AnW - Woche 2

29.02.24
Nicdas Wehrli

Heute:

I. Minitest
II. Theory Recap
- Satz von Dirac
- Hamilton DP
- Hamiltonkreis Aufgabe
- NP-Vollständigkeit
- Traveling Salesman
- Matchings

I. Minitest

Password: precision

Sei $G = (V, E)$ ein zusammenhängender Graph und sei die Anzahl seiner Knoten mit ungeradem Grad gerade. Dann enthält G eine Eulertour.

Bitte wählen Sie eine Antwort:

Wahr ○

Falsch ⊗

Für $G=(V,E)$ zsh. gilt:

G enthält eine Eulertour $\Leftrightarrow$ $\deg(v)$ gerade $\forall v \in V$

siehe Skript Satz 1.31

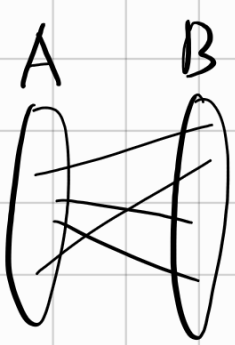
Sei n eine natürliche Zahl und G ein bipartiter Graph auf $2n + 1$ Knoten. Dann gilt: G enthält keinen Hamiltonkreis.

Bitte wählen Sie eine Antwort:

Wahr ⊗

Falsch ○

Sei $G=(A \dot\cup B, E)$ bipartit.



Aber da $|V| = 2n+1$, gilt $|A| \neq |B|$

Sei $\langle v_1, v_2, \dots, v_{2n+1}, v_1 \rangle$ ein Hamiltonkreis.

Sei o.b.d.A. $v_1 \in A$. Dann ist $v_2 \in B$, $v_3 \in A$, ...

i.e. $v_i \in \begin{cases} A & i \equiv_2 1 \\ B & \text{sonst} \end{cases}$, $\Rightarrow v_{2n+1} \in A \Rightarrow \{v_{2n+1}, v_1\} \in E$ ist ein Widerspruch!

Satz: Ein bipartiter Graph $G = (A \cup B, E)$ mit $|A| \neq |B|$ kann keinen Hamiltonkreis enthalten.

Der Kreis auf acht Knoten (C_8) ist 2-zusammenhängend.

Bitte wählen Sie eine Antwort:

Wahr ⊗

Falsch ○

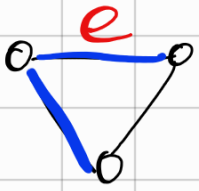
2 zsh., da wir mind. 2 Knoten entfernen müssen, um den Zusammenhang zu zerstören.

Sei G ein zusammenhängender Graph, $T(G)$ ein Spannbaum von G und $e \in E(T(G))$ eine Kante im Spannbaum. Es gilt: e ist eine Brücke in G.

Bitte wählen Sie eine Antwort:

Wahr ○

Falsch ⊗



T

$e \in E(T(G))$

aber e ist keine Brücke

Gegeben sind für einen Graphen G zwei Blöcke A, B mit $A \neq B$. Dann liegt keine Kante sowohl in A als auch in B.

Bitte wählen Sie eine Antwort:

Wahr ⊗

Falsch ○

Wir haben die Blöcke über eine Äquivalenzrelation ~ definiert.

Definition: Sei $G = (V, E)$. Wir definieren eine Äquivalenzrelation auf E durch

$$e \sim f :\Longleftrightarrow \begin{cases} e = f, & \text{oder} \\ \exists \text{ Kreis durch } e \text{ und } f \end{cases}$$

Nehmen wir per Widerspruch an, es existiert eine Kante $e \in A, e \in B$.

Per Transitivität gilt: $e \sim f \Rightarrow f \in A, \forall f \in E$

aber auch $e \sim g \Rightarrow g \in B, \forall g \in E$

Aber da $\{f \in E \mid e \sim f\} = \{g \in E \mid e \sim g\}$

folgt $A = B$, was unserer Annahme widerspricht.

$\Rightarrow$ Es existiert keine Kante die in A und B liegt.

Ein zusammenhängender Graph $G = (V, E)$ mit $|V| \geq 3$, der keinen Artikulationsknoten enthält, ist 2-zusammenhängend.

Bitte wählen Sie eine Antwort:

- Wahr ☒
- Falsch ○

G ist zsh.

kein Artikulationsknoten $\Rightarrow$ wir können eine beliebige Knotenmenge $|X| < 2$ vom Graph entfernen, ohne den Zsh. zu zerstören.

Da $|V| \geq 3$ folgt G ist 2 zsh.

Für das metrische TSP-Problem gibt es einen 2-Approximationsalgorithmus, aber keinen 4-Approximationsalgorithmus

Bitte wählen Sie eine Antwort:

Wahr ○

Falsch ⊗

Ein k-Approx.algorithmus gibt eine maximal k-mal schlechtere Lösung als das Optimum.

Da eine maximal 2-mal schlechtere Lösung auch maximal 4-mal schlechter ist, ist die Aussage falsch.

Seien $k, \ell > 1$ und k, ℓ beide ungerade. Das $(k \times \ell)$-Gitter enthält keinen Hamiltonkreis.

