Home

An Adaptive Algorithm Based on Learning Automata for Determination of Number of Guard Channels

Hamid Beigy and M. R. Meybodi

Soft Computing Laboratory
Computer Engineering Department
Amirkabir University of Technology
Tehran, Iran
{beigy, meybodi}@ce.aut.ac.ir

Abstract - In this paper, we introduce an adaptive algorithm based on learning automata for determination of number of guard channels. This algorithm adapts the number of guard channels in a cell dynamically based on the current estimate of dropping probability of handoff calls. The proposed algorithm minimizes the blocking probability of new calls subject to the constraint on the dropping probability of handoff calls. In the proposed policy, a learning automaton is used to find the optimal number of guard channels. The proposed algorithm does not need any a priori information about the input traffic of the cellular network. The simulation results show that the performance of this algorithm is close to the performance of guard channel policy under the low handoff traffic conditions for which we need to know all traffic parameters in advance.

I. INTRODUCTION

In the last decade, there is an increase in the popularity of mobile computing systems, which results an increase for channel demands. Since number of allocated channels for this purpose is limited, the cellular and micro cellular networks are introduced, in which the service area is partitioned into regions called cells. Introduction of micro cellular networks leads to improvement of network capacity but increases the expected rate of handoff. When a mobile host moves across the cell boundary, handoff is required. If an idle channel is available in the destination cell, then the handoff call is resumed; otherwise the handoff call is dropped. The dropping probability of handoff calls (B_h) and the blocking probability of new calls (B_n) are important quality of service (QoS) measures of the cellular networks. Since the disconnection in the middle of a call is highly undesirable, dropping of handoff calls is more serious than blocking of new calls. In order to control the dropping probability of handoff calls and the blocking probability of new calls, the *call admission* policies are introduced. The call admission policies determine whether a new call should be admitted or blocked. Both blocking probability of new calls (B_n) and dropping probability of handoff calls (B_h) are affected by call admission control policies. Blocking more new calls generally improves dropping probability of handoff calls and admitting more new calls generally improves blocking probability of new calls.

The simplest call admission policy is called *guard channel* (GC) policy [1]. Suppose that the given cell has C full duplex channels. The guard channel policy reserves a subset of channels allocated to a cell for sole use of handoff calls (say $C-T$ channels). These channels are called *guard channels*. Whenever the channel occupancy exceeds the certain threshold T, the guard channel policy rejects new calls until the channel occupancy goes below the threshold. The guard channel policy accepts handoff calls as long as channels are available. The description of guard channel policy is given algorithmically in figure 1.

```
if (HANDOFF CALL) then
    if $c(t) < C$ ) then
        accept call
    else
        reject call
    end if
end if

if (NEW CALL) then
    if ($c(t) < C - T$) then
        accept call
    else
        reject call
    end if
end if
```

Fig. 1. Guard channel policy

As the number of guard channels increased, the dropping probability of handoff calls will be reduced while the blocking probability of new calls will be increased [2]. It has been shown that there is an optimal threshold T^* in which the blocking probability of new calls is minimized subject to the hard constraint on the dropping probability of handoff calls [3]. Algorithms for finding the optimal number of guard channels are given in [3, 4]. The GC policy reserves an integral number of guard channels for handoff calls. If the parameter B_h is considered, the guard channel policy gives very good performance, but the parameter B_n is degraded to great extent. In order to have more control on blocking probability of new calls and the dropping probability of handoff calls, the *limited fractional guard channel policy* (LFG) is introduced [3]. In LFG policy, when the system is in state T, new calls are accepted with probability π. From states $T+1$ to C, only handoff calls are accepted and in states 0 to $T-1$, both types of calls are accepted. The description of limited fractional guard channel policy is given algorithmically in figure 2.

```
if (HANDOFF CALL) then
    if $c(t) < C$ ) then
        accept call
    else
        reject call
    end if
end if

if (NEW CALL) then
    if ($c(t) < C - T$ and $rand(0,1) \leq \pi$) then
        accept call
    else
        reject call
    end if
end if
```

Fig. 2. Limited fractional guard channel policy

It has been shown that there is an optimal threshold T^* and an optimal value of π^* for which the blocking probability of new calls is minimized subject to the hard constraint on the dropping probability of handoff calls [3]. The algorithm for finding such optimal parameters is given in [3]. These algorithms assume that the input traffic is a stationary process with known parameters. Since the input traffic is not a stationary process and its parameters are unknown a priori, the optimal number of guard channels is different for different traffic. In such cases the *dynamic guard channel* policy can be used. In dynamic guard channel policy, the number of guard channels varies during the operation of the cellular network.

