Available online at www.sciencedirect.com

ScienceDirect

Ad Hoc Networks 6 (2008) 79–91

www.elsevier.com/locate/adhoc

The polarized gossip protocol for path discovery in MANETs ☆

Roberto Beraldi *

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

Received 5 October 2005; received in revised form 3 May 2006; accepted 26 August 2006
Available online 2 October 2006

Abstract

In this paper we present a novel probabilistic protocol for path discovery in Mobile Ad Hoc Networks (MANETs). The protocol implements what we call a polarized gossiping algorithm. While in the classical gossip algorithm each node forwards a message with the same probability, our proposal is characterized by a variable gossiping probability, which is high enough only for sustaining the spreading process towards the destination. The gossiping probability of a node is determined by the difference between its proximity to the destination and the proximity to the destination of the node from which the message was received. Differently from other proposals no external location service support, e.g., via GPS, is required. Rather, the proximity is estimated from the "inside" of the network using periodic beacons for determining the time elapsed since a node met the destination and the dwell time of a node with the destination. These information are then exploited by nodes to modulate their gossiping probability. The paper reports a mathematical model for the analysis of the algorithm along with an extensive simulation study of its implementation, which shows the suitability of the proposed solution.

Keywords: Gossip protocols; Routing; MANET; Probabilistic search

1. Introduction

Mobile Ad hoc NETworks (MANETs) are an important class of ad hoc spontaneous networks with many potential applications in the real life [7]. One fundamental primitive in a MANET is searching a path from a source node to a destination. This is the key ingredient in pure reactive routing protocols, in which a logical path from a source node to the destination is discovered on-demand, e.g., DYMO [4], AODV [13] as well as in hybrid protocols, e.g., SHARP [14].

A widely used algorithm for on-demand path discovery is flooding the whole network with route request control packets: each node simply forwards the packet when received for the first time and discards it otherwise. Under an ideal MAC (no collision) such a protocol guarantees that a path is discovered.

Despite many optimizations are possible to reduce the cost of a search of this kind [19] and to alleviate the negative effects of the so called

---

☆ A preliminary version of this work appeared in the ICDCS's Workshop on Wireless Ad hoc and Sensor Networks (WWASN 2006), July 4 2006, Lisboa, Portugal.

* Tel.: +39 06 49918502.
E-mail address: beraldi@dis.uniroma1.it

doi:10.1016/j.adhoc.2006.08.002

broadcast storm problem [18], such an algorithm has two main intrinsic limits to its efficiency. First, message request control packets are spread uniformly all over the network, thus touching regions where the destination cannot be located. Second, a node receives the same message a number of times equal to the number of neighbors; since at routing layer the message is processed just once this wastes resources.

Probabilistic algorithms can be used to address these points. Gossip protocols are a class of simple probabilistic protocols in which nodes forward the message with probability p. Krishnamachari et al. have studied the behavior of uniform gossip in random geometric graph, a widely accepted model for MANETs, see [10]. They have shown that it there exists a critical probability p_c such that for a gossiping probability $p < p_c$ the gossip process quickly dies out, while for p higher than p_c the whole network is covered. This transition phase phenomenon has been first observed in the scope of percolation theory, introduced in 1957 by Broadbent and Hammersley [3]. Experimental results reported in [16] seem to indicate that when collisions are taken into account such a sharp threshold does not exit; moreover, it there exits an optimal gossiping probability that guarantees the maximum efficiency of the search. In a relatively high node density scenario, such a value is as low as 0.1, i.e., only 10% of nodes are required to propagate the message; also, due to collisions using plain flooding in the studied setting produces a severe performance degradation. This confirms gossip as a mainstream for increasing efficiency.

A first study on gossip for route discovery has been experimentally carried out by Haas et al. in [8]; they show that for large networks, a simple gossip uses up to 35% fewer messages than flooding, and that the performance of AODV routing relying on gossip-based flooding is improved even in small networks of 150 nodes. Although uniform gossip can reduce the number of redundant transmissions, it still spreads the message isotropically.

In this paper, we introduce a new kind of gossip algorithm, dubbed polarized gossip. The polarized gossip algorithm is characterized by a polarizing node, n^*, and two gossiping probabilities, p_F and p_B. The key difference with uniform gossip is that the relaying probability of a node is determined by the node from which the message is being received. More precisely, the algorithm prescribes that if a node i receives a message for the first time and from node j, it forwards the message with probability p_F if i is closer than j to the destination and with probability p_B otherwise. We explicitly note that to obtain such a behavior it is not necessary that a node knows its real current distance from n^*. Rather, nodes can simply estimate their distances with a precision high enough to assure that the message is gossiped with the nominal probabilities. To exemplify, suppose the correct relative position is known with probability ρ. Then the polarized gossip algorithm with $p_F = \rho$ and $p_B = 1 - \rho$ is straightforwardly implemented by letting a node to forward the message if it estimates to be closer than the sending node to the destination and discards it otherwise. In other words, the gossiping probability is translated into a margin on the correctness of estimations.

Informally speaking, the macro behavior we want to obtain is to let the requesting packets spreading into the network driven by two gossiping processes: one occurring into a subset of nodes, say Z, which includes nodes forming a path from the source node to the destination and characterized with a gossip probability p_F; the other propagating in the opposite general direction and having gossiping probability p_B. Our aim is set $p_F > p_c$ while for the other spurious process $p_B < p_c$; hence, only the transmissions towards the wrong direction die out while the other survive, see Fig. 1.

In order to perform efficient estimations with a sufficiently high accuracy, we exploit a very simple mechanism based on periodic beacons. Such a technique has already been applied to unicast routing in

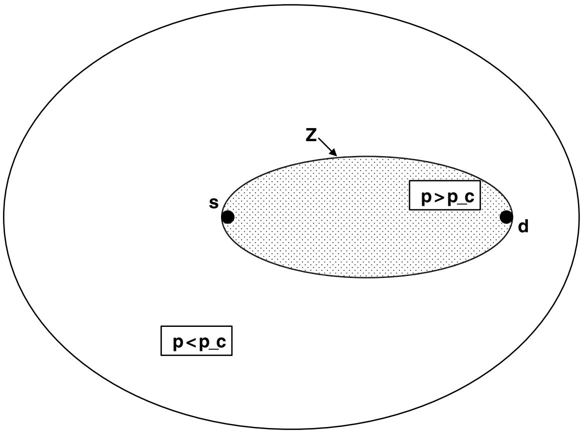

Fig. 1. A sketch of the partition of the node of a network. The forwarding probability for nodes belonging to Z is higher than the critical value p_c. The nodes outside Z forward a packet with probability lower than p_c.

