Quiz

Consider the Subset Sum problem for a given array $A[1..n]$ of non-negative integers.
The subproblem $T(i, s)$ is defined as 1 if and only if there exists a subset of the first i elements that sum up to s.

Which of the following recurrences is correct?

wir brauchen A[i] nicht — wir brauchen A[i], d.h. es muss eine Teilsumme der ersten i-1 Elemente geben, die zu s - A[i] summiert.

- [x] a. $T(i,s) = T(i-1,s) \vee T(i-1, s-A[i])$
- [ ] b. $T(i,s) = T(i-1,s) \vee T(i, s-A[i])$
- [ ] c. $T(i,s) = T(i-1, s-A[i])$
- [ ] d. $T(i,s) = T(i-1,s) \wedge T(i, s-A[i])$

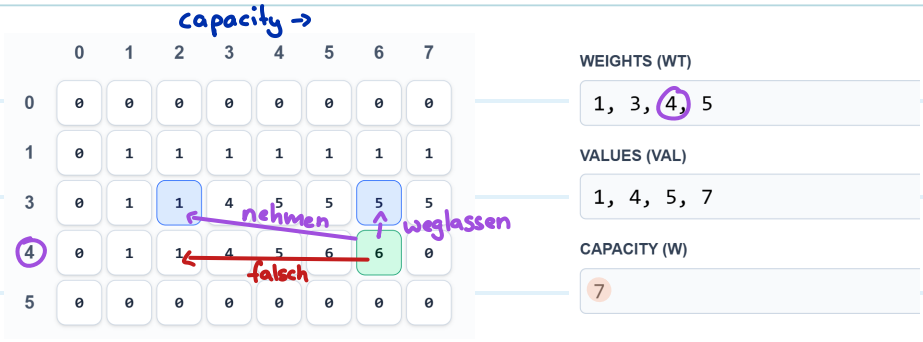
Consider the Knapsack Problem. Let $P(i, w)$ be the maximum profit obtainable using the first i items with total weight at most w. You may assume that $w \geq w_i$.
Is the following recurrence correct?

item i weglassen — item i nehmen

$$P(i,w) = \max\{P(i-1,w),\ p_i + P(i, w-w_i)\}.$$

i-1

- [ ] a. Yes, because it considers both including and excluding item i
- [x] b. No, because it may allow item i to be selected more than once.



Assuming that $P \neq NP$, which of the following statements are true or false?

True	False	
○	[x]	There is a polynomial-time algorithm for the Knapsack Problem.
[x]	○	There is a polynomial-time algorithm for the Knapsack problem when all profits are integers between 1 and 100. $O(n\,P) \leq O(n \cdot 100n) \leq O(n^2)$ weil $P = \sum_{i=1}^{n} p_i \leq n \cdot 100$
[x]	○	There is a polynomial-time algorithm for the Knapsack problem when all weights are integers between 1 and 100. $O(n\,W) \leq O(n \cdot 100n) \leq O(n^2)$ weil $W \leq \sum_{i=1}^{n} w_i \leq n \cdot 100$

(falls $W > \sum_{i=1}^{n} w_i$ können wir alle Elemente packen → trivial)

The algorithm for Longest Ascending Subsequence has computed the following DP table. Here, $M(i, \ell)$ is the smallest possible ending value of an ascending subsequence of length ℓ within a given array $A[1..i]$.

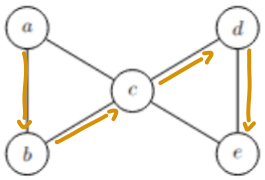
$i \backslash \ell$	1	2	3	4	5	6
1	3	∞	∞	∞	∞	∞
2	3	7	∞	∞	∞	∞
3	3	7	8	∞	∞	∞
4	3	4	8	∞	∞	∞
5	3	4	5	∞	∞	∞
6	3	4	5	9	∞	∞

A was [3, 7, 8, 4, 5, 9]

What is the length of a longest ascending subsequence of $A[1..6]$?

Answer: 4

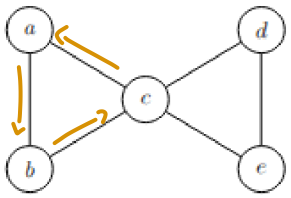
Consider the graph shown in the figure below. Each vertex is labeled with its name inside.



Consider the vertex sequence $W_1 = (a, b, c, d, e)$. Which of the following are true?

True	False	
☒	○	W_1 is a walk.
☒	○	W_1 is a path.
○	☒	W_1 is a circuit. must be closed
○	☒	W_1 is a cycle. must be closed

Consider the same graph from the previous question. Consider now the vertex sequence $W_2 = (a, b, c, a)$. Which of the following are true?



True	False	
☒	○	W_2 is a walk.
○	☒	W_2 is a path. a is not allowed to occur twice
☒	○	W_2 is a circuit
☒	○	W_2 is a cycle.

There is a graph with exactly five vertices, whose degrees are given by $1, 2, 2, 2$ and 4.

○ True

☒ False

$\sum_{v \in V} \deg(v) = 2|E|$

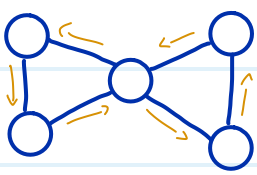
Summe aller Knotengrade — ist eine gerade Zahl

1+2+2+2+4 = 11 → ungerade

Which of the following are true?

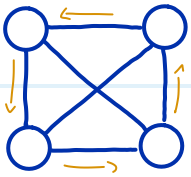
True	False	
○	☒	Every graph that has an Euler tour also has a Hamiltonian cycle.
○	☒	Every graph that has a Hamiltonian cycle also has an Euler tour.

has Eulertour



but no Hamiltonian cycle

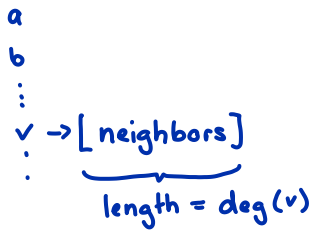
has Hamiltonian cycle



but no Eulertour (as it has two nodes with odd degree)

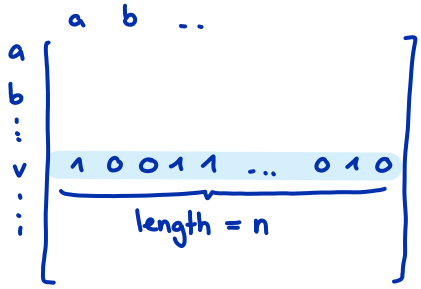
A graph with n vertices and m edges is stored using adjacency lists. In general, what is the time needed to enumerate all neighbors of a vertex v?

