

Vorlesung

Graphen und Algorithmen

Prof. Dr. Matthias Kriesell

Inhaltsverzeichnis

1. Bäume	3
1.1. Breiten- und Tiefensuchbäume	3
1.1.1. Breitensuche	8
1.1.2. Tiefensuche	10
1.2. Bäume kleinsten Gewichtes und Matroide	12
1.3. Das Traveling-Salesman-Problem	15
1.4. Der Satz von Courcelle	18
2. Matchings	20
2.1. Matchings in bipartiten Graphen	20
2.2. Faktorsätze	23
3. Flüsse	30
3.1. Das Max-Flow-Min-Cut-Theorem	30
A. Lektüre	34
B. Sätze, die in der Übung bewiesen werden	35
Stichwortverzeichnis	36

1. Bäume

1.1. Breiten- und Tiefensuchbäume

1.1. Ein (einfacher, ungerichteter) **Graph** ist ein paar $G = (V, E)$, bestehend aus einer Menge V von **Ecken** (engl. *vertices*) und einer Menge $E \subseteq \{\{x, y\} \mid x \neq y \text{ aus } V\}$ von **Kanten** (engl. *edge*).

$$V(G) := V \quad (1.1)$$

$$E(G) := E \quad (1.2)$$

Für Kanten wird auch üblicherweise die Schreibweise xy anstelle von $\{x, y\}$ benutzt. Also ist auch $xy = yx$. In alter Literatur wurde teilweise auch $x \in G$ und $e \in G$ anstelle von $x \in V$ und $e \in E$ benutzt. Diese Schreibweise wird nicht mehr benutzt!

In dieser Definition wurden nur einfache ungerichtete Graphen definiert. Es gibt jedoch auch Definitionen für gerichtete Graphen (also $xy \neq yx$) und Graphen mit Mehrfachkanten (Graphen mit mehreren unterschiedlichen Kanten xy). Es kommt bei einfachen, ungerichteten Graphen also nur darauf an, wie die Knoten miteinander verbunden sind, nicht wie häufig oder in welche Richtung! Insbesondere ist es auch egal, wie man einen Graphen zeichnet, solange die Anzahl Knoten und die Verbindungen zwischen den Knoten passen. Als Beispiel seien hier zwei Zeichnungen des PETERSEN-Graphs in [Abbildung 1.1](#) gezeigt.

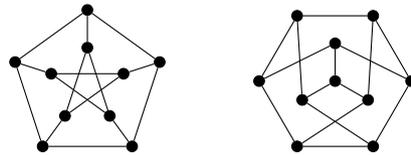


Abbildung 1.1.: Zwei Varianten des gleichen Graphen (PETERSEN-Graph)

1.2. Für $xy \in E$ sind $x, y \in V$ **benachbart** bzw. x ist **Nachbar** von y .

Alle benachbarten Graphen seien **endlich** (G endlich $\iff V$ (und E) sind endlich).

1.3. Ein Graph H heißt **Teilgraph** des Graphen G (in Zeichen: $H \leq G$), falls $V(H) \subseteq V(G)$ und $E(H) \subseteq E(G)$.

Für Teilgraphen ist zu beachten, dass nicht jede Kombination aus $V' \subseteq V(G)$ und $E' \subseteq E(G)$ ein Graph ist. Werden beispielsweise Kanten in E' aufgenommen, die Knoten enthalten, welche nicht in V' enthalten sind, ist das resultierende Paar $H = (V', E')$ *kein* Graph und damit auch kein Teilgraph von G . Man kann jedoch Teilgraphen basierend auf einer Teilmenge von $V(G)$ konstruieren.

1.4. Für $X \subseteq V(G)$ sei

$$G[X] := (X, \{xy \in E(G) \mid x, y \in X\}) \quad (1.3)$$

Ein solcher Graph $G[X]$ heißt dann der von X **induzierte** Teilgraph.

Für $F \subseteq E(G)$ sei

$$G[F] := (V(G), F) \quad (1.4)$$

der von F **induzierte** Teilgraph.

Es gilt $G[X] \leq G$ und $G[F] \leq G$.

Beispiele für induzierte Graphen sind in [Abbildung 1.2](#) und [Abbildung 1.3](#) gezeigt. Während der aus Knoten induzierte Teilgraph nur eine Teilmenge aus Knoten im Teilgraph hat, sind in einem aus Kanten induzierten Teilgraphen stets alle Knoten aus dem ursprünglichen Graphen enthalten, auch, wenn sie mit keiner Kante des Teilgraphen verbunden sind.

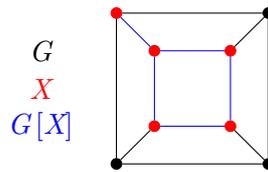


Abbildung 1.2.: Aus Knoten induzierter Teilgraph

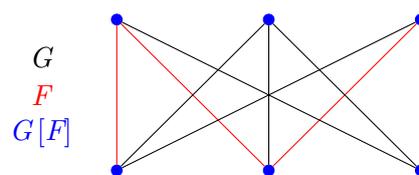


Abbildung 1.3.: Aus Kanten induzierter Teilgraph

1.5. Für $X \subseteq V(G)$ entsteht

$$G - X := (V(G) \setminus X, \{xy \in E \mid x, y \in V(G) \setminus X\}) \quad (1.5)$$

$$= G[V(G) \setminus X] \quad (1.6)$$

durch **Löschung** der Ecken in X aus G .

Für $F \subseteq E(G)$ entsteht

$$G - F := (V(G), E(G) \setminus F) \quad (1.7)$$

$$= G[E(G) \setminus F] \quad (1.8)$$

durch **Löschung** der Kanten in F aus G .

Besteht X aus nur einer Ecke x , so wird auch die Notation $G - x := G - \{x\}$ benutzt. Analog wird $G - e := G - \{e\}$ für $F = \{e\}$ benutzt. Es gilt wieder $G - X \leq G$ und $G - F \leq G$, also auch $G - x \leq G$ und $G - e \leq G$.

Als nächstes beschäftigen wir uns mit der Definition von Zusammenhang.

1.6. Für zwei Ecken $a, b \in E(G)$ heißt eine nichtleere Folge $P = x_0, \dots, x_l$ von paarweise verschiedenen Ecken x_j mit

$$x_0 = a \quad (1.9)$$

$$x_l = b \quad (1.10)$$

$$\forall i \in \{1, \dots, l\} : x_{i-1}x_i \in E(G) \quad (1.11)$$

ein **Weg** von a nach B der **Länge** l , auch a, b -**Weg** genannt.

Die Ecken a, b dieses Weges heißen **Endecken**, alle anderen sind **innere Ecken**.

Besonders zu beachten ist hierbei, dass die Länge des Weges 1 kleiner ist als die Anzahl Knoten in P ! Für den Weg können auch Kanten- und Knotenmengen definiert werden:

1.7.

$$V(P) := \{x_0, \dots, x_l\} \quad (1.12)$$

$$E(P) := \{x_{i-1}x_i : i \in \{1, \dots, l\}\} \quad (1.13)$$

$(V(P), E(P))$ ist auch wieder ein Teilgraph und wird manchmal auch a, b -Weg (also wie oben) genannt. Diese Teilgraphen sind immer kreisfrei, da gemäß Definition die Ecken $x_j \in V(P)$ paarweise verschieden sind. Insbesondere sind auch einzelne Kanten oder auch einzelne Knoten Wege.

Aus der Teilgraphendarstellung eines Weges lässt sich die Folgendarstellung (bis auf die Durchlaufrihtung) zurückgewinnen.

1.8. Mit $a \sim_G b \iff \exists a, b\text{-Weg in } G$ wird eine Äquivalenzrelation auf $V(G)$ definiert. Die Äquivalenzklassen heißen **Zusammenhangskomponenten** oder **Komponenten**. Auch die von den Komponenten induzierten Teilgraphen heißen **Komponenten**.

Diese Relation ist eine Äquivalenzrelation, da sie reflexiv, symmetrisch und transitiv ist. Die Reflexivität und Symmetrie sind offensichtlich erfüllt, da der Graph ungerichtet ist und für jeden a, b -Weg auch immer ein b, a -Weg existiert. Für die Transitivität kann man einen a, b -Weg und b, c -Weg im Graphen betrachten. Um einen a, c -Weg zu finden, verfolge man den a, b -Weg, bis man auf einen Knoten stößt, der auf dem b, c -Weg liegt. Verfolgt man ab hier den b, c -Weg, hat man einen a, c -Weg gefunden. [Abbildung 1.4](#) zeigt solche a, b - und b, c -Wege im Graphen.

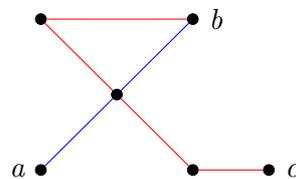


Abbildung 1.4.: a, b -Weg und b, c -Weg in einem Graphen

1.9. Ein Graph mit höchstens einer Komponente heißt **zusammenhängend**. Ist ein Graph nicht zusammenhängend, so heißt er **unzusammenhängend**.

Als abkürzende Schreibweisen werden in dieser Vorlesung auch *zshgd* und *zh* für „zusammenhängend“ benutzt.

1.10. Ein Graph T heißt **Baum**, falls T zusammenhängend ist und $T - e \forall e \in E(T)$ unzusammenhängend ist. T ist also **minimal zusammenhängend**.

1.11. Ein **Wald** ist ein Graph, dessen Komponenten Bäume sind. Ein zusammenhängender Wald ist ein Baum.

Verfahren 1.1: Sei $x_0 \in V(G)$. (*) Wenn es unter den bereits gewählten Ecken x_0, x_1, \dots, x_l eine Ecke x_t gibt, die einen Nachbarn $y \in V(G) \setminus \{x_0, \dots, x_l\}$ besitzt, dann setze $x_{l+1} = y$ und $f(l+1) := t$ und iteriere (*).

Satz 1.1. *Verfahren 1.1 endet mit einem Teilbaum.*

$$T := (\{x_0, \dots, x_l\}, \{x_t x_{f(t)} \mid t \in \{1, \dots, l\}\}). \quad (1.14)$$

Dabei ist $\{x_0, \dots, x_l\} = V(T)$ die Komponente von G , die x_0 enthält.

Ein Beispiel dafür ist [Abbildung 1.5](#). Das Verfahren ist nicht deterministisch, da in jedem Schritt ein Nachbar eines beliebigen Knotens aus der Menge gewählt werden kann.

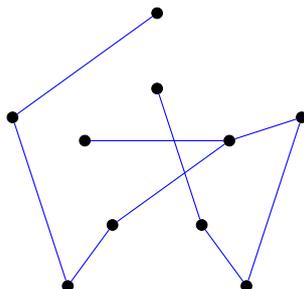


Abbildung 1.5.: Erzeugung eines Baums aus einem PETERSEN-Graph

Bemerkung: Für $t \in \{0, \dots, l\}$ liefert $x_t, x_{f(t)}, x_{f(f(t))}, \dots, x_0$ den x_t, x_0 -Weg in T .

Beweis. Das Verfahren endet, weil in jedem Schritt die Folge x_0, \dots, x_n um eine noch nicht dort enthaltene Ecke x_{l-1} vergrößert wird und $V(G)$ endlich ist.

Sei $T_s = (\{x_0, \dots, x_s\}, \{x_t x_{f(t)} \mid t \in \{1, \dots, s\}\})$. Zeige induktiv über s , dass T_s ein Teilbaum von G ist.

