Journal of Computer Science and Information Technology
June 2014, Vol. 2, No. 2, pp. 149-172
ISSN: 2334-2366 (Print), 2334-2374 (Online)

Published by American Research Institute for Policy Development

Channel Assignment Scheme based on Learning Automata for Clustered Wireless Ad-Hoc Networks

Shiva Shabanzadeh Dehkordi[1] and Javad Akbari Torkestani[2]

Abstract

In this paper we have design a learning automata-based scheme for medium access scheduling in clustered wireless ad-hoc networks. In this scheme, the collision-free intra-cluster communications are organized by the cluster-heads using learning automata rules. The advantage of applying learning automatons in this scheme is that each cluster-head learns the traffic parameters of its own cluster members. Each cluster-head monitors the intra-cluster transmissions and coordinates these transmissions to avoid collisions. In the proposed polling scheme, a portion of bandwidth is assigned to each cluster member Commensurate with its need such as traffic load. The simulation results show that the proposed polling scheme outperforms the existing methods in term of almost all metrics of interest.

Keywords Channel assignment; Polling; Ad-hoc networks

1. Introduction

The medium access control protocols can be divided into fixed assignment, demand-assignment, and contention-access protocols [1]. Fixed-assignment protocols are those for which, as the name implies, channel assignments are fixed, regardless of the transmission requirements.

Frequency division multiple access (FDMA) [2], code division multiple access (CDMA) [3], and time division multiple access (TDMA) [4] schemes are some fixed-assignment MAC (medium access control) layer protocols.

[1] Department of computer engineering, science and research branch, Islamic Azad University, Arak-Iran. Email: shi.sh1986@yahoo.com,
[2] Department of computer engineering, science and research branch, Islamic Azad University, Arak-Iran. Email: j-akbari@iau-arak.ac.ir

Among the controlled access MAC protocols, TDMA is the most commonly used in wireless ad-hoc networks. In TDMA scheme, a single channel is time-shared. That is, use of the channel is divided among several hosts by allowing each host to access the channel periodically, but only for a small period of time referred to as time slot [111].

Demand-assignment protocols like polling [5], trunking [6] and reservation [7] methods schedule the channel access based on the demand of the hosts for packet transmission. In both fixed-assignment and demand-assignment protocols, the medium access control algorithms are collision-free. These protocols are also referred to as contention-free protocols since the hosts do not compete to seize the channel. In contention-access (or random access) protocols, the hosts contend for channel access, and the hosts that lose it try again later. Since the collisions are not prohibited by the contention-access protocols, they require a method for detecting and recovering the collisions. ALOHA [8] and carrier sense multiple-access (CSMA) [9] are some well-known contention-based protocols. Polling medium access scheme is a demand-based access scheme in which a centralized controller asks the hosts, in a cyclic predetermined order, whether they have data to transmit or not. Due to the recent advances in communication systems, some other variations of the polling scheme have been also considered.

These variations deal with non-cyclic allocation policies, which include random, Markovian, or more generally, non-deterministic allocation policies. In a polling scheme, controller polls (one by one) the hosts to give them an opportunity to access the medium. The hosts that have no packet to be transmitted (or do not need the channel access) decline, and the other hosts begin the packet transmission upon receiving the query. In polling scheme, the centralized controller is responsible for coordinating the transmissions, and so polling is a collision-free scheme. In this scheme, the entire bandwidth is available for the host which is permitted to transmit data. Although in realistic scenarios, traffic load of the different hosts is not the same; the major drawback of the basic polling scheme is to give the same importance (or equal access to the channel) to all hosts. A prioritized polling system may provide better results. Furthermore, the polling scheme suffers from the substantial overhead caused by the large number of messages generated by the controller to query the communicating hosts. Polling is an access scheme which is based on a centralized control system.

Therefore, in ad-hoc networks, polling cannot be a practical channel assignment policy because of the lack of fixed infrastructures and centralized administrations. One of the solutions to solve the above mentioned problem and implement the polling system in ad-hoc networks is clustering. In the clustering method, the network is subdivided into several non-overlapping groups. In clustered multi-hop ad hoc networks, channel assignment problem is divided into two categories: inter-cluster and intra-cluster channel assignment problems. The first (i.e., inter-cluster) channel assignment problem is concerned with assigning an interference-free channel to each cluster, and the second one aims at distributing (proper) portions of the channel assigned to each cluster between its members. A polling scheme can be effectively used to solve the intra-cluster channel assignment problem by which the channel access requests are scheduled. In a cluster-based polling scheme, each cluster-head assumes the role of the centralized coordinator for distributing the bandwidth portions between the cluster members. Such a polling scheme guarantees a collision-free channel access within each cluster. The proposed polling scheme is based on an adaptive algorithm by which the active nodes are polled with a higher probability. This results in a higher throughput and lower packet delays.

In this paper, we propose an adaptive polling-based medium access scheme for clustered wireless ad-hoc networks with unknown traffic parameters. In this scheme, each cluster-head is equipped with a learning automaton whose action-set includes an action for each of its cluster members. Each cluster-head randomly chooses one of its actions according to its action probability vector. Then, the cluster member corresponding to the selected action is permitted to transmit its packets or in other words, the cluster selected action is polled. If the selected cluster member has a packet to transmit, the cluster-head rewards (or increases the probability of polling) the selected cluster member, and penalizes it otherwise. Indeed, exploiting the learning automaton, cluster-head prioritizes its cluster members based on the traffic load. As the proposed algorithm proceeds, the probability of polling a given host converges to the proportion of time it has a packet to be sent according to its traffic load. This probability specifies the portion of bandwidth must be assigned to the host. Through the extensive simulation experiments, the performance of the proposed polling scheme is measured and compared with basic polling scheme, and two-hop polling scheme [15] in terms of channel utilization, waiting time for packet transmission, and control overhead. The obtained results show that the proposed scheme outperforms the others, specifically, under bursty traffic conditions.

The rest of the paper is organized as follows. The next section introduces the related works in the studied field. The learning automata and variable action-set learning automata concepts are presented in section 3. Section 4 describes the proposed learning automata-based polling scheme. Section 5 shows the superiority of the proposed scheme over the existing methods through the simulation experiments, and Section 6 concludes the paper.

2. Related Works

Numerous ways have been proposed to organize the various medium access control methods. Polling systems have been extensively studied for the last three decades because of the applicability to the computer networks and communication systems. Grillo [10] provided a survey on applications of polling scheme in communication systems. Wang et al. [11], proposed an efficient distributed scheduling algorithm based on a prioritized polling policy for multi-hop wireless networks. The proposed algorithm maximizes the spatial and time reuse with an interference-based network model. Lye and Seah [12] also studied a priority-based random polling scheme. A QoS supportive adaptive polling scheme was proposed by Lagkas et al. [13], for wireless networks. In this scheme, an access point polls the wireless nodes in order to grant them permission to transmit. The polled node sends its data directly to the destination node. Yang and Liu [14], also proposed a QoS support bandwidth polling scheme called BBP. In this scheme, to allocate a proper portion of bandwidth to each node, coordinator defines a framing structure of time slots. Coordinator is allowed to poll a node more than once, and this causes it allocates a proper number of slots (or a proper bandwidth portion) to each active node.

