ET4394 Wireless Networking
Transmit Rate Control
Group ALGG1

Alexios Lyrakis (4735811)
Georgios Giannakaras (4747046)

April 4, 2018

1 Introduction

During this project we had the opportunity to work on a simulated wireless link in Matlab and gain experience on the impact that SNR and MCS (modulation and coding schemes) have on the efficiency of a wireless communication. While the IEEE standard defines the specifications for 802.11 MAC protocol and RF-oriented PHY parameters, it does not define any particular instance of Rate Control Algorithm (RCA) and is open to the device manufacturer to improvise. Our main goal was to implement our own RCA and evaluate it on a large set of parameters.

2 Methodology of Work and Plan of Action

Since the RCA in IEEE standard is an open case for improvisation, we studied the most frequently used and fundamental RCAs and other various proposed algorithms to gain a more detailed insight on the topic. Inspired by some basic ideas of already proposed RCAs we created our own algorithm for transmission rate control.

The proposed algorithm is executed iteratively for every received packet and utilizes the estimated SNR and BER (Bit Error Rate) to evaluate the link quality and to decide with which MCS will the next packet be transmitted. The main functionality of our RCA is based on an adaptive look-up table which contains the thresholds of each MCS value. In order to find these thresholds, we initially conducted experiments in which we set a different constant MCS value (ranging from 1 to 9) for each of them. We resulted in the graph presented in figure 1, in which the impact of the MCS value on the BER - SNR is shown. By taking into consideration that any BER larger than zero results in packet loss we noticed which one is approximately the lowest bound of SNR for each MCS value that guarantees a lossless transmission.

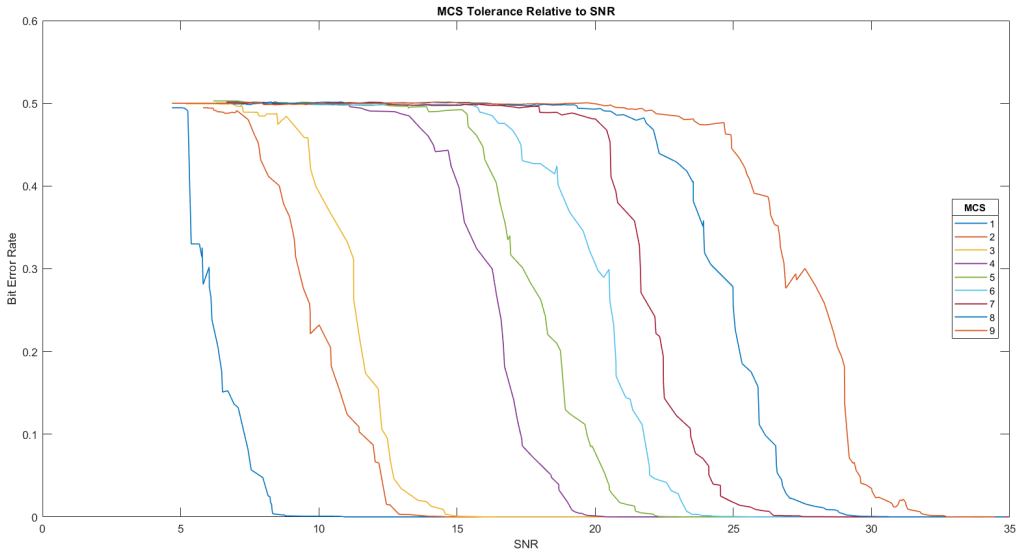


Figure 1: The impact of all possible MCS values to the relationship between Bit Error Rate and SNR. The resulted lines were smoothened by using a moving average.

Based on the above graph we created the look-up table correlating MCS values with SNR thresholds. Subsequently, we conducted more experiments to confirm our observations. The final look-up table is presented in Table 1.