[2] and to event diffusion algorithms for publish/subscribe systems in [1]. The chances of a node being close to another are measured through a positive value, dubbed a *hint*. The hint of i w.r.t. d, h_{id}, is 0 if i and d are one-hop neighbors and $\frac{\Delta T_{id}}{\tau_{id}}$ otherwise, 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 dwell time, i.e., the duration of the last wireless link established between i and d. The correlation between the T_{id} and the Euclidian distance between the two nodes has been studied empirically in [6], where it was originally dubbed the encounter age, and analytically in [2]. The distance–hint relationship has also been investigated in the same papers.

The paper is organized as follows. In Section 2 we summarize related works. Section 3 presents an analytical performance model of the gossip protocol when executed synchronously on a regular grid. Section 4 describes an implementation of the algorithm which exploits hints; simulation results are provided in Section 5 and concluding remarks in Section 6.

2. Related work

Gossip-based protocols for distributed systems have been exploited in several applications. For example, Demeres et al. developed a gossip protocol for replicated database management [5] while Luo et al. exploits gossip for reliable multicast in MANET [12]. As in other similar works, the gossip protocol assumes that a node can send a message to any other node of the network. This means that the protocol is executed above the routing layer. On the contrary our proposal leverages on the local broadcast nature of the transmissions. A node con only gossip to all neighbors via a single local broadcast.

To the best of our knowledge Haas et al. in [8] were the first that investigate how gossip can be exploited for route discovery in MANET. They propose several heuristic to deal with the problems arising from the finite network and incorporate a gossip-based search protocol in AODV. The main result is that for large networks, a simple gossiping uses up to 35% fewer messages than flooding, and that the performance of AODV routing relying on gossip-based flooding is improved even in small networks of 150 nodes. However, no attempt to provide directionality to the search was made.

Sasson et al. have studied probabilistic flooding in ideal and real networks [16]. The main outcome is that collisions seem to interfere with the predicted bimodal behavior of gossip when an ideal MAC is used.

Our approach is comparable to regional gossip as proposed by Li et al. in [11]. Regional gossip allows nodes within an ellipsoid region to forward a packet with some probability. However, a location service, e.g., based on GPS, must be available. On the contrary our algorithm does not require any additional external location support (e.g., satellites); rather it is designed for detecting Z from the inside of the network instead that from the outside.

The estimation technique has its root in the paper from Vetterli et al. [6]. The paper also describes a deterministic algorithm for path discovery which steers the search in the general direction of the destination. In our proposal the encounter age is used as the numerator for computing a hint.

Finally, our approach is somehow related to the ABR routing protocol [17]. 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 dwell time, namely τ_{ij}.

3. Polarized gossip on a finite grid

In this section we analyze a synchronous version of the polarized gossip algorithm running on a square grid topology. Such a particular topology can be considered as a snapshot of a MANET composed of mobile devices moving along streets of a city running east–west and north–south (Manhattan-like city model); the snapshot is taken when the position of any device is approximately given by the intersection of two streets. Each node of our network communicates directly with only the north, east, west and south nodes, i.e., the grid has connectivity degree four, see Fig. 2. Despite the particular topology considered and the simplification introduced, the presented model provides us

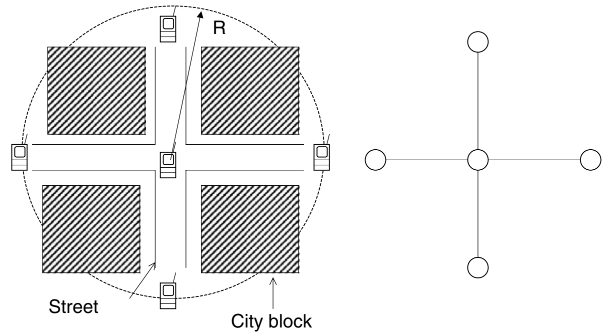

Fig. 2. The Manhattan-like city model (left) and the corresponding topology graph (right); R is the transmission range.

with a first flavor about the main behavior of the algorithm.

The distance between two nodes, say (x_1, y_1) and (x_2, y_2) is the Manhattan distance metric $|x_1 - x_2| + |y_1 - y_2|$. We will denote with dist(i,j) the distance of the node (i,j) from a particular node, called the source node and with NL(i,j) the set of neighbors of (i,j).

We start by considering a uniform gossip algorithm with broadcast implemented by independent unicast transmissions, referred to as *Gossip*(p) (we will later in this section describe a simple expedient for limiting the error introduced assuming independency); we then modify such a model in order to capture the polarization.

Gossip(p) works as follows. During the first time slot, say $k = 1$, the source sends the message and its four neighbors receive it. Each of such nodes then forwards the message to a neighbor during the next time slot, $k = 2$, with a fixed gossiping probability p. The newly reached nodes in turn forward the message during time slot $k = 3$, and so on. In general, with probability p each node re-sends the message to a neighbor when it has been received for the first time.

Let p_{ij}^k be the probability that (i,j) sends the message during time slot k. We have $p_{ij}^1 = 1$ if (i,j) is the source node and 0 otherwise. At time k a node sends the message with probability p provided that the message was received for the first time at time $k - 1$; hence, for $k > 1$ we can write the following recurrent equation:

$$p_{ij}^k = c_{ij}^{k-2}\left[1 - \prod_{(x,y)\in \mathrm{NL}(i,j)} (1 - p_{xy}^{k-1})\right]p,$$

where c_{ij}^k denotes the probability that (i,j) does not receive the message during the first k time slots and the term inside the square brackets the probability that at least a neighbour node sends the message during time slot $k - 1$.

