Mobility-Aware Buffering for Delay-Tolerant Ad Hoc Broadcasting

Abdelmajid Khelil [*]
Technische Universität Darmstadt
Dependable, Embedded Systems and Software Group
Hochschulstr. 10, 64289 Darmstadt, Germany
Tel: (+49) 6151-16-3414, Fax: (+49) 6151-16-4310
khelil@informatik.tu-darmstadt.de

Keywords: MANET, Delay-Tolerant Networking, Broadcasting, Network Partitioning, Hypergossiping.

Abstract

Mobility plays a major role in mobile ad hoc networks (MANETs), since it stresses networking tasks such as routing on the one hand, but aids to increase the network capacity and to overcome network partitioning on the other hand. To benefit from node mobility, a new class of MANET protocols and applications are designed to be delay-tolerant and mobility-aided. The main communication paradigm here is store-and-forward. For delay-tolerant mobility-aided networking, mobility on a large time-scale is a key feature. So far however, a few work is done to adapt store-and-forward concepts to the large time-scale mobility.

Our first step that simplifies the adaptation to node mobility, is the set of novel mobility metrics presented in [1]. These metrics quantify the mobility on a large time-scale and are based on the pair-wise contacts between mobile nodes. In this paper, we show how to exploit these mobility metrics to design an efficient buffering strategy for hypergossiping [2,3], a delay-tolerant mobility-aided MANET broadcasting protocol. The novel buffering strategy detects relevant mobility patterns at run-time, using contact-based mobility metrics, and adapts the buffering decision to the detected mobility pattern.

INTRODUCTION

The number of mobile devices equipped with wireless network interfaces is continuously increasing. Many existing wireless technologies such as WLAN and Bluetooth provide besides an infrastructure-based communication mode an ad hoc communication mode. The ad hoc mode allows mobile devices to directly communicate if they enter each others communication range. If nodes can act as routers, multihop communication between nodes is possible. The so formed networks are referred to as Mobile Ad Hoc Networks (MANET). MANETs are suitable for scenarios where an infrastructure is very costly or even unavailable.

Mobility of network nodes stresses protocols and applications by disrupting routes, changing propagation effects and causing network partitioning. However, it can also be exploited to *increase network capacity* [4] and to *overcome network partitioning*.

To profit from these mobility benefits, a new challenging class of *mobility-aided* applications and protocols have been recently developed. These protocols and applications tolerate higher communication delays. They are grouped under the delay-tolerant networking research field (see the Delay Tolerant Networking Research Group (DTNG) [5] [6] [7] [8] [9]). The communication in delay-tolerant networking architecture basically relies on asynchronous, store-and-forward message delivery. This communication paradigm does not assume a contemporaneous end-to-end connectivity; (some) mobile nodes have to physically transport data from source to destinations.

Because node mobility plays an important role in realizing mobility-aided systems, we designed a set of novel mobility metrics that quantify the mobility of nodes on a large time scale, i.e. for time periods in the range of minutes, hours, or even days. These metric should simplify the design and the adaptation of delay-tolerant mobility-aided protocols and applications. In this paper, we show how we use these metrics to improve the performance of one delay-tolerant and mobility-aided broadcast protocol, called hypergossiping [2, 3].

The remainder of this paper is organized as follows. In Section 2, we define key terms, present our system model, and briefly introduce hypergossiping. Using one real-world scenario, we present in Section 3 the problem and our objectives. In Section 4, we present our approach to reduce the buffering overhead of hypergossiping using some contact-based mobility metrics. In Section 5, we calibrate some parameters and evaluate our work using simulations. Section 6 discusses related work. Finally, Section 7 concludes the paper.

PRELIMINARIES

In this section, we first define our system model. We then review our novel contact-based mobility metrics that we have defined and investigated in detail in [1]. Finally, we briefly introduce hypergossiping, our generalized broadcast strategy for broadcasting in mobile ad hoc networks [2, 3].

---

[*] This work has been partly supported by the European IST project DECOS (EC IST FP6 IP DECOS) No. 511764.

System Model

In this work, we consider MANETs that are formed by N mobile nodes. We assume nodes have no knowledge about their position or speed. We assume in this work that nodes neither crash nor run out of energy. We assume that nodes are uniquely identified, e.g. using their Medium Access Control (MAC) addresses. Nodes move according to an arbitrary mobility model. 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 like in [10].

Nodes in this paper are of similar communication capabilities (communication range and bandwidth) and can only communicate if their sight distance is below the communication range R. Nodes are also of similar storage capabilities.

Broadcast data has typically a temporal and spatial relevance [11]. 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.

Contact-Based Mobility Metrics

In the following we first define the important terms encounter and contact. Then, we briefly review the contact-based mobility metrics, which are defined and investigated in detail in [1].

We say that two nodes *encounter* each other when the distance inbetween becomes smaller than the communication range R. The encounter is said to be lost, if the nodes leave the communication range of each other. We represent an encounter by the two nodes, its time of incidence and its duration. We define a *contact* between two nodes as the list of all encounters between them. A contact between two nodes begins with the first encounter between them, and ends with the last one. We assume nodes manage a history of their encounters in a local *contact table* during a fixed period of time that we refer to as observation interval, or $obs-interval$ for short.

