Discrete Information Source

- A source alphabet, $X = \{x_1, \ldots, x_n\}$.
- A source distribution, $P = \{p_1, \ldots, p_n\}$.
- A code alphabet, $S = \{s_1, \ldots, s_r\}$.
- A set of code words, $U = \{u_1, \ldots, u_n\}$.

Kraft's Inequality

It is a necessary and sufficient condition for the existence of an instantaneous code that

$$\sum_{i=1}^{n} \frac{1}{r^{\ell_i}} \leq 1$$

where r is the size of the code alphabet, and ℓ_i is the length of the code word, u_i.

Proof Let w_1 be the number of code words of length one. Since there are only r symbols in the code alphabet:

$$w_1 \leq r.$$

Kraft's Inequality (contd.)

We observe that there are $r - w_1$ unused symbols in the code alphabet. Let w_2 be the number of code words of length two. There are $r - w_1$ possibilities for the first symbol and r possibilities for the second symbol. It follows that:

$$w_2 \leq (r - w_1)r = r^2 - w_1 r.$$

Kraft's Inequality (contd.)

Let w_3 be the number of code words of length three. There are $(r - w_1)r - w_2$ possibilities for the first two symbols, and r possibilities for the third symbol. It follows that:

$$w_3 \leq [(r - w_1)r - w_2]r = r^3 - w_1 r^2 - w_2 r.$$

Kraft's Inequality (contd.)

If m is the maximum length of the code words, then

$$w_m \leq r^m - w_1 r^{m-1} - w_2 r^{m-2} - \ldots - w_{m-1} r.$$

Dividing by r^m gives

$$\begin{aligned}
w_m r^{-m} &\leq 1 - w_1 r^{-1} - w_2 r^{-2} - \ldots - w_{m-1} r^{-m+1} \\
0 &\leq 1 - w_1 r^{-1} - w_2 r^{-2} - \ldots - w_{m-1} r^{-m+1} - w_m r^{-m} \\
-1 &\leq -w_1 r^{-1} - w_2 r^{-2} - \ldots - w_{m-1} r^{-m+1} - w_m r^{-m}.
\end{aligned}$$

Multiplying by -1 gives

$$\begin{aligned}
w_1 r^{-1} + w_2 r^{-2} + \ldots + w_{m-1} r^{-m+1} + w_m r^{-m} &\leq 1 \\
\sum_{j=1}^{m} w_j r^{-j} &\leq 1.
\end{aligned}$$

Kraft's Inequality (contd.)

But this is just

$$\sum_{j=1}^{m} w_j r^{-j} = \sum_{j=1}^{m} w_j \frac{1}{r^j} \leq 1$$

which can be expanded as follows:

$$\underbrace{\frac{1}{r^1} + \ldots + \frac{1}{r^1}}_{w_1} + \underbrace{\frac{1}{r^2} + \ldots + \frac{1}{r^2}}_{w_2} + \ldots + \underbrace{\frac{1}{r^m} + \ldots + \frac{1}{r^m}}_{w_m} \leq 1.$$

Since $l_1 = l_2 = \ldots = l_{w_1} = 1$ and $l_{w_1+1} = l_{w_1+2} = \ldots = l_{w_1+w_2} = 2$, etc. and since $w_1 + w_2 + \ldots + w_m = n$ it follows that:

$$\underbrace{\frac{1}{r^{l_1}} + \ldots + \frac{1}{r^{l_{w_1}}}}_{w_1} + \underbrace{\frac{1}{r^{l_{w_1+1}}} + \ldots + \frac{1}{r^{l_{w_1+w_2}}}}_{w_2} + \ldots +$$

$$\underbrace{\frac{1}{r^{l_{w_1+w_2+\ldots+w_{m-1}+1}}} + \ldots + \frac{1}{r^{l_{w_1+w_2+\ldots+w_m}}}}_{w_m} = \sum_{i=1}^{n} \frac{1}{r^{\ell_i}} \leq 1.$$

Example

An information source, X, has source alphabet, $\{x_1, x_2, x_3, x_4\}$. We would like to encode messages using a binary code alphabet, $\{0, 1\}$, with code words of the following lengths:

$$\{\ell_1 = 1, \ell_2 = 2, \ell_3 = 3, \ell_4 = 4\}$$

After evaluating Kraft's inequality:

$$\frac{1}{2^1} + \frac{1}{2^2} + \frac{1}{2^3} + \frac{1}{2^4} \approx 0.94 \leq 1$$

we conclude that an instantaneous code with these lengths exists. In fact,

$$\{u_1 = 0, u_2 = 10, u_3 = 110, u_4 = 1110\}$$

is one such code.

