WS 2003/04

Diskrete Strukturen I

Ernst W. Mayr

mayr@in.tum.de
Institut für Informatik
Technische Universität München

02-06-2004

Beispiel

Sei

$$W = \begin{pmatrix} 9 & 11 & 12 & 11 \\ 6 & 3 & 8 & 5 \\ 7 & 6 & 13 & 11 \\ 9 & 10 & 10 & 7 \end{pmatrix}$$

Nachdem wir von jeder Zeile das minimale Gewicht subtrahieren, erhalten wir

$$W' = \begin{pmatrix} 0 & 2 & 3 & 2 \\ 3 & 0 & 5 & 2 \\ 1 & 0 & 7 & 5 \\ 2 & 3 & 3 & 0 \end{pmatrix}$$

Beispiel

Sei

$$W = \begin{pmatrix} 9 & 11 & 12 & 11 \\ 6 & 3 & 8 & 5 \\ 7 & 6 & 13 & 11 \\ 9 & 10 & 10 & 7 \end{pmatrix}$$

Nachdem wir von jeder Zeile das minimale Gewicht subtrahieren, erhalten wir

$$W' = \begin{pmatrix} 0 & 2 & 3 & 2 \\ 3 & 0 & 5 & 2 \\ 1 & 0 & 7 & 5 \\ 2 & 3 & 3 & 0 \end{pmatrix}$$

$$W' = \begin{pmatrix} 0 & 2 & 3 & 2 \\ 3 & 0 & 5 & 2 \\ 1 & 0 & 7 & 5 \\ 2 & 3 & 3 & 0 \end{pmatrix}$$

Nachdem wir von jeder Spalte das minimale Gewicht subtrahieren, erhalten wir

$$W'' = \begin{pmatrix} 0 & 2 & 0 & 2 \\ 3 & 0 & 2 & 2 \\ 1 & 0 & 4 & 5 \\ 2 & 3 & 0 & 0 \end{pmatrix}$$

$$W' \quad = \quad \begin{pmatrix} 0 & 2 & 3 & 2 \\ 3 & 0 & 5 & 2 \\ 1 & 0 & 7 & 5 \\ 2 & 3 & 3 & 0 \end{pmatrix}$$

Nachdem wir von jeder Spalte das minimale Gewicht subtrahieren, erhalten wir

$$W'' \quad = \quad \begin{pmatrix} 0 & 2 & 0 & 2 \\ 3 & 0 & 2 & 2 \\ 1 & 0 & 4 & 5 \\ 2 & 3 & 0 & 0 \end{pmatrix}$$

Diese Matrix enthält eine Diagonale der Größe 3 mit Einträgen $= 0$:

$$W'' = \begin{pmatrix} \boxed{0} & 2 & 0 & 2 \\ 3 & 0 & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ 2 & 3 & \boxed{0} & 0 \end{pmatrix}$$

Aus Satz *17 folgt, dass die maximale Länge einer* 0*-Diagonale gleich der minimalen Anzahl von Zeilen und Spalten ist, die alle* 0*en bedecken. Falls wir noch keine* 0*-Diagonale der Länge* n *haben, iterieren wir folgenden Algorithmus:*

Diese Matrix enthält eine Diagonale der Größe 3 mit Einträgen $= 0$:

$$W'' \quad = \quad \begin{pmatrix} \boxed{0} & 2 & 0 & 2 \\ 3 & 0 & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ 2 & 3 & \boxed{0} & 0 \end{pmatrix}$$

Aus Satz *17 folgt, dass die maximale Länge einer 0-Diagonale*

gleich der minimalen Anzahl von Zeilen und Spalten ist, die alle 0en bedecken. *Falls wir noch keine 0-Diagonale der Länge n haben, iterieren wir folgenden Algorithmus:*

Diese Matrix enthält eine Diagonale der Größe 3 mit Einträgen $= 0$:

$$W'' = \begin{pmatrix} \boxed{0} & 2 & 0 & 2 \\ 3 & 0 & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ 2 & 3 & \boxed{0} & 0 \end{pmatrix}$$

Aus Satz *17 folgt, dass die maximale Länge einer* 0*-Diagonale gleich der minimalen Anzahl von Zeilen und Spalten ist, die alle* 0*en bedecken. Falls wir noch keine* 0*-Diagonale der Länge* n *haben, iterieren wir folgenden Algorithmus:*

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten.