The *Average Encounter Rate (AER)* is defined as the number of new encounters experienced by a node per unit of time. We define the *Average Contact Rate (ACR)* as the number of new contacts experienced by a node per unit of time. Both these metrics quantify the relative mobility between nodes.

The *Average Encounter Frequency (AEF)* is the average number of encounters per contact. It quantifies the mixture between nodes, i.e. how often two nodes encounter each other per unit of time.

We also defined two contact-based metrics based on the encounter and contact durations: The *Average Encounter Duration (AED)* and the *Average Contact Duration (ACD)*. Both metrics quantify the neighborhood stability of nodes.

Contact-based metrics can be defined either at the network level or at the node level. At the network level, this information helps to understand the mixture of the population. If the information is node-centric it describes the relative mobility of that node to the other nodes.

Contact-based metrics may be used at design-time to simplify the evaluation of delay-tolerant protocols and applications, but also at run-time to adapt these protocols and applications to node mobility.

Network-wide metrics are metrics that are not easy to compute at run-time, since they need a high communication overhead. Network-wide metrics are appropriate at the design stage and should be used by developers to well design their protocols for a wide range of mobilities.

Node-centric metrics are easy to acquire at run-time. Encounters can be perceived using a simple neighbor discovery protocol such as HELLO beaconing. These metrics can be then used to adapt protocols and application on-the-fly.

Hypergossiping (HG)

Hypergossiping [2, 3] is a representative of the class of delay-tolerant mobility-aided ad hoc protocols.

Hypergossiping is a generalized MANET broadcasting protocol that considers network partitioning in order to increase the delivery reliability of gossiping in sparse networks. Hypergossiping deploys a heuristic to detect partition joins and rebroadcasts the appropriate messages from buffer on partition join detection.

Hypergossiping combines two strategies to distribute messages to all nodes. The first strategy is called *gossiping* (probabilistic flooding) and aims at efficient distribution of messages within the same partition. The second strategy is called *broadcast repetition* and aims at overcoming network partitioning. Hypergossiping buffers messages and rebroadcasts them on partition joins. For this, hypergossiping utilizes a *partition join detection heuristic* to detect partition joins and a *rebroadcasting protocol* to send the appropriate messages.

The partition join detection heuristic works as follows. Nodes share with their new neighbors an ID-list of Last Broadcast Received (LBR list). If a node encounters a node that has a sufficiently different LBR it assumes that a partition join has just occurred and triggers the rebroadcasting protocol.

The rebroadcasting protocol first sends the complete ID-list of Broadcasts Received (BR list). Nodes receiving this BR list have then to rebroadcast messages from their buffer that have not yet received by the sender of the BR list.

PROBLEM STATEMENT AND OBJECTIVES

In the following, we state the problem and mention our objectives towards more efficient hypergossiping buffering. We investigate one real-world scenario, from which we identify certain mobility patterns that are relevant to mobility-aided information dissemination.

As stated before, for delay-tolerant mobility-aided ad hoc networking mobility plays a major role in data transport. Nodes buffer messages and monitor the network condition, in order to retransmit them whenever the destination becomes more easily reachable.

So far, hypergossiping uses the following simple buffering strategy. The protocol allows all nodes to buffer all received messages as long as they are still relevant, i.e. for the residual lifetime. This strategy unfortunately may produce a high buffering overhead, which is not practical for devices with limited resources and for applications that initiate a lot of broadcasts with high lifetimes. Therefore, we need new strategies that reduce the buffer overhead of hypergossiping. Our over-all objective is to design a buffering strategy with lower demands on buffer space, and without or with low loss of reachability.

Although the random waypoint mobility model is widely used for evaluation of MANET protocols, real users are not likely to move around randomly, but rather move in a realistic fashion based on the correlation between nodes or repeating behavioral patterns. Some users such as soldiers and rescuers tend to move in groups. Others such as busses and trains tend to repeat their movements. However, there are also singular nodes that behave unpredictably. Without loss of generality, we can consider a MANET as a set of node groups that meet and leave over time. Using the example of random waypoint, groups are constituted of single nodes.

We are convinced that mobility characteristics of nodes play a major role to design store-and-forward mechanisms in mobility-aided networking. At the example of mobility-aided delay-tolerant information dissemination, such as hypergossiping, mobility patterns are crucial for developing efficient buffering strategies. Since we are dealing with highly diverse mobility patterns in MANETs, that may change over scenarios or over time within the same scenario, we also believe that the perception of relevant mobility characteristics is a major factor for the adaptation of store-and-forward mechanisms at run-time.

Two examples of mobility patterns, from which buffering strategies could benefit, are examined in the following. Firstly, nodes moving in a group can cooperate concerning message buffering. The group can "select" one of its members to buffer the message. This can reduce the total number of nodes buffering the message during its lifetime, and therefore reduce the number of nodes buffering a certain message. Secondly, nodes that constantly encounter new nodes (e.g. racer cars on the highway) should buffer more messages than other nodes, that rarely encounter new nodes.

	staff	students
encounter rate	low	high
encounter duration	high	low
encounter frequency	≈ 1	≈ 1
contact rate	low	high
contact duration	high	low

Table 1: Campus scenario

It is challenging for mobile nodes to detect such patterns at run-time. Since these patterns have a large time scale, we are convinced that contact-based metrics are valuable for recognizing such patterns and for detecting them at run-time.

