ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

# ΘΕΩΡΙΑ ΠΛΗΡΟΦΟΡΙΑΣ

## $4^{η}$ ΣΕΙΡΑ ΓΡΑΠΤΩΝ ΑΣΚΗΣΕΩΝ

Ειρήνη Δόντη

ΑΜ: 03119839

9ο εξάμηνο

Αθήνα 2023-2024

Για την επίλυση των μαθηματικών σχέσεων της εκάστοτε άσκησης χρησιμοποιούμε κώδικα σε Python.

## <u>**Άσκηση 1**</u>

1. Η εντροπία πηγής είναι ίση με: H(X) = $-\sum_{i=1}^{5}\Pr\{X=i\}\log_2(\Pr\{X=i\})$ = 1.9464 bits σύμφωνα με τον κώδικα που επισυνάπτεται.

2. (a) Το μέσο μήκος κώδικα L(C) υπολογίζεται ως: L(C) = $\sum_{x\epsilon X}\Pr\{X=x\}$l(x) = 2.0000 σύμφωνα με τον κώδικα που επισυνάπτεται.

   (b) Ο κώδικας είναι μοναδικά αποκωδικοποιήσιμος, καθώς καμία λέξη κώδικα δεν είναι πρόθεμα άλλης λέξης κώδικα.

3.

   (a) Το αλφάβητο κώδικα αποτελείται από δύο σύμβολα {0,1}. Οπότε, για να εξετάσουμε αν ισχύει η ανισότητα Kraft ελέγχουμε αν: $\sum_{x\epsilon X} D^{-l(x)} \leq 1$, $\sum_{x\epsilon X} 2^{-l(x)} \leq 1$, 0.6250 ≤ 1. Στην προκειμένη περίπτωση, ικανοποιείται η ανισότητα Kraft.

   (b) Ο κώδικας δεν είναι μοναδικά αποκωδικοποιήσιμος, καθώς η λέξη κώδικα «10» είναι πρόθεμα της λέξης κώδικα «100».

   (c) Εφόσον ο κώδικας δεν είναι μοναδικά αποκωδικοποιήσιμος, όπως εξηγήσαμε παραπάνω, δεν είναι και στιγμιαίος.

   (d) Το μέσο μήκος κώδικα L(C) υπολογίζεται ως: L(C) = $\sum_{x\epsilon X}\Pr\{X=x\}$l(x) = 3.0000 σύμφωνα με τον κώδικα που επισυνάπτεται. Το μέσο μήκος κώδικα μπορεί να βελτιωθεί με τη βοήθεια της συμπίεσης δεδομένων. Στην προκειμένη περίπτωση το μέσο μήκος κώδικα δεν μπορεί να βελτιωθεί, καθώς δεν είναι μοναδικά αποκωδικοποιήσιμος και συνεπώς δεν υπάρχει τρόπος να αποκωδικοποιήσουμε τον κώδικα χωρίς ασάφειες.

4.

(a) Το αλφάβητο κώδικα αποτελείται από δύο σύμβολα $\{0,1\}$. Οπότε, για να εξετάσουμε αν ισχύει η ανισότητα Kraft ελέγχουμε αν: $\sum_{x\epsilon X} D^{-l(x)} \leq 1$,

$\sum_{x\epsilon X} 2^{-l(x)} \leq 1$, $0.9687 \leq 1$. Στην προκειμένη περίπτωση, ικανοποιείται η ανισότητα Kraft.

(b) Ο κώδικας είναι μοναδικά αποκωδικοποιήσιμος, καθώς καμία λέξη κώδικα δεν είναι πρόθεμα άλλης λέξης κώδικα.

(c) Ο κώδικας δεν είναι στιγμιαίος, καθώς η λέξη κώδικα «1» είναι πρόθεμα της λέξης κώδικα «01».

(d) Το μέσο μήκος κώδικα L(C) υπολογίζεται ως: L(C) = $\sum_{x\epsilon X} \Pr\{X = x\} l(x)$ = 2.0500 σύμφωνα με τον κώδικα που επισυνάπτεται. Το μέσο μήκος κώδικα μπορεί να βελτιωθεί με τη βοήθεια της συμπίεσης δεδομένων. Στην προκειμένη περίπτωση το μέσο μήκος κώδικα μπορεί να βελτιωθεί, καθώς είναι μοναδικά αποκωδικοποιήσιμος και συνεπώς υπάρχει τρόπος να αποκωδικοποιήσουμε τον κώδικα χωρίς ασάφειες.

