

A Hybrid ARQ Scheme for Resilient Packet Header Compression

Vijay A Suryavanshi and Aria Nosratinia

Multimedia Communications Laboratory, The University of Texas at Dallas

Richardson, TX 75083-0688, USA

E-mail: {vas021000, aria}@utdallas.edu

Abstract—In this work we address resilient packet header compression in a setup similar to RFC 3095 (RObust Header Compression – ROHC), where a noisy feedback channel is available. We propose a new predictive hybrid ARQ (HARQ), using a systematic convolutional code with delay-limited interleaving. The compressed packet headers are bit interleaved, encoded, and the parity bits are loaded onto the packets in a manner that is consistent with existing standards. Our HARQ design is distinct from previous ones in that it cannot wait until the end of a codeword for retransmission, rather, ARQ’s corresponding to individual packet headers arrive continually during a convolutional codeword and must be processed at that time. In our system, each time a NACK is encountered, the encoder estimates the state of the Viterbi decoder, deciding when a retransmission is necessary. The use of coding in conjunction with a specially designed interleaver, provides a flexible and powerful design methodology that makes it possible to improve the throughput via FEC and interleaving, while strictly limiting the delay to avoid timeout in higher layers. Simulations show that the throughput of the resulting system is superior to ROHC, and the delay is less sensitive to the channel conditions. Furthermore, over a large part of the operating range the delay is comparable to, or smaller than, the delay of ROHC.

I. INTRODUCTION

Header compression algorithms reduce the current overhead of 40 bytes per UDP/TCP packet down to 2–4 bytes. Even though compression efficiency is high, error propagation remains a problem. For unidirectional links, a Forward Error Correction(FEC) has been proposed [1], [2] leading to significant improvements over past results [3], [4].

This paper develops coding for packet header compression on bi-directional links. We develop a new predictive hybrid ARQ (HARQ) technique that is specifically suitable to the application at hand, and is furthermore of interest on its own. The new HARQ technique is designed to act on feedback information indicating loss of parts of a codeword, as is the case in our application as well as (possibly) other applications with long codewords. To rein in the delay, we design a delay-limited interleaver. Simulations show that our proposed system improves the throughput considerably compared to RoHC (RFC 3095) [3], and furthermore in our system the delay profile is less sensitive to channel conditions.

A. Header Compression: Motivation and Background

In [5] it is observed that more than 55% of packets have length less than 200 bytes. Header compression algorithms

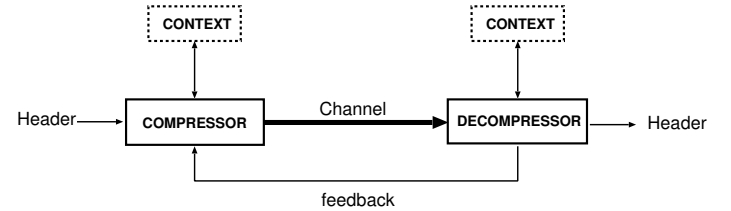


Fig. 1. Basic Header Compression Scheme: CONTEXT buffer stores the current uncompressed packet header

reduce this overhead by exploiting the temporal redundancy among packet headers. Van Jacobson [6] noted that header fields belonging to a particular packet stream¹ are the same or differ by a small amount packet to packet. Thus, header fields can be differentially encoded leading to significant bit savings.

As shown in Figure 1, each node is equipped with a compressor/decompressor and a CONTEXT buffer. Initially a regular (uncompressed) packet header is transmitted and copied into the CONTEXT buffer on each side. The next packet is differentially encoded with respect to the CONTEXT at the transmitter side and sent as a compressed packet. The CONTEXT is then updated by the uncompressed version of this packet. At the receiver side the compressed packet is decompressed with respect to the CONTEXT at the receiver side. Similarly, the CONTEXT at the receiver side is updated by the recently decompressed packet header.

Due to the differential encoding, a lost packet can desynchronize the compressor and decompressor leading to loss of subsequent packets. Note that the CONTEXT at the receiver side is updated only if a packet is correctly received. Whenever a packet is lost subsequent packets - even though they might be correctly received - will fail decompression and not passed on to the upper layers.

