Received 19 February 2022; revised 23 March 2022; accepted 14 April 2022. Date of publication 26 April 2022; date of current version 10 May 2022.

Digital Object Identifier 10.1109/OJCOMS.2022.3169444

Joint Link Rate Selection and Adaptive Forward Error Correction for High-Rate Wireless Multicast

THOMAS GEITHNER [1], FIKRET SIVRIKAYA [2], AND SAHIN ALBAYRAK [1]

[1] DAI-Labor, TU Berlin, 10587 Berlin, Germany

[2] GT-ARC gemeinnützige GmbH, 10587 Berlin, Germany

CORRESPONDING AUTHOR: T. GEITHNER (e-mail: thomas.geithner@dai-labor.de)

This work was supported by the German Research Foundation and the Open Access Publication Fund of TU Berlin.

ABSTRACT Multicast communication over wireless networks has many potential applications, such as real-time audiovisual content distribution or digital signage systems with multiple remote terminals. However, today's common 802.11 networks cannot fully support such applications at the link layer due to the technical challenges in achieving both high transmission rate and high reliability for multicast data transfers. Those challenges primarily emerge from the inapplicability of immediate acknowledgment mechanisms of unicast transmissions to the multicast case and the need for finding a common transmission rate for all receivers with diverse channel conditions. The research literature in this domain mainly addresses the problem at the higher layers of the protocol stack. In this article, we present a novel approach that combines wireless link rate selection with an adaptive packet-level Forward Error Correction (FEC) mechanism in order to achieve high-rate and highly reliable multicast in 802.11 wireless networks. The integration of FEC into the 802.11 MAC layer allows direct interaction between the transmission rate and the coding scheme. As a result, potential packet losses caused by higher link rates can be compensated by the adjustable redundancy provided by FEC. In combination with an aggregated receiver feedback mechanism, this yields improved transmission efficiency and reliability for wireless multicast. We investigate this approach in a simulation environment under a realistic wireless channel model in various application scenarios with up to 50 receivers. The results represent significant performance improvements, in terms of throughput, channel utilization, and packet loss, over the state-of-the-art methods for reliable wireless multicast.

INDEX TERMS 802.11, forward error correction, network coding, rate sampling, reliable multicast, transmission rate control, wireless networks.

I. INTRODUCTION

RELIABLE multicast communication over wireless networks finds increasingly more use cases, such as those in smart home, smart city, and connected mobility application scenarios. Many of these use cases also require high throughput, which is particularly challenging to achieve in combination with high reliability for wireless multicast. A traditional multicast use-case is real-time video or audio content distribution to multiple clients, which would benefit from an efficient data transfer method over wireless networks for reaching mobile devices. This also applies to digital signage systems and similar applications with many remote displays. Less common but also applicable is the reliable non-realtime distribution of content to many wireless clients. Example use-cases of this kind of traffic scenario include offline content upload to digital signage terminals and simultaneous uploads of software packages, firmware updates, or container images in industrial and Internet of Things (IoT) applications. Such applications, among many others, would benefit from efficient multicast transfers for an economical utilization of the limited capacity of wireless channels.

The IEEE 802.11 standard series builds the basis for today's Wireless Local Area Network (WLAN) deployments.

From its earliest days on, it has been optimized for unicast data as the vast majority of applications uses that kind of traffic, while broadcast and multicast data played only a supplemental role. In the basic standard version, the 802.11 technology provides only rudimentary support for mutlicast data, where the efficiency of multicast transmissions is well below that of the unicast transmissions even in the newer standard versions. This lack of efficient multicast support in 802.11 networks provides much potential for improvements.

The transmission rate control as part of the 802.11 Medium Access Control (MAC) stack is responsible for selecting appropriate transmission parameters for data frames, such as Modulation and Coding Scheme (MCS), channel width, spatial streams, and guard interval (depending on the 802.11 version). This is therefore essential for the efficiency and coverage range of transmissions. These parameters are often denoted together as the *transmission rate*. But since this term is somewhat overloaded (e.g., also used to indicate the rate of data traffic sent by an application), we adopt the term *link layer transmission rate*, or *link rate* for short, for avoiding confusion. In the following, we use the term for covering all individual parameters regardless of the specific 802.11 version, unless explicitly stated otherwise. Link layer transmission rate controls for unicast data flows are a well investigated topic and lead to well working solutions used in common 802.11 networks. Since the unicast data flows are directed to one receiver only, the link rate can be selected according to the individual channel conditions for that receiver. Basic approach of most practically relevant unicast rate control algorithms, such as the *Minstrel* algorithm [1], is to use the acknowledgment (ACK) frame for obtaining the required channel state information. Additionally, the ACK mechanism allows retransmissions of lost packet for achieving a certain level of transmission reliability.

When data packets are directed to multiple receivers, as is the case with multicast and broadcast transmissions, the usual ACK mechanism is not applicable since that would cause collision among the ACK frames. Therefore, the traditional 802.11 standard waives the ACK mechanism for group directed transmissions which prevents the use of common link rate control approaches on multicast data and also affects reliability due to the lacking possibility of retransmissions. Besides those limitations due to the unavailability of the frame-level feedback, a multicast rate control faces an additional challenge since the transmission rate being selected must fit for all receivers in that multicast group. Due to these reasons, multicast rate controls can not simply apply the concepts of common unicast approaches. The traditional way of dealing with this difficulty, hereafter denoted as *legacy multicast*, is to usually transmit non-unicast packets at the lowest link rate with the intention of maximizing the transmission reliability. This method is reasonably sufficient as long as multicast is used only for auxiliary network management tasks such as network discovery or route advertisements, but it prevents using multicast over wireless links for high-bandwidth tasks or for reliable transfer of application data.

In this article, we present *Dynamic and Adaptive Link Rate Control for Wireless Reliable Multicast (DEWRiM)* as a novel approach for achieving highly reliable wireless multicast at the link layer for high-rate data traffic, by combining an adaptive, packet-level FEC mechanism with dynamic link rate selection. This FEC should not be confused with the link layer coding, which is an integral part of the 802.11 standard for recovering bit errors as part of the MCS selection. In contrast, the FEC layer in our approach works supplementary on the packet level for recovering lost frames. This builds on our preceding work, which presented the basic mechanisms of a multicast link rate control protocol based on aggregated receiver feedback [2]. The current work provides significant extensions and improvements on several aspects. While the previous work employed a static FEC layer with a fixed code configuration, in this paper we integrate dynamic FEC code selection into the optimization problem jointly with link rate selection. This removes the dependency on empirically chosen FEC codecs for each application scenario. Furthermore, a retransmission mechanism based on FEC is introduced, which help further improve the packet loss handling capability. In the previous paper, User Datagram Protocol (UDP) was solely employed as the transport layer protocol in order to keep the evaluation focus on the link layer performance. Here we also investigate the overall performance by integrating NACK-Oriented Reliable Multicast (NORM) [3] as a transport layer protocol designed for multicast. Finally, we utilize a more realistic wireless channel model in the evaluations, representing an additional challenge for the link adaptation and reliability mechanisms.

To the best of our knowledge, there is no comparable work in the literature that jointly employs an adaptive FEC mechanism with dynamic link rate selection, with the single exception of *InFRA* [4]. However, unlike our work, *InFRA* aims at a special application scenario and follows different concepts for selecting a transmission rate and FEC codec.

DEWRiM aims at general high-rate multicast traffic scenarios such as real-time audiovisual content streaming applications as well as reliable bulk data distribution or IoT message exchange, e.g., as in [5]. On low-rate multicast traffic, such as the IPv6 Neighbor Discovery Protocol (NDP), the additional management overhead would potentially exceed the achievable gain. In comparison to special-purpose multicast rate controls for video traffic, DEWRiM is agnostic to higher network layers and does not make any assumptions on specific application requirements.

The rest of this article is organized as follows. We first provide an overview of the related literature on reliable multicast, wireless link rate control, and adaptive FEC mechanisms in Section II). Then the overall framework for the proposed approach is presented in Section III. Section IV embodies the key mechanisms of jointly adapting the wireless link rate and FEC coding scheme, followed by the comparative evaluations in Section V. Section VI concludes the article with the main takeaways and future work.

II. RELATED WORK

With the evolution of the IEEE 802.11 standards, the problems on multicast traffic have been partly addressed by optional handling methods. The Directed Multicast Service (DMS) mechanism, introduced in the *v*-amendment and part of the 2012 standard version [6], provides a workaround for cases of few multicast receivers within a group by translating the higher layer multicast flow to individual unicast transmissions per receiver at the MAC layer. With the *aa* version [7], the Group Cast with Retries (GCR) mechanism was introduced aiming at improving the reliability of MAC layer multicast transmission. This includes two variants, Group Cast with Retries Unsolicited Retry (GCR–UR) for proactive multiple transmissions as a very rudimentary form of FEC and Group Cast with Retries Block Acknowledgment (GCR–BA) with individual retransmissions by using a multicast version of the block acknowledgement mechanism that was introduced earlier for unicast in the *e*-amendment.

Beyond the 802.11 standard series, the research literature provides various other approaches for improving the handling of multicast data. In the following subsections, we present related approaches on three sub-problems: MAC layer reliability, adaptive FEC techniques, and multicast link rate selection.

A. MAC LAYER MULTICAST RELIABILITY

Reliability of multicast transmissions without limiting scalability is a general challenge, not only in the 802.11 MAC, but also for higher protocol layers. The common approach used in unicast transmissions, a simple Automatic Repeat Request (ARQ) mechanism with immediate ACK frames, is not applicable to multicast transmissions since the simultaneous ACK transmissions would result in excessive collisions. One method for reducing the amount of feedback messages is the usage of negative acknowledgments (NACKs) instead of ACKs, which alone does not solve the feedback collision problem. The NACK mechanism is often combined with leader based concepts, where only one or few receivers transmit feedback information back to the sender. Those two concepts are used together, for example, in the Leader-Based Protocol (LBP) [8], but also in NORM [3] at the transport layer.

Aggregating multiple consecutive feedback information into a single message is another way of decreasing the traffic amount. The block acknowledgment mechanism in GCR–BA is such an example, which combines the feedback aggregation with sequential polling. Some approaches aim at enabling simultaneous feedback mechanisms on the 802.11 technology. Gupta *et al.* [9] propose using out-of-band *tones* outside the 802.11 band with dedicated hardware as collision resistant NACK feedback. Instead of using additional hardware, *HIMAC* [10] uses *unary signals* on dedicated Orthogonal Frequency-Division Multiplexing (OFDM) subcarriers that have been pre-assigned to individual receivers. Those unary signals are used as NACK messages and for a receiver-side rate selection mechanism. In contrast, *RAMCAST* [11] and *FlexVi* [12] use such unary signals as a collision resistant NACK mechanism, where the dedicated OFDM subcarriers are used for a bitmap representing individual packets of a block.

