Prof. Dr. Ir. Joost-Pieter Katoen

Sebastian Junges, Benjamin Kaminski, David Korzeniewski, Tim Quatmann

Übung 10

Hinweise:

- Die Lösungen müssen bis **Donnerstag, den 5. Juli um 16:00 Uhr** in den entsprechenden Übungskasten eingeworfen werden. Sie finden die Kästen am Eingang Halifaxstr. des Informatikzentrums (Ahornstr. 55).
- Die Übungsblätter **müssen** in Gruppen von je 3 Studierenden aus der gleichen Kleingruppenübung abgegeben werden.
- Drucken Sie ggf. digital angefertigte Lösungen aus. Abgaben z.B. per Email sind nicht zulässig.
- Namen und Matrikelnummer sowie die **Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. Abgaben, die aus mehreren Blättern bestehen **müssen geheftet bzw. getackert** werden! Die **Gruppennummer muss sich auf der ersten Seite oben links** befinden.
- **Bei Nichtbeachten der obigen Hinweise müssen Sie mit erheblichen Punktabzügen rechnen!**

Aufgabe 1 (Ein Graphalgorithmus): (5 + 20 = 25 Punkte)

Gegeben sei der folgende Algorithmus:

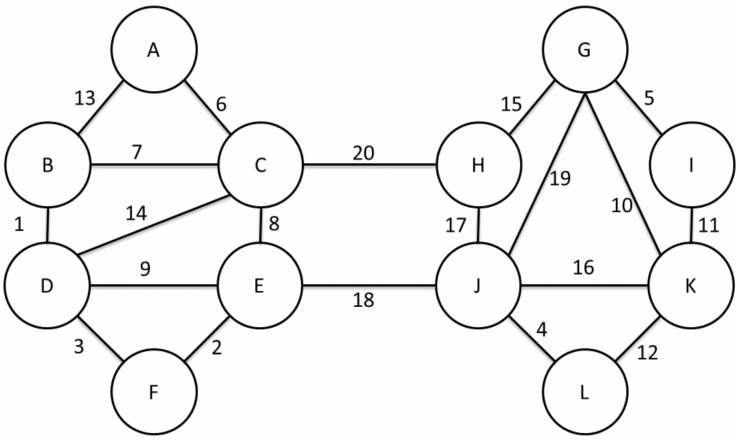
Eingabe: zusammenhängender gewichteter, ungerichteter Graph (V, E, W)

```
Erstelle ein Array E_s aller Kanten absteigend sortiert nach Kantengewicht
for i = 0 to |E| − 1:
    e := E_s[i]
    Lösche Kante e aus E
    Falls (V, E, W) nicht zusammenhängend:
        Füge Kante e zu E hinzu
```

Ausgabe: (V, E)

a) Was berechnet der obige Algorithmus? Begründen Sie ihre Antwort.

b) Führen Sie den obigen Algorithmus auf dem untenstehenden Graphen aus. Geben Sie den Graphen (V, E) nach jeder Iteration der Schleife an.



Aufgabe 2 (Minimale Spannbäume): (15 Punkte)

Beweisen oder widerlegen Sie: Der minimale Spannbaum eines Graphen, dessen Kantengewichte alle paarweise verschieden sind, ist eindeutig.

Aufgabe 3 (Single-Source Single-Target Shortest Path): (25 Punkte)

Betrachten wir zunächst folgende Definition:
Ein Graph mit Koordinaten ist ein 4-Tupel (V, E, W, K) wobei

- (V, E, W) ein gewichteter Graph ist, und
- $K : V \to \mathbb{R} \times \mathbb{R}$ jeden Knoten auf eine Koordinate abbildet.

Zusätzlich nehmen wir an, dass $W(v_1, v_2) \geq \|K(v_1) - K(v_2)\|_2$.

Hinweise:

- Informell bedeutet dies, dass das Gewicht zwischen zwei Knoten mindestens die Länge der Luftlinie ist.

Wir suchen den kürzesten Weg zwischen s und t.
Dazu definieren wir unten zusätzlich eine Funktion $h : V \to \mathbb{R}$.
Wir passen den Dijkstra-Algorithmus so an, dass

- `ExtractMin` nun $\arg\min_{v \in Q} h(v) + \text{dist}[v]$ auswählt, und
- sobald `ExtractMin` t auswählt, terminiert der Algorithmus.

Beweisen oder widerlegen Sie:

a) Für $h(v) = \|K(v) - K(t)\|_1$ berechnet der angepasste Dijkstra-Algorithmus einen kürzesten Pfad von s nach t.

b) Für $h(v) = \|K(v) - K(t)\|_{\max}$ berechnet der angepasste Dijkstra-Algorithmus einen kürzesten Pfad von s nach t.

