Karlsruher Institut für Technologie

Algorithmen II
Vorlesung am 15.01.2013

Parametrisierte Algorithmen

INSTITUT FÜR THEORETISCHE INFORMATIK · PROF. DR. DOROTHEA WAGNER

Karlsruher Institut für Technologie

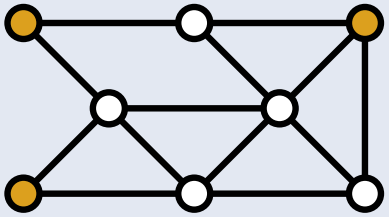
Parametrisierte Algorithmen – Einführung

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

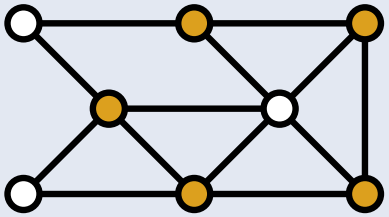
Drei Probleme

Gegeben: Ein Graph $G = (V, E)$, sowie ein Parameter $k \in \mathbb{N}$.

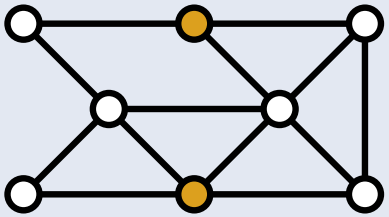
Problem: INDEPENDENT SET
Finde *unabhängige Menge* $V' \subseteq V$ mit $|V'| \geq k$. V' heißt unabhängig genau dann, wenn für alle $v, w \in V'$ gilt $\{v, w\} \notin E$.



Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.



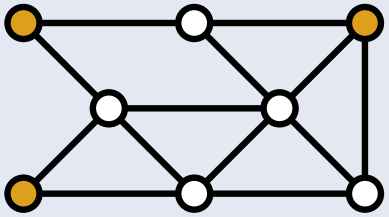
Problem: DOMINATING SET
Finde *Dominating Set* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Dominating Set genau dann, wenn für jeden Knoten $v \in V$ gilt, dass v oder einer seiner Nachbarn in V' enthalten ist.



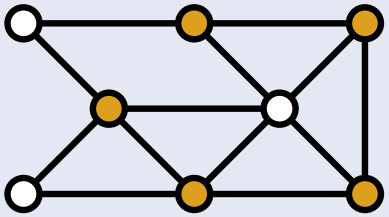
Drei Probleme

Gegeben: Ein Graph $G = (V, E)$, sowie ein Parameter $k \in \mathbb{N}$.

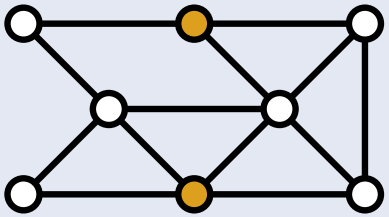
Problem: INDEPENDENT SET
Finde *unabhängige Menge* $V' \subseteq V$ mit $|V'| \geq k$. V' heißt unabhängig genau dann, wenn für alle $v, w \in V'$ gilt $\{v, w\} \notin E$.



Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.



Problem: DOMINATING SET
Finde *Dominating Set* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Dominating Set genau dann, wenn für jeden Knoten $v \in V$ gilt, dass v oder einer seiner Nachbarn in V' enthalten ist.



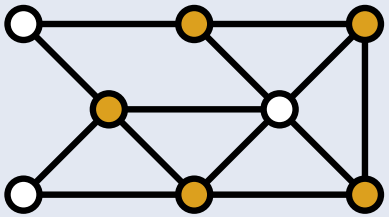
INDEPENDENT SET, VERTEX COVER und DOMINATING SET sind $\mathcal{NP}$-schwer.

$\rightarrow$ Aufzählung aller $\binom{n}{k}$ Teilmengen (Brute-Force) V' mit $|V'| = k$ liefert Algorithmus mit Laufzeit $O(n^k \cdot (n + m))$. Für konstantes k polynomiell! Geht es noch besser?

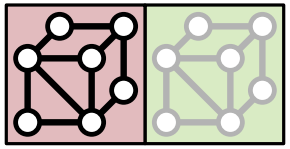
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus für VERTEX COVER

Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.



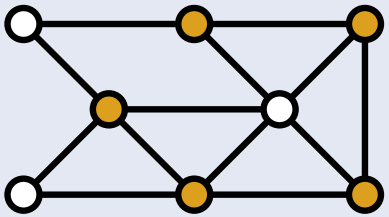
noch zu überdeckender Teilgraph ausgewählte Knoten & überdeckte Kanten



Gibt es ein VERTEX COVER mit maximal 3 Knoten?

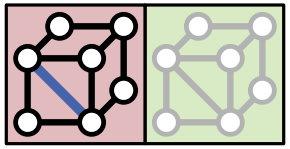
Algorithmus für VERTEX COVER

Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.



noch zu überdeckender Teilgraph — ausgewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden. $\rightarrow$ wähle eine beliebige.

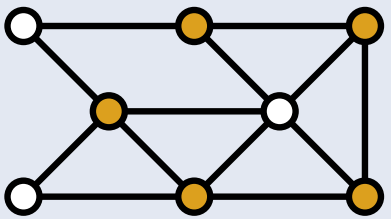


Gibt es ein VERTEX COVER mit maximal 3 Knoten?

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

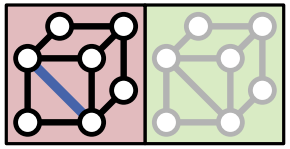
Algorithmus für VERTEX COVER

Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.

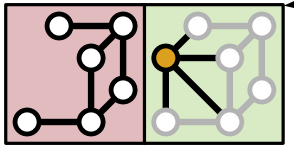


noch zu überdeckender Teilgraph — ausgewählte Knoten & überdeckte Kanten

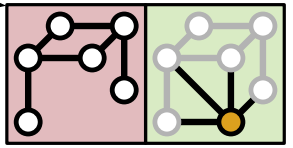
Jede Kante muss noch überdeckt werden. $\rightarrow$ wähle eine beliebige.



Gibt es ein VERTEX COVER mit maximal 3 Knoten?



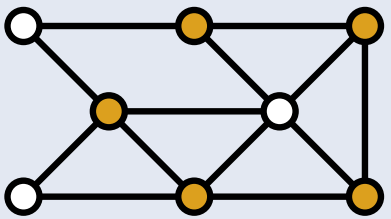
Für die Kante $\{v, w\}$ muss v oder w enthalten sein.
$\rightarrow$ binärer Entscheidungsbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

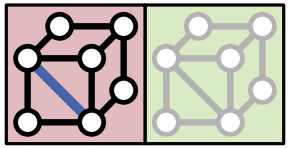
Algorithmus für VERTEX COVER

Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.

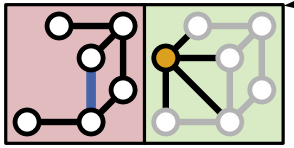


noch zu überdeckender Teilgraph — ausgewählte Knoten & überdeckte Kanten

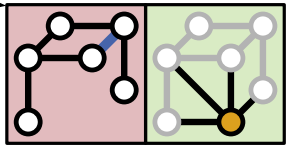
Jede Kante muss noch überdeckt werden. $\rightarrow$ wähle eine beliebige.



Gibt es ein VERTEX COVER mit maximal 3 Knoten?



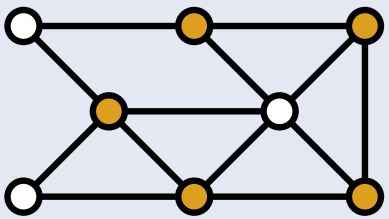
Für die Kante $\{v, w\}$ muss v oder w enthalten sein.
$\rightarrow$ binärer Entscheidungsbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

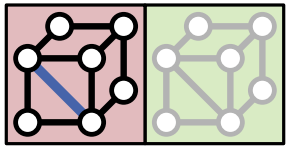
Algorithmus für VERTEX COVER

Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.



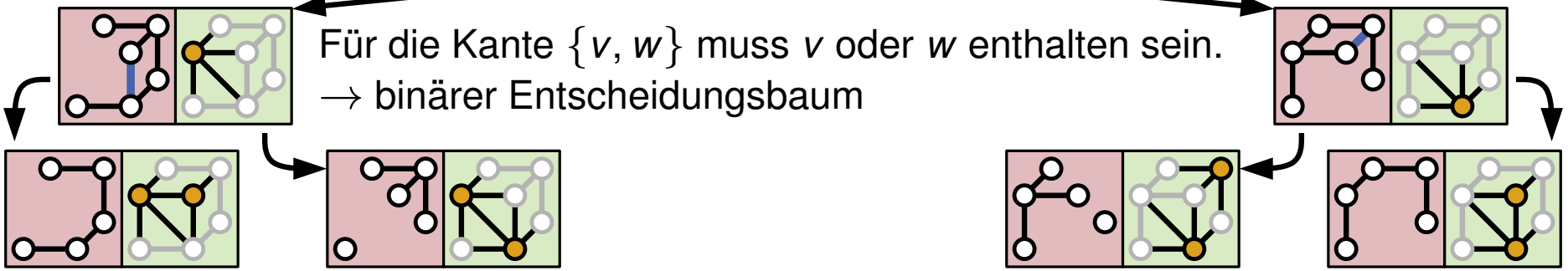
noch zu überdeckender Teilgraph — ausgewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden. → wähle eine beliebige.



