Wireless Netw (2011) 17:1745–1761
DOI 10.1007/s11276-011-0375-2

NARD: Neighbor-assisted route discovery in MANETs

J. Gomez · V. Rangel · M. Lopez-Guerrero · M. Pascoe

Published online: 7 August 2011

Abstract Reactive routing protocols for mobile ad-hoc networks usually discover routes by disseminating control packets across the entire network; this technique is known as brute-force flooding. This paper presents ***NARD***, which stands for **n**eighbor-**a**ssisted **r**oute **d**iscovery protocol for mobile ad-hoc networks. In NARD, a source node floods a limited portion of the network searching not only for the destination node, but also for routing information related to other nodes (called *destination-neighbors*) that were near the destination node recently. Destination-neighbors can be used as anchor points where a second limited flooding takes place in search for the destination node. Because only two limited portions of the network are flooded by control packets near the source and destination nodes, NARD can significantly reduce signaling overhead due to route-discovery compared with other proposals. Simulations with NS-2 were carried out to verify the validity of our approach.

Keywords Routing protocols flooding · Ad-hoc networks · MANET

J. Gomez (✉) · V. Rangel
Department of Electrical Engineering, National Autonomous University of Mexico, Mexico City, Mexico
e-mail: javierg@fi-b.unam.mx

V. Rangel
e-mail: victor@fi-b.unam.mx

M. Lopez-Guerrero · M. Pascoe
Department of Electrical Engineering, Metropolitan Autonomous University, Mexico City, Mexico
e-mail: milo@xanum.uam.mx

M. Pascoe
e-mail: mpascoe@xanum.uam.mx

1 Introduction

A wireless ad-hoc network is a collection of wireless nodes with decentralized administration and self-configuring capabilities. In mobile ad-hoc networks (MANETs), nodes are free to move. Because nodes have a limited transmission range, it is usually necessary that some nodes participate in the routing process by relaying packets among source-destination pairs. This type of routing is also known as multi-hop routing. Since the origins of ad-hoc networking in the 1970s, finding routes among nodes has been a relevant research challenge. In the past years, there has been a significant amount of research going on in this area, e.g., [5, 10, 17].

Routing protocols for ad-hoc networks may be divided as proactive or reactive. Proactive protocols discover routes in advance from any node to all the other nodes in the network, and these routes are periodically updated as changes occur. Proactive protocols have the main advantage that whenever a node needs to send a packet to another node, there is a route already available. Reactive protocols, on the other hand, discover routes on demand, only when they are needed. This operation adds a delay while a route is found. The actual signaling overhead incurred by routing protocols depends on various factors such as network size, node density, node mobility, and traffic load. In the literature, we can find several studies related to performance analyses of proactive and reactive routing protocols under different network conditions or particular scenarios, e.g., [13, 14, 23].

Many routing protocols under consideration by the MANET Group of the IETF are reactive. Routing protocols in this category such as DSR [15], AODV [29], DYMO [2] and others discover a route only when there is a need for it (this is the so called route-discovery phase of a reactive

routing protocol). Once a route is found, data packets can then be forwarded from source to destination. When a route is no longer valid but still required, it must be repaired in order to maintain the flow of packets (route-maintenance phase). Sometimes a route can be repaired locally; however, a local repair is not always feasible and a new route-discovery procedure becomes necessary.

In this paper we focus on the route-discovery phase of reactive routing protocols for ad-hoc networks. Although there are several improved proposals for route-discovery (some of which are discussed later in this paper), the brute-force flooding technique remains as the most widely used protocol in practice. Brute-force flooding operates as follows. A source node transmits a control packet, called *route-request*, to announce its intention to communicate with a certain destination node. Nodes overhearing this route-request will retransmit it, thus increasing the scope of the search. This simple mechanism has the effect of flooding the entire network with control packets from the source node. Because a node may overhear the same route-request packet, but from different nodes, each packet has a unique ID number so that nodes retransmit a route-request only once. Brute-force flooding is adequate for small low-density networks, otherwise it could generate a prohibitive number of control packets. These control packets compete with data packets for the available bandwidth.

This paper presents NARD, an efficient route-discovery protocol for wireless ad-hoc networks. NARD is intended for large ad-hoc networks where traditional flooding is not a practical solution. In NARD, a source node floods a limited portion of the network searching not only for the destination node, but also for information related to other nodes (called *destination-neighbors*) that were near the destination node recently. Such neighbor nodes can be used as anchor points where a second limited flooding takes place in search for the destination node. Because only two limited portions of the network are flooded by control packets, NARD can significantly reduce the signaling overhead of route-discovery procedures compared with brute-force flooding techniques.

The structure of this paper is as follows: Section 2 discusses previous work in this area. In Sect. 3 we present the motivation and a detailed description of NARD. Section 4 provides some guidelines to select the appropriate scopes of the first and second flooding that guarantee with high probability that the destination node will be found and the signaling overhead will be minimized. In Sect. 5 we discuss some considerations related to the costs of implementing the proposal. Section 6 presents a performance evaluation of NARD in a network simulator along with a comparison with brute-force flooding and FRESH [5]. Finally, in Sect. 7, we present some concluding remarks.

2 Related work

In the past years, the work developed by the MANET group of the IETF represents the baseline work in this area of research. As we mentioned before, most MANET routing protocols rely on brute-force flooding to discover routes. During brute-force flooding all nodes are required to retransmit a route-request packet once. Even so, for dense networks we could have various retransmissions of the same route-request over the same location. These retransmissions, however, are unnecessary since they only increase channel contention and collision rate in the network. This inefficiency of brute-force flooding is known in the literature as the *broadcast storm problem* [25]. An ideal flooding would be one where only one route-request packet is overheard at any particular location of the network, and yet the entire network is covered with route-request packets.

There are various techniques known as *efficient flooding* which target a reduction of signaling packets due to the broadcast storm problem. Some of these techniques use heuristics, e.g., [25, 31] where, for example, a node decides to retransmit a route-request only after a random time (in order to reduce collisions) or only if the number of duplicated route-requests it has received is below a certain threshold. Other proposals for efficient flooding use topological information about neighbors to reduce the number of retransmissions, e.g., [6, 32]. In [21] a node retransmits a route-request only if it determines that retransmissions of the same request by other nodes have not reached all of its neighbors. SPAN [3] and MPR [30] use 2-hop topology information so that only the dominant set of nodes within the 2-hop area retransmits route-request packets. In [9] and [26] a source tree around the source node is constructed; nodes retransmit a route-request only if they are not leaves of the tree.

In previous proposals for efficient flooding the entire network is covered with route-request packets employing fewer control packets than brute-force flooding. There are other proposals that, rather than reducing the broadcast storm problem, attempt to reduce the portion of the network where flooding takes place. These proposals attempt to contain the flooding in certain areas, where it is likely to obtain useful information about the destination, and avoid flooding areas where no useful information is likely to be retrieved. The simplest of these techniques would be one that limits the scope of the route-discovery to N hops only [15]. The rationale here is that if a node manages to find the destination node within this limited search, then a big reduction in signaling traffic can be achieved. However, in case the destination node is not found within this limited search, a full search becomes necessary thus generating even more signaling, plus the increased delay involved in finding the destination node with two searches.

An example of a route-discovery protocol that only floods specific areas of the network is FRESH [5]. In FRESH, nodes keep a record of their most recent encounter times with *all* nodes. Instead of searching for the destination node directly, the source node searches in a limited portion of the network for any node that has encountered the destination node more recently than the source node itself. This procedure continues until the destination is finally reached. Because FRESH and NARD try to reduce the scope of the flooding (as opposed to limit the impact of the broadcast storm problem) we compare their performance in Sect. 6.1 It is important to mention that all techniques for efficient flooding previously described can be used in parallel with FRESH and NARD, resulting in even a larger reduction of signaling during route-discovery.

Other proposals intend to reduce signaling overhead by applying different routing schemes for neighbor nodes or distant nodes. For instance, using proactive protocols for routing operations inside a local zone (internal-zone routing) and reactive protocols for outer-zone routing, e.g., ZRP [12] and AOZDV [18]. Some proposals adapt the frequency of link-state updates among nodes, depending on the distance to potential destinations. For instance, link-state entries are exchanged more frequently with nearby nodes and less frequently with farther nodes, e.g., GSR [4] and FSR [11].

A different approach for discovering routes in ad-hoc networks consists in taking advantage of location information or Cartesian Routing [8], which assigns each node a unique identifier and geographic location to send the packets through the closest neighbor to the destination node. A similar approach to Cartesian Routing is GLS [19], which gets location information by means of a location service such as GPS in terms of latitude and longitude. This information is then broadcast periodically by each node using control packets so that all nodes can form a table with identities and geographic positions of their neighbors. When a node needs to forward a packet, it places the destination identity in the header of each packet. Each node receiving this packet consults its neighbor table and chooses the closest neighbor towards the destination based on euclidean distances. GPSR [16] is a geographic routing system that uses a planar sub-graph of the wireless network graph to route around holes. Both GLS and GPSR are designed for large metropolitan area networks, but they need high node density and expensive location devices that do not always work well within urban areas.

We end this section by citing Fireworks [17], which is a protocol for managing multicast groups in mobile ad-hoc networks. Fireworks assumes that members of a multicast group are locally grouped in an ad-hoc network, and therefore, it is more efficient (i.e., it generates less signaling) to simply broadcast multicast packets to all members rather than sending independent unicast packets to each member. Multicast group leaders keep a record of the hop count to each group member in order to limit the scope of the broadcast so that it reaches the outermost member node. As we will show later, Fireworks is similar to the last operational phase of NARD, in both protocols a unicast packet is broadcast in a limited area. However, while in Fireworks this process is related to multicast group communications, in NARD this process is related to the dispersion of nodes as time passes.

