Karlsruher Institut für Technologie

Algorithmen II
Übung am 26.11.2013

INSTITUT FÜR THEORETISCHE INFORMATIK · PROF. DR. DOROTHEA WAGNER

Matroide

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Unabhängigkeitssysteme

Definition 2.8: Unabhängigkeitssystem

Ein Tupel $(M, \mathcal{U})$, wobei $\mathcal{U} \subset 2^M$ ein Mengensystem über einer endlichen Menge M ist, heißt *Unabhängigkeitssystem*, wenn

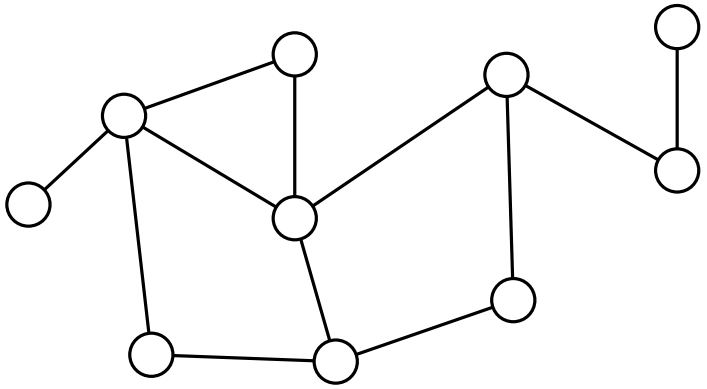
- $\emptyset \in \mathcal{U}$ und
- $I_1 \in \mathcal{U}, I_2 \subseteq I_1 \Rightarrow I_2 \in \mathcal{U}$.

Unabhängigkeitssysteme

Definition 2.8: Unabhängigkeitssystem
Ein Tupel $(M, \mathcal{U})$, wobei $\mathcal{U} \subset 2^M$ ein Mengensystem über einer endlichen Menge M ist, heißt *Unabhängigkeitssystem*, wenn

- $\emptyset \in \mathcal{U}$ und
- $I_1 \in \mathcal{U}, I_2 \subseteq I_1 \Rightarrow I_2 \in \mathcal{U}$.

Beispiel 2: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem $(E, \mathcal{U})$ mit $\mathcal{U} = \{E' \subseteq E : E' \text{ induziert einen Wald in } G\}$ ist ein Unabhängigkeitssystem.



Es gilt:

1. $\emptyset$ ist in $\mathcal{U}$ enthalten.

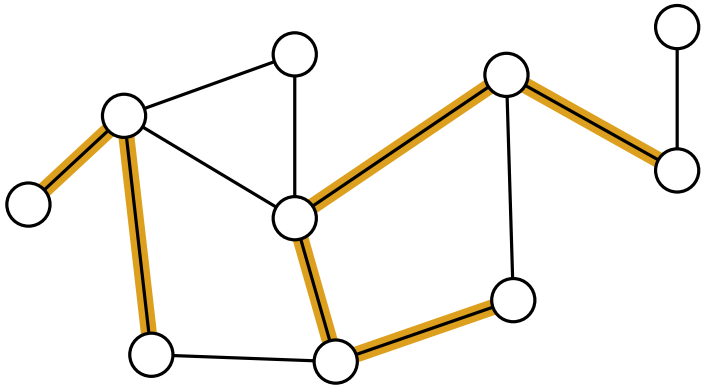
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Unabhängigkeitssysteme

Definition 2.8: Unabhängigkeitssystem
Ein Tupel $(M, \mathcal{U})$, wobei $\mathcal{U} \subset 2^M$ ein Mengensystem über einer endlichen Menge M ist, heißt *Unabhängigkeitssystem*, wenn

- $\emptyset \in \mathcal{U}$ und
- $I_1 \in \mathcal{U}, I_2 \subseteq I_1 \Rightarrow I_2 \in \mathcal{U}$.

Beispiel 2: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem $(E, \mathcal{U})$ mit $\mathcal{U} = \{E' \subseteq E : E' \text{ induziert einen Wald in } G\}$ ist ein Unabhängigkeitssystem.



Es gilt:

1. $\emptyset$ ist in $\mathcal{U}$ enthalten.
2. $I_1 \in \mathcal{U}, I_2 \subseteq I_1 \Rightarrow I_2 \in \mathcal{U}$

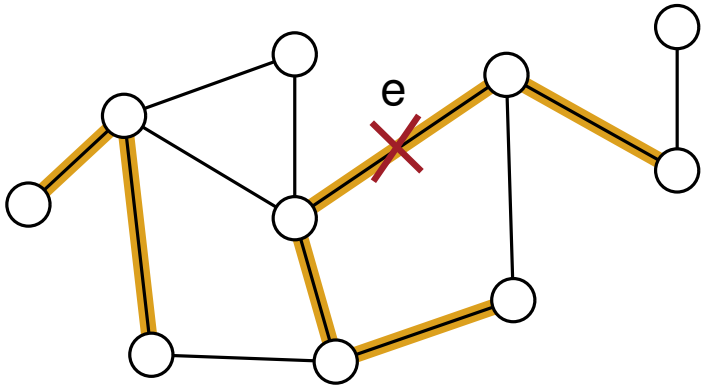
→ I_1 enthält nach Definition keine Kreise.

Unabhängigkeitssysteme

Definition 2.8: Unabhängigkeitssystem
Ein Tupel $(M, \mathcal{U})$, wobei $\mathcal{U} \subset 2^M$ ein Mengensystem über einer endlichen Menge M ist, heißt *Unabhängigkeitssystem*, wenn

- $\emptyset \in \mathcal{U}$ und
- $I_1 \in \mathcal{U}, I_2 \subseteq I_1 \Rightarrow I_2 \in \mathcal{U}$.

Beispiel 2: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem $(E, \mathcal{U})$ mit $\mathcal{U} = \{E' \subseteq E : E' \text{ induziert einen Wald in } G\}$ ist ein Unabhängigkeitssystem.



Es gilt:

1. $\emptyset$ ist in $\mathcal{U}$ enthalten.
2. $I_1 \in \mathcal{U}, I_2 \subseteq I_1 \Rightarrow I_2 \in \mathcal{U}$

→ I_1 enthält nach Definition keine Kreise.

→ Sei $I_2 = I_1 \setminus \{e\}$ für beliebige Kante $e \in I_1$.

→ I_2 enthält ebenfalls keine Kreise $\rightarrow$ I_2 ist Wald.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

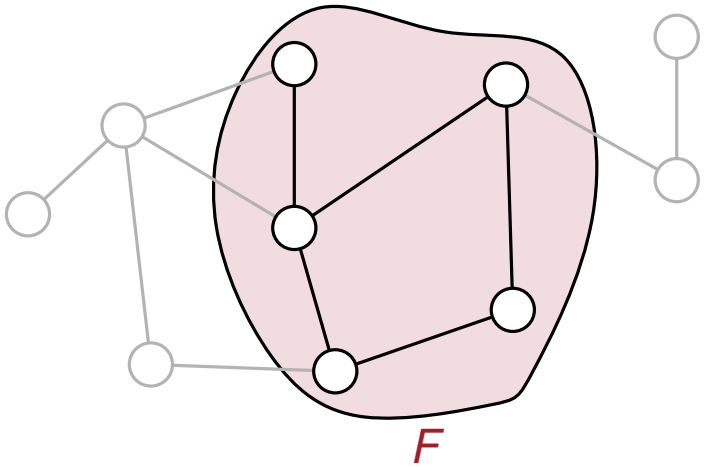
Basis

Definition: Basis, Basissystem & Rang

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem.

- Für $F \subseteq M$ ist jede unabhängige Menge $U \in \mathcal{U}$, $U \subseteq F$ die bezüglich $\subseteq$ maximal ist eine *Basis* von F.
- Eine Basis von M wird auch *Basis des Unabhängigkeitssystems* genannt. Die Menge aller Basen von $(M, \mathcal{U})$ heißt *Basissystem von* $(M, \mathcal{U})$.
- Für $F \subseteq M$ heißt $r(F) := max\{|B| : B \text{ ist Basis von } F\}$ der *Rang von* F.
- Der Rang von M wird auch *Rang des Unabhängigkeitssystems* genannt.

Beispiel: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem $(E, \mathcal{U})$ mit $\mathcal{U} = \{E' \subseteq E : E' \text{ induziert einen Wald in } G\}$ ist ein Unabhängigkeitssystem.



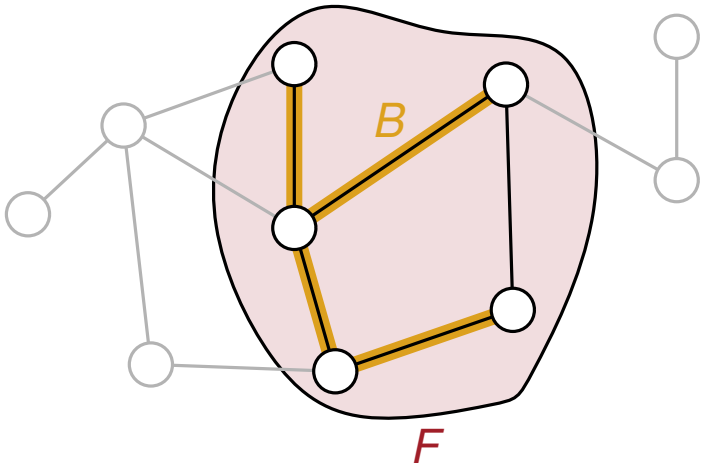
Basis

Definition: Basis, Basissystem & Rang

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem.

- Für $F \subseteq M$ ist jede unabhängige Menge $U \in \mathcal{U}$, $U \subseteq F$ die bezüglich $\subseteq$ maximal ist eine *Basis* von F.
- Eine Basis von M wird auch *Basis des Unabhängigkeitssystems* genannt. Die Menge aller Basen von $(M, \mathcal{U})$ heißt *Basissystem von* $(M, \mathcal{U})$.
- Für $F \subseteq M$ heißt $r(F) := max\{|B| : B \text{ ist Basis von } F\}$ der *Rang von* F.
- Der Rang von M wird auch *Rang des Unabhängigkeitssystems* genannt.

Beispiel: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem $(E, \mathcal{U})$ mit $\mathcal{U} = \{E' \subseteq E : E' \text{ induziert einen Wald in } G\}$ ist ein Unabhängigkeitssystem.



B ist Basis von F mit Rang 4

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

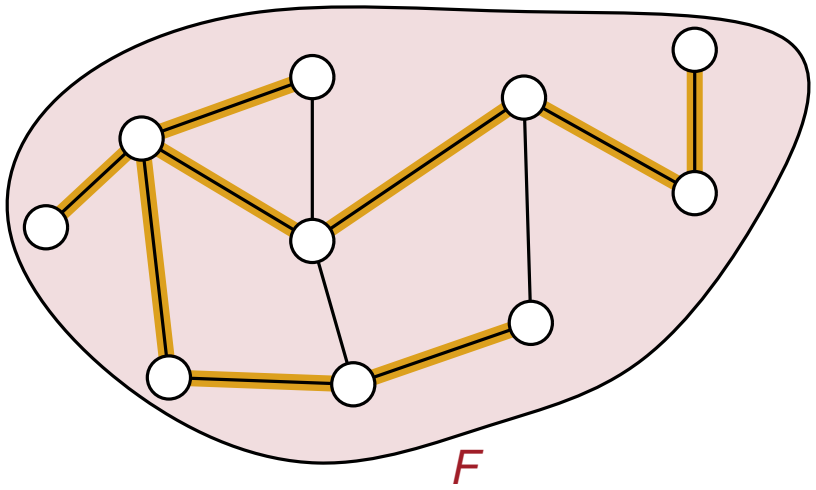
Basis

Definition: Basis, Basissystem & Rang

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem.

- Für $F \subseteq M$ ist jede unabhängige Menge $U \in \mathcal{U}$, $U \subseteq F$ die bezüglich $\subseteq$ maximal ist eine *Basis* von F.
- Eine Basis von M wird auch *Basis des Unabhängigkeitssystems* genannt. Die Menge aller Basen von $(M, \mathcal{U})$ heißt *Basissystem von* $(M, \mathcal{U})$.
- Für $F \subseteq M$ heißt $r(F) := max\{|B| : B \text{ ist Basis von } F\}$ der *Rang von* F.
- Der Rang von M wird auch *Rang des Unabhängigkeitssystems* genannt.

Beispiel: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem $(E, \mathcal{U})$ mit $\mathcal{U} = \{E' \subseteq E : E' \text{ induziert einen Wald in } G\}$ ist ein Unabhängigkeitssystem.



B ist Basis von F mit Rang 9

Aufspannende Bäume sind Basen von $\mathcal{U}$

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

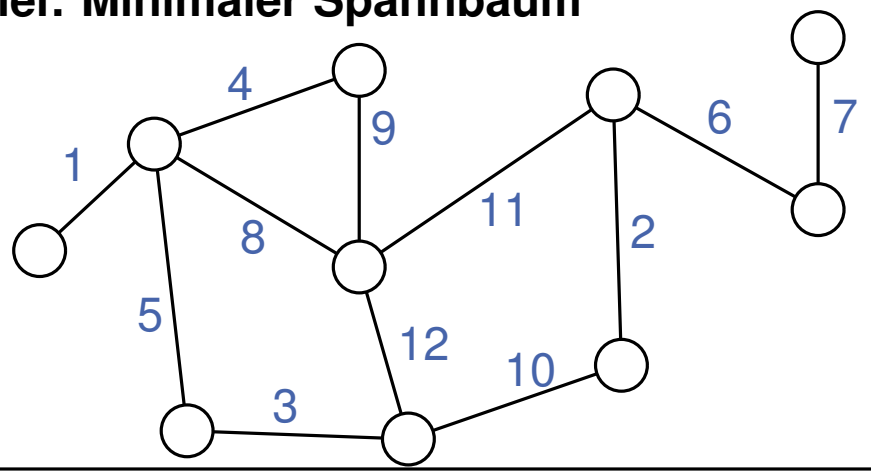
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

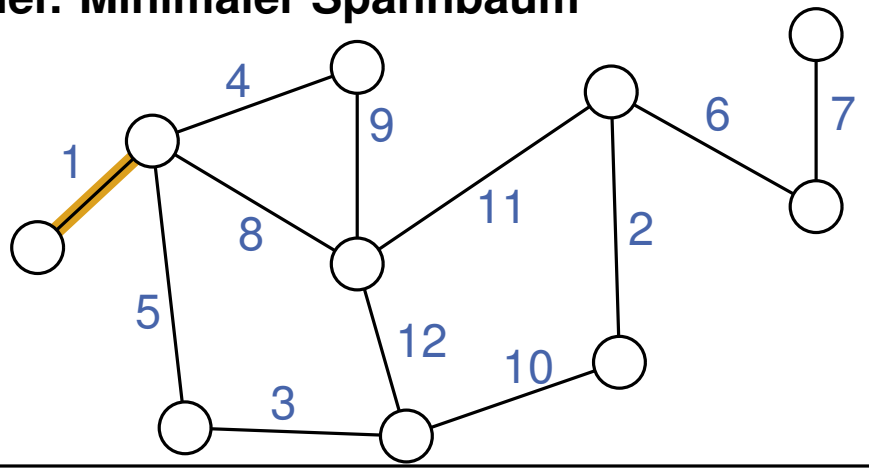
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

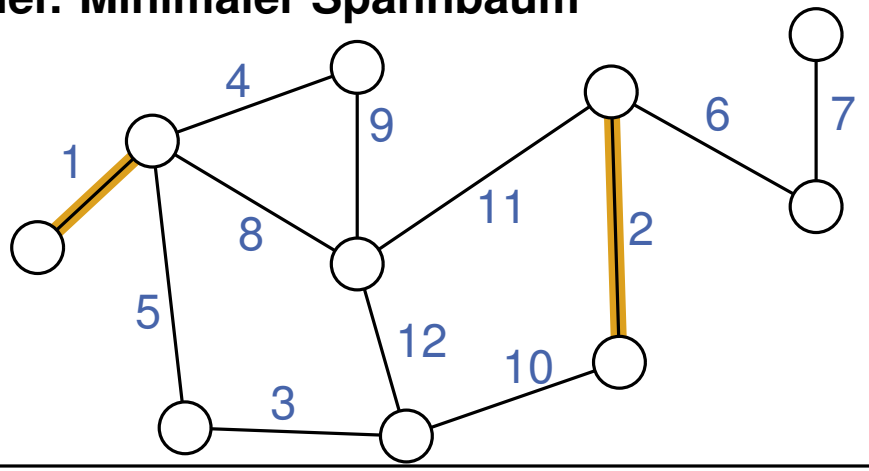
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

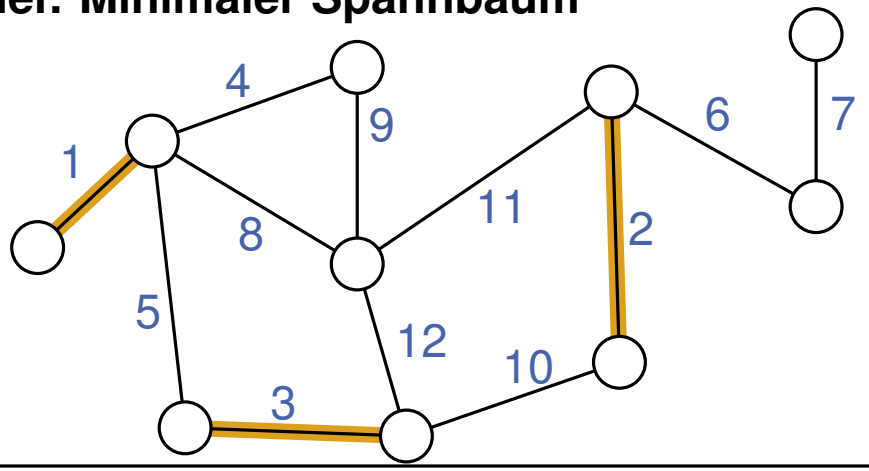
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

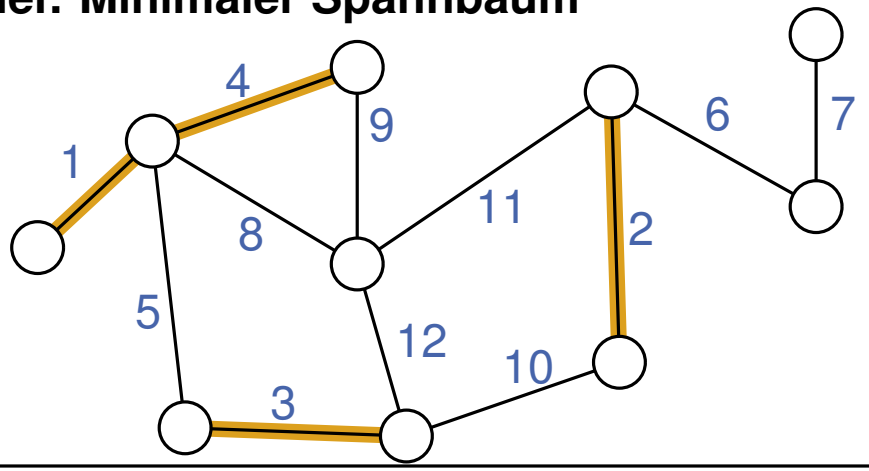
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

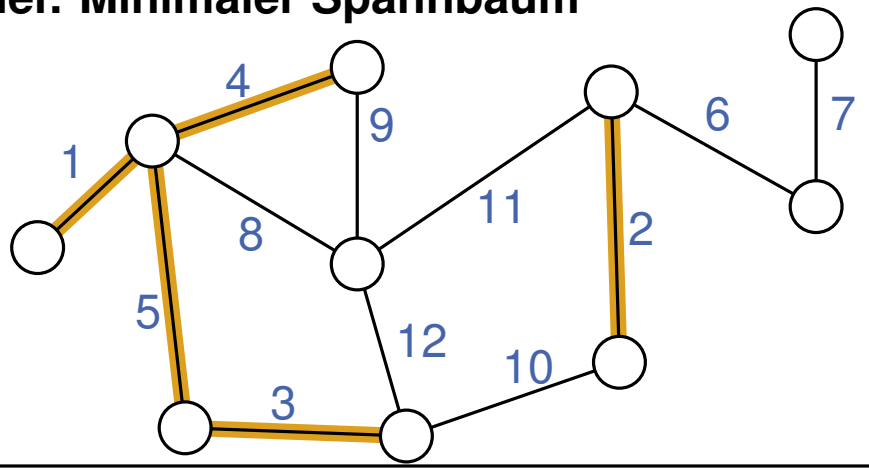
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

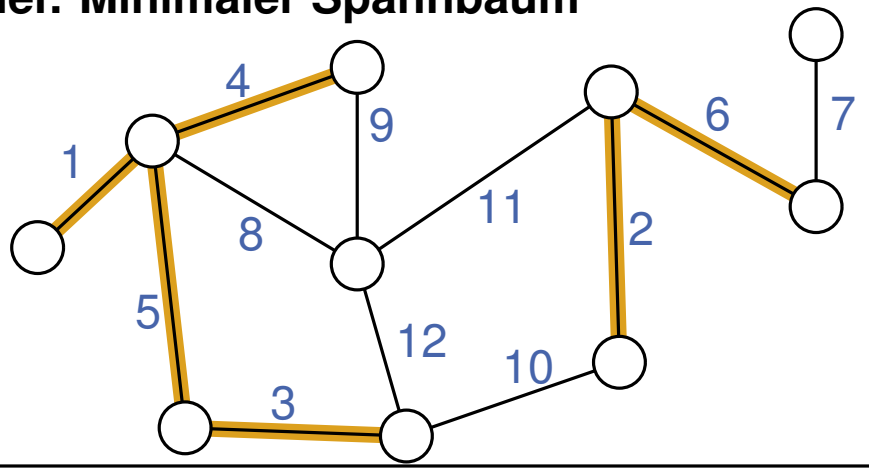
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

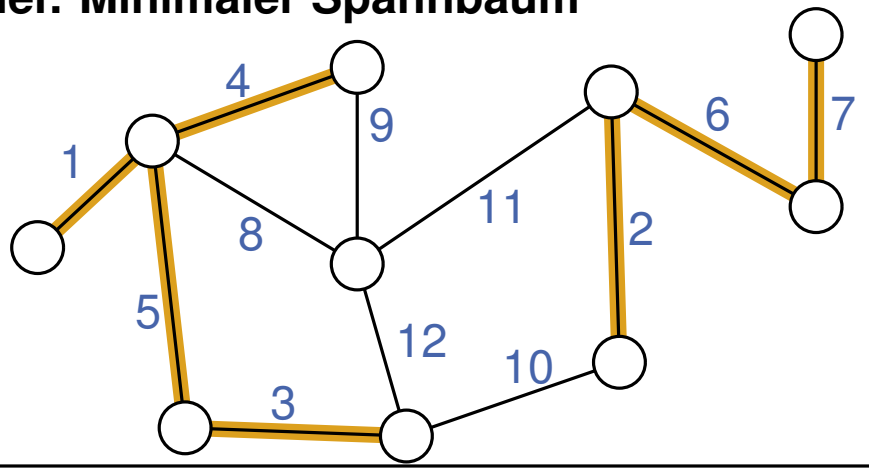
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

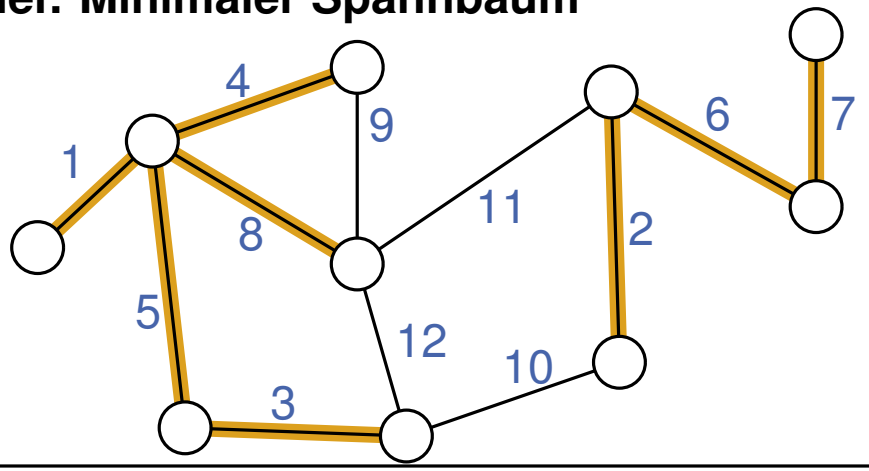
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

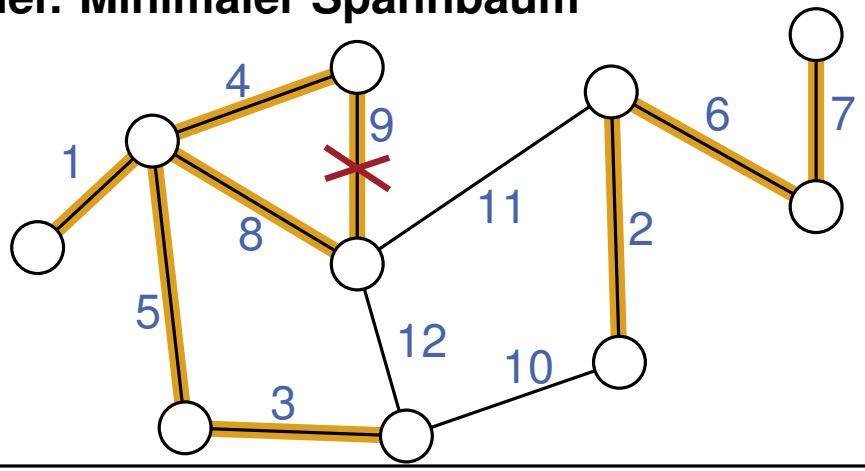
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