Gibt es ein VERTEX COVER mit maximal 3 Knoten?

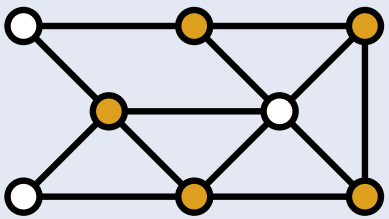
Für die Kante $\{v, w\}$ muss v oder w enthalten sein.
→ binärer Entscheidungsbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

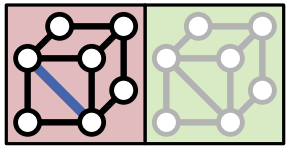
Algorithmus für VERTEX COVER

Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.



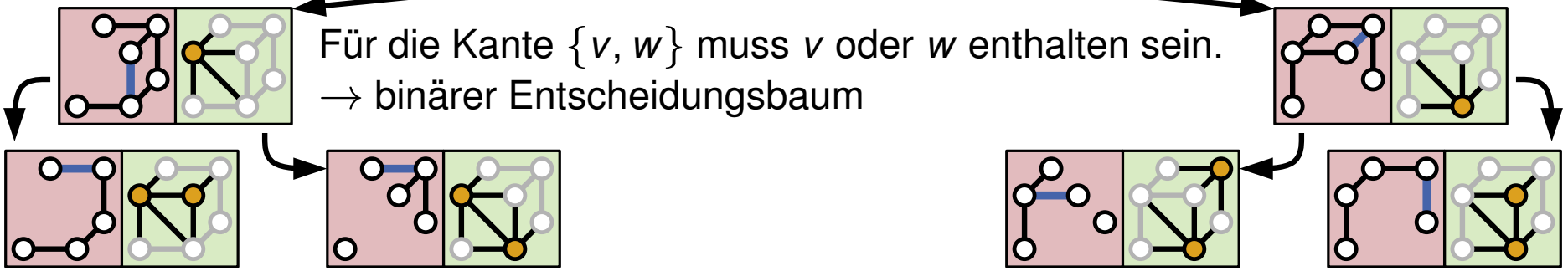
noch zu überdeckender Teilgraph — ausgewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden. → wähle eine beliebige.



Gibt es ein VERTEX COVER mit maximal 3 Knoten?

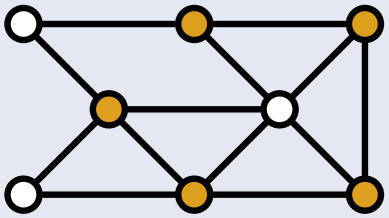
Für die Kante $\{v, w\}$ muss v oder w enthalten sein.
→ binärer Entscheidungsbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

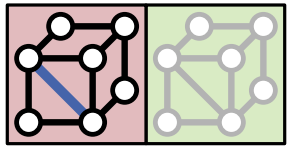
Algorithmus für VERTEX COVER

Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.



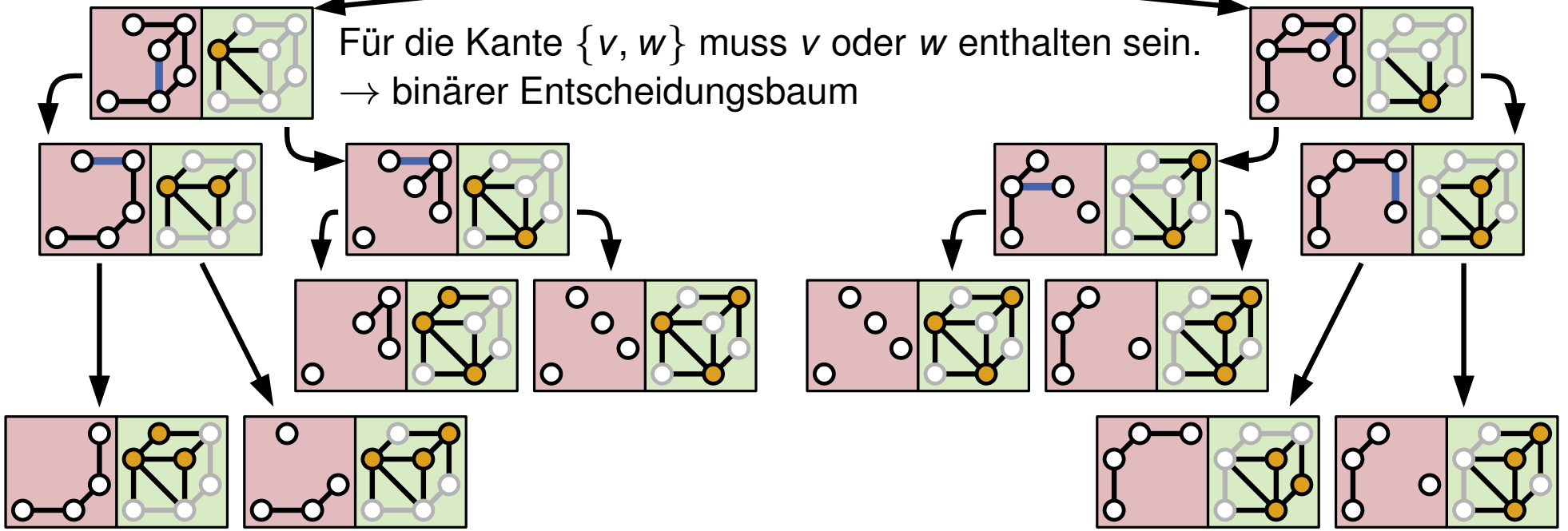
noch zu überdeckender Teilgraph — ausgewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden. → wähle eine beliebige.



Gibt es ein VERTEX COVER mit maximal 3 Knoten?

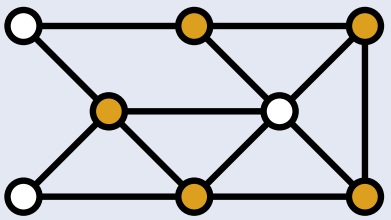
Für die Kante $\{v, w\}$ muss v oder w enthalten sein.
→ binärer Entscheidungsbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

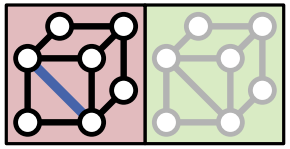
Algorithmus für VERTEX COVER

Problem: VERTEX COVER
Finde *Vertex Cover* $V' \subseteq V$ mit $|V'| \leq k$. V' heißt Vertex Cover genau dann, wenn für jede Kante $\{v, w\} \in E$ gilt $v \in V'$ oder $w \in V'$.



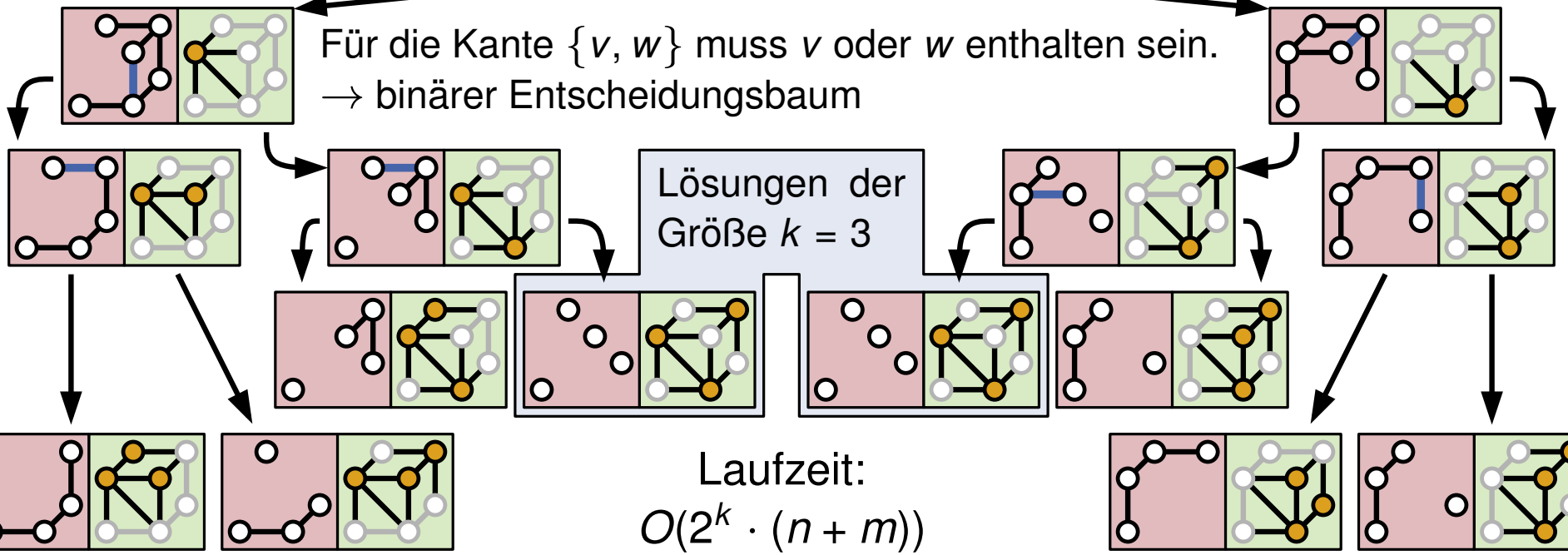
noch zu überdeckender Teilgraph — ausgewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden. → wähle eine beliebige.



