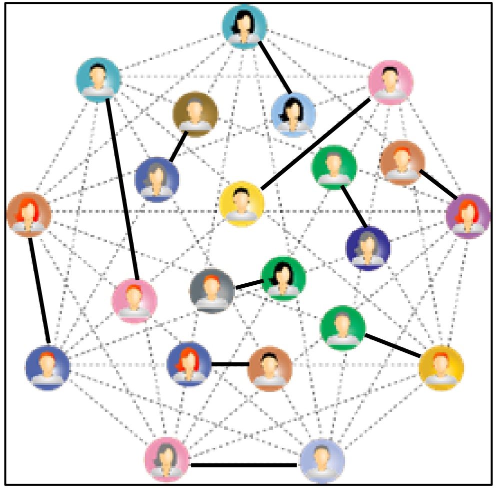
Technische Universität Braunschweig

Netzwerkalgorithmen – Vorlesung #10

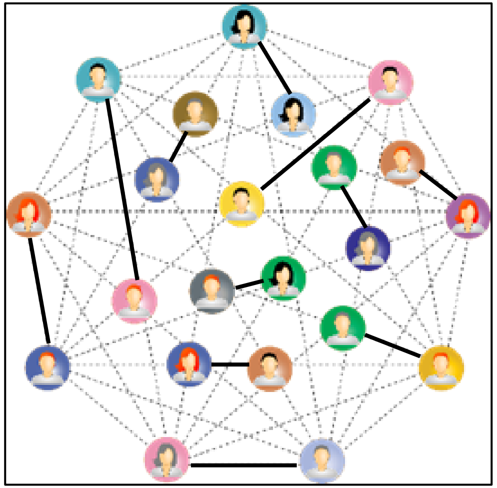
Arne Schmidt

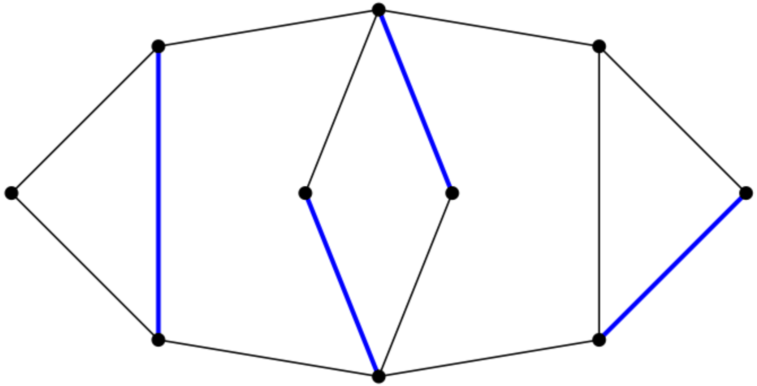
Kapitel 5



Matchings

Matchings





Matchings

Definition 5.1 (Matchings)
Sei $G = (V, E)$ ein Graph
(1) Eine Kantenmenge $M \subseteq E$ heißt **Matching**, wenn $e \cap f = \emptyset$ für je zwei Kanten $e, f \in M$ gilt.

(2) Kanten in M heißen **unabhängig**.

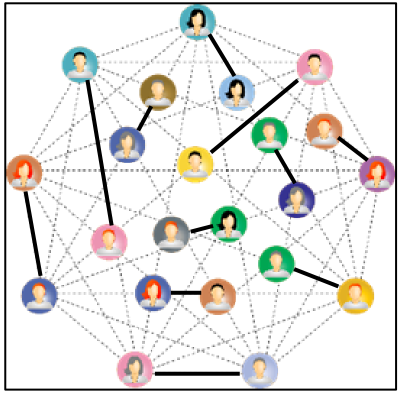
(3) Ein Matching M heißt **perfektes Matching**, wenn $2|M| = |V|$ gilt.

(4) Ein Matching heißt **inklusionsmaximal** (engl. maxim**al**), wenn $M \cup \{e\}$ für jede Kante $e \in E \setminus M$ kein Matching ist.

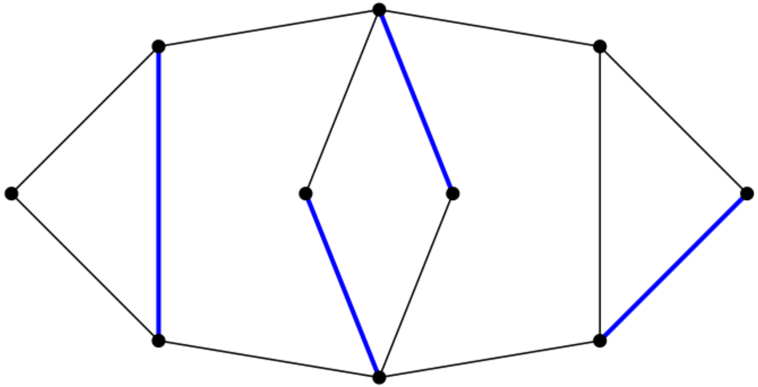
(5) Ein Matching heißt **(kardinalitäts-)maximal** (engl. maxim**um**), wenn kein Matching M' mit $|M| < |M'|$ existiert.

Matchings

Problem 5.2: Maximale Matchings (engl. Maximum Matching)
Gegeben:
  Graph $G = (V, E)$
Gesucht:
  Maximales Matching M.

