Univerzitet u Sarajevu Elektrotehnički fakultet

Predmet: Multimedijalni Sistemi

Laboratorijska vježba 03: Huffmanov algoritam kodiranja

Za izradu laboratorijske vježbe treba koristiti odgovarajuću Jupyter Notebook datoteku. Urađenu vježbu je potrebno konvertirati u PDF format, a zatim je PDF datoteku potrebno predati do postavljenog roka koristeći platformu Zamger.

Ime i prezime studenta, broj indeksa:

Amar Hasečić, 18673/2116

Datum izrade izvještaja:

26.3.2024.

Zadatak 1.

Potrebno je realizirati **Huffmanov algoritam** kodiranja implementacijom funkcije huffman koja kao parametar prima neku tekstualnu poruku, a kao rezultat vraća listu simbola i listu svih kodova dodijeljenih pojedinačnim simbolima. Možete koristiti funkcije kodiraj i entropija koje su implementirane u prethodnoj laboratorijskoj vježbi, kao i dijelove koda koji se mogu iskoristiti za ovaj algoritam (npr. izračunavanje vjerovatnoće pojavljivanja simbola u poruci).

Huffmanov algoritam se temelji na izgradnji stabla od "dna prema vrhu", za razliku od algoritma Shannon-Fano koji izgrađuje stablo od "vrha prema dnu". Koraci huffmanovog algoritma su:

1. Za zadanu listu simbola, izračunati vjerovatnoće pojavljivanja u poruci.
2. Sortirati simbole prema frekvenciji od najveće do najmanje frekvencije.
3. Iz liste uzeti dva elementa s najmanjom vjerovatnoćom pojavljivanja, jednom elementu dodati vrijednost 0, a drugom dodati vrijednost 1. Voditi računa da se dodavanje simbola vrši na suprotan način u odnosu na Shannon-Fano algoritam kodiranja.
4. Prethodno odabrane elemente dodati u zajedničko podstablo. Podstablo se tretira kao jedan element, a njegova vjerovatnoća pojavljivanja jednaka je sumi vjerovatnoća pojavljivanja simbola koji se nalaze u podstablu.
5. Ponovo sortirati listu simbola prema vjerovatnoći pojavljivanja.

6. Ponavljati korake 3, 4 i 5 dok u listi ne preostane samo jedan element.

Detaljne informacije o samom algoritmu moguće je pronaći u Poglavlju 2 materijala za rad na predmetu, na str. 25 - 29.

Dobivena rješenja za kodne riječi nisu jedinstvena, tj. dobivena rješenja za kodne riječi ovise o specifičnim implementacijskim detaljima. Bitno je da implementacija bude realizirana u skladu sa gore opisanim koracima, tj. u skladu sa algoritmom opisanim na predavanjima.

