Näherungsalgorithmen

(Approximationsalgorithmen)

WiSe 2006/07 in Trier

Henning Fernau
Universität Trier
fernau@informatik.uni-trier.de

$\boxed{\textbf{Näherungsalgorithmen}}$ Gesamtübersicht

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

Organisatorisches

Vorlesung: Montag 16-18 Uhr, H6,

Übungen (Daniel Raible): Dienstag 10-12 Uhr, H6, vierzehntägig, Beginn 2. Semesterwoche

Meine Sprechstunde: MO, 14-15 Uhr

Kontakt: fernau@informatik.uni-trier.de

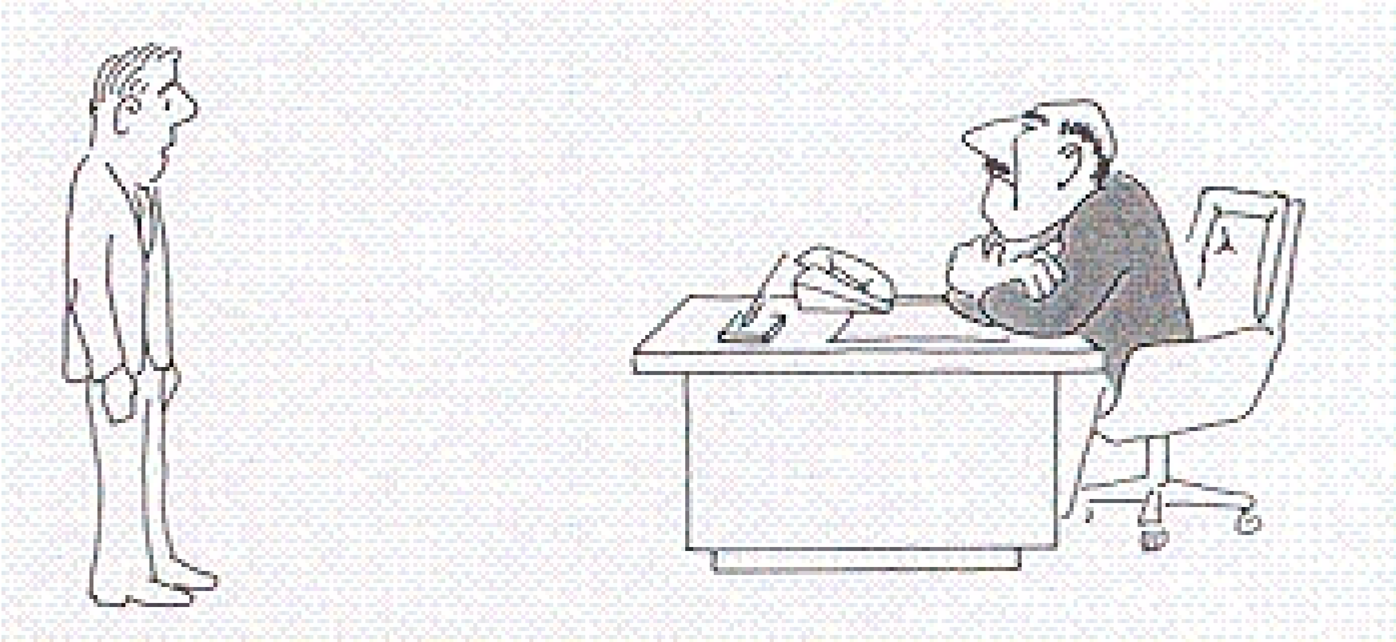
Hausaufgaben / Schein ?! n.V.

Motivation

Viele interessante Probleme (aus der Praxis!) sind NP-hart
$\Rightarrow$ wohl keine Polynomialzeitalgorithmen sind zu erwarten.

