Programmierpraktikum
Algorithm Engineering (FPT-Algorithmen)

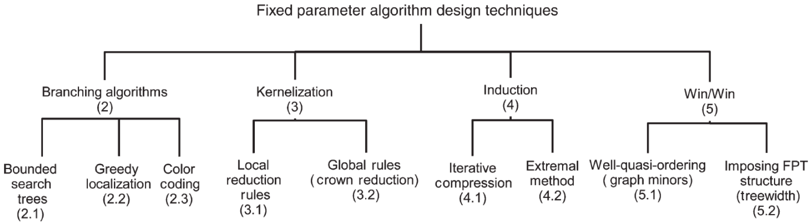
Einführung

Martin Holzer Ignaz Rutter

31. Oktober 2008

Ansätze für NP-schwere Probleme

- Approximative Lösungen
- Average-Case- statt Worst-Case-Analyse
- Randomisierung
- Quantencomputer
- Empirische Untersuchung von Heuristiken auf Benchmarks
- **Entwurf von FPT-Algorithmen**

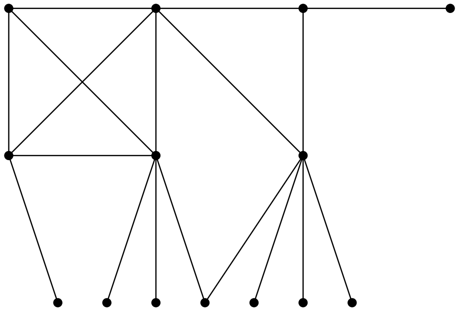

Fixed parameter algorithm design techniques
Branching algorithms (2)
Kernelization (3)
Induction (4)
Win/Win (5)
Bounded search trees (2.1)
Greedy localization (2.2)
Color coding (2.3)
Local reduction rules (3.1)
Global rules (crown reduction) (3.2)
Iterative compression (4.1)
Extremal method (4.2)
Well-quasi-ordering (graph minors) (5.1)
Imposing FPT structure (treewidth) (5.2)

Beispiel: Vertex Cover

Vertex-Cover

Gegeben Graph $G = (V, E)$ mit n Knoten und m Kanten.

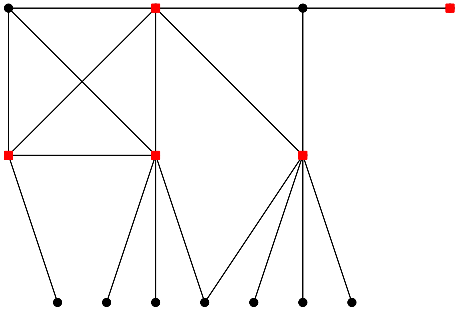
$C \subseteq V$ heißt *Vertex-Cover* von G, falls $C \cap \{u, v\} \neq \emptyset \quad \forall uv \in E$.



Vertex-Cover

Gegeben Graph $G = (V, E)$ mit n Knoten und m Kanten.

$C \subseteq V$ heißt *Vertex-Cover* von G, falls $C \cap \{u, v\} \neq \emptyset \quad \forall uv \in E$.



k-Vertex-Cover

Definition (k-Vertex-Cover)

Gegeben: Graph $G = (V, E)$
Parameter: k
Gesucht: Vertex-Cover der Größe $\leq k$ (k-VC)

- NP-vollständig [Karp '72].
- Eines der "6 basic NP-complete problems" [GJ '79].

Naiver Algorithmus: $O(n^k)$

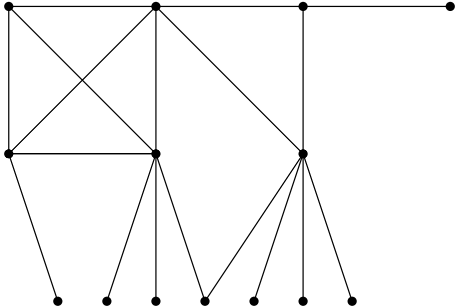
k-VC leichter gemacht

Beobachtung

1. *Sei $v \in V$.*
   VC muß entweder v oder v's Nachbarschaft $N(v)$ enthalten.
2. *Gegeben (G, k).*
   Knoten mit Grad $> k$ sind in jedem k-VC von G enthalten.
3. *Falls $\Delta(G) \leq k$ und $|E| > k^2$, so hat G kein k-VC.*
4. *Fall $\Delta(G) \leq k$ und $|V| > k^2 + k$, so hat G kein k-VC.*

