# Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks[†]

Jun Luo Patrick Th. Eugster Jean-Pierre Hubaux
School of Computer and Communication Sciences
Swiss Federal Institute of Technology (EPFL), CH-1015 Lausanne, Switzerland
Email: {jun.luo, patrick.eugster, jean-pierre.hubaux}@epfl.ch

*Abstract*— **Traditionally, reliable multicast protocols are deterministic in nature. It is precisely this determinism that tends to become their limiting factor when aiming at reliability and scalability, particularly in highly dynamic networks, e.g., ad hoc networks. As probabilistic protocols, gossip-based multicast protocols, recently (re-)discovered in wired networks, appear to be a viable means to "fight fire with fire" by exploiting the non-deterministic nature of ad hoc networks.**

**This paper presents a protocol that is designed to meet a more practical specification of probabilistic reliability; this gossip-based multicast protocol, called Route Driven Gossip (RDG), can be deployed on any basic on-demand routing protocol. RDG is custom-tailored to ad hoc networks, achieving a high level of reliability without relying on any inherent multicast primitive. We illustrate our RDG protocol by layering it on top of the "bare" DSR protocol. We prove the reliability and scalability of RDG through both analysis and simulation.**

## I. Introduction

Reliable multicast protocols in *wired* networks can be roughly classified into three categories. The first category enforces strong reliability guarantees which provide "all-or-nothing" semantics for the successful delivery of a message to a group of nodes, tolerating the failure of a certain number of these nodes (cf. *Reliable Broadcast* [1]). Unfortunately, protocols belonging to this category scale poorly with an increasing group size even in a very stable network.

The second category includes mainly protocols that indeed offer some practical reliability, but are not reliable in the metric of the above-defined category and lack an alternative measure of their reliability. In the Internet, this includes typically protocols building on top of *IP Multicast* [2], e.g., [3], [4]. The ack/nack mechanisms employed by such protocols to improve reliability, unfortunately, also tend to compromise their scalability by heavily loading the network (e.g., leading to *ack implosion*).

The third category of protocols consists of so-called *gossip-based* protocols and has been (re-)discovered rather recently. Roughly, the idea common to members of this family of *probabilistic* protocols (e.g., [5], [6], [7]) is to have each node in a multicast group periodically "talk" to a random set of other nodes in the group about its knowledge of the "state" of the group, e.g., the multicast packets that it has received. Missing packets can then be recovered by nodes in a peer-based style. These protocols equally distribute the load over the nodes in a group and thus also make themselves very resilient to arbitrary node failures. Stochastic models derived from epidemiology enable the protocols to obtain (1) the desired tradeoff between reliability and scalability by adjusting protocol parameters and (2) a performance prediction.

Intuitively, we find that it is appealing to apply a probabilistic scheme in an ad hoc network[1], precisely because the underlying network itself provides little determinisism. Nodes are not connected by any fixed infrastructure, and communication between two such nodes at a given moment might be possible directly, only indirectly, or not at all. This observation has already become the motivation for the successful design of a gossip-based routing protocol for ad hoc networks [8].

It appears that deterministic protocols to multicast in ad hoc networks suffer strongly from an amplification of the tradeoff between reliability and scalability already encountered with such protocols in wired networks. Existing (unreliable) protocols (ad hoc-analogues to IP Multicast) provide no reliability guarantees at all (e.g., [9], [10]), and proposals attempting to detect and repair failures (e.g., [11], [12]) can hardly generate any throughput when the network topology undergoes frequent changes. Finally, no protocols providing strong reliability guarantees in the sense of "all-or-nothing" semantics (the first aforementioned category of protocols) have yet been proposed due to the prohibitive complexity.

Devising a gossip-based multicast protocol for ad hoc networks is however not trivial and, in particular, cannot be straightforwardly achieved by adapting a protocol conceived for wired networks. A seminal approach is given by the *Anonymous Gossip* (AG) protocol [13], a descendant of the *pbcast* [5] protocol that pioneered the recent research efforts on gossip-based protocols for wired networks. Through the concept of anonymous gossip, any agreement on membership is avoided during the gossip-based repair phase. This however shifts the responsibility for the membership management to the *Multicast Ad hoc On Demand Distance Vector* (MAODV) layer [9], which the AG protocol also relies upon for a preliminary, rough packet dissemination. These prerequisites

[†]This work was supported (in part) by National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322. (`http://www.terminodes.org`)

[1]Both mobile ad hoc networks and wireless sensor networks are considered here.

make the AG protocol more difficult to apply in a broader context than the one offered by MAODV. Furthermore, the property of predictable behavior, an important merit of gossip-based protocols, is lost due to the dependence on MAODV to guide the gossips. Other efforts that make use of gossiping techniques for multicasting in ad hoc networks (e.g., [14], [15]) similarly lack an analytical prediction of their reliability.

As a cornerstone in the *Terminodes* [16] project, this paper presents a novel gossip-based multicast protocol for ad hoc networks, designed to meet a more practical specification of probabilistic reliability. Our *Route Driven Gossip* (RDG) protocol (1) uses a pure gossip scheme – gossiping uniformly about multicast packets, negative acknowledgements, and membership information; (2) takes into consideration parameters of the network, e.g., the availability of routing information; (3) does not require a multicast primitive at the network layer and can be deployed on any basic, virtually unmodified, on-demand routing protocol. We illustrate our RDG protocol using the "bare" DSR [17] protocol, i.e., without any multicast extension. We defend our claims of predictable reliability of the protocol by comparing results obtained through a formal analysis based on a stochastic model and results collected from an exhaustive set of simulation experiments performed with the *ns-2* network simulator [18]. The simulation results also confirm the scalability of our protocol. We do not intend either to undermine existing and indeed suggestive proposals for multicast extensions to DSR (e.g., [19]), or to claim in general that multicast protocols for ad hoc networks should be gossip-based in nature. The idea is rather to explore the feasibility of such a probabilistic approach along with a prediction of its performance in a highly dynamic setting, useful for many critical applications such as security services (e.g., distributed key management services [20], [21], certificate distribution and revocation for self-organized public-key infrastructures [22]).

The rest of this paper is organized as follows: Section II describes the network model and specifies more precisely the problem solved. Section III presents our RDG protocol. A formal analysis and simulation results of our RDG protocol are given in Sections IV and V, respectively. Section VI discusses various issues, such as optimizations and reliability metrics. Section VII surveys related work. Finally, Section VIII concludes the paper.

## II. Assumptions and Problem

Before presenting our RDG protocol, we define more accurately our network model and specify the problem solved in that model.

### A. Network Model

The network consists of a set $\mathbb{N}$ of nodes with the same computation and transmission capabilities, communicating through bidirectional wireless links between each other. A unicast routing protocol is available to support packet transmissions between the network nodes (we assume DSR in this paper). $G \subset \mathbb{N}$ is a multicast group with size $|G| = n_G$. Nodes join and leave different groups following the requirements of upper layer applications.

The following assumptions are made on the nodes:

- Every node has a unique physical address or ID.
- The transmission radius for each node is fixed.
- Nodes fail only by crashing, i.e., stopping to function. Crashes are not permanent.

In addition, we assume a CSMA/CA-like MAC layer protocol (e.g., IEEE 802.11) that provides a RTS/CTS-Data/Ack handshake sequence for each transmission.

The information unit for the protocol is the *message*. It can include data packets, as well as membership information. However, the *packet*, the unit for the network layer, is used when data logging and loss detection are carried out. Each packet multicast is uniquely identified by its identifier $pid$, a tuple [group ID, source ID, packet sequence No.], such that a member can detect missing packets by observing gaps in packet ID sequence.

