Karlsruher Institut für Technologie

Algorithmen für Planare Graphen – Übung 7

Laura Merker | 28. Juli 2022

Übungsblatt 7

- Kreise
- Spezialfall von s-t-Pfaden
- Right-First Tiefensuche und Left-First Breitensuche

Kreise

- G planar mit Einbettung, Menge der Facetten F, äußere Facette f_0
- Menge E_i aller Kanten, bei denen beide inzidenten Facetten Entfernung i zur äußeren Facette haben
- Menge $E_{i,i+1}$ aller Kanten, bei denen die angrenzenden Facetten Entfernung i bzw. $i + 1$ haben

(vgl. Entfernung von Rechtskreisen Vorlesung 14, Folie 5)

Kreise

Karlsruher Institut für Technologie

- G planar mit Einbettung, Menge der Facetten F, äußere Facette f_0
- Menge E_i aller Kanten, bei denen beide inzidenten Facetten Entfernung i zur äußeren Facette haben
- Menge $E_{i,i+1}$ aller Kanten, bei denen die angrenzenden Facetten Entfernung i bzw. $i+1$ haben

(vgl. Entfernung von Rechtskreisen Vorlesung 14, Folie 5)

- $E_1, \ldots, E_d, E_{1,2}, \ldots, E_{d-1,d}$ partitionieren $E(G)$

Kreise

Karlsruher Institut für Technologie

- G planar mit Einbettung, Menge der Facetten F, äußere Facette f_0
- Menge E_i aller Kanten, bei denen beide inzidenten Facetten Entfernung i zur äußeren Facette haben
- Menge $E_{i,i+1}$ aller Kanten, bei denen die angrenzenden Facetten Entfernung i bzw. $i+1$ haben

(vgl. Entfernung von Rechtskreisen Vorlesung 14, Folie 5)

- $E_1, \dots, E_d, E_{1,2}, \dots, E_{d-1,d}$ partitionieren $E(G)$
  $\rightarrow$ Kante mit angrenzenden Facetten der Entfernung i und j mit $|i-j| > 1$ widerspricht Def. Entfernung
- Jedes $E_{i,i+1}$ induziert kanten-disjunkte Vereinigung von einfachen Kreisen (knoten-disjunkt?)

Kreise

- G planar mit Einbettung, Menge der Facetten F, äußere Facette f_0
- Menge E_i aller Kanten, bei denen beide inzidenten Facetten Entfernung i zur äußeren Facette haben
- Menge $E_{i,i+1}$ aller Kanten, bei denen die angrenzenden Facetten Entfernung i bzw. $i+1$ haben

(vgl. Entfernung von Rechtskreisen Vorlesung 14, Folie 5)

- $E_1, \ldots, E_d, E_{1,2}, \ldots, E_{d-1,d}$ partitionieren $E(G)$
  $\rightarrow$ Kante mit angrenzenden Facetten der Entfernung i und j mit $|i-j| > 1$ widerspricht Def. Entfernung
- Jedes $E_{i,i+1}$ induziert kanten-disjunkte Vereinigung von einfachen Kreisen (knoten-disjunkt?)
  $\rightarrow$ Betrachte Facetten mit Entfernung $\geqslant i+1$ und deren Zusammenhangskomponenten in G^*
- Bestimme $E_1, \ldots, E_d$ und $E_{1,2}, \ldots E_{d-1,d}$ in Linearzeit

Kreise

- G planar mit Einbettung, Menge der Facetten F, äußere Facette f_0
- Menge E_i aller Kanten, bei denen beide inzidenten Facetten Entfernung i zur äußeren Facette haben
- Menge $E_{i,i+1}$ aller Kanten, bei denen die angrenzenden Facetten Entfernung i bzw. $i + 1$ haben

(vgl. Entfernung von Rechtskreisen Vorlesung 14, Folie 5)

- $E_1, \dots, E_d, E_{1,2}, \dots, E_{d-1,d}$ partitionieren $E(G)$
  $\rightarrow$ Kante mit angrenzenden Facetten der Entfernung i und j mit $|i - j| > 1$ widerspricht Def. Entfernung
- Jedes $E_{i,i+1}$ induziert kanten-disjunkte Vereinigung von einfachen Kreisen (knoten-disjunkt?)
  $\rightarrow$ Betrachte Facetten mit Entfernung $\geqslant i + 1$ und deren Zusammenhangskomponenten in G^*
- Bestimme $E_1, \dots, E_d$ und $E_{1,2}, \dots E_{d-1,d}$ in Linearzeit
  $\rightarrow$ Berechne Entfernung mit Breitensuche in G^* von f_0 aus

Spezialfall von s-t-Pfaden

Gegeben: G planar mit Einbettung, s, t an äußerer Facette

Gesucht: maximale Anzahl von paarweise kanten-disjunkten s-t-Pfaden

Wiederholung: Flüsse

Definition: Flussnetzwerk $(D = (V, A), c : A \to \mathbb{R}_{>0}, s \in V, t \in V)$

- $uv \in A \iff vu \in A$ und $c(uv) = c(vu)$ für alle $uv \in A$
- c ordnet Kanten Kapazitäten zu
- s ist Quelle, t ist Senke

Defintion: s-t-Fluss $\Phi\colon A \to \mathbb{R}$

- Flusserhaltung: $\sum_{uv \in A} \Phi(uv) = 0$ für jeden Knoten $u \neq s, t$.
- Zulässigkeit: $\Phi(uv) \leqslant c(uv)$ für alle Kanten $uv \in A$.
- Antisymmetrie: $\Phi(uv) = -\Phi(vu)$ für alle Kanten $uv \in A$.

Wir maximieren: $\Phi(s) = \sum_{sv \in A} \Phi(sv) = -\Phi(t)$

Spezialfall von *s*-*t*-Pfaden

Gegeben: G planar mit Einbettung, s, t an äußerer Facette

Gesucht: maximale Anzahl von paarweise kanten-disjunkten s-t-Pfaden

Spezialfall von s-t-Pfaden

Gegeben: G planar mit Einbettung, s, t an äußerer Facette

Gesucht: maximale Anzahl von paarweise kanten-disjunkten s-t-Pfaden

Mögliche Ansätze:

- Algorithmus für das Menger-Problem in planaren Graphen
- Berechne maximalen s-t-Fluss, wobei $c(e) = 1$ für alle $e \in E(G)$
- Right-First Tiefensuche (vgl. Vorlesung 14, Folie 8)
  - lösche besuchte Kanten
  - wenn t erreicht wird, enthält Stack einen s-t-Pfad
  - starte dann wieder bei s
  - Korrektheit: Betrachte maximale Menge von (o.B.d.A. sich nicht kreuzenden) s-t-Pfaden von rechts nach links, in jeder Iteration finden wir den gesuchten Pfad oder einen anderen Pfad rechts davon

Right-First-DFS

Algorithmus 1: Right-First-DFS beginnend bei $u \to v$

Füge neuen Knoten s und orientierte Kante (s, u) im Gegenuhrzeigersinn vor e an u ein
Lege (s, u) auf einen Stack
Solange *Stack nicht leer* **wiederhole**
  Betrachte oberste Kante (x, y)
  Wenn *y inzident zu nichtorientierter Kante*
    Orientiere im Gegenuhrzeigersinn bzgl. y nächste nichtorientierte Kante $y \to w$ und lege diese auf den Stack
  sonst
    Entferne (x, y) vom Stack

Right-First-DFS

Algorithmus 1: Right-First-DFS beginnend bei $u \to v$

Füge neuen Knoten s und orientierte Kante (s, u) im Gegenuhrzeigersinn vor e an u ein
Lege (s, u) auf einen Stack
Solange *Stack nicht leer* **wiederhole**
    Betrachte oberste Kante (x, y)
    Wenn y *inzident zu nichtorientierter Kante*
        Orientiere im Gegenuhrzeigersinn bzgl. y nächste nichtorientierte Kante $y \to w$ und lege diese auf den Stack
    sonst
        Entferne (x, y) vom Stack

- Wofür brauchen wir die Kante (s, u)?

