incoming call in cell i. The following strategies can be used to examine different channels for assignment.

Non-retry strategy: In this strategy, when a call arrives at a cell, the LA associated to this cell chooses one of its actions. If the channel corresponding to the chosen action does not interfere with the channels used in neighboring cells, then the chosen channel is assigned to the incoming call and the chosen action is rewarded; otherwise the call is blocked and the chosen action is penalized. The AssignChannel shown in Procedure 2 is executed by a cell upon the receipt of a call to assign a channel.

Retry strategy: In this strategy, when a call arrives at a cell, the LA associated to this cell chooses one of its actions. If the channel corresponding to the chosen action does not interfere with the channels used in its neighboring cells, then the chosen channel is assigned to the incoming call and then it is rewarded; otherwise it is penalized. When the base station fails to assign the chosen channel, it retries in order to choose another channel. The process of retrying repeats until all m channels are tested or a non-interfering channel is found for assignment. When the base station fails to find a channel to assign, the incoming call is blocked. The AssignChannel shown in Procedure 3 is executed by a cell upon the receipt of a call to assign a channel.

4.3. *CLA based dynamic channel assignment Algorithm III*

The results of experiments conducted with Algos. I and II revealed the fact that the blocking probability attained by Algos. I and II is not the minimum attainable blocking probability. In order to alleviate this problem, the third CLA based algorithm will be proposed. In this algorithm, we allow additional information regarding channels in the network to be gathered and used by each cell in order to allocate

Procedure 3. The channel assignment Procedure II executed by each base station using retry strategy.

```
procedure AssignChannel
    set k ← 1
    set found ← false
    while k ≤ m and not found do
        LA chooses one of its actions.
        if selected channel doesn't interfere with channels used in neighboring
        cells then
            assign the channel and reward the selected action of LA
            set found ← true
        else
            penalize the selected action of LA
        end if
        Set k ← k + 1
    end while
    if not found then
        block the incoming call
    end if
end procedure
```

channels. The additional information helps CLA to find an assignment which results in lower blocking probability for the network.

In the rest of this section, we explain two strategies for exchanging additional information: search and update strategies. We call channel assignment algorithms that use search and update based strategies, *search based algorithm* and *update based algorithm*, respectively. In order to assign channels dynamically in a network with n cells and m channels, we use an ACLA with n cells and one L_{R-I} learning automaton with varying action set in each cell. Each learning automaton has action set $\underline{\alpha} = \{1, 2, \ldots, m\}$, where action j corresponds to channel j.

Search based algorithm: In a search based algorithm, when a call arrives at cell i, the following steps are taken by the base station of the cell. First, the base station queries its neighborhoods for their busy channels, if any, and then disables the actions of its learning automaton corresponding to these channels. Second, the learning automaton of cell i chooses one of its enabled actions (channels), if any. This channel is then assigned to the incoming call and the chosen action is rewarded. If the enabled action set is empty, then the incoming call will be blocked. Finally, the base station of the activated cell enables all of its actions and then waits for the arrival of the next call or message. The AssignChannel shown in Procedure 4 is executed by a cell upon the receipt of a call or a message from neighboring cells.

Procedure 4. The channel assignment Procedure III executed by each base station (Search strategy).

```
procedure AssignChannel
    if a call arrives then
        obtain the list of busy channels in the neighboring cells by sending
        message to them.
        disable actions selected by the neighboring learning automata.
        if action set of learning automaton is not empty then
            LA chooses one channel.   ▷ This channel is not used in neighboring
            cells.
            assign the channel and reward the selected action of LA
        else
            block the call
        end if
        enable all channels of LA
    end if
    if a query message is received from a neighboring cell then
        send the list of busy channels to this cell
    end if
end procedure
```

Gathering additional information can be done either using message transmission or using the interference detection hardware. When message transmission is used to gather additional information, it is possible to assign one channel to more than one neighboring cells at the same time. In order to prevent simultaneous usage of one channel by two or more neighboring base stations, a time stamp according to the Lamport's scheme[57] is attached to each request. A base station which currently searches for a channel defers the respond to any request with a higher time stamp than its own until its own request has been completed.

Update based algorithm: In this algorithm, when a call arrives at cell i, the following steps are performed by its base station. First, the learning automaton in the cell chooses one of its free channels (enabled actions), say channel j, if any, and then the base station assigns this channel to the incoming call and the chosen channel is rewarded. If the free channel set is empty, then the incoming call will be blocked. Second, when channel j is assigned to the incoming call, the base station of the cell informs the neighboring cells about this assignment. When a base station finds out that a channel is being used in a neighboring cell, it disables the action corresponding to this channel. Finally, when a call is terminated or handed off, its channel, say channel j is released. Then the base station of the cell informs the neighboring cells about this release and action j of all learning automata in

Procedure 5. The channel assignment Procedure III executed by each base station (Update strategy).

```
procedure AssignChannel
    if a call arrives at the cell then
        if action set of LA assigned to the cell is not empty then
            LA chooses one action from its enabled actions, let α_j be the selected
              action.
            assign the channel and reward the selected action of LA
            send channel j is busy to neighboring cells
        else
            block the call
        end if
    end if
    if an update message is received from one of the neighboring cells that
      channel j is assigned then
        disable the action α_j
    end if
    if an update message is received from one of the neighboring cells that
      channel j is free then
        enable the action α_j
    end if
    if a call using channel j departures from the cell then
        enable the action α_j
        send message channel j is free to neighboring cells
    end if
end procedure
```

the neighboring cells are enabled. The AssignChannel shown in Procedure 5 is executed by a cell upon the receipt of a call or a message from neighboring cells.

The distributed nature of these algorithms and the finite but nondeterministic propagation delays of messages between base stations may lead to cochannel interference. Such a possibility can be prevented as follows: having a candidate for assignment, the base station of the activated cell sends a message to its neighboring cells. Only if all base stations in the neighboring cells approve of the assignment of the candidate channel, then the channel is assigned; otherwise the base station chooses another candidate channel and the process is repeated. To do this, the algorithms may use time stamp with its request messages. However, in these algorithms the request messages are not deferred while waiting for permission to use the candidate channel from all base stations in the cluster if it receives a request from another base station for the same channel. In this case, it responds with reject if the time stamp of this request is greater than the time stamp of its request;

otherwise it responds with grant and aborts its own request. In the case of failing to acquire one channel; the base station tries to acquire another channel which is free according to the local information of the activated base station.

