Einführung in die diskrete Mathematik

Arthur Henninger

24. Oktober 2024

Inhaltsverzeichnis

Kapitel 1

Grundlagen

1. Vorlesung - 08.10.2024

Definition 1.1: Ungerichtete Graphen

Ein ungerichteter Graph ist ein Tripel (V, E, Ψ), wobei V, E endliche Mengen, $V \neq \emptyset$ und

$$\Psi : E \to \{x \subset V \,|\, |X| = 2\} =: \binom{n}{2}.$$

Definition 1.2: Gerichtete Graphen

Ein gerichteter Graph (Digraph) ist ein Tripel (V, E, Ψ), wobei V, E endliche Mengen, $V \neq \emptyset$ und

$$\Psi : E \to \{(v, y) \in V \times V | x \neq y\}.$$

Definition 1.3: Graph

Ein Graph ist ein gerichteter oder ungerichteter Graph.

Notation 1.1

Wir nennen V die Menge der Knoten (engl. "verticies") und E die Menge der Kanten (engl. "edges").

Beispiel 1.1 (Graphen)

ungerichteter bzw. gerichteter Graph:

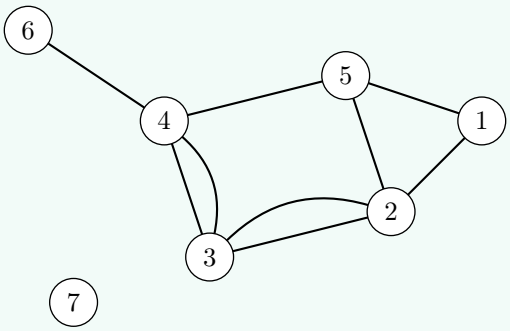


Abbildung 1.1: ungerichteter Graph

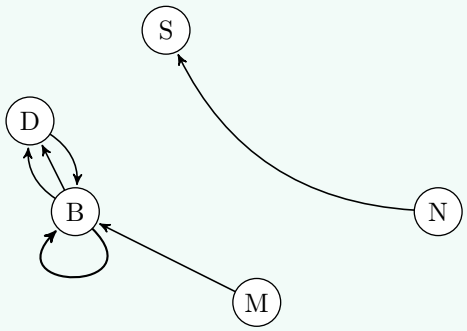


Abbildung 1.2: gerichteter Graph

Definition 1.4: parallele Kanten

Zwei Kanten $e, e' \in E$ heißen parallel, wenn $\Psi(e) = \Psi(e')$.

Definition 1.5: einfacher Graph

Ein Graph heißt einfach, wenn er keine parallelen Kanten besitzt.

Notation 1.2

In diesem Fall identifizieren wir $e \in E$ mit $\Psi(e)$. Der Graph (V, E, Ψ) reduziert sich zu $G = (V, E)$.

Notation 1.3 Sprachgebrauch

- $e = \{x, y\}$ oder $e = (x, y)$ Kante
- e verbindet x und y
- x und y sind benachbart/adjazent
- x ist Nachbar von y
- x und y sind mit e inzident
- $G = (V, E)$, $X, Y \subseteq V(G)$
  Ungerichtete Graphen:

$$
\begin{aligned}
E(X, Y) &:= \{\{x, y\} \in E(G) | x \in X \setminus Y \text{ und } y \in Y \setminus X\} \\
\delta(X) &:= E(X, V(G) \setminus X) \\
\delta(x) &:= \delta(\{x\}) \text{ für } x \in V(G) \\
|\delta(x)| &: \text{ Grad von } x.
\end{aligned}
$$

Gerichtete Graphen:

$$
\begin{aligned}
E^+(X,Y) &:= \{(x,y) \in E(G) | x \in X \setminus Y \text{ und } y \in Y \setminus X\} \\
\delta^+(X) &:= E^+(X, V(G) \setminus X) \\
\delta^-(X) &:= E^+(V(G) \setminus X, X) \\
\delta(X) &:= \delta^+(X) \cup \delta^-(X) \\
\delta^+(x) &= \delta^+(\{x\}) \\
\delta^-(x) &= \delta^-(\{x\}) \\
\delta(x) &= \delta(\{x\}) \\
\left|\delta^+(x)\right| &: \underline{\text{Ausgangsgrad}} \\
|\delta^-(x)| &: \underline{\text{Eingangsgrad}} \\
\delta^+(x) &: \underline{\text{ausgehende Kanten}} \\
\delta^-(x) &: \underline{\text{eingehende Kanten}}.
\end{aligned}
$$

- $\underline{K\text{-regulärer Graph}}$: $|\delta(x)| = K \forall x \in V(G)$.
- Ein Knoten vom Grad 0 heißt $\underline{\text{isolierter Knoten}}$.
- Falls mehrere Graphen betrachtet werden: G, H, F, füge Graphen als Index hinzu: $\delta_G(x), \delta_H(x), \ldots$

Satz 1.1

Für jeden Graphen $G = (V, E)$ gilt:

$$\sum_{x \in V(G)} |\delta(x)| = 2 \cdot |E| .$$

Korollar 1.2

In jedem Graphen ist die Anzahl an Knoten mit ungeradem Grad gerade.

Satz 1.3

Für jeden Digraphen $G = (V, E)$ gilt

$$\sum_{x \in V(G)} \delta^-(x) = \sum_{x \in V(G)} \delta^+(x).$$

Definition 1.6: Teilgraph

Ein Graph $H = (V(H), E(H))$ ist ein Teilgraph (Subgraph, Untergraph) eines Graphen $G = (V(G), E(G))$, falls

$$V(H) \subseteq V(G) \text{ und } E(H) \subseteq E(G).$$

Wir sagen auch: G enthält H (als Teilgraph).

- Falls $V(H) = V(G)$, so ist H ein aufspannender Teilgraph.
- Der Graph H ist induzierter Teilgraph von G, falls

$$V(H) \subseteq V(G) \text{ und } E(H) = \{\{x, y\} \in E(G) | x, y \in V(H)\} .$$

Bemerkung 1.1

Ein induzierter Teilgraph ist insbesondere durch die Knotenmenge festgelegt.

