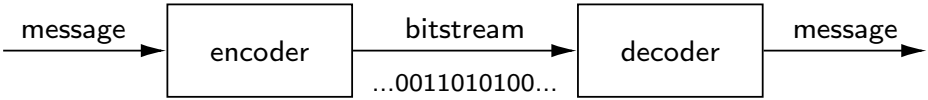
Variable-Length Codes

letter	codeword
A	00
B	01
M	10
N	11

letter	codeword
A	011
B	01
M	0
N	111

letter	codeword
A	0
B	110
M	111
N	10

Mathematical Description of Source Coding



Transmission of new information to receiver

- Message is unknown by receiver

➡ Source can be modeled as a **random process**

Modeling of information sources as random processes

➡ Description using mathematical framework of probability theory

➡ Requires reasonable assumptions with respect to source of information

➡ Characterization of performance by probabilistic averages

➡ Basis for **mathematical theory of communication**

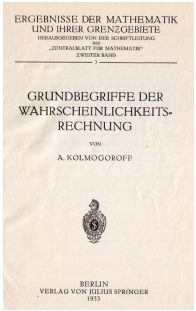
Probability Axioms

Random experiment: Any experiment with uncertain outcome ζ

Sample space $\mathcal{O}$: Union of all possible outcomes ζ (also called **certain event** $\mathcal{O}$)

Event $\mathcal{A}$: Union of zero or more possible outcomes ζ ($\mathcal{A} \subseteq \mathcal{O}$)

Probability $P(\mathcal{A})$: Measure $\mathrm{P}(\mathcal{A})$ assigned to events $\mathcal{A}$ of a random experiment that satisfies the following axioms (KOLMOGOROV):


ERGEBNISSE DER MATHEMATIK
UND IHRER GRENZGEBIETE
HERAUSGEGEBEN VON DER SCHRIFTLEITUNG
DER
„ZENTRALBLATT FÜR MATHEMATIK"
ZWEITER BAND

GRUNDBEGRIFFE DER
WAHRSCHEINLICHKEITS-
RECHNUNG
VON
A. KOLMOGOROFF

BERLIN
VERLAG VON JULIUS SPRINGER
1933

1. Probabilities are non-negative real numbers

$$\mathrm{P}(\mathcal{A}) \geq 0, \quad \forall \mathcal{A} \subseteq \mathcal{O}$$

2. Certain event $\mathcal{O}$ has a probability equal to 1

$$\mathrm{P}(\mathcal{O}) = 1$$

3. Probability of two disjoint events $\mathcal{A}$ and $\mathcal{B}$

$$\mathcal{A} \cap \mathcal{B} = \emptyset \quad \Longrightarrow \quad \mathrm{P}(\mathcal{A} \cup \mathcal{B}) = \mathrm{P}(\mathcal{A}) + \mathrm{P}(\mathcal{B})$$

Conditional Probability and Independence of Events

Conditional Probability $P(\mathcal{A} \,|\, \mathcal{B})$ (Kolmogorov)

- Probability of an event $\mathcal{A}$ given that another event $\mathcal{B}$ has occured

$$\mathrm{P}(\mathcal{A} \,|\, \mathcal{B}) = \frac{\mathrm{P}(\mathcal{A} \cap \mathcal{B})}{\mathrm{P}(\mathcal{B})}, \quad \text{for } \mathrm{P}(\mathcal{B}) > 0$$

➡ **Bayes' Theorem**

$$\mathrm{P}(\mathcal{A} \,|\, \mathcal{B}) = \mathrm{P}(\mathcal{B} \,|\, \mathcal{A}) \cdot \frac{\mathrm{P}(\mathcal{A})}{\mathrm{P}(\mathcal{B})}, \quad \text{for } \mathrm{P}(\mathcal{A}) > 0,\ \mathrm{P}(\mathcal{B}) > 0$$

Independence of Events

- Two events $\mathcal{A}$ and $\mathcal{B}$ are said to be **independent** if and only if

$$\mathrm{P}(\mathcal{A} \cap \mathcal{B}) = \mathrm{P}(\mathcal{A}) \cdot \mathrm{P}(\mathcal{B})$$

➡ For independent events $\mathcal{A}$ and $\mathcal{B}$, with $P(\mathcal{B}) > 0$, we have

$$\mathrm{P}(\mathcal{A} \,|\, \mathcal{B}) = \mathrm{P}(\mathcal{A})$$

Probability Estimation

Empirical Probability

- Repeatable random experiment
- Relative frequency of an event $\mathcal{A}$ in N trials

$$\frac{N(\mathcal{A})}{N} = \frac{\text{number of trials in which } \mathcal{A} \text{ was observed}}{\text{number of total trials}}$$

➡ Empirical probability

$$\mathrm{P}(\mathcal{A}) = \lim_{N \to \infty} \frac{N(\mathcal{A})}{N}$$

Practical Probability Estimation

- Use the approximation

$$\mathrm{P}(\mathcal{A}) = \frac{N(\mathcal{A})}{N}$$

➡ Estimation quality depends on the number of trials N

Random Variables

Random Variable

- Function $X(\zeta)$ of the sample space $\mathcal{O}$ that assigns a real value $x = X(\zeta)$ to each possible outcome $\zeta \in \mathcal{O}$ of a random experiment
- A random variable may take ...
  - a finite number of values
  - a countable infinite number of values
  - an uncountable number of values

Examples for Random Variables

- Dice roll: Number on top face of the die (finite)
- Roulette: Number of pocket the ball lands (finite)
- Microphone: Voltage on output of microphone (uncountable)
- Digital signal: Value of next sample (finite)

Cumulative Distribution Function

Cumulative Distribution Function (cdf)

- Cumulative distribution function $F_X(x)$ of a random variable X

$$F_X(s) = \mathrm{P}(X \leq x) = \mathrm{P}(\{\zeta : X(\zeta) \leq x\})$$

- $F_X(x)$ is also referred to as **distribution** of the random variable X

Joint and Conditional Cumulative Distribution Functions

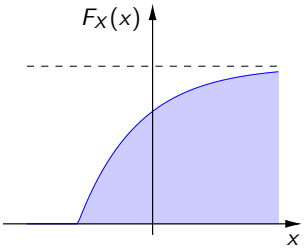
- Joint cdf of two random variables X and Y

$$F_{XY}(x, y) = \mathrm{P}(X \leq x, Y \leq y)$$

- Conditional cdf of a random variable X given another random variable Y

$$F_{X|Y}(x \,|\, y) = \mathrm{P}(X \leq x \,|\, Y \leq y) = \frac{\mathrm{P}(X \leq x, Y \leq y)}{\mathrm{P}(Y \leq y)} = \frac{F_{XY}(x, y)}{F_Y(y)}$$

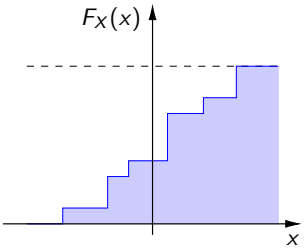
Examples: Cumulative Distribution Functions



Continuous function

- Random variable X can take all values inside one or more non-zero intervals

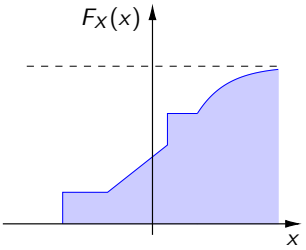
➡ **Continuous random variable**



Staircase function

- Random variable X can only take a countable number of values

➡ **Discrete random variable**



Mixed type

- Random variable X can take all values inside one or more non-zero intervals and a countable number of additional values

Discrete Random Variables

Discrete Random Variables

- A random variable X is called a **discrete random variable** if and only if its cdf $F_X(x)$ is a staircase function

➡ Discrete random variables X can only take values of a countable alphabet

$$\mathcal{A}_X = \{x_0, x_1, x_2, \cdots\}$$

Examples for Discrete Random Variables

- Result of a coin toss: $\mathcal{A}_X = \{0, 1\}$ (0: "head", 1: "tail")
- Number on top face of the die: $\mathcal{A}_X = \{1, 2, 3, 4, 5, 6\}$
- Sample in an 8-bit gray image: $\mathcal{A}_X = \{0, 1, 2, \cdots, 255\}$
- Sample in a 16-bit audio signal: $\mathcal{A}_X = \{-32768, -32767, \cdots, -1, 0, 1, \cdots, 32766, 32767\}$

Probability Mass Function

Probability Mass Function (pmf)

- Probability mass function $p_X(x)$ of discrete random variable X with alphabet $\mathcal{A}_X$

$$p_X(x) = \mathrm{P}(X = x) = \mathrm{P}(\,\{\zeta \in \mathcal{O} : \, X(\zeta) = x\}\,) \quad \text{for} \quad x \in \mathcal{A}_X$$

- Pmfs have the following property

$$\sum_{x \in \mathcal{A}_X} p_X(x) = \mathrm{P}(\mathcal{O}) = 1$$

Joint and Conditional Probability Mass Functions

- Joint pmf of two discrete random variables X and Y

$$p_{XY}(x, y) = \mathrm{P}(X = x, Y = y)$$

- Conditional pmf of a discrete random variable X given another discrete random variable Y

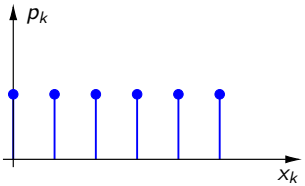
$$p_{X|Y}(x \,|\, y) = \mathrm{P}(X = x \,|\, Y = y) = \frac{\mathrm{P}(X = x, Y = y)}{\mathrm{P}(Y = y)} = \frac{p_{XY}(x, y)}{p_Y(y)}$$

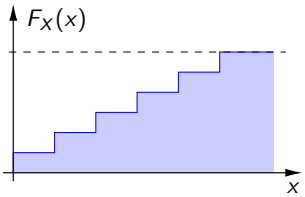
Examples for Discrete Distributions

Uniform

$$p_k = \frac{1}{M}$$

$$(0 \leq k < M)$$

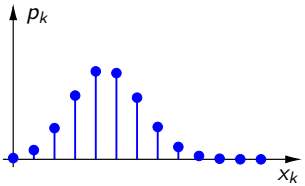


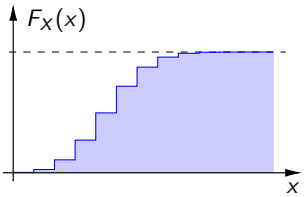


Binomial

$$p_k = \binom{n}{k} p^k (1-p)^{n-k}$$

$$(0 \leq k \leq n)$$

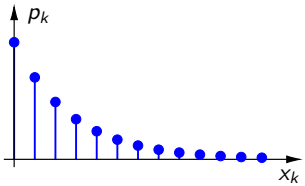


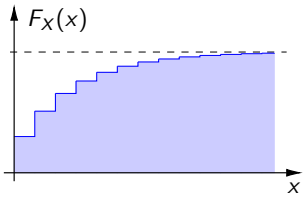


Geometric

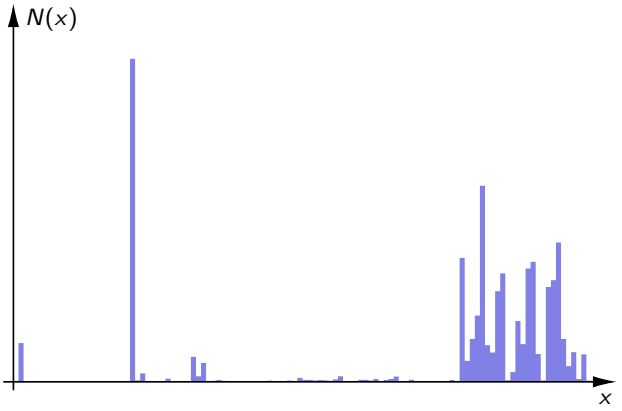
$$p_k = (1-p)^k \, p$$

$$(k \geq 0)$$





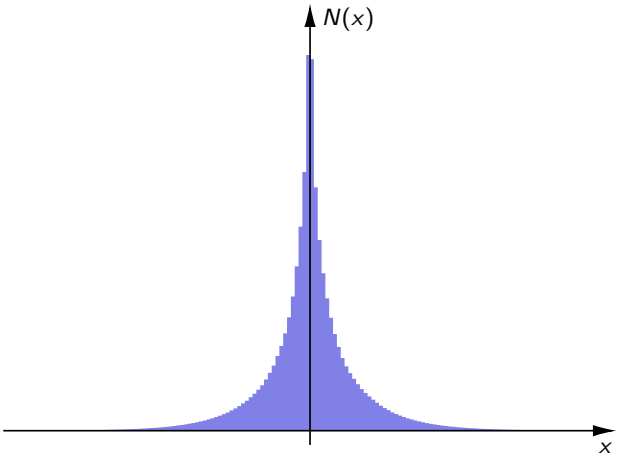
Example: 1D Histogram for English Text



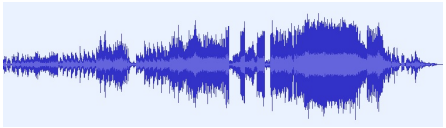
Large English text
(ca. 6 million characters)

```
THE ADVENTURES OF
SHERLOCK HOLMES

BY

SIR ARTHUR CONAN DOYLE

CONTENTS

I. A Scandal in Bohemia
II. The Red-Headed League
III. A Case of Identity
IV. The Boscombe Valley Mystery
V. The Five Orange Pips
VI. The Man with the Twisted Lip
VII. The Adventure of the Blue Carbuncle
VIII. The Adventure of the Speckled Band
IX. The Adventure of the Engineer's Thumb
X. The Adventure of the Noble Bachelor
XI. The Adventure of the Beryl Coronet
XII. The Adventure of the Copper Beeches

...
```

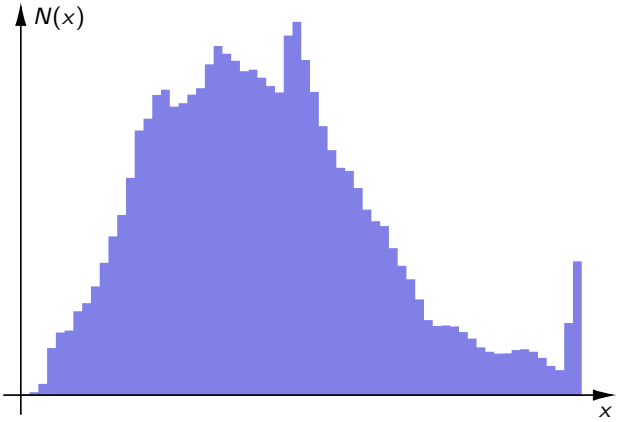
Example: 1D Histogram for Single-Channel Audio



Queen "Bohemian Rhapsody"
(ca. 15 million samples)



Example: 1D Histogram for Natural Gray-Level Images



15 test images (each 768×512)

Expected Values

Expected Values

- Expected value of a function $g(X)$ of a discrete random variable X with alphabet $\mathcal{A}_X$

$$\mathrm{E}\{\, g(X) \,\} = \mathrm{E}_X\{\, g(X) \,\} = \sum_{\forall x \in \mathcal{A}_X} g(x)\, p_X(x)$$

