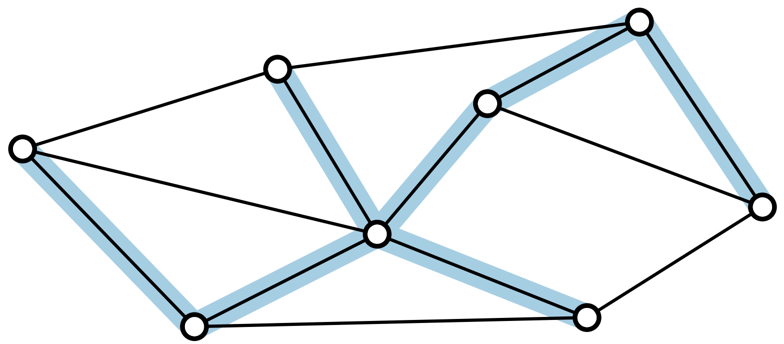
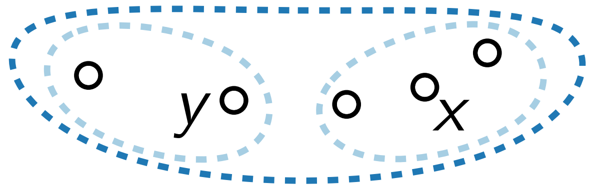
Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Algorithmen und Datenstrukturen

Vorlesung 21:
Minimale Spannbäume





Alexander Wolff

Wintersemester 2025

Motivation

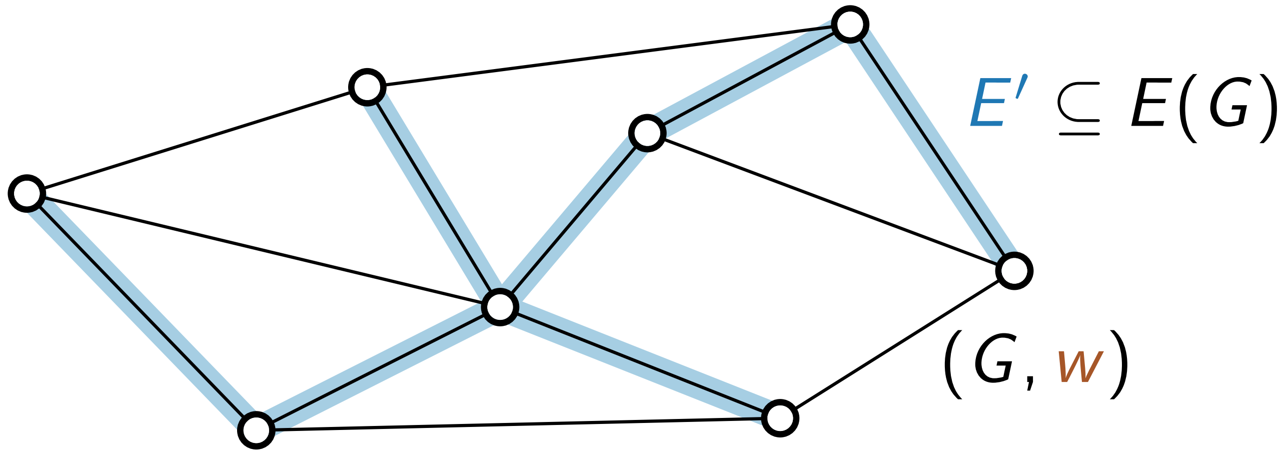
ungerichteter, gewichteter Graph

Gegeben. Zusammenhängendes Straßennetz (G, w), das eine Menge V von n Städten verbindet

$w \colon E(G) \to \mathbb{R}_{>0}$ Kantengewichte
$w(E') := \sum_{e \in E'} w(e)$

Gesucht. Teilnetz $T = (V(G), E')$ mit $E' \subseteq E(G)$, so dass

- alle Städte in T erreichbar sind (T **spannt** G **auf**)
- die „Schneeräumkosten“ $w(E')$ minimal sind unter allen Teilnetzen, die G aufspannen.



z.B. mit $w \equiv$ euklid. Abstände

Minimaler Spannbaum

Wegen der Minimalität von $w(E')$ gilt:

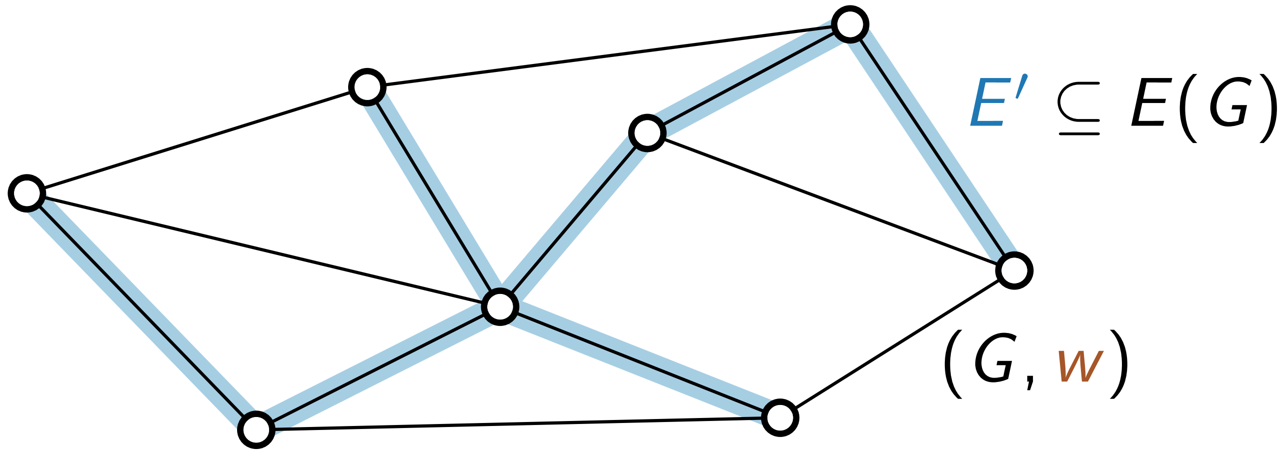
T hat keine Kreise $\Rightarrow$ T ist ein Wald

T „erbt“ Zusammenhang von G $\Rightarrow$ T ist ein Baum

T spannt G auf $\Rightarrow$ T ist Spannbaum von G

T hat minimales Gewicht unter *allen* Spannbäumen von G.

Wir nennen T kurz **minimalen Spannbaum (MSB)** von G.

