Optimizing HARQ Feedback and Incremental Redundancy in Wireless Communications

Mai Zhang, Andres Castillo, Borja Peleato
Electrical and Computer Engineering
Purdue University
West Lafayette IN 47907 Email: {maiz,castil12,bpeleato}@purdue.edu

***Abstract*—Wireless networks have been adjusting their transmit power, modulation order, and coding rate based on the channel conditions for a long time. Such adaptive protocols aim to maximize the overall throughput by striking a trade-off between transmitting as much information as possible and minimizing the probability of loosing such information. However, there is little literature on adaptive retransmissions when failures occur.**

This paper studies the trade-offs that hybrid automatic repeat request (HARQ) protocols face when choosing the type and amount of incremental redundancy (IR) that should be sent when the decoding of a data block fails at the receiver. It proposes a method to optimize such choice for a system with non-ideal error correcting codes (ECC) and limited feedback capabilities. Additionally, it is shown through simulations that the overall data rate can be significantly increased by bundling the acknowledgements.

I. Introduction

The density of wireless devices and their traffic requirements have experienced an exponential growth during the past two decades, and it does not seem like this trend will slow down anytime soon. In order to accommodate such demand, researchers are pushing the limits of the available systems to squeeze as much throughput as possible. Instead of using conservative configurations capable of supporting the required data streams in a wide range of conditions, most systems are now using adaptive schemes to maximize the bandwidth efficiency in every scenario.

A significant amount of work has been devoted to designing algorithms for adapting physical layer parameters such as the transmit power, modulation, coding rate, etc. [1], [2], [3] However, there is little literature on adaptive retransmissions when failures occur. Traditional automatic repeat request (ARQ) schemes operate as follows: when packets are correctly decoded the receiver acknowledges them with a short ACK packet, and when the decoding fails (*e.g.* , the LDPC decoder does not converge or the CRC for the decoded packet is incorrect), the receiver can either send a NACK packet or wait for the transmitter's timeout to expire as if the packet had been lost. If the transmitter receives a NACK or fails to receive an ACK before the timeout, the whole packet is retransmitted, assuming that its data is still relevant[1].

Retransmitting the whole packet is justified when the previous one has been completely lost, but in many cases the receiver is able to recover sections of the packet or obtain information that can be leveraged in subsequent transmissions. In those cases, it is often enough for the transmitter to send a few additional bits, referred to as incremental redundancy (IR), which the receiver can use to recover the whole packet. This is commonly known as Type-II hybrid automatic repeat request (Type-II HARQ) [4].

Hybrid ARQ is not a new concept. Its throughput has been upper-bounded under the assumption of unlimited single bit IR and perfect feedback [5] and several methods have been proposed to optimize the IR block lengths under a finite number of retransmissions [6], [7], [8], [9]. However, most of the work has focused on idealized error correcting codes (ECC), known channels, and either infinite or single bit feedback. This paper proposes a method to optimize the length and type of the IR bits in a more realistic scenario with imperfect ECC codes, limited feedback, acknowledgement bundling, and an overhead cost for each round of incremental redundancy.

Sending an individual ACK/NACK for each transmitted data packet can be rather inefficient, so many systems have started bundling the packets into groups that can be acknowledged with a single bit [10]. This paper uses bundling, but for a different purpose. A single bit of feedback can only convey ACK/NACK information. If the packet (or bundle) has been successfully received (ACK), the transmitter can delete it from its memory, otherwise it will send a fixed block of IR bits or retransmit the whole packet. The receiver does not have enough capacity to specify whether it needs the whole packet to be retransmitted or just a few additional parity bits to correct the remaining errors. We will use bundling to increase the effective feedback capabilities. Instead of sending a small number of feedback bits for each packet, we will send a larger number of bits to convey information about the bundle as a whole.

It is obvious that, for a fixed number of feedback bits per bundle, smaller bundles will provide higher throughput than large ones. Large bundles have a higher probability of suffering a decoding failure and, even if only one codeword fails, the transmitter will need to send incremental redundancy for all of the others. However, in practice the number of feedback bits is limited, *e.g.* to one per codeword. This paper will seek a tradeoff between having small bundles with limited feedback capabilities or large bundles with the ability of making multiple different requests.

