8 Diskrete Optimierung

Definition 8.1. Ein Graph G ist ein Paar $(V(G), E(G))$ bestehend aus

- einer endlichen Menge $V(G)$ von *Knoten* (oder *Ecken*)
- und einer Menge $E(G) \subseteq \binom{V(G)}{2}$ von *Kanten*.

Die *Ordnung* $n(G)$ von G ist $|V(G)|$ und die *Größe* $m(G)$ von G ist $|E(G)|$. Gilt $\{u, v\} \in E(G)$ für $u, v \in V(G)$, so nennt man u und v *adjazent/benachbart*. Gilt $u \in e$ für $u \in V(G)$ und $e \in E(G)$, so nennt man u und e *inzident*.

Für $u \in V(G)$ ist $N_G(u) = \{v \in V(G) \mid \{u, v\} \in E(G)\}$ die *Nachbarschaft von u in G* und $d_G(u) = |N_G(u)|$ der *Grad von u in G*. Der *Minimalgrad von G* ist $\delta(G) = \min\{d_G(u) \mid u \in V(G)\}$ und der *Maximalgrad von G* ist $\Delta(G) = \max\{d_G(u) \mid u \in V(G)\}$.

Sind G und H Graphen mit $V(H) \subseteq V(G)$ und $E(H) \subseteq E(G)$, so ist H *Teilgraph von G*. Gilt weiter $E(H) = \{e \in E(G) \mid e \in V(H)\}$, so ist H *ein induzierter Teilgraph von G*, genauer nennt man H den *von $V(H)$ in G induzierten Teilgraphen $G[V(H)]$*. Zwei Graphen G und H sind *isomorph*, falls eine Bijektion $f : V(G) \to V(H)$ existiert, die die Adjazenz respektiert, d.h.

$$\forall u \in V(G) : \forall v \in V(G) \setminus \{u\} : \{u, v\} \in E(G) \Leftrightarrow \{f(u), f(v)\} \in E(H).$$

Bemerkung 8.2. Die hier definierten Graphen sind *endlich, schlicht und ungerichtet*. Für $\{u, v\} \in E(G)$ schreiben wir kurz uv.

Lemma 8.3 (Handschlaglemma)**.** *Für jeden Graphen G gilt*

$$\sum_{u \in V(G)} d_G(u) = 2|E(G)|,$$

d.h. insb. die Anzahl der Ecken ungeraden Grades ist gerade.

□

Lemma 8.4. *Ist G ein Graph mit $m(G) > 0$, so existiert ein Teilgraph H von G mit*

$$\delta(H) > \frac{m(G)}{n(G)}.$$

Beweis: Wir betrachten folgenden Algorithmus.

Input: Ein Graph G mit $m(G) > 0$.
Output: Ein Teilgraph H von G.
begin
  $H \leftarrow G$;
  while $\delta(H) \leq \frac{m(G)}{n(G)}$ **do**
    Sei $u \in V(H)$ mit $d_H(u) \leq \frac{m(G)}{n(G)}$;
    $H \leftarrow H - u$;
  end
  return H;
end

Algorithm 1: Finde H in G.

□

Definition 8.5. Sei G ein Graph.

(i) Ein *Weg in G zwischen u_0 und u_l der Länge $l \in \mathbb{N}_0$* ist ein Teilgraph P von G mit

$$V(P) = \{u_0, \dots, u_l\} \text{ und } E(P) = \{u_{i-1}u_i \mid 1 \leq i \leq l\}.$$

Schreibweise: $P : u_0u_1 \dots u_l$.

(ii) G ist *zusammenhängend*, falls zwischen je zwei Knoten von G ein Weg in G existiert. Eine *Komponente* von G ist ein maximaler zusammenhängender Teilgraph von G.

(iii) Für $u, v \in V(G)$ ist der *Abstand* $\text{dist}_G(u, v)$ *in G zwischen u und v* gleich der minimalen Länge eines Weges in G zwischen u und v. Der maximale Abstand zweier Knoten in G ist der *Durchmesser* $\text{diam}(G)$ *von G*.

(iv) Ein *Kreis in G der Länge $l \in \mathbb{N}$ mit $l \geq 3$* ist ein Teilgraph C von G mit

$$V(C) = \{u_1, \dots, u_l\} \text{ und } E(C) = \{u_{i-1}u_i \mid 2 \leq i \leq l\} \cup \{u_1u_l\}.$$

Schreibweise: $C : u_1u_2 \dots u_lu_1$.

