

Univerza v Ljubljani  
Fakulteta *za elektrotehniko*



UNIVERZA V LJUBLJANI  
FAKULTETA ZA ELEKTROTEHNIKO

# Gospodarno kodiranje po Huffmanovem postopku

Bor Starčič

mentorja:

asist. dr. Klemen Grm

izr. prof. dr. Simon Dobrišek

Maj 2024

# Kazalo

<b>1</b>	<b>Uvod</b>	<b>2</b>
<b>2</b>	<b>Naloga 1</b>	<b>2</b>
<b>3</b>	<b>Naloga 2</b>	<b>4</b>
3.1	Rezultati . . . . .	4
<b>4</b>	<b>Naloga 3</b>	<b>6</b>
4.1	Rezultati . . . . .	6
<b>5</b>	<b>Zaključek</b>	<b>8</b>

# 1 Uvod

V tej nalogi sem se lotil implementacije Huffmanovega kodiranja za diskretni informacijski vir. Huffmanovo kodiranje je učinkovita metoda za stiskanje podatkov, ki minimizira povprečno dolžino kode, uporabljene za predstavitev niza simbolov. Glavna prednost Huffmanovega kodiranja je njegova sposobnost ustvarjanja optimalnih kod z variabilno dolžino za dani nabor simbolov in njihovih pripadajočih verjetnosti, kar zagotavlja, da imajo najpogostejši simboli najkrajše kode.

## 2 Naloga 1

V tej nalogi sem obravnaval naslednji nabor simbolov z njihovimi pripadajočimi verjetnostmi:

$$S = (s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9) = (0.25, 0.20, 0.15, 0.10, 0.08, 0.07, 0.06, 0.05, 0.04)$$

### Analiza pogostosti

- Verjetnosti so podane kot del vhodnih podatkov.

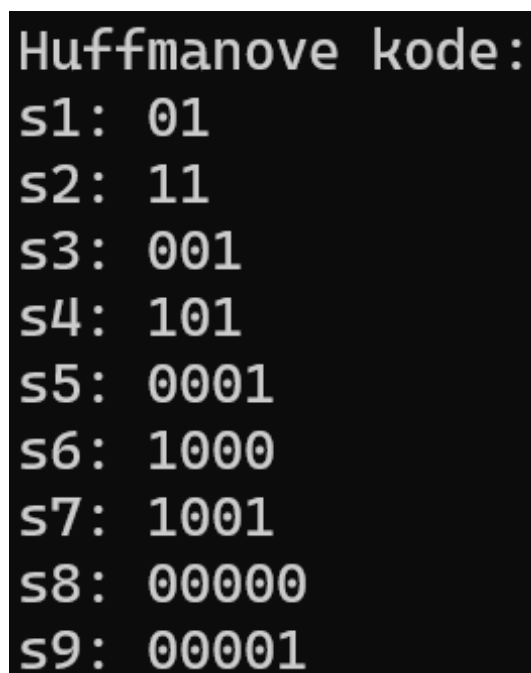
### Izgradnja Huffmanovega drevesa

- $s_9$  in  $s_8$ :  $(0.04 + 0.05 = 0.09)$
- $s_7$  in  $s_6$ :  $(0.06 + 0.07 = 0.13)$
- $0.09$  in  $s_5$ :  $(0.09 + 0.08 = 0.17)$
- $0.13$  in  $s_4$ :  $(0.13 + 0.10 = 0.23)$
- $0.17$  in  $s_3$ :  $(0.17 + 0.15 = 0.32)$
- $0.23$  in  $s_2$ :  $(0.23 + 0.20 = 0.43)$
- $0.32$  in  $s_1$ :  $(0.32 + 0.25 = 0.57)$
- $0.43$  in  $0.57$ :  $(0.43 + 0.57 = 1.00)$

Rezultat Huffmanovega drevesa ima koren s skupno frekvenco 1.00.