[1]In some time sensitive applications, such as VOIP, the data needs to be received within a certain time interval, otherwise it is useless.

The paper will be organized as follows. Section II will introduce the system model and communication link parameters that will be used throughout the rest of the paper. Section III will discuss the different types of IR bits that will be considered and how each of them will be used to improve the available information on a given bundle. Section IV will propose a method for adaptively choosing the number and type of IR bits requested by the receiver based on the current state (number of decoding failures, coding rate, and SNR of the previously received transmission). Finally, Section V will provide simulation results illustrating the performance of the proposed scheme and Section VI concludes the paper.

II. System model

In order to optimize the aforementioned parameters for a given communications system, we will need to characterize some of the components of such system.

A. Channel

Wireless systems probe the channel periodically, adapting the code rate and modulation to remain within a certain frame error rate (FER) range. However, it is impossible to obtain a perfect characterization of the channel for every codeword. Sudden obstructions or beam misalignment in mmWave links can temporarily degrade the quality of the channel, until the next channel sounding cycle.

There exists a certain amount of correlation in the channel experienced by adjacent codewords (adjacent in time or frequency). For simplicity, the channel will be modeled as an interference-free AWGN with variable SNR. It will be assumed that all the codewords in a given bundle experience the same SNR, but it is possible that subsequent transmissions of incremental redundancy are received with different quality.

B. Modulation

When the SNR is high, the transmitter can group multiple bits in a single modulation symbol so as to increase throughput. Most communication systems are limited by the average transmit power, so amplitude modulations are typically chosen since they offer lower probability of error than phase modulations. There are multiple ways in which bits can be mapped to the symbol values. Fig. 1 shows a 16-QAM constellation with two mappings: Gray and superposition. Gray mapping offers lower bit error rate, but it is slightly harder to implement.

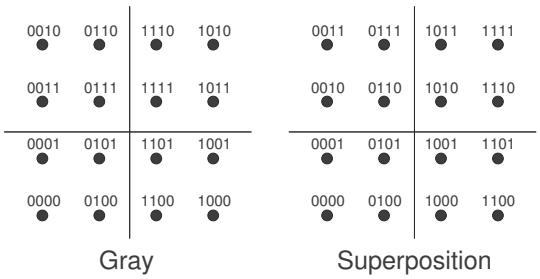

FIGURE 1: Mapping of bits to 16-QAM constellation.

Unfortunately, amplitude modulations with more than two bits per symbol yield different probabilities of error for different bits, regardless of the mapping. Ideally, these symbols could be encoded with non-binary error correction codes, which are not affected by the differences between individual bits. The performance with joint demodulation and decoding could then approach capacity [11], but the computational requirements of non-binary decoders render them impractical for many applications. Binary error correcting codes cannot capture the correlation between bits in the same symbol, having to deal with their marginal distributions instead. The decoder therefore operates as if the bits came from independent BPSK modulations with constant SNR throughout each codeword.

High-order modulation symbols will be constructed using bits from different codewords, with different coding rates according to the probability of error that they will suffer. That is, in a stream of 16-QAM symbols, good bits (MSB in Fig. 1) will be grouped in codewords with high rate and bad bits (second bit in Fig. 1) in codewords with lower rate. By splitting the bits in this way instead of mixing good and bad ones in the same codeword, we obtain a slight performance gain [12].

C. Error Correction

Most standards and prototype systems have chosen to use binary QC-LDPC codes due to their outstanding performance and parallel architecture, which allows very efficient encoding and decoding using parallel shift registers [13], [14]. Furthermore, LDPC codes can be easily punctured or extended to increase or decrease their information rate, respectively.

