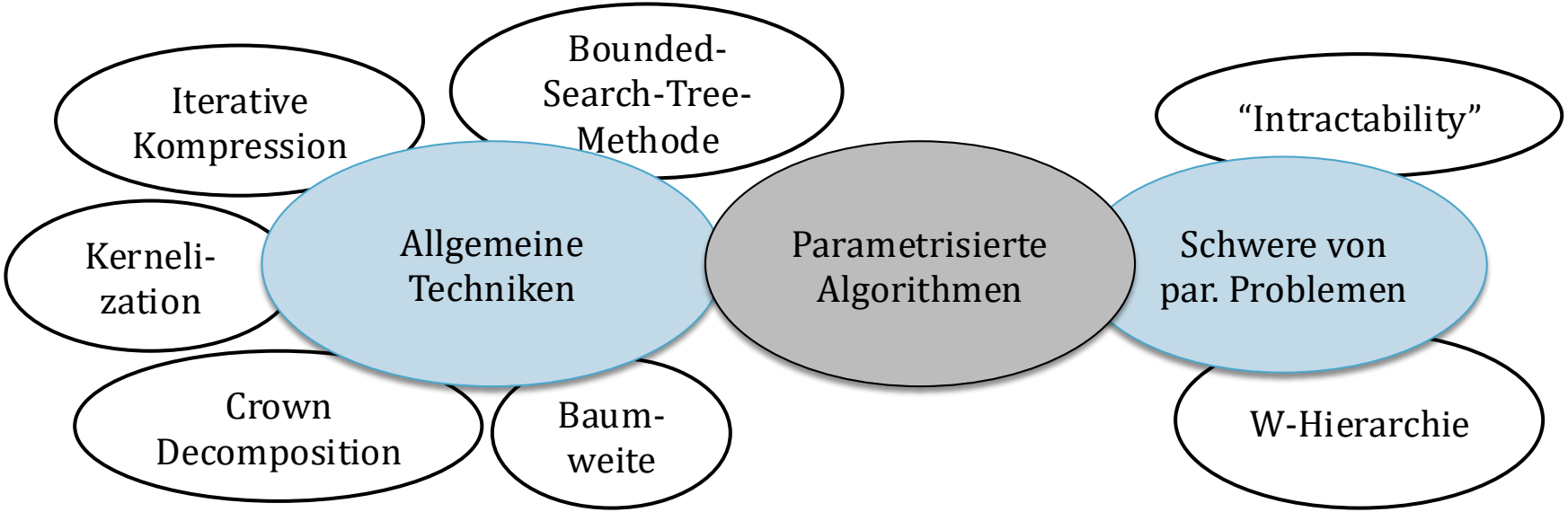
Technische Universität Braunschweig

Parametrisierte Algorithmen

Arne Schmidt

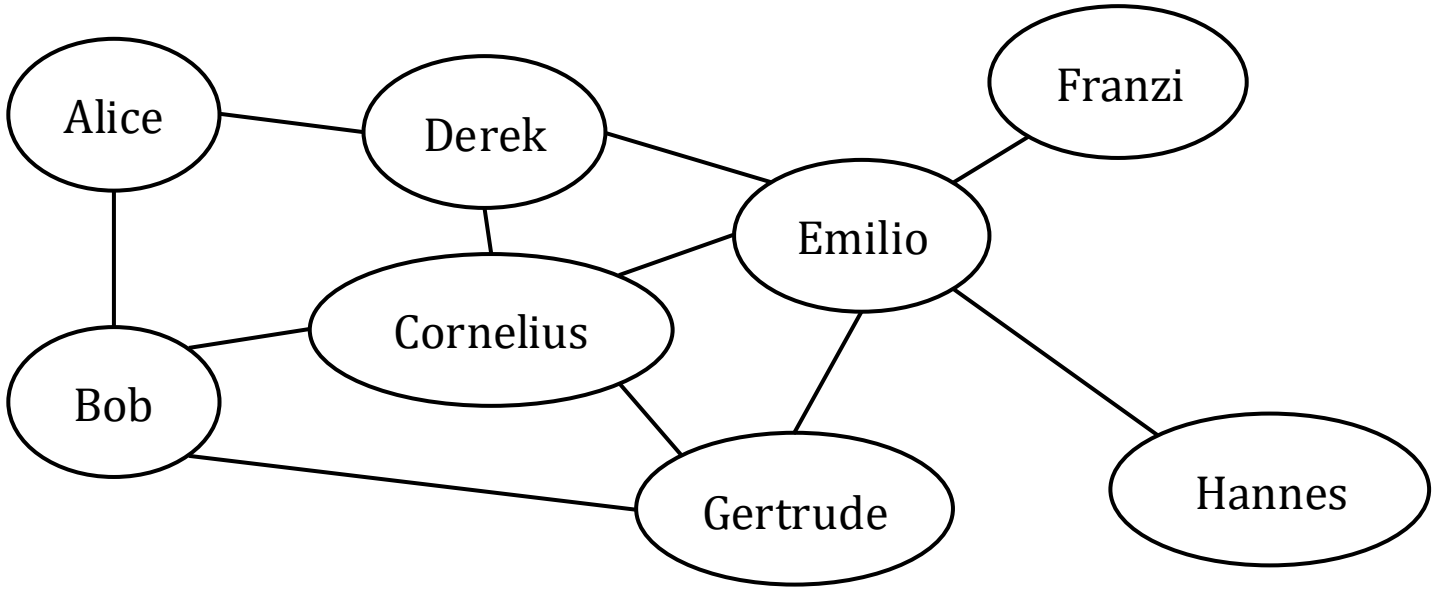
Inhalte der Vorlesung



Kapitel 1.1 – Bar Fight Problem

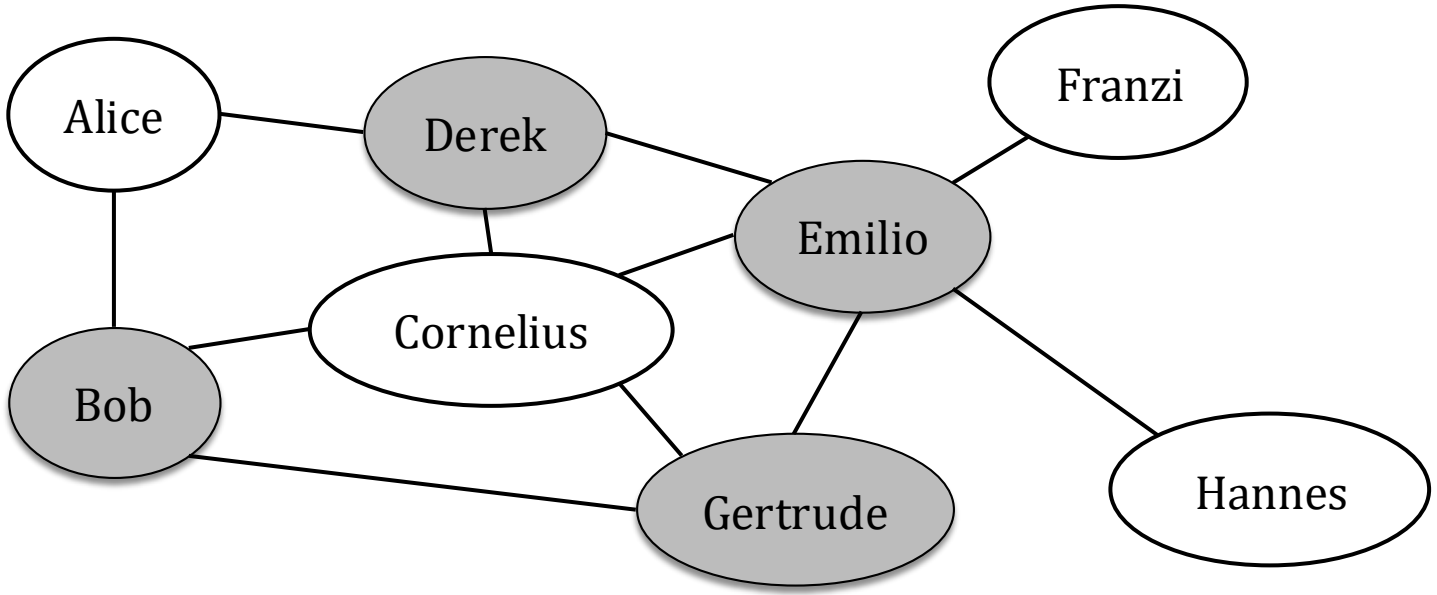
Ein Graph

Genügt es, 4 Knoten zu entfernen, um alle Konflikte aufzulösen?



Ein Graph

Genügt es, 4 Knoten zu entfernen, um alle Konflikte aufzulösen?

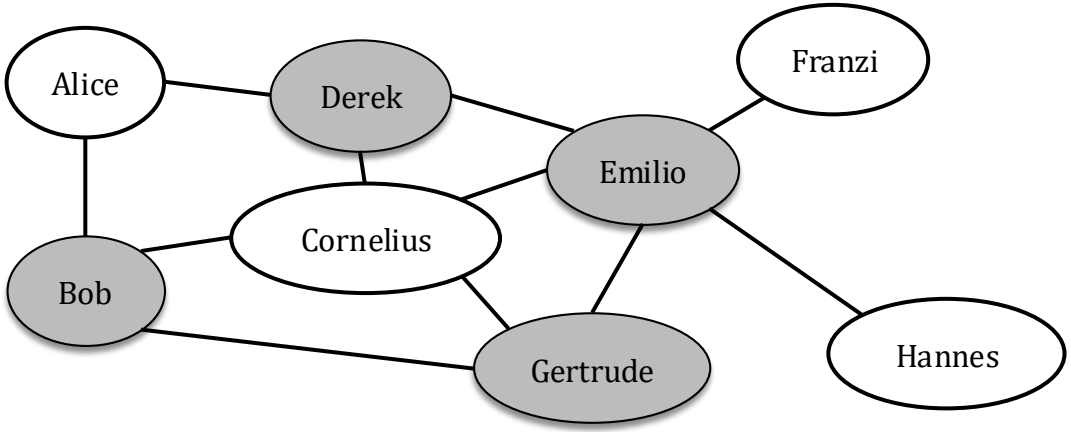


Welche Ideen gibt es, um das Problem algorithmisch (für beliebige Graphen) zu lösen?

Das Bar-Fight-Problem

Problem 1.1 (Bar Fight Problem, aka Vertex Cover)
Gegeben: Ein Graph $G = (V, E)$ und eine Zahl $k \in \mathbb{N}$.
Frage: Existiert $C \subseteq V$ mit

- $|C| = k$
- $\forall \{u, v\} \in E : u \in C \vee v \in C$



Kapitel 1.3 – Formale Definitionen

Parametrisierte Probleme

Definition 1.5
Ein **parametrisiertes Problem** ist eine Sprache $L \subseteq \Sigma^* \times \mathbb{N}$, wobei Σ ein beliebiges, aber festes Alphabet ist.

Für eine Instanz $(x, k) \in \Sigma^* \times \mathbb{N}$ wird k der **Parameter des Problems** genannt.

Beispiel:

- $L = \{(G, k) \mid G$ ist ein Graph mit einem Vertex Cover der Größe $\leq k. \}$.
  "Vertex Cover parametrisiert mit der Größe des Vertex Covers"
- $L = \{(G_k, \Delta) \mid G$ ist ein Graph mit Maximalgrad Δ und Vertex Cover der Größe $\leq k. \}$.
  "Vertex Cover parametrisiert mit der Größe des Maximalgrades."

Komplexitätsklasse FPT

Definition 1.6
Ein parametrisiertes Problem $L \subseteq \Sigma^* \times \mathbb{N}$ heißt **fixed-parameter tractable**, wenn ein Algorithmus A, eine berechenbare Funktion $f: \mathbb{N} \to \mathbb{N}$ und eine Konstante c existieren, sodass für ein gegebenes $(x, k) \in \Sigma^* \times \mathbb{N}$ gilt:

- A entscheidet korrekt, ob $(x, k) \in L$
- Die Laufzeit von A liegt in $O(f(k) \cdot |(x, k)|^c)$

Die Klasse aller fixed-parameter tractable Probleme heißt **FPT**.

