Tutorat 2

RETI, Huffman-Kodierung

Gruppe 9

Präsentator:
Jürgen Mattheis
(juergmatth@gmail.com)

Vorlesung von:
Prof. Dr. Scholl

Übungsgruppenbetreuung:
Tobias Seufert

11. Mai 2023

Universität Freiburg, Lehrstuhl für Rechnerarchitektur

Gliederung

Aufgabe 1

Aufgabe 1 I

RETI

Aufgabe 1.1

Es gilt für $n \geq 0$:

$$
\begin{aligned}
f(0) &= 0 \\
f(1) &= 1 \\
f(n) &= f(n-1) + f(n-2) \qquad (\text{für } n \geq 2)
\end{aligned}
$$

Aufgabe 1 II

RETI

Lösung 1.1

```
int fib(int fib_nr) {
  int fib_i;
  int fib_i_1 = 0;
  int fib_i_2 = 1;
  while (fib_nr) {
    fib_i = fib_i_1 + fib_i_2;
    fib_i_2 = fib_i_1;
    fib_i_1 = fib_i;
    fib_nr = fib_nr - 1;
  }
  return fib_i;
}

void main() {
  print(fib(input()));
}
```

Aufgabe 1 III

RETI

Lösung 1.1

```
# INPUT
CALL INPUT ACC
STORE ACC 30
# PROGRAM
LOADI ACC 0
STORE ACC 31
LOADI ACC 1
STORE ACC 32
LOAD ACC 30
SUBI ACC 1
STORE ACC 30
JUMP== 9
LOAD ACC 31
ADD ACC 32
STORE ACC 33
LOAD ACC 32
STORE ACC 31
LOAD ACC 33
STORE ACC 32
JUMP -11
# OUTPUT
LOAD ACC 33
CALL PRINT ACC
```

Aufgabe 1 IV

RETI

Lösung 1.1

Aufgabe 2

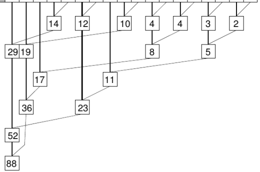
Aufgabe 2 I

Huffman-Kodierung

Lösung 2.1

Zeichen	A	B	C	D	E	G	H	I	K	L	N	P	R	S	T	U	W	Z
Häufigkeit	6	1	2	5	15	2	7	9	2	1	6	2	9	5	7	6	1	2

Zeichen	E	I	R	H	T	A	N	U	D	S	C	G	K	P	Z	B	L	W
Häufigkeit	15	9	9	7	7	6	6	6	5	5	2	2	2	2	2	1	1	1



Anmerkungen

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

Aufgabe 2 II

Huffman-Kodierung

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*

Anmerkungen

- 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

Huffman Kodierung

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

Huffman-Kodierung

Voraussetzungen 3.1

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

Lösung 3.1

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

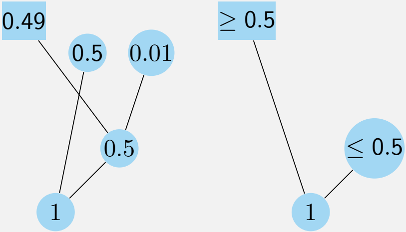
$$\sum_{j=1}^{i-1} p(a_j) + \sum_{j=i+1}^{m} p(a_j) < 0.5$$

