

## ADAPTIVE LIMITED FRACTIONAL GUARD CHANNEL ALGORITHMS: A LEARNING AUTOMATA APPROACH\*

HAMID BEIGY

*Department of Computer Engineering, 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*

Received 9 May 2009

Revised 28 October 2009

In this paper, two learning automata based adaptive limited fractional guard channel algorithms for cellular mobile networks are proposed. These algorithms try to minimize the blocking probability of new calls subject to the constraint on the dropping probability of the handoff calls. To evaluate the proposed algorithms, computer simulations are conducted. The simulation results show that the performance of the proposed algorithms are close to the performance of the limited fractional guard channel algorithm for which prior knowledge about traffic parameters are needed. The simulation results also show that the proposed algorithms outperforms the recently introduced dynamic guard channel algorithms.

*Keywords:* Cellular mobile networks; call admission control; limited fractional guard channel; learning automata; nonstationary environment; adaptive limited fractional guard channel.

### 1. Introduction

The service area of a cellular network is partitioned into regions, called *cells*. Each cell has a base station that manages allocation of channels to mobile stations reside in the cell. When a mobile station moves across the cell boundary and enters to a new cell, handoff is required, which must be transparent to users. If an idle channel is available in the destination cell, the handoff is resumed and a channel is assigned to it; otherwise the call is dropped. Two commonly used performance

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

measures for cellular networks are: *dropping probability of handoff calls* ( $B_h$ ) and *blocking probability of new calls* ( $B_n$ ). The dropping probability of handoff calls represents the probability that a handoff call being dropped during handovers. This probability is defined as the ratio between the number of handoff calls dropped by the system and the total number of handoff calls. The blocking probability of new calls represents the probability that a new call being denied access to the network. This probability is defined as the percentage of new calls that are denied access to the network.

In order to have control on dropping probability of handoff calls and blocking probability of new calls call admission control algorithms are used, which determine whether a call should be either accepted or rejected at the base station. Both the blocking probability of new calls and the dropping probability of handoff calls are affected by the used call admission algorithm. Blocking more new calls generally decreases the dropping probability of handoff calls and admitting more new calls generally decreases the blocking probability of new calls and increases the dropping probability of handoff calls. Since the dropping of a handoff call is usually more objectionable than blocking of a new call, it widely believed that the call admission algorithms must give a higher priority to the handoff calls as compared to the new calls. These call admission algorithms are called *priority based call admission algorithms*. Various priority based call admission algorithms have been reported in the literatures; they can be classified into three broad categories: *equal access sharing with priority*, *reservation based*, and *queueing priority* schemes.<sup>1</sup>

1. *Equal access sharing with priority schemes*: In equal access sharing with priority schemes, all calls have access to all channels but handoff calls have a higher priority than new calls. These schemes can be classified into two main classes: *call thinning* and *new call thinning* schemes. The call thinning schemes accept new calls with a certain probability that depends on the number of ongoing calls in the cell;<sup>2,3</sup> while new call thinning schemes accept new calls with a probability that depends on the number of ongoing new calls in the cell.<sup>4</sup> Both schemes accept the handoff calls when the cell has free channels. Equal access sharing with priority schemes can be static<sup>4</sup> or dynamic.<sup>5</sup> In static schemes, the probability of acceptance of new calls is fixed; while in dynamic schemes the probability of acceptance of new calls is varied when traffic vary.
2. *Reservation based schemes*: In these schemes, a subset of channels is reserved for sole use of handoff calls. These schemes can be divided into two main groups: *equal access sharing with reservation* and *complete partitioning schemes*. In equal access sharing with reservation schemes, a portion of channels is reserved for sole use of handoff calls. Whenever the number of calls (new calls) exceeds a certain threshold, these schemes reject new calls until the number of calls (new calls) goes below the threshold. These schemes accept handoff calls as long as the cell has free channels. When the number of calls is compared with the given threshold, this scheme is called *call bounding*<sup>2,6-12</sup> and when the number of

new calls is compared with the given threshold, it is called *new call bounding*.<sup>4</sup> Equal access sharing with reservation schemes can reserve an integral number<sup>6–11</sup> or a fractional number<sup>2,12</sup> of channels for sole use of handoff calls. Schemes with fractional number of guard channels have more control on the blocking probability of the new calls and the dropping probability of the handoff calls than schemes with integral number of guard channels. These schemes can also be static<sup>2,7–9</sup> or dynamic,<sup>10–12</sup> where in the former schemes, the threshold is fixed while in the later schemes the threshold varies as traffic vary. *Complete partitioning* schemes partition the channels between handoff and new calls by dedicating a certain number of channels to each type of calls.<sup>13</sup>

3. *Queueing priority schemes*: In *queueing priority* schemes, new or handoff calls are accepted whenever there are free channels for that type of calls. When there is no free channel for a type of call, calls may be queued and calls of other types are blocked and cleared from the system. Queueing priority schemes can be divided into three groups: *new call queueing schemes*,<sup>14</sup> *handoff call queueing schemes*,<sup>15–17</sup> and *all call queueing schemes*.<sup>15,18</sup>

In the current literatures, two approaches are commonly proposed for designing call admission algorithms. The first approach, called *static call admission*, considers system parameters such as new and handoff calls as stationary and pursues the design of a static call admission algorithm for the worst case scenario.<sup>2,4,6,8</sup> The second approach, called *adaptive/dynamic call admission*, considers system parameters as nonstationary and pursues the design of a call admission algorithm that adjusts its parameter during the operation of the network.<sup>5,10,11,19,20</sup> In design of adaptive call admission algorithms for cellular networks, two approaches can be used for gathering information regarding traffic in the network. The first approach, called *nonautonomous schemes*, assumes that each base station has some knowledge about the traffic condition in neighboring cells. Such conditions are expressed in terms of the number of new and handoff calls present in each base station, new and handoff arrival rates and/or channel holding time for each class of traffic. The actual implementation of these algorithms will introduce a large communication and processing overhead. The second approach, called *autonomous schemes*, assumes that state information about neighboring cells that is necessary for call admission decision may be deduced from the local information available at the base station.

The optimality of static call admission algorithms has been studied under stationary traffic conditions.<sup>2,7</sup> In real systems the arrival rate is a time varying parameter, resulting in periods of overload and underload, which may degrade the quality of service. In such situations, adaptive schemes have superiority over static schemes. Recently, several adaptive learning automata based call admission algorithms are proposed in the literature for cellular networks. Beigy and Meybodi proposed a learning automata based algorithm that adjusts the probability of acceptance of new calls ( $\pi$ ) for uniform fractional channel policy.<sup>5</sup> In Ref. 12, a learning automata based algorithm is given which adjusts parameters of limited fractional

guard channel policy. It was shown that the learning automaton finds the optimal values of parameters of limited fractional guard channel algorithm. In Ref. 21, two learning automata based algorithms are given which adjust the threshold for guard channel policy and their steady state behavior studied. Beigy and Meybodi proposed cellular learning automata based algorithm which adjusts the threshold for guard channel policy.<sup>22</sup> This algorithm use some status information of neighboring cells to adjust the threshold for guard channel policy.

In this paper, we propose two learning automata based adaptive and autonomous call admission control algorithms based on limited fractional guard channel algorithm. These algorithms use only the current channel occupancy of the given cell and dynamically adjust the number of channels to be reserved for handoff calls. In these algorithms, each base station, independently of other base stations in the network, reserves a number of channels for handoff calls in order to guarantee that the dropping probability of handoff calls does not exceed a predetermined threshold. The proposed algorithms adapt the number of guard channels in a such a way that the blocking probability of new calls is minimized subject to the constraint on the dropping probability of handoff calls. Since the learning automaton starts its learning without any priori knowledge about its environment, the proposed algorithms do not need any prior information about the input traffic. One of the most important advantages of the proposed algorithms is that no status information will be exchanged between neighboring cells. However, the exchange of such status information may increase the performance of the proposed algorithm at the expense of communication and computation overheads. The simulation results show that the performance of these algorithms is near to the performance of the limited fractional guard channel scheme that needs to know all the traffic parameters and better than the dynamic guard channel algorithm.<sup>21</sup>

The rest of the paper is organized as follows. The limited fractional guard channel algorithm is given in Sec. 2 and the learning automata are briefly described in Sec. 3. The proposed algorithms are presented in Sec. 4. The simulation results are given in Sec. 5 and Sec. 6 concludes the paper.

## 2. Limited Fractional Guard Channel Policy

In the limited fractional guard channel algorithm (LFG), a fractional number of channels is reserved in each cell exclusively for the handoff calls.<sup>2</sup> Suppose that the given cell has  $C$  full duplex channels. The LFG algorithm uses two parameters  $T$  and  $\pi$ . In LFG, when the number of occupied channels is below the threshold  $T$ , new calls are accepted. When  $T$  channels are occupied in the cell, new calls are accepted with probability  $\pi$ . When more than  $T$  channels are occupied in the cell, the LFG scheme rejects new calls until the number of occupied channels goes below the threshold  $T$ . This scheme accepts handoff calls as long as the cell has free channels. Parameter  $\pi$  is in fact the probability that a new channel to be served or not. Therefore  $T + \pi$  is the average number of reserved channels. Since in the

LFG algorithm, both  $T$  and  $\pi$  control the acceptance of the new calls, we consider  $T + \pi$  as a control parameter. For studying the blocking performance of the LFG algorithm, we consider a homogenous wireless network where all cells have the same number of channels,  $C$ , and experience the same new and handoff calls arrival rates. In each cell, the arrival of the new calls and the handoff calls are Poisson distributed with rates  $\lambda_n$  and  $\lambda_h$ , respectively and the channel holding time of new and handoff calls are exponentially distributed with the same mean  $\mu^{-1}$ . Note that the same service rate for both types of calls implies that the base station of a cell does not need to discriminate between new and handoff calls, once they are connected. These assumptions have been found reasonable as long as the number of mobile users in a cell is much greater than the number of channels allocated to that cell. Define the state of a cell at time  $t$  by the total number of occupied channels,  $c(t)$ . Thus, the channel occupancy of the cell can be modeled by a continuous time Markov chain with states  $0, 1, \dots, C$ . Figure 1 shows the state transition diagram of a system with  $C$  channels for the LFG algorithm.

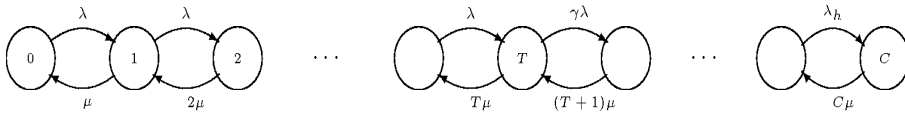


Fig. 1. Markov chain model of cell using LFG algorithm.

Define the steady state probability  $P_n = \lim_{t \rightarrow \infty} \text{Prob}[c(t) = n]$  as the probability of  $n$  channels being occupied. Given this, it is straightforward to derive probability  $P_n$  (for  $n = 0, 1, \dots, C$ ). These state probabilities are given as follows:

$$P_n = \begin{cases} \frac{\rho^n}{n!} & \text{if } n \leq T \\ \gamma a^{-(T+1)} \frac{(\rho a)^n}{n!} & \text{if } T < n \leq C, \end{cases} \quad (1)$$

where

$$P_0 = \left[ \sum_{n=0}^T \frac{\rho^n}{n!} + \gamma a^{-(T+1)} \sum_{n=T+1}^C \frac{(\rho a)^n}{n!} \right]^{-1}, \quad (2)$$

and  $a = \lambda_h / \lambda$ ,  $\rho = \lambda / \mu$ ,  $\gamma = [a + (1 - a)\pi]$ , and  $\lambda = \lambda_n + \lambda_h$ . Given these state probabilities, we can find the dropping probability of the handoff calls,  $B_h(C, T, \pi)$ , and the blocking probability of the new calls,  $B_n(C, T, \pi)$ , as

$$B_h(C, T, \pi) = \gamma a^{-(T+1)} \frac{(\rho a)^C}{C!} P_0, \quad (3)$$

$$B_n(C, T, \pi) = (1 - \pi) \frac{\rho^T}{T!} P_0 + \gamma a^{-(T+1)} P_0 \sum_{n=T+1}^C \frac{(\rho a)^n}{n!}. \quad (4)$$

In the rest of this section, we state two important properties of  $B_n(C, T, \pi)$  and  $B_h(C, T, \pi)$ , which will be used later in the paper. These properties are proved in Appendix A.

*Property 1.*  $B_h(C, T, \pi)$  is a monotonically increasing function of both  $T$  and  $\pi$ .

**Corollary 1.**  $B_n(C, T, \pi)$  is a monotonically increasing function of  $T + \pi$ .

**Proof.** Since  $B_n(C, T, \pi)$  is a monotonically increasing function of both  $T$  and  $\pi$ , it is a monotonically increasing function of  $T + \pi$ .  $\square$

*Property 2.*  $B_n(C, T, \pi)$  is a monotonically decreasing function of both  $T$  and  $\pi$  provided that  $\rho < (T + 1)$  and  $\frac{\lambda_n}{\lambda} < \frac{1}{T+1}$ .