3 NARD

In contrast to brute-force flooding, where the entire network is flooded with control packets, NARD floods only two arbitrarily small regions in the network. One region is centered at the source node while the other region is located in the vicinity of the destination node. In this way, NARD is capable of reducing significantly the number of control packets used for route-discovery, thus freeing bandwidth for data packet transmissions. NARD operation is composed of three phases that we called *neighbor-discovery phase, neighbor-search phase* and *target-search phase*, which are explained below.

3.1 Neighbor-discovery phase

The neighbor-discovery phase is performed by a node with the purpose of determining the identity of its one-hop neighbors. The identity is specified by a 2-tuple containing the IP and MAC addresses. This information is collected by means of two possible procedures. The first one is to overhear packets from communications taking place in the neighborhood. To this end, a node can set its transceiver in promiscuous mode. It is worth pointing out that the overheard packets can be originated by an end point of a connection or forwarded by relay nodes. Due to this reason, from this procedure, a node is only capable of obtaining the MAC addresses of relay nodes in its neighborhood. However, their corresponding IP addresses can be found by a mechanism similar to the one used in the RARP protocol. The second procedure can be used when a node has not transmitted data packets for a while. In this procedure a node transmits an explicit control packet called *Hello*, so that its current neighbors can be aware of its presence and full identity (i.e., both IP and MAC addresses). The signaling overhead due to Hello packets is negligible because they are small and are not retransmitted by other nodes. Hello packets may not be always necessary. The rate of Hello packets transmitted by each node may be constant and independent of the network size. A node can also control the rate of Hello packets according to its own needs

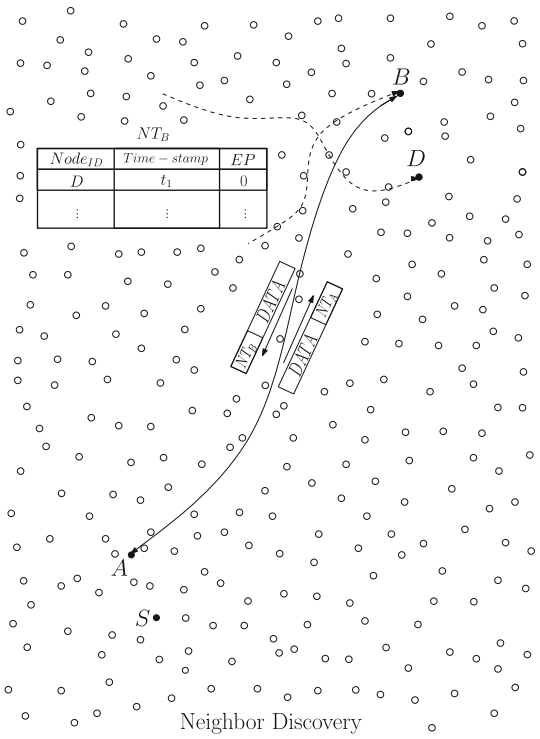

Fig. 1 Neighbor-discovery phase

(e.g., power reserves and network congestion). A node can even stop transmitting Hello packets if it is not expecting any connection with other nodes in the near future. For these reasons, we will not present quantitative overhead measurements of Hello packets in the evaluation section.

In the proposed protocol, nodes collect and store neighbor information in *Neighbor Tables* (*NTs*). These tables are exchanged between the end points of a connection after it is created (see Fig. 1). In this work, the term *connection* refers to any wireless data transfer using either a connection-oriented (TCP) or a connectionless (UDP) communication. Nodes store their own neighbor tables, plus other neighbor tables acquired while communicating with other nodes. Figure 1 illustrates how the neighbor-discovery phase works. This figure depicts the two end points of a connection, i.e., nodes A and B. This connection has been previously discovered using NARD and it is used to transfer data from A to B. These nodes exchange their corresponding neighbor tables (NT_A and NT_B), by attaching them to data packets.

After the end points of a connection have exchanged their neighbor tables, these tables can be updated by reporting their changes only, e.g., entries recently created or modified. Obviously, there is a trade-off between the exchange rate of these updates and the increase on signaling overhead due to this procedure. The main purpose of NARD is to reduce the signaling overhead due to the route-discovery procedure, therefore, the exchange rate of updates must be adjusted according to specific requirements.

Table 1 Neighbor table NT_B

Node ID	*Time stamp*	*EP*
D	t_1	0
⋮	⋮	⋮

As an example, the structure of the neighbor table at node B, NT_B, is illustrated in Table 1. The neighbor table includes the fields *Node ID, Time stamp* and *EP*. The first field contains the identity of a neighbor node. Note that, in our description, this information has been referred for short with a letter (e.g., D), although in reality it consists of the IP and MAC addresses. *Time stamp* is the time when the entry was created or last updated. *EP* (End Point) is a flag that indicates whether the overheard packet was transmitted by an end point of a connection ($EP = 1$) or not ($EP = 0$). A node learns that an overhead packet comes from an end point by comparing the IP and MAC addresses contained in the packet with the information it previously collected in the neighbor-discovery phase. The *EP* flag will be used later in Sect. 4 in order to compute the initial scope of the first flooding. For example, the entry shown in NT_B indicates that node D was a neighbor of node B at time t_1 and D was not an end point of a connection (i.e., $EP = 0$).

When the end points exchange their neighbor tables, they also record some additional information about the route used to transfer them. Such data consider the number of nodes along the route and the time when the table exchange took place. This information will be used to estimate whether or not the route remains valid at a later point in time. This will be described in Sect. 5.2, below.

3.2 Neighbor-search phase

Whenever a node intends to transmit a packet to another node, it checks whether it has a valid route or not. In case a route is available, the packet is relayed to the next hop immediately. When no route is available, the source node creates and transmits a *route-request packet* (*RREQ*) including source and destination IP addresses in it. This initial search is disseminated by its neighbors (*source-neighbors*), which are located within n hops away from the source node. With this *RREQ* the source node searches not only for the destination node, but also for information about recent destination-neighbors. Upon receiving a *RREQ*, a node queries its routing table first. In case there is no route to the destination node, it queries its own neighbor table and other neighbor tables, which were obtained from other nodes during the neighbor-discovery phase. In case a

node has a valid route to either the destination node directly or to past destination-neighbors, it replies via a *route-reply* (*RREP*) unicast packet back to the source node.

Figure 2 shows the operation of neighbor-search phase. In this figure, we can observe that a source node (S) floods an area limited to n hops (shaded area) with a $RREQ_S$ packet and receives routing information from its source-neighbors A, N and P by means of $RREP_A$, $RREP_N$ and $RREP_P$ packets, respectively). This information is related to either destination node, D, or past destination-neighbors. According to Fig. 2, past destination-neighbors are nodes B, M and Q. These nodes are the corresponding end points of valid routes connecting them with nodes A, N and P, respectively.

In case no routing information is collected in this phase (to either the destination node or to destination-neighbors), a search in a larger area becomes necessary. A second search, however, may generate even more signaling overhead and longer delays compared with brute-force flooding techniques. Therefore, it is important to choose the right value of n in order to maximize the probability of finding routing information related to the destination node in the first attempt and with the minimum amount of signaling. The scope of the first search, controlled by parameter n, plays a similar role in NARD as in other limited-search flooding mechanisms such as the one described in FRESH [5]. When n is small, only a small region of the network is flooded with control packets, however there is also a higher probability that no routing information is collected and a second, larger-area search, becomes necessary. An opposite situation occurs when n is large. Choosing a good value of n is therefore a trade-off between the signaling overhead and the probability of finding useful routing information. In Sect. 4 we give some guidelines for choosing the initial value of n.

3.3 Target-search phase

Upon collecting routing information about past destination-neighbors, the source node sends a few unicast packets to these neighbors; we call them *SEARCH* packets (see Fig. 3). We recommend sending more than one *SEARCH* packet because these packets may not always reach the destination-neighbor due to heavy traffic or broken routes. The number of *SEARCH* packets (L) can be estimated as follows. Let us model the successful or failed transmission of a *SEARCH* packet as a Bernoulli trial with success probability of P_s. From the binomial distribution, we can compute the probability that, in L attempts, at least one of them successfully reaches the destination-neighbor. Let us denote this probability by P_T. Given P_s and P_T, we can solve the resulting equation for L; it yields $L = \log(1 - P_T)/\log(1 - P_s)$. For instance, with a probability per attempt of reaching the destination-neighbor of 0.7 and a required probability of reaching it of at least 0.9 we would need approximately two attempts.

Upon receiving a *SEARCH* packet, a destination-neighbor checks whether it has routing information about the destination node; otherwise it constructs a new *RREQ*

