A&W 2026

G-13 Timo Stucki, LFW E 13

Week 1

Mini Quiz

Articulation Points & Bridges (with Exercise)

Cycles

Exercise Preview

Infos

ETH zürich

Feel free to contact me:

- tistucki@student.ethz.ch
- Discord: timostucki

Material:

- timostucki.com

About this exercise group

- Listed as focus group
  - Focus on helping you pass the exam (and get a good grade!)
  - Teaching will be centred around preparing you for the exam → less time for in-depth understanding of theory and hard proofs, more time for exercise walk-throughs and what's most important for the exam
  - I'm there for you if you are struggling
    - Never hesitate to ask questions!
    - Contact me
- What I will not do
  - Skip material
  - Go at a slower pace in the sense that we fall behind with material

(In my opinion, a focus group should neither give you false comfort nor put you at a disadvantage)

Mini Quiz Recap

Mini Quiz Recap

Definition: Sei $G = (V, E)$ ein Graph.

G heisst k-zusammenhängend, wenn gilt:

- $|V| \geq k + 1$ und
- $\forall X \subseteq V$ mit $|X| < k$ ist $G[V \setminus X]$ zusammenhängend

Satz von Menger:

Sei $G = (V, E)$ ein Graph. Dann gilt:

G ist **k-zusammenhängend** genau dann wenn (gdw.)

$\forall u, v \in V, u \neq v$ gibt es **k intern-knotendisjunkte** u-v-Pfade

Mini Quiz Recap

Definition: Sei $G = (V, E)$ ein Graph.

G heisst k-kanten-zusammenhängend, wenn gilt:

- $\forall X \subseteq E$ mit $|X| < k$ ist $(V, E \setminus X)$ zusammenhängend

Satz von Menger (Kanten-Version):

Sei $G = (V, E)$ ein Graph. Dann gilt:

G ist **k-kanten-zusammenhängend** genau dann wenn (gdw.)

$\forall u, v \in V, u \neq v$ gibt es **k kantendisjunkte** u-v-Pfade

!!! IMPORTANT !!!

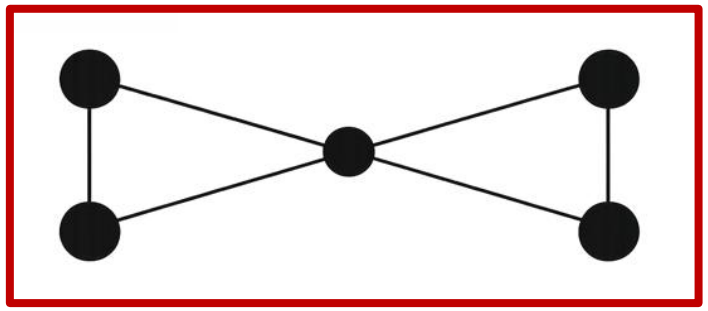
We can always remove a given edge by removing one of the vertices it is incident to

vertex-connectivity ≤ edge-connectivity ≤ minimal degree

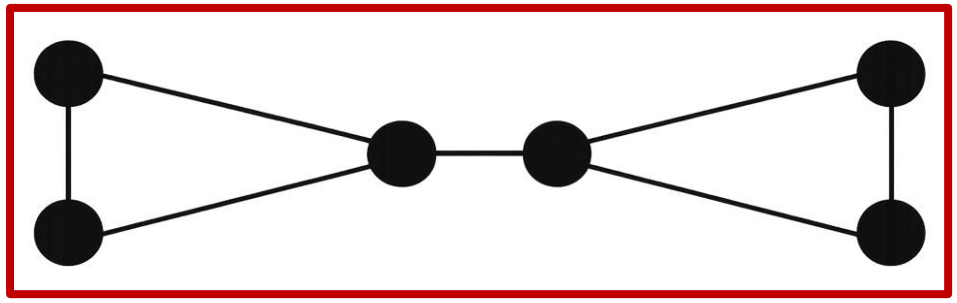
We can always isolate a vertex of minimal degree by removing all its incident edges

!!! IMPORTANT !!!

vertex-connectivity ≤ edge-connectivity ≤ minimal degree

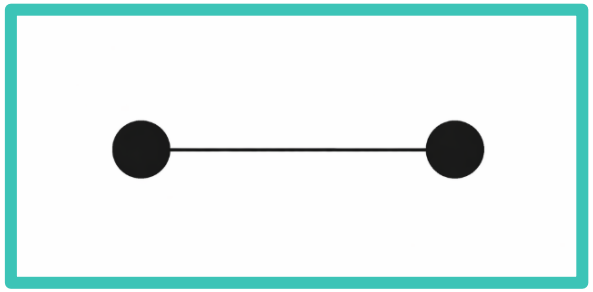


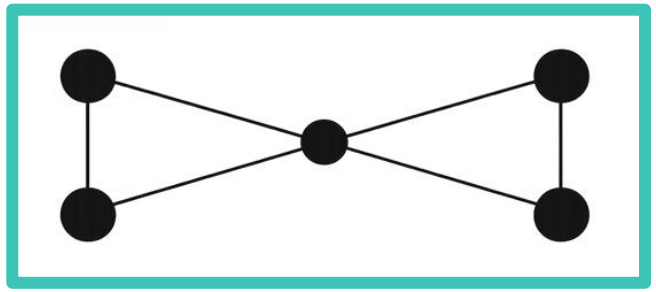
Edge-connectivity does not give guarantees on vertex-connectivity

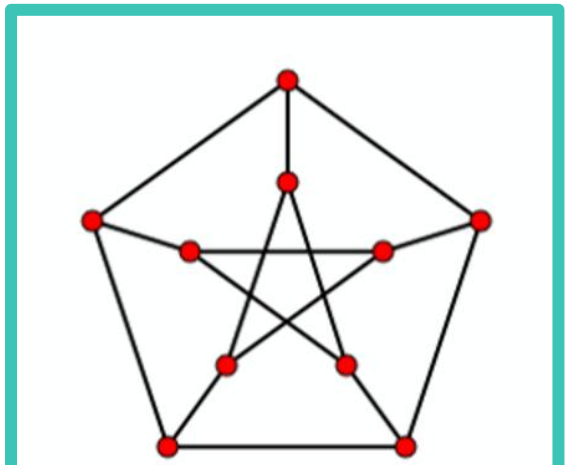


Minimal degree does not give guarantees on edge-connectivity

Graphs for easy counterexamples:





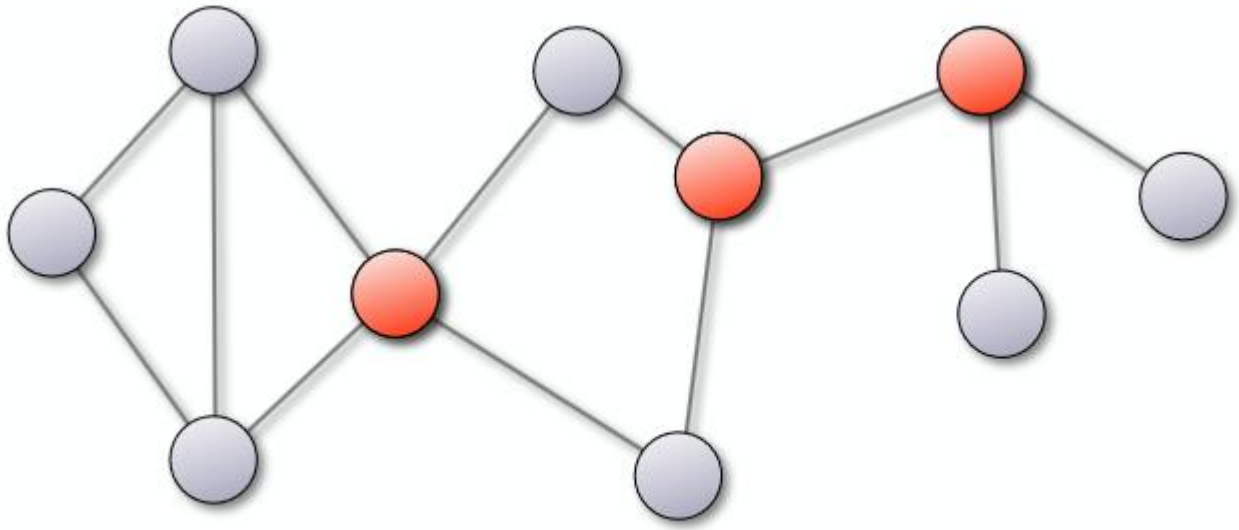


Petersengraph

Articulation Points & Bridges

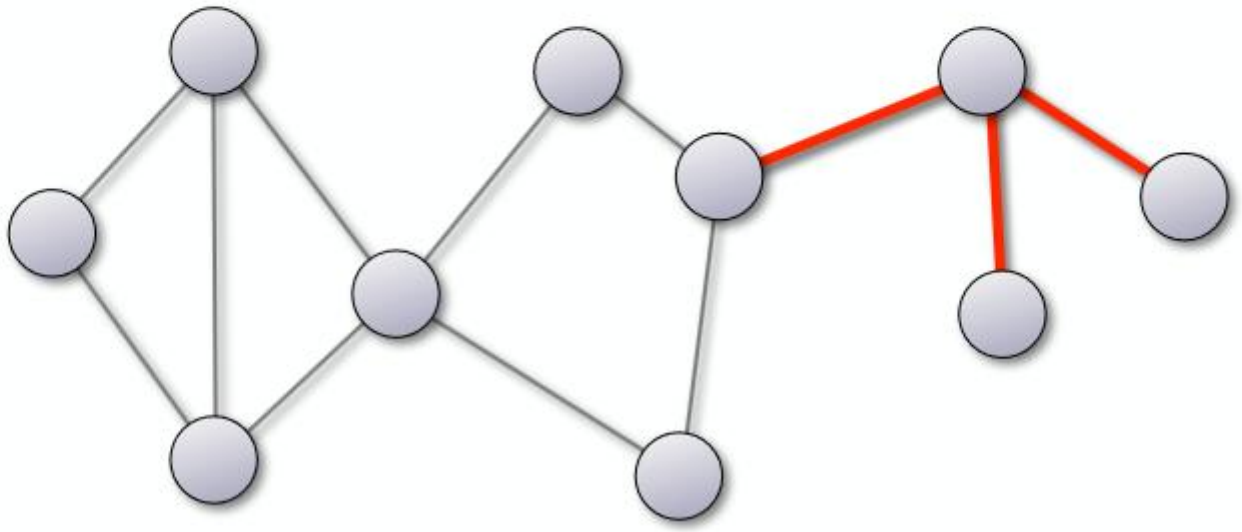
Articulation Points & Bridges

Definition: Sei $G = (V, E)$ ein zusammenhängender Graph.
Ein Knoten $v \in V$ heisst *Artikulationsknoten* (engl. cut vertex)
gdw. $G[V \setminus \{v\}]$ nicht zusammenhängend ist



Articulation Points & Bridges

Definition: Sei $G = (V, E)$ ein zusammenhängender Graph.
Ein Kante $e \in E$ heisst *Brücke* (engl. cut edge)
gdw. $G - e$ nicht zusammenhängend ist



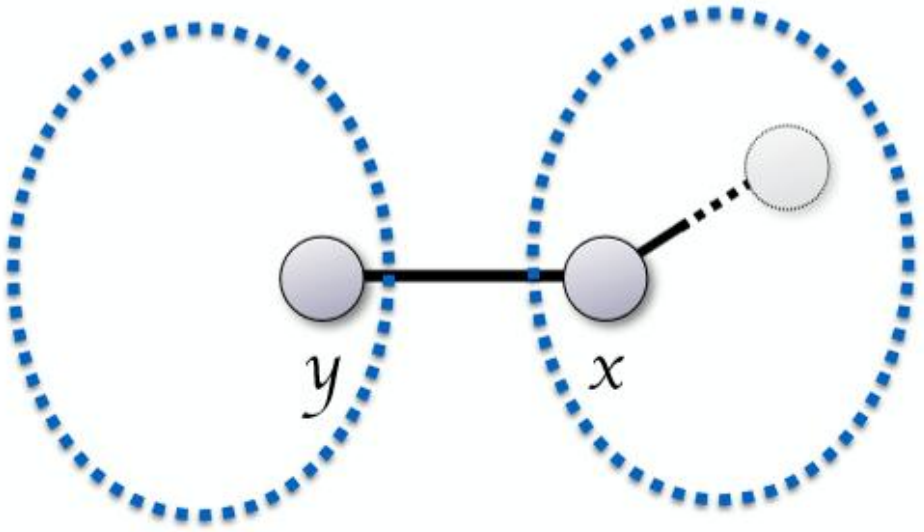
Articulation Points & Bridges !!! IMPORTANT !!!

Lemma: Sei $G = (V, E)$ ein zusammenhängender Graph.
Ist $\{x, y\} \in E$ eine Brücke so gilt:

$\deg(x) = 1$ oder x ist Artikulationsknoten

(und analog für y).

Beweisidee:



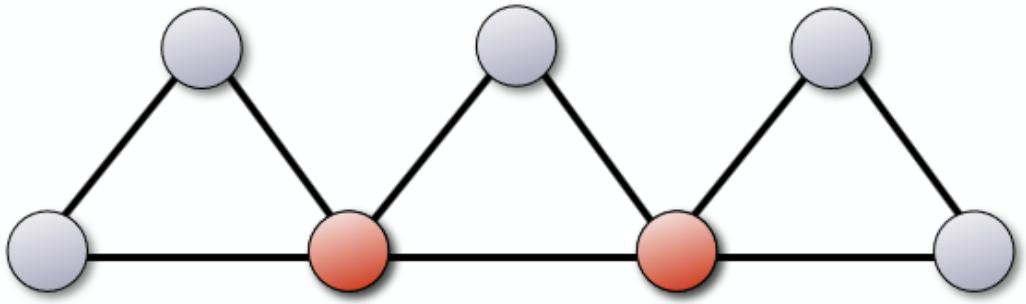
Articulation Points & Bridges !!! IMPORTANT !!!

Lemma: Sei $G = (V, E)$ ein zusammenhängender Graph.
Ist $\{x, y\} \in E$ eine Brücke so gilt:

$\deg(x) = 1$ oder x ist Artikulationsknoten

(und analog für y).

Aber: die Umkehrung gilt i.A. nicht …!!



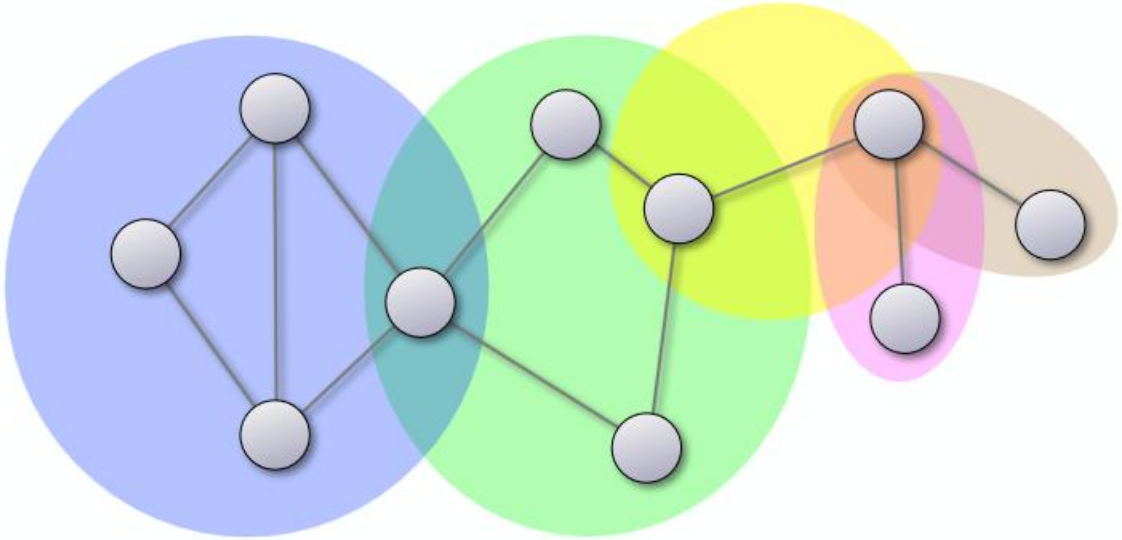
Artikulationsknoten

Articulation Points & Bridges

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

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

Die Äquivalenzklassen nennen wir Blöcke.