- Expected value of function $g(X, Y)$ of two discrete random variable X and Y

$$\mathrm{E}\{\, g(X, Y) \,\} = \mathrm{E}_{XY}\{\, g(X, Y) \,\} = \sum_{x,y} g(x, y)\, p_{XY}(x, y)$$

Conditional Expected Values

- Expected value of function $g(X)$ given an event $\mathcal{B}$ or another random variable Y

$$\mathrm{E}\{\, g(X) \,|\, \mathcal{B} \,\} = \sum_{x} g(x)\, p_{X|\mathcal{B}}(x \,|\, \mathcal{B}) \qquad \text{for } \mathrm{P}(\mathcal{B}) > 0$$

$$\mathrm{E}\{\, g(X) \,|\, Y \,\} = \sum_{x} g(x)\, p_{X|Y}(x \,|\, Y) \qquad \text{(another random variable)}$$

Properties of Expected Values

Important Properties

- Linearity of expected values

$$\mathrm{E}\{\, a\,X + b\,Y \,\} = a \cdot \mathrm{E}\{\, X \,\} + b \cdot \mathrm{E}\{\, Y \,\}$$

- For independent random variables X and Y

$$\mathrm{E}\{\, XY \,\} = \mathrm{E}\{\, X \,\}\, \mathrm{E}\{\, Y \,\}$$

- Iterative expectation rule

$$\mathrm{E}\{\, \mathrm{E}\{\, g(X) \,|\, Y \,\} \,\} = \mathrm{E}\{\, g(X) \,\}$$

Important Expected Values

- **Mean** μ_X of a random variable X

$$\mu_X = \mathrm{E}\{ X \} = \sum_x x \cdot p_X(x)$$

- **Variance** σ_X^2 of a random variable X

$$\sigma_X^2 = \mathrm{E}\Big\{ (X - \mathrm{E}\{ X \})^2 \Big\} = \sum_x (x - \mu_X)^2 \cdot p_X(x)$$

- **Covariance** σ_{XY}^2 of two random variables X and Y, and **correlation coefficient** ϕ_{XY}

$$\sigma_{XY}^2 = \mathrm{E}\Big\{ (X - \mathrm{E}\{ X \}) (Y - \mathrm{E}\{ Y \}) \Big\} = \sum_{x,y} (x - \mu_x)(y - \mu_y) \cdot p_{XY}(x, y)$$

$$\phi_{XY} = \frac{\sigma_{XY}^2}{\sqrt{\sigma_X^2 \cdot \sigma_Y^2}}$$

Random Processes

Discrete-Time Random Process

- Series of random experiments at time instants t_n, with $n = 0, 1, 2, \cdots$
- For each experiment: Random variable $X_n = X(t_n)$
- **Random process**: Series of random variables

$$\boldsymbol{X} = \{X_0, X_1, X_2, \cdots\} = \{X_n\}$$

Discrete-Time Discrete-Amplitude Random Process

- Random variables X_n are discrete random variables
- Each random variable X_n has an alphabet $\mathcal{A}_n$

➡ Type of random processes we consider for lossless coding

Statistical Properties of Random Processes

Characterization of Statistical Properties

- Consider N-dimensional random vector

$$\boldsymbol{X}_k^{(N)} = \{X_k, X_{k+1}, \cdots, X_{k+N-1}\}$$

- N-th order joint cdf

$$F_k^{(N)}(\boldsymbol{x}) = \mathrm{P}\left(\boldsymbol{X}_k^{(N)} \leq \boldsymbol{x}\right) = \mathrm{P}(X_k \leq x_0, X_{k+1} \leq x_1, \cdots, X_{k+N-1} \leq x_{N-1})$$

- N-th order joint pmf

$$p_k^{(N)}(\boldsymbol{x}) = \mathrm{P}\left(\boldsymbol{X}_k^{(N)} = \boldsymbol{x}\right) = \mathrm{P}(X_k = x_0, X_{k+1} = x_1, \cdots, X_{k+N-1} = x_{N-1})$$

- Also: Conditional cdfs and conditional pmfs

Models for Random Processes

Stationary Random Processes

- Statistical properties are invariant to a shift in time
- In this course: Typically restrict our considerations to stationary processes

Memoryless Random Processes

- All random variables X_n are independent of each other

Independent and Identically Distributed (IID) Random Processes

- Random processes that are **stationary** and **memoryless**
- Valid model for fair games: Dice roll or roulette

Markov Processes

- Markov property: Future outcomes do only depend on present outcome, but not on past outcomes

$$\mathrm{P}(X_n = s_n \,|\, X_{n-1} = x_{n-1}, X_{n-2} = x_{n-2}, \cdots) = \mathrm{P}(X_n = x_n \,|\, X_{n-1} = x_{n-1})$$

- Simple model for random processes with memory

Stationary Discrete Markov Processes

Stationary Discrete Random Process with Markov Property

- Simple model for investigating coding of sources with memory
- Statistical properties are completly specified by 1-st order conditional cdf or pmf

$$F(x_n \,|\, x_{n-1}) \;=\; \mathrm{P}(X_n \leq x_n \,|\, X_{n-1} \leq x_{n-1})$$

$$p(x_n \,|\, x_{n-1}) \;=\; \mathrm{P}(X_n = x_n \,|\, X_{n-1} = x_{n-1})$$

- Extension: N-th order stationary discrete Markov processes

Example: Stationary Discrete Markov Process

$\mathcal{A}_X = \{a, b, c\}$

conditional pmf
$p(x_n \,|\, x_{n-1})$

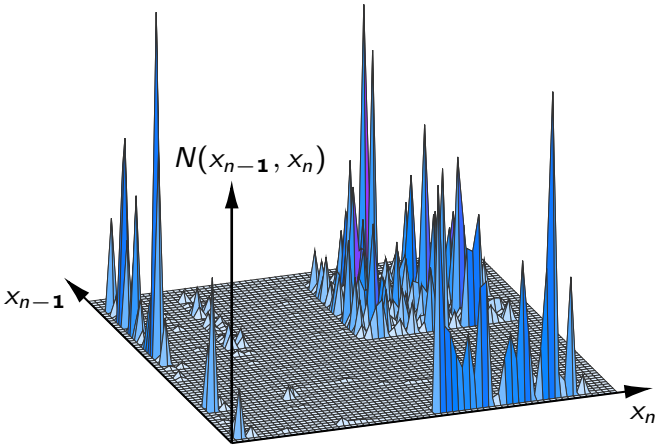
x_n	$p(x_n \,|\, a)$	$p(x_n \,|\, b)$	$p(x_n \,|\, c)$
a	0.90	0.15	0.25
b	0.05	0.80	0.15
c	0.05	0.05	0.60

Question:

What is the marginal pmf $p_X(x)$?

Example: 2D Histogram for English Text

joint histogram of two adjacent characters

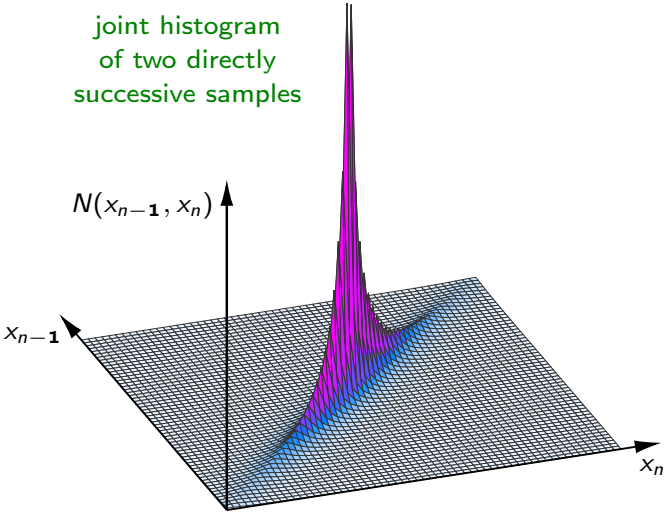


Large English upper-case text
(ca. 6 million characters)

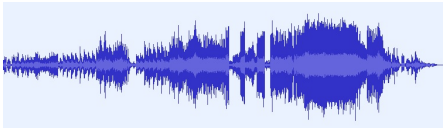
```
THE ADVENTURES OF
SHERLOCK HOLMES

BY

SIR ARTHUR CONAN DOYLE

CONTENTS

I. A SCANDAL IN BOHEMIA
II. THE RED-HEADED LEAGUE
III. A CASE OF IDENTITY
IV. THE BOSCOMBE VALLEY MYSTERY
V. THE FIVE ORANGE PIPS
VI. THE MAN WITH THE TWISTED LIP
VII. THE ADVENTURE OF THE BLUE CARBUNCLE
VIII. THE ADVENTURE OF THE SPECKLED BAND
IX. THE ADVENTURE OF THE ENGINEER'S THUMB
X. THE ADVENTURE OF THE NOBLE BACHELOR
XI. THE ADVENTURE OF THE BERYL CORONET
XII. THE ADVENTURE OF THE COPPER BEECHES

...
```

Example: 2D Histogram for Single-Channel Audio

joint histogram
of two directly
successive samples

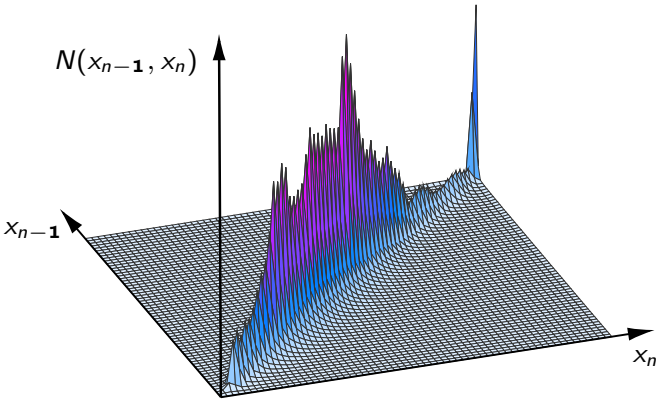


Queen "Bohemian Rhapsody"
(ca. 15 million samples)



Example: 2D Histogram for Natural Gray-Level Images

joint histogram
of two horizontally
adjacent samples



15 test images (each 768×512)

Summary of Mathematical Basics

Probability

- Axiomatic definition, empirical probability
- Conditional probability and independence of events

Discrete Random Variables

- Can take only values of a countable alphabet
- Cumulative distribution function (cdf): Staircase function
- Probability mass function (pmf)
- Expected values: Mean, variance, covariance

Discrete Random Variables

- Sequence of random variables: Model for sources of digital signals
- Types of random processes: Stationary, memoryless, iid, Markov
- Stationary discrete Markov processes: Simple model for sources with memory

Morse Code (first version around 1837)

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

- Example message: $\boldsymbol{s} =$ "BANANAMAN"

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

- Example message: $\boldsymbol{s} =$ "BANANAMAN"
- ➡ Bitstream (code A): $\boldsymbol{b} =$ "010011001100100011" (18 bits)

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

- Example message: $\boldsymbol{s} =$ "BANANAMAN"
- ➡ Bitstream (code A): $\boldsymbol{b} =$ "0100110011001000 11" (18 bits)

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

- Example message: $\boldsymbol{s} =$ "BANANAMAN"
- ➡ Bitstream (code A): $\boldsymbol{b} =$ "010011001100100011" (18 bits)
- ➡ Bitstream (code B): $\boldsymbol{b} =$ "10001000100010100100" (20 bits)

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

- Example message: $\boldsymbol{s} =$ "BANANAMAN"
- ➡ Bitstream (code A): $\boldsymbol{b} =$ "010011001100100011" (18 bits)
- ➡ Bitstream (code B): $\boldsymbol{b} =$ "10001000100010100100" (20 bits)

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

- Example message: $\boldsymbol{s} =$ "BANANAMAN"
- ➡ Bitstream (code A): $\boldsymbol{b} =$ "010011001100100011" (18 bits)
- ➡ Bitstream (code B): $\boldsymbol{b} =$ "10001000100010100100" (20 bits)
- ➡ Bitstream (code C): $\boldsymbol{b} =$ "1100100100111010" (16 bits)

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

- Example message: $\boldsymbol{s} =$ "BANANAMAN"

➡ Bitstream (code A): $\boldsymbol{b} =$ "010011001100100011" (18 bits)

➡ Bitstream (code B): $\boldsymbol{b} =$ "10001000100010100100" (20 bits)

➡ Bitstream (code C): $\boldsymbol{b} =$ "1100100100111010" (16 bits)

➡ **Goal: Minimize average codeword length**

$$\bar{\ell} = \mathrm{E}\{\,\ell(S)\,\} = \sum_k p_k \cdot \ell_k$$

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "010011001100100011" ➡ $\boldsymbol{s} =$ "

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} = \text{“}01001100110010011\text{“} \quad \rightarrow \quad \boldsymbol{s} = \text{“B}$

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} = \text{“}01001100110010011\text{“} \quad \Rightarrow \quad \boldsymbol{s} = \text{“BA}$

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} = \text{“}01001100100100011\text{“} \quad \Rightarrow \quad \boldsymbol{s} = \text{“BAN}$

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} = \text{“}01001100\mathrm{1}1001000\mathrm{11}\text{“}$ ➡ $\boldsymbol{s} = \text{“BANA}$

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100**11**00100011" ➡ $\boldsymbol{s} =$ "BANAN

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100100100011" ➡ $\boldsymbol{s} =$ "BANANA

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010 0011" ➡ $\boldsymbol{s} =$ "BANANAM

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} = \text{“}01001100110010\,00\,11\text{“}$ ➡ $\boldsymbol{s} = \text{“BANANAMA}$

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} = \text{“}01001100110010001\mathrm{1}\text{“}$ ➡ $\boldsymbol{s} = \text{“BANANAMA}\mathrm{N}$

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "10001000100010100100" ➡ $\boldsymbol{s} =$ "

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "1000100001000100100" ➡ $\boldsymbol{s} =$ "B or MN ..."

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "10001000100101000" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "1100100100111010" ➡ $\boldsymbol{s} =$ "

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010001 1" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100010010 0" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "1100100100111010" ➡ $\boldsymbol{s} =$ "B

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "10001000100010100100" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "11001001001110 10" ➡ $\boldsymbol{s} =$ "BA

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010001 1" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100101 00100" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "1100100100111010" ➡ $\boldsymbol{s} =$ "BAN

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010001 1" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100101 00" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "1100100100111010" ➡ $\boldsymbol{s} =$ "BANA

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010001 1" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100101 00100" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "11001001 00111010" ➡ $\boldsymbol{s} =$ "BANAN

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010001 1" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100101 00" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "110010010 0111010" ➡ $\boldsymbol{s} =$ "BANANA

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100100100" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "11001001001110 10" ➡ $\boldsymbol{s} =$ "BANANAM

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100100100" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "11001001001110 10" ➡ $\boldsymbol{s} =$ "BANANAMA

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100100100" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "11001001001110 10" ➡ $\boldsymbol{s} =$ "BANANAMAN"

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "100010000100010100" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "1100100100111010" ➡ $\boldsymbol{s} =$ "BANANAMAN"

Example: Variable-Length Coding for Scalars

- Symbol alphabet: $\mathcal{A} = \{\mathrm{A}, \mathrm{B}, \mathrm{M}, \mathrm{N}\}$

code A

letter	codeword
A	00
B	01
M	10
N	11

code B

letter	codeword
A	010
B	100
M	10
N	0

code C

letter	codeword
A	0
B	110
M	111
N	10

Decoding:

- Code A: $\boldsymbol{b} =$ "01001100110010011" ➡ $\boldsymbol{s} =$ "BANANAMAN"
- Code B: $\boldsymbol{b} =$ "10001000100101000" ➡ $\boldsymbol{s} =$ "B or MN ..."
- Code C: $\boldsymbol{b} =$ "1100100100111010" ➡ $\boldsymbol{s} =$ "BANANAMAN"

➡ **Necessary condition: Unique decodability:**

Each bitstream uniquely represents a single message!

