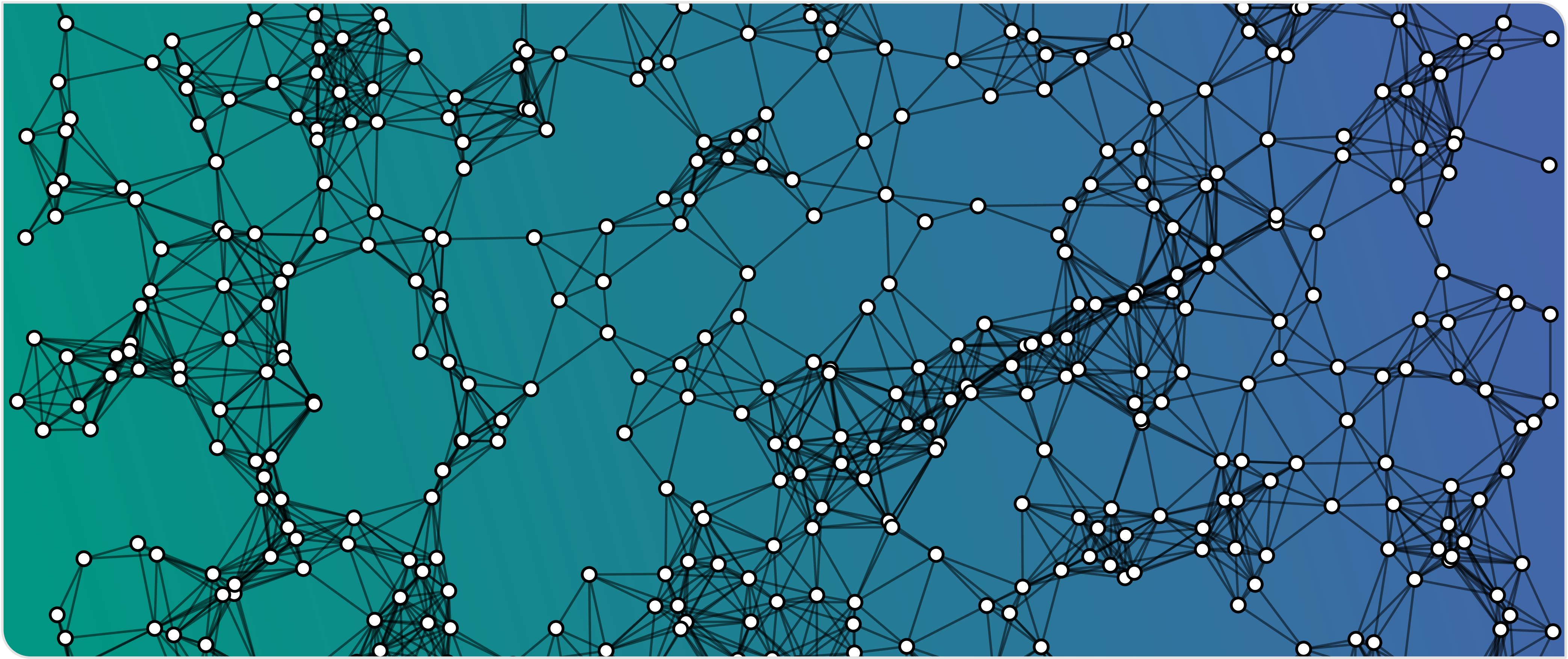
Karlsruher Institut für Technologie

Parametrisierte Algorithmen

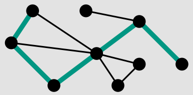
Übung 5



Kleine Rückschau

Umwege

Problem: Longest Path
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



(mit Länge meine ich hier Anzahl Knoten)

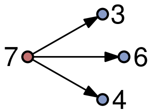
Komplexität

- NP-schwer: Reduktion von Hamiltonian Path
- heute: FPT

Für gerichtete azyklische Graphen (DAGs)

Geht das in poly Zeit?

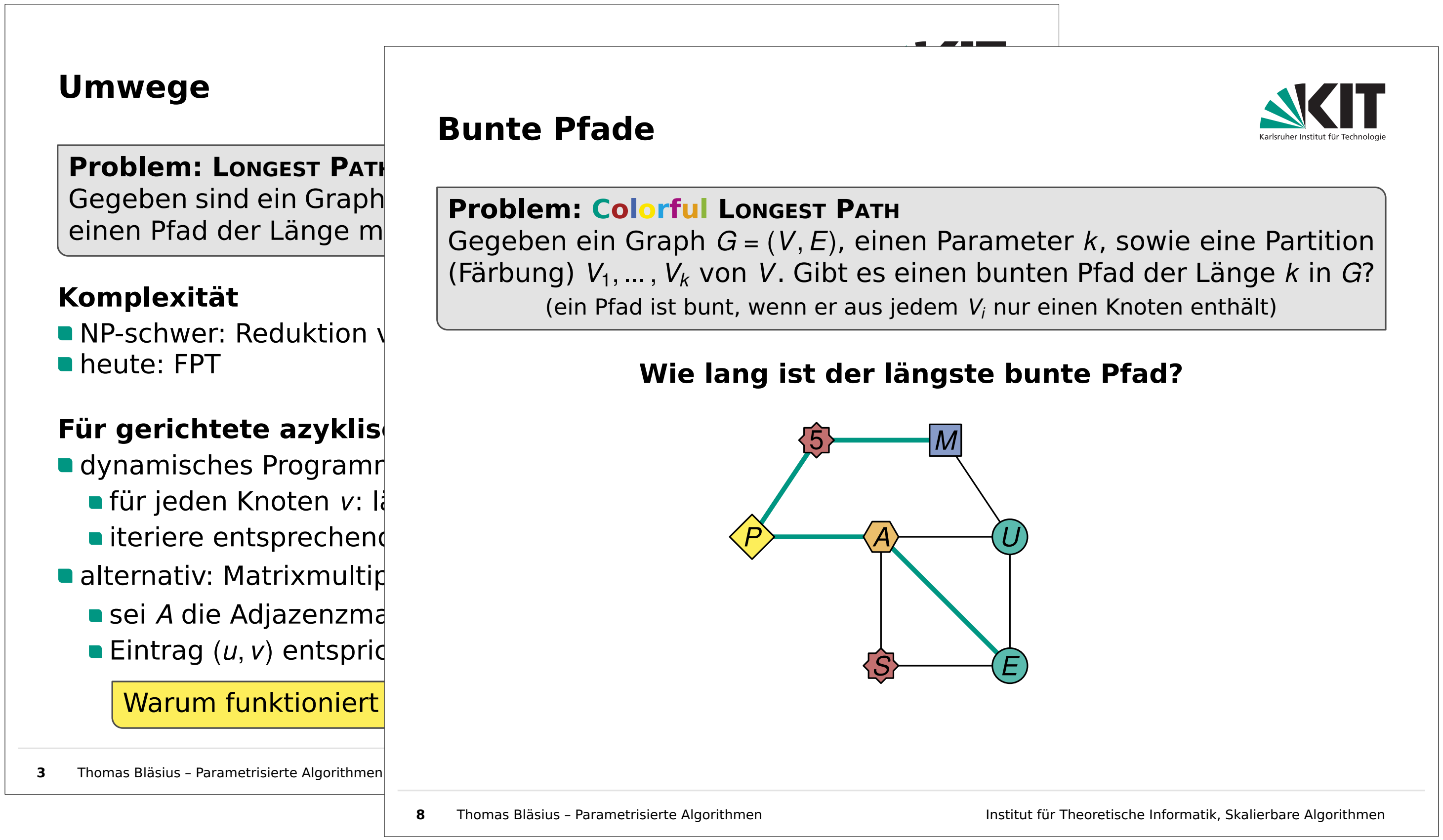
- dynamisches Programm
  - für jeden Knoten v: längster Pfad der bei v startet
  - iteriere entsprechend topologischer Sortierung



