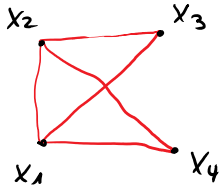
Montag, 15. Juli 2024 10:37

Mathe 2 Vorlesung 15.7.24

Bp.



$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

$$A \cdot A = \begin{pmatrix} 3 & \boxed{2} & 1 & 1 \\ 2 & 3 & 1 & 1 \\ 1 & 1 & 2 & 2 \\ 1 & 1 & 2 & 2 \end{pmatrix}$$

2 Wege von x_1 nach x_2 der Länge 2

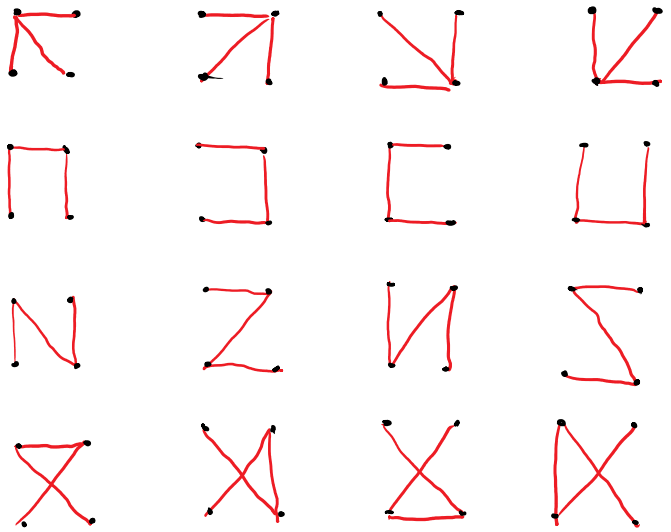
Ein Digraph mit n Knoten ist genau dann zyklenfrei (azyklisch) wenn es ein r mit $1 \leq r \leq n$ gibt und es gilt: $A^r \neq 0$ und $A^s = 0$ für alle $s > r$ (Nullmatrix)

Durchlaufen von Graphen

Def: (aufspannender Baum, Gerüst)

Ein aufspannender Baum eines Graphen enthält keinen Zyklus und ist zusammenhängend (spanning tree)

Frage:  Wieviel aufspannende Bäume besitzt dieser Graph?



Cayley-Formel

Ein vollständiger Graph mit n Knoten hat

n^{n-2}

aufspannende Bäume

Hier: 4 Knoten

$4^{4-2} = 4^2 = 16$ Bäume

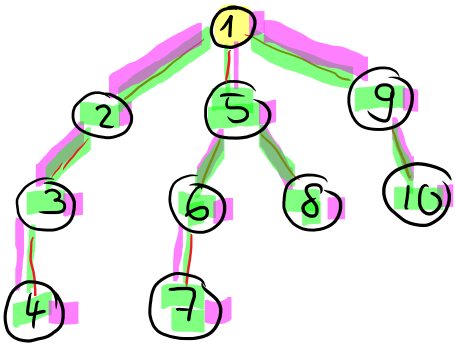
Def: Minimal aufspannender Baum (MST: minimal spanning tree)

T sei ein aufspannender Baum eines bewerteten Graphen G

Dann heißt T MST, wenn die Summe seiner Kantenbewertungen minimal ist.

Erste Suchalgorithmen:

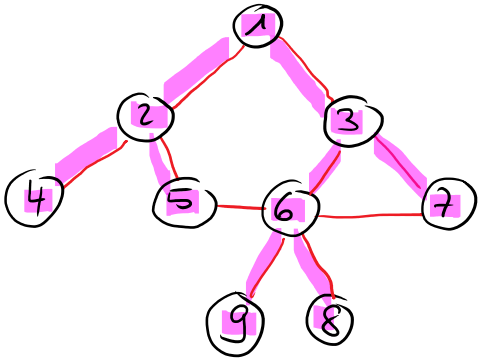
1) Tiefensuche : Startknoten festlegen, als besucht markieren
Nachfolgerknoten suchen, markieren,
wieder Nachfolger suchen
Wenn kein Nachfolger vorhanden, zurück zum
zuletzt besuchten Knoten, dort weiter ....



