Studientag zur Algorithmischen Mathematik

Minimale aufspannende Bäume und Matchings

Winfried Hochstättler

Diskrete Mathematik und Optimierung
FernUniversität in Hagen

22. Mai 2011

Outline

Minimale aufspannende Bäume

Problem

Input: zusammenhängender Graph $G = (V, E)$ mit Kantengewichten $w : E \to \mathbb{R}_+$

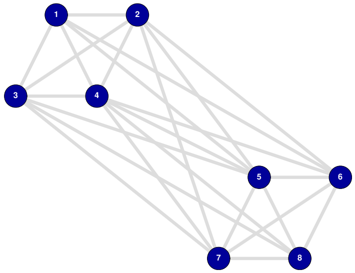
Output: Baum, in dem jeder Knoten vorkommt, mit minimalem Gesamtgewicht.

Minimale aufspannende Bäume

Problem

Input: zusammenhängender Graph $G = (V, E)$ mit Kantengewichten $w : E \to \mathbb{R}_+$

Output: Baum, in dem jeder Knoten vorkommt, mit minimalem Gesamtgewicht.

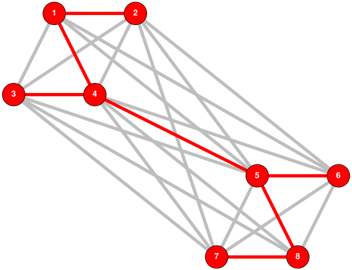


Minimale aufspannende Bäume

Problem

Input: zusammenhängender Graph $G = (V, E)$ mit Kantengewichten $w : E \to \mathbb{R}_+$

Output: Baum, in dem jeder Knoten vorkommt, mit minimalem Gesamtgewicht.



Der Greedy-Algorithmus von Kruskal

Der Algorithmus von Jarník und Prim

Der Algorithmus von Borůvka

wiederhole

(1) jeder Knoten markiert Kante minimalen Gewichts an ihm

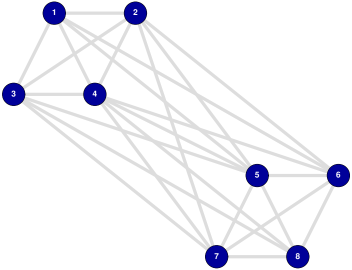
(2) schrumpfe Komponenten zu Knoten

Der Algorithmus von Borůvka

wiederhole

(1) jeder Knoten markiert Kante minimalen Gewichts an ihm

(2) schrumpfe Komponenten zu Knoten

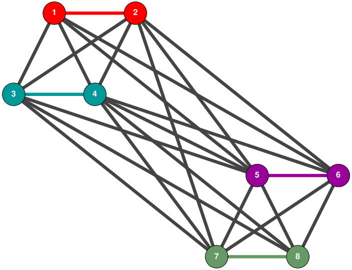


Der Algorithmus von Borůvka

wiederhole

(1) jeder Knoten markiert Kante minimalen Gewichts an ihm

(2) schrumpfe Komponenten zu Knoten

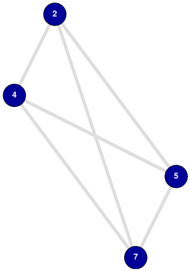


Der Algorithmus von Borůvka

wiederhole

(1) jeder Knoten markiert Kante minimalen Gewichts an ihm

(2) schrumpfe Komponenten zu Knoten

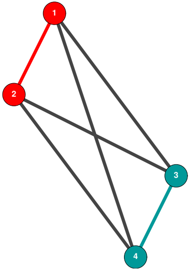


Der Algorithmus von Borůvka

wiederhole

(1) jeder Knoten markiert Kante minimalen Gewichts an ihm

(2) schrumpfe Komponenten zu Knoten



Der Algorithmus von Borůvka

wiederhole

(1) jeder Knoten markiert Kante minimalen Gewichts an ihm

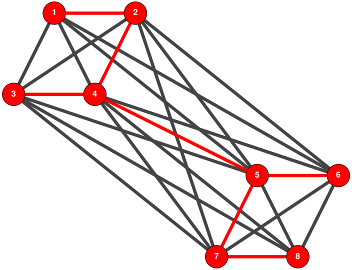
(2) schrumpfe Komponenten zu Knoten

Der Algorithmus von Borůvka

wiederhole

(1) jeder Knoten markiert Kante minimalen Gewichts an ihm

(2) schrumpfe Komponenten zu Knoten



Kreiskriterium und Schnittkriterium

Sei (V, T) ein minimaler aufspannender Baum von $G = (V, E)$.
Für $e \notin T$ sei $C(T, e) :=$ der eindeutige Kreis in $T + e$ (der Fundamentalkreis).