Das Vertex-Cover-Problem parametrisiert mit der Größe k des Vertex Covers ist in FPT, denn es gibt einen Algorithmus, der das parametrisierte Problem

- Korrekt entscheidet
- Laufzeit $O\left(2^k \cdot k^2 + n + m\right)$ besitzt

Komplexitätsklasse XP

Definition 1.7
Ein parametrisiertes Problem $L \subseteq \Sigma^* \times \mathbb{N}$ heißt **stückweise polynomiell** wenn ein Algorithmus A und zwei berechenbare Funktionen $f, g: \mathbb{N} \to \mathbb{N}$ existieren, sodass für ein gegebenes $(x, k) \in \Sigma^* \times \mathbb{N}$ gilt:

- A entscheidet korrekt, ob $(x, k) \in L$
- Die Laufzeit von A liegt in $O\left(f(k) \cdot |(x, k)|^{g(k)}\right)$

Die Klasse aller stückweise polynomiellen Probleme heißt **XP**.

Korollar 1.8
FPT $\subseteq$ XP.

Kapitel 2 – Kernelisation

Reduktionsregeln

Definition 2.1
Eine **Reduktionsregel** (reduction rule) für ein parametrisiertes Problem Q ist eine Funktion $\phi: \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, welche eine Instanz (I, k) von Q auf eine äquivalente Instanz (I', k') von Q abbildet.

Die Funktion ϕ muss in polynomieller Zeit in $|I| + k$ berechnbar sein.

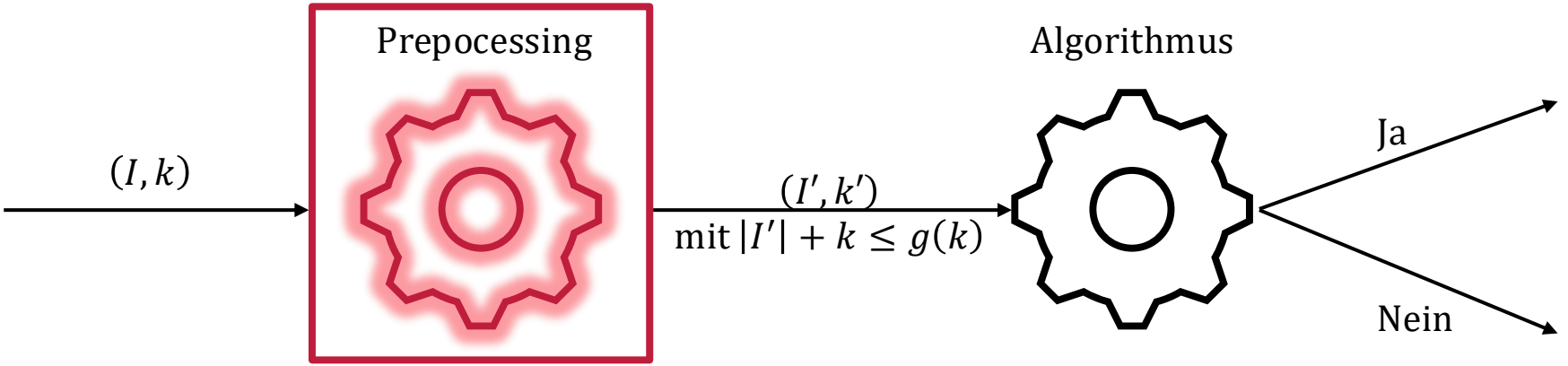
Definition 2.2
Die **Ausgabegröße** des Prepocessingalgorithmus $\mathcal{A}$ ist die Funktion $size_{\mathcal{A}}: \mathbb{N} \to \mathbb{N} \cup \{\infty\}$ mit

$$size_{\mathcal{A}}(k) = \sup\{\, |I'| + k' \mid (I', k') = \mathcal{A}(I, k) \text{ und } I \in \Sigma^* \,\}$$

Ein **Kernel** für ein parametrisiertes Problem Q ist ein Algorithmus $\mathcal{A}$, welcher für eine Instanz (I, k) von Q in polynomieller Zeit eine äquivalente Instanz (I', k') von Q zurückgibt, sodass $size_{\mathcal{A}}(k) \leq g(k)$ für eine berechenbare Funktion $g: \mathbb{N} \to \mathbb{N}$ gilt.

Parametrisierter Algorithmus mit Kernelisation

Theorem 2.3
Ein parametrisiertes Problem Q ist genau dann in FPT, wenn es einen Kernel besitzt.



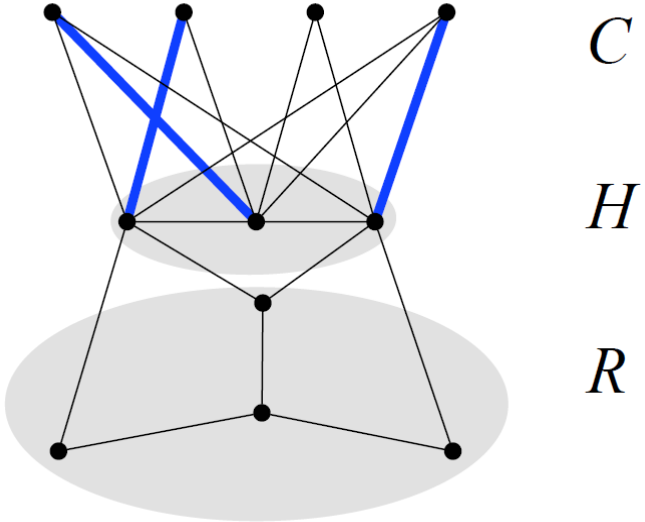
i.d.R Anwenden von Reduktionsregeln

Kapitel 2.4 – Crown Decomposition

Crown Decomposition

Definition 2.25
Eine Kronzerlegung (engl. **Crown Decomposition**) eines Graphen $G = (V, E)$ ist eine Zerlegung von V in drei Mengen $C, H, R \subseteq V$, sodass

1. $C \neq \emptyset$
2. $\forall u, v \in C : \{u, v\} \notin E$
3. $\forall u \in R, v \in C : \{u, v\} \notin E$
4. Sei $E' = \{\{u, v\} \mid u \in C, v \in H\}$, dann existiert ein H überdeckendes Matching.



