ETH zürich
Departement Informatik

Approximations- und Online-Algorithmen

Dr. Hans-Joachim Böckenhauer
Dr. Dennis Komm

https://courses.ite.inf.ethz.ch/appron22

Lösungsvorschläge – Blatt 8

Zürich, 28. April 2022

Lösung zu Aufgabe 12

Wir gehen davon aus, dass die beiden Wege nach links und rechts eine Gerade bilden. Die Weggabelung befindet sich an Position 0.

Solange er das Hotel noch nicht gefunden hat, geht der Wanderer nach und nach zu verschiedenen Positionen und schaut, ob sich das Hotel irgendwo auf der zurückgelegten Strecke befindet. Die i-te Position, die er ansteuert, ist Position $(-2)^i$ für $i \in \mathbb{N}$, also geht er der Reihe nach zu den Positionen

$$2^0, -2^1, 2^2, -2^3, 2^4, -2^5, \ldots .$$

Sei d der Abstand des Hotels zur Weggabelung, und sei $2^j < d \leq 2^{j+1}$ für ein $j \in \mathbb{N}$. Falls j gerade ist, liegt das Hotel im schlechtesten Fall an Position d; falls j ungerade ist, liegt es im schlechtesten Fall an Position $-d$. Damit ist die Strecke, die der Wanderer zurücklegt, bis er das Hotel gefunden hat,

$$2 \cdot (2^0 + 2^1 + 2^2 + \cdots + 2^{j+1}) + d = 2 \cdot \left(\sum_{i=0}^{j+1} 2^i \right) + d = 2 \cdot (2^{j+2} - 1) + d \, .$$

Da die optimale Lösung darin besteht, auf direktem Weg zum Hotel im Abstand d zu gehen, ist die kompetitive Güte des Algorithmus damit

$$\frac{2 \cdot (2^{j+2} - 1) + d}{d} < \frac{2 \cdot 2^{j+2}}{d} + 1 < \frac{8 \cdot 2^j}{2^j} + 1 = 9 \, .$$

Nehmen wir zudem an, dass die Sichtweite des Wanderers eine Längeneinheit beträgt. Der Wanderer kann dann zu Beginn an der Weggabelung einmal Ausschau nach dem Hotel halten und direkt darauf zugehen. Damit verhält sich der Wanderer im bislang nicht berücksichtigten Fall $d < 1$ optimal.

Der Wanderer muss mit dieser Strategie also höchstens 9-mal so weit gehen, wie die Distanz zum Hotel beträgt.

(bitte wenden)

Lösung zu Aufgabe 13

Der Definition von Paging aus der Vorlesung folgend nehmen wir wieder an, dass der Cache jedes Algorithmus mit $(1, \dots, k)$ initialisiert ist.

(a) Für ein beliebiges n betrachten wir die Eingabe $(k+1, k, k+1, k, \dots)$ der Länge n. Es ist offensichtlich, dass MAX einen Seitenfehler in jedem Zeitschritt verursacht. Auf der anderen Seite existiert ein optimaler Offline-Algorithmus OPT, der im ersten Zeitschritt beispielsweise die Seite mit dem Index 1 verdrängt und hiernach keine weiteren Kosten verursacht. Somit ist der kompetitive Faktor von MAX auf dieser Eingabe n.

(b) Analog zu (a) betrachten wir für ein beliebiges n die Eingabe $(k+1, 1, 2, 1, 2, \dots)$ der Länge n. Die Behauptung folgt sofort.

(c) In diesem Fall funktioniert wieder eine analoge Argumentation wie bei den vorangegangenen Aufgaben mit der Eingabe $(k+1, 1, k+1, 1, \dots)$.

(d) Auch hier funktioniert wieder eine analoge Argumentation mit der Eingabe $(k+1, s, k+1, s, \dots)$, wobei s die im ersten Schritt verdrängte Seite bezeichnet.