LDPC codes are characterized by a sparse parity check matrix $H \in \{0,1\}^{(n-k)\times n}$, such that $Hx = \mathbf{0}$ for all codewords x. Received channel values (*i.e.* matched filter output) are processed to obtain a log-likelihood ratio (LLR) for each individual bit b_i as

$$\ell_i = \mathrm{LLR}(b_i|r_i) = \log \frac{p_i(0|r_i)}{p_i(1|r_i)}, \tag{1}$$

where $p_i(0|r_i)$ and $p_i(1|r_i)$ respectively represent the probability of b_i being 0 or 1, given the corresponding received value r_i. These LLRs are then progressively refined by iterative message passing over the Tanner graph of the H matrix, until they converge to a feasible codeword or the algorithm reaches a maximum number of iterations. LDPC decoders very rarely converge to a wrong codeword; it is much more likely that they simply fail to converge by the maximum number of iterations.

Our simulations will focus on the QC-LDPC code of length $n = 648$ and $k = 432$ (rate 2/3) proposed in the 3GPP standard for 802.11n [15], but the techniques proposed here could be applied to any other code by adjusting the FER characteristics. As shown in [12], the FER for this code (and extensions) can be well approximated by

$$P_e(R, SNR) = Q\left(\frac{\mu - R}{\sigma}\right), \tag{2}$$

where $\mu = -0.2 \cdot SNR^{-1.74} + 0.86$, $\sigma = 0.12 \cdot SNR^{-0.42} - 0.08$, SNR is in linear scale, and R represents the code rate.

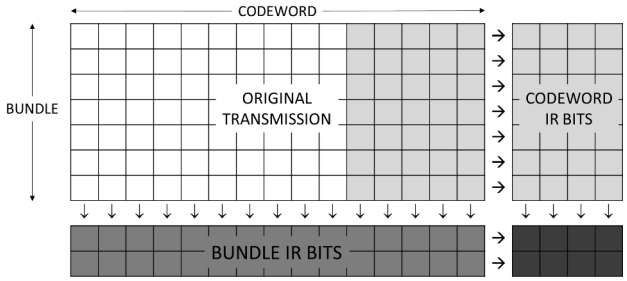

FIGURE 2: Types of incremental redundancy.

D. System

This paper considers a point to point link with a noiseless low-rate feedback channel used for acknowledging packets or requesting incremental redundancy. It will be assumed that in the ideal case there is one bit of feedback available for each codeword, giving us 16 possible feedback messages for a bundle of 4. However, it may turn out that our policy does not require that many feedback messages per bundle, in which case the required number of feedback bits can be even lower. Furthermore, our simulations will perform an unlimited number of incremental redundancy rounds, with a constant overhead penalty for each of them. That means, if any codeword in a bundle fails, the receiver will keep requesting additional bits until the whole bundle can be successfully decoded, but each additional request will have an additional cost on top of that incurred by the requested bits.

III. Incremental redundancy

The term "Incremental Redundancy" (IR) includes any additional bits requested by a receiver so as to attempt the decoding of a codeword (or bundle of codewords) that had previously failed. As shown in Fig. 2, these bits can be of different types, depending on how they are constructed:

1) Chase combining [16]: retransmit some of the bits previously sent. The receiver can use maximal ratio combining to improve their effective SNR. It has the advantage of simplicity, since there is no need to change the decoder, but it offers suboptimal performance.
2) Codeword parity (or extension) bits: generate new parity for a specific codeword by extending the matrix H with new rows and columns representing combinations of bits not previously used. There are ways to avoid complicating the decoder, but it cannot take advantage of correlation between codewords.
3) Bundle parity bits: construct a bitwise erasure code across multiple bundled codewords [17]. The decoding is slightly more complicated, but it couples multiple codewords together, making it possible to exploit their correlation and leverage the received values.

Practical LDPC decoders have limited memory, often insufficient to handle a joint decoding of all the codewords and IR in a bundle. Therefore, we will assume that each codeword is decoded independently, after refining the input LLR values with Chase combining IR and bundle parity bits. Codeword parity bits, however, are part of the codeword and can be directly fed to the LDPC decoder.

