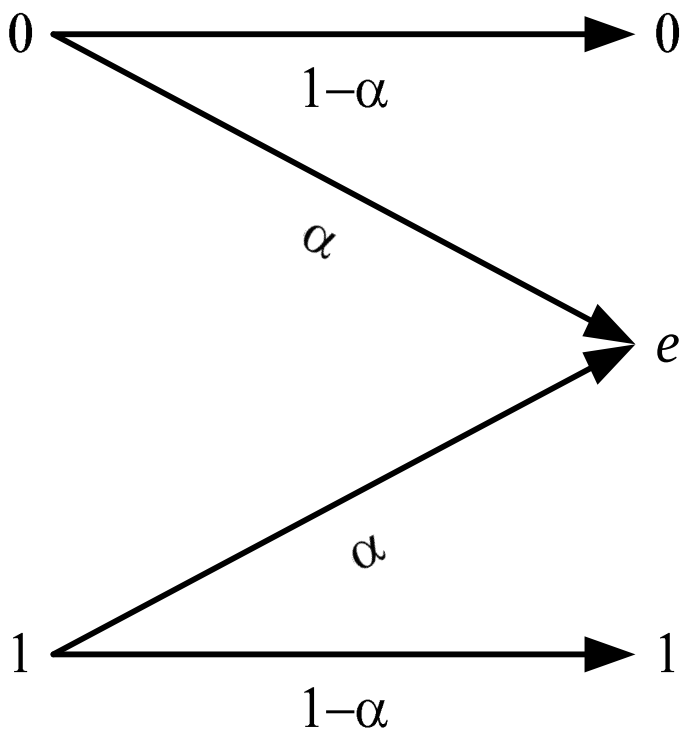
Fountain Codes

FOUNTAIN CODES

THE BINARY ERASURE CHANNEL

- Introduced by Elias in 1955 and regarded as a theoretical model
- Internet changed this notion 40 years later
- On the internet, due to router congestion and CRC errors, sent packets may not reach the destination
- This packet losses can be regarded as erasures

THE BINARY ERASURE CHANNEL



- e – erasure
- α – erasure probability
- Erasure channel Capacity: $C = 1-\alpha$
- Intuitive interpretation: since a proportion α of the bits are lost in the channel, we can recover (at most) a proportion $(1-\alpha)$ of the bits.

CLASSICAL SOLUTIONS

- When a packet did not reach the destination the receiver sends back a requests for retransmission
- Alternatively, the receiver can send back acknowledgement messages for each successfully received packet. The sender keeps track of the missing packets and retransmits them until all have been acknowledged

CLASSICAL SOLUTIONS

- Both solutions guarantees the correct delivery of all the packets regardless of the rate of packet losses
- However, if the rate of packet losses is high, both of these schemes are very inefficient
- The full capacity of the channel is not reached
- According to Shannon theory, the feedback channel is not necessary

BROADCAST CHANNEL WITH ERASURES

- On a broadcast channel with erasures, the repetition schemes are very inefficient, and can lead to network congestion
- An appropriate Forward Error Correction (FEC) Code should achieve the theoretic channel capacity without feedback channel
- With classical codes the design of the fixed rate $R=K/N$, should be performed to worst case conditions
- This restriction makes this coding inefficient also

Reed-Solomon codes for broadcast channels with erasures

- An (N,K) Reed-Solomon code correctly decode the K symbols of the message from K codeword symbols
- However, Reed-Solomon codes are only practical for small values of N and K
- The coding / decoding cost is of order

$$K(N-K)\log_2 N$$

VARIABLE RATE CODES

- If the error probability of a BEC varies, the ideal code should allow on the fly variable encoding rate $R=K/N$
- With Reed-Solomon codes it is not possible to change R on the fly
- Michael Luby (2002) invented a *rateless* code with this propriety

FOUNTAIN CODES

- This code can generate a potentially infinite number of codewords
- Fountain codes are near optimal for every erasure channel, despite the probability of erasure α
- The message m with K symbols can be decoded from K' received codewords, with K' a little larger than K

