CT216 Introduction to Communication Systems
Lecture 3: Channel Coding

Yash M. Vasavada

Professor, DA-IICT, Gandhinagar

09/02/2024

Overview of Today's Talk

1. Block Diagrams

Overview of Today's Talk

1. Block Diagrams
2. Channel Coding

Overview of Today's Talk

1. Block Diagrams
2. Channel Coding
3. Practical Channel Coding
   - Hamming Distance
   - Several FEC Schemes

Overview of Today's Talk

1. Block Diagrams
2. Channel Coding
3. Practical Channel Coding
   - Hamming Distance
   - Several FEC Schemes
4. Iterative Decoding

Overview of Today's Talk

1. Block Diagrams
2. Channel Coding
3. Practical Channel Coding
   - Hamming Distance
   - Several FEC Schemes
4. Iterative Decoding
5. LDPC Codes

Overview of Today's Talk

1. Block Diagrams
2. Channel Coding
3. Practical Channel Coding
   - Hamming Distance
   - Several FEC Schemes
4. Iterative Decoding
5. LDPC Codes
6. A Visual Explanation

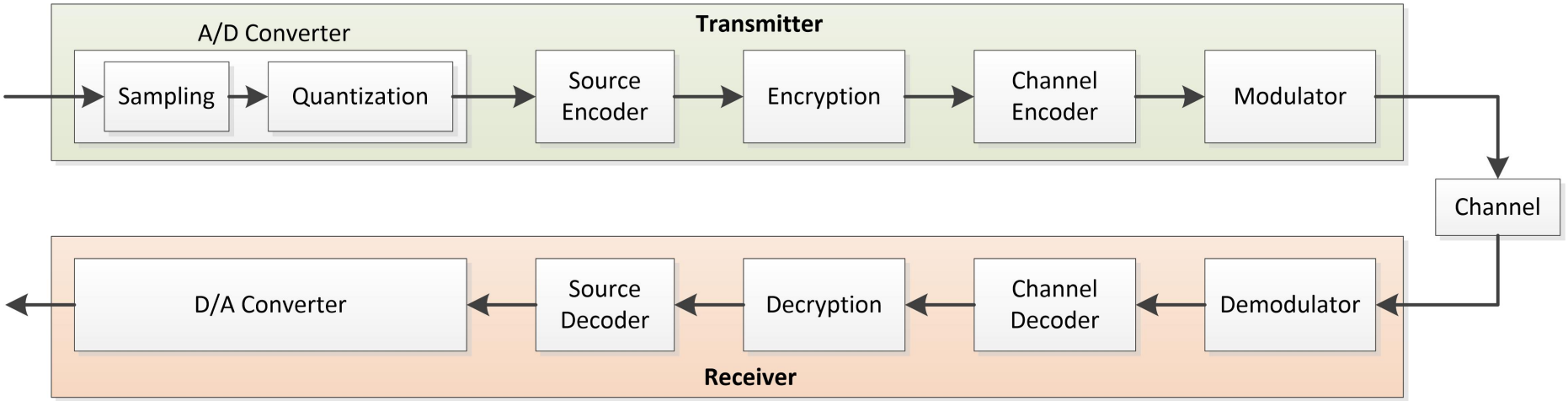
Overview of Today's Talk

1. Block Diagrams
2. Channel Coding
3. Practical Channel Coding
   - Hamming Distance
   - Several FEC Schemes
4. Iterative Decoding
5. LDPC Codes
6. A Visual Explanation
7. Linear Algebraic View

Digital Communication Transceiver

Block Diagram

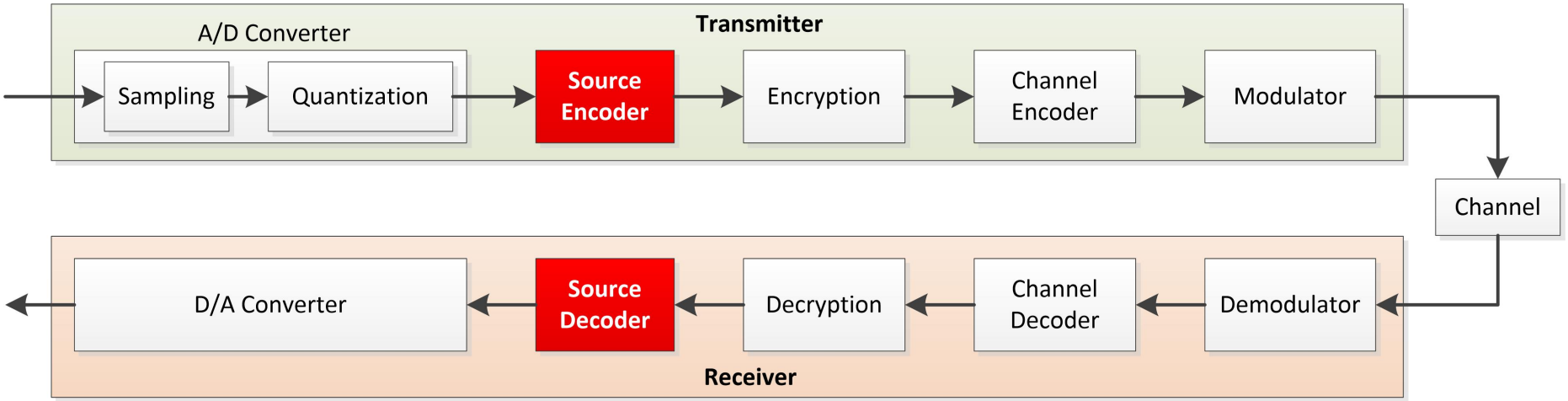
- A conceptual block diagram model of a digital communication transceiver



Digital Communication Transceiver

Block Diagram

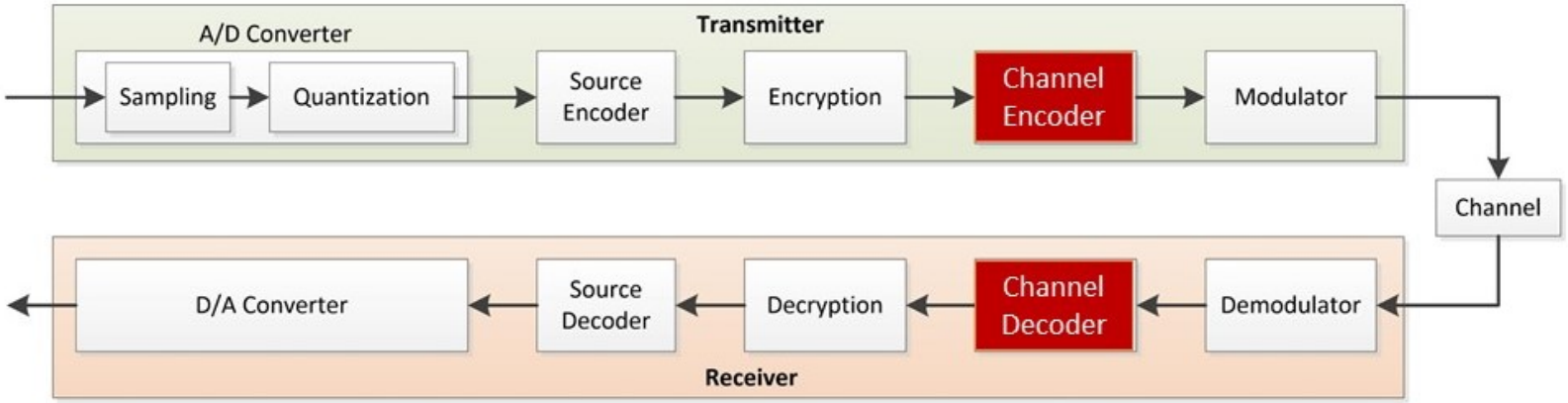
- We will be studying the mathematics and algorithms of source encoding



Digital Communication Transceiver

Channel Coding

- For now, we focus on channel encoding/decoding, also known as FEC (forward error correction) coding



Channel Coding

also known as Forward Error Correction (FEC) Coding

- Purpose: to turn a noisy channel, which can introduce errors in the transmitted symbols, to (ideally) a noiseless channel
  - The motivation for this is clear — if the channel coding works in this ideal manner, you will not hear disturbances when you are talking on the phone

Channel Coding

also known as Forward Error Correction (FEC) Coding

- Purpose: to turn a noisy channel, which can introduce errors in the transmitted symbols, to (ideally) a noiseless channel
  - The motivation for this is clear — if the channel coding works in this ideal manner, you will not hear disturbances when you are talking on the phone
  - One of the 5G service targets is remote surgery; when the surgeon is operating an instrument remotely, using a communication link, that link better be ultra-reliable, i.e., it should be as noiseless as possible.

Channel Coding

also known as Forward Error Correction (FEC) Coding

- Purpose: to turn a noisy channel, which can introduce errors in the transmitted symbols, to (ideally) a noiseless channel
  - The motivation for this is clear — if the channel coding works in this ideal manner, you will not hear disturbances when you are talking on the phone
  - One of the 5G service targets is remote surgery; when the surgeon is operating an instrument remotely, using a communication link, that link better be ultra-reliable, i.e., it should be as noiseless as possible.
- The real-world channels are always noisy, and the task of the channel coding is to turn the real-world channel into a channel as close to ideal as possible

Channel Coding

also known as Forward Error Correction (FEC) Coding

- Purpose: to turn a noisy channel, which can introduce errors in the transmitted symbols, to (ideally) a noiseless channel
  - The motivation for this is clear — if the channel coding works in this ideal manner, you will not hear disturbances when you are talking on the phone
  - One of the 5G service targets is remote surgery; when the surgeon is operating an instrument remotely, using a communication link, that link better be ultra-reliable, i.e., it should be as noiseless as possible.
- The real-world channels are always noisy, and the task of the channel coding is to turn the real-world channel into a channel as close to ideal as possible
  - The channel's "noisiness" is measured always relatively, i.e., as the received signal power P_s to the received noise power ratio, i.e., the SNR P_S/P_n

Channel Coding

also known as Forward Error Correction (FEC) Coding

- Purpose: to turn a noisy channel, which can introduce errors in the transmitted symbols, to (ideally) a noiseless channel
  - The motivation for this is clear — if the channel coding works in this ideal manner, you will not hear disturbances when you are talking on the phone
  - One of the 5G service targets is remote surgery; when the surgeon is operating an instrument remotely, using a communication link, that link better be ultra-reliable, i.e., it should be as noiseless as possible.
- The real-world channels are always noisy, and the task of the channel coding is to turn the real-world channel into a channel as close to ideal as possible
  - The channel's "noisiness" is measured always relatively, i.e., as the received signal power P_s to the received noise power ratio, i.e., the SNR P_S/P_n
  - The channel coding turns the actual physical communication channel with a bad SNR into an "encoded channel" whose performance is equivalent to an actual channel with a good SNR

Channel Coding

also known as Forward Error Correction (FEC) Coding

- At first, it appears that the task is not possible — the channel already has introduced the noise, how can its effect be undone?

Channel Coding

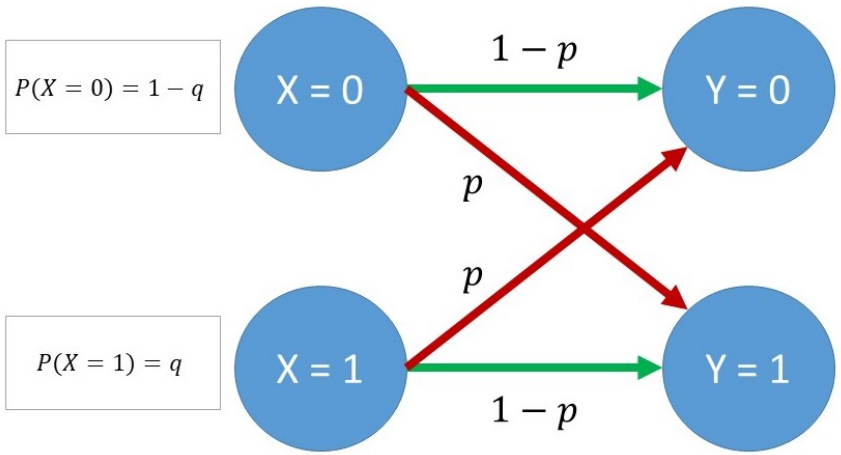
also known as Forward Error Correction (FEC) Coding

- At first, it appears that the task is not possible — the channel already has introduced the noise, how can its effect be undone?
- To answer this, we go back to the cause $\rightarrow$ effect model of the Bayesian belief model
- The fundamental reason why the receiver makes error in the presence of a noisy channel is that when it observes a particular channel output, $Y = 0$ or $Y = 1$, it cannot decide with certainty whether the cause is $X = 0$ or $X = 1$

Channel Coding

also known as Forward Error Correction (FEC) Coding

- At first, it appears that the task is not possible — the channel already has introduced the noise, how can its effect be undone?
- To answer this, we go back to the cause $\rightarrow$ effect model of the Bayesian belief model
- The fundamental reason why the receiver makes error in the presence of a noisy channel is that when it observes a particular channel output, $Y = 0$ or $Y = 1$, it cannot decide with certainty whether the cause is $X = 0$ or $X = 1$
  - ▷ the odds in favor of $X = 1$ are typically neither ∞ nor 0



Channel Coding

The Main Idea

- The art of channel coding is to turn the Bayesian diagram into one that looks like this:

Channel Coding

The Main Idea

- The art of channel coding is to turn the Bayesian diagram into one that looks like this:
- The cardinality of the transmitted message set is maintained the same (two, in this case), but the cardinality of the received message set is, somehow, expanded (to four, in this case)

Channel Coding

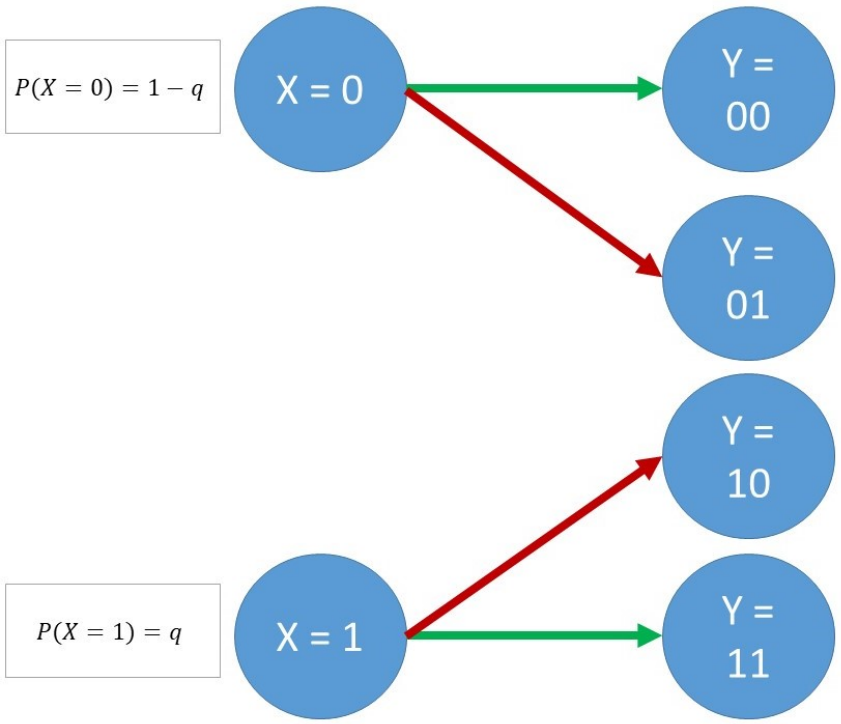
The Main Idea

- The art of channel coding is to turn the Bayesian diagram into one that looks like this:
- The cardinality of the transmitted message set is maintained the same (two, in this case), but the cardinality of the received message set is, somehow, expanded (to four, in this case)
- Furthermore, it is ensured that the expanded set is divided into two non-overlapping subsets, one each for each of the two transmitted symbols

Channel Coding

The Main Idea

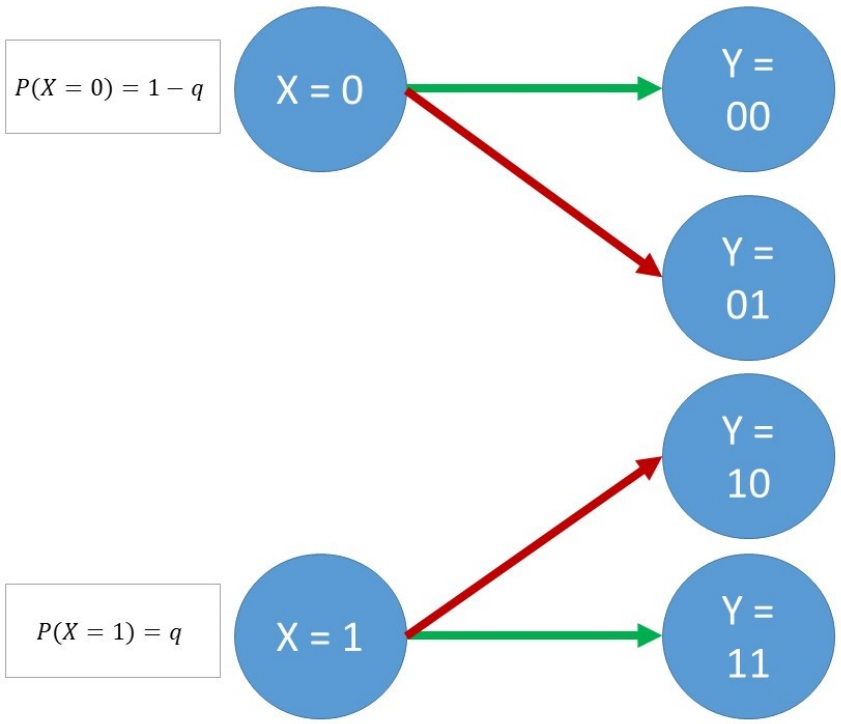
- The art of channel coding is to turn the Bayesian diagram into one that looks like this:
- The cardinality of the transmitted message set is maintained the same (two, in this case), but the cardinality of the received message set is, somehow, expanded (to four, in this case)
- Furthermore, it is ensured that the expanded set is divided into two non-overlapping subsets, one each for each of the two transmitted symbols



Channel Coding

The Main Idea

- The art of channel coding is to turn the Bayesian diagram into one that looks like this:
- The cardinality of the transmitted message set is maintained the same (two, in this case), but the cardinality of the received message set is, somehow, expanded (to four, in this case)
- Furthermore, it is ensured that the expanded set is divided into two non-overlapping subsets, one each for each of the two transmitted symbols



- Note that the receiver knows with certainty whether $X = 0$ or $X = 1$ for any one of the four possible received Y values

Channel Coding: A Basic Scheme

Repetition Code

- Encoding scheme:
  1. Take one bit at a time as the input.
  2. Repeat this bit $n-1$ times
  3. Transmit the resultant n bit long codeword
- Rate of this code is $r = \frac{1}{n}$.
  - $\rightarrow$ Rate r is the ratio of the number of information bits to total number of encoded bits

Channel Coding: A Basic Scheme

Repetition Code

Information bit sequence

0	1	1	1	0	1

Rate 1/3 repetition encoded bit sequence

0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1

BSC

Received bit sequence at the output of BSC

0	0	1	1	0	1	0	1	1	1	1	1	1	1	0	1	1	0

Channel Coding: A Basic Scheme

Repetition Code

What would be a good decoding strategy for this code?

Channel Coding: A Basic Scheme

Decoding of Repetition Code

Information bit sequence

0	1	1	1	0	1

Rate 1/3 repetition encoded bit sequence

0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1

BSC

Received bit sequence at the output of BSC

0	0	1	1	0	1	0	1	1	1	1	1	1	1	0	1	1	0

Decoded information bit sequence

0	1	1	1	1	1

Channel Coding: A Basic Scheme

Decoding of Repetition Code

- The decoding algorithm is called the majority-vote decoding
  - Take $n = 3$ bit block at a time, and decode it as that bit (either 0 or 1) that occurs the majority of times
  - To avoid the confusion in decoding, it maybe preferred to make n an odd number

Channel Coding

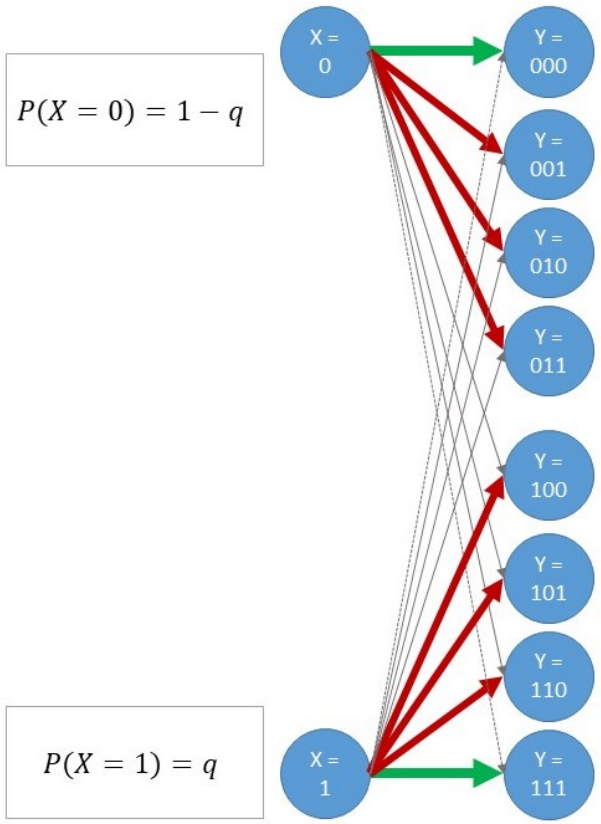
A Basic Scheme

- A view of repeat by n coding

Channel Coding

A Basic Scheme

- A view of repeat by n coding
  - Shown here for $n = 3$
  - Although the two sets of size 4 are created at the receiver, the channel does not become completely noiseless since there is a possibility that the receiver sees a message belonging to the set for $X = 1$ when $X = 0$ is transmitted



Channel Coding

Repeat-by-n Scheme

- There is a price that we pay with this repeat-by-$n = 3$ coding scheme

Channel Coding

Repeat-by-n Scheme

- There is a price that we pay with this repeat-by-$n = 3$ coding scheme
- We could have transmitted three independent bits of information; instead we are using the channel three times (to send three encoded bits) but sending actually only one bit of information

Channel Coding

Repeat-by-n Scheme

- There is a price that we pay with this repeat-by-$n = 3$ coding scheme
- We could have transmitted three independent bits of information; instead we are using the channel three times (to send three encoded bits) but sending actually only one bit of information
- Define as $r = K/N$ the <u>rate</u> of the channel coding scheme:

Channel Coding

Repeat-by-n Scheme

- There is a price that we pay with this repeat-by-$n = 3$ coding scheme
- We could have transmitted three independent bits of information; instead we are using the channel three times (to send three encoded bits) but sending actually only one bit of information
- Define as $r = K/N$ the rate of the channel coding scheme:
  - Here, here K is the number of information bits and $N \geq K$ is the number of encoded bits
  - $0 \leq r \leq 1$

Channel Coding

Repeat-by-n Scheme

- There is a price that we pay with this repeat-by-$n = 3$ coding scheme
- We could have transmitted three independent bits of information; instead we are using the channel three times (to send three encoded bits) but sending actually only one bit of information
- Define as $r = K/N$ the rate of the channel coding scheme:
  - Here, here K is the number of information bits and $N \geq K$ is the number of encoded bits
  - $0 \leq r \leq 1$
- For the repeat-by-n coding, $K = 1$ information bit is repeated $N = n$ times, i.e., the rate $r = 1/n$

Channel Coding

Repeat-by-n Scheme

- There is a price that we pay with this repeat-by-$n = 3$ coding scheme
- We could have transmitted three independent bits of information; instead we are using the channel three times (to send three encoded bits) but sending actually only one bit of information
- Define as $r = K/N$ the rate of the channel coding scheme:
  - Here, here K is the number of information bits and $N \geq K$ is the number of encoded bits
  - $0 \leq r \leq 1$
- For the repeat-by-n coding, $K = 1$ information bit is repeated $N = n$ times, i.e., the rate $r = 1/n$
- The two sets can be made increasingly nonoverlapping by increasing n. This reduces the probability of decoding error $p_{error} = p(e = 1)$ at the receiver but the price paid is that the rate $r \to 0$

Repetition Coding

Probability of Decoding Error

- Decoding error will occur in rate $r = 1/3$ repetition code if 2 or 3 bits are in error
- What is the probability of that occuring? The answer is given by the Binomial PMF

$$p_{error} = \binom{3}{2} p^2 (1-p) + \binom{3}{1} p^3 = 3p^2 + p^3$$

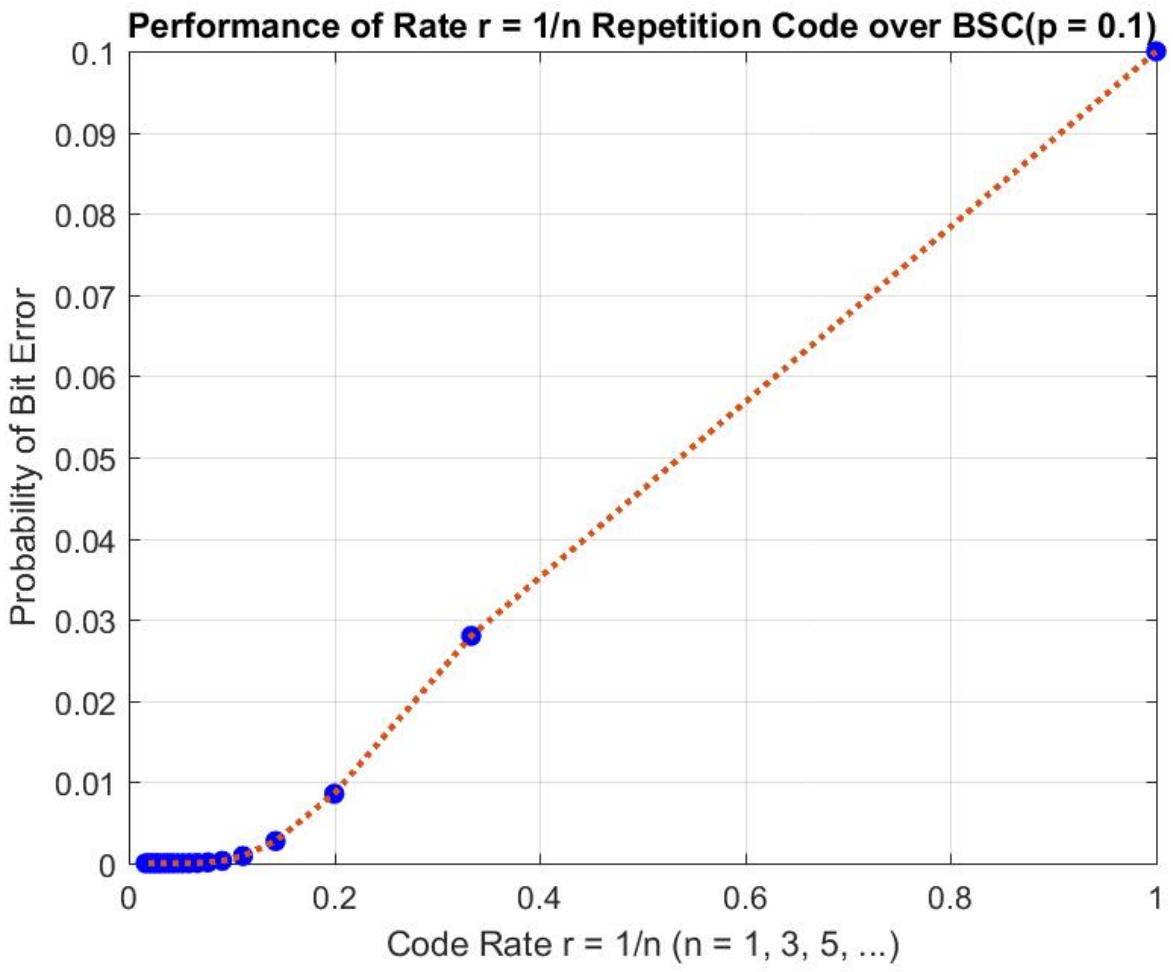
$\rightarrow$ If $p = 0.1$, $p_{error} = 3 \times 0.1^2 \times 0.9 + 0.1^3 \approx 0.03$

