**ECE 8873**
***Data Compression and Modeling***

# Lecture 3:
# Introduction to Lossless Compression

School of Electrical and Computer Engineering
Georgia Institute of Technology
Spring, 2004



## Asymptotic EquiPartition Principle (AEP)

- Let $S$ be the signal observation space (or simply the source) in which a random variable $X$ is defined. Let $X^N = (X_1, X_2, \cdots, X_N)$ be a random sequence in the cartesian product space formed by $S^N = S \times S \times \cdots \times S$. If $N$ is sufficiently large, then any outcome $x^N = (x_1, x_2, \cdots, x_N)$ is almost surely to belong to a subset of $S^N$ having only $2^{NH(S)}$ members, all having probability close to $2^{-NH(S)}$. Or,

$$\left| \frac{-\log p(x^N)}{N} - H(S) \right| < \varepsilon$$



## Global vs. Local Models

- Universal Codes: Collection of codes, pick best.
- Adaptive Codes: Adjust code parameters in real time. Often by fitting model to source and then coding for model.



## Coding as a Task

- Representation of analog signal for digital transmission or storage; often integrated with A/D conversion
- Compression of digital information to reduce transmission or storage requirement; compression can also be realized in analog domain
- efficiency is defined in terms of bandwidth or storage required for the delivery of a fixed amount of information such as a second of speech, a video frame
- Result of coding is a sequence of digital, often binary, symbols
- The sequence of digital symbols may or may not have explicit "**delimiter**."

## Information Rate, Bit Rate

- Bits per second
- Bits per sample
- Bits per pixel
- Bits per letter
- …..

But, information rate or bit rate alone is not a sufficiently accurate indicator of the quality of the reconstructed signal.



## Fixed vs. Variable Rate/Length Coding

- Depends on the nature of the source and practical considerations in implementation
- Relevant coding theorems in Information Theory
- If were not for practical implementations, the two are asymptotically equivalent in terms of source entropy argument.
- Advantages and disadvantages

Alphabet : $A = \{a_1, a_2, \cdots, a_M\}$ code base : $B = \{b_1, b_2, \cdots, b_N\}$

binary code : $B = \{0,1\}$

$$A^* = \cdots \times A \times A^3 \times A \times A^4 \cdots$$

$$B^* = \cdots \times B \times B^2 \times B^5 \times B \cdots$$



## Our Options in the Probability Space

- Let $A = \{a_1, a_2, \cdots, a_M\}$ be the alphabet.
- We can define a probability space based on a random experiment with $A$ as the set of all outcomes.
- Since we ARE most likely to be interested in encoding a sequence of symbols, we can define a probability space based on a composite random experiment with the sample/observation space defined as the cartesian product of $A$ ; that is,

  $A^* = \cdots \times A \times A \times A \cdots$ ⟸ x does not imply independence
- Again, we have the flexibility to redefine the sample space and its mapping to the code set; e.g.

  $A^* = \cdots \times A \times A^3 \times A \times A^4 \cdots$



## Variable Rate and Variable length Coding

Rate:

**Bit** count in a unit time interval.

Variable Rate: (e.g., Huffman Codes)

Source symbol may be emitted at regular time instances; the encoded result, i.e., the corresponding bit stream, however, may not be produced at equal pace.

Length:

**Number of source symbols** (letters) encoded in one encoding cycle

Variable Length: (e.g., Tunstall Codes)

Number of source symbols (letters) encoded in each cycle may not be the same.

## Lossless vs. Lossy Compression

- Lossless compression focuses on the code structure according to the probability of symbols so as to achieve minimum average code length
- Lossy compression involves an additional dimension, namely the amount of distortion if the symbol is replaced by another one as a result of compression.
  - Choice of distortion measures: what matters
  - Simultaneous consideration of probability of occurrence and the incurred distortion



## Lossless Compression

- Assume no obvious redundancy in the original signal representation; e.g.,

  $A = \{a_1, a_2\} = \{0000, 1111\}$ $C = \{0,1\}$
- Then, we need to explore the frequency of occurrences or probability of letters/symbols to minimize the average code length. In doing so, the length of codeword, each of which is used to represent a letter/symbol, has to bear some relationship with the corresponding probability, resulting in variable length code and, assuming that the source puts out letters at regular intervals, variable rate coding.



