

# Hypergossiping: A Generalized Broadcast Strategy for Mobile Ad Hoc Networks

Abdelmajid Khelil, Pedro José Marrón, Christian Becker, Kurt Rothermel

Universität Stuttgart, IPVS/VS, Universitätsstrasse 38, 70569 Stuttgart  
{khelil, marron, becker, rothermel}@informatik.uni-stuttgart.de

**Abstract.** Broadcasting is a commonly used communication primitive needed by many applications and protocols in mobile ad hoc networks (MANETs). Unfortunately, most broadcast solutions are tailored to one class of MANETs with respect to node density and node mobility and are unlikely to operate well in other classes. In this paper, we introduce hypergossiping, a novel adaptive broadcast algorithm that combines two strategies. Hypergossiping uses adaptive gossiping to efficiently distribute messages within single network partitions and implements an efficient heuristic to distribute them across partitions. Simulation results in ns-2 show that hypergossiping operates well for a broad range of MANETs with respect to node densities and mobility levels.

## 1 Introduction

Mobile ad hoc networks (MANETs) are networks formed on-the-fly by mobile nodes equipped with short range communication capabilities. Such networks are suitable for scenarios where an infrastructure is unavailable and communication must be deployed quickly, e.g. in disaster-rescue or military scenarios.

Broadcasting is a common communication mechanism in MANETs. It is frequently deployed for data dissemination, and for topology discovery and maintenance [1]. Flooding is a common approach to realize broadcasting in MANETs because of its topology independence. In flooding-based approaches nodes forward a received message to all their neighbors. Subsequently, all nodes within the network should receive the message.

But flooding exhibits some serious problems. At the two extremes we can consider dense MANETs and sparse MANETs with respect to the node density, i.e. the number of nodes operating in a given area. While dense MANETs encounter so-called *broadcast storms* [2], where collisions on the Media Access Control (MAC) layer extinguish broadcast messages, sparse MANETs are challenged by frequent *network partitioning*, where messages do not reach every node in one flooding round. Two common strategies can be applied to conquer these extreme cases. First, *selective strategies*, e.g. gossiping [3], cause only subset of nodes to forward a message reducing the probability of broadcast storms. Second, selective *repetition of broadcasts*, e.g. hyperflooding [4,5], can be used to overcome network partitions.

Most broadcasting techniques are unfortunately tailored to one class of MANETs and are likely not to operate well in other classes (see related work section). Our main objective is to provide an adaptive broadcast algorithm for a wide range of MANET operation conditions. The main contribution of this paper is hypergossiping, a novel

generalized broadcast mechanism that combines two strategies and provides a configuration depending on the local density of a node, reflected by the number of its neighbors. Using simulation results we show that hypergossiping can be deployed in a wide spectrum of MANETs with respect to node densities and mobility levels.

The remainder of this paper is organized as follows. The next section describes the system model and the requirements on a generalized broadcast strategy for MANETs. In Section 3 we discuss the related work. Section 4 introduces our generalized broadcast strategy, i.e. hypergossiping. In Section 5 we first define the simulation model and the evaluation metrics, then we calibrate the parameters of hypergossiping, evaluate it, and compare it to related work. Section 6 summarizes the paper and gives an overview of ongoing and future work.

## 2 Preliminaries

This section briefly presents the underlying system model of our approach. Based on the characteristics of the system model we derive some important requirements on our generalized broadcast algorithm.

### 2.1 System Model

We consider MANETs that are formed by mobile nodes of similar communication capabilities (communication range and bandwidth). We do not assume nodes to have knowledge about their position or speed. The MANET may show very heterogeneous spatial distribution of nodes, from locally very sparse to very dense, and very heterogeneous node mobility pattern: from low mobile to highly mobile. We assume that devices do not change their trajectories for communication purposes.

Broadcast data has typically a temporal and spatial relevance [6]. Broadcast algorithms have to consider this spatio-temporal relevance while broadcasting. In this paper, we consider only the temporal relevance of data and assume that information becomes irrelevant after a certain period of time, i.e. its *lifetime*. Lifetime is application dependent and may be in the range of seconds, minutes, or even hours.