Right-First-DFS

Algorithmus 1: Right-First-DFS beginnend bei $u \to v$

Füge neuen Knoten s und orientierte Kante (s, u) im Gegenuhrzeigersinn vor e an u ein
Lege (s, u) auf einen Stack
Solange *Stack nicht leer* **wiederhole**
  Betrachte oberste Kante (x, y)
  Wenn y *inzident zu nichtorientierter Kante*
    Orientiere im Gegenuhrzeigersinn bzgl. y nächste nichtorientierte Kante $y \to w$ und lege diese auf den Stack
  sonst
    Entferne (x, y) vom Stack

- Wofür brauchen wir die Kante (s, u)?
  $\to$ sorgt dafür, dass der ganze Graph traversiert wird, falls $u \to v$ eine Brücke ist

Left-First-BFS

Algorithmus 2: Left-First-BFS beginnend bei e^* von rechts nach links bzgl. $u \rightarrow v$

Orientiere alle zu u inzidenten Kanten $u \rightarrow w$ und füge diese, beginnend bei e, im Uhrzeigersinn bzgl. u in eine Queue ein

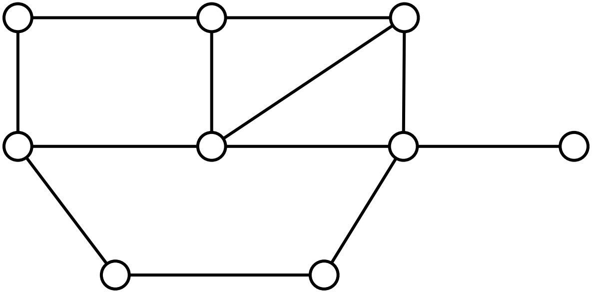
Solange *Queue nicht leer* **wiederhole**

- Betrachte erste Kante (x, y)
- **Für** *alle nichtorientierten Kanten yw im Uhrzeigersinn bzgl. y*
  - Orientiere $y \rightarrow w$ und füge (y, w) in die Queue ein
- Entferne (x, y) aus der Queue

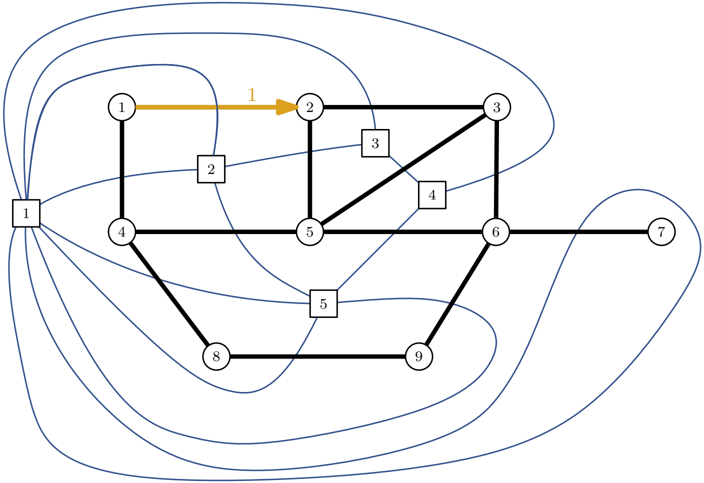
Duale Suche

- Right-First-Tiefensuche in G ausgehend von Kante e: $R = (e = e_1, \dots, e_m)$
- Left-First-Breitensuche in G^*, beginnend bei Kante e^*: $R^* = (e^* = e_1^*, \dots, e_m^*)$.

Die Reihenfolgen R und R^* heißen dual, falls e_i Dualkante zu e_i^* für alle $1 \leqslant i \leqslant m$.
Bestimme Reihenfolge R und $R^\star$ für:



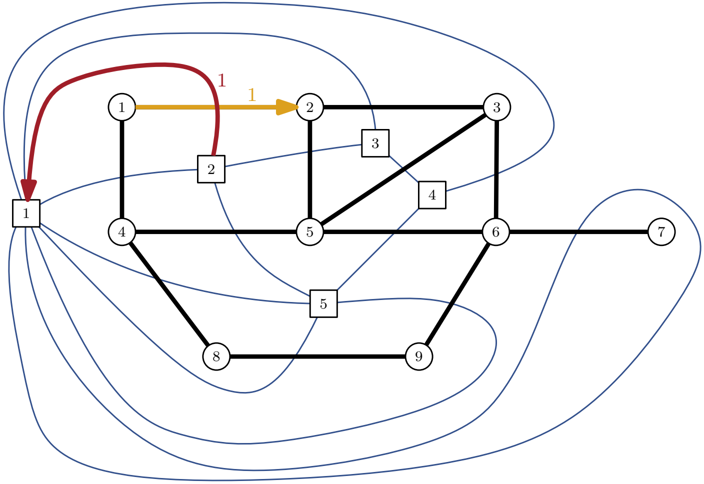
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

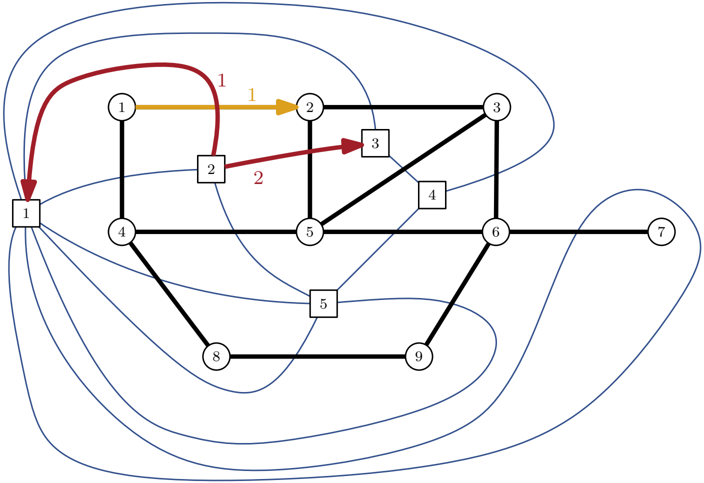
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

