Towards a Practical Scheme for Binary Broadcast Channels with Varying Channel Quality Using Dirty Paper Coding

Gyu Bum Kyung, *Student Member, IEEE,* and Chih-Chun Wang, *Member, IEEE*

***Abstract*—We consider practical schemes for binary dirty-paper channels and broadcast channels (BCs) with two receivers and varying channel quality. With the BC application in mind, this paper proposes a new design for binary dirty paper coding (DPC). By exploiting the concept of coset binning, the complexity of the system is greatly reduced when compared to the existing works. Some design challenges of the coset binning approach are identified and addressed. The proposed binary DPC system achieves similar performance to the state-of-the-art, superposition-coding-based system while demonstrating significant advantages in terms of complexity and flexibility of system design. For binary BCs, achieving the capacity generally requires the superposition of a normal channel code and a carefully designed channel code with non-uniform bit distribution. The non-uniform bit distribution is chosen according to the channel conditions. Therefore, to achieve the capacity for binary BCs with varying channel quality, it is necessary to use quantization codes of different rates, which significantly increases the implementation complexity. In this paper, we also propose a broadcast scheme that generalizes the concept of binary DPC, which we term *soft DPC*. By combining soft DPC with time sharing, we achieve a large percentage of the capacity for a wide range of channel quality with little complexity overhead. Our scheme uses only *one fixed pair of codes for users 1 and 2, and a single quantization code*, which possesses many practical advantages over traditional time sharing and superposition coding solutions and provides strictly better performance.**

***Index Terms*—Broadcast channels (BCs), low-density parity-check (LDPC) codes, iterative decoding, dirty paper coding (DPC), time-varying channels**

I. Introduction

In a broadcast channel (BC), a transmitter sends independent information streams for multiple receivers, respectively, which is a natural model of many practical systems. For example, digital video broadcasting for hand-held devices (DVB-H) is used to transmit information to multiple customers over a common channel [1]. More explicitly, an independent and identically distributed (i.i.d.) BC with two receivers and additive noises is modeled as

$$Y_1 = X + N_1 \text{ and } Y_2 = X + N_2 \tag{1}$$

where X is the input random variable and Y_1 and Y_2 are the output variables of the receivers 1 and 2, respectively. N_1 and N_2 are the corresponding memoryless additive noises. When N_2 is a degraded channel of N_1, the capacity of a BC is characterized by

$$R_1 \le I(X;Y_1|U) \text{ and } R_2 \le I(U;Y_2), \tag{2}$$

for which any rate pair (R_1, R_2) is achievable if and only if (2) is satisfied for some auxiliary random variable U such that $U \to X \to (Y_1, Y_2)$ forms a Markov chain [2]. Superposition coding was proposed in [3] to achieve the above capacity. An important ingredient of superposition coding is the use of a code with non-uniform *a priori* bit distribution. Designing codes with non-uniform priors was first studied as trellis shaping for power saving (and thus throughput enhancement) in [4].

The concept of superposition coding for the BC is highly related to the recent study of dirty paper coding (DPC). For the Gaussian BC, [5] showed that we can use DPC to remove the potential interference of the second user from the first user's perspective. DPC was introduced in [6], which is based on the general formula with noncausal side information [7]. [6] showed a surprising fact that the capacity of a Gaussian channel where the transmitter knows the interference non-causally is the same as the capacity of the corresponding interference-free channel. However, these theoretical results do not directly translate to practical systems achieving the capacity. [8] suggested a practical solution employing the structured binning schemes and *nested codes*. Based on nested lattice codes, [9] designed a new capacity-achieving Gaussian DPC scheme using vector quantization and systematic doping. [10] proposed a superposition-based solution for both binary and Gaussian DPC systems. Furthermore, [11] proposed a near-capacity Gaussian DPC scheme, the achievable noise threshold of which is 0.63dB away from the capacity at 0.25 bit per symbol based on trellis coded quantization. Also, [12] designed nested turbo codes for Gaussian DPC in the high-rate regime.

Since DPC is the footstone of numerous theoretical studies of the Gaussian multiple-input multiple-output (MIMO) BC, most work on designing practical solutions for the MIMO BC is based on DPC. [13] showed that DPC achieves the sum capacity of a Gaussian BC with two receivers. [14], [15] generalized the result to vector BCs with an arbitrary number of users and antennas. [16] designed a DPC-based code for the Gaussian MIMO BC based on nested turbo codes. Recently, [17] proved that DPC can indeed achieve the full capacity region of the Gaussian MIMO BC. DPC also has a variety of

This work was supported in parts by Purdue PRF XR COEUS-07127683, 08126798 and NSF grant CCF-0845968. Parts of this paper were presented at the IEEE WCNC, Budapest, Hungary, April 5-8, 2009.

G. B. Kyung and C.-C. Wang are with the Center for Wireless Systems and Applications (CWSA), the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA (e-mail: {gkyung, chihw}@purdue.edu).

applications in information hiding and watermarking, both of which focus on the setting of the *binary symmetric channel* (BSC).[1] The optimal achievable rate for binary DPC is proved in [18], [19]. Binary DPC is also known as a special case of Gelfand-Pinsker coding [7] or BSC-based channel coding with interference side information at the transmitter.

Most work on the binary BC has focused on the following problem. Given the crossover probabilities p_1 and p_2 of the binary noises N_1 and N_2, how can we achieve the ultimate capacity R_1 and R_2 in (2)? On the other hand, suppose the channel quality of the BC changes over time. An alternative question in practice is thus *given the desired rates R_1 and R_2, how to design a system that can sustain the traffic demand while reacting to a wide range of different channel qualities.* Since different noise levels generally require quantization codes of different rates, the varying channel quality presents a new challenge for the BC code design.

In this paper, we first propose a practical scheme for binary DPC based on coset binning. This new scheme allows us to design flexibly the system parameters such as information/quantization code rates in order to meet the power and data rate requirements of DPC. Simulation results show that the performance of the proposed system is similar to the existing superposition-coding-based solution while admitting lower complexity and better design flexibility. We then extend our DPC scheme and design accordingly a practical binary broadcast scheme, termed soft DPC, which achieves a large percentage of the capacity for binary BCs of varying channel quality with little complexity overhead. Our scheme possesses several practical advantages over the existing superposition-coding-based schemes. For example, the proposed scheme requires a single quantization code and one pair of error control codes of fixed rate/length.

The remainder of this paper is organized as follows. Section II provides an overview of binary DPC and BCs. In Section III, we present the block diagram and the factor graph of the proposed practical binary DPC scheme and the corresponding encoding and decoding mechanisms. In Section IV, we describe a binary BC scheme for varying channel quality based on the concept of *soft DPC*. Section V discusses the detailed DE- and EXIT-chart-based code design. The performance of the proposed schemes is verified in Section VI. Section VII concludes the paper.

II. Analysis of Binary DPC and Binary Symmetric BC

A. Binary DPC

Consider i.i.d. binary additive noise $\boldsymbol{n} = (n_1, n_2, \cdots, n_N)$ with crossover probability p, where N is the codeword length. The signal at the receiver is $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{s} + \boldsymbol{n} = (y_1, y_2, \cdots, y_N)$, where $\boldsymbol{x} = (x_1, x_2, \cdots, x_N)$ is the transmitted signal and $\boldsymbol{s} = (s_1, s_2, \cdots, s_N)$ is a binary interference vector that is known to the sender but unknown to the receiver. A DPC encoder sends $\boldsymbol{x} = f(m, \boldsymbol{s})$, which is a function of the interference $\boldsymbol{s}$ and the information message $m \in \{1, 2, \cdots, 2^{NR}\}$ where R is the code rate. The goal of binary DPC is to optimize the transmission rate R subject to a normalized Hamming weight constraint W on $\boldsymbol{x}$. In [18], [19], the capacity R^* for binary DPC is shown to be the following:

$$R^* = \begin{cases} h(W) - h(p) & \text{if } W_0 \le W \le 1/2 \\ \alpha W & \text{if } 0 \le W \le W_0 \end{cases} \tag{3}$$

where $h(p)$ is the binary entropy function, $W_0 = 1 - 2^{-h(p)}$, and $\alpha = \log((1 - W_0)/W_0)$.

