

Parametrisierte Algorithmen

FPT-Techniken: Suchbäume, Kernbildung, iterative Kompression



Allgemeines

Beteiligte



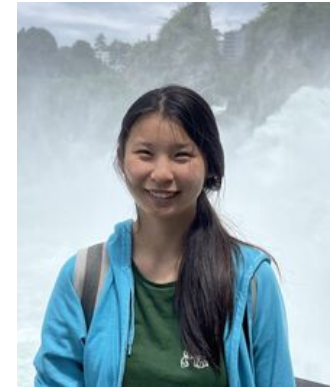
Thomas



Jean-Pierre



Marcus



Wendy

Allgemeines

Beteiligte



Thomas



Jean-Pierre



Marcus



Wendy



Ihr

Allgemeines

Beteiligte



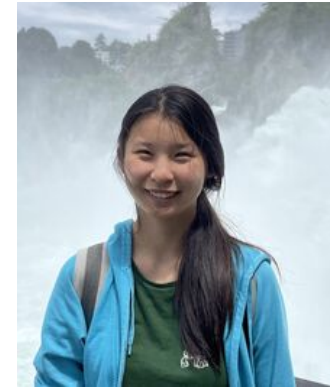
Thomas



Jean-Pierre



Marcus



Wendy



Ihr

Materialien & Infos

- Vorlesungsfolien, Übungsblätter, Online-VL von Sommer 21
- Homepage: https://scale.iti.kit.edu/teaching/2024ws/param_algo/
- Discord: <https://discord.gg/JrM8qaJvCK> (falls ihr schon dort seid: sendet `!help` an scale-bot)

Allgemeines

Beteiligte



Thomas



Jean-Pierre



Marcus



Wendy



Ihr

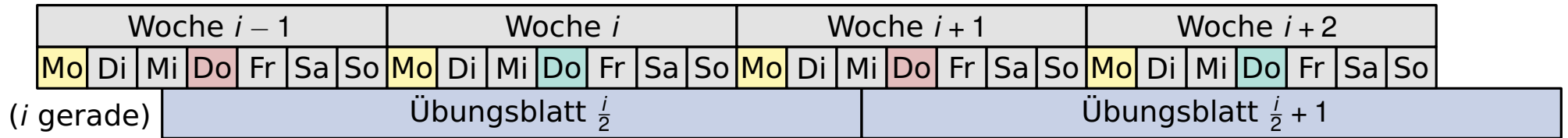
Materialien & Infos

- Vorlesungsfolien, Übungsblätter, Online-VL von Sommer 21
- Homepage: https://scale.iti.kit.edu/teaching/2024ws/param_algo/
- Discord: <https://discord.gg/JrM8qaJvCK> (falls ihr schon dort seid: sendet `!help` an scale-bot)

Voraussetzungen

- gutes algorithmisches Verständnis
- inhaltlich: keine

Ablauf



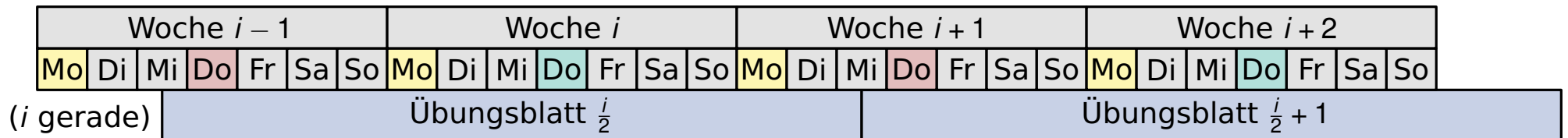
Ablauf

Woche $i - 1$							Woche i							Woche $i + 1$							Woche $i + 2$						
Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So
(i gerade)							Übungsblatt $\frac{i}{2}$														Übungsblatt $\frac{i}{2} + 1$						

Vorlesung

- Vorlesung mit Folien
- neuer Stoff

Ablauf



Vorlesung

- Vorlesung mit Folien
- neuer Stoff

Aktiv-Session

- gemeinsames Erarbeiten weiterführender Aspekte
- Kuriositäten
- gemeinsames Papier-Lesen

Woche $i - 1$							Woche i							Woche $i + 1$							Woche $i + 2$						
Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So
(i gerade)							Übungsblatt $\frac{i}{2}$														Übungsblatt $\frac{i}{2} + 1$						

Vorlesung

- Vorlesung mit Folien
- neuer Stoff

Übungsblätter

- Abgabe in (Zweier-)Gruppen
- korrigiert von Sven

Aktiv-Session

- gemeinsames Erarbeiten weiterführender Aspekte
- Kuriositäten
- gemeinsames Papier-Lesen

Übungen

- mit Jean-Pierre, Marcus, Wendy
- Übung des Stoffs
- Unterstützung bei Übungsblättern
- ???

Woche $i - 1$							Woche i							Woche $i + 1$							Woche $i + 2$						
Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So	Mo	Di	Mi	Do	Fr	Sa	So
(i gerade)							Übungsblatt $\frac{i}{2}$														Übungsblatt $\frac{i}{2} + 1$						

Vorlesung

- Vorlesung mit Folien
- neuer Stoff

Übungsblätter

- Abgabe in (Zweier-)Gruppen
- korrigiert von Sven

Aktiv-Session

- gemeinsames Erarbeiten weiterführender Aspekte
- Kuriositäten
- gemeinsames Papier-Lesen

Übungen

- mit Jean-Pierre, Marcus, Wendy
- Übung des Stoffs
- Unterstützung bei Übungsblättern
- ???

Prüfung

- mündlich
- Zulassung: Übungsschein

Übungsschein

Ziel: $\frac{1}{2}$ der Punkte auf **jedem** Übungsblatt

Übungsschein

Ziel: $\frac{1}{2}$ der Punkte auf **jedem** Übungsblatt

Und wenn ich mal nicht auf die Lösung komme?

Übungsschein

Ziel: $\frac{1}{2}$ der Punkte auf **jedem** Übungsblatt

Und wenn ich mal nicht auf die Lösung komme?

- es gibt auch Punkte für Erklärungen, was nicht funktioniert
- schreibt auf, was ihr versucht habt und warum das nicht geht

Übungsschein

Ziel: $\frac{1}{2}$ der Punkte auf **jedem** Übungsblatt

Und wenn ich mal nicht auf die Lösung komme?

- es gibt auch Punkte für Erklärungen, was nicht funktioniert
- schreibt auf, was ihr versucht habt und warum das nicht geht
- außerdem: viele Gelegenheiten für Unterstützung
 - spricht mit euren Kommiliton:innen
 - fragt in der Übung
 - fragt via Discord

Übungsschein

Ziel: $\frac{1}{2}$ der Punkte auf **jedem** Übungsblatt

Und wenn ich mal nicht auf die Lösung komme?

- es gibt auch Punkte für Erklärungen, was nicht funktioniert
- schreibt auf, was ihr versucht habt und warum das nicht geht
- außerdem: viele Gelegenheiten für Unterstützung
 - spricht mit euren Kommiliton:innen
 - fragt in der Übung
 - fragt via Discord

Und wenn ich ein Blatt mal nicht abgeben kann?

Übungsschein

Ziel: $\frac{1}{2}$ der Punkte auf **jedem** Übungsblatt

Und wenn ich mal nicht auf die Lösung komme?

- es gibt auch Punkte für Erklärungen, was nicht funktioniert
- schreibt auf, was ihr versucht habt und warum das nicht geht
- außerdem: viele Gelegenheiten für Unterstützung
 - spricht mit euren Kommiliton:innen
 - fragt in der Übung
 - fragt via Discord

Und wenn ich ein Blatt mal nicht abgeben kann?

- uns ist klar, dass im Leben auch noch andere Dinge passieren
- spricht uns an, wir finden eine Lösung

Übungsschein

Ziel: $\frac{1}{2}$ der Punkte auf **jedem** Übungsblatt

Und wenn ich mal nicht auf die Lösung komme?

- es gibt auch Punkte für Erklärungen, was nicht funktioniert
- schreibt auf, was ihr versucht habt und warum das nicht geht
- außerdem: viele Gelegenheiten für Unterstützung
 - spricht mit euren Kommiliton:innen
 - fragt in der Übung
 - fragt via Discord

Und wenn ich ein Blatt mal nicht abgeben kann?

- uns ist klar, dass im Leben auch noch andere Dinge passieren
- spricht uns an, wir finden eine Lösung

Unser Ziel

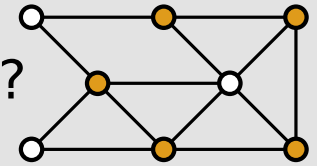
- ihr beschäftigt euch aktiv mit dem Stoff und schreibt Lösungen auf
- Übungsschein ist dann keine große Hürde

Parametrisierte Sichtweise

Problem: k -VERTEX COVER

Gibt es im Graphen $G = (V, E)$ ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)

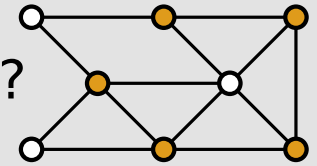


Parametrisierte Sichtweise

Problem: k -VERTEX COVER

Gibt es im Graphen $G = (V, E)$ ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Können wir das effizient lösen?

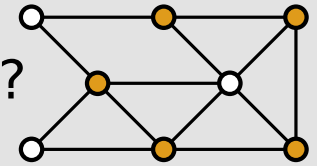
- k -VERTEX COVER ist NP-vollständig

Parametrisierte Sichtweise

Problem: k -VERTEX COVER

Gibt es im Graphen $G = (V, E)$ ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Können wir das effizient lösen?

- k -VERTEX COVER ist NP-vollständig

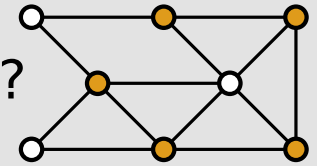
Und was, wenn die betrachteten Graphen gutartig sind?

Parametrisierte Sichtweise

Problem: k -VERTEX COVER

Gibt es im Graphen $G = (V, E)$ ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Können wir das effizient lösen?

- k -VERTEX COVER ist NP-vollständig

Und was, wenn die betrachteten Graphen gutartig sind?

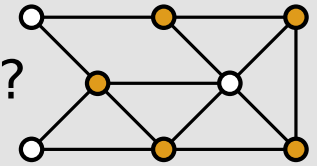
- Was, wenn G kleinen Maximalgrad hat?

Parametrisierte Sichtweise

Problem: k -VERTEX COVER

Gibt es im Graphen $G = (V, E)$ ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Können wir das effizient lösen?

- k -VERTEX COVER ist NP-vollständig

Und was, wenn die betrachteten Graphen gutartig sind?

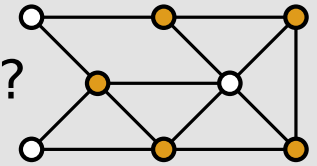
- Was, wenn G kleinen Maximalgrad hat?
- Was, wenn das minimale Vertex Cover in G klein ist?

Parametrisierte Sichtweise

Problem: k -VERTEX COVER

Gibt es im Graphen $G = (V, E)$ ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Können wir das effizient lösen?

- k -VERTEX COVER ist NP-vollständig

Und was, wenn die betrachteten Graphen gutartig sind?

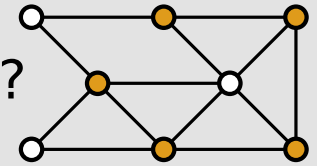
- Was, wenn G kleinen Maximalgrad hat?
- Was, wenn das minimale Vertex Cover in G klein ist?
- Was, wenn G kleine chromatische Zahl hat?

Parametrisierte Sichtweise

Problem: k -VERTEX COVER

Gibt es im Graphen $G = (V, E)$ ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Können wir das effizient lösen?

- k -VERTEX COVER ist NP-vollständig

Und was, wenn die betrachteten Graphen gutartig sind?

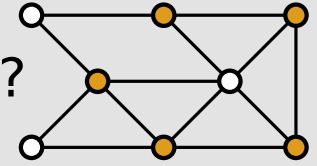
- Was, wenn G kleinen Maximalgrad hat?
- Was, wenn das minimale Vertex Cover in G klein ist?
- Was, wenn G kleine chromatische Zahl hat?
- Oder kleine Cliquenzahl? Kleine Baumweite? Kleine ... ?

Parametrisierte Sichtweise

Problem: k -VERTEX COVER

Gibt es im Graphen $G = (V, E)$ ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Können wir das effizient lösen?

- k -VERTEX COVER ist NP-vollständig

Und was, wenn die betrachteten Graphen gutartig sind?

- Was, wenn G kleinen Maximalgrad hat?
- Was, wenn das minimale Vertex Cover in G klein ist?
- Was, wenn G kleine chromatische Zahl hat?
- Oder kleine Cliquenzahl? Kleine Baumweite? Kleine ... ?

Ziel

- Laufzeitanalyse nicht nur bzgl. der Eingabegröße
- sondern außerdem bzgl. eines (oder mehrerer) Parameter

Grundlegende Definitionen

Definition

Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Grundlegende Definitionen

Definition

Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Beachte: ein Algorithmus löst das parametrisierte Problem L , wenn es für jede Instanz (x, k) korrekt entscheidet ob $(x, k) \in L$

Grundlegende Definitionen

Definition

Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Beachte: ein Algorithmus löst das parametrisierte Problem L , wenn es für jede Instanz (x, k) korrekt entscheidet ob $(x, k) \in L$

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.

Grundlegende Definitionen

Definition

Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Beachte: ein Algorithmus löst das parametrisierte Problem L , wenn es für jede Instanz (x, k) korrekt entscheidet ob $(x, k) \in L$

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.

Welche der Laufzeiten gehören zu einem FPT-Algorithmus?

$A = 2^k(n^4 + k^2)$	$B = k^{\log n}$	$C = 2^{k^2+2 \log n}$	$D = k^2 n!$	$E = n^{17} k^{k!}$
$F = n^{\log k}$	$G = k! n \log n$	$H = n^k n^2$	$I = n^{\frac{3}{2}}$	$J = 2^{k \log n}$

Grundlegende Definitionen

Definition

Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Beachte: ein Algorithmus löst das parametrisierte Problem L , wenn es für jede Instanz (x, k) korrekt entscheidet ob $(x, k) \in L$

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.

Welche der Laufzeiten gehören zu einem FPT-Algorithmus?

$$\begin{array}{ccccc}
 A = 2^k(n^4 + k^2) & B = k^{\log n} & C = 2^{k^2+2\log n} & D = k^2 n! & E = n^{17} k^{k!} \\
 F = n^{\log k} & G = k! n \log n & H = n^k n^2 & I = n^{\frac{3}{2}} & J = 2^{k \log n}
 \end{array}$$

$$\text{FPT} \Leftrightarrow \left\{ \begin{array}{l} \blacksquare k \text{ konstant} \Rightarrow \text{polynomielle Laufzeit} \\ \blacksquare k \text{ konstant} \Rightarrow \text{asymptotische Laufzeit unabhängig von } k \end{array} \right.$$

Beispiel: parametrisiertes Vertex Cover

Definition

Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.