**Corollary 2.**  $B_n(C, T, \pi)$  is a monotonically decreasing function of  $T + \pi$  provided that  $\rho < (T + 1)$  and  $\frac{\lambda_n}{\lambda} < \frac{1}{T+1}$ .

**Proof.** Since  $B_n(C, T, \pi)$  is a monotonically decreasing function of both  $T$  and  $\pi$ , it is a monotonically decreasing function of  $T + \pi$ .  $\square$

### 3. Stochastic Learning Automata

Learning automata (LA) are adaptive decision making units that can learn to choose the optimal action from a set of actions by interaction with an unknown environment. At each instant, the automaton chooses an action from its action probability distribution and applies it to an environment. The environment provides a stochastic response called reinforcement signal, to the automaton, which is used to update the action probability distribution. LAs can be classified into two main groups: finite action-set learning automata (FALA) and continuous action-set learning automata (CALA).<sup>23</sup> When the FALA is used for solving optimization problems, we need to discretize the parameter space, so that actions of the automaton can be possible values of the corresponding parameter. The accuracy of the solution can be increased by choosing a finer discretization and hence increasing the number of actions of automaton, which leads to slow convergence of the learning algorithm. In order to provide a higher rate of convergence for FALA, hierarchical structure LA,<sup>24</sup> discretized LA,<sup>25</sup> estimator algorithms,<sup>26–28</sup> and pursuit algorithms<sup>29–32</sup> have been introduced. A more satisfying solution is to use CALA in which the action-set of the automaton is a continuous variable. A CALA uses a probability distribution function to choose its actions and a learning algorithm to update this function based on the reinforcement signal. Since the action-set is continuous, instead of penalty probabilities for actions, we now have a penalty probability function  $M(\alpha) = E[\beta(\alpha)|\alpha_n = \alpha]$ . CALA has no knowledge about  $M(\cdot)$  and its objective is find an action  $\alpha$  that minimizes  $M(\alpha)$ .

A CALA, whose action probability distribution at instant  $n$  is Gaussian with mean  $\mu_n$  and standard deviation  $\sigma_n$ , is introduced in Ref. 33. At instance  $n$ , the

action chosen by CALA can be represented by a pair  $(\alpha_n, \mu_n)$ , where  $\alpha_n$  is chosen from distribution  $N(\mu_n, \sigma_n)$  and  $\mu_n$  is the mean of the Gaussian distribution. Then the environment, in response to the chosen action, emits two reinforcement signals  $\beta(\alpha)$  and  $\beta(\mu)$  for actions  $\alpha_n$  and  $\mu_n$ , respectively. Finally, CALA uses the rule given below to update its action probability distribution by updating  $\mu_n$  and  $\sigma_n$ .

$$\begin{aligned}\mu_{n+1} &= \mu_n + a f_1[\mu_n, \sigma_n, \alpha_n, \beta(\alpha), \beta(\mu)] \\ \sigma_{n+1} &= \sigma_n + a f_2[\mu_n, \sigma_n, \alpha_n, \beta(\alpha), \beta(\mu)] - K a [\sigma_n - \sigma_L],\end{aligned}\quad (5)$$

where  $f_1$ ,  $f_2$ , and  $\phi$  are

$$\begin{aligned}f_1(\mu, \sigma, \alpha, \beta(\alpha), \beta(\mu)) &= \left[ \frac{\beta(\alpha) - \beta(\mu)}{\phi(\sigma)} \right] \left[ \frac{\alpha - \mu}{\phi(\sigma)} \right] \\ f_2(\mu, \sigma, \alpha, \beta(\alpha), \beta(\mu)) &= \left[ \frac{\beta(\alpha) - \beta(\mu)}{\phi(\sigma)} \right] \left[ \left( \frac{\alpha - \mu}{\phi(\sigma)} \right)^2 - 1 \right] \\ \phi(\sigma) &= (\sigma - \sigma_L) I\{\sigma > \sigma_L\} + \sigma_L,\end{aligned}\quad (6)$$

$\sigma_L$  is the minimum value of standard deviation for Gaussian distribution,  $I$  is indicator function,  $K > 0$ , and  $a \in (0, 1)$  are parameters of the algorithm and  $\phi[\sigma_n]$  is to guarantee  $\sigma_{n+1} \geq \sigma_n$ . The objective of the above rule is the convergence of  $N(\mu_n, \sigma_n)$  to  $N(\alpha^*, 0)$ , where  $M(\alpha)$  attains its minimum at  $\alpha = \alpha^*$ . For this algorithm it is shown that with arbitrary large probability,  $\mu_n$  will converge close to  $\alpha^*$  and  $\sigma_n$  will converge close to  $\sigma_L$ , if we choose  $\mu$  and  $\sigma_L$  sufficiently small and  $K$  sufficiently large.<sup>33</sup> Beigy and Meybodi proposed a CALA whose action probability distribution at instant  $n$  is Gaussian with mean  $\mu_n$  and standard deviation  $\sigma_n$ .<sup>34</sup> At instant  $n$ , this automaton chooses  $\alpha_n$  from distribution  $N(\mu_n, \sigma_n)$  and applies it to the random environment, which emits the reinforcement signal  $\beta_n \in [0, 1]$ . Then, the following rule was used to update the parameters  $\mu_n$  and  $\sigma_n$  of Gaussian distribution:

$$\begin{aligned}\mu_{n+1} &= \mu_n - a \beta(\alpha_n) \sigma_n (\alpha_n - \mu_n), \\ \sigma_{n+1} &= f(\sigma_n),\end{aligned}\quad (7)$$

where  $a$  is learning rate and  $f$  is a function that produces a sequence of  $\sigma_n$  such that  $\sigma_n \geq 0$ ,  $\sum_{n=1}^{\infty} \sigma_n^3 = \infty$ , and  $\sum_{n=1}^{\infty} \sigma_n^4 < \infty$ . It was shown that this algorithm finds the optimal action with a probability as close to unity as desired, i.e.  $\mu_n$  will converge close to  $\alpha^*$ .<sup>34</sup> In these two automata,<sup>33,34</sup> it is assumed that no information about the optimal action is available and at the beginning, the actions are chosen from a near uniform distribution. So the initial distribution is considered as a Gaussian distribution with a random mean and a large value of standard deviation.

In *continuous action reinforcement learning automata*, it is assumed that the actions-set is a bounded continuous random variable over interval  $[\alpha_{min}, \alpha_{max}] \in \mathbb{R}$ .<sup>35</sup> This automaton selects its action  $\alpha_n$  at instant  $n$  from a continuous probability density function,  $f$ , and applies it to a random environment that emits a response

$\beta_n \in [0, 1]$ . It is assumed that no information about the actions is available initially and therefore the probabilities of choosing actions are equal, that is, the initial distribution is uniform. Based on  $\beta_n$ ,  $G(n)$  is updated according to the following rule:

$$G(n+1) = \begin{cases} a[G(n) + (1 - \beta_n)H(\alpha, \alpha_n)] & \text{if } \alpha_n \in [\alpha_{max}, \alpha_{min}] \\ 0 & \text{otherwise,} \end{cases} \quad (8)$$

where  $a$  is a normalization factor and  $H(\alpha, r)$  is a symmetric Gaussian neighborhood function centered on  $\alpha_n$  and has the effect of spreading the reinforcement signals for neighboring actions of the selected action.

Learning automata have been used successfully in many applications such as computer networks,<sup>36–38</sup> solving NP-Complete problems,<sup>39–41</sup> capacity assignment,<sup>42,43</sup> neural network engineering,<sup>44–47</sup> and cellular networks<sup>5,10</sup> to mention a few.

#### 4. Adaptive Limited Fractional Guard Channel Algorithms

The LFG algorithm assumes that traffic is a stationary process with known parameters, but in reality, traffic is a nonstationary process. Even if we assume that traffic is stationary, its parameters may not be known a priori or may vary with time. In either case, the optimal value for the parameters of the LFG algorithm is not known a priori and may vary with time. In nonstationary traffic, adaptive LFG algorithms which adapt the parameters of the LFG as the network operates have superiority over the LFG algorithm. In this section, we consider the call admission problem for two classes of calls: new and handoff calls and present two learning automata based adaptive LFG algorithms to adapt value of  $T + \pi$  for the LFG algorithm. These algorithms can be used particularly when  $\lambda_n$ ,  $\lambda_h$  and  $\mu$  are unknown and possibly time varying. The objective of these algorithms is to adapt parameter  $T + \pi$  in such a way that the blocking probability of new calls is minimized subject to the constraint that the dropping probability of handoff calls be at most  $p_h$ . Since  $T + \pi$  is a continuous parameter, each of the proposed algorithms uses a continuous action-set learning automaton (CALA) for adaptation of the value of parameter  $T + \pi$ . Let  $x(n) = T(n) + \pi(n)$  be the parameter of the LFG algorithm at instant  $n$ , and  $x(n)$  takes values in the interval  $[x_{min}, x_{max}]$ , where  $0 \leq x_{min} < x_{max} \leq C$ . The CALA chooses its actions using a Gaussian distribution,  $N(\mu; \sigma)$ . This Gaussian distribution is updated using the reinforcement signal,  $\beta$ , which is emitted from the environment. Initially, the CALA chooses one of its actions with equal probability using a Gaussian distribution with a large variance.

Since  $x(n)$  and  $\mu(n)$  must be in the interval  $[x_{min}, x_{max}]$ , the above mentioned CALA cannot be used directly to adapt the value of  $T + \pi$ , and hence a projected of them version must be used. In the projected version, a constraint set  $H = \{y | x_{min} \leq y \leq x_{max}\}$  is used for updating  $\mu$  as well as choosing actions of CALA. In the projected version, when the updated value of  $\mu$  goes outside of the



constraint set  $H$ , then  $\mu$  is pushed into  $H$  and also when the action,  $x$ , chosen by the CALA does not belong to  $H$ , then  $x$  is pushed into  $H$ .

The proposed algorithms can be described as follows. Each base station is equipped with a CALA for adapting  $T + \pi$ . When a new call arrives at a given cell, the learning automaton associated to that cell chooses one of its actions, say  $x(n)$ . Let  $T(n) = \lfloor x(n) \rfloor$  and  $\pi(n) = x(n) - \lfloor x(n) \rfloor$ . If the number of busy channels of a cell is less than  $T(n)$ , then the incoming call will be accepted; when the cell has  $T(n)$  busy channels, then the call will be accepted with probability  $\pi(n)$ ; otherwise the incoming call will be blocked. Then the base station computes the current estimate of the dropping probability of the handoff calls and based on the result of the comparison of this quantity with the specified level of QoS,  $p_h$ , a reinforcement signal will be produced. Finally the automaton uses the received reinforcement signal to update the Gaussian distribution by updating  $\mu_n$  and  $\sigma_n$ . The handoff calls will be accepted as long as the cell has free channels.

Depending on the learning algorithm used by the CALA and the way that the reinforcement signal is computed, a variety of algorithms may be obtained, two of which are described in the following sections. The differences between the proposed algorithms are the way that they compute the reinforcement signal and the learning algorithm used to update action probability vector. The first algorithm, which we call ALFG I, uses the short time response while the second algorithm, which we call ALFG II, uses both short time and long time responses to update the mean and the variance of the Gaussian distribution.

#### 4.1. Adaptive limited fractional guard channel Algorithm I (ALFG I)

In this section, we first propose a new CALA and then use it for adaptation of  $T + \pi$  parameter for LFG algorithm. The proposed CALA chooses its actions using Gaussian distribution. The action chosen by the CALA is then applied to the environment, which emits a reinforcement signal from set  $\{-1, +1\}$ . When the value of the reinforcement signal is  $-1$  ( $+1$ ), then the CALA decreases (increases)  $\mu$  by  $a(n)$ , that is

$$\mu(n+1) = \mu(n) + \beta(n)a(n), \quad (9)$$

where  $a(n) \geq 0$  is the learning parameter and is a decreasing function of  $n$  such that

$$\lim_{n \rightarrow \infty} a(n) \rightarrow 0.$$

Decreasing  $a(n)$  gives an implicit averaging and ensures the asymptotic stability of this algorithm at equilibrium point. The variance is updated independent of the reinforcement signal in such a manner that  $\sigma(n)$  is a decreasing function of  $n$ . In order to increase the probability of escaping from nonoptimal points, we don't allow  $\sigma(n)$  converging to zero. Therefore, another sufficiently small parameter  $\sigma_L > 0$  is considered and keep the objective of our algorithm as  $\sigma(n)$  to  $\sigma_L$  and  $\mu(n)$

converging to  $\alpha^*$ . Since the updating algorithm for  $\sigma(n)$  does not automatically guarantee that  $\sigma(n+1) \geq \sigma_L$  ( $\sigma_L > 0$ ), a projected version of  $\sigma(n)$ , denoted by  $\phi(\sigma(n))$  and given below is used:

$$\phi(\sigma) = (\sigma - \sigma_L)I\{\sigma > \sigma_L\} + \sigma_L,$$

where  $I\{\cdot\}$  is the indicator function. It is assumed that no information about actions is available and therefore actions are chosen from a near uniform distribution by setting variance of the Gaussian distribution to a large value.

