UNIVERSITÀ DEGLI STUDI DI MILANO
Dipartimento di Informatica e Comunicazione

RAPPORTO INTERNO Nº RT 20-07

Epidemic Diffusion of Data in Opportunistic Networks

Francesco Giudici, Elena Pagani,
Gian Paolo Rossi

Epidemic diffusion of data in opportunistic networks

Francesco Giudici, Elena Pagani, Gian Paolo Rossi
Information Science and Communication Department
Università degli Studi di Milano, Italy
Email: {fgiudici,pagani,rossi}@dico.unimi.it

***Abstract*—Opportunistic networks have interesting communication behavior that could enable to bring ad hoc networks to people's everyday life. To allow this to happen, several efforts are still required to provide routing strategies capable to efficiently cope with the highly variable network topology and the need to tolerate temporary network partitions. This paper focuses on this argument and proposes the use of epidemic algorithms to provide best effort data delivery to all nodes in an opportunistic scenario. The main contribution of the paper is to show that the approach actually has the ability to deal with the mentioned constraints and outperforms flooding to diffuse packets over the network. The paper presents the first set of simulation results and highlights the parameters that influence the performances of the protocol. These results represent a sort of worst case analysis of the approach and are the starting point to design an adaptive diffusion protocol which is capable to adapt to both the dynamics of the network and the mobility conditions.**

I. Introduction

Opportunistic networks, ONs, are delay tolerant networks in which the wireless connectivity for data diffusion and gathering can be established by exploiting the *contact* opportunity of another radio device in proximity. If the proper contact is not available, data is cached locally waiting for an opportunity of forwarding that mobility will eventually create. In such a wireless infrastructure, each mobile node plays a threefold role: (i) *end system* of one-to-one or one-to-many communications, (ii) packet *forwarder*, when the proper *contact* is met, (iii) packet *transporter* of cached packets, when the network is partitioned. Although opportunistic networks have several characteristics in common with ad-hoc networks, their specific behavior hinder borrowing ad-hoc protocols for the new needs. In particular, the highly variable network topology and the need to tolerate temporary network partitions (due to the fact that nodes can be temporarily out of range) heavily influence the design of routing protocols over opportunistic networks. The network dynamics force to adopt totally stateless and distributed approaches to routing, while the tolerance to network partitions can only be achieved through a proper combined exploitation of caching packets and node mobility. This paper considers the routing over opportunistic networks to support communication schemes of the form one-to-all or one-to-many, where with the term *many* we refer to a dense membership of a group G. The basic idea of this work is the use of epidemic algorithms to act as kernel of a routing protocol fulfilling the above requirements and having more care than flooding of network and node resources. Epidemic algorithms have been extensively studied in the literature and possess interesting networking capabilities that, for instance, have been recently exploited to ensure reliability in MANETs with low mobility [4] and data aggregation over a group in networks with fixed nodes [10]. By relaxing the protocol requirements to meet basic best effort service, the capability of epidemic algorithms to rapidly disseminate data from a source through purely local interactions [7] can be profitably utilized for communications in a mobile and highly dynamic network. In this paper, we describe epidemic algorithms as an alternative to flood packets to provide best effort data diffusion over a dense group in an opportunistic scenario. The main contribution of the paper is to show that the approach actually has the ability to cope with the mentioned constraints of ONs and outperforms flooding to diffuse packets over the network. The paper presents the first set of simulation results and highlights the parameters that influence the performances of the protocol. These results are the starting point to design an adaptive diffusion protocol which is capable to adapt to both the dynamics of the network and the mobility conditions.

II. Epidemic Approaches to Data Diffusion

This section proposes a simple classification of the different types of epidemic algorithms that can be considered for data diffusion to all nodes in the system or to all members of a densely populated group. In the sequel, we use the term *data* to indicate the packet payload generated by the application running at a given source node.

Due to the delay tolerant nature of ONs, data diffusion is assumed not to have real time requirements so that the eventual and best effort delivery to destinations is suitable to meet the application's needs. To this class of application belong, for instance, the diffusion of feeds, advertisements and news (one-to-many scheme) or any other type of asynchronous data, such as e-mail and data from distributed sensors, that need to be collected from server nodes (many-to-one scheme).

In the opportunistic scenario we consider, all nodes are supposed to run the epidemic algorithm and a few of them can act as source of the data to broadcast. Sources are unaware of destination addresses and group cardinality. The communication scheme is one-to-many. Each node p maintains two data structures: the `neighbors_list`$_p$ and the `data_descriptor_list`$_p$. The former is the list of p's neighbors, i.e. the nodes within the radio range of p. The neighborhood information is achievable either from the MAC layer or through network-layer beacons, such as those exchanged by many routing protocols (e.g., [11]). The `data_descriptor_list`$_p$ is the list of the descriptors of the active data the node $knows$, either because they are generated by the node itself or received through epidemic contacts with other nodes. A data descriptor is a tuple $\langle$*source, seq#, data, data-lf*$\rangle$, where *seq#* helps to uniquely associate a data to its source and $data-lf$ represents the data lifetime and is specified by the source;[1] when it expires, all nodes with the data in the `data_descriptor_list`$_p$ remove the entry in the list.