5. Simulation Results

In order to evaluate the proposed algorithms, several computer simulations are conducted and the results are compared with results obtained for channel segregation algorithm[3] and reinforcement learning algorithm.[4] For all of the experiments, it is assumed that there are seven base stations, which are organized in a linear array, shares 5 full duplex and interference free channels. Also the interference constraints between any pair of cells is represented by an integer, which prescribes the minimum gap that must exist between channels assigned to cells in order to avoid interference. The element $c(i,j)$ of constraint matrix C represents the interference constraint between cells i and j. Let $d(i,j)$ represents the normalized distance between the centers of cells i and j, where the distance between centers of adjacent cells are unity. $c(i,j)$ is defined as follows.

$$c(i,j) = \begin{cases} 1 & \text{if } d(i,j) \leq 2 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The elements of matrix C corresponding to pairs of non-interfering cells are defined to be zero. We assume that the arrival of calls is a Poisson process with rate λ and channel holding time of calls is exponentially distributed with mean $\mu = 1/3$. We also assume that no handoff occurs during the channel holding time. The results of experiments reported in this section are obtained from $120,000$ seconds simulations.

5.1. *Experiment 1*

This experiment is conducted to study the average blocking probability of calls for the network. Figures 2 through 4 compare the average blocking probability of calls for the proposed algorithms with that of channel segregation algorithm (CS)[3] and reinforcement learning based algorithm (RL).[4] By carefully inspecting Figs. 2 and 3, it is apparent that the blocking probability of Algos. I and II are higher than CS and RL algorithms in low traffic condition. In high traffic conditions the blocking probabilities of the Algos. I and II are closer to CS and RL algorithms. In low traffic conditions, the number of times a cell is activated is lower and hence the learning automata associated to that cell do not receive enough samples from traffic in order to accurately estimate the traffic parameters. Figure 4 shows that Algo. III has the same blocking probabilities as CS and RL algorithms at the expense of higher messaging overhead when compared to Algos. I and II. One reason for decreasing the blocking probability may be due to the exchange of status information among neighboring cells.

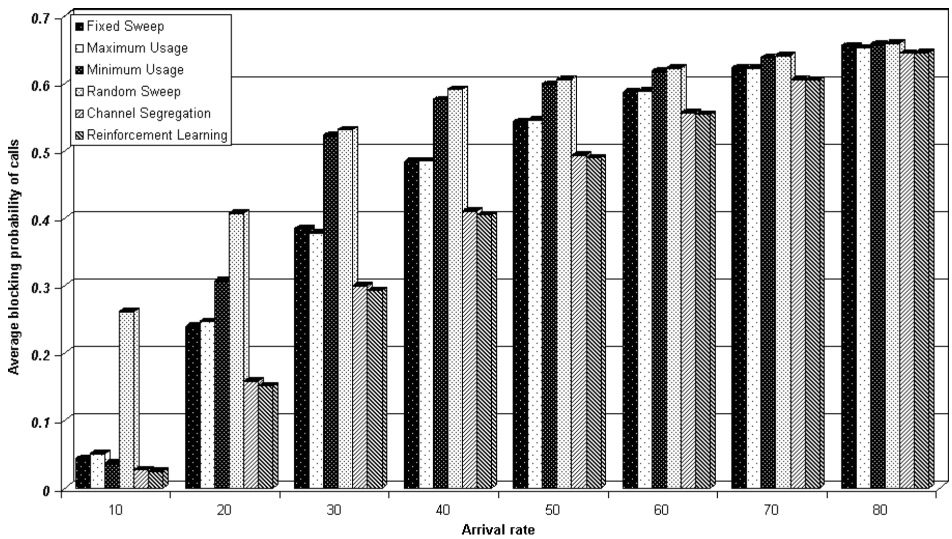

Fig. 2. Average blocking probability for Algo. I.

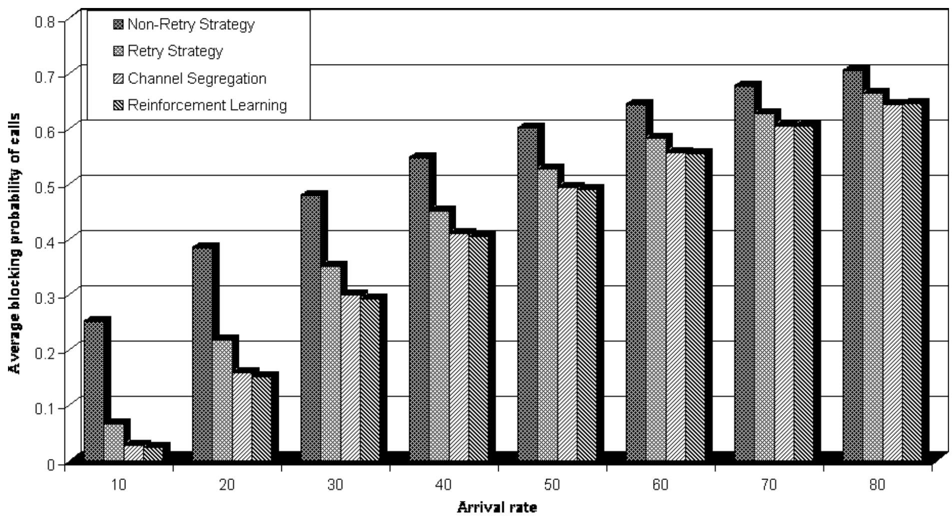

Fig. 3. Average blocking probability for Algo. II.

Figures 5 through 7 show the evolution of blocking probability of the network for a typical run for different strategies. These figures show that the blocking probabilities and interference decrease as the learning proceeds. That is, the CLA is able to better segregate channels among the cells of the network. These figures also show that the average reward of CLA is increasing for all strategies except for the *minimum usage sweeping strategy*.

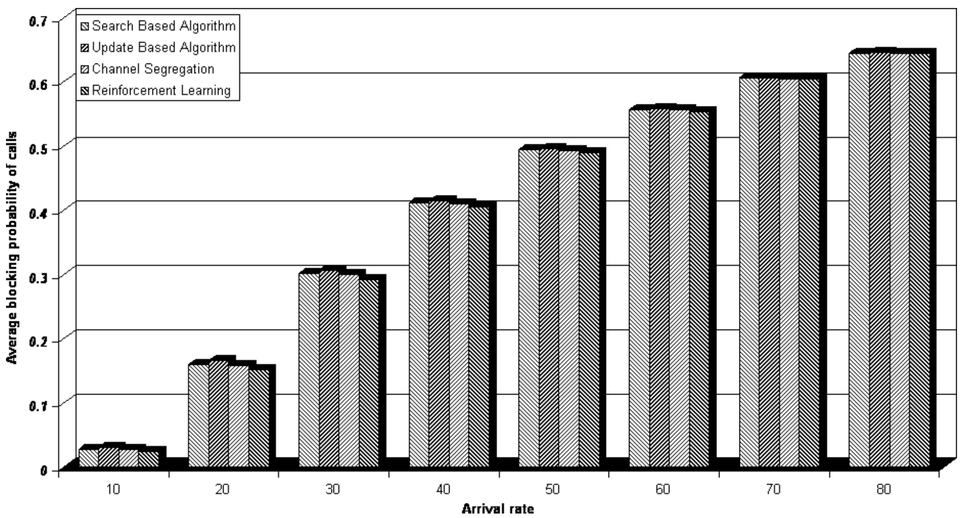

Fig. 4. Average blocking probability for Algo. III.