If the same bit b_i is received twice (Chase combining) with independent noise components, $r_i^{(1)} = b + n^{(1)}$ and $r_i^{(2)} = b + n^{(2)}$, then $p_i(0|r_i^{(1)}, r_i^{(2)})$ is proportional to $p_i(0|r_i^{(1)})p_i(0|r_i^{(2)})$. The same applies for $b_i = 1$. Hence, the joint LLR can be found by simply adding the individual LLR values for each transmission. In the AWGN paradigm, $\mathrm{LLR}(b_i|r_i) = 2 \cdot SNR \cdot r_i$, so Chase combining effectively increases the SNR of the transmitted bits to

$$SNR_{\mathrm{CC}} = \frac{4}{\frac{1}{SNR_0} + \frac{1}{SNR_{\mathrm{IR}}}}, \tag{3}$$

where SNR_0 and SNR_{IR} represent the SNR of the original and incremental redundancy transmission.

Similarly, we can model the effect of the bundle parity bits as an increase in SNR for the failed codewords in the bundle. Suppose that a vector of n bits from different codewords, $\mathbf{b} = [b_1, b_2, \cdots, b_n]$, and their XOR, $x = b_1 \oplus \cdots \oplus b_n$, are transmitted through AWGN channels with different SNR, resulting in received values $\mathbf{r}$ and r_x, respectively. The probability of a specific bit b_i being 0 conditioned on these received values can be found as

$$p_i(0|\mathbf{r}, r_x) = \sum_{\substack{\mathbf{d}\in\{0,1\}^n \\ d_i=0}} \frac{\left(\prod_{j=1}^{n} p_j(d_j|r_j)\right) p_x(\bigoplus \mathbf{d}|r_x)}{\sum_{\mathbf{v}\in\{0,1\}^n} \left(\prod_{j=1}^{n} p_j(v_j|r_j)\right) p_x(\bigoplus \mathbf{v}|r_x)}, \tag{4}$$

where $p_x(\bigoplus \mathbf{v}|r_x)$ represents the probability that $x = v_1 \oplus \cdots \oplus v_n$ given the received value r_x. Unfortunately, the number of terms in Eq. (4) increases exponentially with the bundle size, which makes it impractical. In our simulations, we use a min-sum approximation similar to that used in a min-sum LDPC decoder [18]. Specifically, the updated LLR is calculated as

$$\ell_k^{\mathrm{new}} = \ell_k + \left(\prod_{\substack{i=1 \\ i\neq k}}^{n+1} \operatorname{sign} \ell_i\right) \min_{\substack{i=1\ldots n+1 \\ i\neq k}} \left|\ell_i\right|, \tag{5}$$

where ℓ_{n+1} represents the LLR value for $x = \bigoplus \mathbf{b}$. Fig. 3 shows the SNR resulting from such an update, as a function of the number of failed codewords and assuming that both the original transmission and the IR experienced identical SNR_0.

LDPC decoders can fail to converge, but it is extremely rare for them to converge to the wrong codeword. Successfully decoded codewords can therefore be substituted in the bundle parity equations, reducing them to combinations of the failed codewords. The LLR values for the failed codewords will then be updated using Eq. (5) before attempting to decode them. If the decoding of any previously failed codeword succeeds, the LLRs can be updated again. The decoder does multiple rounds of updating LLRs and decoding, until either all codewords in the bundle are successfully decoded, or some of the codewords repeatedly fail regardless of the effort of updating LLRs. We

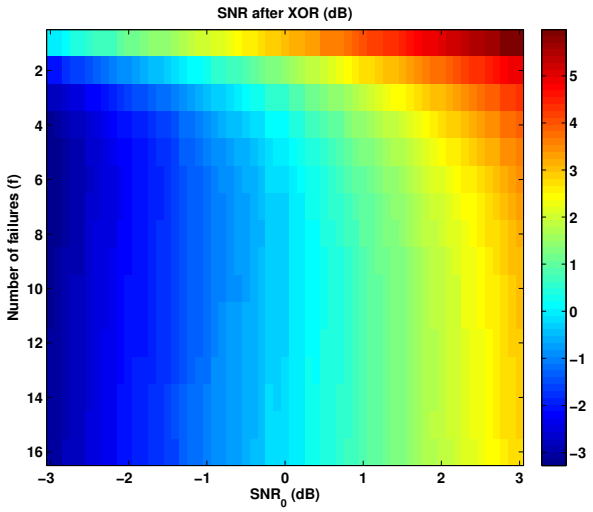

FIGURE 3: SNR after updating the LLRs of f bits based on a transmission of their XOR.