Bitte wählen Sie eine Antwort:

Wahr ⊗

Falsch ○

Satz: Seien m,n ≥ 2.

Ein n×m Gitter enthält einen Hamiltonkreis **gdw** n*m gerade ist.

In einer Tiefensuche sei $w \in V$ das erste Kind von $v \in V$ im DFS-Baum.

Falls $\text{low}[w] \geq \text{dfs}[v]$, so ist v ein Artikulationsknoten.

Bitte wählen Sie eine Antwort:

Wahr ○

Falsch ⊗

v ist Artikulationsknoten

$\iff$ $v = s$ und s hat in T Grad mindestens zwei, oder

$v \neq s$ und es gibt $w \in V$ mit $\{v, w\} \in E(T)$ und $\texttt{low}[w] \geq \texttt{dfs}[v]$.

Wenn $v = s$, dann gilt für alle kinder w

$low[w] \geq dfs[v]$, unabhängig davon ob v ein A.K ist oder nicht.

Für einen zusammenhängenden Graphen $G = (V, E)$, der in einer Adjazenzmatrix gespeichert ist, kann man in Zeit $O(|V|^2)$ alle Artikulationsknoten und alle Brücken berechnen.

Bitte wählen Sie eine Antwort:

Wahr ☒

Falsch ○

DFS-Low with Adjacency Matrix instead of Adj. List

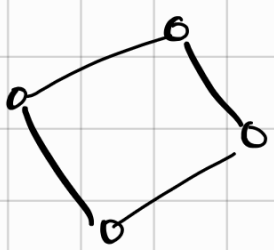
Vollständige Graphen sind die einzigen zusammenhängenden Graphen, welche sowohl eine Eulertour als auch einen Hamiltonkreis enthalten.

Bitte wählen Sie eine Antwort:

Wahr ○

Falsch ☒

Counterexample:



II. Theory Recap

Satz von Dirac

Satz: (Dirac 1952)

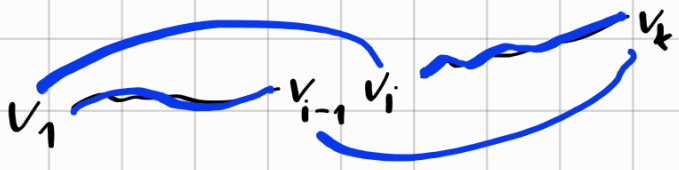
Jeder Graph $G = (V, E)$ mit $|V| \geq 3$ und Minimalgrad $\delta(G) \geq |V|/2$ enthält einen Hamiltonkreis.

1. Wie im Minitest oder/und in der Vorlesung, können wir zeigen, dass der Graph zsh. ist. (bzw. sogar dass es zwischen 2 beliebigen Knoten immer mindestens 1 Pfad der Länge ≤ 2 gibt).

2. Sei $P = \langle v_1, \dots, v_k \rangle$ der längste Pfad im Graph.

Betrachten wir v_1 und v_k:
Alle Nachbarn von v_1 und v_k sind im Pfad P, sonst könnten wir einen längeren Pfad konstruieren. Widerspruch!

Wir suchen nun ein v_{i-1} und v_i im Pfad, so dass $\{v_1, v_i\} \in E$ und $\{v_{i-1}, v_k\} \in E$ (i.e. $v_i \in N(v_1)$, $v_{i-1} \in N(v_k)$)
Denn dann könnten wir aus dem Pfad P ein Kreis $\langle v_1 v_i, \dots, v_k, v_{i-1}, \dots, v_1 \rangle$ konstruieren.



(*)

Wir zeigen nun, dass es immer solch ein Paar v_{i-1} und v_i gibt.

Sei $N_k^+ = \{v_i \text{ in } P \mid v_{i-1} \in N(v_k)\}$ die Menge der Knoten die im Pfad P auf einen Knoten aus der Nachbarschaft von v_k folgen.

Beachte, dass v_1 nicht in N_k^+ sein kann.

Aus $N(v_1), N_k^+ \subseteq \underbrace{V \setminus \{v_1\}}_{|V \setminus \{v_1\}| = n-1}$ und $|N_k^+| \geq \frac{n}{2}$ und $|N(v_1)| \geq \frac{n}{2}$

folgt per Pigeonhole Principle $N_k^+ \cap N(v_1) \neq \emptyset$.

I. e. $\exists\, v_i \in N(v_1) \cap N_k^+$ wobei v_i der i-te Knoten in P

Da $v_i \in N_k^+ \Rightarrow v_{i-1} \in N(v_k)$, haben wir nun ein

Paar $v_i \in N(v_1)$ und $v_{i-1} \in N(v_k)$ gefunden.

