AuW Recap

Axel Montini
amontini@student.ethz.ch

July 29, 2021

Cannot be used during the exam, but it's a nice short recap of everything done in the second semester.

1 Last semester

Go read again about MST algorithms and so on.

2 Graphentheorie

2.1 Zusammenhang

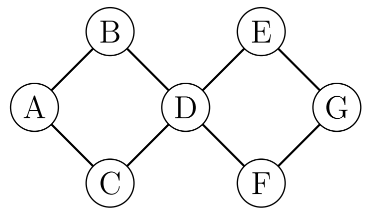
Definition 2.1. A graph $G = (V, E)$ is *k-zusammenhängend* if $|V| \geq k + 1$ and for all $X \subseteq V,\ |X| < k$ the following is true:

$$\text{The graph } G[V \setminus X] \text{ is zusammengehängend}$$

Definition 2.2. A graph $G = (V, E)$ is *k-kanten-zusammenhängend* if for all $X \subseteq E,\ |X| < k$ the following is true:

$$\text{The graph } G(V, E \setminus X) \text{ is zusammengehängend}$$

Note 2.3. A graph can be both 2-kanten-zusammengehängend and be only 1-zusammengehängend at the same time. Example:



Definition 2.4. In a zusammengehängend Graph, *Artikulationsknoten* disconnect the graph when removed. Only 1-zusammengehängend graphs can have Artikulationsknoten

Theorem 2.5. *In zusammenhängende Graphs it's possible to find Artikulationsknoten in $\mathcal{O}(|E|)$ if an adjacency list is used.*

Definition 2.6. A zusammenhängend Graph may contain *Brücke*. In this case, it's **not 2-kanten-zusammenhängend**.

An edge is a bridge if it disconnects the graph when removed.

Theorem 2.7. *Brücke can also be computed in $\mathcal{O}(|E|)$ using an adjacency list.*

Definition 2.8. $G = (V, E)$ is zusammenhängend. For $e, f \in E$ we define the relation

$$e \sim f \Leftrightarrow e = f \text{ or there is a Kreis containing both edges}$$

This is an equivalence relation. Each equivalence class is called a Block (plural Blöcke).

2.2 Kreise

Definition 2.9. An *Eulertour* in a graph G is a closed path (*Zyklus*) that contains every edge $\in E$ exactly once.

A graph containing an Eulertour is called *eulersch*.

Definition 2.10.

Theorem 2.11.

$$\textit{A graph is eulersch} \Leftrightarrow deg(v) \textit{ is even for all vertices}$$

Theorem 2.12. *In a connected and eulersch Graph it's possible to find an Eulertour in* $\mathcal{O}(|E|)$.

Definition 2.13. A *Hamintonkreis* in G is a cycle that goes through every vertex exaclty once. A graph containing an Hamintonkreis is called *hamintonsch.*

Theorem 2.14. *The algorithm seen in class can find an Hamintonkreis in time* $\mathcal{O}(n^2 \cdot 2^n)$ *and memory* $\mathcal{O}(n \cdot 2^n)$, *where* $n = |V|$.

Theorem 2.15. *A bipartite graph* $G = (A \uplus B, E)$ *cannot contain an Hamintonkreis.*

Theorem 2.16 (Dirac)**.** *A graph* G *with* $V \geq 3$ *in which every vertex has at least* $|V|/2$ *neighbors is hamintonsch.*

Definition 2.17. In a complete graph K_n (all vertices are connected together), the metric Traveling Salesman Problem consists in finding an Hamintonkreis C with minimal cost (distance).

Definition 2.18. An α-Approximationsalgorithmus of this problem finds an H.kreis C so that

$$\sum_{e \in C} l(e) \leq \alpha \cdot opt(K_n, l)$$

Meaning that it finds a solution worse by the optimal solution by a factor α.

