Algorithmische Mathematik:
Graphen & Anwendungen

Frühlingssemester 2018
P. Zaspel und I. Kalmykov

Universität Basel

Übungsblatt 10.

zu bearbeiten bis **Dienstag, 15.5.2018, 14:00 Uhr**.

Aufgabe 1. (Kantenkonnektivität)

Die Kantenkonnektivität eines ungerichteten, zusammenhängenden Graphen $G = (V, E)$ ist die minimale Anzahl an Kanten $|E_K|$, $E_K \subset E$, die man entfernen muss, sodass $G' = (V, E \backslash E_K)$ nicht mehr zusammenhängend ist (d.h. für die Anzahl der Zusammenhangskomponenten von G' gilt $\kappa(G') > 1$). Zum Beispiel ist die Konnektivität eines Baumes 1 und die Konnektivität eines Rings 2. Zeigen Sie, wie man die Berechnung der Kantenkonnektivität eines Graphen auf ein Flussproblem zurückführen kann.

(4 Punkte)

Aufgabe 2. (Edmonds-Karp I)

Zeigen Sie graphisch, analog zu dem Beispiel in der Vorlesung, die Schritte des Edmonds–Karp–Algorithmus auf dem Netzwerk in Abbildung 1. Zeichnen Sie zu jeder Iteration den Graphen mit dem zugehörigen Fluss und geben Sie den kürzesten augmentierenden Pfad an.

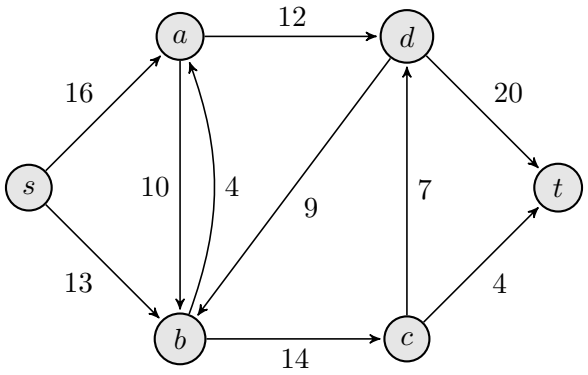

Abbildung 1: Graph für Edmonds–Karp–Algorithmus

(4 Punkte)

Aufgabe 3. (Ford–Fulkerson)

Gegeben sei das Netzwerk mit Fluss f in Abbildung 2. Dabei sei $\sigma = \frac{\sqrt{5}-1}{2}$ und $s = \frac{1}{1-\sigma}$. Der gegebene Fluss resultiert beispielsweise aus dem Nullfluss durch Augmentierung entlang des Wegs $\pi = s, v_3, v_2, t$. Zeigen Sie, dass es eine Folge von augmentierten Wegen $\pi_1, \pi_2, \pi_1, \pi_3, \pi_1, \pi_2, \pi_1, \pi_3, \ldots$ gibt, so dass der Ford–Fulkerson–Algorithmus nicht terminiert. Dazu beachte man, dass $\sigma^n = \sigma^{n+1} + \sigma^{n+2}$ gilt.

Hinweis: $s = \sum_{k=0}^{\infty} \sigma^k = \frac{1}{1-\sigma}$ ist der Wert der geometrischen Reihe für σ.

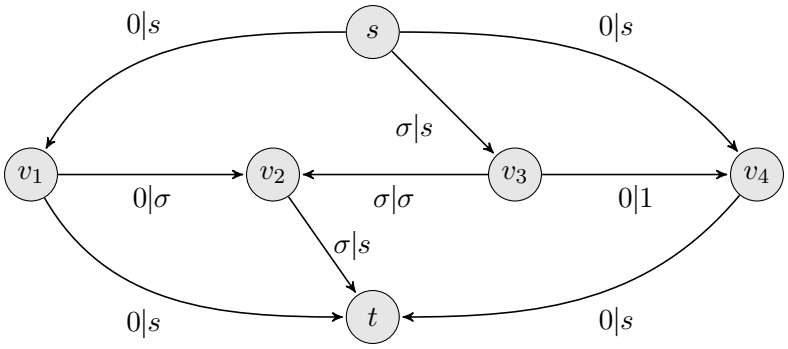

Abbildung 2: Graph für Ford–Fulkerson–Algorithmus

Aufgabe 4. (Edmonds-Karp II)

Implementieren Sie den Edmonds–Karp–Algorithmus. Nutzen Sie die modifizierte Breitensuche aus Algorithmus 2 um den kürzesten augmentierenden s–t Pfad zu finden. Testen Sie Ihre Implementierung an dem Netzwerk aus Abbildung 3. Geben Sie für jede Iteration der `while`–Schleife im Algorithmus 1 den berechneten kürzesten augmentierenden Pfad aus.