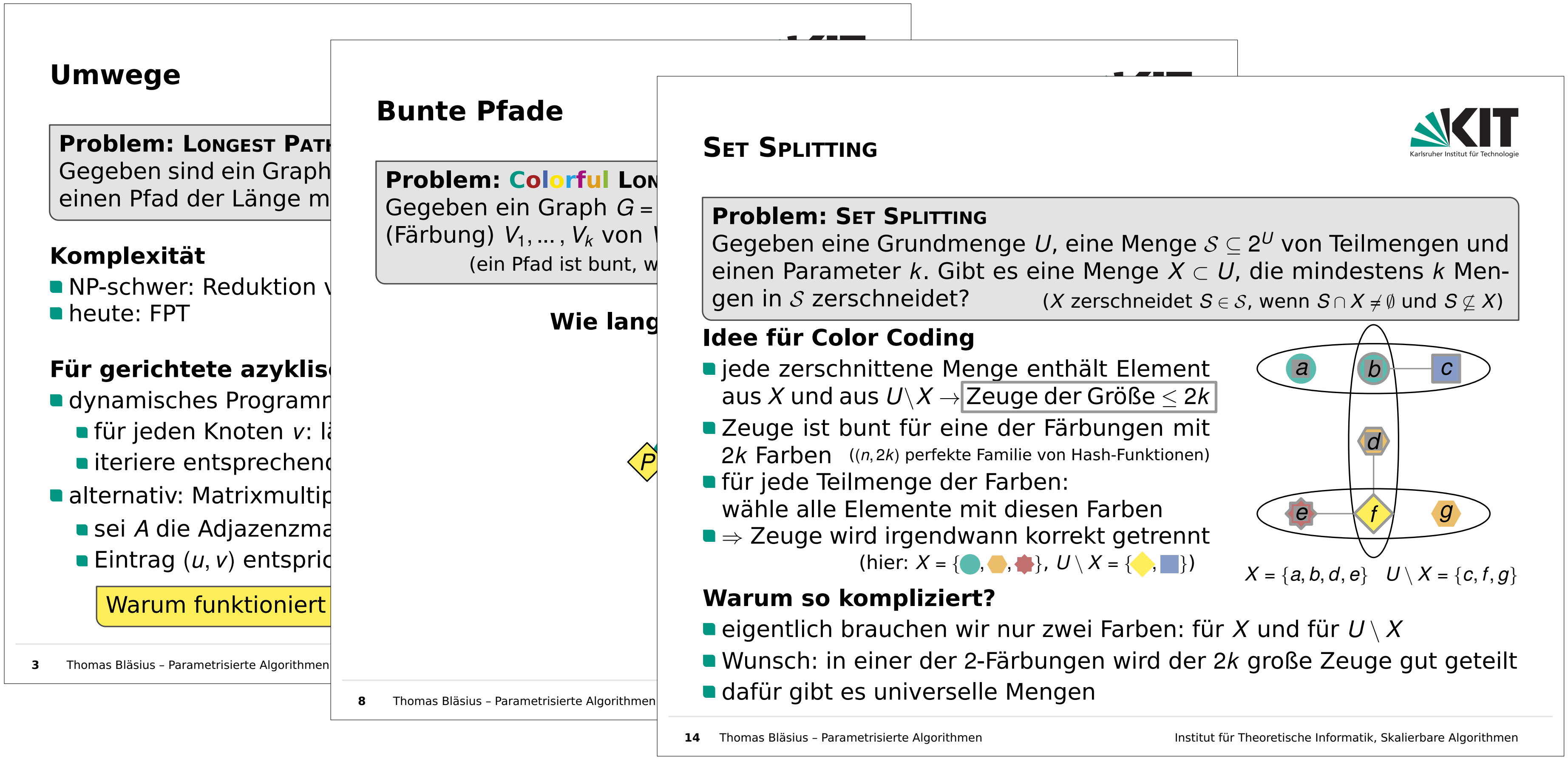
- alternativ: Matrixmultiplikation
  - sei A die Adjazenzmatrix und betrachte A^k
  - Eintrag (u, v) entspricht Anzahl Pfaden den Länge k von u nach v

Warum funktioniert das nicht auch für ungerichtete Graphen?

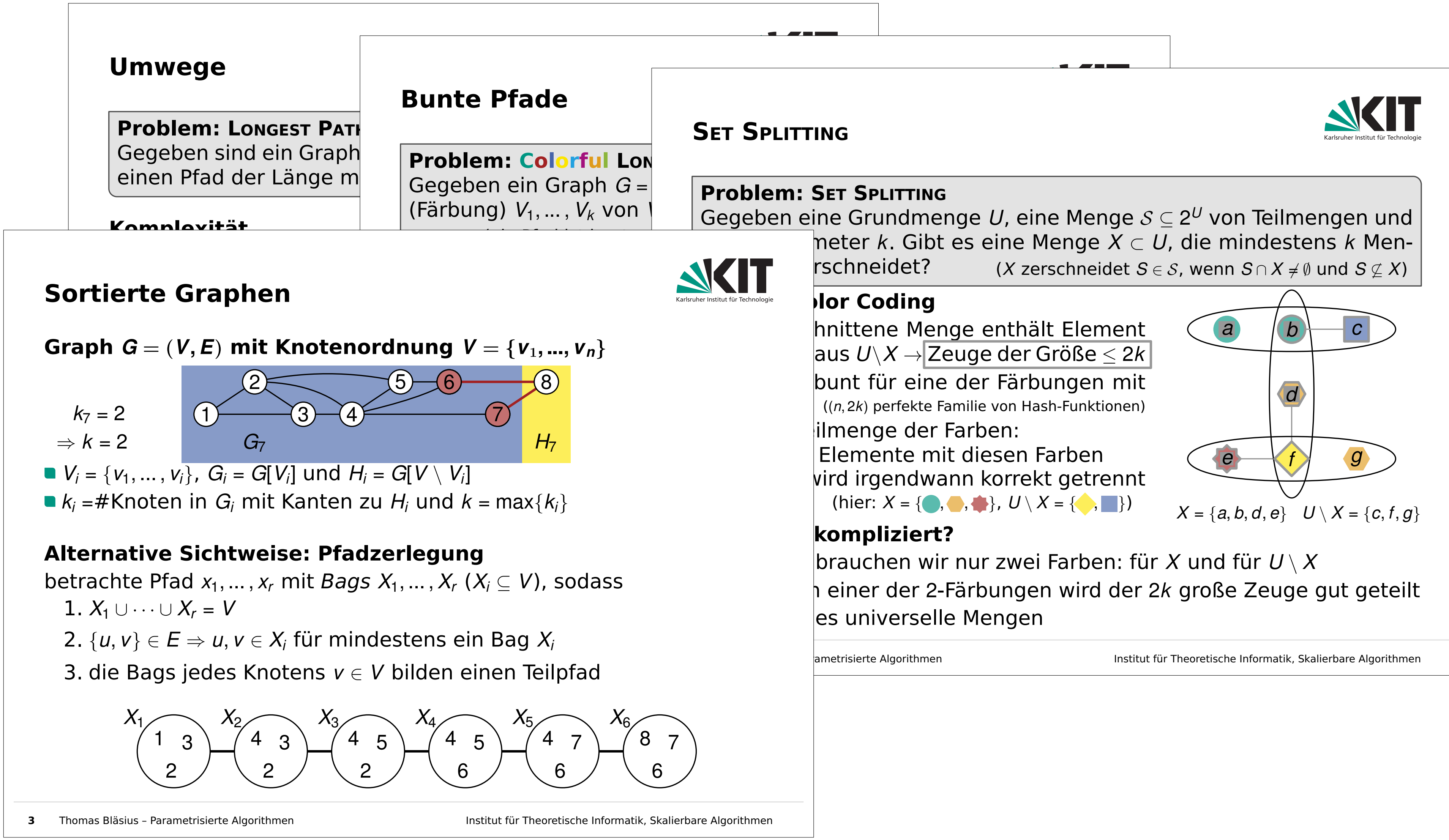
Kleine Rückschau



Kleine Rückschau



Kleine Rückschau



Umwege

Problem: LONGEST PATH
Gegeben sind ein Graph
einen Pfad der Länge m

Komplexität

Bunte Pfade

Problem: Colorful LON
Gegeben ein Graph $G =$
(Färbung) $V_1, \dots, V_k$ von

SET SPLITTING

Problem: SET SPLITTING
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und
meter k. Gibt es eine Menge $X \subset U$, die mindestens k Men-
rschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

lor Coding
hnittene Menge enthält Element
aus $U \setminus X \rightarrow \boxed{\text{Zeuge der Größe} \leq 2k}$
bunt für eine der Färbungen mit
($(n, 2k)$ perfekte Familie von Hash-Funktionen)
ilmenge der Farben:
Elemente mit diesen Farben
wird irgendwann korrekt getrennt
(hier: $X = \{$ ●, ⬢, ✸ $\}$, $U \setminus X = \{$ ◆, ■ $\}$)

$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

kompliziert?
brauchen wir nur zwei Farben: für X und für $U \setminus X$
einer der 2-Färbungen wird der $2k$ große Zeuge gut geteilt
es universelle Mengen

ametrisierte Algorithmen Institut für Theoretische Informatik, Skalierbare Algorithmen

Sortierte Graphen

Graph $G = (V, E)$ mit Knotenordnung $V = \{v_1, \dots, v_n\}$

$k_7 = 2$
$\Rightarrow k = 2$

- $V_i = \{v_1, \dots, v_i\}$, $G_i = G[V_i]$ und $H_i = G[V \setminus V_i]$
- k_i =#Knoten in G_i mit Kanten zu H_i und $k = \max\{k_i\}$