Theorem 2.19. *If there's an* α*-Approximationsalgorithmus with* $\alpha > 1$ *for the TSP with running time* $\mathcal{O}(f(n))$ *then there's also an algorithm that decides whether a graph with* n *vertices is hamintonsch in* $\mathcal{O}(f(n))$.

Theorem 2.20. *For the metric TSP there's a 2-Approximationsalgorithmus with running time* $\mathcal{O}(n^2)$. *It find the MST in* $\mathcal{O}(n^2)$ *and then uses it to find the H.k.*

2.3 Matching

Definition 2.21. A set of edges $M \subseteq E$ is called *Matching* in a graph G if

$$\forall e, f \in M \ (e \cap f = \emptyset)$$

- A vertex v is said to be *überdeckt* by a matching M if the matching contains an edge containing v.

- A matching M is called *perfektes Matching* if every vertex is überdeckt (equivalent: $|M| = |V|/2$).

Definition 2.22. A matching M is said to be:

- *inklusionsmaximal* if $M \cup \{e\}$ is not a matching for all $e \in E \setminus M$.
- *kardinalitätsmaximal* if $|M| \geq |M'|$ for all matchings M' in G.

Note that *kardinalitätsmaximal* $\Rightarrow$ *inklusionsmaximal* (the opposite might not be true).

Algorithm 1: Greedy-Matching

Result: The inklusionsmaximales matching M
while $E \neq \emptyset$ **do**
  choose an edge $e \in E$;
  $M \leftarrow M \cup \{e\}$;
  remove e and all incident edges from G;

Theorem 2.23. *The greedy-matching algorithm finds an inklusionsmaximal Matching in time $\mathcal{O}(|E|)$ for which the following holds: $|M_{Greedy}| \geq \frac{1}{2}|M_{max}|$, where M_{max} is a kardinalitätsmaximales Matching.*

Theorem 2.24 (Berge)**.** *Let M be a matching in G that is not k.maximal, then there is an augmenting path to M.*

Algos at page 65 and 67

Theorem 2.25. *Is n even K_n a complete graph, then it's possible to find a minimal perkeftes Matching in time $\mathcal{O}(n^3)$*

Theorem 2.26. *There's a 3/2-Approximationsalgorithmus for the TSP that runs in $\mathcal{O}(n^3)$.*

Definition 2.27. *Nachbarnschaft einer Knotenmenge $X \subseteq V$:*

$$N(X) := \bigcup_{v \in X} N(v)$$

Theorem 2.28 (Hall, Heiratssatz)**.** *For a bipartite graph $G = (A \uplus B, E)$ theres a matching M with $|M| = |A|$ if and only if $|N(X)| \geq |X|$ for all $X \subseteq A$.*

Definition 2.29. A bipartite graph is called *k-regulär* if every vertex has degree k.

Theorem 2.30. *Let G be a k-regular bipartite graph. Then there's $M_1, ..., M_k$ so that $E = M_1 \uplus M_2 \uplus ... \uplus M_k$ and all $M_i, 1 \leq i \leq k$ are perfect matchings.*

Theorem 2.31. *Is $G = (V, E)$ a 2^k-regular bipartite Graph, then it's possible to find a perfect matching in $\mathcal{O}(|E|)$.*

2.4 Färbungen

Definition 2.32. A *(Knoten)-Färbung* (vertex coloring) of a graph G with k colors is $c : V \rightarrow [k]$, so that

$$c(u) \neq c(v)\ \forall \{u, v\} \in E$$

The *chromatische Zahl* (chromatic number) $X(G)$ of a graph is the minimal amount of colors that can be used to color G.

