Joint Channel Coding and Physical-Layer Network Coding Design for Gaussian Two-Way Relay Channels

Jinhong Yuan

University of New South Wales
Sydney, Australia

7 September, 2011
Outline

Introduction

System Model

Error Performance Analysis

Equivalent Tanner Graph and Message Updates

EXIT Chart Analysis and Code Design

Summary
Introduction

Network Coding (NC) and Physical Layer Network Coding (PNC)

- **NC**: Coding better than suboptimal routing [1]
- **PNC** can enhance the throughput of a multi-user wireless network [2].
- **Channel coded PNC (CPNC)** can approach the capacity (upper bound) of a Gaussian two-way relay channel (TWRC) within 1/2 bit [3].
- The pioneering work on designing practical CPNC schemes was reported in [4].

Motivation

- To date, both convolutional coded or repeat-accumulate (RA) coded PNC schemes have been investigated by simulation.
- Some open questions:
  - Whether the conventional good channel codes remain good for PNC?
  - How to design capacity approaching channel codes for PNC schemes?


Consider a Gaussian TWRC where user A and user B exchange information via an intermediate relay R.

- **Two phases:** uplink phase, the users transmit simultaneously to the relay; downlink phase, the relay broadcasts to the users.
- **No direct link** between A and B, single antenna nodes.
- At each node, the received signal is corrupted by AWGN.
System Model

Uplink Phase of CPNC

- **Messages**: Binary message sequences $b_A \in \{0, 1\}^k$ and $b_B \in \{0, 1\}^k$.

- **Encoding**: The messages of users are encoded with the *same binary linear codes*, generating the coded sequences $c_A \in \{0, 1\}^n$ and $c_B \in \{0, 1\}^n$.

  \[ c_A = b_A G, \quad c_B = b_B G \]

  **Generator matrix**: $G$, and **codebook**: $C$.

  **Code rate of each user**: $R = k/n$.

- **Air interface**: The coded sequences are BPSK modulated, obtaining the signal sequence $x_m = 2c_m - 1 \in \{-1, 1\}^n$, $m \in \{A, B\}$. 
The signal received by the relay

\[ y_R = \sqrt{E_s}x_A + \sqrt{E_s}x_B + n_R \]

The relay decodes the network codeword

\[ c_s \triangleq c_A \oplus c_B \]

and computes network codeword’s message

\[ b_s = b_A \oplus b_B \]

Since the same channel code was used by the two users

\[ c_s = b_s G \]

If the computed NC message \( \bar{b}_s \neq b_s \), a computation error is declared.
The recovered NC message $\overline{b}_s$ is re-encoded, BPSK-modulated, generating $x_R$, then broadcasted to the two users.

User $m$, $m \in \{A, B\}$, receives signal

$$y_m = \sqrt{E_R}x_R + n_m$$

Each user decodes the NC message sequence $b_s = b_A \oplus b_B$, and then recovers the other user’s message by XOR-ing $b_s$ with its own message.
System Model

Remarks

- The operation in the downlink is a standard single-user decoding, followed by a simple XOR operation.
- Focus: uplink
- Key problem: how to efficiently recover the NC message sequence $b_s$ (or the Network codeword $c_s$) at the relay in the uplink.
Optimal Decoding of the Network Codeword

Preliminaries

- Recall the received signal at the relay:
  \[ y_R = \sqrt{E_s}(x_A + x_B) + n_R = \sqrt{E_s}x_s + n_R \]
- The relay receives a “ternary superimposed (SI) signal”:
  \[ x_s \triangleq x_A + x_B \in \{-2, 0, 2\} \]
- Then, recovers the binary network codeword \( c_s = c_A \oplus c_B \).
- The maximum likelihood (ML) decoding of the network codeword \( c_s \)
  \[ \bar{c}_s = \arg \max_{c_s \in C} p(y_R|c_s) \]
- Given each network codeword \( c_s \), there is a set of superimposed signals \( x_s \) associated with it
  \[ \mathcal{X}_s(c_s) \triangleq \{x_s = x_A + x_B : c_A \oplus c_B = c_s, c_A, c_B \in C\} \]
The ML rule:

\[
\bar{c}_s = \arg\max_{c_s \in \mathcal{C}} p(y_R | c_s) \\
= \arg\max_{c_s \in \mathcal{C}} \sum_{x_s \in \mathcal{X}_s(c_s)} p(y_R | x_s, c_s) p(x_s | c_s) \\
= \arg\max_{c_s \in \mathcal{C}} \sum_{x_s \in \mathcal{X}_s(c_s)} p(y_R | x_s, c_s) \frac{1}{|\mathcal{X}_s(c_s)|}
\]

