

Home Browse Search My settings My alerts Shopping cart
Export citation PDF (1293 K) More options...
Search Sc

☒ Show thumbnails in outline

Abstract

Keywords

1. Introduction

2. Preliminaries

2.1. Learning automata theory

2.2. Variable action-set learning automata

2.3. Problem statement

3. Proposed multicast routing algorithm

4. Experimental results

4.1. Packet delivery ratio

4.2. Multicast route lifetime

4.3. End-to-end delay

4.4. Control message overhead



Journal of Network and Computer Applications

Volume 34, Issue 4, July 2011, Pages 1429–1440

Advanced Topics in Cloud Computing

A link stability-based multicast routing protocol for wireless mobile ad hoc networks

Javad Akbari Torkestani^a, , Mohammad Reza Meybodi^b 

^a Department of Computer Engineering, Islamic Azad University, Arak Branch, Arak, Iran

^b Department of Computer Engineering and IT, Amirkabir University of Technology, Tehran, Iran

Received 19 August 2010. Revised 17 February 2011. Accepted 9 March 2011. Available online 21 March 2011.

<http://dx.doi.org/10.1016/j.jnca.2011.03.026>, [How to Cite or Link Using DOI](#) Cited by in Scopus (3)

[Permissions & Reprints](#)

Abstract

Recently, several studies have been conducted to design mobility-based multicast routing protocols for wireless mobile ad hoc networks (MANET). These protocols assume that the mobility parameters of the network are fixed, and so they cannot perform well under real MANET scenarios in which the mobility parameters of the hosts vary over time at random. Finding the optimal solution to the multicast routing problem is incredibly hard, if the mobility parameters are assumed to be random variables. This becomes more difficult when the probability distribution function of these random variables is assumed to be unknown. In this paper, we propose a weighted multicast routing algorithm for MANET in which the mobility parameters are supposed to be random variables with unknown distribution. In this method, the multicast routing problem is first transformed into an equivalent stochastic Steiner tree problem in which the random weight associated with a communication link is its expected duration time. Then, a learning automata-based algorithm is proposed for solving the proxy Steiner tree problem. The aim of the proposed algorithm is to find the most stable multicast route (with the maximum duration) against the host mobility. Experimental results confirm the superiority of the proposed method over the best existing mobility-based multicast routing protocols in terms of the packet delivery ratio, multicast route lifetime, control message overhead, and end-to-end delay.

Keywords

Multicast routing; Link stability; MANET; Steiner tree; Learning automata

1. Introduction

Multicasting is a technique for data routing in networks that allows the same message is forwarded to a group of destinations simultaneously. Multicast routing problem (MRP) is one of the most challenging problems in communication networks in which the source node is connected to a set of destinations, while some cost function is simultaneously minimized. The most efficient method is forwarding the multicast packets along the tree edges, and so the multicast routing problem can be defined as a Steiner tree problem where the multicast group members are terminals (leaf nodes) in the Steiner tree (Wu et al., 2004). Finding the Steiner tree is known to be NP-hard in graph theory (Karp, 1972) even if links have unit cost (Aggarwal et al., 2006). However, the minimal spanning tree problem that is a well known approach for broadcasting has also been used in Chiang et al. (2007) and Rodolakis et al. (2008) to model the multicast routing problem. In MANET due to the host heterogeneity, host mobility, strict resource limitations, and host failure, the characteristics of the network (e.g., residual energy, mobility parameters, transmission power, etc.) are stochastic and vary over time. In these networks, the deterministic approaches in which the network parameters are assumed to be fixed cannot reflect the real behavior of the random environment, and so do not perform well. This paper proposes a multicast routing protocol for MANET in which the stability of the communication links changes with time due to host mobility.

Lee et al. (1999) proposed an on-demand routing technique called ODMRP for multicasting in ad hoc networks in order to reduce the channel overhead and improve scalability. ODMRP is implemented based on the forwarding group concept (Chiang et al., 1998) in which a set of nodes are responsible for forwarding multicast packets along the shortest paths connecting the multicast sender to receivers. ODMRP is a reactive protocol that delivers packets to destinations on a mesh topology using limited flooding of data. Simplicity and scalability are the major strengths of ODMRP. An enhanced version of ODMRP with motion adaptive refresh called E-ODMRP was proposed by Oh et al. (2008) in which the adaptation is driven by receivers' reports on link breakages rather than mobility prediction. The rate at which E-ODMRP periodically refreshes the network information is dynamically adapted to the host mobility. E-ODMRP unifies the local recovery mechanism and receiver joining process. Local recovery is a mechanism to reducing the overhead of the network flooding in which an alternative route to the destination is locally searched upon detection of a breakage. If a local recovery fails, disconnected node tries to connect to the multicast route by a new receiver joining process. That is, on joining a new host or detection of a breakage, the host performs a local route recovery search to graft to the multicast route proactively. Biradar et al. (2010) proposed a link stability-based multicast routing protocol for mobile ad hoc networks. The aim of the proposed scheme is to construct the stable multicast mesh based on the selection of the stable forwarding nodes having higher link stability. In this method, the stability of a communication

Rela

A ne
Expe
Revi
Joun
Opti
Euro
Loca
Com
An e
Com
View

T

Most d
in this

St

View F

ORI

ORI



link is estimated based on the power level, spatial distance, and link quality. The power level of a node is measured based on the power of the signal which is received at its neighbors. Spatial distance is defined as the distance between two neighboring hosts which is computed using the free space propagation model (Dai and Wu, 2005), and the link quality is computed based upon the average of bit errors in a large number of packets. The proposed protocol, hereafter referred to as LSMRM, maintains a data structure called link stability database (LSD) at each node to keep the stability information of the incident links for each node. LSMRM consists of the following three phases. The first phase is the construction of the multicast mesh by the route request and route reply messages. The second phase is to find the stable routes connecting the multicast source and receivers on the basis of the link stability parameters, and the third one is to design a route maintenance mechanism to handle the link failures. The authors compared the performance of LSMRM with ODMRP (Lee et al., 1999) and E-ODMRP (Oh et al., 2008) and showed that it outperforms the others in terms of the packet delivery ratio, control message overhead, required memory, and delay. Su et al. (2001) presented a mobility prediction method to estimate the expiration time of a communication link on the basis of the location and mobility information of the mobile hosts. The aim of the proposed method is to find the stable paths against the host mobility by estimating the time interval during which two neighboring hosts remain connected. They also proposed another method to estimate the link expiration time based on the information of the propagation channel. In this method, each host periodically assesses the transmission power of its neighboring hosts based on the strength of the receiving signals. The changes in transmission power are considered as a measure to estimate the disconnection time (i.e., the time at which two hosts leave the transmission range of each other).