Notation 1.4

"H ist der von $V(H)$ induzierte Teilgraph von G"

$$H := G[V(H)].$$

Für $x \in V(G)$ definiere:

$$G - x := G[V(G) \setminus \{x\}].$$

Für $e \in E(G)$ definiere:

$$G - e := (V(G), E(G) \setminus \{e\}).$$

Für $e \in \binom{V(G)}{2}$ mit $e \notin E(G)$.

$$G + e. = (V(G), E(G) \cup \{e\}).$$

Definition 1.7: vollständiger Graph

$$\left(V, \binom{V}{2}\right) := K_n, \text{ falls } |V| = n.$$

Definition 1.8: Isomorphie

Zwei Graphen G und H heißen isomorph, falls es eine Bijektion $\varphi : V(G) \to V(H)$ gibt, sodass

$$\varphi(\{x, y\}) := \{\varphi(x), \varphi(y)\}$$

eine Bijektion zwischen $E(G)$ und $E(H)$ darstellt. φ ist Isomorphismus. Alternativ kann auch

$$\{x, y\} \in E(G) \iff \{\varphi(x), \varphi(y)\} \in E(H)$$

gelten.

> **Notation 1.5** isomorphe Graphen
> $G \cong H$ oder $G = H$

Bemerkung: Für $G = (V(G), E(G))$ und $H = (V(H), E(H))$ müssen $\varphi : V(G) \to V(H)$ und $\sigma : E(G) \to E(H)$ "kompatible" Bijektionen sein.

> **Notation 1.6** Sprechweise
> F ist Teilgraph von G meint: F ist isomorph zu einem Teilgraphen von G

[. Vorlesung - 2

]10.10.2024

Feststellungen:

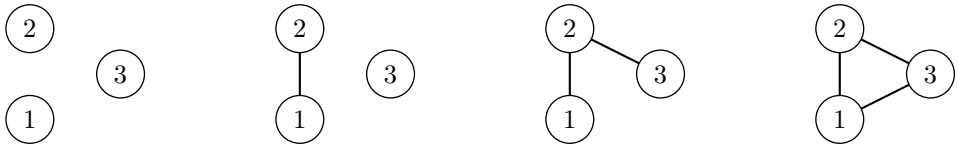
- $\varphi : V(g) \to V(H)$ Isomorphismus $\implies g - x \cong H - \varphi(x \forall x \in V(G))$ (Isomorphie erhält Teilgraphen)
- Isomorphieproblem: Sind G und H isomorph? Ungelöst, d.h. kein polynomieller Algorithmus (polynomielle Laufzeit in den Kanten) bekannt.
  - $O(n^2 \cdot n!) \approx O(2^{n \log n})$ trivial
  - schnellster bekannter Algorithmus für Graphenisomorphie: Babai (2025) Laufzeit $O(2^{\text{poly}(\log n)})$
- Ungelöstes Problem: Wenn ich $\varphi : V(G) \to V(H)$ finde, sodass $G - x \cong H - \varphi(x) \forall x \in V(G)$ gilt, ist dann $G \cong H$. (Außer im Fall der Graphen mit 2 Punkten, die in G verbunden und in H nicht verbunden sind.)
  - andere Formulierung: G Graph, betrachte Multimenge M aller Graphen $G - x, x \in V(G)$. Behauptung: G ist der einzige Graph mit dieser Multimenge (mit Wiederholung) an Teilgraphen, falls $|V(G)| \geq 3$.
  - Name: Graph Reconstruction Problem (scheint offensichtlich zu gelten)
- Ein Isomorphismus von G nach G heißt Automorphismus. Die Menge aller Isomorphismen eines Graphen bildet seine Automorphismengruppe. Jede existente Gruppe ist die Automorphismengruppe eines Graphen.

Nicht isomorphe einfache ungerichtete Graphen:

- $n = 1$:
  (1)

- $n = 2$:

- $n = 3$:



- $n = 4$: 11 Graphen

Wie lange dauert die Erzeugung:

- $2^{\binom{n}{2}} \cdot n! \cdot n^2$ (alle probieren und jeweils Isomorphietest machen)
- Besser: 2^{n-1}
  - Idee: Kanonische Repräsentation: den aller isomorphen Graphen, dessen Adjazenzmatrix als Binärzahl minimal ist
  - Dann kann man bei jedem Graphen unabhängig von anderen Graphen nachtesten, ob es sich bereits um die kanonische Repräsentation handelt.
  - Bemerkung: Es ist im Mittel recht einfach zu testen, ob der Graph die kanonische Repräsentation darstellt (indem man durch Zeilen- oder Spaltenpermutationen versucht, die Binärzahl zu verkleinern). Im Extremfall müssen dennoch alle Spalten- und Zeilenpermutationen getestet werden, dies tritt aber selten auf. Der Algorithmus taugt daher nur zur Findung aller nicht-isomorphen einfachen ungerichteten Graphen gleichzeitig. Insbesondere wird aus einer Repräsentation nicht die kanonische erzeugt, sonst wäre hierdurch ein einfacher Isomorphietest möglich.

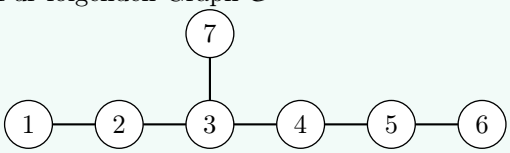
Beispiel 1.2

Automorphismengruppe von G: $\mathrm{Aut}(G)$

- $$|\mathrm{Aut}(\textsc{Graph, der Würfel repräsentiert})| = 48.$$

$$|\mathrm{Aut}(3 \textsc{ Punkte in Reihe})| = |\mathrm{Aut}(2 \textsc{ Punkte in Reihe})| = 2.$$

Für folgenden Graph G



ist $|V(G)| > 1$ aber $|\mathrm{Aut}(G)| = 1$.

