Iterative Decoding and LDPC Codes

Jossy Sayir

Based on courses prepared in collaboration with

Ingmar Land & Gottfried Lechner

Cours d’été à Peyresq, Juillet 2007

Kplus
Kompetenzzentren-Programm

Overview

- Linear Codes
- Coding theorem for the Binary Erasure Channel (BEC)

- Iterative Processing
- Low-Density Parity-Check Codes

- Density Evolution for the BEC
- Density Evolution for other channels

- EXIT Charts
- EXIT Charts for the BEC, area property

- Information Combining

Linear Codes

- Let $\mathbf{G}$ be an KxN matrix of rank K over $GF(q^m)$
- Linear code

$$\mathcal{C} = \{x \in GF(q^m)^N : \exists u \in GF(q^m)^K \mid x = u\mathbf{G}\}$$

- $\mathbf{G}$ is called a generator or encoding matrix of $\mathcal{C}$
- $\mathcal{C}$ is a linear subspace of $GF(q^m)^N$ of dimension K
- Code rate: $R = m \log q \, K/N$

- Pick any Nx$(N-K)$ matrix $\mathbf{H}$ of rank $N-K$ such that $\mathbf{GH}^T = 0$
- Then $\mathcal{C} = \{x \in GF(q^m)^N : x\mathbf{H}^T = 0\}$
- $\mathbf{H}$ is called a parity-check matrix of $\mathcal{C}$

Example over GF(2)

$$(x_1\ x_2 \ldots\ x_{16}) = (u_1 \ldots\ u_6)\begin{pmatrix}
0&1&1&0&0&0&0&0&1&0&1&1&1&0&0&0\\
0&0&0&1&1&0&1&0&0&0&0&1&0&1&1&0\\
1&1&0&0&0&0&0&1&0&1&0&0&0&0&1&1\\
0&0&0&0&1&1&1&0&1&1&1&0&1&1&0&0\\
0&0&0&0&0&1&1&1&0&0&0&0&1&0&0&0\\
0&0&1&1&0&0&0&1&0&0&1&1&0&1&0&0
\end{pmatrix}$$

$N = 16$

$K = 6$

Received: (1 X X X X 1 0 0 0 1 X 1 X X X 1)

Remove columns corresponding to erasures…

$$(x_1\ x_6 x_7\ x_8\ x_9\ x_{10}\ x_{12}\ x_{16}) = (u_1 \ldots\ u_6)\begin{pmatrix}
0&0&0&0&1&0&1&0\\
0&0&1&0&0&0&1&0\\
1&0&0&1&0&1&0&1\\
0&1&1&0&1&1&0&0\\
0&1&1&1&0&0&0&0\\
0&0&0&1&0&0&1&0
\end{pmatrix}$$

Example (continued)

Drop last 2 columns: $(x_1\ x_6 x_7\ x_8\ x_9\ x_{10}) = (u_1 \dots u_6)$ $\begin{pmatrix} 0&0&0&0&1&0\\ 0&0&1&0&0&0\\ 1&0&0&1&0&1\\ 0&1&1&0&1&1\\ 0&1&1&1&0&0\\ 0&0&0&1&0&0 \end{pmatrix}$

(system of equations is overdetermined)

Triangulize problem:

$$(x_1\ x_6 x_7\ x_8\ x_9\ x_{10}) = (u_1' \dots u_6') \begin{array}{c} \begin{pmatrix} 1&0&0&1&0&1\\ 0&1&1&0&1&1\\ 0&0&1&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&1&0\\ 0&0&0&0&0&1 \end{pmatrix} \end{array} \begin{array}{l} g_3\\ g_4\\ g_2\\ g_6\\ g_1\\ g_1+g_4+g_5+g_6 \end{array}$$

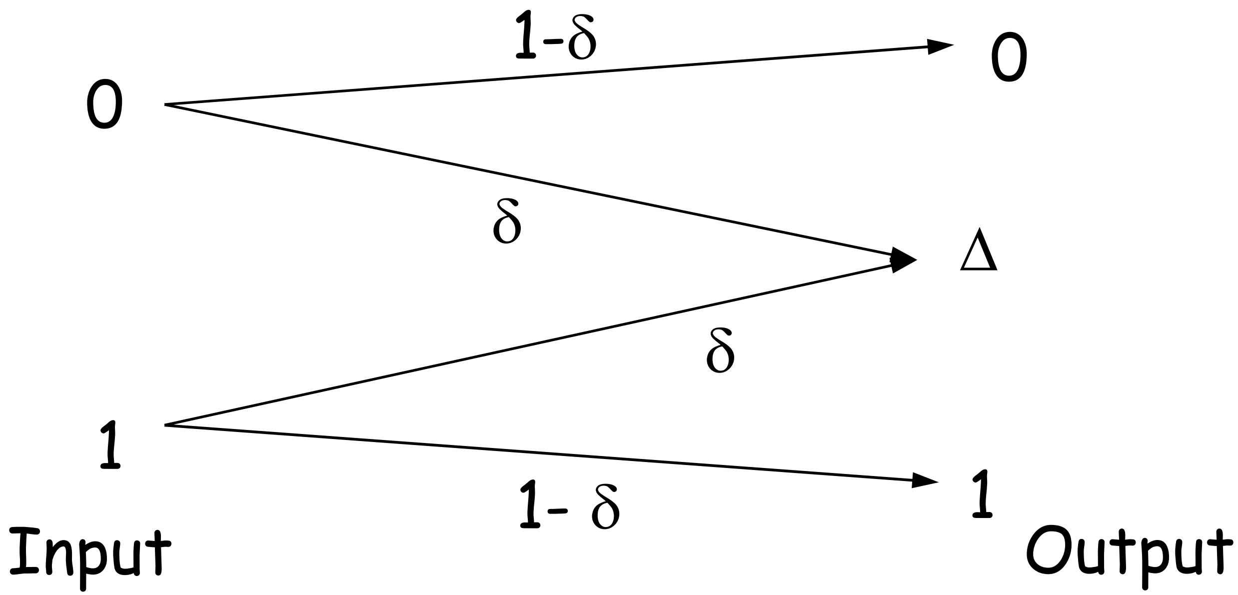
Solution: $(x_1\ x_6 x_7\ x_8\ x_9\ x_{10}) = (1\ 1\ 0\ 0\ 0\ 1)$

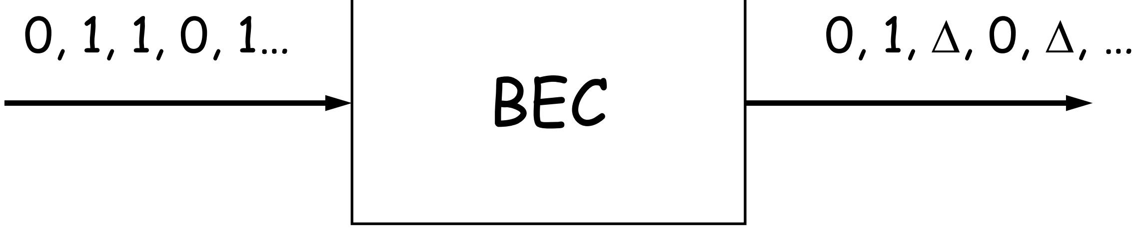
$$= g_1' + g_2' + g_3' + g_4' + g_5' + g_6'$$

$$= g_2 + g_3 + g_5$$

$$(u_1\ u_2\ u_3\ u_4\ u_6) = (0\ 1\ 1\ 0\ 1\ 0)$$

The Binary Erasure Channel





The Big Question

We have learned how to decode a codeword that has been transmitted over a Binary Erasure Channel (by solving a system of equations, whenever possible)

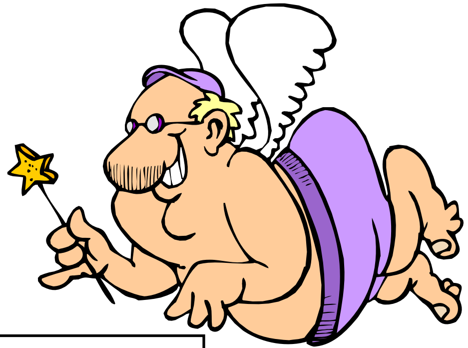
What is the probability of decoding successfully, in function of the erasure probability δ, the code rate R and the length N?

We will suppose that the code V has been selected at random by choosing an encoding matrix G at random among all binary $K \times N$ matrices ($K = NR$)

Converse Coding Theorem

If the information rate R is higher than $1-\delta$, the transmission cannot be arbitrarily reliable

Proof:

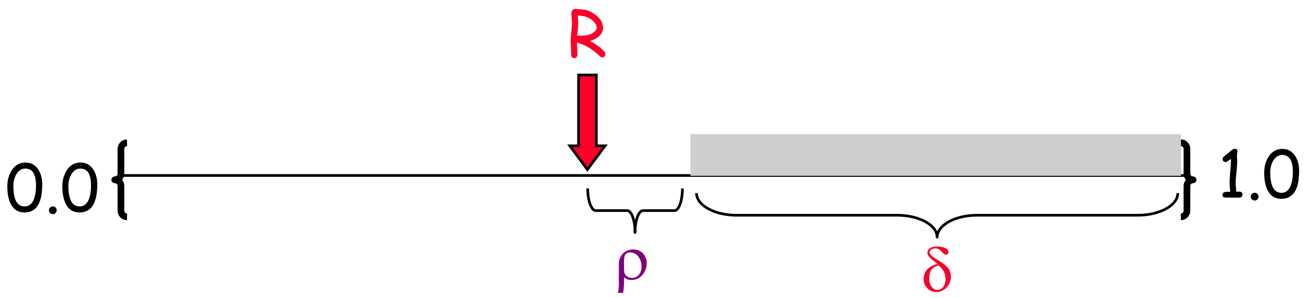


Transmitter Encoder $\xrightarrow{0,\,1,\,1,\,0,\,1\ldots}$ BEC $\xrightarrow{0,\,1,\,\Delta,\,0,\,\Delta,\,\ldots}$

Even if a genie tells the transmitter where the erasures will be, the best strategy is to place the data uncoded in all positions that won't be erased. We can only transmit $1-\delta$ bits per use on average this way.

Coding Theorem

- Let us assume $R = K/N = 1-\delta-\rho$.



- On average, there will be $\delta N = N-K-\rho N$ erasures per block. For now, let us assume that there will be exactly δN erasures every time.
- Our matrix inversion decoder will work if the remaining $K+\rho N$ columns of the encoder matrix are linearly independent, i.e., if the resulting Kx$(K+\rho N)$ matrix has rank K.

$R = K/N = 1-\delta-\rho$

N

010110101011001110101101101010111010101

101011001110101101101010101101010110011

011001110101101101010011001110101101100

111010110110101010110101011100110011101

110101010110101001010011001110100011001

011011010101011010101101010011001110110

100111010110110101001100101100111010001

101101101010011001110101101101010010010

011010101011010101111100111010010101101

101101010010101101010110101101010111001

K

$(1-\delta)N = K + \rho N$

δN

Coding Theorem (cont…)

- The erasures will be placed at random positions in the codeword (not all at the end as in our illustration)
- There are $\binom{N}{\delta N}$ erasure patterns
- Evaluating the probability of success for a given matrix means checking if each of these matrices is invertible…

Random Coding Experiment

- Pick a KxN matrix at random
- Generate an information sequence at random
- Generate the corresponding codeword
- Generate an erasure pattern at random
- What is the probability of success of the matrix inversion decoder?

Rank of a random matrix

- What is the probability that a LxM matrix ($L \leq M$) chosen at random has rank L?

- 1xM: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 2^M-1 choices

- 2xM: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 2^M-1 choices

  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 2^M-2 choices

- 3xM: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 2^M-1 choices

  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 2^M-2 choices

  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 2^M-4 choices

The number of LxM linearly independent matrices is thus:

$$\boxed{\prod_{i=0}^{L-1}(2^M - 2^i)}$$

If our LxM matrix is chosen at random among the $2^{L\times M}$ binary matrices, then the probability that it have rank L is:

$$\boxed{P(\text{rank } L) = \frac{\prod_{i=0}^{L-1}(2^M - 2^i)}{2^{L\times M}} = \prod_{i=M-L+1}^{M}(1-2^{-i})}$$

If $L = M$, we have:

$$P(\text{full rank}) = \frac{1}{2}\frac{3}{4}\frac{7}{8}\frac{15}{16}\frac{31}{32}\frac{63}{64}\cdots\frac{2^L-1}{2^L}$$

If $L \rightarrow \infty$, P(full rank) → 0.2887880950866…

Therefore, $R=1-\delta$ ($\rho=0$), the probability of successful decoding for $N \rightarrow \infty$ tends towards 0.2887880950866…

If $L < M$, we have:

$$P(\text{full rank}) = \underbrace{\qquad\qquad}_{M - L} \frac{15}{16}\,\frac{31}{32}\,\frac{63}{64}\cdots\frac{2^L-1}{2^L}$$

If $M = L+1$, the product starts with 3/4,
if $M = L+2$, the product starts with 7/8, etc...

In our case, $L = K$ and $M = K+\rho N$

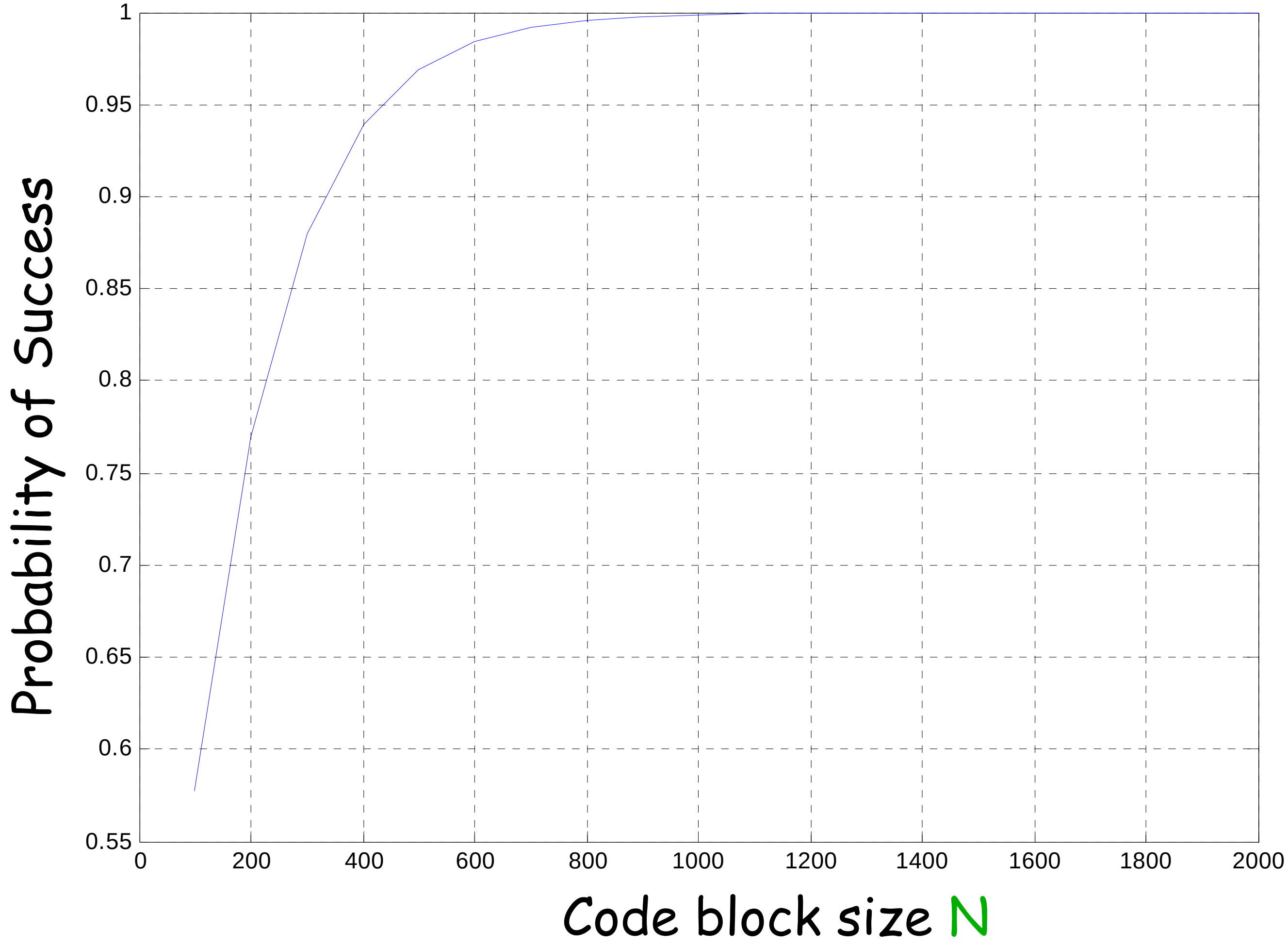
$$\boxed{M - L = \rho N}$$

For a fixed δ and ρ ($R=1-\delta-\rho$), the probability of successful decoding is

$$\boxed{P(\text{success}) = \prod_{i=\rho N+1 \ldots K} (1-2^{-i})}$$

➔ It can be made arbitrarily close to 1 for a given ρ by choosing N large enough

Prob. of successful decoding



ρ = .01

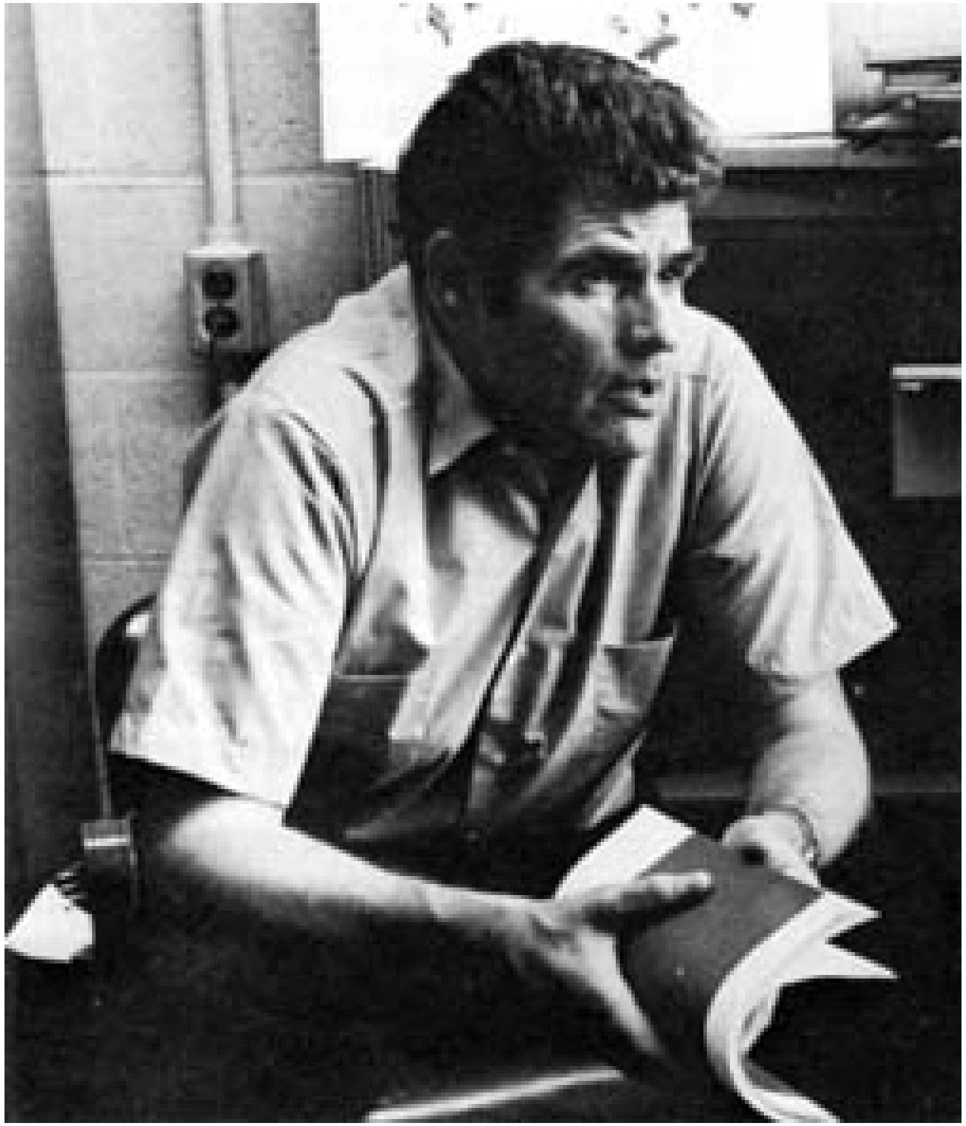
(i.e., R is .01 less than the maximum possible rate)

Information Theory for the Binary Erasure Channel

Peter Elias
1923 - 2001

Professor at MIT,
one of the most original and prolific researchers in information theory,
inventor of convolutional codes

Presented coding theorems for linear codes for the BEC in London, 1955 (without proof…)



What we have learned...