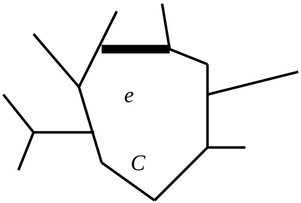
Kreiskriterium und Schnittkriterium

Sei (V, T) ein minimaler aufspannender Baum von $G = (V, E)$. Für $e \notin T$ sei $C(T, e) :=$ der eindeutige Kreis in $T + e$ (der Fundamentalkreis). Für $e \in T$ sei $D(T, e) :=$ alle Kanten zwischen den zwei Komponenten von $T \setminus e$. Wir nennen $D(T, e)$ den Fundamentalschnitt.

Kreiskriterium und Schnittkriterium

Sei (V, T) ein minimaler aufspannender Baum von $G = (V, E)$. Für $e \notin T$ sei $C(T, e) :=$ der eindeutige Kreis in $T + e$ (der Fundamentalkreis). Für $e \in T$ sei $D(T, e) :=$ alle Kanten zwischen den zwei Komponenten von $T \setminus e$. Wir nennen $D(T, e)$ den Fundamentalschnitt.

Kreiskriterium: Jedes $e \in E \setminus T$ hat in $C(T, e)$ maximales Gewicht.

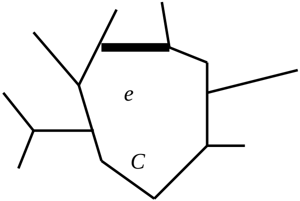


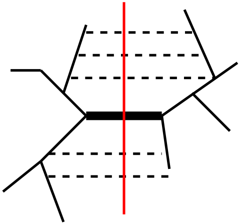
Kreiskriterium und Schnittkriterium

Sei (V, T) ein minimaler aufspannender Baum von $G = (V, E)$. Für $e \notin T$ sei $C(T, e) :=$ der eindeutige Kreis in $T + e$ (der Fundamentalkreis). Für $e \in T$ sei $D(T, e) :=$ alle Kanten zwischen den zwei Komponenten von $T \setminus e$. Wir nennen $D(T, e)$ den Fundamentalschnitt.

Kreiskriterium: Jedes $e \in E \setminus T$ hat in $C(T, e)$ maximales Gewicht.

Schnittkriterium: Jedes $e \in T$ hat in $D(T, e)$ minimales Gewicht.





Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Beweis.

Mit doppeltem Abzählen.

Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Beweis.

Mit doppeltem Abzählen.Wir zählen die knoten- und kantengelabelten Wurzelbäume gelabelten aufspannenden Bäume des K_n.

Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Beweis.

Mit doppeltem Abzählen.Wir zählen die knoten- und kantengelabelten Wurzelbäume gelabelten aufspannenden Bäume des K_n. Jeden knotengelabelten Baum kann man auf n Arten wurzeln und auf $(n-1)!$ Arten kantenlabeln. Die gesuchte Zahl A_n ist also.

$$A_n = n! T_n.$$

Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Beweis.

$$A_n = n!\,T_n.$$

Nun fassen wir die Kantenlabel als Reihenfolge auf, in der wir die Kanten des Wurzelbaumes wählen.

Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Beweis.

$$A_n = n! T_n.$$

Nun fassen wir die Kantenlabel als Reihenfolge auf, in der wir die Kanten des Wurzelbaumes wählen. Den Anfang können wir immer frei wählen, haben also stets n Möglichkeiten.

Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Beweis.

$$A_n = n! T_n.$$

Nun fassen wir die Kantenlabel als Reihenfolge auf, in der wir die Kanten des Wurzelbaumes wählen. Den Anfang können wir immer frei wählen, haben also stets n Möglichkeiten. Für das Ende müssen wir die Wurzel einer anderen Komponente wählen. Dafür haben wir im k-ten Schritt $n - k$ Möglichkeiten.

Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Beweis.

$$A_n = n! T_n.$$

Nun fassen wir die Kantenlabel als Reihenfolge auf, in der wir die Kanten des Wurzelbaumes wählen. Den Anfang können wir immer frei wählen, haben also stets n Möglichkeiten. Für das Ende müssen wir die Wurzel einer anderen Komponente wählen. Dafür haben wir im k-ten Schritt $n - k$ Möglichkeiten. Also ist $A_n = n^{n-1} \cdot (n-1)!$.

Die Cayley-Formel

Satz

Die Anzahl T_n der knotengelabelten aufspannenden Bäume des K_n ist n^{n-2}.

Beweis.

$$A_n = n!T_n.$$

Nun fassen wir die Kantenlabel als Reihenfolge auf, in der wir die Kanten des Wurzelbaumes wählen. Den Anfang können wir immer frei wählen, haben also stets n Möglichkeiten. Für das Ende müssen wir die Wurzel einer anderen Komponente wählen. Dafür haben wir im k-ten Schritt $n - k$ Möglichkeiten. Also ist $A_n = n^{n-1} \cdot (n-1)!$. $\square$

Bipartites Matching

Ein Graph $G = (U \dot{\cup} V, E)$ heißt bipartit, wenn $\forall u, u' \in U : \{u, u'\} \notin E$ und $\forall v, v' \in V : \{v, v'\} \notin E$.

Bipartites Matching

Ein Graph $G = (U \dot{\cup} V, E)$ heißt bipartit, wenn $\forall u, u' \in U : \{u, u'\} \notin E$ und $\forall v, v' \in V : \{v, v'\} \notin E$. Eine Menge von paarweise nicht adjazenten Kanten heißt Matching. Ein Matching ist maximal, wenn es die größtmögliche Anzahl Kanten hat, es ist perfekt, wenn alle Knoten getroffen werden.

Bipartites Matching

Ein Graph $G = (U \dot\cup V, E)$ heißt bipartit, wenn $\forall u, u' \in U : \{u, u'\} \notin E$ und $\forall v, v' \in V : \{v, v'\} \notin E$. Eine Menge von paarweise nicht adjazenten Kanten heißt Matching. Ein Matching ist maximal, wenn es die größtmögliche Anzahl Kanten hat, es ist perfekt, wenn alle Knoten getroffen werden.

Ein *M*-alternierender Weg, ist ein Weg, der abwechselnd Matching- und Nichtmatchingkanten benutzt. Ein *M*-augmentierender Weg ist ein M-alternierender Weg, der in nicht gematcheten Knoten beginnt und endet.

Bipartites Matching

Ein Graph $G = (U \dot{\cup} V, E)$ heißt bipartit, wenn $\forall u, u' \in U : \{u, u'\} \notin E$ und $\forall v, v' \in V : \{v, v'\} \notin E$. Eine Menge von paarweise nicht adjazenten Kanten heißt Matching. Ein Matching ist maximal, wenn es die größtmögliche Anzahl Kanten hat, es ist perfekt, wenn alle Knoten getroffen werden.

Ein *M*-alternierender Weg, ist ein Weg, der abwechselnd Matching- und Nichtmatchingkanten benutzt. Ein *M*-augmentierender Weg ist ein M-alternierender Weg, der in nicht gematcheten Knoten beginnt und endet.

Eine Knotenüberdeckung ist eine Menge von Knoten, die von jeder Kante mindestens einen Endknoten enthält.

Ein Matchingalgorithmus

Satz

Sei M ein Matching in einem bipartiten Graphen. Dann gilt:

$$M \text{ ist } \textit{maximal} \iff \nexists M\text{-augmentierenden Weg}$$

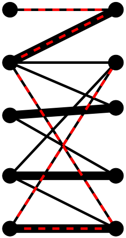
Ein Matchingalgorithmus

Satz

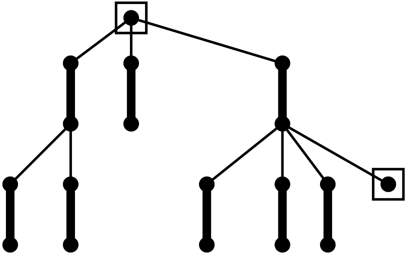
Sei M ein Matching in einem bipartiten Graphen. Dann gilt:

$$M \text{ ist } \textit{maximal} \iff \nexists M\text{-augmentierenden Weg}$$

Beispiel:



Prinzip Matching-Algorithmus:



Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$\max\{|M| \mid M \text{ ist Matching}\} = \min\{|C| \mid C \text{ ist Knotenüberdeckung}\}.$$

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$\max\{|M| \mid M \text{ ist Matching}\} = \min\{|C| \mid C \text{ ist Knotenüberdeckung}\}.$$

Beweis.

