Näherungsalgorithmen

(Approximationsalgorithmen)

WiSe 2024/25 in Trier

Henning Fernau
Universität Trier
fernau@uni-trier.de

24. Oktober 2024

Näherungsalgorithmen Gesamtübersicht

- Organisatorisches
- Einführung / Motivation
- Grundtechniken für Näherungsalgorithmen
- Approximationsklassen (Approximationstheorie)

Organisatorisches

Vorlesung Besprechung bzw. Übungen (nach Absprache)
Termin: donnerstags 10-12 Uhr im H13

Modulprüfungen werden bei uns immer als mündliche Prüfungen abgelegt

Meine Sprechstunde: DO, 13-14 Uhr im F213
Sprechstunde Kevin Mann: DI, 13-14 Uhr im H407

Kontakt: fernau@uni-trier.de, mann@uni-trier.de

Ein allgemeines Überdeckungsproblem

g ist gegeben durch ein Tripel (X, f, w), wobei gilt:

- X ist eine endliche Menge;
- $f : 2^X \to \{0, 1\}$ ist eine *monotone Abbildung*, d.h. $A \subseteq B \to f(A) \leq f(B)$, und es gelte $f(X) = 1$; (Eine Menge C mit $f(C) = 1$ heiße *Überdeckung*.)
- $w : X \to \mathbb{R}^+$ ist die *Gewichtsfunktion*, erweitert zu $w(A) = \sum_{x \in A} w(x)$.

Gesucht: Überdeckung C^* mit kleinstmöglichem Gewicht $OPT(w) = w(C^*)$.

Beobachtungen: Sind $f, g : 2^X \to \{0, 1\}$ monoton, so auch ihr „Maximum“ $h : 2^X \to \{0, 1\}$ mit $h(A) = \max\{f(A), g(A)\}$. Sind $u, v : X \to \mathbb{R}^+$ Gewichtsfunktionen, so auch ihre „Summe“ $u + v := w : X \to \mathbb{R}^+$ mit $w(x) = u(x) + v(x)$.

Gewichtetes Knotenüberdeckungsproblem

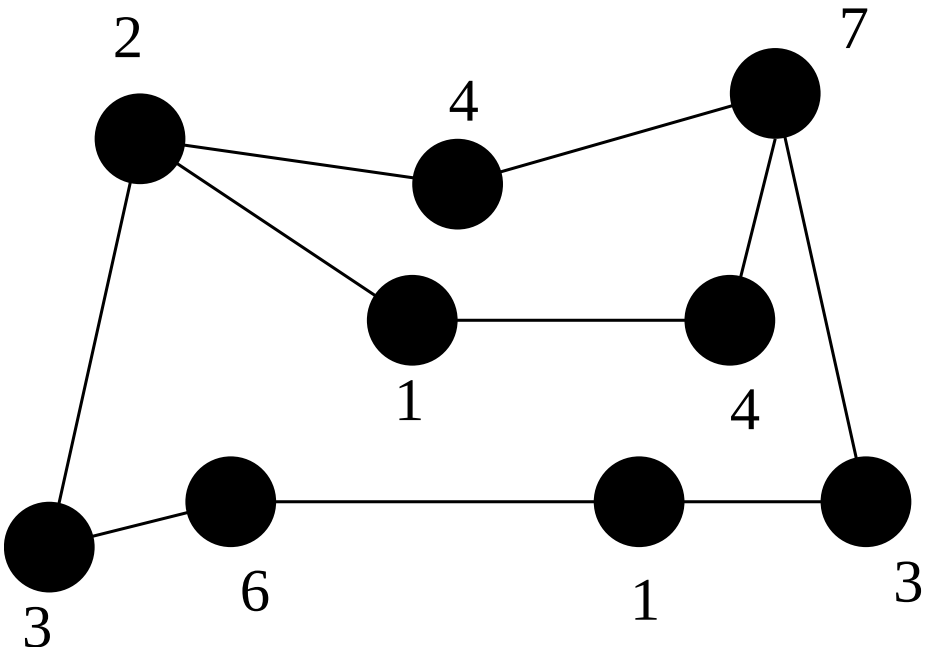
Ggb.: gewichteter Graph, i.Z.: $G = (V, E), w : V \to \mathbb{R}^+$
In obiger Terminologie:

$$X = V, \quad f : V' \mapsto \begin{cases} 0, V' \text{ ist keine Knotenüberdeckung} \\ 1, V' \text{ ist eine Knotenüberdeckung} \end{cases}$$

Bemerke: f ist hier implizit durch E gegeben und muss **nicht** explizit gespeichert werden bzw. gehört nicht explizit zur Eingabe.

Literatur: R. Bar-Yehuda: One for the price of two: a unified approach for approximating covering problems. *Algorithmica*, **27**, 131–144, 2000.

Ein kleines Beispiel:



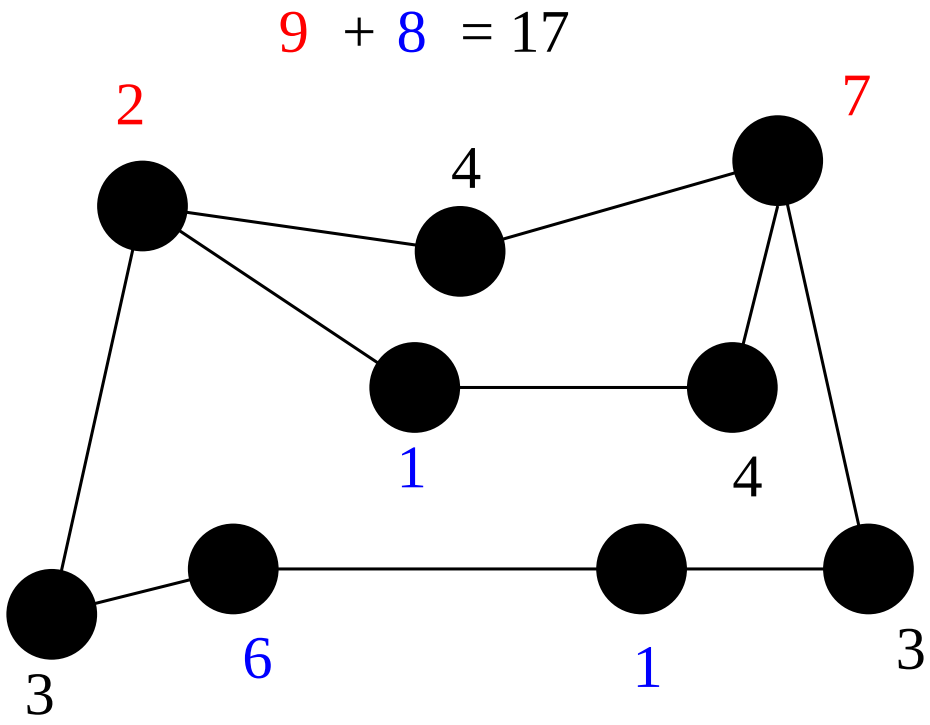
Gewichtsfunktion w definiert durch Zahlen an Knoten.

