Übungen zur Vorlesung
Einführung in die Algorithmik WS 15/16

Moritz Eissenhauer

Hasso-Plattner-Institut
Potsdam
6. Dezember 2015

Lösung zu Übungsblatt 8

Lösung zu Aufgabe 1
Algorithmus:

Tiefe

Input: n Zeitintervalle mit Start- und Endzeitpunkten
Output: Tiefe d

```
 1: Z ← alle Zeitpunkte als Zahl, mit markierung ob sie Start- oder Endzeiten sind.
 2: Sortiere Z aufsteigend nach Zeitpunkt
 3: max ← 0
 4: current ← 0
 5: for zeitpunkt in Z do
 6:     if zeitpunkt ist Startzeit then
 7:         current++
 8:     else
 9:         current--
10:     end if
11:     if current > max then
12:         max ← current
13:     end if
14: end for
15: return max
```

Laufzeit:
Alle Operationen in der Schleife haben konstante Laufzeit, und die Schleife läuft $2n$ mal (es gibt insgesamt $2n$ Zeitpunkte). Insgesamt hat die Schleife eine Laufzeit von $\mathcal{O}(n)$. Das Initialisieren von Z braucht hat ebenfalls eine Laufzeit von $\mathcal{O}(n)$. Zeile 3 und 4 haben jeweils eine $\mathcal{O}(1)$ Laufzeit. Das Sortieren in Zeile 2 braucht $\mathcal{O}(n \log n)$. Die gesamtlaufzeit ist damit $\mathcal{O}(n \log n)$.

Lösung zu Aufgabe 2

a)

$$\begin{aligned} d(1) = 3,\, t(1) = 1 \\ d(2) = 2,\, t(2) = 2 \end{aligned} \tag{1}$$

$d(i)$ sei die Deadline, $t(i)$ die benötigte Zeit des i-ten Jobs.
Lösung des Algorithmus: 1, 2
$\hookrightarrow$ Maximale Überschreitung der Deadline von 1 (Job 2 wird bei $t = 3$ fertig).
Optimale Lösung: 2, 1
$\hookrightarrow$ Keine Deadine wird verletzt.

b)

$$\begin{aligned} d(1) = 4,\, t(1) = 3 \Rightarrow s(1) = 1 \\ d(2) = 3,\, t(2) = 1 \Rightarrow s(2) = 2 \end{aligned} \tag{2}$$

$s(i)$ist die Slack-time vom i-ten Job.
Lösung des Algorithmus: 1, 2
$\hookrightarrow$ Maximale Überschreitung der Deadline von 1. (Job 2 wird bei $t = 4$ fertig)
Optimale Lösung: 2, 1
$\hookrightarrow$ Keine Deadine wird verletzt.

Lösung zu Aufgabe 3

Beobachtung:
Wenn die Teilnehmer in der Reihnfolge $\sigma = 1, 2, 3, ..., n$ starten dann lässt sich die Gesamtzeit berechnen mit:

$$T(\sigma) = \max\left\{\sum_{i=1}^{k} p_i + r_k | 1 \leq k \leq n\right\} \tag{3}$$

Oder: Der k-te Teilnehmer muss die Paddelzeit all seine Vorgänger abwarten und dann noch selbst paddeln und ins Ziel radeln.

a)

Zu Strategie **(1)**:
$n = 2$

$$\begin{aligned} p_1 = 3,\, r_1 = 1 \Rightarrow t_1 = 4 \\ p_2 = 1,\, r_2 = 2 \Rightarrow t_2 = 3 \end{aligned} \tag{4}$$

Lösung des Algorithmus: $\sigma = 1, 2$
$\hookrightarrow T(\sigma) = \max\{6, 4\} = 6$
Optimale Lösung: $\sigma = 2, 1$
$\hookrightarrow T(\sigma) = \max\{5, 3\} = 5$