Table 1: Look-up table for MCS and lowest SNR thresholds

MCS	0	1	2	3	4	5	6	7	8	9
SNR Low Threshold	0	10	15	17	21	25	27	29	32	35

Our algorithm's main functionality is as follows: If the $estimatedSNR$, i.e. SNR of the previous transmitted packet, is above a threshold x but below a threshold y of the look-up table, we choose the MCS corresponding to threshold x. Hence, our algorithm is aggressive in terms of achieving high data rates, and is only limited by the thresholds and the BER. However, since our RCA is also lossless transmission oriented, we implemented countermeasures for facing packet losses when the look-up table thresholds can't prevent them. Whenever such a countermeasure is activated the MCS is decreased, no matter what the look-up table threshold indicates. The first technique indicates that if two packets in a row are lost then the MCS is immediately decreased by 1. The second one is activated if we increase the MCS and the transmitted packet is lost. Then, we decrease the MCS back to the previous value. The third countermeasure proposes that if for three received packets in a row the SNR drops and the SNR value of the last packet is greater than the lowest threshold of the current MCS, but very close to it, then we decrease the MCS. With the above countermeasures we have control on how rapidly a throughput might drop and we prevent continuous losses. The final countermeasure is more aggressive and it is activated if in the last eight transmitted packets four or more packets are lost. If such an event takes place the MCS is decreased by one, but also every threshold in the look-up table is increased by 2, making it more difficult to increase the MCS in the subsequent transmissions. Afterwards, an iterative process takes place. Every five new received packets it is checked if the above technique has to be activated again. If it has, then the MCS is decreased again by one and the thresholds in the look-up table are increased again by 2. However, if after five received packets the technique was not activated, then the thresholds in the look-up table are decreased by one. The thresholds are decreased by one every five new received packets until they reach their initial values. With the last countermeasure we aim to avoid packet losses when the communication link is unstable and significant fluctuations in SNR take place. By increasing the thresholds every time this countermeasure is activated, we implement the adaptive behavior of our look-up table and we make our algorithm more robust to rapid SNR changes and to link misbehaviors.

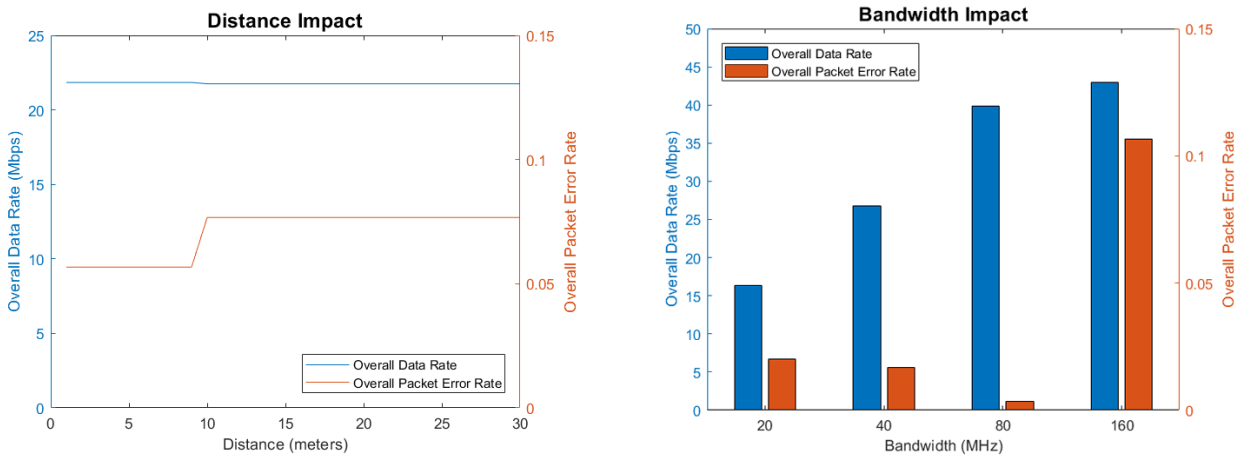
As it is already mentioned we initially studied the rationale behind various RCAs like ARF, RBAR, Onoe, AMRR and SampleRate. Two of our countermeasures are based on ARF. However, we implemented them in a different way and embedded them into our algorithm so that they function efficiently along with our approach.

3 Experimental Results and Conclusions

To evaluate the response and the behavior of our rate control algorithm, we executed a significant number of experiments by varying most of the parameters referring to the waveform, the channel, and the simulation. Initially, we conducted experiments in which we kept all of the parameters constant and changed only one parameter at a time. In this way we were able to observe the impact of the changing variable on the overall data rate and packet error rate, and on our RCA. Subsequently, we executed experiments featuring totally random parameters in all possible combinations so we can gain a more robust insight of our algorithm's efficiency in general. Before we analyze the experiments we have to mention that the overall data rate that we will refer to, is essentially the goodput, as derived from the way it is defined in the Matlab example.

