On the Benefit of Forward Error Correction at IEEE 802.11 Link Layer Level

Floris van Nee and Pieter-Tjerk de Boer

Centre for Telematics and Information Technology (CTIT)
Faculty of Electrical Engineering, Mathematics and Computer Science
University of Twente, Enschede, The Netherlands
f.d.vannee@student.utwente.nl, ptdeboer@cs.utwente.nl

(This paper appeared in Proceedings of the 17th EUNICE Workshop, 2011. The final publication is available at www.springerlink.com.)

Abstract. This study examines the error distribution of aggregated MPDUs in 802.11n networks and whether or not forward error correction like raptor coding at the link layer would be useful in these networks. Several experiments with Qualcomm 4x4 802.11n hardware were performed. Two devices were used in a data link, while a third device sniffed all transmitted packets. The collected data was analyzed and used to calculate the packet error rate which would be obtained if FEC was used in order to determine whether FEC is useful at the link layer. It is shown that the error distribution of A-MPDUs does not follow the binomial distribution. Because of this, the performance of FEC in real networks is worse than for theoretical cases where a binomial distribution is assumed. Therefore, other ways to decrease the packet error rate have more impact than forward error correction.

1 Introduction

Wireless Local Area Networks (WLAN) play an important role in home video networking, because they are cost-effective to install and easy to set-up. However, there are still areas where WLAN needs to improve in order to deliver the performance needed to stream multiple High Definition (HD) streams simultaneously. One of the reasons for this is that video streaming poses different requirements on the transfer link than normal web browsing does. For example, a much lower Packet Error Rate (PER) is required, because video applications are more sensitive to errors in the communications link. Lost packets require retransmission which cause higher latency and lower throughput. To the end user this may be noticeable as artifacts on the video display.

Over the past few years there have been many studies on improving the link quality to allow streaming of multiple video streams. As latency is the most important aspect of video streaming, most studies focus on keeping latency as low as possible by preventing packet loss. One study tried to achieve this by adjusting the algorithm for rate control to use the Signal-to-noise Ratio (SNR) in determining the ideal transmission rate [4]. Another study used forward error correction at the application layer in order to correct errors which occur during transmission [1]. Both studies show some positive results, however they do not use devices which support the latest version of the IEEE 802.11 standard. This standard, IEEE 802.11n, introduces a couple of important improvements for transmitting large amounts of data which make it especially useful for video streaming.

The most important improvements in the 802.11n standard for video streaming are higher data rates and the possibility to use frame aggregation [5]. MPDU aggregation makes it possible to send up to 64 data frames in one single Aggregated MPDU (A-MPDU) frame. Each of these MPDUs in the aggregated frame can be individually acknowledged using a method called block acknowledgement. This makes MPDU aggregation especially suited for real-time streaming, because an error in one MPDU does not result in the loss of one complete transmit opportunity (TXOP). The lost MPDU can simply be retransmitted together with new data frames as part of the next A-MPDU.

One study on forward error correction did note these improvements. They investigated the use of forward error correction for multicast video data by doing simulations [9]. However, their study is completely theoretical and also makes the false assumption that an A-MPDU can contain as many as 1000 MPDUs. Moreover, 802.11a PHY rates are used in their simulation instead of the 802.11n rates.

Because of the increasing data rate of HD video and the need to transmit more video streams simultaneously, it is important to achieve even higher throughput and lower latency. This research investigates whether the use of a forward error correction code at the link layer level would improve the performance of the link. In order to determine the usefulness of such an error correction code, one important aspect of the wireless link should be carefully examined. This aspect is whether individual errors in an A-MPDU are correlated. A forward error correction code is most valuable when the errors are not correlated, but instead follow a binomial distribution with an independent error probability p. If errors are correlated, there is a higher probability that there are too many errors to correct in an A-MPDU. In addition, the probability that the additional error correction MPDUs are received in error also increases. Therefore, there is little benefit in using forward error correction when errors are strongly correlated, because only a small percentage of errors can be corrected in that case. In that case, the reduction in error probability may not be justified by the overhead of the extra error correction data.

Two research questions are introduced:

- What is the distribution of errors in an A-MPDU and which factors influence this distribution?
- Would the performance of the link increase when a forward error correction code is introduced at the link layer?