An and Papavassiliou (2003) argue that the communications in MANET often arise among teams that tend to coordinate their movements. As a result, they propose a multicast routing algorithm in which the network is dynamically and adaptively partitioned into several groups, each with its own mobility behavior and a group-based hierarchical multicast routing. In this approach, the relative mobility of each host with respect to its peers is the criterion by which the stable hosts in a group are determined. Toh and Bunchua (2001) proposed an associativity-based ad hoc multicast (ABAM) routing protocol. ABAM establishes the multicast sessions using the association stability concept introduced in ABR unicast protocol (Toh, 1997). In ABAM, the association stability concept is used to extend the multicast route lifetime by selecting the long lived routes. ABAM applies the association stability technique during multicast tree discovery, selection, and reconstruction phases. Results show the superiority of ABAM over ODMRP in terms of throughput and communication overhead. In Guo and Yang (2008), two multicast routing algorithms are proposed for maximizing the route lifetime in MANETs. The aim of the first proposed algorithm is to reduce the energy consumption in network hosts, and so to extend the route lifetime. To do so, the transmission power of the hosts is periodically readjusted in such a way that the transmission power of each host is tuned to the minimum possible level. The results show that the proposed algorithm saves much more energy compared to the methods which apply single level of transmission power only. The second algorithm attempts to extend the duration of a multicast route by selecting the most stable hosts in each area. In this method, the stability of a host is estimated based on its location and mobility parameters with respect to its neighboring hosts. McDonald and Znati (2004) considered the impact of the link availability on the routing problem in wireless ad hoc networks, and developed an analytical framework for statistically predicting the link stability so as to find the survivable communication links. The availability of a link is predicted based on the velocity and direction of the mobile host and represents the probability of remaining two hosts within the transmission range of one another. That is, how long two neighboring mobile hosts remain connected. The stability of a route is assessed on the basis of the product of the availability of the route's links. They conducted computer simulation to validate the developed framework and showed that the link availability-based route selection method provides substantial improvement in route longevity in contrast with both shortest-path (minimum hop) and fixed-threshold schemes.

All the above mentioned protocols designed for ad hoc multicast routing aim at extension of the lifetime of the multicast routes by selecting the stable hosts. However, the major problem with these multicasting approaches is that they cannot perform well in realistic MANET scenarios in which the mobile hosts are free to change their mobility parameters randomly at any time. This is because these multicasting protocols assume that the host's mobility characteristics are deterministic, while they are stochastic and vary over time. In real ad hoc networks, no assumption can be made on the probability distribution function (PDF) of the host mobility parameters. That is, the host mobility characteristics in MANET are random variables with unknown probability distributions. Under such circumstances, finding an optimal solution to form the stable (resistant to host mobility) multicast routes is incredibly hard.

In this paper, we propose a learning automata-based weighted multicast routing algorithm for wireless MANETs in which the multicast packets are forwarded along the Steiner tree links. The weight associated with a communication link is defined as its expected duration time which is assumed to be a random variable with unknown distribution. Expected link duration time is defined as the period of time during which the link is expected to be connected, and expected duration time of a multicast route is defined as the expected duration time of the weakest link. The aim of the proposed algorithm is to find the most stable multicast route against the host mobility. Obviously, the multicast route with the maximum expected duration time has the most stability in long time. The proposed algorithm is composed of a number of iterations and at each iteration a multicast route is constructed by finding a Steiner tree of the network topology graph. At each iteration, the selected multicast route is rewarded, if its expected duration time is longer than that seen so far and it is penalized otherwise. As the proposed algorithm proceeds, the probability of selecting the most stable multicast route (i.e., the multicast route with the maximum expected duration) converges to one. To show the performance of the proposed algorithm, we have conducted several simulation experiments and compared the obtained results with those of E-ODMRP (Oh et al., 2008) and LSMRM (Biradar et al., 2010). Experimental results show that the proposed algorithm outperforms the others in terms of the packet delivery ratio, multicast route lifetime, control message overhead, and end-to-end delay.

The rest of the paper is organized as follows. The next section provides a brief review of learning automata and represents the problem statement. In Section 3, a mobility-based multicast routing algorithm is proposed for wireless MANETs using learning automata. In Section 4, the performance of the proposed algorithm is evaluated through computer simulation, and Section 5 concludes the paper.

2. Preliminaries

To provide a sufficient background for understanding the basic structures of the multicast routing algorithm, which is proposed in this paper, a brief overview of the learning automata theory and the concept of the expected duration time are presented in Sections 2.1 and 2.2, respectively.

2.1. Learning automata theory

A learning automaton ([Narendra and Thathachar, 1989], [Thathachar and Sastry, 1997] and [Thathachar and Harita, 1987]) is an adaptive decision-making unit that improves its performance by learning how to choose the optimal action from a finite set of allowed actions through 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 instant 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 automaton has shown to perform well in solving combinatorial optimization problems ([Akbari Torkestani and Meybodi, 2010a], [Akbari Torkestani and Meybodi, 2010b], [Akbari Torkestani and Meybodi, in press-a], [Akbari Torkestani and Meybodi, in press-b] and [Akbari Torkestani and Meybodi, in press-c]) and computer networks ([Akbari Torkestani and Meybodi, in press-d], [Akbari Torkestani and Meybodi, 2010c], [Akbari Torkestani and Meybodi, 2010d], [Akbari Torkestani and Meybodi, 2010e], [Akbari Torkestani and Meybodi, 2010f], [Akbari Torkestani and Meybodi, 2011], [Akbari Torkestani and Meybodi, in press-d] and [Akbari Torkestani and Meybodi, in press-e]).

The environment can be described by a triple $E = \{\alpha, \beta, c\}$, where $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ represents the finite set of the inputs, $\beta = \{\beta_1, \beta_2, \dots, \beta_m\}$ denotes the set of the values that can be taken by the reinforcement signal, and $c = \{c_1, c_2, \dots, c_r\}$ 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 environments depending on the nature of the reinforcement signal β can be classified into P-model, Q-model, and S-model. The environments in which the reinforcement signal can take only 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 continuous interval [0,1].

Learning automata can be classified into two main families (Narendra and Thathachar, 1989): fixed structure learning automata and variable structure learning automata. Variable structure learning automata are represented by a triple $\langle \beta, \alpha, T \rangle$, where β is the set of inputs, α is the set of actions, and T is the learning algorithm. The learning algorithm is a recurrence relation, which is used to modify the action probability vector. Let $\alpha(k) \in \alpha$ and $\beta(k)$ denote the action selected by learning automaton and the probability vector defined over the action set at instant k , respectively. Let a and b denote the reward and penalty parameters and determine the amount of increases and decreases in the action probabilities, respectively. Let r be the number of actions that can be taken by learning automaton. At each instant k , the action probability vector $\beta(k)$ is updated by the linear learning algorithm given in Eq. (1), if the selected action $\alpha(k)$ is rewarded by the random environment, and it is updated as given in Eq. (2) if the taken action is penalized:

$$\beta_j(k+1) = \begin{cases} \beta_j(k) + a[1 - \beta_j(k)] & j = i \\ (1-a)\beta_j(k) & \forall j \neq i \end{cases} \quad (1)$$

$$\beta_j(k+1) = \begin{cases} (1-b)\beta_j(k) & j = i \\ \left(\frac{b}{r-1}\right) + (1-b)\beta_j(k) & \forall j \neq i \end{cases} \quad (2)$$

If $a=b$, the recurrence equations (1) and (2) are called linear reward-penalty (L_{R-P}) algorithm, if $a \neq b$ the given equations are called linear reward- ϵ penalty ($L_{R-\epsilon}$), and finally if $b=0$ they are called linear reward-inaction (L_{R-I}). In the latter case, the action probability vectors remain unchanged when the taken action is penalized by the environment. The following presents some definitions on convergence behavior of a learning automaton.

