Dynamic Guard Channel Scheme Using Learning Automata

Hamid Beigy
Soft Computing Laboratory
Computer Engineering Department
Amirkabir University of Technology
Tehran, Iran
beigy@ce.aku.ac.ir

and

M. R. Meybodi
Soft Computing Laboratory
Computer Engineering Department
Amirkabir University of Technology
Tehran, Iran
meybodi@ce.aku.ac.ir

ABSTRACT

In this paper, we introduce a new dynamic guard channel policy, which 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 for which we need to know all traffic parameters in advance. Two advantages of the proposed policy are that it is fully autonomous and adaptive.

Keywords : Cellular Networks, Call Admission Control, Dynamic Guard Channel Policy, Learning Automata

1. INTRODUCTION

With increasing popularity of mobile computing, demand for channels is on the rise. 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 control* (CAC) 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 CAC policy is called *guard channel* (GC) policy [**?**]. 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. 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 [?]. 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 [?]. Algorithms for finding the optimal number of guard channels are given in [?, ?]. 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 [?]. 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 [?]. The algorithm for finding such optimal parameters is given in [?]. 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 [?], 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 is a reinforcement learning technique and has been used successfully in many applications such as telephone and data network routing [?, ?], solving NP-Complete problems [?, ?, ?, ?, ?, ?] and capacity assignment [?], to mention a few. In this paper, we propose an adaptive and autonomous call admission control algorithm, which uses learning automata. 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 dynamic guard channel policy is presented in section 3. The computer simulations is given in section 4 and section 5 concludes the paper.

2. 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. The random environment evaluates the applied action and gives a grade to the selected action of automata. The response from environment (i.e. grade of action) is used by automata to select its next action. By continuing this process, the automata learns to select an action with best grade. The learning algorithm used by automata to determine the selection of next action from the response of environment. An automaton acting in an unknown random environment and improves its performance in some specified manner, is referred to as *learning automata* (LA). Learning automata can be classified into two main families: *fixed structure learning automata* and *variable structure learning automata* [?].

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 learn-

ing 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}) is one of the earliest learning schemes and its recurrence equation for updating action probability vector p is defined as

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

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} learning algorithm has a relatively slow rate of convergence for many real applications. In order to increase the speed of convergence of L_{R-I} algorithm, the discretized learning automata was introduced in [**?**]. In this approach, the probability of selecting an action is restricted to a finite number of values in the interval [0,1]. In order to design faster learning algorithms, estimator algorithms are introduced [**?**]. Estimator algorithms maintain running estimates for the reward probability of actions, and use them for updating the action probability vector. In estimator algorithms, the automaton chooses an action and the environment emits a response to this action. Based on this response, the estimator algorithm updates the running estimate of the reward probabilities for that action. The action probability vector is updated based on both the running estimates of the reward probabilities, and on the response of the environment. The estimator algorithms need a large amount of memory to keep the running estimate of the reward probabilities. In order to decrease memory requirement of estimator algorithms, Pursuit algorithms are introduced [**?**]. Pursuit algorithms are a subset of the estimator algorithms. In Pursuit algorithms, the action probability vector pursues the action that is currently estimated to be the optimal. The Pursuit algorithms increase the probability of the action whose current estimate of being rewarded is maximal based on only the running estimates [**?**] or both the running estimates and the current response of the environment [**?**]. This implies that if, at any time, the action with maximum estimate of reward probability is not the optimal action, then the automaton pursues a wrong action. In order to minimize the probability of pursuing a wrong action, generalization of Pursuit algorithms is introduced [**?**]. This algorithm pursues a set of actions. It has been shown that the Pursuit algorithms are $\epsilon-$optimal.

3. LEARNING AUTOMATA BASED DYNAMIC GUARD CHANNEL POLICY

In this section, we introduce a new learning automata based algorithm (figure **??**) to determine the 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 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 learning automata based guard channel 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 compare this quantity with the specified level of QoS (p_h). If the incoming new call is accepted and the current value of ($\hat{B}_h$) is less than p_h then action α_i is rewarded; otherwise action α_i is penalized. If the incoming new call is blocked and the current value of ($\hat{B}_h$) is greater than p_h then the action α_i is rewarded; otherwise the action α_i is penalized. The comparison of current estimate of dropping probability of handoff calls and the specified level of QoS (p_h) is done to guarantee the specific level of QoS.

The proposed algorithm requires less resources (bandwidth of the wired-line network) than the algorithm given in [**?**] for which the status of all neighboring cells are needed for determination of guard channels. In [**?**], 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.

```
if (NEW CALL) then
    set g ← LA.action ()
    if (c(t) < C − g )then
        accept call
        if (B̂_h < p_h ) then
            reward action g
        else
            penalize action g
        end if
    else
        reject call
        if (B̂_h < p_h ) then
            penalize action g
        else
            reward action g
        end if
    end if
end if
```

Fig. 1. LA based dynamic guard channel algorithm

4. SIMULATION RESULTS

In this section, we compare performance of the guard channel [**?**], the limited fractional guard channel [**?**], and the dynamic guard channel algorithms proposed in this paper. The results of simulations are summarized in table **??**. 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 **??** 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 [**?**] and the optimal parameters of limited fractional guard channel policy is obtained by algorithm given in [**?**].

By inspecting table **??**, it is evident that the performance of dynamic guard channel policy is close to the performance of guard channel policy. 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, which is shown in figure **??**. Figure **??** shows the evolution of the performance parameters for the guard channel policy and the proposed dynamic guard channel policy. The traffic parameters used for figure **??** corresponds to case 10 in table **??**. By carefully inspecting figure **??** and ignoring the transient behavior of the proposed algorithm, it can be concluded that the drooping probability of handoff calls approaches its prescribed value (p_h), while the blocking probability of new calls is less than the corresponding performance parameter of guard channel policy. For more experimentation refer to [**?**].

5. CONCLUSIONS

In this paper, a dynamic guard channel policy based on learning automata is given. 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. 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.

Table 1. Minimize B_n such that $B_h \leq 0.01$ ($\lambda_n = 30, \mu^{-1} = 6, C = 8$)

		GC		LFG		DGC	
Case	λ_h	B_n	B_h	B_n	B_h	B_n	B_h
1	2	0.063507	0.001525	0.031609	0.023283	0.053433	0.010619
2	4	0.077080	0.003538	0.051414	0.020675	0.080966	0.010039
3	6	0.091013	0.005923	0.071632	0.018707	0.125500	0.009964
4	8	0.105002	0.008380	0.092138	0.016706	0.154861	0.010031
5	10	0.120260	0.011877	0.114445	0.015572	0.207490	0.010067
6	12	0.231559	0.004309	0.147902	0.014044	0.245842	0.010017
7	14	0.255346	0.005975	0.204217	0.012675	0.290619	0.009960
8	16	0.275489	0.007999	0.250642	0.011554	0.331478	0.009983
9	18	0.296834	0.010518	0.294441	0.010877	0.377334	0.009953
10	20	0.459183	0.006081	0.384157	0.010182	0.427894	0.010005

Fig. 2. The comparison of guard channel policy and dynamic guard channel policy ($\lambda_h = 20$)