

# Codes and Cryptography

Jorge L. Villar

MAMME, Fall 2015

## PART II

## The Setting (I)

**Perfect environment:** No storage or communication errors or excessive message delivery delays

### Goals:

- represent information in a systematic way to allow storage or transmission
- do it unambiguously and efficiently

**Imperfect environment:** Storage or communication errors occur with some positive probability

### Goals:

- represent information with some redundancy allowing error detection and correction
- without losing too much efficiency

## Outline

- 1 Information Coding: The setting
- 2 Uniquely Decodable Codes
- 3 Codeword Lengths

## Basic Concepts

Information is usually represented as a sequence of symbols in a fixed finite alphabet (e.g., natural languages, sampled audio or digital images)

**Encoding:** “A systematic method to translate sequences from one alphabet to another”

**Decoding:** “Recovering the original sequence from the encoded one”

**Source alphabet:**  $\mathcal{A} = \{\alpha_1, \dots, \alpha_n\}$

**Target alphabet:**  $\mathcal{B} = \{\beta_1, \dots, \beta_r\}$

**Encoding map:**  $E : \mathcal{A} \rightarrow \mathcal{B}^*$

**( $r$ -ary) Code:**  $\mathcal{C} = E(\mathcal{A}) \subset \mathcal{B}^*$

**Codeword:** Element in the code  $\mathcal{C}$

$E$  extends naturally to the set  $\mathcal{A}^*$  by concatenation:

$$E(\emptyset) = \emptyset \quad E(xy) = E(x)E(y)$$

## Some Examples (I)

### Unary representation of natural numbers (digits)

'1' → 1            '4' → 1111            '7' → 1111111  
'2' → 11           '5' → 11111           '8' → 11111111  
'3' → 111           '6' → 111111        '9' → 111111111

Not good for concatenation! Needs a separator or delimiter!

'0' → 0            '5' → 111110  
'1' → 10           '6' → 1111110  
'2' → 110           '7' → 11111110  
'3' → 1110           '8' → 111111110  
'4' → 11110        '9' → 1111111110

Now 23501 = ('2','3','5','0','1') →  $\underbrace{110}_2 \underbrace{1110}_3 \underbrace{111110}_5 \underbrace{0}_0 \underbrace{10}_1$

## Some Examples (III)

### ASCII, another fixed-length binary code

' ' → 00100000	'0' → 00110000	'@' → 01000000
'!' → 00100001	'1' → 00110001	'A' → 01000001
"" → 00100010	'2' → 00110010	'B' → 01000010
'#' → 00100011	'3' → 00110011	'C' → 01000011
'\$' → 00100100	'4' → 00110100	'D' → 01000100
'%' → 00100101	'5' → 00110101	'E' → 01000101
'&' → 00100110	'6' → 00110110	'F' → 01000110
'"' → 00100111	'7' → 00110111	'G' → 01000111
...	...	...

## Some Examples (II)

### Binary representation of natural numbers (digits)

'0' → 0            '4' → 100            '8' → 1000  
'1' → 01           '5' → 101           '9' → 1001  
'2' → 10           '6' → 110  
'3' → 11           '7' → 111

Also needs a separator or turn it into a fixed length code

'0' → 0000        '4' → 0100           '8' → 1000  
'1' → 0001        '5' → 0101           '9' → 1001  
'2' → 0010        '6' → 0110  
'3' → 0011        '7' → 0111

Now 23501 = ('2','3','5','0','1') →  $\underbrace{0010}_2 \underbrace{0011}_3 \underbrace{0101}_5 \underbrace{0000}_0 \underbrace{0001}_1$

## Some Examples (IV)

### QR Codes:

"Hello World!!!"



```

111111110011011001100101111111
10000010000011110010101000001
10111010010111001010001011101
10111010011001110111001011101
10111010011101101011001011101
1000001011100110000101000001
1111110101010101010101111111
0000000000001010011110000000
10010110101100010111110100000
00101101101010111001101110011
11011110011010001101000111110
11000100110100110101100000110
11110011010010001000101000010
1101110111110010100110101110
001111110111001111001111111
11110101100001101110100001000
1110011011010111010000001010
0111100010000010111110101111
1011001001111100010001001011
0001100001011101000010011011
1011001101001010110011111000
0000000011000101110100010010
11111110001011110011101010010
10000010101011001010100010111
1011101001001011001111110010
1011101010011111010100111111
1011101001011100001101011101
100001001011110000010110010
111111101111000001110010010