For any information rate $R < 1-\delta$, it is possible to achieve any desired non-zero probability of error provided that we choose K and N large enough.

(Coding theorem)

In plain English:

Arbitrary reliability is possible up to a certain rate, but we have to work with very large block sizes

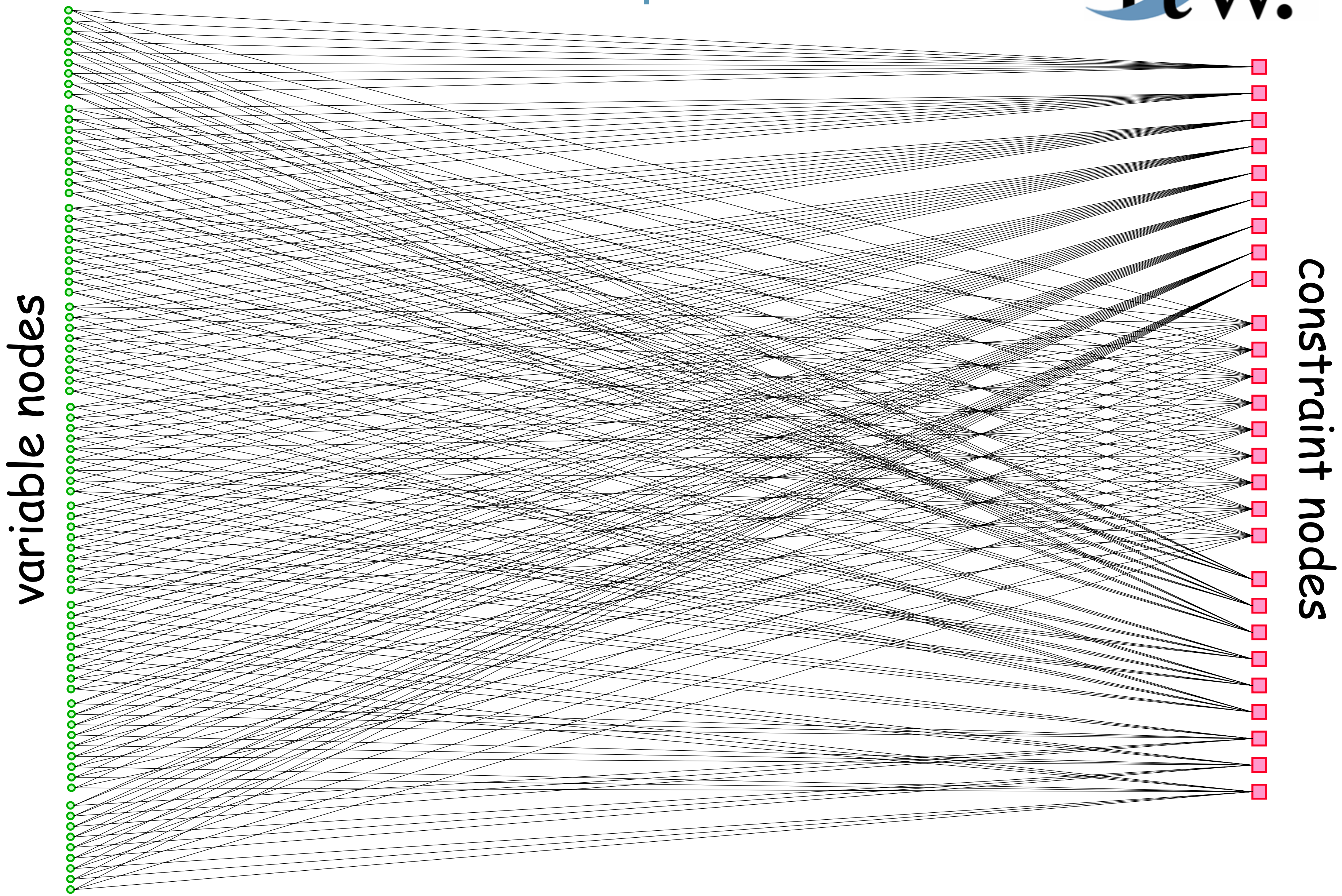
And furthermore:

We can expect to do pretty well by simply choosing our codes at random

SUDOKU

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

SUDOKU Factor Graph



SUDOKU Message Passing

- 81 variables, 27 constraints
- every variable participates in 3 constraints
- every constraint ties 9 variables
- message passing SUDOKU solver
- message = set of possible values
- not every solvable SUDOKU can be solved by mere message passing (only “easy” SUDOKUs)

Iterative Decoding for a Parity-Check Matrix...

Received: (1 X X X X 1 0 0 0 1 X 1 X X X 1)

Parity-Check Matrix:

$$\begin{pmatrix}
0&1&0&0&0&0&0&0&0&0&0&1&1&0&0&1\\
0&0&0&1&0&1&1&0&0&1&0&0&0&0&0&0\\
0&0&1&0&1&0&0&0&1&0&0&0&0&1&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0\\
1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1\\
0&1&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&1&0\\
0&0&1&0&0&0&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&1&1&0&0\\
1&0&0&1&0&0&0&1&0&0&1&0&0&0&0&0
\end{pmatrix}$$

Iterative Decoding

Step 1

$$(1\ X\ X\ \mathbf{0}\ X\ 1\ 0\ 0\ 0\ 1\ X\ 1\ X\ X\ X\ 1)$$

$$\begin{pmatrix}
0&1&0&0&0&0&0&0&0&0&0&1&1&0&0&1\\
0&0&0&1&0&1&1&0&0&1&0&0&0&0&0&0\\
0&0&1&0&1&0&0&0&1&0&0&0&0&1&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0\\
1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1\\
0&1&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&1&0\\
0&0&1&0&0&0&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&1&1&0&0\\
1&0&0&1&0&0&0&1&0&0&1&0&0&0&0&0
\end{pmatrix}$$

Iterative Decoding

Step 2

$$\begin{pmatrix}1 & X & X & 0 & X & 1 & 0 & 0 & 0 & 1 & X & 1 & X & 1 & X & 1\end{pmatrix}$$

$$\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Iterative Decoding

Step 3

$$\begin{pmatrix} 1 & X & X & 0 & X & 1 & 0 & 0 & 0 & 1 & 1 & 1 & X & 1 & X & 1 \end{pmatrix}$$

$$\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Iterative Decoding

Step 4

$$\begin{pmatrix} 1 & X & X & 0 & X & 1 & 0 & 0 & 0 & 1 & 1 & 1 & X & 1 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Iterative Decoding

Step 5

$$
\begin{pmatrix}1 & X & \color{red}{1} & 0 & X & 1 & 0 & 0 & 0 & 1 & 1 & 1 & X & 1 & 0 & 1\end{pmatrix}
$$

$$
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
$$

Iterative Decoding

Step 6

$$\begin{pmatrix}1 & \color{red}{1} & 1 & 0 & X & 1 & 0 & 0 & 0 & 1 & 1 & 1 & X & 1 & 0 & 1\end{pmatrix}$$

$$\begin{pmatrix}
0&1&0&0&0&0&0&0&0&0&0&1&1&0&0&1\\
0&0&0&1&0&1&1&0&0&1&0&0&0&0&0&0\\
0&0&1&0&1&0&0&0&1&0&0&0&0&1&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0\\
1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1\\
0&\boxed{1}&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&1&0\\
0&0&1&0&0&0&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&1&1&0&0\\
1&0&0&1&0&0&0&1&0&0&1&0&0&0&0&0
\end{pmatrix}$$

Iterative Decoding

Step 7

$$(1\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ X\ 1\ 0\ 1)$$

$$\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

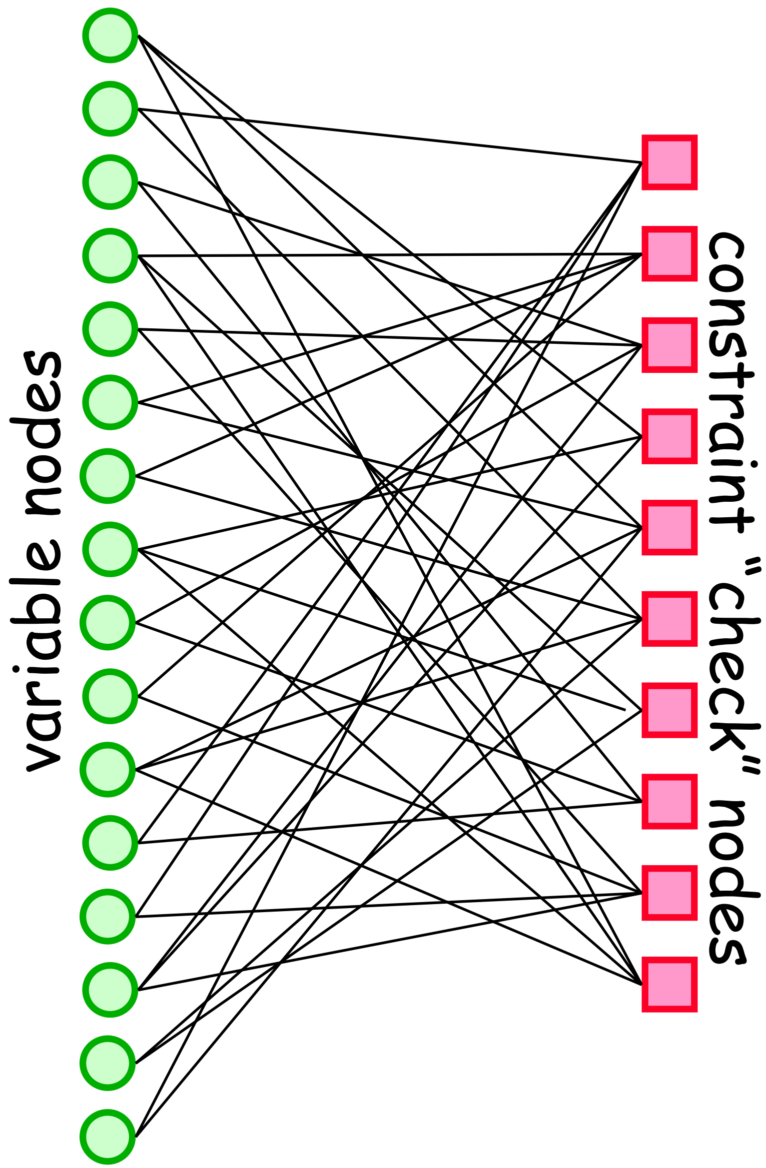
Iterative Decoding

Decoded: (1 1 1 0 0 1 0 0 0 1 1 1 1 1 0 1)

Step 8

$$
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
$$

Factor Graph of Parity-Check Matrix

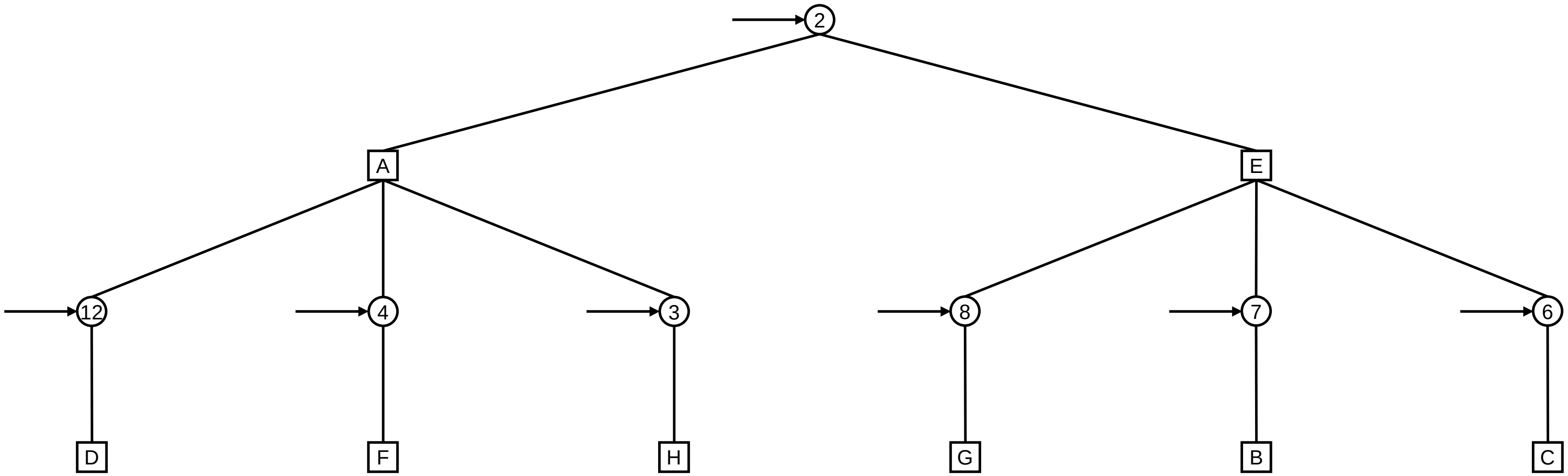


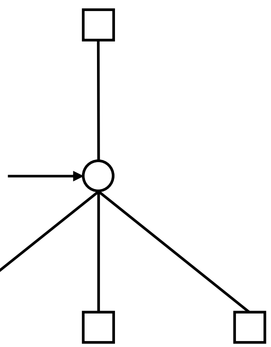
$$H = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 &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

Message Passing Decoder

- N=16 variables, N-K=10 constraints
- number of constraints for a variable = Hamming weight of the corresponding column
- number of variables for a constraint = Hamming weight of the corresponding row
- message passing erasure decoder
- message = value or erasure
- not every decodable received word can be decoded using mere message passing
- message passing for other channels
- message = probability distribution

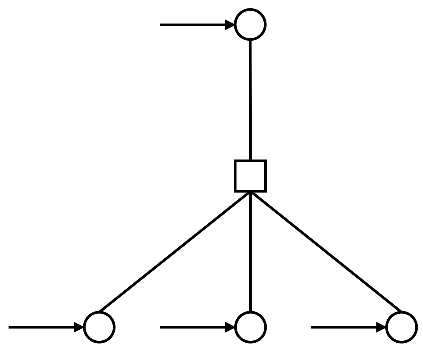
“Tree” Perspective



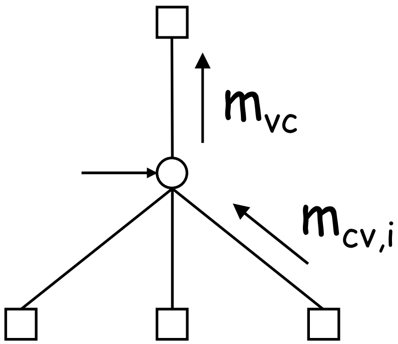


basic elements

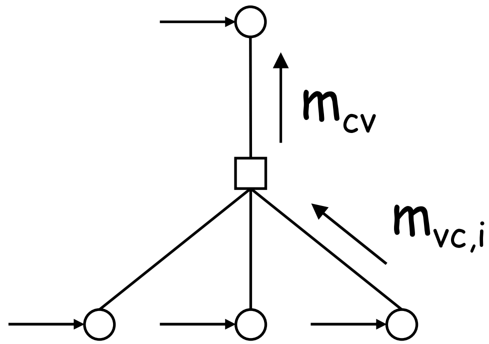
Do you know these codes?



Computation for the BEC



Repeat Code



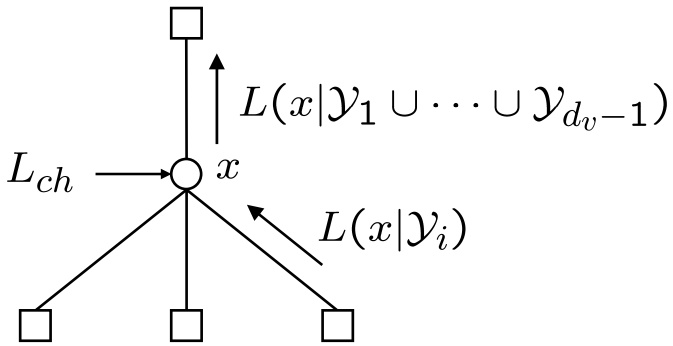
Single Parity-Check Code

$$m_{vc} = \begin{cases} 0, 1 & \text{if at least one } m_{cv,i} \text{ is no erasure or the node itself is no erasure} \\ \triangle & \text{if all } m_{cv,i} \text{ are erasures and the node itself is an erasure} \end{cases}$$

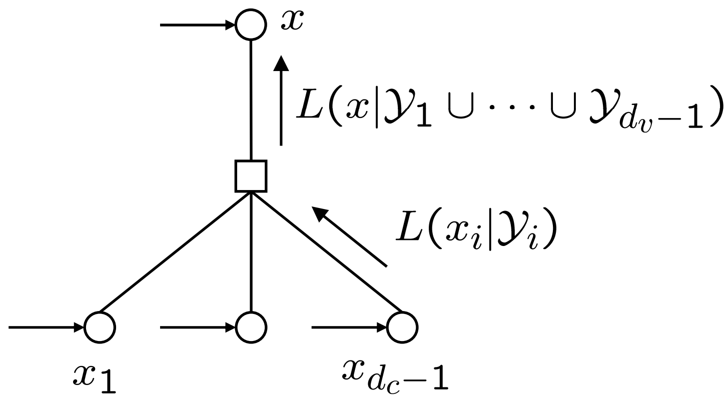
$$m_{cv} = \begin{cases} \sum_i m_{vc,i} & \text{if no } m_{vc,i} \text{ is an erasure} \\ \triangle & \text{if at least one } m_{vc,i} \text{ is an erasure} \end{cases}$$

General Binary Input Channels

Instead of messages 0, 1, Δ we transmit Log-Likelihood Ratios over the edges of the graph.