However, in ad hoc networks, due to the lack of centralized coordination, the polling scheme has not received the attention it deserves. In [15], Dimitriadis and Pavlidou proposed a polling access scheme for clustered, multi-hop ad-hoc networks called two-hop polling (2HP). 2HP is a revised version of the polling scheme tailored for the clustered environments. The authors claim that by this scheme it is possible to utilize inter-cluster links (distributed gateways) without adding much to the complexity of polling. 2HP changes the medium access by giving more liberty to the non cluster-head hosts. In clustered networks, the hosts which belong to the different clusters must communicate through the cluster-heads. This results in many potent links between the hosts not to be used. In the proposed scheme, the members of the neighboring clusters can directly communicate by the inter-cluster connections they have between.

In [16], Tseng and Chen proposed a priority-based polling scheme with reservation for QoS guarantee in wireless ad hoc networks. The proposed scheme combines the priority-based and randomly addressed polling schemes to guarantee QoS constraints. The above mentioned polling schemes are capable of improving the channel utilization if the traffic load is fixed or stationary process with known parameters, while in realistic scenarios the input traffic parameters are unknown and possibly time varying.

3. Learning Automata

A learning automaton ([17], [48], [49], [50], [51]) is a simple adaptive decision-making unit that improves its performance by learning how to choose the optimal action through the repeated interactions with a random environment. The action is chosen at random based on a probability distribution kept over the action-set and at each instance; the given action is served as the input to the random environment. The environment responds the taken action in turn with a reinforcement signal. The action probability vector is updated based on the reinforcement feedback from the environment. The objective of a learning automaton is to find the optimal action from the action-set so that the average penalty received from the environment is minimized. Learning automata have been extensively studied for the last three decades because of the applicability of such a probabilistic learning model in computer and communication problems. The results given in references [52], [53] show that the learning automata can be effectively used for solving the intractable optimization problems. In [50], several attempts have been also made to exhibit the capabilities of the learning automata in dynamic wireless ad hoc networks.

The environment can be described by a triple $E = \{\alpha, \beta, c\}$ where $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_r\}$ represents the finite set of inputs, $\beta = \{\beta_1, \beta_2, ..., \beta_m\}$ denotes the set of the values can be taken by the reinforcement signal, and $c = \{c_1, c_2, ..., c_r\}$ denotes the set of the penalty probabilities, where the element c_i is associated with the given action α_i. if the penalty probabilities are constant, the random environment is said to be a stationary random environment, and if they vary with time, the environment is called a non-stationary environment. The environment depending on the nature of the reinforcement signal $\underline{\beta}$ can be classified into *P*-model, *Q*-model, and *S*-model.

The environments in which the reinforcement signal can only take two binary values 0 and 1 are referred to as *P*-model environments. Another class of the environment allows a finite number of the values in the interval [0, 1] can be taken by the reinforcement signal. Such an environment is referred to as *Q*-model environment. In *S*-model environments, the reinforcement signal lies in the interval [a, b]. The relationship between the learning automaton and its random environment has been shown in figure 1.

Moreover, Learning automata can be classified into two main families [17]: fixed structure learning automata and variable structure learning automata.

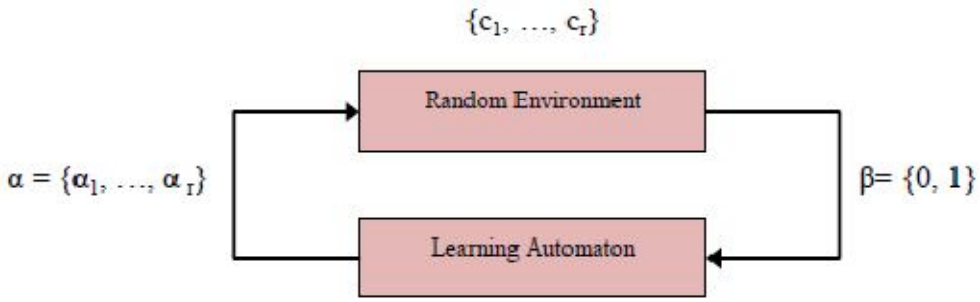

Fig. 1: The Automaton-Environment Feedback Loop

Variable structure learning automata are represented by a triple $\langle \underline{\beta}, \underline{\alpha}, T \rangle$ where $\underline{\beta}$ is the set of inputs, $\underline{\alpha}$ the set of actions, and *T* learning algorithm. The learning algorithm is a recurrence relation, which is used to modify the action probability vector. Let $\alpha(k)$ and $\underline{p}(k)$ denote the action chosen at instance *k* and the action probability vector on which the chosen action is based, respectively. The recurrence equation shown by (1) and (2) is a linear learning algorithm by which the action probability vector $\underline{p}$ is updated. Let $\alpha_i(k)$ be the action chosen by the automaton at instant *k*.

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

When the taken action is rewarded by the environment (i.e., $\beta(n)=0$) and

$$p_j(n+1)=\begin{cases}(1-b).p_j(n) & j=i\\ \left(\frac{b}{r-1}\right)+(1-b)p_j(n) & \forall j \quad j\neq i\end{cases}$$

when the taken action is penalized by the environment (i.e., $\beta(n)=1$). r is the number of actions that can be chosen by the automaton, a(k) and b(k) denote the reward and penalty parameters and determine the amount of increases and decreases of the action probabilities, respectively. If $a(k)=b(k)$, the recurrence Eqs. (1) and (2) are called linear reward–penalty (*LR-P*) algorithm, if $a(k)>>b(k)$ the given equations are called linear reward–ε penalty (*LR-εP*), and finally, if $b(k)=0$ they are called linear reward–inaction (*LR-I*). In the latter case, the action probability vectors remain unchanged when the taken action is penalized by the environment. In the multicast routing algorithm presented in this paper, each learning automaton uses a linear reward–inaction learning algorithm to update its action probability vector. In the following, some convergence results of the learning automata are summarized.

Definition 3.1. The average penalty probability $M(n)$, received by a given automaton is defined

$$M(n)=E[\beta(n)\mid\zeta_n]=\int_{\alpha\in\underline{\alpha}}\zeta_n(\alpha)f(\alpha)$$

Where $\zeta:\underline{\alpha}\rightarrow[0,1]$ specifies the probability of choosing each action $\alpha\in\underline{\alpha}$, and $\zeta_n(\alpha)$ is called the action probability. If no priori information is available about f, there is no basis for selection of action. So, all the actions are selected with the same probabilities. This automaton is called *pure chance automaton* and its average penalty is equal to

$$M_0=E[f(\alpha)]$$

Definition 3.2. A learning automaton operating in a *P*-, *Q*-, or *S*-model environment is said to be *expedient* if

$$\lim_{n\to\infty} E[M(n)] < M_0$$

Expediency means that when automaton updates its action probability function, its average penalty probability decreases. Expediency can also be defined as a closeness of *E[M(n)]* to $f_l = \min_\alpha f(\alpha)$. It is desirable to take an action by which the average penalty can be minimized. In such case, the learning automaton is called *optimal*.

Definition 3.3. A learning automaton operating in a *P*-, *Q*-, or *S*-model environment is said to be *absolutely expedient* if $E[M(n+1) \mid p(n)] < M(n)$ for all n and all $p_i(n)$.

Absolute expediency implies that *M(n)* is a super martingale and *E[M(n)]* is strictly decreasing for all *n* in all stationary environments. If *M(n)<=M0*, absolute expediency implies expediency.