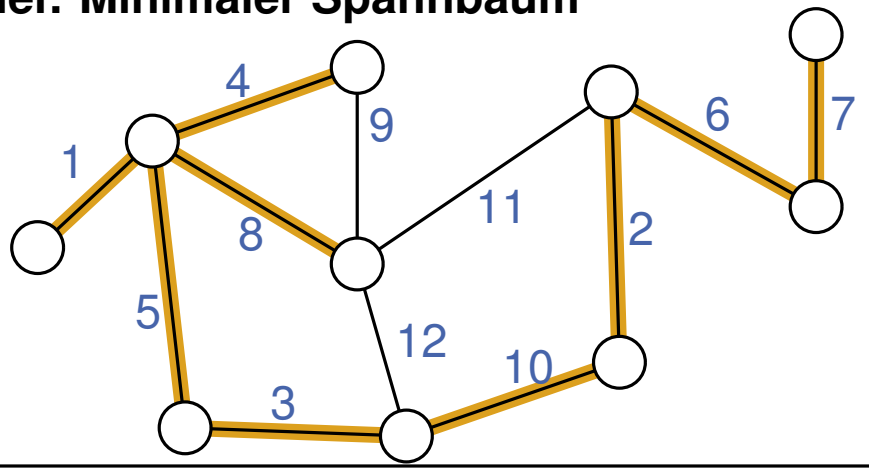
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

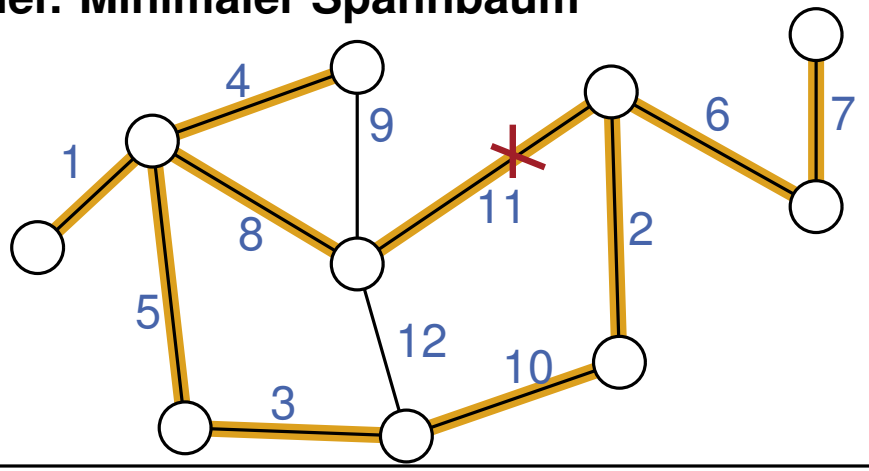
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

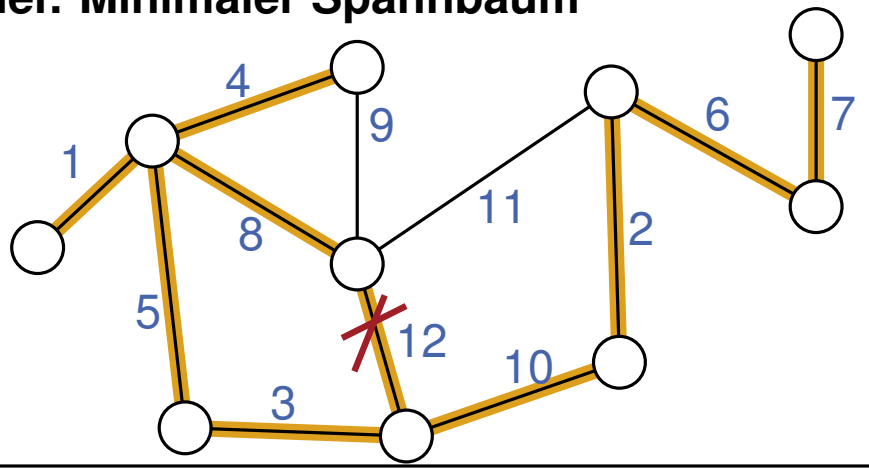
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

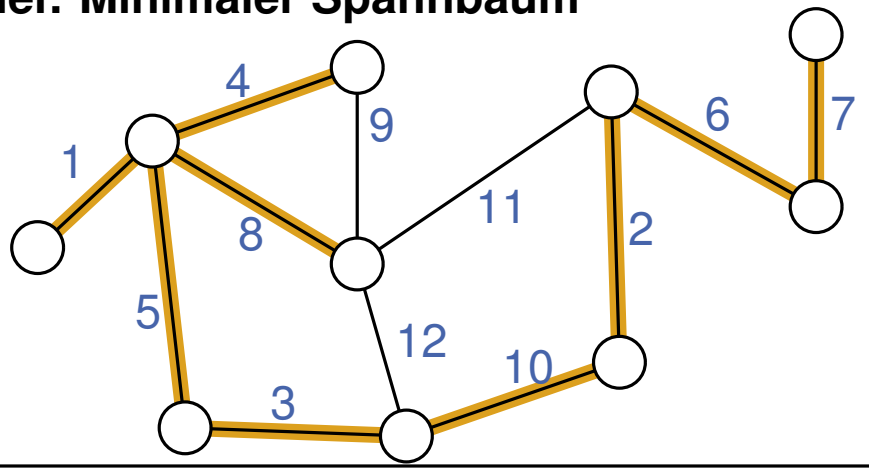
GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Beispiel: Minimaler Spannbaum



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimierungsprobleme

Sei $(M, \mathcal{U})$ ein Unabhängigkeitssystem mit Basissystem $\mathcal{B}$ und Gewichtsfunktion $w\colon M \longrightarrow \mathbb{R}$.

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

Mögliche Lösungsstrategie für diese Probleme: Greedy

GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitessystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Wann liefert das eine optimale Lösung?

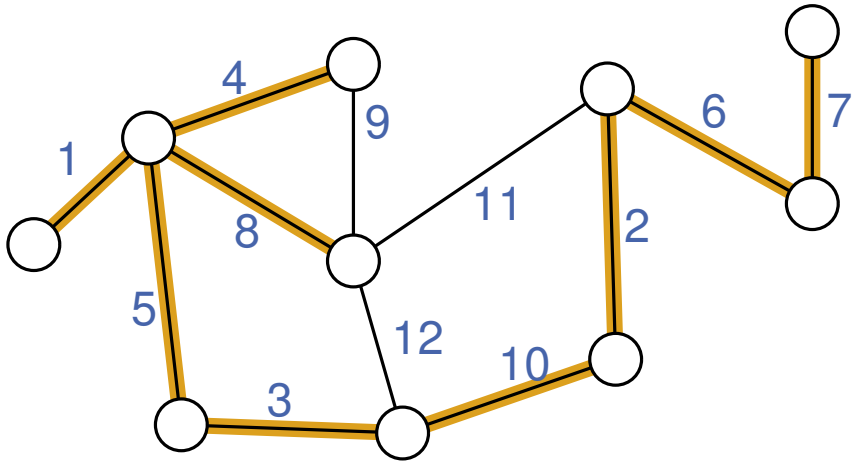
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Matroid

Definition: Matroid
Ein Unabhängigkeitssystem $(M, \mathcal{U})$ heißt *Matroid*, wenn für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, ein $e \in J \setminus I$ existiert, sodass $I \cup \{e\} \in \mathcal{U}$.

Äquivalent kann statt $|I| < |J|$ auch $|I| + 1 = |J|$ gefordert werden

Beispiel: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem $(E, \mathcal{U})$ mit $\mathcal{U} = \{E' \subseteq E : E'$ induziert einen Wald in $G\}$ ist ein Matroid. (**Ohne Beweis.**)



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Optimale Lösung für Matroide

Satz: Minimierungsprobleme auf Matroiden
Für ein Unabhängigkeitssystem $(M, \mathcal{U})$ mit Basissystem $\mathcal{B}$ sind äquivalent:

(*a*) Die Greedy-Methode liefert eine Optimallösung für das Optimierungsproblem über dem Basissystem $\mathcal{B}$.

(*b*) Die Greedy-Methode liefert eine Optimallösung für das Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$.

(*c*) $(M, \mathcal{U})$ ist ein Matroid.

Erinnerung:

Problem: Optimierungsproblem über dem Unabhängigkeitssystem $(M, \mathcal{U})$
Finde unabhängige Menge $U^\star \in \mathcal{U}$ sodass $w(U^\star)$ maximal ist.

Problem: Optimierungsproblem über dem Basissystem $\mathcal{B}$
Finde Basis $B^\star \in \mathcal{B}$ sodass $w(B^\star)$ minimal ist.

GREEDY-METHODE für Optimierungsproblem Π

Sortiere M aufsteigend (absteigend), falls Π Optimierungsproblem über Basissystem (Unabhängigkeitssystem) ist, sei $\ell_1, \ldots, \ell_{|M|}$ die Sortierung.

$I^\star \leftarrow \emptyset$

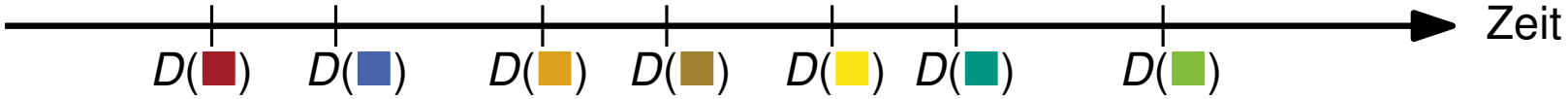
for $i = 1$ **to** $|M|$ **do**
  if $I^\star \cup \{\ell_i\} \in \mathcal{U}$ **then**
    $I^\star \leftarrow I^\star \cup \{\ell_i\}$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

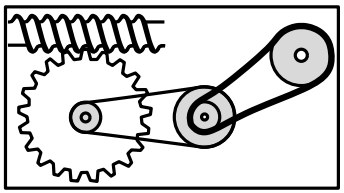
Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleiche Bearbeitungsdauer hat.

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



- Maschine $\mathcal{A}$.



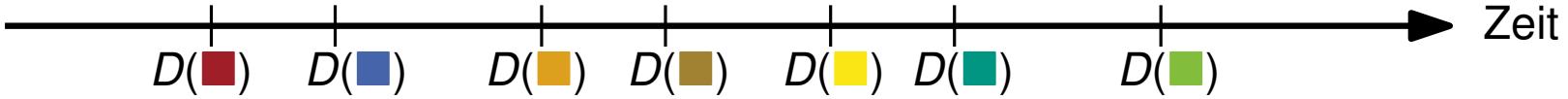
Kann immer nur einen Auftrag gleichzeitig abarbeiten.

Problem 1

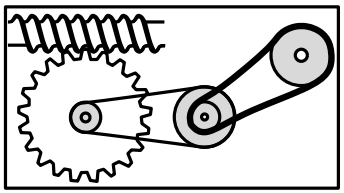
- Menge M an Aufträgen, wobei jeder Auftrag gleiche Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



- Maschine $\mathcal{A}$.



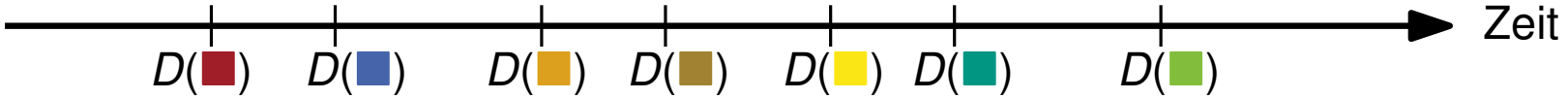
Kann immer nur einen Auftrag gleichzeitig abarbeiten.

Problem 1

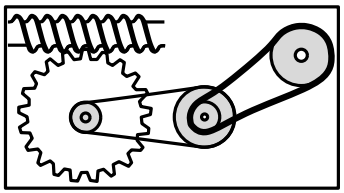
- Menge M an Aufträgen, wobei jeder Auftrag gleiche Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



- Maschine $\mathcal{A}$.



Kann immer nur einen Auftrag gleichzeitig abarbeiten.

$U \subseteq M$ heißt *zulässig*, wenn alle Aufträge in U rechtzeitig von $\mathcal{A}$ abgearbeitet werden können.

$$\mathcal{U} = \{U \subseteq M \mid U \text{ ist zulässig.}\}$$

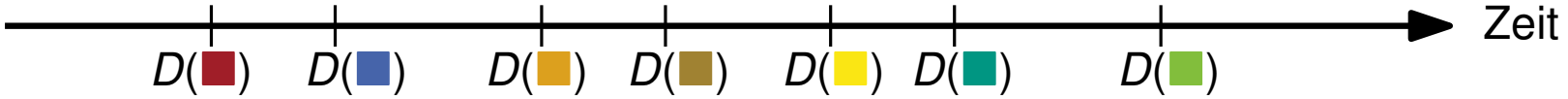
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 1

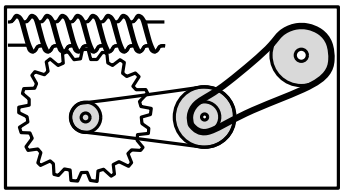
- Menge M an Aufträgen, wobei jeder Auftrag gleiche Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



- Maschine $\mathcal{A}$.



Kann immer nur einen Auftrag gleichzeitig abarbeiten.

$U \subseteq M$ heißt *zulässig*, wenn alle Aufträge in U rechtzeitig von $\mathcal{A}$ abgearbeitet werden können.

$$\mathcal{U} = \{U \subseteq M \mid U \text{ ist zulässig.}\}$$

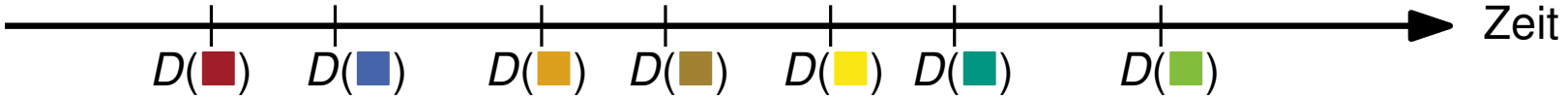
Zeige: $\mathcal{U}$ ist Matroid.

Problem 1

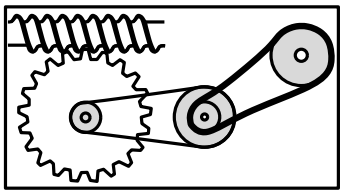
- Menge M an Aufträgen, wobei jeder Auftrag gleiche Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



- Maschine $\mathcal{A}$.



Kann immer nur einen Auftrag gleichzeitig abarbeiten.

$U \subseteq M$ heißt *zulässig*, wenn alle Aufträge in U rechtzeitig von $\mathcal{A}$ abgearbeitet werden können.

$$\mathcal{U} = \{U \subseteq M \mid U \text{ ist zulässig.}\}$$

Zeige: $\mathcal{U}$ ist Matroid.

1. $\emptyset \in \mathcal{U}$
2. $I_1 \in \mathcal{U}, I_2 \subseteq I_1 \Rightarrow I_2 \in \mathcal{U}$

   Wenn alle Aufträge in I_1 rechtzeitig bearbeitet werden können, dann auch die Aufträge in I_2.

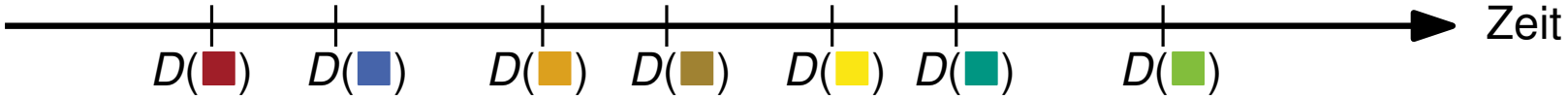
→ $\mathcal{U}$ ist Unabhängigkeitssystem

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.

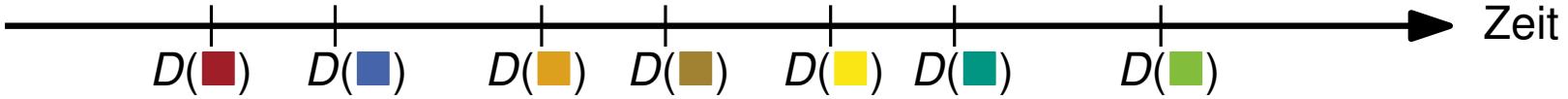


Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

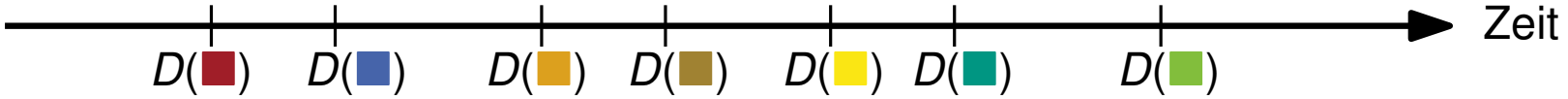
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

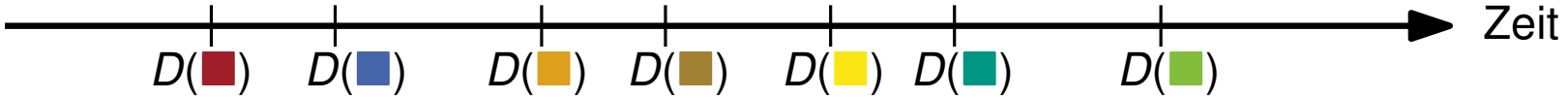
Bearbeitungszeit für j Aufträge beträgt j.

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

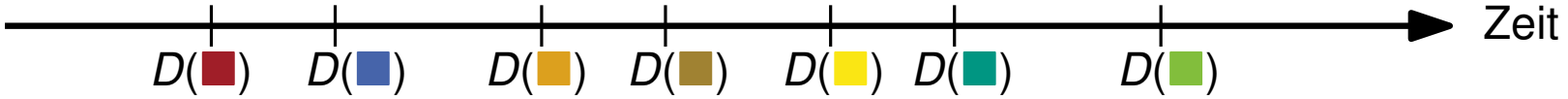
$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

Beweis:

"$\Rightarrow$"

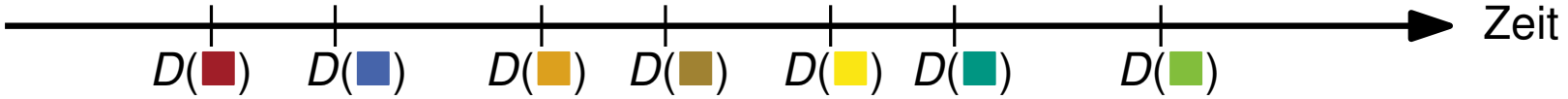
Annahme: $\{A_1, \ldots, A_k\}$ ist zulässig, aber es gibt ein $i \leq k$ mit $D(A_i) < i$.

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

Beweis:

"$\Rightarrow$"

Annahme: $\{A_1, \ldots, A_k\}$ ist zulässig, aber es gibt ein $i \leq k$ mit $D(A_i) < i$.

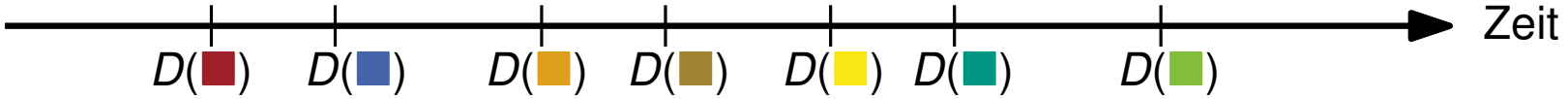
Wegen $D(A_1) \leq \cdots \leq D(A_i)$ müssen i Aufträge in $i - 1$ Zeiteinheiten abgearbeitet werden.

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

Beweis:

"$\Leftarrow$" $\forall i \leq k : D(A_i) \geq i$: Arbeite $A_1, \ldots, A_k$ nacheinander ab.

Wenn A_i abgearbeitet wird, wurden bisher maximal $i - 1$ Zeiteinheiten verbraucht.

Wegen $D(A_i) \geq i$ verbleibt genügend Zeit, um A_i abzuarbeiten.

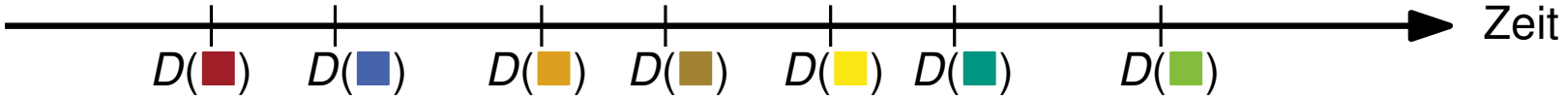
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

$I = \{A_1, \ldots, A_k\} \in \mathcal{U}$ $J = \{A'_1, \ldots, A'_{k+1}\} \in U$ mit $|J| = |I| + 1$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.

Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

$I = \{A_1, \ldots, A_k\} \in \mathcal{U}$ $J = \{A'_1, \ldots, A'_{k+1}\} \in U$ mit $|J| = |I| + 1$

Annahme: $D(A_1) \leq \cdots \leq D(A_k)$ und $D(A'_1) \leq \cdots \leq D(A'_{k+1})$

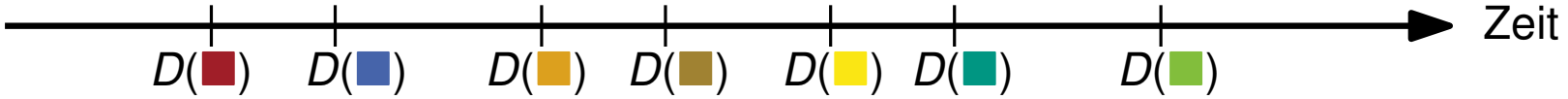
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

$I = \{A_1, \ldots, A_k\} \in \mathcal{U}$ $\quad$ $J = \{A'_1, \ldots, A'_{k+1}\} \in U$ $\quad$ mit $|J| = |I| + 1$

Annahme: $D(A_1) \leq \cdots \leq D(A_k)$ und $D(A'_1) \leq \cdots \leq D(A'_{k+1})$

Sei x maximaler Index mit $A_x \neq A'_{x+1}$:

$\cdots$	A_x	A_{x+1}	$\cdots$	A_{k-1}	A_k
	$\neq$	$=$	$\cdots$	$=$	$=$
$\cdots$	A'_{x+1}	A'_{x+2}	$\cdots$	A'_k	A'_{k+1}

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.

Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

$I = \{A_1, \ldots, A_k\} \in \mathcal{U}$ $J = \{A'_1, \ldots, A'_{k+1}\} \in U$ mit $|J| = |I| + 1$

Annahme: $D(A_1) \leq \cdots \leq D(A_k)$ und $D(A'_1) \leq \cdots \leq D(A'_{k+1})$

Sei x maximaler Index mit $A_x \neq A'_{x+1}$:

$\cdots$	A_x	A_{x+1}	$\cdots$	A_{k-1}	A_k
	$\neq$	$=$	$\cdots$	$=$	$=$
$\cdots$	A'_{x+1}	A'_{x+2}	$\cdots$	A'_k	A'_{k+1}

Zeige: $\{A_1, \ldots, A_k\} \cup \{A'_{x+1}\}$ ist zulässig.

1. Fall: $x = k$

denn: $\forall i \leq k$: $D(A_i) \geq i$ weil I zulässig.

$D(A'_{k+1}) \geq k + 1$ weil J zulässig.

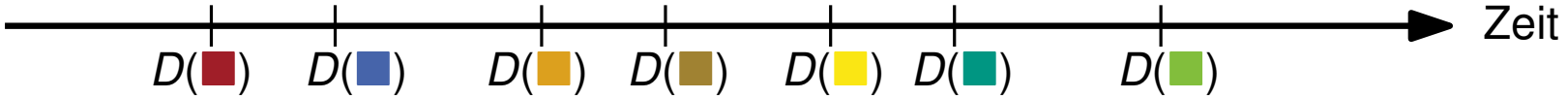
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

$I = \{A_1, \ldots, A_k\} \in \mathcal{U}$ $J = \{A'_1, \ldots, A'_{k+1}\} \in U$ mit $|J| = |I| + 1$

Annahme: $D(A_1) \leq \cdots \leq D(A_k)$ und $D(A'_1) \leq \cdots \leq D(A'_{k+1})$

Sei x maximaler Index mit $A_x \neq A'_{x+1}$:

$\cdots$	A_x	A_{x+1}	$\cdots$	A_{k-1}	A_k
	$\neq$	$=$	$\cdots$	$=$	$=$
$\cdots$	A'_{x+1}	A'_{x+2}	$\cdots$	A'_k	A'_{k+1}

Zeige: $\{A_1, \ldots, A_k\} \cup \{A'_{x+1}\}$ ist zulässig.

2. Fall: $x = j$ für $j < k$

denn: $1 \leq i \leq j$: $D(A_i) \geq i$ weil I zulässig.

$j+1 < i \leq k$: $D(A'_{j+1}) \geq j+1$, $D(A_i) \geq i$ weil J zulässig.

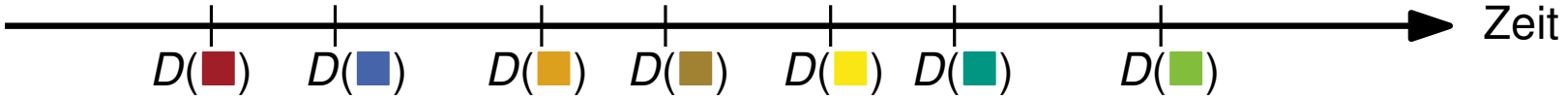
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



Zeige: Für alle $I, J \in \mathcal{U}$ mit $|I| < |J|$, existiert ein $e \in J \setminus I$, sodass $I \cup \{e\} \in \mathcal{U}$

Bearbeitungszeit für j Aufträge beträgt j.