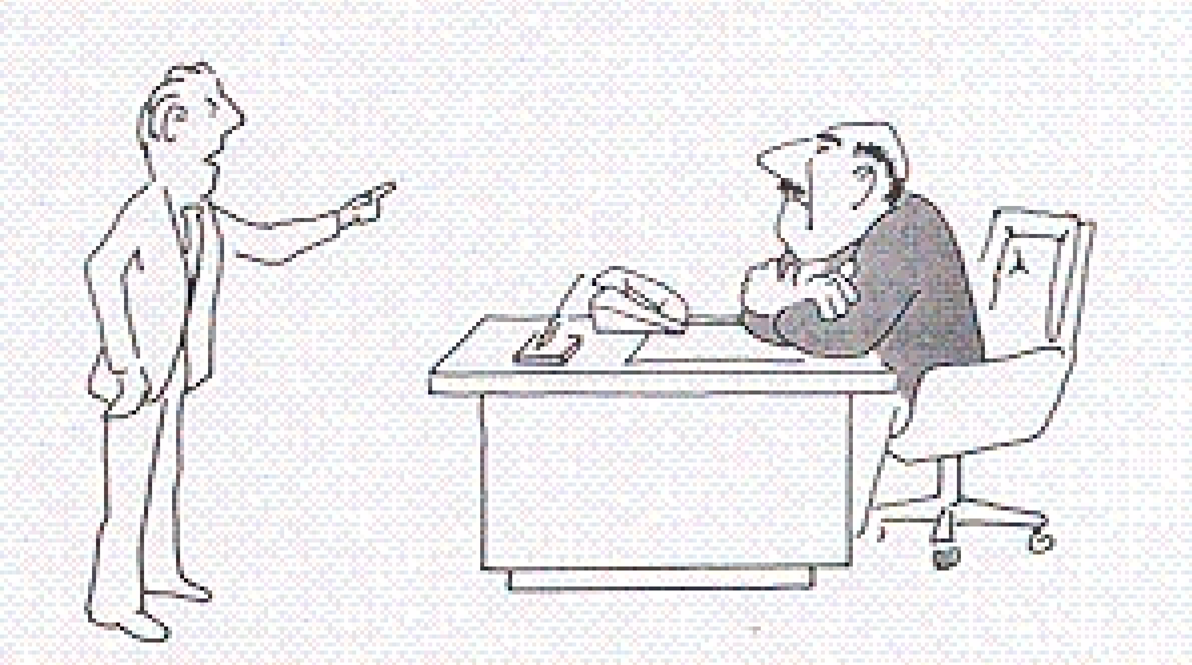
Motivation

siehe `http://max.cs.kzoo.edu/~kschultz/CS510/ClassPresentations/NPCartoons.html` wiederum aus Garey / Johnson



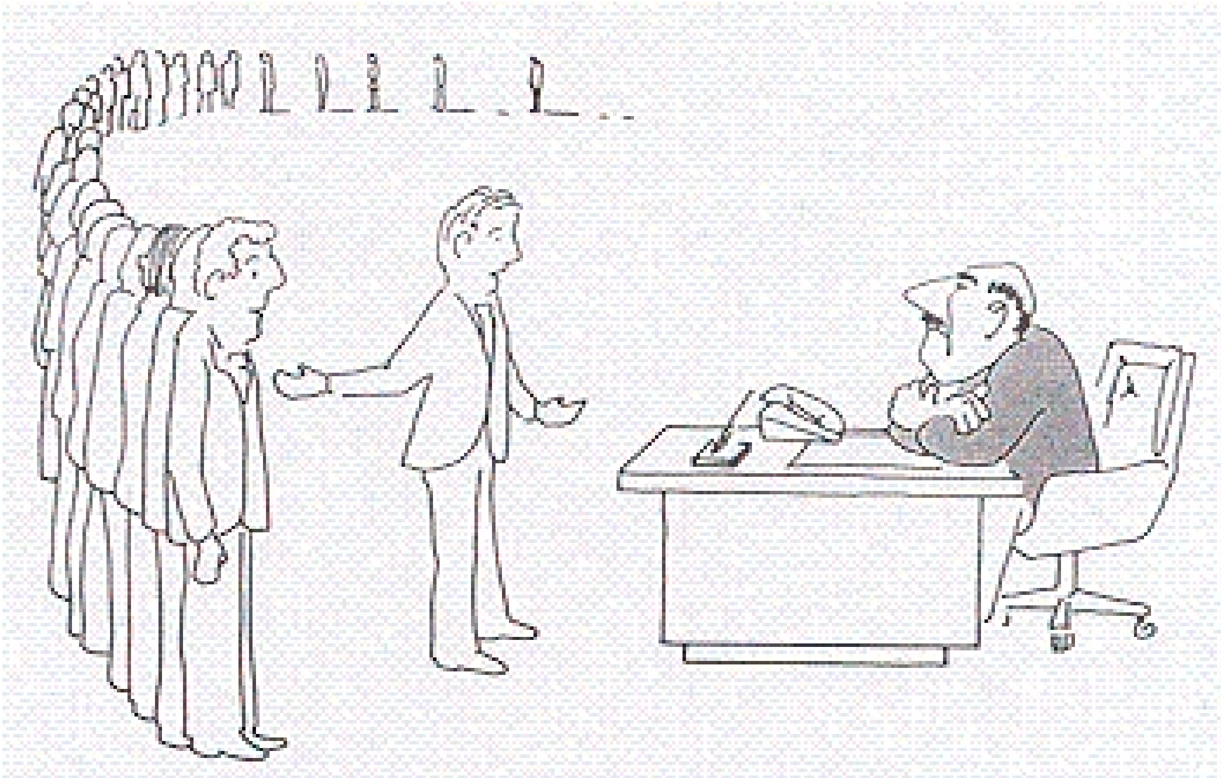
Sorry Chef, aber ich kann für das Problem keinen guten Algorithmus finden. . .