Efficiency of Scalar Variable-Length Codes

Assumptions

- Messages: Finite-length realizations of a **stationary discrete random process** $\boldsymbol{S} = \{S_0, S_1, \cdots\}$
- Random variables $S_n = S$ have a countable alphabet $\mathcal{A} = \{a_0, a_1, a_2, \cdots\}$
- Marginal pmf $p_S(a)$ for the random variables S is known

$$p_k = p_S(a_k) = \mathrm{P}(S = a_k) \qquad \forall a_k \in \mathcal{A}$$

Efficiency of Scalar Variable-Length Codes

Assumptions

- Messages: Finite-length realizations of a **stationary discrete random process** $\boldsymbol{S} = \{S_0, S_1, \cdots\}$
- Random variables $S_n = S$ have a countable alphabet $\mathcal{A} = \{a_0, a_1, a_2, \cdots\}$
- Marginal pmf $p_S(a)$ for the random variables S is known

$$p_k = p_S(a_k) = \mathrm{P}(S = a_k) \qquad \forall a_k \in \mathcal{A}$$

Characterizing the Efficiency

- Codeword lengths ℓ_k: Function of the random variables S_n

$$\ell_k = \ell(a_k)$$

Efficiency of Scalar Variable-Length Codes

Assumptions

- Messages: Finite-length realizations of a **stationary discrete random process** $\boldsymbol{S} = \{S_0, S_1, \cdots\}$
- Random variables $S_n = S$ have a countable alphabet $\mathcal{A} = \{a_0, a_1, a_2, \cdots\}$
- Marginal pmf $p_S(a)$ for the random variables S is known

$$p_k = p_S(a_k) = \mathrm{P}(S = a_k) \qquad \forall a_k \in \mathcal{A}$$

Characterizing the Efficiency

- Codeword lengths ℓ_k: Function of the random variables S_n

$$\ell_k = \ell(a_k)$$

- Efficiency measure: **Average codeword length $\bar{\ell}$ per symbol**

$$\bar{\ell} = \mathrm{E}\{\, \ell(S) \,\} = \sum_{\forall a_k \in \mathcal{A}} \ell(a_k)\, p_S(a_k) = \sum_k \ell_k\, p_k$$

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	**code A**
a	0.5	0
b	0.25	10
c	0.125	11
d	0.125	11
	$\bar{\ell}$	1.5
	uniquely decodable?	

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	**code A**
a	0.5	0
b	0.25	10
c	0.125	11
d	0.125	11
	$\bar{\ell}$	1.5
	uniquely decodable?	**no** (***singular***)

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B
a	0.5	0	0
b	0.25	10	01
c	0.125	11	010
d	0.125	11	011
	$\bar{\ell}$	1.5	1.75
	uniquely decodable?	**no** (***singular***)	

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B
a	0.5	0	0
b	0.25	10	01
c	0.125	11	010
d	0.125	11	011
	$\bar{\ell}$	1.5	1.75
	uniquely decodable?	**no** (***singular***)	**no** (c=b,a)

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B	code C
a	0.5	0	0	0
b	0.25	10	01	01
c	0.125	11	010	011
d	0.125	11	011	111
	$\bar{\ell}$	1.5	1.75	1.75
	uniquely decodable?	**no** ***(singular)***	**no** (c=b,a)	

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B	code C
a	0.5	0	0	0
b	0.25	10	01	01
c	0.125	11	010	011
d	0.125	11	011	111
	$\bar{\ell}$	1.5	1.75	1.75
	uniquely decodable?	**no** (***singular***)	**no** (c=b,a)	**yes** (*delay*)

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B	code C	code D
a	0.5	0	0	0	00
b	0.25	10	01	01	01
c	0.125	11	010	011	10
d	0.125	11	011	111	110
	$\bar{\ell}$	1.5	1.75	1.75	2.125
	uniquely decodable?	**no** (***singular***)	**no** (c=b,a)	**yes** (*delay*)	

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B	code C	code D
a	0.5	0	0	0	00
b	0.25	10	01	01	01
c	0.125	11	010	011	10
d	0.125	11	011	111	110
	$\bar{\ell}$	1.5	1.75	1.75	2.125
	uniquely decodable?	**no** (***singular***)	**no** (c=b,a)	**yes** (*delay*)	**yes**

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B	code C	code D	code E
a	0.5	0	0	0	00	0
b	0.25	10	01	01	01	10
c	0.125	11	010	011	10	110
d	0.125	11	011	111	110	111
	$\bar{\ell}$	1.5	1.75	1.75	2.125	1.75
	uniquely decodable?	**no** (***singular***)	**no** (c=b,a)	**yes** (*delay*)	**yes**	

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B	code C	code D	code E
a	0.5	0	0	0	00	0
b	0.25	10	01	01	01	10
c	0.125	11	010	011	10	110
d	0.125	11	011	111	110	111
	$\bar{\ell}$	1.5	1.75	1.75	2.125	1.75
	uniquely decodable?	**no** (***singular***)	**no** (c=b,a)	**yes** (*delay*)	**yes**	**yes**

Construction of Lossless Codes

Design Goals for Lossless Codes

1. Minimize average codeword length $\bar{\ell}$
2. Retain **unique decodability** of arbitrarily long messages !

Code Examples

a_k	p_k	code A	code B	code C	code D	code E
a	0.5	0	0	0	00	0
b	0.25	10	01	01	01	10
c	0.125	11	010	011	10	110
d	0.125	11	011	111	110	111
	$\bar{\ell}$	1.5	1.75	1.75	2.125	1.75
	uniquely decodable?	**no** (***singular***)	**no** (c=b,a)	**yes** (*delay*)	**yes** (***instantaneous codes***)	**yes** (***instantaneous codes***)

Prefix Codes

Uniquely Decodable Codes

- Necessary condition: Non-singular codes

$$\forall a, b \in \mathcal{A} : a \neq b, \qquad \text{codeword}(a) \neq \text{codeword}(b)$$

➡ Not sufficient

➡ Require: **Each sequence of bits can only be generated by one possible sequence of source symbols**

Prefix Codes

Uniquely Decodable Codes

- Necessary condition: Non-singular codes

$$\forall a, b \in \mathcal{A} : a \neq b, \qquad \text{codeword}(a) \neq \text{codeword}(b)$$

➡ Not sufficient

➡ Require: **Each sequence of bits can only be generated by one possible sequence of source symbols**

Prefix Codes

- One class of uniquely decodable codes
- Property: No codeword for an alphabet letter represents the codeword or a prefix of the codeword for any other alphabet letter

➡ Obvious: Any concatenation of codewords can be uniquely decoded

- Also referred to as **prefix-free codes** or **instantaneous codes**

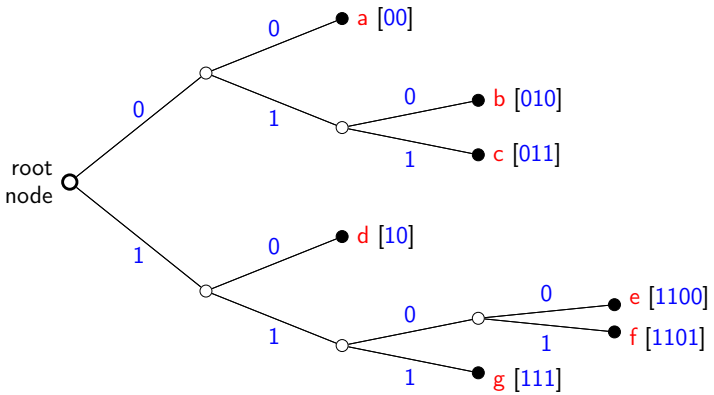
letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111

Binary Code Trees for Prefix Codes

Prefix codes can be represented as binary code trees

- Alphabet letters are assigned to terminal nodes
- Codewords are given by labels on path from the root to a terminal node

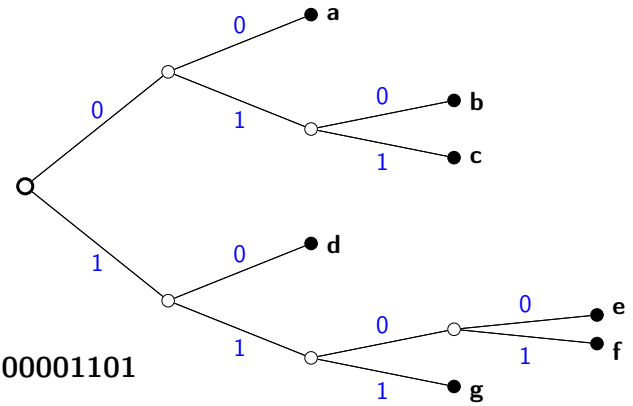
letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



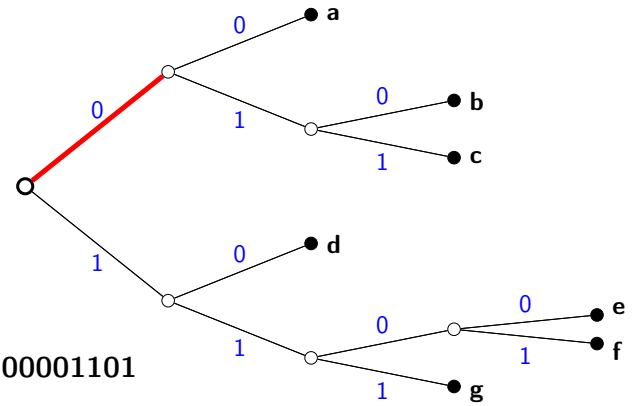
bitstream: **0101100001101**

symbols:

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



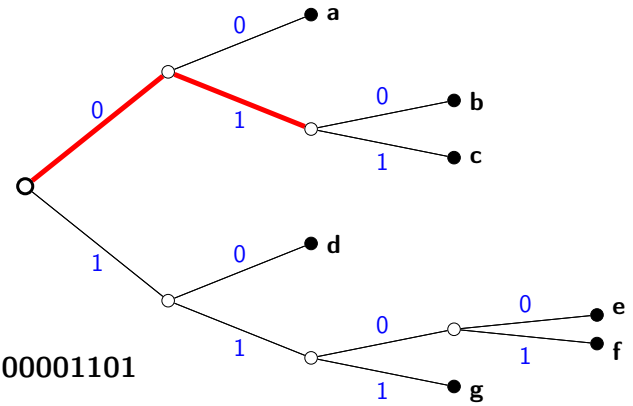
bitstream: 0101100001101

symbols:

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



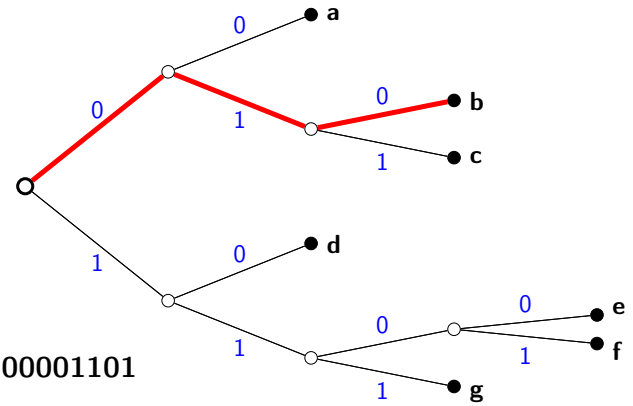
bitstream: 0101100001101

symbols:

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



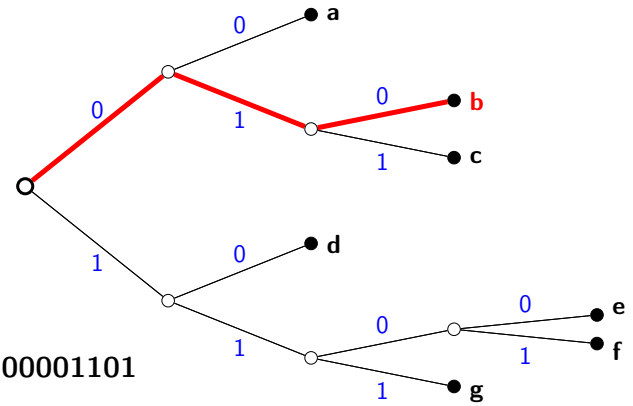
bitstream: 0101100001101

symbols:

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



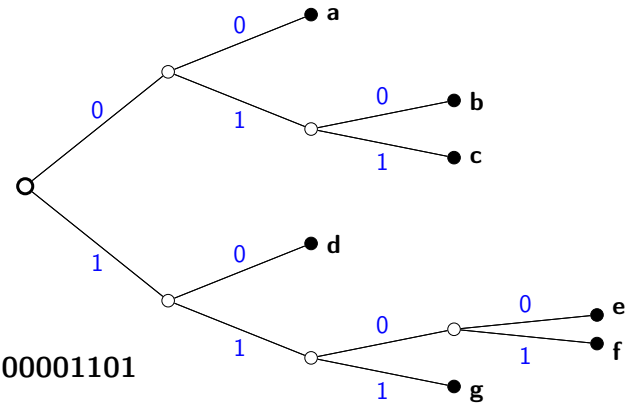
bitstream: **010**1100001101

symbols: **b**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



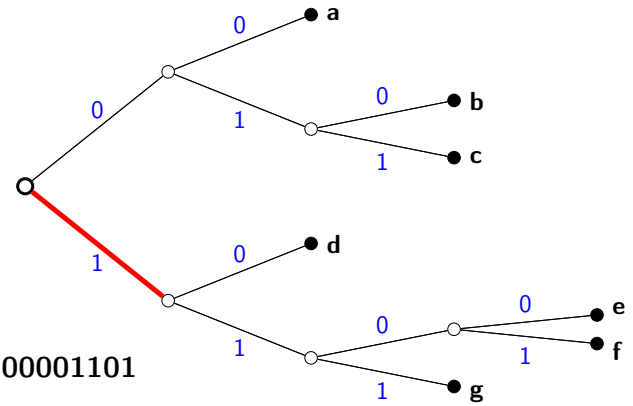
bitstream: 010**1100001101**

symbols: **b**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



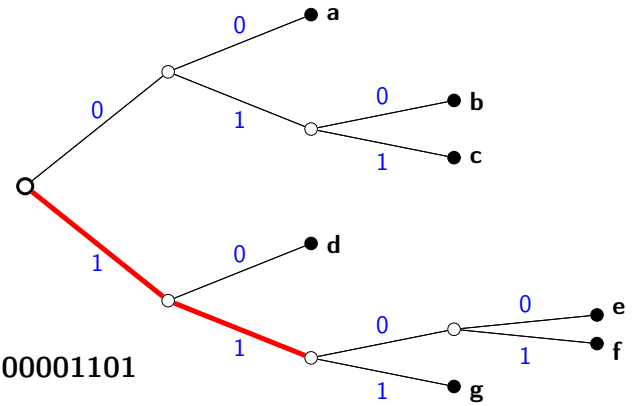
bitstream: 0101100001101

symbols: **b**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



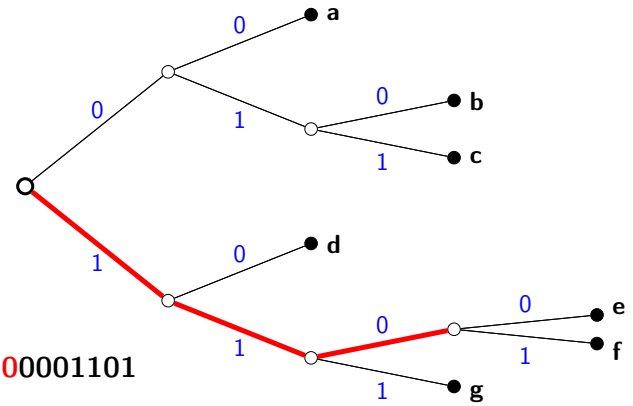
bitstream: 010**11**00001101

symbols: **b**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



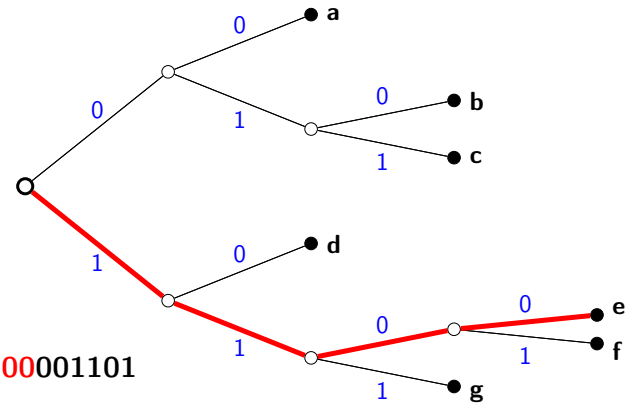
bitstream: 010**1100**0001101

symbols: **b**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



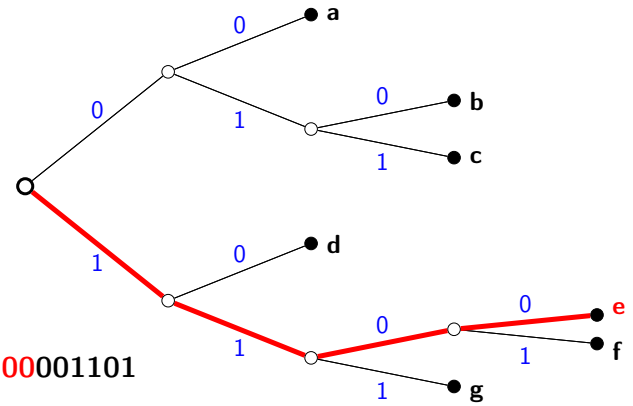
bitstream: 010**1100**001101

symbols: **b**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



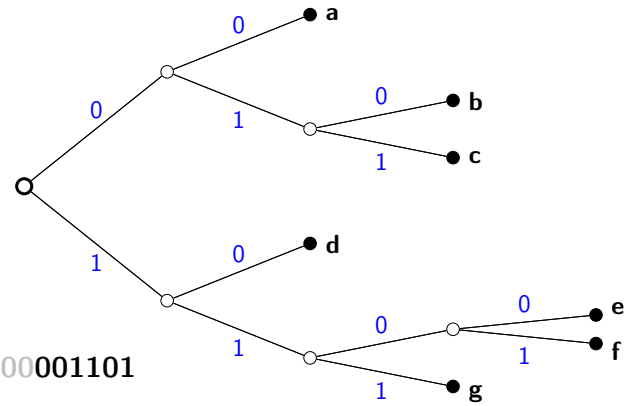
bitstream: 0101100001101

symbols: be

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



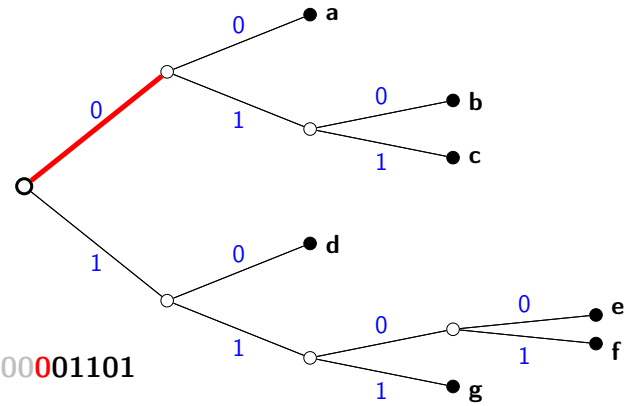
bitstream: 0101100**001101**

symbols: **be**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



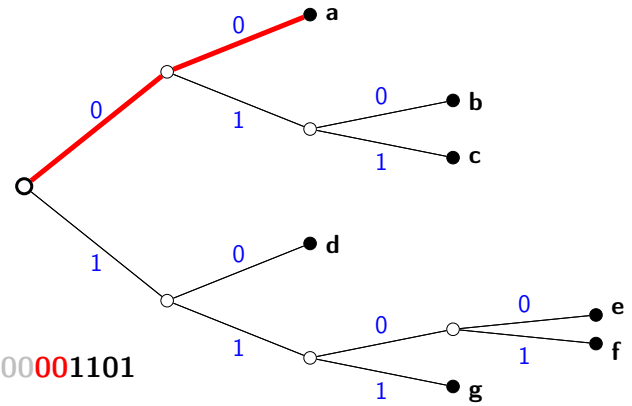
bitstream: 01011000**01101**

symbols: **be**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



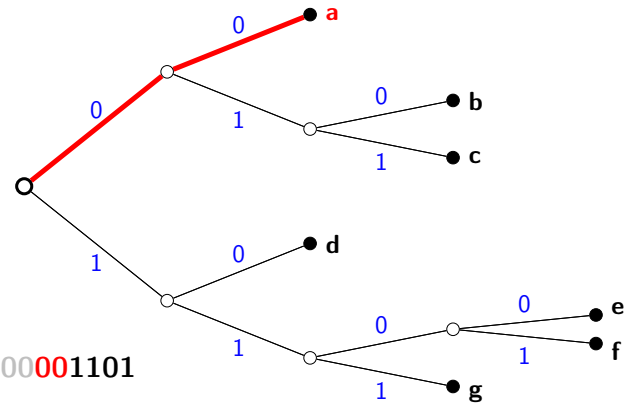
bitstream: 0101100001101

symbols: be

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



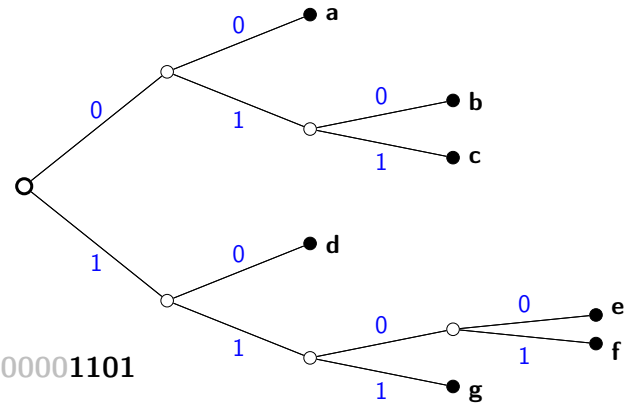
bitstream: 0101100001101

symbols: bea

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



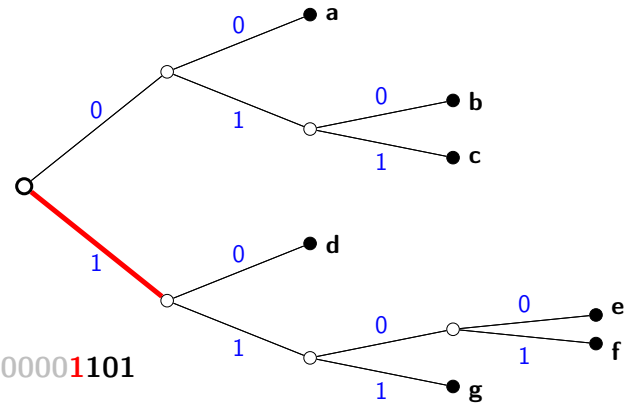
bitstream: 010110000**1101**

symbols: **bea**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



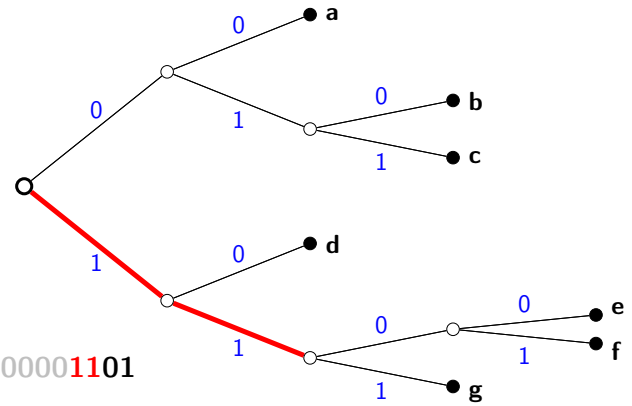
bitstream: 010110000**1101**

symbols: **bea**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



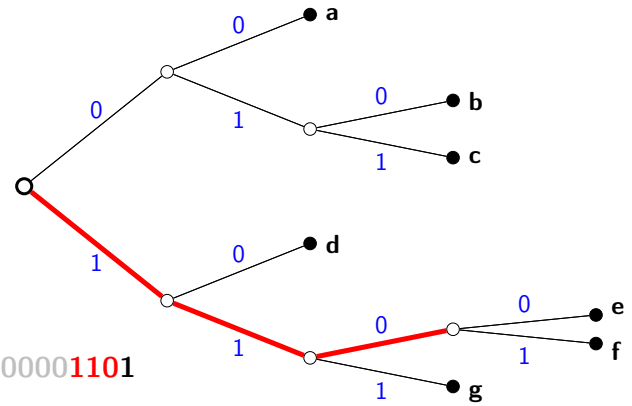
bitstream: 010110000**1101**

symbols: **bea**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



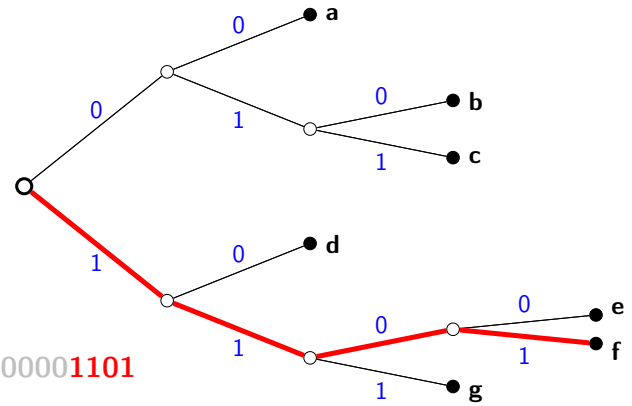
bitstream: 010110000**1101**

symbols: **bea**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



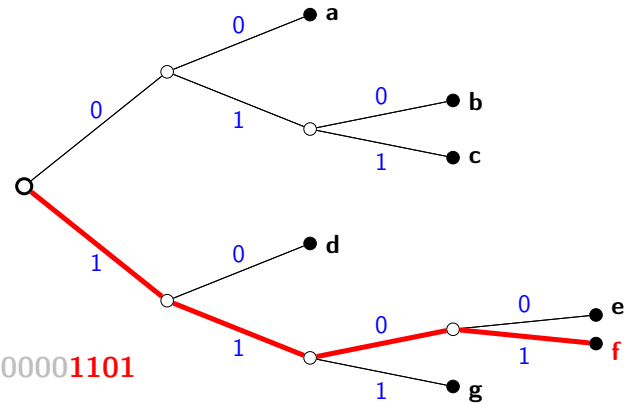
bitstream: 010110000**1101**

symbols: **bea**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



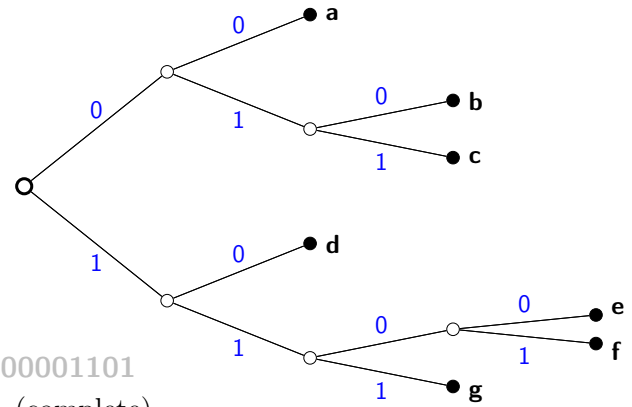
bitstream: 010110000**1101**

symbols: **beaf**

Example: Parsing for Prefix Codes

➡ Read bit by bit and follow code tree from root to terminal node

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



bitstream: 0101100001101

symbols: **beaf** (complete)

Instantaneous Decodability

Encoding of Prefix Codes

- Concatenate codewords for individual symbols of a message

➡ Valid for all scalar variable length codes

Decoding of Prefix Codes

- Represent prefix code as binary tree
- Read bit by bit and follow tree from root to terminal node

Instantaneous Decodability

Encoding of Prefix Codes

- Concatenate codewords for individual symbols of a message

➡ Valid for all scalar variable length codes

Decoding of Prefix Codes

- Represent prefix code as binary tree
- Read bit by bit and follow tree from root to terminal node

Important Property of Prefix Codes

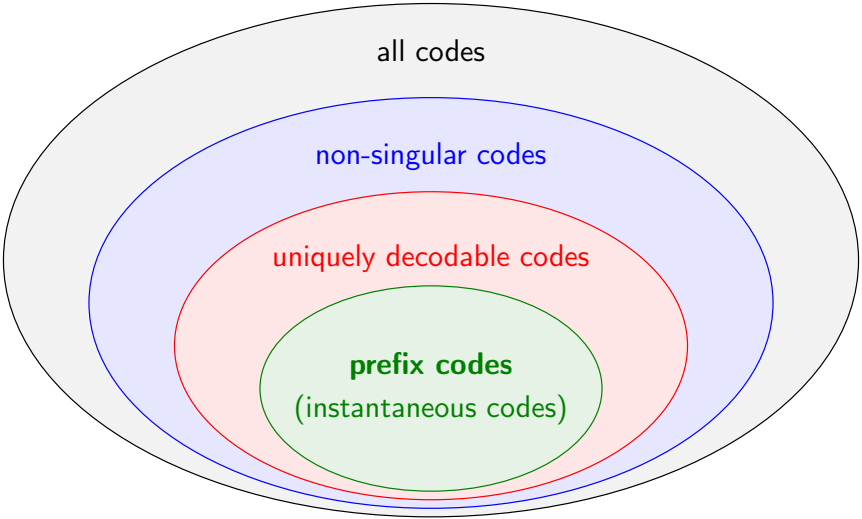
- Not only uniquely decodable, but also **instantaneously decodable**

➡ Can output each symbol as soon as the last bit of its codeword is read

➡ Enables switching between different codeword tables

➡ Straightforward use in complicated syntax

Classification of Codes



Intermediate Results

Prefix Codes

- Uniquely decodable codes
- ➡ Simple encoding and decoding algorithms
- ➡ Instantaneously decodable

Intermediate Results

Prefix Codes

- Uniquely decodable codes
- ➡ Simple encoding and decoding algorithms
- ➡ Instantaneously decodable

Open Questions

Intermediate Results

Prefix Codes

- Uniquely decodable codes
- ➡ Simple encoding and decoding algorithms
- ➡ Instantaneously decodable

Open Questions

1. Are there any other uniquely decodable codes that can achieve a smaller average codeword length than the best prefix code?

