Source Coding

1. Source symbols encoded in binary
2. The average codelength must be reduced
3. Remove redundancy $\Rightarrow$ reduces bit-rate

Consider a discrete memoryless source on the alphabet

$$S = \{s_0, s_1, \cdots, s_k\}$$

Let the corresponding probabilities be

$$\{p_0, p_1, \cdots, p_k\}$$

and codelengths be

$$\{l_0, l_1, \cdots, l_k\}.$$

Then, the average codelength(average number of bits per symbol) of the source is defined as

$$\bar{L} = \sum_{k=0}^{K-1} p_k l_l$$

If L_{min} is the minimum possible value of $\bar{L}$, then the coding efficiency of the source is given by η.

$$\eta = \frac{L_{min}}{\bar{L}}$$

For an efficient code η approaches unity.

The question: What is smallest average codelength that is possible?

The Answer: Shannon's source coding theorem

Given a discrete memoryless source of entropy $\mathcal{H}(s)$, the average codeword length $\bar{L}$ for any distortionless source encoding scheme is bounded by

$$\bar{L} \geq \mathcal{H}(s)$$

Since, $\mathcal{H}(s)$ is the fundamental limit on the average number of bits/symbol, we can say

$$L_{min} \geq \mathcal{H}(s)$$

$$\implies \eta = \frac{\mathcal{H}(s)}{\bar{L}}$$

Data Compaction:

1. Removal of redundant information prior to transmission.
2. Lossless data compaction – no information is lost.
3. A source code which represents the output of a discrete memoryless source should be uniquely decodable.

Source Coding Schemes for Data Compaction

Prefix Coding

1. The *Prefix Code* is variable length source coding scheme where no code is the prefix of any other code.
2. The prefix code is a uniquely decodable code.
3. But, the converse is not true i.e., all uniquely decodable codes may not be prefix codes.

Table 1: Illustrating the definition of prefix code

Symbol	Prob.of Occurrence	Code I	Code II	Code III
s_0	0.5	0	0	0
s_1	0.25	1	10	01
s_2	0.125	00	110	011
s_3	0.125	11	111	0111

Table 2: Table is reproduced from S.Haykin's book on Communication Systems

From 1 we see that Code I is not a prefix code. Code II is a prefix code. Code III is also uniquely decodable but not a prefix code.

Prefix codes also satisfies Kraft-McMillan inequality which is given by

$$\sum_{k=0}^{K-1} 2^{-l_k} \leq 1$$

Kraft-McMillan inequality maps codewords to a binary tree as shown is Figure 1.

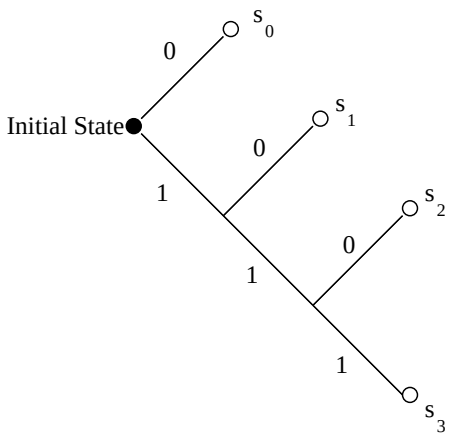

Figure 1: Decision tree for Code II

Given a discrete memoryless source of entropy $\mathcal{H}(s)$, a prefix code can be constructed with an average code-word length $\bar{l}$, which is

bounded as follows:

$$\mathcal{H}(s) \leq \bar{(}L) < \mathcal{H}(s) + 1 \tag{1}$$

The left hand side of the above equation, the equality is satisfied owing to the condition that, any symbol s_k is emitted with the probability

$$p_k = 2^{-l_k} \tag{2}$$

where, l_k is the length of the codeword assigned to the symbol s_k. Hence, from Eq. 2, we have

$$\sum_{k=0}^{K-1} 2^{-l_k} = \sum_{k=0}^{K-1} p_k = 1 \tag{3}$$

With this condition, the Kraft-McMillan inequality tells that a prefix code can be constructed such that the length of the codeword assigned to source symbol s_k is $-\log_2 p_k$. Therefore, the average codeword length is given by

$$\bar{L} = \sum_{k=0}^{K-1} \frac{l_k}{2^{l_k}} \tag{4}$$

and the corresponding entropy is given by

$$\begin{aligned}\mathcal{H}(s) &= \sum_{k=0}^{K-1}\left(\frac{1}{2^{l_k}}\right)\log_2(2^{l_k}) \\ &= \sum_{k=0}^{K-1}\frac{l_k}{2^{l_k}}\end{aligned} \tag{5}$$

Hence, from Eq. 5, the equality condition on the leftside of Eq. 1, $\bar{L} = \mathcal{H}(s)$ is satisfied.

To prove the inequality condition we will proceed as follows:

Let $\bar{L_n}$ denote the average codeword length of the extended prefix code. For a uniquely decodable code, $\bar{L_n}$ is the smallest possible.

Huffman Coding

1. Huffman code is a prefix code
2. The length of codeword for each symbol is roughly equal to the amount of information conveyed.
3. The code need not be unique (see Figure 3)

A Huffman tree is constructed as shown in Figure. 3, (a) and (b) represents two forms of Huffman trees. We see that both schemes have same average length but different variances.

Variance is a measure of the variability in codeword lengths of a source code. It is defined as follows:

$$\sigma^2 = \sum_{k=0}^{K-1} p_k (l_k - \bar{L})^2 \tag{6}$$

where, p_k is the probability of kth symbol. l_k is the codeword length of kth symbol and $\bar{L}$ is the average codeword length. It is reasonable to choose the huffman tree which gives greater variace.