The answer to the first question will give insight into the distribution of errors in A-MPDUs. This information can be used to answer the second question.

In order to obtain the information necessary to answer the research questions, several experiments are conducted. Qualcomm 4x4 802.11n devices are used for these experiments, because these devices have full support for aggregation of MPDUs and are therefore well suited for the tests. They use up to four spatial streams to send and receive data enabling them to transmit at a physical layer data rate of up to 600 Mbps.

The setup for the experiment is as follows. One device continuously sends data to another device, while a third device which is placed close to the sender, sniffs all packets. The data is sent without using FEC at the link layer. The experiment is performed for several types of data. The captured data is then analyzed for A-MPDU length and error rate to obtain an answer to the first research question. This data is then used to calculate the improvement in PER if forward error correction would be used.

This paper first presents an introduction to the topic of A-MPDUs in the 802.11n standard and forward error correction codes. After that, the methods for measuring are explained as well as the methods for analyzing the data. Then, the results and a discussion of the results are presented and finally a conclusion is drawn and an answer to the research questions is formulated.

2 Background

2.1 Frame Aggregation in the IEEE 802.11n Standard

Using the full potential of the 802.11n standard, physical layer data rates of 600Mbps can be obtained. Another important improvement is introduced at the link layer. This improvement is the aggregation of data link frames. By combining several link layer frames, the overhead at the link layer is decreased. At the highest data rates, aggregation of MPDUs can lead to a three to four times increase in channel utilization [3]. The MAC-PHY (medium access control-physical) interface combines a maximum of 64 MPDUs into one large aggregated MPDU. The receiver of the packet deaggregates the A-MPDU in its MAC-PHY interface and sends a special type of acknowledgement back, called a block acknowledgement. This block acknowledgement contains a bitmap of 64 bits, in which each bit corresponds to the success of the transmission of one individual MPDU. In the next A-MPDU, the sender retransmits each MPDU which was not received correctly, possibly together with new MPDUs. This is also illustrated in Figure 1.

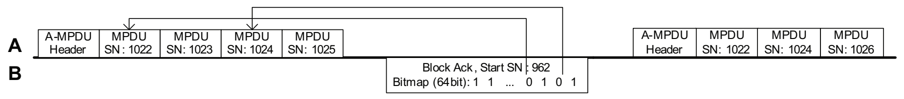

Fig. 1. Simplified flow of transmission of aggregated MPDUs

The most important improvement when using aggregation is the following. In the old approach only one MPDU could be sent after which the sender had to wait before the acknowledgement was received. This process takes one TXOP, so sending ten MPDUs using the old approach takes at least ten TXOPs. In the new approach, these ten MPDUs can be sent in one aggregated MPDU, therefore decreasing the time it takes to transmit the ten MPDUs to only one TXOP.

2.2 Forward Error Correction

Forward error correction has traditionally been used in wireless networks at the physical layer. Here, convolutional codes are used to encode each m-bit information symbol into an n-bit symbol where n is larger than m. Convolution codes are not block codes, but process a continuous bitstream instead. This makes them not suitable for use at the data link layer, because at this layer the individual MPDUs need to be corrected. Normally, the failure of receiving an MPDU leads to a retransmission of that MPDU. However, it is possible to add specific error correction MPDUs to an A-MPDU which are able to recover these lost MPDUs.

These types of forward error correction code are called block erasure codes. In addition, when an error correcting algorithm is such that a message of k information symbols can be encoded into a potentially limitless number of encoding symbols, the code is called a fountain code. One well-known class of fountain codes are raptor codes. The latest generation of raptor codes, RaptorQ [8], are used in this paper when calculations are done on the performance of error correction codes in the measurements. RaptorQ is a systematic code, meaning that the symbols of the original message are included in the set of encoding symbols. Using RaptorQ coding on a source block of size k, the message can be decoded with 99% certainty from the reception of k encoded symbols. For $k + 1$ received symbols, the decoding probability is 99.99% and for $k + 2$ it is 99.999999%.