FOUNTAIN CODES

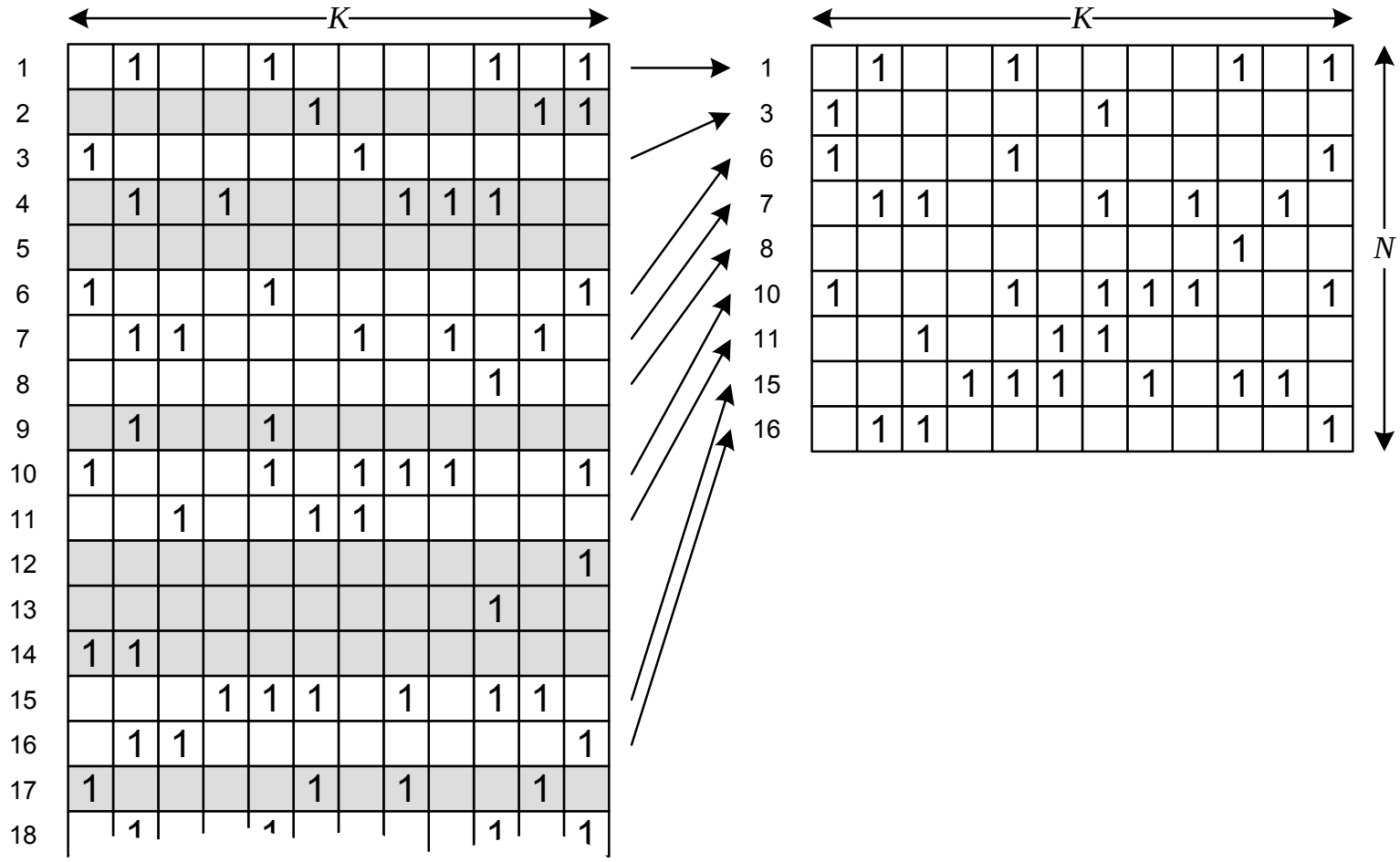
- Consider a message m with K symbols

$$m = \{m_1, m_2, \ldots, m_K\}$$

- To generate the n^{th} codeword symbol the encoder chooses the number d of symbols to combine from a degree distribution ρ
- Then the encoder chooses d symbols at random from m and perform the xor sum
- Encoder generates k random bits $\{G_{nk}\}$ and the transmitted packet cn is set to the bitwise sum,mod2,of the source packets for which $\{G_{nk}\}$ is 1.

$$c_n = \sum_{k=1}^{K} m_k G_{nk}$$

FOUNTAIN CODES



FOUNTAIN CODES

- If $N<K$ the decoder does not have enough codeword symbols to recover the original information
- If $N \geq K$ and G has an $K \times K$ submatrix with inverse, then the receiver can recover the original information. It is possible to use Gaussian elimination and recover the message

$$m_k = \sum_{n=1}^{N} c_n G_{kn}^{-1}$$

FOUNTAIN CODES

- If it is possible to find an invertible $K \times K$ submatrix in the received $N \times K$ matrix, then the solution is unique
- As the matrix is generated at random and we can not predict the columns that we are going to received, the question is:

What is the probability of a $K \times K$ random binary matrix being invertible?

RANDOM MATRICES

- Linear independency
- A set of K vectors v_n in some vector space of dim K is linearly independent if

$$\sum_{n=1}^{K} \alpha_n v_n = 0$$

only with all the α_n=0

Probability of a $K \times K$ random binary matrix G being invertible

- If we have only one vector the probability of being linear independent is the probability of being different from zero

$$\left(1-2^{-K}\right)$$

- With two vectors we have the probability of the second vector being different from zero and different from the first one

$$\left(1-2^{-(K-1)}\right)$$

- For K vectors the probability of all vectors being linear independent

$$\left(1-2^{-K}\right)\times\left(1-2^{-(K-1)}\right)\times\cdots\times\left(1-\frac{1}{8}\right)\times\left(1-\frac{1}{4}\right)\times\left(1-\frac{1}{2}\right)=0.289\cdots$$

Probability of a $K \times K$ random binary matrix G being invertible

- If the number N of vectors is greater than K, with $E=N-K$ (excess), what is the probability $(1-\delta)$ that there is an invertible $K \times K$ submatrix in G?

