

MAC Layer Support for Delay Tolerant Video Transport in Disruptive MANETs

Morten Lindeberg, Stein Kristiansen, Vera Goebel, Thomas Plagemann

▶ **To cite this version:**

Morten Lindeberg, Stein Kristiansen, Vera Goebel, Thomas Plagemann. MAC Layer Support for Delay Tolerant Video Transport in Disruptive MANETs. 10th IFIP Networking Conference (NETWORKING), May 2011, Valencia, Spain. pp.106-119, 10.1007/978-3-642-20757-0_9 . hal-01583420

HAL Id: hal-01583420

https://inria.hal.science/hal-01583420

Submitted on 7 Sep 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

MAC Layer Support for Delay Tolerant Video Transport in Disruptive MANETs

Morten Lindeberg, Stein Kristiansen, Vera Goebel, and Thomas Plagemann

University of Oslo, Department of Informatics
Postboks 1080 Blindern, 0316 Oslo, Norway
[mglindeb, steikr, goebel, plageman]@ifi.uio.no

Abstract. The overall goal of this work is to improve video delivery in emergency and rescue scenarios using sparse MANETs that might be prone to frequent link breaks and network partitions. The core idea of our approach is to reduce the number of MAC layer retransmissions that are likely to fail. We do not drop packets that could not be sent after the final retransmission. Instead we handle them in an overlay for store-carry-forwarding. The design of the overlay protocol takes the instability of the network into account, in such a way that each overlay entity works autonomously and keeps a minimum amount of state. Our experimental results show that we reduce packet loss seen on broken links, while at the same time significantly reducing overhead in terms of the total amount of packets transmitted at the physical layer.

Keywords: Video transmission; MANET; delay tolerant transport; cross-layer optimization

1 Introduction

Mobile ad-hoc networks (MANETs) can provide valuable communication services for emergency and rescue (ER) operations in the absence of a working communication infrastructure. One such service is the transfer of video data from rescue workers wearing head-mounted video cameras to a remote command and control center (CCC). The dynamicity of MANETs, the possibility of short and long-term network partitions, and the fact that forwarding nodes might be small hand held devices introduce many hard research challenges [10]. However, this particular video service is in contrast to classical video streaming more tolerant to delay. Video data that is delayed for seconds, minutes or even hours can still be very useful in the CCC to understand what happened. We envisage in the CCC a video client that visualizes the availability of video sequences on a time axis, and allows play-out of live streams and browsing and play-out of locally stored video. The visualization will be immediately updated at the arrival of (delayed) video data. Therefore, video data that passed the play-out time should not be dropped, but instead delivered as fast as possible. Any video sequence might be still of importance.

In order to improve the delivery efficiency, we reduce in this work the number of MAC layer retransmissions that are likely to fail and prevent packet dropping at the MAC layer. Instead of dropping, packets are passed after the maximum number of unsuccessful retransmissions from the MAC layer to an overlay for delay tolerant store-carry-forwarding, called Dts-Overlay. It utilizes information from the routing table and the MAC layer to identify the most promising conditions for forwarding packets.

Packet loss at the MAC layer mainly happens if the link quality, i.e., signal strength, has degraded due to mobility, but the link is still registered in the routing table. In such cases, packets are passed from the transport layer via IP to the MAC layer, which in turn drops the packets after reaching the retransmission limit (typically seven). This problem exists until the routing protocol recognizes that the link is broken, e.g., through a missing hello message in OLSR. The high frequency of link errors between mobile nodes has stimulated cross-layer approaches like CIFLER [1] to use fast link error detection for fast link error recovery. COLLIE [2] is another approach to immediately react to link failures through link layer rate adaptation. Our solution is different; before the overlay sends a packet it checks both the routing table and the queue length at the link layer (*Link Adapt*). Only if both indicate a working link to the next hop it passes the packet to the transport layer. Obviously, if a link is broken it does not make sense to try several retransmissions. Reducing the maximum number of retransmissions results in lower system load, but might lead also to higher packet loss rate. Therefore, the MAC layer returns in our solution after maximum retransmission trials the packet to the Dts-Overlay instead of dropping it (*MAC Return*). This is similar to the approach presented by Voorhaen et al. [3] where the packet is returned to the IP layer, assuming that an alternate route exists through which the packet can be sent instead. However, this assumption is not valid for scenarios with network partitions. In our ER application domain it might be the case that the CCC is only reachable from the location of the accident via message ferries [4]. Another cause of packet loss in MANETs is the separation of neighbourhood discovery in the routing protocol and address resolution in ARP [5]. We study in our system two solutions to avoid this packet loss. First, we avoid the need for ARP by assuming that the devices used during an ER operation are configured and all IP and MAC addresses are exchanged a priori (*Static ARP*). Second, when a new link appears in the routing table, the overlay provokes the address resolution of the connected node and sends only video packets to the node after its address is resolved (*ARP Adapt*).