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten.

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

1. finde eine minimale Anzahl von e Zeilen und f Spalten $(e + f < n)$, die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten.

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

1. finde eine minimale Anzahl von e Zeilen und f Spalten $(e + f < n)$, die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten.

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten.

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten.

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten.

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

Insbesondere sind die resultierenden Gewichte wieder ≥ 0. Die Anzahl der doppelt (von Zeilen *und* Spalten) überdeckten Positionen ist $e \cdot f$, die Anzahl der nicht überdeckten Positionen ist

$$n^2 - n(e+f) + ef\ .$$

Der resultierende Gewichtsunterschied ist daher

$$\begin{aligned}\Delta w &= (ef)w - (n^2 - n(e+f) + ef)w \\ &= (n(e+f) - n^2)w < 0\end{aligned}$$

Damit muss unsere Iteration enden und wir finden eine 0-Diagonale der Länge n, entsprechend einer optimalen Zuordnung.

Insbesondere sind die resultierenden Gewichte wieder ≥ 0.Die Anzahl der doppelt (von Zeilen *und* Spalten) überdeckten Positionen ist $e \cdot f$, die Anzahl der nicht überdeckten Positionen ist

$$n^2 - n(e+f) + ef \ .$$

Der resultierende Gewichtsunterschied ist daher

$$\begin{aligned} \Delta w &= (ef)w - (n^2 - n(e+f) + ef)w \\ &= (n(e+f) - n^2)w < 0 \end{aligned}$$

Damit muss unsere Iteration enden und wir finden eine 0-Diagonale der Länge n, entsprechend einer optimalen Zuordnung.

Insbesondere sind die resultierenden Gewichte wieder ≥ 0.Die Anzahl der doppelt (von Zeilen *und* Spalten) überdeckten Positionen ist $e \cdot f$, die Anzahl der nicht überdeckten Positionen ist

$$n^2 - n(e+f) + ef \ .$$

Der resultierende Gewichtsunterschied ist daher

$$\begin{aligned} \Delta w &= (ef)w - (n^2 - n(e+f) + ef)w \\ &= (n(e+f) - n^2)w < 0 \end{aligned}$$

Damit muss unsere Iteration enden und wir finden eine 0-Diagonale der Länge n, entsprechend einer optimalen Zuordnung.

Insbesondere sind die resultierenden Gewichte wieder ≥ 0.Die Anzahl der doppelt (von Zeilen *und* Spalten) überdeckten Positionen ist $e \cdot f$, die Anzahl der nicht überdeckten Positionen ist

$$n^2 - n(e + f) + ef \ .$$

Der resultierende Gewichtsunterschied ist daher

$$\begin{aligned} \Delta w &= (ef)w - (n^2 - n(e + f) + ef)w \\ &= (n(e + f) - n^2)w < 0 \end{aligned}$$

Damit muss unsere Iteration enden und wir finden eine 0-Diagonale der Länge n, entsprechend einer optimalen Zuordnung.

In unserem Beispiel ergibt sich

$$W'' = \begin{pmatrix} \boxed{0} & \boxed{2} & \boxed{0} & \boxed{2} \\ 3 & \boxed{0} & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ \boxed{2} & \boxed{3} & \boxed{0} & \boxed{0} \end{pmatrix}$$

Der Algorithmus bestimmt $w = 1$:

$$\Rightarrow \begin{pmatrix} 0 & 2 & 0 & 2 \\ 2 & -1 & 1 & 1 \\ 0 & -1 & 3 & 4 \\ 2 & 3 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 3 & \boxed{0} & 2 \\ 2 & \boxed{0} & 1 & 1 \\ \boxed{0} & 0 & 3 & 4 \\ 2 & 4 & 0 & \boxed{0} \end{pmatrix}$$

