International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-3 Issue-1, October 2013

# A Novel Code Assignment Scheme based on Learning Automata for Clustered Wireless Mobile Ad-hoc Networks

**Amirhosein Fathinavid, Maryam Ansari, Elahe Ahmadpour Samani**

***Abstract*— In this paper, we have designed a code assignment algorithm for ad-hoc networks based on CDMA scheme. In this method, inter -cluster interference free communications have been organized. To do this assignment, proposed algorithm which we name CDMLA, uses learning automata concept. The proposed algorithm allocates interference free code to each cluster head with concept of code spatial reuse. This algorithm is performed based on clustering and a learning automaton is assigned to each cluster head. The learning automata residing in each cluster head allocates a code to its cluster head. We have implemented the system in network simulator GloMoSim. Also, we have rigorously evaluated the performance of our proposed solution by performing a variety of experiments through the extensive simulation experiments. The performance of proposed algorithm is measured, and the results are compared with CS-DCA, LACAA and Hybrid-DCA protocols in terms of the number of used codes, code spatial reuse, blocking rate, waiting time for packet transmission and throughput. Simulation results show that the proposed method outperforms the existing methods in terms of almost metrics of interest under the same conditions.***

***Index Terms*— CDMA, Code assignment Algorithms, Mobile ad-hoc networks, GloMoSim.**

## I. INTRODUCTION

CDMA is a spread spectrum multiple access scheme in which a transmitter spreads the information signal in a wide frequency band by using a spreading code. A receiver uses the same code to retrieve the received signal as well. This approach provides multiple accesses by allowing the simultaneous transmission by different nodes, and is employed to reuse the bandwidth and to reduce the interferences. In CDMA scheme, each group of nodes can be given a shared code. Many codes occupy the same channel, but only nodes associated with a particular code can understand each other. If the codes are orthogonal, or nearly so, so that any bit errors caused by co-channel interference can be handled by forward error correction, multiple nodes may occupy the same band. In the spread spectrum CDMA system, each node needs to know which code must be used for transmitting or receiving a particular packet. Indeed, the receiver should be set to the same code as the designated transmitter. Since the number of available codes is limited, it is impossible to assign a unique code to each transmitter or receiver, and so the concept of the code spatial reuse seems to be promising. In a clustered network, this means that two or more non-neighboring clusters can be assigned the same code. An interference-free code assignment problem is similar to the vertex coloring problem in which the neighboring nodes (clusters) are refrained from choosing the same colors (codes). Graph coloring problem is known to be NP-hard [1]. In CDMA scheme, simultaneous transmissions can be isolated by using different spreading codes. However, a node in a spread spectrum CDMA system needs to know which code should be used for transmitting or receiving a particular packet. In this scheme, a unique code is assigned to each transmitter and this is a trivial problem if the network size is small. But, when we employ the CDMA scheme in a large multi-hop ad-hoc network, the code assignment becomes an intractable problem. The concept of the code spatial reuse is a well-known solution reported in the literature ([2], [3]) by which a host of connections can be handled with a minimum number of codes. Another promising approach to solve the code assignment problem is using the CDMA/TDMA technique [3], [4]. To design a CDMA scheme, two following issues must be considered. First, grouping the hosts into a number of non-overlapping clusters and second, assigning a code to each cluster so that no two neighboring clusters have the same code. In[5], the authors also proposed a distributed cluster formation algorithm. The cluster heads act as local coordinators to resolve channel scheduling, perform power measurement/control, maintain time division frame synchronization, and enhance the spatial reuse of time slots and codes. Using a CDMA scheme, an interference-free code is assigned to each cluster, and a TDMA scheme is used within the clusters. In [6] authors also proposed a CDMA/TDMA based scheme for multimedia support in a self-organizing multi-hop mobile network. They introduced a network architecture in which the nodes are organized into non-overlapping clusters. In this method, the clusters are independently controlled and are dynamically reconfigured as the nodes move. In[6], an interference-free channel access scheduling method is proposed to handle the inter-cluster communications based on graph coloring problem. Due to the node clustering, the proposed method provides spatial reuse of the bandwidth. Furthermore, the bandwidth can be shared or reserved in a controlled fashion in each cluster. [7] proposed a dynamic code assignment algorithm for hybrid Multi-Code.

**Manuscript published on 30 October 2013.**
***** Correspondence Author (s)
**Amirhosein Fathinavid**, Department of Computer Engineering, Hamedan branch (Bahar) , Islamic Azad University, Hamedan, Iran.
**Maryam Ansari**, Department of Computer Engineering, Arak branch, Islamic Azad University, Arak, Iran.
**Elahe Ahmadpour Samani**, Department of Computer Engineering, Arak branch, Islamic Azad University, Arak, Iran.

CDMA/TDMA systems. The proposed code assignment scheme takes into account the time-varying traffic characteristics of the mobile users. In this algorithm, the base station assigns more codes to the mobiles during the congestion period. The extra codes are then released when the congestion subsides. In this method, the congestion is predicted based on the queue length of the mobiles.

