Rückblick:

52

4.12.07

Alg. 3.17

(1) $R := \{s\}$, $Q := \{s\}$, $T := \emptyset$, $l(s) := 0$

(2) WHILE ($Q \neq \emptyset$) DO {

wähle erstes Element ~~aus~~ $v \in Q$.

(3) IF (es gibt kein $w \in V \setminus R$ mit $e = \{v, w\} \in E$) THEN

$Q := Q \setminus \{v\}$

(4) ELSE {

wähle $v \in V \setminus R$ mit $e = \{v, w\} \in E$

setze $R := R \cup \{w\}$, $T := T \cup \{e\}$, hänge w an Q an.

setze $l(w) := l(v) + 1$.

}

}

(5) STOP

Satz 3.18

(i) Verfahren 3.17 ist ein Algorithmus.

(ii) Die Laufzeit ist $O(n+m)$.

(iii) Am Ende ist für jeden erreichbaren Knoten $v \in R$ die Länge eines kürzesten Weges von s nach v im Baum (R,T) durch $\ell(v)$ gegeben.

(iv) Am Ende ist für jeden erreichbaren Knoten $v \in R$ die Länge eines kürzesten Weges von s nach v im Graphen (V,E) durch $\ell(v)$ gegeben.

Beweis:

Ind. wie (iv)

(i) Wie für Algorithmus 3.7 gelten alle Eigenschaften; zusätzlich ist für jeden Knoten $v \in Q$ per Induktion der Wert $\ell(v)$ tatsächlich definiert.

Wird eine Kante $e = \{u,v\}$ ist $\ell(u)$ tatsächlich definiert. in 4 wird dann $\ell(v)$ definiert.

(ii) Die Laufzeit bleibt von Algorithmus 3.7 erhalten.

(iii) Sei $d_{(R,T)}(s,v)$ die Länge eines kürzesten Weges von s nach v in (R,T). Dann zeigt man ~~über~~ durch Induktion über $d_{(R,T)}(s,v)$, dass für alle Knoten $d_{(R,T)}(s,v) = \ell(v)$ gilt:

Induktionsanfang:

$d_{(R,T)}(s,v) = 0$ gilt genau für $v = s$,

und $\ell(s) = 0$. ($\ell(s)$ ist die Länge eines kürzesten Weges von s nach s)

Induktionsannahme:

Sei $d_{(R,T)}(s,v) = \ell(v)$ für alle

$v \in V$ mit $d_{(R,T)}(s,v) \leq k-1$.

Knoten mit Entf $\leq$ k-1

Induktionsschritt: $k-1 \rightsquigarrow k$

Nur im fertigen konstruierten Baum

Sei $w \in V$ ein Knoten mit

$$d_{(R,T)}(s,w) = k.$$

Dann gibt es im Baum (R,T) einen eindeutigen Weg von s zu w; sei v der Vorgänger von w in diesem Weg, also $\{v,w\} \in T$.

keine Kreise im Baum

Nach Induktionsannahme gilt

$$d_{(R,T)}(s,v) = \ell(v);$$

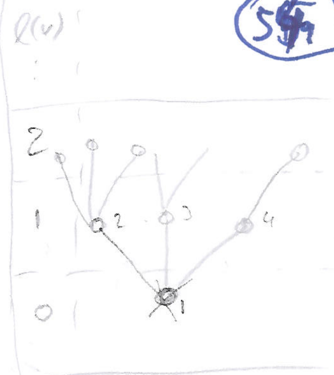
(da $d_{(R,T)}(s,v) \leq k-1$)

außerdem ist

$$d_{(R,T)}(s,w) = d_{(R,T)}(s,v) + 1$$

und $\ell(w) = \ell(v) + 1$, (in 4 gesetzt)

also $d_{(R,T)}(s,w) = \ell(w)$, und die Behauptung gilt.



(iv) Wir brauchen zunächst zwei Eigenschaften von Q.

(a) $\ell(v)$ wächst monoton mit der Aufnahme von v in die Warteschlange Q.

Begründung: Mit Induktion über die Anzahl der Kanten.

Die Aussage gelte für Q: $v_r\, v_{r+1} \dots v_k$ und v_k wurde gerade wg. $\{v_r, v_k\}$ aufgenommen, d.h. $\ell(v_r) \le \ell(v_{r+1}) \le \dots \le \ell(v_k)$.

Werden Knoten aus Q gelöscht, ändert dieses nichts an der Monotonie.

Also: Q: $v_i\, v_{i+1} \dots v_k$

Nun wird $\{v_i, v_{k+1}\}$ aufgenommen. Q: $v_i\, v_{i+1} \dots v_k\, v_{k+1}$

Dann gilt $\ell(v_k) \underset{(v_r,v_k)\in T}{=} \ell(v_r)+1 \underset{\text{da vorher Monotonie galt}}{\le} \ell(v_i)+1 \underset{\text{In (4) gesetzt}}{=} \ell(v_{k+1})$, und damit weiterhin Monotonie in Q.

