Asynchronous Physical-Layer Network Coding

Lu Lu, *Student Member, IEEE,* and Soung Chang Liew, *Fellow, IEEE*

***Abstract*—A key issue in physical-layer network coding (PNC) is how to deal with the asynchrony between signals transmitted by multiple transmitters. That is, symbols transmitted by different transmitters could arrive at the receiver with symbol misalignment as well as relative carrier-phase offset. A second important issue is how to integrate channel coding with PNC to achieve reliable communication. This paper investigates these two issues and makes the following contributions: 1) We propose and investigate a general framework for decoding at the receiver based on belief propagation (BP). The framework can effectively deal with symbol and phase asynchronies while incorporating channel coding at the same time. 2) For unchannel-coded PNC, we show that for BPSK and QPSK modulations, our BP method can significantly reduce the asynchrony penalties compared with prior methods. 3) For QPSK unchannel-coded PNC, with a half symbol offset between the transmitters, our BP method can drastically reduce the performance penalty due to phase asynchrony, from more than 6 dB to no more than 1 dB. 4) For channel-coded PNC, with our BP method, both symbol and phase asynchronies actually improve the system performance compared with the perfectly synchronous case. Furthermore, the performance spread due to different combinations of symbol and phase offsets between the transmitters in channel-coded PNC is only around 1 dB. The implication of 3) is that if we could control the symbol arrival times at the receiver, it would be advantageous to deliberately introduce a half symbol offset in unchannel-coded PNC. The implication of 4) is that when channel coding is used, symbol and phase asynchronies are not major performance concerns in PNC.**

***Index Terms*—Physical-layer network coding, network coding, synchronization.**

I. Introduction

PHYSICAL-LAYER network coding (PNC), first proposed in [1], is a subfield of network coding [2] that is attracting much attention recently. The simplest system in which PNC can be applied is the two-way relay channel (TWRC), in which two end nodes exchange information with the help of a relay node in the middle, as illustrated in Fig. 1.

This paper focuses on TWRC. Compared with the conventional relay system, PNC doubles the throughput of TWRC by reducing the number of time slots for the exchange of one packet from four to two. In PNC, in the first time slot, the two end nodes send signals simultaneously to the relay; in the second phase, the relay processes the superimposed signals of the simultaneous packets and maps them to a network-coded packet for broadcast back to the end nodes.

Despite the potential advantages of PNC, a key issue in PNC is how to deal with the asynchronies between the signals transmitted simultaneously by the two end nodes. That is, symbols transmitted by the two end nodes could arrive at the receiver with symbol misalignment as well as relative carrier-phase offset.

Many previous works (e.g., [1], [3], [4]) found that symbol misalignment and carrier-phase offset will result in appreciable performance penalties. For BPSK modulation, [1] showed that the BER performance penalties due to carrier-phase offset and symbol offset are both 3 dB in the worst case. For QPSK modulation, the penalty can be as large as 6 dB in the worst case when the carrier-phase offset is $\pi/4$ [4]. These results are for unchannel-coded PNC.

These earlier investigations led to a common belief that near-perfect symbol and carrier-phase synchronizations are important for good performance in PNC. This paper shows that this is not exactly true, and that asynchronous PNC can have good performance when appropriate methods are applied.

The study of BPSK unchannel-coded PNC in [1] and [3], for example, made use of suboptimal decoding methods at the relay for asynchronous PNC. Furthermore, the joint effect of symbol and phase asynchronies was not investigated. In this paper, we propose an optimal maximum-likelihood (ML) decoding method that makes use of a belief propagation (BP) algorithm. Our method addresses symbol and phase asynchronies jointly within one framework. We find that our method can reduce the worst-case BER performance penalty of 3 dB in [1] and [3] to less than 0.5 dB.

In QPSK unchannel-coded PNC, the penalty is larger than 6 dB [4] only when the symbols are perfectly aligned and when the phase offset is $\pi/4$ (benchmarked against the perfectly synchronous case in which there are no symbol and phase offsets). We find that using our method, when there is a half symbol misalignment, the penalty is reduced to less than 1 dB. Additionally, an interesting result is that with a half symbol misalignment, the spread of penalties under various carrier-phase offsets is no more than 0.5 dB. This means that symbol misalignment has the effect of desensitizing the performance of the system to carrier-phase offset.

For reliable communication, channel coding is often applied. Therefore, another important issue is how to incorporate channel codes into PNC. We extend our BP method so that it can incorporate channel decoding and deal with asynchrony at the same time. An interesting result when channel coding is adopted in PNC is that with an appropriate BP algorithm, instead of asynchrony penalty, we have asynchrony reward. In particular, both symbol misalignment and phase offset

Manuscript received June 5, 2011; revised October 12, 2011; accepted October 27, 2011. The associate editor coordinating the review of this paper and approving it for publication was W. Lou.

The authors are with the Department of Information Engineering, The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong (e-mail: {ll007, soung}@ie.cuhk.edu.hk).

This work is supported by the General Research Fund, Project Number 414911, and the AoE grant E-02/08, established under the University Grant Committee of the Hong Kong Special Administrative Region, China.

This paper was presented in part at the IEEE International Conference on Communication (ICC) 2011, Kyoto, Japan.

Digital Object Identifier 10.1109/TWC.2011.120911.111067

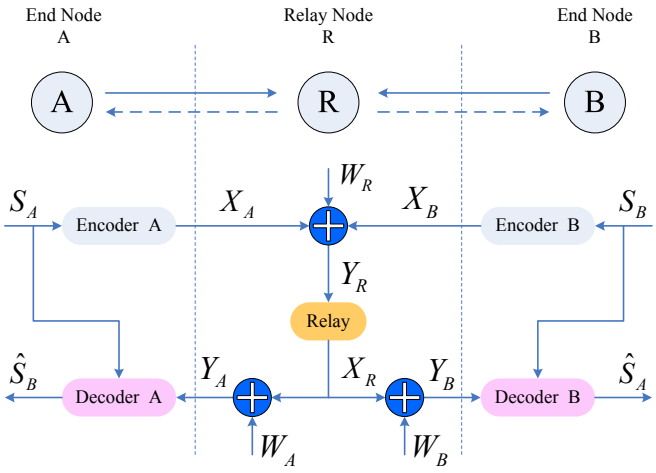


Fig. 1. System model for two way relay channel.

TABLE I
FOUR CASES OF PNC SYSTEMS

	$\Delta = 0$	$\Delta \neq 0$
$\phi = 0$	$\phi = 0, \Delta = 0$ (Case 1)	$\phi = 0, \Delta \neq 0$ (Case 2)
$\phi \neq 0$	$\phi \neq 0, \Delta = 0$ (Case 3)	$\phi \neq 0, \Delta \neq 0$ (Case 4)

Δ is the symbol offset and ϕ is the phase offset

improve BER performance when channel coding is used. In addition, the performance spread arising from all combinations of symbol and phase offsets is only around 1 dB. This suggests that when channel coding is used, symbol and phase asynchronies are not major performance concerns.

The remainder of this paper is organized as follows: Section II overviews related work. Section III introduces the system model of this paper. Section IV presents our BP ML decoding method for asynchronous unchannel-coded PNC. Section V extends the method for channel-coded PNC. Numerical results are given in the subsections of IV-C and V-D, respectively. Finally, Section VI concludes this paper.

II. Related Work

A. Classification

Table I shows the four possible cases for PNC systems. In Table I, $\Delta \in [0, T)$ is the relative symbol offset between the two end nodes A and B, where T is the symbol duration; and $\phi \in [0, 2\pi)$ is the relative phase offset between the RF carriers of the two end nodes. Case 1 is the perfectly synchronized case studied in [1] and [5]; Case 2 is the symbol-asynchronous case studied in [3]; and Case 3 is the phase-asynchronous case studied in [3], and [6]–[8]. Case 4 is the symbol-asynchronous phase-asynchronous case. To our best knowledge, it has not been studied before. This paper proposes a general scheme to tackle all four cases under one framework.

B. Unchannel-coded PNC

Refs. [1] and [3] argued that the largest asynchrony penalty in unchannel-coded PNC is 3 dB for BPSK modulation. However, this conclusion is based on suboptimal decoding.

Ref. [4] mentioned without proof that there is a maximum 6 dB BER performance penalty for QPSK modulation when $\Delta = 0$ and $\phi \neq 0$. To the best of our knowledge, no quantitative results and concrete explanation have been given for the general Δ and ϕ case.

Ref. [7] investigated systems in which symbols are aligned but phases are not. It uses QPSK for uplinks, but a higher order constellation map (e.g., 5QAM) for downlinks when the uplink phase offset is not favorable to using QPSK for the downlinks. That is, it varies the mode of PNC mapping depending on the phase asynchrony. In this paper, we assume the simpler system in which both the uplink and downlink use the same modulation, either BPSK or QPSK.

C. Channel-coded PNC

For channel-coded PNC, an important issue is how to integrate the channel decoding operation and the network coding operation at the relay. Ref. [9] presented a scheme that works well for synchronous channel-coded PNC. The scheme is not amenable to extension for asynchronous channel-coded PNC.

Ref. [8] proposed a method for symbol-synchronous phase-asynchronous channel-coded PNC, assuming the use of Low-Density Parity-Check (LDPC) code. Different from the scheme in [8], our method deals with both phase and symbol asynchronies.

Refs. [10] and [11] investigated OFDM PNC. With OFDM, the symbol offset in the time domain is translated into different phase offsets in different subcarriers in the frequency domain. Since different subcarriers experience different phase offsets, there is an averaging effect as far as performance is concerned, and the system performance is not at the mercy of the worst-case phase asynchrony. The channel-decoding and network-coding process in [10] and [11], however, are performed in a disjoint manner (using an XOR-CD decoder that will be described in Section V-C). By contrast, the joint channel-decoding and network-coding scheme (Jt-CNC) proposed in this paper can yield much better performance (to be presented in Section V-B).

In this paper, we use the Repeat-Accumulate (RA) channel code to explain the principles of Jt-CNC and XOR-CD, as well as in the numerical studies. The general conclusions and qualitative results of our study based on the RA code are also valid for the LDPC code with the use of similar BP algorithms. In particular, our simulations yield similar asynchrony effects for both the LDPC and RA codes. To conserve space, this paper presents the results of the RA code only. LDPC results can be found in Appendix III of our technical report [12].

Convolutional-coded PNC has been studied previously for the symbol-synchronous case [13]. We conjecture that for the asynchronous convolutional-coded PNC, with a decoding scheme similar to ours here, similar qualitative results of the asynchronous effects will also be observed. This is a subject for further study.