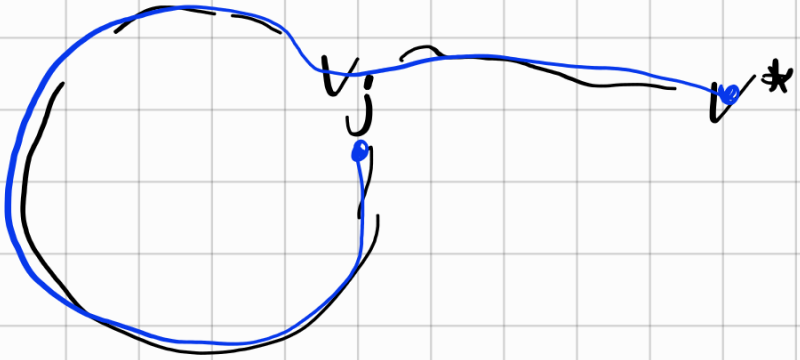
Folglich können wir immer (*) konstruieren.

3. Case Distinction

1. Kreis beinhaltet alle Knoten
   $\Rightarrow$ Hamiltonkreis

2. $\exists\, v^*$ nicht im Kreis

   Da aber G zsh. ist (1.), muss er mindestens mit mit einem Knoten v_j im Kreis verbunden sein. (Möglicherweise über einen Pfad mit mehreren Knoten, die nicht im Kreis sind, sonst direkte Kante)



Dann gäbe es aber ein Pfad P' der Länger wäre als P!
Dies widerspricht dem, dass P der längste Pfad in G ist.
Also kann nur der erste Fall möglich sein.

□

Hamilton DP

Wir könnten alle möglichen Reihenfolge der n Knoten ausprobieren.

Bruteforce $\approx n!$

Mit DP $\approx 2^n$ (genau $O(n^2 2^n)$)

Notation: $[n] = \{1, \dots, n\}$, $\binom{V}{2} = \{ X \subseteq V \mid |X| = 2 \}$

$N(y) = \{ x \in V \mid \{x, y\} \in E \}$ (Nachbarschaft von y)

Wir benutzen eine $2^n \times n$ Tabelle P mit den Einträgen:

$$P_{S,x} := \begin{cases} 1, & \text{es gibt in G einen 1-x-Pfad, der genau die Knoten aus S enthält} \\ 0, & \text{sonst.} \end{cases}$$

wobei $S \subseteq V$.

Wir verwenden um eine Lösung zu finden, folgendes:

G enthält einen Hamiltonkreis $\iff$ $\exists x \in N(1)$ mit $P_{[n],x} = 1$

enthält alle Knoten von [n] genau einmal

Wir haben folgenden Pseudocode:

HAMILTONKREIS $(G = ([n], E))$

1: *// Initialisierung*
2: **for all** $x \in [n], x \neq 1$ **do**
3: $P_{\{1,x\},x} := \begin{cases} 1, & \text{falls } \{1,x\} \in E \\ 0, & \text{sonst} \end{cases}$ // alle Subsets $|S|=2$ als Base case
4: *// Rekursion*
5: **for all** $s = 3$ **to** n **do**
6: **for all** $S \subseteq [n]$ mit $1 \in S$ und $|S| = s$ **do**
7: **for all** $x \in S, x \neq 1$ **do** ← Rekurrenz auf Mengen S mit $|S| = s-1$
8: $P_{S,x} = \max\{P_{S\setminus\{x\},x'} \mid x' \in S \cap N(x), x' \neq 1\}.$
9: *// Ausgabe*
10: **if** $\exists\, x \in N(1)$ mit $P_{[n],x} = 1$ **then**
11: **return** G enthält Hamiltonkreis
12: **else**
13: **return** G enthält keinen Hamiltonkreis

← Wir dürfen x nicht bei 1 anhängen.

In Essenz, könnte man die Rekurrenz wie folgt erklären:

Wenn wir $P_{S,x}$ berechnen wollen, haben wir schon alle Einträge $P_{B,y}$ mit $|B| < |S|$ (insbesondere $|B| = |S|-1$) berechnet.

$\Rightarrow$ für $x \in S$ können wir folgendes feststellen:

$\exists$ 1-x Pfad mit den Knoten in S, falls

ein $y \in S\setminus\{x\}$ existiert, für das es ein 1-y Pfad

mit den Knoten in $S \setminus \{x\}$ gibt.

Daraus folgt die markierte Rekurrenz im Pseudocode.

Kleiner Zusatz:

Wie iterieren wir durch alle Teilmengen $S \subseteq [n]$ der Grösse s?

Folgende Überlegung:

Für einen Graphen mit n Knoten betrachten wir ein Bitstring der Länge n, um die möglichen S darzustellen:

Bsp $n=5$: $\Rightarrow$ Bitstring 10010 beschreibt $\{5,2\}$
00000 " $\{\}$
01011 " $\{4,2,1\}$

i.e. Wenn das i-te Bit 1 ist, gilt $i \in S$
sonst $i \notin S$.

$\Longrightarrow$ Alle möglichen S mit Grösse s, haben Bitstrings mit genau s 1-en.