B. ADAPTIVE PACKET-LEVEL FEC

Instead of using ARQ methods, packet-level FEC is an approach trying to compensate the packet loss by proactively transmitting sufficient redundancy information for recovering the losses at the receiver without the need for individual feedback information. This feature makes FEC well suited for broadcast and multicast communications on different network layers. Traditional block codes are limited in their error recovery capabilities by the boundaries of a single block. Newer approaches from the field of Random Linear Network Coding (RLNC) improve the flexibility by introducing sliding windows covering multiple blocks (also denoted as *generations*) that are shifted as the transmission proceeds. This interleaving approach offers a fast recovery of small losses but also powerful correction capabilities on consecutive failures. Variations of such sliding window mechanisms include, *ASWRNC* [13], where the sliding window is shifted based on receiver feedback information, and *Caterpillar RLNC* [14], which uses a window of fixed size that is shifted automatically.

Using a feedback mechanism provides the ability of adjusting the FEC parameters during the ongoing transmission. *InFRA* [4], for example, uses an FEC layer within the 802.11 MAC with dynamic codec adaption based on explicit receiver feedback information. The feedback-driven sliding window of *ASWRNC* [13] can be considered as an adaptive FEC mechanism, since the sliding window size depends on the reported packet reception. Similarly, Karetsi and Papapetrou propose in [15] an improvement to the sliding window mechanism by adding an ACK message that allows the encoder to reduce the encoding window size so that it covers only unacknowledged packets for reducing the matrix size at the decoder and thus the computing effort. Cohen *et al.* propose in [16] an adaptive coding mechanism that combines the adjustment of the FEC coding scheme as a proactive measure using the reported packet loss with on-demand retransmissions as reaction to unrecoverable failures. In [17], Yamamoto and Yomo use an estimation of the expected packet loss on an 802.11 network for dynamically adjusting the amount of redundancy symbols on an intermediate FEC layer. Lie and Mathur describe in [18] an adaption mechanism that calculates the required code rate based on the reported packet loss for fulfilling a given target Packet Delivery Rate (PDR).

Almost all these adaptive FEC mechanisms are either generic or applied to higher network layers than the 802.11

MAC. In contrast to our work, none of them uses any specific mechanisms of the 802.11 stack, with *InFRA* as the only exception.

C. MUTLICAST TRANSMISSION RATE CONTROL

As described earlier, the transmission rate control for multicast data is a major challenge in 802.11 networks due to the lack of the ACK mechanism. Generally, the literature contains two different directions of solving the rate selection problem. The first uses the Signal-to-Noise Ratio (SNR) or Signal-to-Interference-plus-Noise Ratio (SINR) as criterion for the rate selection. While those approaches work mostly well in simulated environments, they did not find much practical application due to the complexities in consistent measurement and interpretation of SNR values [19], [20]. More advanced approaches try to overcome these weaknesses with more detailed measurements, e.g., using the Channel State Information (CSI) with particular information about subcarrieres and spatial streams [20]. Examples for such multicast transmission rate controls are *REMP* [21] and *ARSM* [22] in combination with leader based concepts, as well as *HiMAC* [10], *FlexVi* [12], *RAMCAST* [11], and *OMACK* [23] with collision resistant feedback methods.

The second type of rate control mechanisms uses the packet transmission success as a decision base for rate selection, e.g., [24]–[27]. Most approaches rely on acknowledgment frames for calculating the transmission success, i.e., PDR. Some approaches, such as the Minstrel algorithm [1], make use of the retransmission mechanism by actively testing other rates than the currently selected one and risking additional losses for gaining information about other candidate rates. This behavior, denoted as *sampling*, is a common mechanism for unicast rate controls, but not directly applicable to multicast due to its dependency on retransmissions. A comparison of different rate selection algorithms based on the GCR–BA feedback mechanism is presented in [24]. A linear increase/multiplicative decrease rate adaption strategy based on the GCR–BA feedback information has been proposed by by Mansour *et al.* [25].

In contrast to the previous approaches, special purpose rate control methods aim at certain use cases with specific requirements. An example of such a use case is real-time video streaming to large groups of receivers. Based on the assumptions that such a video stream can tolerate a certain level of packet loss and that serving most receivers rather than all is sufficient, the constraints for the rate selection can be relaxed. This approach is utilized in *MuDRA* [28] in combination with a leader based concept as well as in *InFRA* [4] by using the Received Signal Strength Indicator (RSSI) as decision metric. *InFRA* uses and internal FEC mechanism in addition and adapts the coding scheme as well, based on the estimated reason of occurred packet losses. Yamamoto and Yomo describe in [17] a concept for improving the upstream video quality from a mobile device over multiple access points (APs) using multicast by evaluating the received beacon frames of the APs for selecting the link rate for the upstream in conjunction with an adaptive FEC mechanism.

To conclude the state of the art, it can be noted that the problem of multicast link rate control for 802.11 networks is still an unsolved issue. Only few approaches in the literature aim at reliability and rate selection for general multicast use cases. So, our approach of integrating an adaptive, packet-level FEC mechanism into the 802.11 MAC layer presents a new concept.

III. GENERAL FRAMEWORK AND DATA PIPELINE

With reference to its origin (Enhanced Rate Control for Wireless Reliable Multicast (EWRiM) [2]) and highlighting its more dynamic and adaptive behavior, we refer to the protocol presented in this paper as *DEWRiM*, which also helps distinguish it from the original version in the rest of the text.

DEWRiM is designed as a modular framework with various processing stages for dynamic selection of the optimal link rate and FEC code rate combination for a multicast receiver group. In what follows, we first provide a holistic view of the framework and then describe the individual pipeline stages on both transmitter and receiver sides. Note that this data pipeline is applied individually and independently to each application flow, i.e., for each multicast receiver group.

A. GENERAL DATA FLOW DESCRIPTION

The concept of DEWRiM is based on combining an adaptive FEC mechanism with link rate selection using an aggregated receiver feedback collection. This requires some additional processing steps for multicast data packets. In the legacy case, a multicast packet would be directly inserted into the TX queue of the transmitter with the default multicast rate being assigned. In our approach, multicast packets go through the following pipeline stages on the transmitter side (Fig. 1) and the receiver side (Fig. 2).

1) PRE-FEC DROP

In case of channel congestion, the queue of packets to be transmitted would grow. On reaching a certain level, newly incoming packets are dropped at this step. This mechanism is essential for some higher layer protocols that depend on packet losses for their internal congestion control mechanisms. Because the following processing steps could cover some congestion-caused packet loss and induce a negative impact on the decision logic of higher layer protocols, the first processing step is a packet drop mechanism based on the TX queue length. Since the additional processing steps, especially the FEC encoding *(b)* and the statistic request messages *(f)*, generate additional packets that are also inserted into the TX queue, the pre-FEC drop mechanism needs to reserve some capacity of the TX queue.

Besides covering the packet loss from higher layers, dropping packets at the TX queue later in the processing pipeline would also distort the packet statistics in *(f)*, because the

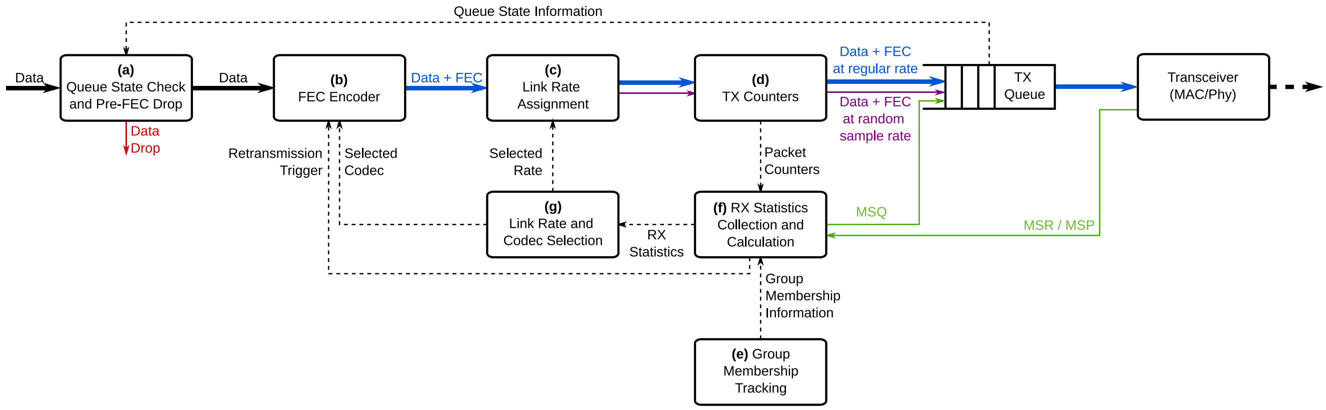


FIGURE 1. Transmission process in DEWRiM.

dropped packets would be counted as transmitted in *(d)*. This would be interpreted as a packet loss on the channel which could lead to the selection of a FEC coding scheme with a higher degree of redundancy (as will be described in detail in Section IV-C), and thus worsen the situation.

2) FEC ENCODER

Second step in the processing pipeline is the adaptive FEC encoding, which will be described in detail in Section IV. The *adaptivity* is realized by selecting a suitable code rate dynamically together with the link rate (detailed description in Section IV-C). In addition to the regular redundancy packets of the selected coding scheme, additional ones can be generated from encoding buffer in case of packet loss reports as a kind of retransmission mechanism.

As a core part of the concept, the FEC layer is used in DEWRiM for handling packet losses on the wireless channel, especially since an additional loss is intentionally accepted by the sampling process. The FEC encoder is able to switch between different coding schemes from a given set $c \in C$, whereby the decision about the currently selected coding scheme c^* is performed in step *(g)*.

3) LINK RATE ASSIGNMENT

The third step is the link rate assignment, where the currently selected rate t^* out for the set of available rates T is assigned to most of the packets belonging to the current flow. But as part of the key aspect of rate-sampling in DEWRiM, a certain portion of the data packets is sent with other link rates for measuring their PDRs and generating valid statistical data. Those portion of the traffic is denoted as “Data + FEC at random sample rate” in Fig. 1. As described more in detailed in [2], those other link rates are selected in our current implementation randomly out of the commonly supported ones.

This approach is comparable to common unicast transmission rate controls as for example the Minstrel algorithm [1]. Since a direct transmission success feedback is not easily applicable on multicast without causing packet collisions on the wireless channel, DEWRiM depends on the following step.

4) TX COUNTERS

The calculation of individual PDRs per link rate in step processing *(f)* relies essentially on information about transmitted packets per rate. This is done by continuously counting the number of packets at each rate. Obtaining the number of transmitted packets during a specific time interval is simply done by subtracting the counter values of end and start times of the interval.

5) GROUP MEMBERSHIP TRACKING

An important auxiliary function is the group membership tracking. Since other functions depend on precise information of multicast group subscriptions of nodes at the wireless channel, that information is provided by this module. The internal mechanisms for gathering this knowledge is not essential for our purposes and could be replaced by any other mechanism providing the same information. In our implementation, we are using simple periodic group membership announcements sent by multicast receivers based on their internal group subscriptions.

6) RX STATISTICS COLLECTION AND CALCULATION

A core component of DEWRiM is the reception statistic calculation for each receiver node at each link rate assigned in *(c)*. This process uses the transmission counter from step *(d)* and the corresponding reception counters collected periodically from all receivers. The reception counters are collected individually from each receiver by sending a *Multicast Statistics Request (MSQ)* message, which is responded with a *Multicast Statistics Report (MSR)* packet.

In addition to the periodic polling by the sender, the receiver can initiate a *Multicast Statistics Proactive Report (MSP)* message when it detects a high loss, which will also be applied to the statistic calculation. More details about the information collection and statistic calculation is given in [2].