(v) Die minimale/maximale Länge eines Kreises in G ist *die Taillenweite/der Umfang $g(G)/c(G)$ von G*.

Lemma 8.6. *Sei G ein Graph.*

(i) Besitzt G einen Weg P zwischen u und v sowie einen Weg Q zwischen v und w, so besitzt G einen Weg R zwischen u und v der Länge höchstens $m(P) + m(Q)$, d.h.

$$d_G(u, w) \leq d_G(u, v) + d_G(v, w).$$

(ii) G besitzt genau dann einen Kreis der Länge höchstens $l + k$, wenn zwei Knoten u und v existieren, für die G zwei verschiedene Wege P und Q der Längen höchstens l und k zwischen u und v besitzt.

(iii) Enthält G einen Kreis, so gilt $g(G) \leq 2\text{diam}(G) + 1$.

□

Bemerkung 8.7. Die Komponenten von G sind die Äquivalenzklassen der Relation $\sim \in V(G)^2$ mit $u \sim v \Leftrightarrow \exists$ Weg in G zwischen u und v.

Lemma 8.8. *Jeder Graph G mit $\delta(G) \geq 2$ enthält einen Kreis der Länge $\geq \delta(G) + 1$.*

□

Definition 8.9. Ein Graph ohne Kreise ist ein *Wald*. Ein *Baum* ist ein zusammenhängender Wald.

Ein Knoten vom Grad höchstens 1 nennt man *Endknoten/Blatt*.

Eine Kante e eines Graphen G ist eine *Brücke* von G, falls $G - e = (V(G), E(G) \setminus \{e\})$ mehr Komponenten besitzt als G.

Lemma 8.10. *Ein Graph G ist genau dann ein Wald, wenn alle Kanten von G Brücken sind.*

□

Lemma 8.11. *Jeder Wald G hat genau $n(G)-m(G)$ Komponenten und mindestens $\Delta(G)$ Endknoten.*

Beweis: Induktion über die Kantenzahl. □

Satz 8.12. *Ist G ein Graph, so sind folgende Aussagen äquivalent.*

(a) G ist ein Baum.

(b) Zwischen je zwei Knoten von G existiert genau ein Weg.

(c) G ist ein minimal zusammenhängender Graph.

(d) G hat $n(G) - 1$ Kanten und keine Kreise.

□

Definition 8.13. Ein *Digraph* D ist ein Paar $(V(D), A(D))$ bestehend aus einer endlichen Menge $V(D)$ von *Knoten* und einer Menge

$$A(G) \subseteq V_D \times V_D \setminus \{(u,u) \mid u \in V_D\}$$

von *gerichteten Kanten* (oder *Bögen*).

$n(D)$	$=$	$\lvert V(D)\rvert$	*(Ordnung von D)*
$m(D)$	$=$	$\lvert A(D)\rvert$	*(Größe von D)*
$N_D^+(u)$	$=$	$\{v \in V(D) \mid (u,v) \in A(D)\}$	*(Außennachbarschaft von u in D)*
$d_D^+(u)$	$=$	$\lvert N_D^+(u)\rvert$	*(Außengrad von u in D)*
$N_D^-(u)$	$=$	$\{v \in V(D) \mid (v,u) \in A(D)\}$	*(Innennachbarschaft von u in D)*
$d_D^-(u)$	$=$	$\lvert N_D^-(u)\rvert$	*(Innengrad von u in D)*
$\Delta^+(D)$	$=$	$\max\{d_D^+(u) \mid u \in V(D)\}$	*(maximaler Außengrad von D)*
$\Delta^-(D)$			*(analog)*
$\delta^+(D)$			*(analog)*
$\delta^-(D)$			*(analog)*

Ein Digraph D' ist ein *Teildigraph* des Digraphen D, falls $V(D') \subseteq V(D)$ und $A(D') \subseteq A(D)$. Für $U \subseteq V(D)$ ist $D[U]$ mit $V(D[U]) = U$ und

$$A(D[U]) = \{(u,v) \in A(D) \mid u,v \in U\}$$

der in D von U *induzierte Teildigraph.*

Definition 8.14. (i) Ein *gerichteter Weg* in D von u_0 nach u_l der Länge $l \in \mathbb{N}_0$ ist ein Teildigraph P von D mit $V(P) = \{u_0, \ldots, u_l\}$ und $A(P) = \{(u_{i-1}, u_i) \mid 1 \le i \le l\}$. Schreibweise: $P : u_0 u_1 \ldots u_l$.

