**P1.** Design two Huffman codes for a source that put letters from a set A = {*a*1, *a*2, *a*3, *a*4, *a*5, *a*6, *a*7} with P(*a*1) = P(*a*3) = 0.2, P(*a*2) = 0.3, P(*a*4) = P(*a*5) = 0.1, P(*a*6) = P(*a*7) = 0.05.

**1.**

Table 1: Probabilities of Letters

|  |  |  |
| --- | --- | --- |
| Letter | Probability | codeword |
| *a*2 | 0.3 | C*(a*2) |
| *a*1 | 0.2 | C*(a*1) |
| *a*3 | 0.2 | C*(a*3) |
| *a*4 | 0.1 | C*(a*4) |
| *a*5 | 0.1 | C*(a*5) |
| *a*6 | 0.05 | C*(a*6) |
| *a*7 | 0.05 | C*(a*7) |

The encoding procedures are illustrated through the following table based on two observations on Huffman codes.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *a*2(0.3) | *a*2(0.3) | *a*2(0.3) | *a*2(0.3)  10  1  00 | *a*1’(0.4)  10  00  1 | *a*2’(0.6)  00 |
| *a*1(0.2) | *a*1(0.2) | *a*1(0.2)  1  00 | *a*5”(0.3) | *a*2(0.3) | *a*1’(0.4)  10 |
| *a*3(0.2) | *a*3(0.2) | *a*3(0.2) | *a*1(0.2) | *a*5”(0.3) |  |
| *a*4(0.1) | *a*4(0.1)  00  1 | *a*5’(0.2) | *a*3(0.2) |  |  |
| *a*5(0.1)  00  1 | *a*5(0.1) | *a*4(0.1) |  |  |  |
| *a*6(0.05) | *a*6’(0.1) |  |  |  |  |
| *a*7(0.05) |  |  |  |  |  |

Table 2: Procedures of Designing Huffman Codes

Therefore the Huffman codes of *a*1 to *a*7 are given as follows:

Table 3: Huffman Codes of Letters

|  |  |  |
| --- | --- | --- |
| Letter | Probability | codeword |
| *a*2 | 0.3 | 00 |
| *a*1 | 0.2 | 10 |
| *a*3 | 0.2 | 11 |
| *a*4 | 0.1 | 011 |
| *a*5 | 0.1 | 0100 |
| *a*6 | 0.05 | 01010 |
| *a*7 | 0.05 | 01011 |

The average length for this code is

L1 = 0.3\*2 + 0.2\*2 + 0.2\*2 + 0.1\*3 + 0.1\*4 + 0.05\*5 + 0.05\*5 = 2.6 bits/symbol

**2.** Minimum variance Huffman code

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *a*2(0.3) | *a*2(0.3) | *a*2(0.3) | *a*2(0.3)  10  1  00 | *a*1’(0.4)  10  00  1 | *a*2’(0.6)  00 |
| *a*1(0.2) | *a*1(0.2) | *a*1(0.2)  1  00 | *a*4”(0.3) | *a*2(0.3) | *a*1’(0.4)  10 |
| *a*3(0.2) | *a*3(0.2)  10 | *a*3(0.2) | *a*1(0.2) | *a*4”(0.3) |  |
| *a*4(0.1)  1  00 | *a*6’(0.1)  00  1 | *a*4’(0.2) | *a*3(0.2) |  |  |
| *a*5(0.1) | *a*4(0.1) | *a*6’(0.1) |  |  |  |
| *a*6(0.05) | *a*5(0.1) |  |  |  |  |
| *a*7(0.05) |  |  |  |  |  |

Table 4: Another Procedures of Designing Huffman Codes

From the above table, we can obtain another Huffman code below.

Table 5: Another Huffman Code with Shorter Variance of Codeword Length

|  |  |  |
| --- | --- | --- |
| Letter | Probability | codeword |
| *a*2 | 0.3 | 00 |
| *a*1 | 0.2 | 10 |
| *a*3 | 0.2 | 11 |
| *a*4 | 0.1 | 0100 |
| *a*5 | 0.1 | 0101 |
| *a*6 | 0.05 | 0110 |
| *a*7 | 0.05 | 0111 |

The corresponding average codeword length is

L2 = 0.3\*2 + 0.2\*2 + 0.2\*2 + 0.1\*4+ 0.1\*4 + 0.05\*4 + 0.05\*4 = 2.6 bits/symbol

**P2.** Encode the sequence (*a*1*a*2*a*1*a*3*a*1) using the floating-point number implementation. The corresponding probabilities are: P(*a*1) = 0.8, P(*a*2) = 0.02, P(*a*3) = 0.18.