Satz 1.4

Es gibt immer mindestens

$$\frac{2^{\binom{n}{2}}}{n!}$$

viele nicht-isomorphe einfache ungerichtete Graphen und mindestens

$$\frac{4^{\binom{n}{2}}}{n!}$$

viele nicht isomorphe einfache gerichtete Graphen auf n Knoten.

Beweis: Betrachte K_n. Dieser hat $\binom{n}{2}$ viele Kanten. Jede Teilmenge der Kantenmenge liefert einen Graphen. Dies sind $2^{\binom{n}{2}}$ Graphen. Maximal $n!$ (Anzahl der Permutationen) davon sind isomorph $\Longrightarrow$ Es gibt mindestens $\frac{2^{\binom{n}{2}}}{n!}$ nicht isomorphe einfache Graphen.
Analog $\frac{4^{\binom{n}{2}}}{n!} = \frac{2^{2\binom{n}{2}}}{n!}$ im gerichteten Fall. □

Man kann zeigen: Es gibt genau $(1 + o(1)) \cdot \frac{2^{\binom{n}{2}}}{n!}$ einfache ungerichete bzw. $(1 + o(1)) \cdot \frac{4^{\binom{n}{2}}}{n!}$ einfache gerichtete Graphen.
"Fast alle Graphen haben eine triviale Automorphismengruppe"

Definition 1.9: Kantenzug, Weg

Ein Kantenzug in einem Graphen ist eine Folge $x_1, e_1, x_2, e_2, \ldots, e_{k-1}, x_k$ mit $k \geq 1$ und $e_i = \{x_i, x_{i+1}\} \in E(G)$ bzw. $e_i = (x_i, x_{i+1}) \in E(G)$. Falls $x_1 = x_k$, so ist der Kantenzug geschlossen.
Falls in einem Kantenzug $x_1, e_1, \ldots, e_{k-1}, x_k$ alle Knoten paarweise verschieden sind, so ist der Graph $P = (\{x_1, \ldots, x_k\}, \{e_1, \ldots, e_{k-1}\})$ ein Weg.

Notation 1.7 Sprachgebrauch

P ist ein $x_1 - x_k$-Weg, P verbindet x_1 mit x_k. x_1, x_k werden die Endknoten von P genannt. Alle anderen Knoten, d.h. $x_2, \ldots, x_{k-1}$ sind die inneren Knoten von P.
Für $x, y \in V(P)$ ist $P_{[x,y]}$ der eindeutige Teilweg in P mit Endknoten x und y.

Lemma 1.5

Es gibt genau dann einen $x - y$-Weg in einem Graphen, wenn es einen $x - y$-Kantenzug gibt.

Beweis aus AlMa I: .

- Per Definition ist ein Weg ein Kantenzug
- Ein Kantenzug kann durch entfernen der Kanten und Knoten zwischen sich wiederholenden Knoten zu einem Weg verkürzt werden.

□

Definition 1.10: Kreis

Falls in einem geschlossenen Kantenzug $x_1, e_1, x_2, \ldots, e_k, x_1$ gilt, dass $x_i \neq x_j$ für $1 \leq i < j \leq k$ so ist der Graph $(\{x_1, \ldots, x_k\}, \{e_1, \ldots, e_k\})$ ein *Kreis*, falls $k \geq 3$, im ungerichteten Fall bzw. $k \geq 2$ im gerichteten Fall. Die *Länge* eines Kreises oder Weges ist die Anzahl seiner Kanten. Außerdem muss $e_1 \neq e_k$ gelten.

---

3. Vorlesung - 15.10.2024

Lemma 1.6

Es sei G ein ungerichteter einfacher Graph, in dem jeder Knoten Grad $\geq k$ hat. Dann enthält G einen Weg der Länge $\geq k$. Falls $k \geq 2$ so enthält G einen Kreis der Länge $\geq k + 1$.

Beweis: Sei P ein längster Weg in G, x einer seiner Endknoten.

$\Longrightarrow$ alle Nachbarn von v liegen in $V(P)\setminus\{x\}$

$\Longrightarrow$ $|\delta(x)| \leq |V(P)| - 1$, es gilt $k \leq |\delta(x)|$

$\Longrightarrow$ $|V(P)| - 1 \geq k$ d.h. Länge des Weges ist $\geq k$

Wähle $a \in V(P)$, sodass $\{x, a\} \in E(P)$ und $P_{[a,x]}$ ist längstmöglich.
$\Longrightarrow$ $P_{[a,x]} + \{x, a\}$ bilder Kreis der Länge $\geq k + 1$ □

Definition 1.11: Extreme Elemente

Sei E Familie von Mengen oder Graphen. $F \in E$ ist *minimales Element*, falls keine echte Teilmenge bzw. kein echter Teilgraph von F in E enthalten ist. Analog definieren wir maximale Elemente.

Definition 1.12: zusammenhängend

Sei G einungerichteter Graph. G heißt *zusammenhängend*, falls es für je zwei Knoten $x, y \in V(G)$ einen $x-y$-Weg in G gibt.
Die maximalen zusammenhängenden Teilgraphen von G heißen *Zusammenhangskomponenten*. Ein Knoten $x \in V(G)$ heißt *Artikulationsknoten* (trennender Knoten), falls $G - x$ mehr Zusammenhangskomponenten hat als G hat.
Eine Kante $e \in E(G)$ heißt *Brücke*, falls $G - e$ mehr Zusammenhangskomponenten als G hat.

Satz 1.7

(a) Ein ungerichteter Graph G ist genau dann zusammenhängend, falls $\delta(X) \neq \emptyset \, \forall \emptyset \subsetneq X \subsetneq V(G)$.

(b) Sei G gerichteter Graph und $r \in V(G)$. Genau dann gibt es einen $r-x$-Weg für jedes $x \in V(G)$, falls $\delta^+(X) \neq \emptyset \, \forall X \subsetneq V(G)$ mit $r \in X$.

