Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding in Sparse Mobile Networks

Khaled A. Harras, Kevin C. Almeroth and Elizabeth M. Belding-Royer

Department of Computer Science, University of California, Santa Barbara, CA 93106-5110, Ph: 805-893-7520, Fax: 805-893-8553
{kharras, almeroth, ebelding}@cs.ucsb.edu

Abstract. The incredible growth in the capabilities and functionality of mobile devices has enabled new applications to emerge. Due to the potential for node mobility, along with significant node heterogeneity, characteristics such as very large delays, intermittent links and high link error rates pose a new set of challenges. Along with these challenges, end-to-end paths are assumed not to exist and message relay approaches are often adopted. While message flooding happens to be a simple and robust solution for such cases, its cost in terms of network resource consumption is unaffordable. In this paper, we focus on the evaluation of different controlled message flooding schemes over large-scale, sparse mobile networks. We study the effect of these schemes on message delay and network resource consumption. Our simulations show that our schemes can save substantial network resources while incurring a negligible increase in the message delivery delay.

Keywords: Mobile Networks, Delay Tolerant Networking

1 Introduction

Today's Internet, as well as various other networks, operate on some often overlooked assumptions. These assumptions include: small end-to-end Round Trip Times (RTTs), a complete end-to-end path between sources and receivers, and end-to-end reliability. Packet switching is assumed to be the right abstraction for end-to-end communication in these networks.

Currently, however, new challenges are surfacing as different kinds of applications and networks emerge, especially with the incredible growth in mobile devices. These devices are driving users' needs and demands toward the expectation of connection availability in all places and at all times. These demands are creating some formidable challenges such as large delays, intermittent end-to-end connectivity and high link error rates. In essence, the underlying network architecture and protocols are becoming increasingly heterogeneous with wide variations in performance characteristics and capabilities. Applications that have these challenges include satellite networks, planetary and interplanetary communication, military/tactical networks, disaster response, and various forms of large-scale or sparse ad-hoc networks.

These new applications and challenges, along with the impossibility of ubiquitous deployment of a fixed wired Internet infrastructure, have stimulated a great

deal of research. Research in the areas of Mobile Ad Hoc Networks (MANETs) [1],[2],[3],[4],[5],[6], disconnected sparse networks [7],[8],[9],[10],[11],[12] and Delay Tolerant Networks (DTNs), [13],[14], has addressed some of these challenges and problems. However, despite the great contribution of these areas, many issues require further research. This can be demonstrated by the following scenario: A person is driving on a highway in southeastern California, carrying his own laptop or PDA, and needs to send an urgent email or submit a transaction. There is no nearby connectivity (in a desert for instance). The user passes other cars, buses or trains that have other people carrying similar devices. These users can serve as relays for the email or transaction and pass it on to others. Eventually, the message reaches someone with Internet connectivity and a direct path to the destination.

This scenario is an example of what we call a *large-scale sparse mobile network*. Most research fails to solve many problems posed by such scenarios. MANETs, for instance, focus more on networks where an end-to-end path is assumed to exist. On the other hand, partially-connected and sparse networking research usually assumes some sort of control over nodes in the network, a large degree of homogeneity, or some degree of knowledge that nodes must carry regarding other nodes in the network (e.g. the path or route a node will take). Clearly these assumptions do not hold in our scenario. DTN research comes closest to addressing the problems that arise from the scenario mentioned above. Our work complements that of the DTN community.

In this paper, we study the impact of controlled message flooding schemes over sparse mobile networks on message delay and network resource consumption. We examine the use of a probabilistic function for message forwarding. We then add a time-to-live (TTL) or kill time value on top of the probabilistic function. Finally, we add the idea of a Passive Cure on top of the other schemes and see what effect it has on the network. The Passive Cure is basically used to "heal" the "infected" nodes in the network.

The rest of this paper is organized as follows. Section 2 reviews areas of research related to our work. Details of our architecture, assumptions and the various controlled flooding schemes are presented in Section 3. Section 4 describes our simulation environment. Section 5 then discusses the results of our simulations. Finally, the paper is concluded in Section 6.

2 Related Work

Research in the areas of MANETs, disconnected sparse mobile networks and delay tolerant networks have tackled issues related to the new challenges and assumptions stated in Section 1. We briefly present some of the solutions that have been proposed to solve some of these challenges.