Gewichtsfunktionen werden auch gerne mengenwertig aufgefasst.

Ein kleines Beispiel: mit möglicher Lösung

rote Knoten: Große-Grad-Heuristik

blaue Knoten aus Heuristik: Grad=1 ? $\rightsquigarrow$ Nimm Nachbar !



Ein hilfreiches Lemma für Überdeckungsprobleme

Zerlegungsbeobachtung für ein allgemeines ÜP:

Sind w_1, w_2 Gewichtsfunktionen, so gilt für die Gewichte $OPT(w_i)$ der jeweiligen kleinstmöglichen Überdeckungen:

$$OPT(w_1) + OPT(w_2) \leq OPT(w_1 + w_2)$$

Beweis: Ist C^* optimal für $w_1 + w_2$, so gilt:

$$OPT(w_1 + w_2) = (w_1 + w_2)(C^*) = w_1(C^*) + w_2(C^*) \geq OPT(w_1) + OPT(w_2)\,. \quad \square$$

Anwendung des Lemmas

Betrachte nun als *Gewichtsreduktionsfunktion* $\delta : X \to \mathbb{R}^+$
mit $\forall x \in X : 0 \leq \delta(x) \leq w(x)$, d.h. δ und $w - \delta$ sind Gewichtsfunktionen.

Setze $\Delta OPT := OPT(w) - OPT(w - \delta)$.
Zerlegungsbeobachtung $\rightsquigarrow$

$$\begin{aligned} \Delta OPT &= OPT(w) - OPT(w - \delta) \\ &\geq OPT(\delta) + OPT(w - \delta) - OPT(w - \delta) = OPT(\delta) \end{aligned}$$

Eine Gewichtsreduktion δ führt daher zu einer Minimumsreduktion um wenigstens $OPT(\delta)$.

Wir nennen δ r-*effektiv*, falls $\delta(X) \leq r \cdot OPT(\delta)$.

Hinweis: Der nächste Beweis funktioniert auch, falls $\delta(X) \leq r \cdot \delta(C^*)$ gilt, C^* optimal für w.

Grundnäherungsalgorithmus für Überdeckungsprobleme

$A(X, f, w)$:

- Wähle r-effektive Gewichtsreduktion δ .
- Berechne durch $B(X, f, w - \delta)$ eine Überdeckung C .
- Gib C aus.

Der erwähnte Algorithmus B wird häufig A selbst wieder sein (oder eine leichte Modifikation).

Approximationen mit konstantem Faktor

Ein Verfahren A heißt *(Faktor) r-Approximation* für ein Minimierungsproblem, falls A eine Lösung C liefert, die nur um einen Faktor von höchstens r schlechter ist als das Minimum C^*.

Satz über lokale Verhältnisse für den Grundnäherungsalgorithmus:
Ist B eine r-Approximation, dann ist A ebenfalls eine r-Approximation.

Beweis:

$$\begin{aligned} w(C) &= (w-\delta)(C)+\delta(C) \text{ [Linearität]} \\ &\leq (w-\delta)(C)+\delta(X) \text{ [Monotonie von } \delta\text{]} \\ &\leq r\cdot OPT(w-\delta)+r\cdot OPT(\delta) \text{ [}B\text{'s Eigenschaft und } \delta \text{ } r\text{-effektiv]} \\ &\leq r\cdot OPT(w) \text{ [Zerlegungsbeobachtung]} \end{aligned}$$

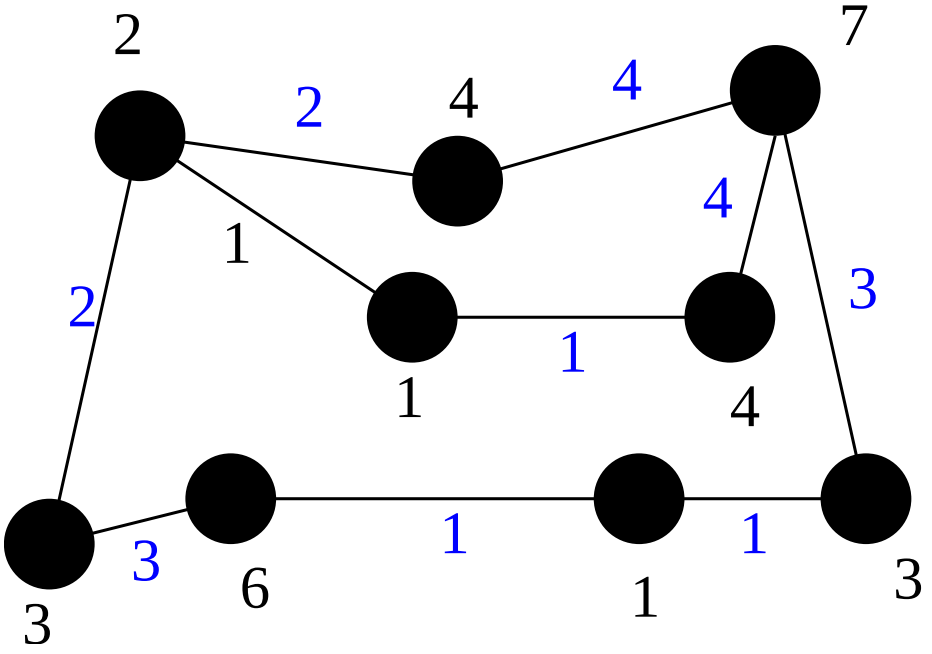
Der Algorithmus von Bar-Yehuda und Even für gewichtetes VC:
Kante e definiert Gewichtsreduktionsfunktion δ_e durch

$$\delta_e(v) = \begin{cases} \min\{w(u) \mid u \in e\} & , \quad v \in e \\ 0 & , \quad v \notin e \end{cases}$$