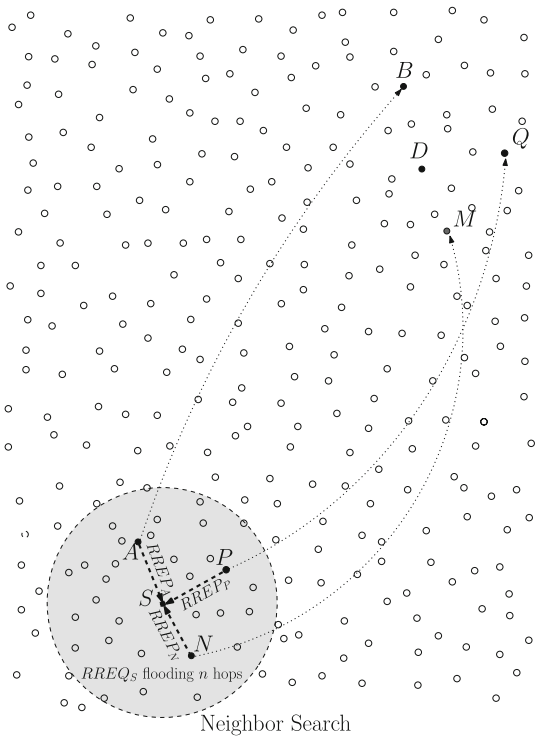

Fig. 2 Neighbor-search phase

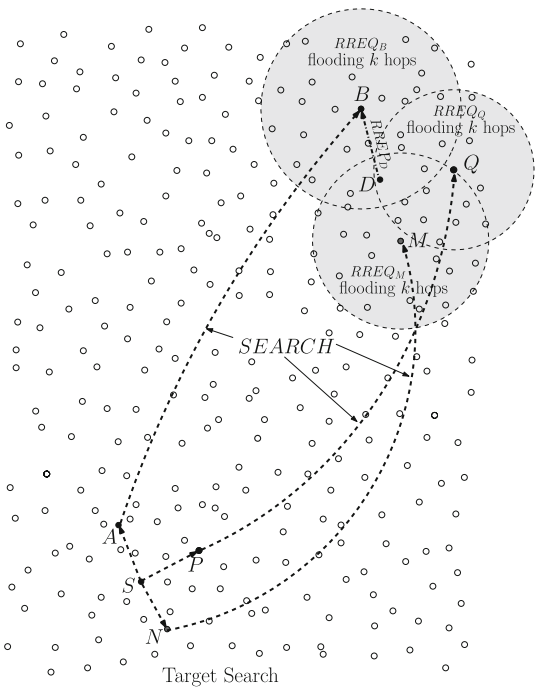

Fig. 3 Target-search phase

packet with the IP address of the destination node as the target node. This new *RREQ* is flooded to an area limited by k hops. Upon receiving a *RREQ*, the destination node replies back with a *RREP* to the neighbor that sent the request which in turn forwards the *RREP* to the source node so data transfer between source and destination can initiate.

The scope of the second flooding (controlled by parameter k) can be determined according to the last encounter time with the destination node as well as node mobility characteristics. A high mobility scenario makes routing information quickly become obsolete as time passes, making it necessary to search in a larger area. A discussion about how to choose parameter k is provided in Sect. 4. Under this scheme, only a small region of the network near the destination node will be flooded by control packets as it is illustrated in Fig. 3. In this figure, source node S has recovered routes from its source-neighbors A, N and P about past destination-neighbors (i.e., nodes B, M and Q). Node S then sends *SEARCH* packets, which are forwarded by nodes A, N and P, intended to reach nodes B, M and Q, respectively. Upon receiving these packets, nodes B, M and Q then perform a new search for node D. Nodes B, M and Q flood an area limited to k hops (shaded areas) with $RREQ_B$, $RREQ_M$ and $RREQ_Q$ packets, respectively. Node D then selects the first one that arrives and replies back to it with a $RREP_D$, in this case, node B. Finally, node B forwards this reply to node S. At this time, a route from S to D is established and data transfer between nodes S and D can begin.

In NARD, the final route between source and destination nodes is, in the worst case, the concatenation of three routes; a first route from the source node to a source-neighbor located inside the first flooding (neighbor-search flooding), a second route from that source-neighbor to a destination-neighbor, and a third route from that destination-neighbor to the destination node. According to Fig. 3, the route from S to D will be formed by a concatenation of three multi-hop routes, i.e., $S \rightarrow A, A \rightarrow B$ and $B \rightarrow D$. We will further discuss this issue in Sect. 4.3.

It is important to note that although parameters n and k appear to have similar roles (i.e., they both limit the scope of the floodings) finding a good value for them obeys quite different guidelines. On one hand, parameter n is related to how much traffic is present in the network. On the other hand, parameter k is related to node dispersion as time passes. This is detailed below.

4 Selecting the scope of the neighbor and target searches

The performance of NARD, as many other routing protocols used in ad-hoc networks, is influenced by node and network dynamics [20, 22]. In NARD, in particular, choosing the scope of the first and second flooding (parameters n and k) plays a key role in the performance and accuracy of NARD.

4.1 The scope of the neighbor search (parameter n)

The scope of the neighbor search is controlled by parameter n. In this search, a source node looks for a source-neighbor which is storing routing information related to either recent destination-neighbors or the destination node itself. Choosing a large value for n increases the area of the search and thus the probability of finding more routing information (more than one source-neighbor with useful information can be found). This may lead toward a successful reach of the destination node, however more signaling overhead is generated. An opposite trade-off applies when the value of n is small. But, it is simple to prove that node density and traffic conditions impact the selection of n. For instance, with high node density, more nodes overhear each transmission on the average, however, neighbor information is useless unless there is data traffic that propagates this local information to other parts of the network (remember that end points exchange their neighbor tables at the beginning of a connection). In practice, with no global view of node density and traffic conditions in the network, a node needs to guess the value of n based on its local view of the network only. For example, a node can approximate the total number of connections in the network by extrapolating the number of end points it overhears within its range. Now we present a method that can be used to choose the initial value of n.

Figure 4 illustrates a network where node S is about to send a $RREQ_S$ limited to n hops looking for routing information related to node D. Following the notation in Fig. 4, let EP_{area} be the number of connection end points (either source or destination) that are located within a particular *area*. As we just mentioned, the source node can determine $EP_{\pi R^2}$ (the number of end points located within a circular area of radius R, i.e., the transmission range) by counting the number of entries in its neighbor table having the *EP* flag set to one. Assuming that end points in the network are homogeneously distributed, the source node can approximate the number of end points located within $\pi (nR)^2$ (the equivalent circular area of a flooding limited to n hops) as

$$EP_{\pi(nR)^2} \approx \frac{\pi(nR)^2}{\pi R^2} EP_{\pi R^2} = n^2 EP_{\pi R^2}. \quad (1)$$

Let us define parameter β as the desired number of end points (source-neighbors), to be found in the first flooding, so that they are associated to a destination-neighbor that met the target node recently. Ideally, the value of β can be

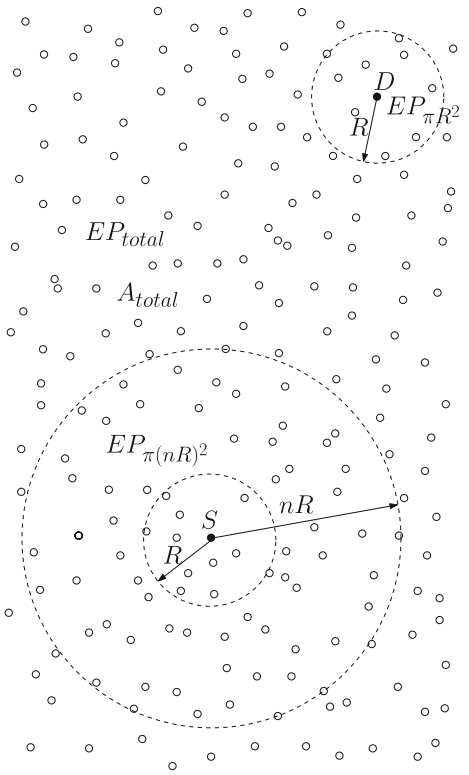

Fig. 4 Parameters involved in the calculation of n include the number of connections, transmission range, and the total area of the network

as small as one, however due to a number of factors (e.g., packet losses and route duration), higher values may be needed. Parameter β satisfies the following equation:

$$\beta \approx EP_{\pi(nR)^2} \frac{\pi R^2}{A_{total} - \pi(nR)^2}, \tag{2}$$

where A_{total} is the total area of the network. The value of A_{total} is unknown to the source node, however, we believe its value can be estimated indirectly. One way to do this, for example, is to monitor the length of the longest detected route in the network, and then assume that this length is the diameter of the network. Most routing protocols are capable of doing this. In DSR, for example, route length can be estimated easily by counting the number of intermediate hops of the routing field located in the header of each data packet.

In Eq. 2 we can set the value of β to the desired value and solve it for n, which controls the size of the first flooding. Therefore, the value of n is found from Eq. 2 and Eq. 1 as

$$n \approx \left\lceil \frac{1}{R} \sqrt{\frac{\beta A_{total}}{\pi(EP_{\pi R^2} + \beta)}} \right\rceil, \tag{3}$$

where $\lceil x \rceil$ is the Ceil function of a real number x. It is possible that this initial choice for n may result in retrieving no routing information or retrieving too much of it. This may be due to estimation errors on either the maximum route length or the number of connections. In these cases, it is necessary to use an adaptive algorithm to adjust its value in further searches according to how successful the initial value of n was. A simple algorithm that can perform iteratively this task is the following:

(1) compute n using Eq. 3 and launch the neighbor-search flooding
(2) increase the value of n in further searches by one if no routing information was retrieved in the previous attempt
(3) repeat (2) until routing information is retrieved
(4) decrease the value of n in further searches if too many nodes replied back to the source node (i.e., $\beta >> 1$). As previously mentioned, ideally we need a single source-neighbor to relay the *SEARCH* packet (i.e., $\beta = 1$). However, we recommend to set the value of β to more than one because *SEARCH* packets may either get lost before reaching its intended destination-neighbor, or because destination-neighbors may not find the destination node during the second search.

4.2 The scope of the target search (parameter k)

Let us consider the problem of determining the scope of the target search so that two nodes that were neighbors time ago can be reached within k hops now. How fast nodes move around and how old was the last time both nodes were neighbors determines the optimum scope of the target search. An old entry in a neighbor table or high mobility conditions (i.e., fast spreading of the nodes) make necessary to find the destination node in a larger area.

Let us assume that all nodes in the network move according to a two-dimensional Random Way-Point (RWP) mobility model [24] as follows. At time zero all nodes are located randomly within the network, then each node selects at random a new position and moves toward it with a rectilinear trajectory at a constant speed. Nodal speed does not need to be the same for all the nodes, but it must be uniformly distributed within $[v_{min}, v_{max}]$, where $v_{min} > 0$ m/s and $v_{max} > v_{min}$. When the node reaches its intended destination a new cycle starts by selecting at random another speed and target position. Additional cycles are performed until the simulation ends. In order to get the size of the flooding during the target search, we need to obtain an expression to compute the relative distance between two nodes as a function of time.