In unserem Beispiel ergibt sich

$$W'' = \begin{pmatrix} \boxed{0} & \boxed{2} & \boxed{0} & \boxed{2} \\ 3 & \boxed{0} & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ \boxed{2} & \boxed{3} & \boxed{0} & \boxed{0} \end{pmatrix}$$

Der Algorithmus bestimmt $w = 1$:

$$\Rightarrow \begin{pmatrix} 0 & 2 & 0 & 2 \\ 2 & -1 & 1 & 1 \\ 0 & -1 & 3 & 4 \\ 2 & 3 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 3 & \boxed{0} & 2 \\ 2 & \boxed{0} & 1 & 1 \\ \boxed{0} & 0 & 3 & 4 \\ 2 & 4 & 0 & \boxed{0} \end{pmatrix}$$

In unserem Beispiel ergibt sich

$$W'' = \begin{pmatrix} \boxed{0} & \boxed{2} & \boxed{0} & \boxed{2} \\ 3 & \boxed{0} & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ \boxed{2} & \boxed{3} & \boxed{0} & \boxed{0} \end{pmatrix}$$

Der Algorithmus bestimmt $w = 1$:

$$\Rightarrow \begin{pmatrix} 0 & 2 & 0 & 2 \\ 2 & -1 & 1 & 1 \\ 0 & -1 & 3 & 4 \\ 2 & 3 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 3 & \boxed{0} & 2 \\ 2 & \boxed{0} & 1 & 1 \\ \boxed{0} & 0 & 3 & 4 \\ 2 & 4 & 0 & \boxed{0} \end{pmatrix}$$

In dem durch die Matrix

$$W = \begin{pmatrix} 9 & 11 & 12 & 11 \\ 6 & 3 & 8 & 5 \\ 7 & 6 & 13 & 11 \\ 9 & 10 & 10 & 7 \end{pmatrix}$$

gegebenen bipartiten Graphen hat also das durch die markierten Kanten gegebene perfekte Matching minimales Gewicht.

Bemerkung: Bei geeigneter Implementierung ist die Laufzeit des Algorithmus $O(n^3)$.

In dem durch die Matrix

$$W = \begin{pmatrix} 9 & 11 & \boxed{12} & 11 \\ 6 & \boxed{3} & 8 & 5 \\ \boxed{7} & 6 & 13 & 11 \\ 9 & 10 & 10 & \boxed{7} \end{pmatrix}$$

gegebenen bipartiten Graphen hat also das durch die markierten Kanten gegebene perfekte Matching minimales Gewicht.

Bemerkung: Bei geeigneter Implementierung ist die Laufzeit des Algorithmus $O(n^3)$.

In dem durch die Matrix

$$W = \begin{pmatrix} 9 & 11 & \boxed{12} & 11 \\ 6 & \boxed{3} & 8 & 5 \\ \boxed{7} & 6 & 13 & 11 \\ 9 & 10 & 10 & \boxed{7} \end{pmatrix}$$

gegebenen bipartiten Graphen hat also das durch die markierten Kanten gegebene perfekte Matching minimales Gewicht.

Bemerkung: Bei geeigneter Implementierung ist die Laufzeit des Algorithmus $O(n^3)$.

Das Problem des chinesischen Postboten

Gegeben ist ein zusammenhängender, gewichteter Multigraph $G = (V, E, w)$.

Gesucht ist ein Kreis minimalen Gewichts, der jede Kante mindestens einmal enthält.

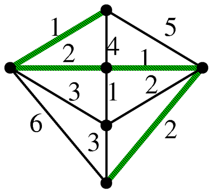
Beispiel ((In der optimalen Lösung werden die dickeren grünen Kanten zweimal verwendet))

Das Problem des chinesischen Postboten

Gegeben ist ein zusammenhängender, gewichteter Multigraph $G = (V, E, w)$.

Gesucht ist ein Kreis minimalen Gewichts, der jede Kante mindestens einmal enthält.

Beispiel ((In der optimalen Lösung werden die dickeren grünen Kanten zweimal verwendet))



Algorithmus: Sei U die Menge der Knoten ungeraden Grades, $|U| = 2k$.

