Parameterisierte Algorithmen

WS 2011/12 in Trier

Henning Fernau
fernau@uni-trier.de

Parameterisierte Algorithmen | Gesamtübersicht

- Einführung
- Grundbegriffe

- Problemkerne
- Suchbäume
- Graphparameter
- Weitere Methoden
- Komplexitätstheorie—parameterisiert

Wie erhalte ich Kernreduktionen?

- Fokus auf Extremen (z. B. kleingradige / großgradige Knoten)
- Verwendung bekannter (nicht-trivialer) mathematischer Aussagen
- Dafür häufig wichtig: Betrachtung annotierter Probleme
- Speziell: Einsatz von Katalysatoren
- Verallgemeinerung bekannter einfacher Regeln

Fokus auf Extremen bei Kernreduktionen

Bsp.: Buss' Regel bei VC

Wie wäre diese Regel für MMVC umzuformulieren?!

MMVC-Regeln

Reduktionsregel 1 *Lösche isolierte Knoten (Beibehaltung des Parameters)*

Reduktionsregel 2 *Ist v ein Knoten vom Grad größer gleich k, dann JA.*

Satz 1 *MMVC gestattet einen Kern der Größe k^2 (gemessen in der Knotenzahl).*

Wie sieht “vollständiges Regelsystem” aus ?

Wie erhalte ich Kernreduktionen?

- Fokus auf Extremen (z. B. kleingradige / großgradige Knoten)
- Verwendung bekannter (nicht-trivialer) mathematischer Aussagen
- Dafür häufig wichtig: Betrachtung annotierter Probleme
- Speziell: Einsatz von Katalysatoren
- Verallgemeinerung bekannter einfacher Regeln

Zurück zu Nonblocker: Knoten vom Grad 1

Erweiterte Instanz (mit Annotation):
Graph mit ausgezeichnetem Knoten c in der dominierenden Menge.
Wegen seiner Funktion heißt c auch Katalysator oder Dominator.

(Hinter-)Grund: Ist x Nachbarknoten eines Blattes, so kann x mit Dominator c zu neuem Dominator $[x, c]$ verschmolzen (identifiziert) und das Blatt entfernt werden. Diese Regel werden wir in Regel 4 verallgemeinern.

Mehr hilfreiche Mathematik... (ohne Beweis)

Satz 2 (Blank 1973; McCuaig / Shepherd 1989) *Hat ein zusammenhängender Graph* $G = (V, E)$ *Minimalgrad zwei und gehört er nicht zu einem von sieben "Ausnahme-Graphen" (jeweils mit maximal sieben Knoten), so besitzt* G *eine dominierende Menge mit höchstens* $2/5 \cdot |V|$ *vielen Knoten.*

Wir können diesen Satz ausnutzen für einen Problemkernalgorithmus durch Einführen von Katalysatorregeln und Dekatalysatorregeln.

Die “alten” Regeln (verallgemeinert)

Sei (G, c, k_d) eine Instanz von NB (annotiert mit Katalysator c).

Reduktionsregel 3 (verallgemeinerte Regel für isolierte Knoten) *Ist* C *eine vollständige Graphkomponente ohne* c*, reduziere zu* $(G - C, c, k_d - (|C| - 1))$*.*

Reduktionsregel 4 (verallgemeinerte Blattregel) *Gibt es* $x \in V(G)$ *mit* $U \subseteq N(x)$*, sodass* $N(U) \subseteq U \cup \{x\}$ *und* $c \notin U$*, dann reduziere zu der Instanz* $(G[c = x] - U, [c, x], k_d - |U|)$*.*

(De-)Katalysatorregeln: Hin zu annotierten Instanzen und zurück

Katalysatorregel Ist (G, k_d) eine Nonblocker-Instanz $G = (V, E)$, dann ist (G', c, k_d) eine äquivalente annotierte Instanz, wobei $c \notin V$ ein neuer Knoten ist und $G' = (V \cup \{c\}, E)$.

Dekatalysatorregel Sei (G, c, k_d) eine annotierte Nonblocker-Instanz. Die neue nicht-annotierte Nonblocker-Instanz (G', k'_d) entsteht wie folgt:

Verbinde c mit drei neuen Knoten u, v und w. Ferner führe neue Kanten uv und vw ein. Alle andere Knoten und Kanten bleiben unverändert. Dies beschreibt den neuen Graphen G'. Setze $k'_d = k_d + 3$.

Achtung: Keine echte Kernreduktion !

Algorithm 1 A kernelization algorithm for NB

Input(s): an instance (G, k_d) of NB

Output(s): an equivalent instance (G', k'_d) of NB with $V(G') \subseteq V(G)$, $|V(G')| \leq 5/3 \cdot k'_d$ and $k'_d \leq k_d$ OR ✓

if G has more than seven vertices **then**
  Apply the catalyzation rule.
  Exhaustively apply Rules 3 and 4 for neighborhoods U up to size two.
  Apply the decatalyzation rule.
  {This leaves us with a reduced instance (G', k'_d).}
  if $|V(G')| > 5/3 \cdot k'_d$ **then**
    return ✓
  else
    return (G', k'_d)
  end if
else
  Solve by table look-up and answer accordingly.
end if

Eine alternative Idee für das Einführen des Katalysators

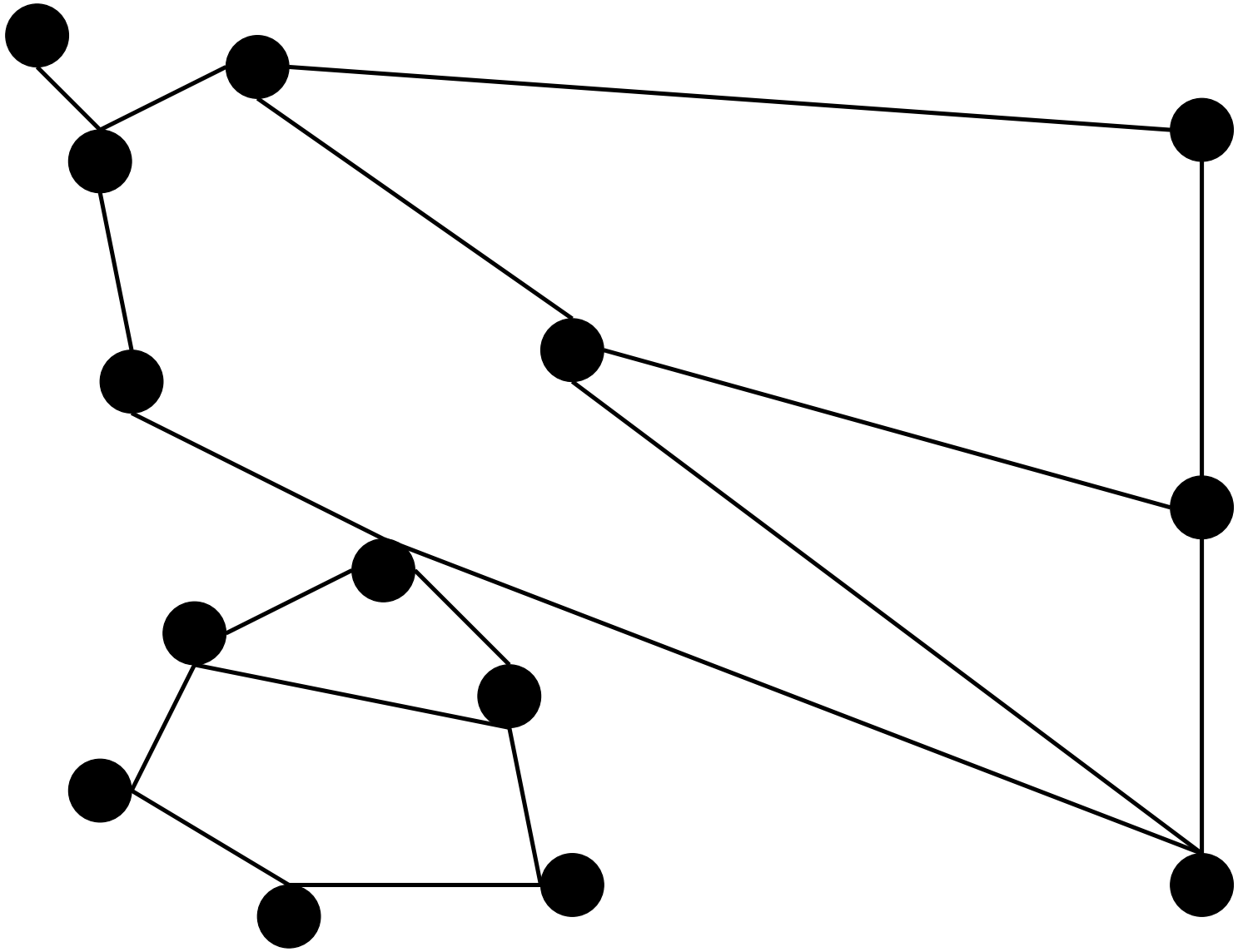
katalytisches Verzweigen anstelle der Katalysatorregel.

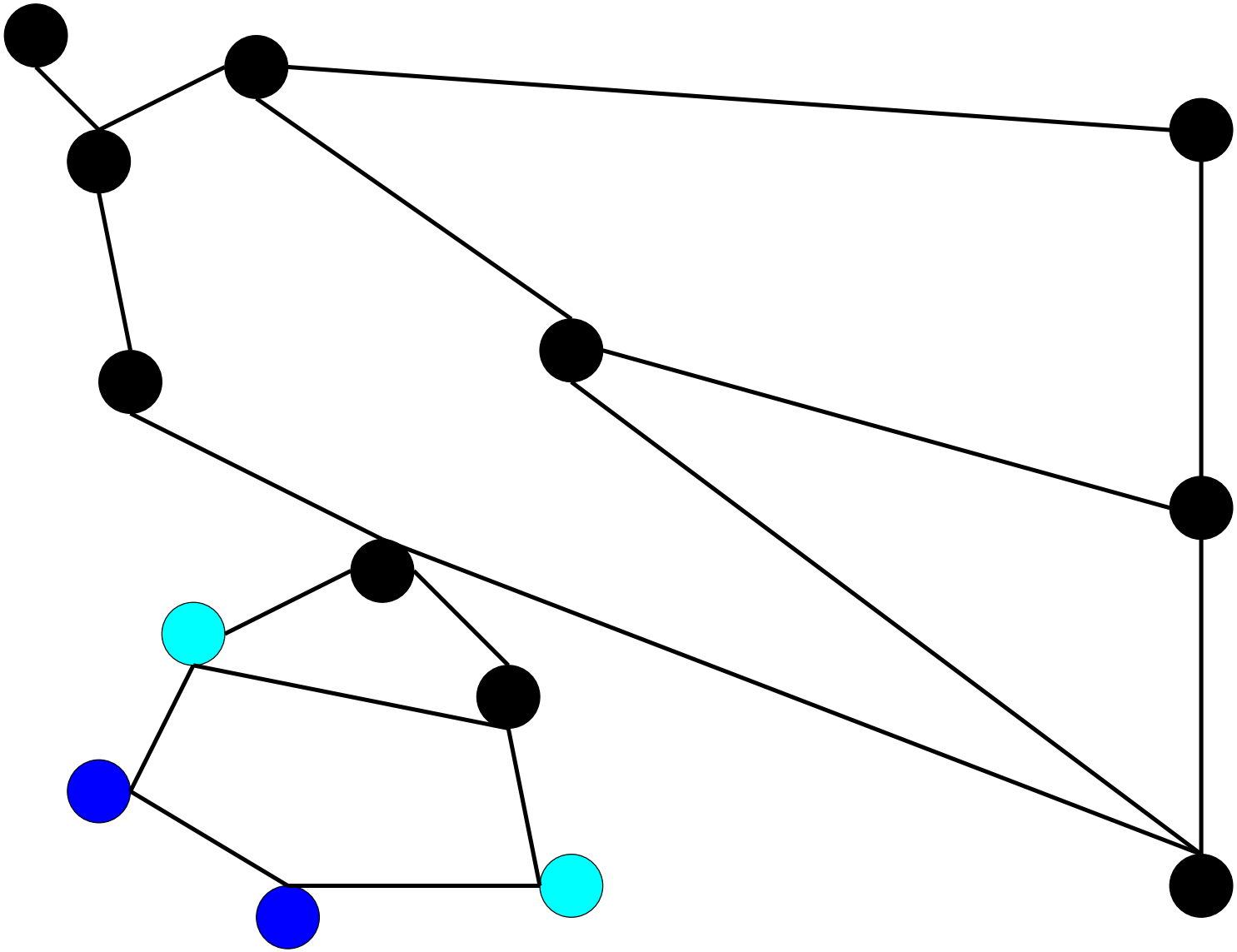
Die folgenden Folien zeigen eine “falsche Wahl”. (Katalysator in rot)