### B. Problem Definition

Our goal is to design a multicast protocol for ad hoc networks, which achieves probabilistic reliability. Instead of providing perfect guarantees like "all packets sent by a source will eventually be received by all group members", we provide one that roughly states "if some group member sends out a flow of $M$ packets, a certain group member receives a fraction $\zeta$ of all $M$ packets with probability $\pi_M(\zeta)$ $(\zeta, \pi_M \in [0,1])$". Here $\zeta$ and $\pi$ are termed *reliability degree* and *reliability probability distribution*, respectively. The reliability of the protocol defined by $\pi_M(\zeta)$ is expected to be predictable given simple information like packet loss ratio, whereas the scalability requirements are such that increasing network size and mobility should only result in a modest degradation of reliability.

## III. Route Driven Gossip Protocol

This section presents in detail our Route Driven Gossip (RDG) protocol after providing related background.

### A. Overview of Dynamic Source Routing (DSR) Protocol

DSR is an on-demand routing protocol making use of source routing and an aggressive caching policy. The protocol is on-demand since it floods route requests in the network upon routing packets to a destination without an available corresponding routing path. The source routing mechanism makes the routing paths loop-free, while providing certain topological information. With the aggressive caching policy, DSR tries to cache all routing paths that it learns (it even taps such information from the MAC layer if the "promiscuous" receive mode is enabled.).

### B. Design Characteristics

Traditional gossip protocols are characterized as *view driven gossip* because the destinations of each gossip are determined by the view[2] of the membership at the source. According to

[2] *View* is a data structure to store the membership information.

our observations, a view driven protocol is unsuitable for ad hoc networks with on-demand routing (e.g., DSR and AODV), since a node cannot always have routing paths to all the nodes in its view. If each node would request the paths to its gossip destinations for each gossip task, heavy network traffic would be generated, reducing the efficiency of the protocol.

The design of our gossip-based protocol has been influenced by the following observations on ad hoc networks working with an on-demand routing protocol:

- **Routing information is precious**, because the costs to obtain such information are considerably high. In our case, the routing information for group members covers not only the routing paths but also the relationships between a certain member and its routing paths. It is possible that either there is no routing path to a known member or an existing routing path leads to a member that is unknown to the source. In order to make the best of these resources, the protocol should maintain as many relationships as possible and try to use them while they are fresh.
- **Route requests are costly**, due to the flooding nature of the route request. We can, however, benefit from this feature by requesting the routing paths to several group members with only one request message. Although the network traffic is greatly reduced in the request phase, the massive reply messages in the reply phase afterward may congest the network. Therefore, one needs to be careful in dealing with the route reply.

### C. Protocol Presentation

In order to overcome the problems with *view driven* protocols in ad hoc networks and to integrate the observations stated above, we propose a *route driven* protocol. Our Route Driven Gossip (RDG) protocol relies only on partial views for each member; these random subviews result from the randomness of routing information that nodes can have. RDG uses a *pure* gossip scheme. The spread of the information is propelled mainly by a *gossiper-push* (each group member forwards multicast packets to a random subset of the group) but complemented by a *gossiper-pull* (multicast packets piggyback negative acknowledgements of the forwarding group member).

*1) Basic Data Structures:* There is one protocol instance for each group $G$. Besides the identifier of a group (*Gid*), the following four data structures are used for the protocol:

- **Data Buffer (*Buffer*):** This buffer stores data packets received. It is divided into two parts: *Buffer.new* stores the packets to be gossiped in the future; the other packets are stored in *Buffer.old* in preparation for responding to gossiper-pulls. If the size limit of the buffer is reached, the oldest packets are removed.
- **Active View (*AView*):** This view contains the IDs of known members to which at least one routing path is known.
- **Passive View (*PView*):** Contains the IDs of known members to which no routing path is currently available.
- **Remove View (*RView*):** Contains the IDs of members that have indicated their desire to leave.

Therefore, each $node_i \in G$ has five data structures: $Gid_i^G$, $Buffer_i^G$, $AView_i^G$, $PView_i^G$, and $RView_i^G$.

*2) Operations:* Our RDG protocol offers seven operations, which are grouped into three sessions corresponding to their functionality. The *join* session defines the behavior of the node interested in joining a group and the reactions of other group members. The *leave* session defines the behavior of the node intending to leave the group and the reactions. The GOSSIP task is periodically executed by a node (if there are messages to disseminate). Furthermore, nodes react to the gossip messages received. In relation to the GOSSIP task, two design parameters are defined here: the *fanout* ($F$) is the number of gossip destinations randomly selected from the *AView* for each gossip emission; the *quiescence threshold* ($\tau_q$) is related to each data packet: a packet will be removed from *Buffer.new* after having been gossiped for $\tau_q$ times. Section IV-B.2 discusses how to set these parameters.

We extend the ROUTEREQUEST and ROUTEREPLY primitives provided by DSR for our purposes:

- GROUPREQUEST [Fig.1(a)]: This primitive extends the ROUTEREQUEST of DSR by requesting routing paths to multiple nodes at the same time. The GROUPREQUEST puts the group ID (*gid*) in the Target Address field of the DSR header. Only group members can respond to the message.
- GROUPREPLY [Fig.1(b)]: This primitive is equivalent to a ROUTEREPLY but with the *gid* attached to it, such that a node receiving such a message can distinguish it from a usual ROUTEREPLY.

The RDG protocol can be rather easily adapted to other on-demand routing protocols by accordingly implementing these primitives.

*3) Protocol Behavior:* The pseudo-codes for all the above operations are provided here, followed by detailed descriptions. The *gossip* and *leave* sessions are reported together, since the dissemination of a leave indication relies on the *gossip* session. Note that lists like *Buffer* have a maximum size, noted $|L|_m$ for a given list $L$.

*Join session (Fig. 1):*

- A node intending to join a group floods the network with a GROUPREQUEST message to search for other group members whilst announcing its existence.
- Upon receiving a GROUPREQUEST from a certain member, all members update their *AView* with the new ID. They also return a GROUPREPLY to the request initiator with probability $P_{reply}$.
- The initiator of the GROUPREQUEST also updates its *AView* after receiving the GROUPREPLY.

By recording the route of each incoming packet, DSR ensures that a new element in *AView* has a corresponding route entry in the DSR routing table. The validity of this relationship is periodically checked and the *AView* and *PView* are updated accordingly. When the size of *AView* drops below some threshold $\tau_v$, the node has to reinitiate a *join* session.

**procedure** JOIN($gid$)
  GROUPREQUEST($id_i$, $gid$)

**upon** RECEIVEGROUPREPLY($id$, $gid$) **do**
  $AView_i^{gid} \leftarrow AView_i^{gid} \cup \{id\}$

(a) *Join* indication emission and reply reception

**upon** RECEIVEGROUPREQUEST($id$, $gid$) **do**
  **for all** group $G$ that $i$ belongs to **do**
    **if** $Gid_i^G = gid$ **then**
      $AView_i^{gid} \leftarrow AView_i^{gid} \cup \{id\}$
      GROUPREPLY($id_i$, $gid$) with probability $P_{reply}$

(b) *Join* indication reception

Fig. 1. *Join* session at node $i$

*Gossip/leave session (Fig. 2 and 3):*

- Each member of the group periodically (every $T$ ms)[3] generates a gossip message and gossips it to $F$ other nodes randomly chosen from *AView*. The message includes packets stored in *Buffer.new*, and the $id$ of the most recent missing packet. It also piggybacks its view on the membership (if the node intends to leave, only the field of *rview* is valid). A data packet is removed from *Buffer.new* after having been gossiped for $\tau_q$ times.

