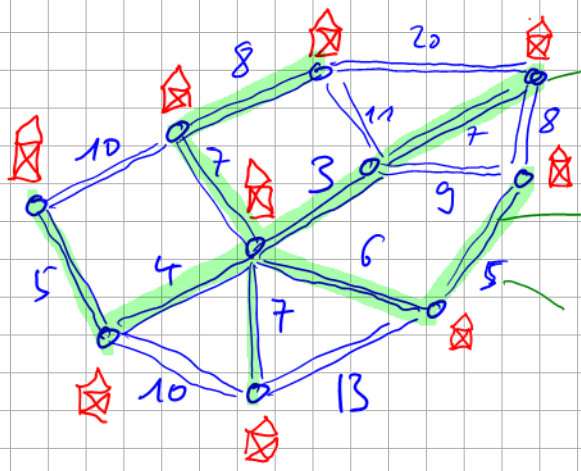
Minimale Spannbäume

gegeben $G = (V, E, c)$ ungerichtet, zusammenhängend

Beispiel



Anschluss/Verteiler V

Wasserleitung E

Kosten (Herstellung/Wartung) c

gesucht: möglichst günstiges Wassernetz, das jedes Haus versorgt

gesucht: Minimaler Spannbaum $T = (V, E')$

(a) Baum: T zusammenhängend und zyklenfrei

(b) Spannbaum: Knotenmenge V

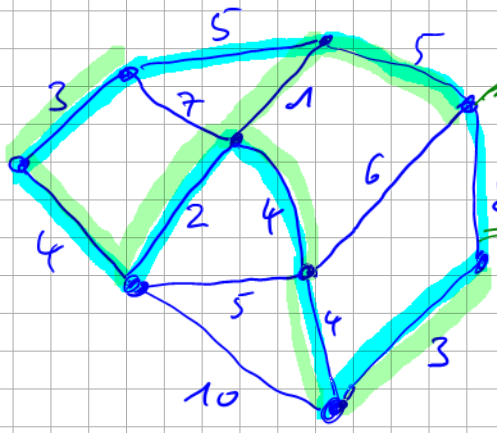
(c) minimal: $c(T) = \sum_{e \in E'} c(e)$ minimal

Problem des Handelsreisenden (TSP)

$c(p) = 38$

$c(T) = 26$

$2 \cdot c(T) = 52$



Stadt V

Entfernung c

Strasse E

gesucht: Rundweg p, der jede Stadt 1x enthält mit minimalem Gewicht c(p)

Hamilton-Pfad: NP-schweres Problem

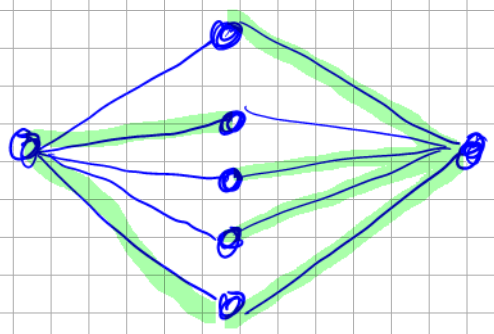
Sei p eine exakte Lösung des TSP. Dann ist p bei dem eine Kante entfernt wird (p') ein Spannbaum.

Der minimale Spannbaum T: $c(T) \leq c(p')$

$2c(T) \leq 2 \cdot c(p') \leq 2 \cdot c(p)$.

Beobachtung:

#Spannbäume kann exponentiell sein



$\Theta(2^n)$ × links/rechts Auswahl

$\Rightarrow \Theta(2^n)$

Algorithmus (Kruskal)

① Sortiere Kanten aufsteigend nach Gewicht

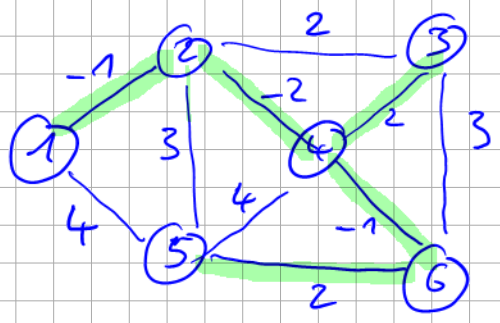
$c(e_1) \leq c(e_2) \leq \ldots \leq c(e_m)$ $(m = |E|)$

② $A \leftarrow \emptyset$

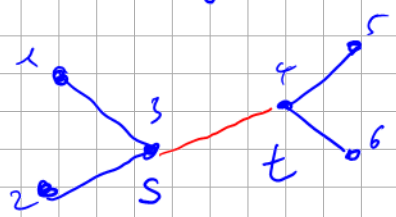
for $k \leftarrow 1$ to m

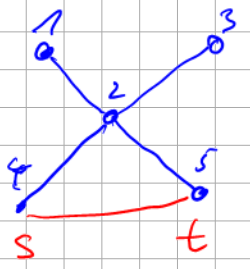
  if $A \cup \{e_k\}$ kreisfrei

    $A \leftarrow A \cup \{e_k\}$



Wie findet man Zyklen?





makeSet(i)

Find(i) Find(s) = Find(t) $\Rightarrow$ $\exists p : s \rightsquigarrow t$

Union(S,T) $\Rightarrow$ Find(s) = Find(t) $\forall s \in S, t \in T$

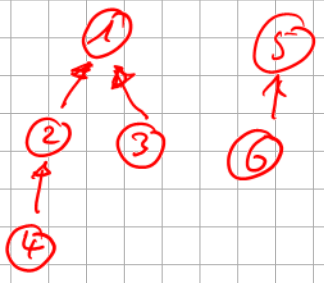
Union-Find-Datenstruktur

Union (Find(1), Find(2))

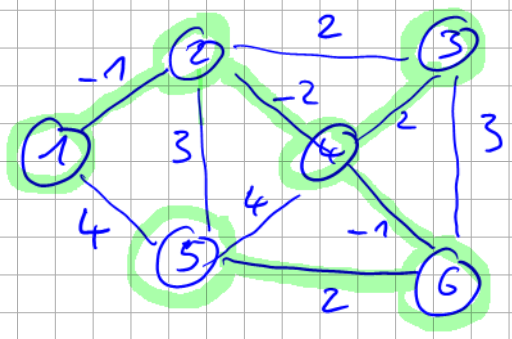
Union (Find(1), Find(3))

als Array

Index	1	2	3	4	5	6
Eintrag	1	1	1	2	6	6



Union (2, 4)


2
-1
-2
2
3
3
4
4
-1
2
1
2
3
4
5
6