In what follows, we describe the proposed call admission algorithm, which uses the above learning automaton. In this algorithm, the base station of each cell executes the CALLADMISSION shown in Procedure 1 and can be described as : initially  $\sigma$  is set to a large value and  $\mu$  is set to a random value in the interval  $[0, C]$ , where  $C$  is the number of channels allocated to the cell. When a new call arrives at a given cell, the learning automaton associated to that cell chooses one of its actions, say  $x(n)$ . Let  $T(n) = \lfloor x(n) \rfloor$  and  $\pi(n) = x(n) - \lfloor x(n) \rfloor$ . If the number of busy channels of a cell is less than  $T(n)$ , then the incoming call will be accepted; when the cell has  $T(n)$  busy channels, then a call will be accepted with probability  $\pi(n)$ ; otherwise the incoming call will be blocked. Then the base station computes the current estimate of the dropping probability of the handoff calls and based on this quantity and the prespecified level of QoS ( $p_h$ ), the reinforcement signal is produced according to the following rule:

$$\beta(n) = \text{sign} \left[ \hat{B}_h - p_h \right], \quad (10)$$

where  $\hat{B}_h$  is the current estimate of the dropping probability of handoff calls. It is evident that when  $\hat{B}_h$  is less (greater) than  $p_h$ , then  $\beta$  becomes  $-1$  ( $+1$ ) and hence,  $\mu$  must be decreased (increased). Finally the automaton updates the  $\mu_n$  and  $\sigma_n$ , accordingly. The base station accepts handoff calls as long as the cell has free channels.

#### 4.2. Adaptive limited fractional guard channel Algorithm II (ALFG II)

The algorithm proposed in this section uses a new CALA which not only updates  $\mu$  but it also updates  $\sigma$  according to the reinforcement signal, which enables the algorithm to adapt itself to time varying environments. The proposed CALA rather than using the current reinforcement signal of the environment for updating  $\sigma$  and  $\mu$ , uses previous reinforcement signals as well as the current reinforcement signal. It seems that if the proposed algorithms, in addition to the current reinforcement signal, also uses previous reinforcement signals for updating  $\mu$  and  $\sigma$ , it is possible for the algorithms to have a higher speed of convergence.

The CALA used in this algorithm chooses its actions from a Gaussian distribution  $N(\mu, \sigma)$ . The action chosen by the automaton is applied to the environment

---

**Procedure 1** Learning automata based adaptive limited fractional guard channel algorithm I (ALFG I)

---

**procedure** CALLADMISSION

**if** (a new call is arrived at the cell) **then**  
    choose an action of CALA from  $N(\mu(n), \sigma(n))$  and call it  $x$   
    **if** ( $x < 0$ ) **then**  
       $x \leftarrow 0$   
    **else if** ( $x > C$ ) **then**  
       $x \leftarrow C$   
    **end if**  
     $T \leftarrow \lfloor x \rfloor$   
     $\pi \leftarrow x - \lfloor x \rfloor$   
    **if** ( $c(n) < T$ ) **then**  
      accept the incoming call  
    **else if** ( $(c(n) = T)$  and ( $rand(0, 1) \leq \pi$ )) **then**  
      accept the incoming call  
    **else**  
      reject the incoming call  
    **end if**  
    compute an estimate of the dropping probability of handoff calls ( $\hat{B}_h$ )  
     $\beta(n) \leftarrow \text{sign} [\hat{B}_h - p_h]$   
     $\sigma(n+1) \leftarrow \phi(f(\sigma(n)))$   
     $\mu(n+1) \leftarrow \mu(n) + \beta(n)a(n)$   
    **if** ( $\mu(n) < 0$ ) **then**  
       $\mu(n) \leftarrow 0$   
    **else if** ( $\mu(n) > C$ ) **then**  
       $\mu(n) \leftarrow C$   
    **end if**  
**else if** (a handoff call is arrived at the cell) **then**  
    **if**  $c(n) < C$  **then**  
      accept the incoming call  
    **else**  
      reject the incoming call  
    **end if**  
**end if**  
**end procedure**

---

emits a reinforcement signal,  $\beta(n)$ , from interval  $[-1, +1]$ . When  $\beta(n)$  is negative (positive), then  $\mu(n)$  will be increased (decreased) by  $a(n)S(n)\beta(n)$ , that is

$$\mu(n+1) = \mu(n) - a(n)\beta(n)S(n), \quad (11)$$

where  $S(n)$  is the average value of  $|\beta(k)|$  taken over the previous steps from step 1 up to and including step  $n$  and given by the following equation:

$$S(n) = S(n-1) + \frac{1}{n} [|\beta(n)| - S(n-1)]. \quad (12)$$

It should be noted that when the automaton approaches its optimal value,  $S$  approaches zero. According to the updating algorithm,  $\sigma(n)$  will be changed slowly

when the selected action is far from its optimal value and will be decreasing fast as the selected action is near to its optimal value. The  $\sigma(n)$  is update according to the following rule:

$$\sigma(n+1) = \phi(\sigma(n) - a(n)[1 - bS(n)]), \quad (13)$$

where  $\phi(\cdot)$  is a projection function described later. In this algorithm,  $0 < a(n) < 1$  is the learning rate and  $b \geq 1$  is a constant. In order to have a stable equilibrium point, the proposed algorithm uses a monotonically decreasing learning parameter  $a(n)$ , that is

$$\lim_{n \rightarrow \infty} a(n) \rightarrow 0.$$

In this CALA, we want  $N(\mu(n), \sigma(n))$  converging to  $N(\alpha^*, 0)$  where the conditional expected value of  $|\beta|$  (absolute value of  $\beta$ ) given  $\alpha$  attains its minimum value. Since  $\sigma(n)$  cannot converge to zero, another parameter,  $\sigma_L > 0$ , is used in which the objective of learning is converging  $\sigma(n)$  to  $\sigma_L$  with  $\mu(n)$  converging to  $\alpha^*$ . Since the updating algorithm for  $\sigma(n)$  does not automatically guarantee that  $\sigma(n+1) \geq \sigma_L$  ( $\sigma_L > 0$ ), a projected version of  $\sigma(n)$ , denoted by  $\phi(\sigma(n))$  and given below is used:

$$\phi(\sigma) = (\sigma - \sigma_L)I\{\sigma > \sigma_L\} + \sigma_L,$$

where  $I\{\cdot\}$  is the indicator function. In what follows, we describe the proposed call admission algorithm, which uses the above learning automaton. In the proposed call admission algorithm, the base station of each cell executes the CALLADMISSION shown in Procedure 2 and can be described as: initially  $\sigma$  is set to a large value and  $\mu$  is set to a random value in the interval  $[0, C]$ , where  $C$  is the number of channels allocated to the cell. When a new call arrives at a given cell, the learning automaton associated to that cell chooses one of its actions, say  $x(n)$ . Let  $T(n) = \lfloor x(n) \rfloor$  and  $\pi(n) = x(n) - \lfloor x(n) \rfloor$ . If the number of busy channels of a cell is less than  $T(n)$ , then the incoming call will be accepted; when the cell has  $T(n)$  busy channels, then a call will be accepted with probability  $\pi(n)$ ; otherwise the incoming call will be blocked. Then the base station computes the current estimate of the dropping probability of the handoff calls and based on this quantity and the prespecified level of QoS ( $p_h$ ), the reinforcement signal is produced using according to the following rule:

$$\beta(n) = \hat{B}_h - p_h, \quad (14)$$

where  $\hat{B}_h$  is the current estimate of the dropping probability of handoff calls. It is evident that when  $\hat{B}_h$  is less than  $p_h$ , then  $\beta$  becomes negative and hence,  $\mu$  must be increased. When  $\hat{B}_h$  is greater than  $p_h$ , then  $\beta$  becomes positive and hence,  $\mu$  must be decreased. The produced reinforcement signal ( $\beta(n)$ ) shows the relative distance of the selected action ( $\alpha(n)$ ) from the optimal action. Finally, the CALA updates  $\mu(n)$ ,  $\sigma(n)$ , and the long term reinforcement signal accordingly. The base station accepts the handoff calls as long as the cell has free channels.

**Procedure 2** Learning automata based adaptive LFG algorithm II (ALFG II)**procedure** CALLADMISSION  **if** (a new call is arrived at the cell) **then**    choose an action of CALA from  $N(\mu(n), \sigma(n))$  and call it  $x$     **if** ( $x < 0$ ) **then**       $x \leftarrow 0$     **else if** ( $x > C$ ) **then**       $x \leftarrow C$     **end if**     $T \leftarrow \lfloor x \rfloor$      $\pi \leftarrow x - \lfloor x \rfloor$     **if** ( $c(n) < T$ ) **then**

accept the incoming call

**else if** ( $(c(n) = T)$  and  $(rand(0, 1) \leq \pi)$ ) **then**

accept the incoming call

**else**

reject the incoming call

**end if**    compute an estimate of the dropping probability of handoff calls ( $\hat{B}_h$ )     $\beta(n) \leftarrow \hat{B}_h - p_h$      $S(n) \leftarrow S(n-1) + \frac{1}{n} [|\beta(n)| - S(n-1)]$      $\mu(n+1) \leftarrow \mu(n) - a(n)\beta(n)S(n)$      $\sigma(n+1) = \phi(\sigma(n) - a(n)[1 - bS(n)])$     **if** ( $\mu(n) < 0$ ) **then**       $\mu(n) \leftarrow 0$     **else if** ( $\mu(n) > C$ ) **then**       $\mu(n) \leftarrow C$     **end if**  **else if** (a handoff call is arrived at the cell) **then**    **if**  $c(n) < C$  **then**

accept the incoming call

**else**

reject the incoming call

**end if**  **end if****end procedure****5. Simulation Results**

The objective of this section is to study the performance of the proposed call admission algorithms and compare their performance with the performance of some of the existing call admission algorithms. In order to evaluate the proposed algorithms, two network configurations are considered. In the first network configuration, a homogenous wireless network where all cells have the same number of channels and experience the same new and handoff calls arrival rates is used. For a homogenous cellular network the performance of the system can be studied by focusing on a single cell, under the assumption that the neighboring cells show independent random behavior. In the second network configuration, we study the performance

of the proposed algorithms for one dimensional cellular network. The simulation results presented in this section measure the performance of the different schemes with respect to mainly two criteria, namely *dropping probability of handoff calls* and *blocking probability of new calls*.

### 5.1. Homogenous network configuration

The first set of simulations is based on the single cell of homogenous cellular network system. Each cell has 8 full duplex channels ( $C = 8$ ). We use the following function to update the variance of ALFG I algorithm:

$$\sigma(n) = \frac{1}{\lfloor \frac{n}{10} \rfloor^{1/3}}, \quad (15)$$

where  $\lfloor \cdot \rfloor$  denotes the floor function. The role of floor function, division by 10, and exponent of  $1/3$  is to slow down the rate of decay for  $\sigma(n)$ . The sequence of real numbers  $\{\sigma(n)\}$  is such that  $\sigma(n) \geq 0$ ,  $\sum_{n=1}^{\infty} \sigma^3(n) = \infty^a$ ,  $\sum_{n=1}^{\infty} \sigma^4(n) < \infty$ .  $\sum_{n=1}^{\infty} \sigma^3(n) = \infty$  ensures that the sum of increments to the initial mean,  $\mu(0)$ , can be arbitrary large, so that any finite initial value of  $\mu(0)$  can be transformed into the optimal value. At the same time,  $\sum_{n=1}^{\infty} \sigma^4(n) < \infty$  ensures that the variance in  $\mu(n)$  is finite and the mean cannot diverge to infinity. Equation (15) implies that  $\sigma(n) \rightarrow 0$  as  $n \rightarrow \infty$  and in limit,  $\sigma(n)$  tends to zero and the action of learning automaton becomes equal to the mean. In order to study the proposed algorithms, six experiments are conducted whose results are given later in this section. In all experiments, we assume that the arrival of new calls is Poisson process with rate  $\lambda_n$  fixed at 30 calls and the arrival of handoff calls is Poisson process with rate  $\lambda_h$  varied between 2 and 20 calls. We also assume that the duration of calls is exponentially distributed with mean 6. We conducted experiments for  $p_h = 0.01$  and  $\sigma_L = 0.001$ .

