The Peeling Decoder : Theory and some Applications

Krishna R. Narayanan
Thanks to Henry Pfister, Avinash Vem, Nagaraj Thenkarai Janakiraman, Kannan Ramchandran, Jean-Francois Chamberland

Department of Electrical and Computer Engineering
Texas A&M University

Introduction

Message passing algorithms

- Remarkably successful in coding theory
- Used to design capacity-achieving codes/decoders for a variety of channels
- Tools have been developed to analyze their performance

Two main goals

Goal 1

Review some developments in modern coding theory and show how to analyze the performance of a simple peeling decoder for the BEC and p-ary symmetric channels.

Two main goals

Goal 1

Review some developments in modern coding theory and show how to analyze the performance of a simple peeling decoder for the BEC and p-ary symmetric channels.

Goal 2

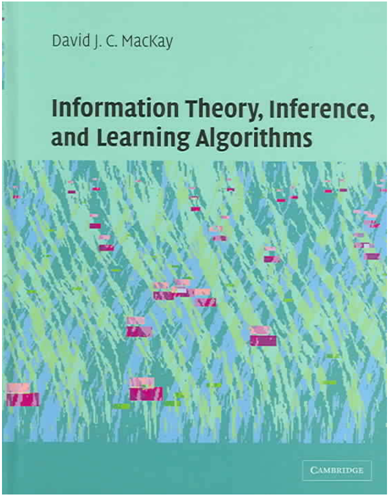
Show that the following problems have the same structure as channel coding problems and show how to use the peeling decoder to solve them.

Problems

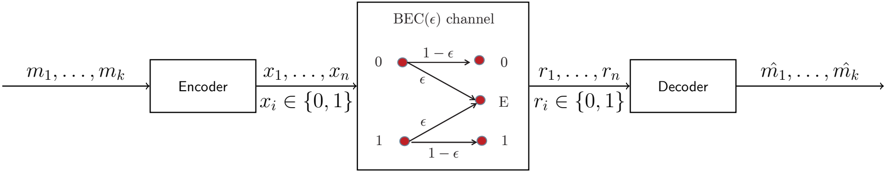
- Uncoordinated massive multiple access
- Sparse Fourier transform (SFT) computation
- Sparse Walsh-Hadamard transform computation
- Compressed sensing
  - Data stream computing
  - Group testing
  - Compressive phase retrieval

Remembering Sir David MacKay

David Mackay's rediscovery of LDPC codes and his very interesting book on Information Theory has undoubtedly had a big influence on the field.



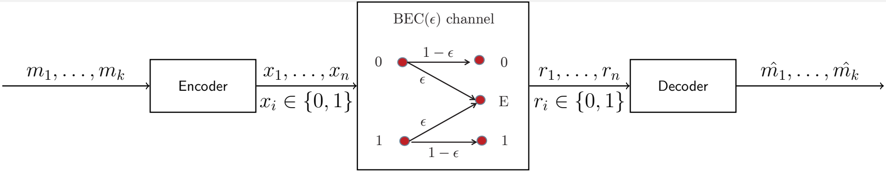
Binary erasure channel (BEC) and erasure correction



Channel coding problem

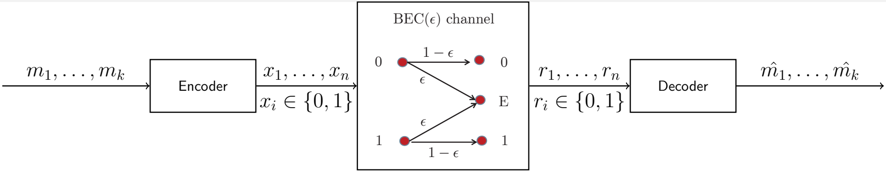
- Transmit a message $\underline{m} = [m_1, \ldots, m_k]^T$ through a binary erasure channel
- Encode the k-bit message $\underline{m}$ into a n-bit codeword $\underline{x}$
- Redundancy is measured in terms of rate of the code $R = k/n$

Binary erasure channel (BEC) and erasure correction



Capacity achieving sequence of codes

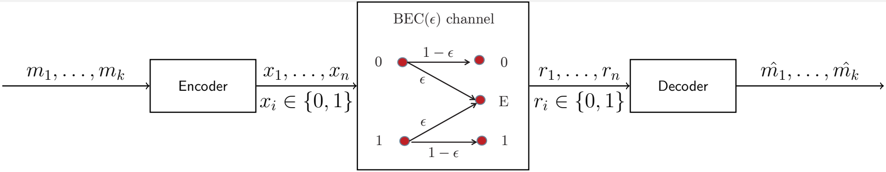
Binary erasure channel (BEC) and erasure correction



Capacity achieving sequence of codes

- Capacity $C(\epsilon) = 1 - \epsilon$

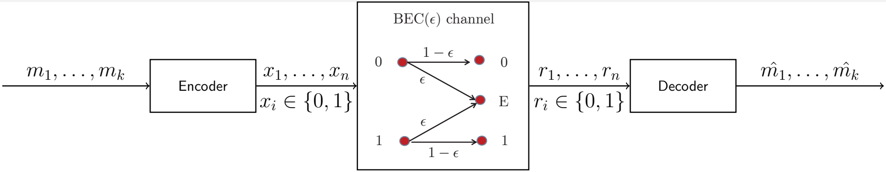
Binary erasure channel (BEC) and erasure correction



Capacity achieving sequence of codes

- Capacity $C(\epsilon) = 1 - \epsilon$
- A sequence of codes $\{\mathcal{C}^n\}$
- Probability of erasure P_e^n
- Rate R^n
- Capacity achieving if $P_e^n \to 0$ as $n \to \infty$ while $R^n \to C$

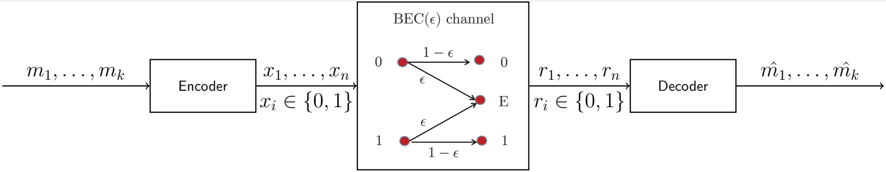
Binary erasure channel (BEC) and erasure correction



Capacity achieving sequence of codes

- Capacity $C(\epsilon) = 1 - \epsilon$
- A sequence of codes $\{\mathcal{C}^n\}$
- Probability of erasure P_e^n
- Rate R^n
- Capacity achieving if $P_e^n \to 0$ as $n \to \infty$ while $R^n \to C$
- Find efficient encoders/decoders in terms encoding and decoding complexities

Binary erasure channel (BEC) and erasure correction



Capacity achieving sequence of codes

- Capacity $C(\epsilon) = 1 - \epsilon$
- A sequence of codes $\{\mathcal{C}^n\}$
- Probability of erasure P_e^n
- Rate R^n
- Capacity achieving if $P_e^n \to 0$ as $n \to \infty$ while $R^n \to C$
- Find efficient encoders/decoders in terms encoding and decoding complexities

Significance of the erasure channel

- Introduced by Elias in 1954 as a toy example
- Has become the canonical model for coding theorists to gain insight

(n, k) Binary linear block codes - basics

$\mathbf{G}$ is a $n \times k$ generator matrix

$$\begin{bmatrix} g_{1,1} & \cdots & g_{k,l} \\ \vdots & \ddots & \vdots \\ \vdots & \ddots & \vdots \\ g_{n,1} & & g_{k,l} \end{bmatrix} \begin{bmatrix} m_1 \\ \vdots \\ m_k \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}$$

Example - (6,3) code

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

(n,k) Binary linear block codes - basics

$\mathbf{G}$ is a $n \times k$ generator matrix

$$\begin{bmatrix} g_{1,1} & \cdots & g_{k,l} \\ \vdots & \ddots & \vdots \\ \vdots & \ddots & \vdots \\ g_{n,1} & & g_{k,l} \end{bmatrix} \begin{bmatrix} m_1 \\ \vdots \\ m_k \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}$$

Example - (6,3) code

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Parity check matrix - $\mathbf{H}$ is a $(n-k) \times n$ matrix s.t. $\mathbf{HG} = \mathbf{0} \Rightarrow \mathbf{H}\underline{x} = 0$

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

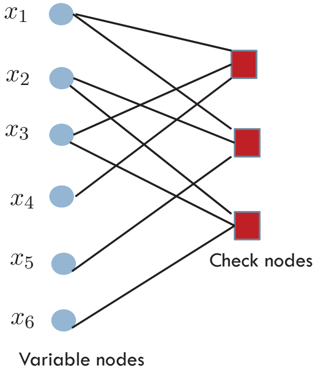
Tanner graph representation of codes

$$
\begin{array}{c}
\quad x_1, x_2, x_3, x_4, x_5, x_6 \\
H = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}
\end{array}
$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

$$x_2 \oplus x_3 \oplus x_6 = 0$$



- Gallager'63, Tanner'81
- Parity check matrix implies that $\mathbf{H}\underline{x} = 0$
- Code constraints can be specified in terms of a bipartite (Tanner) graph

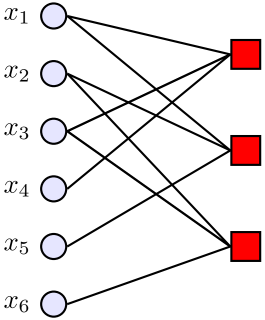
Peeling decoder for the BEC

$$H = \begin{array}{c} x_1, x_2, x_3, x_4, x_5, x_6 \\ \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

$$x_2 \oplus x_3 \oplus x_6 = 0$$



Tanner Graph

- Zyablov and Pinsker'74, Luby et al '95
- Remove edges incident on known variable nodes and adjust check node values
- If there is a check node with a single edge, it can be recovered

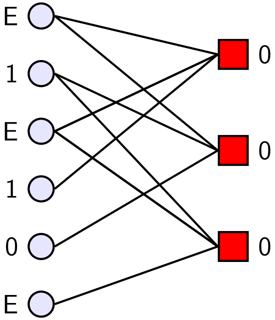
Peeling decoder for the BEC

$$\begin{array}{c} \quad\quad x_1, x_2, x_3, x_4, x_5, x_6 \\ H = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

$$x_2 \oplus x_3 \oplus x_6 = 0$$



Received block

- Zyablov and Pinsker'74, Luby et al '95
- Remove edges incident on known variable nodes and adjust check node values
- If there is a check node with a single edge, it can be recovered

