Performance Evaluation of an Adaptive Error Control Protocol in Wireless Networks

Neda Nikaein, Christian Bonnet
Institut Eurécom
2229, Route des Crêtes
B.P. 193
06904 Sophia Antipolis, France
{nikaein, bonnet}@eurecom.fr

Abstract – Wireless channels are highly affected by unpredictable factors such as cochannel interference, adjacent channel interference, propagation path loss, shadowing and multipath fading. The unreliability of media degrades the transmission quality seriously. Automatic Repeat reQuest (ARQ) and Forward Error Correction (FEC) schemes are frequently used in wireless environments to reduce the high bit error rate of the channel. In this paper, we propose an adaptive error control scheme for wireless networks based on dynamic variation of error control strategy as a function of the channel bit error rate, desired QoS and number of receivers. Reed-Solomon erasure codes are used throughout this study because of their appropriate characteristics in terms of powerful coding and implementation simplicity. Simulation results show that our adaptive error control protocol decreases the waste of bandwidth due to retransmissions or extra coding overheads while satisfying the QoS requirements of the receivers.
keywords: QoS, adaptive error control, ARQ, FEC, Markov model, wireless networks.

1 Introduction

Recently, the emergence of new multimedia applications has created a strong need for the support of *Quality of Service* (QoS). In response, the Internet is moving from a best effort model to a system, capable of supporting a range of traffic characteristics and service requirements. The main obstacle in order to enable users to have access to Internet and multimedia applications in wireless environments is the high error rate of the wireless channels. In fact, wireless channels are highly affected by unpredictable factors such as cochannel interference, adjacent channel interference, propagation path loss, shadowing and multipath fading. As a result, most of the wireless systems are equipped with a complementary error control protocol at the link layer.

Basically, there are two main error recovery mechanisms: Automatic Repeat Request (ARQ) and Forward Error Correction (FEC). ARQ tries to retransmit the lost packets while FEC transmits some redundant data with the original ones. FEC is frequently used in wireless environments but it can not assure full reliability unless coupled with ARQ. In this paper, we study the use of different error control protocols as a function of the QoS requirements of receivers as well as the wireless channel conditions.

We take a multicast communication mode. If we consider multicast communication as a general communication mode where the traffic is sent to a set of receivers, unicast and broadcast communications can be viewed as special cases where the traffic is only sent to one receiver or to all receivers respectively. Having a framework for QoS provisioning in the case of multicast communications means that the same general rule can be applied to any communication mode. Note that we suppose a single QoS per multicast session. It means that all members of a given group are supposed to have the same QoS requirements. Other approaches like layered multicast [1] may be used in the case of receivers with different QoS needs but this is not the subject of this study.

[2] showed that the use of hybrid ARQ/FEC protocol improves the performance of error control schemes for wireless links in most cases. However, the choice of the coding scheme depends on several parameters. A high degradation of the channel bit error rate may cause a high retransmission rate. On the other hand, even in good channel conditions, the retransmission rate increases enormously if there is a high number of receivers in a session. Hence, choosing a fixed coding scheme may cause the waste of bandwidth during the normal behavior of the channel since the redundant information is not required due to the low bit error rate of the channel. On the other hand, during the temporary degradation of the network, the amount of redundancy may not be sufficient for receivers to recover from transmission errors. Even with good channel conditions,

if there is a high number of receivers, the redundancy level of a code may not be sufficient. Therefore, the use of adaptive coding schemes for wireless channels is an issue that has to be studied thoroughly.

An adaptive algorithm needs to estimate the channel conditions of all receivers listening to the same session in order to adjust its parameters dynamically based on an optimization criteria. Adaptive schemes have already been proposed in different contexts. It has been proposed for real-time applications in order to cope with retransmission delays in Internet [3] [4] [5] as well as in wireless networks [6] [7] [8] [9]. It has also been proposed for multicast communications [10] [11]. We observe that all the adaptive coding schemes designed for multicast communications are based on a fixed environment. The other works have considered a wireless network but their adaptation scheme is designed for a point-to-point communication mode. Our proposed approach is different from other adaptive algorithms since it is capable to adapt itself not only to the channel conditions but also to the number of receivers. It is based on a predictive mechanism in the sense that it forwards a certain number of redundant packets in the network before their necessity. It attempts to decrease the used bandwidth as much as possible while maintaining the desired QoS parameters.

