

Understanding How Information Theory Shapes Our Digital World

Brian Kurkoski
kurkoski@jaist.ac.jp

Japan Advanced Institute of Science and Technology

November 15, 2024



北陸先端科学技術大学院大学

The Role of Information Theory in Today's Digital World

Information theory is the mathematical foundation of efficiently transmitting, compressing, and processing data, enabling reliable communication in our digital world.

- ▶ **Foundation of Modern Communication:** Enables reliable data transmission over noisy channels, making global communication possible.
- ▶ **Data Compression:** Powers file compression (e.g., ZIP, JPEG), saving storage and bandwidth by encoding information more efficiently.
- ▶ **Data Security and Encryption:** Supports encryption protocols, ensuring secure, private communication over the internet.
- ▶ **Machine Learning and AI:** Provides mathematical tools for data processing, crucial in pattern recognition and decision-making systems.

The Role of Information Theory in Today's Digital World

- ▶ Claude Shannon established the mathematical foundation of information theory in the 1940s,
 - ▶ Proved fundamental limits of reliable transmission and compression
 - ▶ “Einstein of information”
 - ▶ Since then, many researchers built on his work
- ▶ Proof of fundamental limits did not tell us how to build practical codes.
- ▶ Recently, we have practical and efficient algorithms that bring Shannon's theory into everyday applications

Outline

In this talk, consider two everyday applications of information theory:

1. Reliable communications.
2. Data compression.

Videos:

- ▶ Will show parts of videos during my presentation.
- ▶ Produced by the IEEE Information Theory Society
- ▶ Videos I am using are available on YouTube:
 - ▶ “Hamming and Low Density Parity Check Codes”
 - ▶ “The Beauty of Lempel-Ziv Compression”

Part 1: Reliable Communications on Unreliable Channels



SEND_\$100



SEND_\$500

- ▶ Often we send important information
- ▶ One error can completely change the meaning
- ▶ Everyday we use wireless communications and data storage, and it is very reliable.
- ▶ Why is it so reliable?

Many Types of Noisy Communications

In communication systems, noisy corruptions will certainly occur.

Examples of **wireless** communication systems:

- ▶ Mobile phone 4G, 5G, etc.
- ▶ WiFi
- ▶ Satellite communications
- ▶ Digital TV (Digital video broadcast, DVB)

Examples of **wired** communication systems:

- ▶ Fiber optic communication systems
- ▶ Wired ethernet

Examples of **data storage** systems:

- ▶ Flash memories and SSDs
- ▶ Hard disk drives (Cloud storage)

 All channels have noise

Corruption or Noise Causes Errors



1010101



1110100

One model of noisy corruption:

- ▶ Information are expressed as bits 0 and 1.
- ▶ The information is transmitted over a noisy channel.
- ▶ Noise may change 0 to 1. Or 1 may change to 0. This change is random.

Another Kind of Corruption: Symbols are Erased



1110100010



11??100?1?

Another model of noisy corruption:

- ▶ Some information is erased.
- ▶ Bits 0 and 1 replaced with an unknown symbol “?”
- ▶ This is called an **erasure**.

1110100010



11??100?1?

Adding Redundancy

- ▶ Errors can be reduced by using redundancy¹
- ▶ Language naturally has redundancy

Suppose you received the following message, where some symbols have been erased:

DO YO_ HA_E ANY QUES_IO_S?

Because of the redundancy of language, you can see the original message is:

¹redundancy: inclusion of extra components which are not strictly necessary to functioning, in case of failure in other components

Adding Redundancy

- ▶ Errors can be reduced by using redundancy¹
- ▶ Language naturally has redundancy

Suppose you received the following message, where some symbols have been erased:

DO YO_ HA_E ANY QUES_IO_S?

Because of the redundancy of language, you can see the original message is:

DO YOU HAVE ANY QUESTIONS?

¹redundancy: inclusion of extra components which are not strictly necessary to functioning, in case of failure in other components

Adding Redundancy with a Repeat Code

Alice sends the message 1010101 to Bob. She adds redundancy by sending it twice:

Alice



1010101 1010101

Bob



?01010? 10??10?

- ▶ Some bits are erased, randomly.
- ▶ If a bit is erased in one position, Bob can **recover** it
- ▶ If a bit is erased in both positions, Bob **cannot recover** it

?01010?

10??10?



101010?