Definition 2.1. The average penalty probability $M(n)$, received by a given automaton, is defined as

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

where $\zeta_n: \alpha \rightarrow [0,1]$ specifies the probability of choosing each action $\alpha \in \alpha$, 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 2.2. A learning automaton operating in a P-, Q-, or S-model environment is said to be expedient if

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

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 $f = \min_{\alpha} f(\alpha)$. 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 2.3. A learning automaton operating in a P-, Q-, or S-model environment is said to be absolutely expedient if

$$E[M(n+1) | \beta(n)] < M(n)$$

for all n and all $\beta(n)$.

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 $M(n) < M_0$, absolute expediency implies expediency.

Definition 2.4. A learning automaton operating in a P-, Q-, or S-model environment is said to be optimal if

$$\lim_{n \rightarrow \infty} E[M(n)] = \xi_1$$

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

$$\lim_{n \rightarrow \infty} E[M(n)] \leq \xi_1 + \varepsilon$$

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.

2.2. Variable action-set learning automata

A variable action-set learning automaton is an automaton in which the number of actions available at each instant changes with time. It has been shown in Thathachar and Harita (1987) that a learning automaton with a changing number of actions is absolutely expedient and also ε -optimal, when the reinforcement scheme is L_{RC-1}. Such an automaton has a finite set of n actions, $\underline{a} = \{a_1, a_2, \dots, a_n\}$. $A = \{A_1, A_2, \dots, A_m\}$ denotes the set of action subsets and $A(k) \subseteq A$ is the subset of all the actions that 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 $\Psi(k) = \{\Psi_1(k), \Psi_2(k), \dots, \Psi_m(k)\}$ defined over the possible subsets of the actions, where $\Psi_i(k) = \text{prob}[A(k) = A_i | A_i \in A, 1 \leq i \leq 2^n - 1]$.

$\tilde{p}_i(k) = \text{prob}[a(k) = a_i | A(k)]$, $a_i \in A(k)$ denotes the probability of choosing action a_i , conditioned on the event that the action subset $A(k)$ has already been selected and $a \in A(k)$ too. The scaled probability $\hat{p}_i(k)$ is defined as

$$\hat{p}_i(k) = \frac{p_i(k)}{K_i(k)}$$

where $K_i(k) = \sum_{a_j \in A(k)} p_j(k)$ is the sum of the probabilities of the actions in subset $A(k)$ and $p_i(k) = \text{prob}[a(k) = a_i]$.

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 n . Before choosing an action, the probabilities of all the actions in the selected subset are scaled as defined in equation above. 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. Finally, the probability vector of the actions of the chosen subset is rescaled as $p_i(k+1) = \tilde{p}_i(k+1)K_i(k)$ for all $a \in A(k)$.

2.3. Problem statement

In this section, the multicast routing problem under real MANET scenarios is modeled with the Steiner tree problem in stochastic graphs, where the duration of a communication link is defined as the random weight of its corresponding graph edge. Under such circumstances, the Steiner tree with the maximum expected weight constructs the most stable multicast route of the network. Before stating the multicast routing problem, the following preliminaries and definitions must be presented.

Definition 2.6. Given an edge-weighted graph $G<V, E, W>$, where V is the set of vertices, E is the set of edges, and W denotes the set of weights associated with the graph edges, and a subset of graph vertices, $R \subseteq V$, called terminal set. The Steiner tree problem is defined as finding a tree with the minimum weight connecting the terminals. It must be noted that the Steiner tree may include some vertices other than terminals called the Steiner vertices. The Steiner tree connecting the multicast source to the multicast receivers (as terminals) forms a multicast route through which the multicast messages can be sent with the minimum cost. That is, the multicast tree can be modeled by a Steiner tree.

Definition 2.7. Let $G<V, E, W>$ be a weighted graph, where V denotes the vertex set, E is the edge set, and W is the set of weights associated with the graph edges (or vertices). Graph G is stochastic, if the weight associated with the graph edges (or vertices) is a random variable.

Definition 2.8. Given an edge-weighted stochastic graph $G<V, E, W>$, and a terminal set $R \subseteq V$. Let S be the set of Steiner (multicast) trees connecting the terminals (multicast source and receivers). Let $E[w_s]$ be the expected weight of the Steiner tree $s \in S$. The stochastic Steiner tree problem is defined as finding a tree with the minimum expected weight connecting the terminals. That is, s^* is the solution of the stochastic Steiner tree problem, if $E[w_{s^*}] = \min_{s \in S} E[w_s]$.

In wireless MANETs, due to the host mobility, the communication links connecting the neighboring hosts are extremely fragile. For this reason, the constructed routes may rapidly lose their validity. To enhance the stability of the communication routes, they must be composed of the long duration links. The duration of a link, which is hereafter referred to as link duration time (LDT), is defined as the time interval during which the link is available (or two hosts stay connected). The LDT of a link connecting two hosts increases as the relative mobility between two hosts becomes slower. The relative mobility between two hosts is based on the movement direction and mobility speed of the two hosts. Let v_i^t and θ_i^t denote the mobility speed and movement direction of host h_i at instant t , and v_j^t and θ_j^t be those of mobile host h_j , where $\theta_i^t, \theta_j^t \in [0, 2\pi]$. The relative mobility of host h_i with respect to host h_j (or vice versa) at instant t is defined as

$$[(v_i^t)^2 + (v_j^t)^2 - 2v_i^t v_j^t \cos(\theta_i^t - \theta_j^t)]^{1/2} \quad (3)$$

Let (x_i^t, y_i^t) and (x_j^t, y_j^t) be the coordinates of mobile hosts h_i and h_j at instant t provided by global positioning system (GPS), respectively. Let r denotes the radio-propagation range of two hosts h_i and h_j . If hosts h_i and h_j are within the radio-transmission range of each other at instant t , the LDT of link (h_i, h_j) after instant t , $\text{LDT}_{(i,j)}^t$, is determined as

$$\text{LDT}_{(i,j)}^t = \left\lceil \frac{-(ab+cd) + \sqrt{(a^2+c^2)(b^2+d^2) - (ad-bc)^2}}{(a^2+b^2)} \right\rceil$$

$$\begin{aligned} \text{where} \\ \mathbf{a} &= \mathbf{v}_i^t \cos \theta_i^t - \mathbf{v}_j^t \cos \theta_j^t \\ \mathbf{b} &= \mathbf{x}_i^t - \mathbf{x}_j^t \\ \mathbf{c} &= \mathbf{v}_i^t \sin \theta_i^t - \mathbf{v}_j^t \sin \theta_j^t \\ \mathbf{d} &= \mathbf{y}_i^t - \mathbf{y}_j^t \end{aligned} \quad (4)$$

$\text{LDT}_{(i,j)}^t$ denotes the amount of time two hosts h_i and h_j remain within the propagation range of each other. The links with the smaller LDT are more prone to breakage, and so our proposed multicast routing algorithm aims at finding the multicast tree including the links with the highest LDT (i.e., the most stable links) to extend the route lifetime. The lifetime of a given route is determined as the LDT of the weakest link (i.e., the link with the minimum LDT) connecting two hosts along the route.