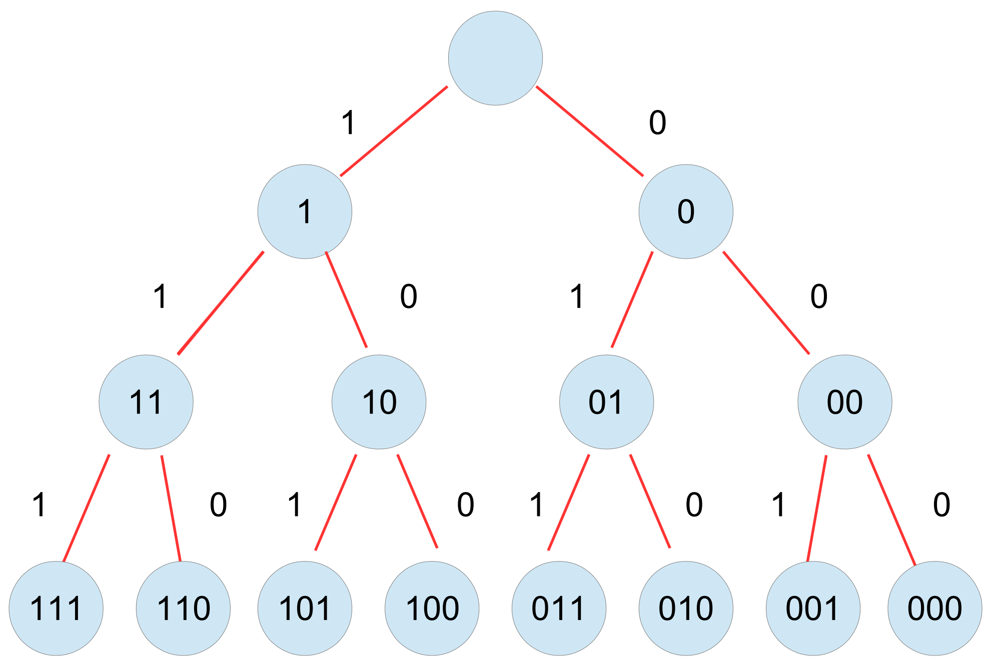


Figure 1: Complete binary tree of height three showing code words associated with interior vertices and leaves.