Much of the research on MANETs has focused on routing, introducing various proactive, reactive or hybrid routing protocols that try to find a path from a source to a destination [1],[2],[3],[4],[5],[6]. All of these routing algorithms, however, assume the existence of an end-to-end path. In other words, if there is no route, no communication can occur, and there is no attempt to relay the message.

Due to this, these protocols are unlikely to find routes in sparsely connected networks. Our work specifically deals with cases in which this assumption is likely not to be true.

With the growing realization that there are many cases where an end-to-end connection cannot be assumed, several solutions have been proposed. All of these solutions rely on some form of store-and-forward relaying of messages [7],[8],[9],[10],[12],[11]. However, these solutions make assumptions that cannot be applied to the scenario proposed in Section 1. Some, for instance, assume control over node movement [7]. Others assume knowing the path that some nodes will take, as well as the time at which the node will take that path, as in message ferrying [12]. Finally, some consider only static nodes as in Data Mules [10]. Epidemic Routing [8] avoids such assumptions by simply flooding the network. The problem, however, is that Epidemic Routing creates a significant traffic load that consumes network resources and does not scale well. Our work adds to that of Epidemic Routing by finding ways to control this network flood and resource consumption.

Finally, within the Internet Research Task Force (IRTF), the Delay Tolerant Networking Research Group (DTNRG)[1] evolved from the Inter-Planetary Networking Research Group (IPNRG)[2]. The group focuses on the construction of a complete architecture that supports various protocols to achieve connectivity among heterogeneous networks in extreme environments [15],[13]. Routing issues in such environments have recently been investigated as well [14]. Our work complements that of the DTNRG by looking at a special case within the general architecture they present [13]. We also fill in the gap of looking at routing issues (through controlled flooding schemes) in sparse mobile networks in particular, and look at cases where no knowledge "oracles" [14] are assumed to exist.

3 System Architecture

This section describes our proposed architecture. We first introduce our assumptions, and briefly explain components of our architecture, along with the functionality and behavior of each component. Following that, we introduce the notion of *willingness*, which is a reflection of the degree at which nodes are willing to participate in relaying messages. Finally, we present the different schemes we use to control message flooding.

3.1 Basic Architectural Components and Assumptions

To satisfy the problems and challenges posed by the scenario mentioned in Section 1, we propose a transport layer overlay architecture for sparse mobile networks. Message forwarding and handling is done by this overlay layer and handles the heterogeneous protocols and node characteristics. This approach also complies with the basic DTN architecture [13].

[1] http://www.dtnrg.org/
[2] http://www.ipnsig.org/

There are two important assumptions made in our system. The first is **Node Blindness**, where the nodes in the network do not know any information regarding the state, location or mobility patterns of other nodes. The second is **Node Autonomy**, where each node has independent control over itself and its movement. The reason behind these assumptions is to closely model the real world scenarios described earlier. When driving through a sparse environment, any given node has no knowledge of other nodes that come within its range (Blindness), and it is autonomous in its movement (Autonomy).

In Figure 1 we show that the nodes in the network are divided into three types. A **Sender Node "S"** is the node that initiates the transmission of a message to a destination in the network. A **Forwarder Node "F"** is any node that carries the message from the sender, or another forwarder, with the aim of relaying it to the ultimate node. Finally, the **Ultimate Node "U"** is basically the final destination.

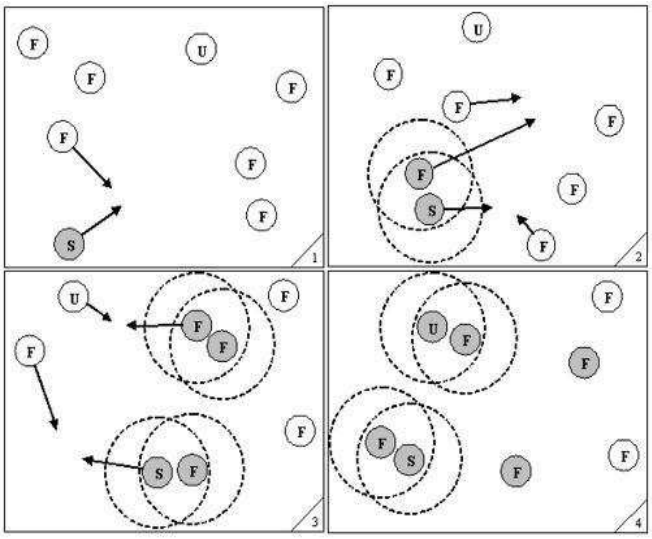


Fig. 1. Message propagation over a sparse mobile network. Shaded nodes are those carrying a copy of the message. Arrows indicate the direction of movement.