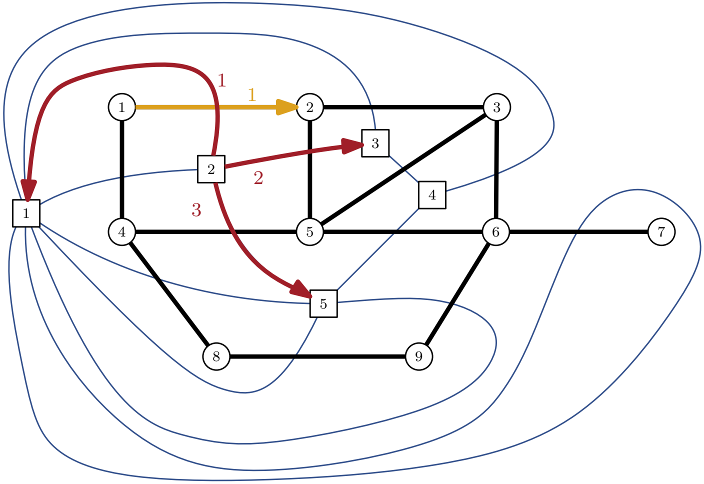
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

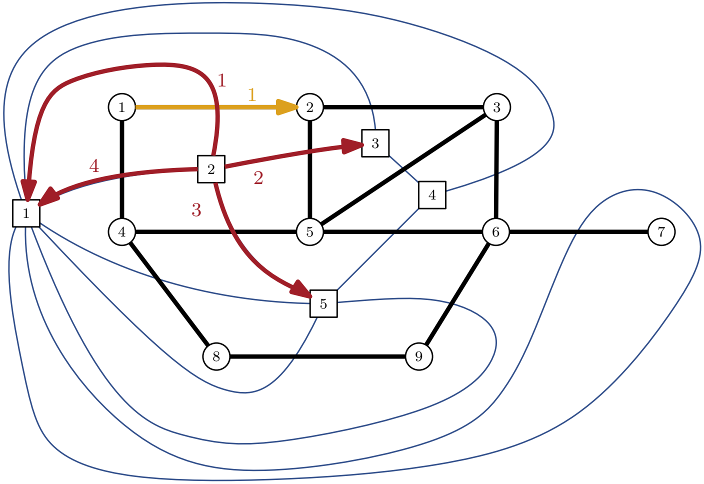
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

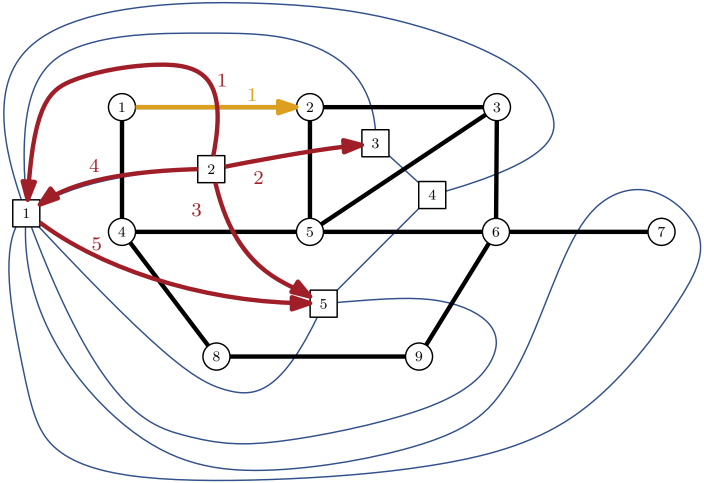
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

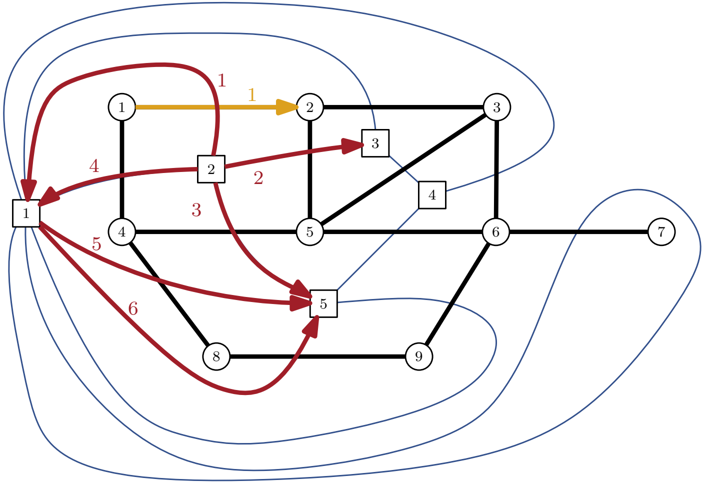
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

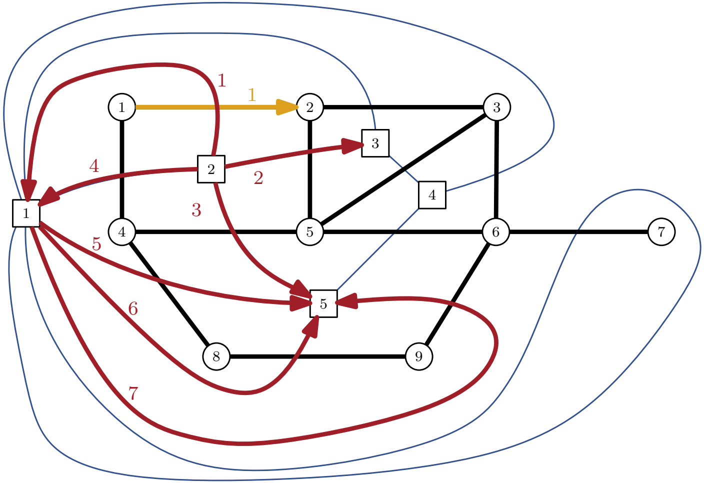
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

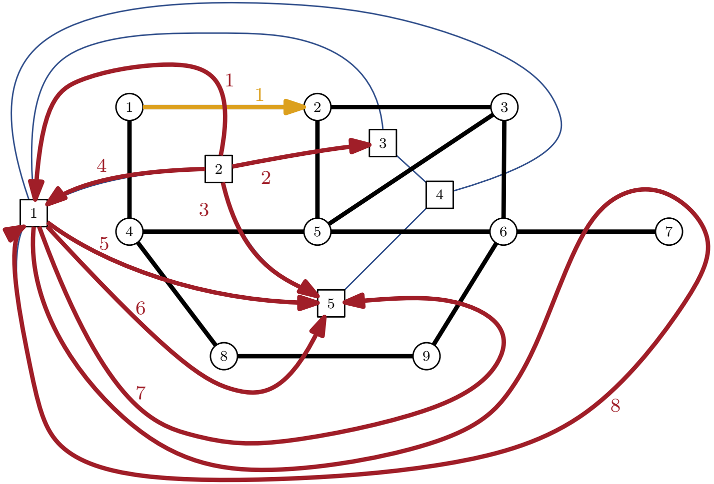
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