Damit haben wir sogar noch mehr gezeigt:

Q: $v_i\, v_{i+1} \dots v_k\, v_{k+1}$

$$\ell(v_i) \le \dots \le \ell(v_{k+1}) = \ell(v_i)+1.$$

d.h., die Längen $\ell(v)$ der Knoten in Q unterscheiden sich um höchstens 1.

Wegen der Monotonie gilt $\ell(v_i)+1 \ge \ell(w)$ für $w \in R$ (evtl. w nicht mehr in Q)

Damit gilt,

(b) , dass bei der Auswahl eines Knotens $v \in Q$ in Schritt (2) es keinen Knoten $w \in R$ mit $\ell(w) > \ell(v)+1$ geben kann.

(iv) Zunächst gilt, dass die Funktion $\ell(v)$ monoton mit der Aufnahme von v in die Warteschlange Q wächst.

$\ell(v_1) \leq \ldots \leq \ell(v_k)$
$v_1 v_2 \ldots v_k$

Außerdem gilt, dass bei Auswahl eines Knotens $v \in Q$ in Schritt (2) keinen Knoten $w \in R$ mit

$$\ell(w) > \ell(v) + 1 \qquad (*)$$

geben kann: So ein Knoten kann nicht vor v ausgewählt worden sein (wegen der Monotonie von Q), andererseits gibt es in der Warteschlange Q zum Zeitpunkt der Auswahl von v nur Knoten, die von Vorgängern von v in der Auswahlreihenfolge durch eine Kante erreichbar sind.

Jetzt nehmen wir an, am Ende des Algorithmus gibt es Knoten $w \in V$ mit

$$d(s,w) < d_{(R,T)}(s,w) = \ell(w).$$

Unter diesen Knoten betrachte einen mit minimalem Abstand von s in G mit dieser Eigenschaft.

Sei P ein kürzester s-w-Weg in G, und sei $e = \{v,w\}$ die letzte Kante in P,
d.h. $d(s,w) = d(s,v) + 1$.

Wir haben $d(s,v) = d_{(R,T)}(s,v)$,

aber $d(s,w) < d_{(R,T)}(s,w)$,

also gehört e nicht zu T.

Außerdem ist

$$\ell(w) = d_{(R,T)}(s,w) > d(s,w) = d(s,v)+1$$
$$= d_{(R,T)}(s,v)+1 = \ell(v)+1.$$

Wäre $w \in R$, hätten wir wegen

$$\ell(w) > \ell(v)+1$$

einen Widerspruch zur Eigenschaft $(*)$

Also muss $w \notin R$ gelten; dann verbindet aber die Kante $e = \{v,w\}$ zum Zeitpunkt der Entfernung von v aus Q v mit einem Knoten $w \notin R$, im Widerspruch zur Abfrage in 3.

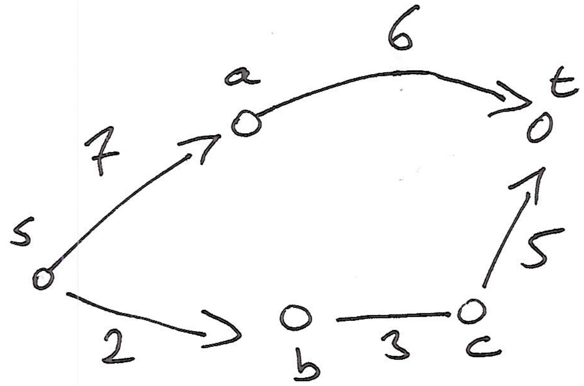
$\square$

3.8 Ausblick: Algorithmische Probleme auf Graphen

Problem 1: Kürzeste Weg mit Kantenlängen

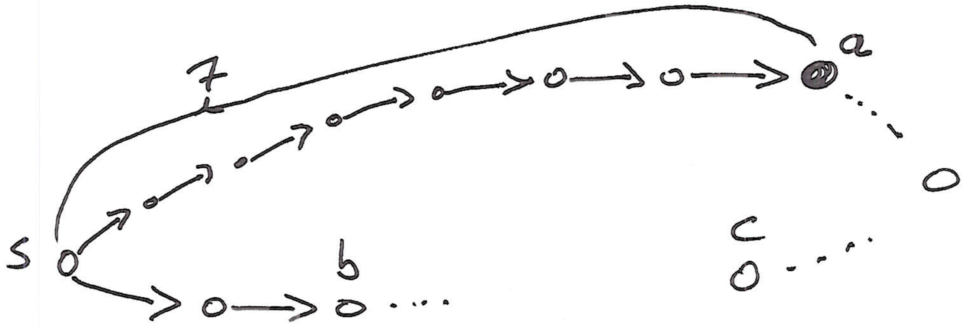
Gegeben: Graph $G = (V, E)$, $c: E \to \mathbb{N}$ Kantenlänge, zwei Knoten $s, t \in V$.