Gibt es ein VERTEX COVER mit maximal 3 Knoten?

Für die Kante $\{v, w\}$ muss v oder w enthalten sein.
→ binärer Entscheidungsbaum



Laufzeit:
$O(2^k \cdot (n + m))$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Fixed Parameter Tractable (FPT)

- Der Brute-Force Ansatz hat Laufzeit $O(n^k \cdot (n+m))$.
  - Polynomielle Laufzeit für konstantes k.
- Der Algorithmus für VERTEX COVER benötigt $O(2^k \cdot (n+m))$ Zeit.
  - Polynomielle Laufzeit für konstantes k und
  - Asymptotische Laufzeit hängt nicht von k ab, falls k konstant.

Fixed Parameter Tractable (FPT)

- Der Brute-Force Ansatz hat Laufzeit $O(n^k \cdot (n+m))$.
  - Polynomielle Laufzeit für konstantes k.
- Der Algorithmus für VERTEX COVER benötigt $O(2^k \cdot (n+m))$ Zeit.
  - Polynomielle Laufzeit für konstantes k und
  - Asymptotische Laufzeit hängt nicht von k ab, falls k konstant.

Definition: Fixed Parameter Tractable **(Definition 10.1)**

Ein parametrisiertes Problem Π heißt *fixed parameter tractable*, wenn es in $O(\mathcal{C}(k) \cdot p(n))$ gelöst werden kann. Dabei ist n die Eingabegröße, p ein Polynom, k der Parameter und $\mathcal{C}$ eine berechenbare Funktion, die nur von k abhängt.

Definition: Komplexitätsklasse FPT **(Definition 10.2)**

FPT ist die Klasse aller Problem, die fixed parameter tractable sind.

Bemerkung: **(Bemerkung 10.3)**

- VERTEX COVER ist in FPT.
- Es ist unbekannt ob INDEPENDENT SET oder DOMINATING SET in FPT sind. Man vermutet, dass sie nicht in FPT sind.

Parametrisierte Komplexitätstheorie (Exkurs)

Definition: Parametrisierte Reduktion **(Definition 10.4)**

Seien Π und Π' parametrisierte Probleme. Eine *parametrisierte Reduktion* von Π auf Π' besteht aus einer Funktion f, die einer Instanz $\mathcal{I}$ mit Parameter k von Π eine Instanz $\mathcal{I}'$ von Π' zuordnet, sowie Funktionen $\mathcal{C}', \mathcal{C}'' : \mathbb{N} \to \mathbb{N}$, sodass gilt:

- $\mathcal{I}' = f(\mathcal{I}, k)$ kann in $O(\mathcal{C}''(k) \cdot p(n))$ berechnet werden (n ist Eingabegröße von $\mathcal{I}$ und p ein Polynom).
- $(\mathcal{I}, k)$ ist Ja-Instanz von Π genau dann wenn $(\mathcal{I}', \mathcal{C}'(k))$ ist Ja-Instanz von Π'.

Parametrisierte Komplexitätstheorie (Exkurs)

Definition: Parametrisierte Reduktion **(Definition 10.4)**

Seien Π und Π' parametrisierte Probleme. Eine *parametrisierte Reduktion* von Π auf Π' besteht aus einer Funktion f, die einer Instanz $\mathcal{I}$ mit Parameter k von Π eine Instanz $\mathcal{I}'$ von Π' zuordnet, sowie Funktionen $\mathcal{C}', \mathcal{C}'' : \mathbb{N} \to \mathbb{N}$, sodass gilt:

- $\mathcal{I}' = f(\mathcal{I}, k)$ kann in $O(\mathcal{C}''(k) \cdot p(n))$ berechnet werden (n ist Eingabegröße von $\mathcal{I}$ und p ein Polynom).
- $(\mathcal{I}, k)$ ist Ja-Instanz von Π genau dann wenn $(\mathcal{I}', \mathcal{C}'(k))$ ist Ja-Instanz von Π'.

Bemerkung: Vergleich zur polynomiellen Reduktion

- Ist Π polynomiell auf Π' reduzierbar, so folgt aus $\Pi' \in \mathcal{P}$ auch $\Pi \in \mathcal{P}$.
  $\to \Pi$ ist unter Vernachlässigung polynomieller Laufzeit nicht schwerer als Π'.
- Ist Π parametrisiert auf Π' reduzierbar, so folgt aus $\Pi' \in$ FPT auch $\Pi \in$ FPT.
  $\to \Pi$ ist unter Vernachläsigung von $O(\mathcal{C}(k) \cdot p(n))$ Laufzeit nicht schwerer als Π'.

Parametrisierte Komplexitätstheorie (Exkurs)

Definition: Parametrisierte Reduktion **(Definition 10.4)**

Seien Π und Π' parametrisierte Probleme. Eine *parametrisierte Reduktion* von Π auf Π' besteht aus einer Funktion f, die einer Instanz $\mathcal{I}$ mit Parameter k von Π eine Instanz $\mathcal{I}'$ von Π' zuordnet, sowie Funktionen $\mathcal{C}', \mathcal{C}'' : \mathbb{N} \to \mathbb{N}$, sodass gilt:

- $\mathcal{I}' = f(\mathcal{I}, k)$ kann in $O(\mathcal{C}''(k) \cdot p(n))$ berechnet werden (n ist Eingabegröße von $\mathcal{I}$ und p ein Polynom).
- $(\mathcal{I}, k)$ ist Ja-Instanz von Π genau dann wenn $(\mathcal{I}', \mathcal{C}'(k))$ ist Ja-Instanz von Π'.

Bemerkung: Hierarchie von Komplexitätsklassen **(Bemerkung 10.5)**

- Es gibt eine Hierarchie von Komplexitätsklassen $W[t]$, die vermutlich echt ist und FPT enthält: $\text{FPT} \subseteq W[1] \subseteq W[2] \ldots$.
- Mithilfe der parametrisierten Reduktion lässt sich ein Vollständigkeitsbegriff definieren, um die schweren Probleme in $W[t]$ zu identifizieren.
- INDEPENDENT SET ist $W[1]$-vollständig, DIMINATING SET ist $W[2]$-vollständig.

Parametrisierte Algorithmen – Grundtechniken

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Zwei Grundtechniken

1. **Kernbildung:** Reduziere die Instanz durch Anwendung verschiedener Regeln auf einen (schweren) Problemkern.
2. **Tiefenbeschränkte Suchbäume:** Erschöpfende Suche in einem geeigneten Suchbaum mit beschränkter Tiefe. (zuvor für VERTEX COVER gesehen)

Kernbildung

1. Kernbildung: Reduziere die Instanz durch Anwendung verschiedener Regeln auf einen (schweren) Problemkern.

Idee:

- Reduziere Instanz $(\mathcal{I}, k)$ in $O(p(|\mathcal{I}|))$ Zeit auf eine äquivalente Instanz $\mathcal{I}'$, sodass $|\mathcal{I}'|$ nur von k (und nicht von $|\mathcal{I}|$) abhängt.
- Löse $\mathcal{I}'$ mit erschöpfender Suche.

Kernbildung

1. Kernbildung: Reduziere die Instanz durch Anwendung verschiedener Regeln auf einen (schweren) Problemkern.

Idee:

- Reduziere Instanz $(\mathcal{I}, k)$ in $O(p(|\mathcal{I}|))$ Zeit auf eine äquivalente Instanz $\mathcal{I}'$, sodass $|\mathcal{I}'|$ nur von k (und nicht von $|\mathcal{I}|$) abhängt.
- Löse $\mathcal{I}'$ mit erschöpfender Suche.