All core design decisions are based on measurements performed in real-world experiments. Our ns-3 based evaluation shows that the combination of these mechanisms leads to minimal packet loss, reduced overhead, and neglectable increased delay. The remainder of this paper is structured as follows: Section 2 presents system design and implementation, Section 3 the evaluation, related work is discussed in Section 4, and Section 5 summarizes the main achievements and presents future work.

2 Design and Implementation

This section explains the design of our solution which is based on UDP, IP, OLSR [6], and cross-layer enabled IEEE 802.11b (see Figure 1). First, we analyze MAC layer issues and describe the details of our MAC support. Then, we describe Dts-Overlay, and how we address ARP related packet loss. Finally, we present our implementation.

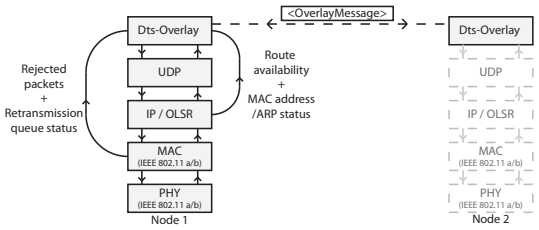


Fig. 1. Cross-layer Information Flow in Dts-Overlay.

2.1 MAC Layer Issues

IEEE 802.11 minimizes link layer packet loss through positive acknowledgements (ACK frames) and retransmissions if no ACK is received. In current implementations, the default value for the maximum limit of (re-)transmissions is seven, i.e., after seven unsuccessful transmissions the packet is dropped. There are two reasons for packet loss in IEEE 802.11, collisions and link quality. For the first one, more retransmissions will likely result in successful transmissions, while this is not true for links of inadequate quality. In MANETs, link quality might be degrading quickly due to mobility and links might not exist even if they are listed in the routing table, i.e., the routing table is outdated. Obviously, it does not make sense to initiate packet retransmission over a low quality link and especially not over a non-existing link. Reducing in these situations the maximum retransmission limit saves energy and bandwidth, and reduces the probability of collisions. The latter is especially important in highly loaded networks. Results from our experiments with a Nokia N900 as forwarding node in a small MANET [7] reveal a high amount of MAC layer retransmissions when reaching the saturation point in a real wireless ad hoc network setting. The high amount of retransmissions is not only consuming the senders energy and link bandwidth, but also energy and bandwidth of all nodes that are in communication distance to the sender. To investigate this further and identify a better limit for number of retransmissions, we study the overall effect of different retransmission limits in Section 3.

Four cases need to be considered when the MAC layer in the original 802.11 implementation would drop packets: (1) The link towards the destination is

down, but there exist another route. (2) The link towards the destination is down, and there exist no alternative route. (3) The receiving node is not able to forward traffic at the rate received from the link [7]. (4) Congestion and collisions lead to taildropping in the MAC retransmission queue at the sender side, or rather unlikely, the packet is lost due to collision in all retransmission trials.

For case (1), it is sufficient to hand the packet back to the network layer for re-routing, like in [3]. For case (2), the packet should be handed to the Dts-Overlay for being temporarily stored in the store-carry-forward buffer. For case (3), retransmissions only worsen node contention, thus we either re-route, or temporarily store packets in the forward buffer. Case (4) should only happen in overload situations, thus it is necessary to reduce the load, i.e., less retransmissions and temporary storage. To strive for simplicity and to be minimal invasive with the MAC layer we hand in all cases the packet to the Dts-Overlay. The cost of this is limited. We have measured the costs of passing a packet through an overlay instead of passing it directly to the network layer, on a Nokia 810 running in energy saving mode with low CPU and bus frequency. Results from our two-hop test-bed indicate that throughput is not worsened, and the additional delay is approximately 10 %. In higher CPU frequency settings, this share of additional delay is even reduced.

