

Tutorat 2

RETI, Huffman-Kodierung

Gruppe 4

Präsentatorin:
Gabriella Frank
(gabriella-frank@web.de)

Vorlesung von:
Prof. Dr. Scholl

Übungsgruppenbetreuung:
Tobias Seufert

9. Mai 2023

Universität Freiburg, Lehrstuhl für Rechnerarchitektur

Gliederung

Aufgabe 1

Aufgabe 2

Aufgabe 3

Aufgabe 1

Aufgabe 1 I

XYZ

Aufgabe 1.1

Es gilt für $n \geq 0$:

$$f(0) = 0$$

$$f(1) = 1$$

$$f(n) = f(n-1) + f(n-2) \quad (\text{für } n \geq 2)$$

Aufgabe 1 II

XYZ

Lösung 1.1



```
1  int fib(int fib_nr) {
2      int fib_i;
3      int fib_i_1 = 0;
4      int fib_i_2 = 1;
5      while (fib_nr) {
6          fib_i = fib_i_1 + fib_i_2;
7          fib_i_2 = fib_i_1;
8          fib_i_1 = fib_i;
9          fib_nr = fib_nr - 1;
10     }
11     return fib_i;
12 }
13
14 void main() {
15     print(fib(input()));
16 }
```

Aufgabe 1 III

XYZ

Lösung 1.1



```
1  # INPUT
2  CALL INPUT ACC
3  STORE ACC 30
4  # PROGRAM
5  LOADI ACC 0
6  STORE ACC 31
7  LOADI ACC 1
8  STORE ACC 32
9  LOAD ACC 30
10 SUBI ACC 1
11 STORE ACC 30
12 JUMP== 9
```

```
13 LOAD ACC 31
14 ADD ACC 32
15 STORE ACC 33
16 LOAD ACC 32
17 STORE ACC 31
18 LOAD ACC 33
19 STORE ACC 32
20 JUMP -11
21 # OUTPUT
22 LOAD ACC 33
23 CALL PRINT ACC
```

Aufgabe 1 IV

XYZ

Lösung 1.1



```

Stack: 01
Instruction: STORE ACC 31;
ACC: 3
ACC_SIMPLE: 3
IN1: 0
IN1_SIMPLE: 0
IN2: 0
IN2_SIMPLE: 0
PC: 2147483666
PC_SIMPLE: 10
SP: 2147483693
SP_SIMPLE: 45
BAF: 2147483658
BAF_SIMPLE: 2
CS: 2147483651
CS_SIMPLE: 3
DS: 2147483671
DS_SIMPLE: 23
SRAM:
00000 JUMP 0;
00001 2147483648
00002 0 <- BAF
00003 CALL INPUT ACC; <- CS
00004 STORE ACC 30;
00005 LOAD1 ACC 0;
00006 STORE ACC 31;
00007 LOAD1 ACC 1;
00008 STORE ACC 32;
00009 LOAD ACC 30;
00010 SURE ACC 1;
00011 STORE ACC 30;
00012 JUMP= 0;
00013 LOAD ACC 31;
00014 ADD ACC 32;
00015 STORE ACC 33;
00016 LOAD ACC 32;
00017 STORE ACC 31;
00018 LOAD ACC 33; <- PC
00019 STORE ACC 32;
00020 JUMP -11;
nah

00021 LOAD ACC 31;
00022 CALL PRINT ACC;
00023 0 <- DS
00024 0
00025 0
00026 0
00027 0
00028 0
00029 0
00030 5
00031 1
00032 3
00033 5
00034 0
00035 0
00036 0
00037 0
00038 0
00039 0
00040 0
00041 0
00042 0
00043 0
00044 0
00045 0 <- SP
UART:
00000 0
00001 0
00002 0
EPRM:
00000 LOAD1 DS -2097152; <- IN1 <- IN2
00001 MULT1 DS 1024;
00002 MOVE DS SP;
00003 MOVE DS BAF; <- ACC
00004 MOVE DS CS;
00005 ADD1 SP 45;
00006 ADD1 BAF 2;
00007 ADD1 CS 3;
00008 ADD1 DS 23;
00009 MOVE CS PC;

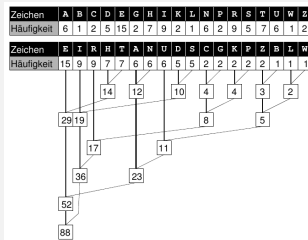
```

Aufgabe 2

Aufgabe 2 I

XYZ

Lösung 2.1



Sidenote

- ▶ die Codewörter müssen ausgehend von der Wurzel zu den Blättern aufgebaut werden, sonst ergibt sich kein Präfixcode