**procedure** LEAVE($gid$)
  $leaveFlag_i^{gid} \leftarrow true$

**task** GOSSIP($gid$) {/* *Executed every $T$ ms* */}
  /* Step 1: Generate message and disseminate it */
  **if** $leaveFlag_i^{gid} = true$ **then**
    $m.rview \leftarrow id_i$
  **else**
    $m.data \leftarrow Buffer_i^{gid}.new$
    $m.gpull \leftarrow pid$ of the most recent missing packet
    /* gossiper-pull */
    $m.rview \leftarrow$ a random entry in $RView_i^{gid}$
    $m.view \leftarrow$ a random entry in $AView_i^{gid} \cup PView_i^{gid}$
  $DS \leftarrow$ random set of $F$ members in $AView$
  **for all** $id \in DS$ **do**
    SENDGOSSIP($gid$, $id_i$, $m$, $id$)
  /* Step 2: Update the data buffer */
  **if** $leaveFlag_i^{gid} = false$ **then**
    **if** $\exists\ pkt \in Buffer_i^{gid}.new$ that has been gossiped more than $\tau_q$ times **then**
      $Buffer_i^{gid}.old \leftarrow Buffer_i^{gid}.old \cup \{pkt\}$
      $Buffer_i^{gid}.new \leftarrow Buffer_i^{gid}.new \backslash \{pkt\}$

Fig. 2. *Gossip/leave* session at node $i$ – message emission

- A group member receiving a gossip message will (1) remove the obsolete member from its view, (2) add the new member to the view, (3) update the data buffer with new packets, and (4) respond to the gossiper-pull. The gossiper-pull is responded to only if the data packet requested will not be gossiped again (the request might be satisfied by the upcoming gossip).
- A packet received upon gossiper-pull is delivered if it is still missing. The data buffer is updated accordingly.

**upon** RECEIVEGOSSIP($gid$, $id_s$, $m$) **do**
  /* Step 1: Remove obsolete member from the view */
  $AView_i^{gid} \leftarrow AView_i^{gid} \backslash \{m.rview\}$
  $PView_i^{gid} \leftarrow PView_i^{gid} \backslash \{m.rview\}$
  $RView \leftarrow RView \cup \{m.rview\}$
  **while** $|RView| > |RView|_m$ **do**
    remove a random element from $RView$
  /* Step 2: Add new member to the view */
  **if** $m.view \notin (AView_i^{gid} \cup PView_i^{gid})$ **then**
    **if** there exists a route to that node **then**
      $AView_i^{gid} \leftarrow AView_i^{gid} \cup \{m.view\}$
    **else**
      $PView_i^{gid} \leftarrow PView_i^{gid} \cup \{m.view\}$
  /* Step 3: Update $Buffer$ with new packets */
  **for all** $pkt \in m.data$ **do**
    **if** $pkt \notin Buffer_i^{gid}$ **then**
      $Buffer_i^{gid}.new \leftarrow pkt$
      DELIVER($pkt$) /* to the upper layer */
  **while** $|Buffer_i^{gid}| > |Buffer_i^{gid}|_m$ **do**
    remove the oldest element from $Buffer$
  /* Step 4: Respond to the gossiper-pull */
  **if** $m.gpull \in pid$ list of $Buffer_i^{gid}.old$ **then**
    SENDGOSSIPRESPONSE($gid$, $id_i$, $pkt_{m.gpull}$, $id_s$)

**upon** RECEIVEGOSSIPRESPONSE($gid$, $id$, $pkt$) **do**
  **if** $pkt \notin Buffer_i^{gid}$ **then**
    $Buffer_i^{gid}.old \leftarrow pkt$
    DELIVER($pkt$) /* to the upper layer */

Fig. 3. *Gossip/leave* session at node $i$ – message reception

Nodes along routing paths to gossip destinations belonging to the same group as those destinations, when forwarding a packet they have not received yet, also deliver the packet and update their data buffers (not shown in the code). Due to its unpredictability, this operation will not be taken into account in the analysis in the next section, making the protocol perform better than expected.

Note that the packet salvaging function of DSR is disabled while a gossip message is on its way, i.e., packets are dropped immediately whenever the routing path becomes obsolete or the sending buffer overflows. In fact, the redundancy provided by our RDG protocol automatically offsets the packet loss.

*4) Topology-aware RDG:* The basic RDG protocol presented above can be qualified as a *brute force* protocol. It can be made aware of the network topology for improved efficiency. We call the variant TA-RDG, i.e., topology-aware RDG. The design of this variant is based on the assumption that the underlying routing protocol can provide some partial topological information, e.g., we can have the information about the path length from the routing table of DSR. The heuristics based on DSR work like this: different weights are assigned to the members in *AView* according to the length of the routing paths to them, i.e., the longer the path the lower the weight, such that a node chooses a "near" member with higher probability to gossip. A simple way to implement this

[3]In order to save bandwidth, we apply the *binary exponential backoff* algorithm to adjust the period when there is no new packet to be sent or no lost packet to be requested.

is to choose weights inversely proportional to the length of the corresponding routing paths. The locality of the traffic resulting from this optimization greatly reduces the network load and, as shown by simulations, improves the reliability in most cases.

### D. Example of Protocol Operation

We assume a single group $G$ of size $n_G = 10$ within a 20 nodes network. The matrix $V$ in Fig. 4 shows the active view of the group at each member at a give point in time. Fig. 5 gives a visual illustration of the behaviour of the protocol with respect to the dissemination of one packet. Assuming $F = 2$ and $\tau_q = 2$, the packet initiated by member 15 infects the whole group in only 3 rounds in spite of the fact that no member has a full view of the membership while nodes move and even fail.

| V | 1 | 2 | 3 | 5 | 8 | 9 | 10 | 13 | 15 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | ⋆ | ⋆ | | ⋆ | | | | | | ⋆ |
| 2 | ⋆ | ⋆ | | | ⋆ | | ⋆ | | ⋆ | ⋆ |
| 3 | | | ⋆ | ⋆ | | ⋆ | | | ⋆ | |
| 5 | ⋆ | | ⋆ | ⋆ | | ⋆ | | | ⋆ | ⋆ |
| 8 | ⋆ | ⋆ | | | ⋆ | | ⋆ | | ⋆ | |
| 9 | | | ⋆ | ⋆ | | ⋆ | ⋆ | ⋆ | ⋆ | |
| 10 | ⋆ | ⋆ | | ⋆ | ⋆ | ⋆ | ⋆ | ⋆ | ⋆ | |
| 13 | | | | | | ⋆ | ⋆ | ⋆ | ⋆ | |
| 15 | | ⋆ | ⋆ | ⋆ | | ⋆ | ⋆ | | ⋆ | |
| 19 | ⋆ | ⋆ | | ⋆ | | | | | | ⋆ |

Fig. 4. Illustration of the active view (other records like passive view are omitted). A "⋆" at $V_{ij}$ means that member $j$ is in the *AView* of member $i$.

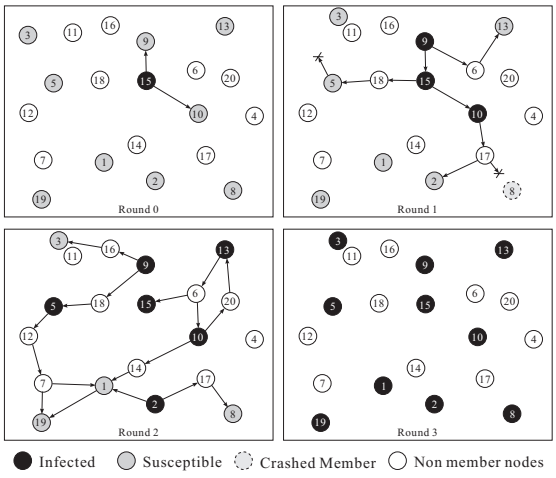


Fig. 5. An example of one "run" of the protocol with $F = 2$ and $\tau_q = 2$ within a group of size 10. A member may receive duplicates of the same packet (e.g., member 1 at round 2). On the other hand, the packet can get lost at a certain round due to nodes crashing or moving (e.g., members 8 and 3 in round 1), but these losses will be compensated with high probability at a later round.

