Graphalgorithmen

Johannes Köbler

Institut für Informatik
Humboldt-Universität zu Berlin

WS 2023/24

Einführung

- Viele praktisch relevante Problemstellungen lassen sich durch graphentheoretische Probleme modellieren, wie zum Beispiel
  - Finden kürzester Wege zwischen Städten
  - Berechnung von Flüssen und Schnitten in Netzwerken
  - Zuordungsprobleme (Berechnung von Matchings)
  - Färbungsprobleme (Knoten- und Kantenfärbungen)
  - planare Einbettungen von Schaltkreisen usw.
- In der Graphentheorie werden kombinatorische Eigenschaften von Graphen erforscht
- In diesem Modul steht der Entwurf von effizienten Algorithmen auf Graphen im Mittelpunkt

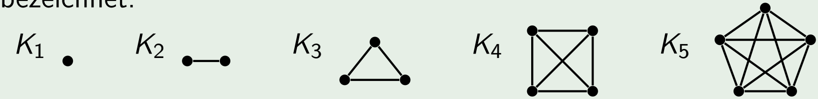
Graphentheoretische Grundlagen

Definition

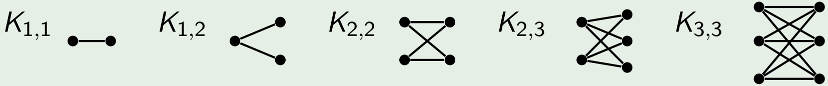
- Ein (ungerichteter) Graph ist ein Paar $G = (V, E)$, wobei
  V - eine endliche Menge von Knoten/Ecken und
  E - die Menge der Kanten ist
- Hierbei gilt $E \subseteq \binom{V}{2} := \{\{u, v\} \subseteq V : u \neq v\}$
- Die Nachbarschaft von $v \in V$ in G ist $N_G(v) = \{u \in V : \{u, v\} \in E\}$
- Der Grad von v in G ist $\deg_G(v) = |N_G(v)|$
- Der Minimalgrad von G ist $\delta(G) = \min_{v \in V} \deg_G(v)$ und der Maximalgrad von G ist $\Delta(G) = \max_{v \in V} \deg_G(v)$
- Im Fall $\delta(G) = \Delta(G) = k$ heißt G k-regulär
- Jeder Knoten $u \in V$ vom Grad ≤ 1 heißt Blatt und die übrigen Knoten (vom Grad ≥ 2) heißen innere Knoten von G
- Falls G aus dem Kontext ersichtlich ist, lassen wir den Index G weg und schreiben auch einfach $N(v)$, $\deg(v)$, δ usw.

Beispiel

- Der vollständige Graph auf n Knoten, d.h. $|V| = n$ und $E = \binom{V}{2}$, wird mit K_n und der leere Graph auf n Knoten, d.h. $E = \emptyset$, wird mit E_n bezeichnet:



- Der vollständige bipartite Graph auf $n = a + b$ Knoten, d.h. es gilt $V = A \cup B$ mit $A \cap B = \emptyset$, $|A| = a \geq 1$, $|B| = b \geq 1$ und $E = \{\{u, v\} : u \in A, v \in B\}$, wird mit $K_{a,b}$ bezeichnet:



Dieser wird oft auch in der Form $G = (A, B, E)$ notiert

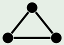
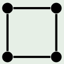
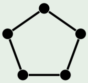
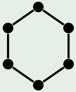
Beispiel (Fortsetzung)

- Der Pfad mit $n \geq 1$ Knoten wird mit P_n bezeichnet:

  P_2  P_3  P_4  P_5 

  Seine Länge ist $n - 1$

- Der Kreis mit $n \geq 3$ Knoten wird mit C_n bezeichnet:

  C_3  C_4  C_5  C_6 

◁

Definition Sei $G = (V, E)$ ein Graph.

- Eine Knotenmenge $U \subseteq V$ heißt unabhängig oder stabil, wenn es keine Kante von G mit beiden Endpunkten in U gibt, d.h. es gilt $E \cap \binom{U}{2} = \emptyset$
- Die Stabilitätszahl ist

$$\alpha(G) = \max\{|U| : U \text{ ist stabile Menge in } G\}$$

- Eine Knotenmenge $U \subseteq V$ heißt Clique, wenn jede Kante mit beiden Endpunkten in U in E ist, d.h. es gilt $\binom{U}{2} \subseteq E$
- Die Cliquenzahl ist

$$\omega(G) = \max\{|U| : U \text{ ist Clique in } G\}$$

Definition (Fortsetzung)

- Ein Graph $G' = (V', E')$ heißt Sub-/Teil-/Untergraph von $G = (V, E)$, falls $V' \subseteq V$ und $E' \subseteq E$ ist
- Im Fall $V' = V$ wird G' auch ein (auf)spannender Teilgraph von G genannt und wir schreiben für G' auch $G - E''$ (bzw. $G = G' \cup E''$), wobei $E'' = E - E'$ die Menge der aus G entfernten Kanten ist
- Im Fall $E'' = \{e\}$ schreiben wir für G' auch einfach $G - e$ (bzw. $G = G' \cup e$)
- Ein k-regulärer spannender Teilgraph von G wird auch als k-Faktor von G bezeichnet
- Ein d-regulärer Graph G heißt k-faktorisierbar, wenn sich G in $\ell = d/k$ kantendisjunkte k-Faktoren $G_1, \ldots, G_\ell$ zerlegen lässt

Definition (Fortsetzung)

- Ein Subgraph $G' = (V', E')$ von $G = (V, E)$ heißt (durch V') induziert, falls $E' = E \cap \binom{V'}{2}$ ist
- Für G' schreiben wir dann auch $G[V']$ oder $G - V''$, wobei $V'' = V - V'$ die Menge der aus G entfernten Knoten ist
- Ist $V'' = \{v\}$, so schreiben wir für G' auch einfach $G - v$ und im Fall $V' = \{v_1, \ldots, v_k\}$ auch $G[v_1, \ldots, v_k]$
- Ein Weg ist eine Folge von (nicht notwendig verschiedenen) Knoten $v_0, \ldots, v_\ell$ mit $\{v_i, v_{i+1}\} \in E$ für $i = 0, \ldots, \ell - 1$
- Die Länge des Weges ist die Anzahl der durchlaufenen Kanten, also ℓ
- Im Fall $\ell = 0$ heißt der Weg trivial
- Ein Weg $(v_0, \ldots, v_\ell)$ heißt auch v_0-v_ℓ-Weg

Definition (Fortsetzung)

- G heißt zusammenhängend, falls es für alle Paare $\{u, v\} \in \binom{V}{2}$ einen u-v-Weg gibt
- Die durch die Äquivalenzklassen $V_i \subseteq V$ der Relation

$$Z = \{(u, v) \in V \times V : \text{ es gibt in } G \text{ einen } u\text{-}v\text{-Weg}\}$$

  induzierten Teilgraphen $G[V_i]$ heißen Zusammenhangskomponenten (engl. connected components) oder einfach Komponenten von G
- Ein u-v-Weg heißt einfach oder u-v-Pfad, falls alle durchlaufenen Knoten verschieden sind
- Ein Zyklus ist ein u-v-Weg mit $u = v$

Definition (Schluss)

- Eine Menge von Pfaden heißt
  - disjunkt, wenn je zwei Pfade in der Menge keine gemeinsamen Knoten haben,
  - kantendisjunkt, wenn je zwei Pfade in der Menge keine gemeinsamen Kanten haben, und
  - knotendisjunkt, wenn je zwei Pfade in der Menge höchstens gemeinsame Endpunkte haben
- Ein Kreis ist ein Zyklus $(v_1 \ldots, v_\ell, v_1)$ der Länge $\ell \geq 3$, für den $v_1, \ldots, v_\ell$ paarweise verschieden sind
- Ein Graph heißt kreisfrei, azyklisch oder Wald, falls er keinen Kreis enthält
- Ein Baum ist ein zusammenhängender Wald

Graphentheoretische Grundlagen

Definition

- Ein gerichteter Graph oder Digraph ist ein Paar $G = (V, E)$, wobei
  - V - eine endliche Menge von Knoten/Ecken und
  - E - die Menge der Kanten ist
- Hierbei gilt $E \subseteq V \times V = \{(u, v) : u, v \in V\}$
- Kanten der Form (u, u) heißen Schlingen
- Sei $v \in V$ ein Knoten von G
  - Die Nachfolgermenge von v ist $N^+(v) = \{u \in V : (v, u) \in E\}$
  - Die Vorgängermenge von v ist $N^-(v) = \{u \in V : (u, v) \in E\}$
  - Die Nachbarmenge von v ist $N(v) = N^+(v) \cup N^-(v)$
  - Der Ausgangsgrad von v ist $\deg^+(v) = |N^+(v)|$ und der Eingangsgrad von v ist $\deg^-(v) = |N^-(v)|$
  - Der Grad von v ist $\deg(v) = \deg^+(v) + \deg^-(v)$
  - Ist $\deg^-(v) = 0$, so heißt v Wurzel von G, und v heißt Blatt, falls $\deg^+(u) = 0$ ist

Graphentheoretische Grundlagen

Definition (Fortsetzung)

- Ein (gerichteter) $\boldsymbol{v_0}$-$\boldsymbol{v_\ell}$-Weg in G ist eine Folge von Knoten $v_0, \ldots, v_\ell$ mit $(v_i, v_{i+1}) \in E$ für $i = 0, \ldots, \ell - 1$
- Ein (gerichteter) Zyklus ist ein gerichteter u-v-Weg mit $u = v$
- Ein gerichteter Weg heißt einfach oder (gerichteter) Pfad, falls alle durchlaufenen Knoten verschieden sind
- Ein (gerichteter) Kreis in G ist ein gerichteter Zyklus $(v_1 \ldots, v_\ell, v_1)$ der Länge $\ell \geq 1$, für den $v_1, \ldots, v_\ell$ paarweise verschieden sind
- G heißt kreisfrei oder azyklisch, wenn G keinen gerichteten Kreis hat
- G heißt gerichteter Wald, wenn G kreisfrei ist und jeder Knoten $v \in V$ Eingangsgrad $\deg^-(v) \leq 1$ hat
- G heißt zusammenhängend, wenn es in G für jedes Knotenpaar $u \neq v \in V$ einen u-v-Pfad oder einen v-u-Pfad gibt
- G heißt stark zusammenhängend, wenn es in G für jedes Knotenpaar $u \neq v \in V$ sowohl einen u-v-Pfad als auch einen v-u-Pfad gibt

Definition

- Die Adjazenzmatrix eines Graphen bzw. Digraphen $G = (V, E)$ mit (geordneter) Knotenmenge $V = \{v_1, \ldots, v_n\}$ ist die $(n \times n)$-Matrix $A = (a_{ij})$ mit den Einträgen

$$a_{ij} = \begin{cases} 1, & \{v_i, v_j\} \in E \\ 0, & \text{sonst} \end{cases} \qquad \text{bzw.} \qquad a_{ij} = \begin{cases} 1, & (v_i, v_j) \in E \\ 0, & \text{sonst} \end{cases}$$

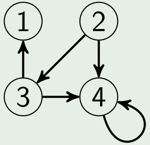
- Für ungerichtete Graphen ist die Adjazenzmatrix symmetrisch mit $a_{ii} = 0$ für $i = 1, \ldots, n$

Definition (Fortsetzung)

- Bei der Adjazenzlisten-Darstellung wird für jeden Knoten v_i eine (einfach oder doppelt verkettete) Liste mit seinen Nachbarn verwaltet
- Im gerichteten Fall verwaltet man entweder nur die Liste der Nachfolger oder zusätzlich eine weitere für die Vorgänger
- Falls die Anzahl der Knoten statisch ist, organisiert man die Adjazenzlisten in einem Feld, d.h. das Feldelement mit Index i verweist auf die Adjazenzliste von Knoten v_i
- Falls sich die Anzahl der Knoten dynamisch ändert, so werden die Adjazenzlisten typischerweise ebenfalls in einer Liste verwaltet

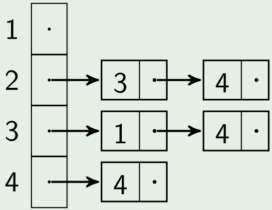
Beispiel

- Betrachte den gerichteten Graphen $G = (V, E)$ mit $V = \{1, 2, 3, 4\}$ und $E = \{(2, 3), (2, 4), (3, 1), (3, 4), (4, 4)\}$



- Dieser hat folgende Adjazenzmatrix- und Adjazenzlisten-Darstellung:

	1	2	3	4
1	0	0	0	0
2	0	0	1	1
3	1	0	0	1
4	0	0	0	1



Färben von Graphen

Definition Sei $G = (V, E)$ ein Graph.

- Eine Abbildung $f\colon V \to \{1, \dots, k\}$ heißt *k-Färbung* (oder einfach *Färbung*) von G, wenn $f(u) \neq f(v)$ für alle $\{u, v\} \in E$ gilt
- In diesem Fall heißt G *k-färbbar*
- Die *chromatische Zahl von G* ist

$$\chi(G) = \min\{k \in \mathbb{N} : G \text{ ist } k\text{-färbbar}\}$$

Beispiel

- $\chi(E_n) = 1$
- $\chi(K_{n,m}) = 2$
- $\chi(K_n) = n$
- $\chi(C_n) = \begin{cases} 2, & n \text{ gerade} \\ 3, & \text{sonst} \end{cases}$

Die Frage, ob ein gegebener Graph k-färbbar ist, ist für jedes feste $k \geq 3$ schwierig

k-Färbbarkeit (k-COLORING):

Gegeben: Ein Graph G

Gefragt: Ist G k-färbbar?

Satz

Das Entscheidungsproblem k-COLORING ist für $k \geq 3$ NP-vollständig

Färben von Graphen

Lemma

$n/\alpha(G) \leq \chi(G) \leq n - \alpha(G) + 1$

Beweis.

