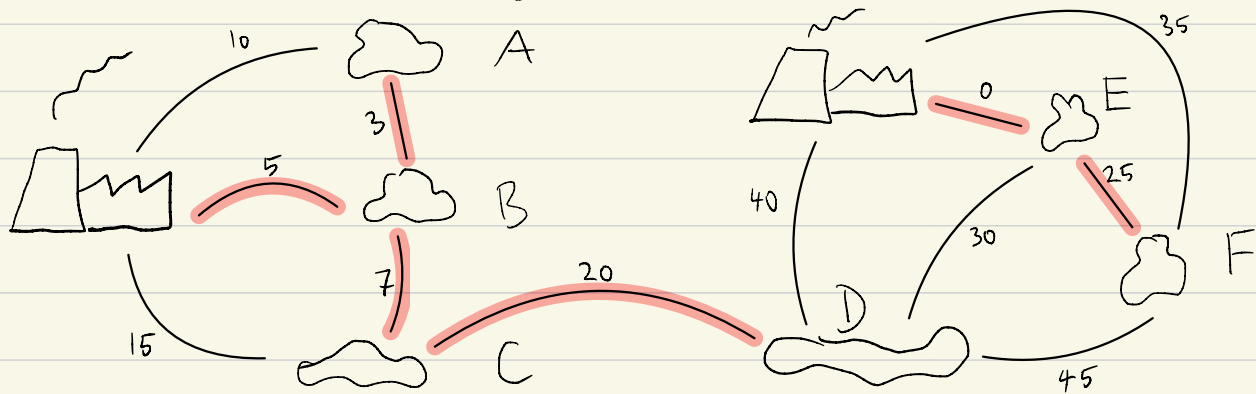


①

1926: Elektrifizierung Mährens (Region in Tschechien)

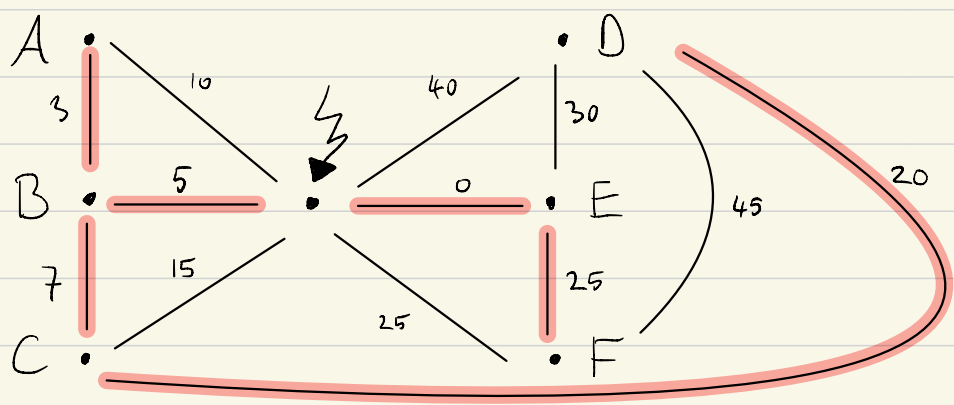


Frage an Otakar Boruvka (Mathematiker)

welche Stromleitungen bauen, so dass

- alle Städte mit Strom versorgt
- Gesamtkosten minimal

gewichteter Graph



Ziel: zusammenhängender Teilgraph mit minimalem Gewicht

Graph $G = (V, E)$ zusammenhängend

Kantengewichte: $\{w(e) \geq 0\}_{e \in E}$

(spanning)

aufspannende Kanten: $A \subseteq E$, Graph (V, A) z.hängend

Gewicht: $w(A) = \sum_{e \in A} w(e)$

können aufsp. Kanten mit min. Gewicht Kreis enthalten?