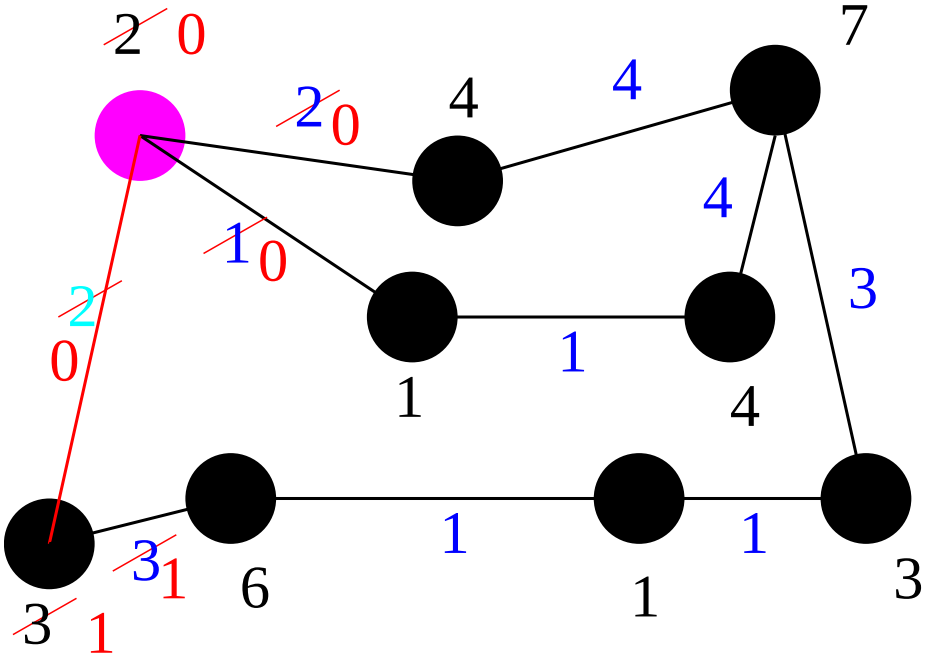
1. Rekursive Variante
BErec $(G = (V, E), w)$

- Falls $\forall e \in E : \delta_e = 0$, gib $C = \{v \in V \mid w(v) = 0\}$ aus; exit.
- Nimm irgendeine Kante e aus G (mit $\delta_e \neq 0$);
- Berechne BErec $(G, w - \delta_e)$

Unser kleines Beispiel mit Gewichtsreduktionsfunktion δ_e:
In blau sind die Werte der zu e gehörenden (auf e) konstanten Funktion eingetragen.

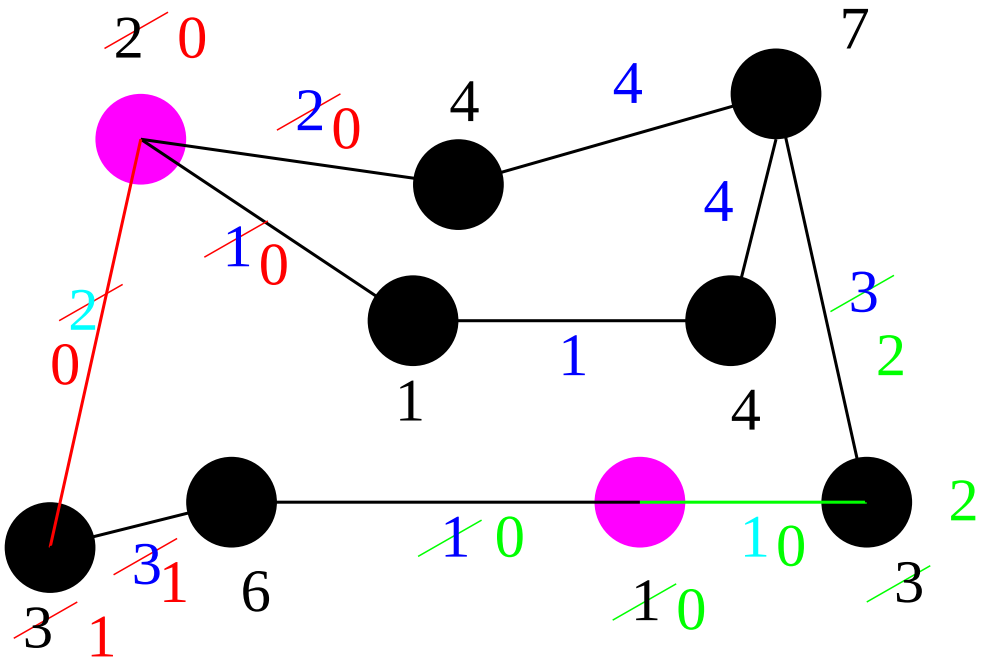


Unser kleines Beispiel mit Gewichtsreduktionsfunktion δ_e:
Wir wählen die Kante links, reduzieren dann die Gewichte der inzidenten beiden Knoten und hernach die Werte der adjazenten Kanten; in violett erscheinen die Überdeckungs-Knoten



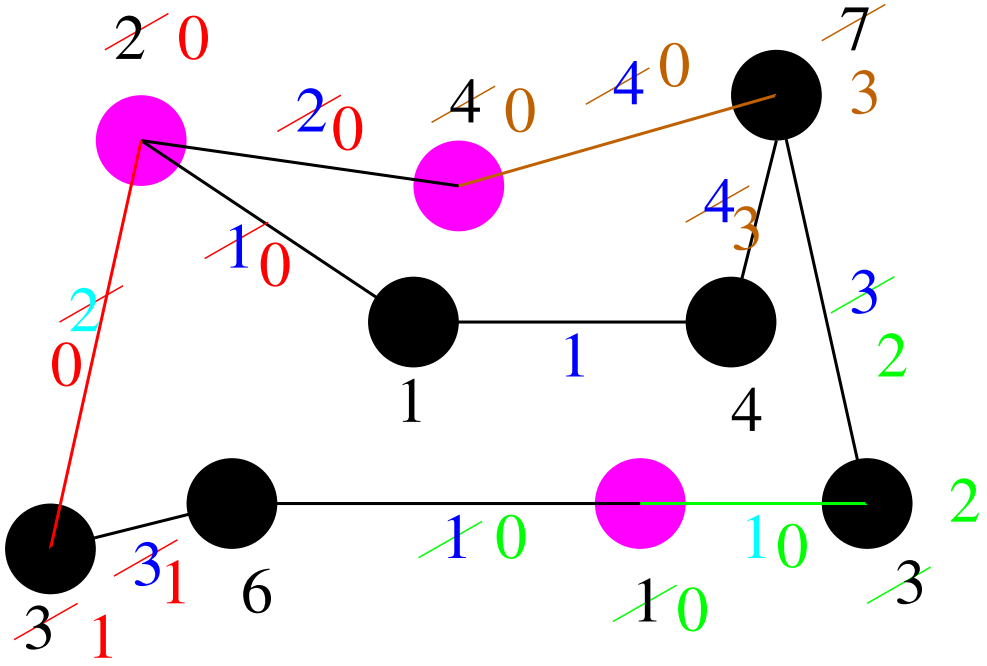
Unser kleines Beispiel mit Gewichtsreduktionsfunktion δ_e:

Wähle die grüne Kante rechts unten

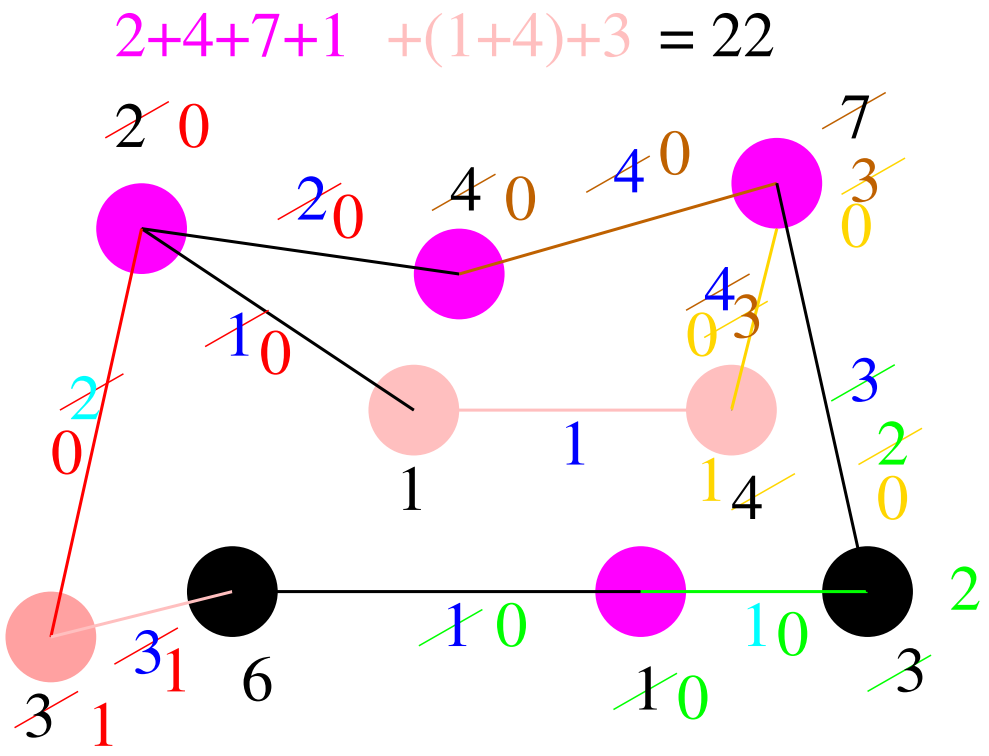


Unser kleines Beispiel mit Gewichtsreduktionsfunktion δ_e:

Wähle die braune Kante rechts oben

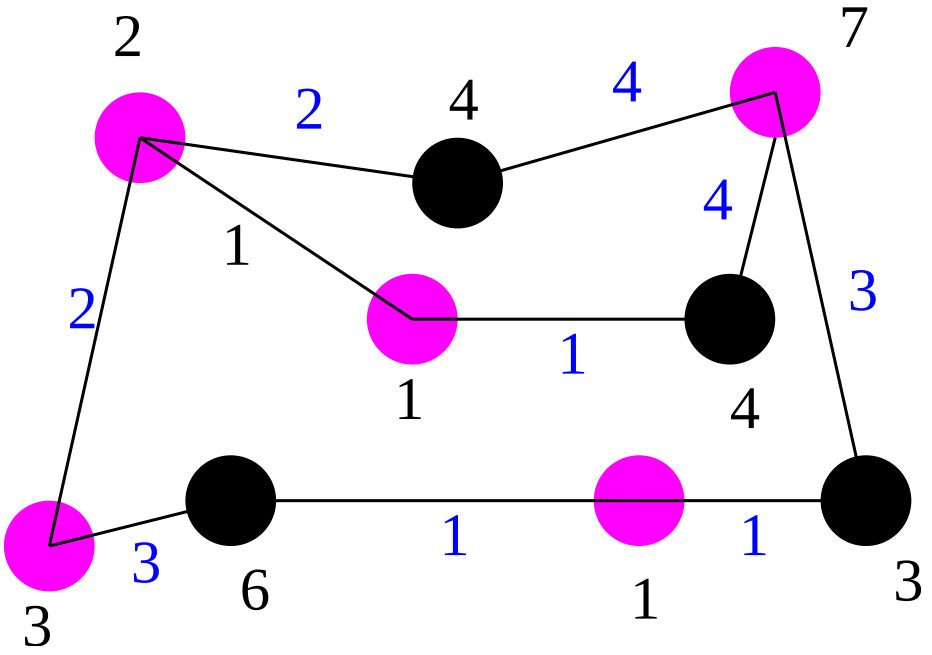


Unser kleines Beispiel mit Gewichtsreduktionsfunktion δ_e und Lösung:
erst gelb, dann die pinken Kanten



Unser kleines Beispiel;

die gewonnene Lösung enthält eine inklusions-minimale mit Gewicht 14:



Der Algorithmus von Bar-Yehuda und Even für gewichtetes VC:

2. Iterative Variante

BEiter $G = (V, E), w)$

- Für jedes $e \in E$
  - Bestimme $\varepsilon = \min\{w(v) \mid v \in e\}$
  - Für jedes $v \in e$
    setze $w(v) = w(v) - \varepsilon$
- Gib $C = \{c \in V \mid w(v) = 0\}$ aus.

Satz: Der Algorithmus von Bar-Yehuda und Even ist eine 2-Approximation.

Beweis: für BErec durch Induktion über die Rekursionstiefe t:

- Ist $t = 0$, so liefert BErec sogar eine optimale Lösung.
- Ist die Behauptung, BErec liefere 2-Approximationen, für $t = 0, .., T$ gezeigt, so liefert der Satz über die lokalen Verhältnisse den Induktionsschritt, denn δ_e ist 2-effektiv.
  Ist nämlich $C^*_{\delta_e}$ eine optimale Überdeckung für δ_e, so wird insbesondere e abgedeckt, d.h.

$$\delta_e(C^*_{\delta_e}) = \min\{w(v_1), w(v_2)\};$$

  andererseits ist natürlich

$$\delta_e(V) = \delta_e(v_1) + \delta_e(v_2) = 2 \cdot \min\{w(v_1), w(v_2)\},$$

  also $\delta_e(V) \leq 2 \cdot OPT(\delta_e)$.