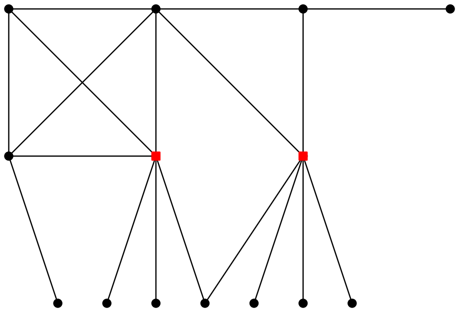
Beispiel Reduktion

parametrisierte Instanz $(G, 5)$



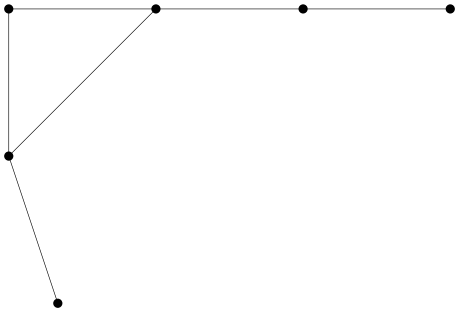
Beispiel Reduktion

parametrisierte Instanz $(G, 5) \rightsquigarrow (G', 3)$



Beispiel Reduktion

parametrisierte Instanz $(G, 5) \rightsquigarrow (G', 3)$



Algorithmus nach Buss

Kernbildung:

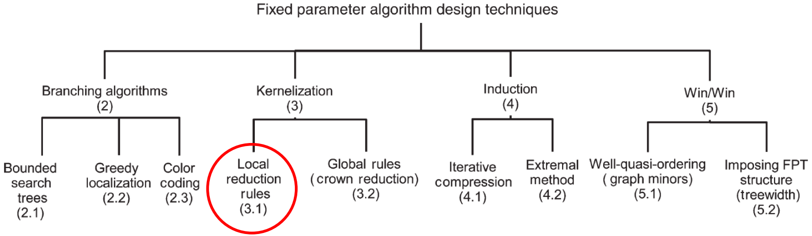
Sei $H = \{v \in V : \deg(v) > k\}$
Falls $|H| > k$: NEIN — STOP!
Sei $G' = (V', E') = G|_{V-H}$ und $k' = k - |H|$
Falls $|E'| > k^2$: NEIN — STOP!

$\rightsquigarrow$

Vollständige Exploration:

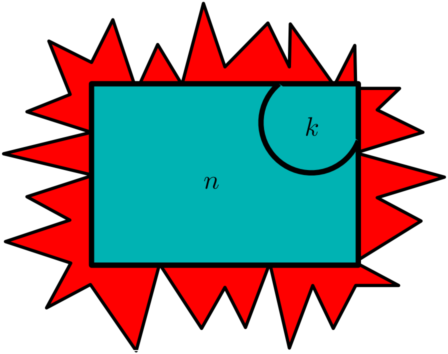
Gehe durch alle k'-elementigen Teilmengen C von V':
    Falls C Abdeckung von G': $H \cup C$ — STOP!
NEIN

$\rightsquigarrow$ Kerngröße $\mathcal{O}(k^2)$, Laufzeit: $\mathcal{O}(kn + 2^k k^{2k+2})$.