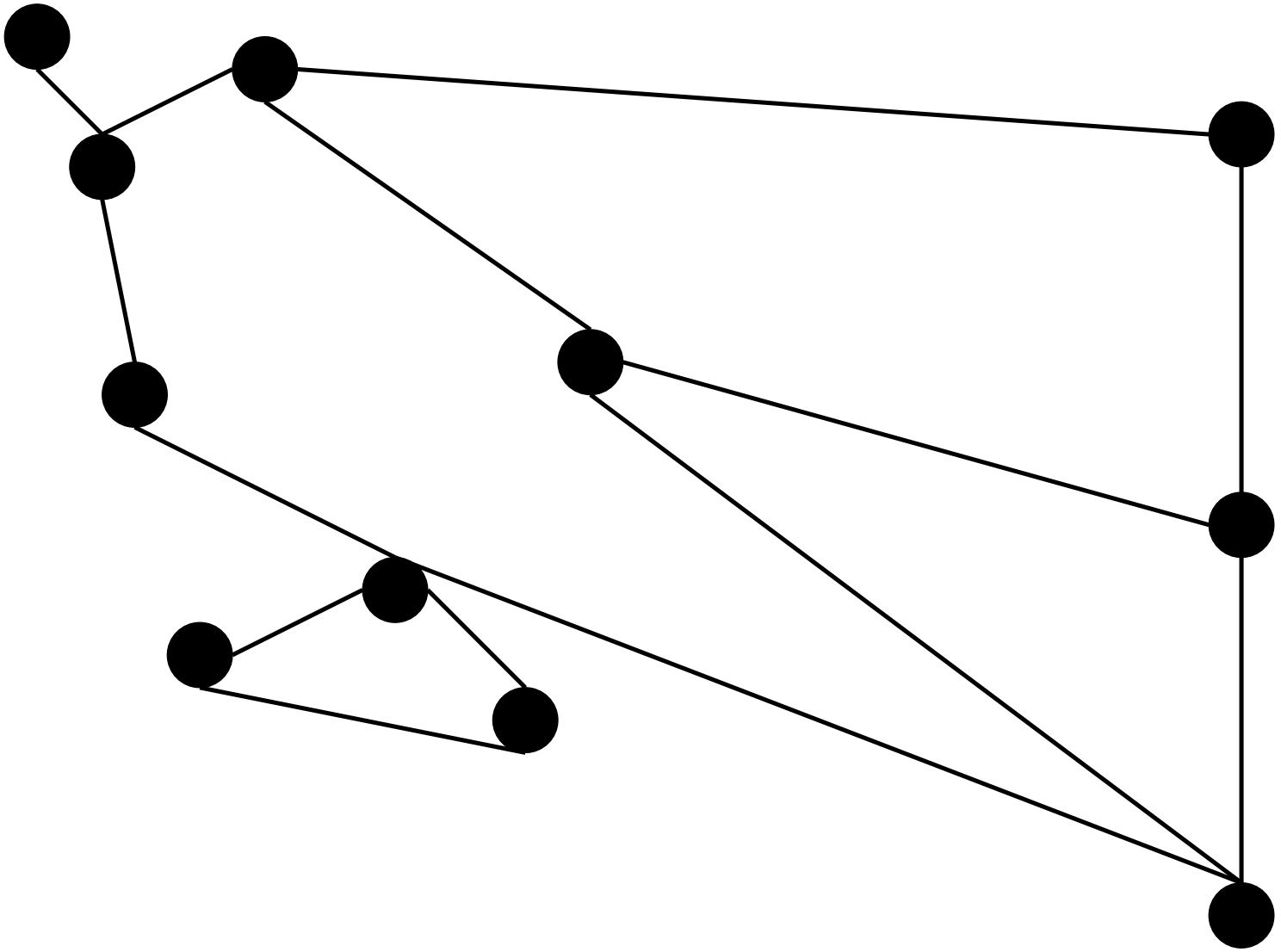
Es wird eine weitere Regel benutzt, um sämtliche Ausnahmegraphen abzuhandeln (insgesamt werden damit alle Graphen mit zwei benachbarten Knoten vom Grad zwei reduziert); dies erklärt auch, weshalb die Dekatalysatorregel nicht etwas einfacher sein kann.

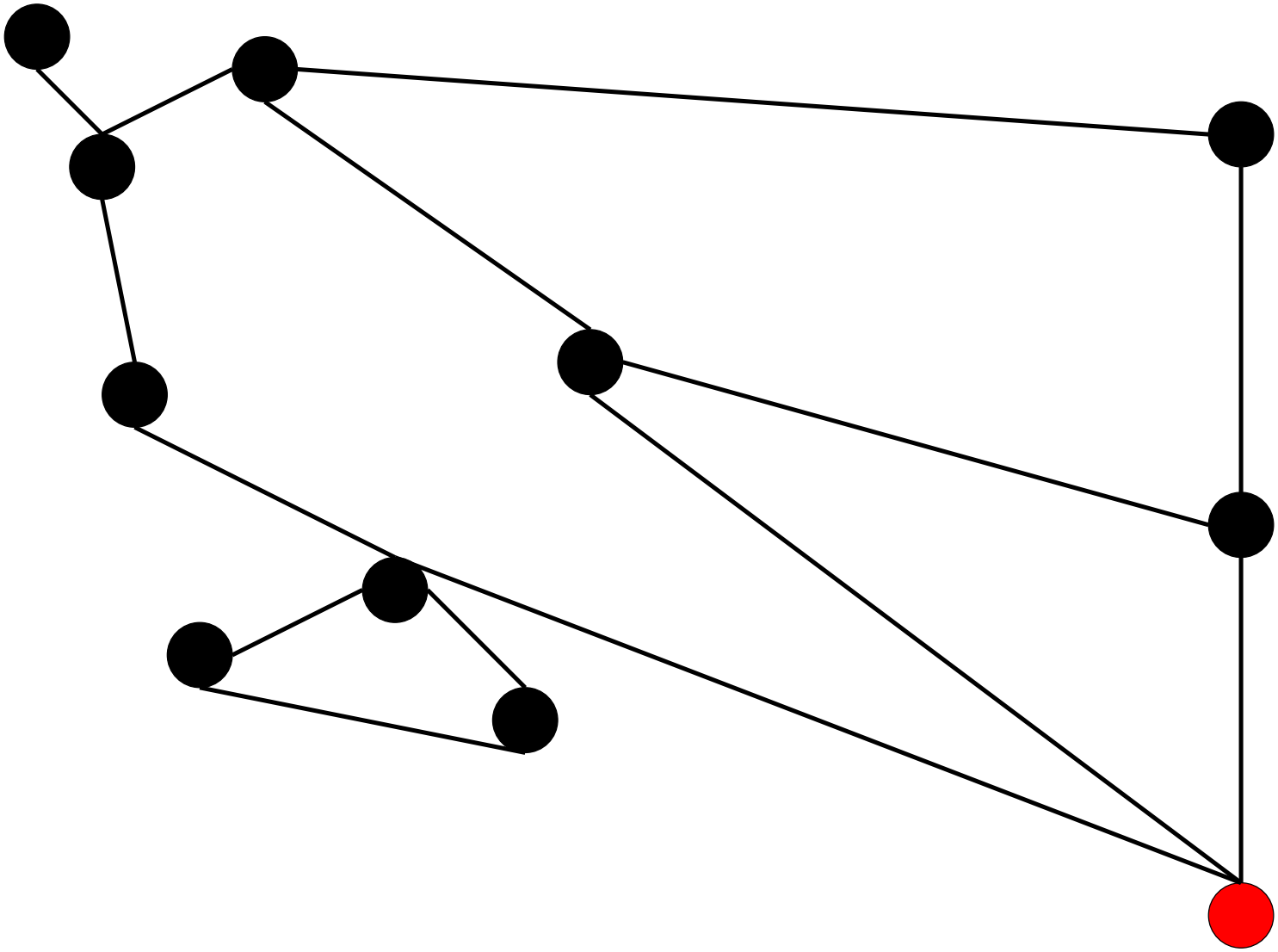
Reduktionsregel 5 *Falls in* $G = (V, E)$ *folgende Situation für* $v, v', u, u' \in V$ *vorliegt:* $\deg(v) = \deg(v') = 2$, $v \neq c$, $v' \neq c$, $N(v) = \{u, v'\}$, $N(v') = \{u', v\}$, $u \neq u' \rightsquigarrow$ *reduziere zu* $(G[u = u'] - \{v, v'\}, c, k_d - 2)$

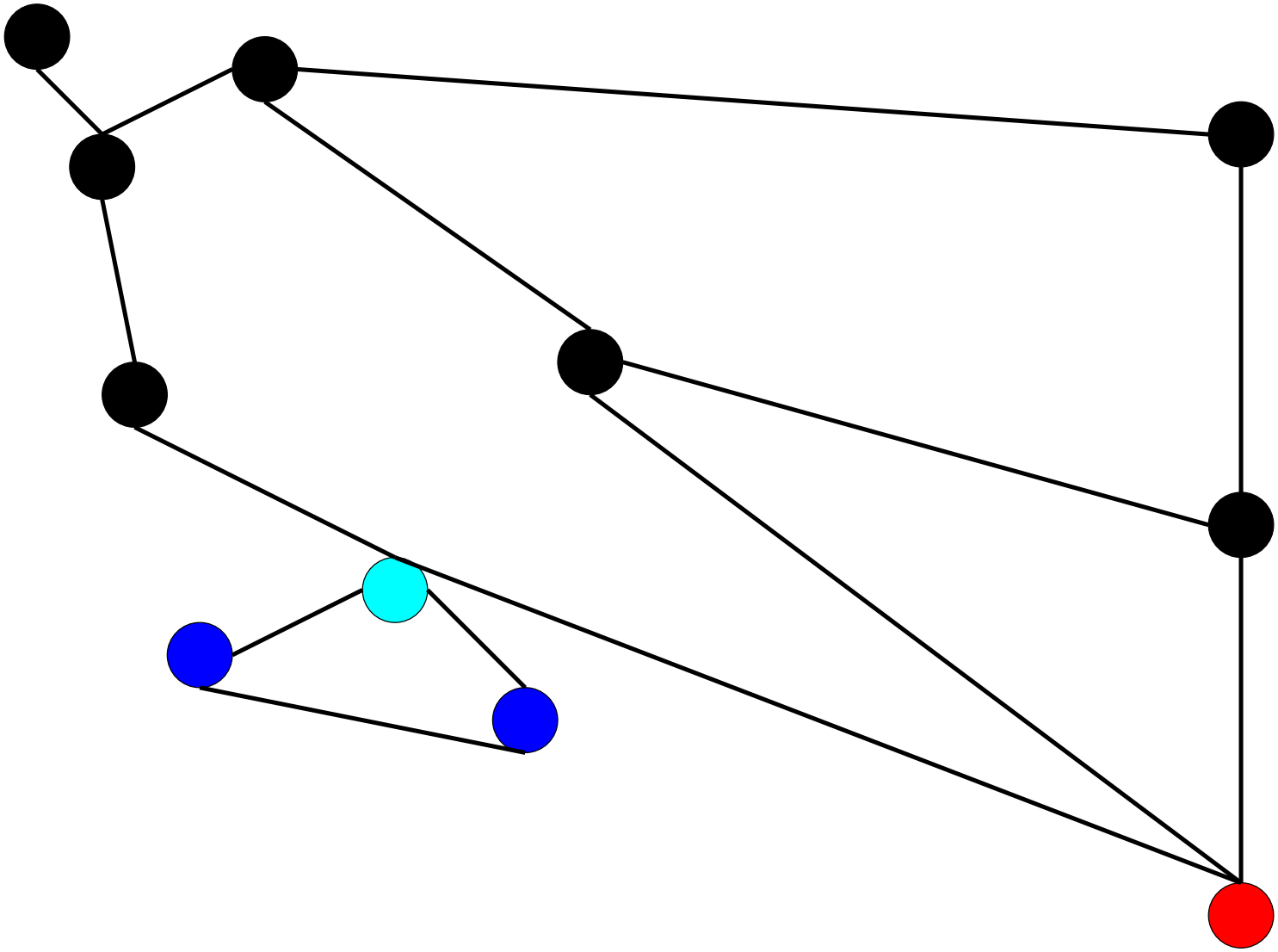
Im Folgenden wird die Arbeit der Reduktionsregeln an einem Beispiel gezeigt; außerdem sieht man, dass z.B. die letzte Regel auch ohne Katalysator arbeitet. Dann wird die Lösung “rückwärts” für den ursprünglichen Graphen erarbeitet und schließlich eine bessere Katalysatorwahl gezeigt.