$$\delta(E) \leq 2^{-E}$$

- Where δ is probability of failure and E is the number of redundant packets

Probability of a random binary matrix G being invertible

- The number of packets $N=K+E$ in order to have (on average) a guarantee of decoding of $(1-\delta)$ is

$$\approx K + \log_2 1/\delta$$

- So, an excess of E packets increases the probability of success to at least

$$\left(1-\delta\right) = \left(1-2^{-E}\right)$$

COMPUTATIONAL COST

- The encoding cost is $K/2$ symbol operations by codeword
- The decoding cost has two components
  - The matrix inversion with K^3 operations by Gaussian elimination
  - The application of matrix inverse to the received symbols which costs $K^2/2$
- When the value of K increases, random linear fountain codes approximate to the Shannon limit
- Problem to solve: find a coding and decoding technique with lower cost, preferably linear

SPARSE RANDOM MATRICES

- The coding and decoding computational cost can be reduced if the coding matrix G is sparse
- Even for matrices with a small average number of ones per row is possible to find an invertible coding matrix

BALLS AND BINS

- Suppose that we throw N balls to K bins at random
- **Question:** After throwing $N=K$ balls what fraction of the bins is empty?
- Answer: The probability that a ball hits one of the K bins is $1/K$. The complement is $(1\text{-}1/K)$, and the probability that a bin is empty after N balls is

$$\left(1-\frac{1}{K}\right)^N \approx e^{-N/K}$$

- For $N=K$ the probability of a certain bin is empty is $1/e$ and the fraction of empty bins would be $1/e$ also

BALLS AND BINS

- After throwing N balls the expected number of empty bins is

$$Ke^{-N/K}$$

- This expected number δ of empty bins is small for large N. So we can say that the probability of all bins have a ball is given by $(1-\delta)$ only if

