DATA COMPRESSION

HOMEWORK 1

STUDENT ID:

108368017

STUDENT:

ZI-YANG LIN

ADVISOR:

CHIU-CHING TUAN

National Taipei University of Technology

2019

Problem 1

Let X be a random variable with an alphabet $H = \{1,2,3,4,5\}$. Please determine $H(X)$ for the following three cases of probability mass function $p(i) = prob[X = i]$. (15%)

(a) $p(1) = p(2) = \frac{1}{2}$

Ans:

$$
\begin{aligned}
H(X) &= -(\frac{1}{2})\log_2(\frac{1}{2}) + \frac{1}{2}\log_2(\frac{1}{2})) \\
&= -(-\frac{1}{2} - \frac{1}{2}) \\
&= 1\ bits/symbol
\end{aligned}
$$

(b) $p(i) = \frac{1}{4}, for\ i = 1,2,3,\ and\ p(4) = p(5) = \frac{1}{8}$

Ans:

$$
\begin{aligned}
H(X) &= -(3 \times \frac{1}{4}\log_2(\frac{1}{4}) + 2 \times \frac{1}{4}\log_2(\frac{1}{4})) \\
&= -(-1.5 - 0.75) \\
&= 2.25\ bits/symbol
\end{aligned}
$$

(c) $P(i) = 2^{-i}, for\ i = 1,2,3,4, and\ p(5) = \frac{1}{16}$

Ans:

$$
\begin{aligned}
H(X) &= -(\sum_{i=1}^{4} 2^{-i}\log_2 2^{-i} + \frac{1}{16}\log_2\frac{1}{16}) \\
&= -(0.5 \times (-1) + 0.25 \times (-2) + 0.125 \times (-3) + 0.0625 \times (-4) + 0.0625 \times (-4)) \\
&= 1.875\ bits/symbol
\end{aligned}
$$

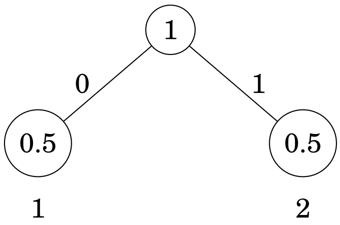
Problem 2

Design a Huffman code C for the source in Problem 1.

(a) Specify your codewords for individual pmf model in Problem 1.

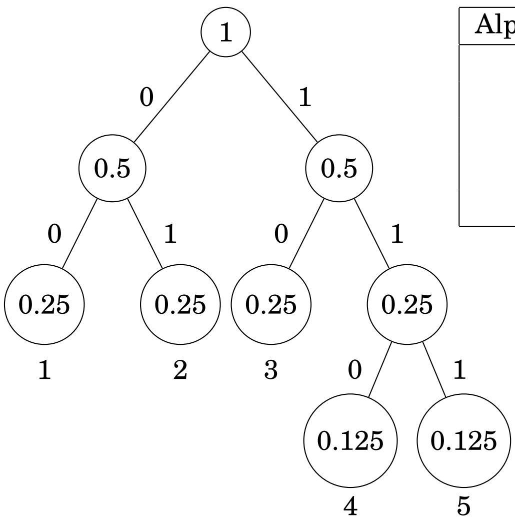
Ans:

??



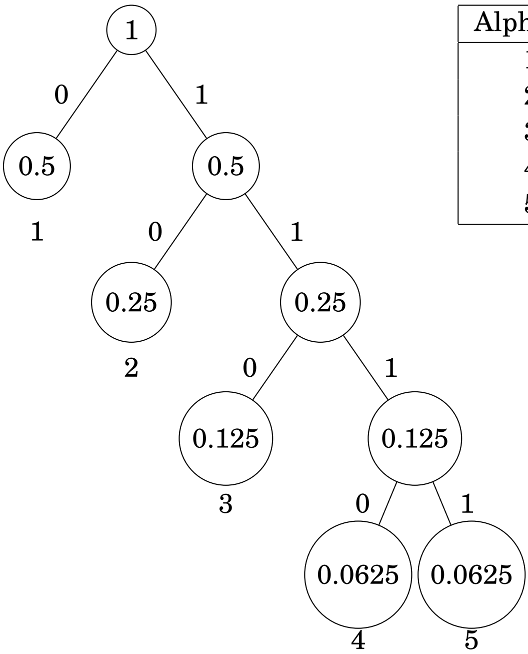
Alphabet	P	Codeword
1	0.5	0
2	0.5	1

??