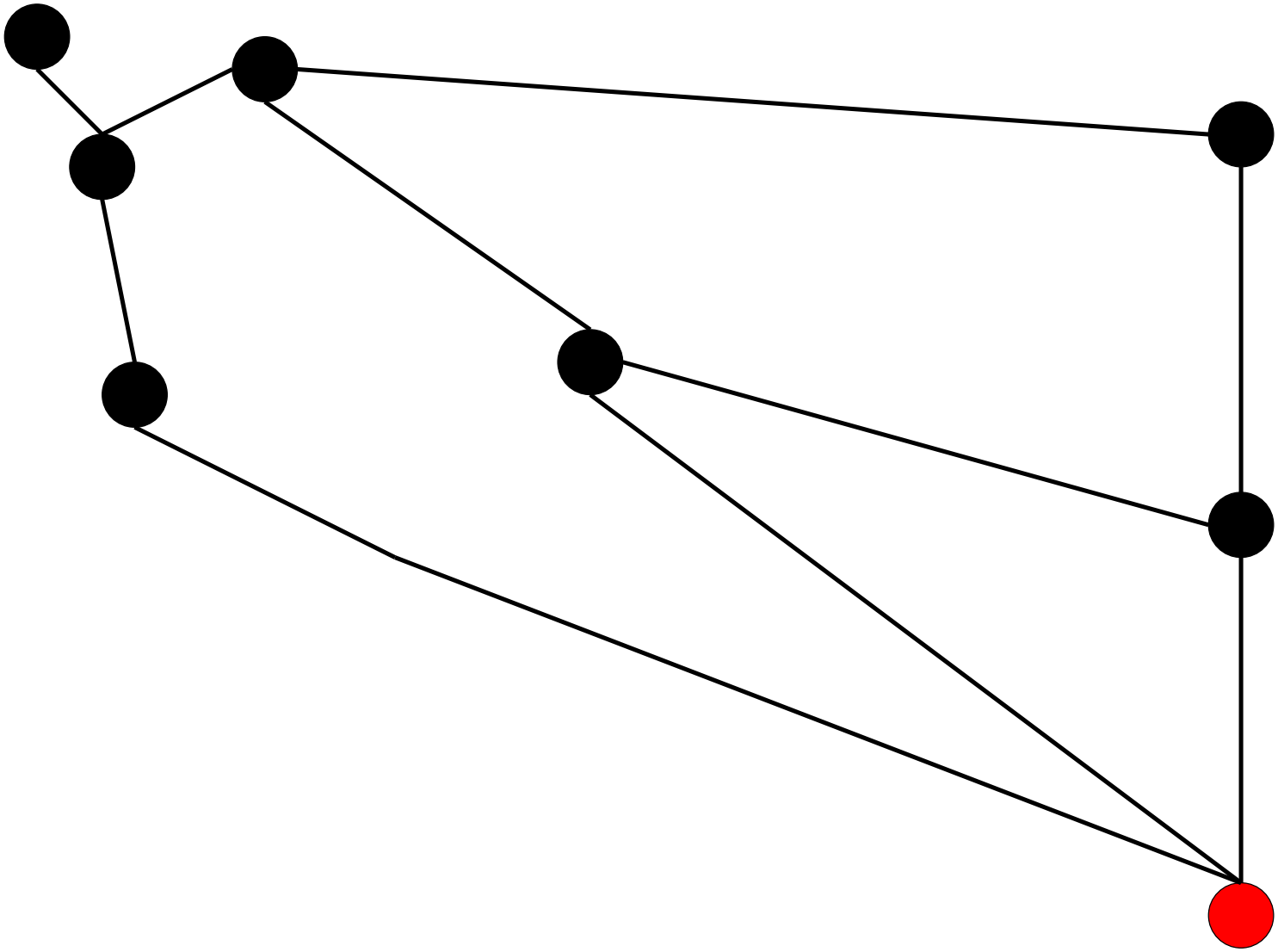


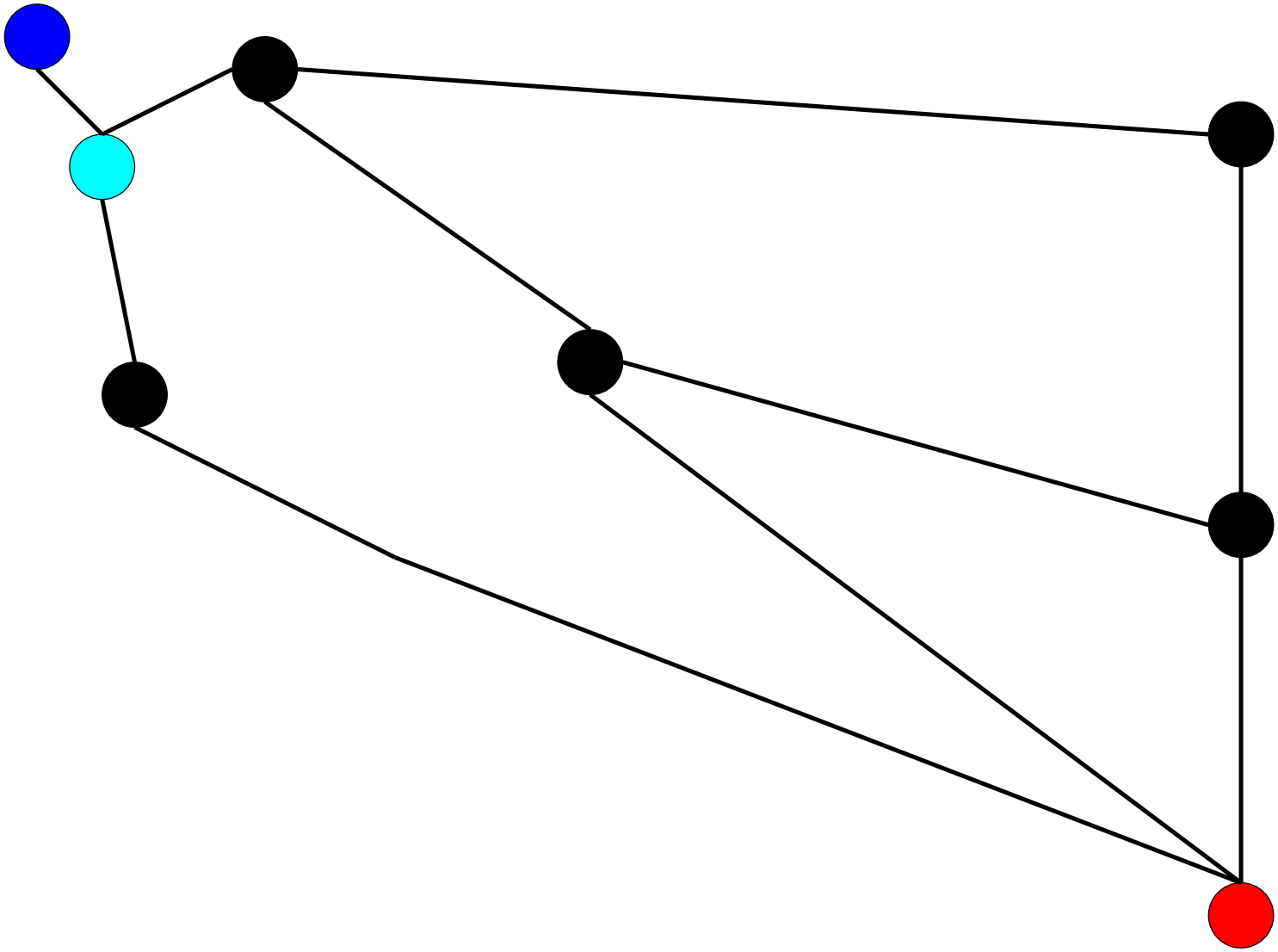


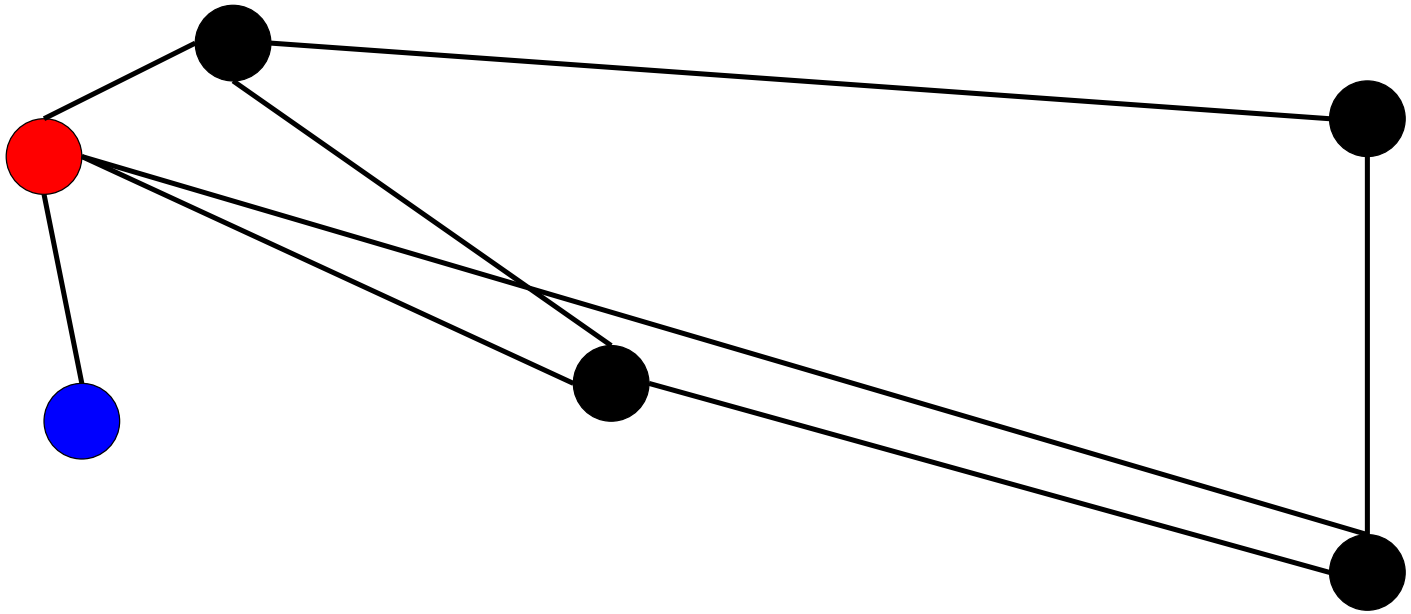


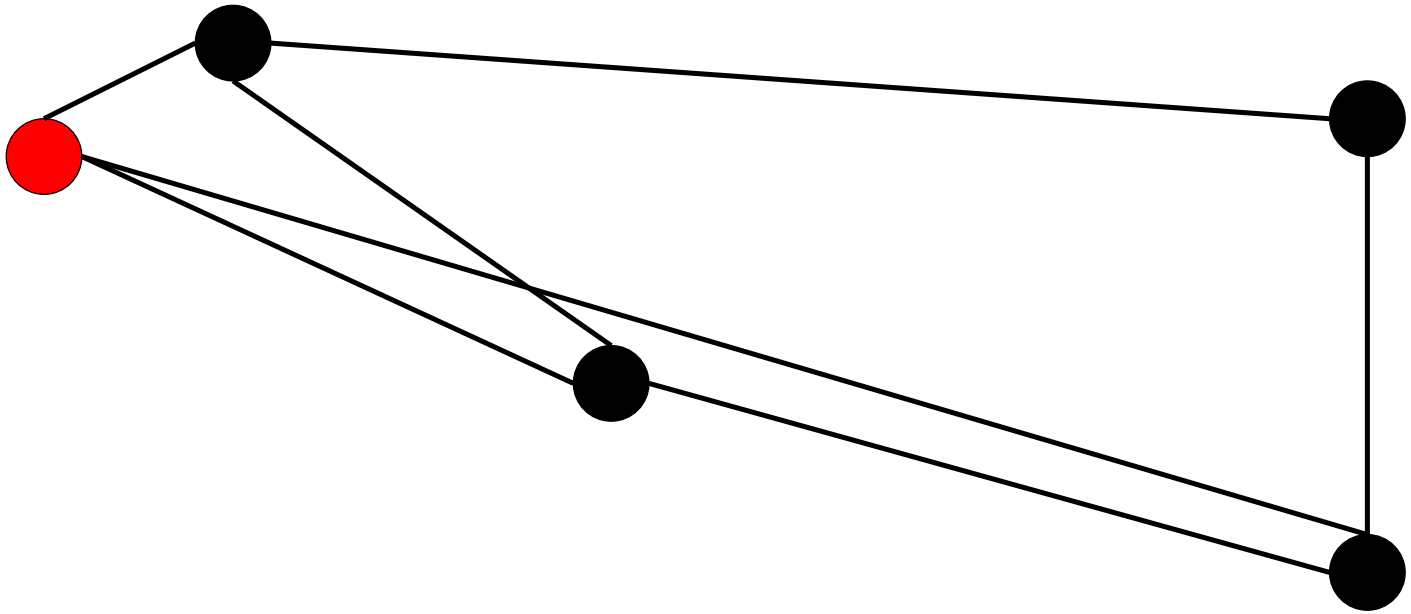


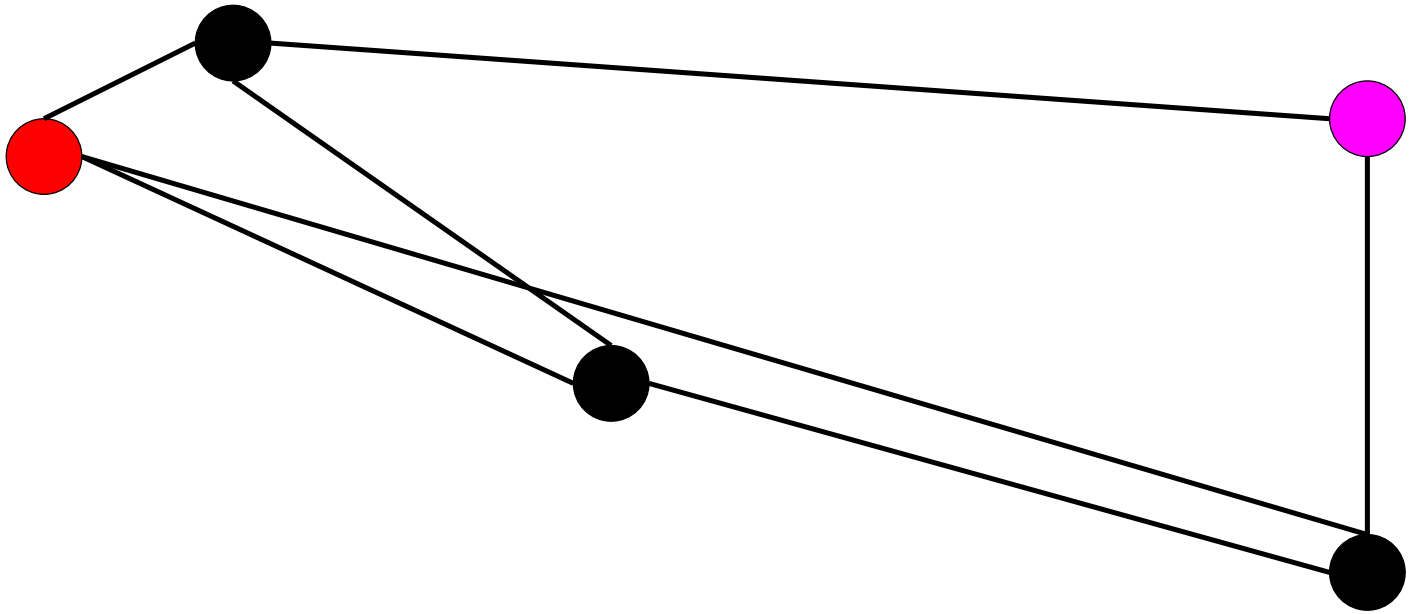


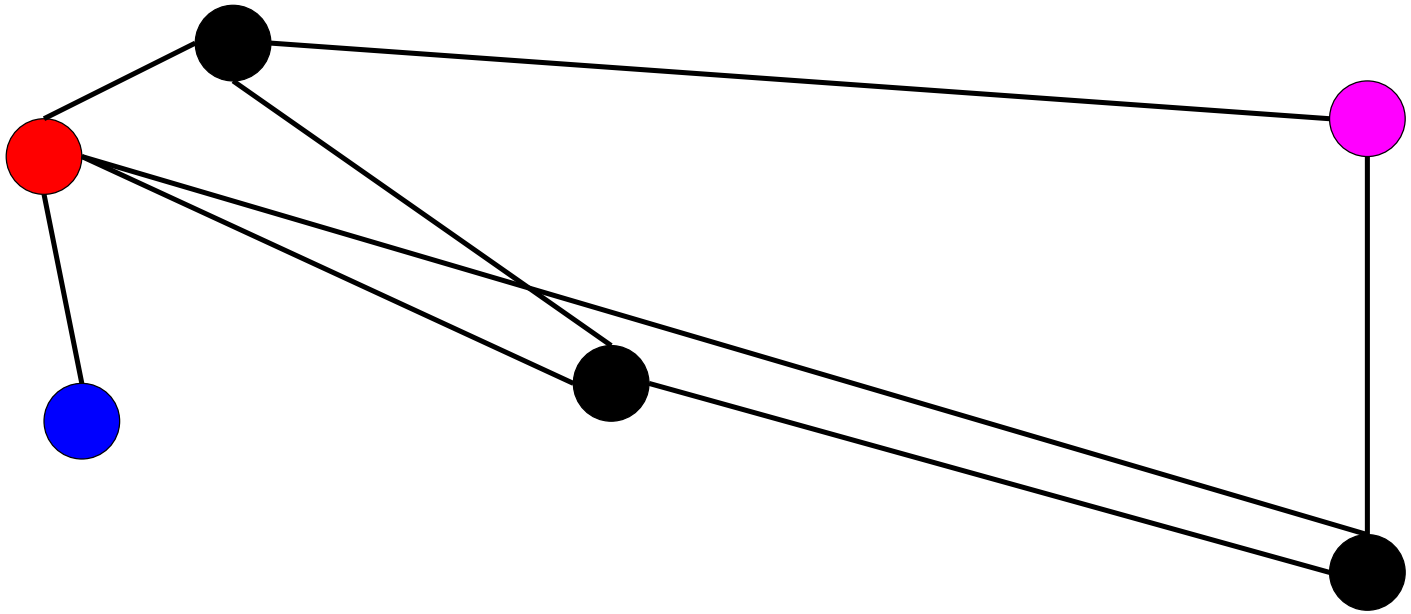


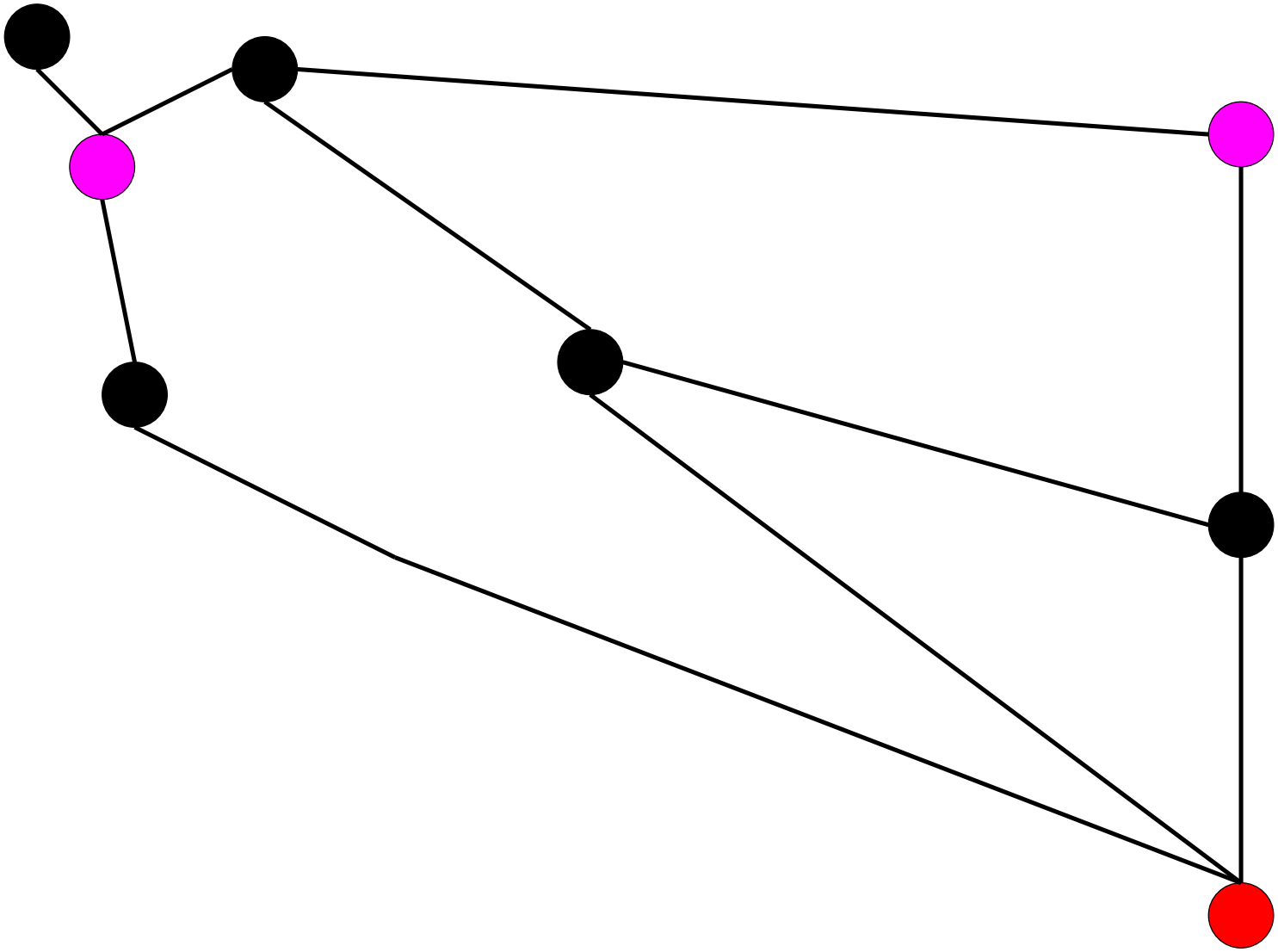


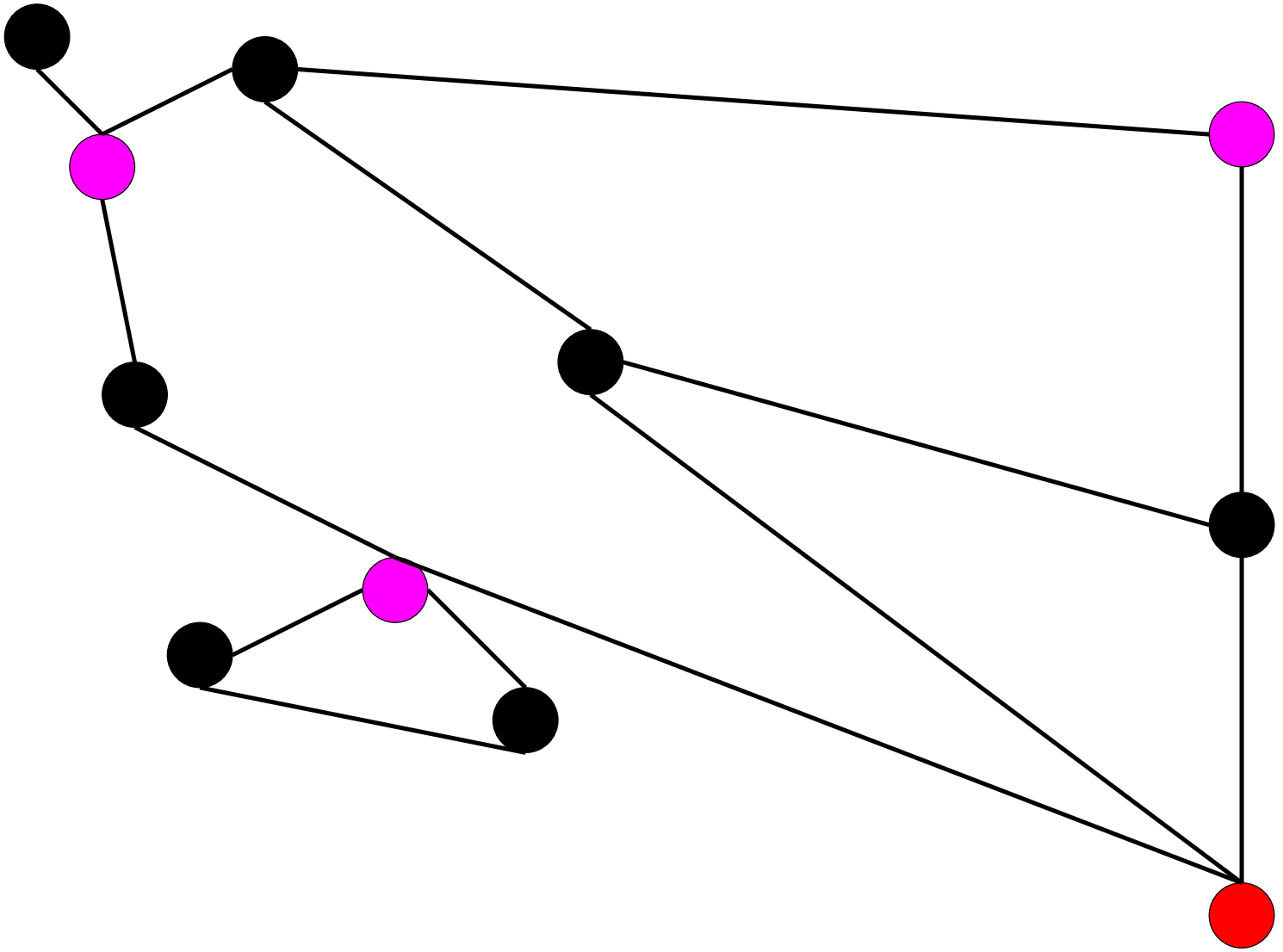


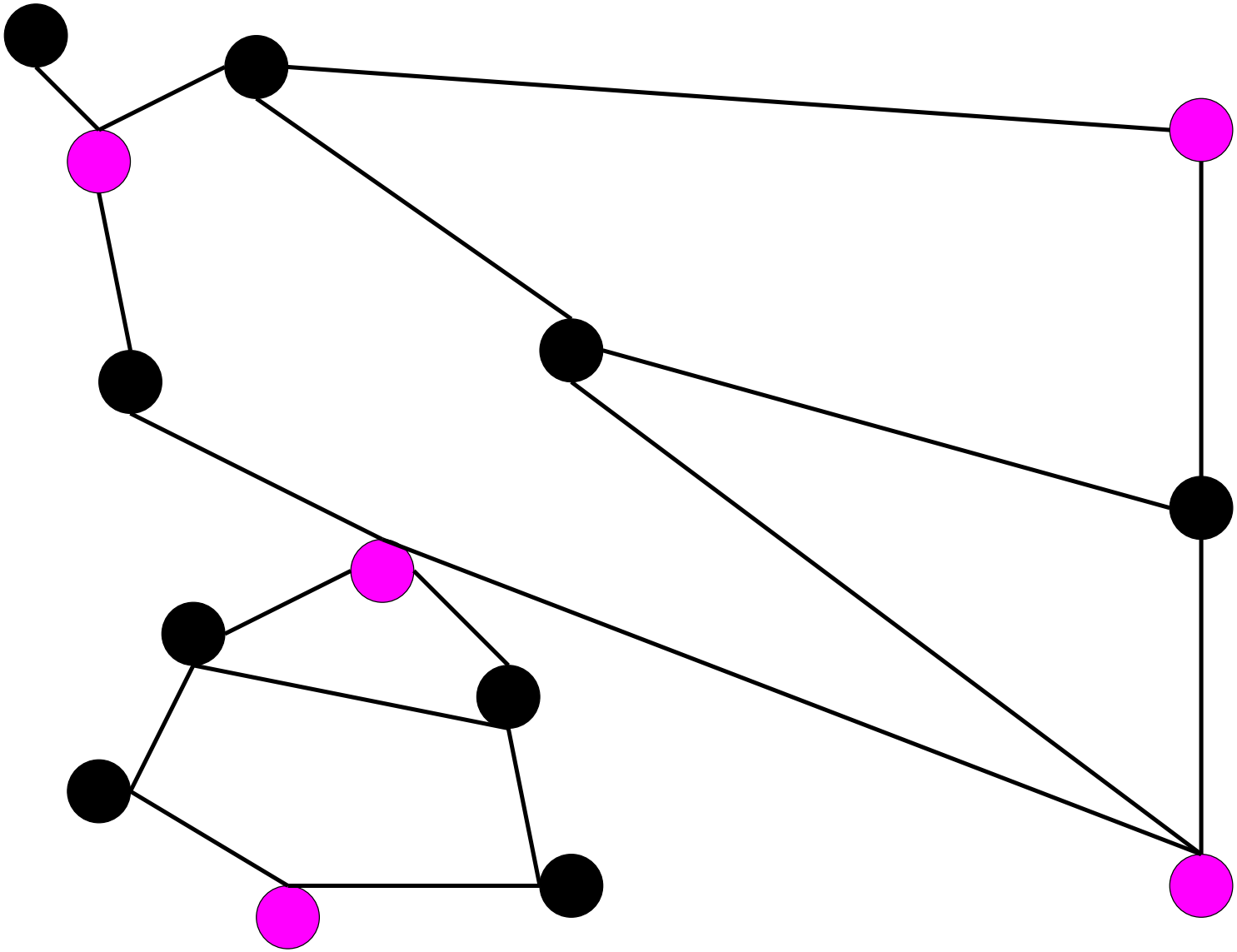


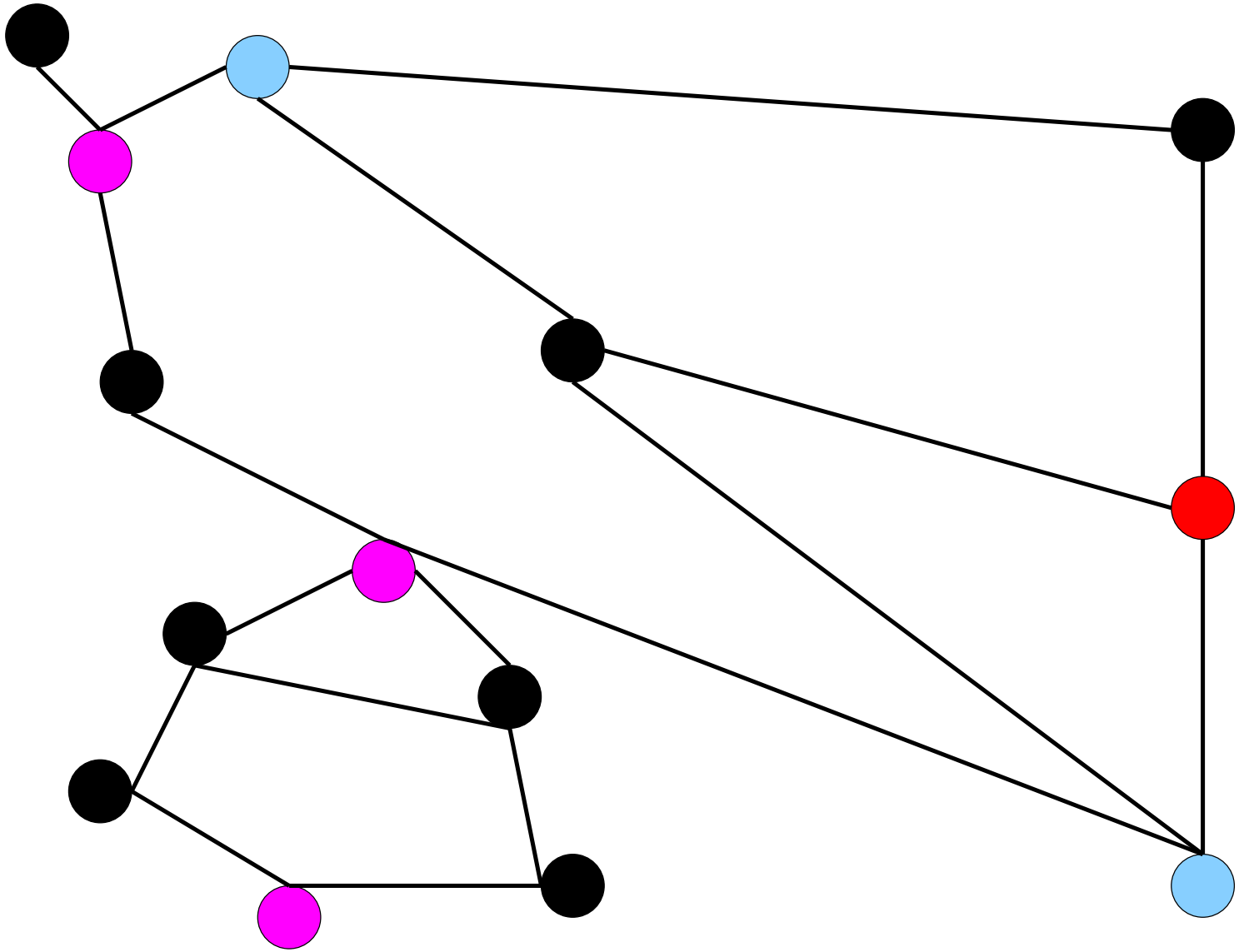












Die Stirling-Formel liefert:

Folgerung 3 *Nonblocker kann in Zeit* $\mathcal{O}^*(3.0701^{k_d})$ *gelöst werden.*

Hinweis: Die $\mathcal{O}^*$-Notation "vernachlässigt" sowohl konstante Faktoren als auch polynomielle Anteile und ist daher sehr geeignet für FPT.

Hinweis: Hiermit lässt sich eine (recht) lange als offen geltende Frage leicht beantworten:

Folgerung 4 *Es gibt einen Algorithmus zum Auffinden kleinster dominierender Mengen in einem Graphen mit* n *Knoten, dessen Laufzeit sich durch* $\mathcal{O}^*(c^n)$ *abschätzen lässt für ein* $c \in (1, 2)$ *unabhängig von* n*.*

Intermezzo

- exakte Exponentialzeitalgorithmen
- siehe: Buch von Fomin / Kratsch aus 2010
- Bei “Mengenauswahlproblemen” (wie DS oder VC) oft trivial: Algorithmus mit Laufzeit $\mathcal{O}^*(2^n)$, wobei n Anzahl der Elemente der Grundmenge (z.B. Knotenmenge): Probiere alle Teilmengen aus.
- Weit weniger trivial: Algorithmus, dessen Laufzeit sich durch $\mathcal{O}^*(c^n)$ abschätzen lässt für ein $c \in (1, 2)$ unabhängig von n
- alternative Sichtweise (gemäß dieser Vorlesung): Parameterisiere mit n.

Kurzer Ausflug: Parameterisierung nach der Knotenzahl