### 2.2 Requirements

Because node density heavily influences the performance of broadcasting, and MANETs may show a wide range of node densities, the first requirement on a generalized broadcast strategy for MANETs is to adapt to the density of the network, in order to reduce broadcast storms and overcome partitioning. Since global state in MANETs is hard to obtain and spatial distribution of nodes may change continuously, the second requirement on such a strategy is that nodes independently adapt to local MANET characteristics. Third, different instances of the adaptive broadcast strategy have to interoperate in order to deliver messages through the network where different instances are present due to heterogeneity of density.

### 3 Related Work

Most of existing broadcast algorithms are developed for unpartitioned MANETs, and subsequently break in partitioned ones. They are also optimized for specific scenarios, and subsequently do not support a broader range of MANET situations. In [7] and [8] authors provide two comparative studies for these non-adaptive schemes.

In order to suit these schemes to a broader range of operation conditions, [9] and [10] adapted some of them to local MANET characteristics. In [9] the authors proposed three adaptive schemes, namely, adaptive counter-based (ACB), adaptive location-based (ALB), and neighbor-coverage (NC) scheme. Using simulations the authors derived the best appropriate counter-threshold respectively coverage-threshold for ACB respectively ALB as a function of the number of neighbors. The authors adapted the NC scheme by adjusting dynamically the HELLO interval to node mobility reflected by neighborhood variation, so that the needed 2-hop topology information gets more accurate. Although this optimization, NC scheme still has the main drawback that neighborhood information may be inaccurate in congested networks. The authors showed that these adaptive schemes outperform the non-adaptive schemes and recommend ACB if location information is unavailable and simplicity is required. We will compare our strategy to ACB. [10] introduced the density-aware probabilistic flooding. Nodes use the following forward probability:  $p = \inf\{1, 1/n\}$ , where  $n$  is the current number of neighbors. We will also compare our strategy to this scheme. ACB, ALB, NC and density-aware probabilistic flooding support a broad range of dense MANETs but they still show poor performance in partitioned networks.

The first step towards a single solution for all MANET situations was the integrated scheme presented in [5]. We refer to this scheme as integrated flooding (IF). Nodes switch at run-time between three flooding schemes, namely, plain flooding, scoped flooding, and hyperflooding. Authors recognize mobility as main cause of broadcast partitioning [11] and switch between these schemes according to the relative node mobility. To this end, nodes include velocity information (speed and direction) in HELLO beacons. To the best of our knowledge, IF is the single existing adaptive MANET broadcast protocol that considers network partitioning. Unfortunately, IF shows some drawbacks. First, IF adapts to node mobility but not to node density, which makes it break for sparse low-mobile networks that encounter partitions frequently. Second, scoped flooding uses a predefined forward threshold, which makes IF less efficient than the above adaptive schemes in highly dense scenarios. Third, relying on velocity information presents a strong limitation of the deployment of IF. In Section 5.6, we compare our solution to IF.

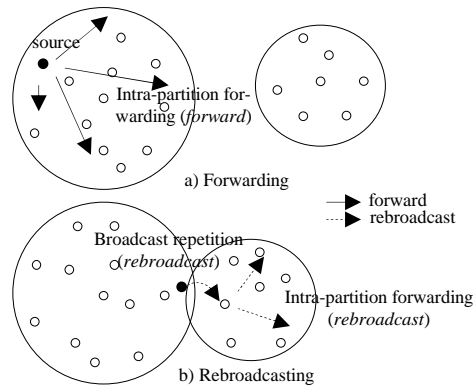
### 4 Hypergossiping (HG)

In this section, we present our generalized solution fulfilling the requirements above.

