Technische Universität Braunschweig

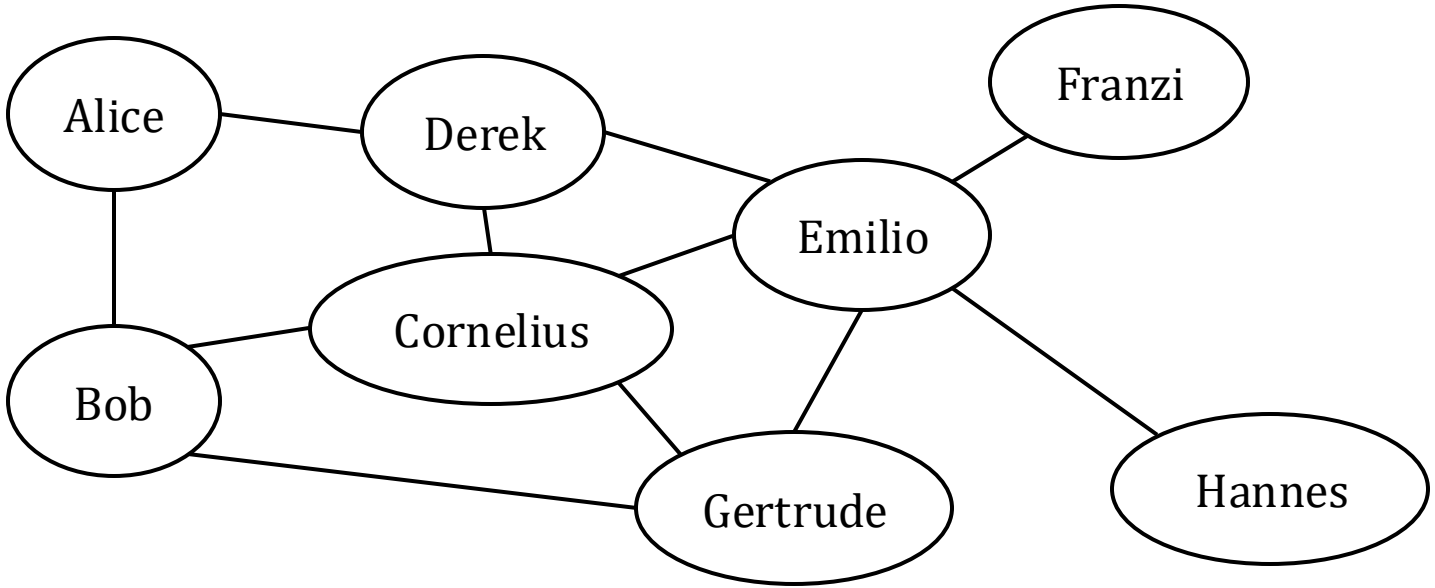
Parametrisierte Algorithmen

Arne Schmidt

Recap – Vertex Cover

Ein Graph mit Menschen

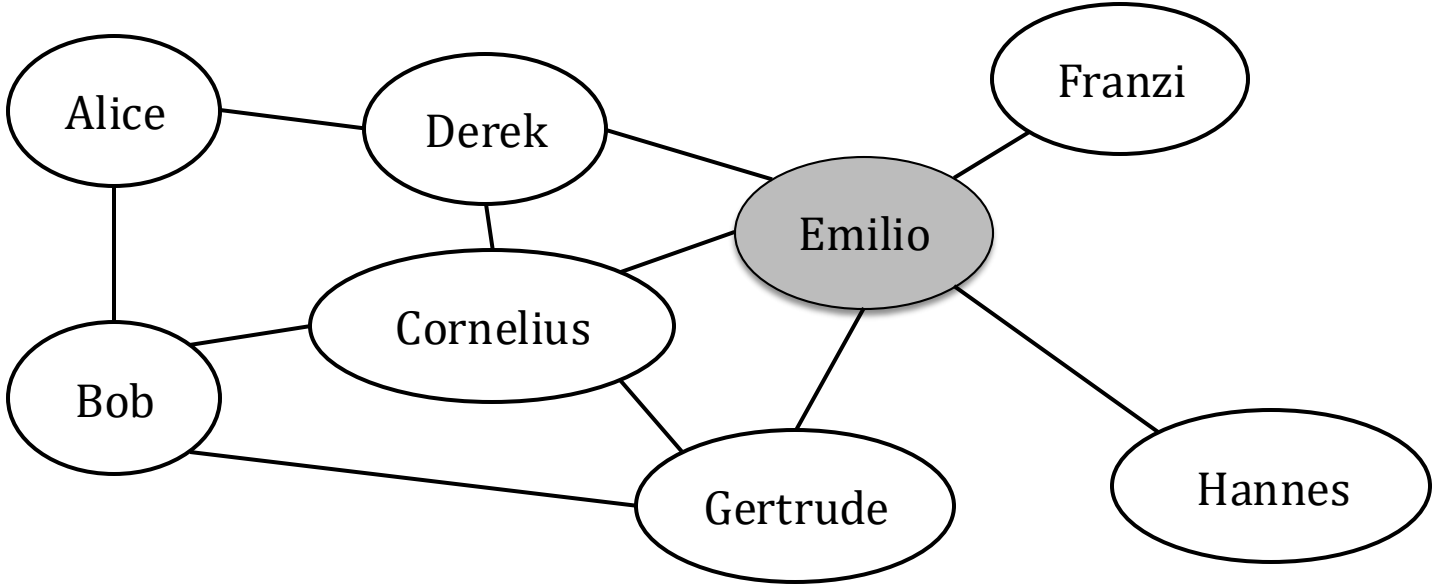
Genügt es, 4 Personen zu entfernen, um alle Konflikte aufzulösen?



Ansatz 3: Finden eines Kerns

Idee: Betrachte Knoten von hohem Grad.
Hier: Emilio **muss** im Vertex Cover sein.