5. (a) Ο κώδικας δεν είναι μοναδικά αποκωδικοποιήσιμος, καθώς η λέξη κώδικα «01» είναι πρόθεμα της λέξης κώδικα «0010».

(b) Εφόσον ο κώδικας δεν είναι μοναδικά αποκωδικοποιήσιμος, όπως εξηγήσαμε παραπάνω, δεν είναι και στιγμιαίος.

(c) Ο κώδικας δεν είναι βέλτιστος, καθώς υπάρχουν κώδικες που μπορούν να συμπιεστούν σε λιγότερα bits.

(d) Ο κώδικας δεν είναι κώδικας Huffman, καθώς η λέξη κώδικα «01» είναι πρόθεμα της λέξης κώδικα «0010».

## <u>**Άσκηση 2**</u>

1. Το αλφάβητο κώδικα αποτελείται από δύο σύμβολα {0,1}. Οπότε, για να εξετάσουμε αν ισχύει η ανισότητα Kraft ελέγχουμε αν: $\sum_{x \epsilon X} D^{-l(x)} \leq 1$, $\sum_{x \epsilon X} 2^{-l(x)} \leq 1$, $0.8750 \leq 1$.

2. Οι τροποποιημένες κωδικές λέξεις ώστε να μειωθεί το μέσο μήκος λέξης και να ικανοποιείται η ανισότητα Kraft είναι: 0, 10, 1111, 11110, 11101, 11111. Εξετάζουμε αν ισχύει η ανισότητα Kraft ελέγχοντας αν: $\sum_{x \epsilon X} D^{-l(x)} \leq 1$, $\sum_{x \epsilon X} 2^{-l(x)} \leq 1$, $0.90625 \leq 1$. Επιβεβαιώνουμε ότι στην προκειμένη περίπτωση, ικανοποιείται η ανισότητα Kraft.

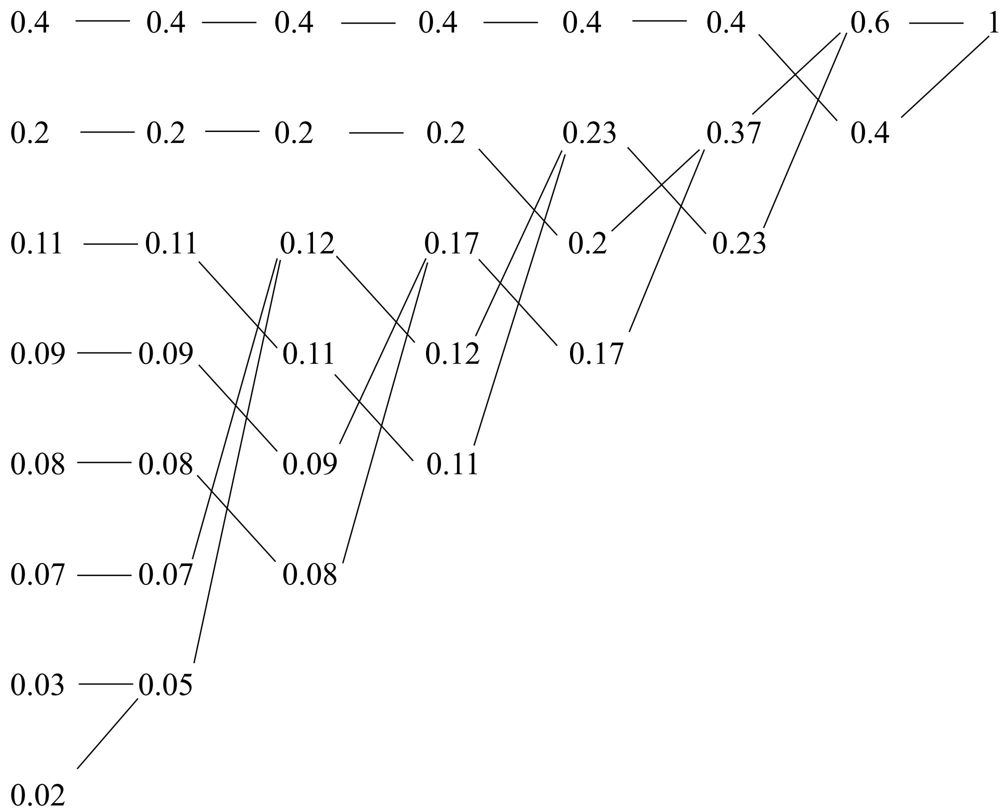
3. Με τη βοήθεια του μαθηματικού εργαλείου, επιβεβαιώνουμε ότι ικανοποιείται η ανισότητα Kraft με τιμή $\sum_{x \epsilon X} 2^{-l(x)}$ μεγαλύτερη από το πρώτο ερώτημα και βάσει αυτού, μπορούμε να πούμε ότι, οι κωδικές λέξεις έχουν μικρότερο μέσο μήκος από τους αρχικές κωδικές λέξεις.