#### 4.1 Approach

In general we can consider a MANET as a set of partitions, which may join or split over time. Thus, we decide to superpose the following two strategies to realize hyper-

gossiping. The first strategy allows an efficient broadcasting within a single partition of the MANET. We refer to this strategy as “intra-partition forwarding” (Fig. 1a)). The second strategy permits an efficient broadcast repetition on partition joins. We call this strategy “broadcast repetition”. To this end nodes have to buffer messages during their lifetime and to retransmit an adequate subset of them on partition joins. After broadcast repetition the first strategy can continue to distribute the message to the joining nodes (Fig. 1b)). Note the difference between *forward* and *rebroadcast* explained in Fig. 1. Depending on the mobility of nodes, the node spatial distribution and the lifetime, messages will succeed to other partitions or not.



**Fig. 1:** Approach

In our approach, we assume that each node stores the list of IDs of messages received or originated with their remaining lifetime in a so-called *broadcast\_table*. Thus, nodes are capable to decide, whether a received copy of a given message is the first one. Nodes continuously decrement the lifetimes. Nodes purge entries from the *broadcast\_table* and possibly from the buffer, when the correspondent lifetime expires. When a message is forwarded or rebroadcasted, the remaining lifetime is included in the message.

## 4.2 Intra-Partition Forwarding

The analysis of the broadcast storm problem [2] suggests gossiping or probability-based flooding, thus we chose gossiping for intra-partition forwarding.

On receiving the first copy of a given message, gossiping forwards or rebroadcasts the message, with a probability  $p$ , to all nodes in the receiver’s communication range. In order to reduce broadcast storms, we follow a couple of strategies. First, nodes delay each intra-partition-forward for a random time between 0 and  $fDelay$ , which reduces the number of collisions. Second, we adapt the gossiping probability  $p$  to the node’s current number of neighbors, which reduces forward redundancy, contention, and collisions. For this purpose nodes acquire the number of their neighbors by means of periodic HELLO beaconing.

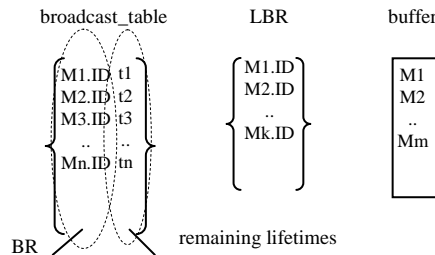
### 4.3 Broadcast Repetition

The common strategy to overcome partitioning is the repetition of broadcasting. For this purpose nodes need two mechanisms; one to detect *when* to rebroadcast and a second to decide *what* to rebroadcast. In the following we introduce our novel partition detection heuristic and rebroadcasting protocol.

We let nodes share with their neighbors the IDs of recently received or locally originated packets. We arbitrarily term this list by “Last Broadcast Received” or short *LBR* (Fig. 2). The rationale behind this is that two neighboring nodes that belong to the same partition should have received the same broadcasts that had taken place in this partition. By this means nodes are able to conclude whether they are populating the same partition. If a node receives an LBR that “sufficiently” differs from its own LBR, the node can conclude with a certain confidence that it is joining a new partition. We denote the maximum allowed size of an LBR by *maxLBRlength*. Nodes trigger rebroadcasting only if the overlap between received LBR and own LBR does not exceed a given percentage of the own LBR. We denote this percentage threshold by “intersection threshold” (*IS\_threshold*). In order to provide an accurate detection of partition joining, *maxLBRlength* and *IS\_threshold* have to be dimensioned appropriately. In Sec. 5.4 we show how we calibrate *maxLBRlength* and *IS\_threshold*.

This strategy is suitable to detect both causes of broadcast interruption: Network partitioning and broadcast storms. First, if two partitions say P1 and P2 join, some nodes of partition P1 will receive LBRs from other nodes belonging to the formerly partition P2. In this way nodes are able to detect the join event. Second, if a broadcast stops to progress within a partition due to collision or contention, nodes that received the broadcast may detect this on receiving the LBR of one neighbor that has not yet received the packet.

In order to save bandwidth, nodes exploit the existing HELLO beaconing to share their LBRs. Exchanging the LBRs is only necessary if a new neighbor is detected. Thus we do not include the LBR in each HELLO beacon but only in that beacon that just follows the discovering of a new neighbor. This delays the broadcast repetition until the next discovery of a new neighbor, in case of broadcast interruptions caused by broadcast storms.