In[4], the authors proposed a self-organized dynamic channel assignment scheme called CS-DCA to improve the spectrum efficiency for a TDMA microcellular system. The proposed scheme takes advantage of the channel segregation method introduced by[8], In channel segregation scheme, the channels are shared and dynamically assigned to the neighboring cells. In this method, for each cell, a dynamic priority is associated with every channel. When a call arrives, the channel with the highest priority is selected. If the selected channel is in use by the neighboring cells, the priority of the selected channel decreases, the next-highest priority channel is selected, and the same operation is repeated for the remaining channels. Otherwise, the selected channel is assigned to the call and the priority of the channel increases. This process is repeated until a free channel is found . In[9] the authors proposed a greedy based dynamic channel assignment algorithm called GB-DCA for cellular mobile networks. The proposed algorithm reduces the call blocking probability and increases the traffic-carrying capacity of the entire network. GB-DCA dynamically allocates the channels based on a greedy method. It uses an exhaustive search scheme for finding the co-channels. However, GB-DCA is designed for the cellular networks, where the wireless communications are limited to the same cell. In this case, only the neighboring cells are refrained from using the same codes. The code assignment problem becomes significantly harder when the GB-DCA is applied in multi-hop ad hoc networks. In [3],the authors proposed a CDMA/TDMA scheme for clustered wireless ad-hoc networks in which GB-DCA is used to handle interference-free code assignments. Wu designed a dynamic channel assignment algorithm called Hybrid-DCA to make the best use of available channels by taking advantage of the spatial reuse concept. In this approach, the TDMA scheme is overlaid on top of the CDMA scheme to divide the bandwidth into smaller chunks. Hybrid-DCA forms the channel as a particular time slot of a particular code. It borrows the color-based cluster formation algorithm proposed in [2] for clustering the wireless ad hoc networks. It also uses the channel segregation-based dynamic channel assignment algorithm (CS-DCA) proposed in[4] to assign the collision-free intra-cluster channel accesses. In Hybrid DCA, the increase in spatial reuse is achieved by GB-DCA [9] and the decrease in control overhead by CS-DCA ([4]).

The above mentioned channel assignment schemes are practical when the input traffic is fixed or a stationary process with known parameters. This paper aims at designing a dynamic frame length CDMA/TDMA scheme for clustered wireless ad hoc networks with unknown traffic parameters. In a CDMA/TDMA scheme, intra-cluster communications are scheduled by the cluster-head using a TDMA scheme, and a CDMA scheme is overlaid on the TDMA to organize the interference-free inter cluster communications. Therefore, to design the proposed scheme, the following three important problems must be considered; Cluster formation, code assignment (in CDMA scheme), and slot assignment (in TDMA scheme) problems. In this paper, we propose a learning automata-based algorithm to solve the code assignment problem. By the proposed cluster formation algorithm, the network is partitioned into a small number of clusters, each with a cluster-head and a number of cluster members. Inter-cluster connections are handled by a CDMA scheme, in which an interference-free code must be assigned to each cluster. To do so, we propose a code assignment algorithm based on the vertex coloring problem in which the neighboring clusters are refrained from choosing the same codes.

Through the extensive simulation experiments, the performance of the proposed CDMA scheme is measured and compared with CS-DCA [4] and Hybrid-DCA [3] in terms of the number of clusters, code and channel spatial reuse, blocking rate, waiting time for packet transmission, control overhead and throughput. The obtained results show that the proposed scheme outperforms others in almost all metrics of interest, specifically, under burst traffic conditions.

The rest of the paper is organized as follows. The next section introduces the learning automata, and Section 3 describes the proposed overlaid CDMA/TDMA scheme. Section 4 shows the superiority of the proposed scheme over the existing methods through the simulation experiments. Section 5 concludes the paper.

## II. LEARNING AUTOMATA

A learning automaton ([10], [15], [19] , 1987,[11], [10]) 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 nd 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 [20],[21] show that the learning automata can be effectively used for solving the intractable optimization problems. In[18] several attempts have been also made to exhibit the capabilities of the learning automata in dynamic wireless ad hoc networks. We briefly review some of the fundamental concepts below.

### A. *The Automata*

The Automaton is typically defined by a quintuple *{A, B, Q, F(.,.), G(.)}* where [18]:

1) *A={1, 2, ..., r}* is the set of outputs or actions, and *(t)* is the action chosen by the automaton at any instant *t.*
2) B is the se t of inputs to the automaton, $\{\beta_1, \beta_2, ..., \beta_r\}$. *β(t)* is the input at any instant *t* while the set *B* can be finite or infinite.
3) *Q={q1(t), q2(t), ..., qS(t)}* is the set of finite states, where *q(t)*denotes the state of the automaton at any instant *t.*


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

4) *F(.,.): Q×BQ* is a mapping in terms of the state and input at the instant *t*, such that *q(t+1)=F[q(t), β(t)]*. It is called a transition function, i.e., a function that determines the state of the automaton at any subsequent time instant *t + 1*. This mapping can either be deterministic or stochastic depending on the environment in which the automaton operates.

5) *G()*: is a mapping *G*: QA, and is called the output function. Depending on the state at a particular instant, this function determines the output of the automaton at the same instants: α(t)=G[q(t)]. This mapping can, again, be considered to be either deterministic or stochastic, depending on the environment in which the automaton operates [19]. Without loss of generality, G is deterministic.

B. The Environment

The environment, E typically refers to the medium in which the automaton functions. The environment possesses all the external factors that affect the actions of an automaton. Mathematically, an environment can be abstracted by a triple {A, C, B}. A, B, and C are defined as follows:

- A={α1, α2, ..., αr } represents a finite input set,
- B={β1, β2, ..., βl } is the output set of the environment,
- C={c1, c2, ..., cr } is a set of penalty probabilities, where element ci ε C corresponds to an input action αi.

The process of learning is based on a learning loop involving two entities: the random environment (RE) and the learning automata, as described in Figure 1. A learning automaton has finite set of actions and at each stage it chooses one of them. The choice of an action depends on the state of learning automata represented by an action probability vector. For each action chosen by the learning automata, the environment gives a reinforcement signal with unknown probability distribution. Then upon receiving the reinforcement signal, the learning automaton updates its action probability vector by employing a learning algorithm.

In the process of learning, the learning automaton continuously interacts with the environment to process responses to its various actions. Finally, through sufficient interactions, the learning automaton attempts to learn the optimal action offered by the RE. The actual process of learning is represented as a set of interactions between the RE and the learning automata.