Code Rate and Single-Parity Check Code

The code rate R is an important part of an error-correcting code

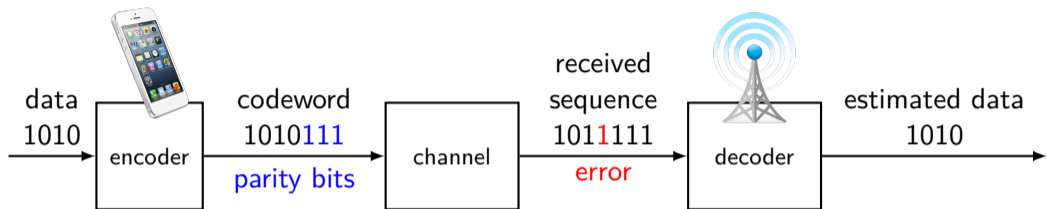
$$\text{Code rate } R = \frac{\# \text{ of message bits}}{\# \text{ of codeword bits}}$$

There is code rate tradeoff:

- ▶ High code rate R carries more information.
- ▶ Low rate R is more reliable.

 Code Rate

Error-Correcting Codes: Adding Redundancy Using Parity Bits



The communication system is modeled by three parts:

- ▶ An **encoder** adds parity bits (or redundancy) — the message is now longer
- ▶ The **channel** randomly adds errors
- ▶ The **decoder** recovers the original message — if there are not too many errors.

Parity-Check Code


- ▶ A **parity code** has one parity bit.
- ▶ Choose that parity bit so the codeword has **an even number of ones**

For example, if the information is 11001, it has 3 ones, an odd number.

Then the **parity bit** is 1:

codeword = 11001**1**

so that the codeword has 4 ones, an **even number**.

 Parity-check code

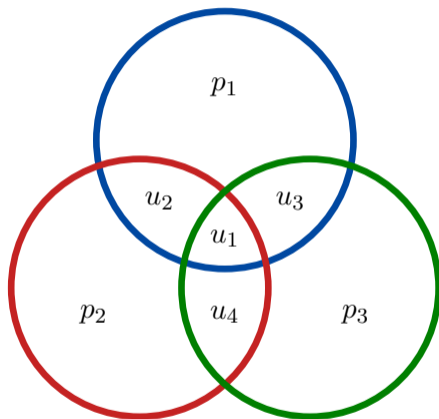
Error-Correcting Code: Hamming Code

Graphical representation of a Hamming code.

- ▶ Data bits are u_1, u_2, u_3, u_4
- ▶ Parity bits are p_1, p_2, p_3 .
- ▶ Codeword is
$$\mathbf{c} = [u_1, u_2, u_3, u_4, p_1, p_2, p_3]$$

The seven code bits must satisfy the following condition:

The number of 1's inside each circle must be even.



Example of Encoding Hamming Code



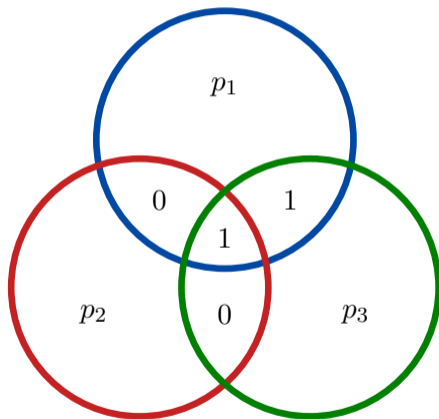
Alice wants to send:

$$u_1, u_2, u_3, u_4 = 1, 0, 1, 0$$

She chooses p_1, p_2, p_3 by following the rule:

The number of 1's inside each circle must be even.

What is p_1 ?



Example of Encoding Hamming Code



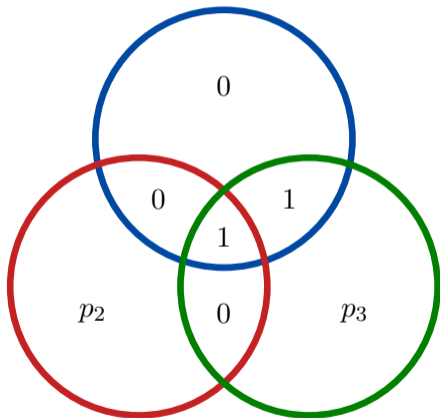
Alice wants to send:

$$u_1, u_2, u_3, u_4 = 1, 0, 1, 0$$

She chooses p_1, p_2, p_3 by following the rule:

The number of 1's inside each circle must be even.

What is p_1 ? What is p_2 ?



