Available online at www.sciencedirect.com
SCIENCE DIRECT®
Ad Hoc Networks 4 (2006) 547–566
Ad Hoc Networks
www.elsevier.com/locate/adhoc

A hint-based probabilistic protocol for unicast communications in MANETs ☆

R. Beraldi *, L. Querzoni, R. Baldoni

Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Via Salaria 113, Roma 00198, Italy

Received 24 December 2004; received in revised form 9 May 2005; accepted 3 June 2005
Available online 13 July 2005

Abstract

Point-to-point transmissions represent a fundamental primitive in any communication network. Despite many proposals have appeared in the literature, providing an efficient implementation of such an abstraction in Mobile Ad Hoc Networks (MANETs) still remains an open issue.

This paper proposes a probabilistic protocol for unicast packet delivery in a MANET. Unlike the classical routing protocols, in our proposal packet forwarding is not driven by a previously computed path. Rather, the nodes of the network exploit a set of routing meta-information (called *hints*) to discover a path to the destination on-the-fly. This assure robustness against topological changes, while requiring a very low overhead.

A node gathers hints from the nodes located within a small number of hops (called the protocol's lookahead) from itself. As showed through simulations, very good performance can be obtained with small lookahead. The main statistical properties of hints have been investigated through an analytical model, which is also reported in the paper.

Keywords: MANET; Routing; Probabilistic protocols

1. Introduction

Mobile Ad hoc networks (MANETs) are emerging as an important class of ad hoc networks with many applications in the real life [7]. In a MANET two nodes can communicate directly only when they lie within one another's transmission range. This is quite an unlikely condition since the network's diameter is usually much bigger than such a range, hence unicast packet transmission is

☆ The work described in this paper was supported by the Italian Ministry of Education, University, and Research (MIUR) under the "IS-MANET" project. A preliminary version of this work appeared in the 11th International Conference on Parallel and Distributed Systems, ICPADS 2005, Fukuoka, Japan.

* Corresponding author. Tel.: +39 06 4991 8502.
E-mail addresses: beraldi@dis.uniroma1.it (R. Beraldi), querzoni@dis.uniroma1.it (L. Querzoni), baldoni@dis.uniroma1.it (R. Baldoni).

1570-8705/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2005.06.004

multi-hop in nature. Multi-hop communications are challenging in time-varying topologies. Despite many proposals appeared in the literature, finding an efficient solution to such a problem still remains an open issue.

The usual approach to implement unicast can be classified as deterministic and structure-based. The goal of a protocol of this kind, namely a routing protocol, is to maintain some logical structure on top of the physical network, which nodes exploit for taking their packet forwarding decisions. For example, in the class of proactive routing protocols such a structure is a set of N trees, each rooted at one of the N nodes forming the MANET, which are stored in a distributed fashion into the nodes' routing table. The next-hop node is obtained by a simple table's lookup and thus packet latency is minimized. However, maintaining the routing information can become not efficient under frequent topological changes. Representative proactive protocols are: Destination-Sequenced Distance Vector (DSDV) [5], Optimization Link State Routing (OLSR) [12] and Topology Based on Reverse Path Forwarding (TBRPF) [3].

In reactive protocols a logical path from a source node to the destination is discovered on-demand and maintained afterwards. The main drawback of such a solution is that the delay required to discover or repair a path can be perceived at the application level; moreover an interference with the congestion control mechanism can arise (for example, a route break detection normally takes several seconds [17]). Examples of protocols of this family are: Dynamic Source Route (DSR) [13] and Ad hoc On Demand Distance Vector (AODV) [6].

A vast number of routing protocols are proposed in the literature along these lines, with several variants, e.g., hybrid protocols [10]. The interested reader can refer to [1] for a survey on routing protocols.

Probabilistic algorithms provide an interesting alternative to realize unicast packet delivery in dynamic networks, due to their resilience to topological changes, as well as the low-overhead, scalability and locality properties they enjoy. In the context of MANET, probabilistic protocols have already been applied to improve the route discovery process [9], as well as an alternative to implement network wide broadcast [15]. However, at the best of our knowledge, only few papers have explored the more important option of embedding the probabilistic logic in the forwarding process itself. The solutions appeared in the literature aim at defining a biased random walk. For example, in [14] the behavior of a community of ants is mimed. Tuning the parameters of the protocol is, however, not trivial in this technique. Moreover, before data packet can be sent, a path still needs to be discovered through natural random walkers.

In this paper we introduce a framework for probabilistic forwarding in mobile environments, which exploits meta-information (in the form of hints) to direct a packet towards the general direction of the destination. Some preliminary idea of the protocol can be found in [2]. A hint h_{id} computed by a node i w.r.t. a destination node d, is a positive value which indicates the chance of i being in the neighborhood of d. The lower the hint the higher such a probability, with the singular case of $h_{id} = 0$ when i and d are 1-hop neighbors i.e., they lie within one another's transmission range. Each node i periodically receives the hints for d from the nodes located at distance at most of L hops from itself (the value L is called the *Look-ahead* of the protocol).

Packets are forwarded as follows. On receiving a packet destined to d, a node i forwards it to the neighbor from which the best (lowest) hint has been received, and such it did not see the packet before. If no nodes can be selected the packet is discarded. This procedure is repeated at each forwarding step. This algorithm is clearly resilient to topological changes. If the selected next-hop node cannot be reached, another one can promptly be used. Moreover, we argue that a small lookahead can be sufficient for a node to gather correct hints; thus, a small amount of control overhead is required. There is no guarantee that the next-hop node lies on a path towards the destination and this qualifies the protocol as a probabilistic one.

The hint of i w.r.t. d is defined as $h_{id} = \frac{\Delta T_{id}}{\tau_{id}}$, where ΔT_{id} is the time elapsed since d has most recently moved out of the i's transmission range, and τ_{id} is the duration of the last wireless link established between i and d, hereafter also called the

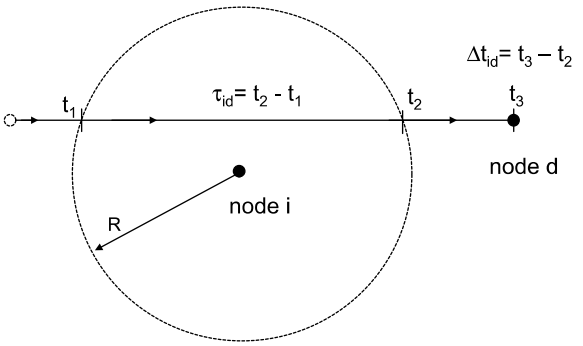


Fig. 1. Meaning of time information.

contact time between i and d (see Fig. 1). The contact time provides a way to gain a first order estimation of the relative speed between i and d, in the following sense: if we assume that the trajectory of d with respect to i does not change during the contact of i with d and that the probability that i comes in contact with d starting from a given point is independent from the speed, then the higher the relative speed the lower the expected value of τ_{id}; hence, the expected value of the hint increases with the speed. Note that the chances of an invariant trajectory increase when the relative speed is high.

As a consequence, we can assume the expected value of a hint to be proportional to the Euclidian distance between i and d. Some more insight into such a relationship has been learned through an analytical model and empirically exploiting the random waypoint mobility model [13]. Clearly, for sufficient dense networks a hint also provides a way to estimate the distance in number of hops between i and d.

The performance of the proposed protocol has been studied through simulations and compared against AODV. The results show that a high delivery probability associated to a low average delay can be obtained at the cost of a modest overhead. Moreover, a remarkably reduction in the variance of the delivery delay w.r.t. AODV was also observed in the simulation results.

The paper is organized as follows: the next section summarizes the main works related to our approach; Section 3 provides model and assumptions; Section 4 describes the hint based protocol; in Section 5 an analytical model for studying the hint–distance correlation is presented. Simulation results are provided in Section 6 and a conclusion in Section 7.

2. Related work

FRESH, see [8], is a path search algorithm which exploits the time elapsed since the last encounter of nodes with the destination in order to reduce the cost of a path discovery. By definition an encounter between two nodes happens when these nodes are neighbors. To discover a path, a node searches for any intermediate node that encountered the destination more recently than itself. The intermediate node then searches for another node that encountered the destination yet more recently and so on, until the destination is reached. In this way, a flood-based search is steered in the general direction of the destination. In our proposal the encounter age is used as the numerator for computing a hint. The fundamental difference with *FRESH* is that our goal is to eliminate the need of any network-wide operation. On the contrary, *FRESH* requires a reactive routing protocol to deliver a packet.

The estimation of the meeting likelihood between nodes is the key ingredient adopted in [11]. In essence, a node forwards a packet when a neighboring node provides a higher meeting likelihood than itself. The similarity with our scheme is that in both cases a packet is routed towards the destination exploiting the chances that a node is better than the one currently holding the packet: in the former better means closer, while in the latter it means meeting the destination within a shorter time. The main difference with our approach is that the protocol in [11] is designed for networks subject to partitions; in fact, it follows a store-carry-forward paradigm.

Termite is a biologically inspired probabilistic routing protocol which exploits the principle of swarm intelligence to define rules for each packet to follow [14]. As a packet is dispatched from the source to the destination, it follows the pheromone for its destination through the network while leaving pheromone for its source. Upon arriving to a node, say i, a packet with destination d is routed

randomly to one of the i's neighbor, say j, based on the amount of pheromone associated to d which is presents on the link connecting i to j. The value of pheromone of a link decays with time, while it is increased when packets travel along the link. Essentially, in this way a packet follows a biased random walk on the topology graph. Tuning the protocol's parameters, e.g., the decay rate as well as the number of route request packets required before sending data packets, is not a trivial task.