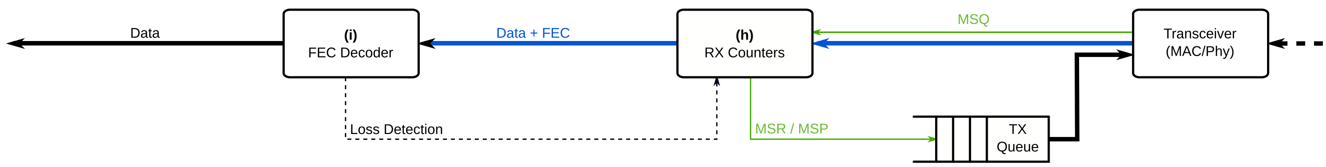


FIGURE 2. Reception process in DEWRiM.

7) LINK RATE AND FEC CODEC SELECTION

Core part of DEWRiM is the joint selection of link rate and FEC codec by minimizing the channel utilization while fulfilling a given PDR limit. The selection process is based on the receiver statistics calculated in *(f)* and uses the worst case among all receivers for each rate. The result of this process is the pair (t^*, c^*), indicating the selected link layer transmission rate $t^* \in T$ and the corresponding FEC codec $c^* \in C$ to be used in step *(b)*. Details of this processing step are explained in Section IV-C.

8) RX COUNTER COLLECTION

In comparison with the processing steps at the sender, the receiver side handling is significantly simpler, as shown in Fig. 2. After receiving the encoded data packets in at the transceiver, they are forwarded to the RX counter process, where the packets are counted per rate. On the reception of a Multicast Statistics Request (MSQ) message, the corresponding counters are sent as Multicast Statistics Report (MSR) message back to the sender node.

9) FEC DECODER

After counting the data packets, they are forwarded to the FEC decoder, which restores the original packet stream. Since DEWRiM uses a sliding window RLNC mechanism for the FEC layer (as will be described more in detail in Section IV-A), the receiver works opportunistically and does not need to know the codec currently selected by the transmitter.

On detecting a high packet loss based on the decoder queue length (due to undecodable packets, for example at 50% of the decoder queue length), a loss detection trigger is sent to the RX counter process *(h)*, which then creates a Multicast Statistics Proactive Report (MSP) message to provoke a fast reaction by the sender outside the regular MSQ/MSR interval. Such MSP messages serve two purposes: fast link rate and FEC codec adaptation by the sender, as well as transmitting additional coded packets for compensating the detected losses. This mechanism will be described in detail in the next section. The FEC decoder ensures that the data packets are forwarded to the higher layers in the correct order as they have been encoded in the FEC encoding step at the sender before. In case of a packet loss on the wireless channel, the decoder is able to recover data packets if it receives sufficient redundancy information afterwards by the price of an increased delay. In the worst case that packets can not be recovered, that loss is detected at the decoder stage when the corresponding slot is ousted by subsequent packets. The resulting delay depends on the queue length and the packet rate.

B. PROACTIVE REPORTING AND RETRANSMISSION MECHANISM

The proactive reporting mechanism can be triggered by any receiver node on detecting a high packet loss at the FEC decoder and has already been described in [2]. That basic mechanism is used here with two modifications. Firstly, the Exponentially Weighted Moving Average (EWMA) filter is applied to the minimum PDR vector, similar to the case of regular report messages, and secondly, a new packet recovery mechanism has been added.

In addition to the MSP message, which carries the relevant information for the receiver statistics update, a second message is transferred in the same management frame but as an independent information element. This message contains information about undecodable (or missing) packets in the FEC decoder buffer, in detail the sequence number of the first one and their total number (f_{m}). In case the FEC encoder buffer still contains pending frames, the sender node can transmit additional coded packets. This is easily possible, since the RLNC provides the flexibility for creating an almost arbitrary number of coded frames for a block. For recovering the lost frames completely, at least f_{m} newly transmitted packets need to be received by the decoder. Taking the packet loss into account, the number of retransmitted packets f_{r} should be greater than f_{m}. After link rate and FEC coding scheme (t^*, c^*) have been updated at the sender on reception of the MSP message, we can assume that the degree of redundancy provided by the selected coding scheme c^* is sufficient for handling the expected packet loss. Thus, f_{r} can be calculated by multiplying f_{m} by the inverse code rate of c^*:

$$f_{\mathrm{r}} = \left\lceil f_{\mathrm{m}} \frac{n(c^*)}{k(c^*)} \right\rceil. \tag{1}$$

As will be explained in Section IV-A, $n(c^*)$ and $k(c^*)$ represent the number of coded, respectively uncoded, symbols per block of the selected FEC codec c^*. Together with this retransmission mechanism, the proactive reporting provides a method for preventing packet losses at higher layers in many cases, even under changing conditions.

IV. JOINT ADAPTATION OF LINK RATE AND FEC CODEC

In the original version of *EWRiM* [2], we used only one configurable but static FEC coding scheme that had to be empirically chosen. Additionally, the link rate decision was performed based on a fixed link layer PDR threshold. Under adverse conditions, this could lead to situations where a lower link rate is chosen when it could have been more efficient to use a higher link rate but with a higher degree of redundancy at the FEC layer. Therefore, we developed an improved approach that jointly selects link rate and FEC coding scheme, achieving the best possible performance under the given conditions. Our approach is based on a given minimum target PDR provided to higher layers by the FEC denoted as d_{FEC}. From this starting point, we select the best combination of transmission rate and coding scheme $(t^*, c^*) \in T \times C$ in terms of channel utilization fulfilling the PDR constraint.

The joint link rate and FEC codec selection consists of three steps, which will be described in the following subsections: (i) Loss Rate Calculation, (ii) Channel Allocation Estimation, and (iii) Joint Link Rate and Coding Scheme Selection.

A. LOSS RATE CALCULATION

Sliding window FEC is a relatively new approach to overcome the limitations of traditional block based FEC methods. Even though RLNC can be used for other purposes too, it can serve as a special FEC mechanism on network transmissions. *Caterpillar RLNC* [14] introduced a practical RLNC approach with a finite sliding window. The general concept of using a sliding window FEC at an intermediate MAC layer has already been described in the previous version of *EWRiM* [2]. Here, we introduce as novelty an adaptive version that allows adjusting the coding scheme dynamically.

Using RLNC as an FEC mechanism allows two different modes of operation, either applying the coding to all frames (*full coding*), or transferring the original packets uncoded and only the additional redundancy frames coded (*systematic coding*). While systematic coding is comparable to traditional FEC approaches and can provide a better recoverability for our use case (as will be shown subsequently), full coding might have advantage in other cases (e.g., benefiting from overhearing in multi-hop networks). The sliding window FEC coding scheme as used in our approach is defined by three parameters:

1) k: number of source packets (resp. symbols) per block
2) n: number of coded packets per block (including uncoded symbols in case of systematic coding)
3) w_{k}: size of the sliding window (counted on uncoded packets)

The following considerations also cover the case of conventional block (resp. generation) based coding schemes (without a sliding window) as a special case, where w_{k} is simply equal to k. Subsequently, we will use the notation $k/n/w_{\text{k}}$ for specifying a coding scheme. As mentioned before, C represents the set of available coding schemes and $c \in C$ a specific one. For referring to the specific elements $(k/n/w_{\text{k}})$ of a coding scheme c, we will use the function-like notation, for example $k(c)$, in the following. Since w_{k} denotes the number of source packets per window, but the following considerations also require the total number of packets in a window including the redundancy symbols, we will use w_{n} for that purpose. As shown in (2), w_{n} is calculated by multiplying w_{k} with the inverse of the *code rate* k/n:

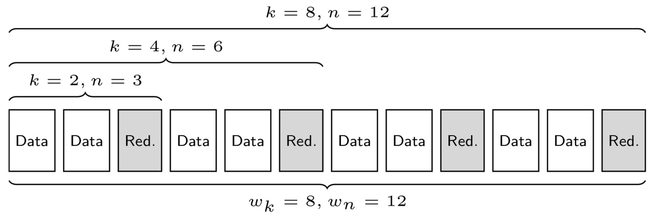


FIGURE 3. **Example for the generated packet sequence of equivalent coding schemes (2/3/8, 4/6/8, and 8/12/8).**

$$w_{\text{n}} = \left\lceil w_{\text{k}} \frac{n}{k} \right\rceil. \tag{2}$$

It should be noted that this coding scheme specification does not define the positions of redundancy frames within a block. Since the sliding window is independent of the block boundaries, redundancy packets can be placed at any position within a block, without weakening the coding, rather than at the end only as in traditional block (respectively generation) based coding schemes. This aspect is not relevant for our statistical considerations regarding the resulting packet loss, but could affect the delay for packet decoding. Therefore, we use an implementation that distributes the redundancy frames uniformly within a block. This has the effect that, for example, the coding schemes 4/6/32 and 8/12/32 are identical, not only for the loss rate calculation but also in the resulting sequence of generated redundancy packets. Fig. 3 illustrates this equivalence of coding schemes (for clarity of presentation, only with a widows size of 8).

At the receiver, all packets within a window can be decoded if w_{k} coded packets have been received.[1] Thus, the decoding of all packets is successful if at least w_{k} out of w_{n} frames are received. Assuming that the individual link layer packet losses are independent of each other (each with probability $1-d_{\text{link}}$, where d_{link} denotes the link layer PDR), the number of received frames (x) is binomially distributed, which can be expressed by:

$$B(x|1-d_{\text{link}}, w_{\text{n}}) = \binom{w_{\text{n}}}{x} (1-d_{\text{link}})^x d_{\text{link}}^{w_{\text{n}}-x}. \tag{3}$$

Therefore, the probability of loosing at most $w_{\text{n}}-w_{\text{k}}$ packets (which is the amount of redundancy) can be calculated using the cumulative binomial distribution as

$$F(w_{\text{n}}-w_{\text{k}}|1-d_{\text{link}}, w_{\text{n}}) = \sum_{i=0}^{w_{\text{n}}-w_{\text{k}}} \binom{w_{\text{n}}}{i} (1-d_{\text{link}})^i d_{\text{link}}^{w_{\text{n}}-i}. \tag{4}$$

1. We assume here that the random coefficient vectors within a window are really linearly independent of each other. The minimal probability of any linear dependency among the coefficient vectors is neglected for our calculations.

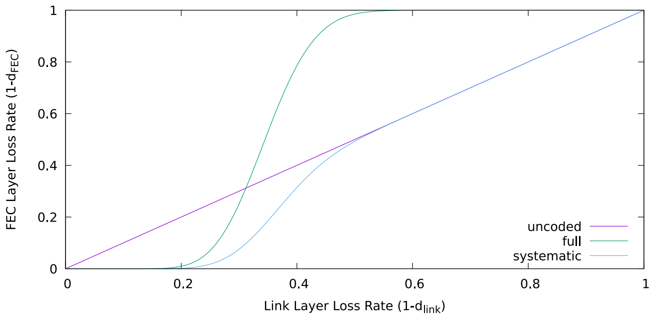


FIGURE 4. Example of FEC layer loss probability over link layer loss probability for the coding scheme 8/12/32.