The use of BP has also been proposed in a number of places in the literature in the context of multi-user detection [14] and joint detection and decoding in the presence of phase noise and frequency offset [15]. In [16], BP over a factor graph that describes the joint probability law of all unknowns and observations is used to decode one of two users in the presence

of Gauss-Markov (non-block) fading. This work is along the line of collision resolution [17] rather than PNC mapping. In this paper, we assume that the channels can be perfectly estimated, leaving out the detailed estimation procedure. Ref. [16] provides a nice way to integrate the problem of channel estimation and detection using BP. The application of the technique in PNC is an interesting area for further work.

III. System Model

We study the two-way relay channel as shown in Fig. 1, in which nodes A and B exchange information with the help of relay node R. We assume that all nodes are half-duplex, i.e., a node cannot receive and transmit simultaneously and there is no direct link between A and B.

We consider a two-phase transmission scheme consisting of an uplink phase and a downlink phase. In the uplink phase, nodes A and B transmit packets to node R simultaneously. In the downlink phase, based on the overlapped signals received from A and B, R constructs a network-coded packet and broadcast the packet to A and B. Upon receiving the network-coded packet, A (B) then attempts to recover the original packet transmitted by B (A) in the uplink phase using self-information [1].

This paper focuses on the performance of the uplink phase because the performance of the downlink phase is similar to that in a conventional point-to-point link. Consider the uplink phase. If only A transmitted, then the received complex baseband signal at R (i.e., the received signal after down-conversion from the carrier frequency and low-pass filtering) would be

$$y_R(t) = \sum_{n=1}^{N} h_A x_A[n] p(t - nT) + w_R(t), \tag{1}$$

where $h_A = \sqrt{P_A}$ is the received signal amplitude; $(x_A[n])_{n=1,\ldots,N}$ are the symbols in the packet of A; $p(t-nT)$ is the pulse shaping function for the baseband signal; and $w_R(t)$ is the additive white Gaussian noise (AWGN). In the PNC set-up, A and B transmit simultaneously. In this case, the received complex baseband signal at R is

$$y_R(t) = \sum_{n=1}^{N} \{h_A x_A[n] p(t - nT) + h_B x_B[n] p(t - \Delta - nT)\} + w_R(t), \tag{2}$$

where $h_B = \sqrt{P_B} e^{j\phi}$ (ϕ is the relative phase offset between the signals from A and B due to phase asynchrony in their carrier-frequency oscillators and the difference in the path delays of the two uplink channels); $(x_B[n])_{n=1,\ldots,N}$ are the symbols in the packet of B; and Δ is a time offset between the arrivals of the signals from A and B. This paper assumes that the channel state information h_A and h_B can be perfectly estimated at the relay R. We assume, however, that the transmitters A and B do not know the relative phase offset. This means that A and B cannot perform precoding to remove the phase offset. Without loss of generality, we assume the signal of A arrives at R earlier than the signal of B. Furthermore, we assume Δ is within one symbol period T. Thus, $0 \le \Delta < T$.[1] We refer to Δ and ϕ as the symbol and phase offsets (or misalignments) at R, respectively. For simplicity, we assume power control (equalization) so that $P_A = P_B = P$. Furthermore, for convenience, we assume time is expressed in unit of symbol duration, so that $T = 1$. We can then rewrite (2) as

$$y_R(t) = \sqrt{P} \sum_{n=1}^{N} \left\{x_A[n] p(t - n) + x_B[n] p(t - \Delta - n) e^{j\phi}\right\} + w_R(t). \tag{3}$$

In general, the pulse shaping function $p(t)$ can take different forms. The discussion on different pulse shaping functions, however, is beyond the scope of this paper. To bring out the essence of our results in the simplest manner, throughout this paper, we assume the rectangular pulse shape: $p(t) = rect(t) = u(t+1) - u(t)$.

A critical design issue is how relay R makes use of $y_R(t)$ to construct a network-coded packet for broadcast to nodes A and B in the downlink phase. In this paper, we assume that R first oversamples $y_R(t)$ to obtain $2N+1$ signal samples. It then uses the $2N+1$ signal samples to construct an N-symbol network-coded packet for broadcast to A and B. The matched filtering and oversampling procedure described below is similar to that in [18], [19]. For $n = 1, \ldots, N$,

$$\begin{aligned} y_R[2n-1] &= \frac{1}{\Delta\sqrt{P}} \int_{(n-1)}^{(n-1)+\Delta} y_R(t) dt \\ &= \frac{1}{\Delta} \int_{(n-1)}^{(n-1)+\Delta} \left(x_A[n] + x_B[n-1] e^{j\phi} + \frac{w_R(t)}{\sqrt{P}} \right) dt \\ &= x_A[n] + x_B[n-1] e^{j\phi} + w_R[2n-1], \end{aligned}$$

$$\begin{aligned} y_R[2n] &= \frac{1}{(1-\Delta)\sqrt{P}} \int_{(n-1)+\Delta}^{n} y_R(t) dt \\ &= \frac{1}{1-\Delta} \int_{(n-1)+\Delta}^{n} \left(x_A[n] + x_B[n] e^{j\phi} + \frac{w_R(t)}{\sqrt{P}} \right) dt \\ &= x_A[n] + x_B[n] e^{j\phi} + w_R[2n], \end{aligned} \tag{4}$$

and

$$\begin{aligned} y_R[2N+1] &= \frac{1}{\Delta\sqrt{P}} \int_{N}^{N+\Delta} y_R(t) dt \\ &= \frac{1}{\Delta} \int_{N}^{N+\Delta} \left(x_B[N] e^{j\phi} + \frac{w_R(t)}{\sqrt{P}} \right) dt \\ &= x_B[N] e^{j\phi} + w_R[2N+1], \end{aligned}$$

where $x_B[0] = 0$, and $w_R[2n-1]$ (also $w_R[2N+1]$) and $w_R[2n]$ are a zero-mean complex Gaussian noise with variance $N_0/(2P\Delta)$ and $N_0/(2P(1-\Delta))$, respectively, for both the real and imaginary components. Note that the powers in $x_A[n]$ and $x_B[n]$ have been respectively normalized to one unit, and $P/N_0 = PT/N_0 = E_s/N_0$ is the SNR per symbol for the

[1] If Δ is more than one symbol period, we could generalize our treatment here so that N is larger than the number of symbols in a packet. The packets will only be partially overlapping, with non-overlapping symbols at the front end and tail end. Essentially, our assumption of Δ being within one symbol period implies that we are looking at the "worst case" with maximum overlapping between the two packets. When there are additional non-overlapping symbols at the front and tail ends, the decoding will have better error probability performance.

signal from node A or node B. Based on $(y_R[n])_{n=1,\ldots,2N+1}$, relay R constructs a network-coded packet $(x_R[n])_{n=1,\ldots,N}$ for broadcast to end nodes A and B

The lower part of Fig. 1 includes a schematic diagram. It incorporates channel coding into the PNC system. This paper adopts the following notations:

- $S_i = (s_i[1], s_i[2], \ldots, s_i[M])$ denotes the source packet of node *i*, $i \in \{A, B\}$;
- $X_i = (x_i[1], x_i[2], \ldots, x_i[N])$ denotes the channel-coded packet of node *i*, $i \in \{A, B\}$;
- $Y_R = (y_R[1], y_R[2], \ldots, y_R[N], y_R[N+1], \ldots, y_R[2N+1])$ denotes the received packet (with the aforementioned oversampling) at relay node R;
- $W_R = (w_R[1], w_R[2], \ldots, w_R[N], w_R[N+1], \ldots, w_R[2N+1])$ denotes the receiver noise at node R;
- $X_R = (x_R[1], x_R[2], \ldots, x_R[N])$ denotes the network-coded packet at relay node R;
- $Y_i = (y_i[1], y_i[2], \ldots, y_i[N])$ denotes the received PNC packet at node *i*, $i \in \{A, B\}$;
- $W_i = (w_i[1], w_i[2], \ldots, w_i[N])$ denotes the receiver noise at node *i*, $i \in \{A, B\}$;
- $\hat{S}_i = (\hat{s}_i[1], \hat{s}_i[2], \ldots, \hat{s}_i[N])$ denotes the decoded source packet of node *i*, $i \in \{A, B\}$, at the other end node;

where M is the number of source symbols, and N is the number of channel-coded symbols. Note here that, throughout this paper, we focus on BPSK and QPSK in our analytical and simulation results, although the framework can be extended to more complex constellations. For BPSK, $s_i[\cdot], x_i[\cdot], \hat{s}_i[\cdot], x_R[\cdot] \in \{-1, 1\}$; for QPSK, $s_i[\cdot], x_i[\cdot], \hat{s}_i[\cdot], x_R[\cdot] \in \{(1+j)/\sqrt{2}, (-1+j)/\sqrt{2}, (-1-j)/\sqrt{2}, (1-j)/\sqrt{2}\}$, and $y_i[\cdot], y_R[\cdot], w_i[\cdot], w_R[\cdot] \in \mathbb{C}$.

IV. Unchannel-coded PNC

This section focuses on unchannel-coded PNC, where each end node transmits the source information without channel coding. Thus, with respect to the bottom part of Fig. 1, we have $X_A = S_A$ and $X_B = S_B$. For asynchronous unchannel-coded PNC, we investigate the use of Belief Propagation (BP) in the PNC mapping process to deal with phase and symbol asynchronies. We find that symbol misalignment can drastically reduce the performance penalty due to phase offset.

A. Synchronous Unchannel-coded PNC

We first give a quick review of synchronous unchannel-coded PNC. The two end nodes transmit their packets $X_A = S_A$ and $X_B = S_B$ without channel coding. The relay R receives the combined signals with $\phi = 0$ and $\Delta = 0$. The received baseband packet at R is $Y_R = X_A + X_B + W_R$ with *N* symbols. Relay R transforms Y_R into a network-coded packet $X_R = f(Y_R)$ with N symbols for transmission in the downlink phase.

For this case, with reference to (4), since $\Delta = 0$, the variance of the noise term $w_R[2n-1]$ is infinite, and the signal is contained only in the even terms $y_R[2n]$. Thus, we can write

$$y_R[2n] = x_A[n] + x_B[n] + w_R[2n], \tag{5}$$

where $n = 1, \ldots, N$, and $w_R[2n]$ is zero-mean Gaussian noise with variance $\sigma^2 = N_0/(2P)$ for both the real and imaginary components.

For BPSK, $x_i[n] \in \{-1, 1\}$. Only the real component of $w_R[2n]$ needs to be considered. For QPSK, since we are considering a synchronous system, the in-phase and quadrature-phase components in (5) are independent; it can therefore be considered as two parallel BPSK systems. Thus, in the following, we only consider BPSK.