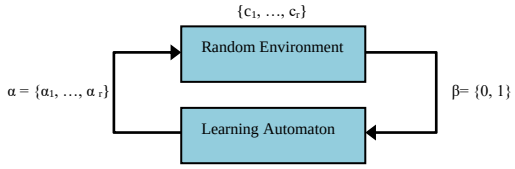


Fig. 1. The Automaton-Environment feedback loop

Learning automatons can be classified into two main families: fixed structure learning automata and variable structure learning automata. A variable-structure automaton is defined by the quadruple {α, β, p, T} in which α={ α1, α2, …, αn} represents the action set of the automata, β={ β1, β2, …, βr} represents the input set, p={p1, p2, …, pr} represents the action probability set, and finally p(n+1)=T[α(n), β(n), p(n)] represents the learning algorithm. This automaton operates as follows. Based on the action probability set p, automaton randomly selects an action αi and performs it on the environment. Having received the environment's reinforcement signal, automaton updates its action probability vector based on equation (1) for favorable responses, and on equation (2) for unfavorable ones [19].

$$p_i(n+1) = p_i(n) + a.(1 - p_i(n))$$
$$p_j(n+1) = p_j(n) - a.p_j(n) \quad \forall j \quad j \neq i \qquad (1)$$
$$p_i(n+1) = (1-b).p_j(n)$$

$$p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j \quad j \neq i \qquad (2)$$

In these two equations, a and b are reward and penalty parameters respectively. For a = b, learning algorithm is called LR-P, for a << b it is called LRεP, and for b=0 it is called 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 2.1. The average penalty probability M(n), received by a given automaton is defined as

$$M(n) = E[\beta(n)\zeta_n] = \int_{\alpha \in \underline{\alpha}} \zeta_n(\alpha) f(\alpha) \qquad (3)$$

Where $\zeta : \underline{\alpha} \rightarrow [0,1]$ specifies the probability of choosing each action $\alpha \in \underline{\alpha}$ and $\zeta_n(x)$ 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 2.2. 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 to be optimal.

### C. *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, in some applications, learning automata with changing number of actions needed. In [18] a group of variable action-set learning automata is used to solve the vertex coloring problem, or in [18] variable action-set learning automata cooperate to find the virtual backbone of the ad hoc networks. It has been shown in[19], [20],[21] that a learning automaton with a changing number of actions is absolutely expedient and also e-optimal, when the reinforcement scheme is $L_{R-I}$. 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.


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

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 $q_i(k) = prob[A(k) = A_i \mid A_i \in A, 1 \le i \le 2^n - 1]$, and $\hat{p}_i(k) = prob[\alpha(k) = \alpha_i \mid A(k), \alpha_i \in A(k)]$ is the probability of choosing action $\alpha_i$ 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

$$\hat{p}_i(k) = p_i(k) \mid K(k) \qquad (4)$$

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 a $\alpha_i \in A(k)$. The absolute expediency and ε-optimality of the method described above have been proved in [19].

## III. THE PROPOSED CDMA SCHEME (CDMLA)

To design a CDMA scheme for clustered wireless ad-hoc networks, we encounter the following two intricate problems. The first problem is dividing the network into a minimum number of non-overlapping clusters. The second problem is to assign an interference-free code to each cluster considering the concept of the maximum code spatial reuse. That is, this problem is assigning the minimum number of codes to the clusters so that no two neighboring clusters are assigned the same codes. This problem is similar to the graph (vertex) coloring problem in graph theory which is known to be NP-hard[1]. Our CDMA scheme proposes a learning automata-based solution to each of the above mentioned problems, which are described in detail below.

### *A. Learning automata-based cluster formation algorithm*

In ad hoc networks, the network performance is significantly degraded as the network becomes larger. The theoretical analysis [12] shows that even under the optimal circumstances, the throughput of each host rapidly declines as the network size increases. Among the solutions proposed for solving the scalability problem in ad hoc networks, network clustering has attracted a lot of attention. The main idea behind the clustering approach is to group together the network hosts that are in physical proximity. The clusters provide a hierarchical structure to abstract the large scale networks which can be simply and locally organized [13], [14]. A clustering algorithm is a method for dividing the network into clusters so that each cluster includes a number of cluster members and a cluster-head (CH) with which the members can directly communicate. Due to the host mobility, strict resource limitations (e.g., bandwidth and power limitations), and hard to predict topology changes, clustering in ad hoc networks aims at dividing the network hosts into a minimum number of groups with the maximum stability.

In this section, we propose a learning automata-based approximation algorithm for clustering the wireless ad hoc networks. In this algorithm, a network of learning automata isomorphic to the network graph is formed by assigning each host $h_i$ a learning automaton $A_i$. Since we associate a learning automaton with each host, hereafter, host $h_i$ may be called as learning automaton $A_i$ and vice versa. The resulting network of learning automata can be described by a duple $<\underline{A}, \underline{\alpha}>$, where $A=\{A_1, A_2, \ldots, A_n\}$ denotes the set of learning automata corresponding to the vertex set of the network graph, and $\underline{\alpha} = \{\underline{\alpha_1}, \underline{\alpha_2}, \ldots, \underline{\alpha_n}\}$ denotes the set of action-sets, in which $\underline{\alpha_i} = \{\alpha_j^i \mid h_i \text{ is a neighbor of } h_j \text{ or } i = j\}$ defines the set of actions can be taken by learning automata $A_i$. The action-set of host $h_i$ (or learning automaton $A_i$) includes an action for each of its neighboring hosts as well as an action for itself. Choosing action $\alpha_i^j$ by host $h_i$ means that host $h_i$ selects host $hj$ as its cluster-head. That is, each host can chooses one of its neighboring hosts or itself as its cluster-head. The proposed cluster formation algorithm consists of a number of stages, and at each stage, each host picks its cluster-head among its neighbors or declares itself as a cluster-head. The following steps briefly describe a sample stage of the proposed cluster formation algorithm which is executed at host $h_i$.