Beobachtung: kann aufspannende Kanten mit min. Gewicht
immer ohne Kreis wählen

Beweis: entferne Kante auf Kreis solange bis kein Kreis

Spannbaum: $T \subseteq E$ aufspannend, kein Kreis enthalten

minimaler Spannbaum (MST): S.baum mit min. Gewicht

weitere Anwendungen

"clustering": zerlege Graph in k "gute" Teile

genauer: Teilgraph mit k ZHKs und min. Gewicht

- berechne MST

- entferne $k-1$ schwerste Kanten

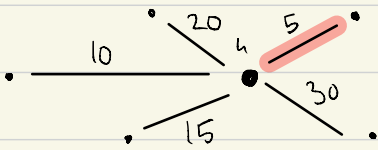
Korrektheit hier nicht
besprochen

Annahme: alle Kanten gewichte verschieden

(vereinfacht Analyse; Algorithmen auch sonst korrekt)

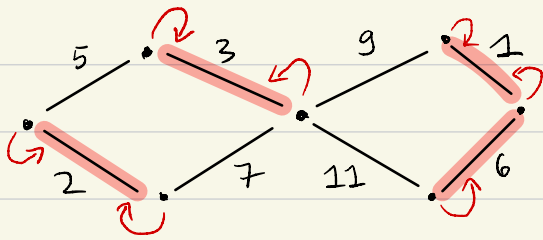
sichere Kante: enthalten in jedem MST

Beispiel: betrachte Knoten u ; welche inzidente Kante sicher?



Vermutung: minimale Kante an Knoten sicher

bilden diese Kanten MST?

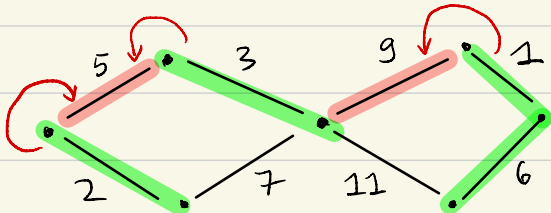


nicht aufspannend

$\leadsto$ erhalten Wald F (forest)

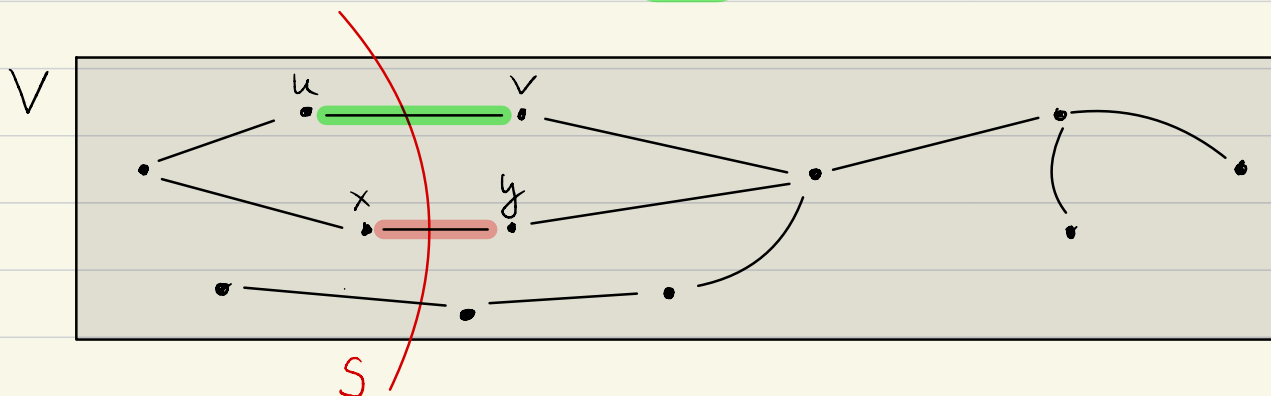
weitere sichere Kanten?

Vermutung: minimale Kante an ZHK von F sicher



Schnittprinzip

(bestätigt unsere Vermutungen)

 $\forall S \subseteq V$. minimale Kante uv an S sicher ($u \in S, v \notin S$)Beweis: (indirekt) betrachte Spannb Baum T , $uv \notin T$ Zu zeigen: T nicht MSTbetrachte $xy \in T$ an S (also $w(xy) > w(uv)$)ersetze xy durch uv in T $\leadsto$ erhalte besseren Spannb Baum $\leadsto T$ nicht MST

← warum?

aufspannend?

wichtig (sonst Beweis falsch):wähle xy auf Kreis in $T \cup \{uv\}$

Boruvka (G):

$F \leftarrow \emptyset$ (sichere Kanten)

while F nicht Spannbaum:

$(S_1, \dots, S_k) \leftarrow \text{ZHKs von } F$

$(e_1, \dots, e_k) \leftarrow \text{minimale Kanten an } S_1, \dots, S_k$

$F \leftarrow F \cup \{e_1, \dots, e_k\}$

Laufzeit pro Iteration: $O(|V| + |E|)$

- berechne ZHKs mit DFS
- gehe einmal durch alle Kanten
- um minimale für jede ZHK zu finden

Anzahl Iterationen: $\leq \log |V|$

jede Iteration halbiert Anzahl ZHKs

$|\{e_1, \dots, e_k\}| \geq k/2$
da jede Kante für höchstens 2 ZHKs minimal sein kann

totale Laufzeit: $O((|V| + |E|) \cdot \log |V|)$

Variation 1: auf eine ZHK konzentrieren ^⑥

Prim (G, s) :

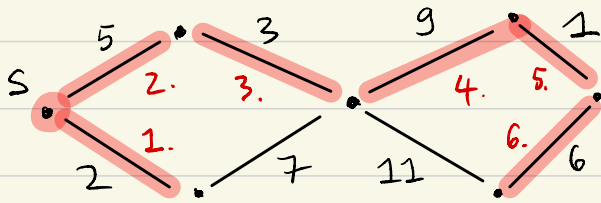
$F \leftarrow \emptyset$

$S \leftarrow \{s\}$ (ZHK von s in F)

while F nicht Spannbaum:

$u^*v^* \leftarrow$ minimale Kante an S ($u^* \in S, v^* \notin S$)

$F \leftarrow F \cup \{u^*v^*\}; \quad S \leftarrow S \cup \{v^*\}$



Laufzeit:

wie minimale Kante schnell finden?

wie Dijkstra: priority queue (min-heap)

Prim (G, s): (mit min-heap)

$H \leftarrow \text{make-heap}(V, \infty), S \leftarrow \emptyset$

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

$\text{decrease-key}(H, s, 0)$

while $H \neq \emptyset$:

$v^* \leftarrow \text{extract-min}(H)$

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

for $v^*v \in E, v \notin S$:

$d[v] \leftarrow \min \{d[v], \underline{w(v^*v)}\}$

$\text{decrease-key}(H, v, d[v])$

Unterschied Dijkstra:
 $d[v^*] + w(v^*v)$

Laufzeit:

wie Dijkstra (und Boruvka)

$O((|V| + |E|) \cdot \log |V|)$

Variation 2: sichere Kanten sortiert nach Gewicht

Kruskal(G):

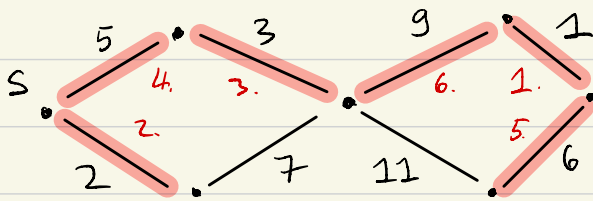
$F \leftarrow \emptyset$ (sichere Kanten)

for $uv \in E$, aufsteigend sortiert

if u, v in verschiedenen ZHKs von F

$F \leftarrow F \cup \{uv\}$

Beispiel



Laufzeit: pro Iteration: $O(|V|)$ (ZHKs berechnen)

Anzahl Iterationen: $\leq |E|$

geht es besser?

Idee: Datenstruktur für ZHKs von F

Union-find Datenstruktur ($|V|=n$)

Operationen:

$\text{make}(V)$: erstelle Datenstruktur für $F = \emptyset$

$\text{same}(u, v)$: teste ob u, v in derselben ZHK von F

$\text{union}(u, v)$: vereinige ZHKs von u und v

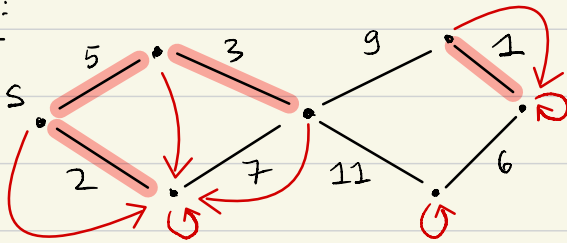
(füge Kante uv zu F hinzu)

Speicher:

$\text{rep}[v]$: eindeutiger Repräsentant der ZHK von V

($\text{rep}[u] = \text{rep}[v] \Leftrightarrow \text{ZHK}(u) = \text{ZHK}(v)$)

Beispiel:



Implementierung:

make(V): $\text{rep}[v] \leftarrow v$ für $v \in V$ $O(n)$

same(u, v): teste ob $\text{rep}[u] = \text{rep}[v]$ $O(1)$

union(u, v): for $x \in V$, $\text{rep}[x] = \text{rep}[u]$ $O(n)$
 $\text{rep}[x] \leftarrow \text{rep}[v]$

geht es besser als $O(n)$ für union?