In [5], a dynamic guard channel algorithm is proposed in which the number of guard channels in any particular cell is adjusted with number of ongoing calls in neighboring cells. Since all ongoing calls in neighboring cells are potential to handoff, the number of these ongoing calls determines a current estimate of handoff. In this algorithm, when a new or handoff call arrives at a neighboring cell, the number of guard channels is increased by a fractional amount and when a cell is completed or handovers to non-neighboring cells, the number of guard channels is decreased with the same fractional amount. This algorithm must have an up to date status of neighboring cells. The transmission of cell's status leads to loss of some bandwidth allocated to the user traffic on the wired-line network. In order to attain reasonable bandwidth, the call admission control algorithm must use less status information.

Learning automaton have been used successfully in many applications such as telephone and data network routing [6, 7], solving NP-Complete problems [8, 9], capacity assignment [10] and neural network engineering [11, 12, 13, 14] to mention a few. In this paper, we propose an adaptive algorithm based on learning automata for determination of number of guard channels. This algorithm uses only the current channel occupancy of the given cell and dynamically adjusts the number of guard channels. The proposed algorithm minimizes the blocking probability of new calls 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 algorithm does not need any a priori information about input traffic. One of the most important advantage of the proposed algorithm is that no status information will be exchanged between neighboring cells. The exchange of such status information increase the performance of the proposed algorithm. The simulation results show that the performance of this algorithm are near to performance of guard channel policy that knows all traffic parameters.

The rest of this paper is organized as follows: The learning automata briefly is given in section 2. The proposed learning automata based algorithm for determination of number of guard channels is presented in section 3. The computer simulations is given in section 4 and section 5 concludes the paper.

II. LEARNING AUTOMATA

The automata approach to learning involves the determination of an optimal action from a set of allowable actions. An automaton can be regarded as an abstract object that has finite number of actions. It selects an action from its finite set of actions. This action is applied to a random environment. Then the environment evaluates the applied action and supplies a grade to action of automata. The response from environment (i.e. grade of action) is used by automaton to select its next action. By continuing this process, the automaton learns to se-

Top

lect an action with best grade. The *learning algorithm* is used by automaton to determine the selection of its next action from the response of environment. The interaction of an automata with its environment is shown in figure 3.

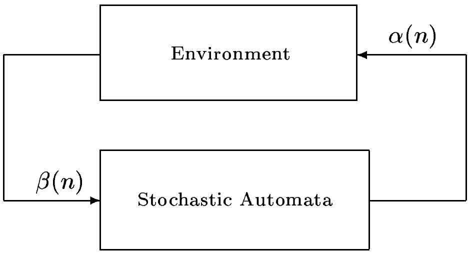

Fig. 3. The interaction of automata and its environment

An automaton acting in an unknown random environment and improves its performance in some specified manner, is referred to as *learning automata* [15]. Environment refers to the aggregate of all external conditions and influences affecting the life and development of an organism. The mathematical model of a random environment is described by triple $E \equiv \{\underline{\alpha}, \underline{\beta}, \underline{S}\}$, where $\underline{\alpha} = \{\alpha_1, \alpha_2, \cdots, \alpha_r\}$ with $2 \leq r < \infty$ shows a finite set of inputs applied to the environment, $\underline{\beta} = \{\beta_1, \beta_2, \cdots, \beta_m\}$ or $\underline{\beta} = \{[a, b]\}$ represents the set of outputs of environment, and $\underline{S} = \{s_1, s_2, \cdots, s_r\}$ denotes the set of *penalty strengths*, where s_k corresponds to the input α_k. The input and output of the environment at discrete time n (for $n = 0, 1, 2, \cdots$) are shown by $\alpha(n)$ and $\beta(n)$, respectively, whereas $\beta(n)$ is in the interval $[0, 1]$ [15]. The values of s_k (for $k = 1, 2, \cdots, r$) are unknown. Note that the $\{s_i\}$ are unknown initially and it is desired that as a result of the interaction between the automaton and the environment arrives at the action which presents it with the minimum penalty response in an expected sense.