We take a finite state Markov chain in order to model the radio channel. The advantage of such a model lies on its facility to capture the burstiness of the error process as well as to predict the future states of the channel based on its present state. Prediction is useful due to the memory that exists in the physical channel. Our proposed scheme tries to take advantage of the channel memory in order to obtain better performance. We use *Reed-Solomon Erasure* (RSE) codes because of their appropriate characteristics in terms of powerful coding and implementation simplicity.

The paper is organized as follows. We start by some background information about coding and Reed-Solomon Erasure codes. We explain the QoS metrics that we have taken in order to analyze the effect of our adaptive scheme. Then, we present the finite state Markov model used in this paper. Our proposed prediction method as well as our adaptation policy are presented afterwards. Finally, we illustrate some simulation results comparing the performance of our adaptive error control protocol with other protocols.

2 Coding Aspects

Coding consists of adding redundant information to data in order to allow the receiver to recover the original data even in the presence of transmission errors. Basically, a code transforms a *data block* of k symbols $d = (d_{k-1}, d_{k-2}, ..., d_0)$ into a *coded block* of n symbols $C = (c_{n-1}, c_{n-2}, ..., c_0)$. In a system that uses FEC for error control, the sender and the receiver use a mutually agreed code to protect the data. If a coded block can be divided into the data part and the redundancy part, then the code is said to be a *systematic code*. A systematic code generates a coded block consisting of an unaltered copy of the data block followed by the $h = n - k$ redundant symbols. The advantage of a systematic code is that in case a receiver receives the data block correctly, no decoding is necessary.

A Reed-Solomon erasure code is a Reed-Solomon code with symbols defined over the *Galois Field* $GF(2^m)$, designed to recover from erasures. It is represented as $RSE(n, k)$ and it has a symbol size of m bits. A Reed-Solomon erasure code has the capacity to recover from h erasures with only h redundant symbols. This characteristic makes this kind of code particularly powerful to cope with transmission packet losses. The parameters of such a code are:

Number of symbols in a coded block:	$n = 2^m - 1$,
Number of redundant symbols:	$h = n - k$,

In the sender side, the RSE encoder takes k data packets and generates h redundant packets to form a coded block of $n = k + h$ packets. All the packets have a size of m bits. If the receiver gets at least k packets out of the $k + h$ transmitted packets correctly, it can reconstruct the original data. Note that the loss unit is a packet and a packet payload is considered as a symbol.

2.1 Implementation Issues

McAuley proposed a hardware architecture for RSE codes in [12] using a symbol size $m = 8$ and $m = 32$. Rizzo proposed a software implementation of RSE codes in [13] with a symbol size in the range of $m = 4, ..., 16$. RSE coders with large symbol size are difficult to implement. Normally, the packet size is in the order of hundreds or thousands of bits. In this case, we need to consider a packet as l symbols of m bits and the coding can be implemented using l parallel RSE coders, each operating on a symbol size of m bits.

Since the number of elements of the $GF(2^m)$ with a symbol size of m is limited to 2^m, it is important to choose an RSE code with $n < 2^m$. If we take $m = 8$, we will have a maximum block length $n = 255$ which is sufficient in our case.

In the following, we use the software RSE coder developed by Rizzo in the systematic form with a symbol size $m = 8$. The encoding and decoding speeds of this software coder have been tested in various platforms from high speed workstations to small portable systems [13] [14] and have been shown to be in the order of Mega Bytes per second.

In order to have variable error correcting capabilities, we are interested to modify the coding parameters k and h of an RSE code. This is feasible by using *shortening* and *puncturing* techniques [15]. Shortening consists of adding a certain number of information symbols equal to zero to the original information in the encoding phase. Let's consider a Reed Solomon erasure code of $RSE(n, k)$. We can generate a set of shortened code $RSE(n-b, k-b)$ with $1 \leq b \leq k-1$ and an error correcting capability, h', equal to h. These shortened codes have their b high order information symbols equal to zero. Code puncturing involves not transmitting (deleting) certain redundant symbols. Puncturing allows a coder to change its number of redundant packets h while shortening allows it to change its number of data packets k. The shortened and punctured codes can use the same encoder/decoder pair as their original code.