Definition 3.4. A learning automaton operating in a *P*-, *Q*-, or *S*-model environment is said to be *optimal* if

$$\lim_{n\to\infty} E[M(n)] = f_l$$

Optimality implies that asymptotically the action for which penalty function attains its minimum value is chosen with probability one. While optimality appears a very desirable property, certain conditions in a given situation may preclude its environment. In such cases, a suboptimal performance is desirable. Such property is called ε-optimality and is defined in the following definition. **Definition 3.5.** A learning automaton operating in a *P*-, *Q*-, or *S*-model environment is said to be ε-optimal if

$$\lim_{n\to\infty} E[M(n)] < f_l + \varepsilon$$

can be obtained for any $\varepsilon>0$ by a proper choice of the parameters of the learning automaton. ε-optimality implies that the performance of the learning automaton can be made as close to the optimal as desired.

3.1. Variable Action-Set Learning Automata

A variable action-set learning automaton is an automaton in which the number of actions available at each instant varies with time. In comparison with a variable action-set learning automaton, learning automaton with a fixed action-set is much easier to deal with. Fixed action-set learning automata are also easier for analysis. Therefore, the variable action-set learning automata have not received the attention they deserve. However, a learning automaton with a changing number of actions is absolutely expedient and also ε-optimal, when the reinforcement scheme is *LR-I* . Such an automaton has a finite set of *n* actions, $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$. $A = \{A_1, A_2, \ldots, A_m\}$ denotes the set of action subsets and $A(k) \subseteq \alpha$ is the subset of all the actions can be chosen by the learning automaton, at each instant *k*. The selection of the particular action subsets is randomly made by an external agency according to the probability distribution $q(k) = \{q_1(k), q_2(k), \ldots, q_m(k)\}$ defined over the possible subsets of the actions, where

$$q_i(k) = prob[A(k) = A_i \mid A_i \in A, 1 \le i \le 2^n - 1]$$

$\hat{p}_i(k) = prob[\alpha(k) = \alpha_i \mid A(k), \alpha_i \in A(k)]$ is the probability of choosing action ai, conditioned on the event that the action subset *A(k)* has already been selected and also $\alpha_i \in A(k)$ The probability of choosing the disabled actions is set to zero and the scaled probability $\hat{p}_i(k)$ is defined as

$$\hat{p}_i(k) = p_i(k) / K(k) \tag{3}$$

Where $K(k) = \sum_{\alpha_i \in A(k)} p_i(k)$ is the sum of the probabilities of the actions in subset *A(k)*, and $p_i(k) = prob[\alpha(k) = \alpha_i]$.

The procedure of choosing an action and updating the action probabilities in a variable action-set learning automaton can be described as follows. Let *A(k)* be the action subset selected at instant *k*. Before choosing an action, the probabilities of all the actions in the selected subset are scaled as defined in Eq. (3). The automaton then randomly selects one of its possible actions according to the scaled action probability vector $\hat{p}(k)$. Depending on the response received from the environment, the learning automaton updates its scaled action probability vector. Note that the probability of the available actions is only updated. In some cases, we need to enable the removed actions again. To do so, the probability vector of the actions of the chosen subset is rescaled as $p_i(k+1) = \hat{p}_i(k+1)K(k)$, for all $\alpha_i \in A(k)$.

4. The Proposed Learning Automata-based Channel Assignment Scheme

Channel utilization significantly decreases, if the same portions of bandwidth are assigned to all the hosts. This is due to the fact that in realistic scenarios, the traffic load of the different hosts is not the same, and so each host should be assigned a portion of bandwidth proportional to its traffic parameter. The main purpose of the proposed channel assignment scheme is to represent the new solution for polling-based medium access scheme in order to increasing the channel utilization and efficient use of bandwidth in wireless ad-hoc networks. In this scheme, the abilities of learning automata are used to increase the performance of channel assignment. In this scheme each cluster-head is equipped with a (variable action-set) learning automata. The action-set of the learning automaton assigned to a cluster-head includes an action for each of its cluster members. Therefore, the action-set cardinality of each cluster-head is equal to the number of its cluster members. After the cluster formation, each host sends the energy level to its cluster-head. The cluster-heads collect information on the energy levels of its members and then calculate the probability of each host in the cluster as follows.

$$p_j^i = \frac{energylevel_j}{\sum_{m=1}^{n} energlevel_m} \quad i \neq j$$

In the equation (5), P^i_j denotes the probability of the cluster member *CMj*, *energylevelj* is the energy level of the cluster member *CMj*, and *n* is total number of cluster members in *ith* cluster. In this equation, the initial selection probability of each action (host) is formed based on the energy levels. Let ij p $p^i = \{p_1^i, p_2^i, \ldots, p_{n-1}^i\}$ denotes the action probability vector of the learning automaton which is assigned to cluster-head *CHi*. In the beginning, each cluster-head selects one of the actions with high probability according to its action probability vector. Then, the cluster-head sends a POLL message to the cluster member according to selected action. If the selected cluster member has data packets for transmitting, it replies by transmitting its packets. Otherwise, if the selected cluster member has no packet to transmit, the cluster member sends a NDATA message to its cluster-head. This message means that no data packets for transmitting. If the cluster-head receives data packets from the selected cluster member, it rewards the selected action. In other words, the cluster-head increases the corresponding probability of choosing action based on equation (1). With this action the cluster-head increases the portion of bandwidth assigned to the cluster member.

Otherwise, the cluster-head decreases the probability by penalizing the selected action. Afterwards, the cluster-head randomly selects the one of its actions and performs the same operations as before. By this scheme, the probability of selecting a cluster member which it has the packets for transmitting increases. The same portion of bandwidth is assigned to all of cluster members initially. As the algorithm proceeds, a portion of bandwidth assigned to each cluster members or the number of times which is polled for each member converges to the proportion of time it has data packets to be transmitted. The flowchart of proposed polling scheme is depicted in figure 2. In the proposed scheme, when a host joins a cluster it first sends the *JREQ* message with its energy level to its cluster-head. The cluster-head calls *Join-Request* procedure upon receiving a *JREQ* message and the cluster-head adds the node's *ID* and node's energy level from the *JREQ* message to its cluster-member list as a newly joined member.

Then, cluster-head updates its action-set by adding a new action to the action-set according to equation (4). When a host joins the cluster; the action probability vector is updated as follows:

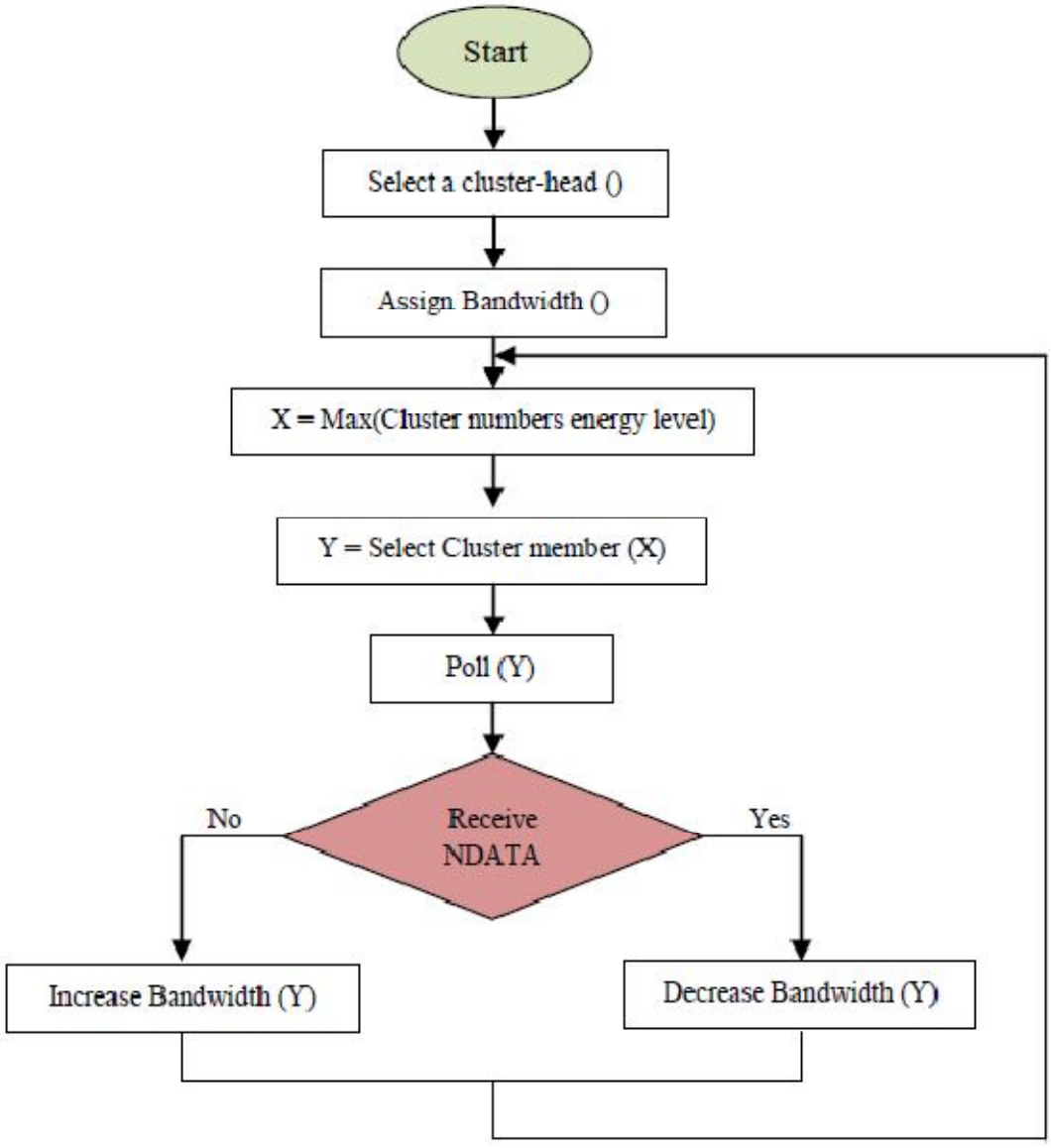

Fig. 2: Flowchart of Proposed Polling Scheme

$$p_j^i = \begin{cases} \frac{n-1}{n} \times \frac{energylevel_j}{\sum_{k=1}^{n} energylevel_k} & j \neq n, k \neq i \\ \frac{1}{n} & otherwise \end{cases} \quad (4)$$

Where, p^i_j denotes the probability of polling the cluster member *CMj* by the cluster-head *CHi*. Indeed, the cluster member *CMj* has permit to access the channel by cluster-head *CHi*. In this equation, the probability of choosing a newly joining host is lesser than probabilities of the other cluster members. Moreover, cluster members which have more energy levels have a greater chance for selecting.

When a host decides to leave a cluster, it sends a *LREQ* message to the cluster-head. The cluster-head call the *Leave-request* procedure upon receiving *LREQ* message. In this procedure, the cluster-head eliminates the ID of sender node from the cluster members' list, and updates its action-set by deactivating the action associated with the leaving cluster member. So, the action probability vector of the cluster-head *CHi* is updated based on equation (5). Assume that in the *i*th cluster with *CHi* as the cluster-head there exist *k* members. Also, assume the cluster member *CMr* decides to leave its cluster. The action probability vector of *CHi* is updated as follows:

$$p_j^i = \begin{cases} \frac{energylevel_j}{\sum_{k=1}^{n} energylevel_k} \times \frac{p_r^i}{1-p_r^i} + p_j^i & j \neq n, k \neq i \\ 0 & otherwise \end{cases}$$

However, not only a LREQ message can be used for informing the cluster-head that a host is leaving, but also in our proposed method no explicit control message is required to leave a cluster. When a cluster member leaves a cluster; after a short period of time; the portion of bandwidth allocated to the leaved cluster member tends toward zero. Also, the probability of choosing this host (to access the channel) converges to zero. Unlike the other polling method, traffic load varies with time. Therefore, the proposed method adapts to the changing traffic conditions and this is the main advantage of our method.

Under this assumption, the random environment in which the learning automaton operates on it, is a non stationary environment and so the penalty probabilities are directly proportional to the traffic load and vary with time. The steps of proposed algorithm are shown in flowchart in figure 3.

5. Evaluation

In this section, we have implemented the proposed protocol by Glomosim simulator [54], [55], a scalable discrete event simulator developed by UCLA.

5.1. Simulation Settings

The network area size is 1000*1000 (in m2). The mobility model is the random waypoint model. The minimum speed is 5 m/s, and the maximum speed is 15 m/s. We have used the IEEE 802.11 for distributed wireless sensor networks as the MAC layer protocol. DSR[3] protocol is used in network layer for routing. The number of hosts varies from 60 to 200 hosts. At each host, the arrival of the new connections is Poisson distributed with arrival rates of 5, 10, 15, and 20 connections per minute. Each host has a particular traffic load which is defined by randomly choosing from the above mentioned arrival rates at the beginning of each simulation. The radius of transmission range of all hosts is set to be the same, which is 250(m) throughout the simulation process. Initial energy level of each node is 5(mW) and radio transmit power is 10 (in dBm). The size of all data packets is set to 512 bytes. Each simulation experiment is executed for 1,000 s. The simulation results reported in this paper are averaged over 100 runs.