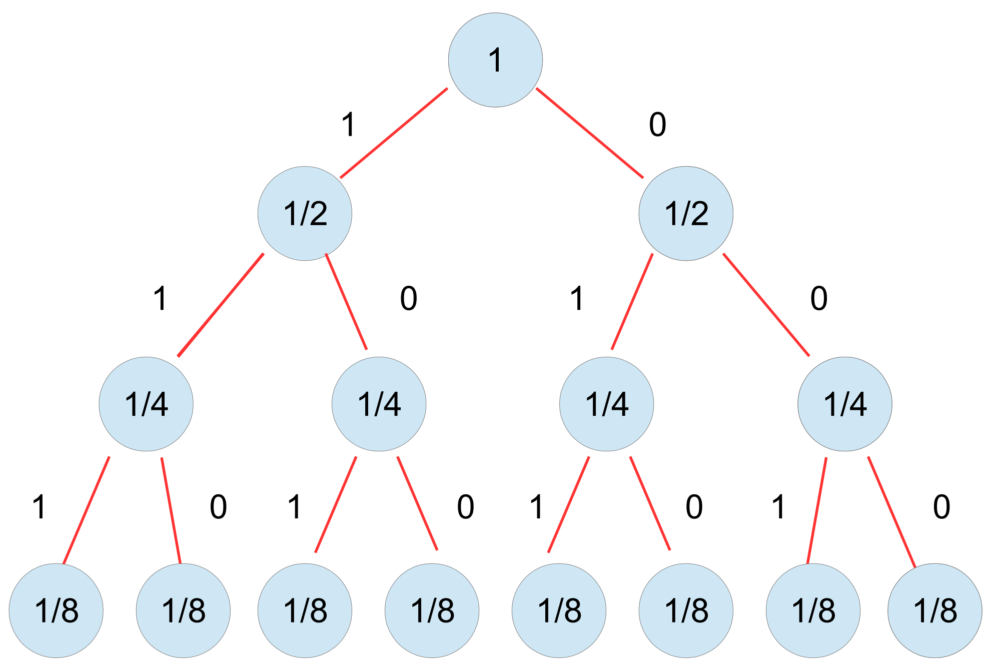


Figure 2: Complete binary tree of height three showing $1/2^{\ell_i}$ for each vertex where ℓ_i is the length of the code word associated with the vertex.

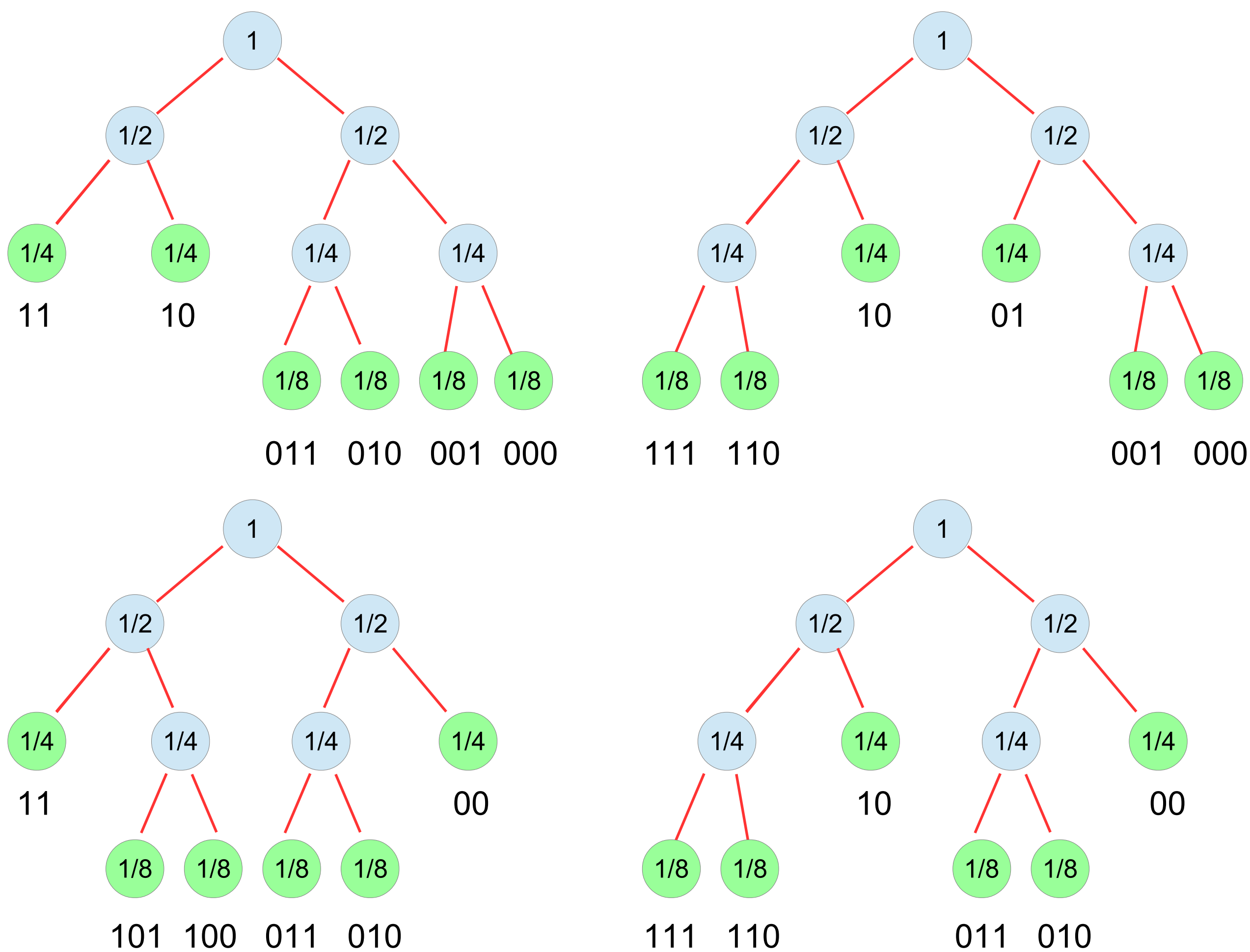


Figure 3: Four different instantaneous binary codes for a source alphabet of length six. Code words are associated with leafs not interior vertices and $\sum_{i=1}^{6} 1/2^{\ell_i} = 1$ in all cases.

