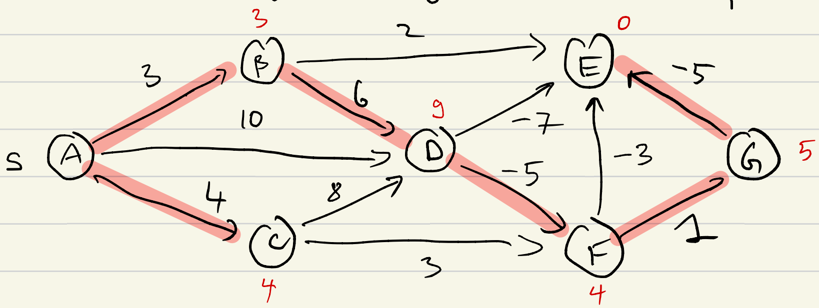
Kürzeste Wege in gewichteten Graphen



ger. Graph $G=(V,E)$, Startknoten $s \in V$,

Kantenkosten $c(e)$, $e \in E$

gesucht: günstigste Wege von s (shortest paths)

Wegkosten: $W = (v_0, v_1, \dots, v_{\ell-1}, v_\ell)$ kurz: $v_0 \xrightarrow{W} v_\ell$

$c(W) :=$ Summe der Kantenkosten

$= c(v_0, v_1) + \dots + c(v_{\ell-1}, v_\ell)$

negative Kantenkosten OK

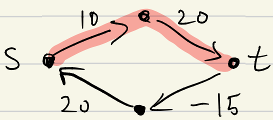
werden sehen: nicht-negative Kantenkosten einfacher

Formal:

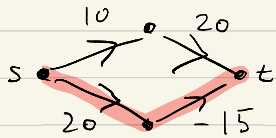
$$d(u,v) = \min \{ \operatorname{cost}(W) \mid u \xrightarrow{W} v \}$$

Dreiecksungleichung: $d(u,v) \le d(u,w) + d(w,v)$

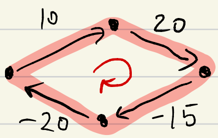
Beispiele:



$d(s,t) = 30$



$d(s,t) = 5$



$d(s,t) = -\infty$

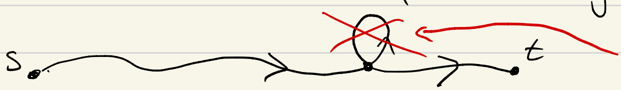
wiederhole neg. Zyklus beliebig oft

$\nexists$ günstigster Weg

Annahme: kein neg. Zyklus von s erreichbar

Lemma: Annahme $\Rightarrow$ günstigste Wege

Können immer zu Pfaden abgekürzt werden



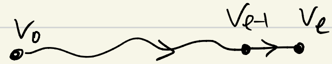
$c(\text{Zyklus}) \geq 0$

Lemma: Annahme $\Rightarrow$ $d(s,s) = 0$

Eigenschaft günstigster Wege

Falls $v_0, \dots, v_{\ell-1}, v_\ell$ günstigst,

dann auch $v_0, \dots, v_{\ell-1}$



Shortest path tree

ger. Baum T mit Wurzel s (siehe Beispiel am Anfang)

enthält alle günstigsten Wege von s:

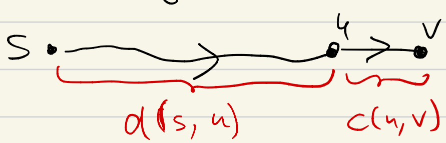
alle Pfade von s in T sind günstigste Wege in G

existiert immer (später: Beweis per Algorithmus)

Rekurrenz

$$d(s,v) = \min_{\text{Vorgänger } u} d(s,u) + c(u,v)$$

betrachte günstigsten Weg von s nach v



Dynamische Programmierung?

Berechnungsreihenfolge unklar!