Beispiel: parametrisiertes Vertex Cover

Definition

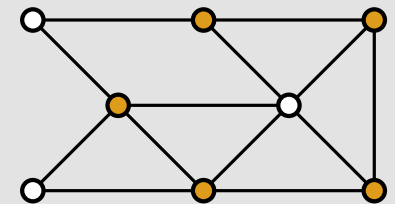
Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Beispiel: parametrisiertes Vertex Cover

Definition

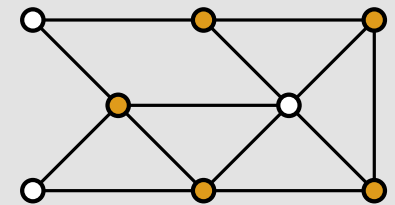
Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Komplexität

- VERTEX COVER ist NP-vollständig

Beispiel: parametrisiertes Vertex Cover

Definition

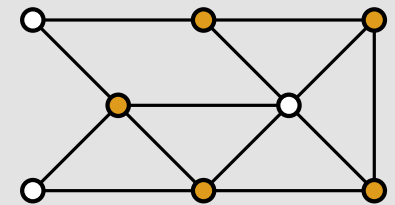
Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Komplexität

- VERTEX COVER ist NP-vollständig
- Aufzählung aller $\binom{n}{k}$ Teilmengen der Größe k (Brute-Force):

Beispiel: parametrisiertes Vertex Cover

Definition

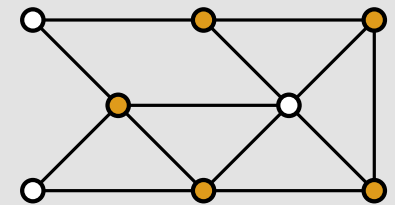
Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.


Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Komplexität

- VERTEX COVER ist NP-vollständig
- Aufzählung aller $\binom{n}{k}$ Teilmengen der Größe k (Brute-Force): $O(n^k m)$



 polynomiell für konstantes k !

Beispiel: parametrisiertes Vertex Cover

Definition

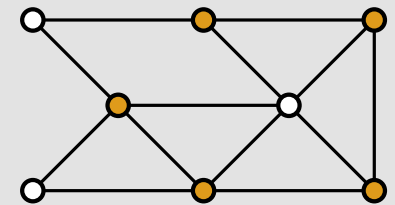
Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$. Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ ist k der **Parameter**.

Definition

Ein parametrisiertes Problem L ist **fixed parameter tractable (FPT)**, wenn es einen Algorithmus gibt, der L in $O(f(k)n^c)$ löst. Dabei ist f eine berechenbare Funktion, n die Eingabegröße und c eine Konstante.

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Komplexität

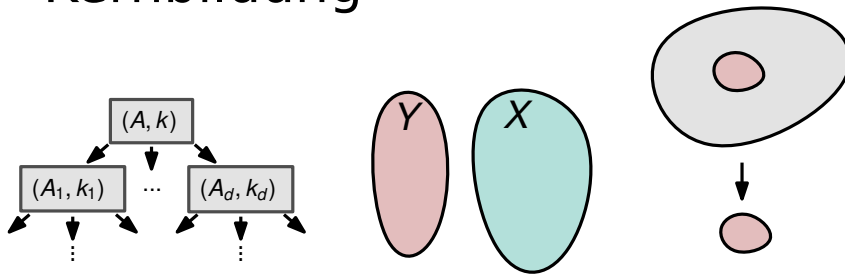
- VERTEX COVER ist NP-vollständig
- Aufzählung aller $\binom{n}{k}$ Teilmengen der Größe k (Brute-Force): $O(n^k m)$
- gilt VERTEX COVER \in FPT?

polynomiell für konstantes $k!$

Inhalt

Basic Toolbox

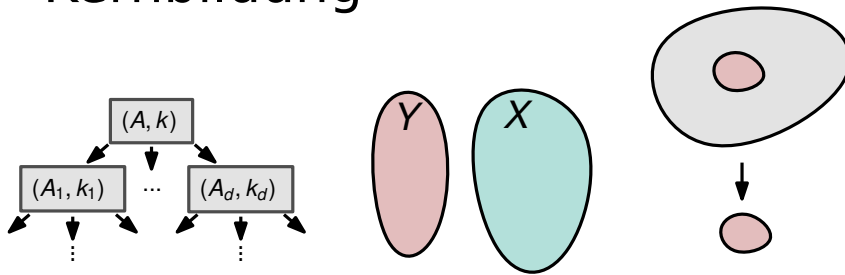
- beschränkte Suchbäume
- iterative Kompression
- Kernbildung



Inhalt

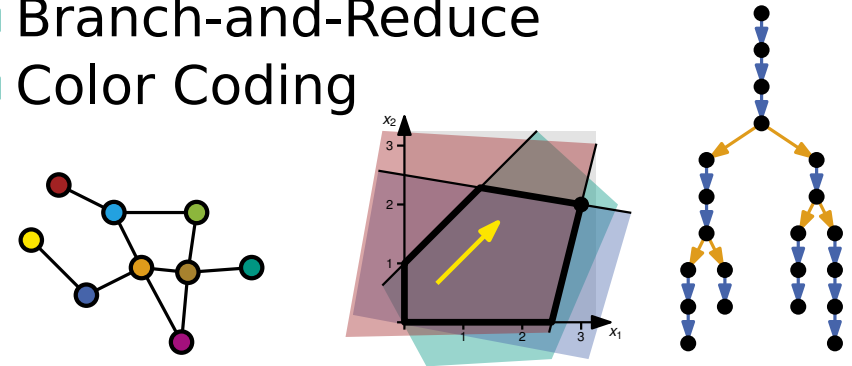
Basic Toolbox

- beschränkte Suchbäume
- iterative Kompression
- Kernbildung



Erweiterte Toolbox

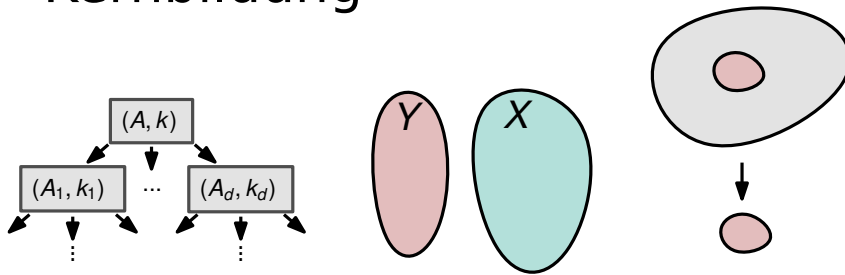
- lineare Programme
- Branch-and-Reduce
- Color Coding



Inhalt

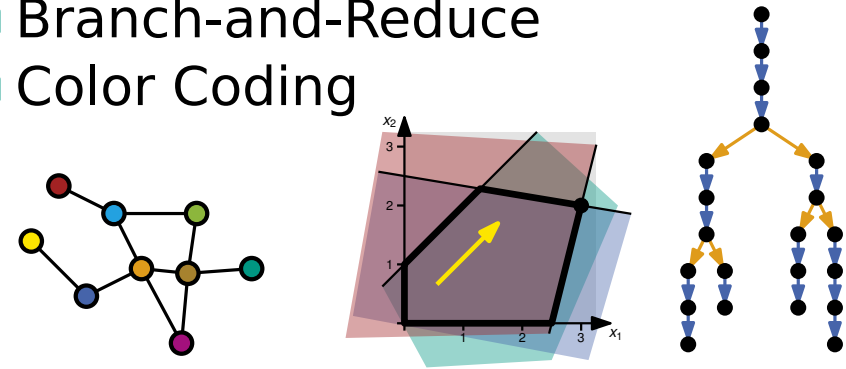
Basic Toolbox

- beschränkte Suchbäume
- iterative Kompression
- Kernbildung



Erweiterte Toolbox

- lineare Programme
- Branch-and-Reduce
- Color Coding



Baumweite

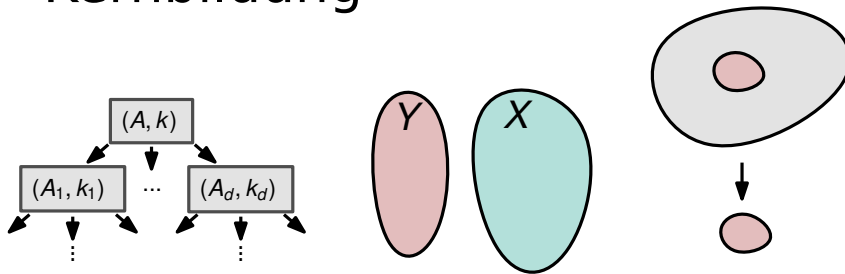
- dynamische Programme
- chordale & planare Graphen
- Courcelles Theorem



Inhalt

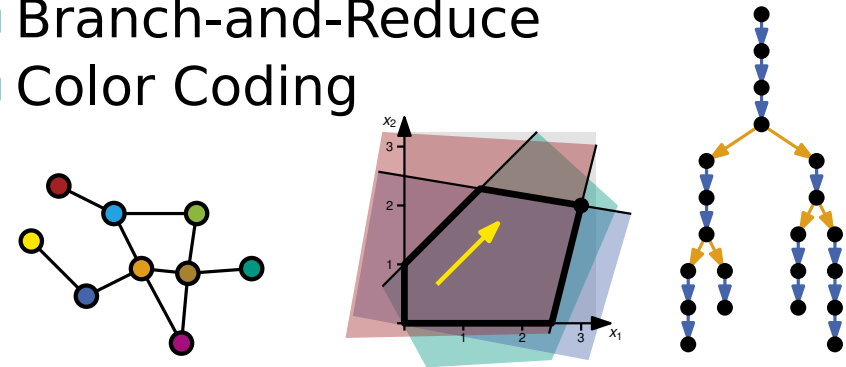
Basic Toolbox

- beschränkte Suchbäume
- iterative Kompression
- Kernbildung



Erweiterte Toolbox

- lineare Programme
- Branch-and-Reduce
- Color Coding



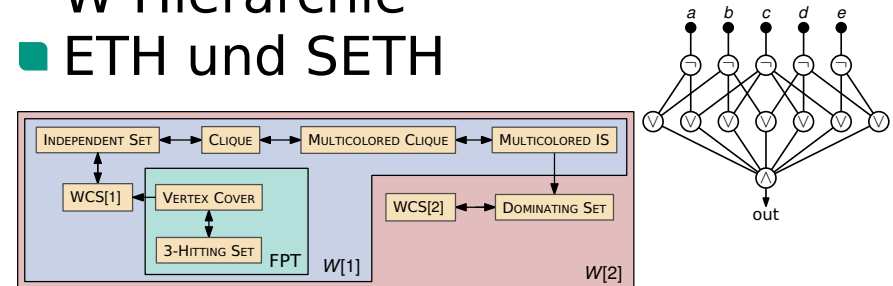
Baumweite

- dynamische Programme
- chordale & planare Graphen
- Courcelles Theorem



Untere Schranken

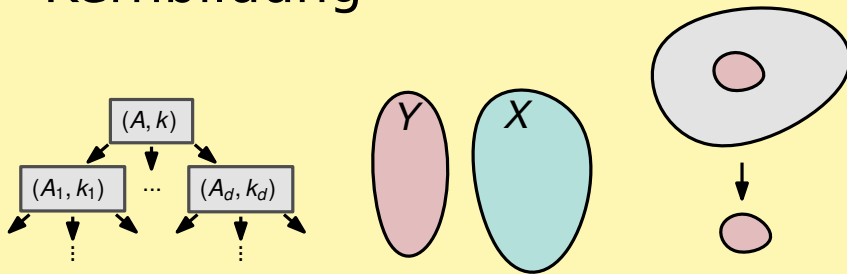
- parametrisierte Reduktionen
- boolesche Schaltkreise und die W-Hierarchie
- ETH und SETH



Inhalt

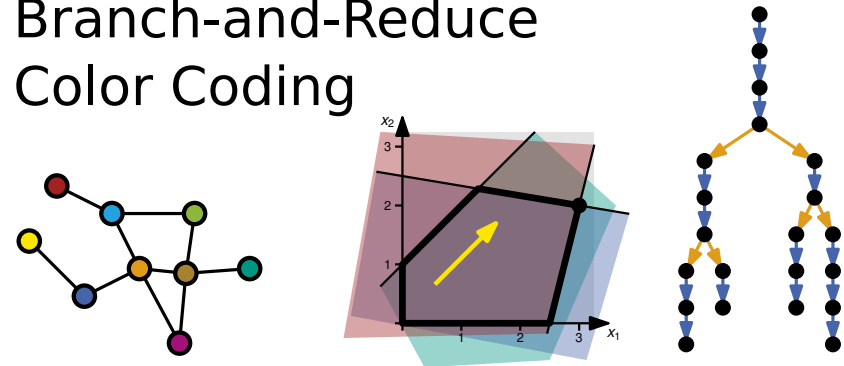
Basic Toolbox

- beschränkte Suchbäume
- iterative Kompression
- Kernbildung



Erweiterte Toolbox

- lineare Programme
- Branch-and-Reduce
- Color Coding



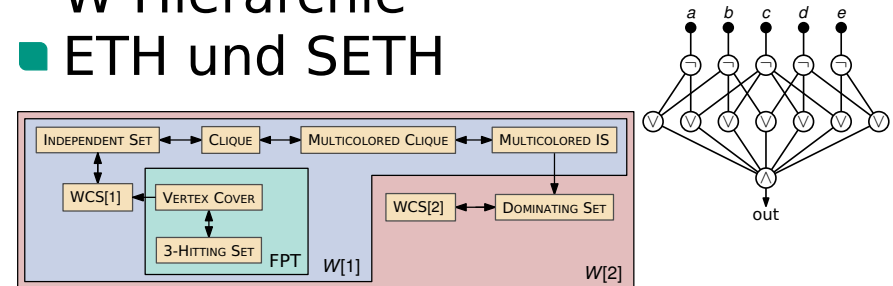
Baumweite

- dynamische Programme
- chordale & planare Graphen
- Courcelles Theorem



Untere Schranken

- parametrisierte Reduktionen
- boolesche Schaltkreise und die W-Hierarchie
- ETH und SETH



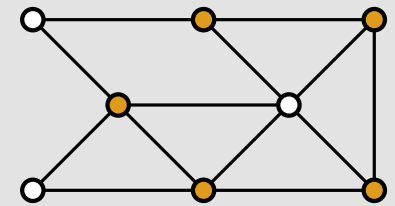
Beschränkter Suchbaum

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .

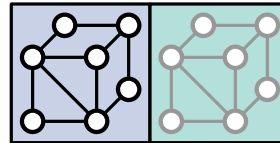
Gibt es ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



noch zu überdeckender Teilgraph

gewählte Knoten & überdeckte Kanten



Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?