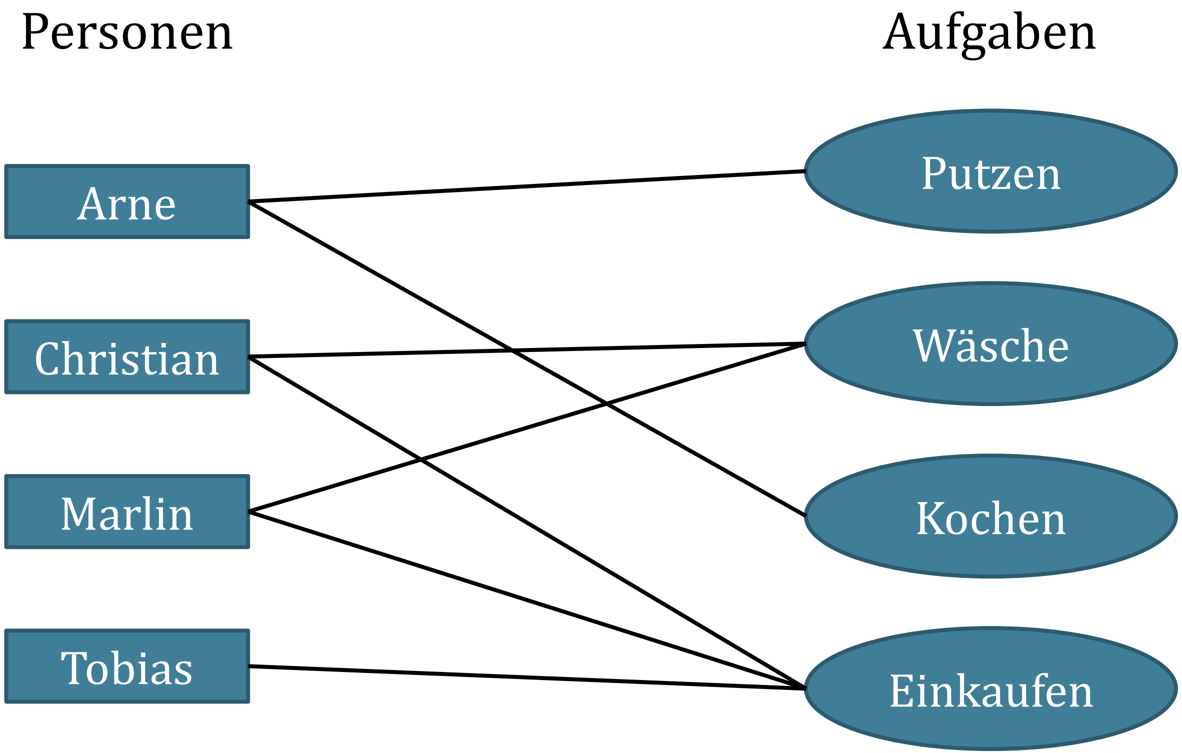


Perfektes Matching



Inklusionsmaximales Matching
Gibt es ein besseres?

Aufgabenverteilung



5.1 Bipartite Graphen

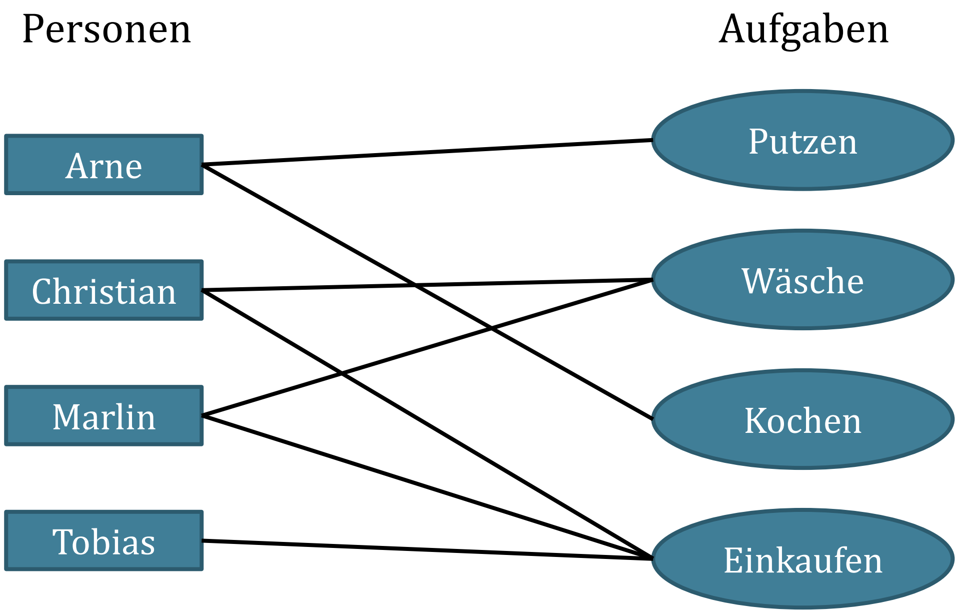
Bipartite Graphen

Definition 5.3 (Bipartite Graphen)
Sei $G = (V, E)$ ein Graph
(1) G heißt **bipartit**, wenn sich V disjunkt in zwei Knotenmengen V_1, V_2 zerlegen lässt, sodass $u \in V_1$ und $v \in V_2$ für jede Kante $\{u, v\} \in E$ gilt.

(2) Mit $K_{a,b}$ wird der **vollständig bipartite Graph** bezeichnet, d.h. $|V_1| = a, |V_2| = b$ und zwischen jedem Paar $(u, v) \in V_1 \times V_2$ verläuft eine Kante.

Satz 5.4
Sei G ein Graph. Dann sind folgende Aussagen äquivalent.
1. G ist bipartit.
2. G ist zweifärbbar.
3. G enthält keine Kreise ungerader Länge.

