

9. Übungsblatt für Track ε zur Vorlesung Entwurf und Analyse von Algorithmen, WS 14/15

Abgabe: Bis Freitag, 16.01.2015, 12:00 Uhr, Kasten im Treppenhaus 48-6.

Basisaufgaben

B9.1: Wahrscheinlichste Pfade

3 Punkte

Gegeben sei ein markierter Digraph $G = (V, E, r)$ mit $r : E \rightarrow [0, 1] \subseteq \mathbb{R}$. Wir interpretieren die Kanten $e \in E$ als Leitungen eines Kommunikationsnetzwerks und deren Markierung als die *Zuverlässigkeit* der entsprechenden Verbindung (Kante), indem wir $r(e)$ als die Wahrscheinlichkeit dafür auffassen, dass eine Kommunikation über Verbindung e nicht scheitert. Dabei nehmen wir an, dass alle Wahrscheinlichkeiten unabhängig voneinander sind.

Entwerfen Sie einen Algorithmus, der in Zeit $\mathcal{O}(|V|^2)$ die verlässlichste Verbindung zwischen zwei gegebenen Knoten berechnet.

B9.2: Superstars

3 Punkte

Ein *Superstar* ist eine Person, die von allen anderen Personen gekannt wird, die selbst aber keine andere Person kennt.

- Wenn Sie eine Menge von n Personen und die Relation „ a kennt b “ als Digraph modellieren, also „ a kennt b “ $\iff (a, b) \in E$, wie können Sie dann einen Superstar charakterisieren?
- Sei der Digraph aus Teilaufgabe a) als $n \times n$ Adjazenzmatrix A gegeben. Entwerfen Sie einen Algorithmus, der bei Eingabe A in Zeit $o(n^2)$ entscheidet, ob die durch A modellierte Population einen Superstar besitzt.

Begründen Sie die Korrektheit Ihres Algorithmus und weisen Sie eine dem Zwecke der Aufgabenstellung dienliche Laufzeitschranke nach.

B9.3: Durchmesser von Graphen

3 Punkte

Zur Erinnerung: Der Durchmesser eines (Di)Graphen $G = (V, E)$ ist definiert als

$$d(G) := \max \{ \text{dist}(u, v) \mid u, v \in V \} ,$$

wobei $\text{dist}(u, v)$ die Länge eines kürzesten Weges von u nach v ist.

Entwerfen Sie einen Algorithmus, der in Zeit $\mathcal{O}(|V|^3)$ den Durchmesser eines beliebigen (Di)Graphen findet. Beweisen Sie die Korrektheit und Laufzeitabschätzung Ihres Entwurfs.

B9.4: Minimal Bottleneck Spanning Trees

3 Punkte

Sei $G = (V, E, g)$ ein gewichteter Graph und $T = (V, E')$ ein Spannbaum von G . Wir bezeichnen die Kante

$$e^*(T) := \arg \max_{e \in E'} g(e)$$

als *Engstelle* von T und entsprechend

$$g^*(T) := \max_{e \in E'} g(e)$$

als *Engstellengewicht* von T . Unter allen Spannbäumen von G nennen wir die mit minimalen Engstellengewichten $g^*(_)$ auch *engstellenoptimale Spannbäume*.

- a) Seien G ein (beliebiger) Graph und T ein Spannbaum von G . Zeigen oder widerlegen Sie:

$$T \text{ minimaler Spannbaum} \implies T \text{ engstellenoptimaler Spannbaum.}$$

- b) Seien G ein (beliebiger) Graph und T ein Spannbaum von G . Zeigen oder widerlegen Sie:

$$T \text{ engstellenoptimaler Spannbaum} \implies T \text{ minimaler Spannbaum.}$$

- c) Geben Sie einen Algorithmus an, der das folgende Problem in Zeit $\mathcal{O}(|V| + |E|)$ löst.

Eingabe: Ein Graph $G = (V, E)$ mit Gewichtungsfunktion $g : E \rightarrow \mathbb{R}^+$ sowie $B \in \mathbb{N}$.

Frage: Gibt es einen Spannbaum T von G mit Engstellengewicht $g^*(T) \leq B$?

Aufbauaufgaben

A9.1: Bellman-Ford

3 + 2 + 2 Punkte

Sei $G = (V, E)$ mit $V = [1..n]$ ein zusammenhängender Graph mit Markierungsfunktion $g : E \rightarrow \mathbb{R}$ – man beachte den kleinen, aber wesentlichen Unterschied zu den bisher betrachteten Markierungsfunktionen – und $s \in V$ fest.

Der notationellen Einfachheit definieren wir eine auf beliebige Knotenpaare erweiterte Gewichtsfunktion $h : V \times V \rightarrow \mathbb{R} \cup \{\infty\}$ so:

$$h(i, j) := \begin{cases} 0 & , i = j \\ g(\{i, j\}) & , i \neq j \wedge \{i, j\} \in E \\ \infty & , \text{sonst} \end{cases} \quad (1)$$

Nun sei für $i, j \in [1..n]$ folgende Rekursionsgleichung gegeben:

$$B_s(i, j) = \begin{cases} h(s, j) & , i = 1 \\ \min_{k \in [1..n]} (B_s(i-1, k) + h(k, j)) & , \text{sonst} \end{cases} \quad (2)$$

- Zeigen Sie: Wenn G keine Kreise mit negativem Gesamtgewicht hat, so löst B_s das SSSPP für G und s , das heißt $B_s(n, t)$ gibt die Länge eines kürzesten Weges von s nach t an.
- Geben Sie einen Algorithmus an, der B_s für beliebige Graphen (und s) in Polynomialzeit berechnet, das heißt

$$[B_s(n, 1), \dots, B_s(n, n)]$$

zurückgibt. Bestimmen Sie die Laufzeit Ihres Algorithmus.

- Können Sie Ihren Algorithmus aus b) so modifizieren, dass er ohne (asymptotischen) Mehraufwand *feststellt*, ob G negative Kreise enthält?

A9.2: Kantengewichte in MSTs

5 Punkte

Sei $G = (V, E, g)$ ein gewichteter Graph und seien $T_1 = (V, E_1)$ und $T_2 = (V, E_2)$ minimale Spannbäume von G . Wir bezeichnen mit

$$\text{count}(A, c) = |\{e \in A \mid g(e) = c\}|$$

die Anzahl Kanten eines bestimmten Gewichts c in einer Kantenmenge $A \subseteq E$.

Zeigen oder widerlegen Sie: Für alle $c \in \mathbb{R}$ gilt

$$\text{count}(E_1, c) = \text{count}(E_2, c) .$$