The Zone Routing Protocol (ZRP) is a hybrid proactive/reactive protocol [10] based on the notion of a zone. A zone of radius k is defined for each node i as the set of nodes at distance at most k hops from i. Paths to nodes inside a zone are maintained in a proactive way, while nodes outside the zone are reached by discovery a path on-demand. The only analogy with our approach is that in both cases the partial view of a node goes beyond the 1-hop neighbors.

Finally, our approach is somehow related to the ABR routing protocol [16]. The distinctive feature of ABR is the use of "associativity" as a primary metric in order to select more stable and thus long-lived routes. In ABR, each node generates a periodic beacon and counts the beacons received from its neighbors to update their associativity ticks, which are reset if not received for a suitable period of time. The protocol assumes that a high value of the associativity of a node i w.r.t. a node j indicates that the two nodes are likely to remain close to each other, since it was able to receive many consecutive beacons; thus, the wireless link between i and j is classified as long-lived. In our approach, a similar technique based on beacons is used to estimate the contact time, namely τ_{ij}.

3. Model and assumptions

We consider a system composed of N mobile devices, hereafter also called nodes, each uniquely identified. We say that two nodes are direct neighbors, or simply neighbors, if they lie within one another's transmission range. We assume that a node, on the average, has $\widetilde{N}$ neighbors. We call the Lookahead Zone of node i with parameter L, $Z_L(i)$, the set of nodes at distance less than or equal to L hops from i.

Two neighbor nodes can communicate according to the following primitives. A node i can issue: (i) *send*(pkt,j) to send the packet pkt to j and knowing the result of the transmission; (ii) *bcast*(pkt) to unreliably send the packet pkt to the nodes within the transmission range. In this case, the result of the transmission is unknown.

Each node maintains a local clock, not synchronized with the others. The value of the local clock is accessible through the *clock*( ) primitive.

The transmission range is independent from the node and is a circle with radius R, centered at the sending node. A node can immediately detect when any other is moving in or out of its range. When the distance between two nodes, say i and j, is decreasing (increasing) and it reaches R at time t, we say that i and j come in contact (loose their contact) at time t. From the time when they came in contact until they lost the contact we call them neighbors. This time interval is called the contact time or, equivalently, the duration of the link between i and j, denoted τ_{ij}.

4. The hint-based protocol

4.1. Algorithm description

Let us first describe the algorithm in general terms. Each node i is in charge of computing the hint h_{id} for any possible destination node d. The hint $h_{id}(t)$ computed at time t is zero if at this time i and d are neighbors, while it is ∞ if they never came in contact before t. If, however, they were 1-hop neighbors in the past and their last contact was lost at time t^*, then the hint is $h_{id}(t) = \frac{t-t^*}{\tau_{id}}$. Hints are disseminated within at most L hops from the estimating node.

When a node n needs to forward a packet destined to d, it sends the packet to the neighbor n^* which provides the best hint among the neighbors that never forwarded the same packet before. Note that a hint can be generated by a node behind n^*. If no such nodes can be found, then the packet is discarded.

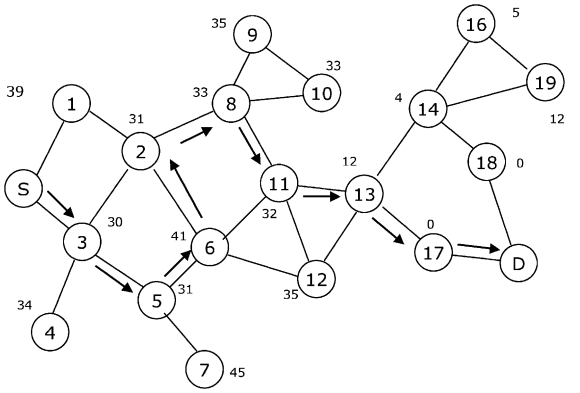


Fig. 2. An example of packet forwarding.

Fig. 2 shows an example of packet forwarding from the source node S to destination D, assuming lookahead $L = 1$. Hints are reported close to the generating nodes. Note that node 2 sends the packet to 8, instead of to 3, since the source has already used the node 3 before.

4.2. Algorithm details

We now describe how the components of our protocol, namely (i) Hint Computation, (ii) Hint Gathering and (iii) Packet forwarding, could be implemented.

4.2.1. Hint computation

Each node i broadcast a heartbeat packet every ΔT_B s and uses the beacon received from the neighbors to manage a vector of time information, VH_i, which stores the relevant time information for other potential destinations. In particular, the entry for a destination j, say $VH_i[j]$, stores: $VH_i[j].t_{\text{start}}$, the time when the first heartbeat from j was detected; $VH_i[j].t_{brk}$, the time when the link with j was broken (this value is 0 if j is currently a neighbor); $VH_i[j].t_{\text{last}}$, the time of the last heartbeat received from j; and $VH_i[j].\tau$, the duration of the link with j. All these values are initialized to ∞.

If the node i receives a heartbeat from j at time t, it sets $VH_i[j].t_{\text{last}} = t$ and, if $VH_i[j].t_{brk} \neq 0$, then it sets $VH_i[j].t_{brk} = 0$ and $VH_i[j].t_{\text{start}} = t$.

When i misses $M \geqslant 1$ heartbeats from$j$ it sets $VH_i[j].\tau = VH_i[j].t_{\text{last}} - VH_i[j].t_{\text{start}}$ and $VH_i[j].t_{brk} = VH_i[j].t_{\text{last}}$.

The hint at time t computed by the node j for the destination d is given by

$$h_{jd} = \begin{cases} 0 & \text{if } VH_j[d].t_{brk} = 0, \\ \frac{t - VH_j[d].t_{brk}}{VH_j[d].\tau} & \text{if } 0 < VH_j[d].t_{brk} < \infty, \\ \infty & \text{otherwise,} \end{cases}$$

where the values in the above computation are the ones stored in the hint vector at time t.

4.2.2. Hint dissemination

Each node manages a hint table with schema $\langle DEST, HINT, NEIGH, HOP, GEN, TTL \rangle$. A tuple in the table HT_i of a node i, namely (d, h, n, hop, g, b), indicates that for the destination d the hint h has been received from the neighbor node n and that such a hint was generated by the node g, which is hop hops away from i when using n as next hop; the tuple has time-to-live b, $0 \leqslant b \leqslant B$. To denote sub-tables, we will use conditions on the attribute of the table. For example, $HT_i[DEST = d]$ represents the sub-table composed of all entries for the destination d.

Hint dissemination is achieved by broadcasting control messages, which are carried by beacon packets. However, if the size of the control information does not fit into a single beacon packet then additional control packets are used.[1]

For each destination d, the control message sent by a node i carries $L - 1$ lists, namely $AL_d^1, \ldots, AL_d^{L-1}$ as well as its own hint, h_{id}; each list is composed of a number of triples, say (h, g, b), where h is a hint, g the generating node and b the time-to-live of the triple. A tuple (d, h, n, k, g, b) in HT_i produces a triple (h, g, b) in AL_d^k if h is the lowest hint in $HT_i[NEIGH = n,\ DEST = d,\ HOP = k]$ and g does not belong to $HT_i[DEST = d,\ HOP < k]$. In other words, h is the best hint generated by a node g, which is reachable in k hops through n. The same generating node can be seen via different neighbors and at different distance. This redundancy is required to promptly recover from a link breakage and provides resilience to topological changes. The

[1] When the protocol is started, say at local time t^*, the first transmission is scheduled uniformly at random in the range $[t^* \ldots t^* + \Delta T_B]$.

pseudocode code of hint dissemination protocol is reported in Fig. 3.

When a node i receives a control packet from a neighbor, say j, it extracts the triple with the lowest hint from each list AL_d^k, say the triple (h,g,b), and creates (or updates if already present) the entry $(d,h,j,k+1,g,b)$ in HT_i. Clearly, if a triple with $g=i$ appears in AL_d^k, then it is ignored in the computation. Moreover, if an entry for g is already present in the table, and g is seen via a different neighbor, then the distance is used as a measure of the freshness of the information, i.e., the value associated to the lowest distance overwrites the others.

To better understand the update mechanism, let us consider the Fig. 4; the update process occurring in a static portion of a network is here sketched for $L=2$. To simplify the example, only the parts of the tables devoted to a given destination are shown while the time-to-live column is omitted.

Suppose that all nodes started the dissemination protocol at the same time with an empty table and, without loss of generality, that the node 1 schedules the first transmission, then node 2, and so on. The initial hint computed by a node is depicted close to the node and, for convenience, its fractional part is assumed to be equal to the id of the node; e.g., the first node 1 computes the hint 0.1. For the sake of simplicity we also assume that the hint increases at steps of 0.01 at every new computation; e.g., the second time node 1 finds 0.11, the third 0.12 and so on. The first "round" of dissemination ends after all nodes have received the packets from all neighbors; this is indicated by a dashed line.

Fig. 4 shows the content of the tables as the time passes; the entries are the ones calculated after processing the incoming control packets, which are indicated by an arrow with the sending nodes close to it. The updates occurring in a round are split into two parts if a node receives further packet after sending its own control packet. For example, node 2 receives a packet from node 1 and, after sending its own control packet, it receives other two packets from 4 and 6. Finally, the entries containing the information sent in a control packet are drawn in bold; since $L=2$ they all correspond to neighbor nodes.