Für $s = 0$: $T_0 = (\{x_0\}, \emptyset)$ ist Teilbaum. ✓

Sei nun T_0, \dots, T_s Teilbaum von G . **z:** Auch T_{s+1} ist ein Teilbaum von G .

Nach IV gibt es in T_j für jedes $j \in \{0, \dots, s\}$ einen x_j, x_0 -Weg, also gilt $x_j \sim_G$ lücke Wegen $x_{s+1} x_{f(s+1)} \in E(T_{s+1})$ gilt auch $x_{s+1} \sim x_{f(s+1)}$, also

$$\forall j \in \{0, \dots, s\} : x_{s+1} \sim_{T_{s+1}} x_0 \quad (1.15)$$

$$\iff \forall i, j \in \{0, \dots, s+1\} : x_i \sim_{T_{s+1}} x_j \quad (1.16)$$

$$\implies T_{s+1} \text{ zusammenhängend} \quad (1.17)$$

$$\text{z} : T_{s+1} - e \text{ unzusammenhängend } \forall e \in E(T_{s+1}) \quad (1.18)$$

Für $e = x_{s+1} x_{f(s+1)}$ ist x_{s+1} zu keiner Kante in $T_{s+1} - e$ inzident; insbesondere gibt es keinen x_{s+1}, x_0 -Weg in $T_{s+1} - e$. Für $e \in E(T_{s+1}) \setminus \{x_{s+1} x_{f(s+1)}\} = E(T_s)$ ist nach IV zumindest $T_s - e$ unzusammenhängend. Daher existieren $a, b \in V(T_s - e) = V(T_s)$ so, dass kein a, b -Weg in T_s existiert.

Gäbe es einen a, b -Weg P in $T_{s+1} - e$, so enthält P die Kante $x_{s+1} x_{f(s+1)}$ oder die Ecke x_{s+1} . In beiden Fällen enthält P die Ecke x_{s+1} und zwar als *innere Ecke*. Aber x_{s+1} ist in T_{s+1} mit nur einer Kante inzident (nämlich $x_{s+1} x_{f(s+1)}$) und kann daher nicht innere Ecke eines Weges in T_{s+1} sein. ↯

Folglich gibt es keinen solchen a, b -Weg in T_{s+1} . Entsprechend ist $T_{s+1} - e$ unzusammenhängend. Somit ist T_{s+1} ein Baum.

Sei H (bzw. $V(H)$) die Komponente von G , die x_0 enthält. \underline{z} : $V(T) = V(H)$.

T ist zusammenhängender Teilgraph von G , der x_0 enthält. Daher $V(T) \subseteq V(H)$.

\underline{z} : $V(T) \not\subseteq V(H)$.

Andernfalls gäbe es ein $z \in V(H) \setminus V(T)$. Da H zusammenhängend ist und $x_0 \in V(H)$ existiert ein z, x_0 -Weg Q in H .

$$Q = \underbrace{y_0}_{=z}, y_1, \dots, y_k \underbrace{y_k}_{=x_0} \quad (1.19)$$

Daher existiert $j \in \{0, \dots, k\}$ mit $y_j \in V(T)$ und $y_{j+1} \notin V(T)$. Also ist $y_{j+1} = x_t$ für $t \in \{0, \dots, l\}$. Also existiert nach Abbruch der Iteration von [Verfahren 1.1](#) mit x_0, \dots, x_l eine Ecke $y = y_{j+1} \in V(G) \setminus \{x_0, \dots, x_l\}$ mit einem Nachbarn $x_t, t \in \{0, \dots, l\}$. $\not\downarrow$

Somit ist $V(H) = V(T)$. \square

1.12. Ein Teilgraph H von G heißt **aufspannend**, falls $V(H) = V(G)$. Ein Teilbaum von G heißt **Spannbaum**, falls er aufspannend ist.

[Verfahren 1.1](#) liefert also einen Spannbaum derjenigen Komponente von G , die x_0 enthält. Die gezielte Auswahl des ältesten oder jüngsten gesehenen Nachbarn in [Verfahren 1.1](#) führt zu **Breitensuche** bzw. **Tiefensuche**.

1.1.1. Breitensuche

1.13. Die Länge eines kürzesten a, b -Weges im Graphen G heißt **Abstand** von a, b . Bezeichnung: $d_G(a, b)$.

Gibt es keinen solchen Weg, so setze $d_G(a, b) = \infty$.

Also $\min\{|E(P)| : P \text{ ein } a, b\text{-Weg}\} \cup \{\infty\} = d_G(a, b)$. Als Beispiel sei hier [Abbildung 1.6](#) gezeigt. Indem man irgendeinen Pfad im Graphen sucht, kann man zwischen zwei Ecken a, b immer eine obere Schranke für $d_G(a, b)$ finden. Durch systematisches Aufspannen des Graphen mit Breitensuche kann man den Abstand aller Ecken zu a im Graphen und damit auch $d_G(a, b)$ bestimmen.

zeichnung
1.5

Satz 1.2. $d_G : V(G) \times V(G) \rightarrow \mathbb{N}$ ist für zusammenhängende G eine **Metrik**, also

$$(1) \quad d_G(a, b) \geq 0 \text{ und } d_G(a, b) = 0 \iff a = b$$

$$(2) \quad d_G(a, b) = d_G(b, a) \forall a, b \in V(G)$$

$$(3) \quad d_G(a, b) + d_G(b, c) \leq d_G(a, c) \forall a, b, c \in V(G)$$

Beweis. Siehe Übung. \square

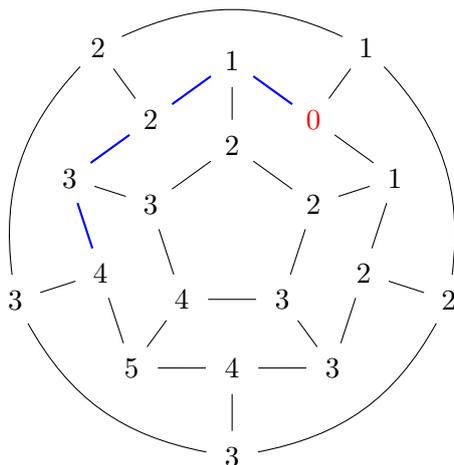


Abbildung 1.6.: Weg und Abstände zwischen zwei Punkten in einem Dodekaeder

1.14. Sei G zusammenhängend und x_0 eine Ecke von G . Ein Spannbaum T von G heißt **Breitensuchbaum** (BFS-Tree, Breadth-First-Search-Tree) bei x_0 , falls $d_T(z, x_0) = d_G(z, x_0) \forall z \in V(G)$.

Verfahren 1.2 (Breitensuche): (*) Wenn es unter den bereits gewählten Ecken x_0, \dots, x_l eine Ecke x_t gibt, die einen Nachbarn $y \in V(G) \setminus \{x_0, \dots, x_l\}$ hat, wähle x_t so, dass t kleinstmöglich ist. Iteriere (*).

Verfahren 1.2 erzwingt, dass wir immer die Nachbarn in den Baum aufnehmen, die wir schon am längsten kennen.

Satz 1.3 (Breitensuche). *Verfahren 1.2* endet mit einem Breitensuchbaum

$$T = (\{x_0, \dots, x_l\}, \{x_t x_{f(t)} \mid t \in \{1, \dots, l\}\}) \quad (1.20)$$

von G bei der Ecke x_0 .

Beweis. Nach **Satz 1.1** endet das Verfahren mit einem Spannbaum T von G . Offenbar gilt: $d_G(x_0, x_0) = d_T(x_0, x_0)$.

\underline{z} : Induktiv über s :

$$d_G(x_s, x_0) = d_T(x_s, x_0) \geq d_T(x_{s-1}, x_0) \text{ für } s > 0. \quad (1.21)$$

Die Behauptung (1.21) gilt für $s = 0$. Sei nun die Behauptung „bis“ s bewiesen (IV).

\underline{z} : (1.21) gilt für $s + 1$ anstelle von s .

Sei P ein kürzester x_{s+1}, x_0 -Weg in G und sei x_j die erste Ecke in der Folgenderstellung von P mit $j \leq s$. Nach Wahl von $f(s + 1)$ gibt es kein x_t mit $t < f(s + 1)$, die einen

Nachbarn außerhalb in $V(G) \setminus \{x_0, \dots, x_s\}$. Daher ist $j \geq f(s+1)$. Daher gilt:

$$d_T(x_{s+1}, x_0) \stackrel{T \leq G}{\geq} d_G(x_{s+1}, x_0) \quad (1.22)$$

$$\geq 1 + d_G(x_j, x_0) \quad \text{da } x_j \text{ nicht die erste Ecke auf } P \quad (1.23)$$

$$\geq 1 + d_G(x_{f(s+1)}, x_0) \quad \text{Monotonie von } (d_G(x_0, x_0), d_G(x_1, x_0), \dots) \quad (1.24)$$

$$\stackrel{\text{(IV)}}{=} 1 + d_T(x_{f(s+1)}, x_0) \quad (1.25)$$

$$= d_T(x_{s+1}, x_0) \quad (1.26)$$

Also gilt überall Gleichheit, insbesondere:

$$d_T(x_{s+1}, x_0) = d_G(x_{s+1}, x_0) \quad (1.27)$$

Für $s = 0$ gilt sowieso:

$$d_T(x_{s+1}, x_0) \geq 0 = d_T(x_s, x_0) \quad (1.28)$$

Für $s > 0$: Nach Wahl von $f(s)$ gibt es kein x_t mit $t < f(s)$, das einen Nachbarn außerhalb von x_0, \dots, x_{s-1} hat. Daher gilt $f(s+1) \geq f(s)$. Die Monotonieeigenschaft in der Induktionsvoraussetzung zeigt:

$$d_T(x_{s+1}, x_0) = d_T(x_{f(s+1)}, x_0) + 1 \quad (1.29)$$

$$\geq d_T(x_{f(s)}, x_0) + 1 \quad \text{da } f(s+1) \geq f(s) \wedge f(s+1) < s+1 \quad (1.30)$$

$$= d_T(x_s, x_0). \quad (1.31)$$

Dies zeigt (1.21). Also ist T Breitensuchbaum. \square

1.1.2. Tiefensuche

1.15. Ein Spannbaum T eines zusammenhängenden Graphen G heißt ein **Tiefensuchbaum** bei x_0 ($x_0 \in V(G)$), auch **DFS-Tree** (Depth-First-Search), falls für jedes $xy \in E(G)$ die Ecke y in dem x, x_0 -Weg von T vorkommt oder die Ecke x in dem y, x_0 -Weg von T vorkommt.

Verfahren 1.3 (Tiefensuche): Sei x_0 Ecke des zusammenhängenden Graphen G .

(*) Wenn es unter den bereits gewählten Ecken x_0, \dots, x_l eine Ecke x_t gibt, die einen Nachbarn $y \in V(G) \setminus \{x_0, \dots, x_l\}$ hat, wähle x_t so, dass t größtmöglich ist. Iteriere (*).

Das Verfahren wählt zunächst immer den längstmöglichen Weg. Am Anfang wird ein sehr langer Weg durch den Graphen gewählt, bis man sich nicht mehr weiter von der Wurzel entfernen kann. Im Beispiel des PETERSEN-Graph (Abbildung 1.7) reicht dies sogar so weit, ein Tiefensuchbaum völlig unverzweigt ist.