## IV. Analysis

This section provides an analytical evaluation of our RDG protocol (however, without considering the topology-awareness, in order to simplify the tractability). The goal is to show that the reliability of the protocol is predictable given certain design parameters and information about the network. This claim is confirmed by simulations in the next section.

### A. Model

We consider a single group $G$ composed of $|G| = n_G = n$ members and observe its behavior in terms of the dissemination of a *single* packet ("one run"), but also a *continuous* flow of packets (which is more practical than related efforts considering only the "one run" part). According to the terminology of epidemiology [23], a member that has received a certain packet is termed *infected*, otherwise *susceptible*. An infected member attempting to share the packet with others (i.e., a member who keeps gossiping the packet) is called *infectious*.

We analyze our protocol in a network composed of a static set of nodes running closely synchronized. More precisely, nodes gossip in synchronous rounds ($T$ ms, identical for all nodes), and there is an upper bound on the network latency which is smaller than $T$.

The probability of packet loss is closely related to the movement and traffic pattern, as well as to the length of the considered routing path. By assuming an identical and independent probability of failure $p_f$ for each node along a routing path in a certain network environment, the probability of losing a certain gossip message can be expressed as a function of the number of hops, $H$, of that routing path. We further assume that the lengths $H$ of all routing paths between two members follow the same distribution $P(H)$. On the other hand, $p_f$ can be split into two parts: (1) $p_{f_c}$ represents the probability of packet loss due to node crash; (2) $p_{f_{mo}}$ accounts for the effects of node mobility and buffer overflow. While $p_{f_c}$ can be set according to empirical results, $p_{f_{mo}}$ is determined by the movement and traffic pattern.

In reality, the size of the *AView* for a given member may vary between $\tau_v$ and $n - 1$. However, the value could be maintained very close to $n - 1$ by assuming a low mobility network. Furthermore, we expect that the protocol can keep approximately the same view size in a high mobility network, assuming that other protocols running in parallel would infuse routing information to the nodes.

Due to its irregularity, the effects of the gossiper-pull procedure can hardly be considered in the analysis, making the present analysis a lower bound.

### B. Stochastic Behavior of Packet Dissemination

The predictable reliability of our RDG protocol is conveyed in two steps. We first show that the *single packet dissemination reliability* is predictable given certain network information, and based on the results, we discuss the *reliability probability distribution* $\pi_M(\zeta)$.

*1) Single Packet Dissemination Reliability:* Let $m$ be a message generated by a certain member. We use $s_r \in \{1, \cdots, n\}$ and $\Delta s_r = \mathbf{E}[s_r - s_{r-1}]$ to denote the number of members infected with $m$ **after** round $r$ and the average[4] number of members infected **within** round $r$, respectively. If we define the *state space* $\mathcal{E} = \{1, \cdots, n\}$, the sequence of random variables $\{s_r\}_{r \geq 0}$ forms a stochastic process with values taken from $\mathcal{E}$.

[4]Setting $\Delta s_r = s_r - s_{r-1}$ would make $\Delta s_r$ a random variable, leading to a state space unfeasible for analysis.

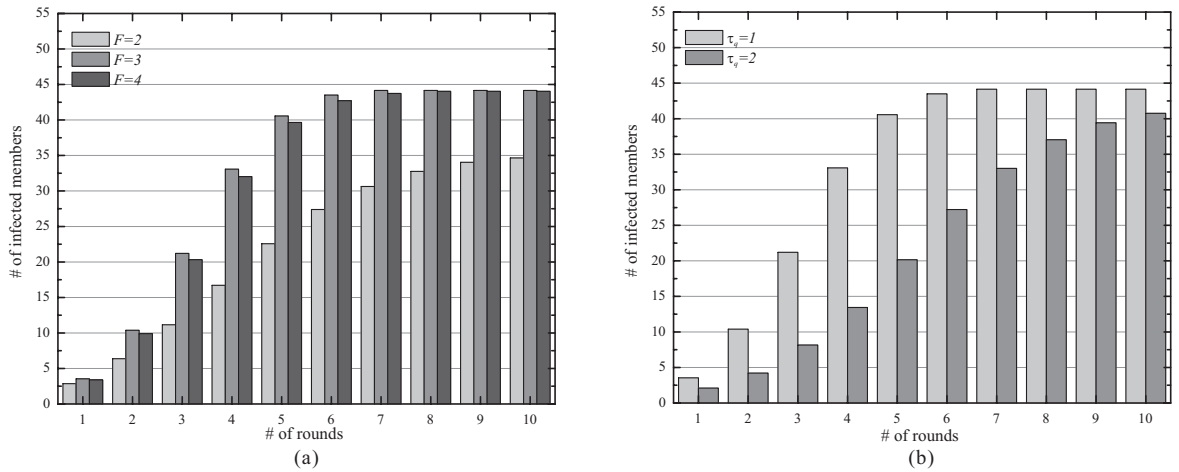


Fig. 6. Expected number of infected members at a given round for $n = 50$ within a network of 100 nodes with each node having a maximum speed of 2m/s and an average pause time of 40s. (a) $\tau_q = 1$ with different values for $F$. (b) $F = 3$ with different values for $\tau_q$.

*a) Recurrence Relation:* Given the probability $p$ that a certain member is infected by a gossip message, $q = 1 - p$ represents the probability of non-infection. With $s_r = i$ and $\sum_{t=1}^{\tau_q} \Delta s_{r+1-t} = \delta$ in the current round, we introduce a binary random variable, $X_k$, for each of the remaining $n - i$ susceptible members, where $P(X_k = 0) = q^\delta$, i.e., the probability that a certain susceptible member is not infected in the next round is the probability that it is not infected by any of the $\delta$ infectious members. It is clear that $s_{r+1} - s_r = \sum X_k$ follows a binomial distribution. Given an anticipated number of $j$ infected members in the next round, the transition probability $p_{(i,j)\delta}$ is expressed as:

$$\begin{aligned} p_{(i,j)\delta} &= P(s_{r+1} = j | s_r = i, \sum_{t=1}^{\tau_q} \Delta s_{r+1-t} = \delta) \\ &= P(\sum X_k = j - i) \\ &= \begin{cases} \binom{n-i}{j-i}(1-q^\delta)^{j-i} q^{\delta(n-j)} & j \geq i \\ 0 & j < i \end{cases} \end{aligned} \quad (1)$$

Then, with the convention that message $m$ is injected into the system at round $r = 0$ by the originating member, the initial distribution of $s_r$ is given by:

$$P(s_0 = j) = \begin{cases} 1 & j = 1 \\ 0 & j > 1 \end{cases} \quad (2)$$

Having the initial distribution and transition matrix $\mathcal{P}_\delta = \{p_{(i,j)\delta}\}_{i,j,\delta \in \mathcal{E}}$, $\nu_r$, the distribution of $s_r$, is then computed as:

$$\nu_{r+1}^T = \nu_r^T \mathcal{P}_\delta \quad (3)$$

Where $\nu_r(i) = P(s_r = i)$ is the $i$th element of the column vector $\nu_r$.

