Algorithms and Probability PVK

Day 1

Michelle Honegger - CAB G59

Organization

- Theory Recap
- Exercises
- Coding
- Your session, you decide

Today

Graph Theory

- Connectedness, Cut-Vertices/-Edges, DFS with Low Values
- Hamiltonian Cycles, Dirac
- TSP 2 Approximation, Matchings, TSP 1.5 Approximation
- Colorings

Tuesday

Probability

- Intro, Add/Mult Rule, Inclusion/Exclusion Principle, CombinatoricsMinimale Schnitte
- Conditional Probability, Law of Total Probability, Bayes, Independence
- Random Variables, PDF/CDF, EV, Linearity of Expectation, Indicator Variables
- Distributed Algorithms, Variance, Std, Coupon collector
- Distributions

Wednesday

Randomised Algorithms

- Intro, Las Vegas, Monte Carlo, Üppige Auswahl
- Primality Testing
- Quick Sort, Quick Select
- Finding Duplicates, Hase-Igel
- Rainbow Algorithm

Thursday

Networks

- Intro, Max-Flow-Min-Cut Theorem
- Various Problems to solve with Networks I
- Various Problems to solve with Networks II
- Bootstrapping

Friday

Geometric Algorithms

- Smallest Enclosing Circle
- Convexity, Jarvis Wrap
- Local Repair

Theory Recap

k-Connectedness

- A Graph is k-connected, if it holds that:
  - $|V| \geq k + 1$ und $\forall X \subseteq V$ is $G[V \backslash X]$ connected
  - One has to delete at least k vertices in order to lose connectedness
- A Graph ist k-edge-connected, if it holds that:
  - $\forall X \subseteq E$ mit $|X| < k$ is $(V, E \backslash X)$ connected

K-connectedness Cont'd

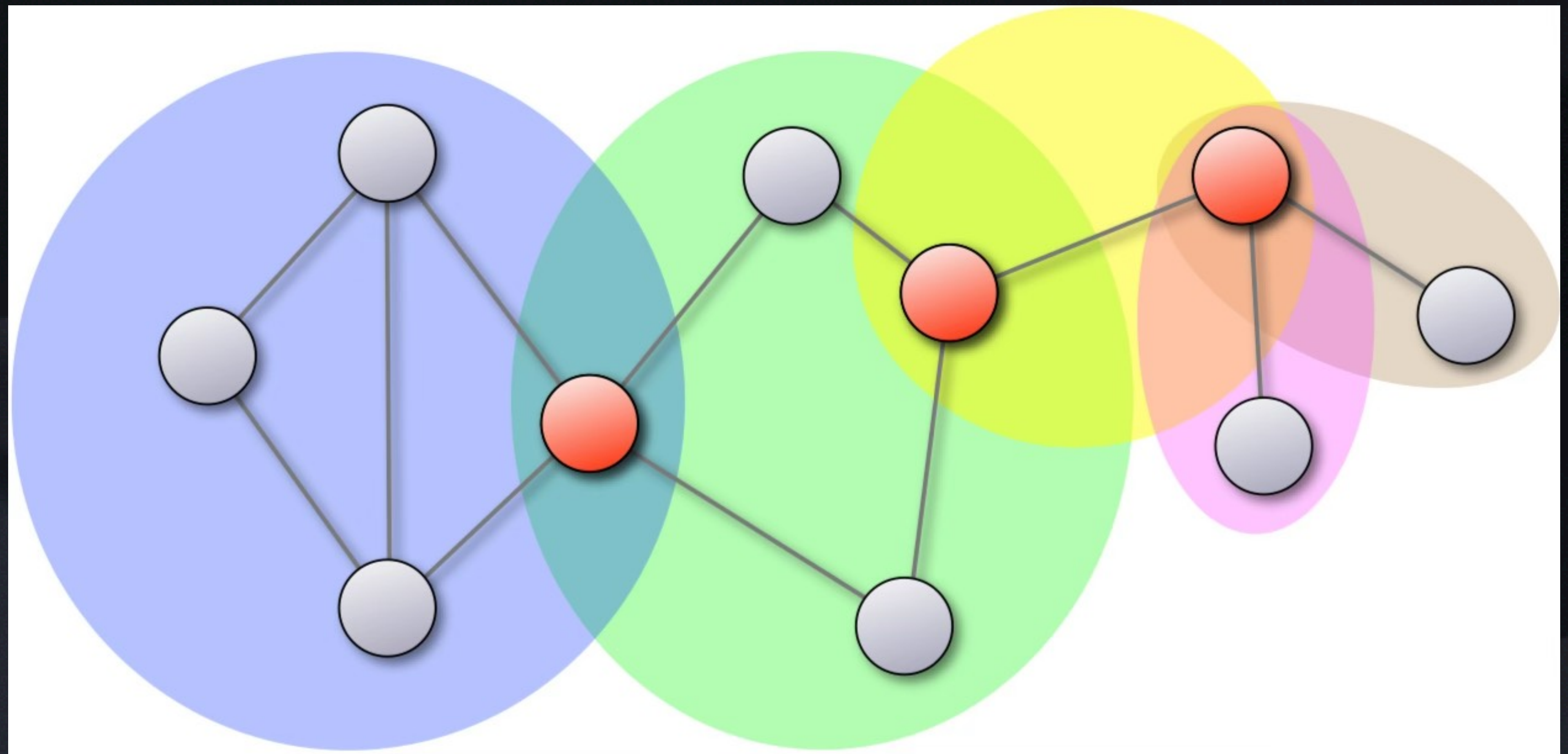
- Menger:
  G is k-connected iff. $\forall u, v \in V, u \neq v$ there exist k internally-vertex-disjoint u-v-paths
  G is k-edge-connected iff. $\forall u, v \in V, u \neq v$ there exist k edge-disjoint u-v-paths