Beispiel: VERTEX COVER **(Beispiel 10.6)**

Um ein Vertex Cover $V' \subseteq V$ für $G = (V, E)$ mit $|V'| \leq k$ zu bestimmen, kann man folgende Beobachtungen nutzen:

- Für $v \in V$ liegt v selbst oder seine gesamte Nachbarschaft $N(v)$ in V'.
- V' enthält jeden Knoten $v \in V$ mit $\deg(v) > k$.
- Falls $\Delta(G) \leq k$ und $|E| > k^2$, so hat jedes Vertex Cover mehr als k Knoten. Dabei ist $\Delta(G)$ der Maximalgrad von G.

Kernbildung

Beispiel: VERTEX COVER (Beispiel 10.6)

Um ein Vertex Cover $V' \subseteq V$ für $G = (V, E)$ mit $|V'| \leq k$ zu bestimmen, kann man folgende Beobachtungen nutzen:

- Für $v \in V$ liegt v selbst oder seine gesamte Nachbarschaft $N(v)$ in V'.
- V' enthält jeden Knoten $v \in V$ mit $\deg(v) > k$.
- Falls $\Delta(G) \leq k$ und $|E| > k^2$, so hat jedes Vertex Cover mehr als k Knoten. Dabei ist $\Delta(G)$ der Maximalgrad von G.

VERTEX COVER(G, k)

$H \leftarrow \{v \in V \mid \deg(v) > k\}$

if $|H| > k$ **then** Gib aus: „G hat kein Vertex Cover der Größe k“

else

$k' \leftarrow k - |H|$

$G' \leftarrow G - H$

if $|E(G')| > k \cdot k'$ **then** Gib aus: „G hat kein Vertex Cover der Größe k“

else

entferne alle isolierten Knoten aus G'

berechne ein Vertex Cover der Größe k' im verbleibenden Graphen

Kernbildung

Beispiel: VERTEX COVER (Beispiel 10.6)

Um ein Vertex Cover $V' \subseteq V$ für $G = (V, E)$ mit $|V'| \leq k$ zu bestimmen, kann man folgende Beobachtungen nutzen:

- Für $v \in V$ liegt v selbst oder seine gesamte Nachbarschaft $N(v)$ in V'.
- V' enthält jeden Knoten $v \in V$ mit $\deg(v) > k$.
- Falls $\Delta(G) \leq k$ und $|E| > k^2$, so hat jedes Vertex Cover mehr als k Knoten. Dabei ist $\Delta(G)$ der Maximalgrad von G.

VERTEX COVER(G, k)	$O(2^k \cdot k^2 + n \cdot k)$
$H \leftarrow \{v \in V \mid \deg(v) > k\}$	$O(n \cdot k)$
if $\|H\| > k$ **then** Gib aus: „G hat kein Vertex Cover der Größe k“	$O(1)$
else	
$k' \leftarrow k - \|H\|$	$O(1)$
$G' \leftarrow G - H$	$O(n \cdot k)$
if $\|E(G')\| > k \cdot k'$ **then** Gib aus: „G hat kein Vertex Cover der Größe k“	$O(k \cdot k')$
else	
entferne alle isolierten Knoten aus G'	$O(n)$
berechne ein Vertex Cover der Größe k' im verbleibenden Graphen	$O(2^k \cdot k^2)$

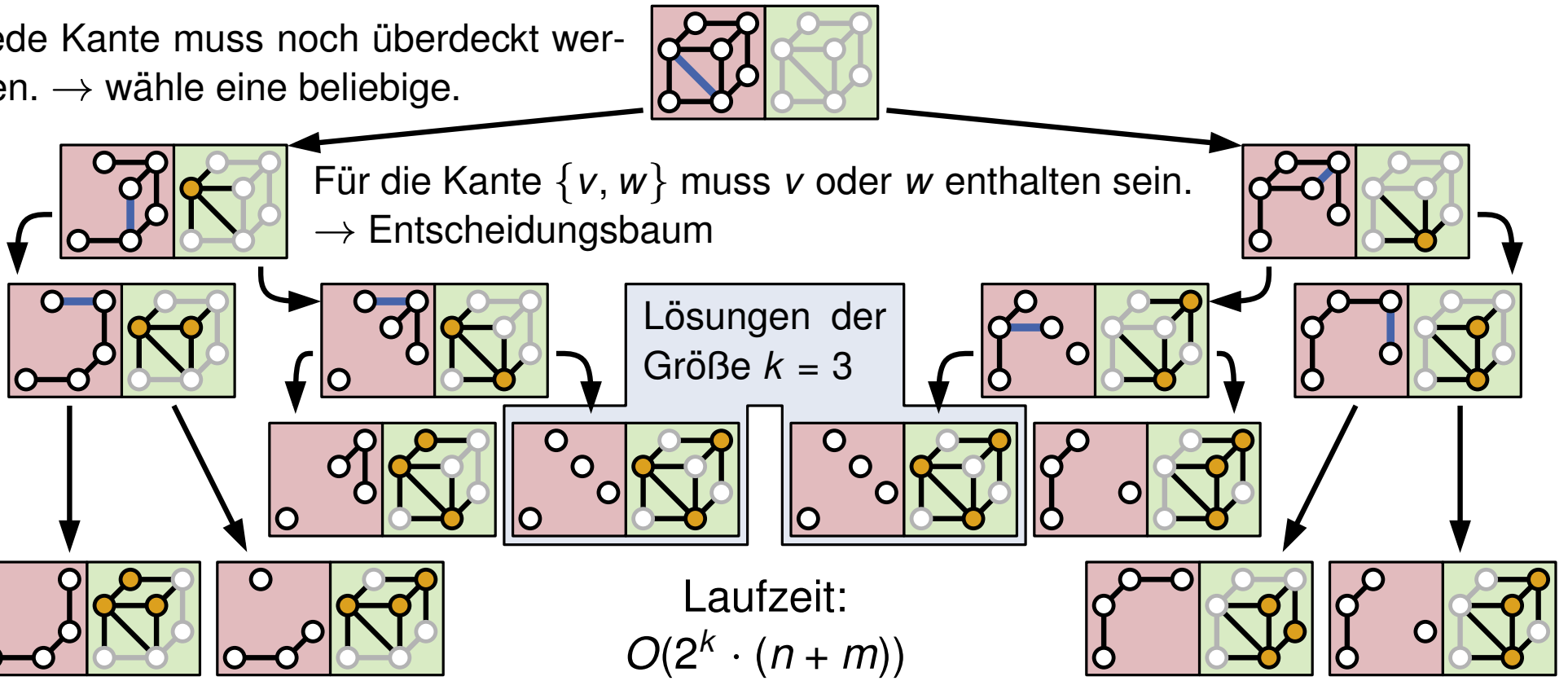
Tiefenbeschränkte Suchbäume

2. Tiefenbeschränkte Suchbäume: Erschöpfende Suche in einem geeigneten Suchbaum mit beschränkter Tiefe. (zuvor für VERTEX COVER gesehen)

Erinnerung:

noch zu überdeckender Teilgraph — ausgewählte Knoten & überdeckte Kanten

Jede Kante muss noch überdeckt werden. → wähle eine beliebige.



Kann der Faktor 2^k verbessert werden?

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Tiefenbeschränkte Suchbäume

2. Tiefenbeschränkte Suchbäume: Erschöpfende Suche in einem geeigneten Suchbaum mit beschränkter Tiefe. (zuvor für VERTEX COVER gesehen)

Laufzeit hängt wesentlich von der Struktur des Baumes ab.

Binärbaum: $T(k) = T(k-1) + T(k-1) + c$ (wie im vorherigen Fall)

Allgemein: $T(k) \leq T(k-t_1) + \cdots + T(k-t_s) + c$

$\rightarrow$ Der Vektor $(t_1, \ldots, t_s)$ heißt *Verzweigungsvektor*. (Für einen Binärbaum also $(1,1)$)

Die Basis b im Term b^k hängt vom Verzweigungsvektor ab.

Verzweigungsvektor	$(1,1)$	$(1,2)$	$(1,3)$	$(1,4)$	$(1,5)$	$(2,3)$	$(3,3)$	$(3,3,6)$	$(3,4,6)$
Basis b	2	1.618	1.4656	1.3803	1.325	1.325	1.26	1.342	1.305

Tiefenbeschränkte Suchbäume

2. Tiefenbeschränkte Suchbäume: Erschöpfende Suche in einem geeigneten Suchbaum mit beschränkter Tiefe. (zuvor für VERTEX COVER gesehen)

Laufzeit hängt wesentlich von der Struktur des Baumes ab.

