Message Fragmentation in Opportunistic DTNs

Mikko Pitkänen[1], Ari Keränen[2] and Jörg Ott[2]
[1]Helsinki Institute of Physics, Technology Programme
[2]Helsinki University of Technology, Department of Communications and Networking

Abstract

Delay-tolerant networking is used for communication in challenged environments such as MANETs, in which links are unstable and end-to-end paths between communicating nodes may not exist. Messages may be significantly larger than packets in IP networks. Large messages lead to longer transfer times rendering it more likely that a link breaks in the middle of a message transfer. This motivates investigating how to support partial message transfers through fragmentation. In this paper, we formulate fragmentation independent of routing algorithms, introduce several fragmentation strategies, and evaluate these by simulations to derive recommendations for using fragmentation in DTNs.

1 Introduction

Delay-tolerant Networking (DTN) refers to communication in challenged environments where traditional instant packet forwarding mechanisms fail due to, e.g., high delays, intermittent connectivity, or non-existing end-to-end paths. Unlike Internet routers, which forward packets instantaneously (or discard them if no route is found), DTN nodes may store packets for an extended period of time until the next suitable forwarding opportunity, called *contact*, becomes available. Mobile DTN nodes may physically carry packets while stored. DTNs may comprise a network infrastructure (e.g., for space communication) which may be fixed or mobile, form mobile ad-hoc networks (e.g., for interpersonal communications), or be a combination of both. While IP networks run on "small" packets in conjunction with end-to-end transport protocols to exchange application data, DTN nodes may send messages of arbitrary size, also termed *bundles*, using hop-by-hop reliability mechanisms to deal with potentially large end-to-end delays.

In IP networks, the link MTU size determines whether a given packet can be transmitted toward a particular next hop. If the packet is too large, it is fragmented into several pieces (provided that fragmentation is allowed). A link will usually last orders of magnitude longer than the transmission of an IP packet takes, so that the MTU size limitation is the only reason for IP layer fragmentation.

With DTNs and arbitrary message sizes, no notion of an MTU has been defined so far. But since messages may be large (many megabytes, even gigabytes), the *contact duration* may easily become the limiting factor for message forwarding: assuming fast contact establishment, it will take some ten seconds to reliably transmit a message of 1 MB over a 1 Mbit/s wireless link. Links in DTNs may not last long enough to even transmit just a single message. Hence, support for partial message transfer may enable communication over short-lived links and avoid wasting link capacity with incomplete transmissions.

This motivates fragmentation for DTNs. The DTN architecture [1] defines two types of fragmentation:

1) With *proactive fragmentation*, a node splits a bundle into fragments *prior* to transmitting it—which is similar to IP fragmentation. The fragmentation decision may be based upon knowledge of the link availability or account for buffer limitations on the next hop, among others.

2) For *reactive fragmentation*, the forwarder starts transmitting an entire bundle and, when interrupted by a sudden link failure, the forwarding node and the next hop reconcile the remaining (i.e., not yet transmitted) and the already received portions into valid bundles.[1]

In this paper, we provide an analysis on the impact of fragmentation mechanisms in DTNs, focusing on opportunistic environments. After reviewing related work in section 2, we define fragmentation and its interaction with routing and forwarding, introduce evaluation metrics, and provide some intuition on its expected impact in section 3. We present our simulation results in section 4 and conclude this paper with a brief assessment and some recommendations on using fragmentation in mobile DTNs in section 5.

2 Related Work

Delay-tolerant Networking has been applied to various challenging networking environments, including space communications, sparse sensornets, and opportunistic mobile ad-hoc networks. The DTN architecture [1] proposes a *bundle layer* [3] to span different (inter)networks. Messages—*bundles*—of arbitrary size and finite TTL are forwarded hop-by-hop between DTN nodes (*bundle routers*) as best-effort traffic.

[1]It is important that both forwarder and next hop agree on how much data was successfully forwarded to avoid any losses. We assume that this is achieved by a suitable convergence layer, e.g. [2].