**Fig. 2:** Definition of BR, LBR and buffer

To allow rebroadcasting nodes should buffer messages that should be rebroadcasted. If not otherwise stated, we assume that nodes buffer all received and originated messages during their lifetimes, i.e.  $m=n$  (Fig. 2). A node triggers rebroadcasting by

MAC-broadcasting its *BR list* (Fig. 2). Thus neighbors know which packets the sender has already received and can select from buffer the packets that missed this sender. On receiving these new packets the sender gossips them so they can reach all joining nodes. To increase efficiency nodes schedule the rebroadcasting at a random time between 0 and *rDelay* upon the reception of BR lists. Nodes cancel rebroadcasting if a neighbor starts to rebroadcast before the scheduled time. To reduce the probability of collisions nodes do not rebroadcast all packets at once but wait a random time between 0 and *fDelay* before rebroadcasting next packet (Fig. 3).

```

Var IS_threshold, maxLBRlength;
Var lifetime, fDelay, rDelay;
List myLBR; myBR; broadcast_table;

On receiving a DATA message (msg) M:
// do gossiping(p):
if(M is received for the first time) {
  deliver M;
  insert a copy of M to buffer;
  insert {M.ID} to myLBR;
  insert {M.ID, lifetime} to broadcast_table;
  if (random(1.0) <= p) {
    wait (random(fDelay));
    send M to all neighbors;
  }
} else {discard M}

On discovering new neighbor:
insert myLBR to next HELLO beacon;
On expiration of lifetime of M:
if (M in buffer) {delete M from buffer};
if (M.ID in myLBR) {delete M.ID from myLBR};
delete entry of M from broadcast_table;

On receiving HELLO with LBR:
// do partition detection:
is=card(myLBR∩recvLBR)/card(myLBR);
if(is <= IS_threshold){
  send BR to neighbors;
}

On receiving HELLO with BR:
// do broadcasting repetition:
Set timeout = random(rDelay);
On receiving msg with ID in (myBR - recvBR): exit();
On timeout:
Foreach buffered msg M with ID in (myBR - recvBR){
  wait (random(fDelay));
  rebroadcast M;
}

```

Fig. 3: Pseudo-code for hypergossiping

## 5 Performance Evaluation

In this section, we introduce the simulation model and define our evaluation metrics. We then study the performance of hypergossiping and compare it to related work.

### 5.1 Simulation Model

For evaluation we use the network simulator ns-2 [12]. We use as physical layer the TwoRayGround propagation model and as MAC layer the IEEE 802.11 standard. We generate *N* mobile nodes in a 1000mx1000m field, where these nodes move according to the random waypoint mobility model. We use the following communication load model: at the beginning of the simulation *S* nodes initiate broadcasting at a random time between 1 and 3 seconds, and continue to send packets with a constant packet rate. We use a fixed lifetime value during a simulation, i.e. all senders use the same lifetime for all packets they generate. For the same simulation scenario we ran 10 passes with 10 different movement traces and considered the average. Table 1 summarizes the simulation parameters of our experiments, which show a wide range of node

density and mobility. Packet rate values cover most of the MANET application scenarios known from the MANET literature.

**Table 1.** Simulation parameters.

Parameter	Value (s)
Simulation area	1000m x 1000m
Number of nodes	N in {50,100,200,300, 500,800}
Com. range	R = 100m
Com. rate	r = 1 Mbit/s
Data packet size	280 bytes
Movement pattern	Random waypoint
- Max speed	- v in {3, 12.5, 20, 30} m/s
- Pause	- Uniform betw. 0 and 2s
fDelay	10ms
rDelay	100ms
Lifetime	[5 .. 1800] s
Packet rate	[0.001 .. 1] packets/s
HELLO beaconing	Random betw. 0.75s and 1.25s

## 5.2 Evaluation Metrics

For the evaluation of broadcast protocols the following metrics are typically used:

- *Reachability (RE)*: the ratio of mobile hosts receiving the packet to the total number of mobile hosts. This metric measures the delivery reliability of the broadcast algorithm.