The basic mechanism of node interaction is shown in Figure 1. The interaction of nodes is similar to that in Epidemic Routing [8], where each node continuously tries to relay the message to other nodes within range that do not already have the message. We look at an example where a sender node, S, needs to send a message.

At this point, node S initiates a periodic beacon for neighbor discovery purposes, where it announces that it has a message that needs to be forwarded to a certain destination. When S comes within range of one or more forwarder nodes F (or even the ultimate node U), the beacon is received and an ack is sent to S

from each node that received the beacon and does not have a copy of the message. When S receives an ack for its beacons, it simply broadcasts the message to its neighbors. Once the message is received, the forwarder node starts to act as a sender node. It sends its own beacons, and both nodes travel through the network looking for either another forwarder to pass the message on, or for the ultimate node. The message gradually propagates through the network until it eventually reaches the ultimate node. This process results in the overuse of network resources through continuous and repetitive flooding of messages. On the other hand, the advantage of this approach is the high delivery rate and small delay.

3.2 Modelling Node Willingness

Generally speaking, the frequency at which a sender or forwarder node tries to forward a message depends on many factors. Some of these factors include node state (power or buffer space, for instance) or message state (size or priority of message). We model this as the **Willingness** of a node. The willingness of a node is the degree at which a node actively engages in trying to re-transmit a message. Willingness can be modelled in terms of three main variables. First, the **Beacon Interval** is the amount of time a sender or forwarder node waits before sending a new beacon. Second, the **Times-to-Send** is the number of times a node successfully forwards a message to other nodes in the network before it stops forwarding the message. Third, the **Retransmission Wait Time** is the amount of time a node waits without beaconing before it tries to resend the message to other nodes in the network. The source node includes the value of these parameters as part of the message header. This way, the forwarder nodes can set their willingness levels accordingly.

To help clarify how these three variables affect the behavior of a given node in the network, we introduce the following simple example. Let us assume that the beacon interval = 1 sec, times-to-send = 2, and the retransmission wait time = 50 sec. This means that when a sender wants to send a message, it sends a beacon every second to find other nodes that would carry the message. Once the sender node finds a forwarder node (by receiving an ack for its beacon), it transmits the message and decrements the times-to-send by one. The sender then waits for 50 seconds before it resumes sending beacons every second to look for the next node to which to forward the message. This process repeats until the times-to-send reaches zero. The forwarder nodes that received the message in both cases start acting as the original sender (assuming that all nodes have the same willingness).

3.3 Specific Controlled Flooding Schemes

In this paper, we study different controlled flooding schemes. We are careful to base our schemes on simple, non-chatty and elegant algorithms. This is because in sparse and highly mobile networks, complex or chatty algorithms waste the short time nodes have when they come within range of each other. The schemes we introduce are the following:

1) Basic Probabilistic (BP): When talking about the *willingness* of a node in the previous section, we implied that forwarder nodes have the same willingness as the sender node. To more closely emulate reality, however, we choose a uniform distribution probabilistic function that determines the willingness of the nodes to transmit a given message. Based on the result of this function, a forwarder may choose not to forward the message at all, forward it at half the willingness of the sender or forward it at the same level of willingness as the sender.

2) Time-to-Live (TTL): In this scheme, we add a time-to-live value. The TTL here determines how many times the message is forwarded before it is discarded. We add the TTL on top of the BP scheme since the BP scheme is a more realistic representation of how nodes act regarding the choice of forwarding messages.

3) Kill Time: Here, we add a time stamp to the message on top of the BP scheme. The time stamp is the time interval after which the message should no longer be forwarded. This is an absolute universal life-time for the message. This could be very useful if the sender node knows how long it will be disconnected. This is also a good way to set the maximum time a node should keep a message in its buffer if the times-to-send (TTS) variable of that message does not reach zero.