Since a node can send a message at least once, the probabilities $p_{ij}^{k_1}$ and $p_{ij}^{k_2}$, for any $k_1 \neq k_2$, correspond to mutually exclusive events. Thus, for $k \geqslant 1$ we can write:

$$c_{ij}^k = \prod_{(x,y)\in \mathrm{NL}(i,j)} \left(1 - \sum_{s=1,\ldots,k} p_{xy}^s\right).$$

To illustrate how the above model can be useful for analyzing the case when the independency assumption is removed, we observe that (i,j) can receive a message not prior to time $k = \mathrm{dist}(i,j)$, i.e., when the message follows the shortest path connecting the source to the node. This is the first chance for observing such a node receiving the message. In particular, the neighbors of the source have just one chance at $k = 1$ and, indeed, they do receive the message with probability one. The nodes at distance 2 get their first chance at time 2; the second chance is however at time 4. In fact, a node at distance 2, say D, does not receive the message only when the node connecting D to the source, say n_1, has not resent the message received from the source, see Fig. 3. Then, the message can only arrive to D along a path of distance 4. Similarly the next chance is at time 6 (in this case, the node n_2 of the figure does not send the message). It should be clear that such a restriction has a domino effect on all nodes, so that a node at distance d can receive the message only at times $d + 2(c - 1)$, for $c = 1, 2, \ldots$

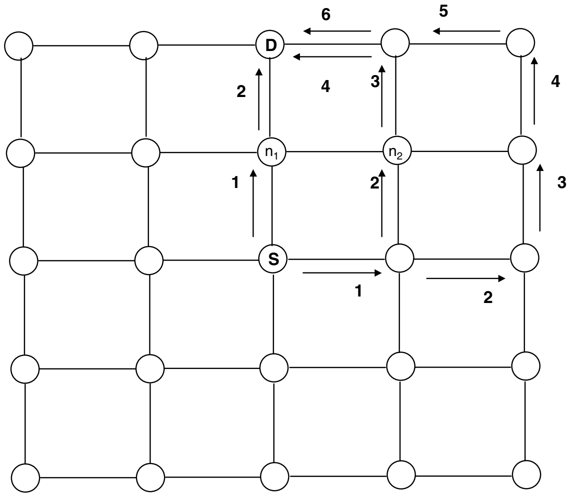

Fig. 3. An example of message propagation.

This observation is exploited for defining the probability that (i,j) receives the message within c chances, given by:

$$\text{phit}_{ij}(c) = 1 - \prod_{(x,y)\in \text{NL}(i,j)} \left(1 - \sum_{k=0}^{c-1} p_{xy}^{\text{dist}(x,y)+2k}\right). \qquad (1)$$

If broadcast is implemented by a single transmission, the probability a node discards a message received for the first time is $1-p$; on the contrary, when independency is assumed such a probability is lower since it is given by $(1-p)^n$, where $n > 1$ is the number of the neighbors of the node. Thus, the hitting time computed by Eq. (1) considering all the chances ($c \to \infty$) is an upper bound of the actual hitting probability. However, we argue that while the error caused by independent transmissions increases with c, it is very likely for a node to receive the message within the first few chances. Thus, we can try to obtain a good upper bound approximation of the exact value by using Eq. (1) with c small.

To validate such a hypothesis, the source and the target nodes were placed at different positions in a 30×30 grid. Fig. 4 reports the hit probability for two representative cases, i.e., when the nodes were placed on the main diagonal or on a line. Note that the overestimation of hitting the target at the first chance ($c = 1$) – and consequently during subsequent chances – is much higher for the diagonal case since many independent paths connecting the two nodes exist; on the contrary, when the nodes are on a line just one shortest path exists. We can observe how the hitting time is always upper bounded by the probability computed with $c = 3$ and that for the diagonal deployment the upper bound is already obtained with $c = 1$.

We can now analyze the polarized gossip protocol by slightly modifying the above model. We consider a Polarized Gossip algorithm, dubbed *PGossip*(p_{F},p_G), in which a node sends the message with probability p_{F} if the message has been received at least from a node farther than itself to the destination, and with probability $p_{\text{B}} < p_{\text{F}}$ otherwise. Clearly, *Gossip*(p) = *PGossip*(p,p). Let $F(i,j)$ be the set of neighbors that are farther than (i,j) to the destination and $B(i,j)$ the others. Suppose that (i,j) receives the message at time $k-1$ for the first time. The node will then forward the message with probability p_{F}, if it was sent from at least a node belonging to $F(i,j)$ and with probability p_{B} otherwise.

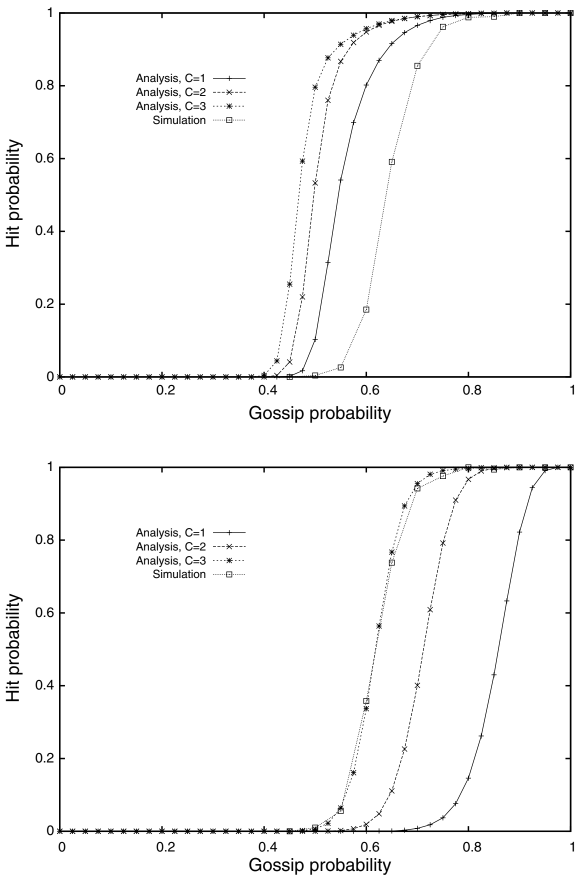

Fig. 4. Gossip performance, analysis versus simulation 30×30 grid. Top: nodes placed along a diagonal, source node at $(3,3)$, destination at $(26,26)$; bottom: nodes placed on a horizontal line, source node at $(3,15)$, destination at $(26,15)$.

Let the probability that none of the nodes in $F(i,j)$ send the message during time slot k be denoted as f_{ij}^k. We can write:

$$f_{ij}^k = \prod_{(x,y)\in F(i,j)} (1 - p_{xy}^k).$$

Similarly, the probability that none of the nodes in $B(i,j)$ send the message is given by:

$$b_{ij}^k = \prod_{(x,y)\in B(i,j)} (1 - p_{xy}^k).$$

Hence for the polarized gossip algorithm we can write, for $k > 1$, the following recurrent equation:

$$p_{ij}^k = c_{ij}^{k-2}((1 - f_{ij}^{k-1})p_{\text{F}} + f_{ij}^{k-1}(1 - b_{ij}^{k-1})p_{\text{B}}).$$

Fig. 5 shows $\text{phit}_{ij}(3)$ calculated for a 30×30 grid under *Gossip*(0.7), *PGossip*(0.7, 0) and *PGossip*(0.7, 0.30). In Fig. 5(a)–(c) gossiping is initiated by the node $(15,15)$, while the destination is at position