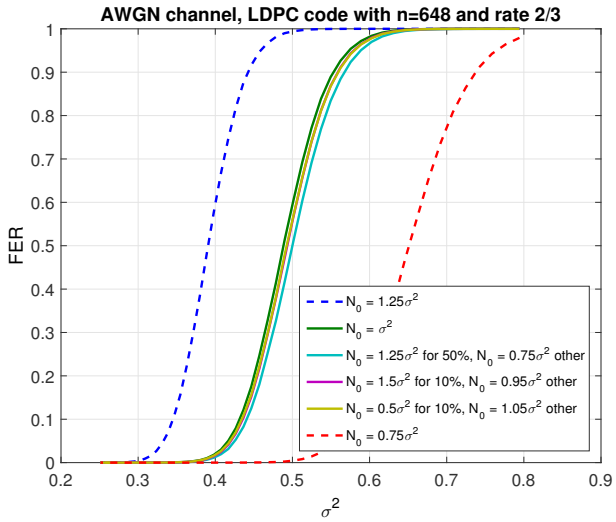

FIGURE 4: Probability of decoding failure as a function of noise power, for a signal with $E_b = 1$. Solid curves, corresponding to cases with the same SNR_{eff}, are very close.

call the former case a successful bundle, and the latter a failed bundle or a bundle error.

Finally, codeword extension bits reduce the rate of the code. The effect that this has on the probability of decoding success is intrinsic to the code being used. Typically, this performance is measured under the assumption that both the transmitted codeword and its associated IR are received with the same SNR, but this may not be the case in practice. Decoding failures often happen due to sudden falls in SNR, whereas IR is generally transmitted with higher SNR by adjusting the modulation order, transmit power, re-aligning the beam, re-estimating the channel, etc.

Our derivations can be greatly simplified by defining the effective SNR of a codeword as

$$SNR_{\text{eff}} = \left(E\left[\frac{1}{SNR} \right] \right)^{-1}. \tag{6}$$

Assuming that the energy per bit E_b is constant throughout the codeword, the effective SNR is just E_b divided by the average noise power. Fig. 4 shows the probability of decoding failure for different noise power mixtures within a codeword. It shows that the probability of failure mainly depends on the effective SNR, not on the SNR variance. The solid curves, which can barely be distinguished, all have the same effective SNR while the dashed ones illustrate the effect of a 25% increase and decrease in SNR.

Summarizing, Chase combining and bundle parity bits can be used to increase the SNR for some bits in the failed codewords, according to Eq. (3) and Fig. 3. This will in turn increase the effective SNR of those codewords, while extension IR bits reduce their code rate.

IV. Optimization

This section describes a method to optimize the type and number of IR bits requested based on the information available at the receiver, so as to minimize a given cost function. In order to make the problem manageable, the SNR and rate R are quantized to take a finite number of values. Furthermore, the number of IR bits requested is also restricted to a small pre-defined discrete set, so as to limit the number of feedback bits required to make such request. The HARQ protocol for a bundle of codewords can then be modeled as a Markov Decision Process (MDP) with a finite set of actions and states:

- State: $s = (f, SNR, R)$, where f represents the number of codewords in the bundle whose decoding failed, SNR their average signal to noise ratio, and R their coding rate. If the failed codewords have different SNR, we take the lowest one.
- Action: $A(s) = (\alpha, \beta)$, where α represents the number of extension bits requested (for each codeword in the bundle) and β represents the number of bundle parity bits requested. No Chase combining bits will be requested since extension bits always offer better performance for our system model [16].
- Cost: $C = b\alpha + \beta + \mathbb{I}_{\{\alpha+\beta\}} K$, where b denotes the number of codewords per bundle, and $\mathbb{I}_x$ represents an indicator function taking value 1 when $x \neq 0$ and 0 otherwise. This cost counts the number of IR bits sent in a certain round, plus a penalty K to account for the overhead.