- a. $\Theta(1)$
- b. $\Theta(1 + \deg(v))$ → falls deg(v) = 0 brauchen wir Θ(1) Zeit um das herauszufinden
- c. $\Theta(n)$
- d. $\Theta(n + m)$



A graph with n vertices and m edges is stored using an adjacency matrix. In general, what is the time needed to enumerate all neighbors of a vertex v?

- a. $\Theta(1)$
- b. $\Theta(1 + \deg(v))$
- c. $\Theta(n)$
- d. $\Theta(n + m)$



We are given a connected graph in which every vertex has even degree. The graph is stored using adjacency lists. Then we can output an Euler tour in this graph in $O(n + m)$ time.

- True — Hierholzer's Algorithm
- False

Discussion of Exercise Sheet 7

Exercise 7.4 *String counting* **(2 bonus points).** → Peergrading Aufgabe für heute

Given a binary string $S \in \{0,1\}^n$ of length n, let $f(S)$ be the number of times "11" occurs in the string, i.e. the number of times a 1 is followed by another 1. In particular, the occurrences do not need to be disjoint. For example $f(\text{“}\underline{11}10\underline{11}\text{”}) = 3$ because the string contains three underlined 1s that are followed by another 1. Given n and k, the goal is to count the number of binary strings S of length n with $f(S) = k$.

Describe a DP algorithm that, given positive integers n and k with $k < n$, reports the required number. Your solution should have complexity at most $O(nk)$.

Dimension: DP[1...n][0...k][0..1]

Teilproblem: DP[i, j, ℓ] := number of strings of length i with j occurences of "11" that end in ℓ

Base Case:

DP[1, 0, 0] = 1 → the string "0"

DP[1, 0, 1] = 1 → the string "1"

DP[1, j, 0] = 0 für 1 ≤ j ≤ k → a string of length 1 cannot contain "11"

DP[1, j, 1] = 0 für 1 ≤ j ≤ k → a string of length 1 cannot contain "11"

Rekursion:

Fall ℓ = 0: $DP[i,j,0] = \underbrace{DP[i-1,j,0] + DP[i-1,j,1]}_{\text{alle strings der Länge } i-1 \text{ mit } j \text{ mal "11" und beliebiger Endung}}$

Fall ℓ = 1: $DP[i,j,1] = \underbrace{DP[i-1,j,0]}_{\text{die 1 gehört nicht zu einem "11" Paar}} + \overbrace{DP[i-1,j-1,1]}^{:=0 \text{ falls } j=0}$ (die 1 gehört zu einem "11" Paar)

Berechnungsreihenfolge:

```
for i = 1 .. n
    for j = 0 .. k
        for ℓ = 0 .. 1: berechne DP[i, j, ℓ]
```

Lösung: DP[n, k, 0] + DP[n, k, 1]

Laufzeit: Θ(nk)

Graphen

Graph $G = (V, E)$ mit $n = |V|$ Knoten und $m = |E|$ Kanten

Knotenmenge / vertices / nodes V, darf nicht leer sein. oft ist $V = \{1, 2, \dots, n\}$

Kantenmenge / edges / links: E wobei $E \subseteq \{ \{u, v\} \mid u \in V, v \in V, u \neq v \}$

- $\{u, v\}$: Kante zwischen u und v
- $u \neq v$: es darf keine Schleifen geben

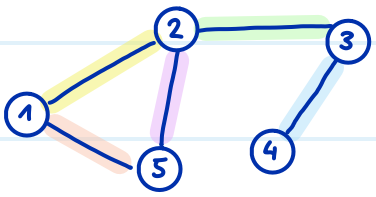
Benachbart / adjacent Knoten u und v sind benachbart, falls $\{u, v\} \in E$

Anliegend / incident: Kante $\{u, v\}$ ist zu u und v anliegend

Grad / degree: $\deg(v)$ = Anzahl Kanten, die zu v anliegen

Bsp: $V = \{1, 2, 3, 4, 5\}$

$E = \{\{1,2\}, \{2,3\}, \{3,4\}, \{2,5\}, \{1,5\}\}$



→ 2 und 3 sind benachbart

→ $\{3,4\}$ ist zu 3 und 4 anliegend

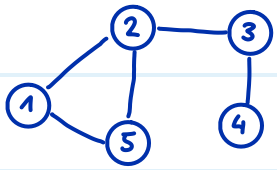
→ $\deg(5) = 2$, $\deg(4) = 1$

Erreicht / reaches: Knoten u erreicht v, falls es einen Weg von u nach v gibt

Zusammenhängend / connected: Jeder Knoten u erreicht jeden Knoten v

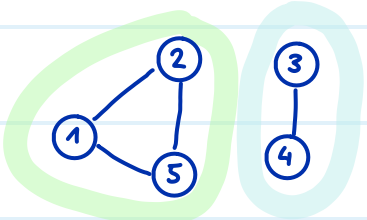
Zusammenhangskomponente (ZHK) / connected component: Eine Äquivalenzklasse der Relation "u erreicht v"

Bsp:



→ zusammenhängend

→ 2 erreicht 4



$ZHK_1 = \{1, 2, 5\}$

$ZHK_2 = \{3, 4\}$

Weg / walk: Eine Folge von $\ell+1$ Knoten $(v_0, v_1, .., v_\ell)$, sodass es alle zwei aufeinanderfolgende Knoten durch eine Kante verbunden sind

Die Länge ℓ ist die Anzahl der Kanten

Knoten und Kanten dürfen mehrfach verwendet werden.