The capacity R^* can be achieved by the following *random binning* scheme. Let $\mathcal{C}_0$ and $\mathcal{C}_1$ be two randomly chosen codes, among which we term $\mathcal{C}_0$ the quantization code and term $\mathcal{C}_1$ the information-bearing code. Let R_0 and R_1 be the code rates for $\mathcal{C}_0$ and $\mathcal{C}_1$, respectively. First, the message $m \in \{1, 2, \cdots, 2^{NR_1}\}$ is mapped to a codeword $\boldsymbol{c}_1 \in \mathcal{C}_1$. For each $\boldsymbol{c}_1$, $\boldsymbol{c}_1 + \mathcal{C}_0$ forms a bin. Given the interference $\boldsymbol{s}$, the encoder finds $\boldsymbol{c}_0 \in \mathcal{C}_0$ such that $\boldsymbol{c}_1 + \boldsymbol{c}_0$ is the closest to $\boldsymbol{s}$. Then, we transmit the difference $\boldsymbol{x} = \boldsymbol{c}_0 + \boldsymbol{c}_1 - \boldsymbol{s}$. The received signal thus becomes $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{s} + \boldsymbol{n} = \boldsymbol{c}_0 + \boldsymbol{c}_1 + \boldsymbol{n}$. With W being the normalized Hamming weight constraint on $\boldsymbol{x}$, the code rate R_0 of $\mathcal{C}_0$ must satisfy [2]:

$$R_0 > 1 - h(W). \tag{4}$$

At the receiver, the decoder estimates $\hat{m}$ by finding a pair of codewords $\hat{\boldsymbol{c}}_0 \in \mathcal{C}_0$ and $\hat{\boldsymbol{c}}_1 \in \mathcal{C}_1$ such that $\hat{\boldsymbol{c}}_1 + \hat{\boldsymbol{c}}_0$ is the closest to $\boldsymbol{y}$. The resulting $\hat{\boldsymbol{c}}_1$ is then mapped back to $\hat{m}$. By choosing an R_0 sufficiently close to the lower bound $1 - h(W)$, the achievable rate R_1 can be made arbitrarily close to $R^* = h(W) - h(p)$. This scheme thus achieves the capacity for the case in which $W_0 \le W \le 1/2$. For the case in which $0 \le W \le W_0$, time-sharing is used to achieve the DPC capacity.

B. Binary Symmetric BC

We consider the binary symmetric BC. That is, the additive noises N_1 and N_2 in (1) are Bernoulli distributed with crossover probabilities p_1 and p_2, respectively. Based on (2) the capacity region of the binary symmetric BC (assuming $p_1 < p_2$) is

$$R_1 \le h(W * p_1) - h(p_1) \text{ and } R_2 \le 1 - h(W * p_2) \tag{5}$$

where the $*$ operation is defined by $p * q \triangleq p(1 - q) + (1 - p)q$. Two different classes of capacity-approaching binary BC schemes are discussed as follows.

1) Superposition Coding: Superposition coding was proposed in [3] to achieve the capacity of the BC. The transmitter has two codebooks $\tilde{\mathcal{C}}_1$ and $\tilde{\mathcal{C}}_2$ for each receiver, respectively. m_1 and m_2 are encoded as $\tilde{\boldsymbol{c}}_1 \in \tilde{\mathcal{C}}_1$ and $\tilde{\boldsymbol{c}}_2 \in \tilde{\mathcal{C}}_2$ independently where m_1 and m_2 denote the messages intended for receivers 1 and 2, respectively. Then, the transmitter sends $\boldsymbol{x} = \tilde{\boldsymbol{c}}_1 + \tilde{\boldsymbol{c}}_2$ over the channel. Since receiver 1 is facing a better channel than receiver 2, receiver 1 can decode m_2 first and subtract $\tilde{\boldsymbol{c}}_2$ from the received signal $\boldsymbol{y}_1$. m_1 can then be decoded from the subtracted signal. Receiver 2 can find m_2 directly from the received signal $\boldsymbol{y}_2$. An important ingredient of the superposition coding approach is that we require the code $\tilde{\mathcal{C}}_1$ to be of non-uniform *a priori* distribution.

[1] The BSC also closely models the scenario in which the decoder outputs the hard decision value based on the channel observation.

2) DPC-based scheme: An alternative scheme for binary BC is based on the following DPC-based construction:

$$\tilde{\boldsymbol{c}}_2 = \boldsymbol{c}_2 \tag{6}$$

Choose $\boldsymbol{c}_0 \in \mathcal{C}_0$ such that $\boldsymbol{c}_1 + \boldsymbol{c}_0$ is the closest to $\tilde{\boldsymbol{c}}_2$ (7)

$$\tilde{\boldsymbol{c}}_1 = \boldsymbol{c}_1 + \boldsymbol{c}_0 - \tilde{\boldsymbol{c}}_2 \tag{8}$$

$$\boldsymbol{x} = \tilde{\boldsymbol{c}}_1 + \tilde{\boldsymbol{c}}_2 = \boldsymbol{c}_1 + \boldsymbol{c}_0. \tag{9}$$

First, m_1 and m_2 are encoded into $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$. Then in (6) and (7), we treat the information-bearing codeword $\boldsymbol{c}_2$ as interference $\tilde{\boldsymbol{c}}_2$ and use a quantizer to find $\boldsymbol{c}_0 \in \mathcal{C}_0$ such that $\boldsymbol{c}_1 + \boldsymbol{c}_0$ is the closest to $\tilde{\boldsymbol{c}}_2$. We then construct a new codeword $\tilde{\boldsymbol{c}}_1$ by (8) and transmit the summation of $\tilde{\boldsymbol{c}}_1$ and $\tilde{\boldsymbol{c}}_2$ as in (9).

To decode $\boldsymbol{c}_2$ at receiver 2, we notice that the use of a quantization codeword $\boldsymbol{c}_0$ ensures that $\boldsymbol{c}_0 + \boldsymbol{c}_1$ is within a distance W from the codeword $\boldsymbol{c}_2$. Therefore, equivalently, receiver 2 can decode $\boldsymbol{c}_2$ as if facing an i.i.d. BSC with crossover probability $W * p_2$. There are two ways of decoding $\boldsymbol{c}_1$ at receiver 1: successive canceling decoding and the interference-oblivious DPC decoding. With successive canceling decoding, receiver 1 decodes $\tilde{\boldsymbol{c}}_2$ first and subtracts it from $\boldsymbol{y}$. With DPC decoding, receiver 1 does not decode $\tilde{\boldsymbol{c}}_2$ but directly decodes $\boldsymbol{c}_1$ from $\boldsymbol{y} = \boldsymbol{c}_1 + \boldsymbol{c}_0 + \boldsymbol{n}_1$.

Note that from the transmitter side, both DPC and superposition coding need to use a quantization code and thus have comparable complexity. However, by performing DPC encoding at the transmitter (instead of superposing coding), receiver 1 has the freedom to choose between DPC decoding and successive canceling decoding. When receiver 1 is able to afford higher complexity, successive decoding leads to higher achievable throughput. When receiver 1 is limited in terms of computation power (such as the case for the hand-held devices) or when the channel quality faced by receiver 1 is much better than the decodable threshold, we perform DPC decoding for its simplicity. Also, with DPC decoding, if desired, we can also maintain the privacy of user 2 by not revealing the codebook of user 2 to receiver 1.