Frage: (Warum) liefert die iterative Variante dasselbe ?

Wie gut ist die gefundene Lösung „wirklich“?

Wichtiges Problem beim *Benchmarking* (Leistungsvergleich) heuristischer Algorithmen für NP-schwere Probleme.

Hierbei sind auch Approximationen hilfreich, nämlich durch „möglichst ungeschickte Wahl.“ Das haben wir schon im Beispiel gesehen!

Wir könnten eine Lösung vom Gewicht 22 erhalten haben.

$\rightsquigarrow$ Eine kleinstmögliche Überdeckung C^* hat Gewicht $w(C^*) \geq 11$.

Der Algorithmus von Clarkson: Hilfsdefinitionen

δ_e als Hilfsgröße aus dem vorigen Algorithmus BErec

$N(v)$: *Menge der Nachbarn* von Knoten v (*offene Nachbarschaft*).

$d(v)$ sei der *Grad* (engl.: degree) des Knoten v, also: $d(v) = |N(v)|$.
Setze $d'(v) = |N'(v)|$ mit $N'(v) = \{u \in N(v) \mid w(u) > 0\}$ für Graph mit Knotengewichten w:

$\varepsilon(v) = \frac{w(v)}{d(v)}, \quad \varepsilon'(v) = \frac{w(v)}{d'(v)}$ (undef. bei Division durch Null)

Gewichtsreduktionsfunktionen (falls $\varepsilon^{(\prime)}(v)$ definiert):

$$\delta_v^{(\prime)}(u) = \begin{cases} w(v) & , \quad u = v \\ \varepsilon^{(\prime)}(v) & , \quad u \in N^{(\prime)}(v) \\ 0 & , \quad \text{sonst} \end{cases}$$

Der Algorithmus von Clarkson: Rekursive Variante

Crec $(G = (V, E), w)$ Annahme $E \neq \emptyset$

- Falls $\forall e \in E,\ \delta_e = 0$, gib $C = \{c \in V \mid w(v) = 0\}$ aus; exit.
- Suche Knoten $v \in V$, der $\varepsilon'(v)$ minimiert.

  Es ex. ein v, sodass $\varepsilon'(v)$ def. ist, nach dem ersten Schritt.
- Berechne Crec$(G, w - \delta'_v)$.

Der Algorithmus von Clarkson: Iterative Variante Citer $(G, (V, E), w)$

- $C := \emptyset$
- Solange $E \neq \emptyset$, tue
  - Suche $v \in V$, der $\varepsilon(v)$ minimiert.
  - Für jeden Nachbarn $u \in V$ von v: setze $w(u) := w(u) - \varepsilon(v)$.
  - Setze $G := G - v$ und $C := C \cup \{v\}$.
- Gib C aus.

Was müssen wir zeigen ?

- Die iterative Variante liefert dasselbe wie die rekursive.
  Beachte: Bei Crec bleiben nullgewichtete Knoten unbeachtet (ε'), bei Citer werden sie bevorzugt gelöscht.

- Die rekursive Variante ist eine 2-Approximation:
  (per Induktion aus dem Satz über lokale Verhältnisse).
  Wesentlich ist dafür noch zu zeigen, dass δ_v eine 2-effektive Gewichtsfunktion ist.

Überlegen Sie sich, wie der Algorithmus von Clarkson auf unserem Beispiel arbeitet.

Lemma: δ_v ist eine 2-effektive Gewichtsfunktion.

Beweis:

a) Gewichtsfunktion: Klar für $u \notin N^{(\prime)}[v] = N^{(\prime)}(v) \cup \{v\}$ (*abgeschlossene Nachbarschaft*).
Ist $u \in N^{(\prime)}(v)$, so ist $\delta_v^{(\prime)}(u) = \frac{w(v)}{d^{(\prime)}(v)} \leq \frac{w(u)}{d^{(\prime)}(u)} \leq w(u)$ aufgrund der minimalen Wahl von v.

b) 2-effektiv:

$$\begin{aligned}
\delta_v^{(\prime)}(V) &= \delta_v^{(\prime)}(v) + \delta_v^{(\prime)}(N^{(\prime)}(v)) + \delta_v^{(\prime)}(V - N^{(\prime)}[v]) \\
&= w(v) + \left|N^{(\prime)}(v)\right| \frac{w(v)}{d^{(\prime)}(v)} + 0 \\
&= 2 \cdot w(v) = 2 \cdot OPT(\delta_v^{(\prime)})
\end{aligned}$$

Randomisierte r-Approximation

Eine Wahrscheinlichkeitsverteilung auf dem Raum der Gewichtsreduktionsfunktionen zu (X, f, w) heit r-*effektiv*, falls

$$\mathcal{E}[\delta(X)] \leq r \cdot \mathcal{E}[\delta(C^*)]^*, \quad C^* \text{ ist Opt. bzgl. } w$$

Erinnerung: Der Erwartungswert ist ein „lineares Funktional“,
d.h. insbesondere: $\mathcal{E}[A + B] = \mathcal{E}[A] + \mathcal{E}[B]$.
Damit überträgt sich fast wörtlich der (Beweis vom) Satz über lokale Verhältnisse vom deterministischen Fall.

*$\mathcal{E}$ ist Erwartungswert (bzgl. der angenommenen Verteilung);
wir nehmen die stärkere Def. von r-effektiv aus dem det. Fall

Satz über lokale Verhältnisse — Randomisierte Version

Betrachte dazu folgenden *Grundalgorithmus* (bei Verteilung F) $A(X, f, w)$

- Wähle Gewichtsreduktion $\delta : X \to \mathbb{R}^+$gemäß r-effektiver Verteilung F.
- Berechne durch $B(X, f, w - \delta)$ eine Überdeckung C.
- Gib C aus.

Falls B eine Überdeckung C liefert mit

$$\mathcal{E}[(w - \delta)(C)] \leq r \cdot \mathcal{E}[OPT(w - \delta)],$$

dann gilt: A liefert eine randomisierte r-Approximation.

Beispiel für randomisierte Approximation: RandomGreedyVC $(G = (V, E), w)$

- Setze $C := \{v \in V \mid w(v) = 0\}$ und $G := G - C$.
- Solange $E \neq \emptyset$:
  - $\overline{w} := \left(\sum_{v \in V} \frac{d(v)}{w(v)}\right)^{-1}$
  - Wähle zufällig $v \in V$ mit Wahrscheinlichkeit $p(v) = \frac{d(v)}{w(v)} \cdot \overline{w}$
  - Setze $C := C \cup \{v\}$ und $G := G - v$.
- Gib C aus.