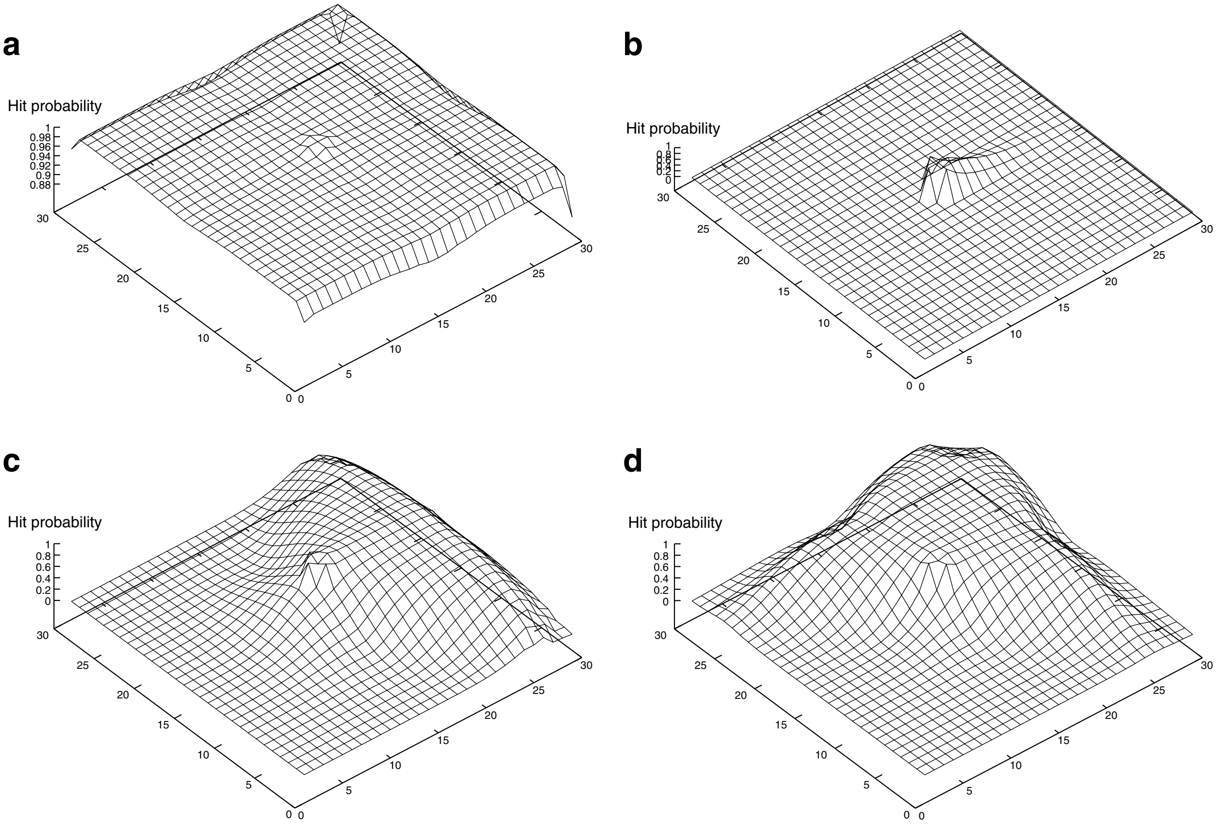

Fig. 5. An example of gossip polarization, source node placed at (15, 15): (a) Gossip (0.7), (b) PGossip (0.7,0) target at (15, 26), (c) PGossip (0.7, 0.3) target at (15, 26), (d) PGossip (0.7, 0.3) target at (26, 26).

(26, 15). We observe that under uniform gossip all nodes receive the message with "high probability" (the hit probability was about 0.98) or, equivalently, that the gossip process spreads uniformly across the grid. *PGossip*(0.7, 0) hits the destination with a probability as low as 0.028. This setting is particularly unfavorable as only the nodes along the strait line from the source to the destination send the message; thus, the message reaches the destination with probability $(0.7)^{10}$. However, when setting $p_{\mathrm{F}} = 0.7$ and $p_{\mathrm{B}} = 0.3$ the situation changes radically; the hit probability grows to more than 0.9 while backwards message spreading does not occur. Nodes far away from the destination do not receive any message. The backward probability servers as a support to message spreading towards the target, while backward message propagation quickly dies out. The result of another representative experiment is reported in Fig. 5(d). The polarizing node is now placed close to a corner of the grid, namely at (26, 26). The directionality in the spreading process is again visible.

4. Implementing gossip polarization via estimations

This section describes how the Polarized Gossip algorithm can conveniently be implemented by exploiting simple estimations about the relative positions between nodes. The proposed protocol assumes that any node j is able to estimate its current distance of itself from the destination, say $\widetilde{E_j}$ which is attached to the message before its retransmission.

The general forwarding logic of the protocol works as follows (a more detailed description is given in the next section). When a node i receives the message from j and for the first time it compares its own current estimate, $\widetilde{E_i}$, with the one carried into the message; then it decides to send the message with probability p_{F} if $\widetilde{E_i} < \widetilde{E_j}$ and with probability $p_{\mathrm{b}} < p_{\mathrm{f}}$ otherwise.

To elaborate upon the effectiveness of this implementation, suppose that the confidence of estimations is such that the true relative position is known in ρ percentage of the cases, i.e.,

$$Pr\left\{E_i < E_j | \widetilde{E_i} < \widetilde{E_j}\right\} = \rho > 0.5.$$

When a node receives the message from another node farther than itself from the destination, the message is forwarded with probability $p_{\mathrm{F}} = \rho p_{\mathrm{f}} + (1-\rho)p_{\mathrm{b}}$ (the node makes the correct estimation so that it decides to send the message with probability p_{f} or it fails in making the estimation so that decides to send the message with probability p_{b}). Similarly, $p_{\mathrm{B}} = \rho p_{\mathrm{b}} + (1-\rho)p_{\mathrm{f}}$. After some simple manipulation we can see that the difference between the forwarding and backward gossiping probabilities is $p_{\mathrm{F}} - p_{\mathrm{B}} = (p_{\mathrm{f}} - p_{\mathrm{b}})(2\rho - 1)$. Thus, the difference increases with ρ and with the difference between p_{f} and p_{b}. To provide some useful concrete example, in Fig. 6 we have reported the hit probability as a function of p_{b} for different values of ρ when $p_{\mathrm{f}} = 1$. The sample scenario is similar to the one described in the previous section. Nodes are deployed on a 30×30 grid, the source node occupies the center of the grid while the target node is $(26, 15)$. We can see that for any p_{b} increasing ρ always provides a higher hit probability. Thus, as expected setting p_{b} according to the worst case (i.e., the lowest value of ρ which characterizes the estimation technique) is a conservative solution. We can observe how even a modest value of p_{b} is able to increase the hit probability significantly. On the other hand, since $p_{\mathrm{b}} \ll p_{\mathrm{c}}$ directionality is guaranteed, i.e., message spreading in the wrong direction dies out.