[3] Dynamic Source Routing

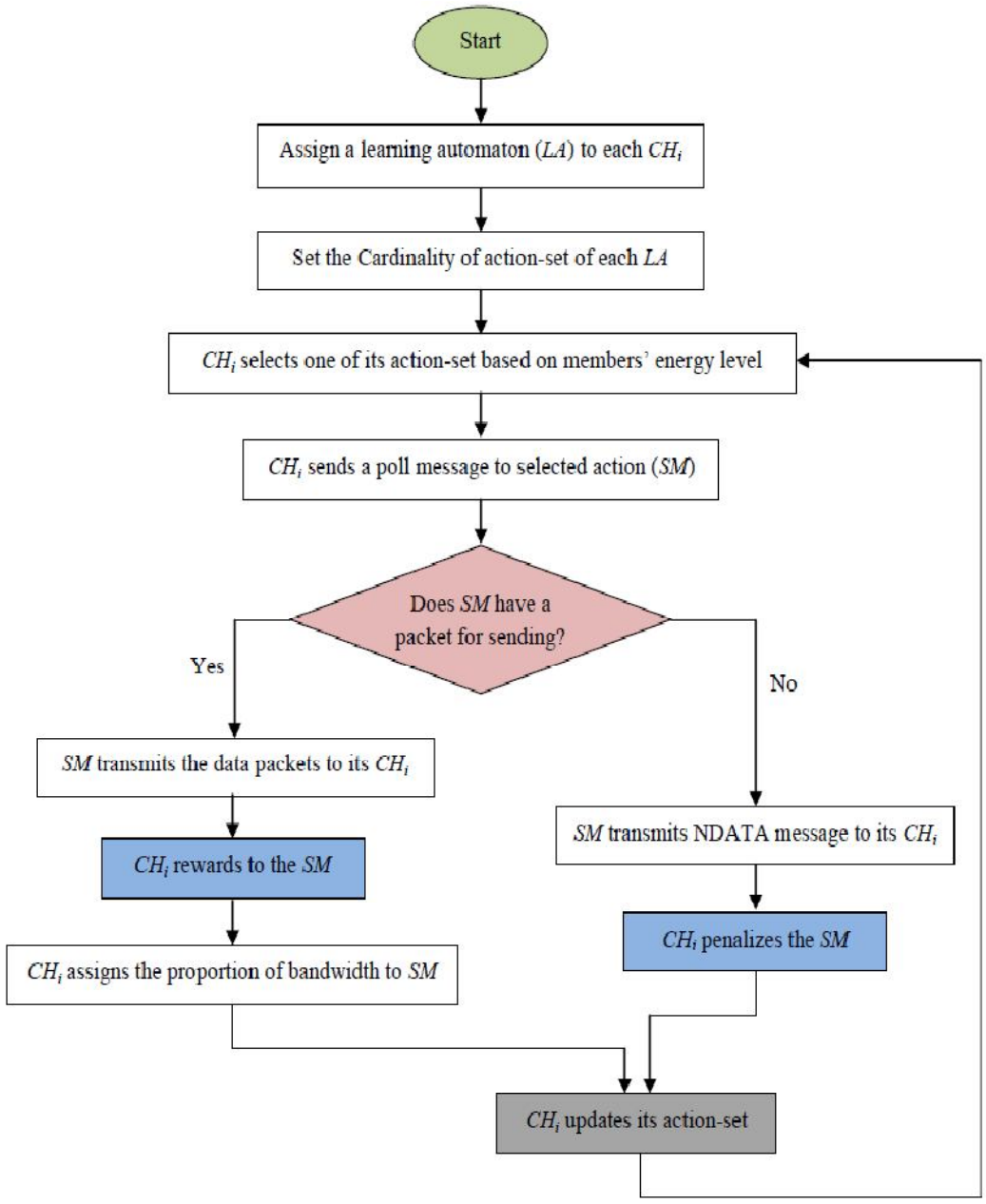

Fig. 3. The Steps of Proposed Algorithm

5.2. Performance Metrics

- Channel utilization. Channel utilization of a given host is defined as the ratio of the number of packets it transmits to the number of packets it receives per unit time. The channel utilization for the whole network is defined as the average channel utilization of the hosts. The traffic load of the various hosts is different in realistic scenarios, so this metric can be optimized by LAAP in which the bandwidth portion assigned to each host is proportional to its traffic load.
- Waiting time for packet transmission. This metric is defined as the average time each packet has to wait in the queue before transmission. This is the time between the arrival and transmission for each packet.
- Control overhead. This metric is defined as the average number of (non-data) control packets related to the channel access scheduling (or polling) process generated per unit time. Due to the scarce bandwidth in ad-hoc networks, this metric must be reduced as much as possible.

5.3. Performance Analysis of Proposed Method

To study the performance of the proposed polling-based medium access scheme, we have conducted several simulation experiments. In these experiments, the results obtained from the proposed scheme are compared with those of basic polling scheme (BPS), and two-hop polling scheme (2HPS) [15] in terms of performance metrics which we have described in previews section. Figure 4 shows the channel utilization versus the number of hosts for different algorithms. From the results shown in this figure, it is observed that the proposed algorithm has higher channel utilization in comparison with BPS and 2HPS algorithms. The reason is that proposed algorithm assigns the portion of bandwidth to each host proportional to the host requirements (i.e. traffic load). On the other hand, the portions of bandwidth assigned to the hosts witch leave their cluster is distributed among other cluster members.

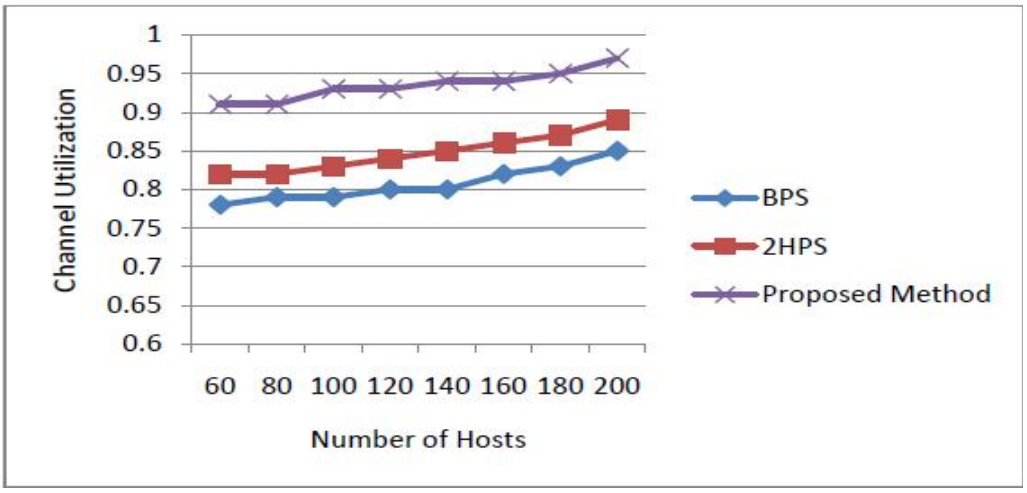

Fig. 4. Channel Utilization vs. Number of Hosts

Figure 5 shows the channel utilization of proposed method versus increasing arrival rate of new connections in the network in comparison with BPS and 2HPS.