978-1-4244-2100-8/08/$25.00

The bundle protocol specification defines the syntax and handling of bundles and also includes fragmentation: fragments are represented as bundles which can be fragmented further (like in IP). While the syntax is in place, little guidance is available on how and when to use fragmentation. Most routing protocols defined so far consider transferring bundles only in an all-or-nothing fashion (e.g., MaxProp [4], Spray-and-Wait [5], Prophet [6], and RAPID [7]), which makes routing and forwarding design simpler. Exceptions are protocols fragmenting messages to add redundancy for increased robustness of message delivery, as seen with erasure- or network-coding [8, 9, 10].

For IP networks, fragmentation has been discouraged for various reasons [11], including increased packet loss probability, and is generally avoided. For datagram-based communications (which is closest to the DTN-style messaging), it is usually up to the application layer to segment application data units so that they do not exceed the path MTU size, following the concept of *Application Layer Framing* [12], which is prominently used for RTP media streams [13, 14]. A similar approach may be pursued with DTNs: an application may also limit the impact of a challenged environment by using smaller messages in the first place.

Application level message fragmentation has been proposed for overcoming limited communication and storage resources in store-and-forward networks already in the late 1980s [15]. The authors discuss many issues also applicable to communication scenarios targeted in our paper, however, they assume high reliability of the message transfers and their intermediaries communicate over more stable links.

Transferring large resources in small messages without explicit acknowledgments may lead to reliability degradation in best-effort networks. For example, in sparse opportunistic networks exploiting human mobility, the observed message delivery ratio is often very low, and applying fragmentation at the source leads to further deterioration which can be countered by adding redundancy [16]. A digital fountain [17] has been proposed to distribute information in a way that lost fragments of the resource can be regenerated from the received fragments with low overhead. Recent research has also proposed coding methods [18] to be used in zero-configuration sensor networks to increase the amount of data that reaches the destination.

While coding and other forms of redundancy increase the data volume representing a particular message in the network (and hence contribute to congestion), applying plain fragmentation does not. In the following, we strive to better understand the impact of different types of fragmentation on DTNs, specifically aiming at MANET environments.

3 Fragmentation in Mobile DTNs

We consider a sparse network of nodes N_i which send, forward, and receive messages m. We refer to the originator of a message as N_s and the destination as N_d. A message is forwarded by intermediate nodes until its time-to-live expires or it is discarded, e.g., due to congestion.

A message m contains a message header ***H(m)*** with control (source, destination) and payload-related information (size, fragmentation offset and granularity, checksum, etc.); a routing-specific header ***R(m)*** for forwarding and deletion decisions (including time-to-live, and other information, if any, needed by the routing protocol); and the payload U_m of size $S(U_m)$ containing the application data (a *resource*).

When fragmenting, a node N_i divides the bundle payload—which may be the complete resource or a resource fragment—into two or more non-overlapping fragments $F_1, F_2, ..., F_n$ as show in figure 1. A fragment of the original payload U_m is denoted as $U_m[a, b]$ which indicates that the fragment starts at offset a, ends at offset b (both inclusive), and is of size $b - a + 1$; with $U_m = U_m[0, S(U_m) - 1]$. ***H(m)*** and ***R(m)*** are copied into each of the resulting fragments; fragmentation updates the fragmentation offset and size in ***H(m)*** for each fragment, but does not modify ***R(m)***. Subsequent message or fragment forwarding will update ***R(m)*** according to the routing protocol (indicated by ***R'(m)*** and ***R"(m)*** in figure 1), but not change ***H(m)***. No information about the fragmenting node(s) is included in the resulting fragments.

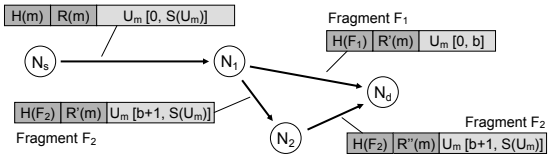

Figure 1. Message fragmentation example

Fragments are subsequently treated as independent bundles—for routing, forwarding, buffer management, etc.—and are reassembled only at their final destination.[2] For re-assembly, fragments are collected and maintained until their time-to-live expires (which is provided in absolute time, copied upon fragmentation, and hence will be identical for all fragments). If multi-copy routing algorithms are used, several fragments may overlap—entirely or in part, depending on the fragmentation scheme—in which case we assume the overlapping parts to be consistent.[3] The message is considered delivered only if the destination can recover the entire resource from the received.