4. Ο κώδικας είναι κώδικας Huffman, καθώς καμία λέξη κώδικα δεν είναι πρόθεμα άλλης λέξης κώδικα.

## Άσκηση 3

1.

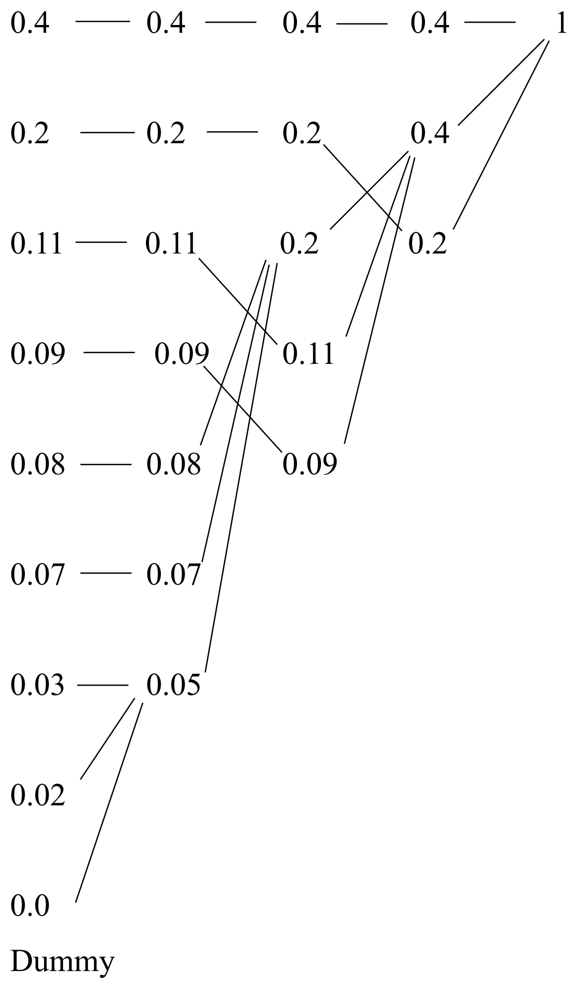
Η διαδικασία σχηματισμού του κώδικα Huffman παρουσιάζεται παρακάτω:




Ο κώδικας που προκύπτει είναι ο {1, 000, 011, 0010, 0011, 0100, 01010, 01011}.

Το μέσο μήκος κώδικα L(C) υπολογίζεται ως: L(C) = $\sum_{x\epsilon X} \Pr\{X = x\}$l(x) = 2.5400 σύμφωνα με τον κώδικα που επισυνάπτεται. Η εντροπία υπολογίζεται ως H(X) = $-\sum_{i=1}^{8} \Pr\{X = i\}\log_2(\Pr\{X = i\})$ = 2.4808 bits σύμφωνα με τον κώδικα που επισυνάπτεται. Παρατηρούμε ότι για το μέσο μήκος λέξης του κώδικα Huffman και για την εντροπία ισχύει ότι $H_D(X) \leq L^* \leq H_D(X) + 1$ (σχέση 5.140 βιβλίου).

2. Υπολογίζουμε τον βέλτιστο τριαδικό κώδικα.




Ο κώδικας που προκύπτει είναι ο {0, 2, 11, 12, 100, 101, 1020, 1021}.