*b) Determining Parameters:* According to our assumptions, the probability $p$ can be estimated by taking two conditions into account: (1) the considered node is chosen as the gossip destination and (2) the gossip message is successfully received. This results in the following expression:

$$p = \overbrace{P_{gossip}}^{(1)} \overbrace{P_{succ}}^{(2)} = \left(\frac{F}{n-1}\right) P_{succ} \quad (4)$$

Given a certain length (in hops) $h$ of a routing path, the probability of successfully delivery is expressed as $P_{succ} = (1-p_f)^h$. According to Bayes's rule of exclusive and exhaustive causes [24]:

$$P_{succ} = \sum_h (1-p_f)^h P(H = h) = \mathbf{E}_H[(1-p_f)^H] \quad (5)$$

Therefore, $p$ is expressed as:

$$p = \left(\frac{F}{n-1}\right) E_H[(1-p_f)^H] \quad (6)$$

The distribution of $H$ and the value of $p_f$ are the network information we need. We discuss their estimations in Appendix.

*2) Evaluation of the Single Packet Dissemination Reliability:* With Equation 3, we can recursively compute the distribution of the number of infected nodes with respect to the two parameters $F$ and $\tau_q$. The evaluation reported here is for a group of size 50 within a network of 100 nodes with each node having a maximum speed of 2m/s and an average pause time of 40s. Fig. 6 shows the progression of the infection. From Fig. 6(a), it is easy to observe that the reliability of the protocol with $F = 3$ is better than the one with $F = 2$, because the fanout has a significant effect on the reliability. However, when we further increase the fanout, the reliability again decreases instead of increasing. The reason is that increasing the fanout has the same effect as increasing the number of connections, and $p_f$ increases dramatically because of the network congestion. A similar reason accounts for what happens when $\tau_q$ changes from 1 to 2.

In fact, there is always a trade-off between certain requirements on reliability and the introduced overhead, characterized by the values of $F$ and $\tau_q$. Considering the network capacity imposes a further limitation not considered in other efforts

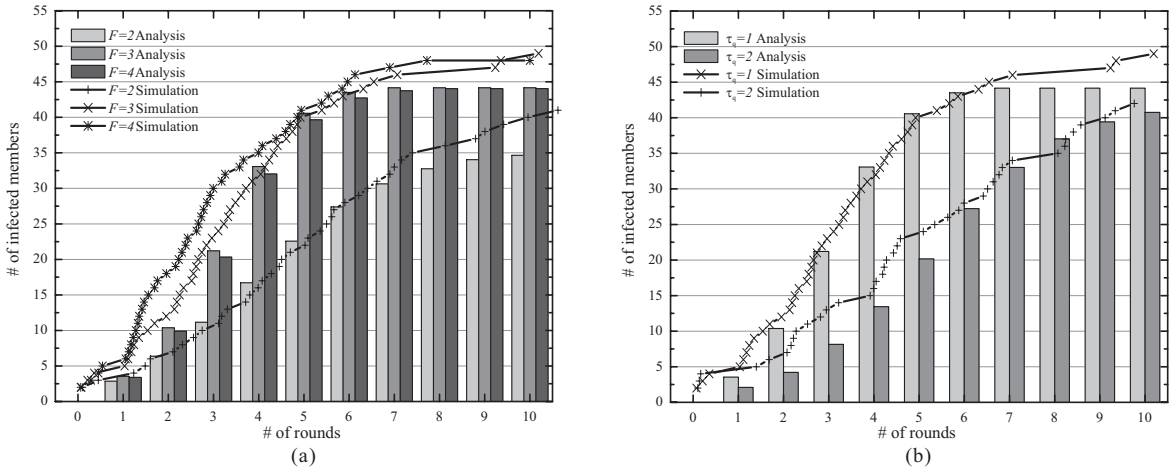


Fig. 7. Average number of infected members (simulation results) vs expected number of infected members (analytical results) in time (expressed in rounds) with $n = 50$ and a maximum node speed of 2m/s. (a) $\tau_q = 1$ with different values for $F$. (b) $F = 3$ with different values for $\tau_q$.

(considerably large $F$ [25], unbounded $\tau_q$ [7]). According to our analysis, for all network settings considered in this paper, the optimal values of the parameters for the RDG protocol are always $F = 3$ and $\tau_q = 1$.

*3) Reliability Probability Distribution:* Having the single packet dissemination reliability measure $\nu(i)$[5], the reliability of disseminating a flow of $M$ packets, i.e., $\pi_M(\zeta)$, can be expressed as:

$$\pi_M(\zeta) \quad = \quad \binom{M}{\lceil \zeta M \rceil} p_1^{\lceil \zeta M \rceil} (1 - p_1)^{\lceil (1-\zeta) M \rceil} \qquad (7)$$

where $p_1 = \sum i \cdot \nu(i)/n$ is the probability that a certain group member receives a single packet. Here we again use Bayes's rule of exclusive and exhaustive causes and assume that the receptions of two distinct packets are independent events.

## V. Simulations

This section presents the practical evaluation of our RDG protocol. We first compare our simulation results with the corresponding analytical ones in order to confirm the predictability of our RDG protocol. We evaluate the reliability of RDG by comparing with the Anonymous Gossip (AG) [13] protocol[6] and the protocol efficiency by measuring using the metric defined in [26].

### A. Model

The version of *ns-2* we have made use of includes the Monarch Project wireless and mobile extensions. Besides various implementations of ad hoc routing protocols, e.g., DSR, the Monarch extensions incorporate a radio model based on the Lucent WaveLAN IEEE 802.11 product, which provides a 2Mbps transmission rate and a nominal range of 250m. We adopt the two-ray ground reflection model as the radio propagation model.

[5]The subscript $r$ is dropped hereafter, because we always consider the final distribution after the last round.

[6]Comparisons with AG on efficiency are desirable but infeasible due to the big differences between the assumptions about the underlying mechanisms.

We simulated a mobile ad hoc network with 100 to 200 nodes in a 1000m $\times$ 1000m area, operating over 360 seconds of simulated time. The movement pattern was defined by the Random Waypoint model. Each node had a maximum speed between $2 \sim 20$m/s and an average pause time of 40s.

The network contained a single multicast group where half of the nodes are group members. Beginning at 10 seconds, the members consecutively joined the group until around 60 seconds. Then one of the members started to generate constant bit rate (CBR) traffic at regular intervals of 200ms with each packet having a length of 64 bytes until 340 seconds. All nodes left the group at 350 seconds. The gossip period was also set to 200ms. Each simulation was carried out 10 times with different scenario files created by *ns-2*.

### B. Single Packet Dissemination Reliability

Fig. 7 shows some comparisons between analytical and simulation results of the basic RDG protocol. We can have two main observations: (1) The simulation results follow the trend of the analytical ones very well. This basically means that the theoretical prediction of the relationship between the reliability and the latency is valid; (2) The simulation results exhibit higher reliability than the analytical ones. This is not a surprise, as we have already stated, the analytical result is a lower bound.

We also observe that the reliability of our protocol with $F = 4$ is slightly better than the one with $F = 3$ for the early rounds, which seems to contradict the theoretical prediction. In fact, since we allow a group member to deliver any packet that it forwards to others, the more connections the group members set, the more group members get unexpectedly infected in each round. This result seems to suggest that both $F = 4$ and $F = 3$ are "good" values with $\tau_q$ set to 1. However, the former is much less efficient than the latter.