- Sei G ein Graph und sei c eine $\chi(G)$-Färbung von G
- Dann sind die Mengen $S_i = \{u \in V : c(u) = i\}$, $i = 1, \ldots, \chi(G)$, stabil
- Wegen $|S_i| \leq \alpha(G)$ folgt also

$$n \;=\; \sum_{i=1}^{\chi(G)} |S_i| \;\leq\; \chi(G)\alpha(G)$$

- Für den Beweis von $\chi(G) \leq n - \alpha(G) + 1$ sei S eine stabile Menge in G mit $|S| = \alpha(G)$
- Dann ist $G - S$ k-färbbar für ein $k \leq n - |S|$
- Da wir alle Knoten in S mit der Farbe $k + 1$ färben können, folgt $\chi(G) \leq k + 1 \;\leq n - \alpha(G) + 1$ □

Färben von Graphen

Lemma

$\binom{\chi(G)}{2} \leq m$ und somit $\chi(G) \leq {}^1\!/_2 + \sqrt{2m + {}^1\!/_4}$

Beweis.

Zwischen je zwei Farbklassen einer optimalen Färbung muss es mindestens eine Kante geben □

Die chromatische Zahl steht auch in Beziehung zur Cliquenzahl $\omega(G)$ und zum Maximalgrad $\Delta(G)$

Lemma

$\omega(G) \leq \chi(G) \leq \Delta(G) + 1$

Färben von Graphen

Lemma

$\omega(G) \leq \chi(G) \leq \Delta(G) + 1$

Beweis.

- Die erste Ungleichung folgt, da die Knoten einer größten Clique unterschiedliche Farben erhalten müssen
- Um die zweite Ungleichung zu erhalten, betrachten wir den

Algorithmus `greedy-color`

1 **input** ein Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$
2 $c(v_1) := 1$
3 **for** $i := 2$ **to** n **do**
4 $\quad F_i := \{c(v_j) : j < i, v_j \in N(v_i)\}$
5 $\quad c(v_i) := \min\{k \geq 1 : k \notin F_i\}$

- Da in Zeile 5 für die Farbe $c(v_i)$ von v_i nur $|F_i| \leq \Delta(G)$ Farben verboten sind, gilt $c(v_i) \leq \Delta(G) + 1$ $\square$

Färben von planaren Graphen

- Ein Graph G heißt planar, wenn er so in die Ebene einbettbar ist, dass sich zwei verschiedene Kanten höchstens in ihren Endpunkten berühren
- Dabei werden die Knoten von G als Punkte und die Kanten von G als Verbindungslinien (genauer: Jordankurven) zwischen den zugehörigen Endpunkten dargestellt
- Bereits im 19. Jahrhundert wurde die Frage aufgeworfen, wie viele Farben höchstens benötigt werden, um eine Landkarte so zu färben, dass aneinander grenzende Länder unterschiedliche Farben erhalten
- Offensichtlich lässt sich eine Landkarte in einen planaren Graphen transformieren, indem man für jedes Land einen Knoten zeichnet und benachbarte Länder durch eine Kante verbindet
- Länder, die sich nur in einem Punkt berühren, gelten dabei nicht als benachbart

Färben von planaren Graphen

- Die Vermutung, dass 4 Farben ausreichen, wurde 1878 von Kempe „bewiesen“ und erst 1890 entdeckte Heawood einen Fehler in Kempes „Beweis“
- Übrig blieb der 5-Farben-Satz
- Der 4-Farben-Satz wurde erst 1976 von Appel und Haken bewiesen
- Hierbei handelt es sich jedoch nicht um einen Beweis im klassischen Sinne, da zur Überprüfung der vielen auftretenden Spezialfälle Computer benötigt werden

Satz (Appel, Haken 1976)

Jeder planare Graph ist 4-färbbar

Färben von planaren Graphen

Satz (Appel, Haken 1976)

Jeder planare Graph ist 4-färbbar

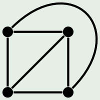
- Aus dem Beweis des 4-Farben-Satzes von Appel und Haken lässt sich ein 4-Färbungsalgorithmus für planare Graphen mit einer Laufzeit von $\mathcal{O}(n^4)$ gewinnen
- Im Jahr 1997 fanden Robertson, Sanders, Seymour und Thomas einen einfacheren Beweis für den 4-Farben-Satz, welcher zwar einen deutlich schnelleren $\mathcal{O}(n^2)$ Algorithmus liefert, aber ebenfalls nur mit Computer-Unterstützung verifizierbar ist

Färben von planaren Graphen

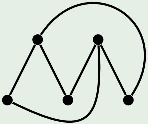
Beispiel

Wie die folgenden Einbettungen von K_4 und $K_{2,3}$ in die Ebene zeigen, sind diese Graphen planar

K_4:



$K_{2,3}$:



◁

- Zur Beantwortung der Frage, ob auch K_5 und $K_{3,3}$ planar sind, betrachten wir die Gebiete, die bei der Einbettung von (zusammenhängenden) Graphen in die Ebene entstehen
- Dabei gehören 2 Punkte zum selben Gebiet, falls es zwischen ihnen eine Verbindungslinie gibt, die keine Kante des eingebetten Graphen kreuzt oder berührt
- Nur eines dieser Gebiete ist unbeschränkt und dieses wird als äußeres Gebiet bezeichnet

Färben von planaren Graphen

- Die Anzahl der Gebiete von G bezeichnen wir mit $r(G)$ oder kurz mit r
- Die begrenzenden Kanten eines Gebietes g bilden den Rand von g
- Die Anzahl dieser Kanten bezeichnen wir mit $d(g)$, wobei Kanten, an die g von beiden Seiten grenzt, doppelt gezählt werden
- Der Rand $rand(g)$ eines Gebiets g ist die (zirkuläre) Folge aller an g grenzenden Kanten, wobei die Kanten auf dem Rand von g so durchlaufen werden, dass g „in Fahrtrichtung links“ liegt
- Dies hat zur Folge, dass jeder Knoten u, der über eine Kante e erreicht wird, über die im Uhrzeigersinn nächste Kante e' wieder verlassen wird
- Auf diese Weise erhalten wir für jeden Knoten u eine (zirkuläre) Ordnung π_u aller mit u inzidenten Kanten
- Wir nennen das Tripel $G' = (V, E, R)$ eine ebene Realisierung des Graphen $G = (V, E)$, falls es eine Einbettung von G in die Ebene gibt, deren Gebiete die Ränder in R haben
- In diesem Fall nennen wir $G' = (V, E, R)$ auch einen ebenen Graphen

Färben von planaren Graphen

- Führen zwei Einbettungen von G in die Ebene auf dieselbe Randmenge R, so werden sie als äquivalent angesehen
- Eine andere Möglichkeit, Einbettungen bis auf Äquivalenz kombinatorisch zu beschreiben, besteht darin, für jeden Knoten u die (zirkuläre) Ordnung π_u aller mit u inzidenten Kanten anzugeben
- Man nennt $\pi = \{\pi_u : u \in V\}$ ein Rotationssystem für G, falls es eine entsprechende Einbettung gibt
- Rotationssysteme haben den Vorteil, dass sie in Adjazenzlistendarstellung ohne zusätzlichen Platzaufwand gespeichert werden können, indem man die zu u adjazenten Knoten gemäß π_u anordnet
- Ist G nicht zusammenhängend, so betten wir die Komponenten von G in die Ebene ein und fassen alle Ränder, die bei diesen Einbettungen entstehen, zu einer Randmenge R zusammen
- Da jede Kante zur Gesamtlänge $\sum_g d(g)$ aller Ränder den Wert 2 beiträgt (sie wird genau einmal in jeder Richtung durchlaufen), folgt

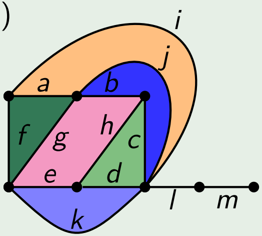
$$\sum_g d(g) = 2m(G)$$

Färben von planaren Graphen

Beispiel

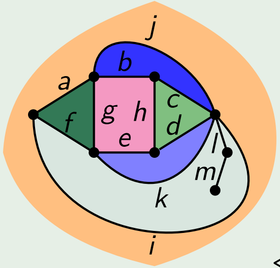
- Die nebenstehenden Einbettungen von $G = (V, E)$ in die Ebene haben 7 Gebiete mit den Rändern

$$R = \{(a, f, g), (a, j, i), (b, g, e, h), (b, c, j), (c, h, d), (d, e, k), (f, i, l, m, m, l, k)\}$$



- Der zugehörige ebene Graph ist $G' = (V, E, R)$ und das zugehörige Rotationssystem ist

$$\pi = \{(a, f, i), (a, j, b, g), (b, c, h), (e, k, f, g), (d, e, h), (c, j, i, l, k, d), (l, m), (m)\}$$



- Man beachte, dass sowohl in R als auch in π jede Kante genau zweimal vorkommt
- Anstelle von (zirkulären) Kantenfolgen kann man die Elemente von R und π natürlich auch durch entsprechende Knotenfolgen beschreiben

◁

Färben von planaren Graphen

Satz (Polyederformel von Euler, 1750)

Für einen zusammenhängenden ebenen Graphen $G = (V, E, R)$ gilt

$$n(G) - m(G) + r(G) = 2 \qquad (*)$$

Beweis durch Induktion über die Kantenzahl $m(G) = m$

$\boldsymbol{m = 0}$**:** Da G zusammenhängend ist, muss in diesem Fall $n = 1$ sein

- Somit ist auch $r = 1$, also $(*)$ erfüllt

$\boldsymbol{m - 1 \rightsquigarrow m}$**:** Sei G ein zusammenhängender ebener Graph mit m Kanten

- Ist G ein Baum, so entfernen wir ein Blatt und erhalten einen zusammenhängenden ebenen Graphen G' mit $n' = n - 1$ Knoten, $m' = m - 1$ Kanten und $r' = r$ Gebieten
- Nach IV folgt $n - m + r \ = (n - 1) - (m - 1) + r \ = n' - m' + r' \ = 2$
- Falls G kein Baum ist, entfernen wir eine Kante auf einem Kreis in G und erhalten einen zusammenhängenden ebenen Graphen G' mit $n' = n$ Knoten, $m' = m - 1$ Kanten und $r' = r - 1$ Gebieten
- Nach IV folgt $n - m + r \ = n - (m - 1) + (r - 1) \ = n' - m' + r' \ = 2$ $\square$

Färben von planaren Graphen

Korollar Sei $G = (V, E)$ ein planarer Graph mit $n \geq 3$ Knoten

Dann ist $m \leq 3n - 6$. Falls G dreiecksfrei ist, gilt sogar $m \leq 2n - 4$.

Beweis.

- O.B.d.A. sei G zusammenhängend
- Wir betrachten eine beliebige planare Einbettung von G
- Da $n \geq 3$ ist, ist jedes Gebiet g von $d(g) \geq 3$ Kanten umgeben
- Daher ist $2m = i = \sum_g d(g) \;\geq 3r$ bzw. $r \leq 2m/3$
- Eulers Formel liefert

  $$m = n + r - 2 \;\leq n + 2m/3 - 2,$$

  was $(1 - {}^2\!/_3)m \leq n - 2$ und somit $m \leq 3n - 6$ impliziert
- Wenn G dreiecksfrei ist, ist jedes Gebiet von $d(g) \geq 4$ Kanten umgeben
- Daher ist $2m = i = \sum_g d(g) \;\geq 4r$ bzw. $r \leq m/2$
- Eulers Formel liefert daher $m = n + r - 2 \;\leq n + m/2 - 2$,
  was $m/2 \leq n - 2$ und somit $m \leq 2n - 4$ impliziert

$\square$

Färben von planaren Graphen

Korollar

Die Graphen K_5 und $K_{3,3}$ sind nicht planar

Beweis.

- Wegen $n(K_5) = 5$ und $m(K_5) = \binom{5}{2} = 10$ gilt

$$m(K_5) = 10 \not\leq 9 \;= 3n(K_5) - 6$$

- Wegen $n(K_{3,3}) = 6$ und $m(K_{3,3}) = 3 \cdot 3 = 9$ gilt

$$m(K_{3,3}) = 9 \not\leq 8 \;= 2n(K_{3,3}) - 4$$

□

Färben von planaren Graphen

Als weitere interessante Folgerung aus der Polyederformel können wir zeigen, dass jeder planare Graph einen Knoten v vom Grad $\deg(v) \leq 5$ hat

Korollar

Jeder planare Graph hat einen Minimalgrad $\delta \leq 5$

Beweis.

Die Annahme $\delta \geq 6$ impliziert $n \geq 3$ und führt auf die Ungleichung

$$m = {}^1\!/_2 \sum_{u \in V} \deg(u) \ \geq {}^1\!/_2 \sum_{u \in V} 6 \ = 3n,$$

was im Widerspruch zu $m \leq 3n - 6$ steht $\square$

Definition Sei $G = (V, E)$ ein Graph und seien $u, v \in V$ mit $u \neq v$

- Durch Fusion von u und v entsteht aus G der Graph $G_{uv} = (V - \{v\}, E')$ mit
$$E' = \{e \in E : v \notin e\} \cup \{\{u, v'\} \mid v' \in N_G(v) \setminus \{u\}\}$$
- Ist $\{u, v\}$ eine Kante von G (also $\{u, v\} \in E$), so sagen wir auch, G_{uv} entsteht aus G durch Kontraktion der Kante $\{u, v\}$
- Hat zudem v den Grad 2 mit $N_G(v) = \{u, w\} \notin E$, so sagen wir auch, G_{uv} entsteht aus G durch Überbrückung des Knotens v bzw. G aus G_{uv} durch Unterteilung der Kante $\{u, w\}$

Definition (Fortsetzung)

- G heißt zu H kontrahierbar, falls der Graph H aus einer isomorphen Kopie von G durch wiederholte Kontraktionen gewonnen werden kann
- In diesem Fall nennen wir den Graphen H auch eine Kontraktion von G bzw. den Graphen G eine Expansion von H
- H heißt zu G unterteilbar, falls G aus einer isomorphen Kopie von H durch wiederholte Unterteilungen von Kanten gewonnen werden kann
- In diesem Fall nennen wir G auch eine Unterteilung von H bzw. H eine Überbrückung von G

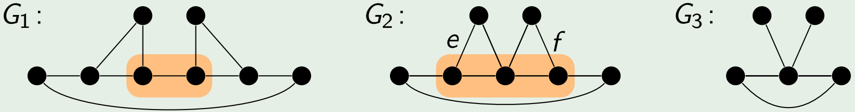
Definition (Schluss)

- H heißt Minor von G, wenn ein Teilgraph von G zu H kontrahierbar ist
- H heißt topologischer Minor von G, wenn ein Teilgraph G' von G eine Unterteilung von H ist (bzw. H eine Überbrückung von G' ist)
- G heißt H-frei, falls H kein Minor von G ist
- Für eine Menge $\mathcal{H}$ von Graphen heißt G $\mathcal{H}$-frei, falls kein $H \in \mathcal{H}$ ein Minor von G ist