Ein *ungerichteter Weg* in D von u_0 nach u_l der Länge $l \in \mathbb{N}_0$ ist ein Teildigraph P von D mit $l+1$ verschiedenen Knoten $u_0, \ldots, u_l$ und l verschiedenen gerichtete Kanten $e_0, \ldots, e_{l-1}$ mit $e_i \in \{(v_i, v_{i+1}), (v_{i+1}, v_i)\}$ für $0 \le i \le l-1$.

(ii) D ist *stark zusammenhängend*, falls für alle $(u, v) \in V(D)^2$ ein gerichteter Weg in D von u nach v existiert. Die maximalen stark zusammenhängenden Teildigraphen von D sind die *starken Zusammenhangskomponenten* von D.

(iii) Ist D ein Digraph, so ist *der D unterliegende (ungerichtete) Graph* der Graph G mit $V(G) = V(D)$ und

$$E(G) = \left\{ uv \in \binom{V(G)}{2} \mid (u, v) \in A(D) \text{ oder } (v, u) \in A(D) \right\}.$$

D ist *schwach zusammenhängend*, falls G zusammenhängend ist. Die maximalen schwach zusammenhängenden Teildigraphen von D sind die *schwachen Zusammenhangskomponenten* von D.

(iv) Ein *gerichteter Kreis* in D der Länge $l \in \mathbb{N}$ mit $l \geq 2$ ist ein Teildigraph C von D mit $V(C) = \{u_1, \ldots, u_l\}$ und $E(C) = \{(u_{i-1}, u_i) \mid 2 \leq i \leq l\} \cup \{(u_l, u_1)\}$. Schreibweise: $C : u_1 u_2 \ldots u_l u_1$.

Ein *ungerichteter Kreis* in D der Länge $l \in \mathbb{N}$ mit $l \geq 2$ ist ein Teildigraph C von D mit l verschiedenen Knoten $u_1, \ldots, u_l$ und l verschiedenen gerichtete Kanten $e_1, \ldots, e_l$ mit $e_i \in \{(v_i, v_{i+1}), (v_{i+1}, v_i)\}$ für $1 \leq i \leq l-1$ und $e_l \in \{(v_l, v_1), (v_1, v_l)\}$.

Satz 8.15. *(Minty 1960) Sei D ein Digraph und sei $e \in A(D)$. Ist e schwarz gefärbt und sind alle anderen gerichteten Kanten von D rot, schwarz oder grün gefärbt, so gilt genau eine der folgenden zwei Aussagen:*

(i) Es existiert ein ungerichteter Kreis in D, der e enthält, in dem alle gerichteten Kanten rot oder schwarz sind und in dem alle schwarzen gerichteten Kanten die gleiche Orientierung haben.

(ii) Es existiert ein ungerichteter Schnitt

$$\{(u, v) \in A(D) \mid |\{u, v\} \cap U| = 1\}$$

mit $U \subseteq V(D)$, der e enthält, in dem alle gerichtete Kanten grün oder schwarz sind und in dem alle schwarzen gerichteten Kanten die gleiche Orientierung haben.

Beweis: Der folgende Algorithmus sucht zunächst nach einem Kreis wie in (i). Wird dieser nicht gefunden, so ergibt sich ein Schnitt wie in (ii).

Input: Ein Digraph D, eine gerichtete Kante (x, y) von D und eine Färbung $f : A(D) \to \{s, r, g\}$ mit $f((x, y)) = s$.
Output: Entweder ein gerichteter Kreis C wie in (i) oder eine Menge U wie in (ii).
begin
  $m(y) \leftarrow 1$;
  for $u \in V(D) \setminus \{y\}$ **do**
    $m(u) \leftarrow 0$;
    $p(u) \leftarrow \emptyset$;
  end
  while $\exists v, w \in V(D)$ *mit* $m(v) = 1$, $m(w) = 0$ *und entweder* $(v, w) \in f^{-1}(s) \cup f^{-1}(r)$ *oder* $(w, v) \in f^{-1}(r)$ **do**
    $m(w) \leftarrow 1$;
    $p(w) \leftarrow v$;
  end
  if $m(x) = 1$ **then**
    Sei l minimal mit $p^l(x) = y$;
1   **return** $C : p^l(x) \dots p^2(x)p(x)xp^l(x)$;
  else
2   **return** $U = m^{-1}(1)$;
  end
end

Algorithm 2: Algorithmus zu Minty's Satz

□

Definition 8.16. Sei D ein Digraph. Eine *topologische Ordnung* von D ist eine Abbildung $f : V(D) \to [n(D)]$ mit $\forall (u, v) \in A(D) : f(u) < f(v)$. D heißt *azyklisch*, falls D keine gerichteten Kreise besitzt.