### C. Reliability Probability Distribution $\pi_M(\zeta)$

Fig. 8 shows the reliability of both our basic protocol (RDG) and its variant (TA-RDG) with different network sizes and

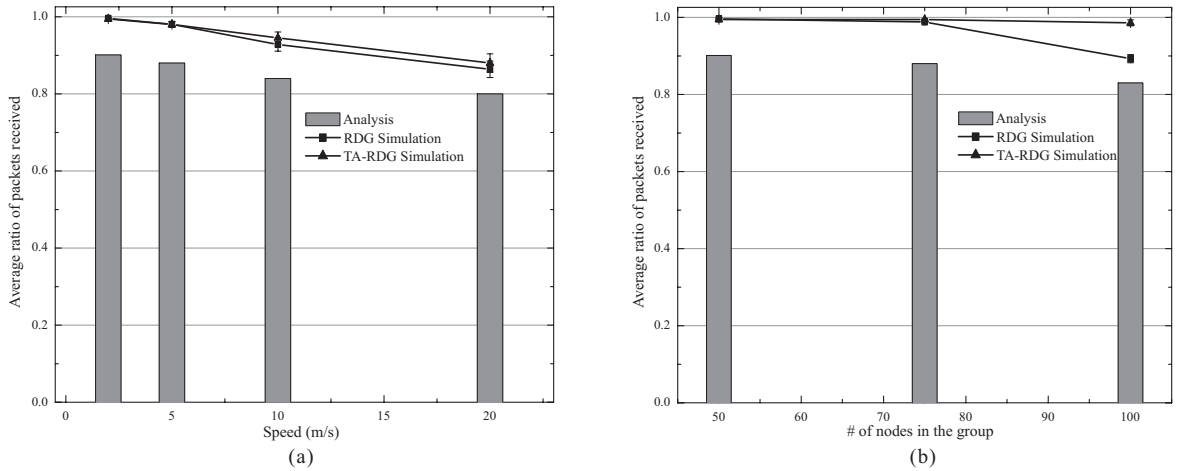


Fig. 8. (a) Reliability of the protocol in a group of $n = 50$ with maximum node speed varying between 2m/s and 20m/s. (b) Reliability of the protocol with a group size varying between 50 and 100 while the maximum node speed is 2m/s. The parameters are $F = 3$ and $\tau_q = 1$ for both cases.

mobility patterns. We provide here the mean value of $\zeta$ and its $95\%$ confidence interval, which characterize the reliability distribution. The simulation results again exhibit higher reliability than the analytical ones, because the latter do not take the gossiper-pull into consideration. In fact, the gossiper-pull greatly improves the reliability of the protocol. We also note that only a slight reliability degradation is observed when the group size or mobility is increased, illustrating our claims of scalability. As expected, TA-RDG performs better than RDG in most cases. The improvement is significant in large groups.

### D. Comparing Anonymous Gossip (AG) and TA-RDG

A systematic comparison between TA-RDG and AG [13] (discussed in Section I) is hard, due to their different design goals. We compare them in the context of small groups, which should actually favor AG since RDG is designed for larger groups. The comparison is done by superimposing a figure from [13] with corresponding simulation results for RDG (for the same scenario). The figure shows that RDG is more reliable than AG in most cases. Furthermore, AG cannot compete with RDG in terms of scalability because it is based on the underlying multicast protocol whose overhead is much larger than the one of the unicast protocol that RDG is based on. Finally, the reliability of the AG protocol is not as predictable as RDG's is, since it relies on the existence of an unpredictable multicast tree.

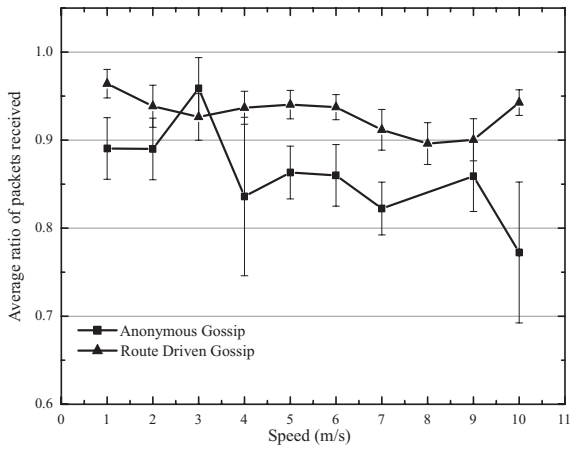


Fig. 9. Reliability of the AG and RDG protocols in a network of 40 nodes with approximately one-third of them in a group, located within a square of 200m×200m. The maximum node speed varies between $1 \sim 10$m/s and the average pause time is 40ms. The transmission range is 75m.

### E. Efficiency Evaluation for TA-RDG

The *bandwidth cost* defined in [26] is a good way to evaluate the overhead for unicast encapsulated multicast protocols. It basically counts the total network-level hops traveled by all packets corresponding to the delivery of one packet to the whole group. Fig. 10 shows the linearity between the bandwidth costs and group size, with the requirement of $\mathbf{E}[\zeta] \geq 0.99$. The errorbars represent the standard deviations. Considering certain similarities between TA-RDG and the protocol in [26], making comparisons would have been desirable.

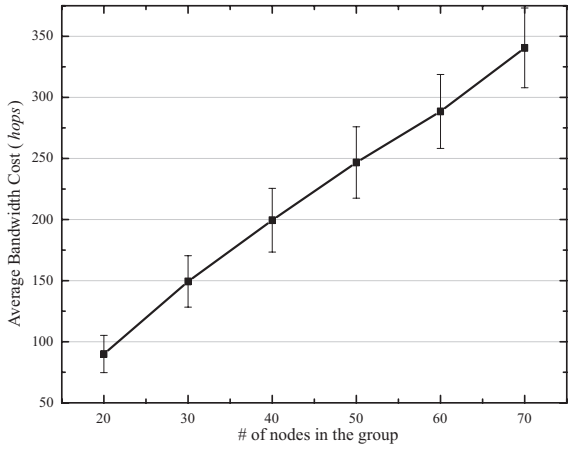


Fig. 10. Efficiency evaluation for TA-RDG in a network of 100 nodes with the number of nodes in a group varying from 20 to 70. The maximum node speed is 2m/s and the average pause time is 40ms.

This is, unfortunately, not feasible because the simulations in [26] do not concern the unicast routing and provide no reliability evaluation for a certain efficiency measurement.

## VI. Discussion

In this section, we discuss the possibility of evaluating our RDG protocol with an alternative specification to $\pi_M(\zeta)$ and potential optimizations of RDG.

### A. Protocol Evaluation against $\Delta$-Reliability

Based on the previous analysis and the protocol description, we also evaluate here the reliability of our RDG protocol in the face of another specification defined in [27] consisting of the following three properties:

- *Validity: correct process $p$ multicasts $m$ $\Rightarrow$ $p$ delivers $m$.* This can be trivially shown based on the protocol description.
- *Integrity: $m$ is delivered at most once for each correct process $p$, and only if $sender(m)$ multicasts $m$ before.* Since we do not consider Byzantine failures, no packet will be generated from the air. However, due to the limitations of buffers holding digests, the uniqueness of delivery is, despite unique packet identifiers, hard to ensure infinitely. This problem is not unique to our approach, especially since packet identifiers are not unbounded, but are reused. In practice, buffers have however proven sufficient capacity to avoid the observation of any duplicate delivery.
- *Agreement: correct process $p$ delivers $m$ $\Rightarrow$ a fraction $\rho$ of correct processes deliver $m$ with probability $\psi$.* By taking $\rho = s/n$, our protocol satisfies this property with probability $\psi(\rho) = \nu(s)$.

### B. Optimizations

The following are some optimization heuristics. The reason that we do not apply them to our protocol at this stage is that, although improving performance, they somewhat put the randomness embedded into the protocol at stake, making performance prediction hard.