Färben von planaren Graphen

Beispiel

Betrachte folgende Graphen:



- G_2 ist ein Minor von G_1, da G_2 durch Kontraktion der in G_1 markierten Kante entsteht; entsprechend ist G_3 ein Minor von G_2 und auch von G_1
- G_3 ist sogar ein topologischer Minor von G_2
- Entfernen wir nämlich aus G_2 die beiden Kanten e und f, so ist der resultierende Teilgraph G_2' eine Unterteilung von G_3
- G_2 selbst ist aber keine Unterteilung von G_3, da G_2 im Gegensatz zu G_3 Knoten vom Grad 3 hat
- Zudem sind G_2 und G_3 keine topologischen Minoren von G_1, da alle Knoten von G_1 einen Grad ≤ 3 haben, aber G_2 und G_3 einen Knoten vom Grad 4 haben ◁

Färben von planaren Graphen

- Es ist klar, dass die Klasse $\mathcal{K}$ der planaren Graphen zwar unter Subgraphbildung, Kontraktion, Unterteilung und Überbrückung abgeschlossen ist, aber nicht unter Fusion
- Folglich ist jeder (topologische) Minor und jede Unterteilung eines planaren Graphen ebenfalls planar
- Nach Definition lässt sich jeder (topologische) Minor H von G aus einem zu G isomorphen Graphen durch wiederholte Anwendung folgender Operationen gewinnen:
  - Entfernen einer Kante oder eines Knotens
  - Kontraktion einer Kante (bzw. Überbrückung eines Knotens)
- Da die Kontraktionen (bzw. Überbrückungen) o.B.d.A. auch zuletzt ausgeführt werden können, gilt hiervon auch die Umkehrung
- Zudem ist leicht zu sehen, dass zwei Graphen G und H genau dann (topologische) Minoren voneinander sind, wenn sie isomorph sind

Färben von planaren Graphen

Satz (Kempe 1878, Heawood 1890)

Jeder planare Graph G ist 5-färbbar

Beweis durch Induktion über die Knotenzahl n von G

$\boldsymbol{n = 1}$**:** Klar

$\boldsymbol{n-1 \rightsquigarrow n}$**:** Sei G ein planarer Graph mit $n(G) = n$ Knoten

- Da G planar ist, existiert ein Knoten u mit $\deg(u) \leq 5$
- Nun konstruieren wir zu G wie folgt einen Minor G':
  - Im Fall $\deg(u) \leq 4$ sei $G' = G - u$, d.h. wir entfernen u aus G
  - Andernfalls hat u zwei Nachbarn v und w, die nicht durch eine Kante verbunden sind (sonst wäre K_5 ein Teilgraph von G)
  - In diesem Fall sei $G' = (G_{vu})_{vw}$, d.h. wir kontrahieren die beiden Kanten $\{u, v\}$ und $\{u, w\}$ zum Knoten v
- Da G' ein Minor von G ist, ist G' planar

Färben von planaren Graphen

Satz (Kempe 1878, Heawood 1890)

Jeder planare Graph G ist 5-färbbar

Beweis (Fortsetzung)

- Da G' zudem höchstens $n - 1$ Knoten hat, hat G' nach IV eine 5-Färbung c'
- Wir erweitern c' wie folgt zu einer 5-Färbung c von G
  - Im 2. Fall geben wir dem Knoten w die Farbe $c(w) = c'(v)$
  - Da (in beiden Fällen) die Nachbarn von u in G höchstens vier verschiedene Farben haben, können wir nun auch u eine Farbe $c(u) \leq 5$ geben $\square$

Färben von planaren Graphen

- Kuratowski konnte 1930 beweisen, dass jeder nichtplanare Graph G den $K_{3,3}$ oder den K_5 als topologischen Minor enthält
- Für den Beweis benötigen wir folgende Definitionen und ein Lemma

Definition Sei $G = (V, E)$ ein Graph.

- Eine Menge $S \subseteq V$ heißt Separator in G, wenn es zwei Knoten $u, v \in V \setminus S$ gibt, zwischen denen in $G - S$ kein u-v-Weg existiert
- Ist $|S| = k$, so nennen wir S auch einen k-Separator zwischen u und v oder auch einen u-v-Separator der Größe k
- Für $0 \le k < n(G)$ heißt G k-zusammenhängend, falls G keinen $(k-1)$-Separator hat
- Die größte Zahl $k < n(G)$, für die G k-zusammenhängend ist, heißt die Zusammenhangszahl von G und wird mit $\kappa(G)$ bezeichnet
- Ein Knoten $s \in V$ heißt Schnittknoten oder Artikulation von G, wenn es zwei Knoten $u, v \in V \setminus \{s\}$ gibt, so dass zwar $\{s\}$ ein u-v-Separator, aber $\emptyset$ kein u-v-Separator ist

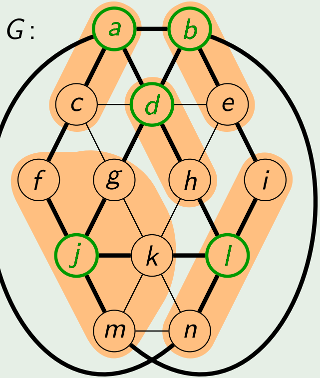
Beispiel Sei G ein Graph mit n Knoten

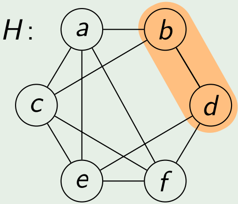
- $\kappa(G) = n - 1$ gilt genau dann, wenn $G = K_n$ ist
- $\kappa(G) \geq 1$ gilt genau dann, wenn G zusammenhängend und $n \geq 2$ ist
- $\kappa(G) = 1$ gilt genau dann, wenn G zusammenhängend ist und G mindestens einen Schnittknoten hat oder $G = K_2$ ist
- $\kappa(G) \geq 2$ gilt genau dann, wenn je 2 Knoten von G auf einem gemeinsamen Kreis liegen (siehe Übungen)

Färben von planaren Graphen

Beispiel

- Der Graph G ist nicht planar, da er den K_5 als Minor enthält
- K_5 ist sogar ein topologischer Minor von G, da wir die dünnen Kanten entfernen und in dem resultierenden Teilgraphen alle Knoten vom Grad 2 überbrücken können





- Der Graph H enthält den K_5 zwar immer noch als Minor
- Aber nicht als topologischen Minor

◁

Färben von planaren Graphen

Lemma

Jeder Graph $G = (V, E)$, der nicht planar ist, hat einen

- 2-zusammenhängenden Untergraphen $U = (V', E')$ und einen
- 3-zusammenhängenden topologischen Minor $M = (V'', E'')$,

die minimal nicht planar sind, d.h. U und M sind nicht planar und für alle $e' \in E'$ und $e'' \in E''$ sind die Graphen $U - e'$ und $M - e''$ planar

Beweis.

- Wir entfernen zuerst solange Kanten und Knoten aus G, bis wir aus dem verbliebenen Teilgraphen $U = (V', E')$ keine weiteren Kanten oder Knoten entfernen können, ohne dass U planar wird
- U ist zusammenhängend, da andernfalls mindestens eine Komponente von U nicht planar ist und wir alle übrigen Komponenten entfernen könnten, ohne dass U planar wird
- U ist sogar 2-zusammenhängend

Färben von planaren Graphen

Beweis (Fortsetzung)

- Wäre U nicht 2-zusammenhängend, würde U einen Schnittknoten s enthalten und $U - s$ in $k \geq 2$ Komponenten $U_i = U[V_i]$, $i = 1 \ldots, k$, zerfallen
- Dann wäre aber mindestens ein Teilgraph $U'_i = U[V_i \cup \{s\}]$ nicht planar und wir könnten alle Knoten in $V - V_i$ entfernen, ohne dass U planar wird
- Falls U 3-zusammenhängend ist, können wir $M = U$ setzen
- Andernfalls können wir U durch einen topologischen Minor $M(U)$ von U ersetzen, der wie U minimal nicht planar ist, aber weniger Knoten als U hat (siehe nächste Folie)
- Dies wiederholen wir solange bis wir einen 3-zusammenhängenden Minor M von U erhalten
- Da alle Graphen mit ≤ 4 Knoten planar sind, finden wir M nach spätestens $n(U) - 5$ Ersetzungsschritten

Färben von planaren Graphen

Beweis (Schluss)

- Falls U nicht 3-zusammenhängend ist, hat U einen Separator $S = \{u, v\}$, d.h. $U - S$ zerfällt in $k \geq 2$ Komponenten $U_i = U[V_i]$, $i = 1 \ldots, k$
- Dann sind die Graphen

  $$U_i' = U[V_i \cup \{u, v\}] \cup \{u, v\}, i = 1, \ldots, k$$

  wie U 2-zusammenhängend, da je 2 Knoten auf einem Kreis liegen
- Mindestens ein U_i' ist nicht planar (z.B. U_1'), da sonst U planar wäre
- Nun erhalten wir den Graphen $M(U) = U_1'$ wie folgt als topologischen Minor von $U_1 \cup U_2 = U[V_1 \cup V_2 \cup \{u, v\}]$ (und damit von U):
  - Wähle in $U[V_2 \cup \{u, v\}]$ einen u-v-Pfad P und entferne aus $U_1 \cup U_2$ alle Knoten und Kanten von $U[V_2 \cup \{u, v\}]$, die nicht auf P liegen
  - Überbrücke P zur Kante $\{u, v\}$
- Dann hat $M(U)$ weniger Knoten als U
- Zudem ist $M(U)$ wie U minimal nicht planar $\square$

Definition Sei G ein Graph und sei K ein Kreis in G

- Ein Teilgraph B von G heißt Brücke von K in G, falls
  - B nur aus einer Kante besteht, die zwei Knoten von K verbindet und nicht auf K liegt, oder
  - $B - K$ eine Komponente von $G - K$ ist und B aus $B - K$ durch Hinzufügen aller Kanten zwischen $B - K$ und K (und der zugehörigen Endpunkte auf K) entsteht
- Brücken, die nur aus einer Kante bestehen, werden auch als Sehnen von K bezeichnet

Definition (Fortsetzung)

- Die Knoten von B, die auf K liegen, heißen Kontaktpunkte von B
- Zwei Brücken B und B' von K heißen inkompatibel, falls
  - B Kontaktpunkte u, v und B' Kontaktpunkte u', v' hat, so dass diese vier Punkte in der Reihenfolge u, u', v, v' auf K liegen, oder
  - B und B' mindestens 3 gemeinsame Kontaktpunkte haben

Es ist leicht zu sehen, dass in einem planaren Graphen kein Kreis mehr als zwei inkompatible Brücken haben kann

Färben von planaren Graphen

Satz (Kuratowski 1930)

Für einen Graphen G sind folgende Aussagen äquivalent:

(i) G ist planar

(ii) G enthält weder den $K_{3,3}$ noch den K_5 als topologischen Minor

Beweis.

- Die Implikation von (i) nach (ii) folgt aus der Abgeschlossenheit der planaren Graphen unter (topologischer) Minorenbildung
- Die Implikation von $ii)$ nach $i)$ zeigen wir durch Kontraposition
- Sei also G nicht planar
- Dann hat G nach obigem Lemma einen 3-zusammenhängenden nicht planaren topologischen Minor $M = (V, E)$, so dass $M - e$ für jede Kante $e \in E$ planar ist
- Wir entfernen eine beliebige Kante $e = \{u, v\}$ aus M
- Dann ist $M - e$ planar

Färben von planaren Graphen

Beweis (Fortsetzung)

- Da $M - e$ 2-zusammenhängend ist, gibt es in $M - e$ einen Kreis K durch die beiden Knoten u und v (siehe Übungen)
- Wir wählen K zusammen mit einer ebenen Realisierung H von $M - e$ so, dass K möglichst viele Gebiete in H einschließt
- Für zwei Knoten a, b auf K bezeichnen wir mit $K[a, b]$ die Menge aller Knoten, die in H auf dem Bogen von a nach b (im Uhrzeigersinn) auf K liegen
- Zudem sei $K[a, b) = K[a, b] \setminus \{b\}$; die Mengen $K(a, b)$ und $K(a, b]$ sind analog definiert
- Die Kanten jeder Brücke B von K in $M - e$ verlaufen in H entweder alle innerhalb oder alle außerhalb von K
- Im ersten Fall nennen wir B eine innere Brücke und im zweiten eine äußere Brücke

Beweis (Fortsetzung)

- Es ist klar, dass K in H mindestens eine innere und mindestens eine äußere Brücke haben muss (sonst könnten wir e zu H hinzufügen)
- Zudem hat jede äußere Brücke B genau zwei Kontaktpunkte: einen Knoten auf dem Bogen $K(u, v)$ und einen Knoten auf $K(v, u)$
- Andernfalls hätte B nämlich mindestens 2 Kontaktpunkte auf $K[u, v]$ oder auf $K[v, u]$ und wir könnten K zu einem Kreis K' erweitern, der in H mehr Gebiete einschließt (bzw. ausschließt) als K, was der Wahl von K und H widerspricht
- Da M 3-zusammenhängend ist, muss B zudem eine Sehne sein
- K hat in M außer den Brücken in $M - e$ noch zusätzlich die Brücke e
- Wir wählen nun eine innere Brücke B, die sowohl zu e als auch zu mindestens einer äußeren Brücke $e' = \{x, y\}$ inkompatibel ist
- Eine solche Brücke muss es geben, da

Färben von planaren Graphen

Beweis (Fortsetzung)

- Eine solche Brücke muss es geben, da wir sonst alle mit e inkompatiblen inneren Brücken nach außen klappen und e als innere Brücke hinzunehmen könnten, ohne die Planarität zu verletzen
- Da $e = \{u, v\}$ und $e' = \{x, y\}$ inkompatibel sind, können wir annehmen, dass die vier Endpunkte von e und e' in der Reihenfolge u, x, v, y auf K liegen
- Wir benutzen nun die drei Brücken e, e' und B von K, um eine Unterteilung des K_5 oder des $K_{3,3}$ in M zu finden, indem wir
  - entweder eine Menge $A \subseteq V$ mit fünf Knoten angeben, die paarweise durch zehn knotendisjunkte Pfade verbunden sind,
  - oder zwei disjunkte Mengen $A_1, A_2 \subseteq V$ mit jeweils drei Knoten angeben, so dass alle Knoten in A_1 mit allen Knoten in A_2 durch insgesamt neun knotendisjunkte Pfade verbunden sind

