COMSATS Institute of Information Technology

ECI750 Multimedia Data Compression

Lecture 3

Information Theory and Coding – II

Dr. Shadan Khattak

Department of Electrical Engineering

COMSATS Institute of Information Technology - Abbottabad

Coding

- Coding
  - It is the **assignment of binary sequences to elements of an alphabet**.
  - The set of binary sequences is called a *code*.
  - The individual members of the set are called *codewords*.
  - An alphabet is a collection of symbols called *letters*.
  - For example:
    - The **English alphabet** has 26 letters.
    - The alphabet of this book consist of 26 uppercase letters, 26 lowercase letters, and punctuation marks.
    - ASCII codes:

a: 1000011

A: 1000001

, : 0011010

Dr. Shadan Khattak
Department of Electrical Engineering
CIIT - Abbottabad

Coding

- Coding
  - When same number of bits are used to represent each symbol in an alphabet, the code is called a *fixed-length code*.
  - The average number of bits used per symbol is called the *rate* of the code (or simply the *code rate*).
  - If we use fewer bits to represent symbols that occur more often, on the average, we would use fewer bits per symbol. This type of code is called a *variable-length code*.

Coding

- **Uniquely Decodable Codes**
  - Is it sufficient for a code to have a good code rate?
  - Let’s consider an example to find the answer.
  - Alphabet S has four letters a_1, a_2, a_3, a_4 i.e., $S = \{a_1, a_2, a_3, a_4\}$ with the following probabilities: $P(a_1) = \frac{1}{2}, P(a_2) = \frac{1}{4}, P(a_3) = P(a_4) = \frac{1}{8}$.
  - Entropy of S: 1.75 bits/symbol
  - Let’s consider four different coding schemes for this source.

Coding

- **Uniquely Decodable Codes**

Letters	Probability	Code 1	Code 2	Code 3	Code 4
a_1	0.5	0	0	0	0
a_2	0.25	0	1	10	01
a_3	0.125	1	00	110	011
a_4	0.125	10	11	111	0111
Average length		1.125	1.25	1.75	1.875

$$Average\ Length\ (l) = \sum_{i=1}^{4} P(a_i)n(a_i)$$

$P(a_i)$: $Probability\ of\ the\ letter a_i$

$n(a_i)$: $number\ of\ bits\ in\ the\ codeword\ for\ letter a_i$

Coding

- **Uniquely Decodable Codes**
  - Code 1:
    - Same code assigned to two different letters
    - ambiguous decoding.
    - ⇒ *Code should be unique*
  - Code 2:
    - Transmitter sends: $a_2 a_1 a_1 (100)$
    - At the receiver side, 100 can be decoded as: $a_2 a_1 a_1$ or $a_2 a_3$
    - Can be decoded in many ways
    - ⇒ *not uniquely decodable*
  - Code 3:
    - Uniquely decodable
    - Simple decoding rule:
      - accumulate bits until you have a 0 or you get three consecutive 1's.
    - The decoder knows the moment a code is complete ⇒ *Instantaneous code*

Coding

- **Uniquely Decodable Codes**
  - Code 4:
    - Even simpler decoding rule:
      - Accumulate bits until you see a 0. The bit before 0 is the last bit of the previous codeword.
    - The decoder has to wait till the beginning of the next codeword before it knows that the current codeword is complete.

Coding

- **<u>Example:</u>** Consider the coding scheme below. Using this code, can we decode the following message: 011111111111111111

Letter	Codeword
a_1	0
a_2	01
a_3	11

- Instantaneous? No
- Uniquely decodable? Yes

⇒ *For a code to be uniquely decodable, it is not necessary to be instantaneous!*

Coding

- **Unique Decodability Test**
  - So how can we systematically test if a code is uniquely decodable?
  - To know this, let's understand a few terms first.
  - Prefix: If the beginning sub-sequence of codeword b is equal to codeword a, then a is a *prefix*.
  - Dangling Suffix: If a is a prefix of b, then the sub-sequence of b excluding the prefix a, is called a *dangling suffix*.