4) Passive Cure: The final scheme or optimization we introduce is a Passive Cure. The idea is that once the destination (Ultimate node) receives the message, it generates a Passive Cure to "heal" the nodes in the network after they have been "infected" by the message. The ultimate node "cures" the forwarder that passed the message to it by sending a cure-ack instead of a normal ack that is sent when a beacon is received. Now whenever that forwarder or the ultimate destination detect any other node sending that same message, they send a cure-ack to that node to prevent future retransmissions.

4 Simulation Environment

The main goal for our simulations in this paper is to compare the performance of the schemes described in Section 3 and study their impact on our metrics.

To start, we first describe our simulation environment. We conducted our simulations using the GloMoSim network simulator. Since we do not have real data describing our targeted scenarios, we use the random way-point model since, intuitively, it most closely approximates the scenarios we are concerned with. The node speed ranges between 20 to 35 meters per second (to get a range of almost 45 to 80 miles per hour); the rest period is between 0 and 10 seconds. Every point in our results is taken as an average of ten different seeds. We added to the simulator an overlay layer that handles the storing and forwarding of messages, as well as the enforcement of the various stopping techniques that we evaluate. The parameters used in our simulations are summarized in Table 1.

Other important parameters in our simulations are:

- **Beacon Interval:** Time period after which a beacon is sent. It ranges from 0.5 sec to 50 sec with a nominal value of 1 sec.

Table 1. Simulation Parameters

Parameter	Value Range	Nominal Value
Number of Nodes	10 to 250	100
Terrain	$10km^2$ to $50km^2$	$10km^2$
Simulated Time	1hour to 24 hours	6 hours
Transmission Range	250m	250m

- **Times-to-Send (TTS):** Number of times to successfully forward a message. It ranges from 1 to 50 with a nominal value of 10.
- **Retransmission Wait Time (RWT):** Time period after which a node tries to resend a message. It ranges from 0 sec to 500 sec with a nominal value of 50 sec.
- **Time-to-Live (TTL):** The number of hops after which a message is discarded. It ranges from 2 to 10 with a nominal value of 7
- **Kill Time:** The absolute time after which the message is discarded. It ranges from 1000 sec to 21600 sec (6 hours, the nominal simulation duration time) with a nominal value of 5000 sec.

We consider two metrics in evaluating our controlled flooding schemes. First, **Network Efficiency** is represented by the total number of messages sent by the nodes in the network. Second, **Overall Delay** is the total time that elapses from when a node wants to send a message until the intended destination (ultimate node) receives that message for the first time.

5 Results

A representative set of our simulation results is presented in this section. All the results are shown for one sender node trying to send one message to a single ultimate destination. We hope to achieve two main goals with our study. First, we hope to better understand how a controlled flooding based overlay network behaves in general. Second, we want to know how the specific flooding schemes we are testing affect the network efficiency and overall delay.

5.1 Basic Behavior

Before applying our probabilistic scheme, we analyze how the network acts assuming full willingness of all the nodes in the network along with enforcing some basic level of message control. This scheme is very similar to the Epidemic Routing scheme but with some control over message forwarding. We use the times-to-send (TTS) variable to represent node willingness. We investigate how varying the TTS and the network density affect our metrics. The retransmission wait time (RWT) in these experiments is 1 sec.

Figure 2(a) shows the total number of messages sent by all the nodes in the network and Figure 2(b) shows the delay. Overall, Figure 2 shows that increasing

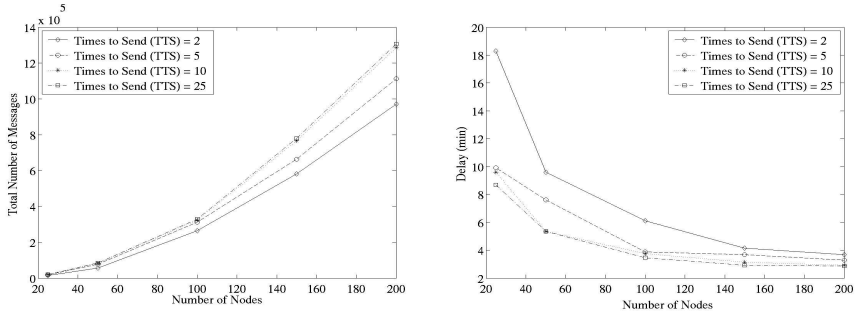


Fig. 2. Impact of changing the number of nodes and times-to-send (TTS) on the total number of messages (a) and overall delay (b) in the basic scheme.