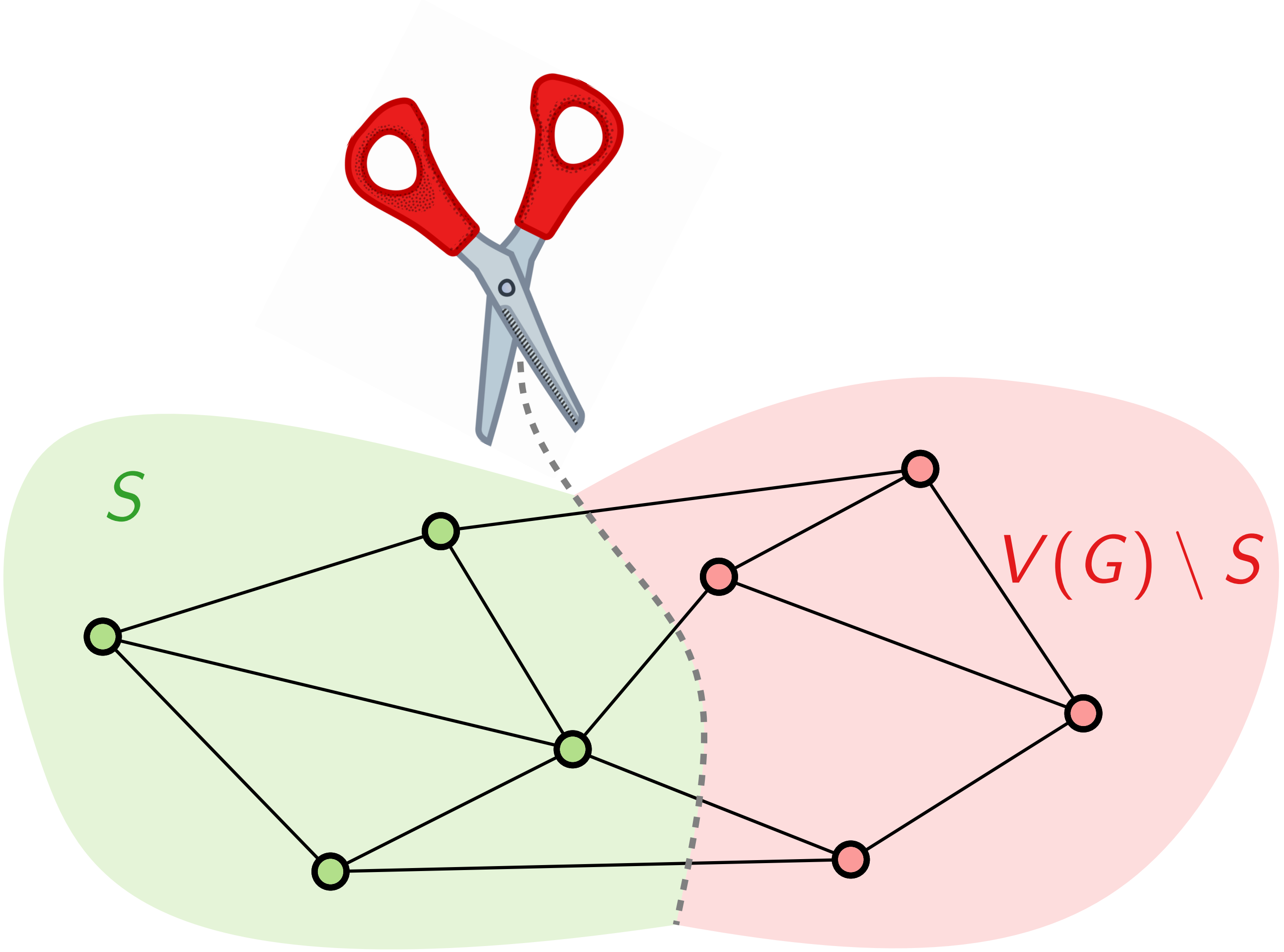


Beob. $|E'| = |V(G)| - 1$

Otakar Borůvka
*1899 Ostroh, Mähren
† 1995 Brünn

Schnitte

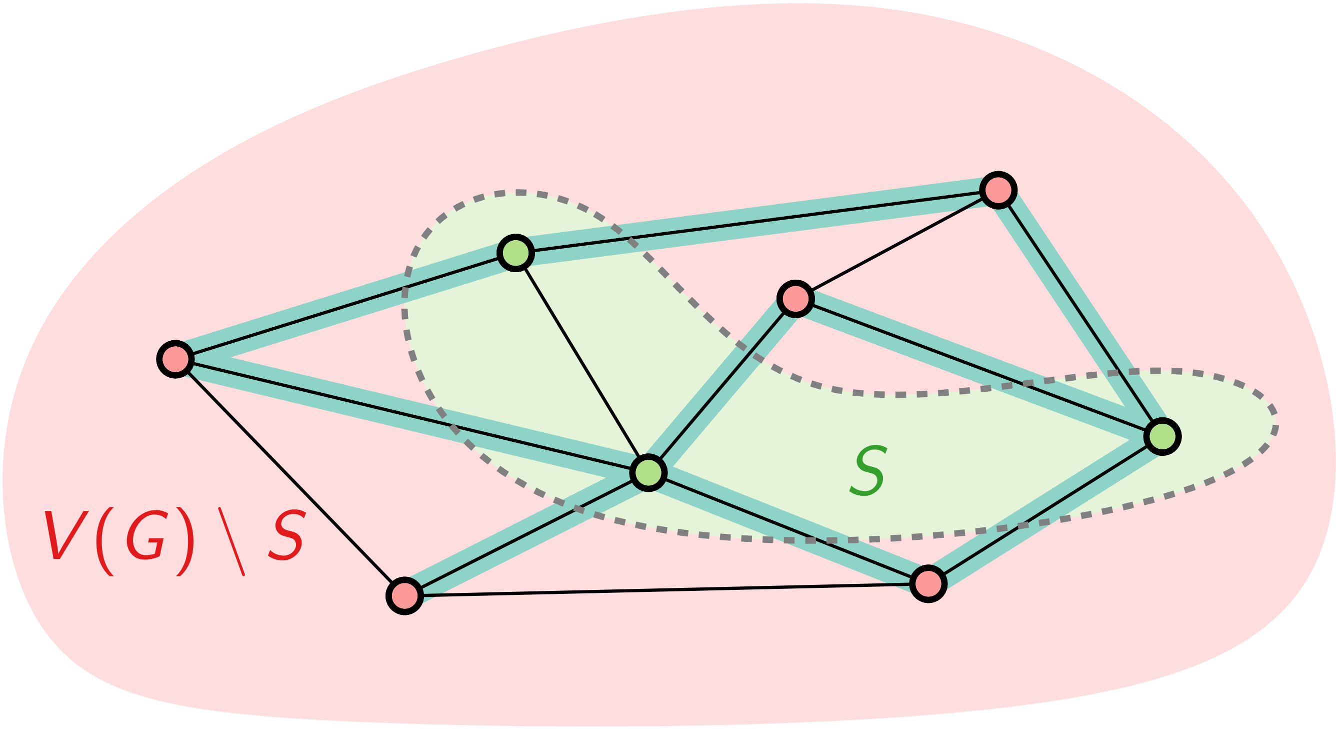
Def. Ein **Schnitt** $(S, V(G) \setminus S)$ eines Graphen G ist eine Zerlegung von $V(G)$ in zwei Teilmengen.



Schnitte

Def. Ein **Schnitt** $(S, V(G) \setminus S)$ eines Graphen G ist eine Zerlegung von $V(G)$ in zwei Teilmengen.

Eine Kante uv **kreuzt** $(S, V(G) \setminus S)$, wenn $u \in S$ und $v \in V(G) \setminus S$ (oder andersherum).

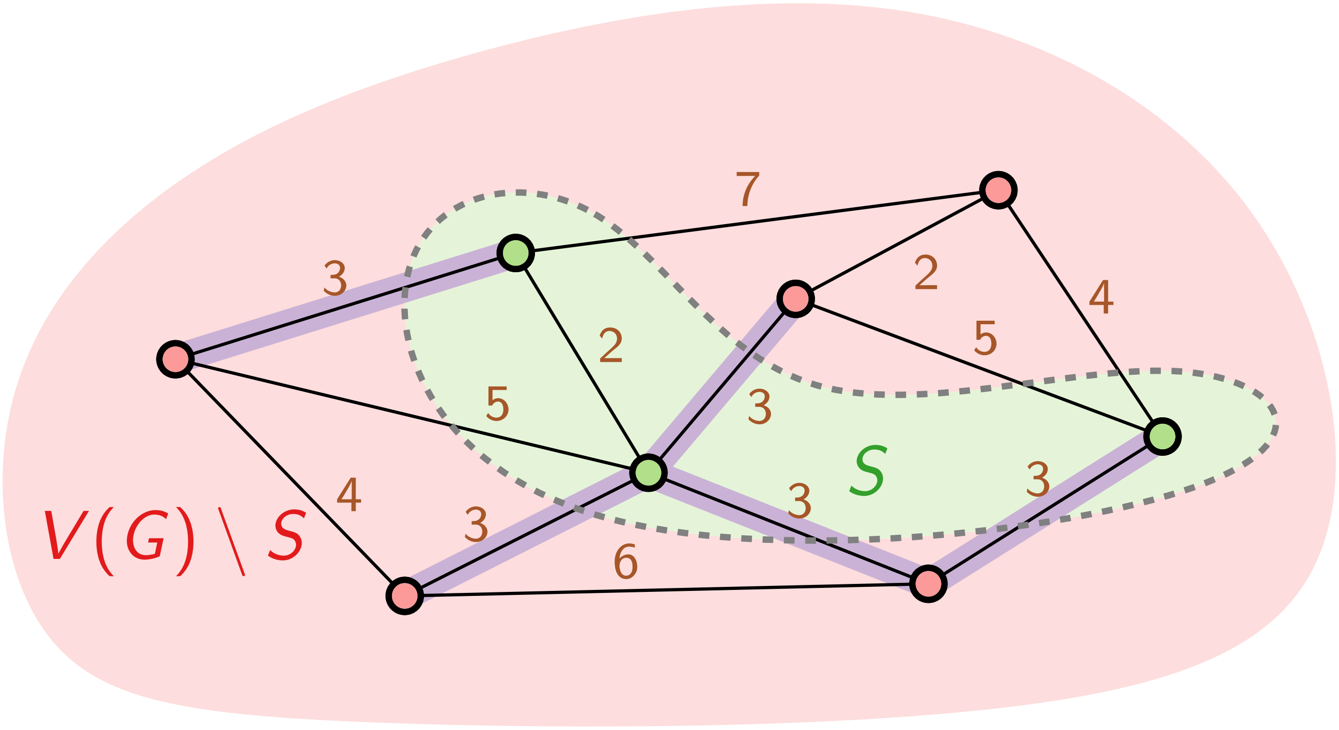


Schnitte

Def. Ein **Schnitt** $(S, V(G) \setminus S)$ eines Graphen G ist eine Zerlegung von $V(G)$ in zwei Teilmengen.

Eine Kante uv **kreuzt** $(S, V(G) \setminus S)$, wenn $u \in S$ und $v \in V(G) \setminus S$ (oder andersherum).

Eine Kante uv, die einen Schnitt kreuzt, ist **leicht**, wenn alle Kanten, die den Schnitt kreuzen, mindestens $w(uv)$ wiegen.



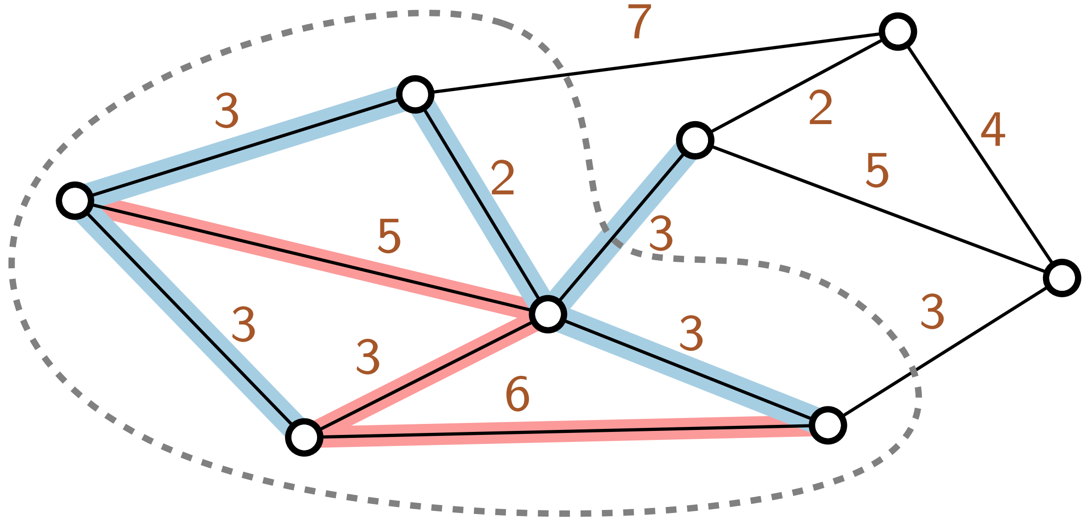
Allgemeiner Greedy-Algorithmus

Färbe alle Kanten des Graphen:

- blau: Kante für den MSB
- rot: Kante nicht für den MSB
- ungefärbt: Noch nicht entschieden