- Use multicast to disseminate gossip messages. By exploiting the potential multicast support provided by DSR, the gossiper node builds a source tree based on the available routing information. Only one message is transmitted through a certain tree edge. Different copies of the message are generated only at the bifurcation node.
- Assign $P_{reply}$ adaptively at each member depending on the distance to the initiator of the GROUPREQUEST, i.e., the longer the path the bigger the value for $P_{reply}$. If a "near" member receives a GROUPREPLY from a "distant" one after it decides not to reply to the GROUPREQUEST, it would append its own reply to the packet before forwarding it. This optimization reduces the probability of different members along the same path separately generating a ROUTEREPLY, and hence reduces the bandwidth consumption.
- Add a directional flavor to the gossip scheme. A node would carefully select the directions of the gossip by directing the message to target peripheral members, i.e., the members that might not receive the gossip message in the current round, according to the knowledge of this node on the gossip messages it receives. The awareness of direction could be obtained by a GPS system, but also by a GPS-free mechanism (e.g., [28]).

## VII. Related Work

This section summarizes previous work related to our RDG protocol, with respect to both gossiping mechanisms and multicasting techniques.

### A. Gossiping in Wired Networks

The *Probabilistic Broadcast* (*pbcast*) [5] protocol has in much rejuvenated the interest in gossip-based protocols that find their origins at Xerox where they were initially used for replicated database maintenance [29].

The two phases of *pbcast* (a first phase based on an unreliable multicast primitive and a second phase making use of gossips for repairing packet losses) are merged into one phase by the *Lightweight Probabilistic Broadcast* (*lpbcast*) [7] protocol. By gossiping uniformly about data packets, digests, as well as membership information, *lpcast* provides reliability similar to *pbcast* without imposing a complete membership view on the members and has inspired the design of our RDG protocol.

By taking the network topology into account when gossiping, *Directional Gossip* (DG) [6] gains in efficiency. In short, a *weight* is computed for each neighbor node, representing the connectivity of that given node. The larger the weight of a node, the higher the possibility for it to receive a given packet from other nodes. When gossiping, nodes with higher weights are hence chosen with a smaller probability, reducing redundant sends. In particular, LANs are represented by single nodes to distant LANs, and "long" routes between two such representatives are seldom chosen. This is similar to our TA-RDG protocol, where routes with less hops are chosen more often for gossiping.

While the DG protocol does not provide any analytical evaluation whatsoever, protocols such as *pbcast* and *lpbcast* are analyzed in much detail based on a recurrence relation establishing the probability for the possible number of infected nodes at all gossip rounds. Alternatively, protocols are modelled by differential equations (e.g., [29]), or random graph theory (e.g., [25]). The latter protocol is tightly coupled to its analysis, in the sense that a particular packet is gossiped only once by a given node. Roughly, in such a model, there is a sharp threshold for the required fanout around $\log n$ ($n$ being the number of members in a multicast group) to ensure that with very high probability all nodes will receive a given multicast packet despite node and transmission failures.

### B. Gossiping in Ad Hoc Networks

The benefits of gossiping techniques have, rather recently, also been exploited in ad hoc networks. In this context, gossip-based protocols are not favored for obtaining an analytical prediction of their performance in terms of reliability, but more for the practical observation that they generate less traffic than, for instance, flooding approaches.

While the exploitation of this observation is briefly mentioned in [14], [15], it is more closely investigated by Haas et al. in [8] for the dissemination of routing messages. Since deterministic flooding techniques do not necessarily ensure that, in practice, every node sees a given information either, gossiping techniques yield results close to those of flooding protocols, yet imposing far less load on the network.

Prior to that, Vahdat and Becker [30] have also employed gossiping techniques for unicast routing. Their idea is to ensure that packets are eventually delivered even if there is no path between the source and the destination for some time. Such an approach is very interesting, but tends to require relatively high buffering capacities at individual nodes if all unicast traffic is handled that way. Just like all other gossip-based protocols for ad hoc networks we know of, this effort does not include any analytical performance estimation.

### C. Stateless Multicast

Our RDG protocol also follows a recent shift towards stateless multicasting [31], [26]. While the *Differential Destination Multicast* (DDM) [31] protocol explicitly calls the unicasting function to disseminate multicast packets, the protocol presented in [26] builds an overlay multicast packet distribution tree on top of the underlying unicast routing protocol, and multicast packets are encapsulated in a unicast envelop and transmitted between the nodes in the group. While reducing the control overhead of the multicast session, the protocol leads to overweighted packet headers. This problem, known from unicast source routing but amplified in the case of multicasting, limits the protocol's scalability in terms of the group size. Our RDG protocol avoids this problem by choosing only a subset of the group for each gossip session.

## VIII. Conclusions

In this paper, we have studied the problem of reliable multicast in ad hoc networks, assuming that no multicast primitive is available at the networking layer. To the best of our knowledge, this problem was never tackled before.

We have defined the problem of probabilistic reliability in a more practical way and proposed a solution based on gossiping, which is particularly well suited to the challenging peculiarities of the networks considered here. We have described its operations and developed an analysis of its performance, based on which the parameters (fanout and quiescence threshold, notably) can be fine tuned; we have shown the rapid propagation of data to all reachable members of the group; we have confirmed these results by simulations. Finally, we have compared our solution with the closest work already performed in this field.

In terms of future work, we intend to optimize our RDG protocol with respect to its overhead. We expect this to help us improve the practicality of RDG, in the sense of the modest cost incurred by the added reliability. This might give an indication on how our RDG protocol could be used by upper layer applications in an efficient way.

## IX. Acknowledgements

The authors wish to thank the anonymous reviewers for their valuable feedback for this work. The authors would also like to thank Milan Vojnovic, Rachid Guerraoui, Mario Cagalj, and Catherine Rosenberg for several instructive discussions.

## References

[1] V. Hadzilacos and S. Toueg, "Fault-tolerant broadcasts and related problems," in *Distributed Systems*, chapter 5, pp. 97–145. Addison-Wesley, 2 edition, 1993.

[2] S.E. Deering and D.R. Cheriton, "Multicast routing in datagram internetworks and extended LANs," *ACM Trans. on Computer Systems*, vol. 8, no. 2, pp. 85–110, 1990.

[3] S. Floyd, V. Jacobson, C-G. Liu, S. McCanne, and L. Zhang, "A reliable multicast framework for light-weight sessions and application level framing," *IEEE/ACM Trans. on Networking*, vol. 5, no. 6, pp. 784–893, 1997.

[4] S. Paul, K.K. Sabnani, J.C. Lin, and S. Bhattacharyya, "Reliable multicast transport protocol," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 3, pp. 784–893, 1997.

[5] K.P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky, "Bimodal multicast," *ACM Trans. on Computer Systems*, vol. 17, no. 2, pp. 41–88, 1999.

[6] M-J. Lin and K. Marzullo, "Directional gossip: gossip in a wide area network," in *Proc. of European Dependable Computing Conference (EDCC-3)*, 2000.

[7] P. Eugster, S. Handurukande, R. Guerraoui, A.M. Kermarrec, and P. Kouznetsov, "Lightweight probabilistic broadcast," in *Proc. of IEEE International Conference on Dependable Systems and Networks (DSN 2001)*, 2001, pp. 443–452.

[8] Z.J. Haas, J.Y. Halpern, and L. Li, "Gossip-based ad hoc routing," in *Proc. of INFOCOM 2002*, 2002.

[9] E.M. Royer and C.E. Perkins, "Multicast operation of the ad-hoc on-demand distance vector routing protocol," in *Proc. of the 5th annual ACM/IEEE international conference on Mobile Computing and Networking (MobiCom)*, 1999, pp. 207–218.

[10] S.J. Lee, M. Gerla, and C.C. Chiang, "On-demand multicast routing protocol," in *Proc. IEEE Wireless Communications and Networking Conference, 1999 (WCNC. 1999)*, 1999, vol. 3, pp. 1298 –1302.

[11] E. Pagani and G. P. Rossi, "Providing reliable and fault tolerant broadcast delivery in mobile ad-hoc networks," *Mobile Networks and Applications*, vol. 5, no. 4, pp. 175–192, 1999.

