Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Chair for
INFORMATICS I
Efficient Algorithms and
Knowledge-Based Systems

Institute for Informatics

Approximationsalgorithmen

Einführung
und
Vertex Cover

1. Vorlesung

Philipp Kindermann

Wintersemester 2019/20

Organisatorisches

Vorlesung: freitags, 10:15–11:45 Uhr, ÜR I
Übung: dienstags, 10:15–11:45 Uhr, SE I (ab nächste Woche!)

Organisatorisches

Vorlesung: freitags, 10:15–11:45 Uhr, ÜR I
Übung: dienstags, 10:15–11:45 Uhr, SE I (ab nächste Woche!)

Vorlesung: (hoffentlich) interaktiv & „hands-on“

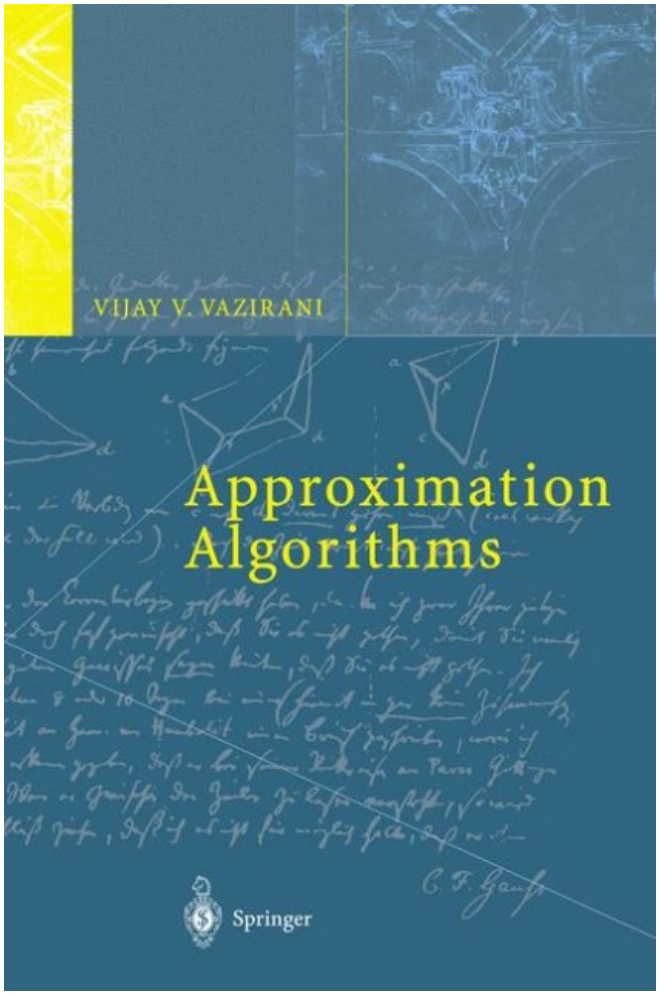
Organisatorisches

Vorlesung: freitags, 10:15–11:45 Uhr, ÜR I
Übung: dienstags, 10:15–11:45 Uhr, SE I (ab nächste Woche!)

Vorlesung: (hoffentlich) interaktiv & „hands-on“

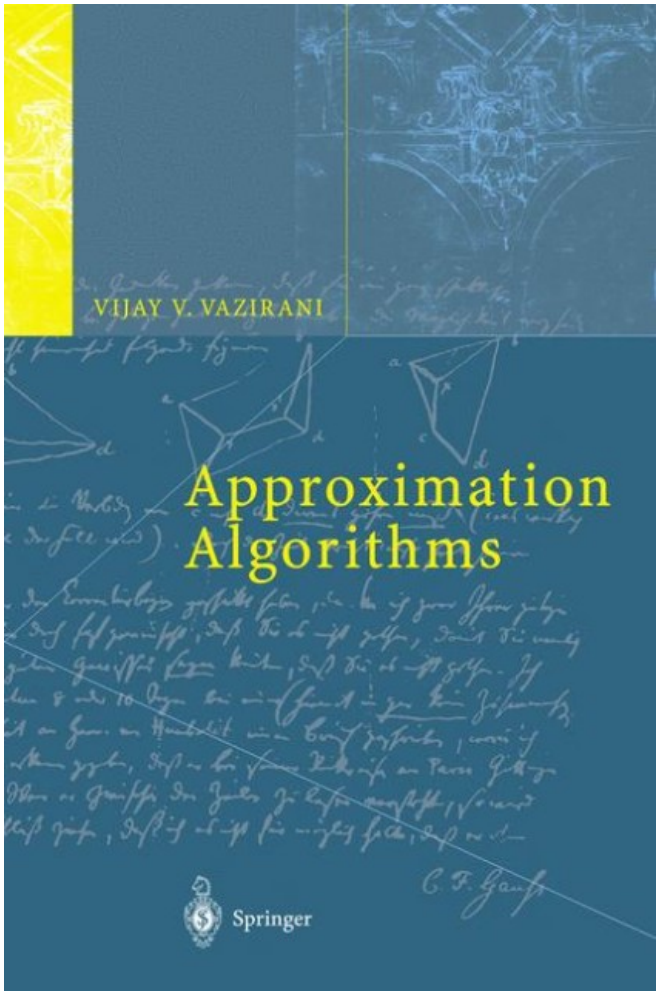
Bei Erreichen von 50% der Übungspunkte und einer bestandenen Prüfung gibt es einen Bonus in Höhe von 0,3 Notenpunkten.

Bücher zur Vorlesung

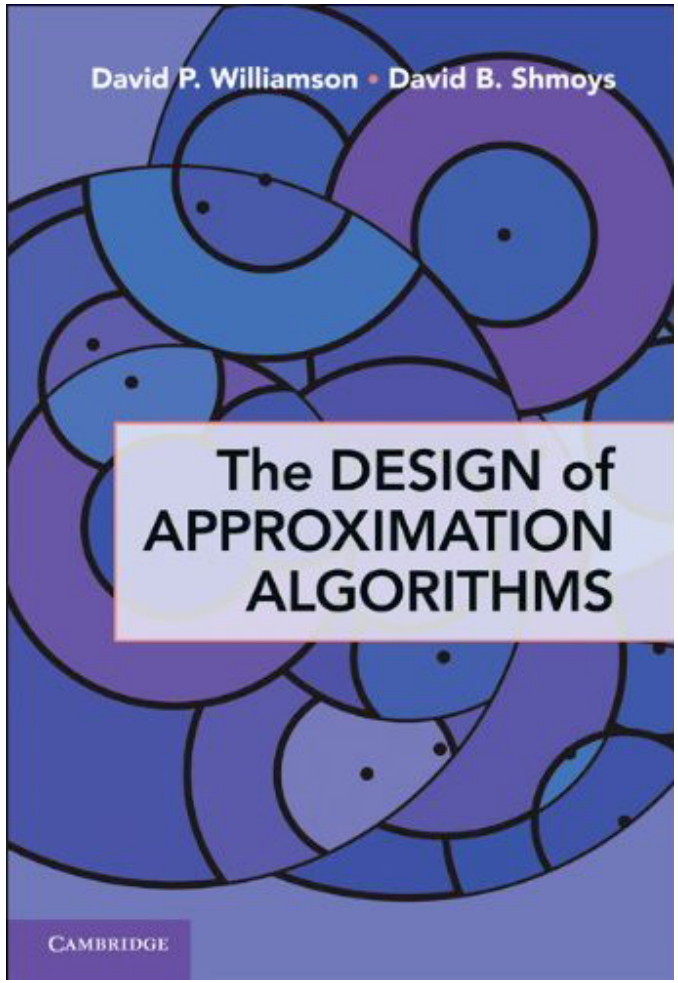


Vijay V. Vazirani:
Approximation
Algorithms
Springer-Verlag, 2003.

Bücher zur Vorlesung



Vijay V. Vazirani:
Approximation Algorithms
Springer-Verlag, 2003.



D. P. Williamson & D. B. Shmoys:
The Design of Approximation Algorithms
Cambridge-Verlag, 2011.

Approximationsalgorithmen

„All exact science is dominated by the idea of approximation.“ – Bertrand Russell

Inhalt

Kombinatorische Algorithmen

- Einführung (Vertex Cover)
- Set Cover via Greedy
- Shortest Superstring via Reduktion auf SC
- Steinerbaum via MST
- Mehrwegeschnitt via Greedy
- k-Zentrum via param. Pruning
- Min-Deg-Spanning-Tree & lokale Suche
- Rucksackproblem via DP & Skalierung
- Euklidisches TSP via Quadtrees

LP-basierte Algorithmen

- Einführung in LP-Dualität
- Set Cover via LP-Runden
- Set Cover via Primal-Dual-Schema
- Maximum Satisfiability
- Scheduling und Extrempunktlösungen
- Steinerwald via Primal-Dual

Approximationsalgorithmen

- Viele Optimierungsprobleme sind NP-schwer (Paradebeispiel: Problem des Handlungsreisenden).

Approximationsalgorithmen

- Viele Optimierungsprobleme sind NP-schwer (Paradebeispiel: Problem des Handlungsreisenden).
- Also können optimale Lösungen nicht effizient bestimmt werden (außer P=NP).

Approximationsalgorithmen

- Viele Optimierungsprobleme sind NP-schwer (Paradebeispiel: Problem des Handlungsreisenden).
- Also können optimale Lösungen nicht effizient bestimmt werden (außer P=NP).
- Gute Näherungslösungen lassen sich jedoch häufig effizient ermitteln!

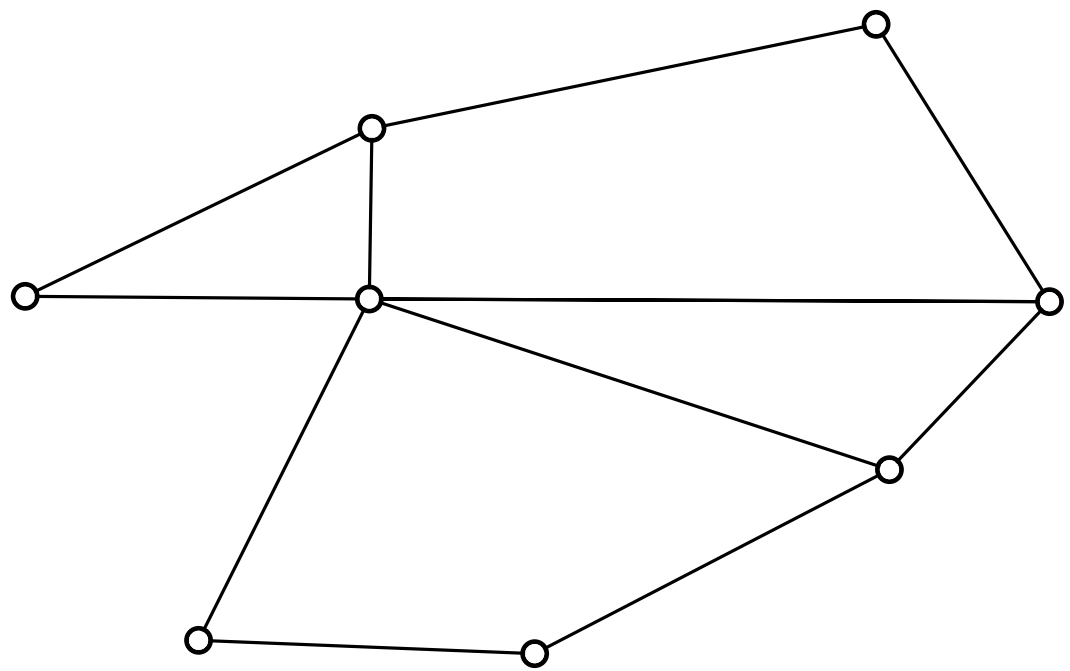
Approximationsalgorithmen

- Viele Optimierungsprobleme sind NP-schwer (Paradebeispiel: Problem des Handlungsreisenden).
- Also können optimale Lösungen nicht effizient bestimmt werden (außer P=NP).
- Gute Näherungslösungen lassen sich jedoch häufig effizient ermitteln!
- Wir studieren Techniken für Entwurf und Analyse von Approximationsalgorithmen anhand konkreter Optimierungsprobleme.

Vertex Cover (card.)

Geg. Graph $G = (V, E)$

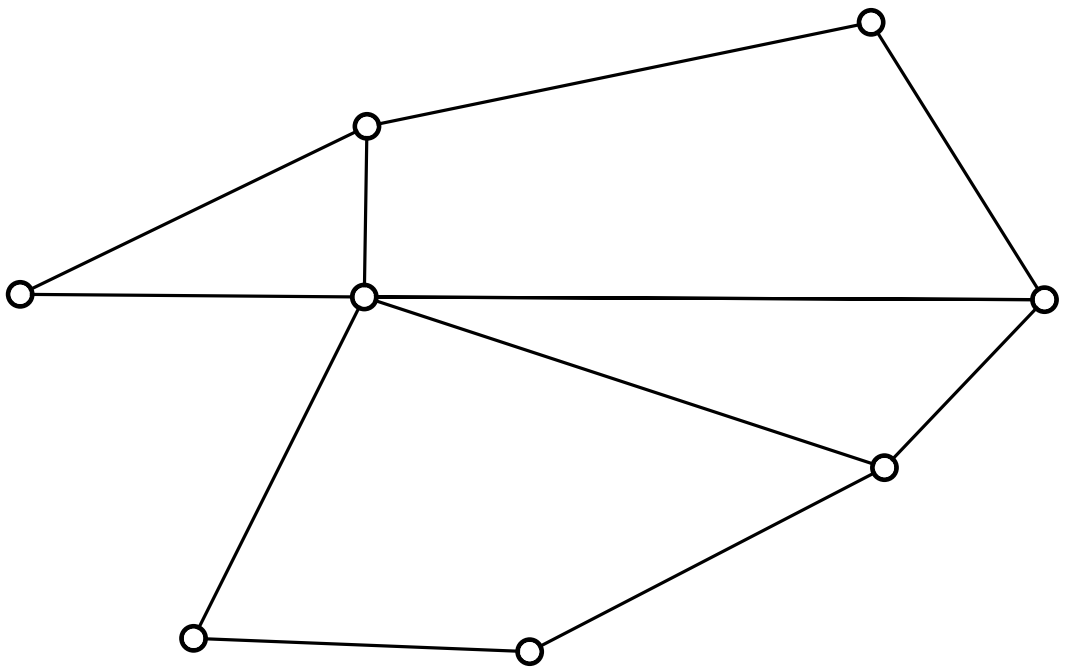
Ges.



Vertex Cover (card.)