Claim: Ist $\delta(v) > k$ für einen Knoten $v \in V$, dann muss v im Vertex Cover sein.

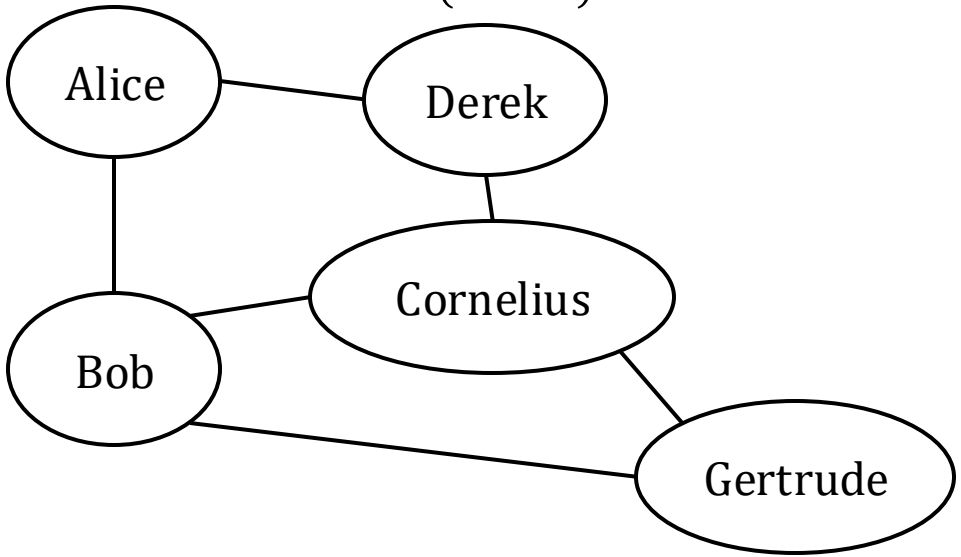
k = 4



Ansatz 3: Finden eines Kerns

Claim: Ist $\delta(v) > k$ für einen Knoten $v \in V$, dann muss v im Vertex Cover sein.
Durch wiederholtes Ausführen entsteht ein Graph, in welchem jeder Knoten Grad höchstens k besitzt. → Maximal $O(k^2)$ Knoten und Kanten.
→ Bounded Tree Search liefert in $O(2^k \cdot k^2)$ eine Antwort.