We consider an *original code* $RSE(N_{max}, K_{max})$ with $H_{max} = N_{max} - K_{max}$. Using the shortening technique, we can derive a *basic code* $RSE(N, K)$ with the same number of redundant packets $H = H_{max} = N - K$. From this basic code, we can create a large set of RSE codes $RSE(n, k)$ with $k \leq K$ and $h \leq H$ using the shortening and puncturing techniques. The software coder proposed by Rizzo can be easily extended to support multiple block sizes and multiple redundant packets as in [14]. The only implication of such a coder is that it needs to support the maximum data block size K_{max} which is normally bigger than the actual data block size k. However, taking the maximum data block size allows us to use a single generator matrix that can support up to K_{max} data packets which is important if we need to vary our coding parameters.

3 QoS Metrics

We take *efficiency* and *packet loss rate* as our QoS metrics. The first metric considers the effect of our adaptive scheme on bandwidth and gives us an evaluation of how much overhead our scheme adds compared to other schemes. It is defined as the inverse of the average number of transmissions required by all receivers to receive a packet correctly.

The second metric evaluates the loss probability before and after our adaptive scheme. It is defined as the probability that at least one receiver can not receive a packet correctly after the first transmission. This metric allows us to observe the decrease of loss rate due to the utilization of the adaptive protocol. It gives us a precise measure of the effectiveness of our protocol in reducing the loss rate.

Throughout this paper, we suppose that the loss events at different receivers are independent. We assume that all bit errors in a received packet are detected thanks to its CRC field and no control messages are lost. In case of multicast, the traffic is transmitted to all receivers using the broadcast mechanism of the radio rather than sending a separate copy for each receiver.

4 Finite State Markov Model

Markov chains have been extensively used in the literature to capture the bursty nature of the error sequences generated by a wireless channel. Previous studies [16], [17] show that a first order Markov chain provides a good approximation of the error process in fading channels. Furthermore, the parameters of the model can be easily mapped to real physical quantities in case of a Rayleigh fading channel. We take a finite state Markov model as in [18]. This model is depicted in Figure 1. As it can be seen, the channel states associated with consecutive symbols are assumed to be neighboring states. This assumption is true for a slow fading channel where the SNR varies slowly compared to the symbol interval T.

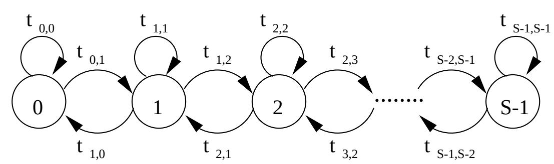


Figure 1: Finite state Markov model

Let $0 = \lambda_0 < \lambda_1 < \lambda_2 < ... < \lambda_S = \infty$ be the thresholds of the received SNR. The channel is said to be in state s where $s \in \{0, 1, 2, ..., S-1\}$ if the received SNR is in the interval $[\lambda_s, \lambda_{s+1})$. Associated with each state, there is a Binary Symmetric Channel (BSC) with the error probability e_s.

Assuming that the channel fades slowly with respect to the symbol interval, T, the Markov transition probabilities can be approximated using the level crossing rate and the SNR density function. Recall that Rayleigh fading results in an exponentially distributed distortion of the signal [19].

$$t_{s,s+1} \approx \frac{1}{\pi_s} exp(-\frac{\lambda_{s+1}}{\bar{\lambda}}) f_d T \sqrt{\frac{2\pi\lambda_{s+1}}{\bar{\lambda}}} \quad (1)$$

$$t_{s,s-1} \approx \frac{1}{\pi_s} exp(-\frac{\lambda_s}{\bar{\lambda}}) f_d T \sqrt{\frac{2\pi\lambda_s}{\bar{\lambda}}} \quad (2)$$

$$t_{s,s} = 1 - t_{s,s-1} - t_{s,s+1} \quad (3)$$

$$t_{0,0} = 1 - t_{0,1} \quad (4)$$

$$t_{S-1,S-1} = 1 - t_{S-1,S-2} \quad (5)$$

In the above expressions, $\bar{\lambda}$ is the average SNR and f_d is the maximum doppler frequency given by $f_d = \frac{v f_c}{c}$ with v the vehicle speed, f_c the carrier frequency and c the speed of light $(3 \times 10^8 m/s)$. The steady state probabilities, π_s, are:

$$\pi_s = \int_{\lambda_s}^{\lambda_{s+1}} f(\lambda)d\lambda = exp(\frac{-\lambda_s}{\bar{\lambda}}) - exp(\frac{-\lambda_{s+1}}{\bar{\lambda}}) \tag{6}$$

The error probabilities of each state e_s can be related to the received SNR according to the modulation scheme used in the system.

$$e_s = \frac{\int_{\lambda_s}^{\lambda s+1} f(\lambda)e_m(\lambda)d\lambda}{\int_{\lambda_s}^{\lambda s+1} f(\lambda)d\lambda} = \frac{1}{\pi_s}\int_{\lambda_s}^{\lambda s+1} f(\lambda)e_m(\lambda)d\lambda \tag{7}$$

where $e_m(\lambda)$ is the modulation function relating bit error probability to received SNR. Simplified expression for e_s is provided in [18] for a BPSK scheme. The average error rate of the model can be found as $e = \sum_{s=0}^{S-1} \pi_s e_s$.