- Generalization to rate $1/n$ repetition code:

$$p_{error} = \sum_{k=(n+1)/2}^{n} \binom{n}{k} p^k (1-p)^{n-k}$$

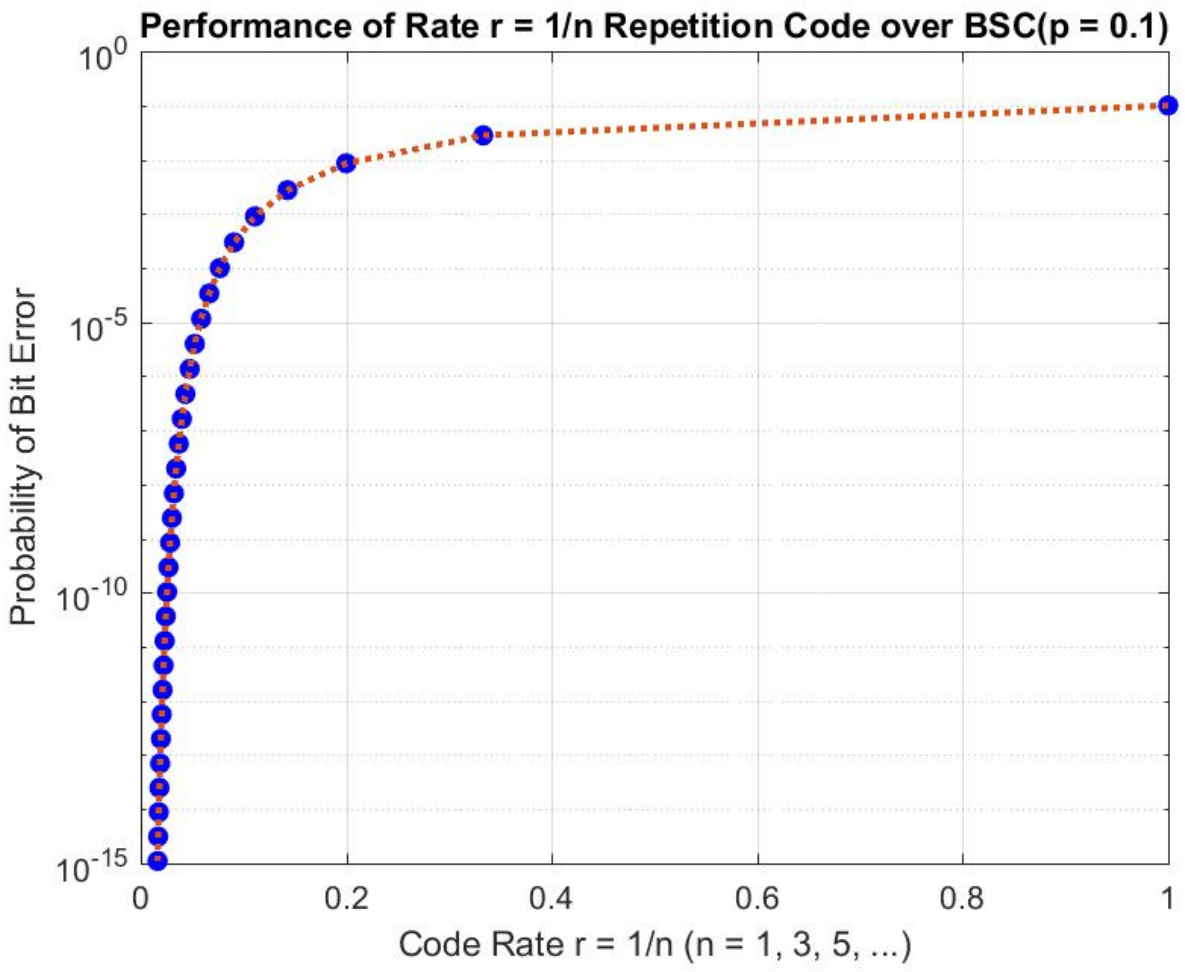
Performance of Repetition Code

over BSC(p)



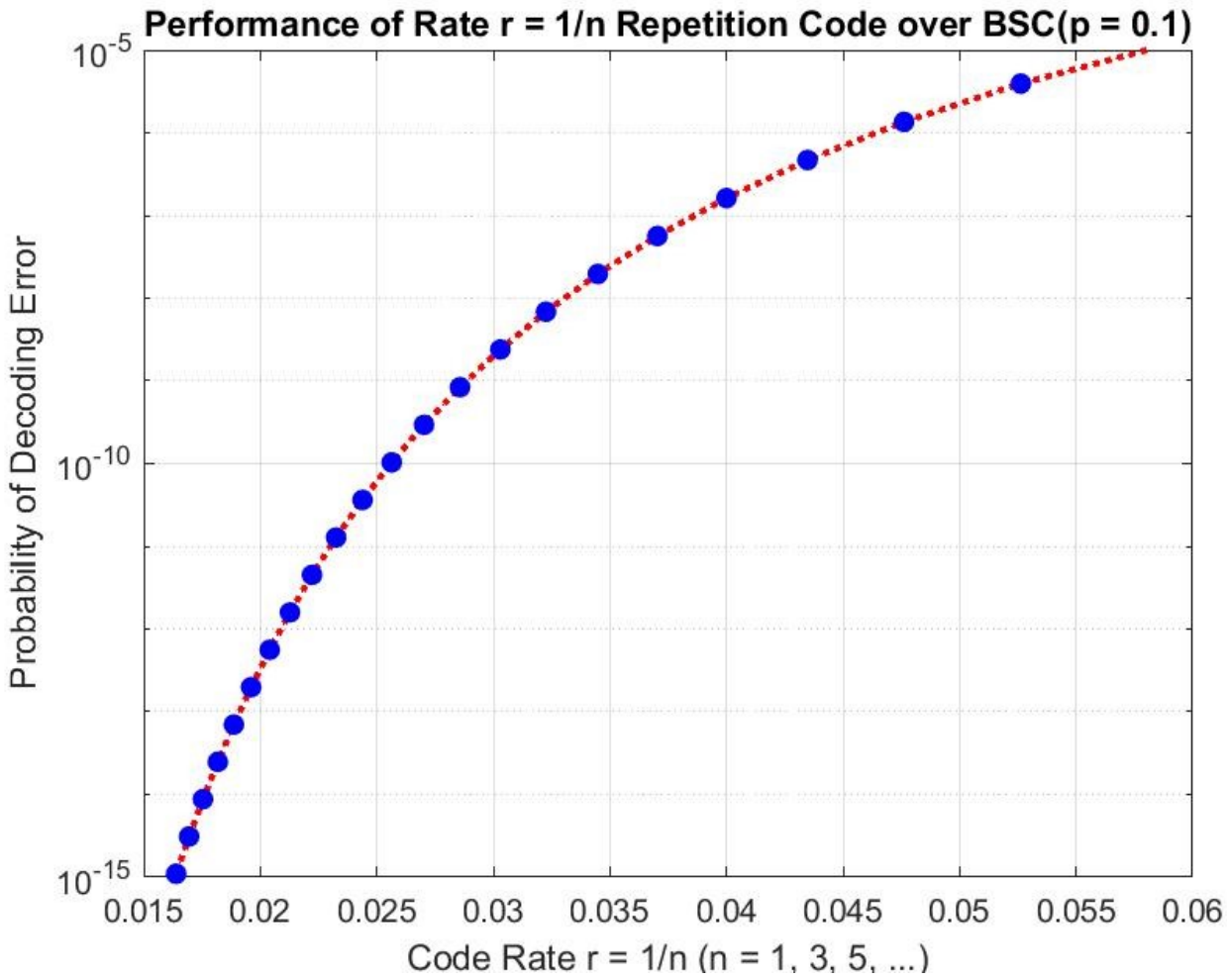
Performance of Repetition Code

over BSC(p)



Performance of Repetition Code

over BSC(p)



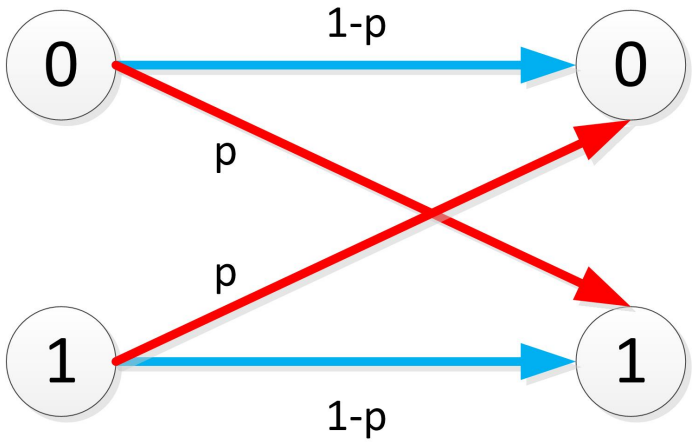
Performance of Repetition Code

over BSC(p)

- If $p = 0.1$, and we would like to design a repetition code that reduces this to $p_{error} = 10^{-15}$, we need $r = 0.016$, i.e, $n = 63$
  - $\rightarrow$ Repeat each bit sixty-three times!
- Error probability p_{error} can be arbitrarily reduced, but the price paid is huge. The rate r of information transfer is reduced significantly.

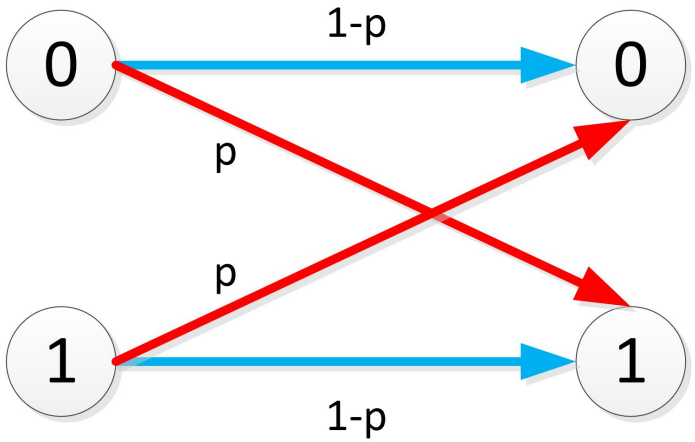
Several Questions about Information Transfer over BSC(p)

- The binary symmetric channel (BSC(p)) model under consideration
  - We will assume that the value of p is known both at the sender and the receiver



Several Questions about Information Transfer over BSC(p)

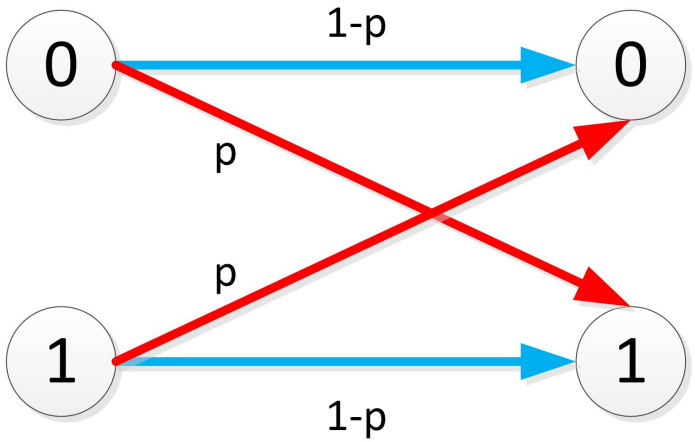
- The binary symmetric channel (BSC(p)) model under consideration
  - We will assume that the value of p is known both at the sender and the receiver



- Let us say that the transmitter is a Bernouilli($q = 0.5$) source which sends 1000 bits/sec, and the BSC p value is 0. What is the rate at which the information is received over this BSC($p = 0$)?

Several Questions about Information Transfer over BSC(p)

- The binary symmetric channel (BSC(p)) model under consideration
  - We will assume that the value of p is known both at the sender and the receiver



- Let us say that the transmitter is a Bernouilli($q = 0.5$) source which sends 1000 bits/sec, and the BSC p value is 0. What is the rate at which the information is received over this BSC($p = 0$)?
  - Obviously: 1000 bits/sec

Several Questions about Information Transfer over BSC(p)

- Question 1: Suppose $p = 0.1$. What is the rate at which the information is received over this BSC($p = 0.1$)?

Several Questions about Information Transfer over BSC(p)

- Question 1: Suppose $p = 0.1$. What is the rate at which the information is received over this BSC($p = 0.1$)?
- Question 2: For what value of p does the rate of information transfer over the BSC become zero?

Several Questions about Information Transfer over BSC(p)

- Question 1: Suppose $p = 0.1$. What is the rate at which the information is received over this BSC($p = 0.1$)?
- Question 2: For what value of p does the rate of information transfer over the BSC become zero?
  - ▷ One answer to Q2: the information transfer rate becomes zero when $p = 1$
    - → This answer is not right. When $p = 1$, the BSC behaves in a deterministic - nonrandom - manner since it flips each bit without fail. Therefore, the receiver just flips each received bit at the output of the channel to obtain the data at the full rate of 1000 bits/sec with zero probability of error

Several Questions about Information Transfer over BSC(p)

- Question 1: Suppose $p = 0.1$. What is the rate at which the information is received over this BSC($p = 0.1$)?
- Question 2: For what value of p does the rate of information transfer over the BSC become zero?
  - ▷ One answer to Q2: the information transfer rate becomes zero when $p = 1$
    - → This answer is not right. When $p = 1$, the BSC behaves in a deterministic - nonrandom - manner since it flips each bit without fail. Therefore, the receiver just flips each received bit at the output of the channel to obtain the data at the full rate of 1000 bits/sec with zero probability of error
  - ▷ One answer to Q1: 900 bits/sec (just subtract 10% of bits that are going to be received in error)

Several Questions about Information Transfer over BSC(p)

- Question 1: Suppose $p = 0.1$. What is the rate at which the information is received over this BSC($p = 0.1$)?
- Question 2: For what value of p does the rate of information transfer over the BSC become zero?
  - ▷ One answer to Q2: the information transfer rate becomes zero when $p = 1$
    - → This answer is not right. When $p = 1$, the BSC behaves in a deterministic - nonrandom - manner since it flips each bit without fail. Therefore, the receiver just flips each received bit at the output of the channel to obtain the data at the full rate of 1000 bits/sec with zero probability of error
  - ▷ One answer to Q1: 900 bits/sec (just subtract 10% of bits that are going to be received in error)
    - → This answer is not correct either. The strategy of determining the received data rate as $1000 \times (1 - p)$ does not work as we have seen earlier for the case of $p = 1$.

Several Questions about Information Transfer over BSC(p)

- Question 1: Suppose $p = 0.1$. What is the rate at which the information is received over this BSC($p = 0.1$)?
- Question 2: For what value of p does the rate of information transfer over the BSC become zero?
  - ▷ One answer to Q2: the information transfer rate becomes zero when $p = 1$
    - → This answer is not right. When $p = 1$, the BSC behaves in a deterministic - nonrandom - manner since it flips each bit without fail. Therefore, the receiver just flips each received bit at the output of the channel to obtain the data at the full rate of 1000 bits/sec with zero probability of error
  - ▷ One answer to Q1: 900 bits/sec (just subtract 10% of bits that are going to be received in error)
    - → This answer is not correct either. The strategy of determining the received data rate as $1000 \times (1 - p)$ does not work as we have seen earlier for the case of $p = 1$.
    - → Also, although the receiver knows that the BSC flips 10% of the bits in average, how can it get the remaining 900 bits/sec since it does not know which bits the BSC has flipped?

Several Questions about Information Transfer over BSC(p)

- These adhoc answers - based on guesswork - don't turn out right, they do not satisfy the logical expectation

Several Questions about Information Transfer over BSC(p)

- These adhoc answers - based on guesswork - don't turn out right, they do not satisfy the logical expectation
- Could it be that if we answer these questions using the mathematics of the information transfer that we have studied, the results are logically more satisfactory?

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- To answer Question 2, note that when $p = 0.5$, $p(X \mid Y) = p(X)$.

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- To answer Question 2, note that when $p = 0.5$, $p(X \mid Y) = p(X)$.
  - i.e., the BSC output Y becomes independent of the transmitted X

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- To answer Question 2, note that when $p = 0.5$, $p(X \mid Y) = p(X)$.
  - ▷ i.e., the BSC output Y becomes independent of the transmitted X

Why?

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- To answer Question 2, note that when $p = 0.5$, $p(X \mid Y) = p(X)$.
  - i.e., the BSC output Y becomes independent of the transmitted X

Why?

  - Since

$$p(Y = 0 \mid X = 0) = 1 - p = 0.5; \quad p(Y = 0 \mid X = 1) = p = 0.5,$$

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- To answer Question 2, note that when $p = 0.5$, $p(X \mid Y) = p(X)$.
  - i.e., the BSC output Y becomes independent of the transmitted X

Why?

  - Since

$$p(Y = 0 \mid X = 0) = 1 - p = 0.5; \quad p(Y = 0 \mid X = 1) = p = 0.5,$$

  - We have the following:

$$\begin{aligned} p(Y = 0) &= p(Y = 0, X = 0) + p(Y = 0, X = 1) \\ &= p(Y = 0 \mid X = 0)p(X = 0) + p(Y = 0 \mid X = 1)p(X = 1) \\ &= 0.5 \times (1 - q + q) = 0.5 \end{aligned}$$

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- To answer Question 2, note that when $p = 0.5$, $p(X \mid Y) = p(X)$.
  - i.e., the BSC output Y becomes independent of the transmitted X

Why?

  - Since

$$p(Y = 0 \mid X = 0) = 1 - p = 0.5; \quad p(Y = 0 \mid X = 1) = p = 0.5,$$

  - We have the following:

$$\begin{aligned} p(Y = 0) &= p(Y = 0, X = 0) + p(Y = 0, X = 1) \\ &= p(Y = 0 \mid X = 0)p(X = 0) + p(Y = 0 \mid X = 1)p(X = 1) \\ &= 0.5 \times (1 - q + q) = 0.5 \end{aligned}$$

  - i.e., regardless of the value of q, $p(Y = 0) = p(Y = 1) = 0.5$

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- Therefore,

$$
\begin{aligned}
p(X=0 \mid Y=0) &= p(Y=0 \mid X=0)p(X=0)/p(Y=0) \\
&= 0.5 \times p(X=0)/0.5 \\
&= p(X=0).
\end{aligned}
$$

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- Therefore,

$$\begin{aligned} p(X=0 \mid Y=0) &= p(Y=0 \mid X=0)p(X=0)/p(Y=0) \\ &= 0.5 \times p(X=0)/0.5 \\ &= p(X=0). \end{aligned}$$

- Similarly, it can be shown than

$$\begin{aligned} p(X=1 \mid Y=0) &= p(X=1) \\ p(X=0 \mid Y=1) &= p(X=0) \\ p(X=1 \mid Y=1) &= p(X=1). \end{aligned}$$

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- Therefore,

$$\begin{aligned} p(X=0 \mid Y=0) &= p(Y=0 \mid X=0)p(X=0)/p(Y=0) \\ &= 0.5 \times p(X=0)/0.5 \\ &= p(X=0). \end{aligned}$$

- Similarly, it can be shown than

$$\begin{aligned} p(X=1 \mid Y=0) &= p(X=1) \\ p(X=0 \mid Y=1) &= p(X=0) \\ p(X=1 \mid Y=1) &= p(X=1). \end{aligned}$$

- Thus, we have shown that $p(X \mid Y) = p(X)$. This, however, implies that the conditional entropy $H(X \mid Y) = H(X)$

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- Therefore,

$$\begin{aligned} p(X=0 \mid Y=0) &= p(Y=0 \mid X=0)p(X=0)/p(Y=0) \\ &= 0.5 \times p(X=0)/0.5 \\ &= p(X=0). \end{aligned}$$

- Similarly, it can be shown than

$$\begin{aligned} p(X=1 \mid Y=0) &= p(X=1) \\ p(X=0 \mid Y=1) &= p(X=0) \\ p(X=1 \mid Y=1) &= p(X=1). \end{aligned}$$

- Thus, we have shown that $p(X \mid Y) = p(X)$. This, however, implies that the conditional entropy $H(X \mid Y) = H(X)$
- Therefore, the information transfer $I(X;Y) = H(X) - H(X \mid Y) = 0$.

Several Questions about Information Transfer over BSC(p)

Answer of Q2

- Therefore,

$$\begin{aligned} p(X=0 \mid Y=0) &= p(Y=0 \mid X=0)p(X=0)/p(Y=0) \\ &= 0.5 \times p(X=0)/0.5 \\ &= p(X=0). \end{aligned}$$

- Similarly, it can be shown than

$$\begin{aligned} p(X=1 \mid Y=0) &= p(X=1) \\ p(X=0 \mid Y=1) &= p(X=0) \\ p(X=1 \mid Y=1) &= p(X=1). \end{aligned}$$

- Thus, we have shown that $p(X \mid Y) = p(X)$. This, however, implies that the conditional entropy $H(X \mid Y) = H(X)$
- Therefore, the information transfer $I(X;Y) = H(X) - H(X \mid Y) = 0$.
  - Since Y becomes independent of X in this case, X fails to exert the influence on Y. The transmitter ceases to supply the information to the receiver.

Several Questions about Information Transfer over BSC(p)

Answer of Q1

- To answer Question 1, we show next that for the BSC(p), $H(X \mid Y) = H_b(p)$, where

$$H_b(p) = -(p \log_2(p) + (1-p) \log_2(1-p)$$

is the binary Entropy function

Several Questions about Information Transfer over BSC(p)

Answer of Q1

- To answer Question 1, we show next that for the BSC(p), $H(X \mid Y) = H_b(p)$, where

$$H_b(p) = -(p \log_2(p) + (1-p) \log_2(1-p)$$

is the binary Entropy function

  - Since

$$p(Y = 0 \mid X = 0) = 1 - p; \quad p(Y = 0 \mid X = 1) = p,$$

    and

$$p(X = 0) = p(X = 1) = 0.5$$

  - we obtain the following:

Several Questions about Information Transfer over BSC(p)

Answer of Q1

- To answer Question 1, we show next that for the BSC(p), $H(X \mid Y) = H_b(p)$, where

$$H_b(p) = -(p \log_2(p) + (1-p)\log_2(1-p)$$

is the binary Entropy function

  - Since

$$p(Y=0 \mid X=0) = 1-p; \quad p(Y=0 \mid X=1) = p,$$

and

$$p(X=0) = p(X=1) = 0.5$$

  - we obtain the following:

$$\begin{aligned} p(Y=0) &= p(Y=0, X=0) + p(Y=0, X=1) \\ &= p(Y=0 \mid X=0)p(X=0) + p(Y=0 \mid X=1)p(X=1) \\ &= 0.5 \times (1-p+p) \\ &= 0.5 \end{aligned}$$

Several Questions about Information Transfer over BSC(p)

Answer of Q1

- Therefore,

$$p(X=0\,|\,Y=0) = p(Y=0\,|\,X=0)p(X=0)/p(Y=0)$$
$$= (1-p)\times 0.5/0.5 = 1-p.$$

- Similarly,

$$p(X=1\,|\,Y=0) = p(Y=0\,|\,X=1)p(X=1)/p(Y=0)$$
$$= p\times 0.5/0.5 = p.$$

- Thus, when $Y=0$ is received, the information source X reduces to the Bernouilli(p) RV from Bernouilli($q=0.5$) RV
- Similarly, when $Y=1$ is received, the information source X again reduces to the Bernouilli(p) RV from Bernouilli($q=0.5$) RV
  - This is evident by the symmetry of the BSC, and it can be mathematically shown by repeating the above derivation for $Y=1$
- Therefore, $H(X\,|\,Y) = H_b(p)$ is proven

Several Questions about Information Transfer over BSC(p)

Answer of Q1

- Therefore,

$$p(X=0\,|\,Y=0) = p(Y=0\,|\,X=0)p(X=0)/p(Y=0)$$
$$= (1-p)\times 0.5/0.5 = 1-p.$$

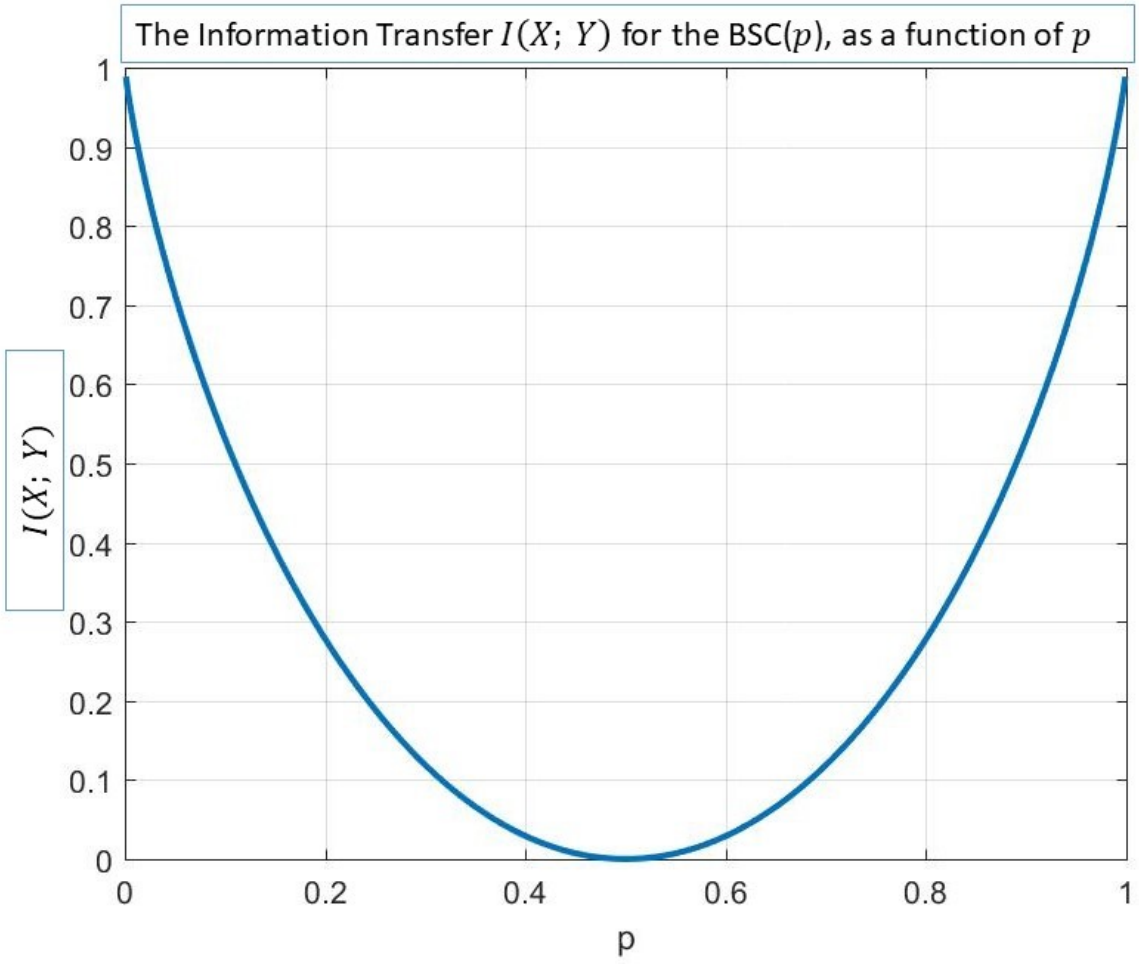
- Similarly,

$$p(X=1\,|\,Y=0) = p(Y=0\,|\,X=1)p(X=1)/p(Y=0)$$
$$= p\times 0.5/0.5 = p.$$

- Thus, when $Y=0$ is received, the information source X reduces to the Bernouilli(p) RV from Bernouilli($q=0.5$) RV
- Similarly, when $Y=1$ is received, the information source X again reduces to the Bernouilli(p) RV from Bernouilli($q=0.5$) RV
  - This is evident by the symmetry of the BSC, and it can be mathematically shown by repeating the above derivation for $Y=1$
- Therefore, $H(X\,|\,Y) = H_b(p)$ is proven
- Since $H(X)=1$ bit, the information transfer equals $I(X;Y) = 1 - H_b(p)$ bits

Performance of the Ideal Channel Code

over BSC(p)



Repetition Code versus the Ideal Channel Code

over BSC(p)

- A theoretical possibility: over BSC($p = 0.1$), the information can be transmitted with arbitrarily low p_{error} with rate r as high as $1 - H_b(p = 0.1) = 0.53$!
  - → Rate r does not have to be reduced to near zero
  - → In fact, p_{error} can be reduced to 10^{-15}, or 10^{-100} or 10^{-1000} at the same rate of 0.53