Aufgabe 2 II

XYZ

Lösung 2.1



Zeichen	E	I	R	H	T	A	N	U	D	S	C	G	K	P	Z	B	L	W
Code	000	100	110	0010	0011	0100	0101	0110	1010	1011	11100	11101	11110	11111	011100	011101	011110	011111

Sidenote

- ▶ Ein Präfixcode ist ein Code, bei dem kein Codewort Präfix eines anderen Codewortes ist.
- ▶ Z.B. wäre ein Code mit den Codewörtern $a = 10$, $b = 101$ kein Code, da man beim Dekodieren einer Nachricht beim Lesen von 10 nicht weiß, ob damit a oder, bei einer nachfolgenden 1, das Wort b gemeint ist.
- ▶ Einen Präfixcode kann man also „online“ dekodieren, d.h. man liest Zeichen für Zeichen und bei einem Matching mit einem Wort im Wörterbuch hat man das ursprüngliche Wort gefunden.

Aufgabe 2 III

VV7

Lösung 2.2

- *Dekodierung: TI IST KLASSE*

$\underbrace{0011}_T$
 $\underbrace{100}_I$
 $\underbrace{100}_I$
 $\underbrace{1011}_S$
 $\underbrace{0011}_T$
 $\underbrace{11110}_K$
 $\underbrace{011110}_L$
 $\underbrace{0100}_A$
 $\underbrace{1011}_S$
 $\underbrace{1011}_S$
 $\underbrace{000}_E$

- *Mögliches Problem: Codierung von Blättern hin zur Wurzel → kein Präfix-Code, Dekodierung nicht möglich (s.o.)*

Lösung 2.3

- *Damit andere die Nachricht dekodieren (d.h. lesen) können, muss man natürlich auch das Wörterbuch (d.h. die generierte Code-Tabelle) mitliefern.*
- *Da damit jedem, der die kodierte Nachricht erhält, auch die Dekodierungstabelle zugänglich ist, kann natürlich jeder die Nachricht dekodieren. D.h. die kodierte Nachricht genügt keinen kryptologischen Ansprüchen.*
- *Was man allerdings erreicht hat, ist eine Datenkomprimierung (bei großen Texten).*

Aufgabe 3

Aufgabe 3 I

XYZ

Voraussetzungen 3.1

$$\sum_{j=1}^m p(a_j) = 1, p(a_i) > 0.5$$

Lösung 3.1

$$\sum_{j=1}^{i-1} p(a_j) + p(a_i) + \sum_{j=i+1}^m p(a_j) = 1$$

$$\sum_{j=1}^{i-1} p(a_j) + \sum_{j=i+1}^m p(a_j) < 0.5$$

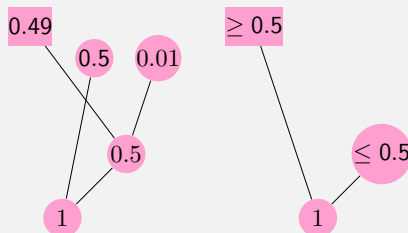
Aufgabe 3 II

XYZ

Lösung 3.1



- ▶ Beim Bau des binären Baums sind alle Häufigkeitsteilsummen kleiner als $p(a_i)$, daher wird a_i erst zu dem Baum hinzugefügt, nachdem alle andere Knoten bereits zu einem Knoten zusammengefügt wurden.
- ▶ Dieser wird durch eine Kante mit einem neuen Knoten (Wurzel) verbunden, die andere Kante der Wurzel führt direkt zu a_i .



Aufgabe 3 I

XYZ

Voraussetzungen 3.1

- ▶ $p(a_i) < \frac{1}{3}$ und $|c(a_i)| = 1$

Lösung 3.1

- ▶ Wegen $|c(a_i)| = 1$ ist a_i einer der beiden direkten Vorgänger der Wurzel des Codebaumes. Für den anderen Vorgänger v gilt:
 - a) Er ist mit einer Häufigkeitssumme $> 2/3$ beschriftet, da die Summe an der Wurzel 1 ist und $p(a_i) < 1/3$.
 - b) Der Vorgänger v ist kein Blatt wegen $m \geq 3$.
- ▶ Also hat v mindestens einen Vorgänger v_1 mit Häufigkeitssumme größer als $\frac{1}{3}$. Der Algorithmus hätte dann aber zum Zeitpunkt des Einfügens von v nicht v_1 , sondern einen Knoten mit einer kleineren Häufigkeitssumme als Vorgänger von v auswählen müssen. Ein solcher Kandidat wäre z.B. a_i mit $p(a_i) < 1/3$ gewesen.
- ▶ Widerspruch!

Aufgabe 3 II

XYZ

Lösung 3.1

