

Lecture 3: Source Coding Theorem and Huffman Coding

ENGR 76 lecture notes — April 7, 2026

David Tse, Stanford University

Overview

In the previous lecture we stated the bound

$$H(X) \leq \bar{\ell}_{\min} \leq H(X) + 1$$

for the minimum expected codeword length over all prefix-free codes. In this lecture we explain why this bound is true. We first formulate the optimization problem, introduce Kraft's inequality, show that any prefix-free code must satisfy the lower bound $\bar{\ell} \geq H(X)$, and then show that Shannon's construction achieves an average length at most $H(X) + 1$. We conclude with the recursive structure behind the Huffman algorithm for constructing an optimal prefix-free code.

1 The optimization problem

Let

$$X \in \{0, 1, \dots, M-1\}$$

be a discrete random variable with

$$P(X = i) = p_i, \quad i = 0, 1, \dots, M-1.$$

We want to design a prefix-free binary code with codeword lengths

$$\ell_0, \ell_1, \dots, \ell_{M-1}$$

so as to minimize the expected codeword length

$$\bar{\ell} = \sum_{i=0}^{M-1} p_i \ell_i.$$

Thus the problem is

$$\min_{\{\ell_i\}} \sum_{i=0}^{M-1} p_i \ell_i \quad \text{subject to the code being prefix-free.}$$

We denote by $\bar{\ell}_{\min}$ the minimum value over all prefix-free codes.

2 Kraft's inequality

The prefix-free constraint can be translated into a clean numerical condition on the lengths.

Kraft's inequality

For any prefix-free binary code with codeword lengths $\ell_0, \ell_1, \dots, \ell_{M-1}$,

$$\sum_{i=0}^{M-1} 2^{-\ell_i} \leq 1.$$

A useful way to visualize Kraft's inequality is through dyadic intervals: a codeword of length ℓ_i occupies an interval of width $2^{-\ell_i}$, and prefix-freeness means these intervals are disjoint.

For a “reasonable” full prefix-free code tree, we in fact have equality:

$$\sum_{i=0}^{M-1} 2^{-\ell_i} = 1.$$

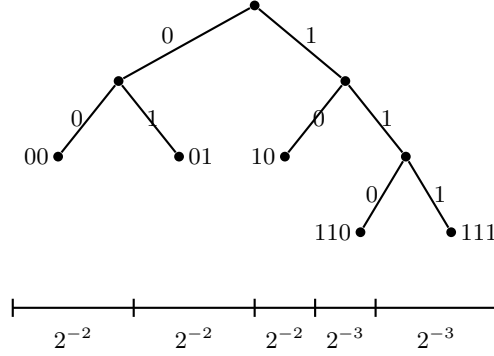


Figure 1: A prefix-free code tree with codewords 00, 01, 10, 110, 111. The corresponding codeword lengths are 2, 2, 2, 3, 3, so

$$\sum_i 2^{-\ell_i} = 3 \cdot 2^{-2} + 2 \cdot 2^{-3} = \frac{3}{4} + \frac{1}{4} = 1.$$

The intervals below the tree are drawn to scale, with widths $2^{-\ell_i}$.

2.1 Converse: Kraft's inequality is also sufficient

The important additional point is that Kraft's inequality is not only necessary, but also sufficient for the existence of a prefix-free code.

Converse to Kraft's inequality

If positive integers $\ell_0, \ell_1, \dots, \ell_{M-1}$ satisfy

$$\sum_{i=0}^{M-1} 2^{-\ell_i} \leq 1,$$

then there exists a binary prefix-free code whose codeword lengths are exactly $\ell_0, \ell_1, \dots, \ell_{M-1}$.

A simple way to see this is to sort the lengths in increasing order and assign codewords greedily on the binary tree.