5 Prediction Method

We take two different error control protocols. In the first protocol, P1, we use an ARQ mechanism. The second protocol, P2, uses an ARQ/FEC scheme with RSE codes. In order to have a fair comparison of P1 and P2, we consider that both protocols send k data packets before waiting for a feedback.

We consider a finite state Markov model as described in the previous section. In order to investigate the effect of packet level FEC, we are interested to model the process of successful or unsuccessful packet transmission. [20] showed that a Markov approximation for a packet loss process is a good model for a broad range of parameters. In fact for typical data rates (e.g. more than 64 Kb/s) and for environments commonly considered (e.g. carrier frequency of about 1-2 GHZ and typical pedestrian and vehicular speeds), we can assume that the channel is constant during a packet interval T. With this assumption, the packet loss probability of each state, p_s, can be calculated as in a BSC model with the error probability e_s. For a packet of length L bits, we have:

$$p_s = 1 - (1 - e_s)^L \tag{8}$$

5.1 Efficiency

Let us first consider the scenario P1 where a sender multicasts data to R receivers using an ARQ scheme. The sender retransmits the original packet if there is at least one receiver that has not received the packet correctly. The sender sends k packets at a time before waiting for a feedback. Generally the sender is not aware of a packet loss unless it receives a negative feedback from one of the receivers. In this case, it can only retransmit the lost packet after the retransmission delay. We assume that the channel remains at the same state during the time spanning the end of the transmission of a block and the beginning of the transmission of the next block. It is clear that if this time interval is longer than the correlation time of the channel, the assumption that the channel stays at the same state is not correct. We define L_r as the number of times that a packet gets lost by a receiver. We assume that the state transitions occur at the beginning of a time slot of unit length and then a packet is transmitted. The probability that a receiver loses a packet exactly l times is:

$$P(L_r = l) = \sum_{s=0}^{S-1} P_s(L_r = l), \tag{9}$$

$$P_s(L_r = l) = \begin{cases} \sum_{i=0}^{k-1} \Big[P_s(i, k-1)t_{s,s}(1-p_s) \\ +P_{s-1}(i, k-1)t_{s-1,s}(1-p_s) \\ +P_{s+1}(i, k-1)t_{s+1,s}(1-p_s) \Big] & l = 0 \\ \sum_{i=0}^{k-1} \Big[P_s(i, k-1)t_{s,s}p_s \\ +P_{s-1}(i, k-1)t_{s-1,s}p_s \\ +P_{s+1}(i, k-1)t_{s+1,s}p_s \Big] & l = 1, \ldots \end{cases}$$

$P_s(L_r = l)$ is the probability that a receiver loses a packet l times with the channel ending in state s. $P_s(i, k-1)$ represents the probability to have i packet losses in $k-1$ packet transmissions with the channel ending in state s. [21] calculated the probability to have i errors in j transmissions in a Gilbert-Elliot model using recursion. Using the same approach, we can calculate the probability to have i packet losses among j transmitted packets, $P(i, j)$, in a finite state Markov chain. Let $P_s(i, j)$ be the probability to have i packet losses among j transmitted packets with the channel ending in state s. As before, we assume that state transitions occur at the beginning of a time slot of unit length and then a packet is transmitted. Extending the equation from a Gilbert-Elliot model to a finite state Markov model, the probability to have i packet losses in j packet transmissions is:

$$P(i, j) = \sum_{s=0}^{S-1} P_s(i, j) \tag{10}$$

$$\begin{aligned}P_s(i,j) =&P_s(i,j-1)t_{s,s}(1-p_s)\\&+P_{s-1}(i,j-1)t_{s-1,s}(1-p_s)\\&+P_{s+1}(i,j-1)t_{s+1,s}(1-p_s)\\&+P_s(i-1,j-1)t_{s,s}p_s\\&+P_{s-1}(i-1,j-1)t_{s-1,s}p_s\\&+P_{s+1}(i-1,j-1)t_{s+1,s}\end{aligned}$$