Theorem 2.33. *A graph is bipartite if and only if doesn't contain any Kreis of odd length.*

Theorem 2.34 (Vierfarbensatz)**.** *Every map can be colored with 4 colors.*

Algorithm 2: Greedy-Färbung

Data: G
Result: array c mapping each vertex to a color
$c(v_1) \leftarrow 1$;
for $i = 2, \dots, n$ **do**
  $c(v_i) \leftarrow \min \{k \in \mathbb{N} \mid k \neq c(u)$ for all $u \in N(v_i) \cap \{v_1, ..., v_{i-1}\}\}$

Theorem 2.35. *Let G be a connected graph and $C(G)$ the amount of colors used by the Greedy-Färbung algorithm. Then*

$$\mathcal{X}(G) \leq C(G) \leq \Delta(G) + 1$$

Where $\Delta(G) := \max_{v \in V} deg(v)$ is the max degree in the graph.
The running time is $\mathcal{O}(|E|)$ if an adjacency list is used.

Theorem 2.36 (Brooks)**.** *Let G be a connected graph that is neither complete nor an odd Kreis ($G \neq K_n$ and $G \neq C_{2n+1}$). Then*

$$\mathcal{X}(G) \leq \Delta(G)$$

And there's an algorithm that can color the vertices of G in time $\mathcal{O}(|E|)$ and with $\Delta(G)$ colors.

Theorem 2.37 (Mycielski-Konstruktion)**.** *For all $k \geq 2$ there's a triangle-free graph G_k with $\mathcal{X}(G) \geq k$.*

Theorem 2.38. *Every 3-färbbaren graph can be colored in time $\mathcal{O}(|E|)$ with $\mathcal{O}(\sqrt{|V|})$ colors.*

3 Randomized algorithms

3.1 Grundbegriffe und Notationen

Definition 3.1. A *diskreter Wahrscheinlichkeitsraum* is defined through a *Ergebnismenge* $\Omega = \{\omega_1, \omega_2, ...\}$ of *Elementarereignissen.*

A probability $\Pr[\omega_i]$ corresponds to each ω_i.

$$0 \leq \Pr[w_i] \leq 1, \quad \sum_{\omega \in \Omega} \Pr[\omega] = 1$$

A set $E \subseteq \Omega$ is called *Ereignis.* The probability is defined as

$$\Pr[E] := \sum_{\omega \in E} \Pr[\omega]$$

The *Komplementärereignis* zu E is defined as $\overline{E} := \Omega \setminus E$

Lemma 3.2. *For Ereignisse A, B:*

1. $\Pr[\emptyset] = 0, \Pr[\Omega] = 1$
2. $0 \leq \Pr[A] \leq 1$
3. $\Pr[\overline{A}] = 1 - \Pr[A]$
4. *If $A \subseteq B$ then* $\Pr[A] \leq \Pr[B]$

Theorem 3.3 (Additionssatz)**.** *When the Ereignisse are pairwise disjoint then*

$$\Pr\left[\bigcup_{i=1}^{n} A_i\right] = \sum_{i=1}^{n} \Pr[A_i]$$

For infinite sets of disjoint Ereignissen $A_1, A_2, \ldots$ then

$$\Pr\left[\bigcup_{i=1}^{\infty} A_i\right] = \sum_{i=1}^{\infty} \Pr[A_i]$$

Theorem 3.4 (Siebformel, Prinzip der Inklusion/Exklusion)**.** *For Ereignisse $A_1, \ldots, A_n$ $(n \geq 2)$:*

$$\Pr\left[\bigcup_{i=1}^{n} A_i\right] = \sum_{l=1}^{n} (-1)^{l+1} \sum_{1 \leq i_1 < \ldots < i_l \leq n} \Pr[A_{i_1} \cap \ldots \cap A_{i_l}]$$

Lemma 3.5 (A special case of the Siebformel)**.** *Let $\Omega = A_1 \cup \ldots \cup A_n$ with $\Pr[\omega] = 1/|\Omega|$, where A_i are finite sets. Then*

$$|\bigcup_{i=1}^{n} A_i| = \sum_{l=1}^{n} (-1)^{l+1} \sum_{1 \leq i_1 < \ldots < i_l \leq n} |A_{i_1} \cap \ldots \cap A_{i_l}|$$

Corollary 3.6 (Boolsche Ungleichung, Union Bound)**.** *For some Ereignisse $A_1, \ldots, A_n$:*

$$\Pr\left[\bigcup_{i=1}^{n} A_i\right] \leq \sum_{i=1}^{n} \Pr[A_i]$$

For an infinite set of Ereignisse, replace n with ∞.