(z. Bsp. $n=3$: alle möglichen S mit $|S|=2$
011, 101, 110)

Nun können wir noch diese Bitstrings als Bitrepräsentation von Integers betrachten. (reichen von 0 bis 2^n-1)

k-te Stelle 1 $\Rightarrow$ kes

$\Rightarrow$ für P[i][x] $\rightsquigarrow$ $i = \ldots 0100 \ldots 1_2$

Um dies in Java umzusetzen, braucht ihr gewisse Bit-manipulationen:

z.B.
Um zu testen ob das i-te Bit eines Integers 1 ist (ies?)

```
boolean inSet(int S, int i){
    return (S & (1 << (i-1))) != 0;
}
```

Ausserdem kann man `Integer.bitCount(s)` verwenden,

um die Anzahl 1 in der Bitrep. von s finden.

Aufgabe 1 – *Hamiltonkreise (In-Class Exercise)*

Sei $G = (V, E)$ ein Graph mit n Knoten, $V = \{v_1, \ldots, v_n\}$. Sei W die Menge aller geschlossenen Wege von v_1 nach v_1 der Länge n in G. Für $2 \leq i \leq n$ sei $A_i \subseteq W$ die Menge der Wege in W, die nicht den Knoten v_i besuchen. Ausserdem sei $A := \bigcup_{i=2}^{n} A_i$.

(a) Sei H die Menge der Hamiltonkreise in V mit Start-/Endpunkt v_1. In welcher Beziehung stehen die Mengen H, W, und A zueinander? Welche Gleichung erfüllen demzufolge die Grössen $|H|$, $|W|$, und $|A|$?

(2) (1)

$W = H \cup A$ wobei $H \cap A = \emptyset$.

$\Rightarrow |W| = |H| + |A|$

Beweis: Jeder Weg $w \in A$ besucht mind. 1 Knoten nicht $\Rightarrow$ $w \notin H \Rightarrow H \cap A = \emptyset$. (1)

$H \cup A \subseteq W$:

Jeder Weg $w \in H$ ist ein geschlossener Weg der Länge n von v_1 nach v_1. $\Rightarrow$ $w \in W \Rightarrow H \subseteq W$

$A_i \subseteq W \quad \forall\, 2 \leq i \leq n$ per Definition

$\Rightarrow A = \bigcup_{i=2}^{n} A_i \subseteq W$

$\Rightarrow H \cup A \subseteq W$

$W \subseteq H \cup A$:

Sei $w \in W$ beliebig.

Case Distinction

v_1 in w per Def.

Case $\exists\, 2 \leq i \leq n$ s.d. v_i nicht in w

$\Rightarrow w \in A_i \Rightarrow w \in A$

Case w besucht alle Knoten.

Da w Länge n hat und Start-/Endknoten v_1

$\Rightarrow$ w ist Hamiltonkreis mit $\_\_\_\_\_ \circ$

$\Rightarrow$ $w \in H$

$\Rightarrow$ $W \subseteq A \cup H$

$\Rightarrow$ $W = A \cup H$ (2)

□

(b) Für jede Menge $S \subseteq V$ mit $v_1 \notin S$ definieren wir W_S als die Menge aller Wege in W, die keine Knoten in S besuchen. Insbesondere ist $A_i = W_{\{v_i\}}$. Zeigen Sie:

$$|A| = \sum_{\emptyset \neq S \subseteq V, v_1 \notin S} (-1)^{|S|+1} |W_S|.$$

Das erinnert uns an:

Satz 1.35. *(Siebformel, Prinzip der Inklusion/Exklusion)* Für endliche Mengen $A_1, \ldots, A_n$ $(n \geq 2)$ gilt:

$$\begin{aligned}\left|\bigcup_{i=1}^{n} A_i\right| &= \sum_{l=1}^{n} (-1)^{l+1} \sum_{1 \leq i_1 < \cdots < i_l \leq n} |A_{i_1} \cap \cdots \cap A_{i_l}| \\ &= \sum_{i=1}^{n} |A_i| - \sum_{1 \leq i_1 < i_2 \leq n} |A_{i_1} \cap A_{i_2}| + \sum_{1 \leq i_1 < i_2 < i_3 \leq n} |A_{i_1} \cap A_{i_2} \cap A_{i_3}| \\ &\quad - \cdots + (-1)^{n+1} \cdot |A_1 \cap \cdots \cap A_n|.\end{aligned}$$

$$|A| = \left|\bigcup_{i=2}^{n} A_i\right| = \sum_{\emptyset \neq S \subseteq \{2,\ldots,n\}} (-1)^{|S|+1} \left|\bigcap_{i \in S} A_i\right|$$

Jetzt müssen wir noch $W_S = \bigcap_{i \in S} A_i$ beweisen.

$\bigcap_{i \in S} A_i \subseteq W_S$

Sei: $w \in \bigcap_{i \in S} A_i$ beliebig.

$\Rightarrow w \in W$ besucht Knoten i nicht für alle $i \in S$.

$\Rightarrow w \in W_S$

$W_S \subseteq \bigcap_{i \in S} A_i$

Sei $w \in W_S$.

$\Rightarrow w \in W$ besucht keine Knoten in S.

