Evaluation of Queueing Policies and Forwarding Strategies for Routing in Intermittently Connected Networks

Anders Lindgren, Kaustubh S. Phanse
Department of Computer Science and Electrical Engineering
Luleå University of Technology
SE-971 87 Luleå, Sweden
{dugdale,kphanse}@sm.luth.se

Abstract

Delay tolerant networking (DTN), and more specifically the subset known as intermittently connected networking, is emerging as a solution for supporting asynchronous data transfers in challenging environments where a fully connected end-to-end path between a source and destination may never exist. Message delivery in such networks is enabled via scheduled or opportunistic communication based on transitive local connectivity among nodes influenced by factors such as node mobility. Given the inherently store-and-forward and opportunistic nature of the DTN architecture, the choice of buffer management policies and message forwarding strategies can have a major impact on system performance. In this paper, we propose and evaluate different combinations of queueing policies and forwarding strategies for intermittently connected networks. We show that a probabilistic routing approach along with the correct choice of buffer management policy and forwarding strategy can result in much performance improvements in terms of message delivery, overhead and end-to-end delay.

1 Introduction

There are many harsh and challenging environments, for example, deep space communication [6], digital content delivery in rural areas with under-developed infrastructure [13, 15], wildlife and habitat monitoring [9, 14], where traditional networking solutions are not viable and where even extensions such as ad hoc networking do not provide a definite solution. These environments are typically characterized by disruption of communication links leading to frequent and long durations of network partitioning, long delays, limited resources and heterogeneity.

Several solutions have been proposed to handle routing in such networks. These range from pure *epidemic* routing [17], where each message is "flooded" through the network to reach as large part of the network as possible (thus also reaching its destination), to more sophisticated mechanisms. Such mechanisms can make use of special knowledge of the scenario [1, 9, 13, 14] or can be probabilistic routing protocols [11] that try to establish the probability that a certain node will be able to deliver a message to its destination based on previous events. However, an important issue that has been largely disregarded in previous work is the impact of buffer management policies and forwarding strategies on the performance of the communication system. This is crucial given the inherently store and forward nature of the DTN architecture, and is a critical piece of the tradeoffs that exist between the overhead of random opportunistic policies and the ability to exploit knowledge. In this paper, we propose a set of queueing policies and forwarding strategies. We compare the performance of two routing protocols in presence of different combinations of the proposed policies.

The rest of the paper is organized as follows. Section 2 describes some related work on routing protocols for intermittently connected networks and gives the background to the work presented in this paper. In Section 3 we propose some queueing policies and forwarding strategies, which are later evaluated in this paper. Section 4 describes the simulation setup used for the evaluations, while Section 5 presents the results of our performance evaluation. We summarize our conclusions and discuss future work in Section 6.

2 Routing Protocols for Intermittently Connected Networks

2.1 Epidemic Routing

Vahdat and Becker present a routing protocol for intermittently connected networks called Epidemic Routing [17]. This protocol relies on the theory of epidemic algorithms by doing pairwise information of messages between nodes as they come in

contact with each other to eventually deliver messages to their destination. Hosts buffer messages if no path to the destination is currently available. An index of these messages, called a *summary vector*, is kept by the nodes, and when two nodes meet they exchange summary vectors. After this exchange, each node can determine if the other node has any message not previously received by it. In that case, the node requests the messages from the other node. The message exchange is illustrated in Figure 1. This means that as long as buffer space is available, messages will spread like an epidemic of a disease through the network as nodes meet and "infect" each other.

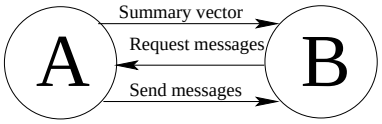


Figure 1: Epidemic Routing message exchange.

2.2 PRoPHET

If buffer space and bandwidth are unlimited, Epidemic Routing is likely to be good at message delivery. This is, however, not the case in reality, where resources such as bandwidth and buffer space are constrained. Furthermore, in real life, nodes often follow a non-random mobility pattern. To leverage mobility and use scarce resources efficiently, we have previously proposed the *Probabilistic Routing Protocol using History of Encounters and Transivitiy (PRoPHET)* [10, 11].

To accomplish this, a probabilistic metric called *delivery predictability*, $P_{(A,B)} \in [0,1]$, is established at every node A for each known destination B. This indicates how likely it is that this node A will be able to deliver a message to a particular destination B and is used in deciding which messages are being transferred between two nodes as they meet. Instead of just exchanging summary vectors upon encounter, nodes also exchange information about the P-values they have for known destinations. This information is used to update the probability information at the receiving node, and is also used to determine whether or not to forward a particular message to the encountered node. This decision is made according to the *forwarding strategy* in use (such as those described in Section 3.2).