$$N > K \log_e \frac{K}{\delta}$$

THE LT CODE

Encoder

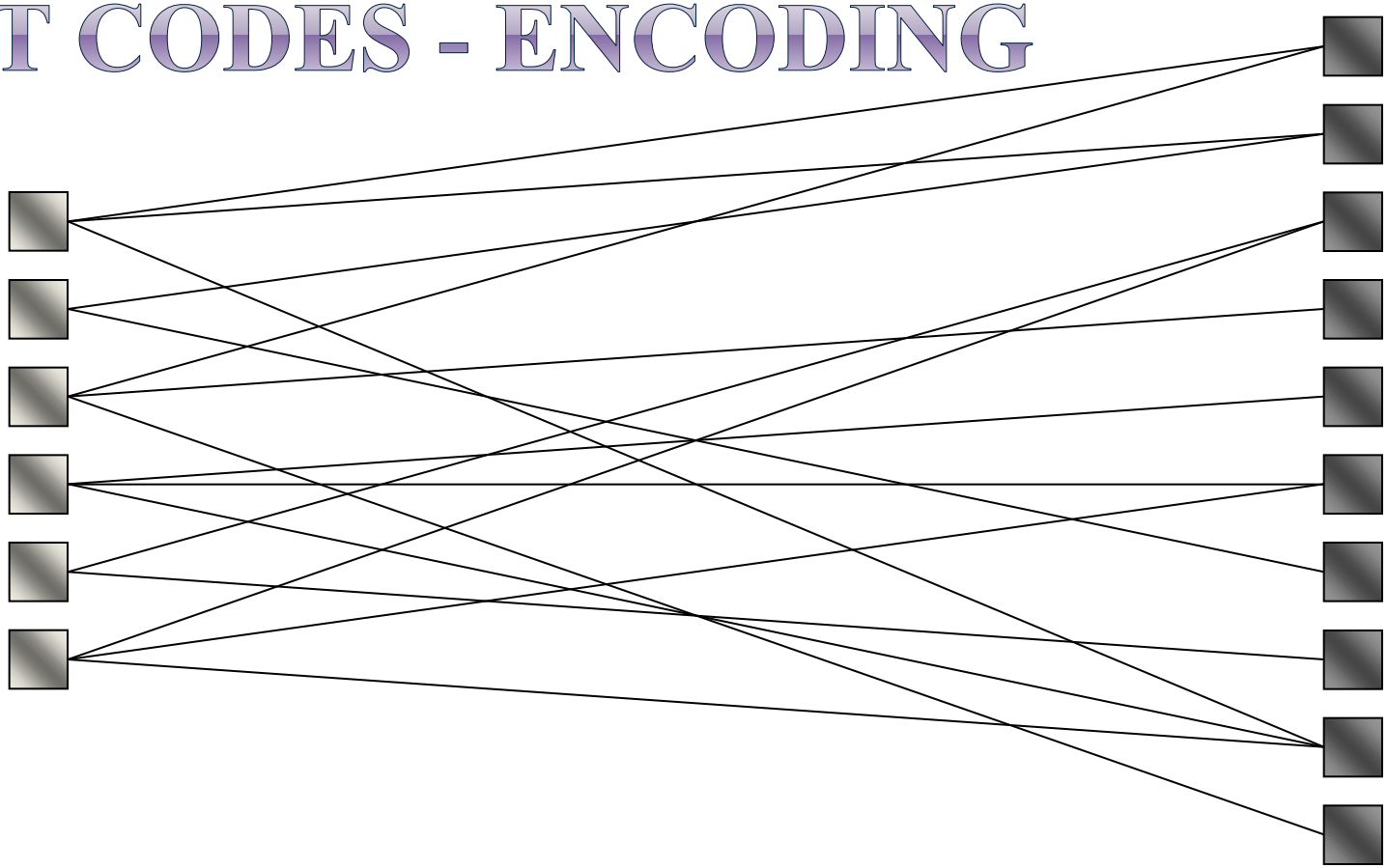
- Consider a message m with K elements

$$m = \{m_1, m_2, m_3, \ldots, m_K\}$$

1. Choose at random the degree d_n of the codeword from a degree distribution $\rho(d)$.
2. Choose at random and uniformly, d_n distinct input symbols and sum them using the XOR operation.

- This encoding defines a sparse and irregular encoding matrix

LT CODES - ENCODING



d	2	2	2	1	1	2	1	1	3	1
v	(101000)	(110000)	(000011)	(001000)	(000100)	(000101)	(010000)	(000010)	(100101)	(001000)

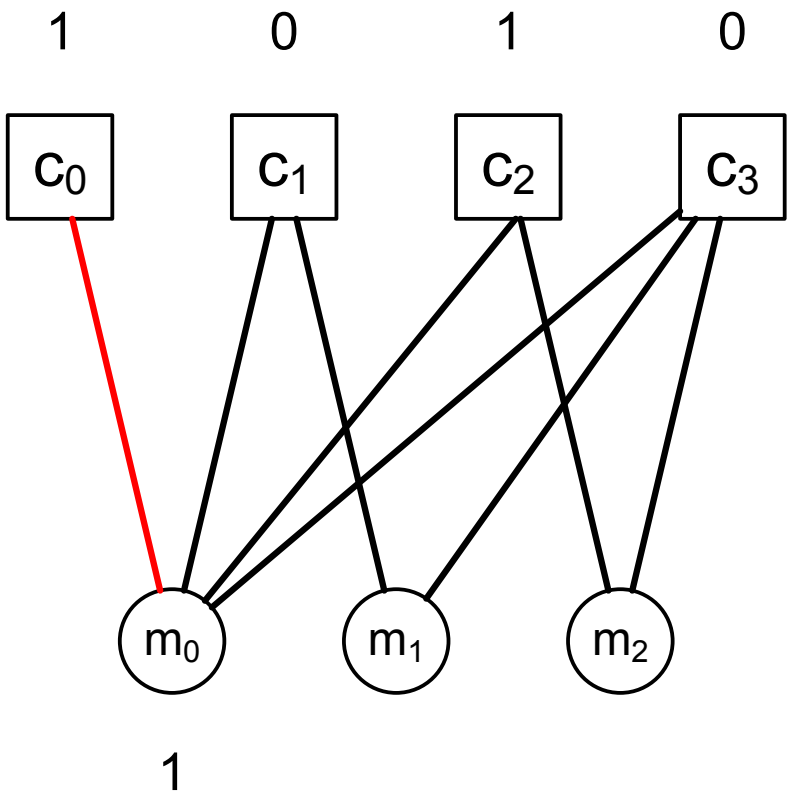
THE LT CODE

Decoder

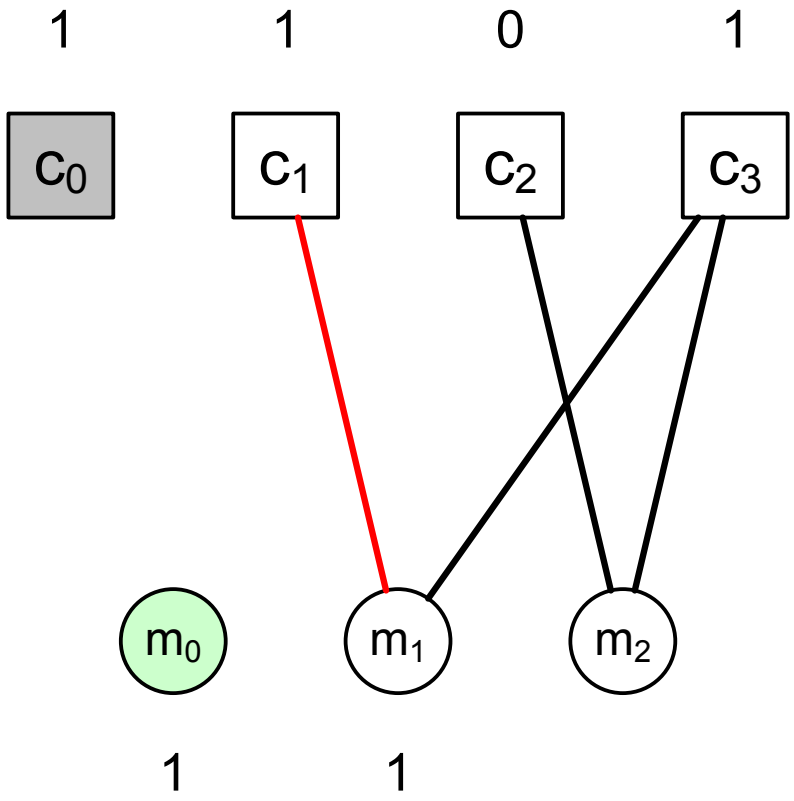
- The decoder must recover m form $c=Gm$ supposing G known
- If some of the codeword symbols are equal to one of the message symbols, then it is possible to decode by the following algorithm
  1. Find a codeword c_n with degree one. If it is not possible to find one halt and report fail
  2. Set $m_i=c_n$
  3. Add m_i (with XOR) to all codewords c_n that are connected to m_i
  4. Remove all the edges connected to m_i
  5. Repeat 1 to 4 until all m_i are decoded