In the following we motivate, using the example of a campus scenario, the idea of mobility-aware buffering and the major role that contact-based mobility metrics can play for the perception of relevant mobility patterns.

Consider the MANET formed by mobile devices carried by students and staff members on a campus during working hours. Firstly, we qualitatively analyze the properties of the mobility of nodes on a large time scale using our contact-based metrics. Secondly, we show the usefulness of these metrics for delay-tolerant information dissemination in the considered MANET.

Staff members are generally grouped into departments, which in their turn are grouped in faculties. Offices for one department or one faculty are normally grouped geographically. Staff members work most of time in their offices, and sometimes meet each other. We assume that nodes located in different campus buildings can not communicate directly with each other. Also we assume that not all nodes located in the same building can communicate directly. From these observations, we can conclude that mobile devices carried by staff members form a relatively stable network topology. Thus they show low encounter and contact rates but high encounter and contact durations.

Students commute frequently between departments, faculties, classrooms, libraries and cafeterias. Therefore, their encounter and contact rates are higher, but their contact and encounter durations are lower than that of staff members. Table 1 shows a qualitative analysis of the contact-based metrics for both groups.

We suggest the following two simple heuristics to reduce buffer overhead. Firstly, students and not staff members have to buffer messages, since students commute more probably between different network partitions. Therefore, students are more suitable as a transport mechanism between partitions. From Table 1 hypergossiping can easily approximate, whether a node is suitable for buffering the message. Nodes with higher contact rate should be chosen. Secondly, nodes moving in a group intuitively do not all have to buffer the same message. However group members should cooperate to buffer messages.

MOBILITY-AWARE PROBABILISTIC BUFFERING

In order to exploit the previous observations on mobility characteristics of real users with the purpose of reducing the buffering overhead of hypergossiping, we propose a new challenging buffering strategy for this protocol. Our approach uses contact-based mobility metrics to compute the utility of buffering a certain broadcast message for future rebroadcasting.

Overview

As stated before, our objective is to limit the number of nodes buffering a given message without reducing the delivery ratio too much.

As assumed in the system model, nodes are of similar storage capabilities. Therefore, we do not select the buffering nodes with respect to the node storage capabilities. In this paper, we only consider the nodes' mobility pattern for the selection of buffering nodes. In MANETs that are composed of nodes of heterogeneous capabilities we propose to also use other information such as available memory, CPU power, and available battery energy to differentiate nodes, while defining the utility of a certain node to buffer a certain message. For example, we should use laptops instead of PDAs if both can be used to buffer a message.

Our approach attempts to allow the node, which is most likely to deliver the message and with lowest cost, to buffer the messages. We use utility metrics to try to determine the node most appropriate to buffer the message and to deliver it to the destination. Since we differentiate nodes only with regard to their movement patterns, we should analyze the patterns most relevant for buffering and design efficient concepts to detect these patterns at run-time. Afterwards, we should define metrics for the selection of the appropriate nodes with regards to their movement patterns.

Our approach is based on two components. The first component efficiently detects two movement patterns relevant for the store-and-forward mechanism of hypergossiping. We realize this component based on particular contact-based mobility metrics. The second component is used by nodes to compute the buffering utility for each newly received message. This component defines a buffering utility and provides a method for computing it, depending on the mobility pattern perceived by the first component.

Detecting the Relevant Mobility Patterns

In the following, we present our approach, which uses the contact-based metrics for an efficient detection of the mobility patterns relevant for the reduction of the buffering overhead of hypergossiping.

Relevant Mobility Patterns:

As stated before, we consider a MANET as a set of node groups that meet and leave over time. Some real-world scenarios, such as the campus scenario, confirm this view of the MANET. If we consider a node moving in isolation as a one-node group, we can transform every mobility model, where nodes move uncorrelated, to a group mobility model. Thus, by appropriate choice of parameters, existing mobility models, such as random waypoint and graph-based, can be considered as group mobility models.

While investigating the campus scenario, we showed that nodes moving in a group or nodes roaming very frequently between different groups may play a particular role for store-and-forward information dissemination. Therefore, we are looking to exploit these movement patterns to reduce the buffering overhead of hypergossiping.

The *group movement pattern* helps the group members to share the buffering task. Our approach is to let nodes moving in a group *cooperate* in order to select nodes that buffer a certain message.

Nodes showing a *roaming pattern* are good candidates to buffer messages and transfer the broadcast from one partition to the next. Roaming nodes in our campus scenario are mainly the students.

The store-and-forward feature of hypergossiping takes place on a large time scale (given by the lifetime of data). Subsequently, the mobility patterns that are relevant for hypergossiping should be of a similar time-scale. Since our contact-based metrics model the mobility on a large-time scale, we are convinced that these metrics are valuable for capturing the group movement and roaming movement patterns.

Pattern Detection:

In this section we show how we deploy contact-based metrics to characterize the mobility patterns mentioned above.