Repetition Code versus the Ideal Channel Code

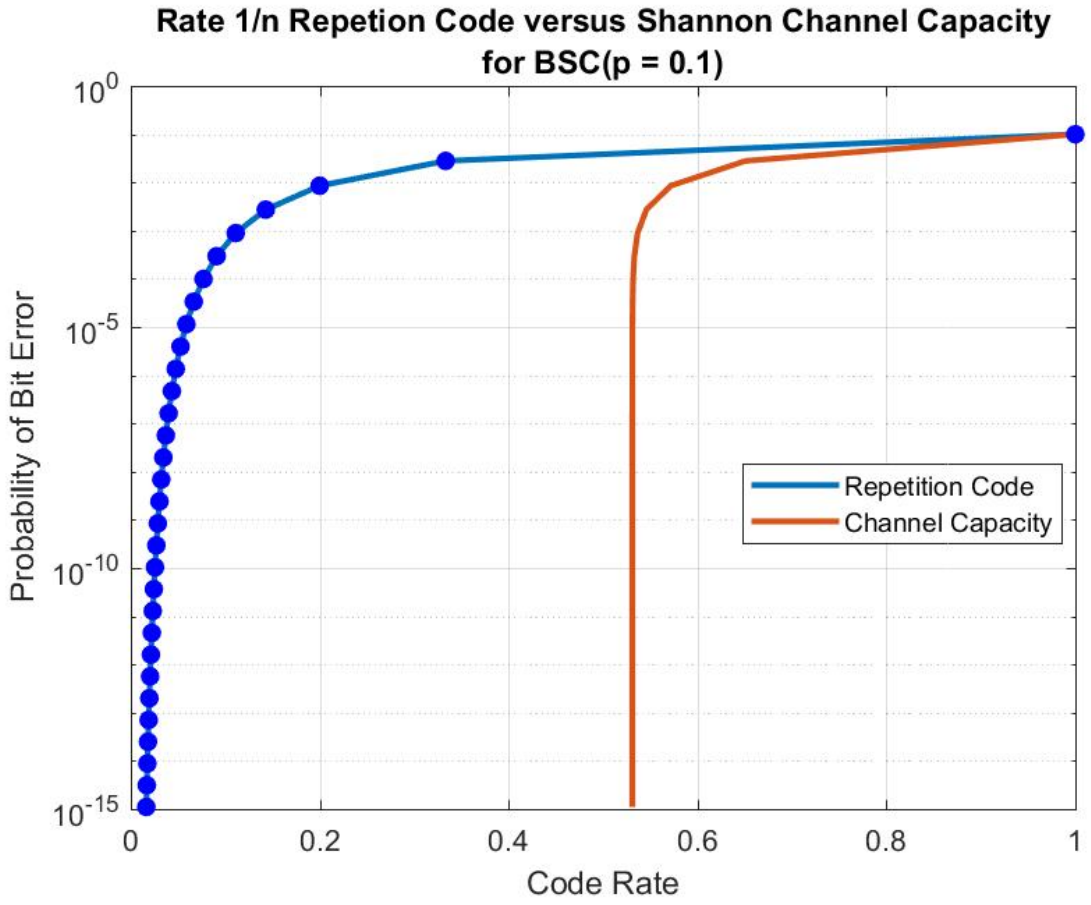
over BSC(p)

- A theoretical possibility: over BSC($p = 0.1$), the information can be transmitted with arbitrarily low p_{error} with rate r as high as $1 - H_b(p = 0.1) = 0.53$!
  - → Rate r does not have to be reduced to near zero
  - → In fact, p_{error} can be reduced to 10^{-15}, or 10^{-100} or 10^{-1000} at the same rate of 0.53
    - ▷ Prior to 1948, the general belief was that attempting to reduce $p_{error} \to 0$ requires $r \to 0$.
    - ▷ Shannon showed that nonzero, positive-valued, r is possible even if p_{error} is required to approach 0

Repetition Code versus the Ideal Channel Code

over BSC(p)

- The information transfer $I(X;Y)$ is called the channel capacity



Channel Coding Theorem

- A fundamental theorem of the Information Theory for the Noisy Channels [Shannon 1948 paper]:
  - → The rate r of the channel coding scheme $r \leq I(X;Y) = H(X) - H(X \mid Y)$.
    - ▷ Rate can be made arbitrarily close to, but not greater than, $I(X;Y)$

Channel Coding Theorem

- A fundamental theorem of the Information Theory for the Noisy Channels [Shannon 1948 paper]:
  - → The rate r of the channel coding scheme $r \leq I(X;Y) = H(X) - H(X \mid Y)$.
    - ▷ Rate can be made arbitrarily close to, but not greater than, $I(X;Y)$
  - → This is the (Noisy) Channel Coding Theorem

Channel Coding Theorem

- A fundamental theorem of the Information Theory for the Noisy Channels [Shannon 1948 paper]:
  - → The rate r of the channel coding scheme $r \leq I(X;Y) = H(X) - H(X \mid Y)$.
    - ▷ Rate can be made arbitrarily close to, but not greater than, $I(X;Y)$
  - → This is the (Noisy) Channel Coding Theorem
- How to prove this theorem?

Design of Channel Coding

- We have seen the objective of the channel coding: to make a noisy channel appear noiseless

Design of Channel Coding

- We have seen the objective of the channel coding: to make a noisy channel appear noiseless
- We have also previewed the design of the channel coding: introduce the redundancies in the transmitted bits

Design of Channel Coding

- We have seen the objective of the channel coding: to make a noisy channel appear noiseless
- We have also previewed the design of the channel coding: introduce the redundancies in the transmitted bits
- Now we look at the practical design techniques for the channel coding, and also consider how the redundant bits that the channel encoder adds help in the noise removal

Number of Valid Codwords versus Number of Possible Received Words

- Notations:

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits
  - $0 \leq r = k/n \leq 1$: the rate of the channel code

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits
  - $0 \leq r = k/n \leq 1$: the rate of the channel code
  - We will interchangeably use small-case or capitalized notations k or K; n or N; except, for the rate r, for which the capitalized notation R is reserved for the source coding rate

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits
  - $0 \leq r = k/n \leq 1$: the rate of the channel code
  - We will interchangeably use small-case or capitalized notations k or K; n or N; except, for the rate r, for which the capitalized notation R is reserved for the source coding rate
- A consequence of $k < n$:

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits
  - $0 \leq r = k/n \leq 1$: the rate of the channel code
  - We will interchangeably use small-case or capitalized notations k or K; n or N; except, for the rate r, for which the capitalized notation R is reserved for the source coding rate
- A consequence of $k < n$:
  - Since the length of the information bit sequence equals k bits, there are total 2^k valid codewords that the encoder can generate

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits
  - $0 \leq r = k/n \leq 1$: the rate of the channel code
  - We will interchangeably use small-case or capitalized notations k or K; n or N; except, for the rate r, for which the capitalized notation R is reserved for the source coding rate
- A consequence of $k < n$:
  - Since the length of the information bit sequence equals k bits, there are total 2^k valid codewords that the encoder can generate
  - However, since the length of the received bit sequence is n bits, there are total $2^n \geq 2^k$ bit sequences that the receiver can observe

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits
  - $0 \leq r = k/n \leq 1$: the rate of the channel code
  - We will interchangeably use small-case or capitalized notations k or K; n or N; except, for the rate r, for which the capitalized notation R is reserved for the source coding rate
- A consequence of $k < n$:
  - Since the length of the information bit sequence equals k bits, there are total 2^k valid codewords that the encoder can generate
  - However, since the length of the received bit sequence is n bits, there are total $2^n \geq 2^k$ bit sequences that the receiver can observe
  - In the absence of the noise, the receiver observes only one of 2^k valid codewords; however, in the presence of noise, the receiver can also additionally observe any one of $2^n - 2^k$ invalid codewords

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits
  - $0 \leq r = k/n \leq 1$: the rate of the channel code
  - We will interchangeably use small-case or capitalized notations k or K; n or N; except, for the rate r, for which the capitalized notation R is reserved for the source coding rate
- A consequence of $k < n$:
  - Since the length of the information bit sequence equals k bits, there are total 2^k valid codewords that the encoder can generate
  - However, since the length of the received bit sequence is n bits, there are total $2^n \geq 2^k$ bit sequences that the receiver can observe
  - In the absence of the noise, the receiver observes only one of 2^k valid codewords; however, in the presence of noise, the receiver can also additionally observe any one of $2^n - 2^k$ invalid codewords
  - The main idea of channel coding: make the receiver <u>intelligent</u> so that it can decide whether it is observing an invalid codeword (in which case it declares that "the channel has introduced an error") and possibly correct this error

Number of Valid Codwords versus Number of Possible Received Words

- Notations:
  - n: length of codeword in bits; $1 \leq k \leq n$: length of the information bit sequence in bits
  - $0 \leq r = k/n \leq 1$: the rate of the channel code
  - We will interchangeably use small-case or capitalized notations k or K; n or N; except, for the rate r, for which the capitalized notation R is reserved for the source coding rate
- A consequence of $k < n$:
  - Since the length of the information bit sequence equals k bits, there are total 2^k valid codewords that the encoder can generate
  - However, since the length of the received bit sequence is n bits, there are total $2^n \geq 2^k$ bit sequences that the receiver can observe
  - In the absence of the noise, the receiver observes only one of 2^k valid codewords; however, in the presence of noise, the receiver can also additionally observe any one of $2^n - 2^k$ invalid codewords
  - The main idea of channel coding: make the receiver <u>intelligent</u> so that it can decide whether it is observing an invalid codeword (in which case it declares that "the channel has introduced an error") and possibly correct this error
- To obtain a conceptual understanding of how the receiver can do the above, we introduce the notion of Hamming Distance among the codewords

Hamming Weight and Hamming Distance

for Binary Sequences

- Hamming Weight is simply the number of ones in the sequence

Hamming Weight and Hamming Distance

for Binary Sequences

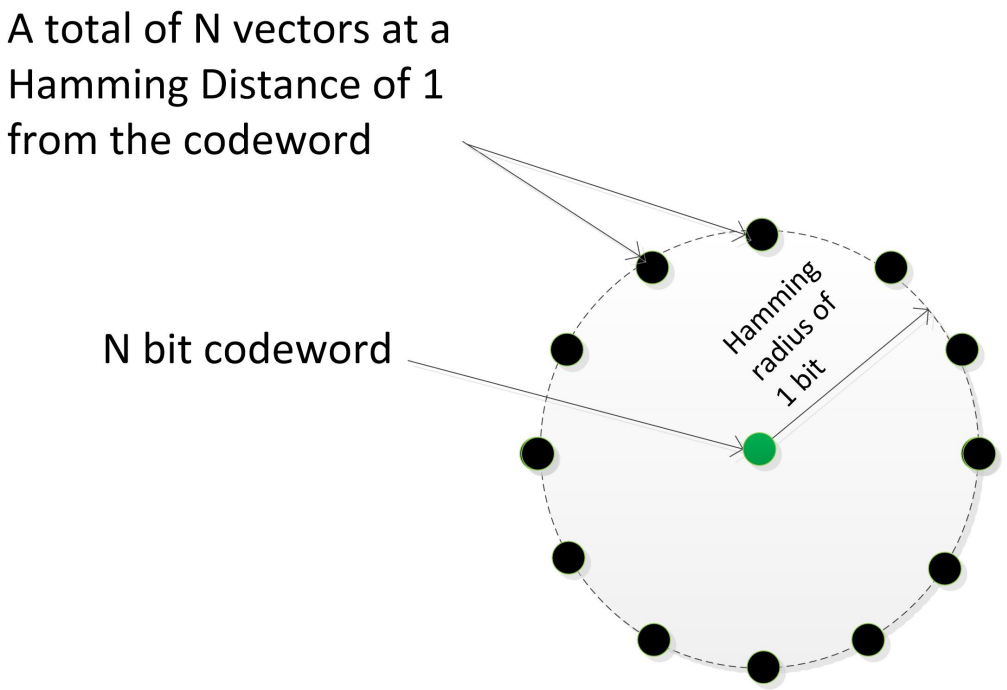
- Hamming Weight is simply the number of ones in the sequence
- Hamming Distance $d_H^{m,n}$:
  - → Let $\mathbf{c}_m$ and $\mathbf{c}_n$ be two codewords, and $\mathbf{e}_{m,n} = \mathbf{c}_m \oplus \mathbf{c}_n$ be the difference vector (also binary) between these codewords
    - ▷ $\mathbf{e}_{m,n}$ has ones only in those places where the bits of $\mathbf{c}_m$ and $\mathbf{c}_n$ are differing; $\mathbf{e}_{m,n}$ is zero otherwise
  - → Hamming Distance $d_H^{m,n}$ between $\mathbf{c}_m$ and $\mathbf{c}_n$ is the Hamming Weight of $\mathbf{e}_{m,n}$
    - ▷ Hamming Distance is simply the number of places in which two binary sequences differ

Minimum Hamming Distance d_{min}

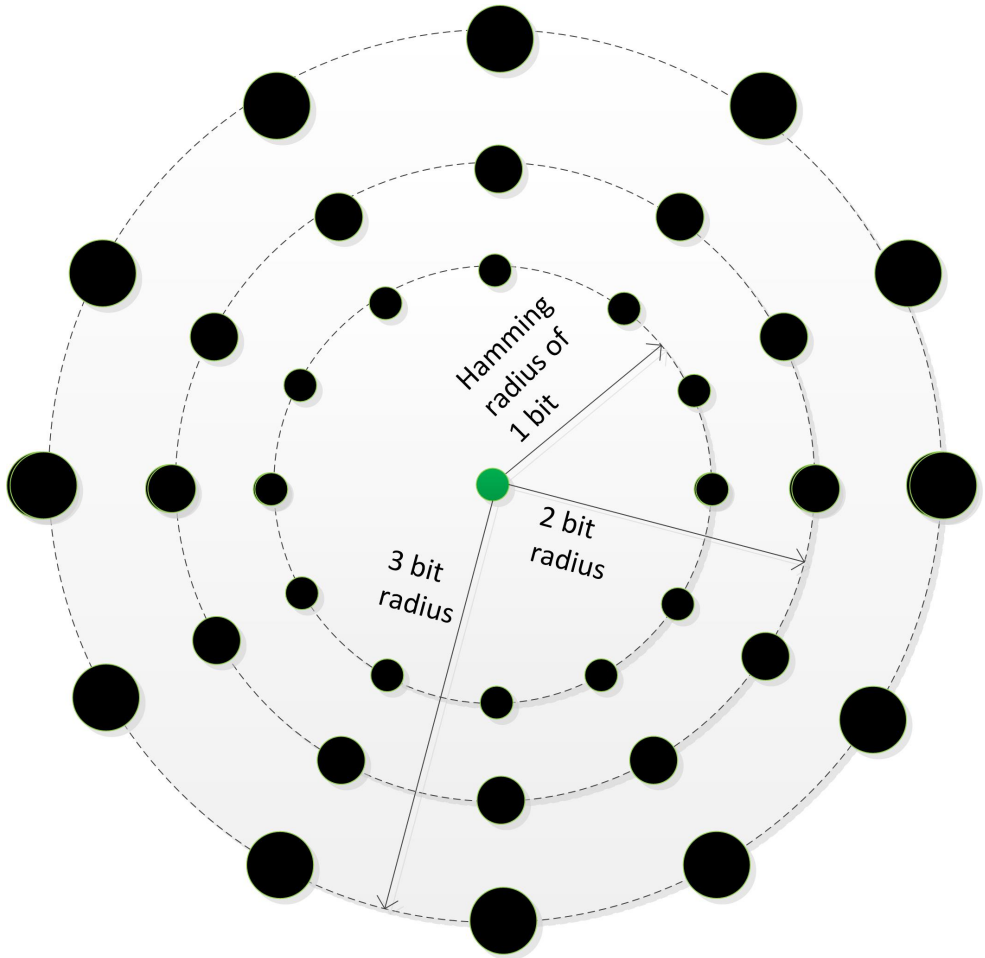
for A Channel Code

- $d_H^{\min}$ is defined for a channel code with rate $r = \dfrac{K}{N}$
  - $\rightarrow$ This code takes K bit information sequence and generates N bit codeword
    - Thus, there are a total of 2^K codewords
- $d_H^{\min}$ for this channel code is the minimum Hamming distance between any two pairs of this channel code
- $d_H^{\min}$ relates to the error detection and error correction capabilities of the channel code. This can be visualized by drawing Hamming Circles as shown next

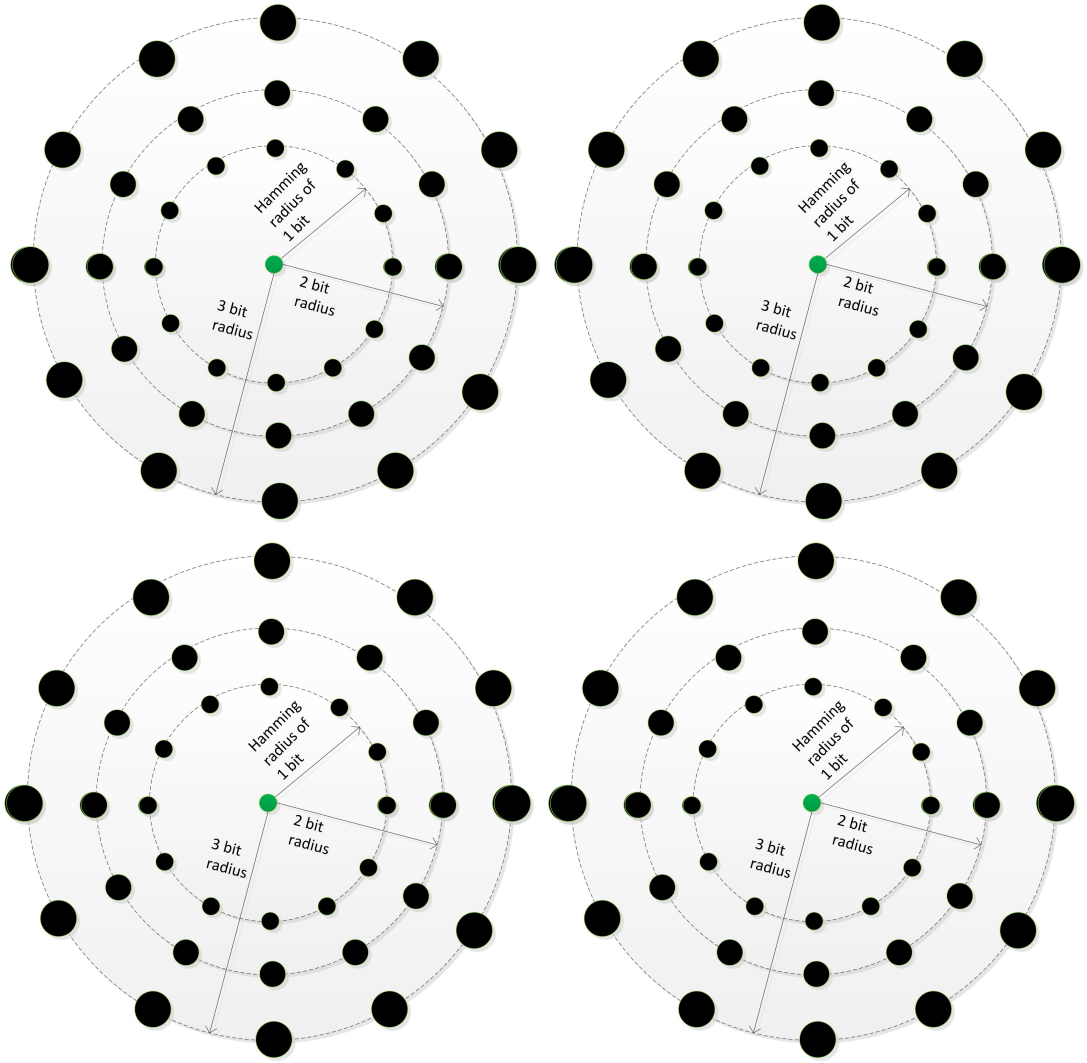
Hamming Circle of Radius 1 around a codeword



Hamming Circles around a codeword



Hamming Circles around several codewords



Minimum Hamming Distance Decoder

- Detects the transmitted codeword as the one at the center of the Hamming sphere in which the received word falls in

Minimum Hamming Distance Decoder

- Detects the transmitted codeword as the one at the center of the Hamming sphere in which the received word falls in
- Can be shown to be the optimal Bayesian (ML) receiver provided all 2^k transmitted codewords are equally likely to have been transmitted

Minimum Hamming Distance Decoder

- Detects the transmitted codeword as the one at the center of the Hamming sphere in which the received word falls in
- Can be shown to be the optimal Bayesian (ML) receiver provided all 2^k transmitted codewords are equally likely to have been transmitted
- We have actually proven this for the repetition code

Minimum Hamming Distance Decoder

- Detects the transmitted codeword as the one at the center of the Hamming sphere in which the received word falls in
- Can be shown to be the optimal Bayesian (ML) receiver provided all 2^k transmitted codewords are equally likely to have been transmitted
- We have actually proven this for the repetition code
- For practical channel coding design, this is the main challenge — since the both 2^k and 2^n become very large numbers as k and n increase. Therefore, a brute-force distance minimization becomes infeasible to implement

Error Detection t_d and Error Correction t_c Capability of a Channel Code

- Error detection capability t_d in bits:

Error Detection t_d and Error Correction t_c Capability of a Channel Code

- Error detection capability t_d in bits:
  - Defined as the maximum number of bits that can be flipped by the BSC above which the detector may not be able to identify that the BSC has introduced bit errors

Error Detection t_d and Error Correction t_c Capability of a Channel Code

- Error detection capability t_d in bits:
  - Defined as the maximum number of bits that can be flipped by the BSC above which the detector may not be able to identify that the BSC has introduced bit errors
- Error correction capability t_c in bits:

Error Detection t_d and Error Correction t_c Capability of a Channel Code

- Error detection capability t_d in bits:
  - Defined as the maximum number of bits that can be flipped by the BSC above which the detector may not be able to identify that the BSC has introduced bit errors
- Error correction capability t_c in bits:
  - Defined as the maximum number of bits that can be flipped by the BSC at or below which the minimum distance detector is guaranteed to decode the correct (i.e., the transmitted) codeword

Error Detection t_d and Error Correction t_c Capability of a Channel Code

- Error detection capability t_d in bits:
  - Defined as the maximum number of bits that can be flipped by the BSC above which the detector may not be able to identify that the BSC has introduced bit errors
- Error correction capability t_c in bits:
  - Defined as the maximum number of bits that can be flipped by the BSC at or below which the minimum distance detector is guaranteed to decode the correct (i.e., the transmitted) codeword
- Both t_d and t_c are functions of $d_H^{\min}$

Error Detection t_d and Error Correction t_c Capability of a Channel Code

- Error detection capability t_d in bits:
  - Defined as the maximum number of bits that can be flipped by the BSC above which the detector may not be able to identify that the BSC has introduced bit errors
- Error correction capability t_c in bits:
  - Defined as the maximum number of bits that can be flipped by the BSC at or below which the minimum distance detector is guaranteed to decode the correct (i.e., the transmitted) codeword
- Both t_d and t_c are functions of $d_H^{\min}$
- To see this, we will draw simpler, two-dimensional, versions of the Hamming spheres for different values of $d_H^{\min}$ (note that different $d_H^{\min}$ imply different coding schemes)

Error Detection and Error Correction Capabilities

of a Channel Code

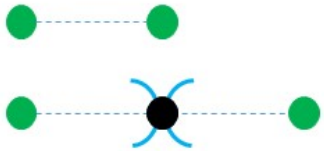
d_H^{min}	t_c	t_d
1 bit	0 bit	0 bit

: Valid codeword

: Not a valid codeword

Error Detection and Error Correction Capabilities

of a Channel Code



d_H^{min}	t_c	t_d
1 bit	0 bit	0 bit
2 bits	0 bit	1 bit

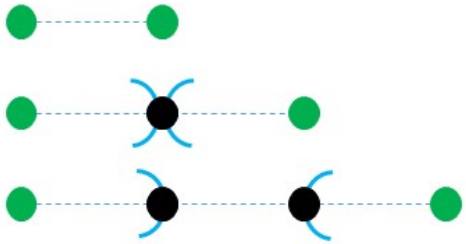
: Valid codeword

: Not a valid codeword

: Detection Boundary

Error Detection and Error Correction Capabilities

of a Channel Code



d_H^{min}	t_c	t_d
1 bit	0 bit	0 bit
2 bits	0 bit	1 bit
3 bits	1 bits	2 bit

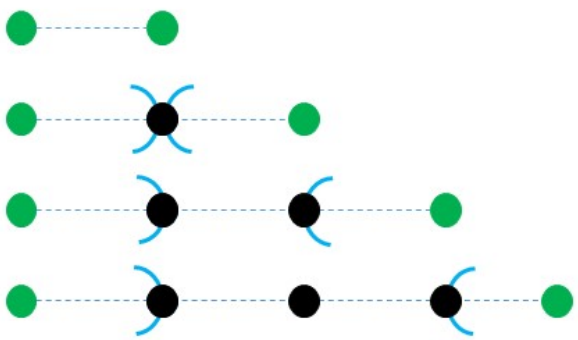
: Valid codeword

: Not a valid codeword

: Detection Boundary

Error Detection and Error Correction Capabilities

of a Channel Code



d_H^{min}	t_c	t_d
1 bit	0 bit	0 bit
2 bits	0 bit	1 bit
3 bits	1 bits	2 bit
4 bits	1 bits	3 bit

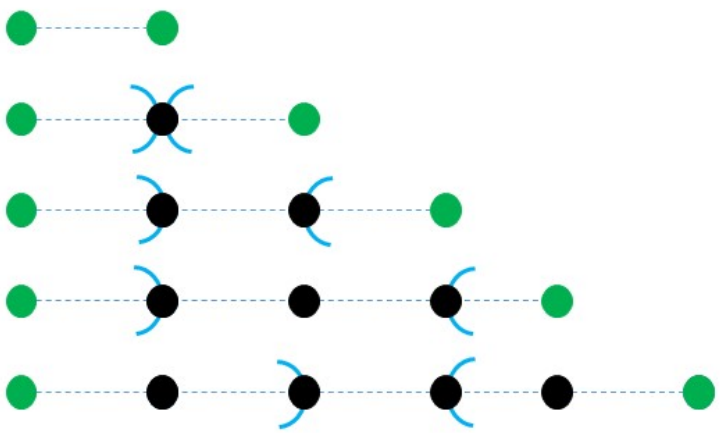
: Valid codeword

: Not a valid codeword

: Detection Boundary

Error Detection and Error Correction Capabilities

of a Channel Code



d_H^{min}	t_c	t_d
1 bit	0 bit	0 bit
2 bits	0 bit	1 bit
3 bits	1 bits	2 bit
4 bits	1 bits	3 bit
5 bits	2 bits	4 bit

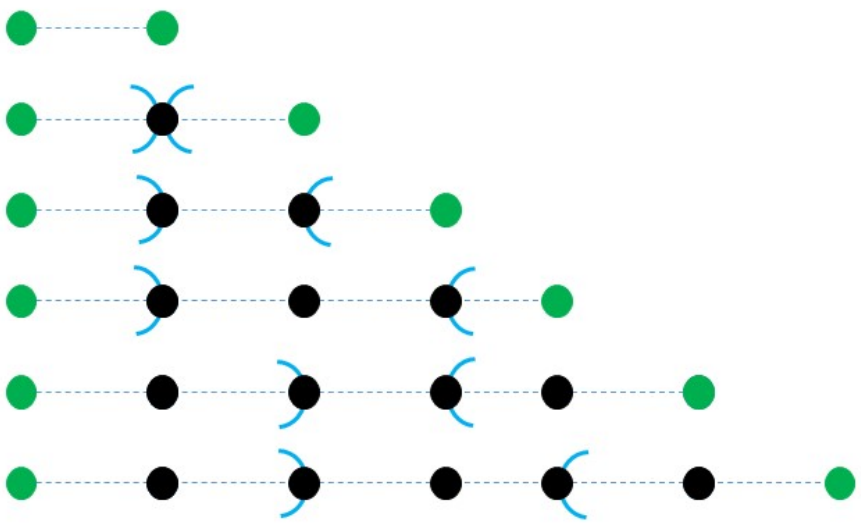
: Valid codeword

: Not a valid codeword

: Detection Boundary

Error Detection and Error Correction Capabilities

of a Channel Code



d_H^{min}	t_c	t_d
1 bit	0 bit	0 bit
2 bits	0 bit	1 bit
3 bits	1 bits	2 bit
4 bits	1 bits	3 bit
5 bits	2 bits	4 bit
6 bits	2 bits	5 bit