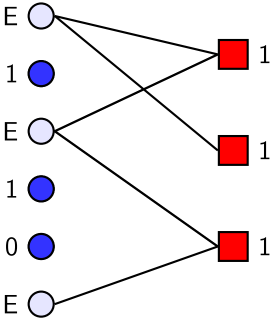
Peeling decoder for the BEC

$$H = \begin{array}{c} \begin{array}{cccccc} x_1, & x_2, & x_3, & x_4, & x_5, & x_6 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

$$x_2 \oplus x_3 \oplus x_6 = 0$$



Peeling Step 1

- Zyablov and Pinsker'74, Luby et al '95
- Remove edges incident on known variable nodes and adjust check node values
- If there is a check node with a single edge, it can be recovered

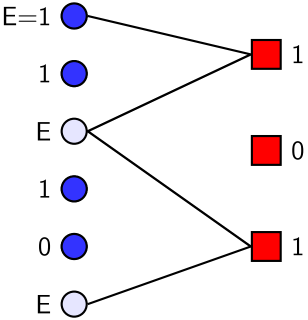
Peeling decoder for the BEC

$$H = \begin{array}{c} x_1, x_2, x_3, x_4, x_5, x_6 \\ \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

$$x_2 \oplus x_3 \oplus x_6 = 0$$



Peeling Step 2

- Zyablov and Pinsker'74, Luby et al '95
- Remove edges incident on known variable nodes and adjust check node values
- If there is a check node with a single edge, it can be recovered

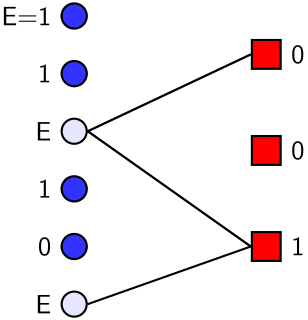
Peeling decoder for the BEC

$$H = \begin{array}{c} x_1, x_2, x_3, x_4, x_5, x_6 \\ \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

$$x_2 \oplus x_3 \oplus x_6 = 0$$



Peeling Step 2

- Zyablov and Pinsker'74, Luby et al '95
- Remove edges incident on known variable nodes and adjust check node values
- If there is a check node with a single edge, it can be recovered

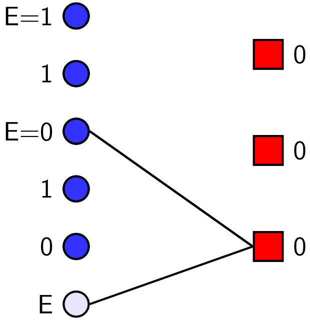
Peeling decoder for the BEC

$$H = \begin{array}{c} \begin{array}{cccccc} x_1, & x_2, & x_3, & x_4, & x_5, & x_6 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

$$x_2 \oplus x_3 \oplus x_6 = 0$$



Peeling Step 3

- Zyablov and Pinsker'74, Luby et al '95
- Remove edges incident on known variable nodes and adjust check node values
- If there is a check node with a single edge, it can be recovered

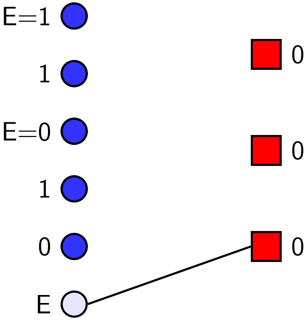
Peeling decoder for the BEC

$$H = \begin{array}{c} x_1, x_2, x_3, x_4, x_5, x_6 \\ \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

$$x_2 \oplus x_3 \oplus x_6 = 0$$



Peeling Step 3

- Zyablov and Pinsker'74, Luby et al '95
- Remove edges incident on known variable nodes and adjust check node values
- If there is a check node with a single edge, it can be recovered

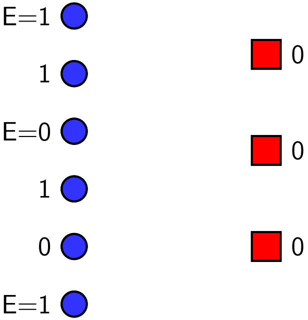
Peeling decoder for the BEC

$$
\begin{array}{c}
\quad x_1, x_2, x_3, x_4, x_5, x_6 \\
H = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}
\end{array}
$$

$$x_1 \oplus x_3 \oplus x_4 = 0$$

$$x_1 \oplus x_2 \oplus x_5 = 0$$

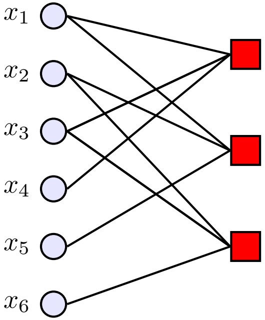
$$x_2 \oplus x_3 \oplus x_6 = 0$$



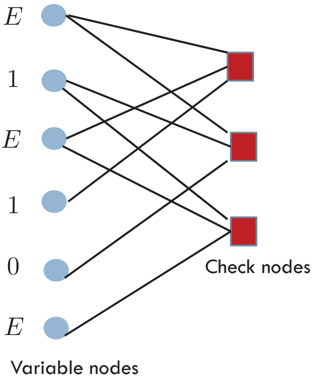
Peeling Step 4

- Zyablov and Pinsker'74, Luby et al '95
- Remove edges incident on known variable nodes and adjust check node values
- If there is a check node with a single edge, it can be recovered

Message passing decoder for the BEC

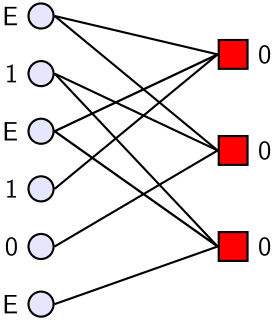


Tanner Graph

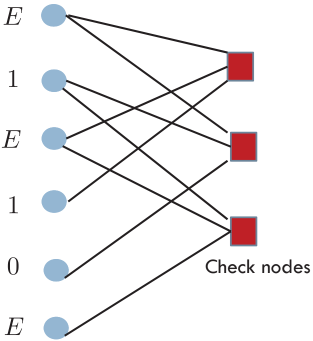


- Pass messages between variable nodes and check nodes along the edges
- Messages $\in$ {value of var node (NE), erasure (E)}
- Var-to-check node message is NE if at least one incoming message is NE
- Check-to-var node message is NE if all other incoming messages are NE

Message passing decoder for the BEC



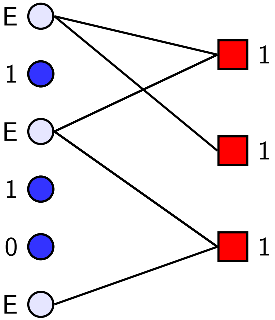
Received block



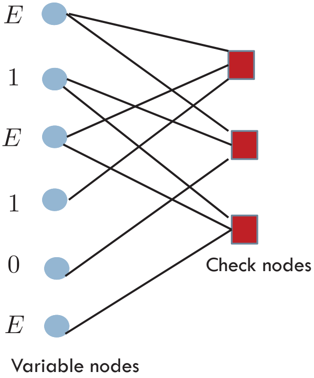
Variable nodes

- Pass messages between variable nodes and check nodes along the edges
- Messages $\in$ {value of var node (NE), erasure (E)}
- Var-to-check node message is NE if at least one incoming message is NE
- Check-to-var node message is NE if all other incoming messages are NE

Message passing decoder for the BEC

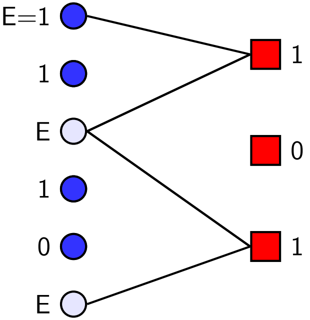


Peeling Step 1

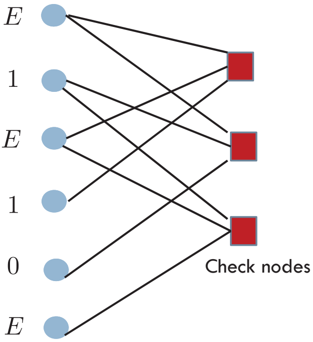


- Pass messages between variable nodes and check nodes along the edges
- Messages $\in$ {value of var node (NE), erasure (E)}
- Var-to-check node message is NE if at least one incoming message is NE
- Check-to-var node message is NE if all other incoming messages are NE

Message passing decoder for the BEC

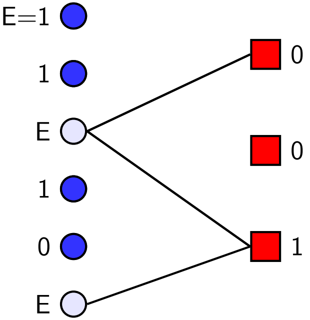


Peeling Step 2

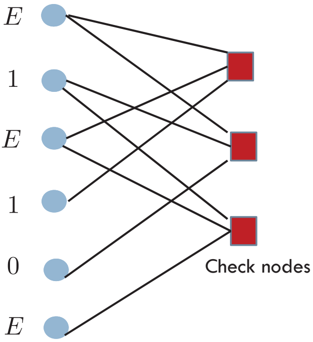


- Pass messages between variable nodes and check nodes along the edges
- Messages $\in$ {value of var node (NE), erasure (E)}
- Var-to-check node message is NE if at least one incoming message is NE
- Check-to-var node message is NE if all other incoming messages are NE

Message passing decoder for the BEC

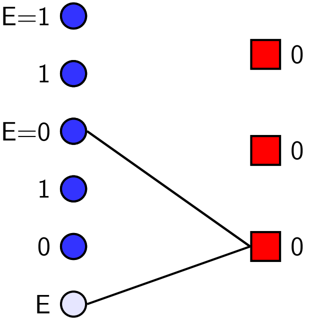


Peeling Step 2

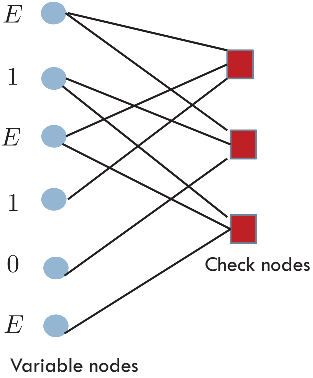


- Pass messages between variable nodes and check nodes along the edges
- Messages $\in$ {value of var node (NE), erasure (E)}
- Var-to-check node message is NE if at least one incoming message is NE
- Check-to-var node message is NE if all other incoming messages are NE