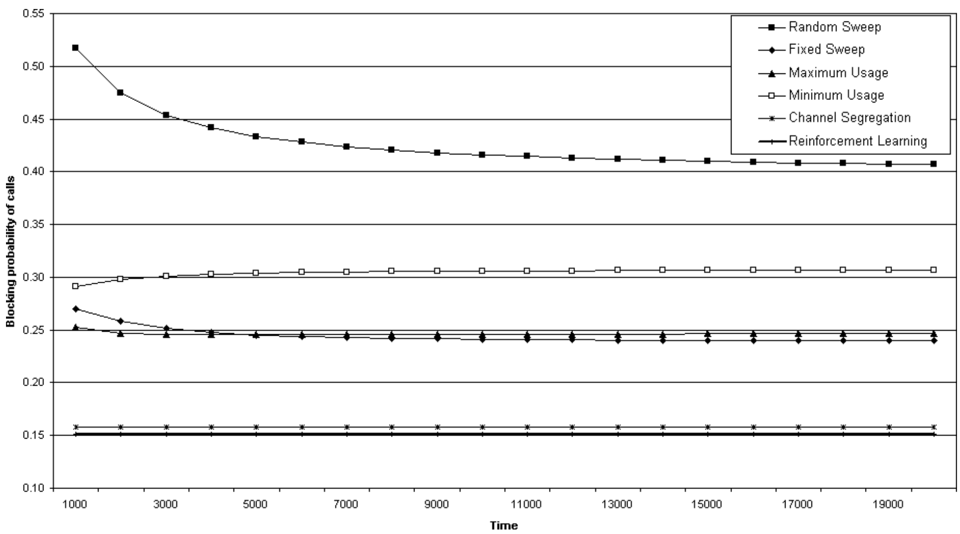

Fig. 5. Evolution of blocking probability for a typical run for Algo. I.

5.2. *Experiment 2*

This experiment is conducted to study the cochannel interference. Figures 8 through 10 show the evolution of interferences of the network for a typical run for different strategies. These figures show that the interference decrease as the learning proceeds. That is, the CLA segregates channels among the cells of the network. These figures show that the average reward of CLA is increasing for all strategies except

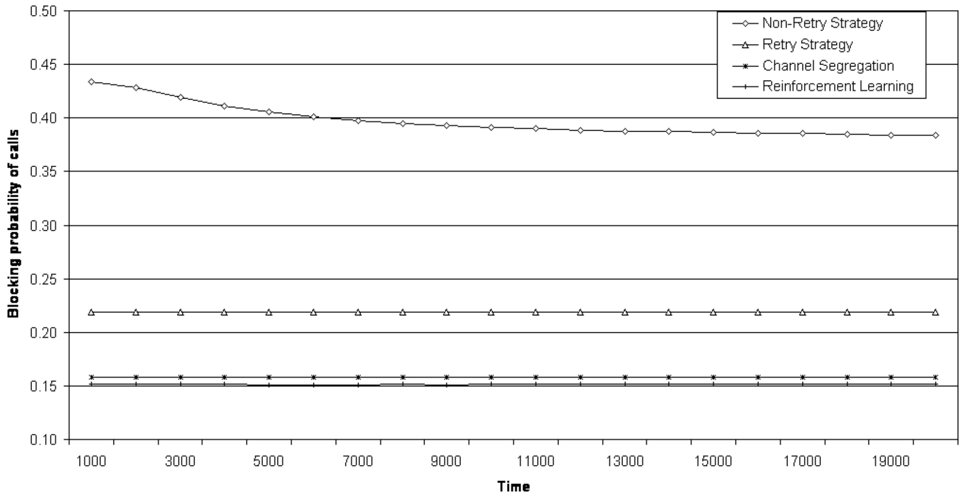

Fig. 6. Evolution of blocking probability for a typical run for Algo. II.

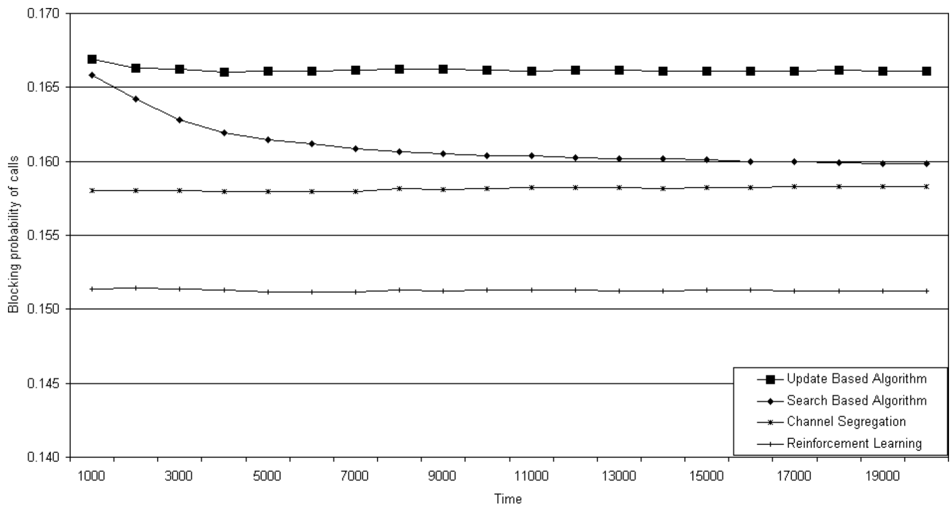

Fig. 7. Evolution of blocking probability for a typical run for Algo. III.

for the *minimum usage sweeping strategy*. For Algo. II, which uses retry strategy, and Algo. III, which uses update strategy, we do not have any interference and so they are not shown in these figures.