Let us consider a particular time index *n*, and omit the index *n* in our notation for simplicity. The *a posteriori* probability of the combination of source symbols (x_A, x_B) is given by

$$\begin{aligned}\Pr(x_A, x_B|y_R) &= \frac{\Pr(y_R|x_A, x_B)}{4\Pr(y_R)} \\ &= \frac{1}{4\Pr(y_R)\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(y_R - x_A - x_B)^2}{2\sigma^2}\right\}. \end{aligned} \tag{6}$$

Let us use $x_i = 1$ represent bit 0 and $x_i = -1$ represent bit 1. Suppose that the downlink transmission also uses BPSK. For PNC output x_R, we assume we want to get the XOR mapping, i.e., $x_R = x_A \oplus x_B$ [1]. Then, $x_R = 1$ if $x_A = x_B$, and $x_R = -1$ if $x_A \neq x_B$. The following decision rule can be used to map y_R to x_R:

$$\begin{aligned} &\Pr(x_A = 1, x_B = 1|y_R) + \Pr(x_A = -1, x_B = -1|y_R) \\ &\underset{x_R=-1}{\overset{x_R=1}{\gtrless}} \Pr(x_A = 1, x_B = -1|y_R) + \Pr(x_A = -1, x_B = 1|y_R) \\ &\Rightarrow \left(\exp\left\{-\frac{(y_R-2)^2}{2\sigma^2}\right\} + \exp\left\{-\frac{(y_R+2)^2}{2\sigma^2}\right\}\right) \\ &\quad \underset{x_R=-1}{\overset{x_R=1}{\gtrless}} 2\exp\left\{-\frac{y_R^2}{2\sigma^2}\right\}. \end{aligned} \tag{7}$$

B. BP-UPNC: A Belief Propagation Decoding Algorithm for Asynchronous Unchannel-coded PNC

We now consider the asynchronous unchannel-coded PNC. In the synchronous case, the sampled observations $y_R[\cdot]$ in different symbol periods are independent. In general, $(x_A[n], x_B[n])$ affects only $y_R[n]$ and not $y_R[k], k \neq n$. Thus, we have the simple decision rule as in (7). This is not the case with the asynchronous case. In particular, for optimal detection of $(x_A[n], x_B[n])$, we need to look at the whole sequence of observed samples Y_R. That is, we need to look at $\Pr(x_A[n], x_B[n]|Y_R)$, not just $\Pr(x_A[n], x_B[n]|y_R[n])$. A belief propagation (BP) algorithm can be used to compute $\Pr(x_A[n], x_B[n]|Y_R)$. BP is essentially a "chained" application of Bayes' rule for the computation of $\Pr(x_A[n], x_B[n]|Y_R)$.

BP is a framework for generating inference-making algorithms for graphical models, in which each node represents a variable (note: could be a vector variable, such as $(x_A[\cdot], x_B[\cdot])$ in asynchronous PNC.). Some of the nodes (variables) can be observed while others cannot be observed. The relationships among the nodes are represented by edges between them in the graph. The goal of BP is to compute the marginal probability distribution of each unobserved node x_i conditioned on the observations at all the observed nodes $y_j \forall j$. That is, BP aims to compute $P(x_i|y_j \forall j) \forall i$. It does so by means of a sum-product message-passing algorithm. Note that x_i's and y_j's are

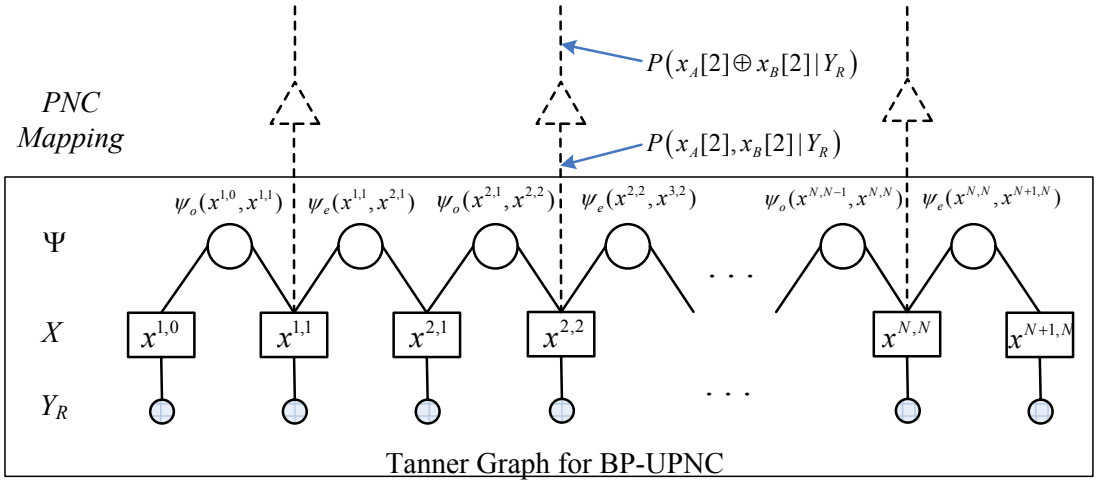


Fig. 2. Tanner graph of the BP-UPNC and PNC mapping after BP algorithm on the Tanner graph. The triangle nodes perform the PNC mapping described in Section IV-B.

interrelated and not independent, hence the graphical model to describe their relationships. As will be seen shortly, for our purpose in asynchronous PNC, the graphical model is a Tanner graph, the observed nodes are the received signal samples, and the unobserved nodes are the 2-tuples consisting of pairs of symbols from the two end nodes.

The BP algorithm was first proposed by Judea Pearl in 1988, assuming that the underlying graphical model is a tree [20]. For tree graphs, it can be shown that the BP algorithm is an implementation of the Bayes' theorem, and it yields the exact conditional marginal probability distributions of the unobserved nodes with no approximation. The BP algorithm was later applied to "loopy" graphs as if they were tree graphs (i.e., the same Bayes' formula for tree graphs is used in loopy graphs). Thus, for loopy graphs, the computed conditional marginal probabilities are only approximations. This BP approximation has been found to work well empirically for many problems with loopy graphs in terms of the accuracy of the approximate results it yields as well as its convergence speed [20], [21].

Although not absolutely needed, it is sometimes more convenient to introduce another kind of nodes called the factor nodes in the graph. A factor node models the inter-relationship among a group of variable nodes. The resulting graph, which includes the factor nodes, is referred to as the Tanner graph or simply the factor graph. The observed variable nodes are referred to as the evidence nodes, and the unobserved nodes are referred to as the variable nodes.

Let us now look at the asynchronous unchannel-coded PNC setup and explain how BP is applied. Recall from the notation definitions at the end of Section III that in asynchronous unchannel-coded PNC the two end nodes transmit their packets $X_A = S_A$ and $X_B = S_B$ without channel coding. The relay node R receives the combined signals with $\phi \neq 0$ and/or $\Delta \neq 0$. The received baseband packet at R is $Y_R = X_A + X_B + W_R$ with $2N + 1$ symbols. Node R transforms Y_R into a network-coded packet $X_R = f(Y_R)$ with N symbols for transmission in the downlink phase.

We present a scheme for the PNC mapping $X_R = f(Y_R)$ based on BP. Our scheme deals with symbol and phase asynchronies jointly. We refer to this scheme as BP-UPNC.

1) Tanner Graph of BP-UPNC: We make use of the oversampled symbols in (4) to construct a Tanner graph [22] as shown in Fig. 2. In the Tanner graph, Y_R denotes the evidence nodes, and there are $2N + 1$ such nodes; Ψ denotes the factor nodes (also known as the constraint nodes, the compatibility nodes or check nodes in the literature); and X denotes the variable nodes (also known as the source nodes). For simplicity, we use $x^{i,j}$ to denote the joint symbol $(x_A[i], x_B[j])$. The correlation between two adjacent joint symbols is modeled by the compatibility functions (i.e., check nodes) $\psi_o(x^{n,n-1}, x^{n,n})$ and $\psi_e(x^{n,n}, x^{n+1,n})$ for the odd and even compatibility nodes:

$$\psi_o(x^{n,n-1}, x^{n,n}) = \begin{cases} 1 & \text{if } x_A[n] \text{ in } x^{n,n-1} \text{ and } x^{n,n} \\ & \text{are equal} \\ 0 & \text{otherwise} \end{cases}$$

$$\psi_e(x^{n,n}, x^{n+1,n}) = \begin{cases} 1 & \text{if } x_B[n] \text{ in } x^{n,n} \text{ and } x^{n+1,n} \\ & \text{are equal} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

We first decode the combination $(x_A[n], x_B[n])$ in X. Note that the Tanner graph has a tree structure. This means that BP can find the "*exact*" *a posteriori* probability $P(x^{n,n}|Y_R)$ for $n = 1, \ldots, N$. Furthermore, the solution can be found after only one iteration of the message-passing algorithm [22]. From the decoded $P(x^{n,n}|Y_R)$, we can then find the maximum *a posteriori* probability (MAP) XOR value

$$\begin{aligned} x_R[n] &= \arg\max_x P\left(x_A[n] \oplus x_B[n] = x | Y_R\right) \\ &= \arg\max_x \sum_{x^{n,n}:\ x_A[n] \oplus x_B[n] = x} P\left(x^{n,n} | Y_R\right). \end{aligned} \tag{9}$$

In summary, BP can converge quickly and is MAP-optimal as far as the BER of $x_A[n] \oplus x_B[n]$ is concerned. Note that MAP optimal is also ML optimal here because the *a priori* probability $P(x^{n,n})$ for different values of $x^{n,n}$ are equally likely.