DECODING EXAMPLE - 1

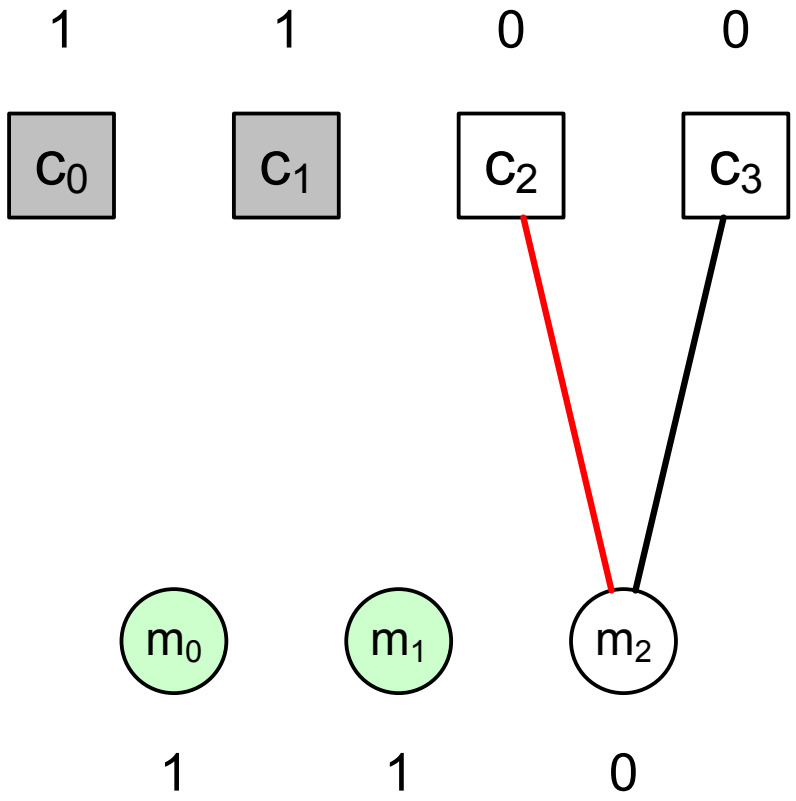
$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \end{bmatrix}$$



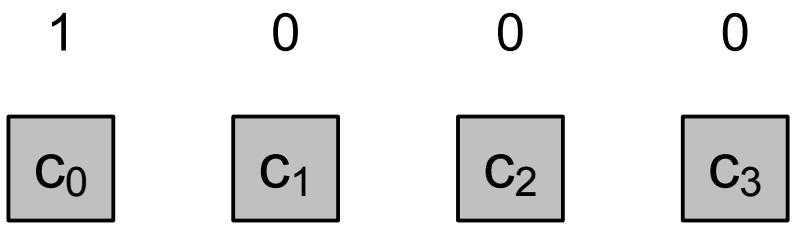
DECODING EXAMPLE – 2

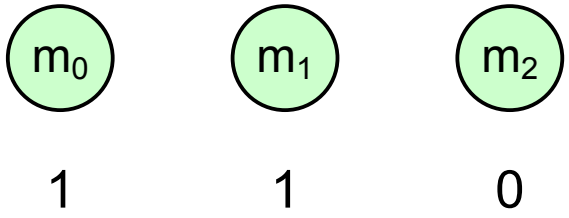


DECODING EXAMPLE – 3



DECODING EXAMPLE - 4





THE DEGREE DISTRIBUTION

- Each codeword is a linear combination of d symbols from the message m
- The degree d is chosen at random from a degree distribution $\rho(d)$
- There are two design conflicts:
  - The degree of some codewords should be high to guarantee that all the message symbols are covered
  - The degree of some codewords should be low in order to start the decoding process and keep going

SOLITON DISTRIBUTION

- Can we design a degree distribution that guarantees the optimal Shannon limit of decoding the K symbols of the message after K received codewords?
- We want a distribution that on average guarantees that just one message symbol is uncovered at each iteration
- Such a distribution is the Soliton