Repeat Code

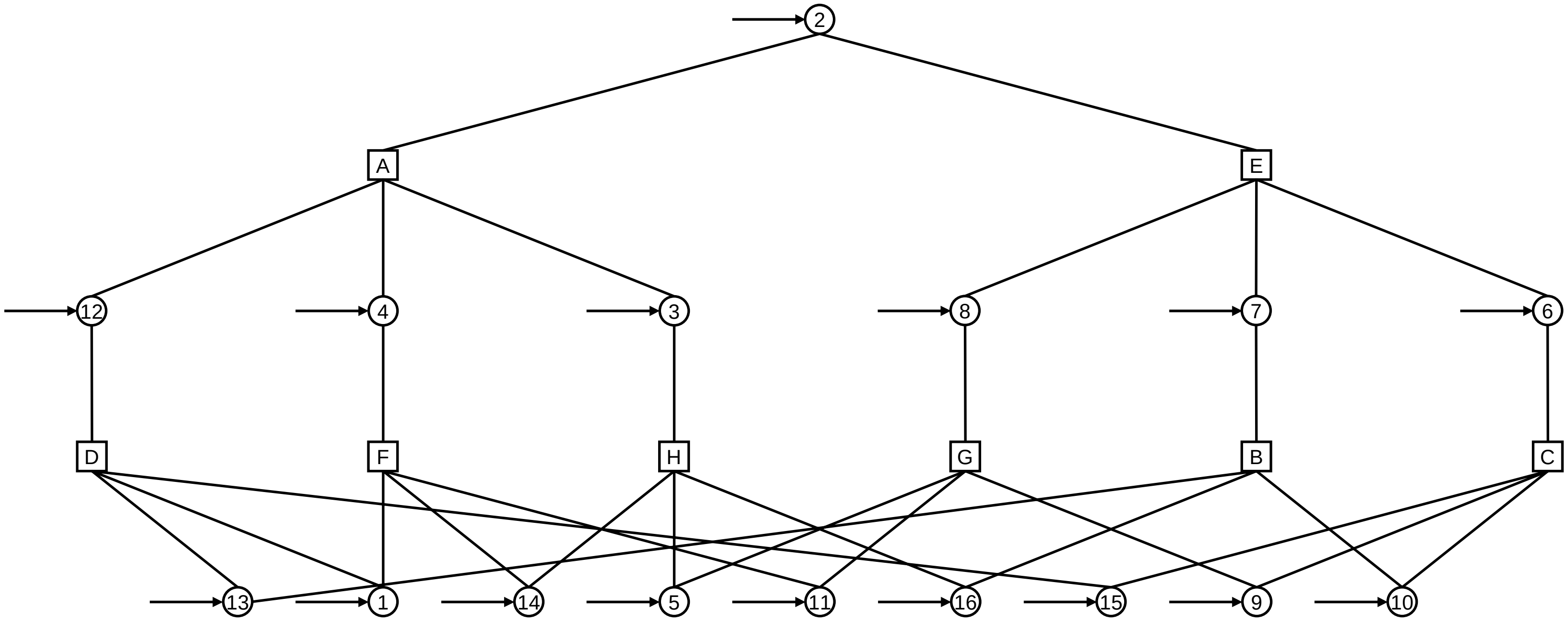


Single Parity-Check Code

$$L(x|\mathcal{Y}_1 \cup \cdots \cup \mathcal{Y}_{d_v-1}) =$$
$$= L_{ch} + \sum_i L(x|\mathcal{Y}_i)$$

$$L(x|\mathcal{Y}_1 \cup \cdots \cup \mathcal{Y}_{d_v-1}) =$$
$$= 2 \cdot \tanh^{-1}\left[\prod_i \tanh \frac{L(x_i|\mathcal{Y}_i)}{2}\right]$$

“Tree” Perspective in Practice



When using random codes with finite block length, the tree perspective is only true for a few steps.

Iterative Decoding

We were lucky. This decoder will not always work…

What property must a parity-check matrix have for iterative decoding to succeed with a high probability?

In effect, iterative decoding is possible in this setting whenever the matrix can be triangulated by simple row & column swaps

Low-Density Parity-Check Matrix

$$
\begin{array}{cc}
\begin{pmatrix}
0&1&0&0&0&0&0&0&0&0&0&1&1&0&0&1\\
0&0&0&1&0&1&1&0&0&1&0&0&0&0&0&0\\
0&0&1&0&1&0&0&0&1&0&0&0&0&1&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0\\
1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1\\
0&1&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&1&0\\
0&0&1&0&0&0&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&1&1&0&0\\
1&0&0&1&0&0&0&1&0&0&1&0&0&0&0&0
\end{pmatrix}
&
\begin{array}{c}
4\\4\\4\\3\\4\\4\\3\\3\\4\\4
\end{array}
\quad \text{Row weights}
\\
\begin{array}{cccccccccccccccc}
3&2&2&3&2&2&2&2&2&2&3&2&2&3&2&2
\end{array}
&
\end{array}
$$

Column weights

The parity-check matrix must have a low density.

Low-Density Parity-Check Coding

Invented by Robert G. Gallager in his PhD thesis, MIT, 1963

From the Website of the Eduard Rhein Stiftung

Bob Gallager

(re-discovered by MacKay from Cambridge, England in 1999)

Regular LDPC Codes

$$
\begin{array}{c|c}
\left(\begin{array}{cccccccccccccccccccc}
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0
\end{array}\right)
&
\begin{array}{c}
6 \\ 6 \\ 6 \\ 6 \\ 6 \\ 6 \\ 6 \\ 6 \\ 6 \\ 6
\end{array}
\\
\begin{array}{cccccccccccccccccccc}
3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3
\end{array}
&
\end{array}
$$

Row weights

$d_c = 6$

Column weights

$d_v = 3$

Columns and rows in the parity-check matrix of a *regular* LDPC code have fixed weights d_v and d_c (as opposed to *irregular* LDPC codes)

Regular Codes

How many '1's are there in the parity-check matrix?

Answer: $n_1 = d_v N \qquad\qquad = 3 \times 20 = 60$

$\qquad\qquad = d_c (N-K) \qquad\quad = 6 \times 10 = 60$

The design rate of the code is: $R = K/N = 1 - d_v/d_c$

If the rows of the parity-check matrix are linearly independent, the design rate is the true code rate

A few thoughts

- Practical implementation typically $(d_v,d_c)=(3,6)$, blocklength $N = 10^5$

  Very, very sparse parity-check matrix (only 6 ones in a row of length 10^5, 3 ones in a column of 5×10^4)
- How are the distance properties of a code affected by the low-density property of H?
- Will the decoder performance depend on the particular H or only on the code?
- How likely is it that a random code has a low-density parity-check matrix? Can we "de-densify" a high-density parity-check matrix?
- How well will the message-passing decoder work in practice? Can it approach capacity?

Concentration theorems
(Luby, Mitzenmacher, Shokrollahi, Spielman / Richardson & Urbanke)

- We will only consider expected decoding behavior over all graphs of a given degree distribution (not the behavior for one specific graph/code)

- Concentration: the probability that the performance of a specific code diverge by ε from the expected performance over all graphs converges to 0 exponentially in the code length N

  ⇨ expected performance suffices

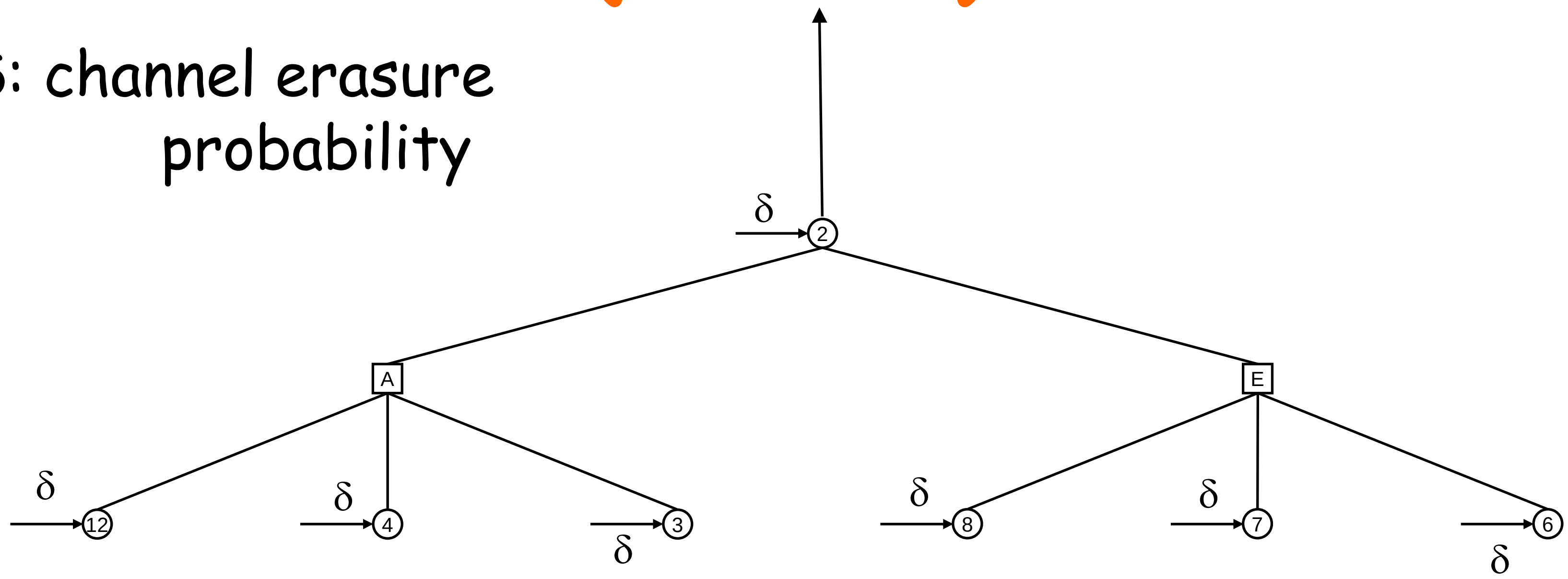
- Cycle-free behavior: for any iteration number n_{it}, one can choose a code length N so that the expected performance at iteration n_{it} is as near as desired to the decoder performance under the tree assumption

  ⇨ tree behavior suffices for $N \rightarrow \infty$

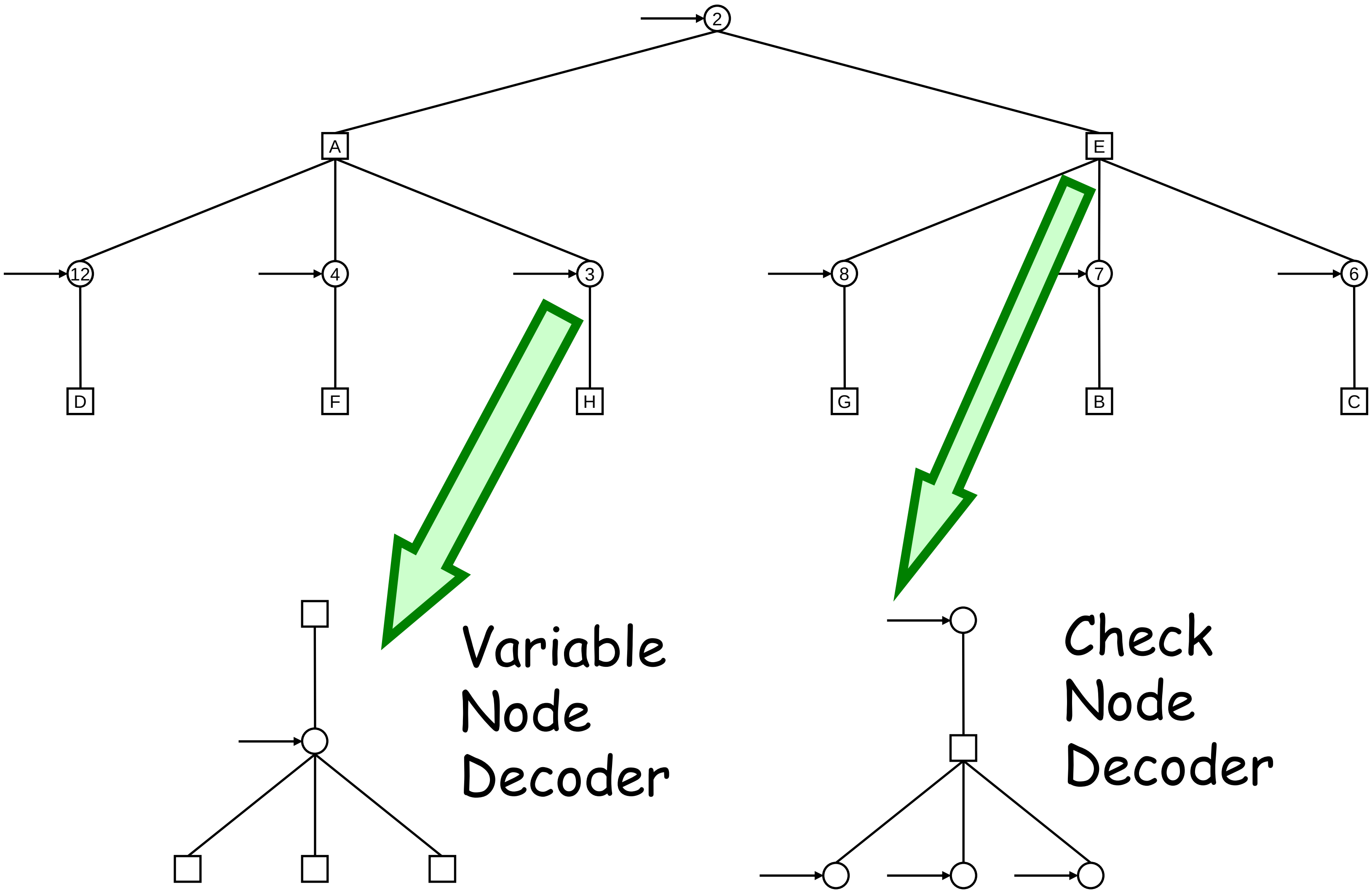
Tree Decoding Performance for the BEC

P(erasure) = ?

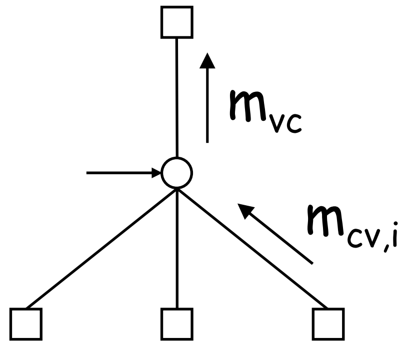
δ: channel erasure probability



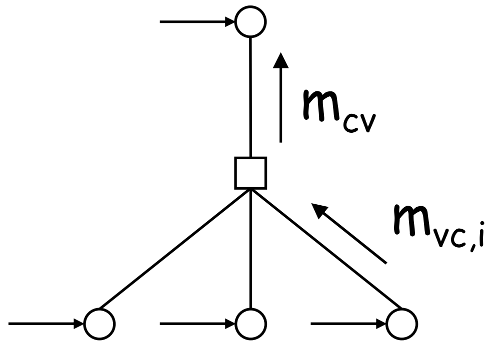
Reminder: Elements of the Tree Perspective



Reminder: Decoding for the BEC



Repeat Code



Single Parity-Check Code

$$m_{vc} = \begin{cases} 0, 1 & \text{if at least one } m_{cv,i} \text{ is no erasure or the node itself is no erasure} \\ \Delta & \text{if all } m_{cv,i} \text{ are erasures and the node itself is an erasure} \end{cases}$$

$$m_{cv} = \begin{cases} \sum_i m_{vc,i} & \text{if no } m_{vc,i} \text{ is an erasure} \\ \Delta & \text{if at least one } m_{vc,i} \text{ is an erasure} \end{cases}$$

“Simulating Simulating”
(Tom Richardson, ISIT 2004, Chicago)

- Simulation: run message-passing decoding for codewords transmitted over a binary erasure channel and measure the resulting error probability
- Simulating simulation: compute probability distributions of messages passing through the decoder
- Instead of m_{vc} and m_{cv}, compute $^*m_{vc} = P(m_{vc} = 0,1,\Delta)$ and $^*m_{cv} = P(m_{cv} = 0,1,\Delta)$

Question

What is $^*m_{vc}$ at iteration 0 ?

Notation: $^*m_{vc} = \big(\, P(m_{vc} = 0), P(m_{vc} = 1), P(m_{vc} = \Delta) \,\big)$

Intuitively: $^*m_{vc}(0) = \big(\, (1-\delta)/2, (1-\delta)/2, \delta \,\big)$

Why is this correct??