Ist dies nicht der Fall, geht man zunächst so weit wie möglich. Danach geht man so wenig wie möglich wieder zurück, bis man eine Ecke findet, die noch nicht in $V(T)$ ist.

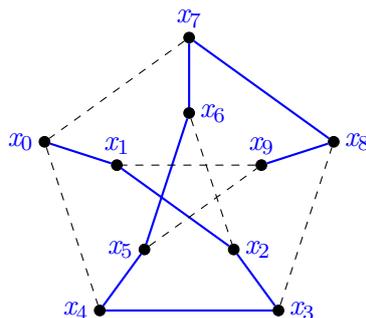


Abbildung 1.7.: Tiefensuche in einem PETERSEN-Graph

Jeden dieser Pfade, die sich ohne Backtracking finden lassen, können als Segmente im Baum betrachtet werden. Sobald man in einem solchen Segment einmal Backtracking gemacht hat, kann man keine Ecken mehr finden, die tiefer in diesem Segment Anschluss haben. Dies wird später im Beweis gezeigt.

Satz 1.4 (Tiefensuche). *Das Verfahren endet mit einem Tiefensuchbaum*

$$T = (\{x_0, \dots, x_l\}, \{x_t x_{f(t)} \mid t \in \{1, \dots, l\}\}) \quad (1.32)$$

von G bei der Ecke x_0 .

Beweis. Durch Satz 1.1 endet das Verfahren mit einem Spannbaum T von G .

\mathcal{Z} : Induktiv über s :

- Unter den Ecken x_0, \dots, x_s haben nur die Ecken des x_s, x_0 -Weges in T Nachbarn in $V(G) \setminus \{x_0, \dots, x_s\}$.
- Für $s > 0$ sind unter den Ecken x_0, \dots, x_{s-1} nur die Ecken des $x_{f(s)}, x_0$ -Weges in T zu x_s benachbart.

Für $s = 0$ gelten (a) und (b) offensichtlich. Nehmen wir nun an, dass die beiden Aussagen „bis“ s gelten (IV).

\mathcal{Z} : (a) und (b) gelten sinngemäß für $s + 1$ anstelle von s .

Nach IV (a) liegt $x_{f(s+1)}$ auf dem x_s, x_0 -Weg. Sei nun $x_j, j \leq s + 1$ zu einer Ecke $y \neq x_j$ in G außerhalb von x_0, \dots, x_s benachbart. Für $j \leq s$ liegt x_j auf dem x_s, x_0 -Weg in T (IV). Nach Wahl von x_{s+1} kann kein x_t mit $t > f(s + 1), t \leq s$, Nachbarn in G außerhalb von x_0, \dots, x_s haben. Folglich gilt $j \leq f(s + 1)$.

Die Folge der Indizes $s, f(s), f(f(s)), \dots, 0$ entlang des x_s, x_0 -Weges in T ist absteigend. Also liegt x_j auf dem $x_{f(s+1)}, x_0$ -Weg in T . Damit ist der Induktionsschluss für (b) erbracht und x_j liegt auf dem x_{s+1}, x_0 -Weg in T . Für $j = s + 1$ gilt letzteres trivialerweise! Damit ist der Induktionsschluss für (a) erbracht.

Aus (b) folgt: für $x_s x_j \in E(G)$ mit $j < s$: x_j liegt auf dem x_s, x_0 -Weg. Somit gilt $\forall xy \in E(G) : y$ liegt auf dem x, x_0 -Weg oder x liegt auf dem y, x_0 -Weg. \square

Grafik
Segmente
Tiefensuche zum
Beweis

1.16. Ist T ein Baum und $x_0 \in V(T)$, so wird durch

$$y \leq z : \iff y \text{ liegt auf dem } x_0, z\text{-Weg in } T \quad (1.33)$$

eine Ordnung auf $V(T)$ definiert.

Folglich ist ein Spannbaum T des Graphen G genau dann ein Tiefensuchbaum von G bei x_0 , wenn die Endecken jeder Kante von G bzgl \leq vergleichbar sind.

Name für def.

1.2. Bäume kleinsten Gewichtes und Matroide

Als nächstes betrachten wir gewichtete Graphen. Diese können in der Realität bspw. Kosten (bspw. Fahrzeit) entsprechen.

1.17. Sei G ein Graph. Ein **Kreis** der **Länge** $l \geq 3$ in G ist eine Folge $C = x_0, \dots, x_{l-1}, x_0$ derart, dass x_0, \dots, x_{l-1} ein Weg ist und $x_{l-1}x_0 \in E(G)$. Setze

$$V(C) = \{x_0, \dots, x_{l-1}\} \subseteq V(G) \quad (1.34)$$

$$E(C) = \{x_0x_1, x_1x_2, \dots, x_{l-2}x_{l-1}, x_{l-1}x_0\} \subseteq E(G) \quad (1.35)$$

Auch der Teilgraph $(V(C), E(C))$ heißt **Kreis**.

Aus dem Teilgraphen lässt sich die Folgendarstellung bis auf Startecke und Durchlaufrichtung vollständig rekonstruieren. Insgesamt gibt es $2l$ mögliche Folgen, die sich aus dem Teilgraphen rekonstruieren lassen (eine Folge pro Ecke und Durchlaufrichtung).

Zudem ist ein Graph ohne Kreise ein **Wald** und ein zusammenhängender Graph ohne Kreise ein **Baum**.

1.18. Für einen Graphen G heißt $F \subseteq E(G)$ **kreisfrei**, falls $G[F] = (V(G), F)$ keinen Kreis enthält (also ein Wald ist).

Übung: Für einen nichtleeren Wald G mit c Komponenten gilt:

$$|E(G)| = |V(G)| - c \quad (1.36)$$

Für einen Baum mit einer Ecke ist dies trivialerweise wahr. Für größere Bäume kann man induktiv immer eine Blatt finden, was man löschen kann. Für diesen verkleinerten Baum gilt die Aussage dann gemäß IV. Durch erneutes Anhängen des Blattes kann man dann zeigen, dass die Gleichung gilt. Wenn man eine Ecke löscht, die kein Blatt ist, ist der Graph nicht mehr zusammenhängend und damit wächst c um 1. Das gleicht die Verringerung der Anzahl Kanten wieder aus.

Satz 1.5 (Austauschlemma für Graphen). *Sind F, F' zwei kreisfreie Kantenmengen des Graphen G mit $|F| < |F'|$, so gibt es eine Kante $e \in F' \setminus F$ mit $F \cup \{e\}$ kreisfrei.*

Beweis. Seien c, c' die Anzahlen der Komponenten von $G[F]$ bzw. $G[F']$. Wäre die Eckenmenge jeder Komponente von $G[F']$ ganz in der Eckenmenge einer Komponente von $G[F]$ enthalten, so folgt $c' > c$. Es gilt aber

$$c' = |V(G[F'])| - |E(G[F'])| \quad (1.37)$$

$$= |V(G)| - \underbrace{|F'|}_{> |F|} \quad (1.38)$$

$$< |V(G)| - |F| \quad (1.39)$$

$$= |V(G[F])| - |E(G[F])| \quad (1.40)$$

$$= c \quad (1.41)$$

Also gibt es eine Komponente von $G[F]$ mit Endpunkten in mindestens zwei Komponenten von $G[F']$. Also gibt es eine Kante $e = wz$ in $F' \setminus F$ mit Endpunkten in verschiedenen Komponenten von $G[F]$. Dann ist auch $F' \cup \{e\}$ kreisfrei. \square

Eine vergleichbare Eigenschaft ist auch in Vektorräumen zu beobachten. Bekannt ist dies als das Ergänzungslemma oder Austauschlemma von STEINITZ. Sind F, F' zwei linear unabhängige Teilmengen im Vektorraum V über K und ist $|F| < |F'| < \infty$, so gibt es einen Vektor $\vec{x} \in F' \setminus F$, sodass $F \cup \{\vec{x}\}$ linear unabhängig bleibt.

Die maximal kreisfreien Teilmengen eines zusammenhängenden Graphen G sind genau die Kantenmengen seiner Spannbäume. Spannbäume lassen also sich nicht nur darüber charakterisieren, dass sie minimal zusammenhängend sind, sondern auch maximal kreisfrei. Je zwei Spannbäume haben dieselbe Mächtigkeit, nämlich $|V(G)| - 1$.

Im Folgenden wird mit einer **Gewichtsfunktion** $w : E(G) \rightarrow \mathbb{R}_{\geq 0}$ gearbeitet, die Kanten ein Gewicht zuordnet. Solch ein Gewicht kann bspw. Kosten für Strecken in einem Transportnetz entsprechen.

1.19. Für gegebenes $w : E(G) \rightarrow \mathbb{R}_{\geq 0}$ heißt ein Spannbaum T von G ein **Spannbaum minimalen Gewichtes**, falls

$$w(T) := \sum_{e \in E(T)} w(e) \leq w(S) = \sum_{e \in E(S)} w(e) \quad (1.42)$$

für jeden Spannbaum S von G gilt.

Ein Spannbaum kann bspw. genutzt werden, um in einem großen Netzwerk einen Pfad im Backbone zu jedem Knoten zu finden. Möglicherweise möchte man einen Spannbaum minimalen Gewichtes finden um möglichst billig zu jedem Knoten Pakete senden zu können.

Satz 1.6 (Kruskal, Greedy-Algorithmus). *Sei G ein zusammenhängender Graph und $w : E(G) \rightarrow \mathbb{R}_{\geq 0}$.*

Verfahren 1.4: Setze $F := \emptyset$.

(*) Wenn es in $E(G) \setminus F$ eine Kante e gibt, für die $F \cup \{e\}$ kreisfrei bleibt, wähle e so, dass $w(e)$ kleinstmöglich ist, setze $F := F \cup \{e\}$. Iteriere (*).

Das Verfahren endet mit einem Spannbaum von G minimalen Gewichtes, nämlich $T = G[F]$.

Es ist im Übrigen nicht wichtig, dass die Gewichtsfunktion w nur nichtnegative Gewichte zurückgibt. Es hilft jedoch beim Rechnen, wenn die Werte nur nichtnegativ sind. In der Praxis werden zudem überwiegend ganze Zahlen statt reeller Zahlen benutzt.

Das Verfahren endet mit einem maximal kreisfreien Graphen, da solange wie möglich und ausschließlich Kanten wählen, sodass der Graph kreisfrei bleibt. Da G zusammenhängend ist, enthält der Graph am Ende des Verfahrens alle Ecken aus G . Damit endet das Verfahren auch mit einem Spannbaum. Nicht selbstverständlich ist jedoch, dass der Spannbaum auch minimalen Gewichtes ist.

Beweis. Das Verfahren endet mit einer maximal kreisfreien Kantenmenge F , also ist $G[F]$ ein Spannbaum von G . ✓

Sei e_1, \dots, e_m die Folge der Kanten aus F in der Reihenfolge ihres Auftretens in der Iteration. Sei f_1, \dots, f_m die Folge der Kanten eines beliebigen Spannbaumes B in der Reihenfolge aufsteigender Einzelgewichte. $m = |V(G)| - 1$.