From the probability model, we know that

F*x*(0) = 0, F*x*(1) = 0.8, F*x*(2) = 0.82, F*x*(3) = 1

Let *u*(0) = 1 and *l*(0) = 0.

We first calculate

*l*(1) (1) = *l*(0) + (*u*(0) - *l*(0)) F*x*(0) = 0 + (1 – 0)0 = 0

*u*(1) (1) = *l*(0) + (*u*(0) - *l*(0)) F*x*(1) = 0 + (1 – 0)0.8 = 0.8.

The interval [0, 0.8) is not either in the upper or lower haft of [0, 1). The second element of the sequence is 2. Therefore, we have

*l*(2) (12) = *l*(1) + (*u*(1) - *l*(1)) F*x*(1) = 0 + (0.8 – 0)0.8 = 0.64

*u*(2) (12) = *l*(1) + (*u*(1) - *l*(1)) F*x*(2) = 0 + (0.8 – 0)0.82 = 0.656.

The interval [0.64, 0.656) is in the upper haft [0, 1), so we have the binary code “1” and rescale *l*(2) (12) and *u*(2) (12) as follows:

*l*(2) (12) = 2\*(0.64-0.5) = 0.28

*u*(2) (12) = 2\*(0.656-0.5) = 0.312.

The 3rd element is 1 (*a*1) and we have

*l*(3) (121) = *l*(2) + (*u*(2) - *l*(2)) F*x*(0) = 0.28 + (0.312 – 0.28)0 = 0.28

*u*(3) (121) = *l*(1) + (*u*(2) - *l*(2)) F*x*(1) = 0.28 + (0.312 – 0.28)0.8 = 0.3056.

Since [0.28, 0.3056) is in [0, 0.5), we encode a “0” and the code is “10” now. *l*(3) and *u*(3) are rescaled as:

*l*(3) = 2\*0.28 = 0.56

*u*(3) = 2\*0.3056 = 0.6112.

The interval [0.56, 0.6112) is in the upper haft [0, 1), so we encode a “1” and the code is “101” now and rescale *l*(3) (121) and *u*(3) (121) as follows:

*l*(3) (121) = 2\*(0.56-0.5) = 0.12

*u*(3) (121) = 2\*(0.6112-0.5) = 0.2224.

The 4th element is 3 (*a*3) and we have

*l*(4) (1213) = *l*(3) + (*u*(3) - *l*(3)) F*x*(2) = 0.12 + (0.2224 – 0.12)0.82 = 0.203968

*u*(4) (1213) = *l*(3) + (*u*(3) - *l*(3)) F*x*(3) = 0.12 + (0.2224 – 0.12)1 = 0.2224.

Since [0.203968, 0.2224) is in [0, 0.5), we encode a “0” and the code is “1010” now. *l*(4) and *u*(4) are rescaled as:

*l*(4) = 2\*0.203968= 0.407936

*u*(4) = 2\*0.2224 = 0.4448.

Since [0.407936, 0.4448) is in [0, 0.5), we encode a “0” and the code is “10100” now. *l*(4) and *u*(4) are rescaled as:

*l*(4) = 2\*0.407936= 0.815872

*u*(4) = 2\*0.4448 = 0.8896.

The interval [0.815872, 0.8896) is in the upper haft [0, 1), so we encode a “1” and the code is “101001” now and rescale *l*(4) (1213) and *u*(4) (1213) as follows:

*l*(4) (1213) = 2\*(0.815872-0.5) = 0.631744

*u*(4) (1213) = 2\*(0.8896-0.5) = 0.7792.

The 5th element is 1 (*a*1) and we have

*l*(5) (12131) = *l*(4) + (*u*(4) - *l*(4)) F*x*(0) = 0.63174 + (0.7792 – 0.63174)0 = 0.63174

*u*(5) (12131) = *l*(4) + (*u*(4) - *l*(4)) F*x*(1) = 0.63174 + (0.7792 – 0.63174)0.8 = 0.749708.

The interval [0.63174, 0.749708) is in the upper haft [0, 1), so we encode a “1” and the code is “1010011”.