Verwende zwei Regeln:

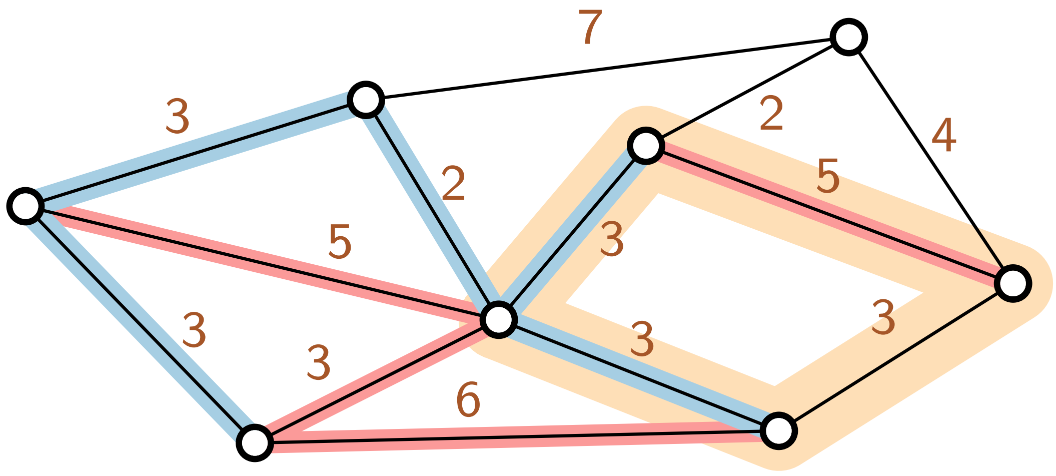
Blaue Regel:
Wähle Schnitt, den keine blaue Kante kreuzt
Färbe leichte Kante blau



Allgemeiner Greedy-Algorithmus

Färbe alle Kanten des Graphen:

- blau: Kante für den MSB
- rot: Kante nicht für den MSB
- ungefärbt: Noch nicht entschieden



Verwende zwei Regeln:

Blaue Regel:
Wähle Schnitt, den keine blaue Kante kreuzt
Färbe leichte Kante blau

Rote Regel:
Wähle Kreis ohne rote Kante
Färbe größte ungefärbte Kante auf Kreis rot

GreedySpannbaum(G, w)
Wende blaue Regel oder rote Regel an,
bis alle Kanten gefärbt sind.
Gib $E' = \{$blaue Kanten$\}$ zurück

Satz. GreedySpannbaum findet einen minimalen Spannbaum.

Beweis Greedy Algorithmus

GreedySpannbaum(G, w)
Wende blaue Regel oder rote Regel an,
bis alle Kanten gefärbt sind.
Gib $E' = \{$blaue Kanten$\}$ zurück

Farbinvariante (FI): Es gibt einen MSB T:
- T enthält alle blauen Kanten.
- T enthält keine rote Kante.

FI ist am Anfang offensichtlich erfüllt.

Lemma. Die blaue Regel hält die Farbinvariante aufrecht.

Lemma. Die rote Regel hält die Farbinvariante aufrecht.

Lemma. GreedySpannbaum färbt alle Kanten.

Satz. GreedySpannbaum findet einen minimalen Spannbaum.

Beweis.
- Jede Kante ist entweder blau oder rot.
- Es gibt einen MSB T, der alle blauen Kanten und keine rote Kante enthält.

$\Rightarrow$ Blaue Kanten bilden MSB $\square$

Beweis der blauen Regel

Farbinvariante (FI): Es gibt einen MSB T:

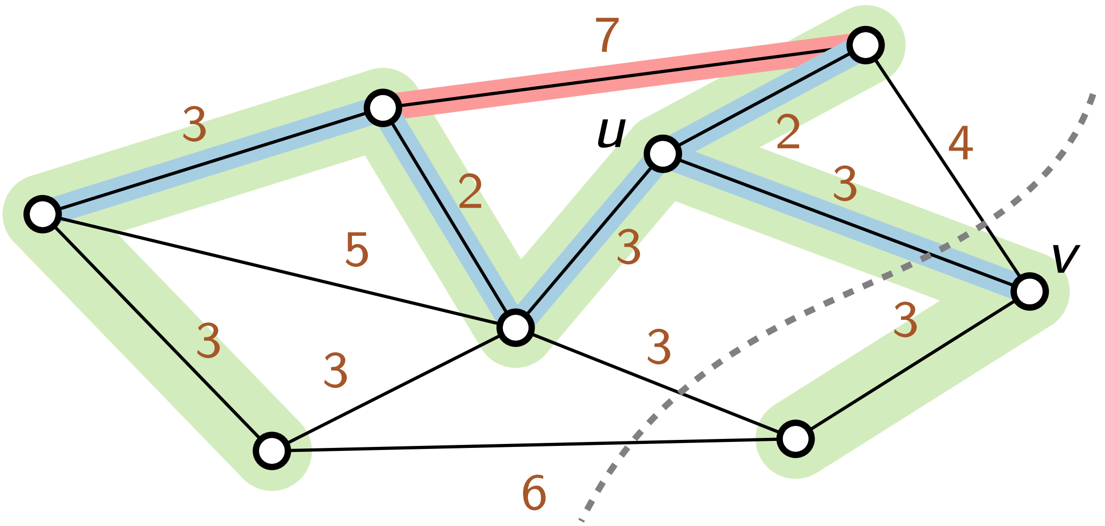
- T enthält alle blauen Kanten.
- T enthält keine rote Kante.

Blaue Regel:
Wähle Schnitt, den keine blaue Kante kreuzt.
Färbe leichte Kante blau.

Lemma. Die blaue Regel hält die Farbinvariante aufrecht.

Beweis. Alle Kanten ungefärbt $\Rightarrow$ jeder MSB **bezeugt** FI.
Sei T minimaler Spannbaum, der FI bezeugt.
Sei $uv \in E$ von blauer Regel ausgewählte Kante.

1. Fall: $uv \in E(T)$ $\Rightarrow$ FI bleibt erhalten.



Beweis der blauen Regel

Farbinvariante (FI): Es gibt einen MSB T:
- T enthält alle blauen Kanten.
- T enthält keine rote Kante.

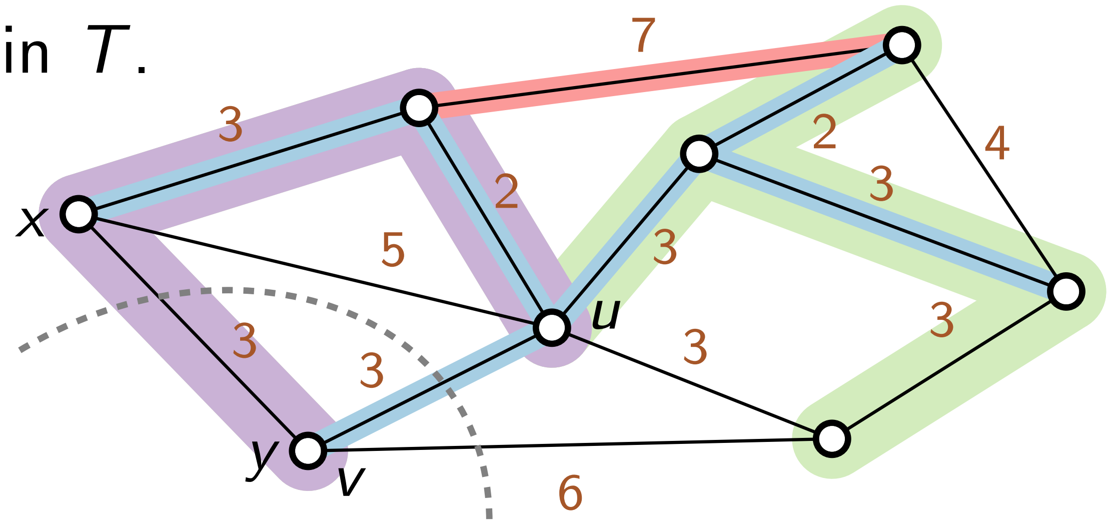
Blaue Regel:
Wähle Schnitt, den keine blaue Kante kreuzt.
Färbe leichte Kante blau.

Lemma. Die blaue Regel hält die Farbinvariante aufrecht.

Beweis. Alle Kanten ungefärbt $\Rightarrow$ jeder MSB **bezeugt** FI.
Sei T minimaler Spannbaum, der FI bezeugt.
Sei $uv \in E$ von blauer Regel ausgewählte Kante.

1. Fall: $uv \in E(T)$ $\Rightarrow$ FI bleibt erhalten.
2. Fall: $uv \notin E(T)$ $\Rightarrow$ Es gibt Pfad p von u nach v in T.

$\Rightarrow$ p enthält Kante xy, die Schnitt kreuzt



Beweis der blauen Regel

Farbinvariante (FI): Es gibt einen MSB T:
- T enthält alle blauen Kanten.
- T enthält keine rote Kante.

Blaue Regel:
Wähle Schnitt, den keine blaue Kante kreuzt.
Färbe leichte Kante blau.

Lemma. Die blaue Regel hält die Farbinvariante aufrecht.

Beweis. Alle Kanten ungefärbt $\Rightarrow$ jeder MSB **bezeugt** FI.
Sei T minimaler Spannbaum, der FI bezeugt.
Sei $uv \in E$ von blauer Regel ausgewählte Kante.

1. Fall: $uv \in E(T)$ $\Rightarrow$ FI bleibt erhalten.
2. Fall: $uv \notin E(T)$ $\Rightarrow$ Es gibt Pfad p von u nach v in T.