Sei $\{A_1, ..., A_k\} \subseteq M$ von k paarweise verschiedene Aufträge mit $D(A_1) \leq \cdots \leq D(A_k)$

$$\{A_1, \ldots, A_k\} \text{ ist zulässig} \quad \Leftrightarrow \quad \forall i \leq k : D(A_i) \geq i$$

$I = \{A_1, \ldots, A_k\} \in \mathcal{U}$ $\quad$ $J = \{A'_1, \ldots, A'_{k+1}\} \in U$ $\quad$ mit $|J| = |I| + 1$

Annahme: $D(A_1) \leq \cdots \leq D(A_k)$ und $D(A'_1) \leq \cdots \leq D(A'_{k+1})$

Sei x maximaler Index mit $A_x \neq A'_{x+1}$:

$\cdots$	A_x	A_{x+1}	$\cdots$	A_{k-1}	A_k
	$\neq$	$=$	$\cdots$	$=$	$=$
$\cdots$	A'_{x+1}	A'_{x+2}	$\cdots$	A'_k	A'_{k+1}

Damit ist $\mathcal{U}$ ein Matroid.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

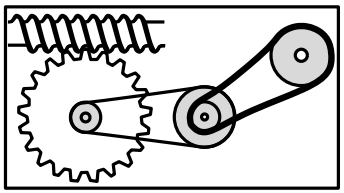
Problem 1

- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.

- Maschine $\mathcal{A}$.



Kann immer nur einen Auftrag gleichzeitig abarbeiten.

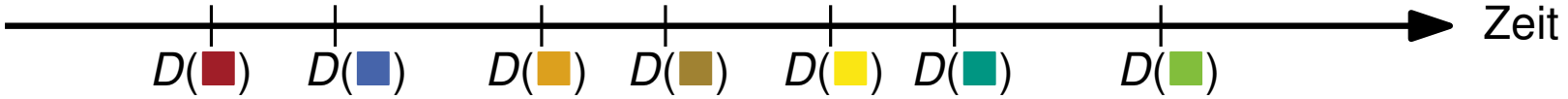
Problemstellung:
Zahle Strafe $P(A)$, wenn Auftrag nicht abgearbeitet wurde.
Wie Aufträge abarbeiten, damit gesamte Strafe minimiert wird?

Problem 1

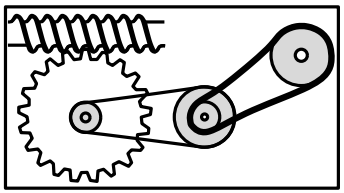
- Menge M an Aufträgen, wobei jeder Auftrag gleich Bearbeitungsdauer hat.

Normiere Aufträge: Bearbeitungsdauer 1

- Jeder Auftrag A hat eine Deadline $D(A) \in \mathbb{R}^+$ bis zu der er fertig sein muss.



- Maschine $\mathcal{A}$.



Kann immer nur einen Auftrag gleichzeitig abarbeiten.

Problemstellung:
Zahle Strafe $P(A)$, wenn Auftrag nicht abgearbeitet wurde.
Wie Aufträge abarbeiten, damit gesamte Strafe minimiert wird?

Sortiere M nicht-aufsteigend nach Strafe: $M = \{A_1, \ldots, A_k\}$ mit $P(A_i) \geq P(A_{i+1})$

$S \leftarrow \emptyset$

for $i = 1$ **to** k **do**
  if $S \cup \{A_i\}$ *ist zulässig* **then**
    $S \leftarrow S \cup \{A_i\}$

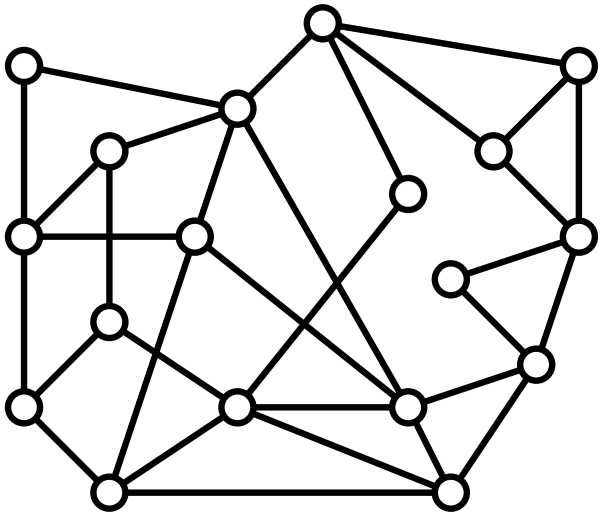
Sortiere S nicht-absteigend nach Deadlines und führe entsprechend Aufträge aus.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Kreisbasen

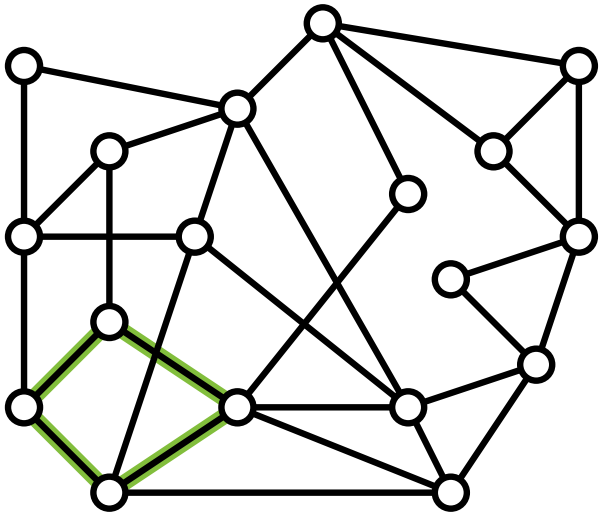
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Motivation



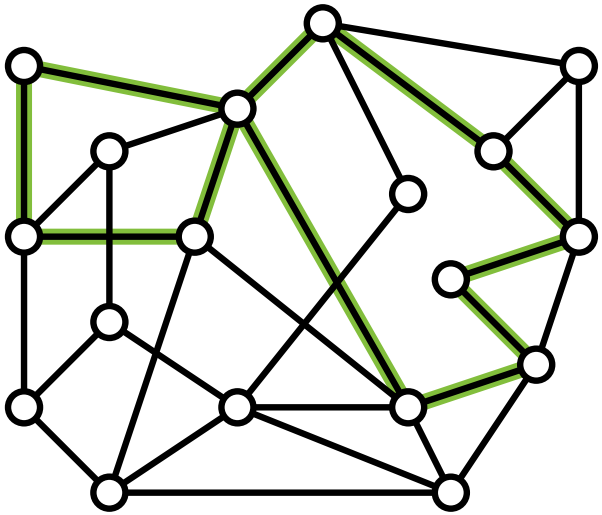
- Ein Graph kann sehr viele Kreise haben.

Motivation



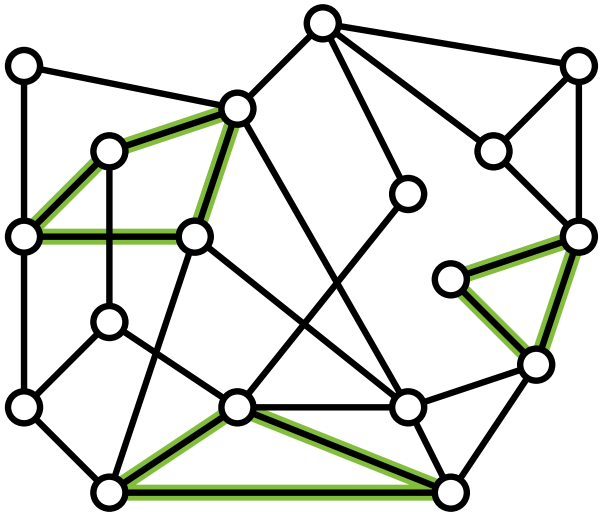
- Ein Graph kann sehr viele Kreise haben.

Motivation



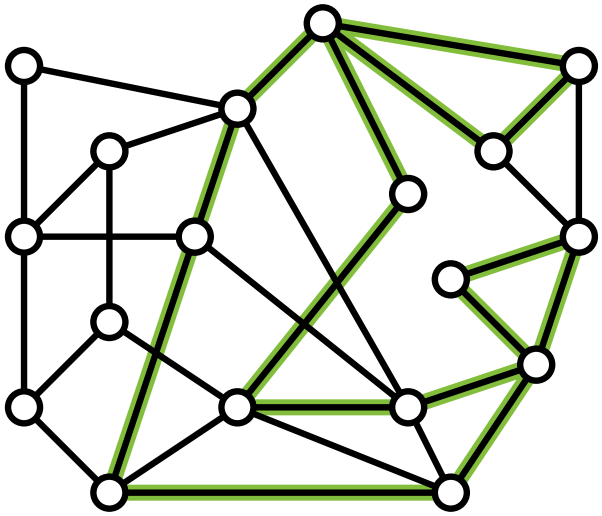
- Ein Graph kann sehr viele Kreise haben.

Motivation



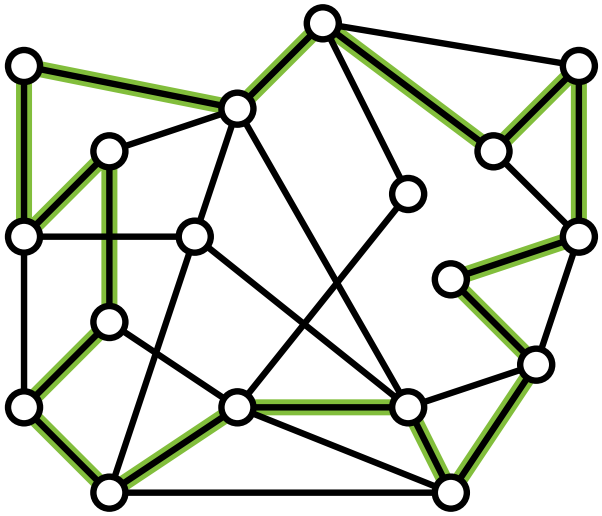
- Ein Graph kann sehr viele Kreise haben.

Motivation



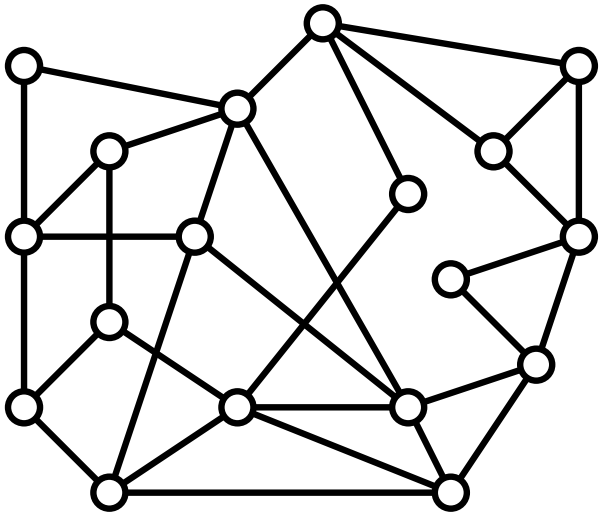
- Ein Graph kann sehr viele Kreise haben.

Motivation

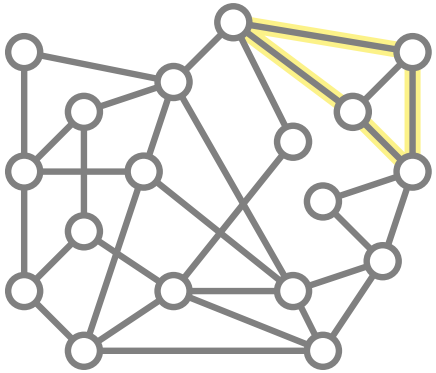


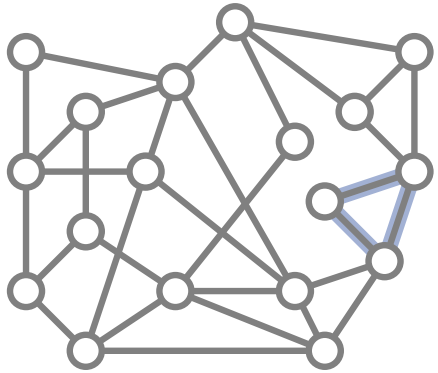
- Ein Graph kann sehr viele Kreise haben.

Motivation

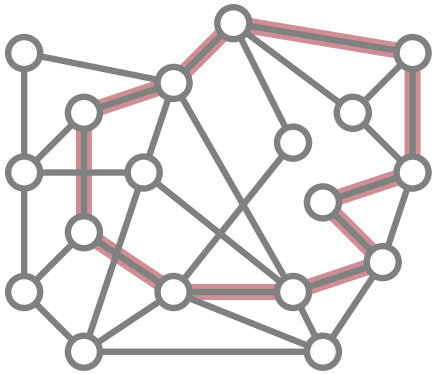


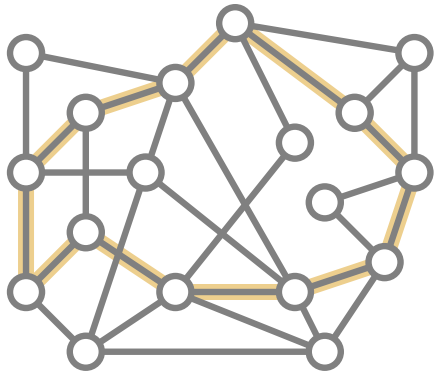
- Ein Graph kann sehr viele Kreise haben.



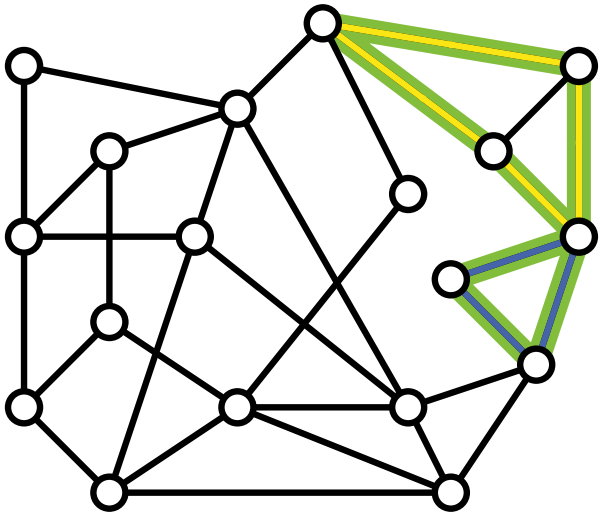


- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

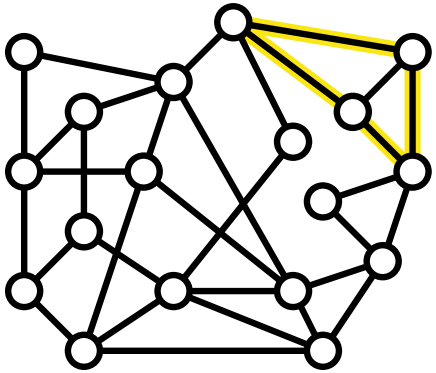


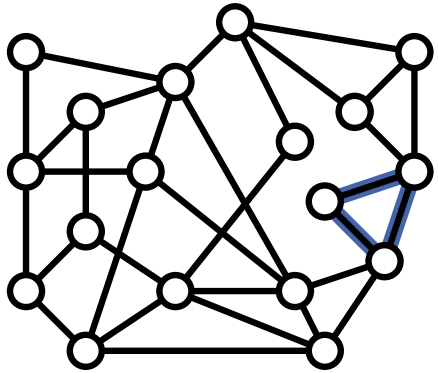


Motivation

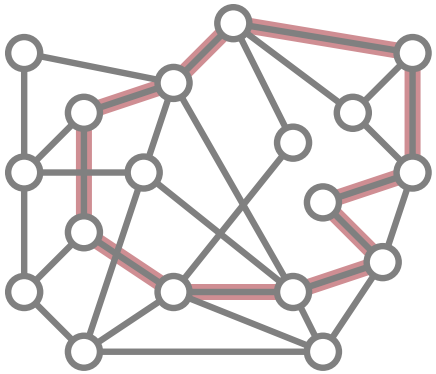


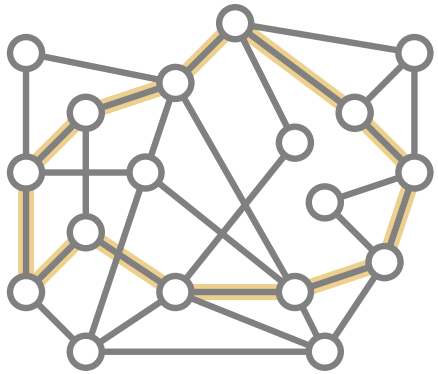
- Ein Graph kann sehr viele Kreise haben.



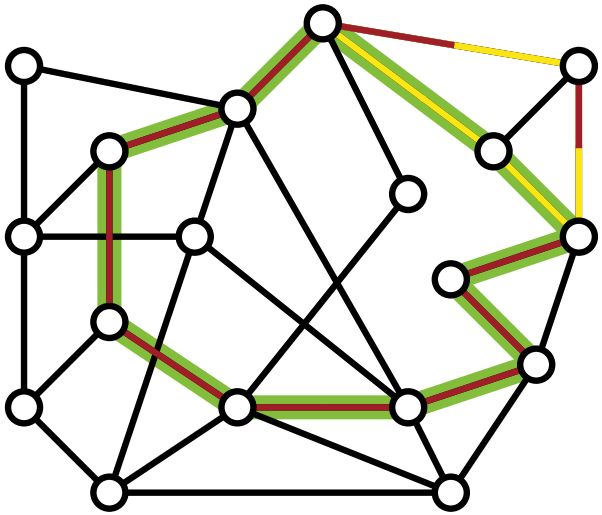


- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

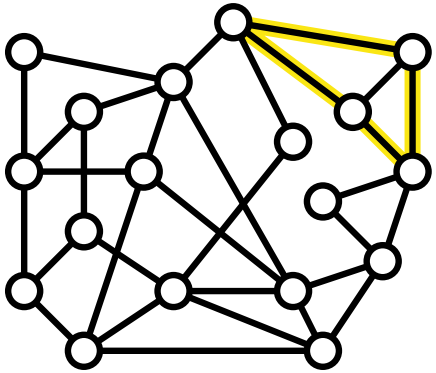


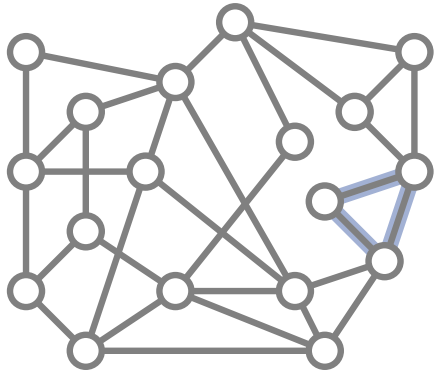


Motivation

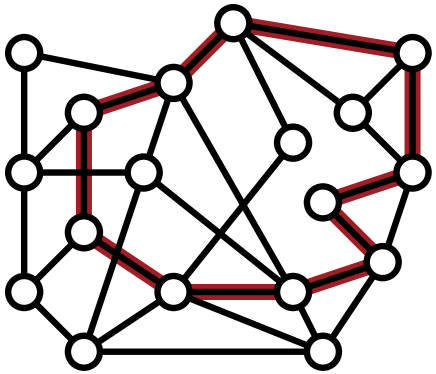


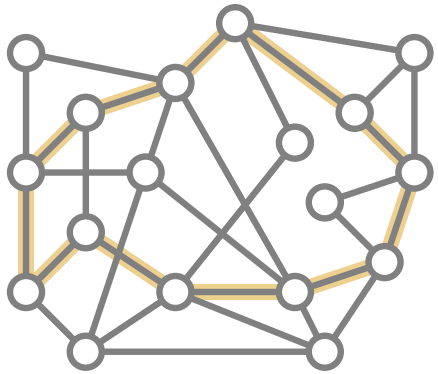
- Ein Graph kann sehr viele Kreise haben.





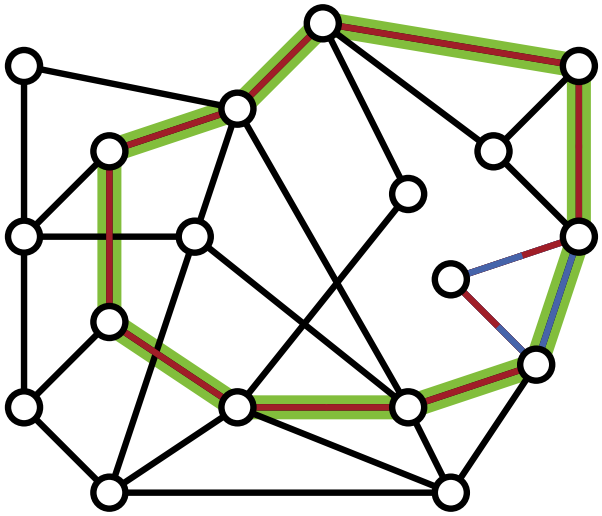
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?



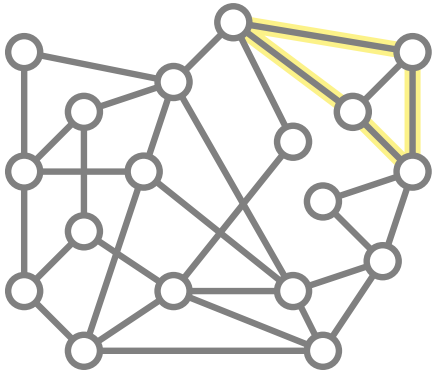


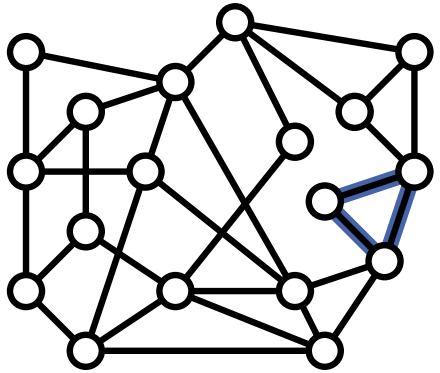
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Motivation

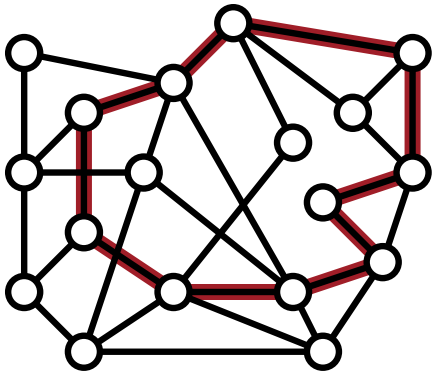


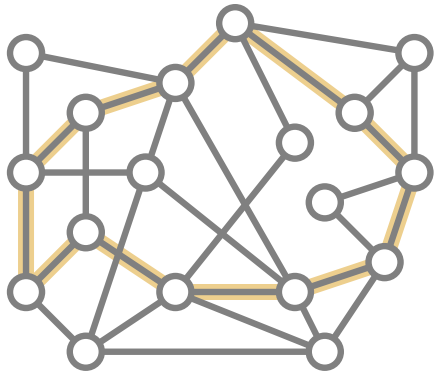
- Ein Graph kann sehr viele Kreise haben.



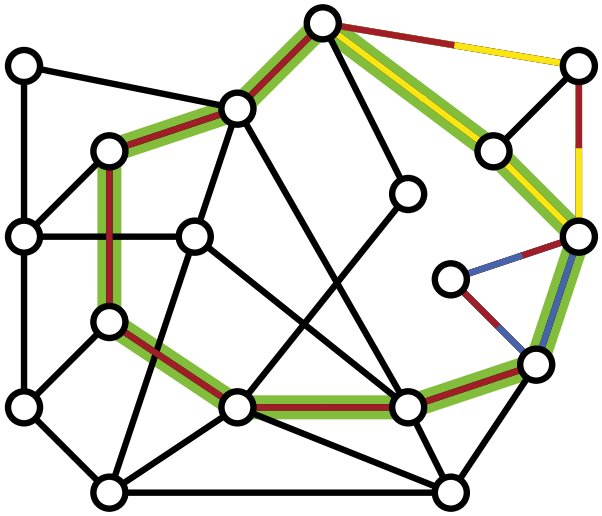


- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

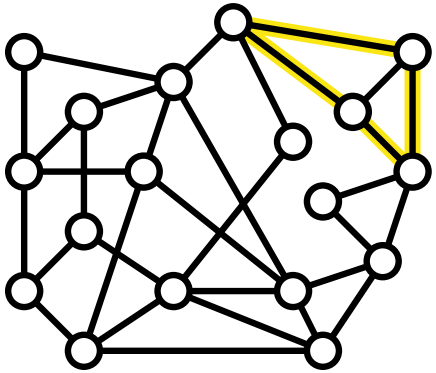


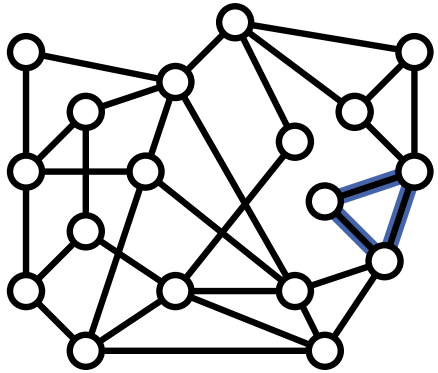


Motivation

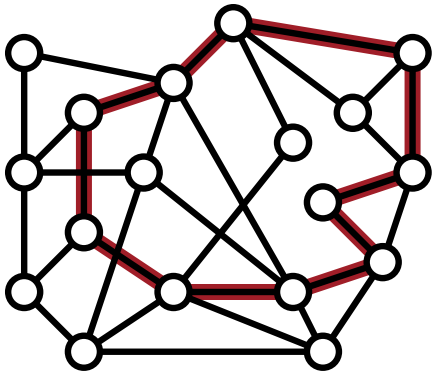


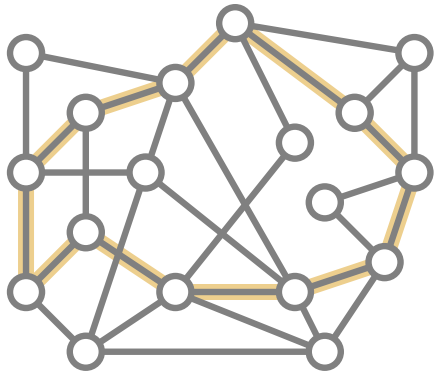
- Ein Graph kann sehr viele Kreise haben.