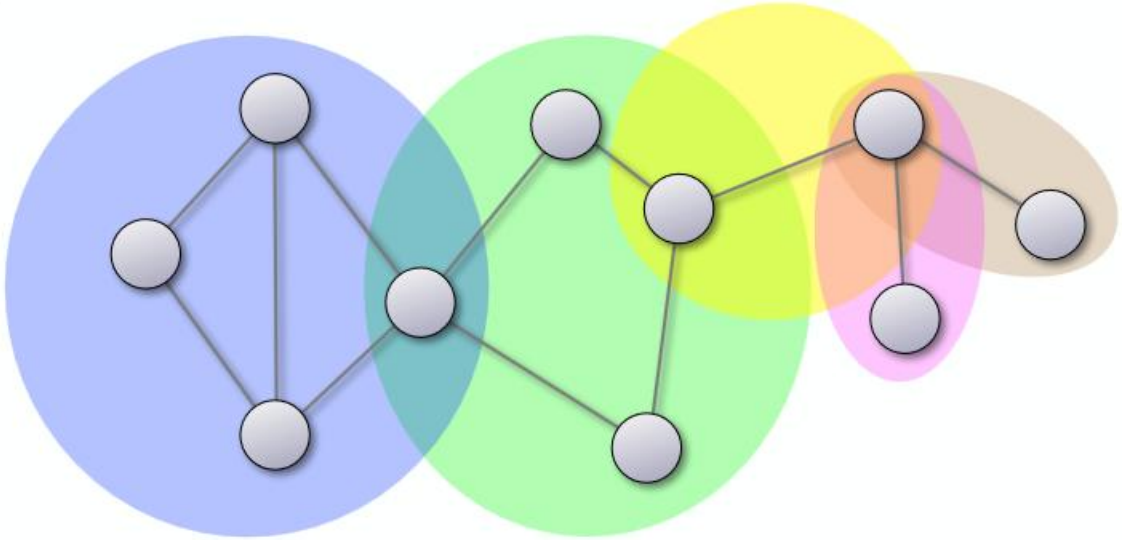
Transitiv? $e \sim f$ und $f \sim g$ dann $e \sim g$

Articulation Points & Bridges

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

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

Die Äquivalenzklassen nennen wir Blöcke.



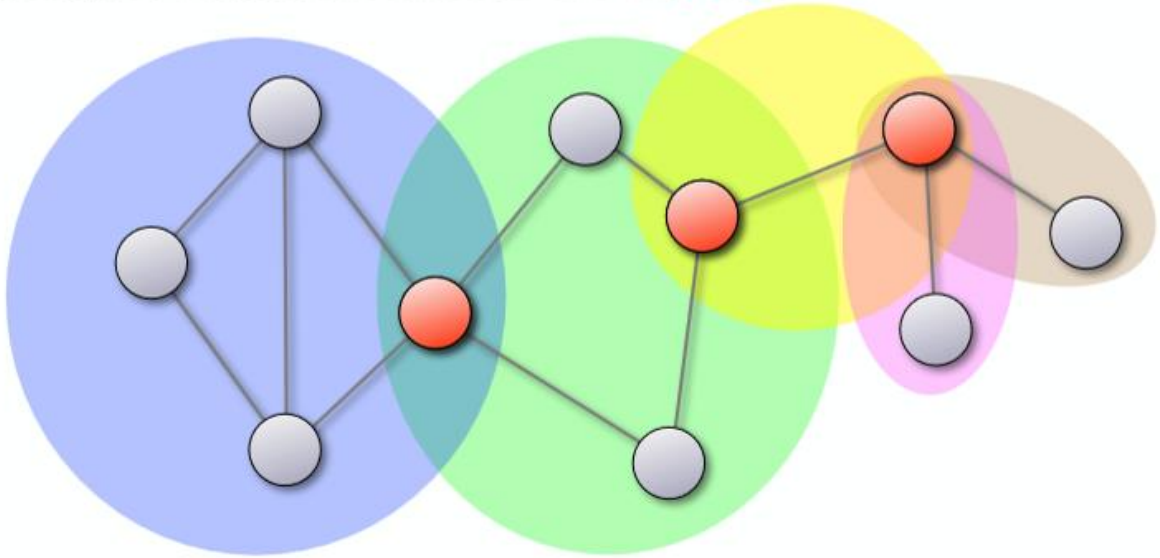
Transitiv? $e \sim f$ und $f \sim g$ dann $e \sim g$

Articulation Points & Bridges

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

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

Die Äquivalenzklassen nennen wir Blöcke.



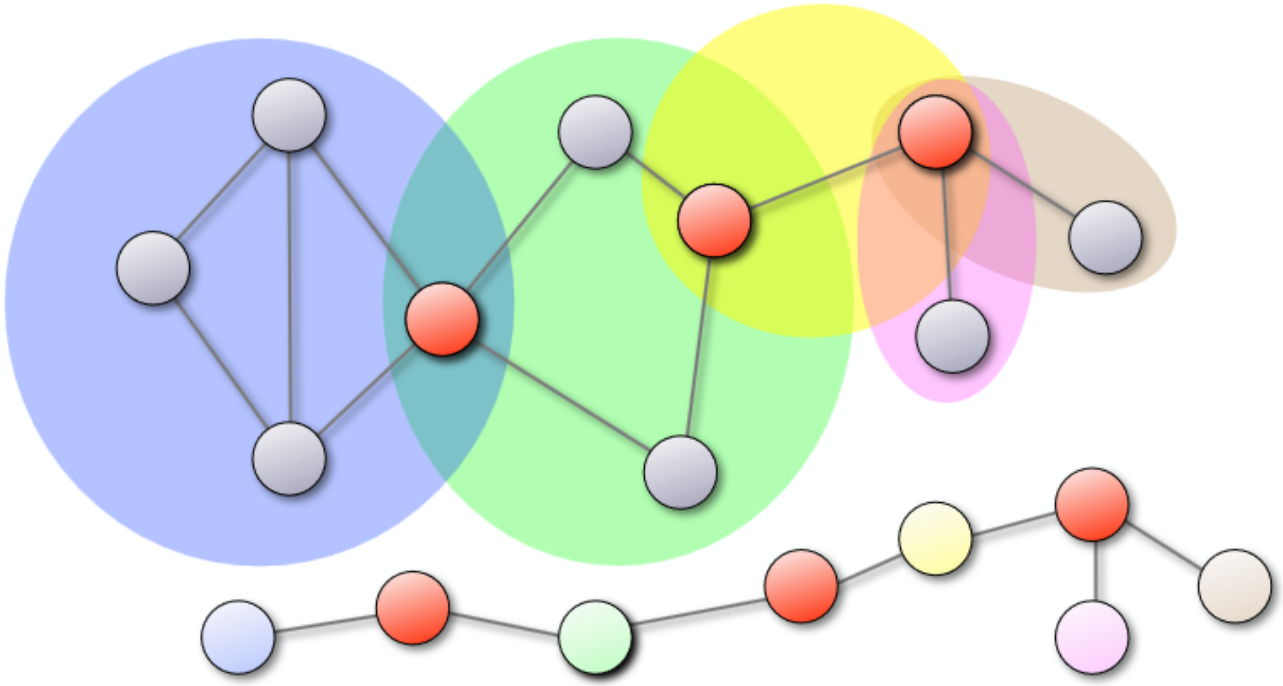
Lemma: Zwei Blöcke schneiden sich — wenn überhaupt — immer in einem Artikulationsknoten.