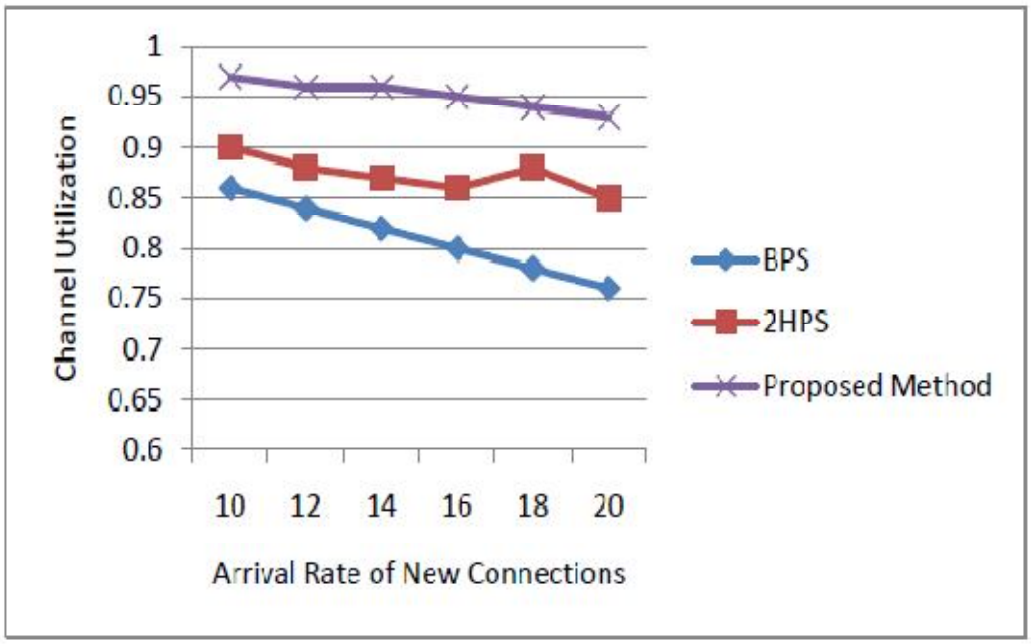

Fig. 5. Channel Utilization vs. Traffic Load

In the figure 5, the channel utilization of proposed method is higher than the BPS and 2HPS. Also, we can observe that the results of 2HPS are better than BPS. Because the inter-cluster links which 2HPS supports them, provide inter-cluster shortcuts for packet transmission between the neighboring hosts apart from the cluster-head. So, 2PHS have improved the average transmission rate compared to BPS. The results of simulation in figure 6 show the average waiting time with variation of hosts' number for different algorithms. The average waiting time for packet transmission is shorter that 2HPS and BPS.

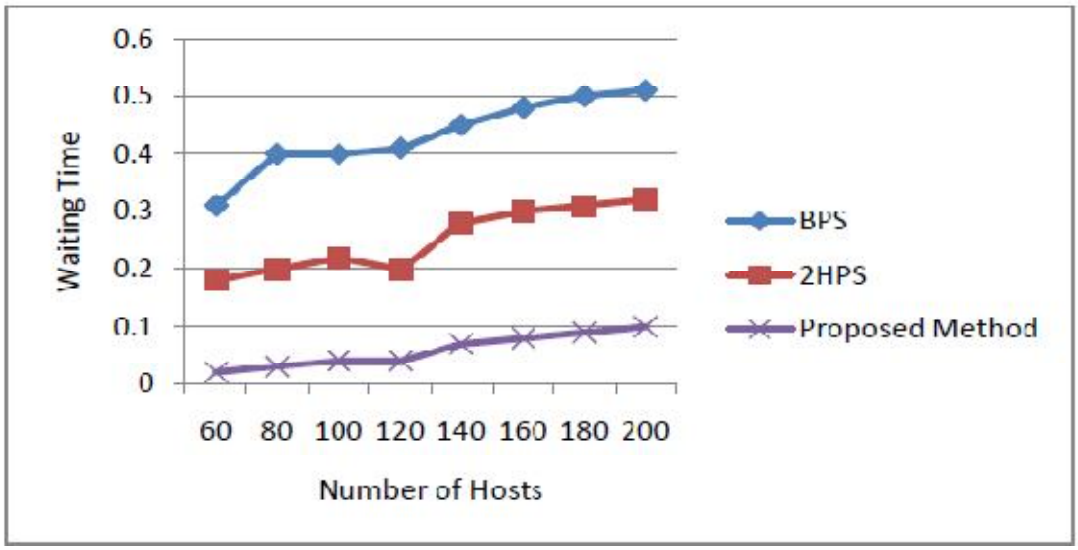

Fig. 6. The Average Waiting Time for Packet Transmission Versus the Network Size

The important reason for this behavior is that in the proposed scheme is assigned a portion of bandwidth to each host proportional to its need. Therefore, in proposed algorithm, those hosts with higher traffic load have more chance to access the channel, and these hosts with a higher traffic load have shorter waiting time for packet transmission. The results show that the average waiting time of 2HPS is shorter than BPS considerably. Because of 2HPS strongly supports inter-cluster links. The average waiting time for packet transmission increases with increasing the traffic load. Because of the number of connections requested by the host increases, and the packet buffering rate increases too. This results in a large number of connections to be directly established between the (non cluster-head) neighboring hosts of the different clusters, and so a large amount of packets are forwarded through these links.

In figure 7, the simulation results of average waiting time in variation of traffic load as a function of arrival rate of new connections are illustrated.

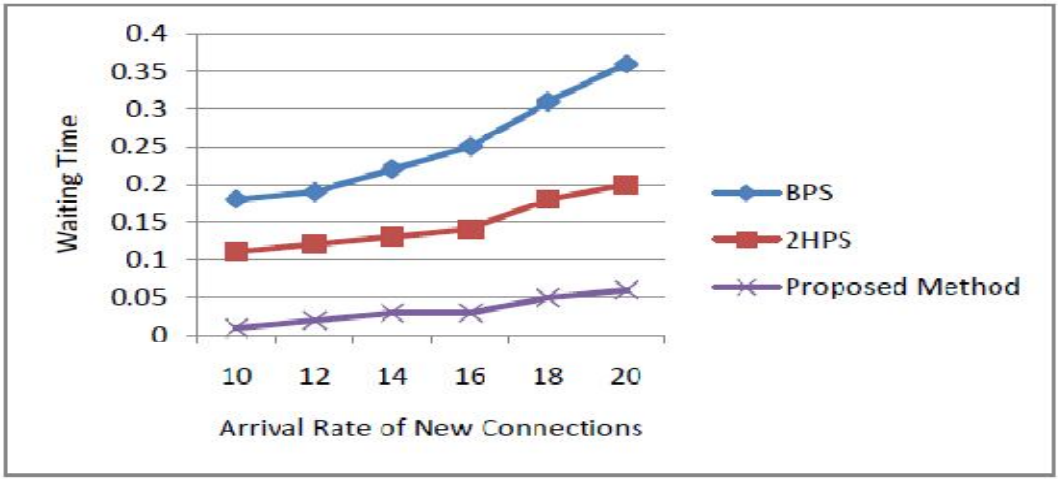

Fig.7: The Average Waiting Time for Packet Transmission Versus the Traffic Load

As we have discussed on the figure 6, in this figure (figure 7), average waiting time for packet transmission increases as the traffic load increases. The reason of this outcome is the number of requested connections by each host increases and so the packets' buffering rate increases too. However, we observe that the average waiting time of proposed method for packet transmission is shorter than 2HPS and BPS considerably, and 2HPS outperforms BPS. In the next simulation, we have evaluated the effects of hosts' number on the overhead of network. The control overhead is calculated as the total number of control messages per second generated by three algorithms. Figure 8 shows the control overhead with variation in number of hosts.

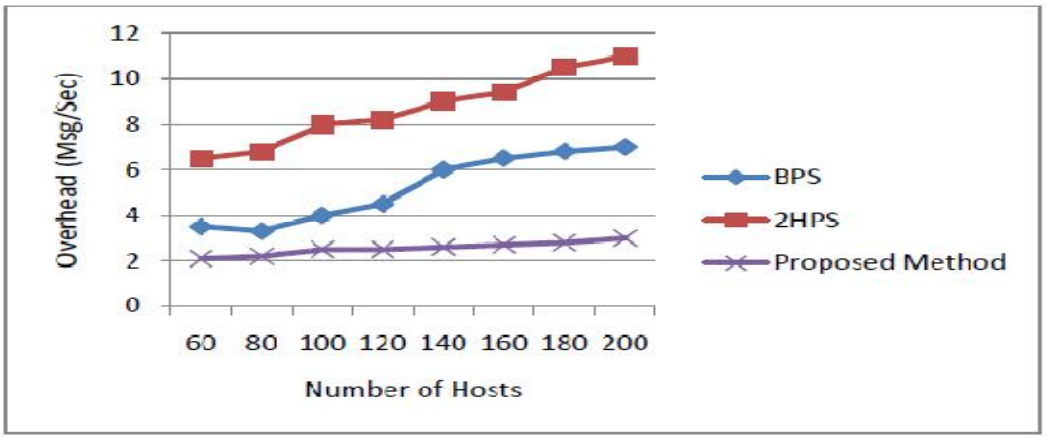

Fig. 8. Control Overhead vs. Number Of Hosts

The results show that the proposed algorithm outperforms BPS and 2HPS in point of control overhead. This is due to the fact that proposed algorithm does not need extra control packets for adjusting the portions of bandwidth. Moreover, in proposed algorithm, a cluster-head learns to poll a cluster member which it has data packets with a higher probability. Figure 9 shows the control overhead with variation of traffic load or arrival rate of new connections.