Die beste Antwort wäre hier aber...



...Ich kann aber beweisen, dass es für das Problem keinen guten Algorithmus geben kann !

Was die Komplexitätstheorie statt dessen liefert. . .



. . . Ich kann aber beweisen, dass das alle anderen auch nicht können !

Heuristische Verfahren

- Ziel: schnelle Laufzeit
- „hoffentlich“ wird „gute“ Lösung gefunden.
  $\hookrightarrow$ keine „mathematische“ Garantie, nur „Empirie“.
- typische Beispiele: Greedy-Verfahren

Randomisierte Verfahren

- finden optimale Lösung „mit großer Wahrscheinlichkeit“.
  Hinweis: Eine Vorlesung „randomisierte Algorithmen“ wird manchmal angeboten; oder auch Seminar zum Thema.

Parametrisierte Verfahren

- finden stets optimale Lösung.
- versuchen, den nicht-polymiellen Laufzeitanteil auf einen (als klein angenommenen) sogenannten Parameter zu beschränken.
  Hinweis: Spezialvorlesung „parametrisierte Algorithmen“
  wird im Wechsel mit „Näherungsalgorithmen“ angeboten

Näherungsverfahren

- sind „Heuristiken mit Leistungsgarantie".

- Güte von Näherungsverfahren kann
  - absolut oder
  - relativ zum Optimum gemessen werden

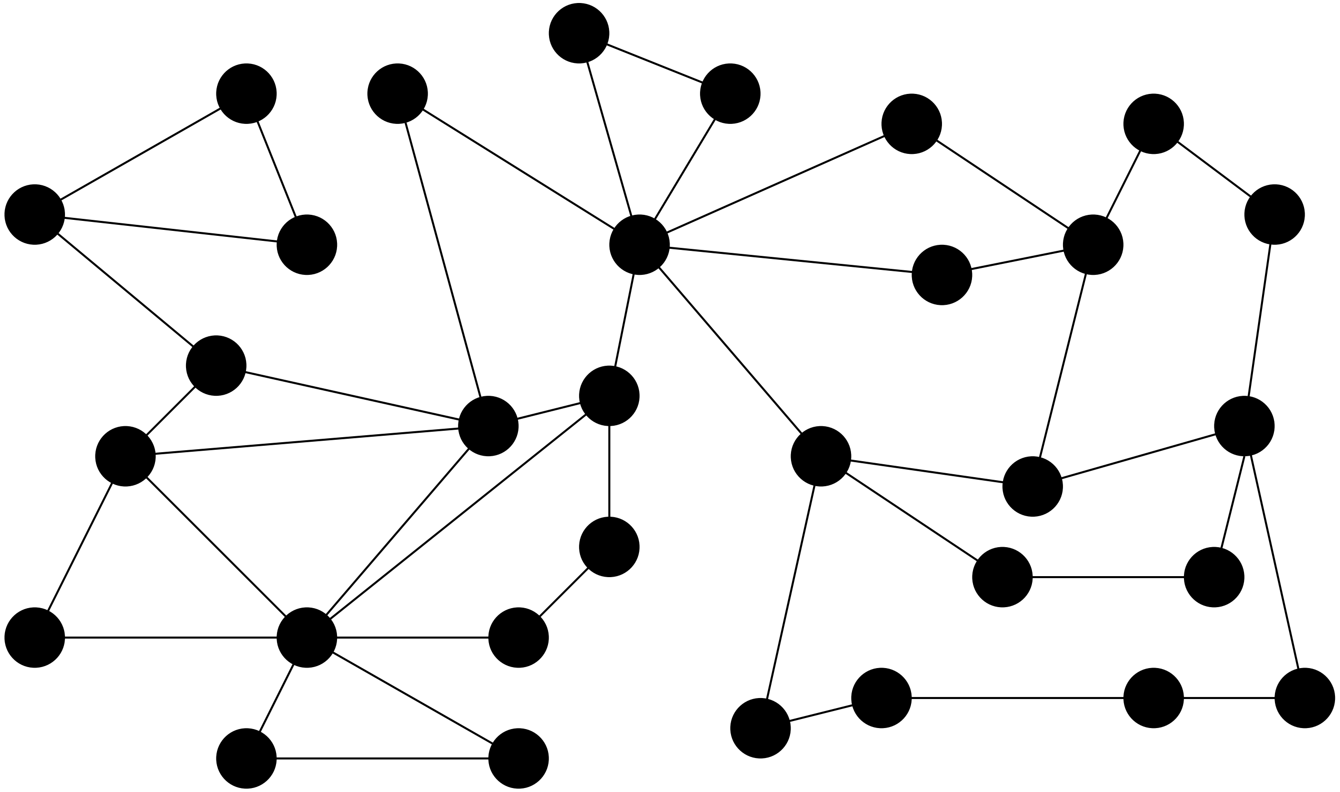
Eine erste Definition für ein einführendes Beispiel