Geg. Graph $G = (V, E)$

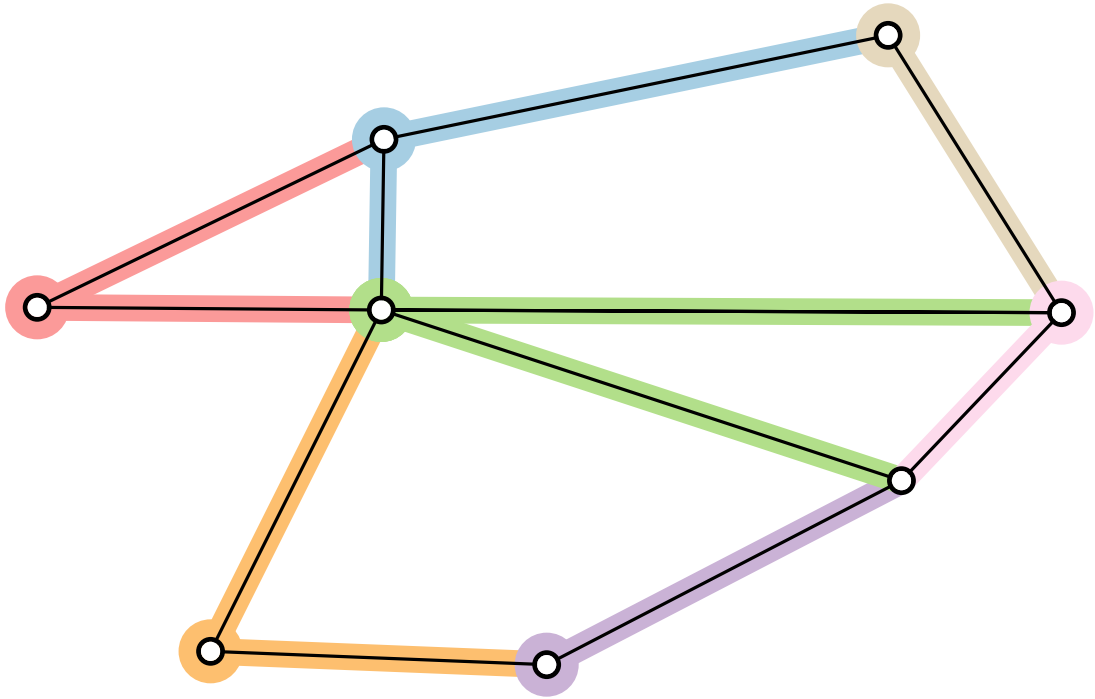
Ges. eine kleinste **Knotenüberdeckung**, d.h. eine kleinste Knotenmenge $V' \subseteq V$, die alle Kanten **überdeckt** (für alle Kanten $uv \in E$ gilt $u \in V'$ oder $v \in V'$)



Vertex Cover (card.)

Geg. Graph $G = (V, E)$

Ges. eine kleinste **Knotenüberdeckung**, d.h. eine kleinste Knotenmenge $V' \subseteq V$, die alle Kanten **überdeckt** (für alle Kanten $uv \in E$ gilt $u \in V'$ oder $v \in V'$)

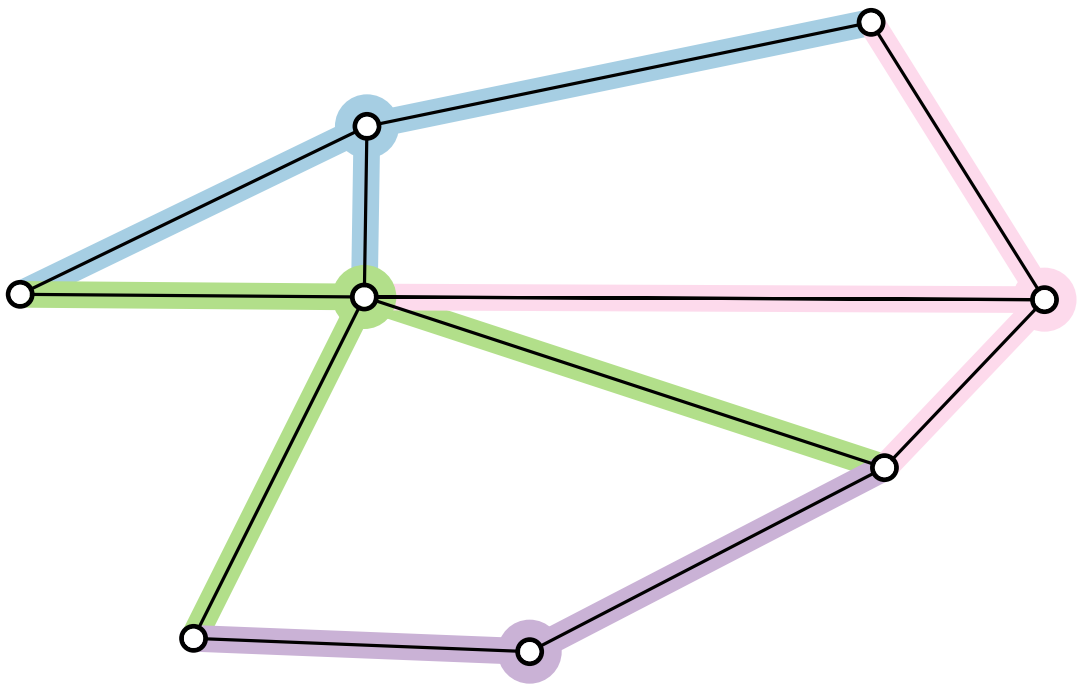


(irgend-) eine Knotenüberdeckung

Vertex Cover (card.)

Geg. Graph $G = (V, E)$

Ges. eine kleinste **Knotenüberdeckung**, d.h. eine kleinste Knotenmenge $V' \subseteq V$, die alle Kanten **überdeckt** (für alle Kanten $uv \in E$ gilt $u \in V'$ oder $v \in V'$)

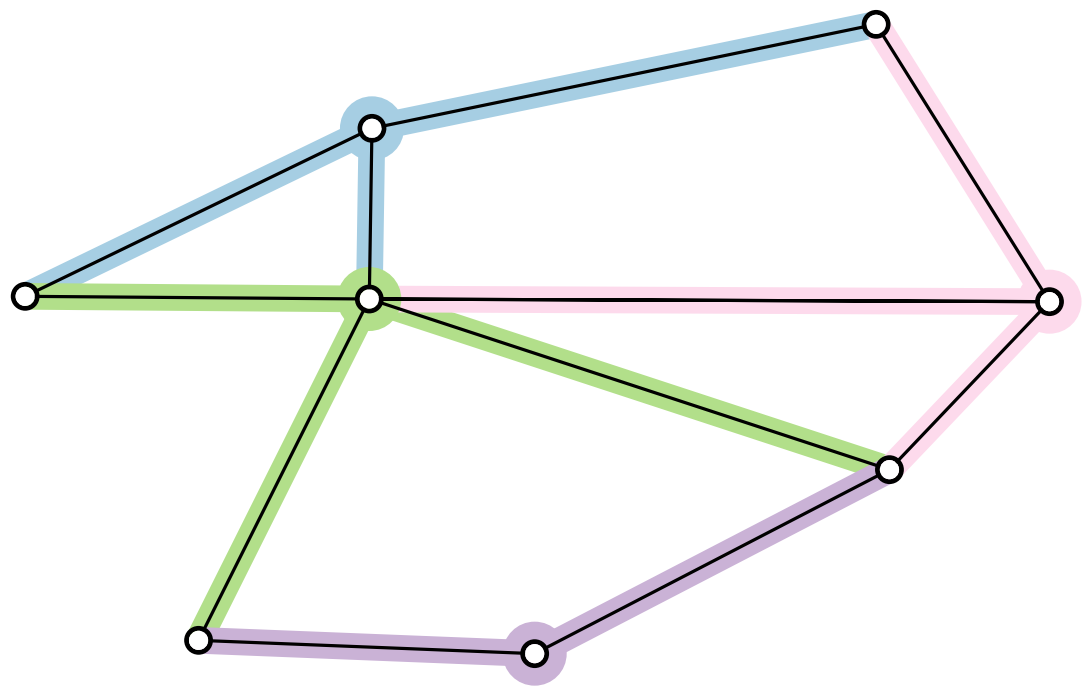


Optimum (OPT $= 4$)

Vertex Cover (card.)

Geg. Graph $G = (V, E)$

Ges. eine kleinste **Knotenüberdeckung**, d.h. eine kleinste Knotenmenge $V' \subseteq V$, die alle Kanten **überdeckt** (für alle Kanten $uv \in E$ gilt $u \in V'$ oder $v \in V'$)

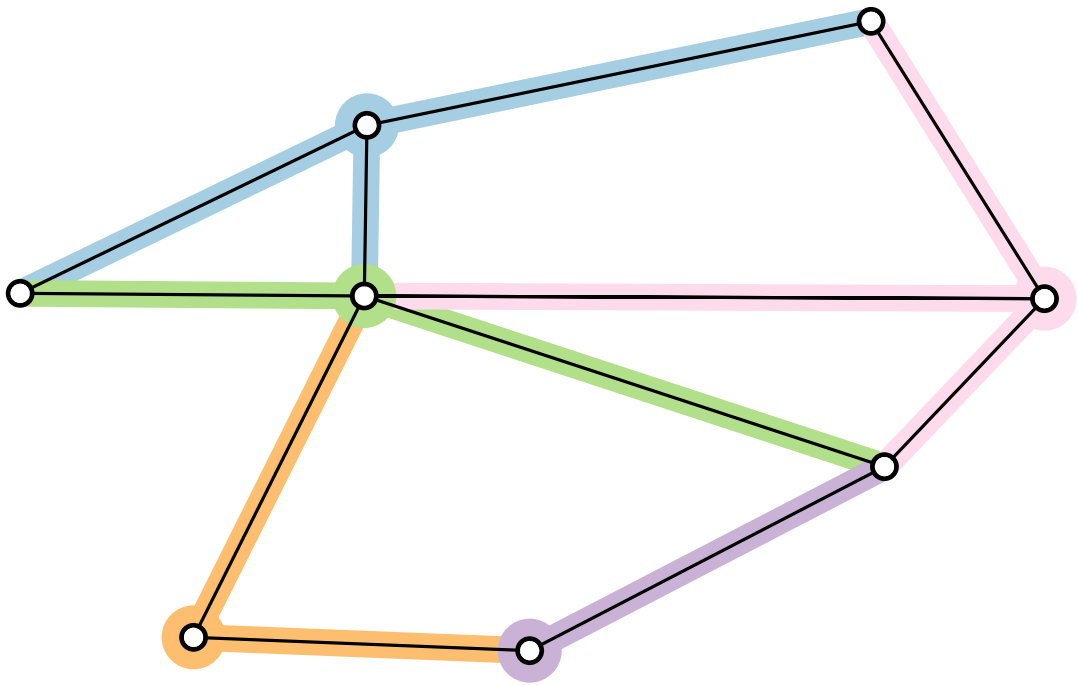


Optimum (OPT $= 4$) – aber i.A. NP-schwer zu berechnen :-(

Vertex Cover (card.)

Geg. Graph $G = (V, E)$

Ges. eine kleinste **Knotenüberdeckung**, d.h. eine kleinste Knotenmenge $V' \subseteq V$, die alle Kanten **überdeckt** (für alle Kanten $uv \in E$ gilt $u \in V'$ oder $v \in V'$)



„gute“ Näherungslösung (5/4-Approximation)

NP-Optimierungsproblem

Ein **NP-Optimierungsproblem** Π ist gegeben durch

NP-Optimierungsproblem

Ein **NP-Optimierungsproblem** Π ist gegeben durch

- eine Menge D_Π von **Instanzen** (die Größe einer Instanz $I \in D_\Pi$ bezeichnen wir mit $|I|$),

NP-Optimierungsproblem

Ein **NP-Optimierungsproblem** Π ist gegeben durch

- eine Menge D_Π von **Instanzen** (die Größe einer Instanz $I \in D_\Pi$ bezeichnen wir mit $|I|$),
- für jede Instanz $I \in D_\Pi$ eine Menge $S_\Pi(I) \neq \emptyset$ **zulässiger Lösungen** für I, so dass

NP-Optimierungsproblem

Ein **NP-Optimierungsproblem** Π ist gegeben durch

- eine Menge D_Π von **Instanzen** (die Größe einer Instanz $I \in D_\Pi$ bezeichnen wir mit $|I|$),
- für jede Instanz $I \in D_\Pi$ eine Menge $S_\Pi(I) \neq \emptyset$ **zulässiger Lösungen** für I, so dass
  - für jedes $s \in S_\Pi(I)$ ist $|s|$ polynomiell beschränkt in $|I|$ und

NP-Optimierungsproblem

Ein **NP-Optimierungsproblem** Π ist gegeben durch

- eine Menge D_Π von **Instanzen** (die Größe einer Instanz $I \in D_\Pi$ bezeichnen wir mit $|I|$),
- für jede Instanz $I \in D_\Pi$ eine Menge $S_\Pi(I) \neq \emptyset$ **zulässiger Lösungen** für I, so dass
  - für jedes $s \in S_\Pi(I)$ ist $|s|$ polynomiell beschränkt in $|I|$ und
  - für jedes Paar (s, I) lässt sich effizient entscheiden, ob $s \in S_\Pi(I)$,

NP-Optimierungsproblem

Ein **NP-Optimierungsproblem** Π ist gegeben durch

- eine Menge D_Π von **Instanzen** (die Größe einer Instanz $I \in D_\Pi$ bezeichnen wir mit $|I|$),
- für jede Instanz $I \in D_\Pi$ eine Menge $S_\Pi(I) \neq \emptyset$ **zulässiger Lösungen** für I, so dass
  - für jedes $s \in S_\Pi(I)$ ist $|s|$ polynomiell beschränkt in $|I|$ und
  - für jedes Paar (s, I) lässt sich effizient entscheiden, ob $s \in S_\Pi(I)$,
- eine effizient berechenbare **Zielfunktion** obj_Π, die jedem (s, I) mit $s \in S_\Pi(I)$ einen Wert $\mathrm{obj}_\Pi(I, s) \geq 0$ zuordnet.

NP-Optimierungsproblem

Ein **NP-Optimierungsproblem** Π ist gegeben durch

- eine Menge D_Π von **Instanzen** (die Größe einer Instanz $I \in D_\Pi$ bezeichnen wir mit $|I|$),
- für jede Instanz $I \in D_\Pi$ eine Menge $S_\Pi(I) \neq \emptyset$ **zulässiger Lösungen** für I, so dass
  - für jedes $s \in S_\Pi(I)$ ist $|s|$ polynomiell beschränkt in $|I|$ und
  - für jedes Paar (s, I) lässt sich effizient entscheiden, ob $s \in S_\Pi(I)$,
- eine effizient berechenbare **Zielfunktion** obj_Π, die jedem (s, I) mit $s \in S_\Pi(I)$ einen Wert $\mathrm{obj}_\Pi(I, s) \geq 0$ zuordnet.
- Π ist Minimierungs- oder Maximimierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$

Für $I \in D_\Pi$ ist $|I| =$

$S_\Pi(I) =$

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?

- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$?

$\text{obj}_\Pi(I, s) =$

Π is M.....imierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$ Menge aller Graphen

Für $I \in D_\Pi$ ist $|I| =$

$S_\Pi(I) =$

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?

- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$?

$\text{obj}_\Pi(I, s) =$

Π is M.....imierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$ Menge aller Graphen

Für $I \in D_\Pi$ ist $|I| =$

$G{=}(V, E)$ $S_\Pi(I) =$

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?

- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$?

$\text{obj}_\Pi(I, s) =$

Π is M.....imierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$ Menge aller Graphen

Für $I \in D_\Pi$ ist $|I| =$ Anzahl der Knoten $|V|$

$G{=}(V, E)$ $S_\Pi(I) =$

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?

- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$?

$\text{obj}_\Pi(I, s) =$

Π is M.....imierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$ Menge aller Graphen

Für $I \in D_\Pi$ ist $|I| =$ Anzahl der Knoten $|V|$

$G{=}(V, E)$ $S_\Pi(I) =$ Menge aller Vertex Cover von G

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?

- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$?

$\text{obj}_\Pi(I, s) =$

Π is M.....imierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$ Menge aller Graphen

Für $I \in D_\Pi$ ist $\quad |I| =$ Anzahl der Knoten $|V|$

$G{=}(V, E)$ $\quad S_\Pi(I) =$ Menge aller Vertex Cover von G

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?

  $s \subseteq V \Rightarrow |s| \leq |V| = |I|$

- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$?

$\text{obj}_\Pi(I, s) =$

Π is M.....imierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$ Menge aller Graphen

Für $I \in D_\Pi$ ist $\quad |I| =$ Anzahl der Knoten $|V|$

$G{=}(V, E)$ $\quad S_\Pi(I) =$ Menge aller Vertex Cover von G

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?
  $s \subseteq V \Rightarrow |s| \leq |V| = |I|$
- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$? Überprüfe, ob alle Kanten überdeckt sind.

$\text{obj}_\Pi(I, s) =$

Π is M.....imierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$ Menge aller Graphen

Für $I \in D_\Pi$ ist $\quad |I| =$ Anzahl der Knoten $|V|$

$G{=}(V, E)$ $\quad S_\Pi(I) =$ Menge aller Vertex Cover von G

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?
  $s \subseteq V \Rightarrow |s| \leq |V| = |I|$
- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$? Überprüfe, ob alle Kanten überdeckt sind.

$\text{obj}_\Pi(I, s) = |s|$

Π is M.....imierungsproblem.

Vertex Cover als NP-Optimierungsprob.

Aufgabe: Bestimme die Werte für $\Pi =$ Vertex Cover.

$D_\Pi =$ Menge aller Graphen

Für $I \in D_\Pi$ ist $\quad |I| =$ Anzahl der Knoten $|V|$

$G{=}(V, E)$ $\quad S_\Pi(I) =$ Menge aller Vertex Cover von G

- Warum gilt für jedes $s \in S_\Pi(I)$, dass $|s| \in \text{poly}(|I|)$?
  $s \subseteq V \Rightarrow |s| \leq |V| = |I|$
- Wie lässt sich für ein Paar (s, I) effizient entscheiden, ob $s \in S_\Pi(I)$? Überprüfe, ob alle Kanten überdeckt sind.

$\text{obj}_\Pi(I, s) = |s|$

Π is M.in.imierungsproblem.

Optimum, optimaler Zielfunktionswert

Sei Π ein Minimierungsproblem und $I \in D_\Pi$ eine Instanz für Π.

Optimum, optimaler Zielfunktionswert

Sei Π ein Minimierungsproblem und $I \in D_\Pi$ eine Instanz für Π.
Eine zulässige Lösung $s^* \in S_\Pi(I)$ heißt **Optimum**, wenn $\mathrm{obj}_\Pi(I, s^*)$ minimal unter allen zulässigen Lösungen für I ist.

Optimum, optimaler Zielfunktionswert

Maximierungsproblem

Sei Π ein Minimierungsproblem und $I \in D_\Pi$ eine Instanz für Π.
Eine zulässige Lösung $s^* \in S_\Pi(I)$ heißt **Optimum**, wenn $\mathrm{obj}_\Pi(I, s^*)$ minimal unter allen zulässigen Lösungen für I ist.

maximal

Optimum, optimaler Zielfunktionswert

Maximierungsproblem

Sei Π ein Minimierungsproblem und $I \in D_\Pi$ eine Instanz für Π.
Eine zulässige Lösung $s^* \in S_\Pi(I)$ heißt **Optimum**, wenn $\mathrm{obj}_\Pi(I, s^*)$ minimal unter allen zulässigen Lösungen für I ist. maximal

Der optimale Zielfunktionswert $\mathrm{obj}_\Pi(I, s^*)$ wird auch mit $\mathrm{OPT}_\Pi(I)$ oder kurz OPT bezeichnet.

Approximationsalgorithmus

Sei Π ein Minimierungsproblem und $\alpha \in \mathbb{Q}^+$. Ein Faktor-α-Approximationsalgorithmus für Π ist ein effizienter Algorithmus, der für **jede** Instanz $I \in D_\Pi$ eine Lösung $s \in S_\Pi(I)$ berechnet, so dass

Approximationsalgorithmus

Sei Π ein Minimierungsproblem und $\alpha \in \mathbb{Q}^+$. Ein Faktor-α-Approximationsalgorithmus für Π ist ein effizienter Algorithmus, der für **jede** Instanz $I \in D_\Pi$ eine Lösung $s \in S_\Pi(I)$ berechnet, so dass

$$\frac{\mathrm{obj}_\Pi(I, s)}{\mathrm{OPT}_\Pi(I)}$$

Approximationsalgorithmus

Sei Π ein Minimierungsproblem und $\alpha \in \mathbb{Q}^+$. Ein Faktor-α-Approximationsalgorithmus für Π ist ein effizienter Algorithmus, der für **jede** Instanz $I \in D_\Pi$ eine Lösung $s \in S_\Pi(I)$ berechnet, so dass

$$\frac{\mathrm{obj}_\Pi(I, s)}{\mathrm{OPT}_\Pi(I)} \leq \alpha.$$

Approximationsalgorithmus

Sei Π ein Minimierungsproblem und $\alpha : \mathbb{N} \to \mathbb{Q}$ ~~$\alpha \in \mathbb{Q}^+$~~. Ein Faktor-α-Approximationsalgorithmus für Π ist ein effizienter Algorithmus, der für **jede** Instanz $I \in D_\Pi$ eine Lösung $s \in S_\Pi(I)$ berechnet, so dass

$$\frac{\mathrm{obj}_\Pi(I, s)}{\mathrm{OPT}_\Pi(I)} \leq \cancel{\alpha}.\ \alpha(|I|)$$

Approximationsalgorithmus

Maximierungsproblem $\alpha : \mathbb{N} \to \mathbb{Q}$

Sei Π ein Minimierungsproblem und $\cancel{\alpha \in \mathbb{Q}^+}$. Ein Faktor-$\alpha$-Approximationsalgorithmus für Π ist ein effizienter Algorithmus, der für **jede** Instanz $I \in D_\Pi$ eine Lösung $s \in S_\Pi(I)$ berechnet, so dass

$$\frac{\mathrm{obj}_\Pi(I, s)}{\mathrm{OPT}_\Pi(I)} \overset{\geq}{\underset{\leq}{}} \cancel{\alpha}.\ \alpha(|I|)$$

Approximationsalg. für Vertex Cover

Ideen?

Approximationsalg. für Vertex Cover

Ideen?

- Kanten-Greedy
- Knoten-Greedy (siehe Übung)
- nicht verkleinerbare Knotenüberdeckungen

Approximationsalg. für Vertex Cover

Ideen?

- Kanten-Greedy
- Knoten-Greedy (siehe Übung)
- nicht verkleinerbare Knotenüberdeckungen

Abschätzung?

Approximationsalg. für Vertex Cover

Ideen?

- Kanten-Greedy
- Knoten-Greedy (siehe Übung)
- nicht verkleinerbare Knotenüberdeckungen

Abschätzung?

Problem: Wie können wir $\mathrm{obj}_\Pi(I, s)/\mathrm{OPT}$ abschätzen – wenn OPT schwer zu berechnen ist?

Approximationsalg. für Vertex Cover

Ideen?

- Kanten-Greedy
- Knoten-Greedy (siehe Übung)
- nicht verkleinerbare Knotenüberdeckungen

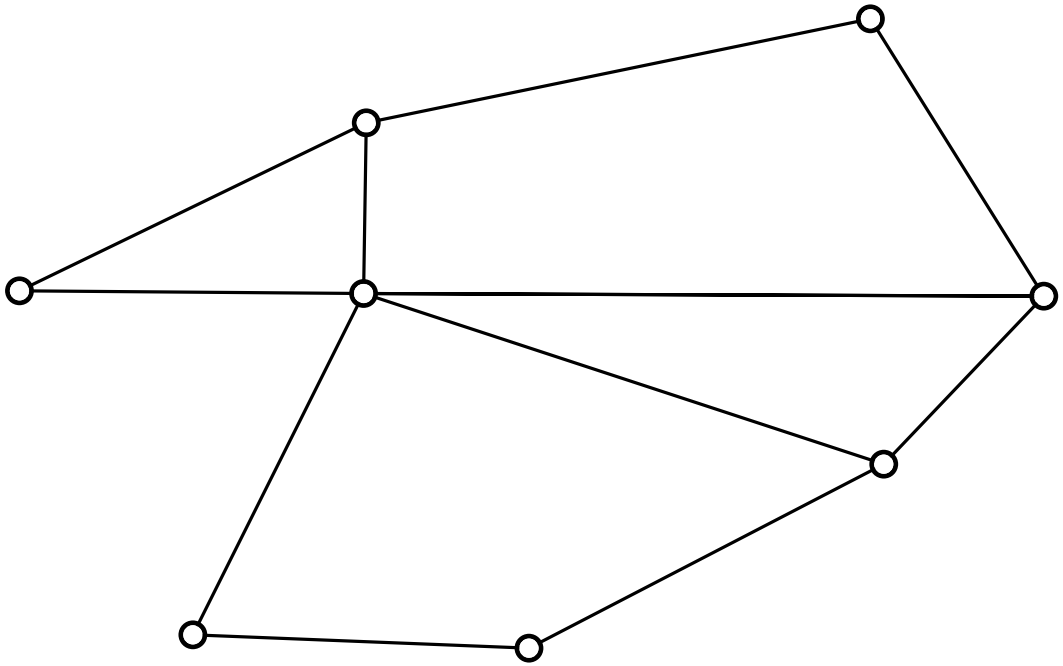
Abschätzung?

Problem: Wie können wir $\mathrm{obj}_\Pi(I, s)/\mathrm{OPT}$ abschätzen – wenn OPT schwer zu berechnen ist?

Vorgehen: Finde „gute“ untere Schranke $U \leq \mathrm{OPT}$ für OPT und vergleiche diese mit der Approximation.

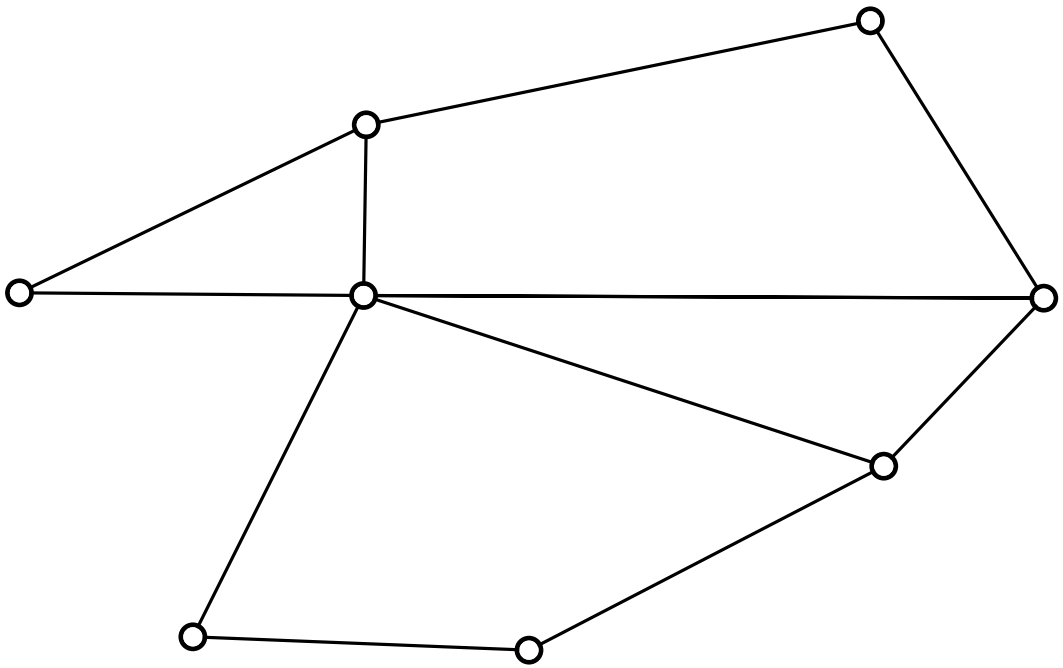
$$\frac{\mathrm{obj}_\Pi(I, s)}{\mathrm{OPT}} \leq \frac{\mathrm{obj}_\Pi(I, s)}{U}$$

Untere Schranke durch Matchings



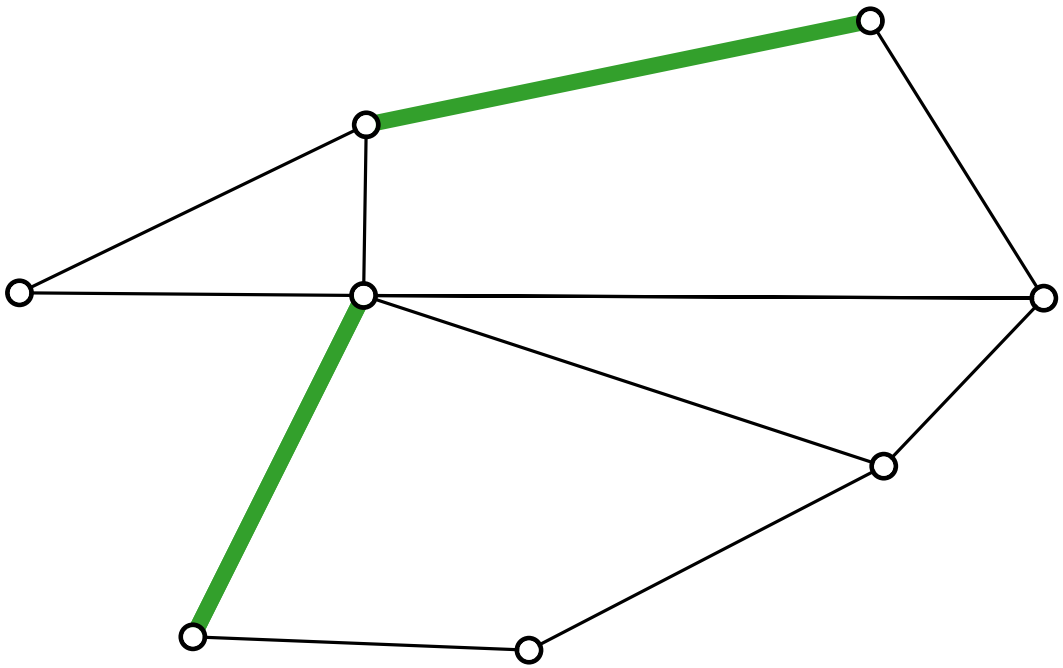
Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.