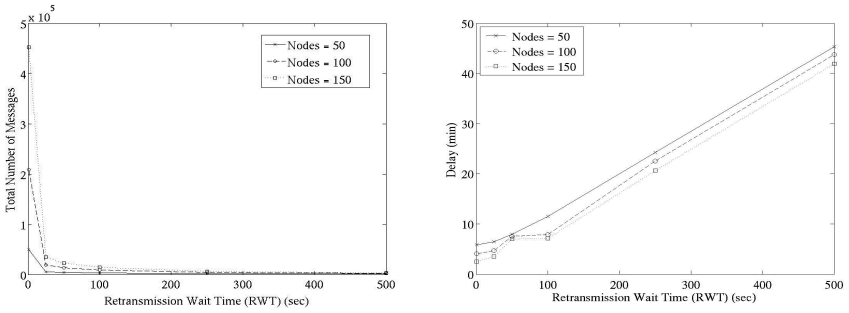
the density of the network results in a large increase in the total number of messages sent in the network. One interesting point is that an increase in the network density results in a significant decrease in the overall delay only up to a certain point (Number of nodes = 100), after which the decrease in delay that we achieve is overshadowed by the corresponding increase in cost. On the other hand, increasing the willingness beyond a certain point (TTS = 10) does not have a large effect on either metric. The reason for this is that the nodes in such sparse networks do not encounter each other often enough to consume the large value of the times-to-send (TTS).

5.2 Basic Probabilistic Behavior

After applying our basic probabilistic scheme, we examine how the network density, retransmission wait time (RWT) and times-to-send (TTS) impact our metrics. We assume that 25% of the nodes in the network have zero willingness, 25% have full willingness and 50% of the nodes forward the message with only half the willingness (i.e. half the TTS of the source node).

Figure 3 shows the result of varying the network density while keeping the TTS set to 10. One interesting observation is the significant drop in terms of the total number of messages when the RWT increases, with only a corresponding small increase in delay (until RWT reaches 100). With a small RWT, the message spreads rapidly through the network, and in a very short time, many forwarders are actively trying to send the message. As the RWT increases, the message does not initially reach as many forwarders. This results in a significant drop in the number of messages. Also, with a large RWT, the TTS is not consumed as quickly, so a forwarder node stays in forwarder mode longer, thus rejecting receipt of the same message. With a small RWT, the TTS is quickly consumed. Because the nodes do not keep any state or cache, they are ready to receive the same message and start forwarding it again.

It is important to note here that similar patterns are generated when keeping the number of nodes constant, while varying the TTS. We have also seen similar



(a) (b)

Fig. 3. Impact of changing the retransmission wait time (RWT) and number of nodes on the total number of messages (a) and overall delay (b).

patterns when we used only the basic controlled scheme, but with a much larger scale in terms of the total number of messages. This shows how a uniform based probabilistic scheme performs much better. The results are not shown due to space limitation.

5.3 Time-to-Live (TTL)

With the improvement in the basic probabilistic scheme, we build the other schemes on top of it to see their impact on our metrics. Figure 4 shows the impact of adding TTL only and adding TTL + Passive Cure, to the basic probabilistic (BP) scheme. We discuss the Passive Cure results later and here we focus on the TTL.

The first result we observe in Figure 4 is the large decrease in the total number of messages sent in the network (note the scale is significantly smaller than in figures 2(a) and 3(a)). This is because the TTL puts a limit on the number of times a message is forwarded. Previously, messages endlessly propagated throughout the network with no limit other than the nodes' willingness.