Idee: verwalte alle Knoten einer ZHK als Liste

Invariante:

$\text{members}[\text{rep}[u]]$ ist Liste aller Knoten in $\text{ZHK}(u)$

union(u, v):

for $x \in \text{members}[\text{rep}[u]]$:

$\text{rep}[x] \leftarrow \text{rep}[v]$

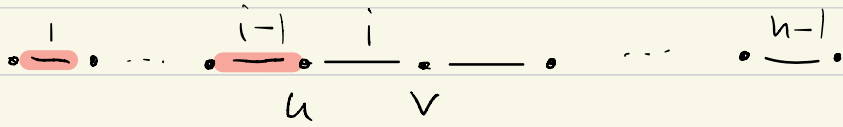
$\text{members}[\text{rep}[v]] \leftarrow \text{members}[\text{rep}[v]] \cup \{x\}$

Laufzeit: $O(|\text{ZHK}(u)|)$

worst-case



i-ter union Aufruf:



$$|\text{ZHK}(u)| = i \leadsto \underline{\text{Laufzeit: } \Theta(i)}$$

total: $\Theta(1 + 2 + \dots + (n-1)) = \Theta(n^2)$

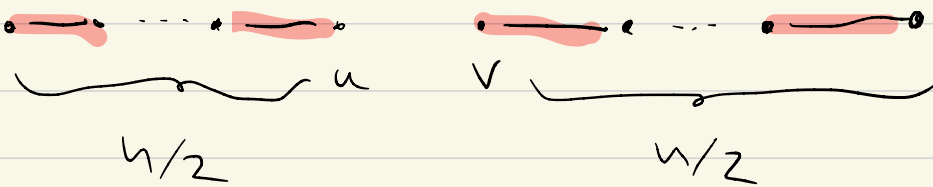
geht es besser?

Idee: durchlaufe nur Knoten der kleineren ZHK

union Laufzeit:

$$\Theta(\min\{|\text{ZHK}(u)|, |\text{ZHK}(v)|\})$$

worst-case $\Theta(n)$ für einzelne union Aufruf



$$\min\{|\text{ZHK}(u)|, |\text{ZHK}(v)|\} = n/2$$

Amortisierte Analyse

(12)

totale Laufzeit aller union Aufrufe $O(n \cdot \log n)$

$\leadsto$ im Mittel nur $O(\log n)$ pro union Aufruf

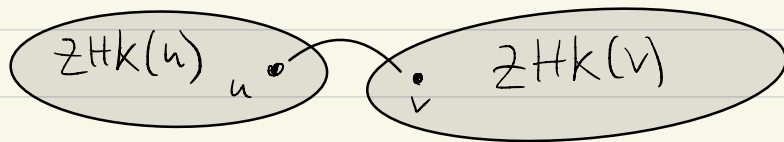
⚠ Mittel nur über Aufrufe; Graph bleibt worst-case

Beweis:

$N_u :=$ Anzahl Änderungen von $\text{rep}[u]$

totale Laufzeit $O\left(\sum_{u \in V} N_u\right)$

Wie gross kann N_u sein?



$\text{rep}[u]$ nur verändert wenn $|ZHK(u)| \leq |ZHK(v)|$

$\leadsto$ Grösse der ZHK von u mindestens verdoppelt

$\leadsto N_u \leq \log_2 n$

$\leadsto \sum_{u \in V} N_u \leq n \cdot \log_2 n$

Laufzeit Kruskal: $O\left(\underbrace{|E| \cdot \log |E|}_{\text{Sortieren}} + \underbrace{|V| \cdot \log |V|}_{\text{union find}}\right)$