- (Vertex-)Connectedness ≤ Edge-Connectedness ≤ Minimal Degree

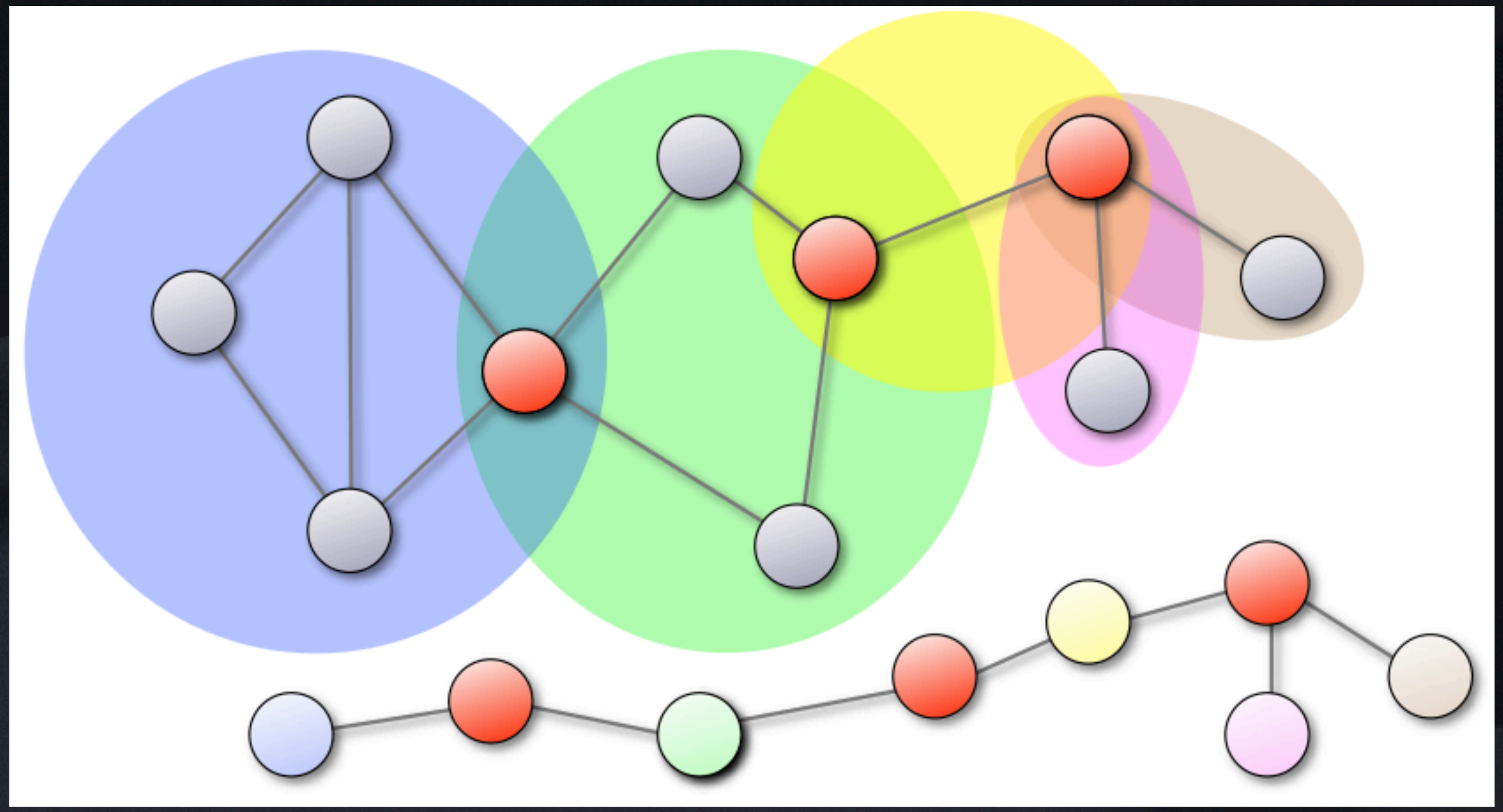
Articulation Points & Bridges ★

- A vertex is called **Articulation Point**, if the graph after its removal is not connected anymore.
- An edge is called **Bridge**, if the graph after its removal is not connected anymore.
- Bridge {x,y} -> deg(x) = 1 or x ist Articulation Point (Other Direction doesn’t hold)

Blocks

- $e \sim f :\Leftrightarrow \begin{cases} e = f, & \text{or} \\ \exists \text{ cycle through } e \text{ and } f \end{cases}$
- Two Blocks overlap - if at all - always in an articulation point
- E and f are edges
- 