#### 5.1.1. Experiment 1:

This experiment is conducted to study the behavior of the proposed call admission control algorithms in a homogenous network. This is done by observing the behavior of the algorithm during a single run. Each run takes 2,000,000 simulation cycles. In these simulations, algorithm ALFG I is ran with parameter  $a = 0.025$  and algorithm ALFG II is ran with parameters  $a = 0.025$  and  $b = 1$ . Figures 2 through 6 show the behavior of the proposed algorithms for a typical run. Figures 2 and 3 show that the blocking probability of new calls converges to a stationary point and oscillates around it while the constraint on the dropping probability of handoff calls are satisfied. Figures 5 and 6, which show channel usage in the cell, indicate that algorithm ALFG II has higher channel utilization in comparison to algorithm ALFG I. This fact also observable by looking at Figs. 2 and 3.

<sup>a</sup>  $\sum_{n=1}^{\infty} \left( \frac{1}{\lfloor \frac{n}{10} \rfloor^{1/3}} \right)^3 = \sum_{n=1}^{\infty} \frac{1}{\lfloor \frac{n}{10} \rfloor} \geq \sum_{n=1}^{\infty} \frac{10}{n} = \infty$ . The last equality is because of the divergence of harmonic series  $\sum \frac{1}{n}$ .

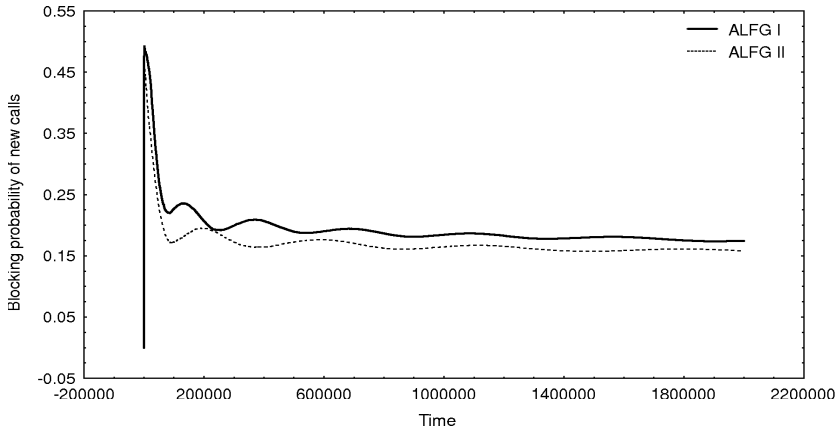


Fig. 2. Blocking probability of new calls for a typical run for  $\lambda_n = 10$ .

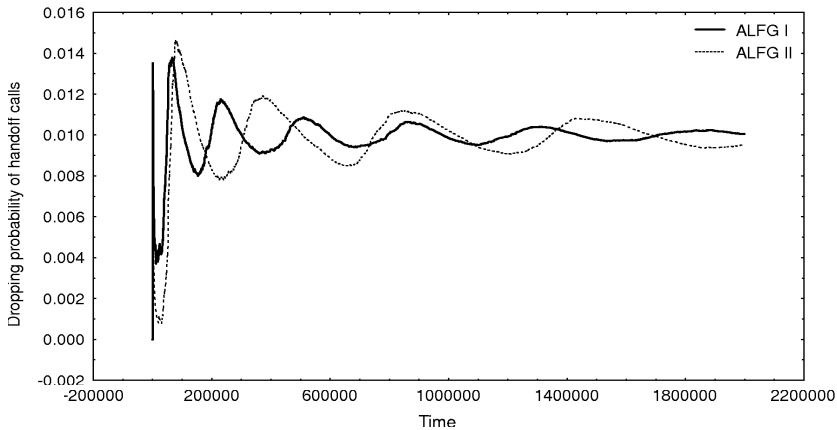


Fig. 3. Dropping probability of handoff calls for a typical run for  $\lambda_n = 10$ .

### 5.1.2. Experiment 2:

In this experiment, the performance of the proposed call admission control algorithms are measured by averaging over 30 different runs. The results of experiments are summarized in Figs. 7 through 12. Each run takes 2,000,000 simulation cycles. In these simulations, algorithm ALFG I is ran with parameter  $a = 0.025$  and algorithm ALFG II is ran with parameters  $a = 0.025$  and  $b = 1$ . The results reported in Figs. 7 through 11 confirm the results reported for experiment 1. Figure 12 indicate that the standard deviation is approaching  $\sigma_L$ , that is the algorithm satisfies the constraint on the dropping probability of handoff calls.

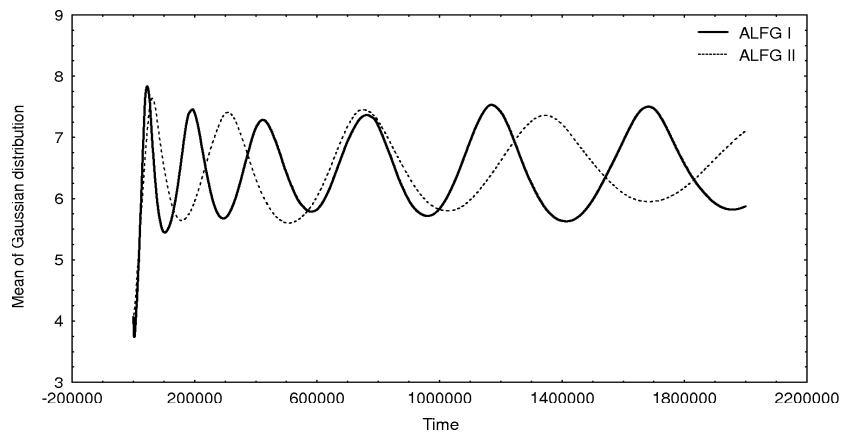


Fig. 4. The mean value of Gaussian distribution for a typical run for  $\lambda_n = 10$ .

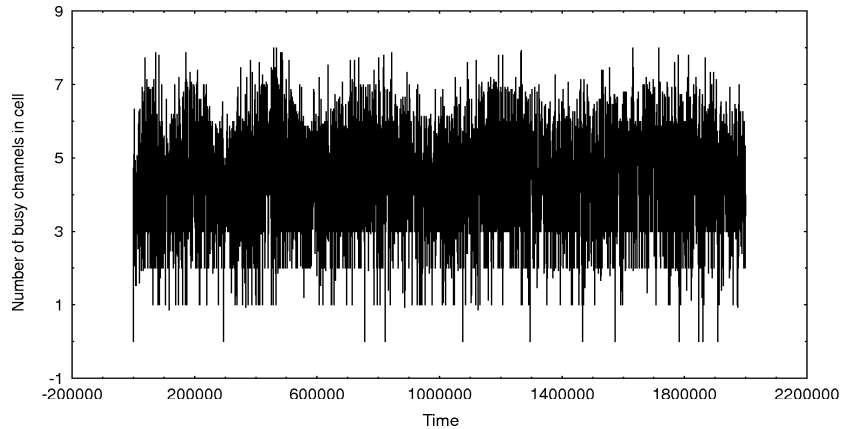


Fig. 5. The number of busy channels of ALFG I for a typical run for  $\lambda_n = 10$ .

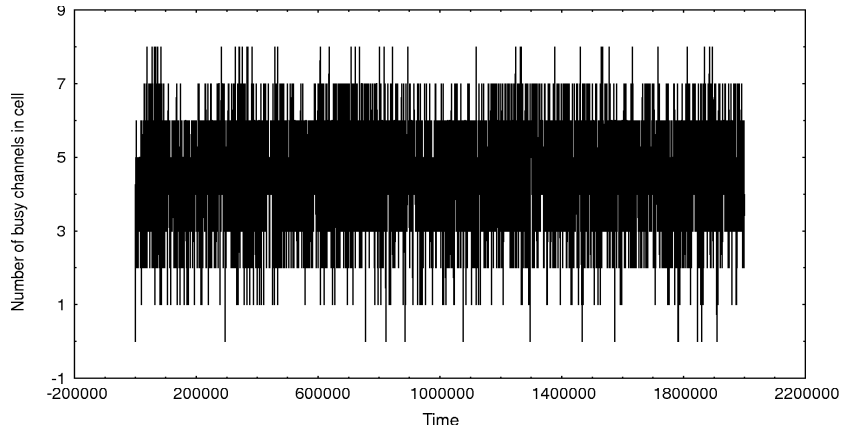


Fig. 6. The number of busy channels of ALFG II for a typical run for  $\lambda_n = 10$ .



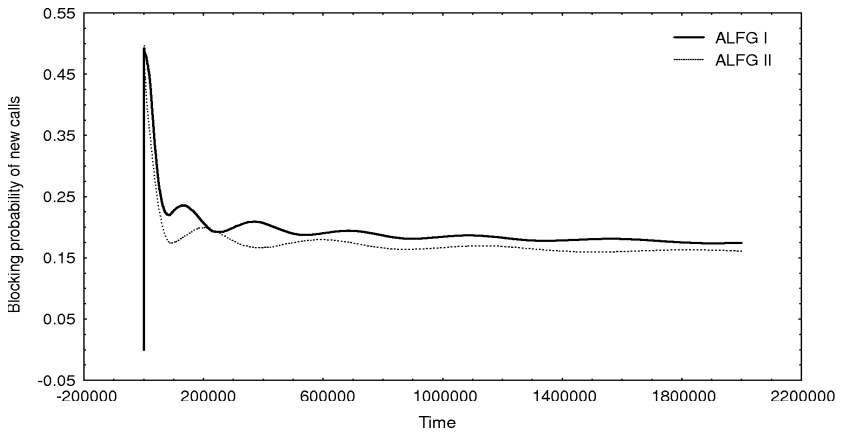


Fig. 7. Blocking probability of new calls for  $\lambda_n = 10$ .

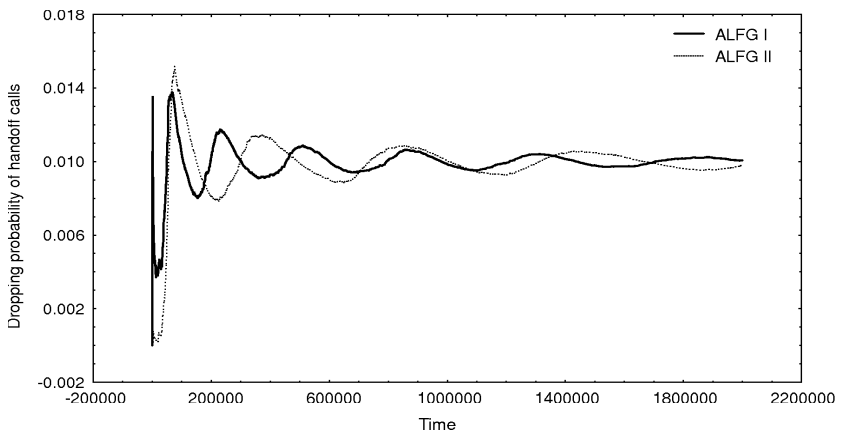


Fig. 8. Dropping probability of handoff calls for  $\lambda_n = 10$ .

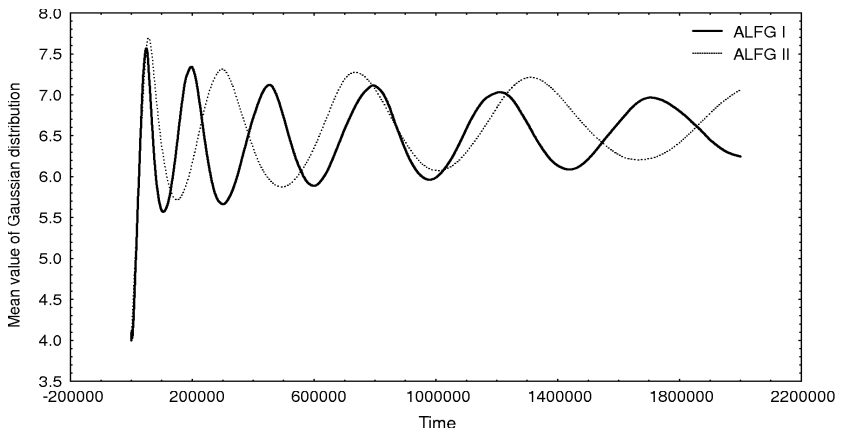


Fig. 9. The mean value of Gaussian distribution for  $\lambda_n = 10$ .

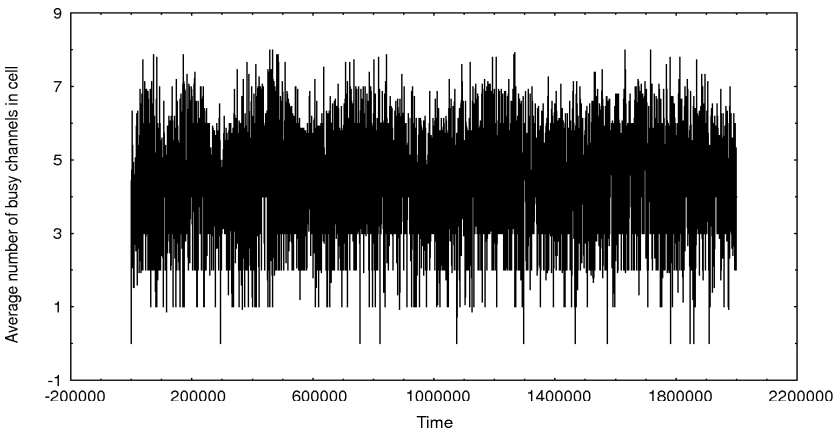


Fig. 10. The number of busy channels of ALFG I for  $\lambda_n = 10$ .