Beweis: Prop 3.13 und 3.14 in AlMa I □

Definition 1.13

- Ein ungerichteter einfacher Graph heißt *Wald*, falls er keinen Kreis enthält.
- Ein *Baum* ist ein zusammenhängender Wald.
- Ein *spannender Baum* ist ein spannender Teilgraph, der Baum ist.
- Ein *Blatt* ist ein Knoten vom Grad 1 in einem Baum.

Frage 1

Wie viele nicht-isomorphe Bäume auf n Knoten gibt es?

Lösung

Bäume liegen meist nicht in der trivialen Automorphismengruppe (Gibt es zum Beispiel 2 Blätter an einem Knoten, kann man diese aufeinander mappen).

Proposition 1.8

Jeder Baum mit mindestens zwei Knoten hat mindestens 2 Blätter.

Beweis: AlMa I □

Satz 1.9

Sei G ungerichteter einfacher Graph auf n Knoten. Dann sind äquivalent:

(a) G ist ein Baum

(b) zwischen je 2 Knoten in G gibt es einen eindeutigen Weg

(c) G ist minimaler Graph mit Knotenmenge $V(G)$ und $\delta(X) \neq \emptyset \, \forall \emptyset \subsetneq X \subsetneq V(G)$.

(d) G ist minimaler zusammenhängender Graph auf $V(G)$

(e) G ist maximaler kreisfreier Graph

(f) G hat $n-1$ Kanten und ist kreisfrei

(g) G hat $n-1$ Kanten und ist zusammenhängend.

Beweis: Satz 3.20 in AlMa I □

Korollar 1.10

Ein Wald auf n Knoten mit k Zusammenhangskomponenten hat $n-k$ Kanten. (Lemma 3.19b AlMa I)

Beweis: Jede Zusammenhangskomponente ist Baum mit n_i Knoten $i = 1, \dots, k$. Diese haben zusammen

$$\sum_{i=1}^{k} (n_i - 1) = -k + \sum_{i=1}^{k} n_i = -k + n$$

Kanten. □

Korollar 1.11

Ein ungerichteter Graph ist genau dann zusammenhängend, wenn er einen spannenden Baum enthält.

Beweis: Wegen (d) $\implies$ (a) in Satz 1.9. □

Definition 1.14

- Für einen Digraphen G ist der *zugrunde liegende ungerichtete Graph* derjenige Graph G', den man aus G erhält, indem man jedes $(x, y) \in E(G)$ durch $\{x, y\} \in E(G')$ ersetzt (parallele Kanten können entstehen). Umgekehrt heißt G *Orientierung* von G'.
- Ein Digraph heißt *zusammenhängend*, falls sein zugrundeliegender ungerichteter Graph zusammenhängend ist.
- Ein Digraph heißt *Branching*, falls er keine Kreise enthält und $|\delta^-(x)| \leq 1 \, \forall x \in V(G)$.
- Der einem Branching zugrunde liegende ungerichtete Graph ist ein Wald.
- Eine *Arboreszenz* ist ein zusammenhängendes Branching. Der einer Arboreszenz zugrunde liegende ungerichtete Graph ist ein Baum $\implies$ Bei n Knoten hat die Arboreszenz $n-1$ Kanten $\implies$ es gibt genau einen Knoten r mit $\delta^-(r) = \emptyset$. Der Knoten r heißt *Wurzel* der Arboreszenz. Ein Knoten v mit $\delta^+(v) = \emptyset$ heißt *Blatt*.

Beispiel 1.3

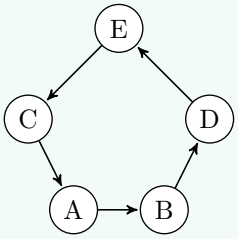


Abbildung 1.3: Kreis

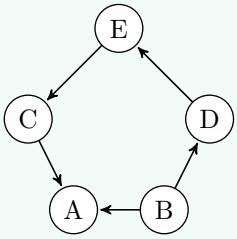


Abbildung 1.4: kein Kreis

Satz 1.12

Sei G Digraph mit n Knoten und $r \in V(G)$. Dann sind äquivalent:

(a) G ist Arboreszenz mit Wurzel r

(b) G ist Branching mit $n-1$ Kanten und $\delta^-(r) = \emptyset$

(c) G hat $n-1$ Kanten und jeder Knoten ist von r aus erreichbar

(d) Jeder Knoten ist von r aus erreichbar, aber das Entfernen einer beliebigen Kante zerstört diese Eigenschaft.

(e) G ist kantenminimaler Graph mit $\delta^+(X) \neq \emptyset \, \forall X \subsetneq V(G), r \in X$

(f) $\delta^-(r) \neq \emptyset$ und $\forall v \in V(G)$ gibt es eindeutigen $r-v$-Kantenzug

(g) $\delta^-(v) = \emptyset$ und $|\delta^-(v)| = 1 \, \forall v \in V(G) \setminus \{r\}$ und G enthält keinen Kreis.

Beweis:

- (a) $\Longrightarrow$ (b) zusammenhängendes Branching, zugrunde liegender Graph ist Baum $\Longrightarrow \delta^-(r) = \emptyset, n-1$ Kanten.
- (b) $\Longrightarrow$ (c) $n-1$ Kanten, $\forall v \neq r$ gilt $|\delta^-(v)| = 1 \Longrightarrow$ "verfolge" rekursiv die eingehenden Kanten zurück. Die Folge muss in r enden und wir haben einen $r-v$-Weg gefunden.
- (c) $\Longrightarrow$ (d) Folgt aus Satz 1.9 (d) $\Longleftrightarrow$ (g)
- (d) $\Longrightarrow$ (e) Folgt aus Satz 1.7 (b)
- (e) $\Longrightarrow$ (f) $\delta^-(r) = \emptyset$ folgt aus Kantenminimalität, Satz 1.7 $\Longrightarrow$ $r-v$-Kantenzug existiert $\Longrightarrow$ $r-v$-Weg. Sei P ein $r-v$-Weg. Sei Q ein anderer $r-v$-Kantenzug $\Longrightarrow$ Q enthält mindestens eine Kante, die nicht in P enthalten ist. Sei e letzte Kante entlang des $r-v$-Kantenzugs Q, die nicht in P liegt. $\Longrightarrow$ e kann entfernt werden ohne die Eigenschaft in (e) zu zerstören.
- (f) $\Longrightarrow$ (g) $\forall v \in V(G) \setminus \{r\}$ ist $|\delta^-(v)| \geq 1$, da sonst v von r nocht erreichbar wäre.
  Annahme: $|\delta^-(v)| \geq 2 \Longrightarrow \exists (a,v), (b,r)$. $\exists r-a$-Weg und $r-b$-Weg. $\Longrightarrow \exists 2$ veschiedene $r-v$-Kantenzüge
  (Alternative: $\left|\delta^{-1}(v)\right| = 1 \, \forall v \in V_(G) \setminus \{r\}$, Kreis $\Longrightarrow$ r ist nicht enthalten, d.h. $\nexists r-v$-Kantenzug für einen Knoten des Kreises)
- (g) $\Longrightarrow$ (a) Nach Definition ist G Branching mit $n-1$ Knoten. Satz 1.9 (f) $\Longrightarrow$ (G) Arboreszenz.