Wäre $w(F) > w(E(B))$, dann gäbe es einen Index $i \in \{1, \dots, m\}$ mit $w(e_i) > w(f_i)$. (Wähle i kleinstmöglich.)

$$F'' := \{e_1, \dots, e_{i-1}\} \quad (1.43)$$

ist kreisfrei.

$$F' := \{f_1, \dots, f_{i-1}, f_i\} \quad (1.44)$$

ist kreisfrei.

$$(1.45)$$

$|F''| < |F'|$ impliziert nach [Satz 1.5](#): Es gibt $f \in F' \setminus F''$ so, dass $F'' \cup \{f\}$ kreisfrei bleibt. f ist folglich eine Option für die Wahl von e_i im i -ten Iterationsschritt gewesen. Es gilt nach der Wahl von e_i : $w(e_i) \leq w(f)$. Aber $w(f) \leq w(f_i) < w(e_i)$. ✗

Also gilt tatsächlich

$$w(E(B)) \geq w(F) = w(E(G[F])). \quad (1.46)$$

□

Im Beweis wurde im wesentlichen das Austauschlemma und die Eigenschaft, dass jede Teilmenge einer kreisfreien Menge wieder kreisfrei ist, verwendet. Dies legt nahe, dass [Satz 1.6](#) sich in allgemeinere Situation übertragen lässt.

1.20. Ein Paar $M = (E, \mathfrak{F})$ aus einer endlichen Menge E und einer Menge \mathfrak{F} von Teilmengen von E heißt ein (endliches) **Matroid** auf E , falls gilt:

- (1) $\emptyset \in \mathfrak{F}$,
- (2) aus $F \subseteq F' \in \mathfrak{F}$ folgt $F \in \mathfrak{F}$ und
- (3) zu $F, F' \in \mathfrak{F}$ mit $|F| < |F'|$ gibt es ein $e \in F' \setminus F$ mit $F \cup \{e\} \in \mathfrak{F}$.

Die Mengen aus \mathfrak{F} heißen gewöhnlich **unabhängig**, bezüglich \subseteq maximale unabhängige Mengen heißen **Basen** von M .

Aus der dritten Bedingung ergibt sich sofort, dass je zwei Basen eines Matroids dieselbe Mächtigkeit haben. [Satz 1.6](#) lässt sich auf die allgemeinere Situation von Matroiden übertragen.

Satz 1.7 (Greedy-Algorithmus für Matroide). *Sei $M = (E, \mathfrak{F})$ ein Matroid und $w : E \rightarrow \mathbb{R}_{\geq 0}$.*

Verfahren 1.5: Setze $F := \emptyset$.

(*) Wenn es in $E \setminus F$ ein e gibt, für das $F \cup \{e\} \in \mathfrak{F}$ gilt, wähle e so, dass $w(e)$ kleinstmöglich ist, setze $F := F \cup \{e\}$ und iteriere (*).

Das Verfahren endet mit einer Basis F von M minimalen Gesamtgewichts

$$w(F) := \sum_{e \in F} w(e). \quad (1.47)$$

Beweis. Das Verfahren liefert offenbar eine maximal unabhängige Menge, also eine Basis F . Seien e_1, \dots, e_m die Elemente, die im Verlauf des Algorithmus gewählt wurden, und zwar in der Reihenfolge ihrer Auswahl. Seien f_1, \dots, f_m die Elemente einer beliebigen Basis B mit minimalem Gesamtgewicht in der Reihenfolge aufsteigender Einzelgewichte. Wäre $w(F) > w(B)$, so wäre $w(e_i) > w(f_i)$ für mindestens ein i und wir wählen i kleinstmöglich. Nach der Eigenschaft (2) für Matroide sind $F := \{e_1, \dots, e_{i-1}\}$ und $F' := \{f_1, \dots, f_i\}$ unabhängig; nach der Austauscheseigenschaft (3) für Matroide gibt es ein $f \in F' \setminus F$ so, dass $F \cup \{f\}$ unabhängig bleibt; f ist also eine Option bei der i -ten Iteration von (*), d. h. bei der Wahl von $e = e_i$, gewesen, und somit gilt $w(e_i) \leq w(f)$. Es gilt aber $w(f) \leq w(f_i) < w(e_i)$. ⚡

Also ist $w(F)$ minimal. □

1.3. Das Traveling-Salesman-Problem

Das Traveling-Salesman-Problem (kurz: TSP) lässt sich wie folgt beschreiben: Man finde eine Rundreise durch gegebene Städte derart, dass man jede Stadt genau einmal besucht (außer der ersten, die man ein zweites Mal sieht, am Ende der Reise) und die unter allen solchen Rundreisen minimale Gesamtkosten verursacht. Die variablen Kostenanteile entstehen in diesem Modell natürlich entlang der Reisewege (die Summe aller Kosten in den

Städten selber darf als konstant angenommen werden); für je zwei Städte A, B sind die Kosten $w(A, B)$ für die Reise von A nach B bekannt, und die zu minimierenden Kosten ergeben sich durch Summation. Wir wollen einige Zusatzannahmen ins Spiel bringen, dass nämlich erstens diese Kosten nichtnegativ sind, dass zweitens $w(A, B) = w(B, A)$ für je drei Städte A, B, C ist (Dreiecksungleichung). Hierdurch wird die Diskussion ein wenig vereinfacht, das neue Problem nennt man auch das *metrische* TSP.

Da die Zahl der Rundreisen durch n Städte im wesentlichen $n!$ ist und daher explodiert, ist „Durchprobieren“ hier nicht die Methode der Wahl. (Tatsächlich gehört das Problem zu den NP-vollständigen Problemen, siehe). Begnügt man sich dagegen mit einem Verfahren, das zwar nicht die optimale, jedoch eine Lösung von beweisbar (!) „guter“ Qualität liefert, sieht die Welt anders aus: Wir beschreiben, wie mit Hilfe von [Satz 1.6](#) eine Rundreise schnell gefunden werden, deren Kosten weniger als doppelt so groß wie die einer optimalen Rundreise sind. Daneben bietet eine wie auch immer ermittelte „gute“ obere Schranke für die Kosten einer optimalen Rundreise die Möglichkeit, in auf verzweigter Suche beruhenden Verfahren den Suchraum einzuschränken (branch-and-cut).

Ref Abschnitt 4.3

Verfahren 1.6:

- (1) Sei V die Menge der Städte, etwa: $|V| = n$, E die Menge aller zweielementigen Teilmengen von V und $w : E \rightarrow \mathbb{R}_{\geq 0}$ die Kostenfunktion. Wir sind nun mit $G := (V, E)$ in der Situation des Satzes von Kruskal, der es uns erlaubt, einen Spannbaum T von G minimalen Gesamtgewichtes zu finden.
- (2) Entlang dieses Spannbaums finden wir eine Folge $R = x_0, \dots, x_{2n-1}$ derart, dass es zu jeder Kante e von T genau zwei Indizes j, k gibt mit $x_j x_{j+1} = x_k x_{k+1} = e$ (alle Indizes modulo $2n$, Beweis zum Beispiel induktiv durch Löschen von Blättern).
- (3) Zunächst sei das *Gesamtgewicht* $w(S)$ einer Folge $S = x_0, \dots, x_{l-1}$ definiert durch:

$$w(S) := \sum_{i=0}^{l-1} w(x_i x_{i+1}). \quad (1.48)$$

Die im zweiten Schritt konstruierte Folge enthält jede Ecke aus V *mindestens* einmal und hat Gesamtkosten $w(R) \leq 2w(T)$. Wir dünne sie schrittweise aus: Ist S wie oben und $x_i = x_j$ für $i < j$, so streichen wir x_j aus der Folge und erhalten S' . In den Gesamtkosten tritt dann anstelle des Doppelterms $w(x_{i-1}x_i) + w(x_i x_{i+1})$ der Term $w(x_{i-1}x_{i+1})$ auf. Infolge der Dreiecksungleichung wachsen die Kosten beim Übergang von S nach S' nicht. Durch Iteration erhält man schließlich eine Rundreise S , in der jede Stadt *genau* einmal vorkommt und nach wie vor gilt $w(S) \leq 2w(T)$ (Vor der Iteration werden die Ecken x_i für $i > j$ umindiziert auf x_{i-1} .)

Sei nun $Q = z_0, \dots, z_{n-1}$ eine optimale Rundreise. Obwohl wir Q nicht kennen, können wir doch die Kosten von Q und S vergleichen, denn $P := (V, \{z_i z_{i+1} \mid i \in \{0, \dots, n-1\}\})$ ist ein Spannbaum von G und $w(Q) = w(P) + w(z_{n-1}z_0) \geq w(P) \geq w(T)$. Infolgedessen kommt $w(S) \leq 2w(T) \leq 2w(Q)$, d. h. die Kosten unserer Lösung sind, wie versprochen, nach oben beschränkt durch das Doppelte der Kosten einer optimalen Rundreise.

Der beste bekannte Gütefaktor ist 1.5 (CHRISTOFIDES Algorithmus), und es lässt sich zeigen, dass das Problem, eine Rundreise mit Gütefaktor 1.0081 zu finden, zu den NP-vollständigen Problemen gehört. Für den Fall euklidischer Kosten (d. h. die Städte sind Punkte in der Ebene und die Kosten ihr euklidischer Abstand) kann dagegen für jedes $c > 1$ ein Polynomialzeitapproximationsalgorithmus mit Gütefaktor c angegeben werden. Man spricht in diesem Fall von einem *Polynomialzeitapproximationsschema*; die Klasse aller Probleme mit dieser Eigenschaft heißt entsprechend *PTAS* (engl. *polynomial time approximation scheme*).

1.4. Der Satz von Courcelle

Die allermeisten Optimierungs- und Entscheidungsprobleme auf Graphen sind trivial oder sehr leicht, wenn man sie auf die Teilklasse der Bäume einschränkt. Es liegt nahe, dass auch „baumartige“ Graphen ganz erheblich einfach zu behandeln sind. Dies ist tatsächlich durch einen Satz der theoretischen Informatik sichergestellt, den wir hier „nur“ beschreiben wollen.

Wie misst man die „Baumartigkeit“ von Graphen? Hierzu beschreiben wir zuerst, wie ein Graph überhaupt in etwas „Baumartiges“ zerlegt werden kann:

1.21. Ist G ein Graph, so heißt ein Paar $z = (T, (X_s)_{s \in V(T)})$ bestehend aus einem nichtleeren Baum T und einer mit den Ecken aus T indizierten Familie von Teilmengen X_s von $V(G)$ eine **Baumzerlegung** von G , falls gilt:

- (i) $\cup_{s \in V(T)} X_s = V(G)$,
- (ii) $\forall e \in E(G) \exists s \in V(T) : V(e) \subseteq X_s$ und
- (iii) $\forall p, q, s \in V(T) : \text{Kommt } q \text{ im } p, s\text{-Weg in } T \text{ vor, so gilt } X_q \supseteq X_p \cap X_s.$

Der Baum T heißt dann **Zerlegungsbaum**. Die X_s heißen **Taschen**.

Der Zerlegungsbaum T enthält also als Elemente die Taschen X_s . Alle Taschen zusammen enthalten alle Knoten $V(G)$ und jede Kante $e \in E(G)$ ist vollständig von einer Tasche „umschlossen“ (d. h. es gibt eine Tasche, in der beide Endknoten von e enthalten sind). Jede Tasche ist wiederum ein Unterbaum von G und damit gelten innerhalb einer Tasche alle Zusammenhänge, die für Bäume gelten.