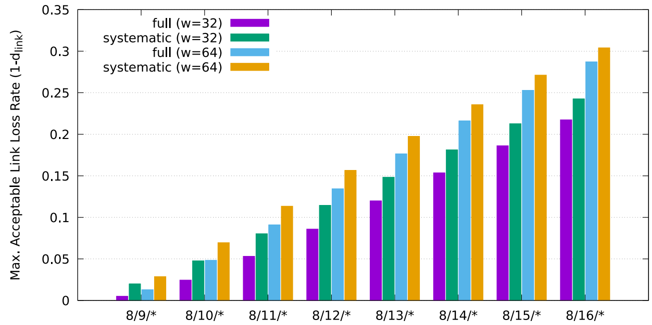


FIGURE 5. Example values of acceptable link loss rates for different coding schemes on a given FEC loss limit of 10^{-4}.

In case of full coding (transferring only coded frames), the resulting FEC layer PDR d_{FEC} can directly be derived from (4) as

$$d_{\text{FEC\_f}} = F(w_{\text{n}} - w_{\text{k}}|1 - d_{\text{link}}, w_{\text{n}}). \tag{5}$$

On systematic coding (transferring uncoded source frames plus coded redundancy frames), a packet can be received in two different ways, either directly (uncoded) with probability d_{link}, or indirectly by using coded redundancy packets similar to the fully coded case. The resulting FEC layer PDR can be expressed as

$$d_{\text{FEC\_s}} = d_{\text{link}} + (1 - d_{\text{link}})d_{\text{FEC\_f}}. \tag{6}$$

The fact that systematic coding provides a lower resulting packet loss rate than full coding might not intuitively be plausible, but it can be illustrated with a simple example: Let us assume a codec with a block size $k = 4$ with two redundancy packets ($n = 6$), and for simplicity a window size equal to the block size ($w_k = 4$). If, for example, the packets numbered 3, 5, and 6 got lost on the transmission, the FEC decoder would not be able to reconstruct any of the source frames because less than k (only 3) packets were received. But in case of systematic coding, three out of four packets are received, since the loss fortunately occurred on the redundancy packets (numbers 3 and 6) and only one source frame (number 5). These resulting loss probabilities ($1-d_{\text{FEC}}$) for the example coding scheme 8/12/32 are plotted in Fig. 4 together with the (hypothetical) fully uncoded case as a reference. As can be observed in this plot, systematic coding provides a lower resulting FEC layer loss rate than full coding under the same conditions.

For our purposes, we are interested in the question of which link layer PDR d_{link} is required for a specific coding scheme to achieve the given FEC layer PDR limit. These values can be computed for any given coding scheme and FEC layer PDR. If the target d_{FEC} is static, this can be done in advance or once while initializing the encoder. That allows us to use a simple numeric approximation without causing any significant runtime efforts.

Some example values for $d_{\text{FEC}} = 10^{-4}$ are shown in Fig. 5. It should be noted that larger window w_{k} provides a higher link loss tolerance than a coding scheme at the same code rate but with a shorter window. The resulting limits for d_{link} of a coding scheme c for fulfilling a given d_{FEC} will be denoted as $\vartheta(c)$, or more specifically as $\vartheta_{\text{s}}(c)$ and $\vartheta_{\text{f}}(c)$ when referring to systematic or full coding.

Applying an FEC coding to the transmission leads to extra costs. The most obvious one is the additional network overhead caused by the supplemental packets and larger frame lengths. This aspect will be discussed in detail in the following Section IV-B as an integral component of DEWRiM. Besides that, FEC encoder and decoder require buffers for at least w_k packets. This storage space requirement is neglected in our considerations; however, more important than the buffer space, FEC encoding and decoding require some computing effort that grows with the window size and which might be relevant for some use cases. On the encoder, the effort grows generally linearly with the window size, whereas the decoding effort scales generally with its third power. More detailed considerations regarding the computing effort can be found in [14]. The decoding effort is mainly caused by the matrix operations that can highly benefit from parallelization techniques using the Single Instruction Multiple Data (SIMD) features of modern microprocessors. Practical implementation aspects and performance measurements can be found in [29]. The average computing effort can additionally be reduced by actively managing the coding window with an ACK mechanism so that it only covers unacknowledged packets as in [15].

One adverse effect of FEC coded transmissions is the decoding delay in case of lost packets. This effect is minimized by the sliding window mechanism and by the chosen packet sequence, as illustrated in Fig. 3, but it can not be avoided. In the worst case, decoding of lost packets is delayed by the length of the coding window size. Depending on the application, a large window is not always useful, for example on time-critical constraints with a low packet rate. The packet delay caused be the FEC coding is analyzed in one of our evaluation scenarios in Section V-B.

The selection of an appropriate window size is not part of this article since it includes aspects going beyond the scope of our considerations, especially the computing effort and the application requirements. In our evaluation, we used a moderate window size of 32 (cf. Table 1). As a future improvement, a dynamic window size adjustment, e.g., as in [15], could being considered.

B. CHANNEL ALLOCATION ESTIMATION

Selecting the best pair of link layer transmission rate and FEC coding scheme (t^*, c^*) requires evaluating the transmission time for a whole block of frames rather than just a single packet. Obviously, the number of redundancy frames within a block is a major aspect influencing the channel resource utilization. But besides that, the length of frames depends on the encoding too. In addition to the required header for the FEC, the length of encoded frames needs to cover the maximum length of all previous frames within the encoding window. This causes additional overhead compared to uncoded transmissions on flows of diverging packet sizes.

Let $a(l, t)$ be a function that represents the channel allocation time of a packet with length l at link rate t including overheads like lower layer headers and preamble. Then, the channel allocation time for a whole block can be calculated as presented in (7):

$$a_{\text{block}}\big(l_{\text{avg}}, l_{\text{max}}, c, t\big) = n_{\text{uncoded}} a\big(l_{\text{avg}} + h_{\text{uncoded}}, t\big) + n_{\text{coded}} a(l_{\text{max}} + h_{\text{coded}}, t). \quad (7)$$

The number of coded, respectively uncoded, packets per block is expressed by n_{coded} and n_{uncoded}, which depends on the applied coding mechanism (with k and n as part of the coding scheme c):

$$n_{\text{uncoded}}(c) = \begin{cases} 0 & \text{on full coding} \\ k(c) & \text{on systematic coding} \end{cases} \quad (8)$$

$$n_{\text{coded}}(c) = \begin{cases} n(c) & \text{on full coding} \\ n(c) - k(c) & \text{on systematic coding.} \end{cases} \quad (9)$$

Furthermore, h_{uncoded} and h_{coded} represent the FEC header lengths for uncoded and coded packets. The average and maximum packet lengths within a window (l_{avg} and l_{max}) can easily be calculated if the packets in the window are known, but they need to be predicted as the selection of (t^*, c^*) affects the forthcoming packet transmissions.

For comparing different coding schemes, the allocation time can be normalized by the number of source symbols in a block as a_{avg}:

$$a_{\text{avg}}\big(l_{\text{avg}}, l_{\text{max}}, c, t\big) = \frac{a_{\text{block}}\big(l_{\text{avg}}, l_{\text{max}}, c, t\big)}{k(c)}. \quad (10)$$

The quotient of average packet length l_{avg} and average channel allocation time a_{avg} will be used as the link rate metric λ given in (11), which allows a comparison between different flows.

$$\lambda\big(l_{\text{avg}}, l_{\text{max}}, c, t\big) = \frac{l_{\text{avg}}}{a_{\text{avg}}\big(l_{\text{avg}}, l_{\text{max}}, c, t\big)}. \quad (11)$$

A difficult task is predicting the average and maximum packet lengths for the future transmissions. Since any general assumption about the traffic flow characteristic is almost impossible and could only be done for specific application scenarios, it can only rely on past information. Instead of calculating the average and maximum over a fixed window (for example, of the same size as the encoding w_{n}), we use the *EWMA* for l_{avg} and the *Exponentially Weighted Moving Standard Deviation (EWMSTD)* for estimating l_{max}. This provides higher flexibility than a fixed window by simply adjusting the weight α_l without much effort (as for example always iterating a long list of the past packet lengths). In step i, the calculation of the average length $l_{\text{avg},i}$ and the corresponding standard deviation $l_{\text{dev},i}$ is done as follows (for further information, see also [30]):

$$l_{\text{avg},i} = \alpha_l l_i + (1 - \alpha_l) l_{\text{avg},i-1} \quad (12)$$

$$l_{\text{dev},i} = \sqrt{(1 - \alpha_l)\Big(l_{\text{dev},i-1}^2 + \alpha_l\big(l_{\text{avg},i-1} - l_i\big)^2\Big)}. \quad (13)$$

Without the step index i, l_{avg} and l_{dev} always refer to the most recent values. These values are updated on each source packet of length l_i entering the encoding process. As an estimation for the maximum packet, we use the average packet length plus the standard deviation with a constant factor β (e.g., $\beta = 2$):

$$l_{\text{max}} = l_{\text{avg}} + \beta l_{\text{dev}}. \quad (14)$$

For well-known packet length distributions, it would be possible to evaluate the accuracy of this prediction. But since the general characteristic is unpredictable, we use this as a pragmatic solution and do not contemplate it further in our context. A possible misprediction of the average or maximum length would not have any fatal consequences but could, in the worst case, lead to an incorrect estimation for the channel utilization, and hence to a potentially suboptimal choice of (t^*, c^*), as will be explained hereafter.

C. JOINT LINK RATE AND CODING SCHEME SELECTION

The joint selection of link rate and coding scheme as a pair (t^*, c^*) is based on the minimum PDR vector $\overrightarrow{\delta^*}$ as a result of the aggregated receiver feedback as described in [2]. This vector contains the minimum measured PDR per link rate $t \in T$ among all receiver nodes. The joint selection algorithm is presented in Algorithm **1**. In general, a suitable coding scheme must be able to recover the losses for a given link rate t, which means that the observed PDR must reach at least the required PDR threshold $\vartheta(c)$ for a coding scheme candidate c. As mentioned in Section III-A3, DEWRiM uses a rate sampling mechanism that might cause additional packet losses, in the worst case up to the sampling ratio ϱ. Thus, ϱ is considered as a potential loss rate and the resulting PDR is $\delta_t^* - \varrho$.

As the first step in the selection process on lines 4 to 13, the algorithm selects c_t, for each link rate t, as the coding scheme with the minimum transmission overhead (maximum code rate) that fulfills the given PDR constraint d_{FEC}, which has been included in the calculation of $\vartheta(c)$. To illustrate this with a small example, assume that the measured PDR δ_t^* for a link rate t is 95% and that the sampling ratio ϱ is

Algorithm 1 Joint Codec and Link Rate Selection

Input: C, T, ϑ, $\overrightarrow{\delta^*}$, ϱ
Output: (t^*, c^*)
1: $c^* \leftarrow c_{\text{default}}$, $t^* \leftarrow t_{\text{default}}$
2: $a_{\min} \leftarrow \infty$
3: **for** $t \in T$ **do**
4: $\quad c_{\max} \leftarrow 0$
5: $\quad$ **for** $c \in C$ **do**
6: $\quad\quad$ **if** $(\vartheta(c) \leq \delta^*_t - \varrho) \wedge (\frac{k(c)}{n(c)} > c_{\max})$ **then**
7: $\quad\quad\quad c_{\max} \leftarrow \frac{k(c)}{n(c)}$
8: $\quad\quad\quad c_t \leftarrow c$
9: $\quad\quad$ **end if**
10: $\quad$ **end for**
11: $\quad$ **if** $c_{\max} = 0$ **then** ▷ no suitable coding scheme found
12: $\quad\quad$ **continue** ▷ skip link rate t
13: $\quad$ **end if**
14: $\quad a_{\text{temp}} \leftarrow a_{\text{avg}}(l_{\text{avg}}, l_{\max}, c_t, t)$
15: $\quad$ **if** $a_{\text{temp}} < a_{\min}$ **then**
16: $\quad\quad t^* \leftarrow t$
17: $\quad\quad c^* \leftarrow c_t$
18: $\quad\quad a_{\min} \leftarrow a_{\text{temp}}$
19: $\quad$ **end if**
20: **end for**

10%. This means that any coding scheme c must be able to fulfill the desired FEC layer PDR d_{fec} at a link layer PDR d_{link} of 85% (or even less).

