|  |  |
| --- | --- |
|  | |
| **Semestrální projekt č. 4** | |
|  | |
| David Papaj | |
|  | |
|  |  |
| Semestrální projekt  2021 |  |
|  |  |
|  | |

OBSAH

[zadání 3](#_Toc69075366)

[1 popis programu a jeho funkcí 4](#_Toc69075367)

[2 Huffmanova metoda 5](#_Toc69075368)

[2.1 Přidání f pro každý symbol 6](#_Toc69075369)

[2.1.1 Huffman zakódování +source code(formátováni textu) 6](#_Toc69075370)

[2.2 Vstup/Výstup pro zakódování 7](#_Toc69075371)

[2.3 Test daného vstupu – Toto je novy test pro testovani daného kodu 7](#_Toc69075372)

zadání

Vytvořte program (funkci) pro zakódování vstupního textového řetězce pomocí metod efektivního kódování. Využijte Shannon - Fanovu metodu a metodu Huffmanovu.

##### Podrobnosti k zadání:

* vstupem programu bude text (textový řetězec o délce minimálně 100 znaků včetně mezer)
* analýzou textu (spočítáním jednotlivých znaků) sestavíte tabulku (příklad viz níže) s pravděpodobnostmi výskytu jednotlivých znaků abecedy

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Znaky: | a1 | a2 | a3 | a4 | a5 | a6 | a7 | a8 | a9 | a10 | .... | an |
| Pravděpodobnosti: | 0.05 | 0.14 | 0.095 | 0.018 | 0.14 | 0.048 | 0.015 | 0.13 | 0.16 | 0.016 | .... | 0.01 |

* k jednotlivým znakům abecedy přiřadíte kódová slova dle metod efektivního kódování (Shannon-Fanova a Huffmanova metoda)
* výstupem programu pak bude binární řetězec, který bude představovat zakódovaný text dle jednotlivých metod
* výstup programu bude navíc obsahovat informaci o velikosti průměrné délku kódového slova a efektivnosti kódu
* zdrojový text programu by měl obsahovat komentáře popřípadě popis
* **výstupem semestrálního projektu bude textový dokument (zpráva o řešení), který bude zahrnovat následující části:**
  + **zadání semestrálního projektu**
  + **popis programu a jeho funkcí**
  + **na konkrétním příkladu ukázka jednotlivých výstupů programu (na konkrétním textu prezentujete výsledky dle obou metod, využijete výstupů vytvořeného programu)**
  + **zhodnocení výsledků a závěr**

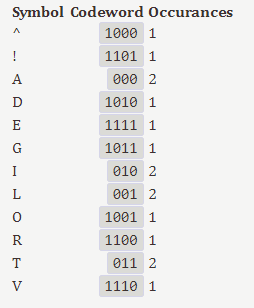
popis programu a jeho funkcí

Daný program zakóduje vstupní hodnotu (text) a zakóduje pomocí Huffmanovy metody.

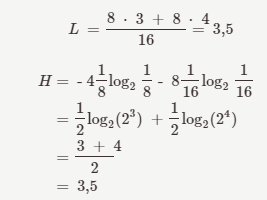
Kde poté co vložíme text, který chceme zakódovat, získáme zakódování text. Dále získáme délku kódu a symboly, ke kterým dostaneme dále frekvenci a kód.

Huffmanova metoda

Huffmanovo kódování zprávy DIGITAL OVERALT! je



Máme celkem 16 symbolů, ale pouze 12 jedinečných, kde je počet výskytů buď 1 nebo 2. To znamená, že každou pravděpodobnost lze zapsat jako zlomek, kde čitatel je 1 a jmenovatel je mocnina 2 (8 nebo 16). To znamená, že každou pravděpodobnost symbolu lze zapsat jako 1 /2k pro k = 3 , 4. V tomto případě (pokud $ k $ je celé číslo) víme, že Huffmanovo kódování bude kódovat zprávy bez kódovací redundance, což znamená, že entropie se rovná průměrnému počtu bitů na symbol. Můžeme tedy najít entropii v této zprávě výpočtem průměrného počtu bitů na symbol po kódování, které nezahrnuje žádné logaritmy.



Přidání f pro každý symbol

def create\_prob\_model(text):  
 *"""  
 Vytvoření alphabety na základě zprávy a ke každému symbolu přidáme její frekvenci  
 """* prob\_model = defaultdict (int) # Každý nový znak má hodnotu 0  
 for char in text:  
 prob\_model[char] += 1 / len (text)  
  
 return prob\_model

Huffman zakódování +source code(formátováni textu)

def huff\_encode(text):  
 *"""Huffmanovo kodování vstupní zprávy .  
  
 Args:  
 text: String, pořadí znaků(symbolů)  
  
 Returns:  
 source\_code: Dict kde každá položka je {symbol: (frekvence, kod)}  
 encoded\_message: String, bitový řetězec kodové vstupní zprávy pomocí source\_code  
  
 """* prob\_model = create\_prob\_model (text)  
  
 heap = [[freq,[sym,""]] for sym,freq in prob\_model.items ()]  
 heapify (heap)  
 while len (heap) > 1:  
 lo = heappop (heap)  
 hi = heappop (heap)  
 for pair in lo[1:]:  
 pair[1] = '0' + pair[1]  
 for pair in hi[1:]:  
 pair[1] = '1' + pair[1]  
 heappush (heap,[lo[0] + hi[0]] + lo[1:] + hi[1:])  
 symbol\_code\_pairs = sorted (heappop (heap)[1:],key=lambda p:(len (p[-1]),p))  
  
 """Kod pro formátování textu """  
 source\_code = {}  
 for pair in symbol\_code\_pairs:  
 symbol = pair[0]  
 codeword = pair[1]  
 freq = prob\_model[symbol]  
 source\_code[symbol] = (freq,codeword)  
  
 encoded\_message = ''  
 for symbol in text:  
 encoded\_message += source\_code[symbol][1]  
  
 return source\_code,encoded\_message

Vstup/Výstup pro zakódování

"""Vložení textu, který chceme přeložit"""  
text ="Toto je test"  
source\_code,encoded\_message = huff\_encode (text)  
textde=len(encoded\_message)  
print ("Text: {}".format (text))  
print ("Zakodovaný text: {}".format (encoded\_message))  
print ("Délka textu: {}\n".format (textde))  
  
print ("Symbol Frekvence Kod")  
for symbol,pair in sorted (source\_code.items ()):  
 print("{0:>2} {1:>9,.2f} {2:>8}".format(symbol, pair[0], pair[1]))

Test daného vstupu – Toto je novy test pro testovani daného kodu