Satz 8.17. *Ein Digraph D hat genau dann eine topologische Ordnung, wenn er azyklisch ist.*

□

Bemerkung 8.18. Datenstrukturen für Graphen G :

- Adjazenzmatrix ($O(n^2)$, dichte/dünne Graphen)
- Inzidenzmatrix ($\{0, 1\}^{V(G) \times E(G)}$)
- Kantenliste ($O(m \log n)$)
- Adjazenzliste ($O(m \log n + n \log m)$, Kanten plus Pointer auf die Anfänge der einzelnen Listen)

Analog für Digraphen, z.B. Inzidenzmatrix ($a_{u,(u,v)} = -1$ und $a_{v,(u,v)} = 1$).
Laufzeit $\sim$ Anzahl der "elementaren" Operationen.

Definition 8.19. Eine *Arboreszenz mit Wurzel r* ist ein Digraph D mit $r \in V(D)$, der aus einem (ungerichteten) Baum entsteht, indem man die Kanten so orientiert, dass für jeden Knoten u in D ein gerichteter Weg von r nach u existiert. Ein *Branching* ist ein Digraph, dessen schwache Zusammenhangskomponenten Arboreszenzen sind.

Algorithmus 8.20. GRAPH SCANNING ALGORITHMUS

Input: *Ein Graph G/ Digraph D und ein Knoten r von G/D.*

Output: *Die Menge R der Knoten, die in G/D von r aus auf Wegen/gerichteten Wegen erreichbar sind und eine Menge $T \subseteq E(G)/A(D)$, für die (R,T) Baum/Arboreszenz mit Wurzel r ist.*

begin
  $R \leftarrow \{r\}$; $Q \leftarrow \{r\}$; $T \leftarrow \emptyset$; $l(r) \leftarrow 0$;
  while $Q \neq \emptyset$ **do**
1   *Wähle $v \in Q$;*
2   **if** $\exists\ w \in V \setminus R$ *mit* $e = vw \in E(G)$ / $e = (v,w) \in A(D)$ **then**
        Wähle solch ein w;
        $R \leftarrow R \cup \{w\}$; $Q \leftarrow Q \cup \{w\}$; $T \leftarrow T \cup \{e\}$; $l(w) \leftarrow l(v) + 1$;
      else
        $Q \leftarrow Q \setminus \{v\}$;
      end
  end
  return (R,T);
end

Lemma 8.21. *Der* GRAPH SCANNING ALGORITHMUS *arbeitet korrekt.*

□

Lemma 8.22. *Der* GRAPH SCANNING ALGORITHMUS *kann so implementiert werden, dass er eine Laufzeit von $O(n(G) + m(G))$ besitzt. Man kann die Komponenten eines Graphen in linearer Zeit bestimmen.*

□

Bemerkung 8.23. Spezialfälle des GRAPH SCANNING ALGORITHMUS.

Depth-First-Search (DFS): Wähle in der **while**-Schleife (Zeile 1) als $v \in Q$ jeweils die Knoten, die als letzte zu Q hinzugefügt wurde. (LIFO, Tremaux Tarry vor 1900, König 1936)

Breadth-First-Search (BFS): Wähle in der **while**-Schleife (Zeile 1) als $v \in Q$ jeweils die Konten, die als erstes zu Q hinzugefügt wurde. (Moore 1959)

Die entsprechenden Ausgaben des GRAPH SCANNING ALGORITHMUS nennt man DFS-Bäume, DFS-Arboreszenzen, BFS-Bäume, BFS-Arboreszenzen.

Lemma 8.24. *BFS-Bäume enthalten kürzeste Wege von r zu allen von r aus erreichbaren Knoten, d.h. man kann die Abstände von einer Ecke r zu allen erreichbaren Knoten in linearer Zeit bestimmen.*

□