Then, the lines 14 to 19 in the algorithm choose the pair (t^*, c^*) with the lowest resulting channel allocation time, using a_{avg} (cf. (10)).

This selection does not differentiate between coding schemes of equal code rates. As noted before, coding schemes with the same values for k and n and same window size are equivalent. The coding window size is implicitly included in this calculation due to the resulting values for $\vartheta(c)$, but not directly since it does not affect the code rate and the caused computing effort is not considered here.

V. EVALUATION AND RESULTS

After describing the internal mechanisms of DEWRiM, this section focuses on its detailed behaviour analysis in specific example scenarios, followed by the performance comparisons with other transfer mechanisms for two distinct use cases. For these investigations, we implemented DEWRiM[2] in ns-3 by using the *Kodo* library [31] for the FEC mechanism. In its current version, DEWRiM is applied to the legacy 802.11 versions *g* and *a*. Compared to the evaluation environment used in [2], we employed a more realistic and more challenging wireless channel model in the simulation setup:

- The *SpectrumPhy* and *SpectrumChannel* models
- A combination of *Nakagami* propagation loss model and *three-log-distance* propagation loss model
- The *Yans* error rate model

The general evaluation setup consists of one node in AP mode that delivers a multicast stream to a certain number of receivers in an 802.11 network. Our evaluations use up to 50 multicast receivers served by the AP node directly. This is large enough for covering many practical scenarios, though DEWRiM can theoretically support even larger number of receiver groups. As stated earlier, DEWRiM is intended for high-rate multicast traffic where the efficiency gain exceeds the additional management effort. Our approach also depends on enough data packets in a multicast stream for generating sufficient sample packets and for efficiently applying the FEC.

In one scenario, we are using the NORM protocol for reliable multicast content transmissions. Since no NORM implementations were available for the Network Simulator 3 (ns-3) environment, we used the Naval Research Laboratory (NRL) NORM [32] implementation in the Direct Code Execution (DCE) framework [33] (similarly to one of our previous publications [34]). The current DCE implementation relies on outdated versions of the *glibc* library. Therefore, the simulation had been encapsulated into a *Docker* container, which made it independent of the host system environment.

In the following sections, we first show the internal behavior of DEWRiM, followed by its performance comparison with other transfer methods. We study both fixed-rate and rate-adaptive traffic scenarios, using UDP and NORM transport protocols, respectively.

A. DETAILED PROTOCOL ANALYSIS

This section demonstrates the internal mechanisms of DEWRiM in two scenarios with different data traffic characteristics: first in an unreliable UDP case with a fixed-rate application traffic and then in a reliable NORM protocol case with an adaptive-rate application traffic. Except for the generated traffic pattern, both scenarios share the same setup with one sender node and ten receivers located around the sender in a spiral arrangement with distances between 10 m to 50 m, as shown in Fig. 6. This topology is deliberately chosen as a challenging case for multicast rate control, since the distance from sender, and thus channel conditions, largely vary among the receivers.

1) FIXED-RATE UDP TRAFFIC

Without loss of generality, non-adaptive traffic can be considered as the default case for multicast transmissions in today's applications. As the simplest case of this traffic type, we use a UDP stream with a fixed UDP payload size of 1024 bytes and a constant packet interval of 0.25 s with 2000 packets in total (resulting in a total duration of 50 s).

Fig. 7 illustrates the temporal behaviour of the rate selection process with the time as x-axis. Here, the selected link rate, the calculated link rate metric, and the sample frames refer to the left y-axis, while the selected FEC code rate corresponds to the right y-axis. We observe that the rate

2. Source code available at http://dx.doi.org/10.14279/depositonce-12735.

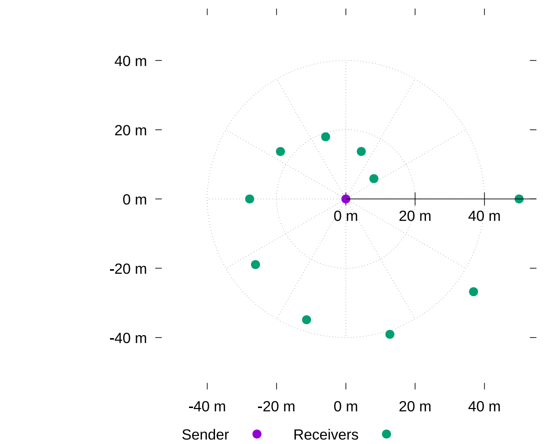


FIGURE 6. Node positions for the example scenarios.

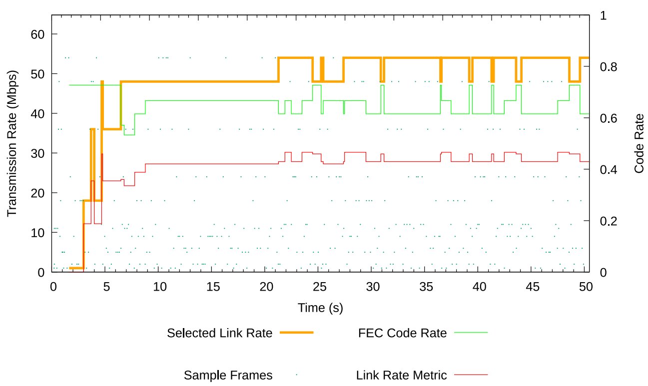


FIGURE 7. Rate selection on UDP traffic.

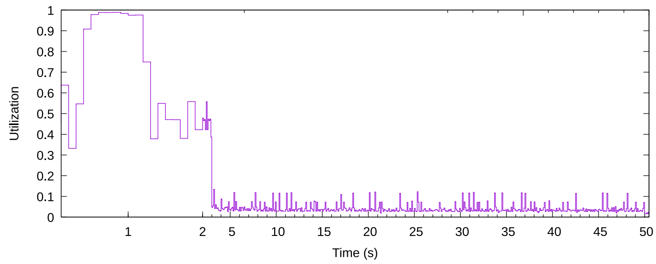


FIGURE 8. Channel utilization on UDP traffic.[3]

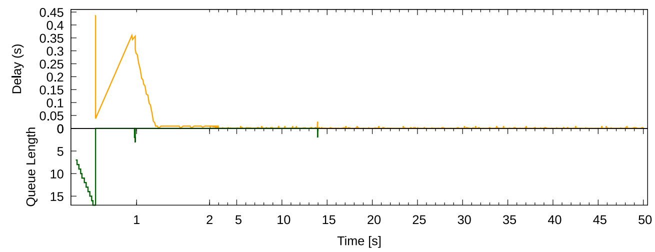


FIGURE 9. Delay and queue lengths on UDP traffic.[3]

selection starts at about 1.5 s after collecting sufficient data on the receiver statistics, and the rate control starts applying higher link rates at 3.0 s. The code rate of the chosen FEC codec has to be considered in parallel, as the FEC layer compensates the potential packet losses caused by the sampling process. When the rate selection switches from 36 Mbps to 48 Mbps at about 6.5 s, the FEC codec simultaneously changes to a higher degree of redundancy. The resulting calculated link rate metric, as explained in Section IV-B, shows the estimated effect of this combined selection. As can be seen in the following process, the link rate metric gets updated whenever the link rate or the FEC codec changes.

Since the application traffic remains constant for the whole duration, the effect of the link rate and FEC codec selection is directly visible in the resulting channel utilization in Fig. 8. It should be noted that a low utilization is the desired result in this scenario, since the application layer data rate is fixed and a lower channel utilization means a higher transmission efficiency in terms of resource consumption.

A possible side effect of the FEC layer is increased delay due to the later decoding of lost packets. The delay values in Fig. 9 are calculated for the whole transmission between sender and receiver applications. Correspondingly, the decoder queue length at the receiver reaches a maximum of 17 during the channel congestion due to lost frames. The resulting transmission delay is mainly caused by this decoding stall (17×0.25 s $= 0.425$ s). After the link rate is adjusted at about 2.0 s, the delay and receiver queue length remain at substantially low levels for the remaining period.

2) RATE-ADAPTIVE NORM TRAFFIC

Application or transport layer data-rate adaptivity in multicast networks is a challenging issue in general due to the necessity of avoiding feedback collisions (similar to the difficulties within the 802.11 layer). Therefore, multicast is not commonly used for bulk data transfers or similar applications that are trying to maximize the data rate according to channel conditions. One transport layer protocol aiming at such use-cases is NORM [3], which provides reliability and rate adaptation to the higher layers for multicast transmissions, similar to Transmission Control Protocol (TCP) for unicast. These characteristics have significant implications on our rate control approach. Instead of reducing the channel utilization while selecting more efficient link rates, NORM would use the gained capacity for a faster transmission. On the other hand, random packet losses on the network that are not caused by congestion could lead to a slower transmission than theoretically possible (similar to the behaviour of TCP).

This evaluation scenario shows the interaction between the NORM protocol and our rate control. The network setup is identical to the previous example, but instead of a fixed-rate equal-sized UDP data flow, we use NORM for transferring 10 MiB of data to all receiver nodes simultaneously.

3. Note the non-linear scaling on the x-axis; the first two seconds are expanded for readability.

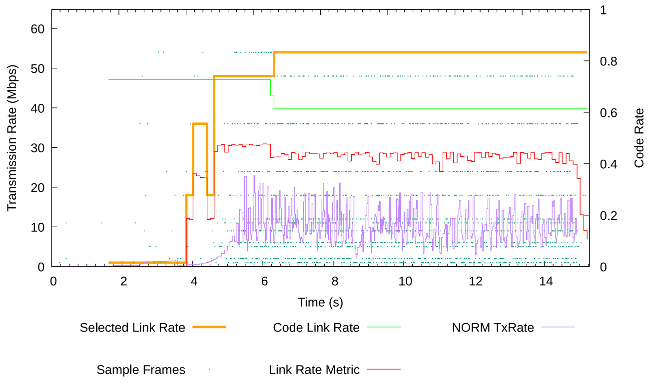


FIGURE 10. Rate selection on NORM traffic.

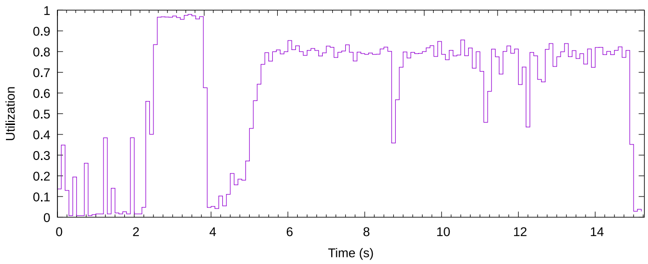


FIGURE 11. Channel utilization on NORM traffic.