3.1 Proactive Fragmentation

Proactive fragmentation may occur at any node; the decision is taken *prior* to message forwarding to the next hop and may be based, e.g., on knowledge about the expected

[2]Merging the fragments in the intermediaries raises many questions, e.g., about routing information (how to merge two different routing headers?), and is beyond the scope of this paper.

[3]This could be verified by integrity protecting the entire message.

uptime of the link(s) to the next hop or along the path. A special case is *source fragmentation*: the resource (=payload) is passed from the application to the bundle layer, where the message *H(m)* and routing headers *R(m)* are created. The originating node N_s splits the payload into n non-overlapping fragments of approximately equal size (which is our choice in this paper, but need not be the case), to which the headers are prepended and adapted as needed. Then, N_s sends out the fragments sequentially, starting with the beginning of the payload. Intermediary nodes forward the fragments unchanged, i.e., no further fragmentation takes place. At the destination node (N_d), all n fragments are needed to re-assemble the resource.

In this paper, we only consider proactive fragmentation at the source, not at intermediaries as, with proper link uptime estimates, the latter is close to reactive fragmentation.

3.2 Reactive Fragmentation

For reactive fragmentation, any two nodes N_i and N_j in the network are able to fragment a message with a payload larger than a single byte. Fragmentation is triggered when a connection breaks during a message transfer from N_i to N_j. From the message headers (assumed to be transmitted first) and the payload part of the message m received by N_j before connection breakdown, one fragment, F_1, is created. The part of the message not yet delivered to N_j is transformed into the other fragment F_2.[4] Both fragments inherit the headers from m, with the fragmentation information (offset, payload size) adjusted; $R(F_1)$ is updated to reflect the successful forwarding while $R(F_2)$ remains unchanged. The unfragmented bundle at the forwarding node is discarded and the two fragments proceed to exist independently; this allows maintaining different routing headers and keeps fragmentation independent of the routing protocols. Fragments may be fragmented again.

We also investigate a variant of reactive fragmentation in which the size of the resulting fragments is not arbitrary, but limited to pre-determined boundaries (defined by the originator). This is often referred to as *toilet paper* approach [19], and allows, e.g., for fragment authentication [20]. Fragmentation boundaries also increase the probability that fragmenting copies of a single resource in different paths result in identical fragments. As only identical fragments are detected as duplicates, this may help duplicate detection and thus reduce redundant traffic.

3.3 Fragmentation and Routing

We do not invent a new routing scheme, but introduce message fragmentation to existing ones. The basic fragmentation information is part of *H(m)* and common across all routing protocols, whereas routing headers *R(m)* are protocol-specific.[5] Fragmentation takes place in the queues of the forwarding node and the next hop as described above to minimize the interaction with routing.

We explore the impact of fragmentation on six routing protocols. Two use single copy approaches, *Direct Delivery* and *First Contact* [21]; two perform variants of flooding, *epidemic* [22] and *Prophet* [6]; *MaxProp* [4] floods but explicitly clears messages once delivered; and *Spray-and-Wait* [5] limits the number of copies created per message.

For proactive (source) fragmentation, the source creates n fragments instead of one message. As complete messages and fragments do not differ from a routing point of view, no routing-specific actions are needed.

With both types of reactive fragmentation, no changes are needed for the single-copy algorithms: one part F_1 has been passed to the next hop, the other F_2 is kept at the forwarding node. For multi-copy routing schemes, if fragmentation occurs, the original message m ceases to exist in the forwarding node and is replaced by two pieces, F_1 and F_2, which inherit the forwarding properties of the original (e.g., the number of remaining copies with Spray-and-Wait). Subsequently, only F_1 and F_2 will be disseminated by the forwarding node and their routing headers will be updated independently. This is not relevant for *epidemic* and *Prophet*, but for *Spray-and-Wait* and *MaxProp*. As node pairs operate independently, this may result in a mixture of overlapping fragments of different sizes which will only be detected as duplicates if the byte ranges of the fragments match exactly.