Group Movement Detection: A set of nodes moving in a group is a set of nodes, which show very correlated movements. At the macroscopic-level group members show a certain physical proximity. According to this, we define a group as follows:

Definition: Two mobile nodes A and B belong to a group, over a time interval $[t_1, t_2]$, if they show a strong correlation between their position coordinates during this time interval.

The group members are identified based on a time window w, i.e. the length of the time interval $[t_1, t_2]$ (e.g. 5 minutes).

We assume that the groups are not known in advance and that they can form dynamically. We therefore need mechanisms to discover groups in the MANET.

We assume that the geographical proximity of group members is within the scale of the communication range or lower. Therefore, we expect that group members are within the communication range of each other, most of time. We also tolerate group members leaving each others communication range for a short time period and then encountering each other again. Fig. 1 outlines a contact between node A and node B that consists of three encounters. It is obvious on the macroscopic-level that nodes A and B move in a group. Observing the contact-based metrics of nodes moving in a group we can expect them to have long encounter durations (subsequently also higher contact duration), or short encounter duration but a long *contact duration* with the other group members.

Thus, we define two nodes as moving in a group, if they show a long contact duration.

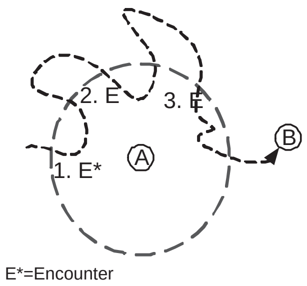


Figure 1: Detection of nodes moving in a group

In this way, one node is able to detect neighbors that move with it in a group. Again, the length of the contact history should be set appropriately (depending on the application). If two nodes encounter each other for some minutes and then leave, should we consider this encounter as a group movement or not? This decision depends on the time scale of the information dissemination process, which in its turn depends on the delay-tolerance of the broadcast application. We modeled the delay tolerated by the application using the so-called lifetime of broadcast data. Considering the information dissemination using hypergossiping, we conclude from the observations above that the contact history time depends on the lifetime of broadcast data. In the evaluation section, we calibrate the length of the contact history.

The higher the current encounter duration of a certain node with a certain neighbor, the higher the probability that both nodes are moving in the same group. In this paper, we simply assume that, if two nodes have a contact duration higher than 80% of contact history time, then both nodes are moving in a group.

Roaming Movement Detection: Nodes that frequently move between groups constantly encounter new nodes. Considering the contact-based metrics for these nodes, we expect that these node are characterized by high contact rates.

Buffering Utility

Our approach for buffering is utility-based. Each node defines a utility for each received message. The utility presents a metric for the relevance of buffering the message. The higher the utility, the more useful the buffering.

To this end, we define a utility metric that we call *buffering utility*, $u(n,m_i) \in [0,1]$, at every node n and for each broadcast message m_i received by that node. This indicates how useful it is for the node n to buffer the message m_i for broadcast repetition. If a node receives a broadcast message for the first time, it computes the buffering utility of that message and buffers it with a probability equal to the utility value $p(n,m_i) = u(n,m_i)$ and for a time period equal to the message's residual lifetime.

This strategy offers the following two advantages. Firstly, the buffering of the messages will be shared equally between nodes with the same utility value. This increases the fairness of buffering. Secondly, the strategy is completely decentralized, since there is no need for coordination between nodes, in order to determine which node buffers which message.

The buffering utility has to be updated by a node according to the mobility patterns detected by that node. Since we aim to consider two mobility patterns, we superimpose two components to calculate the buffering utility. The first part is to update the metric depending on the group movement pattern. The second is to update the utility with respect to the roaming degree of the node.

Group-Based Buffering:

We propose the cooperation of nodes moving in a group and suggest the following approach. We assume that nodes currently moving in a group are also likely to remain in the same group. Nodes belonging to the same group should cooperate in order to share the buffering of messages. The subsequent step is to fix which node has to buffer the messages. Clustering and centralized coordination to distribute the buffering task is one approach that we should avoid since it produces high message overhead, especially in highly mobile networks, where the clustering algorithms have to be run more frequently.

In the following, we consider a set of nodes moving in a group. We denote this set of nodes as G. We denote the number of the members of group G as $|G|$. The utility should be defined inversely proportional to the number of group members. As an example, we deploy the simple function $1/x$. We propose to compute the group-based buffering utility as shown in Eq (1).

$$u_{group}(n,m_i) = \frac{1}{|G|} \quad (1)$$

It is obvious that $u_{group}(n,m_i) \in [0,1]$ since $|G| \geq 1$. Please note that we use the same utility for all messages, since we are only considering the mobility characteristics. We then set the probability for probabilistic buffering equal to the utility as shown in Eq (2).

$$p_{group}(n,m_i) = u_{group}(n,m_i) = \frac{1}{|G|} \quad (2)$$

The probability that the group of nodes G fails to buffer the message is shown in Eq (3), Eq (4) and Eq (5).

$$p_e = \prod_{n \in G}(1 - p_{group}(n,m_i)) \quad (3)$$

$$p_e = \prod_{n \in G}(1 - \frac{1}{|G|}) = (1 - \frac{1}{|G|})^{|G|}) \quad (4)$$

For large groups, p_e is on average equal to:

$$p_e \approx 1/e \approx 1/2.72 \approx 0.37 \approx 37\% \quad (5)$$

The probability of successful message buffering (p_s) is given in Eq (6).

$$p_s = 1 - p_e \approx 1 - 1/e \approx 0.63 \approx 63\% \quad (6)$$

Using this simple calculation we recommend increasing the buffering utility and probability by introducing an efficiency parameter k as follows:

$$u_{group_2}(n,m_i) = p_{group_2}(n,m_i) = min\{\frac{k}{|G|}, 1.0\} \quad (7)$$

The probability of buffering failure is then $p_e \approx e^{(-k)}$. If $k \geq |G|$ all nodes buffer and we get the performance of the original algorithm. In the evaluation section we vary the value of k and calibrate it.