Ein großer Zerlegungsbaum T zerlegt also den Graphen G in viele Teile, die in baumartiger Weise verbunden sind, und wird somit der Aufgabe, den Graphen entsprechend zu strukturieren, gerecht (sofern die Teile wirklich verschieden sind). Dabei mag es jedoch geschehen, dass einzelne Taschen X_s der Zerlegung groß bzw. beliebig komplex sind (sieht man einmal von dem konzeptionellen Einwand ab, dass man aus einer Zerlegung immer eine mit viel größerem Zerlegungsbaum konstruieren kann). Fordert man dagegen, dass die Taschen gleichmäßig klein sind, so wird für große Graphen ein großer Zerlegungsbaum erzwungen, der den Graphen in übersichtliche Teile strukturiert. Diese Idee kann man wie folgt umsetzen:

1.22. Die **Weite** von z ist definiert durch

$$w(z) := \max \{ |X_s| \mid s \in V(T) \} - 1 \quad (1.49)$$

und die **Baumweite** von G ist

$$w(G) := \min \{ w(z) \mid z \text{ ist Baumzerlegung} \} \quad (1.50)$$

Je kleiner die Baumweite ist, desto baumartiger ist der Graph. Bäume selber haben Baumweite 1, Kreise haben Baumweite 2 (in die Länge gezogen sehen sie ja wie Bäume aus, sogar wie Wege), etc.

Das Wissen um eine beschränkte Baumweite kann man wie folgt nutzen: Ausgehend von den Blättern des Zerlegungsbaumes löst man das Problem zunächst lokal in den von den korrespondierenden Taschen induzierten Teilgraphen und fügt dann, übergehend zu größeren und größeren Hauptästen des Zerlegungsbaums, lokale Lösungen bzw. Antworten mehr und mehr zusammen, um schließlich mit einer globalen Lösung bzw. Antwort zu enden. COURCELLE hat entdeckt, dass sich eine große Klasse von Graphenproblemen auf diese Weise lösen lässt, nämlich alle, die sich durch eine Formel in *erweiterter monadic second order logic*. Dies sind aussagenlogische Formeln, wobei über Ecken, Kanten, sowie Mengen von Ecken oder Kanten quantifiziert werden darf (MSO_2 -Formeln). Gewöhnlich wird neben \in und $=$ ein Inzident- und Adjazenzprädikat gefordert, was aber unter unseren Modellannahmen direkt abgebildet werden kann:

$$\text{inc}(x, e) := x \in e \quad (1.51)$$

$$\text{adj}(x, y) := (x \neq y) \wedge \exists e \in E : x \in e \wedge y \in e \quad (1.52)$$

Dies ermöglicht zunächst nur Formulierungen von *Entscheidungsproblemen* wie: Besitzt der Eingabegraph einen aufspannenden Kreis? Man kann sich dabei stufenweise einen Prädikate-Fundus von MSO_2 -formulierbaren Eigenschaften aufbauen. Zum Beispiel ist die Eigenschaft $X \subseteq V$, eine echte nichtleere Teilmenge von V zu sein, so formulierbar:

$$\text{proper}(X) := \exists y \in V \exists z \in V : y \in X \wedge z \notin X \quad (1.53)$$

und die Eigenschaft von $C \subseteq E$, dass $G[C]$ zusammenhängend ist, so:

$$\text{conn}(C) := \forall X \subseteq V : \text{proper}(X) \implies (\exists y \in V \exists z \in V : y \in X \wedge z \notin X \wedge xy \in E), \quad (1.54)$$

wobei $xy \in E$ eine (vielleicht) etwas lesbarere Variante des Adjazenzprädikats $\text{adj}(x, y)$ darstellt.

Satz 1.8 (COURCELLE). *Jede in MSO_2 formulierbare Grapheigenschaft kann in Linearzeit auf jeder Klasse von Graphen beschränkter Baumweite entschieden werden.*

Ein entsprechender Satz gilt für eine große Klasse von Optimierungsproblemen. Darin darf die Ecken- und Kantenmenge des Eingabegraphen ganzzahlig gewichtet werden und man kann nach einer Ecken- oder Kantenmenge maximalen oder minimalen Gewichts mit einer in MSO_2 formulierbaren Eigenschaft fragen, was natürlich Fragen nach Ecken- oder Kantenmengen maximaler oder minimaler Kardinalität einschließt. Zum Beispiel kann nach einer größtmöglichen *Clique*, d. h. einer Menge paarweise benachbarter Ecken, gefragt werden. Dies ist ein im allgemeinen schweres Optimierungsproblem, das aber – genau wie das im allgemeinen schwere Entscheidungsproblem, ob es einen aufspannenden Kreis gibt – auf Graphen beschränkter Baumweite in Polynomialzeit gelöst werden kann.

2. Matchings

2.1. Matchings in bipartiten Graphen

Lücke

$$V(M) := \cup M \quad (2.1)$$

$$|V(M)| = 2 |M| \quad (2.2)$$

2.1. Ein Matching M des Graphen G heißt **Matching** $A \subseteq V(G)$, falls $A \subseteq V(M)$.

2.2. Für $X \subseteq V(G)$ sei die **Nachbarschaft** von X definiert durch:

$$N_G(X) := \{y \in V(G) \setminus X \mid \exists z \in X : yz \in E(G)\} \quad (2.3)$$

Die Nachbarschaft $N_G(X)$ umfasst alle Ecken, die durch eine Kante mit einer der Ecken aus X inzidieren. [Abbildung 2.1](#) zeigt eine Eckenmenge X , ihre Nachbarschaft sowie die Kanten, die diese verbinden.

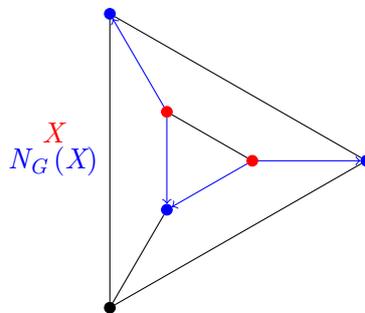


Abbildung 2.1.: Nachbarschaft einer Eckenmenge in einem Graphen

2.3. Für zwei Mengen X und Y sei die **symmetrische Differenz** von X und Y definiert durch:

$$X \Delta Y := (X \cup Y) \setminus (X \cap Y) \quad (2.4)$$

$$= (X \setminus Y) \cup (Y \setminus X) \quad (2.5)$$

Satz 2.1 (Heiratssatz, HALL). *Sei A Klasse einer 2-Färbung des bipartiten Graphen G . Genau dann gibt es ein Matching von A in G , wenn die **Hall-Bedingung** erfüllt ist:*

$$\forall X \subseteq A : |N_G(X)| \geq |X| \quad (2.6)$$

2.4. Sei M ein Matching des (nicht notwendig bipartiten) Graphen G . Ein Weg $P = x_0, x_1, \dots, x_l$ heißt **M -alternierend**, falls für jedes $i \in \{0, \dots, l-1\}$ die Kante $x_i x_{i+1}$ aus G genau dann zu M gehört, wenn i ungerade ist.

Ein solches P heißt **Verbesserungsweg**, falls er in einer Ecke außerhalb von $V(M)$ beginnt und endet, d. h. $x_0, x_l \notin V(M)$.

Ein solcher Verbesserungsweg ist in [Abbildung 2.2](#) gezeigt. Solche Wege beginnen immer mit einem Knoten x_0 , der nicht Teil des Matchings ist und enden mit einem Knoten x_l (hier: x_{2n+1}), welcher auch nicht Teil des Matchings ist. Weiterhin ist der weg ein alternierender Weg, d. h. die Kanten sind abwechselnd Teil des Matchings und nicht Teil des Matchings, wobei die erste Kante nicht Teil des Matchings ist.

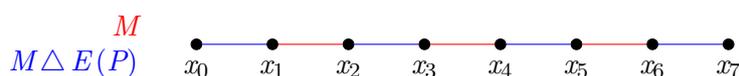


Abbildung 2.2.: Verbesserungsweg

Beweis. (2.6) ist notwendig für die Existenz eines Matchings von A , denn ist M ein Matching von A , so gilt

$$\forall X \subseteq A : |N_G(X)| \geq |N_{G[M]}(X)| = |X|. \quad (2.7)$$

Ist P ein Verbesserungsweg, so hat P ungerade Länge und $M \triangle E(P)$ ist ein Matching von G mit einer Kante mehr als M ($|M \triangle E(P)| = |M| + 1$).

Seien G, A wie in den Voraussetzungen des Satzes und $B := V(G) \setminus A$. Sei M ein **größtes** Matching.

Annahme: M ist kein Matching von A . Wir finden eine Menge X , die die HALL-Bedingung verletzt (dies zeigt der Satz). Es gibt dann eine Ecke $x_0 \in A \setminus V(M)$.

Sei R die Menge aller y für die es einen in x_0 beginnenden und in y endenden M -alternierenden Weg gibt. Es gilt $R \setminus \{x_0\} \subseteq V(M)$. $y \in R \cap A \setminus \{x_0\}$ muss nach Konstruktion mit einer Kante aus M inzidieren, für $y \in R \cap B$ gilt das auch, da sonst ein Verbesserungsweg von x_0 nach y existieren würde.

Daher gibt es keine Kante aus M , die eine Ecke $y \in R$ mit einem $z \in V(G) \setminus R$ verbindet: Richtig für x_0 ; ist $y \in R \cap A$ Endpunkt eines M -alternierenden Weges $P = x_0, \dots, x_l = y$, so ist $y \in V(M)$, genauer $x_{l-1}y \in E(M)$, $x_{l-1} \in R$. Ist $y \in R \cap B$ Endpunkt eines M -alternierenden Weges $P = x_0, \dots, x_l = y$ und $z \in V(G) \setminus R$ mit $yz \in M$, so ist $P^+ = x_0, \dots, x_l, z$ M -alternierend, folglich $z \in R$. ζ

Somit liegen mit einer Ecke einer Kante aus M schon *beide* Ecken in R .

Es gibt keine Kante in G , die eine Ecke $y \in R \cap A$ mit einer Ecke $z \in V(G) \setminus R$ verbindet: Sonst wäre y Endpunkt eines M -alternierenden Weges $P = x_0, \dots, x_l = y$ und $P^+ = x_0, \dots, x_l, z$ M -alternierend, also $z \in R$. \downarrow

Somit gilt $N_G(R \cap A) \subseteq R \cap B$ und $|R \cap A| = |R \cap B| + 1$.

$$\implies |N_G(R \cap A)| \leq |R \cap B| \quad (2.8)$$

$$= |R \cap A| - 1 \quad (2.9)$$

$$< |R \cap A| \quad (2.10)$$

Also verletzt $X := R \cap A$ die HALL-Bedingung. \square

Aus diesem Beweis lässt sich ein Algorithmus erstellen, der entweder ein Matching von A findet oder zeigt, dass es ein solches Matching nicht geben kann.

Satz 2.1 ist **Toncias** (The obviously necessary condition is also sufficient). Der Begriff wurde durch C. St. J. A. Nash-Williams etabliert. Viele Sätze sind von dieser Bauart.

Die nächste Frage ist nun, wie man für allgemeine Graphen große Matchings findet.

Es ist schwierig zu entscheiden (NP-vollständig), ob ein Graph einen aufspannenden Kreis, einen sog. **Hamilton-Kreis** hat. Wenn es einen solchen Kreis gibt, gibt es automatisch auch ein perfektes Matching, welches alle Ecken enthält. Das Ermitteln eines solchen HAMILTON-Kreises ist extrem aufwändig, aber die Vorbedingung lässt sich sehr leicht prüfen.