4.1. Estimating distance via hints

To estimate the distance of a node from the destination we follow the same technique proposed in [2]. The key idea is the notion of a *hint*. The 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. A hint 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 dwell time, i.e., duration of the last wireless link established between i and d, see Fig. 7.

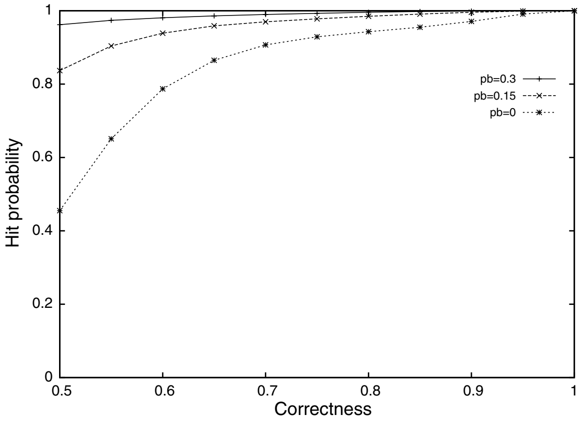

Fig. 6. Hit probability as a function of estimation correctness.

Hints are computed as follows. Each node i sends a heartbeat packet every ΔTs and uses a vector of time information, VH_i, to store relevant time information for other potential destinations. In particular, the entry for a destination j, $\mathrm{VH}_i[j]$, stores: $\mathrm{VH}_i[j].t_{\mathrm{start}}$, the time when the first heartbeat from j was detected; $\mathrm{VH}_i[j].t_{\mathrm{brk}}$, the time when the link with j was detected to be broken (this value is 0 if j is currently a neighbor); $\mathrm{VH}_i[j].t_{\mathrm{last}}$, the time of the last heartbeat received from j; and $\mathrm{VH}_i[j].\tau$, 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 $\mathrm{VH}_i[j].t_{\mathrm{last}} = t$; moreover, if $\mathrm{VH}_i[j].t_{\mathrm{brk}} \neq 0$, then it sets $\mathrm{VH}_i[j].t_{\mathrm{brk}} = 0$ and $\mathrm{VH}_i[j].t_{\mathrm{start}} = t$. To remove entries a soft state approach is adopted. When i has not received the heartbeat from j since $\alpha\Delta T$ time units ($\alpha > 1$ is a real number) it sets $\mathrm{VH}_i[j].\tau = \mathrm{VH}_i[j].t_{\mathrm{last}} - \mathrm{VH}_i[j].t_{\mathrm{start}}$ and $\mathrm{VH}_i[j].t_{\mathrm{brk}} = \mathrm{VH}_i[j].t_{\mathrm{last}}$. The hint at time t computed by the node j for the destination d is:

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

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

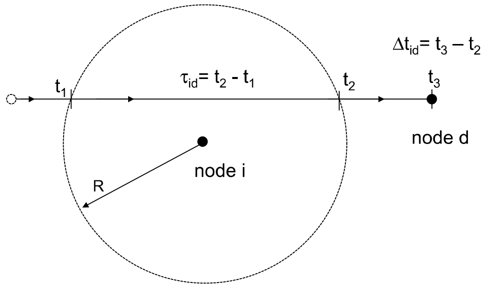

Fig. 7. Meaning of time information.

It has been empirically shown and supported by analytical results that a positive correlation between the hint the distance exits, see [2]. More specifically, let E be Euclidian distance between two nodes and h their hint; the conditional expected distance given a hint h, $\mathbf{E}[E|h]$, increases as the hint increases (this holds until a given value, say h^*, after which it remains unchanged). Formally, for any h_1 and h_2 such that $h_2 < h^*$ we have $h_1 < h_2 \Rightarrow \mathbf{E}[E|h_1] < \mathbf{E}[E|h_2]$. Then, as long as hints are sufficiently low, they can be used as a surrogate of the distance to estimate the relative position of two nodes w.r.t. the destination. Such an observation provides us with the key to understand the following implementation of the polarized algorithm.

4.2. Protocol description

In this section we describe the polarized gossip protocol which exploits hints computed as detailed above. We propose two variants. In the first case, only the beacon is sent. We call this protocol biased gossip with *No – Lookahead*. In the second variant the beacon sent by a node i piggybacks all hints the node computes for all the other nodes, i.e., the values h_{ij} for any $j \neq i$. In this case the receiving node stores the hints into a Hint Table, whose entries have lifetime $\alpha\Delta T$s (for a discussion of the efficiency of this propagation mechanism see [2]). We call this protocol biased gossip with *Lookahead*.

The pseudo-code of the biased gossip protocol is reported in Fig. 8. The protocol assumes that the request packet carries two main information: the local hint computed by the sending node (Hint), and the path accumulated in the packet so far (Path), in a way similar to DSR [9]. If the receiving node has never seen the packet, then it determines the best hint (*hint*) for the destination as well as its own current local hint (*local_hint*). If no lookahead is used then such hints are the same and a zero hint means that the destination is a one-hop neighbor; in this case the node replies back to the path's requesting node. If Lookahead is enabled these two hints are in general different; also the destination is at one hop when *local_hint* = 0 (the node then adds its own id to the accumulated path) and at two-hops if *local_hint* ≠ 0 and *hint* = 0 (in this case the id of the neighbor node which advises the zero hint is also added to the path). When a node has no hint for a destination it sends the packet with probability one.