Beweis (Fortsetzung)

Fall 1: B hat einen Kontaktpunkt $k_1 \notin \{u, x, v, y\}$

- Aus Symmetriegründen können wir $k_1 \in K(u, x)$ annehmen
- Da B weder zu e noch zu e' kompatibel ist, hat B weitere Kontaktpunkte $k_2 \in K(x, y)$ und $k_3 \in K(v, u)$, wobei $k_2 = k_3$ sein kann

Fall 1a: Ein Knoten $k_i \in \{k_2, k_3\}$ liegt auf dem Bogen $K(v, y)$

- In diesem Fall existieren 9 knotendisjunkte Pfade zwischen $\{u, x, k_i\}$ und $\{v, y, k_1\}$

Fall 1b: $K(v, y) \cap \{k_2, k_3\} = \emptyset$

- In diesem Fall ist $k_2 \in K(x, v]$ und $k_3 \in K[y, u)$
- Dann gibt es in B einen Knoten $w \notin \{k_1, k_2, k_3\}$, von dem aus knotendisjunkte Pfade zu k_1, k_2 und k_3 existieren
- Folglich gibt es 9 knotendisjunkte Pfade zwischen $\{u, w, x\}$ und $\{k_1, k_2, k_3\}$

Färben von planaren Graphen

Beweis (Fortsetzung)

Fall 2: Alle Kontaktpunkte von B liegen in der Menge $\{u, x, v, y\}$

- Da B inkompatibel zu e und e' ist, müssen in diesem Fall alle vier Knoten u, x, v, y zu B gehören
- Sei P_0 ein u-v-Pfad in B und sei P_1 ein x-y-Pfad in B
- Sei w der erste Knoten auf P_0, der auch auf P_1 liegt und sei z der letzte solche Knoten

Fall 2a: $w = z$

- Dann gibt es in B vier knotendisjunkte Pfade von w zu den vier Knoten u, x, v, y
- Somit existieren in M 10 knotendisjunkte Pfade zwischen den Knoten w, u, x, v, y

Beweis (Schluss)

Fall 2: Alle Kontaktpunkte von B liegen in der Menge $\{u, x, v, y\}$

- Da B inkompatibel zu e und e' ist, müssen in diesem Fall alle vier Knoten u, x, v, y zu B gehören
- Sei P_0 ein u-v-Pfad in B und sei P_1 ein x-y-Pfad in B
- Sei w der erste Knoten auf P_0, der auch auf P_1 liegt und sei z der letzte solche Knoten

Fall 2b: $w \neq z$

- Durch w und z wird der Pfad P_1 in drei Teilpfade $P_{x'w}$, P_{wz} und $P_{zy'}$ unterteilt, wobei die Indizes die Endpunkte bezeichnen und $\{x', y'\} = \{x, y\}$ ist
- Somit gibt es in B drei Pfade zwischen w und jedem Knoten in $\{u, z, x'\}$ und zwei Pfade zwischen z und jedem Knoten in $\{v, y'\}$, die alle 5 knotendisjunkt sind
- Folglich gibt es in M 9 knotendisjunkte Pfade zwischen $\{u, x', z\}$ und $\{v, y', w\}$ □

Färben von planaren Graphen

Eine unmittelbare Folgerung aus dem Satz von Kuratowski ist folgende Charakterisierung der Klasse der planaren Graphen

Korollar (Wagner 1937)

Ein Graph G ist genau dann planar, wenn er $\{K_{3,3}, K_5\}$-frei ist

Beweis.

- Falls G planar ist, kann G weder den $K_{3,3}$ noch den K_5 als Minor enthalten
- Falls G weder den $K_{3,3}$ noch den K_5 als Minor enthält, ist G nach Kuratowski planar, da G diese Graphen dann auch nicht als topologische Minoren enthalten kann $\square$

Färben von planaren Graphen

Satz (Robertson und Seymour, 1983-2004)

Für jede Graphklasse $\mathcal{K}$, die unter Minorenbildung abgeschlossen ist, gibt es eine endliche Menge $\mathcal{H}$ von Graphen mit

$$\mathcal{K} = \{G : G \text{ ist } \mathcal{H}\text{-frei}\}$$

- Wird $\mathcal{H}$ möglichst klein gewählt, so sind die Graphen in $\mathcal{H}$ bis auf Isomorphie eindeutig bestimmt und heißen verbotene Minoren für $\mathcal{K}$
- Eine Folgerung aus diesem Satz ist, dass jede unendliche Graphklasse zwei Graphen G und H enthält, so dass H ein Minor von G ist
- Das Problem, für zwei gegebene Graphen G und H zu entscheiden, ob H ein Minor von G ist, ist zwar NP-vollständig (da sich das Hamiltonkreisproblem darauf reduzieren lässt)
- Für einen festen Graphen H ist das Problem aber effizient entscheidbar

Färben von planaren Graphen

Satz (Robertson und Seymour, 1995)

Für jeden Graphen H gibt es einen $O(n^3)$-zeitbeschränkten Algorithmus, der für einen gegebenen Graphen G entscheidet, ob er H-frei ist

Korollar

Die Zugehörigkeit zu jeder unter Minorenbildung abgeschlossenen Graphklasse $\mathcal{K}$ ist in Zeit $O(n^3)$ entscheidbar

- Der Entscheidungsalgorithmus für $\mathcal{K}$ lässt sich allerdings nur angeben, wenn wir die verbotenen Minoren für $\mathcal{K}$ kennen
- Leider ist der Beweis des Satzes auf der vorigen Folie in dieser Hinsicht nicht konstruktiv
- Daher führt der Nachweis, dass $\mathcal{K}$ unter Minorenbildung abgeschlossen ist, nicht automatisch zu einem effizienten Erkennungsalgorithmus für $\mathcal{K}$
- Für die planaren Graphen sind die verbotenen Minoren jedoch bekannt; tatsächlich lässt sich Planarität in Linearzeit entscheiden

Färben von chordalen Graphen

- Chordale Graphen treten in vielen Anwendungen auf, z.B. sind alle Intervall- und alle Splitgraphen chordal
- Wir werden sehen, dass sich für chordale Graphen effizient eine optimale Knotenfärbung berechnen lässt

Definition

Ein Graph $G = (V, E)$ heißt chordal oder trianguliert, wenn jeder Kreis $K = (u_1, \ldots, u_\ell, u_1)$ der Länge $\ell \geq 4$ in G mindestens eine Sehne hat

Färben von chordalen Graphen

- Ein Graph G ist also genau dann chordal, wenn er keinen induzierten Kreis der Länge $\ell \geq 4$ enthält (ein induzierter Kreis ist ein induzierter Teilgraph $G[V']$, $V' \subseteq V$, der ein Kreis ist)
- Dies zeigt, dass die Klasse der chordalen Graphen unter induzierter Teilgraphbildung abgeschlossen ist (aber nicht unter Teilgraphbildung)
- Jede solche Graphklasse $\mathcal{G}$ ist durch eine Familie von (minimalen) verbotenen induzierten Teilgraphen H_i charakterisiert, die bis auf Isomorphie eindeutig bestimmt sind
- Die Graphen H_i gehören also nicht zu $\mathcal{G}$, aber sobald wir einen Knoten daraus entfernen, erhalten wir einen Graphen in $\mathcal{G}$
- Die Klasse der chordalen Graphen hat alle Kreise C_n der Länge $n \geq 4$ als verbotene induzierte Teilgraphen

Färben von chordalen Graphen

Definition

Ein x-y-Separator S heißt (inklusions-)minimal, wenn $S \setminus \{s\}$ für jedes $s \in S$ kein x-y-Separator ist

Lemma Für einen Graphen G sind folgende Aussagen äquivalent:

1. G ist chordal
2. Jeder minimale x-y-Separator S in G ist eine Clique
3. Jedes Paar von nicht adjazenten Knoten x und y in G hat einen x-y-Separator S, der eine Clique ist

Beweis von 1 $\Rightarrow$ 2

- Angenommen, G hat einen minimalen x-y-Separator S, der zwei nicht adjazente Knoten u und v enthält
- Seien $G[V_1]$ und $G[V_2]$ die beiden Komponenten in $G - S$ mit $x \in V_1$ und $y \in V_2$

Färben von chordalen Graphen

Beweis von 1 $\Rightarrow$ 2

- Angenommen, G hat einen minimalen x-y-Separator S, der zwei nicht adjazente Knoten u und v enthält
- Seien $G[V_1]$ und $G[V_2]$ die beiden Komponenten in $G - S$ mit $x \in V_1$ und $y \in V_2$
- Da $S' = S - \{s\}$ für jeden Knoten $s \in S$ kein x-y-Separator ist, existiert in $G[V - S']$ ein x-y-Pfad P, auf dem der Knoten s liegt
- Zudem muss der linke Nachbar von s auf dem Pfad P in V_1 und der rechte in V_2 liegen
- Insbesondere hat also in G jeder der beiden Knoten u und v sowohl einen Nachbarn in V_1 als auch in V_2
- Betrachte die beiden Teilgraphen $G_i = G[V_i \cup \{u, v\}]$ $(i = 1, 2)$ und wähle jeweils einen kürzesten u-v-Pfad P_i in G_i
- Da deren Länge ≥ 2 ist, ist $K = P_1 \cup P_2$ ein Kreis der Länge ≥ 4
- Nach Konstruktion ist zudem klar, dass K keine Sehnen in G hat $\square$

Färben von chordalen Graphen

Lemma

Für einen Graphen G sind folgende Aussagen äquivalent

1. G ist chordal
2. Jeder minimale x-y-Separator S in G ist eine Clique
3. Jedes Paar von nicht adjazenten Knoten x und y in G hat einen x-y-Separator S, der eine Clique ist

Beweis von 2 $\Rightarrow$ 3

Klar, da je zwei nicht adjazente Knoten x und y einen x-y-Separator S haben, und S eine Clique ist, wenn wir S inklusionsminimal wählen □

Färben von chordalen Graphen

Lemma

Für einen Graphen G sind folgende Aussagen äquivalent

1. G ist chordal
2. Jeder minimale x-y-Separator S in G ist eine Clique
3. Jedes Paar von nicht adjazenten Knoten x und y in G hat einen x-y-Separator S, der eine Clique ist

Beweis von 3 $\Rightarrow$ 1

- Angenommen, $G = (V, E)$ ist nicht chordal
- Dann gibt es in G einen induzierten Kreis K der Länge ≥ 4
- Auf diesem liegen zwei nicht adjazente Knoten x und y
- Dann muss jeder x-y-Separator S in G mindestens zwei Knoten u und v auf K enthalten
- Da K keine Sehne hat, ist $\{u, v\} \notin E$ und S ist keine Clique $\square$

Färben von chordalen Graphen

Definition Sei $G = (V, E)$ ein Graph und sei $k \geq 0$

- Ein Knoten $u \in V$ vom Grad k heißt *k-simplizial* (oder einfach *simplizial*), wenn alle Nachbarn von u paarweise adjazent sind
- G heißt *k-Baum*, wenn G aus K_k durch sukzessives Hinzufügen von k-simplizialen Knoten erzeugt werden kann

- Zusammenhängende chordale Graphen können als eine Verallgemeinerung von Bäumen aufgefasst werden
- Ein Graph ist ein Baum, wenn er aus K_1 durch sukzessives Hinzufügen von 1-simplizialen Knoten erzeugt werden kann
- Wir werden sehen, dass ein zusammenhängender Graph G genau dann chordal ist, wenn er aus K_1 durch sukzessives Hinzufügen von simplizialen Knoten erzeugt werden kann
- Äquivalent hierzu ist, dass G durch sukzessives Entfernen von simplizialen Knoten auf einen isolierten Knoten reduziert werden kann

Färben von chordalen Graphen

Definition Sei $G = (V, E)$ ein Graph

Eine lineare Ordnung $(u_1, \ldots, u_n)$ auf V heißt perfekte Eliminationsordnung (PEO) von G, wenn u_i für $i = 1, \ldots, n$ simplizial in $G[u_1, \ldots, u_i]$ ist

- Wir eliminieren die Knoten von G also in der Reihenfolge $u_n, \ldots, u_2, u_1$
- Es ist klar dass der K_n alle $n!$ lineare Ordnungen auf V als PEO hat
- Das folgende Lemma verallgemeinert die bekannte Tatsache, dass jeder nicht vollständige Baum T (also $T \notin \{K_1, K_2\}$) mindestens zwei nicht adjazente Blätter hat

Färben von chordalen Graphen

Lemma

Falls ein chordaler Graph $G = (V, E)$ nicht vollständig ist, besitzt er mindestens zwei simpliziale Knoten, die nicht adjazent sind

Beweis durch Induktion über die Knotenzahl n von G

- Für $n \leq 2$ (IA) ist die Behauptung klar
- Für den IS sei $n(G) \geq 3$
- Da G nicht vollständig ist, enthält G zwei nichtadjazente Knoten x_1 und x_2
- Sei S ein minimaler x_1-x_2-Separator der Größe $k \geq 0$
- Dann ist S nach obigem Lemma eine Clique in G
- Seien $G[V_1]$ und $G[V_2]$ die beiden Komponenten von $G - S$ mit $x_i \in V_i$
- Wir zeigen die Existenz zweier simplizialer Knoten $s_i \in V_i$, $i = 1, 2$

Färben von chordalen Graphen

Beweis (Fortsetzung)

- Betrachte die Teilgraphen $G_i = G[V_i \cup S]$
- Da G_i chordal ist und weniger als n Knoten hat, ist G_i nach IV entweder eine Clique oder G_i enthält mindestens zwei nicht adjazente simpliziale Knoten y_i, z_i
- Falls G_i eine Clique ist, ist $s_i = x_i$ simplizial in G_i, und da x_i keine Nachbarn außerhalb von $V_i \cup S$ hat, ist s_i dann auch simplizial in G
- Ist G_i keine Clique, kann höchstens einer der beiden Knoten y_i, z_i zu S gehören (da S eine Clique und $\{y_i, z_i\} \notin E$ ist)
- O.B.d.A. sei $y_i \in V_i$; dann hat $s_i = y_i$ keine Nachbarn außerhalb von $V_i \cup S$ und somit ist s_i auch simplizial in G $\square$

Färben von chordalen Graphen

Satz

Ein Graph ist genau dann chordal, wenn er eine PEO hat

Beweis.