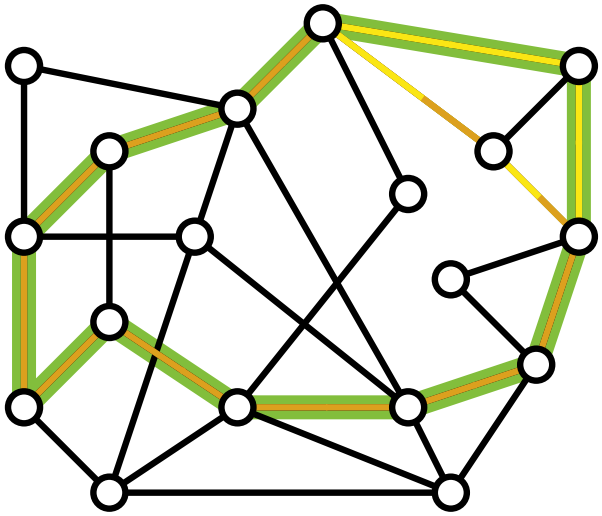


- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

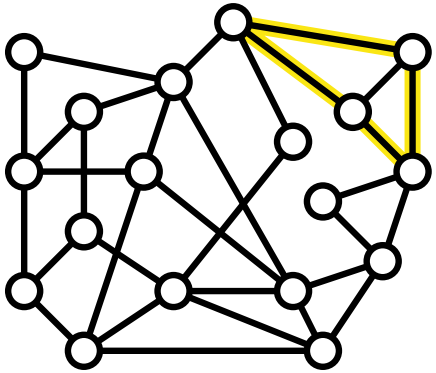


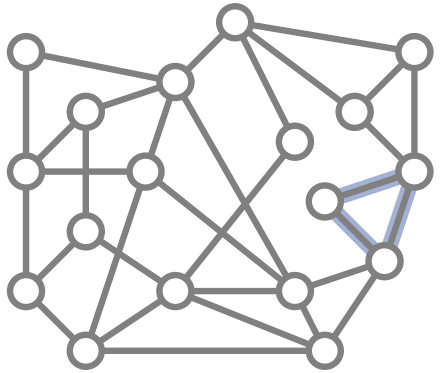


Motivation

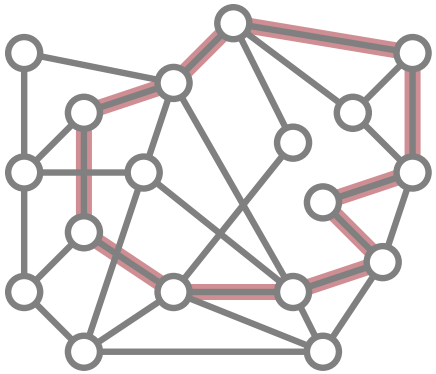


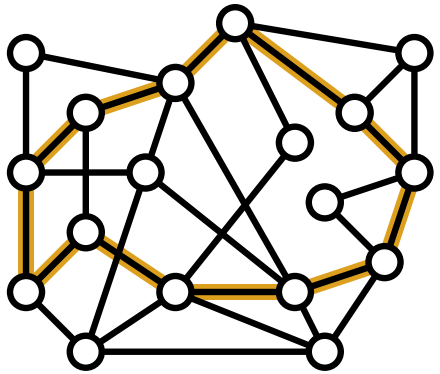
- Ein Graph kann sehr viele Kreise haben.





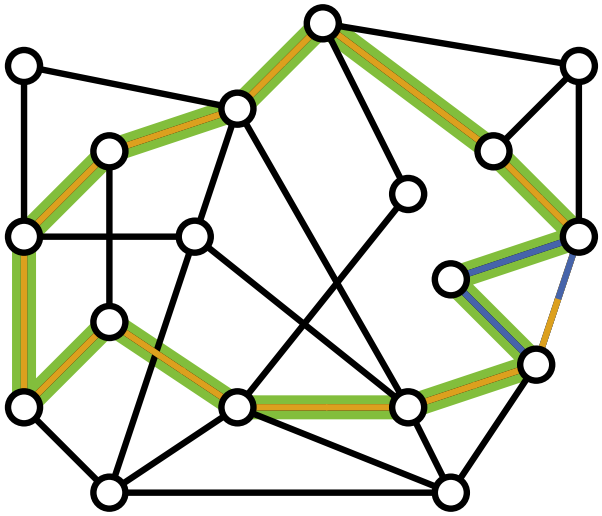
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?



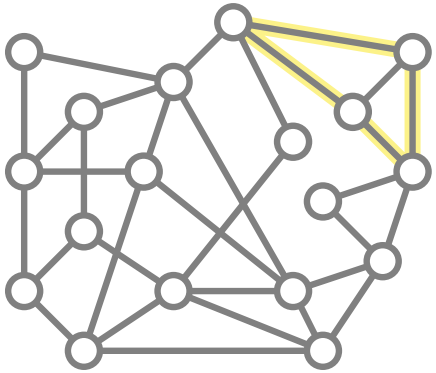


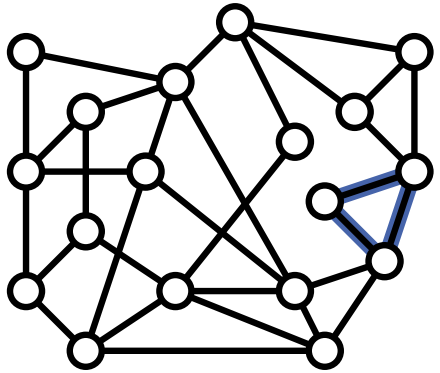
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Motivation

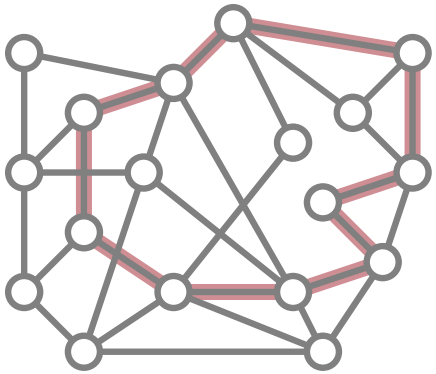


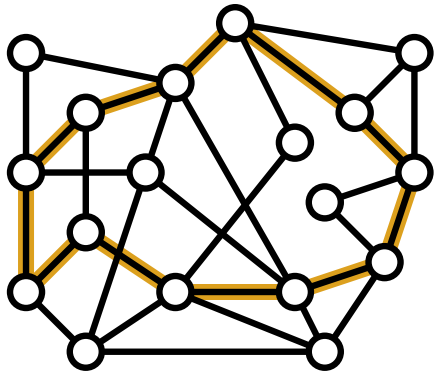
- Ein Graph kann sehr viele Kreise haben.



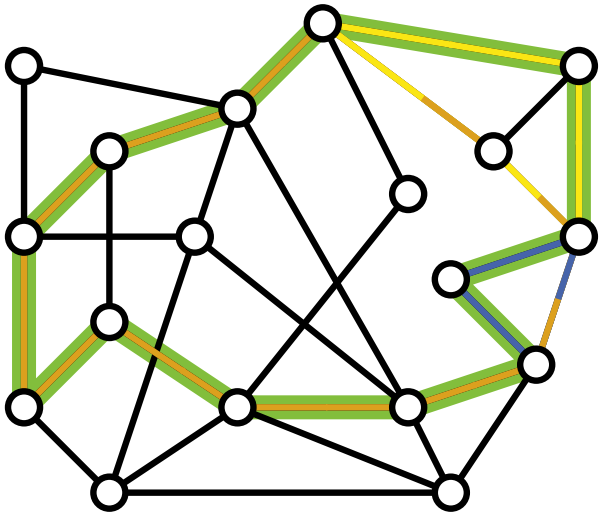


- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

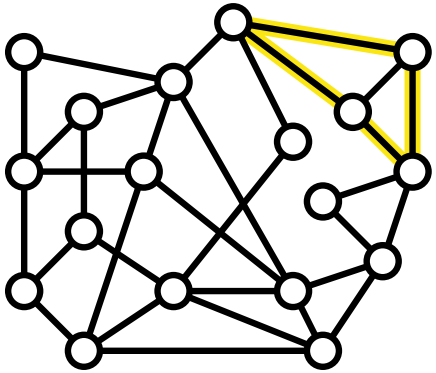


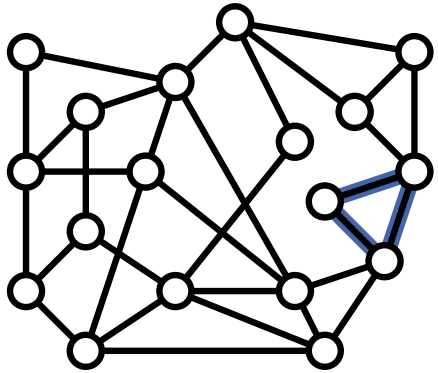


Motivation

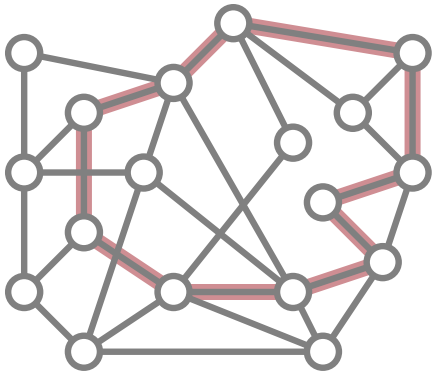


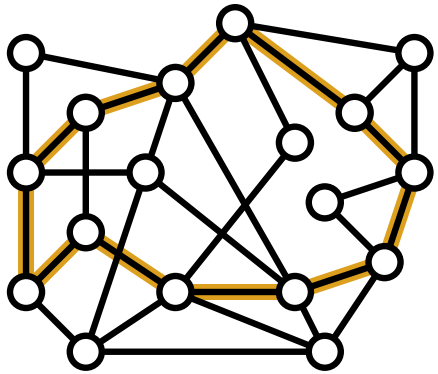
- Ein Graph kann sehr viele Kreise haben.





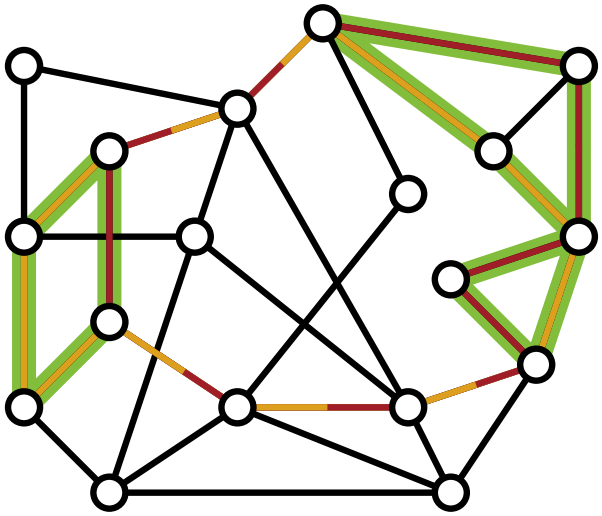
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?



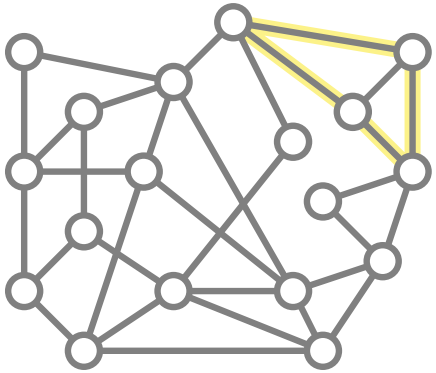


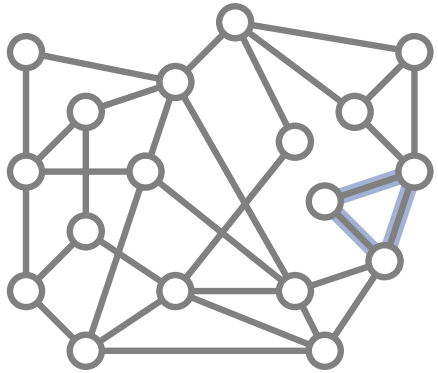
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Motivation

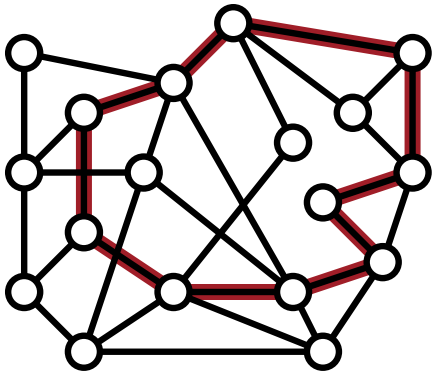


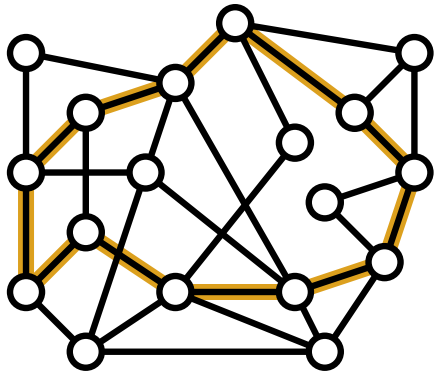
- Ein Graph kann sehr viele Kreise haben.



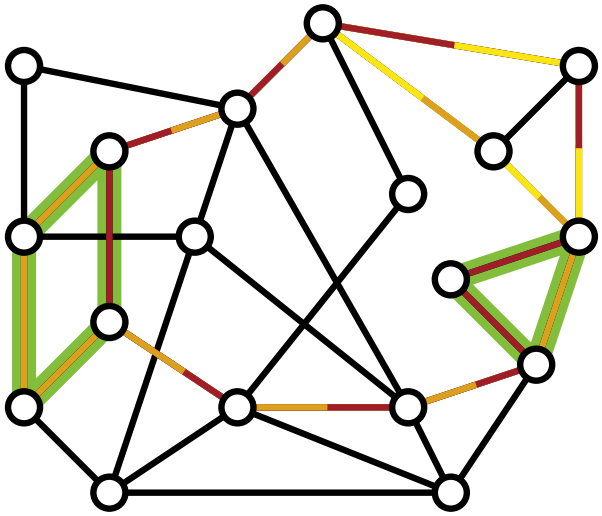


- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

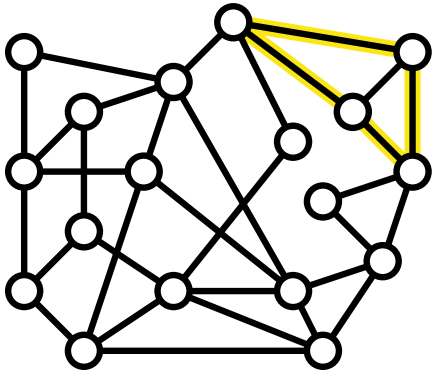


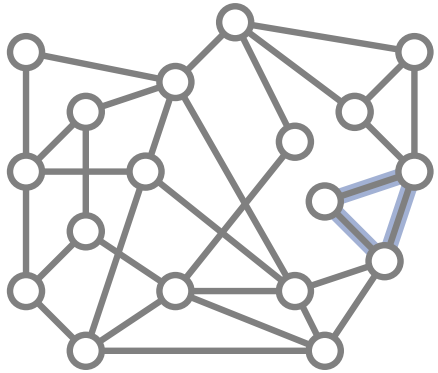


Motivation

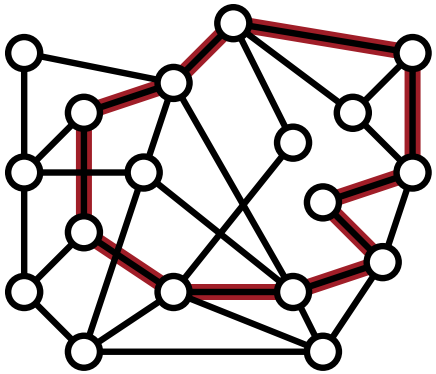


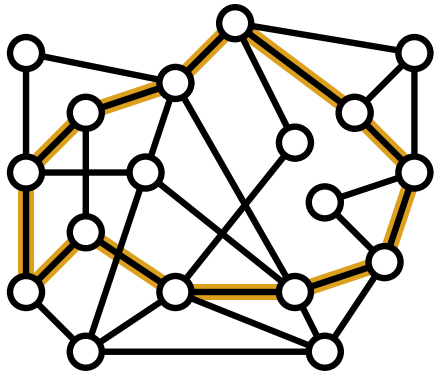
- Ein Graph kann sehr viele Kreise haben.



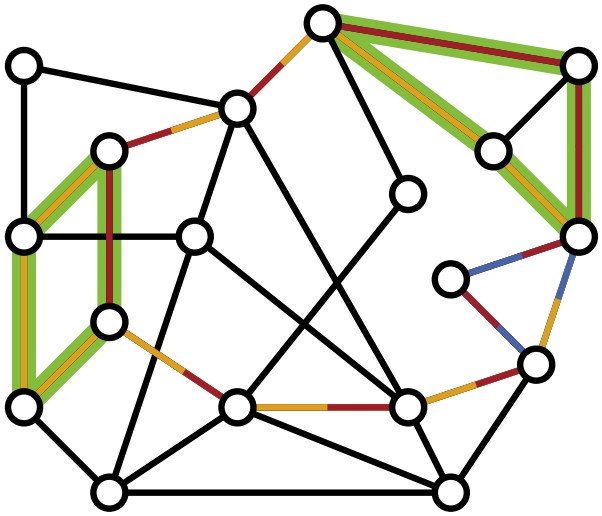


- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

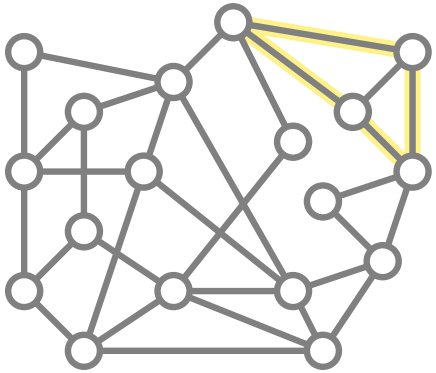


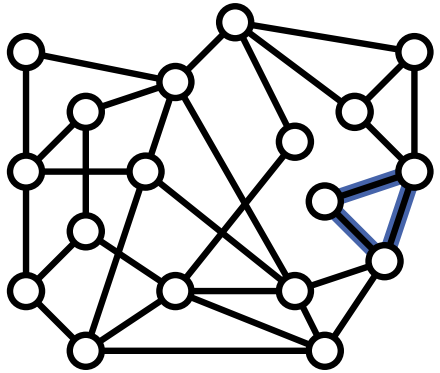


Motivation

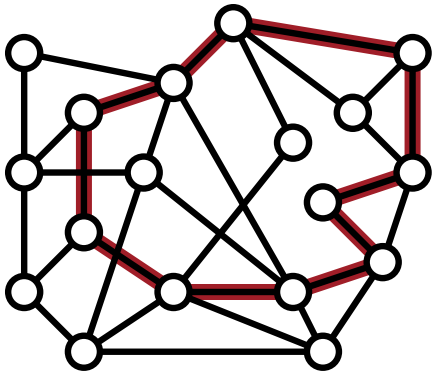


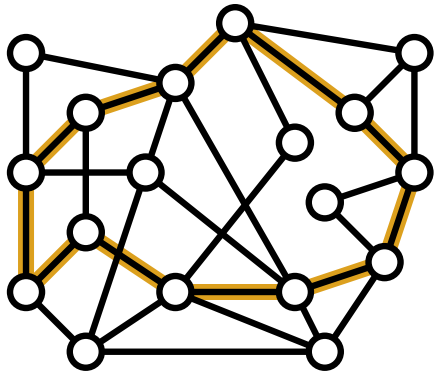
- Ein Graph kann sehr viele Kreise haben.





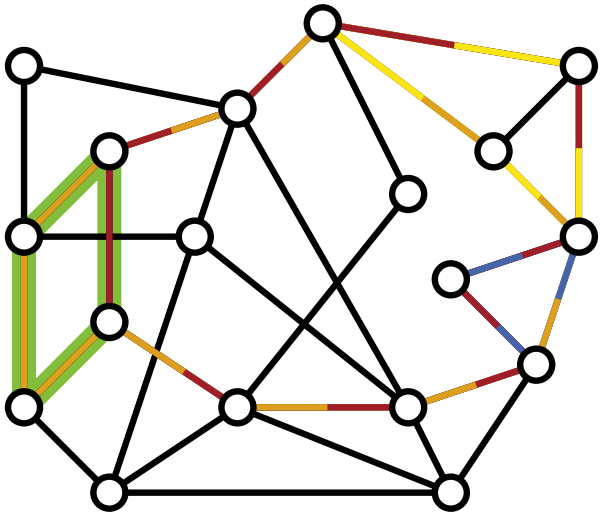
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?



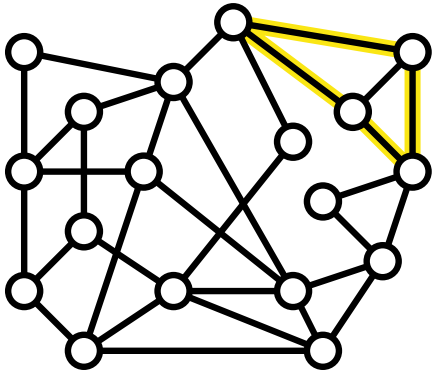


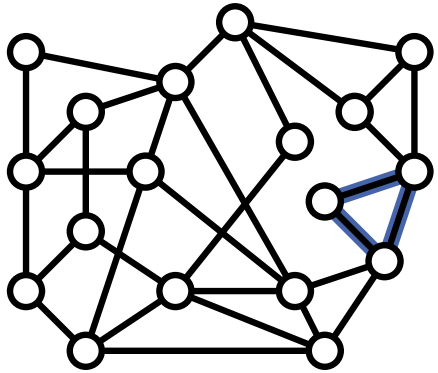
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Motivation

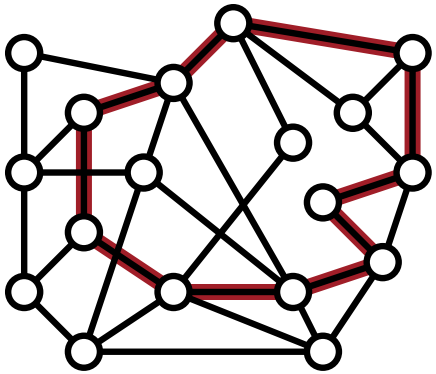


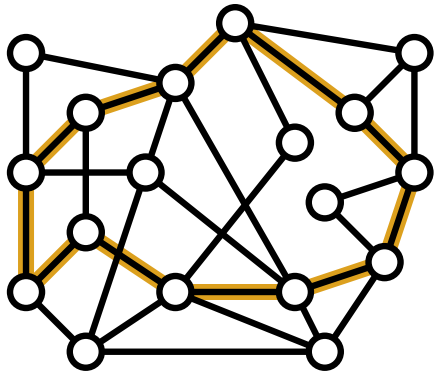
- Ein Graph kann sehr viele Kreise haben.





- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?



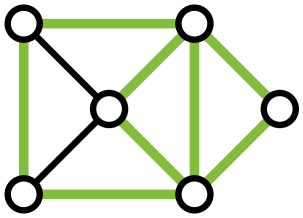


Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

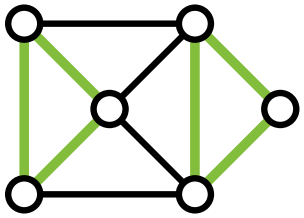
Kreise in Graphen

Definition: Kreis **(Definition 5.1)**

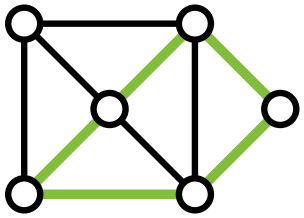
Ein Teilgraph $C = (V_C, E_C)$ von $G = (V, E)$ (d.h. $V_c \subseteq V, E_c \subseteq E$) heißt *Kreis* in G, falls alle Knoten aus V_C in C geraden Grad haben. Falls C zusammenhängend ist und alle Knoten aus V_C Grad zwei haben, so heißt C *einfacher Kreis*.



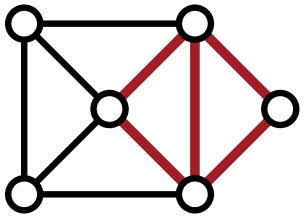
Kreis



Kreis



einfacher Kreis

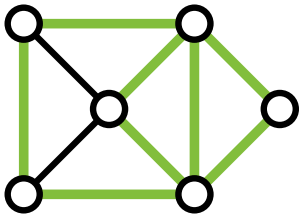


kein Kreis

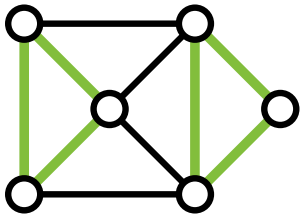
Kreise in Graphen

Definition: Kreis **(Definition 5.1)**

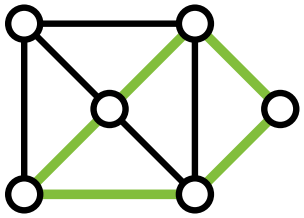
Ein Teilgraph $C = (V_C, E_C)$ von $G = (V, E)$ (d.h. $V_c \subseteq V, E_c \subseteq E$) heißt *Kreis* in G, falls alle Knoten aus V_C in C geraden Grad haben. Falls C zusammenhängend ist und alle Knoten aus V_C Grad zwei haben, so heißt C *einfacher Kreis*.