4.2.1. A remark on hint availability and hint dissemination cost

The presented implementation of the polarized gossip algorithm is best suited for very dynamic settings, where the path lifetime is expected to be short,

```
Procedure Forward($pkt,p$) / * Behavior of node i, pkt packet to forward, p background gossip probability * /
1.if ($pkt.id$) already seen Discard($pkt$); return;
2. Let $hint$ be the best hint seen for $pkt.dest$ by the node and $local\_hint$ the hint computed by the node.
3.if ($hint == 0$) then reply back to the source;
4.if (($0 < hint < \infty) \wedge (hint < pkt.Hint)) \vee ((hint == \infty) \wedge (pkt.Hint == \infty)$)
        $pkt.Hint = local\_hint; pkt.Path[pkt.Hop++] = i;$
        bcast($pkt$)
fi;
5.if ($0 < hint < \infty) \wedge (hint > pkt.Hint$)
        with probability $p$ do
        $pkt.Hint = local\_hint; pkt.Path[pkt.Hop++] = i$
        bcast(pkt)
fi;
6. Discard $pkt$; return ;
End.
```

Fig. 8. Pseudocode of the hint-based biased gossip protocol.

say in the order of 10 s, see [15], and a reasonable fraction of nodes meet the destination. This happens when nodes move independently from each other and without restrictions (i.e., according to the hypothesis of the classical random waypoint mobility model). As an example, these assumptions could be satisfied when setting up an area for a first aid operation, where nodes represent car with radio-equipped portables driving (very) quickly from one place to another. As far as the hint dissemination cost is concerned, when lookahead is used a node can use an array whose indexes represent destinations. In a network with N nodes, the array is at most $(N-1)c_h$ bits in size, where c_h is the number of bits required to code a single hint. For example, with 300 nodes and $c_h = 4$ bits, this overhead is less than 150 bytes.

5. Performance study

To asses the performance of the biased protocol we used a custom discrete event simulator, already adopted in [2]. The simulator has the following main characteristics. Packet transmissions are governed by an ideal scheduler: the transmission of a new packet is initiated if the channel is sensed free for a random assessment delay (RAD) randomly chosen in the range $[0,\ldots,T]$s and a packet reception event is notified to a sender's neighbor provided that it remained for the whole duration of the transmission within the sender's transmission range and such that no collisions with other transmissions occurred in the meanwhile. A FIFO buffer of 20 packets in size is used at each node, packets are 1024 bytes in length, the transmission bandwidth is 11 Mbps and the transmission radius 250 m.

We have considered 300 nodes moving into a square shaped region of edge 2.5 km according to the Random Waypoint mobility model with zero pause time and speed chosen uniformly at random in the range $[1,\ldots,v]$ m/s. We considered three scenarios: low mobility ($v = 10$ m/s), medium mobility (20 m/s) and high mobility (30 m/s).

As far as the traffic is concerned, one third of node issue a path request for a given node every 10 s. The beacon interval was set $\Delta T = 500$ s, while $\alpha = 1.2$ (see Section 4.1 for a definition of α).

The following metrics were estimated during a simulation:

- Path Found, ratio of the number of path found to the number of path requests;
- Bcast, average number of broadcast packets sent by a node for path discovered (hint dissemination is not included, this cost is discussed separately in the text);
- Coverage, average number of nodes that received at least one request packet.

The statistical data were collected for 1500 s after a warm-up period of 500 s. The cases when the source sees a zero hint for the destination are *not considered*. The transient behavior as well as the hint dissemination cost are discussed separately in the text.

5.1. Numerical results

5.1.1. Effect of the MAC performance

Collision control in broadcast transmissions is normally performed by letting a node waiting for a random delay, picked uniformly at random in $[0,\ldots,T]$. The longer the time interval the lower the probability that a collision occurs. Before analyzing the performance of our protocol we have simulated a uniform gossip protocol for quantifying the impact of collisions on the general behavior of a probabilistic protocol. The results obtained for $T = 10$ ms and $T = 1$ s have been reported in Fig. 9. For a low waiting time, increasing the gossip probability beyond an optimal value has the effect of increasing the number of collisions, which is equivalent to a reduction in the chances of packets being correctly received. This is consistent with the results given in [16], where a gossip protocol using a detailed model of the 802.11 MAC protocol is simulated. By increasing the waiting time T the number of collisions are reduced; the percentage of covered nodes now resembles the typical bimodal behavior. This results are consistent with ones reported in [8] for an ideal MAC (no collision).

In the rest of the experiments we deliberately set the waiting time to $T = 1$ s in order to reduce collisions. Our aim is in fact to understand the performance of the protocol minimizing the interference with a specific MAC. This is also justified by noting that the increase in the transmission bandwidth boosted by the technological progress has the effect of reducing the period of vulnerability of a packet and thus the chances of collisions.

5.1.2. Steady state performance

Performance results of the biased protocol with no lookahead are reported in Fig. 10 for different

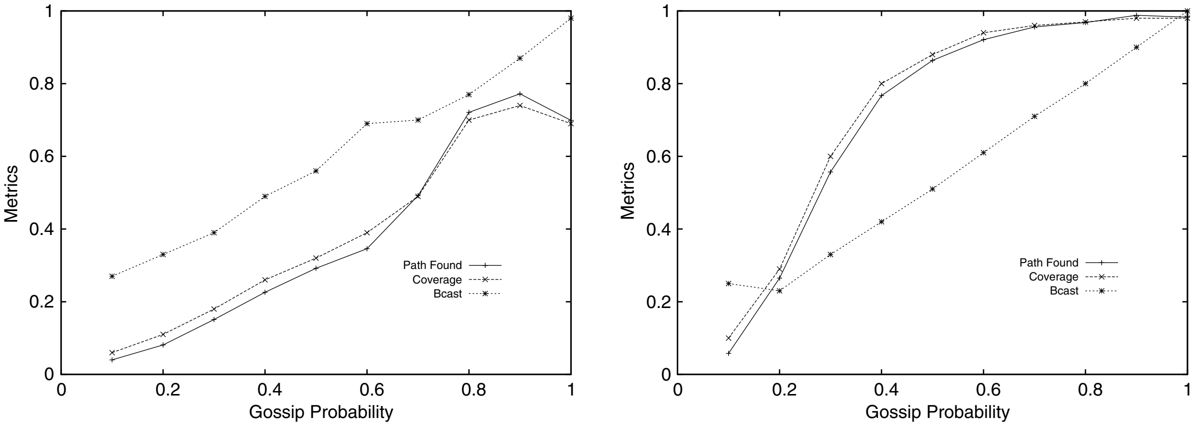

Fig. 9. Uniform gossip performance, maximum waiting time $T = 10$ ms (left) and $T = 1000$ ms (right).