Let nodes j and l be two mobile nodes of the network located at point O at time t_0, see Fig. 5. Mobile nodes move with speeds v_j and v_l along rectilinear trajectories defined

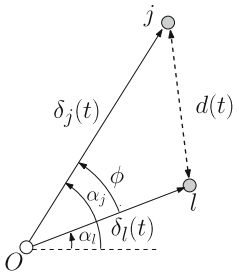

Fig. 5 Relative distance between two mobile nodes

by angles α_j and α_l, respectively. These angles are randomly distributed within [0, 2π]. Additionally, $\delta_j(t) = v_j(t - t_0)$ and $\delta_l(t) = v_l(t - t_0)$ describe the traveled distance by nodes j and l at a given time t, respectively. Here, parameter t_0 represents the *time stamp* in which the neighbor information was recorded in the tables of nodes j and l. In turn, t is the time in which the search process begins. Angle ϕ describes the angular difference between the trajectories followed by nodes j and l, i.e., $\phi = \alpha_j - \alpha_l$. Then, using the Law of Cosines, we can obtain the relative distance ($d(t)$) between two mobile nodes, j and l at a given time t, i.e.,

$$d^2(t) = \delta_j^2(t) + \delta_l^2(t) - 2\delta_j(t)\delta_l(t)\cos(\phi). \quad (4)$$

Now, let us define angle θ as $\theta = \frac{\phi}{2}$, i.e., half of the angular difference between the trajectories followed by nodes j and l. By using the trigonometric identity $\cos(2x) = 1 - 2\sin^2(x)$, making a change of variable and solving for $d(t)$, we obtain

$$d(t) = \sqrt{\left(\delta_j(t) - \delta_l(t)\right)^2 + 4\delta_j(t)\delta_l(t)\sin^2(\theta)}. \quad (5)$$

Angle θ corresponds to a random variable uniformly distributed in the interval $[-\pi, \pi]$. In Fig. 6(a), we show the relative distance between two mobile nodes as a function of θ.

Clearly, the relative distance between nodes j and l is restricted in the interval $d_{min}(t) \le d(t) \le D_{max}(t)$. The minimum relative distance, given by $d_{min}(t) = \left|\delta_j(t) - \delta_l(t)\right|$, happens when both nodes move along the same direction. The maximum relative distance, given by $D_{max}(t) = \left|\delta_j(t) + \delta_l(t)\right|$, happens when both nodes move along opposite directions. With some abuse of notation, let us drop the explicit dependence of time in the previous variables, e.g., $d_{min}(t)$ becomes d_{min}, $D_{max}(t)$ becomes D_{max} and so on. We want to obtain the probability that the relative distance d is less than a given value $\hat{d}$, this is $P(d \le \hat{d})$. In this case the value $\hat{d}$ represents the scope (i.e., radius) of the target search after k hops. To obtain this probability, it is necessary to get either the probability density function (PDF) or the cumulative distribution function (CDF) of d.

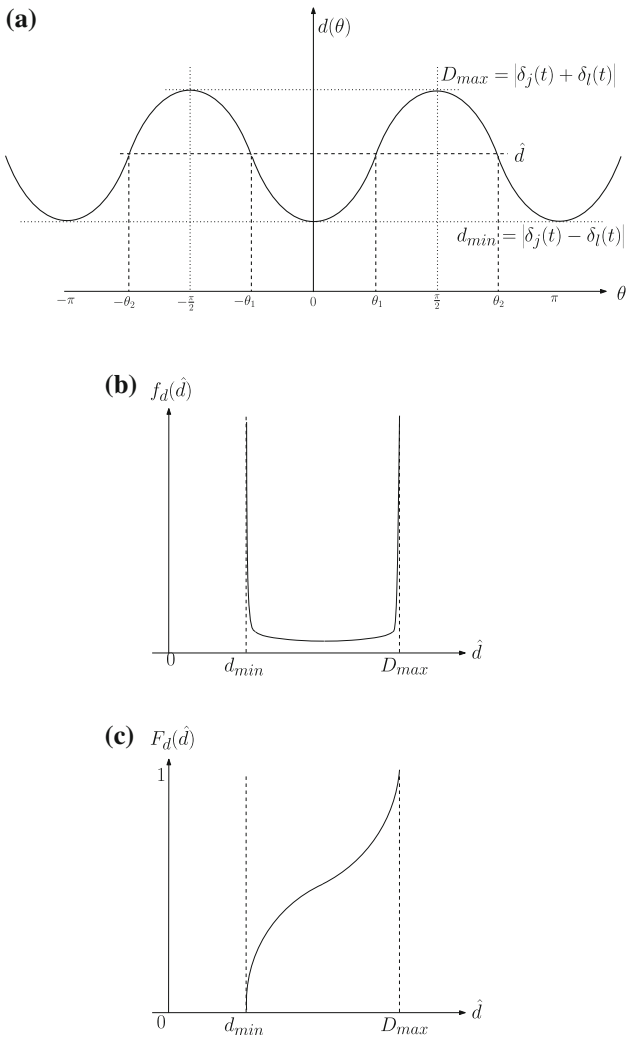

Fig. 6 **a** Relative distance between two mobile nodes. **b** Probability density function (*PDF*) of d. **c** Cumulative distribution function (*CDF*) of d. The minimum and maximum relative distances are $d_{min} = \left|\delta_j(t) - \delta_l(t)\right|$ and $D_{max} = \left|\delta_j(t) + \delta_l(t)\right|$, respectively

It is well known that, under mild conditions, we can obtain the PDF of a function of a random variable $Y = g(X)$ from the PDF of X. Let x_1, x_2, … be the solutions of $y = g(x)$ for a specific y value, $f_Y(y)$ can be found by [27]

$$f_Y(y) = \sum_i \frac{f_X(x_i)}{\left|\frac{\partial y}{\partial x}\right|_{x_i}}. \quad (6)$$

The summation index i in Eq. 6 depends on the total number of solutions at every y. The actual solutions $x_1, x_2, \ldots, x_i, \ldots$ must be found in terms of y.

By following the procedure indicated by Eq. 6, we can obtain the PDF for the relative distance d. Here the random variables Y and X would be d and θ, respectively. If θ is not restricted to a closed interval, the relative distance d would be a continuous and periodic variable with an infinite

number of solutions. But, in this case, the variable θ is restricted to lie in the interval $[-\pi, \pi]$. So, any given distance $\hat{d}$, would only have four solutions, i.e., θ_1, θ_{-1}, θ_2 and θ_{-2}. Also, due to the symmetry, we also know that $\theta_{-1} = -\theta_1$ and $\theta_2 = \pi - \theta_1$ and $\theta_{-2} = -\theta_2$, see Fig. 6(a). Then, the PDF for the relative distance d would be

$$f_d(\hat{d}) = \sum_i \frac{f_\theta(\theta_i)}{\left|\frac{\partial d}{\partial \theta}\right|_{\theta_i}}; \quad i = \pm 1, \pm 2 \tag{7}$$

where θ_i are the four solutions of Eq. 5 for a given distance $\hat{d}$, i.e.,

$$\hat{d} = \sqrt{\left(\delta_j(t) - \delta_l(t)\right)^2 + 4\delta_j(t)\delta_l(t)\sin^2(\theta_i)}; \quad i = \pm 1, \pm 2 \tag{8}$$

But θ is a uniformly distributed random variable, so its PDF would be

$$f_\theta(\theta) = \begin{cases} \frac{1}{2\pi}; & -\pi \le \theta \le \pi \\ 0; & otherwise. \end{cases} \tag{9}$$

Therefore,

$$f_\theta(\theta_i) = \frac{1}{2\pi}; \quad -\pi \le \theta_i \le \pi; \quad i = \pm 1, \pm 2 \tag{10}$$

Now, we obtain $\left|\frac{\partial d}{\partial \theta}\right|_{\theta_i}$ from Eq. 5, as follows

$$\left|\frac{\partial d}{\partial \theta}\right|_{\theta_i} = \frac{4\delta_j(t)\delta_l(t)|\sin(\theta_i)\cos(\theta_i)|}{\sqrt{\left(\delta_j(t) - \delta_l(t)\right)^2 + 4\delta_j(t)\delta_l(t)\sin^2(\theta_i)}}, \tag{11}$$

or

$$\left|\frac{\partial d}{\partial \theta}\right|_{\theta_i} = \frac{4\delta_j(t)\delta_l(t)\left|\sin(\theta_i)\sqrt{1 - \sin^2(\theta_i)}\right|}{\hat{d}}. \tag{12}$$

Solving for $\sin^2(\theta_i)$ from Eq. 8, we obtain

$$\sin^2(\theta_i) = \frac{\hat{d}^2 - \left(\delta_j(t) - \delta_l(t)\right)^2}{4\delta_j(t)\delta_l(t)}; \quad i = \pm 1, \pm 2 \tag{13}$$

Replacing Eq. 13 in Eq. 12 and, after some algebra, it can be simplified as

$$\left|\frac{\partial d}{\partial \theta}\right|_{\theta_i} = \frac{\sqrt{\left(\hat{d}^2 - d_{min}^2\right)\left(D_{max}^2 - \hat{d}^2\right)}}{\hat{d}}. \tag{14}$$

Replacing Eq. 10 and Eq. 14 in the summation given by Eq. 7 we obtain the PDF for the relative distance d, given by:

$$f_d(\hat{d}) = \frac{2}{\pi}\frac{\hat{d}}{\sqrt{\left(\hat{d}^2 - d_{min}^2\right)\left(D_{max}^2 - \hat{d}^2\right)}}; \quad d_{min} \le \hat{d} \le D_{max}. \tag{15}$$

The CDF for the relative distance d, could be found by:

$$F_d(\hat{d}) = \int_{d_{min}}^{\hat{d}} f_d(\tau)d\tau. \tag{16}$$

After solving the integral in Eq. 16, we obtain:

$$F_d(\hat{d}) = \frac{1}{2} - \frac{1}{\pi}tan^{-1}\left[\frac{\left(D_{max}^2 - \hat{d}^2\right) - \left(\hat{d}^2 - d_{min}^2\right)}{2\sqrt{\left(\hat{d}^2 - d_{min}^2\right)\left(D_{max}^2 - \hat{d}^2\right)}}\right]; \quad d_{min} \le \hat{d} \le D_{max}. \tag{17}$$

Figure 6(b), (c) depict the behavior of the PDF and CDF of d, respectively. Table 2 shows some values of the CDF of d, obtained by means of Eq. 17 for different values of $\hat{d}$. From this table we can observe that if we want to find the target with high probability, for all practical purposes the radius of the target search should be the maximum relative distance D_{max}. We emphasize that D_{max} is calculated at the time when the search process begins. As a consequence, the older the *time stamp* associated to an overheard node is, the larger the scope of the required flooding will be. Until now we have considered for simplicity that both nodes j and l where initially located at the same location at time t_0. If we consider now that the maximum separation between both nodes at time t_0 is R meters (the transmission range), then the scope of the flooding during the target search should be:

$$k \approx \left\lceil \frac{D_{max} + R}{R} \right\rceil, \tag{18}$$

or, equivalently:

$$k \approx \left\lceil \frac{D_{max}}{R} + 1 \right\rceil, \tag{19}$$

where k is measured in number of hops. Note that under static conditions $D_{max} = 0$ and the above equation correctly yields the value of $k = 1$.