Satz: RandomGreedyVC ist eine randomisierte 2-Approximation.

Beweis:

1. Überführe RandomGreedyVC in eine äquivalente rekursive Form.

2. Zeige die Behauptung durch vollständige Induktion über die Rekursionstiefe unter Benutzung der randomisierten Version des Satzes über lokale Verhältnisse.

3. Dazu ist zu klären: Wie sieht die Wahrscheinlichkeitsverteilung auf dem Raum der Gewichtsreduktionsfunktionen aus?
   Die Reduktionsfunktion $\delta^v(u) = w(u)\delta_{uv}$ wird mit Wahrscheinlichkeit $p(v)$ gezogen, alle übrigen Gewichtsfunktionen mit Wahrscheinlichkeit 0.
   $\sum_{v \in V} p(v) = 1$ gilt nach Definition von $p(v)$.

4. Zu zeigen bleibt: die Wahrscheinlichkeitsverteilung ist 2-effektiv.

   a) $\mathcal{E}[\delta(C^*)] = \sum_{v \in V} \delta^v(C^*) \cdot p(v) = \sum_{v \in C^*} w(v)p(v) = \sum_{v \in C^*} d(v) \cdot \overline{w} \geq |E| \cdot \overline{w}$

   b) $\mathcal{E}[\delta(V)] = \sum_{v \in V} w(v)p(v) = \sum_{v \in V} d(v)\overline{w} = 2 \cdot |E| \cdot \overline{w}$

Verallgemeinerung: **(△-)Hitting-Set**

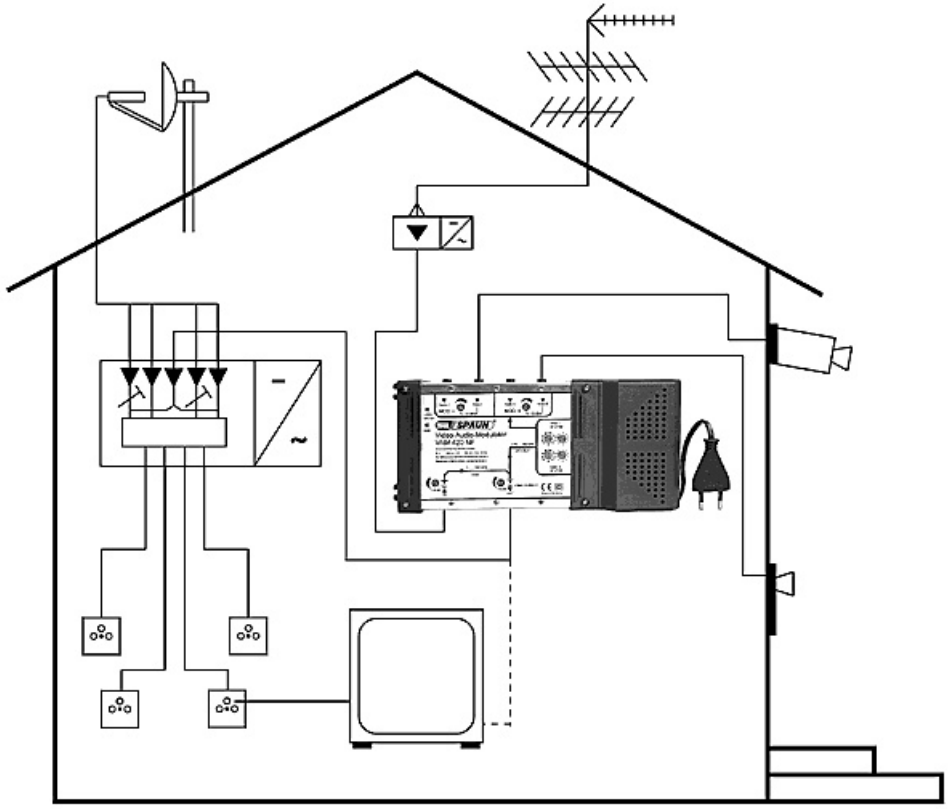
Ggb.: Grundmenge U („Universum“) von n Elementen $x_1, \ldots, x_n$ und Mengen $S_1, \ldots, S_m \subseteq U$, genannt *Hyperkanten*

Ges.: Kleinstmögliche Menge $C \subseteq U$, die alle Hyperkanten „trifft“,
d.h.: $\forall\, 1 \leq k \leq m\ C \cap S_k \neq \emptyset$ (*Überdeckungseigenschaft*).
Ist bekannt, dass $\forall 1 \leq k \leq m : |S_k| \leq \Delta$ für eine Konstante Δ, so lassen sich die Algorithmen von Bar-Yahuda, Clarkson (!) und Random-Greedy auch für *△-Hitting-Set* lesen. Insbesondere: Kante e definiert Gewichtsreduktionsfunktion δ_e durch

$$\delta_e(v) = \begin{cases} \min\{w(u) \mid u \in e\} & , \quad v \in e \\ 0 & , \quad v \notin e \end{cases}$$

Satz: △-Hitting-Set ist Faktor-△-approximierbar.

Motivation: Systemanalyse á la Reiter



Was ist ein System? (nach R. Reiter)

- Systembestandteile (Komponenten) C
- Systembeschreibung (wie? $\leadsto$ Logik) SD:
  Aussagen über erwartetes Systemverhalten,
  d.h., Beziehungen zwischen den Komponenten.
- beobachtetes Systemverhalten (Observationen) OBS

Was ist ein fehlerbehaftes System?

- spezielles Prädikat $ab(c)$ für jede Komponente $c \in C$:
  kennzeichnet abnormes Verhalten (Fehler)
  SD enthält auch Aussagen der Form:
  „Wenn $ab(c)$, dann gilt:. . . “ bzw.
  „Wenn $\neg ab(c)$, dann gilt:. . . “

- ein System (C, SD, OBS) ist *fehlerbehaftet*, wenn in

$$SD \cup OBS \cup \{\neg ab(c) \mid c \in C\}$$

  ein Widerspruch zu erkennen ist.