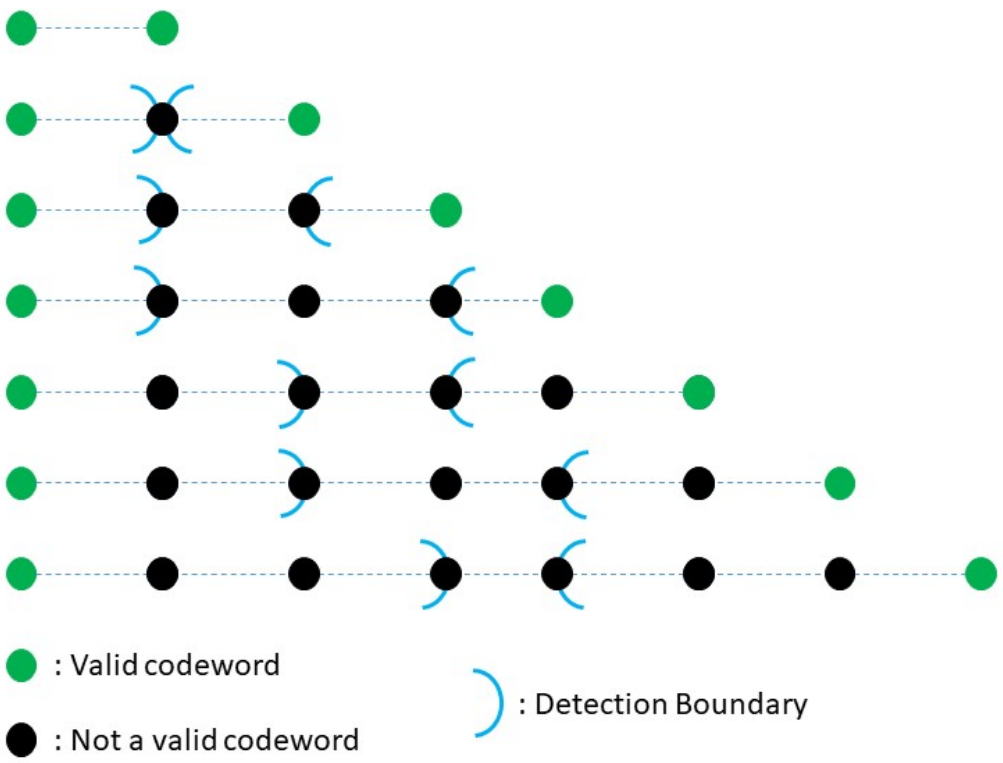
: Valid codeword

: Not a valid codeword

: Detection Boundary

Error Detection and Error Correction Capabilities

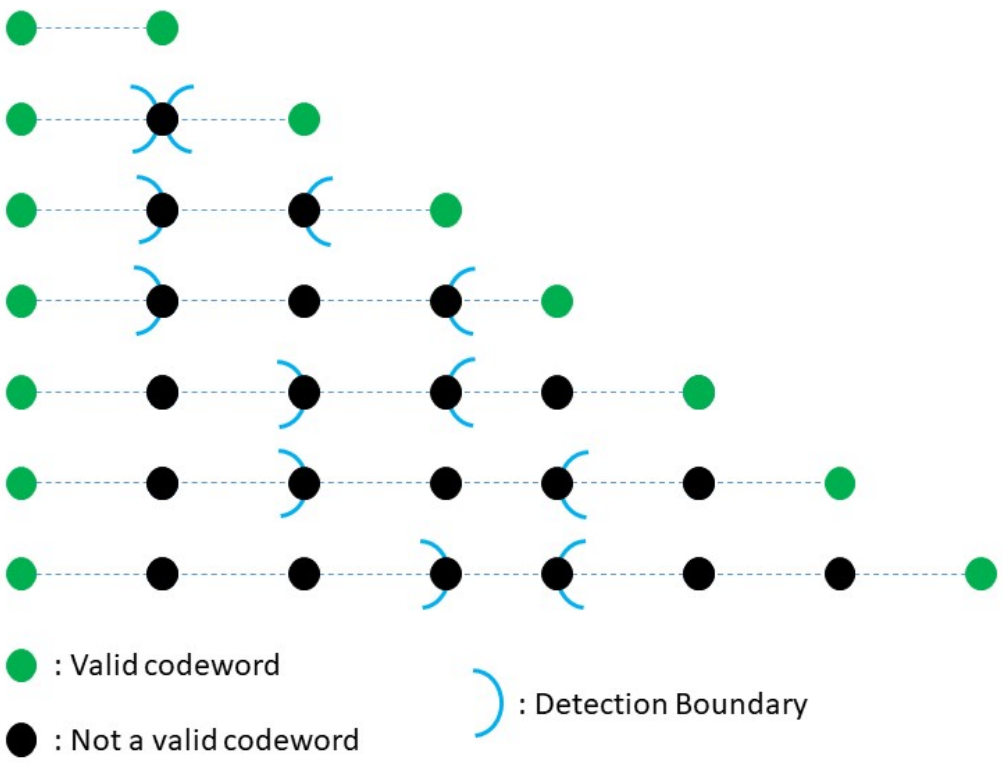
of a Channel Code



d_H^{min}	t_c	t_d
1 bit	0 bit	0 bit
2 bits	0 bit	1 bit
3 bits	1 bits	2 bit
4 bits	1 bits	3 bit
5 bits	2 bits	4 bit
6 bits	2 bits	5 bit
7 bits	3 bits	6 bit

Error Detection and Error Correction Capabilities

of a Channel Code



d_H^{min}	t_c	t_d
1 bit	0 bit	0 bit
2 bits	0 bit	1 bit
3 bits	1 bits	2 bit
4 bits	1 bits	3 bit
5 bits	2 bits	4 bit
6 bits	2 bits	5 bit
7 bits	3 bits	6 bit

- $t_d = d_H^{min} - 1$ **bits**
- $t_c = \left\lfloor \frac{d_H^{min}-1}{2} \right\rfloor$ **bits**

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:
  - → $\mathbf{m} = [m_0, m_1, \ldots, m_{k-1}]^T$: denotes a $k \times 1$ vector of message or information bits

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:
  - → $\mathbf{m} = [m_0, m_1, \ldots, m_{k-1}]^T$: denotes a $k \times 1$ vector of message or information bits
    - ▷ Here, $\mathbf{v}^T$ denotes the transpose of a vector $\mathbf{v}$. In this case, it makes a horizontal $1 \times k$ vector a tall $k \times 1$ vector.

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:
  - → $\mathbf{m} = [m_0, m_1, \ldots, m_{k-1}]^T$: denotes a $k \times 1$ vector of message or information bits
    - ▷ Here, $\mathbf{v}^T$ denotes the transpose of a vector $\mathbf{v}$. In this case, it makes a horizontal $1 \times k$ vector a tall $k \times 1$ vector.
  - → $\mathbf{c} = [c_0, c_1, \ldots, c_{n-1}]^T$: denotes an $n \times 1$ vector of encoded bits

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:
  - → $\mathbf{m} = [m_0, m_1, \ldots, m_{k-1}]^T$: denotes a $k \times 1$ vector of message or information bits
    - ▷ Here, $\mathbf{v}^T$ denotes the transpose of a vector $\mathbf{v}$. In this case, it makes a horizontal $1 \times k$ vector a tall $k \times 1$ vector.
  - → $\mathbf{c} = [c_0, c_1, \ldots, c_{n-1}]^T$: denotes an $n \times 1$ vector of encoded bits
- We will relate $\mathbf{m}$ and $\mathbf{c}$ through matrix products. Several notes:

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:
  - → $\mathbf{m} = [m_0, m_1, \ldots, m_{k-1}]^T$: denotes a $k \times 1$ vector of message or information bits
    - ▷ Here, $\mathbf{v}^T$ denotes the transpose of a vector $\mathbf{v}$. In this case, it makes a horizontal $1 \times k$ vector a tall $k \times 1$ vector.
  - → $\mathbf{c} = [c_0, c_1, \ldots, c_{n-1}]^T$: denotes an $n \times 1$ vector of encoded bits
- We will relate $\mathbf{m}$ and $\mathbf{c}$ through matrix products. Several notes:
  - → The input to the matrix products are binary (i.e., have elements that are either 0 or 1)

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:
  - → $\mathbf{m} = [m_0, m_1, \ldots, m_{k-1}]^T$: denotes a $k \times 1$ vector of message or information bits
    - ▷ Here, $\mathbf{v}^T$ denotes the transpose of a vector $\mathbf{v}$. In this case, it makes a horizontal $1 \times k$ vector a tall $k \times 1$ vector.
  - → $\mathbf{c} = [c_0, c_1, \ldots, c_{n-1}]^T$: denotes an $n \times 1$ vector of encoded bits
- We will relate $\mathbf{m}$ and $\mathbf{c}$ through matrix products. Several notes:
  - → The input to the matrix products are binary (i.e., have elements that are either 0 or 1)
  - → All the elements of the matrices are also binary

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:
  - → $\mathbf{m} = [m_0, m_1, \ldots, m_{k-1}]^T$: denotes a $k \times 1$ vector of message or information bits
    - ▷ Here, $\mathbf{v}^T$ denotes the transpose of a vector $\mathbf{v}$. In this case, it makes a horizontal $1 \times k$ vector a tall $k \times 1$ vector.
  - → $\mathbf{c} = [c_0, c_1, \ldots, c_{n-1}]^T$: denotes an $n \times 1$ vector of encoded bits
- We will relate $\mathbf{m}$ and $\mathbf{c}$ through matrix products. Several notes:
  - → The input to the matrix products are binary (i.e., have elements that are either 0 or 1)
  - → All the elements of the matrices are also binary
  - → The output of the matrix product is also turned into binary by taking modulo-two operation

Information and Encoded Bits as Vectors

- Consider the input block of k information bits and the output block of n encoded bits of a rate $r = k/n$ channel coding scheme as vectors:
  - → $\mathbf{m} = [m_0, m_1, \ldots, m_{k-1}]^T$: denotes a $k \times 1$ vector of message or information bits
    - ▷ Here, $\mathbf{v}^T$ denotes the transpose of a vector $\mathbf{v}$. In this case, it makes a horizontal $1 \times k$ vector a tall $k \times 1$ vector.
  - → $\mathbf{c} = [c_0, c_1, \ldots, c_{n-1}]^T$: denotes an $n \times 1$ vector of encoded bits
- We will relate $\mathbf{m}$ and $\mathbf{c}$ through matrix products. Several notes:
  - → The input to the matrix products are binary (i.e., have elements that are either 0 or 1)
  - → All the elements of the matrices are also binary
  - → The output of the matrix product is also turned into binary by taking modulo-two operation

A remark: in the advanced channel coding techniques, the above are not limited to be binary-valued. However, in this class, we will not consider non-binary channel codes

Generator and Parity Check Matrices

- The generator matrix $\mathbf{G}$ is a matrix of size $n \times k$ used to convert the message bit vector $\mathbf{m}$ to the encoded vector $\mathbf{c}$

$$(\mathbf{c})_{n\times 1} = (\mathbf{G})_{n\times k}\,(\mathbf{m})_{k\times 1}$$

As noted on the prior slide, although an explicity modulo-two operation at the output of this matrix-vector product is not shown, but it is implicit

Generator and Parity Check Matrices

- The generator matrix **G** is a matrix of size $n \times k$ used to convert the message bit vector **m** to the encoded vector **c**

$$(\mathbf{c})_{n\times 1} = (\mathbf{G})_{n\times k}\,(\mathbf{m})_{k\times 1}$$

  As noted on the prior slide, although an explicity modulo-two operation at the output of this matrix-vector product is not shown, but it is implicit
- The parity check matrix **H** is a matrix of size $u \times n$, such that

$$(\mathbf{H})_{u\times n}\,(\mathbf{c})_{n\times 1} = (\emptyset)_{u\times 1} \tag{1}$$

  - → Again the modulo-two operation is implicit
  - → Here $u = n - k$ is the number of parity bits (i.e., the redundancy bits) introduced by the encoding process (i.e., the matrix **G**)
  - → $\emptyset$ is an all-zero vector of size $u \times 1$

Generator and Parity Check Matrices

for the Rate $r = 1/5$ Repetition Code

- What are **G** and **H** matrices for the Rate $r = 1/5$ Repetition Code?

Generator and Parity Check Matrices

for the Rate $r = 1/n$ Repetition Code

- For the rate $1/n$ repetition code ($k = 1, u = n - 1$):
  - ▷ The parity check matrix is as follows:
    - → $\mathbf{H} = \left[1_{u\times 1}, \mathbf{I}_{u\times u}\right]$, i.e., it is a matrix of size $u \times u + 1 = (n-1) \times n$
    - → The first column of $\mathbf{H}$ of the repetition code is a (tall or vertical) vector of u ones
    - → The Identity matrix $\mathbf{I}_{u\times u}$ populates last $u = n - 1$ columns and all u rows of $\mathbf{H}$
    - → The first bit (the message bit) of $\mathbf{c}$ is copied $n - 1$ times in the repetition code. Each of $n - 1$ rows of $\mathbf{H}$ checks that the sum of the message bit and each repeated bit is 0
  - ▷ The generator matrix is as follows:
    - → $\mathbf{G} = [1, 1, \ldots, 1]^T$ is a tall vector of size $n \times (k = 1)$
    - → This ensures that in the product $\mathbf{G}\,\mathbf{m}$, the single message bit m is repeated n times

Practical Channel Coding Techniques

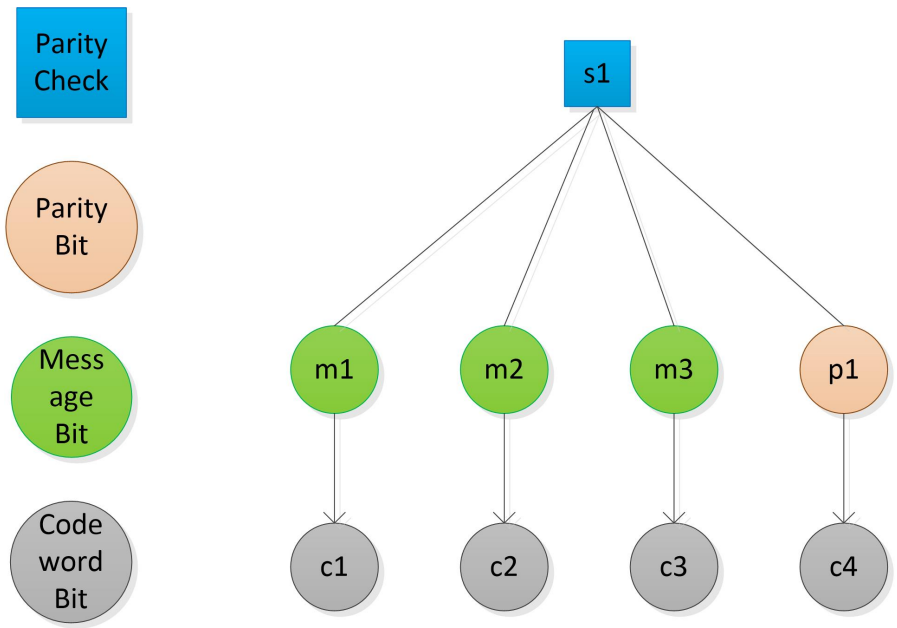
Coding Scheme 2: Single Parity Check (SPC) Code

- Encoding scheme:
  1. Take a block of $n-1 \geq 1$ bits as input.
  2. Add an extra n^{th} bit, called the parity check bit, which is 0 if the input block of n bits has even number of 1's, and it is 1 otherwise
  3. Transmit the resultant n bit long codeword
- Let $\{m_1, m_2, \ldots, m_{n-1}\}$ be the input block. The parity bit added is given as $p_n = \left(\sum_{k=1}^{n-1} m_k\right)_{mod-2}$
- Rate of this code is $r = \frac{n-1}{n}$.
  - $\rightarrow$ Rate r is the ratio of the number of information bits to total number of encoded bits

Practical Channel Coding Techniques

Single Parity Check (SPC) Code

- Notations:
  - $\{m_k\}$: info bits,
  - $\{p_k\}$: parity bits,
  - $\{s_k\}$: check bits,
  - $\{c_k\}$: codeword bits
  - If $\mathbf{c} \stackrel{\text{def}}{=} [c_1, c_2, \ldots, c_n]^T$ is an SPC codeword, $s_1 = \sum_{i=1}^{n} c_i$ has to be zero, where this sum is modulo two.



Practical Channel Coding Techniques

Single Parity Check (SPC) Code

- Decoding scheme:
  1. Take a block of n bits as input.
  2. Compute the modulot-two sum of these bits
  3. If this sum is zero, determine that zero or an even number of bit errors have occurred. If nonzero, determine that one or an odd number of bit errors have occurred
- Rate $(n-1)/n$ SPC code can detect one bit of error, but it cannot correct it

Generator and Parity Check Matrices

for the Rate $r = 4/5$ SPC Code

- What are **G** and **H** matrices for the Rate $r = 4/5$ SPC Code?

Generator and Parity Check Matrices

for the Rate $r = (n-1)/n$ SPC Code

- For the rate $(n-1)/n$ SPC code ($k = n-1, u = 1$):
  - ▷ The parity check matrix is as follows:
    - → $\mathbf{H} = [1, 1, \ldots, 1]$, i.e., it is just a matrix (a horizontal vector) of size $1 \times n$ consisting of all ones
    - → $\mathbf{H}\,\mathbf{c}$ in Eq. 1 denotes the single parity check (whose output is a 1×1 zero-valued bit)
    - → The coded bits $\mathbf{c}$, by definition of the SPC code, satisfy this check, hence $\mathbf{H}\,\mathbf{c} = 0$. In simple terms, this shows that all bits of c add up to 0 (modulo-two)
  - ▷ The generator matrix is as follows:
    - → $\mathbf{G} = \begin{bmatrix} \mathbf{I}_{k\times k} \\ 1_{1\times k} \end{bmatrix}$
    - → The Identity matrix $\mathbf{I}_{k\times k}$ populates first $k = n-1$ rows and all k columns of $\mathbf{G}$; the last row is a (horizontal) vector of size $1 \times k$ of all ones. The overall size of $\mathbf{G}$ is, thus, $(k+1) \times k = n \times (n-1)$, as required
    - → This ensures that in the product $\mathbf{G}\,\mathbf{m}$, the first $k = n-1$ bits are directly the message bits, and the last bit (i.e., the parity bit) is the modulo-two sum of all $n-1$ message bits
- Remarks:
  - → Practice the operation of $\mathbf{G}$ and $\mathbf{H}$ for the SPC and the repetition codes, for some example values of n, k and u
  - → Observe a duality between $\mathbf{G}$ and $\mathbf{H}$ matrices of the SPC and the repetition codes ($\mathbf{G}$ of one looks like $\mathbf{H}$ of the other, and vice versa)

Practical Channel Coding Techniques

Coding Scheme 3: Product Codes

- k (number of information bits that get encoded) is a perfect square number $(4, 9, 16, 25, \ldots)$
- $n = \left(\sqrt{k} + 1\right)^2 = k + 1 + 2\sqrt{k}$ is the number of encoded bits
- Encoding Strategy:
  - Place the information bits in $\sqrt{k} \times \sqrt{k}$ array
  - Encode each row with an SPC
  - Encode each column with an SPC

Practical Channel Coding Techniques

Product Codes

- k (number of information bits that get encoded) is a perfect square number $(4, 9, 16, 25, \ldots)$
- $n = \left(\sqrt{k} + 1\right)^2 = k + 1 + 2\sqrt{k}$ is the number of encoded bits
- Encoding Strategy:
  - Place the information bits in $\sqrt{k} \times \sqrt{k}$ array
  - Encode each row with an SPC
  - Encode each column with an SPC

Practical Channel Coding Techniques

An Example ($n = 9, k = 4, u = 5, r = 4/9$) Product Code

- $p_1 = m_1 + m_2; p_2 = m_3 + m_4$
- $p_3 = m_1 + m_3; p_4 = m_2 + m_4$
- $p_5 = p_1 + p_2 = p_3 + p_4 = m_1 + m_2 + m_3 + m_4$

m1	m2	p1
m3	m4	p2
p3	p4	p5

- A homework assignment: determine the generator matrix **G** of size 9×4 and the parity check matrix of size 5×9 for this product code.

An Iterative Decoding Scheme

for the Product Code received over a BEC channel

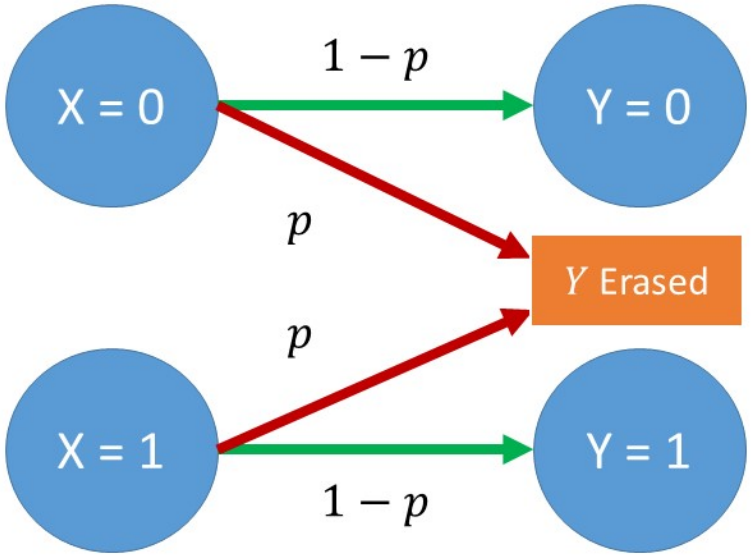
- For understanding how the product codes can be decoded, we will take the following example of $(9, 4)$ product code

m1= 1	m2= 0	p1= 1
m3= 1	m4= 1	p2= 0
p3= 0	p4= 1	p5= 1

An Iterative Decoding Scheme

for the Product Code received over a BEC channel

- The model of the BEC channel is as follows:



Iterative Methods

An Example Decoding for the Product Code

- We consider that the transmitted codeword is received over the BEC channel. In this example, as shown below, we take that as many as five bits are erased

Transmitted Codeword

c

1	0	1
1	1	0
0	1	1

BEC →

Received Word with Erasures

r

?	?	1
1	?	0
0	?	?

Iterative Methods

An Example Decoding for the Product Code

- We now consider a simple iterative method that performs the SPC decoding on each row, followed by each column. This counts as one iteration. The method iterates over multiple iterations until either the transmitted codeword is recovered or the iterations do not make any changes
  - Since we are considering the BEC channel, and there are three bits in each row or column, the SPC is able to correct a row or column if it has no more than one bit erased
  - Work the example on the next slide on your own to improve the understanding

Iterative Methods

An Example Decoding for the Product Code

Received Word
r

?	?	1
1	?	0
0	?	?

Iterative Methods

An Example Decoding for the Product Code

Received Word
r

?	?	1
1	?	0
0	?	?

→

Row Decoding:
Iteration 1

?	?	1
1	1	0
0	?	?

Iterative Methods

An Example Decoding for the Product Code

Received Word
r

?	?	1
1	?	0
0	?	?

→

Row Decoding:
Iteration 1

?	?	1
1	1	0
0	?	?

→

Column Decoding:
Iteration 1

1	?	1
1	1	0
0	?	1

Iterative Methods

An Example Decoding for the Product Code

Received Word
r

?	?	1
1	?	0
0	?	?

Row Decoding:
Iteration 1

?	?	1
1	1	0
0	?	?

Column Decoding:
Iteration 1

1	?	1
1	1	0
0	?	1

Row Decoding:
Iteration 2

1	0	1
1	1	0
0	1	1

Iterative Methods

An Example Decoding for the Product Code

Received Word
r

?	?	1
1	?	0
0	?	?

Row Decoding:
Iteration 1

?	?	1
1	1	0
0	?	?

Column Decoding:
Iteration 1

1	?	1
1	1	0
0	?	1

Row Decoding:
Iteration 2

1	0	1
1	1	0
0	1	1

Decoding Successful
All erasures are un-erased

1	0	1
1	1	0
0	1	1

Practical Channel Coding Techniques

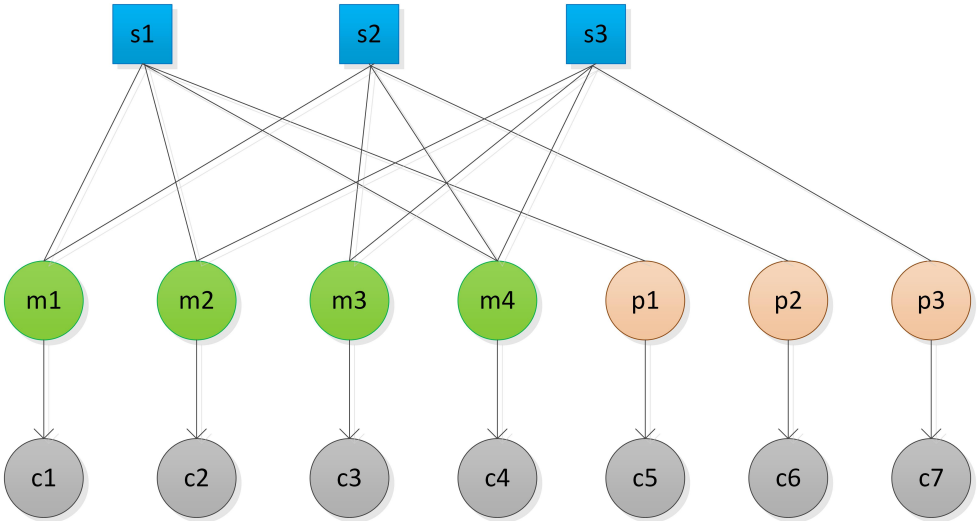
Coding Scheme 4: Rectangular Parity Codes

- Product Codes mentioned above are one of a family of codes known as Rectangular Parity Codes (RPC)
  - → The code defined on the prior two slides can be considered a Square Parity Code
- See Fig. 6-3 of MIT Lecture 6 (posted on Google Classroom) for an example Rectangular Parity Code with three rows and five columns ($n = 14, k = 8, u = 6, r = 4/7$)

Practical Channel Coding Techniques

Coding Scheme 5: Hamming Code

- Hamming ($n = 7, k = 4, u = 3, r = 4/7$) code is essentially three SPC codes:



$$s_1 = m_1 + m_2 + m_4 + p_1$$
$$s_2 = m_1 + m_3 + m_4 + p_2$$
$$s_3 = m_2 + m_3 + m_4 + p_3$$

- A homework assignment: determine the generator matrix $\mathbf{G}$ of size 7×4 and the parity check matrix of size 3×7 for the Hamming code. Also answer the following questions:
  - → The columns of $\mathbf{H}$ together make an interesting pattern. What is that?
  - → Given $\mathbf{H}$, the matrix $\mathbf{G}$ can be determined (and vice versa). How?

Practical Channel Coding Techniques

Hamming Code versus Rectangular Parity Code (RPC)

- The parameters n, k, u and r of the Hamming Code matches with that of the RPC code
- Furthermore, $d_H^{\min}$ for both these codes is 3 bits, i.e., $t_c = 1$ bit
- However, Hamming codes are more efficient. Each of the three parity bits does the same amount of "work" (in protecting three message bits). In the RPC codes, some parity bits do more "work" and the others less

Iterative Methods

Do they also reside in the human brain?

- Can you read the following paragraph with scrambled letters of each word where the first and the last letters are at their correct positions?

Iterative Methods

Do they also reside in the human brain?

- Can you read the following paragraph with scrambled letters of each word where the first and the last letters are at their correct positions?
  - *Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.*

Iterative Methods

Do they also reside in the human brain?

- Can you read the following paragraph with scrambled letters of each word where the first and the last letters are at their correct positions?
  - *Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.*
- If yes, maybe some sort of iterative decoding and estimation is at play in your brain

Iterative Methods

Do they also reside in the human brain?

- Can you read the following paragraph with scrambled letters of each word where the first and the last letters are at their correct positions?
  - *Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.*
- If yes, maybe some sort of iterative decoding and estimation is at play in your brain
- However, this does not work all the time. Try reading the following where the words are scrambled in the same manner as above:

Iterative Methods

Do they also reside in the human brain?

- Can you read the following paragraph with scrambled letters of each word where the first and the last letters are at their correct positions?
  - *Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.*
- If yes, maybe some sort of iterative decoding and estimation is at play in your brain
- However, this does not work all the time. Try reading the following where the words are scrambled in the same manner as above:
  - *In the Vcraiiton are, a levloy eamlred geren, pirlaalty frmoueltad form asirnec, was uesd in fcaibrs and ppaluor falrol hresesdeads.*

Iterative Methods

Do they also reside in the human brain?