Es gibt einen Algorithmus zum Auffinden kleinster dominierender Mengen in einem Graphen mit n Knoten, dessen Laufzeit sich durch $\mathcal{O}^*(c^n)$ abschätzen lässt für ein $c \in (1,2)$ unabhängig von n.

Idee: Unterscheide zwei Fälle:
1. Es gibt keine Nonblocker-Menge der Größe mindestens $k_n > 1/2 \rightsquigarrow$
Suche in Zeit $\mathcal{O}^*(3.0701^{k_n})$ mit $3.0701^{k_n} < 2$; nach k, $1 \leq k \leq k_n$, sodass k die Größe der größtmöglichen Nonblocker-Menge ist.
2. Es gibt eine Nonblocker-Menge der Größe mindestens $k_n > 1/2$; ist k_n optimal? $\rightsquigarrow$

Probiere alle Mengen der Größe mindestens k_n als Nonblocker-Mengen aus, mit $\left(\begin{array}{c} n \\ k_n n \end{array} \right) < 2^n$.

Wähle hierbei k_n so, dass $3.0701^{k_n n} = \left(\begin{array}{c} n \\ k_n n \end{array} \right)$.

Aufgabe: Wie (genau) folgt aus unserem bereits abgeleiteten parameterisierten Algorithmus für VC ein Algorithmus zum Auffinden kleinster Knotenüberdeckungsmengen mit Laufzeit $\mathcal{O}^*(c^n)$ mit $c < 2$? Ermitteln Sie eine gute Laufzeitabschätzung für diesen Algorithmus.

Wie erhalte ich Kernreduktionen?

- Fokus auf Extremen (z. B. kleingradige / großgradige Knoten)
- Verwendung bekannter (nicht-trivialer) mathematischer Aussagen
- Dafür häufig wichtig: Betrachtung annotierter Probleme
- Speziell: Einsatz von Katalysatoren
- Verallgemeinerung bekannter einfacher Regeln

Eine weitere Reduktionsidee für VC: Kronen

Es sei $G = (V, E)$ ein Graph. Eine unabhängige Menge I von G heißt Krone gdw. für alle nicht-leere Mengen $U \subseteq N(I)$: $|U| \leq |N(U) \cap I|$.