3.4 Metrics and Expected Impact

To evaluate the impact of the fragmentation in DTNs, we consider two metrics: the *delivery ratio* and the *delivery latency* of the messages sent during the simulation. In our simulations, we use a warm up period of 30 minutes to reach a steady state; all messages created during this period are ignored. A full simulation lasts for 6 simulated hours. The messages are sent one way from the source to the destination. For the delivery ratio, we relate the number of unique delivered messages to the number of messages sent. In addition, we consider fragment delivery statistics to see whether there is some bias on the successful delivery of fragments, i.e., whether some parts of a message are delivered with larger probability than the others. Latency (or delay) is measured as the time between message creation and delivery. Again, we take a look at the latency distribution of individual fragments.

Fragmentation is expected to have both positive and negative impacts on the message delivery ratio. As discussed in the introduction, fragmentation should be beneficial as soon as contact durations become the limiting factor since, without fragmentation, the shortest contact duration along

[4]Note that the payloads of the two fragments may overlap if parts of the payload were received by N_j but the corresponding acknowledgments did not reach N_i; however, we do not consider this case in our simulations.

[5]For the bundle protocol specification [3], most of this is part of the primary block, further fragmentation information and routing-specific headers *R(m)* can be added as separate extension blocks.

a path would determine the upper size limit for messages along this path. With proactive fragmentation at the source, such a limit no longer applies to the message but to individual fragments—yet, the maximum acceptable message size will usually be unknown. In contrast, reactive fragmentation allows precisely matching the per-link limit. In either case, this aspect of fragmentation is expected to increase the delivery ratio because, given a certain message size, fragmentation allows exploiting more (shorter) contact opportunities and decreases the fraction of unsuccessful message transfers due to interruptions. Thus, fragmentation increases the connectivity of the network.

However, as noted above, messages in DTNs often exhibit poor delivery ratios, particularly in opportunistic networks. If a message is split into two fragments, both need to arrive for successful message delivery; if either of them is lost, so is the entire message.[6] Message loss may be the result of 1) misguided routing leading to message expiration prior to delivery, 2) discarding at a node due to congestion, or 3) node loss (e.g., due to battery depletion).

In a network setup with deterministic routing (e.g., determined via a link-state algorithm as in [21, 23]), ideally, no messages will be misrouted or delayed unnecessarily. In opportunistic networks, we assume the probability p_x that a next hop N_x will lead to the destination N_d in time, with $p < 1$ unless $N_x = N_d$. Hence, even in an uncongested network, if two fragments are forwarded to different next hops N_a and N_b (which will only be the case if the contact opportunity is too short for sending both to the same next hop), the delivery probability will be $p = p_a \times p_b$. Since two paths must work, the delivery ratio will be lower than without fragmentation.

In any case, bundles are subject to congestion-induced deletion at each node. Assuming a message deletion probability p, the delivery probability of message which needs to traverse k nodes is $(1-p)^k$. If the message is split into two fragments, each with a deletion probability q (different since they are smaller), the delivery probability is $(1-q)^{2k}$. As the routing protocols used in our simulations do not make discard decisions as a function of the message size (but rather of message age), we have $p = q$. Hence, the delivery ratio is expected to decrease the more hops have to be traversed and the more fragments are created.

In summary, intuitively, fragmentation should improve the delivery ratio in scenarios with with short contacts. But messages should be fragmented as late as possible so that the impact of misrouting, node loss and congestion-incurred deletion is minimized. This argues in favor of reactive fragmentation being superior compared to any other approach.

[6]For illustration purposes, we stick to a simple example with two fragments and single copy routing here. But even with the multi-copy case, every discarded bundle reduces the chances of "its" resource being delivered. Similar considerations apply when using more than two fragments since all parts of the resource need to be received at the destination.

4 Evaluation

We run several simulations using the Opportunistic Networking Environment (ONE) simulator [24]. We use two different mobility models:[7] The more realistic map-based model, the Helsinki City Scenario (HCS), has nodes moving in a part of the downtown Helsinki area. For comparison, we choose the Random Way Point (RWP) using an area of 1 km×1 km. Nodes connect using bi-directional Bluetooth links at 2 Mbit/s and have 100 MB FIFO message buffer; the oldest messages are dropped first (except for MaxProp which has its own buffer handling scheme). Messages are sent with a time-to-live of 120 minutes. We use the binary version of Spray-and-Wait and allow for 10 message copies. Messages delivered to the destination are handed over to the application and immediately stop occupying buffer space.