Message passing decoder for the BEC

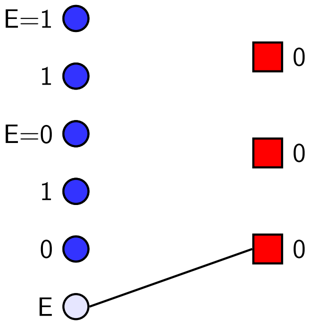


Peeling Step 3

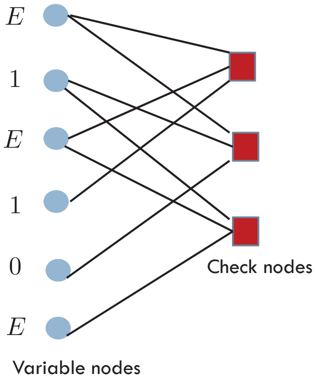


- Pass messages between variable nodes and check nodes along the edges
- Messages $\in$ {value of var node (NE), erasure (E)}
- Var-to-check node message is NE if at least one incoming message is NE
- Check-to-var node message is NE if all other incoming messages are NE

Message passing decoder for the BEC

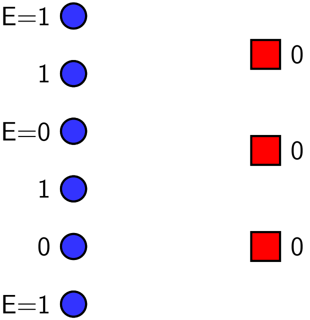


Peeling Step 3

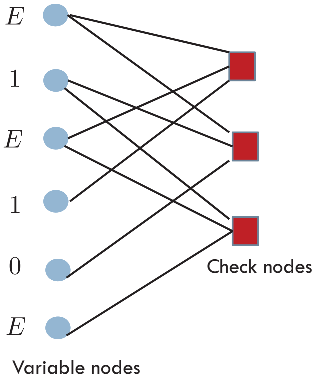


- Pass messages between variable nodes and check nodes along the edges
- Messages $\in$ {value of var node (NE), erasure (E)}
- Var-to-check node message is NE if at least one incoming message is NE
- Check-to-var node message is NE if all other incoming messages are NE

Message passing decoder for the BEC



Peeling Step 4

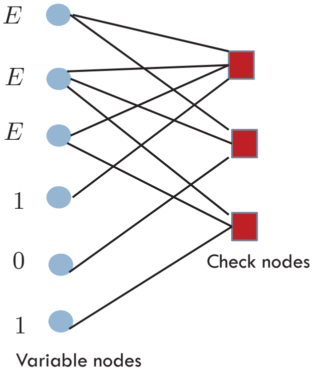


- Pass messages between variable nodes and check nodes along the edges
- Messages $\in$ {value of var node (NE), erasure (E)}
- Var-to-check node message is NE if at least one incoming message is NE
- Check-to-var node message is NE if all other incoming messages are NE

Peeling decoder is a greedy decoder

$$\mathbf{H} = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{matrix} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

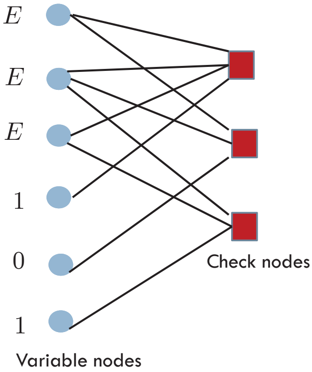
$$\begin{aligned} x_1 \oplus x_2 \oplus x_3 \oplus x_4 &= 0 \\ x_1 \oplus x_2 \oplus x_5 &= 0 \\ x_2 \oplus x_3 \oplus x_6 &= 0 \end{aligned}$$



Peeling decoder is a greedy decoder

$$
\mathbf{H} = \begin{array}{cccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{array}
$$
$$
\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}
$$

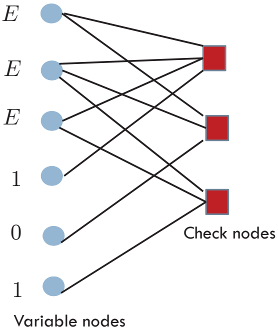
$$
\begin{aligned}
x_1 \oplus x_2 \oplus x_3 \oplus x_4 &= 0 \\
x_1 \oplus x_2 \oplus x_5 &= 0 \\
x_2 \oplus x_3 \oplus x_6 &= 0
\end{aligned}
$$



Linearly independent set of equations

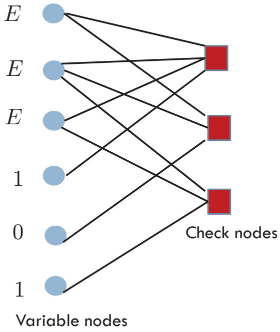
$$
\begin{aligned}
x_1 \oplus x_2 \oplus x_3 &= x_4 \\
x_1 \oplus x_2 &= x_5 \\
x_2 \oplus x_3 &= x_6
\end{aligned}
$$

Degree distributions



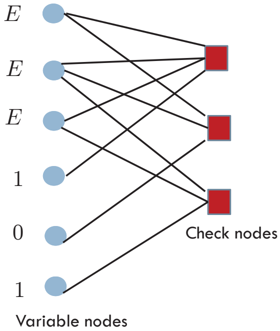
- VN d.d. from node perspective - $L(x) = \sum_i L_i x^i = \frac{3}{6}x + \frac{2}{6}x^2 + \frac{1}{6}x^3$

Degree distributions



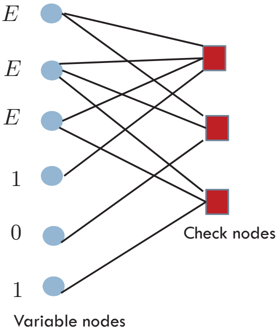
- VN d.d. from node perspective - $L(x) = \sum_i L_i x^i = \frac{3}{6}x + \frac{2}{6}x^2 + \frac{1}{6}x^3$
- VN d.d. from edge perspective - $\lambda(x) = \sum_i \lambda_i x^{i-1} = \frac{3}{10} + \frac{4}{10}x + \frac{3}{10}x^2$

Degree distributions



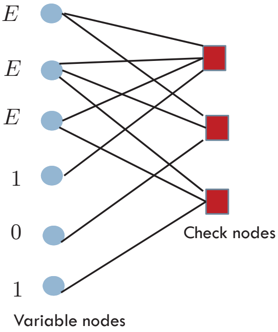
- VN d.d. from node perspective - $L(x) = \sum_i L_i x^i = \frac{3}{6}x + \frac{2}{6}x^2 + \frac{1}{6}x^3$
- VN d.d. from edge perspective - $\lambda(x) = \sum_i \lambda_i x^{i-1} = \frac{3}{10} + \frac{4}{10}x + \frac{3}{10}x^2$
- CN d.d. from node perspective - $R(x) = \sum_i R_i x^i = \frac{2}{3}x^3 + \frac{1}{3}x^4$

Degree distributions



- VN d.d. from node perspective - $L(x) = \sum_i L_i x^i = \frac{3}{6}x + \frac{2}{6}x^2 + \frac{1}{6}x^3$
- VN d.d. from edge perspective - $\lambda(x) = \sum_i \lambda_i x^{i-1} = \frac{3}{10} + \frac{4}{10}x + \frac{3}{10}x^2$
- CN d.d. from node perspective - $R(x) = \sum_i R_i x^i = \frac{2}{3}x^3 + \frac{1}{3}x^4$
- CN d.d. from edge perspective - $\rho(x) = \sum_i \rho_i x^{i-1} = \frac{6}{10}x^2 + \frac{4}{10}x^3$

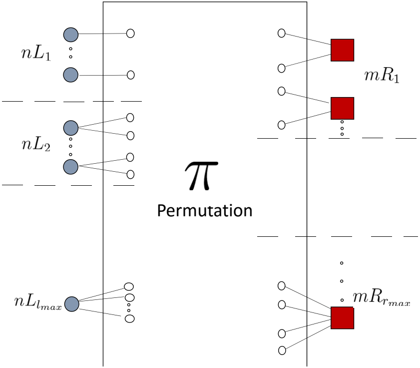
Degree distributions



- Rate - $r(\lambda, \rho) = 1 - \frac{l_{\text{avg}}}{r_{\text{avg}}} = 1 - \frac{\int_0^1 \rho(x)\ dx}{\int_0^1 \lambda(x)\ dx}$

- VN d.d. from node perspective - $L(x) = \sum_i L_i x^i = \frac{3}{6}x + \frac{2}{6}x^2 + \frac{1}{6}x^3$
- VN d.d. from edge perspective - $\lambda(x) = \sum_i \lambda_i x^{i-1} = \frac{3}{10} + \frac{4}{10}x + \frac{3}{10}x^2$
- CN d.d. from node perspective - $R(x) = \sum_i R_i x^i = \frac{2}{3}x^3 + \frac{1}{3}x^4$
- CN d.d. from edge perspective - $\rho(x) = \sum_i \rho_i x^{i-1} = \frac{6}{10}x^2 + \frac{4}{10}x^3$

LDPC code ensemble



LDPC(n, λ, ρ) ensemble

- Ensemble of codes obtained by using different permutations π
- Assume there is only one edge between every var node and check node
- For every n, we get an ensemble of codes with the same (λ, ρ)
- Low density parity check (LDPC) ensemble if graph is of low density

Analysis of the message passing decoder

- If we pick a code uniformly at random from the LDPC(n, λ, ρ) ensemble and use it over a BEC(ϵ) with l iterations of message passing decoding, what will be the probability of erasure P_e^n in the limit $l, n \to \infty$?

Analysis of the message passing decoder

- If we pick a code uniformly at random from the LDPC(n, λ, ρ) ensemble and use it over a BEC(ϵ) with l iterations of message passing decoding, what will be the probability of erasure P_e^n in the limit $l, n \to \infty$?
  - Analyze the average prob. of erasure over the ensemble
  - For almost all realizations P_e^n concentrates around the average

Analysis of the message passing decoder

- If we pick a code uniformly at random from the LDPC(n, λ, ρ) ensemble and use it over a BEC(ϵ) with l iterations of message passing decoding, what will be the probability of erasure P_e^n in the limit $l, n \to \infty$?
  - Analyze the average prob. of erasure over the ensemble
  - For almost all realizations P_e^n concentrates around the average

Relevant literature

- Papers by Luby, Mitzenmacher, Shokrollahi, Spielman, Stemann 97-'02
- Explained in Modern coding theory by Richardson and Urbanke
- Henry Pfister's course notes on his webpage

Analysis of the message passing decoder

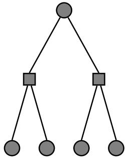
Computation graph

Computation graph $\mathcal{C}_l(x_1, \lambda, \rho)$ of bit x_1 of depth l (l-iterations) is the neighborhood graph of node x_1 of radius l.