In real MANET scenarios, the mobility characteristics of the mobile host (i.e., speed and direction) are stochastic and change in different epochs; so the LDT of the communication links varies with time. Under such circumstances, the instant mobility parameters (i.e., the parameters determined for a single epoch) are not able to exhibit the real mobility behavior of the mobile host. Therefore, to predict the long-term mobility behavior of the host we define the expected link duration time (ELDT) as the average of the instant LDTs over different epochs. An epoch is defined as the period of time during which both the mobility speed and the movement direction of a mobile host are constant. A host experiences a new epoch, if at least one of these parameters changes. Each mobile host h_i broadcasts its location information as well as its average mobility parameters (i.e., \mathbf{v}_i and θ_i) to its neighboring hosts immediately, if it enters a new epoch or it is requested by its neighboring hosts. Each neighboring host h_j computes the ELDT of link (h_i, h_j) on the basis of the recently received information as follows:

$$\mathbb{E}[\text{LDT}_{(i,j)}^T] = \frac{1}{k} \sum_{t=1}^k \text{LDT}_{(i,j)}^t \quad (5)$$

where T is the time interval over which the duration time of link (h_i, h_j) is averaged, and k denotes the number of times each host and its neighbors experience new epochs during time interval T . After time interval T , the LDT that is expected for link (h_i, h_j) at epoch $k+1$ can be determined based on the k samples taken during time interval T . Therefore, as time interval T becomes longer, the ELDT of link (h_i, h_j) estimated by Eq. (5) gets closer to the mean of the PDF defined over the LDT of this link. At each epoch, each host must compute the ELDT of the links connecting it to its neighboring hosts that experience a new epoch and assign it as a weight to the link.

The ELDT of all the links must be updated for the host that experiences a new epoch.

A multicast route $m \in M$, where M is the set of multicast routes connecting a given multicast source to a group of multicast receivers, cannot be used anymore, if at least one of its communication links breaks or fails. Therefore, the duration of a multicast route is degraded to the duration of the weakest link connecting two hosts within the multicast route. At instant t , the expected duration of multicast route m is defined as

$$\mathbb{E}[\text{MDT}_m^t] = \min_{\forall (i,j) \in m} \mathbb{E}[\text{LDT}_{(i,j)}^t] \quad (6)$$

$\mathbb{E}[\text{MDT}_m^t]$ represents the amount of time, which is expected for the multicast route m to remain connected after instant t . This is defined as the minimum duration of a link along the multicast route. $m \in M$ is the most stable multicast route, if it has the maximum duration as

$$\mathbb{E}[\text{MDT}_{m^*}^t] = \max_{\forall m \in M} \mathbb{E}[\text{MDT}_m^t] \quad (7)$$

The multicast routing algorithm we propose in the next section aims to find the (most stable) multicast route with the maximum duration as defined in the equation above for MANET scenarios where the link duration varies over time.

3. Proposed multicast routing algorithm

Finding the Steiner tree of the network topology graph is a well known approach for solving the multicast routing problem in (or for constructing the multicast tree of) ad hoc networks. A deterministic Steiner tree algorithm aims at finding the minimum weight tree connecting the terminals, where the edge weight is assumed to be fixed. The deterministic case of the Steiner tree problem has been well studied and until now several powerful algorithms have been proposed. However, when the edge weight varies with time, the optimum solution of the Steiner tree problem is extremely hard to find. For simplicity, the existing multicast routing algorithms assume that the weight associated with the communication links is fixed. These algorithms do not perform well in realistic networks, where the network parameters such as the throughput of a communication link may change with time as a result of congestion, accidents, or so on. Finding the optimal multicast tree becomes more difficult when the probability distribution function (PDF) of the edge weight is unknown. It is obvious that there is no a priori knowledge of the distribution parameters in ad hoc networks. Therefore in this paper, a stochastic Steiner tree algorithm, in which the weight associated with the communication links is a random variable with unknown PDF, is proposed for constructing the multicast route in ad hoc networks. This paper shows the capabilities of the learning automata for solving such a difficult problem.

As shown in Fig. 1, the proposed multicast routing algorithm, which is called Learning automata-based Link stability Multicast Routing algorithm (LLMR) is composed of a number of iterations, and each iteration starts when a multicast session is requested by the multicast source. The following briefly describes different steps of the k th iteration of the proposed algorithm. In the beginning of an iteration, when the multicast source decides to initiate a multicast session, it calls procedure Multicast Routing (MR) to construct a multicast route, and then waits for receiving the Route REPLY (RREP) message. Based on the stability of the constructed multicast route, multicast source decides to reward or penalize the chosen links by which the route is formed. Let m_k be the multicast route constructed at k th iteration, which is a vector containing the hosts that are located on the way from the multicast source to all the multicast receivers. m_k is initialized by the multicast source to its ID number. Let Ω_{k-1} denotes the dynamic threshold of k th iteration. Ω_k is defined as the expected duration time of the most stable multicast route constructed until iteration k as follows:

$$\Omega_k = \max_{j \leq k} \mathbb{E}[\text{MDT}_{m_j}^t] \quad (8)$$

Ω_k is initially set to zero where $k=0$. If m_k is the most stable multicast route constructed so far (i.e., $\Omega_k \geq \Omega_{k-1}$), the hosts along the multicast route m_k are rewarded by a ReWARDing (RWD) message.

As mentioned earlier, host h terminates the current iteration and returns the information of the constructed route to the multicast source by a RREP message, if the selected route comprises all the multicast receivers. When a host receives a RREP message, it calls procedure Route REPLY (RREP). This

Fig. 1. Pseudo-code of the proposed multicast routing algorithm.

As mentioned earlier, host h terminates the current iteration and returns the information of the constructed route to the multicast source by a RREP message, if the selected route comprises all the multicast receivers. When a host receives a RREP message, it calls procedure Route REPLY (RREP). This

Fig. 2. Pseudo-code of procedure MR.

As mentioned earlier, host h terminates the current iteration and returns the information of the constructed route to the multicast source by a RREP message, if the selected route comprises all the multicast receivers. When a host receives a RREP message, it calls procedure Route REPLY (RREP). This

procedure aims at sending the selected route information back to the multicast source through the path connecting host h_i to multicast source s . It must be noted that the hosts, which are not along the above mentioned path, are not involved in procedure RREP. Therefore, when a host receives the RREP message, it first checks to see if its ID number matches to the receiver ID. If so, it resends the RREP message along its incoming link stored in RT_{in} , otherwise it does nothing. Figure 3 shows procedure RREP, which is run at host h_i . As shown in lines 13–15 of algorithm LLMR, the multicast source assesses the stability of the selected route and rewards or penalizes it on receiving the RREP message.

```

Procedure RREP( $h_i$ )
1: Receive RREP message
2: Reply procedure
3: If  $h_i \neq s$  then
4:   If  $h_i \neq receiver\_ID$  then
5:      $h_i \leftarrow receiver\_ID$ 
6:    $h_i$  resends RREP message to  $h_i$  along link  $(h_i, h_j)$ 
7: End procedure

```

Fig. 3. Pseudo-code of procedure RREP.

As shown in lines 11 and 12 of procedure MR, from the link (action) probability vector, host h_i sends a RREQ message to all its one hop neighboring hosts. The hosts that are within the transmission range of the sender host (i.e., hear the RREQ message) reply the RREQ message on receiving it. The reply message sent by h_j includes the host's current location information (i.e., $[x_j, y_j]$) as well as the average mobility speed and movement direction (i.e., \bar{v}_j and $\bar{\theta}_j$) of the host as described in Section 2.2. The sender host h_i waits for a period of time for receiving the reply messages and then calls procedure Link-Set-Formation to construct the link-set and define the link probability vector on the basis of the received information. In the proposed multicast routing algorithm, each action of a host (or of the learning automaton corresponding to the host) is associated with one of its incident links. Therefore, the action might be referred to as link.