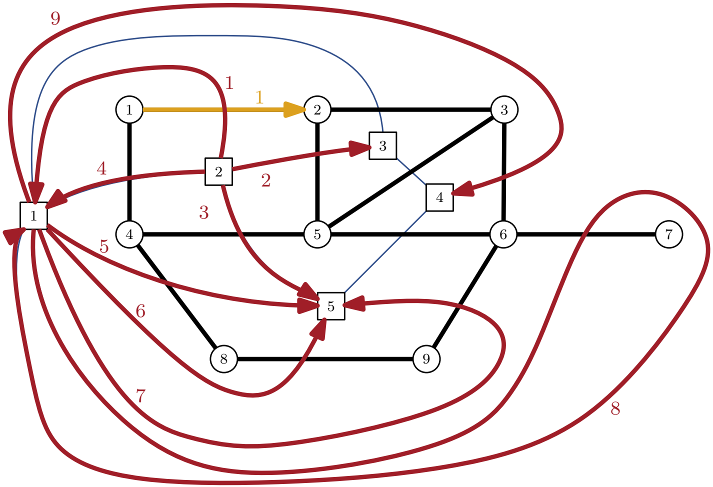
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

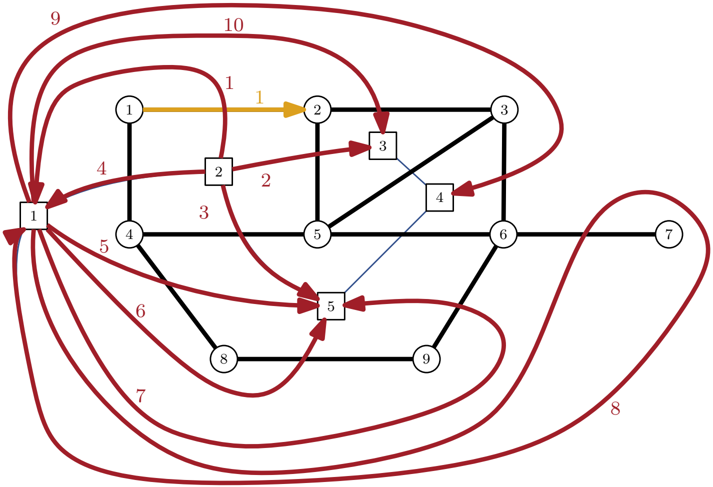
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

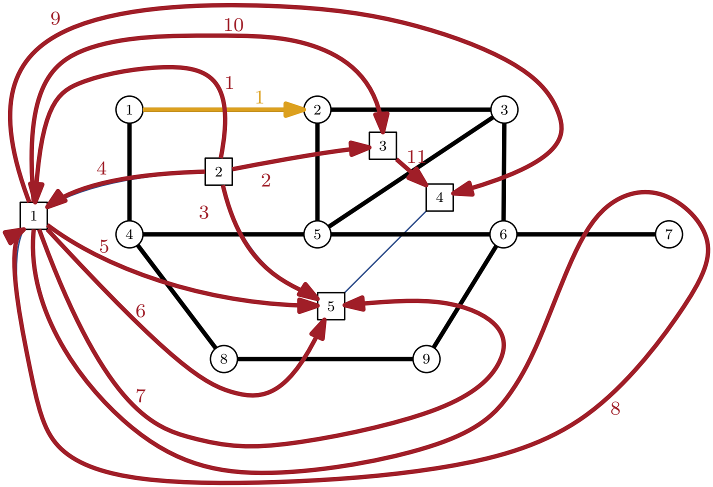
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

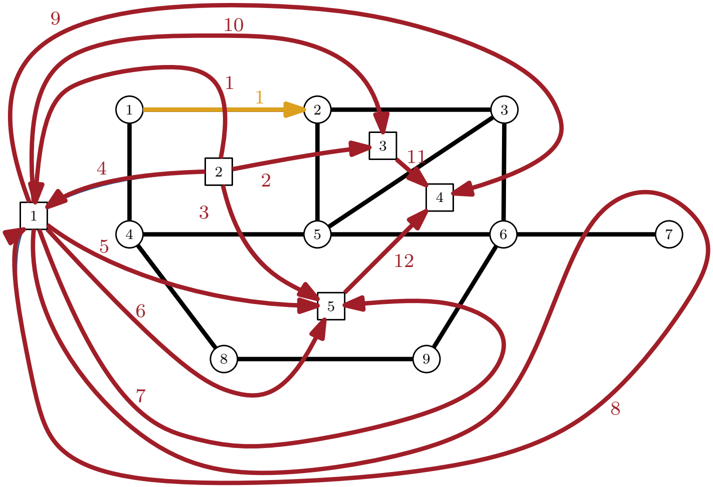
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

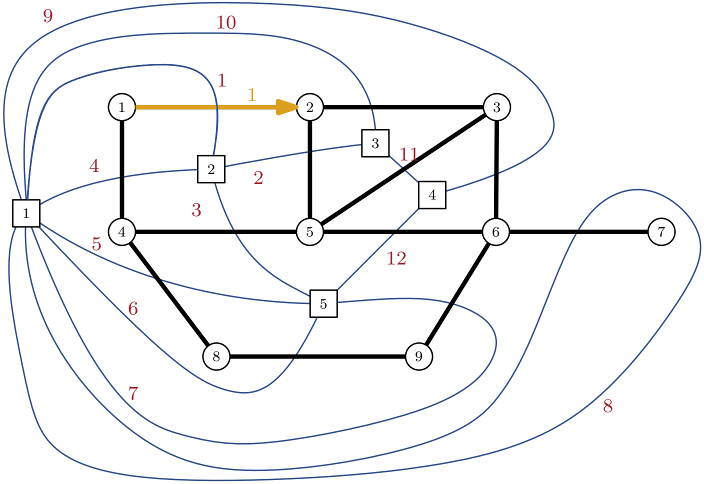
Duale Suche



Left-First-BFS auf G^*

- Queue
- Im Uhrzeigersinn hinzufügen

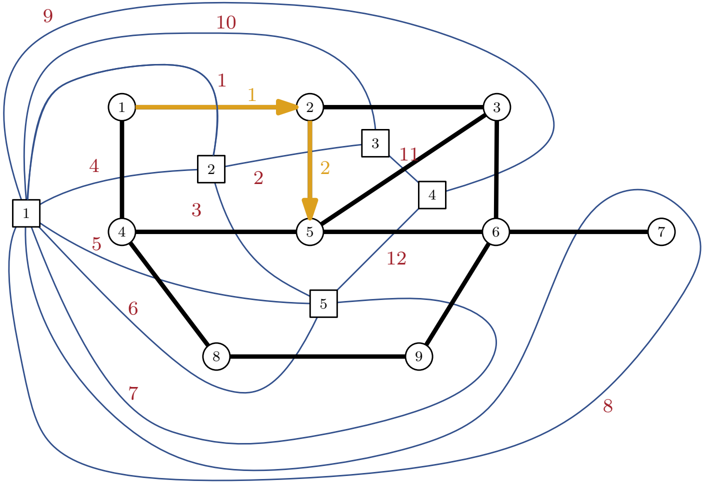
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

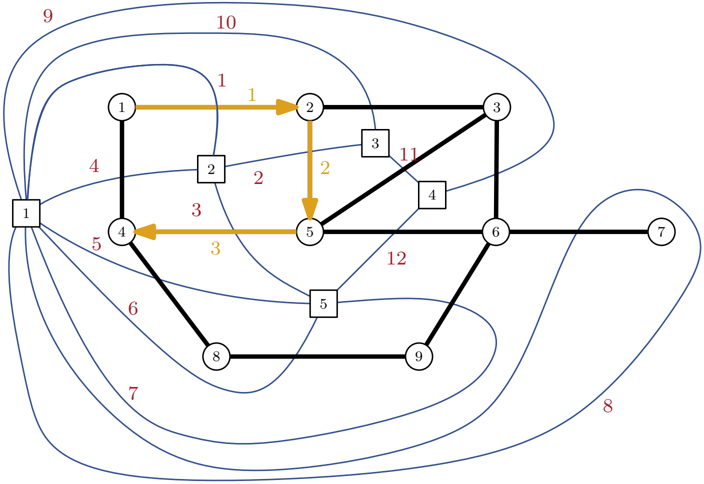
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