Roaming-Based Buffering:

Cross-moving nodes that roam between different groups are expected to show very low encounter frequencies (≈ 1). These nodes should buffer more messages than other nodes. Therefore, the utility should be defined inversely proportional to the encounter frequency. As an example, we deploy the simple function $1/x$.

$$u_{roam}(n,m_i) = \frac{1}{AEF} = \frac{ACR}{AER} \quad (8)$$

Again AEF, ACR and AER are the node's Average Encounter Frequency, Contact Rate and Encounter Rate respectively.

Similar to the group-based buffering, we set the roaming probability as depicted in Eq (9).

$$p_{roam}(n,m_i) = u_{roam}(n,m_i) = \frac{1}{AEF} \quad (9)$$

Please note that $p_{roam}(n,m_i) \in [0,1]$ since $AEF \geq 1$.

Integrated Probabilistic Buffering:

In general, different factors such as node capabilities, MANET characteristics and message properties may impact the buffering decision of nodes. To consider different factors, we propose that nodes define different utilities for different factors and to superimpose these utilities depending on their relevance for buffering.

In this work, we investigate only mobility patterns and exactly two mobility patterns; group motion and roaming patterns. Therefore, we consider both utilities for a buffering decision, i.e. group-based and roaming-based utilities. Hence, we should weight each probability and use the average for buffering decision (Eq. 10).

$$p(n,m_i) = \alpha * p_{group}(n,m_i) + (1-\alpha) * p_{roam}(n,m_i) \quad (10)$$

The appropriate value of α depends on the decision, which mobility pattern roaming or group-motion is more appropriate for reducing buffer overhead. In the remainder of this work, we only focus on group movement patterns as examples, and therefore select $\alpha = 1$.

PERFORMANCE EVALUATION

In this section, we first introduce the simulation model and the performance metrics. Then, we calibrate the length of the contact history and the efficiency buffering parameter k. Finally, we present the performance of hypergossiping with mobility-aware buffering and compare it to the performance of the original algorithm.

Simulation Model

We generate N mobile nodes in a 1000mx1000m field, where these nodes move according to an arbitrary mobility model. The mobility models that we consider in this paper are the RPGM group mobility model [12], the random waypoint (rw) model [13] and the graph-based mobility model [14]. For all mobility models, we vary the nodal speed between 0 and a maximum speed value, and select a pause time uniformly between 0 and 2s. For the RPGM mobility model, nodes are generated in groups consisting of 10 ± 5 nodes. The group members are also geographically grouped, since nodes are located at a maximum of 10m away from the group center. Table summarizes the simulation parameters of our experiments.

Parameters	Value(s)
Sim. area	1000m x 1000m
Num. of nodes	$N \in [50, 300]$
Comm. range	R = 100m
Bandwidth	r = 1 Mbit/s
Message size	280 bytes
Movement	$\in$ {rw, rpgm, graph-based}
Lifetime	600 s
Sim. time	obs-interval+lifetime+20s

Table 2: Simulation parameters

We use a random HELLO-beaconing period between 0.75s and 1.25s. A neighbor is removed from the neighbor list if during 2s no beacon is received from this neighbor. Before initiating hypergossiping, we run a warm-up phase for a period of time equal to the obs-interval. This allows nodes to have a complete history of contacts, before starting hypergossiping.

We use the following broadcast traffic model. 25 nodes initiate broadcasting at a random time between 1 and 3 seconds after the warm-up phase. Broadcast messages remain relevant during their lifetime. For the same simulation scenario we ran 5 passes with 5 different movement traces and considered the average.

Performance Metrics

For the evaluation of hypergossiping, we use the following metrics:

- *REachability (RE)*: the ratio of mobile nodes receiving the message to the total number of mobile nodes. This metric measures the delivery reliability of the broadcast algorithm.
- *Delay*: Average end-to-end delay over all receivers.
- *MNFR*: Mean Number of Forwards and Rebroadcasts per node and message. MNFR measures the efficiency of the broadcast algorithm.

For the evaluation of the buffering strategy, we define the following evaluation metrics:

- *BUFF-ratio*: is the ratio of nodes buffering a given message to the total number of nodes that have received this message. We note that BUFF-ratio $\in [0, 1]$.
- *Average number of encounters*: The major additional overhead to perceive contact-based mobility metrics is the storage overhead for the contact table. This is simply given by the number of encounters forming the table. For each encounter we need 4 bytes to store the encounter ID (e.g. MAC address), 4 bytes for its time of incidence and 4 bytes for its duration. Therefore, we need 12 bytes in total, for each encounter.