In Link-Set-Formation procedure, for each passive host h_j , the initial link-set is formed as follows: let $L_{h_i}(k)$ be the link-set of host h_i at iteration k . For all hosts h_j replying the RREQ message sent out by h_i , communication link, (h_i, h_j) is added to $L_{h_i}(k)$. Let us assume that $\Delta(k)$ denotes the degree of host, h_i at iteration k (i.e., the number of hosts that are adjacent to host, h_i). The initial probability of choosing each communication link (h_i, h_j) is set to $1/\Delta(k)$. In mobile ad hoc networks, the network topology frequently changes and so the link-set of the mobile hosts rapidly loses its validity. Therefore, the link-set of the activated hosts must be updated at each iteration as follows. Link (h_i, h_j) must be removed from link-set $L_{h_i}(k)$, if h_j is not within the radio-propagation range of host h_i (or h_j does not reply the RREQ message) anymore (see lines 12–17). Link (h_i, h_j) must be added to link-set $L_{h_i}(k)$, if it did not already belong to $L_{h_i}(k)$, but h_j now replies the RREQ message (see lines 18–23). Link (h_i, h_j) must be retained if host h_j is still replying the RREQ message. In each case, the link probability vector of the hosts is updated as shown in Fig. 4 as the case may be.

```

Procedure Link-Set-Formation( $h_i, h_j$ )
1: Initialize
2: Set  $L_{h_i}(k)$  to the link probability vector of host  $h_i$  at iteration  $k$ 
3: Set  $\Delta(k)$  to the degree of host  $h_i$  at iteration  $k$ 
4: Set  $p_{h_i}(k)$  to the action probability of link  $(h_i, h_j)$  at iteration  $k$ 
5: Reply procedure
6: If  $h_j$  is not within the radio-propagation range of host  $h_i$  then
7:   Remove link  $(h_i, h_j)$  from  $L_{h_i}(k)$ 
8:    $L_{h_i}(k) \leftarrow L_{h_i}(k) - (h_i, h_j)$ 
9:    $\Delta(k) \leftarrow \Delta(k) - 1$ 
10: Else
11:    $L_{h_i}(k) \leftarrow L_{h_i}(k) \cup (h_i, h_j)$ 
12:    $\Delta(k) \leftarrow \Delta(k) + 1$ 
13:   If  $h_j$  does not reply RREQ message then
14:      $L_{h_i}(k) \leftarrow L_{h_i}(k) - (h_i, h_j)$ 
15:      $\Delta(k) \leftarrow \Delta(k) - 1$ 
16:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
17:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
18:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
19:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
20:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
21:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
22:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
23:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
24:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
25:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
26:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
27:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
28:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
29:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
30:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
31:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
32:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
33:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
34:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
35:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
36:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
37:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
38:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
39:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
40:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
41:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
42:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
43:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
44:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
45:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
46:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
47:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
48:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
49:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
50:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
51:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
52:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
53:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
54:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
55:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
56:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
57:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
58:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
59:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
60:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
61:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
62:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
63:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
64:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
65:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
66:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
67:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
68:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
69:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
70:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
71:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
72:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
73:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
74:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
75:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
76:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
77:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
78:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
79:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
80:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
81:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
82:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
83:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
84:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
85:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
86:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
87:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
88:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
89:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
90:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
91:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
92:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
93:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
94:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
95:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
96:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
97:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
98:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
99:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
100:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
101:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
102:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
103:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
104:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
105:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
106:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
107:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
108:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
109:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
110:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
111:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
112:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
113:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
114:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
115:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
116:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
117:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
118:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
119:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
120:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
121:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
122:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
123:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
124:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
125:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
126:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
127:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
128:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
129:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
130:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
131:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
132:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
133:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
134:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
135:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
136:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
137:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
138:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
139:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
140:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
141:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
142:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
143:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
144:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
145:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
146:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
147:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
148:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
149:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
150:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
151:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
152:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
153:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
154:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
155:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
156:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
157:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
158:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
159:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
160:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
161:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
162:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
163:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
164:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
165:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
166:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
167:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
168:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
169:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
170:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
171:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
172:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
173:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
174:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
175:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
176:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
177:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
178:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
179:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
180:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
181:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
182:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
183:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
184:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
185:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
186:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
187:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
188:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
189:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
190:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
191:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
192:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
193:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
194:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
195:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
196:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
197:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
198:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
199:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
200:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
201:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
202:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
203:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
204:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
205:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
206:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
207:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
208:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
209:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
210:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
211:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
212:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
213:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
214:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
215:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
216:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
217:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
218:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
219:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
220:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
221:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
222:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
223:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
224:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
225:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
226:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
227:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
228:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
229:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
230:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
231:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
232:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
233:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
234:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
235:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
236:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
237:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
238:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
239:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
240:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
241:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
242:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
243:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
244:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
245:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
246:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
247:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
248:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
249:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
250:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
251:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
252:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
253:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
254:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
255:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
256:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
257:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
258:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
259:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
260:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
261:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
262:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
263:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
264:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
265:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
266:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
267:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
268:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
269:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
270:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
271:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
272:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
273:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
274:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
275:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
276:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
277:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
278:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
279:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
280:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
281:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
282:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
283:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
284:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
285:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
286:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
287:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
288:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
289:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
290:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
291:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
292:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
293:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
294:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
295:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
296:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
297:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
298:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
299:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
300:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
301:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
302:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
303:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
304:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
305:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
306:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
307:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
308:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
309:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
310:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
311:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
312:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
313:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
314:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
315:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
316:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
317:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
318:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
319:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
320:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
321:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
322:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
323:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
324:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
325:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
326:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
327:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
328:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
329:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
330:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
331:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
332:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
333:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
334:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
335:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
336:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
337:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
338:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
339:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
340:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
341:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
342:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
343:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
344:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
345:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
346:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
347:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
348:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
349:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
350:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
351:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
352:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
353:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
354:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
355:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
356:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
357:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
358:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
359:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
360:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
361:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
362:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
363:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
364:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
365:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
366:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
367:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
368:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
369:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
370:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
371:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
372:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
373:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
374:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
375:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
376:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
377:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
378:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
379:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
380:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
381:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
382:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
383:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
384:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
385:      $p_{h_i}(k) \leftarrow \frac{1}{\Delta(k)}$ 
386:   For all  $(h_i, h_j) \in L_{h_i}(k)$  do
387:      $p_{h_i}(k) \left$ 
```

Fig. 5. Pseudo-code of procedure RWD.