Pfad / path: Ein Weg, der keinen Knoten mehrfach verwendet

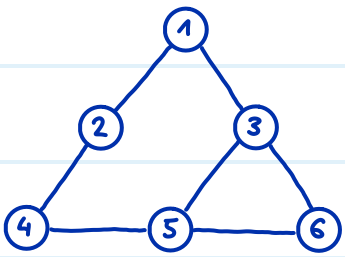
Geschlossener Weg / closed walk: Ein Weg sodass $v_0 = v_\ell$ (Startknoten = Endknoten)

Zyklus / circuit: Ein geschlossener Weg, der keine Kante mehrfach verwendet, und Länge ≥ 1 hat

Kreis / cycle: Ein Zyklus, der keinen Knoten mehrfach verwendet ausser $v_0 = v_\ell$

Achtung: Zyklus ≠ Cycle

Aufgabe:



Wie viele ZHKs gibt es? 1

Wie viele verschiedene Pfade von 1 nach 6 gibt es? 4: (1,3,6), (1,3,5,6), (1,2,4,5,6), (1,2,4,5,3,6)

Wie viele verschiedene Wege von 1 nach 6 gibt es? ∞

Wie viele verschiedene Pfade der Länge 4 von 1 nach 6 gibt es? 1: (1,2,4,5,6)

Wie viele verschiedene Wege der Länge 3 von 1 nach 3 gibt es? 4: (1,2,1,3), (1,3,1,3), (1,3,5,3), (1,3,6,3)

Wie viele verschiedene Kreise gibt es? 3: (3,5,6,3), (1,3,5,4,2,1), (1,3,6,5,4,2,1)

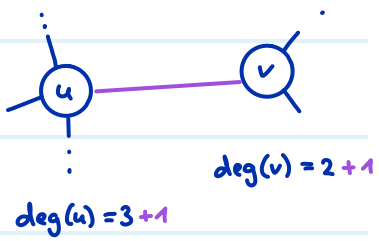
Welche mögliche Längen kann ein Weg von 1 nach 5 haben? {2,3,4,5,6,..}

Wie viele verschiedene Zyklen gibt es? 3: (3,5,6,3), (1,3,5,4,2,1), (1,3,6,5,4,2,1)

Handschlag-Lemma: $\sum_{v \in V} \deg(v) = 2\,|E|$

Summe aller Knotengrade — 2 · Anzahl Kanten

Intuition: Jedes mal, wenn wir eine neue Kante $\{u, v\}$ hinzufügen, werden $\deg(u)$ und $\deg(v)$ genau um eins grösser, d.h. die Summe aller Knotengerade wird genau um 2 grösser.

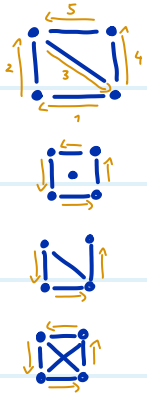


Eulerweg: Ein Weg, der jede Kante genau einmal verwendet

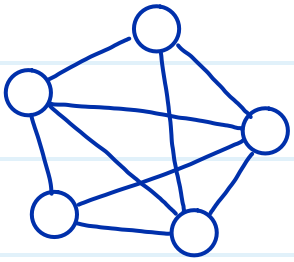
Eulerzyklus: Ein Zyklus, der jede Kante genau einmal verwendet

Hamiltonpfad: Ein Pfad, der jeden Knoten genau einmal verwendet

Hamiltonkreis: Ein Kreis, der jeden Knoten genau einmal verwendet



Aufgabe:

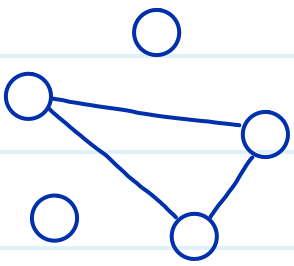


Eulerweg? ✓

Eulerzyklus? ✗

Hamiltonpfad? ✓

Hamiltonkreis? ✓

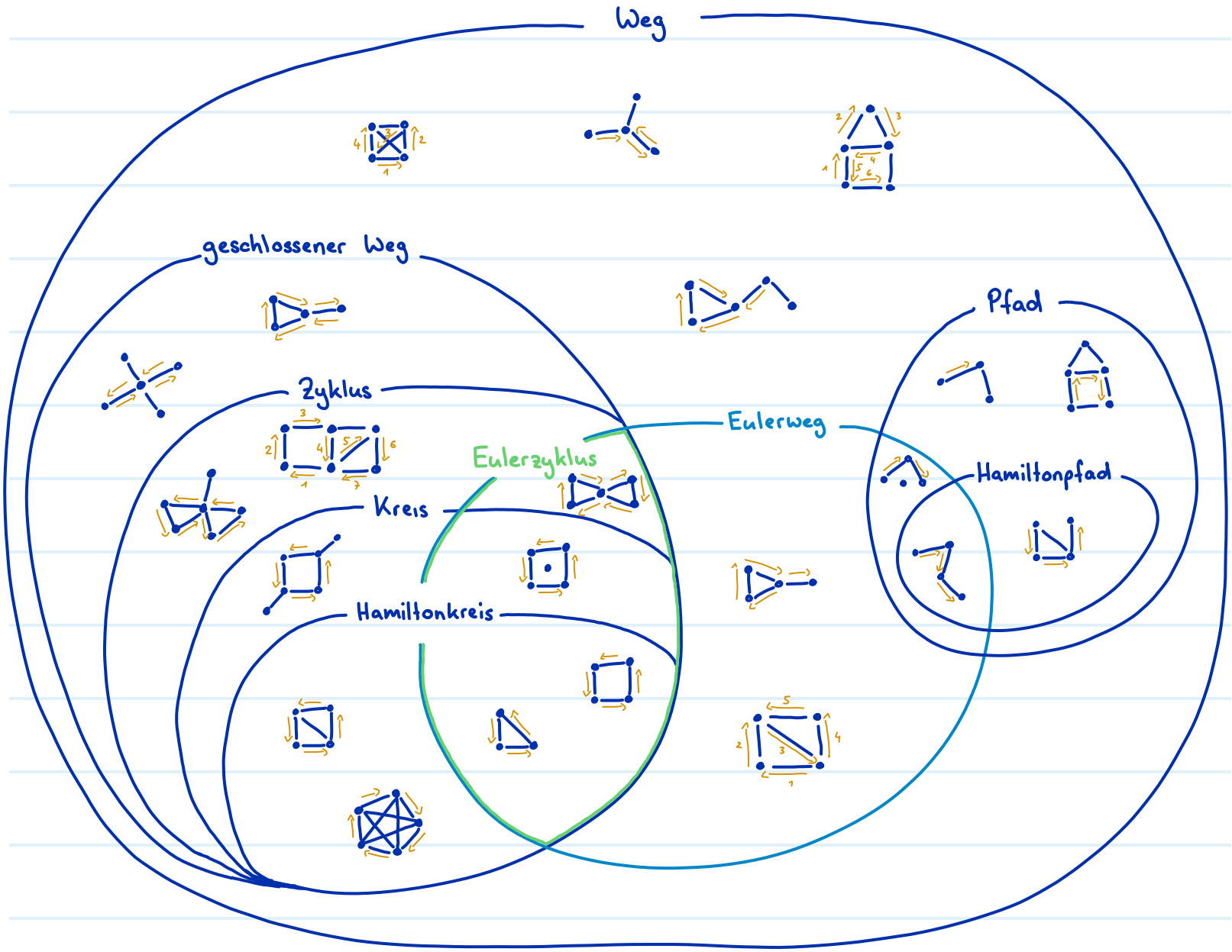


Eulerweg? ✓

Eulerzyklus? ✓

Hamiltonpfad? ✗

Hamiltonkreis? ✗


Weg
geschlossener Weg
Zyklus
Kreis
Hamiltonkreis
Eulerzyklus
Eulerweg
Pfad
Hamiltonpfad

Satz von Euler: Sei G ein zusammenhängender Graph Dann gilt:

G hat einen Eulerzyklus <=> Alle Knotengerade sind gerade

Intuition wir laufen in jeden Knoten gleich oft hinein wie hinaus

Allgemein: Sei G ein Graph Dann gilt:

G hat einen Eulerzyklus <=> Alle Knotengerade sind gerade und

alle Kanten sind in derselben ZHK

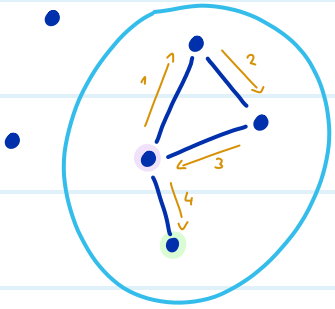
G hat einen Eulerweg <=> Alle Knotengerade sind gerade ausser höchstens zwei und

alle Kanten sind in derselben ZHK

"höchstens zwei": wenn Start ≠ Endknoten, dann laufen wir in den

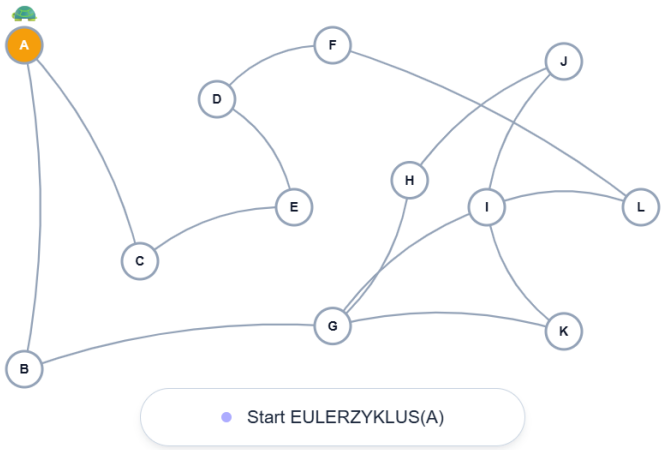
Startknoten einmal weniger hinein als hinaus und in den

Endknoten einmal mehr hinein als hinaus



Alle Kanten in derselben ZHK

Algorithmus von Hierholzer:



EULER TOUR $\mathcal{O}(m)$

```
1  EULERZYKLUS(v_0)
2    W ← FINDEZYKLUS(v_0)
3    laufe W ab; for jeden besuchten Knoten v:
4      if deg(v) > 0 then
5        Z ← FINDEZYKLUS(v)
6        füge Z an der aktuellen Stelle in W ein
7    return W

9  FINDEZYKLUS(v_0)
10   ℓ ← 0
11   repeat
12     if es gibt in E eine zu v_ℓ inzidente Kante e = {v_ℓ, u} then
13       v_{ℓ+1} ← u
14       ℓ ← ℓ + 1
15       lösche e aus E
16     else return W = (v_0, v_1, ..., v_ℓ)
```

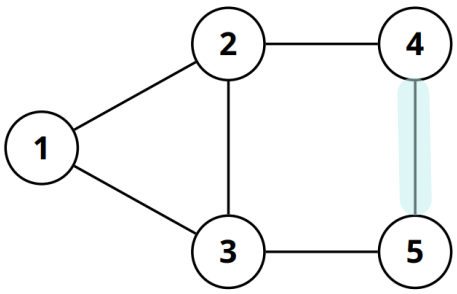
Laufzeit $O(m)$ mit Adjazenzliste

Algorithmus für Hamiltonkreis?

- NP vollständig
- bester bekannter Algorithmus hat exponentielle Laufzeit
- nächstes Semester

Adjazenzmatrix und Adjazenzliste

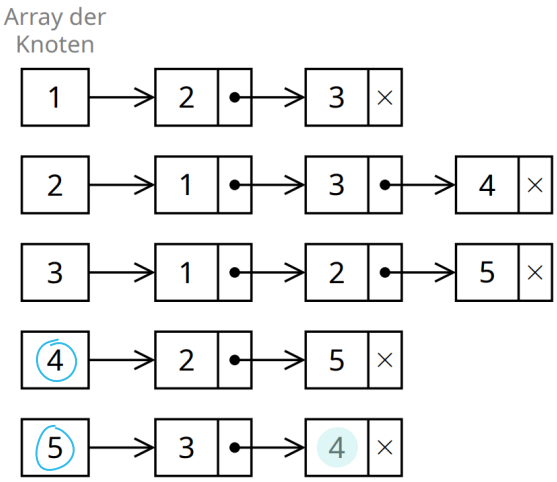
Graph ($n = 5, m = 6$)