Example of Encoding Hamming Code



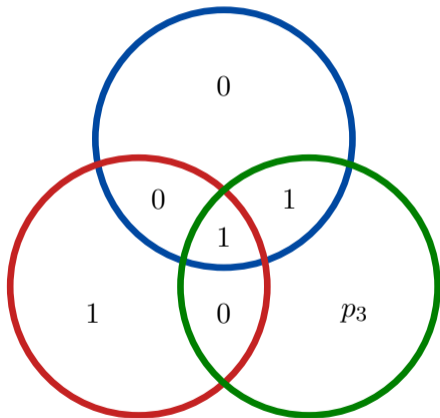
Alice wants to send:

$$u_1, u_2, u_3, u_4 = 1, 0, 1, 0$$

She chooses p_1, p_2, p_3 by following the rule:

The number of 1's inside each circle must be even.

What is p_1 ? What is p_2 ? What is p_3 ?



Example of Encoding Hamming Code



Alice wants to send:

$$u_1, u_2, u_3, u_4 = 1, 0, 1, 0$$

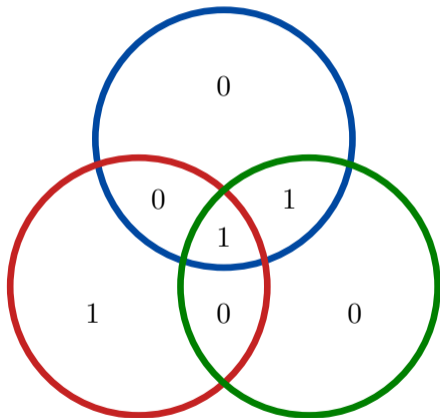
She chooses p_1, p_2, p_3 by following the rule:

The number of 1's inside each circle must be even.

What is p_1 ? What is p_2 ? What is p_3 ?

Alice sends:

$$1, 0, 1, 0, 0, 1, 0$$



Decoding



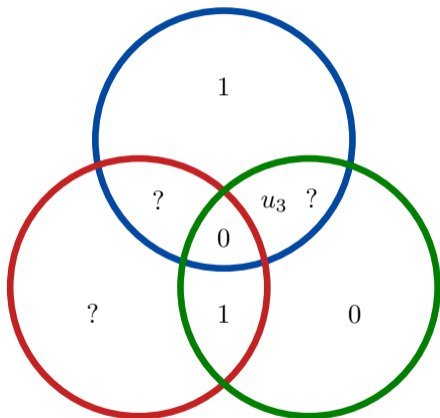
Bob receives a message

0, ?, ?, 1, 1, ?, 0

What was Alice's message?

He knows the decoding rule:

- ▶ Green circle has one "?" $u_3 =$



Decoding



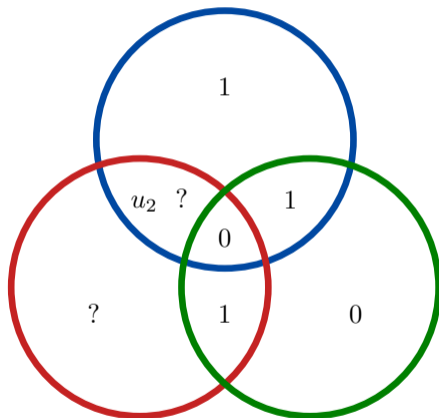
Bob receives a message

$0, ?, ?, 1, 1, ?, 0$

What was Alice's message?

He knows the decoding rule:

- ▶ **Green circle** has one "?" $u_3 = 1$
- ▶ **Blue circle** now has one "?" $u_2 =$



Decoding



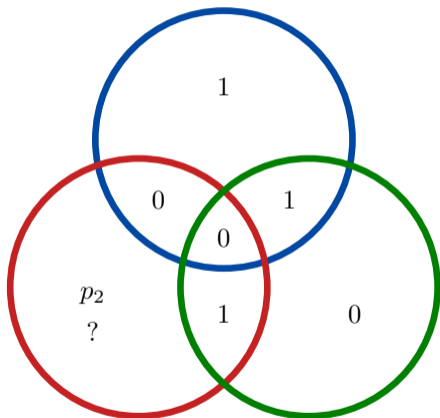
Bob receives a message

0, ?, ?, 1, 1, ?, 0

What was Alice's message?