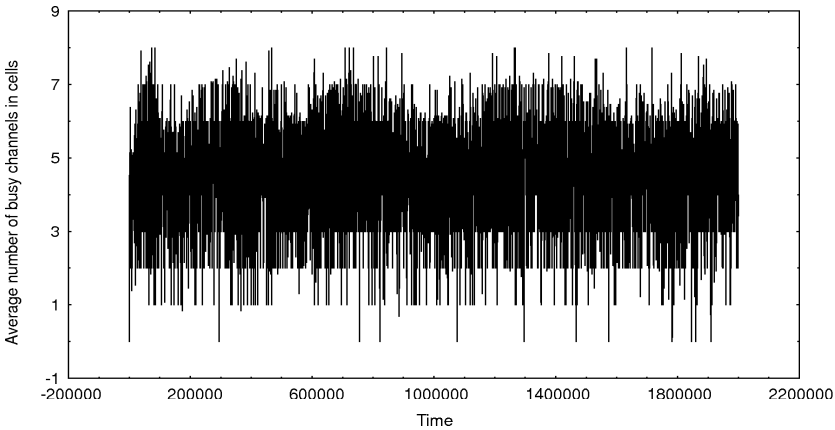


Fig. 11. The number of busy channels of ALFG II for  $\lambda_n = 10$ .

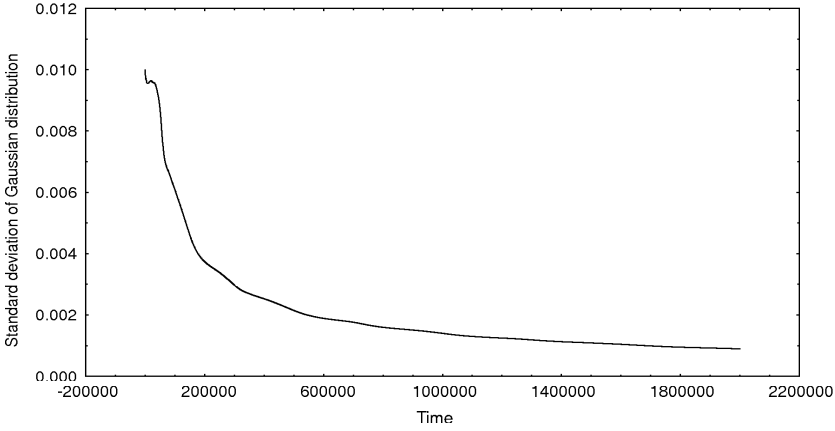


Fig. 12. The value of standard deviation of Gaussian distribution for ALFG II for  $\lambda_n = 10$ .

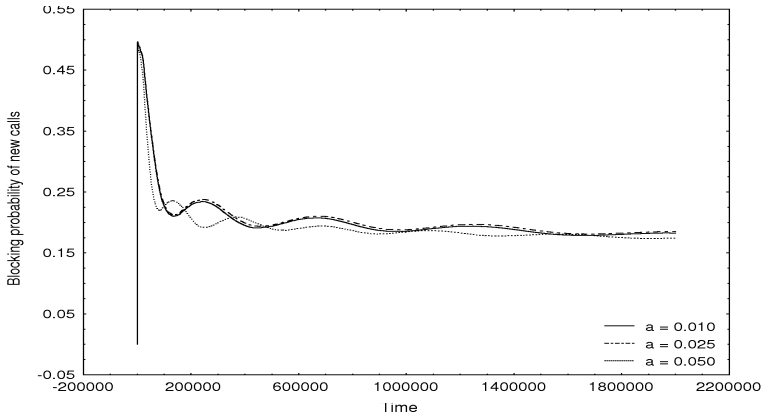


Fig. 13. Effect of learning rate on the blocking probability of new calls of ALFG I for  $\lambda_n = 10$ .

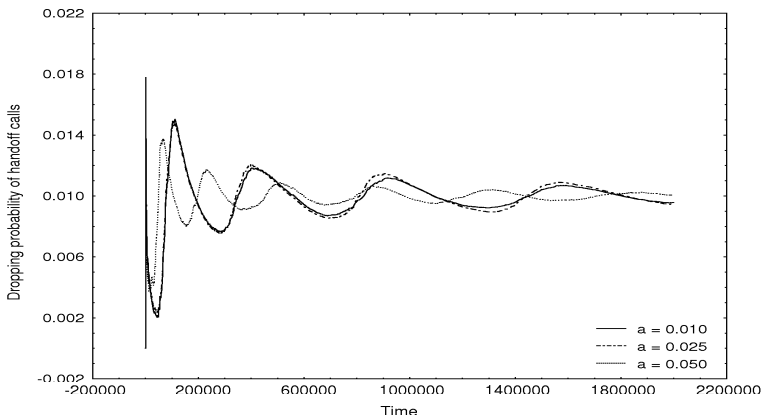


Fig. 14. Effect of learning rate on the dropping probability of handoff calls of ALFG I for  $\lambda_n = 10$ .

### 5.1.3. Experiment 3:

This experiment is conducted to study the effect of learning rate on the performance of the proposed call admission control algorithms. The results of experiments are summarized in Figs. 13 through 18 are obtained by averaging over 30 runs. Each run takes 2,000,000 simulation cycles. In these simulations, algorithm ALFG I is ran with parameter  $a = 0.025$  and algorithm ALFG II is ran with parameters  $a = 0.025$  and  $b = 1$ . Figures 13 and 14 shows that the behavior of algorithm ALFG I is not very sensitive to the learning parameter. Figure 15 indicate that the blocking probability of new calls improves as learning parameter  $a$  increases. The best results are obtained for  $a = 0.05$  when  $b$  is taken to be 1. Figure 16 shows that the blocking probability of new calls of algorithm ALFG II is not very sensitive to the learning parameter  $b$  when  $a$  is set to be 0.025. Figures 17 and 18 show that the dropping probability of handoff calls of algorithm ALFG II is not very sensitive to the learning parameters  $a$  and  $b$ .

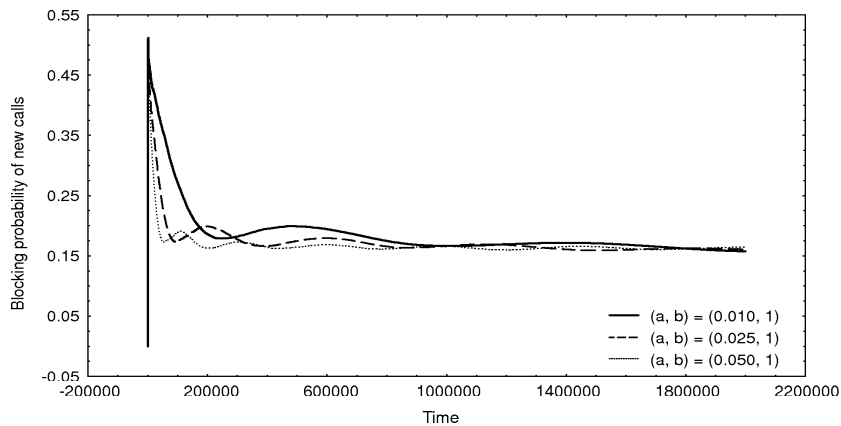


Fig. 15. Effect of learning rate on the blocking probability of new calls of ALFG II for  $\lambda_n = 10$ .

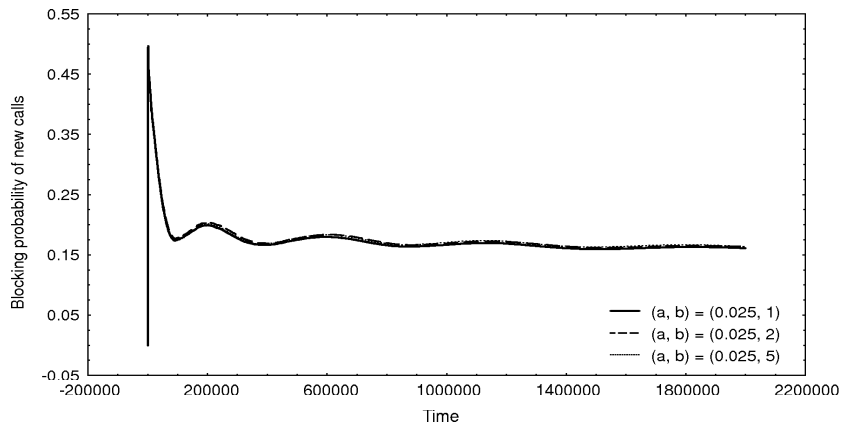


Fig. 16. Effect of learning rate on the blocking probability of new calls of ALFG II for  $\lambda_n = 10$ .

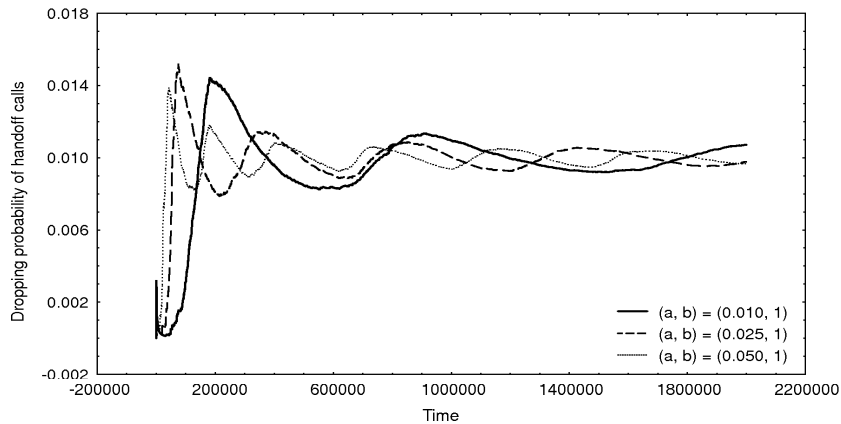


Fig. 17. Effect of learning rate on the dropping probability of handoff calls of ALFG II for  $\lambda_n = 10$ .

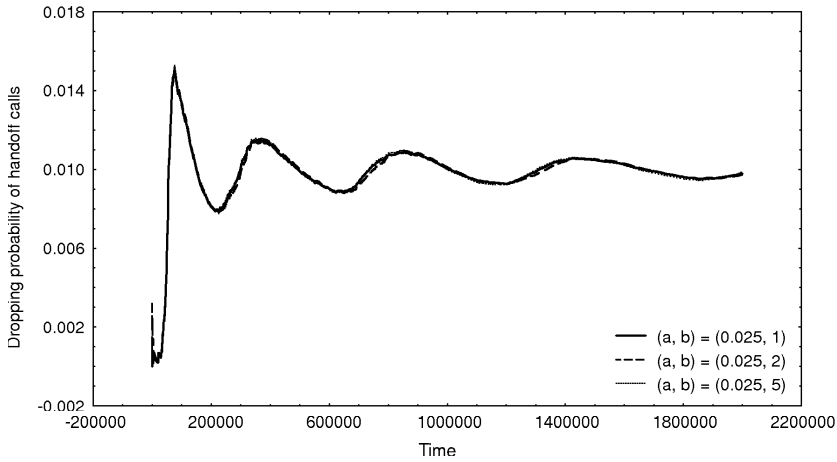


Fig. 18. Effect of learning rate on the dropping probability of handoff calls of ALFG II for  $\lambda_n = 10$ .

Table 1. The number of guard channels found by different algorithms.

| Case | $\lambda_h$ | LFG      | ALFG I    | ALFG II   |
|------|-------------|----------|-----------|-----------|
| 1    | 2           | 7.886353 | 7.6032828 | 7.7675834 |
| 2    | 4           | 7.623169 | 7.4371406 | 7.5296238 |
| 3    | 6           | 7.414429 | 7.2343069 | 7.3934577 |
| 4    | 8           | 7.242554 | 7.1537274 | 7.1726112 |
| 5    | 10          | 7.095825 | 6.6538288 | 7.0222077 |
| 6    | 12          | 6.869019 | 6.6139195 | 6.8826819 |
| 7    | 14          | 6.502563 | 6.4868276 | 6.3727648 |
| 8    | 16          | 6.231812 | 6.3589895 | 6.3222743 |
| 9    | 18          | 6.020142 | 6.2538798 | 6.0542565 |
| 10   | 20          | 5.515015 | 5.4076487 | 5.7346833 |

#### 5.1.4. Experiment 4:

This experiment is conducted to compare the number of guard channels found by the proposed call admission control algorithms and the algorithm given in Ref. 2. The results of experiments, which are summarized in Table 1, are obtained by averaging over 30 runs. Each run takes 2,000,000 simulation cycles. In these simulations, algorithm ALFG I is ran with parameter  $a = 0.025$  and algorithm ALFG II is ran with parameters  $a = 0.025$  and  $b = 1$ . The number of guard channels computed by both algorithms ALFG I and ALFG II are very near to the one computed by LFG for handoff rates ( $\lambda_h$ ) in Table 1. Table 1 indicates that ALFG II performs better than ALFG I for all cases except for case 7 for which the algorithm ALFG I performs better.