3 Measurements

3.1 Equipment and Setup

All the measurements were done with Qualcomm 4x4 802.11n wireless devices. These devices use four spatial streams to send and receive data and have full support for aggregation of frames. The setup of the measurements is displayed in Figure 2. In essence, two wireless devices are sending data to each other, while a third one is sniffing all packets. As can be seen in Figure 2, A is connected to the access point with a gigabit ethernet connection. Consequently, when A sends data to B, the data is first sent to the access point and then the data is wirelessly transmitted to B. B wirelessly sends an acknowledgement back which is received by the access point. Because it is a MAC level acknowledgement, it is not propagated all the way back to A.

The sniffer passively participates in the link, capturing all wireless communications between the access point and B. The captured packets are collected on a computer using Wireshark [2]. Because the sniffer should capture as many

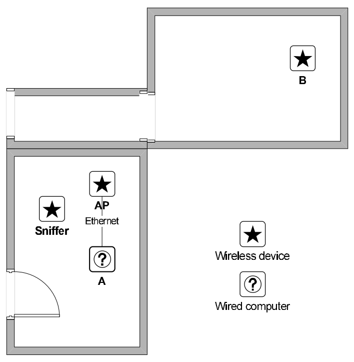

Fig. 2. The setup of the measurements

packets as possible to get accurate results, care has to be taken where to place the sniffer. Data frames are sent at a much higher data rate than (block) acknowledgements. Therefore, acknowledgements have a much larger range than data frames. In the setup of the measurements, the sniffer should be placed such that both data and acknowledgement frames are captured. The best place for this is close to the sender of the data frames.

3.2 Procedure

In order to obtain the data necessary for analyzing, several tests were performed. For all except one test, Iperf was used as the tool to generate the data to transmit [6]. For the remaining test, a high definition video was streamed from *A* to *B*. One requirement for the data set was to be sufficiently large to draw conclusions. There have to be several hundred captured A-MPDUs for each possible A-MPDU size to get meaningful results. After some initial tests using different sizes of data sets, it was concluded that a set of one million packets would be sufficiently large. Therefore, the sniffer was set to capture approximately one million packets per test.

Another important aspect to mention is the variety of the tests. It is important to perform tests with different parameters to obtain a broad overview of the problem. Therefore, tests are performed at several application layer data rates and also using different network layer protocols. The difference between using TCP and UDP as the network protocol is that UDP is one-way communication and TCP is two-way. When using TCP, acknowledgements are also sent at network layer level. These acknowledgements are normal data frames at the link layer. Because the communication is two-way, there is a probability that collisions occur. Therefore, three tests were performed: one using Iperf to transmit UDP data at 60Mbps, one using Iperf to transmit TCP data at the maximum available rate (which was 200Mbps) and one last test where an HD movie was streamed at 15Mbps.

3.3 Information Extraction

To obtain an answer to the research questions, the distribution of errors in an A-MPDU needs to be extracted from the acquired data. A plugin for Wireshark was developed for this purpose. The plugin collects statistics of the data set by having a function which is called once per captured packet.

The easiest way to collect statistics about the error distribution of A-MPDUs is to simply split the data set by the size of A-MPDUs. Then, the probability can be calculated that exactly x MPDUs in an A-MPDU of size s are received in error. This leads to the error distribution per A-MPDU size.

One difficulty in this approach is how to detect whether or not the transmission of a packet was successful. There are two methods to do this. First, it is possible to examine the block acknowledgements transmitted by the receiver of the data A-MPDU. This block acknowledgement contains information about which MPDUs were received. Second, the retransmission flag in an MPDU can be used to detect whether the packet is retransmitted. Logically, when a packet is retransmitted, it must have been sent before, though not received. The second approach is better suited for the analysis, because it is more accurate. Sometimes block acknowledgements are not sent. For example, in the case that one complete A-MPDU fails, the receiver does not receive anything, thus it does not send a block acknowledgement back. Using the first approach, this would be treated as if nothing happened. The second approach does detect such cases.

The approach still relies on the assumption that the sniffer captures all packets. However, this is not the case. Sometimes the sniffer does not receive the packet, though the packet does arrive at the receiver. Also, sometimes the packet is received neither by the sniffer nor by the intended receiver. The first case is not a problem, as the packet reaches the intended receiver so the sniffer does not have to count an error. The second case poses a larger problem. When the retransmit of the lost packets occurs, the sniffer does not know in which A-MPDU the original packets were sent. Therefore, it does not know what the size of the original A-MPDU was and in which error distribution the retransmission should be placed. This problem is illustrated in figure 3. For the analysis, these retransmissions were placed in a special group with unknown A-MPDU size.