Flooding can be considered a trivial example of epidemic algorithm: as soon as a node knows a new data, it immediately attempts to broadcasts the data to all its neighbors, except (when possible) the one from which the packet has been received. The time-to-live (TTL) of the packet is set to ∞. The pseudo-code for this approach is shown by Algorithm 1. Flooding has known drawbacks that become very critical when referred to opportunistic networks. The generation of $O(N^2)$ packets, with N the number of nodes in the system, negatively affects the device batteries lifetime. This progressively reduces the connectivity opportunity and favours the generation of network partitions. Interestingly, the same partitions it concurs to generate turn against the flooding performances. In fact, as shown by simulations (sec.IV), network partitions become a barrage that flooding cannot get through and, consequently, heavily reduce its capability to reach all the nodes in the system.

[1]As ONs are delay tolerant networks, the lifetime must be not too short, that is, it must not be a parameter critical for the correct operations of the algorithm.

Algorithm 1 Pseudo-code for the flooding approach

```
Initiator:
when data generated do
   broadcast (data) to all one-hop neighbors;
   add data to local data_descriptor_list;
end do

Node:
when data received do
   if 1st time this data received then
      add data to local data_descriptor_list;
      broadcast (data) to all one-hop neighbors except the
      one from which the data has been received;
   end if
end do
```

Unlikely flooding, a notable characteristic of many epidemic algorithms is that data are *cached* by nodes, and their diffusion takes place periodically. This mechanism fits well with the behavior of opportunistic networks; in fact, in the time interval between two consecutive rounds of the algorithm, nodes move, change their neighborhood and get over partitions, thus allowing to reach the entire population of nodes. For our purposes, we introduce a simple classification of epidemic algorithms based on which node, of a pair of communicating nodes, is responsible for initiating the data exchange and on whether the exchange is unidirectional or bidirectional.

As far as the initiator is concerned, approaches can be classified as receiver-based and source-based. In the **receiver-based** case, a node periodically chooses one (or more) of its neighbors and asks it for data. This is also known as PULL approach, whose pseudo-code is provided by Algorithm 2: the `get_neighbors` procedure extracts one or more nodes from the `neighbors_list`; the `get_data` procedure extracts one or more data from the `data_descriptor_list`. All nodes start executing both the Active and the Passive threads at the bootstrap; as the initialization of the Active Thread, the timer for periodic execution is set. In this and the following algorithms we assume that, as soon as a data is generated, the node inserts it in the local `data_descriptor_list`. The PULL approaches may differ for the amount of exchanged data: a node can send one or a subset of the data in its list, chosen according to either a deterministic or

Algorithm 2 Pseudo-code for the PULL approach

```
Receiver $p$: //Active Thread
when timer expires do
    $contact \leftarrow$ get_neighbors();
    send (request for data) to $contact$;
    receive (reply) from $contact$;
    if reply includes unknown data then
        add data in reply to local
        data_descriptor_list;
    end if
    set timer;
end do

Node: //Passive Thread
when request received from $p$ do
    if any data in data_descriptor_list then
        $data \leftarrow$ get_data();
        send ($data$) to $p$;
    else
        send (empty reply) to $p$;
    end if
end do
```

a probabilistic policy, or even all the data. In some proposals, nodes initially exchange a digest of the owned data to selectively decide the data to forward. The PULL approach has been, for instance, proposed in [6] for updating multiple copies in distributed databases.

This approach is not promising for the scenario we are considering where very few sources are supposed to diffuse to many receivers. However, the PULL based approach could be considered in combination with a PUSH based approach when the push process has likely diffused the data to the most part of destinations. As simulations will show in the sequel, at this point the few receivers have many sources to query for data and a PULL based approach can be helpful to selectively reach the total population of nodes at a reduced cost in terms of exchanged packets.

The aforementioned receiver-based approach differs from the mechanisms proposed for warning dissemination in vehicular networks, such as IVG [1], where the diffusion is initiated by the source and continued by forwarders according to a PUSH scheme, but the decision on which node should further forward the data is left to the recipients in communication range with the current forwarder. In this paper, the classification puts the emphasis on the node that is in charge of data forwarding, rather than on the nodes deciding the identity of the next forwarder(s).

Source-based approaches can be further classified in PUSH and PUSH/PULL. In both cases, a node p having a data d_p to diffuse periodically selects a neighbor q and sends d_p to it. In the PUSH/PULL approach, if q has a data d_q, then p and q exchange the missing data at the contact opportunity. The pseudo-code of the two approaches is shown by Algorithms 3 and 4. All nodes start executing the Passive Thread at the bootstrap; the Active Thread is started when the first data is inserted into the local data_descriptor_list.[2] As we mentioned before, in the PULL approach, a data can be diffused only after the explicit request of an active node. By contrast, in the source-based approaches, whenever a node has a data it becomes an active forwarder of the data. As a consequence, the number of active forwarders increases with the time.