Gesucht: Ein kürzester Weg von s nach t.



Würde man die Kantenlängen ignorieren und BFS anwenden wäre das Ergebnis 13 (und damit nicht optimal)

Ersetzt man eine Kante der Länge $c(e)$ durch $c(e)$ Kanten der Länge 1, könnte man auf diesen Graphen BFS anwenden.

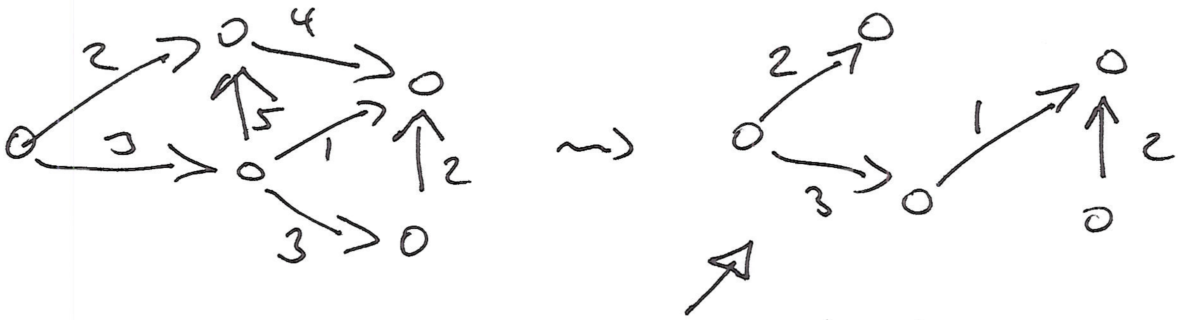


Allerdings wird in diesem Graphen die Laufzeit von BFS ziemlich schlecht.

Problem 2: Kostengünstigste Netzwerke

Gegeben: Graph $G=(V,E)$. $c: E \to \mathbb{N}$ Kantengewichte

Gesucht: Ein zusammenhängender Teilgraph T mit $\sum_{e \in T} c(e)$ minimal.



Hat man pos. Kantengewichte, gibt es in der Lösung nie einen Kreis (man könnte eine Kante weglassen und hätte eine bessere Lösg)

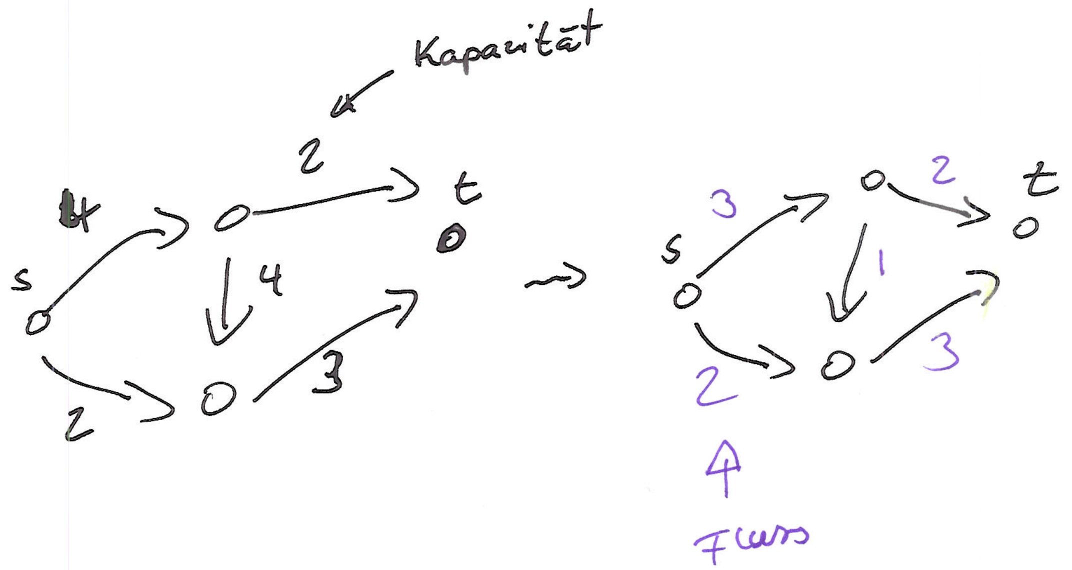
Deshalb spricht man auch von:

Minimalen aufspannenden Bäumen.

Problem 3: Flussprobleme

Gegeben: Graph $G=(V,E)$. Kapazitäten der Kanten: $c: E \to \mathbb{N}$. Zwei Knoten s,t.

Gesucht: Maximaler Fluss von s nach t (der die Kapazitäten auf den Kanten nicht überschreitet)



Knoten $v \in V \setminus \{s,t\}$ leiten das Gut nur weiter.

Beispiele

- Verkehr (evtl. mehrere „Quellen")
- Datenpakete