Minimum Euclidean distance decoding

The most likely superimposed signal sequence \(\bar{x}_s\) is found by

\[
\bar{x}_s = \arg\max_{x_s \in \mathcal{X}_s} p(y_R | x_s) \\
= \arg\min_{x_s \in \mathcal{X}_s} |y_R - x_s|^2
\]

Mapping the most likely superimposed signal sequence to the network codeword

\(\bar{x}_s \mapsto \bar{c}_s\)
Our goal is to find the **error probabilities** of the above ML computation. To do this, we need

- find the cardinality of set $\mathcal{X}_s(c_s)$
- obtain the **distance spectrum** of the CPNC scheme.
- the error probability not only depends on **Hamming Distance**, but also the **Euclidean Distances** from the superimposed signals, even for binary modulation

**Theorem 1.** The cardinality of the set $\mathcal{X}_s(c_s)$ is given by

$$|\mathcal{X}_s(c_s)| = 2^{\text{Rank}(G^{S^c(c_s)})}$$

$G^{S^c(c_s)}$ is obtained by removing the columns indexed by $t$ where $c_s(t) = 1$ from the original generator matrix $G$. 
Punctured Codebook Approach

Example 1. Consider a (7, 4) Hamming code with

\[ G = \begin{bmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 0 & 1
\end{bmatrix}. \]

Let \( c_s \) be a certain codeword in \( C \), e.g., \( c_s = [0 \ 0 \ 1 \ 1 \ 0 \ 1 \ 0] \). Then, we have \( S(c_s) = \{3, 4, 6\} \). Deleting Column 3, 4 and 6 of \( G \), we obtain

\[ G^{S(c_s)} = \begin{bmatrix}
1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1
\end{bmatrix}. \]

The cardinality of the set \( X_s(c_s) \) is given by

\[ |X_s(c_s)| = 2^{\text{Rank}(G^{S(c_s)})} \]
Why Hamming and Euclidean Distances?

Example with (7, 4) Hamming Code

- Recall $c_s = c_A \oplus c_B$, $x_s = x_A + x_B$;

- Transmitted Codewords
  
  $c_A = 0000000$, $x_A = -1 - 1 - 1 - 1 - 1 - 1 - 1$;
  $c_B = 0111001$, $x_B = -1 + 1 + 1 + 1 - 1 - 1 + 1$;
  $c_s = 0111001$, $x_s = -2 0 0 0 -2 -2 0$;

- If decodes to $c_s^* = 1110000$, $d_H(c_s, c_s^*) = 3$.
  
  $c_A^* = 0000000$, $x_A = -1 - 1 - 1 - 1 - 1 - 1 - 1$;
  $c_B^* = 1110000$, $x_A = +1 + 1 + 1 - 1 - 1 - 1 - 1$;
  $x_s^* = 0 0 0 -2 -2 -2 -2$;
  $d_E^2(x_s, x_s^*) = (-2)^2 + (2)^2 + (2)^2 = 12$.

- $c_A^* = 0001111$, $x_A = -1 - 1 - 1 + 1 + 1 + 1 + 1$;
  $c_B^* = 1111111$, $x_A = +1 + 1 + 1 + 1 + 1 + 1 + 1$;
  $x_s^* = 0 0 0 +2 +2 +2 +2$;
  $d_E^2(x_s, x_s^*) = (-2)^2 + (-2)^2 + (-4)^2 + (-4)^2 + (-2)^2 = 44$.

