Prof. Dr. Leif Kobbelt
Thomas Ströder, Fabian Emmes
Sven Middelberg, Michael Kremer

Informatik 8
RWTH Aachen

Datenstrukturen und Algorithmen (SS 2013)

Übungsblatt 10

Abgabe: Montag, **08.07.2013**, 14:00 Uhr

- Die Übungen sollen in Gruppen von zwei bis drei Personen bearbeitet werden.
- Schreiben Sie die Namen jedes Gruppenmitglieds sowie alle Matrikelnummern auf die abgegebenen Lösungen.
- Schreiben Sie die Namen jedes Gruppenmitglieds sowie alle Matrikelnummern auch in die Quellcode-Dateien.
- Geben Sie Ihre Lösungen am **Anfang** der Globalübung, montags, 14:00 Uhr, ab.
- Schicken Sie den jeweiligen Quellcode bitte per **E-Mail** direkt an Ihre/n Tutor/in.
- Geben Sie außerdem den ausgedruckten Quellcode zusammen mit den schriftlichen Lösungen ab.
- Zu spät abgegebene Lösungen werden nicht bewertet.
- Sofern nicht anders gefordert, müssen alle Lösungen und Zwischenschritte kommentiert werden.

Prof. Dr. Leif Kobbelt
Thomas Ströder, Fabian Emmes
Sven Middelberg, Michael Kremer

Informatik 8
RWTH Aachen

Aufgabe 1 (*Maximaler Abstand in Graphen* [**6 Punkte**])

Sei $G = (V, E)$ ein Baum mit Gewichtsfunktion $w : E \to R^+$. Gesucht sind zwei Knoten $u, v \in V$, deren Abstand maximal ist (d.h. die Kosten des kürzesten Pfades von u nach v sollen maximal sein). Beschreiben und begründen Sie einen Algorithmus, der die beiden Knoten u und v und den Abstand von u und v in Laufzeit $O(|V|)$ bestimmt.

Lösungsvorschlag

G ist ein Baum $\Rightarrow$ G ist ungerichtet, zusammenhängend und kreisfrei.
Idee: Dynamische Programmierung, bearbeite alle Teilbäume, ausgehend von den Blättern. Speichere pro Teilbaum mit Wurzel $v \in V$:

a) längsten Pfad p_v von v zu einem Blatt,

b) längsten Pfad P_v im Teilbaum mit Wurzel v.

Initialisiere alle Blätter mit $P_v = p_v = \emptyset$.
Sei nun v ein innerer Knoten mit t_v bereits bearbeiteten Teilbäumen mit Wurzeln u_i, $1 \leq i \leq t_v$.

1. Berechne den Pfad p_v mit Länge
$$\max_i \{w(p_{u_i}) + w(v, u_i)\} \quad .$$
2. Berechne analog dazu den zweitlängsten Pfad p'_v von v zu einem Blatt.
3. Berechne längsten Pfad zwischen zwei Blättern, der v enthält. Dieser setzt sich aus den Pfaden p_v und p'_v zusammen. Vergleiche mit den längsten Pfaden der Teilbäume und finde
$$\max \left\{ \max_i \{w(P_{u_i})\}, w(p_v) + w(p'_v) \right\} \quad .$$
Der dazugehörige Pfad ist P_v.

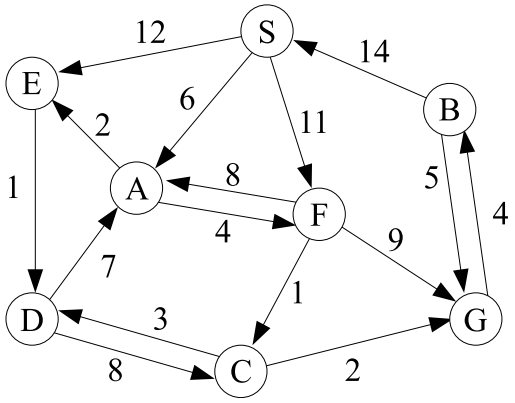
An der Wurzel des Baumes ist dann der unter 3. berechnete Pfad der insgesamt längste.
Laufzeit: Jeder Knoten $v \in V$ wird genau einmal besucht. Dabei müssen jeweils t_v Teilbäume betrachtet werden. Die Berechnung der Pfade p_v, p'_v und P_v benötigt einen Aufwand von jeweils $O(t_v)$. Insgesamt beträgt der Aufwand also
$$\sum_{v \in V} O(t_v) = O(|E|) = O(|V| - 1) = O(|V|) \quad .$$

Prof. Dr. Leif Kobbelt
Thomas Ströder, Fabian Emmes
Sven Middelberg, Michael Kremer

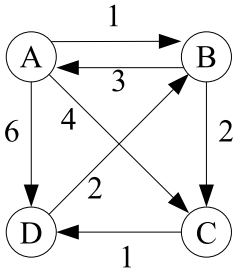
Informatik 8
RWTH Aachen

Aufgabe 2 (*Kürzeste Pfade* [**4 Punkte**])

1. Wenden Sie Dijkstras Algorithmus auf den folgenden Graphen G mit Startknoten S an. Geben Sie die Werte aller oberen Schranken nach jedem Durchlauf der Hauptschleife von Dijkstras Algorithmus in Form einer Tabelle an. Die Tabelle enthält also eine Spalte für jeden Knoten $v \in G$. Pro Iteration soll eine Zeile hinzugefügt werden, die die aktuellen oberen Schranken $\delta(S, v)$ enthält. Markieren Sie dabei in jeder Zeile den Knoten, den Sie auswählen. [2 Punkte]