1. Bestimme $\mathsf{d}(u, v)$ für alle $u, v \in U$.
2. Bestimme auf dem K_{2k} mit Kantengewichtung $w(\{u, v\}) = \mathsf{d}(u, v)$ ein perfektes Matching M minimalen Gewichts.
3. Füge die den Kanten in M entsprechenden kürzesten Pfade in G ein und bestimme im resultierenden Graphen einen Eulerkreis. Dieser ist eine Lösung.

Algorithmus: Sei U die Menge der Knoten ungeraden Grades, $|U| = 2k$.

1. Bestimme $\mathsf{d}(u, v)$ für alle $u, v \in U$.
2. Bestimme auf dem K_{2k} mit Kantengewichtung $w(\{u, v\}) = \mathsf{d}(u, v)$ ein perfektes Matching M minimalen Gewichts.
3. Füge die den Kanten in M entsprechenden kürzesten Pfade in G ein und bestimme im resultierenden Graphen einen Eulerkreis. Dieser ist eine Lösung.

Algorithmus: Sei U die Menge der Knoten ungeraden Grades, $|U| = 2k$.

1. Bestimme $\mathsf{d}(u,v)$ für alle $u, v \in U$.
2. Bestimme auf dem K_{2k} mit Kantengewichtung $w(\{u,v\}) = \mathsf{d}(u,v)$ ein perfektes Matching M minimalen Gewichts.
3. Füge die den Kanten in M entsprechenden kürzesten Pfade in G ein und bestimme im resultierenden Graphen einen Eulerkreis. Dieser ist eine Lösung.

Planare Graphen

Definition

Ein Graph heißt *planar, wenn man ihn so in die Ebene einbetten kann, dass sich keine Kanten kreuzen.*
Ein ebener Graph ist ein planarer Graph zusammen mit einer planaren Darstellung in der Ebene.

Beispiel

1. K_4 ist planar.
2. K_5 ist nicht planar.
3. $K_{2,n}$ ist planar.
4. $K_{3,3}$ ist nicht planar.

Definition

Ein Graph heißt *planar, wenn man ihn so in die Ebene einbetten kann, dass sich keine Kanten kreuzen.*
Ein ebener Graph ist ein planarer Graph zusammen mit einer planaren Darstellung in der Ebene.

Beispiel

1. K_4 ist planar.
2. K_5 ist nicht planar.
3. $K_{2,n}$ ist planar.
4. $K_{3,3}$ ist nicht planar.

Planare Graphen

Definition

Ein Graph heißt *planar, wenn man ihn so in die Ebene einbetten kann, dass sich keine Kanten kreuzen.*
Ein ebener Graph ist ein planarer Graph zusammen mit einer planaren Darstellung in der Ebene.

Beispiel

1. K_4 ist planar.
2. K_5 ist nicht planar.
3. $K_{2,n}$ ist planar.
4. $K_{3,3}$ ist nicht planar.

Planare Graphen

Definition

Ein Graph heißt *planar, wenn man ihn so in die Ebene einbetten kann, dass sich keine Kanten kreuzen.*
Ein ebener Graph ist ein planarer Graph zusammen mit einer planaren Darstellung in der Ebene.

Beispiel

1. K_4 ist planar.
2. K_5 ist nicht planar.
3. $K_{2,n}$ ist planar.
4. $K_{3,3}$ ist nicht planar.

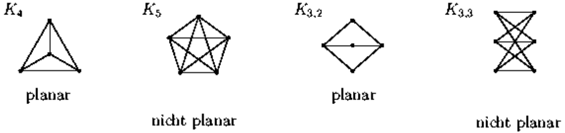
Planare Graphen

Definition

Ein Graph heißt *planar, wenn man ihn so in die Ebene einbetten kann, dass sich keine Kanten kreuzen.*
Ein ebener Graph ist ein planarer Graph zusammen mit einer planaren Darstellung in der Ebene.