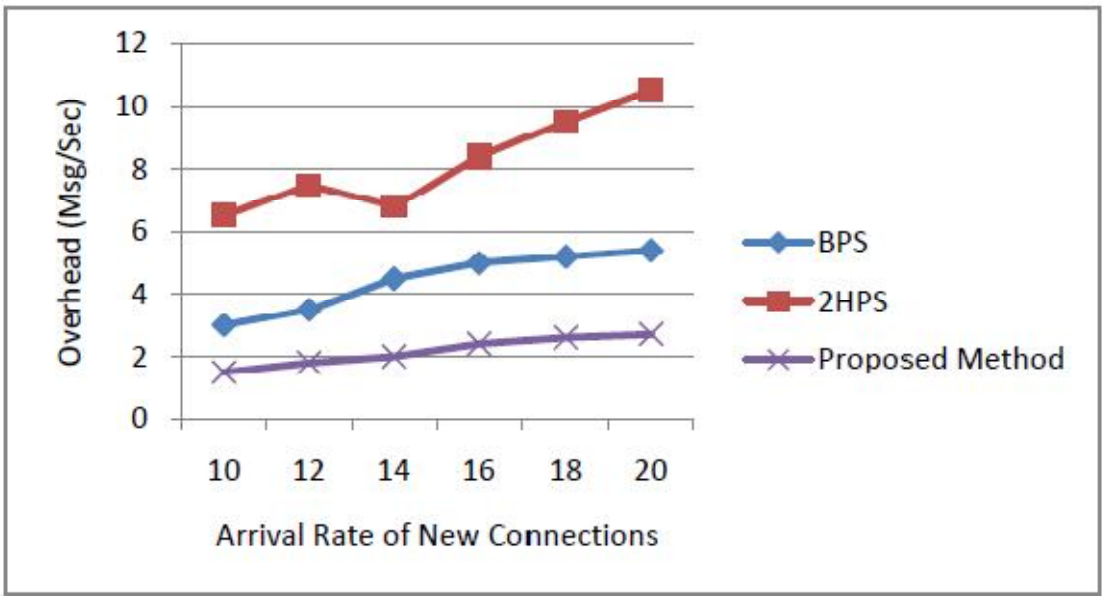

Fig. 9. Control Overhead vs. the Traffic Load

As we have discussed on the previews results, in the figure 9, the proposed algorithm outperforms PBS and 2HPS, and BPS is superior to 2PHS in terms of control overhead. This is due to the cost of maintaining the inter-cluster links in 2PHS. From the results in figure 9, it can be seen that control overhead increases as the number of connection increases.

6. Conclusion

In this article, we have represented a polling channel assignment scheme in clustered wireless ad-hoc networks. Our method increases the channel utilization and efficient use of bandwidth in wireless ad-hoc networks. In this scheme, each cluster-head is responsible for coordinating intra-cluster transmissions. In the polling scheme for medium access scheduling is used learning automata. Taking advantage of learning automaton, each cluster-head learns the traffic parameters of its own cluster members. Each cluster member is assigned a portion of bandwidth proportional to its needs. The simulations show that our proposed method not has a proper performance in terms of channel utilization but also it has the superiority over the existing methods in terms of waiting time for packet transmission and control overhead.

References

Callaway, E. H. (2005). The wireless sensor network MAC. In I. Stojmenovi´ c(Ed.), Handbook of sensor networks: Algorithms and architectures, New Jersey: Wiley-Inter sciences.

Saleh, A. A. M. (1982). Inter-modulation analysis of FDMA satellite systems employing compensated and uncompensated TWTs. IEEE Transactions on Communications, 30(5), pp. 1233–1242.

Lee, W. C. Y. (1991). Overview of cellular CDMA. IEEE Transactions on Vehicular Technology, 40(2), pp. 291–302.

Sekimoto, T., Puente, J. G. (1968). A satellite time-division multiple-access experiment. IEEE Transactions on Communications, 16(4), 581–588.

Capone, A., Gerla, M., Kapoor, R. (2001). Efficient polling schemes for bluetooth pico cells. In proceeding of the IEEE international conference on communications (ICC2001), Vol. 7, pp. 1990–1994, Finland.

Chrapkowski, A., Grube, G. (1991). Mobile trunked radio system design and simulation. In Proceedings of the IEEE vehicular technology conference, pp. 245–250.

Goodman, D. J., Valenzuela, R. A., Gayliard, K. T. and Ramamurthi, B. (1989). Packet reservation multiple-access for local wireless communications. IEEE Transactions on Communications, 37, pp. 885–890.

Abramson, N. (1970). The ALOHA system-another alternative for computer communications. In Proceedings of the AFIPS fall joint computer conference, Vol. 37, pp. 281–285.

Kleinrock, L., Tobagi, F. A. (1975). Packet switching in radio channels: Part I—carrier sensemultiple-access modes and their throughput-delay characteristics. IEEE Transactions on Communications, 23(12), pp. 1400–1416.

Grillo, D. (1990). Polling mechanism models in communication systems—some application examples. In H. Takagi (Ed.), stochastic analysis of computer and communication systems, pp. 659–698.

Wang, K., Peng, M. G., Wang, W.-B. (2007). Distributed scheduling based on polling policy with maximal spatial reuse in multi-hop WMNs. The Journal of China Universities of Posts and Telecommunications, 14, pp. 22–27.

Lye, K., Seah, K. (1992). Random polling scheme with priority. Electronics Letters, 28, pp. 1290–1291.

Lagkas, T. D., Papadimitriou, G. I., and Pomportsis, A. S. (2006). QAP: A QoS supportive adaptive polling protocol for wireless LANs. Computer Communications, 29, pp. 618–633.

Yang, C. C., Liu, C.-F. (2004). A bandwidth-based polling scheme for QoS support in bluetooth. Computer Communications, 27, pp. 1236–1247.

Dimitriadis, G., Pavlidou, F. N. (2003). Two-hop polling: An access scheme for clustered, multi-hop ad hoc networks. International Journal of Wireless Information Networks, 10(3), pp. 149–158.

Tseng, C. C., Chen, K. C. (2002). Priority polling with reservation wireless access protocol for multimedia ad-hoc networks. In proceedings of vehicular technology conference, Vol. 2, pp. 899–903.

Narendra, K. S., Thathachar, M. A. L. (1989). Learning automata: An introduction. Newyork: Printice-Hall.

Thathachar,M. A. L., Harita, B. R. (1987). Learning automata with changing number of actions. IEEE Transactions on Systems, Man, and Cybernetics, SMG17, pp. 1095–1100.