Articulation Points & Bridges

Definition: Sei $G = (V, E)$ ein zusammenhängender Graph.

Der Block-Graph von G ist der bipartite Graph $T = (A \uplus B, E_T)$ mit

- $\mathrm{A} = \{\text{Artikulationsknoten von } G\}$.
- $\mathrm{B} = \{\text{Blöcke von } G\}$.
- $\forall a \in A, b \in B: \quad \{a, b\} \in E_T \iff a \text{ inzident zu einer Kante in } b$.



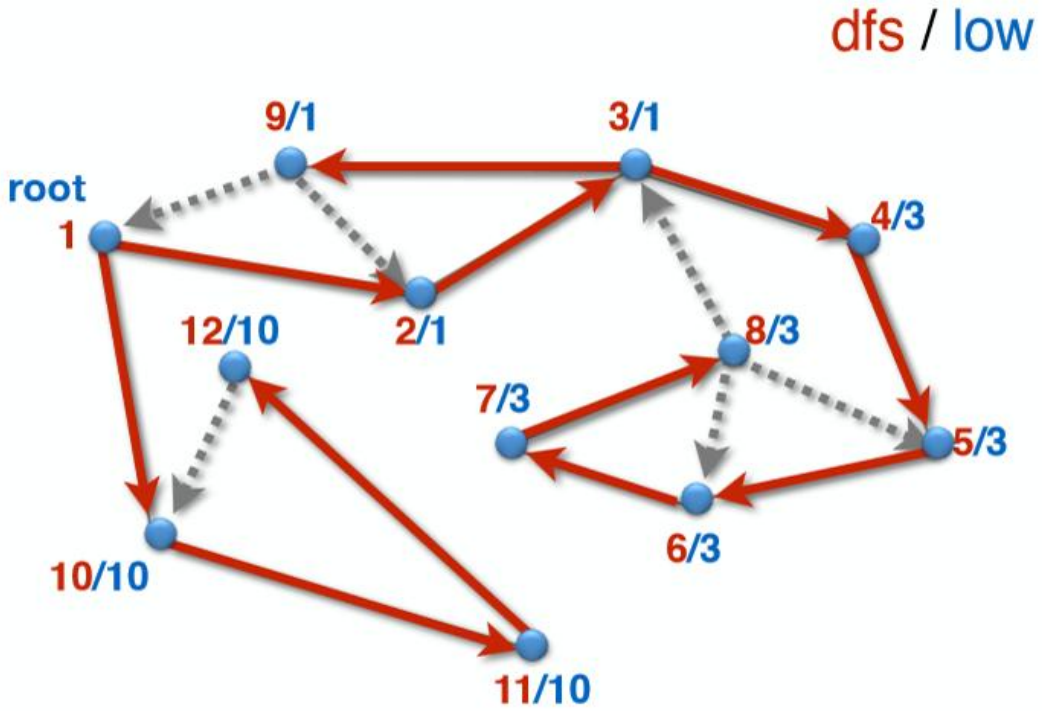
Satz: Ist G zusammenhängend, so ist der Blockgraph von G ein Baum.

Articulation Points & Bridges

DFS-VISIT-ITERATIVE(G, v)

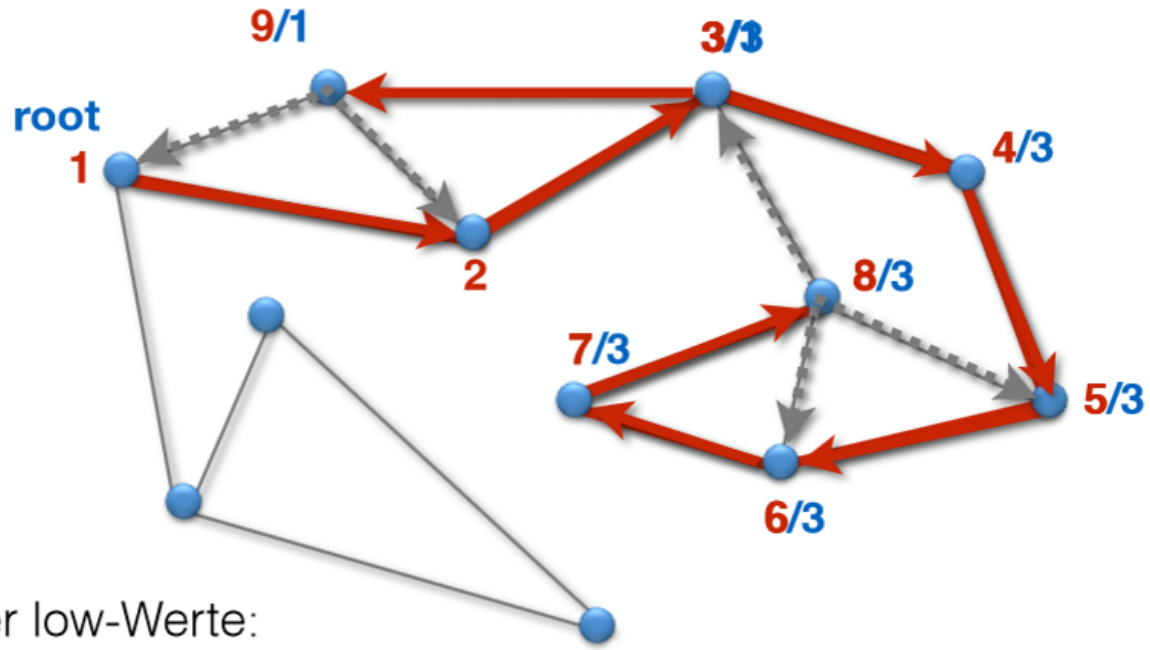
```
1  S ← ∅
2  PUSH(S, v)
3  while S ≠ ∅ do
4      w ← POP(S)
5      if w noch nicht besucht then
6          Markiere w als besucht
7          for each (w, x) ∈ E in reverse order do
8              if x noch nicht besucht then
9                  PUSH(S, x)
```

Articulation Points & Bridges



$\text{low}[v] :=$ kleinste dfs-Nummer, die man von v aus durch einen gerichteten Pfad aus (beliebig vielen) Baumkanten und maximal einer Restkante erreichen kann.