The forthcoming experiments where executed with approximately the same parameters as the default Matlab example and with 300 transmitted packets. In figure 2 (a) is depicted the result of an experiment in which distance was varying. We expected that as the distance increases we would have significant drops in overall data rate and growth in overall PER. However, we notice this behavior only in one point (at 9 meters). We repeated this experiment many times with various configurations and we had approximately the same results, so this might be an issue of the way the simulation handles the distance. In figure 2 (b) is depicted the outcome of an experiment

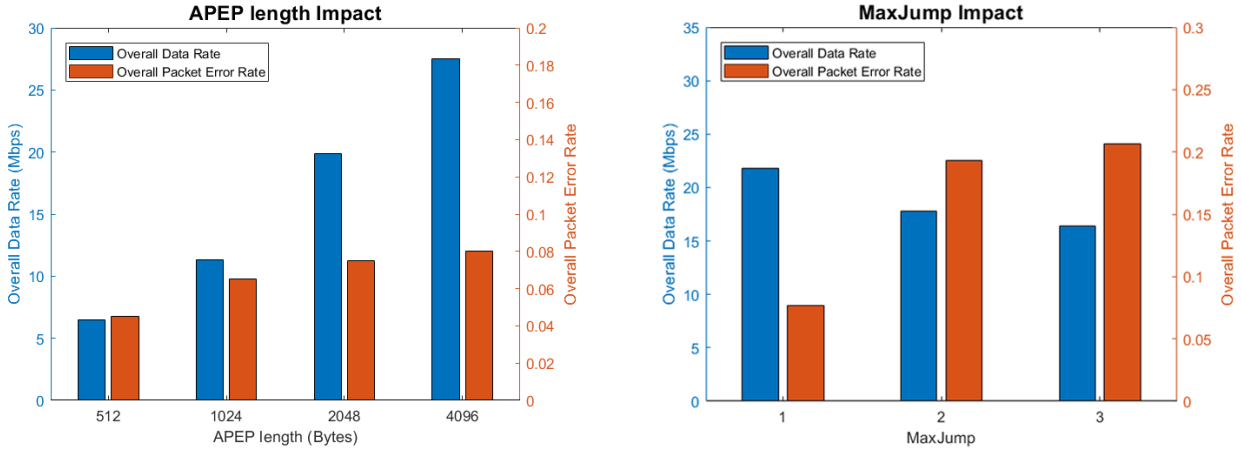
in which bandwidth was varying. As we know higher channel bandwidths can support a higher data rate and more bandwidth. On the other hand as the bandwidth increases there might be congestion and channel interference. Indeed in figure 2 (b) we see that overall data rate increases along with bandwidth and PER has its higher value at 160 MHz.



(a) (b)

Figure 2: (a) The impact of distance, between the transmitter and the receiver, on overall data rate and packet error rate. (b) The impact of bandwidth on overall data rate and packet error rate.

Another experiment that we conducted regards the APEP length. With a constant number of transmitted packets if we raise the frame length it is reasonable to achieve higher overall data rates, as the transmitted data will be more. In addition, the overall PER will grow too, as larger frames are more prompt to bit errors, resulting in more packet losses. These facts are also depicted in our experiment's outcome in figure 3 (a). Another parameter that affects the SNR and that we modified is the maxJump. The maxJump controls the maximum SNR difference between one packet and the next. The greater the maxJump is, the more fluctuations in SNR will take place and the quality of the channel will become unstable. Indeed, as the figure 3 (b) depicts overall PER increases along with maxJump and accordingly overall data rate decreases.



(a) (b)

Figure 3: (a) The impact of frame length on overall data rate and packet error rate. (b) The impact of maxJump parameter on overall data rate and packet error rate.

In figure 4 the behavior of our RCA is presented in an experiment with random parameters and 600 transmitted packets. We observe that the SNR fluctuates a lot between the transmitted packets and also throughout the whole experiment. However, our RCA adapts to these rapid changes and manages to retain a high throughput and only some packets are lost. It is worth

mentioning that the MCS behavior is proportional to the one of estimated SNR, denoting that the implemented RCA adapts quickly into various changes of channel's conditions.