2.2 Dts-Overlay

The fundamental task of the Dts-Overlay, is to make forwarding decisions, and store transit packets when it is not meaningful to forward them. The Dts-Overlay is formed by autonomously working instances on each node, and packets are forwarded by Dts-Overlay hop-by-hop. Forwarding decisions are based on link status from the MAC layer, and route status from the routing protocol. It is the goal to transport all packets as close as possible to the destination, i.e., the CCC in ER. Dts-Overlay handles packets as follows: (1) If OLSR reports that no route to the destination exists, we check for recent routes in the network topology history, which is kept by Dts-Overlay. (2) If a recent route suggests a next-hop that is within the range of the current node, packets are sent to this next-hop. This is based on the heuristic that the next hop node on the recent route to the destination is probably closer to the receiver. (3) If there are no recent routes, we temporarily suspend transmission by maintaining the packet in a store-carry-forward buffer part of the Dts-Overlay. (4) If the next-hop is identified, we check to see if the MAC layer retransmission queue is filling (representing link status). In that case, we also suspend transmission.

To efficiently leverage message ferries, we currently assume that their IP address is in an a priori defined address range, i.e., message ferries can be identified when their IP address appears in the routing table. Message ferries are used as follows: when no route is found to the destination (CCC), and there exist route(s) to carrier nodes, the packets are forwarded to the closest carrier. To avoid packets looping back, we do not forward packets from carrier to carrier.

In a separate thread of execution, we monitor route updates in the routing table. As route notifications appear, we loop through all suspended packets and see if any of the suspended packets matches the new route(s). If so, they are forwarded to the suggested next-hop. Figure 2 shows the main components of the system, and the main flow of function calls and information exchange between them.

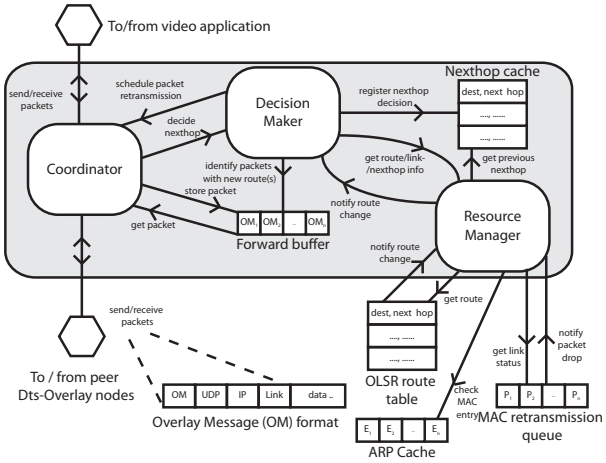


Fig. 2. Dts-Overlay System Design.

The **Coordinator** component implements the main interface (send / receive primitives) for the video streaming applications, and holds the UDP socket used for all peer communication. It is responsible for forwarding and receiving packets from other Dts-Overlay peers through the network, and packets scheduled for retransmission from the store-carry-forward buffer.

The main function of the **Decision Maker** is to provide the next-hop IP address for a given packet destination, and control the store-carry-forward buffer. Currently, it implements a simplified and adapted version of the TCP congestion control algorithm to control the send rate in which the feedback is not provided by the receiving node, but instead by the local link status. It is the responsibility of the **Resource Manager** to collect and monitor network state information. Figure 2 illustrates from where it obtains this information. The resource manager serves as information broker, and provides data to the Decision Maker. It also serves as the entry point for packets that reach the final retransmission limit at the MAC layer.

We have implemented the store-carry-forward packet buffer as a drop tail FIFO queue. The size of this queue represents a tradeoff between memory consumption, and delay tolerant packet forwarding. Currently, we have a relatively large queue size (500 MB) to avoid tail dropping. Recent developments in solid-state drives provide relatively cheap, large and fast storage to the mobile devices.

Thus, we expect that for low bit rate video streams within relative short network scenarios, it is realistic to expect that the buffer sizes do not limit the store-carry-forward capabilities of nodes. Packet exchange between peers is performed through **overlay messages**. Currently, the header is very simple; it only keeps the IP address of the destination. This information is lost at the IP layer, since overlay messages are only sent hop-by-hop.

2.3 Address Resolution

Carter et al. [3] identified ARP as one cause for packet loss in case several packets are sent immediately after an ARP request before the resolution process has finished, because only one packet could be buffered in ARP. Despite substantially increased buffer space in ARP in the recent ns-3 implementation, we identified a severe packet loss between the Dts-Overlay and the MAC layer. Packets are still silently dropped at the IP layer, when a MAC address for a given IP destination (reported by the routing protocol) has not been discovered. Further investigations revealed that this happens in cases the ARP reply is lost. To avoid these packet drops, we have implemented and evaluated two different approaches. First, we avoid the need for ARP by assuming that all devices in an ER operation are configured a priori with fixed IP addresses and all IP and MAC addresses are stored on all devices. In our second solution, we assure that the address resolution process is successfully finished before sending video packets, by sending an empty dummy packet to the IP destination. This leads to some overhead in terms of transmitted bytes, but our results show that the gain in packet reception at the destination is significant.