As mentioned above, the parameter k is mainly related to the dispersion of nodes as time passes. In this calculation we have ignored the probability that nodes j or l change their trajectories from the time they were neighbors to the

Table 2 Flooding radius versus target search success probability

$\hat{d}$	$F_d(\hat{d})$
d_{min}	0.00
$(3d_{min} + D_{max})/4$	0.25
$(d_{min} + D_{max})/2$	0.40
$(d_{min} + 3D_{max})/4$	0.60
$(d_{min} + 7D_{max})/8$	0.72
$(d_{min} + 15D_{max})/16$	0.80
D_{max}	1.00

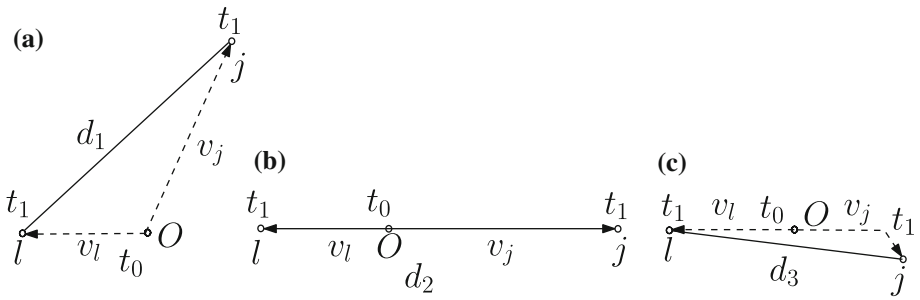

Fig. 7 Examples of dispersion between two mobile nodes (Random Way-Point)

time either node launches a flooding in search of the other. Although changes in direction are considered in the Random Way-Point mobility model, the time-scale of direction changes is longer than the time it takes NARD to discover routes. As we will see in the next section, because we need a valid route in order to deliver a *SEARCH* packet to a destination-neighbor, successful reception of *SEARCH* packets is already a filter that indicates that not many topology changes have occurred. Equation (19) is already a worst case scenario even if trajectory changes do happen. In order to illustrate this, in Fig. 7 we show three different trajectories for node j and the resulting distance to node l when both nodes were initially located at the center of the coordinate system at time t_0. In all these cases, nodes l and j move at constant speeds, i.e., v_l and v_j, respectively, during the interval $[t_0, t_1]$. In the first case, see Fig. 7a, node j does not change its trajectory, thus reaching a relative distance d_1 with respect to node l at time t_1. In the second case, shown in Fig. 7(b), node j moves in the exact opposite direction of node l leading to the maximum distance $d_2 = D_{max} = v_l(t_1 - t_0) + v_j(t_1 - t_0)$. Finally, in the third case (Fig. 7c), node j changes its direction within the interval $[t_0, t_1]$ reaching a distance d_3. Under the assumption that the node speed does not change, condition $d_2 \geq d_3$ holds.

4.3 NARD routes versus optimal routes

As we mentioned before, in NARD the final route between source and destination nodes is, in the worst case, the concatenation of three routes: a first route from the source node to a source-neighbor located inside the first flooding (neighbor-search flooding), a second route from that node to a destination-neighbor node, and a third route from that destination-neighbor to the destination node. This final route, however, may have more hops than necessary compared with a route obtained by a MANET protocol using brute-force flooding or efficient flooding for example. In such cases, there are already route shortening algorithms [15] available that can be used to remove some unnecessary links. There is, however, a simple route optimization that can be applied to NARD in order to reduce extra hops in routes from the very beginning. For this we refer to Fig. 3, where we observe that the source node uses three different routes to reach the destination node. In this case node D can simply reply back to the source node along the route involving fewer hops. This mechanism is already in use in many reactive routing protocols such as DSR [15].

5 Neighbor table considerations

So far in the presentation of NARD we have not discussed the cost of storing or transmitting neighbor tables. Obvious questions related to this aspect include how big neighbor tables are and how many of them need to be stored at each node as time goes on. Below we discuss these two important issues and show that under realistic scenarios neighbor tables have a relatively small size and it is only necessary to keep a subset of neighbor tables acquired from other nodes.

5.1 Size of a neighbor table

The structure of neighbor tables was already presented in Sect. 3, what we discuss now is the size of these tables (i.e., number of entries or overheard nodes). For static networks the maximum number of entries in a neighbor table can be easily found as

$$NT_{size} = \rho \pi R^2 - 1, \tag{20}$$

where ρ is the node density, and πR^2 is the area covered by the transmission range of each node. The density of ad-hoc networks is still a question mark since not many network deployments are available for comparison purposes at this point in time. However, we argue that node density cannot be too high otherwise it will begin to impact the performance of other network components such as the MAC layer.

While moving, a node overhears a different subset of nodes as it travels across the network. Figure 8 illustrates the area covered by the moving coverage area of a mobile node as it roams in the network. Clearly, the more space a node covers the more nodes it will overhear and the larger

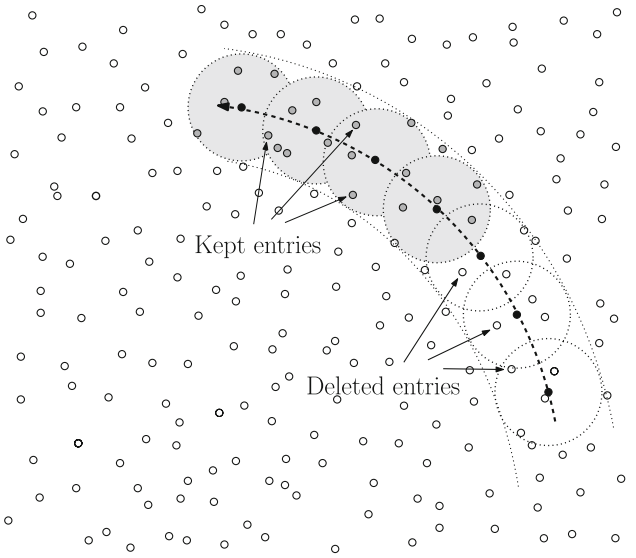

Fig. 8 Relation between mobility and number of entries in neighbor tables

the number of entries in its neighbor table. Fortunately, it is not necessary to keep a record of all of the overheard nodes in neighbor tables as time goes on. In NARD it is recommended that all nodes periodically remove old entries in neighbor tables for which the associated flooding is too large. A simple way to accomplish this goal is to represent a neighbor table as a circular buffer data structure of a fixed length. When a node overhears a new neighbor, it writes down the information in the next empty record of the circular buffer. In case the buffer is already filled, the oldest record in it is overwritten when a new neighbor is detected, but this is exactly what we want to achieve. The size of the circular buffer can be determined manually based on memory capacity or signaling overhead considerations. Mobility plays a role also in determining the size of the buffer. The associated flooding required to find a fast moving node quickly increases as time passes, making the information about that node become less and less useful as time goes on. Since the size of the circular buffer puts a limit on the time interval that an overheard node is kept in this temporary memory, the faster nodes move the smaller the required size of the buffer.

5.2 Number of neighbor tables

In the description of NARD we mentioned that the two end points of a connection always exchange their own neighbor tables after the connection is created. As a result, we should expect that a node may have acquired several neighbor tables as time goes on in case it participated actively as either source or destination. These tables, if not deleted, may consume important storage resources. Fortunately again there is no need to keep all neighbor tables all the time. A neighbor table is associated to the route on which it was transported between the end points of a connection previously established. Once this route is no longer valid, the corresponding table may be discarded.

There are various route-duration models (e.g., [1, 28]) that, given the mobility patterns of moving nodes, speed and number of intermediate hops, can estimate the duration of routes in mobile ad-hoc networks. Most of these models consider that the route duration (R_d), measured from the time a route is created or last updated to the time it is no longer valid, decreases as the speed and number of hops in the route increase according to an expression of the form [1]

$$R_d \approx \frac{R}{\lambda_0 v N_h}, \tag{21}$$

where R_d is the route duration, R is the transmission range, λ_0 is a constant of proportionality ($0.5 < \lambda_0 < 1.2$), v is the speed of movement and N_h is the number of intermediate hops in the route. A node receiving a neighbor table from another node keeps a record of the route along its expected duration (using Eq. 21). In NARD, once the expected route duration expires (i.e., because a member of the route probably moves away from the route), there is no reason to keep its associated neighbor table.