Intermediate Results

Prefix Codes

- Uniquely decodable codes
- ➡ Simple encoding and decoding algorithms
- ➡ Instantaneously decodable

Open Questions

1. Are there any other uniquely decodable codes that can achieve a smaller average codeword length than the best prefix code?
2. What is the **minimum average codeword length** for a given source?

Intermediate Results

Prefix Codes

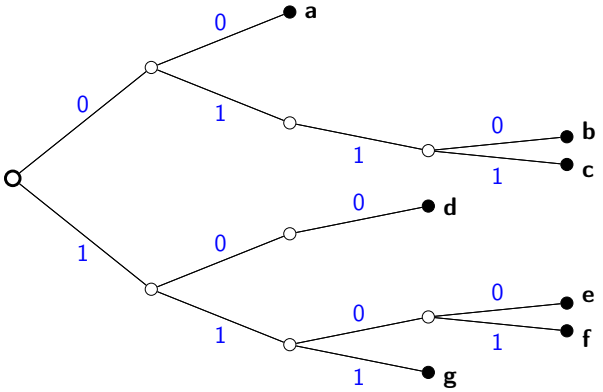
- Uniquely decodable codes
- ➡ Simple encoding and decoding algorithms
- ➡ Instantaneously decodable

Open Questions

1. Are there any other uniquely decodable codes that can achieve a smaller average codeword length than the best prefix code?
2. What is the **minimum average codeword length** for a given source?
3. How can we develop an **optimal code** for a source with given pmf?

Prefix Codes with Structural Redundancy

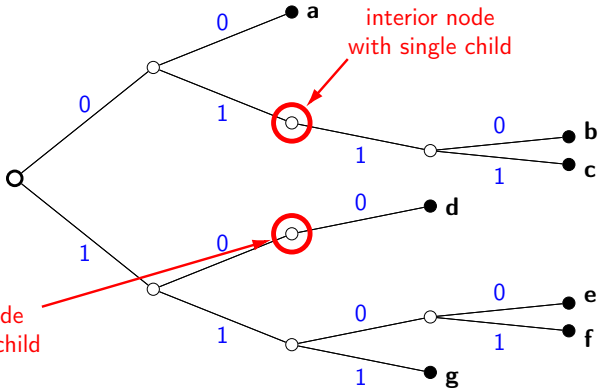
letter	codeword
a	00
b	0110
c	0111
d	100
e	1100
f	1101
g	111



Binary code tree is **not a full binary tree** (also: improper binary tree)

Prefix Codes with Structural Redundancy

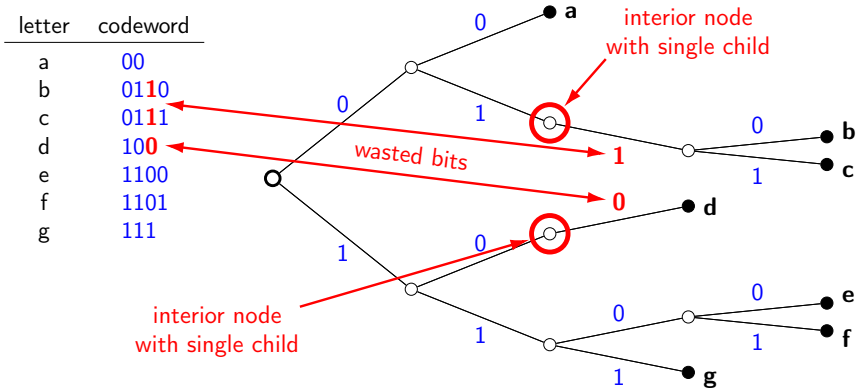
letter	codeword
a	00
b	0110
c	0111
d	100
e	1100
f	1101
g	111



Binary code tree is **not a full binary tree** (also: improper binary tree)

- There are interior nodes with only one child

Prefix Codes with Structural Redundancy

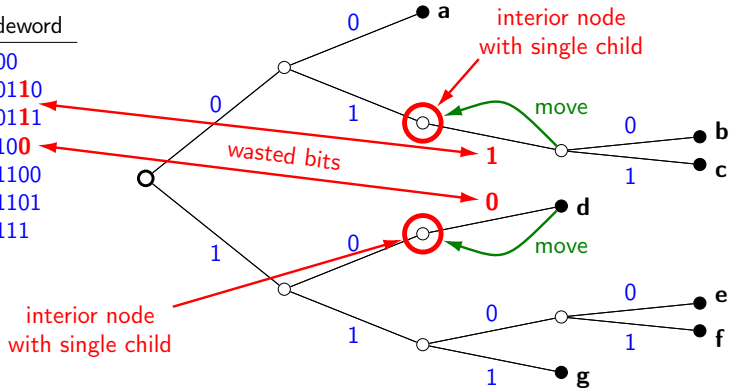


Binary code tree is **not a full binary tree** (also: improper binary tree)

- There are interior nodes with only one child
- ➡ Results in wasted bit (for one or more codewords)

Prefix Codes with Structural Redundancy

letter	codeword
a	00
b	0110
c	0111
d	100
e	1100
f	1101
g	111

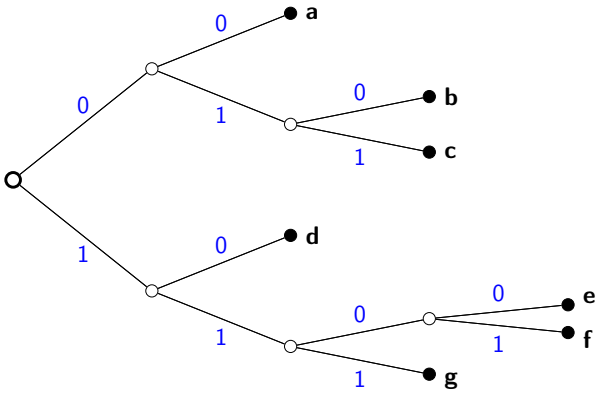


Binary code tree is **not a full binary tree** (also: improper binary tree)

- There are interior nodes with only one child
- ➡ Results in wasted bit (for one or more codewords)
- ➡ Average codeword length can be decreased by **moving single child node(s)**

Prefix Codes without Structural Redundancy

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



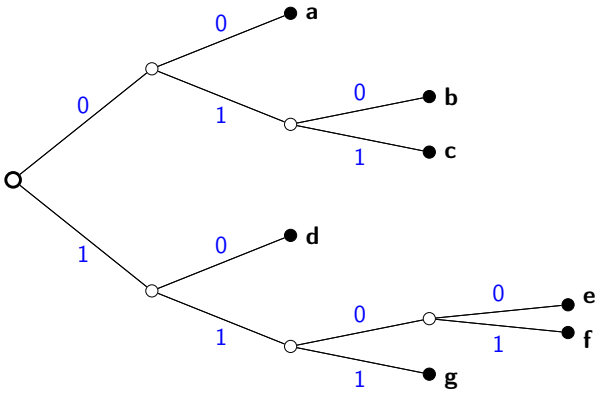
Binary code tree is a **full binary tree** (also: proper binary tree)

- All nodes have either no or two childs

➡ All bits in codewords are required

Prefix Codes without Structural Redundancy

letter	codeword
a	00
b	010
c	011
d	10
e	1100
f	1101
g	111



Binary code tree is a **full binary tree** (also: proper binary tree)

- All nodes have either no or two childs

➡ All bits in codewords are required

➡ But: The code may still be inefficient for a given source

Measure for Structural Redundancy of Prefix Codes

- Consider measure: $\zeta = \sum_{\forall k} 2^{-\ell_k}$

Measure for Structural Redundancy of Prefix Codes

- Consider measure: $$\zeta = \sum_{\forall k} 2^{-\ell_k}$$

Analysis of this measure ζ:

➡ Only root node

$$\zeta_{\text{root}} \;=\; 2^0 = 1$$

Measure for Structural Redundancy of Prefix Codes

- Consider measure:

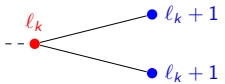
$$\zeta = \sum_{\forall k} 2^{-\ell_k}$$

Analysis of this measure ζ:

➡ Only root node

$$\zeta_{\text{root}} = 2^0 = 1$$

➡ Adding two childs at node with ℓ_k



$$\begin{aligned} \zeta_{\text{new}} &= \zeta_{\text{old}} - 2^{-\ell_k} + 2 \cdot 2^{-(\ell_k+1)} \\ &= \zeta_{\text{old}} \end{aligned}$$

Measure for Structural Redundancy of Prefix Codes

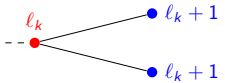
- Consider measure: $\zeta = \sum_{\forall k} 2^{-\ell_k}$

Analysis of this measure ζ:

➡ Only root node

$$\zeta_{\text{root}} = 2^0 = 1$$

➡ Adding two childs at node with ℓ_k



$$\begin{aligned} \zeta_{\text{new}} &= \zeta_{\text{old}} - 2^{-\ell_k} + 2 \cdot 2^{-(\ell_k+1)} \\ &= \zeta_{\text{old}} \end{aligned}$$

➡ Adding one child at node with ℓ_k

$$\begin{aligned} \zeta_{\text{new}} &= \zeta_{\text{old}} - 2^{-\ell_k} + 2^{-(\ell_k+1)} \\ &< \zeta_{\text{old}} \end{aligned}$$

Kraft Inequality for Prefix Codes

Kraft Inequality

- Prefix codes γ always have

$$\zeta(\gamma) = \sum_{\forall k} 2^{-\ell_k} \leq 1$$

Kraft Inequality for Prefix Codes

Kraft Inequality

- Prefix codes γ always have

$$\zeta(\gamma) = \sum_{\forall k} 2^{-\ell_k} \leq 1$$

➡ Prefix codes without structural redundancy (full binary code tree)

$$\zeta(\gamma) = \sum_{\forall k} 2^{-\ell_k} = 1$$

➡ Prefix codes with structural redundancy (not a full binary code tree)

$$\zeta(\gamma) = \sum_{\forall k} 2^{-\ell_k} < 1$$

Construction Of Prefix Codes For Given Codeword Lengths

Given: Ordered set of N codeword lengths $\{\ell_0, \ell_1, \ell_2, \cdots, \ell_{N-1}\}$, with $\ell_0 \leq \ell_1 \leq \ell_2, \leq \cdots \leq \ell_{N-1}$, that satisfies the Kraft inequality

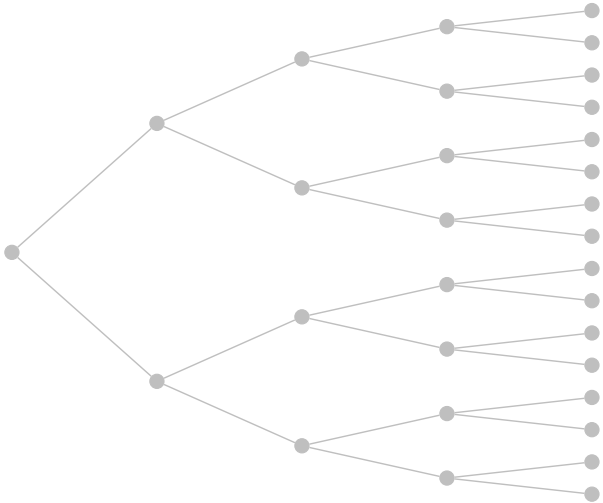
$$\sum_{\forall k} 2^{-\ell_k} \leq 1$$

Prefix Code Construction

1. Start with balanced tree of maximum depth
2. Init codeword length index $k = 0$
3. Choose any node of depth ℓ_k and prune tree at this node
4. Increment codeword length index $k = k + 1$
5. If $k < N$, proceed with 3

Prefix Code Construction Example

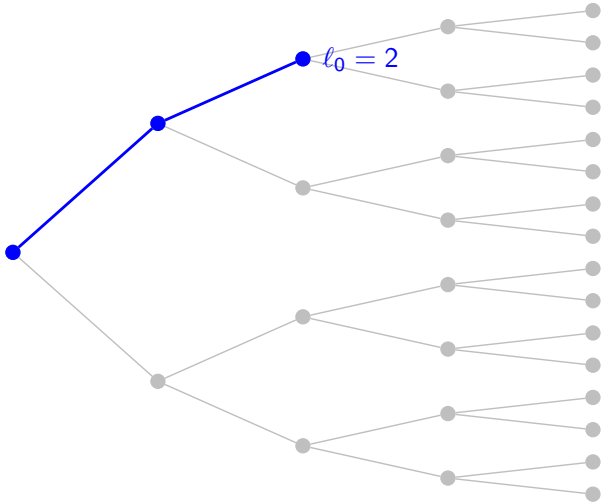
k	ℓ_k
0	2
1	2
2	3
3	3
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

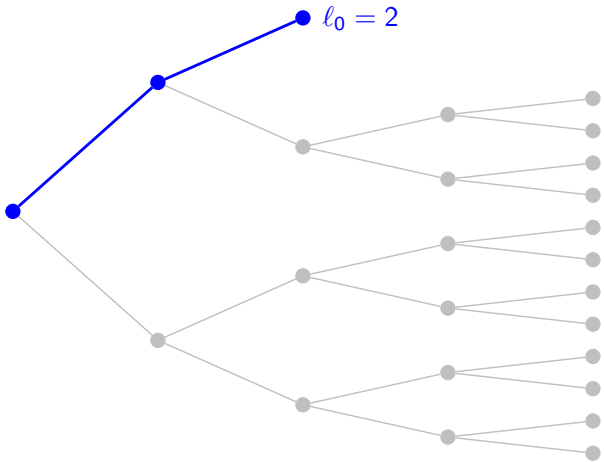
k	ℓ_k
0	**2**
1	2
2	3
3	3
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