Eine Krone I hat die folgende Eigenschaft (Heiratssatz von Hall):
Sei $G = (V_1, V_2; E)$ ein bipartiter Graph mit $|V_1| \leq |V_2|$.
G besitzt ein Matching der Größe $|V_1|$ gdw. für alle $U \subseteq V_1$ ist $|U| \leq |N(U)|$.).
$H := N(I)$ wird in I (gepaart) gematched. (H heißt auch Kopf der Krone.)

Reduktionsregel 6 *Ist* $(G = (V, E), k)$ *eine VC Instanz und* $I \subseteq V$ *eine Krone mit Kopf* $H = N(I)$*, so reduziere zu* $(G - (I \cup H), k - |H|)$.

Problem: Wie findet man (große) Kronen?

Standarddefinition für Kronen

Eine Kronenzerlegung eines Graphen $G = (V, E)$ ist ein Tripel (I, H, X) mit $I \cup H \cup X = V$, $I \cap H = \emptyset$, $I \cap X = \emptyset$, $H \cap X = \emptyset$, sodass

1. I ist eine unabhängige Menge in G.
2. H ist ein Separator (zwischen I und X).
3. H wird in I gepaart durch eine injektive Abbildung $\mu : H \to I$.

Dann ist I eine Krone in der Definition der vorigen Folie.

Die Kronenreduktionsregel sagt aus:
Ist (I, H, X) Kronenzerlegung von G, so kann man sich zur Berechnung (kleinster!) Knotenüberdeckungen auf die Betrachtung von $G[X]$ zurückziehen.

Lemma 5 *Alg. 3 kombiniert mit Regel 6, angesetzt auf* (G, k)*, liefert entweder korrekt NEIN oder liefert eine reduzierte Instanz* (G', k') *von VC mit* $|V(G')| \leq 3k$*.*

Beobachte hierzu:

- Finden wir ein Matching M (d.h., M_1 oder M_2) in G mit $|M| > k$, so ist (G, k) eine $\times$-Instanz wegen des Lemmas von König und Egerváry.

- Enthalten weder M_1 noch M_2 mehr als k Kanten, so hat $G[M_1 \cup M_2]$ höchstens $3k$ Knoten.