- Can you read the following paragraph with scrambled letters of each word where the first and the last letters are at their correct positions?
  - *Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.*
- If yes, maybe some sort of iterative decoding and estimation is at play in your brain
- However, this does not work all the time. Try reading the following where the words are scrambled in the same manner as above:
  - *In the Vcraiiton are, a levloy eamlred geren, pirlaalty frmoueltad form asirnec, was uesd in fcaibrs and ppaluor falrol hresesdeads.*
- See an interesting discussion at https://www.dictionary.com/e/typoglycemia/

Iterative Decoding on a Graph

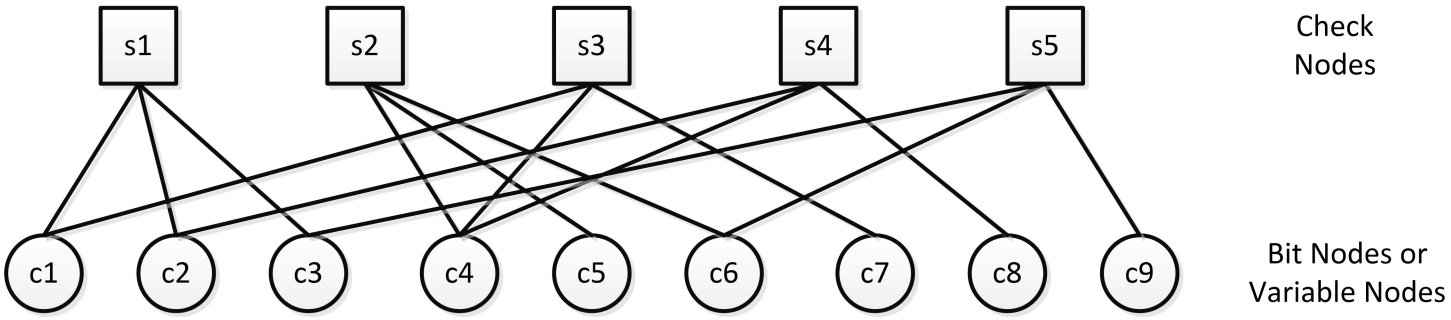
- Although the row $\rightarrow$ column iterations provide a good basis for the iterative decoding, this viewpoint can further be enhanced by considering the decoding on a "graph" representation of the parity check matrix $\mathbf{H}$
- As noted earlier, the parity check equations for $(9, 4)$ product code can be written as a matrix product:

$$\mathbf{H}_{(N-K)\times N} \cdot \mathbf{c}_{N\times 1} = \mathbf{0}_{N-K\times 1}$$

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Tanner Graph

- This parity check matrix **H** can be represented by a Tanner Graph, which is a Bipartite Graph:
  - This has two types of nodes:
    1. A total of $N - K$ check nodes or CNs, and
    2. A total of N coded bit nodes or variable nodes or VNs
  - Nodes of one type are never connected to the nodes of the same type. Resulting binary partition of the nodes of graph is the reason why this is called Bipartite Graph
  - Degree of a node is the number of the edges connected to that node. The degree of a CN is denoted as d_c and the degree of a VN is denoted as d_v.



Decoding on Tanner Graph

for the BEC(p) Channel

Decoding can be better visualized and implemented using the Tanner graph framework:

1. Set the iteration index $t = 0$
2. Load each variable node VN i with the observed bits/erasures at the output of the channel
3. Message from i^{th} VN to j^{th} CN: described on the next slide
   - ▷ Also calculate the estimate $\hat{\mathbf{c}}^t$ of the decoded codeword
4. Message from j^{th} CN to i^{th} VN: described on the next slide
5. Are all erasures corrected or no more corrections possible? OR is $t > t_{\max}$ (where $t_{\max}$ is the maximum number of iterations allowed)?
   - ▷ If yes, terminate and obtain the decoded codeword as $\hat{\mathbf{c}}^t$
   - ▷ If not, increment the iteration index (i.e., $t \leftarrow t + 1$) and go to Step 3

Decoding on Tanner Graph

A Note on the Tanner Graph Implementation

- In Step 3: the algorithm has to visit each VN i, where $1 \leq i \leq N$, and for each visited VN, it has to
  - compute the message that it has to send to all of the CNs $\{j\}$ that are connected to it, AND
  - calculate the estimate $\hat{c}_i$ of the ith bit of the decoded codeword
- In Step 4: the algorithm has to visit each CN j, where $1 \leq j \leq U = N - K$ is the number of parity bits, and for each visited CN, it has to
  - compute the message that it has to send to all of the VNs $\{i\}$ that are connected to it
- The above is purely a software implementation of the bi-partite graph given the parity check $\mathbf{H}$ matrix, and it is reuseable for different channel types and different types (hard-decision or the soft-decision) of decoding

Message Passing Decoding on Tanner Graph

for BEC: VN $\rightarrow$ CN

2. Message from i^{th} VN to j^{th} CN for $(t+1)^{th}$ iteration: each VN $1 \leq i \leq N$ sends j^{th} CN connected to it
   - ▷ **if** $d_v - 1$ messages from the CNs, other than j^{th} CN, received at i^{th} VN at t^{th} iteration and the received value over the channel at i^{th} VN are all erasures
     - $\rightarrow$ an erasure
   - ▷ **otherwise**
     - $\rightarrow$ the value of the non-erased message

Message Passing Decoding on Tanner Graph

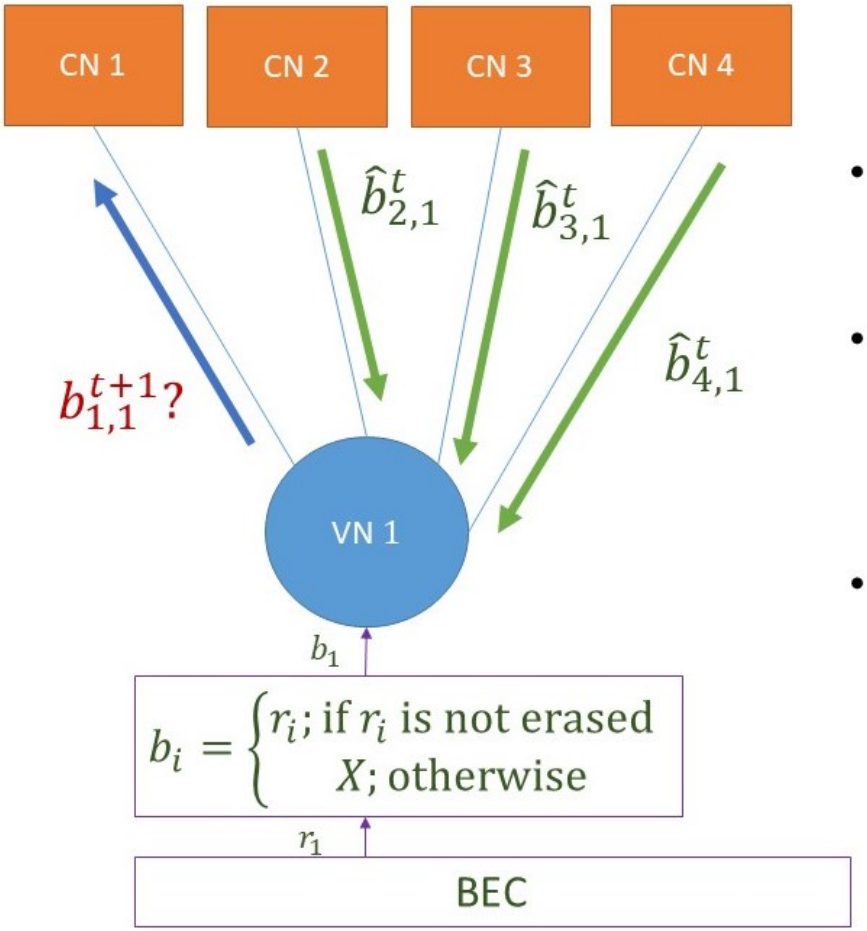
for BEC: VN $\rightarrow$ CN

2. Message from i^{th} VN to j^{th} CN for $(t+1)^{th}$ iteration: each VN $1 \leq i \leq N$ sends j^{th} CN connected to it
   - ▷ **if** $d_v - 1$ messages from the CNs, other than j^{th} CN, received at i^{th} VN at t^{th} iteration and the received value over the channel at i^{th} VN are all erasures
     - $\rightarrow$ an erasure
   - ▷ **otherwise**
     - $\rightarrow$ the value of the non-erased message

- A note: above holds for all iterations t except the first one $t = 0$. In the first iteration $(t = 0)$, each VN simply sends to the CNs the value it receives over the channel.

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here); for the BEC



- <u>An example connectivity in the bipartite graph:</u> $VN\ i = 1$ is connected to $CNs\ j = 2, j = 3$ and $j = 4$
- <u>Notations:</u> (t is the iteration index)
  - $b^t_{i,j}$: the bit message (either 0, 1, or erasure X) sent by $VN\ i$ to $CN\ j$
  - $\hat{b}^t_{j,i}$: the bit message sent by $CN\ j$ to $VN\ i$
- <u>Problem statement:</u> determine the value of $b^{t+1}_{1,1}$ that $VN\ 1$ sends to $CN\ 1$ in iteration $t+1$ given the values of $\hat{b}^t_{2,1}$, $\hat{b}^t_{3,1}$ and $\hat{b}^t_{4,1}$ received from CNs 2, 3 and 4 in iteration t

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here); for the BEC

- If one or multiple of $\hat{b}^t_{2,1}$, $\hat{b}^t_{3,1}$, $\hat{b}^t_{4,1}$ and b_1 are not erased, they are all the same value (either 0 or 1) (since a VN is a repetition code)

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here); for the BEC

- If one or multiple of $\hat{b}^t_{2,1}$, $\hat{b}^t_{3,1}$, $\hat{b}^t_{4,1}$ and b_1 are not erased, they are all the same value (either 0 or 1) (since a VN is a repetition code)
  - In this case, set $b^{t+1}_{1,1}$ to this common unerased value
- Else
  - Set $b^{t+1}_{1,1}$ as an erasure X

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here); a simple change for the BSC

- Since each VN is a repetition code, the decoding follows that of the repetition code in the BSC

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here); a simple change for the BSC

- Since each VN is a repetition code, the decoding follows that of the repetition code in the BSC
- Set $b_{1,1}^{t+1}$ as the majority-vote on $\hat{b}_{2,1}^{t}$, $\hat{b}_{3,1}^{t}$, $\hat{b}_{4,1}^{t}$ and b_1

Message Passing Decoding on Tanner Graph

for the BEC: CN $\rightarrow$ VN

3. Message from j^{th} CN to i^{th} VN at t^{th} iteration: each CN $1 \leq j \leq N - K$ sends to i^{th} VN connected to it
   - ▷ **if** <u>any one</u> of $d_c - 1$ messages received at this CN at t^{th} iteration from all the VNs connected to it except i^{th} VN is an erasure
     - $\rightarrow$ an erasure
   - ▷ **otherwise**
     - $\rightarrow$ the modulo-two sum of $d_c - 1$ messages from all the VNs connected to j^{th} CN, except i^{th} VN

Message Passing Decoding on Tanner Graph

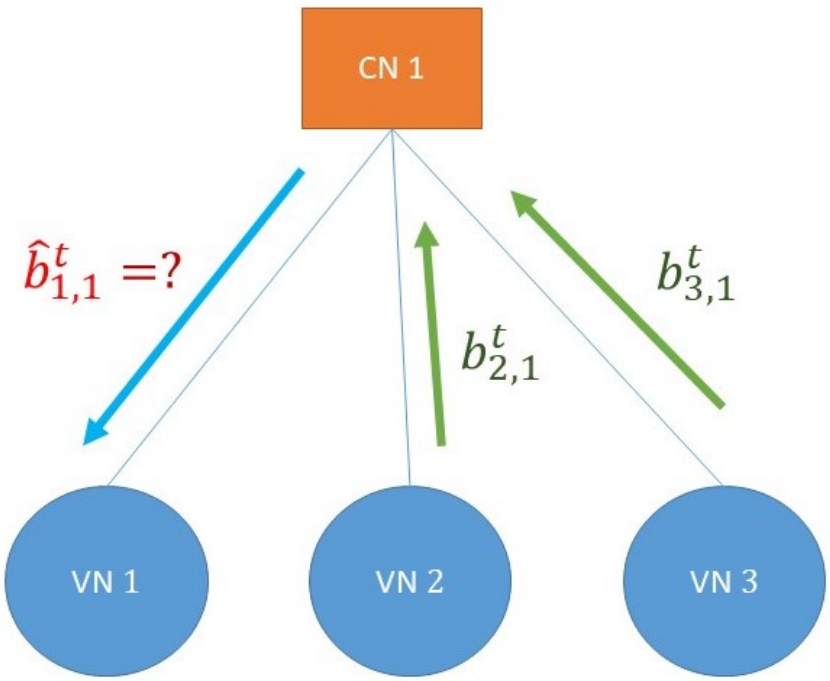
for the BEC: CN $\rightarrow$ VN

3. Message from j^{th} CN to i^{th} VN at t^{th} iteration: each CN $1 \leq j \leq N - K$ sends to i^{th} VN connected to it
   - ▷ **if** any one of $d_c - 1$ messages received at this CN at t^{th} iteration from all the VNs connected to it except i^{th} VN is an erasure
     - $\rightarrow$ an erasure
   - ▷ **otherwise**
     - $\rightarrow$ the modulo-two sum of $d_c - 1$ messages from all the VNs connected to j^{th} CN, except i^{th} VN

- A note: in both VN $\rightarrow$ CN and CN $\rightarrow$ VN message passing, the message sent to a node never depends on the message received from that node.
- To do so would imply positive feedback (or cyclic feedback) which is never desirable

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)



- An example of the connectivity in the bipartite graph: CN 1 is connected to VNs 1, 2 and 3
- Problem statement: determine the value of $\hat{b}^t_{1,1}$ that CN 1 sends to VN 1 in iteration t given the values of $b^t_{2,1}$ and $b^t_{3,1}$ that it has received from VNs 2 and 3 in iteration t

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- The following is the appropriate strategy since a CN is an SPC code

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- The following is the appropriate strategy since a CN is an SPC code
- If any one or multiple of $b^t_{2,1}$ and $b^t_{3,1}$, $\hat{b}^t_{4,1}$ and b_1 are erased

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- The following is the appropriate strategy since a CN is an SPC code
- If any one or multiple of $b_{2,1}^t$ and $b_{3,1}^t$, $\hat{b}_{4,1}^t$ and b_1 are erased
  - Set $\hat{b}_{1,1}^{t+1}$ to an erasure X (since the parity check equation cannot be computed given the presence of X)
- Else
  - Set $\hat{b}_{1,1}^{t+1}$ as modulo-two sum $b_{2,1}^t + b_{3,1}^t$ (this satisfies the parity check constraint at CN j)

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here); a simple change for the BSC

- Since each CN is an SPC code, the decoding follows that of the SPC code in the BSC

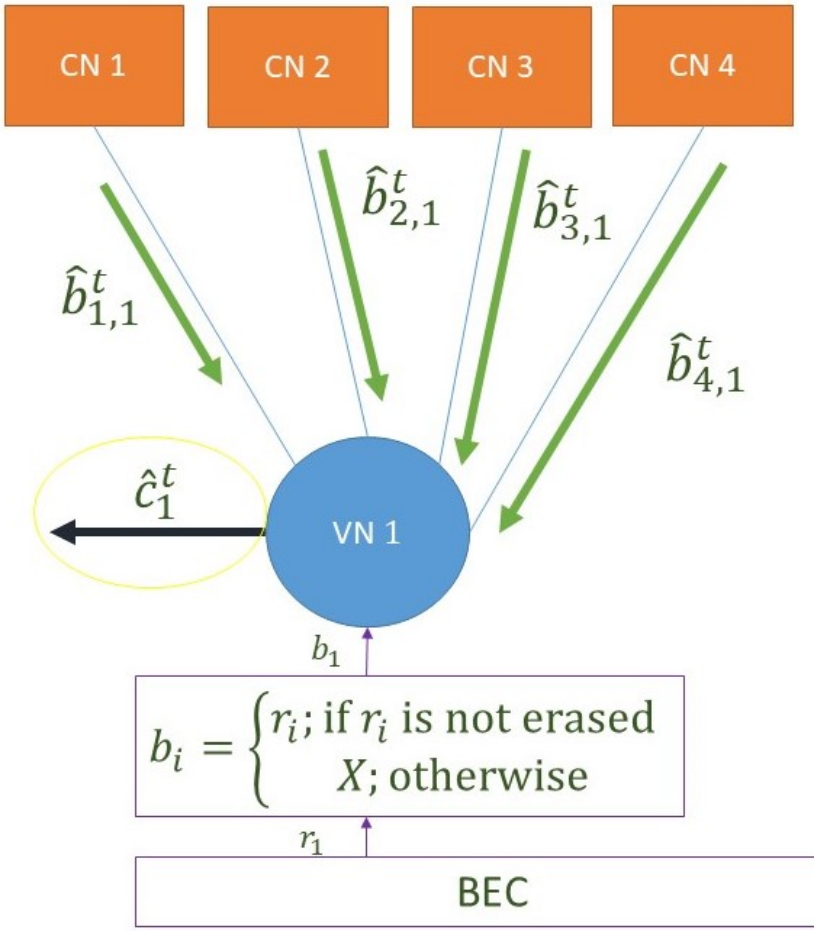
Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here); a simple change for the BSC

- Since each CN is an SPC code, the decoding follows that of the SPC code in the BSC
- Set $\hat{b}_{1,1}^{t+1}$ as modulo-two sum $b_{2,1}^{t} + b_{3,1}^{t}$ (this satisfies the parity check constraint at CN j)

Bit Decision for VN i at t^{th} Iteration

($i = 1$ in the hypothetical example considered here)



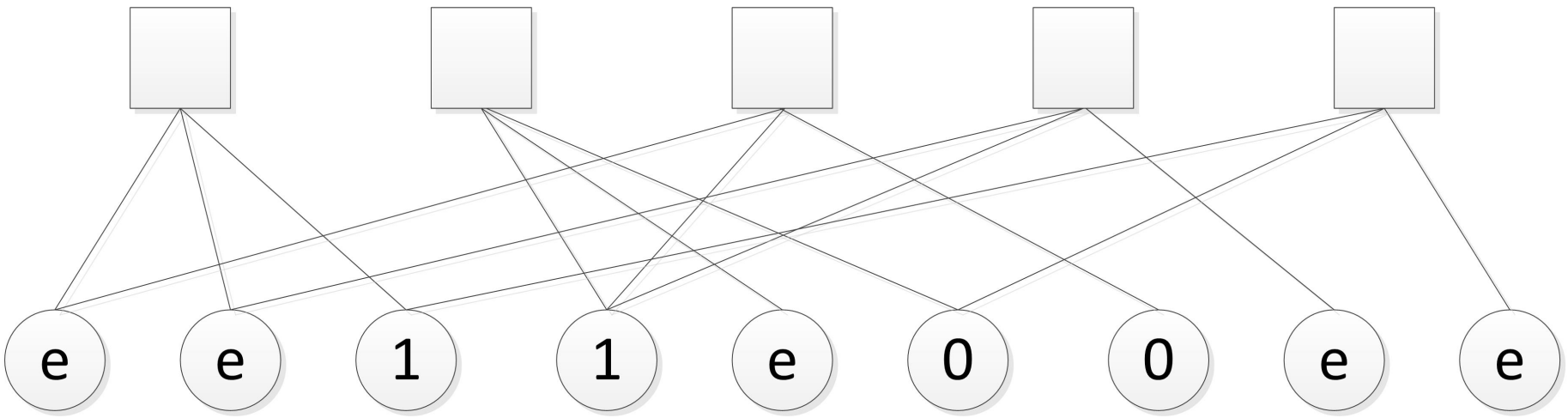
Bit Decision for VN i at t^{th} Iteration

($i = 1$ in the hypothetical example considered here)

- The bit decisions are not part of the looping process — there is no concern of positive feedback mentioned earlier
- Therefore, set $\hat{c}_1^t$ as the majority-vote on $\hat{b}_{1,1}^t$, $\hat{b}_{2,1}^t$, $\hat{b}_{3,1}^t$, $\hat{b}_{4,1}^t$ and b_1

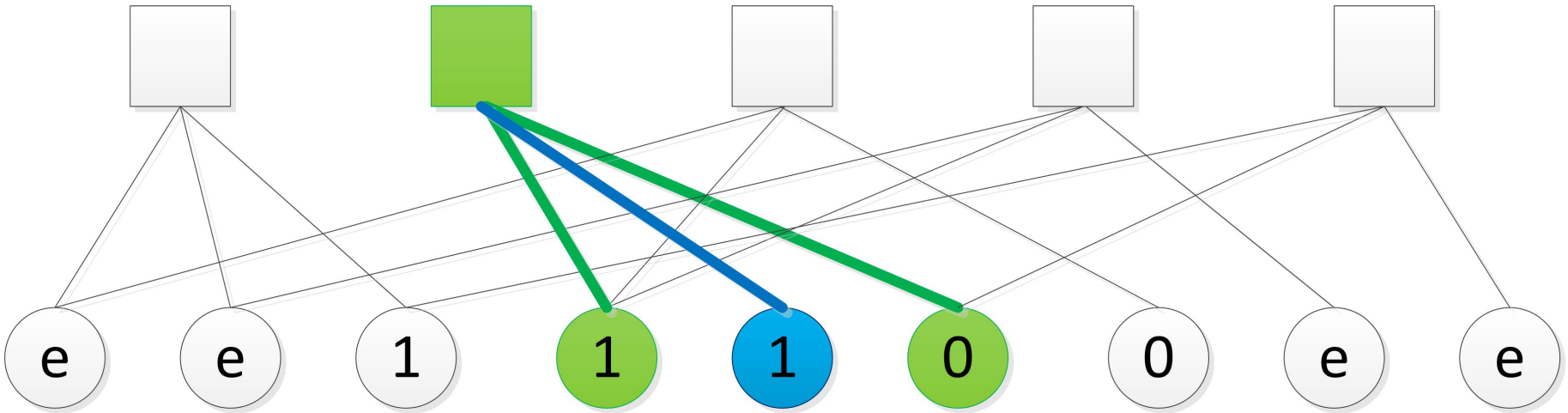
Decoding on Tanner Graph: An example for the BEC

Step 1: Load the Variable Nodes with the Received Codeword Bits



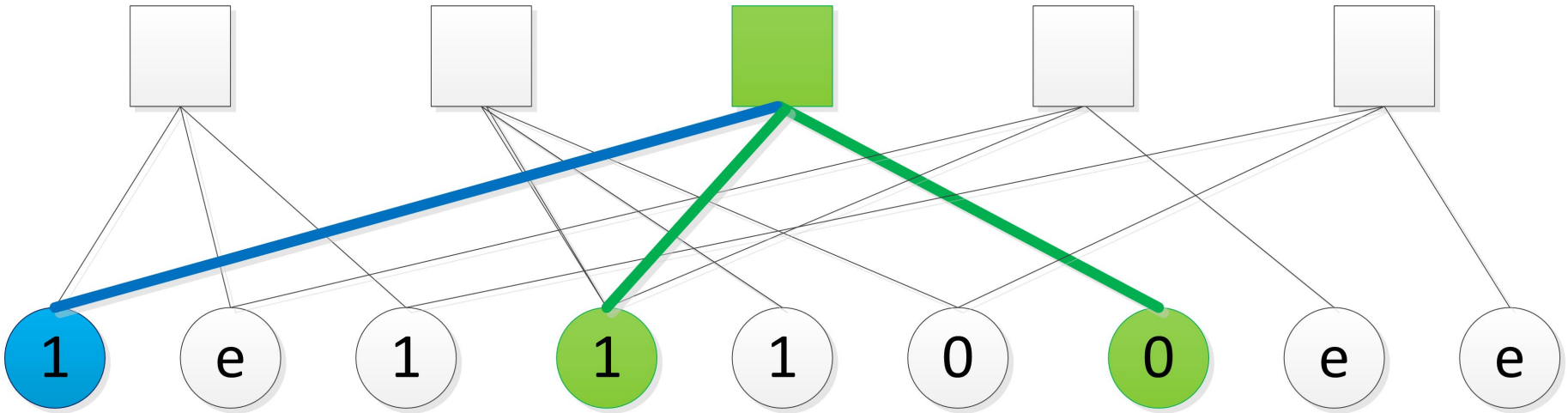
Message Passing on Tanner Graph (Steps 3 and 4): An example for the BEC

Iteration 1, CN 2



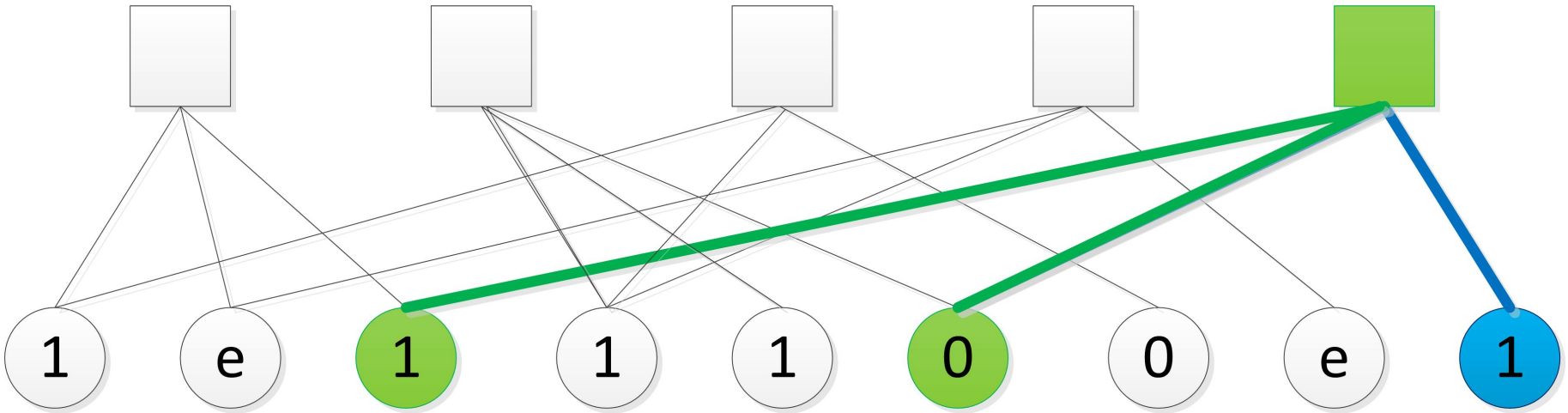
Message Passing on Tanner Graph (Steps 3 and 4): An example for the BEC

Iteration 1, CN 3



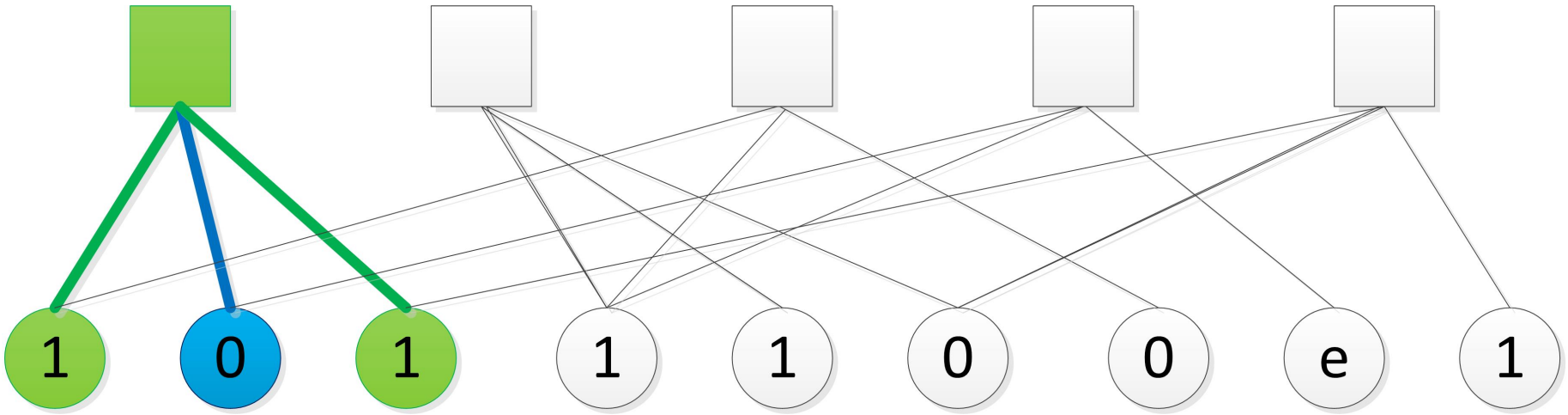
Message Passing on Tanner Graph (Steps 3 and 4): An example for the BEC

Iteration 1, CN 5



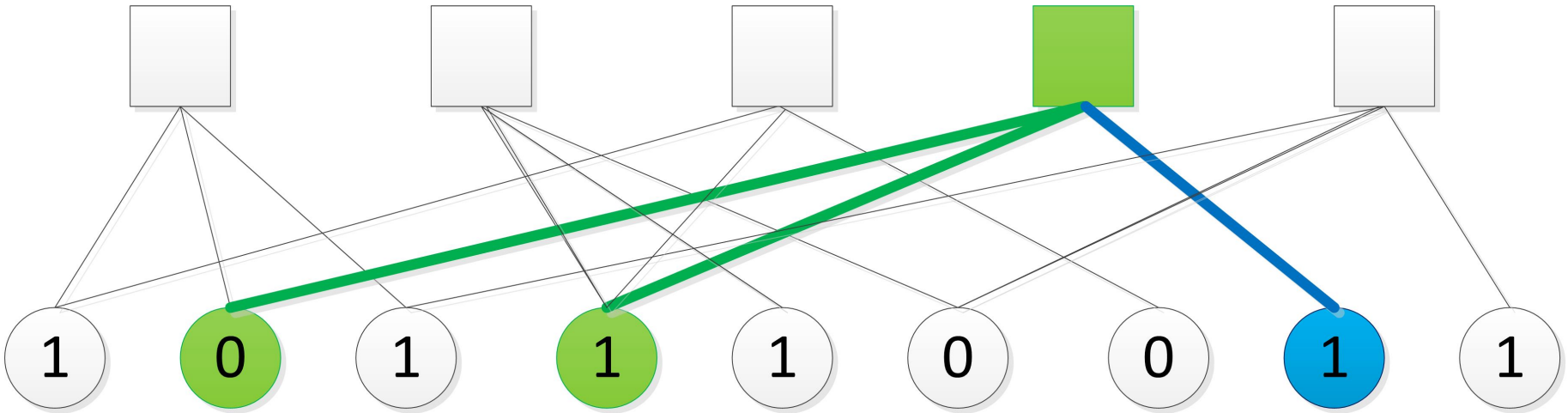
Message Passing on Tanner Graph (Steps 3 and 4): An example for the BEC

Iteration 2, CN 1



Message Passing on Tanner Graph (Steps 3 and 4): An example for the BEC

Iteration 3, CN 4



Hard versus Soft Decision Decoding

- Suppose for a rate 1/5 repetition code, each 1 is sent using +1 v and each 0 is sent using −1 v (this is known as BPSK or antipodal modulation)
- Suppose that a codeword $\mathbf{c} = [1, 1, 1, 1, 1]$ is sent, and the receiver, due to the channel noise, receives it as $\mathbf{r} = \mathbf{c} + \mathbf{n} = [1.1, 0.9, -0.01, -0.02, -0.05]$
- BPSK demodulator decides that a bit 1 was sent if the received voltage is positive-valued, and bit 0 was sent if the received voltage is negative-valued

Hard versus Soft Decision Decoding

- Hard Decision Decoding:
  - → Convert each received sample into a bit (make a hard decision): $\mathbf{y} = [1, 1, 0, 0, 0]$
  - → Perform majority voting decoding to decide that bit 0 was sent.
    - ▷ This is because the hard decisions $\mathbf{y} = [1, 1, 0, 0, 0]$ is nearer in Hamming distance to $[0, 0, 0, 0, 0]$ than to $[1, 1, 1, 1, 1]$, and so the wrong codeword $[0, 0, 0, 0, 0]$ will be chosen
    - ▷ Wrong decision, results in a bit error
- Soft Decision Decoding:
  - → Will recognize that the first two bits are strongly received. Next three bits are not very strong, and therefore, place less weight on them.
  - → In principle, the soft decision decoding works on the Bayesian Theorem, whereas the hard decision decoding is an ad-hoc method that is preferred because it may be computationally simpler to implement

Hard versus Soft Decision Decoding

- Note that **r**, i.e., the vector of soft decisions, contains more information than **y**, vector of hard decisions
  - $\rightarrow$ It is always better to not throw away information when decoding
- Hard Decision Decoding chooses the codeword acording to the minimum Hamming Distance. Soft Decision Decoding instead chooses the codeword that minimizes the Euclidean Distance.
- In general, soft decision decoding improves the performance by ≈ 2 dB.