Let now consider the updating process in more details. The first control packet is sent by node 1, which just advices its own hint, namely 0.1. As a consequence, nodes 2,3 and 6 create the tuple $(n=1,hop=1,h=0.1,g=1,b=3)$ in their tables. When node 2 sends the packet it includes its own hint, $h=0.2$, and the one received from 1; the packet is received by nodes 4 and 6; all of them create a tuple $(2,2,0.1,1,3)$ in their tables. Node 3

Procedure Update() / * *Behavior of node i* * /
Init: Select the initial time randomly in $[clock()...clock()+\Delta T_B]$

1. **Every** ΔT_B **s do**
2. *Delete Stale Entries from HT*
3. *Prepare* AL_d^k, $\forall$ d, $k<L$
4. **bcast** $(\bigcup_{k,d} AL_k^d \bigcup_d h_{id})$
5. *Decrease TTLs*

1. **Upon receiving a control packet do**
2. t $\leftarrow$ *GetBestIn* (AL_d^k)
3. *UpdateTableWith* (t)

End.

Fig. 3. Pseudocode of hint dissemination.

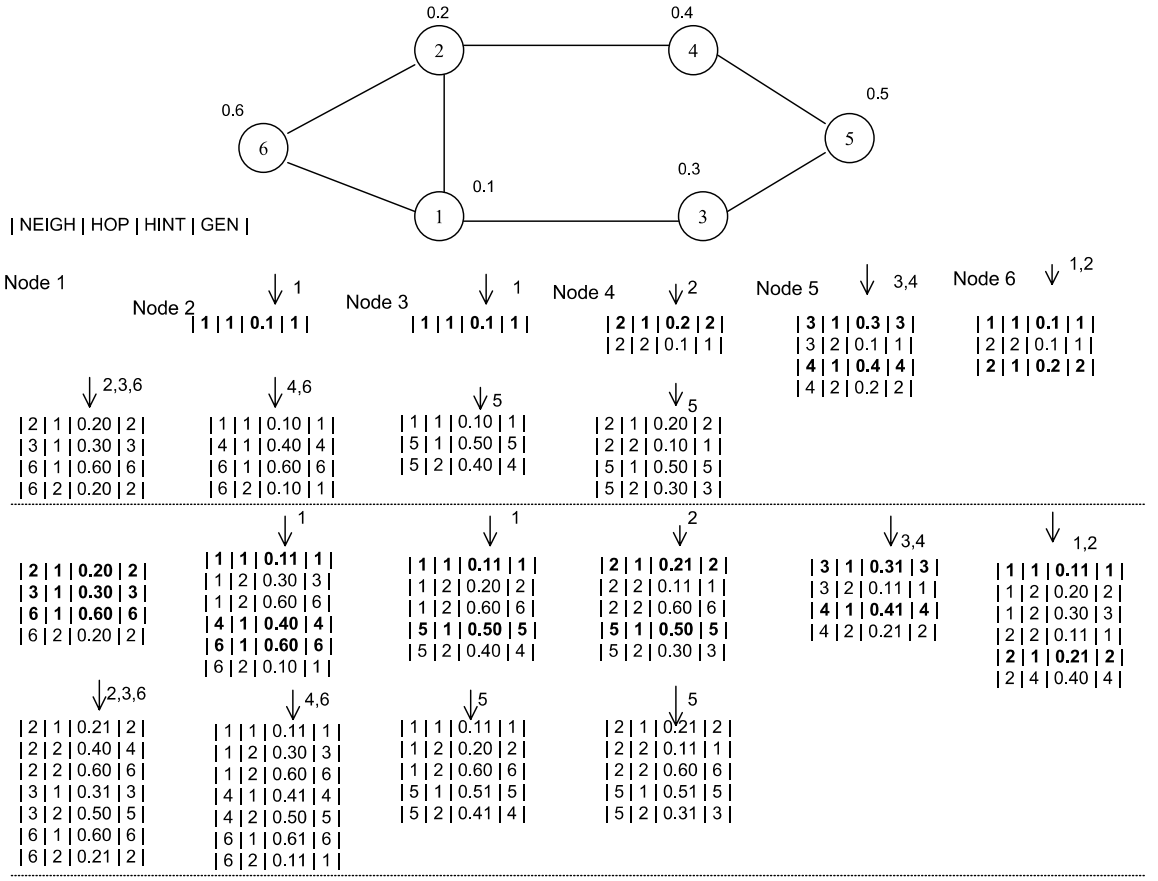


Fig. 4. An example of hint dissemination.

then sends the packet with its hint plus the ones received from 1 and 2; the corresponding entries are created in the tables of node 5. Similarly, node 4 sends a control packet containing its own hint and the ones received from 2, and so on.

Please note that the packet sent by node 6 contains the first hint generated by 1, i.e., $h = 0.1$. Accordingly, when 2 receives the packet, it creates the entry $(6, 2, 0.1, 1, 3)$. However, on receiving the second control packet from 1, which carries the new hint 0.11, node 2 updates the above tuple to $(6, 2, 0.11, 1, 2)$.

To evaluate the average number of control bits sent by a node, let us assume that c_d bits are used to represent the destination's id, c_g bits the generating node, c_{hop} the number of hops, c_h the hint, and c_b the time-to-live. To code a list AL_d^k we need $(c_d + c_{hop}) + \widetilde{N} \times (c_h + c_g + c_b)$ bits, and hence the number of bits required when considering all nodes and lookahead L is $(N-1) \times (L-1) \times [(c_d + c_{hop}) + \widetilde{N} \times (c_h + c_g + c_b)]$ bits plus $(N-1) \times c_h$ bits for the hint generated by the sending node itself. As a numerical example, for a network of $N = 100$ nodes, assuming $\widetilde{N} = 10$ nodes (see [4]), $L = 2$, $c_g = c_d = 8$, $c_b = c_{hop} =$ and $c_h = 4$ bits, the ≈2.1 KB is required.[2]

4.2.3. Packet forwarding

The pseudo-code of the forwarding algorithm is reported in Fig. 5. The protocol assumes that each

[2] According to the maximum size specified in the 802.11g standard, which is 4095 bytes, in practical terms this means that in this case only one packet can be used.

Procedure Forward(pkt, d) / * *Behavior of node* i * /
1. $S = \{j : VH[j].t_{brk} = 0\} - \{j : pkt.V[j] = 1\}$;
2. Order S w.r.t. hints in $HT[][d]$;
3. $pkt.V[i] = 1$;
4. **for** $k = 1$ **...** $|S|$ **do**
5. $send\_ok =$**send**(pkt, S_k);
6. **if** ($send\_ok$) return 1;
7. $VH[S_k].t_{brk} = clock()$;
8. $VH[S_k].\tau = t - VH[S_k].t_{start}$;
9. **endfor**;
10. Discard pkt; return 0;

End.

Fig. 5. The packet forwarding procedure.

packet is equipped with a vector V of visited nodes, initialized to 0. Upon receiving the packet destined to d, a node i determines an ordered list of possible next-hop nodes (line 1). The order is determined by the value of the hints, with ties broken at random (line 2). Then, it tries to forward the packet to the node at the head of the list (line 5). If the transmission fails then the time information associated to the first node, S_1, are updated (7–8) and i can promptly use S_2, the second node of the list; if also this attempt fails, the third node could be used, and so on, until either the packet is successfully sent or the list is empty and consequently the packet is dropped. A simple modification can be introduced here: before giving up, the node i could solicit hint gathering from the neighbors, in the hope that a new neighbor is available.

The above forwarding protocol satisfies the following property.

Property 1. *The Hybrid Probabilistic protocol either delivers a packet to the destination within at most* $N - 1$ *forwarding steps, or it discards the packet within at most* $N - 2$ *forwarding steps.*

This property follows from the definition of the algorithm. Suppose that after $N - 2$ forwarding step the packet has neither delivered or discarded. Since at each forwarding step a new node is selected, therefore the packet visited $N - 1$ nodes; thus, the remaining node is the destination. The node that is currently holding the packet can perform another forwarding steps only if the destination is its neighbor (in this case a total of $N - 1$ steps are required to deliver the packet), otherwise it must discard the packet. Note that the topology of the network can change during packet forwarding.

5. Hint–distance correlation

This section describes a discrete-time discrete-space analytical model for studying the hint–distance relationship. The model is inspired to the so called Manhattan-like topology, which is used to represent a city with major streets running east–west and north–south. In this topology, the user can move along either the horizontal or the vertical direction.

In particular, we will derive the conditional probability that the distance between two nodes is k assuming that the time elapsed since they lost their contact is t. The relative speed is also captured by the model.

5.1. Model

Consider two mobiles, a and b, that can move on a 2-D torus with $2H \times 2H$ points (see Fig. 6(a)). The elementary movements are discrete

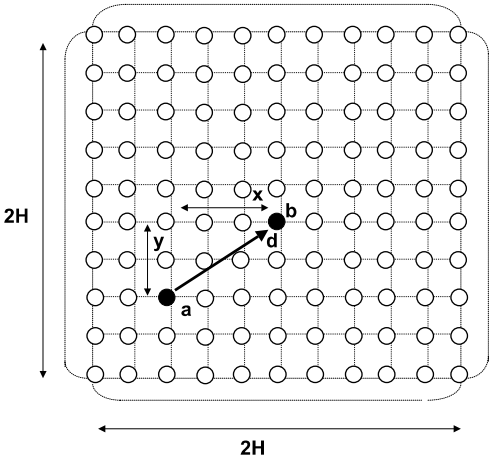


Fig. 6. A configuration on a $2H \times 2H$ torus.

in time. At each time tick a mobile decides either, with probability $\alpha < 1$, to remain in the same place or, with probability $\beta = \frac{1-\alpha}{4}$, to move to an adjacent point.

The relative position of the mobiles at time t is expressed by a vector of random variables, $\mathbf{D}_t = (x, y)$, where x and $y \in [0, \ldots, H]$ (recall that the topology is wrapped). The pair $\mathbf{d} = (x, y)$ is called a configuration (see Fig. 6(a)). The distance norm is $\|(x, y)\| = x + y$.