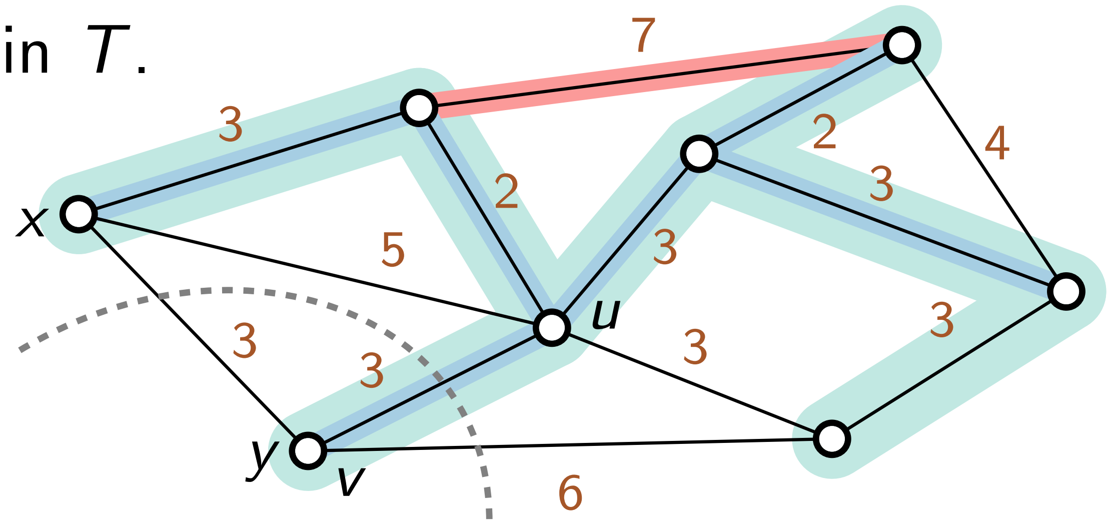
$\Rightarrow$ p enthält Kante xy, die Schnitt kreuzt

keine blaue Kante kreuzt $\Rightarrow$ Kante xy ist ungefärbt.

leichte Kante $\Rightarrow$ $w(xy) \geq w(uv)$

Wähle $E' = E(T) \cup \{uv\} \setminus \{xy\}$.

$\Rightarrow$ $T' = (V(T), E')$ ist MSB, der FI bezeugt. $\square$



Beweis der roten Regel

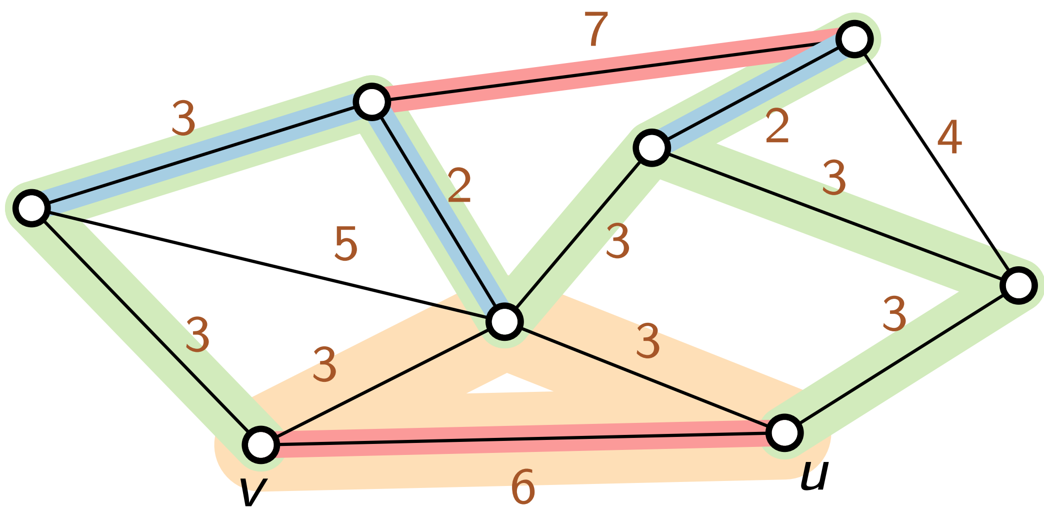
Farbinvariante (FI): Es gibt einen MSB T:
- T enthält alle blauen Kanten
- T enthält keine rote Kante

Rote Regel:
Wähle Kreis ohne rote Kante.
Färbe größte ungefärbte Kante auf Kreis rot.

Lemma. Die rote Regel hält die Farbinvariante aufrecht.

Beweis. Sei T min. Spannbaum, der FI bezeugt.
Sei K von roter Regel ausgewählter Kreis.
Sei $uv \in E$ von roter Regel gefärbte Kante.

1. Fall: $uv \notin E(T) \Rightarrow$ FI bleibt erhalten.



Beweis der roten Regel

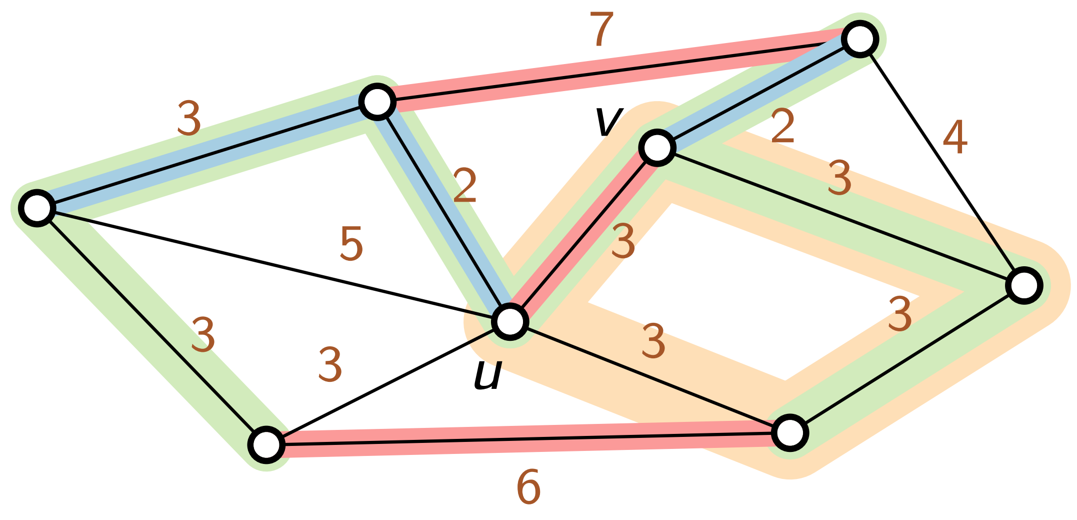
Farbinvariante (FI): Es gibt einen MSB T:
- T enthält alle blauen Kanten
- T enthält keine rote Kante

Rote Regel:
Wähle Kreis ohne rote Kante.
Färbe größte ungefärbte Kante auf Kreis rot.

Lemma. Die rote Regel hält die Farbinvariante aufrecht.

Beweis. Sei T min. Spannbaum, der FI bezeugt.
Sei K von roter Regel ausgewählter Kreis.
Sei $uv \in E$ von roter Regel gefärbte Kante.

1. Fall: $uv \notin E(T)$ $\Rightarrow$ FI bleibt erhalten.
2. Fall: $uv \in E(T)$



Beweis der roten Regel

Farbinvariante (FI): Es gibt einen MSB T:
- T enthält alle blauen Kanten
- T enthält keine rote Kante

Rote Regel:
Wähle Kreis ohne rote Kante.
Färbe größte ungefärbte Kante auf Kreis rot.

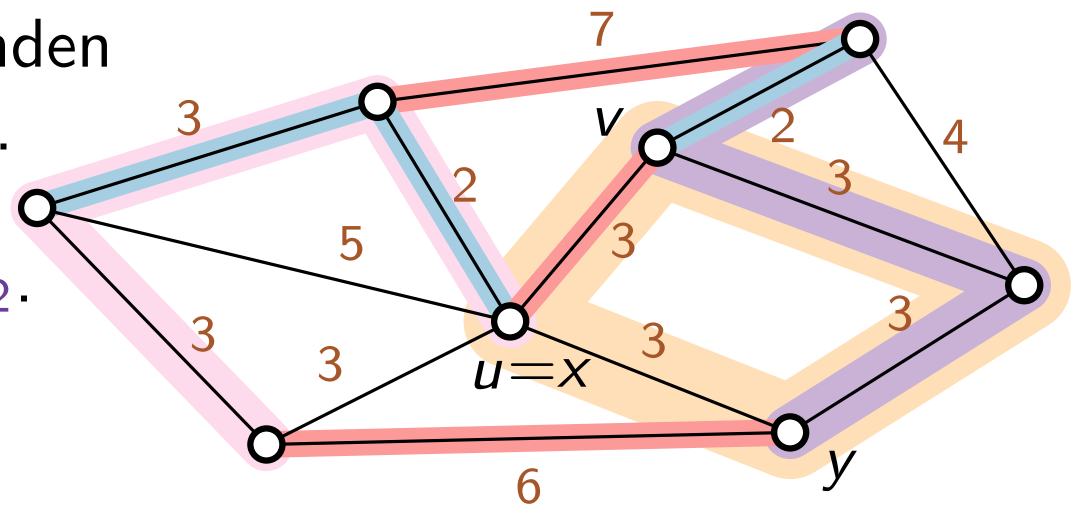
Lemma. Die rote Regel hält die Farbinvariante aufrecht.

Beweis. Sei T min. Spannbaum, der FI bezeugt.
Sei K von roter Regel ausgewählter Kreis.
Sei $uv \in E$ von roter Regel gefärbte Kante.

1. Fall: $uv \notin E(T)$ $\Rightarrow$ FI bleibt erhalten.
2. Fall: $uv \in E(T)$ $\Rightarrow$ $E(T) \setminus \{uv\}$ bildet aufspannenden Wald mit zwei Bäumen T_1, T_2.
Sei $u \in T_1$, $v \in T_2$.

$xy \notin E(T)$

$\Rightarrow$ Es gibt Kante $xy \neq uv$ in K mit $x \in T_1, y \in T_2$.



Beweis der roten Regel

Farbinvariante (FI): Es gibt einen MSB T:
- T enthält alle blauen Kanten
- T enthält keine rote Kante

Rote Regel:
Wähle Kreis ohne rote Kante.
Färbe größte ungefärbte Kante auf Kreis rot.