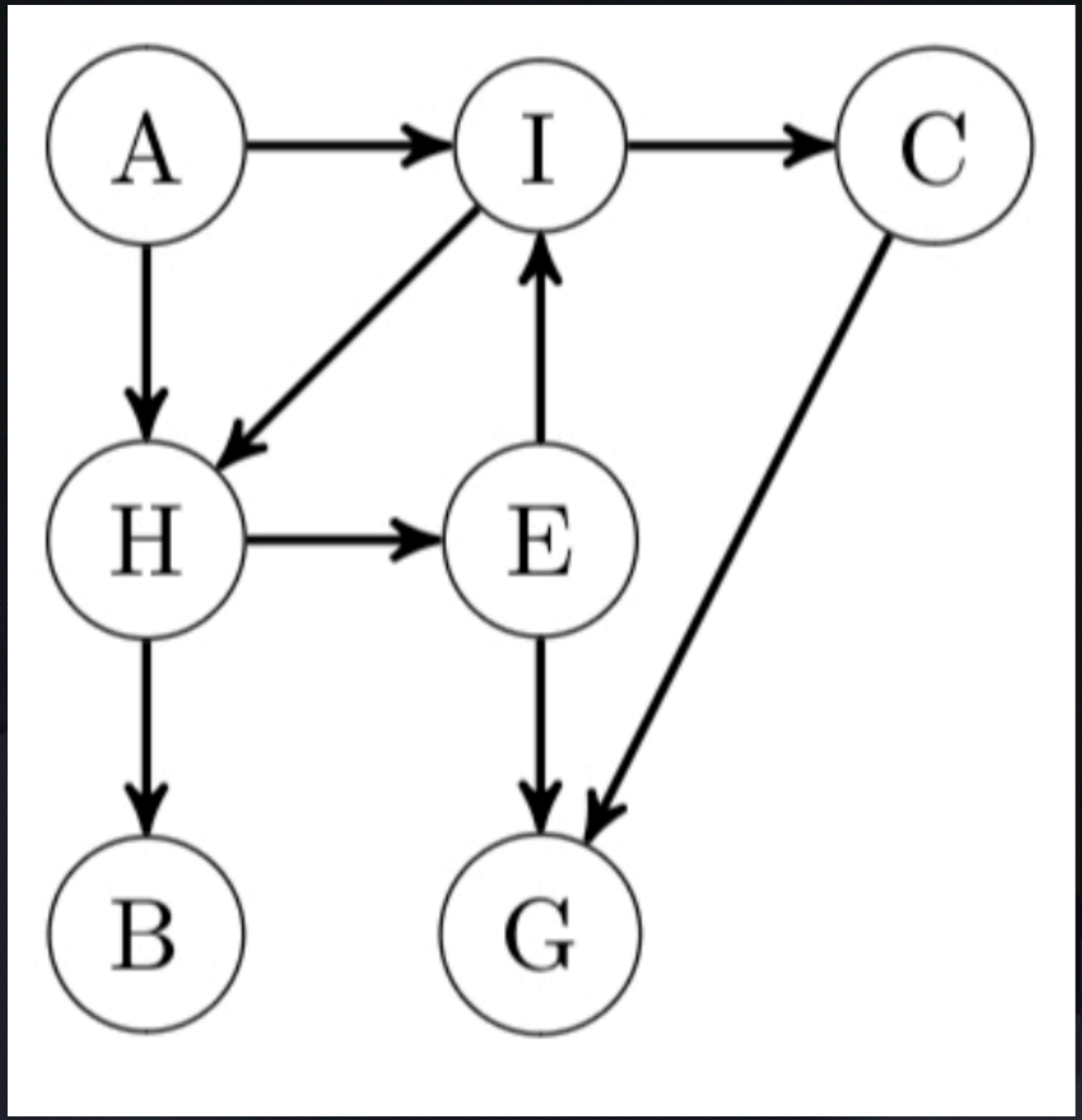
Blockgraph

- Bipartite Graph $T = (A \uplus B, E_T)$
  - A = {Articulation point of G}, B = {Block of G}
  - $\forall a \in A, b \in B : a, b \in E_T \Leftrightarrow$ a incident to edge in b
- 

Recap DFS/BFS

- BFS with FIFO Queue - Useful for Shortest Paths

- DFS with LIFO Stack - Useful for Articulation Points/ Bridges



DFS with Low-Values

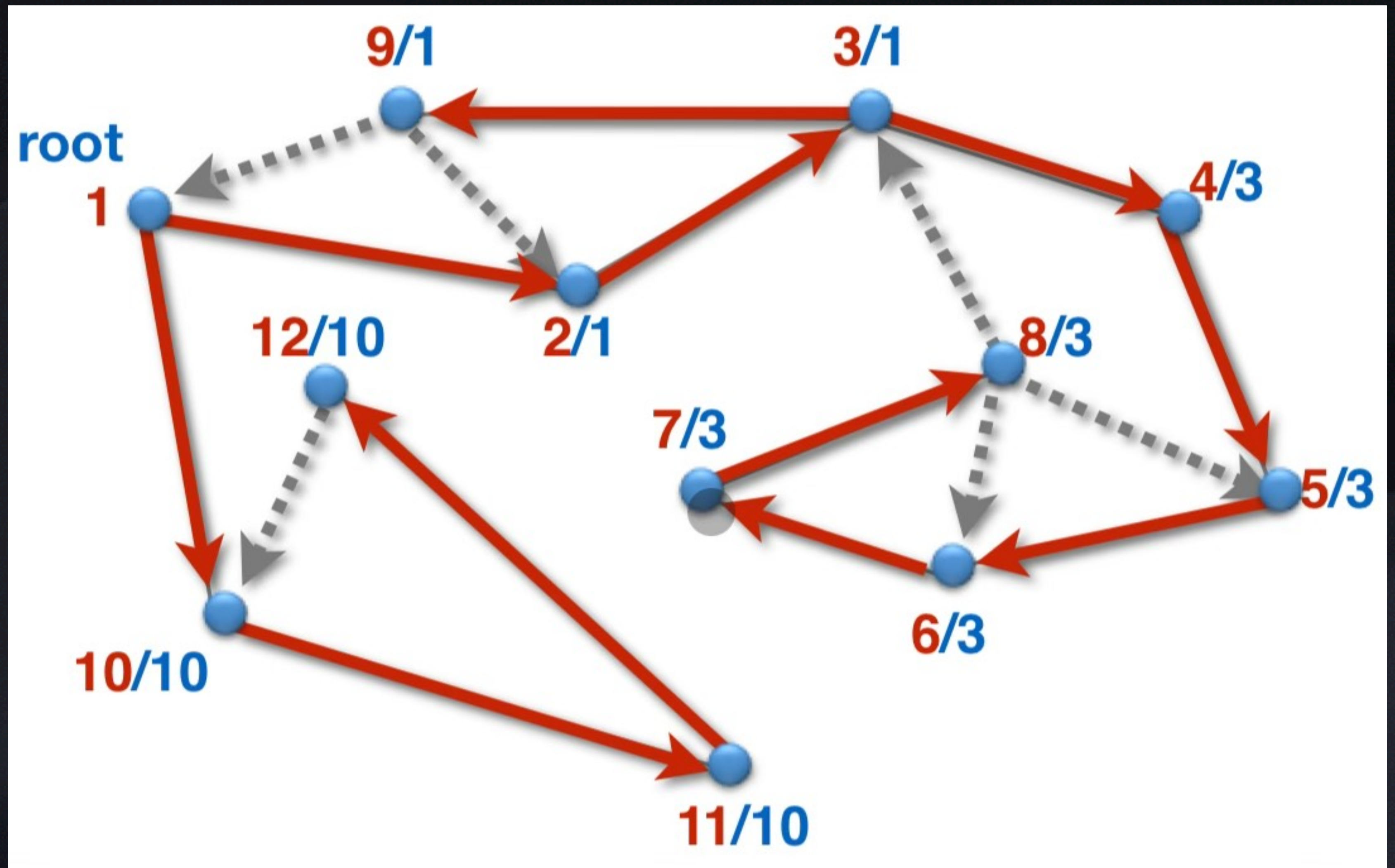
★

- low[v]:= smallest dfs-number, that can be reached from v via a directed path with tree edges and at most one rest edge.