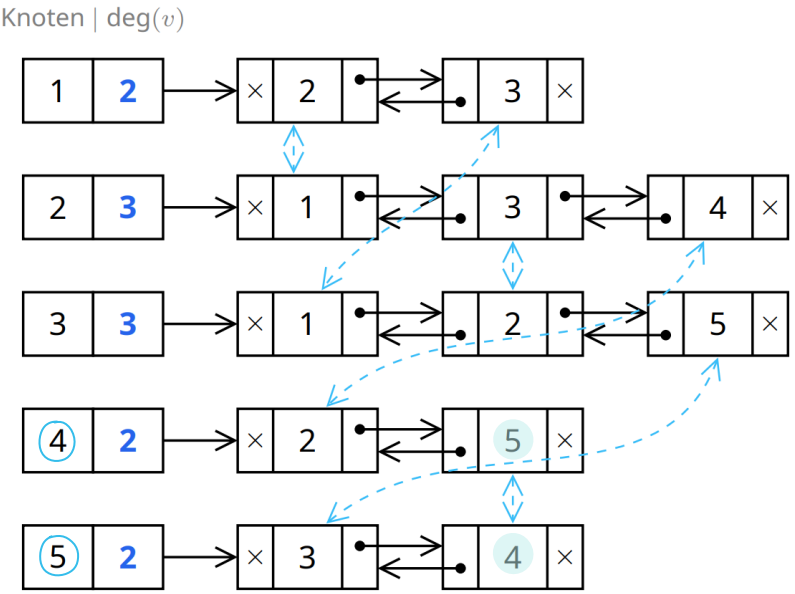
Adjazenzmatrix

$u \setminus v$	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	1	0
3	1	1	0	0	1
4	0	1	0	0	1
5	0	0	1	1	0

Adjazenzlisten



Erweiterte Adjazenzlisten



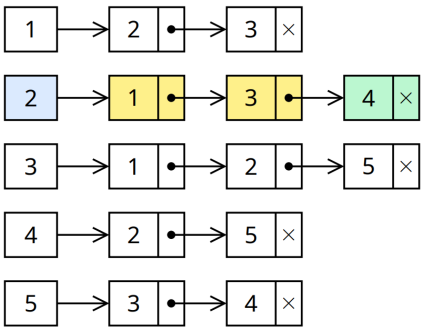
Operation	Adj.-Matrix	Adj.-Listen	erweitert
Ist $\{u, v\} \in E$?	$O(1)$	$O(\deg(u))$	
Finde einen Nachbarn von v	$O(n)$	$O(1)$	
Finde alle Nachbarn von v	$O(n)$	$O(\deg(v))$	
Finde $\deg(v)$	$O(n)$	$O(\deg(v))$	$O(1)$
Lösche Kante $e = \{u, v\}$	$O(1)$	$O(\deg(u) + \deg(v))$	$O(1)$
Füge Kante $e = \{u, v\}$ ein	$O(1)$	$O(\deg(u))$	
Lösche bzw. füge Knoten v ein[4]	$O(n^2)$	$O(n + m)$	$O(n)$

Ist $\{u, v\} \in E$**?** (Example: $\{2, 4\} \in E$?)

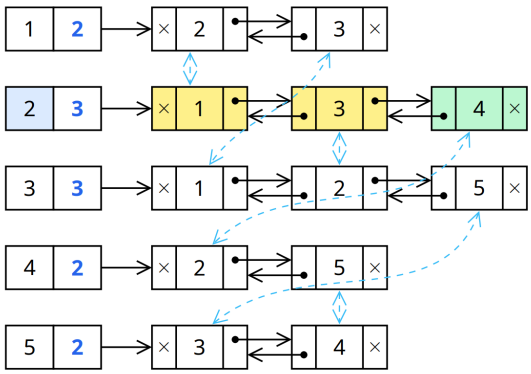
Adjazenzmatrix: $O(1)$

$u \setminus v$	1	2	3	**4**	5
1	0	1	1	0	0
2	1	0	1	**1**	0
3	1	1	0	0	1
4	0	1	0	0	1
5	0	0	1	1	0

Adjazenzlisten: $O(\deg(u))$



Erweiterte Adjazenzlisten: $O(\deg(u))$

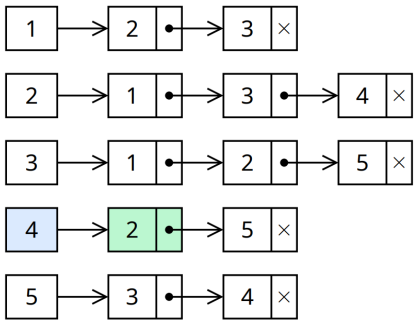


Finde einen Nachbarn von v (Example: $v = 4$)

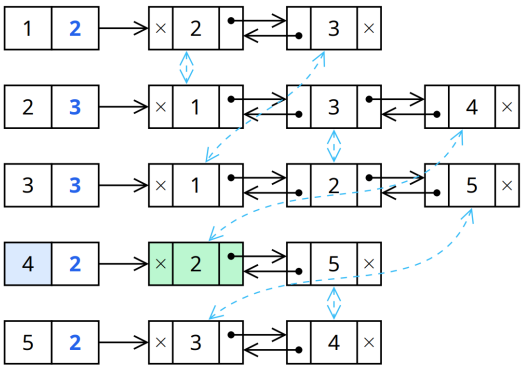
Adjazenzmatrix: $O(n)$

$u \setminus v$	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	1	0
3	1	1	0	0	1
4	0	**1**	0	0	1
5	0	0	1	1	0

Adjazenzlisten: $O(1)$



Erweiterte Adjazenzlisten: $O(1)$

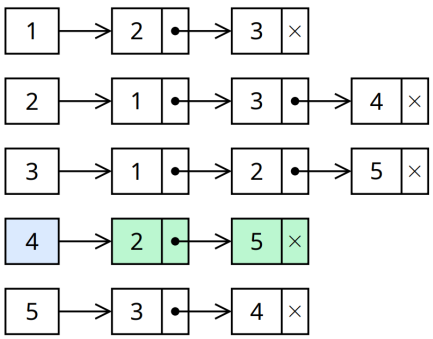


Finde alle Nachbarn von v (Example: $v = 4$)

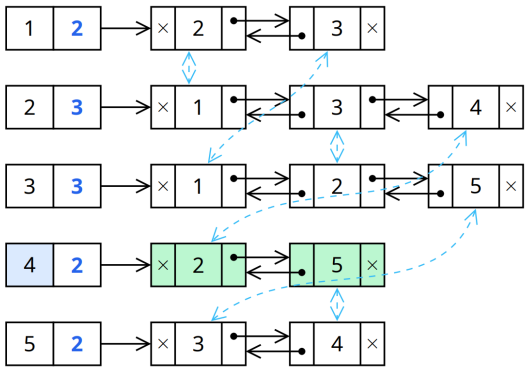
Adjazenzmatrix: $O(n)$

$u \setminus v$	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	1	0
3	1	1	0	0	1
4	0	**1**	0	0	**1**
5	0	0	1	1	0