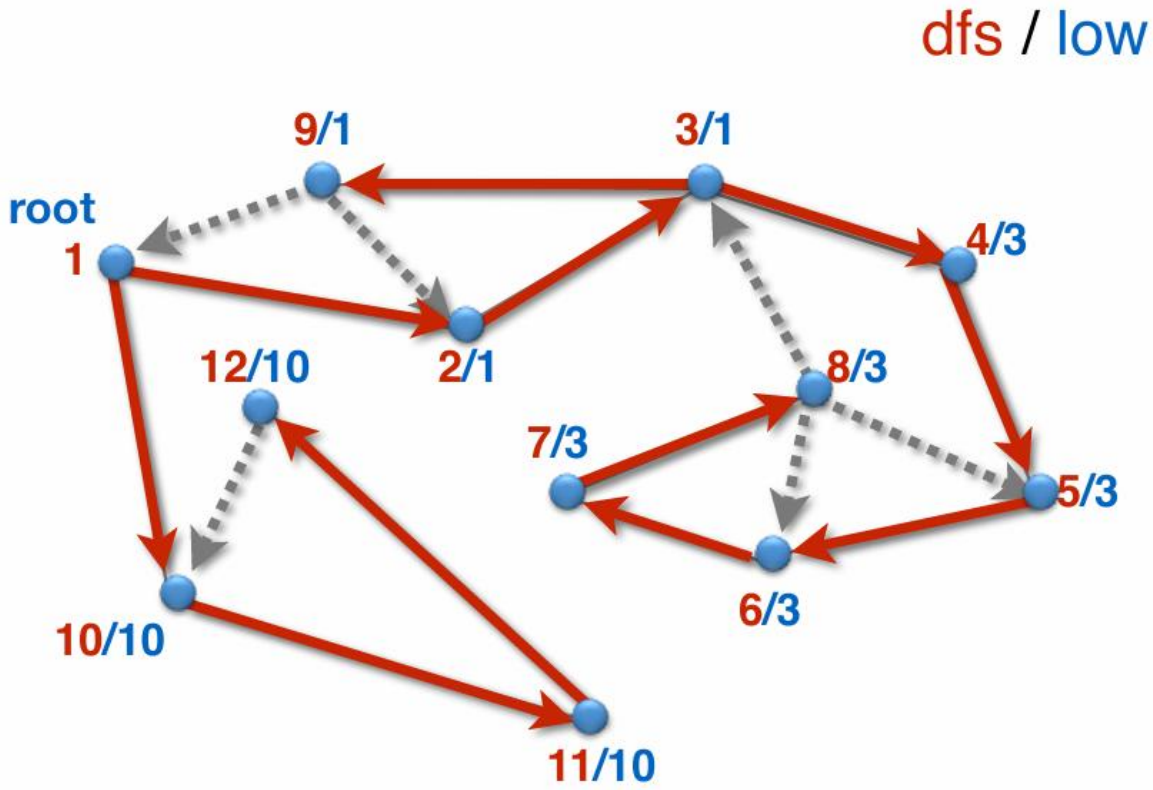
Articulation Points & Bridges



Berechnung der low-Werte:

- Initialisierung mit dfs-Wert
- ggf update, wenn Restkanten gefunden werden
- ggf update, wenn Algorithmus während des backtracks zum Knoten zurückkehrt

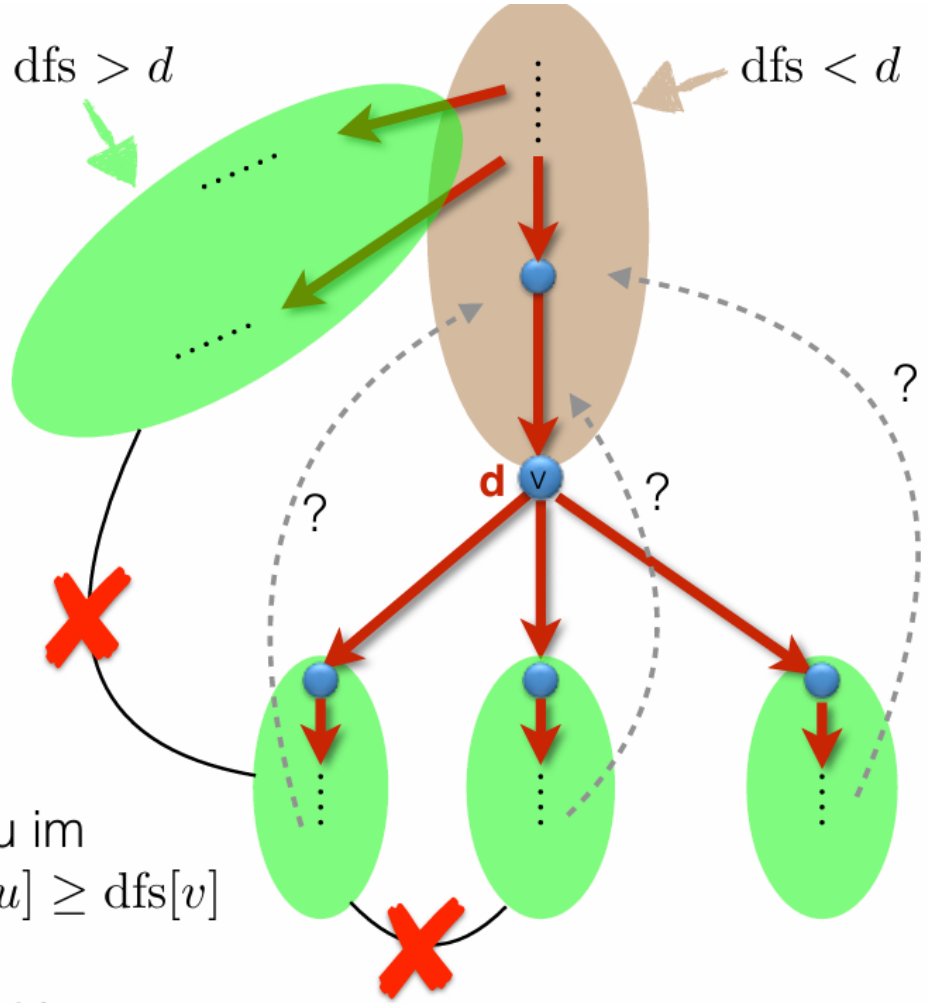
Articulation Points & Bridges



Alle low-Werte sind in Zeit O(|V|+|E|) berechenbar:

$$\text{low}[v] = \min\left(\text{dfs}[v],\quad \min_{(v,w)\in E}\begin{cases}\text{dfs}[w], & \text{falls } (v,w) \text{ Restkante}\\ \text{low}[w], & \text{falls } (v,w) \text{ Baumkante}\end{cases}\right)$$

Articulation Points & Bridges



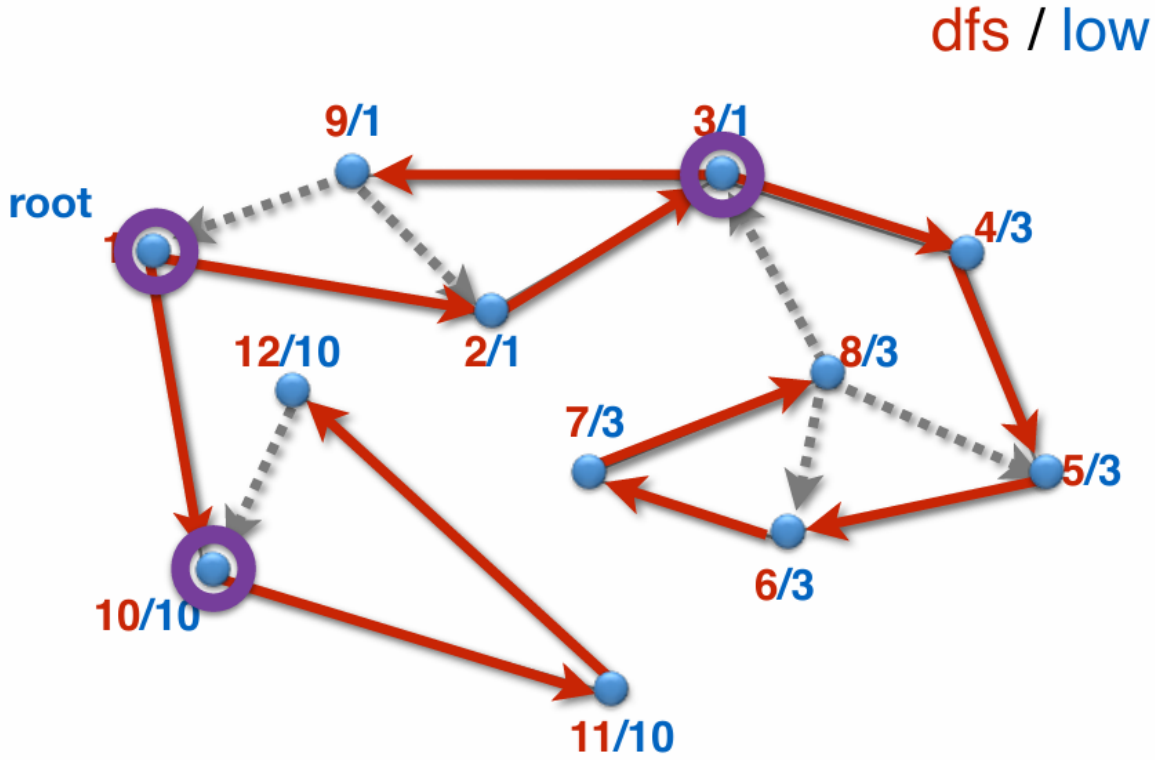
v ist genau dann Artikulations-knoten, wenn