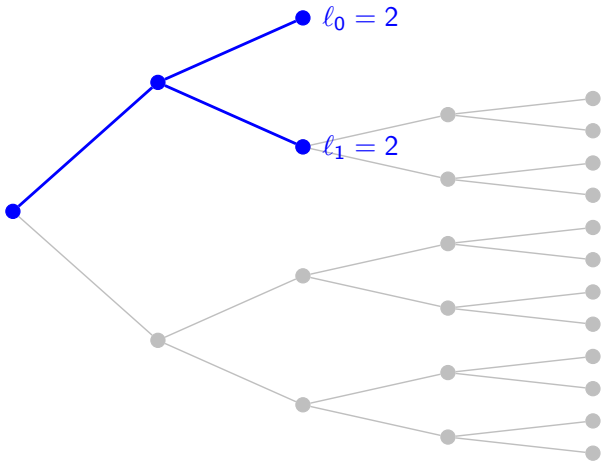
k	ℓ_k
0	**2**
1	2
2	3
3	3
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

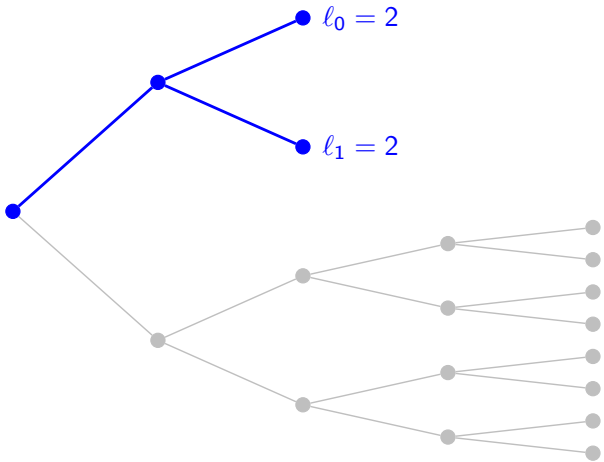
k	ℓ_k
0	**2**
1	**2**
2	3
3	3
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

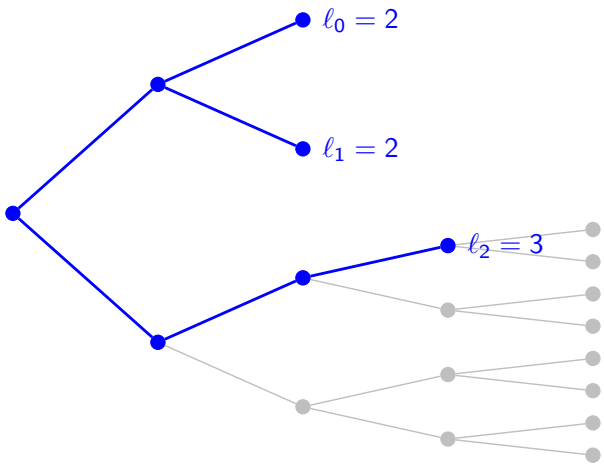
k	ℓ_k
0	**2**
1	**2**
2	3
3	3
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

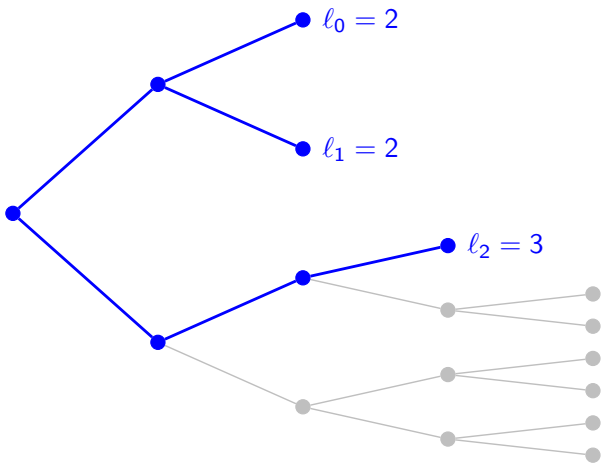
k	ℓ_k
0	**2**
1	**2**
2	**3**
3	3
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

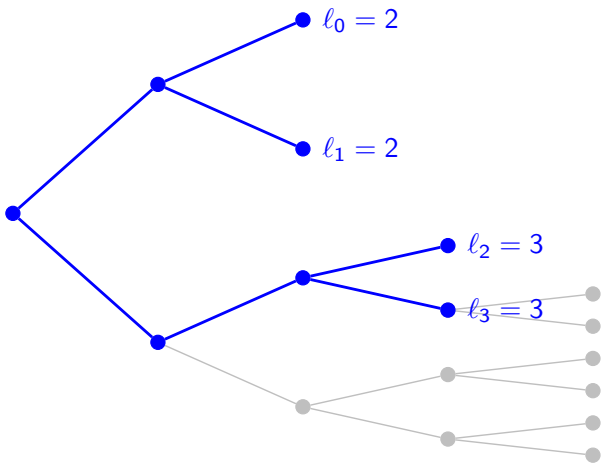
k	ℓ_k
0	**2**
1	**2**
2	**3**
3	3
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

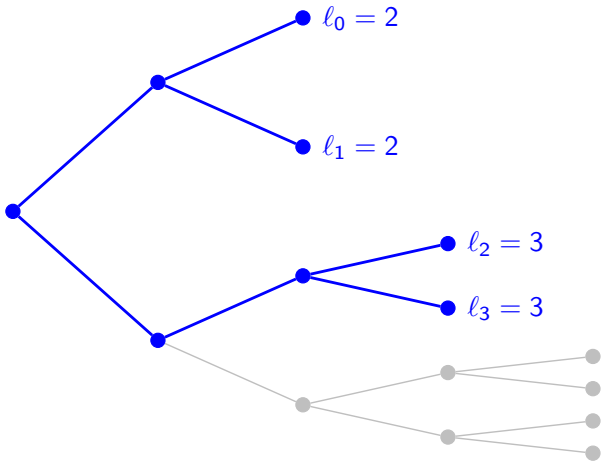
k	ℓ_k
0	**2**
1	**2**
2	**3**
3	**3**
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

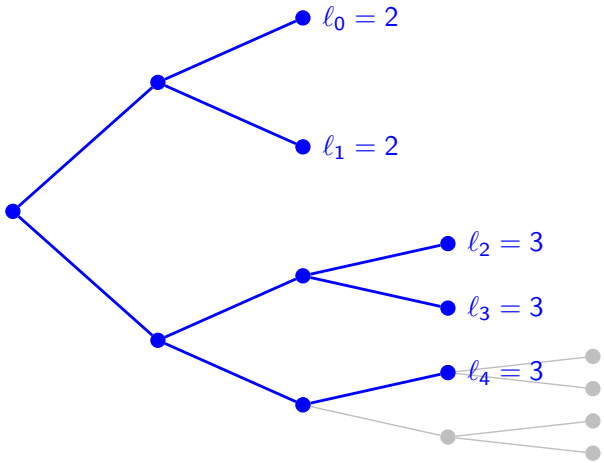
k	ℓ_k
0	**2**
1	**2**
2	**3**
3	**3**
4	3
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

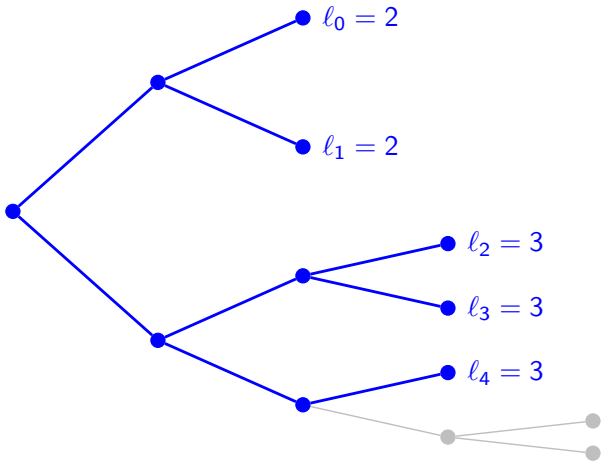
k	ℓ_k
0	**2**
1	**2**
2	**3**
3	**3**
4	**3**
5	4
6	4



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

k	ℓ_k
0	**2**
1	**2**
2	**3**
3	**3**
4	**3**
5	4
6	4

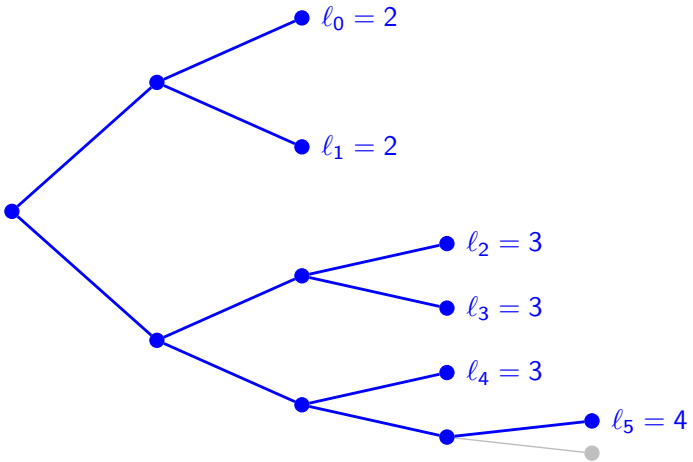


$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Prefix Code Construction Example

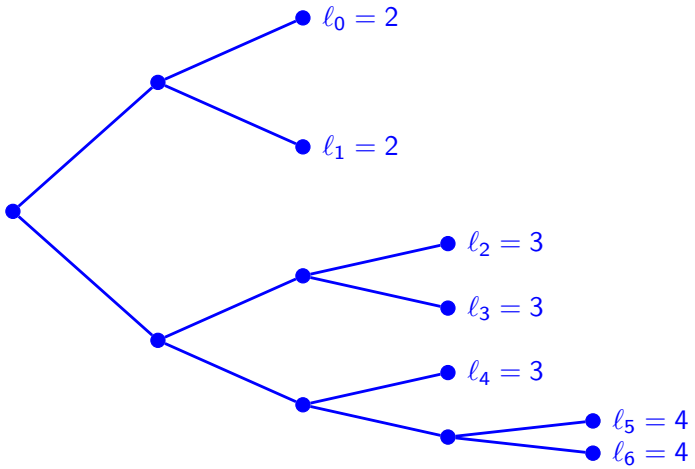
k	ℓ_k
0	**2**
1	**2**
2	**3**
3	**3**
4	**3**
5	**4**
6	4

$$\sum_{\forall k} 2^{-\ell_k} = 1$$



Prefix Code Construction Example

k	ℓ_k
0	**2**
1	**2**
2	**3**
3	**3**
4	**3**
5	**4**
6	**4**



$$\sum_{\forall k} 2^{-\ell_k} = 1$$

Is This Code Construction Always Possible ?

- Observation: Selection of a node at depth ℓ_k removes $2^{\ell_i - \ell_k}$ choices at depth $\ell_i \geq \ell_k$

➡ Remaining choices $n(\ell_i)$ at depth $\ell_i \geq \ell_k$ are given by

$$\begin{aligned} n(\ell_i) &= 2^{\ell_i} - \sum_{\forall k<i} 2^{\ell_i - \ell_k} = 2^{\ell_i} \cdot 1 - \sum_{\forall k<i} 2^{\ell_i - \ell_k} \\ &\geq 2^{\ell_i} \left(\sum_{\forall k} 2^{-\ell_k} \right) - \sum_{\forall k<i} 2^{\ell_i - \ell_k} = \sum_{\forall k \geq i} 2^{\ell_i - \ell_k} \\ &= 2^{\ell_i - \ell_i} + \sum_{\forall k>i} 2^{\ell_i - \ell_k} = 1 + \sum_{\forall k>i} 2^{\ell_i - \ell_k} \\ &\geq 1 \end{aligned}$$

$\sum_{\forall k} 2^{-\ell_k} \leq 1$:

➡ **For each set of codeword lengths $\{\ell_k\}$ that satisfies the Kraft inequality, we can always construct prefix code**

Kraft-McMillan Inequality

Kraft-McMillan: Necessary Condition for Unique Decodability

For each uniquely decodable code, the set of codeword lengths $\{\ell_k\}$ must fulfill

$$\sum_{\forall k} 2^{-\ell_k} \leq 1$$

- Already shown for prefix codes
- Must also be satisfied for all uniquely decodable codes (proof on next slide)

Proof of Kraft-McMillan Inequality

$$\left(\sum_{\forall x} 2^{-\ell(x)}\right)^N = \sum_{\forall x_0}\sum_{\forall x_1}\cdots\sum_{\forall x_{N-1}} 2^{-\ell(x_0)}\cdot 2^{-\ell(x_1)}\cdot\ \ldots\ \cdot 2^{-\ell(x_{N-1})} = \sum_{\forall \boldsymbol{x}^N} 2^{-\ell(\boldsymbol{x}^N)}$$

$$= \sum_{\ell^N=1}^{N\cdot\ell_{\max}} K\left(\ell^N\right)\cdot 2^{-\ell^N}$$

$$\leq \sum_{\ell^N=1}^{N\cdot\ell_{\max}} 2^{\ell^N}\cdot 2^{-\ell^N}$$

$$= \sum_{\ell^N=1}^{N\cdot\ell_{\max}} 1 = N\cdot\ell_{\max}$$

$$\sum_{\forall x\in\mathcal{A}} 2^{-\ell(x)} \leq \sqrt[N]{N\cdot\ell_{\max}}$$

$N\to\infty$:

$$\sum_{\forall x\in\mathcal{A}} 2^{-\ell(x)} \leq \lim_{N\to\infty}\sqrt[N]{N\cdot\ell_{\max}} = 1$$

N	:	number of symbols in a message
$\ell_{\max}$	:	maximum codewode length per symbol
$\boldsymbol{x}^N$	:	message of N symbols
ℓ^N	:	combined codeword length for N symbols
$K(\ell^N)$	:	number of combined codewords with combined length ℓ^N

(1) there are only 2^ℓ distinct bit sequences of length ℓ

$$K(\ell^N) \leq 2^{\ell^N}$$

(2) we require unique decodability for arbitrary long messages

$$N\to\infty$$

Practical Importance of Prefix Codes

We have shown:

1. All uniquely decodable codes fulfill the Kraft-McMillan inequality
2. For each set of codeword lengths that fulfills the Kraft-McMillan inequality, we can construct a prefix code

➡ **There are no uniquely decodable codes that have a smaller average codeword length than the best prefix code**

Practical Importance of Prefix Codes

We have shown:

1. All uniquely decodable codes fulfill the Kraft-McMillan inequality
2. For each set of codeword lengths that fulfills the Kraft-McMillan inequality, we can construct a prefix code

➞ **There are no uniquely decodable codes that have a smaller average codeword length than the best prefix code**

Prefix Codes

- Simple decoding algorithm
- Not only uniquely decodable, but also **instantaneously decodable**

➞ **All variable-length codes used in practice are prefix codes**

Divergence Inequality

Kullback-Leibler Divergence (for pmfs)

