Lösungen von Übungsblatt 7

Algorithmen (WS 2018, Ulrike von Luxburg)

Lösungen zu Aufgabe 1

a) EG und FH / DF / CE und BG / EF / IJ / AB / AI

b)

Knoten	A	B	C	D	E	F	G	H	I	J
Parent	E	E	E	F	-	E	E	F	E	I

c) AB / BG / EG / CE / EF / FH / DF / AI / IJ

d) Sei $n = |V|$ und $m = |E|$.

Die anfängliche Initialisierung (Programmzeilen 1-5) hat Zeitaufwand $t_1 \in \mathcal{O}(n)$.

Das Erstellen der Priority-Queue (Programmzeile 6) hat Zeitaufwand $t_2 \in \mathcal{O}(n \log n)$.

Den Aufwand der beiden verschachtelten Schleifen (äußere while-Schleife und innere for-Schleife) schätzen wir nicht mit der üblichen Vorgehensweise (Anzahl Durchläufe äußerer Schleife $\times$ Anzahl Durchläufe innerer Schleife) ab, sondern separat nach Programmzeilen:

Die Programmzeilen 7+8 tragen $t_3 = \mathcal{O}(n \log n)$ zur Gesamtzeit bei, da die while-Schleife (Zeile 7) genau n Mal durchlaufen wird, und darin jedesmal (Zeile 8) in Zeit $\mathcal{O}(\log n)$ das nächste Element aus der Priority-Queue geholt wird.

Die restlichen Programmzeilen (9-13) betrachten wir lösgelöst von der äußeren Schleife wie folgt: Die innere for-Schleife wird *insgesamt* (also kumuliert über alle Durchläufe der äußeren while-Schleife) m Mal durchlaufen, wobei in jedem Durchlauf höchstens Aufwand $\mathcal{O}(\log n)$ für die decreaseKey-Operation anfällt. Damit tragen die Programmzeilen 9-13 insgesamt $t_4 = \mathcal{O}(m \log n)$ zur Gesamtlaufzeit bei.

Zusammengenommen erhalten wir somit als Gesamtlaufzeit $t_1 + t_2 + t_3 + t_4 \in \mathcal{O}(n \log n + m \log n) = \mathcal{O}(m \log n)$, wobei die letzte Gleichheit gilt, weil der Graph zusammenhängend ist, und daher $m \geq n - 1$, also $n \in \mathcal{O}(m)$.

Lösungen zu Aufgabe 2

a) G' habe also minimales Gewicht. Es bleibt zu zeigen, dass G' dann bereits ein Baum ist. Angenommen, G' wäre *kein* Baum. Dann gäbe es (da nach Voraussetzung zusammenhängend) einen Zyklus. Von diesem könnte man eine beliebige Kante entfernen (mit positivem Gewicht), und man erhielte einen immernoch zusammenhängenden Teilgraphen auf V. Allerdings mit echt geringerem Gewicht als G', im Widerspruch dazu, dass bereits G' minimal war. Also muss G' bereits ein Baum sein.

b) Wir zeigen "Wenn jeder Z-Schnitt genau eine Kante hat, dann ist G ein Baum" per Kontraposition: Angenommen also, G wäre *kein* Baum. Dann muss G (da nach Voraussetzung zusammenhängend) einen Zyklus enthalten. Wir wollen nun einen Z-Schnitt konstruieren, der mehr als eine Kante enthält. Sei v irgendein Knoten dieses Zyklus. Wir können den Graphen G wie folgt in zwei Mengen zerteilen. Zunächst entfernen wir v aus dem Graphen. Sei die Menge A nun die Menge aller Knoten, die von den Knoten im Rest des Kreises (mit Tiefensuche) aus erreicht werden können. Sollten wir dabei auch wieder v erreichen, nehmen wir v nicht in die Menge A. Diese Menge A bildet schließlich eine Zusammenhangskomponente. Sei die Menge B nun der Knoten v und die Menge aller Knoten, die wir von v aus erreichen können (wenn wir wieder den ursprünglichen Graphen betrachten) und noch nicht in A enthalten sind. Damit haben wir den Graphen in zwei Zusammenhangskomponenten geteilt. Der Schnitt dazwischen ist also ein Z-Schnitt. Weil aber der Kreis, abgesehen von v vollständig in A liegt, führt eine Hin- und eine Rückkante zu v. Der Z-Schnitt enthält also mindestens 2 Kanten.

Warum dieser Aufwand? Wenn wir nur den Schnitt $\{v\} \cup (V \setminus \{v\})$ betrachten, enthält dieser zwar mindestens 2 Kanten aufgrund des Kreises, aber dieser Schnitt ist nicht zwingend ein Z-Schnitt. Möglicherweise ist der Graph auf $(V \setminus \{v\})$ nicht mehr zusammenhängend. Doch das ist eine Eigenschaft eines Z-Schnitt. Die Menge B kann also bildlich als der Rattenschwanz

von v gesehen werden, sodass wir schließlich auf zwei zusammenhängende Teile des Graphen kommen.