Untere Schranke durch Matchings

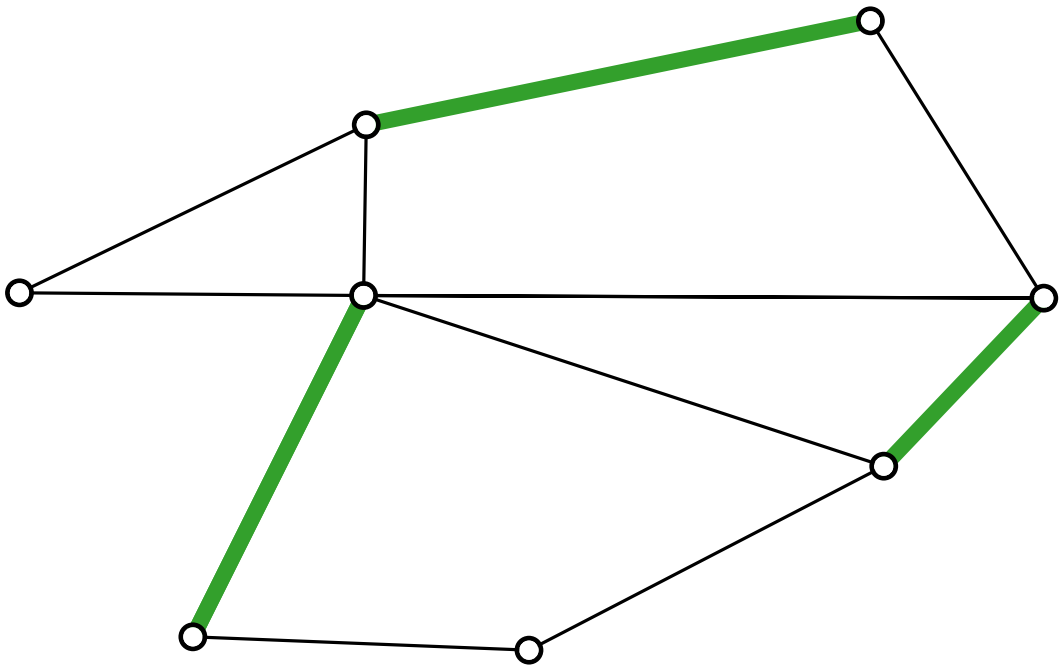
Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.



Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.

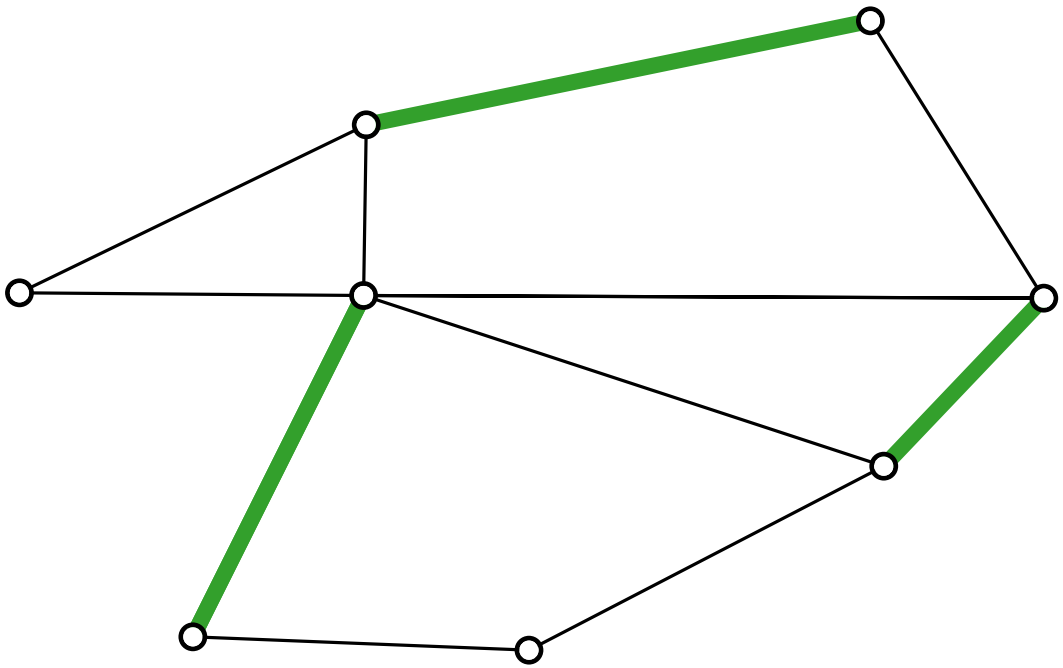


Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.

OPT $\geq$



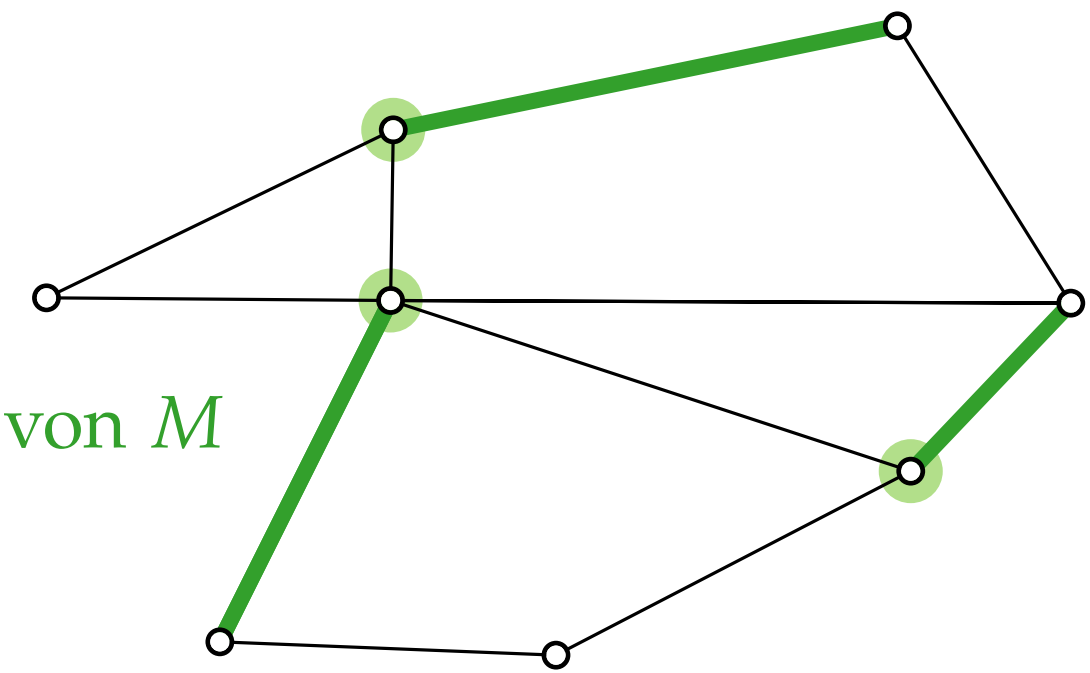
Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.

OPT $\geq$

Knotenüberdeckung von M



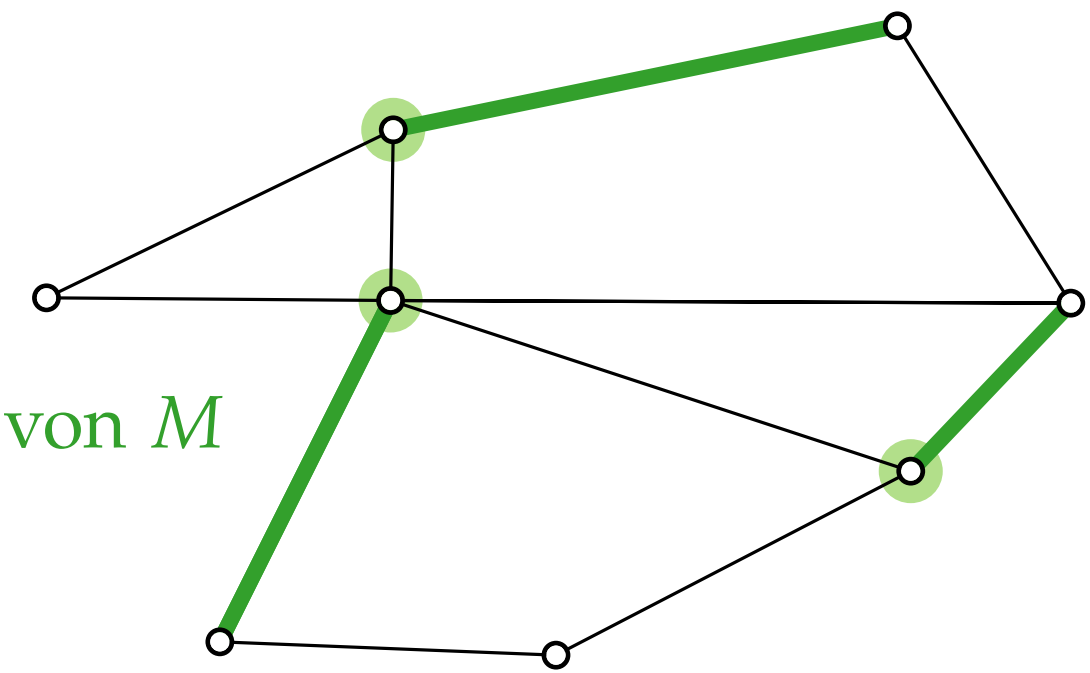
Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.

$\mathrm{OPT} \geq |M|$

Knotenüberdeckung von M

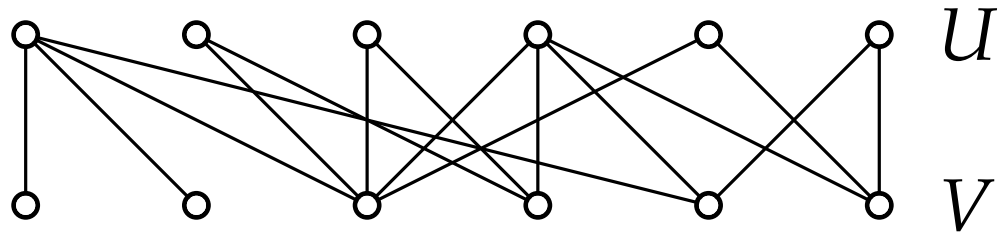


Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.

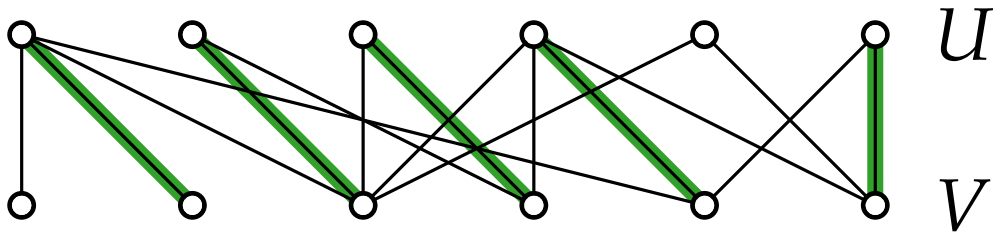
Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



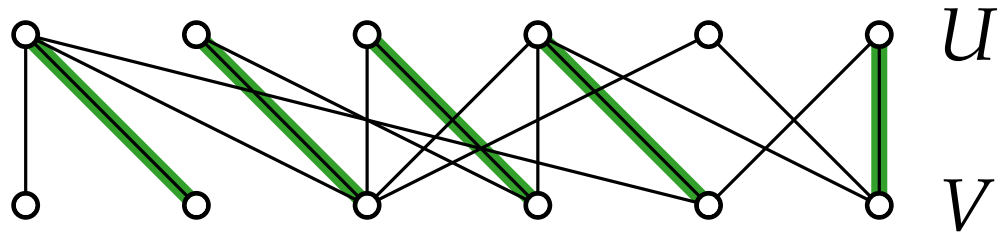
Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Satz von König

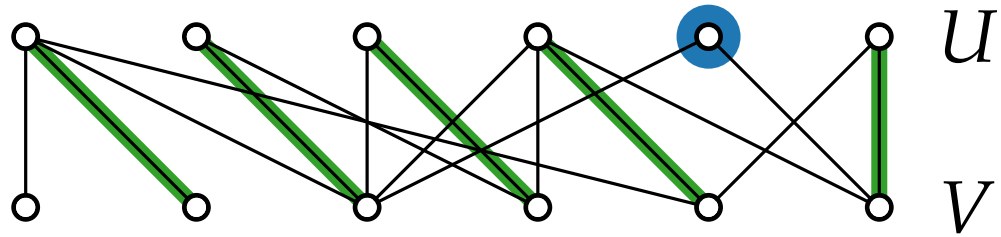
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U

Satz von König

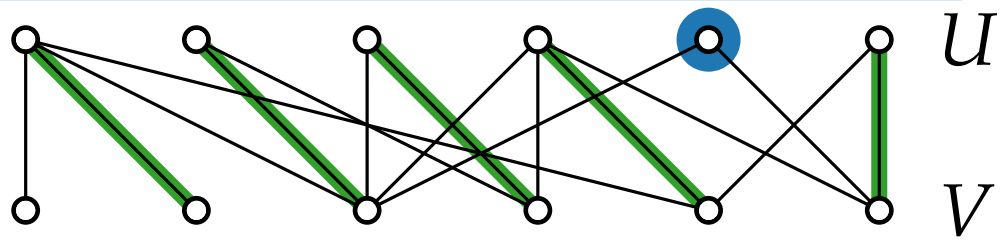
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U