Binärbaum: $T(k) = T(k-1) + T(k-1) + c$ (wie im vorherigen Fall)

Allgemein: $T(k) \leq T(k-t_1) + \cdots + T(k-t_s) + c$

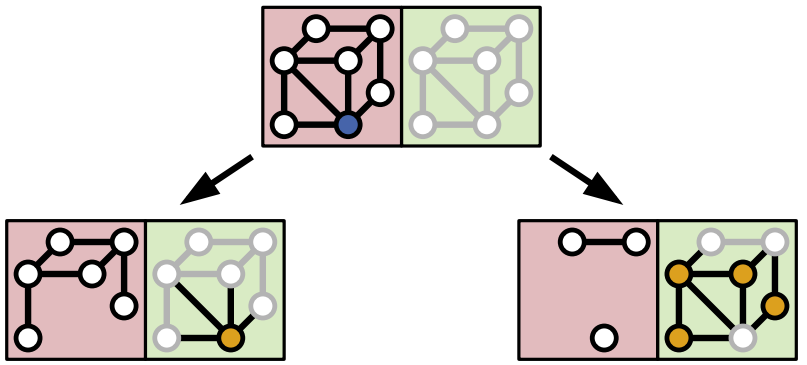
$\rightarrow$ Der Vektor $(t_1, \ldots, t_s)$ heißt *Verzweigungsvektor*. (Für einen Binärbaum also (1, 1))

Die Basis b im Term b^k hängt vom Verzweigungsvektor ab.

Verzweigungsvektor	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(2, 3)	(3, 3)	(3, 3, 6)	(3, 4, 6)
Basis b	2	1.618	1.4656	1.3803	1.325	1.325	1.26	1.342	1.305

Idee für VERTEX COVER:

- Annahme: Es gibt immer einen Knoten v mit $\deg(v) \geq 4$.
- Wähle in jedem Schritt entweder v oder alle seine Nachbarn.
- Verzweigungsvektor: (1, 4).
  Ergibt Laufzeit $O(kn + 1.3803^k \cdot k^2)$



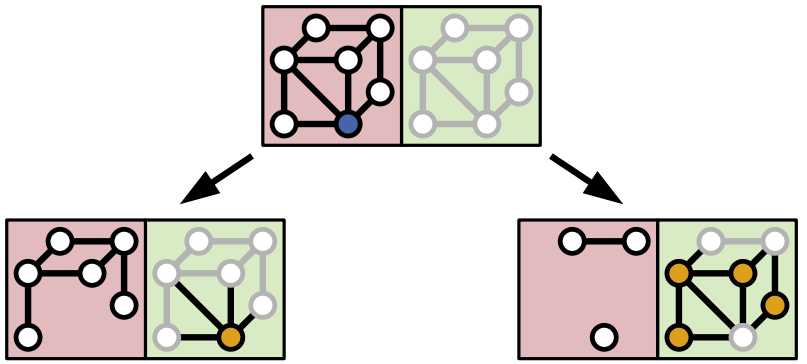
Tiefenbeschränkte Suchbäume

Verzweigungsvektor	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(2, 3)	(3, 3)	(3, 3, 6)	(3, 4, 6)
Basis b	2	1.618	1.4656	1.3803	1.325	1.325	1.26	1.342	1.305

Idee für VERTEX COVER:

- Annahme: Es gibt immer einen Knoten v mit $\deg(v) \geq 4$ gibt.
- Wähle in jedem Schritt entweder v oder alle seine Nachbarn.
- Verzweigungsvektor: $(1, 4)$.

  Ergibt Laufzeit $O(kn + 1.3803^k \cdot k^2)$



Problem: Man findet nicht immer einen Knoten mit Mindestgrad 4.
Idee: Verzweige an jedem Knoten je nach eintretendem Fall mit Verzweigungsvektor $(1, 5)$, $(2, 3)$, $(3, 3)$, $(3, 4, 6)$ oder $(3, 3, 6)$ (oder höchstens einmal $(1, 4)$).
$\Rightarrow$ Laufzeit wird dominiert durch $(3, 3, 6) \Rightarrow$ Gesamtlaufzeit $O(kn + 1.342^k \cdot k^2)$.

Verzweigungsregeln

Im Folgenden:

- Menge von Regeln zur Verzweigung (sodass immer eine der Regeln anwendbar ist).
- Erlaubte Verzweigungsvektoren: $(1,5)$, $(2,3)$, $(3,3)$, $(3,4,6)$, $(3,3,6)$, höchstens einmal $(1,4)$
- Wende bei jeder Verzweigung die Regel mit der kleinsten Nummer an.

Regel 1: Es existiert ein Knoten v mit $\deg(v) = 1$.

Regel 2: Es existiert ein Knoten v mit $\deg(v) \geq 5$.

Regel 3: Es existiert ein Knoten v mit $\deg(v) = 2$.

Regel 4: Es existiert ein Knoten v mit $\deg(v) = 3$.

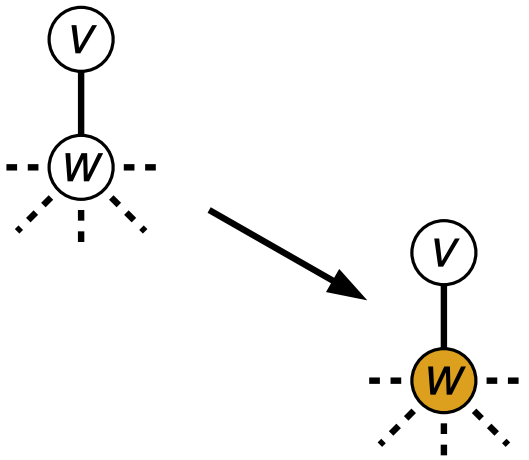
Regel 5: Alle Knoten haben Grad 4.

$\rightarrow$ Offensichtlich ist immer eine der Regeln anwendbar.

Verzweigungsregeln

Regel 1: Es existiert ein Knoten v mit $\deg(v) = 1$.

Sei w der Nachbar von v.

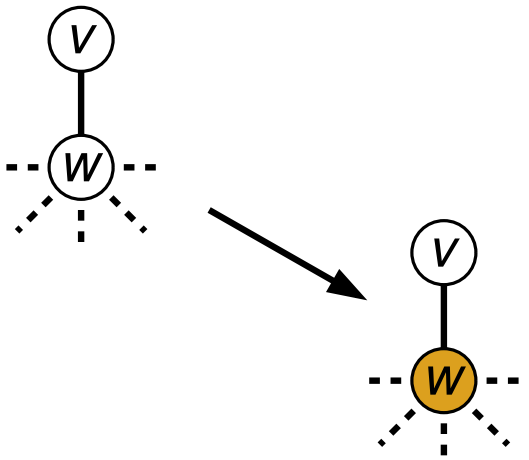


Es kann nie besser sein, v statt w zu nehmen.
$\Rightarrow$ keine Verzweigung nötig.

Verzweigungsregeln

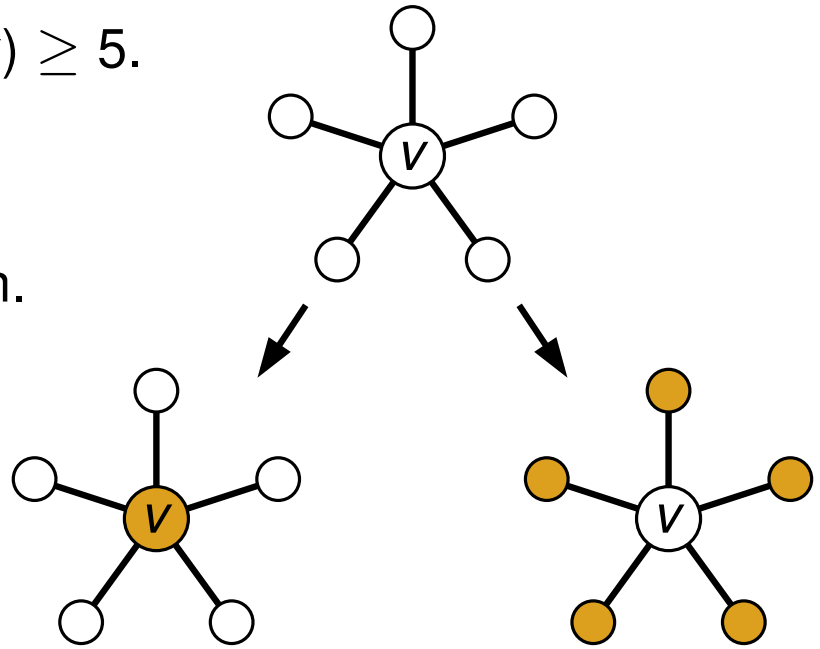
Regel 1: Es existiert ein Knoten v mit $\deg(v) = 1$.

Sei w der Nachbar von v.