Für die Rückrichtung ist zu zeigen: "Wenn G ein Baum ist, dann muss jeder Z-Schnitt genau eine Kante haben." Da G zusammenhängend ist, muss jeder Z-Schnitt mindestens eine Kante enthalten. Zudem kann kein Z-Schnitt $A \cup B$ mehr als zwei Kanten haben, da es sonst einen Kreis im Graphen gäbe, der über eine Kante des Z-Schnitts von A nach B und über eine andere von B nach A führt.

Lösungen zu Aufgabe 3

Im Folgenden bezeichnen wir das Gewicht eines beliebigen Spannbaumes R von G bezüglich der ursprünglichen Kantengewichte mit

$$w_G(R)$$

und bezüglich der veränderten Kantengewichte (tatsächlich wird nur das Gewicht einer Kante e verändert) mit

$$\widetilde{w}_G(R).$$

Da T minimaler Spannbaum von G bezüglich der ursprünglichen Kantengewichte ist, gilt

$$w_G(T) \leq w_G(R) \tag{1}$$

für jeden Spannbaum R von G.

a) Falls das Gewicht einer Kante e erhöht wird, gilt für jeden Spannbaum R von G

$$w_G(R) \leq \widetilde{w}_G(R),$$

wobei Gleichheit genau dann gilt, falls e keine Kante von R ist. Es folgt daher für jeden Spannbaum R von G wegen $e \in E \setminus E'$ (d.h. e ist keine Kante von T) und (1)

$$\widetilde{w}_G(T) = w_G(T) \leq w_G(R) \leq \widetilde{w}_G(R).$$

Dies zeigt, dass T auch minimaler Spannbaum bezüglich der veränderten Kantengewichte ist.

b)
- 1. Fall: $e \in E'$

  Sei $w(e)$ das ursprüngliche Gewicht von e und $\widetilde{w}(e)$ das verringerte Gewicht. Es gilt also $-w(e) + \widetilde{w}(e) < 0$. Nun gilt für jeden Spannbaum R

  $$\widetilde{w}_G(R) = \begin{cases} w_G(R) - w(e) + \widetilde{w}(e) & \text{falls } R \text{ die Kante } e \text{ enthält} \\ w_G(R) & \text{sonst.} \end{cases}$$

  Damit gilt im Fall $e \in E'$ wegen (1)

  $$\widetilde{w}_G(T) = w_G(T) - w(e) + \widetilde{w}(e) \leq w_G(R) - w(e) + \widetilde{w}(e) \leq \widetilde{w}_G(R)$$

  für jeden Spannbaum R, was zeigt, dass T auch minimaler Spannbaum bezüglich der veränderten Kantengewichte ist und wir nichts weiter zu tun brauchen.

- 2. Fall: $e \notin E'$

  Berechne den minimalen Spannbaum im veränderten Graphen wie folgt: Füge e zu T hinzu (d.h. betrachte den Graphen $T+e$). $T+e$ enthält einen Zyklus. Entferne aus diesem Zyklus die Kante mit dem höchsten Gewicht.

  Begründung: Dass der so konstruierte Graph $\widetilde{T}$ ein Spannbaum von G ist, ist klar. Um zu zeigen, dass er minimaler Spannbaum bezüglich der veränderten Kantengewichte ist, zeigen wir, dass Kruskal's Algorithmus angewandt auf den veränderten Graphen genau $\widetilde{T}$ erzeugt. Wir vergleichen dazu die Anwendung von Kruskal's Algorithmus auf den ursprünglichen Graphen mit jener auf den veränderten Graphen.

Kruskal's Algorithmus durchläuft die nach Gewicht sortierten Kanten und fügt eine Kante zur Menge A hinzu, wenn ihre Hinzunahme keinen Zyklus erzeugt. Wir nehmen an, dass bei der Anwendung von Kruskal's Algorithmus auf den ursprünglichen Graphen die Kanten so sortiert sind, dass der Algorithmus den minimalen Spannbaum T berechnet (dies ist nur im Falle von Kanten mit gleichem Gewicht relevant; man überlegt sich leicht, dass eine solche Sortierung existiert). Bei Anwendung von Kruskal's Algorithmus auf den veränderten Graphen seien die Kanten genau gleich sortiert, nur dass die Kante e genau so weit nach vorne gerückt wird wie mindestens nötig (falls es mehrere Kanten mit Gewicht $\widetilde{w}(e)$ gibt, reihen wir e also als letzte dieser Kanten in die Sortierung ein, wobei $\widetilde{w}(e)$ wieder das verringerte Gewicht der Kante e bezeichne).