$\Rightarrow \forall i \in S$: w besucht i nicht

$\Rightarrow \forall i \in S$: $w \in A_i \Rightarrow w \in \bigcap_{i \in S} A_i$

□

(c) Geben Sie einen (effizienten) Algorithmus an, der als Input G, v_1 und S bekommt, und der $|W_S|$ berechnet. Welche Laufzeit hat Ihr Algorithmus?
Hinweis: Benutzen Sie Algorithmen als Subroutinen, die Sie aus diesem oder letztem Semester kennen. Von solchen aus den Vorlesungen bekannten Routinen brauchen Sie natürlich weder Implementierung noch Korrektheit neu zu diskutieren. Achten Sie jedoch wie immer darauf, dass Sie genau angeben, was Sie der Subroutine als Input geben.

Gesucht: # Wege $v_1 \to v_1$ der Länge n im induzierten Teilgraphen $G_S := G[V \setminus S]$

Ähnlich wie Floyd-Warshall

1. Konstruiere Adjazenzmatrix A_{G_S} durch entfernen der i-ten Reihe und i-ten Kolonne für alle $i \in S$ von A_G.

2. Berechne $R := (A_{G_S})^n$ durch $O(\log n)$ Matrixmults. (Strassen)

3. Return $(R)_{1,1}$.

Laufzeit: $O(n^{2.807})$ per Strassen Matrixmult.

$\Rightarrow O(n^{2.807} \log n)$ ☑

(d) Entwerfen Sie einen Algorithmus, der als Input G und v_1 bekommt, und der $|A|$ berechnet. Der Algorithmus darf exponentielle Laufzeit haben (jedoch höchstens um einen polynomiellen Faktor schlechter als 2^n), aber er soll nur polynomiell viel Speicherplatz benötigen.

Hint: Benutze (b) und (c).

For $\emptyset \neq S \subseteq \{2, \dots, n\}$ do
1. Compute $|W_S|$ with (c).
2. Add $(-1)^{|S|+1} |W_S|$ to Accumulator A'

Return A'

Correctness: by (b) we have $A' = |A|$.

Speicher: $O(n^2)$ für den Algorithmus in (c).
(kann bei jedem neuen Aufruf neu überschrieben werden)

und da $A' \leq 2^n \cdot n^n$
brauchen wir maximal $O(n \log n)$ Speicher
($\log(2^n n^n) = \log(2^n) + \log n^n = n \log 2 + n \log n$)

Laufzeit: $O(2^n \cdot n^{2.807} \cdot \log n)$
(2^n: Anzahl Subsets; $n^{2.807} \cdot \log n$: (c))

(e) Entwerfen Sie einen Algorithmus, der als Input G bekommt, und der die Zahl der Hamiltonkreise in G bestimmt. Der Algorithmus soll dieselben Laufzeit- und Speicheranforderungen erfüllen wie in (d).

1. Berechne $|W|$ mit (c) (Input: $G, v_1, \emptyset$)
2. Berechne $|A|$ mit (d)
3. Return $|W| - |A|$

Correctness: Folgt von (d), (c) und (a)

Laufzeit: Gleich wie (d), da $|W|$ in poly-zeit berechnet.

Speicher: $O(n^2)$ wie in (d) beschrieben.

(f) Entwerfen Sie einen Algorithmus, der entscheidet, ob es in einem Graphen G einen Hamiltonkreis gibt, und der dieselben Laufzeit- und Speicheranforderungen erfüllt wie in (d).

Teste ob (e) 0 zurückgibt.

Korrektheit/Laufzeit/Speicher folgt aus (e)

P vs NP , NP-Vollständigkeit

P ist die Klasse aller Probleme, die in polynomieller Zeit lösbar sind. ($in\ O(n^c),\ c \in \mathbb{R}$)

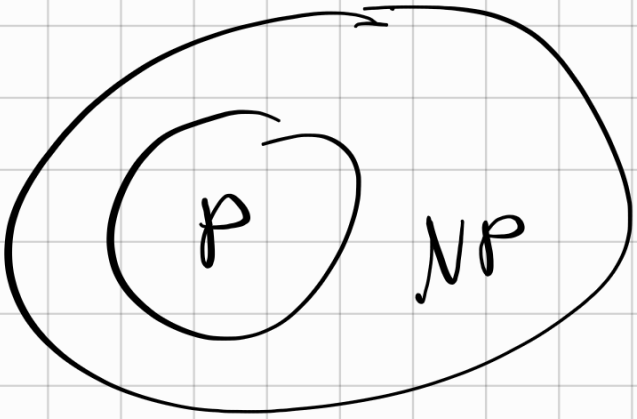
Bsp.
- One-to-All shortest Path
- All-to-All shortest Path
- Minimum Spanning Tree
- ⋮

$\} \in P$

NP ist die Klasse aller Probleme, die nichtdeterministisch polynomiell lösbar sind.

Bsp:
- Hamiltonkreis
- Satisfiability
- ⋮

$\} \in NP$



Ein Problem π aus NP heisst NP-vollständig, falls gilt

$$\pi \in P \quad \Rightarrow \quad P = NP$$

i.e. falls TT deterministisch polynomiell lösbar ist, sind alle Probleme in NP det. polynomiell lösbar.

Satz: (Karp 1972)
Das Problem

„Gegeben ein Graph G = (V, E), enthält G einen Hamiltonkreis?"

ist NP-vollständig.