Alphabet	P	Codeword
1	0.25	00
2	0.25	01
3	0.25	10
4	0.125	110
5	0.125	111

??



Alphabet	P	Codeword
1	0.5	0
2	0.25	10
3	0.125	110
4	0.0625	1110
5	0.0625	1111

(b) Compute the expected codeword length and compare with the entropy for your codes in (a).

Ans:

??

$$\begin{aligned} expected\ codeword\ length &= 0.5 \times 1 + 0.5 \times 1 \\ &= 1\ bits/symbol\ \textbf{(Equal Entropy)} \end{aligned}$$

??

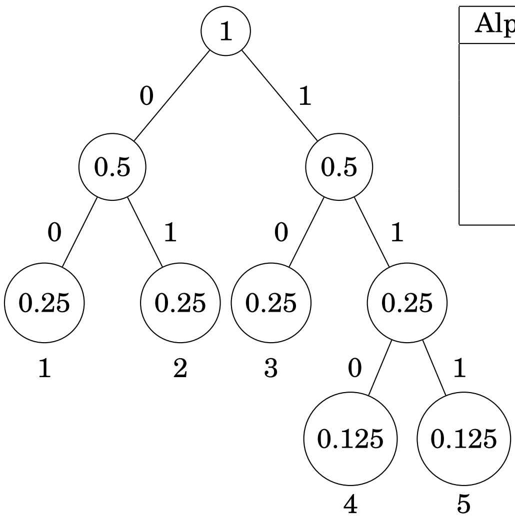
$$\begin{aligned} expected\ codeword\ length &= 0.25 \times 2 + 0.25 \times 2 + 0.25 \times 2 + 0.125 \times 3 + 0.125 \times 3 \\ &= 2.25\ bits/symbol\ \textbf{(Equal Entropy)} \end{aligned}$$

??

$$\begin{aligned} expected\ codeword\ length &= 0.5 \times 1 + 0.25 \times 2 + 0.125 \times 3 + 0.0626 \times 4 + 0.0625 \times 4 \\ &= 4.125\ bits/symbol\ \textbf{(NOT Equal Entropy)} \end{aligned}$$

(c) Design a code with minimum codeword length variance for the pmf model in Problem 1.(b)

Ans:



Alphabet	P	Codeword
1	0.25	00
2	0.25	01
3	0.25	10
4	0.125	110
5	0.125	111

Problem 3

Empirical distribution. In the case a probability model is not known, it can be estimated from empirical data. Lets say the alphabet is $H = \{1, 2, 3, \ldots, m\}$. Given a set of observations of length N, the empirical distribution is given by $p = total\ number\ of\ symbol\ 1/N,\ for\ i = 1, 2, 3, ..., m$. Please determine the empirical distribution for **santaclaus.txt**, which is an ASCII file with only lower-cased English letters (i.e., $a \sim z$), space and CR (carriage return), totally 28 symbols. The file can be found on the class web site. Compute the entropy.

Ans:

The source code for this problem are available at
`https://github.com/Yang92047111/2019_Data_Compression.git`.
After I executed the program the entropy is 4.121 bits/symbol.

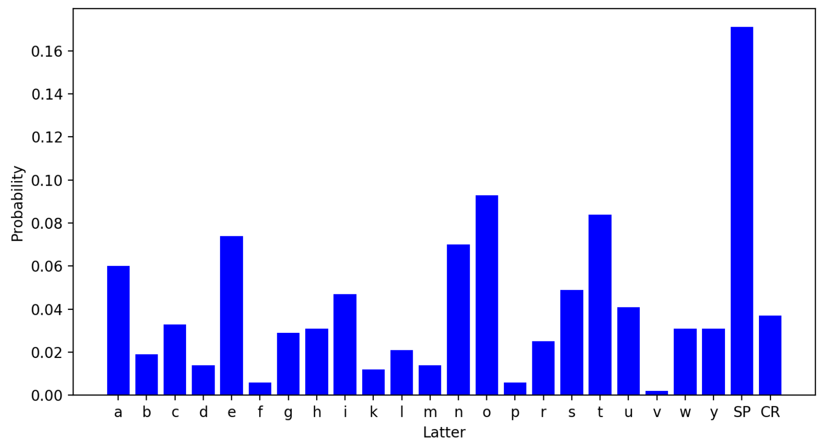


Figure 1: Empirical distribution for **santaclaus.txt**

Problem 4

Write a program that designs a Huffman code for the given distribution in Problem 3.