```

# Outline

- 1 Information Coding: The setting
- 2 Uniquely Decodable Codes
- 3 Codeword Lengths

# An Example of Non-Uniquely Decodable Code (I)

**Alphabets:**

$$\mathcal{A} = \{\alpha_1, \dots, \alpha_5\}$$

$$\mathcal{B} = \{0, 1\}$$

**Code:**

$$\mathcal{C} = \{01, 001, 010, 10010, 111\}$$

**Encoding map:**

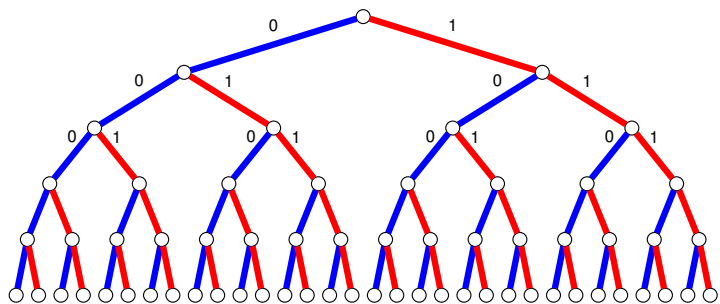
$$E(\alpha_1) = 01; \quad E(\alpha_2) = 001; \quad E(\alpha_3) = 010;$$

$$E(\alpha_4) = 10011; \quad E(\alpha_5) = 111$$

**Problem: Decode the sequence**

100100110010010111

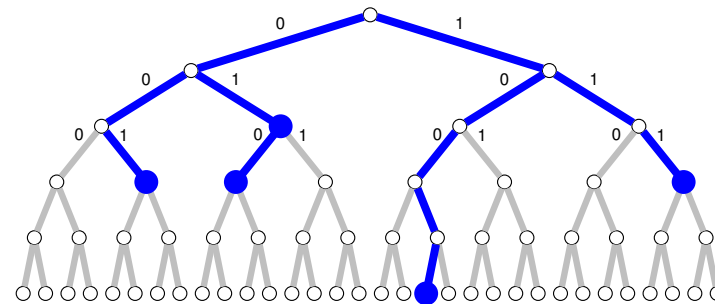
# An Example of Non-Uniquely Decodable Code (II)



$$\mathcal{C} = \{01, 001, 010, 10010, 111\}$$

100100110010010111

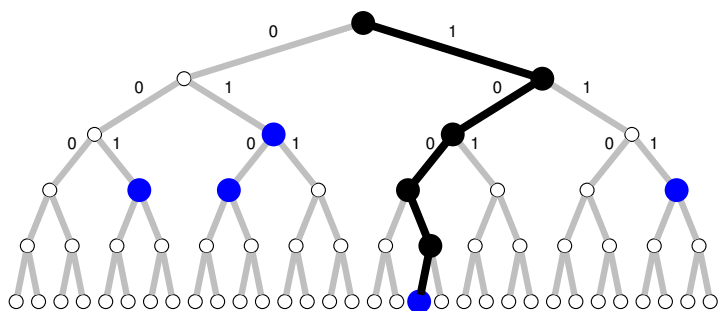
# An Example of Non-Uniquely Decodable Code (II)



$$\mathcal{C} = \{01, 001, 010, 10010, 111\}$$

100100110010010111

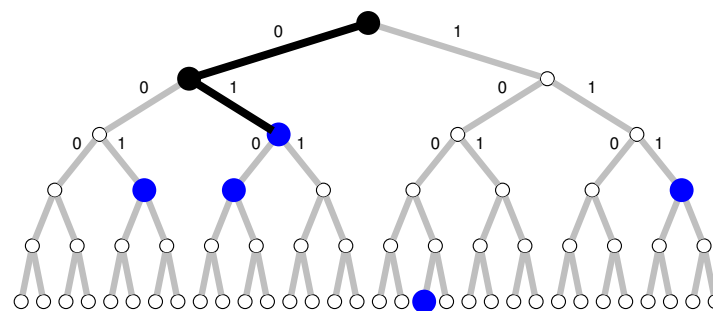
# An Example of Non-Uniquely Decodable Code (II)