Algorithmus 1 (Edmonds–Karp)

Input: Netzwerk $N = (G, c, s, t)$, $G = (V, E)$ gerichtet,
Output: Fluss $f : E \to \mathbb{R}$, s.d. f maximal.

1: $\forall e \in E : f(e) \leftarrow 0$
2: $d = 0$
3: **while** $d < \infty$ **do**
4: $[d, p]$ = BFSEK(N,f)
5: Augmentiere p um d
6: **end while**
7: **return** f

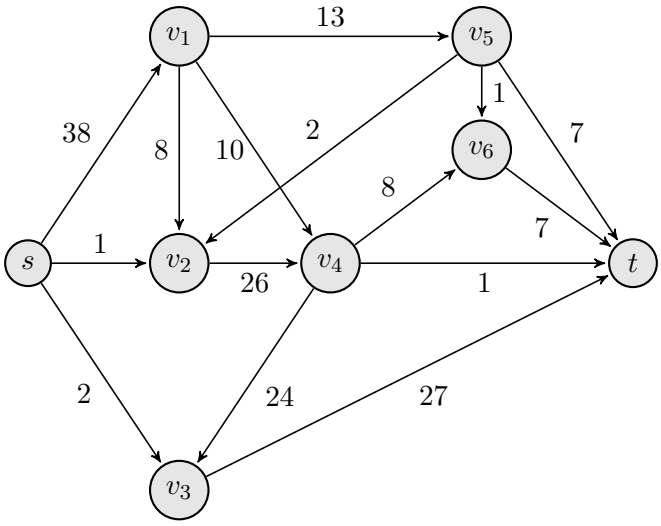

Abbildung 3: Graph zum Testen der Implementierung

Algorithmus 2 (BFSEK)

Input: Netzwerk $N = (G, c, s, t)$, $G = (V, E)$ gerichtet, $f : E \to \mathbb{R}$ - Fluss auf N

Output: $p : V \to V$, $d \in \mathbb{R}$, sodass
falls ein kürzester f-augmentierender s–t Pfad existiert:
$p(t)$, $p(p(t))$, ... - kürzester f-augmentierenden s–t Pfad,
d - Minimum über $c(e) - f(e)$, falls e Vorwärtskante in π,
bzw. $f(e)$, falls e Rückwärtskante in π, bzgl. aller Kanten e auf π,
sonst:
$d = \infty$, $p(t)$ - undefiniert.

1: $R \leftarrow \{s\}$
2: Initialisiere Q als Queue
3: $Q \leftarrow \{s\}$
4: $\forall v \in V : D(v) \leftarrow \infty$
5: $d \leftarrow \infty$
6: **while** $Q \neq \varnothing$ **do**
7: Wähle v (als das erste Element) aus Q
8: **for** $w \in (\text{post}(v)) \cap (V \backslash R)$ mit $c((v,w)) - f((v,w)) > 0$ **do**
9: $p(w) \leftarrow v$
10: $D(w) \leftarrow \min\{c((v,w)) - f((v,w)), D(v)\}$
11: **if** $w = t$ **then**
12: $d \leftarrow D(w)$
13: **return** (d, p)
14: **end if**
15: $R \leftarrow R \cup \{w\}$, $Q \leftarrow Q \cup \{w\}$
16: **end for**
17: **for** $w \in (\text{pre}(v)) \cap (V \backslash R)$ mit $f((w,v)) > 0$ **do**
18: $p(w) \leftarrow v$
19: $D(w) \leftarrow \min\{f((w,v)), D(v)\}$
20: **if** $w = t$ **then**
21: $d \leftarrow D(w)$
22: **return** (d, p)
23: **end if**
24: $R \leftarrow R \cup \{w\}$, $Q \leftarrow Q \cup \{w\}$
25: **end for**
26: $Q \leftarrow Q \backslash \{v\}$
27: **end while**

(4 Punkte)