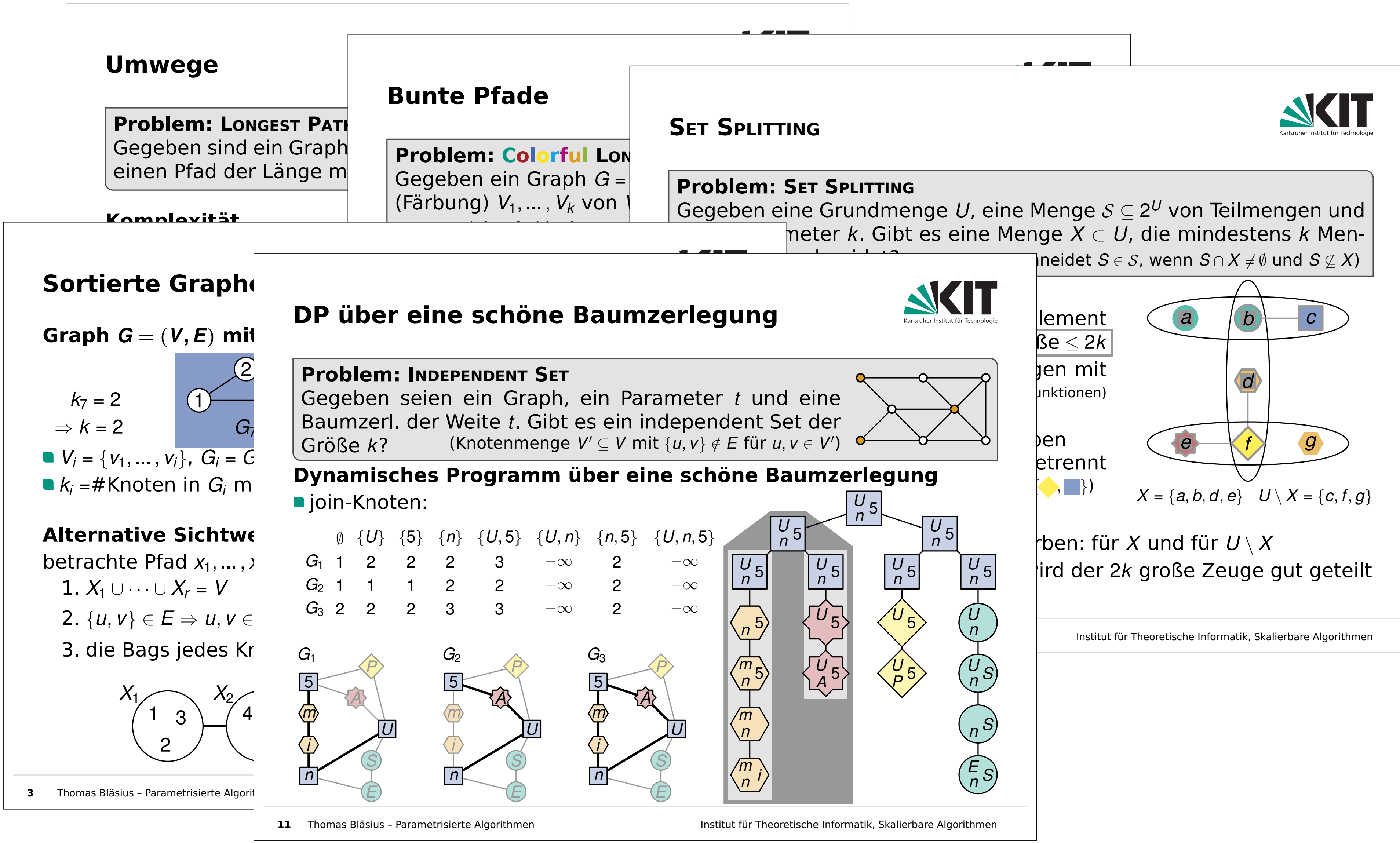
Alternative Sichtweise: Pfadzerlegung

betrachte Pfad $x_1, \dots, x_r$ mit *Bags* $X_1, \dots, X_r$ ($X_i \subseteq V$), sodass

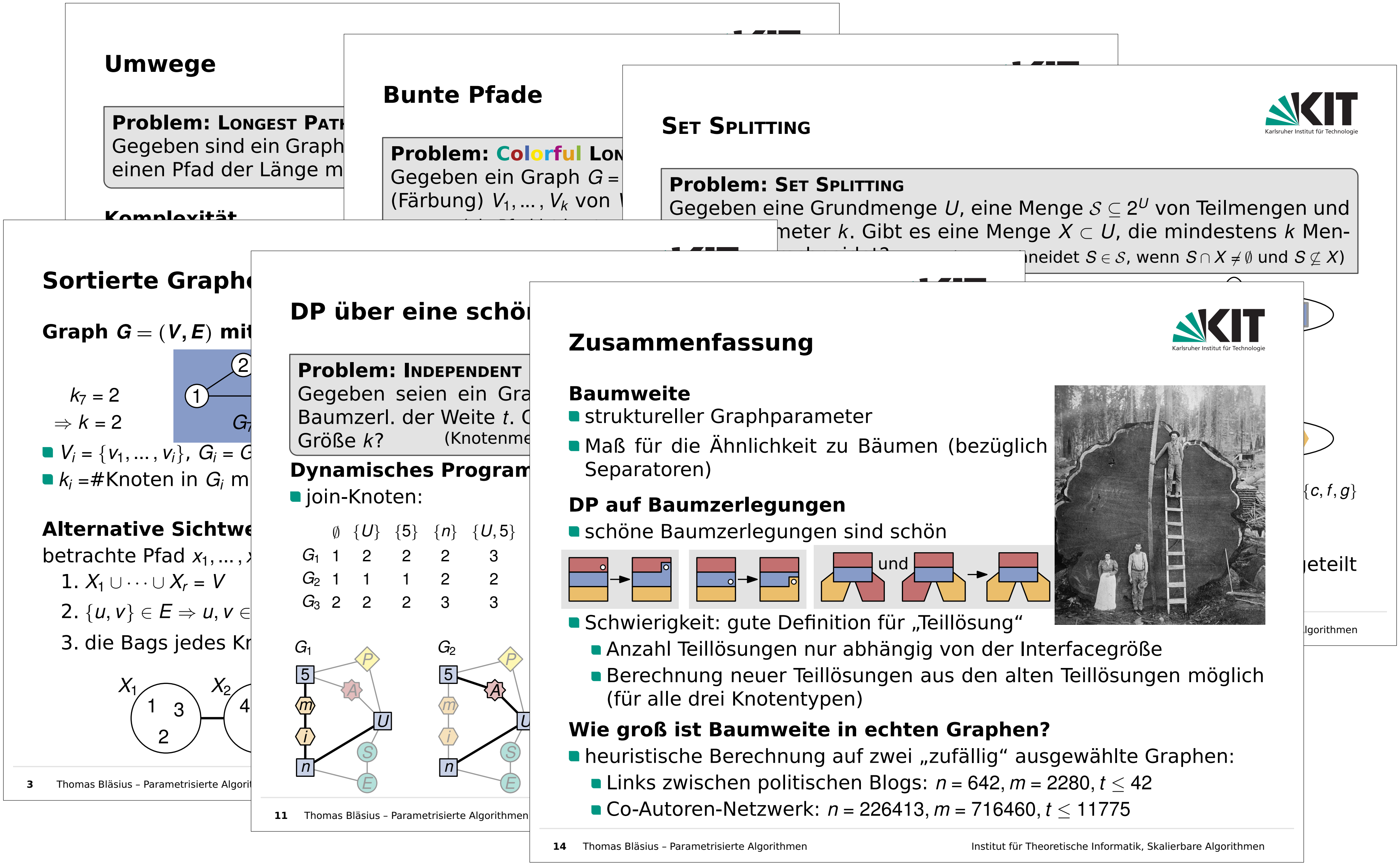
1. $X_1 \cup \dots \cup X_r = V$
2. $\{u, v\} \in E \Rightarrow u, v \in X_i$ für mindestens ein Bag X_i
3. die Bags jedes Knotens $v \in V$ bilden einen Teilpfad

3 Thomas Bläsius – Parametrisierte Algorithmen Institut für Theoretische Informatik, Skalierbare Algorithmen

Kleine Rückschau



Kleine Rückschau



Heute

Übungsblatt 4

Übungsblatt 5

Heute

Übungsblatt 4

- VERTEX COVER in bipartiten Graphen
- MAX SAT

Übungsblatt 5

Heute

Übungsblatt 4

- Vertex Cover in bipartiten Graphen
- Max Sat

Übungsblatt 5

- Sterne
- Longest Cycle
- Dominating Set auf speziellen Graphen
- Coding: Closest String

Heute

Übungsblatt 4

- Vertex Cover in bipartiten Graphen
- Max Sat

Übungsblatt 5

- Sterne
- Longest Cycle
- Dominating Set auf speziellen Graphen
- Coding: Closest String

Außerdem

- mehr zu Baumweite
- schöne Baumzerlegungen

Übungsblatt 5

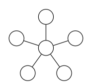
Parametrisierte Algorithmen
Wintersemester 2024/2025
`scale.iti.kit.edu`

Übungsblatt 5

Abgabe bis 15. Januar 2025

Aufgabe 1: Sterne — 8 Punkte

Ein *5-Stern* ist folgender Graph:



Gegeben einen ungerichteten Graphen $G = (V,E)$ und Parameter k soll entschieden werden, ob G mindestens k knotendisjunkte induzierte 5-Sterne enthält. Verwende *color coding* um zu zeigen, dass dieses Problem in FPT liegt.

Aufgabe 2: Longest Cycle — 12 Punkte

Gegeben sei ein Graph $G = (V,E)$ und ein Parameter k. Bei Longest Cycle geht es darum, zu entscheiden ob es in G einen Kreis der Länge *mindestens* k in G gibt. Gib einen FPT-Algorithmus für dieses Problem an.

Hinweis: Beachte, dass es Graphen gibt, die keinen Kreis der Länge genau k enthalten, aber dennoch einen Kreis der Länge mindestens k.

Aufgabe 3: Dominating Set auf speziellen Graphen — 10 Punkte

Betrachte folgende parametrisierte Variante von Dominating Set. Der Graph G ist gegeben zusammen mit einer Knotenordnung $V = \{v_1, \dots, v_n\}$. Die *Länge* einer Kante $\{v_i, v_j\}$ ist $|i - j|$. Der Parameter k ist die maximale Kantenlänge in G. Gib einen FPT-Algorithmus an, der diese Parametrisierung von Dominating Set löst.