- *Delay*: Average end-to-end delay over all receivers.

- *MNF(R)*: Mean Number of Forwards (and Rebroadcasts) per node and packet. MNF(R) is an efficiency metric of the broadcast algorithm.

In Table 2 we illustrate the above metrics for hypergossiping. We denote by  $t_s$  the origination time of the packet and by  $t_i$  the arrival time of the packet at node  $i$ . With respect to a given broadcast packet we define the following four sets of nodes:

- *Forwd*: Nodes that forward the packet.

- *Reb*: Nodes that rebroadcast the packet.

- *R(H)*: Nodes reached by means of rebroadcasting.

- *R(HG)*: Nodes reached by HG, i.e. by means of either forwarding or rebroadcasting.

*Gain* measures the mean number of additionally reached nodes per rebroadcast. It presents a suitable efficiency metric for broadcast repetition strategies.

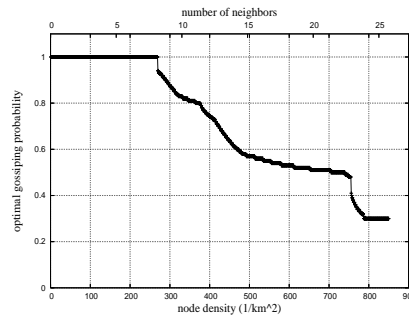
**Table 2.** Evaluation metrics.

Metric	Symbol	Value
Reachability	RE	$= \text{card}\{R(HG)\} / N$
Mean Number of Forwards	MNF	$= \text{card}\{\text{Forwd}\} / N$
Mean Number of Forwards & Rebroadcasts	MNFR	$= (\text{card}\{\text{Reb}\} + \text{card}\{\text{Forwd}\}) / N$
Average end-to-end delay over all receivers	delay	$= \frac{1}{\text{card}\{R(HG)\}} \sum_{i \in R(HG)} (t_i - t_s)$
Gain or mean number of additionally reached nodes per rebroadcast	gain	$= \text{card}\{R(H)\} / \text{card}\{\text{Reb}\}$

### 5.3 Adaptation of Gossiping

In this section, we adapt gossiping to local node density by determining the appropriate gossiping probability in function of number of neighbors.

In previous work [13], we have investigated the impact of density on data distribution in a MANET. Based on an epidemic model the optimal gossiping probability was calculated depending on the node density (Fig. 4).



**Fig. 4:** Adaptation of gossiping

Consistent with our second requirement of generalized broadcasting strategy, we let every node set locally and independently the gossip probability. Given  $n$  the number of neighbors and  $R$  the communication range, a node computes easily its local density by:

$$d = \frac{n+1}{\pi \times R^2} \quad (1)$$

According to this value the node has to set on-the-fly the gossiping probability. To avoid the computation of local node density, which also assumes that nodes know their communication range, we recommend that nodes select the gossiping probability depending on the current number of neighbors  $n$ . We realize that by scaling the x-axis of Fig. 4 using formula (1).

### 5.4 Calibration of Partition Detection

In this paper, we evaluate our partition detection by measuring the efficiency of rebroadcasting. A suitable efficiency metric for rebroadcasting is the mean number of additionally reached nodes per rebroadcast, i.e. its gain. The higher its gain, the more efficient is rebroadcasting. To dimension the partition detection parameters, i.e. IS\_threshold and maxLBRlength, we select the values with maximal gain.

For calibration we arbitrarily fix lifetime to 60s and maximum speed to 30m/s and we vary IS\_threshold in {0%, 25%, 50%, 75%, 100%} and maxLBRlength in {1, 5, 10, 25, 50, 100}. For every combination we compute the gain and select the combination that maximizes the gain. LBR serves anyway for the identification of a certain partition until next join. An identification should consider the size of the partition (reflected by number of nodes) and the number of broadcasts originated per unit of time in that partition (reflected by the packet rate). That is why we repeated the calibration process for a wide range of number of nodes and packet rates. The combina-

tions, for which the gain is maximal, are listed in Table 3. We observe that the denser is the network or the more congested it is, the smaller IS\_threshold but the higher maxLBRlength should be selected. We repeated these steps for lifetime values of 600s and 1800s and concluded that these combinations are almost independent from the lifetime value.