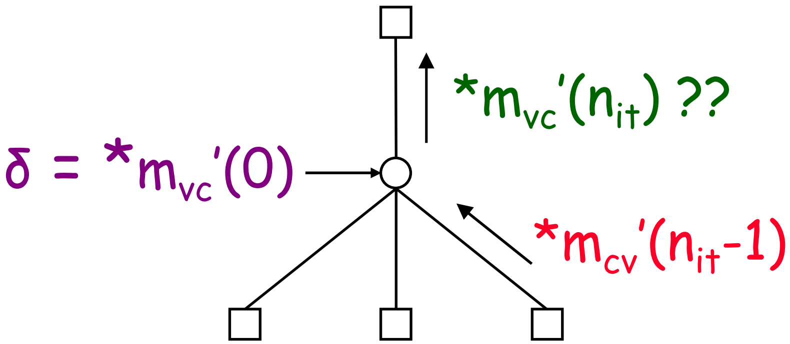
Linear codes…

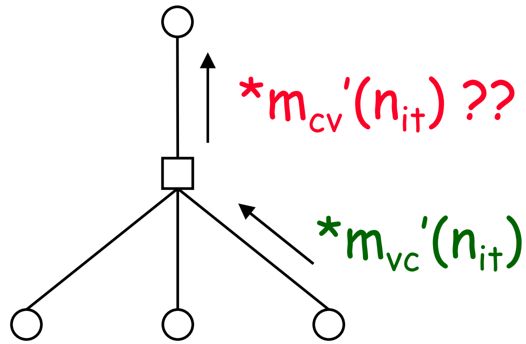
It suffices to consider binary messages

$m_{vc/cv}'(n_{it})$ = 1 if erasure, 0 if non-erasure

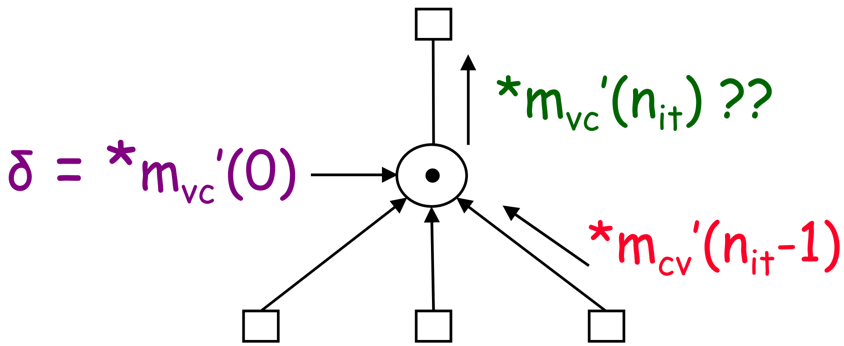
and track $^*m_{vc/cv}'(n_{it}) = P\big(\, m_{vc/cv}'(n_{it}) = \Delta \,\big)$

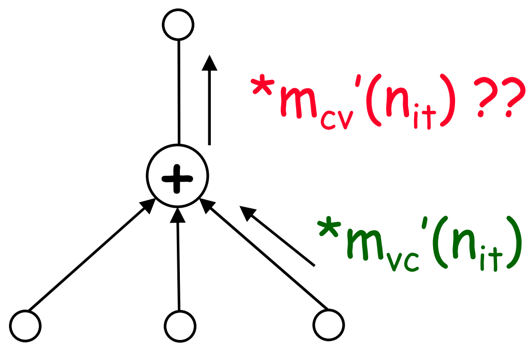
Combining Erasure Probabilities





Equivalently:



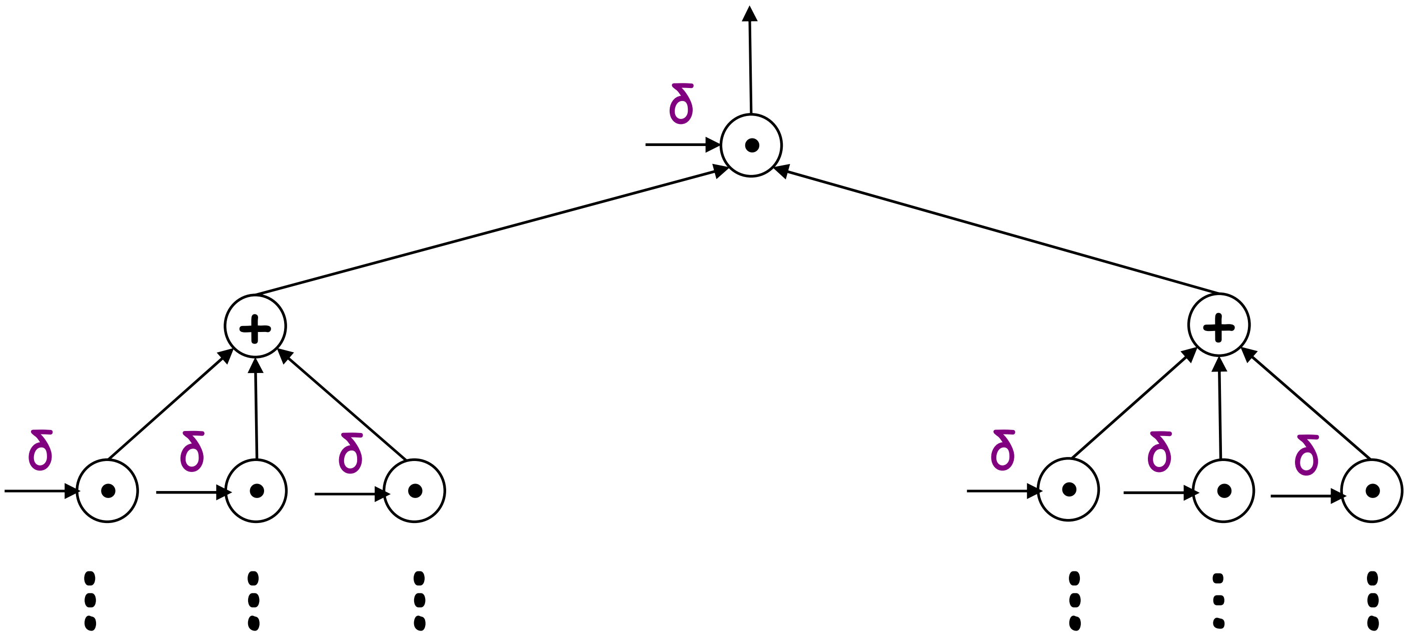


AND - operation

OR - operation

AND-OR Tree

$P(1) = ?$



Let's simplify notation:

$$p_0 = \delta = {}^*m_{vc}'(0)$$

$$q_k = {}^*m_{cv}'(k) \; ,$$

$$p_k = {}^*m_{vc}'(k)$$

Solution

$$q_k = 1 - (1 - p_{k-1})^{d_c - 1}$$

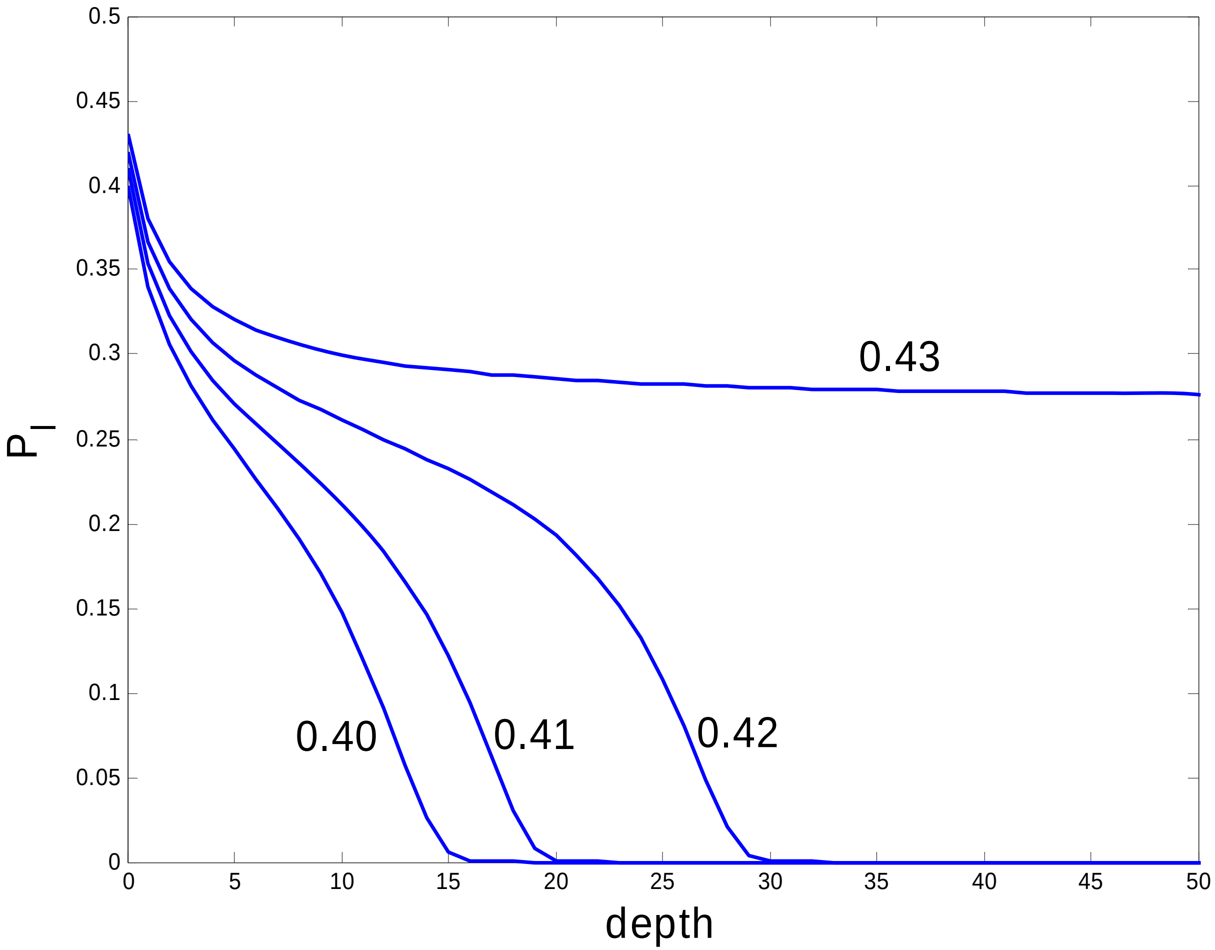
$$p_k = p_0 \, q_k^{d_v - 1}$$

$$p_k = p_0 \left(1 - (1 - p_{k-1})^{d_c - 1}\right)^{d_v - 1}$$

Applying the recursion

$$p_{k+1} = p_0(1-(1-p_k)^{d_c-1})^{d_v-1}$$

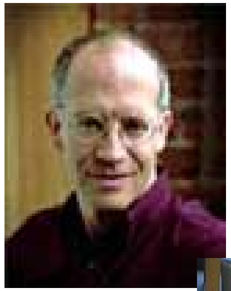
d_v=3 d_c=6



Threshold: 0.42944

BEC & Irregular LDPC Codes

Luby, Mitzenmacher, Shokrollahi & Spielman



Michael G. Luby was at Univ. of Berkeley and founded Digital Fountain after co-inventing irregular LDPC codes

Michael Mitzenmacher got his PhD from Berkeley in 1996, worked for DEC Research, then joined Harvard University in 1999 as Assistant Professor

Daniel A. Spielman got his PhD from MIT in 1995, did a post-doc in Berkeley 1995-96, then went back to MIT as Assistant Professor

M. Amin Shokrollahi got his Dipl.-Ing. from the Univ. of Karlsruhe, his Dr. from the Univ. of Bonn. 1991, worked as a researcher in Berkeley, joined Bell Labs from 1998 to 2000, then Digital Fountain. He has been a Professor at EPF Lausanne since 2003.

Entropy/Uncertainty

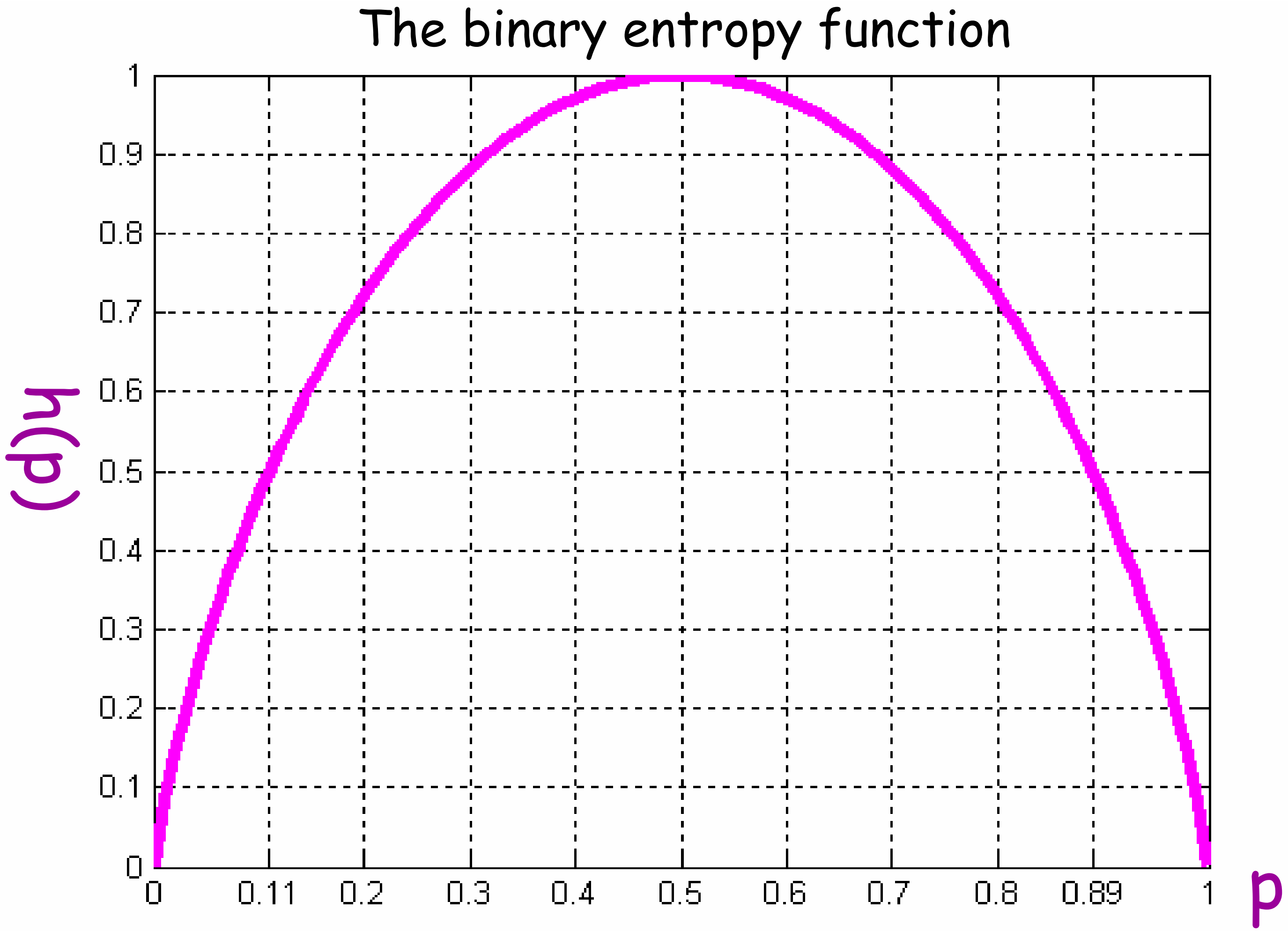
$$H(P_X) = -\sum_x P_X(x) \log_b P_X(x)$$

- $H(P_X)$ is the entropy of the probability distribution of X
- The word "entropy" is used because of the similarity between the formula for H and the entropy in physics
- $H(P_X)$ is a measure for our uncertainty about the value of X
- Alternatively, we write $H(X)$ for $H(P_X)$
- Keep in mind that $H(.)$ is always a function of a probability distribution!

Binary Entropy Function

$L = 2, P_X(0) = p, P_X(1) = 1\text{-}p$

$H(X) = h(p)$



Entropy Bounds

For an L-ary random variable X,

$$0 \leq H(X) \leq \log_b L$$

with equality on the right when $P_X(x) = 1/L$ for all x,
with equality on the left when $P_X(x) = 1$ for one x
and $P(X) = 0$ for all other x.

Example:

Random experiment: pick a student at random

Random variable X: 1 if female, 0 if male
Y: 1 if wears glasses, 0 if not
Z: 1 if Norwegian, 0 if not

Equivocation

The equivocation, or average conditional entropy, of X given Y is defined as

$$H(X|Y) = \sum_y P_Y(y)\, H(P_{X|Y=y})$$

Warning: do not confuse with $H(X|Y=y) = H(P_{X|Y=y})$

Properties of the equivocation

"Conditioning can only reduce entropy"

$$0 \leq H(X|Y) \leq H(X)$$

equality on the left if Y essentially determines X
equality on the right if X and Y are independent

Warning: $H(X|Y=y)$ can be larger than $H(X)$!!
"Conditioning on events can increase entropy"

Chain rule: $H(XY) = H(Y) + H(X|Y)$

$$H(X_1 \ldots X_N) = H(X_1) + H(X_2|X_1) + \ldots + H(X_N|X_1 \ldots X_{N-1})$$

Mutual Information

The mutual information between X and Y is

$$I(X;Y) = H(X) - H(X|Y)$$
$$= H(Y) - H(Y|X)$$

$$0 \leq I(X;Y) \leq \min[H(X),H(Y)]$$

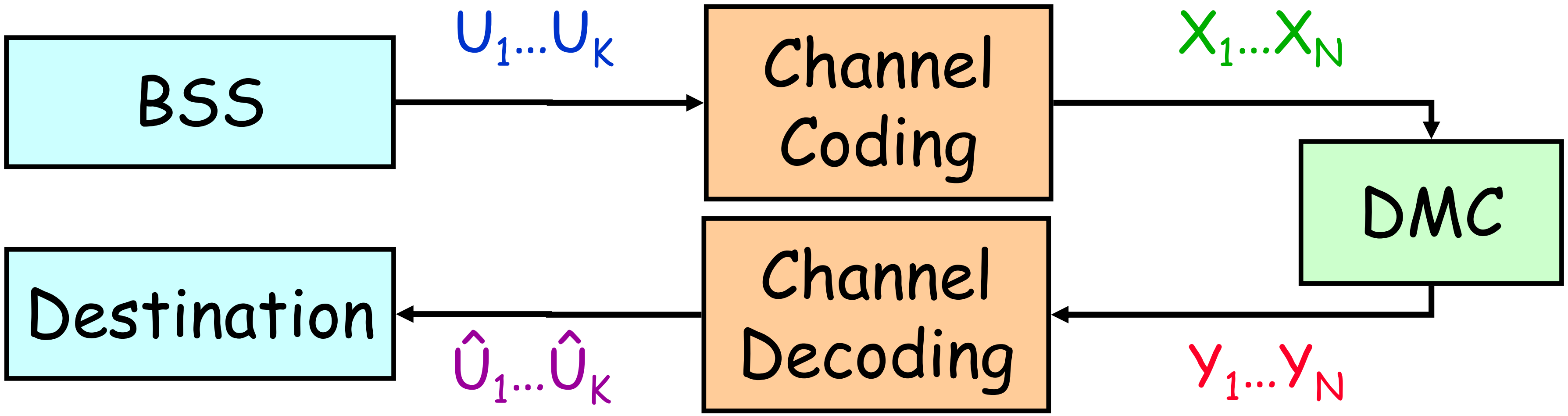
equality on left if X and Y independent
equality on right if X essentially determines Y
or vice-versa

$I(X;Y)$ is a function of the joint distribution P_{XY}

The essence of Mutual Information

- $I(X;Y)$ tells us how much uncertainty is reduced about X by knowing Y (or vice-versa)
- $I(X;Y)$ tells us how much information X gives about Y (or vice-versa)
- $I(X;Y)$ is a very general type of correlation measure: it is 0 when X and Y are independent (and thus uncorrelated) and maximized when X is a function of Y or vice-versa

Mutual Information



$$\lim_{N\to\infty} P_B = 0 \Rightarrow \lim_{N\to\infty} I(\underline{U};\underline{\hat{U}}) = K$$

$$\Rightarrow \lim_{N\to\infty} I(\underline{U};\underline{\hat{U}})/N = \lim_{N\to\infty} I(\underline{X};\underline{Y})/N = R$$

$$\boxed{\begin{aligned}\lim_{N\to\infty} P_b = 0 \Rightarrow \lim I(U_i;\hat{U}_i) &= \lim I(U_i;\underline{Y}) \\ &= \lim I(X_i;\underline{Y}) \\ &= \mathbf{1}\ , \text{ for all } i\end{aligned}}$$

(for binary codes)