Coding

- **<u>Unique Decodability Test</u>**
  - Check if any codeword is a prefix of another codeword
  - If yes, add the dangling suffix to the code as a codeword.
  - Repeat the procedure until:
    1. You get a dangling suffix that is a codeword
    2. There are no more unique dangling suffixes
  - If you get (1), the code is not uniquely decodable

Coding

- **Example**:
  - Consider Code 5
  - List of codewords: {0, 01, 11}
  - Codeword 0 is a prefix of the codeword 01.
  - The dangling suffix is 1.
  - No other pair for which one element of the pair is the prefix of the other.
  - After augmenting the dangling suffix, the new codeword list is: {0, 01, 11, 1}
  - Codeword 0 is a prefix of codeword 01. 1 is the dangling suffix which has been added as a new codeword.
  - Codeword 1 is a prefix of codeword 11. Again, 1 is the dangling suffix which was not originally a codeword.
  - There is no dangling suffix now. So we cannot add to the list any further.
  - $\Rightarrow$ *uniquely decodable*

TABLE 2.2 Code 5.

Letter	Codeword
a_1	0
a_2	01
a_3	11

Dr. Shadan Khattak
Department of Electrical Engineering
CIIT - Abbottabad

Coding

- **Example:**
  - Consider Code 6
  - List of codeword: {0, 01, 10}
  - Codeword 0 is a prefix of the codeword 01.
  - The dangling suffix is 1.
  - No other pair for which one element of the pair is the prefix of the other.
  - After augmenting the dangling suffix, the new codeword list is: {0, 01, 10, 1}
  - Codeword 0 is a prefix of codeword 01. 1 is the dangling suffix which has been added as a new codeword.
  - Codeword 1 is a prefix of codeword 10. 0 is the dangling suffix which is already a codeword.
  - There is no dangling suffix now. So we cannot add to the list any further.
  - $\Rightarrow$ *not uniquely decodable*

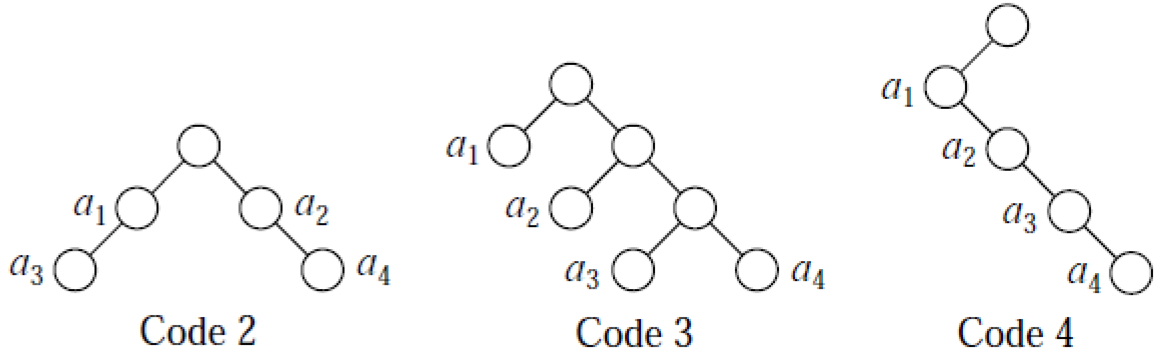
TABLE 2.3 **Code 6.**

Letter	Codeword
a_1	0
a_2	01
a_3	10

Dr. Shadan Khattak
Department of Electrical Engineering
CIIT - Abbottabad

Coding

- **Prefix Codes**
  - A code in which no codeword is a prefix of another code is called a *prefix-free code (or, a prefix code)*
  - A code tree is a binary tree where each codeword of a code corresponds to a node in the binary tree.
  - A prefix code has all codewords on leaf nodes



Dr. Shadan Khattak
Department of Electrical Engineering
CIIT - Abbottabad

Coding

- **The Kraft-McMillan Inequality**
  - Two important points:
    - Is there a necessary condition on the codeword lengths of uniquely decodable codes? Yes
    - Can we always find a prefix code that satisfies the above necessary condition? Yes

    ⇒ if we have a uniquely decodable code that is not a prefix code, we can always find a prefix code with the same codeword length.
  - ***Theorem 1:*** *Let C be a code with N codewords with lengths* $l_1, l_2, \dots, l_N$*. If C is uniquely decodable, then*

$$K(C) = \sum_{i=1}^{N} 2^{-l_i} \leq 1$$

(This inequality is known as the *Kraft-McMillan inequality*)

Coding

- <u>**The Kraft-McMillan Inequality**</u>
  - ***Theorem 2:*** *Given a set of integers* $l_1, l_2, \ldots, l_N$ *that satisfy the inequality* $\sum_{i=1}^{N} 2^{-l_i} \leq 1$, *we can always find a prefix code with codeword lengths* $l_1, l_2, \ldots, l_N$.

Dr. Shadan Khattak
Department of Electrical Engineering
CIIT - Abbottabad