Messages of the same size (500 KB–5 MB) are sent from a random source to a random destination at intervals uniformly distributed in the range $[25, 35]$ seconds. In one series of simulations, this interval is maintained across all message sizes (thus increasing the offered load), in another it is used only for the smallest message and increased for larger ones so that the total offered load is constant. The delivery ratio and average delay are recorded for all delivered messages. We record these metrics for unfragmented messages and for three different fragmentation mechanisms: proactive (using 3, 5, and 10 equally sized fragments, but we report details only for the 3 fragment case), reactive, and toilet paper (with 100 KB fragments).

4.1 Helsinki City Scenario (HCS)

With HCS, node mobility is based on simulating 80 mobile users moving by foot, 40 by car, and 6 by trams in the streets downtown Helsinki. Each node represents a user moving with realistic speed along the shortest paths between different points of interest (POIs) and random locations. The nodes are divided into four different groups having different POIs and different, pre-determined probabilities to choose a next group-specific POI or a random place to visit. The trams follow real tram routes in Helsinki.

Figure 2 illustrates the message delivery ratios; their performance in decreasing order is: MaxProp, Spray-and-Wait, epidemic, Prophet, direct delivery, and first contact. With all the routing protocols (except for first contact), both types of reactive fragmentation consistently show improved delivery ratio compared to the non-fragmented case, whereas proactive fragmentation constantly performs worse. The reactive toilet paper approach performs slightly better than reactive fragmentation without predefined boundaries; we attribute this to the to better duplicate detection leading to better buffer utilization and reduced congestion losses. While

[7]We also considered the real-world traces from the CRAWDAD archive, but their typical temporal resolution for contact durations is rather coarse; e.g., the Haggle traces use 5 min granularity.

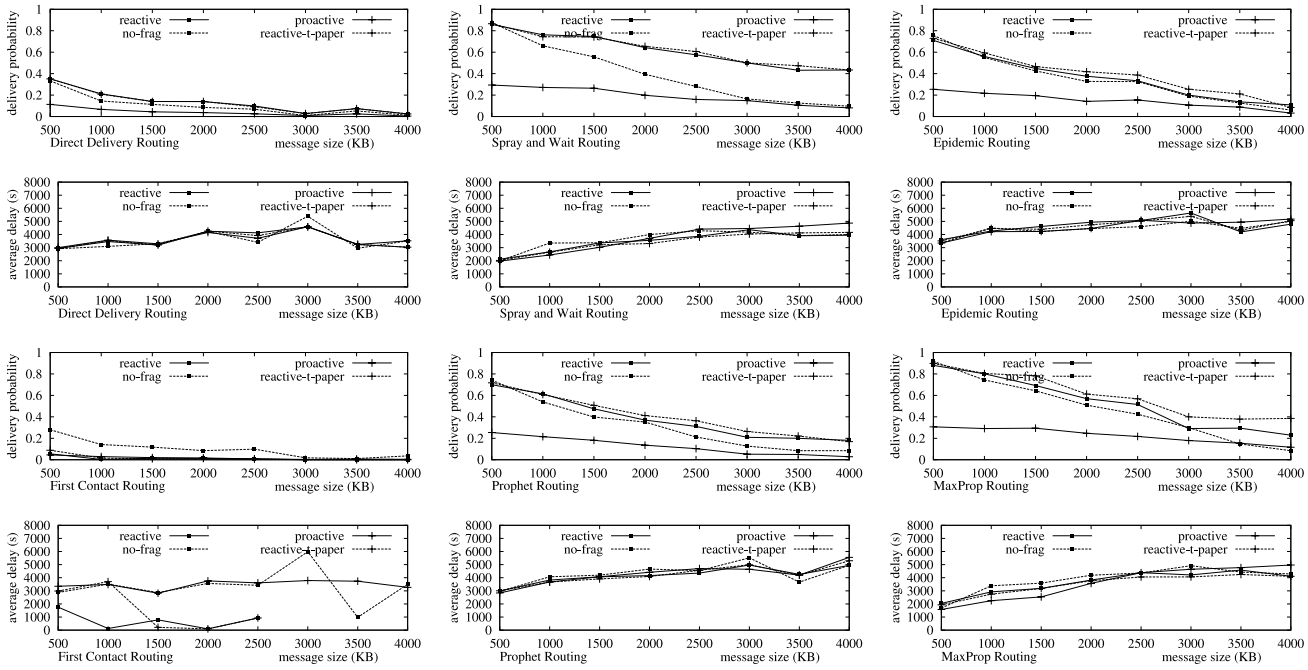

Figure 2. HCS mobility. Delivery probabilities (1st and 3rd row) and latencies (2nd and 4th row).