Iterative Decoding

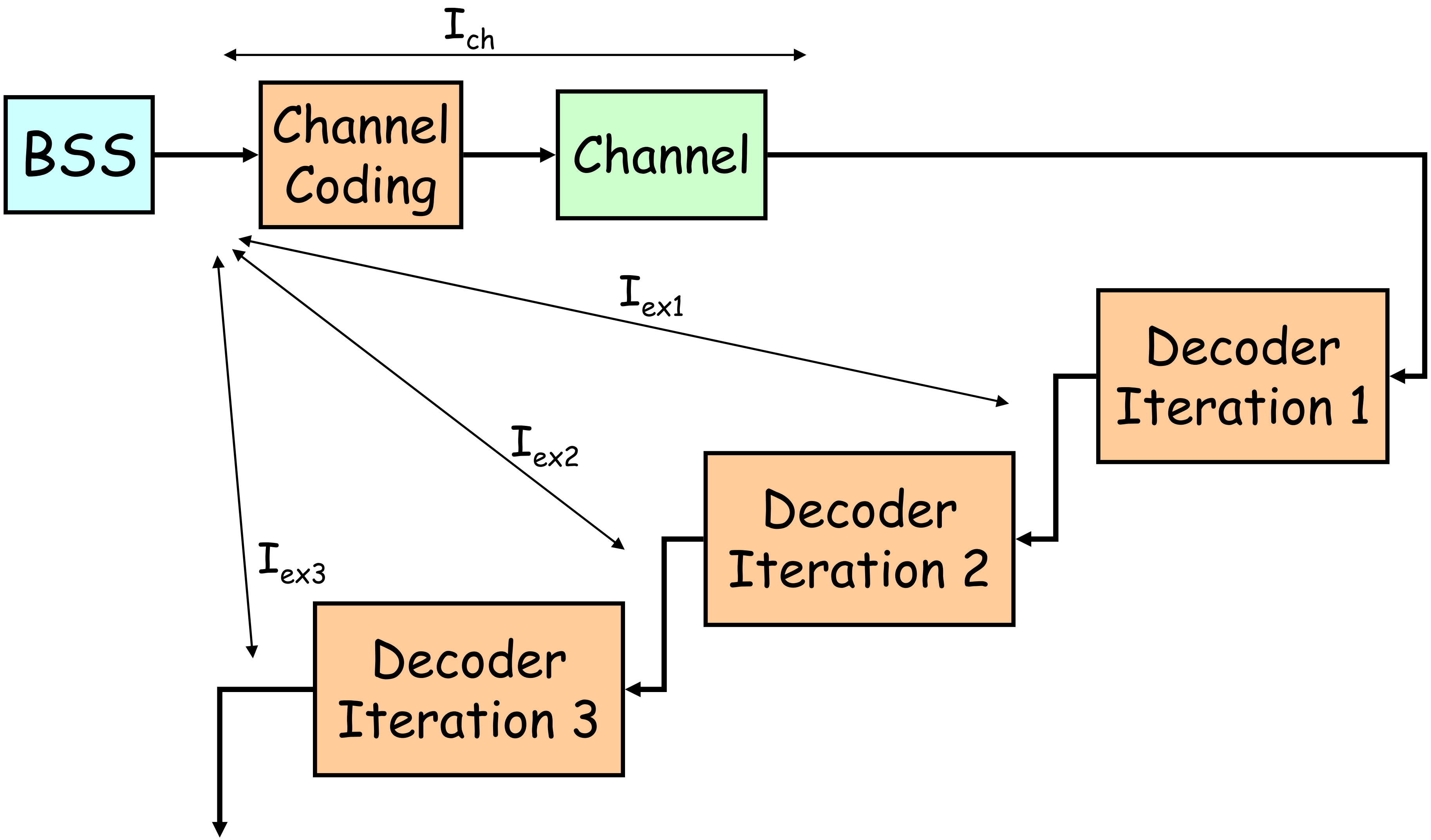
- How does the mutual information evolve in an iterative decoding algorithm?

- We have learned that it is possible to optimize LDPC codes so as to maximize their threshold

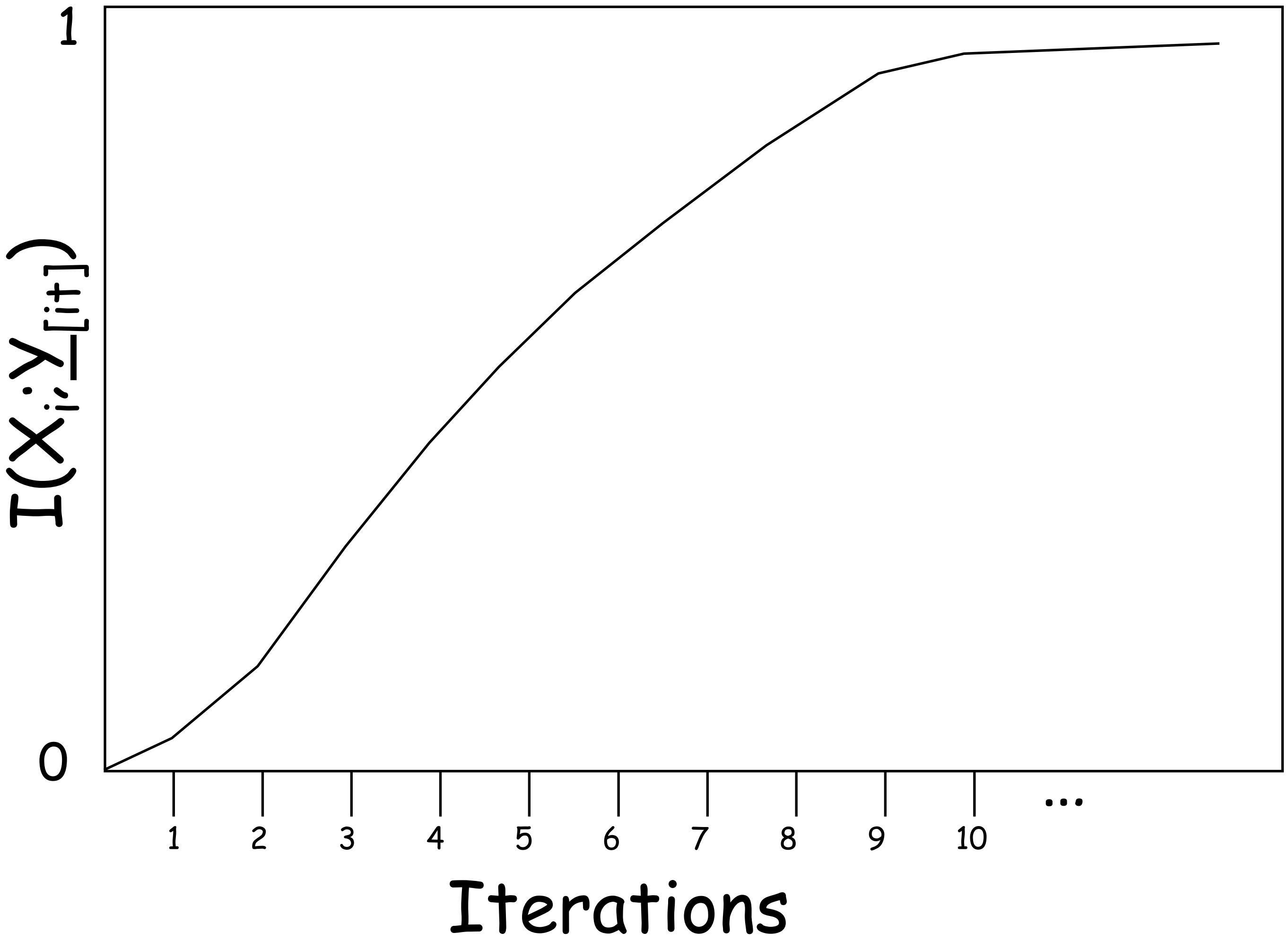
- We will see that we can design capacity-achieving, iteratively decodable families of LDPC codes!!
  (i.e., threshold → capacity)

- What is the implication in terms of mutual information?

Iterative = Cascaded Decoding!



Mutual Information Trajectory



Mutual Information Trajectory

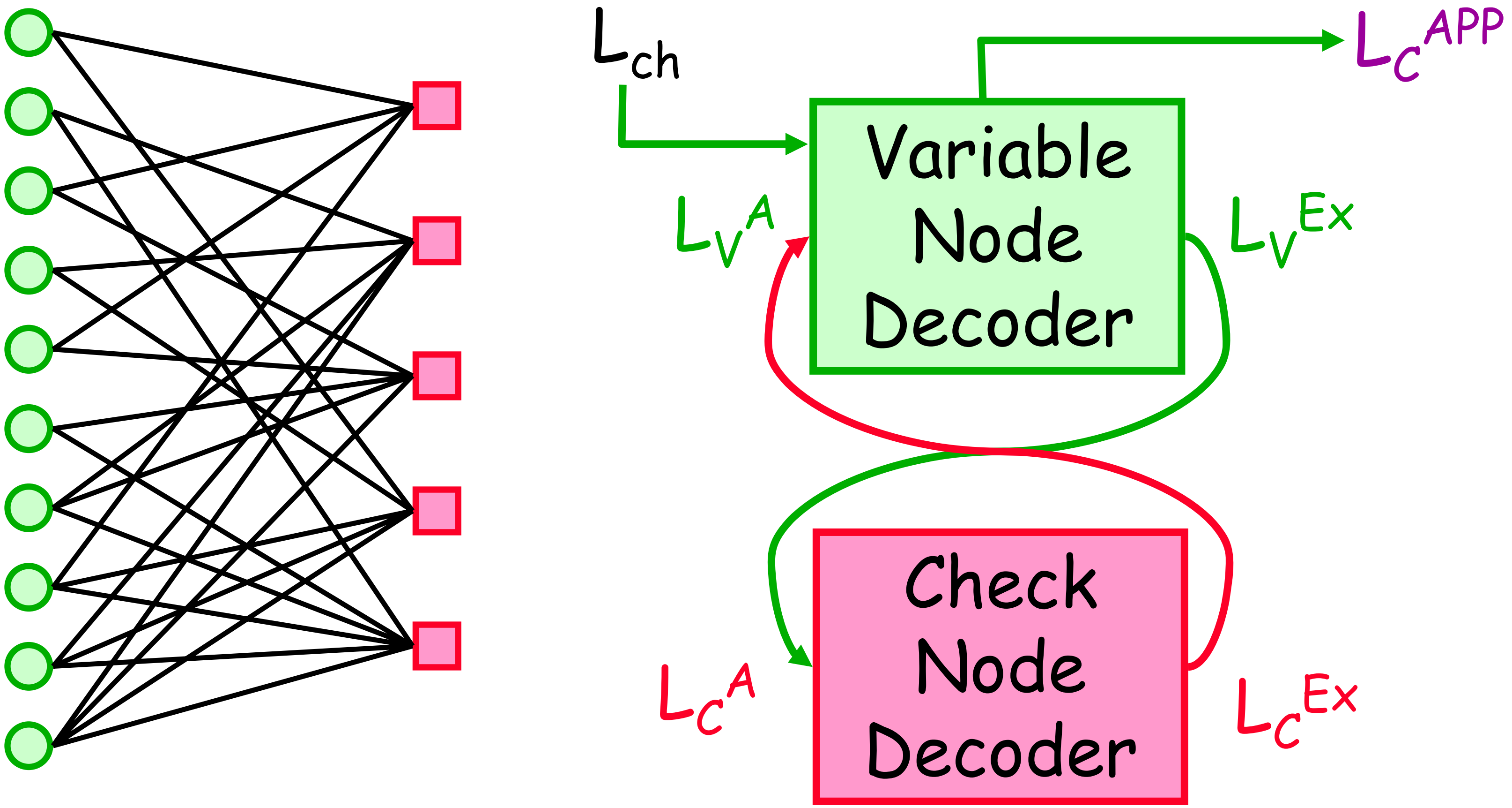
- The L-values calculated in the tree are optimal in the sense of a MAP-calculator, i.e., $L(X_i|\underline{y}_{[it]})$ is a sufficient statistic for $\underline{y}_{[it]}$:

$$I(X_i\,;\,L(X_i|\underline{y}_{[it]})) = I(X_i\,;\,\underline{y}_{[it]})$$

- We can also draw the trajectory at half-iterations (after variable nodes & after check nodes)

- But: the output messages of variable nodes and check nodes are extrinsic L-values, whereas the mutual information trajectory we consider now is for a-posteriori L-values

Message Passing



Tracking of Messages

Tracking of messages would mean tracking of pdfs

(→ Density Evolution)

Instead of tracking the pdfs we reduce the problem to tracking of mutual information between the messages and the codeword which are scalar quantities

$$I_A = \frac{1}{N} \sum_{n=1}^{N} I(X_i; A_i)$$

$$I_E = \frac{1}{N} \sum_{n=1}^{N} I(X_i; E_i)$$

$$I_A^{(1)} = 0$$

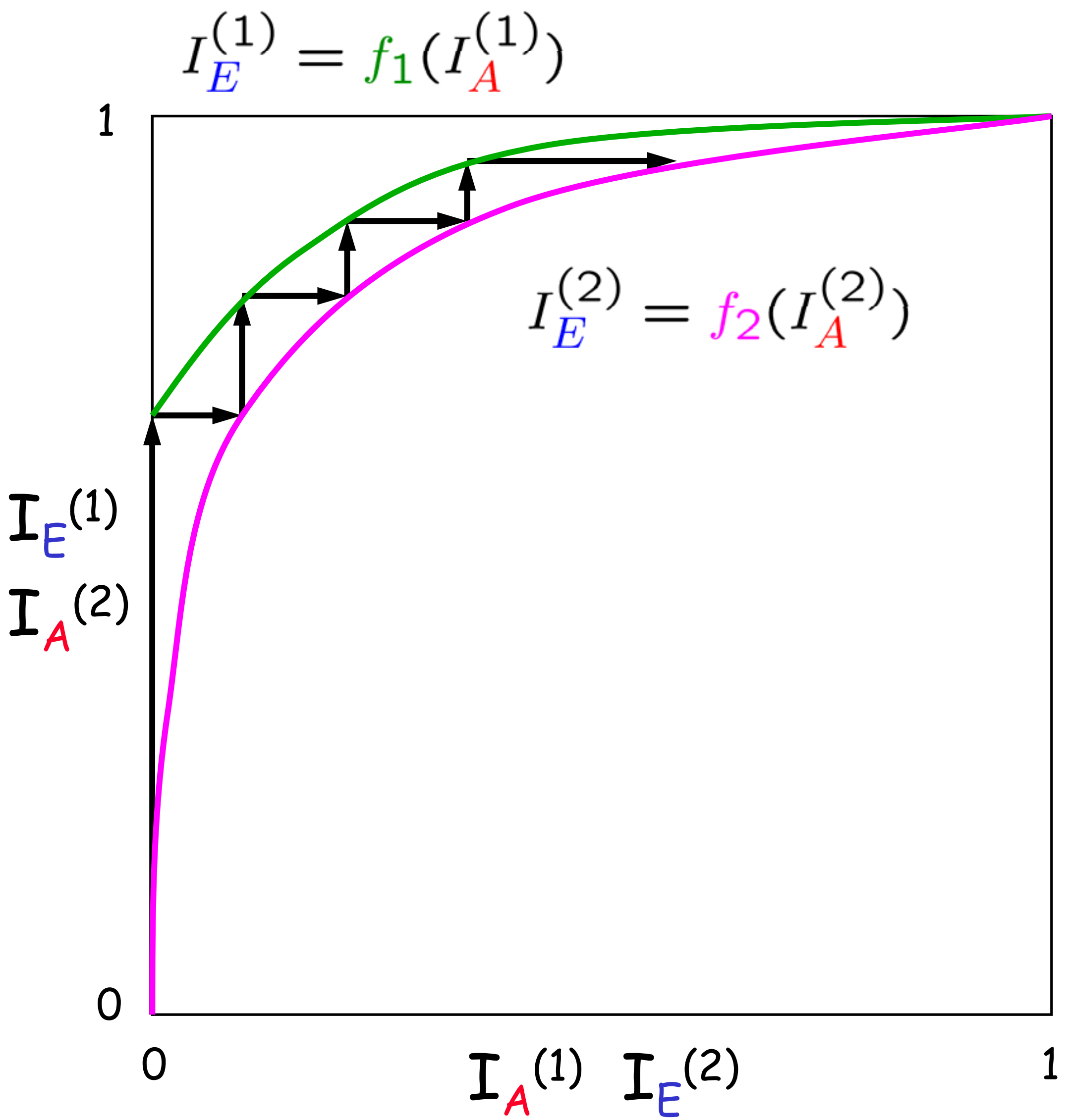
$$\Rightarrow I_E^{(1)} \rightarrow I_A^{(2)}$$