III. Binary DPC—the Proposed Practical System

A. Practical Binary DPC and the Corresponding Challenges

When implementing the two codes $\mathcal{C}_0$ and $\mathcal{C}_1$ for binary DPC (see Section II-A) by the off-the-shelf binary convolutional and LDPC codes, there is an initialization problem due to the usage of traditional bit-wise belief propagation (BP) decoder. Recall that with DPC encoder, the end user receives $\boldsymbol{y} = \boldsymbol{c}_0 + \boldsymbol{c}_1 + \boldsymbol{n}$, where $\boldsymbol{c}_0$ and $\boldsymbol{c}_1$ are the chosen convolutional/LDPC codewords. Since the bit distributions of both codes $\mathcal{C}_0$ and $\mathcal{C}_1$ are uniform on $\{0, 1\}$, the BP decoder cannot extract any information from $\boldsymbol{c}_0 + \boldsymbol{c}_1 + \boldsymbol{n}$ even in a noise-free channel (when $\boldsymbol{n} = \boldsymbol{0}$). The reason is that from a bit perspective, the uniformly distributed $\boldsymbol{c}_0$ is equivalent to a random noise with crossover probability $p = 0.5$. Therefore, no bit-based BP decoder can extract any information for $\boldsymbol{c}_1$. Similarly, $\boldsymbol{c}_1$ is equivalent to a $p = 0.5$ noise for $\boldsymbol{c}_0$. As a result, neither of the individual decoders for $\boldsymbol{c}_0$ and $\boldsymbol{c}_1$ can be initialized, let alone the joint iterative decoding loop. It is worth noting that if an optimal maximum *a posteriori* probability (MAP) decoder is used, one can extract the information of $\boldsymbol{c}_1$ from the received signal $\boldsymbol{c}_0 + \boldsymbol{c}_1$. Initialization problem is caused by the suboptimality of bit-based BP decoder. In this section, we propose to solve this problem by introducing *edge erasing* when combining $\boldsymbol{c}_0$ and $\boldsymbol{c}_1$.

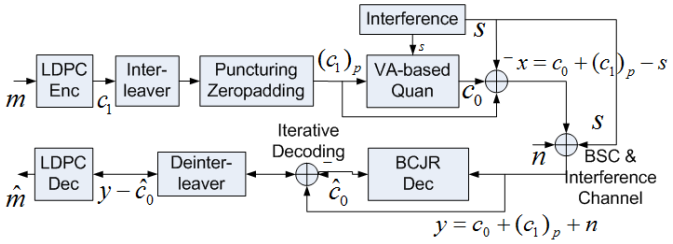


Fig. 1. The block diagram of the proposed binary DPC system.

B. System Overview

We propose a practical binary DPC scheme based on the concept of coset binning. Fig. 1 shows the corresponding block diagram. Based on the information message m, the LDPC encoder first chooses $\boldsymbol{c}_1$. We use a random interleaver between the LDPC encoder and Viterbi-algorithm-based (VA-based) quantizer, which reduces the dependence of two encoders. Then, $\boldsymbol{c}_1$ is punctured randomly by a fixed portion e and is padded with zeros in those punctured positions. This puncturing is the key element in the proposed system, which will be elaborated shortly after. Let $(\boldsymbol{c}_1)_p$ be the punctured and zero-padded sequence. $(\boldsymbol{c}_1)_p$ and the known interference $\boldsymbol{s}$ are fed into the VA-based quantizer, which chooses $\boldsymbol{c}_0$ such that $\boldsymbol{c}_0 + (\boldsymbol{c}_1)_p - \boldsymbol{s}$ is closest to zero in terms of the Hamming weight. This procedure of the VA-based quantizer is similar to trellis shaping in [4]. The transmitter then sends $\boldsymbol{x} = \boldsymbol{c}_0 + (\boldsymbol{c}_1)_p - \boldsymbol{s}$. Due to the interference $\boldsymbol{s}$ and the binary symmetric noise $\boldsymbol{n}$ with crossover probability p, the received sequence is $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{s} + \boldsymbol{n} = \boldsymbol{c}_0 + (\boldsymbol{c}_1)_p + \boldsymbol{n}$. At the receiver, iterative decoding is performed between the BCJR decoder and the LDPC decoder, which will be discussed in detail in the next subsection.

C. The Factor Graph of the Proposed Scheme

The joint iterative decoder of the proposed binary DPC scheme is described by the corresponding factor graph [20] of the proposed scheme. In our simulation, we use the QC-LDPC code for its implementation advantages, including fast encoding and structured interconnection. Fig. 2 depicts the factor graph of our system. We use blank circles to denote the information variable nodes and the black circles to denote the parity variable nodes. In addition, the squares represent the check nodes. The interleaver Π_1 connects the information variable nodes and the check nodes and Π_2 connects the parity variable nodes and the check nodes. We use the bit values of the parity variable nodes as our $\boldsymbol{c}_1$. One can also view our system as a nonsystematic code. The encoded parity bits $\boldsymbol{c}_1$ are then permuted by interleaver Π_3 and connected to the bits of the convolutional code. Since some positions of $(\boldsymbol{c}_1)_p$ are zero-padded, it is equivalent to having some check nodes between

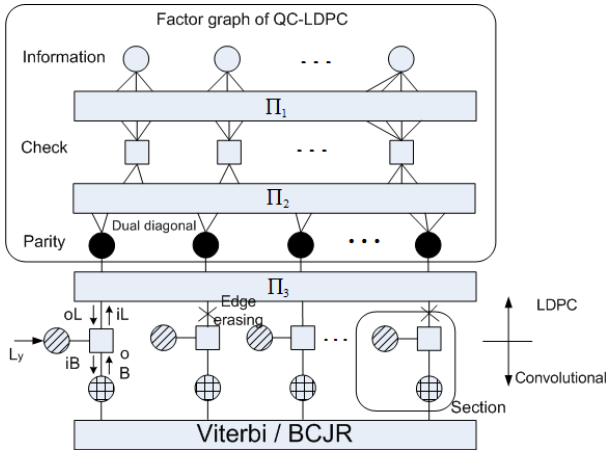


Fig. 2. The factor graph of our system. iL is the inbound LLR message to the LDPC parity nodes and oL is the outbound message from the LDPC parity nodes. Similarly, iB (oB resp.) is the inbound (outbound resp.) LLR message to (from resp.) the BCJR decoder.

the LDPC code and the convolutional code being disconnected from Π_3 in Fig. 2. We call this operation *edge erasing*.

In Fig. 2, the circles with slashes represent $\boldsymbol{y}$. Let I be the set of bits that are punctured. We then notice that the overall received symbol y_i is $y_i = c_{0i} + 1_{\{i \notin I\}} c_{1i} + n_i$ where c_{0i}, c_{1i}, and n_i are the i-th components of $\boldsymbol{c}_0$, $\boldsymbol{c}_1$, and $\boldsymbol{n}$, respectively, and $1_{\{\cdot\}}$ is the indicator function. In addition, let us define some log-likelihood ratio (LLR) edge messages as follows. Let iL be the inbound LLR message entering the LDPC parity nodes and let oL be the outbound message originating from the LDPC parity nodes. Similarly, let iB (resp. oB) be the inbound (resp. outbound) LLR message to (resp. from) the BCJR decoder. The received values of the i-th component y_i is used to compute the LLR message $L_{y_i} = (-1)^{y_i} \log \frac{1-p}{p}$.

The joint iterative decoding starts from the BCJR decoder. For those $i \notin I$, the corresponding iB is zero regardless of the input LLR message L_{y_i}. The reason is that the corresponding oL is initialized to zero and when any input LLR message of a check node is zero, the LLR output of the check node is always zero. On the other hand, for those $i \in I$, we have $iB = L_{y_i}$ due to edge erasing. Those iB on $i \in I$ are thus used to initialize the BCJR decoder. After the first iteration of the BCJR decoder, the extrinsic information oB is delivered to the BP decoder. For those positions $i \notin I$, iL is generated from L_{y_i} and oB using the standard check node LLR message map [20]. iL plays a role as the *a priori* information to the BP decoder. For those $i \in I$, we simply set $iL = 0$. After a certain number[2] of iterations of BP decoding based on the input iL, the BP decoder generates the extrinsic information oL, which is transmitted back to the BCJR decoder. iB can then be computed from L_{y_i} and oL for the positions $i \notin I$. The BCJR decoder can now start the second round of decoding based on the new iB for those $i \notin I$ and the L_{y_i} for those $i \in I$.

[2]This number can be chosen adaptively as will be explained in Section V.