Algorithm 2 An efficient algorithm to find a crown in a graph, called GETCROWN

Input(s): an graph G

Output(s): a crown $I \subseteq V(G)$

Greedily find a maximal matching M_1 of G.
Let O (outsiders) be the set of vertices from G that are not in the matching.
if $O \neq \emptyset$ **then**
  Let $I' := \emptyset$.
  Find a maximum matching M_2 in the bipartite graph B with vertex set $O \cup N(O)$ and edge set $E_B = \{uv \mid u \in O \wedge v \in N(O)\}$.
  Let I collect all vertices in O that are not matched by M_2.
  while $I' \neq I$ **do**
    Let $I' := I$. Let $H := N(I)$.
    Let $I := I \cup \{u \in O \mid \exists v \in H(uv \in M_2)\}$.
  end while
else
  $I := \emptyset$.
end if

Algorithm 3 Using crowns for kernels in the case of VC

Input(s): an instance (G, k) of VC

Output(s): $\times$ if the VC instance has no solution OR a crown $I \subseteq V(G)$ such that $|V(G - N[I])| \leq 3k$

Greedily find a maximal matching M_1 of G.
if $|M_1| > k$ **then**
    return $\times$
else
    Let O (outsiders) be the set of vertices from G that are not in M_1.
    ...
    return I
end if

Notation: Größe einer kleinstmöglichen Knotenüberdeckung von G: $vc(G)$

Zur Korrektheit der Kronenregel

Ist I Krone in $G = (V, E)$ mit Kopf $H = N(I)$, so gilt:

$$vc(G) = vc(G - N[I]) + |H|.$$

Da I Krone, ist H kleinstmögliche Knotenüberdeckung von $G[N[I]]$.
Ist C^* kleinstmögliche Knotenüberdeckung (MVC) von G, so ist $C^* \cap N[I]$ VC von $G[N[I]]$ und $C^* \setminus N[I]$ ist VC von $G - N[I]$.

$$\rightsquigarrow vc(G) = |C^*| = |C^* \setminus N[I]| + |C^* \cap N[I]| \geq vc(G') + |H|.$$

Ist C' MVC on G', setze $\tilde{C} = C' \cup H$. $\tilde{C}$ ist VC von G.

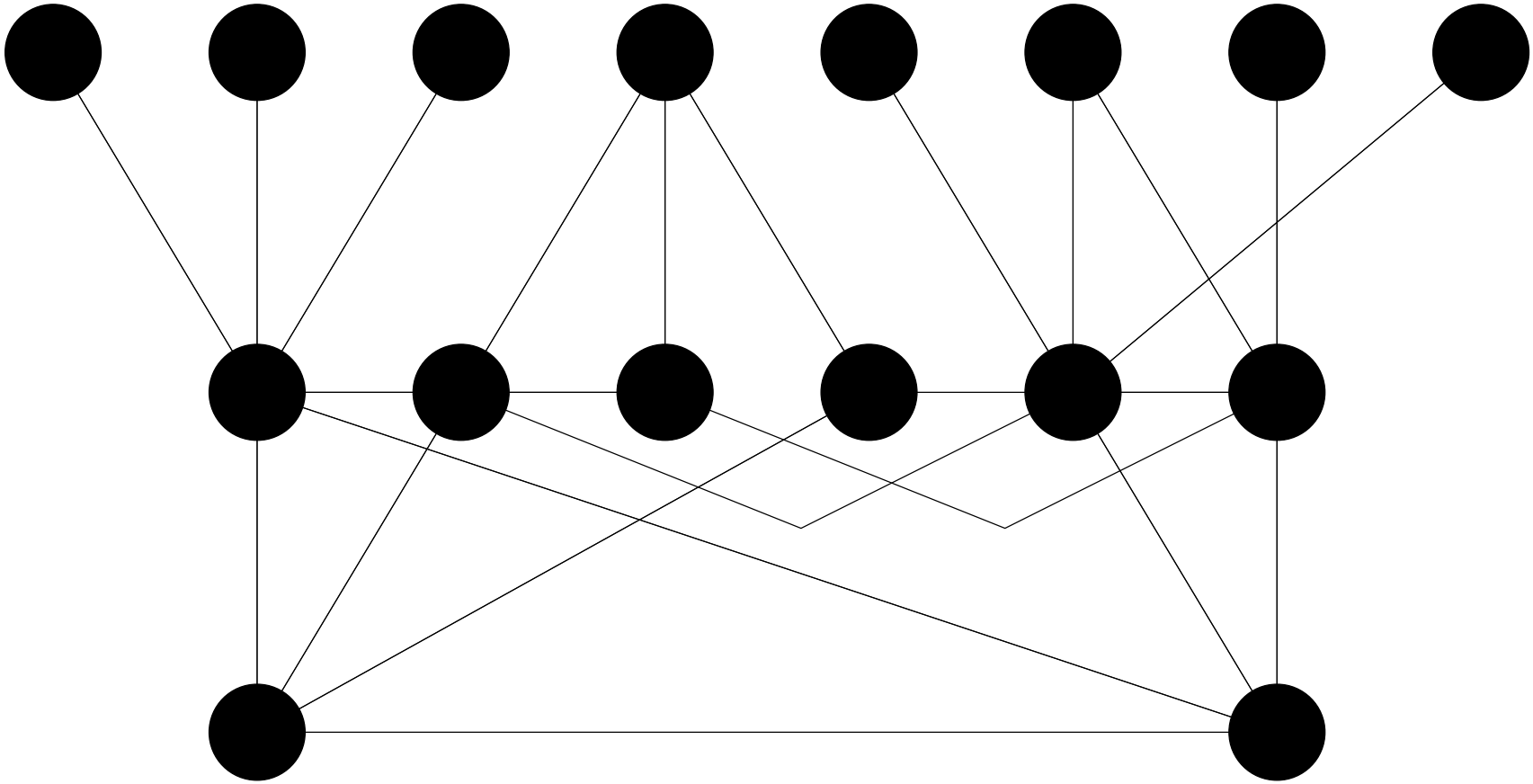
$$\rightsquigarrow vc(G) \leq |\tilde{C}| = vc(G') + |H|.$$

Bem.: “O.E.” $\tilde{C}$ statt C^*, also o.E.: $H \subseteq C^*$.

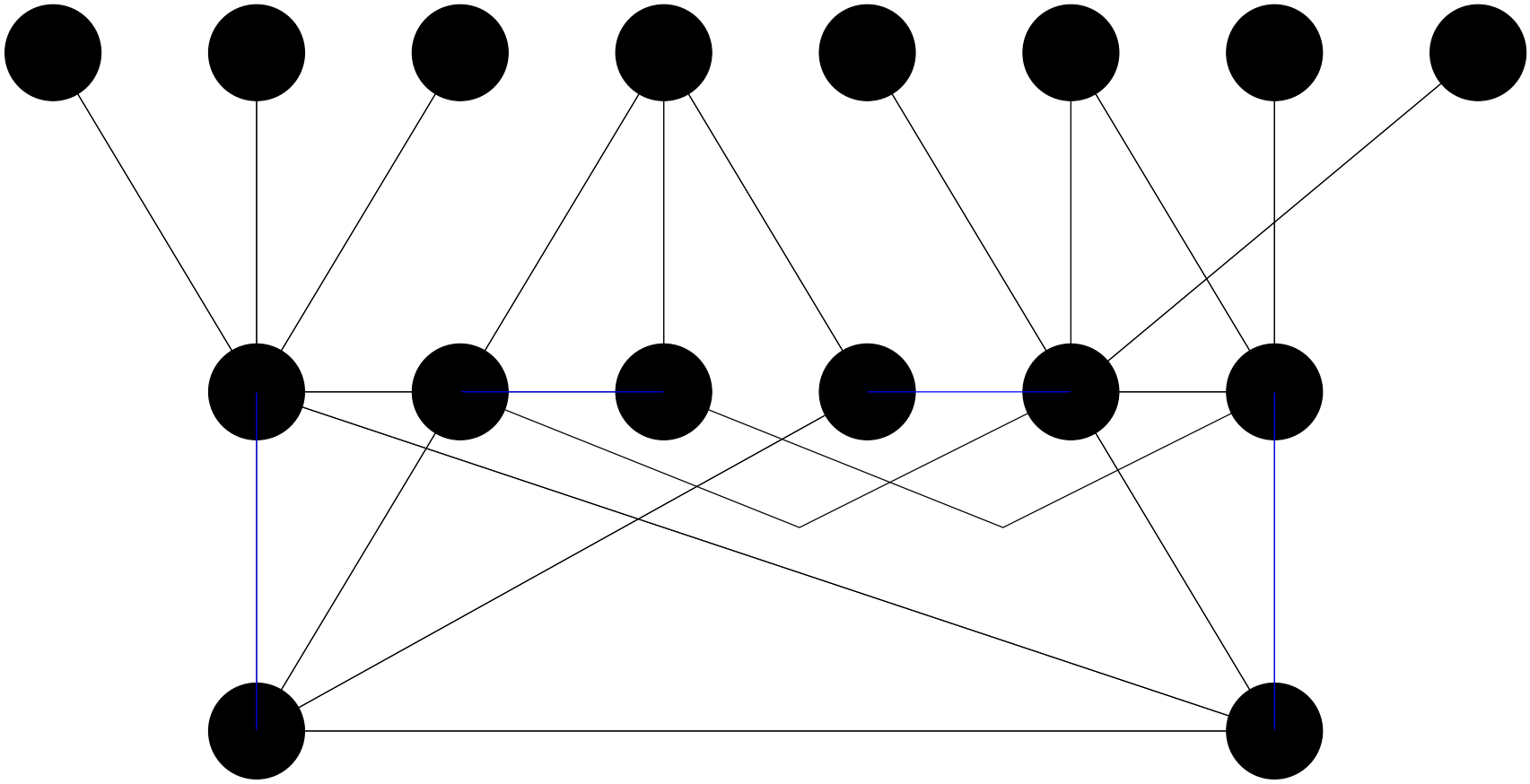
Bemerkungen zum Kronenalgorithmus (Alternative ?!), falls keine $\times$-Instanz:

O.E.: Graph G hat keine isolierten Knoten.
Maximales Matching M_1 erfüllt $|M_1| \leq k$.
$O = V \setminus V(M_1)$ ist unabhängige Menge.
Gilt $|O| \leq k$, so haben wir Kern: $|V| = 2|M_1| + |O| \leq 3k$.
Maximum Matching M_2 in bipartitem Graph $B = (O, N(O), E_B)$ erfüllt $|M_2| \leq k$.
Gilt $|M_2| = |N(O)|$, so ist O Krone in G.
$\rightsquigarrow$ "Neustart" nach Anwendung der Kronenregel.
Im übrigen Fall: $|M_2| \leq k$ und $|M_2| < |N(O)|$.
$I = O \setminus V(M_2)$ ist unabhängige Menge, nicht-leer wegen $|O| > k$.
Wähle $x \in I$ beliebig und konstruiere von x startenden M_2-alternierenden Pfad P;
P induziert in B (somit in G) eine Krone.
$\rightsquigarrow$ "Neustart" nach Anwendung der Kronenregel.