Adjazenzlisten: $O(\deg(v))$



Erweiterte Adjazenzlisten: $O(\deg(v))$

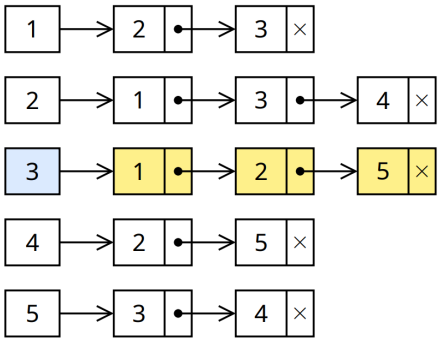


Finde $\deg(v)$ (Example: $v = 3$)

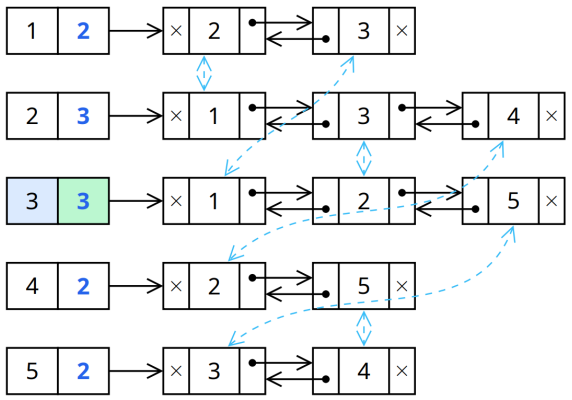
Adjazenzmatrix: $O(n)$

$u \setminus v$	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	1	0
3	**1**	**1**	0	0	**1**
4	0	1	0	0	1
5	0	0	1	1	0

Adjazenzlisten: $O(\deg(v))$



Erweiterte Adjazenzlisten: $O(1)$

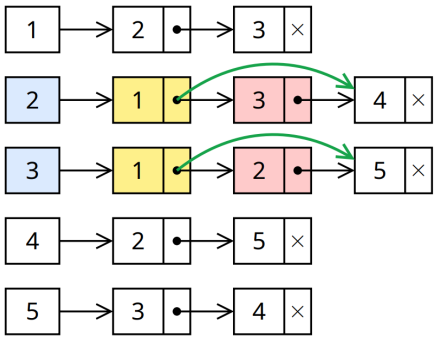


Lösche Kante $e = \{u, v\}$ (Example: Delete $e = \{2, 3\}$)

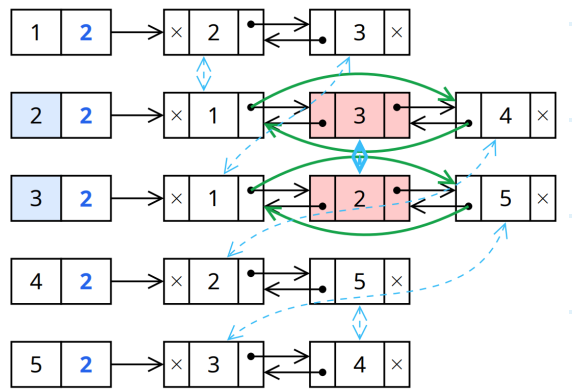
Adjazenzmatrix: $O(1)$

$u \setminus v$	1	**2**	**3**	4	5
1	0	1	1	0	0
2	1	0	**1→0**	1	0
3	1	**1→0**	0	0	1
4	0	1	0	0	1
5	0	0	1	1	0

Adjazenzlisten: $O(\deg(u) + \deg(v))$



Erweiterte Adjazenzlisten: $O(1)$



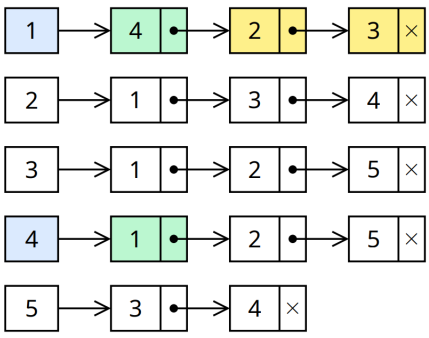
(Annahme: Pointer zu e ist gegeben)

Füge Kante $e = \{u, v\}$ **ein** (Example: Insert $e = \{1, 4\}$)

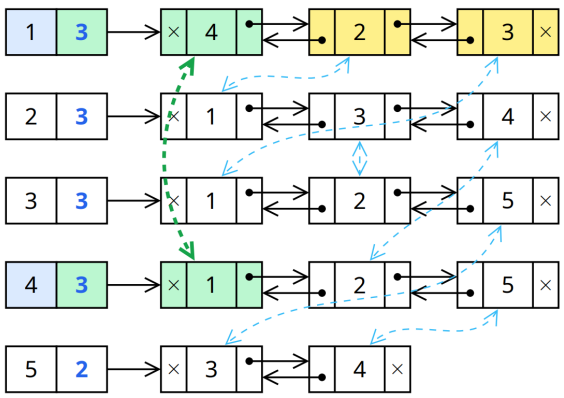
Adjazenzmatrix: $O(1)$

$u \setminus v$	**1**	2	3	**4**	5
1	0	1	1	**0→1**	0
2	1	0	1	1	0
3	1	1	0	0	1
4	**0→1**	1	0	0	1
5	0	0	1	1	0

Adjazenzlisten: $O(\deg(u))$



Erweiterte Adjazenzlisten: $O(\deg(u))$

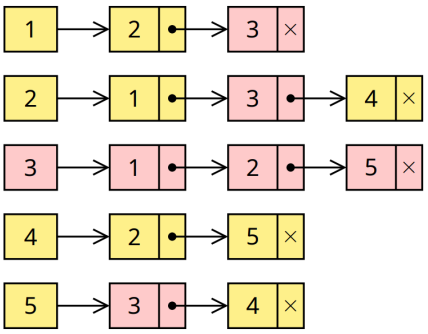


Lösche Knoten v (Example: Delete $v = 3$ and incident edges)