If $\mathbf{d} \in A \doteq \{(0,0),(0,1),(1,0)\}$ then we say that a wireless link between a and b exits or, equivalently, that the mobiles are neighbors. Thus, two users communicate directly when they are at distance at most of one city block from each other.

Let the distance between the mobiles at a random instant of time be the r.v. M. The probability that the distance is k, $Pr\{M = k\}$, where $k = 0, \ldots, 2H$, can be obtained by computing the ratio between the number of points at distance k from an arbitrary point on the torus, say $N(k)$, and the total number of points, $4H^2$. Clearly, $N(0) = N(2H) = 1$. To calculate the other values, let us consider a lattice with $(2H + 1) \times (2H + 1)$ points, which represents the unfolded version of the torus, and assume a Cartesian system exits (see Fig. 7).

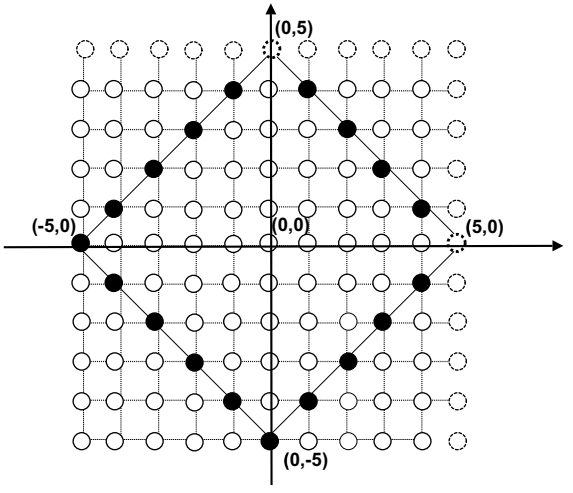


Fig. 7. A 11×11 lattice used to compute the number points (the black ones) at distance 5 from the center.

There are $4k$ points at distance $k > 0$ from the center which lie on the square defined by the corners $(0, \pm k)$ and $(\pm k, 0)$. Thus, for $k < H$, $N(k) = 4k$. For $k \geqslant H$ we have to take into account that in the wrapped version the points (x, H) and $(x, -H)$, as well as (H, y) and $(-H, y)$, are actually the same points (thus they have to be counted as 2 points), while the points (i,j) such that $|i| + |j| > H$ do not exit. Hence

$$N(k) = \begin{cases} 1 & \text{if } k = 0 \text{ or } k = 2H, \\ 4k & 0 < k < H, \\ 4k - 2 & k = H, \\ 4(2H - k) & \text{otherwise.} \end{cases}$$

The required probability is obtained by dividing $N(k)$ by the total number of points, $4H^2$.

$$Pr\{M = k\} = \frac{1}{H^2} \begin{cases} \frac{1}{4} & \text{if } k = 0 \text{ or } k = 2H, \\ k & 0 < k < H, \\ H - \frac{1}{2} & k = H, \\ 2H - k & \text{otherwise.} \end{cases} \tag{1}$$

Let us now calculate the conditional probability of the configuration between the two nodes being (x, y) assuming that at time $k = -1$ they were neighbors and that for all times k, $0 \leqslant k \leqslant t$, they were not, namely $Pr\{\mathbf{D}_t = (x, y) | \mathbf{D}_{-1} \in A \wedge \mathbf{D}_i \notin A, i = 0, \ldots, t\}$. Such a probability will be denoted as $P_c(t, x, y)$.

Consider the discrete-time Markov chain whose states correspond to configurations. The states belonging to A are absorbing states (see Fig. 8). The figure also sketches the state transitions allowed from two representative states, as detailed next in this section. The initial state of the chain, $\mathbf{s}$, belongs to $B = \{(0,2),(2,0),(3,0),(0,3),(2,1),(1,2),(1,1)\}$. The set B is composed of the configurations reached after the wireless link between the mobiles breaks.

If $Pr\{\mathbf{X}_t = (x, y) | \mathbf{X}_0 = \mathbf{s}\}$ is the probability that at time t the state of the chain is (x, y) assuming the initial state $\mathbf{s}$, then the ratio

$$\frac{Pr\{\mathbf{X}_t = (x, y) | \mathbf{X}_0 = \mathbf{s}\}}{\sum_{(i,j) \notin A} Pr\{\mathbf{X}_t = (i, j) | \mathbf{X}_0 = \mathbf{s}\}}$$

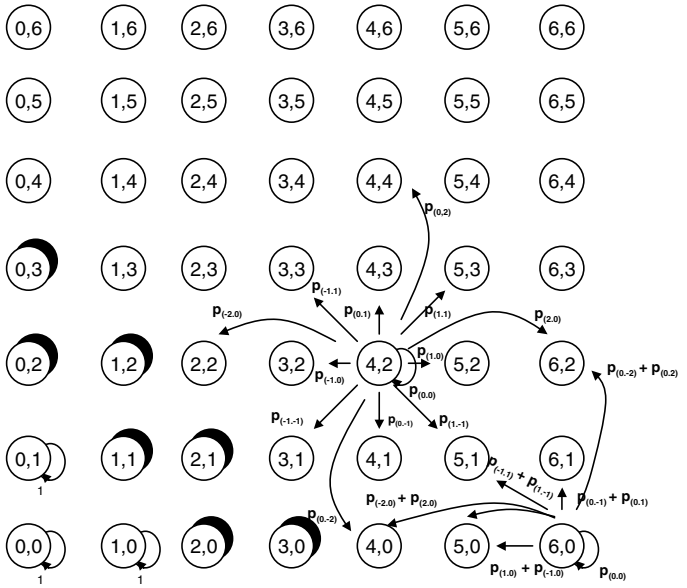


Fig. 8. The Markov chain associated to a 12×12 torus.

gives the probability that at time t the state is (x, y), assuming that (x, y) is not an absorbing state and that the initial state of the chain was $\mathbf{s}$. This is also the probability that the configuration of the two nodes on the torus at time t is (x, y) assuming that after they broke the link the configuration was $\mathbf{s}$ (i.e., $\mathbf{D}_{-1} \in A$, and $\mathbf{D}_0 = \mathbf{s}$) and that they never met again after such a breakage. Thus, we can write

$$P_c(t,x,y) = \sum_{\rho_s s.t. s \in B} \rho_s \frac{Pr\{\mathbf{X}_t = (x,y) | \mathbf{X}_0 = \mathbf{s}\}}{\sum_{(i,j) \notin A} Pr\{\mathbf{X}_t = (i,j) | \mathbf{X}_0 = \mathbf{s}\}}, \tag{2}$$

where ρ_s is the probability of the configuration $\mathbf{s}$ being observed after a link breakage.

Let us now return to the definition of the chain. The state transition probabilities can be conveniently written by exploiting the relationship between two consecutive configurations, $\mathbf{d}$ and $\mathbf{d}'$. For this purpose, we use the update vector $\delta = (\delta_x, \delta_y)$, where δ_x, δ_y are integers such that $|\delta_x + \delta_y| \leqslant 2$ and $|\delta_x|, |\delta_y| \leqslant 2$, to determine a change in the configuration; the relationship can be written as $\mathbf{d}' = \mathbf{d} \oplus_H \delta = (x \oplus_H \delta_x, y \oplus_H \delta_y)$, where the operator $\oplus_H$ is defined as

$$x \oplus_H \delta_x = \begin{cases} x + \delta_x & \text{if } 0 \leqslant x + \delta_x \leqslant H, \\ 2H - (x + \delta_x) & \text{if } x + \delta_x > H, \\ |x + \delta_x| & \text{otherwise.} \end{cases}$$

We associate to the vector δ the probability p_δ with which it can be generated by two elementary movements (see Fig. 9). The probability is readily obtained by thinking one node fixed and the other one performing two movements per unit of time and by assuming $H = \infty$ and $x + y > 2$. Hence,

$$p_{(0,0)} = \alpha^2 + 4\beta^2,$$
$$p_{(1,0)} = p_{(-1,0)} = p_{(0,1)} = p_{(0,-1)} = 2\alpha\beta,$$
$$p_{(2,0)} = p_{(-2,0)} = p_{(0,2)} = p_{(0,-2)} = \beta^2,$$
$$p_{(1,1)} = p_{(1,-1)} = p_{(-1,1)} = p_{(-1,-1)} = 2\beta^2.$$

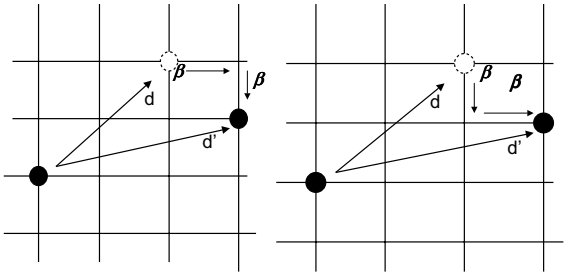


Fig. 9. Computation of $p_{(1,1)}$.

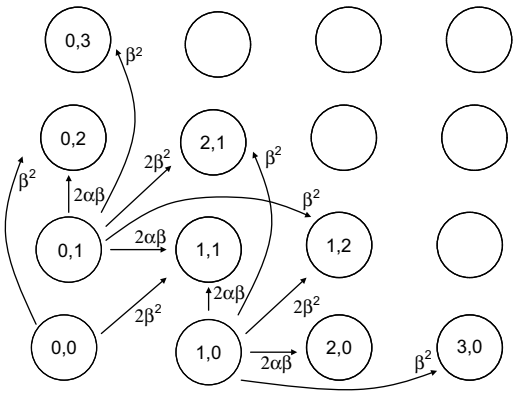


Fig. 10. Computing the probability of the initial state.