[12] S.K.S. Gupta and P.K. Srimani, "An adaptive protocol for reliable multicast in mobile multi-hop radio networks," in *IEEE Workshop on Mobile Computing Systems and Applications*, 1999, pp. 111–122.

[13] R. Chandra, V. Ramasubramanian, and K. Birman, "Anonymous gossip: Improving multicast reliability in mobile ad-hoc networks," in *Proc. 21st International Conference on Distributed Computing Systems (ICDCS)*, 2001, pp. 275–283.

[14] W. Heinzelmann, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *Proc. of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1999, pp. 174–185.

[15] S.Y. Ni, Y.C. Tseng, Y.S. Chen, and J.P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proc. of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1999, pp. 151–162.

[16] J.-P. Hubaux, T. Gross, J.-Y. Le Boudec, and M. Vetterli, "Toward self-organized mobile ad hoc networks: the terminodes project," *IEEE Communications Magazine*, vol. 39, no. 1, pp. 118–124, 2001.

[17] D.B. Johnson, D.A. Maltz, Y-C. Hu, and J.G. Jetcheva, *The dynamic source routing protocol for mobile ad hoc networks (DSR)*, February 2002, Internet-Draft, draft-ietf-manet-dsr-07.txt. Work in progress.

[18] K. Fall and K. Varadhan, Eds., *The ns Manual*, The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, Apr. 2002, Availiable from http://www.isi.edu/nsnam/ns/.
[19] J.G. Jetcheva, Y-C. Hu, D.A. Maltz, and D.B. Johnson, *A simple protocol for multicast and broadcast in mobile ad hoc networks*, July 2001, Internet-Draft, draft-ietf-manet-simple-mbcast-01.txt. Work in progress.
[20] M.K. Reiter, "The Ω key management service," in *Proc. of the 3rd ACM Conference on Computer and Communications Security (CCS)*, January 1996, pp. 38–47.
[21] L. Zhou and Z.J. Haas, "Securing ad hoc networks," *IEEE Network*, vol. 13, no. 6, pp. 24–30, 1999.
[22] J.-P. Hubaux, L. Buttyan, and S. Capkun, "The quest for security in mobile ad hoc networks," in *Proc. of ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2001, pp. 146–155.
[23] J.D. Murray, *Mathematical Biology*, Springer, Berlin, 2nd edition, 1993.
[24] P. Bremaud, *Markov chains*, Springer-Verlag, New York, 1999.
[25] A.-M. Kermarrec, L. Massoulie, and A. Ganesh, "Probabilistic reliable dissmination in large-scale systems," *IEEE Trans. on Parallel and Distributed Systems (to appear)*, 2003.
[26] K. Chen and K. Nahrstedt, "Effective location-guided tree construction algorithms for small group multicast in MANET," in *Proc. of INFOCOM 2002*, 2002, pp. 1192–1201.
[27] P. Eugster, R. Guerraoui, and P. Kouznetsov, "Delta-reliability," Tech. Rep. DSC ID:200110, School of Computer and Communication Sciences, EPFL, 2001.
[28] S. Capkun, M. Hamdi, and J.-P. Hubaux, "GPS-free positioning in mobile ad-hoc networks," *Cluster Computing Journal*, vol. 5, no. 2, pp. 118–124, 2002.
[29] A. Demers, D. Greene, C. Hauser, W. Irish, and J. Larson, "Epidemic algorithms for replicated database maintenance," in *Proc. of 6th Annual ACM Symposium on Principles of Distributed Computing (PODC'87)*, 1987, pp. 1–12.
[30] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad hoc networks," Tech. Rep. CS-2000-06, Duke University, 2000.
[31] L. Ji and M.S. Corson, "Differential destination multicast - a MANET multicast routing protocol for small groups," in *Proc. of INFOCOM 2001*, 2001, pp. 1192–1201.

## APPENDIX

In order to obtain the distribution of $H$, we assume that the network nodes are uniformly distributed within a circle of diameter equal to 10 hops[7]. Then, by repeating the procedure of randomly picking up two points within this circle and computing the distance between them, we obtain the distribution function of $H$ in a numerical way. The distribution $P(H)$ is shown in Fig. 11.

The other important step is to estimate $p_f$. We assume that $p_{f_{mo}} \gg p_{f_c}$, so $p_{f_{mo}}$ is directly used to approximate $p_f$. The estimation of $p_{f_{mo}}$ is done by simulation with *ns-2*. Since this parameter is determined by both movement and traffic pattern, we apply the same movement scenario as to the simulation for our protocol with the heaviest traffic load. The heaviest load of our protocol is when the network is loaded with about $F \times n$ connections and the sending rate is the basic rate imposed by the upper layer times the $\tau_q$. For example, we simulate a scenario of 50 sources and 150 connections for a group of 50 members with $F = 3$. The results are average packet loss ratio $p_l$ and the distribution of the number of hops $H_l$ traveled by a packet before getting dropped (See Fig. 12 for an example). It is easy to see that $P(H_l = 1)p_l = \sum_h p_{f_{mo}} P(H = h)$. In fact, both sides of the equation give the probability that a packet gets lost at the first hop. Therefore, we have $p_{f_{mo}} = P(H_l = 1)p_l$. An example of the values used for the analysis is provided in Fig. 13. It can be observed that $p_{f_{mo}}$ is an increasing function of both $F$ and $\tau_q$.

[7]This means that a node at the end of the diameter should take 10 hops to reach a node at the other end. The uniform distribution also implies that the path length between two nodes is approximately the same as the distance between them.

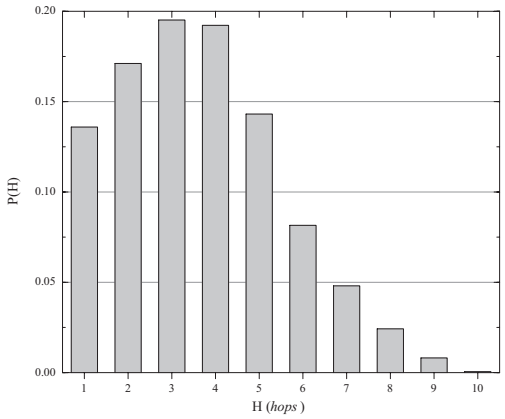


Fig. 11. Distribution of $H$. Here $H$ is the random variable representing the distance between two randomly picked points within a circle. It can be considered as the length in hops of a routing path between two randomly picked network nodes, with the assumption that the nodes are uniformly distributed.

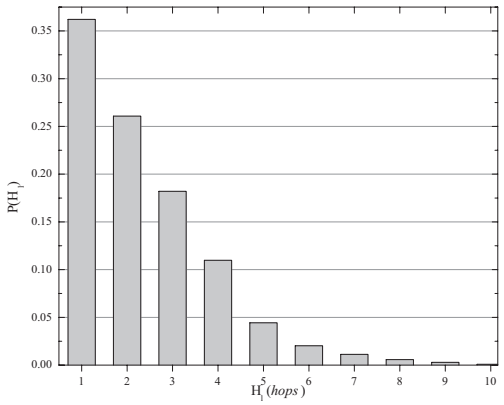


Fig. 12. Distribution of $H_l$ when average packet loss ratio equals to 12.7%, assuming a group size of 50 and a network size of 100 with $F = 3$ and $\tau_q = 1$.

| $F$ \ $\tau_q$ | 1 | 2 |
|---|---|---|
| 2 | 0.0200 | — |
| 3 | 0.0460 | 0.2749 |
| 4 | 0.1686 | — |

Fig. 13. The $p_{f_{mo}}$ with respect to different values of $F$ and $\tau_q$, assuming a group size of 50 and a network size of 100.