- v is articulation point iff. v ≠ root and v has a child w in DFS-tree with low[w] ≥ dfs[v] or v = root and v has at least 2 children in DFS-Tree

- A Tree edge (v,w) is a bridge iff. low[w] > dfs[v]

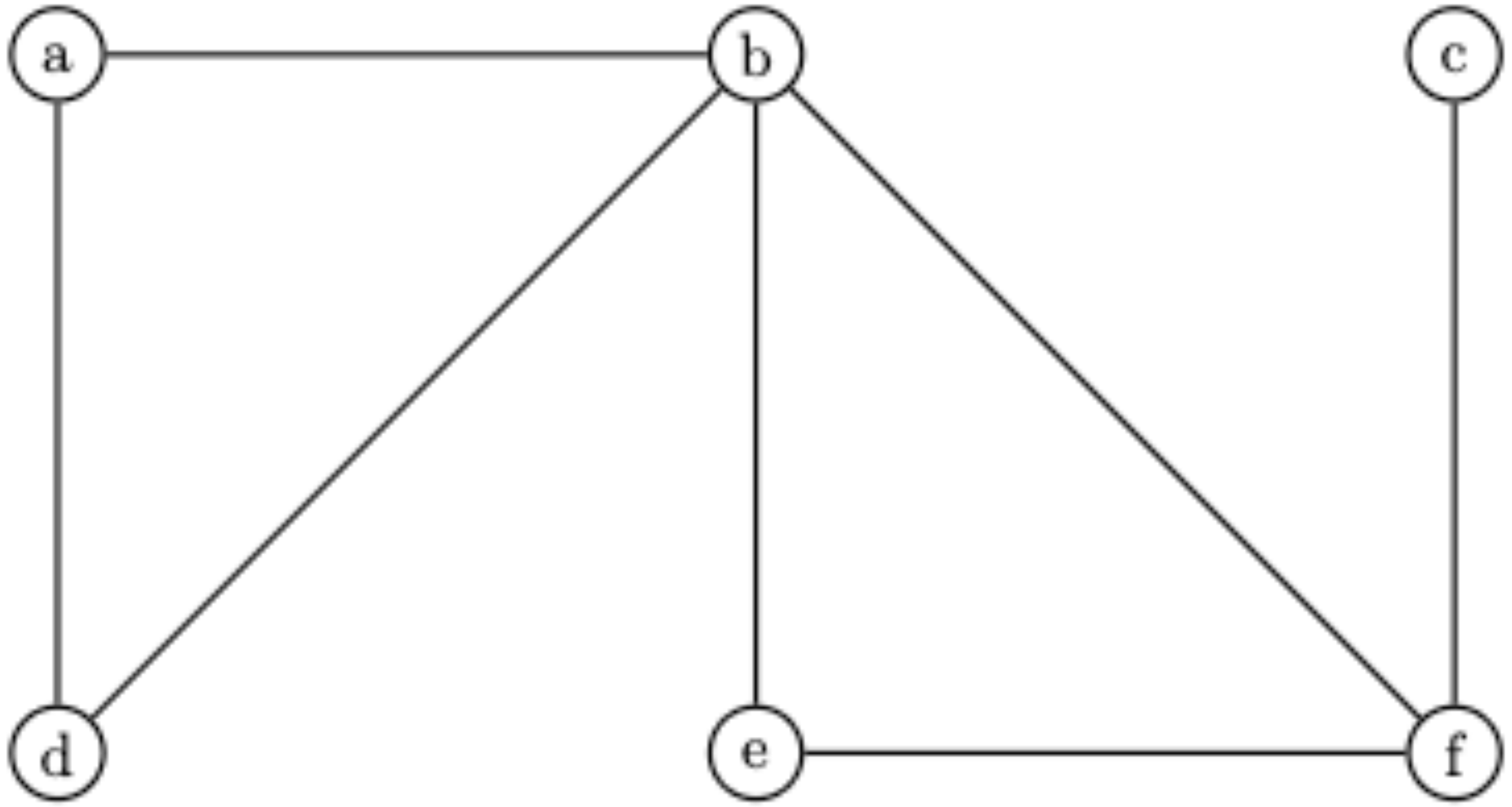
- Runs in O(m)



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.



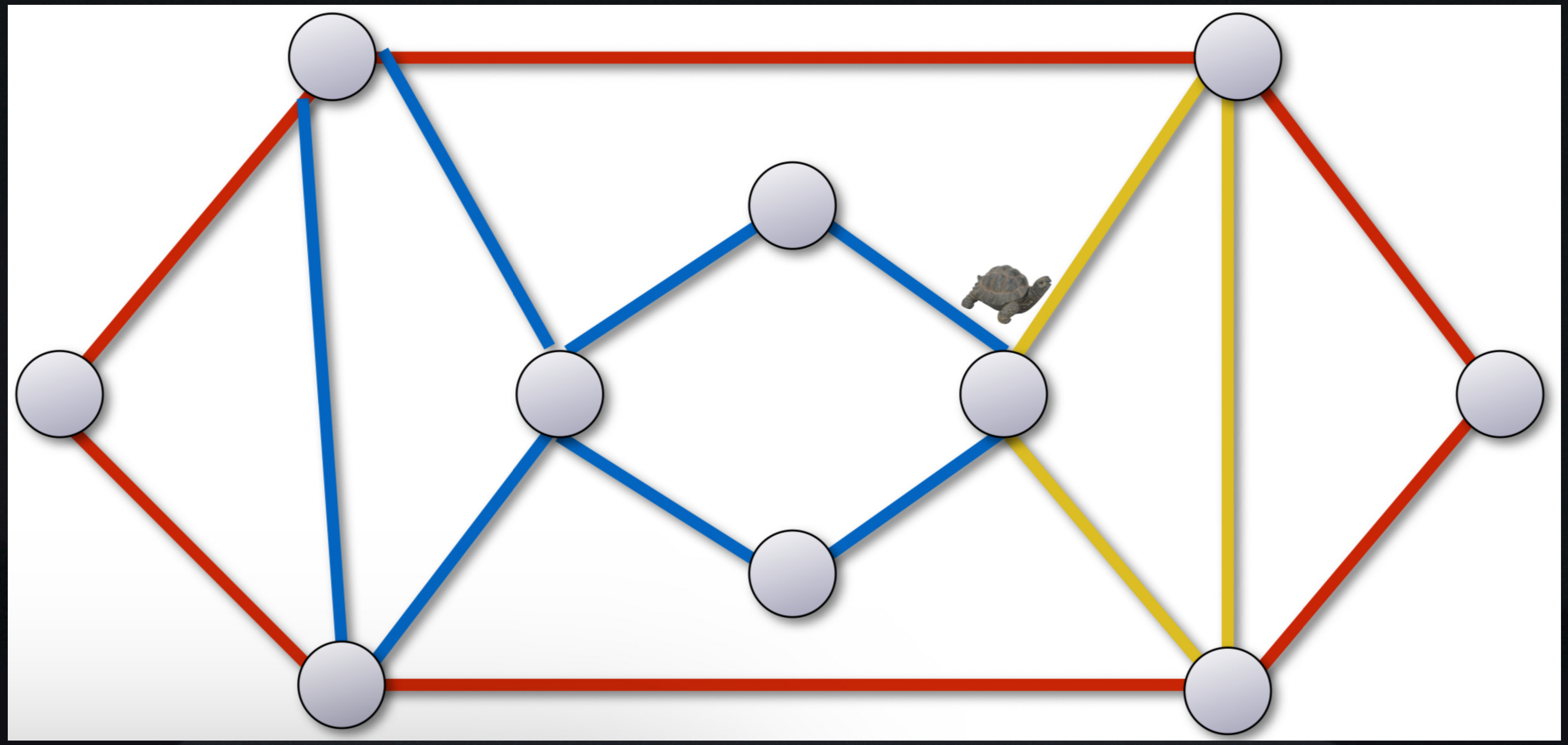
(a) Perform the modified DFS starting at vertex a. Use the corresponding `dfs`- and `low`-values to determine the cut-vertices and bridges of G.