**Table 3.** Calibration of partition detection.

N:	50	100	200	300	500
n:	0,57	2,14	5,28	8,42	14,7
1 packet/s	25%, 100	25%, 100	0%, $\geq 50$	0%, 100	0%, 100
0.1 packets/s	50%, 100	25-50%, 100	25-50%, $\geq 25$	$\leq 50\%$ , $\geq 25$	0%, $\geq 50$
0.001 packets/s	25-75%, $\geq 25$	$\leq 75\%$ , $\geq 25$	$\leq 75\%$ , $\geq 10$	$\leq 50\%$ , $\geq 25$	0%, $\geq 25$

In this work, we use a simple calibration of partition detection. We use for MANETs with higher densities than 200 nodes/km<sup>2</sup> the tuple (0%, 100), otherwise we use the tuple (25%, 100). This calibration is suitable for most of simulated scenarios in Table 3. Consistent with our second requirement, we let every node select locally and independently the IS\_threshold value: At run-time a node sets IS\_threshold to 25% if its current number of neighbors is lower than 6 and 0% otherwise.

## 5.5 Performance of Hypergossiping

After adaptation and calibration we now investigate the performance of HG for a wide range of node density and mobility levels. We select for all scenarios S=25 senders. If v=0m/s, nodes being static do not discover new neighbors and therefore do not trigger broadcast repetition. That is why HG goes into simple gossiping.

Node mobility contributes to overcome network partitioning. It then follows that the higher is the mobility, the higher is the reachability and the lower is the delay. Fig. 5a) shows that the impact of node mobility on reachability is more significant for lower lifetimes. For very short lifetimes HG reachability is similar to that of simple gossiping. Fig. 5a) illustrates that reachability saturates at around 80%. We explain this as follows: at a sendrate of 0.0005 packets/s every sender originates only 1 packet within lifetimes up to 2000s. So for lifetimes considered in Fig. 5a) only 25 messages are relevant in the MANET. LBRs can store all IDs of these messages. The rebroadcasting condition (IS\_threshold $\leq$ 25%) becomes over time stronger and some partitions could not be detected.

We now set arbitrarily the lifetime to 600s and the sendrate to 0.001 packets/s. Fig. 5b) shows that rebroadcasting can strongly increase reachability in sparse MANETs; For 50 nodes and 3m/s reachability increases from 8% to 68%. Hypergossiping also increases reachability if gossiping reachability drops because of collisions; reachability increases from 63% to 92% for 500 nodes and 3m/s. Hypergossiping keeps the MNFR very low while increasing RE; Nodes namely forward or rebroadcast a given message in average maximal 1.1 times (Fig. 5c)). The bend in reachability and MNFR at 200 nodes is due to the simplicity of our partition detection calibration, where IS\_threshold jumps from 25% to 0% by node densities around 200 nodes/km<sup>2</sup> (see Sec. 5.4). In ongoing work, we are looking for smoothing the selection of

IS\_threshold. Further simulation results, which we could not include due to space constraints, show that for other packet rates hypergossiping provides a comparable performance.