SOLITON DISTRIBUTION

- Step 0
  - The expected number of codeword symbols of degree one at step zero should be 1
- Step 1
  - One of the message symbols is decoded and it lower the degree of some of the codeword symbols.
  - At the end of step 1, at most one degree 2 codeword should be connected to the decoded message symbol in order to decrease its degree to one and the process continues
- Step n
  - Continue the process checking at each step that one of the codeword symbols has degree one

SOLITON DISTRIBUTION

$$\rho_1 = 1/K$$

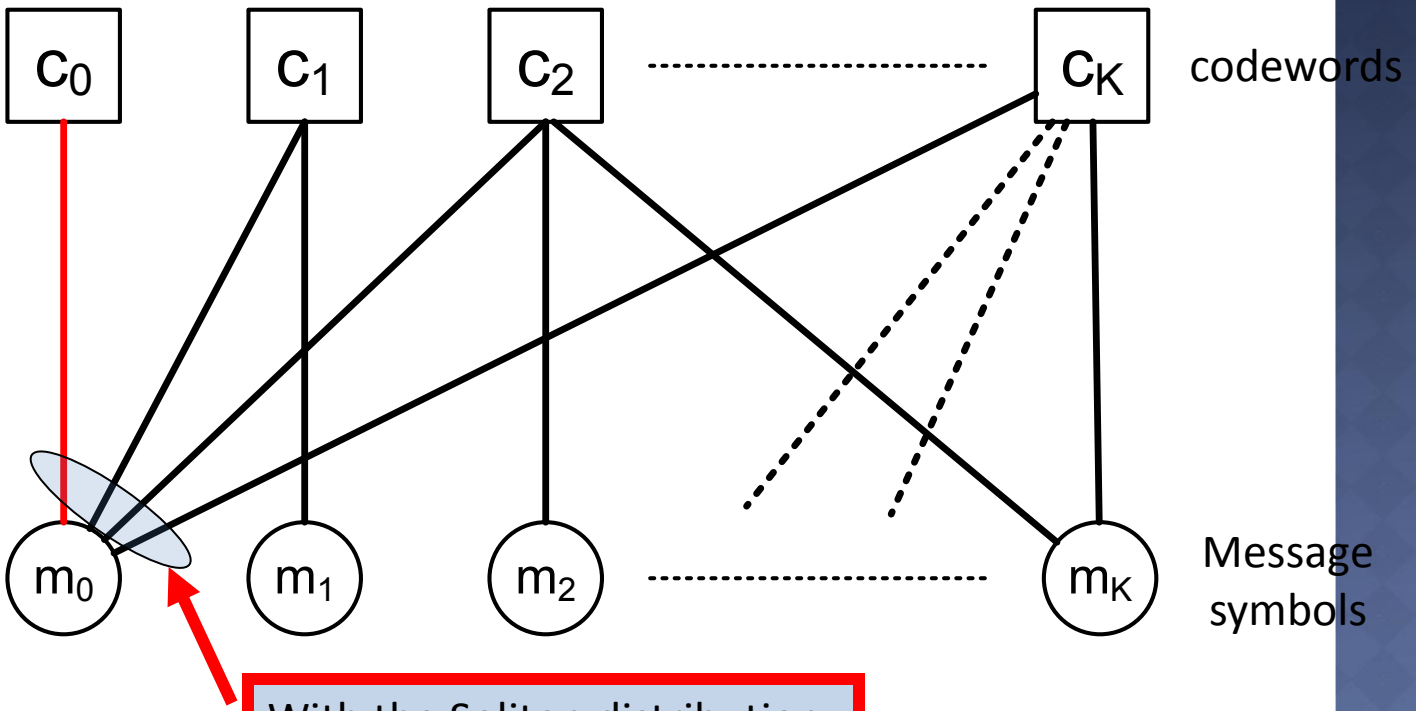
$$\rho_d = \frac{1}{d(d-1)} \qquad \text{for} \, d = 2,3,\ldots,K$$

$$\rho_d = \left\{ \frac{1}{K}, \frac{1}{2}, \frac{1}{6}, \frac{1}{12}, \cdots, \frac{1}{K(K-1)} \right\}$$