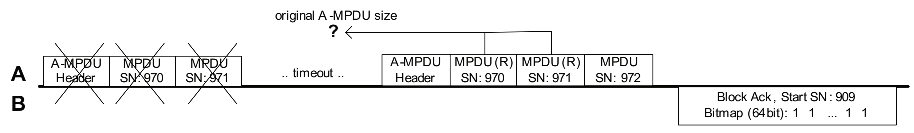

Fig. 3. Analysis difficulty: the original A-MPDU is not received by the sniffer, making it impossible to obtain the size of that original A-MPDU

The data was also split per sender. For example, if TCP data is sent from A to B, the actual data goes from A to B, but TCP acknowledgements are also transmitted from B to A. It is possible that the distribution of errors differs greatly between these two links, so it has to be split in the results.

4 Results

The results of the three experiments are presented in this section. Error distribution plots have been created for each experiment and A-MPDU size. However, because there is a lot of similarity between the figures of different A-MPDU sizes of the same experiment, not all of them are shown. Instead, only a couple of figures which are representative are shown to provide a clear overview of the results.

Each figure contains three lines, two of which show the results of the experiment. They differ in the interpretation of cases where an entire A-MPDU was missed by the sniffer (typically due to a collision). For the 'best-case' line, such events are not counted as errors at all, thus estimating what the distribution would be in the absence of collisions. For the 'worst-case' line, it is assumed that the missed A-MPDU consisted of k MPDUs that were all received in error, with k being the number of retransmissions in the next A-MPDU (obviously, the real size of the missed A-MPDU is not known).

For comparison, the third line shows the theoretical probability that exactly n errors occur, given that at least one error occurs, under the assumption that they follow a binomial distribution (i.e., are independent). The parameter p for this binomial distribution is set to the overall probability that a single MPDU fails in an A-MPDU of that size, as calculated from the measured data for that case.

4.1 UDP 60 Mbps

In Figure 4 the error distribution for A-MPDUs of size 5 can be found. 97% of the A-MPDUs in this measurement had a size between 1 and 11; as the results for these sizes are similar, they are not shown here. Too few A-MPDUs were sent at a size larger than 11 to make those results interesting.

4.2 TCP at Maximum Rate

There was a large spread in the size of A-MPDUs in these results. 93% of the TCP data A-MPDUs were sent at a size between 1 and 30. The TCP acknowledgement A-MPDUs were smaller, but showed similar results. Figure 5 shows the distribution for TCP data A-MPDUs of size 20 and Figure 6 of TCP acknowledgements A-MDPUs of size 10. The results for other A-MPDU sizes are similar.

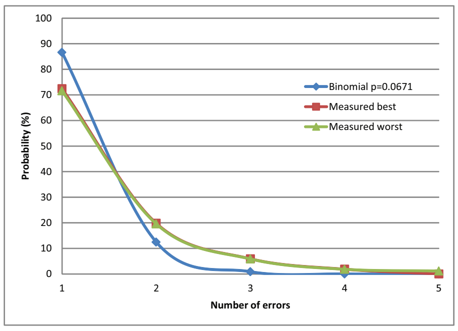

Fig. 4. Error distribution for 60 Mbps UDP traffic and A-MPDU size 5

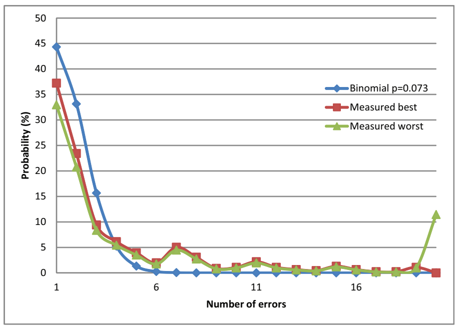

Fig. 5. Error distribution for TCP data traffic at maximum rate and A-MPDU size 20

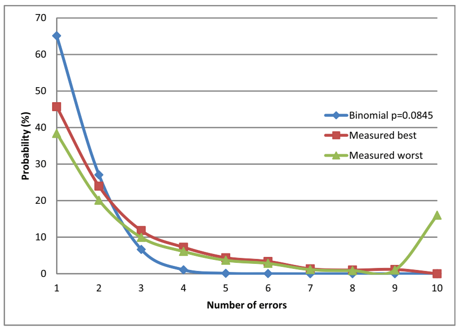

Fig. 6. Error distribution for TCP acknowledgements at maximum rate and A-MPDU size 10