$$\Rightarrow I_E^{(2)} \rightarrow I_A^{(1)}$$

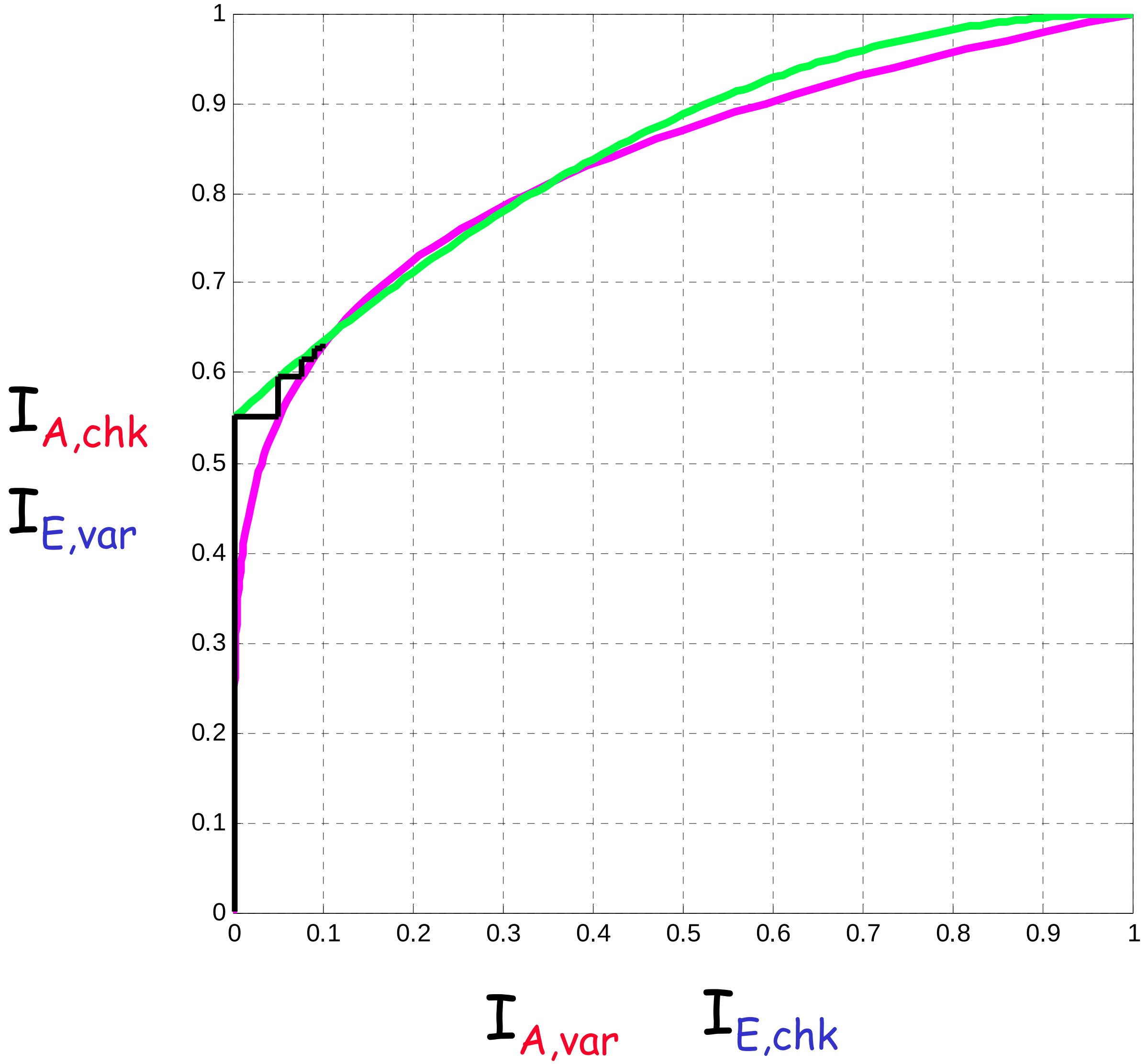
$$\cdots$$

I_A, I_E ..... average symbolwise mutual information

Extrinsic Information Transfer Chart



Intersecting Curves



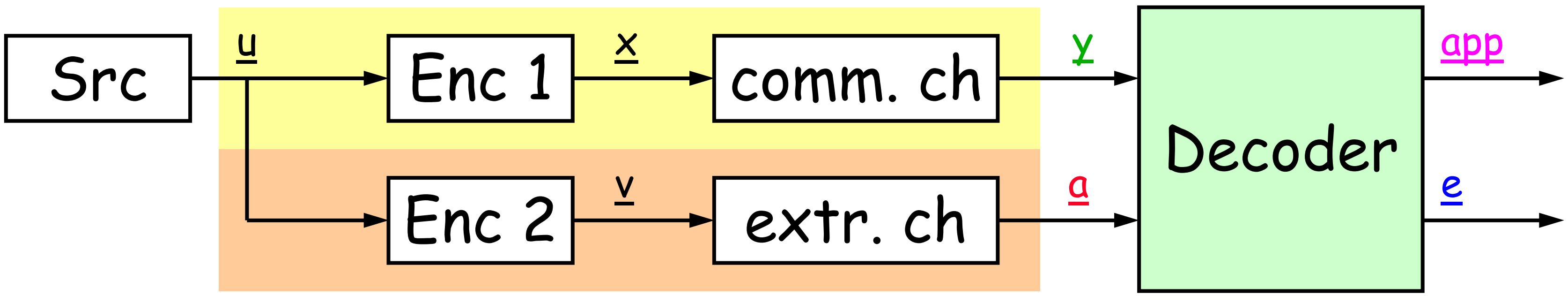
Extrinsic Information Transfer Charts (Stephan ten Brink)

Photo by Jossy Sayir

Stephan did his PhD at the U of Stuttgart, then worked for Bell Labs U.K., then New Jersey. He is currently with RealTek, California.

(Stephan is the person on the right, not on the left)

Computing EXIT transfer functions



A-priori messages are modeled as independent noisy observations of the encoded source.

Assumptions:

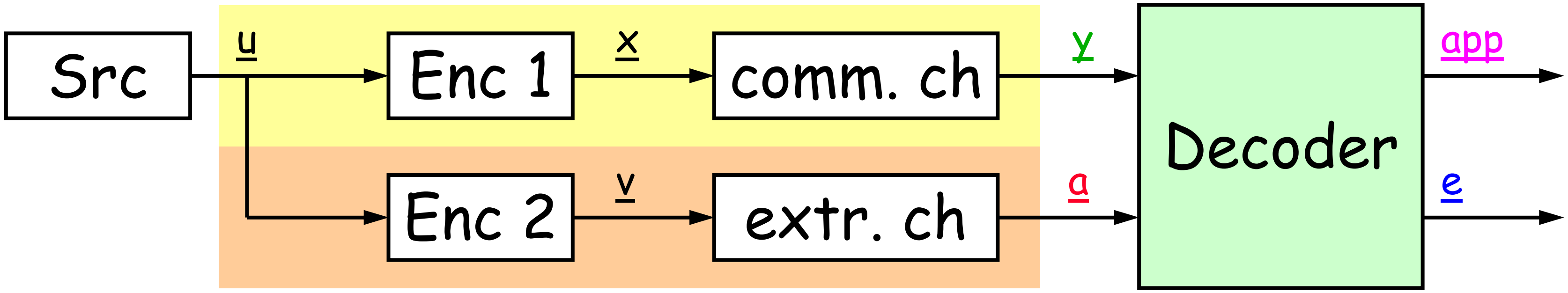
- independent observations
- model for extrinsic channel

$$I_A = I(X_i; A_i) = I(V_i; A_i)$$

$$I_E = I(X_i; E_i) \leq I(X_i; \underline{Y}\,\underline{A}_{\setminus i})$$

with equality if the decoder is "optimal"

Transfer Functions

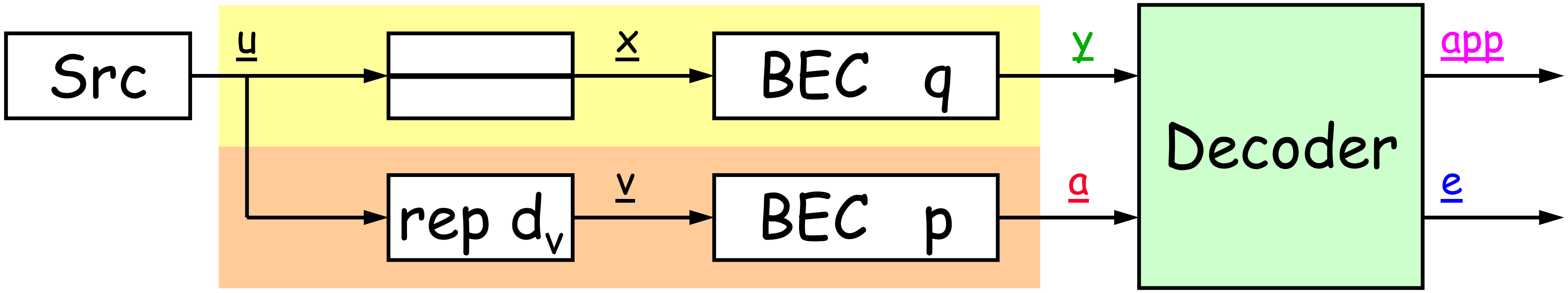


Assuming a model for the extrinsic channel we can vary I_A by varying the channel parameter.

At the output of the decoder we can measure/calculate I_E $\Rightarrow I_E = f(I_A)$

This is only exact if the model of the extrinsic channel is correct!

Variable Nodes and BEC

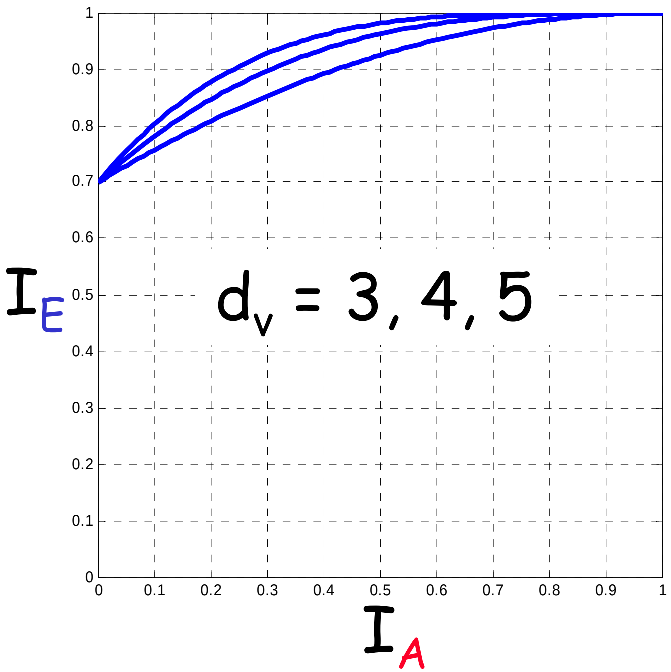


Extrinsic channel is modeled as BEC (exact).

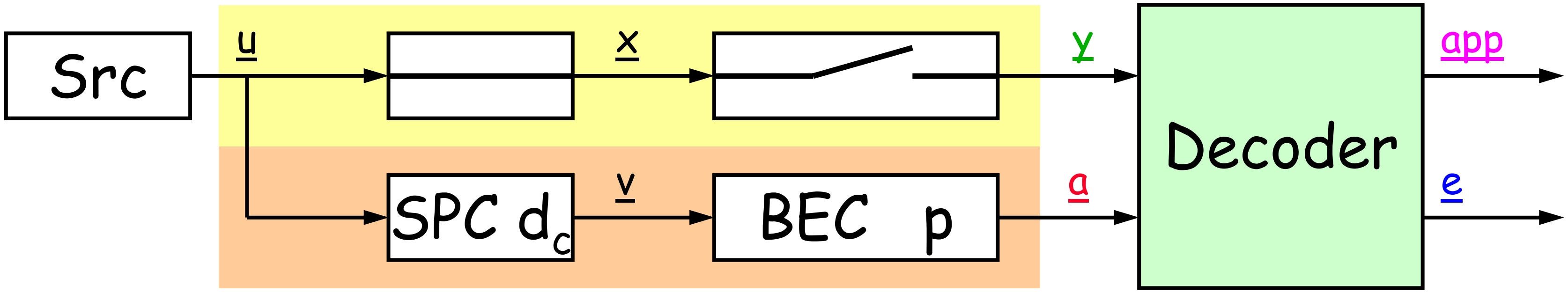
$$I_A = I(X_i; A_i) = I(V_i; A_i) = 1 - p$$

$$I_E = I(X_i; \underline{Y}\,\underline{A}_{\setminus i}) = 1 - qp^{d_v - 1}$$

$$I_E = 1 - q(1 - I_A)^{d_v - 1}$$



Check Nodes and BEC

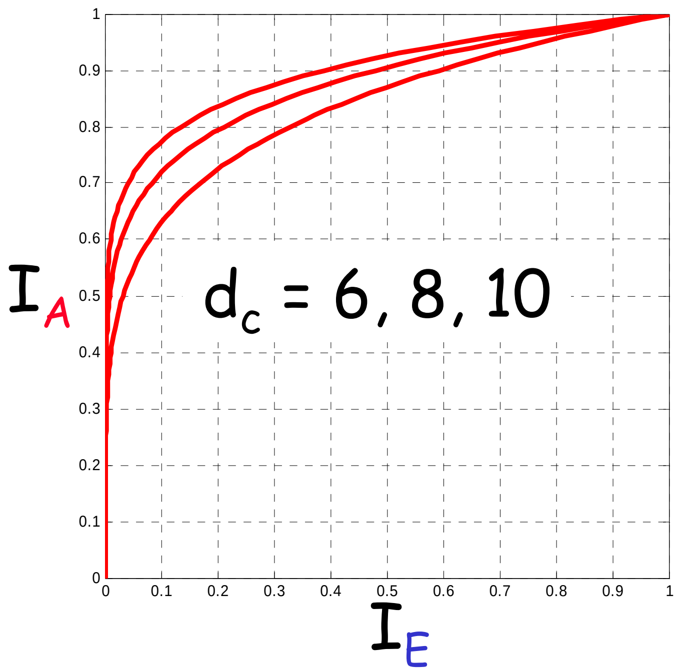


SPC ... single parity check

$$I_A = I(X_i; A_i) = I(V_i; A_i) = 1 - p$$

$$I_E = I(X_i; \underline{Y}\,\underline{A}_{\setminus i}) = (1-p)^{d_c - 1}$$

$$I_E = (I_A)^{d_c - 1}$$



Other Channels

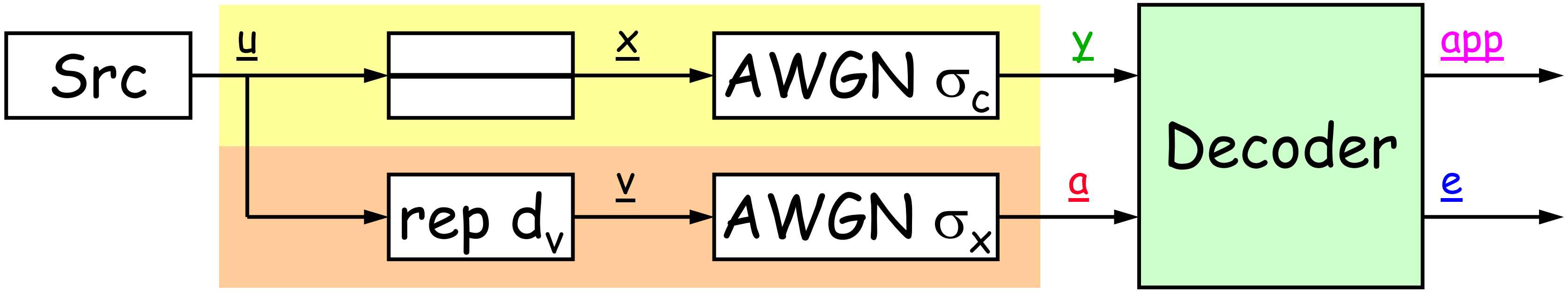
Modeling the extrinsic channel as a BEC is exact if the communication channel is a BEC.

For other communication channels, the assumption of the extrinsic channel is in general an approximation.

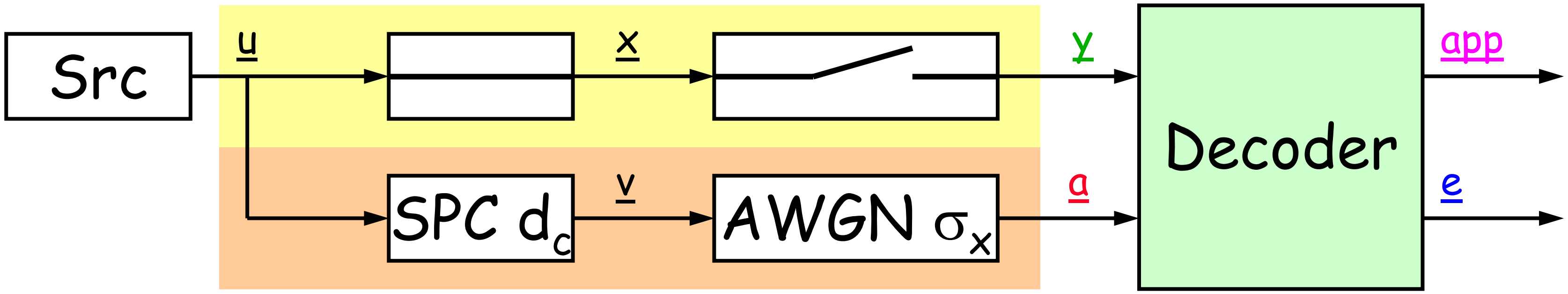
If the communication channel is an AWGN channel, we model the extrinsic channel also as an AWGN, but this is only an approximation!

AWGN Channel

variable nodes



check nodes



Convolutional Codes

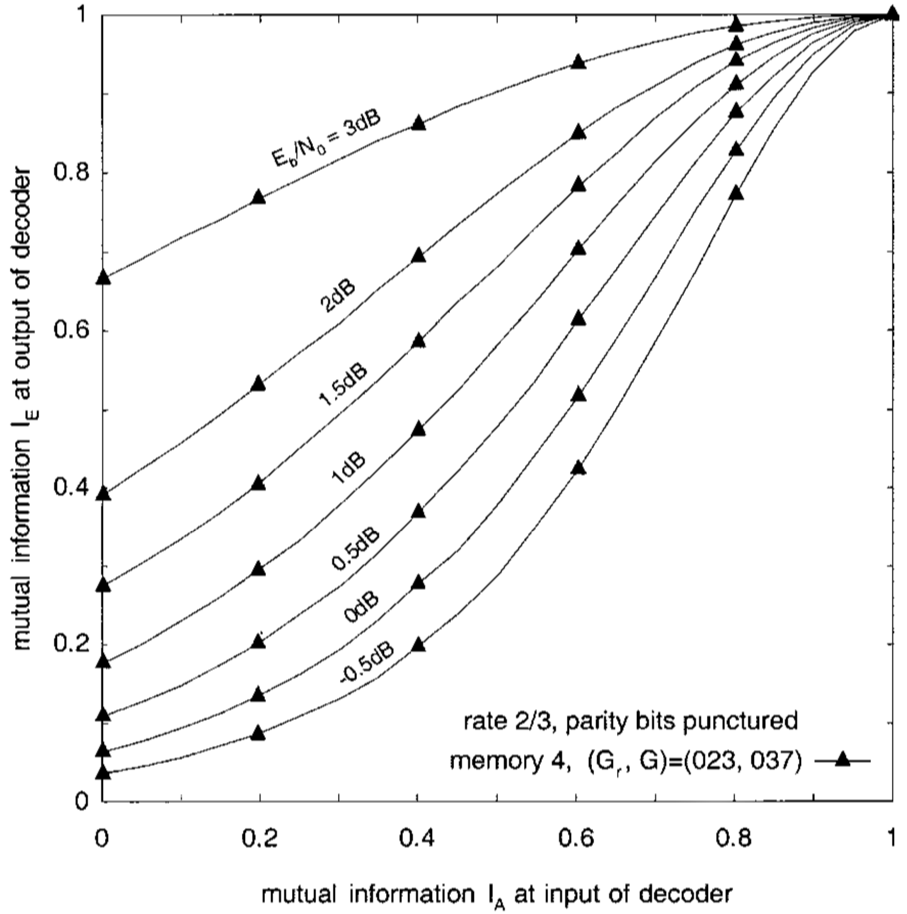
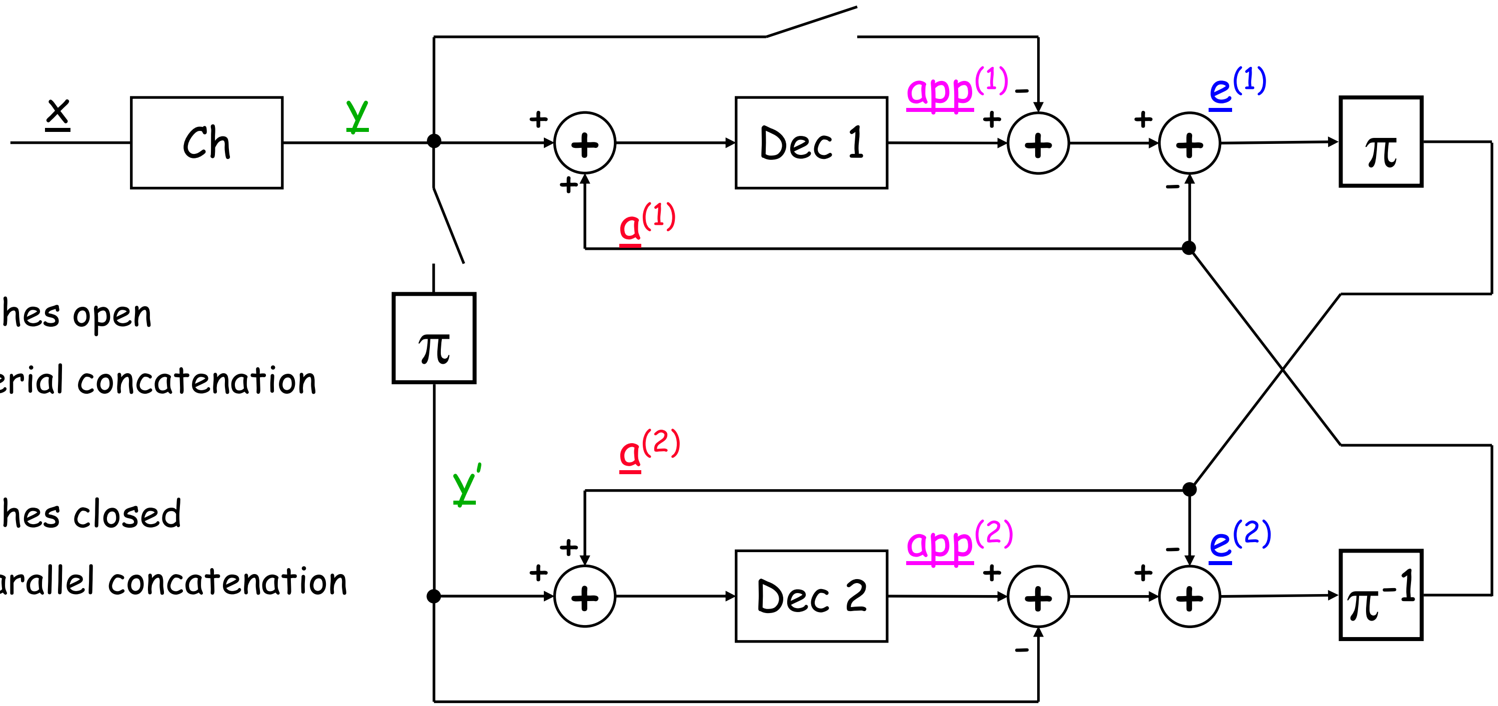


Fig. 2. Extrinsic information transfer characteristics of soft in/soft out decoder for rate 2/3 convolutional code; E_b/N_0 of channel observations serves as parameter to curves.