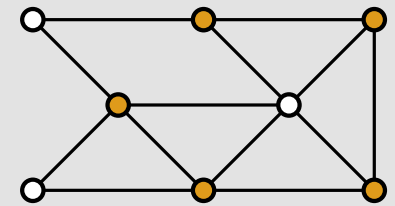
Beschränkter Suchbaum

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .

Gibt es ein Vertex Cover der Größe k ?

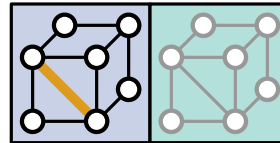
(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



noch zu überdeckender Teilgraph

gewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden \rightarrow wähle eine beliebige

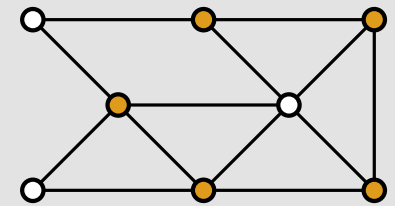


Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?

Beschränkter Suchbaum

Problem: VERTEX COVER

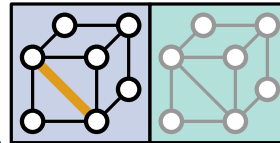
Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



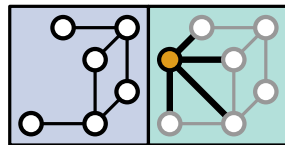
noch zu überdeckender Teilgraph

gewählte Knoten & überdeckte Kanten

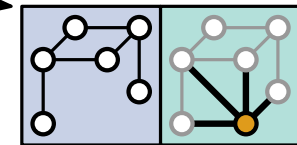
Jede Kante muss noch überdeckt werden \rightarrow wähle eine beliebige



Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?



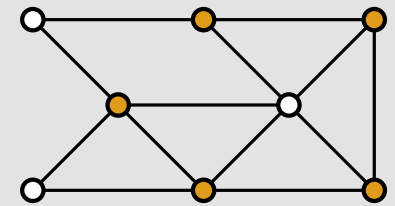
für Kante $\{u, v\}$ muss u oder v enthalten sein
 \rightarrow binäre Entscheidung



Beschränkter Suchbaum

Problem: VERTEX COVER

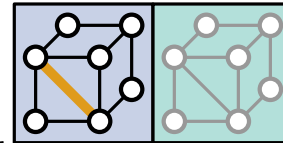
Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



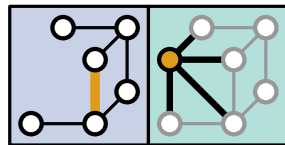
noch zu überdeckender Teilgraph

gewählte Knoten & überdeckte Kanten

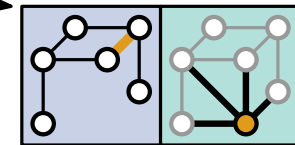
Jede Kante muss noch überdeckt werden \rightarrow wähle eine beliebige



Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?



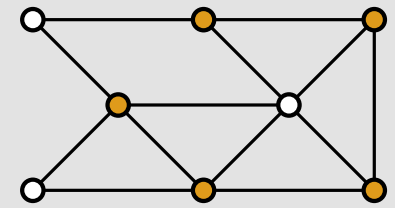
für Kante $\{u, v\}$ muss u oder v enthalten sein
 \rightarrow binäre Entscheidung



Beschränkter Suchbaum

Problem: VERTEX COVER

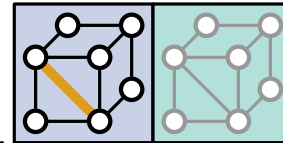
Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



noch zu überdeckender Teilgraph

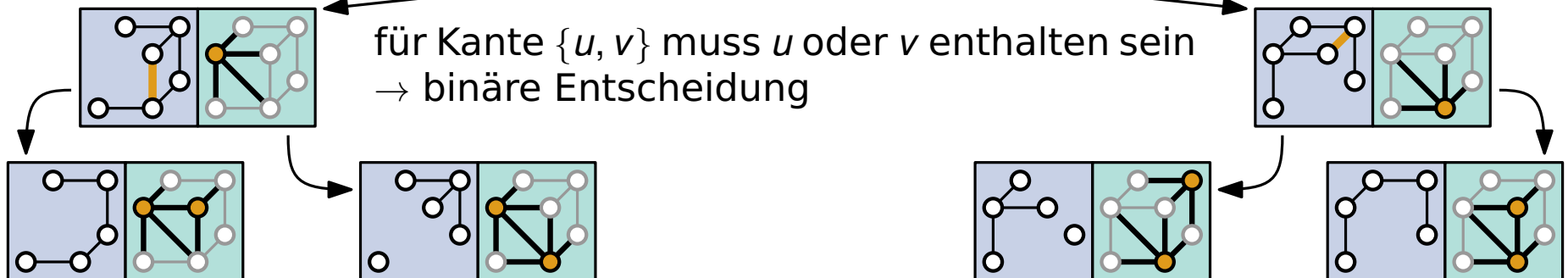
gewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden \rightarrow wähle eine beliebige



Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?

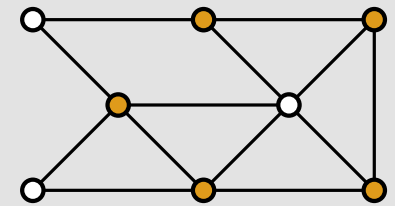
für Kante $\{u, v\}$ muss u oder v enthalten sein
 \rightarrow binäre Entscheidung



Beschränkter Suchbaum

Problem: VERTEX COVER

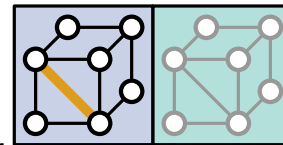
Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



noch zu überdeckender Teilgraph

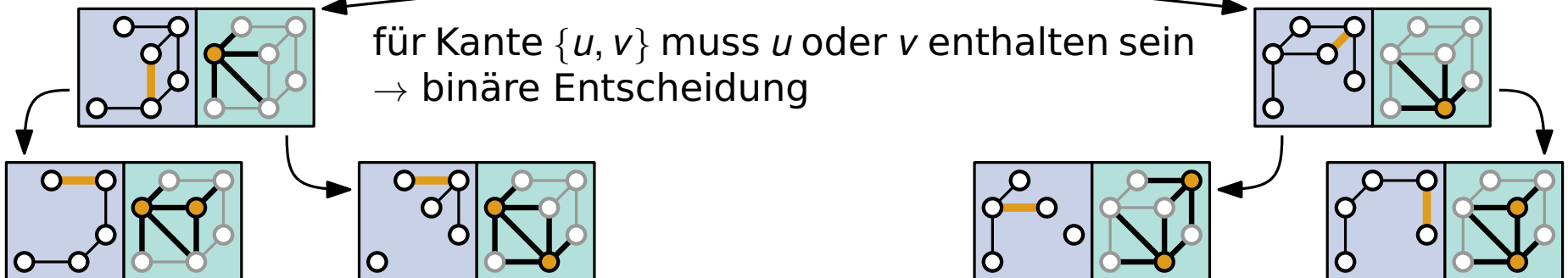
gewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden \rightarrow wähle eine beliebige



Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?

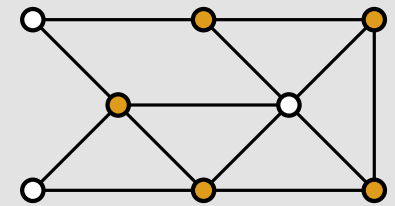
für Kante $\{u, v\}$ muss u oder v enthalten sein
 \rightarrow binäre Entscheidung



Beschränkter Suchbaum

Problem: VERTEX COVER

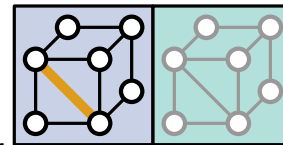
Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



noch zu überdeckender Teilgraph

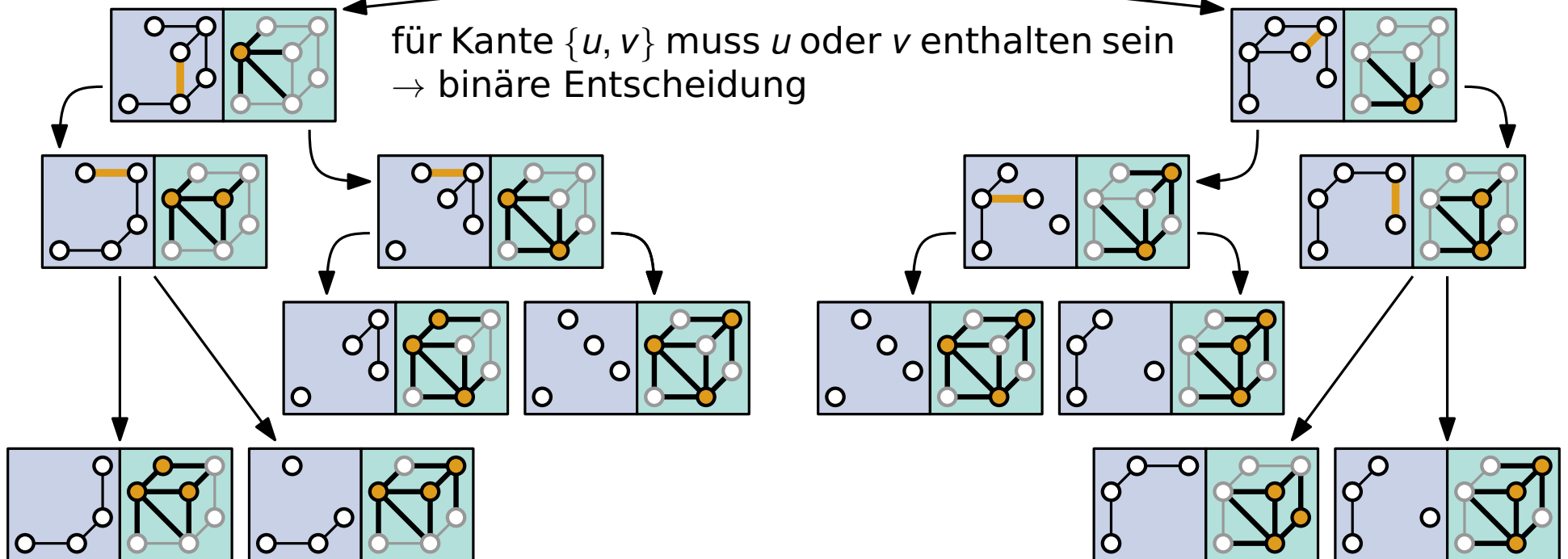
gewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden \rightarrow wähle eine beliebige



Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?

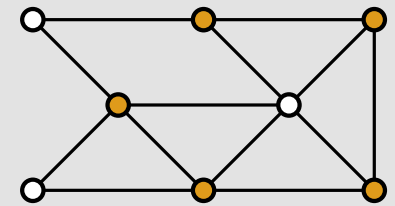
für Kante $\{u, v\}$ muss u oder v enthalten sein \rightarrow binäre Entscheidung



Beschränkter Suchbaum

Problem: VERTEX COVER

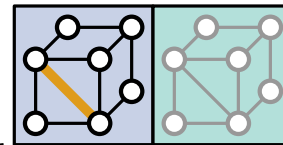
Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



noch zu überdeckender Teilgraph

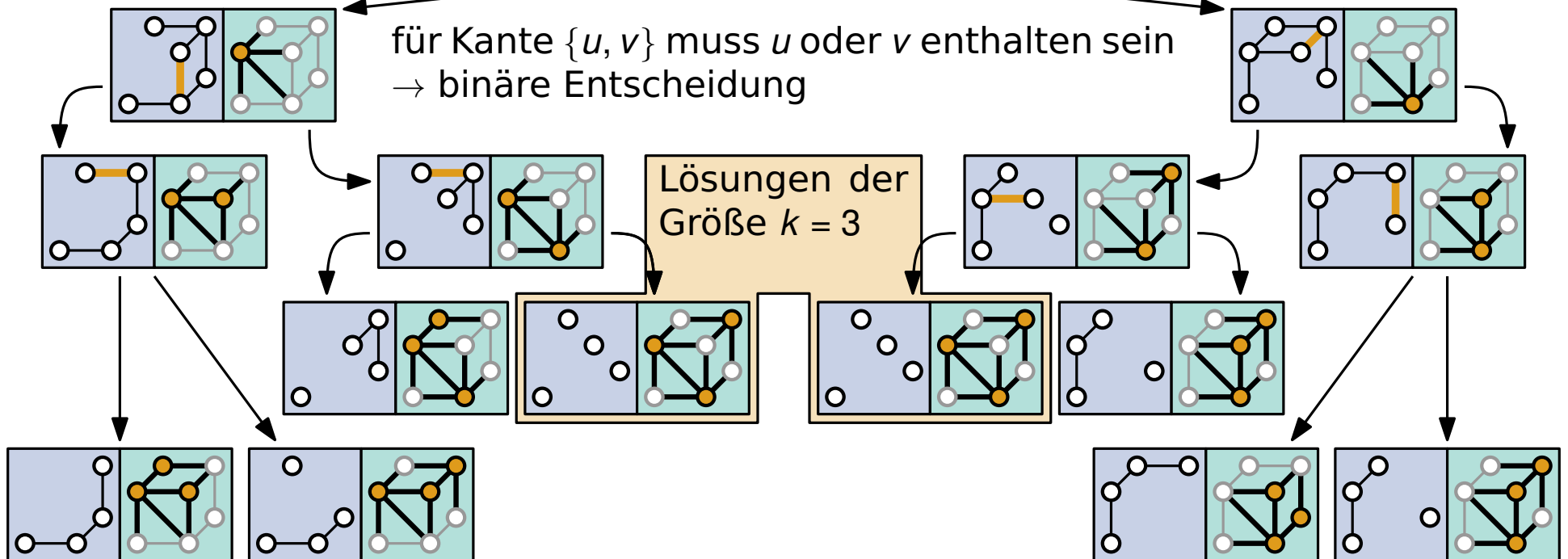
gewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden \rightarrow wähle eine beliebige



Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?