The cluster formation algorithm is a fully distributed algorithm in which each host chooses its cluster-head based solely on the local information received from its neighboring hosts. This algorithm is independently run at each host, and the information upon which the CH selection decision is based is conned to the neighborhood of the host. Furthermore, in the proposed algorithm, the hosts need not to be synchronized, and the neighboring hosts locally form the clusters.

In this algorithm, when a host joins the network, it initially broadcasts a *JREQ* (i.e., join request) message and then waits for a certain period of time. If a *CH* receives the *JREQ* message, it replies by sending back a *JREP* (join reply) message. If the newly joining host receives a *JREP* message, it chooses the sender of the *JREP* message as its CH, and sends a *CH-SEL* message to it. Otherwise, it chooses the neighboring host with the higher *ID* number as its *CH*. In this case, the new cluster-head calls CDMLA algorithm (described later) to receive a code. If the newly joining host receives more than one *JREP* message, it selects the sender with the higher *ID* number as its *CH*.

In mobile ad hoc networks, due to the node mobility and failures, the network topology frequently changes. Due to these dynamics, the network clusters may rapidly lose their validity, and the network must be clustered again. Re-clustering phase in many clustering algorithms reported in the literature is performed periodically for the entire network.


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

In such algorithms, in predetermined time intervals, the normal operation of the network is interrupted, the clustering algorithm is performed on the entire network (producing a completely new clustered infrastructure) and then the normal operation of the network is resumed. Such periodical re-clustering schemes has a number of drawbacks. The very first problem with such schemes is that they consume too much energy because the re-clustering is performed on the entire network. Another problem is that the normal operation of the network is delayed until the re-clustering phase is over. Unlike such re-clustering schemes, firstly, the proposed re-clustering algorithm is performed locally and adaptively whenever it is needed. Secondly, by this method, the re-clustering is performed where the previous infrastructure is not valid anymore. In CDMLA, when a cluster-head decides to leave the network, it broadcasts a *REC-REQ* (re-clustering request) message, and asks its cluster members for a re-clustering process. Each cluster member that receives the *REC-REQ* message calls CDMLA algorithm for finding a new cluster-head. The re-clustering process is locally performed on demand, and the other clusters continue their normal operation during the re-clustering phase. If a cluster member decides to leave the cluster, it sends a *LREQ* (i.e., leave request) message to the cluster-head. Upon receiving the *LREQ* message, cluster-head removes it from the cluster member list. The proposed cluster formation algorithm guarantees to cluster the entire network at each stage. In proposed algorithm, each host chooses its cluster-head, and so the network is partitioned into a number of non-overlapping clusters, in which each host is only associated with a unique cluster-head. As the algorithm proceeds, the number of cluster-heads decreases as the number of members in each cluster increases. Furthermore, in the proposed cluster formation algorithm, the cluster-heads are one, two or at most three-hops away. From clustering algorithm, it can be found that two clusters are not adjacent (or neighboring clusters), if their cluster heads are four-hops or more away from each other. Hence, in a clustered network, some hosts have no cluster-heads, if the minimum distance between a given cluster-head and the other cluster-heads is more than three-hops. The following shows that this case does not occur in clustering algorithm. As shown in Fig. 2(a), cluster heads 1 and 4 are three-hops away. Let us assume that (as shown in Fig. 2(b)) host 5 is located between hosts 2 and 3, i.e., cluster heads 1 and 4 are four-hops away. This way, no cluster-head is associated with host 5, and this contradicts to the above assumption that in the algorithm each host selects its cluster-head. On the other hand, if we suppose that host 5 is associated with a cluster-head, this cluster-head must be (at most) three-hops away from cluster-heads 1 and 4. Therefore, by this algorithm, each cluster head is at most three-hops away from (at least) another cluster head.

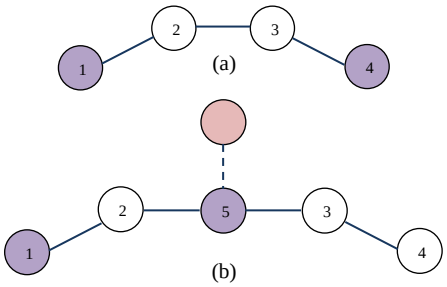


Fig. 2. Clustering a sample network by proposed clustering algorithm

### *B. Codeassignment algorithm based on learning automata(CDMLA)*

In this section, we propose an algorithm for solving the code assignment problem in a clustered wireless ad hoc network based on learning automata concepts. The proposed algorithm, which we call it CDMLA, aims to assign the minimum number of interference-free codes to the clusters. Since the number of available codes is limited, it is impossible to assign a unique code to each cluster, and so we take advantage of the code spatial reuse concept in our proposed algorithm. By this concept, two or more non-neighbor clusters can be assigned the same codes. Such a code assignment problem is similar to the NP-hard vertex-coloring problem in which no two neighboring clusters (or node) have the same code (or color). In CDMLA, the structure is defined in each cluster head node with following fields: $ch_i$ denotes the *i*th cluster, $n_i$ denotes the number of nodes in ith cluster, $N_j^i$ is the number of node *j* in cluster *i*. $P_j^i$ indicates the power of node *j* in cluster *i* and $P_{chi}$ is the power of *i*th cluster head.

In first step, each node in a cluster sends the packet with its energy level to its cluster head. The cluster head upon receive the packets of all nodes inter cluster, stores these information in predefined structure. Then the cluster head calculates the cost of link based on equation (5).

$$cost(ch_i) = \sum_{j=1}^{ni} \frac{d^2(N_j^i, ch_i)}{min(P_j^i, p_{ch_i})} \quad (5)$$

In this equation, $N_j^i$, $ch_i$, , $P_{chi}$ are accordance with the definitions provided and $d^2(N_j^i, ch_i)$ is the distance of node *j* from *i*th cluster.