Startknoten ① Tiefensuche

2) Breitensuche : Startknoten festlegen, als besucht markieren
Markiere nun alle Nachfolgerknoten, wieder markiert
von diesen wieder alle Nachfolgerknoten markieren

Bp:



Breitensuche von ① ab:

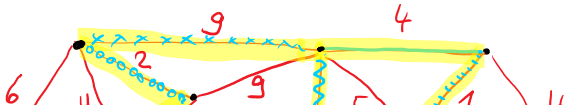
Frage: Wie findet man in einem bewerteten Graphen einen minimal aufspannenden Baum? (MST)

Algorithmus von Kruskal (Joseph Kruskal 1928-2010)
(1956 veröffentlicht)

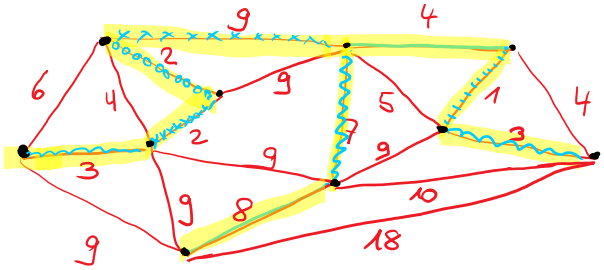
für ungerichtete bewertete Graphen

1) Suche in G (bewertet) die Kante mit der kleinsten Bewertung, falls die Kante mit den schon markierten Kanten einen Zyklus bildet, dann verwerfen, ansonsten hinzufügen

2) Dieser Schritt wird so oft wiederholt, bis nichts mehr hinzugefügt werden kann.



1) Kante mit (1) ""
2) "" (2)



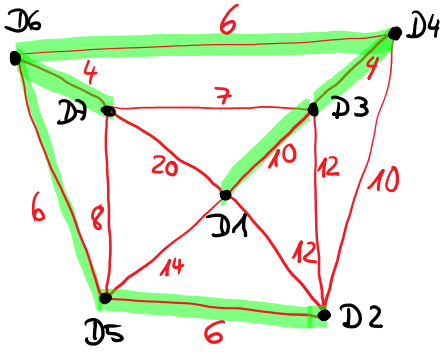
Summe der Kantenbewertung minimal mit: 39

1) Kante mit (1) ||||
2) oooo (2)
3) xxxx (2)
4) ~~ (3)
~~ (3)
5) —— (4)
6) Kante mit 5 ↯
Kante mit 6 ↯
7) ~~~ (7)
8) —— (8)
9 xxxx (9)

In Klammern Kantengewichte

Bp: Sieben Dörfer, die mittels Straßen verbunden werden sollen

	D1	D2	D3	D4	D5	D6	D7
D1	0	12	10	0	14	0	20
D2	12	0	12	10	6	0	0
D3	10	12	0	4	0	0	7
D4	0	10	4	0	0	6	0
D5	14	6	0	0	0	6	8
D6	0	0	0	6	6	0	4
D7	20	0	7	0	8	4	0



Minimale Kosten: 36

Bestimmung kürzester Wege in Graphen

Aufsuchen der kürzesten Wege von einem vorgegebenen Knoten zu allen andern : Single source shortest paths

Aufsuchen der kürzesten Wege zwischen je zwei Knoten
all pairs shortest paths

Fragestellungen sind auch in Digraphen zu lösen

Markierungsalgorithmus von Dijkstra

(Edsger W. Dijkstra (1930-2002))

Lösung zu "single source shortest paths"

Grundidee: man wählt immer diejenige Kante aus, die vom Startknoten aus die kleinste Summe der Kantenbewertungen hat.

Allgemein (Kurzversion)

Geg: $G=(M,K,v,f)$ bewerteter Graph $f(k) \geq 0$ für alle $k \in K$

$D(x_i)$: Bewertung der Distanz vom Startknoten aus

(1) Markiere x_0 (Startknoten) mit $D(x_0)=0$
Setze $i=1$, $M_1 := \{x_0\}$ weiter mit (2)

(2) Sei M^* die Menge aller Knoten x_s der Kanten (x_r, x_s), deren Knoten x_r

in M_i liegt, d.h.

$$M^* = \{ x_s \in M \setminus M_i \mid \text{es gibt eine Kante } (x_r, x_s) \text{ mit } x_r \in M_i \}$$

Menge der benachbarten Knoten zu den bereits markierten Knoten

Falls $M^* = \emptyset$, weiter mit (6)

falls $M^* \neq \emptyset$, weiter mit (3)

(3) Es wird nun zu jedem neuen Nachbarknoten die kürzeste Entfernung vom Startknoten aus über bereits markierte Knoten ermittelt. weiter mit (4)

(4) Markiere diese Kante und bewerte den markierten Knoten

(5) Erweitere die Menge der markierten Knoten

(6) Ende, wenn alle Knoten markiert sind.