Traveling Salesman Problem (TSP)

Gegeben: vollständiger Graph auf n Knoten,
Distanzen zw. je zwei Knoten: $\ell : \binom{[n]}{2} \to \mathbb{R}$
Gesucht: Kürzeste Rundreise:

$$\min_{H:\ \text{Hamiltonkreis}} \sum_{e \in E(H)} \ell(e)$$

Wir können zeigen: TSP ist NP-Vollständig

Dazu beweisen wir, dass wir das Hamiltonkreisproblem auf TSP reduzieren können. I.e. TSP $\in$ P $\Rightarrow$ Hamiltonk. $\in$ P

Graph $G=([n], E)$ $\iff$ vollst Graph $K_n=([n], \binom{[n]}{2})$

mit Längenfunktion $\ell : \binom{[n]}{2} \to \{0, 1\}$

so dass $\ell(\{x, y\}) = \begin{cases} 0, & \text{falls } \{x, y\} \in E \\ 1, & \text{sonst.} \end{cases}$

In einem vollst. Graph gibt es sicher ein HK. und

ein TSP-Algorithmus würde, den mit dem minimalsten Gewicht finden:

G enthält Hamiltonkreis $\iff$ $\text{opt}(K_n, \ell) = 0$

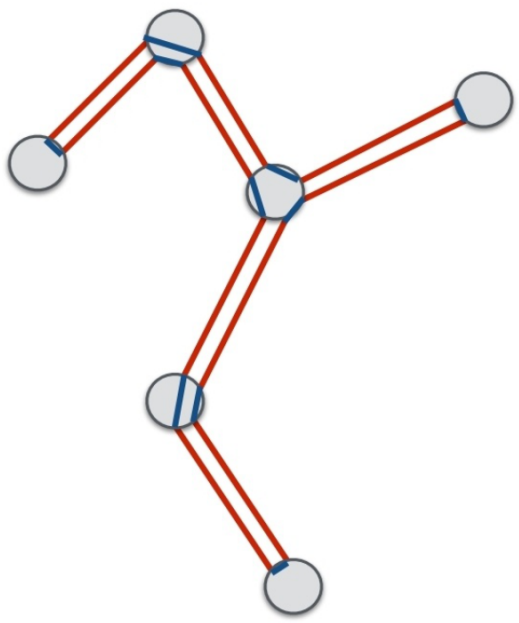
Wenn wir einen poly. C-Approximationsalgorithmus hätten, könnten wir das HK-Problem in polynomieller Zeit lösen.

(Wenn das $\text{opt}(K_n, \ell) = 0$, dann ist eine C-mal schlechtere Lösung immer noch 0)

Hieraus folgt auch: für kein C >0 kann es einen polynominellen C-Approximationsalgorithmus geben (ausser es gilt P=NP).

Nun betrachten wir das metrische Traveling Salesman Problem. Konkret:

Funktion ℓ erfüllt Dreiecksungleichung: $\ell(x,z) \leq \ell(x,y) + \ell(y,z) \qquad \forall x, y, z \in [n]$



1. Bestimme minimalen Spannbaum **T**

   es gilt: $\ell(T) \leq \text{opt}(K_n, \ell)$

2. verdopple alle Kanten von **T**

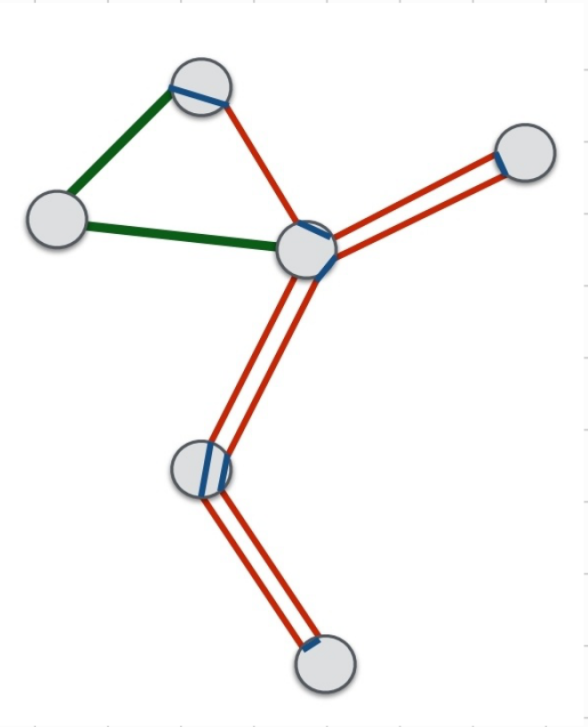
   es gilt: $2\ell(T) \leq 2\text{opt}(K_n, \ell)$

3. bestimme Eulertour **W**

   es gilt: $\ell(W) = 2\ell(T) \leq 2\text{opt}(K_n, \ell)$

Nun noch der 4. te Schritt

4. durchlaufe **W**, mit Abkürzungen, so dass jeder Knoten nur einmal besucht wird ⇒ Hamiltonkreis **C**