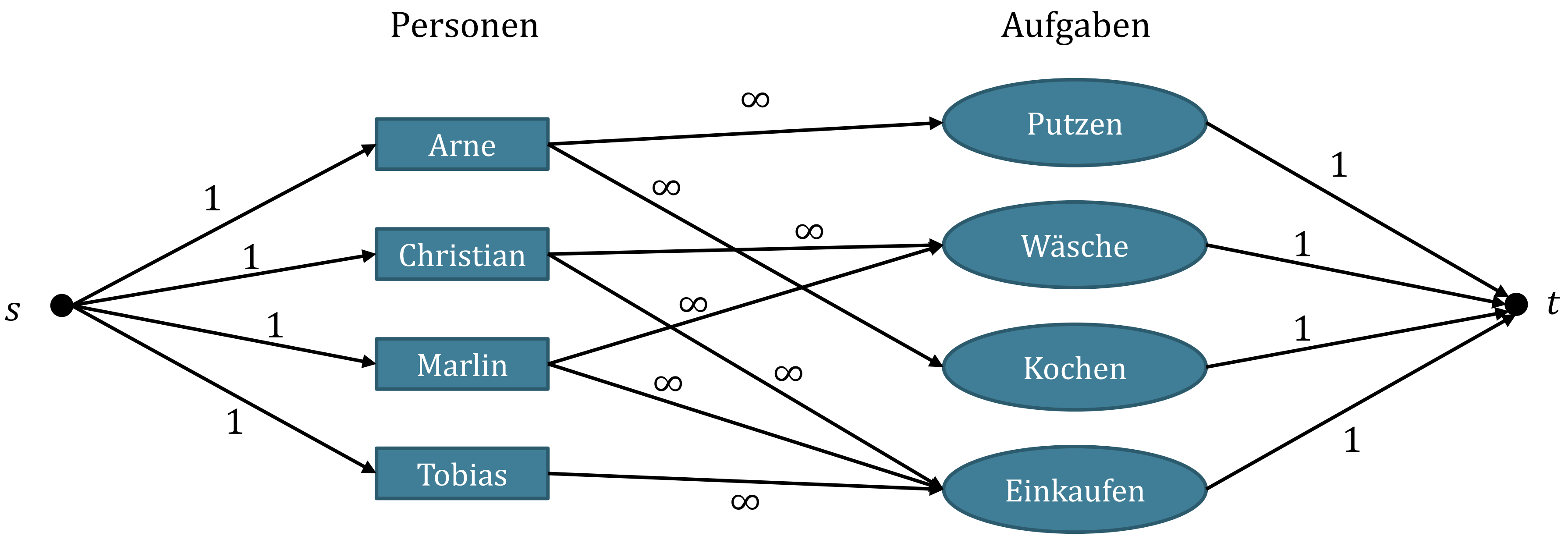
Aufgabenverteilung



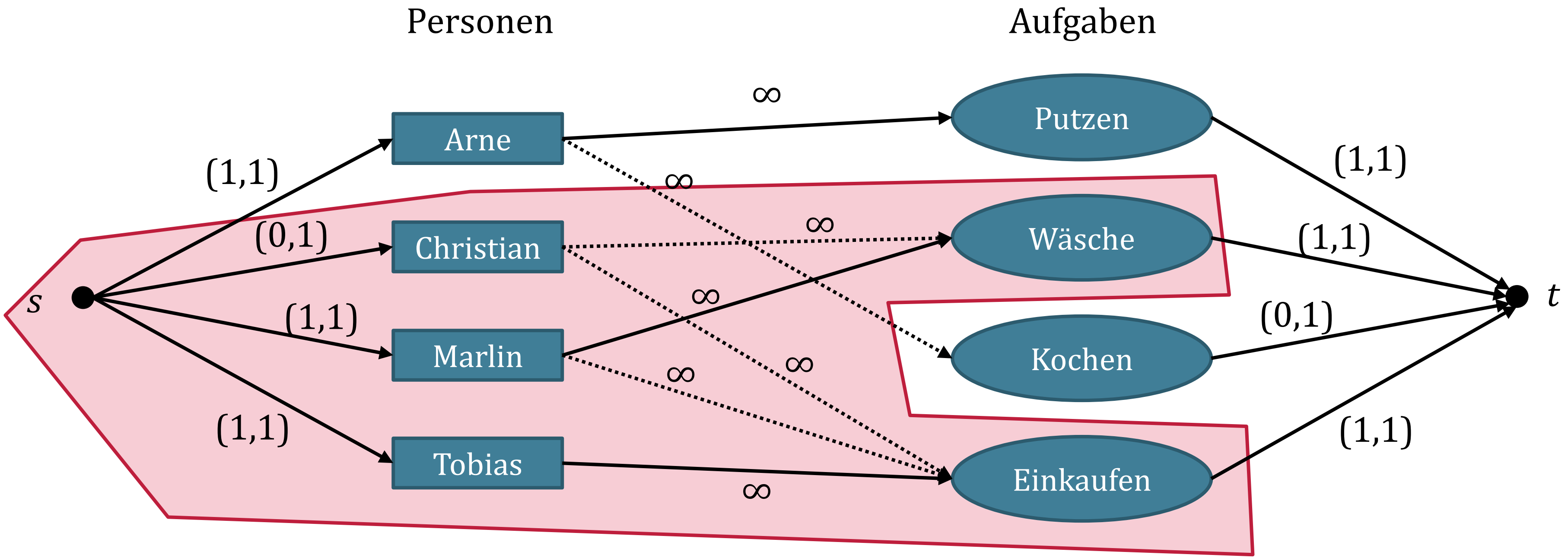
Möchten sich einer Aufgabe zuweisen.

Möchten einer Person zugewiesen werden.