The calculation of the delivery predictabilities has three parts. Whenever a node is encountered, the delivery predictability for that node is updated as shown in Eq. 1. This is done so that a node has higher delivery predictability for nodes that are frequently encountered than for those that are seldomly encountered.

$$P_{(A,B)} = P_{(A,B)_{old}} + \left(1 - P_{(A,B)_{old}}\right) \times P_{init} \tag{1}$$

When two nodes exchange information about other nodes they know about, this information is used to update the delivery predictabilities for other nodes as shown in Eq. 2. Here, node A can update its delivery predictability for each node C that B has knowledge of using the transitive property of PRoPHET.

$$P_{(A,C)} = P_{(A,C)_{old}} + \left(1 - P_{(A,C)_{old}}\right) \times P_{(A,B)} \times P_{(B,C)} \times \beta \tag{2}$$

Finally, there is an aging of the delivery predictabilites, so they are periodically updated as shown in Eq. 3.

$$P_{(A,B)} = P_{(A,B)_{old}} \times \gamma^k \tag{3}$$

In the equations above, $P_{init} \in (0,1]$, $\gamma \in (0,1)$, and $\beta \in [0,1]$ are configurable parameters to the PRoPHET protocol, and k is the number of time units that has passed since the value was last aged.

3 Queueing Policies and Forwarding Strategies

Nodes may have to buffer messages for a long time and in case of congestion decide which messages to drop from its queue. They also have to decide which messages to forward to another node that is encountered. In this section we describe the different queueing policies and forwarding strategies used in this paper for the evaluation in Section 5.

3.1 Queueing Policies

We use the following queue management policies, that define which message should be dropped if the buffer is full when a new message has to be accomodated.

FIFO – First in first out Handle the queue in a FIFO order. The message that was first entered into the queue is the first message to be dropped.

MOFO – Evict most forwarded first In an attempt to maximize the dispersion of messages through the network, this policy requires that the routing agent keeps track of the number of times each message has been forwarded. The message that has been forwarded the largest number of times is the first to be dropped, thus giving messages that have not been forwarded fewer times more chances of getting forwarded.

MOPR – Evict most favorably forwarded first Every node keeps a value FP (initialized to zero) for each message in its queue. Each time the message is forwarded, FP is updated according to Eq. 4, where P is the delivery predictability the receiving node has for the message.

$$FP = FP_{old} + P \tag{4}$$

The message with the highest FP value is the first to be dropped. MOPR can be considered to be a weighted version of MOFO, where instead of increasing a counter by one each time a message is forwarded, it is increased by the delivery predictability of the other node for the destination.

SHLI – Evict shortest life time first In the DTN architecture [2], each message has a timeout value which specifies when it is no longer useful and should be deleted. If this policy is used, the message with the shortest remaining life time is the first to be dropped.

LEPR – Evict least probable first Since the node is least likely to deliver a message for which it has a low P-value, drop the message for which the node has the lowest P-value.

More than one queueing policy may be combined in an ordered set, where the first policy is used primarily, the second policy used only if there is a need to tie-break between messages with the same eviction priority assigned by the primary policy, and so on. As an example, one could select the queueing policy to be {MOFO; SHLI; FIFO}, which would start by dropping the message that has been forwarded the largest number of times. If more than one message has been forwarded the same number of times, the one with the shortest remaining life time will be dropped, and in case of another tie, the FIFO policy will be used to drop the message first received. The use of multiple queueing policies is out of the scope of this paper, and while it is important future work, we consider only one queuing policy at a time in the performance evaluation in this paper.

3.2 Forwarding Strategies

During the information exchange phase, nodes need to decide on which messages they wish to exchange with the peering node. Finite bandwidth and unexpected interruptions may not allow a node to transmit all the messages it would like to forward. In such cases, the order in which the messages are transmitted is important. This section defines the forwarding strategies that we use in our evaluation. Note that if the node being encountered is the destination of any of the messages being carried, those messages should be delivered to the destination irrespective of the forwarding strategy being used. Nodes do not delete messages after forwarding them as long as there is sufficient buffer space available (since it might encounter a better node, or even the final destination of the message in the future), unless the node to which a message was forwarded was its destination.

We use the following notation in our discussions below. A and B are the nodes that meet, and the strategies are described as they should be followed by node A. The destination node is D. $P_{(X,Y)}$ denotes the delivery predictability that a node X has for a destination Y.

GRTR Forward the message only if $P_{(B,D)} > P_{(A,D)}$.

When two nodes meet, a message is sent to the other node if the delivery predictability for the destination of the message is higher at the other node.

GRTRSort Select messages in descending order of the value of $P_{(B,D)} - P_{(A,D)}$. Forward the message only if $P_{(B,D)} > P_{(A,D)}$.

This strategy is similar to GRTR, but it processes the messages in the message queue in a different way. While GRTR scans the queue in a linear way, starting by deciding whether or not to forward the first message, and the continuing like that through the queue, this strategy looks at the difference in P-values for each message between the two nodes, and forwards the messages with the largest difference first. This allows a node to transmit messages with most improvement in delivery predictability first.

GRTRMax Select messages in descending order of $P_{(B,D)}$. Forward the message only if $P_{(B,D)} > P_{(A,D)}$.

This strategy begins by considering the messages for which the encountered node has the highest delivery predictability. The motivation for doing this is the same as in GRTRSort, but based on the idea that it is better to give messages to nodes with high absolute delivery predictabilities, instead of trying to maximize the improvement.

COIN Forward the message only if $X > 0.5$, where $X \in U(0,1)$ is a random variable.