k = 3



Ansatz 3: Finden eines Kerns

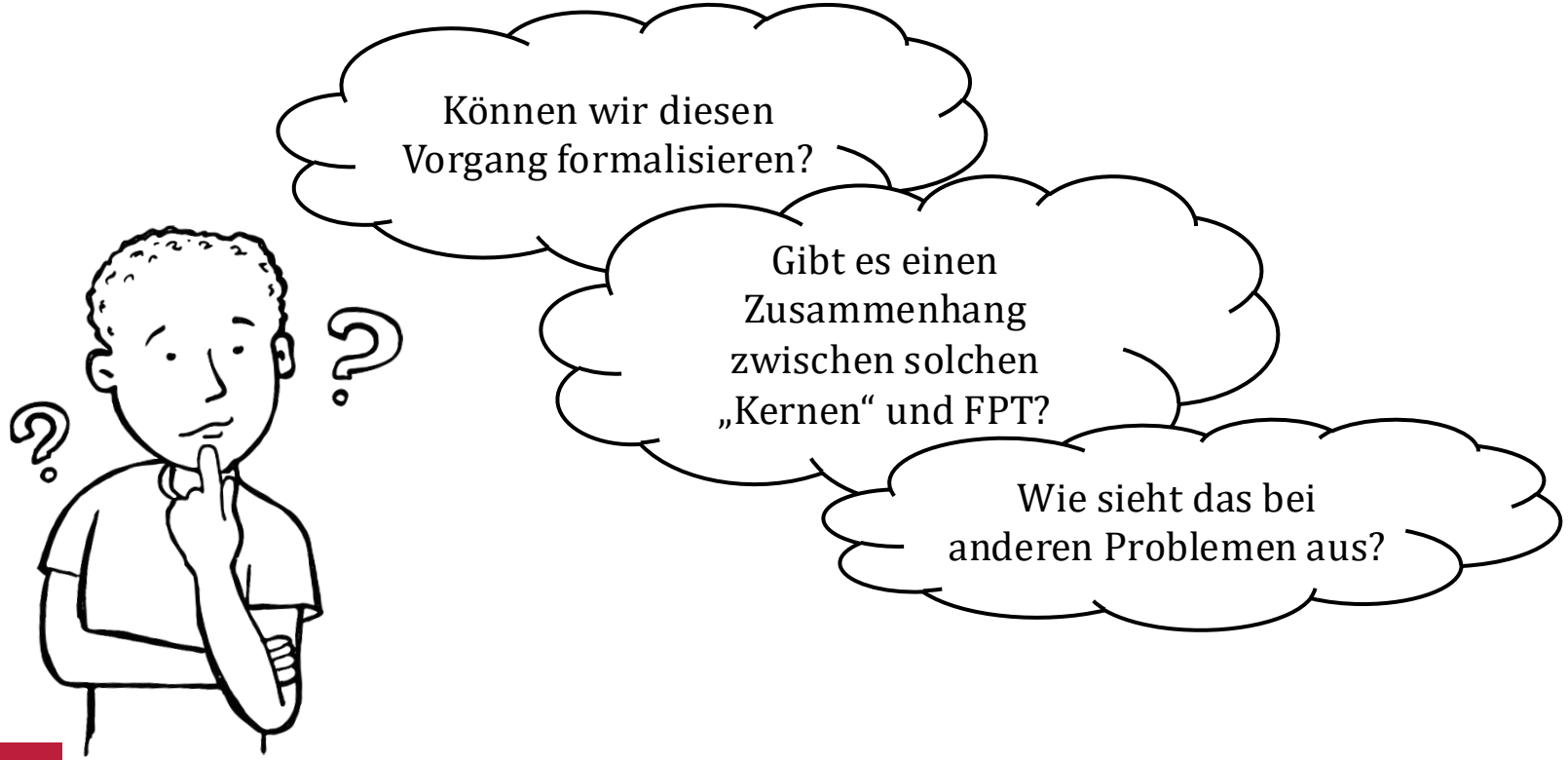
Lemma 1.4:
Das Bar-Fight-Problem lässt sich in Zeit $O\left(2^k \cdot k^2 + n + m\right)$ lösen.