Im veränderten Graphen werden nun alle Kanten mit Gewicht kleiner/gleich $\widetilde{w}(e)$ genauso hinzugefügt wie im ursprünglichen Graphen. Sobald die Kante e betrachtet wird, können zwei Fälle auftreten:

i. e wird nicht hinzugefügt

Dann enthält A (die Menge der bisher hinzugefügten Kanten) zum Zeitpunkt der Betrachtung von e schon so viele Kanten, dass die Hinzunahme von e einen Zyklus erzeugen würde. Die Kanten von A sind alle in T enthalten. Damit ist aber e die schwerste Kante im Zyklus in $T + e$ und es gilt $T = \widetilde{T}$. Alle Kanten, die nach e betrachtet werden, werden genauso wie im ursprünglichen Graphen zu A hinzugenommen oder auch nicht. Damit erzeugt Kruskal's Algorithmus angewandt auf den veränderten Graphen $T = \widetilde{T}$.

ii. e wird hinzugefügt

Wieder enthält A bei der Betrachtung von e genau die bisher betrachteten Kanten, die auch in T enthalten sind. Dann wird e hinzugefügt. Es gibt also mindestens eine Kante im Zyklus von $T+e$, die schwerer als e ist. Die schwerste dieser Kanten bezeichnen wir mit e_s. Kanten, die nach e und vor e_s betrachtet werden, werden genau dann zu A hinzugefügt, wenn sie auch bei Anwendung von Kruskal's Algorithmus auf den ursprünglichen Graphen hinzugefügt werden. Dann wird die Kante e_s betrachtet und nicht hinzugenommen, da ihre Hinzunahme einen Zyklus erzeugen würde. Kanten, die nach e_s betrachtet werden, werden genau dann zu A hinzugefügt, wenn sie auch bei Anwendung von Kruskal's Algorithmus auf den ursprünglichen Graphen hinzugefügt werden. Damit erzeugt Kruskal's Algorithmus angewandt auf den veränderten Graphen $T + e - e_s = \widetilde{T}$.

Anmerkung: Alternativ kann die Korrektheit des angegebenen Verfahrens sehr einfach gezeigt werden, wenn man folgende Charaktersierung minimaler Spannbäume aus der Graphentheorie kennt:
Ein Spannbaum $T = (V, E')$ eines gewichteten Graphen $G = (V, E)$ ist ein minimaler Spannbaum genau dann, wenn für jede Kante $e \in E \setminus E'$ gilt: e ist eine Kante mit maximalem Gewicht in dem eindeutigen Zyklus, der entsteht, wenn man e zu T hinzufügt.

Lösungen zu Aufgabe 4

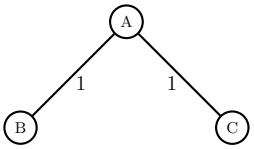
a) Wir nehmen an, G hat die Eigenschaft, dass für jeden Schnitt eine eindeutig bestimmte Schnittkante mit minimalem Gewicht existiert, und es gäbe zwei verschiedene minimale Spannbäume T und T'. Dann enthält T eine Kante $e = (u, v)$, die nicht in T' enthalten ist. Sei S die Menge aller Knoten von G, die in $T - e$ (das ist jener Graph, der entsteht, wenn wir die Kante e aus T entfernen) von v aus über einen Pfad erreichbar sind. Es ist $(S, V \setminus S)$ ein Schnitt von G [da T ein Spannbaum ist, stimmt dabei $V \setminus S$ mit der Menge jener Knoten von G überein, die in $T - e$ von u aus über einen Pfad erreichbar sind].

Sei $A = E(S, V \setminus S)$ die Menge aller Schnittkanten von G bezüglich des eben definierten Schnitts. Nach Voraussetzung existiert eine eindeutig bestimmte Kante $e_{min} \in A$ mit minimalem Gewicht. Wir zeigen, dass $e_{min} = e$ gelten muss: Angenommen $e_{min} \neq e$, dann ist $T - e + e_{min}$ (also jener Graph, der entsteht, wenn wir in T die Kante e gegen e_{min} austauschen) ein Spannbaum, dessen Gewicht geringer ist als jenes von T — Widerspruch.

Sei nun p' ein einfacher Pfad in T', der u mit v verbindet. Der Pfad p' enthält mindestens eine Schnittkante $e' \in A$. Da $e = e_{min}$ nicht in T' enthalten ist, ist $e' \neq e_{min}$ und das Gewicht

von e' größer als jenes von e_{min}. Damit ist aber $T' - e' + e_{min}$ ein Spannbaum mit geringerem Gewicht als T' — Widerspruch.

b) Betrachte folgendes offenes Dreieck:



Offensichtlich ist das Dreieck selbst sein einziger/eindeutiger (minimaler) Spannbaum. Der Schnitt $(\{A\}, \{B, C\})$ hat jedoch keine eindeutig bestimmte Schnittkante mit minimalem Gewicht.

Lösungen zu Aufgabe 5

Wir konstruieren den zugehörigen ungerichteten Graphen auf den Knoten $V = \{1, \ldots, 7\}$ mit den angegebenen gewichteten Kanten. Das Ziel ist dann, einen minimalen Spannbaum zu finden. Mittels Kruskal erhalten wir etwa die Kantenmenge $\{(2,3), (6,7), (2,5), (5,6), (4,6), (1,4)\}$, welche einen minimalen Spannbaum mit Gesamtgewicht 40 liefert. Die minimalen Kosten belaufen sich somit auf 40.000 EUR.