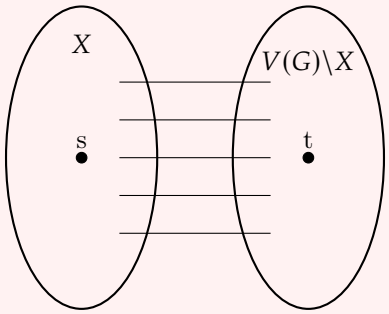
□

Übung: Reihenfolge ändern und Implikationen zeigen

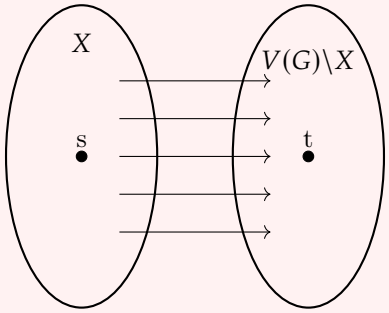
4. Vorlesung - 17.10.2024

Definition 1.15: Schnitt

Ein *Schnitt* in einem ungerichteten Graphen G ist eine Kantenmenge $\delta(X)$ für ein $\emptyset \neq X \subsetneq V(G)$. Es ist $\delta(x) = \delta(V(G)\backslash X)$.



G Digraph: $\delta^+(X)$ ist *gerichteter Schnitt*, falls $\emptyset \neq X \subsetneq V(G)$ und $\delta^-(X) \neq \emptyset$.



- Eine Kantenmenge $F \subseteq E(G)$ *trennt* Knoten $s \in V(G)$ und $t \in V(G)$, falls es einen $s-t$-Weg in G aber nicht in $(V(G), E(G)\backslash F)$ gibt.
- Ein $s-t$-Schnitt im ungerichteten Graphen ist ein Schnitt $\delta(X)$ mit $s \in X, t \notin X$.
- Ein $s-t$-Schnitt im gerichteten Graphen ist ein Schnitt $\delta^+(X)$ mit $s \in X, t \notin X$.
- Teilgraphen in gerichteten Graphen heißen *ungerichteter Weg* bzw. *ungerichteter Kreis*, falls sie ein Weg bzw. Kreis im zugrunde liegenden ungerichteten Graphen sind.
- Ungerichteter Schnitt: Schnitt im zugrunde liegenden ungerichteten Graphen.

Lemma 1.13

Sei G ein Digraph und $e \in E(G)$. Sei e blau gefärbt, alle anderen Kanten seien rot, blau oder grün gefärbt. Dann gilt genau ein der beiden folgenden Aussagen:

(a) Es gibt einen ungerichteten Kreis, der e enthält und dessen weitere Kanten nur rot oder blau sind, wobei die blauen Kanten alle *gleich orientiert* sind

(b) Es gibt einen ungerichteten Schnitt, der e enthält und dessen weitere Kanten nur grün oder blau sind, wobei alle blauen Kanten alle *gleich orientiert* sind.

Beweis: Algorithmisch: Sei $e = (x, y)$. Markiere y. Regel: Ist v bereits markiert und w noch nicht, so markieren wir w, falls eine blaue Kante (v, w) oder eine rote Kante (v, w) oder (w, v) existiert. Enden: Kein weiterer Knoten

kann markiert werden. Wenn w markiert wird: merken verantwortlichen Knoten v als $\text{pred}(w)$.
Ist x am Ende markiert oder nicht?

1. Fall: x markiert: Folge der pred-Funktion ausgehend von x

$$x, \text{pred}(x), \text{pred}(\text{pred}(x)), \ldots \implies \text{alle Kanten rot oder blau.}$$

2. Fall: x ist nicht markiert. Betrachte Menge R aller markierten Knoten $\implies \delta^+(R) \cup \delta^-(R)$ erfüllt (b).

Angenommen, es gibt sowohl einen ungerichteten Kreis C wie in (a) als auch einen Schnitt $\delta^+(R) \cup \delta^-(R)$ wie in (b). Schnitt beider Kantenmengen enthält nur blaue Kanten und e, haben alle dieselbe Orientierung in C und im Schnitt. Widerspruch, da mindestens eine weitere Kante bezüglich des Kreises gleiche Orientierung hat und im Schnitt liegt.
Intuitiv: e führt von einem Teil des Schnitts in einen anderen, der Kreis geht durch beide Schnitte, muss also zurückführen. Die rückführende Kante müsste in beide Richtungen orientiert sein (zurück für Kreis, hin für Schnitt (da in Schnitt und Kreis jeweils gleichorientiert)). □

Definition 1.16: starker Zusammenhang

Ein Digraph G heißt *stark zusammenhängend*, falls es für je zwei $s, t \in V(G)$ einen $s-t$-Weg gibt. Die *starken Zusammenhangskomponenten* sind die maximalen stark zusammenhängenden Teilgraphen.

Korollar 1.14

In einem Digraphen ist jede Kante entweder in einem Kreis oder in einem gerichteten Schnitt enthalten.