for $i = 0, 1, 2, ..., s$ and $j = 1, 2, 3...$

In order for our adaptive algorithm to change its strategy dynamically, it must be able to predict the performance of each of the available error control schemes for the next block before actually transmitting it. We assume that the adaptive algorithm is informed about the channel state of all the receivers at the beginning of the transmission of each block. Once the channel state of all receivers at instant t is known, the algorithm can predict the evolution of channel conditions of the receivers for the next block taking advantage of the fact that the future states of the Markov chain depends only on its present state. Assuming that a receiver is in state s' at the beginning of the transmission, the initial conditions for $P_s(i,j)$ in equation (9) are:

$$P_s(0,0) = \begin{cases}1 \quad \text{if} \quad s=s' & l=0,1\\ 0 \quad \text{otherwise} & l=0,1\\ P_s(L_r=l-1) & l=2,...\end{cases}$$

Using the above initial conditions, we get S different values for $P(i,j)$ and $P(L_r = l)$ depending on the state where the receiver was at the beginning of the transmission. We represent these probabilities by $P(L_r = l|s')$ and $P(i,j|s')$ where s' is the state of a receiver at the beginning of the transmission. We represent the number of receivers in each of the states of the Markov chain by $\{r_0, r_1, .., r_{S-1}\}$ and the total number of receivers as R. It is clear that we have $\sum_{s=0}^{S-1} r_s = R$. The algorithm estimates the efficiency of P1 for R receivers as follows:

$$\begin{aligned}Eff &= \frac{1}{E[M]}\\ &= \frac{1}{\sum_{m=1}^{\infty}\left(1-\prod_{s'=0}^{S-1}(1-P(L_r=m-1|s'))^{r_{s'}}\right)}\end{aligned} \tag{11}$$

Now, we consider protocol P2 where the sender uses an RSE code with a coded block size of n packets containing k original packets and h redundant packets. In this case, the sender sends k original packets followed by h redundant ones. Each receiver can recover from loss if it receives correctly k packets out of the $n = k + h$ transmitted packets. If the receiver can not recover from loss, it asks for a retransmission.

We define $Q(L_r = l)$ as the probability that a receiver loses a packet exactly l times in the case of FEC. $Q(L_r = l)$ is again the sum of $Q_s(L_r = l)$, the probability of a receiver to lose a packet exactly l times with the channel ending in state s. In the presence of FEC, a packet is retransmitted if it is lost by the FEC receiver and if more than $h-1$ out of the other $n-1$ packets of the coded block are lost. In the same way, a packet is considered to be correctly received if it has not been lost or if it has been lost but there are at least $h-1$ packets out of the other $n-1$ packets of the coded block that have been correctly received. Once again, we assume that the channel does not change its state during the interval $T+t$ where t corresponds to the time between the end of the transmission of the last packet of a coded block and the beginning of the transmission of the first packet of the next coded block.

$$Q(L_r = l) = \sum_{s=0}^{S-1} Q_s(L_r = l), \tag{12}$$

$$Q_s(L_r=l) = \begin{cases}\sum_{i=0}^{h-1}\Big[P_s(i,n-1)t_{s,s}p_s\\ +P_{s-1}(i,n-1)t_{s-1,s}p_s\\ +P_{s+1}(i,n-1)t_{s+1,s}p_s\Big]+\\ \sum_{i=0}^{n-1}\Big[P_s(i,n-1)t_{s,s}(1-p_s)\\ +P_{s-1}(i,n-1)t_{s-1,s}(1-p_s)\\ +P_{s+1}(i,n-1)t_{s+1,s}(1-p_s)\Big] & l=0\\ \sum_{i=h}^{n-1}\Big[P_s(i,n-1)t_{s,s}p_s\\ +P_{s-1}(i,n-1)t_{s-1,s}p_s\\ +P_{s+1}(i,n-1)t_{s+1,s}p_s\Big] & l=1,...\end{cases}$$

In order to estimate the efficiency of protocol P2, the adaptive algorithm needs to estimate $Q(L_r = l)$ first. Assuming that a receiver is in state s' at the beginning of a transmission, the initial conditions for $P_s(i,j)$ in equation (12) are:

$$P_s(0,0) = \begin{cases}1 \quad \text{if} \quad s=s' & l=0,1\\ 0 \quad \text{otherwise} & l=0,1\\ Q_s(L_r=l-1) & l=2,...\end{cases}$$