Lemma. Die rote Regel hält die Farbinvariante aufrecht.

Beweis. Sei T min. Spannbaum, der FI bezeugt.
Sei K von roter Regel ausgewählter Kreis.
Sei $uv \in E$ von roter Regel gefärbte Kante.

1. Fall: $uv \notin E(T)$ $\Rightarrow$ FI bleibt erhalten.
2. Fall: $uv \in E(T)$ $\Rightarrow$ $E(T) \setminus \{uv\}$ bildet aufspannenden Wald mit zwei Bäumen T_1, T_2.
Sei $u \in T_1$, $v \in T_2$.

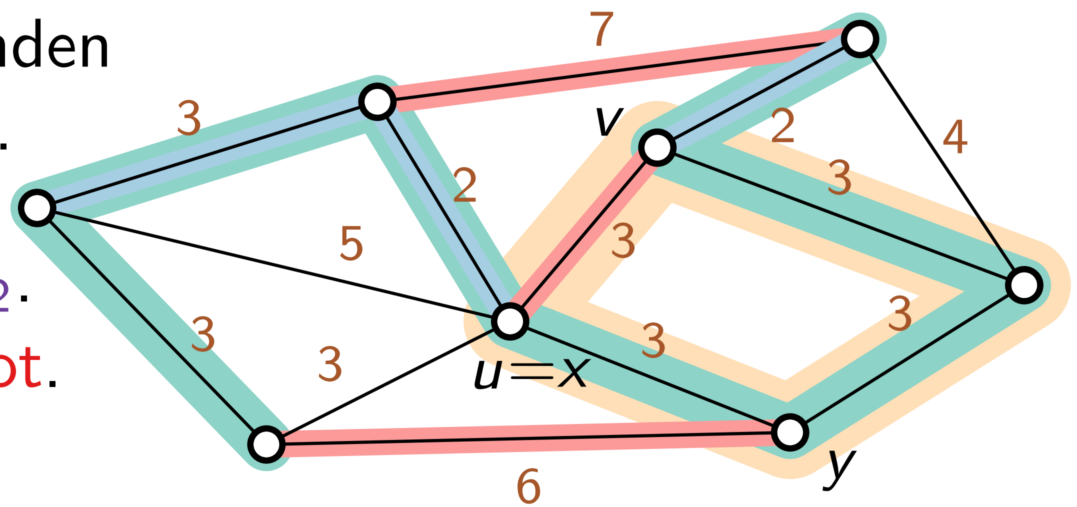
$xy \notin E(T)$

$\Rightarrow$ Es gibt Kante $xy \neq uv$ in K mit $x \in T_1, y \in T_2$.

größte ungefärbte Kante $\Rightarrow w(xy) \leq w(uv)$ ohne rote Kante $\Rightarrow xy$ nicht rot.

Wähle $E' = T(E) \cup \{xy\} \setminus \{uv\}$.

$\Rightarrow T' = (V(T), E')$ ist MSB, der FI bezeugt. $\square$



Alle Kanten werden gefärbt

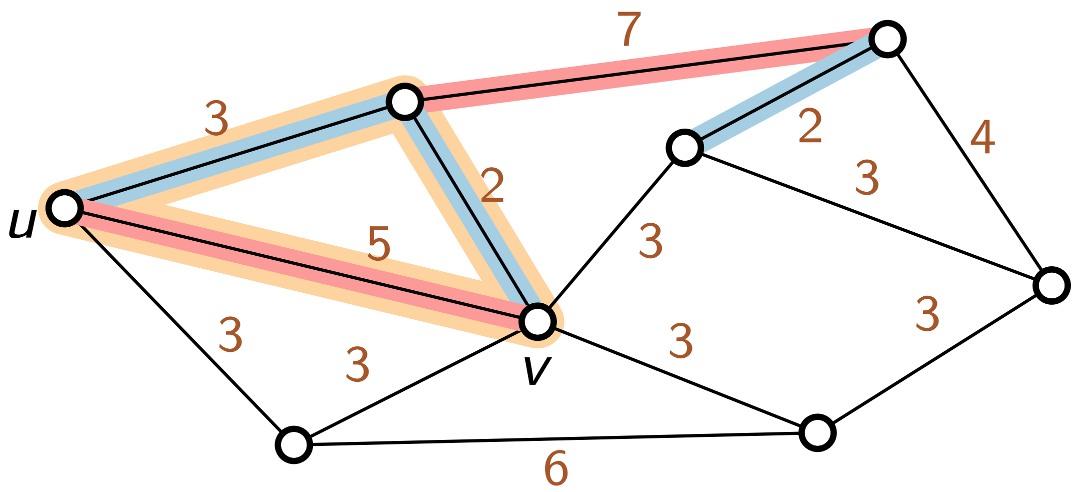
Rote Regel:
Wähle Kreis ohne rote Kante.
Färbe größte ungefärbte Kante auf Kreis rot.

Blaue Regel:
Wähle Schnitt, den keine blaue Kante kreuzt.
Färbe leichte Kante blau.

Lemma. GreedySpannbaum färbt alle Kanten.

Beweis. Blaue Kanten bilden Wald B (ggfs. isolierte Knoten)
Sei uv ungefärbte Kante. Zu zeigen: uv (oder eine andere Kante!) wird gefärbt.

1. Fall: uv verbindet Knoten *eines* Baumes aus B.
Wähle Kreis C: Pfad in B von v zu u + Kante uv
$\Rightarrow$ Kanten auf C alle blau bis auf uv.
$\Rightarrow$ Rote Regel anwendbar auf uv.



Alle Kanten werden gefärbt

Rote Regel:
Wähle Kreis ohne rote Kante.
Färbe größte ungefärbte Kante auf Kreis rot.

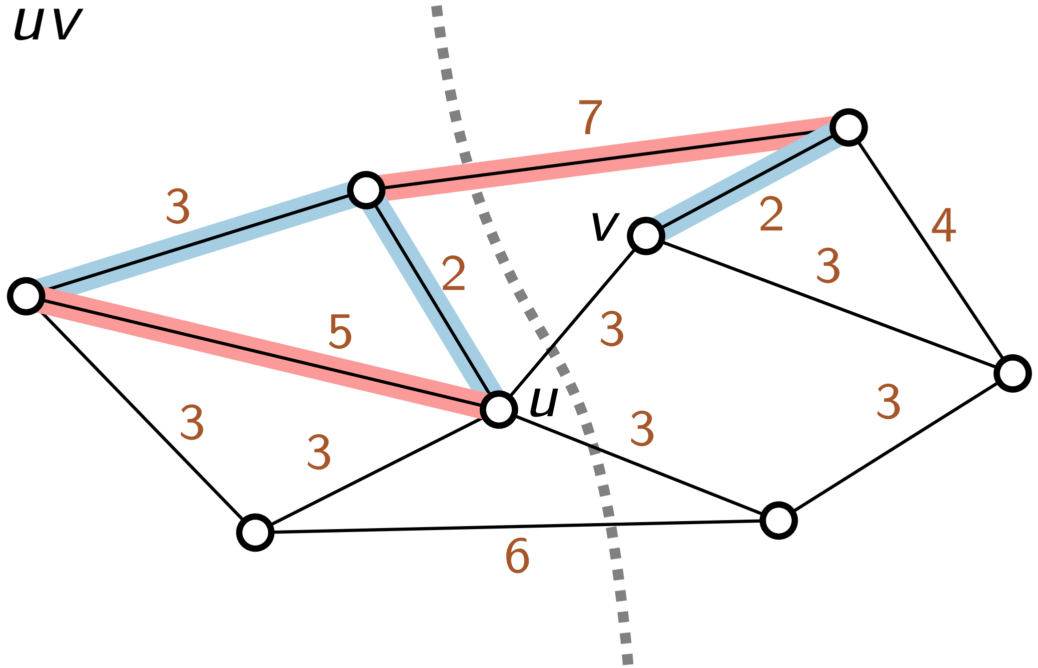
Blaue Regel:
Wähle Schnitt, den keine blaue Kante kreuzt.
Färbe leichte Kante blau.

Lemma. GREEDYSPANNBAUM färbt alle Kanten.

Beweis. Blaue Kanten bilden Wald B (ggfs. isolierte Knoten)
Sei uv ungefärbte Kante. Zu zeigen: uv (oder eine andere Kante!) wird gefärbt.

1. Fall: uv verbindet Knoten *eines* Baumes aus B.
   Wähle Kreis C: Pfad in B von v zu u + Kante uv
   $\Rightarrow$ Kanten auf C alle blau bis auf uv.
   $\Rightarrow$ Rote Regel anwendbar auf uv.

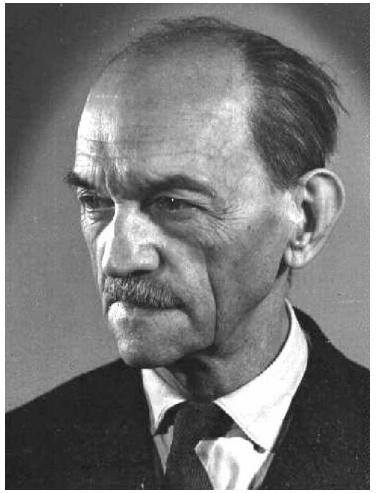
2. Fall: uv verbindet *unterschiedliche* Bäume aus B.
   $\Rightarrow$ Es gibt Schnitt ohne blaue Kanten
   $\Rightarrow$ Blaue Regel anwendbar auf *eine* Kante,
   die den Schnitt kreuzt.