Ans:

```
Letter: a   Encoding Codeword: 0111
Letter: b   Encoding Codeword: 110010
Letter: c   Encoding Codeword: 10011
Letter: d   Encoding Codeword: 010111
Letter: e   Encoding Codeword: 1011
Letter: f   Encoding Codeword: 01101111
Letter: g   Encoding Codeword: 01100
Letter: h   Encoding Codeword: 10000
Letter: i   Encoding Codeword: 0001
Letter: k   Encoding Codeword: 010110
Letter: l   Encoding Codeword: 110011
Letter: m   Encoding Codeword: 011010
Letter: n   Encoding Codeword: 1010
Letter: o   Encoding Codeword: 001
Letter: p   Encoding Codeword: 0110110
Letter: r   Encoding Codeword: 01010
Letter: s   Encoding Codeword: 0100
Letter: t   Encoding Codeword: 1101
Letter: u   Encoding Codeword: 0000
Letter: v   Encoding Codeword: 01101110
Letter: w   Encoding Codeword: 10001
Letter: y   Encoding Codeword: 10010
Letter: SP   Encoding Codeword: 111
Letter: CR   Encoding Codeword: 11000
```

Figure 2: Huffman encode result for **santaclaus.txt**

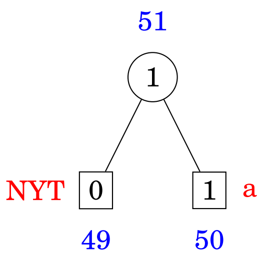
Problem 5

Let X be a random variable with an alphabet H, i.e., the 26 lower-case letters. Use adaptive Huffman tree to find the binary code for the sequence
a a b b a.
You are asked to use the following 5 bits fixed-length binary code as the initial codewords for the 26 letters. That is
a: 00000
b: 00001
⋮
z: 11001
Note: Show the Huffman tree during your coding process.

Ans:

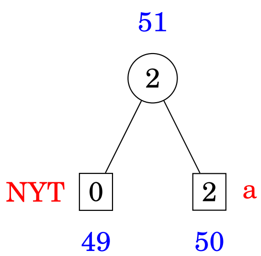
1. Initial step:
   $Total\ nodes = 2m - 1 = 26 \times 2 - 1 = 51$

2. **a** encoded:



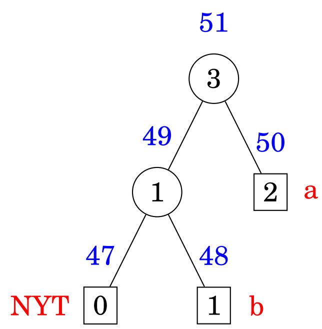
00000
a

3. **a a** encoded:



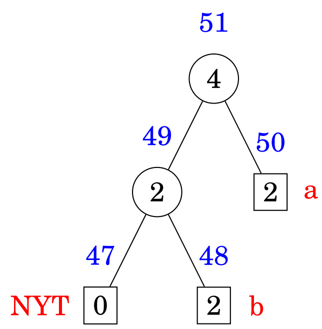
00000 1
a **a**

4. **a a b** encoded:



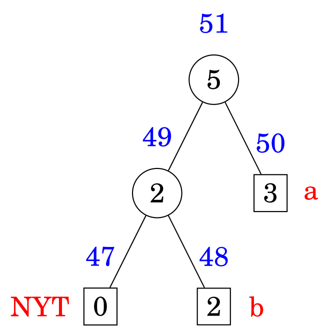
00000	1	0	00001
a	**a**	**NYT**	**b**

5. **a a b b** encoded:



00000	1	0	00001	01
a	**a**	**NYT**	**b**	**b**

6. **a a b b a** encoded:



00000	1	0	00001	01	1
a	**a**	**NYT**	**b**	**b**	**a**

Problem 6

(a) Find the Golomb code of n=21 when m=4.

Ans:

$$2^{\lceil \log_2^m \rceil} - m = 2^2 - 4 = 0$$
$$encoding\ 21 = 21 \div 4 = 5 \ldots 1 = 111110\ 01$$

(b) Find the Golomb code of n=14 when m=4.

Ans:

$$2^{\lceil \log_2^m \rceil} - m = 2^2 - 4 = 0$$
$$encoding\ 14 = 14 \div 4 = 3 \ldots 2 = 1110\ 10$$

(c) Find the Golomb code of n=21 when m=5.