5.3. *Experiment 3*

This experiment is conducted to study the messaging overhead of the proposed algorithms and compare them with CS and RL algorithms. Figures 11 through 13

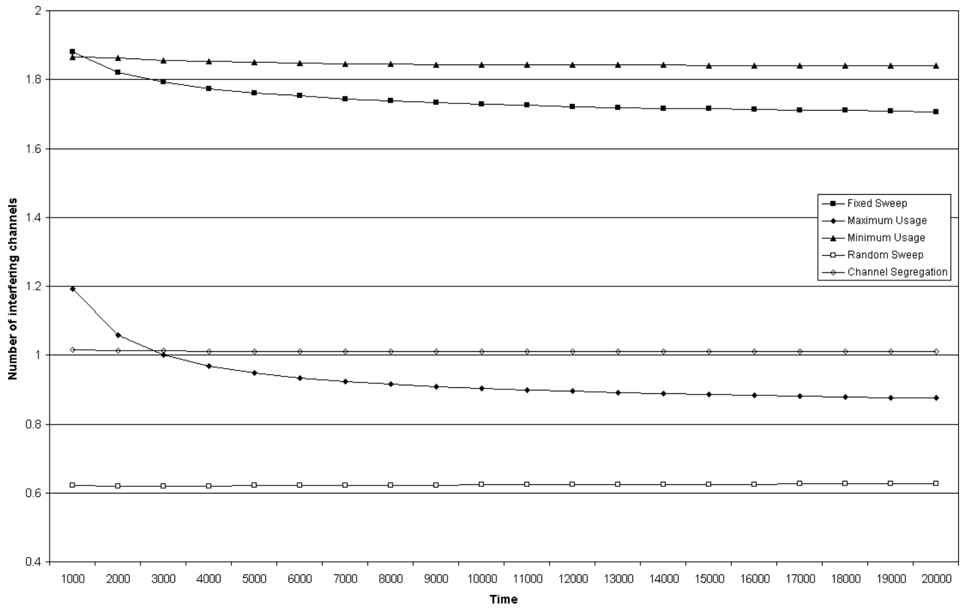

Fig. 8. Evolution of interference among channels assigned to neighboring cells for Algo. I.

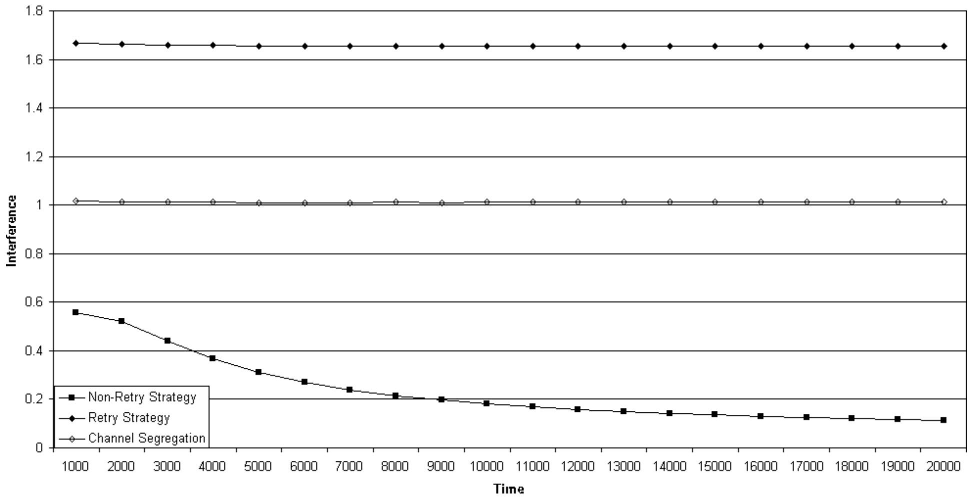

Fig. 9. Evolution of interference among channels assigned to neighboring cells for Algo. II.

show the number of messages transmitted per call among the neighboring cells. These figures also show that the proposed algorithms have lower network overheads and hence consume lower network resources, because of exchanging a small amount status information among neighboring cells.

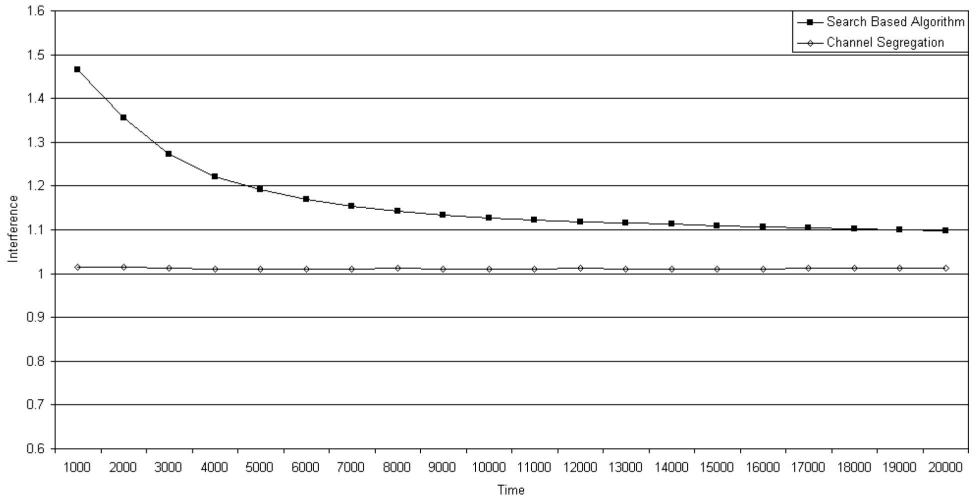

Fig. 10. Evolution of interference among channels assigned to neighboring cells for Algo. III.