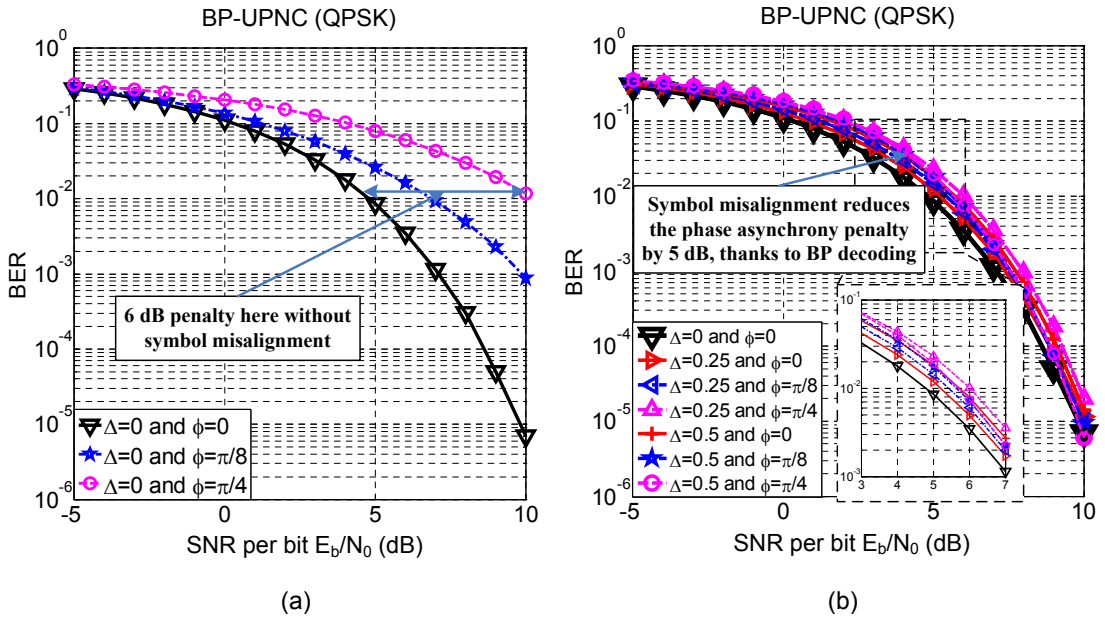


Fig. 3. BER of the uplink XORed value $x_A[n] \oplus x_B[n]$ in BP-UPNC for QPSK modulated unchannel-coded PNC. (a): BP-UPNC without symbol asynchrony ($\Delta = 0$); (b): BP-UPNC with symbol asynchrony ($\Delta \neq 0$). Note that $s_A[n] = x_A[n]$ and $s_B[n] = x_B[n]$ for unchannel-coded PNC.

2) Message Update Rules: Let us consider QPSK. In particular, we define $\chi = \{1+j, -1+j, \ -1-j, 1-j\}$ as the symbol set. With reference to (4), we have $x_A[n] = a/\sqrt{2}$ and $x_B[n] = b/\sqrt{2}$, where $a, b \in \chi$. Define $p_k^{a,b} = P\left(x_A[\lceil k/2 \rceil] = a/\sqrt{2}, x_B[\lfloor k/2 \rfloor] = b/\sqrt{2} \ \middle| y_R[k]\right)$. Note that here, $p_k^{a,b}$ is computed based on $y_R[k]$ only, and not on the whole Y_R. Also, $p_k^{a,b}$ is fixed and does not change throughout the message passing algorithm in the Tanner graph. $P_{2n-1}^{a,b}$ and $P_{2n}^{a,b}$, $n = 1, 2, \ldots, N$, are given as follows:

$$p_{2n-1}^{a,b} = P\left(x_A[n] = \frac{a}{\sqrt{2}}, x_B[n-1] = \frac{b}{\sqrt{2}} \middle| y_R[2n-1]\right)$$
$$\propto \frac{1}{2\pi\sigma^2/\Delta} \exp\left\{-\frac{\left(y_R^{\mathrm{Re}}[2n-1] - \mathrm{Re}\left(a + be^{j\phi}\right)/\sqrt{2}\right)^2}{2\sigma^2/\Delta}\right\} \cdot$$
$$\exp\left\{-\frac{\left(y_R^{\mathrm{Im}}[2n-1] - \mathrm{Im}\left(a + be^{j\phi}\right)/\sqrt{2}\right)^2}{2\sigma^2/\Delta}\right\}, \qquad (10)$$

$$p_{2n}^{a,b} = P\left(x_A[n] = \frac{a}{\sqrt{2}}, x_B[n] = \frac{b}{\sqrt{2}} \middle| y_R[2n]\right)$$
$$\propto \frac{1}{2\pi\sigma^2/(1-\Delta)} \exp\left\{-\frac{\left(y_R^{\mathrm{Re}}[2n] - \mathrm{Re}\left(a + be^{j\phi}\right)/\sqrt{2}\right)^2}{2\sigma^2/(1-\Delta)}\right\} \cdot$$
$$\exp\left\{-\frac{\left(y_R^{\mathrm{Im}}[2n] - \mathrm{Im}\left(a + be^{j\phi}\right)/\sqrt{2}\right)^2}{2\sigma^2/(1-\Delta)}\right\}. \qquad (11)$$

Note that except for the first and last symbols, each of $p_{2n-1}^{a,b}$ and $p_{2n}^{a,b}$ has 16 possible combinations (4 possibilities for a and 4 possibilities for b). The first and last symbols have 4 possibilities, as follows: $p_1^{a,0} = P(x_A[1] = a\sqrt{2}, x_B[0] = 0 | y_R[1]) \propto \frac{1}{2\pi\sigma^2/\Delta} \exp\left\{-\frac{\left(y_R^{\mathrm{Re}}[1] - \mathrm{Re}\left(a\sqrt{2}\right)\right)^2}{2\sigma^2/\Delta}\right\} \cdot \exp\left\{-\frac{\left(y_R^{\mathrm{Im}}[1] - \mathrm{Im}\left(a\sqrt{2}\right)\right)^2}{2\sigma^2/\Delta}\right\}$, and $p_{2N+1}^{0,b} = P(x_A[N+1] = 0, x_B[N] = b\sqrt{2} | y_R[2N+1]) \propto \frac{1}{2\pi\sigma^2/\Delta} \exp\left\{-\frac{\left(y_R^{\mathrm{Re}}[2N+1] - \mathrm{Re}\left(be^{j\phi}\sqrt{2}\right)\right)^2}{2\sigma^2/\Delta}\right\} \cdot \exp\left\{-\frac{\left(y_R^{\mathrm{Im}}[2N+1] - \mathrm{Im}\left(be^{j\phi}\sqrt{2}\right)\right)^2}{2\sigma^2/\Delta}\right\}$.

Given the evidence node values computed by (10) and (11), the message update equations for BP-UPNC could then be derived using the standard *sum-product* formula of BP. The message update equations can be found in our technical report [12]. However, readers familiar with BP can readily derive these equations themselves.

Given the message update equations, we then compute the right-bounded messages and left-bounded messages in Fig. 2. By right-bound messages, we mean messages propagating from left to right, such as that from node $x^{n,n}$ to node $\psi_e(x^{n,n}, x^{n+1,n})$ and that from $\psi_e(x^{n,n}, x^{n+1,n})$ to $x^{n+1,n}$. We start with the leftmost right-bound message (i.e., that from $x^{1,0}$ to $\psi_o(x^{1,0}, x^{1,1})$) and compute the right-bound messages iteratively from left to right. Similar comments apply to left-bound messages.

Since the Tanner graph for BP-UPNC has a tree structure, the decoding of the joint probability $P(x_A[n], x_B[n]|Y_R)$ can be done by passing the messages only once in each direction [21]. Note that we only use $P(x_A[n], x_B[n]|Y_R)$, and not $P(x_A[n+1], x_B[n]|Y_R)$, when applying the PNC mapping in (9), because the information required to get $P(x_A[n] \oplus x_B[n]|Y_R)$ is fully captured in $P(x_A[n], x_B[n]|Y_R)$.

C. Numerical Results

This section presents simulation results for BP-UPNC. We compare the performance of asynchronous unchannel-coded PNC with that of the perfectly synchronized case [1].

1) Summary of Results: Our simulations yield the following findings:

- For BPSK, the 3 dB BER performance penalty due to phase or symbol asynchrony using the decoding methods in [1], [3] is reduced to less than 0.5 dB with our method.
- For QPSK, the BER performance penalty due to phase asynchrony can be as high as 6-7 dB when the symbols are aligned, as can be seen from Fig. 3(a). However, with half symbol misalignment, our method can reduce the penalty to less than 1 dB.

The general conclusion is that misalignment makes the system more robust against phase asynchrony. If one could control the symbol timings (e.g., [23] presented a method to control the timings of symbols from different sources), it would be advantageous to deliberately introduce a half symbol offset in unchannel-coded PNC. This conclusion is supported by the results in Fig. 3, and is detailed below.

2) Detailed Description: Fig. 3 shows the simulation results for QPSK. The BPSK results can be found in [12]. Essentially, with our BP-UPNC algorithm, the penalty due to either symbol or phase asynchrony is small (less than 0.5 dB). The x-axis is the average SNR per bit of both end nodes, and the y-axis is the BER for the uplink XORed value $S_A \oplus S_B$.

For each data point, we simulate 10,000 packets of 2,048 bits, with relative symbol offsets of 0, 1/4 and 1/2. The performance of 3/4 symbol offset will be similar to that of 1/4 if we aim to decode $s_A[n+1] \oplus s_B[n]$ rather than $s_A[n] \oplus s_B[n]$. We use the synchronous case as a benchmark to evaluate the effect of asynchrony. For QPSK with symbol synchrony but without phase synchrony, as can be seen from Fig. 3(a), the BER performance penalty can be as large as 6 to 7 dB. However, with a half symbol misalignment, as can be seen from Fig. 3(b), the penalty is reduced to within 1 dB (compared with the benchmark case where symbol and phase are perfectly synchronized). In other words, symbol asynchrony can ameliorate the penalty due to phase asynchrony. This can be explained by the "diversity and certainty propagation" effects elaborated in the next subsection. In addition, we note from Fig. 3(b) that when there is a half symbol offset, the phase offset effect becomes much less significant. Specifically, the spread of SNRs for a fixed BER under different phase offsets is less than 0.5 dB.

D. Diversity and Certainty Propagation

In QPSK, each symbol has four possible values. Thus, the joint symbol from both sources has 16 possible values. The constellation map of the joint symbol varies according to the phase offset. Fig. 4(a) shows the constellation map of a joint symbol when the phase offset is $\pi/4$, where the 16 diamonds corresponding to the 16 possibilities. For example, for synchronous PNC, a point with value $1+(1-\sqrt{2})j$ corresponds to the joint symbol $x^{n,n} = (1+j, -1-j)$ in Fig. 4(a) due to the phase shift (i.e., $1+(1-\sqrt{2})j = (1+j)+(-1-j)e^{j\pi/4}$). In PNC, the 16 possibilities need to be mapped to four XOR possibilities for the PNC symbol. In Fig. 4(a), the diamonds are grouped into groups of four different colors. The diamonds of the same color are to be mapped to the same XOR PNC symbol according to $x_A \oplus x_B$. In this mapping process, some of the constellation points are more prone to errors than other constellation points, and the BER is dominated by these bad constellation points.