(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.

Eulerian tours

- Every edge exactly once - same start as end vertex
- Exists iff. Degree of all vertices is even
- Can be found in O(|E|)



- If all vertices except for 2 have even degree there exists a eulerian path

Hamiltonian cycle

- Every vertex exactly once - same start as end vertex
- In general NP-complete

Hamiltonian Cycles

Special cases

- n x m Grid **contains** Hamiltonian cycle iff. **n*m even** (for n, m ≥ 2)
- Ikosaeder **contains** Hamiltonian cycle
- d-dimensional Hypercupe **contains** Hamiltonian cycle
- Every Graph G = (V, E) with |V| ≥ 3 and minimum degree δ(G) ≥ |V|/2 **contains** Hamiltonian cycle (Dirac)
- Petersengraph **doesn’t contain** Hamiltonian cycle

Exam Task

(a) Sei G ein Graph, der einen Hamiltonkreis hat. Dann ist G 2-zusammenhängend.

WAHR ☐ FALSCH ☐

Begründung/Gegenbeispiel: *(2 Punkte)*

TSP

- Given: complete Graph on n Vertices, distances between all pairs of vertices:
$l : \binom{[n]}{2} \to \mathbb{R}$

- Seeked: shortest roundtrip: $\min_{H:Hamiltoniancycle} \sum_{e \in E(H)} l(e)$

- NP-complete, because if TSP in polynomial time solvable, then Hamiltonian cycle would also be solvable in polynomial time.

Metrical TSP

2-Approximation

- Function l fulfils Triangle inequality: $l(x, z) \leq l(x, y) + l(y, z) \qquad \forall x, y, z \in [n]$
    - Find MST T it holds that: $l(T) \leq opt(K_n, l)$
    - Double all edges of T: $2l(T) \leq 2opt(K_n, l)$
    - Find Eulerian Tour W: $l(W) = 2l(T) \leq 2opt(K_n, l)$
    - Walk through W with shortcuts, such that every vertex is visited only once => Hamiltonian cycle C: $l(C) \leq l(W) = 2l(T) \leq 2opt(K_n, l)$

Exercises

- How many edges are there in K_n?
- Show that every Graph G = (V,E) with |V| = n ≥ 3 and |E| ≥ $\frac{(n-1)(n-2)}{2} + 1$ is connected.
- Find a Graph G = (V,E) with |V| = n ≥ 3 and |E| ≥ $\frac{(n-1)(n-2)}{2} + 1$ which doesn't contain a Hamiltonian cycle.
- Show that every Graph G = (V,E) with the property, that deg(u) + deg(v) ≥ n for $u, v \in V$ not adjacent, contains a Hamiltonian cycle.
- Show that every Graph G = (V,E) with |V| = n ≥ 3 and |E| ≥ $\frac{(n-1)(n-2)}{2} + 2$, contains a Hamiltonian cycle.

Matchings

- **Matching**: A Edgeset M $\subseteq$ E is called Matching in a Graph G = (V,E), if there is no vertex adjacent to more than one edge in M.
- A Vertex v is **covered** by M, if there exists an edge e $\in$ M, that contains v.
- **Perfect Matching**: If all Vertices are covered by one edge in M, then: $|M| = |V|/2$
- **Inklusionsmaximal (maximal)**: If there doesn't exist any Matching M' with M $\subseteq$ M' and $|M'| > |M|$
- **Kardinalitätsmaximal (maximum)**: If there doesn't exist any Matchen M' with $|M'| > |M|$

Matchings

- If M_{inc} a inklusionsmaximales (maximal) Matching and M_{max} a kardinalitätsmaximales (maximum) Matching, then: $|M_{inc}| \geq |M_{max}/2|$