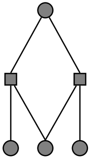
Analysis of the message passing decoder

Computation graph

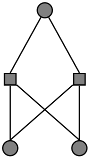
Computation graph $\mathcal{C}_l(x_1, \lambda, \rho)$ of bit x_1 of depth l (l-iterations) is the neighborhood graph of node x_1 of radius l. Consider the example $\mathcal{C}_{l=1}(\lambda(x) = x, \rho(x) = x^2)$



$1 - O(1/n)$



$O(1/n)$

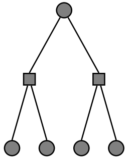


$O(1/n^2)$

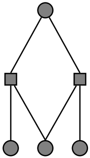
Analysis of the message passing decoder

Computation graph

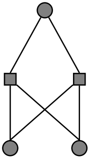
Computation graph $\mathcal{C}_l(x_1, \lambda, \rho)$ of bit x_1 of depth l (l-iterations) is the neighborhood graph of node x_1 of radius l. Consider the example $\mathcal{C}_{l=1}(\lambda(x) = x, \rho(x) = x^2)$



$1 - O(1/n)$



$O(1/n)$



$O(1/n^2)$

Computation tree

For fixed (l_{max}, r_{max}), in the limit of large block lengths a computation graph of depth-l looks like a tree with high probability

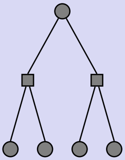
Analysis of the message passing decoder

Computation Tree Ensemble-$\mathcal{T}_l(\lambda, \rho)$

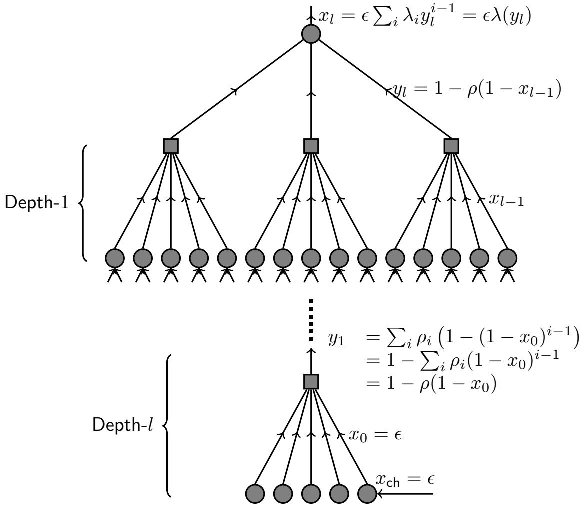
Ensemble of bipartite trees of depth l rooted in a variable node (VN) where

- Root node has i children(CN's) with probability L_i
- Each VN has i children(CN's) with probability λ_i
- Each CN has i children(VN's) with probability ρ_i

Example: $\mathcal{C}_{l=1}(\lambda(x) = x, \rho(x) = x^2)$



Density evolution



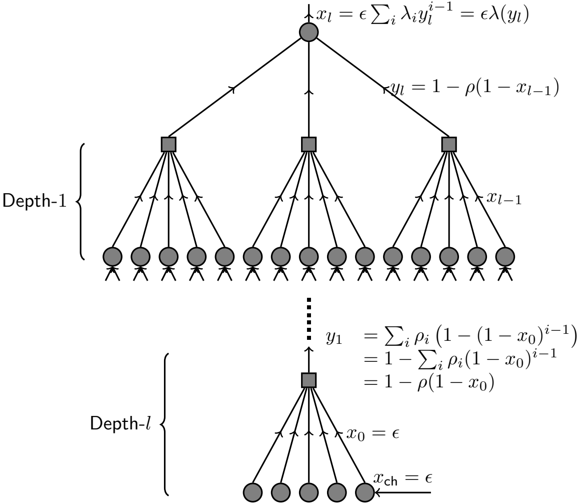
Recall

- $\rho(x) = \sum_i \rho_i x^{i-1}$
- $\sum_i \rho_i = 1$
- $\lambda(x) = \sum_i \lambda_i x^{i-1}$
- $\sum_i \lambda_i = 1$

Recursion

$$x_0 = \epsilon$$

Density evolution



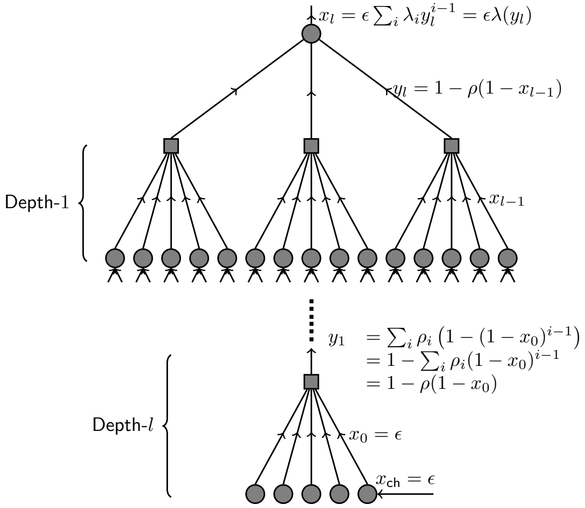
Recall

- $\rho(x) = \sum_i \rho_i x^{i-1}$
- $\sum_i \rho_i = 1$
- $\lambda(x) = \sum_i \lambda_i x^{i-1}$
- $\sum_i \lambda_i = 1$

Recursion

$$
\begin{aligned}
x_0 &= \epsilon \\
y_l &= 1 - \rho(1 - x_{l-1})
\end{aligned}
$$

Density evolution



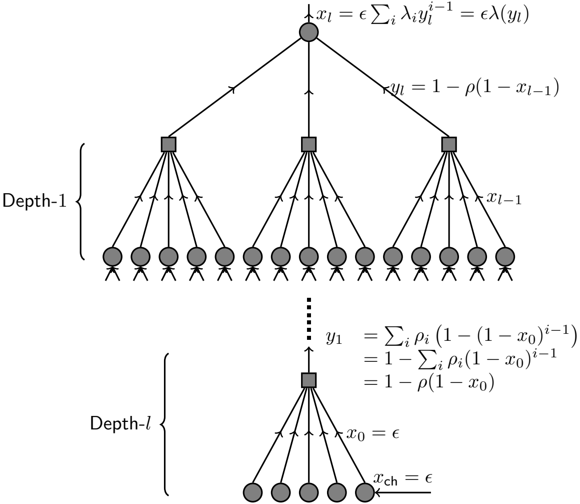
Recall

- $\rho(x) = \sum_i \rho_i x^{i-1}$
- $\sum_i \rho_i = 1$
- $\lambda(x) = \sum_i \lambda_i x^{i-1}$
- $\sum_i \lambda_i = 1$

Recursion

$$
\begin{aligned}
x_0 &= \epsilon \\
y_l &= 1 - \rho(1 - x_{l-1}) \\
x_l &= \epsilon\lambda(y_l)
\end{aligned}
$$

Density evolution



Recall

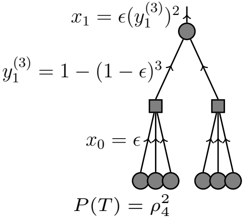
- $\rho(x) = \sum_i \rho_i x^{i-1}$
- $\sum_i \rho_i = 1$
- $\lambda(x) = \sum_i \lambda_i x^{i-1}$
- $\sum_i \lambda_i = 1$

Recursion

$$
\begin{aligned}
x_0 &= \epsilon \\
y_l &= 1 - \rho(1 - x_{l-1}) \\
x_l &= \epsilon\lambda(y_l) \\
x_l &= \epsilon\lambda(1 - \rho(1 - x_{l-1}))
\end{aligned}
$$

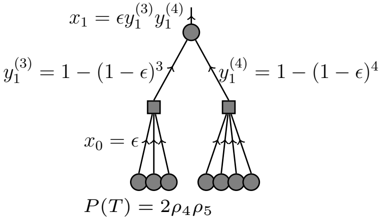
Analysis of the message passing decoder

$\lambda(x) = x^2, \rho(x) = \rho_4 x^3 + \rho_5 x^4$



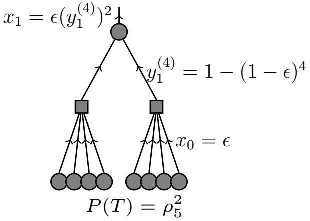
Analysis of the message passing decoder

$\lambda(x) = x^2, \rho(x) = \rho_4 x^3 + \rho_5 x^4$



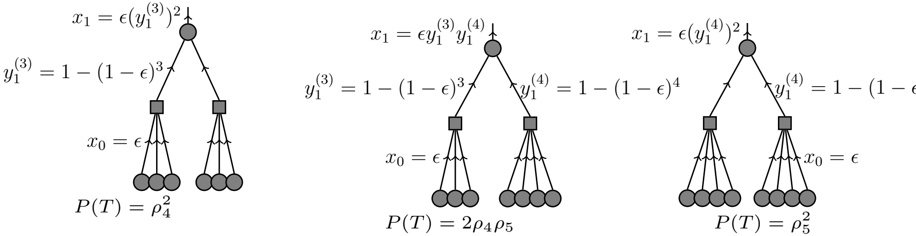
Analysis of the message passing decoder

$\lambda(x) = x^2, \rho(x) = \rho_4 x^3 + \rho_5 x^4$



Analysis of the message passing decoder

$\lambda(x) = x^2, \rho(x) = \rho_4 x^3 + \rho_5 x^4$



$$
\begin{aligned}
\mathbb{E}_{\mathsf{LDPC}(\lambda,\rho)}[x_1] &= \sum_{T \in \mathcal{T}_1(\lambda,\rho)} P(T) * x_1(T,\epsilon) \\
&= \epsilon(\rho_4 y_1^{(3)} + \rho_5 y_1^{(4)})^2 \\
&= \epsilon(1 - \rho_4(1-\epsilon)^3 - \rho_5(1-\epsilon)^4)^2 \\
&= \epsilon\lambda\left(1 - \rho(1-\epsilon)\right)
\end{aligned}
$$

