Datenstrukturen und Algorithmen

Abgabe 11
Abgabe: 19.07.2017

Georg C. Dorndorf Matr.Nr. 366511
Adrian C. Hinrichs Matr.Nr. 367129

# 5	# 6	# 7	# 8	# 9	Σ
3/6	3/6	6/6	2/6	5/8	14/32

Aufgabe 5

Es ist $w[] = \{3, 2, 1, 3, 2, 2, 2\}$ und $c[] = \{5, 8, 6, 7, 2, 4, 3\}$. Damit der Algorithmus trotz der in der Aufgabenstellung genannten Einschränkung funktioniert kann man die von einander abhängigen Gegenstände als einen neuen Gegenstand modellieren. Wir erhalten dann neue Arrays in denen der 1. und 6. sowie der 5. und 7. Gegenstand zusammengefasst ist. $w[] = \{3 + 2, 2, 1, 3, 2 + 2\} = \{5, 2, 1, 3, 4\}$ und $c[] = \{5 + 4, 8, 6, 7, 2 + 3\} = \{9, 8, 6, 7, 5\}$

	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	9	9	9	9	9	9
2	0	0	8	8	8	9	9	17	17	17	17
3	0	6	8	14	14	14	15	17	23	23	23
4	0	6	8	14	14	15	21	21	23	23	24
5	0	6	8	14	14	15	21	21	23	23	24

Tabelle 1: Die vom Algorithmus bestimmte Tabelle C

welche Gegenstände sollen mitgenommen werden? max. Wert? −2

Aufgabe 6

Aufgabe 7

1. Z-1 D-2 L-2 U-2 A-3 C-3 G-3 X-4 F-5 V-5 Y-5 R-6 H-8 W-8 O-9 I-13 N-16 T-22 E-26 S-27
2. L-2 U-2 A-3 C-3 G-3 ZD-3 X-4 F-5 V-5 Y-5 R-6 H-8 W-8 O-9 I-13 N-16 T-22 E-26 S-27
3. A-3 C-3 G-3 ZD-3 LU-4 X-4 F-5 V-5 Y-5 R-6 H-8 W-8 O-9 I-13 N-16 T-22 E-26 S-27
4. G-3 ZD-3 LU-4 X-4 F-5 V-5 Y-5 AC-6 R-6 H-8 W-8 O-9 I-13 N-16 T-22 E-26 S-27
5. LU-4 X-4 F-5 V-5 Y-5 AC-6 GZD-6 R-6 H-8 W-8 O-9 I-13 N-16 T-22 E-26 S-27
6. F-5 V-5 Y-5 AC-6 GZD-6 R-6 H-8 LUX-8 W-8 O-9 I-13 N-16 T-22 E-26 S-27
7. Y-5 AC-6 GZD-6 R-6 H-8 LUX-8 W-8 O-9 FV-10 I-13 N-16 T-22 E-26 S-27
8. GZD-6 R-6 H-8 LUX-8 W-8 O-9 FV-10 YAC-11 I-13 N-16 T-22 E-26 S-27
9. H-8 LUX-8 W-8 O-9 FV-10 YAC-11 GZDR-12 I-13 N-16 T-22 E-26 S-27
10. W-8 O-9 FV-10 YAC-11 GZDR-12 I-13 HLUX-16 N-16 T-22 E-26 S-27
11. FV-10 YAC-11 GZDR-12 I-13 HLUX-16 N-16 WO-17 T-22 E-26 S-27
12. GZDR-12 I-13 HLUX-16 N-16 WO-17 FVYAC-21 T-22 E-26 S-27
13. HLUX-16 N-16 WO-17 FVYAC-21 T-22 GZDRI-25 E-26 S-27
14. WO-17 FVYAC-21 T-22 GZDRI-25 E-26 S-27 HLUXN-32
15. T-22 GZDRI-25 E-26 S-27 HLUXN-32 WOFVYAC-36
16. E-26 S-27 HLUXN-32 WOFVYAC-36 TGZDRI-47
17. HLUXN-32 WOFVYAC-38 TGZDRI-47 ES-53
18. TGZDRI-47 ES-53 HLUXNWOFVYAC-70
19. HLUXNWOFVYAC-70 TGZDRIES-100
20. HLUXNWOFVYACTGZDRIES-170 (v)

Der Binärbaum zu dem durch obigen Algortihmus hergeleiteten HUFFMAN-CODE ist am Ende unter Abbildung 1 abgebildet.

anderen Sortierung als VL

Aufgabe 8

a)

b)

c)

d)

e)

f)

Aufgabe 9

a)

Sei $\mathfrak{C} := \{1, 2, 4, 10, 20, 50\}$

Gegeben: Der in der Aufgabenstellung spezifizierte GREEDY ALGORITHMUS $A : \mathbb{N}_0 \leftarrow \mathfrak{C}^n, n \in \mathbb{N}_0$

Funktionenfamilie? "←"?

Notation: Unter missbrauch der Notation finden alle Operationen und Vergleiche auf Elementen einer Menge, welche durch die künstliche? disjunkte Vereinigung zweier anderer Mengen entstanden ist, rekursiv auf deren ersten Einträgen statt.? Diese sind immer Elemente aus $\{1, 2, 4, 10, 20, 50\}$.