Beispiel

1. K_4 ist planar.
2. K_5 ist nicht planar.
3. $K_{2,n}$ ist planar.
4. $K_{3,3}$ ist nicht planar.


K_4
planar
K_5
nicht planar
$K_{3,2}$
planar
$K_{3,3}$
nicht planar

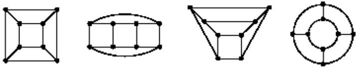
Bemerkung: **Satz von Kuratowski**

Ein Graph ist genau dann planar, wenn er weder eine Unterteilung des K_5 noch des $K_{3,3}$ als Teilgraph enthält.

Ein und derselbe Graph kann jedoch viele „verschiedene“ Einbettungen in die Ebene haben:

Bemerkung: **Satz von Kuratowski**
Ein Graph ist genau dann planar, wenn er weder eine Unterteilung des K_5 noch des $K_{3,3}$ als Teilgraph enthält.

Ein und derselbe Graph kann jedoch viele „verschiedene" Einbettungen in die Ebene haben:



Bei einer ebenen Einbettung eines Graphen bezeichnet man ein zusammenhängendes Gebiet des Komplements der Einbettung als *Facette*.

Es gilt:

Satz (Eulersche Polyederformel)

Sei $G = (V, E)$ ein zusammenhängender planarer Graph, f die Anzahl der Facetten einer ebenen Einbettung von G, $n = |V|$ und $e = |E|$. Dann ist

$$n - e + f = 2 \ .$$

Bei einer ebenen Einbettung eines Graphen bezeichnet man ein zusammenhängendes Gebiet des Komplements der Einbettung als *Facette*.

Es gilt:

Satz (Eulersche Polyederformel)

Sei $G = (V, E)$ *ein zusammenhängender planarer Graph,* f *die Anzahl der Facetten einer ebenen Einbettung von* G*,* $n = |V|$ *und* $e = |E|$*. Dann ist*

$$n - e + f = 2 \,.$$

Beweis.

Der Beweis erfolgt durch Induktion über die Anzahl der Kanten. Da G als zusammenhängend vorausgesetzt ist, gilt sicher $e \geq n - 1$. *Induktionsanfang:* Für $e = n - 1$ ist G ein Baum, enthält also keinen Kreis, es gibt daher genau eine Facette.
Induktionsschritt: Sei nun $e > n - 1$. Dann muss G einen Kreis C enthalten. Entfernen wir eine Kante dieses Kreises, dann verschmelzen in der Einbettung zwei Facetten zu einer (*Jordanscher Kurvensatz*), also verringern sich f und e jeweils um 1. Da gemäß Induktion nunmehr $n - e + f = 2$, gilt diese Relation auch für die Einbettung von G. □

Beweis.

Der Beweis erfolgt durch Induktion über die Anzahl der Kanten. Da G als zusammenhängend vorausgesetzt ist, gilt sicher $e \geq n - 1$. *Induktionsanfang:* Für $e = n - 1$ ist G ein Baum, enthält also keinen Kreis, es gibt daher genau eine Facette. *Induktionsschritt:* Sei nun $e > n - 1$. Dann muss G einen Kreis C enthalten. Entfernen wir eine Kante dieses Kreises, dann verschmelzen in der Einbettung zwei Facetten zu einer (*Jordanscher Kurvensatz*), also verringern sich f und e jeweils um 1. Da gemäß Induktion nunmehr $n - e + f = 2$, gilt diese Relation auch für die Einbettung von G. □

Beweis.

Der Beweis erfolgt durch Induktion über die Anzahl der Kanten. Da G als zusammenhängend vorausgesetzt ist, gilt sicher $e \geq n - 1$. *Induktionsanfang:* Für $e = n - 1$ ist G ein Baum, enthält also keinen Kreis, es gibt daher genau eine Facette. *Induktionsschritt:* Sei nun $e > n - 1$. Dann muss G einen Kreis C enthalten. Entfernen wir eine Kante dieses Kreises, dann verschmelzen in der Einbettung zwei Facetten zu einer (*Jordanscher Kurvensatz*), also verringern sich f und e jeweils um 1. Da gemäß Induktion nunmehr $n - e + f = 2$, gilt diese Relation auch für die Einbettung von G. $\square$

Beweis.