### Dodelitev kod

- Prehodil sem drevo, da sem dodelil kode:
  - $s_1$ : 01
  - $s_2$ : 11
  - $s_3$ : 001
  - $s_4$ : 101
  - $s_5$ : 0001
  - $s_6$ : 1000
  - $s_7$ : 1001
  - $s_8$ : 00000
  - $s_9$ : 00001

A screenshot of a terminal window with a black background and white text. The text displays the Huffman codes for nine symbols, labeled s1 through s9. The title 'Huffmanove kode:' is at the top, followed by the symbol and its corresponding binary code on separate lines.

```
Huffmanove kode:
s1: 01
s2: 11
s3: 001
s4: 101
s5: 0001
s6: 1000
s7: 1001
s8: 00000
s9: 00001
```

Slika 1: Izpis kode za 1. nalogo

## 3 Naloga 2

V drugi nalogi sem razširil program za binarno kodiranje po Huffmanovem postopku za poljubne datoteke. Namesto znakov sem obravnaval vse možne 8-bitne vzorce. Program izračuna verjetnosti pojavljanja vzorcev, ustvari Huffmanov kod in z njim zakodira vhodno datoteko v izhodno.

Izhodna datoteka s končnico .huf vsebuje tabelo kodnih zamenjav in zakodirane vzorce. Program izračuna učinkovitost kodiranja iz povprečnih dolžin kodnih zamenjav ter oceni učinkovitost glede na velikost kompresiranih datotek. Omogočeno je tudi dekodiranje kompresiranih datotek nazaj v izvirne.

Najprej izračunam verjetnosti pojavljanja vseh 8-bitnih vzorcev v vhodni datoteki. Na podlagi izračunanih verjetnosti program zgradi Huffmanovo drevo in dodeli kode vsakemu vzorcu. Zakodiramo vhodno datoteko z uporabo Huffmanovih kod in shrani rezultat v izhodno datoteko s končnico .huf. V glavi datoteke je shranjena tabela kodnih zamenjav. Omogočeno je tudi dekodiranje kompresiranih datotek nazaj v izvirne vhodne datoteke. Učinkovitost kodiranja izračunam glede na velikost kompresirane datoteke in povprečno dolžino kodnih zamenjav.

### 3.1 Rezultati

Na slikah so prikazani podatki o prvotni velikosti datoteke, velikosti po kompresiji, razmerju kompresije in preverjanju ujemanja MD5 med izvirno in dekomprimirano datoteko.

```
Prvotna velikost: 40000 bajtov  
Velikost po kompresiji: 14899 bajtov  
Razmerje kompresije: 2.68  
Ujemanje MD5: True
```

Slika 2: Izpis kode za 2. nalogo z datoteko .bin

```
Prvotna velikost: 400 bajtov  
Velikost po kompresiji: 349 bajtov  
Razmerje kompresije: 1.15  
Ujemanje MD5: True
```

Slika 3: Izpis kode za 2. nalogo z krajšo datoteko .bin

```
Prvotna velikost: 1038950 bajtov  
Velikost po kompresiji: 598907 bajtov  
Razmerje kompresije: 1.73  
Ujemanje MD5: True
```

Slika 4: Izpis kode za 2. nalogo z datoteko .txt

```
Prvotna velikost: 1442 bajtov  
Velikost po kompresiji: 1072 bajtov  
Razmerje kompresije: 1.35  
Ujemanje MD5: True
```

Slika 5: Izpis kode za 2. nalogo z krajšo datoteko .txt

Rezultati prikazujejo, da je Huffmanovo kodiranje zelo učinkovito pri stiskanju tako binarnih kot besedilnih datotek. Razmerje kompresije se razlikuje glede na vrsto datotek, vendar so v vseh primerih rezultati zadovoljivi. Velikost datoteke pa ni imela znatnega vpliva na uspešnost kompresije, razlika pri rezultatih najverjetneje nastaja zaradi drugačne pogostosti znakov v datotekah. Uspešnost kompresije potrjuje tudi dejstvo, da je preverjanje MD5 pokazalo popolno ujemanje med izvorno in dekomprimirano datoteko, kar pomeni, da je bil postopek stiskanja in dekompresije brez izgub.

## 4 Naloga 3

V tretji nalogi sem program dopolnil tako, da namesto posameznih bajtov vhodne datoteke kodira pare oz. trojice bajtov. Program ovrednoti razlike v entropijski uspešnosti in velikosti kompresiranih datotek pri uporabi zlogov dolžine večje od 1.