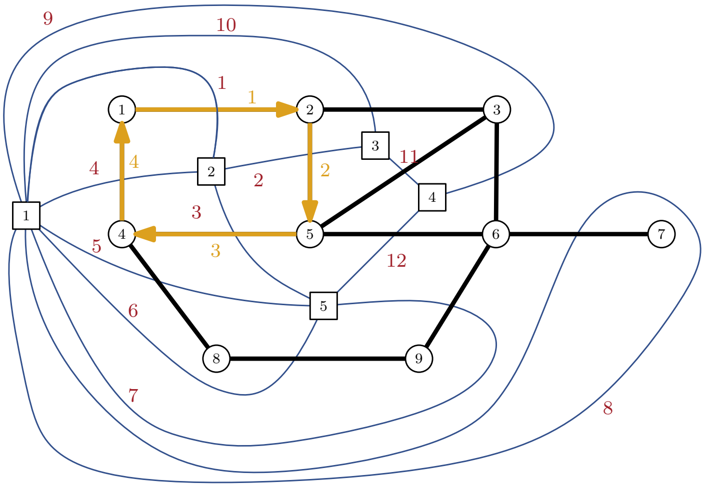
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

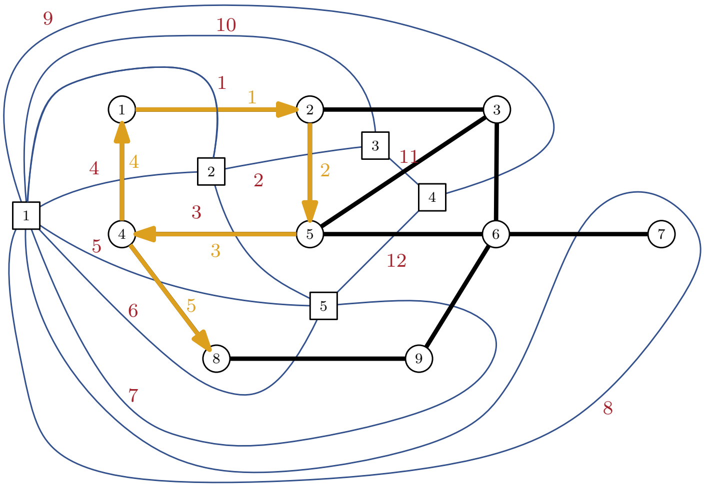
Duale Suche



Right-First-DFS auf *G*

- Stack
- Gegen den Uhrzeigersinn hinzufügen

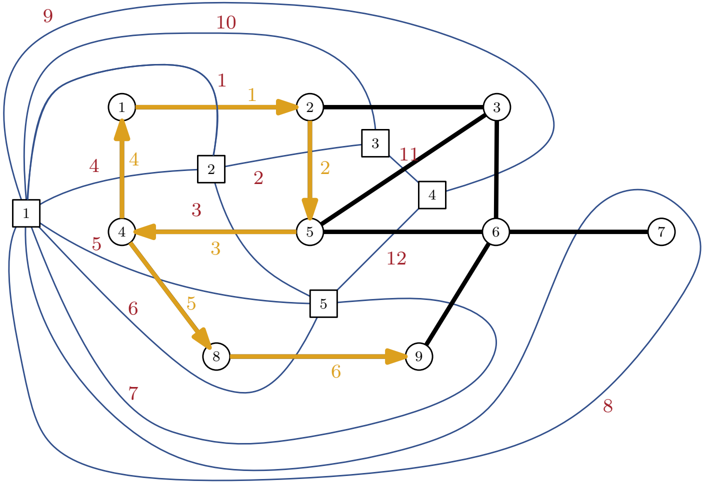
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

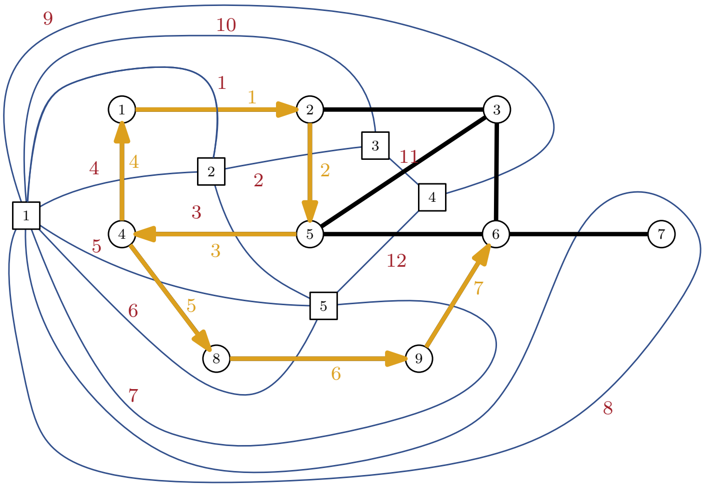
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

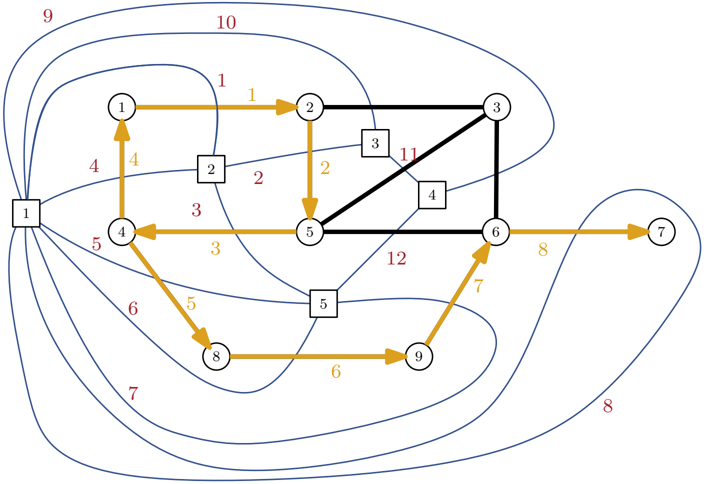
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

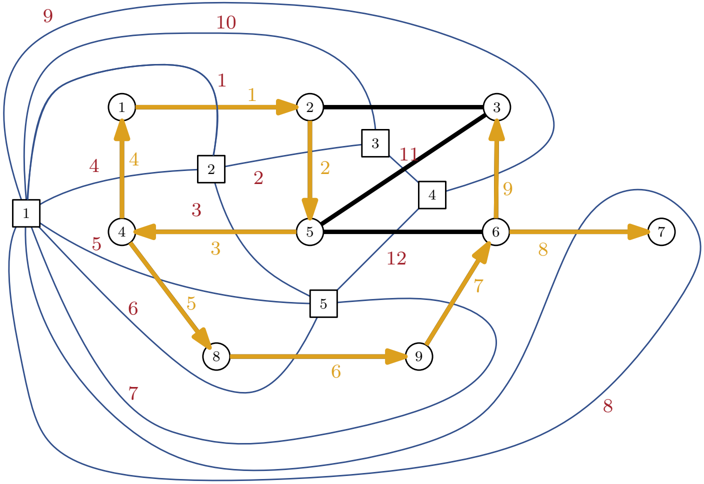
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