- Falls G chordal ist, lässt sich eine PEO gemäß obigem Lemma bestimmen, indem wir für $i = n, \dots, 2$ sukzessive einen simplizialen Knoten u_i in $G - \{u_{i+1}, \dots, u_n\}$ wählen
- Für die umgekehrte Richtung sei $(u_1, \dots, u_n)$ eine PEO von G
- Wir zeigen induktiv, dass $G_i = G[u_1, \dots, u_i]$ chordal ist
- Für $i \leq 3$ (IA) ist das klar
- Für den IS ($i \geq 4$) benutzen wir, dass u_i simplizial in G_i ist
- Daher enthält jeder Kreis K der Länge ≥ 4 in G_i, auf dem u_i liegt, eine Sehne zwischen den beiden Kreisnachbarn von u_i
- Also ist mit G_{i-1} auch G_i chordal $\square$

Färben von chordalen Graphen

Korollar

Es gibt einen Polynomialzeitalgorithmus A, der für einen gegebenen Graphen G eine PEO berechnet, falls G chordal ist, und andernfalls einen induzierten Kreis der Länge ≥ 4 ausgibt

Beweis.

- A versucht wie im Beweis von obigem Satz eine PEO zu bestimmen
- Stellt sich heraus, dass $G_i = G - \{u_{i+1}, \ldots, u_n\}$ keinen simplizialen Knoten u_i hat, so ist G_i (und somit auch G) nicht chordal
- Daher gibt es in G_i zwei nicht adjazente Knoten x und y, für die kein x-y-Separator eine Clique ist
- Berechnen wir für x und y einen beliebigen minimalen x-y-Separator S, so ist S keine Clique
- Wie im Beweis von 1 $\Rightarrow$ 2 auf Folie 66 können wir dann einen induzierten Kreis K der Länge ≥ 4 in G_i konstruieren
- Da G_i ein induzierter Teilgraph von G ist, ist K auch ein induzierter Kreis in G $\square$

Färben von chordalen Graphen

Algorithmus `chordal-color`(V, E)

berechne eine PEO $(u_1, \ldots, u_n)$ für $G = (V, E)$
starte `greedy-color` mit der Knotenfolge $(u_1, \ldots, u_n)$

Satz

Für einen gegebenen chordalen Graphen $G = (V, E)$ berechnet `chordal-color` eine k-Färbung c von G mit $k = \chi(G) = \omega(G)$

Beweis.

- Sei k die größte von `chordal-color` zugewiesene Farbe und sei u_i ein beliebiger Knoten mit $c(u_i) = k$
- Da $(u_1, \ldots, u_n)$ eine PEO von G ist, ist u_i simplizial in $G[u_1, \ldots, u_i]$
- Somit bilden die Nachbarn u_j von u_i mit $j < i$ eine Clique und wegen $c(u_i) = k$ bilden sie zusammen mit u_i eine k-Clique
- Also gilt $\chi(G) \le k \le \omega(G)$, woraus wegen $\omega(G) \le \chi(G)$ die Gleichheit $k = \chi(G) = \omega(G)$ folgt $\square$

Färben von chordalen Graphen

- Um `chordal-color` in Linearzeit zu implementieren, benötigen wir einen Linearzeit-Algorithmus zur Bestimmung einer PEO
- Rose, Tarjan und Lueker haben 1976 einen solchen Algorithmus angegeben, der auf lexikografischer Breitensuche (kurz LexBFS oder LBFS, engl. lexicographic breadth-first search) basiert
- Bevor wir diese Variante der Breitensuche vorstellen, gehen wir kurz auf verschiedene Ansätze zum Durchsuchen von Graphen ein
- Der Algorithmus `GraphSearch` wählt einen beliebigen Knoten und sucht zunächst nach allen von diesem Knoten aus erreichbaren Knoten
- Wird bei dieser Suche ein neuer Knoten v ausgehend von einem Knoten u über die Kante $\{u, v\}$ entdeckt, so wird v zur Ausgabeliste L hinzugefügt und $\texttt{parent}(v) = u$ gesetzt
- Zudem werden die bereits entdeckten, aber noch nicht abgearbeiteten Knoten in der Menge A der aktiven Knoten gespeichert
- Sobald alle aktiven Knoten abgearbeitet sind, wird eine neue Suche von einem bisher unerreichten Knoten gestartet, sofern er existiert

Algorithmus GraphSearch(V, E)

$R := \emptyset$ // Menge der erreichten Knoten
$L := ()$ // Ausgabeliste
repeat
  wähle $u \in V \setminus R$
  append(L, u) // u wurde neu entdeckt
  parent$(u) := \perp$
  $R := R \cup \{u\}$
  $A := \{u\}$ // Menge der aktiven Knoten
  while $A \neq \emptyset$ **do**
    wähle u aus A
    if $\exists v \in N(u) \setminus R$ **then**
      append(L, v) // v wurde neu entdeckt
      parent$(v) := u$
      $R := R \cup \{v\}$; $A := A \cup \{v\}$
    else entferne u aus A // u ist abgearbeitet
until $R = V$
return(L)

Durchsuchen von Graphen

- Der Algorithmus `GraphSearch`(V, E) findet in jedem Durchlauf der repeat-Schleife eine neue Komponente des Eingabegraphen $G = (V, E)$
- Dies bedeutet, dass alle Knoten, die zu einer Komponente gehören, konsekutiv in der Ausgabeliste $L = (u_1, \ldots, u_n)$ auftreten
- Abgesehen vom ersten Knoten jeder Komponente hat dabei jeder Knoten u_k einen Nachbarn $u_i = \texttt{parent}(u_k)$ mit $i < k$
- Wir fassen diese Eigenschaften der Ausgabeliste L in folgender Definition zusammen

Definition Sei $G = (V, E)$ ein Graph

Eine lineare Ordnung $(u_1, \ldots, u_n)$ auf V heißt Suchordnung (SO) von G, wenn für jedes Tripel $j < k < \ell$ gilt:

$$u_j \in N(u_\ell) \setminus N(u_k) \Rightarrow \exists i < k\colon i \neq j \wedge u_i \in N(u_k)$$

Satz

Für jeden Graphen $G = (V, E)$ gibt der Algorithmus $\texttt{GraphSearch}(V, E)$ eine SO von G aus

Beweis.

- Ein Knoten u_k erhält nur dann den Wert $\texttt{parent}(u_k) = \perp$, wenn alle Knoten u_j mit $j < k$ bereits abgearbeitet sind und diese nur Nachbarn u_ℓ mit $\ell < k$ haben
- Falls also ein Vorgänger u_j von u_k mit einem Nachfolger u_ℓ von u_k verbunden ist, liefert die `parent`-Funktion einen Nachbarn $u_i = \texttt{parent}(u_k)$ von u_k mit $i < k$
- Da $u_j \notin N(u_k)$ ist, gilt zusätzlich $i \neq j$ $\square$

Durchsuchen von Graphen

- Die $\mathtt{parent}$-Funktion liefert einen gerichteten Wald $W = (V, E_{\mathtt{parent}})$, dessen Kantenmenge $E_{\mathtt{parent}}$ aus allen Kanten der Form $(\mathtt{parent}(v), v)$ mit $\mathtt{parent}(v) \neq \perp$ besteht
- Die Kanten von W werden auch als Baumkanten (kurz B-Kanten) und W wird auch als Suchwald von $G = (V, E)$ bezeichnet
- Für jeden Knoten $v \in V$ gibt es genau eine Wurzel w in W, von der aus v in W erreichbar ist
- Der eindeutig bestimmte w-v-Pfad $P = (u_0, \ldots, u_l)$ in W mit $u_0 = w$ und $u_l = v$ lässt sich ausgehend von $u_l = v$ unter Verwendung der $\mathtt{parent}$-Funktion mittels $u_{i-1} = \mathtt{parent}(u_i)$ für $i = l, \ldots, 1$ berechnen
- P wird auch als $\mathtt{parent}$-Pfad von v bezeichnet
- Es ist klar, dass 2 Knoten v und v' genau dann in einer Komponente von G liegen, wenn sie die gleiche Wurzel haben

Durchsuchen von Graphen

- Realisieren wir die Menge A der aktiven Knoten als einen Keller S, so erhalten wir eine spezielle Suchstrategie, die als Tiefensuche (kurz DFS, engl. depth first search) bezeichnet wird
- Die Benutzung eines Kellers bewirkt, dass die Suche nach Entdeckung eines neuen Knotens $v \in N(u)$ zuerst bei einem Nachbarn von v fortgesetzt wird, bevor weitere Nachbarn von $u = \texttt{parent}(v)$ an die Reihe kommen
- Im Gegensatz zur DFS werden bei der Breitensuche (kurz BFS, engl. breadth first search) zuerst alle anderen Nachbarknoten von $u = \texttt{parent}(v)$ entdeckt bevor die Suche bei v fortgesetzt wird
- Diese spezielle Suchstrategie erhält man, wenn die Menge A der aktiven Knoten als eine Warteschlange Q realisiert wird
- In diesem Fall findet der resultierende Algorithmus $\texttt{BFS}(V, E)$ einen kürzesten Weg vom Startknoten u zu allen von u aus erreichbaren Knoten

Algorithmus `GraphSearch`(V, E)

$R := \emptyset$ // Menge der erreichten Knoten
$L := ()$ // Ausgabeliste
repeat
  wähle $u \in V \setminus R$
  append(L, u) // u wurde neu entdeckt
  parent$(u) := \perp$
  $R := R \cup \{u\}$
  $A := \{u\}$ // Menge der aktiven Knoten
  while $A \neq \emptyset$ **do**
    wähle u aus A
    if $\exists v \in N(u) \setminus R$ **then**
      append(L, v) // v wurde neu entdeckt
      parent$(v) := u$
      $R := R \cup \{v\}$; $A := A \cup \{v\}$
    else entferne u aus A // u ist abgearbeitet
until $R = V$
return(L)

Algorithmus DFS(V, E)

```
R := ∅            // Menge der erreichten Knoten
L := ()           // Ausgabeliste
repeat
   wähle u ∈ V \ R
   append(L, u)       // u wurde neu entdeckt
   parent(u) := ⊥
   R := R ∪ {u}
   S := (u)           // Keller der aktiven Knoten
   while S ≠ () do
      u := top(S)
      if ∃v ∈ N(u) \ R then
         append(L, v)       // v wurde neu entdeckt
         parent(v) := u
         R := R ∪ {v}; push(S, v)
      else pop(S)                  // u ist abgearbeitet
until R = V
return(L)
```

Durchsuchen von Graphen

Definition Sei $G = (V, E)$ ein Graph.

Eine lineare Ordnung $(u_1, \dots, u_n)$ auf V heißt Tiefensuchordnung von G (auch DFS-Ordnung oder kurz DO), wenn für jedes Tripel $j < k < \ell$ gilt:

$$u_j \in N(u_\ell) \setminus N(u_k) \Rightarrow \exists i\colon\ j < i < k \wedge u_i \in N(u_k)$$

Satz

Für jeden Graphen $G = (V, E)$ gibt $\mathtt{DFS}(V, E)$ eine DO von G aus

Beweis.

Siehe Übungen □

Algorithmus GraphSearch(V, E)

```
R := ∅        // Menge der erreichten Knoten
L := ()       // Ausgabeliste
repeat
  wähle u ∈ V \ R
  append(L, u)      // u wurde neu entdeckt
  parent(u) := ⊥
  R := R ∪ {u}
  A := {u}      // Menge der aktiven Knoten
  while A ≠ ∅ do
    wähle u aus A
    if ∃v ∈ N(u) \ R then
      append(L, v)     // v wurde neu entdeckt
      parent(v) := u
      R := R ∪ {v}; A := A ∪ {v}
    else entferne u aus A     // u ist abgearbeitet
until R = V
return(L)
```

Algorithmus $\mathtt{BFS}(V, E)$

```
R := ∅        // Menge der erreichten Knoten
L := ()       // Ausgabeliste
repeat
   wähle u ∈ V \ R
   append(L, u)       // u wurde neu entdeckt
   parent(u) := ⊥
   R := R ∪ {u}
   Q := (u)       // Warteschlange der aktiven Knoten
   while Q ≠ () do
      u := dequeue(Q)     // u wird bei Entnahme komplett abgearbeitet
      for all v ∈ N(u) \ R do
         append(L, v)     // v wurde neu entdeckt
         parent(v) := u
         R := R ∪ {v}
         enqueue(Q, v)
until R = V
return(L)
```

Durchsuchen von Graphen

Definition Sei $G = (V, E)$ ein Graph.

Eine lineare Ordnung $(u_1, \ldots, u_n)$ auf V heißt Breitensuchordnung von G (auch BFS-Ordnung oder kurz BO), wenn für jedes Tripel $j < k < \ell$ gilt:

$$u_j \in N(u_\ell) \setminus N(u_k) \Rightarrow \exists i < j \colon u_i \in N(u_k)$$

Satz

$\mathtt{BFS}(V, E)$ gibt für jeden Graphen $G = (V, E)$ eine BO von G aus

Beweis.

- Sei $u_j \in N(u_\ell) \setminus N(u_k)$ für ein Tripel $j < k < \ell$
- Da $\mathtt{BFS}(V, E)$ eine SO von G ausgibt, wissen wir bereits, dass der Knoten $u_i = \mathtt{parent}(u_k) \in N(u_k)$ einen Index $i < k$ hat
- Da u_k beim Abarbeiten von u_i zu Q hinzugefügt wird, muss u_i vor u_j entdeckt (bzw. abgearbeitet) werden, da u_ℓ im Fall $j < i$ vor u_k zu Q hinzugefügt würde (nämlich spätestens beim Abarbeiten von u_j) $\square$

Durchsuchen von Graphen

- BFS-Ordnungen lassen sich anschaulich anhand der Adjazenzmatrix charakterisieren
- Sei $(u_1, \ldots, u_n)$ eine BO für $G = (V, E)$ und sei $A = (a_{ij})$ die Adjazenzmatrix von G mit $a_{ij} = 1 \Leftrightarrow \{u_i, u_j\} \in E$
- Weiter seien $z_i = a_{i1} \ldots a_{i,i-1}$ die Präfixe der Zeilen von A, die unterhalb der Diagonale verlaufen
- Sind nun die ersten j Einträge $a_{k1} \ldots a_{kj}$ einer Zeile s_k Null, so muss dies auch für jede Zeile s_ℓ mit $\ell > k$ gelten, da im Fall $a_{\ell j} = 1$ der Knoten $u_j \in N(u_\ell) \setminus N(u_k)$ wäre und somit ein $i < j$ mit $a_{ki} = 1$ existieren müsste
- Dies bedeutet, dass s_ℓ mindestens so viele Nullen als Präfix hat wie z_k
- Es ist aber möglich, dass z_k bspw. mit 00010... beginnt und z_ℓ mit 00011...