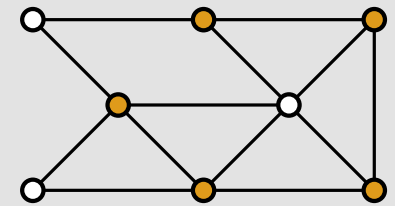
für Kante $\{u, v\}$ muss u oder v enthalten sein \rightarrow binäre Entscheidung



Beschränkter Suchbaum

Problem: VERTEX COVER

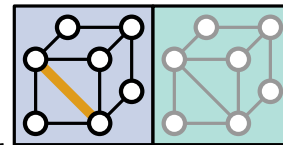
Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



noch zu überdeckender Teilgraph

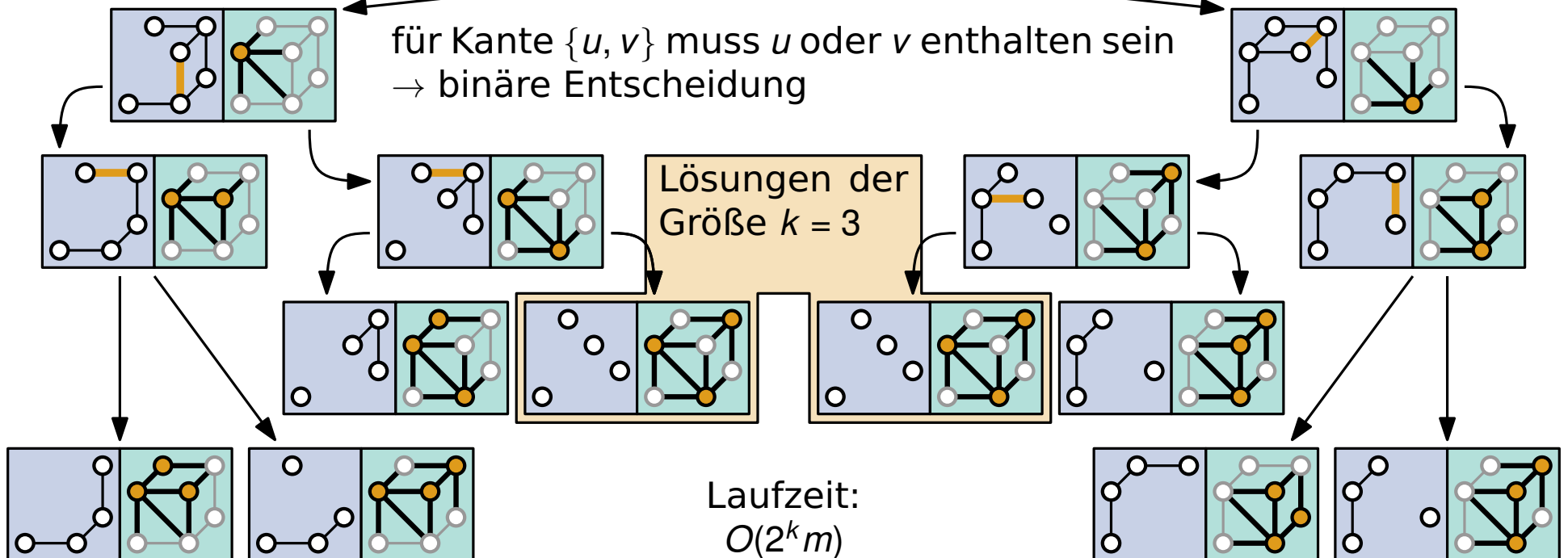
gewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden \rightarrow wähle eine beliebige



Gibt es ein Vertex Cover mit maximal $k = 3$ Knoten?

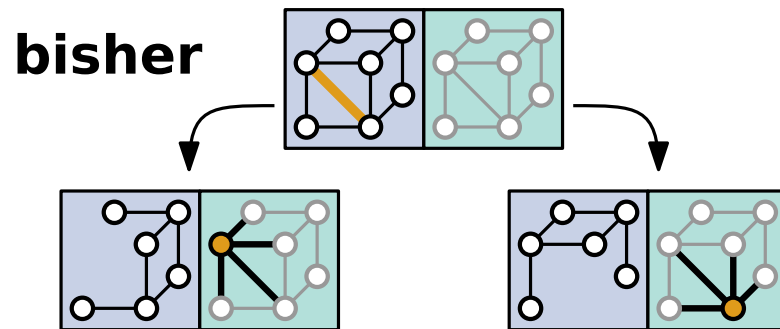
für Kante $\{u, v\}$ muss u oder v enthalten sein \rightarrow binäre Entscheidung



Beschränkterer Suchbaum

Geht es schneller?

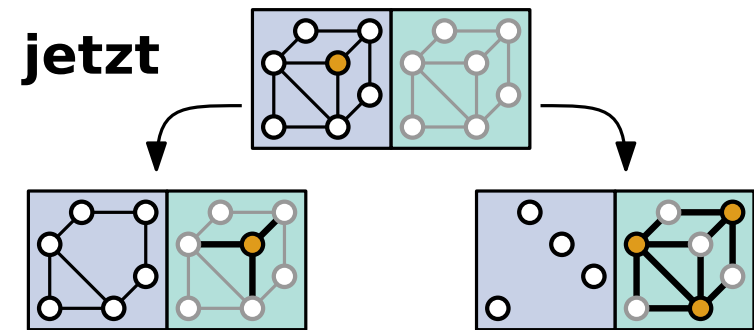
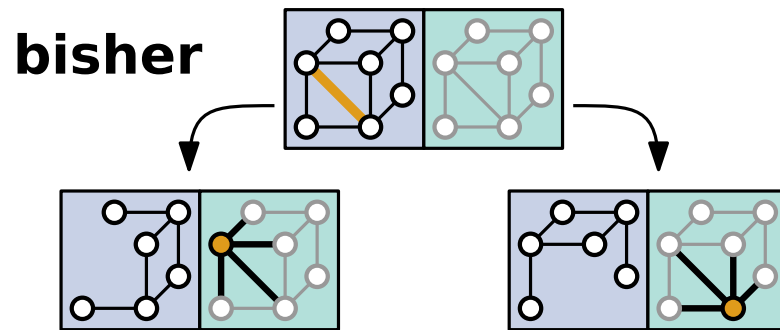
- Ziel: reduziere die Größe des Suchbaums (bisher: 2^k Blätter)
- Verzweigungsregel bisher: für Kante $\{u, v\}$ wähle entweder u oder v



Beschränkterer Suchbaum

Geht es schneller?

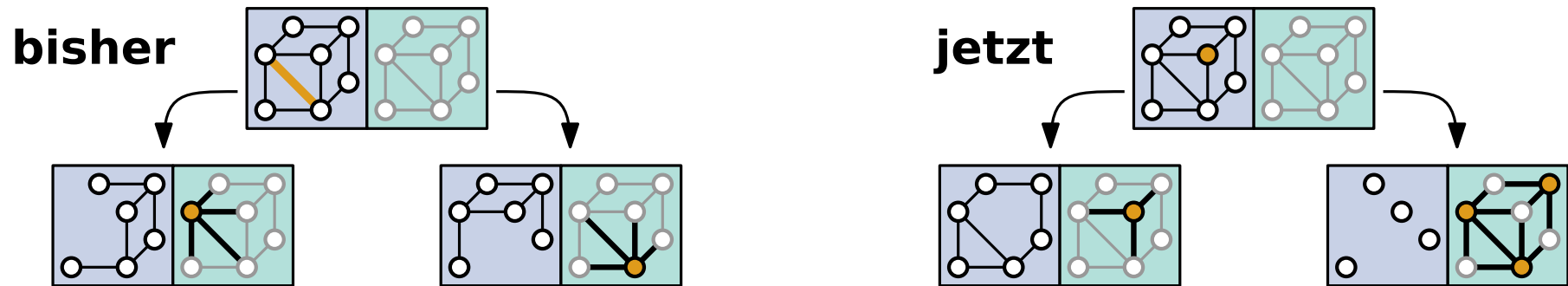
- Ziel: reduziere die Größe des Suchbaums (bisher: 2^k Blätter)
- Verzweigungsregel bisher: für Kante $\{u, v\}$ wähle entweder u oder v
- neue Verzweigungsregel: für Knoten v wähle entweder v oder $N(v)$



Beschränkterer Suchbaum

Geht es schneller?

- Ziel: reduziere die Größe des Suchbaums (bisher: 2^k Blätter)
- Verzweigungsregel bisher: für Kante $\{u, v\}$ wähle entweder u oder v
- neue Verzweigungsregel: für Knoten v wähle entweder v oder $N(v)$

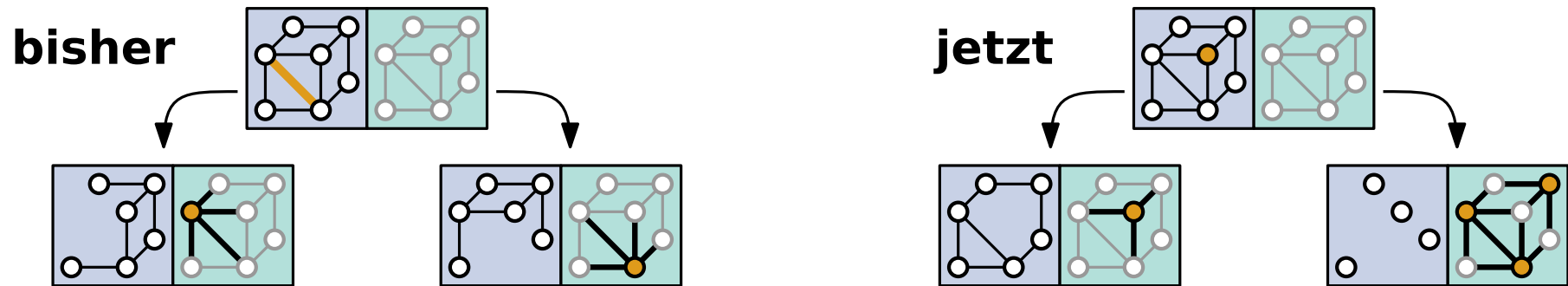


Wie viele Blätter hat der resultierende Baum abhängig von k ?

Beschränkterer Suchbaum

Geht es schneller?

- Ziel: reduziere die Größe des Suchbaums (bisher: 2^k Blätter)
- Verzweigungsregel bisher: für Kante $\{u, v\}$ wähle entweder u oder v
- neue Verzweigungsregel: für Knoten v wähle entweder v oder $N(v)$



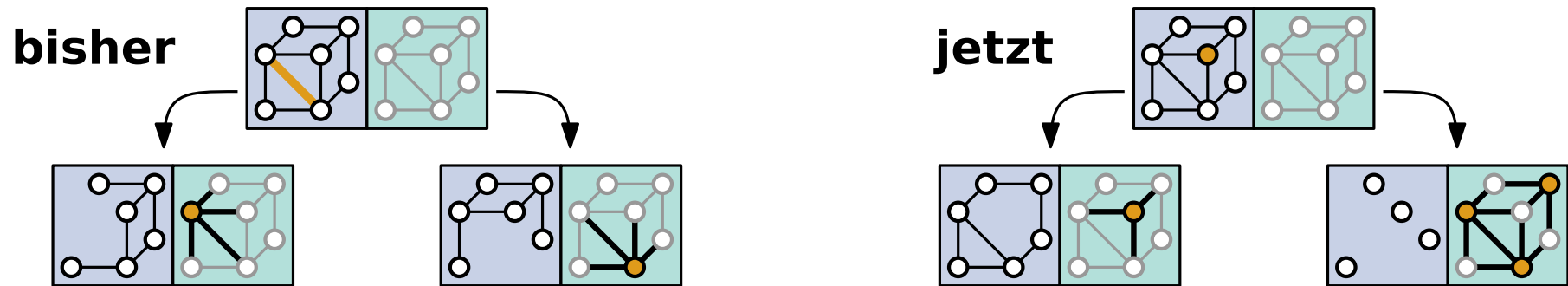
Wie viele Blätter hat der resultierende Baum abhängig von k ?

- wähle für v immer einen Knoten mit $\text{Grad} \geq 2$

Beschränkterer Suchbaum

Geht es schneller?

- Ziel: reduziere die Größe des Suchbaums (bisher: 2^k Blätter)
- Verzweigungsregel bisher: für Kante $\{u, v\}$ wähle entweder u oder v
- neue Verzweigungsregel: für Knoten v wähle entweder v oder $N(v)$



Wie viele Blätter hat der resultierende Baum abhängig von k ?

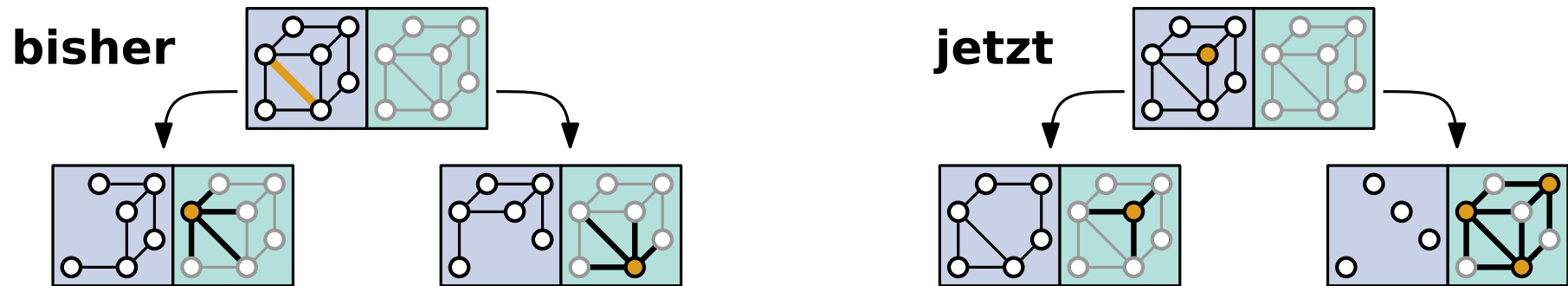
- wähle für v immer einen Knoten mit $\text{Grad} \geq 2$

- obere Schranke für #Blätter:
$$T(k) = \begin{cases} T(k-1) + T(k-2), & \text{für } k \geq 2 \\ 1, & \text{sonst} \end{cases}$$

Beschränkterer Suchbaum

Geht es schneller?

- Ziel: reduziere die Größe des Suchbaums (bisher: 2^k Blätter)
- Verzweigungsregel bisher: für Kante $\{u, v\}$ wähle entweder u oder v
- neue Verzweigungsregel: für Knoten v wähle entweder v oder $N(v)$



Wie viele Blätter hat der resultierende Baum abhängig von k ?

- wähle für v immer einen Knoten mit $\text{Grad} \geq 2$

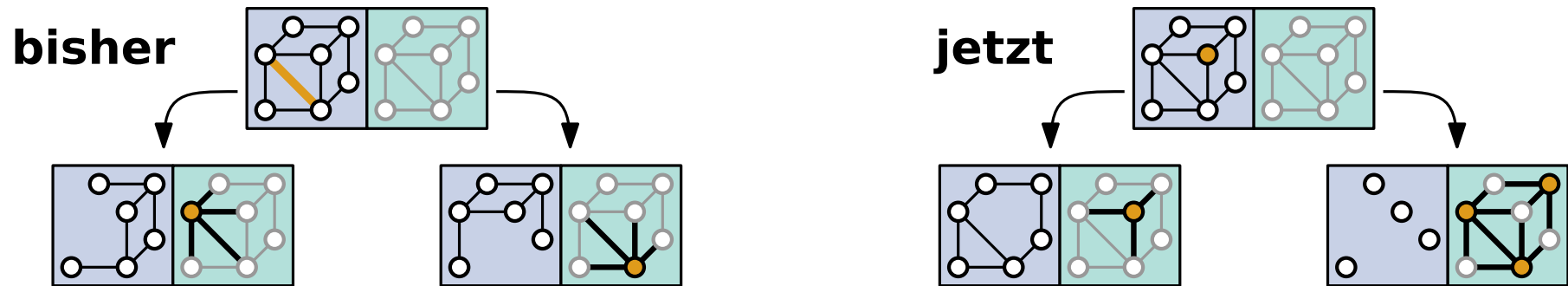
- obere Schranke für #Blätter: $T(k) = \begin{cases} T(k-1) + T(k-2), & \text{für } k \geq 2 \\ 1, & \text{sonst} \end{cases}$

- wir erhalten: $T(k) \leq 1,6181^k$

Beschränkterer Suchbaum

Geht es schneller?