Matchings

Greedy Algorithmus

- Runtime: O(|E|)
- $|M_{greedy}| \geq |M_{max}/2|$

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

Augmenting Paths

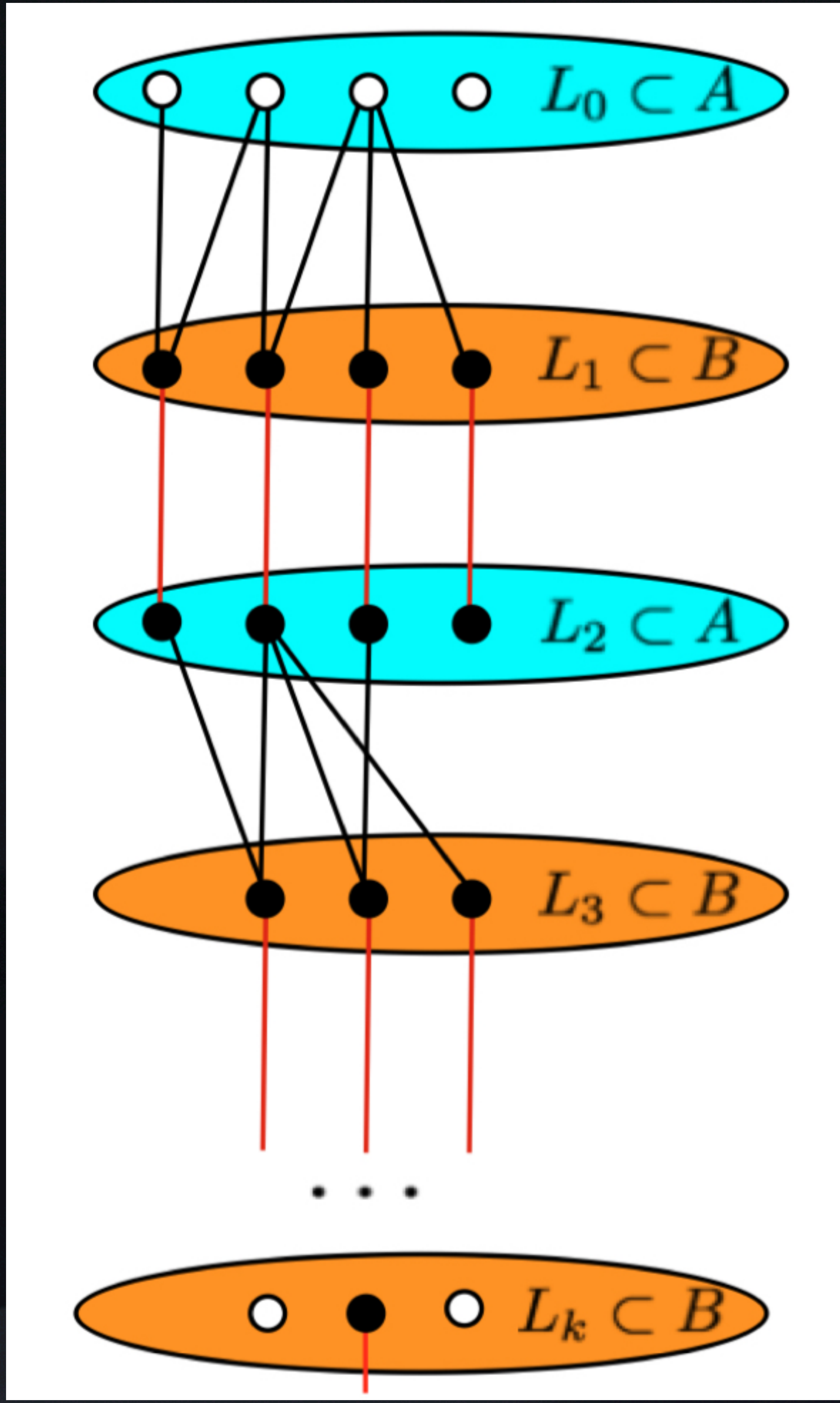
Hopcroft and Karp Algorithm

- Runtime: 1 Iteration O(|V|+|E|)
- Runtime: all Paths $O(\sqrt{|V|} * |E|)$
- Blossom-Algorithmus for general graphs in O(|V|*|E|)

MAXIMAL_MATCHING $(G = (A \oplus B, E))$ (Hopcroft und Karp)

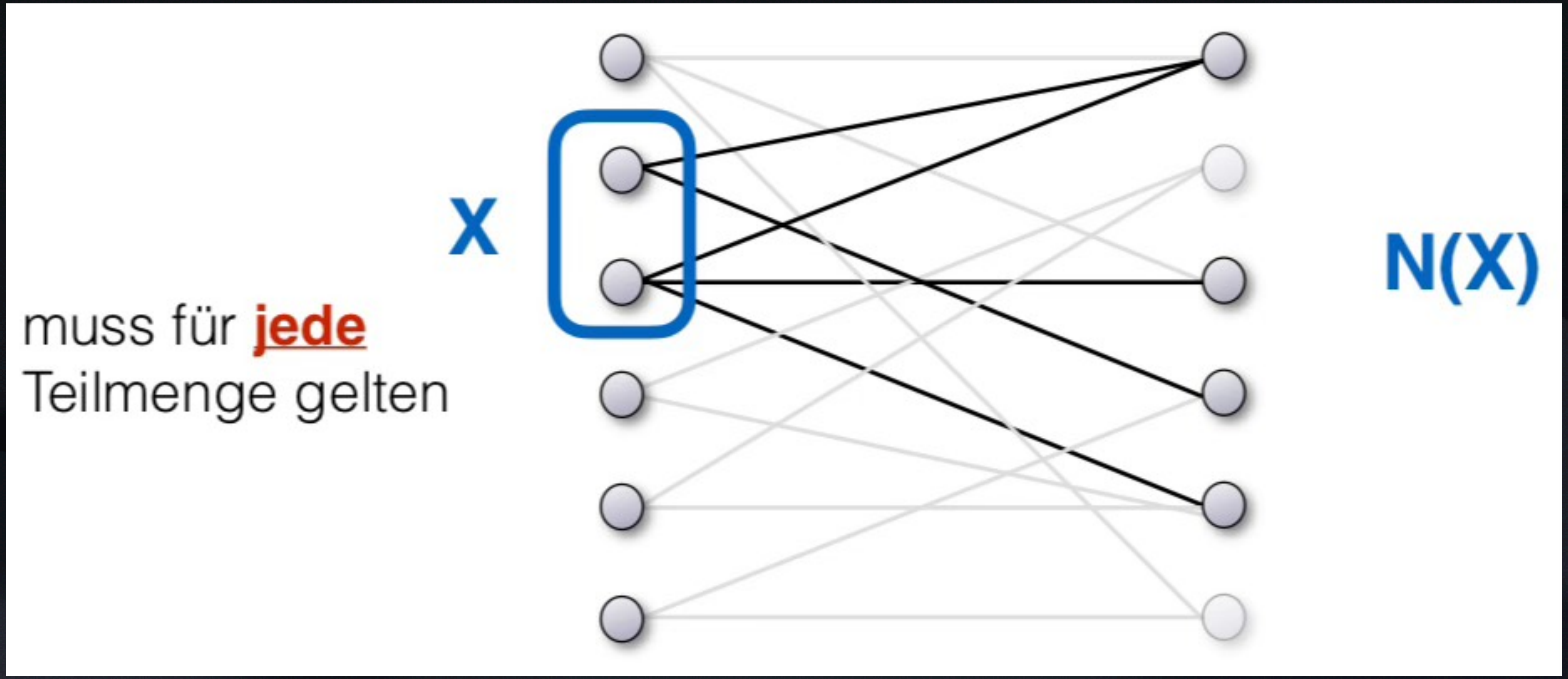
$M := \{e\}$ für irgendeine Kante $e \in E$.
while es gibt noch augmentierende Pfade **do**
    $k :=$ Länge eines kürzesten augmentierenden Pfades
    Finde eine inklusionsmaximale Menge S von paarweise disjunkten augmentierenden Pfaden der Länge k.
    for all P aus S **do**
        $M := M \oplus P$. // *augmentiere entlang der Pfade aus S*
return M