At depth ℓ , there are 2^ℓ binary strings. If some shorter codewords have already been assigned, then the number of strings at depth ℓ ruled out by the prefix condition is exactly

$$\sum_{j: \ell_j < \ell} 2^{\ell - \ell_j}.$$

Therefore the number of available strings at depth ℓ is

$$2^\ell - \sum_{j: \ell_j < \ell} 2^{\ell - \ell_j} = 2^\ell \left(1 - \sum_{j: \ell_j < \ell} 2^{-\ell_j} \right).$$

If we wish to assign all codewords having length ℓ , say N_ℓ of them, then Kraft's inequality implies

$$\sum_{j: \ell_j < \ell} 2^{-\ell_j} + N_\ell 2^{-\ell} \leq 1.$$

Multiplying by 2^ℓ , this becomes

$$N_\ell \leq 2^\ell \left(1 - \sum_{j: \ell_j < \ell} 2^{-\ell_j} \right),$$

which says exactly that there are enough available nodes at depth ℓ to place all codewords of that length.

So one can assign the codewords level by level without conflict, and the resulting code is prefix-free. Hence Kraft's inequality exactly characterizes the feasible integer length vectors for prefix-free codes.

3 The lower bound $\bar{\ell} \geq H(X)$

We now show that every prefix-free code must have average length at least the entropy.

Claim

For any prefix-free code,

$$\bar{\ell} \geq H(X).$$

Hence

$$\bar{\ell}_{\min} \geq H(X).$$

Because of Kraft's inequality, the lengths must satisfy

$$\sum_{i=0}^{M-1} 2^{-\ell_i} \leq 1.$$

For an optimal full prefix-free code tree, we may take equality:

$$\sum_{i=0}^{M-1} 2^{-\ell_i} = 1.$$

Define

$$x_i = 2^{-\ell_i}.$$

Then

$$\ell_i = -\log_2 x_i,$$

and the optimization becomes

$$\min_{\{x_i\}} \sum_{i=0}^{M-1} p_i (-\log_2 x_i) \quad \text{subject to} \quad \sum_{i=0}^{M-1} x_i = 1,$$

with the additional restriction that each x_i must be of the form $2^{-\ell_i}$ for some integer ℓ_i .

If we now allow the x_i 's to be arbitrary positive real numbers satisfying

$$\sum_{i=0}^{M-1} x_i = 1, \quad x_i > 0,$$

then we obtain a *relaxation* of the original optimization problem. Relaxing the constraints can only decrease the minimum value, so the minimum of

$$-\sum_{i=0}^{M-1} p_i \log_2 x_i$$

over all probability vectors (x_i) gives a lower bound on the minimum achievable expected codeword length.

This relaxed optimization is minimized when

$$x_i^* = p_i,$$

and the minimum value is

$$-\sum_{i=0}^{M-1} p_i \log_2 p_i = \sum_{i=0}^{M-1} p_i \log_2 \frac{1}{p_i} = H(X).$$

Therefore every prefix-free code satisfies

$$\bar{\ell} \geq H(X).$$

4 When does equality hold?

The above argument suggests the ideal real-valued lengths

$$\ell_i^* = \log_2 \frac{1}{p_i}.$$

Indeed,

$$\sum_{i=0}^{M-1} 2^{-\ell_i^*} = \sum_{i=0}^{M-1} 2^{-\log_2(1/p_i)} = \sum_{i=0}^{M-1} p_i = 1.$$

If the ℓ_i^* were integers, then they would be valid codeword lengths and we would get

$$\bar{\ell}_{\min} = H(X).$$

Thus equality holds exactly when all probabilities are dyadic:

$$p_i = 2^{-k_i} \quad \text{for integers } k_i.$$

In general, however,

$$\log_2 \frac{1}{p_i}$$

need not be an integer, so it cannot always be used directly as a codeword length.

5 Shannon code and the upper bound

A natural way to convert the ideal real-valued lengths into valid integer lengths is to round up:

$$\ell_i^{(S)} = \left\lceil \log_2 \frac{1}{p_i} \right\rceil.$$

This is called the **Shannon code**.

Since

$$\left\lceil \log_2 \frac{1}{p_i} \right\rceil \leq \log_2 \frac{1}{p_i} + 1,$$

we get

$$\sum_i p_i \ell_i^{(S)} \leq \sum_i p_i \left(\log_2 \frac{1}{p_i} + 1 \right) = H(X) + \sum_i p_i = H(X) + 1.$$

Therefore there exists a prefix-free code (due to the converse to Kraft's inequality) such that

$$\bar{\ell} \leq H(X) + 1.$$

Combining this with the lower bound gives the source coding theorem for prefix-free codes.