2.4 Implementation

We have adapted an already existing UDP based application for transmitting video traces (UdpTraceClient) which is part of the ns-3 code distribution. It generates UDP packets with packet sizes based on a given video trace, and adds a sequence number and timestamp to each packet. Dts-Overlay is implemented in C++ as an ns-3 application. This should make it easily portable to real devices.

We achieve cross-layer parameter exchange through the use of the ns-3 object aggregation system. In essence, ns-3 models network nodes and its protocol instances as objects. The smart pointer system enables access to these objects. This means that our Resource Manager component, realized as an object, can interact with, e.g., the OLSR routing protocol instance through function calls. We have implemented function calls within the ns-3 implementation of the IEEE 802.11 MAC layer, the ARP protocol, and the OLSR routing protocol that enables the Resource Manager to get the necessary state data.

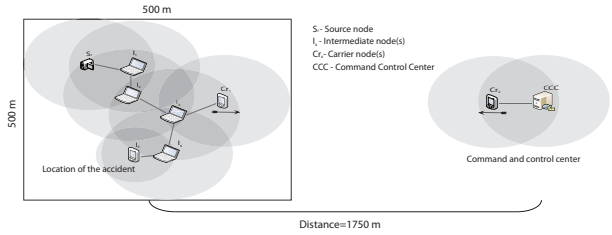


Fig. 3. ER Scenario.

3 Evaluation

In this section, we consider three network topologies, (1) ER scenario, (2) sparse MANETs, and (3) dense MANETs. The simulations are conducted in ns-3[1], version 3.9. For the **ER Scenario**, we have created a mobility model that reflects realistic ER scenarios. It contains two network partitions and a set of designated message ferries moving between them (see Figure 3). On-location nodes move with a speed of 2 m/s (approximately walking speed) 10 s pause time, according to the Random Walk mobility model, within a 500 m x 500 m area. Carrier nodes pause both at the CCC and on location for 60 seconds, allowing data exchange. During the carrier phase, they move following a straight line at 10 m/s, the distance is 1750 m. The CCC node is not moving at all.

In addition to the ER scenario, we consider (2) **Sparse MANET** with area size 1250 m x 1250 m, and (3) **Dense MANET** with area size 250 m x 250 m. Each with 20 nodes, Random Walk mobility, node speed of 2 m/s, and 10 s pause time. In all network scenarios, nodes use IEEE 802.11b in ad-hoc mode. The devices simulate direct-sequence spread spectrum (DSSS) modulation, and a constant data transmission mode of 11 Mbps. For modelling the wireless channel, we utilize the constant speed propagation delay model, and the Friis propagation loss model.

3.1 Workload and metrics

The workload consists of a single unicast stream of the video entitled Foreman, obtained from the Video Traces Research Group[2]. The resolution is CIF (352x288) with a 25 fps frame rate, and the video is pre-encoded using H.264 baseline profile with a target bitrate of 256 kb/s. We packetize the H.264 encoded video into a .mp4 container, with MTU size set to 1400 bytes. The 12-second video stream is repeated 300 times, i.e., we obtain a total duration of 1 hour.

For the dense scenario, our purpose is to identify how MAC support handles packet collisions. To achieve this, four UDP streams are sent between randomly

[1] Available at http://www.nsnam.org/
[2] http://trace.kom.aau.dk/

selected intermediate nodes. Each UDP stream comprises 7500 packets per second with a uniform packet size of 1024 bytes. The resulting bitrate is higher than the available bandwidth, thus packet collisions are provoked.

We use four metrics to evaluate our solution: the percentage of video packets that are correctly received at the destination, denoted packet reception Rx_v; number of packets that are stored in intermediate nodes, denoted buffered packets B_{uf}; the percentage of lost packets, denoted packet loss P_l; and the overhead in terms of the total number of bytes of video packets transmitted at the physical layer, denoted Tx_t. Since we support delay tolerant video transfer it does not make sense to use video quality measures based on signal to noise ratio. All numbers are averages from five experiment runs, unless otherwise stated, and we present the standard deviation (σ).

3.2 Results

In the MAC Support evaluation, we compare five configurations (**C3** - **C7**) to evaluate the effect of MAC Return (MR) and Link Adapt (LA). The configuration settings are shown in Table 1.

Table 1. MAC Support Evaluation Configurations.