Beweis: Lemma 1.13: Färbe alle Kanten blau. □

Korollar 1.15

In einem Digraphen G sind äquivalent:

(a) G ist stark zusammenhängend

(b) G enthält keinen gerichteten Schnitt

(c) G ist zusammenhängend und jede Kante von G liegt in einem Kreis.

Beweis:

- (a) $\implies$ (b): Angenommen, G enthält gerichteten Schnitt, d.h. $X \subsetneq V(G), X \neq \emptyset, \delta^-(X) = \emptyset$. Dann gilt: $\delta^+(V(G) \backslash X) = \emptyset \implies$ wegen 1.7 (b) gibt es Knoten x, y sodass kein $x-y$-Weg existiert.
- (b) $\implies$ (c): Korollar 1.14 $\implies$ Jede Kante liegt in einem Kreis, Annahme nicht zusammenhängend $\exists \emptyset \subsetneq X \subsetneq V(G)$ sodass $\delta^-(X) = \emptyset$ Widerspruch.
- (c) $\implies$ (a): Sei $r \in V(G)$ beliebig. Behauptung: $\exists\, r-x$-Weg in $G \, \forall x \in V(G)$. Angenommen nicht, d.h. $\exists\, x \in V(G)$, aber keinen $r-x$-Weg. Nach Satz 1.7 (b) gibt es $X \subsetneq V(G)$ mit $\delta^+(X) = \emptyset$ und $r \in X$.
  G ist zusammenhängend $\implies \delta^-(X) \neq \emptyset$. Sei $e \in \delta^-(X)$. e ist Kante eines Kreises $\implies$ Widerspruch

□

Korollar 1.16

Ein Digraph ist genau dann stark zusammenhängend, wenn es für jeden Knoten $r \in V(G)$ eine aufspannende Arboreszenz mit Wurzel r gibt.

Beweis: Sei G stark zusammenhängend $\implies \exists\, r-x$-Weg für alle $x \in V(G)$. Wähle minimalen Teilgraphen mit dieser Eigenschaft. Nach Satz 1.12 (d) $\implies$ (a) ist dieser Teilgraph eine aufspannende Arboreszenz mit Wurfel r.
Satz 1.12 (a) $\implies$ (d): $\exists\, r-x$-Weg für alle $x \in V(G)$ □

Definition 1.17: azyklische Digraphen

Ein Digraph heißt *azyklisch*, wenn er keinen Kreis enthält.

Frage 2

G Graph, n Knoten, linear viele Kanten in n, die in linearer Zeit gespeichert werden sollen (Matrix mit 0en initalisieren zählt auch). Speichere den Graphen, sodass folgende Frage in konstanter Zeit beantwortbar ist:

- Sind die Knoten i und j mit $1 \leq i < j \leq n$ durch eine Kante in G verbunden?

5. Vorlesung - 22.10.2024

Solution: Wir initialisieren die Matrix nicht. Wir nummerieren die Kanten durch und schreiben statt einer 1 jeweils die Nummer. Wenn dann eine Kante geprüft wird, lesen wir die Nummer aus der Matrix aus und gucken in der Edge-Liste, ob an der Stelle tatsächlich die Kante steht.

Definition 1.18: azyklisch

Ein Digraph heißz *azyklisch*, wenn er keine Kreise enthält.

Korollar 1.17

Folgende Aussagen über einen Digraphen sind äquivalent:

(a) G ist azyklisch

(b) Jede Kante gehört zu einem gerichteten Schnitt

(c) Die starken Zusammenhangskomponenten von G enthalten je nur 1 Knoten

Beweis:

- (a) $\iff$ (b): Korrolar 1.14 Jede Kante ist entweder in einem Schnitt oder in einem Kreis
- (a) $\implies$ (c): Sei H starke Zusammenhangskomponente von G. Jede Kante in H liegt in einem Kreis $\implies$ H enthält nur einen Knoten
- (c) $\implies$ (a) Annahme $\exists$ Kreis $\implies$ dieser wäre Teil einer starken Zusammenhangskomponente, $\lightning$ 1 Knoten pro Zusammenhangskomponente

□

Lemma 1.18

Jeder azyklische Digraph besitzt einen Knoten x mit $\delta^+(x) = \emptyset$.

Beweis: Wähle Weg P maximaler Länge. O.B.d.A. Länge ≥ 1. Sei x Endkonten von P mit $\delta^-_P(x) = 1$ (Endknoten in den eine Kante hinführt). $\implies$ $\delta^+(x) = \emptyset$, sonst $\exists (x, y) \in E(G)$ mit $y \in V(P) \setminus \{x\}$ wegen Maximalität von P. $\implies$ $P_{[y,x]} + \{x, y\}$ ist Kreis $\lightning$. □

Definition 1.19: topologische Ordnung

Die *topologische Ordnung* eines Digraphen G ist die Ordnung der Knoten $v_1, v_2, \ldots, v_n$ in $V(G)$, sodass gilt $\forall (v_i, v_j) \in E(G) \implies i < j$.

Satz 1.19

Ein Digraph besitzt genau dann eine topologische Ordnung, wenn er azklisch ist.

Beweis: $\Longrightarrow$ Kreis besitzt keine topoligische Ordnung.

$\Longleftarrow$ $|V(G)| = 1$ trivial. Sonst habe G $n \geq 2$ Knoten. Sei x mit $\delta^+(x) = \emptyset$. Sei $v_1, \ldots, v_{n-1}$ topolische Ordnung von $G - x$. Dann ist $v_1, \ldots, v_{n-1}, x$ eine topologische Ordnung von G.