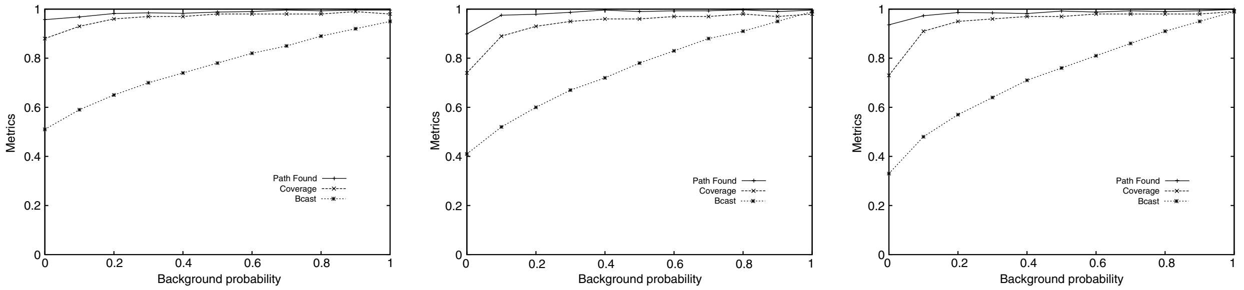

Fig. 10. Performance of the biased gossip protocol without lookahead: maximum speed 10 m/s (left), 20 m/s (center) and 30 m/s (right).

values of mobility. The duration of the validity of the hint-distance correlation decreases as the speed is increased. On the other hand, a high mobility degree allows node to come frequently in contact with each other, thus producing valid hints. This explain why the probability to find a path is higher for the low and high mobility scenarios compared to the middle scenario. Note that the normalized number of transmissions (which is an indicator of the protocol's directionality) always decreases as the speed is increased, since nodes have more chances to come in contact with each other (recall that if a node has never seen the destination then it sends the packet with probability one). As expected, regardless the mobility degree all metrics increase as the background probability increases. Moreover, the probability to find a path is increased at the cost of a higher number of transmission and coverage.

The performance results of the biased protocol with lookahead are reported in Fig. 11. This figure

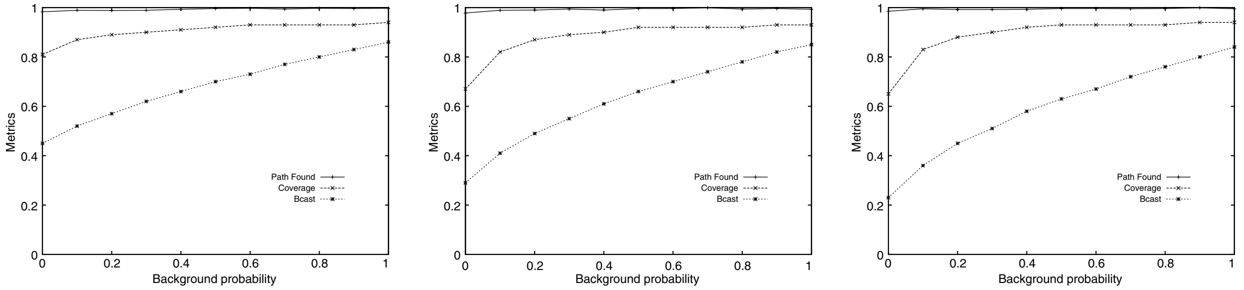

Fig. 11. Performance of the biased gossip protocol with lookahead: maximum speed 10 m/s (left), 20 m/s (center) and 30 m/s (right).

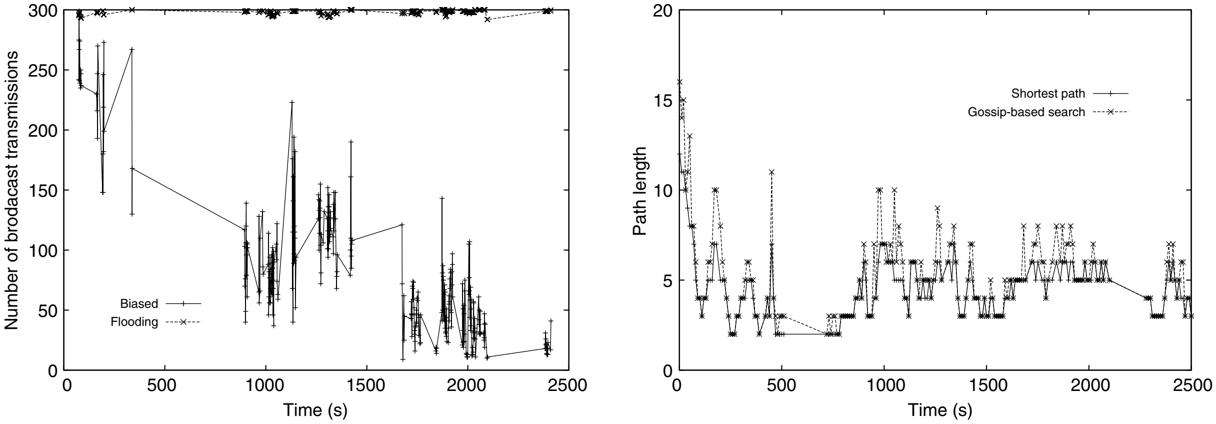

Fig. 12. A simulation trace of the searching cost for paths of length 6 hops (left) and of the path length (right).

shows how lookahead improves the performance. Good search performances are here obtained also with $p = 0$: 723 paths over 734 requests were found at the cost of about 20% of nodes sending the request in the high mobility scenario, while about 60% of nodes have to process a requesting packet.

As far as the path length is concerned, we have observed that is was only slightly higher (≈10%) than the topological distance computed before the search is started.

5.1.3. Transient behavior

To investigate the transient behavior of the protocol we have analyzed a trace of the number of broadcast transmissions per path request, which is generated by the biased protocol without lookahead in the low mobility scenario when the source is at 6 hops from the destination, see Fig. 12. Since hints are initially set to ∞ at the beginning of the simulation all nodes send the requesting packet. As time passes, nodes meet the destination thus gaining a hint; this reduces the average number of retransmissions for subsequent requests. Finally, the right side graph in Fig. 12 shows a trace of the length of the discovered path as well as the shortest one as a function of time. As the time passes the difference between these two lengths decreases; this is again an effect of having more valid hints available. The cases when the distance is one hop are not reported in the trace.

5.1.4. Hint dissemination cost

The broadcast transmissions required to disseminate hints in the network is a fixed cost which is shared among the actual nodes issuing a path request. The cost per path request can then be expressed as b/r, where b denotes the beacon rate and r the average route request rate. Fig. 13 reports such a cost as a function of r and b given as a parameter. The cost is clearly well amortized when a suitable number of route requests is issued in the network, i.e., r is reasonable high. And, this is likely to happen in the dynamic environment for which the implementation is tailored to. For example, in a network with the size considered in the experiments (300 nodes) and by assuming an average path lifetime of 10 s [15], a classical client/server application with one third of nodes acting as a client will generate 10 route requests per second. Even with a beacon rate $b = 2$, this extra overhead is roughly 0.2 transmissions per node per path request.