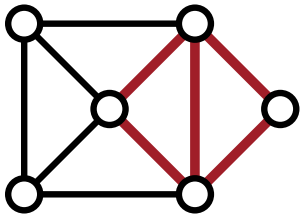
Kreis



Kreis



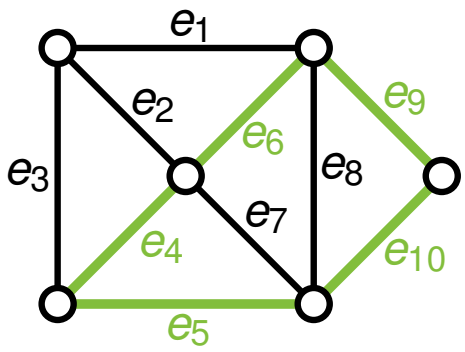
einfacher Kreis



kein Kreis

Fasse Kreis als Kantenmenge $E' \subseteq E = \{e_1, \ldots, e_m\}$ auf und kodiere E' als Vektor $X^{E'}$ mit

$$X_i^{E'} := \begin{cases} 1, & \text{falls } e_i \in E' \\ 0, & \text{sonst} \end{cases}$$



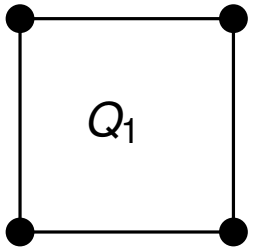
$$X^{E'} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} \begin{matrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \\ e_9 \\ e_{10} \end{matrix}$$

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.

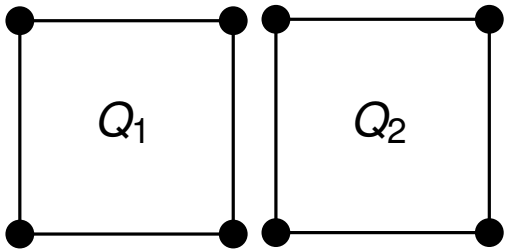
Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



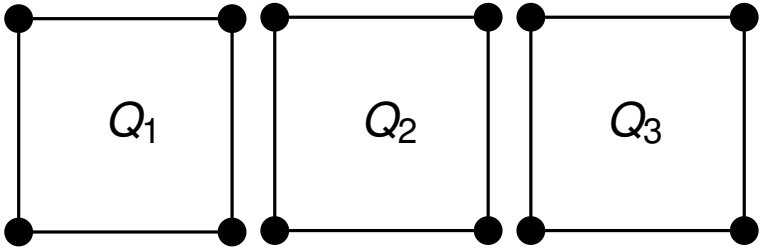
G_1

$|\mathcal{C}_1| = 1$



G_2

$|\mathcal{C}_2| = 3$



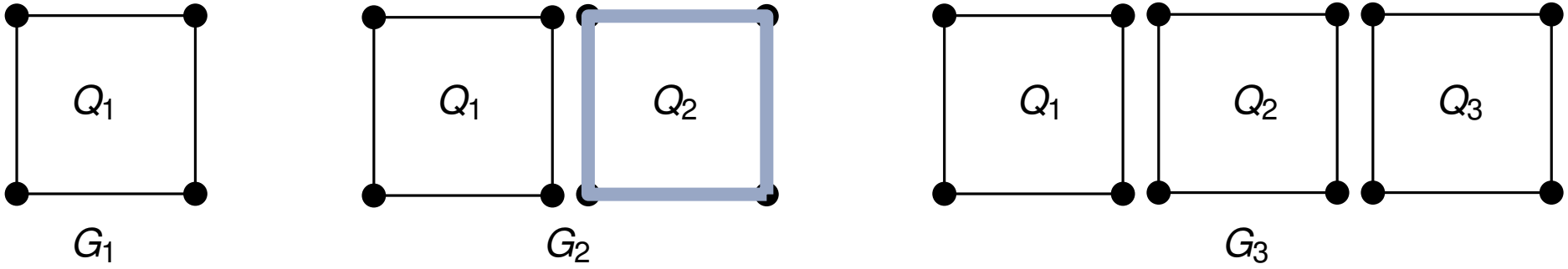
G_3

$|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



G_1 G_2 G_3

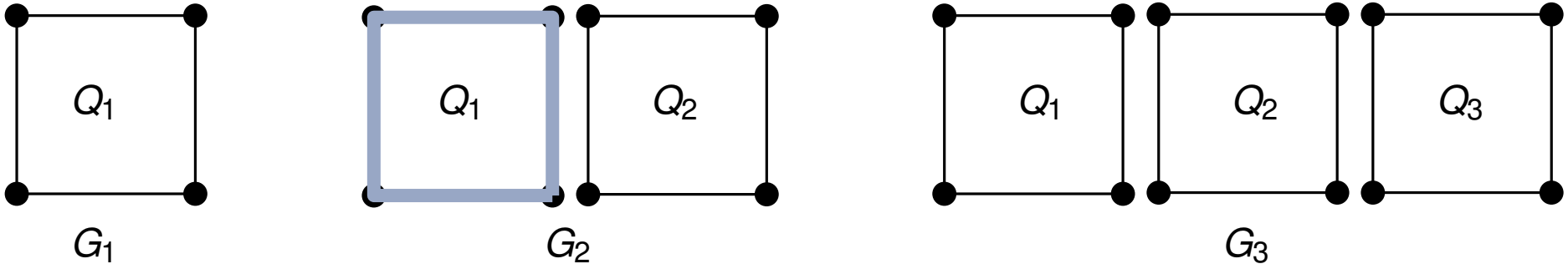
$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



G_1 G_2 G_3

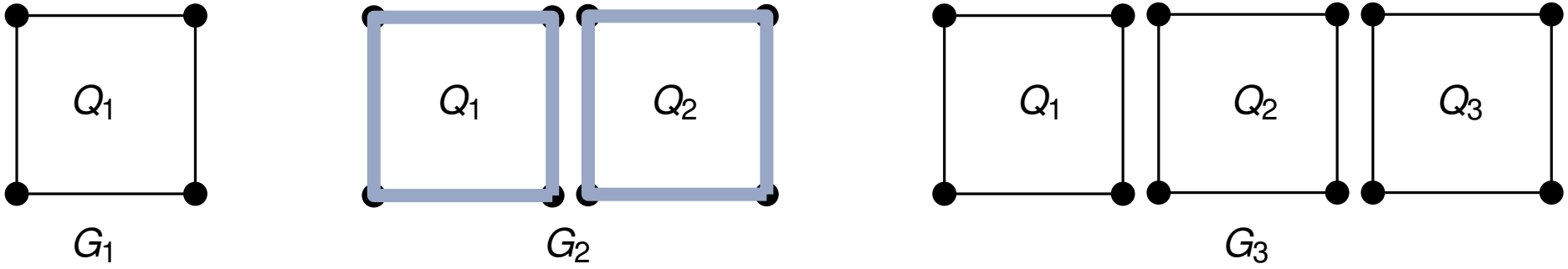
$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



G_1 G_2 G_3

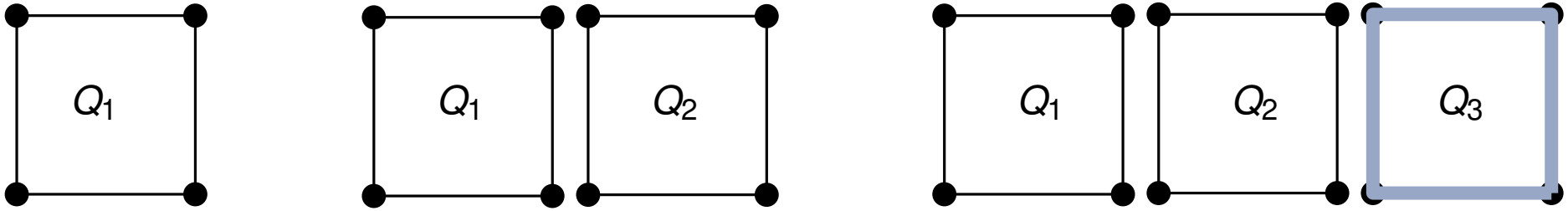
$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



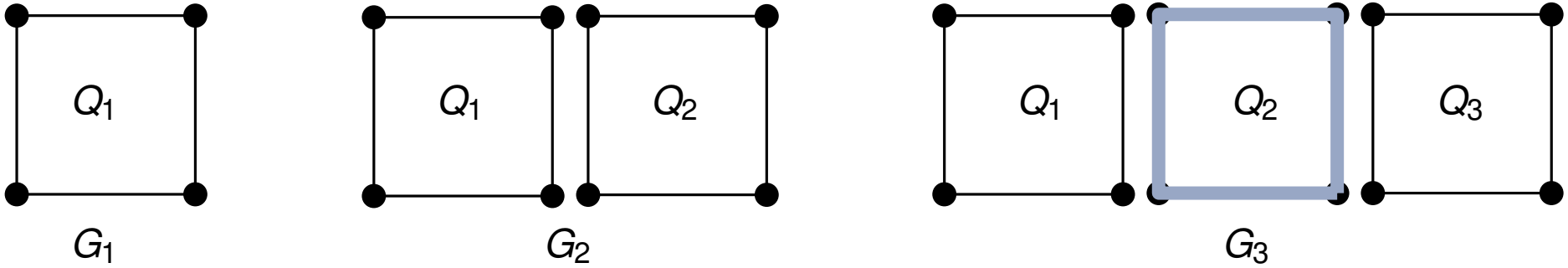
G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



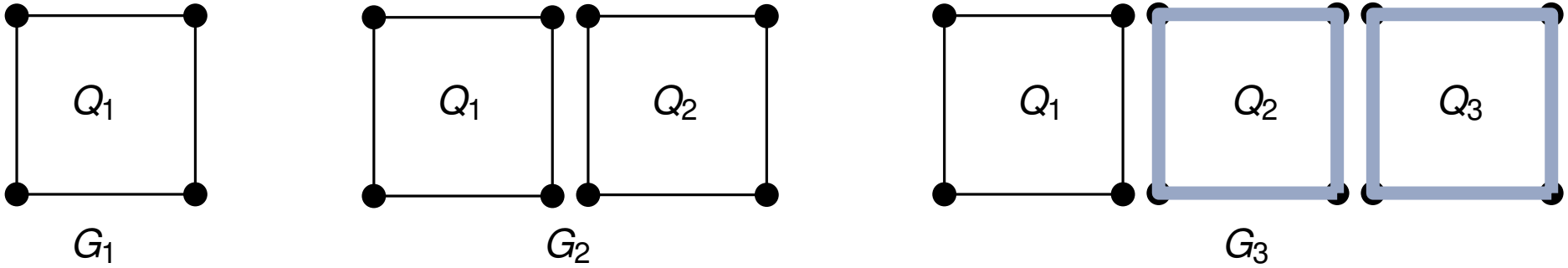
G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



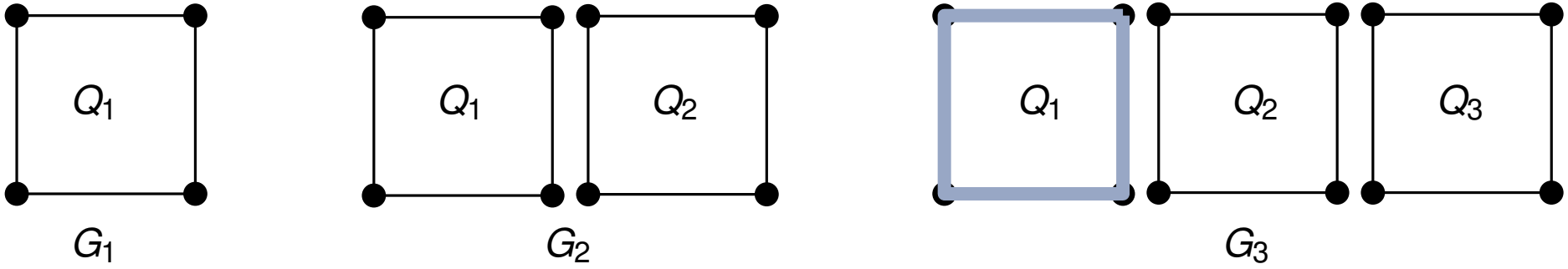
G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



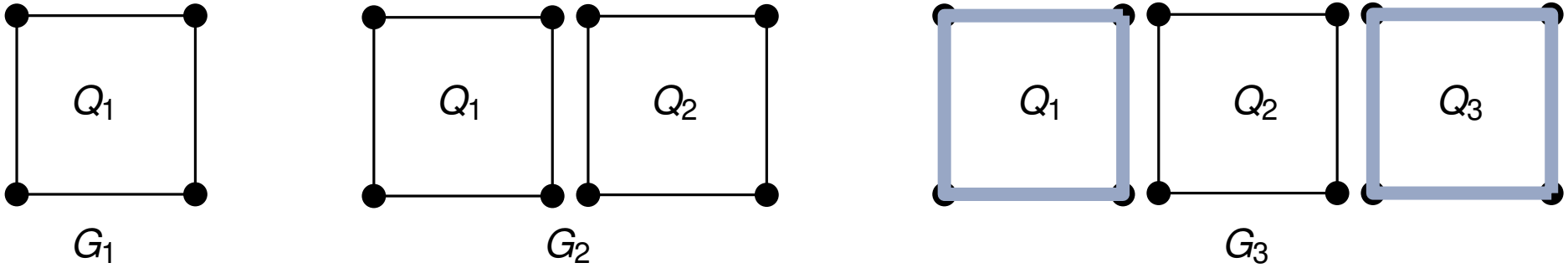
G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



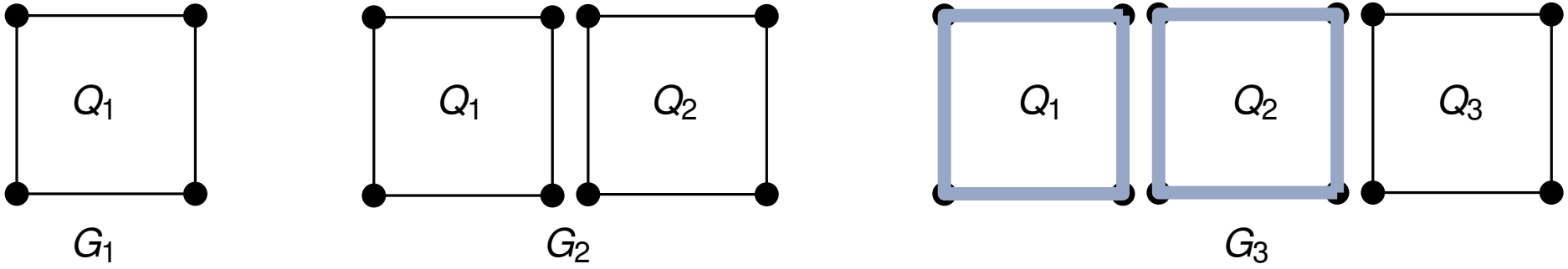
G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



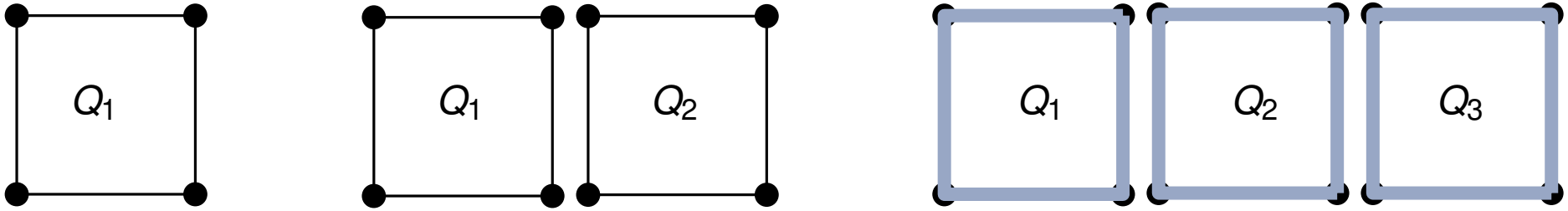
G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



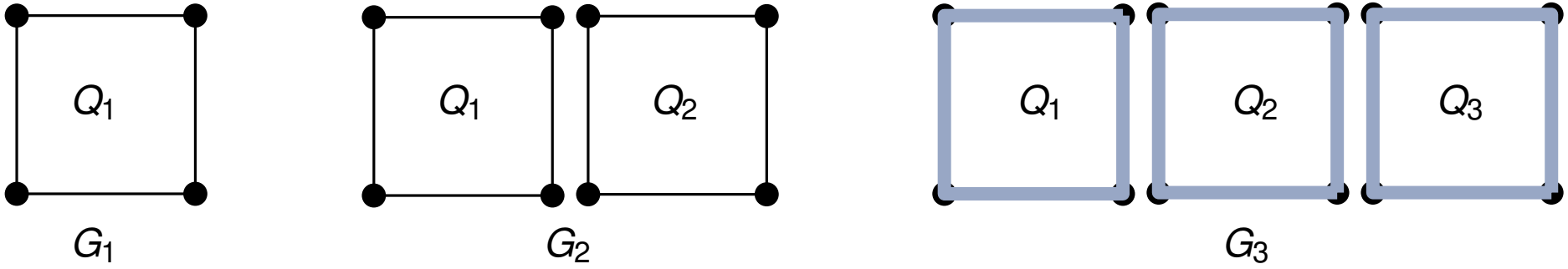
G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

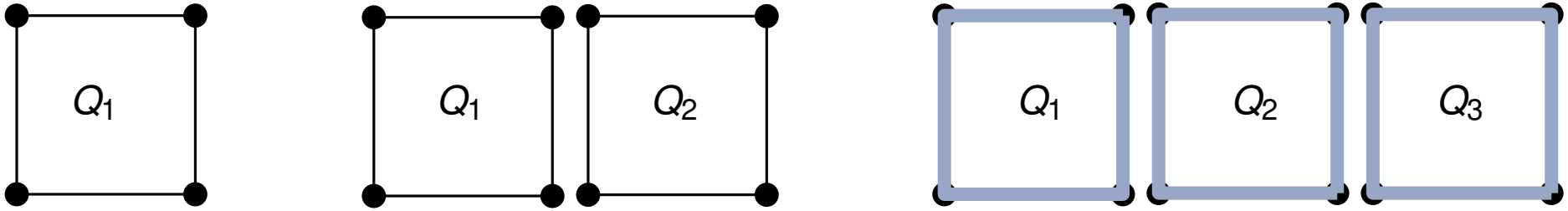
Begründung:

Ein Kreis C in G_i setzt sich aus einer beliebigen Kombination von $Q_1, \ldots, Q_i$ zusammen:

Für alle $j = 1 \ldots, i$ gilt: Entweder ganz Q_j ist in C enthalten oder gar nicht.

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Begründung:

Ein Kreis C in G_i setzt sich aus einer beliebigen Kombination von $Q_1, \ldots, Q_i$ zusammen:

Für alle $j = 1 \ldots, i$ gilt: Entweder ganz Q_j ist in C enthalten oder gar nicht.

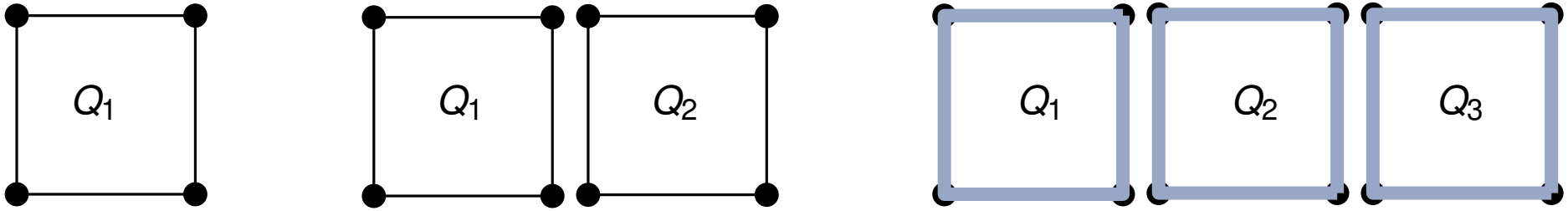
Beschreibe C als binären Vektor $v = (b_1, \ldots, b_i)$:

$$b_j := \begin{cases} 1 & Q_j \text{ ist in } C \text{ enthalten} \\ 0 & \textit{sonst} \end{cases}$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 4

(a) Gesucht: Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **exponentiell** in $|E_i|$.



G_1 G_2 G_3

$|\mathcal{C}_1| = 1$ $|\mathcal{C}_2| = 3$ $|\mathcal{C}_3| = 7$

$G_i = i$ Kopien eines einfachen Kreises Q mit 4 Knoten. (Anzahl Kanten steigt linear)

Begründung:

Ein Kreis C in G_i setzt sich aus einer beliebigen Kombination von $Q_1, \ldots, Q_i$ zusammen:

Für alle $j = 1 \ldots, i$ gilt: Entweder ganz Q_j ist in C enthalten oder gar nicht.

Beschreibe C als binären Vektor $v = (b_1, \ldots, b_i)$:

$$b_j := \begin{cases} 1 & Q_j \text{ ist in } C \text{ enthalten} \\ 0 & \textit{sonst} \end{cases}$$

Es gibt $2^i - 1$ gültige Kombinationen. Nullvektor gehört nicht dazu.

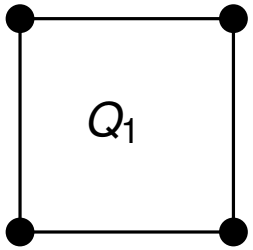
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 4

b) Gesucht ist Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **linear** in $|E_i|$:

Problem 4

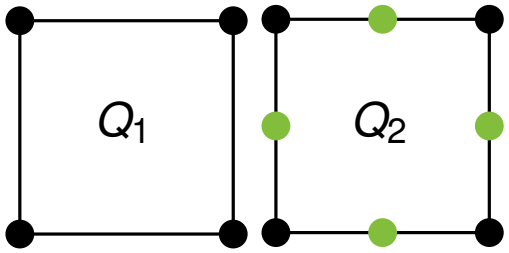
b) Gesucht ist Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **linear** in $|E_i|$:



G_1

$|\mathcal{C}_1| = 1$

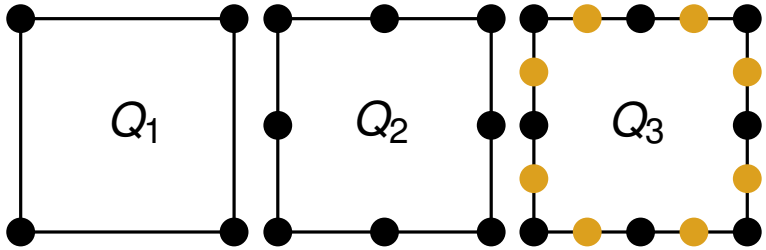
$|E_1| = 4$



G_2

$|\mathcal{C}_2| = 3$

$|E_2| = 4 + 8$



G_3

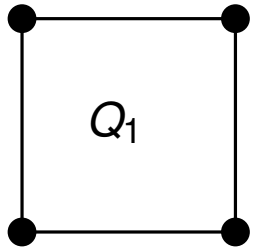
$|\mathcal{C}_3| = 7$

$|E_3| = 4 + 8 + 16$

Familie G_i der 4er-Kreis-Kopien mit Unterteilungen.

Problem 4

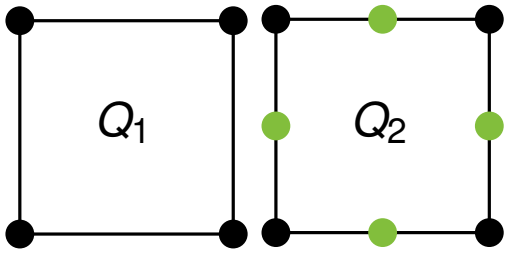
b) Gesucht ist Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **linear** in $|E_i|$:



G_1

$|\mathcal{C}_1| = 1$

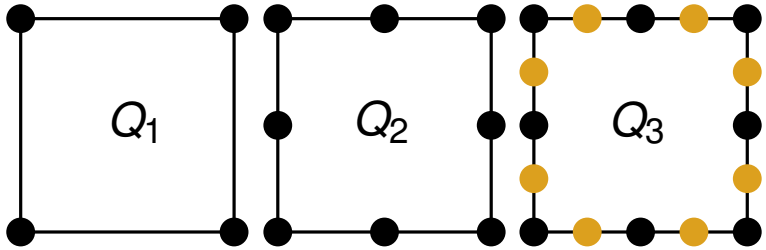
$|E_1| = 4$



G_2

$|\mathcal{C}_2| = 3$

$|E_2| = 4 + 8$



G_3

$|\mathcal{C}_3| = 7$

$|E_3| = 4 + 8 + 16$

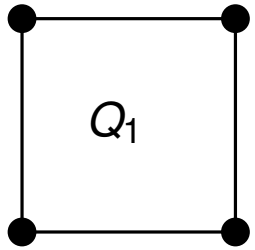
Familie G_i der 4er-Kreis-Kopien mit Unterteilungen.

$$|E_i| = \sum_{j=0}^{i-1} 4 \cdot 2^j = 4 \sum_{j=0}^{i-1} 2^j = 4(2^i - 1) = 4 \cdot 2^i - 4$$

(Verwende geometrische Reihe.)