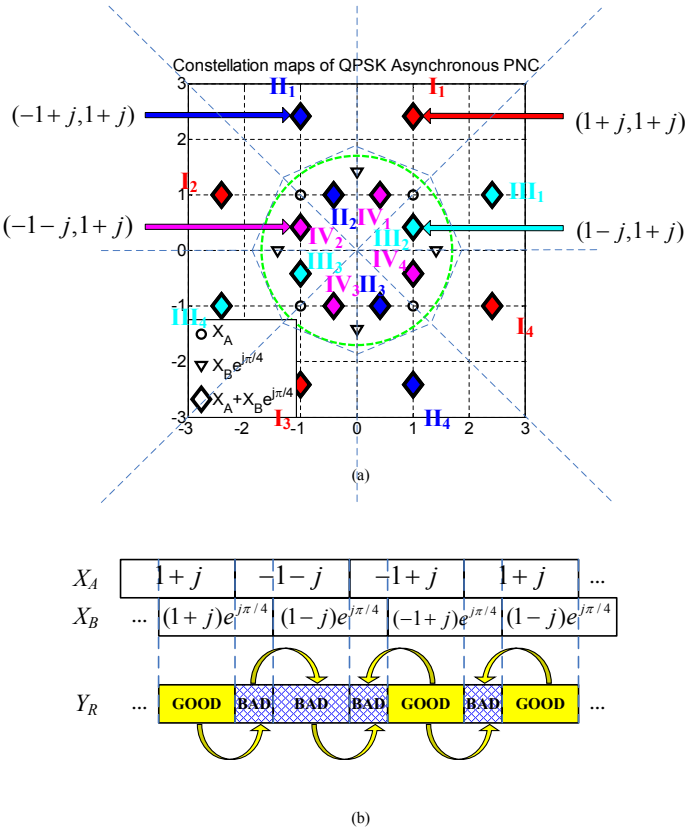


Fig. 4. Certainty propagation illustration. (a) Constellation map for phase asynchronous QPSK PNC (with symbol synchronization). There are four XORed PNC symbols for QPSK. Constellation points of the same color in the figure should be mapped to a same PNC symbol. And the amplitude of each symbol is $\sqrt{2}$; (b) An illustration of certainty propagation. The yellow symbols denote the good constellation points and the white symbols with grids denote the bad constellation points of (a), respectively.

For symbol-synchronous phase-asynchronous unchannel-coded PNC, with reference to Fig. 4(a), the error probability for PNC mapping can be calculated by summing over the probability density areas that fall outside the correct decoding region. The decision region can be divided into 16 areas, with each area corresponding to the decoding area of one pair of symbols, e.g., $x_A = 1+j$, $x_B = 1+j$. For illustration, let us suppose that the joint symbol being transmitted is $\mathrm{I}_1 = (1+j, 1+j)$. The correct XOR mapping is therefore $1+j$. The decoding will be correct if the observation y_R falls within the decoding areas associated with I_1, I_2, I_3, and I_4. Thus, the error probability given the transmitted joint symbol is $(1+j, 1+j)$ is (note: for notational simplicity, we use I_k to denote the joint symbol as well as the decoding region mapped to the joint symbol below)

$$\iint\limits_{y_R \notin \mathrm{I}_k \forall k} \frac{1}{2\pi\sigma^2} \exp\left\{-\frac{\left(y_R^{\mathrm{Re}} - 1/\sqrt{2} - 1/\sqrt{2}\right)^2}{2\sigma^2}\right\} \cdot \exp\left\{-\frac{\left(y_R^{\mathrm{Im}} - 1/\sqrt{2} - 1/\sqrt{2}\right)^2}{2\sigma^2}\right\} dxdy. \quad (12)$$

The error probability conditioned on other joint symbols being transmitted can be obtained in a similar manner. In general, a closed-form expression for the error probability is

difficult to get, since the 2-D integration in (12) is carried out over irregular regions. Even if we only consider the dominant region with the highest probability, I_1, in the above integration, a closed-form is still hard to get. Nevertheless, we do not need closed-form expressions to see that the error probabilities for some joint symbols will be higher than those of other joint symbols. For example, for the joint symbol IV_1, with reference to Fig. 4(a), the correct decision region is much smaller compared with that of I_1. This is because other constellation points are closer to IV_1 in the 2-D space. In particular, the dominant region IV_1 given joint symbol IV_1 is transmitted is considerably smaller than the dominant region I_1 given joint symbol I_1 is transmitted.

With reference to Fig. 4(a), the eight diamonds within the green circle are "bad constellation points". The two closest constellation points adjacent to a bad constellation point are mapped to different XOR values, but the distances between them and the bad constellation point are small. By contrast, the eight points outside the green circle are "good constellation points" because the distances to other constellation points are large. When symbols are synchronized (i.e., $\Delta = 0$), there are altogether N joint symbols in a packet. On average, half of them will be bad constellation points with high BER. The overall error probability associated with the symbol-synchronous case will be dominated by the poor performance associated with the bad constellations points. This explains the large phase penalty observed in Fig. 3(a).

Now, consider what if there is a symbol offset, say $\Delta = 0.5$. We have $2N + 1$ joint symbols, out of which about N will be good constellation points. A symbol from a source is combined with two symbols from the other source in two joint symbols received at the relay. Both the joint symbols have to be bad for poor performance. Thus, the diversity itself may give some improvement. In addition, there is a certainty propagation effect, as explained below.

Consider a good constellation point, say $x^{n,n} = (x_A[n], x_B[n]) = \mathrm{I}_1 = (1 + j, 1 + j)$. For this point, we may be able to decode not just the XOR, but the individual values of $x_A[n]$ and $x_B[n]$ with high certainty. In particular, we may know that $x_B[n] = 1 + j$ with high certainty. Now, suppose that the next joint symbol $x^{n+1,n} = (x_A[n + 1], x_B[n]) = \mathrm{III}_2 = (1 - j, 1 + j)$, which is a bad constellation point. But we note that III_2 is a bad constellation point only when both $x_A[n+1]$ and $x_B[n]$ in $x^{n+1,n}$ are unknown. Given that $x_B[n]$ is known with high certainty from $x^{n,n}$, and it is $1 + j$, the number of possible constellation points for $x^{n+1,n}$ is only four rather than 16. Namely, besides the correct point III_2, the three other possibilities are I_1, II_1, and IV_2 in Fig. 4(a). These four points are far apart and thus the probability of correctly decoding III_2 becomes much larger. We see that the certainty in $x_B[n] \in x^{n,n}$ is propagated to $x^{n+1,n}$ so that $x^{n+1,n}$ can also be decoded with high certainty. The certainty of $x_A[n+1]$ in $x^{n+1,n}$ in turn can propagate to $x^{n+1,n+1}$, and so on and so forth. In general, certainty can propagate along successive symbols from left to right, as well as from right to left, as shown in Fig. 4(b), significantly reducing BER. This gives an intuitive explanation for the results in Fig. 3(b), in which the phase penalty is reduced when there is a offset between the symbols from the two end nodes.

V. Channel-coded PNC

This section generalizes the BP algorithm for application in channel-coded PNC. In the last section, we show that the BP algorithm can exploit symbol offset in unchannel-coded PNC to reduce the phase asynchrony penalty. Interestingly, we find that, in channel-coded PNC the phase penalty may become a phase reward. More importantly, we find that channel coding can have the effect of making the system performance a lot less sensitive to both symbol and phase asynchronies.

The structure of channel-coded PNC is shown at the bottom part of Fig. 1. The channel inputs X_i are the channel-coded symbols constructed by performing channel coding operation Γ_i on the source symbols S_i, $i \in \{A, B\}$. Each source node has M symbols with a coding rate of M/N. The relay R performs matched-filtering and sampling to get Y_R. It transforms Y_R to a channel-coded network-coded packet X_R with N symbols.

In this paper, we assume all nodes use the same channel code. That is $\Gamma_i = \Gamma$, $\forall i \in \{A, B, R\}$. In particular, we use the Repeat Accumulate (RA) channel code [9], [18], [24] to illustrate our main ideas and for our numerical studies. Other channel codes amenable to decoding by the BP algorithm can also be used.

There are several ways to construct belief-propagation algorithms for channel-coded PNC (BP-CPNC). We study two such BP-CPNC methods, Jt-CNC and XOR-CD (to be detailed in Sections V-B and V-C).

A. Channel-decoding and Network-Coding (CNC) Process

Recall that we wish to perform network coding on the received overlapped channel-coded packets. Specifically, based on the received signal Y_R, the relay wants to produce an output packet $X_R = f(Y_R)$ for broadcast to nodes A and B, and X_R is an estimate of $\Gamma(S_A \oplus S_B)$.

We could get X_R directly from the received packet Y_R without first channel decoding the source packets (i.e., we simply map Y_R to X_R on a symbol-by-symbol basis as in unchannel-coded PNC without performing channel decoding). We could also try to decode the XORed source packets $S_A \oplus S_B$ from Y_R, and then re-channel encode them to get X_R. In this paper, we consider the second method. The second method generally has better performance because the relay performs channel decoding to remove errors. The first method corresponds to end-to-end channel-coded PNC while the second method corresponds to link-by-link channel-coded PNC [9], [25].

The basic idea in link-by-link channel-coded PNC is shown in Fig. 5. It consists of two parts.

Part 1: The operation performed by the first part is referred to as the Channel-decoding and Network-Coding (CNC) process in [9]. It maps Y_R to $S_A \oplus S_B$. Note that the number of symbols in Y_R is more than the number of symbols in $S_A \oplus S_B$ because of channel coding. Importantly, CNC involves both channel decoding and network coding, since CNC decodes the received signal Y_R not to S_A and S_B individually, but to the network-coded source packet $S_A \oplus S_B$.

Part 2: The relay channel encodes $S_A \oplus S_B$ to $X_R = \Gamma(S_A \oplus S_B)$. The relay then broadcast $\Gamma(S_A \oplus S_B)$ to nodes A and B.

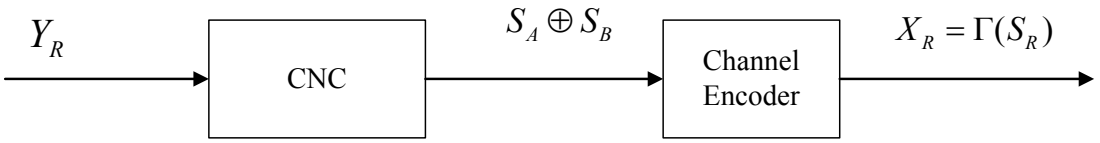


Fig. 5. Link-by-link channel-coded PNC, including the Channel-decoding and Network-Coding (CNC) process and the channel re-encoding process.