reactive fragmentation improves the delivery, we also observe that it becomes less effective with increased message sizes: in this scenario (see figure 4 left), 80% of the contact times are less than 6 s and one third less than 2 s (allowing the transfer of 1.5 MB and 500 KB, respectively), more fragments are created, but not all of them delivered. For multi-copy routing protocols performing buffer management (Spray-and-Wait, MaxProp), the gain achieved by fragmentation increases with the message size due to less congestion and, consequently, fewer fragment losses.

The fragment size distribution for both reactive fragmentation schemes reflects the contact time distribution and amplifies it since messages are not reassembled on the path (figure 4 center): across all message sizes, 80–90% of the fragments are smaller than 400 KB in size (equivalent to 2 s transmission time). This shows how short contact durations lead to a need to perform fragmentation when allowed.

The different fragmentation mechanisms impact the message delivery ratios but the latencies are fairly similar across all approaches (except for direct delivery where it is pure coincidence when source and destination meet). The latency increases slightly with the offered load for all multi-copy routing protocols (not shown). This suggests that these are determined by the overall queue lengths in the system.

4.2 Random Way Point (RWP)

Figure 3 illustrates our simulations results with RWP node movement. The number of nodes and other parameters of the simulation are unchanged from the HCS case except that now nodes walk slowly (at 0.5–1.0 m/s) in an area of 1 km^2 without obstacles. Figure 4 (left) shows that this scenario creates much longer contact durations, allowing transfer of larger messages.

We observe that fragmentation is no longer able to improve the delivery ratio, but in most cases leads to a (slight) decrease whereas the delivery latency, again, remains roughly the same across all schemes. With longer contact duration for RWP (median 13 s, mean 16.9 s) compared to HCS (median $<$ 2 s, mean 4.7 s), nodes can afford waiting for a future (sufficiently long) contact to forward their messages in one piece. Hence, fragmentation is less vital to message delivery and its connectivity improvement cannot make up for potential fragment loss due to misrouting or deletion (no node loss). This is confirmed when looking at the fragment size distribution for Spray-and-Wait (figure 4 right): A significant number of messages can be delivered unfragmented even if fragmentation is enabled.

4.3 Parameter Variations

The above simulation scenarios allow for many variations. We have run both HCS and RWP both with constant load and constant message generation intervals: this yields qualitatively the same observations but leads (expectedly) to a stronger performance degradation with increasing message size and to a less pronounced difference between the schemes under high load.