Threshold

Convergence condition

$$x_l = \epsilon\lambda(1 - \rho(1 - x_{l-1})) = f(\epsilon, x_{l-1})$$

$$\begin{array}{rl} x_l \text{ converges to 0 if} & f(\epsilon, x) < x, \; x \in (0, \epsilon] \\ \text{There is a fixed point if} & f(\epsilon, x) = x, \; \text{for some } x \in (0, \epsilon] \end{array}$$

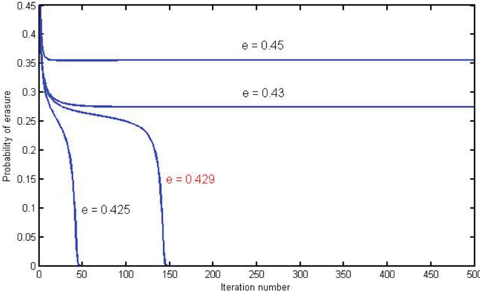
Threshold

Convergence condition

$$x_l = \epsilon\lambda(1-\rho(1-x_{l-1})) = f(\epsilon, x_{l-1})$$

x_l converges to 0 if $\quad f(\epsilon, x) < x,\ \ x \in (0, \epsilon]$

There is a fixed point if $\quad f(\epsilon, x) = x,\ $ for some $x \in (0, \epsilon]$



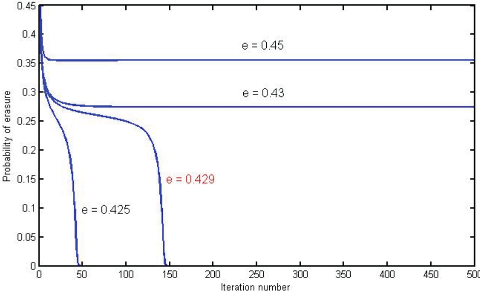
Threshold

Convergence condition

$$x_l = \epsilon\lambda(1-\rho(1-x_{l-1})) = f(\epsilon, x_{l-1})$$

x_l converges to 0 if $\quad f(\epsilon, x) < x,\ \ x \in (0, \epsilon]$

There is a fixed point if $\quad f(\epsilon, x) = x,\ $ for some $x \in (0, \epsilon]$



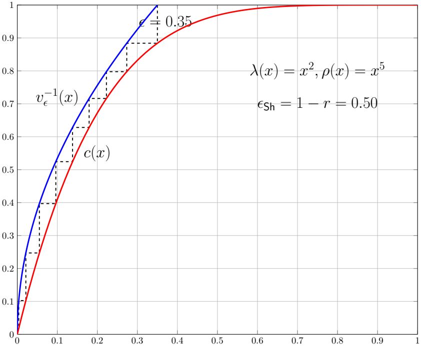
The threshold $\epsilon^{\text{BP}}(\lambda, \rho)$ is defined as

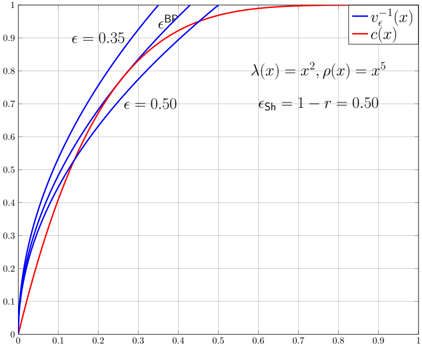
$$\epsilon^{\text{BP}}(\lambda, \rho) = \sup\{\epsilon \in [0,1] : x_l \to 0 \text{ as } l \to \infty\}$$

Exit charts - Ashikmin, Kramer, ten Brink'04

Node functions

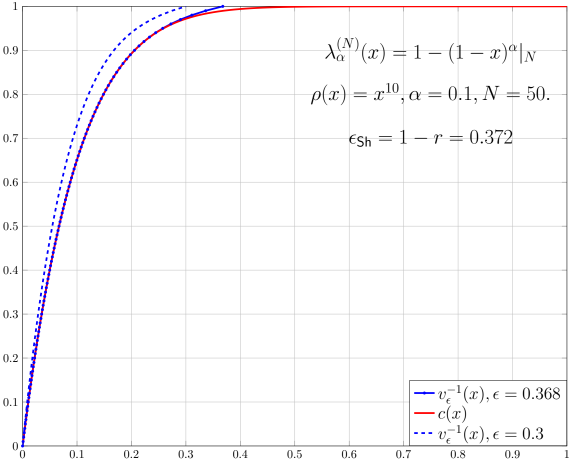
- Var node function: $v_\epsilon(x) = \epsilon\lambda(x)$
- Check node function: $c(x) = 1 - \rho(1 - x)$





Optimality of EXIT chart matching

- Var node function: $v_\epsilon(x) = \epsilon\lambda(x)$
- Check node function: $c(x) = 1 - \rho(1 - x)$



Summary

- Understand what degree distributions $(\lambda(x), \rho(x))$ mean

Summary

- Understand what degree distributions $(\lambda(x), \rho(x))$ mean
- Given a (λ, ρ) and ϵ, what will be the P_e^n as $l, n \to \infty$?

Summary

- Understand what degree distributions $(\lambda(x), \rho(x))$ mean
- Given a (λ, ρ) and ϵ, what will be the P_e^n as $l, n \to \infty$?
- Can you compute the threshold?

Summary

- Understand what degree distributions $(\lambda(x), \rho(x))$ mean
- Given a (λ, ρ) and ϵ, what will be the P_e^n as $l, n \to \infty$?
- Can you compute the threshold?
- Is a $(\lambda(x), \rho(x))$ pair optimal?

Back to theory: from erasures to errors

Finite field with p elements

p is prime

- $\mathbb{F}_p - \{0, 1, 2, \ldots, p-1\}$
- $a \oplus b = (a+b) \mod p$
- $a \odot b = (ab) \mod p$
- We can $+, \times, \div$, inverses
- W is a (primitive) element such that $1, W, W^2, \ldots, W^{p-1}$ are distinct

Finite field with p elements

p is prime

- $\mathbb{F}_p - \{0, 1, 2, \ldots, p-1\}$
- $a \oplus b = (a+b) \mod p$
- $a \odot b = (ab) \mod p$
- We can $+, \times, \div$, inverses
- W is a (primitive) element such that $1, W, W^2, \ldots, W^{p-1}$ are distinct

Example $\mathbb{F}_5$

- $W = 2$
- $W^0 = 1, W^1 = 2, W^2 = 4, W^3 = 3$

Finite field with p elements

p is prime

- $\mathbb{F}_p - \{0, 1, 2, \dots, p-1\}$
- $a \oplus b = (a+b) \mod p$
- $a \odot b = (ab) \mod p$
- We can $+, \times, \div$, inverses
- W is a (primitive) element such that $1, W, W^2, \dots, W^{p-1}$ are distinct

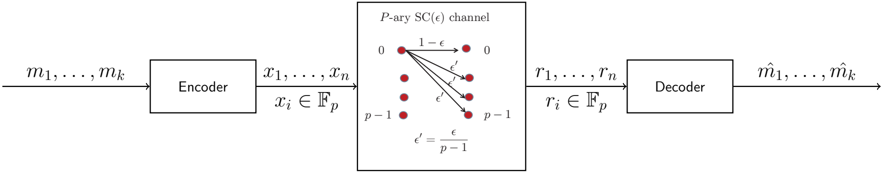
Example $\mathbb{F}_5$

- $W = 2$
- $W^0 = 1, W^1 = 2, W^2 = 4, W^3 = 3$

p need not be prime

- Everything can be extended to finite fields with $q = 2^r$ elements
- May be extended to integers - not sure

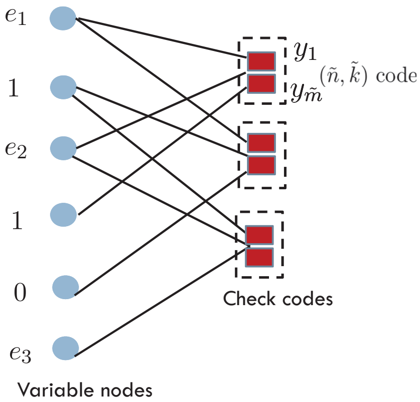
p-symmetric channel and error correction



Error correction coding

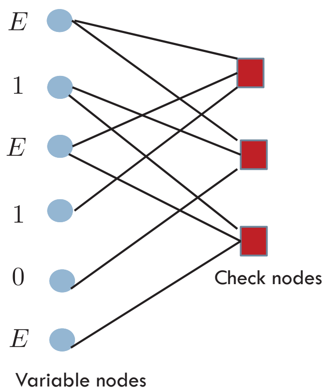
- Another simple channel model which has been extensively considered
- Has been the canonical model for algebraic coding theorists

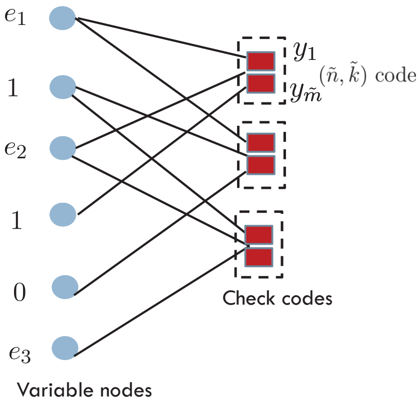
Generalized LDPC code and error channels



- GLDPC introduced by Tanner in 1981
- Each check is a $(\tilde{n}, \tilde{k})$, t-error correcting code
- If there are $\leq t$ errors in a check, it can be recovered
- For now, assume no miscorrections

Peeling process is same for erasure and error channels





- Assume 1-error correcting check code and no miscorrections
- One-to-one correspondence between messages passed - DE can be used
- Not optimal for the error channel but it is not bad at high rates
- Spatially coupled versions are optimal at high rates (Jian, Pfister and N)

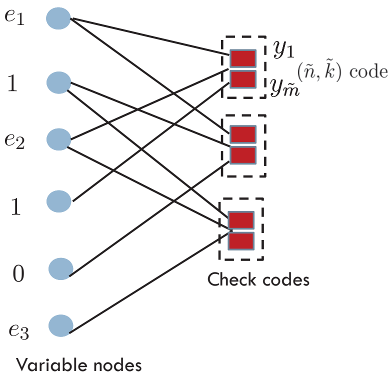
Erasures to errors - tensoring and peeling