Several schemes have been proposed to avoid and/or reduce error propagation. In [4] the decompressor tries to locally correct the CONTEXT by applying the differences twice (the *twice* algorithm). On uni-directional links an uncompressed packet is transmitted at regular intervals [6]. In [1], [2], a FEC scheme is proposed wherein correctly received packets can be decompressed even if some packets are lost. In RFC 3095 [3]

¹packets belonging to a given source-destination pair

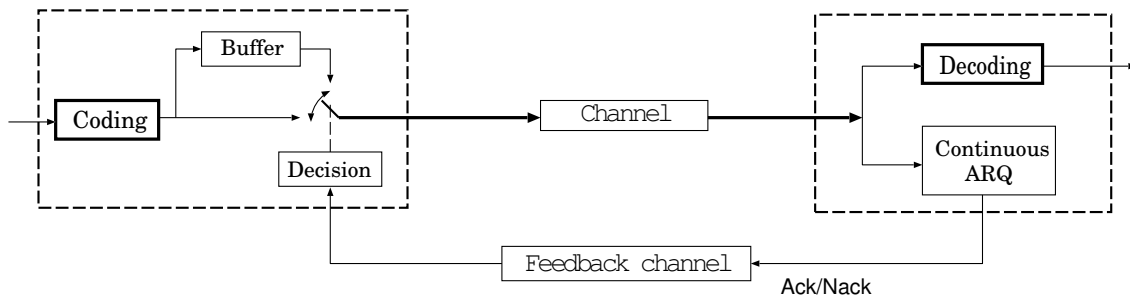


Fig. 2. Predictive Hybrid ARQ: A decision is made each time a NACK is received.

a sliding window of compressed packet headers is maintained and each incoming packet is compressed with respect to the contents of the current window. In the next sections we discuss the FEC approach and RFC 3095 in detail.

B. Coding Packet Headers

A novel approach to improve error resilience in header compression scheme via FEC coding is introduced in [1], [2]. In the absence of feedback when CONTEXT are desynchronized the receiver has to discard all the packets until it receives a uncompressed header. The basic idea in [1] is to apply Reed-Solomon codes on the packet headers. Thus when a packet header is lost it can be recovered and the CONTEXT on each side remain synchronized. Simulation results showed the benefits of coding packet headers, but Reed-Solomon decoding introduces complexity and delay into the system. To overcome this issue, a similar approach as in [1] but based on convolutional codes and delay limited interleaving is proposed in [2]. The Viterbi algorithm [7] is used in decoding convolutional codes. Compared to the quadratic complexity ($O(N^2)$) in decoding Reed-Solomon codes, Viterbi decoding is complexity linear in codeword length.

Figure 3 and 4 summarize the basic technique employed in [2]. The data bits belonging to an uncompressed header and compressed headers that follow are interleaved and presented to a recursive convolutional encoder. The resulting parity bits from the encoder are interleaved and distributed over the compressed packets uniformly. This uniform distribution of parity bits ensures that the system is not vulnerable to a specific packet being lost. Compared to the existing schemes, simulation results in [2] again highlight the benefits of coding.

C. RFC 3095 – ROHC

ROHC outlined in RFC 3095 achieves compression by defining compression profiles and mapping functions to derive untransmitted header fields at the receiver. ROHC employs a specialized differential encoding method known as Window-based Least Significant Bit (W-LSB) encoding. The compressor maintains a sliding window of already transmitted compressed packet headers. Based on the values in the current window a number k is derived and only the k LSB bits of the header field are transmitted. For a detail explanation of W-LSB the reader is referred to [2], [3].

When feedback is not present, the ROHC scheme specifies transmission of uncompressed headers at regular intervals. This is the U-mode (uni-directional mode) of ROHC also referred to as ROHC-U. In presence of feedback, ROHC either operates in the Bidirectional Optimistic mode (ROHC-O) or Bidirectional Reliable mode (ROHC-R). In this paper we concentrate on the ROHC-O mode due to its sparse usage of the feedback channel and hence is the preferred choice in literature. When in O-mode, the decompressor upon the receipt of a uncompressed packet always responds by an ACK. This allows the compressor to send compressed packets² by transiting to a higher compression state. A lost or damaged packet triggers a NACK from the decompressor resulting in retransmission of update information from the compressor.