This strategy is similar to the ordinary Epidemic Routing, but to reduce the number of transfers, there is a "coin toss" that determines if a message should be forwarded or not. As this strategy does not consider the delivery predictabilities in making its decision, it will not be used in a PRoPHET system. Nonetheless, it serves as an interesting benchmark to compare the performance of our proposed delivery predictability estimation in [11] to that of a simple random pruning of Epidemic Routing.

GRTRSort and GRTRMax require a node to reorder the messages in its queue for every encounter. This being a regular sorting problem, the worst case complexity is $O(m \log m)$, where m is the number of messages in the queue [3]. By using good datastructures, this could be reduced to $O(n \log n)$ where n is the number of destinations for which there is messages in the queue. Given that the number of nodes in a typical intermittently connected network will range from tens of nodes to utmost a few thousands, the computational overhead should not pose any problem.

4 Simulation Setup

In our evaluation, we have used a simple high level simulator written in Java. The simulator abstracts away the lower layers and focuses on the operation of the routing protocol. The nodes use a wireless communication channel that has a range of 100 meters, and nodes are able to transmit one message every second. We compare the different queueing policies and forwarding strategies presented in Section 3 for PRoPHET and Epidemic Routing.

To ensure that the results of the simulations are useful, it is important that the models that are used are realistic. Considering the transitive kind of communication studied in this paper, it is particularly important to select a good mobility model. In some of the previous work, the authors have used random waypoint mobility or variations of it [16, 19], mobility that has been constrained to some fixed schedule [13, 18], or mobility data gathered from some real-life measurements [8]. In our simulations we have used the community mobility model developed in [11], inspired by the mobility of nodes in a Saami community [4, 15]. As we want to ensure that this model gives us a good representation of real mobility, we evaluated it using the same tests used in previous work that has been done on characterizing properties of human mobility [5]. Doing these tests, we could see that the properties of the community model is relatively close to real mobility and a definite improvement over previous models such as the random way-point mobiliy model.

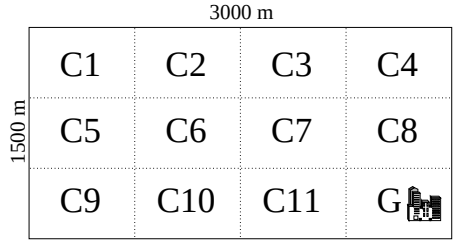


Figure 2: Community Mobility Model

We consider a $3000m \times 1500m$ area as shown in Figure 2. This area is divided into 12 subareas; 11 communities (C1-C11), and one "gathering place" (G). Each node has one home community that it is more likely to visit than other places. For each community there are 10 nodes that recognize it as their home community. In each community, and at the gathering place, there is also a stationary "gateway" (randomly placed within the community) that intermittently generates traffic destined for other communities. The mobility in this scenario is such that nodes select a destination and a speed (randomly chosen between 10 and 30 m/s). On reaching the destination, it pauses for a while, and then selects a new destination and speed. The destinations are selected such that if a node is within its home community, there is a higher probability that it will go to the gathering place as compared to other places, and if it is away from its home community, it is very likely that it will return to the home community. Table 1 shows the probabilities of different destinations being chosen depending on the current location of a node. To ensure that our results are valid in scenarios with other topology and mobility patterns, we have also tested the protocol in a scenario where the random way-point mobility model [7] was used. The results achieved in those tests showed the same trends as the ones presented in this paper, so we believe that our results are valid in more general scenarios than the one presented in this paper.

Table 1: Destination selection probabilities

From \ To	**Home**	**Gathering place**	**Elsewhere**
Home	-	0.8	0.2
Elsewhere	0.9	-	0.1

Every tenth second, two randomly chosen community gateways generate a message for a gateway at another community or at the gathering place. Five seconds after each such message generation, two randomly chosen mobile nodes generate a message to a randomly chosen destination. After 3000 seconds the message generation ceases (after a total of 1200 messages have been generated) and the simulation is run for another 8000 seconds to allow messages to be delivered. A *warm up* period of 500 seconds is used in the beginning of the simulations before message generation commences, to allow the delivery predictabilities of PRoPHET to initialize. Table 2 shows the values for PRoPHET parameters kept fixed in our simulations. These values were chosen based on previous experience with the protocol.

We compare the performance of the various forwarding strategies and queueing policies (and the combinations therein) using the following four metrics. The *message delivery ability*, i.e. the number of messages delivered to their respective destinations, is a primary metric. Applications using this kind of communica-

Table 2: Parameter settings

Parameter	P_{init}	β	γ
Value	0.75	0.25	0.98

tion should be delay-tolerant, but it is still of interest to consider the *message delivery delay* to find out how much time it takes a message to be delivered. Finally, we also study the *overhead* and *weighted overhead* that is incurred by the message exchanges between nodes. The overhead is calculated as the total number of message exchanges between nodes, and the weighted overhead is the number of message exchanges between nodes per message that is delivered to its destination. The size of the delivery predictability information is linearly bounded by the number of nodes in the network (which in most realistic cases will be fairly small), and will be the same regardless of the queueing policy and forwarding strategy used. Thus, this is not considered in the overhead calculation.