- $c_A^* = 0010011$, $x_A = -1 - 1 + 1 - 1 - 1 + 1 + 1$;
  $c_B^* = 1100011$, $x_A = +1 + 1 - 1 - 1 - 1 + 1 + 1$;
  $x_s^* = 0 0 0 -2 -2 +2 +2$;
  $d_E^2(x_s, x_s^*) = (-2)^2 + (2)^2 + (-4)^2 + (-2)^2 = 28$. 
Error Performance Analysis

Pair-wise error probability

Consider the genuine transmitted signal sequences $x_A$ and $x_B$, $x_s = x_A + x_B$ and its network codeword $c_s$.

Let $c_s^*$ be the wrong network codeword been detected.

The pair-wise error probability (PEP)

$$P(x_s \rightarrow c_s^*)$$

is determined by the distance between two network codewords.

Each competing network codeword $c_s^*$ has a set of superimposed signals $\mathcal{X}_s(c_s^*)$

$$P(x_s \rightarrow c_s^*) = P(x_s \rightarrow \mathcal{X}_s(c_s^*))$$

We partition the competing set $\mathcal{X}_s(c_s^*)$ into subsets according to its Euclidean distance.

$$\mathcal{X}_s^d(x_s^*, c_s) \triangleq \{x_s \in \mathcal{X}_s(c_s) : \|x_s - x_s^*\|^2 = d^2\}$$

We define the pair-wise distance spectrum (PDS) between $x_s^*$ and $\mathcal{X}_s(c_s)$ as

$$\mathbb{J}(x_s^*, c_s) \triangleq \{(d_0, |\mathcal{X}_s^{d_0}(x_s^*, c_s)|), \ldots, (d_N, |\mathcal{X}_s^{d_N}(x_s^*, c_s)|)\}.$$
Error Performance Analysis

Pair-wise Distance Spectrum (PDS)
An illustration of PDS

\[ \triangle: \text{Genuine transmitted superimposed sequence } x_s \]
\[ \bigcirc: \text{The superimposed sequences of a competing set } X_s(c_s^*) \]

The points on the inner-most circle is called the "minimum distance subset".
Theorem 2. The PDS w.r.t. $x_s$ and $\mathcal{X}_s(c_s^*)$ is given by

$$J(x_s, c_s^*) = \frac{A(C^{Sc}(c_s) \cap S^c(c_s^*))}{O(c_s^*)}.$$  

where $A(\cdot)$ is the weight distribution of $C^{Sc}(c_s) \cap S^c(c_s^*)$ and

$$O(c_s) = 2^{nR - \text{Rank}(G^{Sc}(c_s))}.$$  

Corollary 1. The cardinality of the "minimum distance subset" is upper-bounded by

$$|\mathcal{X}_s^{d_0}(x_s^*, c_s)| = 2^{|\text{Rank}(G^{Sc}(c_s)) - \text{Rank}(G^{Sc}(c_s^*) \cap S^c(c_s))|}$$

$$\leq 2^{|S(c_s^*) \cap S^c(c_s)|} = 2^{d_{10}(c_s^*, c_s)}$$

where $d_{10}(c_s^*, c_s) \triangleq |S(c_s^*) \cap S^c(c_s)|$.

Error Performance Analysis

Pair-wise Error Probability Upper Bound

- With the union bound technique, the pair-wise error probability (PEP) that the decoder recovers $c_s^* \neq c_s$ is upper bounded by

$$P_e(x_s, x_s(c_s^*)) \leq \sum_{i=0}^{N} |\chi^d_i (x_s, c_s^*)| \cdot Q\left(\frac{E_s d_H(c_s^*, c_s) + i \cdot 4E_s}{\sigma^2}\right)$$

(3)

where $N = |S(c_s) \cap S^c(c_s^*)|$.

- To compute the PEP union bound, we need to find the PDS $\mathbb{J}(c_s^*, c_s)$.

- For a short code, $\mathbb{J}(c_s^*, c_s)$ can be determined based on Theorem 2.

- However, as $n$ increases, the number of distinct rows in $C^{S^c(c_s^*) \cap S^c(c_s)}$ increases exponentially with $n$ and the task quickly becomes prohibitive.