Der Beweis erfolgt durch Induktion über die Anzahl der Kanten. Da G als zusammenhängend vorausgesetzt ist, gilt sicher $e \geq n - 1$. *Induktionsanfang:* Für $e = n - 1$ ist G ein Baum, enthält also keinen Kreis, es gibt daher genau eine Facette. *Induktionsschritt:* Sei nun $e > n - 1$. Dann muss G einen Kreis C enthalten. Entfernen wir eine Kante dieses Kreises, dann verschmelzen in der Einbettung zwei Facetten zu einer (*Jordanscher Kurvensatz*), also verringern sich f und e jeweils um 1. Da gemäß Induktion nunmehr $n - e + f = 2$, gilt diese Relation auch für die Einbettung von G. □

Satz

Für jeden planaren Graphen $G = (V, E)$ mit $|V| \geq 3$ Knoten gilt

$$|E| \leq 3|V| - 6\ .$$

Beweis.

Betrachte eine beliebige Einbettung von G. Sei f die Anzahl der Facetten. Jede Facette wird von mindestens 3 Kanten berandet, jede Kante berandet höchstens zwei Facetten. Daraus folgt, dass $3f \leq 2|E|$. Mit der eulerschen Polyederformel ergibt sich

$$\frac{2}{3}|E| \geq f = |E| - |V| + 2\ ,$$

also

$$\frac{1}{3}|E| \leq |V| - 2\ .$$

□

Satz

Für jeden planaren Graphen $G = (V, E)$ mit $|V| \geq 3$ Knoten gilt

$$|E| \leq 3|V| - 6 \,.$$

Beweis.

Betrachte eine beliebige Einbettung von G. Sei f die Anzahl der Facetten. Jede Facette wird von mindestens 3 Kanten berandet, jede Kante berandet höchstens zwei Facetten. Daraus folgt, dass $3f \leq 2|E|$. Mit der eulerschen Polyederformel ergibt sich

$$\frac{2}{3}|E| \geq f = |E| - |V| + 2 \,,$$

also

$$\frac{1}{3}|E| \leq |V| - 2 \,.$$

$\square$

Satz

Für jeden planaren Graphen $G = (V, E)$ mit $|V| \geq 3$ Knoten gilt

$$|E| \leq 3|V| - 6\ .$$

Beweis.

Betrachte eine beliebige Einbettung von G. Sei f die Anzahl der Facetten. Jede Facette wird von mindestens 3 Kanten berandet, jede Kante berandet höchstens zwei Facetten. Daraus folgt, dass $3f \leq 2|E|$. Mit der eulerschen Polyederformel ergibt sich

$$\frac{2}{3}|E| \geq f = |E| - |V| + 2\ ,$$

also

$$\frac{1}{3}|E| \leq |V| - 2\ .$$

□

Satz

Für jeden planaren Graphen $G = (V, E)$ mit $|V| \geq 3$ Knoten gilt

$$|E| \leq 3|V| - 6 \,.$$

Beweis.

Betrachte eine beliebige Einbettung von G. Sei f die Anzahl der Facetten. Jede Facette wird von mindestens 3 Kanten berandet, jede Kante berandet höchstens zwei Facetten. Daraus folgt, dass $3f \leq 2|E|$. Mit der eulerschen Polyederformel ergibt sich

$$\frac{2}{3}|E| \geq f = |E| - |V| + 2 \,,$$

also

$$\frac{1}{3}|E| \leq |V| - 2 \,.$$

□

Korollar 20: Der K_5 ist nicht planar.

Beweis.

Für den K_5 gilt $e = \binom{5}{2} = 10$ und $n = 5$, aber

$$10 \not\leq 3 \cdot 5 - 6\ .$$

$\square$

Korollar 20: Der K_5 ist nicht planar.

Beweis.
Für den K_5 gilt $e = \binom{5}{2} = 10$ und $n = 5$, aber

$$10 \not\leq 3 \cdot 5 - 6\ .$$

$\square$