Idee:
1. Reduziere den Graphen mit der angegebenen Regel, bis sie nicht mehr anwendbar ist.
2. Führe auf dem Rest Ansatz 2 (Bounded Tree Search) aus.

Schritt 1:
Kann in Linearzeit erfolgen. Liefert einen Graphen mit $n', m' \in O(k^2)$ vielen Knoten und Kanten.

Schritt 2:
Rekursionstiefe ist maximal k, Verzweigungsgrad der Rekursion ist 2.
→ $O\left(2^k\right)$ Knoten im Suchbaum.
Den jeweils neuen Graphen zu berechnen dauert maximal $O(n' + m') = O(k^2)$ Zeit.

Fragen


Können wir diesen Vorgang formalisieren?
Gibt es einen Zusammenhang zwischen solchen „Kernen“ und FPT?
Wie sieht das bei anderen Problemen aus?

Kapitel 2 – Kernelisation

Reduktionsregeln

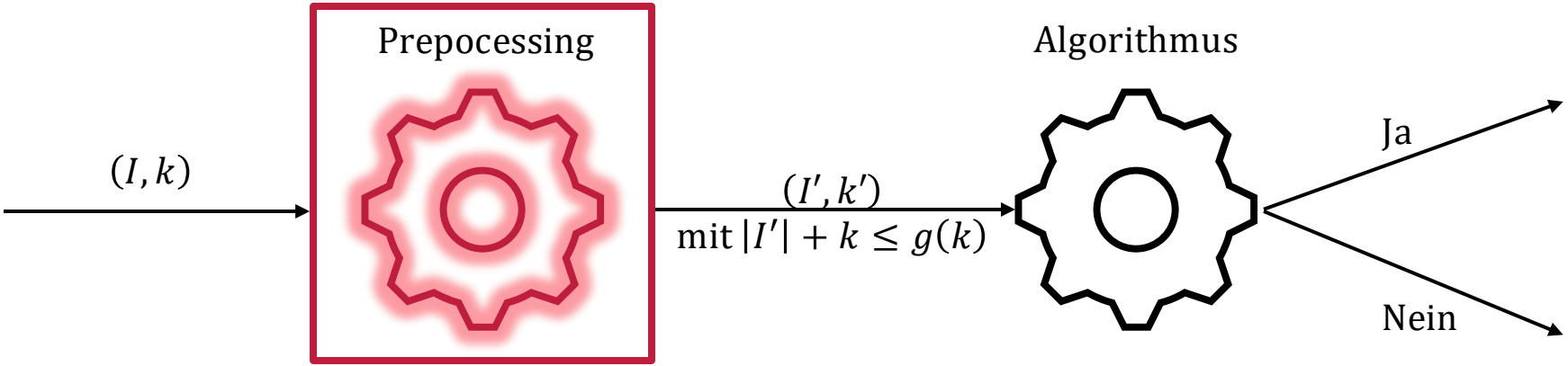
Definition 2.1
Eine **Reduktionsregel** (reduction rule) für ein parametrisiertes Problem Q ist eine Funktion $\phi: \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, welche eine Instanz (I, k) von Q auf eine äquivalente Instanz (I', k') von Q abbildet.

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

Bemerkungen:
Äquivalenz heißt, beide Instanzen sind entweder ja- oder nein-Instanzen.

Eine Reduktionsregel wird auch "sicher" (engl. safe) genannt, wenn die Äquivalenz gilt.

Parametrisierter Algorithmus mit Kernelisation



Polynomielle Zeit

i.d.R Anwenden von Reduktionsregeln

Kernel

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.