Problem 4

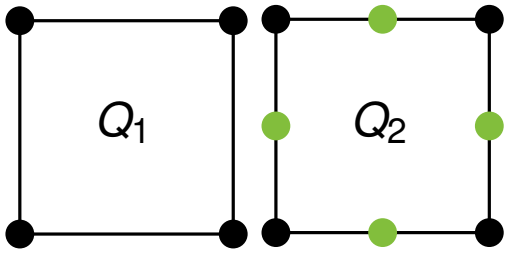
b) Gesucht ist Familie $(G_i)_{i \in \mathbb{N}}$ mit $|\mathcal{C}_i|$ **linear** in $|E_i|$:



G_1

$|\mathcal{C}_1| = 1$

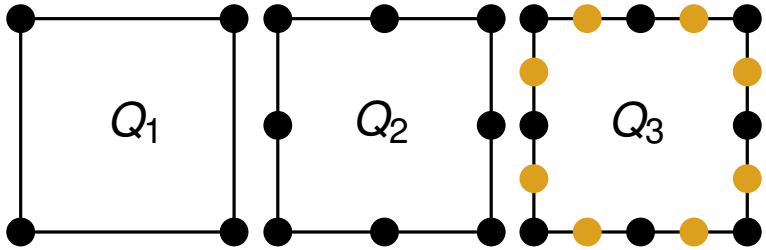
$|E_1| = 4$



G_2

$|\mathcal{C}_2| = 3$

$|E_2| = 4 + 8$



G_3

$|\mathcal{C}_3| = 7$

$|E_3| = 4 + 8 + 16$

Familie G_i der 4er-Kreis-Kopien mit Unterteilungen.

$$|E_i| = \sum_{j=0}^{i-1} 4 \cdot 2^j = 4 \sum_{j=0}^{i-1} 2^j = 4(2^i - 1) = 4 \cdot 2^i - 4$$

(Verwende geometrische Reihe.)

Ein Kreis C in G_i setzt sich aus einer beliebigen Kombination von $Q_1, \ldots, Q_i$ zusammen:

Entweder ganz Q_i ist in C enthalten oder gar nicht.

Deshalb: $|\mathcal{C}_i| = 2^i - 1$

Kreisraum

Definition: Kreisraum
Sei $\mathcal{C}$ die Menge aller Kreise in $G = (V, E)$. Dann induziert $\mathcal{C}$ den Vektorraum der Vektoren X^c, $c \in \mathcal{C}$ über dem Körper GF(2), genannt *Kreisraum* von G.

Erinnerung: $GF(2)$ ist der Körper mit zwei Elementen $\{0, 1\}$ und den Verknüpfungen + und $\cdot$ mit

+	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

$a_1 + \ldots + a_k = 1 \Leftrightarrow$ **ungerade** Anzahl a_i auf 1 gesetzt.
$a_1 + \ldots + a_k = 0 \Leftrightarrow$ **gerade** Anzahl a_i auf 1 gesetzt.

Kreisraum

Definition: Kreisraum

Sei $\mathcal{C}$ die Menge aller Kreise in $G = (V, E)$. Dann induziert $\mathcal{C}$ den Vektorraum der Vektoren X^c, $c \in \mathcal{C}$ über dem Körper GF(2), genannt *Kreisraum* von G.

Erinnerung: $GF(2)$ ist der Körper mit zwei Elementen $\{0, 1\}$ und den Verknüpfungen + und · mit

+	0	1
0	0	1
1	1	0

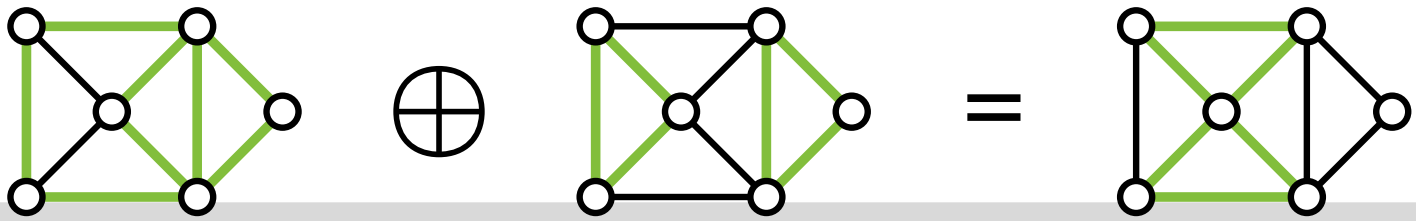
·	0	1
0	0	0
1	0	1

$a_1 + \ldots + a_k = 1 \Leftrightarrow$ **ungerade** Anzahl a_i auf 1 gesetzt.

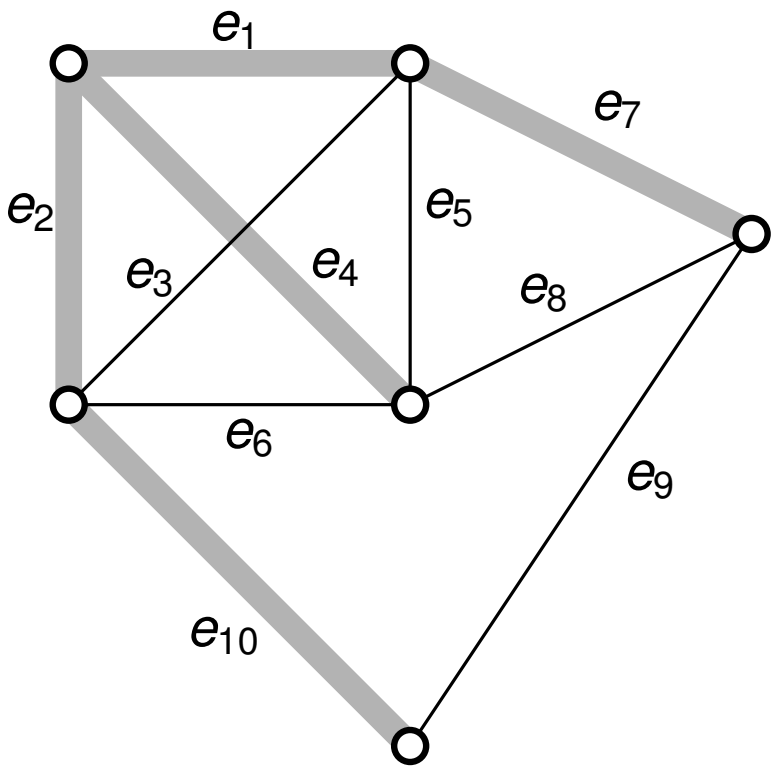
$a_1 + \ldots + a_k = 0 \Leftrightarrow$ **gerade** Anzahl a_i auf 1 gesetzt.

Definition: Summe von Kreisen – symmetrische Differenz

Die Addition im Kreisraum von G induziert eine Operation $\oplus$ auf C durch $c_1 \oplus c_2 = (E_{c_1} \cup E_{c_2}) \setminus (E_{c_1} \cap E_{c_2})$. Dies ist die *symmetrische Differenz* beider Kantenmengen.

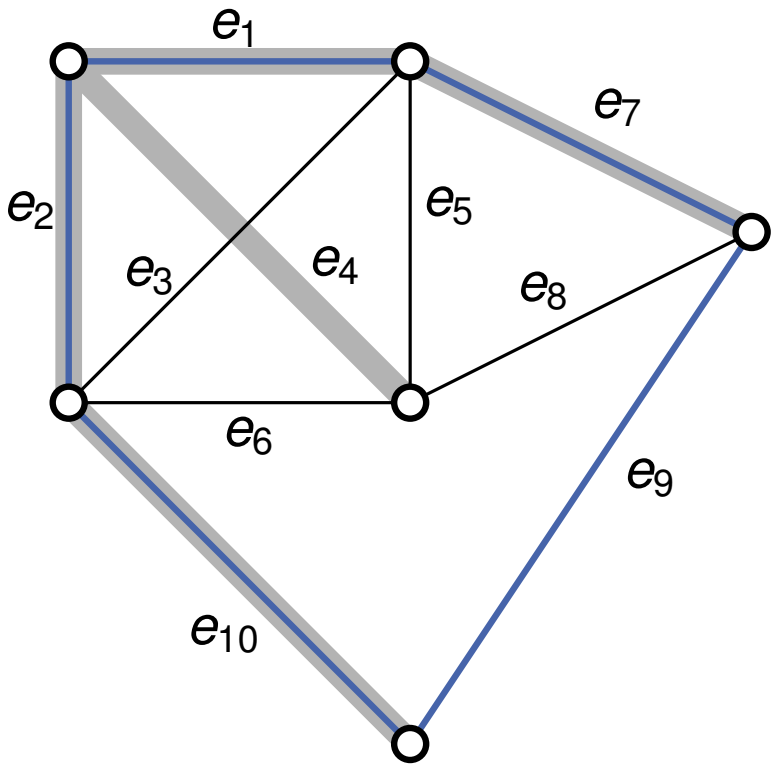


Fundamentalbasis



Spannbaum T:

Fundamentalbasis

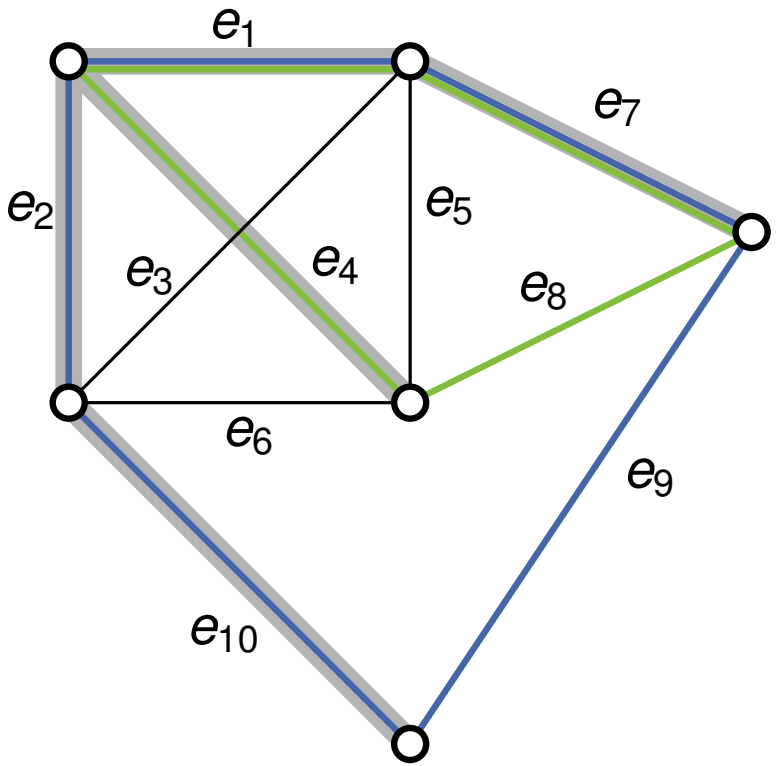


Spannbaum T:

Kante e_9 induziert Kreis:

$C_1 = e_1 - e_7 - e_9 - e_{10} - e_2$

Fundamentalbasis



Spannbaum T:

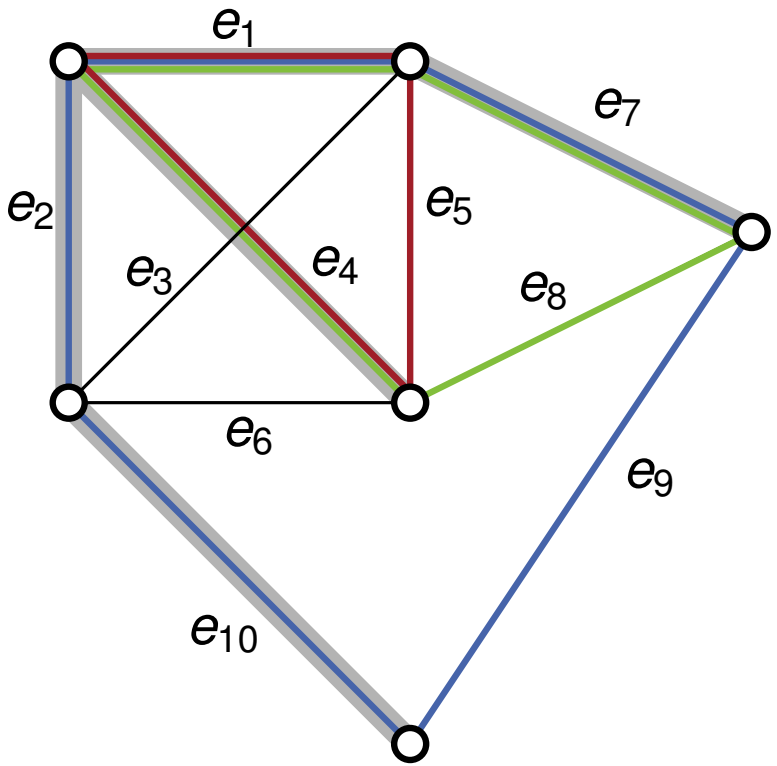
Kante e_9 induziert Kreis:

$C_1 = e_1 - e_7 - e_9 - e_{10} - e_2$

Kante e_8 induziert Kreis:

$C_2 = e_1 - e_7 - e_8 - e_4$

Fundamentalbasis



Spannbaum T:

Kante e_9 induziert Kreis:

$C_1 = e_1 - e_7 - e_9 - e_{10} - e_2$

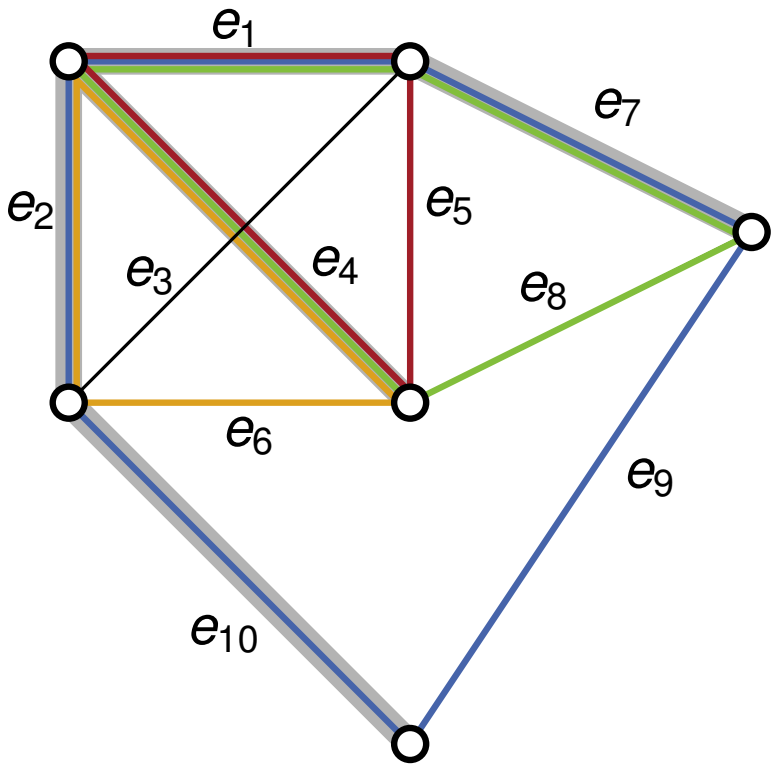
Kante e_8 induziert Kreis:

$C_2 = e_1 - e_7 - e_8 - e_4$

Kante e_5 induziert Kreis:

$C_3 = e_1 - e_5 - e_4$

Fundamentalbasis



Spannbaum T:

Kante e_9 induziert Kreis:

$C_1 = e_1 - e_7 - e_9 - e_{10} - e_2$

Kante e_8 induziert Kreis:

$C_2 = e_1 - e_7 - e_8 - e_4$

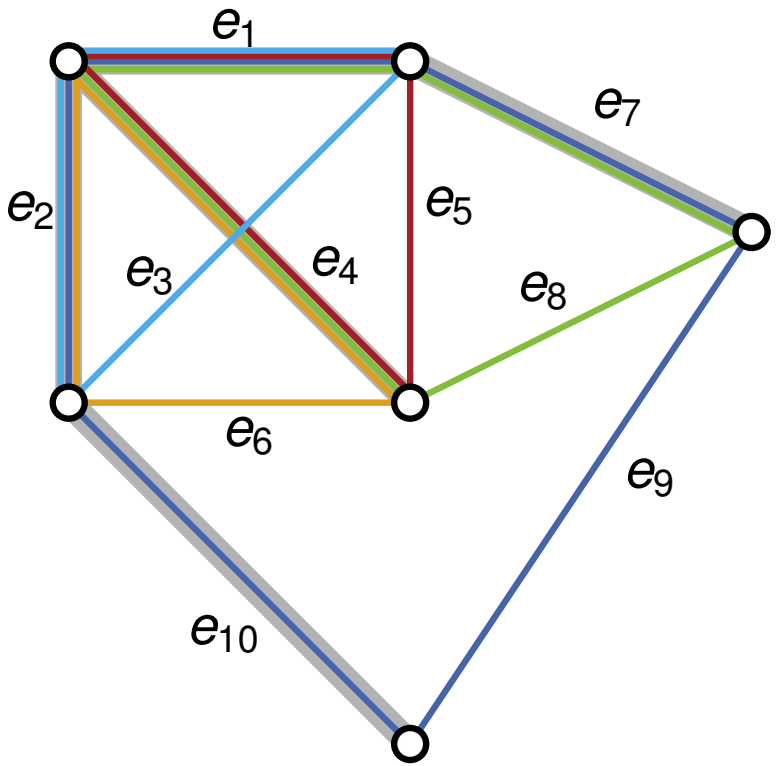
Kante e_5 induziert Kreis:

$C_3 = e_1 - e_5 - e_4$

Kante e_6 induziert Kreis:

$C_4 = e_2 - e_4 - e_6$

Fundamentalbasis



Spannbaum T:

Kante e_9 induziert Kreis:

$C_1 = e_1 - e_7 - e_9 - e_{10} - e_2$

Kante e_8 induziert Kreis:

$C_2 = e_1 - e_7 - e_8 - e_4$

Kante e_5 induziert Kreis:

$C_3 = e_1 - e_5 - e_4$

Kante e_6 induziert Kreis:

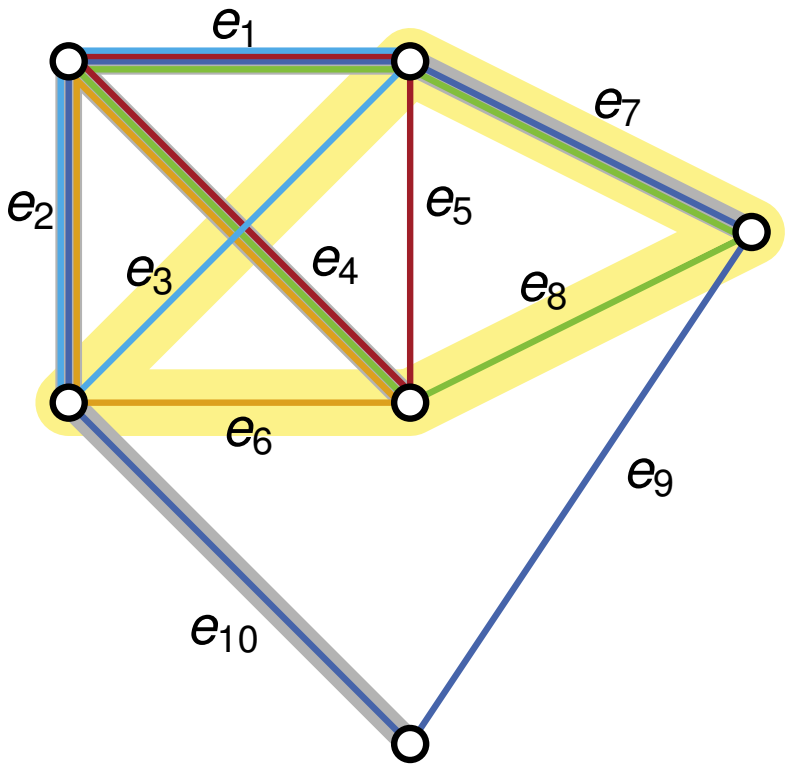
$C_4 = e_2 - e_4 - e_6$

Kante e_3 induziert Kreis:

$C_5 = e_1 - e_3 - e_2$

C_1, C_2, C_3, C_4 und C_5 bilden Fundamentalbasis.

Fundamentalbasis



Spannbaum T:

Kante e_9 induziert Kreis:

$C_1 = e_1 - e_7 - e_9 - e_{10} - e_2$

Kante e_8 induziert Kreis:

$C_2 = e_1 - e_7 - e_8 - e_4$

Kante e_5 induziert Kreis:

$C_3 = e_1 - e_5 - e_4$

Kante e_6 induziert Kreis:

$C_4 = e_2 - e_4 - e_6$

Kante e_3 induziert Kreis:

$C_5 = e_1 - e_3 - e_2$

C_1, C_2, C_3, C_4 und C_5 bilden Fundamentalbasis.

Darstellung anderer Kreise bezüglich der gewählten Basis:

$e_3 - e_7 - e_8 - e_6 = e_1 - e_7 - e_8 - e_4 \oplus e_2 - e_4 - e_6 \oplus e_1 - e_3 - e_2$

2. Problem

Gegeben:

- ungerichteter, zusammenhängender Graph $G = (V, E)$
- aufspannender Baum $T = (V, E_T)$ in G

Fundamentalbasis B_t des Kreisraumes $\mathcal{C}$ von G definiert als

$$B_T := \{C_e \mid e \in E \setminus E_T,\ C_e \in \mathcal{C},\ \textit{mit}$$

$$E_{C_e} = \{e = \{u, v\}\} \cup \{\text{Pfadkanten von } u \text{ nach } v \text{ in } T\}\}$$

2. Problem

Gegeben:

- ungerichteter, zusammenhängender Graph $G = (V, E)$
- aufspannender Baum $T = (V, E_T)$ in G

Fundamentalbasis B_t des Kreisraumes $\mathcal{C}$ von G definiert als

$$B_T := \{C_e \mid e \in E \setminus E_T,\ C_e \in \mathcal{C},\ \textit{mit}$$
$$E_{C_e} = \{e = \{u, v\}\} \cup \{\text{Pfadkanten von } u \text{ nach } v \text{ in } T\}\}$$

1. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ linear unabhängig ist.

Wiederholung: Eine Teilmenge $B = \{b_1, \ldots, b_n\} \subseteq V$ eines K-Vektorraums V heißt *linear unabhängig*, wenn gilt:

$$\sum_{i=1}^{i=n} a_i b_i = 0,\ a_i \in K \quad \Longleftrightarrow \quad a_i = 0 \quad \forall\, i,$$

d.h., der Nullvektor lässt sich nur als triviale Linearkombination der Vektoren in B schreiben.

2. Problem

Gegeben:

- ungerichteter, zusammenhängender Graph $G = (V, E)$
- aufspannender Baum $T = (V, E_T)$ in G

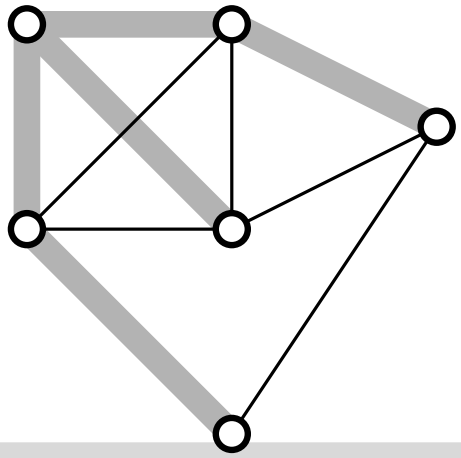
Fundamentalbasis B_t des Kreisraumes $\mathcal{C}$ von G definiert als

$$B_T := \{C_e \mid e \in E \setminus E_T,\ C_e \in \mathcal{C},\ mit$$

$$E_{C_e} = \{e = \{u, v\}\} \cup \{\text{Pfadkanten von } u \text{ nach } v \text{ in } T\}\}$$

1. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ linear unabhängig ist.

- Nach Definition von B_T: Jede Nichtbaumkante nur in einem Kreis in B_T enthalten.



2. Problem

Gegeben:

- ungerichteter, zusammenhängender Graph $G = (V, E)$
- aufspannender Baum $T = (V, E_T)$ in G

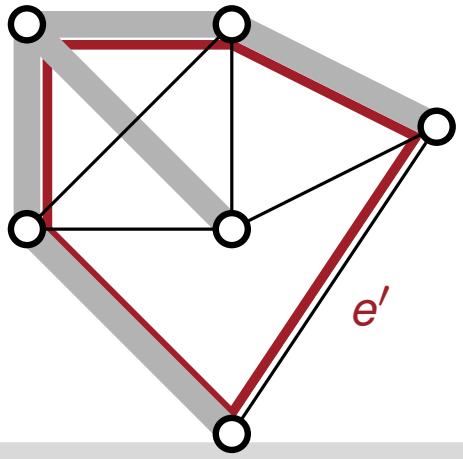
Fundamentalbasis B_t des Kreisraumes $\mathcal{C}$ von G definiert als

$$B_T := \{C_e \mid e \in E \setminus E_T,\ C_e \in \mathcal{C},\ mit$$

$$E_{C_e} = \{e = \{u, v\}\} \cup \{\text{Pfadkanten von } u \text{ nach } v \text{ in } T\}\}$$

1. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ linear unabhängig ist.

- Nach Definition von B_T: Jede Nichtbaumkante nur in einem Kreis in B_T enthalten.
- **Annahme:** Sei $\sum_{e \in E \setminus E_T} a_e C_e = 0$, mit $a_{e'} \neq 0$, für mindestens ein e'.



2. Problem

Gegeben:

- ungerichteter, zusammenhängender Graph $G = (V, E)$
- aufspannender Baum $T = (V, E_T)$ in G

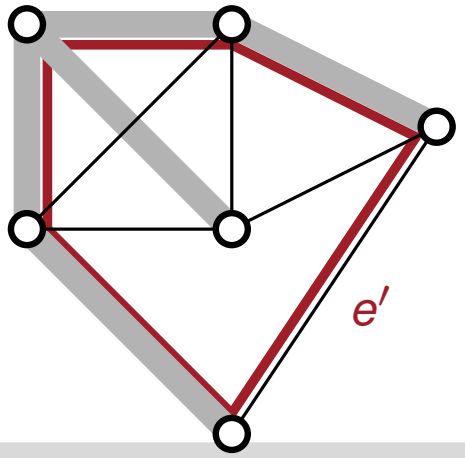
Fundamentalbasis B_t des Kreisraumes $\mathcal{C}$ von G definiert als

$$B_T := \{C_e \mid e \in E \setminus E_T,\ C_e \in \mathcal{C},\ mit$$

$$E_{C_e} = \{e = \{u, v\}\} \cup \{\text{Pfadkanten von } u \text{ nach } v \text{ in } T\}\}$$

1. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ linear unabhängig ist.