Für einen beliebigen Graphen kann man *Valenzen* willkürlich vorschreiben. Dies sind ganze Zahlen, die Ecke zugeordnet sind. Ein Problem ist nun, einen Teilgraphen zu finden, dessen Ecken genau so viele inzidierende Kanten pro Ecke wie die jeweilige Valenz hat. Wenn man an alle Ecken nun die Valenz 2 notiert (einen 2-Faktor ermittelt), könnte man vermuten, dass sich dadurch ein HAMILTON-Kreis finden lässt. Allerdings ist es hier weiterhin möglich, dass der Teilgraph dann kein HAMILTON-Kreis ist (nämlich genau dann, wenn der Teilgraph nicht zusammenhängend ist). Beispielsweise hat der PETERSEN-Graph einen 2-Faktor (z. B. wie in [Abbildung 2.3](#) gezeigt), aber keinen HAMILTON-Kreis.

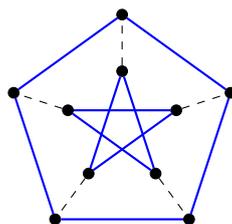


Abbildung 2.3.: 2-Faktor in einem PETERSEN-Graph

Satz 2.2 (Berge). *Sei M ein Matching des Graphen G . Genau dann gibt es ein Matching von G mit mehr Kanten als M , wenn es einen Verbesserungsweg für M gibt.*

Beweis. Existiert ein Verbesserungsweg, so auch ein größeres Matching (für M). Siehe Beweis zu [Satz 2.1](#). Existiere ein Matching N mit $|N| > |M|$. Betrachte $M \triangle N$. In $H = G[M \triangle N]$ ist jede Ecke mit höchstens zwei Kanten inzident. Die Komponenten von H sind daher Kreise oder Wege, die abwechselnd Kanten aus M und N verwenden. Weil $|N| > |M|$ muss eine dieser Komponenten mehr Kanten aus N als aus M benutzen. Diese ist ein Wege Weg ungerade Länge, der mit einer Kante aus N beginnt und endet; dieser Weg ist Verbesserungsweg. \square

2.2. Faktorsätze

Faktorsätze haben zum Ziel, die Existenz von (möglichst großen) Matchings in beliebigen Graphen zu zeigen. Wir betrachten zunächst Komponenten eines Graphen, die nach Lösung von Ecken hervorgehen.

Sei $q(G)$ die Anzahl der *ungeraden* Komponenten eines Graphen. Sei M ein Matching von G , $U = V(G) \setminus V(M)$ („ungematchte Ecken“) und $S \subseteq V(G)$.

Betrachtet man nun eine ungerade Komponente in $G - S$. Dann kann geschehen, dass alle Matchingkanten beide Endpunkte in einer solchen ungeraden Komponenten haben. Da die Komponente jedoch ungerade ist, gibt es eine ungematchte Ecke oder die Ecke kann nur gematcht sein, indem eine Kante nach S gezogen wird. Bei gerade Matchings kommt dies nicht vor. Gibt es jetzt mehr ungerade Komponenten als $|S|$, muss es eine ungematchte Ecke geben, da keine Ecke mehr in S vorhanden ist, die nicht bereits gematcht wurde.

Hieraus ergibt sich also eine untere Schranke für die Anzahl $|U|$ ungematchter Ecken:

$$q(G - S) - |S| \leq |U| = |V(G) \setminus V(M)| \quad (2.11)$$

Im Fall der Gleichheit gilt für jedes Matching M' von G dieselbe Ungleichung.

$$\underbrace{|V(G) \setminus V(M')|}_{|U'|} \geq q(G - S) - |S| = |U| = |V(G) \setminus V(M)| \quad (2.12)$$

Jedes M' ist also höchstens so groß wie M , also ist M ein größtes Matching.

2.5. Eine Menge $S \subseteq V(G)$, für die es ein Matching M gibt mit $q(G - S) - |S| = |V(G) \setminus V(M)|$, heißt **Barriere**.

Da \leq sowieso gilt, ist die Forderung „ \geq “ statt „ $=$ “ schon ausreichend, um zu zeigen, dass S eine Barriere ist.

Nun stellt sich die Frage, wie sich ein solches größtes Matching bzw. eine Barriere finden lässt, um zu zeigen, dass ein Matching ein größtes Matching ist. Der Satz von BERGE zeigt zunächst, dass eine Barriere für jeden Graphen existiert. Also gibt es auch immer ein größtes Matching für einen Graphen.

Bild 2.3

bild 2.3

Beispiele:

- Besitzt G ein **perfektes Matching** M (d. h. $V(M) = V(G)$), dann ist die leere Menge eine Barriere, denn

$$\underbrace{q(G - \emptyset) - |\emptyset|}_{=0} = 0 = |V(G) \setminus V(M)|. \quad (2.13)$$

Da ein perfektes Matching alle Ecken enthält, kann es nur gerade Komponenten in G geben. Auch für jedes $x \in V(G)$ ist $\{x\}$ eine Barriere von G , denn

$$\underbrace{q(G - \{x\})}_{=1} - \underbrace{|\{x\}|}_{=1} = 0 = |V(G) \setminus V(M)|. \quad (2.14)$$

Für perfekte Matchings lassen sich also ganz leicht Barrieren finden.

- Besitzt für ein gewisses x der Graph $G - x$ ein perfektes Matching M , dann ist \setminus eine Barriere von G , denn

$$\underbrace{q(G - \emptyset)}_{\geq 1} - |\emptyset| \geq 1 = |V(G) \setminus V(M)|. \quad (2.15)$$

Gäbe es keine ungerade Komponente, könnte es keine ungerade Anzahl Ecken geben. Also muss es mindestens eine ungerade Komponente geben.

Graphen, in denen für *jede* Ecke x der Graph $G - x$ ein perfektes Matching besitzt, werden als **faktorkritisch** bezeichnet. Sie besitzen immer die Barriere \emptyset und haben in der Regel eine große Anzahl größter Matchings.

Natürlich reichen diese Beispiel nicht aus, um zu zeigen, dass alle Graphen solche Barrieren besitzen. Dies möchten wir induktiv zeigen, wobei wir aus Barrieren kleinerer Instanzen Barrieren größerer Instanzen zeigen. Dafür werden zunächst *Induktionsmotoren* aufgestellt, also Zusammenhänge, die uns die Induktion ermöglichen.

Lemma 2.3 (Induktionsmotor (A)). *Sind G und H disjunkte Graphen (d. h. $V(G) \cap V(H) = \emptyset$) und sind S, T Barrieren von G bzw. H , so ist $S \dot{\cup} T$ eine Barriere von $G \cup H$ mit*

$$G \cup H = (V(G) \cup V(H), E(G) \cup E(H)). \quad (2.16)$$

Beweis. Also gibt es Matchings M, N von G und H mit

$$q(G - S) - |S| = |V(G) \setminus V(M)| \quad (2.17)$$

$$q(H - T) - |T| = |V(H) \setminus V(N)| \quad (2.18)$$

und $M \cap N = \emptyset$ (\checkmark). Daher gilt für die Vereinigung $G \cup H$:

verweis
bild 2.4

bild 2.4

underbraces
schick
machen

$$q\left(\underbrace{(G \cup H)}_{(1)} - \underbrace{(S \cup T)}_{(2)}\right) = \underbrace{q(G - T) + q(H - T)}_{(1)} - \underbrace{|S| - |T|}_{(2)} \quad (2.19)$$

$$= |V(G) \setminus V(M)| + |V(H) \setminus V(N)| \quad (2.20)$$

$$= |V(G \cup H) \setminus V(M \cup N)| \quad (2.21)$$

Mit $M \cup N$ existiert folglich ein Matching von $G \cup H$ so, dass

$$q((G \cup H) - (S \cup T)) - |S \cup T| = |V(G \cup H) \setminus V(M \cup N)| \quad (2.22)$$

ist. Also ist $S \cup T$ Barriere von $G \cup H$. \square

Lemma 2.4 (Induktionsmotor (B)). *Ist x eine Ecke, die in jedem größten Matching von G enthalten ist (eine sogenannte **wesentliche** Ecke) und ist S eine Barriere von $G - x$, dann ist $S \cup \{x\}$ eine Barriere von G .*

Beweis. Sei M ein größtes Matching von G . Nach Voraussetzung gibt es ein Matching N von G mit

$$q((G - x) - S) - |S| = |V(G - x) \setminus V(N)|. \quad (2.23)$$

N kann kein größtes Matching von G sein, denn sonst würde N die Ecke x enthalten, da sie wesentlich ist. Also:

$$q(G - (S \cup \{x\})) - |S \cup \{x\}| = q((G - x) - S) - |S| - 1 \quad (2.24)$$

$$= |V(G - x) \setminus V(N)| - 1 \quad (2.25)$$

$$= |V(G) \setminus V(N)| - 1 - 1 \quad (2.26)$$

$$= |V(G) \setminus V(N)| - 2 \quad (2.27)$$

$$= |V(G)| - \underbrace{|V(N)|}_{\geq |V(M)| - 2} - 2 \quad (2.28)$$

$$\geq |V(G)| - (|V(M)| - 2) - 2 \quad (2.29)$$

$$= |V(G)| - |V(M)| \quad (2.30)$$

$$= |V(G) \setminus V(M)| \quad (2.31)$$

Nebenbemerkung:

$$|N| < |M| \implies |V(N)| \leq |V(M)| - 2 \quad (2.32)$$

Daraus ergibt sich:

$$q(G - (S \cup \{x\})) - |S \cup \{x\}| \geq |V(G) \setminus V(M)| \quad (2.33)$$

Also ist $S \cup \{x\}$ Barriere von G . \square

Mit diesen beiden Lemmata lässt sich nun der Satz von BERGE und TUTTE zeigen:

Satz 2.5 (BERGE, TUTTE). *Jeder zusammenhängende Graph G ohne wesentliche Ecken ist faktorkritisch.*

Die Erkenntnis, dass die Abwesenheit wesentlicher Ecken bereits ausreicht, damit ein Graph faktorkritisch ist, ist überraschend.

Beweis. \mathcal{Z} : (*) Für jedes größte Matching M und zwei Ecken $x \neq y$ von G ist $V(M) \cap \{x, y\} \neq \emptyset$. Dies wird induktiv über den Abstand $d_G(x, y)$ gezeigt.

Angenommen: $\{x, y\} \cap V(M) = \emptyset$. Betrachte kürzesten x, y -Weg P in G . P hat die Länge ≥ 2 , denn sonst wäre $M \cup \{xy\}$ ein Matching von G . ζ

Also besitzt P eine innere Ecke w .

$$d_G(w, x), d_G(w, y) < d_G(x, y) \quad (2.34)$$

Nach Induktionsvoraussetzung ist $w \in V(M)$.

Weil w nach Voraussetzung nicht wesentlich ist, gibt es ein größtes Matching N von G mit $w \notin V(N)$. Nach Induktionsvoraussetzung gilt $V(N) \cap \{w, x\} \neq \emptyset$, also $x \in V(N)$, analog $y \in V(N)$.

bild 2.5

Der Graph $H := G[E(M) \triangle E(N)]$ besteht aus Wegen und Kreisen gerader Länge. Auch die Wege haben gerade Länge, weil sie sonst Verbesserungswege für M oder N wären. Jede der Ecken w, x, y liegt in einem Weg in H von gerader Länge, und zwar als Endpunkt. Weil x und y beide aus $V(N)$ kommen, w aber nicht, gehören x, y nicht zum selben Weg in H . Also gehört w nicht zum selben Weg in H wie x oder wie y . w (und y) gehören nicht zu dem Weg in H , der x enthält.