Note that once again, we have different $Q(L_r = l)$ probabilities depending on the channel state at the beginning of the transmission. We represent these probabilities by $Q(L_r = l|s')$ where s' is the channel state of a receiver at the beginning of the transmission. The algorithm predicts the efficiency of protocol P2 as follows:

$$\begin{aligned} Eff &= \frac{1}{E[M]} \qquad (13) \\ &= \frac{k}{n} \frac{1}{\sum_{m=1}^{\infty} \left(1 - \prod_{s'=0}^{S-1} (1 - Q(L_r = m-1|s'))^{r_{s'}}\right)} \end{aligned}$$

Note that in all the above formulas we have $t_{s-1,s} = 0$ for $s = 0$ and $t_{s,s+1} = 0$ for $s = S - 1$.

5.2 Packet Loss Rate

The next QoS metric is the packet loss rate which is the probability to have at least one receiver that has not received a packet correctly after the first transmission. Considering protocol P1, the probability to receive a packet correctly after the first transmission is $P(L_r = 0)$. Once again, we define r_s as the number of receivers in state s. $P(L_r = 0|s')$ is the probability of a receiver to receive a packet correctly after the first transmission with the receiver being in state s' at the beginning of the transmission. The packet loss rate of protocol P1 is estimated as follows:

$$PLR = 1 - \prod_{s'=0}^{S-1} \Big[P(L_r = 0|s')\Big]^{r_{s'}} \qquad (14)$$

In case of protocol P2, the probability that a receiver gets a packet correctly after the first transmission is $Q(L_r = 0)$. We represent this probability with the receiver being in state s' at the beginning of the transmission by $Q(L_r = l|s')$. The adaptive algorithm estimates the packet loss rate of protocol P2 as below:

$$PLR = 1 - \prod_{s'=0}^{S-1} \Big[Q(L_r = 0|s')\Big]^{r_{s'}} \qquad (15)$$

6 Adaptation Policy

Let $C = \{c_0, c_1, ..., c_k\}$ be the set of RSE codes available at the sender. The sender can either choose the ARQ/FEC error control protocol with an RSE code in C or a pure ARQ protocol. According to the variations of SNR, the receiver channel may be in one of the states of the Markov model at each instant t. We assume that the sender knows the state of the Markov chain at the transmission time for all receivers. Let's define the *transmission status* at time t as the set of all tuples (s, r_s) where $s \in \{0, 1, ..., S-1\}$ is the channel state in the Markov model and r_s is the number of wireless receivers in state s at time t.

Before transmitting, the adaptive algorithm in the sender must estimate the efficiency and packet loss rate of the ARQ/FEC protocol using all the available coding schemes as well as the ARQ protocol as a function of the transmission status. It then tries to find the protocol satisfying the desired packet loss rate. If there are several protocols satisfying this criteria, the algorithm must choose the one with the highest efficiency in order to minimize the use of bandwidth. Note that our adaptive approach is predictive rather than reactive since the sender tries to predict the channel conditions as well as the evolution of QoS metrics for all receivers before actually sending a block. The sender then chooses a protocol according to its predictions.

The time is divided into transmission rounds. Each transmission round corresponds to the transmission of n packets in case of FEC and k packets in case of ARQ. A transmission round ends when the sender is informed about the reception states of all receivers. The adaptive algorithm is repeated at the end of each transmission round. Basically, the algorithm goes through the following steps:

1. At the beginning of the algorithm, the sender determines the desired packet loss rate of the session. It also determines the transmission status.

2. The sender estimates the packet loss rate of the ARQ protocol as well as the ARQ/FEC protocol using all the available coding schemes, based on the transmission status. If it finds several protocols satisfying the QoS metrics of the session, it chooses the one with the highest efficiency. It then adjusts its parameters and starts the transmission of the block.