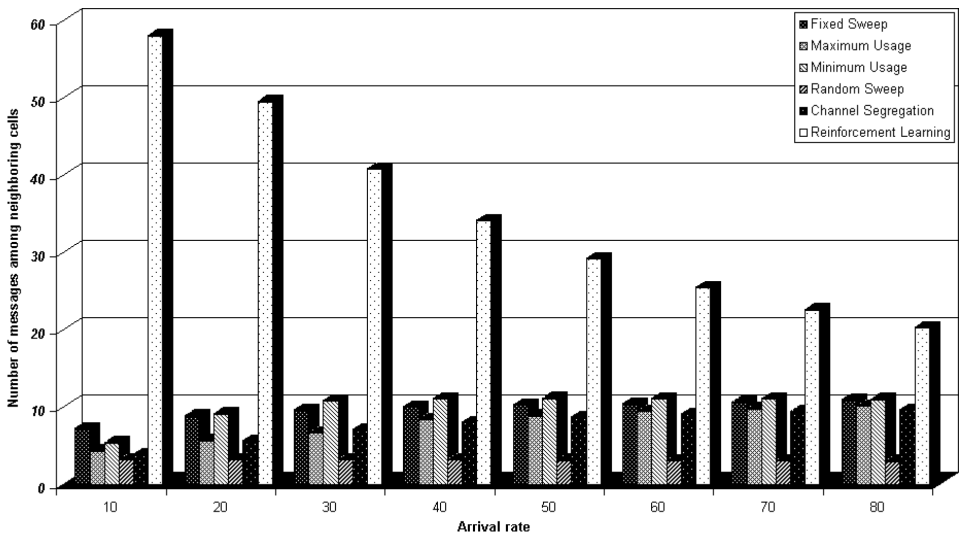

Fig. 11. Number of messages transmitted among neighboring cells for Algo. I.

Exchange of status messages waste the bandwidth of a wire-line network and also increases the response time of the channel assignment algorithm. The overhead of message exchange among the neighboring base stations can be eliminated by using interference detection hardware in each base station. By using such a hardware, each base station can determine used channels in its neighboring cells

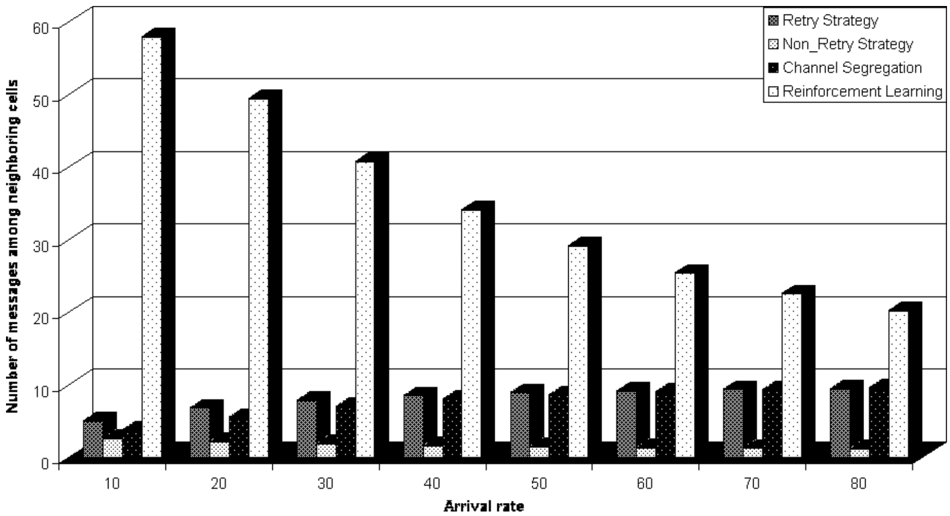

Fig. 12. Number of messages transmitted among neighboring cells for Algo. II.

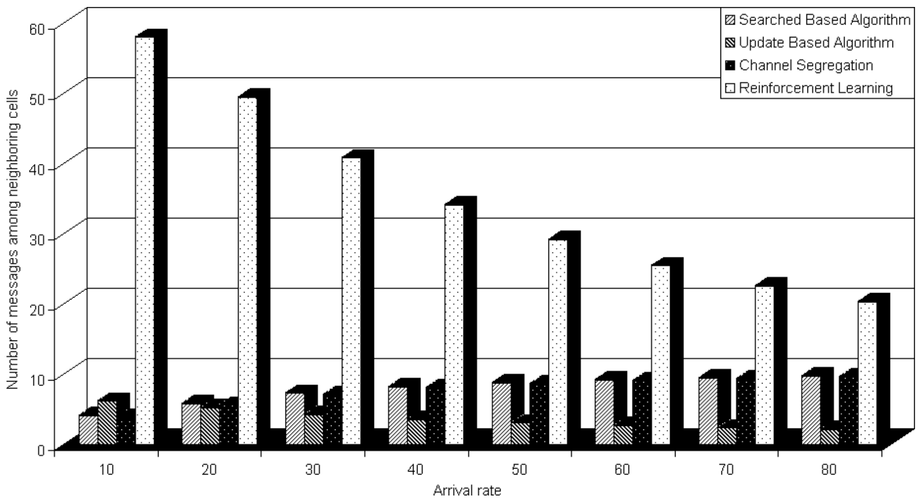

Fig. 13. Number of messages transmitted among neighboring cells for Algo. III.

without any message transmission. Using the statistical data of interference detection, each learning automaton in a given cell learns the interference map among the base stations. The cochannel interference will never happen if the results of the interference detection hardware are always correct and there are no change in the propagation conditions during channel holding time. Interference detection results, however, are not always correct because of fading phenomenon. Even, if the interference detection results are correct at the time of observation, the channel may

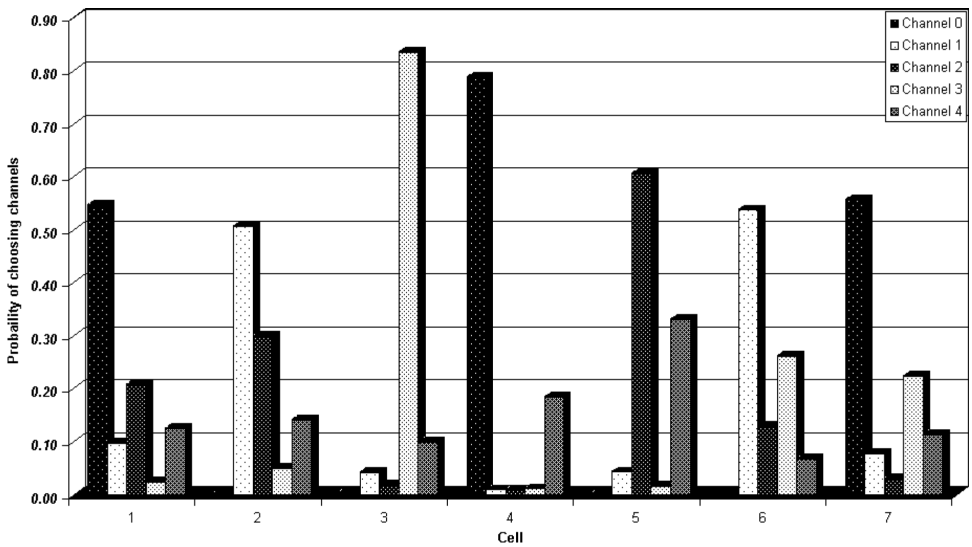

Fig. 14. Probability of assigning different channels to different cells for fixed sweep strategy.