1) v ≠ root, und v hat ein Kind u im DFS-Baum mit $\mathrm{low}[u] \geq \mathrm{dfs}[v]$

oder

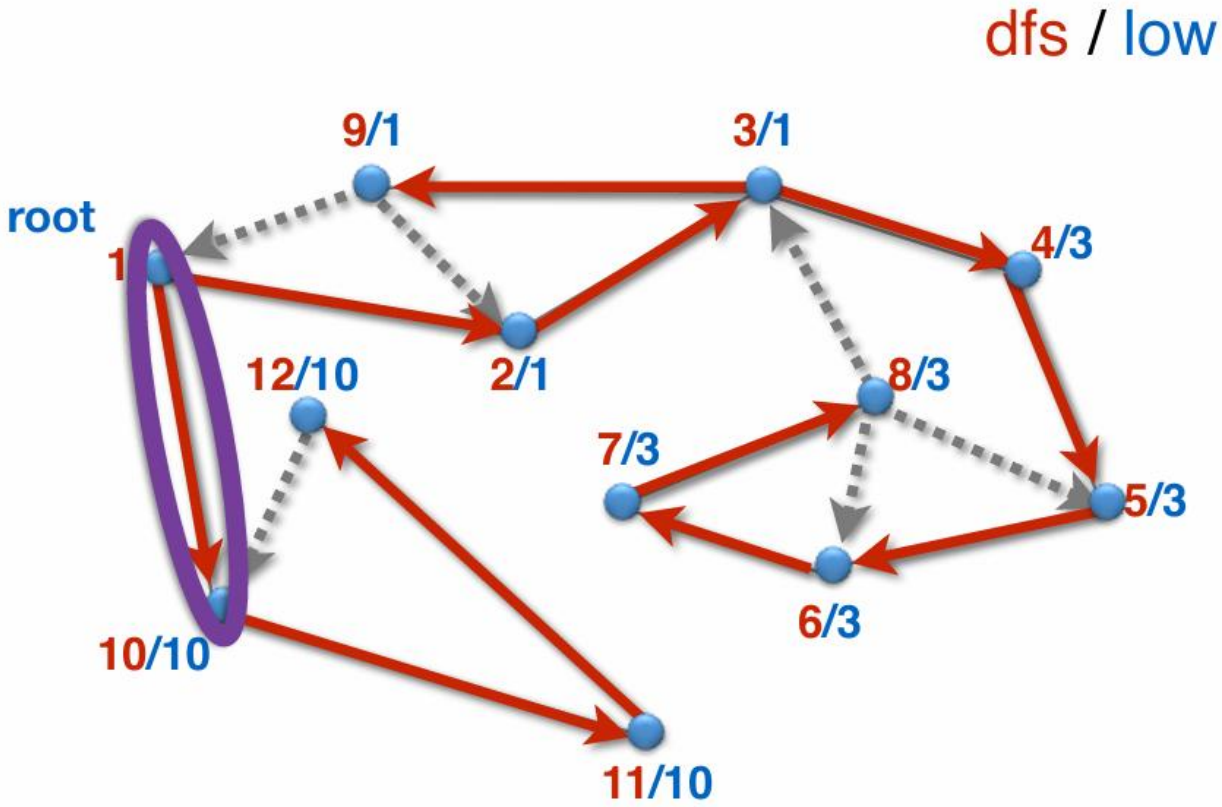
2) v = root, und v hat mindestens zwei Kinder im DFS-Baum.

Articulation Points & Bridges



Ein Knoten v ist genau dann ein Artikulationsknoten, wenn
1) v ≠ root, und v hat ein Kind u im DFS-Baum mit $\mathrm{low}[u] \geq \mathrm{dfs}[v]$,
oder
2) v = root, und v hat mindestens zwei Kinder im DFS-Baum.

Articulation Points & Bridges



Eine Baumkante e = (v,w) (v Elternknoten, w Kindknoten) ist genau dann eine Brücke, wenn $\mathrm{low}[w] > \mathrm{dfs}[v]$.

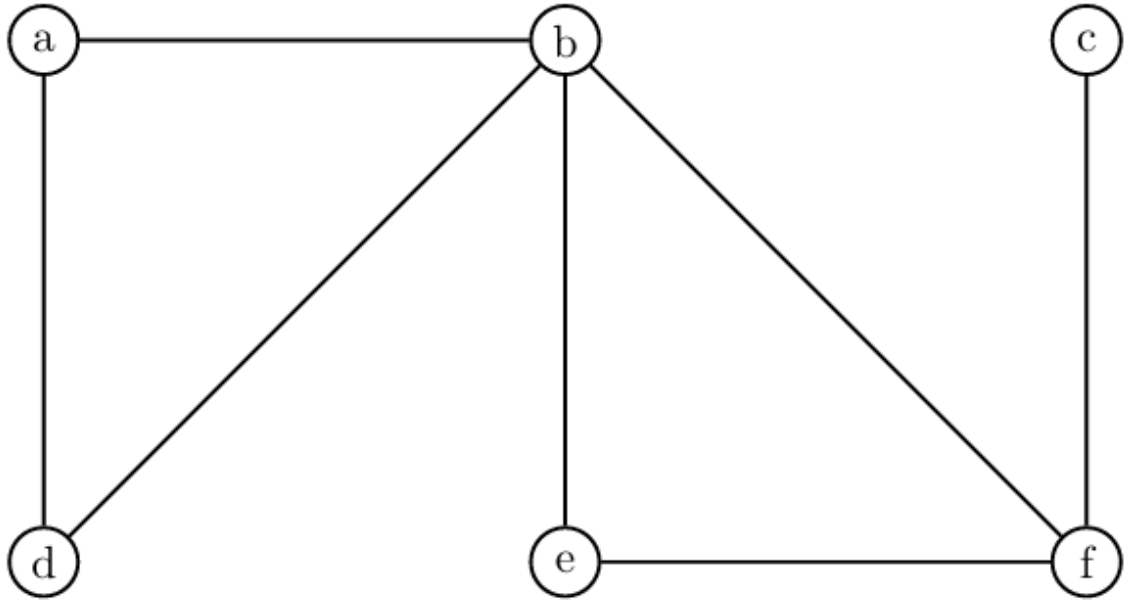
Restkanten sind niemals Brücken. **But why?**

Exercise S2.1

Exercise S2.1 – *Efficient Search for Brigdes and Cut-Vertices*

In the lecture we have seen how to find bridges (Brücken) and cut-vertices (Artikulationsknoten) in a graph. The algorithm uses a modified DFS and computes values `dfs[v]` and `low[v]` for ever vertex v.

Consider the following graph G.



(a) Perform the modified DFS starting at vertex a. Use the corresponding `dfs`- and `low`-values to determine the cut-vertices and bridges of G.

(b) Try the same analysis, but starting in b. Do the `dfs`- and `low`-values change? Do the cut-vertices and bridges change?

(c) Compute the block graph of G.