3. At the end of a transmission round, the sender again determines the transmission status. It then repeats the step 2.

7 Simulation Results

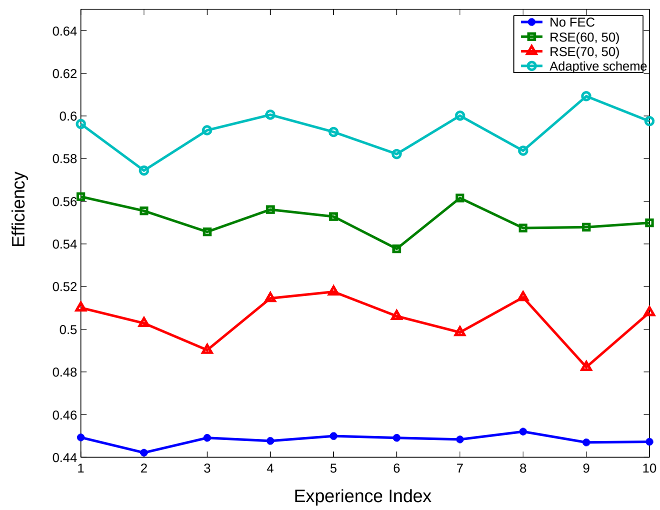
We have carried out several simulations in OPNET which is an event-driven simulation tool. We take a 20 Mb/s data rate for our wireless network. The carrier frequency is 5.2 GHz. The data and control packets have 54 and 9 bytes respectively. We use a BPSK modulation scheme. The average SNR is 34 dB corresponding to an average bit error probability of 10^{-4}. All the receivers are located within a distance of 23 meters from their base station. The wireless channel is modeled by a 3 state Markov model in the OPNET environment. The state s_0 of the Markov model corresponds to a Bad state with $e_0 \approx 1$, the state s_1 corresponds to an intermediate state with a non-zero error probability $e_1 \approx 2 \times 10^{-5}$ and the state s_2 corresponds to a Good state with a zero error probability $e_2 \approx 0$. In order to have an error probability of $e_0 \approx 1$, λ_1 must be equal to 2dB in BPSK. For a zero error probability $e_2 \approx 0$ in state s_2, we also need to fix λ_2 at 34dB in BPSK. Knowing the threshold values of the Markov model, all the other parameters can be easily found as in Section 4. For our adaptive scheme, we take an original code $RSE(255, 235)$ and a basic code $RSE(70, 50)$. Using this basic code, we can vary the coding parameters such that for any used code $RSE(n, k)$, we have $k \leq 50$ and $h \leq 20$.

For each scenario, we have carried out ten different simulations, each with a different seed. Figure 2 compares the efficiency and the packet loss rate of our proposed adaptive scheme with a pure ARQ protocol, an ARQ/FEC protocol using $RSE(60, 50)$ and another hybrid protocol using $RSE(70, 50)$. The number of receivers is fixed at 1000. We have chosen a $PLR = 50\%$ in order to reduce the retransmission rate by a half in case of adaptive scheme. From this figure, we can observe that the adaptive scheme provides the best efficiency. It also has a PLR less than 50% as it was expected. Although other fixed hybrid protocols provide better packet loss rates, they have a lower efficiency compared to our adaptive scheme.

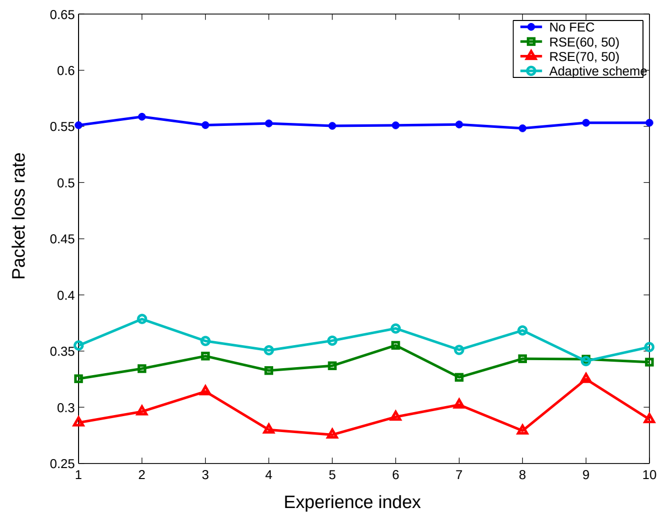
8 Conclusion

In this paper, we proposed an adaptive algorithm capable to switch between an ARQ and a set of ARQ/FEC error control protocols. The coding scheme used in the ARQ/FEC protocols is based on RSE codes. The adaptive algorithm chooses the best error control mechanism as a function of the channel bit error rate, the channel state of the receivers and the desired QoS metric of the receivers while maximizing efficiency. We used a finite state Markov chain as our wireless channel model. This model allowed us to predict the future states of the channel for each receiver based on its current channel state. Simulation results showed that the use of adaptive mechanism is useful in order to save bandwidth while maintaining the QoS metrics below their thresholds.