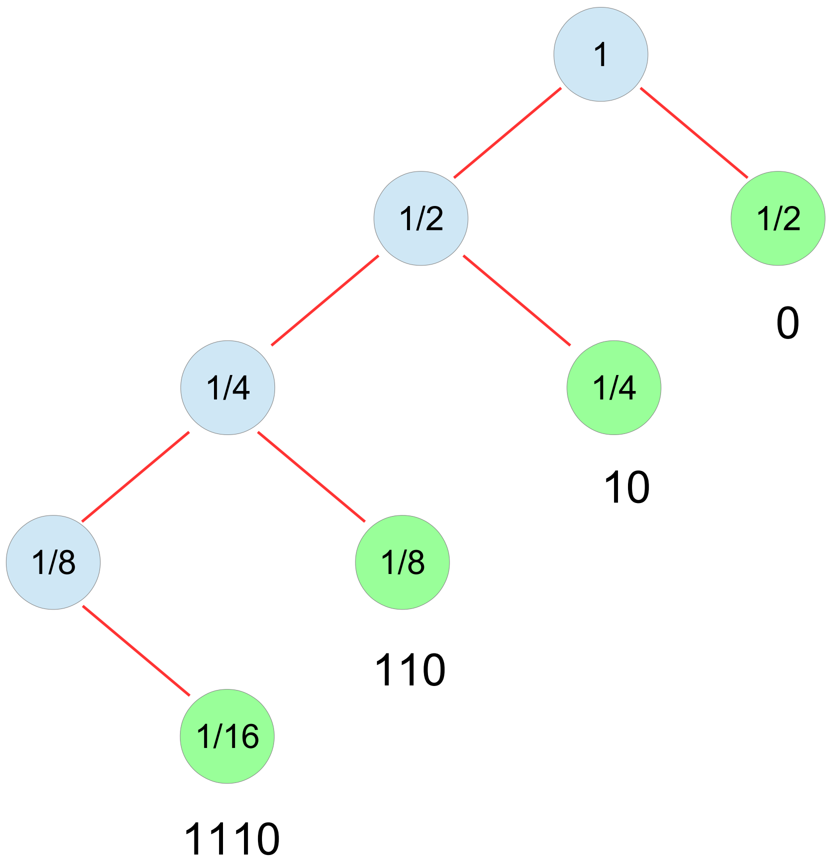


Figure 4: An instantaneous binary code where $\{\ell_1 = 1, \ell_2 = 2, \ell_3 = 3, \ell_4 = 4\}$ and $\sum_{i=1}^{4} 1/2^{\ell_i} \approx 0.94$.

Source Coding Theorem

Consider a set of n code words, $U = \{u_1, \ldots, u_n\}$, with lengths, $L = \{\ell_1, \ldots, \ell_n\}$, and probability distribution, $P = \{p_1, \ldots, p_n\}$. All code words are composed of symbols from the code alphabet, $\{s_1, \ldots, s_r\}$. If Kraft's inequality is satisfied, then

$$\frac{H_U}{\log r} \leq \langle L \rangle = \sum_{i=1}^{n} p_i \ell_i$$

with equality iff $p_i = 1/r^{\ell_i}$ for $1 \leq i \leq n$.

Source Coding Theorem (contd.)

Proof We will show that $\frac{H_U}{\log r} \leq \langle L \rangle$ by showing that $H_U - \langle L \rangle \log r \leq 0$:

$$
\begin{aligned}
H_U - \langle L \rangle \log r &= -\sum_{i=1}^{n} p_i \log p_i - \log r \sum_{i=1}^{n} p_i \ell_i \\
&= -\sum_{i=1}^{n} (p_i \log p_i + p_i \ell_i \log r) \\
&= \sum_{i=1}^{n} p_i \log \left(\frac{1}{p_i r^{\ell_i}} \right)
\end{aligned}
$$

Source Coding Theorem (contd.)