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

$$\otimes$$

$$B = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & W & W^2 & W^3 & W^4 & W^5 \end{bmatrix}$$

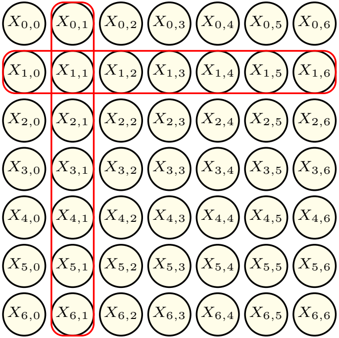
$$\tilde{\mathbf{H}} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & W^2 & W^3 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & W & 0 & 0 & W^4 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & W & W^2 & 0 & 0 & W^5 \end{bmatrix}$$



- W is a primitive element in the field
- Each check is a 1-error correcting code
- If there is exactly one error in a check, it can be recovered

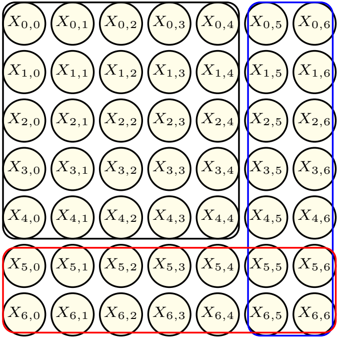
Product code

- Special case of generalized LDPC code
- Let component code $\mathcal{C}$ be an $(\tilde{n}, \tilde{k}, \tilde{d}_{\text{min}})$ linear code
- Well-known that $\mathcal{P}$ is an $(\tilde{n}^2, \tilde{k}^2, \tilde{d}^2_{\text{min}})$ linear code



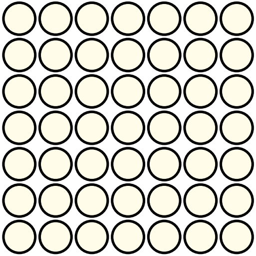
Product code

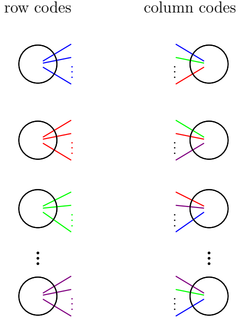
- Special case of generalized LDPC code
- Let component code $\mathcal{C}$ be an $(\tilde{n}, \tilde{k}, \tilde{d}_{\mathsf{min}})$ linear code
- Well-known that $\mathcal{P}$ is an $(\tilde{n}^2, \tilde{k}^2, \tilde{d}^2_{\mathsf{min}})$ linear code



Peeling decoding of product codes

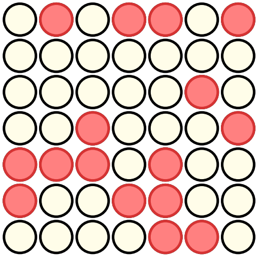
- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding



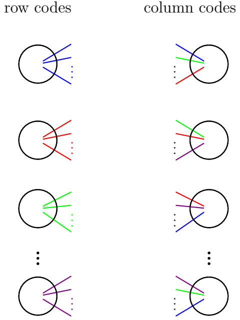


Peeling decoding of product codes

- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding

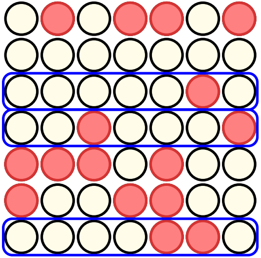


Received block

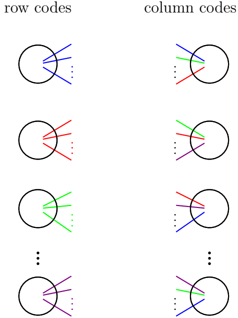


Peeling decoding of product codes

- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding

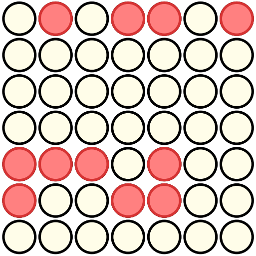


Row decoding

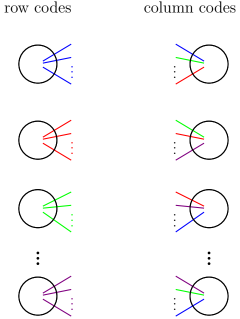


Peeling decoding of product codes

- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding

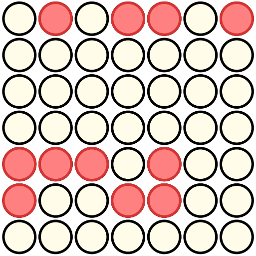


Row decoding

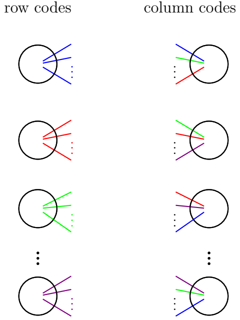


Peeling decoding of product codes

- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding

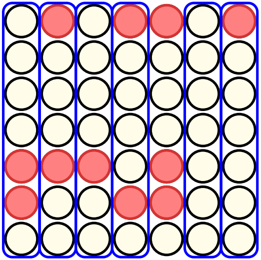


Column decoding

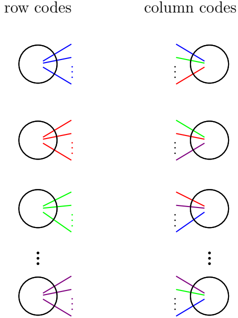


Peeling decoding of product codes

- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding

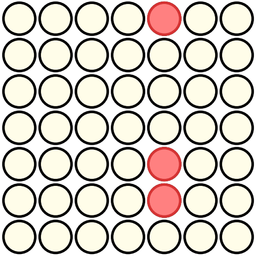


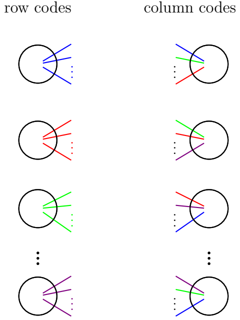
Column decoding



Peeling decoding of product codes

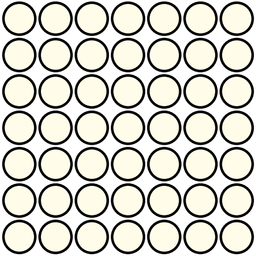
- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding



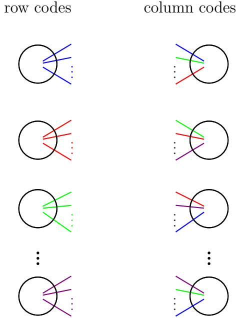


Peeling decoding of product codes

- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding

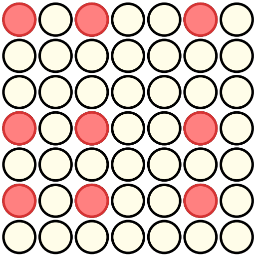


Decoding successful

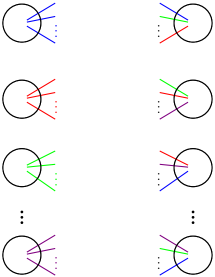


Peeling decoding of product codes

- Hard-decision "cascade decoding" by Abramson in 1968
- Identical to a peeling decoder
- Example: $t = 2$-error-correcting codes, bounded distance decoding



Or trapped in a stopping set



Density Evolution(DE) for Product Codes -Justesen et al

What is different about DE?

- Graph is highly structured
- Neighborhood is not tree-like
- Remarkably, randomness in the errors suffices!

Density Evolution(DE) for Product Codes -Justesen et al

What is different about DE?

- Graph is highly structured
- Neighborhood is not tree-like
- Remarkably, randomness in the errors suffices!

Assumptions

- Errors are randomly distributed in rows and columns
- # errors in each row/col $\sim$ Poisson(M))

Density Evolution(DE) for Product Codes -Justesen et al

What is different about DE?

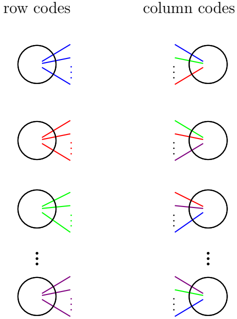
- Graph is highly structured
- Neighborhood is not tree-like
- Remarkably, randomness in the errors suffices!

Assumptions

- Errors are randomly distributed in rows and columns
- # errors in each row/col $\sim$ Poisson(M))

Main Idea

- Removal of corrected vertices (degree$\leq t$) from row codes $\Leftrightarrow$ removal of random edges from column codes uniformly at random
- # of errors in row/column changes after each iter
- Track the distribution



DE continued

Tail of the Poisson distribution

$$\pi_t(m) = \sum_{j \geq t} \mathrm{e}^{-m} m^j / j!$$

Effect of first step of decoding

If the # errors is Poisson with mean M, Mean # of errors after decoding is

$$m(1) = \sum_{j \geq t+1} j\mathrm{e}^{-M} M^j / j! = M \pi_t(M)$$

Evolution of degree distribution($d = 2$) - first iteration

Row decoding

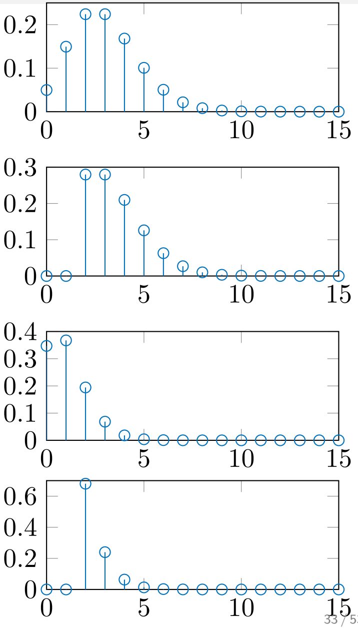
- Before row decoding
  - Distribution: Poisson(M), Mean: M
- After row decoding
  - Distribution: Truncated Poisson(M)
  - Mean: $M\pi_t(M) = m(1)$

Column decoding

- Before column decoding
  - Distribution: Poisson($m(1)$),Mean: $m(1)$
- After column decoding
  - Distribution: Truncated Poisson($m(1)$)
  - Mean: $m(2) = M\pi_t(m(1))$

After every decoding

- Distribution is a Truncated Poisson($m(j)$)
- $P[\#errors = i] = b\frac{m(j)^i}{i!}$