Es kann nie besser sein, v statt w zu nehmen.
$\Rightarrow$ keine Verzweigung nötig.

Regel 2: Es existiert ein Knoten v mit $\deg(v) \geq 5$.



Wähle entweder v oder alle seine Nachbarn.
$\Rightarrow$ Verzweiungsvektor $(1, 5)$

Verzweigungsregeln

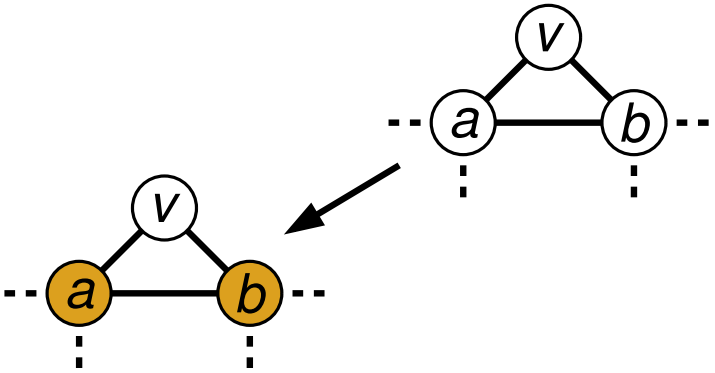
Regel 3: Es existiert ein Knoten v mit $\deg(v) = 2$.

Fall 3.1: Die Nachbarn a und b von v sind adjazent.

$\Rightarrow$ Zwei der Knoten v, a, b müssen gewählt werden.

$\Rightarrow$ Es ist nie besser v zu wählen.

$\Rightarrow$ Keine Verzweigung nötig.



Verzweigungsregeln

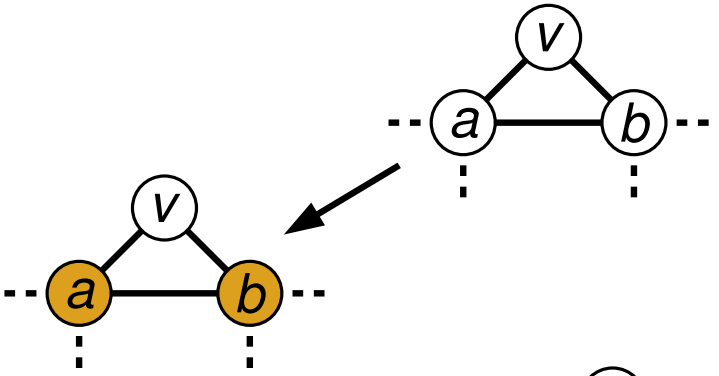
Regel 3: Es existiert ein Knoten v mit $\deg(v) = 2$.

Fall 3.1: Die Nachbarn a und b von v sind adjazent.

$\Rightarrow$ Zwei der Knoten v, a, b müssen gewählt werden.

$\Rightarrow$ Es ist nie besser v zu wählen.

$\Rightarrow$ Keine Verzweigung nötig.

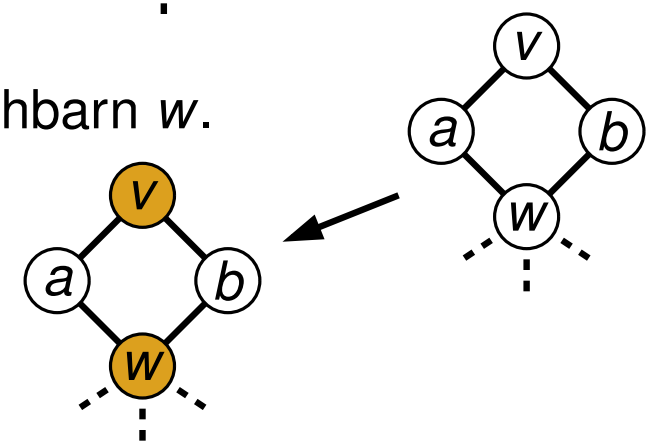


Fall 3.2: a und b haben Grad zwei und gemeinsamen Nachbarn w.

$\Rightarrow$ Zwei der Knoten v, a, b, w müssen gewählt werden.

$\Rightarrow$ Es ist nie besser a und b zu wählen.

$\Rightarrow$ Keine Verzweigung nötig.



Verzweigungsregeln

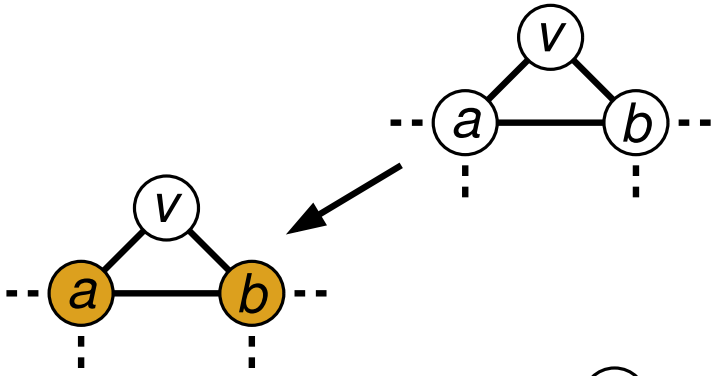
Regel 3: Es existiert ein Knoten v mit $\deg(v) = 2$.

Fall 3.1: Die Nachbarn a und b von v sind adjazent.

$\Rightarrow$ Zwei der Knoten v, a, b müssen gewählt werden.

$\Rightarrow$ Es ist nie besser v zu wählen.

$\Rightarrow$ Keine Verzweigung nötig.

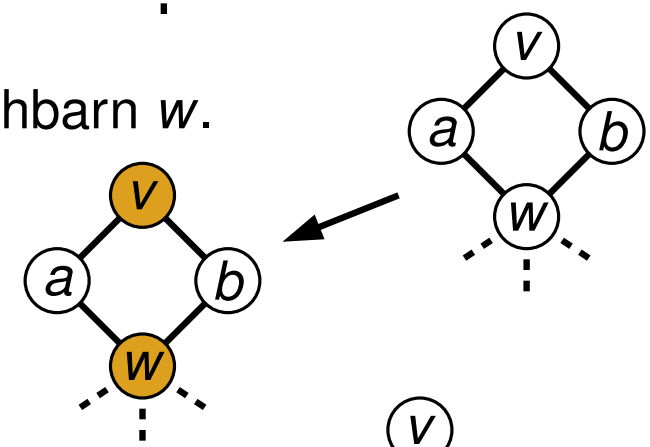


Fall 3.2: a und b haben Grad zwei und gemeinsamen Nachbarn w.

$\Rightarrow$ Zwei der Knoten v, a, b, w müssen gewählt werden.

$\Rightarrow$ Es ist nie besser a und b zu wählen.

$\Rightarrow$ Keine Verzweigung nötig.



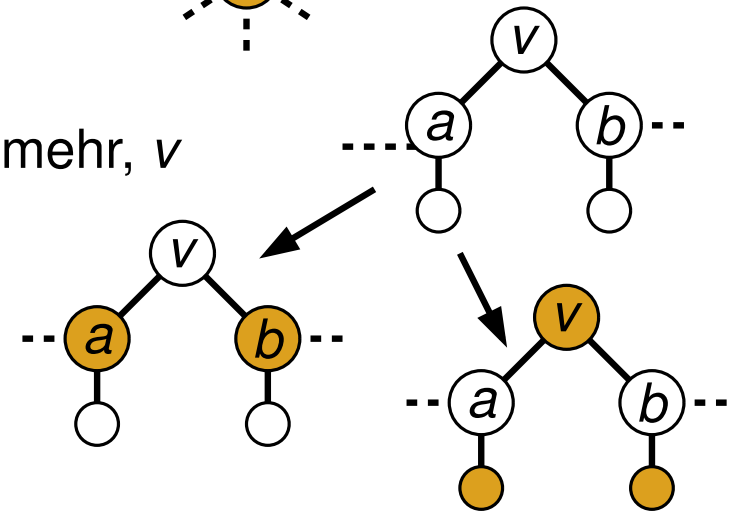
Fall 3.3: sonst

Falls a oder b gewählt wurden, macht es keinen Sinn mehr, v zu wählen.

$\Rightarrow$ Wähle entweder a und b oder $N(a) \cup N(b)$.