Zu zeigen: Der Algorithmus findet immer eine optimale Lösung.
Sei eine Optimale Lösung $\mathfrak{O}$ gegeben. Es ist $\mathfrak{O}$ eine Menge an Münzen, so dass für alle anderen Lösungen $\mathfrak{l}$ gilt: $|\mathfrak{O}| \leq |\mathfrak{l}|$. Dies ist Äquivalent zu der Aussage, dass für die vom Algorithmus gefundenen (aufsteigend geordneten und als Tupel aufgefassten) optimalen Lösungen $\mathfrak{O} = (\mathfrak{o}_0, ..., \mathfrak{o}_n)$ die Lösung $\mathfrak{o}_0, ..., \mathfrak{o}_{n-1} = A(\sum_{i\in[1,n-1]} \mathfrak{o}_i)$ eine Optimale Lösung für das in der Aufgabenstellung spezifizierte Problem mit dem Zeilwert $\sum_{i\in[1,n-1]} \mathfrak{o}_i$ ist.

$$\Leftrightarrow A(\sum_{i\in[1,n]} \mathfrak{o}_i) = A(\sum_{i\in[1,n-1]} \mathfrak{o}_i) \leftarrow (\mathfrak{o}_n)$$

(Wobei folgende definition gilt: $\leftarrow$: $\mathfrak{C}^n \times \mathfrak{C} \rightarrow \mathfrak{C}^{n+1}, ((c_1, ..., c_n), c_{n+1}) \mapsto (c_1, ..., c_n, c_{n+1})$)

Sei ab nun ein bel. aber festes n gegeben.
Beweis per Wiedersprueh
Anngenommen, dass $(\mathfrak{o}_1, ..., \mathfrak{o}_{n-1})$ zwar eine Optimale Lösung für $\sum_{i\in[1,n-1]} \mathfrak{o}_i$, aber $A(\sum_{i\in[1,n-1]} \mathfrak{o}_i) \leftarrow (\mathfrak{o}_n)$ keine Optimale Lösung für $\sum_{i\in[1,n]} \mathfrak{o}_i$ ist. Nun müssen verschiedene $i_1, ..i_m \in \mathbb{N}^{\leq n}$ mit $m < n - 1$ existieren, so dass $(\mathfrak{o}_{i_1}, ..., \mathfrak{o}_{i_m})$ immer noch eine sortierte Lösung für oben genanntes Problem ist. (*Auf Deutsch: Man muss wenigstens ein mal wenigstens zwei Münzen durch eine Gößere ersetzen können*). Nun kann die gößte Münze aus der Lösung nicht durch eine Größere ersetzt werden, da sie entweder selber die größte mögliche Münze (.50€) ist oder der Zielwert kleiner als 50 Cent ist und man nicht durch Addieren des Wertes zweier Münzen den Wert der nächst höheren Münze überschreiten kann[1]. (Der Algorithmus sucht für $\mathfrak{o}_n$ den größten Wert kleinergleich dem Zielwert). Daher gilt:

$$(\mathfrak{o}_{i_1}, ..., \mathfrak{o}_{i_m}) = (\mathfrak{o}_{i_1}, ..., \mathfrak{o}_{i_m-1}, \mathfrak{o}_n)$$

Daraus folgt jedoch, dass in der Lösung für $\sum_{i\in[1,n-1]} \mathfrak{o}_i$ mindestens zwei Münzen durch eine Größere ersetzt werden können, wodurch diese nicht Optimal wäre. Dies widerspricht der Annahme. Nach dem Prinzip der Vollständigen Induktion gilt nun die Behauptung, dass A für jeden durch eine Optimale Münzmenge darstellbaren Wert eine Optimale Lösung findet.

□

Es kann ofensichtlich jeder Betrag dargestellt werden, daher kann man jeden Betrag auch durch eine Optimale Münzmenge darstellen.

QED

[1] Um missinterpretation der Verschachtelung der Und- und Oderverknüpfungen in diesem Satz zu verhindern, wird das —im Deutschen orthografisch falsche— Oxford-Komma verwendet. Ich bitte dies zu entschuldigen

b)

Da auch bei dieser Menge der Betrag einer Münze nicht durch addition der Beträger zweier kleinerer Münzen überschritten werden kann, ist der Beweis von Aufgabenteil a anwendbar. Das stimmt nicht. −1
Münze 11, 5, 1
Ziel 15

QED

c) opt: 3.5, greedy: 11, 4.1

Durch schrafes hinsehen fällt auf, dass eine Optimale Lösung für das Problem zu dem Wert 104 Cent gegeben ist durch (70, 34), findet der Algorithmus folgendes Ergebnis:

$$\begin{aligned}
A(104) &= A(4) \leftarrow (100) \\
&= (A(3) \leftarrow (1)) \leftarrow (100) = A(3) \leftarrow (1, 100) \\
&= (A(2) \leftarrow (1)) \leftarrow (1, 100) = A(2) \leftarrow (1, 1, 100) \\
&= (A(1) \leftarrow (1)) \leftarrow (1, 1, 100) \\
&= A(1) \leftarrow (1, 1, 1, 100) \\
&= (A(0) \leftarrow (1)) \leftarrow (1, 1, 1, 100) \\
&= \varnothing \leftarrow (1, 1, 1, 1, 100) \\
&= (1, 1, 1, 1, 100) \\
&\neq (34, 70)
\end{aligned}$$

−2
Warum ist denn klar, dass der Alg. dafür die richtige Lösung findet?

Also ist der Algorithmus für diese Münzsystem nicht Optimal.

/5