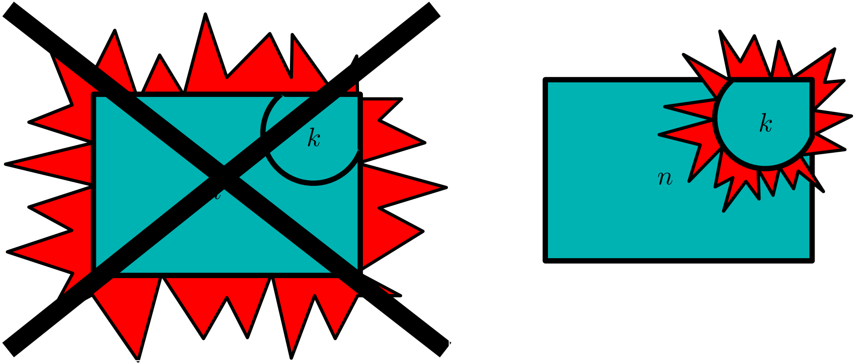

Fixed parameter algorithm design techniques
Branching algorithms (2)
Kernelization (3)
Induction (4)
Win/Win (5)
Bounded search trees (2.1)
Greedy localization (2.2)
Color coding (2.3)
Local reduction rules (3.1)
Global rules (crown reduction) (3.2)
Iterative compression (4.1)
Extremal method (4.2)
Well-quasi-ordering (graph minors) (5.1)
Imposing FPT structure (treewidth) (5.2)

FPT – Die Theorie

Intuition



Intuition



Überblick

- **Ziel**: Einschränkung der Laufzeitexplosion: $\mathcal{O}(f(k) \cdot n^{\mathcal{O}(1)})$
- **Kunst**: Identifikation von 'kleinem' Parameter k
- $\oplus$: Optimalität, beweisbare Laufzeitschranken
- $\ominus$: exponentielle Laufzeit
- **Hoffnung/Herausforderung**:
  $\longrightarrow$ kleiner Kern in Polynomialzeit
  $\longrightarrow$ 'erträgliches' $f(k)$

Definitionen

Parametrisiertes Problem

$L \subseteq \Sigma^* \times \Sigma^*$ (meist $\subseteq \Sigma^* \times \mathbb{N}$)

Definitionen

Parametrisiertes Problem

$L \subseteq \Sigma^* \times \Sigma^*$ (meist $\subseteq \Sigma^* \times \mathbb{N}$)

Fixed-parameter tractable

$L \in \mathcal{FPT}$, falls in $f(k) \cdot |x|^{O(1)}$ bestimmbar, ob $(x, k) \in L$, wo f berechenbare Funktion nur in k

Definitionen

Parametrisiertes Problem

$L \subseteq \Sigma^* \times \Sigma^*$ (meist $\subseteq \Sigma^* \times \mathbb{N}$)

Fixed-parameter tractable

$L \in \mathcal{FPT}$, falls in $f(k) \cdot |x|^{O(1)}$ bestimmbar, ob $(x, k) \in L$, wo f berechenbare Funktion nur in k

Reduktion auf Problemkern

$(x, k) \mapsto (x', k')$ mit

- $k' \leq k$, $|x'| \leq g(k)$
- $(x, k) \in L$ gdw. $(x', k') \in L$
- Reduktion in Polynomialzeit

SAT

- Klausellänge: 2SAT $\in \mathcal{P}$ vs. 3SAT $\in \mathcal{NPC}$
- #Variablen: $\mathcal{O}(2^n)$
- #Klauseln: $\mathcal{O}(1.24^m)$
- Formellänge: $\mathcal{O}(1.08^\ell)$

Kemeny Score: Beispiel

Abstimmungsergebnis:

- #1: $A > B > C$
- #2: $A > C > B$
- #3: $B > C > A$

Kompromissreihenfolge:

- $A > B > C$

Abstände:

#	A–B	A–C	B–C	$\sum$
1	0	0	0	0
2	0	0	1	1
3	1	1	0	2
KS				**3**

Kemeny Score: Komplexität

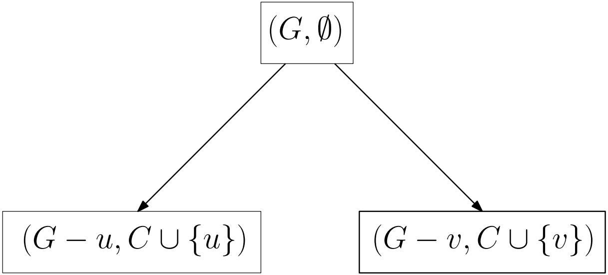
Kemeny Score

- **Gegeben:** Kandidatenmenge, Abstimmungsergebnis, Parameter k
- **Gefragt:** Ist Kemeny-Score höchstens k?