D. Comparison to the Existing Works

For comparison, we briefly describe the existing superposition-coding-based binary DPC scheme in [10]. In [10], a non-uniformly-distributed code $\boldsymbol{c}_1$ is constructed by concatenating a q-ary LDPC code with a symbol mapper such that symbols 0 to $q-k-1$ are mapped to 0 while symbols $q-k$ to $q-1$ are mapped 1 for some predetermined k. This way, the resulting code will have non-uniform bit distribution k/q. Then [10] chooses the quantization codeword $\boldsymbol{c}_0$ by the VA-based quantizer. For any BSC with cross-over probability p and weight constraint W, [10] selects the same R_0 as in our scheme (satisfying (4)). Nonetheless, to achieve the DPC capacity using a non-uniform $\mathbf{GF}(q)$ code $\boldsymbol{c}_1$, one has to choose the values of q and k such that they jointly satisfy $(k/q) * p = W$ [10]. For example, if a $\mathbf{GF}(8)$ LDPC code is used, then when $p = 0.1$, the possible $W = (k/q) * p$ values one can use are $W = 0.2$ ($k/q = 1/8$), $W = 0.3$ ($k/q = 2/8$), and $W = 0.4$ ($k/q = 3/8$). To flexibly support different W values such as $W = 1/3$ or $W = 1/4$, a high-order $\mathbf{GF}(q)$ has to be used, which significantly increases the complexity of the system. For comparison, our proposed scheme uses only binary LDPC codes, which admits significant complexity advantages over any non-binary LDPC codes.

In contrast with the BSC model considered in this work, [9] focuses on the Gaussian channel DPC problem based on the similar techniques of trellis shaping and joint iterative decoding. From the decoding's perspective, the aforementioned initialization problem is unique to the BSC and does not exist in the Gaussian channel model. The proposed DPC scheme addresses the decoder initialization problem by a new structured code design that is optimized by a joint DE and EXIT chart analysis.

IV. The Proposed Binary Broadcast Scheme

A. Binary BC with Varying Channel Quality

Motivated by the new binary DPC scheme in the previous section, we also consider the following binary BC problem with varying channel quality. Namely, consider a binary BC with 2 receivers as described in (1). Let R_0, R_1, and R_2 be the code rates of $\mathcal{C}_0$, $\mathcal{C}_1$, and $\mathcal{C}_2$ in (6)–(9). Similar to the DPC problem, to achieve the BC capacity (5), the quantization code rate R_0 has to satisfy (4) in order to let $\mathcal{C}_1$ have non-uniform bit distribution $Bernoulli(W)$. We use the following example to illustrate the challenges of a practical binary BC scheme. To that end, we first observe that most existing works focus on achieving the optimal rates R_1 and R_2 given the crossover probabilities p_1 and p_2. In contrast, we are interested in finding the range of p_1 and p_2 that can sustain a given rate pair (R_1, R_2).

From (3)–(5), different channel quality requires the $\tilde{\mathcal{C}}_1$ code to be of different *a priori* distributions, which in turns requires several quantization codes of different rates. Consider a fixed rate pair $R_1 = 0.36$ and $R_2 = 0.125$ for example. The corresponding capacity curve is illustrated in terms of p_1 and p_2 in Fig. 3. However, infinitely many R_0's may be necessary to achieve the capacity region, which is not possible in practice. For such a given rate pair, if we use $R_0 = 1/3$,

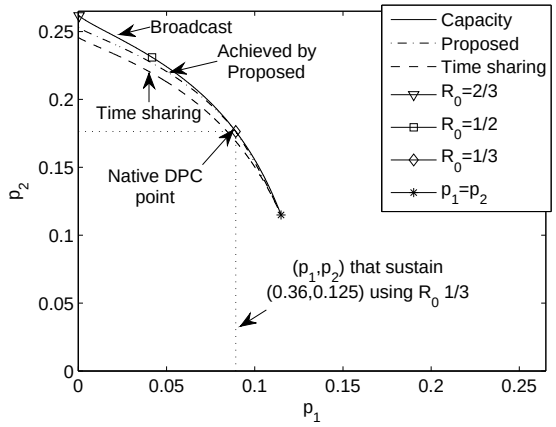


Fig. 3. The capacity of the binary symmetric BC ($R_1 = 0.36$, $R_2 = 0.125$)

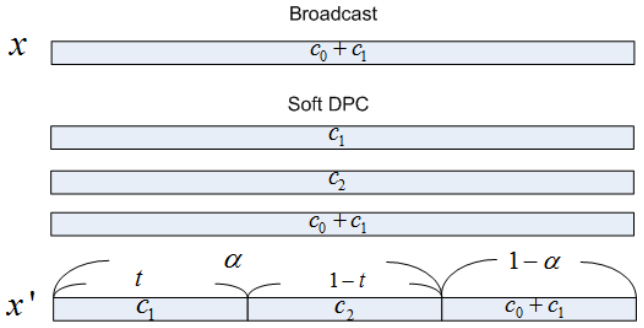


Fig. 4. The transmitted signal $\boldsymbol{x}$ of the binary broadcast system and $\boldsymbol{x}'$ of soft DPC.

then rate (0.36,0.125) can be sustained if $p_1 \leq 0.0892$ and $p_2 \leq 0.1764$ (the dotted rectangle of Fig. 3). In Fig. 3, two more achievable points for $R_0 = 2/3$ and $1/2$ are depicted. If we are allowed to use multiple quantization codes $\mathcal{C}_0$ with different R_0 values, the achievability region is thus the union of multiple rectangles. Since quantization codes of different rates are required to achieve the capacity with varying channel quality, it is thus necessary to design rate-compatible quantization codes. However, the common choices of five different rates $1/6, 2/6, \cdots, 5/6$ for rate-compatible codes generally do not have the fine granularity necessary for achieving the capacity region in Fig. 3 (there still exist holes in the achievable region). Since in practice, it is difficult to design a rate-compatible quantization code with the desired granularity, our goal is thus given a small number of codebooks (say one for codes $\mathcal{C}_0$, $\mathcal{C}_1$, and $\mathcal{C}_2$, respectively), how to design the system such that we can achieve a large percentage of the capacity for a broad range of channel parameters. In this work, we assume that p_1 and p_2 change after each time-frame but the channel side information (p_1, p_2) is fed back to the transmitter at the beginning of each time frame. Within the same time frame, we assume the channel quality (p_1, p_2) remains constant.

Another possible approach to achieve the desired rates R_1 and R_2 is to use time sharing without any superposition coding or DPC techniques. The capacity curve of time sharing is plotted in Fig. 3 as the dashed curve. Time sharing covers a large area of the optimal capacity. Nonetheless, given two codes $\mathcal{C}_1$ and $\mathcal{C}_2$, we have to puncture the $\mathcal{C}_1$ and $\mathcal{C}_2$ codes heavily, which generally results in bad performance. Another way of performing time sharing is to construct multiple codes of different rates/lengths from scratch (instead of just puncturing a single mother code) and carefully select the individual codes used in a single frame of transmission, which may easily lead to too many combinations for practical implementation. For example, if the time sharing coefficient is $t = 0.5$, to achieve the desired rate pair (0.36, 0.125), we can combine two $R_1 = 0.36$ and $R_2 = 0.125$ codes with 1/2 puncturing or use $R_1 = 0.72$ and $R_2 = 0.25$ codes with the codeword length being $N/2$ respectively. The complexity of using multiple codes increases when a large number of codes are considered. Moreover, there still exist holes in the achievable rate region if we do not have sufficiently many combinations of rates/lengths.

B. The Proposed Scheme - Soft DPC

In this subsection, we show how to extend the proposed practical DPC scheme for a binary BC with varying channel. The proposed scheme uses a single fixed quantization code $\mathcal{C}_0$ and a fixed pair of information-bearing codes $\mathcal{C}_1$ and $\mathcal{C}_2$. The corresponding code rates are (R_0, R_1, R_2), respectively. Note that when a classic BC scheme is used, a given rate tuple, say $(R_0, R_1, R_2) = (1/3, 0.36, 0.125)$, can only achieve the dotted rectangle region as shown in Fig. 3. We will present a new concept, soft DPC, to cover both the right-hand side and the left-hand side of the single achievable point, respectively. For easier reference, given the rate tuple (R_0, R_1, R_2), the (p_1, p_2) vector achievable by the classic DPC-based BC scheme is termed the *native DPC point*.