Algorithm 3 Pseudo-code for the PUSH approach

```
Active Thread:
when timer expires do
    $contact \leftarrow$ get_neighbors();
    $data \leftarrow$ get_data();
    send ($data$) to $contact$;
    set timer;
end do

Passive Thread:
when data received do
    if notifi ed data not in data_descriptor_list then
        add data to data_descriptor_list;
        if 1st data in data_descriptor_list then
            start Active Thread;
        end if
    else
        discard duplicate data;
    end if
end do
```

It is worth to notice that the PUSH/PULL approach is suitable to accelerate the data diffusion in a multiple source network condition or when multiple data from the same source are concurrently active. The PUSH and PUSH/PULL approaches have been, for instance, proposed in [6] for the management of replicated databases and in [10] for data aggregation in wireless sensor networks.

The performances and functionalities of the three described epidemic approaches is highly influenced by several design issues and by their combined effects. Table I reports the most relevant among them. The approaches

[2]If at a certain time all data are expired, the node halts the Active Thread, to possibly re-start it later when a new data is either generated or received.

Algorithm 4 Pseudo-code for the PUSH/PULL approach

```
Active Thread:
when timer expires do
   $contact \leftarrow$ get_neighbors();
   $data \leftarrow$ get_data();
   send ($data$) to $contact$;
   receive (data) from $contact$;
   if data received not in data_descriptor_list then
      add data to data_descriptor_list;
   else
      discard duplicate data;
   end if
   set timer;
end do

Passive Thread:
when data received do
   if notified data not in data_descriptor_list then
      add data to data_descriptor_list;
      if 1st data in data_descriptor_list then
         start Active Thread;
      end if
   else
      discard duplicate data;
   end if
   if other data in data_descriptor_list then
      $data \leftarrow$ get_data();
      send ($data$) to active node;
   end if
end do
```

TABLE I
CHARACTERISTICS OF EPIDEMIC APPROACHES

aspect	PULL	PUSH	PUSH/PULL
neighbor choice	×	×	×
epidemic round	×	×	×
# data sent	×		×
data choice	×		×
# neighbors		×	×
broadcast usage		×	×
stop condition		×	×

may all adopt different policies (either probabilistic or based on local information) to choose neighbor for data forwarding (`get_neighbors`). Neighbor selection can be performed by either sender or receivers. The epidemic round determines how often a node executes the Active Thread. In both the PULL and the PUSH/PULL approaches, a node may decide to reply with more than one data to its querier or to the Active Thread respectively, and may adopt different policies to extract those data from the local `data_descriptor_list`. When data are pushed, different strategies can be adopted to decide the number of in-range neighbors that are selected to act as data forwarder. A very conservative approach will choose a single forwarder, while a more aggressive policy will exploit the broadcast nature of the radio channel to elect as forwarder all the in-range nodes. The latter approach has obvious effects on the data diffusion speed, but, conversely, requires to randomize the forwarding operations in order to control the collision rate and the duplicate generation. Finally, the periodic diffusion of a data, adopted by both PUSH and PUSH/PULL approaches, is useless when the data has been delivered to all nodes. The optimal behavior is, of course, to ensure that all nodes stop diffusing a data when that data has been received at the last node. This condition requires a global knowledge of the nodes' state and thus cannot be implemented, but can be approximated by designing some proper *stop conditions* [6]. When the stop condition is locally verified, the data is no longer diffused.

III. MOBILITY MODELS

To evaluate the performance of epidemic approaches, we focused on a *campus scenario*, where students, faculty and staff are equipped with wireless portable devices (palmtops, PDAs, 3G cellphones), and may either roam through the campus area or group in interest points such as classrooms, the student-services office or the library.

A great deal of research is currently ongoing in order to characterize mobility models suitable for opportunistic networks. As observed in [2], choosing an appropriate mobility model is fundamental to perform an accurate analysis of protocols behavior. Mobility could be extracted from traces of movements of nodes in real settings; several traces are for instance provided by the CRAWDAD community [5]. The use of traces with simulators creates, however, some problems. Their timescale are hardly scaled to the simulation time and they generally model the specific behavior of a given mobility scenario thus loosing the generality required during the protocol design phase. Traces are more likely useful during the validation process that during the design and performance analysis phase. In [3], [12], traces have been used to derive statistical distributions describing mobility of nodes in opportunistic networks. However, we believe that synthetic models can hardly capture the characteristics of people mobility and, more specifically, we envision three patterns for a campus scenario:

- nodes could *aggregate geographically* in certain sites. This model captures the high density of students in a classroom, or of people in the library;
- *functional aggregation* may describe the behavior of users who cooperate, and are thus likely to meet often, but not necessarily in a fixed site and not necessarily all together. E.g., a subset of the members of a research group could meet in the office of one of them, or in the cafeteria;
- nodes could move in a *swarm*, such as in the case of a group of students visiting a site. Some models describing this pattern are presented in [2].