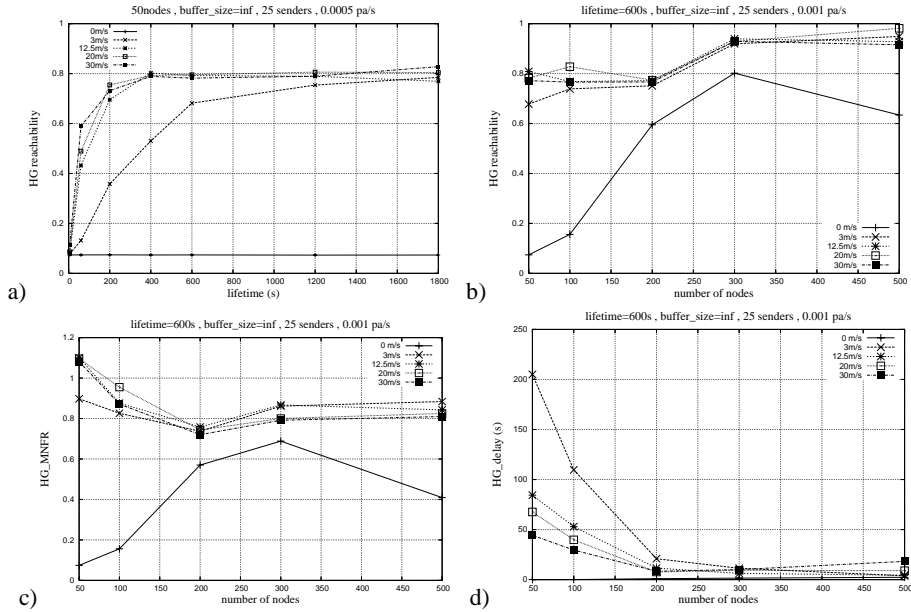


Fig. 5: Impact of lifetime, and node density and mobility.

## 5.6 Comparison to Related Work

Now we compare hypergossiping to the density-aware probabilistic flooding (PROB-FLOOD) [10], to the adaptive counter-based scheme (ACB) [9] and to the integrated flooding (IF) [5]. For this study we consider 25 senders that send at 0.001 packets/s.

For ACB we use the dynamic threshold given in [9]. ACB uses a random time span to count redundant packet receptions and possibly forwards the message after this span. This time period is comparable to  $fDelay$ . We choose the same value for these parameters, i.e. 10 ms, which is also used in [7].

For IF we use the following parameters (most of them are stated in [5]). In scoped flooding a node forwards a new received message, if at minimum 15% of its neighbors are not covered by the sender. We note here that scoped flooding needs 2-hop topology information, which means that nodes have to include their neighbor list in HELLO beacons. Hyperflooding holds packets for fixed time period and rebroadcasts them on discovering a new neighbor. Nodes install scoped flooding if their relative speed to all their neighbors is lower than 10 m/s. If the relative speed is higher than 25 m/s hyperflooding is selected. Otherwise plain flooding is deployed (message is forwarded exactly once by each node). We note that for max speed values until 12.5 m/s, nodes will never reach the higher switching threshold of IF (i.e. 25 m/s) and thus hyperflooding will never be installed in such configuration. For higher values of max

speed hyperflooding will possibly be installed by some nodes, which should increase the reachability of IF. For max speed equals 3 m/s the max relative node speed is 6 m/s, that is why only scoped flooding is installed by IF.

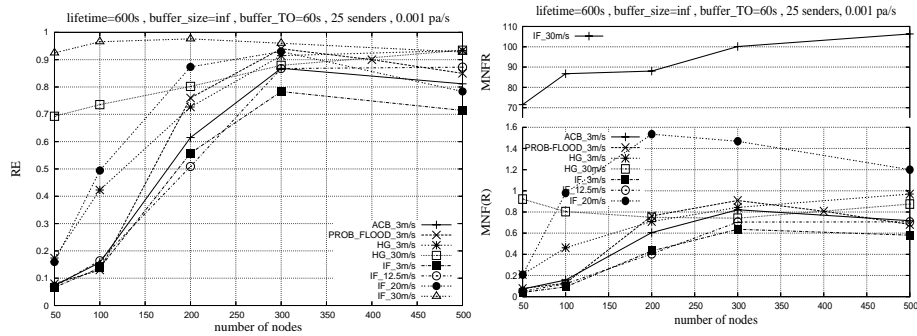


Fig. 6: Comparison to related work

For HG we use a lifetime value of 600 s and the same buffering strategy as IF. IF buffers all packets for a given fixed time, called *buffer\_TO*. We set the value of *buffer\_TO* to 60s. ACB and PROB-FLOOD show almost mobility-independent performance and thus we present for these protocols only results for 3m/s.