II. PREDICTIVE HYBRID ARQ

In this section we propose a new hybrid ARQ scheme to address the needs of header compression, which is also of interest by itself. In the previous section, we saw that headers are coded across many packets. When we have feedback, the encoder learns of the status of a header *in the middle of a codeword*. Thus we develop a new HARQ technique that can utilize information about reception or non-reception of *parts* of a codeword. In each instance that at NACK is received, the encoder will predict whether or not the decoder can recover from it (due to the presence of coding). According to this prediction, the encoder decides when to retransmit (some of) the lost segments.

The new method is distinct from existing HARQ techniques in that it acts on partial losses in the codeword. This is especially relevant when the codeword is long and multiple ACK/NACKs may be available in the duration, a subject that, to our knowledge, has not been broached to date. We speculate that this technique may find application with various other codes that have long codewords, for example LDPC codes and raptor codes.

In the existing HARQ schemes [8] the decoder waits until all the bits corresponding to the codeword are received, then tries to decode the codeword and request retransmission if necessary. We propose a new predictive HARQ technique

²Note that the transitions involved at the compressor are more complex than explained here. Again the reader is referred to [3]

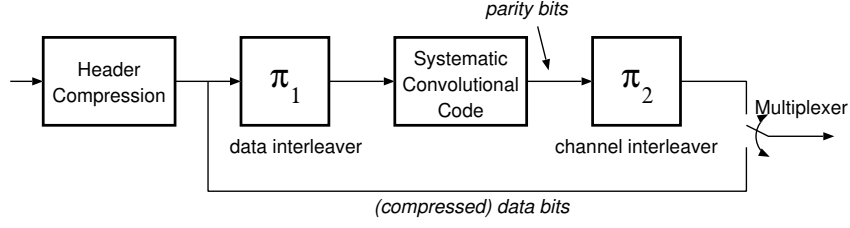


Fig. 3. Interleaving and Convolutional Encoding of header bits. This represents the Coding block in Figure 2

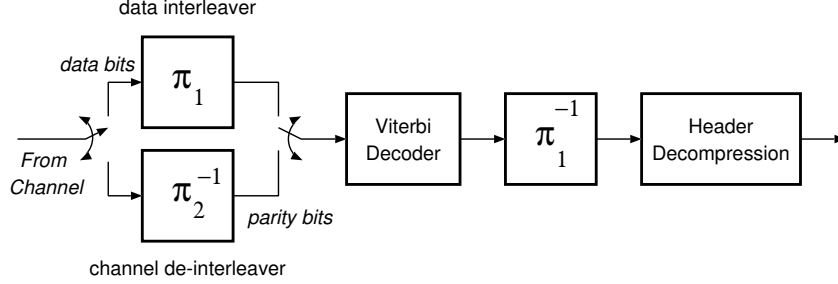


Fig. 4. Deinterleaving and Viterbi Decoding of header bits. This represents the Decoding block in Figure 2

wherein the decoder continues to receive information from the encoder which aids in successful decoding.

As shown in Figure 2, each time a packet is lost, the decoder transmits a NACK³ to the encoder. The “Coding” and “Decoding” blocks in Figure 2 are expanded in Figures 3 and 4, respectively. Once a NACK is received, the transmitter will mimic the decoding operation and will determine if this packet loss will result in a decoding failure. If so, it will retransmit the packet. Due to non-zero round-trip time, a buffer is provided at the encoder to keep the (compressed) headers in case they need be retransmitted.

For example, consider encoding of 10 packet headers as shown in Figure 3 where each packet is denoted by P_i , $i \in \{1, 2, \dots, 10\}$. Assume that packets P_1, P_4 and P_9 are lost. The decoder at the receive side immediately sends a NACK corresponding to the sequence number belonging to P_1 . The transmitter mirrors the receiver’s Viterbi decoding and finds out that even with the loss of P_1 the decoding can still be successful, so no action takes place. Next, the decoder at the transmit side receives NACK for P_4 . Note that P_1 has also gone missing. With the bits in *both* packets missing, the transmitter determines that decoding will fail, so the transmitter will retransmit P_4 . Similarly, retransmission of P_9 depends upon the outcome of decoding at the transmit side. Note that decoding is attempted by erasing only bits belonging to P_1 and P_9 since P_4 is already retransmitted and available at the receiver.