Es gilt $\left|N(a) \cup N(b)\right| \geq 3$

$\Rightarrow$ Verzweigungsvektor $(2, 3)$



Verzweigungsregeln

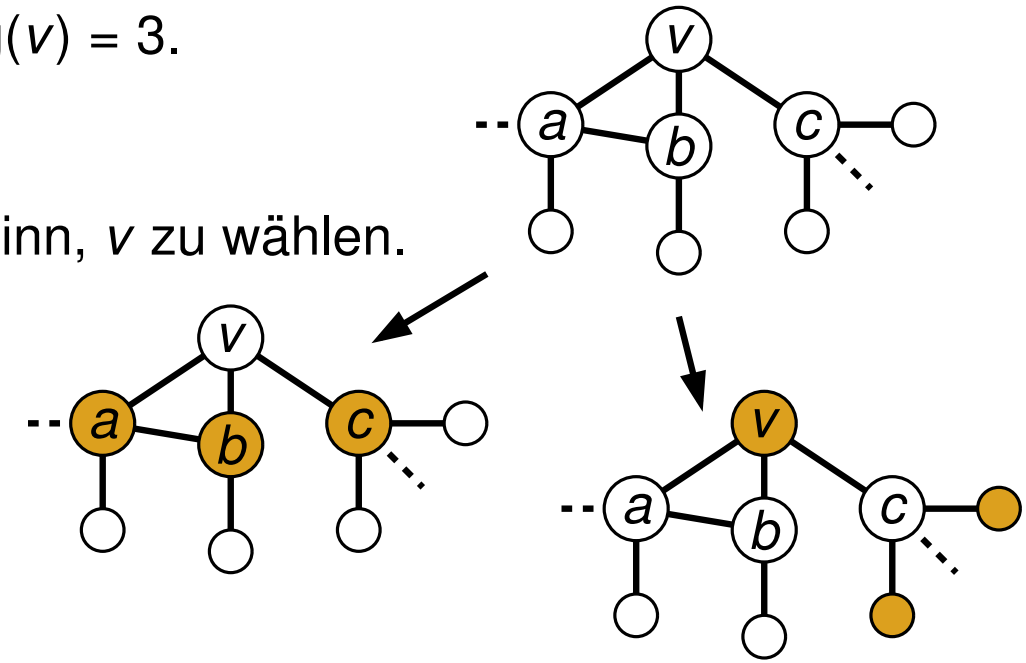
Regel 4: Es existiert ein Knoten v mit $\deg(v) = 3$.

Fall 4.1: v ist Teil eines Dreiecks v, a, b.

Falls c gewählt wurde, macht es keinen Sinn, v zu wählen.

$\Rightarrow$ Füge entweder $N(v)$ oder $N(c)$ hinzu.

$\Rightarrow$ Verzweigungsvektor $(3, 3)$



Verzweigungsregeln

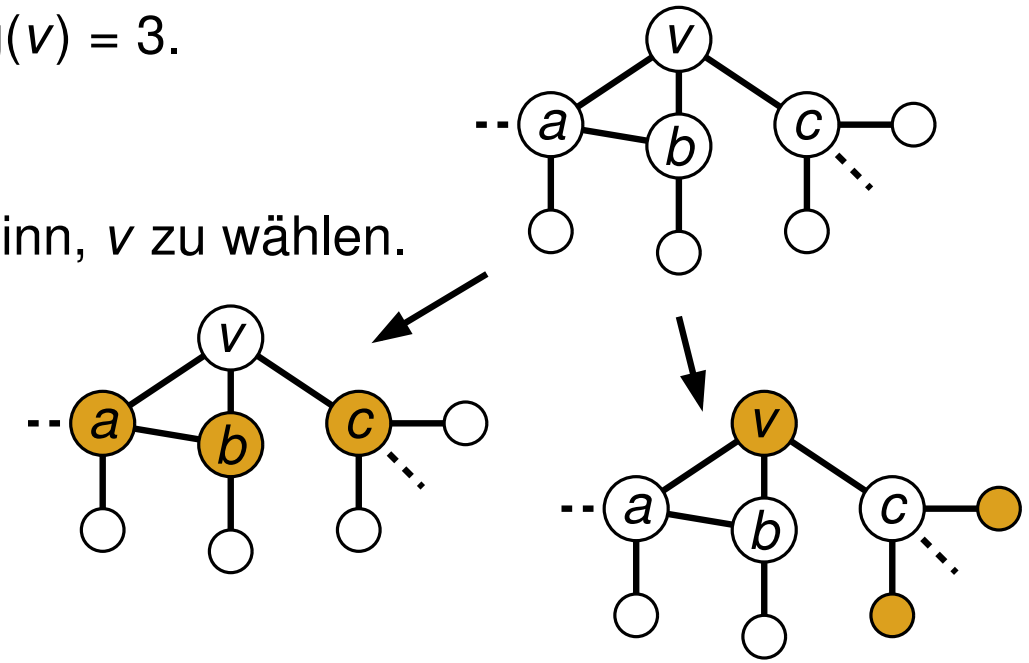
Regel 4: Es existiert ein Knoten v mit $\deg(v) = 3$.

Fall 4.1: v ist Teil eines Dreiecks v, a, b.

Falls c gewählt wurde, macht es keinen Sinn, v zu wählen.

$\Rightarrow$ Füge entweder $N(v)$ oder $N(c)$ hinzu.

$\Rightarrow$ Verzweigungsvektor $(3, 3)$

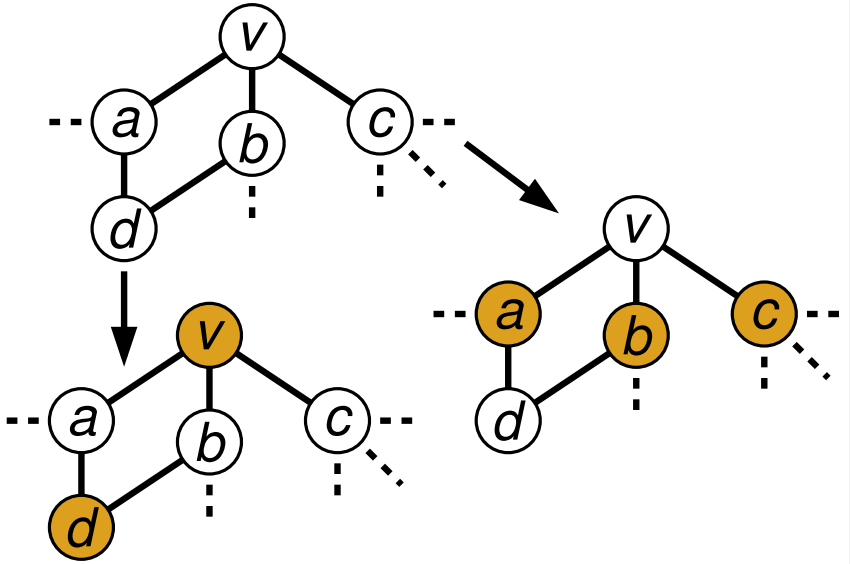


Fall 4.2: v ist Teil eines Vierecks v, a, d, b.

In jedem Vertex Cover sind zumindest v und d oder a und b enthalten. Wenn a und b enthalten sind macht es keinen Sinn v zu wählen.

$\Rightarrow$ Füge entweder a, b und c oder v und d hinzu.

$\Rightarrow$ Verzweigungsvektor $(2, 3)$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Verzweigungsregeln

Regel 4: Es existiert ein Knoten v mit $\deg(v) = 3$.

Fall 4.3: v ist in keinem Drei- oder Viereck enthalten und es gilt $\deg(a) \geq 4$ (für b und c analog).

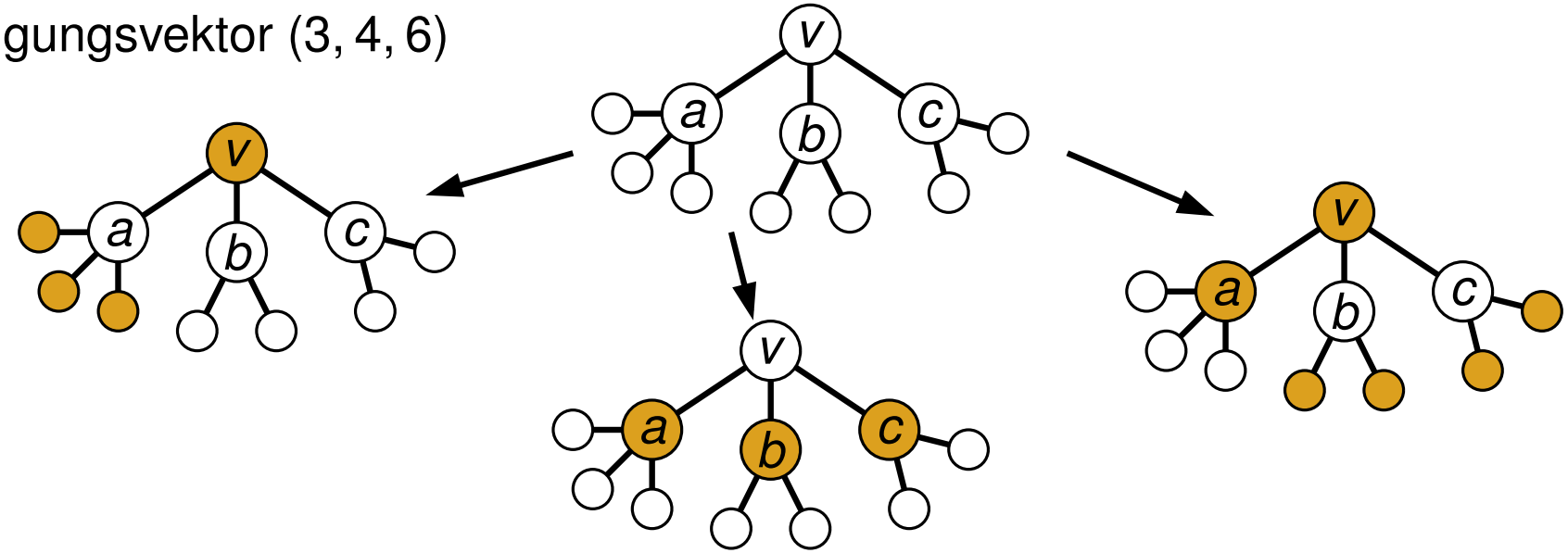
Ein Vertex Cover, das a nicht enthält, enthält $N(a)$.