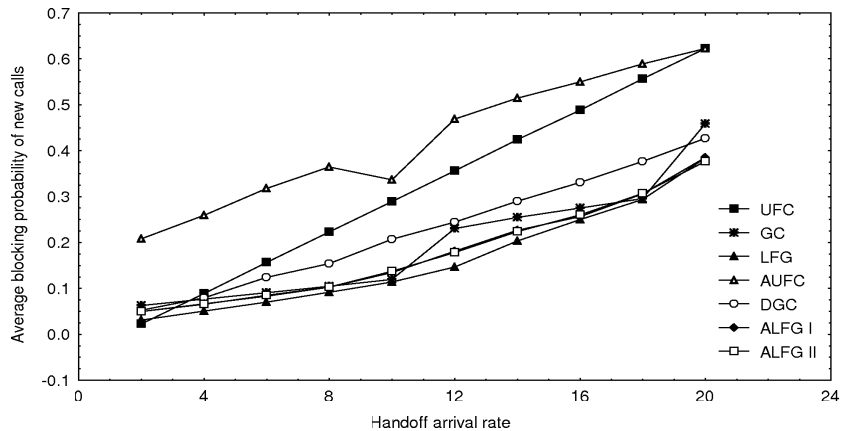


Fig. 19. Blocking probability of new calls for  $\lambda_n = 30$ .

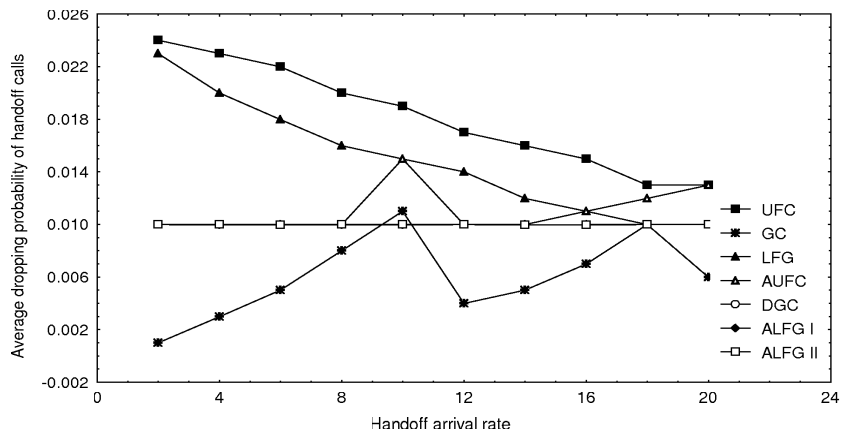


Fig. 20. Dropping probability of handoff calls for  $\lambda_n = 30$ .

5.1.5. Experiment 5:

This experiment is conducted to study the performance of the proposed call admission control algorithms and compare with some related algorithms. The results obtained for the proposed algorithms are compared with the results obtained for uniform fractional channel (UFC),<sup>3</sup> guard channel (GC),<sup>6</sup> limited fractional guard channel (LFG),<sup>2</sup> adaptive uniform fractional channel (AUFC),<sup>5</sup> and dynamic guard channel algorithm (DGC).<sup>10</sup> The results of experiments are summarized in Figs. 19 and 20 are obtained by averaging over 30 runs. Each run takes 2,000,000 simulation cycles. In these simulations, algorithm ALFG I is ran with parameter  $a = 0.025$  and algorithm ALFG II is ran with parameters  $a = 0.025$  and  $b = 1$ . The optimal number of guard channels for guard channel and limited fractional guard algorithms

are obtained by the algorithms given in Refs. 8 and 2, respectively and the optimal parameters of uniform fractional channel algorithm is obtained by algorithm given in Ref. 3.

By inspecting Figs. 19 and 20, it is evident that (a) the constraint on the dropping probability of handoff calls is maintained by all the adaptive algorithms, except AUFC, (b) the blocking probability of new calls for the adaptive algorithms, which don't need to know the traffic parameters, are close to the blocking probability of new calls for the static algorithms, which need to know the traffic parameters, (c) the proposed algorithms have the same performance in homogenous networks.

#### 5.1.6. *Experiment 6:*

This experiment is conducted to study the performance of the proposed algorithms for different number of channels allocated to each cell. Each point in these figures is obtained by averaging over 30 runs. Each run takes 2,000,000 simulation cycles. In these simulations, algorithm ALFG I is ran with parameter  $a = 0.025$  and algorithm ALFG II is ran with parameters  $a = 0.025$  and  $b = 1$ . Figures 21 and 22 show the blocking probability of new calls and the dropping probability of handoff calls for different number of channels allocated to the cell. These figures show that algorithm ALFG II has better performance when the number of channels allocated to the cell is less than 10 but both algorithms have the same performance when the number of channels allocated to the cell is greater than 10 for the given traffic condition.

### 5.2. *Linear network configuration*

The second set of simulations is conducted on a one dimensional cellular network with 18 cells, which for example can model a street or a highway. Each cell has 8 full duplex channels ( $C = 8$ ). We assume that the arrival of calls is Poisson process with rate  $\lambda_n$ , the duration of calls and dwell time<sup>b</sup> of calls are exponentially distributed with mean 18 and 3, respectively. We further assume that the mobile users in the network move with constant speed in random direction. We conducted simulations for  $p_h = 0.01$ ,  $\sigma_L = 0.01$ , and different values of  $\lambda_n$ . Each point in these figures is obtained by averaging over 30 runs. Each run takes 2,000,000 simulation cycles. In these simulations, algorithm ALFG I is ran with parameter  $a = 0.025$  and algorithm ALFG II is ran with parameters  $a = 0.025$  and  $b = 1$ . Figures 23 through 26 show the blocking probability of new calls for different arrival rates of new calls. These figures show that ALFG II has lower blocking probability of new calls than ALFG I and DGC algorithms.

Figures 27 through 29 show the dropping probability of handoff calls for different cells and different arrival rates of new calls. Figures 27 and 28 show that ALFG II has lower dropping probability of handoff calls than ALFG I. By comparing these figures with Figs. 23 and 24, it can be concluded that ALFG II in low and medium

<sup>b</sup>The average time that a mobile user resides in each cell.

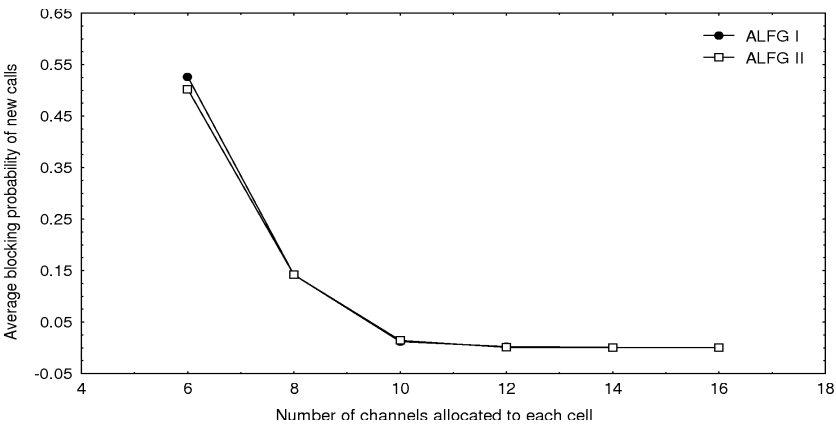


Fig. 21. Blocking probability of new calls for  $\lambda_n = 30$  and  $\lambda_h = 10$ .

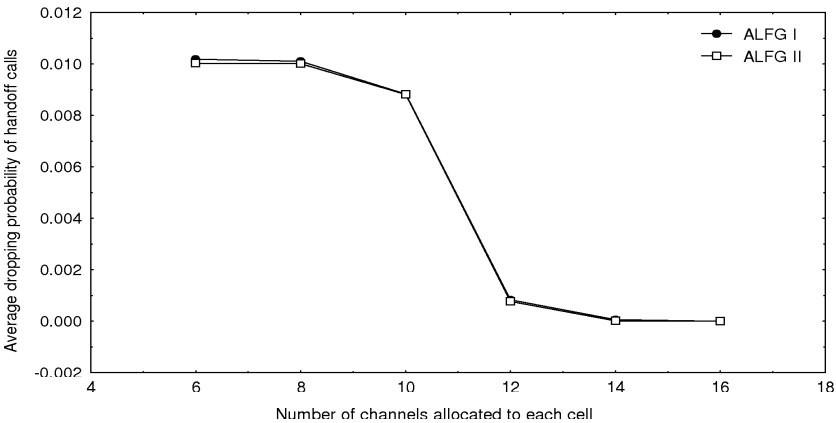


Fig. 22. Dropping probability of handoff calls for  $\lambda_n = 30$  and  $\lambda_h = 10$ .

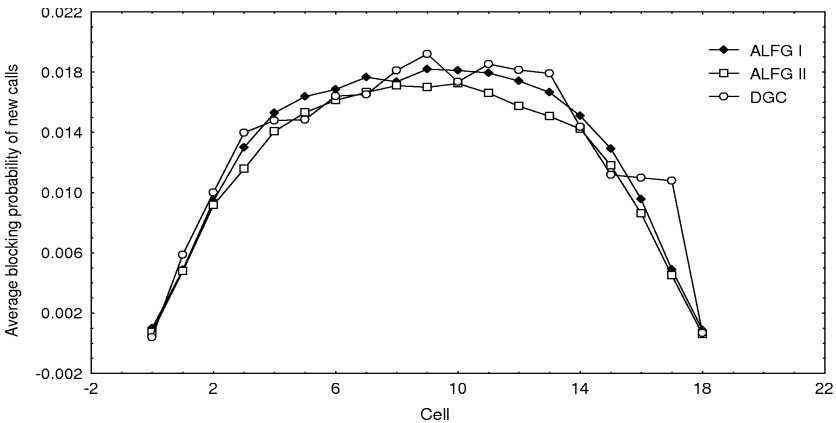


Fig. 23. Blocking probability of new calls for  $\lambda_n = 10$ .



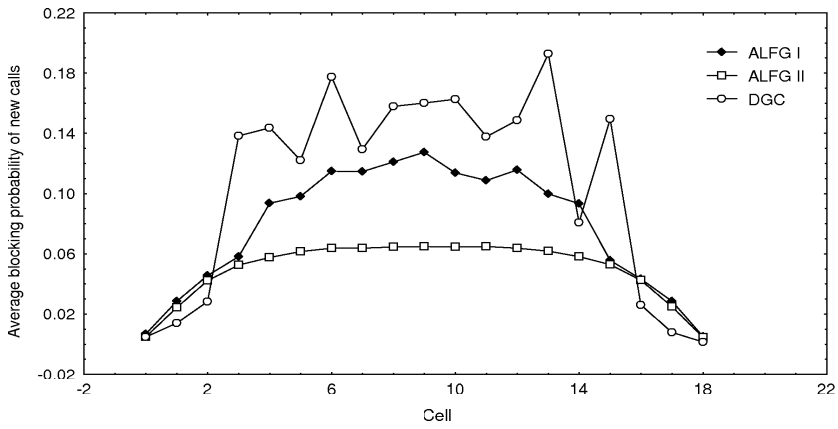


Fig. 24. Blocking probability of new calls for  $\lambda_n = 15$ .

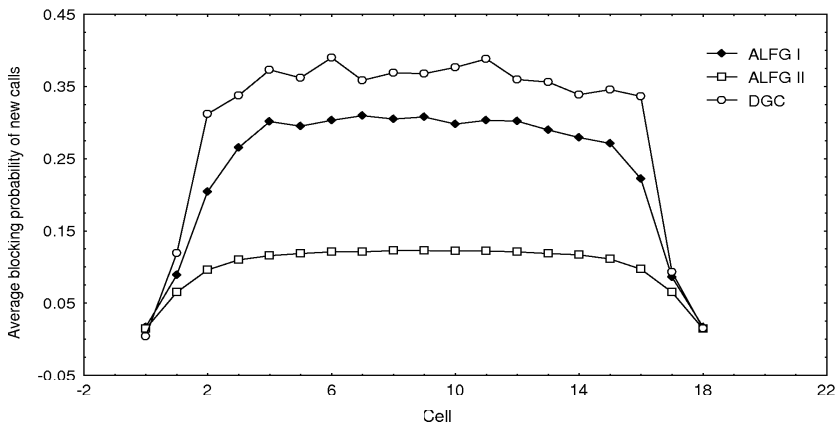


Fig. 25. Blocking probability of new calls for  $\lambda_n = 20$ .