The cluster head node stores all of calculated costs for each node in its table. In the next step, the cluster head selects one accessible code. The size of this code is the maximum queue (cluster table) length. In addition, each cluster head is informed of the selected code by its neighboring cluster heads through outer cluster communications. Fig. 3 shows the process of exchanging the messages in a cluster sample ad-hoc network with four clusters.

As shown in fig. 3(a), we assume that cluster-head CH chooses code 1, $CH_2$ chooses code 2, $CH_3$ chooses code 3, and $CH_4$ chooses code 1. This is an interference-free code assignment. Each cluster broadcasts a message to announce its code. These messages are forwarded to the cluster-head of neighboring clusters. As shown in fig. 3(d), all cluster-heads after three-hops receive the information of its neighboring clusters. It can be seen that, $CH_2$ and $CH_3$ receive the information of $CH_1$ only. Each cluster-head compares the received codes with its selected code and finds no interference.

Now, each cluster-head compares its selected code with the codes which are being used within its neighboring clusters. Then, learning automata residing in cluster head rewards or penalizes the selected action (selected code) as follows:

- If this code is against the selected code by neighboring cluster heads, then the cluster head rewards the desired node.


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

- If the cost of desired cluster head link is more than the cost of cluster-heads link of its neighboring clusters which either select same code, then the cluster head rewards to the node.
- Otherwise, the cluster head penalizes the desired node.

The action of rewarding or penalizing performs based on learning rules (equation 2 and 3). This algorithm continuous until the all of cluster heads can be reward. Because the nodes in a network have mobility and updating of the clusters performs periodically. The updating is in point of number of nodes in each cluster and the energy level of the nodes. Fig. 4 shows the pseudo code of proposed algorithm.

### C. *Convergence behaviorof the proposed algorithm*

As mentioned earlier, in proposed algorithm each learning automaton operates independent of the others in a network of learning automata. In [10], [11] and [15], the convergence of such an automaton (with two actions) to the optimal solution in stationary or non-stationary environments has been proved. This convergence proof can be similarly generalized for a learning automaton with more actions. For the convergence speed of the learning automata-based algorithms, it should be noted that the convergence speed of a learning automaton is directly proportional to the learning rate, and its convergence rate to the optimal solution is inversely proportional to this parameter. When the learning rate decreases, the convergence speed also decreases (convergence rate increases), and the convergence speed significantly increases as the learning rate converges to one. This property of the learning automata enables us to make a trade-off between the costs (time complexity and message complexity) of algorithm and the optimality of the obtained solution. For instance, a trade-off between the number of clusters (or the number of used codes) and the number of iterations of algorithm (i.e., the running time of algorithm) can be made. In ad hoc networks, where the hosts suffer from the strict resource limitations (e.g., bandwidth or power), the communication and processing overheads should be kept as low as possible.

On the other side, a near optimal solution is usually sufficient in many applications of these networks. Therefore, by a proper choice of the learning rate, an acceptable solution can be provided for different applications in a reasonable time. That is, the running time of the proposed algorithms can be accommodated to the required optimality of the solution for different applications. In simulation experiments, we examined different learning rates and observed that 0.1 is a proper choice for optimizing different parameters by which the solution optimality and cost reached a compromise. In ad hoc networks, the obtained solutions rapidly lose their validity. Therefore, in our algorithm which is proposed for ad hoc networks, we sacrifice the optimality of the results in favor of the running time and message overhead (costs) of algorithm, and suffice to a near optimal solution of cluster formation and code assignment problems, although it can be seen that our proposed algorithm outperform the existing methods in almost all metrics of interest.

## IV. EVALUATION

In this section, we have implemented the proposed protocol by Glomosim simulator [16][17], a scalable discrete event simulator developed by UCLA. Then, we have compared the efficiency of CDMLA with the similar methods.

### A. *Simulation settings*

Our simulation scenarios are presented in table 1.

Table1. Simulation settings

| | |
|---|---|
| **Routing Protocol** | DSR |
| **Mac Layer Protocol** | Mac 802.11 |
| **Network Size** | 1000 * 1000 $m^2$ |
| **Node Placement** | Random |
| **Size of Packets** | 512 bit |
| **Band Width** | 2 mb/s |
| **Radio Range of Nodes** | 250 m |
| **Number of Hosts** | 60 – 200 |
| **Simulation Time** | 1000s |
| **Number of Used Codes** | 4 – 20 |

Moreover, every host in the network is modeled as unlimited buffer for saving and sending packets. The rate of new connections' entrance is according to position distribution with means 5, 10, 15 and 20 of connections/min.

### B. *Performance metrics*

In these experiments, the performance of the proposed algorithm is evaluated in terms of the following metrics of interest:

- Code spatial reuse: This metric is defined as the average number of times a code can be used. That is, code spatial reuse is the average number of clusters, which are assigned the same code. This metric is used to assess the performance of the code assignment algorithm (CDMLA).
- Number of used codes: It is defined as the number of codes, which are assigned to the clusters. This metric is inversely proportional to the code spatial reuse.
- Blocking rate: It is defined as the ratio of the number of the blocked connections to the total number of requested connections. This metric can be divided into two categories; Blocking rate due to no code, and blocking rate due to no slot. The first defines the rate of connections blocked due to the lack of available codes, and the second defines the rate of blocked connections due to the lack of free slots.
- 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.
- Throughput: This metric is defined as the ratio of the average number of packets transmitted per slot to the total number of packets received. The traffic load of the various hosts is different in realistic scenarios.

### C. *Simulation results*

In this section, we have evaluated our proposed algorithm CDMLA based on the performance metrics. Then we have compared the results of simulations between CDMLA and LACAA [18], CS-DCA and Hybrid-DCA[3].