Stephan ten Brink, "Convergence Behavior of Iteratively Decoded Parallel Concatenated Codes", IEEE Trans. Comm. October 2001

Serial / Parallel Concatenation



switches open

→ serial concatenation

switches closed

→ parallel concatenation

Serial concatenation:

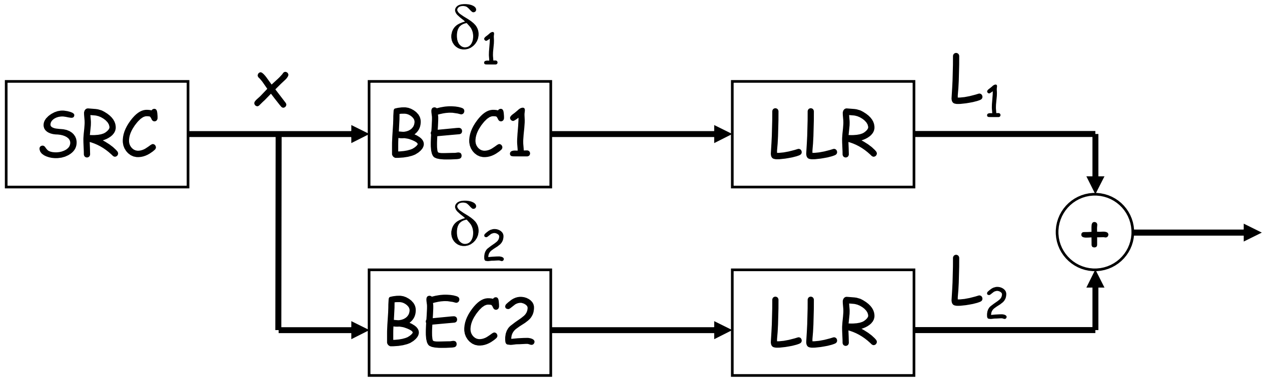
$\underline{e} = \underline{app} - \underline{a}$

Parallel concatenation:

$\underline{e} = \underline{app} - \underline{a} - \underline{y}$

Summing LLRs

What is the effect on mutual information when we add L-values?

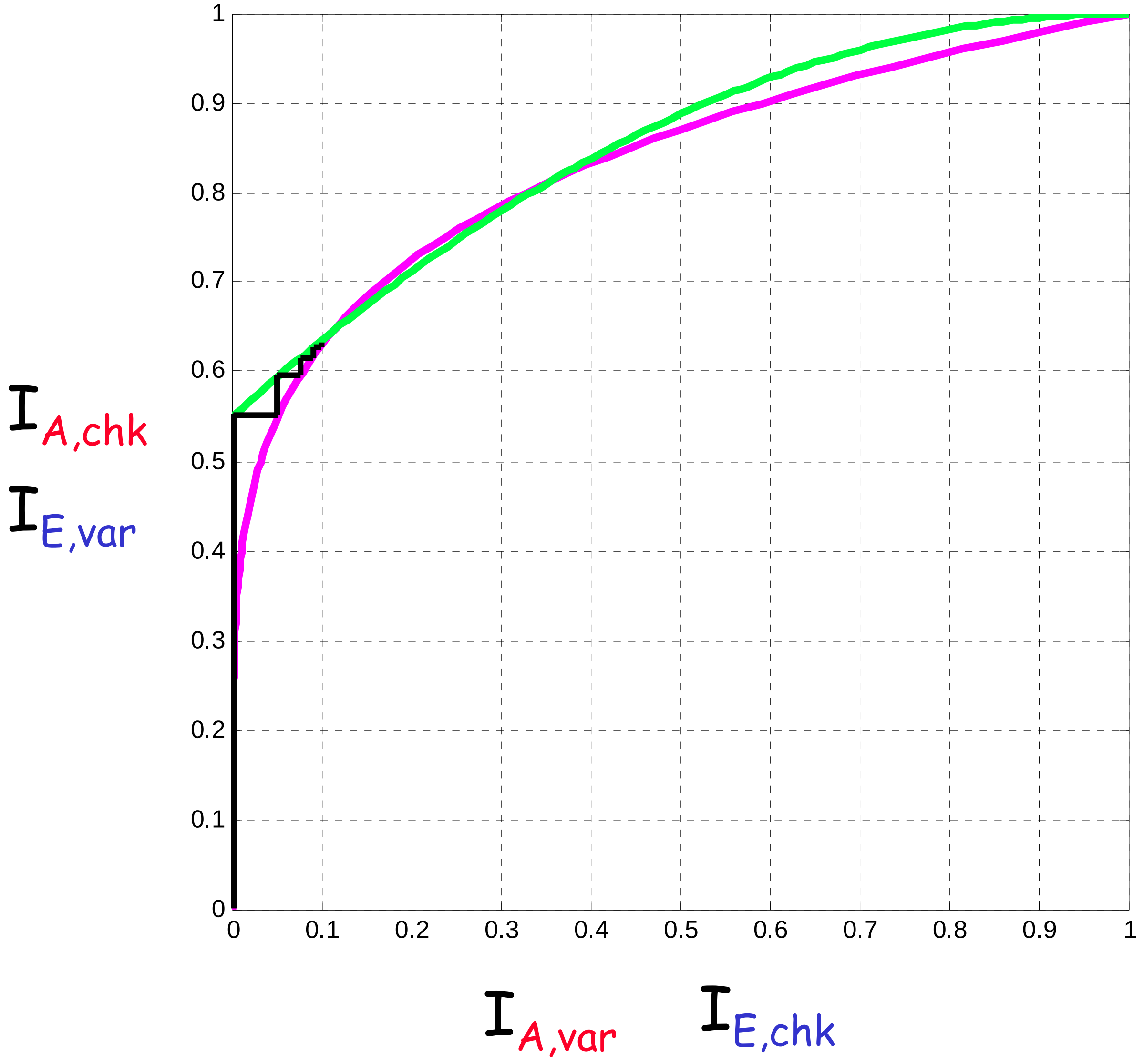


$I_1 = I(X;L_1) = 1 - \delta_1$

$I_2 = I(X;L_2) = 1 - \delta_2$

$$I(X;L_1L_2) = 1 - \delta_1\delta_2 = 1 - (1-I_1)(1-I_2)$$

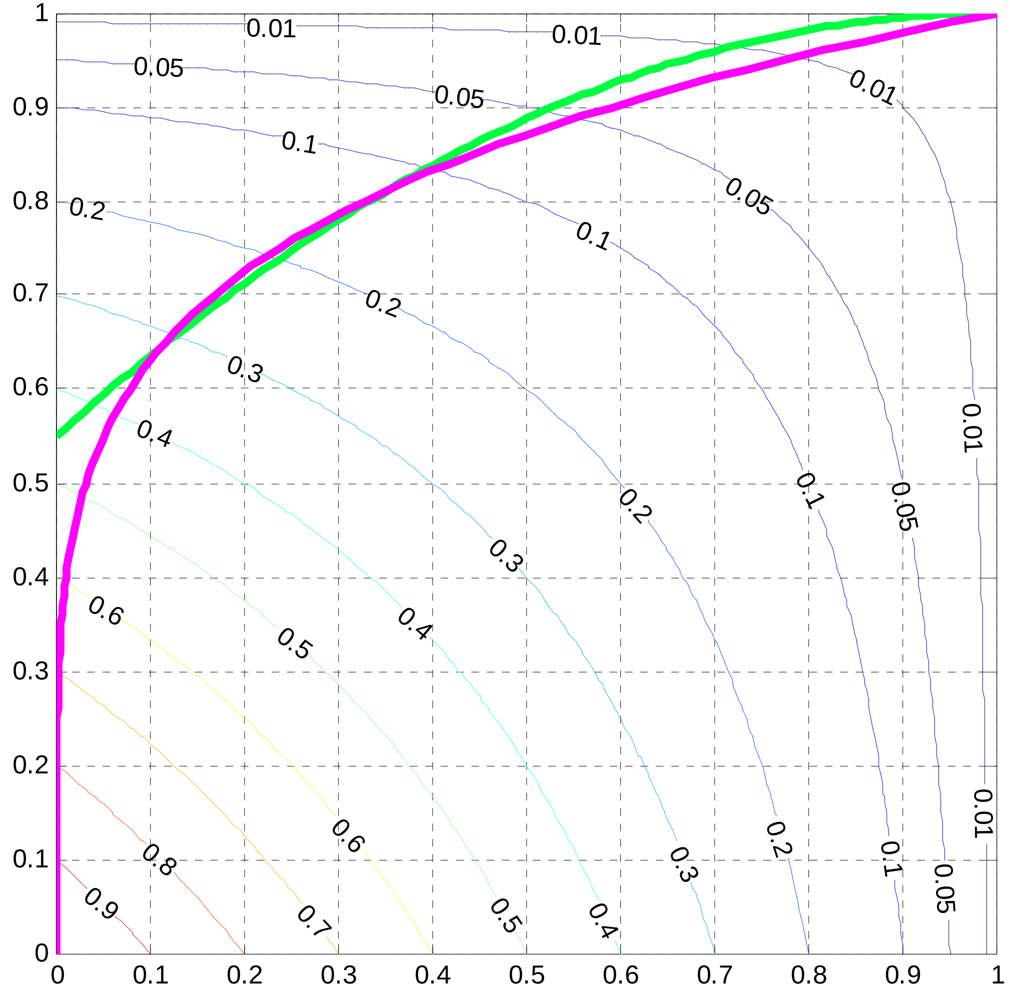
Intersecting Curves



BER from EXIT Chart (BEC)

app = a + e

$$I(X;APP) = 1 - P_b = 1 - (1-I_A)(1-I_E)$$



Independent Observations

Messages received from the extrinsic channel are independent observations, which is only fulfilled if N $\rightarrow \infty$

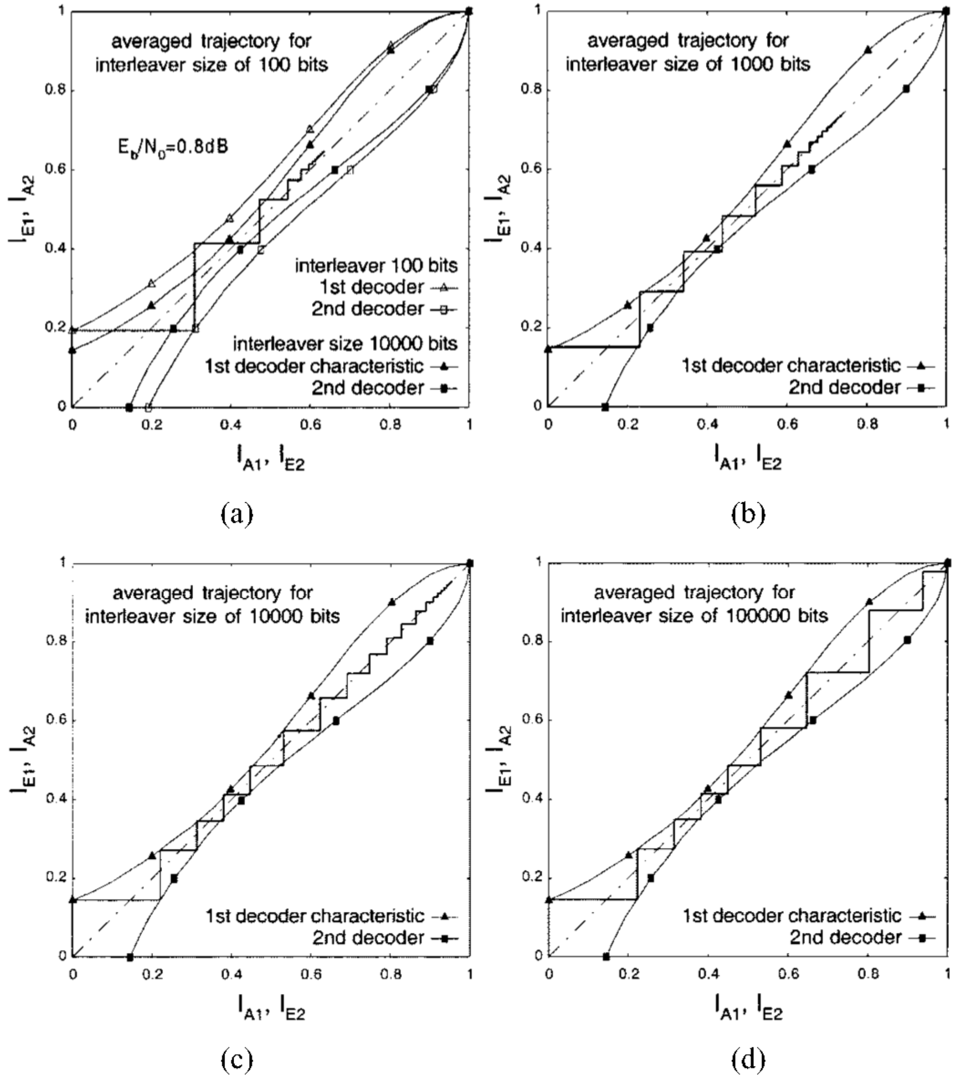


Fig. 8. Averaged decoding trajectories for different interleaver lengths; PCC rate 1/2 memory 4, $(G_r, G) = (023, 037)$; averaged over 10^8 information bits.

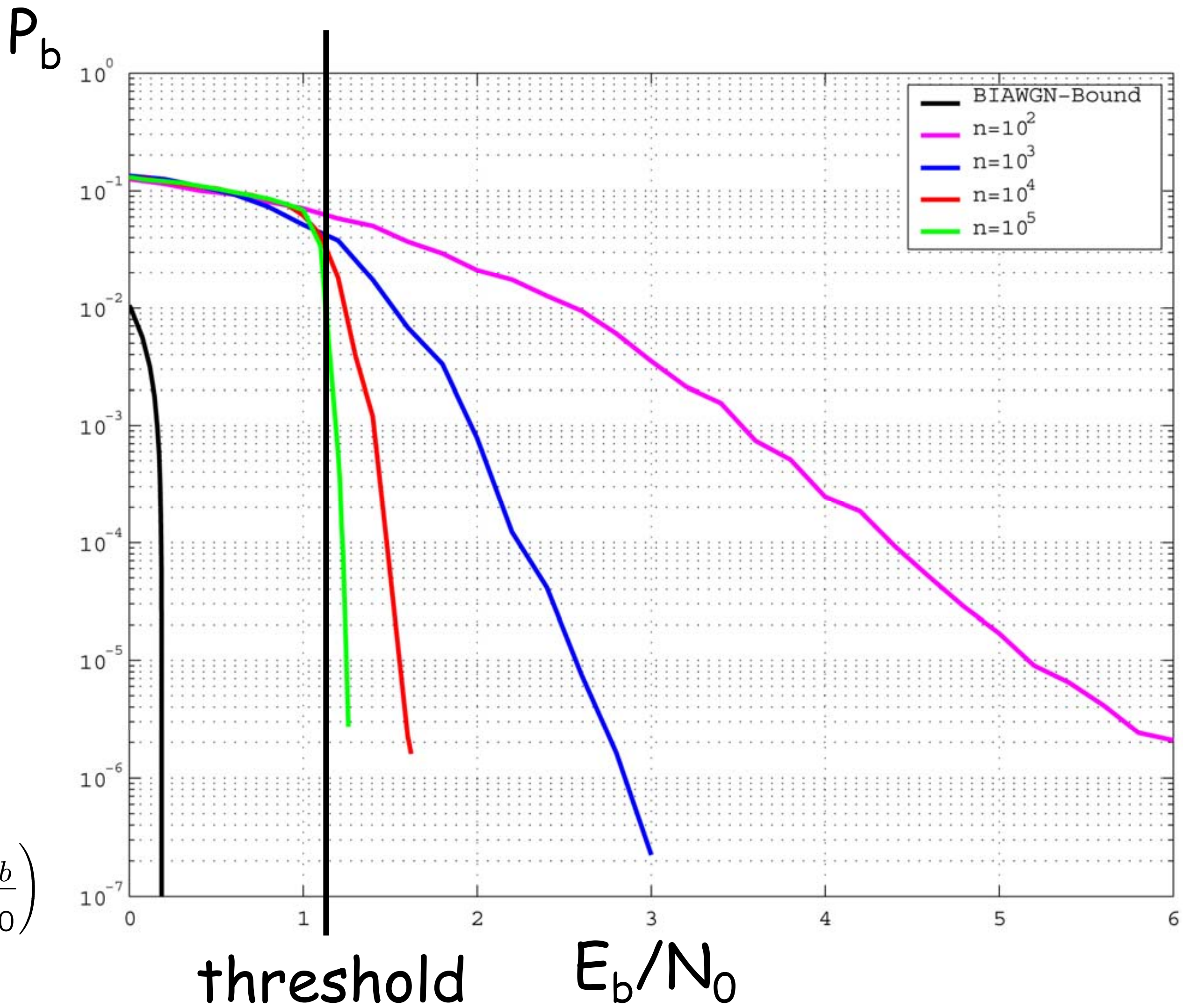
Statistics

We use statistical quantities, which are only correct if $N \to \infty$

$$E_s = R \cdot E_b \qquad \sigma^2 = \frac{N_0}{2}$$

$$E_s := 1$$

$$\frac{E_b}{N_0} = \frac{E_s}{R2\sigma^2} \Rightarrow \sigma^2 = \frac{1}{2R} \cdot \left(\frac{E_b}{N_0}\right)$$