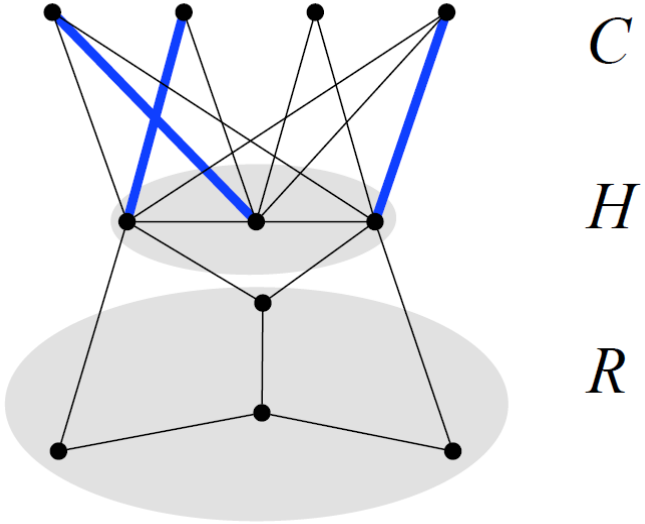
Bemerkung:
Eine Zerlegung einer Menge teilt die Menge in disjunkte Mengen.

Crown Decomposition

Lemma 2.26 (Kronen-Lemma)
Sei $G = (V, E)$ ein Graph ohne isolierte Knoten und $|V| \geq 3k + 1$. Es existiert ein Poly.-Zeit-Algorithmus, welcher

1. entweder ein Matching der Größe $k + 1$,
2. oder eine Crown Decomposition von G

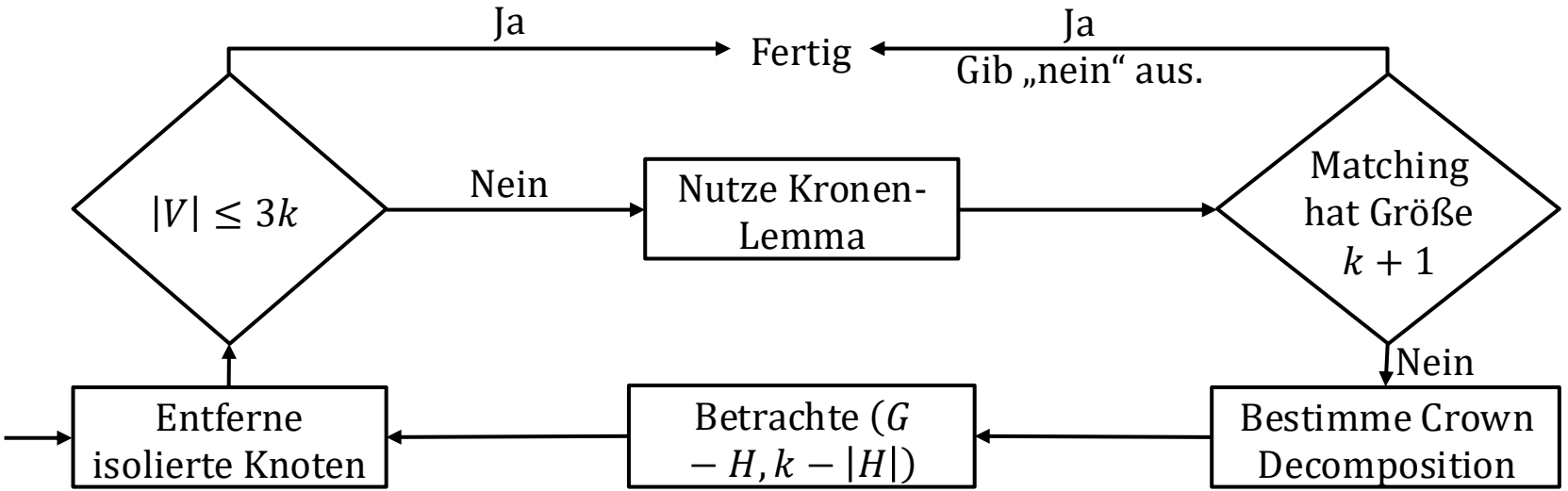
bestimmt.



Neuer Kernel für Vertex Cover

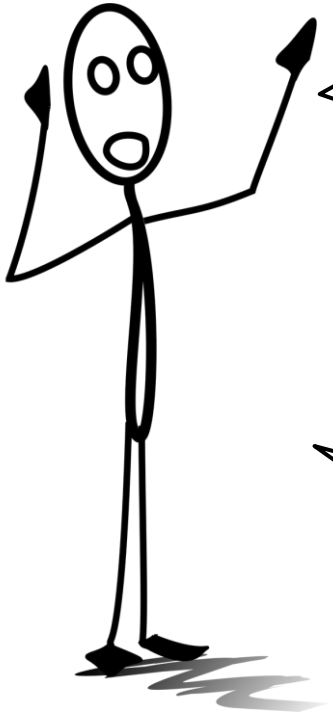
Theorem 2.27
Vertex Cover besitzt einen Kernel der Größe $3k$.

Beweisskizze:



Kapitel 3 – Bounded Tree Search

Allgemeine Idee



Für eine gegebene Instanz (I, k), triff $O\big(f(k)\big)$ viele Entscheidungen, unter den Bedingungen:

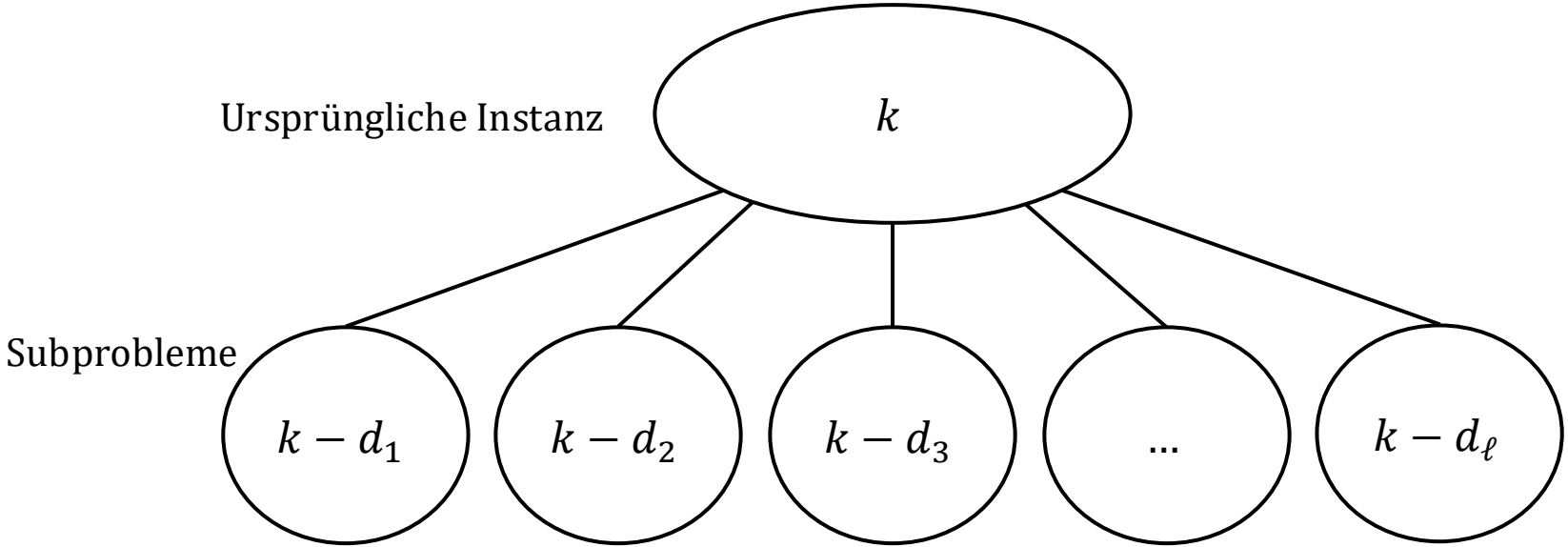
- $f: \mathbb{N} \to \mathbb{N}$ ist eine berechenbare Funktion.
- k für jede Instanz der Entscheidung kleiner wird
- I eine ja-Instanz ist, gdw. unter allen Entscheidungen eine ja-Instanz ist.
- Die Entscheidungen in poly. Zeit im Input berechnet werden können.

Damit ergibt sich eine Laufzeit von:

$$O\big(\nu(k) \cdot \text{poly}(|I|, k)\big)$$

Dabei ist $\nu(k)$ die Anzahl besuchter Knoten im Entscheidungsbaum.

Branching Algorithmen



Definition 3.3:
Wird der branching Algorithmus rekursiv $\ell \geq 2$ Mal mit Parametern $k - d_i, i \in \{1, \dots, \ell\}$ aufgerufen, so heißt $(d_1, \dots, d_\ell)$ der **Branching-Vektor** der Rekursion

Laufzeiten

Lemma 3.4
Sei $(d_1, d_2, \dots, d_\ell)$ ein Branching-Vektor eines rekursiven Branching-Algorithmus $\mathcal{A}$. Dann ist die Laufzeit von $\mathcal{A}$ beschränkt durch $T(k) \in O\left(\lambda_0^k\right)$, wobei λ_0 die Nullstelle des **charakteristischen Polynoms**

$$P(\lambda) = \lambda^d - \lambda^{d-d_1} - \dots - \lambda^{d-d_\ell}$$

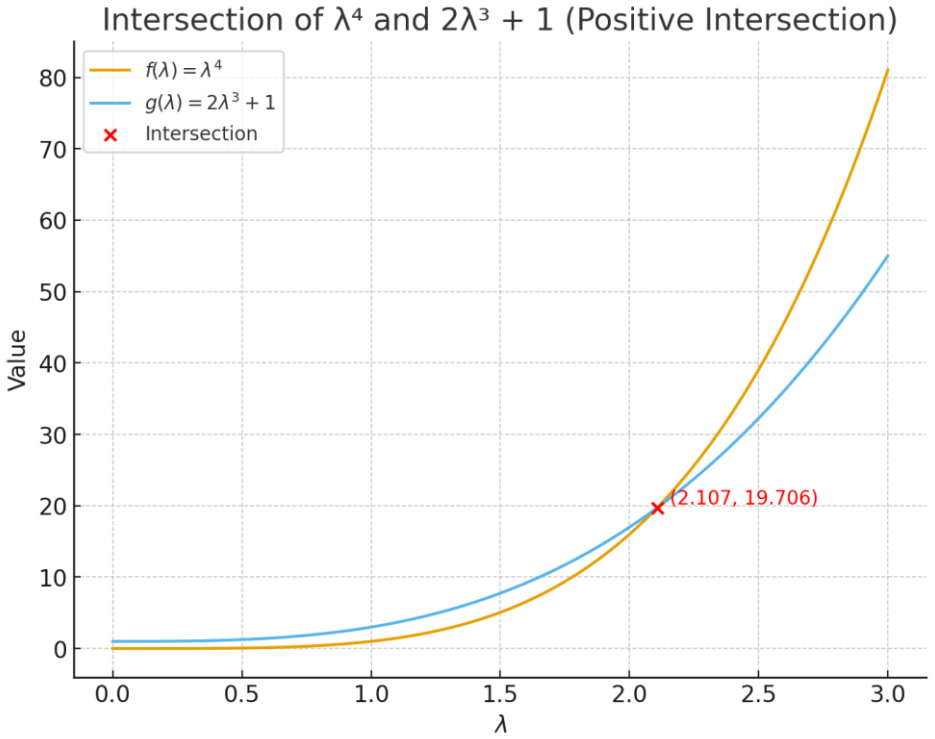
mit $d \coloneqq \max(d_1, d_2, \dots, d_\ell)$ ist.

Lemma 3.5
Ein charakteristisches Polynom $P(\lambda) = \lambda^d - \lambda^{d-d_1} - \dots - \lambda^{d-d_\ell}$ mit $d \coloneqq \max(d_1, d_2, \dots, d_\ell)$ besitzt genau eine positive (reelle) Nullstelle λ_0.

Schranken für λ_0

Lemma 3.5
Ein charakteristisches Polynom $P(\lambda) = \lambda^d - \lambda^{d-d_1} - \cdots - \lambda^{d-d_\ell}$ mit $d \coloneqq \max(d_1, d_2, \ldots, d_\ell)$ besitzt genau eine positive (reelle) Nullstelle λ_0.