Offensichtlich kann kein Matching größer als eine Knotenüberdeckung sein. Ist eine Farbklasse ganz gematched, so fertig.

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$\max\{|M| \mid M \text{ ist Matching}\} = \min\{|C| \mid C \text{ ist Knotenüberdeckung}\}.$$

Beweis.

Andernfalls

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ *bipartit. Dann gilt:*

$$\max\{|M| \mid M \text{ ist Matching}\} = \min\{|C| \mid C \text{ ist Knotenüberdeckung}\}.$$

Beweis.

Andernfalls sei U_2 der Menge der im Algorithmus nicht erreichten Knoten von U und V_2 die Menge der erreichten Knoten von V_1.

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ *bipartit. Dann gilt:*

$$\max\{|M| \mid M \textit{ ist Matching}\} = \min\{|C| \mid C \textit{ ist Knotenüberdeckung}\}.$$

Beweis.

Andernfalls sei U_2 der Menge der im Algorithmus nicht erreichten Knoten von U und V_2 die Menge der erreichten Knoten von V_1. Dann gibt es keine Kanten von $U \setminus U_2$ nach $V \setminus V_1$, und keine Matchingkante von V_1 nach U_2.

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$\max\{|M| \mid M \textit{ ist Matching}\} = \min\{|C| \mid C \textit{ ist Knotenüberdeckung}\}.$$

Beweis.

Andernfalls sei U_2 der Menge der im Algorithmus nicht erreichten Knoten von U und V_2 die Menge der erreichten Knoten von V_1. Dann gibt es keine Kanten von $U \setminus U_2$ nach $V \setminus V_1$, und keine Matchingkante von V_1 nach U_2. Andererseits sind alle Knoten von V_1, U_2 gematched.

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$\max\{|M| \mid M \text{ ist Matching}\} = \min\{|C| \mid C \text{ ist Knotenüberdeckung}\}.$$

Beweis.

Andernfalls sei U_2 der Menge der im Algorithmus nicht erreichten Knoten von U und V_2 die Menge der erreichten Knoten von V_1. Dann gibt es keine Kanten von $U \setminus U_2$ nach $V \setminus V_1$, und keine Matchingkante von V_1 nach U_2. Andererseits sind alle Knoten von V_1, U_2 gematched. Also ist $C = V_1 \cup U_2$ eine Knotenüberdeckung und $|C| = |M|$. $\square$

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$\max\{|M| \mid M \text{ ist Matching}\} = \min\{|C| \mid C \text{ ist Knotenüberdeckung}\}.$$

Satz (Heiratssatz von Frobenius))

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$G \text{ hat perfektes Matching} \iff |U| = |V| \wedge \forall H \subseteq U : |N(H)| \geq |H|.$$

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$\max\{|M| \mid M \text{ ist Matching}\} = \min\{|C| \mid C \text{ ist Knotenüberdeckung}\}.$$

Satz (Heiratssatz von Frobenius))

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$G \text{ hat perfektes Matching} \iff |U| = |V| \wedge \forall H \subseteq U : |N(H)| \geq |H|.$$

Beweis.

Falls es kein perfektes Matching gibt, so ist $|M| = |C \cap U| + |C \cap V| < n = |C \cap V| + |V \setminus C|$.

Die Sätze von König und Frobenius

Satz (König)

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$\max\{|M| \mid M \text{ ist Matching}\} = \min\{|C| \mid C \text{ ist Knotenüberdeckung}\}.$$

Satz (Heiratssatz von Frobenius))

Sei $G = (U \dot\cup V, E)$ bipartit. Dann gilt:

$$G \text{ hat perfektes Matching} \iff |U| = |V| \wedge \forall H \subseteq U : |N(H)| \geq |H|.$$

Beweis.

Falls es kein perfektes Matching gibt, so ist $|M| = |C \cap U| + |C \cap V| < n = |C \cap V| + |V \setminus C|$. Also ist $|C \cap U| < |V \setminus C| \leq |N(C \cap U)|$. □

Stabile Hochzeiten