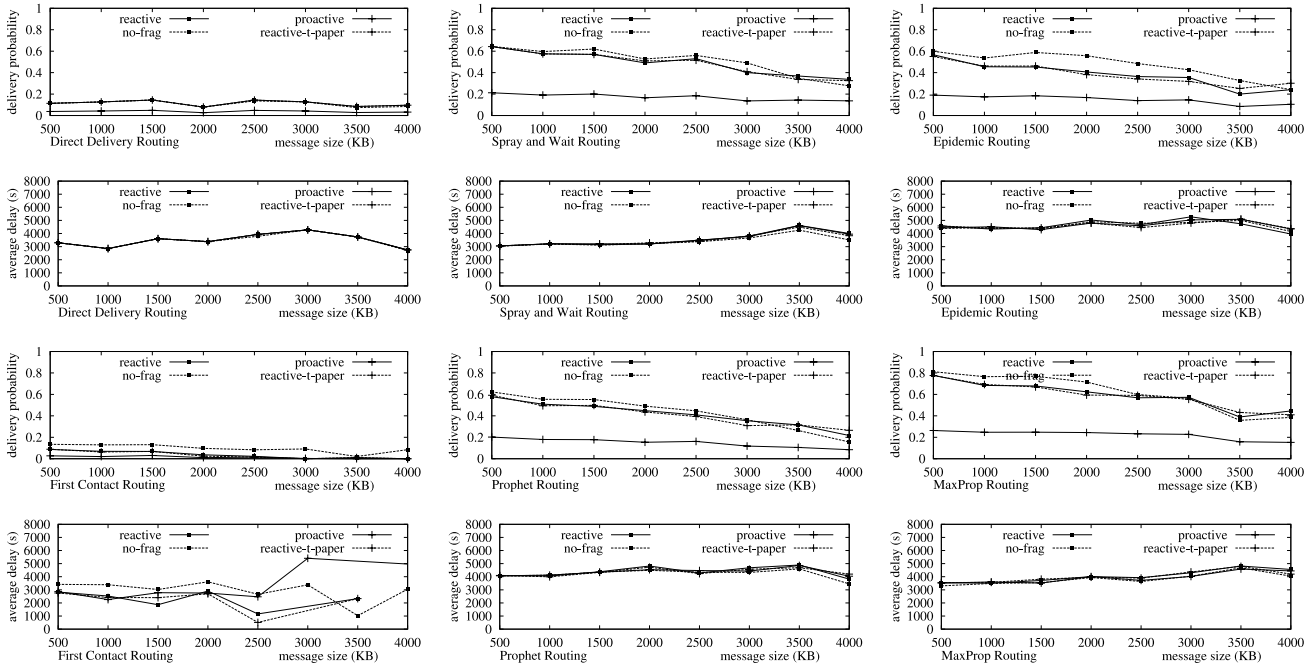

Figure 3. RWP mobility. Delivery probabilities (1st and 3rd row) and latencies (2nd and 4th row).

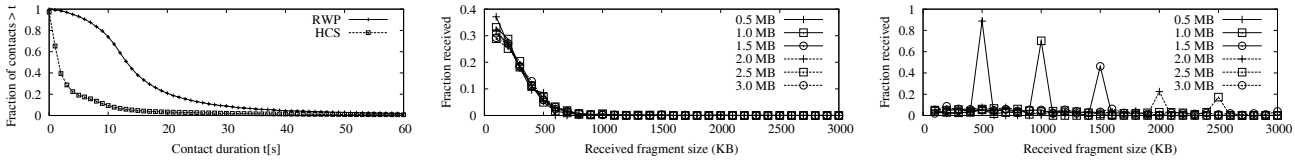

Figure 4. Contact duration (left); received fragment size distribution: HCS (center) and RWP (right)

For proactive (source) fragmentation, we experimented with different numbers of equal-sized fragments per resource: using 5 or 10 fragments resulted in a significant decrease in delivery ratio.

We have carried out a first set of simulations with an uncongested network for all the above scenarios by sending two messages per hour—which effectively eliminates the congestion losses. The simulation results indicate that early fragmentation may have a negative impact: proactive (source) fragmentation appears to suffer from spreading fragments across multiple paths, whereas the reactive toilet paper approach (which prevents the creation of small fragments and hence limits path diversity) yields the best results. Generally, fragmentation expectedly improves the delivery rate of large messages.

We also investigated which fragments of a message are received by the destination (not counting duplicates). The FIFO queuing resulted in the parts of a message sent first by the source to be delivered more frequently with both types of reactive fragmentation; it was less strong with proactive fragmentation. We found this bias across all routing protocols and introduced random ordering of messages and fragments in each intermediate node buffer to overcome this bias. However, the effect was observed as long as the transmission order at the source remains the same.

5 Conclusion and Next Steps

We have investigated the effect of fragmentation on the message delivery success in DTNs. In our simulations, reactive fragmentation appears to be a suitable tool for improving the delivery ratio of messages while the communication latency remains the same. When using predefined fragmentation boundaries, our simulations show slightly better performance than plain reactive fragmentation: it helps reducing congestion and avoiding small ("silly") fragments. The effects are largely independent of the routing protocol and so is the implementation of fragmentation which can be done simply for messages in the queue.

However, the prerequisite is that, for a given message size and contact duration distribution, fragmentation effectively increases overall connectivity. If alternative routes

can be found which last sufficiently long for the entire message, premature message fragmentation (on the first contacts) makes the negative aspects dominate so that, like with IP networks, fragmentation may be considered harmful. Whether or not fragmentation is appropriate thus depends on the scenario and, lacking prior knowledge, may need to be learned dynamically by the node.