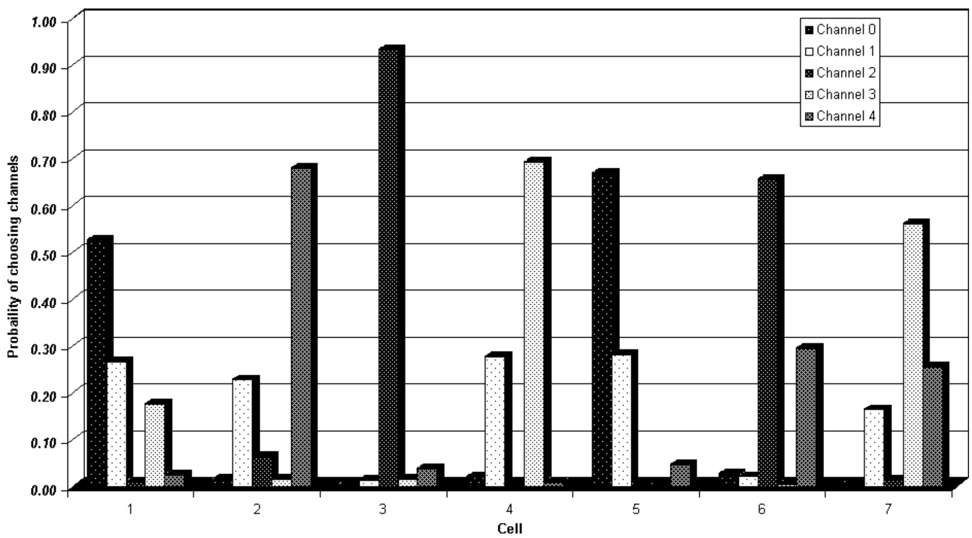

Fig. 15. Probability of assigning different channels to different cells for maximum usage strategy.

suffer interference from changes in the propagation due to the movement of the mobile user. Therefore, it is impossible to perfectly avoid the interference so far, as the channel assignment is based on the result of hardware detection hardware. There is also another source of interference, which happens when two base stations try to use the same channel at the same time since their corresponding interference

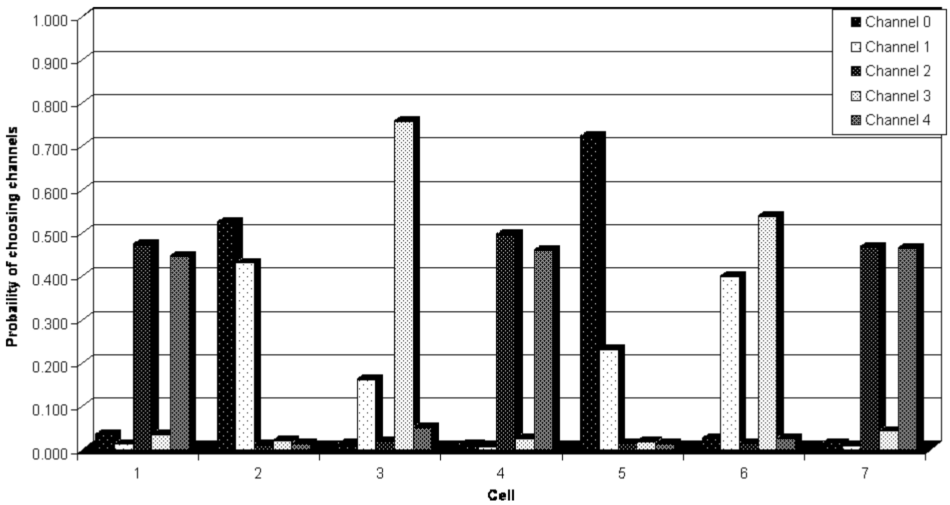

Fig. 16. Probability of assigning different channels to different cells for non-retry strategy.

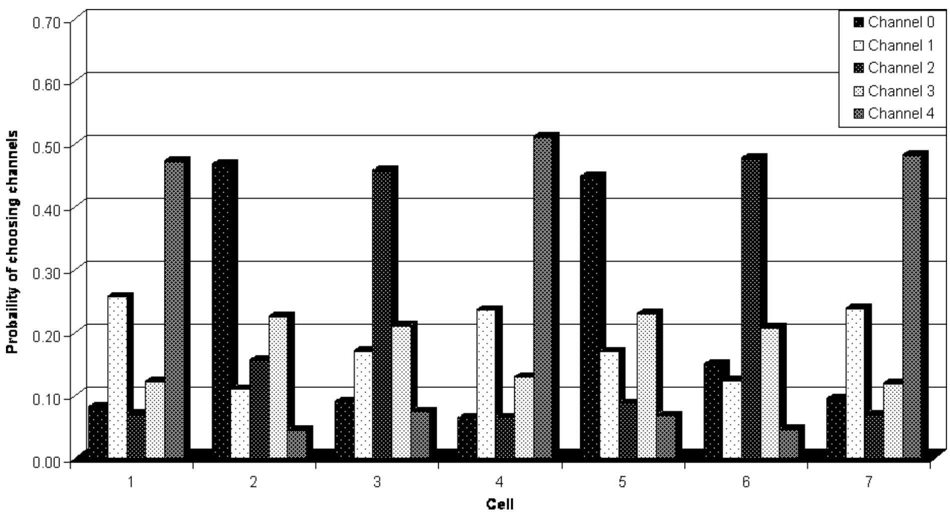

Fig. 17. Probability of assigning different channels to different cells for search based strategy.

detection results show that this channel is idle. The probability of this phenomenon decreases as the learning proceeds. This is because as the learning proceeds, the segregation of channels among the cells becomes more definite.