Ist a sowie einer von b oder c enthalten, so macht es keinen Sinn v zu wählen.

$\Rightarrow$ Wenn a enthalten ist, sind entweder b und c enthalten oder beide nicht.

$\Rightarrow$ Füge entweder $N(a)$ oder a, b und c oder a, $N(b)$ und $N(c)$ hinzu.

$\Rightarrow$ Verzweigungsvektor $(3, 4, 6)$



Verzweigungsregeln

Regel 4: Es existiert ein Knoten v mit $\deg(v) = 3$.

Fall 4.4: v ist in keinem Drei- oder Viereck enthalten und es gilt $\deg(a) = \deg(b) = \deg(c) = 3$

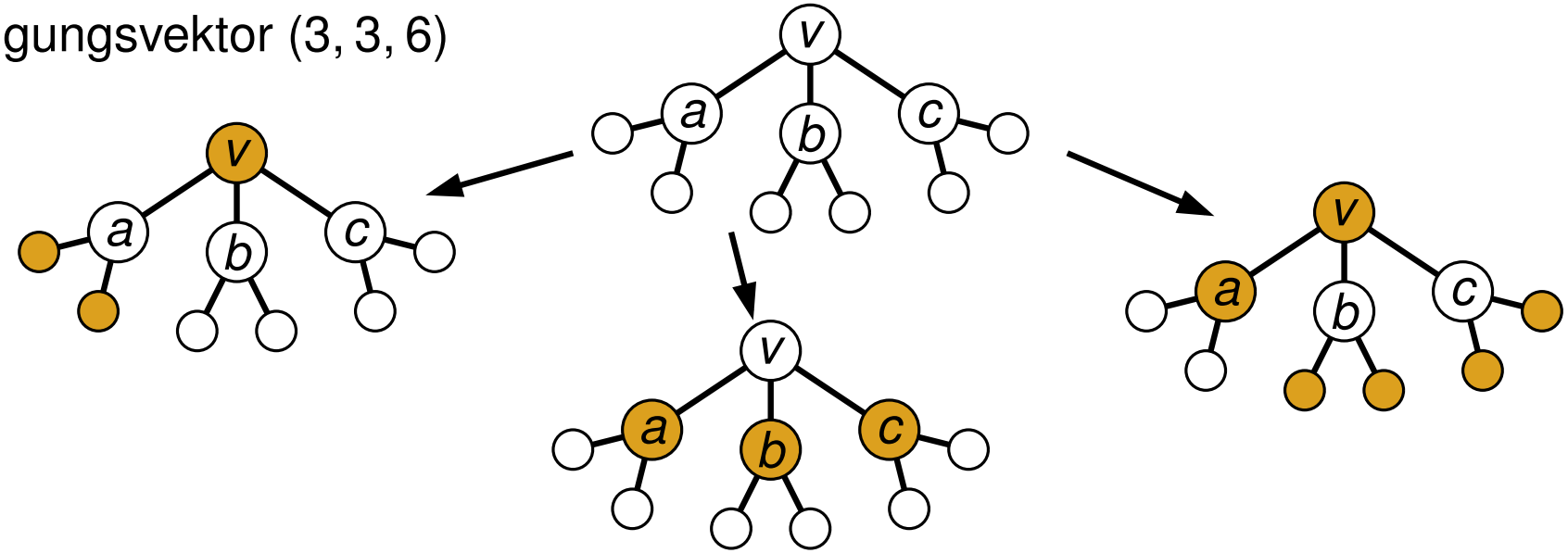
Ein Vertex Cover, das a nicht enthält, enthält $N(a)$.

Ist a sowie einer von b oder c enthalten, so macht es keinen Sinn v zu wählen.

$\Rightarrow$ Wenn a enthalten ist sind entweder b und c beide enthalten oder beide nicht.

$\Rightarrow$ Füge entweder $N(a)$ oder a, b und c oder a, $N(b)$ und $N(c)$ hinzu.

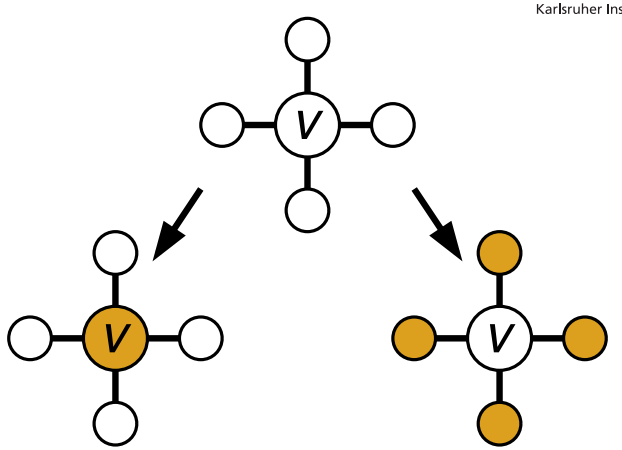
$\Rightarrow$ Verzweigungsvektor $(3, 3, 6)$



Verzweigungsregeln

Regel 5: Alle Knoten haben Grad 4.

Wähle einen Knoten v oder alle seine Nachbarn.
$\Rightarrow$ Verzweiungsvektor $(1, 4)$



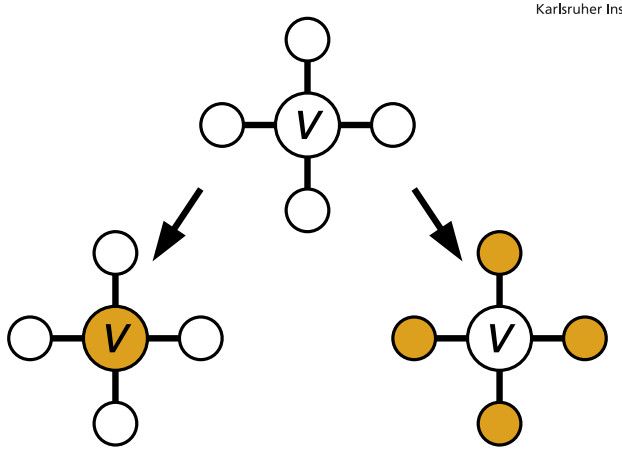
Beachte:

- Die Regel wird nur angewendet, wenn jeder Knoten Grad 4 hat.
- Jeder Subgraph enthält Knoten mit kleinerem Grad.
  $\Rightarrow$ Regel 5 wird pro Pfad von der Wurzel zu einem Blatt nur einmal ausgeführt.

Verzweigungsregeln

Regel 5: Alle Knoten haben Grad 4.

Wähle einen Knoten v oder alle seine Nachbarn.
$\Rightarrow$ Verzweiungsvektor $(1, 4)$



Zusammenfassung:

- Menge von Regeln zur Verzweigung (sodass immer eine der Regeln anwendbar ist).
- Erlaubte Verzweigungsvektoren: $(1, 5)$, $(2, 3)$, $(3, 3)$, $(3, 4, 6)$, $(3, 3, 6)$, höchstens einmal $(1, 4)$
- Wende bei jeder Verzweigung die Regel mit der kleinsten Nummer an.

Regel 1: Es existiert ein Knoten v mit $\deg(v) = 1$.

Regel 2: Es existiert ein Knoten v mit $\deg(v) \geq 5$.

Regel 3: Es existiert ein Knoten v mit $\deg(v) = 2$.

Regel 4: Es existiert ein Knoten v mit $\deg(v) = 3$.

Regel 5: Alle Knoten haben Grad 4.

Laufzeit für VERTEX COVER:

$$O(kn + 1.342^k \cdot k^2)$$