Reduktion auf Flüsse

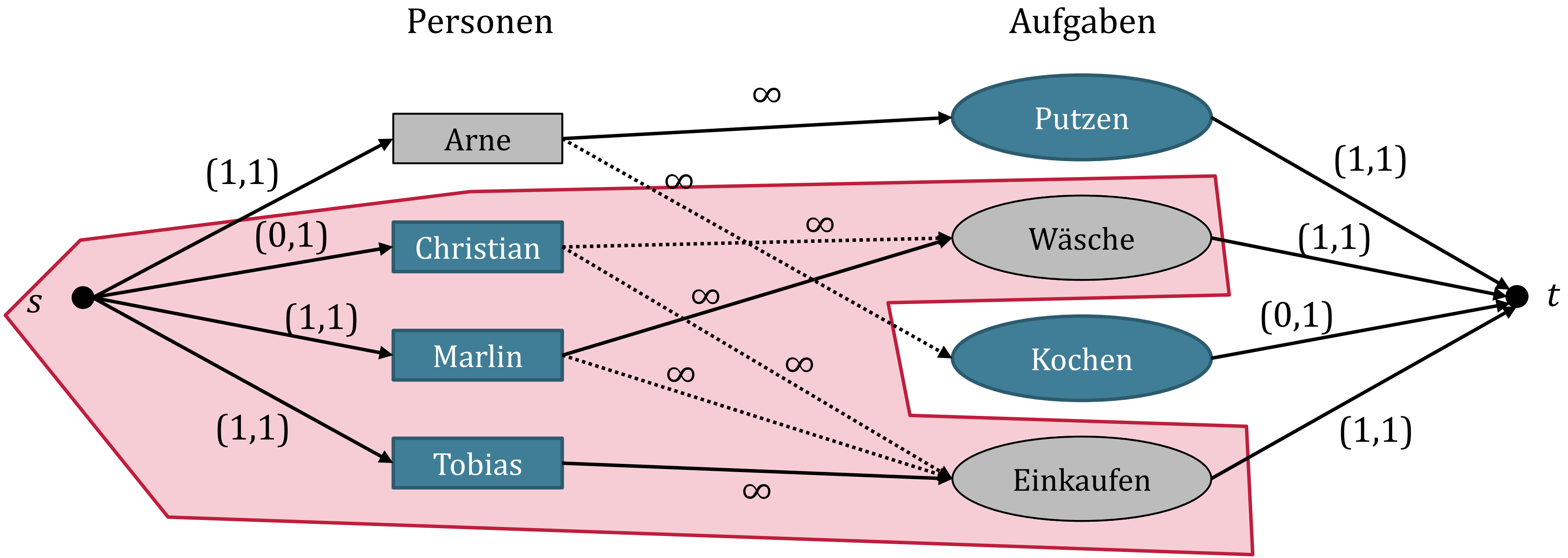


Reduktion auf Flüsse

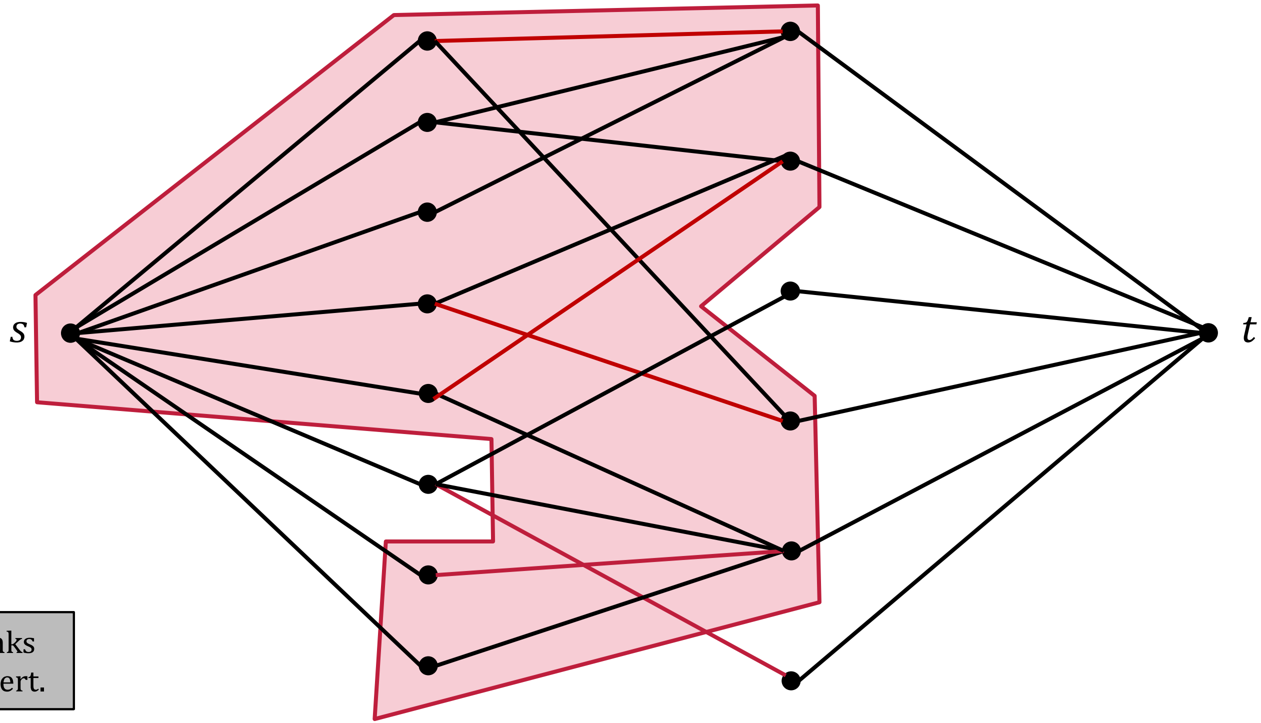


MinCut = MaxFlow = 3 ⇒ MaxMat = 3

Reduktion auf Flüsse

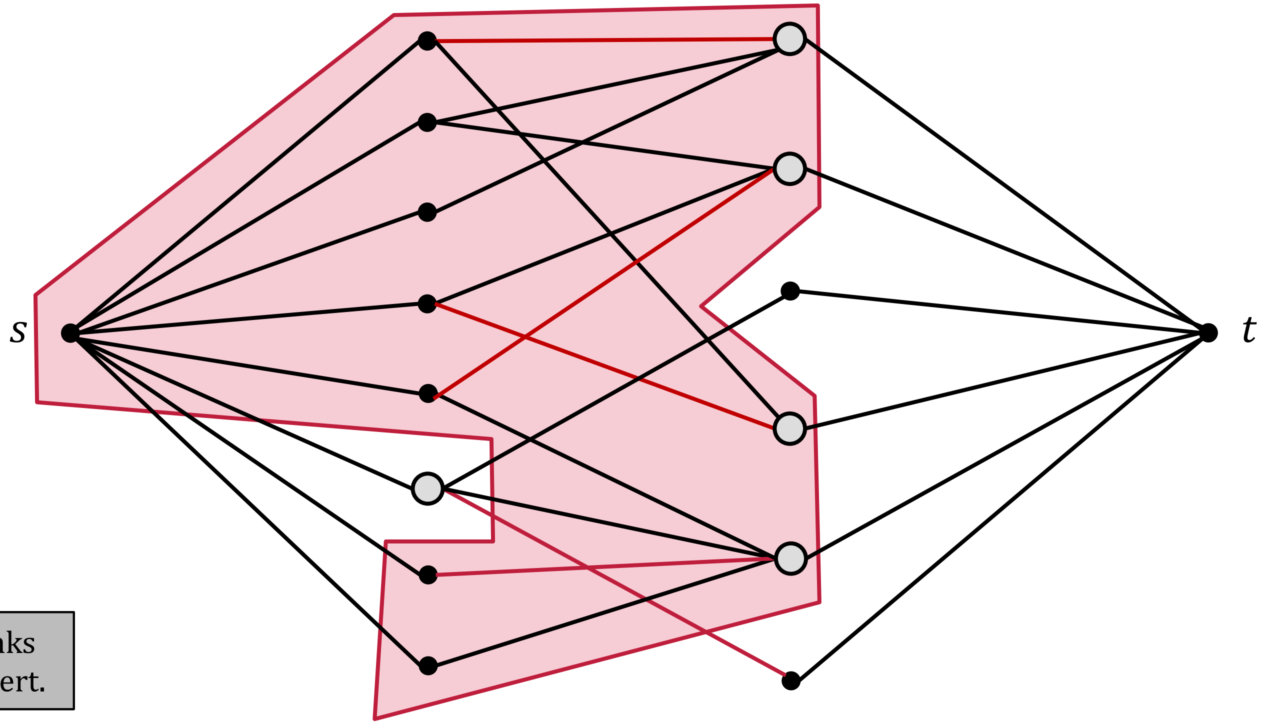


Weiteres Beispiel



Kanten sind von links nach rechts orientiert.

Weiteres Beispiel



Kanten sind von links nach rechts orientiert.

Vertex Cover

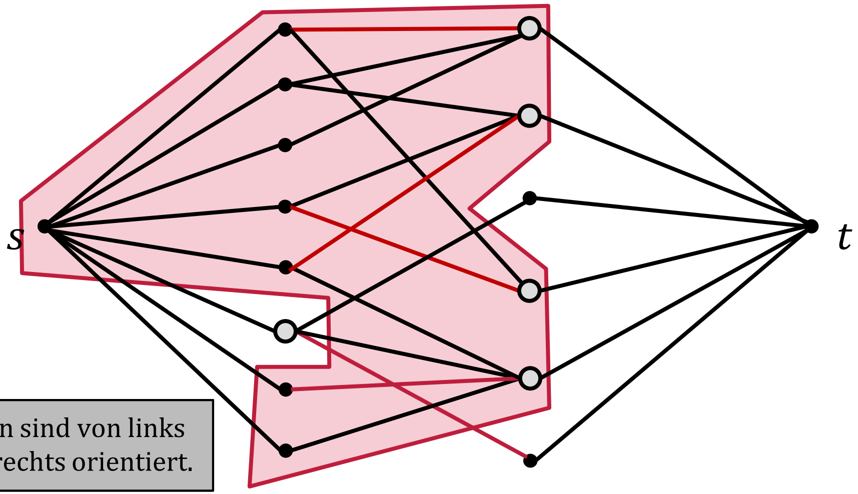
Definition 5.5 (Vertex Cover)
Sei $G = (V, E)$ ein Graph.
(1) Eine Knotenmenge $C \subseteq V$ heißt **Vertex Cover**, wenn $u \in C$ oder $v \in C$ für jede Kante $e = \{u, v\} \in E$ gilt.
(2) Ein Vertexcover heißt **(kardinalitäts-)minimal** (engl. minimum), wenn es kein Vertex Cover C' mit $|C| > |C'|$ gibt.

Lemma 5.6
Sei G ein Graph. Für jedes Matching M und jedes Vertex Cover C gilt $|M| \leq |C|$.