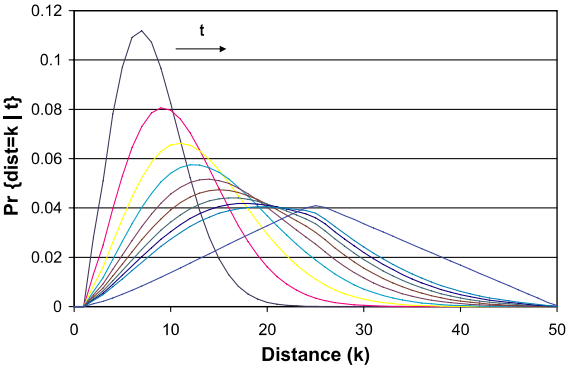


Fig. 11. Expected distance vs. elapsed time.

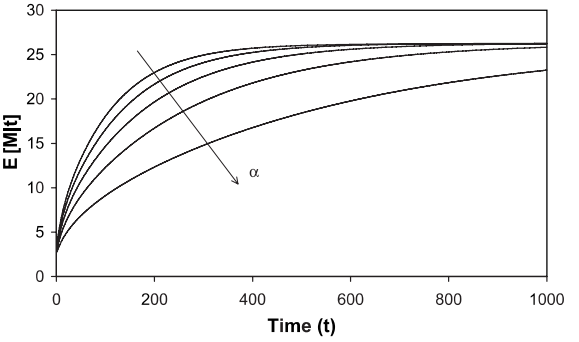


Fig. 12. Conditional expected distance vs. elapsed time.

The probability of transition from state (x,y) to state (x',y'), $P_{(x,y),(x',y')}$, is one if $(x',y') = (x, y) \in A$ (these are the absorbing states), and zero if $|x - x'| + |y - y'| > 2$. For the other cases (i.e., $|x - x'| + |y - y'| \leqslant 2$ and $(x,y) \notin A$):

$$P_{(x,y),(x',y')} = \sum_{\forall \delta s.t. (x,y) \oplus_H \delta = (x',y')} p_\delta$$

As far as the probability ρ_s is concerned, we note that the probability of observing the configuration $\mathbf{s} \in B$ after a breakage, namely $\rho'_s \doteq Pr\{\mathbf{D}_{-1} \in A, \mathbf{D}_0 = s\}$, is given by (see Fig. 10):

$$\rho'_{(2,0)} = \rho'_{(0,2)} = Pr\{M = 0\}\beta^2 + \frac{Pr\{M = 1\}}{2} 2\alpha\beta,$$
$$\rho'_{(1,1)} = Pr\{M = 0\}4\alpha\beta Pr\{M = 0\} + 2\beta^2,$$
$$\rho'_{(3,0)} = \rho'_{(0,3)} = \frac{Pr\{M = 1\}}{2}\beta^2,$$
$$\rho'_{(1,2)} = \rho'_{(2,1)} = \frac{Pr\{M = 1\}}{2} 3\beta^2$$

from which, normalizing, we obtain $\rho_s = \frac{\rho'_s}{\sum_{z \in B} \rho'_z}$. P_c can now be obtained by solving the chain and using (2).

5.2. Numerical results

We now give some numerical result assuming $H = 25$. Fig. 11 shows the probability that, after t time units the link between the mobile was broken, the distance between the mobiles is k, namely $\sum_{x+y=k} P_c(t,x,y)$. The curves are calculated for $\alpha = 0$ and for $t = 10, 20, \ldots, 90, 1000$. For a large t such a distribution approaches the distribution of M, see Eq. (1). We can see that for small values of t there is a high probability that the mobiles are close to each other.

Fig. 12 plots the average distance of the two mobiles assuming t units from the link breakage are elapsed. The probability that does not move, α, is given as a parameter and takes on the values $0, 0.2, 0.6, 0.8$. Note that α determines the relative speed of the mobiles. The average value increases with time and approaches the average distance observed at a random time between the mobiles. The time interval during which an appreciable difference between these two distances exists can be taken as a measure of the lifetime of the hint–distance correlation. The shortest duration is clearly achieved when $\alpha = 0$, since at each time tick a mobile changes its position.

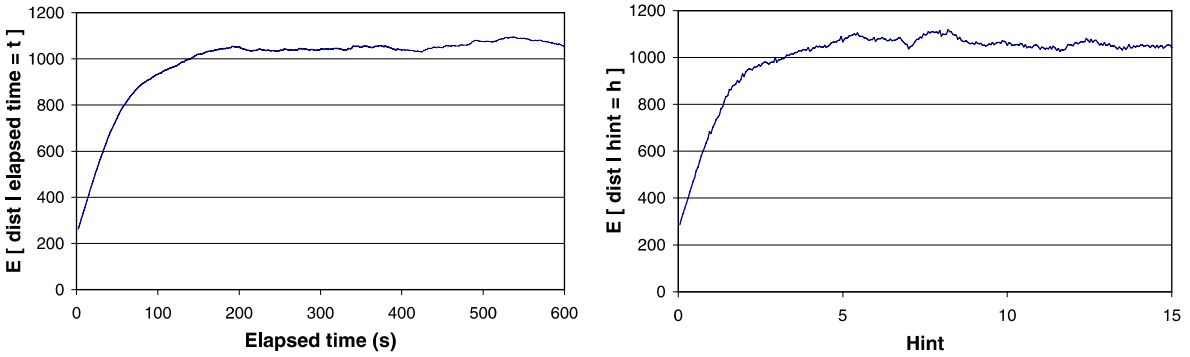


Fig. 13. Empirical conditional expected distance vs. elapsed time since the last contact (left), and elapsed time (right).

In the left part of Fig. 13 we report the empirical conditional expected euclidian distance of a node from the destination given the elapsed time since the last contact with the destination. The same quantity as a function of the hint is reported in the right. The values were estimated by simulating two nodes moving into an a square area with edge 1.5 km according to the Random Waypoint mobility model [13] with zero pause time, minimum speed 1 m/s and maximum speed 20 m/s. The destination node sent a beacon every 250 ms. The duration of the simulation was 500,000 s. It can be clearly seen how both relationships follow a shape similar to the one found in our model.[3]

Consider now the case of several mobiles moving independently on the torus. One of them plays the role of target mobile, say *dest*, while the other ones are interested in delivery messages to it. Each mobile i registers the time since it has most recently seen the target in a local variable, say t_i.

Suppose that the configuration of a node j w.r.t. *dest* is $\mathbf{d}_j$ and that j needs to forward a packet destined to *dest*. Further assume that j can decide among two nodes, say n_1 and n_2 whose times are t_1 and $t_2 = t_1 + \Delta T$. What is the probability, say P_{opt}, that n_1 is closer than n_2 to the destination? This value tell us the probability that the node providing the lowest elapsed time is the correct choice. We have

$$P_{\text{opt}} = \frac{\sum_{(x_1,y_1),(x_2,y_2)\in Z_L(\mathbf{d}_j),\|(x_1,y_1)\|<\|(x_2,y_2)\|} P_c(t_1,x_1,y_1)P_c(t_1+\Delta T,x_2,y_2)}{\sum_{(x_1,y_1),(x_2,y_2)\in Z_L(\mathbf{d}_j)} P_c(t_1,x_1,y_1)P_c(t_1+\Delta T,x_2,y_2)},$$

where $Z_L(\mathbf{d}_j)$ is the set of points of the torus at distance less than or equal to L from j.

Fig. 14 shows such a probability for the configuration of $\mathbf{d}_j = (5,5)$ and $L = 2$. The probability that n_1 is the correct choice increases with ΔT and as t_1 decreases. The figure also reports the probability for a random selection. Fig. 15 shows such a probability as a function of L. The value of the probability of a random choice is also reported.

Although the model studied above is simple, we guess that it is able to capture some general properties of a mobile environment and hence used to derive some general principles. If t_1 and τ_1 (t_2 and τ_2) are the time information of the node $n_1(n_2)$ w.r.t. a target node, then we can assume that, on the average, the probability that n_1 is closer than n_2 to the target is higher than the one associated to a random choice between the two nodes when:

[3] In a preliminary simulation study, we have also considered the time since the last contact of a node with the destination as hints. However, the performance results were worst than when using our current definition.

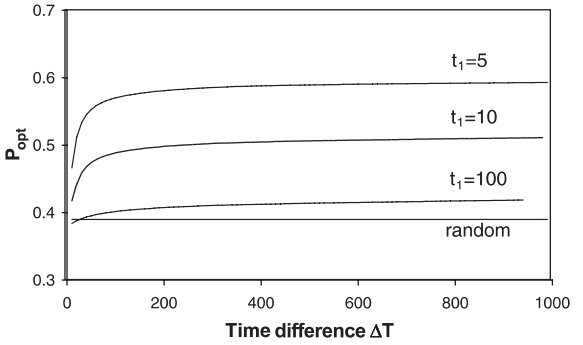


Fig. 14. Probability n_1 is closer than n_2, given $t_2 = t_1 + \Delta T$.

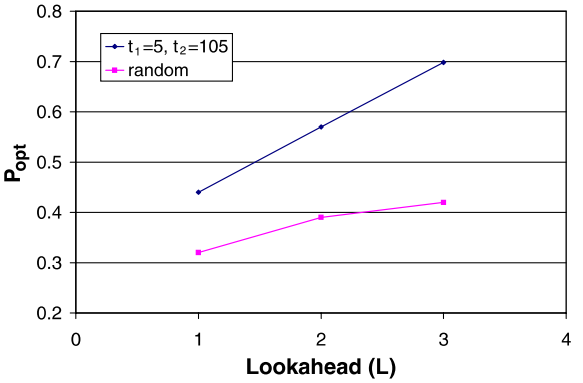


Fig. 15. Probability that n_1 is closer than n_2, given that $t_1 = 5$ and $t_2 = 105$, as a function of L; $t_1 = 5$, $t_2 = 105$.