## Fundamentals

- A lossless code consists of
  - An encoder $\alpha: \quad A^* \to B^* = \{0,1\}^*$

    If $x \in A$ and $\alpha(x)$ appear as a parsable units in their respective domain, we can speak of its code length $l(\alpha(x))$ . We are most interested in the average code length $E[l(\alpha(x))]$

    watch out for $E$ *(over what space)?*
  - A decoder $\beta: \quad B^* \to A^*$ such that $\beta(\alpha(X)) = X$

Questions of interest:

1. How small can $E[l(\alpha(x))]$ be?
2. How to achieve optimal performance?
3. How to practically achieve near optimal performance?



## Uniquely Decodable Codes (UDC)

- Consider fixed-rate or fixed length code:

| letter | code |
|---|---|
| a1 | 00 |
| a2 | 01 |
| a3 | 10 |
| a4 | 11 |

- Sequences that contain this set of code are uniquely decodable, only if we know where the code blocks start; *may need synchronization from time to time*.

## Uniquely Decodable Codes

- Unique decodability: any sequence of codewords can be decoded in one and only one way.

| Letter | Codeword |
|---|---|
| a1 | 0 |
| a2 | 01 |
| a3 | 11 |

0111111111111111111

Uniquely decodable if the overall length is known

| Letter | Codeword |
|---|---|
| a1 | 0 |
| a2 | 01 |
| a3 | 10 |

0101010101010101010

Not uniquely decodable

- Review Morse code - its use of space as the "delimiter" makes the code a ternary code – not an efficient use of bit resources - but eliminates the ambiguity issue.



## Uniquely Decodable Codes

- A sequence of uniquely decodable codewords must also be parsable:
  - Embedded parsing
  - Explicit parsing
- Instantaneous parsing/decoding vs. non-instantaneous parsing/decoding
- Example:

| Letter | Codeword |
|---|---|
| a1 | 0 |
| a2 | 10 |
| a3 | 110 |
| a4 | 111 |

| Letter | Codeword |
|---|---|
| a1 | 0 |
| a2 | 01 |
| a3 | 011 |
| a4 | 0111 |



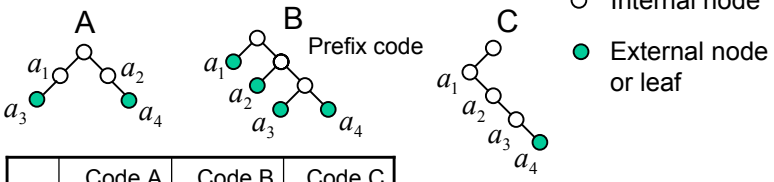
## Test for Unique Decodability

- $||\mathbf{a}||=k$, $||\mathbf{b}||=n$, $k<n$.
- If $\mathbf{a}=(\mathbf{b})_1^k$, $\mathbf{a}$ is a prefix of $\mathbf{b}$ and $(\mathbf{b})_{k+1}^n$ is the dangling suffix.
- Test procedure:
  1. Start with the codebook (the list of all codewords);
  2. For each codeword, find all codewords that are its prefix; append the list with the corresponding dangling suffix;
  3. Repeat 2 for every entry of the larger codebook until
     - Obtaining a dangling suffix that is a codeword – **not uniquely decodable**;
     - no more unique dangling suffixes – **uniquely decodable**.



## Code Structure & Tree Representation

- Prefix(-free) Code - A code in which no codeword is a prefix of another codeword – ensures unique decodability but not vice versa
- Code tree:




| | Code A | Code B | Code C |
|---|---|---|---|
| $a_1$ | 0 | 0 | 0 |
| $a_2$ | 1 | 10 | 01 |
| $a_3$ | 00 | 110 | 011 |
| $a_4$ | 11 | 111 | 0111 |

Codewords of a prefix code are only associated with external nodes (e.g. code B)

## Two Important Results

- Necessary condition on the codeword lengths of uniquely decodable codes.
- There always exists a prefix code that satisfies the necessary condition.

**The Kraft-McMillan Inequality**

Let *C* be a code with $N$ codewords, each of length $l_1, l_2, \cdots, l_N$, respectively. If *C* is uniquely decodable, $K(C) = \sum_{i=1}^{N} 2^{-l_i} \le 1$

**Existence of Prefix Code**

Given a set of integers $l_1, l_2, \cdots, l_N$ that satisfy the inequality $\sum_{i=1}^{N} 2^{-l_i} \le 1$, we can always find a prefix code with codeword lengths $l_1, l_2, \cdots, l_N$.