□

Definition 1.20: Adjazenzmatrix

Eine *Adjazenzmatrix* für $G = (V, E)$ ist

$$A = (a_{x,y})_{x,y \in V(G)} = \begin{cases} 1 \text{ falls } \{x, y\} \in E(G) \ni (x, y) \\ 0 \text{ sonst.} \end{cases}$$

Definition 1.21: Inzidenzmatrix

Die *Inzidenzmatrix* für $G = (V, E)$ ungerichtet bzw. gerichtetes G ist

$$a_{x,e} = \begin{cases} 1 \text{ falls } x \in e \\ 0 \text{ sonst} \end{cases}$$

$$a_{x,e} = \begin{cases} -1 \text{ falls } e \text{ in } x \text{ beginnt} \\ 1 \text{ falls } e \text{ in } x \text{ endet} \\ 0 \text{ sonst.} \end{cases}$$

Definition 1.22

- *dünner Graph*: $O(n^2)$ viele Kanten
- *Adjazenzlisten*: Merke in einer Liste für jeden Knoten die Menge seiner Nachbarn

Bemerkung 1.2

	Adjazenzmatrix	Adjazenzliste
Speicherbedarf	$O(n^2)$	$O(n + m)$
Kante einfügen	$O(1)$	$O(1)$
Kante entfernen	$O(1)$	$O(1)$
Knoten einfügen	$O(n)$	$O(1)$
Knoten entfernen	$O(n)$	$O(n)$
Test ob $\{x, y\} \in E$	$O(1)$	$O(n)$

Satz 1.20

Der Algorithmus Graph Scanning ist korrekt und kann mit Laufzeit $O(n + m)$ implementiert werden.

Beweis: Satz 3.22 in AlMa I. □

Definition 1.23

Wir sagen, ein Graphenalgorithmus hat lineare Laufzeit, wenn seine Laufzeit in $O(n + m)$ liegt.

Algorithm 1 Graph Scanning

Input: Einen Graphen G, ein Knoten $s \in V(G)$
Output: Die Menge $R \subseteq V(G)$ der von s aus erreichbaren Knoten und eine Menge $T \subseteq E(G)$, sodass (R, T) eine Arboreszenz btw. ein Baum mit Wurzel s ist.
$R := \{s\}, Q := \{s\}, T := \emptyset$
while $Q \neq \emptyset$ **do**
    wähle $v \in Q$ beliebig
    if $\exists e = (v, w)$ bzw. $e = \{v, w\}$ mit $w \in V(G) \backslash R$ **then**
        $R := R \cup \{w\}, Q := Q \cup \{w\}, T := T \cup \{e\}$
    else
        $Q := Q \backslash \{v\}$
    end if
end while

Satz 1.21
Die Zusammenhangskomponenten eines ungerichteten Graphen können in linearer Zeit bestimmt werden.

Beweis: Korollar 3.23 in AlMa I. □

Bemerkung 1.3

- *Tiefensuche*: LIFO-Stack: last-in-first-Out für Q, DFS (Depth-First-Search), DFS-Baum: (R, T)
- *Breitensuche*: FIFO-Stack: first-in-first-Out für Q, BFS (Breadth-First-Search), BFS-Baum: (R, T)
-

  $\text{dist}(v, w) :=$ Länge eines kürzesten Weges in G, $v, w \in V(G)$. ∞, falls kein $v - w$-Weg existiert.

Algorithm 2 DFS

Input: Einen Graphen G, ein Knoten $s \in V(G)$
Output: Die Menge $R \subseteq V(G)$ der von s aus erreichbaren Knoten
$R := \emptyset$
DFS-visit (w)
function DFS-VISIT(v)
    $R := R \cup \{v\}$
    for w mit $(v, w) \in E(G)$ bzw. $\{v, w\} \in E(G)$ **do**
        if $w \notin R$ **then**
            DFS-visit (w)
        end if
    end for
end function

Satz 1.22
Ein BFS-Baum enthält kürzeste Wege von einem Knoten s zu allen anderen von s aus erreichbaren Knoten. Die Werte $\text{dist}(s, w) \forall w \in V(G)$ können in $O(n + m)$ errechnet werden.

Beweis: AlMa I Satz 3.24 □

Definition 1.24: Pre- und Post-Order-Nummerierung

- *Pre-Order-Nummerierung*: Nummeriere Knoten in der Reihenfolge, in der DFS sie besucht (wenn hinzugefügt wird)
- *Post-Order-Nummerierung*: nummeriere Knoten in der Reihenfolge, in der DFS sie abarbeitet (wenn entfernt wird)

---

6. Vorlesung - 24.10.2024

Ziel: topologische Ordnung in $O(n+m)$. $\exists x$ mit $\delta^+(x) = \emptyset$, $\exists y$ mit $\delta^-(y) = \emptyset$.

Algorithm 3 Topologische Ordnung

1: **Input:** Ein Digraph G
2: **Output:** topologische Ordnung, falls existiert
3: $i := 0$
4: **while** $\exists v \in V(G)$ mit $\delta^-(v) = \emptyset$ **do**
5: $\quad i := i+1, v_i := v, G := G - v$
6: **end while**
7: **if** $V(G) \neq \emptyset$ **then** "G besitzt keine topologische Ordnung"
8: **end if**

Satz 1.23

Der Algorithmus 3 findet in linearer Zeit eine topologische Ordnung, falls G azyklisch ist.

Beweis:

- Korrektheit: Satz 1.19 und Lemma 1.18
- Laufzeit: Berechne $|\delta^-(v)|$ $\forall v \in V(G)$ in linearer Zeit. $R := \{v | \delta^-(v) = \emptyset\}$. Aktualisierung, falls v aus G entfernt wird: $\forall (v,w)$: erniedrige $|\delta^-(w)|$ um 1. Falls dabei $|\delta^-(w)| = 0$ wird $\implies$ für w zu R hinzu.

□

Korollar 1.18

In einem azyklischen Digraphen kann zu einem gegebenen Knoten s in linearer Zeit die Länge *längster Wege* zu allen anderen Knoten im Graphen bestimmt werden.