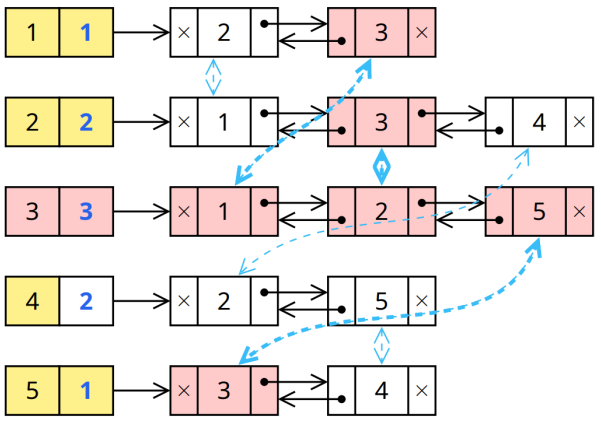
Adjazenzmatrix: $O(n^2)$

$u \setminus v$	1	2	**3**	4	5
1	0	1	1	0	0
2	1	0	1	1	0
3	1	1	0	0	1
4	0	1	0	0	1
5	0	0	1	1	0

Adjazenzlisten: $O(n+m)$



Erweiterte Adjazenzlisten: $O(n)$

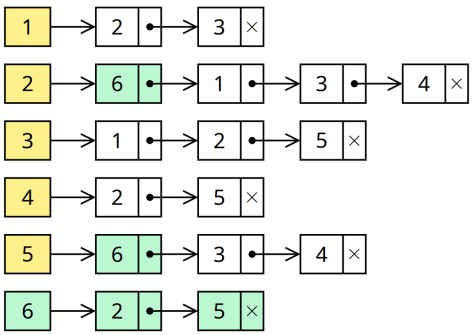


Füge Knoten v **ein** (Example: Insert $v = 6$ with edges $\{2,6\}, \{5,6\}$)

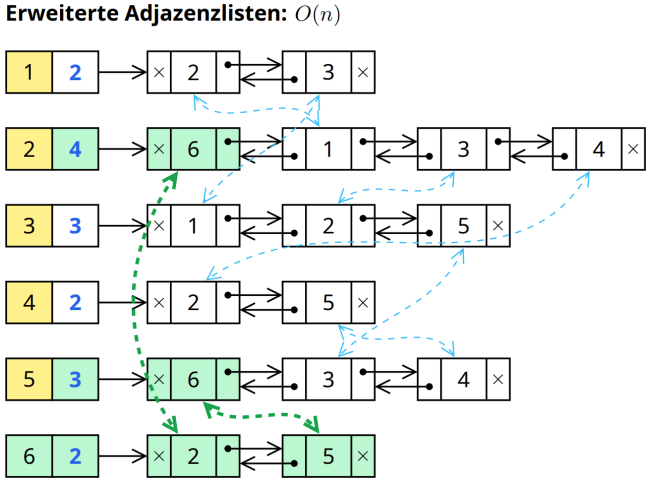
Adjazenzmatrix: $O(n^2)$

$u \setminus v$	1	2	3	4	5	**6**
1	0	1	1	0	0	0
2	1	0	1	1	0	**1**
3	1	1	0	0	1	0
4	0	1	0	0	1	0
5	0	0	1	1	0	**1**
6	0	**1**	0	0	**1**	0

Adjazenzlisten: $O(n)$



Erweiterte Adjazenzlisten: $O(n)$



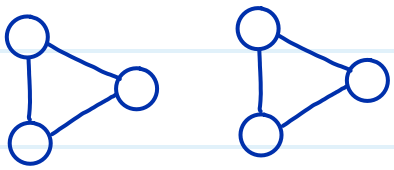
Tipps für Graph-Beweisaufgaben:

- Immer genaue mathematische Definitionen der Begriffe verwenden
- Beweisstrategien aus DiskMath sind besonders hilfreich:
  - Kontraposition: $A \to B \equiv \neg B \to \neg A$
  - Widerspruchsbeweis: $\neg A \to \bot$
  - Fallunterscheidung
  - Gegenbeispiel
  - Pigeon hole Prinzip

Aufgaben:

Prove or disprove: Jeder Graph, in dem alle Knotengrade gerade sind, ist zusammenhängend.

Counterexample:



other counterexample. ○ ○

Prove or disprove: Sei G ein Graph und $a, b, c \in V$ drei verschiedene Knoten von G. Wenn es ein Weg von a nach b, und ein Weg von b nach c gibt, dann gibt es ein Pfad von a nach c.

$A \to B$

Beweis: Sei $(a, v_1, v_2, \dots, v_k, b)$ ein Weg von a nach b

Sei $(b, w_1, w_2, \dots, w_e, c)$ ein Weg von b nach c

Dann ist $(a, v_1, v_2, \dots, v_k, b, w_1, w_2, \dots, w_e, c)$ ein Weg von a nach c

Solange ein Knoten d doppelt vorkommt, kürzen wir ab

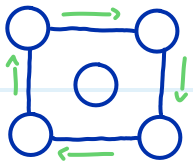
D.h falls der Weg die Form $(\overbrace{x_1, \dots, x}^{\text{Anfang}}, d, \overbrace{y_1, \dots, y_j}^{\text{Mitte}}, d, \overbrace{z_1, \dots, z_k}^{\text{Ende}})$ hat,

kürzen wir zu $(x_1, \dots, x, d, z_1, \dots, z_k)$ ab.

So erhalten wir einen Pfad von a nach c

Prove or disprove: Wenn ein Graph G einen Eulerzyklus Z hat, sodass Z ein Kreis ist, dann muss G einen Hamiltonkreis haben.

Counterexample:



Prove or disprove: Sei G ein Graph mit $n \geq 3$ Knoten. Wenn G zusammenhängend ist und $m = n - 1$ viele Kanten hat, dann muss es einen Knoten mit Grad 1 geben.

Widerspruchsbeweis: $(\neg A \to \bot)$

Annahme: G hat keinen Knoten mit Grad 1 $(\neg A)$

$\Rightarrow \forall v \in V: \deg(v) \geq 2$ (kann nicht 1 sein per Annahme und nicht 0 sein weil G zusammenhängend ist)

$\Rightarrow \sum_{v \in V} \deg(v) \geq \sum_{v \in V} 2$

$\Rightarrow \sum_{v \in V} \deg(v) \geq 2n$ $(|V| = n)$

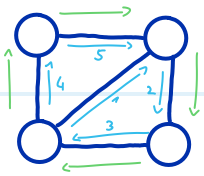
$\Rightarrow 2m \geq 2n$ (Handschlag-Lemma)

$\Rightarrow m \geq n$

$\Rightarrow$ Widerspruch zu $m = n-1$ $(\bot)$

Prove or disprove: Jeder Graph, der einen Hamiltonkreis und einen Eulerweg hat, muss auch einen Eulerzyklus haben.