Lemma 3.6
Sei $P(\lambda) = \lambda^d - \sum_{i=1}^{d} a_i \lambda^{d-i}$ mit $d \in \mathbb{N}$, $\forall i \in \{1, 2, \ldots, d\}$: $a_i \in \mathbb{N}$. Dann ist die Nullstelle $\lambda_0 \in [1, \max a_i + 1]$



Kapitel 4 – Iterative Compression

$(\star)$-Kompressionsproblem

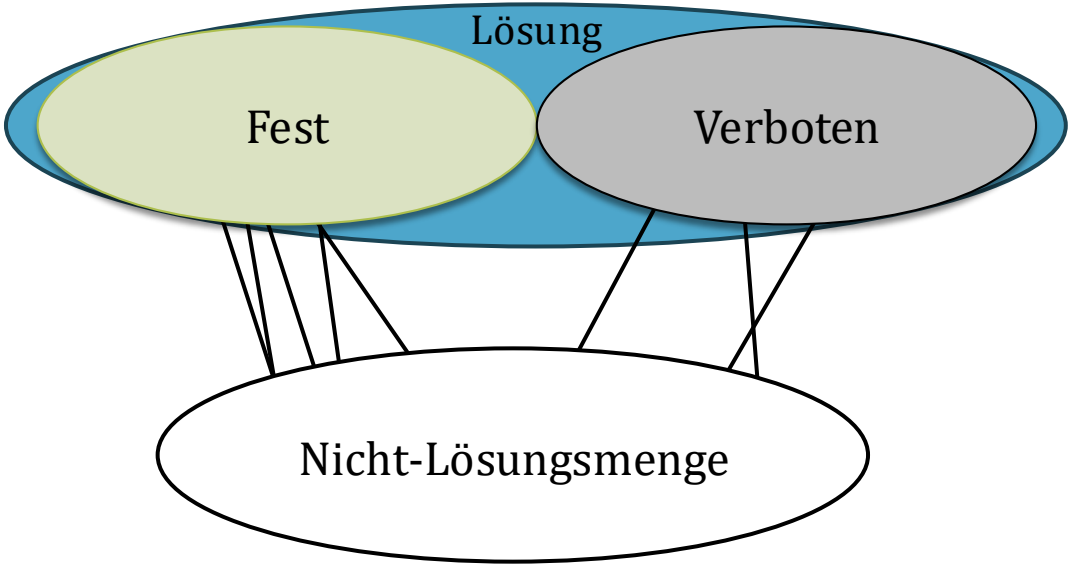
Problem 4.8
Gegeben: Instanz (I, k) eines Problems $(\star)$, $k \in \mathbb{N}$ und eine Lösung L_I der Größe $k + 1$.
Gesucht: Eine Lösung L'_I der Größe k oder die Ausgabe, dass keine Lösung existiert.

Korollar 4.9:
Existiert für ein Problem $(\star)$ ein Algorithmus für Problem 4.8 in Zeit $O(f(k)n^c)$ für eine Konstante c, dann kann $(\star)$ in Zeit $O(f(k)n^{c+1})$ gelöst werden.

Algorithmus-Idee

Für eine gegebene Lösung:

- Halte einen Teil fest.
  → Fixer Teil der Lösung
- Prüfe, ob der restliche Teil der Lösung durch eine bessere Lösung der Nicht-Lösungsmenge ersetzt werden kann.
- Probiere alle Möglichkeiten für den fixen Teil.



$(\star)$-Disjoint-Problem

Problem 4.10
Gegeben: Instanz $(I = W \dot{\cup} X, k)$ eines Problems $(\star)$, mit W ist eine Lösung zu I.
Gesucht: Eine Lösung $L \subseteq X$ der Größe k oder die Ausgabe, das keine Lösung existiert.

Korollar 4.11:
Existiert für ein Problem $(\star)$ ein Algorithmus für Problem 4.10 in Zeit $O\left(g(k)n^{O(1)}\right)$, dann existiert ein Algorithmus für Problem 4.8 mit Laufzeit

$$O\left(\sum_{i=0}^{k}\binom{k+1}{i}g(k-i)n^{c+1}\right)$$

gelöst werden.

Speziell für $g(k) = \alpha^k$ für ein α, dann besitzt der $(\star)$-Kompressionsalgorithmus Laufzeit $O\left((1+\alpha)^k n^{O(1)}\right)$

Kapitel 5 – Baumweite

Baumweite

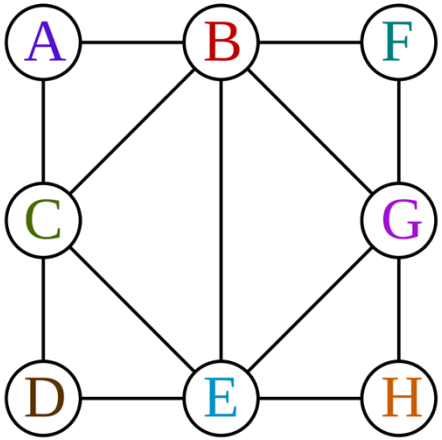
Definition 5.1:
Eine **Baumzerlegung** (Tree Decomposition) eines Graphens G ist ein Tupel $\mathcal{T} = \left(T, \{X_t\}_{t \in V(T)}\right)$, wobei T ein Baum ist mit Knoten t, welche Teilmengen $X_t \subseteq V(G)$ ("Bags") entsprechen. Dabei gelten 3 Punkte:

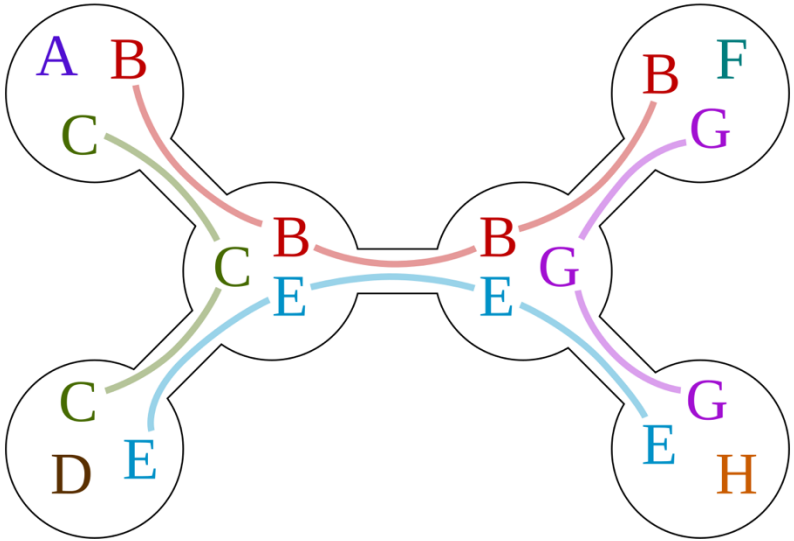
1. $\bigcup_{t \in V(T)} X_t = V(G)$
2. $\forall \{u, v\} \in E(G): \exists t \in V(T)$ mit $u, v \in X_t$
3. $\forall u \in V(G): T[\{ t \in V(T) \mid e \in X_t \}]$ ist zusammenhängend

Die **Weite** (width) einer Baumzerlegung $\mathcal{T}$ ist $\text{width}(\mathcal{T}) := \max_{t \in V(T)} (|X_t| - 1)$

Die **Baumweite** (tree width) eines Graphen G ist $\text{tw}(G) := \min_{\text{Baumzerlegung } \mathcal{T} \text{ von } G} \left(\text{width}(\mathcal{T})\right)$

Beispiel



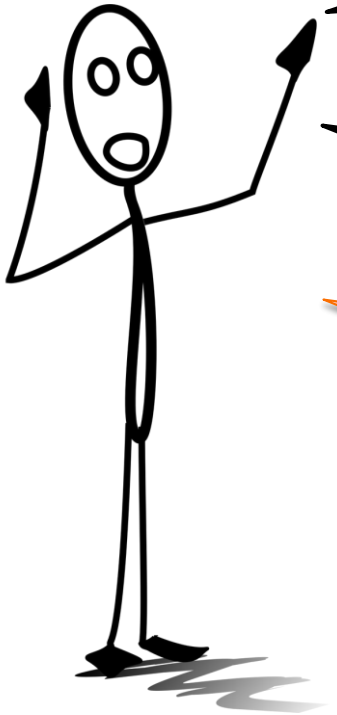


Minor

Definition 5.2:
Seien H, G Graphen. H ist ein **Minor** von G, wenn dieser durch wiederholtes Durchführen der folgenden Operationen aus G entsteht:

1. Lösche einen Knoten (samt inzidenter Kanten)
2. Lösche eine Kante
3. Kontrahiere eine Kante

Lemma 5.3:
Seien G ein Graph und H ein Minor von G. Dann gilt $\mathrm{tw}(H) \leq \mathrm{tw}(G)$



Zu jeder Baumzerlegung existiert eine schöne Baumzerlegung.

Nutze diese Zerlegung zusammen mit Dynamic Programming, um ein Problem zu lösen.

Probleme werden oft einfach auf Graphen mit beschränkter Baumweite!

Kapitel 6 – Intractability

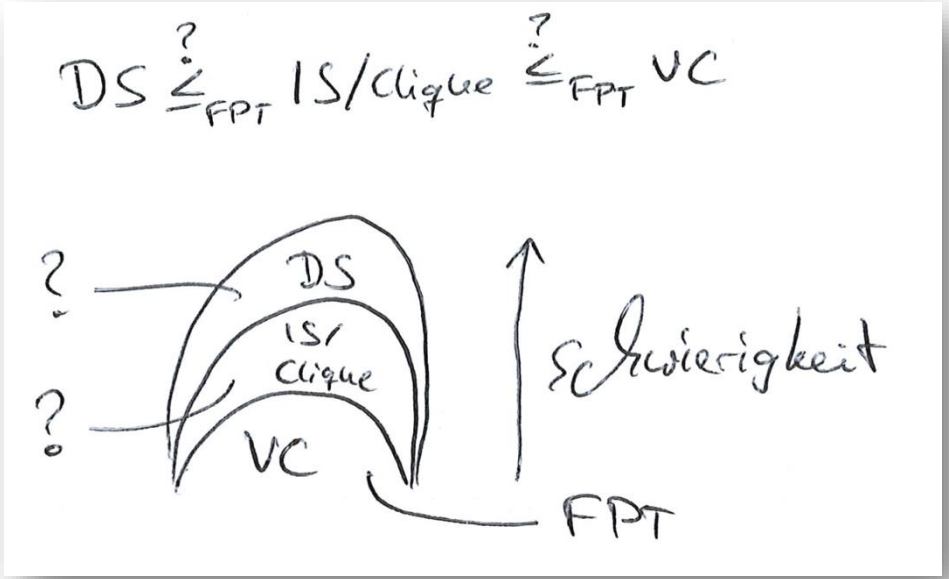
FPT-Reduktion

Definition 6.1:
Seien $A, B \subseteq \Sigma^* \times \mathbb{N}$ parametrisierte Probleme. Eine parametrisierte Reduktion von A auf B ($A \leq_{FPT} B$) ist ein Algorithmus, der jede Instanz $(x, k) \in A$ auf eine Instanz $(x', k') \in B$ abbildet, sodass