He knows the decoding rule:

- ▶ **Green circle** has one "?" $u_3 = 1$
- ▶ **Blue circle** now has one "?" $u_2 = 0$
- ▶ **Red circle** now has one "?" $p_2 =$



Decoding



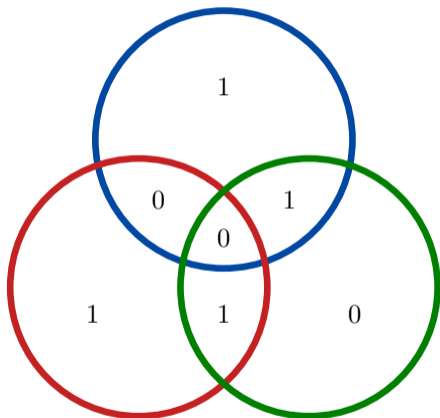
Bob receives a message

0, ?, ?, 1, 1, ?, 0

What was Alice's message?

He knows the decoding rule:

- ▶ **Green circle** has one "?" $u_3 = 1$
- ▶ **Blue circle** now has one "?" $u_2 = 0$
- ▶ **Red circle** now has one "?" $p_2 = 1$
- ▶ Codeword is 0, 0, 1, 1, 1, 1, 0 has been decoded!



What is an LDPC Code?

- ▶ Low-density parity-check (LDPC) codes are a type of error correcting code.
- ▶ LDPC codes have 1000s of bits. (The Hamming code has only 7 bits.)
- ▶ Each bit is a few parity checks — it is “low density” (i.e. one bit is inside only a few circles)
- ▶ The structure is not important. A **random** arrangement is good enough.

Key idea: LDPC codes use small, overlapping randomly assigned parity check sets.

 LDPC Codes

Conclusion of Part 1: Why Are LDPC Codes Important?

- ▶ **High Performance:** Achieves error rates close to the theoretical Shannon limit.
- ▶ **Efficient Decoding:** Enables fast, iterative decoding suitable for hardware implementation.
- ▶ **Widespread Applications:** Used in Wi-Fi, 5G, satellite communication, and data storage and many more applications.

Part 2: Managing the Explosion of Data in the Digital Age

▶ **Everyday Data Creation:**

- ▶ We are generating massive amounts of data daily—videos on social media, IoT devices, medical imaging and much more.
- ▶ Storing this data efficiently is essential.

▶ **Two Key Approaches for Data Storage:**

- ▶ **Larger Storage:** Using larger data storage systems.
- ▶ **Mathematical Methods:** Reducing the space data takes through compression techniques.

▶ **Mathematical Methods — Lossless Compression:**

- ▶ Ensures data can be compressed and decompressed without losing any information.



Who Won the Horse Race?

- ▶ Imagine a race of 8 horses.
- ▶ We want a code to send the winner of the race.
- ▶ Use as few bits as possible, on average.

Who Won the Horse Race?

- ▶ Imagine a race of 8 horses.
- ▶ We want a code to send the winner of the race.
- ▶ Use as few bits as possible, on average.

Simple code to transmit the winner

Adios	000
Big Brown	001
Cigar	010
Deep Impact	011
Easy Goer	100
Funny Cide	101
Go Man	110
Hyperion	111

This simple code requires 3 bits to transmit the winning horse.

Compressing Information

Using as few bits as possible, Alice sends the winner of the horse race.



Big Brown

001



Who is the winner?

Adios	000
Big Brown	001
Cigar	010
⋮	⋮
Hyperion	111

Using the same codebook, Bob can decode the message.

Compressing Information

Using as few bits as possible, Alice sends the winner of the horse race.



Big Brown

001



Big Brown

Adios	000
Big Brown	001
Cigar	010
⋮	⋮
Hyperion	111

Adios	000
Big Brown	001
Cigar	010
⋮	⋮
Hyperion	111

Using the same codebook, Bob can decode the message.

Odds of Winning

- ▶ Now, suppose we know the probability of winning.
- ▶ High-probability horse has a short codeword
- ▶ Low-probability horse has a long codeword
- ▶ The **average length** is now 2 bits, less than 3 bits!

		variable-length code
Adios	$\frac{1}{2}$	0
Big Brown	$\frac{1}{4}$	10
Cigar	$\frac{1}{8}$	110
Deep Impact	$\frac{1}{16}$	1110
Easy Goer	$\frac{1}{64}$	111100
Funny Cide	$\frac{1}{64}$	111101
Go Man	$\frac{1}{64}$	111110
Hyperion	$\frac{1}{64}$	111111

Compressing Information — Shorter Codebook

As before, Alice can send the message to Bob, now using 2 bits per message on average.



Big Brown

10



Who is the winner?

Adios	0
Big Brown	10
Cigar	110
⋮	⋮
Hyperion	111111

Bob and Alice must use the same codebook.