Rješenje:

```
def huffman(poruka):

    simboli = list(set(poruka))
    vjerovatnoce = [0]*len(simboli)
    kodovi = ['']*len(simboli)

    for i in range(len(simboli)):
        for simbol in poruka:
            if simboli[i] == simbol:
                vjerovatnoce[i]+=1
        vjerovatnoce[i]/=len(poruka)

    simbol_kod_p = []
    for i in range(len(simboli)):
        simbol_kod_p.append(([(simboli[i], kodovi[i])],
vjerovatnoce[i]))


    simbol_kod_p = sorted(simbol_kod_p, key=lambda x: x[0]) # po
alfabetu radi testova

    if len(simbol_kod_p) == 1:
        return [simboli[0]], ['0']

    while len(simbol_kod_p) != 1:
        #sortiranje po vjerovatnocama
        simbol_kod_p = sorted(simbol_kod_p, key=lambda x: x[1],
reverse=True)

        #zadnje dvije liste sa najmanjim vjerovatnocama
        predzadnji = simbol_kod_p[len(simbol_kod_p) - 2]
        zadnji = simbol_kod_p[len(simbol_kod_p) - 1]

        #dodavanje nule i jedinice na kodove simbola
        modified_predzadnji = [(kod[0], '0' + kod[1]) for kod in
predzadnji[0]]
        predzadnji = (modified_predzadnji, predzadnji[1])

        modified_zadnji = [(kod[0], '1' + kod[1]) for kod in
zadnji[0]]
```

```
        zadnji = (modified_zadnji, zadnji[1])


        #spajanje zadnje i predzadnje podliste i racunanje njihove
ukupne vjerovatnoce
        spojen_zadnji_predzadnji = (predzadnji[0] + zadnji[0],
predzadnji[1] + zadnji[1])

        #brisanje zadnjeg i predzadnjeg i dodavanje novog skupa
smbola, ovo bi bilo ono podstablo
        del simbol_kod_p[len(simbol_kod_p) - 1]
        del simbol_kod_p[len(simbol_kod_p) - 1]
        simbol_kod_p.append(spojen_zadnji_predzadnji)


    simboli = [pair[0] for pair in simbol_kod_p[0][0]]
    kodovi = [pair[1] for pair in simbol_kod_p[0][0]]

    return simboli, kodovi

def kodiraj(poruka, simboli, kodovi):
    kodirana = ""
    for simbol in poruka:
        kodirana+=kodovi[simboli.index(simbol)]
    return kodirana

import math
def entropija(poruka):

    simboli = list(set(poruka))
    vjerovatnoce = [0]*len(simboli)
    for i in range(len(simboli)):
        for simbol in poruka:
            if simboli[i] == simbol:
                vjerovatnoce[i]+=1
        vjerovatnoce[i]/=len(poruka)

    rezultat = 0

    for v in vjerovatnoce:
        rezultat += v * math.log(v,2)

    rezultat *= -1

    return abs(rezultat)
```

