23.04.13

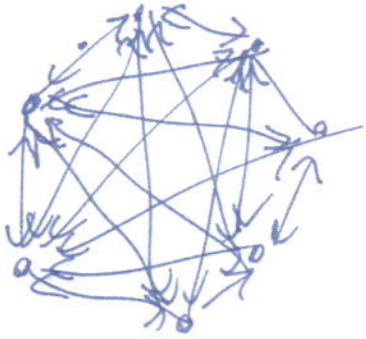
(15)

Beweis:

Wir machen uns klar:

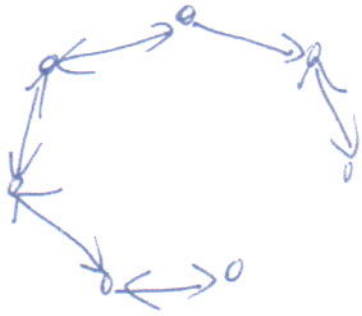
$(a) \Rightarrow (g) \Rightarrow (e) \Rightarrow (d) \Rightarrow (f) \Rightarrow (b) \Rightarrow (c) \Rightarrow (a)$

(Das nennt man „Ringschluss"; praktisch, weil man nicht 21 Äquivalenzen (42 Schlüsse!!



und auch nicht

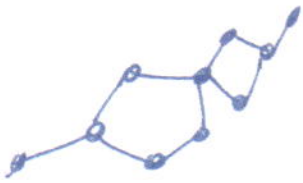
6 Äquivalenzen (12 Schlüsse!)



braucht, sondern nur 7!

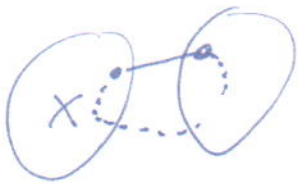
(a) $\Rightarrow$ (g)

Weil T zshgd. ist, muss es je einen Pfad geben; gäbe es zwischen zwei Knoten mehr als einen Pfad, so enthielte die Vereinigung der Pfade einen Kreis, was nicht sein kann, weil T kreisfrei ist.



(g) $\Rightarrow$ (e)

T ist zusammenhängend; könnte man eine Kante entfernen, ohne dass die Eigenschaft aus (e) verloren ginge, müsste es einen zweiten Pfad geben.

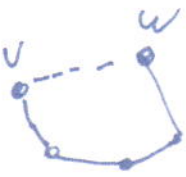


(e) $\Rightarrow$ (d)

Ist klar!

(d) $\Rightarrow$ (f) ~~ist klar~~

Da alle Knoten bereits verbunden sind, liefert jede eingefügte Kante einen Kreis.



(f) $\Rightarrow$ (b)

und (b) $\Rightarrow$ (c)

Wir verwenden Lemma 2.5! (gleiche Aussage)

Man zeigt durch Induktion über m, dass für Wälder mit n Knoten, m Kanten und p Zshs.komp.

(*) $n = m+p$ gilt.

(Wie oben gesehen:
1 Kante hinzufügen reduziert Zshgs.komp. um 1!)

(c) $\Rightarrow$ (a)

Sei T zshgd mit $n-1$ Kanten.

Angenommen, es gibt darin einen Kreis; dann entfernen wir daraus eine Kante. Das machen wir fort, bis keine Kreise mehr existieren, schließlich haben wir k Kanten entfernt.

Der verbleibende Graph ist immer noch zshgd. + kreisfrei und hat $m = (n-1) - k$ Kanten, aber nur $p=1$ Komponente.

Also ist

(*) $n = m+p = (n-1-k) + 1$,

d.h. $k=0$ □

2.3 Berechnung optimaler Bäume

Historisch:
- Borůvka (1926), Kruskal (1956) } „Kruskal"
- Jarník (1930), Prim (1957) } „Prim"

Später mehr!

2.3.1. Kruskals Algorithmus

Algorithmus 2.7 (Kruskal)

Eingabe: G: zshgder. ungerichteter Graph
c: Kantengewichte $E(G) \to \mathbb{R}$

Ausgabe: Aufspannender Baum T minimalen Gewichts

(1) Sortiere die Kanten nach Gewicht, so dass
$$c(e_{\pi(1)}) \leq c(e_{\pi(2)}) \leq \ldots \leq c(e_{\pi(m)})$$

(2) Setze $T := (V(G), \emptyset)$

(3) FOR $i=1$ TO m DO
IF ($T + e_{\pi(i)}$ enthält keinen Kreis) THEN
$T := T + e_{\pi(i)}$ ← Kurznotation für $(V(G), E(T) \cup \{e_{\pi(i)}\})$

Laufzeit ?!

(1) Sortieren der Kanten : $O(m \log m)$
$(= O(m \log n))$

Also Kritisch:

(3) Teste Kreisfreiheit !

→ Problem:
Gegeben : kant. Graph G' mit mindestens $(n-1)$ Kanten, Kante e
Frage : Ist $G'+e$ kreisfrei ?

→ Lösung in $O(n)$ mit DFS oder BFS

Insgesamt m-mal, also $O(mn)$ für diesen Schritt,
also auch für den Algorithmus.

Besser ?!

Satz 2.8
Kruskals Algorithmus lässt sich so implementieren,
dass die Laufzeit $O(m \log m)$ ist.