The objective is to minimize the total cost until success, which in effect maximizes the overall throughput, *i.e.* we wish to find

$$A(s) = \arg\min_{\alpha,\beta} E\{\text{Total cost} | s, \alpha, \beta\} \tag{7}$$

for all s. Action (α, β) will reduce the rate and increase the SNR, taking the system from state $s_1 = (f_1, SNR_1, R_1)$ to $s_2 = (f_2, SNR_2, R_2)$, with $f_2 \leq f_1$, SNR_2 given by Fig. 3,

and

$$R_2 = \frac{k}{k + \alpha R_1} R_1 \tag{8}$$

where k is the number of information bits in a codeword. SNR_2 and R_2 are assumed to be deterministic, but the number of failures f_2 follows a binomial distribution with pmf

$$P(f_2|s_1, \alpha, \beta) = \binom{f_1}{f_2} p^{f_2} (1-p)^{f_1 - f_2}, \tag{9}$$

where p represents the probability of decoding failure with the IR. It is important to realize that this decoding with IR cannot be treated as an independent attempt; the probability of failure should be conditioned on the fact that the decoding failed without IR. Therefore,

$$p = \frac{P_e(SNR_2, R_2)}{P_e(SNR_1, R_1)}, \tag{10}$$

where $P_e(SNR, R)$ represents the probability of decoding failure for a single codeword with the given effective SNR and rate, approximated in Eq. (2).

The optimal number of bits to be requested when the system is in state s can be found as

$$\begin{aligned} A(s) &= (\alpha^\star(s), \beta^\star(s)) \\ &= \arg\min_{\alpha,\beta} E\{\text{Total cost}|s, \alpha, \beta\} \\ &= \arg\min_{\alpha,\beta} b\alpha + \beta + \sum_{s'} P(s'|s, \alpha, \beta) V(s'), \end{aligned} \tag{11}$$

where $P(s'|s, \alpha, \beta)$ denotes the probability of transitioning from s to s' by sending (α, β) bits, and $V(s)$ denotes the expected total cost to get from state s to success:

$$\begin{aligned} V(s) &= E\{\text{Total cost}|s, A(s)\} \\ &= b\alpha^\star(s) + \beta^\star(s) + \mathbb{I}_{\{A(s)\}} K + \sum_{s'} P(s'|s, A(s)) V(s'). \end{aligned} \tag{12}$$

The transition probability $P(s'|s, A(s))$ is given by Eq. (9) if Fig. 3 and Eq. (8) hold, and is 0 otherwise.

With the states discretized to a finite number of values, we can use the value iteration algorithm [19] to optimize $V(s)$ and $A(s)$ for all s. Essentially, it starts with a random value function and alternates between updating the policy $A(s)$ according to Eq. (11) and the value $V(s)$ according to Eq. (12), until convergence. At that point $A(s)$ stores the policy to be followed whenever the system is in state s.

V. Numerical results

This section provides simulations illustrating the policies obtained with the proposed method and comparing their data rate with that obtained by sending fixed amounts of IR for each individual codeword. For this purpose, the codewords were bundled in groups of $b = 4$ and the overhead penalty was set at $K = 300$. This overhead can include actual bits sent (*e.g.* packet headers) as well as virtual penalties for the additional latency or complexity.