Eine *Knotenüberdeckung* (engl: vertex cover) eines Graphen $G = (V, E)$ ist eine Menge $C \subseteq V$ derart, dass
für jede Kante $e = \{v_1, v_2\} \in E$ gilt: $e \cap C \neq \emptyset$, d.h.,
e wird durch einen Knoten aus C abgedeckt.

Knotenüberdeckungsproblem VC: Finde kleinstmögliche Knotenüberdeckung !

Hinweis: VC ist eines der grundlegenden NP-harten Probleme.

Ein Beispiel: Wie groß ist ein kleinstmögliches VC ?

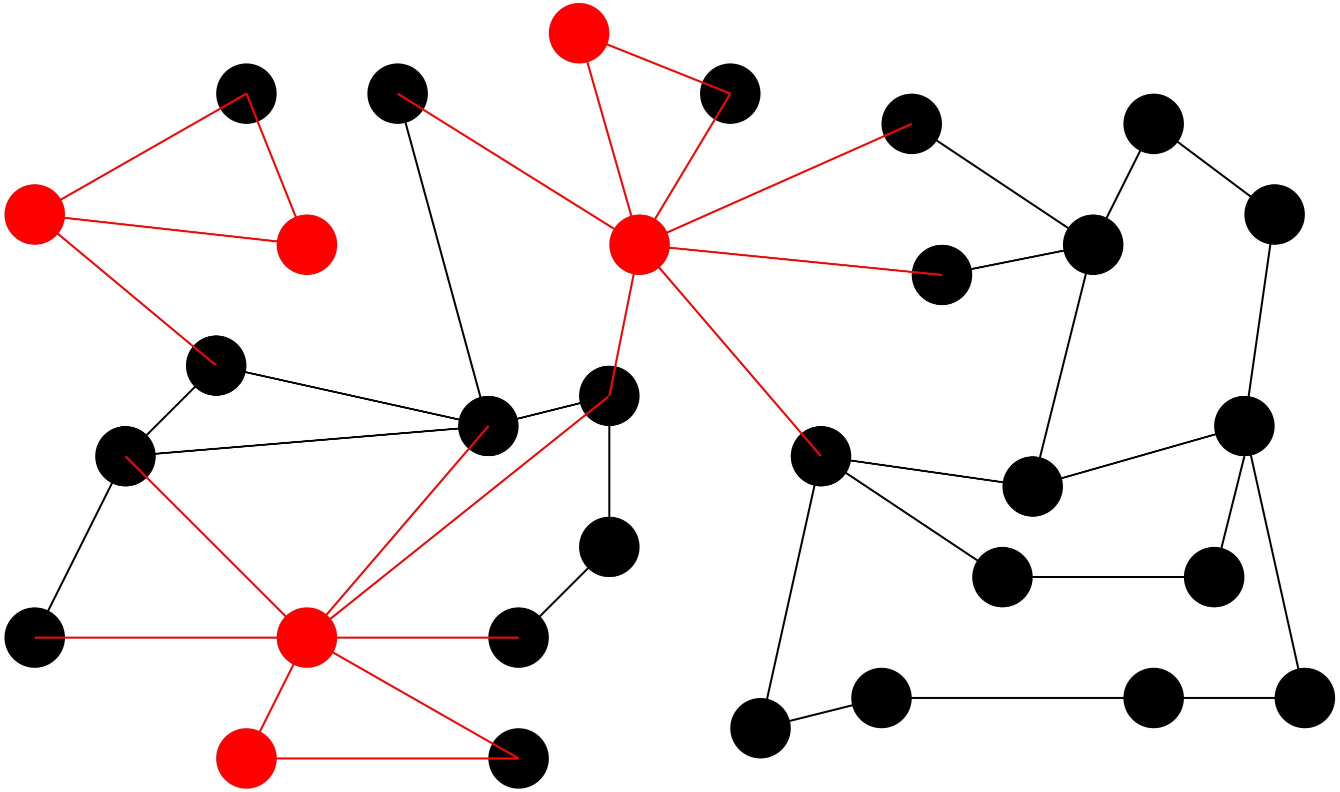


Einfache Beobachtungen und Regeln

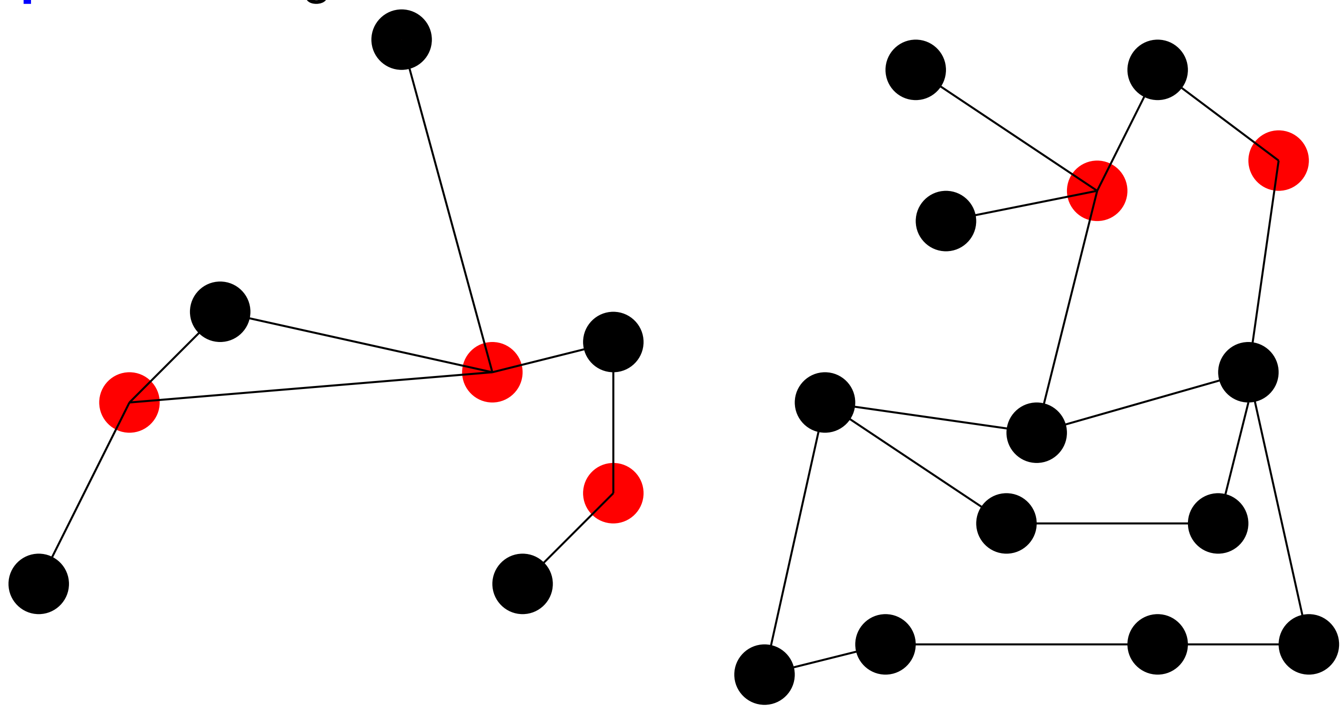
- Zwei Knoten in einem Dreieck gehören in ein VC.

- Wenn Wahlmöglichkeit besteht, nimm solche Knoten, die “noch mehr” abdecken können.
  $\leadsto$ Nimm Nachbarn eines Grad-1-Knotens ins VC.
  $\leadsto$ Gibt es in einem Dreieck einen Knoten vom Grad zwei, so nimm dessen beide Nachbarn ins VC.