5.4. *Experiment 4*

The aim of this experiment is to find out which of the proposed algorithms better segregate channels to cells. To do this, we observe the final probability of assigning

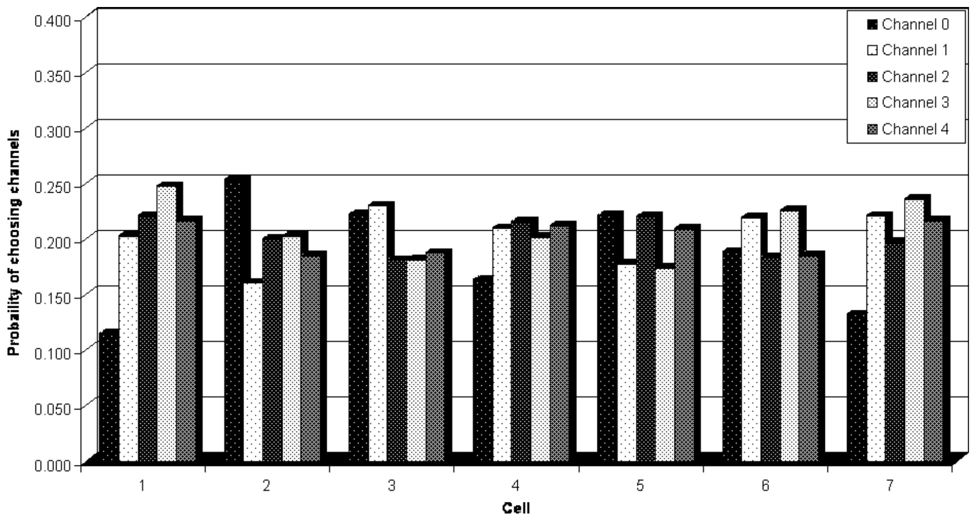

Fig. 18. Probability of assigning different channels to different cells for update based strategy.

different channels to different cells for a typical run. Figures 14 through 18 show the final probability of assigning different channels to different cells. These figures show that Algo. I (when using fixed or maximum usage strategies), Algo. II (when using non-retry strategy), and Algo. III (when using search strategy), are able to segregate different channels to different cells and other algorithms are not able to do this.

6. Conclusions

In this paper, an application of asynchronous cellular learning automata to cellular mobile system was presented and three dynamic channel assignment algorithms were proposed. In order to show the effectiveness of proposed algorithms, which are self-organizing channel assignment algorithms, computer simulations were conducted. Major conclusions that can be drawn from the simulation results are (1) The proposed algorithms have the same blocking probabilities as similar existing algorithms, (2) The interference between channels decreases as the learning process proceeds, that is, the CLA is able to segregate channels among the cells of the network, (3) The proposed algorithms have lower network overheads and hence consume less network resources, (4) Exchanging more status information wastes the network resources but increases the performance of the proposed algorithms.

Acknowledgment

The authors would like to thank the referees for their useful suggestions and comments.

References

1. I. Katzela and M. Naghshineh, Channel assignment schemes for cellular mobile telecommunication systems: A comprehensive survey, *IEEE Personal Communications* **3** (1996) 10–31.
2. W. K. Hale, Frequence assignment: Theory and applications, *Proc. IEEE* **68** (1980) 1497–1514.
3. Y. Furuya and Y. Akaiwa, Channel segregation — A distributed channel allocation scheme for mobile communication systems, *IEICE Trans.* **74**(6) (1991) 1531–1537.
4. S. Singh and D. P. Bertsekas, Reinforcement learning for dynamic channel allocation in cellular telephone systems, in *Advances in Neural Information Processing Systems: Proceedings of the 1996 Conference*, Cambridge (MIT Press, MA, 1997).
5. J. Li, N. B. Shroff and E. K. P. Chong, Channel carrying: A novel handoff scheme for mobile cellular networks, *IEEE/ACM Trans. Network.* **4** (1999) 35–50.
6. J. A. Zoellner and C. L. Beall, A breakthrough in spectrum conserving frequency assignment technology, *IEEE Trans. Electromagn. Compatibility*, EMC-12 (1977) 313–310.
7. J. Zerovnik, Experiments with a randomized algorithm for frequency assignment problem, Technical Report 97–27, Ecole Normale Superieure du Lyon, Lyon, France, September 1997.
8. A. Capone and M. Trubian, Channel assignment problem in cellular systems: A new model and a tabu search algorithm, *IEEE Trans. Veh. Technol.* **48** (1996) 1252–1260.
9. M. O. Berger, Neural channel assignment: The fast way, *Proc. IEEE Vehicular Technology Conference*, (1995), pp. 1557–1560.
10. K. A. Smith, *Solving Combinatorial Optimisation Problems Using Neural Networks*, PhD thesis Department of Electrical and Electronic Engineering, University of Melbourne, Melbourne, Austuralia, March 1996.
11. K. A. Smith and M. Palaniswami, Static and dynamic channel assignment using neural networks, *IEEE J. Selected Areas in Commun.* **5** (1997) 238–249.
12. W. K. Lai and G. G. Coghill, Channel assignment through evolurionary optimization, *IEEE Trans. Veh. Technol.* **45** (1996) 91–96.
13. K. A. Smith, A genetic algorithm for channel assignment problem, Technical Report, School of Business Systems, Monash University, Clayto, Victoria, Australia, (1998).
14. J. Kind, T. Niessen and R. Mathar, Theory of maximum packing and related channel assignment strategies for cellular radio networks, *Mathematical Methods of Operations Research* **48** (1998) 1–16.
15. A. Kulshershtha and K. N. Sivarajan, Maximum packing channel assignment in cellular networks, *IEEE Trans. Veh. Technol.* **48** (1999) 858–872.
16. J. S. Engel and M. Peritsky, Statistically optimum dynamic server assignment in systems with interfering servers, *IEEE Trans. Veh. Technol.*, VT–22 (1973) 203–209.
17. M. Zhang and T. S. Yum, The nonuniform compact pattern allocation algorithm for cellular mobile systems, *IEEE Trans. Veh. Technol.*, VT–40 (1991) 387–391.
18. B. Eklundh, Channel utilization and blocking probability in a cellular mobile telephone system with directed retry, *IEEE Trans. Communications*, Vol. COM-34, April 1986.
19. M. Zhang and T. S. Yum, Comparisons of channel assignment strategies in cellular mobile telephone systems, *IEEE Trans. Veh. Technol.* **38** (1989) 211–215.
20. S. S. Kuek, Ordered dynamic channel assignment scheme with reassignment in highway microcell, *IEEE Trans. Veh. Technol.* **41** (1992) 271–277.