- Ziel: reduziere die Größe des Suchbaums (bisher: 2^k Blätter)
- Verzweigungsregel bisher: für Kante $\{u, v\}$ wähle entweder u oder v
- neue Verzweigungsregel: für Knoten v wähle entweder v oder $N(v)$



Wie viele Blätter hat der resultierende Baum abhängig von k ?

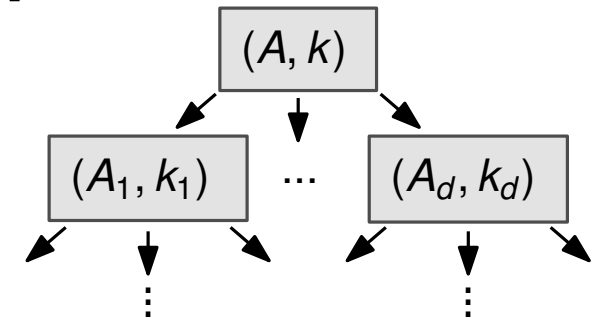
- wähle für v immer einen Knoten mit $\text{Grad} \geq 2$
- obere Schranke für #Blätter:
$$T(k) = \begin{cases} T(k-1) + T(k-2), & \text{für } k \geq 2 \\ 1, & \text{sonst} \end{cases}$$
- wir erhalten: $T(k) \leq 1,6181^k$
- Wie kommt man auf 1,6181? Geht es besser? \longrightarrow nächste Vorlesung

Grundlegende Techniken

Beschränkter Suchbaum

(Bounded Search Tree)

- für eine Instanz (A, k) bilde, in Zeit n^c , $(A_1, k_1), \dots, (A_d, k_d)$, sodass:
 (A, k) ist lösbar $\Leftrightarrow (A_i, k_i)$ ist lösbar für ein $i \in [1, d]$

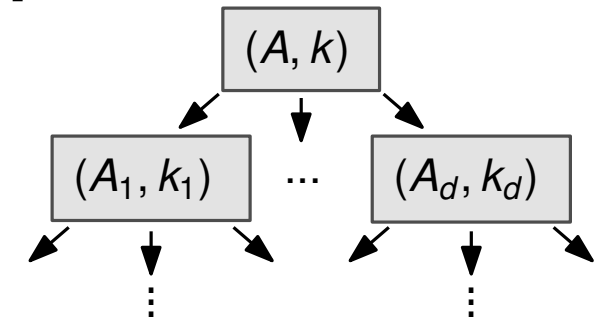


Grundlegende Techniken

Beschränkter Suchbaum

(Bounded Search Tree)

- für eine Instanz (A, k) bilde, in Zeit n^c , $(A_1, k_1), \dots, (A_d, k_d)$, sodass:
 (A, k) ist lösbar $\Leftrightarrow (A_i, k_i)$ ist lösbar für ein $i \in [1, d]$
- beschränke d durch eine Funktion $f_1(k)$

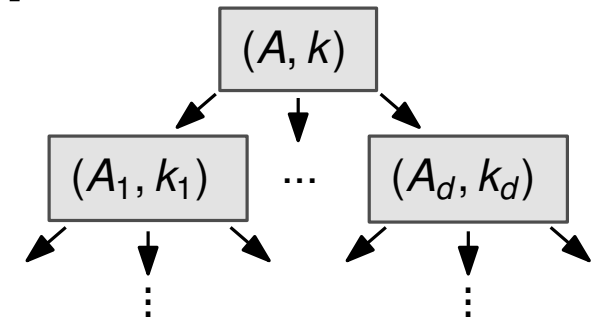


Grundlegende Techniken

Beschränkter Suchbaum

(Bounded Search Tree)

- für eine Instanz (A, k) bilde, in Zeit n^c , $(A_1, k_1), \dots, (A_d, k_d)$, sodass:
 (A, k) ist lösbar $\Leftrightarrow (A_i, k_i)$ ist lösbar für ein $i \in [1, d]$
- beschränke d durch eine Funktion $f_1(k)$
- beschränke Verzweigungstiefe durch $f_2(k)$
 (Beispiel: Parameter wird in jedem Schritt kleiner)

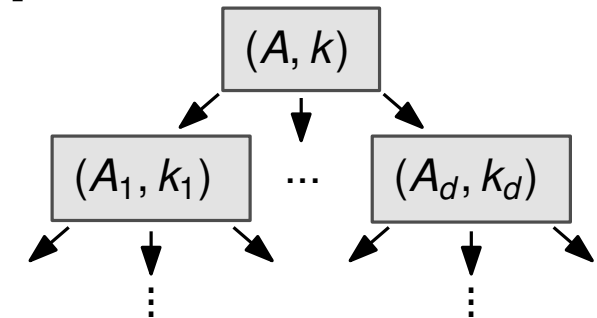


Grundlegende Techniken

Beschränkter Suchbaum

(Bounded Search Tree)

- für eine Instanz (A, k) bilde, in Zeit n^c , $(A_1, k_1), \dots, (A_d, k_d)$, sodass:
 (A, k) ist lösbar $\Leftrightarrow (A_i, k_i)$ ist lösbar für ein $i \in [1, d]$
- beschränke d durch eine Funktion $f_1(k)$
- beschränke Verzweigungstiefe durch $f_2(k)$
 (Beispiel: Parameter wird in jedem Schritt kleiner)
- \Rightarrow FPT-Algorithmus mit Laufzeit $O(f_1(k)^{f_2(k)} n^c)$



Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

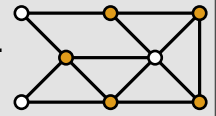
- lösche einen isolierten Knoten

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .

Gibt es ein Vertex Cover der Größe k ?

(Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

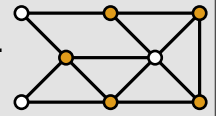
- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

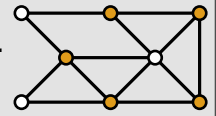
- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Lemma

Reduktionsregel 1 ist sicher.

(neue Instanz lösbar \Leftrightarrow alte Instanz lösbar)

Beweis: offensichtlich

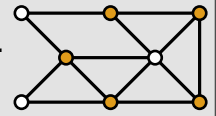
Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Lemma

Reduktionsregel 2 ist sicher.

(neue Instanz lösbar \Leftrightarrow alte Instanz lösbar)

Beweis

- entweder v oder jeder Nachbar von v muss im VC enthalten sein
- wählt man v nicht, so wird das VC sicher zu groß ($\deg(v) > k$)

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

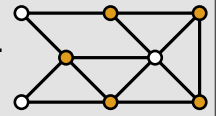
Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

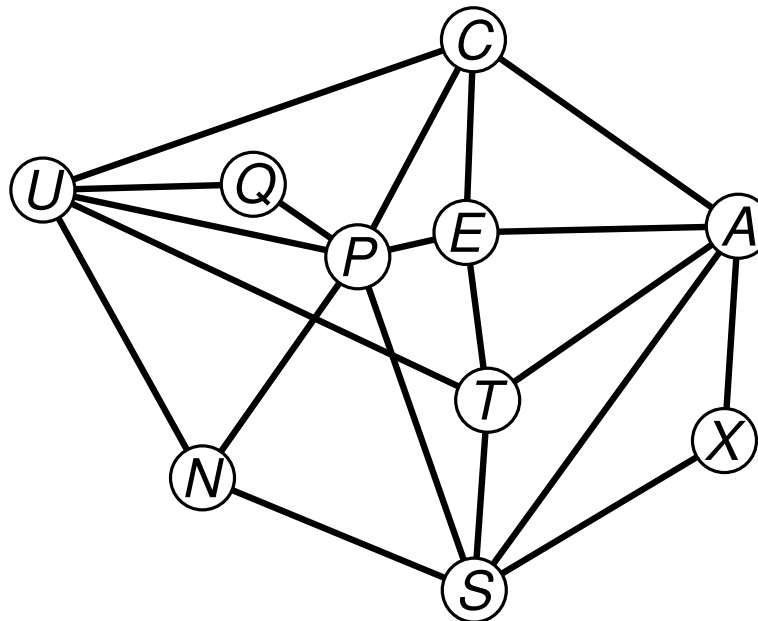
Wende Regel 2 so oft an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



$k = 5$



Lösungswort:

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

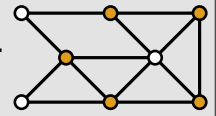
Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

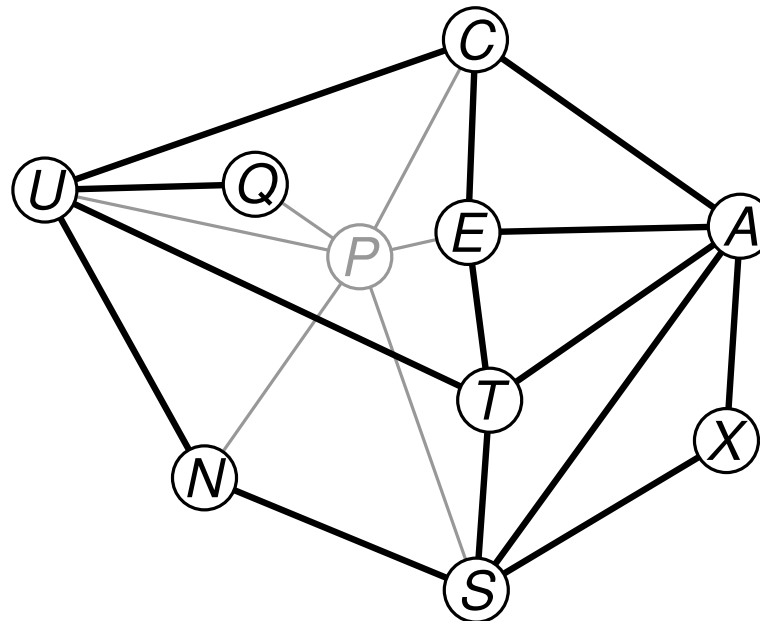
Wende Regel 2 so oft an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



$k = 4$



Lösungswort: \textcircled{P}

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

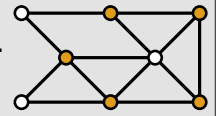
Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

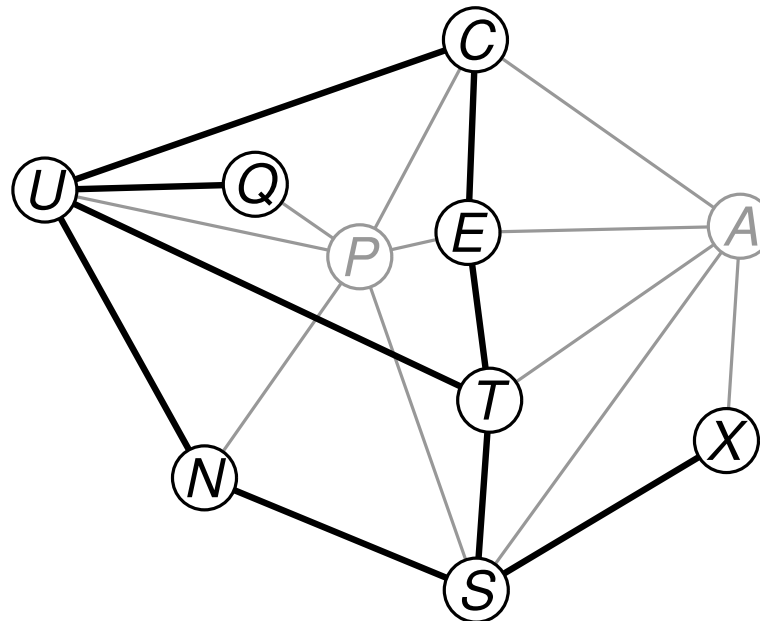
Wende Regel 2 so oft an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



$k = 3$



Lösungswort: (P) (A)

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

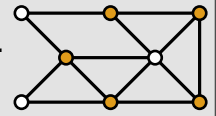
Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

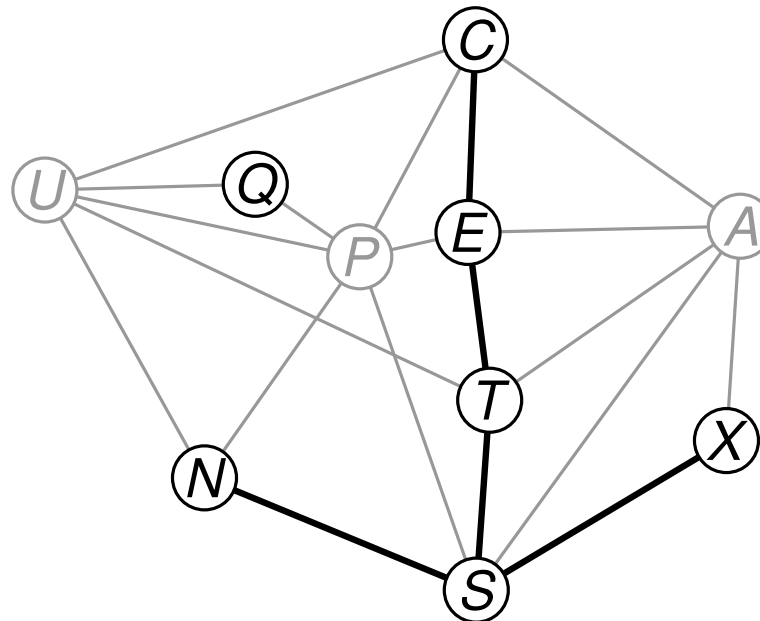
Wende Regel 2 so oft an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



$k = 2$



Lösungswort: (P) (A) (U)

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

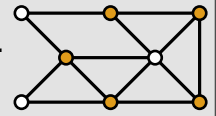
Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