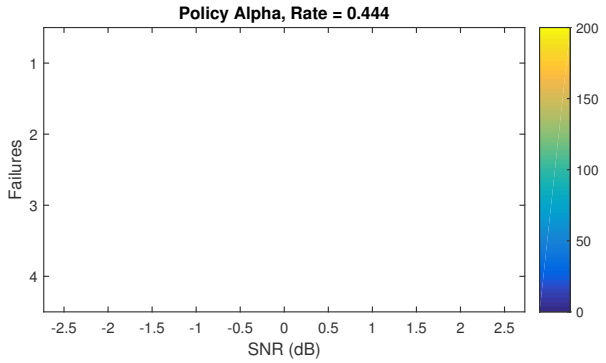

FIGURE 5: Policy obtained for α at rate $\frac{4}{9}$

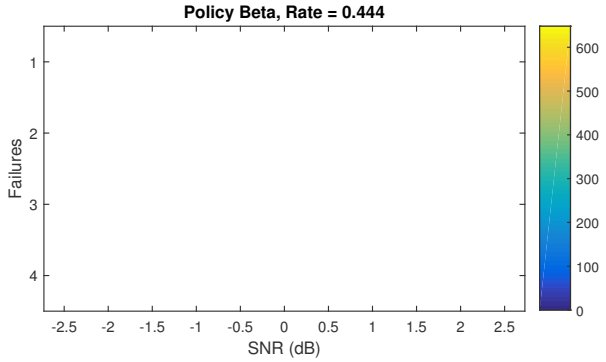

FIGURE 6: Policy obtained for β at rate $\frac{4}{9}$

Figs. 5 and 6 show the policies obtained for α and β at rate $R = \frac{4}{9}$, *i.e.* the number of extension bits (α) and bundle parity bits (β) to be requested after having already received 324 extension bits, as a function of the number of failed codewords remaining in the bundle and the effective SNR of those codewords. It can be observed that, as the SNR decreases, the number of bits to be requested increases. This makes sense, since highly corrupted bundles will require more IR for successful recovery. Also, when the number of failures is small our policy suggests requesting bundle parity bits instead of extension bits. Again, this makes sense since the receiver cannot convey to the transmitter which codewords failed. If it requests extension bits, the transmitter will have to send them for every codeword in the bundle, even for those that have already been successfully decoded. Furthermore, the gain provided by bundle parity bits is much higher when the number of failures is low, as shown in Fig. 3.

The policies we obtained only have 16 possible combinations of (α, β), so 4 bits of feedback are sufficient to specify the retransmission strategy. Therefore our proposed policy requires 1 feedback bit per codeword, which is comparable to traditional fixed IR schemes with individual acknowledgments.

Fig. 7 compares the data rate of these schemes. The data rate is computed by dividing the number of information bits correctly delivered by the total number of bits transmitted, including an overhead of $K = 300$ bits for each IR round. It can be observed that our policy provides significantly higher data

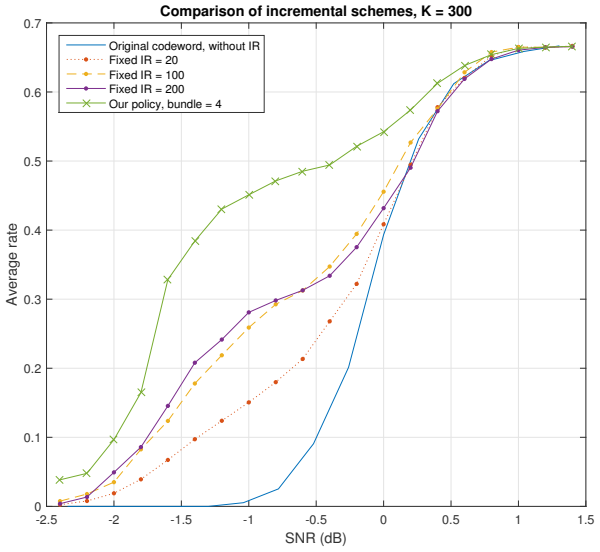

FIGURE 7: Average data rate for different retransmission schemes.

rate than fixed IR ones, regardless of what this fixed number is and the SNR. The difference is specially large in the low SNR regime, since our policy can request many IR bits in the first round whereas fixed IR ones require multiple IR rounds incurring in large overhead penalties.

VI. Conclusion

This paper proposes modeling the allocation of IR bits in a HARQ protocol as a Markov Decision Process seeking to minimize a pre-determined cost function. It described how the problem should be formulated and solved, resulting in a set of policies parameterized by the number of failures per codeword bundle, effective SNR of the received codewords, and coding rate.

Simulation results show that the proposed method provides a significant increase in throughput with respect to traditional fixed-length HARQ schemes, where codewords are individually acknowledged instead of bundled. The increased flexibility in requesting different numbers and types of IR bits more than makes up for the fact that the receiver cannot specify which codewords have failed and which ones have been successfully decoded.