Cycles

A&D Recap

	"non-distinct" vertices	distinct vertices
-	**walk**	**path**
endpoints must be the same	**closed walk** (German: Zyklus)	**cycle** (German: Kreis)

$$O(1) \subset O(\log\log n) \subset O(\log n) \subset O(\sqrt{n}) \subset O(n) \subset O(n\log n) \subset O(n\sqrt{n}) \subset O(n^2) \subset O(2^n) \subset O(n!) \subset O(n^n)$$

Cycles

Hamiltonian Cycle

- A cycle that contains every vertex exactly once

Eulerian Cycle

- A closed walk that contains every edge exactly once

G has a Eulerian cycle $\iff$ G is connected and every vertex has an even degree

Cycles

Satz: Ein zusammenhängender Graph $G = (V, E)$ enthält eine Eulertour

gdw. der Grad jedes Knotens gerade ist.

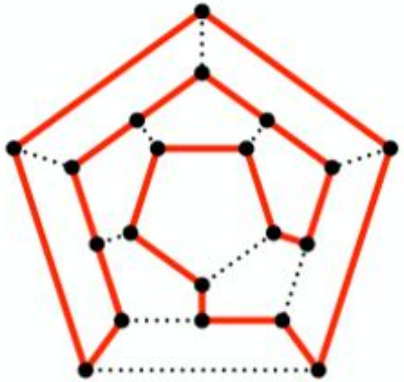
… und eine solche kann man in $O(|E|)$ Zeit finden

Satz: Das Problem

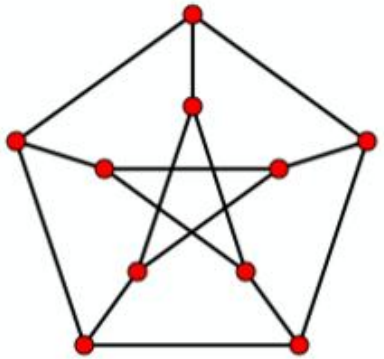
„Gegeben ein Graph $G = (V, E)$, enthält G einen Hamiltonkreis?“

kann man in Zeit $O(|V|^2 \cdot 2^{|V|})$ entscheiden und, falls ja, einen solchen finden.

Cycles



Ikosaeder

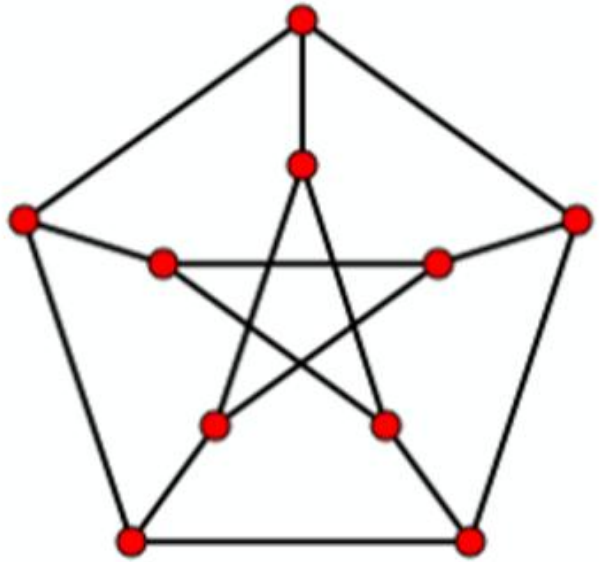


Petersengraph

Cycles

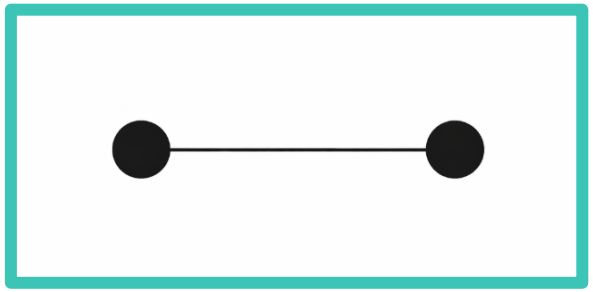
One of the interesting properties of the Petersengraph:

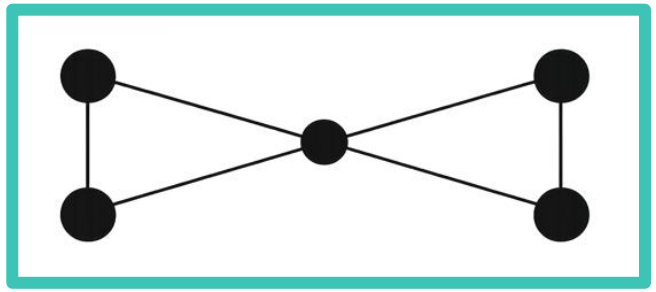
- There is a Hamiltonian path starting from every vertex (, and still, there is no Hamiltonian cycle)

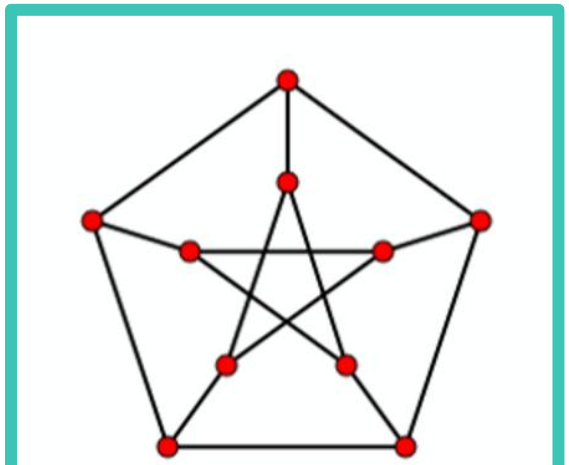


Petersengraph

Graphs for easy counterexamples:

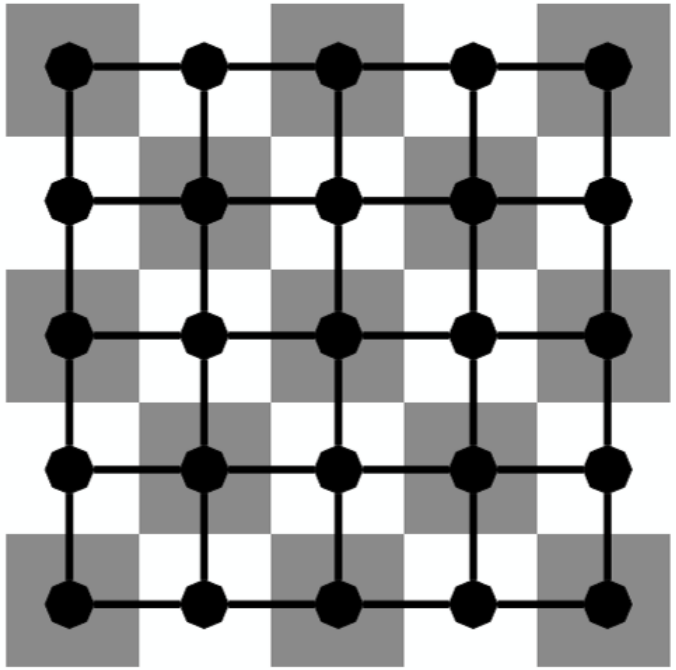






Petersengraph

Cycles



Satz: Seien m,n ≥ 2.

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

Cycles

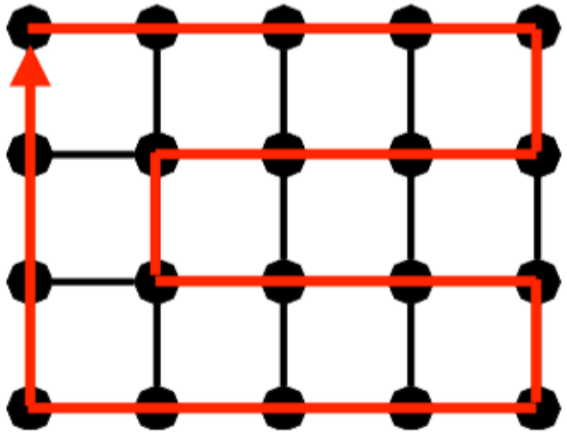
Satz: Seien m,n ≥ 2.