Der Algorithmus von Jarník-Prim (1930/1957)

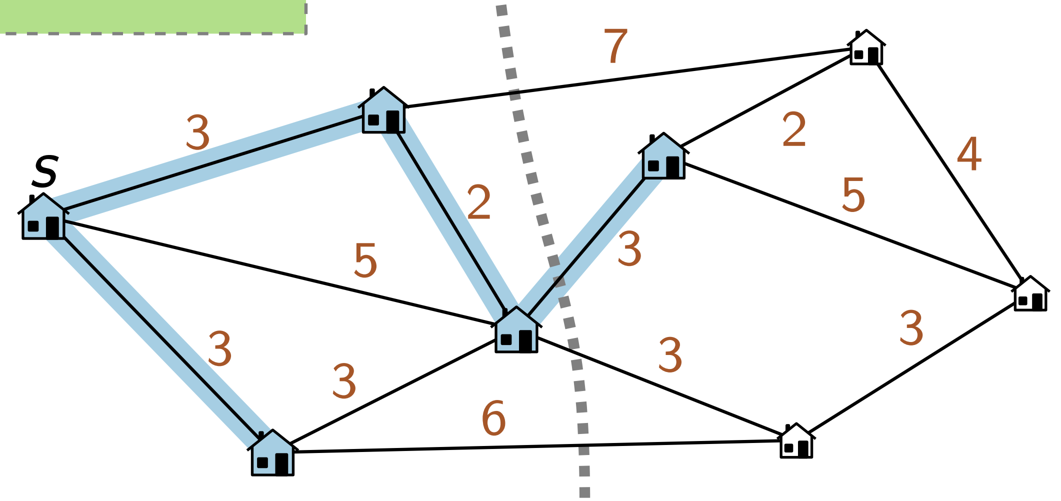
$\textsc{Jarník-Prim}$(Graph G, Weights $w\colon E(G) \to \mathbb{R}_{\geq 0}$), Vertex s)

$S = \{s\}$
$E' = \emptyset$
while not $S == V(G)$ **do**
- Wähle Schnitt $(S, V(G) \setminus S)$. — Blaue Regel
- Färbe leichte Kante uv blau ($u \in S, v \in V(G) \setminus S$).
- $S = S \cup \{v\}$
- $E' = E' \cup \{uv\}$

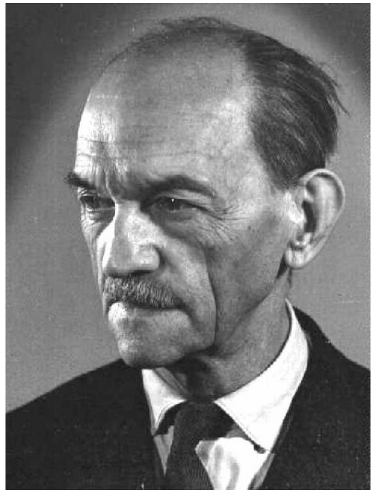


Vojtěch Jarník
* 1897 Prag
† 1970 Prag

Robert C. Prim
* 1921 Sweetwater, TX
† 2021 San Clemente, CA



Der Algorithmus von Jarník-Prim (1930/1957)



Vojtěch Jarník
$*$ 1897 Prag
† 1970 Prag

Robert C. Prim
$*$ 1921 Sweetwater, TX
† 2021 San Clemente, CA

JARNÍK-PRIM(Graph G, Weights $w\colon E(G) \to \mathbb{R}_{\geq 0}$), Vertex s)

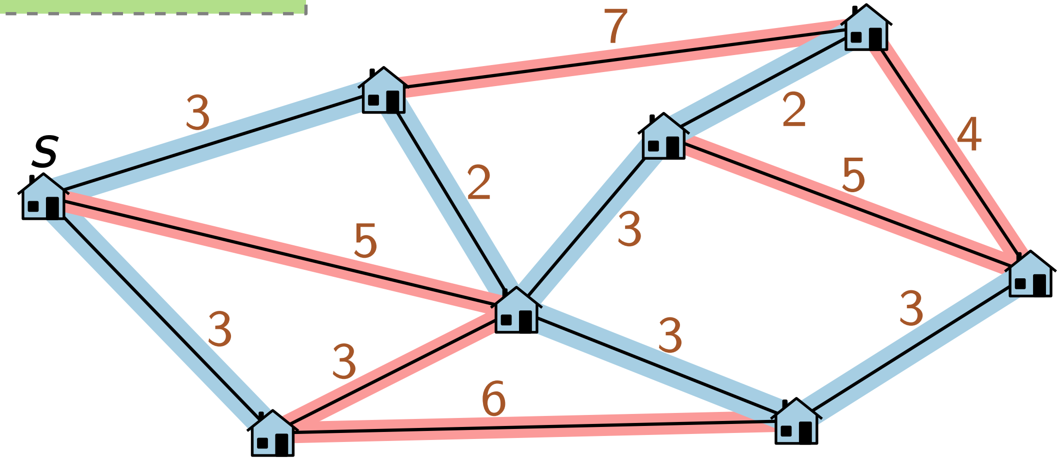
$S = \{s\}$
$E' = \emptyset$
while not $S == V(G)$ **do**
  Wähle Schnitt $(S, V(G) \setminus S)$. — Blaue Regel
  Färbe leichte Kante uv blau $(u \in S, v \in V(G) \setminus S)$.
  $S = S \cup \{v\}$
  $E' = E' \cup \{uv\}$
Färbe alle anderen Kanten rot. — Rote Regel
return E'

Laufzeit?

Wie DIJKSTRA!

$\Rightarrow \mathcal{O}((E + V) \log V)$ — HEAP/RS-BAUM

$\Rightarrow \mathcal{O}(E + V \log V)$ — FIBONACCIHEAP



Der Algorithmus von Kruskal (1956)

$\textsc{Kruskal}(\text{Graph } G, \text{Weights } w \colon E(G) \to \mathbb{R}_{\geq 0})$

$E' = \emptyset$

Sortiere $E(G)$ nicht-absteigend nach Gewicht w.

foreach $uv \in E(G)$ **do** (in sortierter Reihenfolge)

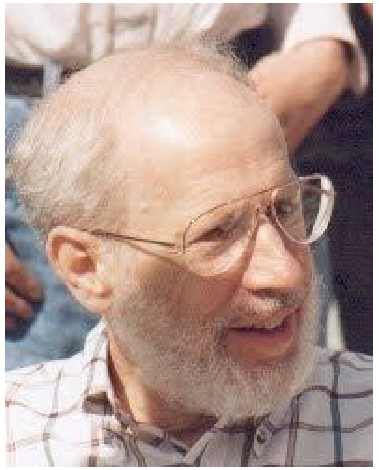
  if $E' \cup \{uv\}$ enthält keinen Kreis **then**

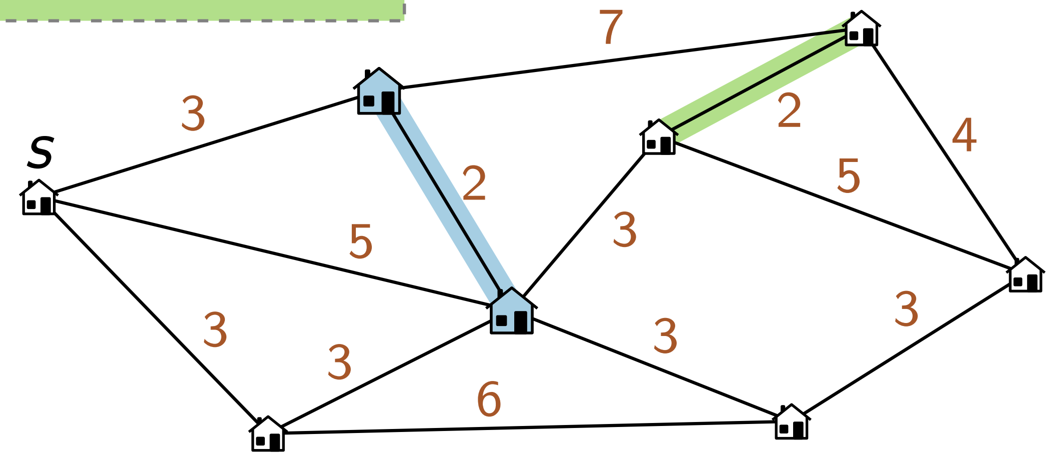
    Färbe uv blau. Blaue Regel

    $E' = E' \cup \{uv\}$

  else

Joseph Bernard Kruskal, Jr.
* 1928 New York, NY
† 2010 Maplewood, NJ



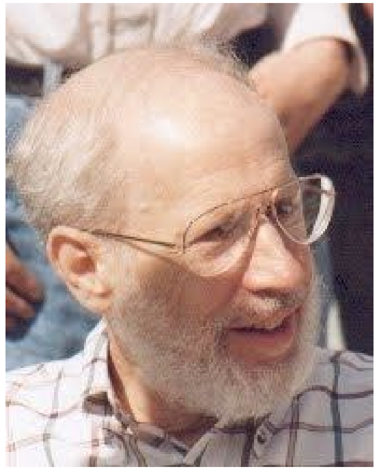


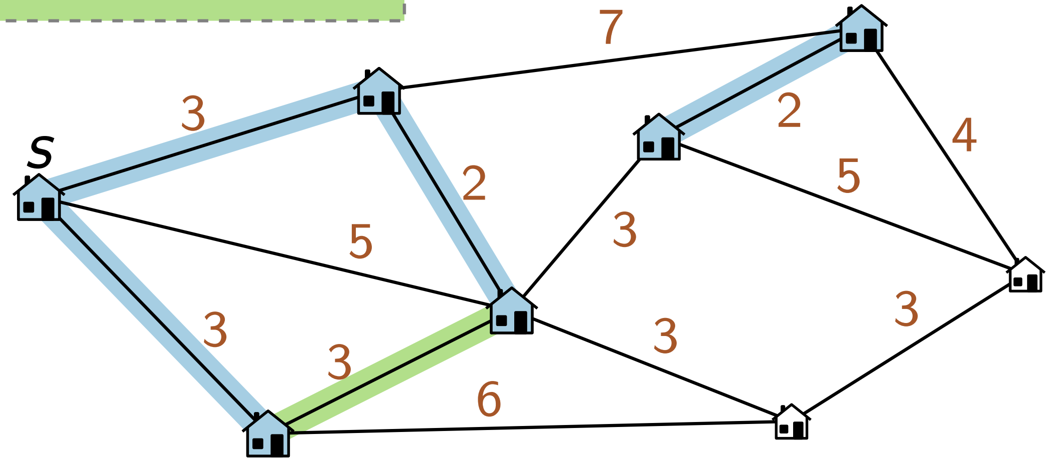
Der Algorithmus von Kruskal (1956)

