

Heimübungen zur Vorlesung  
**Datenstrukturen und Algorithmen**  
SS 2008  
Blatt 14

**AUFGABE 1** (6 Punkte):

Betrachten wir das *Problem der chromatischen Zahl*. Gegeben sei ein ungerichteter Graph  $G$ . Jeder Knoten in  $G$  hat maximal  $d$  Nachbarn. Gesucht ist die minimale Anzahl  $\chi(G)$  an Farben, die notwendig ist, um die Knoten des Graphen derart zu färben, dass adjazente Knoten unterschiedliche Farben haben. Diese minimale Anzahl wird die chromatische Zahl von  $G$  genannt. Es soll der Einfachheit halber Farben durch natürliche Zahlen dargestellt werden.

- a) Entwerfen Sie einen Greedy-Algorithmus mit Laufzeit  $\mathcal{O}(|V| + |E|)$  zur Bestimmung einer Färbung mit maximal  $d + 1$  Farben. Arbeiten Sie dabei vor allem den gierigen Charakter Ihrerer Strategie heraus. Hierbei ist *kein Pseudocode* verlangt.
- b) Beweisen Sie, dass der Algorithmus eine korrekte Färbung liefert.

*Hinweis:* Bisher ist es noch niemanden gelungen, einen Algorithmus zur Bestimmung der chromatischen Zahl anzugeben, welcher Laufzeit  $\mathcal{O}(n^c)$  mit  $n = |V| + |E|$  und konstantem  $c$  hat. Sollte Ihr Algorithmus derart schnell sein, so wird es wohl spezielle Graphen geben, auf denen die chromatische Zahl kleiner als die von Ihren Algorithmus bestimmte Anzahl ist. (Können Sie einen solchen Graphen für Ihren Algorithmus angeben?)

**AUFGABE 2** (6 Punkte):

Es ist ein Grubenunglück passiert, bei dem das ganze Schachtsystem einer Mine zusammengestürzt ist. Einige Minenarbeiter konnten sich in einen Hohlraum retten und hoffen jetzt auf schnelle Hilfe. Es muss von einem entfernten Schacht (*Start*) aus eine Verbindung zu den Verschütteten (*Ziel*) gegraben werden, weil unglücklicherweise eine direkte Bohrung von oben nicht möglich ist. Aus alten Unterlagen wissen Sie die Zusammensetzung des umliegenden Erdreichs und können berechnen, wie lange man für eine Grabung durch einen gewissen Bereich benötigen würde. Ihre Berechnungen ergeben folgende Tabelle:

<i>Start</i>	15	3	31	2
41	5	42	4	24
78	49	106	51	87
15	22	13	36	<i>Ziel</i>

Aus Sicherheitsgründen sind nur Grabungen nach unten oder in Richtung der Verschütteten, also in diesem Fall nach rechts, erlaubt.

*Beispiel:* Ein möglicher Weg ist:  $P = \{r, r, u, u, u, r, r\}$ . Dieser Weg würde  $15 + 3 + 42 + 106 + 13 + 36 = 215$  Zeiteinheiten benötigen.

- a) Formulieren Sie zuerst die rekursive Berechnung der optimal benötigten Zeit. Entwickeln sie daraufhin einen Algorithmus, der nach Eingabe der  $N \times M$ -Matrix mit dynamischer Programmierung sowohl die optimale Zeit als auch den Weg zu den verschütteten Minenarbeitern berechnet.
- b) Zeigen Sie die Korrektheit Ihres Algorithmus und bestimmen Sie die Laufzeit.

**AUFGABE 3** (6 Punkte):

Betrachten Sie nun das *maximale Rechte Teilsummenproblem*. Dabei ist als Eingabe eine Folge  $A = \langle a_1, a_2, \dots, a_n \rangle$  ganzer Zahlen  $a_i \in \mathbb{Z}$  gegeben. Gesucht ist die maximale Summe einer zusammenhängenden Teilfolge  $\langle a_l, a_{l+1}, \dots, a_k \rangle$  von  $A$ , die genau in  $a_k$  endet, also

$$R(k) = \max \left\{ \sum_{i=l}^k a_i \mid 1 \leq l \leq k \right\}$$

. *Tipp:*  $R(0) = 0$

- a) Finden Sie eine rekursive Formulierung für  $R(k)$ .
- b) Geben Sie einen effizienten Algorithmus an, der mit dynamischer Programmierung  $R(k)$  berechnet. Zeigen sie zudem die Korrektheit.

*Hinweis:* Der naive Algorithmus, welcher alle  $\mathcal{O}(n^2)$  möglichen Blöcke jeweils in Zeit  $\mathcal{O}(n)$  aufsummiert und dann das Maximum bestimmt, löst das Problem in Zeit  $\mathcal{O}(n^3)$ . Das Ergebnis Ihrer dynamischen Programmierung muss asymptotisch schneller bestimmt werden.

**AUFGABE 4** (6 Punkte):

Betrachten Sie als etwas komplexeren Fall das *maximale Teilsummenproblem*. Dabei ist als Eingabe eine Folge  $A = \langle a_1, a_2, \dots, a_n \rangle$  ganzer Zahlen  $a_i \in \mathbb{Z}$  gegeben. Gesucht ist die maximale Summe einer zusammenhängenden Teilfolge  $\langle a_l, a_{l+1}, \dots, a_r \rangle$  von  $A$ , also

$$\max \left\{ \sum_{i=l}^r a_i \mid 1 \leq l \leq r \leq n \right\}$$

.

- a) Bezeichne  $S(k)$  den Wert der maximalen Teilsumme in  $\langle a_1, a_2, \dots, a_k \rangle$ . Finden Sie eine rekursive Formulierung für  $S(k)$ . Verwenden Sie dabei gegebenenfalls  $R$  aus der dritten Aufgabe.
- b) Geben Sie einen effizienten Algorithmus an, der mittels dynamischer Programmierung die maximale Teilsumme von  $A$  berechnet. Zeigen Sie zudem die Korrektheit.

*Hinweis:* Der naive Algorithmus, welcher alle  $\mathcal{O}(n^2)$  möglichen Blöcke jeweils in Zeit  $\mathcal{O}(n)$  aufsummiert und dann das Maximum bestimmt, löst das Problem in Zeit  $\mathcal{O}(n^3)$ . Das Ergebnis Ihrer dynamischen Programmierung sollte asymptotisch schneller bestimmt werden.