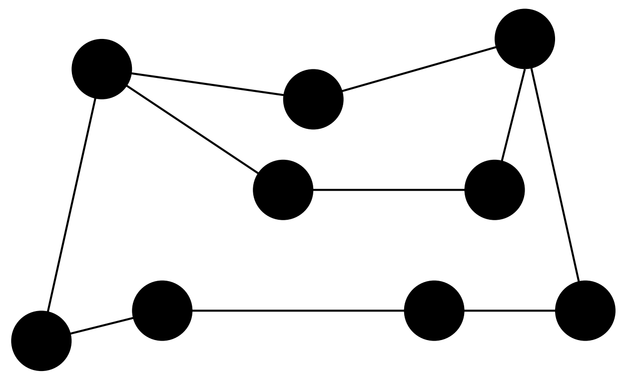
Unser Beispiel: Wir wenden die Regeln an und finden 6 VC-Knoten.



Unser Beispiel: Die Regeln kaskadieren $\leadsto$ 5 weitere VC-Knoten.



Unser Beispiel: Was bleibt übrig ?

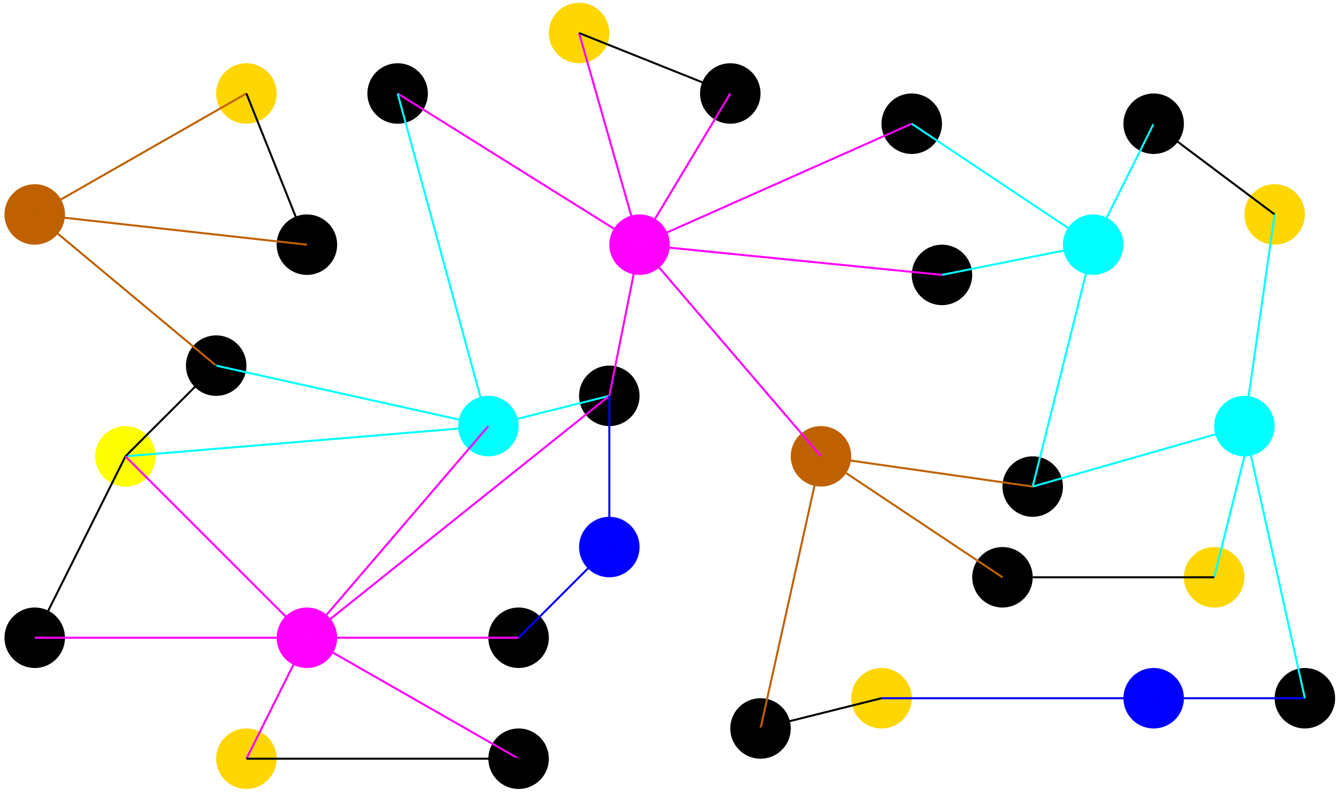


Ein Greedyverfahren GreedyVC (G, C)

- Falls G leer, gib C aus; exit.
- Suche Knoten v mit maximalem Grad in G.
- Berechne GreedyVC $(G - v, C \cup \{v\})$

Hinweis: $G - v$ entsteht aus G, indem v mit anliegenden Kanten aus G gelöscht wird.