To avoid any effects that improper settings of the life time of messages may have on the results of the evaluation (determining the proper life times for messages is out of scope for this paper), messages in our simulations never time out. As the SHLI queueing policy makes its drop decisions based on the remaining life time of messages, we consider each message to have a life time that is longer than the simulation time.

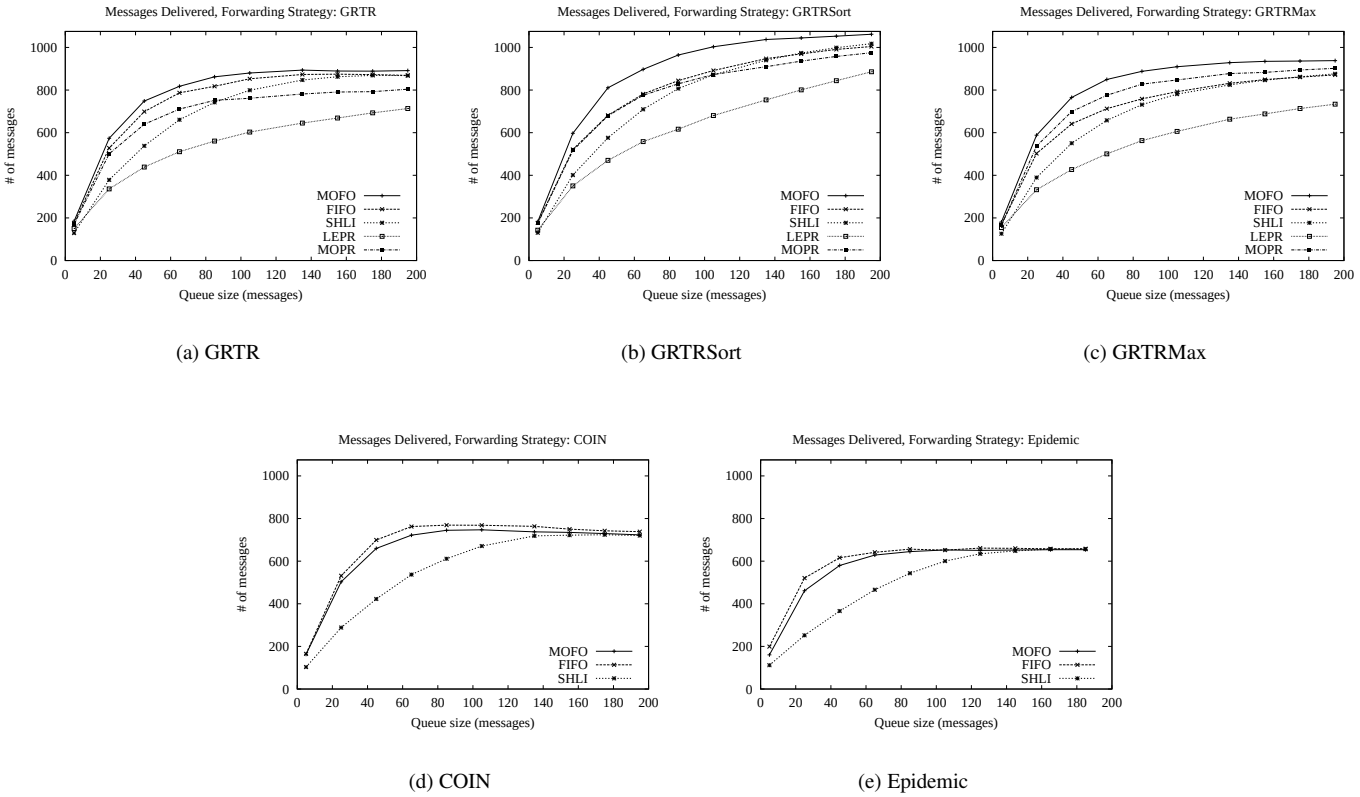
The queueing policies LEPR and MOPR require computation of delivery predictabilities; thus, these policies are not used in conjunction with the COIN and Epidemic forwarding strategies, which do not use delivery predictability in making forwarding decisions. As the delivery predictability calculations are very central in the operation of PRoPHET, and as Epidemic Routing and COIN is mostly included for comparison, it is still important to include these queueing policies in the evaluations where it is possible.

5 Results

The results presented here are averages from 10 simulation runs. We have verified that the results presented are statistically significant at a 95% confidence level (with Bonferroni correction for multiple comparisons) using pairwise t-tests [12]. Unless otherwise stated, the x axis in the graphs show the queue size (the number of messages a node can buffer) in the nodes, and the y axis show the different metrics as outlined above.