Sie Q jener Weg. $N \triangle E(Q)$ ist dann ein größtes Matching von G , das weder x noch w überdeckt. Aber nach IV enthält jedes größte Matching x oder w . ζ

Sei jetzt x Ecke von G . Da x nicht wesentlich ist, gibt es ein größtes Matching M von G mit $x \notin V(M)$. Daher $y \in V(M) \forall y \notin x$ aus $V(G)$. Also ist M ein perfektes Matching von $G - \{x\}$. Also ist G faktorkritisch. \square

Folgerung 2.6. *Jeder Graph G besitzt eine Barriere.*

Beweis. Induktiv über $|V(G)|$. Sei die Behauptung für Graphen weniger als $|V(G)|$ Ecken bewiesen. Wegen Satz 2.3 dürfen wir annehmen, dass G zusammenhängend ist. Wegen Satz 2.4 dürfen wir annehmen, dass G keine wesentlichen Ecken besitzt. Aufgrund von Satz 2.5 ist G faktorkritisch. Dann besitzt G eine Barriere. \square

Siehe auch Edmonds „blossom algorithm“. Dieser ist ein solcher Algorithmus, der eine Barriere für ein größtes Matching zu finden.

Folgerung 2.7 (TUTTE-BERGE-Formel). *Die größte Mächtigkeit eines Matchings des Graphen G ist gleich*

$$\frac{1}{2} \min \{ |V(G)| - q(G - S) + |S| \mid S \subseteq V(G) \} \quad (2.35)$$

Es ist natürlich nicht praktikabel, die Zahl auf diese Weise auszurechnen, jedoch lässt sich die Berechnung gut optimieren, indem Monotonieeigenschaften ausgenutzt werden. Dieses Theorem ist ein sogenanntes *Min-Max-Theorem*.

Beweis. Nach [Satz 2.6](#) besitzt G eine Barriere S . Also gibt es nach der Definition von Barrieren ein größtes Matching M mit

$$q(G - S) - |S| = \underbrace{|V(G) \setminus V(M)|}_{=|V(G)| - |V(M)|} \quad (2.36)$$

$$\implies |V(M)| = |V(G)| - q(G - S) + |S| \quad (2.37)$$

$$\implies |M| = \frac{1}{2} (|V(G)| - q(G - S) + |S|) \quad (2.38)$$

Für jedes weitere $S' \subseteq V(G)$ gilt:

$$q(G - S') - |S'| \leq |V(G) \setminus V(M)| \quad (2.39)$$

$$\implies |V(M)| \leq |V(G)| - q(G - S') + |S'| \quad (2.40)$$

$$\implies |M| \leq \frac{1}{2} (|V(G)| - q(G - S') + |S'|) \quad \forall S' \subseteq V(G) \quad (2.41)$$

□

Folgerung 2.8 (Faktorsatz von TUTTE, 1-Faktor-Satz). *Genau dann besitzt der Graph G ein perfektes Matching, wenn $\forall S \subseteq V(G) : q(G - S) \leq |S|$.*

Diese Bedingung kann man praktisch zwar nicht durch Bilden aller Teilmengen nachrechnen, jedoch lassen sich auch hier wieder Monotonieeigenschaften ausnutzen, um Rechenaufwand zu vermeiden. Dieses Theorem fällt auch wieder in die Kategorie *TONCI-AS*.

Beweis. „ \rightarrow “ Besitze G ein perfektes Matching M . Dann gilt für jedes $S \subseteq V(G)$:

$$q(G - S) - |S| \leq |V(G) \setminus V(M)| \quad (2.42)$$

$$= 0 \quad (2.43)$$

$$\implies q(G - S) \leq |S|. \quad (2.44)$$

✓

„ \leftarrow “ Gelte umgekehrt $\forall S \subseteq V(G) : q(G - S) \leq |S|$. Wegen [Satz 2.6](#) gibt es eine Barriere S von G . Aufgrund der Definition der Barriere gibt es größtes Matching M mit

$$0 \geq q(G - S) - |S| = |V(G) \setminus V(M)| \geq 0 \quad (2.45)$$

$$\implies |V(G) \setminus V(M)| = 0 \quad (2.46)$$

$$\implies V(M) = V(G) \quad (2.47)$$

Also ist M perfekt.

✓

□

Mit diesem Satz lässt sich im Übrigen auch der Satz von HALL auf eine andere Weise zeigen.

2.6. Der **Grad** $d_G(x)$ einer Ecke x im Graphen G ist die Anzahl der mit x inzidenten Kanten.

2.7. Ein Teilgraph H von G heißt f -defname von G , falls $V(H) = V(G)$ (H also **aufspannend**) ist und $\forall x \in V(G) : d_H(x) = f(x)$ (d. h. $f : V(G) \rightarrow \mathbb{Z}_{\geq 0}$).

Beispielsweise ist jeder G aufspannende Kreis (also ein HAMILTON-Kreis) ein 2-Faktor. In dem Fall ist $2(x) = 2$ (also die konstant-2-Funktion von $V(G)$ nach $\mathbb{Z}_{\geq 0}$). Ist M ein perfektes Matching, so ist $(V(G), M)$ ein 1-Faktor. Umgekehrt ist die Kantenmenge eines 1-Faktors von G eine perfektes Matching. Solche eine Umkehrung gilt jedoch nicht für einen Graphen mit aufspannendem Kreis! Wenn man nun eine Methode findet, einen 2-Faktor zu berechnen, heißt das noch nicht, dass sich damit auch ein aufspannender Kreis finden lässt.

Im Folgenden sei nun ein Verfahren gezeigt, um zu entscheiden, ob ein vorgegebener Graph zu vorgegebenem $f : V(G) \rightarrow \mathbb{Z}_{\geq 0}$ einen f -Faktor hat. Rückführung auf den Faktorsatz 2.8.

Verfahren 2.1: Offensichtlich notwendig für die Existenz eines f -Faktors ist

$$\forall x \in V(G) : f(x) \leq d_G(x). \quad (2.48)$$

Ersetze jede Ecke x von G durch einen bipartiten Graphen H_x mit Farbklassen A_x und B_x und allen Kanten ab , $a \in A_x$, $b \in B_x$. Dabei seien die $V(H_x)$ paarweise disjunkt und $|A_x| = d_G(x)$ und $|B_x| = d_G(x) - f(x)$.

Für jede Kante $e \in E(G)$, etwa $e = xy$, wähle $a_e \in A_x$ und $b_e \in A_y$ so, dass alle a_e, a'_e verschieden sind. Füge für jedes e die Kante $a_e b_e$ zum Hilfsgraphen hinzu, diese Kanten bilden ein Matching des Hilfsgraphen.

Dann besitzt G genau dann einen f -Faktor, wenn der Hilfsgraph ein perfektes Matching besitzt.

(Beweis durch Beispiel)

bild 2.6

Man kann jetzt natürlich fragen, ob es ein einfaches Kriterium gibt, um zu prüfen, ob ein f -Faktor existiert. Es gibt tatsächlich eine solche Ungleichung, um die Bedingung zu prüfen. Diese ist jedoch deutlich komplizierter als für 1-Faktoren.

Für $X, Y \subseteq V(G)$ (die nicht notwendigerweis disjunkt sind) sei

$$E_G(X, Y) := \{xy \in E(G) \mid x \in X, y \in Y\} \quad (2.49)$$

bild 2.7

Satz 2.9 (f -Faktorsatz, TUTTE). Sei G Graph, $f : V(G) \rightarrow \mathbb{Z}_{\geq 0}$. Genau dann besitzt G einen f -Faktor, wenn gilt:

$$\forall S, T \subseteq V(G), S \cap T = \emptyset : q(S, T) + \sum_{x \in T} (f(x) - d_{G-S}(x)) \leq \sum_{x \in S} f(x), \quad (2.50)$$

wobei $q(S, T)$ die Zahl der Komponenten C von $G - (S \cup T)$ ist, für die

$$E_G(V(C), T) + \sum_{x \in V(C)} f(x) \quad (2.51)$$

ungerade ist.

3. Flüsse

3.1. Das Max-Flow-Min-Cut-Theorem

Optimierung in Flussnetzwerken hat zum Ziel, einen Strom von irgendwas (bspw. Datenströme, Wasser) so durch das Netzwerk zu transportieren, dass möglichst viel gleichzeitig durch das Netz von einer Quelle zu einer Senke transportiert werden kann. Für diese Betrachtung werden jetzt gerichtete Graphen betrachtet.

3.1. Ein **Digraph** $D = (V, E)$ besteht aus einer Menge V von **Ecken** und einer Menge $E \subseteq \{(x, y) \mid x \neq y \in V\} = V^2 \setminus id_V$ von **Kanten** (engl. *arc* statt *edge*). Schreibe xy statt (x, y) .

Bei Digraphen ist es insbesondere auch möglich, dass es zwei Kanten zwischen zwei Ecken gibt, nämlich wenn eine Kante von der ersten zur zweiten und die andere Kante von der zweiten zur ersten Ecke geht. Allerdings werden solche Fälle für Flussnetzwerke zunächst nicht betrachtet.

Bild 3.1

Auch, wenn hier die Notation xy als Kurzschreibweise für die Kante (x, y) genau wie für ungerichtete Graphen benutzt wird, werden hier gerichtete Kanten bezeichnet. Das bedeutet also, dass $(x, y) = xy \neq yx = (y, x)$.

3.2. Für $X, Y \subseteq V$ sei

$$E_D(X, Y) := E \cap (X \times Y) = \{xy \in E \mid x \in X, y \in Y\} \quad (3.1)$$

die Menge aller Kanten von X nach Y . Für $X \subseteq V$ ist

$$E_D^+(X) := E_D(X, V \setminus X) \quad (3.2)$$

die Menge der **Außenkanten** von X und

$$E_D^-(X) := E_D(V \setminus X, X) \quad (3.3)$$

die Menge der **Innenkanten** von X . Für $x \in V$:

$$E_D^+(x) := E_D^+(\{x\}) \quad (3.4)$$

$$E_D^-(x) := E_D^-(\{x\}) \quad (3.5)$$

Der **Außengrad** einer Ecke x ist:

$$d_D^+(x) := |E_D^+(x)|. \quad (3.6)$$

$$d_D^-(x) := |E_D^-(x)| \quad (3.7)$$

ist der **Innengrad** der Ecke x .

bild 3.2,
bild 3.3

3.3. Zu einem Digraphen ist der **zugrundeliegende ungerichtete Graph**:

$$U(D) = (V(D), \{\{x, y\} \mid (x, y) \in E(D)\}) \quad (3.8)$$

Die Definitionen für ungerichtete Graphen übertragen sich (teilweise auch wortwörtlich) auf Digraphen bzw. können auf dem einem Digraphen D zugrundeliegenden ungerichteten Graphen. Man beachte jedoch, dass beim zugrundeliegenden ungerichteten Graphen zu einem Digraphen Informationen über die Richtungen der Kanten verloren gehen!