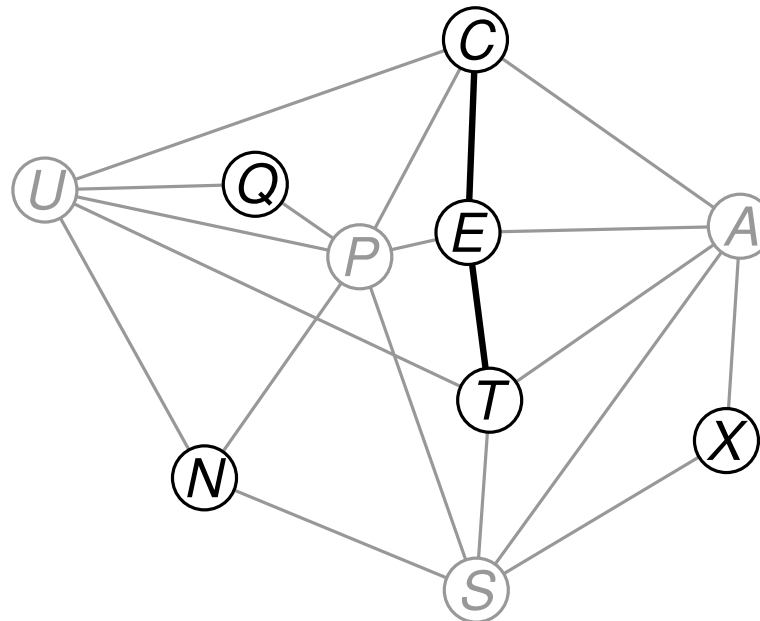
Wende Regel 2 so oft an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



$k = 1$



Lösungswort: (P) (A) (U) (S)

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

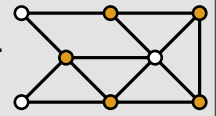
Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

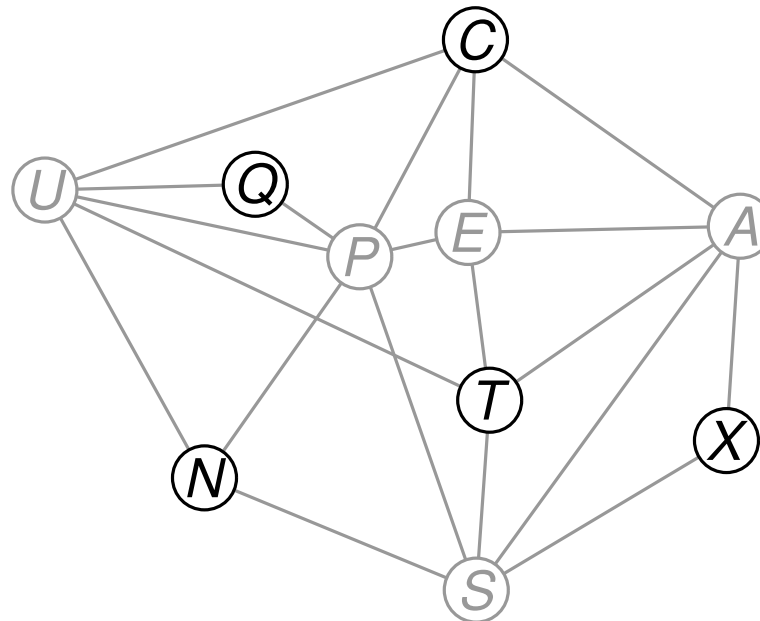
Wende Regel 2 so oft an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



$k = 0$



Lösungswort: (P) (A) (U) (S) (E)

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

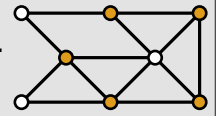
- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

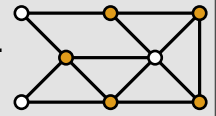
- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---} \circ \text{---} \circ$ (konstant große NEIN-Instanz)

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Lemma

Reduktionsregel 3 ist sicher. (neue Instanz lösbar \Leftrightarrow alte Instanz lösbar)

Beweis

- Regel 2 nicht anwendbar $\Rightarrow \deg(v) \leq k$ für alle $v \in V$

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

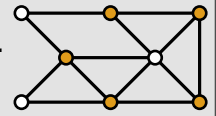
- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Lemma

Reduktionsregel 3 ist sicher. (neue Instanz lösbar \Leftrightarrow alte Instanz lösbar)

Beweis

- Regel 2 nicht anwendbar $\Rightarrow \deg(v) \leq k$ für alle $v \in V$
- jeder Knoten deckt maximal k Kanten ab

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

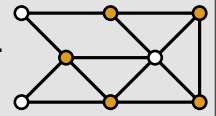
- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Lemma

Reduktionsregel 3 ist sicher. (neue Instanz lösbar \Leftrightarrow alte Instanz lösbar)

Beweis

- Regel 2 nicht anwendbar $\Rightarrow \deg(v) \leq k$ für alle $v \in V$
- jeder Knoten deckt maximal k Kanten ab
- k Knoten decken maximal k^2 Kanten ab

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

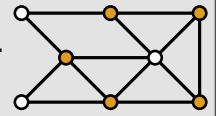
- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Lemma

Reduktionsregel 3 ist sicher. (neue Instanz lösbar \Leftrightarrow alte Instanz lösbar)

Beweis

- Regel 2 nicht anwendbar $\Rightarrow \deg(v) \leq k$ für alle $v \in V$
- jeder Knoten deckt maximal k Kanten ab
- k Knoten decken maximal k^2 Kanten ab
- Instanz nicht lösbar, da $m > k^2$

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

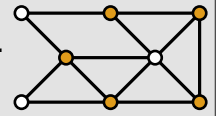
- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Kernbildung

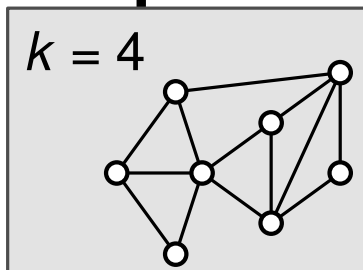
- wende Reduktionsregel so lange an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Beispiel



Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

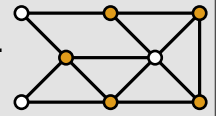
- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Kernbildung

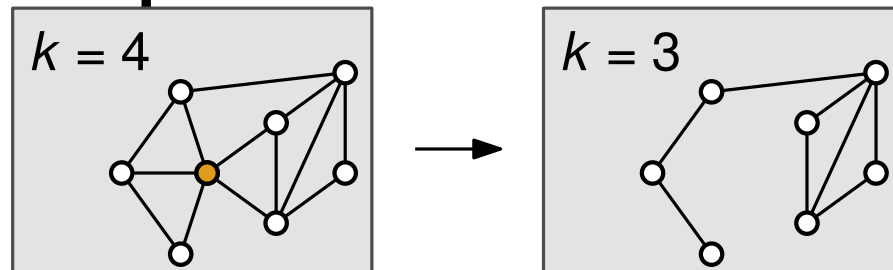
- wende Reduktionsregel so lange an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Beispiel



Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

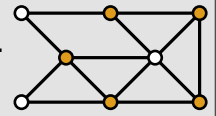
- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Kernbildung

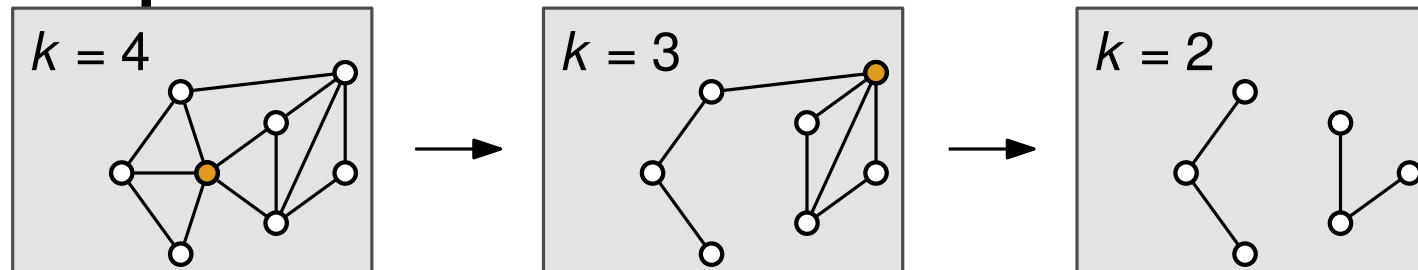
- wende Reduktionsregel so lange an, wie möglich

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Beispiel



Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

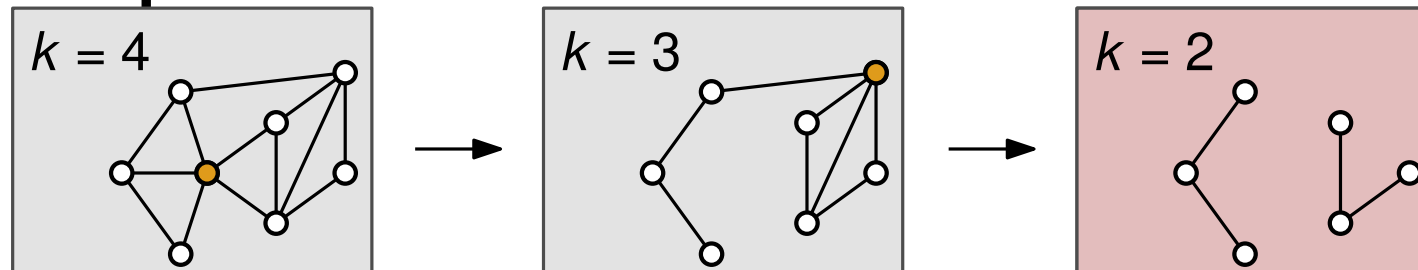
Reduktionsregel 3

- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Kernbildung

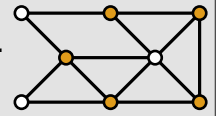
- wende Reduktionsregel so lange an, wie möglich
- übrig bleibt der **Problemkern** der Größe $O(k^2) \rightarrow$ Brute-Force

Beispiel



Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

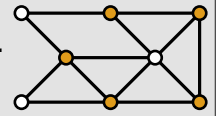
- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Kernbildung

- wende Reduktionsregel so lange an, wie möglich
- übrig bleibt der **Problemkern** der Größe $O(k^2) \rightarrow$ Brute-Force

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Theorem

VERTEX COVER hat einen Kern mit $O(k^2)$ Knoten und $O(k^2)$ Kanten. Er kann in $O(m)$ Zeit berechnet werden.

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

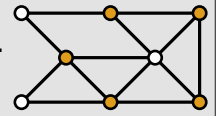
- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Kernbildung

- wende Reduktionsregel so lange an, wie möglich
- übrig bleibt der **Problemkern** der Größe $O(k^2) \rightarrow$ Brute-Force

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Theorem

VERTEX COVER hat einen Kern mit $O(k^2)$ Knoten und $O(k^2)$ Kanten. Er kann in $O(m)$ Zeit berechnet werden.

- Brute-Force auf dem Kern: $O(2^{k^2} k^2)$

Kernbildung am Beispiel Vertex Cover

Reduktionsregel 1

- lösche einen isolierten Knoten

Reduktionsregel 2

- füge einen Knoten v mit $\deg(v) > k$ zum VC hinzu
- neue Instanz: lösche v , verringere k um 1

Reduktionsregel 3

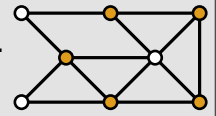
- falls Regel 1 und Regel 2 nicht anwendbar sind und $m > k^2$
- neue Instanz: $(H, 0)$, mit $H = \text{---}$ (konstant große NEIN-Instanz)

Kernbildung

- wende Reduktionsregel so lange an, wie möglich
- übrig bleibt der **Problemkern** der Größe $O(k^2) \rightarrow$ Brute-Force

Problem: VERTEX COVER

Gegeben sind ein Graph $G = (V, E)$ und ein Parameter k .
 Gibt es ein Vertex Cover der Größe k ?
 (Knotenmenge $V' \subseteq V$ mit $e \cap V' \neq \emptyset$ für alle $e \in E$)



Theorem

VERTEX COVER hat einen Kern mit $O(k^2)$ Knoten und $O(k^2)$ Kanten. Er kann in $O(m)$ Zeit berechnet werden.

- Brute-Force auf dem Kern: $O(2^{k^2} k^2)$ (besser: Suchbaum auf dem Kern)

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x, k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x', k') (den **Kern**) überführt, sodass:

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Theorem

Ein entscheidbares parametrisiertes Problem L ist in FPT genau dann wenn es eine Kernbildung für L gibt.

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Theorem

Ein entscheidbares parametrisiertes Problem L ist in FPT genau dann wenn es eine Kernbildung für L gibt.

Welche Richtung ist leicht zu zeigen?

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Theorem

Ein entscheidbares parametrisiertes Problem L ist in FPT genau dann wenn es eine Kernbildung für L gibt.

Beweis (der anderen Richtung)

- sei \mathcal{A} ein Algorithmus, der $(x,k) \in L$ in $f(k)|x|^c$ entscheidet

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Theorem

Ein entscheidbares parametrisiertes Problem L ist in FPT genau dann wenn es eine Kernbildung für L gibt.

Beweis (der anderen Richtung)

- sei \mathcal{A} ein Algorithmus, der $(x,k) \in L$ in $f(k)|x|^c$ entscheidet
- lasse \mathcal{A} auf (x,k) für $|x|^{c+1}$ Schritten laufen

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Theorem

Ein entscheidbares parametrisiertes Problem L ist in FPT genau dann wenn es eine Kernbildung für L gibt.

Beweis (der anderen Richtung)

- sei \mathcal{A} ein Algorithmus, der $(x,k) \in L$ in $f(k)|x|^c$ entscheidet
- lasse \mathcal{A} auf (x,k) für $|x|^{c+1}$ Schritten laufen
- falls \mathcal{A} fertig wird: Problem gelöst \rightarrow trivialer Kern

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Theorem

Ein entscheidbares parametrisiertes Problem L ist in FPT genau dann wenn es eine Kernbildung für L gibt.

Beweis (der anderen Richtung)

- sei \mathcal{A} ein Algorithmus, der $(x,k) \in L$ in $f(k)|x|^c$ entscheidet
- lasse \mathcal{A} auf (x,k) für $|x|^{c+1}$ Schritten laufen
- falls \mathcal{A} fertig wird: Problem gelöst \rightarrow trivialer Kern
- sonst: $|x|^{c+1} < f(k)|x|^c$

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Theorem

Ein entscheidbares parametrisiertes Problem L ist in FPT genau dann wenn es eine Kernbildung für L gibt.

Beweis (der anderen Richtung)

- sei \mathcal{A} ein Algorithmus, der $(x,k) \in L$ in $f(k)|x|^c$ entscheidet
- lasse \mathcal{A} auf (x,k) für $|x|^{c+1}$ Schritten laufen
- falls \mathcal{A} fertig wird: Problem gelöst \rightarrow trivialer Kern
- sonst: $|x|^{c+1} < f(k)|x|^c \Rightarrow |x| < f(k)$

Kernbildung - Formale Definition

Definition

Eine **Kernbildung** für ein parametrisiertes Problem L ist ein Algorithmus, der jede Eingabe (x,k) in polynomieller Zeit (gemessen in $|x|$) in eine Eingabe (x',k') (den **Kern**) überführt, sodass:

- $(x,k) \in L$ genau dann wenn $(x',k') \in L$, und
- $|x'| + k' \leq g(k)$ für eine von x unabhängige Funktion g .

Theorem

Ein entscheidbares parametrisiertes Problem L ist in FPT genau dann wenn es eine Kernbildung für L gibt.