We now take advantage of the fact that $\log a \leq (a-1)\log e$:

$$
\begin{aligned}
H_U - \langle L\rangle \log r &= \sum_{i=1}^{n} p_i \log\left(\frac{1}{p_i r^{\ell_i}}\right) \\
&\leq \log e \sum_{i=1}^{n} p_i \left(\frac{1}{p_i r^{\ell_i}} - 1\right) \\
&\leq \log e \sum_{i=1}^{n} \left(\frac{1}{r^{\ell_i}} - p_i\right) \\
&\leq \log e \sum_{i=1}^{n} \frac{1}{r^{\ell_i}} - \log e \sum_{i=1}^{n} p_i \\
&\leq \log e \sum_{i=1}^{n} \frac{1}{r^{\ell_i}} - \log e
\end{aligned}
$$

Source Coding Theorem (contd.)

Since the Kraft inequality

$$\sum_{i=1}^{n} \frac{1}{r^{\ell_i}} \leq 1$$

is satisfied, it follows that

$$H_U - \langle L \rangle \log r \leq 0$$

which can be rearranged to yield

$$\frac{H_U}{\log r} \leq \langle L \rangle \, .$$

Why the Unit of Entropy is Bits

We just showed that

$$\frac{H_U}{\log r} \leq \langle L \rangle \,.$$

When the code is binary, $\log r$ is one. Consequently, for binary codes:

$$H_U \leq \langle L \rangle \,.$$

We now see the connection between the units of Shannon's entropy and the 0s and 1s which are used to represent information in a computer's memory. A message cannot be encoded using a string of 0s and 1s which is shorter on average than its information content when measured in bits!

Coding Efficiency

The Source Coding Theorem tells us that

$$\frac{H_U}{\log r} \leq \langle L \rangle\,.$$

Since H_U and $\log r$ are positive

$$0 \;\leq\; \frac{H_U}{\log r} \leq \langle L \rangle\,.$$

Dividing the inequality by $\langle L \rangle$ yields a number between zero and one representing the *efficiency* of a code:

$$0 \;\leq\; \frac{H_U}{\langle L \rangle \log r} \;\leq\; 1.$$

Given that some codes are more efficient than others, it is natural to ask how we can find efficient codes.

Balanced Tree Coding

1. Merge the source symbols into r sets, so that the sums of the probabilities in each set are as equal as possible.
2. Assign a unique code alphabet symbol to the members of each set.
3. Repeat this process until the sets are of size r or less.