In Figure 3, the number of delivered messages for the different simulations is shown. At first, a general observation can be made from these graphs. It can be seen that for all the different queueing policies, performance is better for the forwarding strategies using delivery predictabilities as compared to COIN or Epidemic forwarding. This validates the original assumption we had when designing the protocol about the utility of the probability calculations. It shows that while a simple random pruning of Epidemic Routing does give some performance enhancements, it is not able to achieve the same level of performance as when using the delivery predictabilities as defined in PRoPHET. Among the various queueing policies, the MOFO policy gives the best performance for all different queue sizes considered. At larger queue sizes, the difference between MOFO and some of the other policies decreases. By dropping messages that have already been forwarded to many other nodes, MOFO makes sure that the messages dropped are the ones that have been spread most into the network. This explains the good performance achieved by MOFO as it increases the probability that a message will be able to find its way to the destination, and also reduces the risk that a message is dropped without being forwarded even once. The fact that MOPR is getting worse performance than MOFO is however a sign that it might still be possible to make improvements in the delivery predictability calculations to make them estimate the actual delivery probability even better, as that should intuitively result in good performance for MOPR. As the focus for this work is on forwarding strategies and queueing policies, such estimation improvements will be considered in future work. The FIFO queueing policy performed well especially in presence of forwarding strategies that scan messages for transmission from the head of the queue. For these strategies, given the limited contact opportunity between nodes, messages that are at the head of the queue are likely forwarded more times than those at the tail of the queue. As FIFO drops messages from the head of the queue, messages being dropped are thus also most likely to have been forwarded a larger number of times, which makes this similar to using the MOFO policy.

As mentioned above, the benefit of using delivery predictabilities can be clearly seen in the graphs, and it can also be seen that the added complexity of GRTRMax and, even more so, GRTRSort pays off. The forwarding strategies that are somewhat more advanced than the simple GRTR get better performance, especially GRTRSort which in conjunction with the MOFO queueing policy gives the best performance. A result that may seem surprising at first is that the LEPR queueing policy performs poorly in terms of delivery and delay. Intuitively, dropping messages that the node is not very probable to deliver should be a good way to manage the queue. One problem with this is however that as nodes treat all messages equally, there is a high risk of messages being dropped at the source or close to it due to low delivery predictabilities in that region of the network when only few or even no forwards have been done. Further investigation of the distribution of the number of times a message is forwarded before getting dropped by its source revealed that



(a) GRTR (b) GRTRSort (c) GRTRMax

(d) COIN (e) Epidemic

Figure 3: Average delivery rate

this problem affects LEPR a lot more than the other queueing policies. Figure 4 shows the distribution of the number of times a source forwards a message before dropping it in presence of different queueing policies for an example scenario (other settings exhibit similar behavior). It can be clearly seen that when LEPR is being used, a large amount of the messages are dropped by the source before they have been forwarded a single time. While it might be acceptable for other nodes to do so, such behavior by the source can explain why the policy achieve such bad performance. On the other hand, the figure also shows that with MOFO, sources are able to forward almost all messages at least once before having to drop them, which improves the chance of the messages to be delivered. A possible remedy to this problem that will be considered in future work is to prioritize own messages and drop them only after a certain number of forwards, or if there are no other messages to drop.

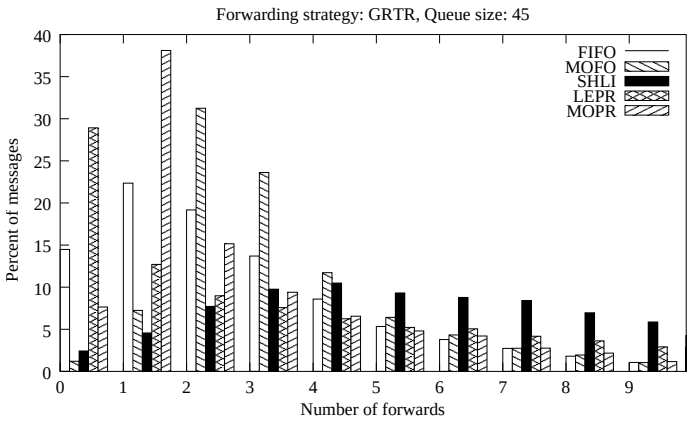
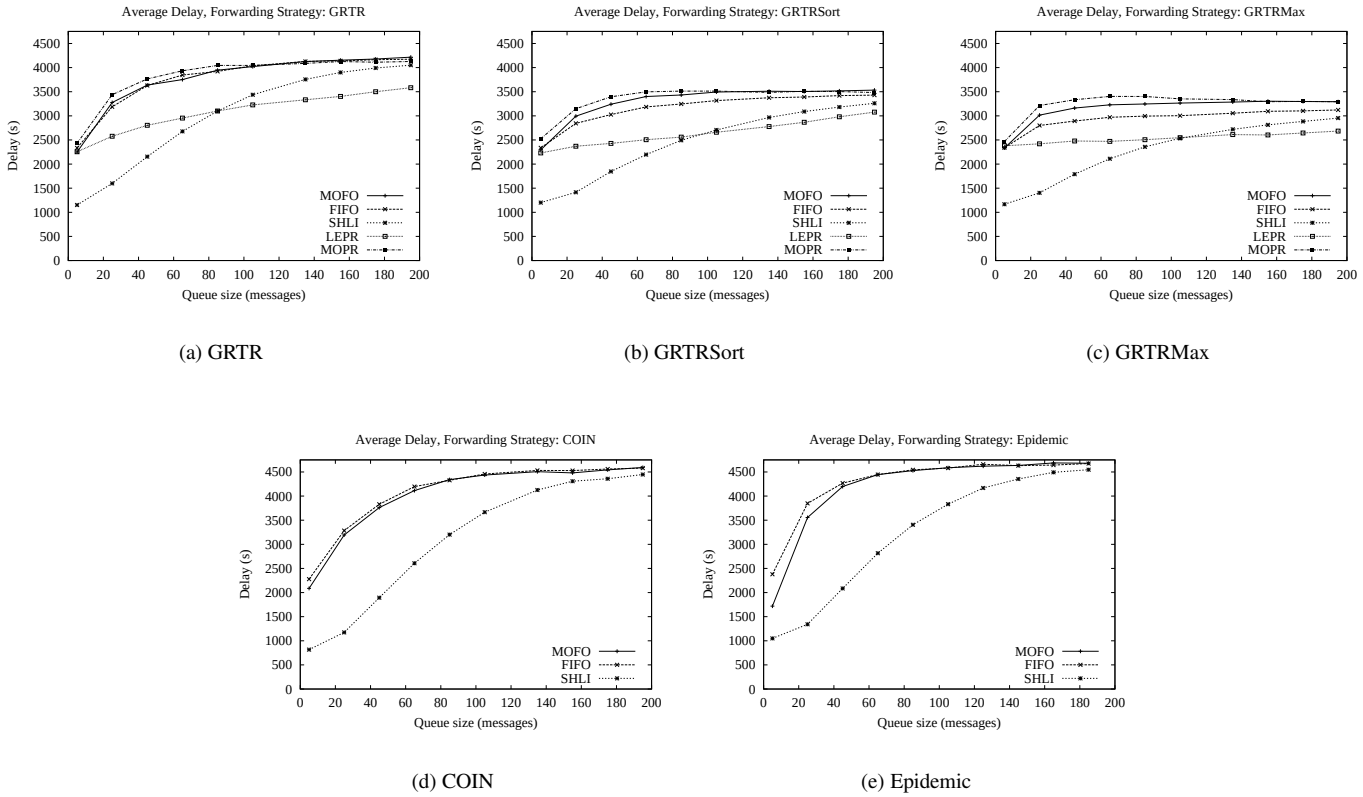