Beweis (der anderen Richtung)

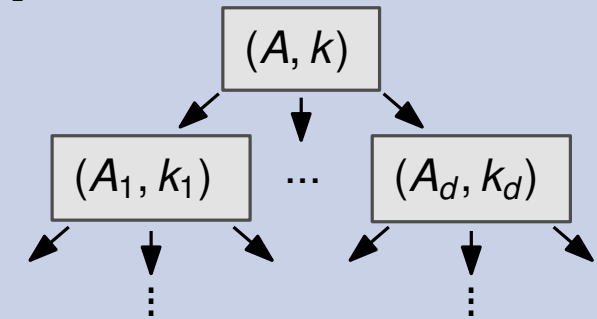
- sei \mathcal{A} ein Algorithmus, der $(x,k) \in L$ in $f(k)|x|^c$ entscheidet
- lasse \mathcal{A} auf (x,k) für $|x|^{c+1}$ Schritten laufen
- falls \mathcal{A} fertig wird: Problem gelöst \rightarrow trivialer Kern
- sonst: $|x|^{c+1} < f(k)|x|^c \Rightarrow |x| < f(k)$
- (x,k) selbst ist schon ein geeigneter Kern \rightarrow gib (x,k) zurück

Grundlegende Techniken

Beschränkter Suchbaum

(Bounded Search Tree)

- für eine Instanz (A, k) bilde, in Zeit n^c , $(A_1, k_1), \dots, (A_d, k_d)$, sodass:
 (A, k) ist lösbar $\Leftrightarrow (A_i, k_i)$ ist lösbar für ein $i \in [1, d]$
- beschränke d durch eine Funktion $f_1(k)$
- beschränke Verzweigungstiefe durch $f_2(k)$
 (Beispiel: Parameter wird in jedem Schritt kleiner)
- \Rightarrow FPT-Algo mit Laufzeit $O(f_1(k)^{f_2(k)} n^c)$



Kernbildung

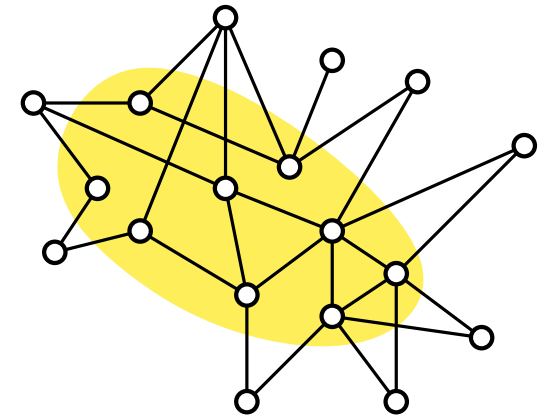
(Kernelization)

- wende sukzessive sichere Reduktionsregeln an
- zeige: übrig bleibt ein Kern, dessen Größe nur von k abhängt

VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?



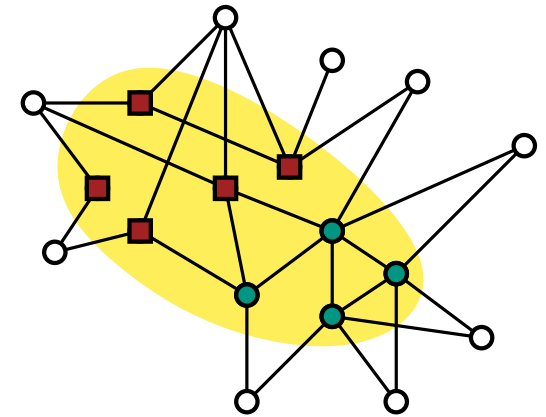
VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Idee

- rate eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?



VERTEX COVER COMPRESSION

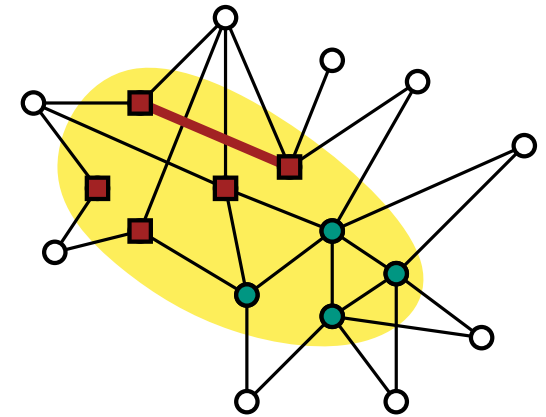
Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Idee

- rate eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?

Fall 1: $G[Y]$ enthält Kante $\Rightarrow Z'$ existiert nicht



VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

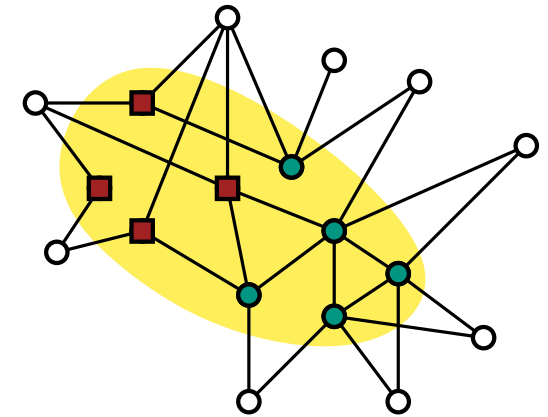
Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Idee

- rate eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?

Fall 1: $G[Y]$ enthält Kante $\Rightarrow Z'$ existiert nicht

Fall 2: sonst gilt:



VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

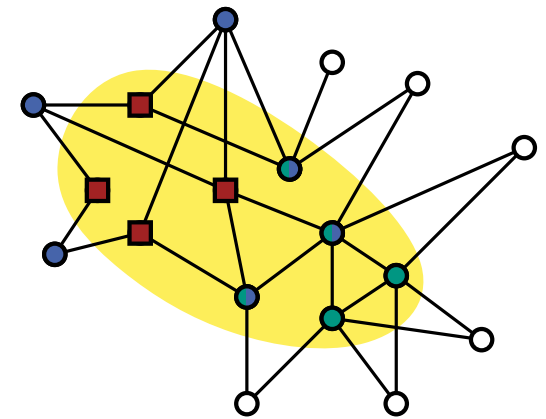
Idee

- rate eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?

Fall 1: $G[Y]$ enthält Kante $\Rightarrow Z'$ existiert nicht

Fall 2: sonst gilt:

- Z' muss alle Nachbarn $N(Y)$ von Y enthalten



VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

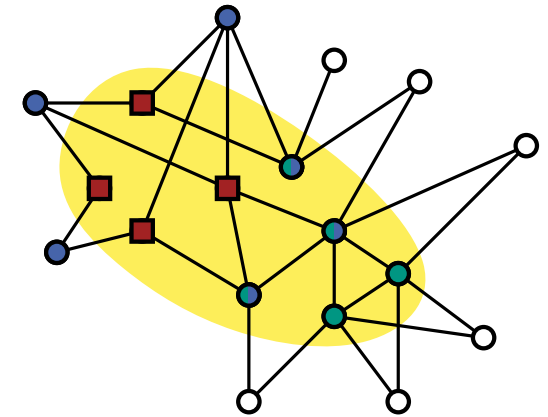
Idee

- rate eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?

Fall 1: $G[Y]$ enthält Kante $\Rightarrow Z'$ existiert nicht

Fall 2: sonst gilt:

- Z' muss alle Nachbarn $N(Y)$ von Y enthalten
- $X \cup N(Y)$ ist ein VC



VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

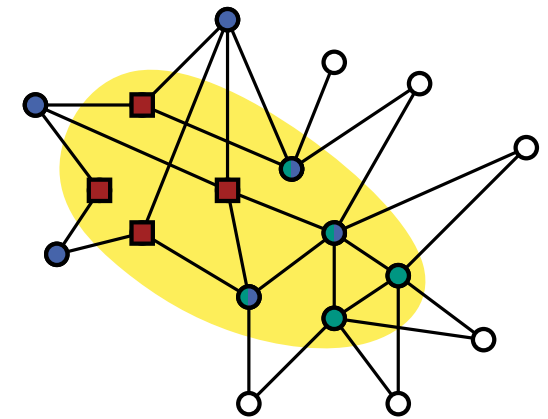
Idee

- rate eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?

Fall 1: $G[Y]$ enthält Kante $\Rightarrow Z'$ existiert nicht

Fall 2: sonst gilt:

- Z' muss alle Nachbarn $N(Y)$ von Y enthalten
- $X \cup N(Y)$ ist ein VC
- es gibt das gewünschte VC $Z' \Leftrightarrow |X \cup N(Y)| \leq k$



VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

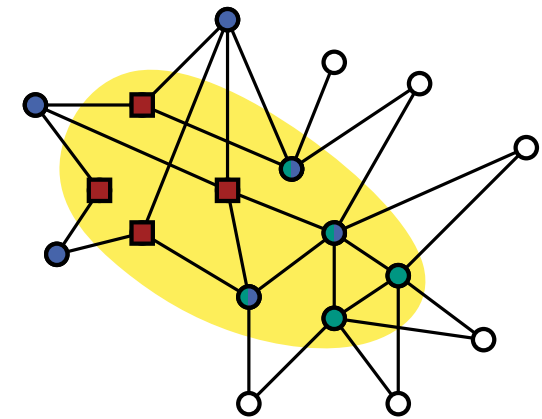
Idee

- wähle eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?

Fall 1: $G[Y]$ enthält Kante $\Rightarrow Z'$ existiert nicht

Fall 2: sonst gilt:

- Z' muss alle Nachbarn $N(Y)$ von Y enthalten
- $X \cup N(Y)$ ist ein VC
- es gibt das gewünschte VC $Z' \Leftrightarrow |X \cup N(Y)| \leq k$



Algorithmus

- wende obige Prozedur auf jede Teilmenge $X \subseteq Z$ an

VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Idee

- wähle eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?

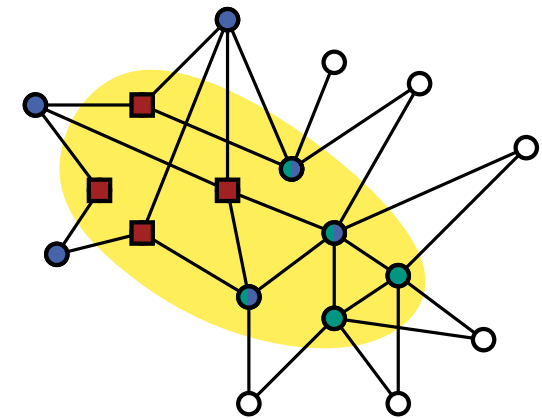
Fall 1: $G[Y]$ enthält Kante $\Rightarrow Z'$ existiert nicht

Fall 2: sonst gilt:

- Z' muss alle Nachbarn $N(Y)$ von Y enthalten
- $X \cup N(Y)$ ist ein VC
- es gibt das gewünschte VC $Z' \Leftrightarrow |X \cup N(Y)| \leq k$

Algorithmus

- wende obige Prozedur auf jede Teilmenge $X \subseteq Z$ an
- es gibt 2^{k+1} solche Teilmengen



VERTEX COVER COMPRESSION

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

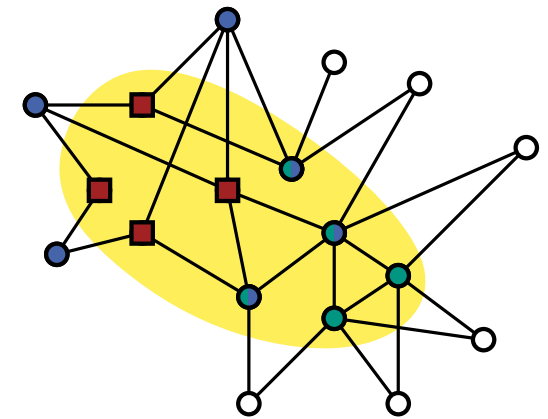
Idee

- rate eine Teilmenge $X \subseteq Z$; $Y = Z \setminus X$
- gibt es VC Z' mit $|Z'| \leq k$ und $Z' \cap Z = X$?

Fall 1: $G[Y]$ enthält Kante $\Rightarrow Z'$ existiert nicht

Fall 2: sonst gilt:

- Z' muss alle Nachbarn $N(Y)$ von Y enthalten
- $X \cup N(Y)$ ist ein VC
- es gibt das gewünschte VC $Z' \Leftrightarrow |X \cup N(Y)| \leq k$



Algorithmus

- wende obige Prozedur auf jede Teilmenge $X \subseteq Z$ an
- es gibt 2^{k+1} solche Teilmengen
- Laufzeit $O(m)$ für jedes $X \Rightarrow$ insgesamt $O(2^k m)$

Iterative Kompression

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Gerade gesehen: $O(2^k m)$ -Algorithmus für VERTEX COVER COMPRESSION

Iterative Kompression

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Gerade gesehen: $O(2^k m)$ -Algorithmus für VERTEX COVER COMPRESSION

Algorithmus für VERTEX COVER

- iteriere über die Subgraphen G_i bestehend aus den Knoten v_1, \dots, v_i

Iterative Kompression

Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Gerade gesehen: $O(2^k m)$ -Algorithmus für VERTEX COVER COMPRESSION

Algorithmus für VERTEX COVER

- iteriere über die Subgraphen G_i bestehend aus den Knoten v_1, \dots, v_i
- Initialisierung: in G_k bilden alle Knoten ein VC der Größe k

Iterative Kompression

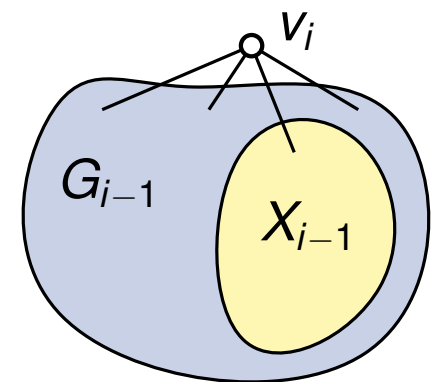
Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Gerade gesehen: $O(2^k m)$ -Algorithmus für VERTEX COVER COMPRESSION

Algorithmus für VERTEX COVER

- iteriere über die Subgraphen G_i bestehend aus den Knoten v_1, \dots, v_i
- Initialisierung: in G_k bilden alle Knoten ein VC der Größe k
- Annahme für $i > k$: für G_{i-1} kennen wir ein VC X_{i-1} der Größe k



Iterative Kompression

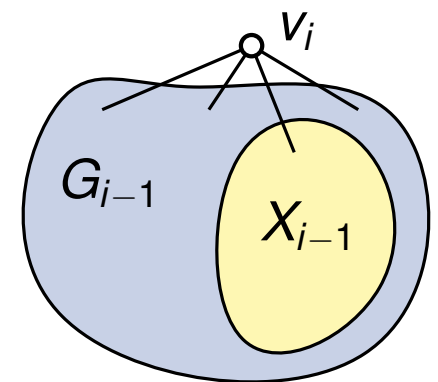
Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Gerade gesehen: $O(2^k m)$ -Algorithmus für VERTEX COVER COMPRESSION