2. Wenden Sie den Algorithmus von Floyd-Warshall auf den unten angegebenen Graphen an, um die Längen aller kürzestens Pfade zu berechnen. Geben Sie die Distanzmatrix nach jedem Durchlauf der Hauptschleife an. [2 Punkte]



Prof. Dr. Leif Kobbelt
Thomas Ströder, Fabian Emmes
Sven Middelberg, Michael Kremer

Informatik 8
RWTH Aachen

Lösungsvorschlag

1. Dijkstra:

	S	A	B	C	D	E	F	G	Vertex
0	0	6	∞	∞	∞	12	11	∞	S
1	0	6	∞	∞	∞	8	10	∞	A
2	0	6	∞	∞	9	8	10	∞	E
3	0	6	∞	17	9	8	10	∞	D
4	0	6	∞	11	9	8	10	19	F
5	0	6	∞	11	9	8	10	13	C
6	0	6	17	11	9	8	10	13	G
7	0	6	17	11	9	8	10	13	B

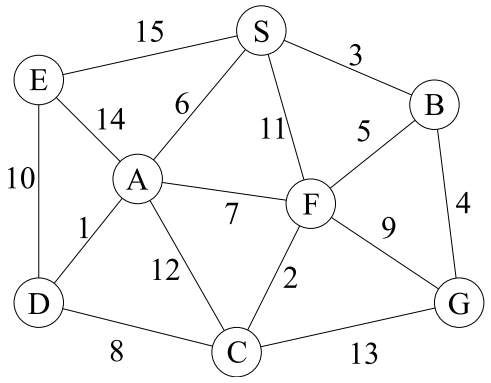
2. Floyd-Warshall:

$$D_0 = \begin{bmatrix} 0 & 1 & 4 & 6 \\ 3 & 0 & 2 & \infty \\ \infty & \infty & 0 & 1 \\ \infty & 2 & \infty & 0 \end{bmatrix}, D_1 = \begin{bmatrix} 0 & 1 & 4 & 6 \\ 3 & 0 & 2 & 9 \\ \infty & \infty & 0 & 1 \\ \infty & 2 & \infty & 0 \end{bmatrix}, D_2 = \begin{bmatrix} 0 & 1 & 3 & 6 \\ 3 & 0 & 2 & 9 \\ \infty & \infty & 0 & 1 \\ 5 & 2 & 4 & 0 \end{bmatrix},$$

$$D_3 = \begin{bmatrix} 0 & 1 & 3 & 4 \\ 3 & 0 & 2 & 3 \\ \infty & \infty & 0 & 1 \\ 5 & 2 & 4 & 0 \end{bmatrix}, D_4 = \begin{bmatrix} 0 & 1 & 3 & 4 \\ 3 & 0 & 2 & 3 \\ 6 & 3 & 0 & 1 \\ 5 & 2 & 4 & 0 \end{bmatrix}$$

Aufgabe 3 (*Minimale Spannbäume* **[6 Punkte]**)

Gegeben sei folgender Graph:



1. Wenden Sie den Algorithmus von Prim an, um einen minimal spannenden Baum des Graphen zu berechnen. Der Startknoten sei S. [3 Punkte]

2. Wenden Sie den Algorithmus von Kruskal an, um einen minimal spannenden Baum des Graphen zu berechnen. [3 Punkte]

Geben Sie die Kanten in der Reihenfolge an, in der sie zur jeweiligen Lösungsmenge hinzugefügt werden. Da alle Kantengewichte verschieden sind, genügt es, jede Kante durch ihr Gewicht zu identifizieren.

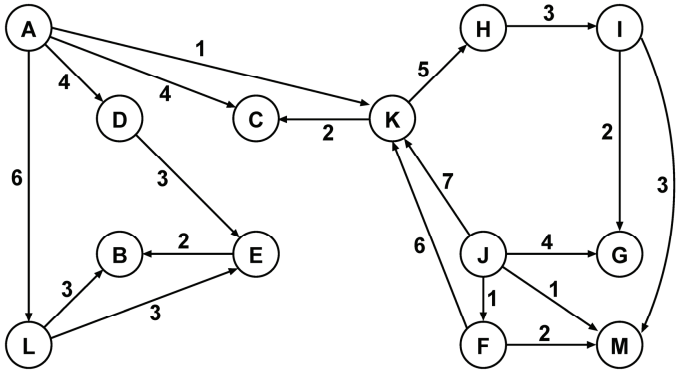
Lösungsvorschlag

1. Prim: $3, 4, 5, 2, 6, 1, 10$

2. Kruskal: $1, 2, 3, 4, 5, 6, 10$

Aufgabe 4 (*Topologisches Sortieren* [**4 Punkte**])

Gegeben sei der folgende gerichtete Graph G:



1. Bestimmen Sie für G eine topologische Sortierung und geben Sie die Knoten als sortierte Folge an. Die Kantengewichte können hierbei vernachlässigt werden. [3 Punkte]

2. Erweitern Sie G um eine beliebige Kante (ohne Gewicht), so dass keine topologische Sortierung für den modifizierten Graphen mehr existiert. [1 Punkt]

Prof. Dr. Leif Kobbelt
Thomas Ströder, Fabian Emmes
Sven Middelberg, Michael Kremer

Informatik 8
RWTH Aachen

Lösungsvorschlag

1. Da eine topologische Sortierung eines Graphen im Allgemeinen nicht eindeutig ist, sind mehrere Lösungen gültig. Hier ist eine mögliche Lösung angegeben:

$$J, F, A, K, H, I, M, G, C, L, D, E, B$$

2. Eine mögliche Kante, die einen Zyklus erzeugen würde, wäre (G, H). Auch hier gibt es viele Möglichkeiten. Jede Kante, die einen Zyklus in G erzeugt, ist eine gültige Lösung.