Kürzeste Wege in azyklischen Graphen

berechne Rekurrenz entlang top Sortierung

$d[s] \leftarrow 0$

$d[v] \leftarrow \infty$ für $v \in V$, nicht erreichbar von s

FOR $v \in V$ in topologischer Reihenfolge

$$d[v] \leftarrow \min_{\text{Vorgänger } u} d[u] + c(u,v)$$

(siehe Beispiel am Anfang)

Laufzeit: $O(|V| + |E|)$ falls Adjazenzliste geg.

⑤

kürzeste Wege in allgemeinen Graphen

finde Kandidaten für günstigste Wege und verwalte obere Schranken $d[v]$ für Distanzen

<u>Invariante:</u> $d[v] \geq d(s,v) \quad \forall v \in V$

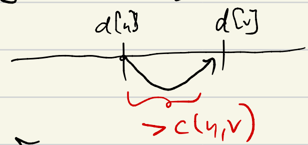
<u>Ziel:</u> $d[v] = d(s,v) \quad \forall v \in V$

<u>Idee:</u> "berechne" Rekurrenz in beliebiger Reihenfolge

$$d[v] \leftarrow \min_{\text{Vorgänger } u} d[u] + c(u,v)$$

<u>gespannte Kante (u,v):</u> $d[v] > d[u] + c(u,v)$

<u>gespannte Kante entspannen:</u>



$\text{relax}(u,v)$: $d[v] \leftarrow d[u] + c(u,v)$

hält Invariante aufrecht!

<u>Ford's Algorithmus:</u>

$d[s] \leftarrow 0, \; d[v] \leftarrow \infty \quad \forall v \in V \setminus \{s\}$

WHILE $\exists$ gespannte Kante $(u,v) \in E$
  relax (u,v)

Ford's algorithmus

korrekt wenn terminiert

Rekurrenz erfüllt:

$$d[v] = \min_{\text{Vorgänger } u} d[u]$$

$$d[s] = 0$$

terminiert $\Leftrightarrow$ $\nexists$ neg. Zyklus

auch günstigste Wege berechnen (anstatt nur Dist.)

erweitere relax(u,v) um:

$p[v] \leftarrow u$

(merke Vorgänger u der aktuelle Wert von $d[v]$ definierte)

Weg $s, p[\cdots p[v] \cdots], \ldots, p[p[v]], p[v], v$

Laufzeit: exponentiell im worst-case (selbst ohne neg. Zyklus)

<u>Verbesserung</u>: sorgfältigere relax Reihenfolge

<u>Bellman-Ford</u>:

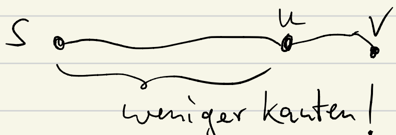
$d[s] \leftarrow 0$, $d[v] \leftarrow \infty$, $v \neq s$

WHILE gespannte Kante existieren

FOR gespannte Kante $(u,v) \in E$

relax(u,v)

<u>Analyse</u>: betrachte günstigsten Weg nach v

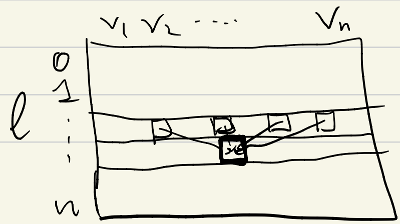


$$d(s,v)^{\leq \ell} = \min\{ c(W) \mid s \xrightarrow{W} v,\ \leq \ell \text{ Kanten}\}$$

<u>Rekurrenz</u>: $$d(s,v)^{\leq \ell} = \min \left\{ \min_{\text{Vorgänger } u} \left\{ d(s,u)^{\leq \ell-1} + c(u,v) \right\},\ d(s,v)^{\leq \ell-1} \right\}$$

$$d(s,v)^{\leq 0} = \begin{cases} 0 & \text{if } v=s \\ \infty & \text{o/w} \end{cases}$$

DP



- berechne Zeilen nacheinander
- $O(|E|)$ Zeit pro Zeile
- $O(|V| \cdot |E|)$ Zeit insgesamt