Calibration

Contact-based group detection, as well as the probabilistic buffering strategy have some parameters that still have to be calibrated, i.e. the observation time period ($obs-interval$) and the efficiency parameter for buffering (k).

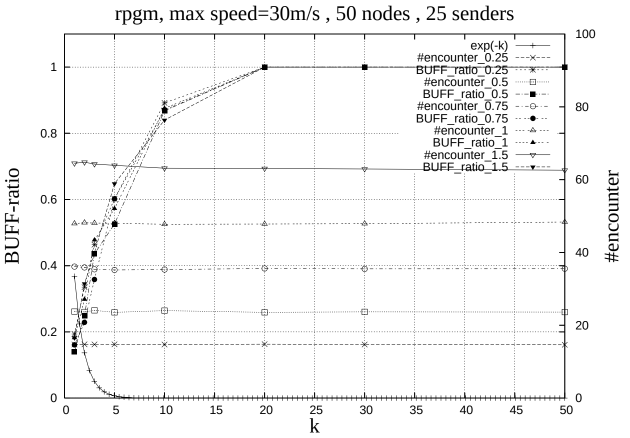
As stated before, the contact history depends on how the application defines a set of nodes to be moving in a group. Since our protocol acts on a time-scale given by the lifetime of broadcast data, we fix the obs-interval depending on the lifetime. We set $obs-interval = obs-scale * lifetime$ and vary obs-scale for calibration. The obs-scale parameter impacts the size of the contact table and the the groups detected. As shown in Fig. 2 a), the average number of encounters per contact table increases if obs-scale increases. Hence, for calibration, we have to minimize the obs-scale value, while keeping the groups detected correctly from the point of view of the application.

The efficiency parameter k impacts the ratio of group members that decide to buffer the message. k also determines the buffering failure, i.e. no group member buffers the message. For calibration, we should minimize k, to reduce as much buffering overhead as possible, but we also have to minimize buffering failures, so that the RE of hypergossiping will not break. According to the error function (Fig. 2 a)), if we tolerate buffering failures of 5% or lower, we have to choose $k \geq 3$.

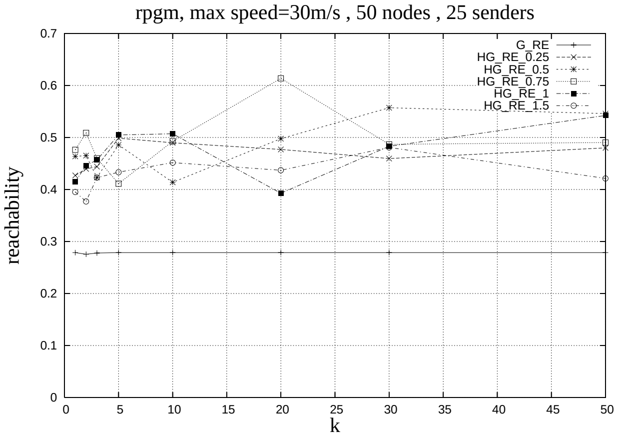
We use the RPGM mobility model for the calibration process. So far, we assumed that the group's geographic proximity is within the range of the communication range. For our settings for the RPGM mobility model, we expect a maximum distance of 20m between two group members. Since we set the communication range to 100m, the assumption above holds. However, if the communication range is lower than 20m, our group detection strategy may not detect some group members. This leads to a higher buffering utility, which increases the number of nodes buffering a certain broadcast message.

From Fig. 2 a), we conclude that for $k = 3$ the buffering overhead of hypergossiping can be reduced by approximately 50%-64%. We therefore select $k = 3$ for our buffering strategy.

After calibration of k, we now calibrate $obs-scale$. The investigation of the reachability of hypergossiping (Fig. 2 b)) shows that probabilistic buffering introduces some oscillations. Due to the probabilistic nature of buffering, the nodes that buffer the message change from one simulation run to another. This may lead the group members that detect a partition join not being those that buffer the appropriate messages.



(a) Encounter number and BUFF-ratio



(b) Impact of k and obs-scale on RE

Figure 2: Calibration of probabilistic buffering strategy

However the impact of k and the $obs-scale$ on the reachability of hypergossiping is not clear. Therefore, the calibration of obs-scale can not be done using this simulation set.

For the calibration of $obs-scale$ we proceed as follows. Intuitively, a message that is received by a node has a residual lifetime equal, on average to half of the lifetime value set by the source of the message. If the node decides to buffer the message, it will buffer it, on average, for the residual half lifetime value. Subsequently, a node has to detect the nodes, with which it has been moving in last half lifetime period and assume that this group will hold for the next half lifetime period. From this observation we propose to use $obs-scale = 0.5$ for the buffering strategy.

Simulation Results

In this section, we present the performance of hypergossiping. In this study, we vary the mobility models, and arbitrarily set the maximum speed of nodes to 3m/s. We set the lifetime of broadcast data to 600s. Nodes maintain contact tables for 300s.

In Fig. 3, we observe that for the random waypoint our strategy does not detect any motion groups and therefore does not reduce the number of nodes buffering a certain broadcast message. Using the graph-based mobility model our strategy detects a few groups for a higher number of nodes and therefore saves some bufferings (about 5% for N=300). Since almost all receivers buffer the message for both random waypoint and graph-based models, the performance of hypergossiping (reachability, MNFR and delay) using probabilistic buffering is very close to the performance hypergossiping without probabilistic buffering (Fig. 4 a) b) and c)).