$\textsc{Kruskal}$(Graph G, Weights $w\colon E(G) \to \mathbb{R}_{\geq 0}$))

  $E' = \emptyset$
  Sortiere $E(G)$ nicht-absteigend nach Gewicht w.
  foreach $uv \in E(G)$ **do** (in sortierter Reihenfolge)
    if $E' \cup \{uv\}$ enthält keinen Kreis **then**
      Färbe uv blau. Blaue Regel
      $E' = E' \cup \{uv\}$
    else
      Färbe uv rot. Rote Regel

Joseph Bernard Kruskal, Jr.
* 1928 New York, NY
† 2010 Maplewood, NJ



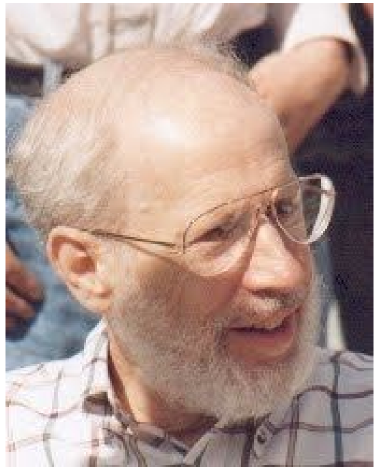


Der Algorithmus von Kruskal (1956)

Kruskal(Graph G, Weights $w \colon E(G) \to \mathbb{R}_{\geq 0}$)

$E' = \emptyset$
Sortiere $E(G)$ nicht-absteigend nach Gewicht w.
foreach $uv \in E(G)$ **do** (in sortierter Reihenfolge)
  if $E' \cup \{uv\}$ enthält keinen Kreis **then**
    Färbe uv blau.   Blaue Regel
    $E' = E' \cup \{uv\}$
  else
    Färbe uv rot.   Rote Regel
return E'

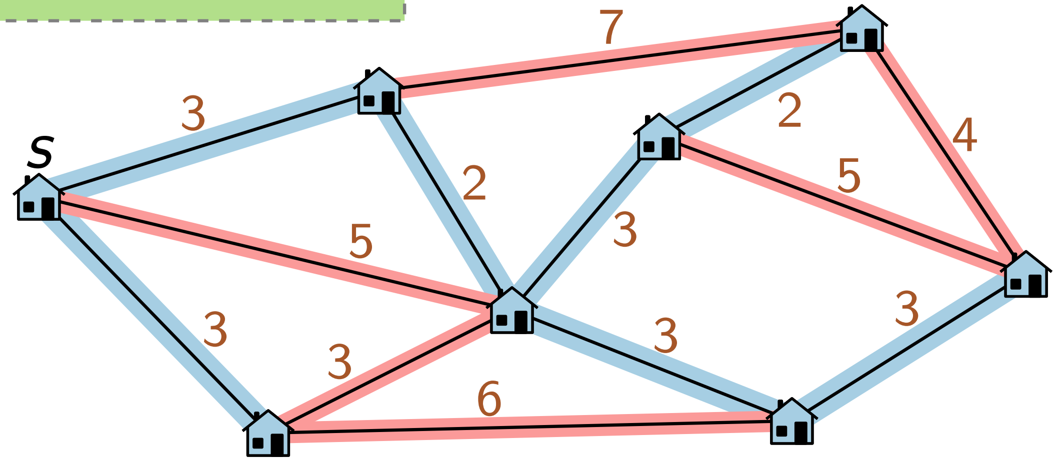
Joseph Bernard Kruskal, Jr.
* 1928 New York, NY
† 2010 Maplewood, NJ



Laufzeit?

$\mathcal{O}(E \log V)$

$\mathcal{O}(E \cdot \alpha(V))$ falls vorsortiert

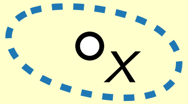
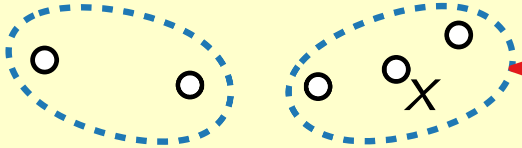
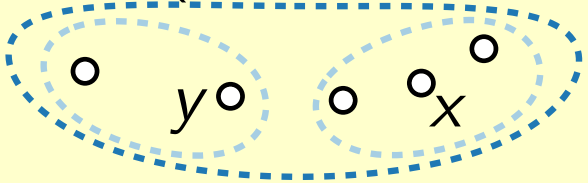


UnionFind-Datenstruktur

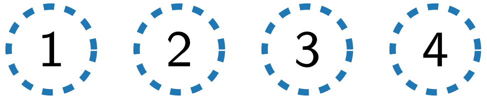
Datenstruktur für **halbdynamische Mengen** (wachsen nur, schrumpfen nicht)

Die halbdynamischen Mengen zerlegen immer eine Grundmenge X. (bei Kruskal: $X = V$)

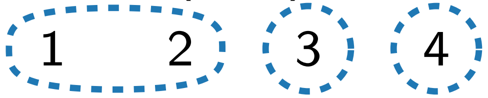
Drei Operationen:

Make(Element x) 	legt die Menge $\{x\}$ an.
Find(Element x) 	liefert (Zeiger auf) die Menge zurück, die momentan x enthält.
Union(Elem. x, Elem. y) 	vereinigt die Mengen, die momentan x und y enthalten.

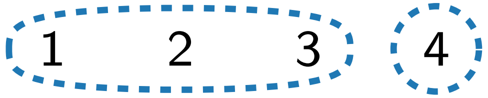
Beispiel.



- Union(1, 2)



- Union(2, 3)



- Find(1) = Find(3)?
  ➡ `true`
- Find(2) = Find(4)?
  ➡ `false`

Anpassung Kruskal

$\textsc{Kruskal}(\text{Graph } G, \text{Weights } w\colon E(G) \to \mathbb{R}_{\geq 0}))$

$E' = \emptyset$ $\quad\quad$ $\forall v \in V(G) : \textsc{Make}(v)$

Sortiere E nicht-absteigend nach Gewicht w

foreach $uv \in E$ **do** (in sortierter Reihenfolge)

  if $E' \cup \{uv\}$ enthält keinen Kreis **then** $\quad\quad$ **if** $\textsc{Find}(u) \neq \textsc{Find}(v)$

    Färbe uv blau $\quad\quad$ Blaue Regel

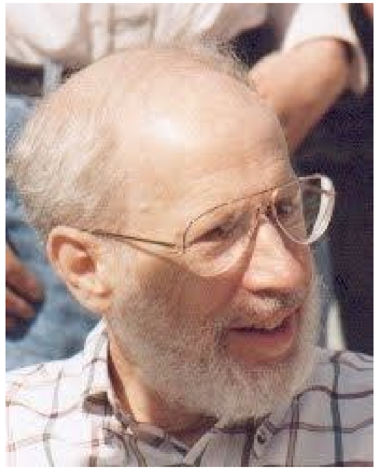
    $E' = E' \cup \{uv\}$ $\quad\quad$ $\textsc{Union}(u, v)$

  else

    Färbe uv rot $\quad\quad$ Rote Regel

return E'

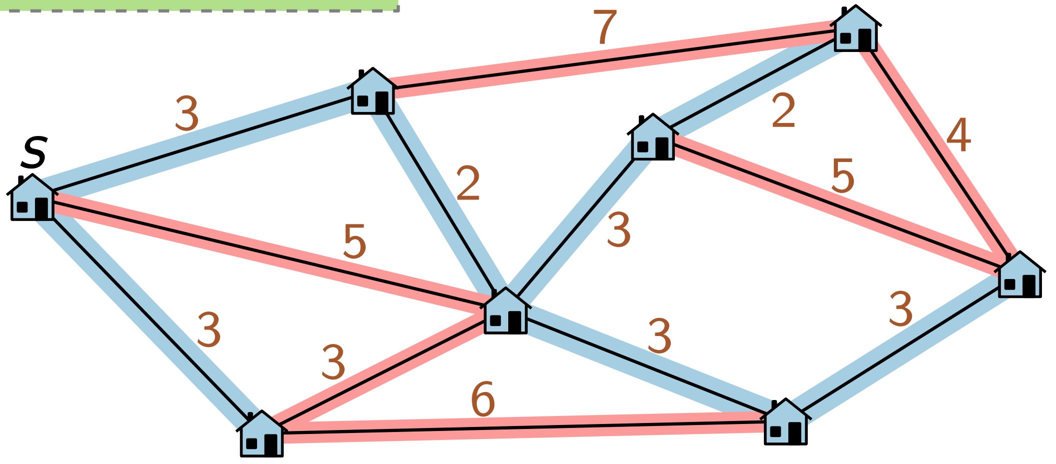
Joseph Bernard Kruskal, Jr.
* 1928 New York, NY
† 2010 Maplewood, NJ



Laufzeit?