Man erinnere sich daran, dass der gegebene Graph vollständig ist und

dass die metrische Eigenschaft gilt:

Funktion ℓ erfüllt Dreiecksungleichung: $\ell(x,z) \leq \ell(x,y) + \ell(y,z) \qquad \forall x,y,z \in [n]$

↳ Aus dieser Eigenschaft folgt:

$$\ell(C) \leq \ell(W) \leq 2 \cdot opt(K_n, \ell)$$

Matchings

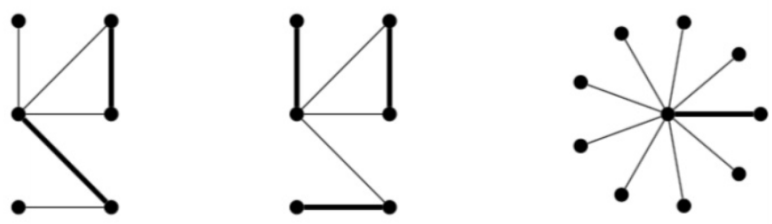
Eine Kantenmenge $M \subseteq E$ heisst **Matching** in einem Graphen $G = (V, E)$, falls kein Knoten des Graphen zu mehr als einer Kante aus M inzident ist.

Anwendung: 2-er Gruppen für Theorieaufgaben bilden ☺

Ein Knoten v wird von M **überdeckt**, falls es eine Kante $e \in M$ gibt, die v enthält.

... a —e— b ... $e \in M \Rightarrow$ a und b überdeckt

Ein Matching M heisst **perfektes Matching**, wenn jeder Knoten durch genau eine Kante aus M überdeckt wird, oder, anders ausgedrückt, wenn $|M| = |V|/2$.



Maximale Matchings

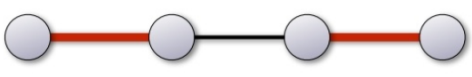
Ein Matching $M \subseteq E$ ist ein **inklusionsmaximales Matching**, wenn es kein Matching M' gibt mit $M \subseteq M'$ und $|M'| > |M|$.

"mit den Kanten, die ich in Matching habe, geht es nicht besser" (≈ das relativ Beste)

Ein Matching $M \subseteq E$ ist ein **(kardinalitäts-)maximales Matching**, wenn es kein Matching M' gibt mit $|M'| > |M|$.

"es geht allg. nicht besser" (≈ das absolut Beste)

inklusionsmaximales Matching
(aber nicht kardinalitätsmaximal)



kardinalitätsmaximales Matching
(auch inklusionsmaximal!)

Greedy Matching

```
GREEDY-MATCHING (G)
1: M ← ∅
2: while E ≠ ∅ do
3:     wähle eine beliebige Kante e ∈ E
4:     M ← M ∪ {e}
5:     lösche e und alle inzidenten Kanten in G
```

Satz: Mit dem Greedy-Algorithmus kann man in Zeit O(|E|) ein inklusionsmaximales Matching M_{Greedy} bestimmen mit

$$|M_{Greedy}| \geq |M_{max}| \ / \ 2,$$

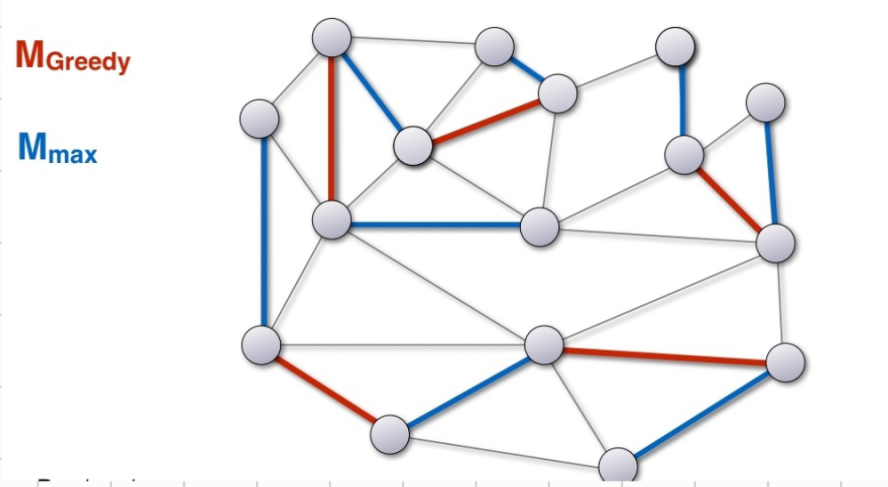
wobei M_{max} ein kardinalitätsmaximales Matching ist.