Soft-Decision Decoding on Tanner Graph

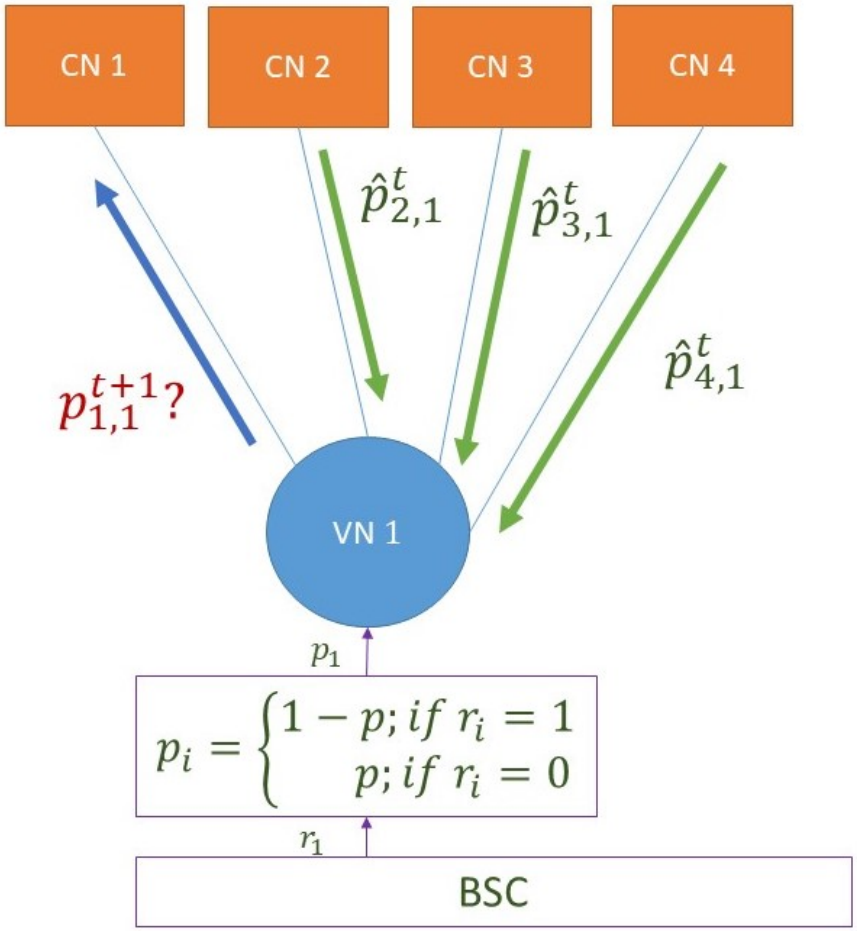
for the BSC(p) Channel

Decoding can be better visualized and implemented using the Tanner graph framework:

1. Set the iteration index $t = 0$
2. Load each variable node VN i with the ~~observed bits/erasures at the output of the channel~~ probability p_i = the conditional probability that VN i is 1 given the received bit r_i
   - For the BSC p, if $r_i = 1$, $p_i = 1 - p$; else $p_i = p$
3. Message from i^{th} VN to j^{th} CN: described on the next slide
   - Also calculate the estimate $\hat{\mathbf{c}}^t$ of the decoded codeword
4. Message from j^{th} CN to i^{th} VN: described on the next slide
5. ~~Are all erasures corrected or no more corrections possible?~~ is the estimated decoded codeword $\hat{\mathbf{c}}^t$ different from the estimate $\hat{\mathbf{c}}^{t-1}$ in the prior iteration? OR is $t > t_{\max}$ (where $t_{\max}$ is the maximum number of iterations allowed)?
   - If yes, terminate and obtain the decoded codeword as $\hat{\mathbf{c}}^t$
   - If not, increment the iteration index (i.e., $t \leftarrow t + 1$) and go to Step 3

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)



- <u>An example connectivity in the bipartite graph:</u> $VN\ i = 1$ is connected to $CNs\ j = 2, j = 3$ and $j = 4$
- <u>Notations:</u> (t is the iteration index)
  - $p^t_{i,j}$: the probability that $VN\ i$ is equal to 1 sent by $VN\ i$ to $CN\ j$
  - $\hat{p}^t_{j,i}$: the probability that $VN\ i$ is equal to 1 sent by $CN\ j$ to $VN\ i$
  - p_i: the probability that $VN\ i$ is equal to 1 *given* the received bit r_i at the BSC output
  - p: the cross-over probability of the BSC, assumed to be known
- <u>Problem statement:</u> determine the value of $p^{t+1}_{1,1}$ that $VN\ 1$ sends to $CN\ 1$ in iteration $t+1$ given the values of $\hat{p}^t_{2,1}$, $\hat{p}^t_{3,1}$ and $\hat{p}^t_{4,1}$ received from CNs 2, 3 and 4 in iteration t

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- Since each VN is a repetition code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- Since each VN is a repetition code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 1 equals 1 at CN 2; AND

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- Since each VN is a repetition code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 1 equals 1 at CN 2; AND
  - VN 1 equals 1 at CN 3; AND

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- Since each VN is a repetition code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 1 equals 1 at CN 2; AND
  - VN 1 equals 1 at CN 3; AND
  - VN 1 equals 1 at CN 4; AND

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- Since each VN is a repetition code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 1 equals 1 at CN 2; AND
  - VN 1 equals 1 at CN 3; AND
  - VN 1 equals 1 at CN 4; AND
  - VN 1 equals 1 given r_1 at the BSC output

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- Since each VN is a repetition code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 1 equals 1 at CN 2; AND
  - VN 1 equals 1 at CN 3; AND
  - VN 1 equals 1 at CN 4; AND
  - VN 1 equals 1 given r_1 at the BSC output
- Assuming that the above three individual events are independent, the probability that VN 1 equals 1 given the values of p_1, $\hat{p}^t_{2,1}$, $\hat{p}^t_{3,1}$ and $\hat{p}^t_{4,1}$ is simply $p^{t+1}_{1,1} = \alpha\, p_1\, \hat{p}^t_{2,1}\, \hat{p}^t_{3,1}\, \hat{p}^t_{4,1}$, where α is a normalization constant

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- Since each VN is a repetition code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 1 equals 1 at CN 2; AND
  - VN 1 equals 1 at CN 3; AND
  - VN 1 equals 1 at CN 4; AND
  - VN 1 equals 1 given r_1 at the BSC output
- Assuming that the above three individual events are independent, the probability that VN 1 equals 1 given the values of p_1, $\hat{p}^t_{2,1}$, $\hat{p}^t_{3,1}$ and $\hat{p}^t_{4,1}$ is simply $p^{t+1}_{1,1} = \alpha\, p_1\, \hat{p}^t_{2,1}\, \hat{p}^t_{3,1}\, \hat{p}^t_{4,1}$, where α is a normalization constant
- By the above reasoning, we can also derive the probability that VN 1 equals 0 given the values of p_1, $\hat{p}^t_{2,1}$, $\hat{p}^t_{3,1}$ and $\hat{p}^t_{4,1}$: this is simply the product $(1 - p^{t+1}_{1,1}) = \alpha\,(1 - p_1)\,(1 - \hat{p}^t_{2,1})\,(1 - \hat{p}^t_{3,1})\,(1 - \hat{p}^t_{4,1})$

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- The likelihood ratio (odds in favor of VN 1 equal to 1) is therefore,

$$\lambda_{1,1}^{t+1} = \frac{p_{1,1}^{t+1}}{1 - p_{1,1}^{t+1}} = \frac{p_1}{1 - p_1} \times \frac{\hat{p}_{2,1}^{t}}{1 - \hat{p}_{2,1}^{t}} \times \frac{\hat{p}_{3,1}^{t}}{1 - \hat{p}_{3,1}^{t}} \times \frac{\hat{p}_{4,1}^{t}}{1 - \hat{p}_{4,1}^{t}}$$
$$= \lambda_1 \times \hat{\lambda}_{2,1}^{t} \times \hat{\lambda}_{3,1}^{t} \times \hat{\lambda}_{4,1}^{t}$$

Belief Generation on the Tanner Graph

VN i to CN j ($i = j = 1$ in the hypothetical example considered here)

- The likelihood ratio (odds in favor of VN 1 equal to 1) is therefore,

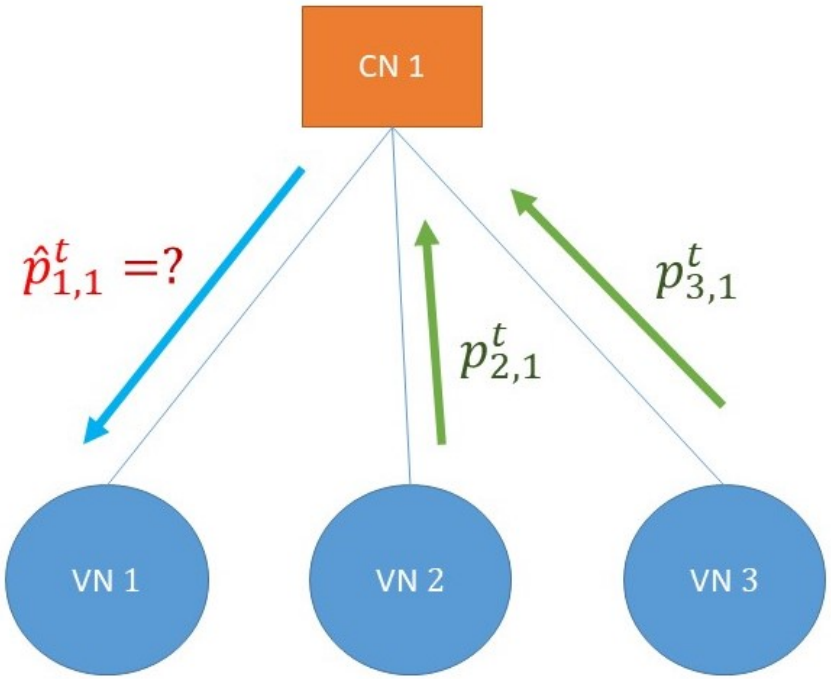
$$\lambda_{1,1}^{t+1} = \frac{p_{1,1}^{t+1}}{1 - p_{1,1}^{t+1}} = \frac{p_1}{1 - p_1} \times \frac{\hat{p}_{2,1}^{t}}{1 - \hat{p}_{2,1}^{t}} \times \frac{\hat{p}_{3,1}^{t}}{1 - \hat{p}_{3,1}^{t}} \times \frac{\hat{p}_{4,1}^{t}}{1 - \hat{p}_{4,1}^{t}}$$
$$= \lambda_1 \times \hat{\lambda}_{2,1}^{t} \times \hat{\lambda}_{3,1}^{t} \times \hat{\lambda}_{4,1}^{t}$$

- The log likelihood ratio $\lambda_{1,1}^{t+1,L} = \log(\lambda_{1,1}^{t+1})$ takes even a simpler form

$$\lambda_{1,1}^{t+1,L} = \lambda_1 + \hat{\lambda}_{2,1}^{t,L} + \hat{\lambda}_{3,1}^{t,L} + \hat{\lambda}_{4,1}^{t,L}$$

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)



- An example of the connectivity in the bipartite graph: CN 1 is connected to VNs 1, 2 and 3
- Problem statement: determine the value of $\hat{p}^t_{1,1}$ that CN 1 sends to VN 1 in iteration t given the values of $p^t_{2,1}$ and $p^t_{3,1}$ that it has received from VNs 2 and 3 in iteration t

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 2 equals 1 AND VN 3 equals 0, OR

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 2 equals 1 AND VN 3 equals 0, OR
  - VN 2 equals 0 AND VN 3 equals 0

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 2 equals 1 AND VN 3 equals 0, OR
  - VN 2 equals 0 AND VN 3 equals 0
  - The probability of the above equals $\hat{p}^t_{1,1} = \beta\left(p^t_{2,1}(1 - p^t_{3,1}) + (1 - p^t_{2,1})p^t_{3,1}\right)$, where β is a normalization constant

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 2 equals 1 AND VN 3 equals 0, OR
  - VN 2 equals 0 AND VN 3 equals 0
  - The probability of the above equals $\hat{p}^t_{1,1} = \beta\left(p^t_{2,1}(1 - p^t_{3,1}) + (1 - p^t_{2,1})p^t_{3,1}\right)$, where β is a normalization constant
- Similarly, the probability that VN 1 is equal to 0 equals the probability of the following joint event:

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 2 equals 1 AND VN 3 equals 0, OR
  - VN 2 equals 0 AND VN 3 equals 0
  - The probability of the above equals $\hat{p}^t_{1,1} = \beta\left(p^t_{2,1}(1 - p^t_{3,1}) + (1 - p^t_{2,1})p^t_{3,1}\right)$, where β is a normalization constant
- Similarly, the probability that VN 1 is equal to 0 equals the probability of the following joint event:
  - VN 2 equals 0 AND VN 3 equals 0, OR

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 2 equals 1 AND VN 3 equals 0, OR
  - VN 2 equals 0 AND VN 3 equals 0
  - The probability of the above equals $\hat{p}^t_{1,1} = \beta\left(p^t_{2,1}(1 - p^t_{3,1}) + (1 - p^t_{2,1})p^t_{3,1}\right)$, where β is a normalization constant
- Similarly, the probability that VN 1 is equal to 0 equals the probability of the following joint event:
  - VN 2 equals 0 AND VN 3 equals 0, OR
  - VN 2 equals 1 AND VN 3 equals 1

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 2 equals 1 AND VN 3 equals 0, OR
  - VN 2 equals 0 AND VN 3 equals 0
  - The probability of the above equals $\hat{p}^t_{1,1} = \beta\left(p^t_{2,1}(1 - p^t_{3,1}) + (1 - p^t_{2,1})p^t_{3,1}\right)$, where β is a normalization constant
- Similarly, the probability that VN 1 is equal to 0 equals the probability of the following joint event:
  - VN 2 equals 0 AND VN 3 equals 0, OR
  - VN 2 equals 1 AND VN 3 equals 1
  - The probability of the above equals $(1 - \hat{p}^t_{1,1}) = \beta\left((1 - p^t_{2,1})(1 - p^t_{3,1}) + p^t_{2,1}p^t_{3,1}\right)$

Belief Generation on the Tanner Graph

CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- Recall each CN is an SPC code, the probability that VN 1 is equal to 1 equals the probability of the following joint event:
  - VN 2 equals 1 AND VN 3 equals 0, OR
  - VN 2 equals 0 AND VN 3 equals 0
  - The probability of the above equals $\hat{p}^t_{1,1} = \beta\left(p^t_{2,1}(1 - p^t_{3,1}) + (1 - p^t_{2,1})p^t_{3,1}\right)$, where β is a normalization constant
- Similarly, the probability that VN 1 is equal to 0 equals the probability of the following joint event:
  - VN 2 equals 0 AND VN 3 equals 0, OR
  - VN 2 equals 1 AND VN 3 equals 1
  - The probability of the above equals $(1 - \hat{p}^t_{1,1}) = \beta\left((1 - p^t_{2,1})(1 - p^t_{3,1}) + p^t_{2,1}p^t_{3,1}\right)$

Belief Generation on the Tanner Graph

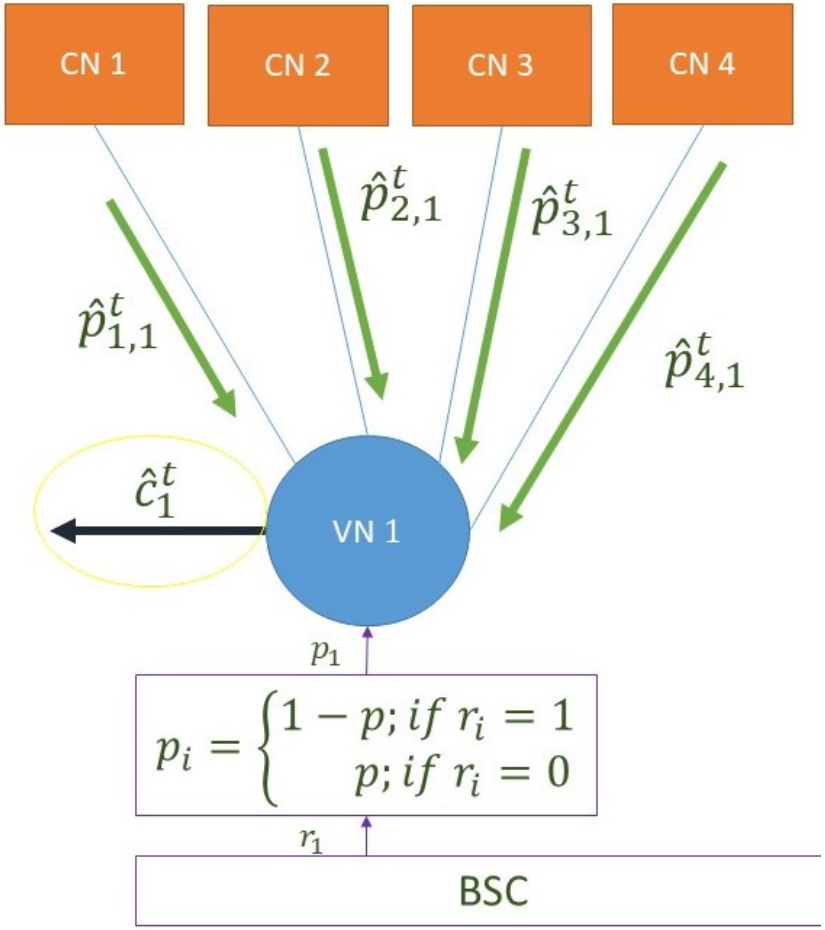
CN j to VN i ($i = j = 1$ in the hypothetical example considered here)

- The likelihood ratio (odds in favor of VN 1 equal to 1) is therefore,

$$\lambda_1 = \frac{\hat{p}^t_{1,1}}{1 - \hat{p}^t_{1,1}} = \frac{p^t_{2,1}(1 - p^t_{3,1}) + (1 - p^t_{2,1})p^t_{3,1}}{(1 - p^t_{2,1})(1 - p^t_{3,1}) + p^t_{2,1}p^t_{3,1}}$$

Bit Decision for VN i at t^{th} Iteration

($i = 1$ in the hypothetical example considered here)



Bit Decision for VN i at t^{th} Iteration

($i = 1$ in the hypothetical example considered here)

- The bit decisions are not part of the looping process — there is no concern of positive feedback mentioned earlier
- Therefore, the likelihood ratio (odds in favor of VN 1 equal to 1) is calculated using all the information available from the CNs and from the channel

$$\lambda_1^t = \frac{p_1}{1-p_1} \times \frac{\hat{p}_{1,1}^t}{1-\hat{p}_{1,1}^t} \times \frac{\hat{p}_{2,1}^t}{1-\hat{p}_{2,1}^t} \times \frac{\hat{p}_{3,1}^t}{1-\hat{p}_{3,1}^t} \times \frac{\hat{p}_{4,1}^t}{1-\hat{p}_{4,1}^t}$$
$$= \lambda_1 \times \hat{\lambda}_{1,1}^t \times \hat{\lambda}_{2,1}^t \times \hat{\lambda}_{3,1}^t \times \hat{\lambda}_{4,1}^t$$

Bit Decision for VN i at t^{th} Iteration

($i = 1$ in the hypothetical example considered here)

- The bit decisions are not part of the looping process — there is no concern of positive feedback mentioned earlier
- Therefore, the likelihood ratio (odds in favor of VN 1 equal to 1) is calculated using all the information available from the CNs and from the channel

$$\begin{aligned}\lambda_1^t &= \frac{p_1}{1-p_1} \times \frac{\hat{p}_{1,1}^t}{1-\hat{p}_{1,1}^t} \times \frac{\hat{p}_{2,1}^t}{1-\hat{p}_{2,1}^t} \times \frac{\hat{p}_{3,1}^t}{1-\hat{p}_{3,1}^t} \times \frac{\hat{p}_{4,1}^t}{1-\hat{p}_{4,1}^t} \\ &= \lambda_1 \times \hat{\lambda}_{1,1}^t \times \hat{\lambda}_{2,1}^t \times \hat{\lambda}_{3,1}^t \times \hat{\lambda}_{4,1}^t\end{aligned}$$

- Decision process is based on the above likelihood ratio, and it is identical to the one we have seen earler

$$\hat{c}_1^t = \begin{cases} 1; & \lambda_1^t \geq 1 \\ 0; & \text{otherwise} \end{cases}$$

Soft-Decision Decoding on Tanner Graph

for the BEC(p) Channel

Decoding can be better visualized and implemented using the Tanner graph framework:

1. Set the iteration index $t = 0$
2. Load each variable node VN i with the ~~observed bits/erasures at the output of the channel~~ probability p_i = the conditional probability that VN i is 1 given the received bit r_i

Soft-Decision Decoding on Tanner Graph

for the BEC(p) Channel

Decoding can be better visualized and implemented using the Tanner graph framework:

1. Set the iteration index $t = 0$
2. Load each variable node VN i with the ~~observed bits/erasures at the output of the channel~~ probability p_i = the conditional probability that VN i is 1 given the received bit r_i
   - For the BEC p, if $r_i = 1$, $p_i = 1$; else if $r_i = 0$, $p_i = 0$, else if $r_i = X$ (erasure), $p_i = 0.5$
3. Message from i^{th} VN to j^{th} CN: described on the next slide
   - Also calculate the estimate $\hat{\mathbf{c}}^t$ of the decoded codeword
4. Message from j^{th} CN to i^{th} VN: described on the next slide
5. ~~Are all erasures corrected or no more corrections possible?~~ is the estimated decoded codeword $\hat{\mathbf{c}}^t$ different from the estimate $\hat{\mathbf{c}}^{t-1}$ in the prior iteration? OR is $t > t_{\max}$ (where $t_{\max}$ is the maximum number of iterations allowed)?
   - If yes, terminate and obtain the decoded codeword as $\hat{\mathbf{c}}^t$
   - If not, increment the iteration index (i.e., $t \leftarrow t + 1$) and go to Step 3

Observations

- The decoder's complexity depends on the degree of the check nodes and the variable nodes.
- The degree of a check/variable node is equal to the Hamming weight of the corresponding row/column of the parity-check matrix.
- To achieve a good decoding success performance, a long code is needed.
- Thus, it is desirable to have a code that is long, yet with small row/column weight.

Low Density Parity Check (LDPC) Codes

- An LDPC code is characterized by a sparse parity-check matrix **H**.
- The row/column weights of this **H** are independent of length N of the codeword.
- Accordingly, the decoder complexity grows only linearly with block length.
- The LDPC codes were first proposed and developed by Robert Gallager in his 1960 PhD dissertation.
- Were forgotten because the decoder could not be implemented.
- Were "rediscovered" in the mid-1990's by MacKay and Neal after turbo codes were developed.

Low Density Parity Check (LDPC) Codes

An example LDPC code parity check matrix from MacKay and Neal (1996):

$$\mathbf{H} = \left[\begin{array}{cccc|cccc|cccc}
1 & & & & & 1 & & 1 & & 1 & & \\
1 & & & 1 & 1 & & & & & & 1 & \\
& 1 & & & 1 & & 1 & & 1 & & & \\
& & 1 & & & 1 & & & & & 1 & 1 \\
& & 1 & & & & 1 & 1 & & & & 1 \\
& 1 & & & 1 & & & & 1 & & 1 & \\
1 & & & 1 & & & 1 & & & 1 & & \\
& 1 & & & & 1 & & 1 & & 1 & & \\
& & 1 & 1 & & & & & 1 & & & 1
\end{array}\right]$$

- Called a $(w_c, w_r) = (3, 4)$ regular code.
- For the regular codes,
  - rows of $\mathbf{H}$ have a constant Hamming weight of w_r; alternatively, the degree of the check nodes d_c is a constant
  - columns of $\mathbf{H}$ have a constant Hamming weight of w_c; alternatively, the degree of the variable nodes d_v is a constant

Analytical Derivation of the Decoding Probability for the BEC(p)

- The performance of the LDPC codes on an ideal bipartite graph can surprisingly be derived purely analytically

Analytical Derivation of the Decoding Probability for the BEC(p)

- The performance of the LDPC codes on an ideal bipartite graph can surprisingly be derived purely analytically
  - A graph with cycles of short length can cause a loss of independent information to be fed back in the iterative decoder.

Analytical Derivation of the Decoding Probability for the BEC(p)

- The performance of the LDPC codes on an ideal bipartite graph can surprisingly be derived purely analytically
  - A graph with cycles of short length can cause a loss of independent information to be fed back in the iterative decoder.
- The following derivation assumes an ideal bipartite graph, i.e., the one without any cycles

Analytical Derivation of the Decoding Probability for the BEC(p)

Check to Variable Node Message

- Decoding involves the exchange of messages between variable nodes and check nodes.