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

Beweis:

„⇒“ Schachbrett-Argument

„⇐“ siehe Skizze:



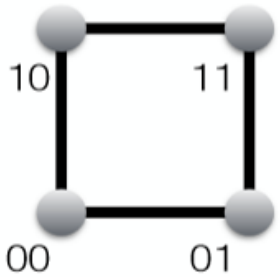
Cycles

d-dimensionaler Hyperwürfel H_d

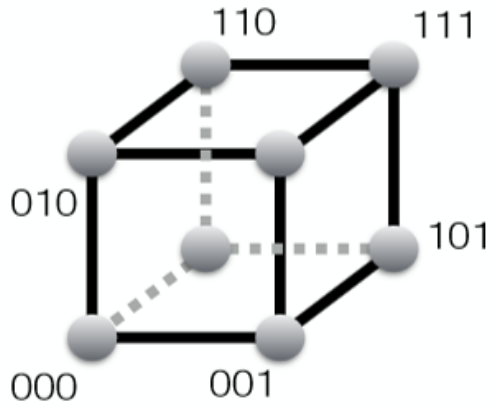
Knotenmenge: $\{0,1\}^d$

Kantenmenge: alle Knotenpaare, die sich in genau einer Koordinate unterscheiden

d=2:



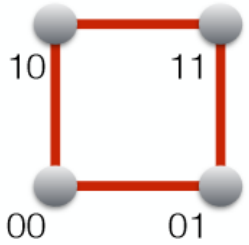
d=3:



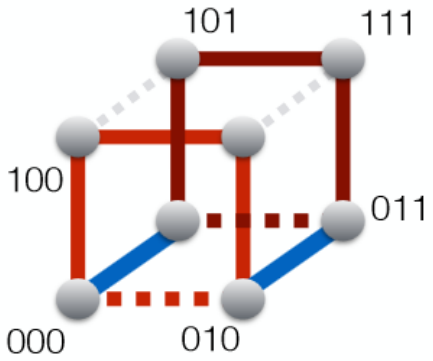
Enthält H_d einen Hamiltonkreis ?

Cycles

d=2:



d=3:



00
10
11
01

analog für d ≧ 4

!!! IMPORTANT !!!

Satz: (Dirac 1952)

Jeder Graph $G = (V, E)$ mit $|V| \geq 3$ und Minimalgrad

$\delta(G) \geq |V|/2$ enthält einen Hamiltonkreis.

Paul Dirac
(1902-1984)

Cycles

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

Dann gilt:

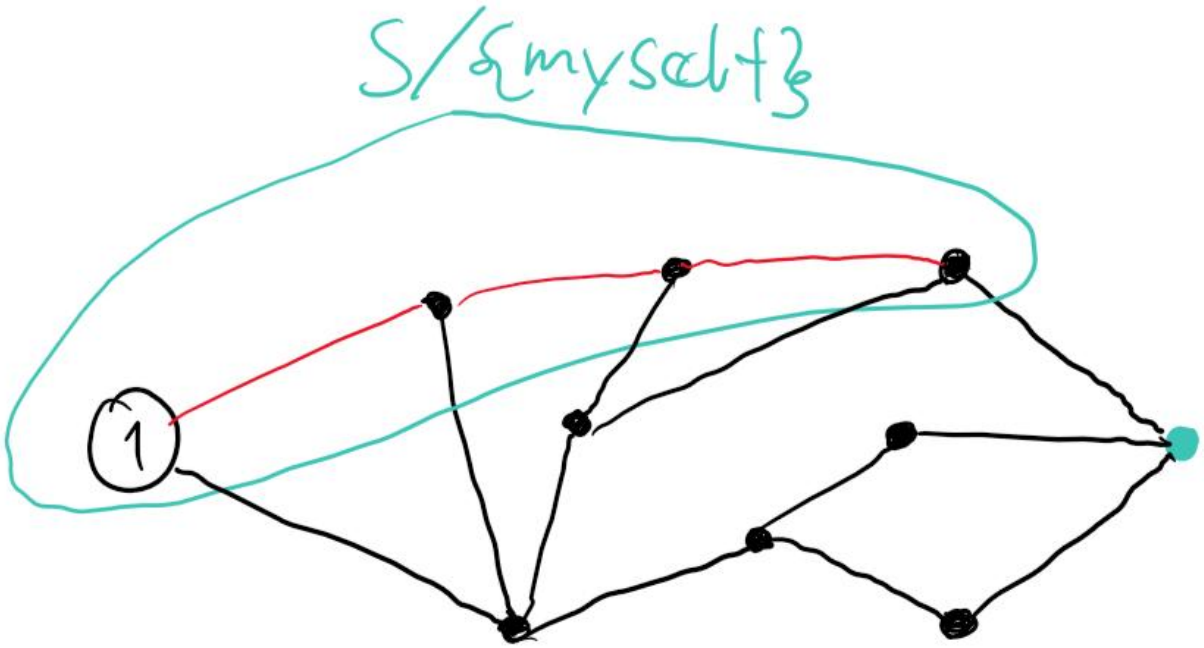
$$G \text{ enthält einen Hamiltonkreis} \iff \exists\, x \in N(1) \text{ mit } P_{[n],x} = 1$$

Cycles

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

1: *// Initialisierung*
2: **for all** $x \in [n]$, $x \neq 1$ **do**
3: $$P_{\{1,x\},x} := \begin{cases} 1, & \text{falls } \{1,x\} \in E \\ 0, & \text{sonst} \end{cases}$$
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**
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

Cycles



What paths with S/{myself} are there already to my neighbours?

Cycles

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

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

Intuition:

$n \cdot$
$2^n \cdot$
n
$\Big\} O(n^2 2^n)$

Satz 1.34. Algorithmus HAMILTONKREIS ist korrekt und benötigt Speicher $O(n \cdot 2^n)$ und Laufzeit $O(n^2 \cdot 2^n)$, wobei $n = |V|$.

Exercise Preview

The following exercise is worth 2 bonus points.

Exercise T1.1 – *Disjoint Paths*

Let $G = (V, E)$ be an undirected graph. For two vertices $s, t \in V$ we denote by $\lambda_{s,t}$ the maximal number of edge-disjoint path between s and t in G.

(a) Show that for three pairwise different vertices $a, b, c \in V$ we have $\lambda_{a,c} \geq \min\{\lambda_{a,b}, \lambda_{b,c}\}$.

(b) Let $x, y, z \in V$ be three pairwise different vertices and $\alpha, \beta \in \mathbb{N}$ such that $\alpha \leq \lambda_{x,y}$, $\beta \leq \lambda_{x,z}$, and $\alpha + \beta \leq \max\{\lambda_{x,y}, \lambda_{x,z}\}$. Show that there are α many x-y paths and β many x-z-path such that all $\alpha + \beta$ paths are pairwise edge-disjoint.
(Hint: try to construct a graph that contains a vertex v such that the task reduces to finding $\alpha + \beta$ edge-disjoint x-v paths.)

A general approach

- Understand the property P you need to prove
- Try to map the problem onto a different one (on a different graph G') s.t.

  P' holds for G' → P holds for G (1)

- Construct the new graph for which a modified property P' is easier to prove
- Prove P'
- Make use of (1)

The following exercise does not give bonus points.

Exercise T1.2 – *Connectivity*

(a) Prove or give a counterexample to the following claim: If G is a connected graph such that every vertex of G has even degree, then G is 2-edge-connected.

(b) Prove or give a counterexample to the following claim: If G is a connected graph such that every vertex has degree at least 4, then G is 2-edge-connected.

(c) Prove or give a counterexample to the following claim: For a graph $G = (V, E)$, we define a relation on *vertices* of G: vertices $a \in V$ and $b \in V$ satisfy $a \sim b$ if there is a cycle in G containing both a and b. Then for every graph G relation $\sim$ is an equivalence relation.