Par.	Beschreibung	Laufzeit
k	Kemeny-Score	$\mathcal{O}^*(1.53^k)$
d	maximale paarweise Distanz	$\mathcal{O}^*((3d+1)!)$
m	#Kandidaten	$\mathcal{O}^*(2^m)$
n	#Wähler	$\mathcal{NP}$-vollständig für $n = 4$

Beschränkte Suchbäume

Beschränkter Suchbaum



- konstruiere Baum mit **Wurzel** $(G, \emptyset)$
- **Zerlegung entlang Kante** $\{u, v\}$
- **falls 'kantenloser' Knoten** (G', C') auf Level $\leq k$, dann gib C' aus
- **Binärbaum der Tiefe** $k \Rightarrow 2^k$ Knoten.
- Laufzeit $\mathcal{O}(n \cdot 2^k)$

Kombination

Einzeltechniken:

- Kernbildung: $\mathcal{O}(k \cdot n + k^2 2^{2k+2})$
- Beschränkter Suchbaum: $\mathcal{O}(n \cdot 2^k)$

Kombination:

1. Kernbildung: $\mathcal{O}(k \cdot n)$
2. Beschränkter Suchbaum auf Kern-Instanz: $\mathcal{O}(k^2 \cdot 2^k)$

Gesamtlaufzeit: $\mathcal{O}(k \cdot n + k^2 \cdot 2^k)$.

Historie – k-Vertex Cover

Urheber	Jahr	Laufzeit	Technik
Fellows/Langston	1986	$f(k) \cdot n^3$	graph minor theorem
Johnson	1987	$f(k) \cdot n^2$	Baumzerlegung, DP
Fellows	1988	$2^k \cdot n$	beschr. Suchbaum
Papadimitriou/ Yannakakis	1993	$m + 3^k \cdot n$	maximales Matching, Suchbaum
Buss	1989	$kn+2^k \cdot k^{2k+2}$	Kernbildung
Balasubramanian, Fellows, Raman	1992	$kn + 2^k \cdot k^2$	Kombination
Niedermeier/ Rossmanith	1999	$kn+1.29^k \cdot k^2$	

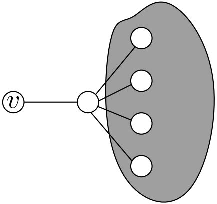
Verbesserungen

$$
\begin{tikzcd}
 & \boxed{(G, k)} \arrow[dl, "C \cup \{v\}"'] \arrow[dr, "C \cup N(v)"] & \\
\boxed{(G - v, k - 1)} & & \boxed{(G - N(v), k - \deg(v))}
\end{tikzcd}
$$

- Verwende Knoten mit **möglichst großem Grad**
- Jeder Knoten mind. Grad 2: $\mathcal{O}(n \cdot 1{,}63^k)$
- Jeder Knoten mind. Grad 3: $\mathcal{O}(n \cdot 1{,}47^k)$
- Jeder Knoten mind. Grad 4: $\mathcal{O}(n \cdot 1{,}31^k)$