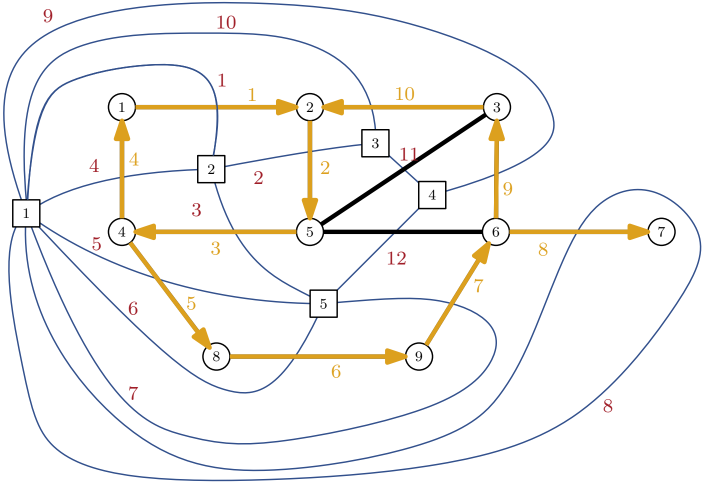
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

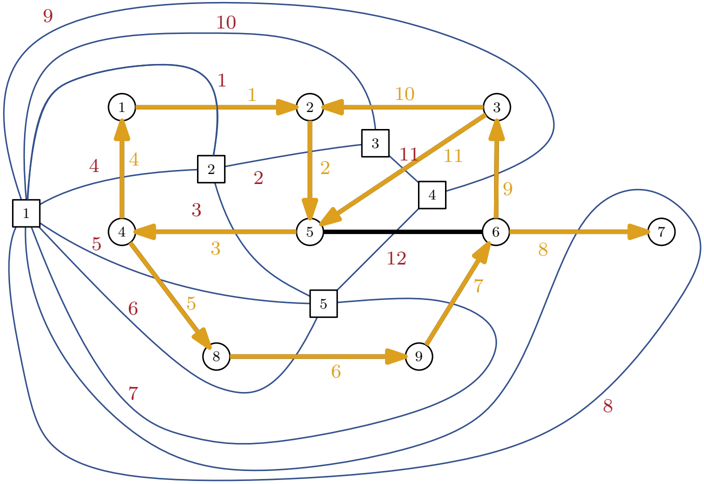
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

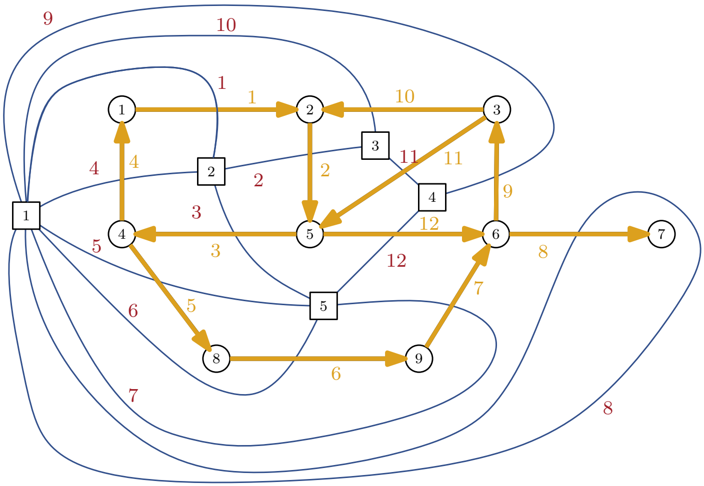
Duale Suche



Right-First-DFS auf G

- Stack
- Gegen den Uhrzeigersinn hinzufügen

Duale Suche



Right-First-DFS auf G

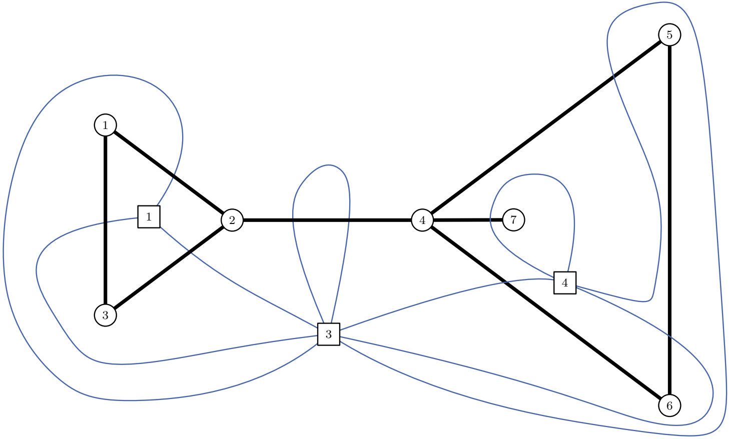
- Stack
- Gegen den Uhrzeigersinn hinzufügen

Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.

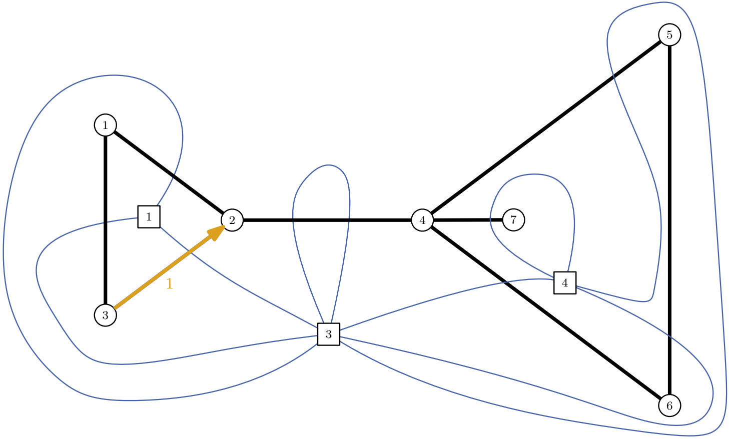
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



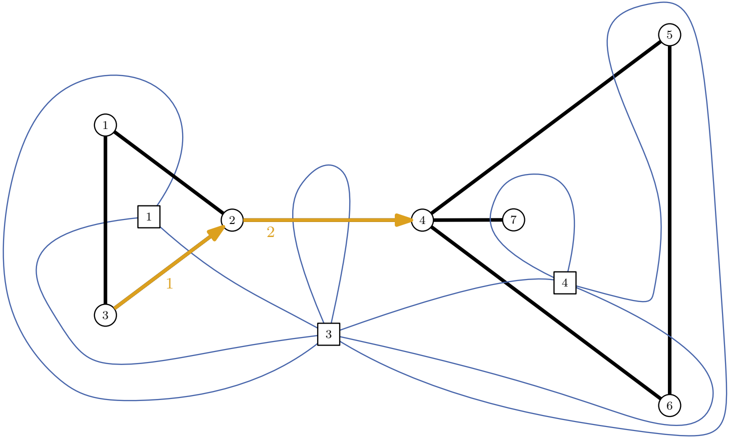
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