Nakon implementacije funkcije, potrebno je biti moguće izvršiti programski kod ispod tako da daje prikazani ispis. Osim toga, potrebno je pored poruke "BBBAAAAAAAABBCDAAA" dodati još četiri primjera poruka kako bi se implementacija testirala. Dodatna četiri primjera poruka trebaju obavezno biti iz sljedećeg skupa {"A", "DDDDDDDDDDDDDDDDD", "ABBBBCDECCCCAAAAAAAAAAAAA", "AAAAAAAAAABBBBBBCCCD"}.

```
import math
poruka = "BBBAAAAAAAABBCDAAA"

print("Izvorna poruka je:")
print(poruka)

(simboli, kodovi) = huffman(poruka)

print("\nPoruka se sastoji od sljedećih simbola:")
print(", ".join(map(str, simboli)))

print("\nSkup dobivenih kodnih riječi je:")
print(", ".join(map(str, kodovi)))

kodirana = kodiraj(poruka, simboli, kodovi)

for i in range(len(simboli)):
  print("\nKod simbola " + simboli[i] + " je " + kodovi [i])

print("\nKodirana poruka je:")
print(kodirana)
print("\nEntropija poruke je:")
print(round(entropija(poruka),3))
print("\nProsječna dužina kodne riječi u poruci je:")
print(round(len(kodirana)/len(poruka),3))
```

```
Izvorna poruka je:
BBBAAAAAAAABBCDAAA

Poruka se sastoji od sljedećih simbola:
A, B, C, D

Skup dobivenih kodnih riječi je:
0, 10, 110, 111

Kod simbola A je 0

Kod simbola B je 10

Kod simbola C je 110

Kod simbola D je 111

Kodirana poruka je:
101010000000001010110111000

Entropija poruke je:
1.411

Prosječna dužina kodne riječi u poruci je:
1.5
```

```
#Test 1
import math
poruka = "A"
print("Izvorna poruka je:")
print(poruka)

(simboli, kodovi) = huffman(poruka)

print("\nPoruka se sastoji od sljedecih simbola:")
print(", ".join(map(str, simboli)))

print("\nSkup dobivenih kodnih riječi je:")
print(", ".join(map(str, kodovi)))

kodirana = kodiraj(poruka, simboli, kodovi)

for i in range(len(simboli)):
  print("\nKod simbola " + simboli[i] + " je " + kodovi [i])

print("\nKodirana poruka je:")
print(kodirana)
print("\nEntropija poruke je:")
print(round(entropija(poruka),3))
print("\nProsječna dužina kodne riječi u poruci je:")
print(round(len(kodirana)/len(poruka),3))
```

```
Izvorna poruka je:
A

Poruka se sastoji od sljedecih simbola:
A

Skup dobivenih kodnih riječi je:
0

Kod simbola A je 0

Kodirana poruka je:
0

Entropija poruke je:
0.0

Prosječna dužina kodne riječi u poruci je:
1.0
```

```
#Test 2

import math
poruka = "DDDDDDDDDDDDDDDDDD"
print("Izvorna poruka je:")
```

```
print(poruka)

(simboli, kodovi) = huffman(poruka)

print("\nPoruka se sastoji od sljedecih simbola:")
print(", ".join(map(str, simboli)))

print("\nSkup dobivenih kodnih riječi je:")
print(", ".join(map(str, kodovi)))

kodirana = kodiraj(poruka, simboli, kodovi)

for i in range(len(simboli)):
  print("\nKod simbola " + simboli[i] + " je " + kodovi [i])

print("\nKodirana poruka je:")
print(kodirana)
print("\nEntropija poruke je:")
print(round(entropija(poruka),3))
print("\nProsječna dužina kodne riječi u poruci je:")
print(round(len(kodirana)/len(poruka),3))
```

```
Izvorna poruka je:
DDDDDDDDDDDDDDDDD

Poruka se sastoji od sljedecih simbola:
D

Skup dobivenih kodnih riječi je:
0

Kod simbola D je 0

Kodirana poruka je:
00000000000000000

Entropija poruke je:
0.0

Prosječna dužina kodne riječi u poruci je:
1.0
```

```
#Test 3

import math
poruka = "ABBBBCDECCCCAAAAAAAAAAAAAA"
print("Izvorna poruka je:")
print(poruka)

(simboli, kodovi) = huffman(poruka)
```

```
print("\nPoruka se sastoji od sljedecih simbola:")
print(", ".join(map(str, simboli)))

print("\nSkup dobivenih kodnih riječi je:")
print(", ".join(map(str, kodovi)))

kodirana = kodiraj(poruka, simboli, kodovi)

for i in range(len(simboli)):
  print("\nKod simbola " + simboli[i] + " je " + kodovi [i])

print("\nKodirana poruka je:")
print(kodirana)
print("\nEntropija poruke je:")
print(round(entropija(poruka),3))
print("\nProsječna dužina kodne riječi u poruci je:")
print(round(len(kodirana)/len(poruka),3))
```

```
Izvorna poruka je:
ABBBBCDECCCCAAAAAAAAAAAAAA

Poruka se sastoji od sljedecih simbola:
A, B, D, E, C

Skup dobivenih kodnih riječi je:
0, 100, 1010, 1011, 11

Kod simbola A je 0

Kod simbola B je 100

Kod simbola D je 1010

Kod simbola E je 1011

Kod simbola C je 11

Kodirana poruka je:
01001001001001110101011111111110000000000000

Entropija poruke je:
1.692

Prosječna dužina kodne riječi u poruci je:
1.731
```

```
#Test 4

import math
poruka = "AAAAAAAAAABBBBBBCCCD"
print("Izvorna poruka je:")
```

```
print(poruka)

(simboli, kodovi) = huffman(poruka)

print("\nPoruka se sastoji od sljedecih simbola:")
print(", ".join(map(str, simboli)))

print("\nSkup dobivenih kodnih riječi je:")
print(", ".join(map(str, kodovi)))

kodirana = kodiraj(poruka, simboli, kodovi)

for i in range(len(simboli)):
  print("\nKod simbola " + simboli[i] + " je " + kodovi [i])

print("\nKodirana poruka je:")
print(kodirana)
print("\nEntropija poruke je:")
print(round(entropija(poruka),3))
print("\nProsječna dužina kodne riječi u poruci je:")
print(round(len(kodirana)/len(poruka),3))
```

```
Izvorna poruka je:
AAAAAAAAAABBBBBBCCCD

Poruka se sastoji od sljedecih simbola:
A, B, C, D

Skup dobivenih kodnih riječi je:
0, 10, 110, 111

Kod simbola A je 0

Kod simbola B je 10

Kod simbola C je 110

Kod simbola D je 111

Kodirana poruka je:
00000000001010101010110110110111

Entropija poruke je:
1.648

Prosječna dužina kodne riječi u poruci je:
1.7
```