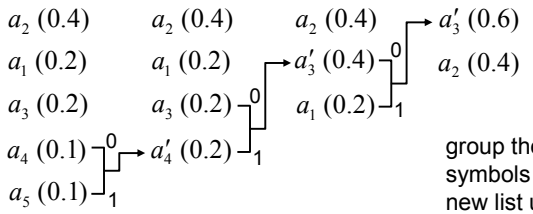
## Consequences

- Any uniquely decodable code has lengths which satisfy the Kraft inequality.
- For any collection of lengths satisfying the Kraft inequality, one can construct a prefix code that has this set of lengths.
- Given ANY uniquely decodable code, one can find a prefix code (using tree structure) with the same set of code lengths.



## Huffman Coding

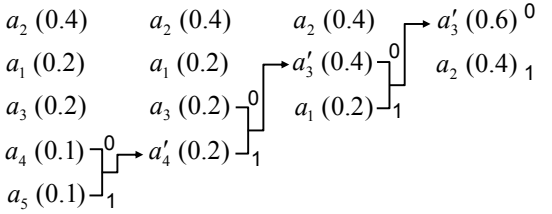
- Huffman codes are prefix codes.
- Huffman code is optimal for a given model or probability measure of the "source" (i.e. subject to the applicability of the measure). [*But, where does this knowledge come from*?]
- Two necessary conditions for an optimal binary prefix code:
  - symbols with higher probability have shorter codewords;
  - the two symbols that occur least frequently have same codeword length; their codewords differ only in the last bit.



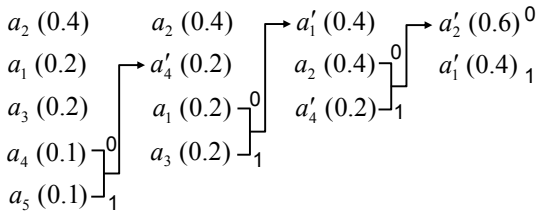

group the least probable symbols and reshuffle the new list until only two left



## Minimum Variance Huffman Codes




$a_2\ (0.4) = 1$
$a_1\ (0.2) = 01$
$a_3\ (0.2) = 000$
$a_4\ (0.1) = 0010$
$a_5\ (0.1) = 0011$




$a_2\ (0.4) = 00$
$a_1\ (0.2) = 10$
$a_3\ (0.2) = 11$
$a_4\ (0.1) = 010$
$a_5\ (0.1) = 011$

## Optimality of Huffman Codes

**Lower bound on expected length**:
The expected length L(C,X) of a (or any) uniquely decodable code is bounded below by H(X).

**Compression limit of symbol codes:**
For a source ensemble X, there exists a prefix code $H(X) \leq L(C,X) < H(X) + 1$. Huffman code is one such code.

Q: When does L(C,X)=H(X)?

when the probabilities are powers of ½.

Note: The upper bound may still be rather loose if the source entropy is low. This can be changed by grouping several symbols, say m, together. The upper bound is then H(X)+(1/m).



## m-ary Huffman Codes

- Two necessary conditions for an optimal m-ary prefix code:
  - symbols with higher probability have shorter codewords;
  - symbols that occur least frequently have same codeword length; their codewords differ only in the last bit.
- In binary Huffman codes, each round of grouping reduces the symbols in the intermediate (or internal) alphabet by one (i.e., 2-1)
- In m-ary Huffman codes, each round of grouping would reduce the symbols of internal alphabet by m-1; if the size of alphabet is M, we group (M modulo (m-1)) least probable symbols in the first round of grouping; in each subsequent reduction, m least probable symbols will be grouped together.
- The rest is similar to binary case.



## Tunstall Codes

- Codewords are of equal length (i.e., no code delimiter needed), but number of consecutive symbols grouped for each encoding cycle is not – variable length coding.
- Coding principle:
  - Source sequence can be parsed into chunks of symbols that appear in the codebook; each chunk corresponds to a codeword of fixed bit length;
  - Maximize the average length of such symbol chunks
- Codebook construction:
  - Split the most probable symbol or chunk and augment it with all elementary symbols; repeat.