Falls $\nexists$ neg. Zyklus, Pfade reichen aus $\rightsquigarrow \ell \leq n-1$

und $d(s,v)^{\leq n-1} = d(s,v)$

Falls $\exists v.\ d(s,v)^{\leq n} < d(s,v)^{\leq n-1}$, dann $\exists$ neg. Zyklus

(andere Richtung gilt auch)

Verbindung zu Bellmann-Ford:

Beh.: nach ℓ Durchgängen der WHILE Schleife

$$d[v] \leq d(s,v)^{\leq \ell}$$

Induktion:

$\ell = 0$ ✓

Induktions-annahme

$\ell \to \ell + 1$: jede Kante betrachtet

$$\forall (u,v) \in E.\ d[v] \leq d(s,u)^{\leq \ell} + c(u,v)$$

$$\Rightarrow d[v] \leq \min \left\{ \underbrace{\min_{\text{Vorgänger } u} d(s,u)^{\leq \ell} + c(u,v),\ d(s,v)^{\leq \ell-1}}_{= d(s,v)^{\leq \ell+1}} \right\}$$

Nicht negative Kantenkosten

verwalte Menge S von Knoten ($s \in S$)

- kürzeste Wege bekannt (I.1)

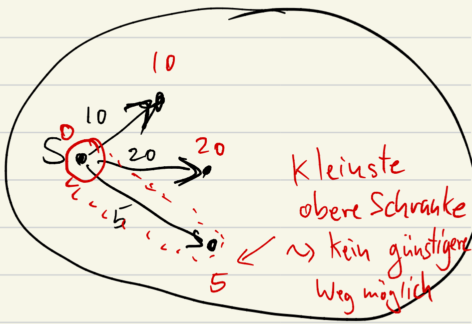
$$\forall v \in S. \quad d[v] = d(s,v)$$

- obere Schranken für $v \in V \setminus S$ (I.2)

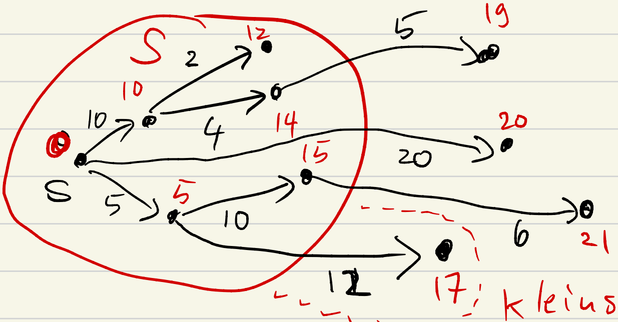
$$d[v] = \min_{\substack{\text{Vorgänger } u \\ u \in S}} \underbrace{d[u]}_{= d(s,u)} + c(u,v)$$

Ziel: vergrössere S

Beginn: $S = \{s\}$



Allgemein:



kleinste obere Schranke ~> kein günstigerer Weg möglich

<u>Beh.:</u> betrachte $v^* \in V \setminus S$ mit $d[v^*]$ minimal

dann: $d(s, v^*) = d[v^*]$

Beweis:

jeder Weg W von s nach v^* muss S verlassen

$$\Rightarrow \quad c(W) \geq \min_{\substack{(u,v) \in E \\ u \in S \\ v \in V \setminus S}} d(u) + c(u,v) = d[v^*]$$

$\square$

Algorithmus (Dijkstra):

$d[s] \leftarrow 0, \quad d[v] \leftarrow \infty, \quad v \in V \setminus \{s\}$

$S \leftarrow \emptyset$

WHILE $S \neq V$:

wähle $v^* \in V \setminus S$ mit $d[v^*]$ minimal

$S \leftarrow S \cup \{v\}$

FOR $v \in V \setminus S$

$$d[v] \leftarrow \min_{\substack{(u,v) \in E \\ u \in S}} d[u] + c(u,v)$$