Figure 4: Number of forwards before dropping a message at its source.

In Figure 5, we plot the average delay for the various cases. COIN and Epidemic also have the longest average delays for all different queueing policies. As messages do not only propagate in the probable direction of the destination, but uniformly through the entire network when these strategies are used, contention will be higher in all parts of the network. This additional



(a) GRTR (b) GRTRSort (c) GRTRMax

(d) COIN (e) Epidemic

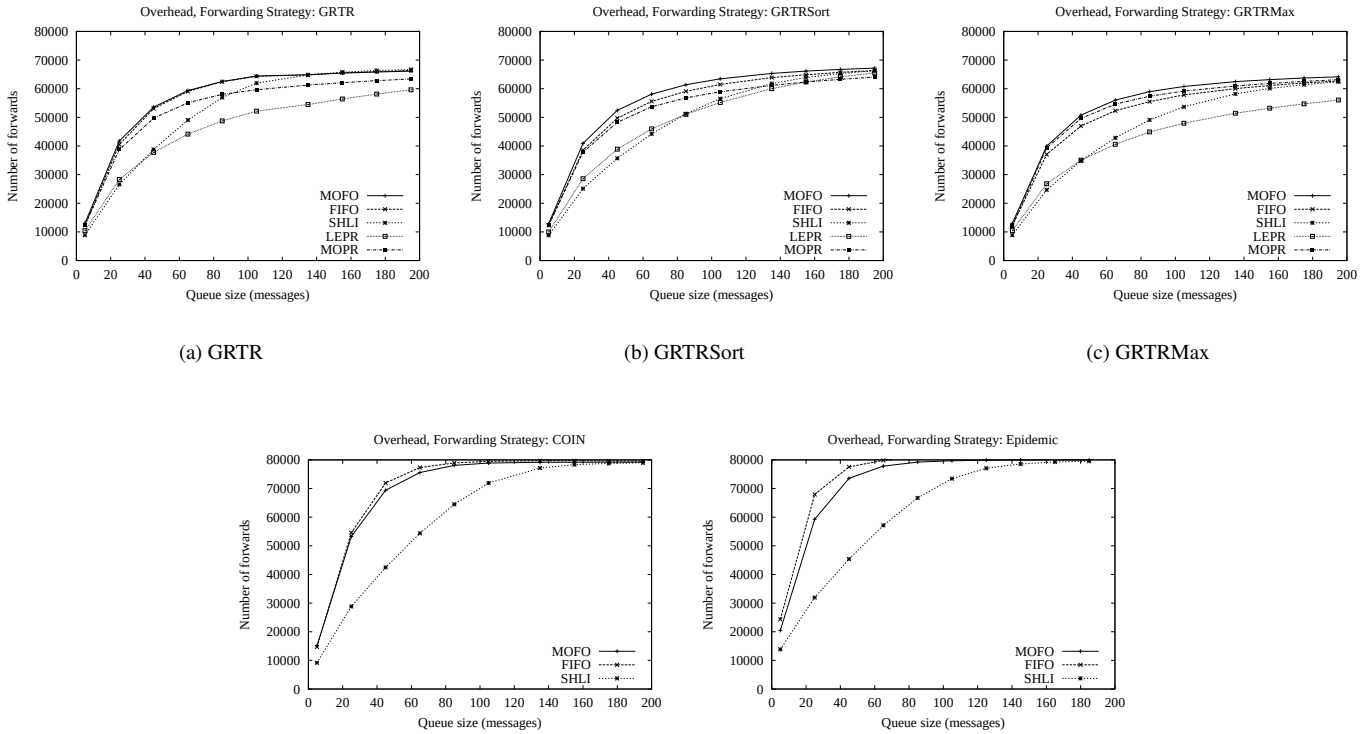
Figure 5: Average delay

contention causes the higher delays that can be seen here. In case of using SHLI, the average delay is very low at short queue lengths and increases almost linearly with the queue length. This is due to SHLI's nature of dropping the oldest messages. At smaller queue lengths, a message is either delivered relatively quickly, or it will get older than the other messages in the queue and thus be dropped.

In Figure 6, we plot the number of forwards that have been made in the network. As expected, the use of Epidemic and COIN result in much more overhead than the other strategies as they are more liberal in forwarding messages. The other forwarding strategies have fairly similar overhead, with GRTRMax being slightly lower. Looking at the results for the various queueing policies it can be seen that MOFO is the one with the largest number of forwards. While this means that more resources are being used for this policy, it should also be noted that it is the policy that managed to get the best delivery rate as well. To illustrate the tradeoff between overhead and delivery ratio, we plot the number of message forwards in the network per successfully delivered message in Figure 7. Despite the extra overhead, MOFO experiences among the lowest number of forwards per delivered message, which in most cases would justify the additional overhead. There could however be scenarios in which nodes are energy-constrained and prefer to trade some of the performance for a reduction in the amount of traffic that needs to be sent.

6 Conclusions and Future Work

In this paper, we have evaluated different queueing policies and forwarding strategies for routing in intermittently connected networks. The results show that there are clear benefits of using delivery predictability calculations instead of just doing a simple random pruning of Epidemic Routing. The evaluation also shows that it is very important to use the system resources in a good way through the selection of queue management policies and forwarding strategies as such choices have a large impact on the performance of the system. While a resource such as storage space often is abundant in many systems, there are still a lot