Konfliktmengen und Diagnosen

Eine *Konfliktmenge* ist eine Menge C' von Komponenten, so dass in

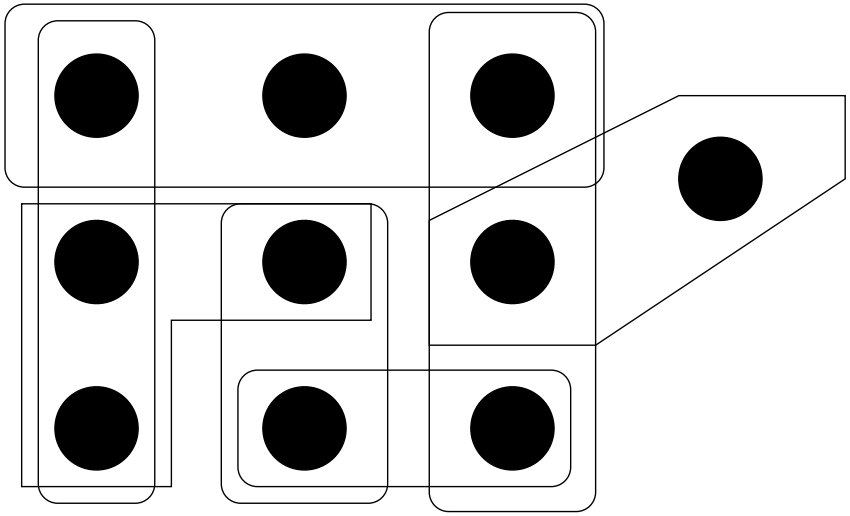
$$SD \cup OBS \cup \{\neg ab(c) \mid c \in C'\}$$

ein Widerspruch zu erkennen ist.

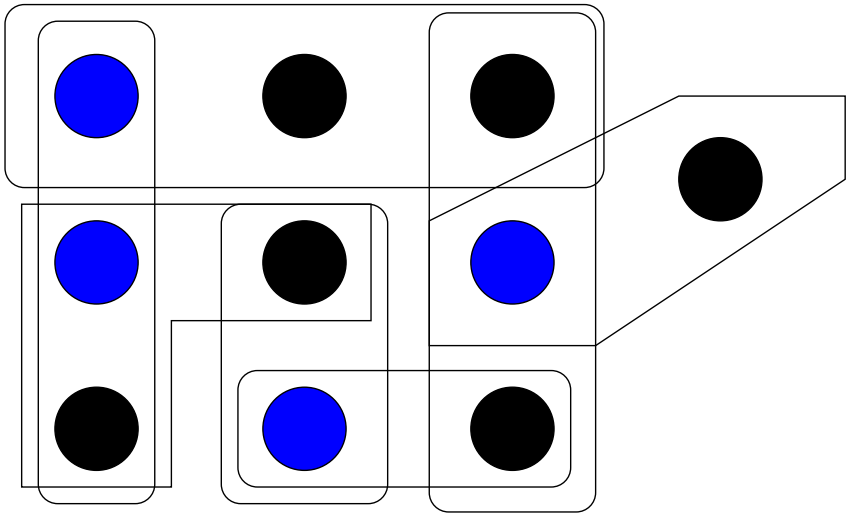
Eine *Diagnose* ist eine möglichst kleine Menge C' von Komponenten, so dass $C \setminus C'$ *keine* Konfliktmenge ist.

Übersetzung in Hitting Set:
Die Hypergraphknoten sind die Komponenten,
die Konfliktmengen sind die Kanten,
die Diagnose die Überdeckungsmenge.

Ein abstrakteres Beispiel



Eine kleinste Überdeckung



Datenreduktionsregeln

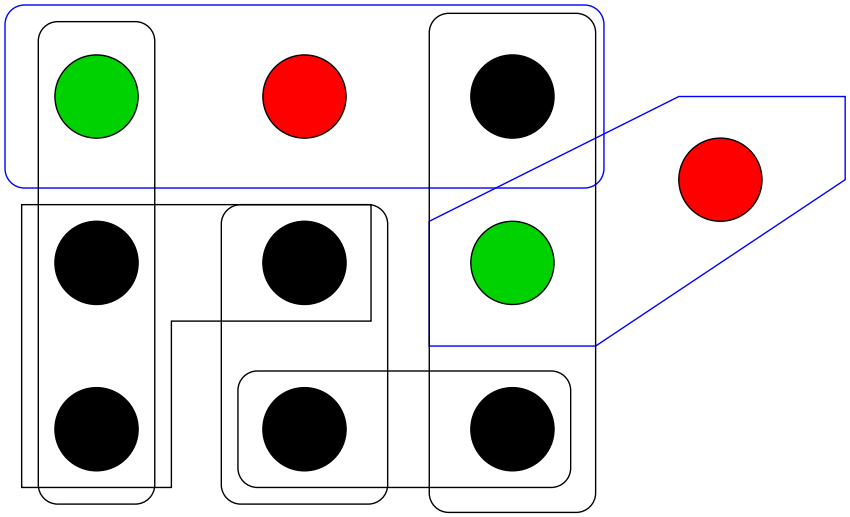
1. Kantendominierung: $f \subset e$. $\rightsquigarrow$ entferne e

2. Kleine Kanten: $e = \{v\} \rightsquigarrow v$ kommt ins HS; entferne e (und alle anderen Kanten, die v enthalten)

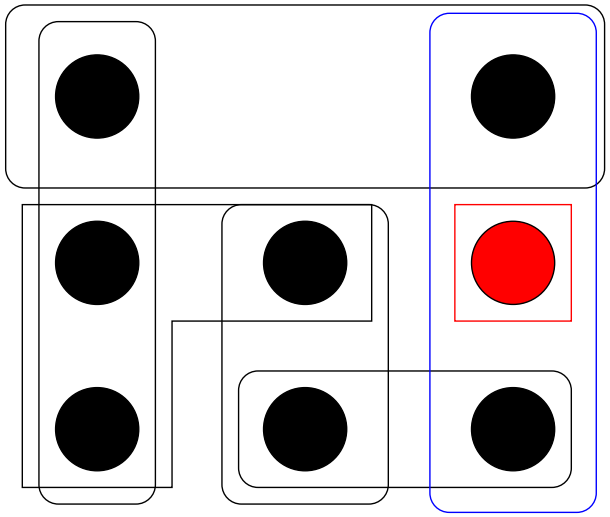
3. Knotendominierung: Ein Knoten x heiße *dominiert* durch einen Knoten y, falls $\{e \in E \mid x \in e\} \subseteq \{e \in E \mid y \in e\} \rightsquigarrow$ entferne x