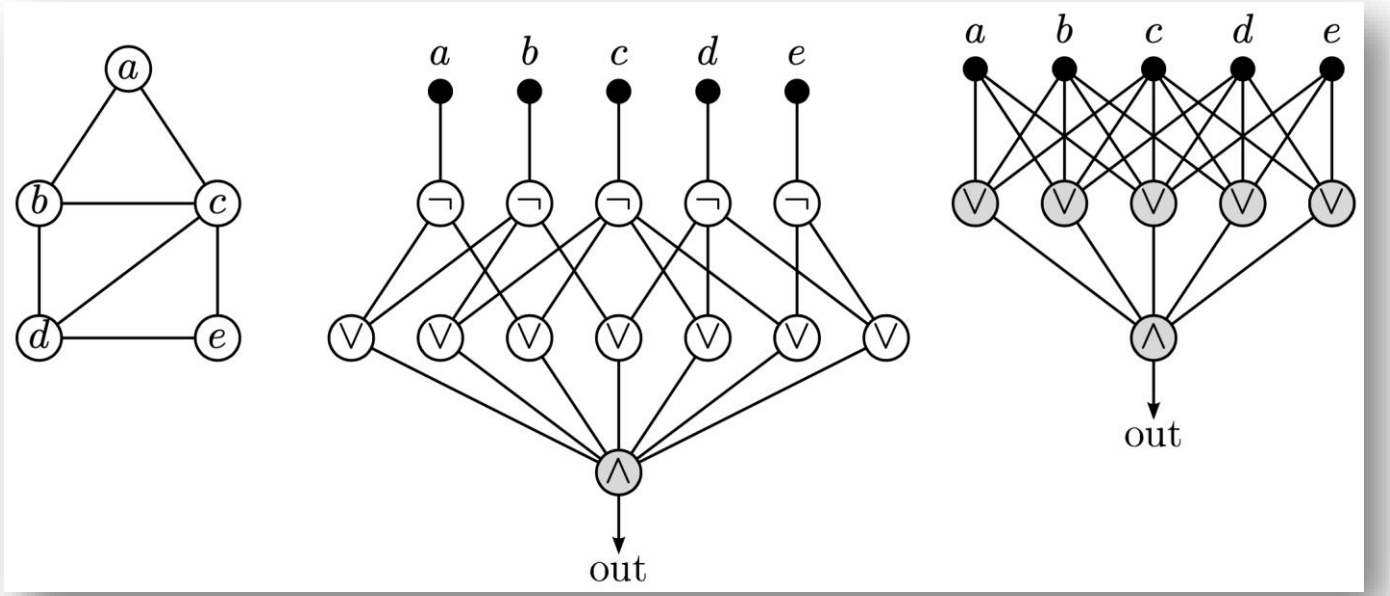
1. (x, k) ist eine Ja-Instanz, gdw. (x', k') ist eine Ja-Instanz
2. $k' \leq g(k)$ für eine berechenbare Funktion g.
3. Laufzeit ist in $O\left(f(k)n^{O(1)}\right)$ für eine berechenbare Funktion f.

Theorem 6.2:
Gilt $A \leq_{FPT} B$ und $B \in FPT$, dann ist auch $A \in FPT$.

Eine Hierarchie von Problemen?



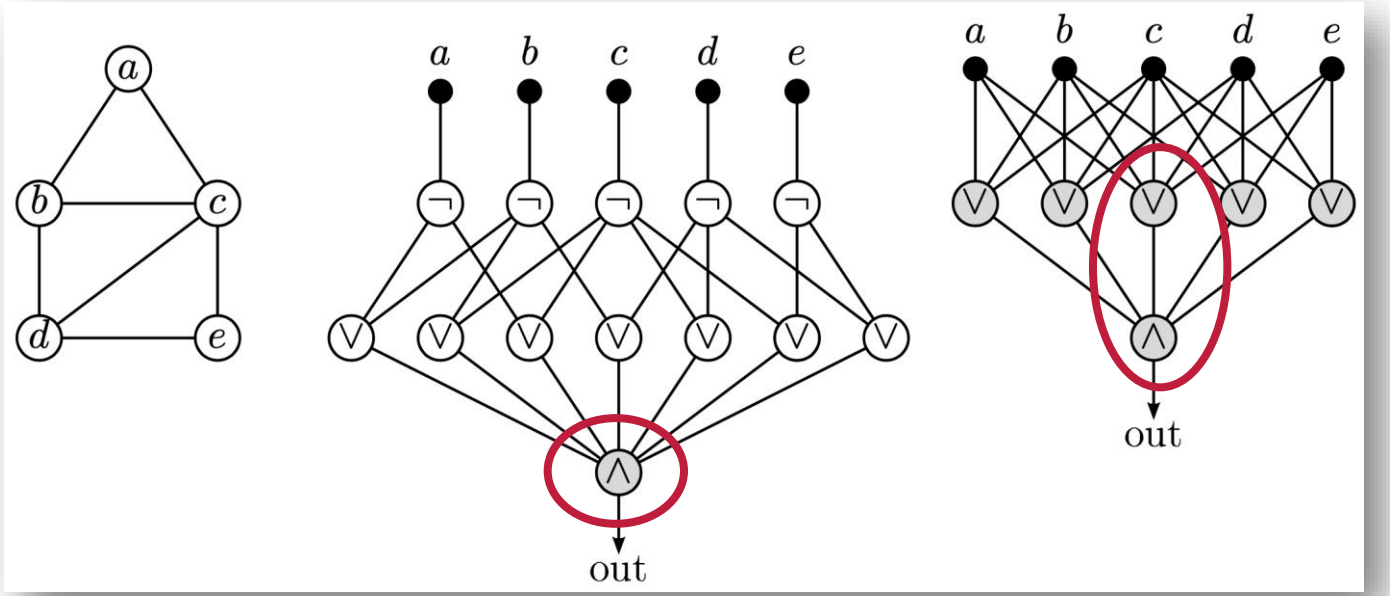
Darstellung als Boolescher Schaltkreis



Independent Set

Dominating Set

Darstellung als Boolescher Schaltkreis



1 Unbeschränkter Knoten

2 Unbeschränkte Knoten

Weft und W-Hierarchie

Definition 6.8:
Sei C ein boolescher Schaltkreis. Die **weft** von C ist die maximale Anzahl an großen Knoten (in-degree > 2) auf einem Pfad von Input-Knoten zum Output-Knoten.

Schaltkreise mit weft $\leq t$ und Tiefe $\leq d$ bezeichnen wir mit $C_{t,d}$.

Definition 6.9:
Für $t \geq 1$ gehört ein parametrisiertes Problem P zur Klasse $W[t]$, falls eine parametrisierte Reduktion von P zu $WCS[C_{t,d}]$ für eine Konstante $d \geq 1$ existiert.

Probleme und Klassen

Problem	Vertex Cover	Vertex Cover	Ind. Set	Ind. Set	Clique	Dom. Set	Weighted t-normalized Satisfiability
Parameter	k	Baum-weite	k	Baum-weite	k	k	k
Klasse	FPT	FPT	W[1]	FPT	W[1]	W[2]	W[t]

Das Ende!