$\mathcal{C} = \{01, 001, 010, 10010, 111\}$

100100110010010111

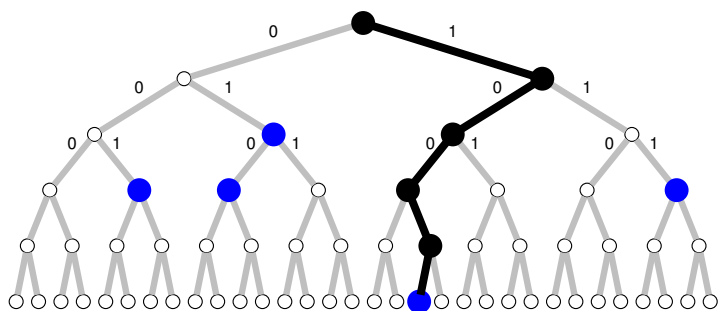
# An Example of Non-Uniquely Decodable Code (II)



$\mathcal{C} = \{01, 001, 010, 10010, 111\}$

10010 01 10010010111

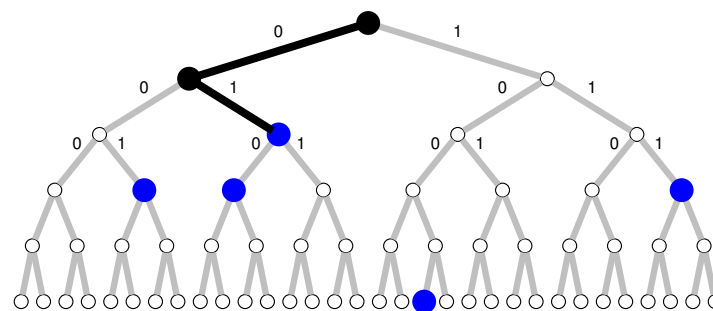
# An Example of Non-Uniquely Decodable Code (II)



$\mathcal{C} = \{01, 001, 010, 10010, 111\}$

10010 01 10010 010111

# An Example of Non-Uniquely Decodable Code (II)

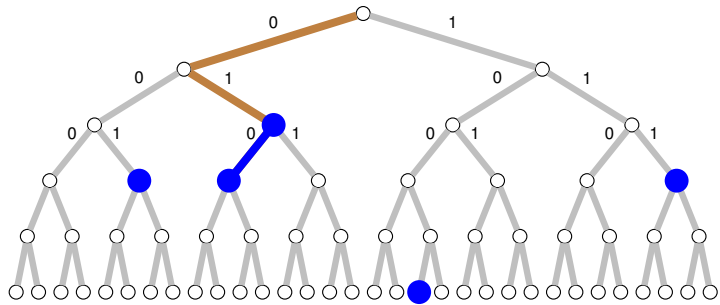


$\mathcal{C} = \{01, 001, 010, 10010, 111\}$

10010 01 10010 01 0111



## An Example of Non-Uniquely Decodable Code (II)



$$\mathcal{C} = \{01, 001, 010, 10010, 111\}$$

01 is a prefix of 010!

## Sardinas-Patterson Theorem

### Definition (Uniquely Decodable)

A code  $\mathcal{C} \subset \mathcal{B}^*$  is **uniquely decodable** if any sequence in  $\mathcal{C}^*$  splits uniquely into codewords, or equivalently, the extended encoding function  $E : \mathcal{A}^* \rightarrow \mathcal{B}^*$  is injective.

We recursively define the sets

$$\mathcal{C}_1 = \mathcal{C} \diamond \mathcal{C} = \{y \in \mathcal{B}^{\geq 1} \mid \exists x \in \mathcal{C} \ xy \in \mathcal{C}\}$$

$$\mathcal{C}_{k+1} = \mathcal{C}_k \diamond \mathcal{C} = \{y \in \mathcal{B}^{\geq 1} \mid \exists x \in \mathcal{C} \ xy \in \mathcal{C}_k \text{ or } \exists x \in \mathcal{C}_k \ xy \in \mathcal{C}\}$$

### Theorem (Sardinas-Patterson [1953])

A code  $\mathcal{C}$  is uniquely decodable if and only if  $\mathcal{C} \cap \left( \bigcup_{k=1}^{\infty} \mathcal{C}_k \right) = \emptyset$

▶ details...