	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**
Retransmission Limit	7	7	7	3	3	3	3
MAC Return	no	no	no	no	no	yes	yes
Link Adapt	no	no	no	no	yes	no	yes
ARP setting	default	static	adapt	adapt	adapt	adapt	adapt

Table 2. ARP Effect on Packet Loss

	C1	**C2**	**C3**
P_l	39.3 (σ:9.4) %	31.7 % (σ:4.1)	19.5 % (σ:5.1)

Effect of ARP: First, we compare the impact of ARP on packet loss in the default setting **C1** with our solution based on pre-configured IP and MAC addresses (**C2**), and our alternative solution to send packets after the resolution process has successfully finished **C3**. Results are averages from five experiment runs with 30-minute duration, ER scenario. We focus on packet loss P_l. Results indicate that packet loss is heavily affected by our second solution. The packet loss with default settings (**C1**) is 39.3 %, 31.7 % with static ARP (**C2**), and 19.5 % with ARP Adapt (**C3**). In **C1** and **C2**, packets are sent immediately also over links with poor quality. In **C3** packet forwarding is delayed until ARP

has successfully finished, which indicates a useful link quality. As a result, we deploy ARP Adapt in remaining experiments.

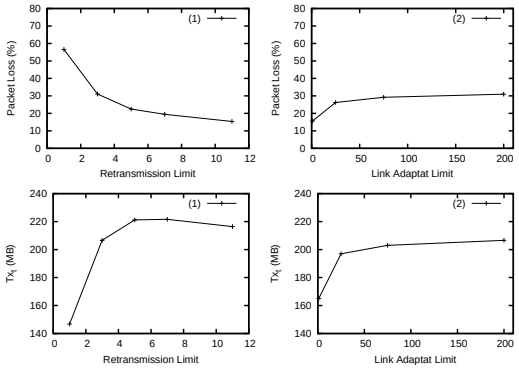


Fig. 4. Effect of Retransmission Limit and Link Adapt Limit Settings.

Retransmission Limit and Link Adapt Limit: To understand and quantify the tradeoffs of reliability and costs related to the maximum number of retransmissions, we measure packet loss and costs for (1) different retransmission limits, and (2) different MAC retransmission queue size limits that prevent the Dts-Overlay to send packets in the LA mechanism. For (1), we show that we reduce overhead by lowering the retransmission limit, at the cost of packet loss. For (2), we see that we actually reduce both packet loss and cost (see Figure 4). The experiments are performed with the default Dts-Overlay configuration in the ER scenario with duration of 30 min. We present the average of five experiment runs.

The highest packet loss occurs with a retransmission limit of one (56.5 %). With a limit of three, packet loss is lower (31 %). A limit of five results in 22.0 %, a limit of seven in 19.5 %, and a limit of eleven in 15.4 % packet loss. At the same time we see an increase in the cost Tx_t of 47 %, from retry limit of one up to eleven. Notice that the curve is almost flat after five retries ($Tx_t = 221$ MB). In our simulation studies, this behaviour can be attributed to tail dropping in the MAC retransmission queue. The current maximum queue size for this queue in ns-3 is 400 packets, which is very often reached in the experiments.

We have found that MAC retransmission queue size serves as a good indication of the link status. If packets destined on a certain link start piling up, the link is probably down. LA stops all ongoing transfers over the specific link, in case the queue starts filling up. Currently, this adaptation limit is set to 75 for our main experiments. The reason is that the limit is beneficial for MR configurations shown by preliminary experimentation. As a pre-study we investigate its effect on default configurations. Notice that we utilize a fixed retry limit of three.

The figure clearly indicates that we get the lowest packet loss with an adaptation limit set to one. At 25, packet reception is worse. Finally at 200, packet loss has almost doubled from the limit set to one, i.e., 31 % of the packets are lost. The gain in reduced packet loss for low limit settings is even combined with reduced cost Tx_t. Such a strict limit means we stop using links in which packets are scheduled for retransmission. For denser networks with more frequent random collisions, we expect such a setting to be too strict, i.e., packets will be kept from being sent over working links.