Billard, E. A., Lakshmivarahan, S. (1999). Learning in multi-level games with incomplete information-part I. IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, 19, pp. 329–339.

Akbari Torkestani, J., Meybodi, M. R. (2010). A new vertex coloring algorithm based on variable action-set learning automata. Journal of Computing and Informatics, 29(3), pp. 447–466.

Nicopolitidis, P., Papadimitriou, G. I. and Pomportsis, A. S. (2006). Exploiting locality of demand to improve the performance of wireless data broadcasting. IEEE Transactions on Vehicular Technology, 55(4), pp. 1347–1361.

Xing, Y., Mathur, C. N., Haleem, M. A., Chandramouli, R. and Subbalakshmi, K. P. (2007). Dynamic spectrum access with QoS and interference temperature constraints. IEEE Transactions on Mobile Computing, 6(4), pp. 423–433.

Meybodi, M. R. (1983). Learning automata and its application to priority assignment in a queuing system with unknown characteristics. Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of Oklahoma, Norman, Oklahoma, USA.

Hashim, A. A., Amir, S. and Mars, P. (1986). Application of learning automata to data compression. In K. S. Narendra (Ed.), Adaptive and learning systems (pp. 229–234). New York: Plenum Press.

Unsal, C., Kachroo, P. and Bay, J. S. (1999). Multiple stochastic learning automata for vehicle path control in an automated highway system. IEEE Transactions on Systems, Man, and Cybernetics-Part A, 29, pp. 120–128.

[26] Barto, A. G., Anandan, P. (1985). Pattern-recognizing stochastic learning automata. IEEE Transactions on Systems, Man, and Cybernetics, SMC-15, pp. 360–375.

Nicopolitidis, P., Papadimitiou, G. I., Sarigiannidis, P. G., Obaidat, M. S. and Pomportsis, A. S. (2011). Adaptive wireless networks using learning automata. IEEE Wireless Communications Magazine, 18(2), pp. 75–81.

Esnaashari, M., Meybodi, M. R. (2010). Dynamic point coverage problem in wireless sensor networks: A cellular learning automata approach. Journal of Ad Hoc and Sensors Wireless Networks, 10(2–3), pp. 193–234.

Esnaashari, M., Meybodi, M. R. (2010). Data aggregation in sensor networks using learning automata. Wireless Networks, 16(3), pp. 687–699.

Akbari Torkestani, J., Meybodi, M. R. (2011). Weighted Steiner connected dominating set and its application to multicast routing in wireless MANETs. Wireless Personal Communications, 60(2), pp. 145–169.

Akbari Torkestani, J., Meybodi, M. R. (2011). A mobility-based cluster formation algorithm for wireless mobile ad-hoc networks. Cluster Computing, 14(4), pp. 311–324.

Haleem, M. A., Chandramouli, R. (2005). Adaptive downlink scheduling and rate selection: A cross-layer design. IEEE Journal On Selected Areas in Communications, 23(6), pp. 1287–1297.

Xing, Y., Chandramouli, R. (2008). Stochastic learning solution for distributed discrete power control game in wireless data networks. IEEE/ACM Transactions on Networking, 16(4), pp. 932–944.

Beigy, H., Meybodi, M. R. (2009). Cellular learning automata based dynamic channel assignment algorithms. International Journal of Computational Intelligence and Applications, 8(3), pp. 287–314.

Nicopolitidis, P., Papadimitriou, G. I., Obaidat, M. S. and Pomportsis, A. S. (2005). Carrier-senseassisted adaptive learning MAC protocols for distributed wireless LANs. International Journal of Communication Systems, 18(7), pp. 657–669.

Beigy, H., Meybodi,M. R. (2011). Learning automata based dynamic guard channel algorithms. Journal of Computers and Electrical Engineering, 37(4), pp. 601–613.

Akbari Torkestani, J., Meybodi, M. R. (2010). An intelligent backbone formation algorithm forwireless ad hoc networks based on distributed learning automata. Computer Networks, 54(5), pp. 826–843.

Misra, S., Tiwari, V. and Obaidat, M. S. (2009). LACAS: Learning automata-based congestion avoidance scheme for healthcare wireless sensor networks. IEEE Journal on Selected Areas in Communications, 27(4), pp. 466–479.

Akbari Torkestani, J. (2012). LAAP: A learning automata-based adaptive polling scheme for clustered wireless ad-hoc networks. Wireless Personal Communication. doi: 10.1007/s11277-012-0615-5.

Akbari Torkestani, J. (2012). An adaptive backbone formation algorithm for wireless sensor networks. Computer Communications, 35(11), pp. 1333–1344.

Akbari Torkestani, J., Meybodi, M. R. (2012). Finding minimum weight connected dominating set in stochastic graph based on learning automata. Information Sciences (to appear).

Akbari Torkestani, J. (2012). An adaptive learning automata-based ranking function discovery algorithm. Journal of Intelligent Information Systems. doi:10.1007/s10844-012-0197-4.

Akbari Torkestani, J. (2014). A stable virtual backbone for wireless MANETS. Telecommunication Systems Journal, 55(1), pp. 137-148.

Akbari Torkestani, J. (2012). A new approach to the job scheduling problem in computational grids. Journal of Cluster Computing, 15(3), pp. 201-210.

Akbari Torkestani, J. (2012). Degree constrained minimum spanning tree problem in stochastic graph. Journal of Cybernetics and Systems, 43(1), pp. 1–21.

Akbari Torkestani, J., Meybodi,M. R. (2011). LLACA: An adaptive localized clustering algorithm for wireless ad hoc networks based on learning automata. Journal of Computers & Electrical Engineering, 3(4), pp. 461–474.

[111] Wu C-M., Dynamic frame length channel assignment in wireless multi hop ad-hoc networks. Computer Communications 2007b; 30:38, pp. 32–40.

Narendra K. S., Thathachar MAL. On the behavior of a learning automaton in a changing environment with application to telephone traffic routing. IEEE Transactions on Systems, Man, and Cybernetics, SMC-I0 1980, pp. 262–9

Lakshmivarahan S., Thathachar MAL. Bounds on the Convergence Probabilities of Learning Automata. IEEE Transactions on Systems, Man, and Cybernetics 1995, SMC-6: pp. 756–63.

Akbari Torkestani J., Meybodi M. R. (2010). An Efficient Cluster-Based CDMA/TDMA Scheme for Wireless Mobile Ad-hoc Networks: A Learning Automata Approach. Journal of Network and Computer Applications, pp. 477–490.

Thathachar MAL, Harita BR. Learning Automata with Changing Number of Actions. IEEE Transactions on Systems, Man, and Cybernetics 1987, SMG17: pp. 1095–100.

Akbari Torkestani J., Meybodi M. R. (2009). Graph Coloring Problem Based on Learning Automata. In: Proceedings of the International Conference on Information Management and Engineering (ICIME 2009), Malaysia, pp. 718–72.

Akbari Torkestani J., Meybodi M. R. (2009). Approximating the Minimum Connected Dominating Set in Stochastic Graphs based on Learning Automata. In: Proceedings of the International Conference on Information Management and Engineering (ICIME 2009), Malaysia, 2009b. pp. 672–76.

Zeng X., Bagrodia R. and Gerla M. GloMoSim: a library for parallel simulation of large-scale wireless networks, In PADS, 1998.

http://pcl.cs.ucla.edu/projects/domains/glomosim.html, 1 May 2006.