Compressing Information — Shorter Codebook

As before, Alice can send the message to Bob, now using 2 bits per message on average.



Big Brown

10



Big Brown

Adios	0
Big Brown	10
Cigar	110
⋮	⋮
Hyperion	111111

Adios	0
Big Brown	10
Cigar	010
⋮	⋮
Hyperion	111111

Bob and Alice must use the same codebook.

Best Codebook Depends on the “Conversation”

Codebook for $\frac{1}{8}, \frac{1}{8}, \dots, \frac{1}{8}$

Adios	000
Big Brown	001
Cigar	010
Deep Impact	011
Easy Goer	100
Funny Cide	101
Go Man	110
Hyperion	111

Codebook for $\frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{64}$

Adios	$\frac{1}{2}$	0
Big Brown	$\frac{1}{4}$	10
Cigar	$\frac{1}{8}$	110
Deep Impact	$\frac{1}{16}$	1110
Easy Goer	$\frac{1}{64}$	111100
Funny Cide	$\frac{1}{64}$	111101
Go Man	$\frac{1}{64}$	111110
Hyperion	$\frac{1}{64}$	111111

Two approaches to the codebook:

- ▶ Use a general codebook for a language — less compression, always works
- ▶ Build a dedicated codebook for a specific conversation — better compression, but works for that conversation.

 General codebook vs. specific codebook

Lempel-Ziv Compression

▶ **What is Lempel-Ziv Compression?**

- ▶ A lossless data compression algorithm developed by Abraham Lempel and Jacob Ziv.
- ▶ Forms the basis for widely used compression formats, such as ZIP and PNG.

▶ **How it Works:**

- ▶ Build a new codebook for each message we send
- ▶ Replaces repeated patterns or sequences with shorter references.
- ▶ Builds a codebook “on the fly”
- ▶ The encoder and decoder build the same codebook, but the codebook is never transmitted.

Sending Message and Codebook

Main idea of Lempel-Ziv compression:

- ▶ Alice should build a dedicated codebook for every message.
- ▶ Build a dedicated codebook by looking at the data
- ▶ But, Alice must send the codebook to Bob.
- ▶ The dedicated codebook works very well, better than a general codebook.


 Sending message and codebook

Lempel-Ziv Compression – Key Idea

Key idea is to rethink how the codebook is constructed:

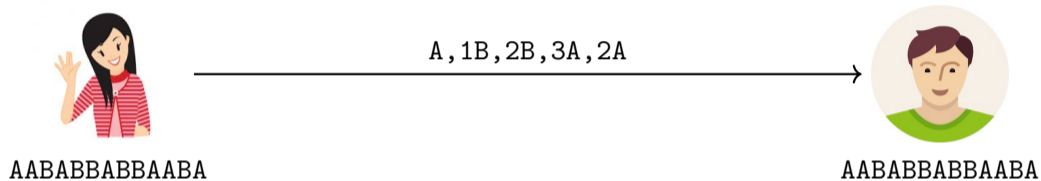
- ▶ The codebook and the message are built together “on the fly”
- ▶ As the message is scanned, the Alice gradually builds the codebook
- ▶ The compressed message is created at the same time
- ▶ Alice only needs to send the compressed message.
- ▶ Bob has the ability to create the same codebook as Alice, from the compressed message.

The codebook is not transmitted!

 Extra Insight of LZ

Conclusion: Lempel-Ziv Compression

The compressed message looks like this:



- ▶ Lossless compression preserves the original data, allowing exact reconstruction after decompression
- ▶ Lempel-Ziv compression achieves high compression rates by building codebooks on the fly.
- ▶ Widely used in many compression formats: ZIP, PNG, GZIP, 7z, RAR

Conclusion: The Impact of Information Theory

- ▶ **Provides the foundation for reliable communication in a noisy world**
 - ▶ Codes like LDPC codes provide accurate data transmission, even over unreliable channels.
 - ▶ Supports technologies like mobile networks, internet protocols, and satellite communication.
- ▶ **Enables efficient data storage through powerful compression techniques**
 - ▶ Reduces data size without losing information, optimizing storage space.
 - ▶ Techniques like Lempel-Ziv powers formats like ZIP, JPEG, and video streaming, making digital storage manageable.
- ▶ **Shaping Our Digital World:**
 - ▶ Information theory is crucial in making today's digital lifestyle possible.
 - ▶ Continues to advance technology, making data handling more reliable and space-efficient.