Zu Strategie **(2)**:
$n = 2$

$$\begin{aligned} p_1 = 1,\, r_1 = 1 \\ p_2 = 2,\, r_2 = 2 \end{aligned} \tag{5}$$

Lösung des Algorithmus: $\sigma = 1, 2$
$\hookrightarrow T(\sigma) = \max\{5, 2\} = 5$
Optimale Lösung: $\sigma = 2, 1$
$\hookrightarrow T(\sigma) = \max\{4, 4\} = 4$

Lösung zu Aufgabe 4

a)

$$
\begin{aligned}
k &= 2 \\
n &= 2 \\
w(1) &= 1,\ g(1) = 0.1 \\
w(2) &= 10,\ g(2) = 2
\end{aligned} \tag{6}
$$

Algorithmus **(1)** wählt nur Geschenk 1 $\Rightarrow ALG_1 = 1$, Algorithmus **(2)** wählt nur Geschenk 2 $\Rightarrow ALG_2 = 10$.

b)

ZZ. $\frac{OPT}{ALG_1}$ kann beliebig groß werden.

Beweis. Sei $k > 0$ beliebig, fest, sei $n = 2$.
Wähle die Geschenke so, dass für ein beliebiges $x > 0$ folgendes gilt:
$w(1) = 1,\ g(2) = k,\ w(2) = x,\ g(1) = \frac{k}{2x}$.
Sei $e(i)$ die effizienz (Wert pro Gewicht) des i-ten Geschenkes.
Aus den gegebenen Angaben lässt sich die Effizienz der ersten beiden Geschenke bestimmen als $e(1) = 2\frac{x}{k}$, $e(2) = \frac{x}{k}$. Offensichtlich ist die Effizienz des ersten Geschenkes höher als die des Zweiten. Der Algorithmus wählt also zuerst Geschenk 1. Da $g(1) = \frac{k}{2x} > 0$ und $g(2) = k$ Lädt der Algorithmus nur das Geschenk 1 auf den Schlitten. ALG_1 ist damit 1. Es gibt aber immer nur eine alternative Lösung $SubOPT = x$ die nur das zweite Geschenk wählt.

$$
\begin{aligned}
ALG_1 = 1 \Rightarrow & \frac{OPT}{ALG_1} = OPT \geq SubOPT = x \\
\Rightarrow & \frac{OPT}{ALG_1} \geq x
\end{aligned} \tag{7}
$$

Da x beliebig groß gewählt werden kann, kann auch $\frac{OPT}{ALG_1}$ beliebig groß werden. □

c)
ZZ. $\frac{OPT}{ALG_2}$ kann beliebig groß werden.

Beweis. Sei $k > 0$ beliebig, fest, sei $n = 3$.
Wähle die Geschenke so, dass für ein beliebiges $x > 0$ folgendes gilt:
$w(1) = 1,\ g(1) = \frac{k}{3x},\ w(2) = 4x,\ g(2) = 2k,\ g(3) = k,\ w(3) = x.$
Für die Effizienz der Geschenke gilt:
$e(1) = 3\frac{x}{k},\ e(2) = 2\frac{x}{k},\ e(3) = \frac{x}{k}$
Die Geschenkmenge $S = 1$, das erste Geschenk was nicht gewählt werden kann ist $x = 2$. Da x den Schlitten überlädt wählt der Algorithmus $S \Rightarrow ALG_2 = w(1) = 1$. Eine alternative Lösung $SubOPT$ wäre nur das Geschenk 3 $\Rightarrow SubOPT = w(3) = x$.

$$
\begin{aligned}
ALG_2 = 1 \Rightarrow & \frac{OPT}{ALG_2} = OPT \geq SubOPT = x \\
\Rightarrow & \frac{OPT}{ALG_2} \geq x
\end{aligned} \tag{8}
$$

Da x beliebig groß gewählt werden kann, kann auch $\frac{OPT}{ALG_2}$ beliebig groß werden. □

Das lößt zwar die Aufgabe nicht, zeigt aber das die Aufgabe auch nicht lösbar ist.