Definition Sei $G = (V, E)$ ein Graph.

Eine lineare Ordnung $(u_1, \ldots, u_n)$ auf V heißt **LexBFS-Ordnung (LBO) von G**, wenn für jedes Tripel $j < k < l$ gilt:

$$u_j \in N(u_l) \setminus N(u_k) \Rightarrow \exists i < j \colon u_i \in N(u_k) \setminus N(u_l)$$

- Ob eine Ordnung $(u_1, \ldots, u_n)$ eine LBO ist, lässt sich wie folgt an der gemäß $(u_1, \ldots, u_n)$ geordneten Adjazenzmatrix A ablesen
- Setzen wir in A alle Einträge a_{ij} mit $i \leq j$ auf 1 und betrachten die Zeilen $z_i' = a_{i1} \ldots a_{i,i-1} 1^{n-i+1}$ der resultierenden Matrix, so sind diese lexikographisch sortiert: $z_i' \geq z_{i+1}'$
- Man erhält sogar eine lexikografische Ordnung auf den Zeilen von A, wenn man nur die Diagonale auf 1 setzt und die Knoten in jeder Menge T_i von Q nach absteigendem Knotengrad in G sortiert

- Bei einer Breitensuche werden die noch nicht besuchten Nachbarn des aktiven Knotens in beliebiger Reihenfolge zur Warteschlange hinzugefügt und auch wieder in dieser Reihenfolge entfernt
- Dagegen werden bei einer LexBFS-Suche die Knoten in der Warteschlange nachträglich umsortiert, falls dies notwendig ist, um eine LexBFS-Ordnung der Knoten zu erhalten
- Der Name von LexBFS rührt daher, dass die Knoten in einer Reihenfolge ausgegeben werden, die eine lexikografische Sortierung der mit Einsen auf die Länge n aufgefüllten Zeilenpräfixe $z_i' = z_i 1^{n-i+1}$ bewirkt
- Eine solche Sortierung kann auch bei einer gewöhnlichen Breitensuche auftreten, ist bei dieser aber nicht garantiert
- In einem ersten Schritt modifizieren wir den `BFS`-Algorithmus, ohne seine Semantik zu verändern
- Anschließend verfeinern wir den resultierenden Algorithmus `BFS'` zum gewünschten `LexBFS`-Algorithmus

- Um eine nachträgliche Umsortierung der Knoten in der Warteschlange zu ermöglichen, bietet es sich an, Q als eine Warteschlange von Knotenmengen T_i zu realisieren (siehe Algorithmus $\mathtt{BFS}'$)
- Innerhalb jeder Menge T_i können die Knoten dann eine beliebige Reihenfolge annehmen, aber die Knoten in T_i müssen vor den Knoten in T_{i+1} zu L hinzugefügt werden
- Entsprechend kann $\mathtt{BFS}'$ als nächsten Knoten jeweils ein beliebiges Element aus der ersten Menge in Q auswählen und abarbeiten
- Hierzu benutzt $\mathtt{BFS}'$ die Prozedur `Dequeue`, welche einen Knoten aus der ersten Menge in Q entfernt und zurückliefert
- Im Unterschied zu `BFS` fügt $\mathtt{BFS}'$ die Knoten erst bei ihrer Entnahme aus der Warteschlange zur Liste L hinzu
- Dies ändert zwar nichts an den möglichen Ausgabefolgen von $\mathtt{BFS}'$
- Die Zeit zwischen dem Einfügen der Knoten in Q und ihrer Entnahme kann jedoch von `LexBFS` zur Umsortierung der Knoten genutzt werden

Algorithmus $\mathtt{BFS}'(V, E)$

$R := \emptyset$ // Menge der erreichten Knoten
$L := ()$ // Ausgabeliste
repeat
  wähle $u \in V \setminus R$; $R := R \cup \{u\}$
  $Q := (\{u\})$ // Warteschlange von Knotenmengen
  while $Q \neq ()$ **do**
    $u :=$ Dequeue(Q) // u wird komplett abgearbeitet
    append(L, u)
    if $N(u) \not\subseteq R$ **then** enqueue$(Q, N(u) \setminus R)$
    $R := R \cup N(u)$
until $R = V$
return(L)

Prozedur $\mathtt{Dequeue}(Q)$

entferne u aus first(Q)
if first$(Q) = \emptyset$ **then** dequeue(Q)
return(u)

- Da die Mengen in Q eine Partition von R bilden, können wir die Menge $V \setminus R$ der noch nicht erreichten Knoten als weitere Menge hinzunehmen, um eine Partition von V zu erhalten
- Von dieser Menge $V \setminus R$ spaltet $\mathtt{BFS}'$ beim Abarbeiten eines Knotens u die Teilmenge $(V \setminus R) \cap N(u) = N(u) \setminus R$ ab und fügt sie im Fall $N(u) \not\subseteq R$ als neue Menge zu Q hinzu
- Wie $\mathtt{BFS}'$ zerlegt auch $\mathtt{LexBFS}$ die Menge V in eine Folge von Knotenmengen T_i und speichert diese in Q (inklusive $V \setminus R$ als letzter Menge)
- Im Unterschied zu $\mathtt{BFS}'$ unterteilt $\mathtt{LexBFS}$ die Mengen in Q aber noch feiner und schränkt dadurch die möglichen Ausgabefolgen stärker ein
- Konkret spaltet $\mathtt{LexBFS}$ beim Abarbeiten eines Knotens u nicht nur die Menge $V \setminus R$ sondern alle Mengen T_i in Q mit $T_i \cap N(u) \notin \{\emptyset, T_i\}$ in die beiden Teilmengen $T_i \cap N(u)$ und $T_i \setminus N(u)$

Durchsuchen von Graphen

Algorithmus $\texttt{LexBFS}(V, E)$

$L := ()$ // Ausgabeliste
$Q := (V)$ // Warteschlange von Knotenmengen
while $Q \neq ()$ **do**
  $u := \texttt{Dequeue}(Q)$ // u wird komplett abgearbeitet
  append(L, u)
  $\texttt{Splitqueue}(Q, N(u))$
return(L)

Prozedur $\texttt{Splitqueue}(Q, S)$

for T in Q with $T \cap S \notin \{\emptyset, T\}$ **do**
  ersetze die Teilfolge (T) in Q durch $(T \cap S, T \setminus S)$

Durchsuchen von Graphen

- Für eine effiziente Implementierung sollte die Schlange $Q = (T_1, \ldots, T_k)$ von Knotenmengen $T_i \subseteq V$ als doppelt verkettete Liste realisiert werden
- Zudem sollte die for-Schleife in der Prozedur `Splitqueue` durch eine Schleife über die Knoten v in der Adjazenzliste $S = N(u)$ ersetzt werden
- Weiterhin sollte für jeden Knoten v in der Adjazenzliste ein Zeiger auf die Menge T_i, die v enthält und auf seinen Eintrag in T_i gespeichert werden

Durchsuchen von Graphen

Definition Sei $G = (V, E)$ ein Graph.

Eine lineare Ordnung $(u_1, \ldots, u_n)$ auf V heißt LexBFS-Ordnung (LBO) von G, wenn für jedes Tripel $j < k < l$ gilt:

$$u_j \in N(u_l) \setminus N(u_k) \Rightarrow \exists i < j \colon u_i \in N(u_k) \setminus N(u_l)$$

Satz

Für jeden Graphen (V, E) gibt $\texttt{LexBFS}(V, E)$ eine LBO $(u_1, \ldots, u_n)$ aus

Beweis.

- Sei $A = (a_{ij})$ die Adjazenzmatrix von G mit $a_{ij} = 1 \Leftrightarrow \{u_i, u_j\} \in E$
- Wir zeigen, dass $(u_1, \ldots, u_n)$ eine LBO ist
- Existiert nämlich im Fall $k < \ell$ ein Index $j < k$ mit $a_{kj} = 0$ und $a_{\ell j} = 1$, so muss es einen Index $i < j$ mit $a_{ki} = 1$ und $a_{\ell i} = 0$ geben
- Ansonsten wäre der Knoten u_ℓ spätestens beim Abarbeiten von u_j in eine Menge vor dem Knoten u_k sortiert worden und könnte daher nicht nach dem Knoten u_k ausgegeben werden $\square$

Färben von chordalen Graphen

Satz

Jede LBO für einen chordalen Graphen G ist eine PEO für G

Beweis.

- Sei $(u_1, \ldots, u_n)$ eine LBO für $G = (V, E)$ und sei $A = (a_{ij})$ die Adjazenzmatrix von G mit $a_{ij} = 1 \Leftrightarrow \{u_i, u_j\} \in E$, wobei wir für a_{ij} auch $A[i, j]$ schreiben
- Wir zeigen, dass G nicht chordal ist, wenn u_i nicht simplizial in $G_i = G[u_1, \ldots, u_i]$ ist, indem wir einen induzierten Kreis $G[u_{i_1}, u_{i_2}, \ldots, u_{i_k}]$ der Länge $k \geq 4$ in G konstruieren
- Falls u_i nicht simplizial in G_i ist, müssen Indizes $i_3 < i_2 < i_1 := i$ mit $A[i_1, i_2] = A[i_1, i_3] = 1$ und $A[i_2, i_3] = 0$ existieren
- Wir erhalten also für $j = 1$ die beiden Gleichungen

$$A[i_j, i_{j+2}] = 1 \text{ und } A[i_{j+1}, i_{j+2}] = 0 \quad (*)$$

Färben von chordalen Graphen

Beweis (Fortsetzung)

- Wir erhalten also für $j = 1$ die beiden Gleichungen

  $$A[i_j, i_{j+2}] = 1 \text{ und } A[i_{j+1}, i_{j+2}] = 0 \quad (*)$$

- Daher muss es einen Index $i_{j+3} < i_{j+2}$ mit $A[i_{j+1}, i_{j+3}] = 1$ und $A[i_j, i_{j+3}] = 0$ geben, wobei wir i_{j+3} möglichst klein wählen
- Dies impliziert $(A[i_{j+1}, r], A[i_j, r]) \neq (1, 0)$ für alle $r < i_{j+3}$
- Zudem folgt $(A[i_{j+1}, r], A[i_j, r]) \neq (0, 1)$ für alle $r < i_{j+3}$, da sonst ein $s < r$ mit $(A[i_{j+1}, s], A[i_j, s]) = (1, 0)$ existieren würde
- Folglich gilt $A[i_{j+1}, r] = A[i_j, r]$ für alle $r < i_{j+3}$ $(**)$
- Falls $A[i_{j+2}, i_{j+3}] = 1$ ist, haben wir für $k := j + 3$ einen Kreis K der Länge $k \geq 4$ mit den Kanten $\{u_{i_1}, u_{i_2}\}$, $\{u_{i_j}, u_{i_{j+2}}\}$ für $j = 1, \ldots, k - 2$, und $\{u_{i_{k-1}}, u_{i_k}\}$ gefunden
- Andernfalls gelten die Gleichungen $(*)$ auch für $j + 1$ anstelle von j und wir können die Indexfolge $i_1 > i_2 > \cdots > i_{j+3}$ weiter fortsetzen

Färben von chordalen Graphen

Beweis. (Schluss)

- Da diese jedoch streng monoton fällt und nur positive Werte annimmt, muss spätestens für $k = n$ der Wert $A[i_{k-1}, i_k] = 1$ sein und es folgt
  1. $A[i_1, i_2] = A[i_j, i_{j+2}] = A[i_{k-1}, i_k] = 1$ für $j = 1, \dots, k-1$ und
  2. $A[i_1, i_4] = A[i_j, i_{j+1}] = A[i_j, i_{j+3}] = A[i_{k-2}, i_{k-1}] = 0$, $2 \le j \le k-3$
- Wegen $(**)$ erfüllt die Indexfolge $i_1, i_2, \dots, i_k$ auch die Gleichungen

$$A[i_j, i_{j'}] = A[i_{j+1}, i_{j'}], \text{ falls } i_{j'} < i_{j+3} \text{ (bzw. } j' \ge j+4) \text{ ist}$$

- Also sind in Spalte $i_{j'}$, $j' \ge 5$, die Einträge der Zeilen $i_1, \dots, i_{j'-3}$ gleich
- Da zudem $A[i_{j'-3}, i_{j'}]$ den Wert 0 hat (siehe 2), folgt
  3. $A[i_1, i_{j'}] = A[i_2, i_{j'}] = \dots = A[i_{j'-3}, i_{j'}] = 0$ für $j' = 4, \dots, k$
- Somit gilt für alle Paare $1 \le j < j' \le k$ die Äquivalenz

$$A[i_j, i_{j'}] = 1 \Leftrightarrow j' = j+2 \text{ oder } j' = j+1 \land j \in \{0, k-1\},$$

weshalb $G[u_{i_1}, \dots, u_{i_k}]$ ein induzierter Kreis in G ist □

Färben von chordalen Graphen

- Damit haben wir einen Linearzeitalgorithmus, der für chordale Graphen eine PEO berechnet
- Da auch `greedy-color` linear zeitbeschränkt ist, können wir den Algorithmus `chordal-color` in Linearzeit implementieren
- Diesen Algorithmus können wir noch so modifizieren, dass er zusammen mit der gefundenen k-Färbung
  - entweder eine Clique C der Größe k (als Zertifikat, dass $\chi(G) = k = \omega(G)$ ist)
  - oder einen induzierten Kreis der Länge ≥ 4 ausgibt (als Zertifikat, dass G nicht chordal ist)
- Tatsächlich findet `chordal-color` auch für viele nicht chordale Graphen eine optimale Färbung (also auch wenn `LexBFS` keine PEO liefert)
- Dies gilt zumindest für alle DH-Graphen (engl. distance hereditary graphs), die Distanzen an ihre induzierten Subgraphen vererben

Der Satz von Brooks

Satz (Brooks 1941)

Für einen zusammenhängenden Graphen G gilt $\chi(G) = \Delta(G) + 1$ genau dann, wenn $G = K_n$ für ein $n \geq 1$ oder $G = C_n$ für ein ungerades $n \geq 3$ ist

Beweis.