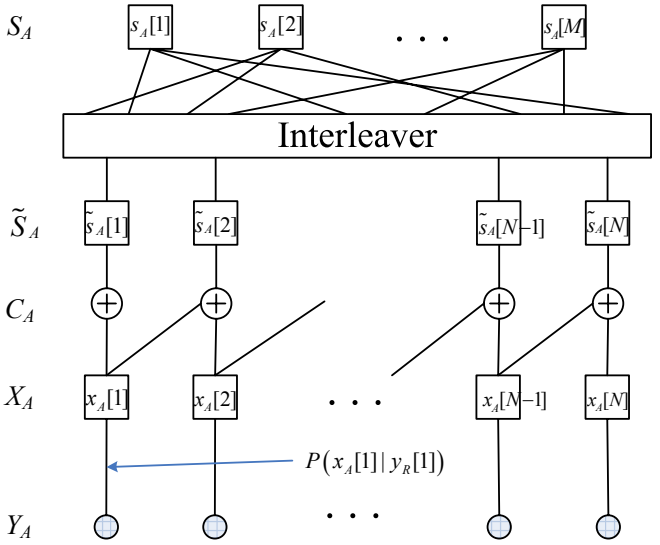


Fig. 6. Tanner graph for standard RA code.

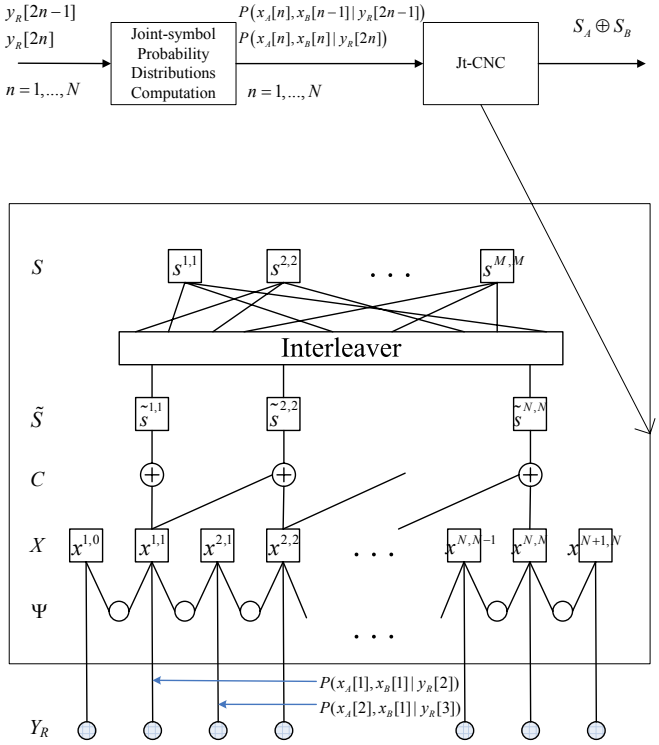


Fig. 7. Tanner graph for Jt-CNC. Upper part: Jt-CNC schematic diagram; Lower part: Tanner graph for Jt-CNC in asynchronous channel-coded PNC. $s^{m,m}$ stands for $(s_A[m], s_B[m])$, $x^{n,n}$ stands for $(x_A[n], x_B[n])$, and $\tilde{s}^{n,n}$ stands for $(\tilde{s}_A[n], \tilde{s}_B[n])$, which are the repeated symbols after the interleaver.

Note that the channel coding in *Part 2* is exactly the same as that in conventional channel-coded point-to-point communication link. Thus, the new distinct element introduced by PNC is the CNC process in *Part 1*. As mentioned in [9] and [25], the CNC component is unique to PNC, and different designs can have different performances and different implementation complexities. We refer interested readers to [25] for a general discussion on different CNC designs. In this paper, we will study two specific CNC designs referred to as Jt-CNC and XOR-CD in Sections V-B and V-C, respectively.

B. Jt-CNC: A Joint Channel-decoding and Network-Coding Scheme

In this subsection, we investigate the Jt-CNC scheme in which channel decoding and network coding are performed jointly in an integrated manner. The two end nodes transmit their packets X_A and X_B with channel coding operation Γ on their corresponding source packets X_i (i.e., $X_i = \Gamma(S_i)$ with $i \in \{A, B\}$). Relay R receives the combined signals with $\phi \neq 0$ and/or $\Delta \neq 0$. The received baseband packet at R is $Y_R = X_A + X_B + W_R$ with $2N + 1$ symbols. In particular, $X_A = (x_A[1], x_A[1], \ldots, x_A[N], x_A[N], 0)$ and $X_B = (0, x_B[1], x_B[1], \ldots, x_B[N], x_B[N])$, respectively. Relay R transforms Y_R into a network-coded packet $X_R = f(Y_R)$ with N symbols for transmission in the downlink phase. Before presenting the scheme, we provide a quick overview of the RA code as the background material.

1) Overview of RA code: Fig. 6 shows the encoding (decoding) Tanner graph [21] of a standard RA code when used in a point-to-point communication link. The encoding process is as follows: from top to bottom in Fig. 6, each source symbol in S_A is first repeated q times ($q = 3$ in Fig. 6); then an interleaver is applied to decorrelate the adjacent repeated bits to get $\tilde{S}_A$; after that, the scrambled bits are accumulated by running XOR operations (represented by nodes C_A) to get the channel coded bits (nodes X_A) for transmission.

In PNC, the relay receives the channel-coded signals from nodes A and B simultaneously. We need to construct a modified Tanner graph for decoding purposes at the relay. We now present the Tanner graph of Jt-CNC. The Tanner graph of Jt-CNC is designed for the computation of $P(s_A[m], s_B[m]|Y_R)$ for $m = 1, \ldots, M$. Once these probabilities are found, we can then obtain the PNC mapping by

$$\begin{aligned} s_A[m] \oplus s_B[m] &= \arg\max_s P(s_A[m] \oplus s_B[m] = s|Y_R) \\ &= \arg\max_s \sum_{\substack{(s_A[m], s_B[m]): \\ s_A[m] \oplus s_B[m] = s}} P(s_A[m], s_B[m]|Y_R). \end{aligned} \quad (13)$$

2) Tanner Graph of Jt-CNC and Message Update Rules: As in the unchannel-coded case, we make use of the oversampled symbols in (4) to construct a Tanner graph [22] as shown in Fig. 7. In the Tanner graph, Y_R is the evidence nodes, and there are $2N + 1$ such nodes. For example, when

RA code with repeat factor of 3 is used, $N = 3M$. In Fig. 7, Ψ is the constraint nodes, and S is the source nodes. Note that Fig. 7 is just the cascade of the Tanner graph in Fig. 2 (without the dotted lines and the PNC mapping represented by the triangles) and the Tanner graph in Fig. 6, except that in Fig. 7 each source node S (or code node X) is a pair that contains $(s_A[n], s_B[n])$ (or $(x_A[n], x_B[n])$), rather than just $s_A[n]$ (or $x_A[n]$).

When decoding, what is fed to the Tanner graph through the evidence nodes at the bottom are $P(x_A[n], x_B[n-1]|y_R[2n-1])$ and $P(x_A[n], x_B[n]|y_R[2n])$ for $n = 1, \ldots, N$, as illustrated in Fig. 7. For QPSK, for example, these input probabilities take on values as in (10) and (11).

With the above evidence node values and the Tanner graph structure, we could then derive the message update equations for Jt-CNC using the standard sum-product formula of BP. We omit the details and refer interested readers to the Appendix II of our technical report [12]. Given the message update equations, we update the messages in the Tanner graph of Fig. 7 in the following sequence: i) right-bound messages below X; ii) left-bound messages below X; iii) upward-bound messages above X; iv) downward-bound messages above X. This sequence of message updates is repeated until the joint probabilities for the source nodes $P(s_A[m], s_B[m]|Y_R)\ \forall m$ converge.

Compared with the Tanner graph for BP-UPNC in Fig. 2, the Tanner graph for Jt-CNC in Fig. 7 does not have a tree structure anymore. Thus, the joint probabilities $P(s_A[m] \oplus s_B[m]|Y_R)$ computed by BP are only approximations [21]. Furthermore, multiple rounds of message updates for the same messages will be needed [21].

C. XOR-CD: A Disjoint Channel-decoding and Network-Coding Scheme

We now look at the XOR-CD scheme in which channel decoding and network coding are performed in a disjoint manner. This scheme was studied in [5]. It was also investigated in [10] in the context of an OFDM-PNC system. Recently, it has been implemented in [11] in the frequency domain via software radio.

Our goal here is to benchmark the performance of Jt-CNC with this scheme. Compared with Jt-CNC, this scheme (referred to as $\text{CNC}_{\text{XOR-CD}}$, or simply XOR-CD) is simpler to implement. The performance, however, is not as good.

The top part of Fig. 8 shows the schematic of XOR-CD. The acronym XOR-CD refers to a two-step process, in which we first apply the PNC mapping on the channel-coded symbols to obtain information on the XOR of the channel-coded symbols: $x_A[n] \oplus x_B[n], n = 1, \ldots, N$; after that, we perform channel decoding on $X_A \oplus X_B$ to obtain $S_R = S_A \oplus S_B$.

The first block of XOR-CD in Fig. 8 computes soft information in the form of the probability distributions of XORed successive symbol pairs: $P(x_A[n] \oplus x_B[n]|Y_R)$ for $n = 1, \ldots, N$. The computation performed by the first block is exactly the same as the PNC mapping in unchannel-coded PNC (see Section IV-B). That is, the first block in Fig. 8 applies the BP-UPNC algorithm proposed in Section IV-B. This block does not exploit of the correlations among the successive symbols induced by channel coding.