Evolution of the degree distribution - jth iteration

Recursion

- $m(0) = M$
- $m(1) = M\pi_t(M)$
- $m(j) = M\pi_t(m(j-1))$

Reduction in the parameter

- Average no. of errors in each row (column) $= m(j)\pi_t(m(j))$
- Decoding of rows reduces the parameter by $\frac{m(j)\pi_t(m(j))}{m(j-1)\pi_t(m(j-1))} = \frac{M\pi(m(j))}{m(j-1)}$
- New parameter is $m(j+1) = M\pi(m(j))$

Threshold

In the limit of large $\tilde{n}$ (length in each dimension), a t-error correcting product code can correct $\tilde{n}M$ errors when

$$M < \min_m \{\frac{m}{\pi_t(m)}\}$$

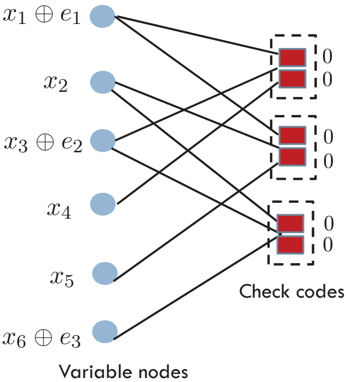
Thresholds for asymptotically large field size

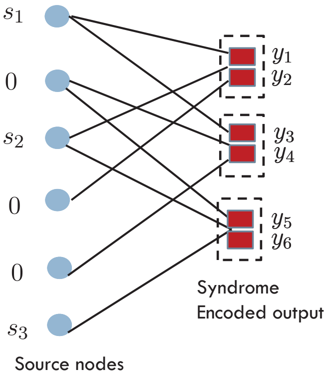
Threshold $= \frac{\#\ of parity symbols}{\#\ of errors}$

	$d=2$	$d=3$	$d=4$	$d=5$	$d=6$	$d=7$	$d=8$
$t=1$	4.0	2.4436	2.5897	2.8499	3.1393	3.4378	3.7383
$t=2$	2.3874	2.5759	2.9993	3.4549	3.9153	4.3736	4.8278
$t=3$	2.3304	2.7593	3.3133	3.8817	4.4483	5.0094	5.5641
$t=4$	2.3532	2.9125	3.5556	4.2043	4.8468	5.4802	6.1033

Notice that $L, K = O\left(N^{\frac{1-d}{d}}\right)$

Syndrome source coding





- $H\underline{x} = 0$
- Receive - $\underline{r} = \underline{x} \oplus \underline{e}$
- $H\underline{r} = H\underline{e} = \underline{y}$
- Recover $\underline{x}$ and sparse $\underline{e}$

- $H\underline{s} = \underline{y}$
- Set $\underline{r} = 0$ (Let a genie add $\underline{x}$ to $\underline{r}$)
- $\underline{y}$ is given to the decoder
- Recover sparse $\underline{s}$

Application 2

Sparse Fast Fourier Transform (SFFT) Computation

Problem Statement

$x[n]$: Time domain signal of length N whose spectrum is K-sparse

$$x[n] \xrightarrow{\text{DFT}} \underbrace{X[k]}_{(K-\text{sparse})}$$

Compute the locations and values of the K non-zero coefficients w.h.p

Sparse Fast Fourier Transform (SFFT) Computation

Problem Statement

$x[n]$: Time domain signal of length N whose spectrum is K-sparse

$$x[n] \xrightarrow{\text{DFT}} \underset{(K-\text{sparse})}{X[k]}$$

Compute the locations and values of the K non-zero coefficients w.h.p

Fast Fourier Transform (FFT)

- Sample complexity: N samples
- Computational complexity: $O(N \log N)$

We want sublinear sample and computational complexity

Sparse Fast Fourier Transform (SFFT) Computation

Problem Statement

$x[n]$: Time domain signal of length N whose spectrum is K-sparse

$$x[n] \xrightarrow{\text{DFT}} \underset{(K-\text{sparse})}{X[k]}$$

Compute the locations and values of the K non-zero coefficients w.h.p

Related work

- Spectral estimation - Prony's method
- More recently Pawar and Ramchandran'13, Hassanieh, Indyk, Katabi'12

SFFT - A Sparse Graph Based Approach

Main Idea - Pawar and Ramchandran 2013

- Sub-sampling in time corresponds to aliasing in frequency
- Aliased coefficients $\Leftrightarrow$ parity check constraints of GLDPC codes
- CRT guided sub-sampling induces a code good for Peeling decoder
- Problem is identical to syndrome source coding

SFFT - A Sparse Graph Based Approach

Main Idea - Pawar and Ramchandran 2013

- Sub-sampling in time corresponds to aliasing in frequency
- Aliased coefficients $\Leftrightarrow$ parity check constraints of GLDPC codes
- CRT guided sub-sampling induces a code good for Peeling decoder
- Problem is identical to syndrome source coding

FFAST for Computing the DFT - Pawar and Ramchandran 2013

- Sampling complexity: $M = O(K)$ time domain samples
- Computational complexity: $O(K \log K)$

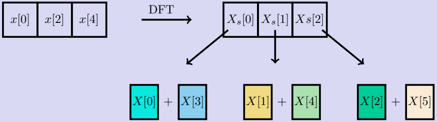
Subsampling and Aliasing - A Quick Review

Subsampling results in aliasing

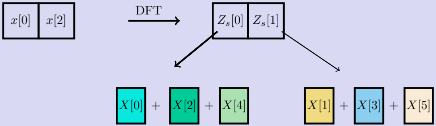
- Let $x[n] \xrightarrow{N-DFT} X[k], \quad k, n = 0, 1, \ldots, N-1$
- Let $x_s[n] = x[mL], \quad m = 0, 1, \ldots, N/L = M$ be a sub-sampled signal
- Let $x_s[m] \xrightarrow{M-DFT} X_s[l]$ be the DFT of the sub-sampled signal
- $\boxed{X_s[l] = M \sum_{p=0}^{L-1} X[l + pM]}$

Aliasing and Sparse Graph Codes

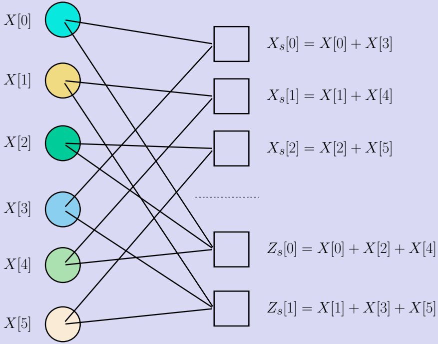
x_s: Sub-sampled by $f_1 = P_1 = 2$



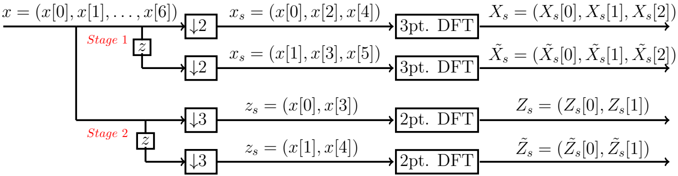
z_s: Sub-sampled by $f_2 = P_2 = 3$



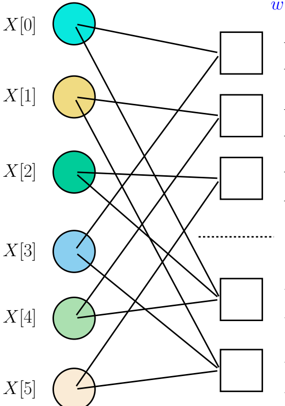
Factor graph



FFAST Algorithm Example



$w = e^{-j\frac{2\pi}{6}}$



$X_s[0] = X[0] + X[3]$
$\tilde{X}_s[0] = X[0]w^0 + X[3]w^3$

$X_s[1] = X[1] + X[4]$
$\tilde{X}_s[1] = X[1]w^1 + X[4]w^4$

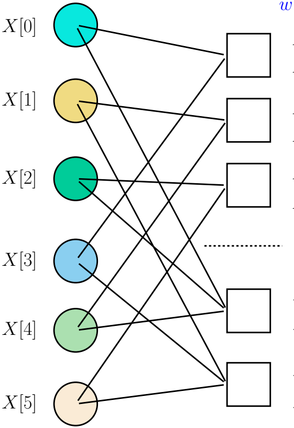
$X_s[2] = X[2] + X[5]$
$\tilde{X}_s[2] = X[2]w^2 + X[5]w^5$

$Z_s[0] = X[0] + X[2] + X[4]$
$\tilde{Z}_s[0] = X[0]w^0 + X[2]w^2 + X[4]w^4$

$Z_s[1] = X[1] + X[3] + X[5]$
$\tilde{Z}_s[1] = X[1]w^1 + X[3]w^3 + X[5]w^5$

Singleton Detection

$w = e^{-j\frac{2\pi}{6}}$



$X_s[0] = X[0] + X[3]$
$\tilde{X}_s[0] = X[0]w^0 + X[3]w^3$

$X_s[1] = X[1] + X[4]$
$\tilde{X}_s[1] = X[1]w^1 + X[4]w^4$

$X_s[2] = X[2] + X[5]$
$\tilde{X}_s[2] = X[2]w^2 + X[5]w^5$

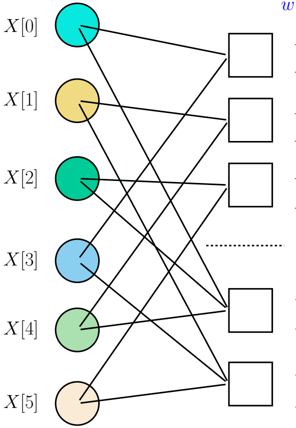
$Z_s[0] = X[0] + X[2] + X[4]$
$\tilde{Z}_s[0] = X[0]w^0 + X[2]w^2 + X[4]w^4$

$Z_s[1] = X[1] + X[3] + X[5]$
$\tilde{Z}_s[1] = X[1]w^1 + X[3]w^3 + X[5]w^5$

Singleton condition for a checknode

- Let $i = \frac{-N}{j2\pi}\log(\frac{\tilde{X}_s[l]}{X_s[l]})$. If $0 \leq i \leq N-1$, then checknode l is a Singleton.
- $Pos(l) = i$ is the only variable node participating and $X_s[l]$ is its value.