Satz 5.7 (Kőnig-Egerváry)
Sei G ein bipartiter Graph. Die Größe eines minimalen Vertex Covers in G entspricht der Größes eines maximalen Matchings in G.

Vertex Cover

Satz 5.7 (Kőnig-Egerváry)
Sei G ein bipartiter Graph. Die Größe eines minimalen Vertex Covers in G entspricht der Größes eines maximalen Matchings in G.



Kanten sind von links nach rechts orientiert.

Sei Q Schnittknotenmenge ohne s.
Betrachte Kanten zwischen:

- $Q \cap V_1$ und $Q \cap V_2$
- $\cancel{Q \cap V_1 \text{ und } V_2 \setminus Q}$ Ist leer!
- $V_1 \setminus Q$ und $Q \cap V_2$
- $V_1 \setminus Q$ und $V_2 \setminus Q$

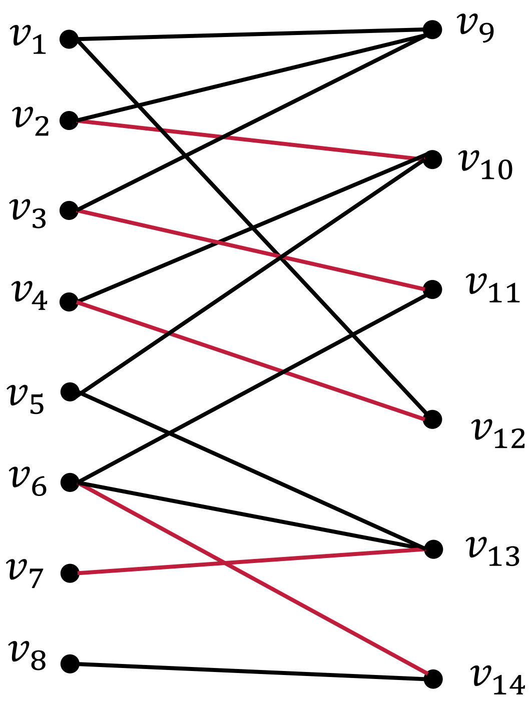
$\Rightarrow C \coloneqq (Q \setminus V_1) \cup (Q \cap V_2)$ ist ein VC.
$|C|$ = Wert des MinCuts. Also ist:
MaxMat = MaxFlow = MinCut = MinVC