- To simplify this task, we now consider an upper bound for the high SNR case.
Error Performance Analysis

Asymptotic Pair-wise Error Probability Bound

- **Lemma:** Asymptotically, the PEP upper bound is approximated as

\[
P_e (x_s, \chi_s(c_s^*)) \lesssim |\chi_s^{d_0} (x_s, c_s^*)| Q \left( \sqrt{\frac{E_s d_H(c_s^*, c_s)}{\sigma^2}} \right)
\]

\[
\leq 2^{d_{10}(c_s^*, c_s)} Q \left( \sqrt{\frac{E_s d_H(c_s^*, c_s)}{\sigma^2}} \right)
\]

- This means that at a high SNR, the PEP is only determined by the minimum distance subset.

- For more insight, we consider a single-user one-way relay (OWRC) case. The PEP of this OWRC is

\[
P_e^{SU} (c_s^*, c_s) \leq Q \left( \sqrt{\frac{E_s d_H(c_s^*, c_s)}{\sigma^2}} \right)
\]

- At high SNRs, the PEP of the CPNC over TWRC is approximately **increased by a factor of (at most) \(2^{d_{10}(c_s^*, c_s)}\)** relative to the single-user case.
Error Performance Analysis

Conditional Word Error Probability

- The word error probability (WEP) conditioned on $x^*_s \in \mathcal{X}_s(c^*_s)$ is
  \[ P_e(c^*_s) \lesssim \sum_{c_s \in \mathcal{C}, c_s \neq c^*_s} 2^{d_{10}(c^*_s, c_s)} Q \left( \sqrt{\frac{E_s d_H(c^*_s, c_s)}{\sigma^2}} \right). \]

- The parameter $d_{10}(c^*_s, c_s)$ is codeword-dependent.

- For random codes, we have
  \[ \Pr \left[ \left| d_{10}(c^*_s, c_s) - \frac{d_H(c^*_s, c_s)}{2} \right| < \varepsilon \right] \xrightarrow{n \to \infty} 1 \]
  for an arbitrarily small $\varepsilon$ [5].

- For long linear codes, we may assume
  \[ d_{10}(c^*_s, c_s) \approx \frac{d_H(c^*_s, c_s)}{2}. \]

- The conditional WEP is
  \[ P_e(c^*_s) \lesssim \sum_{d = d_{\min}(\mathcal{C})}^{d_{\max}(\mathcal{C})} A_d(\mathcal{C}) 2^{\frac{d}{2}} Q \left( \sqrt{\frac{E_s d}{\sigma^2}} \right). \]

Error Performance Analysis

Averaged Word Error Probability and Bit Error Probability

- The averaged WEP of the CPNC is
  \[ P_e = \frac{1}{2nR} \sum_{c_s \in C} P_e (c_s) \lesssim \sum_{d=d_{\text{min}}(C)}^{d_{\text{max}}(C)} A_d (C) 2^d Q \left( \sqrt{\frac{E_s d}{\sigma^2}} \right). \]

- The average BEP of the CPNC is
  \[ P_b \lesssim \sum_{d=d_{\text{min}}(C)}^{d_{\text{max}}(C)} B_d (C) 2^d Q \left( \sqrt{\frac{E_s d}{\sigma^2}} \right) \leq \frac{1}{2} \sum_{d=d_{\text{min}}(C)}^{d_{\text{max}}(C)} B_d (C) \exp \left[ -\frac{d}{2} \left( \frac{E_s}{\sigma^2} - \ln 2 \right) \right] \]

  where \( B_d (C) \) is the average information weight w.r.t. all codewords of weight \( d \).

- The BEP of the single-user case is
  \[ P_{b}^{SU} \leq \frac{1}{2} \sum_d B_d (C) \exp \left[ -\frac{d}{2} \frac{E_s}{\sigma^2} \right]. \]

- At high SNRs, the CPNC scheme relative to the single-user case has a performance degradation of approximately \( \ln 2 \) (in linear scale).
Numerical Results

Hamming Coded PNC

- We consider (7,4) Hamming coded PNC.
- Our analytical results match well with the numerical results.
- The SNR gap between the single-user performance and the two-user CPNC scheme is just under $\ln 2$ (in linear scale.)
Numerical Results

Convolutional Coded PNC

- We consider convolutional coded PNC with various coding rates.
- Our analytical results match well with the numerical results.
- The SNR gap between the single-user performance and the two-user CPNC scheme is just under $\ln 2$ (in linear scale.)
Numerical Results

Repeat-Accumulate (RA) Coded PNC

- We consider a RA coded PNC with code rate $R = 3/4$.
- The performance difference of ln2 is very clear.
- The CPNC significantly outperforms the complete-decoding based scheme by a few dBs.