Aufgabe 4: Closest String — 5 Bonus-Punkte

In dieser Aufgabe sollst du erneut ein Programm implementieren, das das Problem Closest String (in einer leichten Variante) löst. Gegeben sind k Strings der Länge n. Gesucht ist ein minimales D, sodass ein String s^* existiert, der zu jedem anderen String Hamming-Distanz höchstens D hat.

Löse das Problem mittels eines ILP wie in der 3. Übung. Du darfst dir gerne weitere Optimierungen überlegen. Beschreibe in der PDF-Abgabe, welche Änderungen du an der ILP Formulierung aus

1 *bitte wenden*

der Vorlesunge vorgenommen hast und ob es sonst noch Einsichten gab, die dir beim Lösen von dem Problem geholfen haben. Gib außerdem in der PDF-Abgabe für jede Instanz das kleinste D an, für das du eine Lösung gefunden hast.

Gib zusätzlich den Quellcode sowie deine gefundenen Lösungen (im unten beschriebenen Format) als eine ZIP-Datei ab. Für jede gelöste Instanz kannst du einen Punkt bekommen.

Dateiformat Eingabe: In der ersten Zeile steht k, die Anzahl der Strings. In den nächsten k Zeilen ist jeweils ein String gegeben. Alle Strings haben die selbe Länge und verwenden nur die kleinen Buchstaben `a-z`.

Dateiformat Ausgabe: Eine Zeile mit einem String s^*, der minimale Hamming-Distanz zu den gegebenen Strings hat.

Hinweis: Mithilfe der Datei `validator.py` kannst du die Hamming-Distanz deiner Lösung zu den gegebenen Strings bestimmen. Dabei wird das Maximum über alle Hamming-Distanzen ausgegeben.

Hinweis: Für diese Aufgabe benötigst du einen ILP-Solver. Die Anzahl der Programmiersprachen und Solver ist groß, aber du solltest in der Lage sein, die Aufgabe mit den meisten Kombinationen zu lösen. Einer der mit Abstand besten Solver ist `Gurobi`. Um ihn benutzen zu können, brauchst du allerdings eine Lizenz, die du kostenlos mit deiner Studenten-E-Mail erhältst.

Wir wünschen euch schöne Weihnachtsferien!

2

Übungsblatt 4

Karlsruher Institut für Technologie

Übungsblatt 4

Abgabe bis 18. Dezember 2024

Aufgabe 1: Vertex Cover in bipartiten Graphen $7 + 7 = 14$ Punkte

Teilaufgabe (a) Zeige, dass die LP-Relaxierung des ILPs zu Vertex Cover eine ganzzahlige optimale Lösung hat, wenn der Graph bipartit ist.

Hinweis: Zeige, dass die zugehörige Matrix total unimodular ist.

Teilaufgabe (b) Benutze den Dualitätssatz, um den Satz von Kőnig zu beweisen. Der Satz von Kőnig besagt, dass das minimale Vertex Cover und das maximale Matching in einem bipartiten Graphen die gleiche Kardinalität haben.

Aufgabe 2: Max Sat $5 + 11 = 16$ Punkte

Gegeben sei eine boolesche Formel φ (in KNF) mit n Variablen und m Klauseln. Bei dem Problem Max Sat soll eine Variablenbelegung gefunden werden, die möglichst viele Klauseln erfüllt.

Teilaufgabe (a) Gib sichere Reduktionsregeln an, die einen Kern mit maximal $2k$ Klauseln und k Variablen liefern, wobei k die Lösungsgröße ist.

Hinweis: Benutze den Satz von Hall, um die Anzahl der Variablen zu reduzieren.

Satz (Hall's Theorem). *Sei $G = (V_1 \cup V_2, E)$ ein bipartiter Graph. Es gibt genau dann ein Matching in G, das alle Knoten von V_1 abdeckt, wenn $|X| \leq |N(X)|$ für jede Teilmenge $X \subseteq V_1$. Andernfalls kann eine inklusionsminimale Menge $X \subseteq V_1$ mit $|X| > |N(X)|$ effizient gefunden werden.*

Teilaufgabe (b) Gib einen FPT-Algorithmus für die folgende Parametrisierung „above $\frac{m}{2}$“ mit Parameter k an: Gibt es eine Variablenbelegung, die mindestens $\frac{m}{2} + k$ Klauseln erfüllt?

Hinweis: Betrachte Klauseln mit nur einer Variable getrennt von Klauseln mit mindestens 2 Variablen und zeige zunächst, dass viele größere Klauseln dazu führen, dass es eine große Lösung gibt.

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln

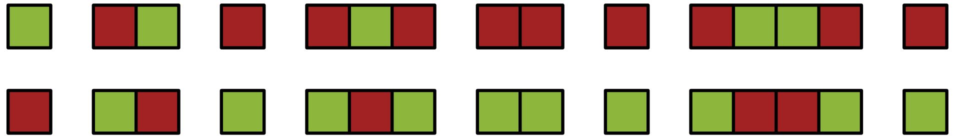
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



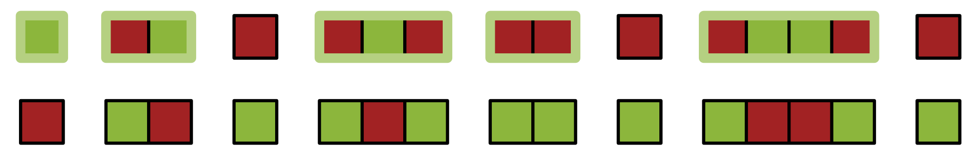
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



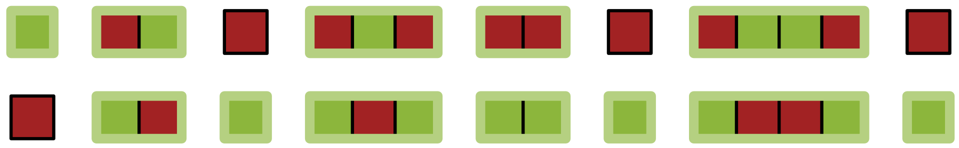
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



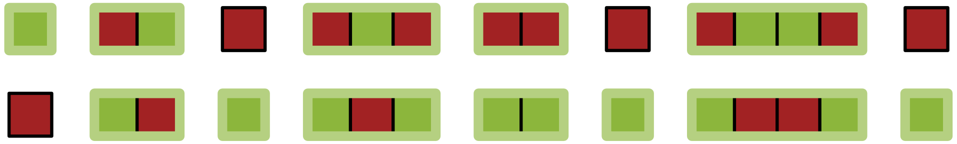
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall

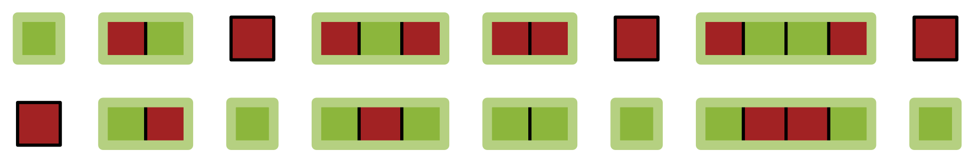
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

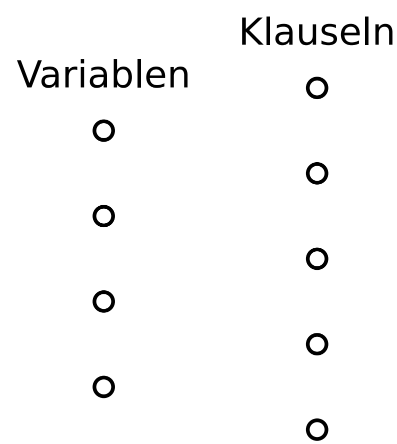
Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall



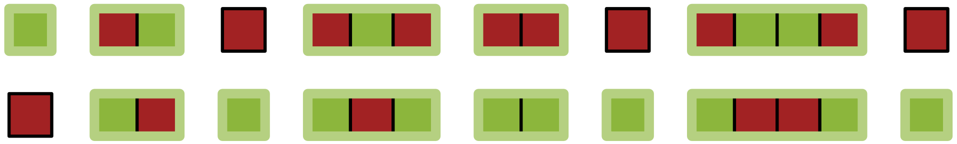
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

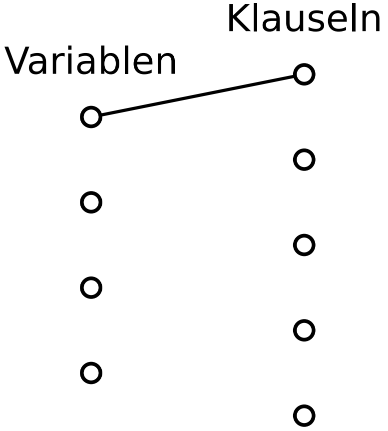
Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
    - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall



$\{v, C\} \in E$
g.d.w.
$v \in C$

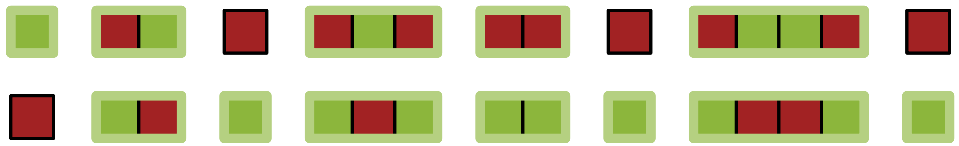
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

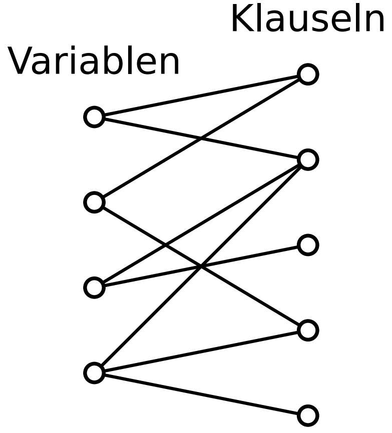
Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall



$\{v, C\} \in E$
g.d.w.
$v \in C$

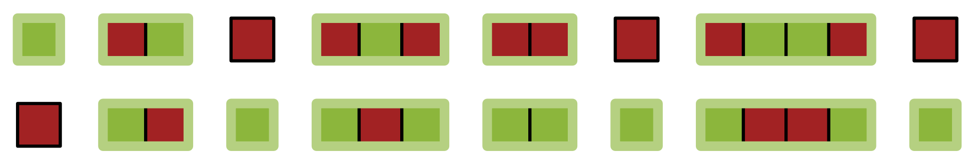
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

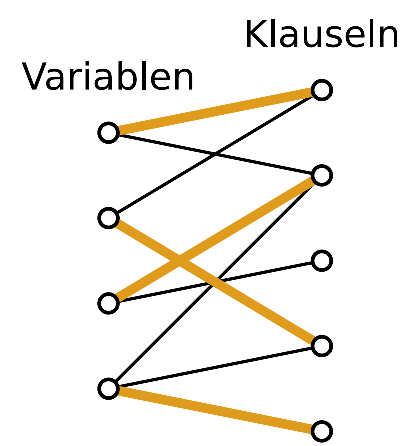
Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall
  - Falls Matching mit allen Variablen: JA-Instanz



$\{v, C\} \in E$
g.d.w.
$v \in C$

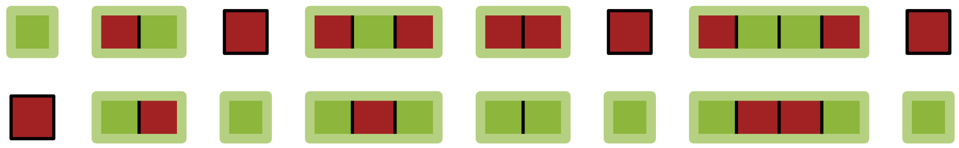
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

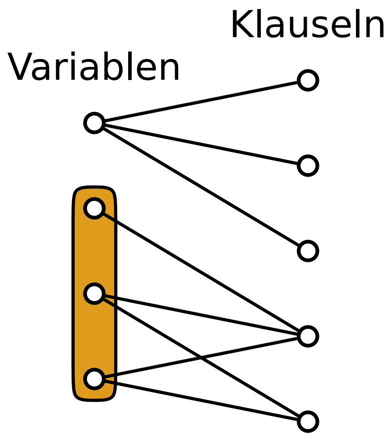
Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall
  - Falls Matching mit allen Variablen: JA-Instanz
  - Sonst: finde inkl. min. Menge von Variablen X mit $|X| > |N(X)|$



$\{v, C\} \in E$
g.d.w.
$v \in C$

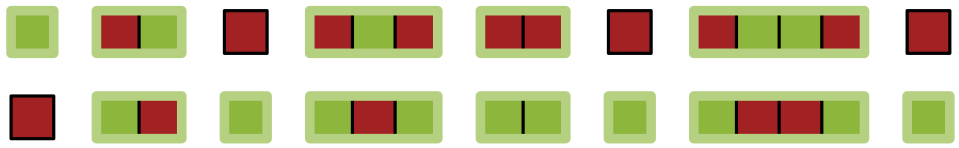
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

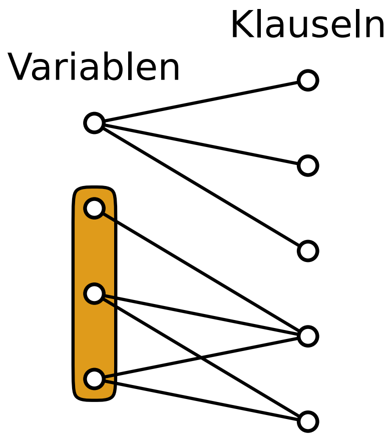
Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall
  - Falls Matching mit allen Variablen: JA-Instanz
  - Sonst: finde inkl. min. Menge von Variablen X mit $|X| > |N(X)|$
  - für $x \in X$ hat $X \setminus x$ perfektes Matching



$\{v, C\} \in E$
g.d.w.
$v \in C$

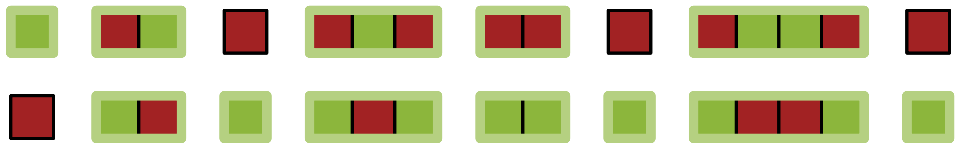
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

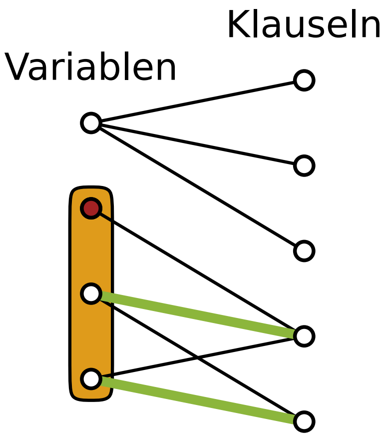
Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall
  - Falls Matching mit allen Variablen: JA-Instanz
  - Sonst: finde inkl. min. Menge von Variablen X mit $|X| > |N(X)|$
  - für $x \in X$ hat $X \setminus x$ perfektes Matching



$\{v, C\} \in E$
g.d.w.
$v \in C$

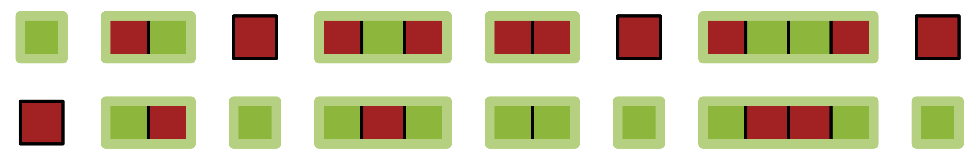
Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

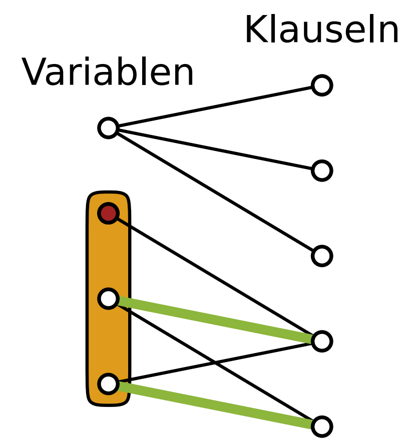
Teilaufgabe (a) Kern mit k Variablen, mit $2k$ Klauseln

Gesucht: Belegung mit mindestenz k erfüllten Klauseln

- Falls $m > 2k$: JA-Instanz:
  - Beliebige Belegung B oder ihr Komplement erfüllt mindestenz die Hälfte der Klauseln



- Falls $n > k$: Satz von Hall
  - Falls Matching mit allen Variablen: JA-Instanz
  - Sonst: finde inkl. min. Menge von Variablen X mit $|X| > |N(X)|$
  - für $x \in X$ hat $X \setminus x$ perfektes Matching
  - reduziere Instanz um X und $N(X)$



$\{v, C\} \in E$
g.d.w.
$v \in C$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2}+k-1 = \frac{m-2}{2}+k$
- Setze u := #unäre Klauseln, g := #größere Klauseln

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2}+k-1 = \frac{m-2}{2}+k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$E[X]$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right]$$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E\left[X_i\right]$$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2}+k-1 = \frac{m-2}{2}+k$
- Setze u := #unäre Klauseln, g := #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E\left[X_i\right] \geq \frac{u}{2} + \frac{3}{4}g$$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2}+k-1 = \frac{m-2}{2}+k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E\left[X_i\right] \geq \frac{u}{2} + \frac{3}{4}g = \frac{m}{2} + \frac{g}{4}$$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E\left[X_i\right] \geq \frac{u}{2} + \frac{3}{4}g = \frac{m}{2} + \frac{g}{4} \Rightarrow \tfrac{m}{2} + \tfrac{g}{4} \text{ Klauseln erfüllbar}$$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E\left[X_i\right] \geq \frac{u}{2} + \frac{3}{4}g = \frac{m}{2} + \frac{g}{4} \Rightarrow \frac{m}{2} + \frac{g}{4} \text{ Klauseln erfüllbar}$$

  - falls $\frac{g}{4} \geq k$: JA-Instanz; sonst: $g < 4k$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E\left[X_i\right] \geq \frac{u}{2} + \frac{3}{4}g = \frac{m}{2} + \frac{g}{4} \Rightarrow \tfrac{m}{2} + \tfrac{g}{4} \text{ Klauseln erfüllbar}$$

  - falls $\frac{g}{4} \geq k$: JA-Instanz; sonst: $g < 4k$
  - $m = u + g < \frac{m}{2} + k + 4k = \frac{m}{2} + 5k \Rightarrow m < 10k$

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E\left[X_i\right] \geq \frac{u}{2} + \frac{3}{4}g = \frac{m}{2} + \frac{g}{4} \Rightarrow \frac{m}{2} + \frac{g}{4} \text{ Klauseln erfüllbar}$$

  - falls $\frac{g}{4} \geq k$: JA-Instanz; sonst: $g < 4k$
  - $m = u + g < \frac{m}{2} + k + 4k = \frac{m}{2} + 5k \Rightarrow m < 10k$
  - Frage nun: sind $\frac{m}{2} + k < 6k$ Klauseln erfüllbar?