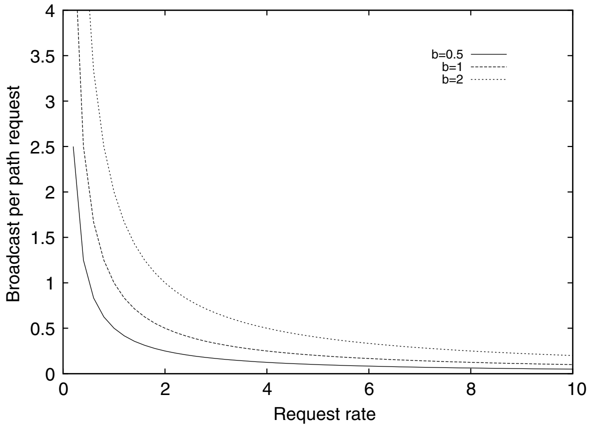

Fig. 13. Hint dissemination overhead.

6. Conclusion

This paper introduces a new class of gossip algorithms designed to direct the gossip process towards a specific polarizing node in the network. The algorithm has been dubbed Polarized Gossip. While in the classical gossip algorithm each node forwards a message with the same probability, our proposal is characterized by a variable gossiping probability, which is high enough only for sustaining the spreading process towards the polarizing node. In the special case of regular 2-D grid, the performance of the algorithm has been analyzed through a mathematical model.

An important application of the algorithm is when used for path discovery in MANET. In this case the polarizing node corresponds to a destination node. The paper then reports an implementation of the algorithm which is tailored for path discovery. The distinguishing feature of the implementation is the use of beacons as a mean to approximately localize the destination.

We have shown by simulations the protocol allows to save up to 80% of broadcast transmissions compared to a pure flooding, while 60% of nodes have to process a requesting packet.

There is room for many optimizations and extensions. For example, other techniques for determining approximately the relative position of two nodes w.r.t. the polarizing node can be investigated. The case of many idempotent polarizing nodes can be useful to apply the algorithm to service discovery.

Acknowledgement

The work described in this paper was partially supported by the Italian Ministry of Education, University and Research (MIUR), under the “IS-MANET” project.

References

[1] R. Baldoni, R. Beraldi, G. Cugola, M. Migliavacca, L. Querzoni, Structure-less Content-Based Routing in Mobile Ad Hoc Networks, in: Proc. IEEE Int. Conf. on Pervasive Services 2005 (ICPS’05), Santorini, Greece (2005) 11–14 July.

[2] R. Beraldi, L. Querzoni, R. Baldoni, A hint-based probabilistic protocol for unicast communications in MANETs, Ad Hoc Networks 4 (2006) 547–566.

[3] S.R. Broadbent, J.M Hammersley, Percolation processes In crystals and mazes, in: Proc. of the Cambridge Philosophical Society (1957) vol. 53, p. 629641.

[4] I.D. Chakeres, Elizabeth M. Royer, Charles E. Perkins, Dynamic MANET On-demand Routing Protocol. IETF Internet Draft, draft-ietf-manet-dymo-02.txt, June 2005, work in progress.

[5] A. Demers et al., Epidemic algorithm for replicated database management, in: Proc. of the sixth annual ACM Symp. on Principles of distributed computing (1987) British Columbia, Canada.

[6] H. Dubois-Ferriere, M. Grossglauser, M. Vetterli, Age matters: efficient route discovery in mobile ad hoc networks using encounter ages, in: Proc. of the 4th ACM Int. Symp. on Mobile ad hoc networking computing.

[7] M. Gerla, From battlefields to urban grids: New research challenges in ad hoc wireless networks, Elsevier Pervasive Mobile Comput. 1 (1) (2005), March.

[8] Z. Haas, J. Halpern, L. Li, Gossip-based ad hoc routing, in: Proc. of IEEE INFOCOM, 2002.

[9] D.B. Johnson, D.A. Maltz, Dynamic Source Routing in Ad Hoc Wireless Networking, in: T. Iemielinski, H. Korth (Eds.), Mobile Computing, Kluwer Academic, 1996 (Chapter 5).

[10] B. Krishnamachari, S.B. Wicker, R.B., M. Pearlman, Critical Density Thresholds in Distributed Wireless Networks, in: Proc. of Communications, Informations and Network Security, H. Bhargava, H.Poor, V. Tarokh, S. Yoon Eds., Kluwer Press, December, 2002.

[11] X.-Y. Li, K. Moaveninejad, O. Frieder, Regional Gossip Routing for Wireless Ad Hoc Networks, Mobile Networks and Applications (MONET), volume 10, February, 2005, pp. 61–77.

[12] J. Luo, P. Th. Eugster, J.-P. Hubaux, Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks, in: Proc. of INFOCOM, 2003.

[13] C.E. Perkins, E.M. Royer, S.R. Das, Ad-hoc On Demand Distance Vecto (AODV) Routing, draft-ietf-manet-aodv-06.txt (work in progress), July 2000.

[14] V. Ramasubramanian, Z.J. Haas, E. Sirer, SHARP: a hybrid adaptive routing protocol for mobile ad hoc networks, in: Proc. of the 4th ACM Int. Symp. on Mobile ad hoc networking and computing (MobiHoc, 2003), Annapolis, Maryland, USA June 01–03, 2003.

[15] N. Sadagopan, F. Bai, B. Krishnamachari, A. Helmy, PATHS: Analysis of PATH Duration Statistics and their Impact on Reactive MANET Routing Protocols, in: Proc. of MobiHoc (2003).

[16] Y. Sasson, D. Cavin, Andre Schiper, Probabilistic broadcast for flooding in wireless mobile ad hoc networks, in: Proc. of IEEE Wireless Communications and Networking Conference (WCNC 2003), March 2003.

[17] C.-K.G Toh, A Novel Distributed Routing Protocol To Support Ad hoc Mobile Computing, in: Proc. IEEE 15th Annual Int. Phoenix Conf. on Computers and Communications, IEEE IPCCC (1996) March 27–29, Phoenix, AZ, USA, pp. 480–486.

[18] S. Ni, Y. Tseng, Y. Chen, J. Sheu, The broadcast storm problem in a mobile ad hoc network, in: Proc. of ACM MOBICOM’99, August 1999.

[19] B. Williams, T. Camp, Comparison of broadcasting techniques for mobile ad hoc networks, in: Proc. of the ACM Int. Symp. on Mobile Ad Hoc Networking and Computing (MOBIHOC) 2002, pp. 194205.

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 University "La Sapienza", Rome. His research interests include distributed systems and dynamic networks.