Matchings



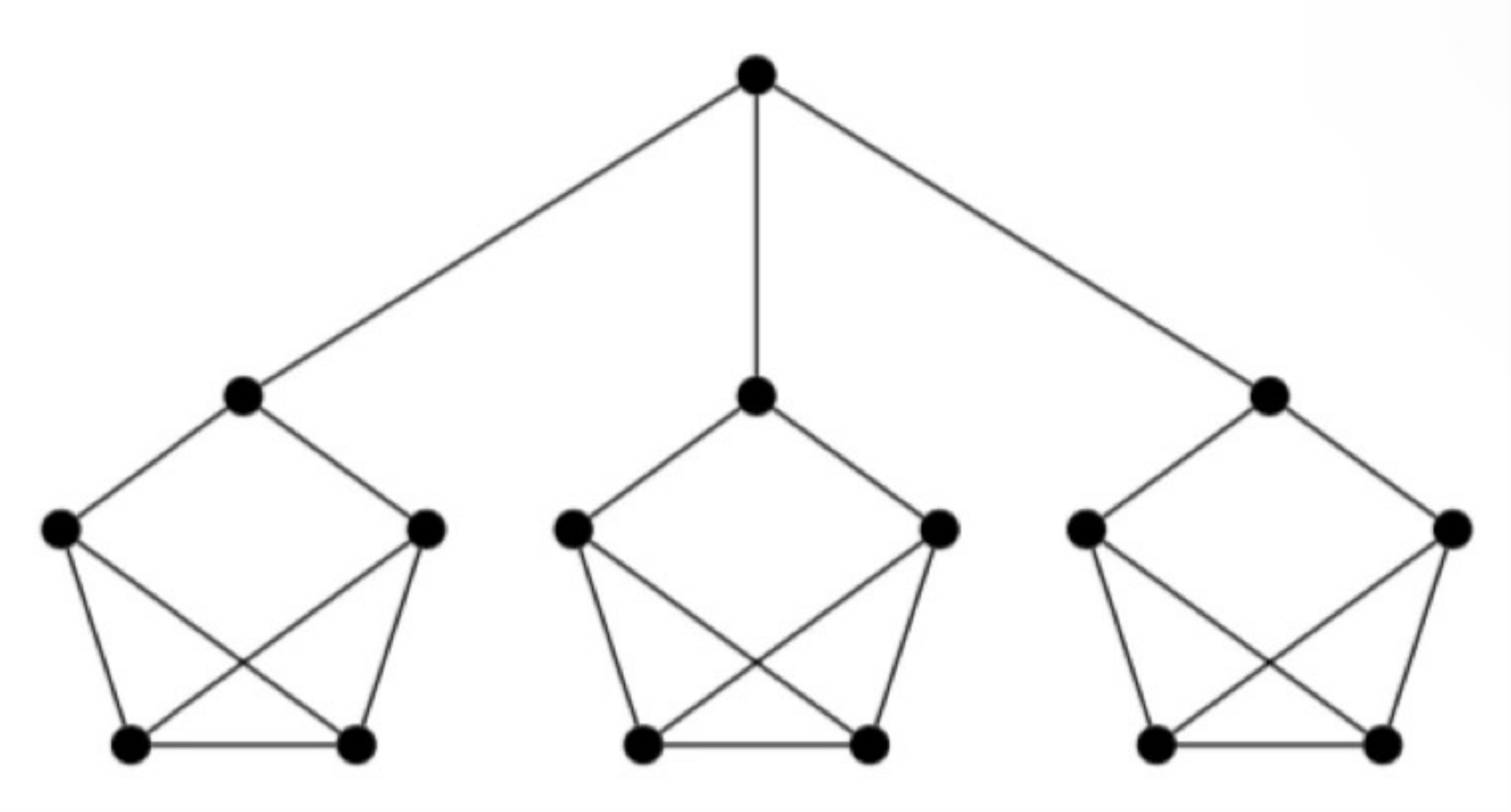
Hall's Theorem (Heiratssatz)

- A bipartite Graph $G=(A \uplus B,E)$ contains a Matching M of cardinality $|M|=|A|$ iff.
  $\forall X \subseteq A : |X| \leq |N(X)|$



Frobenius

- For all k it holds: every k-regular **bipartite** Graph contains a perfect Matching.
- s:= |(edges from X to N(X)|
  - s = k*|X| ⇒ s ≤ k*|N(X)| ⇒ |X| ≤ |N(X)|



Matchings

Gabow Algorithm

- In 2^k-regular bipartite Graphs one can find a perfect matching in O(|E|).

- Because is Graph 2^k-regular, all Vertex Degrees are even ⇒ Graph contains Eulerian tour which we can find in O(|E|).
  - We delete every other edge and iterate until we find the Matching.