Source coding theorem for prefix-free codes

If $\bar{\ell}_{\min}$ is the minimum expected codeword length over all prefix-free codes, then

$$H(X) \leq \bar{\ell}_{\min} \leq H(X) + 1.$$

6 Optimal prefix-free coding: Huffman algorithm

The Shannon code gives a code within one bit of entropy, but it need not be optimal. To achieve $\bar{\ell}_{\min}$, we use the **Huffman algorithm**.

6.1 Base case: two symbols

When $M = 2$, say the alphabet is $\{a, b\}$, the only reasonable prefix-free binary code has one bit for each symbol:

$$a \mapsto 0, \quad b \mapsto 1.$$

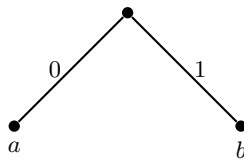


Figure 2: Base case for a prefix-free code on two symbols.

So in this case

$$\ell_a = \ell_b = 1, \quad \bar{\ell} = p_a + p_b = 1.$$

6.2 Structure of an optimal tree in general

Suppose the symbol probabilities are ordered as

$$p_a \geq p_b \geq p_c.$$

In an optimal prefix-free code tree, the symbols with the smallest probabilities should be placed deepest in the tree. In particular, the two least likely symbols should appear as siblings at the lowest level.

This gives the key recursive step.

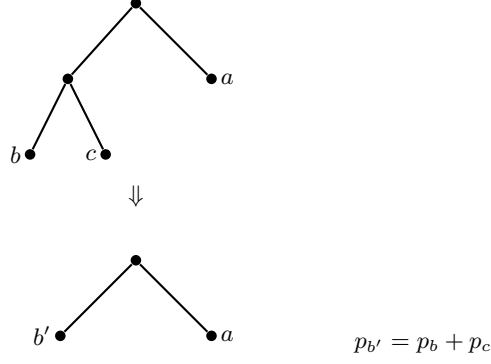


Figure 3: If b and c are the two least likely symbols, they should be siblings at the lowest level. They can be merged into a single super-symbol b' .

6.3 Recursive reduction

For a binary alphabet with two letters, the code is immediate:

$$a \mapsto 0, \quad b \mapsto 1.$$

For a larger alphabet:

1. Find the two letters with the smallest probabilities.
2. Merge them into a single super-letter whose probability is the sum of the two probabilities.
3. Solve the coding problem for the reduced alphabet.
4. Split the super-letter back into the two original letters by appending 0 and 1.

This is the Huffman algorithm.

7 Summary

1. The problem is to minimize

$$\bar{\ell} = \sum_i p_i \ell_i$$

over all prefix-free codes.

2. Prefix-free codeword lengths satisfy Kraft's inequality

$$\sum_i 2^{-\ell_i} \leq 1.$$

3. Replacing the dyadic quantities $x_i = 2^{-\ell_i}$ by arbitrary positive reals satisfying $\sum_i x_i = 1$ gives a relaxation of the optimization problem.
4. This relaxation yields the lower bound

$$\bar{\ell} \geq H(X).$$

5. Equality holds when the ideal lengths

$$\ell_i^* = \log_2 \frac{1}{p_i}$$

are integers, equivalently when the probabilities are dyadic.

6. Shannon's construction

$$\ell_i^{(S)} = \left\lceil \log_2 \frac{1}{p_i} \right\rceil$$

gives the upper bound

$$\bar{\ell} \leq H(X) + 1.$$

7. Therefore

$$H(X) \leq \bar{\ell}_{\min} \leq H(X) + 1.$$

8. An optimal prefix-free code can be constructed recursively by the Huffman algorithm, which repeatedly merges the two least likely symbols.

Takeaway

Entropy gives the fundamental lower bound on the average number of bits needed for lossless compression. Shannon coding gets within one bit of this bound, while Huffman coding achieves the optimal prefix-free code.