3.2 Bedingte Wahrscheinlichkeiten

Definition 3.7. Let A and B be Ereignisse with $\Pr[B] > 0$. The *bedingte Wahrscheinlichkeit $Pr[A|B]$* of A given B is defined through

$$\Pr[A|B] := \frac{\Pr[A \cap B]}{\Pr[B]}$$

Theorem 3.8 (Multiplikationssatz)**.** *The Ereignisse $A_1, \ldots, A_n$ are given. If $\Pr[A_1 \cap \ldots \cap A_n] > 0$, then:*

$$\Pr[A_1 \cap \ldots \cap A_n] = \Pr[A_1] \cdot \Pr[A_2|A_1] \cdot \Pr[A_3|A_1 \cap A_2] \cdot \ldots \cdot \Pr[A_n|A_1 \cap \ldots \cap A_{n-1}]$$

Theorem 3.9 (Satz von der totalen Wahrscheinlichkeit)**.** *$A_1, ..., A_n$ are pairwise disjoint and $B \subseteq A_1 \cup ... \cup A_n$. Then*

$$\Pr[B] = \sum_{i=1}^{n} \Pr[B|A_i] \cdot \Pr[A_i]$$

Analogous for an infinite set of A_is.

Theorem 3.10 (Satz von Bayes)**.** *$A_1, ..., A_n$ pairwise disjoint. Let $B \subseteq A_1 \cup ... \cup A_n$ with $\Pr[B] > 0$. Then for any $i = 1, ..., n$:*

$$\Pr[A_i|B] = \frac{\Pr[A_i \cap B]}{\Pr[B]} = \frac{\Pr[B|A_i] \cdot \Pr[A_i]}{\sum_{j=1}^{n} \Pr[B|A_j] \cdot \Pr[A_j]}$$

Analogous for ∞ instead of n.

3.3 Unabhängigkeit

Definition 3.11. Die Ereignisse A, B are *unabhängig* when

$$\Pr[A \cap B] = \Pr[A] \cdot \Pr[B]$$

Definition 3.12. Die Ereignisse $A_1, ..., A_n$ are unabhängig when for all $I \subseteq \{1, ..., n\}$ with $I = i_1, ..., i_k$:

$$\Pr\left[A_{i_1} \cap ... \cap A_{i_k}\right] = \Pr\left[A_{i_1}\right] \cdot ... \cdot \Pr[A_{i_k}]$$

Lemma 3.13. *Die Ereignisse $A_1, ..., A_n$ are unabängig* ***if and only if***

$$\forall (s_1, .., s_n) \in \{0, 1\}^n \;\; (\Pr[A_1^{s_1} \cap ... \cap A_n^{s_n}] = \Pr[A_1^{s_1}] \cdot \Pr[A_n^{s_n}])$$

where $A_i^0 = \overline{A_i}$ and $A_i^1 = A_i$

Lemma 3.14. *A, B, C unabängige Ereignisse. Then $A \cap B$ and C and $A \cup B$ and C are also independent.*

3.4 Zufallsvariablen

Definition 3.15. A *Zufallsvariable* is $X : \Omega \to \mathbb{R}$, where Ω is the Ergebnismenge of a Wahrscheinlichkeitsraum.