Fig. 10 depicts the progression of the selected link rate and FEC code rate, similar to the previous example. In addition, the purple line indicates the transmission rate selected by the NORM protocol layer. Since we used the TCP-Friendly Multicast Congestion Control (TFMCC) as the default NORM mechanism, this rate is mainly influenced by packet loss events detected and reported at the NORM layer. Due to FEC layer inside DEWRiM, they are mostly not caused by link layer losses, but by the pre-FEC drop mechanism as explained in Section III-A. Since NORM uses a low link rate in the beginning and increases it based on successful reception (slow start phase), only a few sample frames are available in the first seconds (small green dots in the plot). Therefore, the start of the link rate and FEC codec selection is delayed compared to the previous example.

After DEWRiM selects a rate higher than the basic rate at about 3.8 s, the NORM rate also starts increasing and then fluctuates at around 10 Mbps for the rest of the transmission. It should be noted that the calculated link rate metric fluctuates a little, while the link rate and FEC codec remain constant. This is caused by the packet sizes, which are included in the calculation as explained in Section IV-B. Even though NORM uses the maximum packet size for data transmission, the NORM control messages are much smaller, which affects the calculation results.

Fig. 11 presents the corresponding channel utilization. After the slow-start phase of NORM and the initial rate adjustment of DEWRiM, the utilization is kept relatively high but below the full saturation level. This example demonstrates that DEWRiM and NORM generally work well together for transferring multicast data reliably and efficiently over 802.11 networks. But it also shows two aspects leaving some potential for further optimizations. Firstly, the slow start phase of NORM and the starting phase of DEWRiM interfere mutually, resulting in a slower rate adaption compared to fixed-rate traffic. Secondly, the achieved channel utilization stays below the available capacity. These behaviors could be improved by tuning NORM and DEWRiM parameters, e.g., by increasing the TX queue length in DEWRiM, which requires a more detailed investigation.

TABLE 1. DEWRiM parameters.

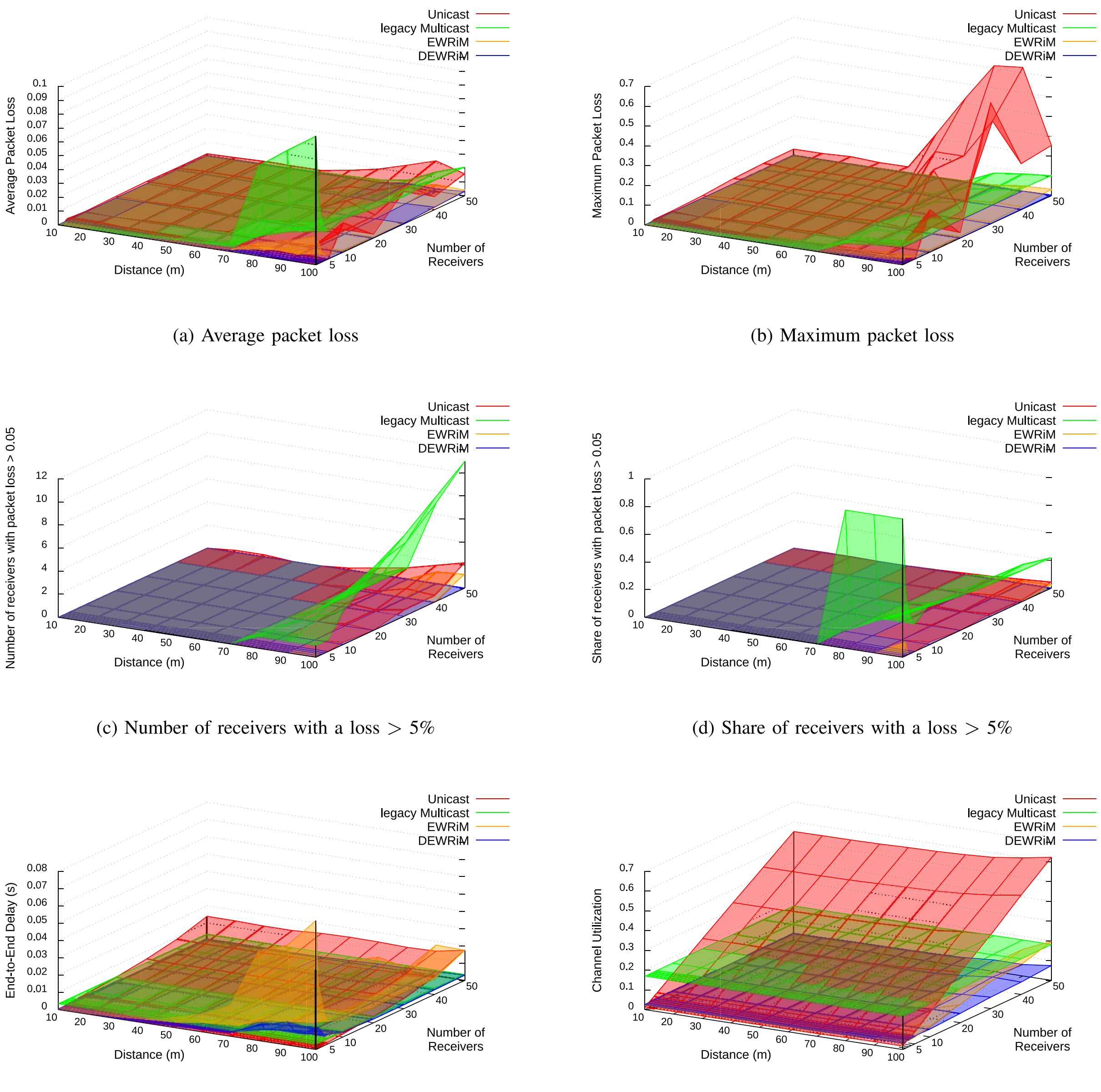
Parameter	Value
MAC TX queue length	64
encoding schemes	8/8/32 to 8/20/32
systematic coding	enabled
proactive report threshold	16 (50% of FEC window size)

B. PERFORMANCE ANALYSIS—UDP SCENARIO

As the first of two comparative performance evaluations, we consider a simple UDP scenario with constant packet rate and packet size. It is inspired by a Real-time Transport Protocol (RTP) audio transmission of 128 kbps that results in a packet interval of 20 ms and a payload size of 332 bytes. In this scenario, DEWRiM is compared to legacy multicast as well as individual unicast transmissions. We use a spiral layout as shown in Fig. 6 with varying receiver group sizes (1 to 50) and maximum distances from the sender (10 m to 100 m), while the minimum distance is fixed at 10 m. Each configuration has been simulated five times. For showing the improvements of DEWRiM over EWRiM, we also include the older version into this comparison. Essential configuration parameters used for DEWRiM are shown in Table 1.

The results are presented in Fig. 12. The first plot (Fig. 12a) shows the average loss among all nodes in the particular setup. In the unicast case, the average loss rises with the distance and the number of receivers to a maximum of about 2.2%, indicating that the loss is mainly caused by the channel utilization. By contrast, the legacy multicast case shows the highest loss on larger distances with only one receiver. This is caused by the spiral layout since the single receiver is located at the maximum distance while all additional ones are placed closer to the sender node. In the EWRiM case, the packet loss is significantly lower than on legacy multicast, but on long distances the static FEC coding scheme is not able to recover all link layer packet losses so that the resulting loss above the FEC layer rises as well. The multicast transmission with DEWRiM induces almost no packet loss, with only small exceptions for many receivers at long distances.

For investigating the averaging effect in the first plot, the second plot (Fig. 12b) presents the maximum loss experienced on a single node. This points out clearly that the loss is not evenly distributed among the receivers. For example, it



(a) Average packet loss

(b) Maximum packet loss

(c) Number of receivers with a loss $> 5\%$

(d) Share of receivers with a loss $> 5\%$

(e) Packet delay

(f) Channel utilization

FIGURE 12. **Performance comparison of DEWRiM with EWRiM, legacy multicast, and unicast-based communication in the UDP scenario.**

reaches a maximum of 63% in the case of unicast transmissions, while no receiver experiences any significant packet loss in the case of DEWRiM.

The level of toleration of packet losses depends on the application scenario and other factors, such as higher layer protocols or codecs on audio or video transmissions. Therefore, a general packet loss limit can hardly be given; but, for instance, values between 1% to 5% can often be found in the literature for audio transmissions. For example, significant impacts to the voice quality of a H.323 transmission occurs on packet loss rates of 5% to 10% according to [35, p. 501]. From that perspective, the question is, how many receivers can be satisfied in a given scenario. Fig. 12c shows the number of receivers above an assumed packet loss threshold of 5% (which should be compared to the total number of receivers) while Fig. 12d presents the share of nodes above that limit. In this scenario, we observe that only DEWRiM is able to serve all receivers with a loss rate below the threshold in all cases.

Another relevant metric for certain use cases is the end-to-end delay at the application layer. This covers not only the transmission delay, but also the effects caused by the higher layers, especially the FEC decoder queue at the receiver in case of EWRiM/DEWRiM. As can be seen in Fig. 12e, the delay in the unicast case grows mainly with the number of receivers while the distance has only a slight impact. For EWRiM and DEWRiM, the increased delay at higher distances is mainly caused by the packet losses on the link layer and the resulting receiver-side queueing in the FEC decoder until the frames can be reconstructed. Since we use a spiral layout (Fig. 6), the lower delay for the cases with an increasing number of receivers is caused by an averaging effect due to many receivers sharing better conditions than the most distant one. The increased delay is the price that has to be paid for the FEC mechanism. In case of EWRiM, the delay exceeds those of all other methods, while that behavior is significantly improved in DEWRiM due to its more flexible FEC encoding and additional repair mechanisms.

A third metric, which is usually not directly recognizable for a user unless the channel is congested, is the channel utilization (Fig. 12f). In case of unicast transmissions, the utilization mainly grows with the number of receivers. Since transmissions in the legacy multicast case are completely independent of the receiver conditions and numbers, the utilization remains constant across the whole range. EWRiM shows a low utilization for largest part, but rises at high distances and many receivers up to the level of the legacy multicast case when choosing the slowest transmission rate as well. This behaviour differs from the results in our previous publication [2] because we adopt a more challenging propagation loss model for this comparison. In the DEWRiM case, maximum distance and the number of receivers have only small effects on the utilization. The maximum distance results in the selection of a lower link rate, which is a clear source of increased utilization at higher distances. However, the increased packet losses in combination with proactive reporting and retransmission mechanism, as described in Section III-B, can also have some negative impact at large distances. Because of the more flexible FEC coding, DEWRiM is able to use higher transmission rates than EWRiM, even under those conditions, by compensating the increased link layer loss with a higher degree of redundancy. Summarizing the utilization comparison, DEWRiM scales well with the number of nodes and distances, while unicast performs better only for very small amounts of receivers (up to two).