Ans:

$$2^{\lceil \log_2^m \rceil} - m = 2^3 - 5 = 3$$
$$encoding\ 21 = 21 \div 5 = 2 \ldots 1 = 110\ 01$$

(d) Find the Golomb code of n=14 when m=5.

Ans:

$$2^{\lceil \log_2^m \rceil} - m = 2^3 - 5 = 3$$
$$encoding\ 14 = 14 \div 5 = 2 \ldots 4 = 110\ 111$$

(e) A two-integer sequence is encoded by Golomb code with m=4 to get the bitstream 11101111000. Whats the decoded two-integer sequence?

Ans:

$$2^{\lceil \log_2^m \rceil} - m = 2^2 - 4 = 0$$
$$\begin{array}{cccc} \underline{1110} & \underline{11} & \underline{110} & \underline{00} \\ 3 & 3 & 2 & 0 \\ \multicolumn{2}{c}{15} & \multicolumn{2}{c}{8} \end{array}$$
$$\text{sequence: } 15,\ 8$$

(f) A two-integer sequence is encoded by Golomb code with m=5 to get the bitstream 11101111000 (the same bitstream as that in (e)). Whats the decoded two-integer sequence?
Hint: The unary code for a positive integer q is simply q 1s followed by a 0.

Ans:

$$2^{\lceil \log_2^m \rceil} - m = 2^3 - 5 = 3$$

$$\begin{array}{cccc} \underline{1110} & \underline{111} & \underline{10} & \underline{00} \\ 3 & 7\text{-}3 = 4 & 1 & 0 \\ \multicolumn{2}{c}{19} & \multicolumn{2}{c}{5} \end{array}$$

sequence: 19, 5

Source code for Problem 3 & 4

huffman.py

```python
# node structure
class Node:
    def __init__(self, freq):
        self.left = None
        self.right = None
        self.father = None
        self.freq = freq
    def isLeft(self):
        return self.father.left == self

# create nodes
def createNodes(freqs):
    return [Node(freq) for freq in freqs]

# create Huffman-Tree
def createHuffmanTree(nodes):
    queue = nodes[:]
    while len(queue) > 1:
        queue.sort(key = lambda item : item.freq)
        node_left = queue.pop(0)
        node_right = queue.pop(0)
        node_father = Node(node_left.freq + node_right.freq)
        node_father.left = node_left
        node_father.right = node_right
        node_left.father = node_father
        node_right.father = node_father
        queue.append(node_father)

    queue[0].father = None
    return queue[0]


# Huffman coding
def huffmanEncoding(nodes, root):
    codes = [''] * len(nodes)
    for i in range(len(nodes)):
        node_tmp = nodes[i]
        while node_tmp != root:
            if node_tmp.isLeft():
                codes[i] = '0' + codes[i]
            else:
```

```
                codes[i] = '1' + codes[i]
            node_tmp = node_tmp.father

    return codes
```

main.py

```
from matplotlib import pyplot as plt
from collections import Counter
import math
import string
import huffman

# calculate probability
def Probability(freq):
    return round((freq / TotalLetter), 3)

# calculate entropy
def Entropy(pr):
    entropy = sum([p * math.log2(1 / p) for p in pr])
    return round(entropy, 3)

# read .txt file
with open("santaclaus.txt") as tf:
    letter = tf.read()

LetterCount = Counter(letter.translate(str.maketrans('', '', string.punctuatio
LetterCount = {k : LetterCount.get(k, 0) for k in string.printable}
LetterCount = {x : y for x, y in LetterCount.items() if y != 0}
TotalLetter = sum(LetterCount.values())

# calculate pmf
pmf = []
y = list(LetterCount.values())
for value in y:
    pmf.append(Probability(value))

print(Entropy(pmf), 'bits/symbol')

x = list(LetterCount.keys())
x[22] = 'SP'
x[23] = 'CR'

# Huffman Encoding
chars_freqs = list(zip(x, y))
```

```
nodes = huffman.createNodes([item[1] for item in chars_freqs])
root = huffman.createHuffmanTree(nodes)
codes = huffman.huffmanEncoding(nodes, root)
print('codeword:')
for item in zip(chars_freqs, codes):
    print ('Letter: %s   Encoding Codeword: %s' % (item[0][0], item[1]))

plt.figure(1)
plt.xlabel('Letter')
plt.ylabel('Frequency')
plt.bar(x, y, color='b')

plt.figure(2)
plt.xlabel('Latter')
plt.ylabel('Probability')
plt.bar(x, pmf, color='b')

plt.show()
```