The *Wertebereich* of a Zufallsvariable is

$$W_X := X(\Omega) = \{x \in \mathbb{R} \mid \exists \omega \in \Omega \text{ with } X(\omega) = x\}$$

Definition 3.16. The *Erwartungswert* of X is $\mathbb{E}[X]$, defined as

$$\mathbb{E}[X] := \sum_{x \in W_x} x \cdot \Pr[X = x]$$

when the sum absolut konvergiert. Otherwise the Erwartungswert is *undefiniert / undefined.*

Lemma 3.17. *Is X a Zufallsvariable, then:*

$$\mathbb{E}[X] = \sum_{\omega \in \Omega} X(\omega) \cdot \Pr[\omega]$$

Theorem 3.18. *X Zufallsvariable wit $W_X \subseteq \mathbb{N}_0$. Then*

$$\mathbb{E}[X] = \sum_{i=1} \infty \Pr[X \geq i]$$

Theorem 3.19. *X a Zufallsvariable. For pairwise disjoint Ereignisse $A_1, ..., A_n$ with $A_1 \cup ... \cup A_n = \Omega$ and $\Pr[A_1], ..., \Pr[A_n] > 0$ then*

$$\mathbb{E}[X] = \sum_{i=1}^{n} \mathbb{E}[X|A_i] \cdot \Pr[A_i]$$

Analogous for ∞ instead of n.

Theorem 3.20 (Linearität des Erwartungswert)**.** *For $X_1, ..., X_n$ and $X := a_1X_1 + ... + a_nX_n + b$ with $a_1, ..., a_n, b \in \mathbb{R}$:*

$$\mathbb{E}[X] = a_1\mathbb{E}[X_1] + ... + a_n\mathbb{E}[X_n] + b$$

Note 3.21. For an Ereignis $A \subseteq \Omega$ the *Indikatorvariable* X_A is defined as

$$X_A(\omega) := \begin{cases} 1, & \text{if } \omega \in A \\ 0, & \text{otherwise} \end{cases}$$

Also $\mathbb{E}[X_A] = \Pr[A]$

Definition 3.22. For a Zufallsvariable X mit $\mu = \mathbb{E}[X]$ the *Varianz* $Var[X]$ is defined through

$$Var[X] := \mathbb{E}[(X - \mu)^2] = \sum_{x \in W_X} (x - \mu)^2 \Pr[X = x]$$

The *Standardabweichung* of X is $\sigma := \sqrt{Var[X]}$.

Theorem 3.23. *For any Zufallsvariable X and $a, b \in \mathbb{R}$:*

$$Var[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$$

$$Var[a \cdot X + b] = a^2 \cdot Var[X]$$

Definition 3.24. For a Zufallsvariable X, $\mathbb{E}[X^k]$ is called *k-te Moment* and $\mathbb{E}[(X - \mathbb{E}[X])^k]$ is called *k-te zentrale Moment*.

3.5 Wichtige diskrete Verteilungen

Maybe write, or use the Formelsammlung

Bernoulli, Binomial, Geometrische, Poisson

3.6 Mehrere Zufallsvariablen

Definition 3.25. Zufallsvariablen $X_1, ..., X_n$ are *unabhängig* if and only if for all $(x_1, ..., x_n) \in W_{X_1} \times ... \times W_{X_n}$ the following holds:

$$\Pr[X_1 = x_1, ..., X_n = x_n] = \Pr[X_1 = x_1] \cdot ... \cdot \Pr[X_n = x_n]$$

Theorem 3.26 (Linearität des Erwartungswerts)**.** *For $X_1, ..., X_n$ and $X := a_1X_1 + ... + a_nX_n$ with $a_i \in \mathbb{R}$:*

$$\mathbb{E}[X] = a_1\mathbb{E}[X_1] + ... + a_n\mathbb{E}[X_n]$$

Theorem 3.27 (Multiplikativität des Erwartungswerts)**.** *For $X_1, ..., X_n$:*

$$\mathbb{E}[X_1 \cdot ... \cdot X_n] = \mathbb{E}[X_1] \cdot ... \cdot \mathbb{E}[X_n]$$