21. H. Jiang and S. Rappaport, CBWL: A new channel assignment and sharing method for cellular communication systems, *IEEE Trans. Veh. Technol.* **43** 1994.
22. S. K. Das, S. K. Sen and R. Jayaram, Dynamic load balancing strategies for channel assignment using selective borrowing in cellular mobile environments, *Proc. ACM/IEEE Int. Conf. Mobile Computing and Networking*, November 1996, pp. 73–84.
23. S. K. Das, S. K. Sen and R. Jayaram, A distributed load balancing algorithm for hot cell problem in cellular mobile networks, *Proc. IEEE Int. Symp. High Performance Distributed Computing*, August 1997, pp. 254–263.
24. S. K. Das, S. K. Sen, R. Jayaram and P. Agrawal, An efficient distributed channel management algorithm for cellular mobile networks, *Proc. IEEE Int. Conf. Universal Personal Communications (ICUPC)*, Octber 1997, pp. 646–650.
25. S. K. Das, S. K. Sen and R. Jayaram, D-LBSB: A distributed load balancing algorithm for channel assignment in cellular mobile networks, *J. Interconnection Networks* **1**(3) (2000) 195–220.
26. H. Beigy, *Intelligent Channel Assignment Algorithms in Cellular Networks: A Learning Automata Approach*, PhD thesis, Amirkabir University of Technology, Tehran, Iran, May 2004.
27. E. Fredkin, Digital machine: A informational process based on reversible cellular automata, *Physica D*, **45** (1990) 254–270.
28. N. H. Packard and S. Wolfram, Two-dimensional cellular automata, *J. Stat. Phys.* **38** (1985) 901–946.
29. M. Mitchell, Computation in cellular automata: A selected review, Technical Report, Santa Fe Institute, Santa Fe, NM, USA, September 1996.
30. J. Kari, Reversability of 2D cellular automata is undecidable, *Physica D* **45** (1990) 379–385.
31. K. S. Narendra and K. S. Thathachar, *Learning Automata: An Introduction* (Printice-Hall, New York, 1989).
32. M. A. L. Thathachar and B. R. Harita, Learning automata with changing number of actions, *IEEE Trans. Syst. Man Cybern.* **SMC-17** (1987) 1095–1100.
33. P. R. Srikantakumar and K. S. Narendra, A learning model for routing in telephone networks, *SIAM J. Control Optim.* **20** (1982) 34–57.
34. B. J. Oommen and E. V. de St. Croix, Graph partitioning using learning automata, *IEEE Trans. Comput.* **45** (1996) 195–208.
35. B. J. Oommen and T. D. Roberts, Continuous learning automata solutions to the capacity assignment problem, *IEEE Trans. Comput.* **49** (2000) 608–620.
36. M. R. Meybodi and H. Beigy, A note on learning automata based schemes for adaptation of BP parameters, *J. Neurocomput.* **48** (2002) 957–974.
37. M. R. Meybodi and H. Beigy, New learning automata based algorithms for adaptation of backpropagation algorithm parameters, *Int. J. Neural Syst.* **12** (2002) 45–68.
38. H. Beigy and M. R. Meybodi, *Call Admission in Cellular Networks: A Learning Automata Approach*, Vol. 2510 of *Springer-Verlag Lecture Notes in Computer Science*, Springer-Verlag, October 2002, pp. 450–457.
39. H. Beigy and M. R. Meybodi, *A Learning Automata Based Dynamic Guard Channel Scheme*, Vol. 2510 of *Springer-Verlag Lecture Notes in Computer Science*, Springer-Verlag, October 2002, pp. 643–650.
40. H. Beigy and M. R. Meybodi, *An Adaptive Uniform Fractional Guard Channel Algorithm: A Learning Automata Approach*, Vol. 2690 of *Springer-Verlag Lecture Notes in Computer Science*, Springer-Verlag, March 2003, pp. 405–409.

41. H. Beigy and M. R. Meybodi, Adaptive uniform fractional channel algorithms, *Iranian J. Electr. Comput. Eng.* **3** (2004) 47–53.
42. H. Beigy and M. R. Meybodi, An adaptive call admission algorithm for cellular networks, *Electr. Comput. Eng.* **31** (2005) 132–151.
43. M. R. Meybodi, H. Beigy and M. Taherkhani, Application of cellular learning automata to image processing, *Proc. First Conf. Mathematics and Communication*, Iranian Telecommunication Research Center, Tehran, Iran, October 2000, pp. 23.1–23.10.
44. M. R. Meybodi, H. Beigy and M. Taherkhani, Cellular learning automata and its applications, *Sharif J. Sci. Technol.* **19**(25) (2003) 54–77.
45. H. Beigy and M. R. Meybodi, Asynchronous cellular learning automata, *Automatica* **44** (2008) 1350–1357.
46. M. R. Meybodi and H. Beigy, A mathematical framework for cellular learning automata, *J. Adv. Complex Syst.* **7** (2004) 295–320.
47. M. R. Meybodi and H. Beigy, Open synchronous cellular learning automata, *J. Adv. Complex Syst.* **10** (2007) 527–556.
48. M. R. Meybodi and M. Taherkhani, Application of cellular learning automata in modeling of rumor diffusion, *Proc. 9th Conf. Electrical Engineering, Power and Water Institute of Technology, Tehran, Iran*, May 2001, pp. 102–110.
49. M. R. Meybodi and M. R. Kharazmi, Image restoration using cellular learning automata," *Proc. 1st Iranian Conf. Machine Vision, Image Processeing and Applications*, Birjand, Iran, May 2000, pp. 244–254.
50. M. R. Kharazmi and M. R. Meybodi, Image segmentation using cellular learning automata, *Proc. 10th Iranian Conf. Electrical Engineering, ICEE'96*, Tabriz, Iran, May 2001.
51. M. R. Kharazmi and M. R. Meybodi, Image restoration using cellular learning automata, *Proc. 2nd Iranian Conf. Machine Vision, Image Processeing and Applications*, Tehran, Iran (2003), pp. 261–270.
52. M. R. Meybodi and M. R. Kharazmi, Application of cellular learning automata to image processing, *J. Amirkabir* **14**(56A) (2004) 1101–1126.
53. M. R. Meybodi and M. R. Khojasteh, Application of cellular learning automata in modelling of commerce networks, *Proc. 6th Ann. Int. Computer Society of Iran Computer Conf. CSICC-2001*, Isfehan, Iran, February 2001, pp. 284–295.
54. H. Beigy and M. R. Meybodi, *A Self-Organizing Channel Assignment Algorithm: A Cellular Learning Automata Approach*, Vol. 2690 of *Springer-Verlag Lecture Notes in Computer Science*, Springer-Verlag (2003), pp. 119–126.
55. M. R. Meybodi and F. Mehdipour, VLSI placement using cellular learning automata, *Proc. 8th Ann. Int. Comput. Soc. Iran Computer Conference CSICC-2001*, Mashad, Iran, March 2003, pp. 195–203.
56. M. Esnaashari and Meybodi, A cellular learning automata based clustering algorithm for wireless sensor networks, *Sensor Lett.* **6**(5) (2008) 723–735.
57. L. Lamport, Time, clocks and the ordering of events in a distributed system, *Commun. ACM* **21** (1978) 558–565.