Najprej izračunam frekvence pojavitev vseh n-bajtnih vzorcev v vhodni datoteki. Program zakodira vhodno datoteko z uporabo Huffmanovih kod, ki temeljijo na n-bajtnih vzorcih, in shrani rezultat v izhodno datoteko s končnico ".huf". V glavi datoteke je shranjena tabela kodnih zamenjav. Omogočeno je tudi dekodiranje kompresiranih datotek nazaj v izvirne vhodne datoteke z uporabo n-bajtnih vzorcev.

### 4.1 Rezultati

Program izračuna in primerja učinkovitost kodiranja glede na dolžino n-bajtnih vzorcev. V tabeli so prikazani rezultati za različne dolžine zlogov, kjer se ocenjuje entropijska uspešnost in velikost kompresiranih datotek.

```
Prvotna velikost: 40000 bajtov  
Velikost po kompresiji: 12653 bajtov  
Razmerje kompresije: 3.16  
Ujemanje MD5: True
```

Slika 6: Uspešnost kompresije z pari bajtov pri datoteki .bin

```
Prvotna velikost: 400 bajtov  
Velikost po kompresiji: 397 bajtov  
Razmerje kompresije: 1.01  
Ujemanje MD5: True
```

Slika 7: Uspešnost kompresije z pari bajtov pri manjši datoteki .bin

```
Prvotna velikost: 40000 bajtov  
Velikost po kompresiji: 16045 bajtov  
Razmerje kompresije: 2.49  
Ujemanje MD5: False
```

Slika 8: Uspešnost kompresije z trojicami bajtov pri datoteki .bin

```
Prvotna velikost: 400 bajtov  
Velikost po kompresiji: 532 bajtov  
Razmerje kompresije: 0.75  
Ujemanje MD5: False
```

Slika 9: Uspešnost kompresije z trojicami bajtov pri manjši datoteki .bin

```
Prvotna velikost: 1442 bajtov  
Velikost po kompresiji: 2410 bajtov  
Razmerje kompresije: 0.60  
Ujemanje MD5: True
```

Slika 10: Uspešnost kompresije z trojicami bajtov pri manjši datoteki .txt

Razširitev Huffmanovega kodiranja na pare in trojice bajtov kaže, da se z večanjem dolžine zlogov kompresija lahko izboljša, vendar je odvisna od strukture podatkov v vhodni datoteki. Večji zlogi lahko bolje izkoristijo vzorce v podatkih, kar vodi do boljših razmerij kompresije. Primerjava rezultatov za različne dolžine zlogov potrjuje, da je izbira optimalne dolžine zloga pomembna za doseganje najboljše učinkovitosti kompresije. Pri kodiranju z trojicami bajtov se je pri nekaterih datotekah pojavil problem, kjer se dekodirana datoteka ni ujemala z originalno. MD5 se ni ujemal zaradi mankanja zadnjega znaka v dekodiranih datotekah ampak se to ni pojavilo pri vseh.



## 5 Zaključek

V tej nalogi smo implementirali Huffmanovo kodiranje in ga uporabili za stiskanje različnih vrst datotek. Prva naloga je pokazala, kako učinkovito je Huffmanovo kodiranje pri zmanjševanju velikosti podatkov s pomočjo optimalnih kod z variabilno dolžino. Druga naloga je razširila program za kodiranje poljubnih binarnih datotek, pri čemer so se kot simboli obravnavali 8-bitni vzorci. Rezultati so pokazali, da je Huffmanovo kodiranje zelo učinkovito pri stiskanju tako binarnih kot besedilnih datotek, pri čemer se je velikost datotek znatno zmanjšala, kar je potrdilo preverjanje MD5.

V tretji nalogi smo program dopolnili tako, da smo kodirali pare in trojice bajtov. Ugotovili smo, da daljši zlogi lahko izboljšajo razmerje kompresije, odvisno od strukture vhodnih podatkov. Večji zlogi bolje izkoristijo vzorce v podatkih, kar vodi do boljših razmerij kompresije. Primerjava različnih dolžin zlogov je potrdila, da je izbira optimalne dolžine zloga ključna za doseganje najboljše učinkovitosti kompresije.

Celotna naloga je pokazala, da je Huffmanovo kodiranje zelo uporabno orodje za stiskanje datotek, kjer se simboli pojavljajo v različnih pogostostih. Zagotavlja visoko učinkovitost in brezizgubno dekompresijo, kar potrjujejo tudi rezultati naših testov.