4. Experimental results

To study the performance of the proposed multicast routing algorithm, several simulation experiments have been conducted in NS2. In all these experiments, a mobile ad hoc network is modeled by randomly and uniformly distributing 100 mobile hosts within a square simulation area of size 1000(m)×1000(m). The mobility model of each host is assumed to be the random waypoint model (Camp et al., 2002). The maximum mobility speed of each host changes from 10 to 70 km/h, and multicast group size ranges from 5 to 30. It is assumed that the mobility characteristics change at the beginning of each epoch. IEEE 802.11 Distributed Coordination Function (IEEE Computer Society LAN MAN Standards Committee, 1997) (DCF) with Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) is used as the medium access control protocol, and two ray ground as the propagation model. The communication channel through which the wireless hosts communicate is a common broadcast channel of capacity 2 mb/s. All mobile hosts are equipped with omnidirectional antennas and have the same radio-propagation range 100 m. Constant Bit Rate (CBR) traffic sources are used to generate the traffics with a rate of 20 p/s (packet per second). The packet size is set to 512 bytes. Each host is modeled as a store-and-forward queuing station and assumed to be aware of its mobility information with the aid of a reliable positioning system. Each simulation experiment is run for 1800 s, and the obtained results are averaged over 50 different runs. For our proposed algorithm, the reward and penalty parameters are set to 0.1 and 0, respectively. Computer simulations are conducted in two groups. The former group studies the impact of the host speed on the performance of multicasting algorithms where the multicast group size is fixed at 10 and host speed changes from 10 to 70 km/h, and the second group of experiments is conducted to evaluate the effect of the multicast group size on the efficiency of algorithms. In these experiments, the host speed is set to 10 km/h and the multicast group size changes from 5 to 30.

To show the efficiency of the proposed multicast routing algorithm, its results are compared with those of E-ODMRP (an enhanced version of ODMRP with motion adaptive refresh proposed by Oh et al. (2008)) and LSMRM (a link stability-based multicast routing algorithm for MANET proposed by Biradar et al. (2010)), which are best existing mobility-based multicast routing protocols. In conducted experiments, the performance of the multicast routing algorithms is measured in terms of the packet delivery ratio, multicast route lifetime, control message overhead, and end-to-end delay.

4.1. Packet delivery ratio

Packet delivery ratio is defined as the average number of packets received at multicast receivers over the number of packets sent out by the multicast source. That is, this metric is defined as the number of packets delivered to the multicast receivers over the number of packets supposed to be received by the multicast receivers. Packet delivery ratio implies the efficiency of the routing technique in a multicasting protocol. Figure 6 shows the results of the first set of experiments conducted to study the impact of the host mobility speed on the packet delivery ratio. In this experiment, the multicast group size is set to 10 and the maximum mobility speed changes from 10 to 70 km/h with increment step of 10. Figure 6 shows the packet delivery ratio against the mobility speed. From the results shown in Fig. 6 it is observed that the packet delivery ratio of all multicast routing protocols decreases as the host speed increases. This is due to the fact that the multicast routes are more prone to breakage as the host speed increases, and so the packet delivery ratio reduces when the probability of link failure increases. Comparing the obtained results we find that LLMR (the proposed multicast routing algorithm) has the highest packet delivery ratio, LSMRM lags far behind it, and E-ODMRP has the worst results. This is because E-ODMRP does not consider the route stability and so it has the highest packet drop rate. LSMRM uses the instant link stability information to find the stable paths, and so it cannot predict the long-term behavior of the links. As expected LLMR has the highest delivery rate because it estimates the expected duration time of the routes over different epochs. Figure 7 shows the packet delivery ratio versus the multicast group size. In this experiment, the maximum host speed is fixed at 10 km/h and multicast group size changes from 5 to 30 with increment step of 5. Comparing the curves depicted in Fig. 7, we find that the ranking given in Fig. 6 remains unchanged here and the proposed algorithm outperforms the others and E-ODMRP produces the lowest delivery rate. Figure 7 also shows that the packet delivery ratio slightly increases as the multicast group size increases. This is because the average number of routes to different destinations increases when the multicast group size grows. As a result, the packet drop rate decreases when there exist several paths to deliver data packets to the multicast receivers.

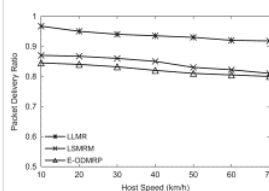


Fig. 6. Packet delivery ratio as a function of host speed (multicast group size=10).

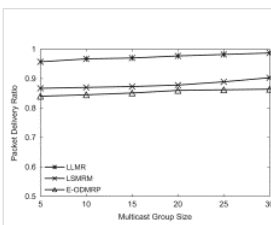


Fig. 7. Packet delivery ratio versus the multicast group size (host speed=10 km/h).

4.2. Multicast route lifetime

Multicast route lifetime is defined as the period of time during which the multicast routes remain connected. This metric is also called multicast route duration and shows the stability of the multicast paths. Multicast route lifetime is measured in second and substantially reduces as the host speed becomes faster. This metric represents the resistance of the multicast routing algorithm against the host mobility and network dynamics. In this section, we conducted several simulation experiments to show the impact of the host speed and multicast group size on the multicast route lifetime. Figure 8 shows the changes in the multicast route lifetime as the host speed increases. As expected, the multicast route duration becomes shorter when the host speed becomes faster. The neighboring hosts leave the propagation range of each other rapidly if the host speed increases. Higher mobility speed makes the communication links more fragile. This reduces the route stability and shortens the multicast route lifetime. From the results shown in Fig. 8, it is obvious that LLMR significantly outperforms the other protocols, and the lifetimes of the multicast routes constructed by LSMRM are very longer in comparison with those of E-ODMRP. This is because E-ODMRP does not consider the route stability and so it has the highest packet drop rate. LSMRM uses the instant information of the communication links to find the stable paths, and so it cannot predict the long-term behavior of the links. On contrary to LSMRM, in the proposed algorithm it is assumed that the route duration changes over time and so the most stable route is defined as the route with the maximum expected duration for a long time. LLMR is capable of predicting the long-term behavior of the communication links and therefore has a higher rate of packet delivery as expected. Figure 9 exhibits the multicast route lifetime as a function of the multicast group size. Comparing the results depicted in this figure no significant changes can be seen in the multicast route lifetime as the multicast group size becomes larger. Like the results given in Fig. 8, these results confirm the superiority of LLMR over LSMRM and E-ODMRP in terms of route lifetime.

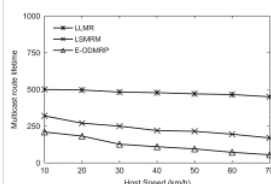


Fig. 8. Multicast route lifetime versus the host speed (multicast group size=10).

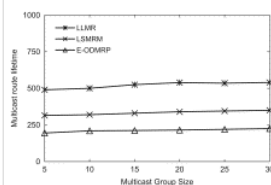


Fig. 9. Multicast route lifetime versus the multicast group size (host speed=10 km/h).

4.3. End-to-end delay

End-to-end delay is defined as the average time elapsed between sending the packets at the multicast source and receiving at all the multicast destinations. It should be noted that the time spent for building a route (route acquisition latency) is also included in the end-to-end delay, if no multicast route is available.