- Es ist klar, dass die Graphen $G = K_n$ für $n \geq 1$ und $G = C_n$ für ungerades $n \geq 3$ die chromatische Zahl $\Delta(G) + 1$ haben
- Um zu zeigen, dass dies die einzigen zusammenhängenden Graphen mit $\chi(G) = \Delta(G) + 1$ sind, betrachten wir verschiedene Fälle
- Falls G nicht regulär ist, können wir ausgehend von einem Knoten u_1 vom Grad $\deg_G(u_1) < \Delta(G)$ eine Suchordnung $(u_1, \ldots, u_n)$ berechnen und G greedy in der umgekehrten Reihenfolge $(u_n, \ldots, u_1)$ $\Delta(G)$-färben
- Dies ist möglich, da jeder Knoten u_i mit $i \geq 2$ zum Zeitpunkt der Berechnung von $c(u_i)$ noch einen ungefärbten Nachbar $\texttt{parent}(u_i)$ und u_1 einen Grad $\deg_G(u_1) < \Delta(G)$ hat

Beweis des Satzes von Brooks (Fortsetzung)

- Falls G regulär, aber nicht 2-zusammenhängend ist, berechnen wir $\Delta(G)$-Färbungen c_i für die einzelnen Blöcke B_i von G
- Dies ist möglich, da jeder Block B_i mindestens einen Schnittknoten enthält und daher höchstens für ein $k < \Delta(G)$ k-regulär ist
- Die Färbungen c_i lassen sich ausgehend von einem beliebigen Wurzelblock des BC-Baums hin zu den Blattblöcken in eine $\Delta(G)$-Färbung c für G transformieren
- Hierzu müssen wir lediglich die Farben im aktuellen Block B_i so umbenennen, dass der Schnittknoten, der B_i mit seinem Elternblock verbindet, die vorgegebene Farbe erhält
- Es bleibt also der Fall, dass G d-regulär und $\kappa(G) \geq 2$ ist
- Der Fall $d = 2$ ist klar, da G ein Kreis sein muss
- Für den Fall $d \geq 3$ benutzen wir folgende Behauptung

Der Satz von Brooks

Behauptung Sei $d \geq 3$ und sei $G \neq K_n$ ein d-regulär Graph mit $\kappa(G) \geq 2$.

Dann gibt es in G einen Knoten u_1, der zwei nicht-adjazente Nachbarn a und b hat, so dass $G - \{a, b\}$ zusammenhängend ist

Beweis des Satzes von Brooks (Schluss)

- Sei also u_1 ein Knoten, der zwei Nachbarn a und b mit $\{a, b\} \notin E$ hat, so dass $G - \{a, b\}$ zusammenhängend ist
- Durchsuchen wir den Graphen $G - \{a, b\}$ ausgehend vom Startknoten u_1, so erhalten wir eine Suchordnung $(u_1, \ldots, u_{n-2})$
- Färben wir G nun greedy mit der Knotenfolge $(a, b, u_{n-2}, \ldots, u_1)$, so erhalten wir eine d-Färbung c für G mit $c(a) = c(b) = 1$
- Zudem hat jeder Knoten u_i mit $i > 1$ einen Nachbarn u_j mit $j < i$, weshalb $c(u_i) \leq \deg(u_i) \leq d$ ist
- Zuletzt erhält auch u_1 eine Farbe $c(u_1) \leq d$, da die Nachbarn a und b von u_1 dieselbe Farbe haben $\square$

Der Satz von Brooks

Behauptung Sei $d \geq 3$ und sei $G \neq K_n$ ein d-regulär Graph mit $\kappa(G) \geq 2$.

Dann gibt es in G einen Knoten u_1, der zwei nicht-adjazente Nachbarn a und b hat, so dass $G - \{a, b\}$ zusammenhängend ist

Beweis der Behauptung

- Da $G \neq K_n$ ist, gibt es einen Knoten x, der zwei Nachbarn $y, z \in N(x)$ mit $\{y, z\} \notin E$ hat
- Falls $G - y$ 2-zusammenhängend ist, ist $G - \{y, z\}$ zusammenhängend und die Behauptung folgt für $u_1 = x$
- Ist $G - y$ nicht 2-zusammenhängend, d.h. $G - y$ hat mindestens zwei Blöcke, dann hat der BC-Baum T von $G - y$ mindestens zwei Blätter
- Da $\kappa(G) \geq 2$ ist, ist y in G zu mindestens einem Knoten in jedem Blatt von T benachbart, der kein Schnittknoten ist
- Wählen wir für a und b zwei dieser Knoten in verschiedenen Blättern, so ist $G - \{a, b\}$ zusammenhängend und somit die Behauptung für $u_1 = y$ bewiesen $\square$

Kantenfärbungen

Neben der Frage, wieviele Farben eine Knotenfärbung eines Graphen benötigt, ist bei vielen Anwendungen auch die Anzahl der Farben für eine Kantenfärbung interessant

Definition Sei $G = (V, E)$ ein Graph und sei $k \in \mathbb{N}$.

- Eine Abbildung $c\colon E \to \{1, \ldots, k\}$ heißt k-Kantenfärbung von G, wenn $c(e) \neq c(e')$ für alle $e \neq e' \in E$ mit $e \cap e' \neq \emptyset$ gilt
- In diesem Fall heißt G k-kantenfärbbar
- Die kantenchromatische Zahl oder der chromatische Index von G ist

  $$\chi'(G) = \min\{k \in \mathbb{N} : G \text{ ist } k\text{-kantenfärbbar}\}$$

- Eine Kantenmenge $M \subseteq E$ heißt Matching in G, falls je zwei Kanten $e \neq e' \in M$ unabhängig sind, d.h. $e \cap e' = \emptyset$
- Die Matchingzahl von G ist

  $$\mu(G) = \max\{|M| : M \text{ ist ein Matching in } G\}$$

Kantenfärbungen

- Eine k-Kantenfärbung c muss also jedem inzidenten Kantenpaar $e \neq e' \in E$ verschiedene Farben $c(e) \neq c(e')$ zuweisen
- Daher bildet jede Farbklasse

  $$E_i = \{e \in E : c(e) = i\}, \; i = 1, \ldots, k$$

  ein Matching von G, d.h. c zerlegt E in k disjunkte Matchings E_i
- Umgekehrt liefert jede Zerlegung von E in k disjunkte Matchings eine k-Kantenfärbung von G

Beispiel

- Für einen Kreis C_n der Länge n gilt

$$\chi'(C_n) = \begin{cases} 2, & n \text{ gerade,} \\ 3, & \text{sonst} \end{cases}$$

- Für den vollständigen Graphen K_n gilt

$$\chi'(K_n) = \begin{cases} n-1, & n \text{ gerade,} \\ n, & \text{sonst} \end{cases}$$

- Für diese Graphen gilt also $\Delta(G) \le \chi'(G) \le \Delta(G) + 1$

Kantenfärbungen

- Neben Graphen treten in manchen Anwendungen auch Multigraphen $G = (V, E)$ auf
- Diese können mehr als eine Kante zwischen zwei Knoten u und v haben, d.h. E ist eine Multimenge auf $\binom{V}{2}$
- Eine Multimenge A auf einer Grundmenge M lässt sich durch eine Funktion $v_A \colon M \to \mathbb{N}$ beschreiben, wobei $v_A(a)$ die Anzahl der Vorkommen des Elements a in A angibt
- Die Mächtigkeit von A ist $|A| = \sum_{a \in A} v_A(a)$
- Ist $G = (V, E)$ ein Multigraph, so gibt also die Funktion v_E für jedes Paar $\{u, v\} \in \binom{V}{2}$ die Anzahl $v_E(u, v)$ der Kanten zwischen den Endpunkten u und v in G an
- Dabei können wir einen Graphen $G = (V, E)$ auch als Multigraphen auffassen, der als Anzahlfunktion v_E die charakteristische Funktion $\chi_E \colon \binom{V}{2} \to \{0, 1\}$ von E verwendet (d.h. $\chi_E(e) = 1 \Leftrightarrow e \in E$)

- Wie bei Graphen gehen wir davon aus, dass jede Kante $e = \{u, v\}$ eines Multigraphen zwei verschiedene Endpunkte $u \neq v$ hat
- In einem Multigraphen $G = (V, E)$ gibt es genau $v_E(e) = v_E(u, v)$ Kanten zwischen zwei Knoten $u \neq v \in V$
- Die Zahl $v_E(e)$ wird auch als (Kanten-)Vielfachheit von e bezeichnet
- Ein wichtiger Parameter von G ist die maximale Kantenvielfachheit

  $$v(G) = \max_{e \in E} v_E(e),$$

  die auch als (Graph-)Vielfachheit von G bezeichnet wird
- Der Grad eines Knotens $u \in V$ ist $\deg_G(u) = \sum_{v \in N(u)} v_E(u, v)$ und der Maximalgrad von G ist wie üblich $\Delta(G) = \max_{u \in V} \deg_G(u)$

- Wir beschreiben eine k-Kantenfärbung für einen Multigraphen $G = (V, E)$ durch eine Funktion c, die jeder Kante $e \in \binom{V}{2}$ eine Menge $c(e) \subseteq \{1, \dots, k\}$ von $|c(e)| = v_E(e)$ Farben zuordnet, so dass $c(e) \cap c(e') = \emptyset$ für alle $e \neq e' \in \binom{V}{2}$ mit $e \cap e' \neq \emptyset$ gilt

Beispiel

- Für einen Multi-Kreis $C_{n,v}$ der Länge n und (uniformer) Vielfachheit v gilt

$$\chi'(C_{n,v}) = \begin{cases} 2v, & n \text{ gerade}, \\ ?, & \text{sonst} \end{cases}$$

Das Kantenfärbungsproblem für (Multi-)Graphen lässt sich leicht auf das Knotenfärbungsproblem für Graphen reduzieren

Definition Sei $G = (V, E)$ ein Multigraph mit $m \geq 1$ Kanten.

Der Kanten- oder Line-Graph von G ist der Graph $L(G) = (V', E')$ mit der Knotenmenge V', die $v_E(e)$ verschiedene Kopien $e^1, \ldots, e^{v_E(e)}$ jeder Kante $e \in \binom{V}{2}$ enthält, und der Kantenmenge $E' = \left\{\{e, e'\} \in \binom{V'}{2} : e \cap e' \neq \emptyset\right\}$

- $L(G)$ ist ein Graph, auch wenn G ein Multigraph ist
- Ist $G = (V, E)$ ein Graph, so hat der Line-Graph $L(G) = (V', E')$ die Knotenmenge $V' = E$

Die folgenden Beziehungen zwischen einem (Multi-)Graphen G und dem zugehörigen Line-Graphen $G' = L(G)$ lassen sich leicht verifizieren

Proposition

Für den Line-Graphen G' eines Multigraphen $G = (V, E)$ gilt:

1. $n(G') = m(G)$
2. $\chi(G') = \chi'(G)$
3. $\alpha(G') = \mu(G)$
4. $\omega(G') \geq \Delta(G)$
5. Für jede Kopie e^i einer Kante $e = \{u, v\}$ von G mit $v_E(e) \geq 1$ gilt
$$\deg_{G'}(e^i) = \deg_G(u) + \deg_G(v) - v_E(e) - 1$$
und somit ist $\Delta(G') \leq 2\Delta(G) - 2$

Damit erhalten wir aus den Abschätzungen

- $n(G')/\alpha(G') \leq \chi(G') \leq n(G') - \alpha(G') + 1$ und
- $\Delta(G) \leq \omega(G') \leq \chi(G') \leq \Delta(G') + 1 \leq 2\Delta(G) - 1$

für $G' = L(G)$ die folgenden Abschätzungen für $\chi'(G)$

Lemma

Für jeden Multigraphen G mit $m \geq 1$ Kanten gilt

- $m(G)/\mu(G) \leq \chi'(G) \leq m(G) - \mu(G) + 1$ und
- $\Delta(G) \leq \chi'(G) \leq 2\Delta(G) - 1$

Kantenfärbungen

Korollar

Für jeden regulären Multigraphen mit einer ungeraden Knotenzahl n und $m \geq 1$ Kanten gilt $\chi' \geq \Delta + 1 \geq 3$

Beweis.

- Wegen $\mu \leq (n-1)/2$ und $m = n\Delta/2$ folgt $\chi' \geq m/\mu \geq n\Delta/(n-1) > \Delta$
- Da n ungerade und $m \geq 1$ ist, folgt $\Delta \geq 2$ $\square$

Der Algorithmus von Vizing

- Wir geben nun einen effizienten Algorithmus zur Berechnung einer $(\Delta(G) + \nu(G))$-Kantenfärbung für einen beliebigen Multigraphen G an
- Hierfür benötigen wir folgende Begriffe für (Multi-)Graphen

Definition Sei $G = (V, E)$ ein Multigraph mit einer k-Kantenfärbung c

- Gelte $1 \leq i \neq j \leq k$ und sei $F \subseteq \{1, \ldots, k\}$
- Ein Nachbar v von u heißt F-Nachbar von u, wenn $c(u, v) \cap F \neq \emptyset$ ist
- Im Fall $F = \{i\}$ nennen wir v auch den i-Nachbarn von u
- Die Farbe i ist frei an einem Knoten $u \in V$ (kurz $i \in \mathit{free}(u)$), falls u keinen i-Nachbarn hat
- Der Multigraph $G_{ij} = (V, E_{ij})$ mit $\nu_{E_{ij}}(e) = |c(e) \cap \{i, j\}|$ heißt (i, j)-Subgraph von G
- Jede Komponente C von G_{ij} heißt (i, j)-Komponente von G
- Ist C ein Pfad oder ein Kreis, so nennen wir C auch einen (i, j)-Pfad bzw. (i, j)-Kreis in G (bzgl. c)

Der Algorithmus von Vizing

- Man sieht leicht, dass jede (i,j)-Komponente C eines Multigraphen G entweder
  - ein Pfad der Länge $\ell \geq 0$ oder
  - ein Kreis gerader Länge oder
  - ein Multigraph $C = (V', E')$ mit $V' = \{u, v\}$ und $v_{E'}(u,v) = 2$ ist
- Zudem können wir aus c eine k-Kantenfärbung c' von G erhalten, indem wir auf den Kanten von C die beiden Farben i und j vertauschen

Satz (Vizing 1964)

- Für jeden Graphen G mit $m \geq 1$ Kanten gilt
$$\chi'(G) \leq \min_{e \in E} \Delta(G - e) + 1 \leq \Delta(G) + 1$$
- Für jeden Multigraphen $G = (V, E)$ und jede Kante $e = \{y_0, y_1\} \in E$ gilt
$$\begin{aligned}\chi'(G) &\leq \max_{y \in V} \left(\deg_G(y) + v_{E-e}(y_0, y)\right) \\ &\leq \max_{y,y' \in V} \left(\deg_G(y) + v_E(y, y')\right) \leq \Delta(G) + v(G)\end{aligned}$$

Der Algorithmus von Vizing

Satz (Vizing 1964)

Für jeden Graphen G mit $m \geq 1$ Kanten gilt
$\chi'(G) \leq \min_{e \in E} \Delta(G - e) + 1 \leq \Delta(G) + 1$

Beweis Wir führen Induktion über die Kantenzahl m.