Counterexample:



hat keinen Eulerzyklus weil es Knoten mit ungeraden Knotengraden gibt

Prove or disprove: Wenn ein Graph G einen Hamiltonpfad besitzt, dann ist G zusammenhängend.

Beweis. Sei $(v_1, v_2, \dots, v_n)$ ein Hamiltonpfad in G

Zu zeigen $\forall x,y \in V$: x erreicht y

Seien $x, y \in V$ beliebig

Fall $x = y$:

$\Rightarrow$ x erreicht y (leerer Weg)

Fall $x \neq y$:

Sei $v_i = x$ und $v_j = y$

o.B.d.A. sei $i < j$ (o.B.d.A. heisst, dass wir eine zusätliche Annahme machen, die die allgemeine Gültigkeit nicht verändert, weil der andere Fall (hier $j < i$) symmetrisch ist)

$\Rightarrow (v_i, v_{i+1}, \dots, v_j)$ ist ein Weg von $v_i = x$ nach $v_j = y$

$\Rightarrow$ x erreicht y

Weil $\forall x,y \in V$: x erreicht y, folgt dass G zusammenhängend ist (per Definition)

Prove or disprove: Sei G ein zusammenhängender Graph, sodass es für jeden Knoten v einen Eulerweg E_v gibt, der bei v startet. Dann hat G einen Eulerzyklus.

Counterexample:



Prove or disprove: Sei G ein zusammenhängender Graph mit $n \geq 4$ Knoten, sodass es für jeden Knoten v einen Eulerweg E_v gibt, der bei v startet. Dann hat G einen Eulerzyklus.

Widerspruchsbeweis:

Annahme: G hat keinen Eulerzyklus ($\neg A$)

$\Rightarrow$ Es gibt einen Knoten $x \in V$ mit ungeradem Grad (Satz von Euler)

Sei $E_v = (x, v_1, \ldots, v_i, y)$

$\Rightarrow$ deg(x) und deg(y) sind ungerade (in y gehen wir einmal mehr hinein als hinaus und in x gehen wir einmal weniger hinein als hinaus)

$\Rightarrow$ $\forall v \in V \setminus \{x,y\}$: deg(v) ist gerade (wir gehen gleich oft hinein wie hinaus)

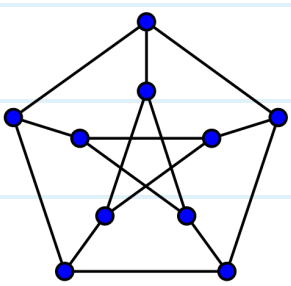
$\Rightarrow$ E_{v_1} muss ein geschlossener Weg sein, weil deg(v_1) gerade ist.

$\Rightarrow$ E_{v_1} ist ein Eulerzyklus (geschlossener Eulerweg = Eulerzyklus)

$\Rightarrow$ Widerspruch ($\bot$)

Prove or disprove: Sei G ein Graph mit $n \geq 4$ Knoten, sodass es für jeden Knoten v einen Hamiltonpfad H_v gibt, der bei v startet. Dann hat G einen Hamiltonkreis.

Counterexample:



Prove or disprove: Jeder Graph G mit $n \geq 3$ Knoten, in dem jeder Knoten v einen Grad von mindestens $\deg(v) \geq n/2$ hat, muss zusammenhängend sein.

Beweis:

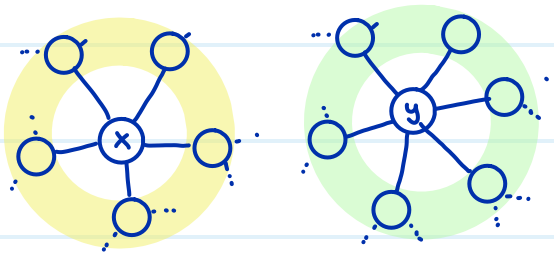
Zu zeigen $\forall x,y \in V$: x erreicht y

Seien $x, y \in V$ beliebig

Fall $x = y$: $\Rightarrow$ x erreicht y (leerer Weg, trivial)

Fall $x \neq y \wedge \{x,y\} \in E$ $\Rightarrow$ x erreicht y (Weg (x,y), trivial)

Fall $x \neq y \wedge \{x,y\} \notin E$:



Sei X die Menge der Nachbaren von x.

Sei Y die Menge der Nachbaren von y

$\Rightarrow |X| \geq \frac{n}{2}$ und $|Y| \geq \frac{n}{2}$ weil $\deg(x) \geq \frac{n}{2}$ und $\deg(y) \geq \frac{n}{2}$

$\Rightarrow x,y \notin X$ und $x,y \notin Y$ (weil x und y nicht benachbart sind)

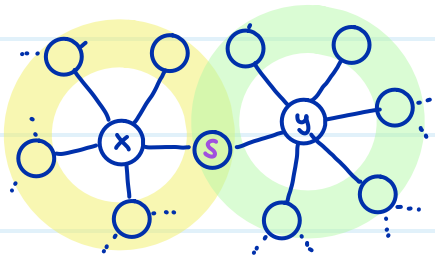
Wir zeigen $X \cap Y \neq \emptyset$ mit einem Widerspruchsbeweis:

Annahme: $X \cap Y = \emptyset$

$\Rightarrow |V| \geq |X| + |Y| + 1 + 1$

$\Rightarrow n \geq \frac{n}{2} + \frac{n}{2} + 2$

$\Rightarrow n \geq n+2$ ist ein Widerspruch



$\Rightarrow$ Weil also $X \cap Y \neq \emptyset$ ist, gibt es einen Knoten $s \in X \cap Y$

$\Rightarrow (x,s,y)$ ist ein Pfad von x nach y

$\Rightarrow$ x erreicht y

Weil $\forall x,y \in V$: x erreicht y, folgt dass G zusammenhängend ist (per Definition)