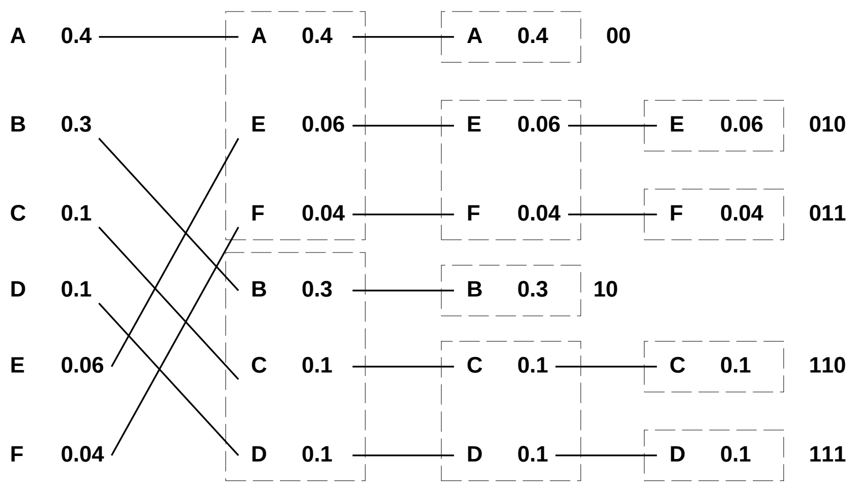


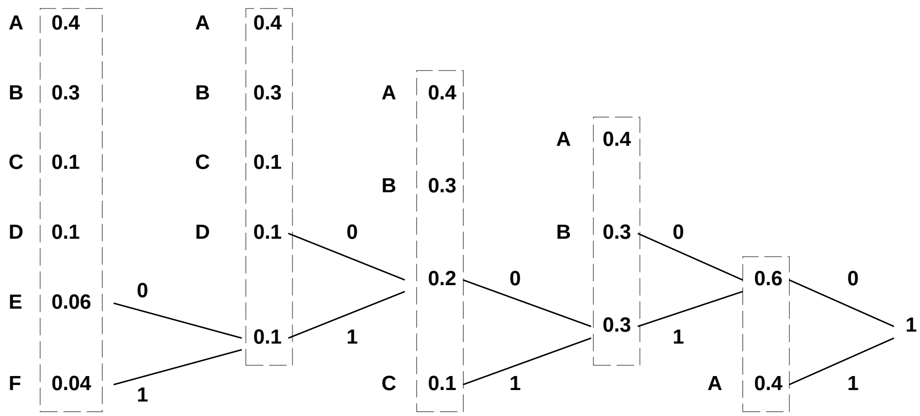
Figure 5: Balanced tree coding example.

Huffman Coding

1. Sort the source alphabet in order of decreasing probability. These are the leaves of the "coding tree."
2. Merge the r source symbols with smallest probability into a new "source symbol" with probability equal to the sum of the r smallest probabilities. This is an interior node of the coding tree.
3. Repeat this process until only one source symbol (with probability one) remains. This is the root of the coding tree.

Huffman Coding (contd.)

4. To find the codeword for a given source symbol trace the coding tree from the root to the source symbol (leaf), *e.g.*, when $r = 2$, add a zero to the codeword when traversing a left branch and a one when traversing a right branch.



A	1
B	00
C	011
D	0100
E	01010
F	01011

Figure 6: Huffman coding example.

Huffman Coding (contd).

- If the number of symbols in the code alphabet is r, then there must be $n = r + k(r-1)$ source alphabet symbols (k integer) in the Huffman code.
- This is because each stage of the Huffman coding algorithm reduces the size of the source alphabet by $r-1$ and there must be r symbols in the final stage to merge to form the root of the coding tree.
- If there are less source alphabet symbols, then one must add source symbols (with probability zero), until $n = r + k(r-1)$ for some integer k.

A	1
B	00
C	01
D	02
E	20
F	21
G	22

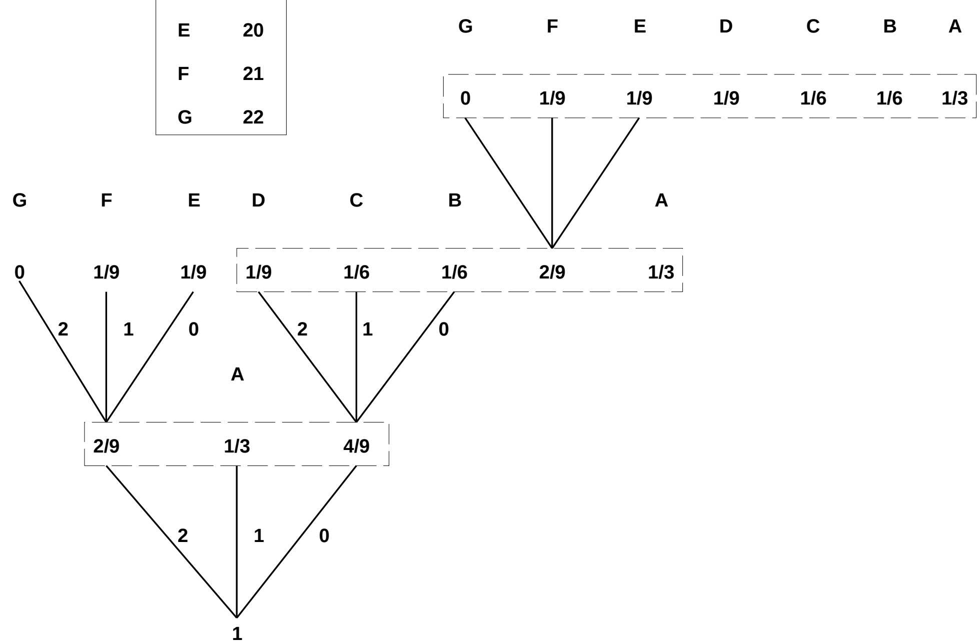


Figure 7: Huffman coding example (3 symbol code alphabet). Note the addition of a source symbol with probability zero.