The random environment can be classified in various ways depending on the nature of the vectors $\underline{S}$ and vector $\underline{\beta}$. According to the nature of the set $\underline{S}$, the random environment could be classified into two groups: stationary and non-stationary environments. Based on the nature of the set $\underline{\beta}$, the random environment could be classified in three classes: P-, Q-, and S-model environments.

The output of P-model environment has the form of $\underline{\beta} = \{\beta_1, \beta_2\}$. This model of environment evaluates the action of automata as *success* or *failure*. In Q-model environments, the output $\beta(n)$ can take a finite number of values in the interval $[0, 1]$. In Q-model environment, response of the environment to the input α_i is in the form of $\bar{\beta}^i_1 < \bar{\beta}^i_2 < \cdots < \bar{\beta}^i_{m_i}$. In S-model environment, response $\bar{\beta}^i$ of environment to its input α_i lies in the interval $[a, b]$.

Learning automata can be classified into two main families: *fixed structure learning automata* and *variable structure learning automata* [15]. Variable structure learning automata is represented by triple $< \beta, \alpha, T >$, where β is a set of inputs actions, α is a set of actions, and T is learning algorithm. The learning algorithm is a recurrence relation and is used to modify the state probability vector. It is evident that the crucial factor affecting the performance of the variable structure learning automata, is learning algorithm. Various learning algorithms have been reported in the literature. Let α_i be the action chosen at time k as a sample realization from probability distribution $p(k)$. The linear reward-inaction algorithm (L_{R-I}) in P-model environments is one of the earliest learning schemes and its recurrence equation for updating action probability vector p is defined as

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

if $\beta(k)$ is zero and p is unchanged if $\beta(k)$ is one. The parameter $0 < a < 1$ is called *step length* and determines the amount of increases (decreases) of the action probabilities. It has been shown that the L_{R-I} learning algorithm is $\epsilon-$optimal.

The L_{R-I} scheme can be extended to Q- and S-model of environments. The Q- and S-model versions of L_{R-I} scheme are denoted by SL_{R-I}[1]. The SL_{R-I} scheme can be expressed as

$$p_j(n+1) = \begin{cases} p_j(n) - a[1 - \beta(n)]p_j(n) & j \neq i \\ \\ p_j(n) + a[1 - \beta(n)] \sum_{k \neq i} p_k(n) & j = i \end{cases} \qquad (2)$$

where the action α_i is selected at instant of n. The SL_{R-I} scheme shares many properties of the L_{R-I} scheme, such as *absolutely expediency* and $\epsilon-$*optimality* [16].

III. The Proposed Algorithm

In this section, we introduce a new learning automata based algorithm (figure 4) for determination of number of guard channels when the parameters λ_n, λ_h, and μ are unknown and possibly time varying. Assume that the cell has C full duplex channels. Let the number of guard channels at time instant t denoted by $g(t)$ is in

[1] The prefix S in reinforcement schemes, such as in SL_{R-I}, is used to denote the Q- and S-model versions derived from P-model schemes.

Top

interval $g(t) \in [g_{\min}, g_{\max}]$, where $0 \leq g_{\min} \leq g_{\max} \leq C$. In the proposed algorithm, each base station has a learning automaton A with $g_{\max} - g_{\min} + 1$ actions, where action α_i denotes that the base station must use $g(t) = g_{\min} + \alpha_i - 1$ guard channels. The proposed algorithm can be described as follows. When a handoff call arrives at the given cell and a channel is available, then the call is accepted; otherwise it is dropped. When a new call arrives at the given cell, learning automaton associated to the cell selects one of its actions, say α_i. If the cell has at least $g_{\min} + \alpha_i - 1$ free channels, then the incoming call is accepted; otherwise it is blocked. Then the base station computes the current estimate of dropping probability of handoff calls ($\hat{B}_h$) and then based on this estimation, the reinforcement signal β is computed. The reinforcement signal is computed by the following expression.