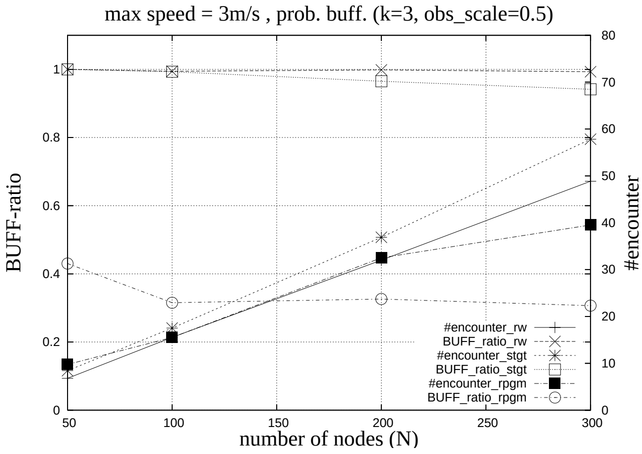
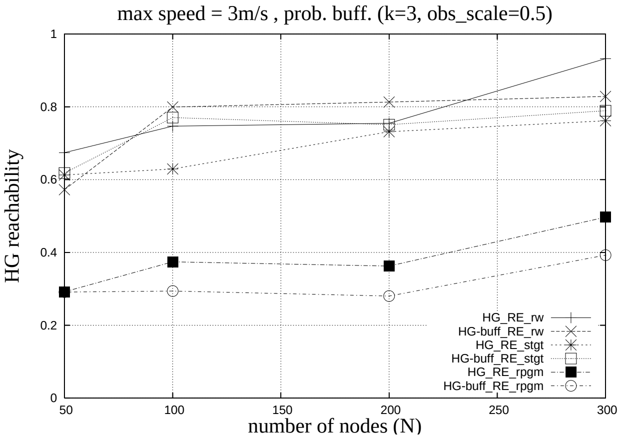


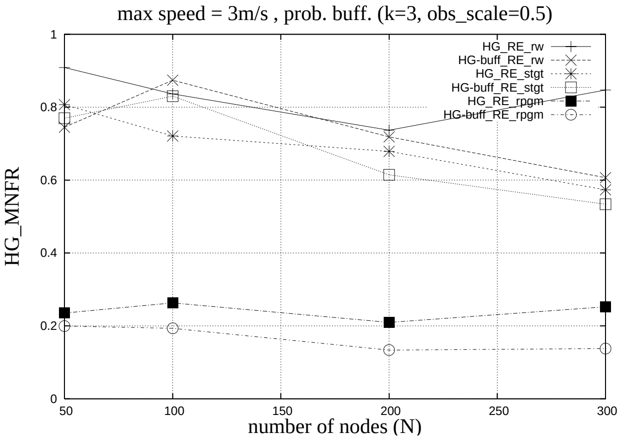
Figure 3: BUFF ratio and number of encounters

The RPGM model shows inherently more motion groups. Our strategy detects many of the groups and prohibits between 63% (N=50 nodes) and 70% (N=300 nodes) of receivers from buffering the broadcast message. As a result the number of broadcast repetitions and therefore MNFR slightly decreases (Fig. 4 b)). Subsequently, the reachability of hypergossiping with probabilistic buffering (Fig. 4 a)) decreases compared to the case without probabilistic buffering. The average end-to-end delay also slightly decreases, since the total number of reached nodes has decreased (Fig. 4 c)).

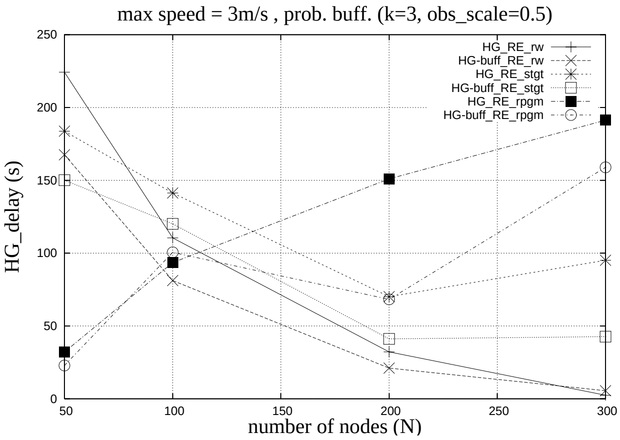
For the overhead caused by the management of contact tables, we note that HELLO beaconing is needed by the hypergossiping protocol anyway. The main additional overhead is therefore the storage overhead for the contact tables. The maximum table size needed, would be for N=300 nodes and for the graph-based mobility model, i.e. 58 encounters (Fig. 3). This implies a storage overhead of 58*12 byte, which



(a) Reachability



(b) MNFR



(c) Delay

Figure 4: Performance of HG with probabilistic buffering

is equivalent to the buffering overhead of 2.5 messages. This demonstrates the limitation of the additional overhead induced by the maintenance of contact-based information. Although, this overhead will be higher for higher obs-interval values and for higher node speeds, we are convinced, that this overhead remains tolerable, compared to the buffer space gained. We note also that this overhead remains constant for higher broadcast traffic, where the saved buffer overhead increases. Furthermore, contact-based mobility information can be used for protocols and applications.