We considered efficiency and packet loss rate for our analysis. The effect of our adaptive protocol on other QoS metrics such as delay, jitter, dropping rate and power consumption of



(a) Efficiency



(b) Packet loss rate

Figure 2: Simulation results for $PLR = 50\%$

mobile terminals is an interesting direction for our future work.

References

[1] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *Proceedings of SIGCOMM 96*, Stanford, CA, August 1996, pp. 117–130.

[2] N. Nikaein and C. Bonnet, "On the performance of FEC for multicast communication on a fading channel," in *Proceedings of International Conference on Telecommunications IEEE ICT'00*, Acapulco, Mexico, May 2000.

[3] Tsunyi Tuan and Kihong Park, "Multiple time scale redundancy control for QoS-sensitive transport of real-time traffic," in *Proceedings of INFOCOM'00*, Tel Aviv, Israel, Mar. 2000.

[4] Jean Bolot and Thierry Turletti, "Adaptive error control for packet video in the internet," in *proccedings of ICIP'96*, Lausanne, Switzerland, Sept. 1996.

[5] Jean-Chrysostome Bolot, Sacha Fosse-Parisis, and Don Towsley, "Adaptive FEC-Based error control for interactive audio in the internet," in *proceedings of INFOCOM'99*, New York, March 1999.

[6] Chi-Yuan Hsu, Antonio Ortega, and Masoud Khansari, "Rate control for robust video transmission over burst-error wireless channels," *IEEE JSAC*, vol. 17, no. 5, pp. 756–773, May 1999.

[7] Daji Qiao and Kang G. Shin, "A two-step adaptive error recovery scheme for video transmission over wireless networks," in *proceedings of INFOCOM'00*, Tel Aviv, Israel, Mar. 2000.

[8] M. Khansari, A. Jalali, E. Dubois, and P. Mermelstein, "Low bit-rate video transmission over fading channels for wireless microcellular systems,," *IEEE Transactions on Circuits and Systems for Video Technolog*, vol. 6, no. 1, pp. 1–11, 1996.

[9] H. Liu and M. El Zarki, "Delay bounded type-II hybrid ARQ for video transmission over wireless networks," in *Proceedings of Conference on Information Sciences and Systems*, Princeton, NJ, March 1996.

[10] P. Godlewski, M. Braneci, and A. Serhrouchni, "An error control scheme for multicast communications over an ATM network," in *Proceedings of Singapore International Conference on Communication Systems (ICCS'94)*, Singapore, November 1994.

[11] Dan Rubenstein, Jim Kurose, and Don Towsley, "Real-time reliable multicast using proactive forward error correction," in *NOSSDAV'98*, Cambridge, England, July 1998, pp. 279–293.

[12] Anthony J. McAuley, "Reliable broadband communication using a burst erasure correcting code," in *Proceedings of SIGCOMM'90*, Philadelphia, Pennsylvania, September 1990.

[13] Luigi Rizzo, "On the feasibility of software fec," *Computer Communication Review*, April 1997, Source code available at http://www.iet.unipi.it/ luigi/fec.html.

[14] Dan Rubenstein, "Increasing the functionality and availability of reed-solomon fec codes: a performance study," Tech. Rep. UMass CMPSCI Technical Report 98-31, University of Massachusetts, August 1998.

[15] S. Lin and D. Costello, *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

[16] M. Zorzi, R.R. Rao, and L.B. Milstein, "On the accuracy of a first-order Markov model for data block transmission on fading channels," in *Proceedings IEEE ICUPC'95*, November 1995, pp. 211–215.

[17] H.S. Wang, "On verifying the first-order Markovian assumption for a Rayleigh fading channel model," in *Proceedings IEEE ICUPC'94*, 1994, pp. 160–164.

[18] Hong Shen Wang and Nader Moayeri, "Finite-state Markov channel - a useful model for radio communication channels," *IEEE Transactions on Vehicular Technology*, vol. 44, no. 1, pp. 163–171, February 1995.

[19] J. G. Proakis, *Digital Communications*, New York: MacGrawhill, 2nd edition, 1989.

[20] M. Zorzi, R.R. Rao, and L.B. Milstein, "Error statistics in data transmission over fading channels," *IEEE Transactions on Communications*, vol. 46, pp. 1468–77, November 1998.

[21] James R. Yee and Edward J. Weldon, "Evaluation of the performance of error-correcting codes on a Gilbert channel," *IEEE Transactions on Communications*, vol. 43, no. 8, pp. 2316–2323, August 1995.