$$\beta = \Psi\left(w\frac{\hat{B}_h}{p_h}\right) \tag{3}$$

where w is a scaling constant and function $\Psi(x) : x \rightarrow [0,1]$ is a project function, which maps x into interval $[0,1]$. It is evident when $\hat{B}_h$ is greater than p_h, then reinforcement signal β is large and near to one and hence the selected action will be penalized. It is evident when $\hat{B}_h$ is smaller than p_h, then reinforcement signal β is small and near to zero and hence the selected action will be rewarded.

```
if (HANDOFF CALL) then
    if $c(t) < C$ ) then
        accept call
    else
        reject call
    end if
end if

if (NEW CALL) then
    set $g \leftarrow$ LA.action ()
    if ($c(t) < C - g$ )then
        accept call
    else
        reject call
    end if
    set $\beta \leftarrow w\frac{\hat{B}_h}{p_h}$
    Update probability vector $p$ using
    $\beta$ and equation (*).
end if
```

Fig. 4. Learning automata based algorithm for determination of number of guard channels

The proposed algorithm requires less resources (bandwidth of the wired-line network) than the algorithm given in [5] for which the status of all neighboring cells are needed for determination of guard channels. In [5], status information must be exchanged between neighboring cells in the case of arrival of a call, departure of a call, and handoff of a call. However, the exchange of status information can be used to sped up the convergence of the proposed algorithm, which results an improvement of the proposed algorithm. Since the learning automata begin their learning without a priori knowledge about its environment, the proposed algorithm does not require any information about input traffic. Even though the priori information about input traffic is not needed by the algorithm, availability of such information may be used to find a better learning algorithm in order to choose a better learning algorithm for adaptation of traffic parameters. The use of a priori information in the proposed algorithm needs to be investigated. The proposed algorithm at the beginning does not perform well but as it proceeds, the performance of the algorithm approaches to its optimal performance. Initially, the proposed guard channels randomly.

IV. SIMULATION RESULTS

In this section, we compare performance of the guard channel [1], the limited fractional guard channel [3], and the dynamic guard channel algorithms proposed in this paper. The results of simulations are summarized in table 1. The simulation is based on the single cell of homogenous cellular network system. In such network, each cell has 8 full duplex channels ($C = 8$). In the simulations, new call arrival rate is fixed to 30 calls per minute ($\lambda_n = 30$), channel holding time is set to 6 seconds ($\mu^{-1} = 6$), and the handoff call traffic is varied between 2 calls per minute to 20 calls per minute. The results listed in table 1 are obtained by averaging 10 runs from 2,000,000 seconds simulation of each algorithm. The objective is to minimize the blocking probability of new calls subject to the constraint that the dropping probability of handoff calls is less than 0.01. The optimal number of guard channels for guard channel policy is obtained by algorithm given in [4] and the optimal parameters of limited fractional guard channel policy is obtained by algorithm given in [3].

Table 1. Comparison of guard channel policy and the proposed learning automata based algorithm

		Guard channel policy		The proposed algorithm	
Case	λ_h	B_n	B_h	B_n	B_h
1	2	0.063507	0.001525	0.065927	0.010747
2	4	0.077080	0.003538	0.095552	0.011314
3	6	0.091013	0.005923	0.127434	0.011180
4	8	0.105002	0.008380	0.150670	0.011930
5	10	0.120260	0.011877	0.191515	0.011240
6	12	0.231559	0.004309	0.209537	0.014020
7	14	0.255346	0.005975	0.227480	0.017515
8	16	0.275489	0.007999	0.243930	0.021078
9	18	0.296834	0.010518	0.259754	0.025059
10	20	0.459183	0.006081	0.278590	0.029722

Top

By inspecting table 1, it is evident that the performance of the proposed learning automata based algorithm for determination of number of guard channels is close to the performance of guard channel policy under the low handoff/new traffic ratio. One reason for the difference in performances of the guard channel policy and the proposed policy is due to the fact that transient behavior of the proposed algorithm. Since, the performance parameters (the blocking probability of new calls and the dropping probability of handoff calls) in the early stages of simulation are far from their desire value, they affect the long-time calculation of the performance parameters. However, such effect can be removed by excluding the transient behaviors of the proposed algorithm. For more experimentation refer to [17].