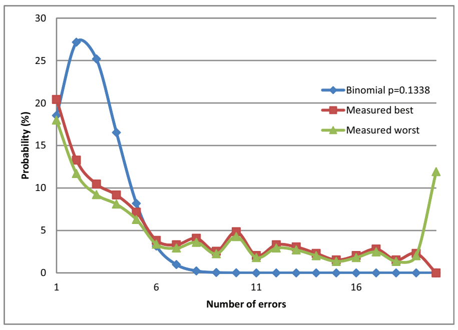

Fig. 7. Error distribution for TCP video data traffic and A-MPDU size 20

4.3 TCP 15 Mbps Video

89% of the A-MPDUs of the TCP video data had a size between 1 and 20. Figure 7 shows the distribution for TCP video A-MPDUs of size 20. Figure 8 shows the individual error probability of a MPDU plotted against the A-MPDU size in this measurement. A similar plot was also obtained for the other experiments, but these are not shown here because of the similarity of the results. Also, Figure 9 shows a plot which illustrates the error probability at a specific location in the A-MPDU. On the horizontal axis the location in the A-MPDU is given. The vertical axis shows the number of errors on that location, relative to the number of errors of the first MPDU in the A-MDPU.

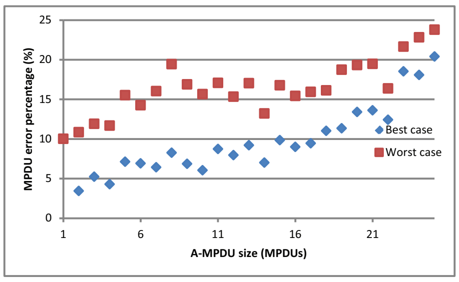

Fig. 8. The error probability of a single MPDU plotted against the A-MPDU size for TCP video traffic at 15 Mbps

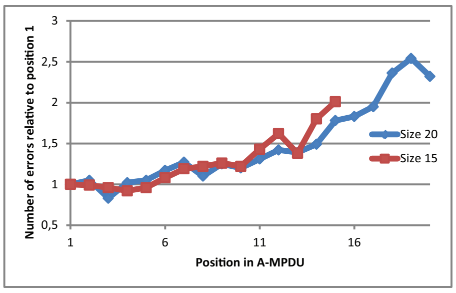

Fig. 9. The number of errors relative to the number of errors on the first place plotted against the location of an MPDU in an A-MPDU

5 Discussion

5.1 Distribution of the Errors

First, the results show a significant difference compared to the theoretical distribution. In almost all cases, the theoretical probability for a single error is larger than the measured probability. Also, at a higher number of MPDU errors in the A-MPDU, the probability is in almost all cases smaller in the theoretical distribution than in the measurements. The theoretical distribution reaches zero quickly, while the measurements show a floor of a few percent for almost all number of errors. This floor could be caused by an error during the training phase of reception. When such an error occurs, the probability of successfully decoding an MPDU decreases significantly, thereby increasing the probability that a large number of MPDUs in the A-MPDU fail.

One clear result is that the error probability of individual MPDUs is strongly related to the size of the A-MPDU in which it was transmitted: MPDUs which are part of a large A-MPDU have a higher error probability than MPDUs in a small A-MPDU.

The UDP traffic measurements follow the theoretical (binomial) distribution more closely than the TCP traffic results, though the curve is still more flat than the binomial distribution. This indicates that even when using one-way traffic like UDP, errors are still slightly correlated. There is almost no difference between the best and worst case scenarios, which indicates that the sniffer captured almost all packets.

The TCP traffic shows larger differences with the theoretical model. The best case, which filters out all completely failed A-MPDUs, is still similar to the best case in UDP traffic. However, the worst case series shows a large peak at the end, indicating that the probability that all MPDUs in an A-MPDU fail increases dramatically when using TCP traffic. A reasonable explanation for this is that the peak is caused by collisions. TCP is two-way communication, whereas UDP is one-way communication. This means that there is a probability that *A*

Table 1. Estimated packet error rates (PER) with various amounts of FEC and data rate reduction.