While we have looked into numerous aspects of fragmentation, its performance related to various levels of congestion, further mobility scenarios, routing protocols, and the use of fragmentation for buffer management should be investigated. If it is confirmed that the first fragments of messages are delivered with a higher ratio, exploring partial message delivery peered with suitable application protocols will be an interesting avenue to pursue.

Acknowledgments

This research was funded by Nokia Research Centre in the SINDTN project and by the Academy of Finland in the DISTANCE project (grant no. 117429).

References

[1] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H.Weiss, "Delay-Tolerant Network Architecture," RFC 4838, 2007.

[2] M. Demmer and J. Ott, "Delay Tolerant Networking TCP Convergence Layer Protocol," Internet Draft draft-irtf-dtnrg-tcp-clayer-01, Work in Progress, 2007.

[3] K. Scott and S. Burleigh, "Bundle Protocol Specification," RFC 5050, November 2007.

[4] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks," in *Proc. IEEE INFOCOM*, 2006.

[5] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks," in *Proc. ACM SIGCOMM WDTN Workshop*, 2005.

[6] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," in *SAPIR Workshop*, 2004.

[7] A. Balasubramanian, B. Levine, and A. Venkataramani, "DTN Routing as a Resource Allocation Problem," in *Proc. ACM SIGCOMM*, 2007.

[8] Y. Wang, S. Jain, M. Martonosi, and K. Fall, "Erasure-Coding Based Routing for Opportunistic Networks," in *Proc. ACM SIGCOMM WDTN Workshop*, 2005.

[9] S. Jain, M. Demmer, R. Patra, and K. Fall, "Using Redundancy to Cope with Failures in a Delay Tolerant Network," in *Proc. ACM SIGCOMM*, 2005.

[10] J. Widmer and J-Y. Le Boudec, "Network Coding for Efficient Communication in Extreme Networks," in *Proc. ACM SIGCOMM WDTN Workshop*, 2005.

[11] C. A. Kent and J. C. Mogul, "Fragmentation considered harmful," in *Proc. ACM SIGCOMM Workshop on Frontiers in computer comm. technology*, 1987.

[12] D. D. Clark and D. L. Tennenhouse, "Architectural Considerations for a new Generation of Protocols," in *Proc. ACM SIGCOMM Symposium on Communication Architectures and Protocols*, 1990.

[13] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550, July 2003.

[14] M. Handley and C. Perkins, "Guidelines for Writers of RTP Payload Format Specifications," RFC 2736, 1999.

[15] B. J. Brachman and S. T. Chanson, "Fragmentation in Store-and-Forward Message Transfer," *IEEE Communications Magazine*, 1988.

[16] M. Pitkänen and J. Ott, "Redundancy and Distributed Caching in Mobile DTNs," in *Proc. ACM SIGCOMM MobiArch Workshop*, August 2007.

[17] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *Proceedings of ACM SIGCOMM*, 1998.

[18] A. Kamra, V. Misra, J. Feldman, and D. Rubenstein, "Growth codes: maximizing sensor network data persistence," *ACM SIGCOMM CCR*, vol. 36, no. 4, pp. 255–266, 2006.

[19] S. Farrell, S. Symington, H. Weiss, and P. Lovell, "Delay-Tolerant Networking Security Overview," Internet Draft draft-irtf-dtnrg-sec-overview-03.txt, Work in progress, July 2007.

[20] C. Partridge, "Authentication for fragments," in *Proc. ACM SIGCOMM HotNets-IV workshop*, 2005.

[21] S. Jain, K. Fall, and R. Patra, "Routing in a Delay Tolerant Network," in *Proc. ACM SIGCOMM*, 2004.

[22] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Technical Report CS-200006, Duke University, April 2000.

[23] M. Demmer and K. Fall, "DTLSR: Delay tolerant routing for developing regions," in *Proc. ACM SIGCOMM NSDR Workshop*, 2007.

[24] A. Keränen and J. Ott, "Increasing Reality for DTN Protocol Simulations," Tech. Rep., Helsinki University of Technology, Networking Laboratory, 2007.