Übungsblatt 4

Gegeben: Formel φ in KNF, mit n Variablen und m Klauseln

Teilaufgabe (b) FPT Algo für mindestenz $\frac{m}{2} + k$ erfüllte Klauseln

- Reduktionsregel 1: Zwei unäre Klauseln $\{v\}$ und $\{\neg v\}$
  - entferne Klausen, suche Lösung der Größe $\frac{m}{2} + k - 1 = \frac{m-2}{2} + k$
- Setze $u :=$ #unäre Klauseln, $g :=$ #größere Klauseln
  - Nach Regel 1: $u < \frac{m}{2} + k$ (oder JA-Instanz)
- Probabilistische Methode: $E[X] \geq k \Rightarrow Pr[X \geq k] > 0$
  Betrachte Anzahl erfüllter Klauseln X bei uniform zufälliger Belegung

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E[X_i] \geq \frac{u}{2} + \frac{3}{4}g = \frac{m}{2} + \frac{g}{4} \Rightarrow \frac{m}{2} + \frac{g}{4} \text{ Klauseln erfüllbar}$$

  - falls $\frac{g}{4} \geq k$: JA-Instanz; sonst: $g < 4k$
  - $m = u + g < \frac{m}{2} + k + 4k = \frac{m}{2} + 5k \Rightarrow m < 10k$
  - Frage nun: sind $\frac{m}{2} + k < 6k$ Klauseln erfüllbar? $\Rightarrow$ Teilaufgabe (a)

Baumweite

Baumweite

Baumzerlegung für Graph $G = (V, E)$

Baum auf den Knoten $x_1, \dots, x_r$ mit Bags $X_1, \dots, X_r$, sodass

1. $X_1 \cup \dots \cup X_r = V$
2. $\{u, v\} \in E \Rightarrow u, v \in X_i$ für mindestens ein Bag X_i
3. die Bags jedes Knotens $v \in V$ bilden einen Teilbaum

Baumweite

- Weite einer Zerlegung: $\max\{|X_i|\} - 1$
- Baumweite eines Graphen: minimale Weite einer Baumzerlegung

Baumweite

Baumzerlegung für Graph $G = (V, E)$

Baum auf den Knoten $x_1, \ldots, x_r$ mit Bags $X_1, \ldots, X_r$, sodass

1. $X_1 \cup \cdots \cup X_r = V$
2. $\{u, v\} \in E \Rightarrow u, v \in X_i$ für mindestens ein Bag X_i
3. die Bags jedes Knotens $v \in V$ bilden einen Teilbaum

Baumweite

- Weite einer Zerlegung: $\max\{|X_i|\} - 1$
- Baumweite eines Graphen: minimale Weite einer Baumzerlegung

Aufgaben

- Zeige: jeder Baum mit mehr als einem Knoten hat Baumweite 1
- Zeige: für beliebige k hat das $k \times k$-Gitter Baumweite höchstens k

Baumweite

Baumzerlegung für Graph $G = (V, E)$

Baum auf den Knoten $x_1, \ldots, x_r$ mit Bags $X_1, \ldots, X_r$, sodass

1. $X_1 \cup \cdots \cup X_r = V$
2. $\{u, v\} \in E \Rightarrow u, v \in X_i$ für mindestens ein Bag X_i
3. die Bags jedes Knotens $v \in V$ bilden einen Teilbaum

Baumweite

- Weite einer Zerlegung: $\max\{|X_i|\} - 1$
- Baumweite eines Graphen: minimale Weite einer Baumzerlegung

Aufgaben

- Zeige: jeder Baum mit mehr als einem Knoten hat Baumweite 1
- Zeige: für beliebige k hat das $k \times k$-Gitter Baumweite höchstens k
- Zeige: sei C eine Clique in G; in jeder Baumzerlegung von G gibt es einen Bag X_i mit $C \subseteq X_i$

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

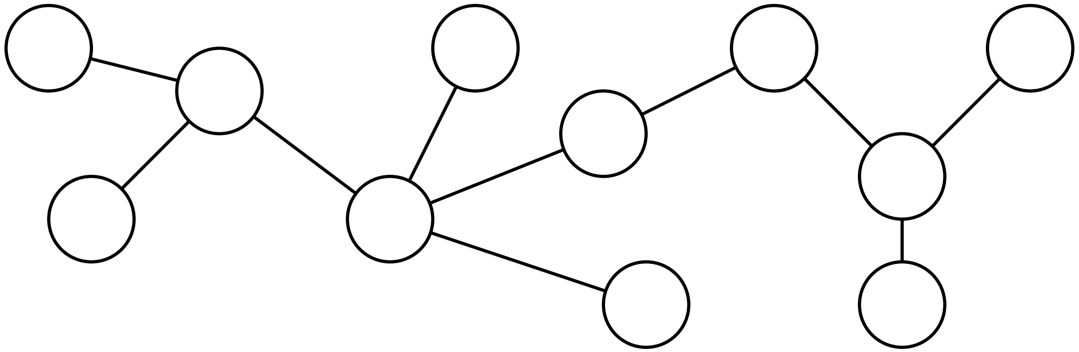
- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$
- betrachte Teilbaum $T_{C'}$ von Bags die C' enthalten, Teilbaum T_v von Bags die v enthalten

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$
- betrachte Teilbaum $T_{C'}$ von Bags die C' enthalten, Teilbaum T_v von Bags die v enthalten

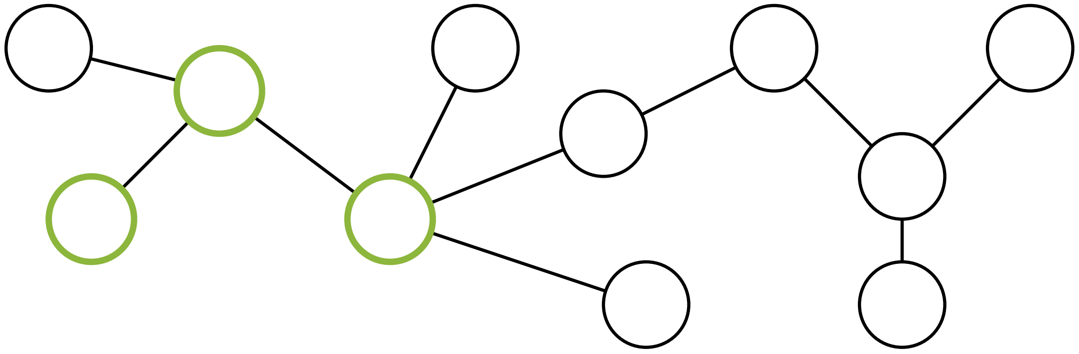


Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$
- betrachte Teilbaum $T_{C'}$ von Bags die C' enthalten, Teilbaum T_v von Bags die v enthalten

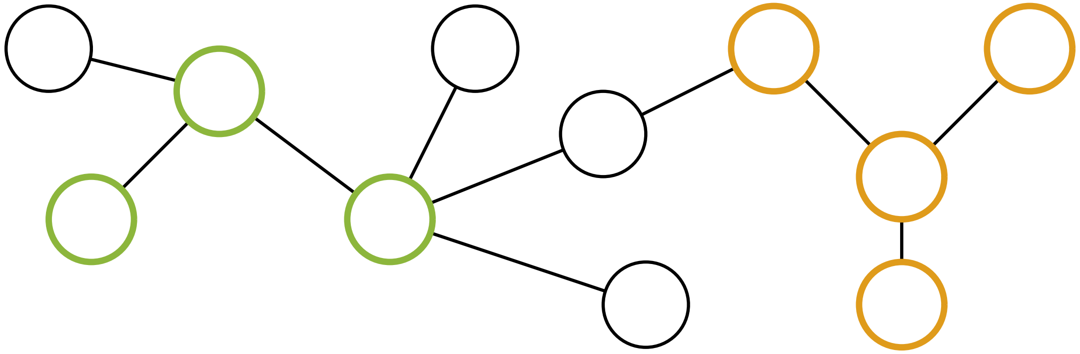


Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$
- betrachte Teilbaum $T_{C'}$ von Bags die C' enthalten, Teilbaum T_v von Bags die v enthalten