Algorithmus für VERTEX COVER

- iteriere über die Subgraphen G_i bestehend aus den Knoten v_1, \dots, v_i
- Initialisierung: in G_k bilden alle Knoten ein VC der Größe k
- Annahme für $i > k$: für G_{i-1} kennen wir ein VC X_{i-1} der Größe k
- $\Rightarrow X_{i-1} \cup \{v_i\}$ ist ein VC der Größe $k + 1$ in G_i



Iterative Kompression

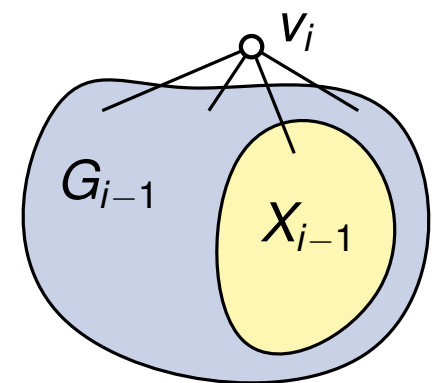
Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Gerade gesehen: $O(2^k m)$ -Algorithmus für VERTEX COVER COMPRESSION

Algorithmus für VERTEX COVER

- iteriere über die Subgraphen G_i bestehend aus den Knoten v_1, \dots, v_i
- Initialisierung: in G_k bilden alle Knoten ein VC der Größe k
- Annahme für $i > k$: für G_{i-1} kennen wir ein VC X_{i-1} der Größe k
- $\Rightarrow X_{i-1} \cup \{v_i\}$ ist ein VC der Größe $k + 1$ in G_i
- verwende Algo für VERTEX COVER COMPRESSION:
 - **Fall 1:** G_i hat ein VC der Größe $k \Rightarrow$ weiter mit G_{i+1}
 - **Fall 2:** G_i hat kein VC der Größe $k \Rightarrow G$ hat kein VC der Größe $k \Rightarrow$ Abbruch



Iterative Kompression

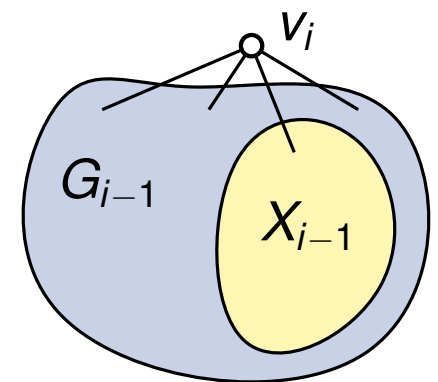
Problem: VERTEX COVER COMPRESSION

Gegeben sind ein Graph G , ein Parameter k und ein VC Z der Größe $k + 1$ in G . Gibt es ein VC der Größe k in G ?

Gerade gesehen: $O(2^k m)$ -Algorithmus für VERTEX COVER COMPRESSION

Algorithmus für VERTEX COVER

- iteriere über die Subgraphen G_i bestehend aus den Knoten v_1, \dots, v_i
- Initialisierung: in G_k bilden alle Knoten ein VC der Größe k
- Annahme für $i > k$: für G_{i-1} kennen wir ein VC X_{i-1} der Größe k
- $\Rightarrow X_{i-1} \cup \{v_i\}$ ist ein VC der Größe $k + 1$ in G_i
- verwende Algo für VERTEX COVER COMPRESSION:
 - **Fall 1:** G_i hat ein VC der Größe $k \Rightarrow$ weiter mit G_{i+1}
 - **Fall 2:** G_i hat kein VC der Größe $k \Rightarrow G$ hat kein VC der Größe $k \Rightarrow$ Abbruch



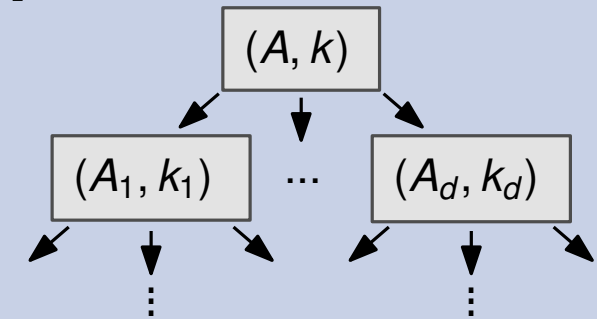
\Rightarrow Algo für VERTEX COVER mit Laufzeit $O(2^k nm)$

Grundlegende Techniken

Beschränkter Suchbaum

(Bounded Search Tree)

- für eine Instanz (A, k) bilde, in Zeit n^c , $(A_1, k_1), \dots, (A_d, k_d)$, sodass:
 (A, k) ist lösbar $\Leftrightarrow (A_i, k_i)$ ist lösbar für ein $i \in [1, d]$
- beschränke d durch eine Funktion $f_1(k)$
- beschränke Verzweigungstiefe durch $f_2(k)$
 (Beispiel: Parameter wird in jedem Schritt kleiner)
- \Rightarrow FPT-Algo mit Laufzeit $O(f_1(k)^{f_2(k)} n^c)$



Kernbildung

(Kernelization)

- wende sukzessive sichere Reduktionsregeln an
- zeige: übrig bleibt ein Kern, dessen Größe nur von k abhängt

Iterative Kompression

(Iterative Compression)

- Kompressionsalgo: löst das Problem unter der Annahme eine etwas zu große Lösung zu kennen
- vergrößere Instanz schrittweise und halte initiale Lösung durch wiederholte Kompression klein

VERTEX COVER \in FPT

VERTEX COVER \in FPT
(wenn man die Lösungsgröße als Parameter verwendet!)

Bonus: Wie verhält sich FPT zu NP?

Bonus: Wie verhält sich FPT zu NP?

Formal korrekte Antwort: gar nicht

- $NP \cap FPT = \emptyset$ (NP enthält Probleme, FPT enthält parametrisierte Probleme)

Bonus: Wie verhält sich FPT zu NP?

Formal korrekte Antwort: gar nicht

- $NP \cap FPT = \emptyset$ (NP enthält Probleme, FPT enthält parametrisierte Probleme)
- sei L entscheidbar; es gibt Parametrisierungen L' und L'' sodass:
 - $L' \in FPT$ (Laufzeit des Algos als Parameter)
 - $L'' \notin FPT$, außer wenn $L \in P$ (konstanter Parameter)

Bonus: Wie verhält sich FPT zu NP?

Formal korrekte Antwort: gar nicht

- $NP \cap FPT = \emptyset$ (NP enthält Probleme, FPT enthält parametrisierte Probleme)
- sei L entscheidbar; es gibt Parametrisierungen L' und L'' sodass:
 - $L' \in FPT$ (Laufzeit des Algos als Parameter)
 - $L'' \notin FPT$, außer wenn $L \in P$ (konstanter Parameter)

Typische Situation: Was wollen wir mit FPT erreichen?

- löse Probleme in NPC halbwegs effizient

Bonus: Wie verhält sich FPT zu NP?

Formal korrekte Antwort: gar nicht

- $NP \cap FPT = \emptyset$ (NP enthält Probleme, FPT enthält parametrisierte Probleme)
- sei L entscheidbar; es gibt Parametrisierungen L' und L'' sodass:
 - $L' \in FPT$ (Laufzeit des Algos als Parameter)
 - $L'' \notin FPT$, außer wenn $L \in P$ (konstanter Parameter)

Typische Situation: Was wollen wir mit FPT erreichen?

- löse Probleme in NPC halbwegs effizient



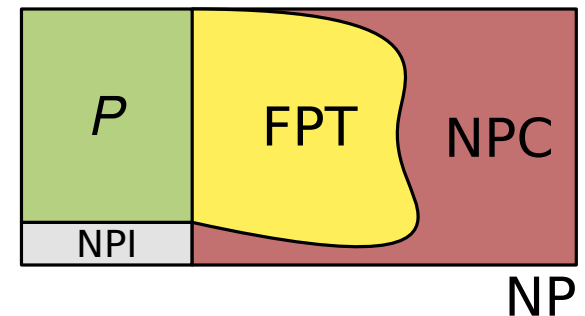
Bonus: Wie verhält sich FPT zu NP?

Formal korrekte Antwort: gar nicht

- $NP \cap FPT = \emptyset$ (NP enthält Probleme, FPT enthält parametrisierte Probleme)
- sei L entscheidbar; es gibt Parametrisierungen L' und L'' sodass:
 - $L' \in FPT$ (Laufzeit des Algos als Parameter)
 - $L'' \notin FPT$, außer wenn $L \in P$ (konstanter Parameter)

Typische Situation: Was wollen wir mit FPT erreichen?

- löse Probleme in NPC halbwegs effizient
- wir betrachten hauptsächlich Probleme in NP



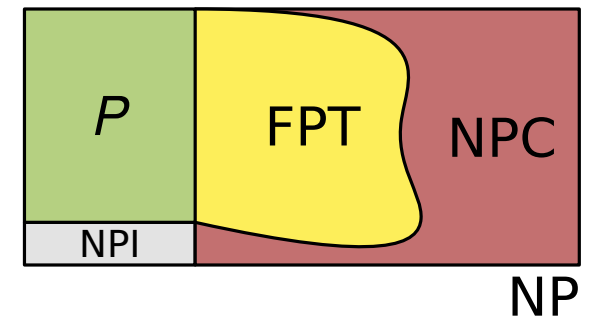
Bonus: Wie verhält sich FPT zu NP?

Formal korrekte Antwort: gar nicht

- $NP \cap FPT = \emptyset$ (NP enthält Probleme, FPT enthält parametrisierte Probleme)
- sei L entscheidbar; es gibt Parametrisierungen L' und L'' sodass:
 - $L' \in FPT$ (Laufzeit des Algos als Parameter)
 - $L'' \notin FPT$, außer wenn $L \in P$ (konstanter Parameter)

Typische Situation: Was wollen wir mit FPT erreichen?

- löse Probleme in NPC halbwegs effizient
- wir betrachten hauptsächlich Probleme in NP
- gute Vorstellung: FPT schlägt eine Brücke zwischen P und NPC
(auch wenn man das formal so nicht sagen kann)



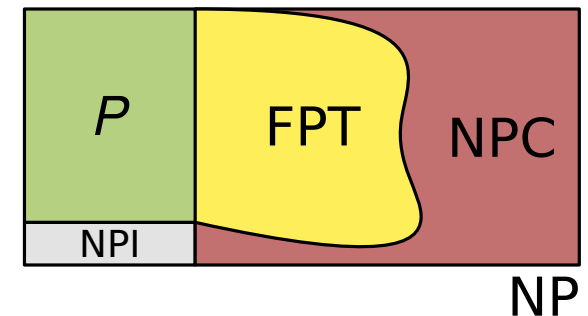
Bonus: Wie verhält sich FPT zu NP?

Formal korrekte Antwort: gar nicht

- $NP \cap FPT = \emptyset$ (NP enthält Probleme, FPT enthält parametrisierte Probleme)
- sei L entscheidbar; es gibt Parametrisierungen L' und L'' sodass:
 - $L' \in FPT$ (Laufzeit des Algos als Parameter)
 - $L'' \notin FPT$, außer wenn $L \in P$ (konstanter Parameter)

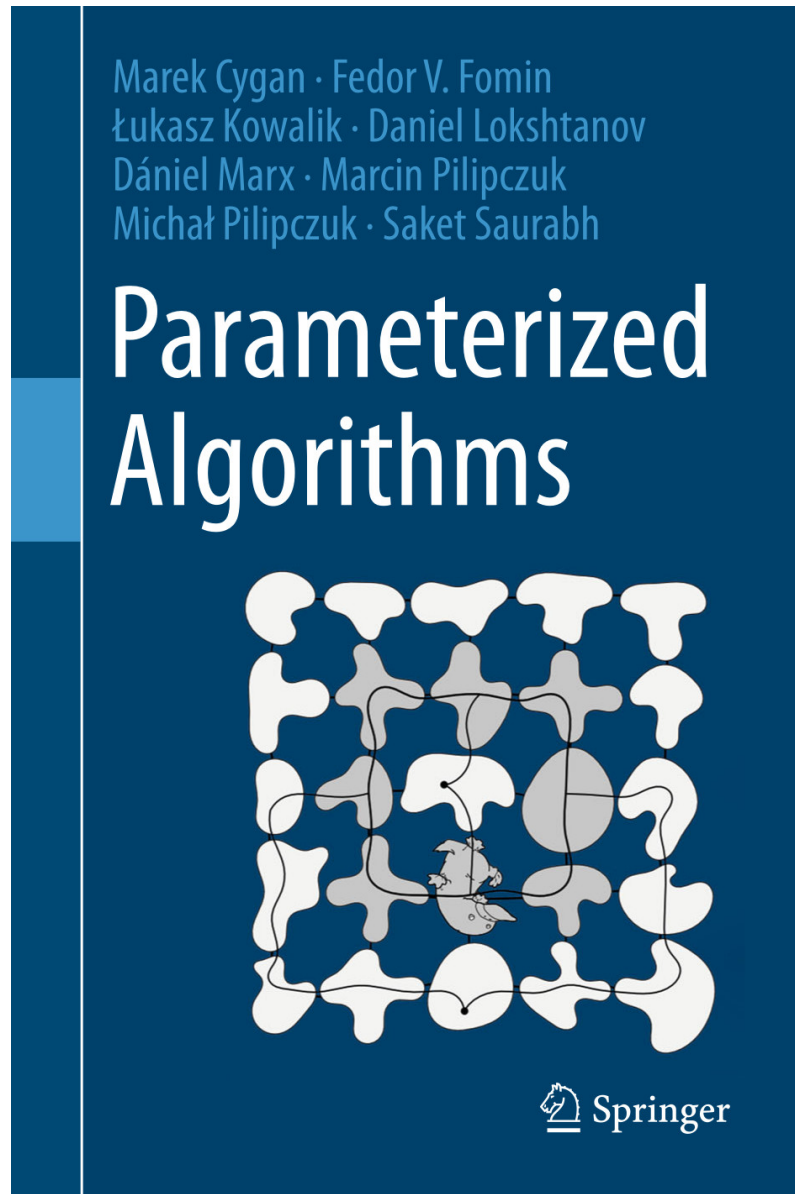
Typische Situation: Was wollen wir mit FPT erreichen?

- löse Probleme in NPC halbwegs effizient
- wir betrachten hauptsächlich Probleme in NP
- gute Vorstellung: FPT schlägt eine Brücke zwischen P und NPC
(auch wenn man das formal so nicht sagen kann)



Zusatzhinweis: para-NP

- para-NP ist das Gegenstück zu NP für parametrisierte Probleme
- FPT verhält sich zu para-NP wie P zu NP



Anmerkungen

- viele der behandelten Themen findet ihr so oder so ähnlich in diesem Buch
- aus dem Uninetz kostenlos abrufbar

link.springer.com/book/10.1007/978-3-319-21275-3