Acknowledgment

This work was supported by AFRL and DARPA under grant 108818 and by Nokia Networks.

References

[1] F. Peng, J. Zhang, and W. E. Ryan, "Adaptive modulation and coding for IEEE 802.11 n," in Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE. IEEE, 2007, pp. 656–661.

[2] C. U. Castellanos, D. L. Villa, C. Rosa, K. I. Pedersen, F. D. Calabrese, P.-H. Michaelsen, and J. Michel, "Performance of uplink fractional power control in UTRAN LTE," in Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE. IEEE, 2008, pp. 2517–2521.

[3] B. Furht and S. A. Ahson, Long Term Evolution: 3GPP LTE radio and cellular technology. Crc Press, 2016.

[4] S. Lin, D. J. Costello, and M. J. Miller, "Automatic-repeat-request error-control schemes," IEEE Communications magazine, vol. 22, no. 12, pp. 5–17, 1984.

[5] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Feedback in the non-asymptotic regime," IEEE Transactions on Information Theory, vol. 57, no. 8, pp. 4903–4925, 2011.

[6] K. Vakilinia, A. R. Williamson, S. V. Ranganathan, D. Divsalar, and R. D. Wesel, "Feedback systems using non-binary LDPC codes with a limited number of transmissions," in Information Theory Workshop (ITW), 2014 IEEE. IEEE, 2014, pp. 167–171.

[7] A. R. Williamson, T.-Y. Chen, and R. D. Wesel, "A rate-compatible sphere-packing analysis of feedback coding with limited retransmissions," in Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on. IEEE, 2012, pp. 2924–2928.

[8] R. D. Wesel, K. Vakilinia, S. V. Ranganathan, and D. Divsalar, "Resource-aware incremental redundancy in feedback and broadcast," in International Zurich Seminar on Communications, 2016, p. 63.

[9] K. Vakilinia, S. V. Ranganathan, D. Divsalar, and R. D. Wesel, "Optimizing transmission lengths for limited feedback with nonbinary LDPC examples," IEEE Transactions on Communications, vol. 64, no. 6, pp. 2245–2257, 2016.

[10] R. Susitaival and M. Meyer, "LTE coverage improvement by TTI bundling," in Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th. IEEE, 2009, pp. 1–5.

[11] M. C. Davey and D. MacKay, "Low-density parity check codes over GF (q)," IEEE Communications Letters, vol. 2, no. 6, pp. 165–167, 1998.

[12] J. Song, B. Peleato, D. J. Love, T. Luo, D. Ogbe, and A. Ghosh, "Optimizing incremental redundancy for millimeter wave wireless communication using low density parity check codes," https://engineering.purdue.edu/~bpeleato/mmWave_LDPC.pdf, 2017.

[13] Z. Li, L. Chen, L. Zeng, S. Lin, and W. H. Fong, "Efficient encoding of quasi-cyclic low-density parity-check codes," IEEE Transactions on Communications, vol. 54, no. 1, pp. 71–81, 2006.

[14] Z. Wang and Z. Cui, "Low-complexity high-speed decoder design for quasi-cyclic LDPC codes," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 15, no. 1, pp. 104–114, 2007.

[15] IEEE, IEEE 802.11n Wireless LAN Medium Access Control MAC and Physical Layer PHY specifications., IEEE 802.11n-D1.0 Std., 2006.

[16] P. Frenger, S. Parkvall, and E. Dahlman, "Performance comparison of HARQ with Chase combining and incremental redundancy for HSDPA," in Vehicular Technology Conference, 2001. VTC 2001 Fall. IEEE VTS 54th, vol. 3. IEEE, 2001, pp. 1829–1833.

[17] T. A. Courtade and R. D. Wesel, "Optimal allocation of redundancy between packet-level erasure coding and physical-layer channel coding in fading channels," IEEE Transactions on Communications, vol. 59, no. 8, pp. 2101–2109, 2011.

[18] T. Richardson and R. Urbanke, Modern coding theory. Cambridge university press, 2008.

[19] D. P. Bertsekas, Dynamic programming and optimal control. Athena scientific Belmont, MA, 1995, vol. 1, no. 2.