Mobility of people in an opportunistic scenario is more likely modeled by the combination of the above three patterns with some random movements. At the best of our knowledge, the research community has not yet produced a synthetic model that addresses these issues or adopts the statistical distributions inferred from traces. For this reason, this is a high priority research topic within our research activity plan. However, the main purpose of the work described in this paper is the analysis of epidemic algorithms in a very basic mobility scenario where nodes move randomly without exploiting any movement regularities. The aim of capturing its native ability to provide hop-by-hop data diffusion in the worst mobility and connectivity conditions. This would represent the benchmark analysis of the protocol that has to be adopted to compare the protocol behavior when more realistic scenarios are adopted and to better understand how the protocol may benefit of aggregation points to improve the efficiency of the diffusion. To some extent, this model represents a worst case condition for epidemic algorithms. Indeed, if some regularity exists in the movement pattern, or assumptions can be done about the node spatial distribution, algorithms could take advantage of this such as done by some two-hop relay algorithms [8]. In [2] the authors notice that this model is sufficiently realistic to represent the way people move in, e.g., a conference setting or a museum. As a consequence, in this work, a Random Waypoint mobility model is adopted, with no pause time and speed variable between $minspeed = 1$ m/s and $maxspeed \in \{1, 9\}$ m/s.

Whatever is the adopted mobility model, it should be assumed that the following *mobility assumption* applies: when a contact occurs, the reciprocal speed is such that the two nodes can set up a communication channel and a significant amount of data is exchanged before they become disconnected. This assumption is reasonable according to results achieved by observations reported in [12], [9], but can be occasionally violated in our simulations.

IV. Performance Evaluation

A. Simulated Conditions and Performance Indexes

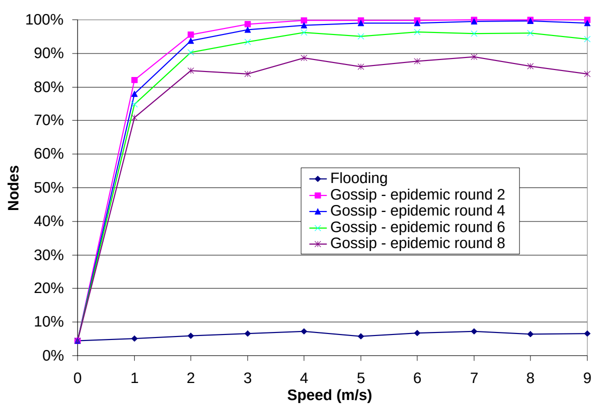
We used simulations to analyze the behavior of a PUSH based algorithm in a simple setting. A single source node is assumed to generate a sequence of data; data $d2$ can be generated when the lifetime of data $d1$ expired. A node randomly selects a single forwarder among in-range neighbors and, for a clearer interpretation of the results, the broadcast nature of the radio channel is not exploited, i.e. the sent data is only received at the chosen forwarder. This choice, on one hand is due to the decision of keeping to the original nature of the epidemic algorithms proposed in the literature; on the other hand, it goes in the direction of obtaining benchmark measures in a worst case scenario, in order to understand how the forwarder selection affects the performance. The rounds of the protocol continue to diffuse data until the data lifetime has been reached without adopting any stop condition to limit the generation of duplicate data. The data lifetime is set to 100 seconds and the epidemic round varies in the range 2 to 8 sec.

A network layer beaconing is assumed to broadcast the node's identifier every `beacon_period`$=1$ sec. Once received a beacon the node stores the sender's identifier in the `neighbors_list` for `beacon_timeout`$=2$ sec.; when a new beacon is received, the entry is refreshed. Up to 100 nodes move in an open space area of 100×100 mt. Nodes are equipped with low power 802.11 radio devices having a 10 mt. communication range.

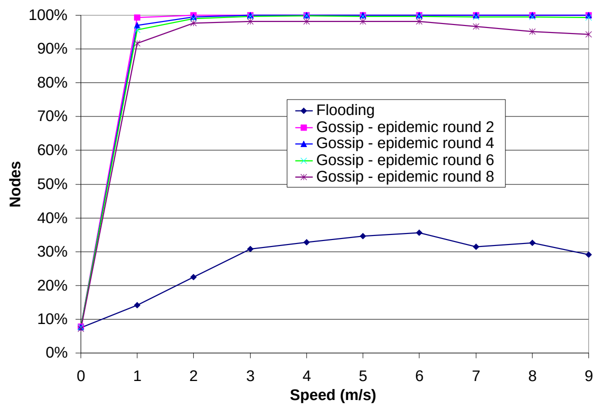
The algorithm has been implemented in the GloMoSim simulator [13] to obtain the results we present in this section. All results are averaged over 100 simulations performed with the same parameters and variable random seed. Effectiveness of the epidemic algorithm is measured in terms of average percentage of covered nodes, i.e. the nodes receiving the data within its lifetime. Efficiency is measured in terms of average and maximum *latency* between the data generation and its reception, and in terms of average number of *duplicate notifications* received by nodes. Of course, duplicate data influence both the network bandwidth and node's battery consumption.