That is, this metric is defined as the time required for multicast route creation (so called route creation delay) as well as the time required for transmitting the multicast packets along the constructed paths (so called transmission delay). Therefore, we first measure the route creation delay and transmission delay separately, and then end-to-end delay as their combination. Fig. 10 and Fig. 11 show the average route creation delay versus the host mobility and multicast group size, respectively. From the results depicted in these figures, it can be seen that the delay of LSMRM is shorter than that of E-ODMRP and LLMR. The results also show that the delay of E-ODMRP is very close to that of LLMR. In route discovery phase, LLMR tries to find the multicast routes with the maximum stability against the host mobility, and this results in the longer route creation delay of LLMR. We also conducted several experiments to assess the transmission delay of the multicasting protocols. The obtained results are shown in Fig. 12 and Fig. 13. Comparing the curves depicted in these figures, we observe that the average transmission time of LLMR is significantly shorter than LSMRM and E-ODMRP. This is because the proposed protocol sends the multicast packets along the most stable and reliable routes. This significantly reduces the route failure rate and so shortens the average time required for switching to another route. To assess the total delay of the multicast routing protocols, we conducted another set of simulation experiments to measure the end-to-end delay as a combination of route creation and transmission delays. End-to-end delay is shown as a function of host speed in Fig. 14. Comparing the results shown in this figure, it can be seen that LLMR has the shortest end-to-end delay and E-ODMRP has the longest one. As mentioned earlier, LLMR is aimed at finding the most stable routes in route discovery phase. This considerably reduces the rate of link breakage resulting from host mobility. In other words, the route failure probability of LLMR is much lower than that of E-ODMRP and LSMRM. Reduction in link failure rate decreases the route reconstruction frequency (or the number of times the multicast routes must be reconstructed due to route failure). As shown in Fig. 10 and Fig. 11, route creation process is the main source of the delay in LLMR, and so low route reconstruction frequency shortens the end-to-end delay. That is why LLMR outperforms the others in terms of end-to-end delay. From the results given in Fig. 8 and Fig. 9, we know that the duration of the route constructed by LSMRM is significantly shorter compared to LLMR. As a result, the multicasting delay in LSMRM is longer than that of LLMR. Obviously, the long delay of E-ODMRP is due to the fact that it does not consider the stability issues. The results given in Figs. 10, 12, and 14 show that the delay of all protocols increases as the host speed increases. This is because the probability of link breakage and consequently the route reconstruction time increases when the hosts move faster. Figure 15 shows the end-to-end delay versus the multicast group size. The curves depicted in this figure show that the delay increases as the number of multicast receivers increases. As shown in Fig. 13 and Fig. 15, the time required for creation (or reconstruction) of a route and the data transmission time increase as the multicast group size increases. Therefore, end-to-end delay is directly proportional to the number of multicast receivers.

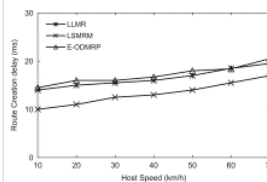


Fig. 10. Route creation delay versus the host speed (multicast group size=10).

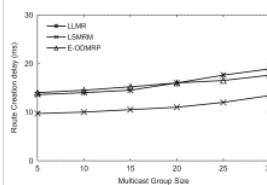


Fig. 11. Route creation delay versus the multicast group size (host speed=10 km/h).

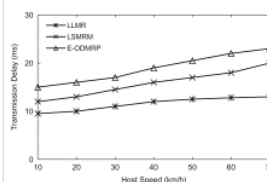


Fig. 12. Transmission delay versus the host speed (multicast group size=10).

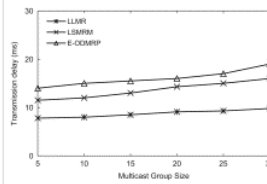


Fig. 13. Transmission delay versus the multicast group size (host speed=10 km/h).

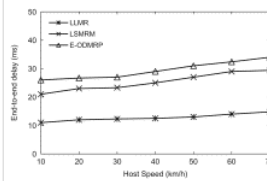


Fig. 14. End-to-end delay as a function of host speed (multicast group size=10).

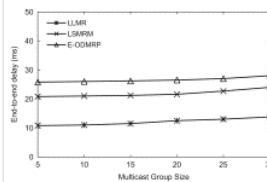


Fig. 15. End-to-end delay as a function of multicast group size (host speed=10 km/h).

4.4. Control message overhead

Control message overhead is defined as the number of (extra) control messages required for finding the multicast route. This metric is measured as the number of control messages that must be sent per second. In this section, we measure the control message overhead of the multicast routing protocols and study the impact of the host mobility speed and multicast group size on this metric. In the first group of experiments, we measure the control message overhead of different algorithms where the host speed ranges from 10 to 70 km/h and the number of multicast receivers is fixed at 5. The obtained results are shown in Fig. 16. The results show that the control message overhead of all protocols increases as the host speed increases. This is because the extra control messages required for frequent route reconstruction (due to link breakage) increases when the hosts move faster. Comparing the results shown in Fig. 16, it is observed that LLMR dramatically outperforms the others in terms of the message overhead and the number of extra messages in LSMRM is much smaller than that of E-ODMRP. This is because LLMR constructs the most stable multicast routes with the longest duration in contrast with E-ODMRP and LSMRM, and this reduces the rate of route reconstruction due to link breakage. Clearly the reduction in route reconstruction rate reduces the rate of extra control messages. Figure 17 shows the control message overhead as a function of multicast group size. Comparing the results, we find that the message overhead in all protocols increases as the number of multicast members increases. It is clear that the number of control messages required for creation or reconstruction of a multicast route increases as the number of multicast receivers that must be covered (by the constructed route) increases.

[Full-size image](#)

Fig. 16. Control message overhead versus the host speed (multicast group size=10).

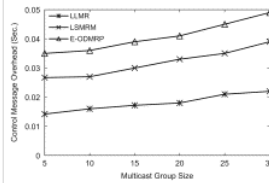


Fig. 17. Control message overhead versus the multicast group size (host speed=10 km/h).

5. Conclusion

The multicast routing problem in a communication network can be efficiently modeled by the Steiner tree problem of the network topology in graph theory. Solving the stochastic version of the Steiner tree problem in which the edge weight varies over time is a promising approach for finding the multicast route in dynamic MANET environments. In this paper, we proposed a weighted learning automata-based multicast routing protocol for wireless MANETs in which the multicast packets are forwarded along the Steiner tree links. In the proposed method, the expected duration time of a communication link (which is the period of time during which a link is expected to be connected) is defined as its weight, which is a random variable with unknown distribution. The expected duration time of a multicast route is defined as the expected duration time of the weakest link. The multicast routes with longer expected duration time are more stable against the host mobility. The aim of the proposed algorithm is to find the most stable route with the maximum duration among all the possible multicast paths. At each iteration of the proposed algorithm a multicast route is constructed by finding a random solution of the stochastic Steiner tree problem in the network topology graph. The constructed multicast route is rewarded, if its expected duration time is longer than those of the previous iterations and it is penalized otherwise. The choice probability of the most stable multicast route converges to one as the proposed algorithm proceeds. To show the efficiency of the proposed algorithm, several computer simulations have been conducted and the obtained results were compared with those of E-ODMRP (Oh et al., 2008) and LSMRM (Biradar et al., 2010). The results showed that the proposed method outperforms the others in terms of the packet delivery ratio, multicast route lifetime, control message overhead, and end-to-end delay.