Satz von König

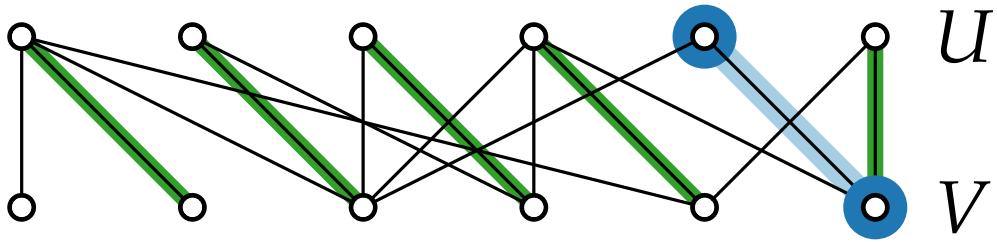
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

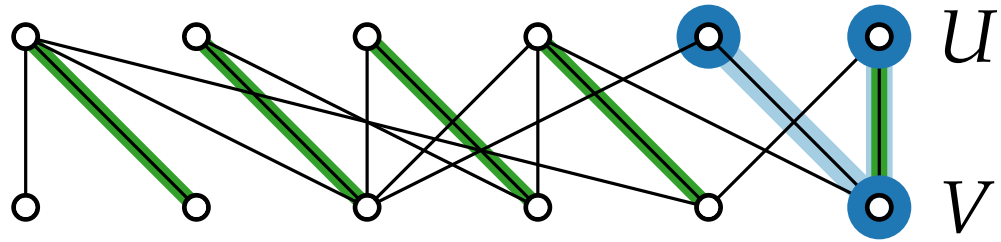
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

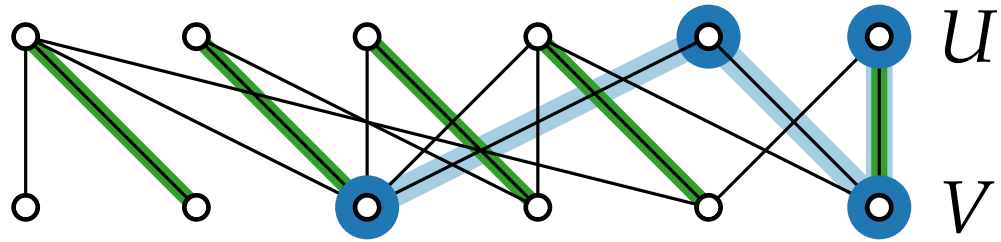
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

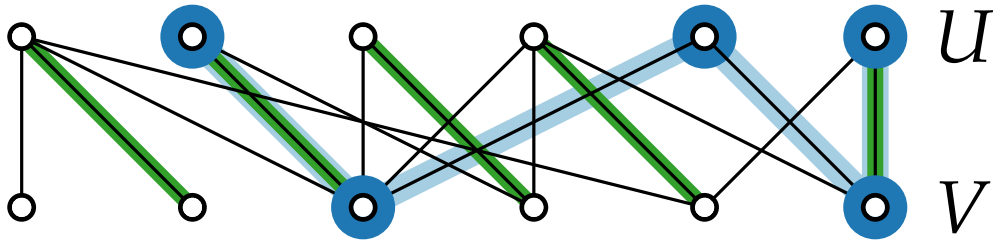
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

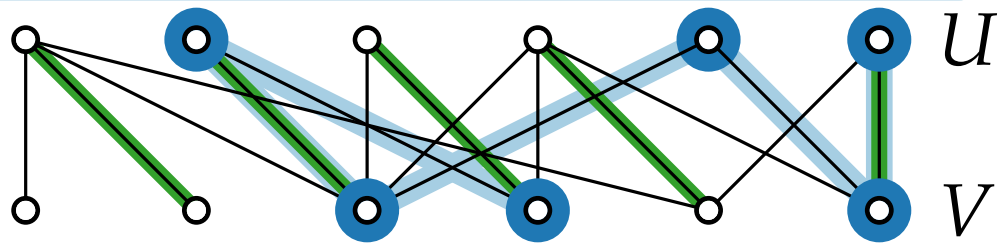
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

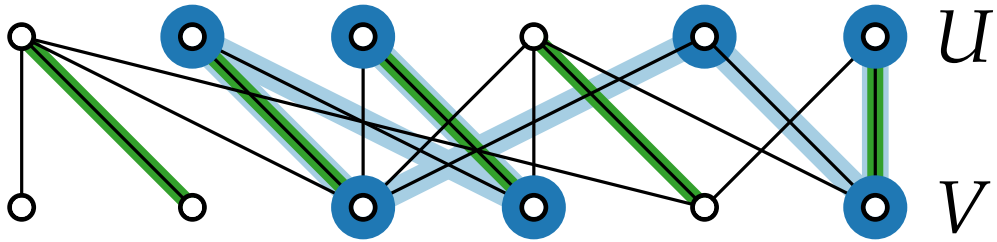
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

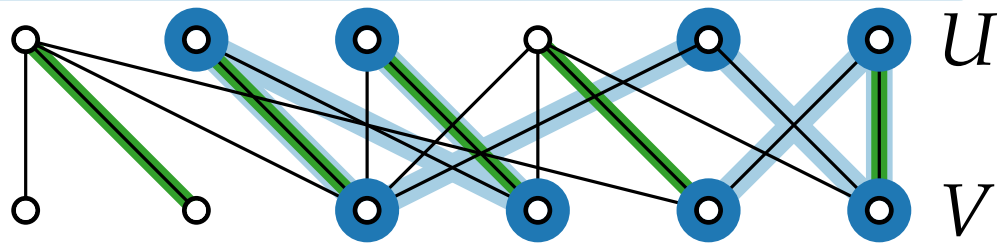
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

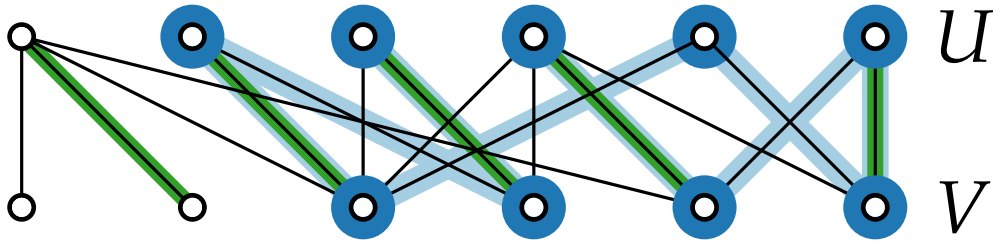
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

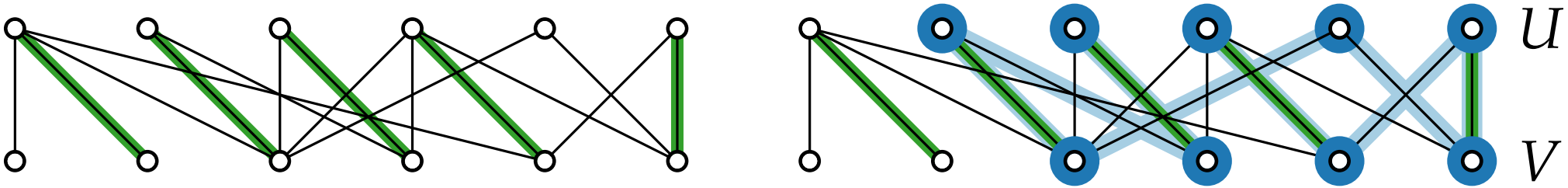
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.

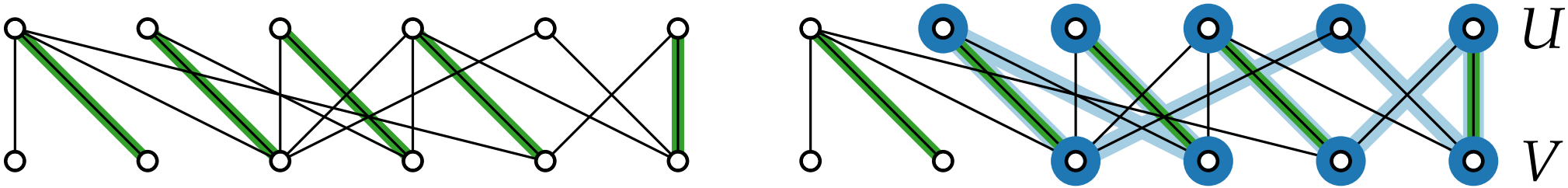


Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S:

Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.

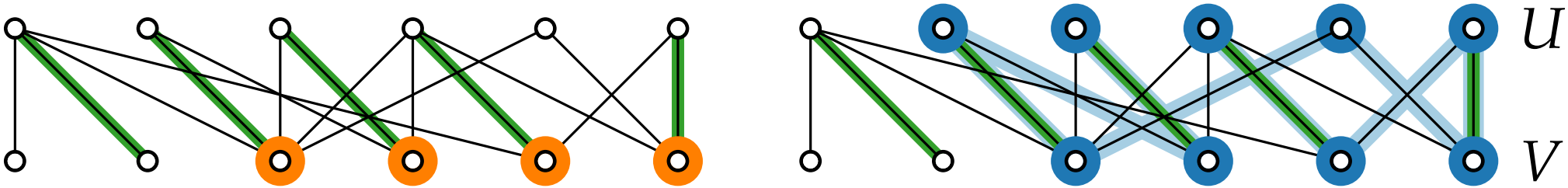


Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z)$

Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.

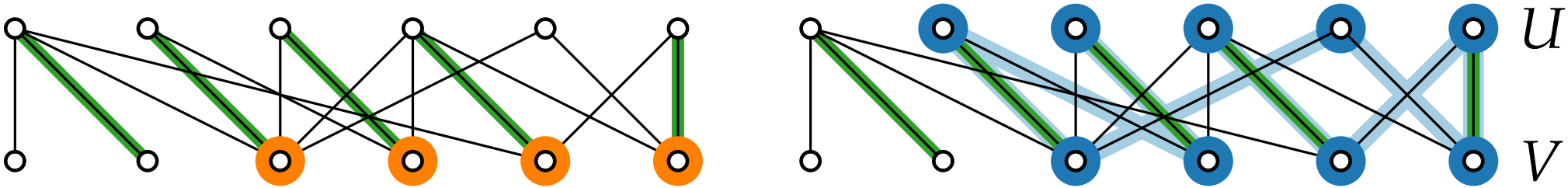


Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z)$

Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.

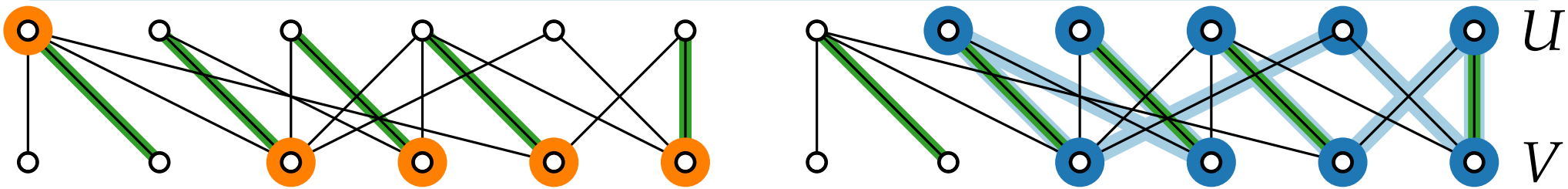


Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z) \cup (U \setminus Z)$

Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.

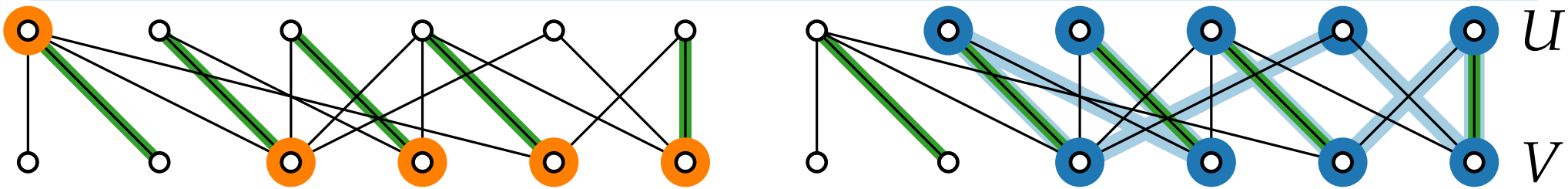


Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z) \cup (U \setminus Z)$

Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



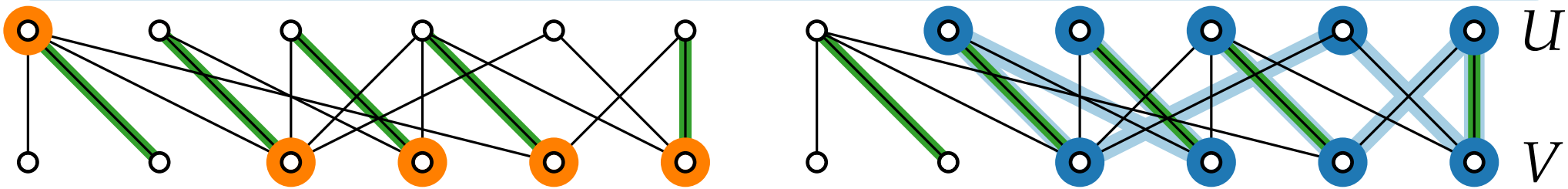
Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.

Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



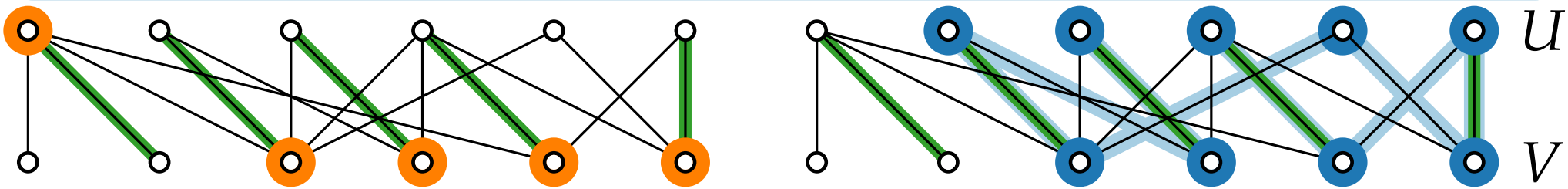
Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.