![Graph showing BER vs SNR for different schemes]
Summary of Performance Analysis

- Interesting to know that the SNR loss of the CPNC scheme relative to the single-user case is about $\ln 2$ (in linear scale) at a high SNR, regardless of the code rate.

- This means a conventional good code tends to perform well asymptotically in a CPNC scheme.

Further Questions

- How to design capacity achieving codes for CPNC schemes in the low SNR region?
Revisit System Model

Uplink Phase and Its Signals

- $b_A, b_B \in \{0, 1\}^k$, $c_A, c_B \in \{0, 1\}^k$.
- Superimposed codeword $c_s \triangleq c_A + c_B \in \{0, 1, 2\}^n$, $x_s \triangleq x_A + x_B \in \{-2, 0, 2\}^n$.
- $y_R$ is a noisy observation of $c_s$.
- Relay needs to compute the network-coded (NC) message sequence $b_N = b_A \oplus b_B \in \{0, 1\}^k$. 
Revisit System Model

Relay Operations

- Relay computes $b_N \in \{0, 1\}^k$ from a noisy observation of $c_s \in \{0, 1, 2\}^n$.
- Various network decoding approaches in [1]:
  - **Method 1**: Complete-decode and forward.
    Relay decodes $b_A$ and $b_B$ first, then computes $b_N = b_A \oplus b_B$

- Similar to CNC1 in [1], but iterative MUD/decoding brings a large gain.
- Multiplexing gain loss at high SNR.

Revisit System Model

Relay Operation

- **Method 2**: Compute and forward.
  Relay decodes *superimposed message sequence* $b_s \triangleq b_A + b_B$, from the noisy observation of the superimposed codeword $c_s \triangleq c_A + c_B$.
  Then, map $b_s = b_A + b_B \mapsto b_N = b_A \oplus b_B$.

- Similar to ACNC in [1], only forwards sufficient information.
- Virtual encoder with ternary inputs and outputs needs to be defined.
- For convolutional code, a *super trellis* or the product of the component code trellises will be useful [2].
- For LDPC or RA code, an *equivalent Tanner graph* (ETG) defined over the superimposed messages.

Message bits $b_m(t)$, $t = 1, \cdots, k$, are repeated $\eta$ times, for $\eta = 2, 3, \cdots, d_v$.

Variable node degree distribution is $\lambda_{\eta} : \lambda_{\eta} \geq 0, \sum_{\eta=2}^{d_v} \lambda_{\eta} = 1$.

Interleaved sequence is encoded by a series of parity-check codes of degrees $\psi$, for $\psi = 1, 2, \cdots, d_c$.

Check node degree distribution is $\rho_{\psi} : \rho_{\psi} \geq 0, \sum_{\psi=1}^{d_c} \rho_{\psi} = 1$. 
Equivalent Tanner Graph

ETG for two users

▶ Input $b_s = b_A + b_B$: ternary
▶ Output $c_s = c_A + c_B$: ternary
▶ How to define/exchange/update extrinsic information or log-likelihood ratios?
▶ For code design, how to model the distribution of the $a$ priori information for density evolution or EXIT chart functions?
Message Updates

- Intrinsic information from $y_R$
  - For $j$th superimposed coded symbol
    
    $p_0(j) = P(c_s(j) = 0|y_R(j))$
    $p_1(j) = P(c_s(j) = 1|y_R(j))$
    $p_2(j) = P(c_s(j) = 2|y_R(j))$
  - Represented in log-likelihood ratio (LLR) form
    
    $\text{LLR}(c_s(j)|y_R(j)) = [\Lambda(j), \Omega(j)]$

Primary LLR:

$\Lambda(j) \triangleq \log \left( \frac{p_0(j) + p_2(j)}{p_1(j)} \right)$

Secondary LLR:

$\Omega(j) \triangleq \log \left( \frac{p_0(j)}{p_2(j)} \right)$

- Primary LLR is the LLR of the network coded bits.