V. CONCLUSIONS

In this paper, we introduced an adaptive algorithm based on learning automata for determination of number of guard channels. The proposed algorithm adapts the number of guard channels in a cell using current estimate of dropping probability of handoff calls. This algorithm minimizes the blocking probability of new calls subject to the constraint on the dropping probability of handoff calls. The simulation results show that the performance of this algorithm is very close to the performance of guard channel policy that knows all traffic parameters in advance under the low handoff/new traffic ratio. The proposed policy has three advantages: 1) does not not require any exchange of information between the neighboring cells leading to less network overheads. 2) does not need any a priori information about the input traffic. 3) the algorithms works for time varying traffics.

References

[1] D. Hong and S. Rappaport, "Traffic Modelling and Performance Analysis for Cellular Mobile Radio Telephone Systems with Priotrized and Nonpriotorized Handoffs Procedure," *IEEE Transactions on Vehicular Technology*, vol. 35, pp. 77–92, Aug. 1986.

[2] S. Oh and D. Tcha, "Priotrized Channel Assignment in a Cellular Radio Network," *IEEE Transactions on Communications*, vol. 40, pp. 1259–1269, July 1992.

[3] R. Ramjee, D. Towsley, and R. Nagarajan, "On Optimal Call Admission Control in Cellular Networks," *Wireless Networks*, vol. 3, pp. 29–41, 1997.

[4] G. Haring, R. Marie, R. Puigjaner, and K. Trivedi, "Loss Formulas and Their Application to Optimization for Cellular Networks," *IEEE Transactions on Vehicular Technology*, vol. 50, pp. 664–673, May 2001.

[5] O. Yu and V. Leung, "Self-Tuning Prioritized Call Handling Mechanism with Dynamic Guard Channels for Mobile Cellular Systems," in *Proceedings of IEEE Vehicular Technology Conference*, pp. 1520–1524, Apr. 1996.

[6] P. R. Srikantakumar and K. S. Narendra, "A Learning Model for Routing in Telephone Networks," *SIAM Journal of Control and Optimization*, vol. 20, pp. 34–57, Jan. 1982.

[7] O. V. Nedzelnitsky and K. S. Narendra, "Nonstationary Models of Learning Automata Routing in Data Communication Networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-17, pp. 1004–1015, Nov. 1987.

[8] B. J. Oommen and E. V. de St. Croix, "Graph Partitioning Using Learning Automata," *IEEE Transactions on Commputers*, vol. 45, pp. 195–208, Feb. 1996.

[9] H. Beigy and M. R. Meybodi, "Solving the Graph Isomorphism Problem Using Learning Automata," in *Proceedings of 5th Annual International Computer Society of Iran Computer Conference, CISCC-2000, Tehran, Iran*, pp. 402–415, Jan. 2000.

[10] B. J. Oommen and T. D. Roberts, "Continuous Learning Automata Solutions to the Capacity Assignment Problem," *IEEE Transactions on Commputers*, vol. 49, pp. 608–620, June 2000.

[11] 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*, vol. 34, pp. 1–26, May 2001.

[12] M. R. Meybodi and H. Beigy, "A Note on Learning Automata Based Schemes for Adaptation of BP Parameters ," *Journal of Neuro Computing*, vol. 48, pp. 957–974, Nov. 2002.

[13] M. R. Meybodi and H. Beigy, "New Class of Learning Automata Based Schemes for Adaptation of Backpropagation Algorithm Parameters," *International Journal of Neural Systems*, vol. 12, pp. 45–68, Feb. 2002.

[14] H. Beigy and M. R. Meybodi, "Backpropagation Algorithm Adaptation Parameters using Learning Automata," *International Journal of Neural Systems*, vol. 11, no. 3, pp. 219–228, 2001.

[15] K. S. Narendra and K. S. Thathachar, *Learning Automata: An Introduction.* New York: Printice-Hall, 1989.

[16] N. Baba, "The Absolutely Expedient Nonlinear Reinforcement Schemes Under the Unknown Multiteacher Environment," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, pp. 580–582, Aug. 1976.

[17] H. Beigy and M. R. Meybodi, "Dynamic Guard Channel scheme Using Learning Automata," Tech. Rep. TR-CE-2002-001, Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran, 2002.