Table 3. Evaluation Results ER Scenario

	C3	**C4**	**C5**	**C6**	**C7**
Rx_v	77.5 % (σ:1.8)	65.7 % (σ:1.6)	66.3 % (σ:1.0)	93.5 % (σ:0.5)	93.6 % (σ:0.7)
B_{uf}	5.1 % (σ:0.7)	4.5 % (σ:0.6)	4.5 % (σ:0.3)	5.6 % (σ:0.4)	6.1 % (σ:0.6)
P_L	17.4 % (σ:2.1)	29.8 % (σ:1.2)	29.2 % (σ:1.2)	0.9 % (σ:0.3)	0.3 % (σ:0.0)
Tx_t	455 MB (σ:19)	412 MB (σ:7)	411 MB (σ:10)	475 MB (σ:15)	454 MB (σ:28)

ER Scenario: Due to the distance between the ER location and the CCC, all packet receptions come via carrier nodes. The numbers in the table show that our MR configurations (**C6** and **C7**) outperform **C3** when it comes to packet reception (16 % improvement), and loss is reduced from 17.4 % to <1 %. Remaining packets are kept in intermediate node buffers. LA (**C5** and **C7**) has little effect in terms of loss and packet reception. By reducing the retransmission limit for default configuration , packet loss is increased to 29.8 % (**C4**), however, we reduce overhead in terms of meaningless physical layer transmissions.

Most notable, the total transmitted bytes for **C7** is in average 1 MB lower than in **C3**. In comparison, **C6** achieves in average 21 MB more byte transmission than **C7**, thus LA improves our results. To summarize, MAC Support (MR plus LA) increases packet reception, reduces packet loss, and reduces cost in ER scenarios.

Table 4. Evaluation Results Sparse Manet Scenario

	C3	**C4**	**C5**	**C6**	**C7**
Rx_v	81.3 % (σ:24.6)	76.6 % (σ:30.7)	76.5 % (σ:30.8)	91.3 % (σ:14.8)	91.4 % (σ:15.0)
B_{uf}	0.0 % (σ:0.0)	0.0 % (σ:0.0)	0.1 % (σ:0.2)	7.3 % (σ:13.6)	8.1 % (σ:14.7)
P_l	18.7 % (σ:24.6)	23.4 % (σ:30.7)	23.4% (σ:30.7)	1.4 % (σ:1.9)	0.5 % (σ:0.6)
Tx_t	397 MB (σ:362)	245 MB (σ:162)	247 MB (σ:165)	381 MB (σ:356)	379 MB (σ:354)

Sparse MANET Scenario: In total, we see that packet reception has decreased for the MR configurations, compared to the ER scenario. We achieve

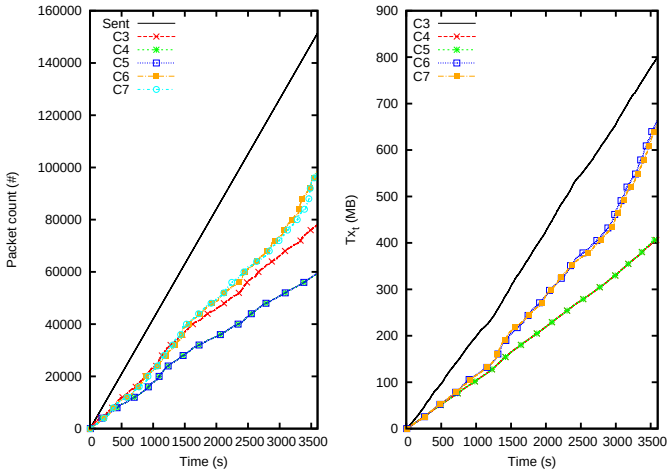


Fig. 5. Sparse MANET single run, Node speed 2m/s.

with 91 % the highest packet reception in both cases (**C6** and **C7**). Packet reception is better ($\approx$ 10 %) than for **C3**, where packet loss is 18.7 %. Remaining packets are kept in the buffers, although we experience 1 % packet loss for **C6**. The **C4** and **C5** configuration achieve both 77 % packet reception, thus LA does not improve packet reception. Packet loss is 23.4 % in these two cases.

Standard deviation is quite high, also for the total transmission of bytes. The reason for this is that the packet loss is in some runs high and in others rather low, depending on the seed for the generation of random mobility. In the following we only use one seed that creates a scenario with high packet loss.

Figure 5 shows Rx_v and Tx_t for one particular experiment run over time. For **C3**, we receive 52 % of the packets. By reducing the retransmission limit to three (**C4**), packet reception is lower, e.g., 39.2 %, and 39.3 % with LA enabled. Packet reception is 64.8 % for **C6**, and 65.0 % for **C7**. This demonstrates that MR is useful in scenarios with high packet loss.

Turning our focus to overhead, we see that overall packet transmission is reduced quite heavily by lowering the retransmission limit from seven to three. For **C3** and **C4**, a reduction of $Tx_t \approx 400$ MB, i.e., a factor of 0.5. Configuration **C7** leads to approximately 63 % more transmitted PHY packets than in **C3**, and close to 65 % more packet reception. It should be noted that those packet that did not reach the destination are kept in forward buffers of intermediate nodes, i.e., no packet loss occurred in **C7**.