$\Lambda(j) \triangleq \log \left( \frac{p_0(j) + p_2(j)}{p_1(j)} \right) = \log \left( \frac{P(c_N(j) = 0|y_R(j))}{P(c_N(j) = 1|y_R(j))} \right)$
Message Updates

Check Node Update Rule

- Update function $f_{\text{CN}}^2$ for degree $\psi = 2$.

\[
\Lambda_Q^{(3)} = \log \left( \frac{1+\exp(\Lambda_P^{(1)}) \exp(\Lambda_P^{(2)})}{\exp(\Lambda_P^{(1)}) + \exp(\Lambda_P^{(2)})} \right)
\]

\[
\Omega_Q^{(3)} = \log \left( \frac{1+\exp(\Omega_P^{(1)}) \exp(\Omega_P^{(2)}) + K_{\text{CN}}}{\exp(\Omega_P^{(2)}) + \exp(\Omega_P^{(1)}) + K_{\text{CN}}} \right)
\]

$p$: a priori, $q$: extrinsic

\[
K_{\text{CN}} = \frac{1+\exp(\Omega_P^{(1)})}{2 \exp(\Lambda_P^{(1)})} \cdot \frac{1+\exp(\Omega_P^{(2)})}{\exp(\Lambda_P^{(2)})}.
\]

- For CN degree $\psi > 2$, successively using $f_{\text{CN}}^2$ to update the extrinsic

\[
(1) \rightarrow (2) \rightarrow f_{\text{CN}}^3 \rightarrow (4) \rightarrow f_{\text{CN}}^2 \rightarrow (\text{temp})
\]

\[
(1) \rightarrow (2) \rightarrow f_{\text{CN}}^2 \rightarrow (3) \rightarrow f_{\text{CN}}^2 \rightarrow (4)
\]
Variable Node Updating Rule

- Variable node (VN) updating rule for degree $\eta$ is

$$\Lambda_Q^{(l)} = (\eta - 2) \log 2 + \sum_{l', l' \neq l}^{\eta} \Lambda_P^{(l')} + K_{VN}$$

$$\Omega_Q^{(l)} = \sum_{l' = 1, l' \neq l}^{\eta} \Omega_P^{(l')}$$

where

$$K_{VN} = \log \left( \frac{1 + \prod_{l' = 1, l' \neq l}^{\eta} \exp \left( \frac{\Omega_P^{(l')}}{\Omega_P^{(l')}} \right)}{\prod_{l' = 1, l' \neq l}^{\eta} \left( 1 + \exp \left( \frac{\Omega_P^{(l')}}{\Omega_P^{(l')}} \right) \right)} \right)$$
Exit Chart

- To illustrate the iteration decoding (mutual information) trajectory.
- To visualize the convergence of the iterative decoding.
- With curve fitting technique, EXIT chart can be used for code design and threshold analysis.
Inner decoder: CN-ACC decoder; Outer decoder: VN decoder.

- Exchange both primary and secondary LLRs.
- Recall: Primary LLR $\Lambda$ linked with NC message $b_N$.
- Tracking only primary LLR: $I_A = I(b_N; \Lambda_P)$, $I_E = I(b_N; \Lambda_Q)$. 
EXIT Chart

EXIT Functions

\[
\begin{align*}
\text{Inner decoder:} & \quad I_E = T_{\text{Inner}} \left( I_A, \mathbb{P}(\Omega_P), \rho_\psi, E_s \right) \\
\text{Outer decoder:} & \quad I_E = T_{\text{Outer}} \left( I_A, \mathbb{P}(\Omega_P), \lambda_\eta \right)
\end{align*}
\]

where \( I_A = I(b_N; \Lambda_P) \).
EXIT Chart

- EXIT functions contains both primary LLR $\Lambda_P$ and secondary LLR $\Omega_P$.
- Primary LLR approaches a consistent Gaussian-like distribution with its mean equal to half of its variance.

$$\Lambda_P = (\sigma^2_\Lambda/2)X_N + n_\Lambda$$

- Secondary LLR $\Omega(j) \triangleq \log(p_0(j)/p_2(j))$ resembles a combination of a Gaussian-like distribution and an impulse at zero.