- Nach Definition von B_T: Jede Nichtbaumkante nur in einem Kreis in B_T enthalten.
- **Annahme:** Sei $\sum_{e \in E \setminus E_T} a_e C_e = 0$, mit $a_{e'} \neq 0$, für mindestens ein e'.
- $C_{e'}$ enthält die Nichtbaumkante e', die in keinem anderen Kreis aus B_T enthalten ist .
- Kann nicht zu Null summiert werden (sym. Differenz)



2. Problem

Gegeben:

- ungerichteter, zusammenhängender Graph $G = (V, E)$
- aufspannender Baum $T = (V, E_T)$ in G

Fundamentalbasis B_t des Kreisraumes $\mathcal{C}$ von G definiert als

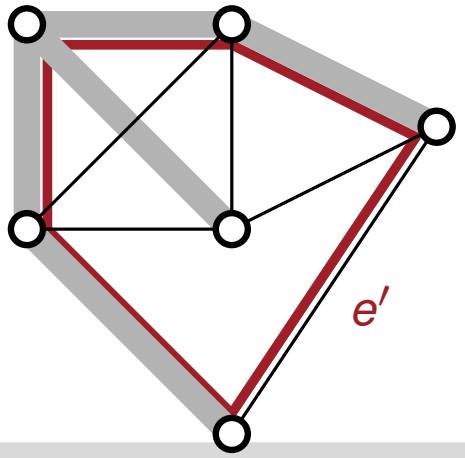
$$B_T := \{C_e \mid e \in E \setminus E_T,\ C_e \in \mathcal{C},\ mit$$

$$E_{C_e} = \{e = \{u, v\}\} \cup \{\text{Pfadkanten von } u \text{ nach } v \text{ in } T\}\}$$

1. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ linear unabhängig ist.

- Nach Definition von B_T: Jede Nichtbaumkante nur in einem Kreis in B_T enthalten.
- **Annahme:** Sei $\sum_{e \in E \setminus E_T} a_e C_e = 0$, mit $a_{e'} \neq 0$, für mindestens ein e'.
- $C_{e'}$ enthält die Nichtbaumkante e', die in keinem anderen Kreis aus B_T enthalten ist .
- Kann nicht zu Null summiert werden (sym. Differenz)

Annahme ist widerlegt.



2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

2. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ ein Erzeugendensystem von $\mathcal{C}$ ist.

Hinweis:

Gehen Sie dabei konstruktiv vor und beschreiben Sie, wie ein beliebiger Kreis durch eine Linearkombination von Elementen aus B_T gebildet werden kann.

2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

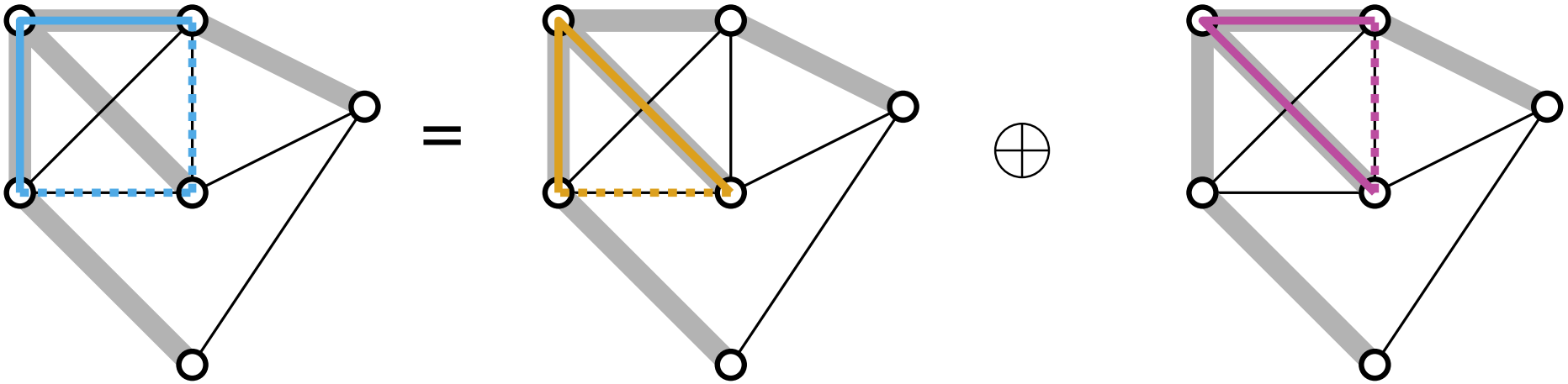
2. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ ein Erzeugendensystem von $\mathcal{C}$ ist.

Hinweis:

Gehen Sie dabei konstruktiv vor und beschreiben Sie, wie ein beliebiger Kreis durch eine Linearkombination von Elementen aus B_T gebildet werden kann.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

Wdh.: C_e ist eindeutiger Kreis in gegebener Fundamentalbasis B_T über Nichtbaumkante e.



2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

2. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ ein Erzeugendensystem von $\mathcal{C}$ ist.

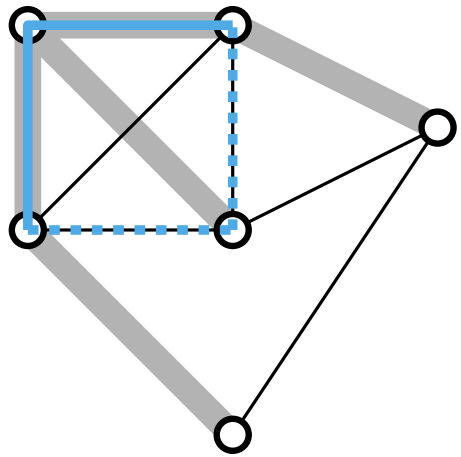
Hinweis:
Gehen Sie dabei konstruktiv vor und beschreiben Sie, wie ein beliebiger Kreis durch eine Linearkombination von Elementen aus B_T gebildet werden kann.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

Wdh.: C_e ist eindeutiger Kreis in gegebener Fundamentalbasis B_T über Nichtbaumkante e.

Betrachte Linearkombination $\sum_{e \in E_C \setminus E_T} C_e$ und zeige für beliebige Kante $e \in E$:

- Falls $e \in E_C$: e bleibt in Linearkombination erhalten.
- Falls $e \notin E_C$: e bleibt nicht in Linearkombination erhalten.



2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

2. Zeigen Sie, dass $B_T \subseteq GF(2)^m$ ein Erzeugendensystem von $\mathcal{C}$ ist.

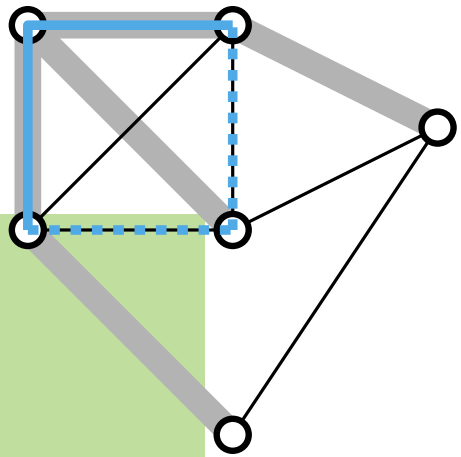
Hinweis:
Gehen Sie dabei konstruktiv vor und beschreiben Sie, wie ein beliebiger Kreis durch eine Linearkombination von Elementen aus B_T gebildet werden kann.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

Wdh.: C_e ist eindeutiger Kreis in gegebener Fundamentalbasis B_T über Nichtbaumkante e.

Betrachte Linearkombination $\sum_{e \in E_C \setminus E_T} C_e$ und zeige für beliebige Kante $e \in E$:

- Falls $e \in E_C$: e bleibt in Linearkombination erhalten.
- Falls $e \notin E_C$: e bleibt nicht in Linearkombination erhalten.



Betrachte die Fälle:

1. e ist Nichtbaumkante in C: $e \in E_C \setminus E_T$
2. e ist Baumkante in C: $e \in E_C \cap E_T$
3. e ist Nichtbaumkante außerhalb von C: $e \in E \setminus (E_C \cup E_T)$
4. e ist Baumkante außerhalb von C: $e \in E_T \setminus E_C$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

2. Problem

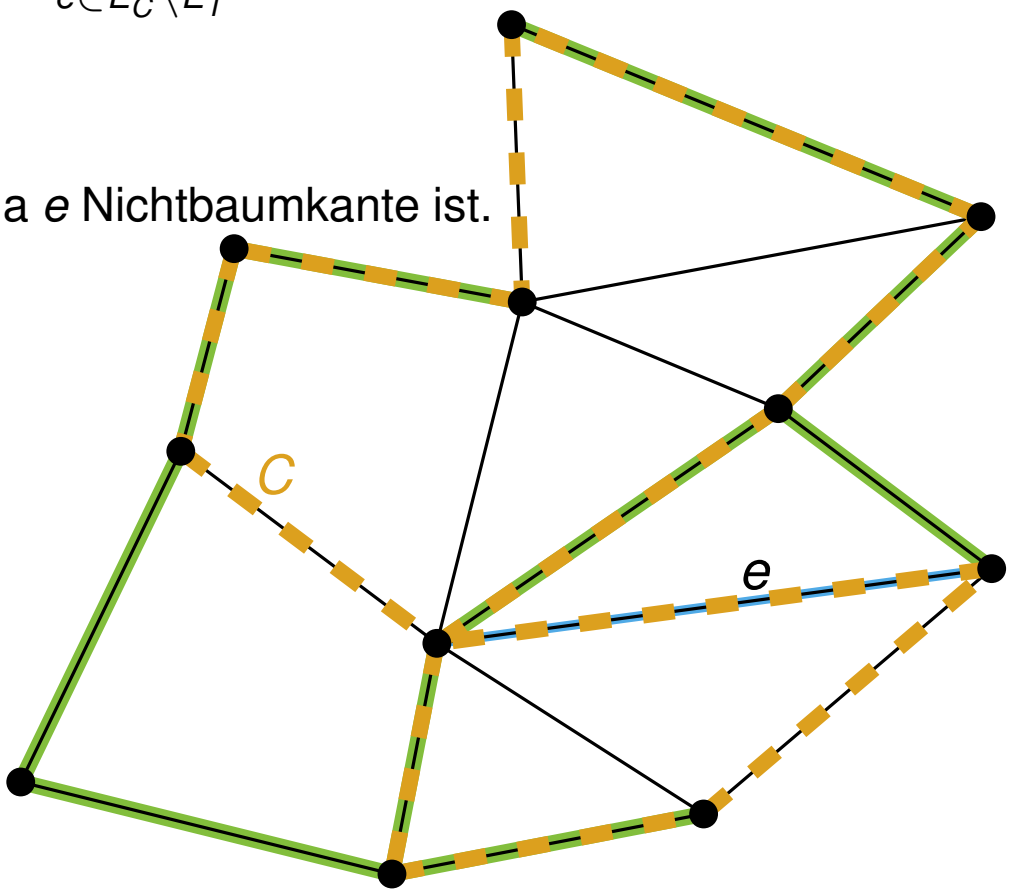
E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

1. Fall: e ist Nichtbaumkante in C

- C_e kommt als Summand vor.
- e kommt nur einmal in Summe vor, da e Nichtbaumkante ist.

$\Rightarrow$ Summe erhält e.

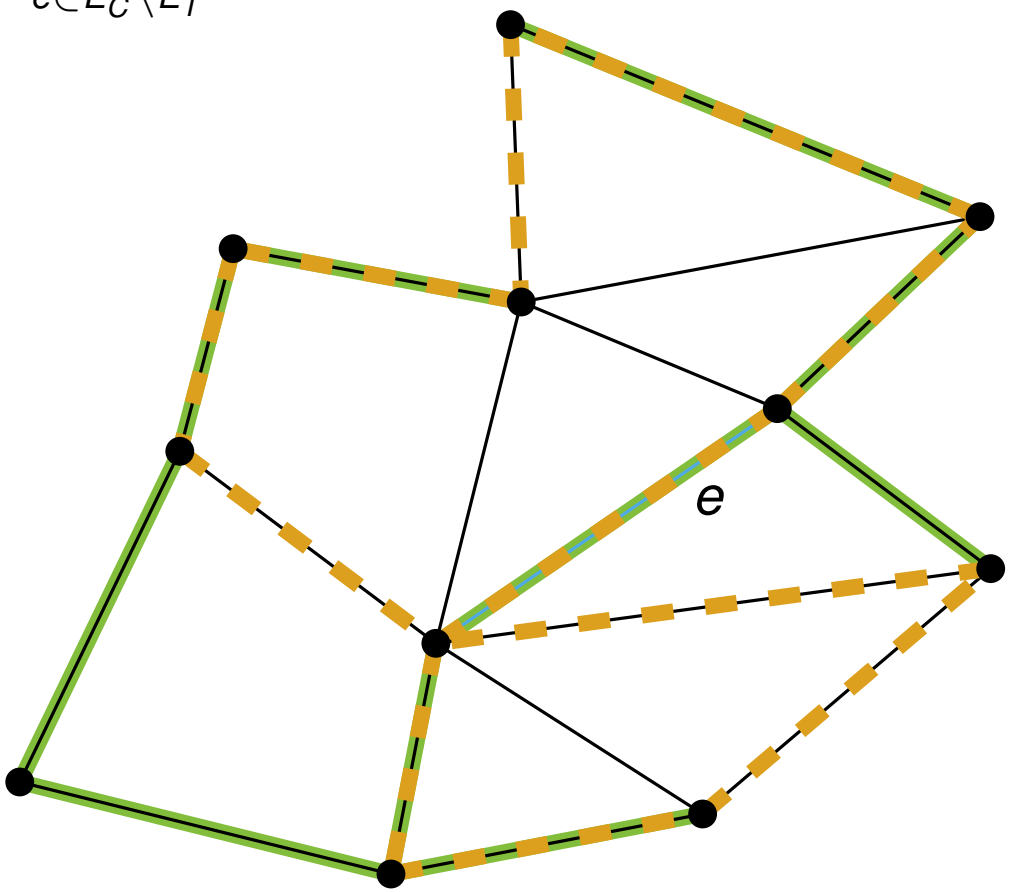


2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

2. Fall: e ist Baumkante in C



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

2. Problem

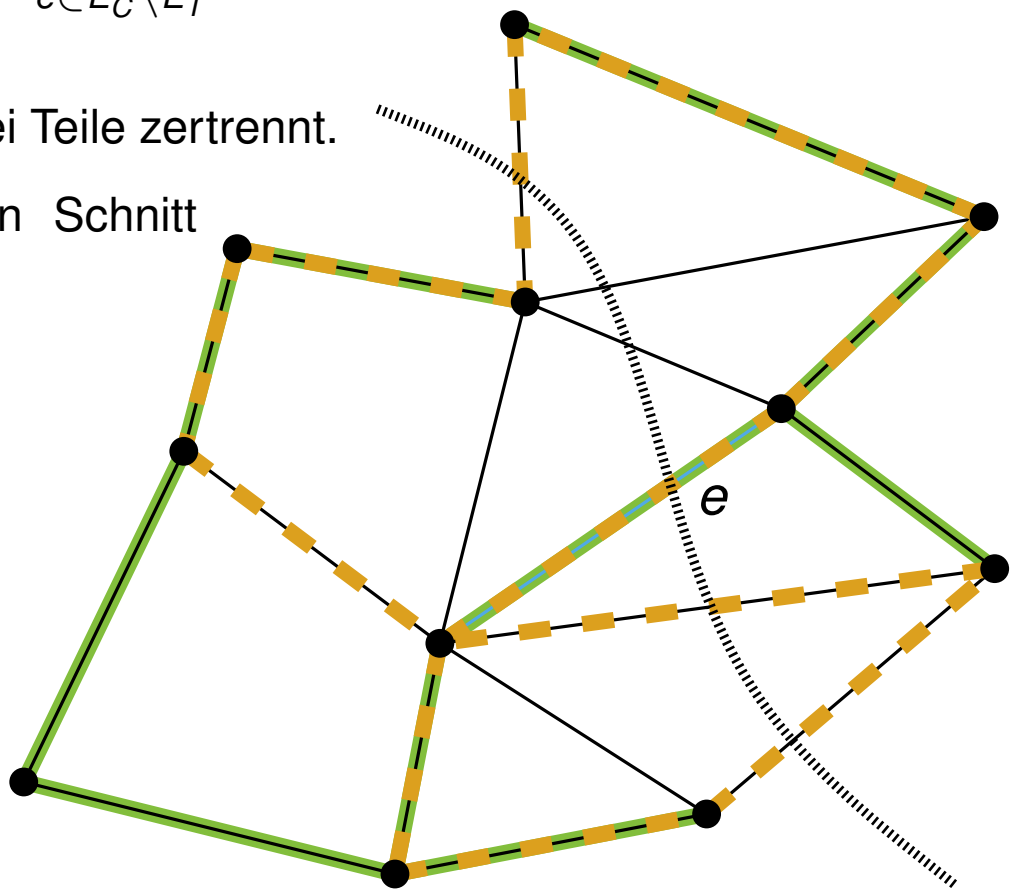
E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

2. Fall: e ist Baumkante in C

- e induziert Schnitt, da e Baum T in zwei Teile zertrennt.
- Anzahl der Kanten aus E_C, die den Schnitt kreuzen, ist **gerade**:

  Gilt für jeden Schnitt und Kreis.



2. Problem

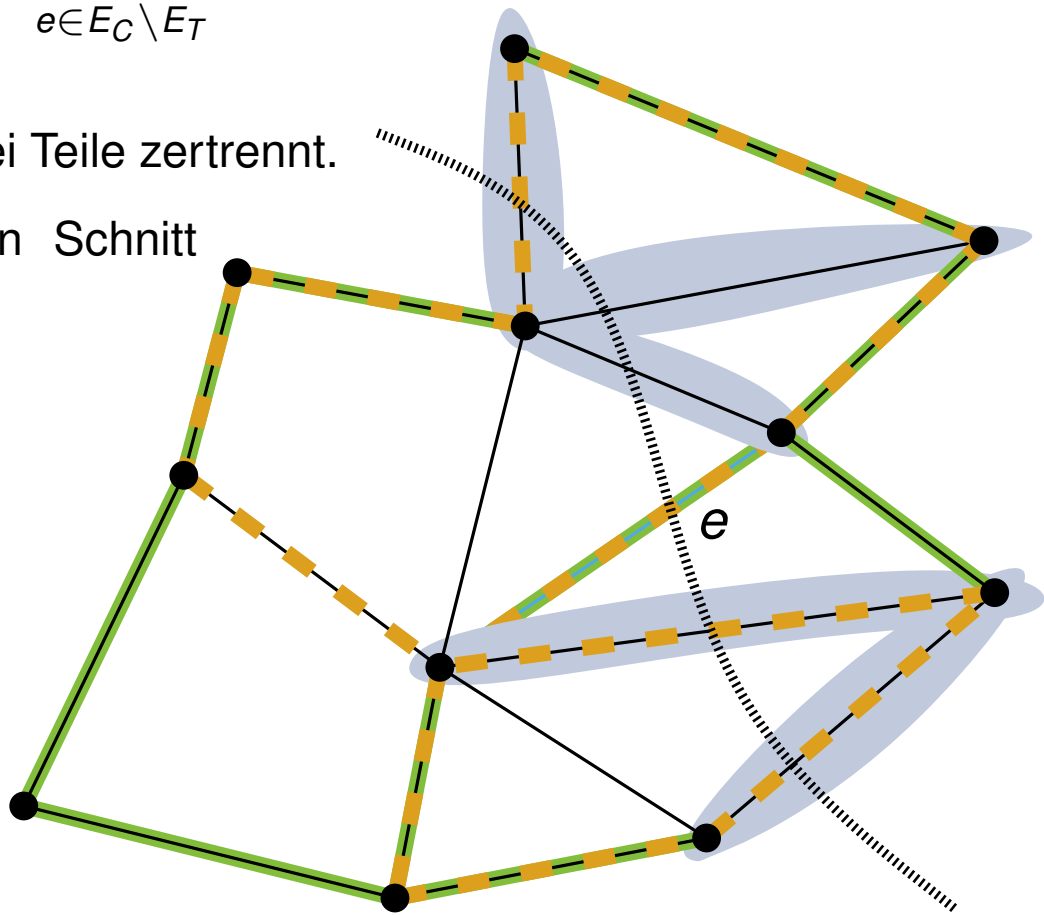
E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

2. Fall: e ist Baumkante in C

- e induziert Schnitt, da e Baum T in zwei Teile zertrennt.
- Anzahl der Kanten aus E_C, die den Schnitt kreuzen, ist **gerade**:

  Gilt für jeden Schnitt und Kreis.
- Alle Kanten außer e, die den Schnitt kreuzen, sind Nichtbaumkanten.



2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

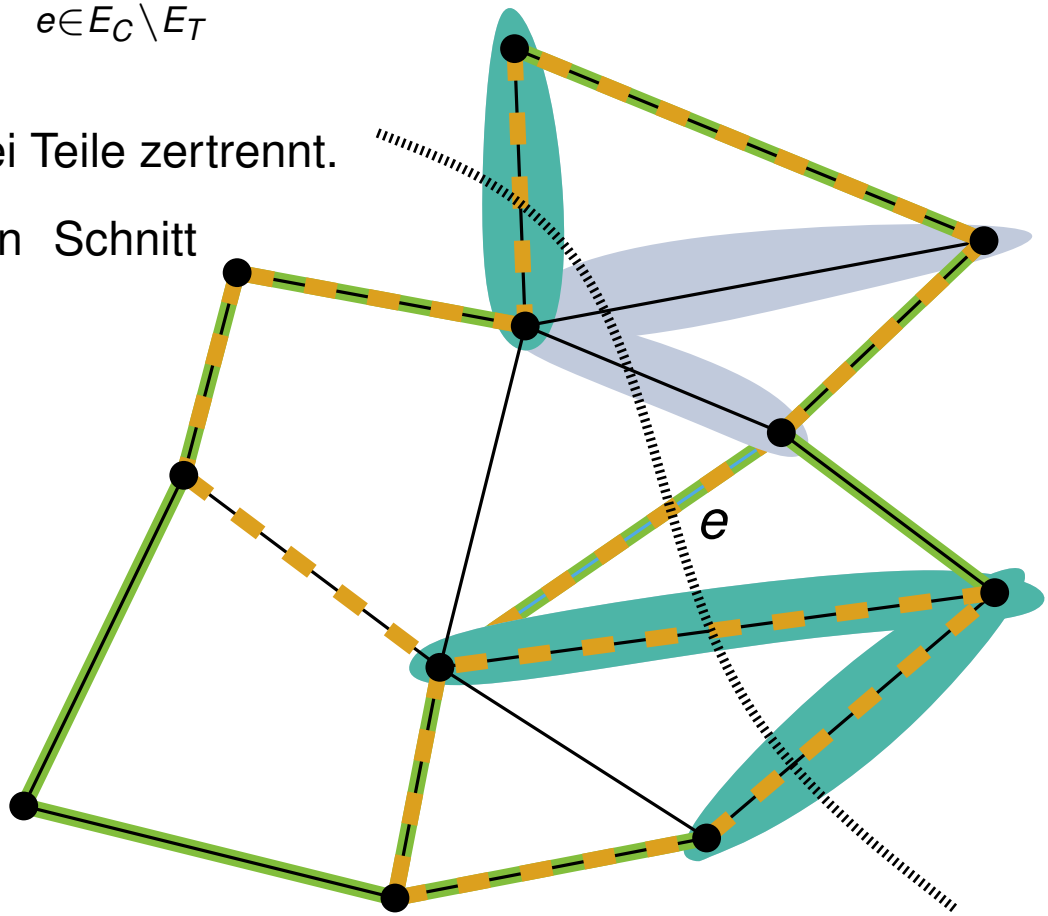
Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

2. Fall: e ist Baumkante in C

- e induziert Schnitt, da e Baum T in zwei Teile zertrennt.
- Anzahl der Kanten aus E_C, die den Schnitt kreuzen, ist **gerade**:

  Gilt für jeden Schnitt und Kreis.
- Alle Kanten außer e, die den Schnitt kreuzen, sind Nichtbaumkanten.

$\Rightarrow$ E_C hat **ungerade** Anzahl an Nichtbaumkanten E', die Schnitt kreuzen.



2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

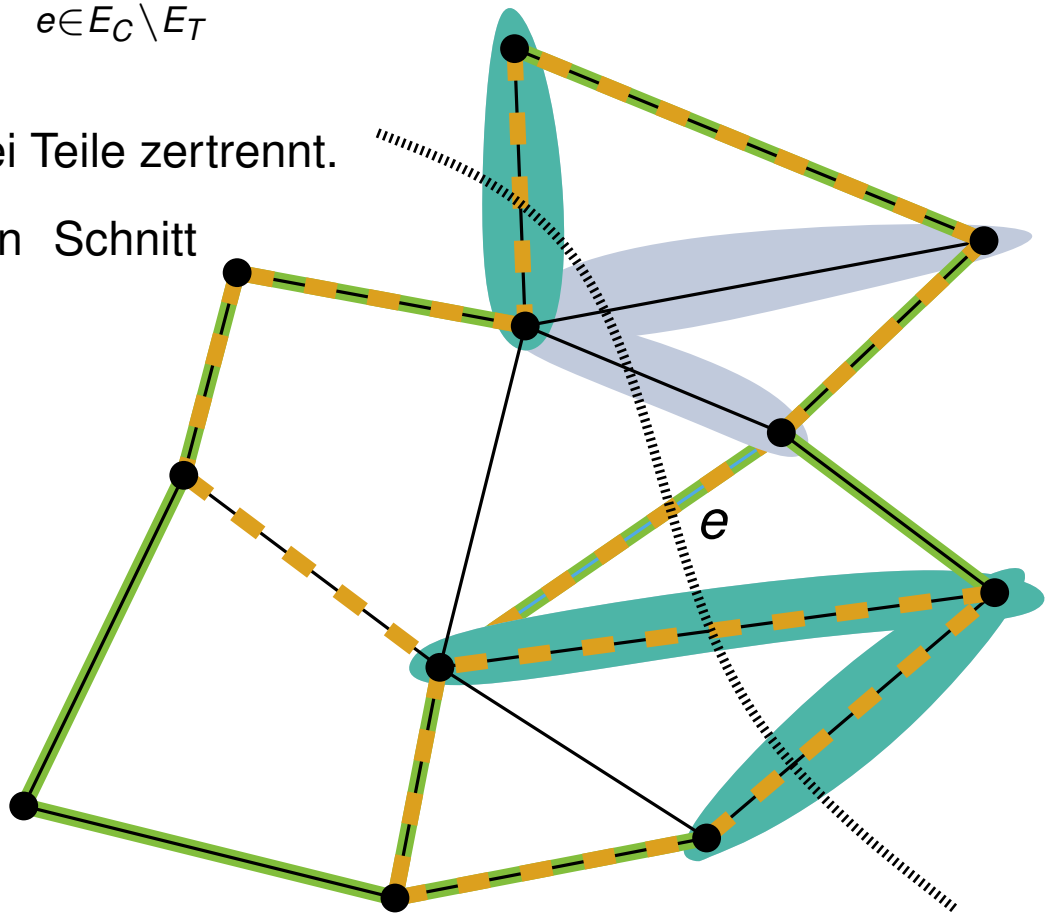
2. Fall: e ist Baumkante in C

- e induziert Schnitt, da e Baum T in zwei Teile zertrennt.
- Anzahl der Kanten aus E_C, die den Schnitt kreuzen, ist **gerade**:

  Gilt für jeden Schnitt und Kreis.
- Alle Kanten außer e, die den Schnitt kreuzen, sind Nichtbaumkanten.