6 Performance evaluation

NARD was implemented using the NS2.28 network simulator [7]. In this implementation of NARD we used the DSR routing protocol as a starting point. We replaced the route-discovery mechanism of DSR (brute-force flooding) with NARD instead. As we mentioned before, NARD is not tied to any routing protocol in particular, and it can be used with any reactive protocol replacing its route-discovery mechanism. In order to set the number of hops that a route-request (*RREQ*) is propagated in the network, we modified the time-to-live value of the *RREQ* (16 hops is the default in DSR), so that we can set its value to either n hops or k hops.

In what follows, we present an experimental performance evaluation of NARD under different network conditions to point out its advantages and disadvantages. In all cases, nodes use the standard IEEE 802.11 MAC protocol running at 2 Mbps. We ran simulations with two types of scenarios. The first one under static conditions (i.e., no mobility model). The second one with mobility according to the Random Way-Point mobility model [24] and mean speeds of 2 and 10 m/s (we selected the speeds at random by sampling the intervals [1.8,2.2] and [9.8,10.2] m/s). Table 3 shows the parameters used in the experiments.

Table 3 Simulation parameters for both NARD and DSR

Parameter	Value
Data rate	1 pkt/s
Packet size	100 bytes
Mean node speed	0, 2 and 10 m/s
Mobility model	Random Way-Point (for 2 and 10 m/s)
Scenario size	2,200 × 1,200 (m^2)
Number of nodes	350
Number of source-neighbors (β)	3
Number of *SEARCH* packets (L)	1
Number of connections	0, 10, 30, 60, 80, 100
Transmission range	250 m
Propagation model	Two-ray ground
Scenario generator	Setdest
Pause time	0 s

For each simulation, Z random connections are created before a route-discovery search is performed with NARD for the $(Z + 1)$st connection (scenarios for Z equal to 0, 10, 30, 60, 80 and 100 previous connections were considered). The results shown in Figs. 9 and 10 are related to the $(Z + 1)$st connection only, and are compared to those obtained by DSR under equivalent conditions. In these figures we did not consider any efficient-flooding technique since, as we mentioned before, they can be used in parallel with NARD, resulting in an even lower signaling overhead. All connections are of type UDP/CBR transmitting a 100-byte long packet per second. In our simulations we considered the most conservative policy regarding how often the neighbor tables are exchanged. They were exchanged after the connection was created. We fixed $n = 1$ and $k = 3$ in the plots of Figs. 9 and 10 in order to compare the performance of NARD with respect to the number of connections and mobility only. We did not make use of the optimum values of n and k in these experiments, otherwise we would be dealing with four different parameters at once. This would not let us observe clearly how each parameter affects NARD in which way (we used optimum values of k and n for the experiments whose results are shown in Figs. 12 and 13, where we compare NARD with FRESH).

Figure 9(a)–(c) show the performance of NARD for the $(Z + 1)$st connection only for 0, 2 and 10 m/s, respectively. This includes:

(1) *Routes to neighbors*. These data are the number of routes, from source-neighbors to destination-neighbors, that were found by the source node during the neighbor-search phase.

(2) *Reached neighbors*. This is the number of destination-neighbors reached by the source node with *SEARCH* packets.

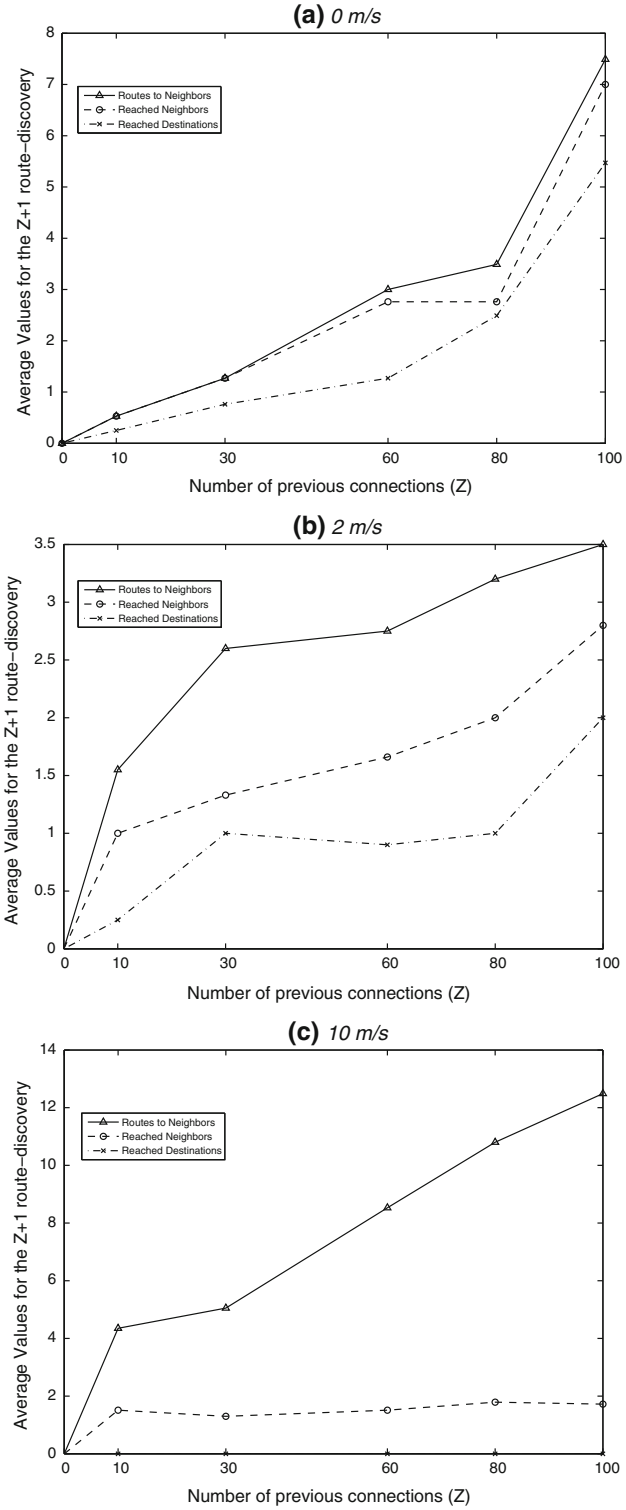

Fig. 9 NARD performance for the $(Z + 1)$st route-discovery at 0 m/s (**a**), 2 m/s (**b**) and 10 m/s (**c**)

(3) *Reached destinations*. This is the number of destination-neighbors reached by *SEARCH* packets that could find the destination node during the second flooding.

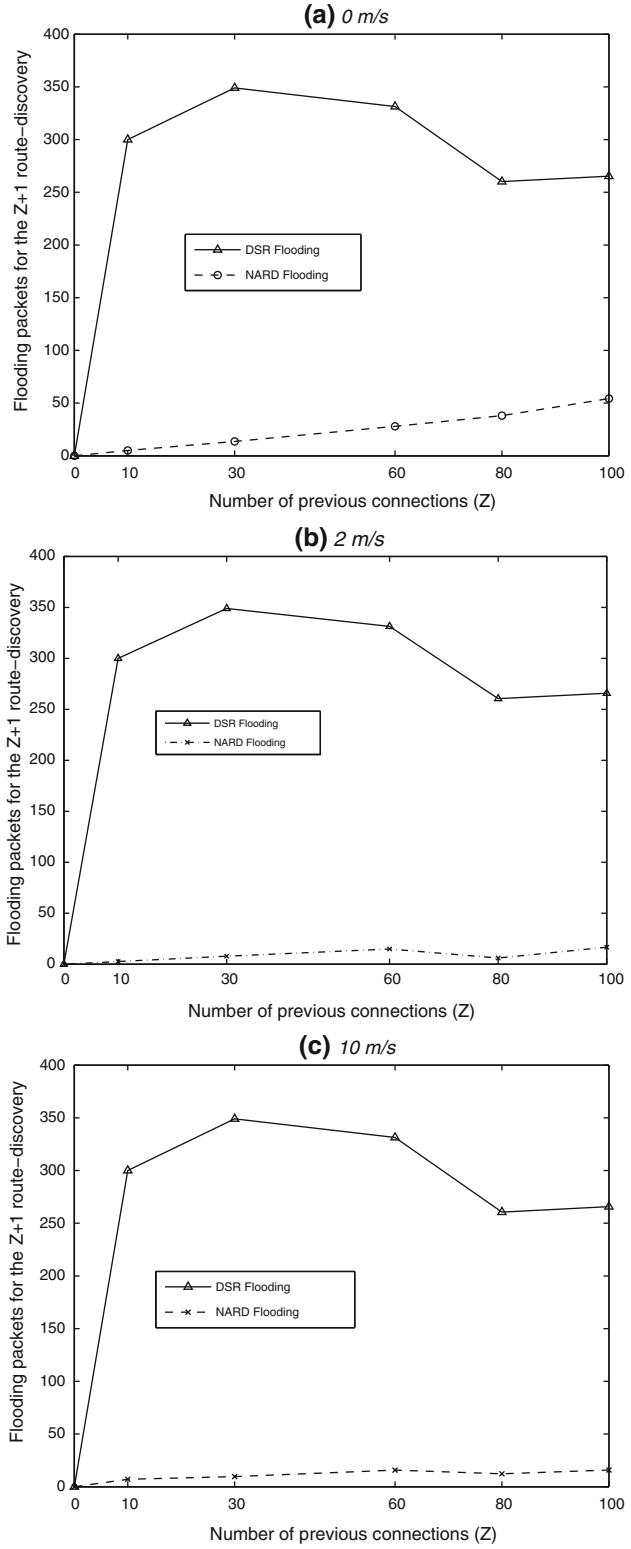

Fig. 10 Flooding performance **a** 0 m/s, **b** 2 m/s and **c** 10 m/s

Figure 9a shows the performance of NARD with no mobility (0 m/s). As expected, we observe that the number of routes to destination-neighbors found in the neighbor-search phase increases as the number of connections in the network rises (remember that for this experiment we fixed the value of n to 1). This is a direct result of the neighbor-discovery phase where end points exchange their corresponding neighbor tables. Probably the main drawback of NARD is that it requires some background traffic to work properly. The good news is that even for a few connections (e.g., 10 connections in Fig. 9a) some information about route to destination-neighbors was retrieved during the neighbor-search phase even for $n = 1$. In this figure we also observe that most *SEARCH* packets sent by the source node reached their intended destination-neighbor except for high traffic conditions. Packets that did not reach the intended destination-neighbor were dropped by the network because of congestion. Once a destination-neighbor received the *SEARCH* packet, it always found the destination node during the second search except again when heavy traffic congestion was present.