In the first simulation, we have investigated the number of codes assigned in comparison with the number of hosts. Fig. 5 shows these results.


*Published By:*
*Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)*

Comparison between these results shows that CDMLA algorithm has used fewer codes than LACAA, Hybrid-DCA and CS-DCA algorithms. The reason is that CDMLA uses calculated link cost for each node and considers the energy levels and the distances between hosts in one cluster, while in the protocols CS-DCA, Hybrid-DCA and LACAA costs are not considered. It is also clear that the difference between CDMLA and LACAA protocols is lower than the other protocols because these two algorithms divides the network is considerably smaller than the number of clusters in Hybrid-DCA and CS-DCA. However, the number of used codes increase decreases by the increase of hosts' number.

In second experiment as shown in figure 6, CDMLA is compared with other protocols in terms of code spatial reuses with variation of number hosts.

From the results shown in fig. 6, we can see that LACAA outperforms the others in terms of the code spatial reuse when the number of host is low, and Hybrid-DCA is superior to CS-DCA. However, when the number of hosts increases (obviously after 100 hosts) CDMLA outperforms the LACAA, Hybrid-DCA and CS-DCA. In the CDMA scheme proposed in this paper, on one hand the number of clusters constructed by learning automata concept is much less than that constructed by CS-DCA and Hybrid-DCA, and, on the other hand, by using energy level of hosts in calculation link cost, the minimum number of codes are assigned to the neighboring clusters. Therefore, the rate of code spatial reuse in CDMLA is less than that in LACAA.

Fig. 7 shows the average time each packet has to wait for in the queue before transmission. The results show that the average waiting time of CDMA for packet transmission is shorter than that of LACAA, Hybrid-DCA, and CS-DCA.

Obviously, in fig. 7, the average waiting time for packet transmission increases as the number of hosts increases. This is due to the fact that in CDMLA, each host based on learning automata residing in cluster head selects one accessible code and sends all packets immediately.

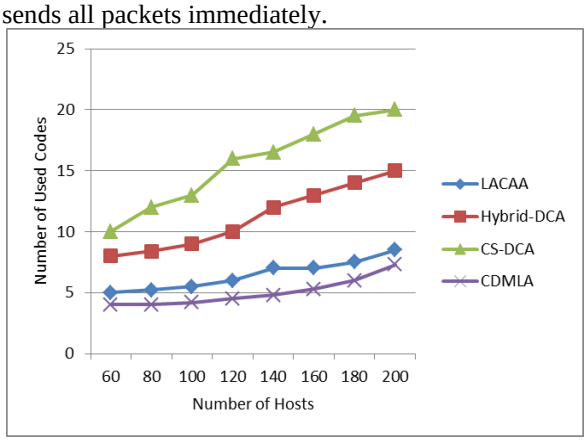


Fig. 6. The code spatial reuse versus the number of hosts

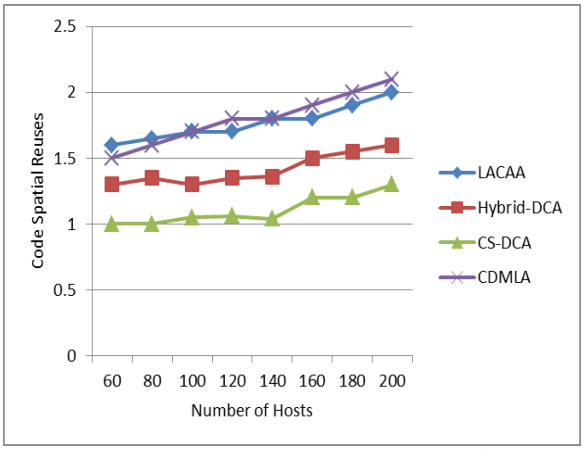


Fig. 6. The code spatial reuse versus the number of hosts

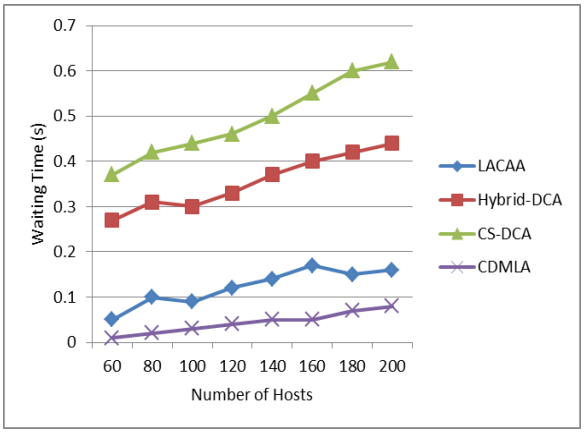


Fig. 7. The average waiting time for packet transmission versus the number of hosts.

So the data packets can be sent to selected paths quickly. Since the packets are generally aggregated in the hosts, CDMLA reduces the average waiting time for packet transmission. Moreover, in LACAA the average waiting time is longer than CDMLA. Because LACAA spends a few time to assign a portion of time frame proportional to its need. Also, the average waiting time of LACAA for packet transmission is shorter than that of Hybrid-DCA, and CS-DCA.

Fig. 8 shows the number of connections blocked due to the lack of the codes versus the number of hosts. From the results shown in this figure, we conclude that the number of blocked connections decreases as the number of hosts increases.

Comparing the results shown in Fig. 8, we observe that the number of blocked connections in CS-DCA is smaller than Hybrid-DCA. This is because CS-DCA uses a larger number of codes than Hybrid-DCA.