Theorem 3.28. *For independent $X_1, ..., X_n$ and $X := X_1 + ... + X_n$ then*

$$Var[X] = Var[X_1] + ... + Var[X_n]$$

3.7 Abschätzen von Wahrscheinlichkeiten

Theorem 3.29 (Ungleichung von Markov)**.** *$X \geq 0$ Zufallsvariable. Then, for all $t > 0 \in \mathbb{R}$:*

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t} \text{ und } \Pr[X \geq t \cdot \mathbb{E}[X]] \leq \frac{1}{t}$$

Theorem 3.30 (Ungleichung von Chebyshev)**.** *X Zufallsvariable and $t > 0 \in \mathbb{R}$. Then*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \frac{Var[X]}{t^2}$$

Theorem 3.31 (Ungleichung von Chernoff)**.** *$X_1, ..., X_n$ independent Beroulli-verteilte Zufallsvariablen with $\Pr[X_i = 1] = p_i$ and $\Pr[X_i = 0] = 1 - p_i$.*

Then for $X := \sum_{i=1}^{n} X_i$:

$$\Pr[X \geq (1+\delta)\mathbb{E}[X]] \leq e^{-\frac{1}{3}\delta^2\mathbb{E}[X]} \text{for all } 0 < \delta \leq 1 \quad (1)$$

$$\Pr[X \leq (1-\delta)\mathbb{E}[X]] \leq e^{-\frac{1}{2}\delta^2\mathbb{E}[X]} \text{for all } 0 < \delta \leq 1 \quad (2)$$

$$\Pr[X \geq t] \leq 2^{-t} \qquad \text{for } t \geq 2e\mathbb{E}[X] \quad (3)$$

3.8 Randomized algorithms

Theorem 3.32. *Let $\mathcal{A}$ be a randomized algorithm that always never gives a wrong answer, but outputs "???" when*

$$\Pr[\mathcal{A}(I) \text{ correct}] \geq \epsilon$$

Then for all $\delta > 0$: let $\mathcal{A}_\delta$ be the algorithm that calls $\mathcal{A}$ until either an answer different from "???" is returned ($\mathcal{A}_\delta$ gives this answer) or until "???" is returned $N = \epsilon^{-1} \ln \delta^{-1}$ times ($\mathcal{A}_\delta$ returns "???").

Then the following is true:

$$\Pr[\mathcal{A}_\delta \text{ correct}] \geq 1 - \delta$$

Theorem 3.33. *Let $\mathcal{A}$ be a randomized algorithm that always gives "Yes" or "No" as answers, where*

$$\Pr[\mathcal{A}(I) = \text{Yes}] = 1 \quad \text{if the correct answer is Yes} \quad (4)$$

$$\Pr[\mathcal{A}(I) = \text{No}] \geq \epsilon \quad \text{if the correct answer is No} \quad (5)$$

Then for all $\delta > 0$: let $\mathcal{A}_\delta$ be the algorithm that calls $\mathcal{A}$ either until "No" is returned (then it returns "No") or until "Yes" is returned $N = \epsilon^{-1} \ln \delta^{-1}$ times (then it returns "Yes").

Then for all instances I:

$$\Pr[\mathcal{A}_\delta(I) \text{ correct}] \geq 1 - \delta$$

Theorem 3.34. *Let $\epsilon > 0$ and $\mathcal{A}$ be a randomized algorithm that always returns either "Yes" or "No", Where*

$$\Pr[\mathcal{A}_\delta(I) \text{ correct}] \geq \frac{1}{2} + \epsilon$$

Then for all $\delta > 0$: let $\mathcal{A}_\delta$ be the algorihm that calls $\mathcal{A}$ a total of $N = 4\epsilon^{-2} \ln \delta^{-1}$ times and returns the answer that was encountered the most.