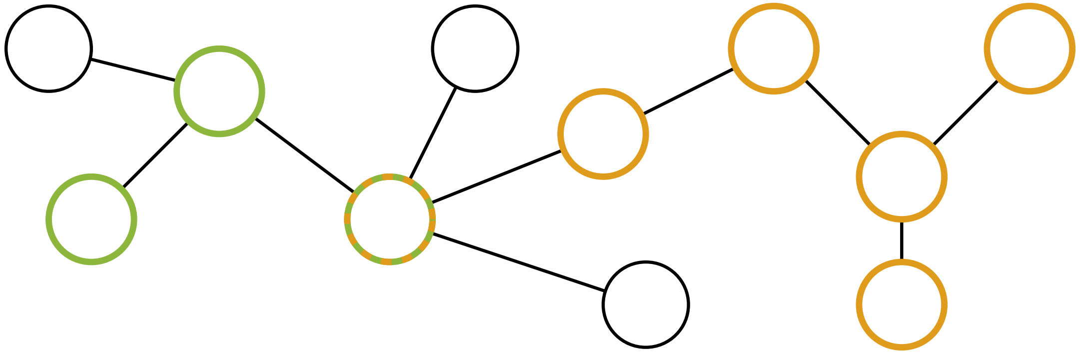


Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$
- betrachte Teilbaum $T_{C'}$ von Bags die C' enthalten, Teilbaum T_v von Bags die v enthalten
- falls $T_{C'} \cap T_v \neq \emptyset$: fertig

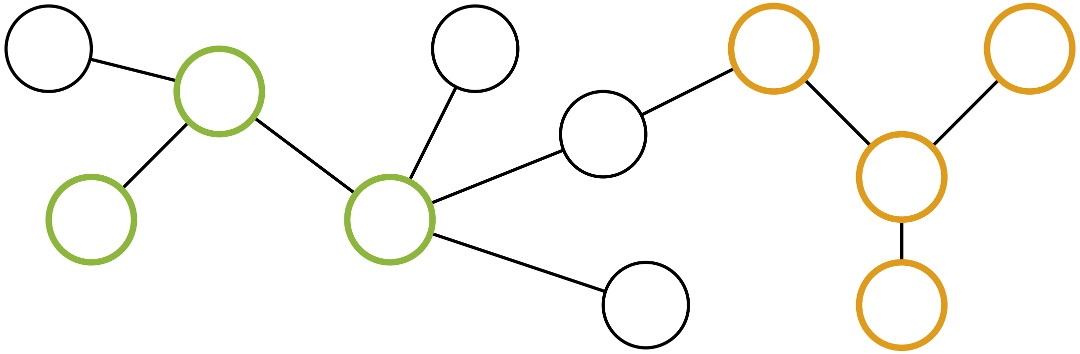


Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$
- betrachte Teilbaum $T_{C'}$ von Bags die C' enthalten, Teilbaum T_v von Bags die v enthalten



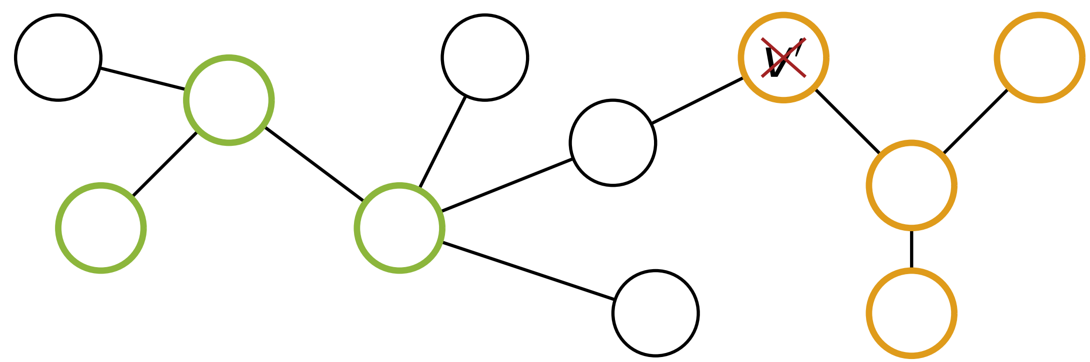
- falls $T_{C'} \cap T_v \neq \emptyset$: fertig
- sonst: ein Knoten $v' \in C'$ ist nicht im $T_{C'}$ am nächsten liegenden Bag von T_v enthalten

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$
- betrachte Teilbaum $T_{C'}$ von Bags die C' enthalten, Teilbaum T_v von Bags die v enthalten



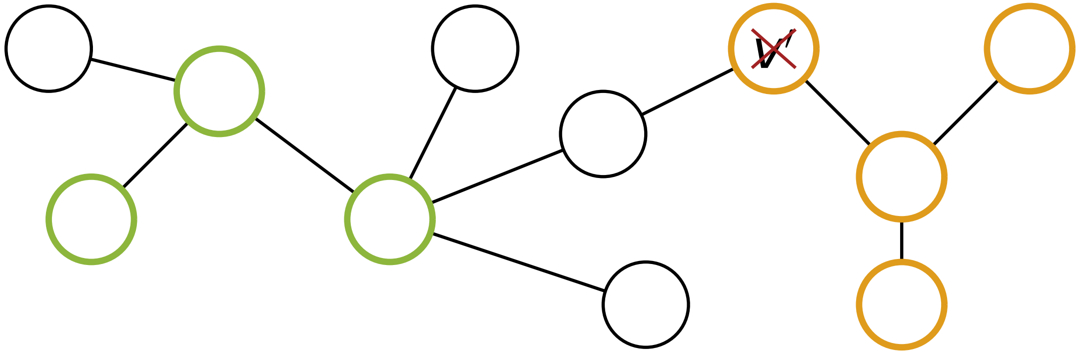
- falls $T_{C'} \cap T_v \neq \emptyset$: fertig
- sonst: ein Knoten $v' \in C'$ ist nicht im $T_{C'}$ am nächsten liegenden Bag von T_v enthalten

Jede Clique ist in einem Bag enthalten

Sei C eine Clique in G. Dann gibt es in jeder Baumzerlegung von G einen Bag X_i mit $C \subseteq X_i$

Beweis: Induktion über $|C|$

- $|C| = 2$: folgt aus Definition
- angenommen wir finden Bag für Cliquen kleiner als $|C|$
- sei $C' = C \setminus \{v\}$ für $v \in C$
- betrachte Teilbaum $T_{C'}$ von Bags die C' enthalten, Teilbaum T_v von Bags die v enthalten



- falls $T_{C'} \cap T_v \neq \emptyset$: fertig
- sonst: ein Knoten $v' \in C'$ ist nicht im $T_{C'}$ am nächsten liegenden Bag von T_v enthalten
- Widerspruch: Kante v, v' in keinem Bag enthalten!

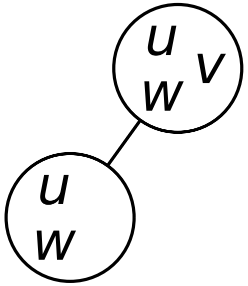
Schöne Baumzerlegungen

Drei Typen von Knoten

Schöne Baumzerlegungen

Drei Typen von Knoten

- x_i ist **introduce-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \cup \{v\}$ für ein $v \in V$



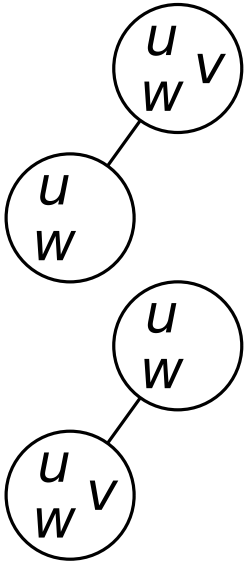
Intuition: DP läuft bottom-up

introduce v

Schöne Baumzerlegungen

Drei Typen von Knoten

- x_i ist **introduce-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \cup \{v\}$ für ein $v \in V$
- x_i ist **forget-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \setminus \{v\}$ für ein $v \in V$



Intuition: DP läuft bottom-up

introduce v

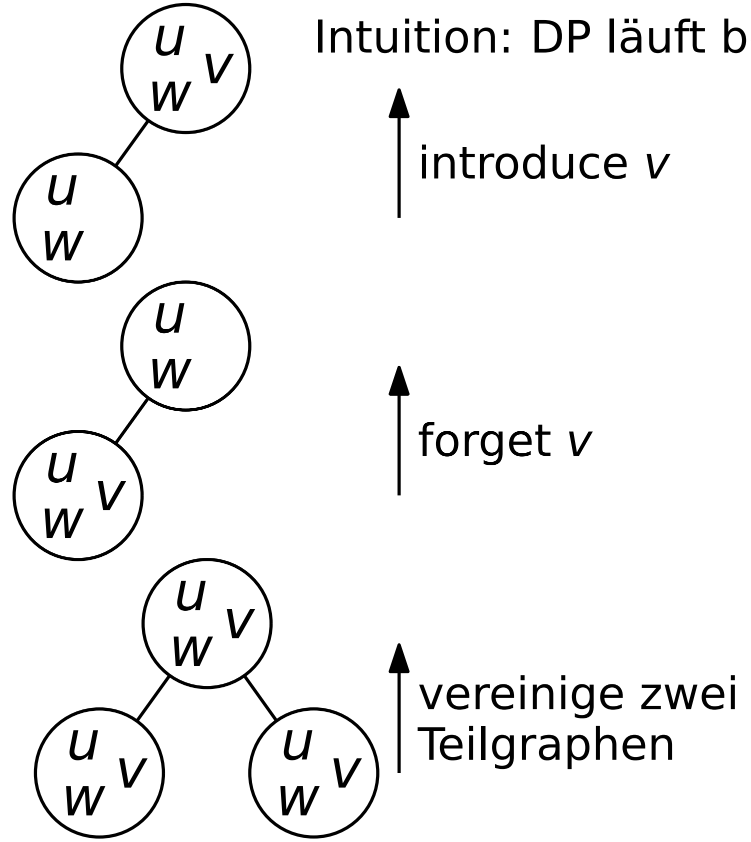
forget v

Schöne Baumzerlegungen

Drei Typen von Knoten

- x_i ist **introduce-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \cup \{v\}$ für ein $v \in V$
- x_i ist **forget-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \setminus \{v\}$ für ein $v \in V$
- x_i ist **join-Knoten** wenn
  - x_i hat genau zwei Kinder x_j, x_k
  - und $X_i = X_j = X_k$

