12. Tutorium - Algorithmen I

Nina Zimbel

08.07.2015

Heute:

- Wiederholungsaufgaben
- Aufgabe zu dynamischer Programmierung
- **Diese Woche:** geht zur Wahl der verfassten Studierendenschaft (Fachschaftssprecher- und StuPa Wahl) Infos dazu gibt es zB im Wahleulenspiegel und im Wahlventil
- **Nächste Woche:** Wiederholung $\rightarrow$ sagt/schreibt mir Themen, die ihr gerne wiederholen wollt!

Währungs-Arbitrage

- Menge von Währungen C mit einem Umtauschkurs von r_{ij} (man erhält r_{ij} Einheiten von Währung j für eine Einheit von Währung i)
- Währungs-Arbitrage: es gibt eine Folge von elementaren Umtauschoperationen (Transaktionen), die mit einer Einheit einer Währung beginnt, und mit mehr als einer Einheit derselben Währung endet

Beschreibe einen Algorithmus, mit dem man für eine gegebenen Umtauschmatrix bestimmen kann, ob Währungs-Arbitrage möglich ist. Beweise die Korrektheit des Algorithmus.

Währungs-Arbitrage

- Menge von Währungen C mit einem Umtauschkurs von r_{ij} (man erhält r_{ij} Einheiten von Währung j für eine Einheit von Währung i)
- Währungs-Arbitrage: es gibt eine Folge von elementaren Umtauschoperationen (Transaktionen), die mit einer Einheit einer Währung beginnt, und mit mehr als einer Einheit derselben Währung endet

Beschreibe einen Algorithmus, mit dem man für eine gegebenen Umtauschmatrix bestimmen kann, ob Währungs-Arbitrage möglich ist. Beweise die Korrektheit des Algorithmus.

- Hinweis: $\log(xy) = \log x + \log y$, $\log(1) = 0$

Algorithmen-Entwurf: Häufigster Wert in einer Folge (SS09)

Gegeben sei ein Feld F von $n \geq 1$ Zahlen aus $1,\ldots,n^3$.
Gesucht ist die Zahl, die in F am häufigsten vorkommt. Wenn es mehrere Zahlen gibt, die gleichhäufig sind, kann eine beliebige gewählt werden.
Geben Sie einen möglichst "guten" Algorithmus an, der dieses Problem löst.

Bewertung:

- 6 Punkte für einen Algorithmus mit erwarteter Laufzeit $O(n \log n)$
- 7 Punkte für einen deterministischen Algorithmus mit Laufzeit $O(n \log n)$
- 8 Punkte für einen Algorithmus mit erwarteter Laufzeit $O(n)$
- 10 Punkte für einen deterministischen Algorithmus mit Laufzeit $O(n)$

Algorithmen-Entwurf: Kürzeste Wege (SS09)

Ein *1-2-Graph* sei ein gerichteter Graph mit Kantengewichten aus 1,2.

Beschreiben Sie einen Algorithmus zur Berechnung von kürzesten Wegen in 1-2-Graphen von einem Startpunkt s aus.

Geforderte Laufzeit: $O(m+n)$, wobei n Anzahl der Knoten und m Anzahl der Kanten

Die Beschreibung des Algorithmus soll textuell oder in kommentiertem Pseudocode erfolgen.

Dynamisches Programmieren

Definition: Zu einem ungerichteten Graph $G = (V,E)$ ist eine Menge $U \subset V$ eine *unabhängige Menge* genau dann, wenn keine Knoten aus U in G benachbart sind, d.h. $\forall u, v \in U : \{u, v\} \notin E$.

Defnition: Zu einem gerichteten Graph $G = (V,E)$ mit Knotengewichtsfunktion $c: V \to \mathbb{N}_{>0}$ ist eine Menge $U \subset V$ eine *maximale unabhängige Menge*, wenn U unabhängige Menge ist und unter allen unabhängigen Mengen das größtmögliche Gesamtgewicht hat, d.h. es exisitiert keine Unabhängige Menge $U' \subset V$ mit $\sum_{u \in U'} c(u) > \sum_{u \in U} c(u)$.

Festlegung: Im Folgenden sei $G := v_1 - v_2 - ... - v_k$ ein Pfad, d.h. G bestehe ausschließlich aus Kanten $\{v_l, v_{l+1}\}$ für $l \in \{1..k-1\}$. Für $l \leq k$ sei weiter $G_l := v_1 - ... - v_l$ der Teilpfad von G, der beim Knoten v_1 beginnt und bis einschließlich Knoten v_l geht. G_0 soll als leerer Pfad interpretiert werden.

a. Geben Sie zu dem abgebildeten Pfad die *maximale unabhängige Menge* U an, indem Sie die Knoten markieren, die zu U gehören. Geben Sie außerdem das Gesamtgewicht von U an. Die Knotengewichte stehen in den Knoten.

b. Es sei $m(l)$ das Gesamtgewicht einer *maximalen unabhängigen Menge* U_l auf G_l. Geben Sie eine Rekurrenz für $m(l)$ an, indem Sie auf die Werte der $m(i)$ für $i < l$ zurückgreifen!

c. Skizzieren Sie einen Algorithmus, der eine *maximale unabhängige Menge* U auf einem Pfad $G = v_1 - ... - v_k$ mit Knotengewichtsfunktion c **berechnet** und **ausgibt**. Der Algorithmus darf die Laufzeit $O(k)$ nicht überschreiten.

d. Skizzieren Sie kurz einen Linearzeit-Algorithmus, der eine maximale unabhängige Menge auf einem ungerichteten **Baum** mit Knotengewichtsfunktion c **berechnet** und **ausgibt**.