Aufgabe 3 II

Huffman-Kodierung

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

Huffman-Kodierung

Voraussetzungen 3.1

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

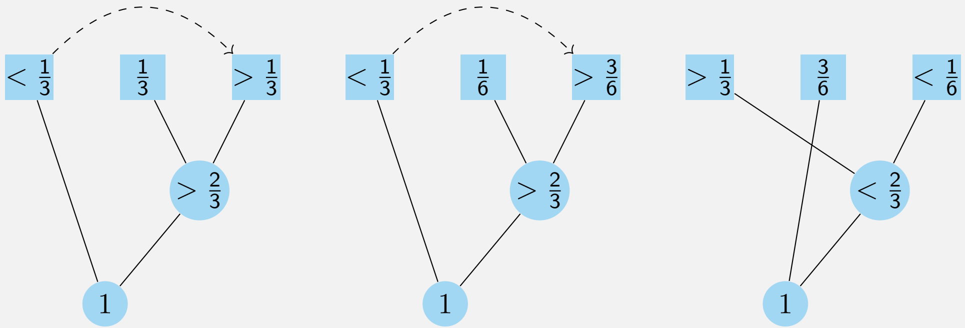
Lösung 3.1

- *Wegen* $|c(ai)| = 1$ *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

Huffman-Kodierung

Lösung 3.1



Bonus

Task 4 I

Beweis durch Widerspruch

- anstatt einen mathematischen Satz S direkt zu beweisen, kann man seine Negation $\neg S$ durch logische Schlussfolgerungen zu einem Widerspruch führen.
- Wenn man die Widerspruchsanahme $\neg S$ zu einem Widerspruch geführt hat, weiß man, dass $\neg S$ immer falsch sein muss. Damit ist die doppelte Negation $\neg\neg S$ von S wahr. Da $\neg\neg S \Leftrightarrow S$ eine Tautologie ist, ist $\neg\neg S$ genau dann wahr, wenn S wahr ist. Damit muss S wahr sein.
- Zerlegung der Implikation:

$$
\begin{aligned}
&\mathit{Voraussetzung}_1 \wedge \cdots \wedge \mathit{Voraussetzung}_n \rightarrow \mathit{Behauptung} \\
&\Leftrightarrow \neg\,(\mathit{Voraussetzung}_1 \wedge \cdots \wedge \mathit{Voraussetzung}_n) \vee \mathit{Behauptung} \\
&\Leftrightarrow \neg\,((\mathit{Voraussetzung}_1 \wedge \cdots \wedge \mathit{Voraussetzung}_n) \wedge \neg \mathit{Behauptung}) \\
&\Leftrightarrow \mathit{Voraussetzung}_1 \wedge \cdots \wedge \mathit{Voraussetzung}_n \wedge \neg \mathit{Behauptung} \rightarrow \perp
\end{aligned}
$$

Task 4 II

Beweis durch Widerspruch

- Anders gesagt: Aus den Vorraussetzungen folgt logisch die Behauptung *genau dann wenn* $Voraussetzung_1 \wedge \cdots \wedge$ $Voraussetzung_n \wedge \neg Behauptung$ *NICHT* gilt.

Anmerkungen

- es gibt neben explizten Voraussetzungen auch implizite Voraussetzungen, die nicht ausdrücklich genannt werden (Rechenregeln und Standard-Definitionen)
- In der Behauptung steht immer eine wahre Aussage. Hat man eine Aussage, die nicht wahr ist, muss man sie negiert in die Behauptung schreiben.

Task 4 III

Beweis durch Widerspruch

Behauptung: $\mathcal{F}$

Beweis: Durch Widerspruch.

Annahme: Die Behauptung wäre falsch, das heißt, $\neg\mathcal{F}$ würde gelten.

Teilbeweis unter Verwendung dieser Annahme, der mit einem Widerspruch endet.

Die Annahme gilt also nicht, deshalb gilt die Behauptung.

Task 4 IV

Beweis durch Widerspruch

Wir zeigen: Eine natürliche Zahl mit geradem Quadrat ist selbst gerade.

Voraussetzung:

$(\star)\ \forall n \in \mathbb{N}(\neg gerade(n) \Leftrightarrow \exists k \in \mathbb{N} \text{ mit } n = 2k+1)$ (Eigenschaft von *gerade*)

Behauptung: $\forall n \in \mathbb{N}(gerade(n^2) \Rightarrow gerade(n))$

Beweis: Durch Widerspruch.

Annahme: Die Behauptung wäre falsch, das heißt, es würde gelten:
$\neg\forall n \in \mathbb{N}(gerade(n^2) \Rightarrow gerade(n))$.

$$
\begin{aligned}
&\Rightarrow \exists n \in \mathbb{N}[gerade(n^2) \wedge \neg gerade(n)] && \text{(Rechenregeln für } \neg) \\
&\Rightarrow \exists n \in \mathbb{N}[gerade(n^2) \wedge \exists k \in \mathbb{N} \text{ mit } n = 2k+1\ ] && \text{(wegen } (\star)) \\
&\Rightarrow \exists n \in \mathbb{N}[gerade(n^2) \wedge \exists k \in \mathbb{N} \text{ mit } n^2 = (2k+1)^2 && \text{(Quadrierung)} \\
&\qquad = 4k^2 + 4k + 1 && \text{(arithmetische Umformung)} \\
&\qquad = 2(2k^2+2k) + 1\,] && \text{(arithmetische Umformung)} \\
&\Rightarrow \exists n \in \mathbb{N}[gerade(n^2) \wedge \neg gerade(n^2)] && \text{(wegen } (\star))
\end{aligned}
$$

Widerspruch.

Die Annahme gilt also nicht, deshalb gilt die Behauptung. ∎