Unser Beispiel: Greedy at work... Farben kodieren Grad $\rightsquigarrow$ 16 VC-Knoten



Ein Suchbaumverfahren SuchbaumVC(G, C, k)

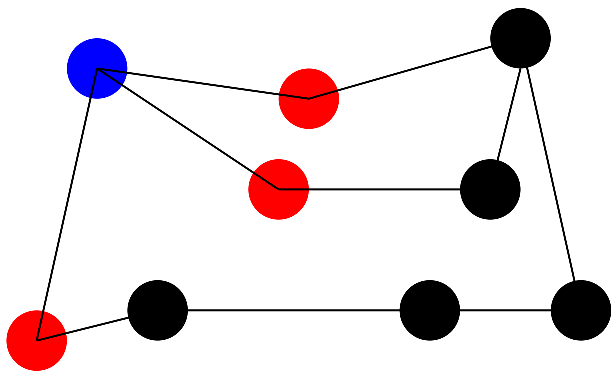
- Falls $E(G)$ leer, gib C aus; exit.

- Falls $k = 0$: exit (Fehlerfall).

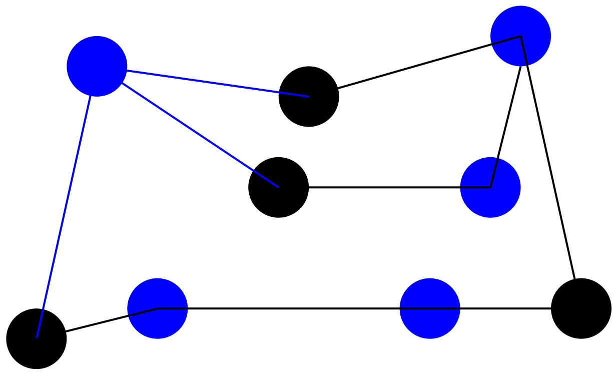
- Nimm irgendeine Kante $e = \{v_1, v_2\}$ aus G und verzweige:

  1. Berechne SuchbaumVC$(G - v_1, C \cup \{v_1\}, k - 1)$
  2. Berechne SuchbaumVC$(G - v_2, C \cup \{v_2\}, k - 1)$
  3. Liefere kleinere Lösung “nach oben”.

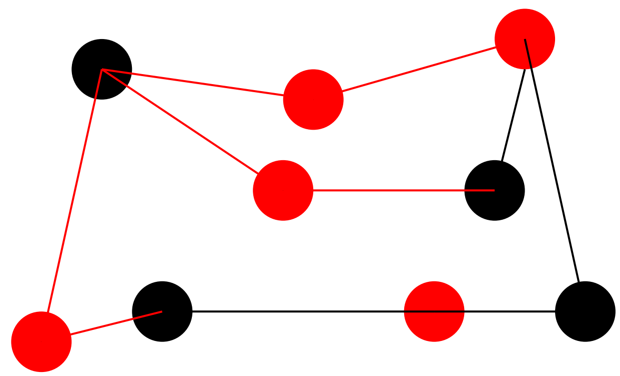
Weitere Idee: Knotenorientierter Suchbaum, kombiniert mit den Regeln $\leadsto$ zwei Zweige



Weitere Idee: Kombiniere Suchbaum mit unseren Regeln der blaue Fall



Weitere Idee: Kombiniere Suchbaum mit unseren Regeln der rote Fall



Ergebnis für unser Beispiel:

Die kleinste Knotenüberdeckung enthält 16 Knoten.

Eine kleinste Knotenüberdeckung wurde von der Heuristik gefunden.

Die Verwendung von sog. Reduktionsregeln hilft, die Verzweigungszahl bei Suchbaumalgorithmen zu verringern.