Satz von König

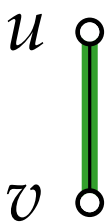
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

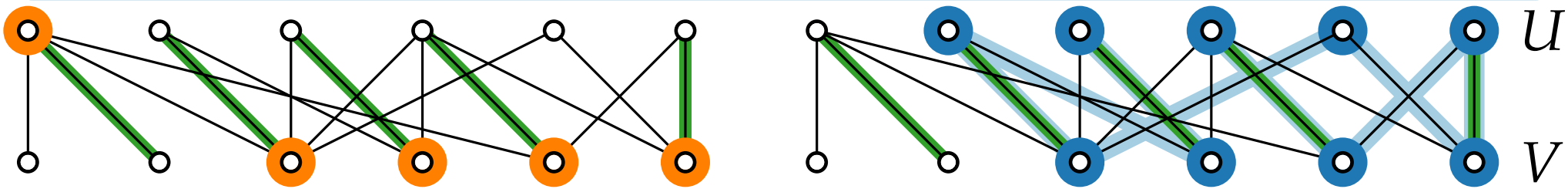
S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.



Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



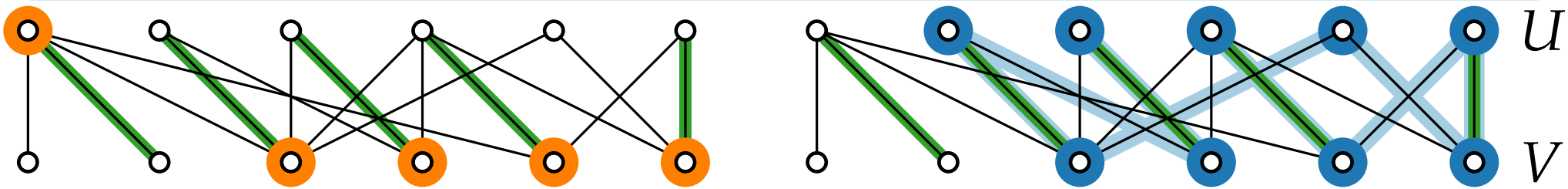
Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.

Satz von König

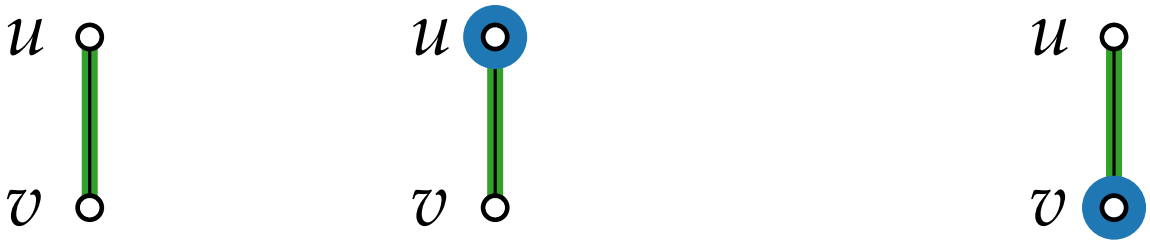
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

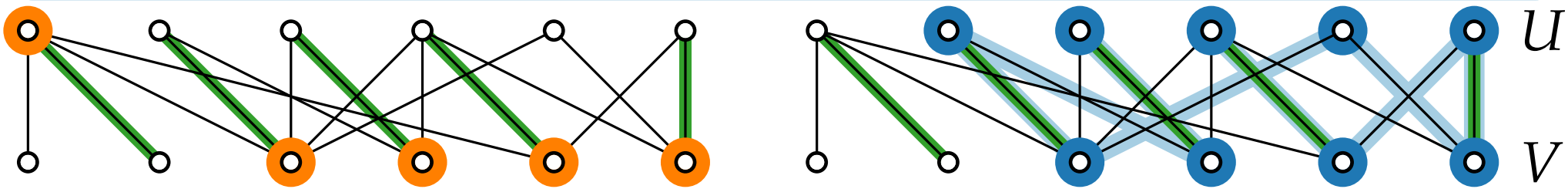
S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.



Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



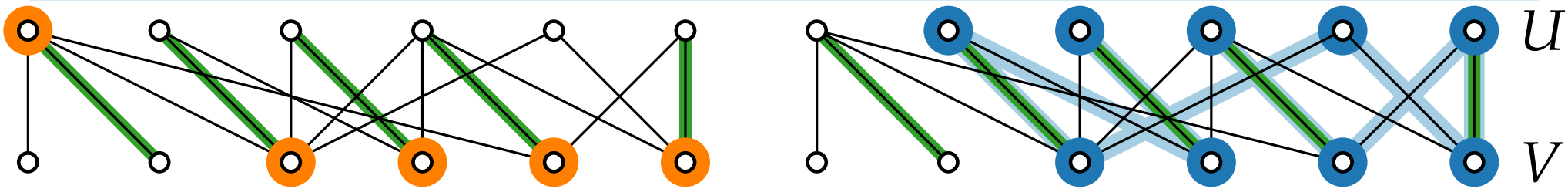
Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.

Satz von König

Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



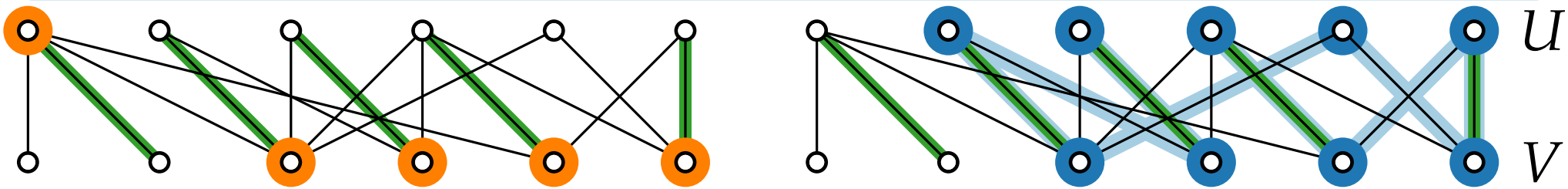
Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.

Satz von König

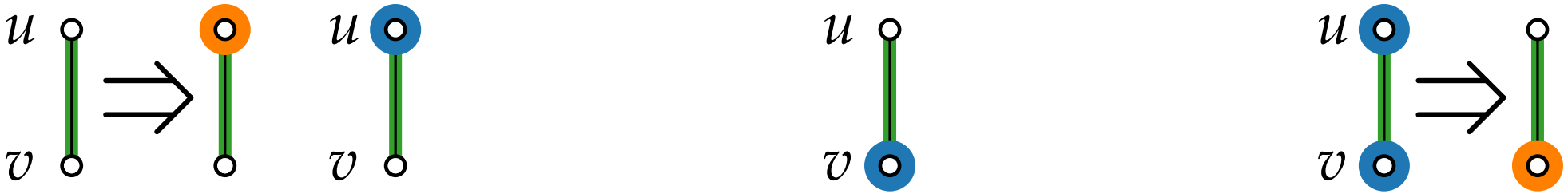
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

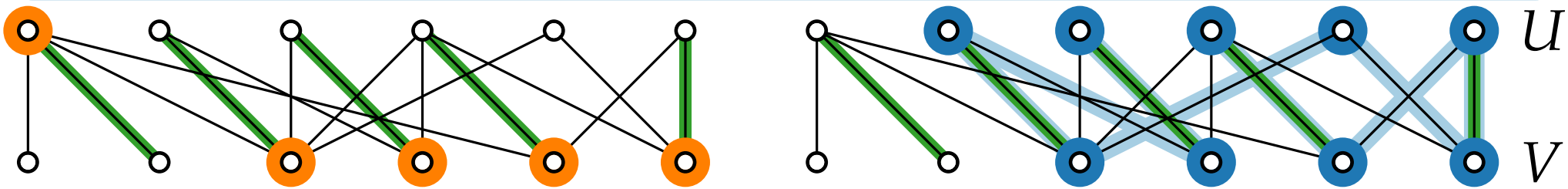
S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.



Satz von König

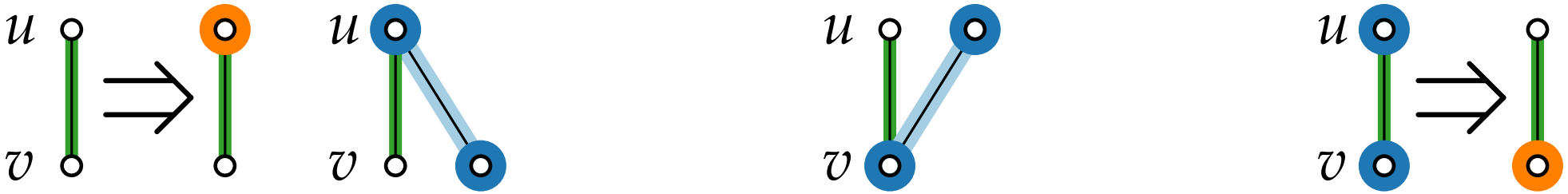
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

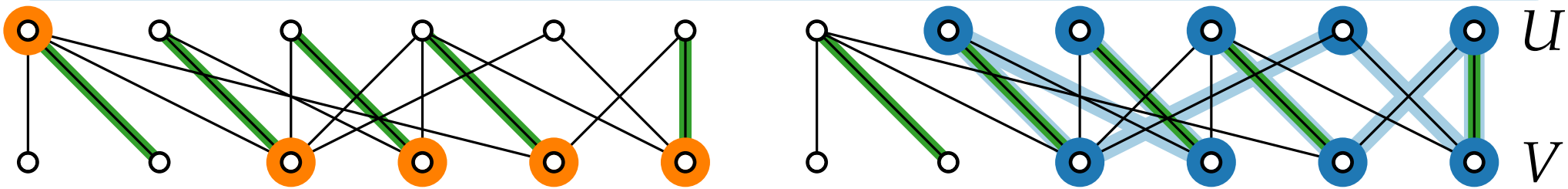
S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.



Satz von König

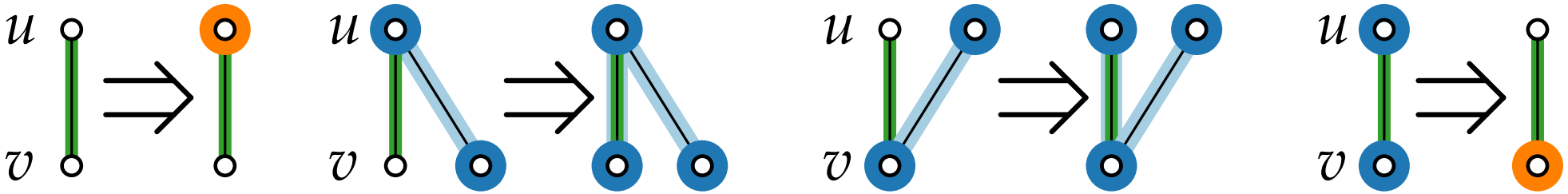
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

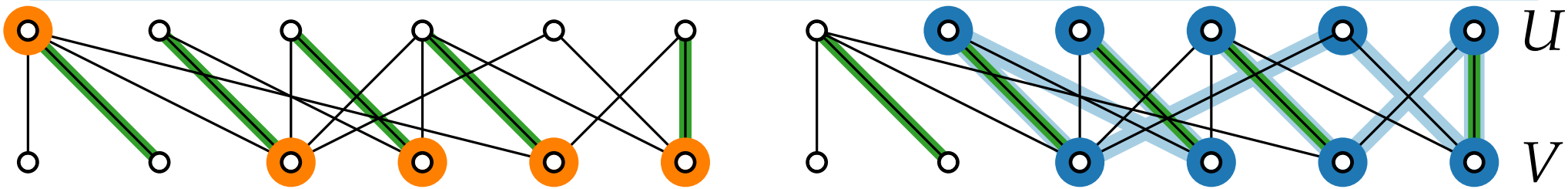
S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.



Satz von König

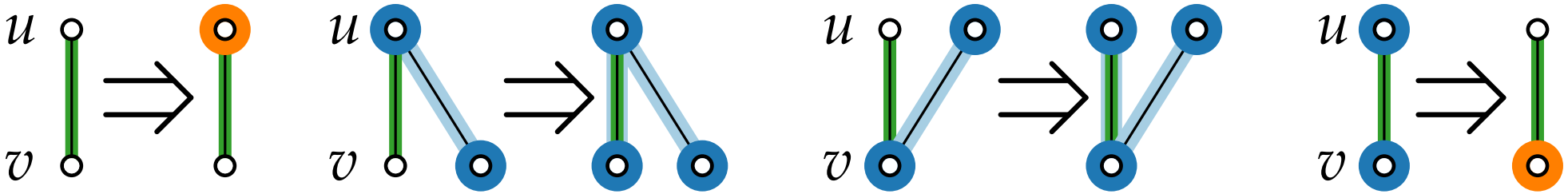
Satz. Sei $G = (U \cup V, E)$ ein bipartiter Graph, S eine kleinste Knotenüberdeckung und M ein größtes Matching. Dann ist $|S| = |M|$.



Z: alle ungematchten Knoten aus U
+ alle durch alternierende Pfade erreichbaren Knoten

S: $(V \cap Z) \cup (U \setminus Z)$

Lemma. Pro Kante aus M ist genau ein Knoten in S.



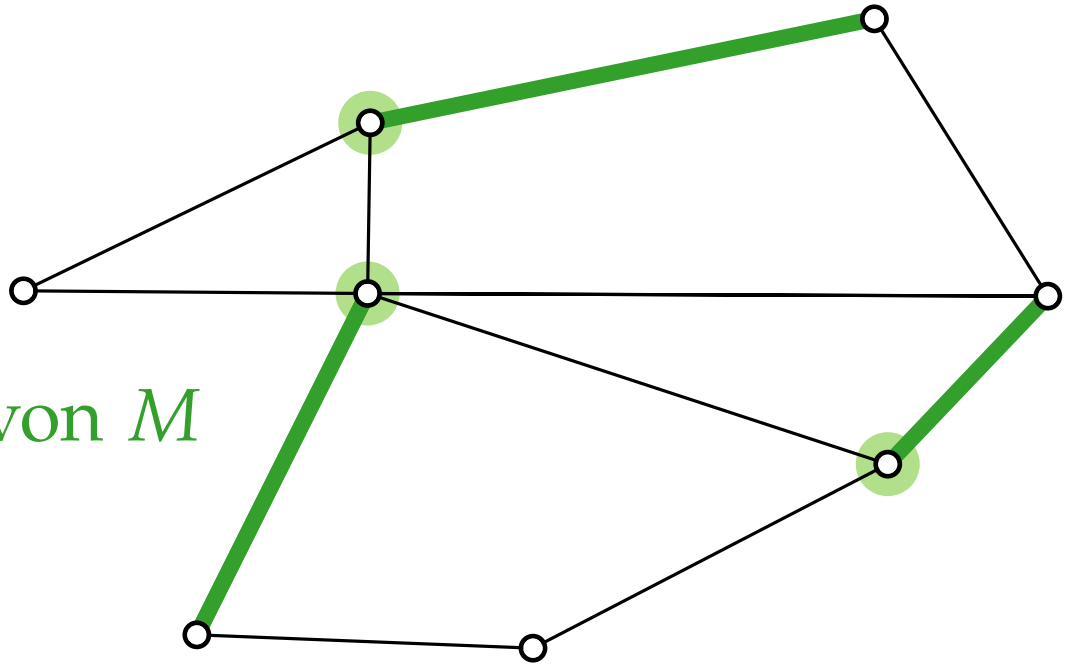
Satz. Vertex Cover für bipartite Graphen lässt sich in $O(\sqrt{V}E)$ Zeit lösen.

Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.

$\mathrm{OPT} \geq |M|$



Knotenüberdeckung von M

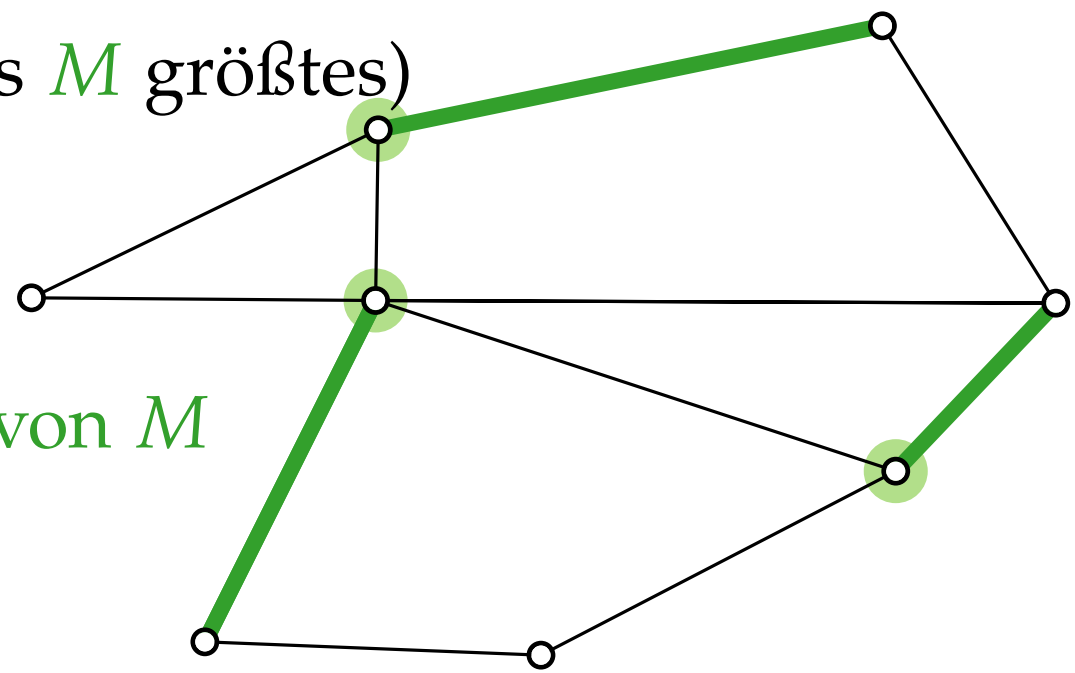
Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.

$\mathrm{OPT} \geq |M|$

bipartit: $\mathrm{OPT} = |M|$ (falls M größtes)



Knotenüberdeckung von M

Untere Schranke durch Matchings

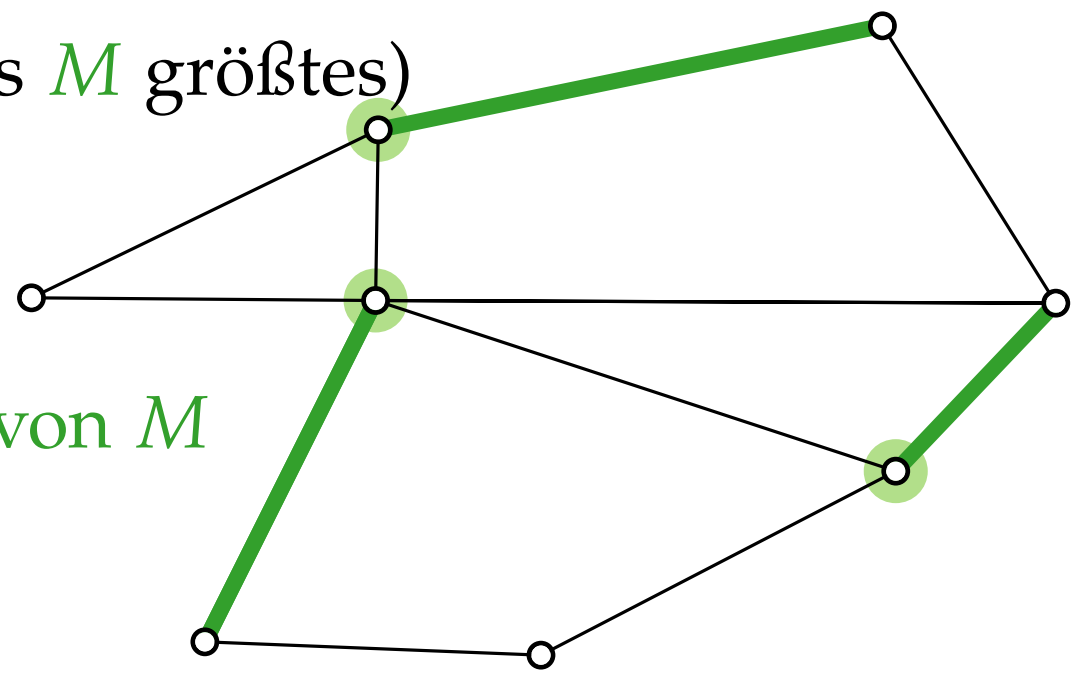
Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.

$$\text{OPT} \geq |M|$$

bipartit: $\text{OPT} = |M|$ (falls M größtes)

allgemein: $\text{OPT} = |M|$?

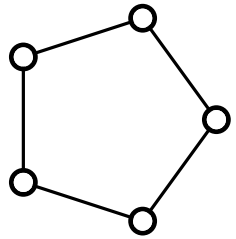


Knotenüberdeckung von M

Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

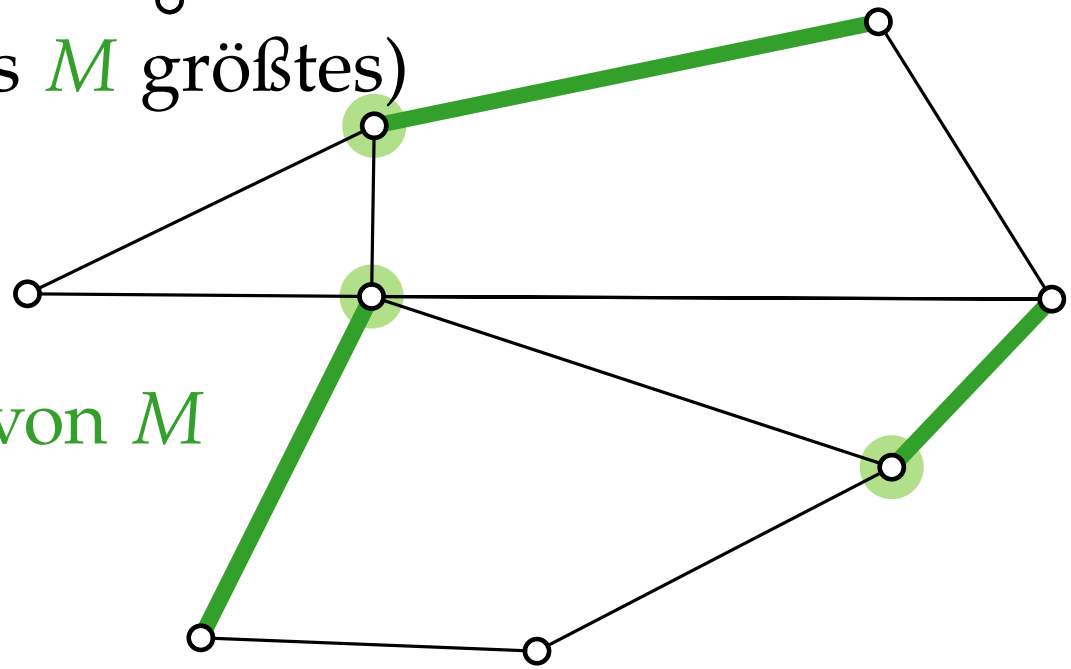
M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.



$$\text{OPT} \geq |M|$$

bipartit: $\text{OPT} = |M|$ (falls M größtes)

allgemein: $\text{OPT} = |M|$?

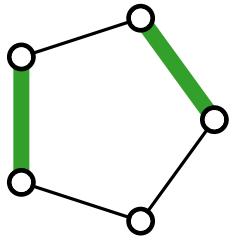


Knotenüberdeckung von M

Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

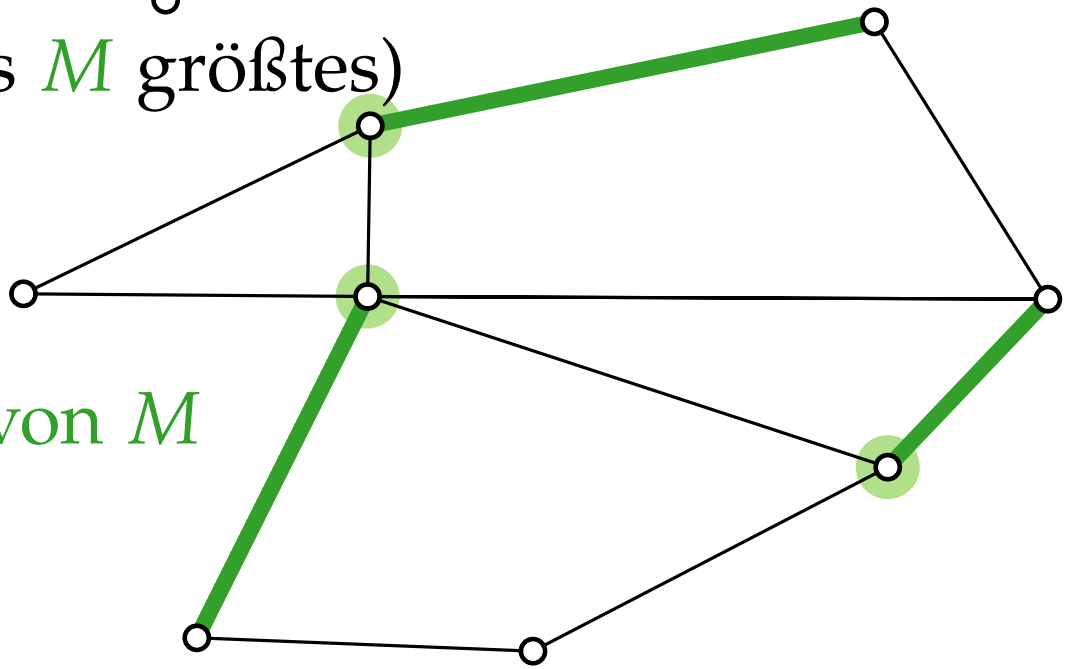
M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.



$$\text{OPT} \geq |M|$$

bipartit: $\text{OPT} = |M|$ (falls M größtes)

allgemein: $\text{OPT} = |M|$?

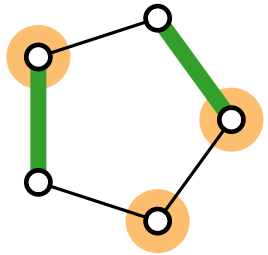


Knotenüberdeckung von M

Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

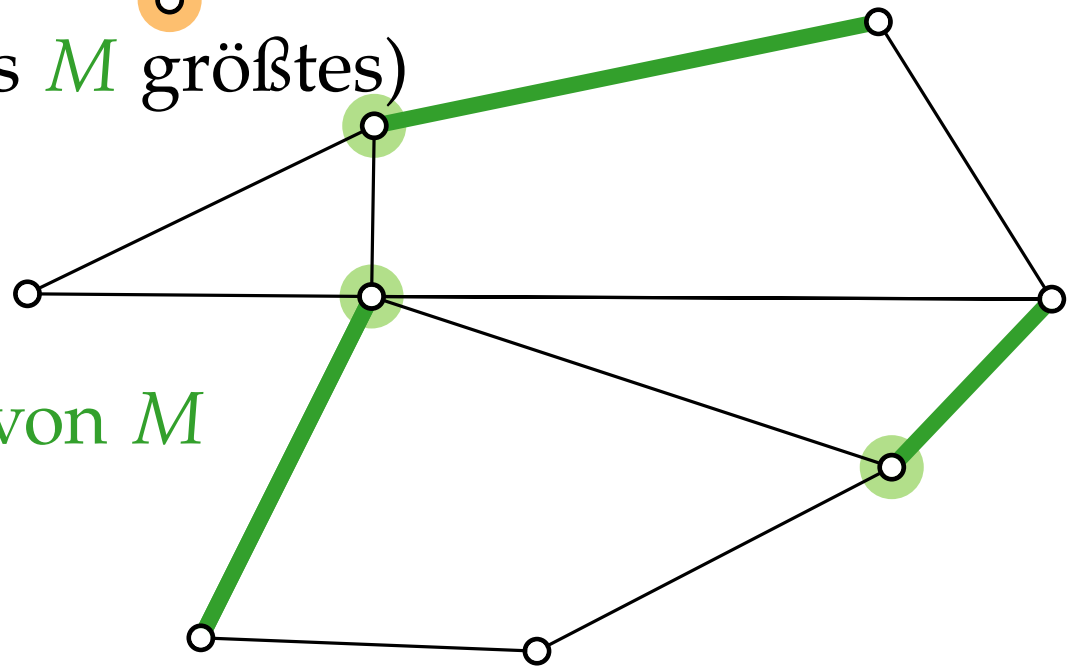
M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.



$$\text{OPT} \geq |M|$$

bipartit: $\text{OPT} = |M|$ (falls M größtes)

allgemein: $\text{OPT} = |M|$?