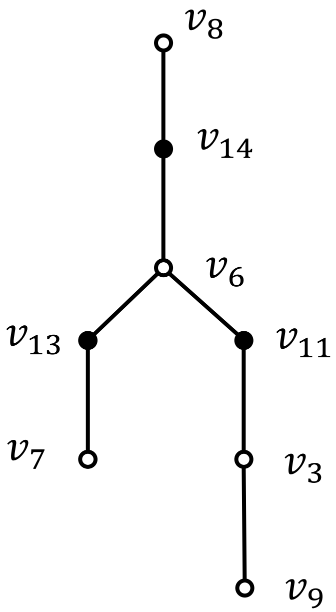
Matching für bipartite Graphen

Algorithmus 5.8
Eingabe:
  Bipartiter Graph $G = (V, E)$
Ausgabe:
  Maximales Matching M

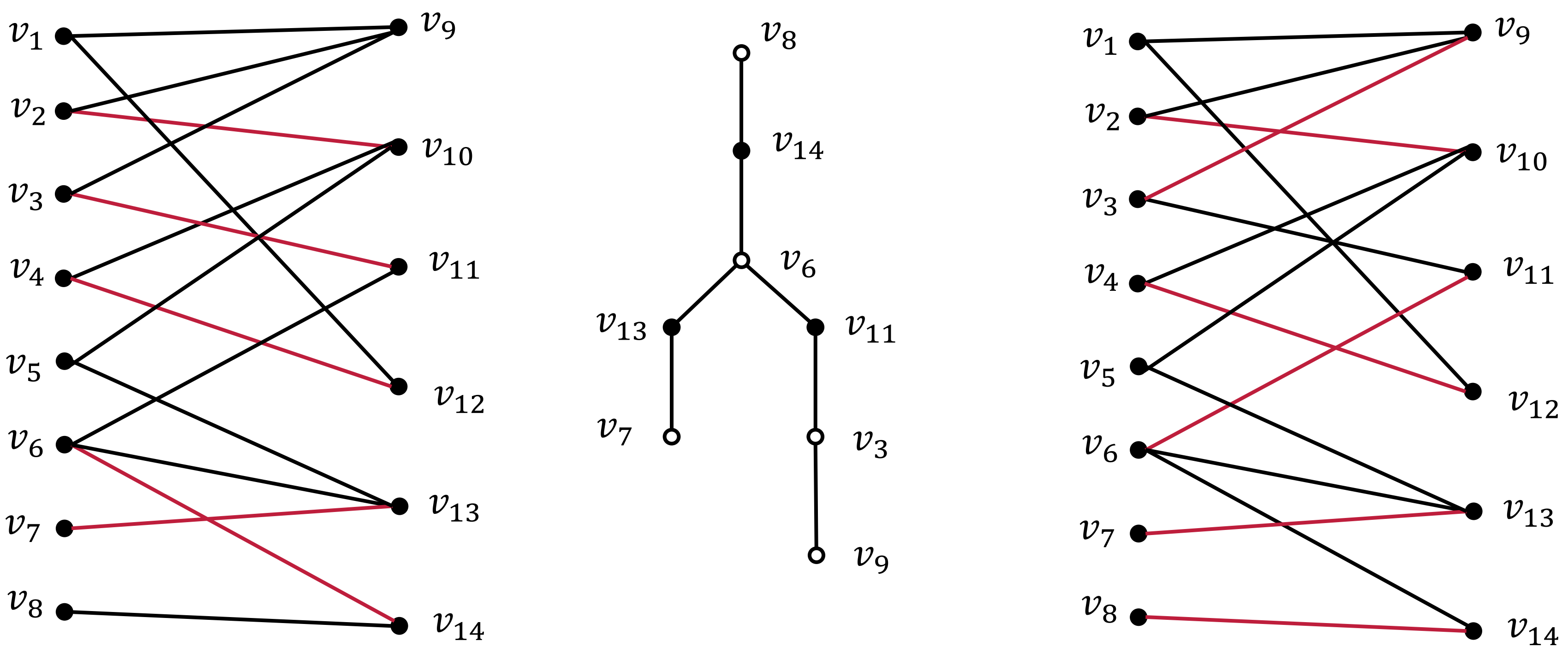
1. Function BIPARTITEMATCHING(G)
2. Set $M := \emptyset$
3. **for** $r \in V_1$ mit r ungematcht **do**
4. Setze $T := (\{r\}, \emptyset)$ und $W(T) := \{r\}$
5. **While** (es ex. Kante $\{v, w\} \in E$ mit $v \in W(T)$ und $w \notin V(T)$)
6. **If** w ist ungematcht **then**
7. Benutze w um augmentierenden Pfad zu bilden.
8. Augmentiere M
9. **If** es ex. Kein ungematchter Knoten mehr **then**
10. **Return** perfektes Matching M
11. **else**
12. Gehe zu Zeile 3.
13. **else**
14. Sei $\{w, z\}$ Matchingkante an w.
15. Füge w, z zu $V(T)$ hinzu und z zu $W(T)$
16. Füge $\{v, w\}, \{w, z\}$ zu $E(T)$ hinzu.
17. **Return** M