- Measure for divergence from a pmf q to a pmf p

$$D(p \,||\, q) = \sum_{\forall k} p_k \, \log_2\left(\frac{p_k}{q_k}\right)$$

Note: In general we have $D(p \,||\, q) \neq D(q \,||\, p)$

Divergence Inequality

Kullback-Leibler Divergence (for pmfs)

- Measure for divergence from a pmf q to a pmf p

$$D(p \,||\, q) = \sum_{\forall k} p_k \, \log_2 \left(\frac{p_k}{q_k} \right)$$

Note: In general we have $D(p \,||\, q) \neq D(q \,||\, p)$

Divergence Inequality

- Divergence is non-negative:

$$D(p \,||\, q) \geq 0$$

with **equality if and only if** $p = q$ (i.e., $\forall k,\ p_k = q_k$)

Proof of Divergence Inequality

Use inequality $\ln x \leq x - 1$ (with equality if and only if $x = 1$)

$$D(p \,||\, q) = \sum_{\forall k} p_k \log_2 \left(\frac{p_k}{q_k} \right)$$

Proof of Divergence Inequality

Use inequality $\ln x \leq x - 1$ (with equality if and only if $x = 1$)

$$D(p \,||\, q) = \sum_{\forall k} p_k \log_2\left(\frac{p_k}{q_k}\right) \qquad \left(\text{use: } \log_2 x = \frac{\ln x}{\ln 2} = -\frac{1}{\ln 2} \ln \frac{1}{x}\right)$$

Proof of Divergence Inequality

Use inequality $\ln x \leq x - 1$ (with equality if and only if $x = 1$)

$$D(p \,||\, q) = \sum_{\forall k} p_k \log_2\left(\frac{p_k}{q_k}\right) \qquad \left(\text{use: } \log_2 x = \frac{\ln x}{\ln 2} = -\frac{1}{\ln 2} \ln \frac{1}{x}\right)$$

$$= -\frac{1}{\ln 2} \sum_{\forall k} p_k \ln\left(\frac{q_k}{p_k}\right)$$

Proof of Divergence Inequality

Use inequality $\ln x \leq x - 1$ (with equality if and only if $x = 1$)

$$D(p \,||\, q) = \sum_{\forall k} p_k \log_2\left(\frac{p_k}{q_k}\right) \qquad \left(\text{use: } \log_2 x = \frac{\ln x}{\ln 2} = -\frac{1}{\ln 2} \ln \frac{1}{x}\right)$$

$$= -\frac{1}{\ln 2} \sum_{\forall k} p_k \ln\left(\frac{q_k}{p_k}\right) \qquad (\text{ apply: } -\ln x \geq 1 - x\ )$$

Proof of Divergence Inequality

Use inequality $\ln x \leq x - 1$ (with equality if and only if $x = 1$)

$$D(p \,||\, q) = \sum_{\forall k} p_k \log_2\left(\frac{p_k}{q_k}\right) \qquad \left(\text{use: } \log_2 x = \frac{\ln x}{\ln 2} = -\frac{1}{\ln 2} \ln \frac{1}{x}\right)$$

$$= -\frac{1}{\ln 2} \sum_{\forall k} p_k \ln\left(\frac{q_k}{p_k}\right) \qquad (\text{ apply: } -\ln x \geq 1 - x\ )$$

$$\geq \frac{1}{\ln 2} \sum_{\forall k} p_k \left(1 - \frac{q_k}{p_k}\right) \qquad (\text{ equality: } \forall k, p_k = q_k\ )$$

Proof of Divergence Inequality

Use inequality $\ln x \leq x - 1$ (with equality if and only if $x = 1$)

$$
\begin{aligned}
D(p \,||\, q) &= \sum_{\forall k} p_k \, \log_2\left(\frac{p_k}{q_k}\right) && \left(\text{use: } \log_2 x = \frac{\ln x}{\ln 2} = -\frac{1}{\ln 2} \, \ln \frac{1}{x}\right) \\
&= -\frac{1}{\ln 2} \sum_{\forall k} p_k \, \ln\left(\frac{q_k}{p_k}\right) && (\text{ apply: } -\ln x \geq 1 - x \,) \\
&\geq \frac{1}{\ln 2} \sum_{\forall k} p_k \left(1 - \frac{q_k}{p_k}\right) && (\text{ equality: } \forall k, p_k = q_k \,) \\
&= \frac{1}{\ln 2} \left(\sum_{\forall k} p_k - \sum_{\forall k} q_k\right)
\end{aligned}
$$

Proof of Divergence Inequality

Use inequality $\ln x \leq x - 1$ (with equality if and only if $x = 1$)

$$D(p \,||\, q) = \sum_{\forall k} p_k \log_2\left(\frac{p_k}{q_k}\right) \qquad \left(\text{use: } \log_2 x = \frac{\ln x}{\ln 2} = -\frac{1}{\ln 2} \ln \frac{1}{x}\right)$$

$$= -\frac{1}{\ln 2} \sum_{\forall k} p_k \ln\left(\frac{q_k}{p_k}\right) \qquad (\text{ apply: } -\ln x \geq 1 - x\ )$$

$$\geq \frac{1}{\ln 2} \sum_{\forall k} p_k \left(1 - \frac{q_k}{p_k}\right) \qquad (\text{ equality: } \forall k, p_k = q_k\ )$$

$$= \frac{1}{\ln 2} \left(\sum_{\forall k} p_k - \sum_{\forall k} q_k\right) = \frac{1}{\ln 2}(1 - 1) = 0$$

Proof of Divergence Inequality

Use inequality $\ln x \leq x - 1$ (with equality if and only if $x = 1$)

$$
\begin{aligned}
D(p\,||\,q) &= \sum_{\forall k} p_k \log_2\left(\frac{p_k}{q_k}\right) && \left(\text{use: } \log_2 x = \frac{\ln x}{\ln 2} = -\frac{1}{\ln 2}\ln\frac{1}{x}\right) \\
&= -\frac{1}{\ln 2}\sum_{\forall k} p_k \ln\left(\frac{q_k}{p_k}\right) && (\text{ apply: } -\ln x \geq 1 - x\ ) \\
&\geq \frac{1}{\ln 2}\sum_{\forall k} p_k \left(1 - \frac{q_k}{p_k}\right) && (\text{ equality: } \forall k, p_k = q_k\ ) \\
&= \frac{1}{\ln 2}\left(\sum_{\forall k} p_k - \sum_{\forall k} q_k\right) = \frac{1}{\ln 2}(1-1) = 0
\end{aligned}
$$

➡ $D(p\,||\,q) \geq 0$ (equality: $p = q$)

Lower Bound for Average Codeword Length

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k$$

Lower Bound for Average Codeword Length

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;\; = \;\; \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) - \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

Lower Bound for Average Codeword Length

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;=\; \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) - \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

[Kraft-McMillan inequality]

Lower Bound for Average Codeword Length

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;=\; \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) - \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

[Kraft-McMillan inequality]

$$\geq \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

Lower Bound for Average Codeword Length

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;=\; \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) - \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

[Kraft-McMillan inequality]

$$\geq \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

$$= \left(\sum_{\forall k} p_k \, \ell_k \right) + \left(\sum_{\forall k} p_k \right) \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

Lower Bound for Average Codeword Length

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k = \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) - \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

[Kraft-McMillan inequality]

$$\geq \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

$$= \left(\sum_{\forall k} p_k \, \ell_k \right) + \left(\sum_{\forall k} p_k \right) \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

$$= \sum_{\forall k} p_k \left(\ell_k + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) \right)$$

Lower Bound for Average Codeword Length

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k = \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) - \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

[Kraft-McMillan inequality]

$$\geq \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

$$= \left(\sum_{\forall k} p_k \, \ell_k \right) + \left(\sum_{\forall k} p_k \right) \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right)$$

$$= \sum_{\forall k} p_k \left(\ell_k + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) \right)$$

$$= \sum_{\forall k} p_k \left(-\log_2 \left(2^{-\ell_k} \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) \right)$$

Lower Bound for Average Codeword Length

$$\begin{aligned}
\bar{\ell} = \sum_{\forall k} p_k \, \ell_k &= \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) - \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) \\
\text{[Kraft-McMillan inequality]} \qquad &\geq \left(\sum_{\forall k} p_k \, \ell_k \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) \\
&= \left(\sum_{\forall k} p_k \, \ell_k \right) + \left(\sum_{\forall k} p_k \right) \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) \\
&= \sum_{\forall k} p_k \left(\ell_k + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) \right) \\
&= \sum_{\forall k} p_k \left(-\log_2 \left(2^{-\ell_k} \right) + \log_2 \left(\sum_{\forall i} 2^{-\ell_i} \right) \right) \\
&= -\sum_{\forall k} p_k \, \log_2 \left(\frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \right)
\end{aligned}$$

Lower Bound for Average Codeword Length (continued)

Define new pmf q with probability masses

$$q_k = \frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \qquad \left(\text{note: } \; q_k \geq 0 \quad \text{and} \quad \sum_{\forall k} q_k = 1 \right)$$

Lower Bound for Average Codeword Length (continued)

Define new pmf q with probability masses

$$q_k = \frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \qquad \left(\text{note: } \; q_k \geq 0 \quad \text{and} \quad \sum_{\forall k} q_k = 1 \right)$$

➡ Continue derivation

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;\geq\; -\sum_{\forall k} p_k \, \log_2 \left(\frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \right)$$

Lower Bound for Average Codeword Length (continued)

Define new pmf q with probability masses

$$q_k = \frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \qquad \left(\text{note: } \; q_k \geq 0 \quad \text{and} \quad \sum_{\forall k} q_k = 1 \right)$$

➡ Continue derivation

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;\geq\; -\sum_{\forall k} p_k \, \log_2 \left(\frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \right) = -\sum_{\forall k} p_k \, \log_2 q_k$$

Lower Bound for Average Codeword Length (continued)

Define new pmf q with probability masses

$$q_k = \frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \qquad \left(\text{note: } \; q_k \geq 0 \quad \text{and} \quad \sum_{\forall k} q_k = 1 \right)$$

➡ Continue derivation

$$\begin{aligned} \bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;&\geq\; -\sum_{\forall k} p_k \, \log_2 \left(\frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \right) = -\sum_{\forall k} p_k \, \log_2 q_k \\ &=\; -\sum_{\forall k} p_k \left(\log_2 q_k + \log_2 p_k - \log_2 p_k \right) \end{aligned}$$

Lower Bound for Average Codeword Length (continued)

Define new pmf q with probability masses

$$q_k = \frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \qquad \left(\text{note: } \; q_k \geq 0 \quad \text{and} \quad \sum_{\forall k} q_k = 1 \right)$$

➡ Continue derivation

$$\begin{aligned} \bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;&\geq\; -\sum_{\forall k} p_k \, \log_2 \left(\frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \right) = -\sum_{\forall k} p_k \, \log_2 q_k \\ &=\; -\sum_{\forall k} p_k \left(\log_2 q_k + \log_2 p_k - \log_2 p_k \right) \\ &=\; -\sum_{\forall k} p_k \, \log_2 p_k + \sum_{\forall k} p_k \, \log_2 \left(\frac{p_k}{q_k} \right) \end{aligned}$$

Lower Bound for Average Codeword Length (continued)

Define new pmf q with probability masses

$$q_k = \frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \qquad \left(\text{note: } q_k \geq 0 \quad \text{and} \quad \sum_{\forall k} q_k = 1 \right)$$

➡ Continue derivation

$$\begin{aligned}
\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;&\geq\; -\sum_{\forall k} p_k \, \log_2 \left(\frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \right) = -\sum_{\forall k} p_k \, \log_2 q_k \\
&= -\sum_{\forall k} p_k \left(\log_2 q_k + \log_2 p_k - \log_2 p_k \right) \\
&= -\sum_{\forall k} p_k \, \log_2 p_k + \sum_{\forall k} p_k \, \log_2 \left(\frac{p_k}{q_k} \right) \\
&= -\sum_{\forall k} p_k \, \log_2 p_k + D(p \,||\, q)
\end{aligned}$$

Lower Bound for Average Codeword Length (continued)

Define new pmf q with probability masses

$$q_k = \frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \qquad \left(\text{note: } \; q_k \geq 0 \quad \text{and} \quad \sum_{\forall k} q_k = 1 \right)$$

➡ Continue derivation

$$\begin{aligned}
\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;&\geq\; -\sum_{\forall k} p_k \, \log_2 \left(\frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \right) = -\sum_{\forall k} p_k \, \log_2 q_k \\
&= \; -\sum_{\forall k} p_k \left(\log_2 q_k + \log_2 p_k - \log_2 p_k \right) \\
&= \; -\sum_{\forall k} p_k \, \log_2 p_k + \sum_{\forall k} p_k \, \log_2 \left(\frac{p_k}{q_k} \right) \\
&= \; -\sum_{\forall k} p_k \, \log_2 p_k + D(p \,||\, q)
\end{aligned}$$

[divergence inequality]

Lower Bound for Average Codeword Length (continued)

Define new pmf q with probability masses

$$q_k = \frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \qquad \left(\text{note: } q_k \geq 0 \quad \text{and} \quad \sum_{\forall k} q_k = 1 \right)$$

➡ Continue derivation

$$\begin{aligned}
\bar{\ell} = \sum_{\forall k} p_k \, \ell_k \;&\geq\; -\sum_{\forall k} p_k \, \log_2 \left(\frac{2^{-\ell_k}}{\sum_{\forall i} 2^{-\ell_i}} \right) = -\sum_{\forall k} p_k \, \log_2 q_k \\
&=\; -\sum_{\forall k} p_k \left(\log_2 q_k + \log_2 p_k - \log_2 p_k \right) \\
&=\; -\sum_{\forall k} p_k \, \log_2 p_k + \sum_{\forall k} p_k \, \log_2 \left(\frac{p_k}{q_k} \right) \\
&=\; -\sum_{\forall k} p_k \, \log_2 p_k + D(p \,||\, q) \\
[\text{divergence inequality}] \qquad &\geq\; -\sum_{\forall k} p_k \, \log_2 p_k
\end{aligned}$$

Entropy and Redundancy

Entropy of a Random Variable X with pmf p_X

$$H(X) = H(p_X) = \mathrm{E}\{ -\log_2 p_X(S) \} = -\sum_{\forall k} p_k \log_2 p_k$$

- Measure for uncertainty about a random variable X (with pmf p_X)
- **Lower bound for average codeword length** of scalar codes γ

$$\bar{\ell}(\gamma) = \sum_{\forall k} p_k \ell_k \geq H(p)$$

Entropy and Redundancy

Entropy of a Random Variable X with pmf p_X

$$H(X) = H(p_X) = \mathrm{E}\{ -\log_2 p_X(S) \} = -\sum_{\forall k} p_k \log_2 p_k$$

- Measure for uncertainty about a random variable X (with pmf p_X)
- **Lower bound for average codeword length** of scalar codes γ

$$\bar{\ell}(\gamma) = \sum_{\forall k} p_k \ell_k \geq H(p)$$

Redundancy: Measure for Efficiency of a Lossless Code γ

- Absolute redundancy $\varrho(\gamma)$ and relative redundancy $r(\gamma)$ of a lossless code γ

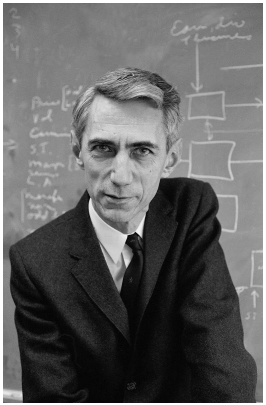
$$\varrho(\gamma) = \bar{\ell}(\gamma) - H(p) \geq 0$$

$$r(\gamma) = \frac{\varrho(\gamma)}{H(p)} = \frac{\bar{\ell}}{H(p)} - 1 \geq 0$$

Historical Reference

- SHANNON introduced entropy as an uncertainty measure for random experiments and derived it based on three postulates

➡ Founding work of the field of "Information Theory"



The Bell System Technical Journal

Vol. XXVII *July, 1948* *No. 3*

A Mathematical Theory of Communication

By C. E. SHANNON

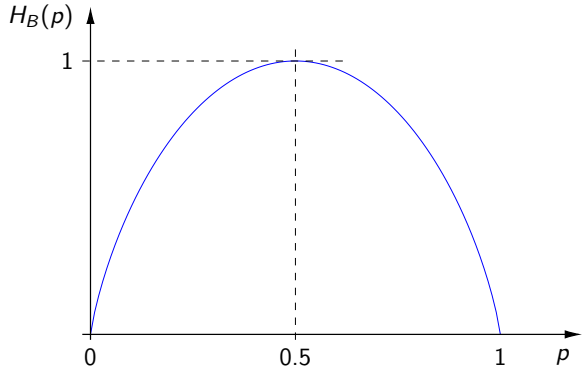
INTRODUCTION

THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist[1] and Hartley[2] on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message and due to the nature of the final destination of the information.