FFAST Decoder



$w = e^{-j\frac{2\pi}{6}}$

$X_s[0] = X[0] + X[3]$
$\tilde{X}_s[0] = X[0]w^0 + X[3]w^3$

$X_s[1] = X[1] + X[4]$
$\tilde{X}_s[1] = X[1]w^1 + X[4]w^4$

$X_s[2] = X[2] + X[5]$
$\tilde{X}_s[2] = X[2]w^2 + X[5]w^5$

$Z_s[0] = X[0] + X[2] + X[4]$
$\tilde{Z}_s[0] = X[0]w^0 + X[2]w^2 + X[4]w^4$

$Z_s[1] = X[1] + X[3] + X[5]$
$\tilde{Z}_s[1] = X[1]w^1 + X[3]w^3 + X[5]w^5$

Peeling decoder

- 1 non-zero value among the neighbors of any right node can be recovered
- Iteratively errors can be corrected and analyzed for random non-zero coeffs

FFAST Decoder Example

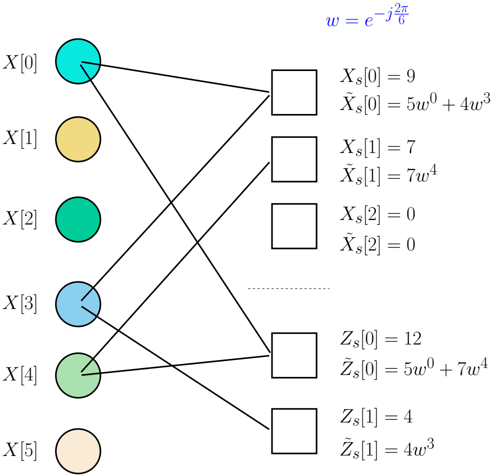
Example 1

Let $N = 6$, and the non-zero coefficients be X[0]=5, X[3]=4, X[4]=7

FFAST Decoder Example

Example 1

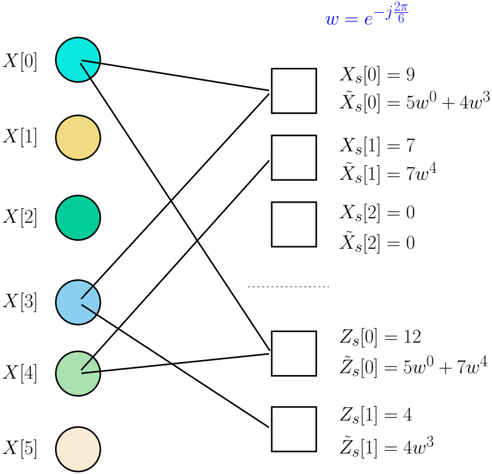
Let $N = 6$, and the non-zero coefficients be X[0]=5, X[3]=4, X[4]=7



FFAST Decoder Example

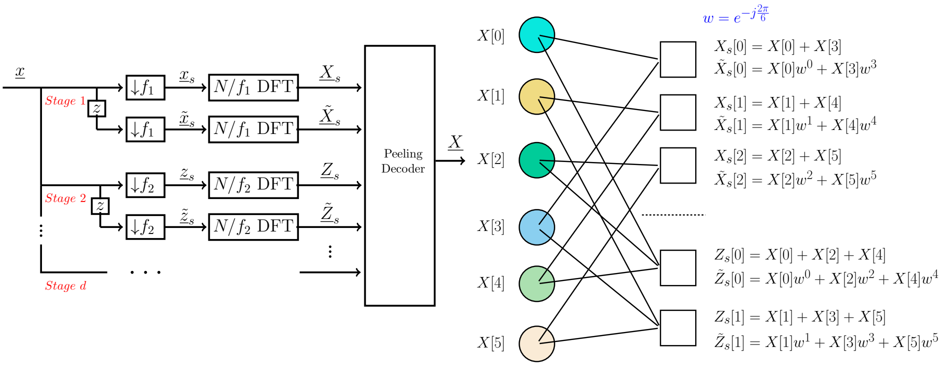
Example 1

Let $N = 6$, and the non-zero coefficients be X[0]=5, X[3]=4, X[4]=7



Yes, recoverable!

Generalization



Reed Solomon component codes

- $(X_s[l_1], \tilde{X}_s[l_1])$ correspond to 2 syndromes of a 1-error correcting RS code
- RS is over the complex field, no miscorrection

Product codes and FFAST ($d = 2$)

- X: K-sparse spectrum of length $N = P_1P_2$ (P_1 and P_2 are co-prime)
- X': $P_1 \times P_2$ matrix formed by rearranging X according to mapping $\mathcal{M}$

$$
\begin{aligned}
X_s[l_1] &= \sum_{i=0}^{P_2-1} X[l_1 + iP_1], \quad 0 \le l_1 \le P_2 - 1 \\
Z_s[l_2] &= \sum_{i=0}^{P_1-1} X[l_2 + iP_2], \quad 0 \le l_2 \le P_1 - 1
\end{aligned}
$$

Mapping

The mapping from $X(r)$ to $X'(i,j)$ is given by

$$(i,j) = \mathcal{M}(r) \equiv (r \mod P_2,\ r \mod P_1).$$

Note: CRT ensures that $\mathcal{M}$ is bijective

$$
\begin{array}{c|c|c|c|c|}
 & Z_s[0] & Z_s[1] & Z_s[2] & Z_s[3] \\
 & \downarrow & \downarrow & \downarrow & \downarrow \\
\hline
X_s[0] \longrightarrow & X[0] & X[5] & X[10] & X[15] \\
\hline
X_s[1] \longrightarrow & X[16] & X[1] & X[6] & X[11] \\
\hline
X_s[2] \longrightarrow & X[12] & X[17] & X[2] & X[7] \\
\hline
X_s[3] \longrightarrow & X[8] & X[13] & X[18] & X[3] \\
\hline
X_s[4] \longrightarrow & X[4] & X[9] & X[14] & X[19] \\
\hline
\end{array}
$$

Product codes and FFAST ($d \geq 3$)

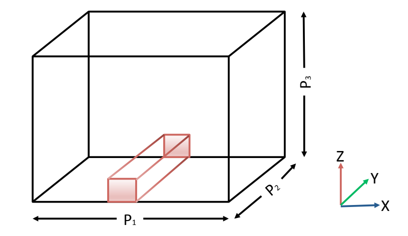
$N = P_1 \times P_2 \times \ldots \times P_d$

$$(i_1, i_2, \ldots, i_d) = \mathcal{M}(r) \equiv (r \mod f_1, r \mod f_2, \ldots, r \mod f_d).$$

Less-sparse regime

$f_i = N/P_i, \; i = 1, 2, \ldots, d$

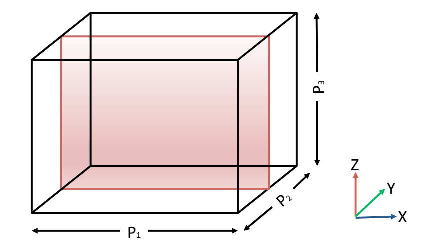
$\underline{d = 3}$



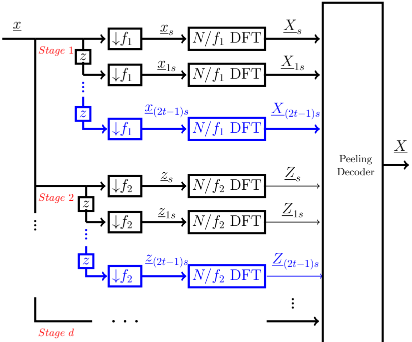
Very-sparse regime

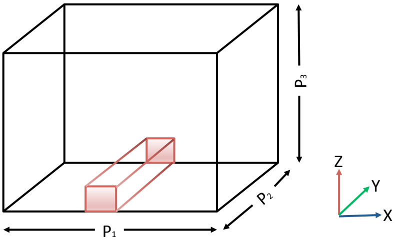
$f_i = P_i, \; i = 1, 2, \ldots, d$

$\underline{d = 3}$



Connections between FFAST and Product Codes





FFAST	$\Leftrightarrow$	Product codes
d stages	$\Leftrightarrow$	d-dimensional product code
$2t$ branches	$\Leftrightarrow$	t-error correcting RS component codes
Non-zero coefficients	$\Leftrightarrow$	Error locations
Recovery of coefficients	$\Leftrightarrow$	Iterative decoding

Thresholds

Theorem 1

Less sparse case: In the limit of large P, the FFAST algorithm with d branches and $2t$ stages can recover the FFT coefficients w.h.p if $K < \frac{2dt}{c_{d,t}}$.

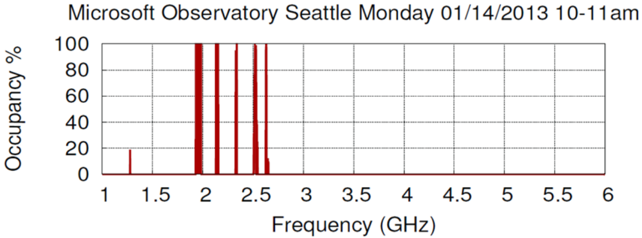
$$c_{d,t} = \min_m \{m/\pi^{d-1}(m)\}$$

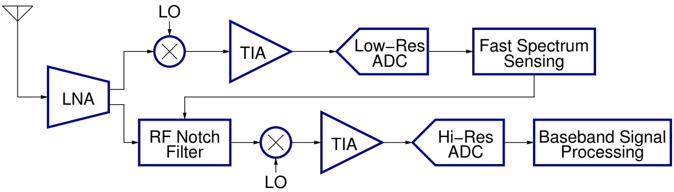
$$\text{Threshold} = \frac{\#\ of\ measurements}{recoverable\ sparsity} = \frac{2dt}{c_{d,t}}$$

	$d=2$	$d=3$	$d=4$	$d=5$	$d=6$	$d=7$	$d=8$
$t=1$	4.0	2.4436	2.5897	2.8499	3.1393	3.4378	3.7383
$t=2$	2.3874	2.5759	2.9993	3.4549	3.9153	4.3736	4.8278
$t=3$	2.3304	2.7593	3.3133	3.8817	4.4483	5.0094	5.5641
$t=4$	2.3532	2.9125	3.5556	4.2043	4.8468	5.4802	6.1033

Notice that $L, K = O\left(N^{\frac{1-d}{d}}\right)$

Interference-tolerant A/D Converter





Open problems

- If we use MAP decoding, is the subsampling procedure optimal?
- What happens when $N = 2^i$?
- Bursty case? Can we have threshold theorems?
- Using this idea in actual applications

Questions?

Thank you!