B. Simulation Results

The average number of reached nodes is shown in Fig.1, for variable group cardinality, node's speed and



(a)



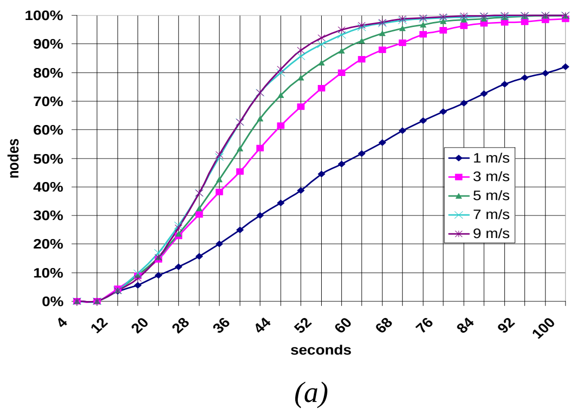
(b)

Fig. 1. Percentage of nodes reached by the data at the end of its lifetime with (a) 25 and (b) 75 nodes

epidemic round. Flooding obtains a very low coverage because it performs a single run of data diffusion: unable to exploit caching and periodic runs, flooding cannot cope with partitions and consumes all its diffusion power inside the partition of the source. This fact is more evident for low node density conditions, where partitions are more likely to occur and "clouds" of nodes are smaller; this hypothesis is confirmed by simulations with fixed nodes that show how flooding coverage grows with node density. When nodes move, flooding performs poorly with respect to the epidemic algorithm although, with high node density, flooding can take some advantage of speed. This is due to the fact that GloMoSim models the wireless channel in an on/off way: within 10 mt. of range, the distance does not affect the channel quality; right over 10 mt. no channel is available. Hence, in high density conditions, partitions are larger and flooding takes a few tens of milliseconds more to cover the source's partition. This time is enough to allow nodes on the border of the partition to slightly move and merge a new partition, thus obtaining a better coverage that, of course, grows with the nodes speed.

By contrast, the PUSH based epidemic approach obtains good performances in all conditions taking great benefit of caching, repeated diffusions and mobility. In low density conditions – where partitions have greater impact – this effect is more evident for increasing speed. A large epidemic round does not perform well for low density, where a small round is more aggressive and more successful in taking advantage of changes in the neighborhood to "infect" new nodes. With a large round, the number of diffusions attempted during the data lifetime is lower. This also implies a higher impact of the *"vanishing neighbor" effect*, consisting in a node choosing as forwarder another node still included in its `neighbors_list` but that actually is out of communication range. In this case, the diffusion is wasted. The impact of this effect depends on the accuracy of the beaconing policy in determining current neighbors. Large epidemic round is less sensitive to speed: as diffusion is performed seldom, neighborhoods change between two consecutive diffusions even at low speed. For high density, speed has less impact and coverage is better: each node has a higher number of neighbors; hence, it more likely chooses as forwarder a node not yet infected before. It is worth to notice that, at the maximum speed, coverage gets worse. This is due to both a more frequent occurrence of the vanishing neighbor effect, and the violations of the mobility assumption. In fact, if two nodes move in the opposite direction, their relative speed is doubled and they are likely to go out of range before exchanging data.

Figg.2 and 3 bring into evidence the impact of node speed. The *ideal* behavior is reaching 100% nodes at time 0; that is, the more vertical is the plot, the better the algorithm performs. The plot behavior depends on a sort of *avalanche effect*: initially only the source propagates the data, and the number of infected nodes increases slowly. Each node infected starts executing the Active Thread (Algorithm 3) on its behalf, thus contributing to the data diffusion. Hence, in a second stage the number of infected nodes grows quickly. Finally, only a few nodes remain to be infected, and they can be reached only if and when an active node chooses them as forwarders. In this last phase, the coverage is almost stable, and the most part of the diffusions are addressed



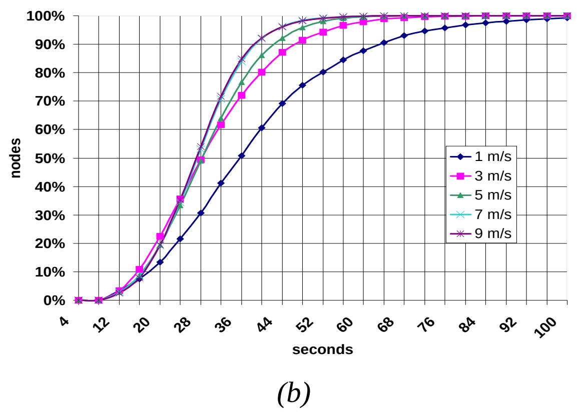
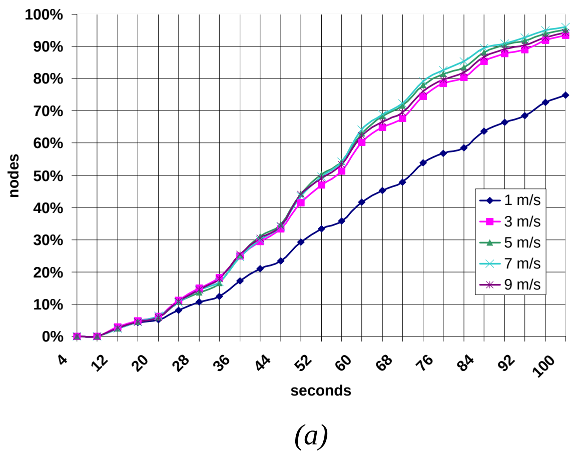