Beweis: Berechne in linearer Zeit topologische Ordnung. Dann

$$\begin{aligned} &L(v) := \text{Länge eines längsten } s-v\text{-Weges} \\ &L(v) := -\infty, \text{ falls } v \text{ vor } s \text{ in der topologischen Ordnung steht.} \\ &L(s) := 0 \\ &L(v) := 1 + \max_{(w,v)\in E(G)} L(w) \\ &\quad \implies \text{lineare Laufzeit .} \end{aligned}$$

□

Ziel: Berechne alle Brücken in einfachen, ungerichteten Graphen in $O(n+mc)$. Es gibt maximal so viele Brücken wie Knoten, denn ein DFS-Baum enthält alle Brücken.
Sei $G = (V, E)$ zusammenhängend, ungerichtet, einfach, T ein DFS-Baum für G mit PreorderNummerierung $p : V(G) \to \{1, \dots, |V(G)|\}$. für $x \in V(G)$ definiere:

$$\text{NACHFAHREN}(x) = \{y \in V(G) | \exists\, x-y\text{-Weg in } T \text{ mit Knoten} x = x_1, \dots, x_k = y \text{ sodass } p(x_i) < p(x_{i+1})\}\,.$$

Wir bemerken:

a) $x \in \text{NACHFAHREN}(x)$

b) $y \in \text{NACHFAHREN}(x), y \neq x \implies p(y) > p(x)$

Definiere den Lowpoint

$$\text{low}(y) := \min\{p(y), \min\{p(t) | \exists\, z \in \text{NACHFAHREN}(y) \text{ und } \{t,z\} \in E(G)\backslash E(T)\}\}.$$

Lemma 1.25

Sei $G = (v, E)$ ein zusammenhängender, einfacher, ungerichteter Graph und T ein DFS-Baum in G mit Preorder Nummerierung p. Eine Kante $\{x, y\} \in E(T)$ mit $p(y) > p(x)$ ist genau dann eine Brücke, wenn

$$\text{low}(y) > p(x).$$

Beweis: Sei $\{x, y\} \in E(T)$ mit $p(y) > p(x)$ eine Brücke in G.
Annahme:

$$\text{low}(y) \leq p(x) \implies \exists\, z \in \text{NACHFAHREN}(y) \text{ und } \{t,z\} \in E(G)\backslash E(T) \text{ mit } p(t) \leq p(x).$$

$\implies y - z$-Weg in $\text{NACHFAHREN}(y) + \{t,z\} + t - y$-Weg in $T + \{x,y\}$ ist Kreis $\lightning$ zu $\{x,y\}$ ist Brücke.
Sei $\{x, y\} \in E(T)$ mit $p(y) > p(x)$ und $\text{low}(y) > p(x)$
Annahme:

$$\begin{aligned} \{x,y\} \text{ ist keine Brücke} &\implies \{x,y\} \text{ ist Kante eines Kreises} \\ &\implies \exists \text{ Kante } \{z,t\} \text{ mit } z \in \text{NACHFAHREN}(y) \text{ und } t \notin \text{NACHFAHREN}(y). \end{aligned}$$

Wegen $\text{low}(y) > p(x)$ und $\text{low}(y) \leq p(t)$ gilt also $p(t) > p(x)$. Dann gilt $p(t) > p(z) \forall z \in \text{NACHFAHREN}(y)$.
$\implies p(t) > p(z) \implies$ DFS hätte wegen Kante $\{z,t\}$ den Knoten z "nach z" besucht. □

Algorithm 4 Berechne Brücken

```
1:  Input: ungerichteter, einfacher, zusammehängender Graph G = (V, E)
2:  Output: Menge B ⊆ E aller Brücken von G
3:  R := ∅, B := ∅, Knotenzähler := 0, s ∈ V beliebig
4:  DFS-visit (s, s)
5:  function DFS-visit(v, Vorgänger)
6:      R := R ∪ {v}, Knotenzähler := Knotenzähler + 1
7:      p(v) := Knotenzähler, low(v) := p(v)
8:      for w mit (v, w) ∈ E(G) bzw. {v, w} ∈ E(G) do
9:          if w ∉ R then
10:             DFS-visit (w)
11:             low(w) := min{low(v), low(w)}
12:             if low(w) < p(v) then
13:                 B := B ∪ {v, w}
14:             else
15:                 low(v) := min{low(v), p(w)} falls w ≠ Vorgänger
16:             end if
17:         end if
18:     end for
19: end function
```

Satz 1.26

Der Algorithmus Berechne Brücken bestimmt in linearer Zeit alle Brücken in einem einfachen, ungerichteten, zusammenhängenden Graphen.

Beweis:

- lineare Laufzeit fertig
- Reicht zu zeigen, $\text{low}(v)$ wird korrekt berechnet, denn dann sagt Lemma 25 Korrektheit des Algorithmus, da alle Brücken Kanten im DFS-Baum sind. Initialisierung: $\text{low}(x) = p(x)$ korrekt. x Blatt im DFS-Baum korrekt, x kein Blatt:

$$\begin{aligned}\text{low}(x) &= \min\{p(x), \min\{\text{low}(y), y \in \textsc{Nachfahren}(x)\}, \min\{p(t) | \{t,x\} \in E(G)\backslash E(T)\}\} \\ &= \min\{p(x), \min\{\text{low}(y), y \in \textsc{Nachfahren}(x), \{x,y\} \in E(T)\}, \min\{p(t) | \{t,x\} \in E(G)\backslash E(T)\}\}.\end{aligned}$$

□

Algorithm 5 Starke Zusammenhangskomponenten

Input: Digraph G
Output: Starke Zusammenhangskomponenten des Digraphen.
1. Tiefensuche mit Post-Order-Nummerierung Ψ (so lange mit noch nicht besuchten starten, bis alle Knoten eine Nummber haben)
2. Tiefensuche auf Knoten mit absteigendem Ψ, laufe duch Kanten in entgegengesetzter Richtung und berücksichtige keine schon benutzten Knoten. Jeder entstehende Teilgraph ist eine Zusammenhangskomponente

Kapitel 2

Bäume und Arboreszenzen

Kapitel 3

Kürzeste Wege

Kapitel 4

Netzwerkflüsse

Kapitel 5

Kostenminimale Flüsse

Kapitel 6

NP-Vollständigkeit