- t_1 is lower than a critical value T^* (time-space correlation is still valid),
- $\tau_1 - \tau_2 \approx 0$ (the duration of wireless link was almost the same),
- and $t_2 - t_1 > 0$ (n_1 has seen the target more recently than n_2).

Moreover, such a difference increases if: (i) t_1 decreases; (ii) $t_2 - t_1$ and/or $\tau_1 - \tau_2$ increase; (iii) the lookahead L increases.

6. Simulation

To asses the performance of the proposed protocol we have developed a discrete event simulation tool, which we used to study the Hint Based Protocol with lookahead $L = k$, HP-k, for $k = 0, 1, \ldots, 3$. The case $L = 0$ corresponds to a random walk and is used to validate the effectiveness of the hint mechanism. Hints are advertised every $\Delta T = 0.5$ s, while $\alpha = 1.5$ and $B = 2$.

HP-k has been compared against an implementation of the AODV protocol. In the AODV protocol we have considered, a node floods the network with a Route REQuest (RREQ) packet for a destination node d when it determines that it needs a route to it. Each node retransmits all packets seen for the first time and, in this cases, sets a backward path to s. When the RREQ packet reaches the destination, a Route REPply (RREP) packet is sent back to the source along the backward path. Each node that participates in forwarding the RREP back to the originator of the request, the node s, creates a forward route to d. If not used, backward paths are deleted after a time-out of 3 s. If a RREP packet is not received within 3 s, a new discovery is initiated. The number of attempts before giving up is 3. When a node of an established path cannot forward a data packet, it sends a Route ERRor (RERR) packet back to the source, which then triggers a new route discovery.

6.1. Simulation model

6.1.1. Transmission primitives

Packet transmissions are governed by an ideal scheduler. A FIFO buffer of 20 packets in size is used at each node. A broadcast packet is served (i.e., initiated) after the channel is free for a Random Assessment Delay (RAD) randomly chosen in the range $[0 \ldots 500]$ ms. The transmission radius is $R = 250$ m. A packet reception is notified to a sender's neighbor provided that it remained for the whole duration of the transmission within the transmission range and such that no collisions with other transmissions occurred in the meanwhile.

The transmission of a new unicast packet is initiated only if it does not collide with others. The reception of the packet is notified if the sender-receiver distance is less than R for the whole duration of the transmission. At the end of the transmission, the scheduler checks whenever other packets awaiting in the sending buffers can be served.

The time required to detect a link breakage is simulated by considering a typical retransmission mechanism with exponential back-off. If the destination of a packet moves out of the range, the sender is notified with a transmission failure after $50T\mu s$, where T is the sum of seven random variables such that the ith r.v. takes on a value in the range $0 \ldots 2^i$. The nominal transmission speed is 11 Mbps.

We guess that such a simplified model captures the main behavior of a typical wireless link layer, while avoiding to bind the results to a specific implementation.

6.1.2. Mobility

Nodes can move into a square shaped region of edge E km according to the Random Waypoint mobility model with zero pause time (see [13]). The default value is $E = 1.5$ km. At the beginning of the simulation nodes are placed uniformly at random in the region. Each node then selects a new point and travels towards it at a constant speed, which is chosen in the range $[1 \ldots v_{\max}]$ m/s uniformly at random. When the node arrives at the destination, it repeats the same behavior.

6.1.3. Traffic

Packets are generated by Constant Bit Rate (CBR) sources at the rate of 5 packets/s and are 512 bytes in length. A source always sends packets to the same destination. The default number of sources is ten. The following table reports the simulation's parameters (see Table 1).

Table 1
Default simulation parameters

Parameter	Values
Simulation time	1500 [s]
Number of nodes	120
Node speed's range	$[1 \ldots v_{\max}]$ [m/s]
Max speed, $v_{\max}$	20 [m/s]
Pause time	0
Transmission radius, R	250 [m]
Look-ahead zone, L	0–3 [hops]
Message transmission rate	5 [msg/s]
Number of CBR sources	10
Message length	512 [bytes]
Transmission speed	11.0 [Mbps]
Random assessment delay	[0 . . . 500] [ms]
Sending buffer	20 [packets]
Update interval, ΔT_B	500 [ms]
Time-to-live, B	2
Allowed number of missed heartbeats, M	1

6.2. Performance metrics

The following metrics were estimated during a simulation:

- Delivery probability, ratio of the number of data packets delivered to the destinations to those generated by the traffic sources.
- Normalized overhead, ratio of the number of control packets to the number of data packets delivered to the destinations. For packets sent over multiple hops each single hop is counted as one transmission.
- Average path length, given in number of hops.
- End to end packet delay, the time elapsed from when a packet is generated by the source until it is delivered to the destination.

A warm up period of 200 s was used before collecting statistical data.

6.3. Simulation results

6.3.1. Varying the speed

We first analyze the performance as a function of the maximum speed (see Fig. 16). Please note that by increasing the maximum velocity, $v_{\max}$, both the mean and the variance of the nodes' speed increase.

Delivery probability. In AODV the probability to deliver a packet decreases as $v_{\max}$ is increased, since an increase in the velocity implies a shorter duration of paths. Two main opposite aspects determine the delivery probability of the HP protocols. On one hand, a high speed helps nodes to come in contact with each other; thus, the probability that a neighbor has a valid hint for the destination increases with the speed. On the other hand, as the speed increases the validity of the correlation is shortened. This explain why initially the delivery probability increases and then decreases. Note that the delivery probability is much lower if hints are ignored (see curve HP0); this validates hints as a useful means to track a destination node.

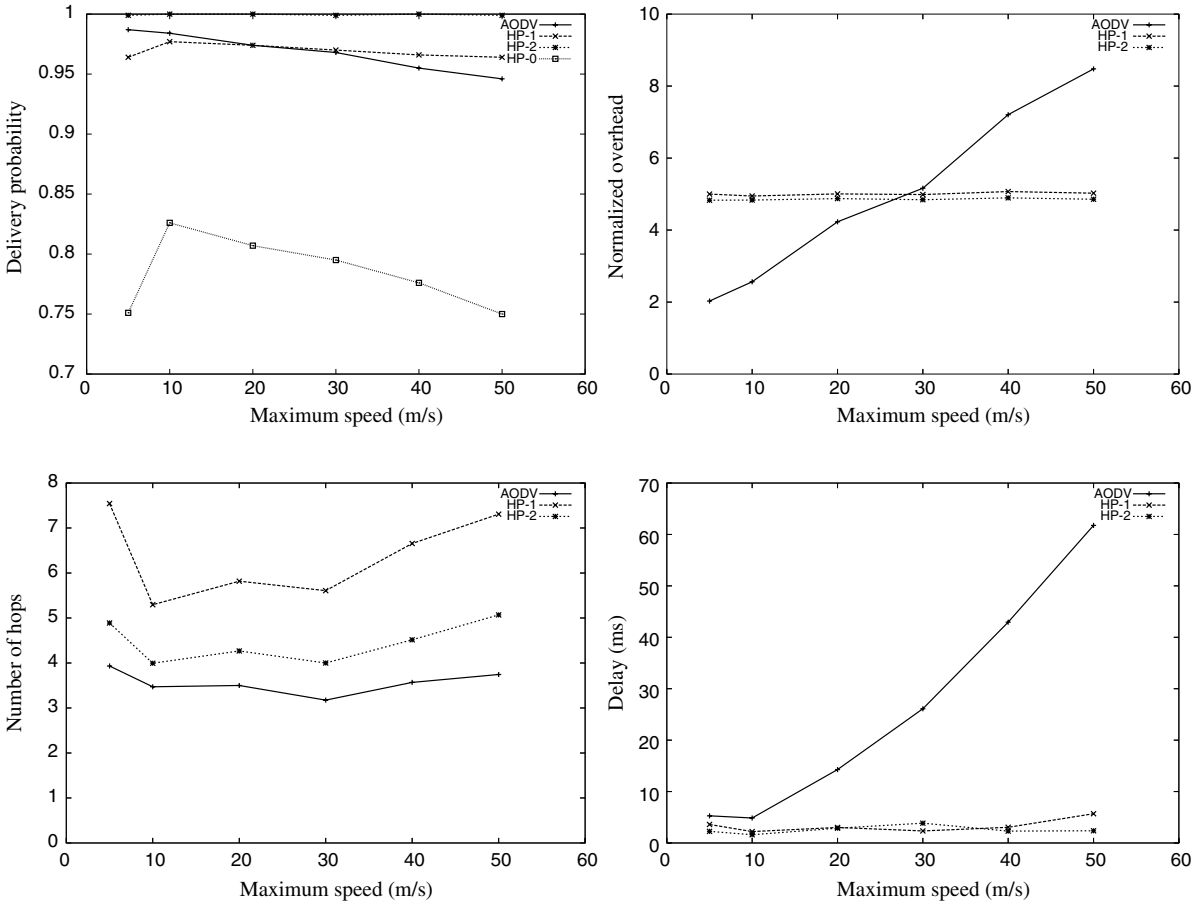


Fig. 16. Performance as a function of the speed.

Normalized overhead. The overhead generated by AODV increases with mobility as a consequence of the shorter paths' lifetime, see Fig. 16. On the contrary, the overhead of the Hint Protocol is almost independent from the speed, since it is generated periodically and the number of delivered packets decreases slightly with the speed; thus, the difference with AODV increases remarkably with the speed.