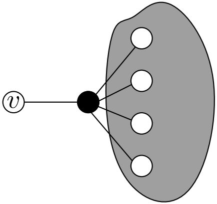
Entfernen von Knoten mit kleinem Grad

- Grad-1-Knoten v: $G' = G - (\{v\} \cup N(v))$, $\quad C = C' \cup N(v)$



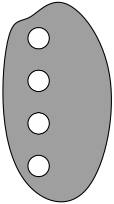
Entfernen von Knoten mit kleinem Grad

- Grad-1-Knoten v: $G' = G - (\{v\} \cup N(v))$, $\quad C = C' \cup N(v)$



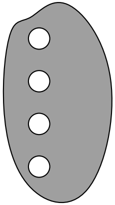
Entfernen von Knoten mit kleinem Grad

- Grad-1-Knoten v: $G' = G - (\{v\} \cup N(v))$, $\quad C = C' \cup N(v)$

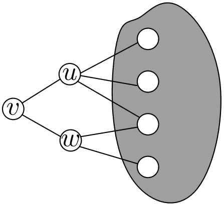


Entfernen von Knoten mit kleinem Grad

- Grad-1-Knoten v: $G' = G - (\{v\} \cup N(v)), \quad C = C' \cup N(v)$

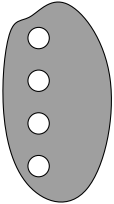


- Grad-2-Knoten

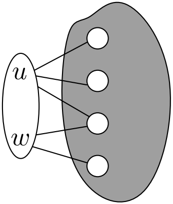


Entfernen von Knoten mit kleinem Grad

- Grad-1-Knoten v: $G' = G - (\{v\} \cup N(v))$, $\quad C = C' \cup N(v)$

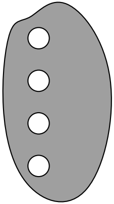


- Grad-2-Knoten

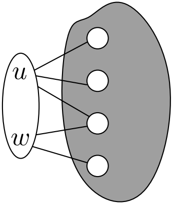


Entfernen von Knoten mit kleinem Grad

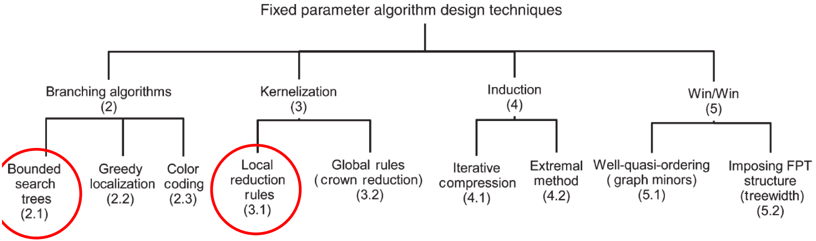
- Grad-1-Knoten v: $G' = G - (\{v\} \cup N(v))$, $\quad C = C' \cup N(v)$



- Grad-2-Knoten v: $uw \in C' \Rightarrow u, w \in C$, sonst $v \in C$



Ausblick


Fixed parameter algorithm design techniques
Branching algorithms (2)
Kernelization (3)
Induction (4)
Win/Win (5)
Bounded search trees (2.1)
Greedy localization (2.2)
Color coding (2.3)
Local reduction rules (3.1)
Global rules (crown reduction) (3.2)
Iterative compression (4.1)
Extremal method (4.2)
Well-quasi-ordering (graph minors) (5.1)
Imposing FPT structure (treewidth) (5.2)

Iterative Kompression

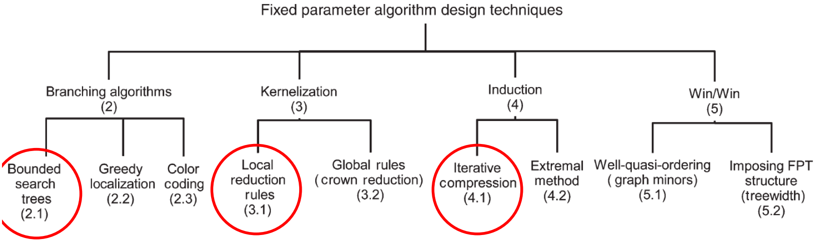
Theorem (o. B.)

Gegeben Graph G und Knotenabdeckung C' mit $|C'| \leq k+1$. In Zeit $\mathcal{O}(2^k \cdot |G|)$ kann optimale Abdeckung von G bestimmt werden.

Betrachte alle nicht-leeren Teilmengen $X \subseteq C'$:

$C'' := (C' \setminus X) \cup N(X)$

falls $|C''| \leq C'$, dann $C' := C''$


Fixed parameter algorithm design techniques
Branching algorithms (2)
Kernelization (3)
Induction (4)
Win/Win (5)
Bounded search trees (2.1)
Greedy localization (2.2)
Color coding (2.3)
Local reduction rules (3.1)
Global rules (crown reduction) (3.2)
Iterative compression (4.1)
Extremal method (4.2)
Well-quasi-ordering (graph minors) (5.1)
Imposing FPT structure (treewidth) (5.2)

Komplexität

- **Analogon** zu $\mathcal{P} \subseteq \mathcal{NP}$:
  $\mathcal{FPT} \subseteq W[1] \subseteq W[2] \subseteq \ldots$
- **Vollständigkeit** bzgl. $W[\cdot]$:
  $\rightarrow$ Independent Set ist $W[1]$-vollständig
  $\rightarrow$ Dominating Set ist $W[2]$-vollständig
- **Reduktionen** $(\ldots)$