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

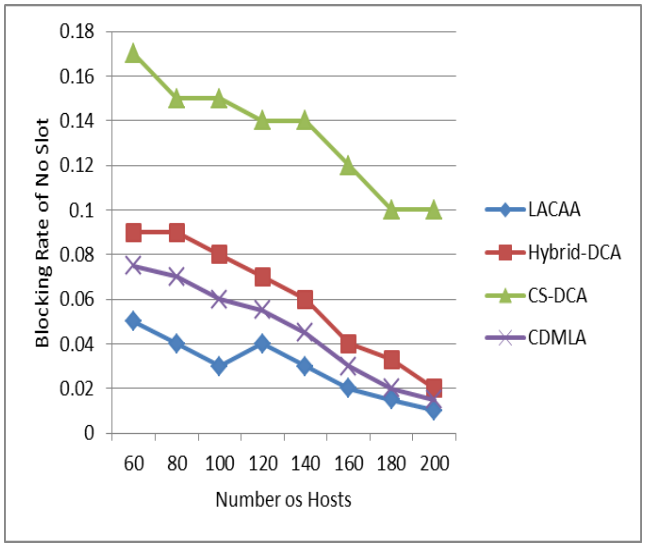


Fig. 8. The blocking rate due to the lack of the codes versus the number of hosts.

We also observe that LACAA has a lower blocking rate compared with CDMLA, CS-DCA and Hybrid-DCA. In LACAA, no code is reserved for the new hosts, the probability of choosing the codes assigned to the leaving hosts are immediately distributed over the cluster members, and the number of slots assigned to each host is proportional to its traffic load. The blocking rate of CDMLA is lower than CS-DCA and Hybrid-DCA but in compared with LACAA, the blocking rate is higher because in this experiment, we use at least number of codes for gathering results.

In the last simulation, we have studied the throughput rate of the network in comparison with the number of hosts. Fig. 9 shows these results.

From the results in this figure, it is observed that LACAA has a higher throughput as compared with CDMLA, CS-DCA and Hybrid-DCA. However, there is not much difference between two protocols LACAA and CDMLA. The low throughput of CDMLA compared to LACAA is the using a large number of control packets for gathering information about the energy levels and the distances of hosts in one cluster. Despite of this point, the throughput curve of CDMLA is partly suitable.

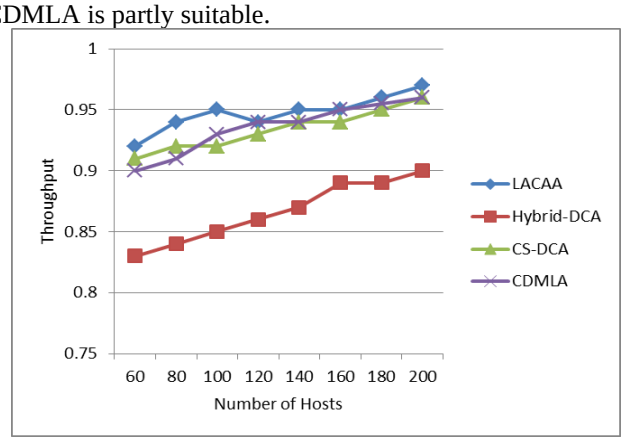


Fig. 9. Throughput versus the number of hosts.

## V. CONCLUSION

In this article, we have proposed a learning automata-based code assignment scheme in clustered wireless mobile ad-hoc networks. To design this scheme, we proposed two learning automata-based algorithms for cluster formation and code assignment (CDMLA) respectively. In this scheme, by clustering algorithm, the wireless hosts were first grouped into a minimum number of non-overlapping clusters. Then, by CDMLA, an interference-free code was assigned to each cluster. Extensive simulation results showed that the proposed CDMA scheme outperforms the existing methods for almost all metrics of interest, specifically, under burst traffic conditions.

## REFERENCES

[1] RM. Karp, Reducibility among Combinatorial Problems. Complexity of Computer Computations. USA: Plenum Press; 1972. pp. 85-103.
[2] STC. Hou and TJ. Tsai, On the Cluster Based Dynamic Channel Assignment for Multi hop Ad hoc Networks. Journal of Communications and Networks, 2002, 4 (1):40-7.
[3] C-M. Wu, Hybrid Dynamic Channel Assignment in Clustered Wireless Multi hop CDMA/TDMA Ad-hoc Networks. Wireless Personal Communications 2007a; pp. 85-105.
[4] Y. Akaiwa and H. Andoh, Channel Segregation Self-Organized Dynamic Channel Allocation Method: Application to TDMA/FDMA Microcellular System, IEEE Journal of Selected Areas in Communications, 1993, 11(6): 949-54.
[5] CR. Richard Lin and M. Gerla, Adaptive Clustering for Mobile Wireless Networks, IEEE Journal of Selected Areas in Communications, 1997, 15(7).
[6] M. Gerla and J. Tsai, Multi cluster, Mobile, Multimedia Radio Network, ACM/Baltzer Journal on Wireless Networks,1995; 1(3):255-65.
[7] C-L. Yang and J-F. Chang, Dynamic Code Assignment in Hybrid MC-CDMA/TDMA Systems, European Transactions on Telecommunications, 2003, 14(1): 49-59.
[8] Y. Furuya and Y. Akaiwa, Channel Segregation, a Distributed Adaptive Channel Allocation Scheme for Mobile Communication Systems, IEICE Transactions, 1991, E74:15, pp. 31-37.
[9] X. Fang, C. Zhu and P. Fan, Greedy-Based Dynamic Channel Assignment Strategy for Cellular Mobile Networks, IEEE Communication Letters, 2000, 4(7): 2-157.
[10] KS. Narendra and KS. Thathachar, Learning Automata: an Introduction. New York: Prentice-Hall, 1989.
[11] S. Lakshmivarahan and MAL. Thathachar, Bounds on the Convergence Probabilities of Learning Automata, IEEE Transactions on Systems, Man, and Cybernetics, 1995, SMC-6:7, pp. 56-63.
[12] P. Gupta and PR. Kumar, The Capacity of Wireless Networks, IEEE Transactions on Information Theory, 2000, 46(2), pp. 388-404.
[13] R. Rajaraman, Topology Control and Routing in Ad hoc Networks: A Survey. SIGACT News, 2002, 33(2), pp. 60-73.
[14] YZ. Chen and AL. Listman, Approximating Minimum Size Weakly Connected Dominating Sets for Clustering Mobile Ad hoc Networks, In: Proceedings of the Third ACM International Symposium on Mobile Ad hoc Networking and Computing (MobiHoc2002), 2002, pp. 57-64.
[15] MAL. Thathachar and PS. Sastry, A Hierarchical System of Learning Automata that Can Learn the Globally Optimal Path, Information Science 1997, 42:7, pp. 43-66.
[16] X. Zeng, R. Bagrodia, and M. Gerla, GloMoSim: a library for parallel simulation of large-scale wireless networks, in: PADS, 1998.
[17] http://pcl.cs.ucla.edu/projects/domains/glomosim.html, 1 May 2006.
[18] J. Akbari Torkestani and MR. Meybodi, An Efficient Cluster-Based CDMA/TDMA Scheme for Wireless Mobile Ad-hoc Networks: A Learning Automata Approach, Journal of Network and Computer Applications, 2011, pp. 477-490.
[19] MAL. Thathachar and BR. Harita, Learning Automata with Changing Number of Actions, IEEE Transactions on Systems, Man, and Cybernetics, 1987, SMG17:10, pp. 95-100.
[20] J. Akbari Torkestani and MR. Meybodi, Graph Coloring Problem Based on Learning Automata, In: Proceedings of the International Conference on Information Management and Engineering (ICIME2009), Malaysia, 2009a, pp. 71-87.
[21] J. Akbari Torkestani J and MR. Meybodi, 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. 67-76.