Then

$$\Pr[\mathcal{A}_\delta(I) \text{ correct}] \geq 1 - \delta$$

Theorem 3.35. *Let $\epsilon > 0$ and $\mathcal{A}$ be a randomized algorithm for a Maximierungsproblem, where*

$$\Pr[\mathcal{A}(I) \ \geq f(I)] \geq \epsilon$$

Then for all $\delta > 0$: let $\mathcal{A}_\delta$ be the algo that calls $\mathcal{A}$ a total of $N = \epsilon^{-1} \ln \delta^{-1}$ times and returns the best answer. Then

$$\Pr[\mathcal{A}_\delta(I) \geq f(I)] \geq 1 - \delta$$

Analogous for Minimierungsprobleme, replace $\geq f(I)$ with $\leq f(I)$.

Note 3.36. Stuff about random pivot selection for QuickSort and so on

Theorem 3.37 (Kleiner fermatscher satz)**.** *If $n \in \mathbb{N}$ is prime, then for all $0 < a < n$*

$$a^{n-1} \equiv 1 \mod n$$

Target shooting:

Theorem 3.38. *Let $\delta, \epsilon > 0$. If $N \geq 3\frac{|U|}{|S|} \cdot \epsilon^{-2} \cdot \ln(2/\delta)$, then the output of the algorithm Target-Shooting is in the interval $\left[(1-\epsilon)\frac{|S|}{|U|}, (1+\epsilon)\frac{|S|}{|U|}\right]$ with probability $1 - \delta$.*

4 Algorithmen

4.1 Graphenalgorithmen

4.1.1 Lange Pfade

Given a graph G, determine whether it contains a path of length $B \in N_0$.

Call this *Long-Path Problem*.

Theorem 4.1. *If Long-Path can be computed in $t(n)$, then it can be detemined whether a graph contains a Hamiltonkreis in time $t(2n-2) + \mathcal{O}(n^2)$.*

4.1.2 Bunte Pfade

A variation. In a colored graph G, a path is *bunt* when all vertices in the path have different colors.

Random Bunte Pfade:

Theorem 4.2. *Let G be a graph with a path of length $k-1$.*

1. *A random coloring with k colors creates a bunten Pfad of length $k-1$ with probability $p_s \geq e^{-k}$.*
2. *On repeated random colorings with k colors the Erwartungswert of the amount of trials needed to find a bunten Pfad of length $k-1$ is $\frac{1}{p_s} \leq e^k$*

Theorem 4.3. *A Monte Carlo algorithm for this problem: Choose a $\lambda > 1$ and repeat the test at most $\lceil \lambda e^k \rceil$ times until it reports that a bunte path of length $k-1$ exists. In this case, the anwser is "Yes", otherwise "No".*

1. *Running time of the algo is $\mathcal{O}(\lambda(2e)^k km)$*
2. *If the algorithm answers "Yes", then the graph has a colorful Path of length $k-1$.*
3. *If the graph has a colorful Path, the probability of a "No" answer is $\leq e^{-\lambda}$*

4.2 Flows and Networks

4.2.1 Definition, Max flow, Min cut

Definition 4.4 (Network)**.** A *Network* is a tuple $N = (V, A, c, s, t)$, where

- (V, A) is a directed graph
- $s \in V$ is the source (Quelle)
- $t \in V$ is the target/sink (Senke)
- $C : A \to \mathbb{R}_0^+$ is the capacity function.