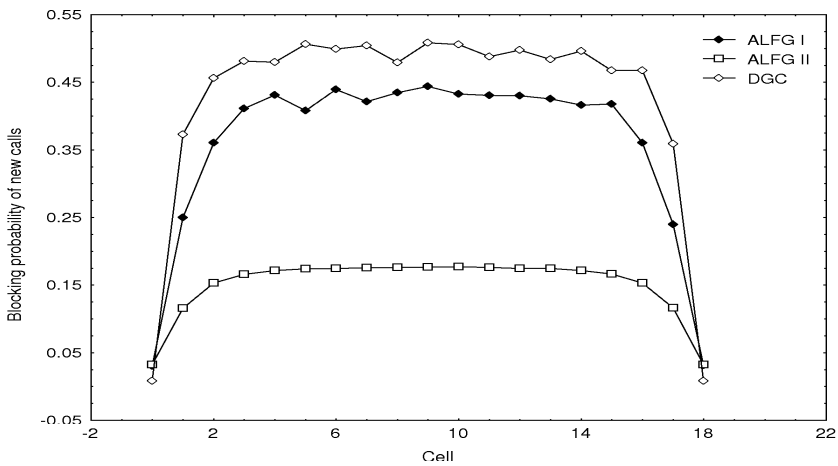


Fig. 26. Blocking probability of new calls for  $\lambda_n = 25$ .

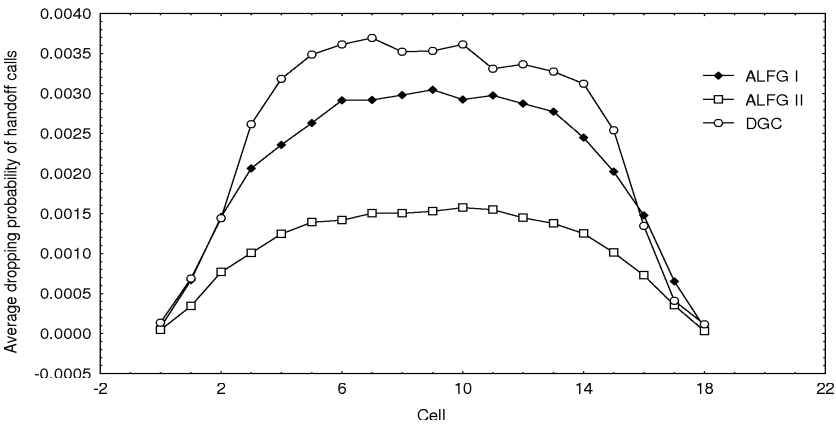


Fig. 27. Dropping probability of handoff calls for for  $\lambda_n = 10$ .

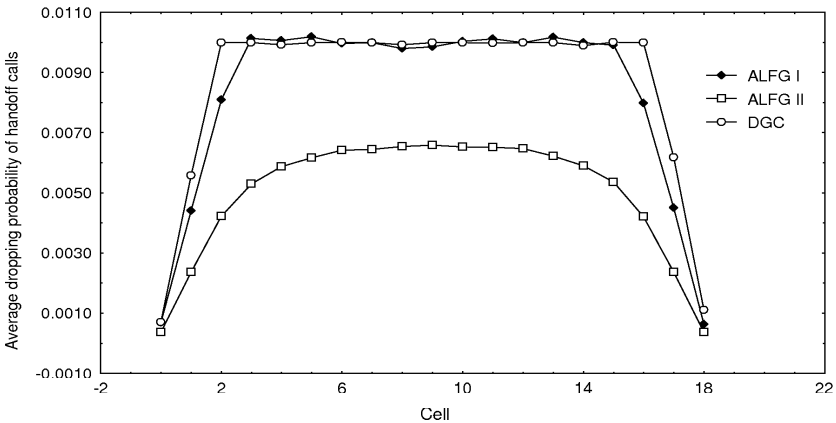


Fig. 28. Dropping probability of handoff calls for for  $\lambda_n = 15$ .

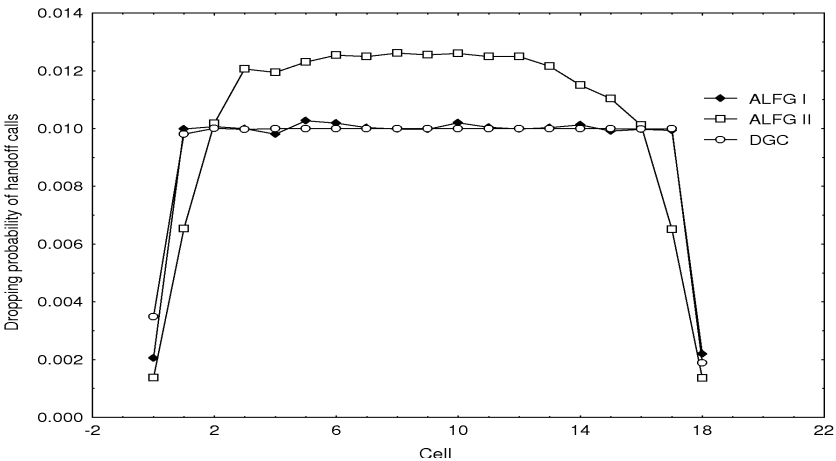


Fig. 29. Dropping probability of handoff calls for for  $\lambda_n = 20$ .

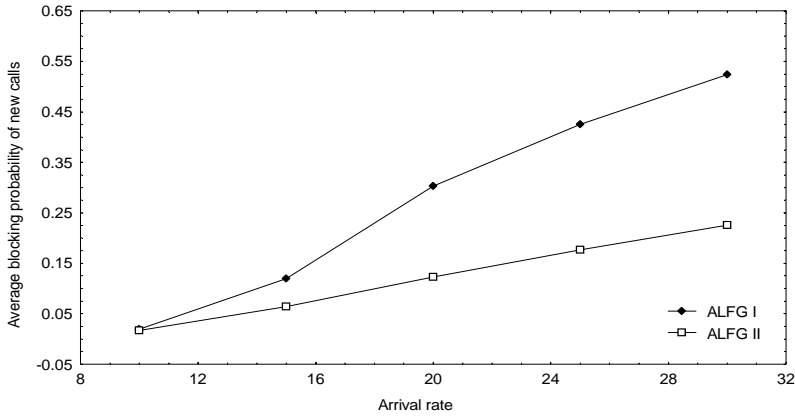


Fig. 30. Blocking probability of new calls for different arrival rates of new calls.

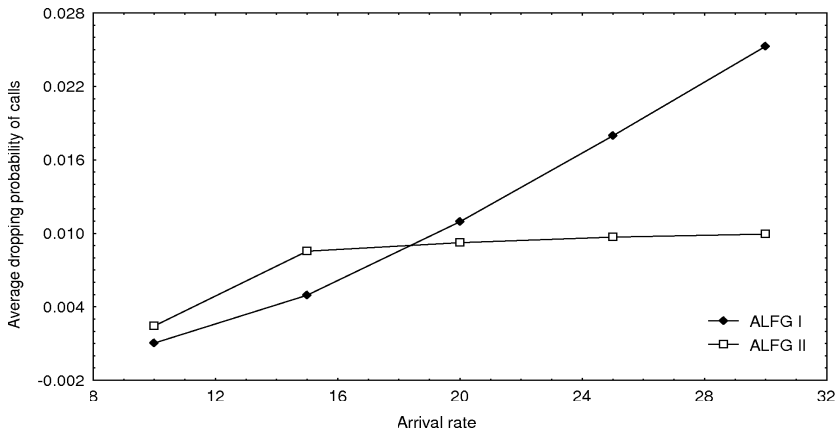


Fig. 31. Dropping probability of handoff calls for different arrival rates of new calls.

traffic conditions has higher channel utilization than ALFG I and Fig. 29 shows that in high traffic conditions, ALFG II cannot maintain the constraint on the dropping probability of handoff calls.

Figures 30 and 31 show the blocking probability of new calls and the dropping probability of handoff calls in cell 8 for different values of arrival rates of new calls, respectively. From Figs. 30 and 31, we can conclude that ALFG II has lower dropping probability of handoff calls and blocking probability of new calls in low and medium traffic conditions. Figure 31 shows that both ALFG I and DGC meet the objective, which is maintaining the upper bound on the dropping probability of handoff calls. Figure 30 shows that ALFG II performs better than ALFG I and DGC. Experiments have shown that a decreasing sequence for  $\{\sigma_n\}$  is not suitable for nonstationary environments. Figure 30 clarifies this point.

## 6. Conclusions

In this paper, two learning automata based autonomous adaptive call admission control algorithms for cellular networks were presented. The objective of these algorithms is to minimize the blocking probability of the new calls subject to the constraint on the dropping probability of the handoff calls. These algorithms have the advantages of a) being able to adaptively set the number of guard channels in environments with unknown and/or changing input traffic, b) possible autonomous operation. Computer simulations are conducted to evaluate the proposed call admission algorithms. The simulation results reveal performance improvement over existing schemes, which may require full knowledge of traffic parameters or assume static traffic parameters known a priori, assumptions that are probably not very realistic.

## Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions which improved the paper.

## Appendix A

**Proof of Property 1:** In order to show that  $B_h(C, T, \pi)$  is a monotonically increasing function of  $T$ , we need to show that  $B_h(C, T, \pi) < B_h(C, T + 1, \pi)$ . Using equation (3) and some algebraic simplifications, we obtain

$$\begin{aligned} & B_h(C, T, \pi) - B_h(C, T + 1, \pi) \\ &= \gamma a^{-(T+1)} \frac{(\rho\alpha)^C}{C!} \left[ \frac{\pi \frac{(\rho\alpha)^{T+1}}{(T+1)!} + \sum_{n=0}^T \frac{\rho^n}{n!}}{D(C, T + 1, \pi) D(C, T, \pi)} \right] (1 - a^{-1}), \\ &< 0, \end{aligned}$$

where  $D(C, T, \pi) = \sum_{n=0}^T \frac{\rho^n}{n!} + \gamma a^{-(T+1)} \sum_{n=T+1}^C \frac{(\rho\alpha)^n}{n!}$ .

In order to show that  $B_h(C, T, \pi)$  is a monotonically increasing function of  $\pi$ , we need to show that  $\frac{\partial B_h(C, T, \pi)}{\partial \pi} > 0$ . Differentiating  $B_h(C, T, \pi)$  with respect to  $\pi$ , we obtain

$$\begin{aligned} \frac{\partial B_h(C, T, \pi)}{\partial \pi} &= \frac{\rho^C}{C!} \left[ \frac{\sum_{n=0}^T \frac{\rho^n}{n!}}{D^2(C, T, \pi)} \right] (1 - a), \\ &> 0. \end{aligned}$$

**Proof of Property 2:** In the first part of the proof, we show that  $B_h(C, T, \pi)$  is a monotonically decreasing function of  $T$ . The blocking probability of new calls can be written as

$$B_n(C, T, \pi) = \frac{(1 - \pi)Z_T + \gamma D_2(C, T, \pi)}{D_1(C, T, \pi) + \gamma D_2(C, T, \pi)},$$

where  $D(C, T, \pi) = \sum_{n=0}^T \frac{\rho^n}{n!} + \gamma a^{-(T+1)} \sum_{n=T+1}^C \frac{(\rho a)^n}{n!}$ ,  $D_2(C, T, \pi) = a^{-(T+1)} \sum_{n=T+1}^C \frac{(\rho a)^n}{n!}$ ,  $Z_T = a^{-T} \frac{(\rho a)^T}{T!}$ , and  $D_1(C, T, \pi) = \sum_{n=0}^T \frac{\rho^n}{n!}$ . In order to show that  $B_h(C, T, \pi)$  is a monotonically decreasing function of  $T$ , we need to show that  $B_n(C, T, \pi) > B_n(C, T+1, \pi)$ . Using equation (4) and some algebraic simplifications, we obtain

$$\begin{aligned}
 & B_n(C, T+1, \pi) - B_n(C, T, \pi) \\
 &= \frac{(1-\pi)Z_{T+1} + \gamma D_2(C, T+1, \pi)}{D(C, T+1, \pi)} - \frac{(1-\pi)Z_T + \gamma D_2(C, T, \pi)}{D(C, T, \pi)}, \\
 &= (1-\pi)Z_T \left[ \frac{\rho}{(T+1)D(C, T+1, \pi)} - \frac{1}{D(C, T, \pi)} \right] \\
 &\quad + a^{-1}\gamma \frac{D_2(C, T, \pi) - Z_T}{D(C, T+1, \pi)} - \gamma \frac{D_2(C, T, \pi)}{D(C, T, \pi)}, \\
 &< (1-\pi) \frac{Z_T}{D(C, T+1, \pi)} \left[ \frac{\rho}{T+1} - 1 \right] \\
 &\quad + \gamma \frac{a^{-1}D_2(C, T, \pi) - a^{-1}Z_T - D_2(C, T, \pi)}{D(C, T, \pi)}. \tag{A.1}
 \end{aligned}$$