Number of hops and delay. In AODV a path is used until it breaks. Thus, even if due to a movement a shorter path is created, such a path will not be used until a new discovery is triggered. The need to discover a path increases with the mobility; hence, the probability that we are using a path longer than the optimal one is lower under high mobility than under low mobility. On the other hand, when the mobility is increased, paths have shorter lifetimes. This explain why the distance in number of hops first decreases and then it increases.

A similar behavior is also observed for HP. In this case, the value of the distance is determined by the two opposite effects previously mentioned for the delivery probability. On one hand, as the speed is increased nodes come in contact with each other more often, and this facilitates hint gathering. Moreover, the validity of the hint–distance correlation is shortened under high mobility.

Note that while the path length of HP1 is considerably longer than AODV's one, under HP2 such a difference is much lower. The length found by HP0, not reported in the figure, was $\approx$15 hops.

The end to end delay reveals an opposite behavior w.r.t. the number of hops; the AODV protocol is characterized by a much higher latency compared to our proposal. The delay is clearly expected to increase with the maximum speed since the number of route requests increases. However, a deeper understanding is learned by looking at the traces reported in Fig. 17, which refer to $L = 2$ and $v_{max} = 20$ m/s. We observed that in AODV the route reply packet sent back to the source is occasionally lost and consequently the new discovery is triggered only after the time-out, which was set to 3 s. During this period the newly generated packets are buffered at the sending node and thus experience a very long delay. Such abrupt variations are much smoother in our approach. In our case the "spikes" are only due time required to detect a link breakage.

6.3.2. Varying the number of sources

In this set of experiments we have studied how the number of sources affects the performance metrics. Results are summarized in the plots reported in Fig. 18.

Delivery probability. The probability to deliver a packet is practically independent from the number of sources. The reason is that we are working outside the congestion region, i.e., the cause of a packet loss is a link breakage in AODV or the lack of new chances in HP; packets were never lost due to a buffer overflow.

Normalized overhead. In AODV the normalized overhead increases with the number of sources, while it decreases when HP is used. This can be easily explained by considering that in AODV a higher number of sources means that a higher number of route discoveries needs to be initiated in the network (recall that in our experiments the source-destination pairs are all disjoints). On the contrary, in HP the number of control packets is constant and independent from the number of sources or destinations; thus, the average number of control packets per delivered packet, i.e. the normalized overhead, decreases if the number of delivered packets increases.

Number of hops and packet delay. By increasing the number of sources, the average traffic being processed by a node increases. As a consequence, a packet experiences a longer delay at each node because a longer time is required to access to the channel and since the average queue size at the sending buffer increases. Thus, the duration of the whole forwarding process is increased too. During such a time interval a shorter path between the source and the destination can be formed, but the packet being forwarded cannot exploit it. This explain way the number of hops increases. Accordingly, the average packet delay also increases with the number of sources.

6.3.3. Varying the number of nodes, fixed fraction of sources

The aim of this set of experiments was to investigate the scalability of the protocol w.r.t. the number of nodes (see Fig. 19). The area of the region was also increased to assure the same node density; hence, the diameter of the network, i.e. the maximum number of hops, increased too. The per-

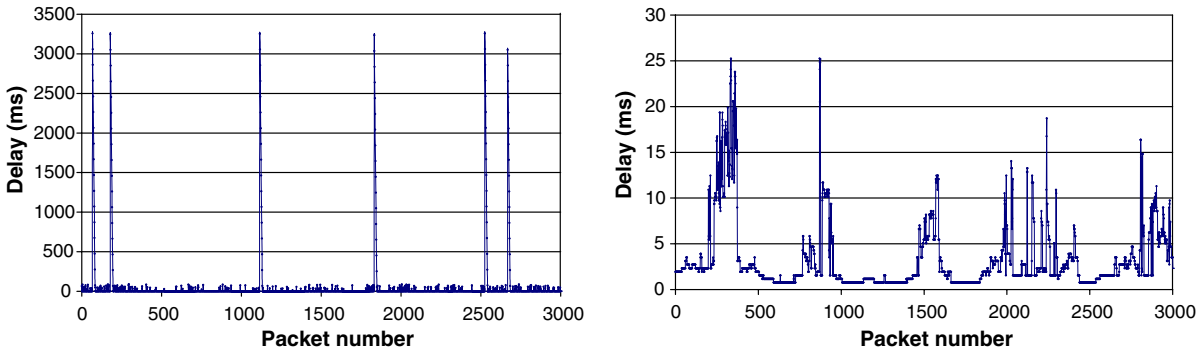


Fig. 17. Simulation trace of AODV (left) and HP2 (right), $L = 2$, $v_{max} = 20$ m/s.

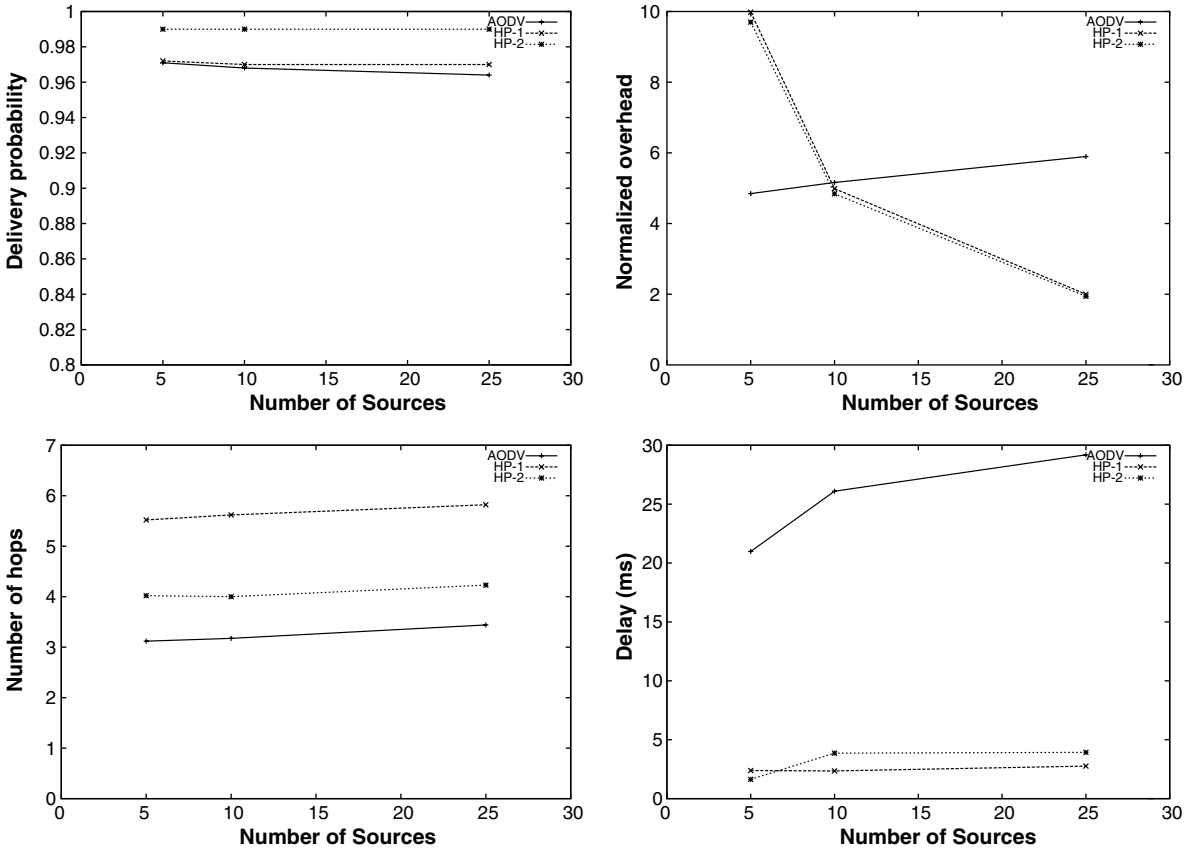


Fig. 18. Performance as a function of the number of sources.

centage of the source-destination pairs respect to the total number of nodes is kept constant to 10%.

Delivery probability. The delivery probability of AODV decreases as a consequence of the shorter path's lifetime, which is in turn due to the higher number of hops. We can observe how HP1 rapidly deteriorates its delivery performance. When the diameter of the network is increased, in fact, hints gathered with such a small lookahead are very likely to provide no valid information, i.e., the probability that a node is closer than another is independent from their hints. Increasing the lookahead to $L = 2$, however, improves the performance considerably. For example, for $N = 480$ nodes the probability to deliver increases from 0.74 if $L = 1$ to 0.98 for $L = 2$. Clearly, $L = 3$ provided the highest delivery probability.

Normalized overhead. The normalized overhead measured in AODV increases with the number of nodes since the number of sources, and thus the number of path discoveries, as well as the cost of a single discovery increases with N. In the HP protocol when the number of nodes is increased the cost of disseminating hints increases, since a single packet cannot carries all the hints. For example, in the experiment with $N = 480$ nodes, 5 control packets were used on the average. Moreover, for HP1 the reduced delivery probability seen is another reason of the increase in the overhead. Nevertheless, the overhead was always lower than the AODV's one.