Analytical Derivation of the Decoding Probability for the BEC(p)

Check to Variable Node Message

- Decoding involves the exchange of messages between variable nodes and check nodes.
- Consider the degree d_c check node.
  - e_t is the probability that the message going from the variable node to the check node is an erasure at t^{th} iteration
  - $e_0 = p$ (recall, p is the original BEC-induced erasure probability)

Analytical Derivation of the Decoding Probability for the BEC(p)

Check to Variable Node Message

- Decoding involves the exchange of messages between variable nodes and check nodes.
- Consider the degree d_c check node.
  - e_t is the probability that the message going from the variable node to the check node is an erasure at t^{th} iteration
  - $e_0 = p$ (recall, p is the original BEC-induced erasure probability)
  - An outgoing message sent over an edge of the Tanner graph (by a particular check node toward a particular variable node) is a function of the incoming messages arriving over the other $d_c - 1$ edges.

Analytical Derivation of the Decoding Probability for the BEC(p)

Check to Variable Node Message

- Decoding involves the exchange of messages between variable nodes and check nodes.
- Consider the degree d_c check node.
  - e_t is the probability that the message going from the variable node to the check node is an erasure at t^{th} iteration
  - $e_0 = p$ (recall, p is the original BEC-induced erasure probability)
  - An outgoing message sent over an edge of the Tanner graph (by a particular check node toward a particular variable node) is a function of the incoming messages arriving over the other $d_c - 1$ edges.
  - For the outgoing message to be correct, all $d_c - 1$ incoming messages must be correct.

Analytical Derivation of the Decoding Probability for the BEC(p)

Check to Variable Node Message

- Decoding involves the exchange of messages between variable nodes and check nodes.
- Consider the degree d_c check node.
  - e_t is the probability that the message going from the variable node to the check node is an erasure at t^{th} iteration
  - $e_0 = p$ (recall, p is the original BEC-induced erasure probability)
  - An outgoing message sent over an edge of the Tanner graph (by a particular check node toward a particular variable node) is a function of the incoming messages arriving over the other $d_c - 1$ edges.
  - For the outgoing message to be correct, all $d_c - 1$ incoming messages must be correct.
  - Probability of this is $(1 - e_t)^{d_c - 1}$.
  - Therefore, the probability of the check node sending an erasure is

Analytical Derivation of the Decoding Probability for the BEC(p)

Check to Variable Node Message

- Decoding involves the exchange of messages between variable nodes and check nodes.
- Consider the degree d_c check node.
  - e_t is the probability that the message going from the variable node to the check node is an erasure at t^{th} iteration
  - $e_0 = p$ (recall, p is the original BEC-induced erasure probability)
  - An outgoing message sent over an edge of the Tanner graph (by a particular check node toward a particular variable node) is a function of the incoming messages arriving over the other $d_c - 1$ edges.
  - For the outgoing message to be correct, all $d_c - 1$ incoming messages must be correct.
  - Probability of this is $(1 - e_t)^{d_c - 1}$.
  - Therefore, the probability of the check node sending an erasure is

$$e_t^c = 1 - (1 - e_t)^{d_c - 1} \tag{2}$$

Analytical Derivation of the Decoding Probability for the BEC(p)

Variable to Check Node Message

- Next consider the degree d_v variable node.

Analytical Derivation of the Decoding Probability for the BEC(p)

Variable to Check Node Message

- Next consider the degree d_v variable node.
  - An outgoing message sent by a particular variable over a particular edge is a function of the incoming messages arriving over the other $d_v - 1$ edges.
  - For the outgoing message to be an erasure at ℓ^{th} iteration, the variable node has to be erased over the channel (this occurs with a probability of $e_0 = p$), <u>and</u> all $d_v - 1$ messages have all to be erased at t^{th} iteration. This occurs with a probability of

Analytical Derivation of the Decoding Probability for the BEC(p)

Variable to Check Node Message

- Next consider the degree d_v variable node.
  - An outgoing message sent by a particular variable over a particular edge is a function of the incoming messages arriving over the other $d_v - 1$ edges.
  - For the outgoing message to be an erasure at ℓ^{th} iteration, the variable node has to be erased over the channel (this occurs with a probability of $e_0 = p$), <u>and</u> all $d_v - 1$ messages have all to be erased at t^{th} iteration. This occurs with a probability of

$$e_{t+1} = e_0 (e_t^c)^{d_v - 1}. \tag{3}$$

Analytical Derivation of the Decoding Probability for the BEC(p)

Variable to Check Node Message

- Next consider the degree d_v variable node.
  - An outgoing message sent by a particular variable over a particular edge is a function of the incoming messages arriving over the other $d_v - 1$ edges.
  - For the outgoing message to be an erasure at ℓ^{th} iteration, the variable node has to be erased over the channel (this occurs with a probability of $e_0 = p$), <u>and</u> all $d_v - 1$ messages have all to be erased at t^{th} iteration. This occurs with a probability of

$$e_{t+1} = e_0 (e_t^c)^{d_v - 1}. \tag{3}$$

- Combining Equations 2 and 3, we obtain

$$e_{t+1} = e_0 \left(1 - (1 - e_t)^{d_c - 1}\right)^{d_v - 1}. \tag{4}$$

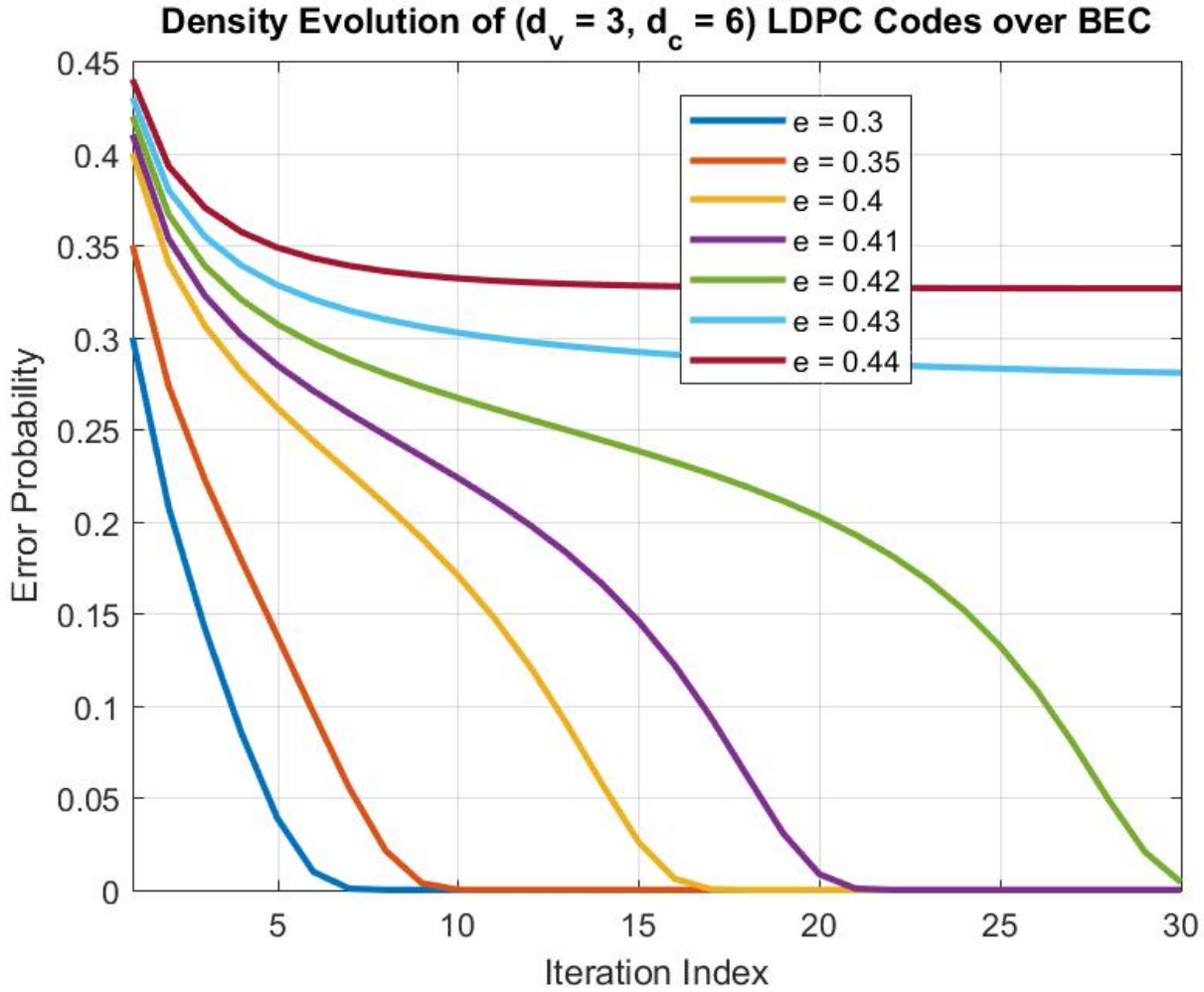
Analytical Derivation of the Decoding Probability for the BEC(p)

using Density Evolution Method

- The method described earlier is called the Density Evolution (DE) method of evaluating the performance of regular LDPC codes
- This DE method assumes independent messages, which is achieved when the girth (i.e., length of the shortest cycle) of the Tanner graph is sufficiently large.
- If $e_t \to 0$ as $t \to \infty$ for a particular channel erasure probability p, a code drawn from the ensemble of all such (d_v, d_c) regular LDPC codes will be able to correctly decode.
- The threshold p^* is the maximum p for which $e_t \to 0$ as $t \to \infty$.
- For the $(d_v = 3, d_c = 6)$ regular code, the threshold e^* is 0.4294.

Analytical Decoding Probability for the BEC(p)

using Density Evolution Method



Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- We will now discuss the Q1 and Q2 that we had considered in the context of a data transmission at the rate of 1000 bits/sec over the BSC(p)

Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- We will now discuss the Q1 and Q2 that we had considered in the context of a data transmission at the rate of 1000 bits/sec over the BSC(p)
- To ensure a reliable information transfer over the BSC, the transmitter needs to allocate a fraction of transmitted bits so that the receiver is not confused about which message was transmitted

Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- We will now discuss the Q1 and Q2 that we had considered in the context of a data transmission at the rate of 1000 bits/sec over the BSC(p)
- To ensure a reliable information transfer over the BSC, the transmitter needs to allocate a fraction of transmitted bits so that the receiver is not confused about which message was transmitted
- Toward this, let us consider the BSC as a source of a binary string of length 1000 bits/sec in the example we have considered

Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- We will now discuss the Q1 and Q2 that we had considered in the context of a data transmission at the rate of 1000 bits/sec over the BSC(p)
- To ensure a reliable information transfer over the BSC, the transmitter needs to allocate a fraction of transmitted bits so that the receiver is not confused about which message was transmitted
- Toward this, let us consider the BSC as a source of a binary string of length 1000 bits/sec in the example we have considered
  - Bit 1 of this BSC string specifies the location where the error has occurred
  - Bit 0 specifies the locations that are error free

Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- Denoting the transmitted binary sequence as $\mathbf{x}$, and the BSC noise string as $\mathbf{n}$, the received binary sequence $\mathbf{y}$ can be written in the following two ways:
  - exclusive-OR of $\mathbf{x}$ and $\mathbf{n}$: $\mathbf{y} = \mathbf{x} \oplus \mathbf{n}$
  - sum (modulo-2) of $\mathbf{x}$ and $\mathbf{n}$: $\mathbf{y} = (\mathbf{x} + \mathbf{n})\ mod - 2$
- An example:
  - $x = 0,\ 1,\ 0,\ 0,\ 1,\ 0,\ 0,\ 1$
  - $n = 0,\ 0,\ 1,\ 0,\ 1,\ 0,\ 0,\ 0$
  - $y = 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 1$

  Notice that where $\mathbf{n}$ is 1, the corresponding bit of $\mathbf{x}$ gets flipped in $\mathbf{y}$

Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- Suppose the transmitted bit string $\mathbf{x}$ has a length of N bits
- The BSC(p) introduces errors that occur with a probability of p
  - $\rightarrow$ The BSC can be thought of as a DMS that generates length N binary string $\mathbf{n}$ in which probability of 1 equals p

Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- Suppose the transmitted bit string $\mathbf{x}$ has a length of N bits
- The BSC(p) introduces errors that occur with a probability of p
  - $\rightarrow$ The BSC can be thought of as a DMS that generates length N binary string $\mathbf{n}$ in which probability of 1 equals p
- Question: how many different binary strings can the transmitted bit sequence $\mathbf{x}$ turn into because of the effect of the BSC(p)?

Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- Suppose the transmitted bit string $\mathbf{x}$ has a length of N bits
- The BSC(p) introduces errors that occur with a probability of p
  - $\rightarrow$ The BSC can be thought of as a DMS that generates length N binary string $\mathbf{n}$ in which probability of 1 equals p
- Question: how many different binary strings can the transmitted bit sequence $\mathbf{x}$ turn into because of the effect of the BSC(p)?
- Answer:
  - $\rightarrow$ For small values of N, it is not possible to specify this.
  - $\rightarrow$ However, as $N \rightarrow \infty$, the BSC(p) induced noise pattern $\mathbf{n}$ gets <u>trapped</u> into a typical set of size $2^{N \times H_b(p)}$
    - ▷ Recall that we have defined $H_b(p) \stackrel{\text{def}}{=} -(p \log_2 p + (1-p)\log_2(1-p))$

Model of Channel Induced Bit Errors

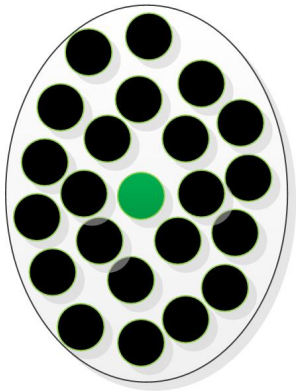
for Binary Symmetric Channel

- A single transmitted binary string of $N \times H_b(p)$ bits can turn into $2^{N \times H_b(p)}$ other strings around the transmitted string

Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- A single transmitted binary string of $N \times H_b(p)$ bits can turn into $2^{N \times H_b(p)}$ other strings around the transmitted string



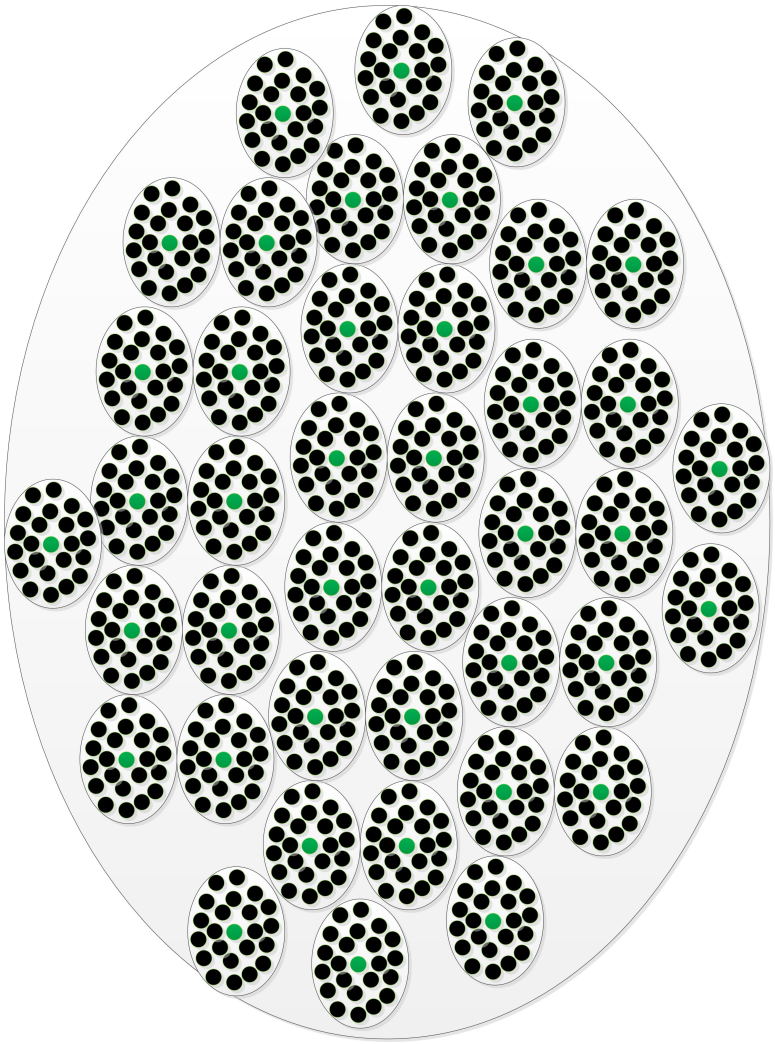
Model of Channel Induced Bit Errors

for Binary Symmetric Channel

- Generalizing, we transmitting N bits over BSC(p), we can send information over $N \times (1 - H_b(p))$ bits and these information bits are ensured to not get corrupted.
- Total number of bits sent: N; number of bits that carry information: $N \times (1 - H_b(p))$; redundant bits required to isolate the transmitted message from the effect of BSC(p): $N \times H_b(p)$

Model of Channel Induced Bit Errors

for Binary Symmetric Channel



Summary

for Binary Symmetric Channel

- When transmitting a total of N bits, we can divide the total set as $2^N = (2^{N-N\times H_b(p)}) \times 2^{N\times H_b(p)}$
- Thus, we form a total of $2^{N-N\times H_b(p)}$ non-overlapping subsets of the full set of size 2^N. Size of each subset is $2^{N\times H_b(p)}$
- Transmitted channel codewords are the centers of these subsets

Summary

for Binary Symmetric Channel

- When transmitting a total of N bits, we can divide the total set as $2^N = (2^{N-N\times H_b(p)}) \times 2^{N\times H_b(p)}$
- Thus, we form a total of $2^{N-N\times H_b(p)}$ non-overlapping subsets of the full set of size 2^N. Size of each subset is $2^{N\times H_b(p)}$
- Transmitted channel codewords are the centers of these subsets
- BSC(p) can turn the transmitted codeword into any one string in the corresponding subset. As $N \to \infty$, the received codeword is <u>ensured</u> to be trapped within this subset.

Summary

for Binary Symmetric Channel

- When transmitting a total of N bits, we can divide the total set as $2^N = (2^{N-N\times H_b(p)}) \times 2^{N\times H_b(p)}$
- Thus, we form a total of $2^{N-N\times H_b(p)}$ non-overlapping subsets of the full set of size 2^N. Size of each subset is $2^{N\times H_b(p)}$
- Transmitted channel codewords are the centers of these subsets
- BSC(p) can turn the transmitted codeword into any one string in the corresponding subset. As $N \to \infty$, the received codeword is <u>ensured</u> to be trapped within this subset.
- Receiver calculates which subset the received string falls into, and then, it takes the transmitted codeword as the center of that subset

Summary

for Binary Symmetric Channel

- When transmitting a total of N bits, we can divide the total set as $2^N = (2^{N-N\times H_b(p)}) \times 2^{N\times H_b(p)}$
- Thus, we form a total of $2^{N-N\times H_b(p)}$ non-overlapping subsets of the full set of size 2^N. Size of each subset is $2^{N\times H_b(p)}$
- Transmitted channel codewords are the centers of these subsets
- BSC(p) can turn the transmitted codeword into any one string in the corresponding subset. As $N \to \infty$, the received codeword is <u>ensured</u> to be trapped within this subset.
- Receiver calculates which subset the received string falls into, and then, it takes the transmitted codeword as the center of that subset
- Since there are $2^{N-N\times H_b(p)}$ total mutually nonoverlapping subsets, the transmitter can set a total of $N - N \times H_b(p)$ bits that carry information. It has to insert the remaining $N \times H_b(p)$ bits as redundant bits that are required to ensure the unique decodeability

Summary (Alternate Version)

for Binary Symmetric Channel

- Transmitter sends a "codeword" which is N bits long.
  - → This is the channel encoded codeword.
  - → Reason it is called a "codeword" is because it is different from the actual information or message bits that come out of the source encoder

Summary (Alternate Version)

for Binary Symmetric Channel

- Transmitter sends a "codeword" which is N bits long.
  - → This is the channel encoded codeword.
  - → Reason it is called a "codeword" is because it is different from the actual information or message bits that come out of the source encoder
- There are total of 2^N possible codewords that are N bit long

Summary (Alternate Version)

for Binary Symmetric Channel

- Transmitter sends a "codeword" which is N bits long.
  - → This is the channel encoded codeword.
  - → Reason it is called a "codeword" is because it is different from the actual information or message bits that come out of the source encoder
- There are total of 2^N possible codewords that are N bit long
- We will assume that the transmitter knows that its codeword has to be transmitted on a BSC(p); i.e., the transmitter knows the value of p.
  - → In reality, the transmitter may not know the value of p, but the engineering design is done for the worst possible value of p that is expected. This worst-case value of p may be possible to figure out, or at least guess

Summary (Alternate Version)

for Binary Symmetric Channel

- Transmitter sends a "codeword" which is N bits long.
  - → This is the channel encoded codeword.
  - → Reason it is called a "codeword" is because it is different from the actual information or message bits that come out of the source encoder
- There are total of 2^N possible codewords that are N bit long
- We will assume that the transmitter knows that its codeword has to be transmitted on a BSC(p); i.e., the transmitter knows the value of p.
  - → In reality, the transmitter may not know the value of p, but the engineering design is done for the worst possible value of p that is expected. This worst-case value of p may be possible to figure out, or at least guess
- In this case, instead of using all 2^N codewords, the transmitter pre-selects $2^{N\times(1-H_b(p))}$ codewords that it will transmit

Summary (Alternate Version)

for Binary Symmetric Channel

- Transmitter sends a "codeword" which is N bits long.
  - → This is the channel encoded codeword.
  - → Reason it is called a "codeword" is because it is different from the actual information or message bits that come out of the source encoder
- There are total of 2^N possible codewords that are N bit long
- We will assume that the transmitter knows that its codeword has to be transmitted on a BSC(p); i.e., the transmitter knows the value of p.
  - → In reality, the transmitter may not know the value of p, but the engineering design is done for the worst possible value of p that is expected. This worst-case value of p may be possible to figure out, or at least guess
- In this case, instead of using all 2^N codewords, the transmitter pre-selects $2^{N\times(1-H_b(p))}$ codewords that it will transmit
  - → This forms the (channel) codeword book (or a dictionary of all the codewords)

Summary (Alternate Version)

for Binary Symmetric Channel

- Transmitter sends a "codeword" which is N bits long.
  - → This is the channel encoded codeword.
  - → Reason it is called a "codeword" is because it is different from the actual information or message bits that come out of the source encoder
- There are total of 2^N possible codewords that are N bit long
- We will assume that the transmitter knows that its codeword has to be transmitted on a BSC(p); i.e., the transmitter knows the value of p.
  - → In reality, the transmitter may not know the value of p, but the engineering design is done for the worst possible value of p that is expected. This worst-case value of p may be possible to figure out, or at least guess
- In this case, instead of using all 2^N codewords, the transmitter pre-selects $2^{N\times(1-H_b(p))}$ codewords that it will transmit
  - → This forms the (channel) codeword book (or a dictionary of all the codewords)
  - → This codeword book is kept at the transmitter (channel encoder) as well as at the receiver (channel decoder)

Summary (Alternate Version)

for Binary Symmetric Channel

- Transmitter sends a "codeword" which is N bits long.
  - → This is the channel encoded codeword.
  - → Reason it is called a "codeword" is because it is different from the actual information or message bits that come out of the source encoder
- There are total of 2^N possible codewords that are N bit long
- We will assume that the transmitter knows that its codeword has to be transmitted on a BSC(p); i.e., the transmitter knows the value of p.
  - → In reality, the transmitter may not know the value of p, but the engineering design is done for the worst possible value of p that is expected. This worst-case value of p may be possible to figure out, or at least guess
- In this case, instead of using all 2^N codewords, the transmitter pre-selects $2^{N\times(1-H_b(p))}$ codewords that it will transmit
  - → This forms the (channel) codeword book (or a dictionary of all the codewords)
  - → This codeword book is kept at the transmitter (channel encoder) as well as at the receiver (channel decoder)
  - → Just like a dictionary, this has a unique (one-to-one) map. Each one of $2^{N\times(1-H_b(p))}$ message block is mapped to a unique N-bit long codeword

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- Any message that transmitter sends is an N-bit codeword

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- Any message that transmitter sends is an N-bit codeword
- This message is, however, one codeword out of a codeword dictionary that has a total of $2^{N\times(1-H_b(p))}$ codewords

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- Any message that transmitter sends is an N-bit codeword
- This message is, however, one codeword out of a codeword dictionary that has a total of $2^{N\times(1-H_b(p))}$ codewords
- Therefore, the transmitter actually needs (not N, but instead) $N \times (1 - H_b(p))$ bits to specify (or select) which codeword is to be sent

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- Any message that transmitter sends is an N-bit codeword
- This message is, however, one codeword out of a codeword dictionary that has a total of $2^{N\times(1-H_b(p))}$ codewords
- Therefore, the transmitter actually needs (not N, but instead) $N \times (1 - H_b(p))$ bits to specify (or select) which codeword is to be sent
- Accordingly, the transmitter takes a block of $N \times (1 - H_b(p))$ message or information bits coming out of the source encoder to select one of $2^{N\times(1-H_b(p))}$ codewords from the dictionary
  - $\rightarrow$ The selected N-bit long codeword is one of $2^{N\times(1-H_b(p))}$ green circles on slide 127

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- Any message that transmitter sends is an N-bit codeword
- This message is, however, one codeword out of a codeword dictionary that has a total of $2^{N\times(1-H_b(p))}$ codewords
- Therefore, the transmitter actually needs (not N, but instead) $N \times (1 - H_b(p))$ bits to specify (or select) which codeword is to be sent
- Accordingly, the transmitter takes a block of $N \times (1 - H_b(p))$ message or information bits coming out of the source encoder to select one of $2^{N\times(1-H_b(p))}$ codewords from the dictionary
  - $\rightarrow$ The selected N-bit long codeword is one of $2^{N\times(1-H_b(p))}$ green circles on slide 127
- The BSC(p) can turn this selected codeword into a total of $2^{NH_b(p)}$ possible N-bit long binary strings that make up the little oval around the selected (green colored) codeword

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- Any message that transmitter sends is an N-bit codeword
- This message is, however, one codeword out of a codeword dictionary that has a total of $2^{N\times(1-H_b(p))}$ codewords
- Therefore, the transmitter actually needs (not N, but instead) $N \times (1 - H_b(p))$ bits to specify (or select) which codeword is to be sent
- Accordingly, the transmitter takes a block of $N \times (1 - H_b(p))$ message or information bits coming out of the source encoder to select one of $2^{N\times(1-H_b(p))}$ codewords from the dictionary
  - $\rightarrow$ The selected N-bit long codeword is one of $2^{N\times(1-H_b(p))}$ green circles on slide 127
- The BSC(p) can turn this selected codeword into a total of $2^{NH_b(p)}$ possible N-bit long binary strings that make up the little oval around the selected (green colored) codeword
  - $\rightarrow$ All the black points in this oval denote all the possible ways that the BSC can turn the transmitted codeword into the received codeword
  - $\rightarrow$ Effect of the BSC is guaranteed to be confined within this little oval

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- When the receiver gets a noisy codeword over BSC, it calculates the "distance" of this received noisy codeword against all the valid, non-noisy, codewords it has in its codebook dictionary

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- When the receiver gets a noisy codeword over BSC, it calculates the "distance" of this received noisy codeword against all the valid, non-noisy, codewords it has in its codebook dictionary
- Receiver selects that codeword as the one that must have been transmitted for which the distance to the received noisy codeword is the smallest

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- When the receiver gets a noisy codeword over BSC, it calculates the "distance" of this received noisy codeword against all the valid, non-noisy, codewords it has in its codebook dictionary
- Receiver selects that codeword as the one that must have been transmitted for which the distance to the received noisy codeword is the smallest
- With this "algorithm", the receiver is guaranteed to select the same codeword that the transmitter has sent. It has overcome the effect of the noise.

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- When the receiver gets a noisy codeword over BSC, it calculates the "distance" of this received noisy codeword against all the valid, non-noisy, codewords it has in its codebook dictionary
- Receiver selects that codeword as the one that must have been transmitted for which the distance to the received noisy codeword is the smallest
- With this "algorithm", the receiver is guaranteed to select the same codeword that the transmitter has sent. It has overcome the effect of the noise.
- With this scheme, the channel encoding completely protects the transmitted information bits of length $N \times (1 - H_b(p)$.