(a) GRTR

(b) GRTRSort

(c) GRTRMax

(d) COIN

(e) Epidemic

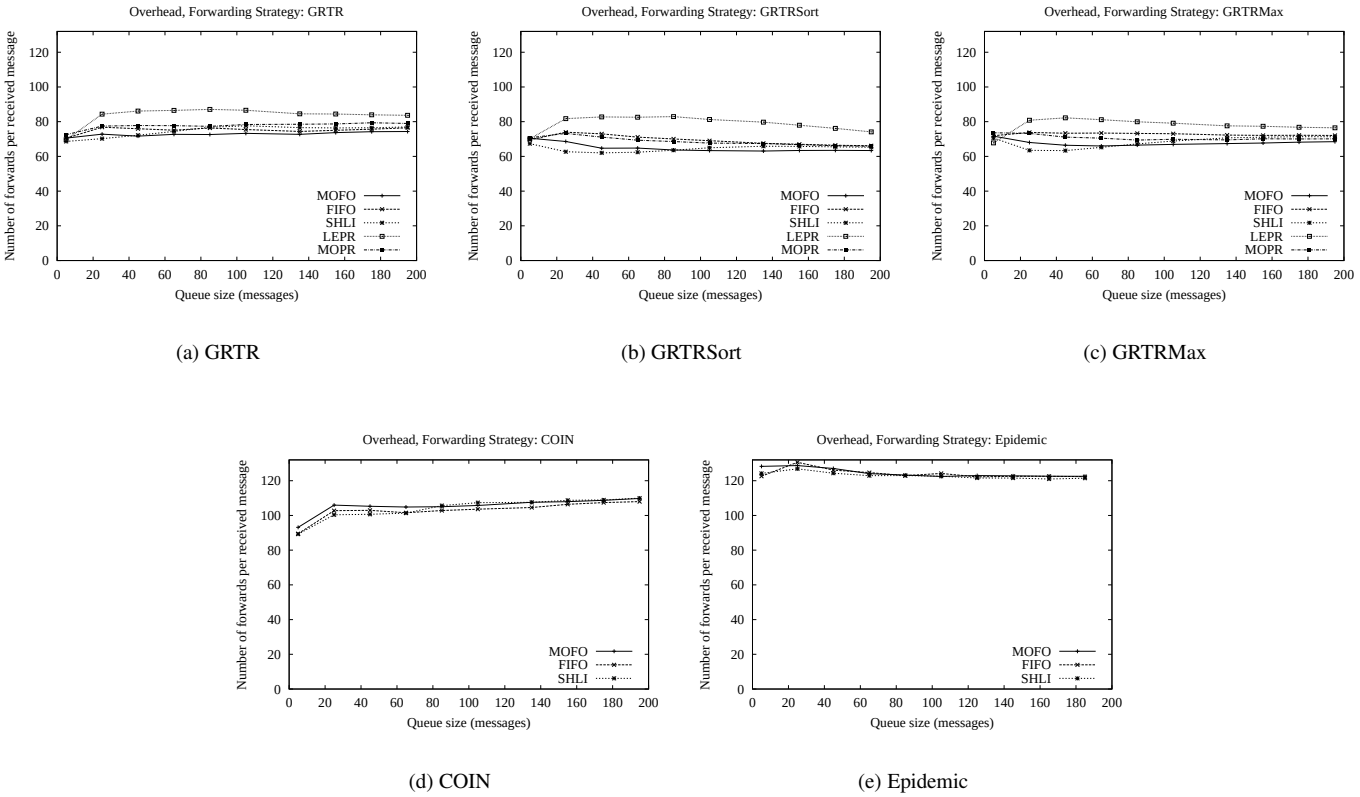
Figure 6: Overhead

of systems (mainly within sensor networking and related fields) where such resources are very limited, and queue sizes cannot exceed a few hundred messages. In such systems it becomes even more crucial to take good care of the limited resources. In designing and selecting a good forwarding strategy, we could see that due to resource constraints such as limited bandwidth, it is not only important to decide which messages to forward, but also to select in which order messages should be forwarded. It is also important to ensure that each message is forwarded a sufficient number of times before its source drops it. This could be clearly seen in the contrast between the MOFO queue management policy in which sources forwarded almost every message at least once, leading to good performance, and LEPR, where many messages were dropped by their source before being forwarded to any other node at all.

From the evaluation presented in this paper, it is evident that the choice of queueing and forwarding policies can greatly impact the performance of routing protocols in a DTN scenario. We plan to extend this work by addressing the issues of congestion control in DTNs. This will involve making routing decisions based on the available buffer space and bandwidth capacity at intermediate nodes. It would be interesting to define queueing policies and forwarding strategies that take into account the size of the messages in the queue. For example, if a transmission opportunity is estimated to last a certain amount of time, the end-to-end delay versus message delivery trade-off could be influenced by the choice of forwarding several small messages, or one large message that occupies the entire transmission opportunity. More effort will also be put into further understanding the effects of topology, mobility, and the ability to make good delivery predictability estimates.

References

[1] Allan Beaufour, Martin Leopold, and Philippe Bonnet. Smart-tag based data dissemination. In *First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA02)*, June 2002.



(a) GRTR (b) GRTRSort (c) GRTRMax

(d) COIN (e) Epidemic

Figure 7: Weighted Overhead

[2] Vinton G. Cerf, Scott C. Burleigh, Adrian J. Hooke, Leigh Torgerson, Robert C. Durst, Keith L. Scott, Kevin Fall, and Howard S. Weiss. Delay-tolerant network architecture. Internet Draft draft-irtf-dtnrg-arch-02.txt (work in progress), March 2003.

[3] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. The MIT Press, 1997.