The main idea is a new three-way time sharing between the native DPC point and the individual codewords $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, respectively. We term this technique *soft DPC*. More specifically, we perform time sharing among three different codewords: $\boldsymbol{c}_1$, $\boldsymbol{c}_2$, and $\boldsymbol{c}_0 + \boldsymbol{c}_1$ as in Fig. 4. In Fig. 4, we use α to denote the time sharing coefficient between the native DPC codeword $\tilde{\boldsymbol{c}}_1 + \tilde{\boldsymbol{c}}_2 = \boldsymbol{c}_0 + \boldsymbol{c}_1$ and the pure information-bearing codewords $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, and we use t as the time sharing coefficient that further splits the usage of $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$. The achievable rate region of such a soft DPC scheme can be computed from (5):

$$\begin{aligned} R_1 &\leq \alpha t(1 - h(p_1)) + (1-\alpha)(h(W * p_1) - h(p_1)), \\ R_2 &\leq \alpha(1-t)(1 - h(p_2)) + (1-\alpha)(1 - h(W * p_2)) \end{aligned} \tag{10}$$

where $0 \leq t \leq 1$ and $0 \leq \alpha \leq 1$. To cover the right-hand side of the native DPC point (the point corresponding to $(R_0, R_1, R_2) = (1/3, 0.36, 0.125)$ in Fig. 3), we choose a fixed $t = t_0 = R_1/(R_1+R_2)$. With this t_0, varying the α value enables the soft DPC scheme to achieve the capacity for both the cases (i) when $p_1 = p_2$ and (ii) when the native DPC point is considered. Varying α, we can finely tune the sustainable (p_1, p_2) values and cover a large portion of the right-hand side of the native DPC point. The achievable region of soft DPC is illustrated in Fig. 3, for which the right-hand side of the native

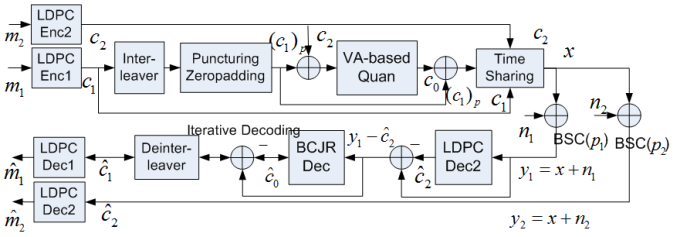


Fig. 5. The block diagram of the practical binary broadcast system.

(p_1, p_2)	(α, t)	(p_1, p_2)	(α, t)
$(0.030, 0.192)$	$(0.2, 0)$	$(0.071, 0.141)$	$(0.2, t_0)$
$(0.045, 0.174)$	$(0.1, 0)$	$(0.081, 0.127)$	$(0.4, t_0)$
$(0.059, 0.040)$	$(0, 0)$		

TABLE I
α AND t ACCORDING TO DIFFERENT CHANNEL QUALITY (p_1, p_2) WHEN $R_0 = 1/3$, $R_1 = 0.36$, AND $R_2 = 1/8$.

DPC point is nearly indistinguishable from the BC capacity region. A broad area is thus covered with a fixed set of $\mathcal{C}_0$, $\mathcal{C}_1$, and $\mathcal{C}_2$.

To achieve the capacity curve that is on the left of the native DPC point in Fig. 3, we need a smaller weight W (achieved by a larger R_0) in the classic BC schemes. We approximate the effect of using a smaller W by the following choice of the soft DPC parameters α and t. In particular, we hardwire the t value to zero and adjust the α value. As a result, the output of the soft DPC scheme $\boldsymbol{x}'$ is now a time-sharing version that combines α portion of $\tilde{\boldsymbol{c}}_2$ and $(1-\alpha)$ portion of $\boldsymbol{c}_0 + \boldsymbol{c}_1$. We notice that $\boldsymbol{c}_0 + \boldsymbol{c}_1$ is the traditional native DPC codeword that is of distance W away from the "interference codeword $\boldsymbol{c}_2$." Therefore, the new soft DPC codeword $\boldsymbol{x}'$ is now an in-between signal that is of distance $(1-\alpha)W$ apart from $\tilde{\boldsymbol{c}}_2$. In a broadest sense, such time sharing coefficients reduce the weight from W to $(1-\alpha)W$, which gives the name *soft DPC*. The achievable rate of soft DPC with $t = 0$ is shown with dashdotted line on the left of the native DPC point in Fig. 3. We note that this scheme is not rate-optimal but achieves a great percentage of the capacity with small complexity. We also notice that the largest rate-deficiency of soft DPC happens at the left end of the capacity curve, namely, the achievable rate of soft DPC is the farthest away from the BC capacity when channel one is almost error free ($p_1 \approx 0$) and the channel p_2 is the noisiest. In practice, this extremely imbalanced realization of varying channel quality occurs only infrequently.

C. Implementation and Operation

The block diagram of the proposed soft-DPC-based BC scheme is provided in Fig. 5. An LDPC code is combined with a VA-based quantizer through an interleaver. Similar to the binary DPC scheme (Fig. 1), we rely on puncturing (or equivalently edge erasing) to address the initialization problem of the decoder. The information-bearing codeword of user 2, denoted by $\boldsymbol{c}_2$, takes over the role of the interference $\boldsymbol{s}$ in the previous DPC system (Fig. 1). A new block of 3-way time sharing is used to assemble soft-DPC codeword from $\boldsymbol{c}_0 + (\boldsymbol{c}_1)_p$, $\boldsymbol{c}_1$, and $\boldsymbol{c}_2$.

In practice, given a rate tuple (R_0, R_1, R_2) (according to the available hardware) and the cross-over probabilities (p_1, p_2) (according to the channel quality feedback), we would like to choose the parameter α and t that ensure successful transmission. To that end, the system designer first constructs a table similar to Table I for different α and t values. In Table I, the (p_1, p_2) column corresponds to the decodable thresholds of the soft-DPC scheme using the given convolutional and LDPC codes with the specified (α, t) values. Once such table is established, we choose suitable (α, t) from the table for any given (p_1, p_2) that is feedback from the receivers. If (p_1, p_2) is not covered in the table, we can interpolate the desired α and t values to cover the new p_1 and p_2 values. If the current (p_1, p_2) value cannot be covered by any combinations of (α, t) parameters, we either reduce the throughput by considering a smaller rate pair (R_1, R_2) supported by the hardware implementation, or when there is no lower-rate code available we declare outage.

V. System and Code Optimization

A. High-Level Description of the Code Optimization

We perform code optimization by combining both DE [21] and the EXIT chart [22]. We use DE to analyze the performance on the LDPC-code side, which tracks LLR distributions in the BP decoder and is used to design the degree distributions of the LDPC code. The EXIT chart is used to analyze the BCJR decoder and the joint iteration between the BCJR and BP decoders. The erasure percentage $e = |I|/N$ is critical to the performance of the proposed system. A large value e results in better initialization of the BCJR decoder as more bits are assigned with nonzero LLR (those with $iB = L_{y_i}$). However, a large e value also prevents the LDPC decoder from receiving decoded soft bit values from the BCJR decoder. A balance needs to be struck between these two effects.

Our goal is to jointly optimize $(\lambda(x), \rho(x))$ and e in order to maximize the threshold p^*, where $(\lambda(x), \rho(x))$ is the degree distribution pair of an LDPC code [23]. A high-level description of the optimization steps is as follows. First of all, given a convolutional code and an LDPC code with the degree distributions $(\lambda(x), \rho(x))$, we use the corresponding BCJR and BP decoders to generate the corresponding EXIT curves for several combinations of p and e. For each e, we compute the corresponding threshold p^* by choosing the largest p such that two EXIT curves do not meet each other. Then we adjust the e value, repeat the EXIT chart computation, and find the optimal e^* that leads to the largest p^*. Fix the value of e^*, we then perform joint DE and EXIT analysis to optimize the degree distributions of the LDPC code. Once the degree distributions of a given e^* are optimized, we fix the optimized degree distributions and re-optimize e by choosing the best e as described in the beginning of this paragraph. In our simulation, the performance improvement is generally small after the first round of iteration and we stop optimization quickly.