- Two models to bound the EXIT functions.
  - Model I: Assume perfect secondary LLR information
    $$\dot{\Omega}_P = \begin{cases} 
    +\infty & \text{if } b_s = 0 \\
    0 & \text{if } b_s = 1 \\
    -\infty & \text{if } b_s = 2 
    \end{cases}$$
  - Model II: Assume no a priori information on the secondary LLR $\Omega_P$, i.e., $\bar{\Omega}_P = 0$.

$$T_{\text{Inner}} \left(I_A, \mathbb{P}(\hat{\Omega}_P), \rho_\psi, E_s\right) \geq I_E \geq T_{\text{Inner}} \left(I_A, \mathbb{P}(\bar{\Omega}_P), \rho_\psi, E_s\right)$$
Model I always gives higher mutual information than Model II.
For CN degrees higher than 2, no much performance difference.
Capacity approaching codes usually have higher degree CNs.

Using Model II will be sufficient in our code design.

Model II is a lower bound and it guarantees the convergence.
## Code Design

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Code Type</th>
<th>CN</th>
<th>VN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Physical-layer network coded (PNC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Regular</strong> $R = 1/3$</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2288</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Bi-regular $R = 3/4$</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5424</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2288</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Irregular</strong> $R = 1/3$</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3353</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.2237</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>21</td>
</tr>
<tr>
<td></td>
<td><strong>Irregular</strong> $R = 3/4$</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3221</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3297</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.478</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>Based on complete decoding</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Irregular</strong> $R = 1/3$</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4963</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1144</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.2004</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.870</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.190</td>
</tr>
<tr>
<td></td>
<td><strong>Irregular</strong> $R = 3/4$</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2672</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5915</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.610</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>19</td>
</tr>
</tbody>
</table>
Code Design

\[ R = \frac{3}{4} \]

- IRA code; \( E_b/N_0 = 3.4 \text{ dB} \)
- Bi-regular RA code; \( E_b/N_0 = 6 \text{ dB} \)

\( I_{E,\text{VN decoder}} - I_{A,\text{CN-ACC decoder}} \)
Code Design

$R = 1/3$

Regular RA code: $E_b/N_0 = 2.2$ dB
IRA code: $E_b/N_0 = 2.1$ dB
Code Design

$R = \frac{3}{4}$
Code Design

$R = 1/3$

The graph shows BER (Bit Error Rate) versus Per-user $E_b/N_0$ (dB). The data points are categorized by the type of computation:

- Regular, $\Lambda$-only-computation
- Irregular, $\Lambda$-only-computation
- Regular, $\Lambda$-$\Omega$-computation
- Irregular, $\Lambda$-$\Omega$-computation

The graph indicates performance measures for different types of computations, with $R = 1/3$ being a key parameter.
Code Design

Capacity limit (complete decoding based scheme) $E_b/N_0 = 4.3$ dB

Capacity limit (Gaussian TWRC upper bound) $E_b/N_0 = 1.67$ dB

$R = 3/4$
Code Design

Per-user $E_b/N_0$ (dB)

Capacity limit (Gaussian TWRC upper bound) $E_b/N_0 = -0.5$ dB
Capacity limit (complete decoding based scheme) $E_b/N_0 = 0.8$ dB

$R = 1/3$
Summary

▶ We investigated performance and design of channel coded PNC scheme.
▶ We proposed a method to compute the pairwise distance spectrum of a CPNC scheme, and asymptotically tight WEP and BEP bounds were derived.
▶ The SNR loss of the CPNC scheme relative to the single-user case is about ln 2 (in linear scale) at a high SNR, regardless of the code rate.
▶ Proposed ETG and general message updating rules.
▶ Present two models to bound the EXIT functions for convergency analysis and code designs.
▶ Design capacity approaching IRA coded PNC schemes.

Further Work

▶ Information-theoretic issues: Capacity?
▶ Practical design issues: Synchronization, channel estimation, power/phase controls?
▶ Extensions: Higher level modulation (lattice coding), fading channels, multihop TWRC, MIMO TWRC, multiway, etc?
Acknowledgement

Tao Huang, University of New South Wales

Tom Yang, CSIRO ICT Centre

Ingmar Land, ITR, University of South Australia