[4] Avri Doria, Maria Udén, and Durga Prasad Pandey. Providing connectivity to the saami nomadic community. In *Proceedings of the 2nd International Conference on Open Collaborative Design for Sustainable Innovation (dyd 02), Bangalore, India*, Dec 2002.

[5] Pan Hui, Augustin Chaintreau, James W Scott, Richard Gass, Jon Crowcroft, and Christophe Diot. Pocket switched networks and the consequences of human mobility in conference environments. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, August 26 2005.

[6] Interplanetary internet special interest group (IPNSIG), Dec 2002. http://www.ipnsig.org.

[7] David B. Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353, chapter 5, pages 153–181. Kluwer Academic Publishers, 1996.

[8] Evan P. C. Jones, Lily Li, and Paul A. S. Ward. Practical routing in delay-tolerant networks. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, New York, NY, USA, 2005. ACM Press.

[9] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li-Shiuan Peh, and Daniel Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proceedings of Tenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X), San Jose, CA*, October 2002.

[10] Anders Lindgren and Avri Doria. Probabilistic routing protocol for intermittently connected networks. Internet Draft draft-lindgren-dtnrg-prophet-00.txt (work in progress), June 2005.

[11] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. In *Proceedings of The First International Workshop on Service Assurance with Partial and Intermittent Resources (SAPIR 2004)*, August 2004.

[12] Douglas C. Montgomery. *Design and Analysis of Experiments*. John Wiley & Sons, New York, 4th edition, 1997.

[13] Alex Pentland, Richard Fletcher, and Amir A. Hasson. A road to universal broadband connectivity. In *Proceedings of the 2nd International Conference on Open Collaborative Design for Sustainable Innovation (dyd 02), Bangalore, India*, Dec 2002.

[14] Tara Small and Zygmunt Haas. The shared wireless infostation model - a new ad hoc networking paradigm (or where there is a whale, there is a way). In *Proceedings of The Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003)*, pages 233–244, June 2003.

[15] Saami network connectivity (SNC) project, March 2005. http://www.snc.sapmi.net/.

[16] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259, New York, NY, USA, 2005. ACM Press.

[17] Amin Vahdat and David Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University, April 2000.

[18] Randolph Y. Wang, Sumeet Sobti, Nitin Garg, Elisha Ziskind, Junwen Lai, and Arvind Krishnamurthy. Turning the postal system into a generic digital communication mechanism. In *Proceedings of ACM SIGCOMM 2004*, August 30 – September 3 2004.

[19] Wenrui Zhao, Mostafa Ammar, and Ellen Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC 2004)*, May 24–26 2004.