Figure 9(b), (c) show the performance of NARD when nodes move at 2 m/s and 10 m/s, respectively. Mobility brings benefits and disadvantages to the performance of NARD. When nodes move, they overhear more and more nodes as time passes, thus increasing their overall knowledge about other nodes in the network. This is in contrast with the static case (0 m/s) where a source node cannot retrieve information from nodes outside its transmission range ($n = 1$). This phenomenon can be observed in Fig. 9(b) where the number of source-neighbors found in the first search rises faster with respect to the number of connections compared with Fig. 9(a). On the other hand, mobility makes routing information less reliable because nodes move around as times passes. This can be observed in Fig. 9(b) where some *SEARCH* packets did not reach their intended destination-neighbor due to broken routes. An extreme example of the impact of mobility can be seen in Fig. 9(c) where only a few *SEARCH* packets reached their intended destination-neighbors, and even if they did, none of them found the destination node during the second search. This poor performance can be corrected once the optimum value of k is chosen according to Eq. 19. Remember that we use a fixed value of k ($k = 3$) in Fig. 9(c) to focus on the impact of background traffic on NARD's performance only. This figure shows that the destination node was already located outside the scope of a flooding limited to 3 hops from the destination-neighbor sending the *RREQ*.

Figure 10(a)–(c) show the number of signaling packets generated by NARD for the $(Z + 1)$st route-discovery only (this includes the sum of the signaling packets generated during neighbor-search and target-search phases). For comparison purposes we also plot the signaling overhead of DSR in the same experiment as a reference. As we can observe in these figures, NARD outperforms DSR by a

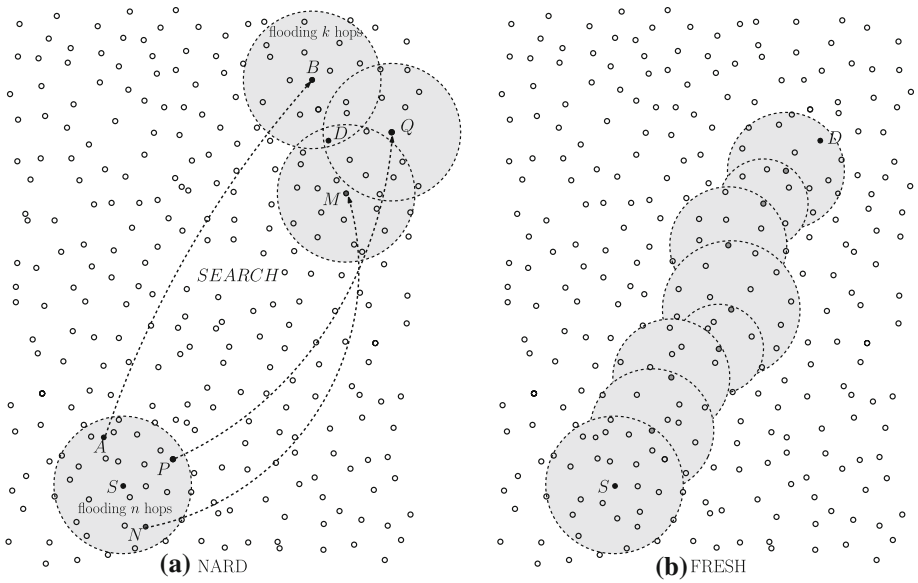

Fig. 11 Flooding footprint of **a** NARD and **b** FRESH

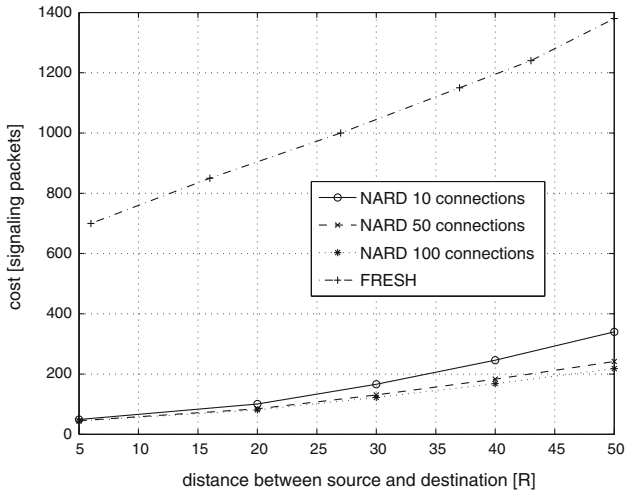

Fig. 12 FRESH versus NARD (constant k)

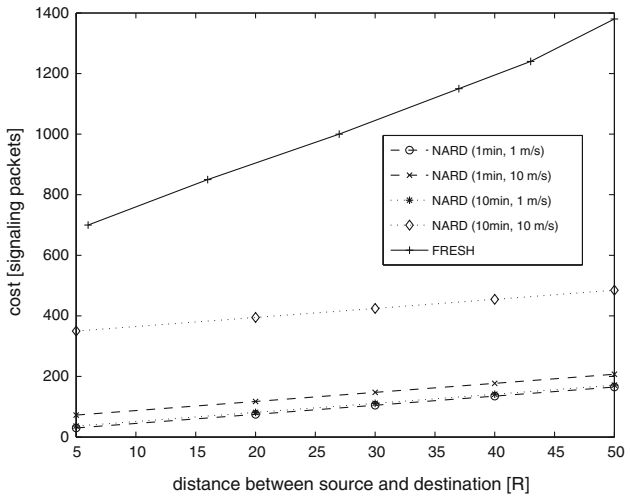

Fig. 13 FRESH versus NARD (constant n)

great factor. This again is due to the fact that NARD floods only two small regions of the network as opposed to DSR where the entire network is flooded with control packets (brute-force flooding).

6.1 NARD versus FRESH

FRESH is a reactive protocol that, similar to NARD, targets the reduction of the area search during route-discovery. We now compare the performance of FRESH with NARD.

In FRESH, nodes keep a record of their most recent encounter times with all nodes. Instead of searching for the destination node directly, the source node searches in a limited portion of the network for any node that has encountered the destination node more recently than did itself. This intermediate node (i.e., anchor) performs a second limited-area search for another node that has encountered the destination even more recently and so on. This procedure continues until the destination is finally reached. Figure 11 roughly describes the footprint left by both protocols [(a) NARD and (b) FRESH] during a search.

In FRESH, the search cost of a single search (i.e., the area of the network covered during the search) which originates at node i and terminates at node $i + 1$ is found from [5] as $C_{FRESH}(i,i+1) = (\xi(|X_i - X_{i+1}|))^2$ for some $1 < \xi < 2$ and X_i is the position of the $i - th$ anchor. The cost is quadratic with the distance because the number of packet transmissions generated by the search is proportional to the number of nodes located in a circular area of radius $|X_i - X_{i+1}|$ where the flooding takes place. Here ξ models the fact that the radius of the search will, on average, be larger than the distance between the two nodes. In the curves showing the performance of FRESH in

Figs. 12 and 13 the authors in [5] used $\xi = 1.3$. In FRESH a route-discovery involves N consecutive searches, as sketched in Fig. 11(b), having the following total search cost in terms of packet transmissions [5]:

$$C_{FRESH} = \rho \sum_{i=1}^{N} (\xi(|X_i - X_{i+1}|))^2, \quad (22)$$

where ρ is the node density. We can find an equivalent cost function for NARD as:

$$C_{NARD} = \rho(nR)^2 + L\beta \frac{|X_S - X_D|}{R} + \rho\beta(kR)^2. \quad (23)$$

The terms in Eq. (23) account for the number of signaling packets generated during neighbor search (first term), transmission of *L SEARCH* unicast packets to each of the β source-neighbors (second term), and packets generated during target search (third term), respectively.

Figures 12 and 13 compare the signaling overhead generated by FRESH and NARD during route-discovery under similar network settings (we used $\beta = 3$ for NARD). We took the values of FRESH shown in both figures directly from [5] and no proactive signaling overhead from Hello packets was included for either protocol. In both figures we use $\rho = 1$ (unit density) and $R = 1$ (unit radius) as in [5]. In Fig. 12 we kept constant the scope of the second flooding ($k = 3$) and focused on the behavior of the first flooding with respect to the background traffic only using Eq. 3. As we can see in this figure the amount of background traffic has only a limited impact on search cost for NARD. In contrast to Fig. 12, in Fig. 13 we kept constant the scope of the first flooding ($n = 3$) and focused on the behavior of the second flooding with respect to how old was the last encounter and the speed of movement using Eq. 19. As encounter times become older and speed of movement increases, NARD incurs in higher costs to reach the destination node.

As we can observe in Figs. 12 and 13, NARD generates less signaling packets than FRESH. This advantage increases as the source-destination distance increases. The fact that NARD uses unicast packets to cover a significant portion of the search (see Fig. 11a) is the key reason of its improved performance. FRESH uses node mobility to actually *move* location information across the network, NARD on the other hand, uses background traffic to perform a similar task. Since usually traffic forwarding moves information faster than moving nodes, NARD will have fresher information about the location of the destination node, which translates into less flooding areas and less signaling overhead.

We believe NARD and FRESH do not need to compete but in fact they can complement each other. Because NARD requires background traffic to work properly, we imagine a framework where a routing protocol can use FRESH in cases where no background traffic is present, and then switch to NARD once some degree of background traffic is detected.