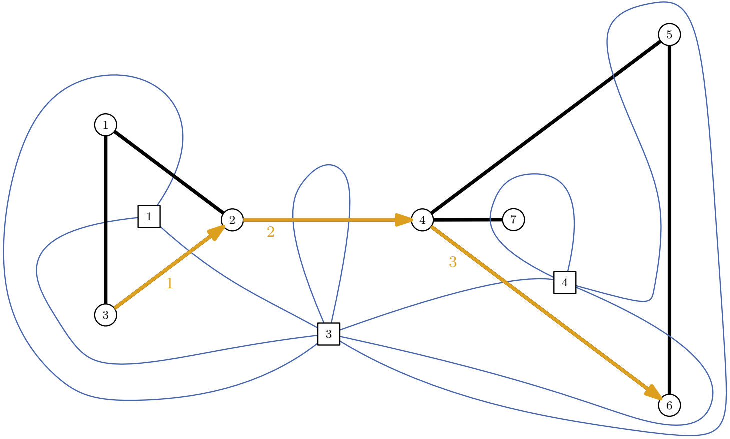
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



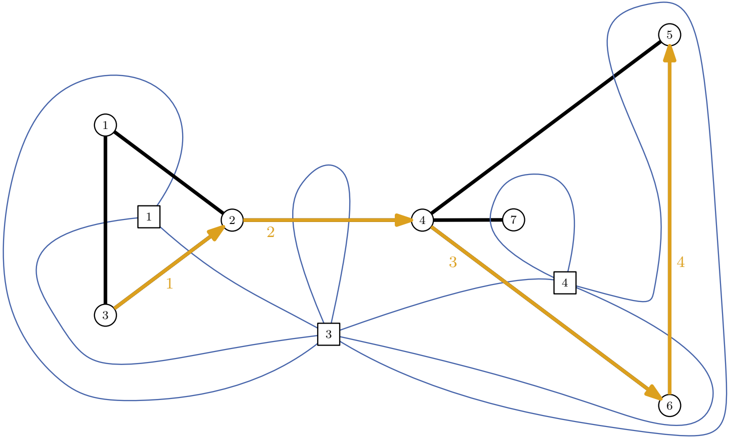
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



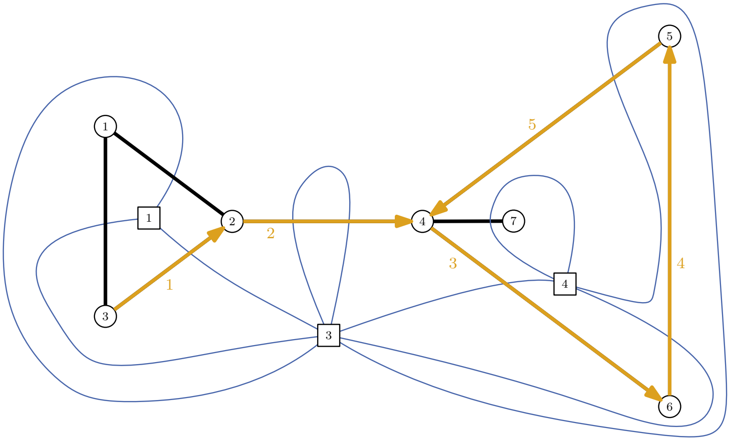
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



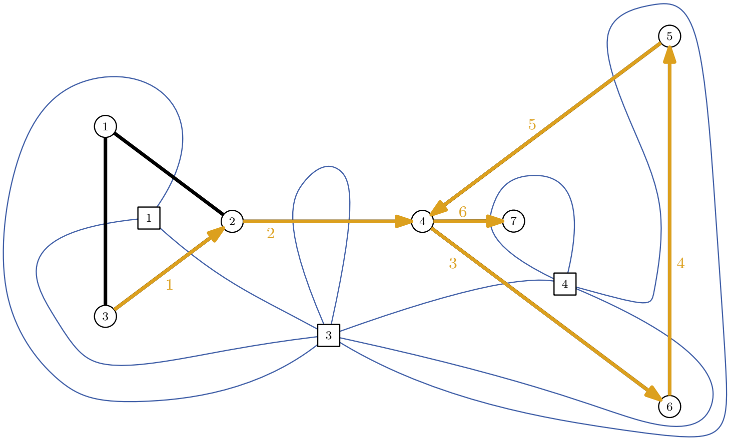
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



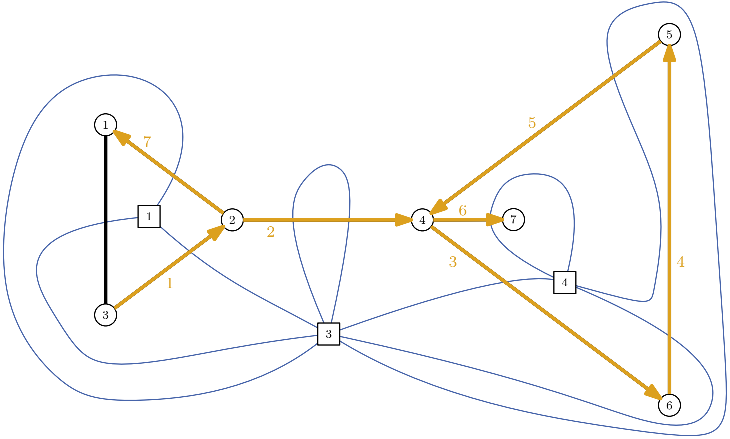
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



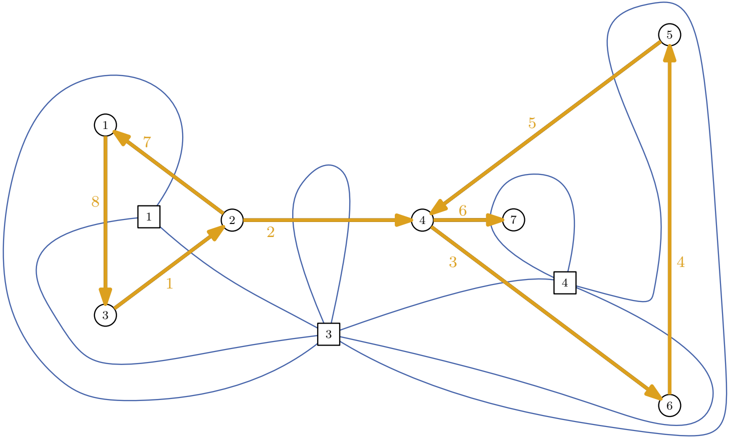
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