R. S. Garfinkel and G. L. Nemhauser. *Integer Programming*. John Wiley & Sons, 1972.
Oft wiederentdeckt: K. Weihe (Zugnetzoptimierung), R. Niedermeier & P. Rossmanith (param. HS, 2003), Ebenso: R. Reiter (Theory of Diagnosis $\rightsquigarrow$ HS Bäume, 1987)

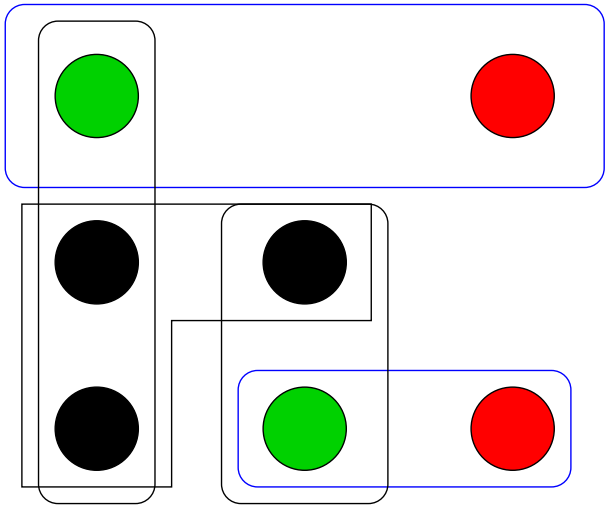
Knotendomination



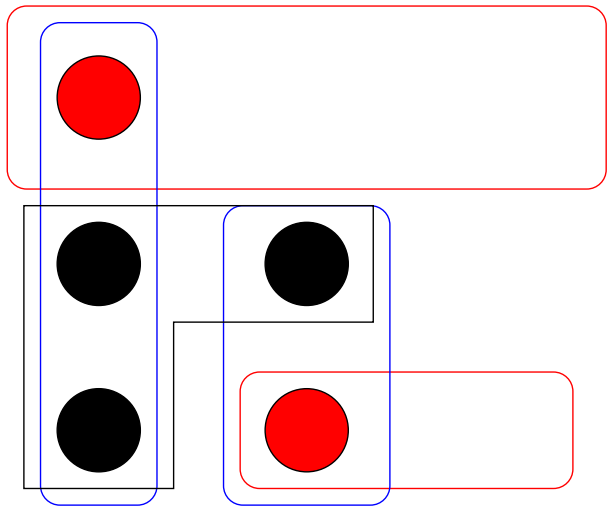
Kantenregeln



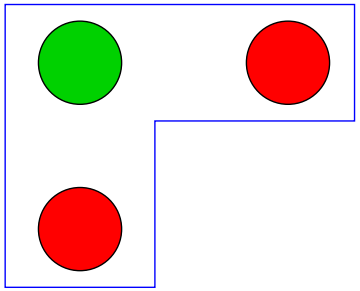
Knotendomination



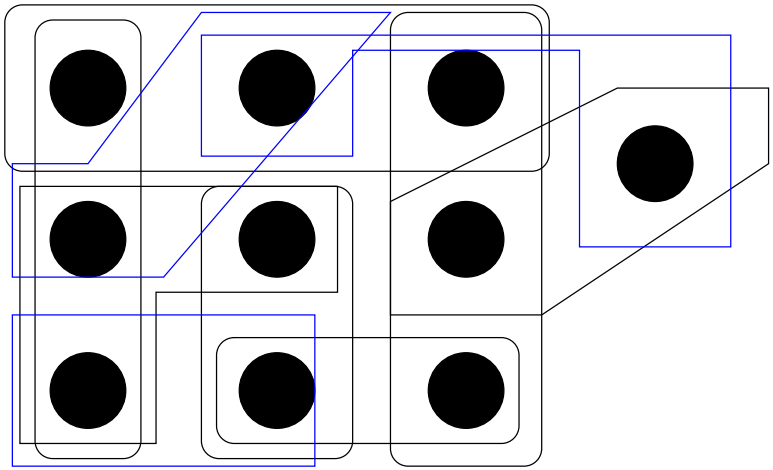
Kantenregeln



Knotendomination



Ein irreduzibles Beispiel

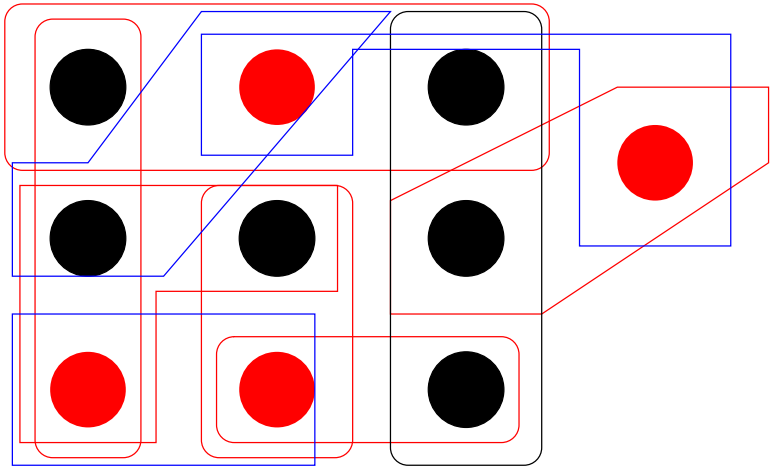


Ein Näherungsverfahren — übersetzt von VC $N1 - 3HS(G = (V, E), C)$

- (Wende Reduktionsregeln an.)
- Falls E leer, gib C aus; exit.
- Nimm irgendeine „kleine Kante“ e aus G
  und berechne $N1 - 3HS(G - e, C \cup e)$ (e aufgefasst als Knotenmenge)

Dies ist ein **3**-Approximations-Verfahren.

Ein irreduzibles Beispiel wird approximiert



Wie gut ist die Approximation „wirklich“ ?

Faktor 3 nur schlimmster Fall !
Wir erreichen 5 Knoten-Lösung mit dem ‚banalen‘ Algorithmus.
Da unser Beispiel das frühere als Teilfall enthält, wissen wir:
4 ist eine untere Schranke.
Am Lauf des Algorithmus sehen wir:
zweimal haben wir zwei Knoten statt möglicherweise einem genommen, beim dritten Schritt waren wir optimal
$\rightsquigarrow$ untere Schranke 3 (auch ohne das früher behandelte Teilproblem).
$\rightsquigarrow$ Der Satz über lokale Verhältnisse gestattet das Auffinden besserer Schranken im konkreten Beispiel.
Tatsächlich gibt es Lösung mit 4 Knoten, und die findet unser Algorithmus ‚fast‘.