References

- Aggarwal et al., 2006 D. Aggarwal, C.K. Dubey, S.K. Mehta
Algorithms on graphs with small dominating targets
ISAAC 2006, LNCS, 4288 (2006), pp. 141–152
- An and Papavassiliou, 2003 B. An, S. Papavassiliou
MHMR: mobility-based hybrid multicast routing protocol in mobile ad hoc wireless networks
Wireless Communication and Mobile Computing, 3 (2003), pp. 255–270
- Akbari Torkestani and Meybodi, in press-a Akbari Torkestani J, Meybodi MR. A learning automata-based heuristic algorithm for solving the minimum spanning tree problem in stochastic graphs. Journal of Supercomputing. Springer Publishing Company; in press-a, doi:10.1007/s11227-010-0484-1
- Akbari Torkestani and Meybodi, in press-b Akbari Torkestani J, Meybodi MR. Learning automata-based algorithms for solving stochastic minimum spanning tree problem. Journal of Applied Soft Computing; in press -b, doi:10.1016/j.asoc.2011.02.017
- Akbari Torkestani and Meybodi, 2010a J. Akbari Torkestani, M.R. Meybodi
Learning automata-based algorithms for finding minimum weakly connected dominating set in stochastic graphs
International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 18 (6) (2010), pp. 721–758
- Akbari Torkestani and Meybodi, 2010b J. Akbari Torkestani, M.R. Meybodi
A New Vertex Coloring Algorithm Based on Variable Action-Set Learning Automata
Journal of Computing and Informatics, 29 (3) (2010), pp. 1001–1020
- Akbari Torkestani and Meybodi, in press-c

Akbari Torkestani J, Meybodi MR. A Cellular learning automata-based algorithm for solving the vertex coloring problem. *Journal of Expert Systems with Applications*; in press-c, doi:10.1016/j.eswa.2011.01.098



[Akbari Torkestani and Meybodi, 2011](#) J. Akbari Torkestani, M.R. Meybodi

A learning automata-based cognitive radio for clustered wireless ad-hoc networks
Journal of Network and Systems Management, 19 (2) (2011), pp. 278–297



[Akbari Torkestani and Meybodi, 2010c](#) J. Akbari Torkestani, M.R. Meybodi

Mobility-based multicast routing algorithm in wireless mobile ad hoc networks: a learning automata approach
Journal of Computer Communications, 33 (2010), pp. 721–735



[Akbari Torkestani and Meybodi, in press-d](#) Akbari Torkestani J, Meybodi MR. Weighted Steiner connected dominating set and its application to multicast routing in wireless MANETs. *Wireless Personal Communications*, Springer Publishing Company; in press-d, doi:10.1007/s11277-010-9936-4



[Akbari Torkestani and Meybodi, 2010d](#) J. Akbari Torkestani, M.R. Meybodi

An efficient cluster-based CDMA/TDMA scheme for wireless mobile ad-hoc networks: a learning automata approach
Journal of Network and Computer applications, 33 (2010), pp. 477–490



[Akbari Torkestani and Meybodi, 2010e](#) J. Akbari Torkestani, M.R. Meybodi

Clustering the wireless ad-hoc networks: a distributed learning automata approach
Journal of Parallel and Distributed Computing, 70 (2010), pp. 394–405



[Akbari Torkestani and Meybodi, 2010f](#) J. Akbari Torkestani, M.R. Meybodi

An intelligent backbone formation algorithm in wireless ad hoc networks based on distributed learning automata
Journal of Computer Networks, 54 (2010), pp. 826–843



[Akbari Torkestani and Meybodi, in press-e](#) Akbari Torkestani J, Meybodi MR. A mobility-based cluster formation algorithm for wireless mobile ad hoc networks. *Journal of Cluster Computing*. Springer Publishing Company; in press-e, doi:10.1007/s10586-011-0161-z



[Biradar et al., 2010](#) R. Biradar, S. Manvi, M. Reddy

Link stability based multicast routing scheme in MANET
Computer Networks, 54 (2010), pp. 1183–1196



[Chiang et al., 2007](#) T.C. Chiang, C.H. Liu, Y.M. Huang

A near-optimal multicast scheme for mobile ad hoc networks using a hybrid genetic algorithm
Expert Systems with Applications, 33 (2007), pp. 734–742



[Chiang et al., 1998](#) C.C. Chiang, M. Gerla, L. Zhang

Forwarding group multicast protocol (FGMP) for multihop, mobile wireless networks
Journal of Cluster Computing, 1 (2) (1998), pp. 187–196



[Camp et al., 2002](#) T. Camp, J. Boleng, V. Davies

A survey of mobility models for ad hoc network research
Wireless Communication and Mobile Computing, 2 (2002), pp. 483–502



[Dai and Wu, 2005](#) Q. Dai, J. Wu

Computation of minimal uniform transmission range in ad hoc wireless networks
Cluster Computing, 8 (2005), pp. 127–133



[Guo and Yang, 2008](#) S. Guo, O. Yang

Maximizing multicast communication lifetime in wireless mobile ad hoc networks
IEEE Transactions on Vehicular Technology, 57 (2008), pp. 2414–2425



[IEEE Computer Society LAN MAN Standards Committee, 1997](#) IEEE Computer Society LAN MAN Standards Committee

Wireless LAN medium access protocol (MAC) and physical layer (PHY) specification, IEEE Standard 802.11-1997
The Institute of Electrical and Electronics Engineers, New York (1997)



[Karp, 1972](#) R.M. Karp

Reducibility among combinatorial problems, complexity of computer computations
Plenum Press, USA (1972) p. 85–103



[Lee et al., 1999](#) Lee SJ, Gerla M, Chiang CC. On-demand multicast routing protocol. In: Proceedings of the IEEE WCNC'99. New Orleans, LA; 1999. p. 1298–302.



[McDonald and Znati, 2004](#) A.B. McDonald, T.F. Znati
Statistical estimation of link availability and its impact on routing in wireless ad hoc networks,
Wireless Communications and Mobile Computing, 4 (2004), pp. 331–349



[Narendra and Thathachar, 1989](#) K.S. Narendra, K.S. Thathachar
Learning automata: an introduction
Prentice-Hall, New York (1989)



[Oh et al., 2008](#) S.Y. Oh, J.S. Park, M. Gerla
E-ODMRP: enhanced ODMRP with motion adaptive refresh
Journal of Parallel and Distributed Computing, 68 (2008), pp. 1044–1053



[Rodolakis et al., 2008](#) G. Rodolakis, A. Laouiti, P. Jacquet, A.M. Naimi
Multicast overlay spanning trees in ad hoc networks: capacity bounds protocol design and performance evaluation
Computer Communications, 31 (2008), pp. 1400–1412



[Su et al., 2001](#) W. Su, S.J. Lee, M. Gerla
Mobility prediction and routing in ad hoc wireless networks
International Journal of Network Management, 11 (2001), pp. 3–30



[Toh and Bunchua, 2001](#) C.K. Toh, S. Bunchua
Ad hoc mobile multicast routing using the concept of long-lived routes
Wireless Communications and Mobile Computing, 1 (2001), pp. 361–379



[Toh, 1997](#) CK Toh
Associativity-based routing for ad hoc mobile networks
Journal on Wireless Personal Communications (1997)



[Thathachat and Sastry, 1997](#) M.A.L. Thathachat, P.S. Sastry
A hierarchical system of learning automata that can learn the globally optimal path
Information Science, 42 (1997), pp. 743–766



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



[Wu et al., 2004](#) Y.F. Wu, Y.L. Xu, C.L. Chen, K. Wang
On the construction of virtual multicast backbone for wireless ad hoc networks
IEEE International Conference on Mobile Ad hoc and Sensor Systems (2004), pp. 294–303



 Corresponding author.

Copyright © 2011 Elsevier Ltd. All rights reserved.

This document was created with Win2PDF available at <http://www.daneprairie.com>.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.