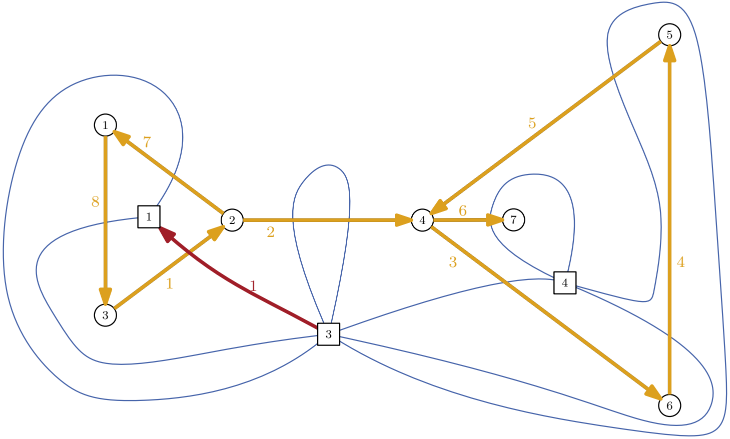
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



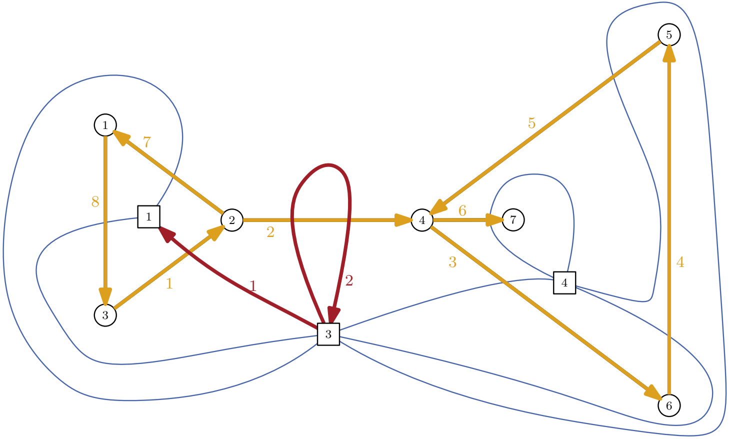
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



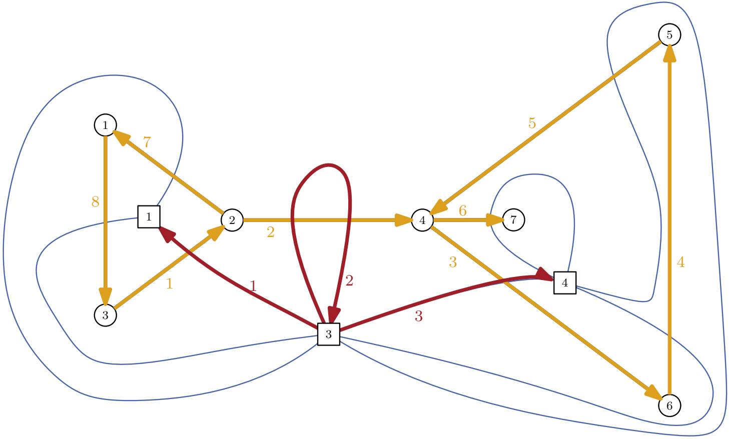
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



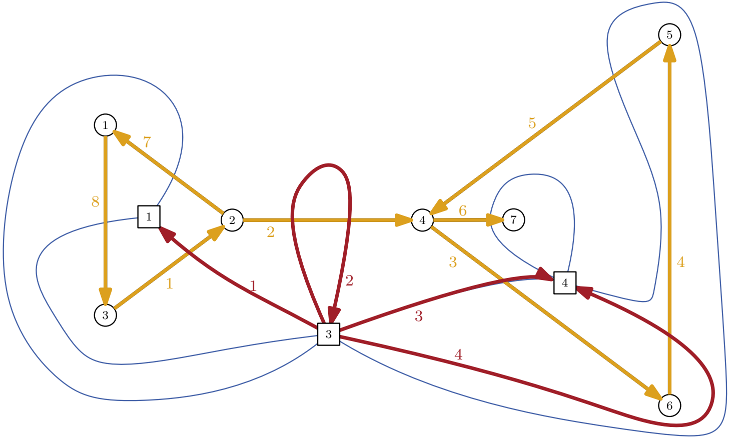
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



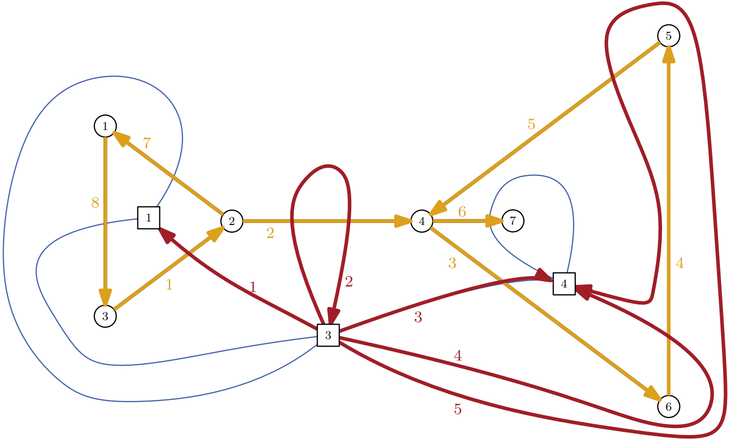
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



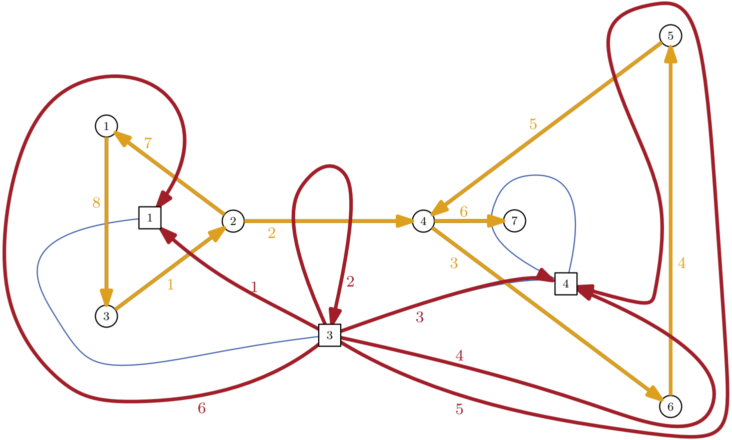
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



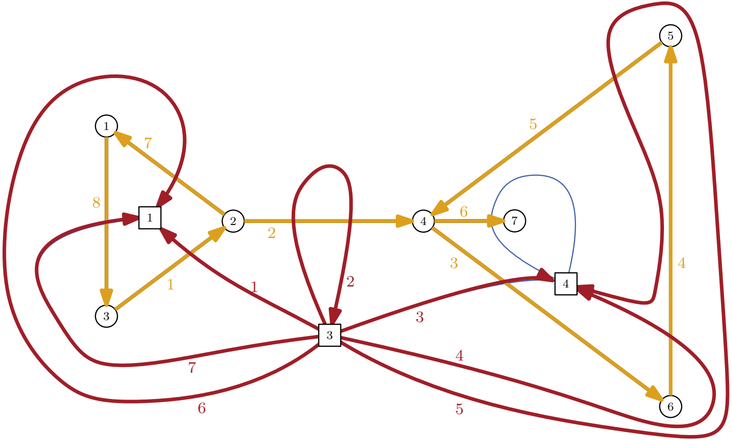
Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.



Duale Suche

Geben Sie einen Graphen an, für den R und R^* nicht dual sind.