In Figure 3 and 4 interleavers are used to shuffle the data and parity bits prior to transmission. This improves decoding capability of the code and thus increases the throughput. However, interleaving always incurs delay. In general the

maximum delay can be as large as the codeword length, which is unacceptable in our application. In the next section we provide a delay limited interleaver design algorithm which can construct an interleaver with a specified delay.

III. DELAY LIMITED INTERLEAVER DESIGN

An interleaver $\pi(i)$ with spread S ensures that adjacent bits within distance S at the input are mapped S distance apart at the output. The delay of an interleaver is defined as follows [9]:

$$\delta^+ = \max_x (\pi(i) - i) \quad (1)$$

A classic block interleaver fills a matrix row wise with input bits and reads them out column wise. If M and N are number of rows and columns respectively of a block interleaver matrix, then the delay of this interleaver is given as [9]:

$$\mathcal{D} = (M - 1) \times (N - 1) \quad (2)$$

We denote by J_{ij} as the i^{th} row and j^{th} column of interleaver matrix \mathbf{J} . The period P of the interleaver is the number of bits interleaved at a given time. If P is the period of the interleaver, then \mathbf{J} has dimension $P \times P$. Figure 5 shows the interleaver matrix \mathbf{J} of a block interleaver with $M = N = 4$ (and therefore $P = M \times N = 16$). Note that the interleaver has delay $\mathcal{D} = 9$ bits with $S = M - 1 = 3$. As can be seen from the figure, each row and column has only one 1. Thus initially a 1 is placed in the 1^{st} row and 1^{st} column and then in the 2^{nd} row and $1 + (S + 1)^{\text{th}}$ column and so on.

Let \mathcal{D}_{max} be the maximum allowable delay of the system (in bits). From 1 it can be seen that if an interleaver needs to be designed with \mathcal{D}_{max} in view, then

$$\pi(i) \leq \mathcal{D}_{max} + i \quad (3)$$

³containing the sequence number (SN) of the lost packet

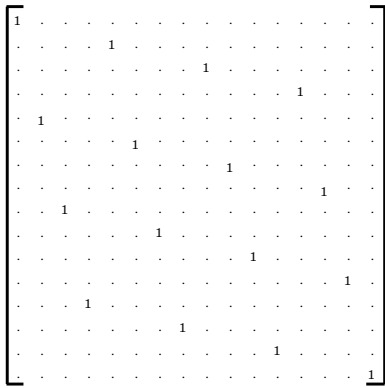


Fig. 5. Block Interleaver with $S = 3$ and $\mathcal{D} = 9$ bits.

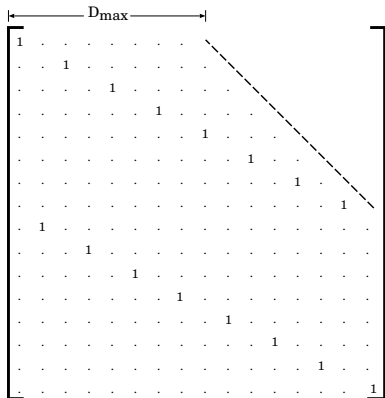


Fig. 6. Delay Constrained Interleaver with $S = 1$ and $\mathcal{D}_{max} = 7$ bits.

As in the case of a block interleaver, we start by placing a 1 in the 1^{st} row and 1^{st} column. Note that the row represents the output bit of the interleaver. The next 1 is placed in the 2^{nd} row and $1 + (S + 1)^{th}$ column. Similarly for the third output bit of the interleaver, a 1 is placed in the 3^{rd} row and $2 + (S + 1)^{th}$ column. At some point, we will need to wrap back once we hit or exceed the delay limit (shown in Figure 6 by a dashed line) or when the column value exceeds the period P of the interleaver. In this case we place a 1 in the next least valued and unoccupied column. We continue to fill the matrix with 1s in a similar fashion.

The resulting interleaver matrix will have a similar block interleaver structure but some of the permutation indices are not allowed. Thus the interleaver matrix is filled as in the case of a block interleaver but will have delay less than \mathcal{D}_{max} . Figure 6 shows the interleaver matrix \mathbf{J} with spread $S = 1$ and maximum delay $\mathcal{D}_{max} = 7$ bits. The resulting \mathbf{J} will have a trapezoidal appearance due to the delay constraint. Next we describe the algorithm to design such an interleaver.