3.4. Aus Kreisen $C = x_1, x_1, \dots, x_{l-1}, x_0$ werden **kontinuierlich gerichtete Kreise** (d. h. $\forall i \in \{1, \dots, l-1\} : (x_{i-1}, x_i) \in E(D), (x_{l-1}, x_0) \in E(D)$), aus Wegen $P = x_0, \dots, x_l$ **kontinuierlich gerichtete Wege** (d. h. $\forall i \in \{1, \dots, l\} : (x_{i-1}, x_i) \in E(D)$).

bild 3.4

Die Relation $x \sim_D y : \iff \exists x, y\text{-Weg in } D$ ist reflexiv und transitiv, aber *nicht* symmetrisch. Die Existenz eines x, y -Weges impliziert nicht automatisch die Existenz eines y, x -Weges!

3.5. Für $f: E(D) \rightarrow \mathbb{R}$ und $F \subseteq E(D)$ sei

$$f(F) := \sum_{e \in F} f(e) \quad (3.9)$$

(statt wie üblich $\{f(e) \mid e \in F\}$).

Während anderswo der Begriff *Netzwerk* auch für Graphen allgemein benutzt wird, gibt es in der Flusstheorie eine feste Definition für Netzwerke:

3.6. Ein **Netzwerk** ist ein Quadrupel $N = (D, c, s, t)$ bestehend aus einem Digraphen D , einer **Kapazitätsfunktion** $c: E(D) \rightarrow \mathbb{R}_{\geq 0}$, einer **Quelle** $s \in V(D)$ und einer **Senke** $t \in V(D) \setminus \{s\}$.

bild 3.5
(ohne das
blaue)

Neben der Kapazitätsfunktion wird in Bildern üblicherweise auch noch die Flussfunktion eingezeichnet. Diese gibt an, wie viel *Fluss* über eine Kante geleitet wird.

3.7. Ein **Fluss** von N (auch: s, t -**Fluss**) ist eine Funktion $f: E(D) \rightarrow \mathbb{R}_{\geq 0}$ mit:

- (i) $f(e) \leq c(e) \forall e \in E(D)$ (Der Fluss darf nie größer als die Kapazität einer Kante sein.)
- (ii) $f(E_D^+(x)) = f(E_D^-(x)) \forall x \in V(D) \setminus \{s, t\}$ (Der eingehende Fluss einer Ecke muss genau so groß wie der ausgehende Fluss sein, außer für Quellen und Senken. Dies ist die sog. KIRCHOFF-Regel.)

Es darf also niemals eine Kante über ihre Kapazität hinaus ausgelastet werden und Kanten können keinen Fluss „speichern“, d. h. sie müssen immer sämtlichen eingehenden Fluss wieder über ausgehende Kanten herausgeben. Die einzige Ausnahme hierfür sind die Quelle und Senke. Für alle anderen Ecken gilt

$$\sum_{e \in E_D^+(x)} f(e) = \sum_{e \in E_D^-(x)} f(x) \quad (3.10)$$

Es ist prinzipiell auch möglich, dass es Fluss in die Quelle hinein bzw. Fluss aus der Senke heraus gibt!

Eine interessante Optimierungsfrage ist jetzt, wie viel Fluss maximal von s nach t „transportiert“ werden kann.

3.8. Die **Stärke** eines Flusses f im Netzwerk N ist:

$$\|f\| := f(E_D^+(s)) - f(E_D^-(s)) \quad (3.11)$$

Als Beispiel sie hier beispielhafter Fluss in gezeigt. An jeder Ecke des markierten Weges kommt eine Einheit Fluss an und eine Einheit Fluss geht aus (außer natürlich für Quelle und Senke). Damit ist die KIRCHHOFF-Bedingung erfüllt. Weiterhin ist der Fluss an jeder Kante kleiner als die Kapazität. Da an der Quelle eine Einheit Fluss abgeht und keine eingeht, ist die Stärke des Flusses $\|f\| = 1$. Man beachte in diesem Beispiel, dass der Weg über die oberen Ecken gar nicht genutzt wurde. Man könnte also prinzipiell mehr Fluss durch das Netzwerk leiten als hier gezeigt, indem Fluss über die oberen Kanten geleitet wird!

bild 3.6

3.9. Ist P ein s, t -Weg in D und $c > 0$, so wird durch

$$f_P(e) = \begin{cases} c & e \in E(P) \\ 0 & \text{sonst} \end{cases} \quad (3.12)$$

ein Fluss f_P der Stärke c definiert, falls $c(e) \geq c \forall e \in E(P)$ gilt. f_P heißt dann **Elementarfluss** von N .

Insbesondere ist f konstant 0 in jedem Netzwerk ein Fluss der Stärke 0. Dieser ist jedoch in der Regel uninteressant. Stattdessen stellt sich die Frage, wie groß denn so ein Fluss höchstens werden kann. Hierfür werden *Schnitte* zur Hilfe gezogen, die den Graphen in zwei Teile zerlegen, wobei ein Teil s und ein Teil t enthält. Dann kann man bestimmen, wie viel Fluss über die Kanten fließt, die über die Schnittgrenze gehen.

3.10. Ein **Schnitt** in N ist eine Teilmenge X von V mit $s \in X$ und $t \notin X$. Die **Kapazität** $\|X\|$ ist die Zahl $c(E_D^+(X))$.

bild 3.7

Die Stärke eines Flusses lässt sich an den Zwischenkanten eines beliebigen Schnittes x von N ablesen.

$$\|f\| = \sum_{x \in X} f(E_D^+(x)) - \sum_{x \in X} f(E_D^-(x)) \quad (3.13)$$

Dies gilt aufgrund der KIRCHHOFF-Bedingung. Wenn man sich für $x \neq s$ diese Summe anschaut, tritt im vorderen Teil immer die Summe des eingehenden Flusses auf, während hinten die Summe des ausgehenden Flusses auf. Für Flüsse innerhalb von X heben sich diese Terme auf, also verbleibt ausschließlich die Stärke des Flusses. Man kann dies auch noch feingranularer betrachten:

$$= \sum_{x \in X} \sum_{\substack{y \in V \\ xy \in E}} f(xy) - \sum_{x \in X} \sum_{\substack{y \in V \\ yx \in E}} f(yx) \quad (3.14)$$

Sobald man nur die Kanten betrachtet, die beide Endpunkte in X haben, sieht man sofort, dass der Fluss für diese immer einmal in der vorderen und einmal in der hinteren Summe auftritt. Am Ende verbleibt nur der Fluss über Kanten, die mit einer Ecke $x \in X$ beginnen und in einer Ecke $y \in V \setminus X$ enden:

$$= \sum_{x \in X} \sum_{\substack{y \in V \setminus X \\ xy \in E}} f(xy) - \sum_{x \in X} \sum_{\substack{y \in V \setminus X \\ yx \in E}} f(yx) \quad (3.15)$$

$$= f(E_D^+(X)) - f(E_D^-(X)) \quad (3.16)$$

Man kann die Stärke eines Flusses nicht nur an bestimmten Schnitten, sondern an jedem beliebigen Schnitt ablesen – insbesondere von dem einfachen Fluss $X = \{s\}$. Das bedeutet aber auch, dass die Kapazitäten der Schnitte obere Schranken für Flüsse sind!

bild 3.8

$$\|f\| = f(E_D^+(X)) - \underbrace{f(E_D^-(X))}_{\geq 0} \quad (3.17)$$

$$\leq f(E_D^+(X)) \quad (3.18)$$

$$\stackrel{(i)}{\leq} c(E_D^+(X)) \quad (3.19)$$

$$\implies \max \{ \|f\| \mid f \text{ ist Fluss von } N \} \leq \min \{ \|X\| \mid X \text{ ist Schnitt von } N \} \quad (3.20)$$

Satz 3.1 (Max-Flow-Min-Cut-Theorem, FORD-FULKERSON). *In jedem Netzwerk N gilt:*

$$\max \{ \|f\| \mid f \text{ Fluss von } N \} = \min \{ \|X\| \mid X \text{ Schnitt von } N \} \quad (3.21)$$

Beweis. Aus der o. g. Vorüberlegung folgt bereits „ \leq “. Die Richtung „ \geq “ benötigt noch weitere Überlegungen. \square

bild 3.9

Backref
Traveling-
Salesman-
Problem
zu Ab-
schnitt 4.3

A. Lektüre

Lehrbücher von:

- R. Diestel
- D. West
- J. A. Bondy
- U. S. R. Murty
- C. Berge
- F. Harary

Wikipedia ist auch okay.

B. Sätze, die in der Übung bewiesen werden

Satz B.1 (TUZA). *Für $k \geq 2$ ist ein Graph k -färbbar, wenn er keine Kreise der Länge l kongruent $+1$ modulo k besitzt.*

Stichwortverzeichnis

- Abstand, 8
- arc, 30
- aufspannend, 28
- Austauschlemma, 13
- Außengrad, 30
- Außenkante, 30

- Barriere, 23
- Basis, 15
- Baum, 6, 12
 - Spann-, 8
 - Tiefensuch-, 10
 - Zerlegungs-, 18
- Baumweite, 18
- Baumzerlegung, 18
- benachbart
 - Knoten, 3
- Berge, 22
- BFS, 9
- Breadth-First-Search, 9
- Breitensuchbaum, 9
- Breitensuche, 8, 9

- Clique, 19
- Courcelle, 19

- DFS, 10
- Differenz
 - symmetrisch, 20
- Digraph, 30
- disjunkt
 - Graph, 24

- Ecke, 3, 30
 - Endecke, 5
 - innere, 5
 - wesentlich, 25

- Elementarfluss, 32
- endlich, 3

- Faktor
 - f , 28
- faktorkritisch, 24
- Faktorsatz
 - f , 28
- Fluss, 31

- Gewicht
 - Gesamt, 17
- Grad, 28
- Graph, 3
 - aufspannend, 8, 28
 - disjunkt, 24
 - faktorkritisch, 24
- Greedy-Algorithmus, 13

- Hall, 21
- Hall-Bedingung, 21
- Heiratssatz, 21

- Innengrad, 30
- Innenkante, 30

- Kante, 3, 30
- Kantenmenge, 5
- Kapazität, 31
 - Schnitt, 32
- Kirchhoff, 31
- Knoten, 30
- Knotenmenge, 5
- Komponenten
 - Zusammenhangs, 6
- Kreis, 12
 - Hamilton, 22

- kontinuierlich gerichtet, 31
- kreisfrei, 12
- logic
 - monadic second order, 19
- Länge
 - Kreis, 12
 - Weg, 5
- Löschung, 5
- Matching, 20
 - größtes, 23
 - perfekt, 24
- Matroid, 15
- Max-Flow-Min-Cut-Theorem, 33
- Metrik, 8
- Min-Max-Theorem, 26
- monadic second order logic, 19

- Nachbar, 3
- Nachbarschaft, 20
- Netzwerk, 31

- Petersen-Graph, 3
- Polynomialzeitapproximationsschema, 17

- Quelle, 31

- Schnitt, 32
- Senke, 31
- Spannbaum, 8
 - minimalen Gewichtes, 13

- Stärke, 32

- Tasche, 18
- Teilgraph, 3
 - induziert, 4
- Tiefensuchbaum, 10
- Tiefensuche, 8, 10, 11
- Toncias, 22
- Traveling-Salesman-Problem, 15

- unabhängig, 15

- Verbesserungsweg, 21

- Wald, 6, 12
- Weg, 5
 - alternierend, 21
 - Verbesserungs-, 21
- Weite
 - Baum-, 18
 - Baumzerlegung, 18
- wesentlich
 - Ecke, 25

- Zerlegungsbaum, 18
- zh, 6
- zshgd, 6
- zugrundeliegender ungerichteter Graph,
 - 31
- zusammenhängend, 6
 - minimal, 6