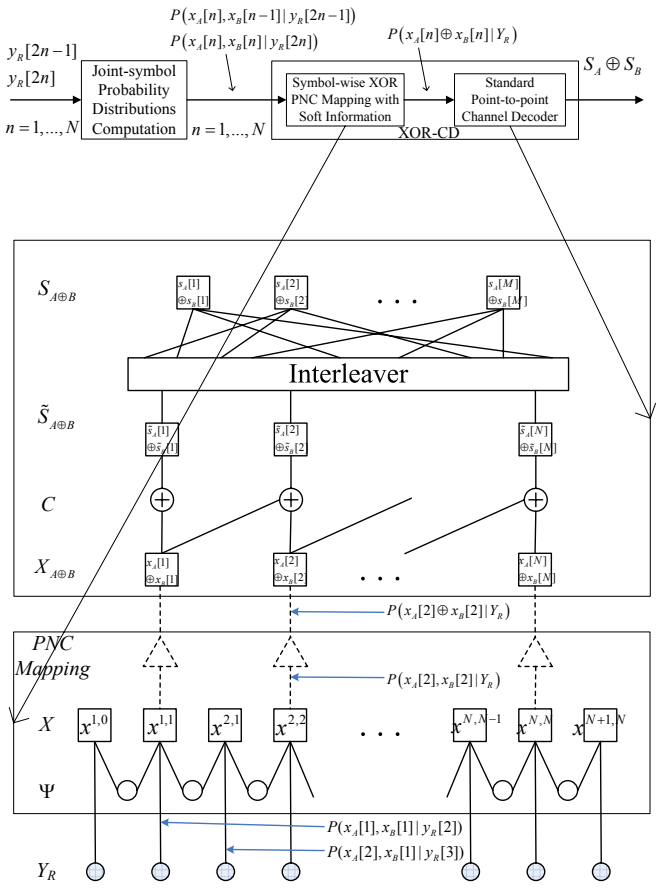


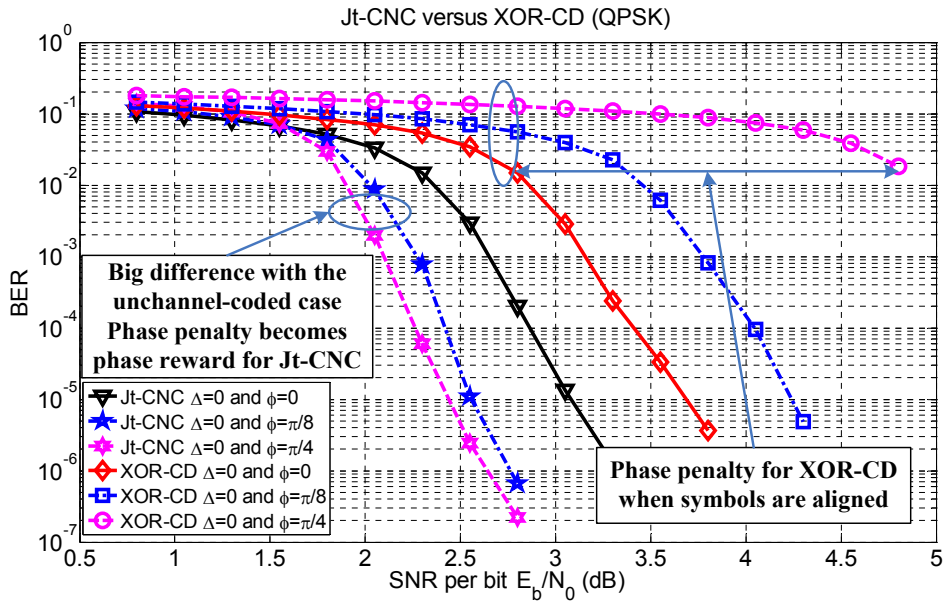
Fig. 8. Tanner graph for XOR-CD. Upper part: XOR-CD schematic diagram; Lower part: Tanner graph for XOR-CD in asynchronous channel-coded PNC. The triangle nodes perform the PNC mapping described in Section IV-B. Note that the channel decoder in the upper block is the standard RA decoder shown in Fig. 6.

The second block makes use of $P(x_A[n] \oplus x_B[n]|Y_R)$ from the first block to perform channel decoding to obtain the pairwise XOR of the source symbols, as shown in the upper rectangular block of the Tanner graph in Fig. 8. The channel decoder in the second block can be exactly the same as the channel decoder of a conventional point-to-point link if Γ is a linear channel code (note: the RA code is linear), as implied by the following result: $\Gamma(S_R) = \Gamma(S_A) \oplus \Gamma(S_B) = X_A \oplus X_B \Rightarrow S_R = \Gamma^{-1}(X_A \oplus X_B)$. This is the reason why XOR-CD is simpler to implement than Jt-CNC[2]. However, the first block loses some useful information in only presenting $P(x_A[n] \oplus x_B[n]|Y_R)$, $n = 1, \ldots, N$, to the second block rather then the joint probabilities $P(x_A[n], x_B[n-1]|y_R[2n-1])$ and $P(x_A[n], x_B[n]|y_R[2n])$, $n = 1, \ldots, N$.

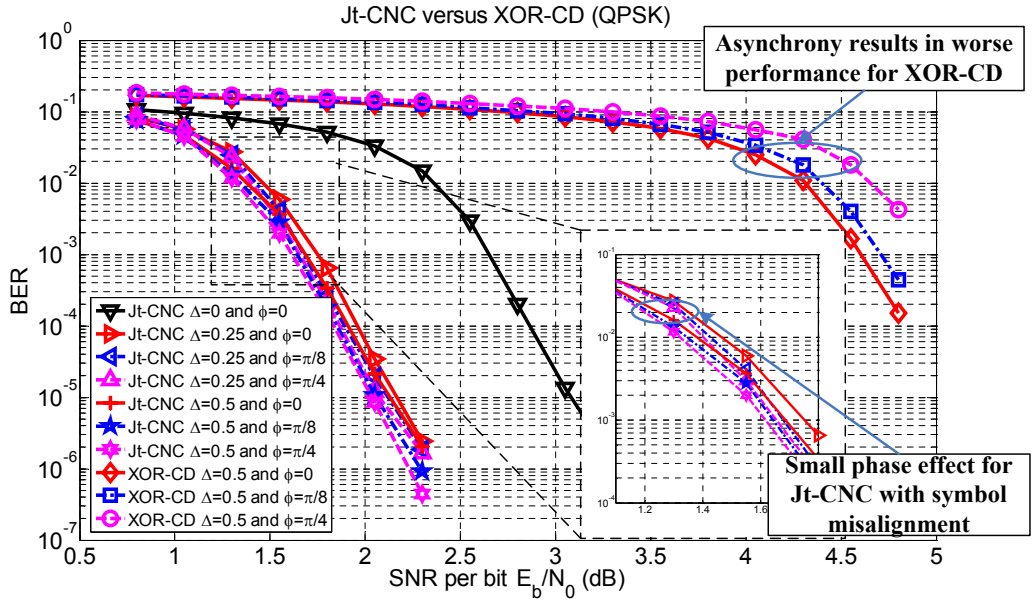
D. Numerical Results

This subsection presents simulation results of Jt-CNC and XOR-CD.

[2]The complexity of Jt-CNC under QPSK modulation is due to the 16 combinations of $(x_A[n], x_B[n])$ and $(s_A[n], s_B[n])$ in the branches of the Tanner graph that the sum-product algorithm has to compute over (see Fig. 7). For XOR-CD, each branch has only 4 combinations in the channel decoding part, thanks to the XOR operation prior to channel decoding (see Fig. 8).



(a)



(b)

Fig. 9. BER of the uplink XORed value $s_A[m] \oplus s_B[m]$ in Jt-CNC and XOR-CD, respectively, for QPSK modulated channel-coded PNC using RA code with repeat factor three: (a) Jt-CNC versus XOR-CD without symbol asynchrony ($\Delta = 0$); (b) Jt-CNC versus XOR-CD with symbol asynchrony ($\Delta \neq 0$). Note that E_b is energy per source bit here.

1) Summary of Results: Our simulations yield the following findings:

- In Jt-CNC, rather than having BER performance penalty due to phase asynchrony (as in BP-UPNC), we have phase reward.
- In Jt-CNC, the performance spread of QPSK channel-coded PNC due to different phase offsets is small (no more than 1 dB), with or without symbol misalignment.
- Jt-CNC achieves significantly better BER performance compared with XOR-CD (3 dB on average).
- In Jt-CNC, as in BP-UPNC, symbol misalignment also desensitizes the system performance to phase asynchrony.

The general conclusion is that in channel-coded PNC, phase and symbol asynchronies are not performance limiting factor with an appropriate CNC algorithm, such as Jt-CNC.

2) Detailed Description:

a) Jt-CNC: We adopt the regular RA code with a coding rate of 1/3 in our simulations. Fig. 9 shows the simulation results of QPSK modulated Jt-CNC scheme with and without symbol misalignment. The simulation results for BPSK modulation can be found in [12], which are qualitatively similar to the QPSK case [12]. The x-axis is the average per-

bit SNR. For fair comparison between unchannel-coded PNC and channel-coded PNC, we shift the curves of the latter by $10\log_{10} 3$ dB to the right to take into account that each bit is repeated 3 times in our RA channel coding. We benchmark the results against the perfect synchronous case where $\Delta = 0$ and $\phi = 0$ [1]. For each data point, we simulated 10,000 packets of 4,096 bits. These 4,096 bits are divided into in-phase and quadrature parts, each having 2,048 bits.

We see from Fig. 9 that, for Jt-CNC, instead of penalty, phase asynchrony actually improves the performance in channel-coded PNC. In the case of QPSK unchannel-coded PNC, recall from Fig. 3(a) that when symbols are aligned, the penalty can be as high as 6-7 dB. However, with channel coding, as can be seen from Fig. 9(a), the penalty goes away and becomes a reward of around 0.5 dB. In the case of symbol misalignments of 1/4 and 1/2 symbol, the reward is around 1 dB.

Another interesting observation is that in the case of symbol misalignment, say $\Delta = 1/4$ and $\Delta = 1/2$, the BER performance is less dependent on the degree of phase asynchrony. In particular, for QPSK, as can be seen from Fig. 9(b), the SNR required to achieve a target BER does not vary much with phase offset.

b) XOR-CD: Let us now look at the performance of XOR-CD. For asynchronous XOR-CD, the PNC XOR processing is first performed on the channel-coded information (using the BP-UPNC method discussed in Section IV-B) to obtain $P(x_A[n] \oplus x_B[n]|Y_R)$, $n = 1, \ldots, N$; after that, channel-decoding is performed on the soft information of $x_A[n] \oplus x_B[n]$ to obtain an estimate of $s_A[n] \oplus s_B[n]$ using the traditional channel decoder for point-to-point communication. The two processes are disjoint.

Fig. 9 shows the BER results of asynchronous XOR-CD. In general, we can see that this scheme, although less complex than Jt-CNC, has significantly worse performance. In addition, instead of phase reward, there is phase penalty. Its performance is far from what could be achieved fundamentally.

The phase penalty is due to its sub-optimality and the absence of Certainty and Diversity Propagation effect, which will be elaborated in the following subsection.

E. Diversity and Certainty Propagation in Jt-CNC

In Section IV-D, we explained that for unchannel-coded PNC, symbol misalignment induces "diversity and certainty propagation" effects that improve the BER performance and make the system robust against phase asynchrony. Similar diversity and certainty propagation effects are also present in channel-coded PNC when Jt-CNC is used. Unlike in unchannel-coded PNC, in which such effects occur only when symbols are misaligned, these positive effects are present in Jt-CNC with or without symbol misalignment. This is because when channel coding is used, the information on each source symbol $s_A[m]$ (or $s_B[m]$) is embedded in multiple channel-coded symbols $x_A[\cdot]$'s (or $x_B[\cdot]$'s) regardless of symbol alignment. The negative effect of a "bad" channel-coded symbol pair $(x_A[i], x_B[i])$ may be overcome by the positive effect of a "good" channel-coded symbol pair $(x_A[j], x_B[j])$ during the channel-decoding process, if there are strong correlations between $(x_A[i], x_B[i])$ and $(x_A[j], x_B[j])$. Take the RA code for example, a channel-coded symbol has a strong correlation with another channel-coded system if the two symbols are separated by only a few hops in the Tanner graph, which allows certainty to propagate. Consequently, thanks to these correlations introduced by channel coding, the issue of phase asynchrony becomes less critical in Jt-CNC. In particular, even when symbols are perfectly aligned, channel-coded PNC with Jt-CNC can remove the phase penalty via diversity and certainty propagation.