## An Example of Non-Uniquely Decodable Code (III)

The code is not **uniquely decodable** (i.e., the extended encoding function is not injective)

$$\mathcal{C} = \{01, 001, 010, 10010, 111\}$$

$$\underbrace{01} \underbrace{001} = \underbrace{010} \underbrace{01}$$

$$\underbrace{01} \underbrace{01} \underbrace{001} \underbrace{001} = \underbrace{010} \underbrace{10010} \underbrace{01}$$

## Sardinas-Patterson Theorem: Details

Assume  $\mathcal{C}$  is not uniquely decodable. Then there exists a minimal  $S_0 \in \mathcal{C}^{\geq 1}$  such that  $S_0 = x_1 x_2 \cdots x_k = y_1 y_2 \cdots y_l$  which implies that  $y_1 = x_1 s_1$  or  $x_1 = y_1 s_1$  for some  $s_1 \in \mathcal{C}_1$ .

Remove the shortest initial codeword (say  $x_1$ ) and define  $S_1$  such that  $S_0 = x_1 S_1$ . Then  $S_1 = x_2 \cdots x_k = s_1 y_2 \cdots y_l$

Similarly,  $s_1 = x_2 s_2$  or  $x_2 = s_1 s_2$  for some  $s_2 \in \mathcal{C}_2$ . Then, remove the shortest of  $s_1$  and  $x_2$ . The remaining sequence  $S_2$  is either  $S_2 = x_3 \cdots x_k = s_2 y_2 \cdots y_l$  or

$$S_2 = s_2 x_3 \cdots x_k = y_2 \cdots y_l.$$

Continue until at some point  $S_i = x_j = s_j$  or  $S_i = s_j = y_j$ . Then,  $\mathcal{C} \cap \mathcal{C}_i \neq \emptyset$ .

◀ go back...

## Sardinas-Patterson Theorem: An Example

$$\mathcal{C} = \{01, 001, 010, 10010, 111\}$$

$$\mathcal{C}_1 = \{0\}$$

$$\mathcal{C}_2 = \{1, 01, 10\}$$

Not a uniquely-decodable code!

As  $\mathcal{C}$  is finite, the sequence  $\mathcal{C}_1, \mathcal{C}_2, \dots$  becomes constant (e.g.,  $\emptyset$ ) or cyclic after a finite number of steps.

In the example,  $\mathcal{C}_1 \neq \mathcal{C}_2 \neq \dots \neq \mathcal{C}_7 = \mathcal{C}_8 = \dots$

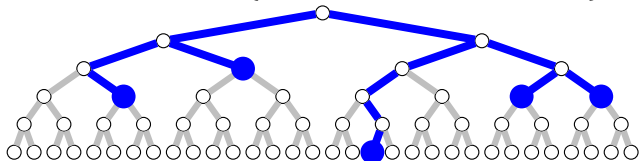
## Prefix-Free Codes

### Definition (Prefix-free)

$\mathcal{C}$  is **prefix-free** if  $\mathcal{C}_1 = \emptyset$

- Any prefix-free code is uniquely-decodable
- Any fixed-length code is prefix-free
- Subtrees rooted at codewords of a prefix-free code are disjoint

An example:  $\mathcal{C} = \{001, 01, 10010, 110, 111\}$



## Sardinas-Patterson Theorem: Another Example

$$\mathcal{C} = \{01, 0011, 011, 10010, 111\}$$

$$\mathcal{C}_1 = \{1\}$$

$$\mathcal{C}_2 = \{0010, 11\}$$

$$\mathcal{C}_3 = \{1\} = \mathcal{C}_1$$

$$\bigcup_{i=1}^{\infty} \mathcal{C}_i = \mathcal{C}_1 \cup \mathcal{C}_2 = \{1, 0010, 11\}$$

Empty intersection with  $\mathcal{C}$

$\mathcal{C}$  is uniquely-decodable!