- Der Fall $m = 1$ (IA) ist klar
- Für den IS sei $G = (V, E)$ ein Graph mit $m \geq 2$ Kanten und sei $k = \min_{e \in E} \Delta(G - e) + 1$
- Wir wählen eine beliebige Kante $e_1 = \{y_0, y_1\} \in E$, so dass für den Graphen $G' = G - e_1 = (V, E')$ die Gleichung $k = \Delta(G') + 1$ gilt
- Dann hat G' wegen $\min_{e \in E'} \Delta(G' - e') + 1 \leq k$ nach IV eine k-Kantenfärbung $c\colon E' \to \{1, \ldots, k\}$ und da an jedem Knoten u mindestens

  $$k - \deg_{G'}(u) \geq k - \Delta(G') = 1$$

  Farben frei sind, folgt $\mathit{free}(u) \neq \emptyset$ für alle $u \in V$
- Betrachte nun folgende Prozeduren `expand` und `recolor`

Der Algorithmus von Vizing

Prozedur `expand`(G, c, y_0, y_1)

$F_0 := \mathit{free}(y_0)$; $\ell := 1$
while $\mathit{free}(y_\ell) \cap F_{\ell-1} = \emptyset$ **do**
  wähle $\alpha_\ell \in \mathit{free}(y_\ell)$ und bestimme den α_ℓ-Nachbar $y_{\ell+1}$ von y_0
  $F_\ell := F_{\ell-1} \cup \{\alpha_\ell\}$; $\ell := \ell + 1$
wähle $0 \leq i < \ell$ minimal mit $\mathit{free}(y_\ell) \cap F_i \neq \emptyset$
if $i = 0$ **then** // $\mathit{free}(y_\ell) \cap \mathit{free}(y_0) \neq \emptyset$
  wähle $\alpha_0 \in \mathit{free}(y_0) \cap \mathit{free}(y_\ell)$; $z := y_\ell$
else // $i \geq 1 \wedge \alpha_i \in \mathit{free}(y_\ell)$
  wähle $\alpha_0 \in \mathit{free}(y_0)$ und berechne den (α_0, α_i)-Pfad P mit Start-
  knoten y_ℓ und vertausche die beiden Farben α_0 und α_i entlang P
  sei z der Endknoten von P
case
  $z = y_0$: `recolor`(i, α_i)
  $z = y_i$: `recolor`(i, α_0)
else `recolor`(ℓ, α_0)

Der Algorithmus von Vizing

Prozedur `recolor`(i, α)

for $j := 1$ **to** $i - 1$ **do** $c(y_0, y_j) := \alpha_j$
$c(y_0, y_i) := \alpha$

Beweis (Fortsetzung)

Wir verifizieren, dass c nach dem Aufruf von `expand`(G, c, y_0, y_1) eine k-Kantenfärbung von G ist:

Fall 1 ($\mathbf{i = 0 \wedge z = y_\ell}$):

Wegen $\alpha_j \in \mathit{free}(y_j)$ kann `recolor` die Kanten $\{y_0, y_j\}$, $1 \le j < \ell$, mit α_j und wegen $\alpha_0 \in \mathit{free}(y_0) \cap \mathit{free}(y_\ell)$ die Kante $\{y_0, y_\ell\}$ mit α_0 färben

Fall 2 ($\mathbf{i > 0 \wedge z = y_0}$):

- P muss den Knoten $z = y_0$ über die Kante $\{y_0, y_{i+1}\}$ erreichen
- Da die Kante $\{y_0, y_{i+1}\}$ vor dem Vertauschen von α_0 und α_i auf dem Pfad P die Farbe α_i hat, hat sie danach die Farbe α_0
- Deshalb kann `recolor` die Kanten $\{y_0, y_j\}$, $1 \le j \le i$, mit α_j färben

Der Algorithmus von Vizing

Beweis (Schluss)

Fall 3 ($\mathbf{i > 0 \wedge z = y_i}$):

- Da $\alpha_i \in \mathit{free}(y_i) \cap \mathit{free}(y_\ell)$ ist, muss die mit y_i inzidente Endkanten des (y_ℓ, y_i)-Pfades P mit α_0 gefärbt sein
- Daher ist die Farbe α_0 nach Vertauschen von α_0 und α_i entlang P an beiden Knoten y_0 und y_i frei
- Folglich kann `recolor` die Kante $\{y_0, y_i\}$ mit α_0 und wegen $\alpha_j \in \mathit{free}(y_j)$ die Kanten $\{y_0, y_j\}$, $1 \leq j < i$, mit α_j färben

Fall 4 ($\mathbf{i > 0 \wedge z \notin \{y_0, y_i\}}$):

- Durch Vertauschen von α_0 und α_i auf P wird die Farbe α_0 an y_ℓ frei
- Wegen $z \notin \{y_0, y_i\}$ bleibt die Farbe α_j für $j \in \{0, i\}$ an y_j frei
- Da für alle $j \in \{1, \ldots, \ell - 1\} - \{i\}$ die Farbe α_j von α_0 und α_i verschieden ist, bleibt α_j sogar für alle $j = 0, \ldots, \ell - 1$ an y_j frei
- Daher kann `recolor` wie in Fall 1 die Kanten $\{y_0, y_j\}$, $1 \leq j < \ell$, mit α_j und die Kante $\{y_0, y_\ell\}$ mit α_0 färben $\square$

Der Algorithmus von Vizing

- Da sich die Prozedur `expand` mit Hilfe geeigneter Datenstrukturen in Zeit $\mathcal{O}(n)$ implementieren lässt und diese Prozedur für jede Kante einmal aufgerufen wird, ergibt sich eine Gesamtlaufzeit von $\mathcal{O}(mn)$
- Wegen $\Delta(G) \leq \chi'(G) \leq \Delta(G) + 1$ kann $\chi'(G)$ nur einen der beiden Werte $\Delta(G)$ oder $\Delta(G) + 1$ annehmen
- Graphen G mit $\chi'(G) = \Delta(G)$ heißen Klasse 1 und Graphen G mit $\chi'(G) = \Delta(G) + 1$ heißen Klasse 2
- Neben den vollständigen Graphen K_n mit gerader Knotenzahl n sind alle bipartiten Graphen Klasse 1
- Zudem sind alle planaren Graphen G mit $\Delta(G) \geq 7$ Klasse 1
- Für $2 \leq d \leq 5$ existieren planare Graphen G mit $\Delta(G) = d$, die Klasse 2 sind; für $d = 6$ ist dies offen
- Das Problem, für einen gegebenen Graphen G mit $\Delta(G) \leq 3$ zu entscheiden, ob er Klasse 1 ist, ist NP-vollständig

Der Algorithmus von Vizing

Der Algorithmus von Vizing lässt sich leicht auf Multigraphen ausdehnen

Satz (Vizing 1964)

Für jeden Multigraphen $G = (V, E)$ und jede Kante $e = \{y_0, y_1\} \in E$ gilt

$$\begin{aligned} \chi'(G) &\leq \max_{y \in V} \left(\deg_G(y) + v_{E-e}(y_0, y) \right) \\ &\leq \max_{y, y' \in V} \left(\deg_G(y) + v_E(y, y') \right) \leq \Delta(G) + v(G) \end{aligned}$$

Beweis.

- Der Färbungsalgorithmus für Multigraphen arbeitet vollkommen analog zum Algorithmus für einfache Graphen auf den Folien 118 und 119
- Wir müssen nur die Prozeduren `expand` und `recolor` entsprechend anpassen und die Anzahl k der Farben groß genug wählen, damit bei jeder Wahl einer Farbe α_ℓ aus der Menge $\mathit{free}(y_\ell)$ noch eine Farbe zur Verfügung steht, die nicht schon früher gewählt wurde

Prozedur $\mathtt{expand}(G, c, y_0, y_1)$

$F_0 := \mathit{free}(y_0)$; $\ell := 1$
while $\mathit{free}(y_\ell) \cap F_{\ell-1} = \emptyset$ **do**
  wähle $\alpha_\ell \in \mathit{free}(y_\ell)$ und sei $y_{\ell+1}$ der α_ℓ-Nachbar von y_0
  $F_\ell := F_{\ell-1} \cup \{\alpha_\ell\}$; $\ell := \ell + 1$

wähle $0 \leq i < \ell$ minimal mit $\mathit{free}(y_\ell) \cap F_i \neq \emptyset$
if $i = 0$ **then** // $\mathit{free}(y_\ell) \cap \mathit{free}(y_0) \neq \emptyset$
  wähle $\alpha_0 \in \mathit{free}(y_0) \cap \mathit{free}(y_\ell)$; $z := y_\ell$
else // $i \geq 1 \wedge \alpha_i \in \mathit{free}(y_\ell)$
  wähle $\alpha_0 \in \mathit{free}(y_0)$ und berechne den (α_0, α_i)-Pfad P mit Start-
  knoten y_ℓ und vertausche die beiden Farben α_0 und α_i entlang P
  sei z der Endknoten von P
case
  $z = y_0$: $\mathtt{recolor}(i, \alpha_i)$
  $z = y_i$: $\mathtt{recolor}(i, \alpha_0)$
else $\mathtt{recolor}(\ell, \alpha_0)$

Prozedur `multiexpand`(G, c, y_0, y_1)

$F_0 := \mathit{free}(y_0)$; $\ell := 1$; $Y := \{y_1\}$; $\mathit{used}(y_1) := \emptyset$
while $\mathit{free}(y_\ell) \cap F_{\ell-1} = \emptyset$ **do**
  wähle $\alpha_\ell \in \mathit{free}(y_\ell) \setminus \mathit{used}(y_\ell)$ und sei $y_{\ell+1}$ der α_ℓ-Nachbar von y_0
  $F_\ell := F_{\ell-1} \cup \{\alpha_\ell\}$; $\ell := \ell + 1$; $\mathit{used}(y_\ell) := \mathit{used}(y_\ell) \cup \{\alpha_\ell\}$;
  if $y_{\ell+1} \notin Y$ **then** $Y := Y \cup \{y_{\ell+1}\}$; $\mathit{used}(y_{\ell+1}) := \emptyset$
wähle $0 \le i < \ell$ minimal mit $\mathit{free}(y_\ell) \cap F_i \neq \emptyset$
if $i = 0$ **then** // $\mathit{free}(y_\ell) \cap \mathit{free}(y_0) \neq \emptyset$
  wähle $\alpha_0 \in \mathit{free}(y_0) \cap \mathit{free}(y_\ell)$; $z := y_\ell$
else // $i \ge 1 \wedge \alpha_i \in \mathit{free}(y_\ell)$
  wähle $\alpha_0 \in \mathit{free}(y_0)$ und berechne den (α_0, α_i)-Pfad P mit Start-
  knoten y_ℓ und vertausche die beiden Farben α_0 und α_i entlang P
  sei z der Endknoten von P
case
  $z = y_0$: `multirecolor`(i, α_i)
  $z = y_i$: `multirecolor`(i, α_0)
else `multirecolor`(ℓ, α_0)

Der Algorithmus von Vizing

Prozedur $\texttt{recolor}(i, \alpha)$

---

for $j := 1$ **to** $i - 1$ **do** $c(y_0, y_j) := \alpha_j$

$c(y_0, y_i) := \alpha$

Der Algorithmus von Vizing

Prozedur `multirecolor`(i, α)

---

for $j := 1$ **to** $i-1$ **do** $c(y_0, y_j) := \{\alpha_j\} \cup c(y_0, y_j) \setminus \{\alpha_{j-1}\}$

$c(y_0, y_i) := \{\alpha\} \cup c(y_0, y_i) \setminus \{\alpha_{i-1}\})$

Beweis (Fortsetzung)

- Folgende Tabelle zeigt die Anzahl k der Farben, die benötigt werden, um der Prozedur `multiexpand` in den Zeilen 3, 8, und 10 die Wahl einer unbenutzten freien Farbe α_ℓ aus der Menge $\mathit{free}(y_\ell)$ zu ermöglichen und so eine Farbe für die neue Kante $e = \{y_0, y_1\}$ zu finden

Knoten	benötigte Farbenzahl k
y_0	$\deg_{G-e}(y_0) + 1 = \deg_G(y_0)$
y_1	$\deg_{G-e}(y_1) + v_{E-e}(y_0, y_1) + 1 = \deg_G(y_1) + v_{E-e}(y_0, y_1)$
$y_\ell, \ell \geq 2$	$\deg_{G-e}(y_\ell) + v_{E-e}(y_0, y_\ell) = \deg_G(y_\ell) + v_{E-e}(y_0, y_\ell)$

- Also reichen $k = \max_{y \in V} \left(\deg_G(y) + v_{E-e}(y_0, y) \right)$ Farben $\square$

Der Algorithmus von Vizing

- Man beachte, dass für jeden Multigraphen die Ungleichungen $\Delta(G) \leq \chi'(G) \leq \Delta(G) + v(G) \leq 2\Delta(G)$ gelten
- In den Übungen leiten wir noch folgende Schranken aus dem Satz von Vising bzw. aus seinem Beweis ab

Korollar

1. Für jeden Multigraphen G gilt

   $$\chi'(G) \leq 3\Delta(G)/2 \text{ (Satz von Shannon)}$$

2. Für jeden bipartiten Multigraphen G (d.h. $\chi(G) \leq 2$) gilt

   $$\chi'(G) = \Delta(G) \text{ (Satz von König)}$$