Remark: When considering binary BCs, the above code optimization can be repeated for any given pair of time-sharing coefficients (α, t). However, since changing the degree

distributions $(\lambda(x), \rho(x))$ is likely to drastically change the implementation, we thus only optimize the degree distributions $(\lambda(x), \rho(x))$ and the erasure portion e for the case of $(\alpha, t) = (0, 0)$ and reuse the same codes when different (α, t) are used. If the decoder hardware is fully programmable, then we can also optimize the code structure for every (α, t) values to attain even greater thresholds.

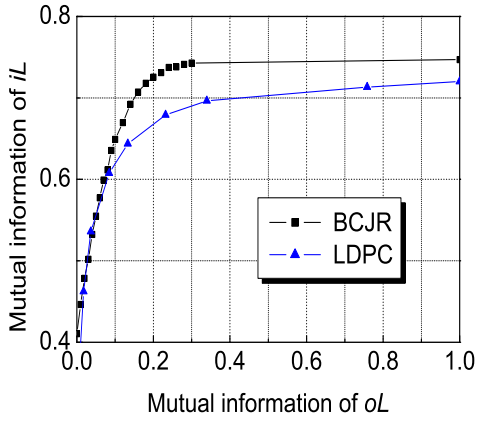
B. The EXIT Chart

To generate the EXIT chart, we use oL and iL as the input and the output of the EXIT chart that describes the convergence of the system. To generate the BCJR EXIT curve, we assume oL being Gaussian distributed with mean and variance $(\mu, 2\mu)$ for some μ. (As also verified in our simulation, even with the channel model being a BSC, the distribution of oL can still be approximated as Gaussian.) Then we plot the BCJR EXIT curve according to [22]. More specifically, for any fixed μ, we use Monte-Carlo simulation to generate many oL and observations $\boldsymbol{y}$ and use the empirical distribution of the resulting iL to compute the mutual information at the output LLR iL of the BCJR decoder. By varying μ and repeating the above procedure, we have different (oL, iL) points that are used to plot the BCJR EXIT curve in Fig. 6. For the LDPC EXIT curve, we use the mutual information of iL to generate a Gaussian message distribution. A pre-defined number of DE iterations are performed within the LDPC-side of the decoder, which generates the probability density function of the LLR messages oL. By computing the corresponding mutual information of oL, we use the corresponding (iL, oL) to obtain the EXIT curve for the LDPC code.

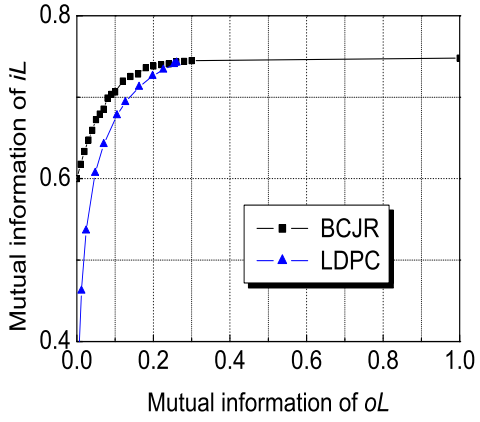
As explained previously, the EXIT chart is used to estimate the threshold p^* of a given e and we select e^* that leads to the largest p^*. To expedite the optimization, we use a greedy search in our simulation. Namely, once p^* is found for a given e, we fix that p^* and choose new e in the following way. We notice that with a fixed p^*, when e increases, both curves are moving upward because more LLR messages L_{y_i} of the BSC (those $i \in I$) are fed to the BCJR decoder but less iL messages (those with $i \notin I$) are fed to the LDPC decoder. Fig. 6 illustrates two such pairs of curves for $e = 0.45$ and $e = 0.49$ at $p = 0.06$ accordingly. With p set to p^* (of the previous e value), we choose a new e such that the two curves are the farthest apart. For example, with a fixed value $p = 0.06$ and $e = 0.45$, the two EXIT curves meet at the initial stage $(0 \leq oL \leq 0.1)$ in Fig. 6(a). The curves for $e = 0.49$ cross around $oL = 0.3$ in Fig. 6(b). Since the curves for $e = 0.47$ in Fig. 7 do not meet each other, we select new $e = 0.47$. This e is then used to obtain the corresponding threshold p^*. The iteration repeats until both e and p^* converge.

C. Optimization of the LDPC code

For the degree optimization of the LDPC code, we compute the LDPC EXIT curve by DE. That is, during our computation of the BCJR EXIT curve, we also record the empirical distribution of iL. Specifically, we record several pmfs of iL according to different μ values for the oL LLR message.



(a) $e = 0.45$



(b) $e = 0.49$

Fig. 6. The EXIT curves of the BCJR decoder and the LDPC decoder for different percentages of edge-erasing at $p = 0.06$.

(A point in the BCJR EXIT curve of Fig. 6 corresponds to one such pmf.) We then use the pmf of iL as the input of DE, perform a pre-defined number of DE iterations, and obtain the final distribution of oL by DE. By computing the mutual information of oL from the corresponding pmf, we can use the EXIT curve of BCJR (with the input being the newly computed mutual information of oL) to quickly find the new pmf of iL for the next round. In this manner, we can test efficiently that given a degree distribution, whether the above EXIT curve and DE computation converge to successful decoding within the allowable number of total iterations. We then use differential evolution [24] to optimize the degree distributions $(\lambda(x), \rho(x))$.

To verify that the accuracy of the above EXIT chart plus DE analysis, we record the LLR distributions under *real BCJR+LDPC decoding* with Monte-Carlo simulation and convert them to the mutual information. Fig. 7 plots one such trajectory of a system with $p = 0.06$ and $e = 0.47$ and block length 10^5. As can be seen, the trajectory of actual decoding fits the prediction of EXIT and DE analysis. The

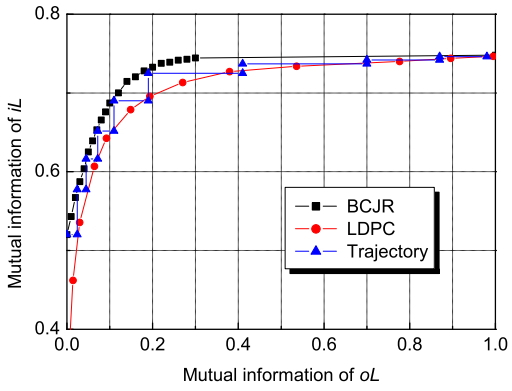


Fig. 7. The EXIT chart for the trajectory of the system at $p = 0.06$ ($e = 0.47$).

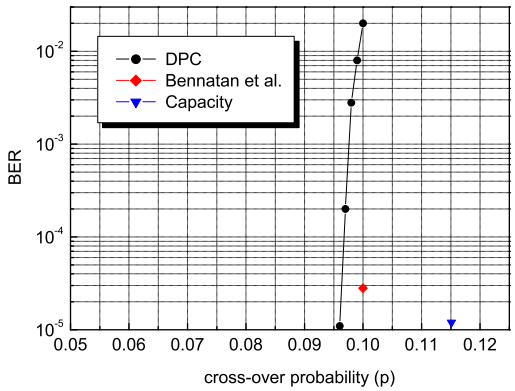


Fig. 8. The bit error rate of the proposed system.

EXIT curve can also be used to decide how many iterations one should proceed within the LDPC decoder. Namely, the number of inner iterations should be large enough so that the resulting mutual information of oL is on the right-hand side of the EXIT curve of the BCJR decoder (see Fig. 7). Usually, a small number of LDPC iterations (e.g. 10-15) are sufficient in the initial stage of the iterative decoder. However, the necessary number of iterations increases as the iterative decoding between the BCJR and the LDPC decoders proceeds, say after 4 outer iterations. By performing a small number of LDPC iterations in the initial stages and a larger number of iterations in the latter stages, we can control the total number of LDPC iterations (summing over both the initial and final stages) in our DPC system to be roughly 100-150, which is comparable to that of practical LDPC decoders for regular i.i.d. channels.