A. The Algorithm

The proposed algorithm accepts as input the allowable delay \mathcal{D}_{max} and an initial guess S_{max} as spread of the interleaver. It can be shown that a necessary and sufficient condition for an interleaver with spread S and period P to exist is that $P \geq S^2$

therefore our initial guess for S_{max} is $\lfloor \sqrt{P} \rfloor$. The algorithm is described in detail as follows:

START

1. Set $S \rightarrow S = S_{max} = \lfloor \sqrt{P} \rfloor$
2. Initialize
 - $\mathbf{J} = 0$
 - $i = j = 0$
 - $J_{ij} = 1$
3. Increment the row value $\rightarrow i = i + 1$
4. If $i \geq S$ goto 8
5. Set $j = i + S + 1$
6. If $j > \mathcal{D}_{max} + i$, then $S = S - 1$ and goto 2
7. Set $J_{ij} = 1$ and goto 3
8. If $j > \mathcal{D}_{max} + i \rightarrow$ goto 9
9. Wrap around \rightarrow select $j = \min_{unoccupied} column$
10. if $j = \{\}$, then $S = S - 1$ and goto 2
11. Set $J_{ij} = 1$ and goto 3

END

At the successful completion of this algorithm we have an interleaver matrix \mathbf{J} adhering to equation 3 and with spread $S \leq S_{max}$.

IV. SIMULATION RESULTS

We compare throughput and delay of the proposed scheme with O-mode of RFC 3095. Two channel models are considered, an i.i.d. channel and a two-state Markov channel and the interleaver is designed with a delay of 890 bits with above design algorithm. The uncompressed packet is of 40 bytes followed by 2 bytes each of 40 compressed packets, the total codeword length is thus 960 bits. For a fair comparison the overall rate of the two systems is kept same. We use the recursive systematic convolutional encoder given in [10] with rate $1/2$, $K = 3$ and generator matrix $g^0 = [101]$ and $g^1 = [111]$.

In all experiments it is assumed that the decoder timeouts after a period of time equivalent to one packet (i.e., the time taken to transmit one packet across the channel) and retransmits the NACK again. The feedback channel is assumed i.i.d. with a constant packet loss rate of 0.1. As shown in Figure 7 and 8 for the iid channel, the proposed scheme performs favorably for all channel conditions except in the case of delay for low packet error rates due to interleaving. However overall the proposed scheme performs well. For the bursty channel it is assumed that the average burst length is 5 packets. Again the proposed scheme performs very well even under bursty condition (Figure 9 and 10) for all channel conditions. The vertical axis in all the simulation plots regarding delay is the average delay in terms of number of packets per transmitted codeword.

V. CONCLUSION

A predictive hybrid ARQ technique is proposed to improve the throughput and delay of header compression systems.

