# 1 Allgemeines

Ein **Graph** $G$ wird beschrieben durch $G = (V, E)$.

Ein Graph $K_n = (V, E)$ ist **vollständig** wenn alle möglichen Kanten für die Knoten V in E enthalten sind.

Ein Knoten $x$ hat den **Grad** $n =: deg(x)$ wenn er genau $n$ Nachbarn hat. Ist $deg(x) = 0$ so ist $x$ ein **isolierter Knoten**. Ist $deg(x) = 1$ so ist $x$ ein **Blatt**.

Ein Graph $G$ ist genau dann **zusammenhängend** wenn jeder Knoten direkt oder indirekt mit jedem anderen Verbunden ist.

Ein Graph $C_n = (V, E)$ ist ein **Kreisgraph** wenn alle Knoten den Grad zwei haben und der Graph zusammenhängend ist.

Eine **Clique** in einem Graphen $G$ ist ein Subgraph dieses Graphen der isomorph zu einem Graphen $K_n$ ist (für ein beliebiges $n$). Das größtmögliche $n =: \omega(G)$ ist die **Cliquenzahl**

Eine **Stabile Menge** in einem Graphen $G$ ist ein Subgraph dieses Graphen der isomorph zu einem Graphen $E_n$ ist (für beliebiges $n$). Das größtmögliche $n =: \alpha(G)$ ist die **Stabilitätszahl**.

Der Graph $E_n = (V, \emptyset)$ ist der **leere Graph**.

Ein **Matching** ist eine Auswahl an eindeutigen Zuordnungen von Elementen einer Menge.

Eine **Knotenüberdeckung (Vertex Cover)** für einen Graphen $G = (V, E)$ ist eine Menge $V' \subset V$ wobei es für jedes $e \in E$ ein $v \in V'$ gibt für das $v \in e$.

# 2 Färbung

**Chromatische Zahl** ist $\chi(G) := min\{k \in \mathbb{N} : \text{G hat k Färbung}\}$.

Für jeden Graphen $G(V, E)$ gilt $\omega(g) \leq \chi(g)$ und $\frac{|V|}{\alpha(G)} \leq \chi(g)$.

Ein Graph $G$ ist **bipartit** wenn $\chi(G) \leq 2$.

Graph ist bipartit $\Leftrightarrow \nexists$ Kreis ungerader Länge in G

## 2.1 Greedy (Färbung)

Für alle Knoten $v \in V$ aus $G = (V, E)$: nehme ausschließlich bekannte Knotenfärbungszahlen aller verbundenen Knoten in eine Menge $N$, Knotenfärbung von v ist die kleinste Zahl aus den natürlichen Zahlen ohne $N$.

# 3 Planarität

Ein Graph ist genau dann planar, wenn er keine Unterteilung des $K_5$ oder des $K_{3,3}$ besitzt

Eulersche Polyederformel: ($R$ sei die Anzahl der Regionen inkl. Außenregion)

$$|V| - |E| + |R| = 2$$

Weiterhin: $|E| \leq 3n - 6$

$2|E| \leq 3|R|$

# 4 Netzwerke

## 4.1 Minimaler Spannbaum

Kruskal, Gewichte aufsteigend betrachten und nur inkludieren wenn neuer Knoten eingebunden wird oder Partitionen verbunden werden.

## 4.2 Floyd, Dajkstra, Kruskal

### 4.2.1 Floyd, Kürzeste Pfade

$d(i, j)$ initialisieren. Für alle $k \in 1$ bis $n$: Für alle Knotenpaare $i, j$ sei $d(i, j) = min(d(i, j), d(i, k) + d(k, j))$.

### 4.2.2 Dajkstra, Kürzeste Pfade bei nichtnegativen Kanten

Nehme Knoten mit minimaler Distanz zum Startknoten - hinzufügen. (Nachbarknoten ggf. updaten.)

### 4.2.3 Kruskal, Minimaler Spannbaum

Aus unbenutzten Kanten die kürzeste wählen, die mit den gewählten keinen Kreis bildet. Wiederholen.

# 5 Komplexitäten (vereinfacht)

## 5.1 Abschätzung nach oben

$$f \in O(g) \Leftrightarrow \lim_{x\to\infty} \left|\frac{f(x)}{g(x)}\right| < \infty \qquad f(n) = O(g(n)) \Leftrightarrow f(n) \leq c \cdot g(n)$$

## 5.2 Abschätzung

$$f \in \Theta(g) \Leftrightarrow 0 < \lim_{x\to\infty} \left|\frac{f(x)}{g(x)}\right| < \infty$$
$$f(n) = \Theta(g(n)) \Leftrightarrow c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$$

## 5.3 Abschätzung nach unten

$$f \in \Omega(g) \Leftrightarrow 0 < \lim_{x\to\infty} \left|\frac{f(x)}{g(x)}\right| \qquad f(n) = \Omega(g(n)) \Leftrightarrow f(n) \geq c \cdot g(n)$$

# 6 Matroide

$$M = (E; U) : \emptyset \in U$$
$$\forall A \in U; \forall B \in P(E) : B \subseteq A \Rightarrow B \in U$$
$$\forall A; B \in U : |B| < |A| \Rightarrow (\exists x \in A \setminus B : B \cup \{x\} \in U)$$

Unabhängigkeitssystem erfüllt nur die ersten beiden Bedingungen.

# 7 Linear Programming

## 7.1 LP

$$max(c^t x) \qquad Ax \leq b \qquad x \geq 0$$

## 7.2 Dual Problem

$$min(b^t y) \qquad A^t y \geq c \qquad y \geq 0$$

## 7.3 Simplex Algorithmus

$$max(2x + 3y)$$

1. Schlupfvariablen einfügen ($x - y \leq 2 \Rightarrow x - y + s_1 = 2, s_1 \geq 0$) also zur Gleichung transformieren.
2. Tableau erstellen

| | $x$ | $y$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | |
|---|---|---|---|---|---|---|---|
| $b_1$ | 1 | -1 | 1 | 0 | 0 | 0 | $s_1$ |
| $b_2$ | 5 | 6 | 0 | 1 | 0 | 0 | $s_2$ |
| $b_3$ | 8 | 9 | 0 | 0 | 1 | 0 | $s_3$ |
| $b_4$ | 8 | 9 | 0 | 0 | 0 | 1 | $s_4$ |
| 0 | -2 | -3 | 0 | 0 | 0 | 0 | |

3. Fertig wenn alle nichtschlupfspalten in der untersten Zeile $\geq 0$ sind.
4. Nimm Nichtschlupfspalte mit betragsmäßig größter unterer Zelle, nimm die Zelle wo $b_n/x : x > 0$ am kleinsten.
5. Skaliere $n$te Zeile so dass eben diese Pivotzelle 1 enthält.
6. Addiere $n$tn Zeile so zu allen anderen dass diese in der Pivotspalte 0 enthalten.
7. Tausche Element ganz rechts in Pivotzeile mit Element ganz oben in Pivotspalte. Wiederholen.

# 8 SAT

k-SAT mit $k \geq 3$ sind NP-schwer. $SAT \leq 3 - SAT \leq Clique$

# 9 Misc

$\sum\limits_{v \in V} deg(v) = 2 \cdot |E|$

Ungewichtetes Scheduling: Tasks mit größter Penalty auf ihre Deadline oder davor setzen, abwärts.

Rucksackproblem