Ein Näherungsverfahren $N1VC(G = (V, E), C)$

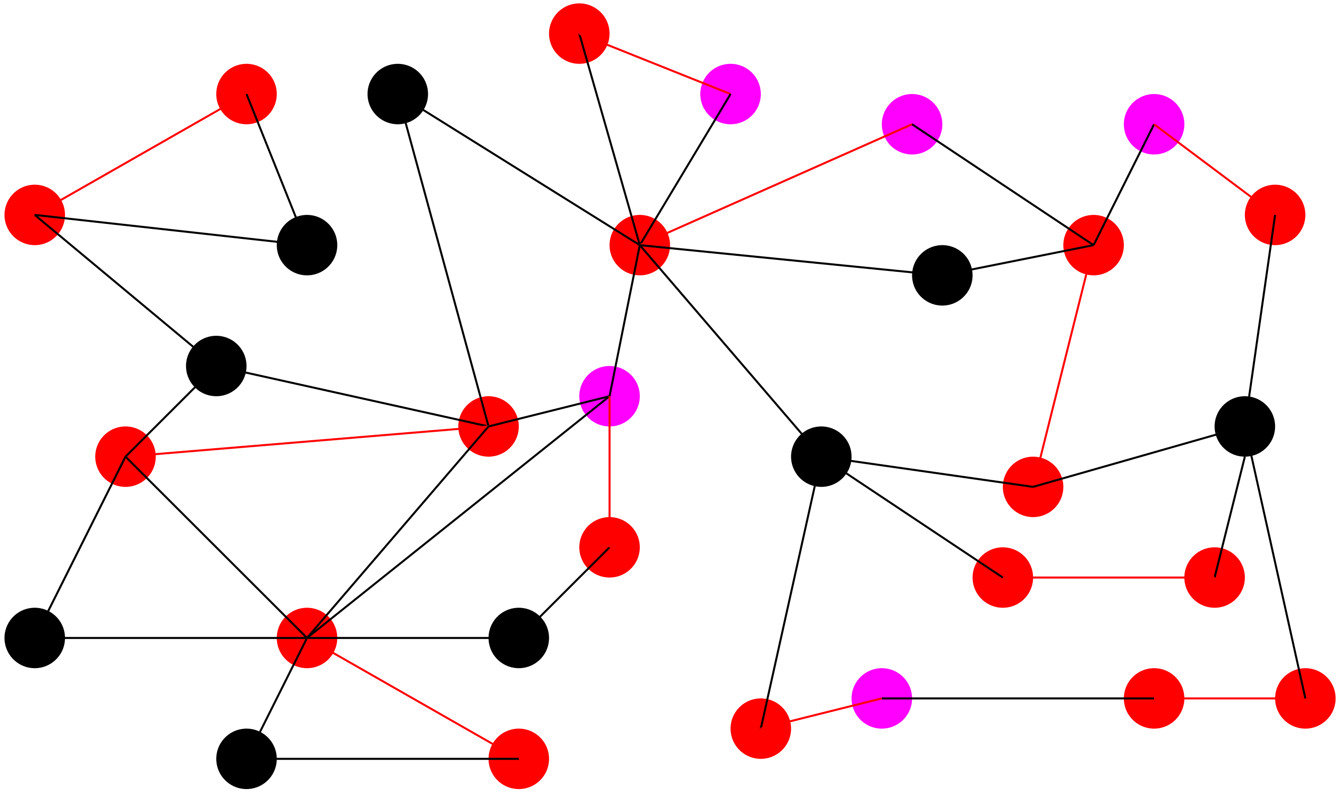
- Falls E leer, gib C aus; exit.

- Nimm irgendeine Kante $e = \{v_1, v_2\}$ aus G
  und berechne $N1VC(G - \{v_1, v_2\}, C \cup \{v_1, v_2\})$

Dies ist ein **2**-Approximations-Verfahren (s.u.)

Unser Beispiel: magentafarbene Knoten gehören nicht zu *(inklusions-)minimaler* Überdeckung; die gefundene Lösung ist daher fast bestmöglich

Wichtig: Minimal (Greedy) versus Minimum VC (NP-hart).



Ein allgemeines Überdeckungsproblem

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

- X eine endliche Menge ist
- $f : 2^X \to \{0, 1\}$ eine monotone Abbildung ist, d.h.
  $A \subseteq B \to f(A) \leq f(B)$, und es gelte $f(X) = 1$,
- $w : X \to \mathbb{R}^+$ ist die Gewichtsfunktion