Bemerkungen:
Ist $g(\cdot)$ eine polynomielle (lineare) Funktion, dann sagen wir, dass Q einen polynomiellen (linearen) Kernel erlaubt.
Oft werden beides, der Prepocessingalgorithmus und dessen Ausgabe, Kernel genannt.

FPT und Kernel

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

Beweis:
An der Tafel!

Bemerkung:
Kernels, die man über die "⇒"-Richtung erhält, haben in der Regel exponentielle Größe.
Man ist eher an polynomiellen oder linearen Kernels interessiert!

Kapitel 2.2 – Beispiele für Kernel

Kapitel 2.2.1 – Vertex Cover

Regeln zur Reduzierung für Vertex Cover

Regel VC1:
Enthält G einen isolierten Knoten v (Grad 0), entferne v. Die neue Instanz ist $(G - v, k)$.

Regel VC2:
Enthält G einen Knoten v mit Grad $> k$, entferne v. Die neue Instanz ist $(G - v, k - 1)$.

Lemma 2.4
Ist (G, k) eine Ja-Instanz und keine der Regeln (VC1 oder VC2) lässt sich anwenden, dann gilt:
$$|V(G)| \leq k^2 + k \text{ und } |E(G)| \leq k^2$$

Beweis: Tafel!

Regeln zur Reduzierung für Vertex Cover

Regel VC1:
Enthält G einen isolierten Knoten v (Grad 0), entferne v. Die neue Instanz ist $(G - v, k)$.

Regel VC2:
Enthält G einen Knoten v mit Grad $> k$, entferne v. Die neue Instanz ist $(G - v, k - 1)$.

Regel VC3:
Sei (G, k) eine Instanz, bei welcher Regeln VC1 und VC2 nicht anwendbar ist. Gib "Nein" aus, falls eine der folgenden Bedingungen erfüllt ist.

a) $k < 0$,
b) $|V(G)| > k^2 + k$
c) $|E(G)| > k^2$

Vertex Cover Kernel

Lemma 2.4
Vertex Cover (parametrisiert mit k) besitzt einen Kernel mit $O(k^2)$ Knoten und $O(k^2)$ Kanten.

Beweisskizze:
Zu zeigen sind folgende Punkte.

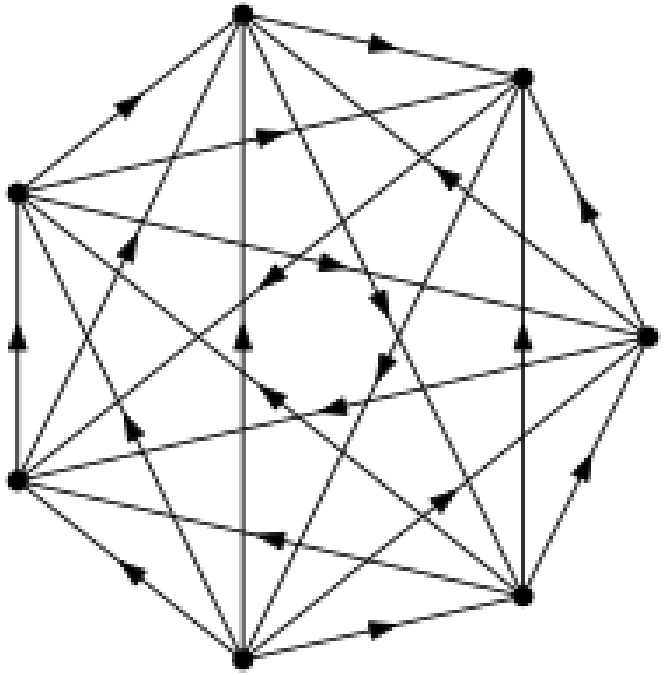
- Jede Ausgangsinstanz kann so transformiert werden, sodass $O(k^2)$ Knoten und $O(k^2)$ Kanten übrig bleiben. (Bereits bewiesen)
- Die Transformation benötigt polynomielle Laufzeit. (selbst!)