		Estimated PER								
	Measured	**1 FEC symbol per A-MPDU**			**2 FEC symbols per A-MPDU**			**3 FEC symbols per A-MPDU**		
A-MPDU size	**PER without FEC**	**FEC**	**FEC binom.**	**equiv. rate reduc.**	**FEC**	**FEC binom.**	**equiv. rate reduc.**	**FEC**	**FEC binom.**	**equiv. rate reduc.**
UDP										
5	6.71%	1.82%	0.84%	<0.01%	0.47%	0.05%	<0.01%	0.09%	<0.01%	<0.01%
10	12.35%	6.96%	5.02%	1.24%	3.92%	1.47%	0.01%	2.28%	0.31%	<0.01%
TCP data at max rate										
10	5.13%	2.59%	1.03%	0.52%	1.49%	0.13%	<0.01%	0.92%	0.01%	<0.01%
20	7.30%	5.27%	3.41%	–	3.99%	1.23%	0.73%	3.18%	0.35%	0.73%
30	13.56%	11.56%	10.27%	–	10.05%	7.18%	–	8.73%	4.54%	1.36%
TCP video data										
10	6.04%	3.47%	1.40%	0.6%	2.04%	0.21%	<0.01%	1.25%	0.02%	<0.01%
20	13.38%	11.19%	8.67%	–	9.45%	4.83%	1.34%	7.99%	2.26%	1.34%
25	20.38%	17.79%	16.41%	–	15.53%	12.51%	–	13.52%	8.87%	2.04%

starts sending TCP data exactly at the same time as B starts sending a TCP acknowledgement. This results in a collision where both complete A-MPDUs are lost. These collisions can be avoided by using RTS/CTS [7], which is why the best-case situation should be observed here when determining the usefulness of error correction. In the best case situation in the figures, the collisions are filtered out of the results.

Not only the distribution of the number of errors per A-MPDU differs between reality and the binomial model, but Figure 9 shows that MPDUs transmitted in the latter part of the A-MPDU have a greater probability of failure. A possible explanation for this is that small variations in the channel since the channel measurement in the preamble of the packet lead to greater inaccuracies later in the reception of the A-MPDU.

In general, using TCP at the highest data rate, more larger A-MPDUs are sent than at the lower data rate of TCP video. However, UDP traffic is sent in even smaller A-MPDUs, even though the data rate of the UDP traffic was higher. This is explained by the fact that UDP A-MPDUs can be sent with less waiting time between the packets. When using TCP, the link also has to be utilized for transmitting the acknowledgements, which means the sender has less time to transmit the frames.

5.2 Estimating the Usefulness of FEC

In order to judge the usefulness of forward error correction, the main questions are how much the PER would decrease when FEC is introduced, and how this compares to the reduction in PER when switching to a lower physical layer data rate. We have used the measured error rates and distributions to estimate both of these PER reductions.

The resulting PER when using FEC is computed by using the measured error distributions to calculate how many retransmissions are needed. With n FEC symbols it is possible to repair up to n errors in an A-MPDU. Therefore, when x errors occur, a retransmission is only needed when $x > n$. In that case, $x - n$ MPDUs need to be retransmitted in order to obtain enough data to decode all MPDUs. (The 1% error rate for decoding RaptorQ encoding symbols when $x = n$ is not taken into account here, but this makes only a small difference.) We can also do this computation under the assumption that the errors would be binomially distributed, by simply taking the measured PER as the parameter for the binomial distribution.

The reduction in PER resulting from reducing the physical data rate can be obtained from Figure 5.12 in Perahia and Stacey (2008). E.g., this Figure indicates that for the high data rates in the range of 405 Mbps to 600 Mbps that were used in these measurements, switching to a 10% lower rate would lead to about 10 times smaller PER. The underlying assumption here is that the channel is AWGN. In our tests, this provided a good estimation, because the path did not change during the tests. However, in real situations this may not be true, a fact that needs to be taken into account when drawing conclusions.