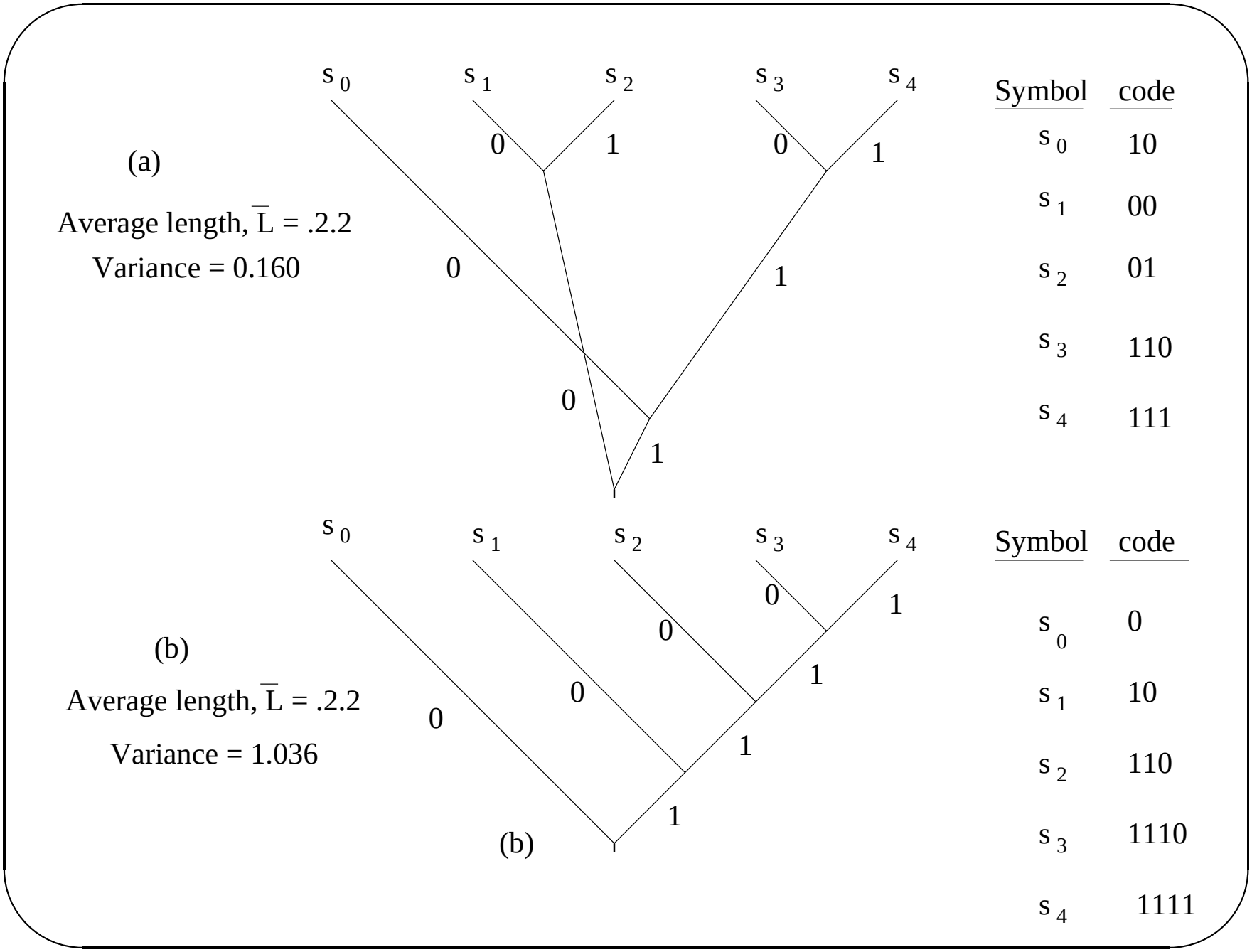

Figure 2: Huffman tree

Drawbacks:

1. Requires proper statistics.
2. Cannot exploit relationships between words, phrases etc.,
3. Does not consider redundancy of the language.

Lempel-Ziv Coding

1. Overcomes the drawbacks of Huffman coding
2. It is an adaptive and simple encoding scheme.
3. When applied to English text it achieves 55% in contrast to Huffman coding which achieves only 43%.
4. Encodes patterns in the text

This algorithm is accomplished by parsing the source data stream into segments that are the shortest subsequences not encountered previously. (see Figure 3 – the example is reproduced from S.Haykin's book on "Communication Systems.")

Let the input sequence be

000101110010100101.....

We assume that 0 and 1 are known and stored in codebook

subsequences stored : 0, 1
Data to be parsed: 000101110010100101.....

The shortest subsequence of the data stream encountered for the first time and not seen before is 00
subsequences stored: 0, 1, 00
Data to be parsed: 0101110010100101.....

The second shortest subsequence not seen before is 01; accordingly, we go on to write

Subsequences stored: 0, 1, 00, 01
Data to be parsed: 01110010100101.....

We continue in the manner described here until the given data stream has been completely parsed. The code book is shown below:

Numerical positions:	1	2	3	4	5	6	7	8	9
subsequences:	0	1	00	01	011	10	010	100	101
Numerical Repre – sentations:			11	12	42	21	41	61	62
Binary encoded blocks:			0010	0011	1001	0100	1000	1100	1101

Figure 3: Lempel-Ziv Encoding