Example: Binary Entropy Function

Consider binary source X with probability mass function: $\{p, 1-p\}$

➡ Entropy of the source: $H(X) = H_B(p) = -p \log_2 p - (1-p) \log_2 (1-p)$



Prefix Codes with Zero Redundancy

We used two inequalities in the derivation of the entropy

1 Kraft-McMillan inequality

$$\sum_{\forall k} 2^{-\ell_k} \leq 1$$

- ➡ Equality if and only if prefix code represents a full binary tree (always possible)
- ➡ Resulting average codeword length: $\bar{\ell} = H(p) + D(p \,||\, q)$ with $q_k = 2^{-\ell_k}$

Prefix Codes with Zero Redundancy

We used two inequalities in the derivation of the entropy

1 Kraft-McMillan inequality

$$\sum_{\forall k} 2^{-\ell_k} \leq 1$$

- ➡ Equality if and only if prefix code represents a full binary tree (always possible)
- ➡ Resulting average codeword length: $\bar{\ell} = H(p) + D(p \,||\, q)$ with $q_k = 2^{-\ell_k}$

2 Divergence inequality

$$D(p \,||\, q) \geq 0 \qquad \text{(equality for } p_k = q_k, \forall k)$$

- ➡ Equality if and only if all codeword lengths are given by $\ell_k = -\log_2 p_k$

Prefix Codes with Zero Redundancy

We used two inequalities in the derivation of the entropy

1 Kraft-McMillan inequality

$$\sum_{\forall k} 2^{-\ell_k} \leq 1$$

- ➡ Equality if and only if prefix code represents a full binary tree (always possible)
- ➡ Resulting average codeword length: $\bar{\ell} = H(p) + D(p \,||\, q)$ with $q_k = 2^{-\ell_k}$

2 Divergence inequality

$$D(p \,||\, q) \geq 0 \qquad \text{(equality for } p_k = q_k, \forall k)$$

- ➡ Equality if and only if all codeword lengths are given by $\ell_k = -\log_2 p_k$

Zero redundancy codes are only possible if all probability masses represent negative integer powers of two

Upper Bound for Achievable Average Codeword Length

Shannon Code

- Set codeword lengths according to $\ell_k = \lceil -\log_2 p_k \rceil$
- Construct prefix code for these codeword length $\{\ell_k\}$

Upper Bound for Achievable Average Codeword Length

Shannon Code

- Set codeword lengths according to $\ell_k = \lceil -\log_2 p_k \rceil$
- Construct prefix code for these codeword length $\{\ell_k\}$

- Can we always construct a prefix code with these codewords lengths? (use $\lceil x \rceil \geq x$)

Yes: $$\sum_{\forall k} 2^{-\ell_k} = \sum_{\forall k} 2^{-\lceil -\log_2 p_k \rceil} \leq \sum_{\forall k} 2^{\log_2 p_k} = \sum_{\forall k} p_k = 1$$

Upper Bound for Achievable Average Codeword Length

Shannon Code

- Set codeword lengths according to $\ell_k = \lceil -\log_2 p_k \rceil$
- Construct prefix code for these codeword length $\{\ell_k\}$

- Can we always construct a prefix code with these codewords lengths? (use $\lceil x \rceil \geq x$)

$$\text{Yes:} \qquad \sum_{\forall k} 2^{-\ell_k} = \sum_{\forall k} 2^{-\lceil -\log_2 p_k \rceil} \leq \sum_{\forall k} 2^{\log_2 p_k} = \sum_{\forall k} p_k = 1$$

- Upper bound for average codeword length? (use $\lceil x \rceil < x + 1$)

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k = \sum_{\forall k} p_k \, \lceil -\log_2 p_k \rceil < \sum_{\forall k} p_k \, (1 - \log_2 p_k) = 1 + H(p)$$

Upper Bound for Achievable Average Codeword Length

Shannon Code

- Set codeword lengths according to $\ell_k = \lceil -\log_2 p_k \rceil$
- Construct prefix code for these codeword length $\{\ell_k\}$

- Can we always construct a prefix code with these codewords lengths? (use $\lceil x \rceil \geq x$)

$$\text{Yes:} \qquad \sum_{\forall k} 2^{-\ell_k} = \sum_{\forall k} 2^{-\lceil -\log_2 p_k \rceil} \leq \sum_{\forall k} 2^{\log_2 p_k} = \sum_{\forall k} p_k = 1$$

- Upper bound for average codeword length? (use $\lceil x \rceil < x + 1$)

$$\bar{\ell} = \sum_{\forall k} p_k \, \ell_k = \sum_{\forall k} p_k \, \lceil -\log_2 p_k \rceil < \sum_{\forall k} p_k \, (1 - \log_2 p_k) = 1 + H(p)$$

➡ **Can always find lossless code with**

$$H(p) \leq \bar{\ell} < H(p) + 1$$

Example of a Shannon Code

a_k	p_k
a	0.16
b	0.04
c	0.04
d	0.16
e	0.23
f	0.07
g	0.06
h	0.09
i	0.15

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$
a	0.16	2.6438...
b	0.04	4.6438...
c	0.04	4.6438...
d	0.16	2.6438...
e	0.23	2.1202...
f	0.07	3.8365...
g	0.06	4.0588...
h	0.09	3.4739...
i	0.15	2.7369...

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$
a	0.16	2.6438...	3
b	0.04	4.6438...	5
c	0.04	4.6438...	5
d	0.16	2.6438...	3
e	0.23	2.1202...	3
f	0.07	3.8365...	4
g	0.06	4.0588...	5
h	0.09	3.4739...	4
i	0.15	2.7369...	3

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	
c	0.04	4.6438...	5	
d	0.16	2.6438...	3	
e	0.23	2.1202...	3	
f	0.07	3.8365...	4	
g	0.06	4.0588...	5	
h	0.09	3.4739...	4	
i	0.15	2.7369...	3	

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	
c	0.04	4.6438...	5	
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	
f	0.07	3.8365...	4	
g	0.06	4.0588...	5	
h	0.09	3.4739...	4	
i	0.15	2.7369...	3	

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	
c	0.04	4.6438...	5	
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	
g	0.06	4.0588...	5	
h	0.09	3.4739...	4	
i	0.15	2.7369...	3	

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	
c	0.04	4.6438...	5	
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	
g	0.06	4.0588...	5	
h	0.09	3.4739...	4	
i	0.15	2.7369...	3	011

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	
c	0.04	4.6438...	5	
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	
h	0.09	3.4739...	4	
i	0.15	2.7369...	3	011

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	
c	0.04	4.6438...	5	
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	
h	0.09	3.4739...	4	1001
i	0.15	2.7369...	3	011

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	10100
c	0.04	4.6438...	5	
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	
h	0.09	3.4739...	4	1001
i	0.15	2.7369...	3	011

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	10100
c	0.04	4.6438...	5	10101
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	
h	0.09	3.4739...	4	1001
i	0.15	2.7369...	3	011

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	10100
c	0.04	4.6438...	5	10101
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	10110
h	0.09	3.4739...	4	1001
i	0.15	2.7369...	3	011

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	10100
c	0.04	4.6438...	5	10101
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	10110
h	0.09	3.4739...	4	1001
i	0.15	2.7369...	3	011

$$H(p) \approx 2.9405$$

$$\bar{\ell} = 3.44$$

$$\varrho(\bar{\ell}) \approx 0.4995 \ (17\%)$$

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	10100
c	0.04	4.6438...	5	10101
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	10110
h	0.09	3.4739...	4	1001
i	0.15	2.7369...	3	011

$$H(p) \approx 2.9405$$

$$\bar{\ell} = 3.44$$

$$\varrho(\bar{\ell}) \approx 0.4995 \ (17\%)$$

$$\sum_k 2^{-\ell_k} = \frac{23}{32} = 0.71875$$

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	10100
c	0.04	4.6438...	5	10101
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	10110
h	0.09	3.4739...	4	1001
i	0.15	2.7369...	3	011

$$H(p) \approx 2.9405$$

$$\bar{\ell} = 3.44$$

$$\varrho(\bar{\ell}) \approx 0.4995 \ (17\%)$$

$$\sum_k 2^{-\ell_k} = \frac{23}{32} = 0.71875$$

➜ **code is redundant / not optimal**

Example of a Shannon Code

a_k	p_k	$-\log_2 p_k$	$\ell_k = \lceil -\log_2 p_k \rceil$	codeword
a	0.16	2.6438...	3	000
b	0.04	4.6438...	5	10100
c	0.04	4.6438...	5	10101
d	0.16	2.6438...	3	001
e	0.23	2.1202...	3	010
f	0.07	3.8365...	4	1000
g	0.06	4.0588...	5	10110
h	0.09	3.4739...	4	1001
i	0.15	2.7369...	3	011

$$H(p) \approx 2.9405$$

$$\bar{\ell} = 3.44$$

$$\varrho(\bar{\ell}) \approx 0.4995 \ (17\%)$$

$$\sum_k 2^{-\ell_k} = \frac{23}{32} = 0.71875$$

➜ **code is redundant / not optimal**

Open Question

➜ **How can we construct an optimal prefix code?**

Summary of Lecture

Unique Decodability

- Necessary condition: Kraft-McMillan inequality for codeword lengths
- Sufficient condition: Prefix codes (i.e., prefix-free codes)

Prefix Codes

- Uniquely and instantaneously decodable
- Simple encoding and decoding algorithm (via binary tree representation)
- No better uniquely decodable codes than best prefix codes

Average Codeword Length and Entropy

- Characterization of efficiency of lossless codes: Average codeword length $\bar{\ell}$
- Entropy as lower bound for avg. codeword length: $\bar{\ell} \geq H(p)$
- Can always construct prefix code with property: $H(p) \leq \bar{\ell} < H(p) + 1$

Exercise 1: Properties of Expected Values

Proof the following properties of expected values

- Linearity

$$\mathrm{E}\{\, a\,X + b\,Y \,\} = a\,\mathrm{E}\{\, X \,\} + b\,\mathrm{E}\{\, Y \,\}$$

- For two independent random variables X and Y, we have

$$\mathrm{E}\{\, XY \,\} = \mathrm{E}\{\, X \,\}\ \mathrm{E}\{\, Y \,\}$$

- Iterative expectation rule

$$\mathrm{E}\{\, \mathrm{E}\{\, g(X) \,|\, Y \,\} \,\} = \mathrm{E}\{\, g(X) \,\}$$

Exercise 2: Correlation and Independence

Investigate the relationship between independence and correlation.

> Two random variables X and Y are said to be *correlated* if and only if their covariance $\sigma^2_{XY} = \mathrm{E}\{\,(X - \mathrm{E}\{\,X\,\})(Y - \mathrm{E}\{\,Y\,\})\,\}$ is not equal to 0.

(a) Can two independent random variables X and Y be correlated?

(b) Are two uncorrelated random variables X and Y also independent?

Exercise 3: Marginal Pmf of Markov Process (Optional)

Given is a stationary discrete Markov process with the alphabet $\mathcal{A} = \{a, b, c\}$ and the conditional pmf

$$p(x_k \mid x_{k-1}) = \mathrm{P}(X_k = x_k \mid X_{k-1} = x_{k-1})$$

listed in the table below

x_n	$p(x_n \mid a)$	$p(x_n \mid b)$	$p(x_n \mid c)$	$p(x_n)$
a	0.90	0.15	0.25	?
b	0.05	0.80	0.15	?
c	0.05	0.05	0.60	?

➡ Determine the marginal pmf $p(x) = \mathrm{P}(X_k = x)$.

Exercise 4: Unqiue Decodability

Given is a discrete iid process **X** with the alphabet $\mathcal{A} = \{a, b, c, d, e, f, g\}$.
The pmf $p_X(x)$ and five example codes are listed in the following table.

x	$p_X(x)$	A	B	C	D	E
a	1/3	1	0	00	01	1
b	1/9	0001	10	010	101	100
c	1/27	000000	110	0110	111	100000
d	1/27	00001	1110	0111	010	10000
e	1/27	000001	11110	100	110	000000
f	1/9	001	111110	101	100	1000
g	1/3	01	111111	11	00	10

(a) Calculate the entropy of the source.

(b) Calculate the average codeword lengths and the redundancies for the given codes.

(c) Which of the given codes are uniquely decodable codes?

Exercise 5: Prefix Codes

Given is a random variable X with the alphabet $\mathcal{A}_X = \{a, b, c, d, e, f\}$.
Two sets of codeword lengths are given in the following table.

letter	set A	set B
a	2	1
b	2	3
c	2	3
d	3	3
e	3	4
f	4	4

(a) For which set(s) can we construct a uniquely decodable code?

(b) Develop a prefix code for the set(s) determined in **(a)**.

(c) Consider the prefix code(s) developed in **(b)**. Is it possible to find a pmf p for which the developed code yields an average codedword length $\bar{\ell}$ equal to the entropy $H(p)$? If yes, write down the probability masses.

Exercise 6: Maximum Entropy (Optional)

Consider an iid process with an alphabet of size N (i.e., the alphabet includes N different letters).

(a) Calculate the entropy H_{uni} for the case that the pmf represents a uniform pmf:

$$\forall k, \quad p_k = \frac{1}{N}$$

(b) Show that for all other pmfs (i.e., all non-uniform pmfs), the entropy H is less than H_{uni}.