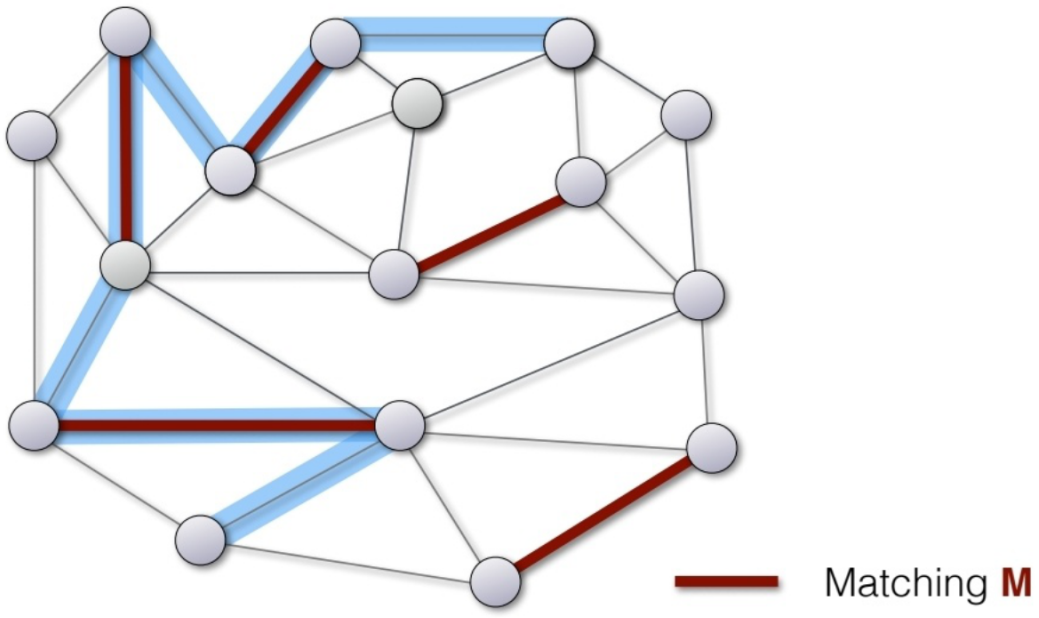
Beweisidee:
Wir sehen, dass M_{Greedy} inklusionsmaximal ist.

(1) Daraus folgt, dass jede Kante in M_{max} entweder in M_{Greedy} sein muss oder zu mind. 1 Kante von M_{Greedy} inzident ist.
(Sonst könnten wir die Kante zu M_{Greedy} hinzufügen. Widerspruch ↯)

(2) Da M_{max} ein Matching ist, berührt jeder Knoten in M_{Greedy} (wovon es $2 \cdot |M_{Greedy}|$ gibt), maximal eine Kante von M_{max}.

$$(1),(2) \implies |M_{Greedy}| \geq \frac{1}{2} |M_{max}|$$





Ein **M-augmentierender Pfad** ist ein Pfad, der abwechselnd Kanten aus M und nicht aus **M** enthält und der in von **M** nicht überdeckten Knoten beginnt und endet.

⇒ durch *tauschen* entlang M können wir das Matching vergrössern

Matching finden

Mit augmentierenden Pfaden:

Nur bipartiten Graphen: $O(|V| \cdot |E|)$

State of the Art:

$O(|V|^{\frac{1}{2}} \cdot |E|)$ für ungewichtete Graphen

$O(|V|^3)$ für gewichtete Graphen

Für das metrische TSP gilt:

aus MST & Matching & Eulertour kann man eine 3/2-Approximation ableiten

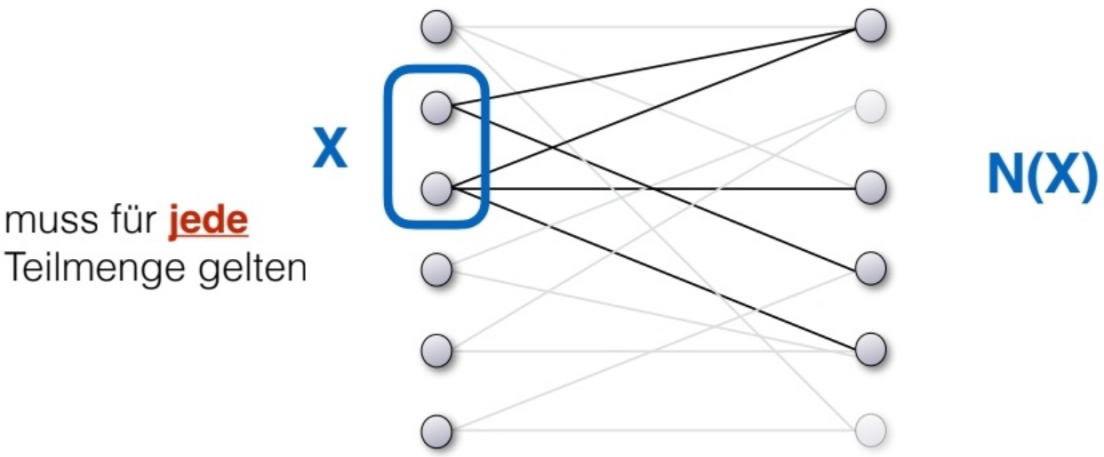
Satz von Hall (Einführung):

Satz: (Hall, Heiratssatz)

Ein bipartiter graph $G=(A \dot\cup B, E)$ enthält ein Matching M der Kardinalität $|M|=|A|$ **gdw**

$\forall X \subseteq A : |X| \leq |N(X)|$

Philip Hall
(1904-1982)



Beweis vllt. nächstes Mal