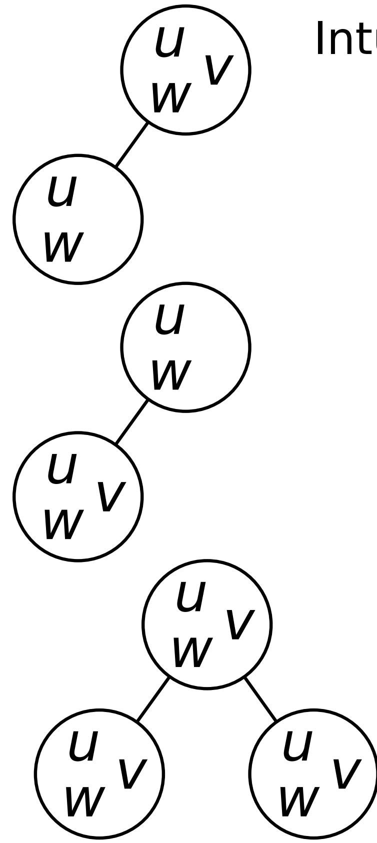
Intuition: DP läuft bottom-up



Schöne Baumzerlegungen

Drei Typen von Knoten

- x_i ist **introduce-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \cup \{v\}$ für ein $v \in V$
- x_i ist **forget-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \setminus \{v\}$ für ein $v \in V$
- x_i ist **join-Knoten** wenn
  - x_i hat genau zwei Kinder x_j, x_k
  - und $X_i = X_j = X_k$



Intuition: DP läuft bottom-up

introduce v

forget v

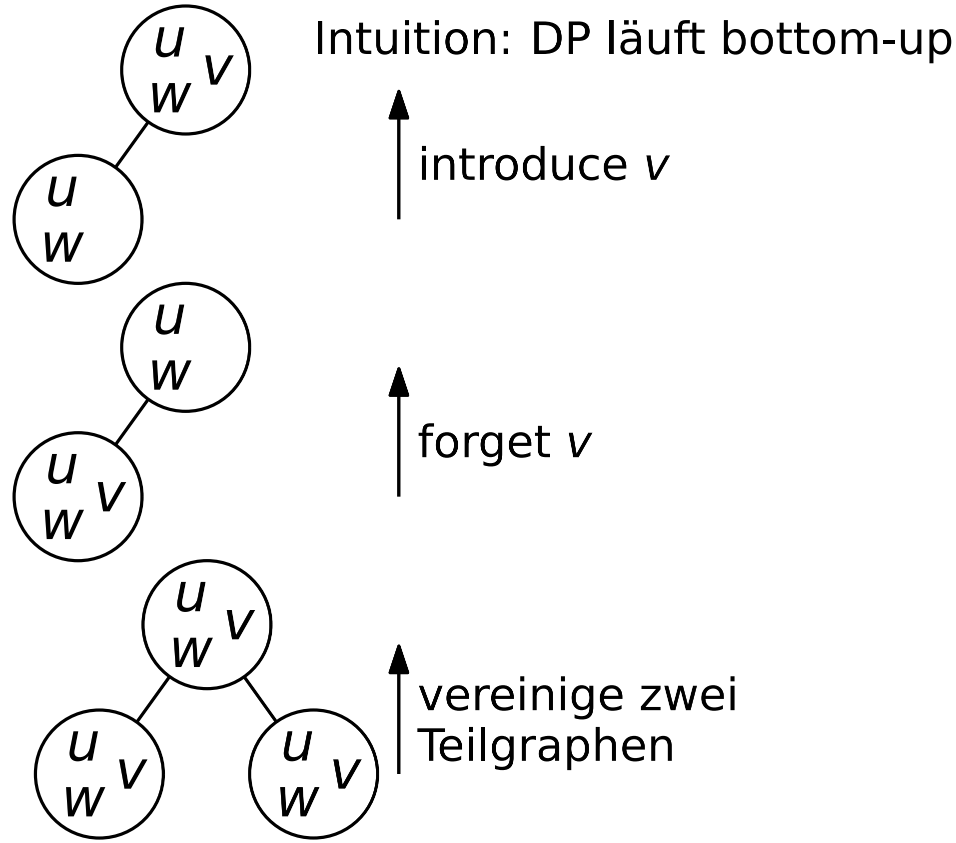
vereinige zwei Teilgraphen

> **Theorem**
> Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t.

Schöne Baumzerlegungen

Drei Typen von Knoten

- x_i ist **introduce-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \cup \{v\}$ für ein $v \in V$
- x_i ist **forget-Knoten** wenn
  - x_i hat genau ein Kind x_j und
  - $X_i = X_j \setminus \{v\}$ für ein $v \in V$
- x_i ist **join-Knoten** wenn
  - x_i hat genau zwei Kinder x_j, x_k
  - und $X_i = X_j = X_k$



Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis: TO-DO

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)
  - für keine zwei benachbarten Bags X_i, X_j gilt $X_i \subseteq X_j$ oder $X_j \subseteq X_i$

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)
  - für keine zwei benachbarten Bags X_i, X_j gilt $X_i \subseteq X_j$ oder $X_j \subseteq X_i$
  - nicht redundante Baumzerlegung hat höchstens n Knoten

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)
  - für keine zwei benachbarten Bags X_i, X_j gilt $X_i \subseteq X_j$ oder $X_j \subseteq X_i$
  - nicht redundante Baumzerlegung hat höchstens n Knoten
- hochgradige Bags können durch Binärbaum ersetzt werden

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)
  - für keine zwei benachbarten Bags X_i, X_j gilt $X_i \subseteq X_j$ oder $X_j \subseteq X_i$
  - nicht redundante Baumzerlegung hat höchstens n Knoten
- hochgradige Bags können durch Binärbaum ersetzt werden
  - Binärbaum mit n Blättern: $n - 1$ innere Knoten

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)
  - für keine zwei benachbarten Bags X_i, X_j gilt $X_i \subseteq X_j$ oder $X_j \subseteq X_i$
  - nicht redundante Baumzerlegung hat höchstens n Knoten
- hochgradige Bags können durch Binärbaum ersetzt werden
  - Binärbaum mit n Blättern: $n - 1$ innere Knoten
- erhalten nicht redundante Baumzerlegung mit Maximalgrad 3

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)
  - für keine zwei benachbarten Bags X_i, X_j gilt $X_i \subseteq X_j$ oder $X_j \subseteq X_i$
  - nicht redundante Baumzerlegung hat höchstens n Knoten
- hochgradige Bags können durch Binärbaum ersetzt werden
  - Binärbaum mit n Blättern: $n - 1$ innere Knoten
- erhalten nicht redundante Baumzerlegung mit Maximalgrad 3
  - Anzahl Knoten: höchstens 2n

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)
  - für keine zwei benachbarten Bags X_i, X_j gilt $X_i \subseteq X_j$ oder $X_j \subseteq X_i$
  - nicht redundante Baumzerlegung hat höchstens n Knoten
- hochgradige Bags können durch Binärbaum ersetzt werden
  - Binärbaum mit n Blättern: $n - 1$ innere Knoten
- erhalten nicht redundante Baumzerlegung mit Maximalgrad 3
  - Anzahl Knoten: höchstens 2n
- Wurzeln und benachbarte Bags durch Pfad mit Insert- und Forget-Knoten ersetzten

Schöne Baumzerlegungen: Beweis

Theorem
Wenn G eine Baumzerlegung mit Weite t hat, dann hat G auch eine schöne Baumzerlegung mit Weite t und $O(t \cdot n)$ Knoten.

Beweis:

- betrachte *nicht redundante* Baumzerlegung von G (Weite t)
  - für keine zwei benachbarten Bags X_i, X_j gilt $X_i \subseteq X_j$ oder $X_j \subseteq X_i$
  - nicht redundante Baumzerlegung hat höchstens n Knoten
- hochgradige Bags können durch Binärbaum ersetzt werden
  - Binärbaum mit n Blättern: $n - 1$ innere Knoten
- erhalten nicht redundante Baumzerlegung mit Maximalgrad 3
  - Anzahl Knoten: höchstens 2n
- Wurzeln und benachbarte Bags durch Pfad mit Insert- und Forget-Knoten ersetzten
  - Anzahl Knoten: in $O(t \cdot n)$