The impact of TTL on delay is not show here. However, we found that as the TTL increases, the overall delay decreases up to a certain point (at TTL = 7), after which it stays constant at 10 minutes. It is important to note that even though there is a probability of message loss when using low TTLs, we think it is a better choice when added to the basic probabilistic (BP) scheme. If a large TTL is set, our simulations show that all messages reach the destination, and the cost is much smaller than in the BP scheme. For instance, if we choose a TTL of 9, we incur a cost of 1000 messages instead of 25000 messages while keeping the overall delay is the same in both cases.

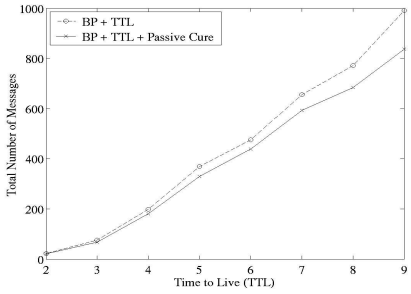


Fig. 4. Impact of adding TTL and Passive Cure to BP.

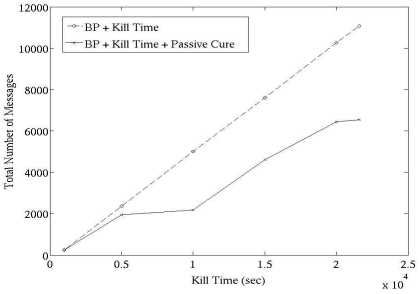


Fig. 5. Impact of adding Kill Time and Passive Cure to BP.

5.4 Kill Time

Figure 5 shows the impact of adding kill time only, and adding kill time + Passive Cure, to the basic probabilistic (BP) scheme. Here we focus on the kill time results, leaving the Passive Cure for later.

The kill time scheme introduces another improvement over the basic probabilistic (BP) scheme alone. The use of a universal time after which a message is discarded certainly stops message transmission and propagation. Figure 5 shows how the total number of messages increases as the kill time is increased. On the other hand, the overall delay (not shown due to space limitation) remains constant at 10 mins, unless the kill time is set too small that some of the messages do not make it to the destination.

5.5 Passive Cure

In Figures 4 and 5, we demonstrate the effect of adding the Passive Cure optimization to the TTL and kill time, respectively. The Passive Cure does not introduce any improvement in terms of delay because it does not help the message get to the ultimate destination any faster.

The effect of the cure is evident only in the total number of messages sent in the network. As Figures 4 and 5 show, when the Passive Cure is added, there is a drop in the total number of messages. This is due to the fact that when the cure starts to operate, the cured nodes stop trying to forward the messages even if the kill time has not been reached or if the TTL has not been consumed.

It is certainly worth mentioning that the Passive Cure optimization has several advantages over the other schemes. First, if the message reaches the ultimate node early, little network flooding may take place since the cure stops the flood early on. Second, the ultimate node receives the message only once. In all the other schemes, the ultimate node may receive the same message multiple times. Finally, the cure may be used as a way to implement end-to-end acknowledgments if it propagates back to the sender node, since the sender would then know that its message reached the ultimate destination.

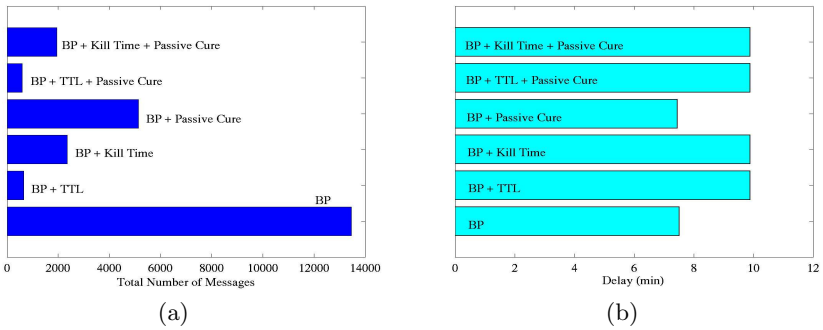


Fig. 6. Comparing the impact of the controlled flooding schemes on the total number of messages (a) and the overall delay (b).

5.6 Comparing Schemes

In this section, we compare the performance of the flooding schemes. Overall, Figure 6 shows the performance of these schemes relative to each other. When looking at Figure 6(a), we observe that the Basic Probabilistic scheme by itself is the most expensive, while the basic probabilistic (BP) + TTL + Passive Cure is the least expensive in terms of the total number of messages sent in the network. Note that the basic probabilistic (BP) scheme already performs better than the basic flooding technique (which is analogous to Epidemic Routing).

