

**Algorithmen und Datenstrukturen**  
SS 2018

**Übungsblatt Global 10: Paradigmen**

**Aufgabe Global 10-1**      *Knobelei: Piratenschatz aufteilen*

Ein Piratenkapitän hat 100 Münzen, die er auf sich und seine 4 Personen starke Mannschaft aufteilen muss. Da Piraten demokratisch organisiert sind, darf jeder bei der Verteilung mitbestimmen. Die Prozedur folgt folgenden Regeln:

1. Der Kapitän schlägt eine Verteilung der Münzen vor. Es gibt nur ganze Münzen.
2. Stimmen mindestens 50% der Besatzung zu, wird der Vorschlag akzeptiert.
3. Falls nicht, so muss der Kapitän über die Planke gehen und der nächste Pirat ersetzt ihn. Eine Rangfolge besteht schon.
4. Jeder Pirat hat als oberstes Ziel, am Leben zu bleiben.
5. Zweitens möchte jeder seinen Gewinn maximieren.
6. Es gibt keine Enthaltung. Im Zweifel lässt jeder Pirat einen Kapitän über die Planke gehen, falls beide Auswahlmöglichkeiten für ihn gleichwertig sind.
7. Jeder Pirat hat die gleiche Zielsetzung und weiß auch, dass dies für die anderen gilt.
8. Piraten können in keiner Weise Absprachen treffen oder sonst zusammen arbeiten. Keine Tricks, Täuschungen, Bestechungen oder sonstige Kommunikation außerhalb der Abstimmungen.
9. Jeder Pirat ist ein ausgezeichneter Logiker.

Wie verteilt der Kapitän die Münzen?

**Aufgabe Global 10-2**      *Münzen und dynamische Programmierung*

Gegeben seien  $n$  verschiedene, positive Münzwerte  $a_1, \dots, a_n$  und ein Betrag  $m$ . Gesucht ist die Zahl der möglichen  $n$ -Tupel  $(k_1, \dots, k_n)$  mit  $\sum_{l=1}^n k_l a_l = m$ . Dies entspricht der Zahl der Möglichkeiten,  $m$  durch die gegebenen Münzwerte darzustellen. Nehmen Sie an, dass beliebig viele Münzen von jeder Sorte zur Verfügung stehen und sich der Betrag  $m = 0$  durch genau einmal 0 Münzen darstellen lässt. Dieses Problem lässt sich in tabellarischer Form mittels dynamischer Programmierung darstellen. Dazu definieren wir Teilprobleme  $t_{i,j}$ , wobei  $i$  die Anzahl der Münzwerte ( $0 \leq i \leq n$ ) sind und  $j$  der Betrag ( $0 \leq j \leq m$ ) ist. Die Lösung des Gesamtproblems ist dann  $t_{n,m}$ . Die folgende Rekursionsgleichung gilt für  $i > 0$ :

$$t_{i,j} = t_{i-1,j} + \begin{cases} t_{i,j-a_i}, & j - a_i \geq 0 \\ 0, & \text{sonst} \end{cases}$$

Geben Sie für  $n = 3, m = 10, a_1 = 2, a_2 = 3, a_3 = 5$  die Matrix  $[t_{i,j}]$  an. Geben Sie zu dem Ergebnis in  $t_{n,m}$  die verschiedenen Möglichkeiten, 10 darzustellen, an. (Achtung: Was ist der Basisfall, von dem der Aufbau der Matrix startet?)

**Aufgabe Global 10-3**     *Greedy Algorithmen, Heuristiken und dynamische Programmierung*

Auf Heuristiken basierende Greedy Algorithmen führen oft nicht zur optimalen Lösung von Problemen. Dennoch finden sich in der Industrie sehr viele Beispiele, bei denen Greedy Algorithmen zur Approximation einer optimalen Lösung eingesetzt werden.

- (a) Warum werden überhaupt Greedy Algorithmen verwendet statt Algorithmen, die eine optimale Lösung garantieren?
- (b) Das Rucksackproblem gehört zu den „schwierigen“ Problemen der Informatik (Stichwort für spätere Vorlesungen: NP-Vollständigkeit). Um eine optimale Lösung für dieses Problem zu finden, können alle Lösungskandidaten naiv durchprobiert werden. Welche Komplexität ergibt sich so?
- (c) Wie groß ist die Komplexität, wenn eine optimale Lösung mit Hilfe von der in der Vorlesung besprochenen dynamischen Programmierung für das Rucksackproblem mit  $n$  verschiedenen Gegenständen und einer Kapazität von  $G$  berechnet wird?