For channel-coded PNC with XOR-CD, from Fig. 9 we see that phase asynchrony still imposes significant penalties. Unlike in Jt-CNC, the certainty propagation effect is weakened in XOR-CD. This is because XOR-CD makes use of $P(x_A[n] \oplus x_B[n]|Y_R)$ and not $P(x_A[n], x_B[n]|Y_R)$ in the channel decoding process. That is, the channel decoding process in XOR-CD deals with $x_A[n] \oplus x_B[n]$ rather than $(x_A[n], x_B[n])$. With only $P(x_A[n] \oplus x_B[n]|Y_R)$, certainties in individual symbol $x_A[n]$ or $x_B[n]$ (for "good" symbol pairs) are lost and cannot be propagated (see Section IV-D on the mechanism of certainty propagation, which requires certainties in individual symbols $x_A[n]$ or $x_B[n]$). That is, knowing $x_A[n] \oplus x_B[n]$ with certainty does not tell us anything about the individual symbols $x_A[n]$ and $x_B[n]$. This explains why we do not observe phase reward in XOR-CD.

VI. Conclusions

This paper has investigated the effects of asynchronies in unchannel-coded and channel-coded PNC systems. Our investigations focus on the effects of carrier-phase and symbol-misalignment asynchronies. We use belief propagation (BP) algorithms at the relay to perform the network coding and, in the case of channel-coded PNC, channel decoding operations.

For unchannel-coded PNC, our BP algorithm is an exact maximum likelihood (ML) decoding algorithm to find symbol-wise XOR of the packets from the two end nodes. With our BP ML algorithm, we find that symbol misalignment has an advantage. In particular, with symbol asynchrony, the performance penalty due to phase asynchrony can be reduced drastically, thanks to a diversity and certainty propagation effect in our algorithm. Essentially, the BER performance becomes not very sensitive to different carrier-phase offsets (around 0.5 dB spread for different phase offsets when the symbols are misaligned by half a symbol, as opposed to more than 6 dB when the symbols are perfectly aligned). Our results suggest that if we could control the symbol offset, it would actually be advantageous to deliberately introduce symbol misalignment so that the system is robust against phase offsets.

For channel-coded PNC, we study two schemes, Jt-CNC and XOR-CD. In Jt-CNC, network coding and channel decoding at the relay are performed jointly in an integrated manner. In XOR-CD, network coding is first performed on the channel-coded symbols before channel decoding is applied; that is, the two processes are disjoint. Both Jt-CNC and XOR-CD make use of BP, but in different ways.

Due to its better performance, we believe Jt-CNC is closer to what can ultimately be achieved in the channel-coded PNC system. In particular, the fundamental performance impacts

of phase and symbol asynchronies are better revealed through the performance results of Jt-CNC. We observe an interesting result that instead of performance penalties, symbol and phase asynchronies can actually give rise to performance rewards. In particular, in channel-coded PNC, unlike in unchannel-coded PNC, phase asynchrony is not a performance limiting factor even when symbols are perfectly aligned. Furthermore, the performance spread arising from all combinations of symbol and phase offsets is only slightly more than 1 dB. The intuitive explanation is as follows. With channel-coding, information on each source symbol is encoded into several channel-coded symbols. Jt-CNC makes use of the correlations among different channel-coded symbols to achieve diversity and certainty propagation effects akin to those in symbol-misaligned unchannel-coded PNC.

Prior to this work, it has often been thought that strict synchronization is needed for PNC. Our work suggests, however, that the penalties due to asynchronies can be nullified to a large extent. In unchannel-coded PNC, symbol asynchrony can reduce the negative effects of phase asynchrony significantly. In channel-coded PNC, with the use of Jt-CNC, both phase and symbol asynchronies have small effects on the performance; and these are positive effects.

REFERENCES

[1] S. Zhang, S. C. Liew, and P. P. Lam, "Hot topic: physical layer network coding," in *Proc. ACM MOBICOM 2006*.

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, pp. 1204–1216, 2000.

[3] S. Zhang, S. C. Liew, and P. P. Lam, "On the synchronization of physical-layer network coding," in *Proc. 2006 IEEE Infomation Theory Workshop*.

[4] Y. Hao, D. Goeckel, Z. Ding, D. Towsley, and K. K. Leung, "Achievable rates for network coding on the exchange channel," in *Proc. 2007 IEEE Mil. Comm. Conf.*.

[5] P. Popovski and H. Yomo, "Physical network coding in two-way wireless relay channels," in *Proc. 2007 IEEE Int. Conf. on Comm.*.

[6] S. Zhang, S. C. Liew, and L. Lu, "Physical layer network coding schemes over finite and infinite fields," in *Proc. 2008 IEEE Global Telecom. Conf.*.

[7] T. Koike-Akino, P. Popovski, and V. Tarokh, "Optimized constellations for two-way wireless relaying with physical network coding," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 5, pp. 773–787, June 2009.

[8] D. Wübben and Y. Lang, "Generalized sum-product algorithm for joint channel decoding and physical-layer network coding in two-way relay systems," in *Proc. 2010 IEEE Glob. Telecom. Conf.*.

[9] S. Zhang and S. C. Liew, "Channel coding and decoding in a relay system operated with physical-layer network coding," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 5, pp. 788–796, June 2009.

[10] F. Rossetto and M. Zorzi, "On the design of practical asynchronous physical layer network coding," in *Proc. 2009 IEEE Workshop on SPAWC*.

[11] L. Lu, T. Wang, S. C. Liew, and S. Zhang, "Implementation of physical-layer network coding," submitted to *Physical Commun.* Available: http://arxiv.org/abs/1105.3416.

[12] L. Lu and S. C. Liew, "Asynchronous physical-layer network coding," technical report. Available: http://arxiv.org/abs/1105.3144.

[13] D. To and J. Choi, "Convolutional codes in two-way relay networks with physical-layer network coding," *IEEE Trans. Wireless Commun.*, vol. 9, no. 9, pp. 2724–2729, Sep. 2010.

[14] J. Boutros and G. Caire, "Iterative multiuser joint decoding: unified framework and asymptotic analysis," *IEEE Trans. Inf. Theory*, vol. 48, no. 7, pp. 1772–1793, July 2002.

[15] A. Barbieri, G. Colavolpe, and G. Caire, "Joint iterative detection and decoding in the presence of phase noise and frequency offset," *IEEE Trans. Commun.*, vol. 55, no. 1, pp. 171–179, Jan. 2007.

[16] Y. Zhu, D. Guo, and M. L. Honig, "A message-passing approach to joint channel estimation, interference mitigation and decoding," *IEEE Trans. Wireless Commun.*, vol. 8, pp. 6008–6018, Dec. 2009.

[17] L. Lu, S. C. Liew, and S. Zhang, "Collision resolution by exploiting symbol misalignment," technical report. Available: http://arxiv.org/abs/0810.0326.

[18] ——, "Channel-coded collision resolution by exploiting symbol misalignment," in *Proc. 2010 IEEE Int. Conf. on Commun.*

[19] ——, "Optimal decoding algorithm for asynchronous physical-layer network coding," in *Proc. 2011 IEEE Int. Conf. on Commun.*

[20] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* Morgan Kaufmann, 1988.

[21] J. Yedidia, W. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," *Exploring Artificial Intelligence in the New Millennium*, vol. 8, pp. 236–239, 2003.

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

[23] H. Rahul, H. Hassanieh, and D. Katabi, "Sourcesync: a cooperative wireless architecture for exploiting sender diversity," in *Proc. 2010 ACM SIGCOMM*.

[24] H. Jin, "Analysis and design of turbo like code," Ph.D. thesis, California Institute of Technology, May 2001.

[25] S. C. Liew, S. Zhang, and L. Lu, "Physical-layer network coding: tutorial, survey, and beyond," submitted to *Physical Communication*. Available: http://arxiv.org/abs/1105.4261.

Lu Lu (S'10) received his B. Eng. degree in Electronic Engineering and Information Science from the University of Science and Technology of China (USTC), Hefei, China, in 2007. He is now a Ph.D. candidate in Department of Information Engineering, the Chinese University of Hong Kong (CUHK), Hong Kong. His current research interests include wireless communication in 802.11 networks, multi-user detection, collision resolution, physical-layer network coding and software-define radios.

Soung Chang Liew received his S.B., S.M., E.E., and Ph.D. degrees from the Massachusetts Institute of Technology. From 1984 to 1988, he was at the MIT Laboratory for Information and Decision Systems, where he investigated Fiber-Optic Communications Networks. From March 1988 to July 1993, he was at Bellcore (now Telcordia), New Jersey, where he engaged in Broadband Network Research. He has been Professor at the Department of Information Engineering, the Chinese University of Hong

Prof. Liew's current research interests include wireless networks, Internet protocols, multimedia communications, and packet switch design. Prof. Liew's research group won the best paper awards in IEEE MASS 2004 and IEEE WLN 2004. Separately, TCP Veno, a version of TCP to improve its performance over wireless networks proposed by Prof. Liew's research group, has been incorporated into a recent release of Linux OS. In addition, Prof. Liew initiated and built the first inter-university ATM network testbed in Hong Kong in 1993. More recently, Prof. Liew's research group pioneers the concept of Physical-layer Network Coding (PNC).

Besides academic activities, Prof. Liew is also active in the industry. He cofounded two technology start-ups in Internet Software and has been serving as consultant to many companies and industrial organizations. He is currently consultant for the Hong Kong Applied Science and Technology Research Institute (ASTRI), providing technical advice as well as helping to formulate R&D directions and strategies in the areas of Wireless Internetworking, Applications, and Services.

Prof. Liew is the holder of eight U.S. patents and Fellow of IEEE, IET and HKIE. He currently serves as Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and Ad Hoc and Sensor Wireless Networks. He is the recipient of the first Vice-Chancellor Exemplary Teaching Award at the Chinese University of Hong Kong. Publications of Prof. Liew can be found in www.ie.cuhk.edu.hk/soung.