C. NORM VS. TCP PERFORMANCE COMPARISON

For a second comparative analysis, we adopt a reliable content delivery scenario. In this scenario we apply the NORM protocol [3], using its NRL implementation [32]. The multicast transmission is realized by using NORM as the transport layer protocol on top of DEWRiM, as well as on EWRiM and legacy multicast for comparison. Also included in the analysis are two unicast TCP cases, one with simultaneous transmissions to all receivers and another one with sequentially scheduled flows avoiding interferences among each other.

We employ the same network layout and DEWRiM parameters as the previous scenario. NORM protocol parameters are empirically set, as given in Table 2. In this scenario, 10 MiB of data are transferred to each receiver and the individual times until the transfer completion is measured. As the number of receivers grows, the simultaneous TCP setup becomes increasingly unstable and the TCP connections run into timeouts preventing meaningful results for those cases. Therefore, we limit the experiments to a maximum of 20 receivers in this scenario.

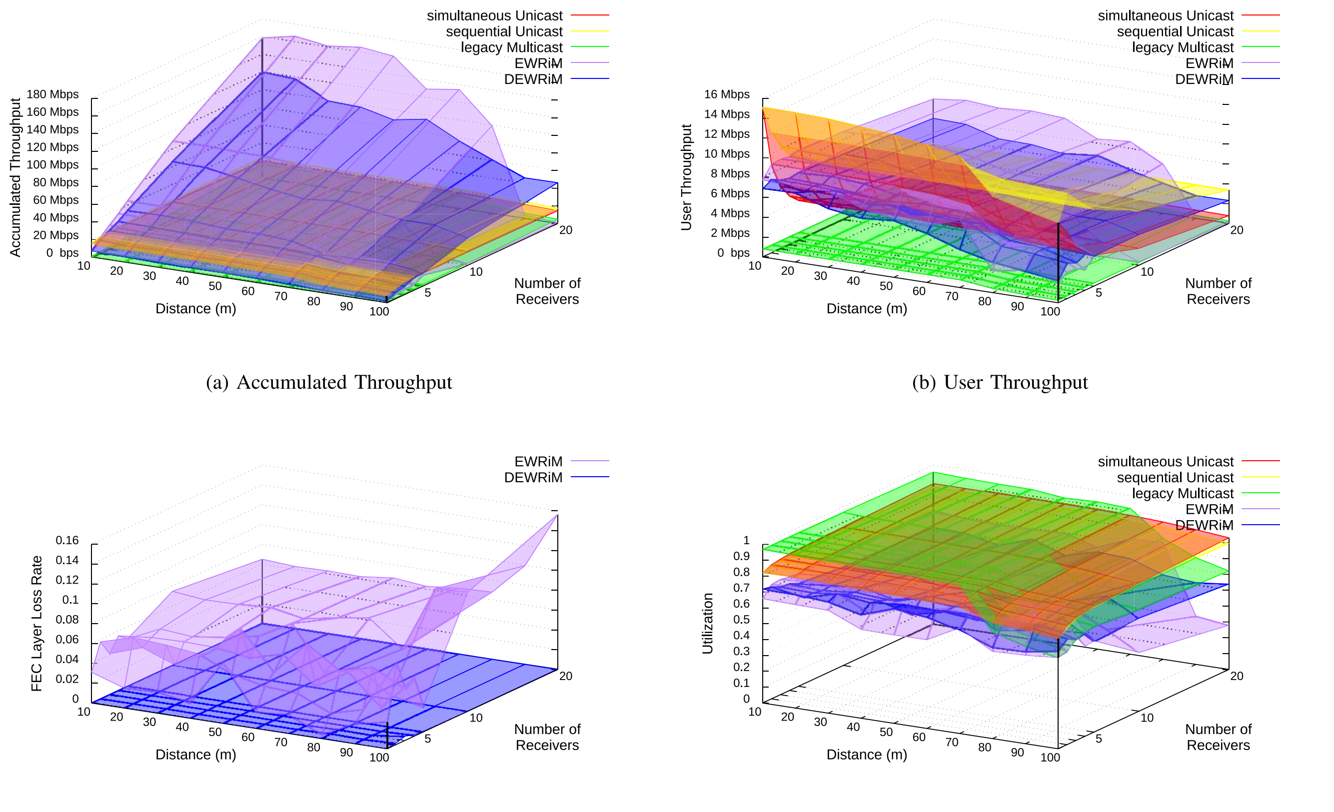
TABLE 2. NORM parameters.

Parameter	Value
block size	64
max. parity	16
auto parity	6
congestion control	TFMCC
backoff factor	4.0

Fig. 13a shows the average individual throughputs, calculated using the time until all transfers are completed (same time for all receivers). These individual throughputs are summed up as a (virtual) network throughput. Since the same transfer is counted multiple times in the multicast case, these values rise above the channel capacity on EWRiM/DEWRiM with many receivers. NORM on EWRiM performs well up to a distance of 80 m and outperforms all other methods for ten or more receivers. However, at distances above 80 m, EWRiM shows substantial performance drops, even below the legacy multicast case, due to the use of static FEC coding scheme. In comparison, DEWRiM performs significantly better in the case of longer receiver distances due to the more flexible FEC coding and the retransmit mechanism. On the other hand, the relatively lower performance at shorter distances is presumably caused mainly by the pre-FEC drop mechanism as explained in Section III-A. Due to the packet loss that becomes visible, the NORM layer reduces the transmission rate and can not fully utilize the channel capacity. As a minor effect, the higher amount of redundancy packets in the adaptive FEC layer of DEWRiM could also contribute to this behaviour.

Fig. 13b presents a more receiver-centric view of throughput performance, where the transmission duration is calculated from the common point when the transfer is initiated to the point when the individual transfer is finished. This includes the time of waiting for the transmission to start in the sequential TCP case. So, the transmissions share the same start times, but have individual end times. The resulting plot shows a different characteristic than the previous one. Similar to the previous plot, EWRiM performs well until 80 m, but drops dramatically beyond that point. In comparison, DEWRiM starts slightly declining with the distance but is able to serve distant receivers too. By using this user-centric metric, the NORM transfer on DEWRiM reaches a level on par with parallel TCP already at three receivers, and outperforms it for all larger receiver groups (independently of the distance).

For another perspective on EWRiM versus DEWRiM comparison, the FEC layer loss is plotted in Fig. 13c. It



(a) Accumulated Throughput

(b) User Throughput

(c) FEC Layer Loss

(d) Channel Utilization

FIGURE 13. **Performance comparison of DEWRiM with EWRiM, legacy multicast and unicast-based communication in the reliable content delivery scenario.**

shows the packet loss measured between the FEC encoder input at the sender and the corresponding decoder outputs at the receivers. Compared to the packet loss metric in the UDP scenario, it does not include the loss caused by the pre-FEC drop and other higher layer effects. While DEWRiM is able to compensate almost all losses, EWRiM shows noticeable loss rates, which have to be handled at the transport layer by NORM. Because NORM tries to adapt to the available channel capacity, this metric is a good indicator for the efficiency of the FEC layer under more stressing conditions than in the UDP scenario, which demonstrates significant improvements of DEWRiM over EWRiM. The plot also shows that the newly introduced adaptive FEC coding scheme selection works as expected and can almost fully compensate the link-layer loss.

The congestion control mechanism used by NORM is TFMCC [36], which utilizes the experienced packet loss for calculating the data transmission rate. This mechanism intends to behave compatible with the TCP congestion control and should result in comparable data rates. Similar to TCP, the TFMCC reacts sensitively to packet losses, even if they are caused by effects other than congestion. The channel utilization can be used as a metric for assessing the rate adaption performance at the transport layer. Fig. 13d shows the achieved utilizations in this scenario. In the case of legacy multicast, the utilization remains close to 1 until 70 m and drops significantly beyond that distance. This indicates that the link layer loss causes NORM to reduce the transmission rate event though the channel capacity would allow a higher rate. Therefore, NORM stays below its potential in that case. In comparison, both TCP cases achieve a utilization of 80% to 90%. With EWRiM, the utilization remains lower and drops significantly at higher distances due to the packet loss not being covered by the FEC layer. In comparison, DEWRiM achieves even lower utilization levels for most of the parameter space, while providing comparably higher throughput at larger distances. The relatively low utilization indicates that the combination of DEWRiM and NORM does not fully exploit their throughput potential and there is some room for further improvement through the joint optimization of their parameter configurations.

These performance results for different scenarios show that DEWRiM successfully exploits the inherent benefits of the wireless multicast concept for reliably distributing data over 802.11 networks. In our current implementation, the internal mechanisms of NORM and DEWRiM do not interact with each other optimally, so that future optimizations could provide further improvements in performance. In this work, our focus has been on the link-layer multicast efficiency and reliability, where NORM was used as an auxiliary transport protocol for evaluating DEWRiM in scenarios with adaptive rate of application traffic. Therefore, the joint optimization of NORM and DEWRiM configuration parameters falls beyond the scope of this paper. Nevertheless, the current version already offers a scalability and efficiency that can not be achieved with unicast and legacy multicast data transfers, as evidenced by the evaluation results.

VI. CONCLUSION

In this article, we presented DEWRiM as a major improvement of our EWRiM protocol previously reported in [2]. The adaptive FEC mechanism provides higher reliability as well as better transmission efficiency in comparison with the static codec case, since it allows the selection of higher link rates. The packet-level FEC within the MAC layer presents a beneficial approach for compensating the link-layer packet loss on multicast transmissions compared to multiple proactive transmissions (like in the GCR–UR) or single packet retransmissions. Treating the selection of link rate and FEC codec as a joint problem significantly improved the transmission efficiency as well as the reliability. As a general approach, DEWRiM provides a solution for different multicast use cases and is not limited to specific scenarios like real-time video distribution.

Our simulations have shown significant improvements of efficiency and performance compared to the other transfer methods that are in current use. In the UDP use case, DEWRiM performs significantly better than the traditional transfer methods; it provides lower packet loss and channel utilization than individual unicast flows and legacy multicast and scales well with the number of receivers. Compared to EWRiM, it improves the handling of difficult channel conditions, especially large transmission distances, due to the better error correction capabilities.

The content delivery use case that employs NORM protocol at the transport layer has shown that DEWRiM is able to maintain the conceptual scalability of multicast for reliable data transfers. Considering the combined receiver throughput as the main comparison metric, the multicast transfer based on DEWRiM and NORM significantly outperforms the traditional, TCP-based unicast transmissions, as well as other multicast methods.

The current version of DEWRiM covers only the legacy versions 802.11b/g/a. Our ongoing research aims at extending DEWRiM to more recent versions (802.11n and beyond). This becomes a more challenging multidimensional problem, as those versions offer multiple independently adjustable parameters (MCS, channel width, and spatial streams). But the main concepts presented here, especially the adaptive FEC layer, would be generally applicable and serve as a solid baseline for the future work. Other possible improvements in future work could cover aspects like dynamic coding window size adjustment and an automated mechanism for enabling different multicast methods (e.g., legacy multicast, DMS, and DEWRiM) depending on the traffic flow characteristics and the receiver group size.

TABLE 3. Summary of the notations used in the paper.