Dense MANET Scenario: Nodes are randomly distributed in the relatively small area compared to signal propagation length, i.e., node density is high. As a result, packet loss is mostly caused by collisions, rather than insufficient signal strength. Configuration **C3** handles collisions well, i.e., no packet loss. For

Table 5. Evaluation Results Dense Manet Scenario

	C3	**C4**	**C5**	**C6**	**C7**
Rx_v	100.0 % (σ:0.4)	99.0 % (σ:0.6)	99.3 % (σ:0.1)	99.8 % (σ:0.0)	99.8 %(σ:0.1)
B_{uf}	0.0 % (σ:0.0)	0.2 % (σ:0.6)	0.0 % (σ:0.1)	3.5 % (σ:0.1)	3.5 % (σ:0.2)
P_l	0.0 % (σ:0.0)	0.8 % (σ:0.1)	0.6 % (σ:0.1)	-3.4 % (σ:0.1)	-3.4 % (σ:0.1)
Tx_t	149 MB (σ:2)	149 MB (σ:3)	98 MB (σ:6)	151 MB (σ:3)	92 MB (σ:8)

configurations **C4** and **C5**, packet collisions cause ≈ 1 % packet loss resulting in ≈ 99 % packet reception. For the MR configurations (**C6** and **C7**), we encompass close to 100 % packet reception and some amount of duplicates, indicated by a negative packet loss of $\approx -3.5\%$. The cause of these duplicates is that our MAC support interferes with the MAC layer functionality to handle frame duplicates due to missing MAC acknowledgements.

We see that **C7** achieves the lowest overhead, from **C3**, a 62 % reduction, which is substantial. In general, a severe reduction applies to all LA configurations, proven highly beneficial in congested and dense MANETs.

4 Related Work

The related work falls into three categories: (1) delay tolerant transport, (2) video delivery over MANETs, and (3) adaptations to MAC or PHY conditions. The typical approach of delay tolerant networking is to provide an alternative to TCP, to achieve some reliability for end-to-end transport. One alternative is to introduce a layer that adds delay tolerance above existing routing protocols, like in SAFT [8]. Reliability is provided through a double control-loop, including end-to-end feedback and TCP hop-by-hop. Applying TCP between each hop substantially increases the number of control packets. Delay-tolerant video transport over IP is achieved in MOMENTUM [9], like in our approach through an overlay. However, only selected nodes, called session nodes, actively transport the video data and require knowledge about all session nodes. Furthermore, adaptations to lower layers are not considered, e.g., to determine the buffer empty rate. Another alternative is to implement a delay tolerant routing protocol, resulting in what is often referred to as space-time routing. This is explored already in [10], the authors suggest that algorithms should take congestion into account, and that performance can still be good without global knowledge. Evaluation results from our autonomous solution support this.

In [11], a detailed survey of research on video streaming over MANETs is given. A popular approach is to combine cross-layered multipath MANET routing with multiple description video coding, often leading to NP-hard optimization problems. It should be noted that all these works assume that one or more routes between source and destination exist.

The effect of MAC layer retransmissions on video quality is studied in [12, 13]. Both efforts aim to adapt the retransmission limit, to better meet user video quality demands. COLLIE [2] and CIFLER [1] enable the routing protocol to

better react to link status. The reduction of packet loss, caused by lack of exact link status for the OLSR routing protocol is targeted in [3]. In case a packet could not be transferred over a link, it is returned to the IP layer assuming another route exists. Thus, tolerance to path disruptions is neither considered nor supported. Authors show, as we do with our related technique, improvements in terms of reduced packet loss.

5 Conclusion

This work reduces packet loss in disruptive MANETs by MAC layer support, and paired with the Dts-Overlay, achieves tolerance to network disruptions. As part of MAC support, our MAC Return algorithm enables us to drastically reduce packet loss, in turn allowing us to reduce overhead by lowering the MAC layer retransmission limit. In addition, our Link Adapt algorithm allows us to reduce overhead in that we do not try to send packets over links that are down. This has been supported by extensive simulation studies. More precisely, we have shown that our approach achieves both a reduction in physical layer packet transmissions, while avoiding packet loss through the use of intermediate node buffering, paired with a higher packet reception rate in both a sparse MANET setting, and in our special purpose ER scenario. In addition, the usability of our approach is strengthened through evaluation results in a dense MANET setting. With high density, we see improvements of packet reception/packet loss, although at the cost of some packet duplicates.