Fig. 2. Cumulative data reception time by nodes with epidemic round set to 2 seconds and with (a) 25 and (b) 75 nodes



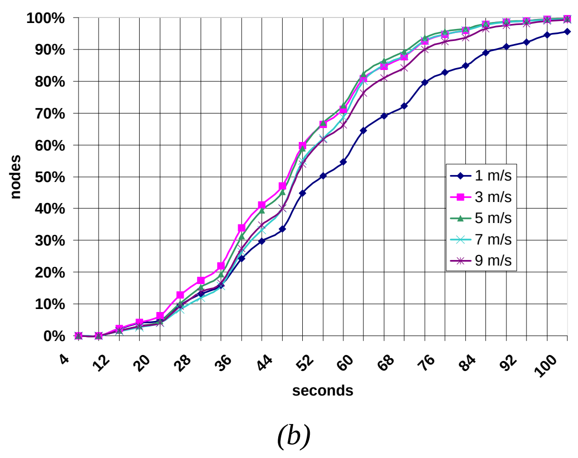


Fig. 3. Cumulative data reception time by nodes with epidemic round set to 6 seconds and with (a) 25 and (b) 75 nodes

to already infected nodes. Under all conditions, diffusion is faster for higher speed, because the node neighborhood changes frequently. As a consequence, the probability of choosing as forwarder a node not yet infected – that is, of performing useful diffusion – increases. Once again, the higher the number of neighbors, the higher the probability of randomly choosing a non infected node. As a side effect, these results bring into evidence the importance of an appropriate neighbor choice in order to achieve good network coverage with low latency. The difference between the plots for maximum and minimum speed is lower in high density conditions. With a long epidemic round (Fig.3), similar results are achieved, although the coverage is worse because of a less aggressive diffusion policy not advantaged by speed, as observed before.

Fig.4 shows the latency in receiving the data, as both the average over all nodes (black lines), and the maximum to reach the last node (grey lines). Flooding is almost independent of speed and density, as observed before. According to the coverage measures, the latency decreases for higher speed and higher density, and for lower (more aggressive) epidemic round. If the epidemic round is long, then high speed penalizes performance because the vanishing neighbor effect and the violation of the mobility assumption lead to a higher probability of failing a diffusion. Moreover, a higher interval elapses before attempting a new diffusion. The maximum latency has a behavior comparable to the average.

In Fig.5, the average number of duplicate notifications received at a node is reported. Flooding generates a very low number of duplicates, only due to multiple paths among nodes; the multiple paths probability and the number of duplicates grow with density. For the PUSH approach, the number of duplicates is higher for high density because more nodes concurrently execute diffusions. This is emphasized for short epidemic round. The duplicate number is affected by speed. At moderate

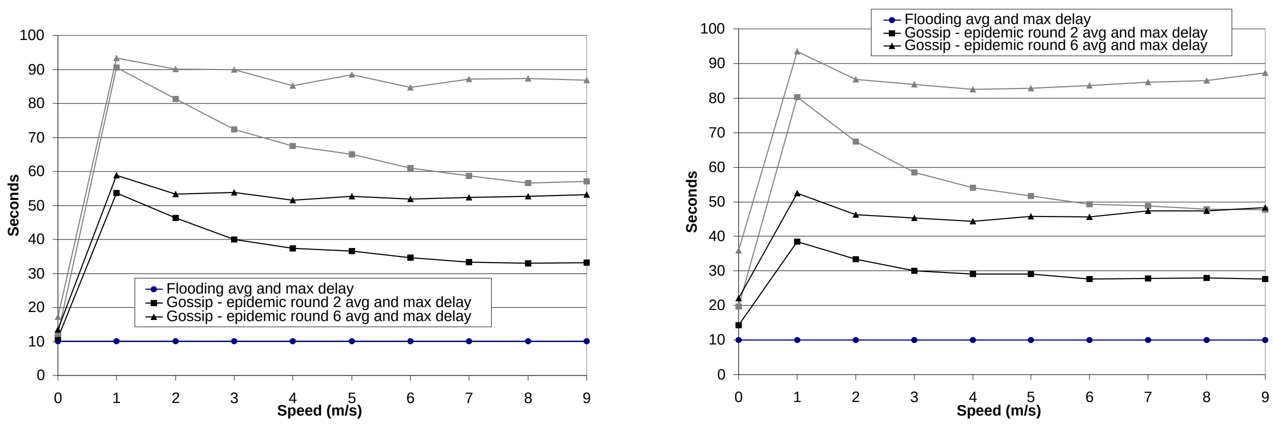


Fig. 4. Average and maximum delay of data reception with (a) 25 and (b) 75 nodes

speed, partitions can be merged and a higher number of nodes is infected. Moreover, neighborhood still changes slowly, and a node can choose the same neighbor many times. At high speed, the vanishing neighbor effect prevents a duplicate reception at an infected node that moved out of range.