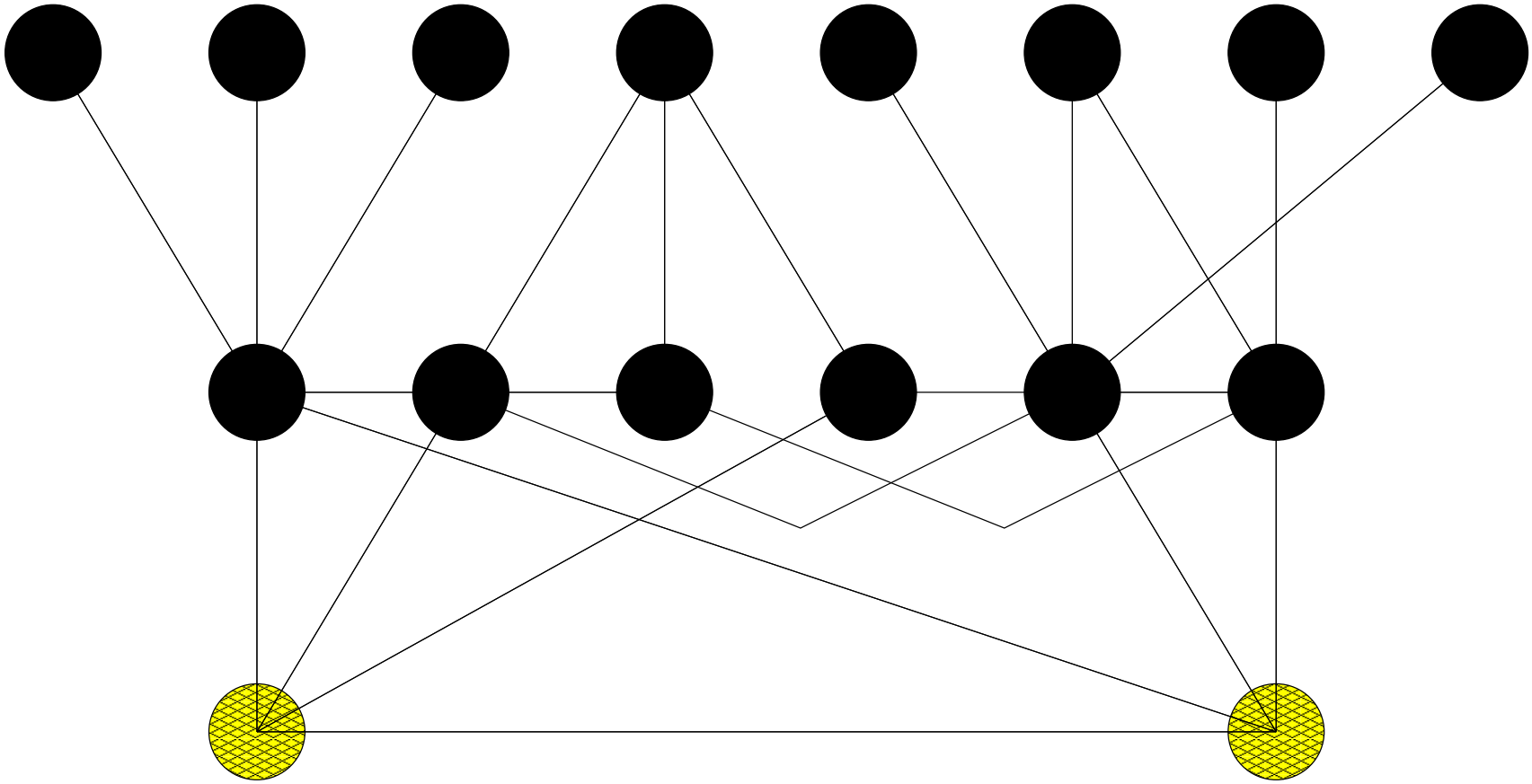
Die Kronenregel an einem Beispiel



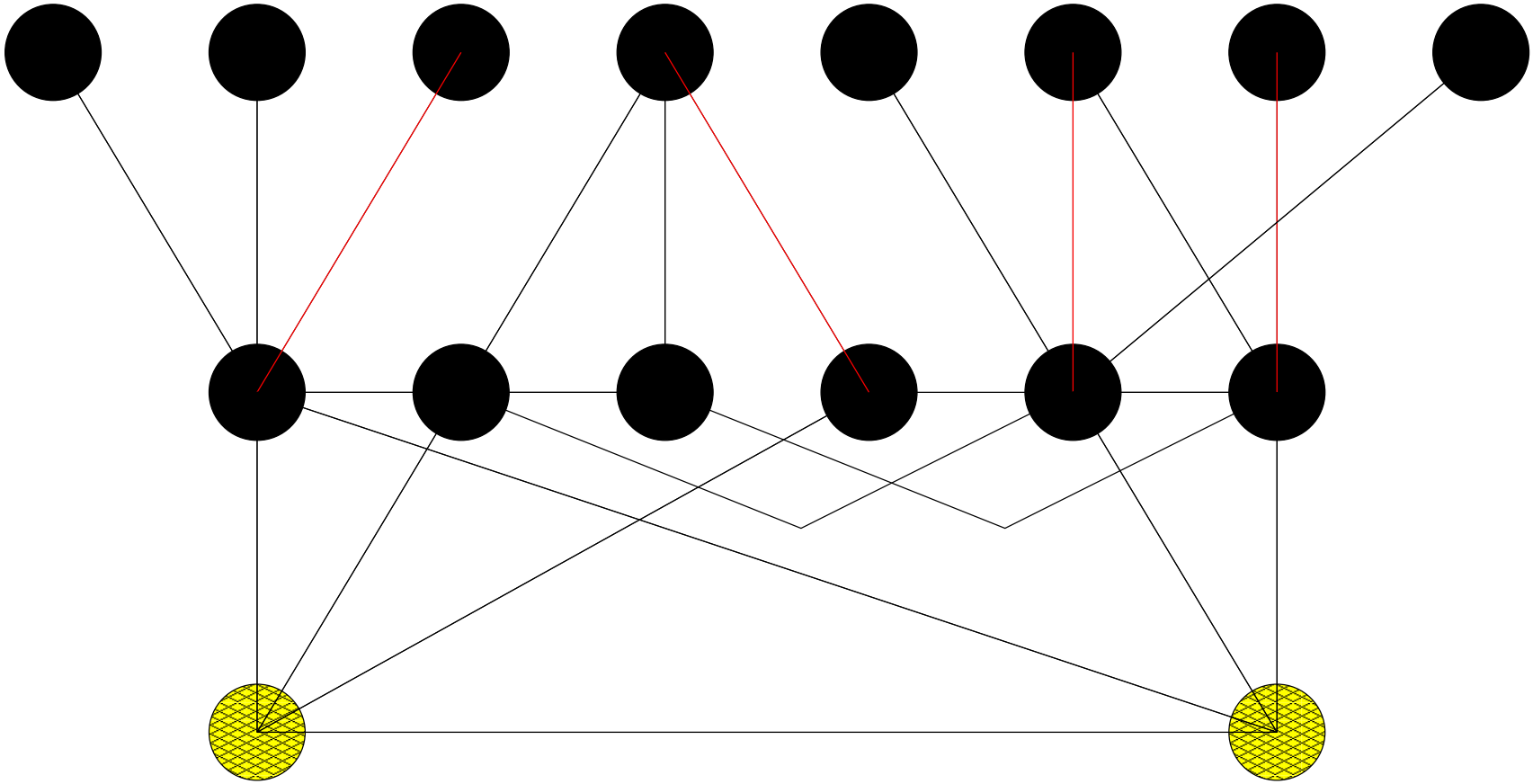
Die Kronenregel an einem Beispiel: Ein maximales Matching M_1



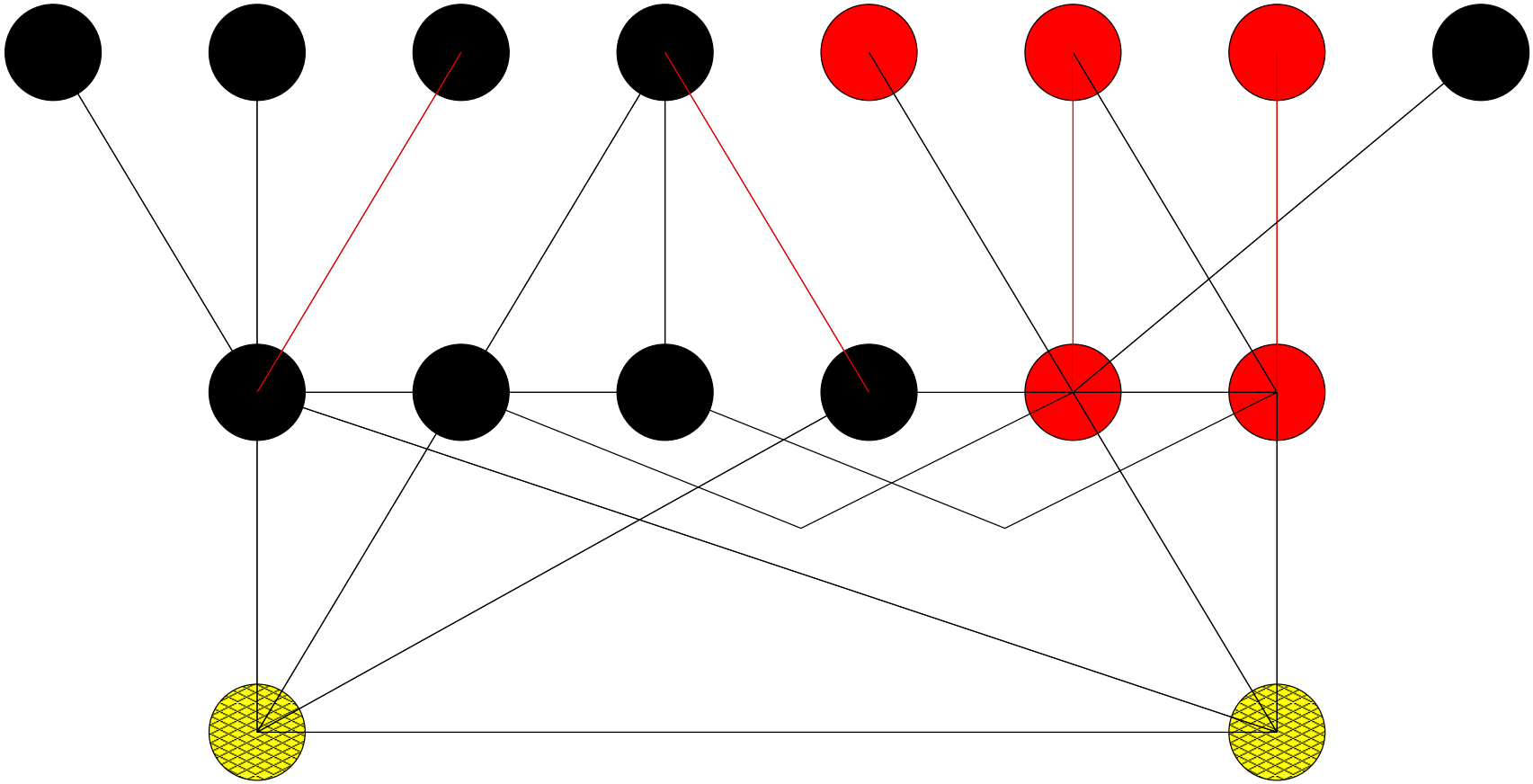
Die Kronenregel an einem Beispiel: $O \cup N(O)$ ist nicht gelb



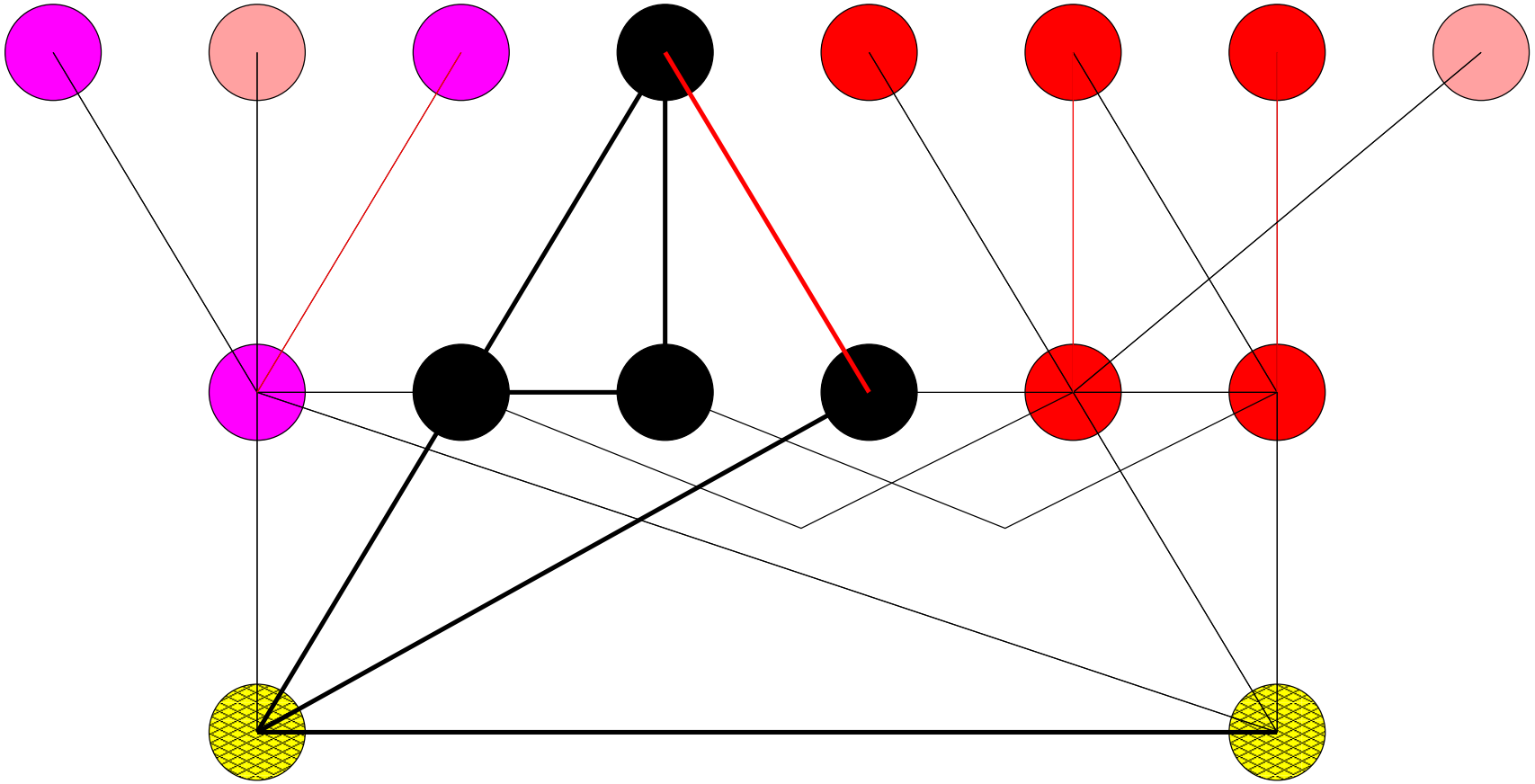
Die Kronenregel an einem Beispiel: Ein rotes Maximum Matching