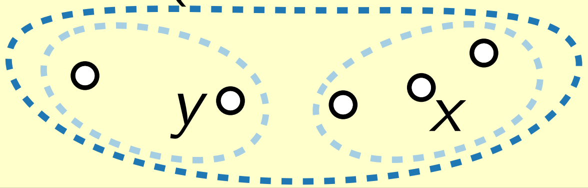
$\mathcal{O}(E \log V)$

$\mathcal{O}(E \cdot \alpha(V))$ falls vorsortiert

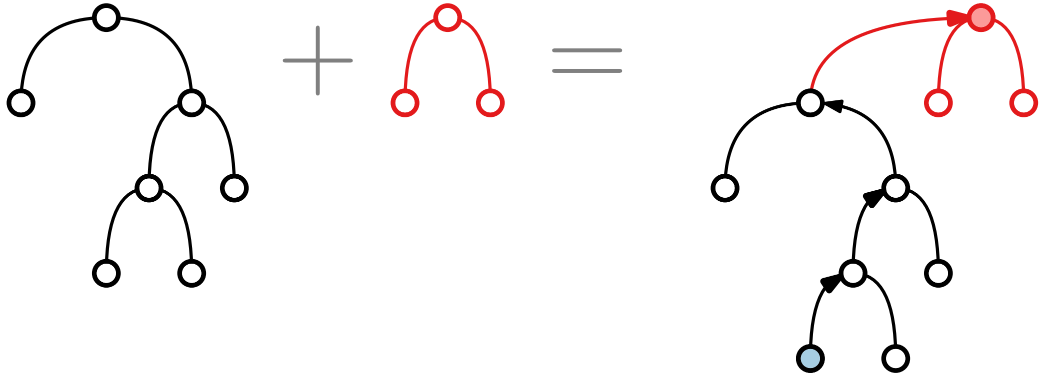


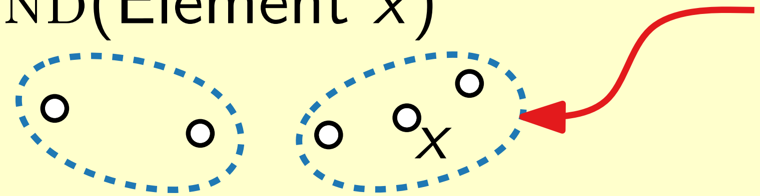
Realisierung Union-Find-Datenstruktur

Baumstruktur für jede Menge

UNION(Elem. x, Elem. y) 	vereinigt die Mengen, die momentan x und y enthalten.

➡ Hänge einen Baum an den anderen:

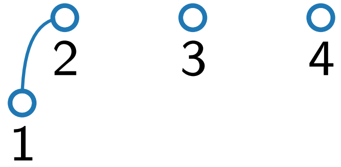


FIND(Element x) 	liefert (Zeiger auf) die Menge zurück, die momentan x enthält.

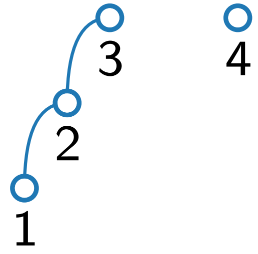
➡ Laufe zur Wurzel, gib Wurzel zurück.

Beispiel.

- UNION(1, 2)



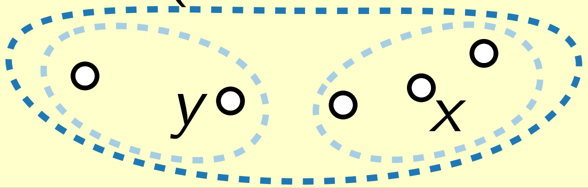
- UNION(2, 3)



- FIND(1)? ➡ 3
- FIND(3)? ➡ 3
- FIND(1) = FIND(3)?
  ➡ `true`

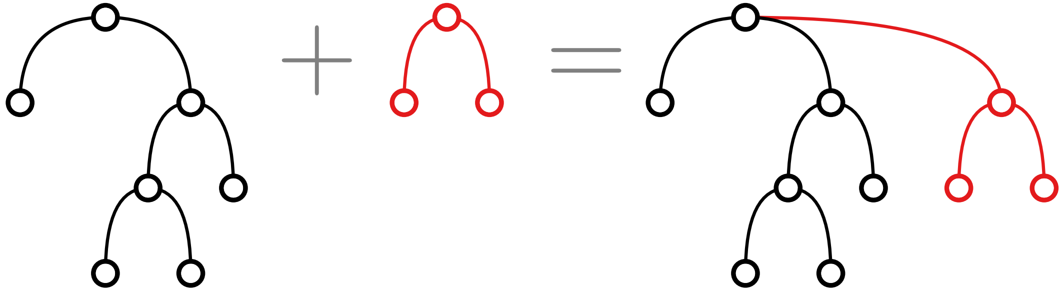
Zwei Verbesserungen

UNION(Elem. x, Elem. y) vereinigt die Mengen, die momentan x und y enthalten.

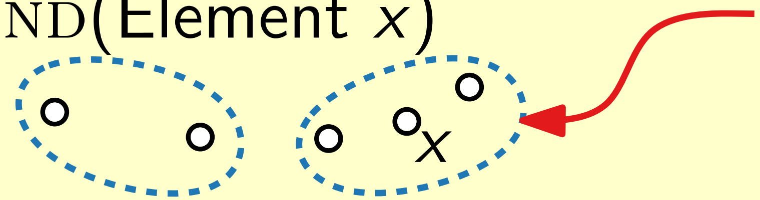


Union-by-Rank: Führe die Op. so aus, dass der neue Baum möglichst geringe Tiefe hat:

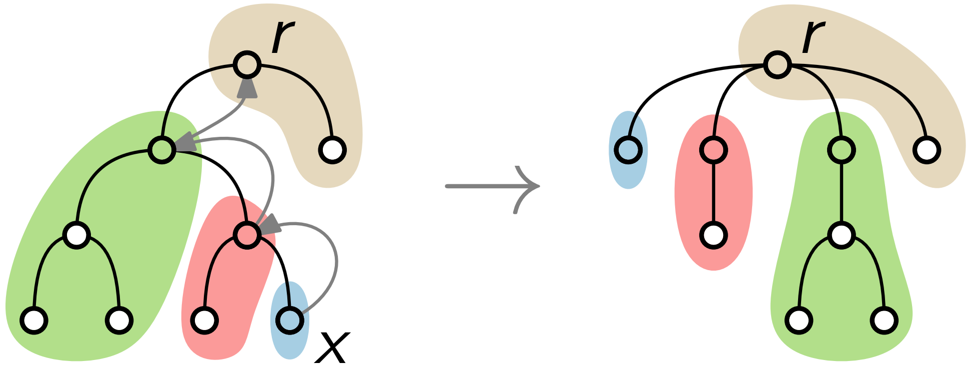
➡ Hänge Baum mit kleinerer Tiefe an den mit größerer.



FIND(Element x) liefert (Zeiger auf) die Menge zurück, die momentan x enthält.



Pfadkompression: Laufe zur Wurzel r, merke alle besuchten Knoten und mache sie zu Kindern von r.



Kosten für Union-Find

Satz. Kosten für $m \times$ Find und $n \times$ Union:

- $\mathcal{O}(n + m \log n)$ mit Union-by-Rank
- $\mathcal{O}(n + m \cdot \alpha(n))$ mit Union-by-Rank und Pfadkompression

$\Omega(n + m \cdot \alpha(n))$ ist untere Schranke für Union-Find [Tarjan '79]

- $\alpha(n)$ ist die inverse Ackermannfunktion
- $\alpha_1(n) = \lceil n/2 \rceil$
- $\alpha_k(n) =$ „wie oft muss ich $\alpha_{k-1}(n)$ auf n anwenden, um auf 1 zu kommen?“
- $\alpha_2(n) = \lceil \log n \rceil$
- $\alpha_3(n) = \log^*(n) = \begin{cases} 0 & \text{wenn } n \leq 1 \\ 1 + \log^*(\log n) & \text{sonst} \end{cases}$
- $\alpha_4(n) = \log^{**}(n) = \begin{cases} 0 & \text{wenn } n \leq 1 \\ 1 + \log^{**}(\log^* n) & \text{sonst} \end{cases}$

...

- $\alpha(n)$ ist das kleinste k, so dass $\alpha_k(n) \leq 3$

z.B. $\log^*(2^{2^{2^2}}) = \log^*(65536) = 4$

$\log^*(2^{2^{2^{2^2}}}) = \log^*(\underbrace{2^{65536}}_{\approx 2 \cdot 10^{19729}}) = 5$

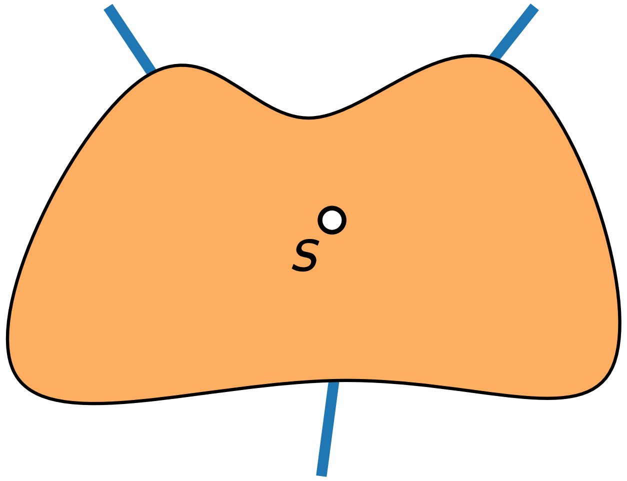
$\alpha(n) \leq 4$ für $n \leq 2^{2^{2^{2^{2^{2^2}}}}} \approx 10^{10^{10^{19729}}}$

$\alpha(n) \leq 5$ für $n \leq 2^{2^{\cdot^{\cdot^{\cdot^2}}}} \Big\} 2^{2^{\cdot^{\cdot^{\cdot^2}}}} \text{ mal} \Big\} 2^{2^{2^2}} \text{ mal}$

Übersicht: Algorithmen für minimale Spannbäume

Jarník-Prim

- geht (wie Dijkstra / BFS) wellenförmig von einem Startknoten aus,
- aktuelle Kantenmenge zusammenhängend,
- Laufzeit $\mathcal{O}(E + V \log V)$.



Kruskal

- bearbeitet Kanten nach aufsteigendem (genauer: nicht-absteig.) Gewicht,
- nach Einfügen der i. Kante gibt es $n - i$ Zusammenhangskomponenten,
- Laufzeit $\mathcal{O}(E \log V)$ oder $\mathcal{O}(E \cdot \alpha(V))$ falls vorsortiert.