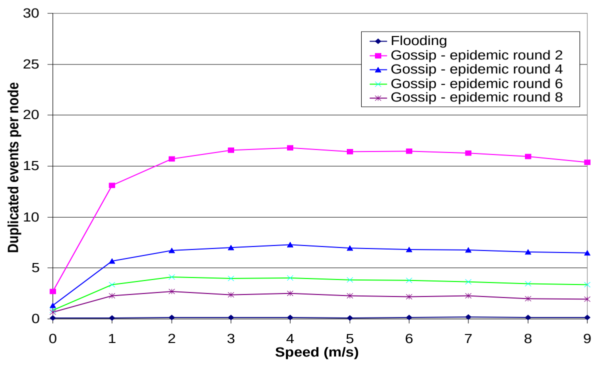
Duplicates waste both bandwidth and energy resources and hinder node coverage, because the next diffusion must be awaited to (hopefully) perform effective work. It is worth to notice that the reduction of the number of duplicates is the main feature to add to an effective epidemic algorithm for an opportunistic scenario. This can be achieved through both a careful forwarder selection, that should consider neighborhood information, and the implementation of stop condition, that should use locally available information, such as the number of received duplicates, to progressively reduce the infection rate as the coverage approximates the group membership. If we consider the dense scenario of Fig.2(b), the PUSH algorithm with nodes moving at 9 m/s reaches all the nodes within 60 seconds. Most part of the duplicates are generated in the remaining 40 seconds. The number of these duplicates can be easily computed as:

$$\#nodes \times \#epidemic\,sessions\,remaining =$$
$$= \#nodes \times \frac{time\,remaining}{epidemic\,round} = 75 \times \frac{40}{2} = 1500$$

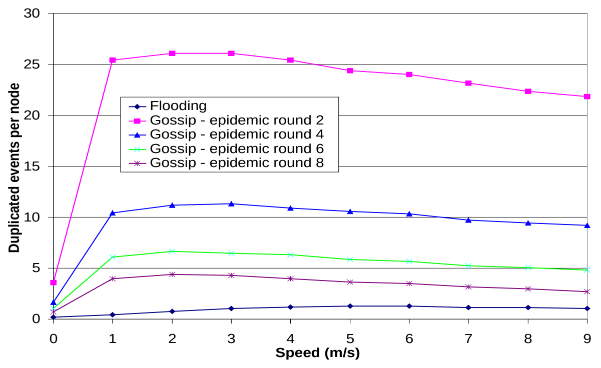
That means that, on average, each node receives $\frac{duplicates}{nodes\,number} = \frac{1500}{75} = 20$ duplicates. If we consider Fig.5(b), we can observe that, being capable of stopping the diffusion when the last node received the data, the number of duplicates generated by the PUSH strategy with epidemic round set to 2 would have been around $25 - 20 = 5$. An epidemic round of 8 seconds obtains similar values at a much higher latency cost.

V. Design Hints for Epidemic Algorithms in ONs

The simulation results discussed in this work provide some useful guideline for the adaptation of existing approaches to the ONs scenario. Results in Figg.1, 2 and 3 bring into evidence the impact of speed on the achieved coverage, because a node is more likely to observe neighborhood changes within an epidemic round thus allowing a quicker infection of new nodes. An adaptive algorithm should involve mechanisms able to monitor neighborhood changes independently of speed, thus overcoming possible erroneous interpretations of the environment status in case nodes move at high speed but in a swarm. The algorithm should also keep some sort of a short term history of previous diffusions, in order to avoid infecting the same nodes more than once and to reduce the impact of duplicate notifications on both bandwidth and battery lifetime. This policy should use local observations of the system at the nodes to save memory and computational resources. Another hint derives from the observation of the effects of high density on algorithm performance (Figg.1, 2, 4): the more dense the nodes, the better both the coverage and the latency. Hence, in an adaptive algorithm, nodes could monitor the cardinality of their neighborhood to achieve awareness of aggregation points and regulate the diffusion round accordingly. As a consequence, in the adaptive algorithm both the policy for choosing forwarders and the epidemic round might dynamically vary according to the system status as observed by nodes. Mechanisms for duplicate suppression and de-synchronization of multiple



(a)



(b)

Fig. 5. Average duplicated data received by each node with (a) 25 and (b) 75 nodes

forwarders activated concurrently must be included, for the purpose of saving both bandwidth and batteries, although possibly at the expenses of latency.