The results are shown in Table 1. For several of the A-MPDU sizes occurring in each of our experiments, both the measured PER without FEC is shown, and the calculated PER for 1, 2 and 3 FEC MPDUs per A-MPDU. For comparison, for each of the FEC cases also the calculated PER with FEC under the binomial assumption is shown, and the PER resulting from an equivalent physical data rate reduction. E.g., for 1 FEC symbol per A-MPDU and an A-MPDU size of 10, using FEC would effectively reduce the data rate by 10%, so we compare it to a 10% physical data rate reduction without FEC. This is only possible if such a datarate is available in 802.11n; where this is not the case, the table shows a '–'.

As can be seen, the PER would not benefit much from adding forward error correction: in all cases, FEC performs worse than switching to an equivalent lower physical data rate. Also, it can be seen that at a given PER, the effectiveness of FEC is reduced due to the correlation of the packet errors (i.e., the deviation from the binomial distribution).

6 Conclusion

Several conclusions can be drawn from the results of the measurements in the previous section. First the answer to the first research question on the error distribution in A-MPDUs. It is shown that the distribution of errors in A-MPDUs does not follow the binomial distribution with independent probability p that a single MPDU fails. By comparing the binomial distribution to the distribution as found in the results, two important differences can be observed. In the first place, the results of the measurements lead to the conclusion that there is a correlation between errors in A-MPDUs. Consequently, the presence of one error in an A-MPDU leads to a higher probability of a second error in the same A-MPDU. Therefore, the slope of the distribution function of errors is flatter than that of the binomial distribution function. Secondly, there is a probability that the entire A-MPDU fails due to a collision. This can be observed in the distribution function as a peak at the end.

There are several factors which determine the distribution of errors as can be observed in the results of the measurements. The most obvious factor is the size of the A-MPDU itself. There is a higher probability that larger A-MPDUs contain errors. Also, the probability that a single MPDU fails increases with A-MPDU size. Another important factor is the existence of another device transmitting on the same frequency, thus also whether or not the link is bidirectional. The probability that a collision occurs is much higher in such links, which can be noticed in the distribution function by a higher peak for complete A-MPDU failure. Furthermore, the application layer data rate does not have much impact on the error rate of the MPDUs, because this does not change the physical layer data rate at which the packets are sent.

For the second research question on the usefulness of forward error correction codes the following can be concluded. The performance gain when using a forward error correction code on the MAC layer depends heavily on the premise that errors on the link are binomially distributed and not correlated. When errors are correlated, there are often too many errors in the packet to correct. The results show that the distribution of errors can not be approximated by the binomial distribution function, because of the correlation of errors. In addition, A-MPDUs are sometimes received completely in error. While the latter can be avoided by using RTS/CTS, the first remains a problem for FEC. Therefore, the performance of raptor coding is worse than suggested by previous studies that assume that MPDUs in an A-MPDU fail indepently [9]. It can be concluded that it is generally better to switch to a lower physical layer data rate than to use raptor coding.

Acknowledgements The authors would like to thank Qualcomm for making their latest 802.11n wireless devices available for this research.

References

1. O. Alay and T. Korakis. An Experimental Study of Packet Loss and Forward Error Correction in Video Multicast over IEEE 802.11b Network. *Proceedings of IEEE CCNC*, 2009.
2. G. Combs. Wireshark. `http://www.wireshark.org`, Dec. 2010.
3. B. Ginzburg and A. Kesselman. Performance analysis of A-MPDU and A-MSDU aggregation in IEEE 802.11n. *Sarnoff Symposium, 2007 IEEE*, pages 1–5, Apr. 2007.
4. I. Haratcherev and J. Taal. Automatic IEEE 802.11 rate control for streaming applications. *WIRELESS COMMUNICATIONS AND MOBILE COMPUTING*, 5:421–437, 2005.
5. IEEE Computer Society. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Draft P802.11-REVmb/D4.0*, June 2010.
6. NLANR/DAST. Iperf. `http://sourceforge.net/projects/iperf`, Dec. 2010.

7. E. Perahia and R. Stacey. *Next Generation Wireless LANs*. Cambridge University Press, Cambridge, United Kingdom, 2008.
8. Qualcomm. Raptorq technical overview. `http://www.qualcomm.com/documents/files/raptorq-technical-overview.pdf`, Dec. 2010.
9. M. Samokhina, K. Moklyuk, S. Choi, and J. Heo. Raptor Code-Based Video Multicast over IEEE 802.11 WLAN. *IEEE APWCS*, 2008.