VI. Performance and Discussions

A. Binary DPC Scheme

In our simulation, we use a convolutional code as the quantization code $\mathcal{C}_0$ and an irregular QC-LDPC code [25] as the channel code $\mathcal{C}_1$, for which efficient encoding and decoding methods are well developed in [26].

For comparison, we use the same parameters that are used also in [10], which includes the code rates $(R_0, R_1) = (0.125, 0.36)$, the codeword length ($N = 10^5$), and the generator polynomials of a rate-$\frac{1}{8}$ recursive systematic convolutional code being

$$(2565, 2747, 3311, 3723, 2373, 2675, 3271, 2473). \quad (11)$$

With a quantization code rate 1/8, the smallest average Hamming weight one can hope to achieve is $W^* = 0.295$ according to (4). In our simulation, the average weight W of the above quantization code is 0.308, which demonstrates the effectiveness of the convolutional quantization code. We use an optimized binary irregular QC-LDPC code with rate 0.36, which has the degree distributions:

$$\begin{aligned} \lambda(x) &= 0.53x + 0.21x^2 + 0.01x^3 + 0.25x^9, \\ \rho(x) &= 0.2x^2 + 0.8x^3. \end{aligned} \quad (12)$$

The parity check matrix of a QC-LDPC code is expressed by the base matrix and the model matrix [26]. The base matrix has size 200×272. The first 200×72 sub-matrix corresponds to the information bits and the last 200×200 sub-matrix corresponds to the parity bits, the latter of which has almost dual diagonal structure in the parity part. We only optimize the degree distribution for the information part while keeping the dual diagonal structure of the parity bits intact. In the end, the optimized degree distributions are described in (12). The large percentage (actually 53%) of the degree-2 variable edges in $\lambda(x)$ is because of the dual diagonal structure. In our system formulation Fig. 2, we combine only the parity part of the QC-LDPC code with the convolutional code. Since the input has 72 information blocks and the output is the 200 parity blocks, the overall code rate is $\frac{72}{200} = 0.36$. The permutation matrices in the model matrix are carefully chosen to reduce the number of short cycles. The bit-error rates (BER) curve of our binary DPC scheme is compared with the results of [10] in Fig. 8. The BER of our code is comparable to their results. At BER $= 3 \times 10^{-5}$, the threshold of our system is $p^* = 0.0965$ while the superposition-coding + **GF**(4)-based system [10] achieves $p^* = 0.1$. It is worth noting that our system is based on practical binary QC-LDPC codes rather than the capacity-approaching random bipartite graph ensemble [27]. In general, the performance of QC-LDPC codes is a bit worse than that of an optimized irregular LDPC codes since QC-LDPC codes must have special structure for efficient encoding and decoding [26]. Moreover, the degree distributions of our QC-LDPC code are also coarsely quantized (see (12)) and we use very small check node degrees (≤ 4) for practical implementation. We believe that the above two issues are the cause of the small $\approx 3.5\%$ inefficiency when compared to the scheme in [10]. For comparison, the theoretical dirty-paper channel capacity of this code is $p^*_{DPC} = 0.1151$ by (3).

B. Binary BC Scheme

We present the simulation results for our practical broadcast scheme. Similar to the previous DPC scheme, we use irregular QC-LDPC codes as the channel codes $\mathcal{C}_1$ and $\mathcal{C}_2$ and a convolutional code as the quantization code $\mathcal{C}_0$. We use

a recursive systematic convolutional code with $R_0 = 1/3$ and the constraint length 11. The corresponding generator polynomial is $(2565, 2747, 3311)$.[3] For any quantization code with $R_0 = 1/3$, the minimal achievable Hamming weight (distortion) after quantization is $W^* = 0.1739$ by (4). For comparison, the simple convolutional quantization code used in the simulation achieves $W = 0.1830$. We choose $\mathcal{C}_1$ with $R_1 = 0.36$ and $\mathcal{C}_2$ with $R_2 = 1/8$. In addition, the codeword lengths of all codes are set to $N = 120000$. The sizes of the base matrices of QC-LDPC codes $\mathcal{C}_1$ and $\mathcal{C}_2$ are 200×272 and 105×120, respectively. The QC-LDPC code $\mathcal{C}_1$ is identical to the information-bearing code used in the binary DPC simulation and the corresponding dual-diagonal structure is described in Section VI.A and by (12). $\mathcal{C}_2$ has a similar dual-diagonal structure. However, unlike $\mathcal{C}_1$, in which we only send the parity bits, for $\mathcal{C}_2$ we send both the information and the parity bits. The overall rate of $\mathcal{C}_2$ is thus $\frac{120-105}{120} = 1/8$. The degree distributions of $\mathcal{C}_2$ are optimized by DE and is described as follows.

$$\begin{aligned} \lambda(x) &= 0.63x + 0.09x^2 + 0.27x^{14}, \\ \rho(x) &= 0.84x^2 + 0.16x^3. \end{aligned} \tag{13}$$

The performance curves of the proposed system are plotted in Fig. 9, which depicts the largest (p_1, p_2) values for which each receiver can successfully decode $\boldsymbol{c}_1$ (or $\boldsymbol{c}_2$) with BER $= 10^{-3}$. For comparison, we also plot the achievable region (also with required BER $= 10^{-3}$) for time sharing schemes based on heavy puncturing, and the capacity curves of the optimal BC, of the proposed soft DPC, and of time sharing in Fig. 9. The performance of the DPC-based broadcast system (without soft DPC) is highlighted as the dashed rectangle in Fig. 9. The curve with diamond markers is the performance of successive canceling decoding while the transmitter chooses to encode the packet by soft DPC. The curve with square markers is the performance for the proposed scheme with DPC decoding. It is worse than the case of successive decoding due to the capacity gap of (3) and (5) for "User 1" decoding. The curve with circle markers is the performance of time sharing. Also, the performance of our system with different code rates of $R_0 = 1/6$, $R_1 = 1/2$, and $R_2 = 1/12$ is depicted in Fig. 10. As seen in Fig. 9 and Fig. 10, a strictly broader area is covered by the proposed broadcast scheme when compared to that of time sharing.

C. Remarks on the Performance

It is worth noting that although there is a performance gap between the practical implementation and the predicted theoretical capacity, such a gap is resulted from the inherent LDPC code performance gap on BSCs rather than from the efficiency loss of the proposed scheme. For example, we have simulated the rate 1/8 code used in "user 2" simulation on a BSC environment. Even when we do not use any broadcast or DPC mechanisms, the LDPC performance of $\mathcal{C}_2$ in the pure BSC has the decodable noise threshold being $p = 0.265$

[3]In the binary DPC simulation, to have a fair comparison we use the same convolutional code as the one used in [10]. For our binary BC results, we use a simpler convolutional code for better practicality.