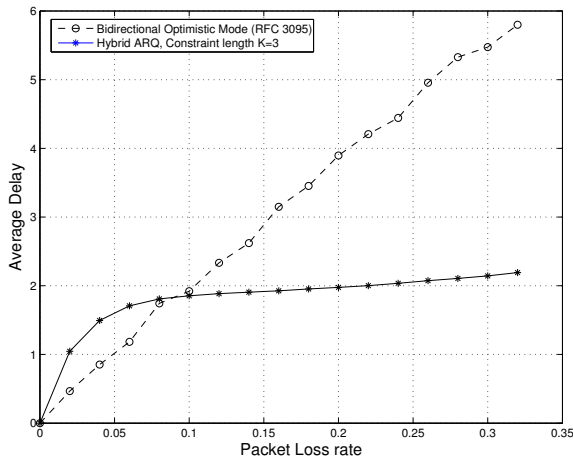


Fig. 7. Delay Comparison: IID Channel

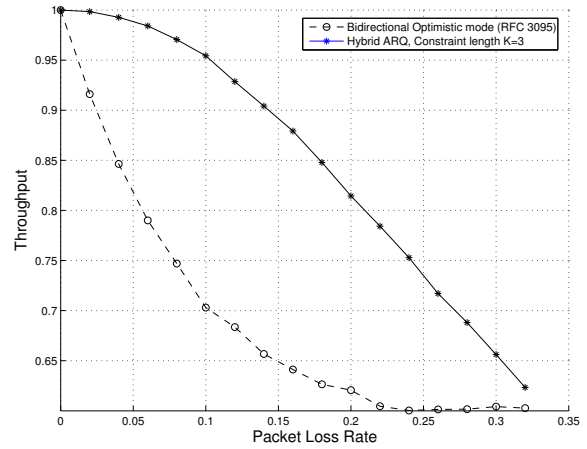


Fig. 8. Throughput Comparison: IID Channel

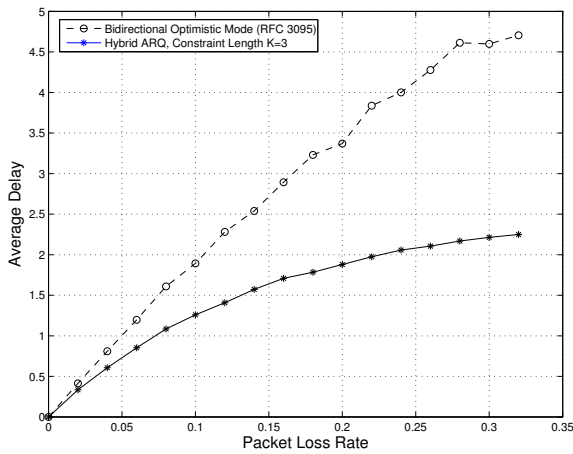


Fig. 9. Delay Comparison: Two-state Markov Channel, Burst length = 5

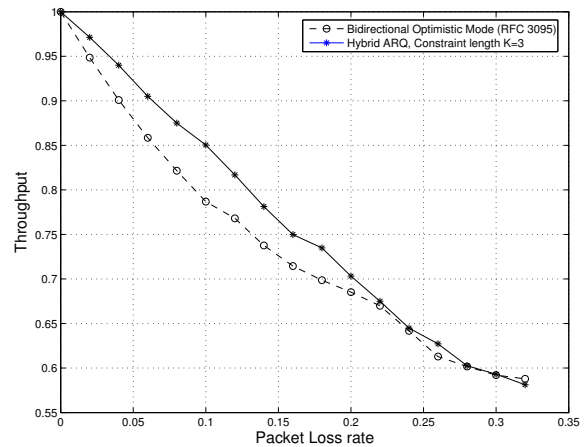


Fig. 10. Throughput Comparison: Two-state Markov Channel, Burst length = 5

For delay sensitive applications, an algorithm to design delay limited interleavers is presented. The proposed ARQ scheme in conjunction with the designed interleavers performs favorably compared to the existing schemes. In fact, the proposed HARQ technique and interleaver design methodology can be applied to other cases where delay is a crucial factor.

REFERENCES

- [1] V. Suryavanshi, A. Nosratinia, and R. Vedantham, "Resilient Packet Header Compression through Coding," *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, vol. 3, pp. 1635–1639, 29 Nov.-3 Dec. 2004.
- [2] V. Suryavanshi and A. Nosratinia, "Convolutional Coding for Resilient Packet Header Compression," (accepted) *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*.
- [3] C. Bormann and et al, "RFC 3095-RObust Header Compression(ROHC): Framework and four profiles: RTP,UDP,ESP,and un-compressed," RFC 3095 IETF Network Working Group, July 2001, <http://www.ietf.org/rfc/rfc3095.txt>.
- [4] M. Degermak, M. Engan, B. Nordgren, and S. Pink, "Low loss TCP/IP header compression for Wireless Networks," in *mobican96*, New York, NY, October 1997.
- [5] CAIDA, "Packet Length Distributions," HTTP Link, August 2004, <http://www.caida.org/analysis/AIX/>.
- [6] V. Jacobson, "TCP/IP Compression for Low-Speed Serial Links," RFC 1144 IETF Network Working Group, Feb 1990, <http://www.ietf.org/rfc/rfc1144.txt>.
- [7] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. 13, pp. 260–269, April 1967.
- [8] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ: Prentice Hall, 1995.
- [9] C. Heegard and S. B. Wicker, *Turbo Coding*. Kluwer International Series in Engineering and Computer Science, 1999.
- [10] P. Frenger, P. Orten, and T. Ottosson, "Convolutional codes with optimum distance spectrum," *IEEE Communications Letters*, vol. 3, pp. 317–319, November 1999.