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

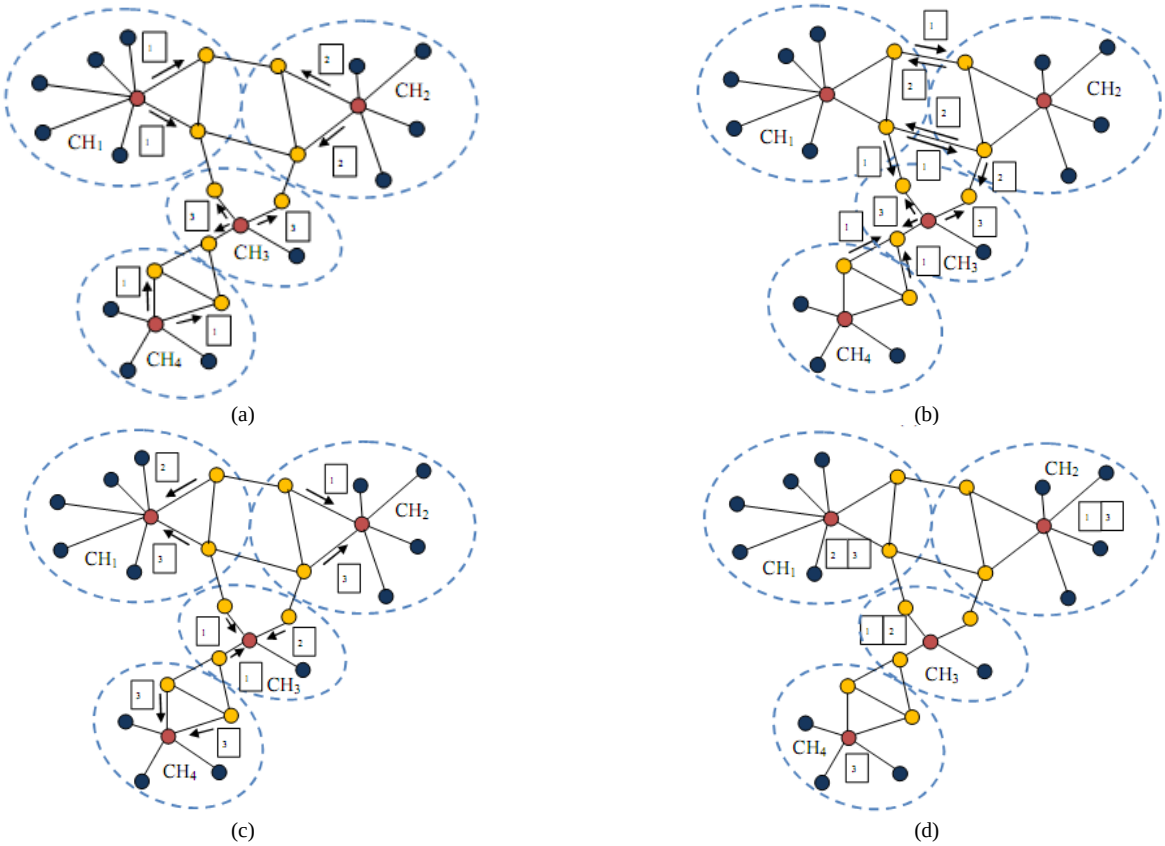


Fig. 3. Process of exchanging message for a sample ad-hoc network

```
For each ch_i
{
    cost (h_i )=0
    For (j=1 to n_i )
    {
        A= d^2( N_j, ch_i)
        B= min (p_j. pch_i)
        cost (h_i) = cost( h_i) +(A/B)
    }
    Select one code (C_i) from access code
    For (j=1 to n_i )
    {
        If (C_i <> C_j)
        {
            Reward(h_i)
        }
        Else if (cost (h_i) > cost (h_j) && C_i == C_j)
        {
            Reward(h_i)
        }
        Else
        {
            Penalize(h_i)
        }
    }
}
```

Fig. 4. The pseudo code of CDMLA


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)