Beispiel


v_1
v_2
v_3
v_4
v_5
v_6
v_7
v_8
v_9
v_{10}
v_{11}
v_{12}
v_{13}
v_{14}


v_8
v_{14}
v_6
v_{13}
v_{11}
v_7
v_3
v_9

Beispiel



v_1
v_2
v_3
v_4
v_5
v_6
v_7
v_8
v_9
v_{10}
v_{11}
v_{12}
v_{13}
v_{14}
v_8
v_{14}
v_6
v_{13}
v_{11}
v_7
v_3
v_9
v_1
v_2
v_3
v_4
v_5
v_6
v_7
v_8
v_9
v_{10}
v_{11}
v_{12}
v_{13}
v_{14}

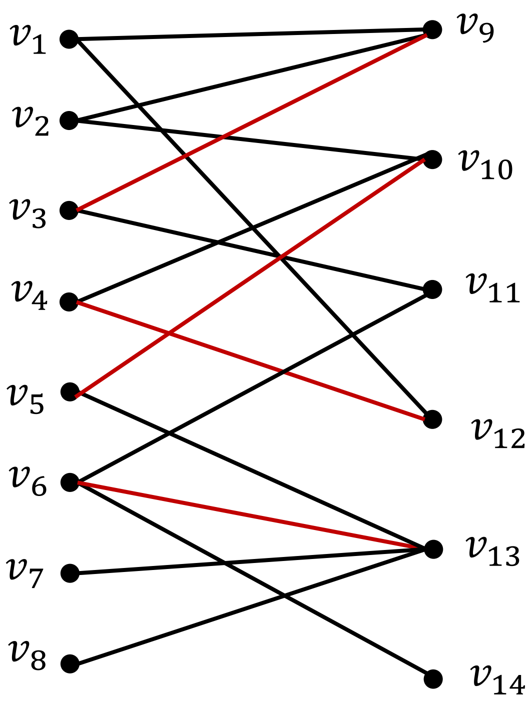
Analyse

Satz 5.9
Algorithmus 5.8 löst Problem 5.2 (Max Matching) korrekt in Zeit $O(mn)$.

Beweis:
Korrektheit klar über Reduktion auf Flüsse.

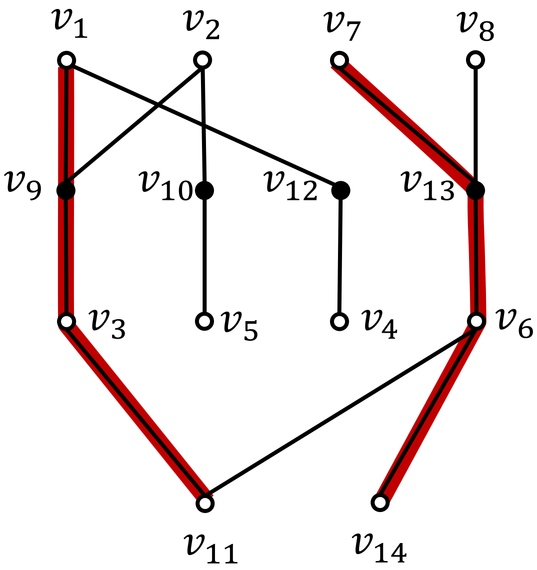
Laufzeit: Wie Ford-Fulkerson mit $f^* = MaxMat \in O(n)$

Suche über Level



Starte von allen ungematchten Knoten eine Suche wie gehabt.

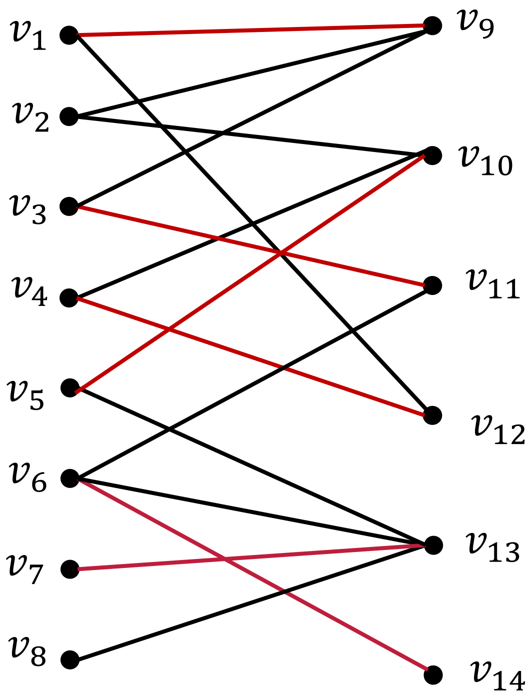
Wie BFS



Wie DFS

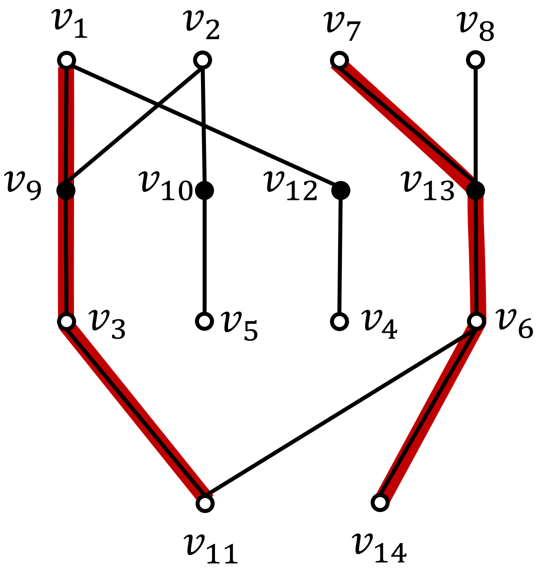
Vom Level mit ungematchten Knoten, starte Suche rückwärts.

Suche über Level



Starte von allen ungematchten Knoten eine Suche wie gehabt.

Wie BFS



Wie DFS

Vom Level mit ungematchten Knoten, starte Suche rückwärts.