Definition 4.5 (Flow)**.** Given a network $N = (V, A, c, s, t)$, a *flow* in N is a function $f : A \to \mathbb{R}$ with the properties:

- $0 \leq f(e) \leq c(e)$ for all $e \in A$, die Zulässigkeit.
- For all $v \in V \setminus \{s, t\}$ the following is true (Flusserhaltung):

$$\sum_{u \in V:\ (u,v) \in A} f(u, v) = \sum_{u \in V:\ (v,u) \in A} f(v, u)$$

- The *Wert/value* of a flow f is defined as
-

$$val(f) := netoutflow(s) := \sum_{u \in V:\ (s,u) \in A} f(s, u) - \sum_{u \in V:\ (u,s) \in A} f(u, s)$$

Lemma 4.6. *The netinflow of the target/sink is equal to the flow value*

$$netinflow(t) := val(f)$$

Definition 4.7 (S-T Schnitt)**.** An s-t-Schnitt (s-t-cut) for a network $N = (V, A, c, s, t)$ is a partition (S, T) of V (meaning that $S \cup T = V$ and $S \cap T = \emptyset$) with $s \in S$ and $t \in T$.

The *capacity* of the s-t-cut is defined as

$$cap(S, T) := \sum_{(u,w) \in (S \times T) \cap A} c(u, w)$$

Lemma 4.8. *Let f be a flow and (S, T) an s-t-cut in the network $N = (V, A, c, s, t)$. Then*

$$val(f) \leq cap(S, T)$$

Theorem 4.9 (Maxflow-Mincut)**.**

$$\max_{f \text{ flow in } N} val(f) = \min_{(S,T) \text{ s-t-cut in } N} cap(S, T)$$

Definition 4.10 (Restnetzwerk)**.** Let $N = (V, A, c, s, t)$ be a network without reversed edges.

The *Restnetzwerk* $N_f := (V, A_f, r_f, s, t)$ is defined as follows:

1. $e \in A$ and $f(e) < c(e)$, then $e \in A_f$ and $r_f(e) = c(e) - f(e)$.
2. $e \in A$ and $f(e) > 0$, then $e^{opp} \in A_f$ and $r_f(e^{opp}) = f(e)$

3. Edges not satisfying any of the two above contitions are not in the Restnetzwerk.

$r_f(e), e \in A_f$ is called the *Restkapazität* of the edge e.

Theorem 4.11. *A flow f in a network N is a max flow* ***if and only if*** *there is no directed path from s to t in the Restnetzwerk N_f.*

For every max flow there's a min s-t-cut.

Theorem 4.12. *The following algorithm computes the max flow.*

Algorithm 3: Ford-Fulkerson

Data: (V, A, c, s, t)
Result: A max flow

```
f ← 0;
while ∃ s-t-path P in (V, A_f) do // Augmenting path
    ϵ smallest Restkapazität on the path P;
    foreach e ∈ A do // Increase flow along the path
        if e on P then
            f'(e) = f(e) + ϵ;
        else if e^opp on P then
            f'(e) = f(e) − ϵ;
        else
            f'(e) = f(e);
return f
```

If the network has no reversed edges and all capacities are integers $\leq U$, then there's an integer maxflow that can be computed in time $\mathcal{O}(mnU)$ (m amount of edges, n amount of vertices).

Proposition 4.13 (Capacity-Scaling, Dinitz-GAbow, 1973)**.** *If all capacities in a network are integers and at most U, then there's an integer maxflow that can be computed in time $\mathcal{O}(mn(1 + \log U))$*

Proposition 4.14 (Dynamic Trees, Sleaton-Tarjan, 1983)**.** *The maxflow of a network N can be computed in time $\mathcal{O}(mn \log n)$*

4.2.2 Bipartite matching as flow problem

Theorem 4.15. *Let $G = (V, E)$ be a bipartite graph, so there's a partition (U, W) of V so that $E \subseteq \{\{u, w\} \mid u \in U, w \in W\}$*

We're looking for the biggest matching in G.

Define a network $N = (V \cup \{s, t\}, A, c, s, t)$, so that

1. *The vertices are the same as in the graph, plus s and t.*
2. *The capacity function c is constant 1.*
3. *Define the edges as:*

$$A := (\{s\} \times U) \cup \{(u, w) \in U \times W \mid \{u, w\} \in E\} \cup (W \times \{t\})$$

The size of the biggest matching in the bipartite graph G is $maxflow(N)$.