Number of hops and delay. The average number of hops increases with the number of nodes since the simulated area is also increased to keep the

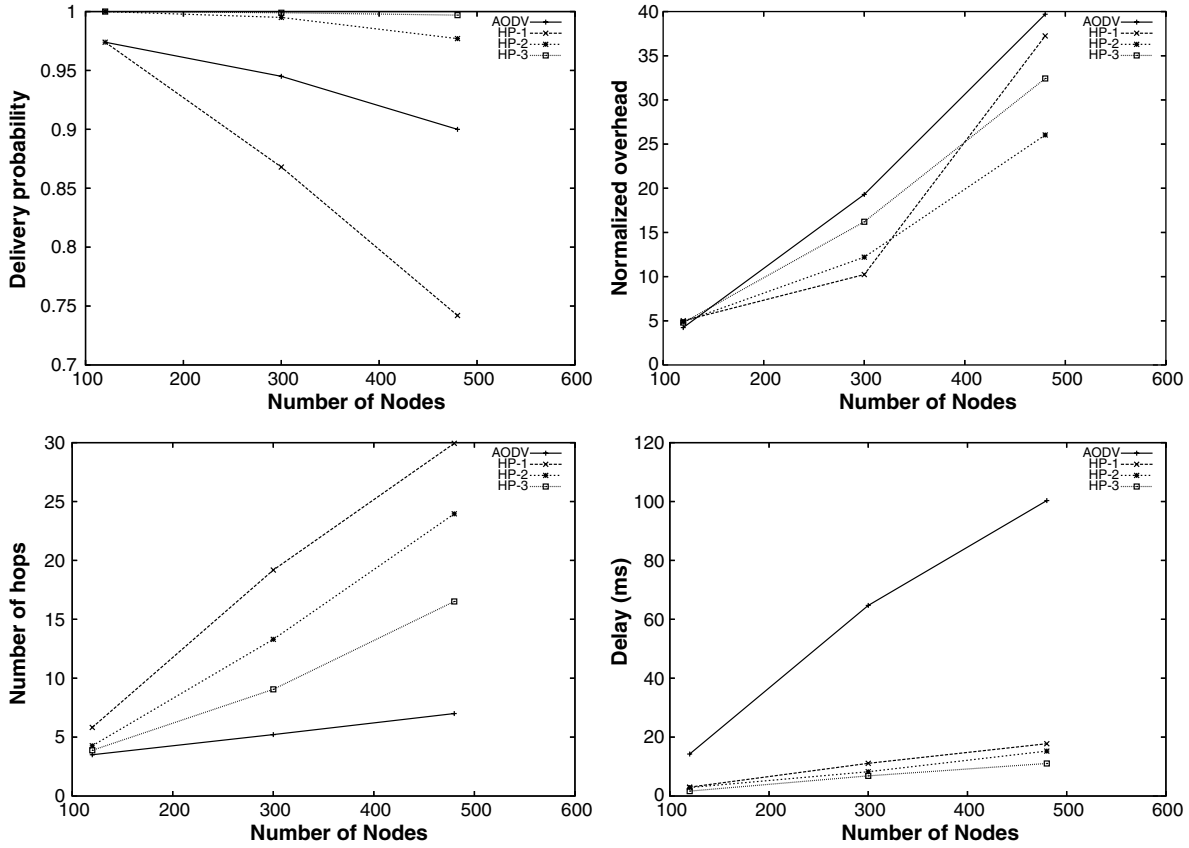


Fig. 19. Performance as function of the number of nodes, number of sources 10%.

node density constant. The increase in the number of hops is moderate in AODV and much more evident when the HP protocols are used, especially for $L = 1$. In this case, in fact, the hints are able to "push" packets towards the destination only when the distance of the packet from the destination is small. In the other cases, a packet is basically forwarded at random since the hint–distance correlation is no more valid. Clearly, an increase in the lookahead provides nodes with much more valid hints; then a reduction in the number of hops can be observed. However, as far the delay is concerned the HP protocol is again able to provide a much shorter value. Moreover, while the delay highly increases in AODV, the HP protocols are much less sensitive to the increase in the network's diameter.

6.4. Varying the update rate

Finally, we have conducted a set of experiments aiming at measuring the impact of ΔT_B on the performance. In these simulations $L = 1$ and $v_{\max} = 20$. In general, the control overhead is inversely proportional to (ΔT_B); thus, small values of ΔT_B produce a very high overhead. However, we found that small values are not necessary to achieve good delivery performance. For example, for ΔT_B varied in the range 100 ms...2.5 s, the delivery probability does not change while the number of hops increases smoothly. Clearly, if ΔT_B is high a node needs to perform several attempts before forwarding a packet, and this produces additional layer 2 control packets. Overall, $\Delta T_B = 500$ ms can be considered a good compromise.

7. Conclusion

This paper proposed a probabilistic protocol for unicast packet delivery in mobile ad hoc networks. Instead of using pre-computed paths, packet forwarding is driven by the meta-information (hints) a node gathers from the neighbors located within a small number L of hops (the protocol's lookahead) from itself.

We have shown through simulations that a small lookahead can result in a very good compromise between the delivery performance and the number of control packets. The paper also reported an analytical model for studying the hint–distance correlation. This approach can be extended to other communication paradigms, like multi-casting, or dynamic group communications.

References

[1] M. Abolhasan, T. Wysocki, E. Dutkiewicz, A review of routing protocols for mobile ad hoc networksAd Hoc Networks, vol. 2, Elsevier, 2004, Issue 1.

[2] R. Beraldi, On message delivery through approximate information in highly dynamic mobile ad hoc networks, in: The Seventh International Symposium on Wireless Personal Multimedia Communications, Italy, 12–15 September 2004.

[3] B. Bellur, R.G. Ogier, F.L. Templin, Topology broadcast based on reverse-path forwarding (TBRPF), in: IETF Internet Draft, draft-ietf-manet-tbrpf-01.txt, March 2001.

[4] C. Bettstetter, On the minimum node degree and connectivity of a wireless multihop network, in: Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2002).

[5] Charles E. Perkins, P. Bhagwat, Highly dynamic Destination Sequenced Distance-vector Routing (DSDV) for mobile computers, Computer Communications Review 1 (1994) 234–244.

[6] C.E. Perkins, E.M. Royer. Ad hoc On-Demand Distance Vector Routing, in: Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, February 1999, New Orleans, LA.

[7] I. Chlamtac, M. Conti, J.J.-N. Liu, Mobile ad hoc networking; imperatives and challanges, Ad Hoc Networks 1 (2003) 13–64.

[8] H. Dubois-Ferriere, M. Grossglauser, M. Vetterli, Age matters: efficient route discovery in mobile ad hoc networks using encounter ages, in: Proc. of ACM MobiHoc2003, 1–3 June 2003, Annapolis, MD.

[9] Z.J. Haas, J.Y. Halpen, Gossip-based ad hoc routing, in: Proceeding of IEEE INFOCOM 2002, New York, 23–27 June 2002.

[10] Z.J. Haas, M.R. Pearlman, The zone routing protocol (ZRP) for ad hoc networks, draft-ietf-manet-zone-zrp-02.txt, (Work in progress) June 1999.

[11] J.A. Davis et al., Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks, in: Proc. of ISWC 2001, 7–9 October 2001, Zurich.

[12] P. Jacquet, P. Muhlethaler, A. Qayyum, Optimized link state routing protocol, Internet Draft, draft-ietf-manetolsr-00.txt, November 1998.

[13] D.B. Johnson, D.A. Maltz, Dynamic source routing in ad hoc wireless networking, in: Mobile Computing, Kluwer Academic, 1996 (Chapter 5).

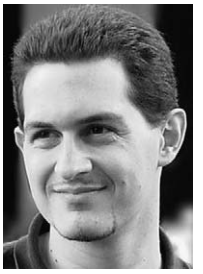
[14] M. Roth, S.Wicker, Termite: emergent ad-hoc networking, in: The Second Mediterranean Workshop on Ad-Hoc Networks, Medhia, Tunisia, 2003.

[15] S.Ni, Y. Tseng, Y. Chen, J. Sheu, The broadcast storm problem in a mobile ad hoc network, in: Proceedings of MobiCom'99, Seattle, WA, August 1999, pp. 151–162.

[16] C.-K.G Toh, A novel distributed routing protocol to support ad hoc mobile computing, in: Proc. IEEE 15th Annual International Phoenix Conference on Computers and Communications, IEEE IPCCC 1996, Phoenix, AZ, USA, 27–29 March, pp. 480–486.

[17] Q. Xue, A. Ganz, Ad hoc QoS on-demand routing (AQOR) in mobile ad hoc networks, Journal of Parallel and Distributed Computing 63 (2) (2003). Academic Press.

Roberto Beraldi received the Laurea degree in computer science from the University of Calabria, Italy, in 1991, and the Ph.D. degree in computer science in 1996. He is currently an assistant professor at the Department of Systems and Computer Science (DIS) at Univeristy "La Sapienza", Rome. His research interests include distributed systems, data dissemination and dynamic networks.



Leonardo Querzoni is a Ph.D. student in Computer Science at University of Rome "La Sapienza". His main research interests are related to information dissemination through publish-subscribe systems, dynamic systems and ad-hoc networks.

Roberto Baldoni is Professor of Distributed Systems at the University of Rome "La Sapienza" where he published more than one hundred peer-reviewed papers (from theory to practice) in the fields of distributed and mobile computing, middleware platforms and information systems. He is the founder of Middleware Laboratory <http://www.dis.uniroma1.it/%7Emidlab> (MIDLAB) whose members actively participate to national and European research projects. Roberto Baldoni regularly serves both as an expert for the EU commission in the evaluation of EU projects and as a member of organizing and program committees of premiership international conferences and workshops of his research areas. He was invited to chair the program committee of the "distributed algorithms" track of the /19th IEEE International Conference on Distributed Computing Systems/(ICDCS-99) and /PC Co-chair of the ACM International Workshop on Principles of Mobile Computing/(POMC).