- Can be generalised for k-regular bipartite Graphs. (Cole, Ost, Schirra)

Metrical TSP

1.5-Approximation

- Function l fulfils Triangle Inequality: $l(x, z) \leq l(x, y) + l(y, z) \qquad \forall x, y, z \in [n]$
    - Find MST T it holds: $l(T) \leq opt(K_n, l)$
    - X:= Vertices with odd degree in T; Find minimal Matching M for X $l(M) \leq \frac{1}{2} opt(K_n, l)$
    - Find Eulerian tour W: $l(W) = l(T) + l(M) \leq \frac{3}{2} opt(K_n, l)$
    - Walk through W with shortcuts, such that every vertex is visited only once => Hamiltonian Cycle C: $l(C) \leq l(W) = l(T) + l(M) \leq \frac{3}{2} opt(K_n, l)$

Exam Task

(f) Sei G ein vollständiger Graph mit einer Gewichtsfunktion, die auch negative Werte annehmen kann. Dann gilt dass das Gewicht eines minimalen Spannbaums höchstens so gross ist wie eine optimale TSP Route.

WAHR ☐ FALSCH ☐

Begründung/Gegenbeispiel: *(3 Punkte)*

Exercise S3.1 – *Bipartite Matching*

Let $G = (A \dot\cup B, E)$ be a bipartite graph.

(a) Prove or disprove: If G has a Hamiltonian cycle, then G contains two disjoint perfect matchings.

(b) Prove or disprove: If G contains two disjoint perfect matchings, then G has a Hamiltonian cycle.

(c) Let $A' \subseteq A$ and $B' \subseteq B$. Assume that there is a matching M_A covering A' and a matching M_B covering B' (these two matchings do not have to be disjoint!). Show that there is a matching M that covers both A' and B' (i.e. it covers $A' \cup B'$). (*Hint:* Which properties has the graph $(V, M_A \cup M_B)$? Try to build a matching using only edges in $M_A \cup M_B$.)

Exercise S4.1 – *Perfect Bipartite Matching*

Let $G = (A \dot{\cup} B, E)$ be a bipartite k-regular graph (i.e. every vertex has degree exactly k).

(a) Show that G contains a perfect matching.

(b) Show that G contains k pairwise disjoint perfect matchings.
Hint: use (a)

Exam Task

Aufgabe 5 – *Maximale Matchings*

In dieser Aufgabe beschäftigen wir uns mit dem in der Vorlesung kennengelernten Algorithmus von Hopcroft und Karp zum Finden maximaler Matchings in bipartiten Graphen.

Sei $G = (A \uplus B, E)$ ein bipartiter Graph mit $|A| = |B| = n$.

(a) Füllen Sie die Lücken "..............." in unten stehendem Algorithmus so aus, dass er ein maximales Matching in G findet. *(3 Punkte)*

function Maximal_Matching(G):

M := ....

while .... **do**:

k :=...

Finde eine inklusionsmaximale Menge S von

....

for all P aus S **do**

M = ...

return M

Colorings

- A Vertex colouring of a Graph G = (V,E) with k colours is a mapping c: V -> [k], such that c(u) ≠ c(v) for all edges {u,v} in E
- The **chromatical number** X(G) is the minimal number of colours needed to colour the graph G
- We can check whether a graph is bipartite in O(|E|) with BFS/DFS
- For each k >= 3 the problem: Given a graph G =(V,E), does it hold that X(G) <= k? NP-complete

Colorings

Greedy Algorithm

- For each order the Greedy Algorithm needs at most $\Delta(G)+1$ colours

Greedy-Färbung (G)

wähle eine beliebige Reihenfolge der Knoten: $V = \{v_1, \ldots, v_n\}$
$c[v_1] \leftarrow 1$
for $i = 2$ **to** $i = n$ **do**
    $c[v_i] \leftarrow \min\{k \in \mathbb{N} \mid k \neq c(u)$ für alle $u \in N(v_i) \cap \{v_1, \ldots, v_{i-1}\}\}$

- $\Delta(G)$:= maximal degree in G

- There exists an order for which the Greedy algorithm needs X(G) colours

- There exist bipartite graphs and an order for which the greedy algorithm needs $|V|/2$ colours

Heuristic

- We are looking for vertex with smallest degree and remove it
- Repeat until we have all vertices
- In case G fulfils that in every subgraph there exists a vertex with degree <= k, the heuristic gives an order which uses at most k+1 colours
- For trees we always find a 2-colouring with the heuristic

Brooks Theorem

- Is G = (V,E) a connected graph, which is neither complete nor a cycle of odd length, then it holds that X(G) <= Δ(G) and there exists an algorithm, which colours the vertices in O(|E|) with Δ(G) colours.

Planar Graphs

- Every Map can be coloured with 4 colours (planar graphs)
- A planar graph is a graph without crossing edges.
- Planar graphs have at least 1 vertex with degree <=5
- Heuristic finds colouring with <= 6 colours

3-Colourability

- Choose vertex with degree > $\sqrt{n}$
- We can do that at most $\sqrt{n}$ times
- Afterwards there are at most $\sqrt{n}$ vertices left
- We need $4\sqrt{n}$ colours

Colourings

Special Cases & Additional Information

- Even cycles are 2-colourable, odd cycles are 3-colourable
- A graph is bipartite iff it doesn't contain an odd cycle as a subgraph
- A block graph G, where we can colour each block in k colours, can be coloured with k colours. (Switch of colour classes)
- Planar graphs can be coloured with 4 colours, heuristic finds colouring with 6.
- Every 3-colourable graph can be coloured in $O(|E|)$ with $O(\sqrt{|V|})$ colours

Exam Tasks

(c) Sei G ein Graph. Wenn es einen Knoten $v \in V(G)$ gibt, für den gilt, dass $G[N(v)]$ mit k Farben gefärbt werden kann, dann kann G mit $k+1$ Farben gefärbt werden.

WAHR ☐ FALSCH ☐

Begründung/Gegenbeispiel: *(2 Punkte)*

(e) Sei G ein zusammenhängender Graph mit 10 Knoten und 20 Kanten. Dann gilt $\chi(G) \leq \Delta(G)$.

WAHR ☐ FALSCH ☐

Begründung/Gegenbeispiel: *(3 Punkte)*