Figure 6(b) shows that most of the schemes have similar overall delay. The advantage is for the basic probabilistic (BP) and basic probabilistic (BP) + Passive Cure schemes. The reason for this is that the complexity introduced by the TTL or the kill time results in an overall increase in delay. However, a two minute increase, with a corresponding large decrease in number of messages and beacons sent, is certainly acceptable.

Figure 6 summarizes the general tradeoffs between the schemes we introduce. For example, adding the Passive Cure to the basic probabilistic (BP) scheme saves about 60% of the total number of messages, with no increase in the overall delay. When adding the TTL to the basic probabilistic (BP), the total number of messages is reduced by more than 90%, while the overall delay increased by only two minutes. This increase in delay does not notably affect most of the applications we envision for this work.

6 Conclusions

In this paper we study the problem of efficient message delivery in delay tolerant sparse mobile networks. We propose several controlled flooding schemes on top of a transport layer overlay architecture. The specific schemes we examine are basic probabilistic, time-to-live, kill time and Passive Cure. We study the impact of these schemes on network efficiency and overall message delivery delay. Our

simulations show that for a given sparse mobile network, the schemes significantly reduce the total number of messages and beacons sent in the network. This occurs with either no increase or only a small affordable increase in the overall message delay.

Our future work includes the transmission of a message to multiple destinations. Also, since flooding based schemes do not perform well in dense environments, we need to add measures to help the nodes modify their behavior when they enter densely populated areas.

References

1. Johnson, D., Maltz, D.: Dynamic source routing in ad hoc wireless networks. In: Mobile Computing. Volume 353. (1996)
2. Perkins, C.: Ad-hoc On-Demand Distance Vector Routing. In: Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA (1999) 90–100
3. Haas, Z., Pearlman, M.: The Performance of Query Control Schemes for Zone Routing Protocol. In: ACM SIGCOMM, Vancouver, Canada (1998) 167–177
4. Royer, E., Toh, C.: A Review of Current Routing Protocols for Ad-hoc Mobile Wireless Networks. IEEE Personal Communications Magazine **6** (1999) 46–55
5. Perkins, C., Bhagwat, P.: Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In: ACM SIGCOMM, London, England (1994) 234–244
6. Ko, Y.B., Vaidya, N.: Location-Aided Routing (LAR) in Mobile Ad hoc Networks. In: ACM MobiCom, Dallas, TX (1998) 66–75
7. Li, Q., Rus, D.: Sending Messages to Mobile Users in Disconnected Ad-Hoc Wireless Networks. In: ACM MobiCom, Boston, MA (2000) 44–55
8. Vahdat, A., Becker, D.: Epidemic Routing for Partially Connected Ad Hoc Networks. Technical Report CS-200006, Duke University (2000)
9. Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L., Rubenstein, D.: Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences With ZebraNet. In ASPLOS (2002)
10. Shah, R., Roy, S., Jain, S., Brunette, W.: Data MULEs: Modeling a Three-Tier Architecture for Sparse Sensor Networks. In IEEE SNPA Workshop (2003)
11. Zhao, W., Ammar, M.: Proactive Routing in Highly-Partitioned Wireless Ad Hoc Networks. In: The 9^{th} IEEE International Workshop on Future Trends of Distributed Computing Systems, San Juan, Puerto Rico (2003)
12. Zhao, W., Ammar, M., Zegura, E.: A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks. In: MobiHoc, Tokyo, Japan (2004)
13. Fall, K.: A Delay-Tolerant Network Architecture for Challenged Internets. In: ACM SIGCOMM, Karlsruhe, Germany (2003)
14. Jain, S., Fall, K., Patra, R.: Routing in a Delay Tolerant Network. In: ACM SIGCOMM, Portland, Oregon (2004)
15. Cerf, V., Burleigh, S., Hooke, A., Togerson, L., Durst, R., Scott, K., Travis, E., Weiss, H.: Interplanetary Internet (IPN): Architectural Definition. IETF Internet Draft, draft-irtf-ipnrg-arch-00.txt (work in progress) (2001)