We can easily conclude from Fig. 6 that HG reachability outperforms PROB-FLOOD and ACB reachability for sparse networks and highly dense networks while keeping MNFR below 1. This is due to that HG remedies both causes of broadcast interruption: Network partitioning and broadcast storms.

Comparing HG and IF we conclude that IF does not provide an efficient partition detection strategy: IF provides namely a very high reachability for highly mobile MANETs, but MNFR ranges from 72 to 106 rebroadcasts per packet and node! (Simulation results show that even for very low *buffer\_TO* values, MNFR is very high for highly mobile scenarios: for *buffer\_TO*=5s and N=100 nodes, MNFR=11). For lower mobility IF installs only scoped flooding and subsequently performs similar to ACB and PROB-FLOOD for sparse networks but worse than these schemes for dense networks, lack of adaptation of scoped flooding to node density.

## 6 Conclusion and Future Work

In this paper, we introduced hypergossiping, an adaptive MANET broadcast algorithm, which presents our first steps towards a single broadcast solution for a wide range MANET operation conditions. Hypergossiping covers larger MANET densities and mobility levels. We have presented a novel method to adapt gossiping probability to node density and reduce the broadcast storms. Moreover, we presented a novel heuristic for repetition of broadcasts in order to overcome different causes of broadcast interruptions, such as network partitioning.

We are working on more sophisticated adaptation techniques of partition detection parameters. In future work, we will investigate different buffer management strategies to reduce buffer overhead. We expect that these strategies have to take into considera-

tion the residual lifetime of buffered packets and that adapting the buffer capacity to node mobility contributes to reduce the buffering overhead of hypergossiping.

## References

1. Z. Cheng, and W.B. Heinzelman, "Flooding Strategy for Target Discovery in Wireless Networks", Proceedings of MSWiM, 2003.
2. S.Y. Ni, Y.C. Tseng, Y.S. Chen, and J.P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network", Int. Conf. on Mobile Computing and Networking (MobiCom), 1999.
3. Z.J. Haas, J.Y. Halpern, and L. Li, "Gossip-Based Ad Hoc Routing", INFOCOM, 2002.
4. K. Obraczka, G. Tsudik, and K. Viswanath, "Pushing the Limits of Multicast in Ad Hoc Networks", Int. Conf. on Distributed Computing Systems (ICDCS), 2001.
5. K. Viswanath, and K. Obraczka, "An Adaptive Approach to Group Communications in Multi-Hop Ad Hoc Networks", Int. Conf. on Networking (ICN), 2002.
6. A. Ouksel, O. Wolfson, B. Xu, "Opportunistic Resource Exchange in Inter-vehicle Ad Hoc Networks", IEEE Int. Conf. on Mobile Data Management (MDM), 2004.
7. B. Williams and T. Camp, "Comparison of Broadcasting Techniques for Mobile Ad Hoc Networks", ACM MOBIHOC, 2002.
8. Y. Yi, M. Gerla, and T.J. Kwon, "Efficient Flooding in Ad hoc Networks: a Comparative Performance Study", IEEE Int. Conf. on Communications (ICC), 2003.
9. Y.C. Tseng, S.Y. Ni, and E.Y. Shih, "Adaptive Approaches to Relieving Broadcast Storms in a Wireless Multihop Mobile Ad Hoc Networks", IEEE Transactions on Computers, V52(5): 545–557, 2003.
10. J. Cartigny, and D. Simplot, "Border Node Retransmission Based Probabilistic Broadcast Protocols in Ad-Hoc Networks", Hawaii Int. Conf. on System Sciences (HICSS), 2003.
11. C. Ho, K. Obraczka, G. Tsudik and K. Viswanath, "Flooding for Reliable Multicast in Multi-Hop Ad Hoc Networks", Proceedings of DIALM, 1999.
12. S. McCanne, and S. Floyd, Ns Network Simulator, <http://www.isi.edu/nsnam/ns/>.
13. A. Khelil, C. Becker, J. Tian, and K. Rothermel, "Epidemic Model for Information Diffusion in MANETs", Proceedings of MSWiM, 2002.