RELATED WORK

In [15, 16], the authors pointed out that the fundamental characteristics of group mobility is the similarity of the velocity. According to this, the authors presented a localized method to detect group members by sharing velocity information with neighbors. Since this approach relies on velocity information, the approach has strong limitations regarding its use in application scenarios. As we are designing a generalized solution for MANETs, we will not consider this approach further.

In [17], the authors proposed a scheme to detect the presence of groups among the nodes of a network by performing a correlation index test on the mobility traces. The method assumes a global view (position of all nodes over the time-interval considered) and is, therefore, unapplicable for our purposes.

In [18] the authors presented a group discovery method that assumes the existence of an up-to-date routing table, which in turn assumes the availability of a proactive routing protocol. Proactive routing protocols are, however, only appropriate for one class of MANETs, i.e. not highly dynamic, and therefore the group discovery strategy is only applicable for some MANET scenarios and not for others. Hence the strategy is not applicable for our generalized dissemination strategy.

CONCLUSION AND FUTURE WORK

In this paper, we have shown the importance of detecting relevant mobility patterns for successful design and adaptation of delay-tolerant mobility-aided ad hoc protocols.

We have demonstrated how contact-based mobility metrics can help developers to design and adapt delay-tolerant ad hoc protocols and applications. Using these metrics we could detect nodes moving in a group and then adapt the buffering strategy of hypergossiping, one delay-tolerant mobility-aided broadcast protocol, to node mobility.

In future work, we propose to consider further relevant features, such as node capabilities or message properties, while computing the buffering utility and the establishment of the integrated utility-based buffering.

REFERENCES

[1] Abdelmajid Khelil, Pedro José Marrón, and Kurt Rothermel. Contact-based mobility metrics for delay-tolerant ad hoc networking. In *Proceedings of The 13th IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, September 2005.

[2] Abdelmajid Khelil, Pedro José Marrón, Christian Becker, and Kurt Rothermel. Hypergossiping: A generalized broadcast strategy for mobile ad hoc networks. In *Proceedings of The 2005 Conference on Communication in Distributed Systems (KiVS)*, February 2005.

[3] Abdelmajid Khelil, Pedro José Marrón, Christian Becker, and Kurt Rothermel. Hypergossiping: A generalized broadcast strategy for mobile ad hoc networks. In *Elsevier Ad Hoc Networks Journal (to appear)*, 2006.

[4] M. Grossglauser and D. Tse. Mobility increases the capacity of ad hoc wireless networks. IEEE/ACM Transactions on Networking, 10(4), pages 388-404, August 2002.

[5] The Delay Tolerant Networking Research Group. dtnrg.

[6] R. Patra S. Jain, K. Fall. Routing in a delay tolerant networking. In *Proc. of SIGCOMM'04*, Aug/Sep 2004.

[7] K. Fall. A delay-tolerant network architecture for challenged internets. In *Proc. of SIGCOMM'03*, September 2003.

[8] V. Cerf et al. Delay-tolerant network architecture. In *Internet draft, draft-irtf-dtnrg-arch-02.txt*, March 2003.

[9] V. Cerf et. al. Delay tolerant network architecture. In *draft-irtf-dtnrg-arch-02.txt*, July 2004.

[10] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), Tokyo, Japan*, May 2004.

[11] A. Ouksel, O. Wolfson, and B. Xu. Opportunistic resource exchange in inter-vehicle ad hoc networks. In *Proceedings of the IEEE International Conference on Mobile Data Management (MDM), Berkeley, USA*, January 2004.

[12] X. Hong, M. Gerla, R. Bagrodia, and G. Pei. A group mobility model for ad hoc wireless networks. In *Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems (MSWiM'99)*, pages 53–60, August 1999.

[13] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, 1998.

[14] J. Tian, J. Haehner, C. Becker, I. Stepanov, and K. Rothermel. Graph-based mobility model for mobile ad hoc network simulation. In *Proceedings of 35th Annual Simulation Symposium (ANSS 35)*, April 2002.

[15] Karen Wang and Baochun Li. Group mobility and partition prediction in wireless ad-hoc networks. In *Proceedings of IEEE International Conference on Communications (ICC 2002)*, pages 1017–1021, April 2002.

[16] Karen Wang and Baochun Li. Efficient and guaranteed service coverage in partitionable mobile ad-hoc networks. In *Proc. of IEEE Joint Conference of Computer and Communication Societies (INFOCOM 2002)*, pages 1089–1098, June 2002.

[17] Z. R. Zaidi, B. L. Mark, , and Roshan K. Thomas. A two-tier representation of node mobility in ad hoc networks. In *Proc. IEEE Sensor and Ad Hoc Communications and Networks (SECON'2004)*, October 2004.

[18] Xiaoyan Hong and Mario Gerla. Dynamic group discovery and routing in ad hoc networks. In *Proceedings of the First Annual Mediterranean Ad Hoc Networking Workshop (Med-hoc-Net 2002)*, September 2002.