The mean degree of this distribution is

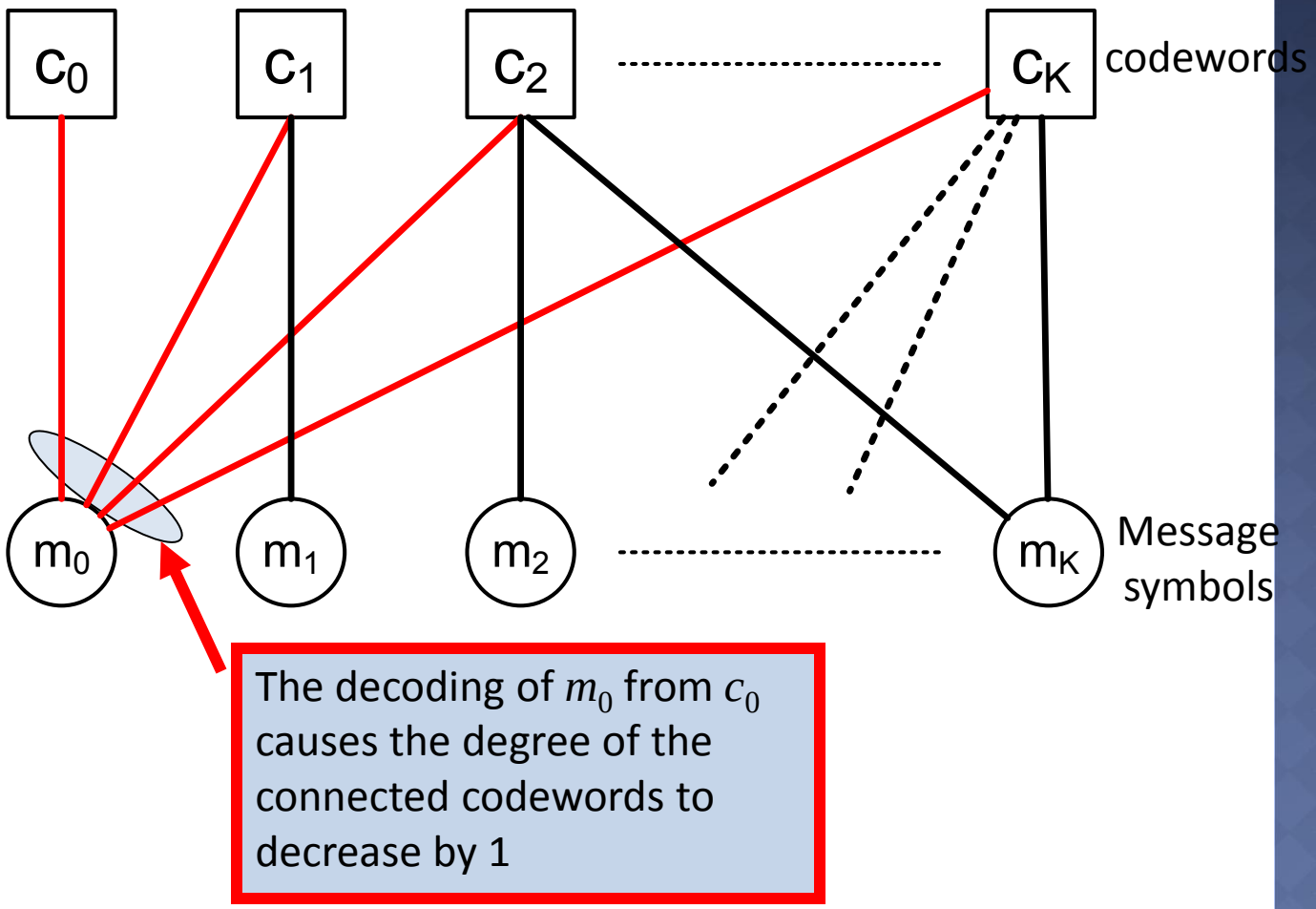
$$\bar{\rho} = \sum_{d=1}^{K} d\rho_d = \sum_{d=1}^{K} \frac{1}{d-1} \approx \log_e K$$

SOLITON DISTRIBUTION



With the Soliton distribution the expected number of edges from each message symbol will be $\log_e K$

SOLITON DISTRIBUTION



SOLITON DISTRIBUTION

- Let $h_t(d)$ be the expected number of codewords of degree d after the t^{th} iteration of the algorithm
- **Step 0** $h_0(d) = K\rho_d$

$$h_0(1) = K\rho_1 = K\frac{1}{K} = 1$$

- **Step 1** $h_1(1) = h_0(2)\frac{2}{K} = K\rho_2\frac{2}{K} = K\frac{1}{2}\frac{2}{K} = 1$

$$h_1(d) = h_0(d)\left(1 - \frac{d}{K}\right) + h_0(d+1)\frac{d+1}{K}$$

Expected number of codewords with degree d+1 that reduced their degree after step 0

Expected number of codewords with degree d that maintained their degree after step 0

Probability of a degree d codeword had an edge to a message symbol

SOLITON DISTRIBUTION

- Step 1 (cont.)

$$h_1(2) = h_0(2)\left(1 - \frac{2}{K}\right) + h_0(2+1)\frac{2+1}{K}$$

$$h_1(2) = K\rho_2\left(1 - \frac{2}{K}\right) + K\rho_3\frac{3}{K}$$

$$h_1(2) = K\frac{1}{2}\left(1 - \frac{2}{K}\right) + K\frac{1}{6}\frac{3}{K} = \frac{K-1}{2}$$

SOLITON DISTRIBUTION

- Step 2

$$h_2(1) = h_1(2)\frac{2}{K-1} = \frac{K-1}{2}\frac{2}{K-1} = 1$$

$$h_2(d) = h_1(d)\left(1 - \frac{d}{K-1}\right) + h_1(d+1)\frac{d+1}{K-1}$$

- We have showed (for the first 3 steps) that the expected number of degree 1 codeword symbols at each step will be 1 if we use the Soliton distribution.