Hinweise:

- $\|(x, v)\|_1 = |x| + |y|$
- $\|(x, y)\|_2 = \sqrt{x^2 + y^2}$
- $\|(x, y)\|_{\max} = \max\{|x|, |y|\}$

Aufgabe 4 (Schiebepuzzle): (10 Punkte)

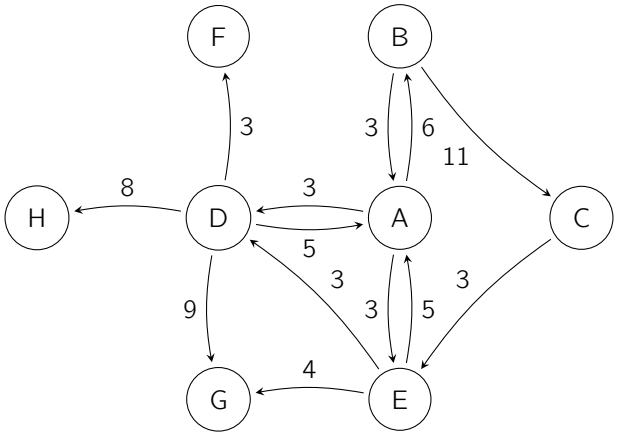
Betrachten Sie das $n-$Schiebepuzzel, wobei ein Quadrat der Größe n mal n gegeben ist. An fast jeder Position in dem Quadrat befindet sich eines der $n^2 - 1$ Puzzelteile, eindeutig beschriftet mit einer Zahl aus dem Interval 1 bis $n^2 - 1$. Es gibt genau eine leere Position. Ein Puzzelteil, das horizontal oder vertikal adjazent zu der leeren Position ist, kann auf die leere Position geschoben werden. Ziel des Puzzles ist es, die Teile an eine vordefinierte Position zu schieben, und zwar in möglichst wenigen Zügen.
In folgendem Beispiel ($n = 3$) ist die leere Position in der zweiten Zeile und der ersten Spalte. Die möglichen Aktionen ergeben sich durch das Verschieben des Puzzelteils 2, 7 oder 4.

2	3	6
	7	8
4	1	5

Formalisieren Sie dieses Problem mit Hilfe von Graphen, und beschreiben Sie, welcher Algorithmus aus der Vorlesung zur Lösung hilfreich ist. Beschreiben Sie zusätzlich, wie groß der Graph in Abhängigkeit von n ist.

Aufgabe 5 (Dijkstra-Algorithmus): (10 Punkte)

Betrachten Sie den folgenden Graphen:

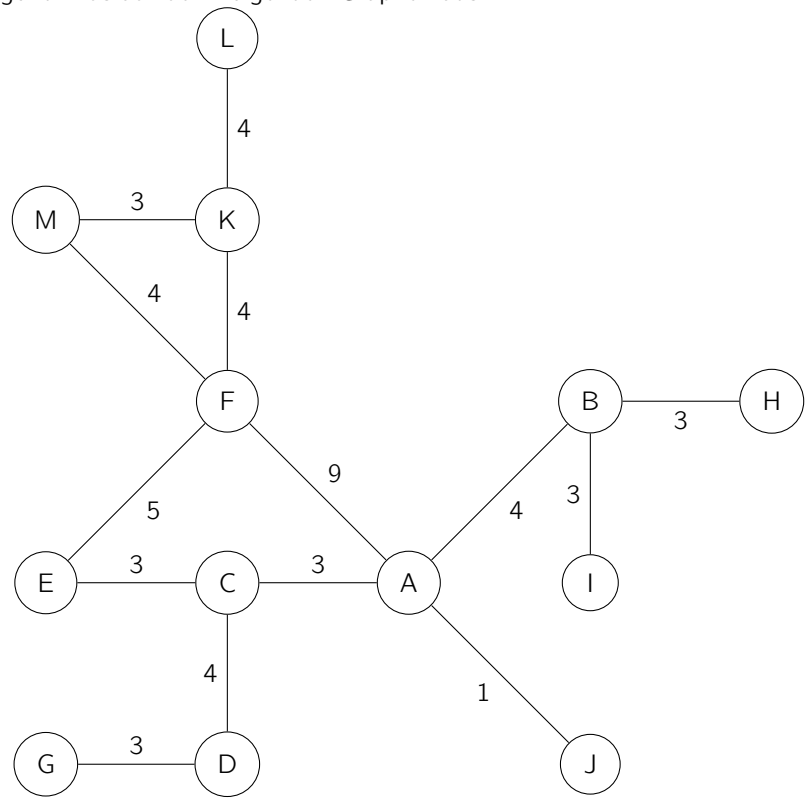


Führen Sie den *Dijkstra* Algorithmus auf diesem Graphen mit dem *Startknoten A* aus. Falls mehrere Knoten für die nächste Iteration zur Wahl stehen, werden die Knoten dabei in alphabetischer Reihenfolge betrachtet. Füllen Sie dazu die nachfolgende Tabelle aus:

Knoten	A						
B							
C							
D							
E							
F							
G							
H							

Aufgabe 6 (Prim-Algorithmus): (15 Punkte)

Führen Sie Prims Algorithmus auf dem folgenden Graphen aus.



Der Startknoten hat hierbei den Schlüssel A. Geben Sie dazu *vor* jedem Durchlauf der äußeren Schleife an,

1. welche Kosten die Randknoten haben (d. h. für jeden Knoten `v` in `pq` die Priorität von `v`, wobei ∞ angibt, dass der entsprechende Knoten noch nicht zum Randbereich gehört)
2. und welchen Knoten `pq.getMin()` wählt, indem Sie den Kosten-Wert des gewählten Randknoten in der Tabelle unterstreichen (wie es in der ersten Zeile bereits vorgegeben ist).

Geben Sie zudem den vom Algorithmus bestimmten minimalen Spannbaum an.

#Iteration	A	B	C	D	E	F	G	H	I
1	<u>0</u>	∞	∞	∞	∞	∞	∞	∞	∞
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									

J	K	L	M
∞	∞	∞	∞

Minimaler Spannbau