Algorithmus von Hopcroft-Karp

Algorithmus 5.10

Eingabe:

Ungerichteter Graph $G=(V,E)$

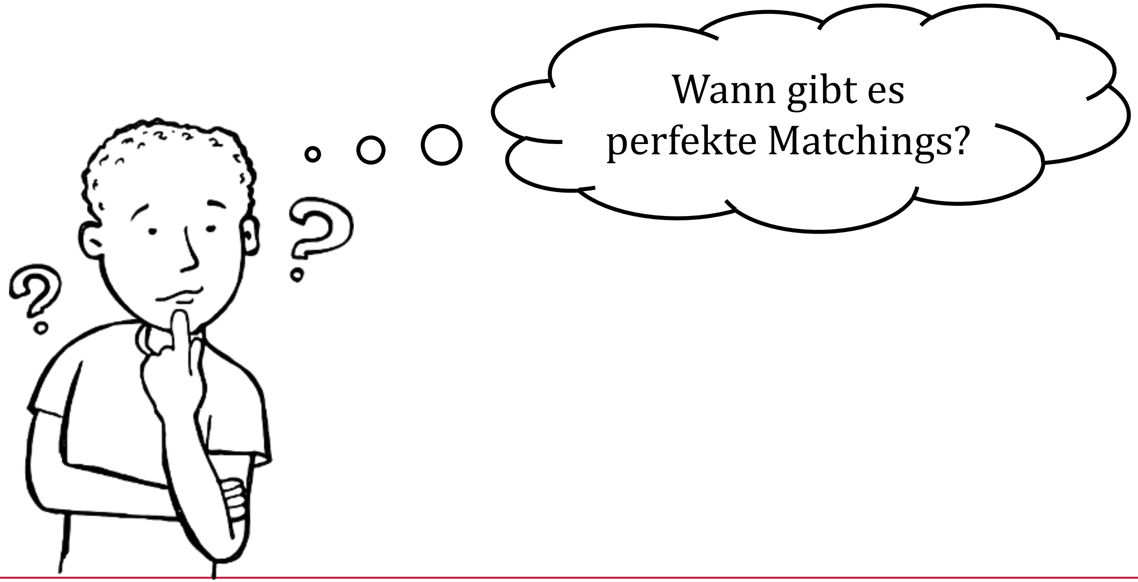
Ausgabe:

Maximales Matching M

1. Function HOPCROFTKARP(G)
2. Setze $M = \emptyset$
3. Setze $R \coloneqq \{v \in V_1 \mid v \text{ ungematcht}\}$
4. Starte von allen Knoten in R BFS-mäßig, um „Level-Graphen“ zu konstruieren
5. Stoppe bei Level, welcher ungematchte Knoten aus V_2 enthält.
6. Sei $F \subseteq V_2$ die Menge an ungematchten Knoten in diesem Level.
7. Suche über diese Level DFS-mäßig augmentierende Pfade zwischen Knoten aus R und F.
8. Augmentiere M.
9. **If** es gab eine Verbesserung und V_1 enthält ungematchte Knoten **then**
10. Gehe zu Zeile 3.
11. **Else**
12. Return M

Analyse

Satz 5.11
Algorithmus von Hopcroft-Karp löst Problem 5.2 (Max Matching) korrekt in Zeit $O(m\sqrt{n})$.



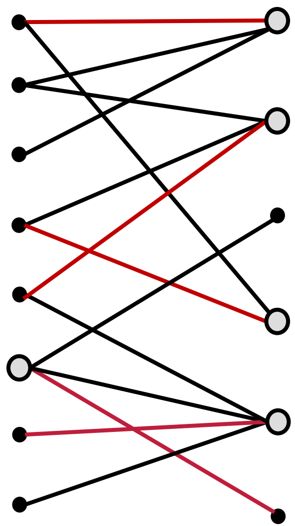
Satz von Hall

Definition 5.11 (Nachbarschaft)
Sei $G = (V, E)$ ein Graph. $N(X)$ bezeichnet die Nachbarschaft einer Menge $X \subseteq V$, also die Menge an Knoten, die adjazent zu Knoten aus X sind.

Satz 5.13
Sei $G = (V, E)$ ein bipartiter Graph mit $V = V_1 \dot{\cup} V_2$. Dann hat G genau dann ein V_1 überdeckendes Matching, wenn $|N(X)| \geq |X|$ für alle $X \subseteq V_1$ gilt.

Satz von Hall

Satz 5.13
Sei $G = (V, E)$ ein bipartiter Graph mit $V = V_1 \dot\cup V_2$. Dann hat G genau dann ein V_1 überdeckendes Matching, wenn $|N(X)| \geq |X|$ für alle $X \subseteq V_1$ gilt.



$\Rightarrow$: Trivial

$\Leftarrow$:
Annahme: Es ex. kein V_1 überdeckendes Matching.
Sei M Matching mit $|M| < |V_1|$.
Satz 5.7: es ex. U Vertex Cover mit $|U| < |V_1|$.
Wähle $U_1 := U \cap V_1$ und $U_2 := U \cap V_2$.
$\Rightarrow N(V_1 \setminus U_1) \subseteq U_2$, also:

$$|N(V_1 \setminus U_1)| \leq |U_2| < |V_1| - |U_1| = |V_1 \setminus U_1|$$

Heiratssatz von Frobenius

Satz 5.13
Sei $G = (V, E)$ ein bipartiter Graph mit $V = V_1 \dot{\cup} V_2$. Dann hat G genau dann ein V_1 überdeckendes Matching, wenn $|N(X)| \geq |X|$ für alle $X \subseteq V_1$ gilt.

Korollar 5.14
Sei $G = (V, E)$ ein bipartiter Graph mit $V = V_1 \dot{\cup} V_2$. Dann hat G genau dann ein perfektes Matching, wenn $|V_1| = |V_2|$ und $|N(X)| \geq |X|$ für alle $X \subseteq V_1$ gilt.

Nächste Woche

Was passiert bei nicht-bipartiten Graphen?

Adaptieren von Algorithmen möglich?

Welche Probleme entstehen?

Welche Schranken existieren?