Since  $\rho < (T+1)$ , the first term in the above inequality is negative and therefore we have

$$\begin{aligned}
 B_n(C, T+1, \pi) - B_n(C, T, \pi) &< \gamma \frac{a^{-1}D_2(C, T, \pi) - a^{-1}Z_T - D_2(C, T, \pi)}{D(C, T, \pi)} \\
 &< \gamma \frac{D_2(C, T, \pi)(a^{-1} - 1) - a^{-1}Z_T}{D(C, T, \pi)} \\
 &< \gamma \frac{D_2(C, T, \pi)(1 - a) - Z_T}{aD(C, T, \pi)}. \tag{A.2}
 \end{aligned}$$

Since  $\lambda_n/\lambda < 1/(T+1)$ , we have

$$\begin{aligned}
 (\alpha^{-1} - 1) &= \frac{\lambda_n}{\lambda_h} < \frac{1}{T+1}, \\
 &< \frac{1}{1 + \left(\frac{\rho\alpha}{T+1}\right) + \dots + \left(\frac{\rho\alpha}{T+1}\right)^{C-T-1}}.
 \end{aligned}$$

Since  $\rho < (T+1)$ , the above inequality becomes

$$\alpha^{-1} - 1 < \frac{1}{1 + \left(\frac{\rho\alpha}{T+1}\right) + \dots + \frac{(\rho\alpha)^{T+1}}{(T+1) \times \dots \times C}},$$

$$\begin{aligned}
&= \frac{\frac{(\rho\alpha)^T}{T!}}{\sum_{n=T+1}^C \frac{(\rho\alpha)^n}{n!}}, \\
&< \frac{\alpha^{-(T+1)} \frac{(\rho\alpha)^T}{T!}}{\alpha^{-(T+1)} \sum_{n=T+1}^C \frac{(\rho\alpha)^n}{n!}}, \\
&< \alpha^{-1} \frac{Z_T}{D_2(C, T, \pi)}.
\end{aligned}$$

The last inequality implies that

$$\frac{1-a}{a} < \frac{1}{a} \frac{Z_T}{D_2(C, T, \pi)}, \quad (\text{A.3})$$

and hence we have

$$(1-a)D_2(C, T, \pi) - Z_T < 0. \quad (\text{A.4})$$

Using the above inequality and inequality (A.1), we conclude that  $B_n(C, T+1, \pi) - B_n(C, T, \pi) < 0$  and hence  $B_n(C, T, \pi)$  is a decreasing function of  $T$ .

In the second part of the proof, we show that  $B_n(C, T, \pi)$  is a monotonically decreasing function of  $\pi$ . Differentiating  $B_n(C, T, \pi)$  with respect to  $\pi$ , we obtain

$$\begin{aligned}
\frac{\partial B_n(C, T, \pi)}{\partial \pi} &= \frac{D_1(C, T, \pi) [D_2(C, T, \pi) - Z_T] - D_2(C, T, \pi) [aD_1(C, T, \pi) + Z_T]}{[D_1(C, T, \pi) + \gamma D_2(C, T, \pi)]^2}, \\
&< \frac{D_1(C, T, \pi)}{[D_1(C, T, \pi) + \gamma D_2(C, T, \pi)]^2} [(1-a)D_2(C, T, \pi) - Z_T]. \quad (\text{A.5})
\end{aligned}$$

By using the above inequality given in (A.4) and (A.5), we can conclude that  $\frac{\partial B_n(C, T, \pi)}{\partial \pi} < 0$  and hence  $B_n(C, T, \pi)$  is a monotonically decreasing function of  $\pi$ .

## References

1. H. Beigy and M. R. Meybodi, User based call admission control policies for cellular mobile systems: a survey, *CSI Journal on Computer Science and Engineering* **1** (2003) 45–58.
2. R. Ramjee, D. Towsley, and R. Nagarajan, On optimal call admission control in cellular networks, *Wireless Networks* **3** (1997) 29–41.
3. H. Beigy and M. R. Meybodi, A new fractional channel policy, *Journal of High Speed Networks* **13** (2004) 25–36.
4. Y. Fang and Y. Zhang, Call admission control schemes and performance analysis in wireless mobile networks, *IEEE Trans. Vehicular Technology* **51** (2002) 371–382.
5. H. Beigy and M. R. Meybodi, Adaptive uniform fractional channel algorithms, *Iranian Journal of Electrical and Computer Engineering* **3** (2004) 47–53 (In Persian).
6. D. Hong and S. Rappaport, Traffic modelling and performance analysis for cellular mobile radio telephone systems with prioritized and nonprioritized handoffs procedure, *IEEE Trans. Vehicular Technology* **35** (1986) 77–92.

7. S. Oh and D. Tcha, Prioritized channel assignment in a cellular radio network, *IEEE Trans. Communications* **40** (1992) 1259–1269.
8. G. Haring, R. Marie, R. Puigjaner, and K. Trivedi, Loss formulas and their application to optimization for cellular networks, *IEEE Trans. Vehicular Technology* **50** (2001) 664–673.
9. H. Beigy and M. R. Meybodi, A general call admission policy for next generation wireless networks, *Computer Communications* **28** (2005) 1798–1813.
10. H. Beigy and M. R. Meybodi, A learning automata based dynamic guard channel scheme, *Lecture Notes in Computer Science* **2510** (2002) 643–650.
11. H. Beigy and M. R. Meybodi, An adaptive algorithm based on learning automata for determination of number of guard channels, *Proc. Ninth Int. Symp. Wireless Systems and Networks (ISWSN'03)*, Dhahran, Saudi Arabia, Mar. 2003, pp. 177–181.
12. H. Beigy and M. R. Meybodi, An adaptive call admission algorithm for cellular networks, *Electrical and Computer Engineering* **31** (2005) 132–151.
13. C. W. Leong and W. Zhuang, Call admission control for voice and data traffic in wireless communications, *Computer Communications* **25** (2002) 972–979.
14. R. Guern, Queuing–blocking system with two arrival streams and guard channels, *IEEE Trans. Communications* **36** (1988) 153–163.
15. C. H. Yoon and C. Kwan, Performance of personal portable radio telephone systems with and without guard channels, *IEEE Journal on Selected Areas in Communications* **11** (1993) 911–917.
16. X. Tian and C. Ji, Bounding the performance of dynamic channel allocation with QoS provisioning for distributed admission control in wireless networks, *IEEE Trans. Vehicular Technology* **50** (2001) 388–397.
17. P. Agrawal, D. K. Anvekar, and B. Naredran, Channel management policies for handovers in cellular networks, *Bell Labs Technical Journals* (1996) 96–110.
18. C. Chang, C. J. Chang, and K. R. Lo, Analysis of hierarchical cellular systems with renegeing and dropping for waiting new calls and handoff calls, *IEEE Trans. Vehicular Technology* **48** (1999) 1080–1091.
19. O. Yu and V. Leung, Self-tuning prioritized call handling mechanism with dynamic guard channels for mobile cellular systems, *Proc. IEEE Vehicular Technology Conference*, Apr. 1996, pp. 1520–1524.
20. J. M. Peha and A. Sutivong, Admission control algorithms for cellular systems, *Wireless Networks* **7** (2001) 117–125.
21. H. Beigy and M. R. Meybodi, Learning automata based dynamic guard channel algorithms, *Technical Reports, Department of computer Engineering, Sharif University of Technology, Tehran, Iran*.
22. H. Beigy and M. R. Meybodi, Asynchronous cellular learning automata, *Automatica* **44** (2008) 1350–1357.
23. M. A. L. Thathachar and P. S. Sastry, Varieties of learning automata: an overview, *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics* **32** (2002) 711–722.
24. M. A. L. Thathachar and K. R. Ramakrishnan, A hierarchical system of learning automata, *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-11** (1981) 236–248.
25. B. J. Oommen and E. Hansen, The asymptotic optimality of discretized linear reward-inaction learning automata, *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-14** (1984) 542–545.

26. M. A. L. Thathachar and P. S. Sastry, A New approach to the design of reinforcement schemes for learning automata, *IEEE Transactions on Systems, Man, and Cybernetics SMC-15* (1985) 168–175.
27. G. I. Papadimitriou, A new approach to the design of reinforcement schemes for learning automata: stochastic estimator learning algorithm, *IEEE Trans. Knowledge and Data Engineering* **6** (1994) 649–654.
28. J. K. Lanctôt and B. J. Oommen, Discretized estimator learning automata, *IEEE Trans. Systems, Man, and Cybernetics* **22** (1992) 1473–1483.
29. B. J. Oommen and J. K. Lanctot, Discretized pursuit learning automata, *IEEE Trans. Systems, Man, and Cybernetics* **20** (1990) 931–938.
30. G. I. Papadimitriou, Hierarchical pursuit nonlinear automata with rapid convergence and high accuracy, *IEEE Transactions on Knowledge and Data Engineering* **6** (1994) 654–659.
31. B. J. Oommen and M. Agache, Continuous and discretized pursuit learning schemes: various algorithms and their comparison, *IEEE Trans. Systems, Man, and Cybernetics-Part B: Cybernetics* **31** (2001) 277–287.
32. M. Agache and B. J. Oommen, Generalized pursuit learning schemes: new families of continuous and discretized learning automata, *IEEE Trans. Systems, Man, and Cybernetics-Part B: Cybernetics* **32** (2002) 738–749.
33. G. Santharam, P. S. Sastry, and M. A. L. Thathachar, Continuous action set learning automata for stochastic optimization, *Journal of Franklin Institute* **331B** (1994) 607–628.
34. H. Beigy and M. R. Meybodi, A new continuous action-set learning automaton for function optimization, *Journal of Franklin Institue* **343** (2006) 27–47.
35. M. N. Howell, G. P. Frost, T. J. Gordon, and Q. H. Wu, Continuous action reinforcement learning applied to vehicle suspension control, *Mechatronics* **7** (1997) 263–276.
36. O. V. Nedzelnitsky and K. S. Narendra, Nonstationary models of learning automata routing in data communication networks, *IEEE Trans. Systems, Man, and Cybernetics SMC-17* (1987) 1004–1015.
37. M. S. Obaidat, G. I. Papadimitriou, A. S. Pomportsis, and H. S. Laskaridis, Learning automata-based bus arbitration for shared-medium ATM switches, *IEEE Trans. Systems, Man, and Cybernetics-Part B: Cybernetics* **32** (2002) 815–820.
38. G. I. Papadimitriou, M. S. Obaidat, and A. S. Pomportsis, On the use of learning automata in the control of broadcast networks: a methodology, *IEEE Trans. Systems, Man, and Cybernetics-Part B: Cybernetics* **32** (2002) 781–790.
39. B. J. Oommen and E. V. de St. Croix, Graph partitioning using learning automata, *IEEE Trans. Computers* **45** (1996) 195–208.
40. M. R. Meybodi and H. Beigy, Utilizing distributed learning automata to solve stochastic shortest path problems, *Int. J. Uncertainty, Fuzziness and Knowledge-Based Systems* **14** (2006) 591–615.
41. H. Beigy and M.R. Meybodi, Solving the graph isomorphism problem using learning automata, *Proc. 5th Annual Int. Computer Society of Iran Computer Conference (CSICC-2000)*, Tehran, Iran, Jan. 2000, pp.402–415.
42. B. J. Oommen and T. D. Roberts, Continuous learning automata solutions to the capacity assignment problem, *IEEE Trans. Computers* **49** (2000) 608–620.
43. B. J. Oommen and T. D. Roberts, Discretized learning automata solutions to the capacity assignment problem for prioritized networks, *IEEE Trans. Systems, Man, and Cybernetics-Part B: Cybernetics* **32** (2002) 821–831.



44. M. R. Meybodi and H. Beigy, Neural network engineering using learning automata: determining of desired size of three layer feedforward neural networks, *Journal of Faculty of Engineering* **34** (2001) 1–26 (In Persian).
45. H. Beigy and M. R. Meybodi, Backpropagation algorithm adaptation parameters using learning automata, *Int. J. Neural Systems* **11** (2001) 219–228.
46. M. R. Meybodi and H. Beigy, New learning automata based algorithms for adaptation of backpropagation algorithm parameters, *Int. J. Neural Systems* **12** (2002) 45–68.
47. M. R. Meybodi and H. Beigy, A note on learning automata based schemes for adaptation of BP parameters, *Journal of Neurocomputing* **48** (2002) 957–974.