SOLITON DISTRIBUTION

- **Theorem:** Suppose that the expected degree distribution holds after t-1 iterations, for all t. Then, $h_t(d)$ satisfies the two conditions

$$h_t(1) = 1$$

$$h_t(d) = \frac{K - t}{d(d-1)} \qquad d > 1$$

ROBUST SOLITON

- Due to the random fluctuations around the mean behaviour, the Soliton distribution behaves poorly in practice. If in one of the steps, there is not a degree one codeword, the decoding process stops
- The Robust Soliton distribution tries to solve this problem by introducing two new parameters, c and δ, to obtain a expected number of degree one codeword symbols at each step of

$$\frac{S}{K} = \frac{c \log_e (K/\delta) \sqrt{K}}{K}$$

instead of $1/K$

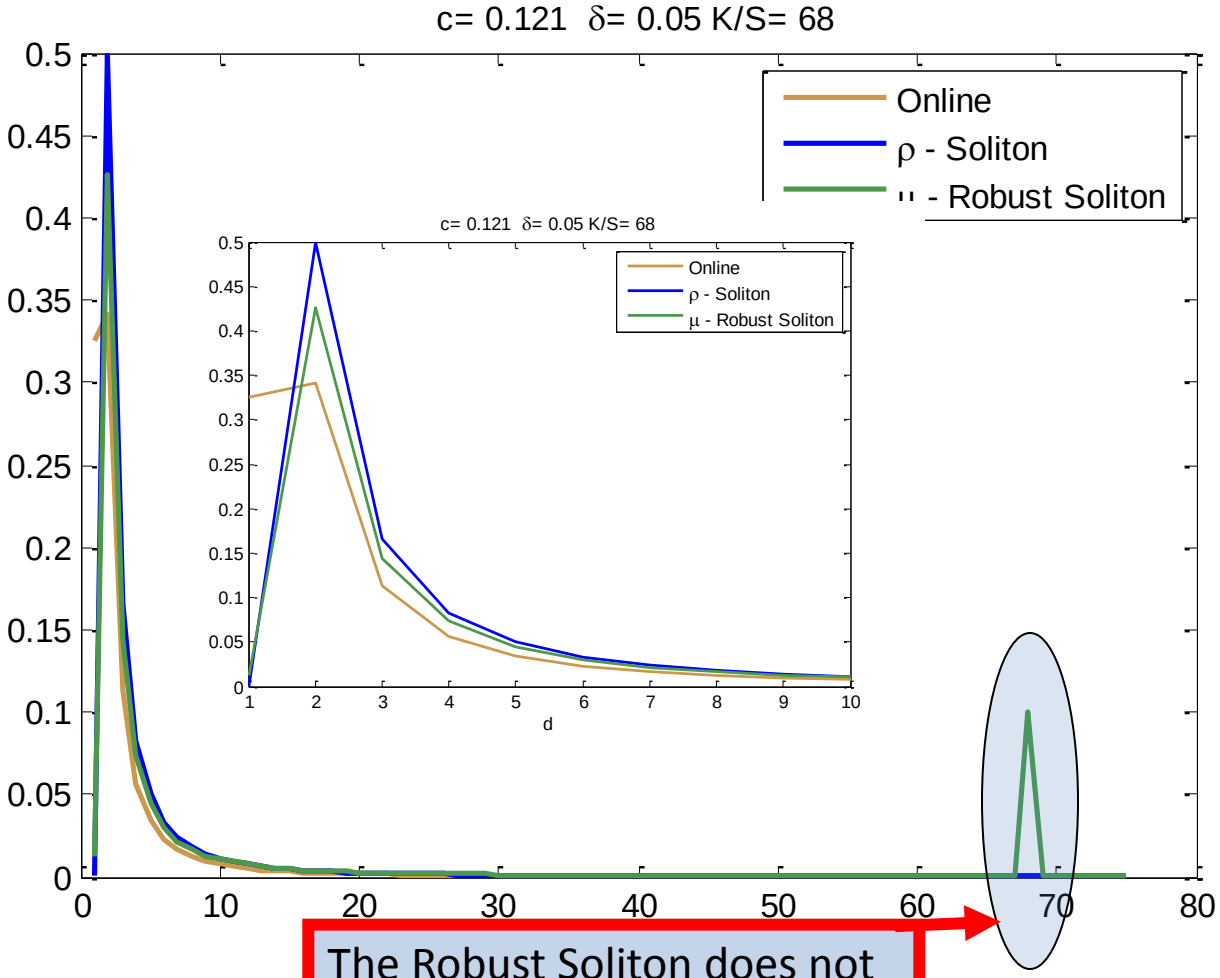
ROBUST SOLITON

- Luby proved that there exists a value of c and δ, given N received codeword symbols the algorithm recover the K message symbols with probability $(1\text{-}\delta)$

$$S = c \log_e(K/\delta)\sqrt{K}$$

$$N = K + 2\log_e(S/\delta)S$$

COMPARING THE DISTRIBUTIONS



The Robust Soliton does not have codewords of degree larger than K/S

ANY QUESTIONS

THANK YOU FOR LISTENING