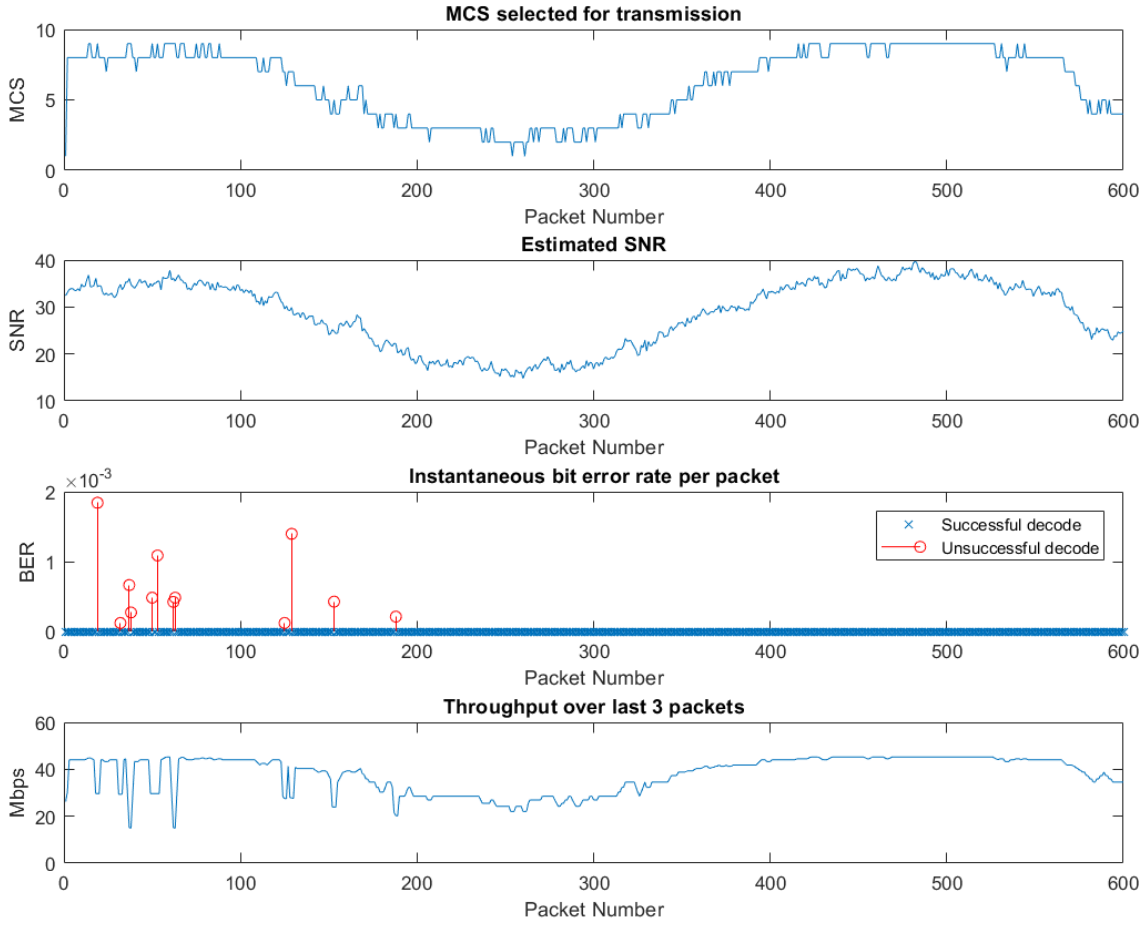


Figure 4: Behavior of our implemented RCA in an experiment with random parameters and 600 transmitted packets.

While we were developing the RCA and to estimate its behavior we executed a lot of experiments with various parameters and different combinations. To further evaluate it, once our algorithm was complete, we conducted 250 different experiments with randomly changing parameters to obtain some actual metrics about the overall data rate and PER. With exactly the same parameters we executed also the same experiments but with Matlab's default RCA, aiming to compare the performance of the two algorithms. The parameters that were continuously modified to their whole reasonable ranges are: Waveform and Channel Bandwidths, APEP Length, Channel Type, Delay Profile, Distance, Seed for Channel's Random Stream, Number of Packets, Mean SNR Value, Amplitude of SNR, and MaxJump. The results for both RCAs are denoted in table 2. Our implemented algorithm resulted to have a better performance, as it has a greater average overall data rate and a smaller average overall PER. It is worth mentioning that while we were conducting experiments we faced a case were in both RCAs the overall data rate was zero and the PER was one. This means that all the transmitted packets failed. In these cases we noticed the same following parameters: BW = 160 MHz, APEP length = 512 bytes, Channel Type = wlanTGacChannel, Delay Profile = Model-F. These cases are excluded from table 2 to not hide valuable information.

Table 2: Overall results of 250 experiments of our RCA and Matlab default RCA

RCA	Min Overall Data Rate	Max Overall Data Rate	Average Overall Data Rate	Min Overall PER	Max Overall PER	Average Overall PER
Implemented	1.1328	52.8248	16.3107	0	0.9602	0.1125
Matlab Default	0.9651	52.1702	15.7455	0	0.9652	0.1337