There are still open issues and remaining challenges. Currently, MAC Return does not support packet fragmentation at the MAC or IP layer. As seen in the dense MANET setting, it can obstruct existing MAC functionality for identifying duplicates. When acknowledgements from the receiver node are lost, we encounter a false negative error. This might lead to the sender retransmitting packets that are already received at the receiver. There are yet parameters to fine-tune. For one, we can improve our rate control algorithm for emptying the store-carry-forward buffer by utilizing physical layer parameters, such as signal strength. We should account for video coding parameters and frame priority in our forwarding decisions. At the network and routing layer, improvements can be done to, e.g., OLSR, by removing links from OLSR data structures known to be down. At the MAC layer, we could benefit from deploying one of the existing rate adaptation algorithms, such as CARA [14]. For future work, we plan to incorporate an energy model in the simulated nodes to enable energy aware forwarding strategies. We aim to implement our work on a real mobile device, such as the Nokia N900. This would strengthen the realism of our performance evaluations. As stated in [7], we are aware that mobile hand helds have lower forwarding capability than what the wireless medium supports. Especially, we hope to investigate how to tackle packet loss due to node contention, as opposed to weak signal strength or packet collisions.

Acknowledgements This work is funded by the VERDIKT program of the Norwegian Research Council, through the DT-Stream project (no. 183312/S10). The authors would like to thank Sergio Cabrero, Isaías Martínez Yelmo and Daniel Rodríguez-Fernández for valuable insights.

References

1. Yackoski, J., Shen, C.C.: Cross-layer inference-based fast link error recovery for manets. In: Wireless Communications and Networking Conference (WCNC). Volume 2., IEEE (2006) 715–722
2. Rayanchu, S., Mishra, A., Agrawal, D., Saha, S., Banerjee, S.: Diagnosing wireless packet losses in 802.11: Separating collision from weak signal. In: International Conference on Computer Communications (INFOCOM), IEEE (2008) 735–743
3. Voorhaen, M., Blondia, C.: Analyzing the impact of neighbor sensing on the performance of the olsr protocol. In: International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, IEEE (2006) 1–6
4. Zhao, W., Ammar, M., Zegura, E.: A message ferrying approach for data delivery in sparse mobile ad hoc networks. In: International symposium on Mobile ad hoc networking and computing (MobiHoc), ACM (2004) 187–198
5. Carter, C., Yi, S., Kravets, R.: ARP considered harmful: manycast transactions in ad hoc networks. In: Wireless Communications and Networking (WCNC). Volume 3., IEEE (2003) 1801–1806
6. Jacquet, P., Muhlethaler, P., Clausen, T., Laouiti, A., Qayyum, A., Viennot, L.: Optimized link state routing protocol for ad hoc networks. In: lnternational Multi Topic Conference (INMIC), IEEE (2001) 62–68
7. Kristiansen, S., Lindeberg, M., Rodriguez-Fernandez, D., Plagemann, T.: On the forwarding capability of mobile handhelds for video streaming over manets. In: SIGCOMM workshop on Networking, systems, and applications on mobile handhelds (MobiHeld), ACM (2010) 33–38
8. Heimlicher, S., Baumann, R., May, M., Plattner, B.: SaFT: Reliable transport in mobile networks. In: International Conference on Mobile Adhoc and Sensor Systems (MASS), IEEE (2006) 477–480
9. Cabrero, S., Pañeda, X.G., Plagemann, T., Goebel, V.: Overlay solution for multimedia data over sparse MANETs. In: International Wireless Communications and Mobile Computing Conference (IWCMC). (2009)
10. Jain, S., Fall, K., Patra, R.: Routing in a delay tolerant network. SIGCOMM Comput. Commun. Rev. **34** (2004) 145–158
11. Lindeberg, M., Kristiansen, S., Plagemann, T., Goebel, V.: Challenges and techniques for video streaming over mobile ad hoc networks. Multimedia Systems **17** (2011) 51–82
12. Chan, A., Lee, S.J., Cheng, X., Banerjee, S., Mohapatra, P.: The impact of link-layer retransmissions on video streaming in wireless mesh networks. In: International Conference on Wireless Internet (WICON), ACM (2008) 1–5
13. Choudhury, S., Sheriff, I., Gibson, J.D., Belding-Royer, E.M.: Effect of payload length variation and retransmissions on multimedia in 802.11a WLANs. In: International conference on Wireless communications and mobile computing (IWCMC), ACM (2006) 377–382
14. Kim, J., Kim, S., Choi, S., Qiao, D.: CARA: Collision-Aware Rate Adaptation for IEEE 802.11 WLANs. In: International Conference on Computer Communications (INFOCOM), IEEE (2006) 1–11