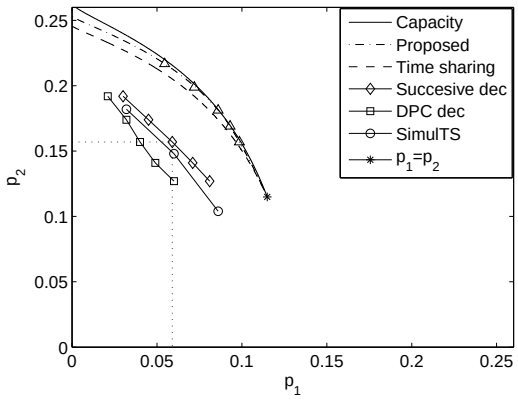


Fig. 9. The performance of soft DPC when $t = 0$ and $t = t_0$ ($R_0 = 1/3$, $R_1 = 0.36$, and $R_2 = 1/8$)

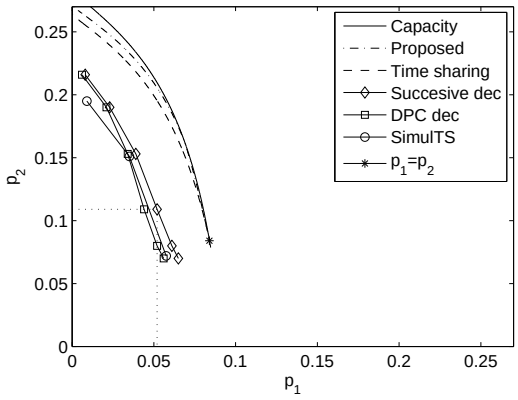


Fig. 10. The performance of soft DPC when $t = 0$ and $t = t_0$ ($R_0 = 1/6$, $R_1 = 1/2$, and $R_2 = 1/12$)

while the true capacity is $p^* = 0.2949$. Such difference $\Delta p = p^* - p = 0.0299$ is resulted from using LDPC codes of finite node degrees. In Fig. 9, we also plot the theoretic capacity by triangle markers and plotted the performance of the proposed scheme with successive decoding by diamond markers. When we set $\alpha = 0$ and $t = 0$, which are the parameter values that are potentially capable of achieving the native DPC point, the empirical decodable threshold of (p_1, p_2) is $(0.059, 0.157)$ while the theoretic capacity of the native DPC point is $(p_1^*, p_2^*) = (0.0892, 0.1764)$. The performance loss is $\Delta(p_1, p_2) = (p_1^* - p_1, p_2^* - p_2) = (0.0302, 0.0194)$, which is comparable to the performance loss 0.0299 of LDPC codes on pure BSCs.

Compared to the pure time-sharing scheme, which suffers from the negative effect of heavy puncturing, the proposed soft-DPC scheme also relies on puncturing when combing codewords during the 3-way time sharing (see Fig. 5). However, the proposed approach does not suffer much from the performance loss of heavy puncturing due to the following reasons. Take the scenario in Fig. 9 for example. The three points of the curve with circular markers correspond to time sharing with $t = 0.5$, 0.6, and 0.7. Therefore, if we would like to achieve the time-sharing point by puncturing, the mother

codes need to be heavily punctured with puncturing rate 0.3–0.7. Let us now turn the focus to the soft DPC scheme. The rightmost two points of the curve with diamond markers corresponds to soft DPC with $(\alpha, t) = (0.2, t_0)$ and $(0.4, t_0)$, respectively, where $t_0 = \frac{0.36}{0.36+0.125} \approx 0.74$. Since in this case the portions of puncturing are $\alpha(1-t)$ and αt, for receivers 1 or 2, respectively, the corresponding puncturing rate ranges from 0.05 to 0.29. Compared to pure time-sharing schemes, the soft DPC scheme exploits only a small percentage of puncturing during the 3-way time-sharing block, which thus does not suffer much from the heavy puncturing used in time sharing.

VII. Conclusions

We have identified various practical challenges of existing superposition-coding-based DPC scheme and the time-sharing scheme for the BC, and have proposed a new practical binary broadcast system using a new concept of soft DPC, which is a 3-way time sharing between the native DPC scheme and the codewords of users 1 and 2. The proposed scheme achieves a large percentage of the capacity region for a wide variety of channel quality and uses only one fixed pair of codes and a single quantization code. Based on off-the-shelf binary LDPC and convolutional codes, practical systems for DPCs and for BCs have been designed and their performances have been verified by simulation. The proposed system admits complexity advantages and enables flexible choices of system parameters when compared to the existing superposition-coding-based schemes that use high-order $\mathbf{GF}(q)$ codes.

References

[1] ETSI, "Transmission system for handheld terminals," *EN 302 204 V1.1.1*, Nov. 2004.

[2] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley, New York, 1991.

[3] T. Cover, "Broadcast channels," *IEEE Trans. Inform. Theory*, vol. 18, no. 1, pp. 2–14, Jan. 1972.

[4] G. D. Forney Jr., "Trellis shaping," *IEEE Trans. Inform. Theory*, vol. 38, no. 2, pp. 281–300, Mar. 1992.

[5] J. Kusuma and K. Ramchandran, "Communicating by cosets and applications to broadcast," in *Proc. 2002 Conf. Information Sciences and Systems*. Princeton, NJ, Mar. 2002.

[6] M. Costa, "Writing on dirty paper," *IEEE Trans. Inform. Theory*, vol. 29, no. 3, pp. 439–441, May 1983.

[7] S. Gel'fand and M. Pinsker, "Coding for channel with random parameters," *Probl. Contr. Inform. Theory*, vol. 9, no. 1, pp. 19–31, Jan. 1980.

[8] R. Zamir, S. Shamai, and U. Erez, "Nested linear/lattice codes for structured multiterminal binning," *IEEE Trans. Inform. Theory*, vol. 48, no. 6, pp. 1250–1276, June 2002.

[9] U. Erez and S. ten Brink, "A close-to-capacity dirty paper coding scheme," *IEEE Trans. Inform. Theory*, vol. 51, no. 10, pp. 3417–3432, Oct. 2005.

[10] A. Bennatan, D. Burshtein, G. Caire, and S. Shamai, "Superposition coding for side-information channels," *IEEE Trans. Inform. Theory*, vol. 52, no. 5, pp. 1872–1889, May 2006.

[11] Y. Sun, Y. Yang, A. Liveris, V. Stankovic, and Z. Xiong, "Near-capacity dirty-paper code design: A source-channel coding approach," *IEEE Trans. Inform. Theory*, vol. 55, no. 7, pp. 3013–3031, Jul. 2009.

[12] Y. Sun, M. Uppal, A. Liveris, S. Cheng, V. Stankovic, and Z. Xiong, "Nested turbo codes for the Costa problem," *IEEE Trans. Commun.*, vol. 56, no. 3, pp. 388–399, Mar. 2008.

[13] G. Caire and S. Shamai, "On the achievable throughput of a multi-antenna Gaussian broadcast channel," *IEEE Trans. Inform. Theory*, vol. 49, no. 7, pp. 1691–1706, July 2003.

[14] W. Yu and J. M. Cioffi, "Trellis precoding for the broadcast channel," in *Proc. IEEE Glob. Telecom. Conf.* San Antonio, TX, Nov. 2001, pp. 1344–1348.

[15] W. Yu and J. M. Cioffi, "Sum capacity of Gaussian vector broadcast channels," *IEEE Trans. Inform. Theory*, vol. 50, no. 9, pp. 1875–1892, Sept. 2004.

[16] M. Uppal, V. Stankovic, and Z. Xiong, "Code designs for MIMO broadcast channels," *IEEE Trans. Commun.*, vol. 57, no. 4, pp. 986–996, Apr. 2009.

[17] H. Weingarten, Y. Steinberg, and S. Shamai, "The capacity region of the Gaussian multiple-input multiple-output broadcast channel," *IEEE Trans. Inform. Theory*, vol. 52, no. 9, pp. 3936–3964, Sept. 2006.

[18] R. Barron, B. Chen, and G. Wornell, "The duality between information embedding and source coding with side information and some applications," *IEEE Trans. Inform. Theory*, vol. 49, no. 5, pp. 1159–1180, May 2003.

[19] S. Pradhan, J. Chou, and K. Ramchandran, "Duality between source coding and channel coding with side information," *IEEE Trans. Inform. Theory*, vol. 49, no. 5, pp. 1181–1203, May 2003.

[20] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[21] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[22] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.

[23] T. J. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoder," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[24] K. Price and R. Storn, "Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimiz.*, vol. 11, pp. 341–359, 1997.

[25] M. P. C. Fossorier, "Quasi-cyclic low-density parity-check codes from circulant permutation matrices," *IEEE Trans. Inform. Theory*, vol. 50, no. 8, pp. 1788–1793, Aug. 2004.

[26] "IEEE P802.16e-2006 draft standards for local and metropolitan area networks part 16: Air interface for fixed broadcast wireless access systems," *IEEE Standard 802.16e*, Feb. 2006.

[27] S.-Y. Chung, G. D. Forney, T. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, no. 2, pp. 58–60, Feb. 2001.