C	set of available FEC codecs
T	set of available link rates
c^*	selected FEC codec ($c^* \in C$)
t^*	selected link rate ($t^* \in T$)
f_{m}	number of missing frames in the FEC decoder buffer
f_{r}	number of frames to be retransmitted
$k(c)$ or short k	number of source frames per block of FEC codec $c \in C$
$n(c)$ or short n	total number of frames per block of FEC codec $c \in C$
$w_{\mathrm{k}}(c)$ or short w_{k}	number of source frames per window of FEC codec $c \in C$
$w_{\mathrm{n}}(c)$ or short w_{n}	total number of frames per window of FEC codec $c \in C$
d_{link}	PDR at link layer
d_{FEC}	PDR delivered by FEC layer (FEC_f for full coding and FEC_s for systematic coding)
$B(\dots)$	binomial distribution function
$F(\dots)$	cumulative binomial distribution function
$\vartheta(c)$	limits for d_{link} on coding scheme $c \in C$ for fulfilling a given FEC layer threshold d_{FEC} ($\vartheta_{\mathrm{s}}(c)$ for $d_{\mathrm{FEC\_s}}$ and $\vartheta_{\mathrm{f}}(c)$ for $d_{\mathrm{FEC\_f}}$)
l	frame length ($l_{\max}$: maximum, l_{avg}: average)
$a(l,t)$	channel allocation time function for a frame of length l at link rate $t \in T$
$a_{\mathrm{block}}(l_{\mathrm{avg}}, l_{\max}, c, t)$	channel allocation time function for a block with average/maximum frame lengths $l_{\mathrm{avg}}/l_{\max}$ at coding scheme $c \in C$ and link rate $t \in T$
$a_{\mathrm{avg}}(l_{\mathrm{avg}}, l_{\max}, c, t)$	average channel allocation time function for a frame in a block with average/maximum frame lengths $l_{\mathrm{avg}}/l_{\max}$ at coding scheme $c \in C$ and link rate $t \in T$
n_{uncoded}	number of uncoded frames per block
n_{coded}	number of coded frames per block
h_{uncoded}	header length of a uncoded frame
h_{coded}	header length of a coded frame
$\lambda(l_{\mathrm{avg}}, l_{\max}, c, t)$	link rate metric
β	constant factor for standard deviation of frame length
$\overrightarrow{\delta^*}$	minimum measured PDR vector as used in [2]
$\overrightarrow{\delta^*_t}$	minimum measured PDR for link rate $t \in T$
$c_t \in C$	optimal coding scheme selected for link rate $t \in T$ for fulfilling the given target PDR $f(c)$
ϱ	sampling ratio
α_l	weight for frame length EWMA/EWMSTD calculation

NOMENCLATURE

The notations used throughout the paper are summarized and explained in Table 3.

REFERENCES

[1] Linux wireless kernel community. "Linux mac80211 minstrel rate control." [Online]. Available: https://wireless.wiki.kernel.org/en/developers/documentation/mac80211/ratecontrol/minstrel (Accessed: Mar. 22, 2022).

[2] T. Geithner and F. Sivrikaya, "Transmission rate sampling and selection for reliable wireless multicast," *Wireless Commun. Mobile Comput.*, vol. 2020, Sep. 2020, Art. no. 8850198. [Online]. Available: https://www.hindawi.com/journals/wcmc/2020/8850198/

[3] B. Adamson, C. Bormann, M. Handley, and J. Macker, "NACK-oriented reliable multicast (NORM) transport protocol," IETF, RFC 5740, Nov. 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5740.txt

[4] Y. Shin, G. Lee, J. Choi, J. Koo, S. Lee, and S. Choi, "InFRA: Interference-aware PHY/FEC rate adaptation for video multicast over WLAN," in *Proc. 14th Annu. IEEE Int. Conf. Sens. Commun. Netw. (SECON)*, Jun. 2017, pp. 1–9.

[5] J.-H. Park, H.-S. Kim, and W.-T. Kim, “DM-MQTT: An efficient MQTT based on SDN multicast for massive IoT communications,” *Sensors*, vol. 18, no. 9, p. 3071, 2018.

[6] *IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks—Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11-2012, Mar. 2012.

[7] *ISO/IEC/IEEE International Standard for Information Technology—Telecommunications and Information Exchange Between Systems local and Metropolitan Area Networks—Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: MAC Enhancements for Robust Audio Video Streaming (Adoption ofIEEE Std 802.11aa-2012)*, ISO/IEC/IEEE Standard 8802-11:2012/Amd.2:2014(E), Mar. 2014.

[8] J. Kuri and S. K. Kasera, “Reliable multicast in multi-access wireless LANs,” in *Proc. IEEE INFOCOM Conf. Comput. Commun. 18th Annu. Joint Conf. IEEE Comput. Commun. Soc.*, 1999, pp. 760–767.

[9] S. K. S. Gupta, V. Shankar, and S. Lalwani, “Reliable multicast MAC protocol for wireless LANs,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, vol. 1, May 2003, pp. 93–97.

[10] A. Chen, D. Lee, G. Chandrasekaran, and P. Sinha, “HIMAC: High throughput MAC layer multicasting in wireless networks,” in *Proc. IEEE Int. Conf. Mobile Ad Hoc Sensor Syst.*, Oct. 2006, pp. 41–50.

[11] Y. D. Park, S. Jeon, K. Kim, and Y.-J. Suh, “RAMCAST: Reliable and adaptive multicast over IEEE 802.11n WLANs,” *IEEE Commun. Lett.*, vol. 20, no. 7, pp. 1441–1444, Jul. 2016.

[12] Y. D. Park, S. Jeon, J. Jeong, and Y.-J. Suh, “FlexVi: PHY aided flexible multicast for video streaming over IEEE 802.11 WLANs,” *IEEE Trans. Mobile Comput.*, vol. 19, no. 10, pp. 2299–2315, Oct. 2016.

[13] B. Li, S. Bi, R. Zhang, Y. Jiang, and Q. Li, “Random network coding based on adaptive sliding window in wireless multicast networks,” in *Proc. IEEE 83rd Veh. Technol. Conf. (VTC Spring)*, 2016, pp. 1–5.

[14] S. Wunderlich, F. Gabriel, S. Pandi, F. H. P. Fitzek, and M. Reisslein, “Caterpillar RLNC (CRLNC): A practical finite sliding window RLNC approach,” *IEEE Access*, vol. 5, pp. 20183–20197, 2017.

[15] F. Karetsi and E. Papapetrou, “A low complexity network-coded ARQ protocol for ultra-reliable low latency communication,” in *Proc. IEEE 22nd Int. Symp. World Wireless Mobile Multimedia Netw. (WoWMoM)*, 2021, pp. 11–20.

[16] A. Cohen, D. Malak, V. B. Bracha, and M. Médard, “Adaptive causal network coding with feedback,” *IEEE Trans. Commun.*, vol. 68, no. 7, pp. 4325–4341, Jul. 2020.

[17] K. Yamamoto and H. Yomo, “Rate adaptation for robust and low-latency video transmissions using multi-AP wireless LAN,” *IEICE Trans. Commun.*, vol. ET05.8, no. 2, pp. 177–185, 2022.

[18] H. Liu and S. Mathur, “A method and system for adapting forward error correction in multicast over wireless networks,” W.O. Patent WO2 009 134 220, 2008.

[19] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, “Link-level measurements from an 802.11b mesh network,” *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 121–132, Aug. 2004.

[20] D. Halperin, “Simplifying the configuration of 802.11 wireless networks with effective SNR,” Ph.D. dissertation, Comput. Sci., Univ. Washington, Washington, DC, USA, 2012.

[21] W.-S. Lim, D.-W. Kim, and Y.-J. Suh, “Design of Efficient multicast protocol for IEEE 802.11n WLANs and cross-layer optimization for scalable video streaming,” *IEEE Trans. Mobile Comput.*, vol. 11, no. 5, pp. 780–792, May 2012.

[22] J. Villalón, Y. Seok, T. Turletti, P. Cuenca, and L. Orozco-Barbosa, *ARSM: Auto Rate Selection Multicast Mechanism for Multi-rate Wireless LANs*. Berlin, Germany: Springer, 2006, pp. 239–250. [Online]. Available: https://link.springer.com/chapter/10.1007/11872153_21

[23] S. Kim and B.-S. Kim, “Rate-adaptive MAC protocol for wireless multicast over OFDMA-based MANETs,” *Wireless Pers. Commun.*, vol. 56, no. 4, pp. 675–692, 2011, doi: 10.1007/s11277-009-9839-4.

[24] S. Paris, N. Facchi, and F. Gringoli, “Chapter 3—Rate adaptation algorithms for reliable multicast transmissions in wireless LANs,” in *Modeling and Simulation of Computer Networks and Systems*, M. S. Obaidat, P. Nicopolitidis, and F. Zarai, Eds. Boston, MA, USA: Morgan Kaufmann, 2015, pp. 55–94.

[25] K. Mansour, I. Jabri, and T. Ezzedine, “A multicast rate adaptation algorithm over IEEE 802.11aa GCR block ack scheme,” in *Proc. 13th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, 2017, pp. 1431–1436.

[26] E. Coronado, R. Riggio, J. Villalón, and A. Garrido, “Joint mobility management and multicast rate adaptation in software–defined enterprise WLANs,” *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 2, pp. 625–637, Jun. 2018.

[27] E. Coronado, R. Riggio, J. Villalón, and A. Garrido, “Efficient real-time content distribution for multiple multicast groups in SDN-based WLANs,” *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 1, pp. 430–443, Mar. 2018.

[28] V. Gupta, C. Gutterman, Y. Bejerano, and G. Zussman, “Experimental evaluation of large scale WiFi multicast rate control,” in *Proc. IEEE INFOCOM 35th Annu. Int. Conf. Comput. Commun.*, 2016, pp. 1–9.

[29] S. Wunderlich, J. A. Cabrera, F. H. P. Fitzek, and M. Reisslein, “Network coding in heterogeneous multicore IoT nodes with DAG scheduling of parallel matrix block operations,” *IEEE Internet Things J.*, vol. 4, no. 4, pp. 917–933, Aug. 2017.

[30] T. Finch. “Incremental calculation of weighted mean and variance.” Jan. 2009. [Online]. Available: https://fanf2.user.srcf.net/hermes/doc/antiforgery/stats.pdf

[31] M. V. Pedersen, J. Heide, and F. H. P. Fitzek, “Kodo: An open and research oriented network coding library,” in *Proc. Netw. Workshops*, 2011, pp. 145–152.

[32] U.S. Naval Research Laboratory. “NACK-oriented reliable multicast (NORM).” 2017. [Online]. Available: http://cs.itd.nrl.navy.mil/work/norm/ (Accessed: Jun. 15, 2017).

[33] H. Tazaki *et al.*, “Direct code execution: Revisiting library OS architecture for reproducible network experiments,” in *Proc. 9th Int. Conf. Emerg. Netw. Exp. Technol. (CoNEXT)*, 2013, pp. 217–228.

[34] T. Geithner and F. Sivrikaya, “Adaptive reliable multicast in 802.11 networks,” in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6.

[35] K. Mansfield and J. Antonakos, *Computer Networking for LANS to WANS: Hardware, Software and Security Networking*, Course Technol., Inc., Bostan, MA, USA, 2009.

[36] W. Kammoun and H. Youssef, “TFMCC-based on a new equation for multicast rate control,” in *Proc. 12th IEEE Int. Conf. Electron. Circuits Syst.*, 2005, pp. 1–4.