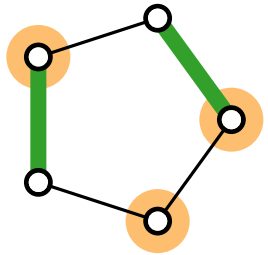


Knotenüberdeckung von M

Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

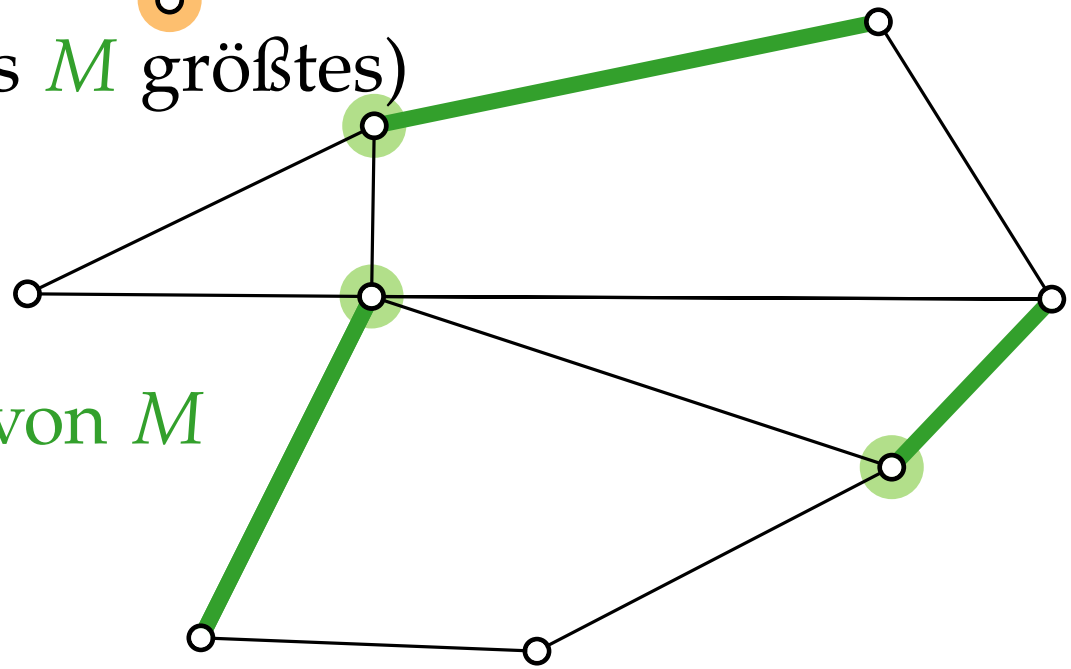
M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.



$$\mathrm{OPT} \geq |M|$$

bipartit: $\mathrm{OPT} = |M|$ (falls M größtes)

~~allgemein: $\mathrm{OPT} = |M|$?~~

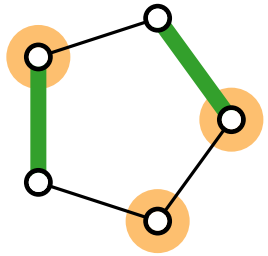


Knotenüberdeckung von M

Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

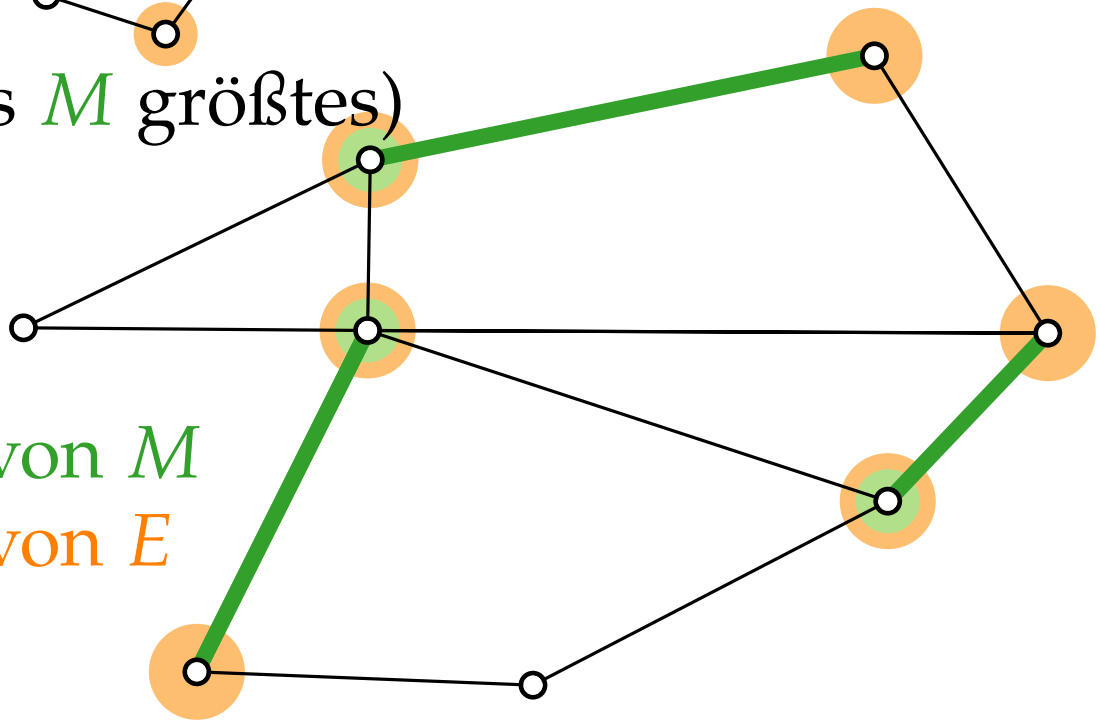
M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.



$$\mathrm{OPT} \geq |M|$$

bipartit: $\mathrm{OPT} = |M|$ (falls M größtes)

~~allgemein: $\mathrm{OPT} = |M|$?~~



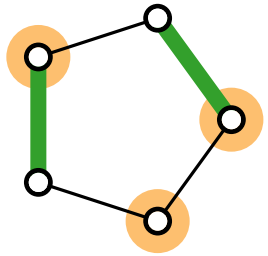
Knotenüberdeckung von M

Knotenüberdeckung von E

Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

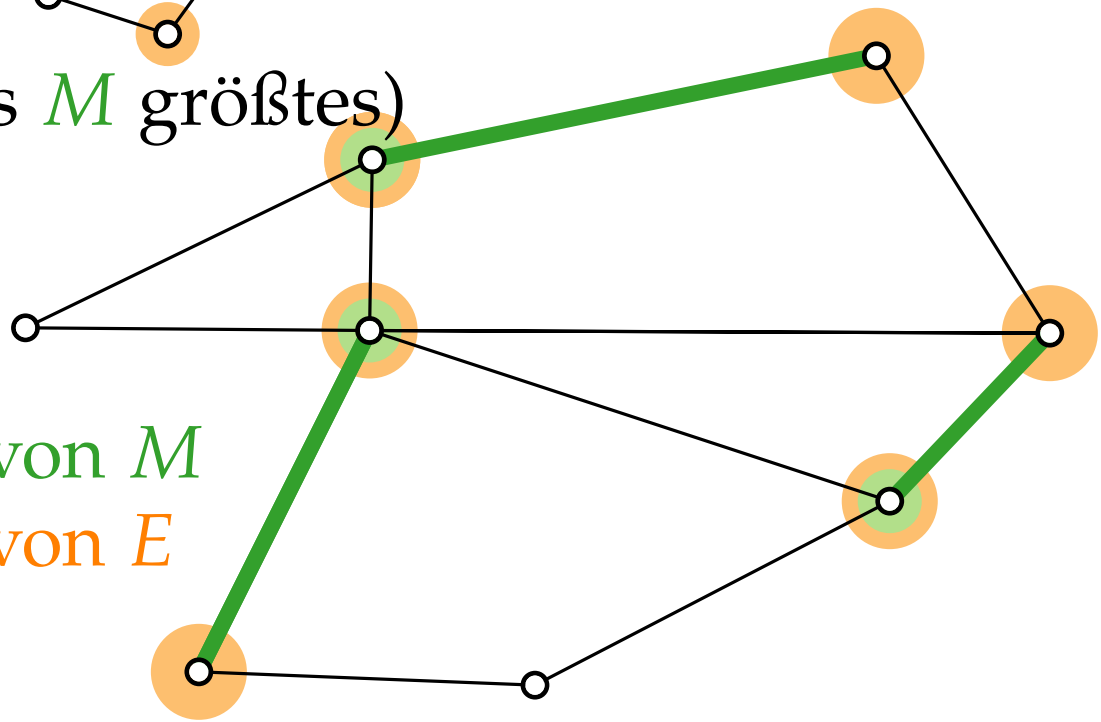
M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.



$$\mathrm{OPT} \geq |M|$$

bipartit: $\mathrm{OPT} = |M|$ (falls M größtes)

~~allgemein: $\mathrm{OPT} = |M|$?~~



Knotenüberdeckung von M

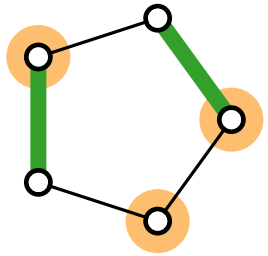
Knotenüberdeckung von E

$\mathrm{ALG} = 2 \cdot |M| \leq$

Untere Schranke durch Matchings

Eine Kantenmenge $M \subseteq E$ eines Graphen $G = (V, E)$ heißt **Matching**, wenn keine zwei Kanten aus M adjazent sind.

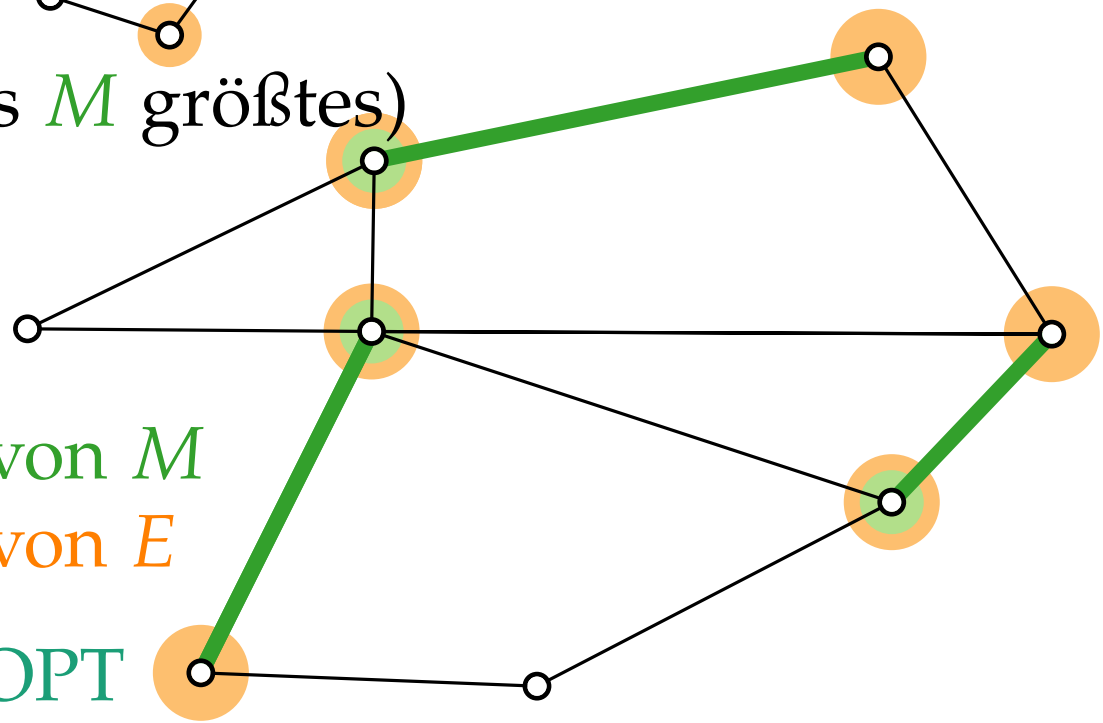
M heißt **nicht erweiterbar**, wenn es kein Matching M' mit $M' \supsetneq M$ für G gibt.



$$\text{OPT} \geq |M|$$

bipartit: $\text{OPT} = |M|$ (falls M größtes)

~~allgemein: $\text{OPT} = |M|$?~~



Knotenüberdeckung von M

Knotenüberdeckung von E

$$\text{ALG} = 2 \cdot |M| \leq 2 \cdot \text{OPT}$$

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$

foreach $e \in E(G)$ **do**

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$

foreach $e \in E(G)$ **do**

  if e nicht adjazent zu Kante in M **then**

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$
foreach $e \in E(G)$ **do**
  if e nicht adjazent zu Kante in M **then**
    $M \leftarrow M \cup \{e\}$

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$
foreach $e \in E(G)$ **do**
  if e nicht adjazent zu Kante in M **then**
    $M \leftarrow M \cup \{e\}$
return $\{\, u, v \mid uv \in M \,\}$

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$
foreach $e \in E(G)$ **do**
  if e nicht adjazent zu Kante in M **then**
    $M \leftarrow M \cup \{e\}$
return $\{\, u, v \mid uv \in M \,\}$

Satz. Obiger Algorithmus ist ein Faktor-2-Approximationsalgorithmus für Vertex Cover.

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$
foreach $e \in E(G)$ **do**
  if e nicht adjazent zu Kante in M **then**
    $M \leftarrow M \cup \{e\}$
return $\{\, u, v \mid uv \in M \,\}$

Satz. Obiger Algorithmus ist ein Faktor-2-Approximationsalgorithmus für Vertex Cover.

Der beste bekannte Approximationsfaktor für Vertex Cover ist

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$
foreach $e \in E(G)$ **do**
  if e nicht adjazent zu Kante in M **then**
    $M \leftarrow M \cup \{e\}$
return $\{\, u, v \mid uv \in M \,\}$

Satz. Obiger Algorithmus ist ein Faktor-2-Approximationsalgorithmus für Vertex Cover.

Der beste bekannte Approximationsfaktor für Vertex Cover ist $2 - \Theta(1/\sqrt{\log n})$

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$
foreach $e \in E(G)$ **do**
  if e nicht adjazent zu Kante in M **then**
    $M \leftarrow M \cup \{e\}$
return $\{\, u, v \mid uv \in M \,\}$

Satz. Obiger Algorithmus ist ein Faktor-2-Approximationsalgorithmus für Vertex Cover.

Der beste bekannte Approximationsfaktor für Vertex Cover ist $2 - \Theta(1/\sqrt{\log n})$
Vertex Cover lässt sich nicht mit Faktor 1,3606 approximieren (außer P=NP)

Approximationsalg. für Vertex Cover

Algorithmus Knotenüberdeckung(G)

$M \leftarrow \emptyset$
foreach $e \in E(G)$ **do**
  if e nicht adjazent zu Kante in M **then**
    $M \leftarrow M \cup \{e\}$
return $\{\, u, v \mid uv \in M \,\}$

Satz. Obiger Algorithmus ist ein Faktor-2-Approximationsalgorithmus für Vertex Cover.

Der beste bekannte Approximationsfaktor für Vertex Cover ist $2 - \Theta(1/\sqrt{\log n})$
Vertex Cover lässt sich nicht mit Faktor 1,3606 approximieren (außer P=NP)
Vertex Cover lässt sich nicht mit Faktor $2 - \Theta(1)$ approximieren, wenn "Unique Game Conjecture" hält.