7 Conclusions

In this work a novel route-discovery protocol for ad-hoc networks called NARD is presented. In NARD, a source node performs a limited-area search looking not only for a destination node, but also for past destination-neighbors. Because NARD floods only two small regions of the network, one around the source and another in the vicinity of the destination, it achieves a lower overhead compared with brute-force flooding. We implemented NARD in NS2.28 where several scenarios were analyzed with different levels of background traffic and node mobility. From the results we observed that NARD generates less signaling overhead compared with brute-force flooding in most scenarios. A downside feature of NARD is that it requires some background traffic to work properly. Fortunately, the simulation results show that even with a low level of traffic NARD shows good performance. A comparison with FRESH showed that NARD generates less signaling packets and this advantage becomes more important as the source-destination distance increases. We believe NARD and FRESH can complement each other depending on the amount of background traffic in the network.

Acknowledgments This work was supported in part by research funds from CONACyT grants 105117 and 105279, by DGAPA - PAPIIT grants IN108910 and IN106609, Texas A&M University-CONACyT grant 2010-049, PAPIME PE 103807 and PROMEP 12711445.

References

1. Bai, F., Sadagopan, N., Krishnamachari, B., Helmy, A. (2004). Modelling path duration distributions in MANETs and their impact on reactive routing protocols. *IEEE Journal on Selected Areas in Communications, 22*(7), 1357–1373.
2. Chakeres, I.,& Perkins, C. (2010). *Dynamic MANET on-demand (DYMO) routing*. IETF Internet Draft, draft-ietf-manet-dymo-18, February, 2010 (in progress).
3. Chen, B., Jamieson, K., Balakrishnan, H., & Morris, R. (2001). Span: An energy-efficient coordination algorithm for topology maintenance in ad-hoc wireless networks. In *Proceedings of Mobicom*.
4. Chen, T. W., & Gerla, M. (1998). Global state routing: A new routing scheme for ad-hoc wireless networks. In *Proceedings of the IEEE international conference on communications*, ICC98, Atlanta, GA, USA.
5. Dubois-Ferrie, H., Grossglauser, M., & Vetterli, M. (2003). Efficient route discovery in mobile ad-hoc networks using encounter ages. In *Proceedings of MobiHOC*, Annapolis, MD, USA.
6. Sakhaee, E., Taleb, T., Jamalipour, A., Kato, N., & Nemoto, Y. (2007). A novel scheme to reduce control overhead and increase

link duration in highly mobile ad-hoc networks. In *Wireless Communications and Networking Conference*.
7. Fall, K., & Varadhan, K. (2003). *The NS manual*. The VINT project, July.
8. Gregory, G. F. (1987). *Routing and addressing problems in large metropolitan-scale internetworks*. USC ISI Report ISI/RR-87-180, March.
9. Garcia-Luna-Aceves, J. J., Spohn, M. (1998). Scalable link-state internet routing. In *Proceedings of IEEE international conference on network protocols*, ICNP.
10. Gomez, J., & Campbell, A. T. (2007). Using variable-range transmission power control in wireless ad-hoc networks. *IEEE Transactions on Mobile Computing*, ISSN 1536-1233, 6, January.
11. Guangyu, M. P., Gerla, M., & Chen, T. W. (2000). Fish-eye state routing: A routing scheme for ad-hoc wireless networks. In *Proceedings of the IEEE international conference on communications*, ICC 2000, New Orleans, LA, USA, pp. 70–74, June, 18–22.
12. Haas, Z. J., & Pearlman, M. R. (1999). Determining the optimal configuration for the zone routing protocol. In *IEEE Journal on Selected Areas in Communications*, pp. 1395–1414.
13. Haerri, J., Filali, F., & Bonnet, C. (2006). Performance comparison of AODV and OLSR in VANETs urban environments under realistic mobility patterns. In *Proceedings of the 5th IFIP mediterranean ad-hoc networking workshop*, Lipari, Italy, June 14–17.
14. Jan, S., Shah, I. A., & Al-Raweshidy, H. (2009). Performance analysis of proactive and reactive routing protocols for mobile ad-hoc grid in e-health applications. In *International conference on communication software and networks*, Los Alamitos, CA, USA, pp. 484–488.
15. Jhonson, D. B., & Maltz, D. A. (2007). *Dynamic source routing protocol for mobile ad-hoc networks*. IETF Request for Comments RFC 4728, February (in progress).
16. Karp, B., & Kung, H. T. (2000) GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceeding of ACM/IEEE MobiComm*, New York, NY, USA.
17. Lap, K. L., Krishnamurthy, S. V., & Falautsos, M. (2005). Fireworks: An adaptive group communication protocol for mobile ad-hoc networks. Networking, Waterloo, Canada.
18. Lee, H. K., Kim, Y. W., & Song, J. S. (2007). AOZDV: An enhanced AODV protocol based on zone routing in MANET. In *Proceedings of the international conference on wireless communications, networking and mobile computing*, WiCom 2007, Shangai, China, pp. 1685–1688, September, 21–25.
19. Li J., & John J. (2000). A scalable location service for geographic ad-hoc routing. In *Proceeding of MOBICOM*, Boston MA, USA.
20. Li, X., Agrawal, D. P., & Zeng, Q.-A. (2004). Impact of mobility on the performance of mobile ad-hoc networks. In *Proceedings of the 3rd IEEE annual wireless telecommunications symposium*, Pomona, CA, USA.
21. Lim, H., & Kim, C. (2000). Flooding in wireless ad-hoc networks. *IEEE Computer Communications*.
22. Lin, G., Noubira, G., & Rajaraman, R. (2004). Mobility models for ad-hoc network simulation. In *Proceedings of INFOCOM*, Hong Kong, China.
23. Mbarushimana, C., & Shahrabi, A. (2007). Comparative study of reactive and proactive routing protocols performance in mobile ad-hoc networks. In *Proceedings of the 21st Iiternational conference on advanced information networking and applications workshops*, AINAW '07, Washington, DC, USA, pp. 679–684.
24. Navidi, W., & Camp, T. (2004). Stationary distributions for the random waypoint mobility model. *IEEE Transactions on Mobile Computing, 3*(1).
25. Ni, S.-Y., Tseng, Y.-C., Chen, Y.-S., & Sheu, J.-P. (1999). The broadcast storm problem in a mobile ad-hoc network. In *Proceedings of IEEE/ACM Mobicom*, Seattle, Washington, USA.
26. Ogier, R. G., Lewis, M. G., Templin, F. L., & Bellur, B. (2002). *Topology broadcast based on reverse-path forwarding (TBRPF) flooding in wireless ad-hoc networks*. IETF Internet Draft (in progress).
27. Papoulis, A., & Unnikrishna Pillai, S. (2002). *Probability, random variables and stochastic processes* (4 Rev., 3rd edn.). New York: McGraw Hill Higher Education.
28. Pascoe, M., Gomez, J., Rangel, V., & Lopez-Guerrero, M. (2010). Route duration modeling for mobile ad-hoc networks. *ACM Wireless Networks Journal (WiNet), 16*(3), 743–757.
29. Perkins, C., Belding-Royer, E., & Chakeres, I. (2003). *Ad-hoc on-demand distance vector (AODV) routing*. IETF Internet draft, draft-perkins-manet-aodvbis-00.txt, October 2003 (in progress).
30. Qayyum, A., Viennot, L., & Laouiti, A. (2000). *Multipoint relaying: An efficient technique for flooding in mobile wireless networks*. INRIA Technical Report.
31. Tseng, Yu-Chee, Ni, Sze-Yao, & Shih, En-Yu (2001). Adaptive aproaches to relieving broadcast storms in a wireless multihop ad-hoc network. In *Proceedings of IEEE INFOCOM*, Anchorage, Alaska, USA.
32. Yen, Y.-S., Chang, H.-C., & Chang, R.-S. (2006). Routing with adaptive path and limited flooding for mobile ad-hoc networks. In *1st International symposium on wireless pervasive computing*.

Author Biographies

J. Gomez received the B.S. degree with honors in Electrical Engineering in 1993 from the National Autonomous University of Mexico (UNAM) and the M.S. and Ph.D. degrees in Electrical Engineering in 1996 and 2002, respectively, from Columbia University and its COMET Group. During his Ph.D. studies at Columbia University, he collaborated and worked on several occasions at the IBM T.J. Watson Research Center, Hawthorne, New York. His research interests cover routing, QoS, and MAC design for wireless ad hoc, sensor, and mesh networks. Since 2002, he has been an Associate Professor with the National Autonomous University of Mexico. Javier Gomez is member of the SNI (level I) since 2004.

V. Rangel obtained his Bachelor Degree in Computer Engineering from the National Autonomous University of Mexico (UNAM). He obtained his Master Degree in Data Communication Systems and his doctoral degree in Telecommunications Engineering from the Centre for Mobile Communications Research, The University of Sheffield (England). His Ph.D. thesis focused on the modeling and analysis of Cable TV networks supporting Cable TV networks supporting broadband Internet traffic. Dr. Rangel is currently a professor at the Department of Telecommunications Engineering, School of Engineering (UNAM).

M. Lopez-Guerrero received his B.Sc. with honors in Mechanical–Electrical Engineering in 1995 and the M.Sc.with honors in Electrical Engineering in 1998, both from the National Autonomous University of Mexico. He received his Ph.D. in Electrical Engineering from the University of Ottawa in 2004. Currently, he is an Associate Professor with the Metropolitan Autonomous University (Mexico City). His areas of interest are in medium access control, traffic control, and traffic modeling.

M. Pascoe-Chalke received his B.Sc. in Mechanical–Electrical Engineering in 1997 and the M.Sc. and the Ph.D. with honors in Electrical Engineering in 2005 and 2010, respectively, all from the National Autonomous University of Mexico (UNAM). His areas of interest include routing protocols, location systems and modeling of node mobility in wireless ad-hoc networks. Currently, he is a visiting professor with the Metropolitan Autonomous University (Mexico City).