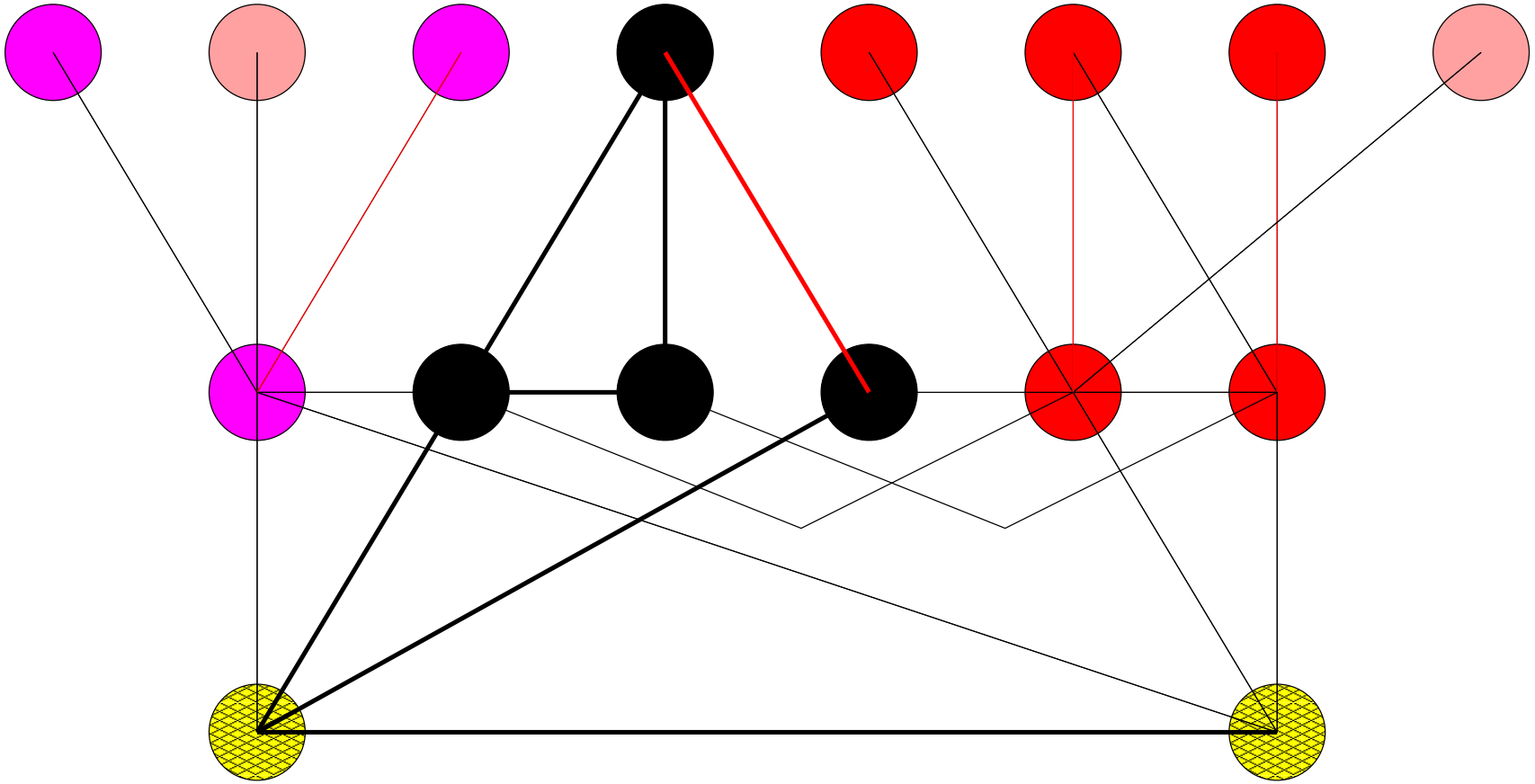
Die Kronenregel an einem Beispiel: Die roten Knoten sind eine Krone



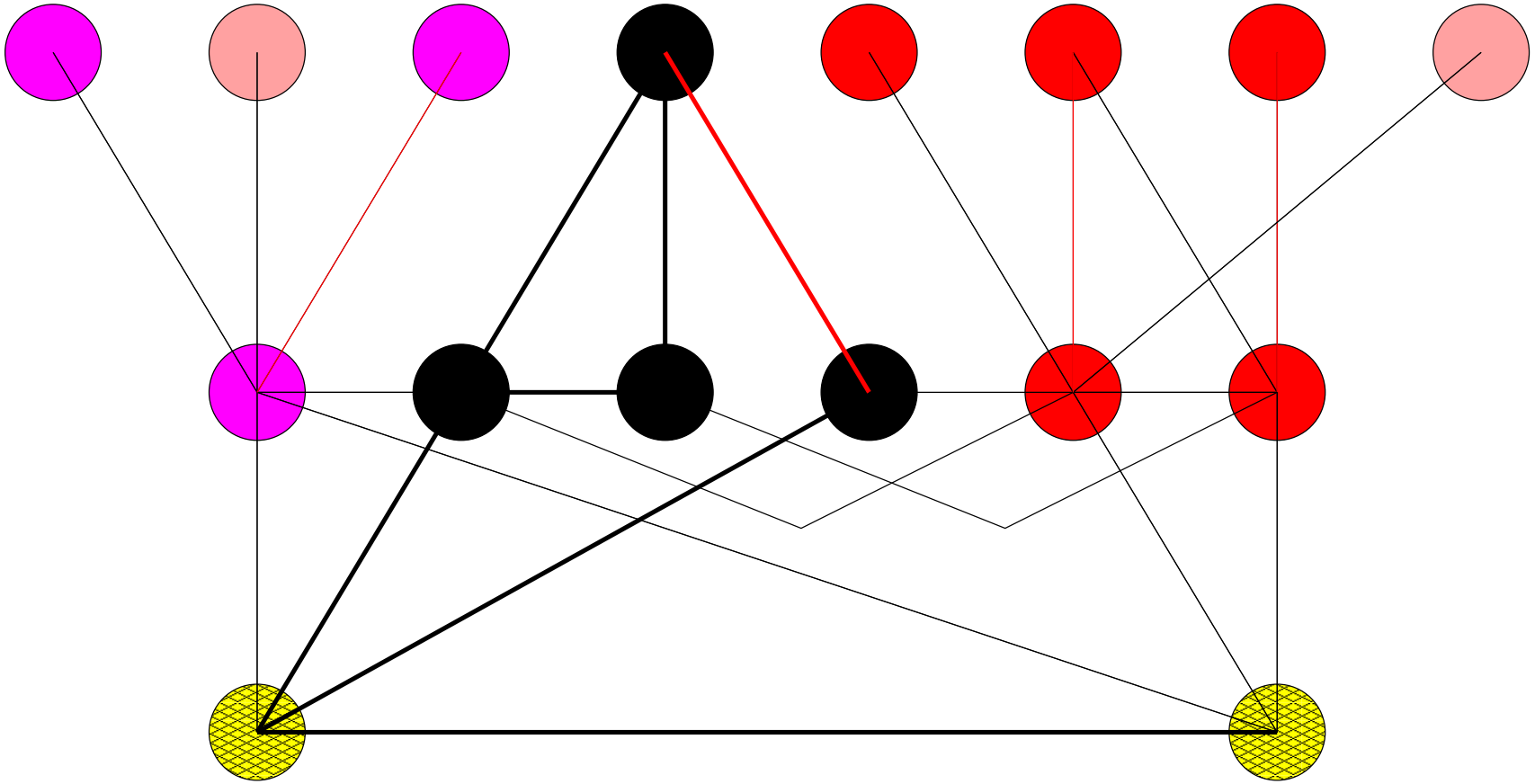
Die Kronenregel an einem Beispiel: Die magenta Knoten sind auch eine Krone



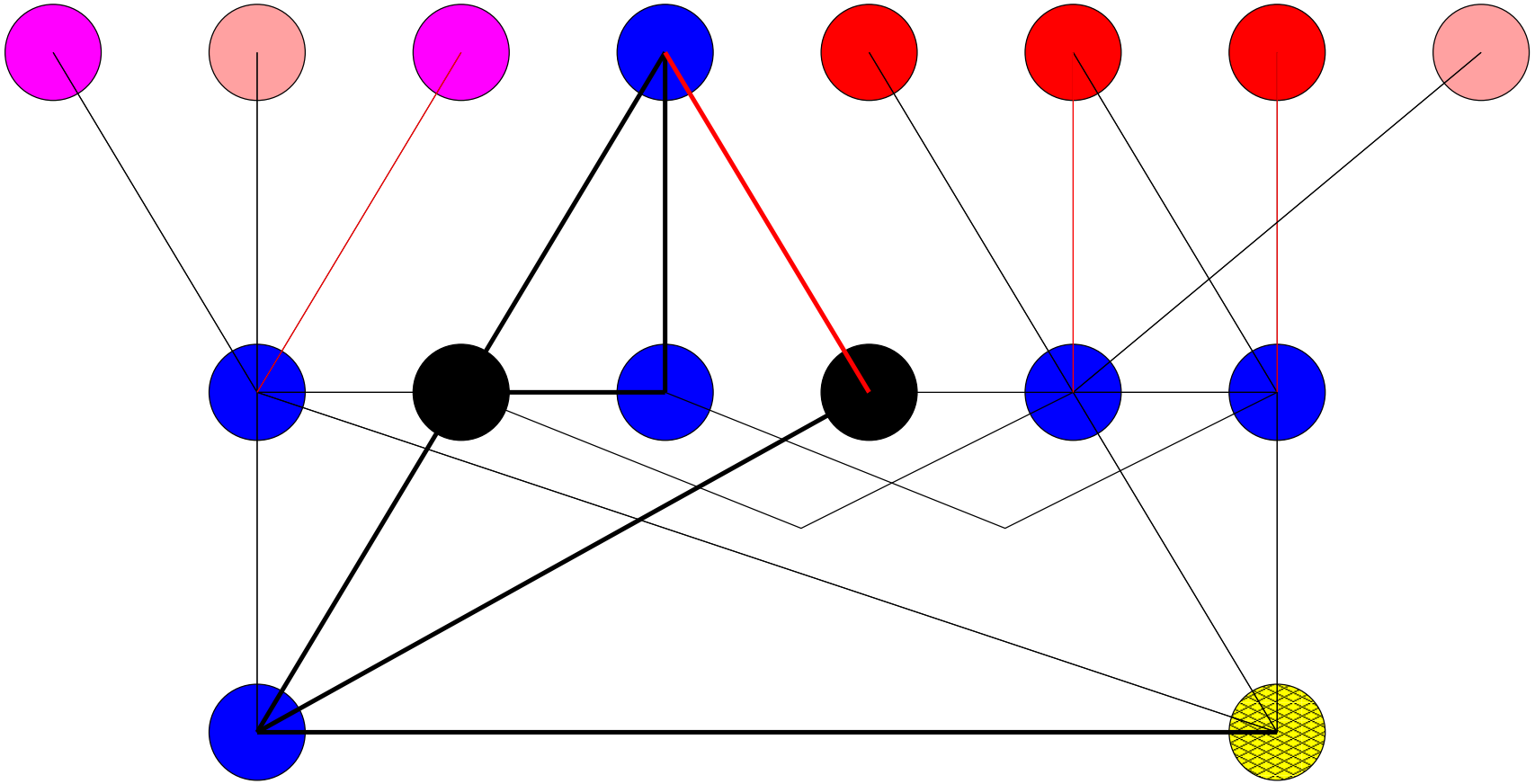
Die Kronenregel an einem Beispiel: Die rosa Knoten werden entfernt (isoliert)



Kern: fette Kanten; Achtung: Regeln werden nochmals ausgelöst



Die blauen Knoten bilden kleinstmögliche Knotenüberdeckung (Kronenregeln)



Abschließende Diskussion: Welcher Kern ist der beste (für VC)?

- Kriterium Kerngröße: Nemhauser-Trotter
- Von der Laufzeit: Buss-Regel, dann Kronenregel
- Denn: Aufsuchen maximaler Matchings in Linearzeit, (bipartite) Maximum Matchings viel “teurer”
- daher praktischerweise:
  Wende zunächst Buss-Regel an, kombiniert mit Regeln für isolierte Knoten und Blätter;
  Dann Suche nach Kronen (evtl. entstehen so auch wieder großgradige Knoten ...)
  Schließlich NT-Regel (auch hier “Restart” möglich)
- Faustregel:
  Solange das Problem in Polynomzeit verkleinert werden kann, tue es!