Summary of Assumptions

- Messages received from the extrinsic channel are independent observations, which is only fulfilled if $N \to \infty$

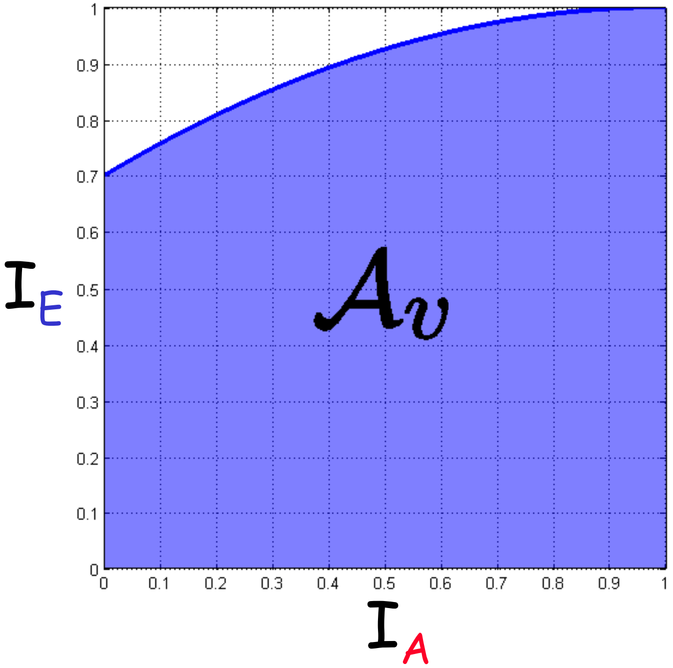
- We use statistical quantities, which are only correct if $N \to \infty$

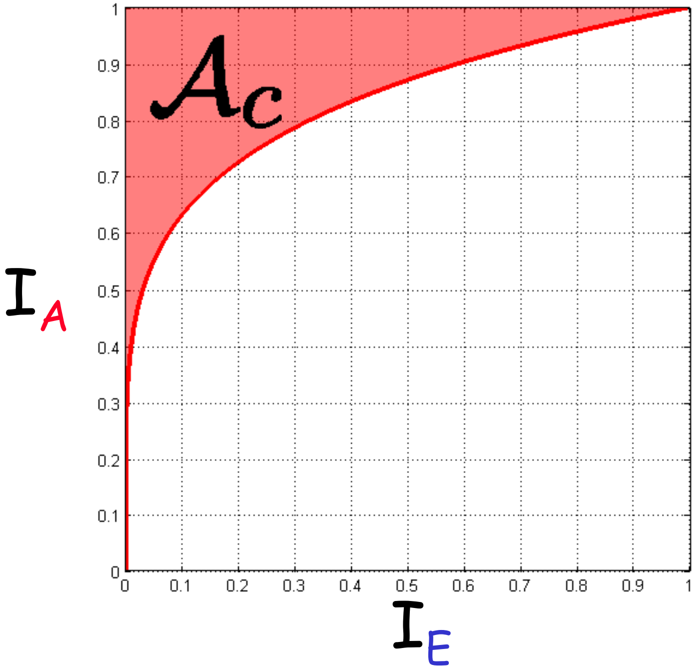
- We model extrinsic messages with an extrinsic channel. This can only be done exact for the BEC. The Gaussian assumption is an approximation.

Area of LDPC Component Codes

$$\mathcal{A}_v = 1 - \frac{1-C}{d_v}$$

$$\mathcal{A}_c = \frac{1}{d_c}$$





Necessary condition for successful decoding:

$$1 - \mathcal{A}_v < \mathcal{A}_c$$

Consequences of Area Property

$$1 - \mathcal{A}_v < \mathcal{A}_c$$

$$1 - 1 + \frac{1 - C}{d_v} < \frac{1}{d_c}$$

$$1 - C < \frac{d_v}{d_c}$$

$$C > 1 - \frac{d_v}{d_c} = R$$

"Surprising" result:

The area property tells us that the decoder can only converge if the rate is smaller than capacity!

More Consequences…

Suppose the condition for convergence is fulfilled

$$1 - \mathcal{A}_v = \gamma \cdot \mathcal{A}_c < \mathcal{A}_c \qquad 0 \leq \gamma < 1$$

$$\mathcal{A}_c = \frac{1}{d_c}$$

$$1 - \mathcal{A}_v = \gamma \cdot \mathcal{A}_c = \frac{1 - C}{d_v}$$

$$R = 1 - \frac{d_v}{d_c} = 1 - \frac{1 - C}{\gamma} = \frac{C - (1 - \gamma)}{\gamma} < C$$

What is the result of this inequality?

Area and Rate Loss

$$R = 1 - \frac{d_v}{d_c} = 1 - \frac{1 - C}{\gamma} = \frac{C - (1 - \gamma)}{\gamma} < C$$

If $\gamma \to 1$ we can transmit at rates that approach capacity.
If $\gamma < 1$ we are bounded from capacity.

$\gamma \to 1$ means that $1 - A_v = A_c$

Furthermore, the curves must not intersect.

$\Rightarrow$ The curves have to be matched.

Code design reduces to curve fitting!

Curve Fitting – Code Mixture

We only considered regular codes, where every symbol has the same properties. Therefore, averaging over all symbols is equivalent to the mutual information of an arbitrarily symbol.

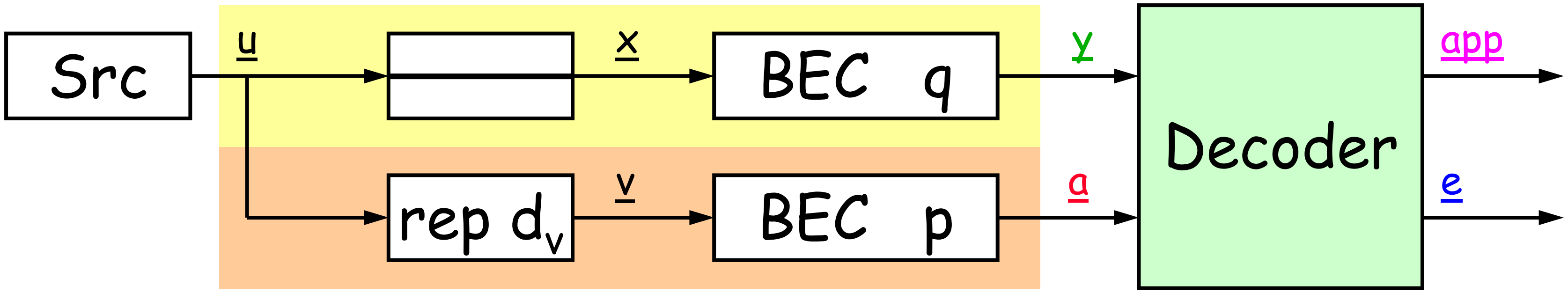
$$I_E = \frac{1}{m} \sum_{i=1}^{m} I(V_i; E_i) = I(V_1; E_1)$$

If we partition m into n_u groups $j=1 \ldots n_u$ each with length l_j, we can write I_E as

$$I_E = \sum_{j=1}^{n_u} \frac{l_j}{m} \left[\frac{1}{l_j} \sum_{i=1}^{l_j} I(V_{ji}; E_{ji}) \right] = \sum_{j=1}^{n_u} \gamma_j I_{E_j} \qquad \gamma_j = \frac{l_j}{m} = \frac{l_j}{\sum_{j=1}^{n_u} l_j}$$

The resulting EXIT function is the weighted average of the EXIT functions of the groups.

Example – Variable Mixture



70% of the variable nodes have d_v=2
30% of the variable nodes have d_v=5

$$\gamma_1 = \frac{0.7 \cdot k \cdot 2}{0.7 \cdot k \cdot 2 + 0.3 \cdot k \cdot 5} = 0.48 \qquad \gamma_2 = \frac{0.3 \cdot k \cdot 5}{0.7 \cdot k \cdot 2 + 0.3 \cdot k \cdot 5} = 0.52$$

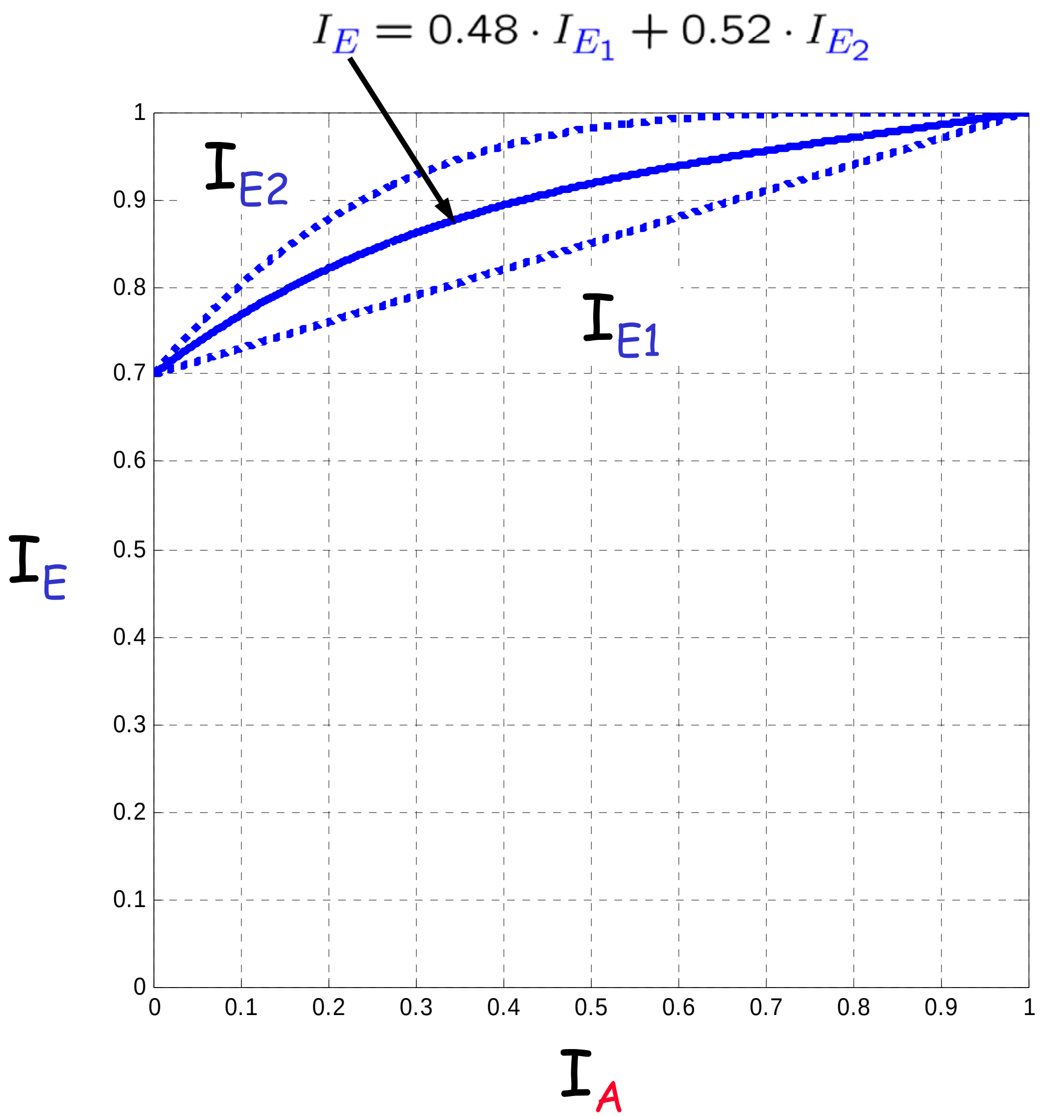
$$I_{E_j} = 1 - qp^{d_{vj}-1} \qquad I_E = \gamma_1 \cdot \left[1 - qp^{d_{v1}-1}\right] + \gamma_2 \cdot \left[1 - qp^{d_{v2}-1}\right]$$

$$I_E(p) = 1 - q \cdot \sum_{j=1}^{n_u} \gamma_j \cdot p^{d_{vj}-1}$$

This is a polynomial in p

Note that $\sum \gamma_j = 1$

Example – Variable Mixture



Curve Fitting

Lets fix the EXIT function of the check node decoder.

$$I_{Ec} = (I_{Ac})^{d_c - 1}$$

For curve fitting, we can exchange the following quantities

$$I_{Ec} = I_{Av} \qquad I_{Ev} = I_{Ac}$$

Therefore, we can write the EXIT function of the variable node decoder as the inverse EXIT function of the check node decoder.

$$I_{Av} = (I_{Ev})^{d_c - 1}$$

$$I_{Ev} = (I_{Av})^{\frac{1}{d_c - 1}} = (1 - p)^{\frac{1}{d_c - 1}}$$

Taylor Series Expansion

$$I_{Ev} = (I_{Av})^{\frac{1}{dc-1}} = (1-p)^{\frac{1}{dc-1}}$$

Assuming for example d_c=5 we can expand I_{Ev} as a Taylor series

$$I_{Ev} = 1 - \left[\frac{1}{4}p + \frac{3}{32}p^2 + \frac{7}{128}p^3 + \cdots\right]$$

Truncating the Taylor series and normalizing the coefficients to 1 results in

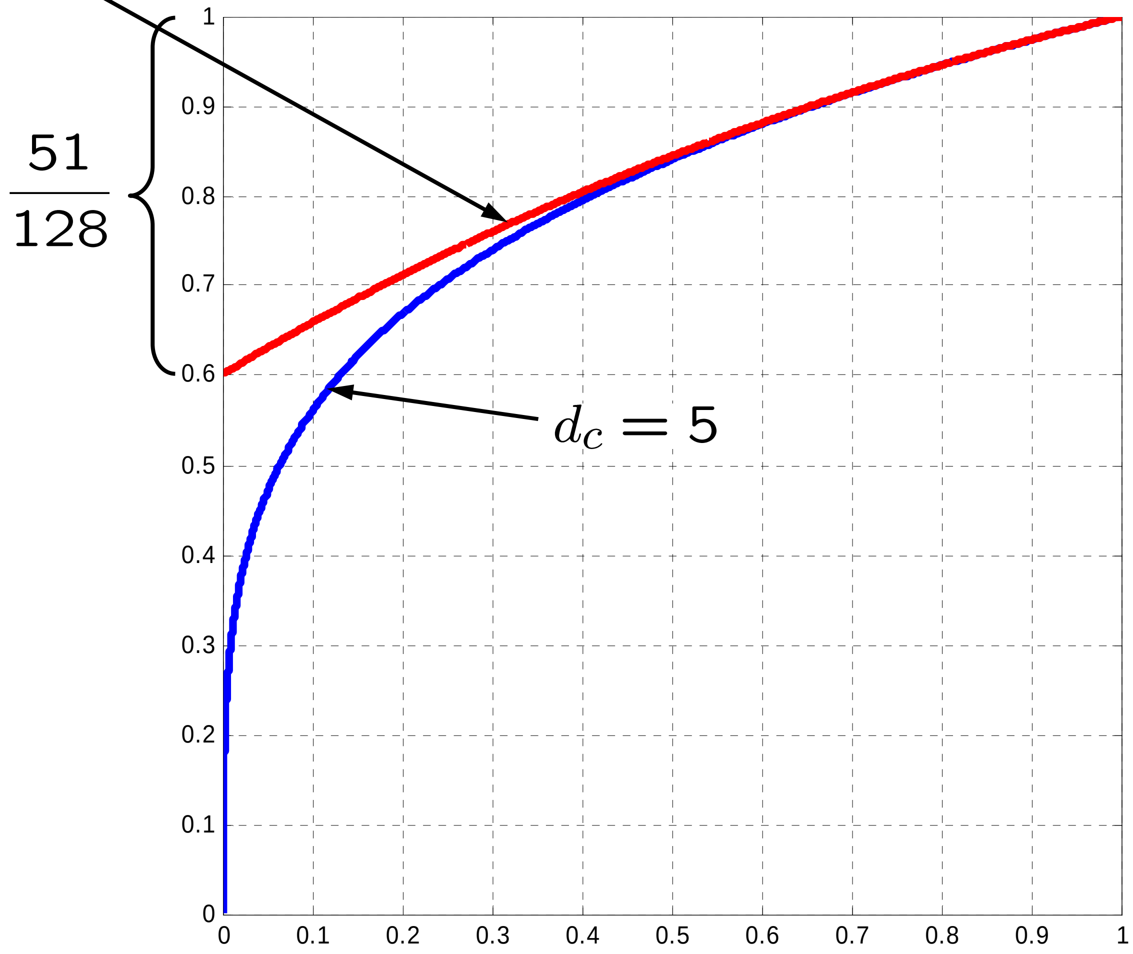
$$I_{Ev} = 1 - \frac{51}{128}\left[\frac{32}{51}p + \frac{12}{51}p^2 + \frac{7}{51}p^3\right]$$

Compare this with the transfer function of the mixture of variable nodes...

$$I_E(p) = 1 - q \cdot \sum_{j=1}^{n_u} \gamma_j \cdot p^{d_{vj}-1}$$

Curve Fitting

$$I_{Ev} = 1 - \frac{51}{128}\left[\frac{32}{51}(1 - I_{Av}) + \frac{12}{51}(1 - I_{Av})^2 + \frac{7}{51}(1 - I_{Av})^3\right]$$



Even more Consequences...

Using the same model as for the variable and check node decoder, it can be shown that the areas for a serial concatenated code with an outer code $R_{out}=k_{out}/n_{out}$ and an inner code $R_{in}=k_{in}/n_{in}$ are given by

$$\mathcal{A}_{out} = 1 - R_{out} \qquad\qquad \mathcal{A}_{in} = \frac{I(\underline{X};\underline{Y})}{n_{in} \cdot R_{in}}$$

The same necessary condition $1-\mathcal{A}_{out} < \mathcal{A}_{in}$ leads to

$$R_{out} \cdot R_{in} < \frac{I(\underline{X};\underline{Y})}{n_{in}} \leq C$$

If the inner code has rate < 1, i.e. $I(\underline{X};\underline{Y})/n_{in} < C$ then we can not achieve capacity with serial concatenated codes!