Kapitel 2.2.2 – Feedback Arc Set in Tournament Graphen

Tournament Graphen

Definition 2.5
Ein gerichteter Graph $D = (V, A)$ heißt **Tournament Graph**, wenn für je zwei Knoten $u, v \in V$ entweder $(u, v) \in A$ oder $(v, u) \in A$ (aber nicht beide).

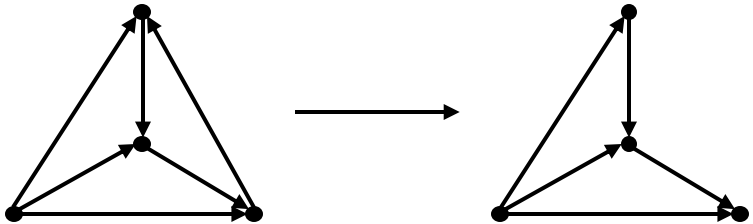
Lemma 2.6
Existiert in einem Tournament Graphen ein gerichteter Kreis, dann existiert ein gerichteter Kreis mit drei Knoten ("Dreieck").



Feedback Arc Set in Tournament Graphen

Problem 2.7 (FAST)
Gegeben: Tournament Graph $D = (V, A)$, Zahl $k \in \mathbb{N}$.
Frage: Exsitiert eine Kantenmenge $F \subseteq A$, sodass $D' = (V, A \setminus F)$ azyklisch ist?

Zwischenfrage:
Gilt FAST $\in$ XP?



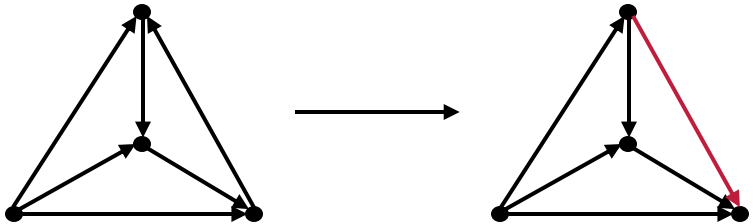
$k = 1$

Lemma 2.8
Sei $D = (V, A)$ ein Tournament Graph und $F \subseteq A$. Dann ist F genau dann ein inklusions-minimales Feedback Arc Set, wenn F eine inklusions-minimale Menge ist, sodass D durch Umdrehen der Kanten in F azyklisch wird.

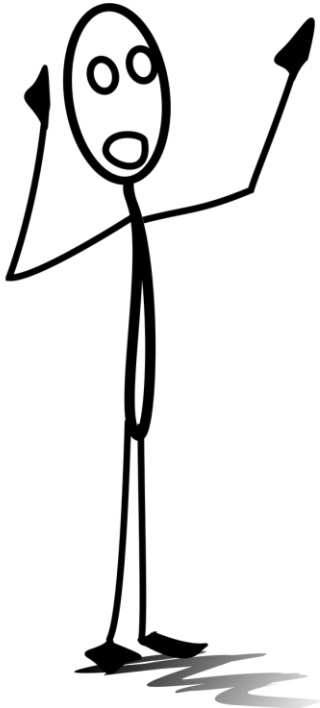
Feedback Arc Set in Tournament Graphen

Problem 2.7 (FAST)
Gegeben: Tournament Graph $D = (V, A)$, Zahl $k \in \mathbb{N}$.
Frage: Exsitiert eine Kantenmenge $F \subseteq A$, sodass $D' = (V, A \setminus F)$ azyklisch ist?

Lemma 2.8
Sei $D = (V, A)$ ein Tournament Graph und $F \subseteq A$. Dann ist F genau dann ein inklusions-minimales Feedback Arc Set, wenn F eine inklusions-minimale Menge ist, sodass D durch Umdrehen der Kanten in F azyklisch wird.



$k = 1$



Konzentriere dich also auf folgende 2 Punkte:

Drehe maximal k Kanten um.

Betrachte als Kreise nur Dreiecke.