Το μέσο μήκος κώδικα L(C) υπολογίζεται ως: L(C) = $\sum_{x \epsilon X} \Pr\{X = x\} l(x)$ = 1.6500 σύμφωνα με τον κώδικα που επισυνάπτεται. Η εντροπία υπολογίζεται ως H(X) = $-\sum_{i=1}^{8} \Pr\{X = i\} \log_3(\Pr\{X = i\})$ = 1.5652 bits σύμφωνα με τον κώδικα που επισυνάπτεται. Παρατηρούμε ότι για το μέσο μήκος λέξης του κώδικα Huffman και για την εντροπία ισχύει ότι $H_D(X) \leq L^* \leq H_D(X) + 1$ (σχέση 5.140 βιβλίου).

## Παράθεση Κώδικα σε Γλώσσα Προγραμματισμού Python

### Άσκηση 1

Είσοδος:

1.

```
from math import log

probs = [0.4, 0.3, 0.2, 0.05, 0.05]
H_x = round(-sum(i*log(i,2) for i in probs),4)
print(f"H(X) = {H_x} bits")
```

2.

```
#a

code = ["1", "01", "101", "0010", "0001"]
expected_len = round(sum(len(code[i])*probs[i] for i in
range(len(code))),4)
print(f"Expected length: {expected_len}")
```

3.

```
def kraft_inequality(code, alphabet_size):

    s = sum(alphabet_size**-len(i) for i in code)
    print(s)
    return s<1

#a
code = ["000", "001", "010", "011", "100"]

print(kraft_inequality(code, 2))

#d
```

```
expected_len = round(sum(len(code[i])*probs[i] for i in
range(len(code))),4)
print(f"Expected length of new code: {expected_len}")
```

4.

```
#a

code = ["1", "01", "000", "0011", "00101"]
def kraft_inequality(code, alphabet_size):
    s = sum(alphabet_size**-len(i) for i in code)
    print(s)
    return s<1

print(kraft_inequality(code, 2))

#d
expected_len = round(sum(len(code[i])*probs[i] for i in
range(len(code))),4)
print(f"Expected length of new code: {expected_len}")
```

5.

```
code = ["1", "01", "000", "0010", "0011"]

expected_len = round(sum(len(code[i])*probs[i] for i in
range(len(code))),4)
print(f"Expected length of new code: {expected_len}")
```

Έξοδος:

1.

```
H(X) = 1.9464 bits
```

2.

```
Expected length: 2.0
```

3.

```
0.625

True
Expected length of new code: 3.0
```

4.

```
0.96875

True
Expected length of new code: 2.05
```

5.

```
Expected length of new code: 2.0
```

## Άσκηση 2

Είσοδος:

1.

```
code = ["0", "10", "11001", "11110", "11100", "11111"]


def kraft_inequality(code, alphabet_size):
    s = sum(alphabet_size**-len(i) for i in code)
    print(s)
    return s<1

print(kraft_inequality(code, 2))
```

2.

```
code = ["0", "10", "1111", "11110", "11101","11111"]


def kraft_inequality(code, alphabet_size):
    s = sum(alphabet_size**-len(i) for i in code)
    print(s)
    return s<1

print(kraft_inequality(code, 2))
```

Έξοδος:

1.

```
0.875
True
```

2.

```
0.90625
True
```

**Άσκηση 3**

Είσοδος:

1.

```
from math import log

code = ["1", "000", "011", "0010", "0011", "0100", "01010",
"01011"]
probs = [0.4, 0.2, 0.11, 0.09, 0.08, 0.07, 0.03, 0.02]
expected_len = round(sum(len(code[i])*probs[i] for i in
range(len(code))),4)
print(f"Expected length of Huffman code: {expected_len}")
H_x = round(-sum(i*log(i,2) for i in probs),4)
print(f"H(X) = {H_x} bits")
```

2.

```
from math import log

code = ["0", "2", "11", "12", "100", "101", "1020", "1021"]
probs = [0.4, 0.2, 0.11, 0.09, 0.08, 0.07, 0.03, 0.02]
expected_len = round(sum(len(code[i])*probs[i] for i in
range(len(code))),4)
print(f"Expected length of Huffman code: {expected_len}")
H_x = round(-sum(i*log(i,3) for i in probs),4)
print(f"H(X) = {H_x} bits")
```

Έξοδος:

1.

```
Expected length of Huffman code: 2.54
H(X) = 2.4808 bits
```

2.

```
Expected length of Huffman code: 1.65
H(X) = 1.5652 bits
```