УНИВЕРСИТЕТ ИТМО

ФГАОУ ВО «Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

Факультет программной инженерии и компьютерной техники

ПРИКЛАДНАЯ МАТЕМАТИКА

Лабораторная работа №2

Построение оптимальных кодов

Лабушев Тимофей

Группа P3302

Санкт-Петербург

2019

Цель работы

Изучение основных принципов эффективного кодирования и приобретение практических навыков построения оптимальных кодов на примере кодов Шеннона-Фано и Хаффмана, оценка их эффективности.

Задание

1. Реализовать процедуры построения кода Шеннона-Фано и оптимального кода Хаффмана
2. Построить коды для текстового файла, распечатать кодовые таблицы, содержащие *символ*, *вероятность*, *кодовое слово*, *длину кодового слова*.
3. Сравнить среднюю длину кодовых слов, полученных двумя алгоритмами.

Исходный код

Построение кода Хаффмана

```
#lang racket

(provide huffman)

(define-struct node (sym prob left right) #:transparent)
(define init-node (λ (prob-entry)
  (node (first prob-entry) (second prob-entry) null null)))
(define group-node (λ (left right)
  (define psum (+ (node-prob left) (node-prob right)))
  (node null psum left right)))

(define huffman (λ (probabilities)
  (let* ([prob-nodes (map init-node probabilities)]
         [tree (huffman-tree prob-nodes)]
         [codelist (fold-tree-to-codelist tree "" '())])
  (sort codelist <= #:key fourth))))

(define huffman-tree (λ (tree)
  (if (= 2 (length tree))
    (group-node (first tree) (second tree))
    (let* ([sorted-tree (sort tree >= #:key node-prob)]
           [updated-tree (match/values (split-at-right sorted-tree 2)
             [(head (list n1 n2)) (cons (group-node n1 n2) head)])])
          (huffman-tree updated-tree)))))

(define fold-tree-to-codelist (match-lambda*
  [(list (struct* node ([sym s] [prob p] [left null] [right null])) code lst)
    (cons (list s p code (string-length code)) lst)]
  [(list (struct* node ([left l] [right r])) code lst)
     (fold-tree-to-codelist r (~a code "1")
       (fold-tree-to-codelist l (~a code "0") lst))]))
```

Построение кода Шеннона-Фано

```
#lang racket

(provide shannon-fano)

(define shannon-fano (λ (probabilities)
  (let* ([codelist (shannon-fano-rec probabilities "" '())]
         [codelist-with-len (map
           (λ (c) (append c (list (string-length (third c))))) codelist)])
  (sort codelist-with-len >= #:key second))))
(define shannon-fano-rec (λ (alphabet code-prefix codes) (match alphabet
  [(list (list sym prob))
    (cons (list sym prob code-prefix) codes)]
  [(list syms ...)
    (match-let ([(list left right) (partition-eqsum syms)])
      (shannon-fano-rec right (~a code-prefix "1")
        (shannon-fano-rec left (~a code-prefix "0") codes)))])))

(define partition-eqsum (λ (lst)
  (partition-eqsum-rec (sort lst >= #:key second) second '() 0 '() 0)))
(define partition-eqsum-rec (λ (lst key left left_sum right right_sum) (match lst
  [(list) (cons left (cons right '()))]
  [(list h tail ...) #:when (< left_sum right_sum)
    (partition-eqsum-rec tail key (cons h left) (+ left_sum (key h)) right right_sum)]
  [(list h tail ...)
    (partition-eqsum-rec tail key left left_sum (cons h right) (+ right_sum (key h)))])))
```

Результаты работы программы

Код Хаффмана

Символ	Вероятность	Кодовое слово	Длина кодового слова
	0.18212	000	3
e	0.09922	110	3
t	0.07453	0011	4
a	0.06545	0110	4
o	0.06015	0111	4
n	0.053	1000	4
s	0.05075	1010	4
i	0.05062	1011	4
r	0.04873	1110	4
h	0.04387	00100	5
l	0.034	01001	5
d	0.03353	01010	5
.	0.03227	01011	5
c	0.02287	11110	5
u	0.02173	11111	5
m	0.021	001010	6
w	0.01735	010000	6
f	0.01703	010001	6
p	0.01378	100100	6
g	0.01372	100101	6
y	0.01315	100110	6
b	0.01168	100111	6
v	0.00877	0010111	7
k	0.00497	00101101	8
x	0.0019	0010110000	10
j	0.00098	0010110011	10
z	0.00077	00101100011	11
q	0.00068	00101100100	11
1	0.00032	001011001010	12
3	0.0002	0010110001000	13
9	0.0002	0010110001001	13
0	0.00018	0010110001011	13
4	0.00015	0010110010110	13
2	0.00015	0010110010111	13
7	0.0001	00101100010100	14
5	0.00003	001011000101011	15
8	0.00003	0010110001010100	16
6	0.00002	0010110001010101	16

Код Шеннона-Фано

Символ	Вероятность	Кодовое слово	Длина кодового слова
	0.18212	111	3
e	0.09922	0111	4
t	0.07453	0011	4
a	0.06545	0001	4
o	0.06015	1011	4
n	0.053	01011	5
s	0.05075	10011	5
i	0.05062	00001	5
r	0.04873	10001	5
h	0.04387	10101	5
l	0.034	01001	5
d	0.03353	01000	5
.	0.03227	1101	4
c	0.02287	001011	6
u	0.02173	10000	5
m	0.021	001001	6
w	0.01735	11001	5
f	0.01703	01101	5
p	0.01378	101001	6
g	0.01372	110001	6
y	0.01315	00000	5
b	0.01168	01100	5
v	0.00877	101000	6
k	0.00497	110000	6
x	0.0019	0010001	7
j	0.00098	010101	6
z	0.00077	1001011	7
q	0.00068	010100	6
1	0.00032	10010011	8
3	0.0002	10010000	8
9	0.0002	100100011	9
0	0.00018	0010101	7
4	0.00015	0010000	7
2	0.00015	10010010	8
7	0.0001	10010101	8
5	0.00003	100100010	9
8	0.00003	10010100	8
6	0.00002	0010100	7

Сравнение кодов

Найдем среднюю длину кодового слова, взвешенную по вероятности появления в тексте:

Код Хаффмана:

```
4.208433333333332
```

Код Шеннона-Фано:

```
4.400283333333333
```

Выводы

В ходе выполнения лабораторной работы было установлено, что код Хаффмана является более оптимальным с учетом вероятности появления символов в исходном тексте.