Summary (Alternate Version, Continued)

for Binary Symmetric Channel

- When the receiver gets a noisy codeword over BSC, it calculates the "distance" of this received noisy codeword against all the valid, non-noisy, codewords it has in its codebook dictionary
- Receiver selects that codeword as the one that must have been transmitted for which the distance to the received noisy codeword is the smallest
- With this "algorithm", the receiver is guaranteed to select the same codeword that the transmitter has sent. It has overcome the effect of the noise.
- With this scheme, the channel encoding completely protects the transmitted information bits of length $N \times (1 - H_b(p)$.
- The cost paid is that the transmitter needed to send N bits instead of $N \times (1 - H_b(p)$ bits, the extra $NH_b(p)$ bits are required as the redundancy bits that overcome the channel induced noise errors.

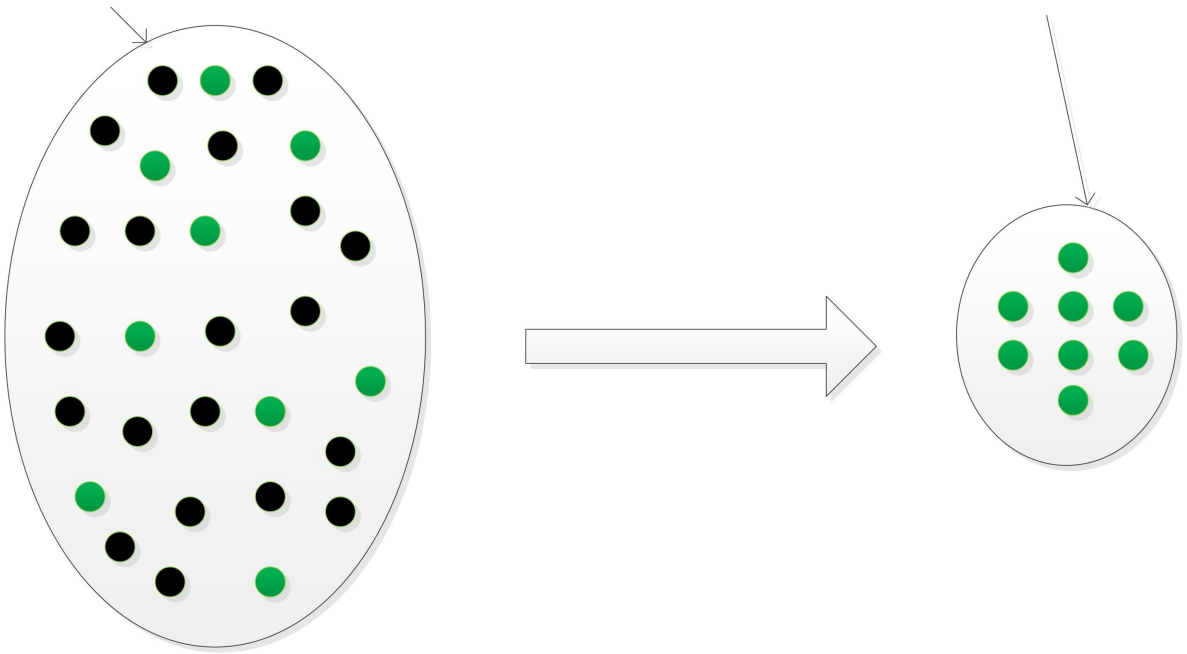
A Non-uniform PMF

Data Compression Possible

- A view of data compression

Before data compression:
Requires N bits
Total size of the set: 2^N

After data compression:
Requires N× H(X) bits
Total size of the set: $2^{N \times H(X)}$



- Members of the Typical Set
- Remaining, belong to subsets with vanishing probability as N becomes large

Data Compression and Error Correction

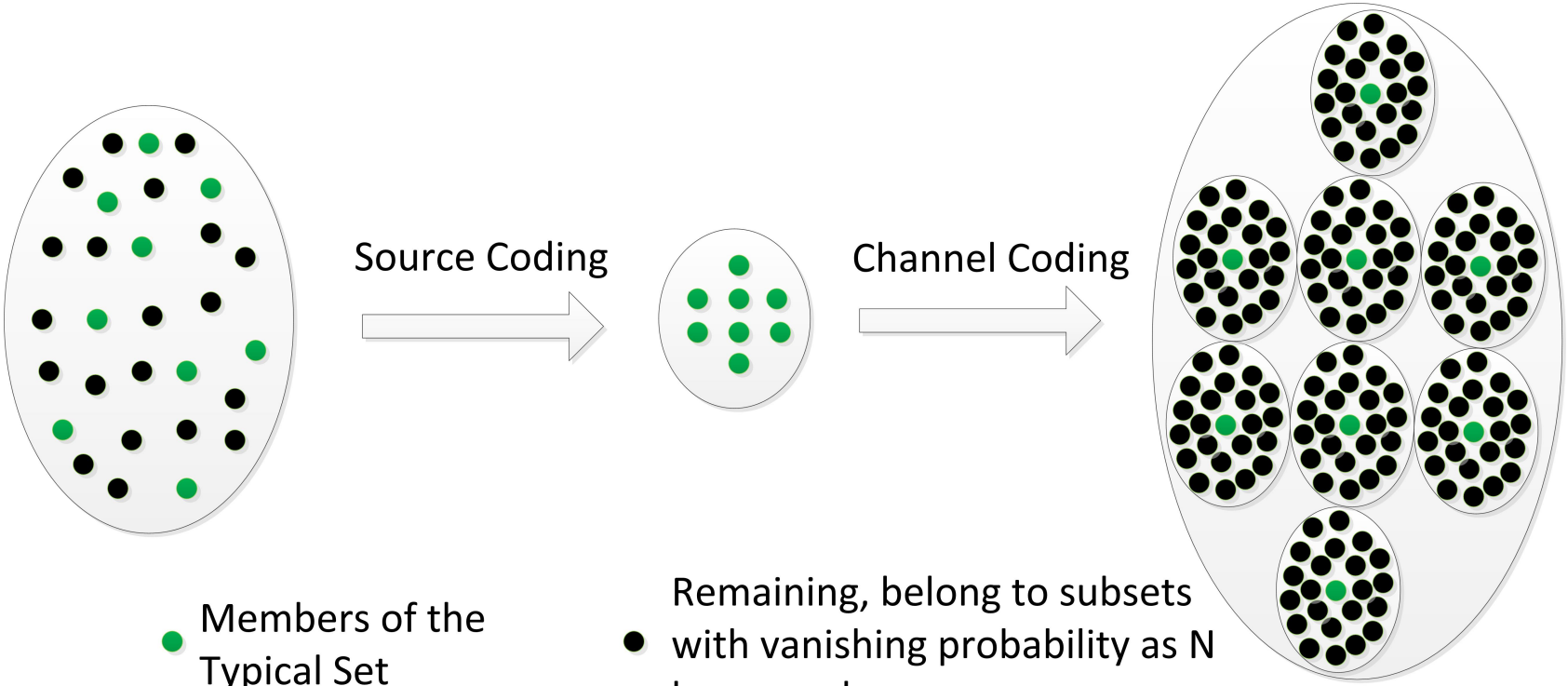
- Data compression and error correction are dual to each other
  - Uncompressed data allows for error correction
  - Compressed data leaves no room for correcting for the errors

Source Coding versus Channel Coding

Uncompressed Source Output

After Compression

After Channel Encoding: Transmitter selects one of green cicrles; Receiver receives any one of green and black circles

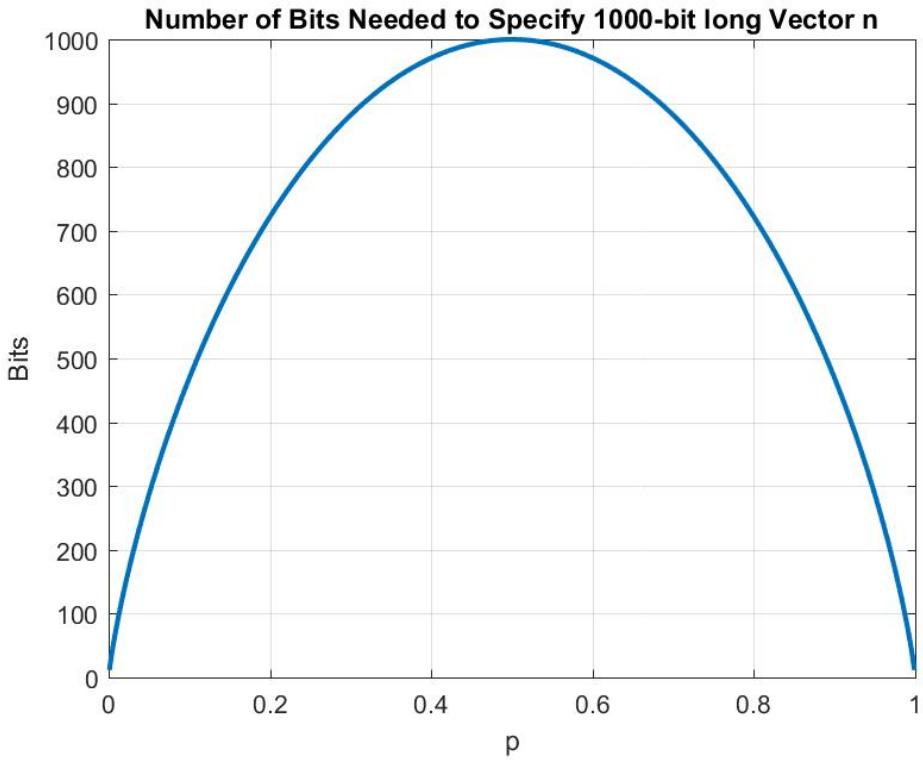


- Members of the Typical Set
- Remaining, belong to subsets with vanishing probability as N becomes large

Binary Symmetric Channel or BSC

Redundancy Bits

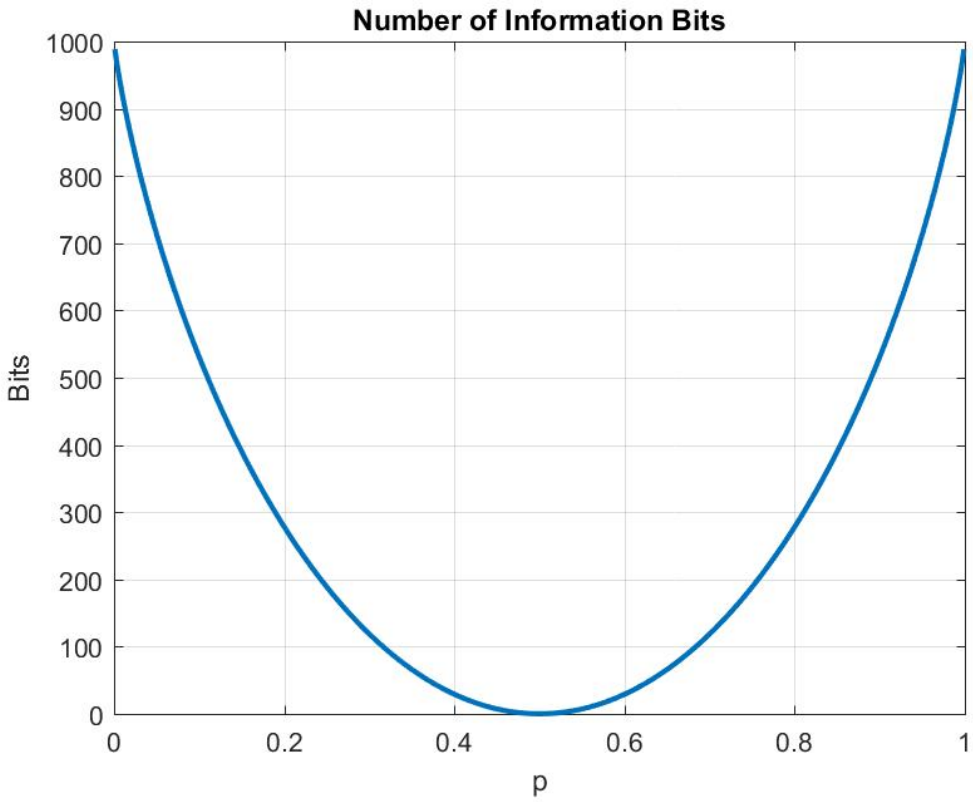
- Smallest number of bits required to isolate the effect of adding the binary string $\mathbf{n}$ of length N bits: $N \times H_b(p)$.
- Transmitter has to insert these bits, but they don't carry information (they ensure that the receiver is not confused about which message was transmitted although the individual bits can be in error)



Binary Symmetric Channel or BSC

Information Bits

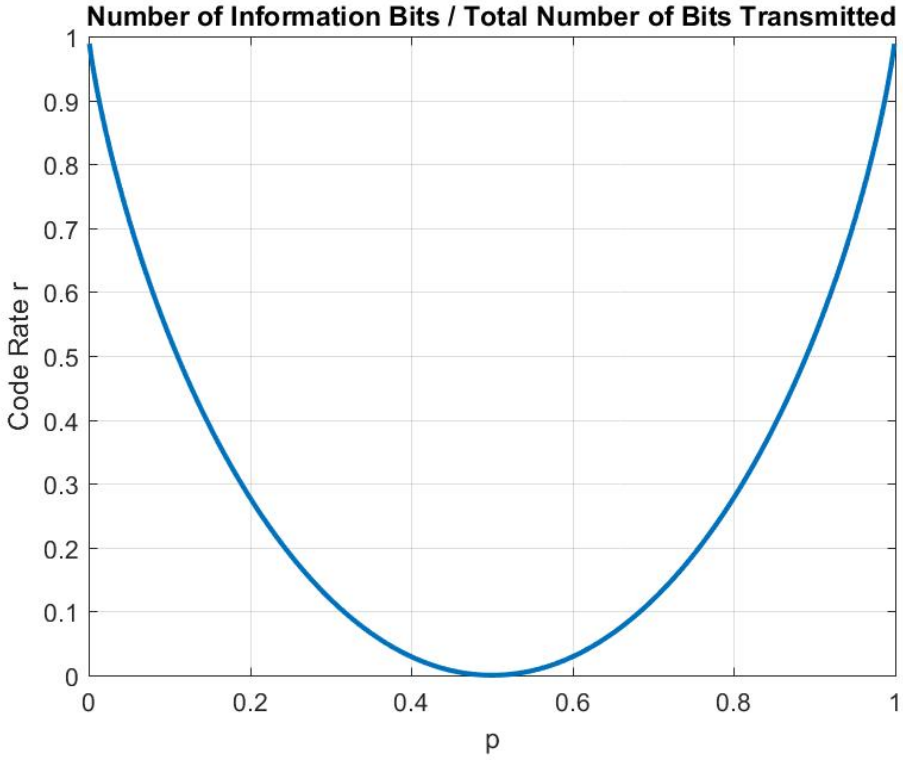
- Maximum number of bits of **x** that can carry the actual sender's information: $N - N \times H_b(p)$.
- This is the actual information transferred in bits. As expected, this becomes zero when $p = 0.5$.



Binary Symmetric Channel or BSC

Normalized Rate

- The normalized (maximum) rate of information transfer over BSC with cross-over probability of p: $r = 1 - H_b(p)$.
- Closer the value of BSC's p to 0.5, smaller the value of r, the greater the required redundancy and smaller the capacity of the channel to carry the actual information



Binary Symmetric Channel or BSC

Summary

- Maximum possible rate of information transfer with error-free reception:

$$C = 1 - H_b(p) bits/bit$$

- Here, $H_b(p) = -p \log_2 p - (1-p) \log_2 (1-p)$ bits is the binary entropy function.
- For the BSC(p), the transmitter is required to allocate $H_b(p)$ bits just to overcome the effect of the noise.
- It can put the information in only remaining $1 - H_b(p)$ bits

Binary Symmetric Channel or BSC

Summary

- When $p = 0.01$, $1 - H_b(p) \approx 0.92$.
  - $\rightarrow$ The transmitter can send the information in 920 bits out of 1000 bits sent every second. The maximum rate of information transfer is 920 bits/sec.
- When $p = 0.1$, $1 - H_b(p) = 0.53$.
  - $\rightarrow$ The transmitter can send the information in 530 bits out of 1000 bits sent every second. The maximum rate of information transfer is 530 bits/sec.

Information Transfer

for Analog Gaussian Noise Channel

- For the analog channel in which the transmitter sends an average power of P_s Watts, and the receiver introduces Gaussian noise of power P_n watts,
  - → $H(Y)$ is $\frac{1}{2}\log_2(P_s + P_n)$ bits.
    - ▷ This occurs when the receiver has highest degree of uncertainty about the analog signal that is received.
  - → $H(Y|X)$ equals $\frac{1}{2}\log_2(P_n)$ bit.
    - ▷ This is because, given X, the only source of randomness in Y arises from the analog Gaussian noise vector $\mathbf{n}$. Therefore, the entropy of Y, <u>conditioned</u> upon X, is the entropy of this Gaussian noise, and its entropy equals $\frac{1}{2}\log_2(P_n)$ bit.
- Thus, $I(X;Y) = \frac{1}{2}\log_2(P_s + P_n) - \log_2(P_n) = \frac{1}{2}\log_2\left(1 + \frac{P_s}{P_n}\right)$ bits.

Information Transfer

for Analog Gaussian Noise Channel

- According to Shannon-Nyquist sampling theorem, if the analog waveforms used for communication have a bandwidth of W Hz, the number of samples per second required is $2W$ samples/second
- Therefore, $I(X;Y) = C = W \log_2 \left(1 + \frac{P_s}{P_n}\right)$ bits/second
- This is the celebrated channel capacity theorem for analog Gaussian noise channels

Application of Linear Algebra

to Derive Some Properties of the Linear Block Codes

- The formulations (**G** and **H** matrices) that we have developed apply to a large class of channel coding schemes known as Linear Block Codes (LBC)
- Several examples that we have seen (the SPC, the repetition, the product code and the RPC, the Hamming code, etc.) all belong to the LBC family
  - → There are more powerful codes — known as the Golay, BCH (Bose-Chaudhari-Hocquenghem), RS (Reed-Solomon) codes — in the LBC family. However, we will not study them in this course
  - → The convolutional codes also technically conform to the LBC definition (i.e., **H** and **G** matrices can be defined for them), however they are conventionally considered as being different from the LBC
- All of the codes belonging to the LBC share some properties that we study next.
  - → These properties are derived by noting that a binary sequence **c** is a valid codeword of the LBC if and only if $\mathbf{H}\,\mathbf{c} = \emptyset$ (i.e., **c** should satisfy all the parity checks)

Application of Linear Algebra

to Derive Some Properties of the Linear Block Codes

1. An all-zero vector $\mathbf{c}$ is always a valid codeword.
2. The sum of two different valid codewords $\mathbf{c}_1$ and $\mathbf{c}_2$ is also a valid codeword.
3. The minimum Hamming distance $d_H^{\min}$ equals the minimum of all Hamming weights of all valid nonzero codewords.
4. Let the binary word $\mathbf{r}$ received over the BSC be given as $\mathbf{r} = \mathbf{c} + \mathbf{e}$, where $\mathbf{c}$ is any one of 2^k transmitted valid codewords and $\mathbf{e}$ is a length n binary sequence introduced by the BSC (the bits of $\mathbf{c}$ at the locations of ones of $\mathbf{e}$ are flipped). Let $\mathbf{s} = \mathbf{H}\,\mathbf{r}$ be $u \times 1$ vector, known as the "syndrome" vector, calculated by the decoder at the receiver. This syndrome vector $\mathbf{s}$ is a function only of $\mathbf{e}$ and not of $\mathbf{c}$, i.e., given a particular $\mathbf{e}$ vector, it does not matter which of 2^k valid codewords was transmitted, the receiver calculates the same syndrome vector $\mathbf{s}$.

Application of the Properties of the Linear Block Codes

Properties 1, 2 and 3

- Properties 1 and 2 on the previous slide are useful in many different instances (e.g., they are used to derive the proof of Property 3)
- Property 3 is quite helpful to determine $d_H^{\min}$
  - → Recall that evaluation of $d_H^{\min}$ requires first a tabulation of all pairwise distances d_H between all possible codewords. Since there is 2^k codewords, the total pairwise distances that need to be calculated is $\binom{2^k}{2} = 2^{2(k-2)}$. This is a huge number.
  - → Instead of this, all we need to do is to determine the smallest weight valid codeword. The weight of this codeword, by the virtue of Property 3 of the LBC, is $d_H^{\min}$
- An example evaluation of $d_H^{\min}$ for $(7, 4)$ Hamming code:
  - → Make any one of $k = 4$ message bits 1, keep the remaining 3 as zeros. You will notice that at least 2 parity bits become 1.
  - → Now make any two of $k = 4$ message bits as ones, keep the remaining 2 as zeros. In this case, at least one parity bit turns to 1.
  - → Thus, there is no Hamming codeword whose weight is smaller than 3. Thus, $d_H^{\min} = 3$.

Application of the Properties of the Linear Block Codes

Decoding of $(7, 4)$ Hamming Code using Property 4

- Let us make use of Property 4 in decoding of $(7, 4)$ Hamming Code, specifically in evaluation of the syndrome vector
- Let us assume that all-zero $\mathbf{c}$ codeword has been transmitted
  - $\rightarrow$ According to Property 4, it does not matter which codeword we consider to have been transmitted in calculation of the syndrome vector
  - $\rightarrow$ Therefore, we take that the simplest codeword, which is the all-zero codeword, has been transmitted (this is a valid codeword according to Property 1)
  - $\rightarrow$ Therefore, $\mathbf{r} = \mathbf{c} + \mathbf{e} = \mathbf{e}$.
- We know that $d_H^{\min} = 3$ for $(7, 4)$ Hamming Code, i.e., $t_c = 1$ bit. Therefore, we will consider all possible $\mathbf{e}$ sequences of length $n = 7$ bits that the BSC can introduce in which either no bit is a 1 or only one bit is a 1
- There is only one $\mathbf{e}$ sequence in which no bit is a 1 — this is the all-zero 7 bit sequence. Let us denote this as $\varnothing$ of size 7×1. The syndrome vector $\mathbf{s} = \mathbf{H}\,\mathbf{r} = \mathbf{H}\varnothing = \emptyset$, where $\emptyset$ is of size 3×1.

Application of the Properties of the Linear Block Codes

Decoding of $(7, 4)$ Hamming Code using Property 4

- Next, consider $\mathbf{e}$ sequences in which one bit is a 1 and the remaining are zeros. There are $\binom{n}{1} = 7$ sequences
- In the calculations on the next slide, we have assigned the coded bits to the message bits and the parity bits as follows: $\mathbf{c} = (c_1 = p_1, c_2 = p_2, c_3 = m_1, c_4 = p_4, c_5 = m_2, c_6 = m_3, c_7 = m_4)^T$.
- Since $\mathbf{r} = \mathbf{c} + \mathbf{e}$, we make use of the following equations to compute the syndrome vector $\mathbf{s} = [s_1, s_2, s_3]^T$.

$$s_1 = m_1 + m_2 + m_4 + p_1 \Rightarrow r_3 + r_5 + r_7 + r_1$$
$$s_2 = m_1 + m_3 + m_4 + p_2 \Rightarrow r_3 + r_6 + r_7 + r_2$$
$$s_3 = m_2 + m_3 + m_4 + p_3 \Rightarrow r_5 + r_6 + r_7 + r_4$$

Application of the Properties of the Linear Block Codes

Decoding of $(7, 4)$ Hamming Code using Property 4

- Consider one $\mathbf{e}$ seqeunce, in which the first bit of $\mathbf{e}$ is a 1 and the rest are zeros, i.e., $\mathbf{r} = \mathbf{e} = [1, 0, 0, 0, 0, 0, 0]^T$. The corresponding syndrome vector, from the set of equations shown on the prior slide, equals $\mathbf{s} = [1, 0, 0]^T$.
- Consider another $\mathbf{e}$ seqeunce, in which the second bit of $\mathbf{e}$ is a 1 and the rest are zeros, i.e., $\mathbf{r} = \mathbf{e} = [0, 1, 0, 0, 0, 0, 0]^T$. The corresponding syndrome vector, from the set of equations shown on the prior slide, equals $\mathbf{s} = [0, 1, 0]^T$.
- Take one more example $\mathbf{e}$ seqeunce, in which the third bit of $\mathbf{e}$ is a 1 and the rest are zeros, i.e., $\mathbf{r} = \mathbf{e} = [0, 0, 1, 0, 0, 0, 0]^T$. The corresponding syndrome vector, from the set of equations shown on the prior slide, equals $\mathbf{s} = [1, 1, 0]^T$.
- Repeating the above calculations, we obtain the table on the next slide

Application of the Properties of the Linear Block Codes

Decoding of $(7, 4)$ Hamming Code using Property 4

- An observation: the location of 1 in **r** vector and the decimal number corresponding to the three bit **s** vector are identical!
- This observation leads to the decoding algorithm on the next slide

r1	r2	r3	r4	r5	r6	r7	s1	s2	s3
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0
0	1	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	1	1	0
0	0	0	1	0	0	0	0	0	1
0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	1	0	0	1	1
0	0	0	0	0	0	1	1	1	1

Application of the Properties of the Linear Block Codes

Decoding of $(7, 4)$ Hamming Code using Property 4

- Decoding rule:
  1. Receive a 7-bit sequence $\mathbf{r}$.
  2. Calculate the syndrome $\mathbf{s}$ using the set of equations on the prior slide
     - $\rightarrow$ These equations are implemented using $\mathbf{H}$ matrix to obtain $\mathbf{s} = \mathbf{H}\,\mathbf{r}$
  3. Take the integer value corresponding to the three-bit $\mathbf{s}$ vector
  4. Flip the bit of $\mathbf{r}$ at the address corresponding to this integer value
  5. Read the message bits from this recovered vector
- If there is only $t_c = 1$ bit of error, this decoder will recover the original transmitted vector

Application of the Properties of the Linear Block Codes

Decoding of $(7, 4)$ Hamming Code using Property 4

- This exercise shows to us the arrangement of the Hamming Spheres for $(7, 4)$ Hamming Code
- There are 2^4 spheres corresponding to 2^4 valid codewords
- There is $\binom{n}{0} = 1$ received vector $\mathbf{r}$ at the center of each sphere. For this $\mathbf{r}$, the vector $\mathbf{e}$ is an all-zero vector, i.e., $\mathbf{r} = \mathbf{c}$
- There is $\binom{n}{1} = 7$ received vectors $\mathbf{r}$ around this center of the sphere. These are the vectors (shown in the table earlier) with one 1 and six 0's.
- The volume of the sphere, therefore, equals $V(n, t_c) = \binom{n}{0} + \binom{n}{1} = 1 + 7 = 8$, i.e., each sphere contains a total of 8 $\mathbf{r}$ vectors
  - $\rightarrow$ Recall that you were asked to evaluate $V(n, t_c)$ in the midterm exam
- Consider the product $2^k \times V(n, t_c)$. This equals $16 \times 8 = 128 = 2^7 = 2^n$. Thus, for the Hamming Code, $2^k \times V(n, t_c) = 2^n$.

Perfect Codes

- It is seen that 2^k non-overlapping Hamming Spheres, each of radius t_c bits, together encompass each of 2^n received vectors $\mathbf{r}$ (none is left out)
  - → This is rare; for typical codewords $2^k \times V(n, t_c) < 2^n$, i.e., some of $\mathbf{r}$ vectors cannot be included in any of 2^k Hamming Spheres
- The codes that satisfy the equality $2^k \times V(n, t_c) = 2^n$ are known as the Perfect Codes.
  - → There are only two other perfect codes besides the Hamming Code — the repetition codes with odd values of n and the Golay code. No other coding scheme is possible which is perfect.

Standard Array Decoding

- We conclude the topic of channel coding by mentioning the standard array based decoding of the LBC
- The idea is to enumerate, i.e., make the list of, all the possible received vectors and segregate them in different Hamming Spheres of radius t_c bits around each of 2^k codewords
- The result is known as the standard array; it's an array or a table, in which all 2^n possible received vectors $\mathbf{r}$ are listed. The table has 2^k columns. The head of each column is a valid codeword $\mathbf{c}$. The remaining entries in that column correspond to nonvalid $\mathbf{r}$ vectors that belong to the Hamming sphere corresponding to the column-head (i.e., a valid codeword)
- A table lookup is performed on receiving a vector $\mathbf{r}$. The column to which this $\mathbf{r}$ belongs in the standard array is looked up. The decoded codeword is the head of that column.
- A problem with this decoding scheme: the memory requirement explodes with increasing n

A Comparision with the Iterative Methods

- We have seen two decoding methods: the syndrome decoding and the standard array decoding
- Both these methods provide good insights into the properties of the LBCs, and they are useful for theory and practice of error correction.
- However, a shortcoming of these methods is that they do not scale well. If the block length n increases, or if the coding method changes, the memory requirements increase exponentially
- A powerful class of algorithms that has gained a lot of importance (relatively recently) uses the iterative methods of decoding (and probabilistic estimation, in general)