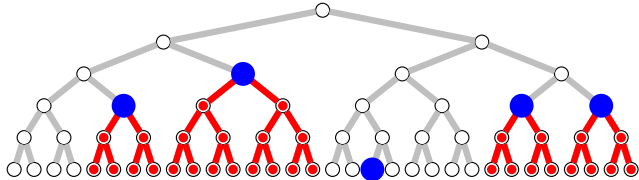
## Outline

- 1 Information Coding: The setting
- 2 Uniquely Decodable Codes
- 3 Codeword Lengths

## Codeword Lengths in a Prefix-Free Code

Subtrees rooted at codewords are disjoint.

$$\mathcal{C} = \{001, 01, 10010, 110, 111\} \quad l_{max} = 5$$



Each  $w \in \mathcal{C}$  of length  $l(w)$  corresponds to  $r^{l_{max}-l(w)}$  leaves

$$\sum_{w \in \mathcal{C}} r^{l_{max}-l(w)} \leq r^{l_{max}} \quad \text{where } l(w) \text{ is the length of } w$$

or

$$\sum_{w \in \mathcal{C}} r^{-l(w)} \leq 1$$

## Kraft's Inequality (II)

Given  $l_1 \leq \dots \leq l_n$  such that  $S = \sum_{i=1}^n r^{-l_i} \leq 1$  define the

increasing sequence  $x_k = \sum_{i=1}^{k-1} r^{-l_i}$ .

Observe that  $x_k r^{l_k}$  is an integer and  $x_k r^{l_k} < S r^{l_k} \leq r^{l_k}$ .

Now, assign to each  $l_k$  the codeword  $w_k \in \{0, \dots, r-1\}^{l_k}$  obtained as the  $r$ -ary representation of  $x_k r^{l_k}$ .

The resulting code is prefix-free. Indeed, to be  $w_k$  a prefix of  $w_m$ , necessarily  $k < m \leq n$  and  $(x_m - x_k) r^{l_m} < r^{l_m - l_k}$ , which is not possible since  $x_m - x_k \geq x_{k+1} - x_k = r^{-l_k}$ .

## Kraft's Inequality (I)

### Theorem (Kraft's Inequality [1949])

There exists a prefix-free  $r$ -ary code  $\mathcal{C}$  with codewords of respective lengths  $l_1, \dots, l_n$  if and only if

$$\sum_{i=1}^n r^{-l_i} \leq 1$$

- the "only if" part comes from the ideas in last slide
- the proof of the "if" part is constructive (next slide)

## McMillan's Inequality (I)

### Theorem (McMillan's Inequality [1956])

There exists a uniquely-decodable  $r$ -ary code  $\mathcal{C}$  with codewords of respective lengths  $l_1, \dots, l_n$  if and only if

$$\sum_{i=1}^n r^{-l_i} \leq 1$$

- the "if" part is a direct consequence of Kraft's inequality
- the proof of the "only if" is more elaborated (see next slide)

## McMillan's Inequality (II)

Let's assume the existence of a uniquely-decodable  $r$ -ary code  $\mathcal{C}$  with codeword lengths  $l_1 \leq \dots \leq l_n$  and define  $S = \sum_{i=1}^n r^{-l_i}$ .

As  $\mathcal{C}$  is uniquely-decodable then all concatenations of  $k$  codewords are different. Now, in the code  $\mathcal{C}^k$  for some  $k > 1$

$$S_k = \sum_{i_1=1}^n \dots \sum_{i_k=1}^n r^{-(l_{i_1} + \dots + l_{i_k})} = \left( \sum_{i_1=1}^n r^{-l_{i_1}} \right) \dots \left( \sum_{i_k=1}^n r^{-l_{i_k}} \right) = S^k$$

Let  $n_j^{(k)}$  be the number of codewords of length  $j$  in  $\mathcal{C}^k$ , which

$$\text{fulfils } n_j^{(k)} \leq r^j. \text{ Then, } S^k = \sum_{j=kl_1}^{kl_n} n_j^{(k)} r^{-j} \leq kl_n - kl_1 + 1.$$

Thus,  $\ln S \leq \frac{\ln(kl_n - kl_1 + 1)}{k}$  and in the limit  $k \rightarrow \infty$  we obtain  $S \leq 1$ .

## Final Remarks

- Prefix-free codes perform as well as uniquely-decodable ones . . . but decoding prefix-free codes is simpler
- The larger the arity is, the easier it is to accommodate a given codeword length sequence to obtain a prefix-free code

Some questions about optimization:

- Which is the smallest arity of the code for a given length sequence?
- Which is the shortest code for given input alphabet and arity?
- **How can we optimize the code length if the input symbols occur with different probabilities?**

## Codes and Cryptography

Jorge L. Villar

MAMME, Fall 2015

**END OF PART II**