Another important aspect to consider is the beaconing mechanism. Indeed, an accurate knowledge of the neighbors is needed in order to make an appropriate choice of the forwarders, avoiding the vanishing neighbor effect that wastes bandwidth and time without performing any useful work. Keeping a small `beacon_period` is not necessarily a good policy, as it requires a lot of bandwidth, and resulting collisions could hinder both neighbor discovery and information diffusion. We have adopted the classical beaconing procedure; yet, possible changes to the beaconing policy should be studied.

The possibility of a *hybrid* approach should be also considered: data could be diffused using a PUSH approach as long as a relevant number of nodes is covered. When the cumulative coverage of Figg.2 and 3 stabilizes, continuing active diffusion is no longer convenient: non infected nodes should start a PULL algorithm to retrieve the data. As at this point more than 90% of the nodes is infected, only a few nodes on average should be inquired before finding one able to reply. The decision about when to switch from a PUSH to a PULL approach should be taken independently by each node, basing on local observations of the system. Such a hybrid policy could reach a coverage comparable to that of the PUSH approach, while decreasing duplicates.

VI. Conclusions and Future Work

In this work, the suitability of epidemic algorithms for diffusing information in opportunistic networks is analyzed. Epidemic algorithms take advantage of mobility to overcome network partitions, thanks to the caching of information and periodic diffusions. Because of these characteristics, they achieve a better network coverage than flooding, thus resulting promising.

However, research work in this field is just in a seminal state and much effort has yet to be done to adapt epidemic algorithms to opportunistic environments. In sec.V, some suggestions are provided basing on the simulation results achieved. Currently, we are studying techniques for implementing stop conditions combined with neighbor choice, with the goal of designing a solution able to accelerate diffusions when a node realizes it is in a neighborhood of nodes not yet infected, and to slow down – or stop – when neighbors already know the information. Epidemic algorithms have been proposed for lazily diffusing data in environments less dynamic than ONs, with a node periodically passing data to another. We further plan to evaluate whether the choice of forwarders is more convenient on behalf of either the sender, or the receivers according to the local system view each one of them owns. In parallel with those activities, we are implementing a mobility model including statistics obtained from traces of real behavior in ONs and also reproducing aggregative behavior. The model will supply movement logs usable as input to main simulators (GloMoSim and NS-2), for further performance evaluation of both existing and novel approaches.

Acknowledgment

This work has been partially funded by the Italian Ministry of University and Research in the framework of the “Context-Aware RouTing Over Opportunistic Networks (CARTOON)” PRIN Project.

References

[1] A. Bachir and A. Benslimane, *A Multicast Protocol in Ad-hoc Networks:Geocast Inter-Vehicles*. Proc. IEEE Vehicular Technology Conference (VTC) 2003-spring, Apr. 2003.

[2] T. Camp, J. Boleng, and V. Davies, *A Survey of Mobility Models for Ad Hoc Network Research*. Wireless Communications and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, 2(5), pp. 483–502, 2002.

[3] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, *Pocket Switched Networks: Real-world mobility and its consequences for opportunistic forwarding*. Technical Report UCAM-CL-TR-617, Computer Laboratory, University of Cambridge, Feb. 2005.

[4] R. Chandra, V. Ramasubramanian, and K. Birman, *Anonymous Gossip: Improving Multicast Reliability in Mobile Ad-Hoc Networks*. Proc. International Conference on Distributed Computing Systems (ICDCS), pp. 275–283, Apr. 2001.

[5] Crawdad. *A Community Resource for Archiving Wireless Data At Dartmouth*. http://crawdad.cs.dartmouth.edu/

[6] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry, *Epidemic algorithms for replicated database maintenance*. Proc. 6th annual ACM Symposium on Principles of Distributed Computing (PODC), pp. 1–12, Aug. 1987.

[7] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker, *An Empirical Study of Epidemic Algorithms in Large Scale Multihop Wireless Networks*. Submitted for publication, Feb. 2002, http://citeseer.ist.psu.edu/ganesan02empirical.html

[8] M. Grossglauser and D. Tse, *Mobility increases the capacity of ad hoc wireless networks*. IEEE/ACM Transactions on Networking, 10(4), pp. 477–486, 2002.

[9] P. Hui A., Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, *Pocket switched networks and human mobility in conference environments*. Proc. ACM SIGCOMM Workshop on Delay-Tolerant Networking (WDTN), pp.244–251, 2005.

[10] A. Montresor, M. Jelasity, and O. Babaoglu, *Gossip-based Aggregation in Large Dynamic Networks*. ACM Transactions on Computer Systems, 23(3), pp.219–252, Aug. 2005.

[11] C.E. Perkins, E.M. Belding-Royer, and S. Das. *Ad Hoc On Demand Distance Vector (AODV) Routing*. IETF RFC 3561, Jul. 2003. Work in Progress.

[12] J. Su, A. Chin, A. Popivanova, A. Goely, and E. de Lara, *User Mobility for Opportunistic Ad-Hoc Networking*. Proc. 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), Dec.2004.

[13] UCLA Parallel Computing Laboratory, *GloMoSim – Global Mobile Information Systems Simulation Library*. University of California at Los Angeles. http://pcl.cs.ucla.edu/projects/glomosim/