$\Rightarrow$ E_C hat **ungerade** Anzahl an Nichtbaumkanten E', die Schnitt kreuzen.

$\Rightarrow$ E' bilden gerade die Kreise in B_T, die e enthalten.



2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

2. Fall: e ist Baumkante in C

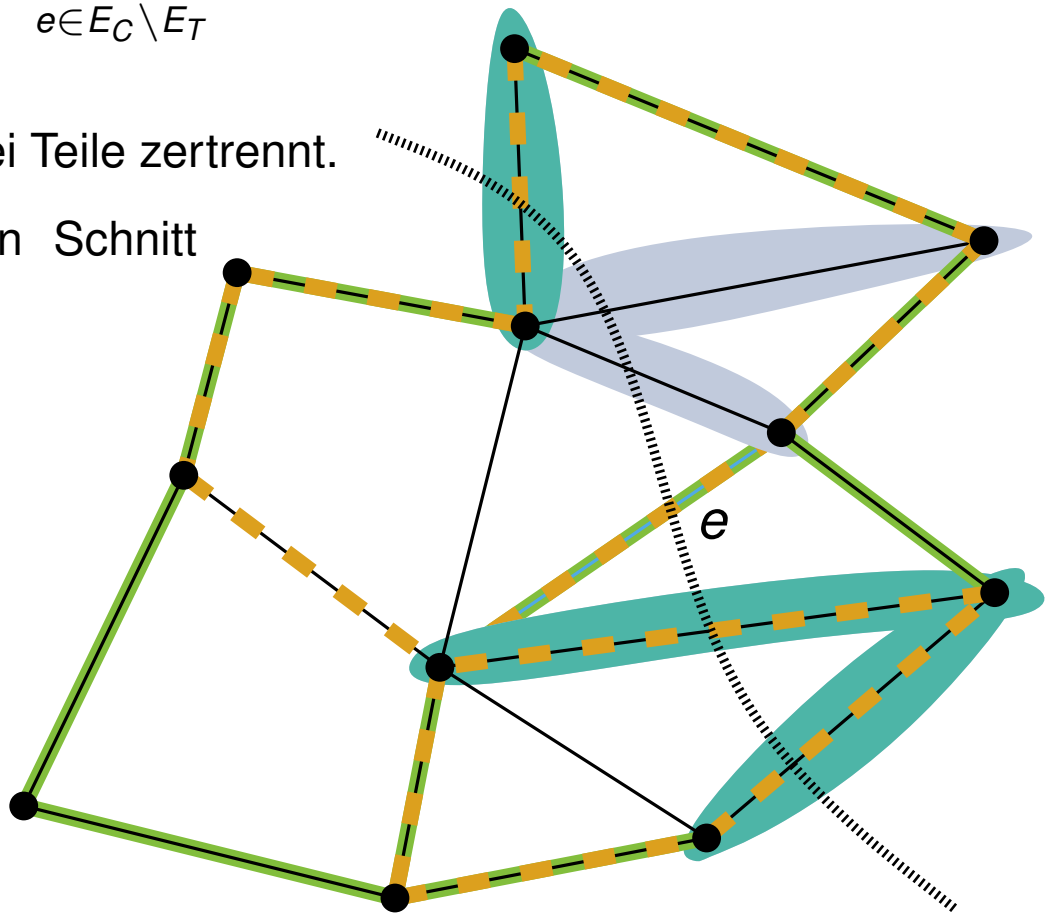
- e induziert Schnitt, da e Baum T in zwei Teile zertrennt.
- Anzahl der Kanten aus E_C, die den Schnitt kreuzen, ist **gerade**:

  Gilt für jeden Schnitt und Kreis.
- Alle Kanten außer e, die den Schnitt kreuzen, sind Nichtbaumkanten.

$\Rightarrow$ E_C hat **ungerade** Anzahl an Nichtbaumkanten E', die Schnitt kreuzen.

$\Rightarrow$ E' bilden gerade die Kreise in B_T, die e enthalten.

$\Rightarrow$ e wird nicht zu 0 summiert.



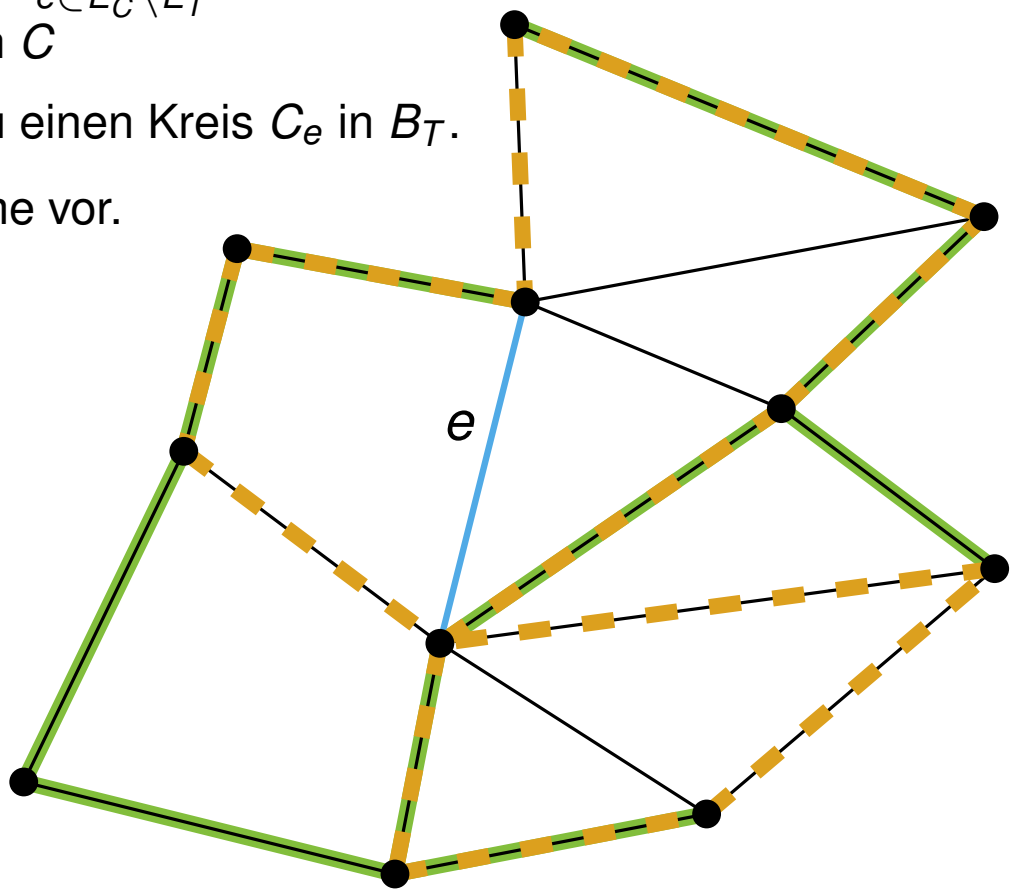
2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

3. Fall: e ist Nichtbaumkante außerhalb von C

- Da e Nichtbaumkante, gibt es genau einen Kreis C_e in B_T.
- Da $e \notin E_C$ kommt C_e nicht in Summe vor.

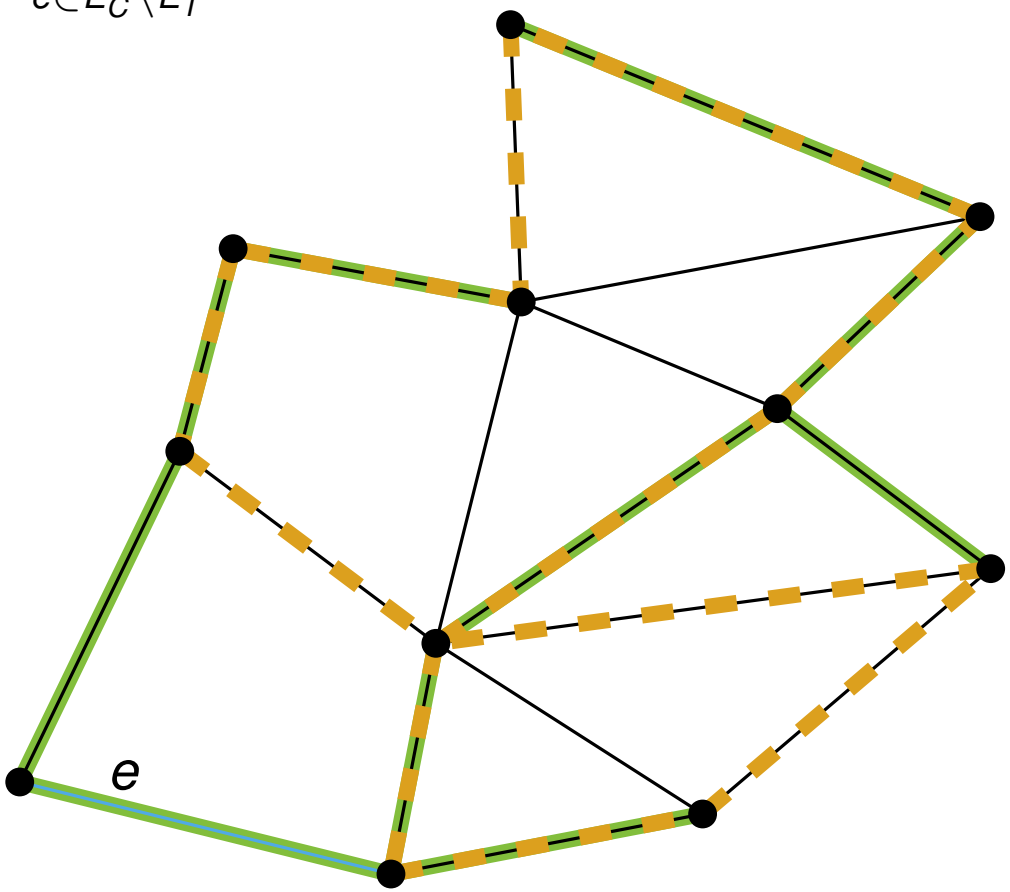


2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

4. Fall: e ist Baumkante außerhalb von C



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

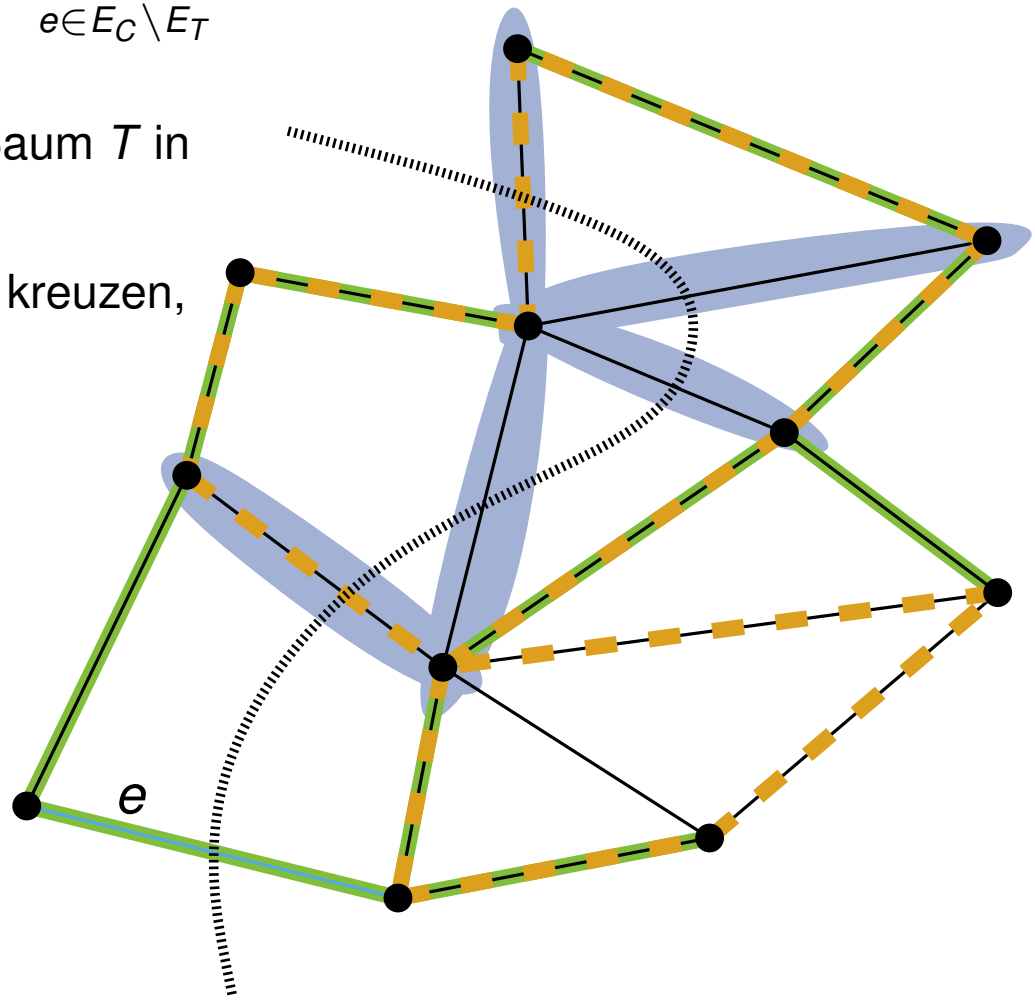
2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

4. Fall: e ist Baumkante außerhalb von C

- e induziert einen Schnitt, da e den Baum T in zwei Teile zertrennt.
- Alle Kanten außer e, die den Schnitt kreuzen, sind Nichtbaumkanten.



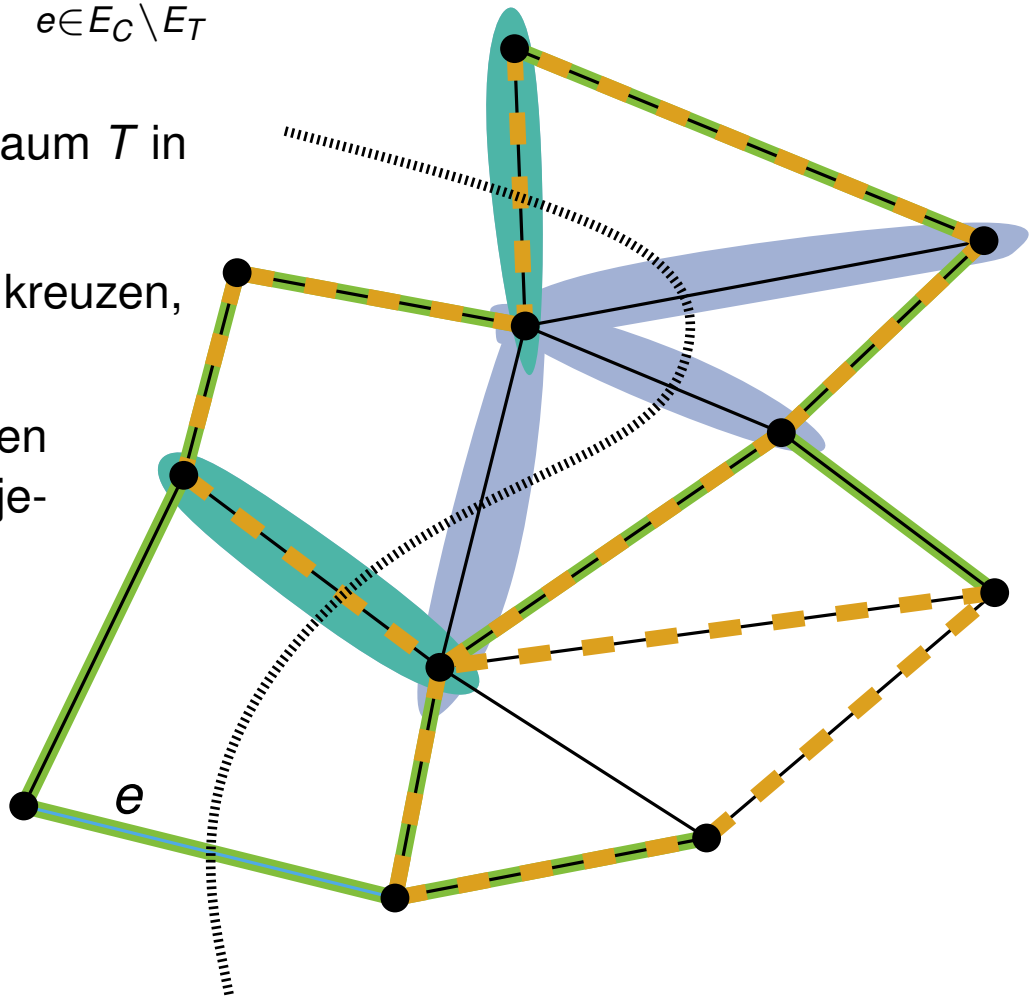
2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

4. Fall: e ist Baumkante außerhalb von C

- e induziert einen Schnitt, da e den Baum T in zwei Teile zertrennt.
- Alle Kanten außer e, die den Schnitt kreuzen, sind Nichtbaumkanten.
- Anzahl der Kanten aus E_C, die den Schnitt kreuzen, ist gerade: Gilt für jeden Kreis und Schnitt.



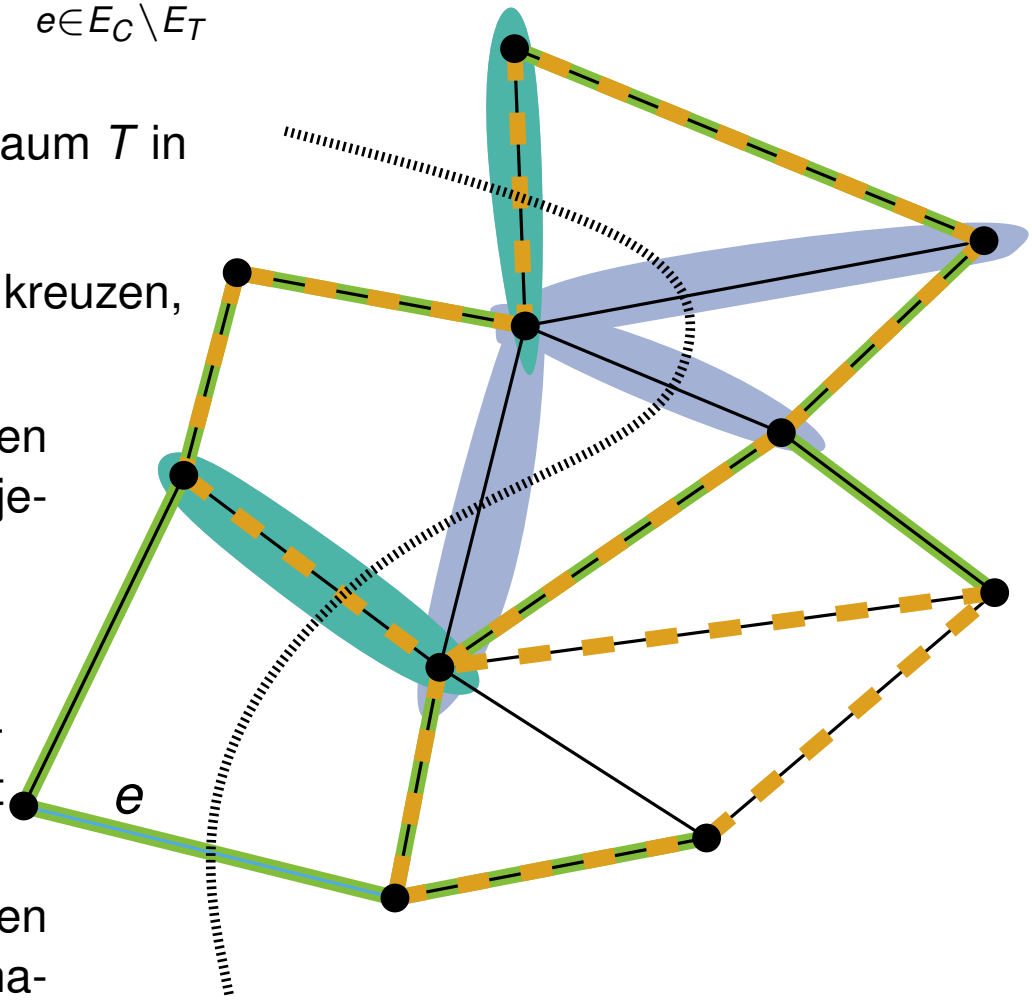
2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

Behauptung: Für jeden Kreis C in G gilt: $C = \sum_{e \in E_C \setminus E_T} C_e$

4. Fall: e ist Baumkante außerhalb von C

- e induziert einen Schnitt, da e den Baum T in zwei Teile zertrennt.
- Alle Kanten außer e, die den Schnitt kreuzen, sind Nichtbaumkanten.
- Anzahl der Kanten aus E_C, die den Schnitt kreuzen, ist gerade: Gilt für jeden Kreis und Schnitt.



$\Rightarrow$ Wegen $e \notin E_C$ ist Anzahl der Nichtbaumkanten E' in E_C die Schnitt kreuzen **gerade**.

$\Rightarrow$ Die Nichtbaumkanten E' entsprechen gerade der Kreise in Linearkombination, die e enthalten.

Folglich: e wird zu 0 aufsummiert.

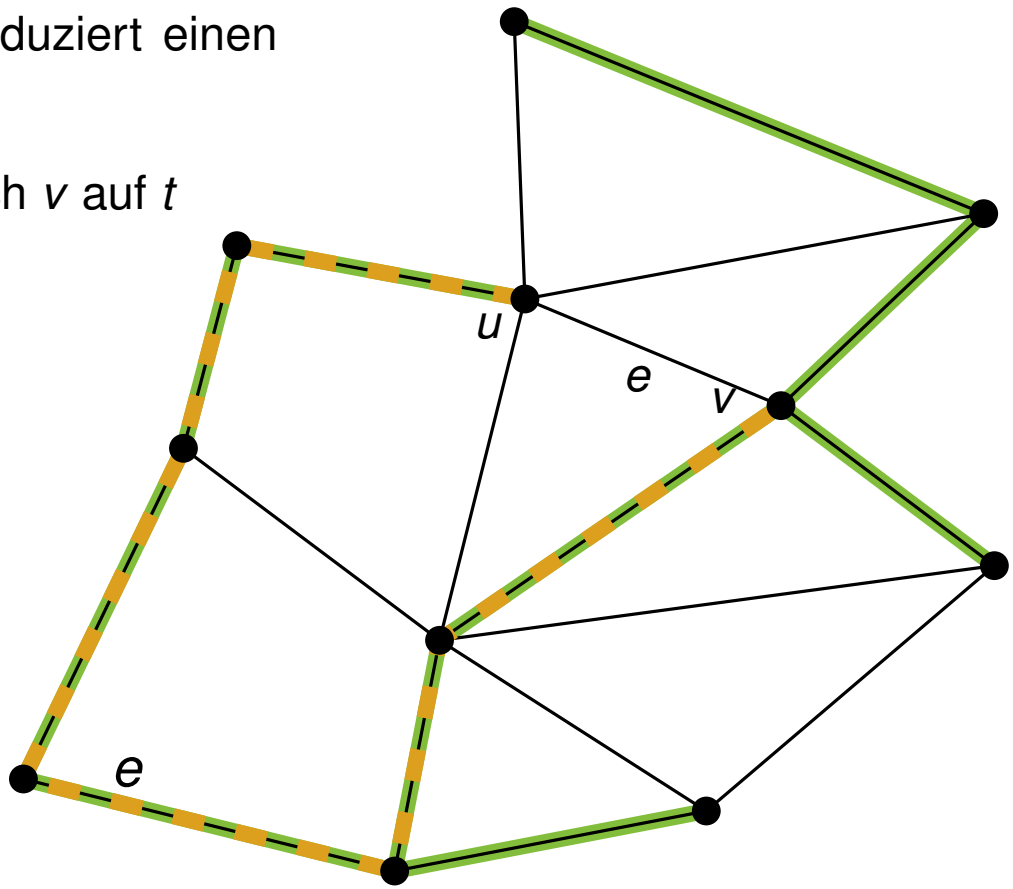
2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

3. Zeigen Sie, dass $|B_T| = m - n + 1$ gilt.

Beob.: Jede Nichtbaumkante $e = \{u, v\}$ induziert einen eindeutigen Kreis C_e:

$\{u, v\}$ + einfacher Weg von u nach v auf t



2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

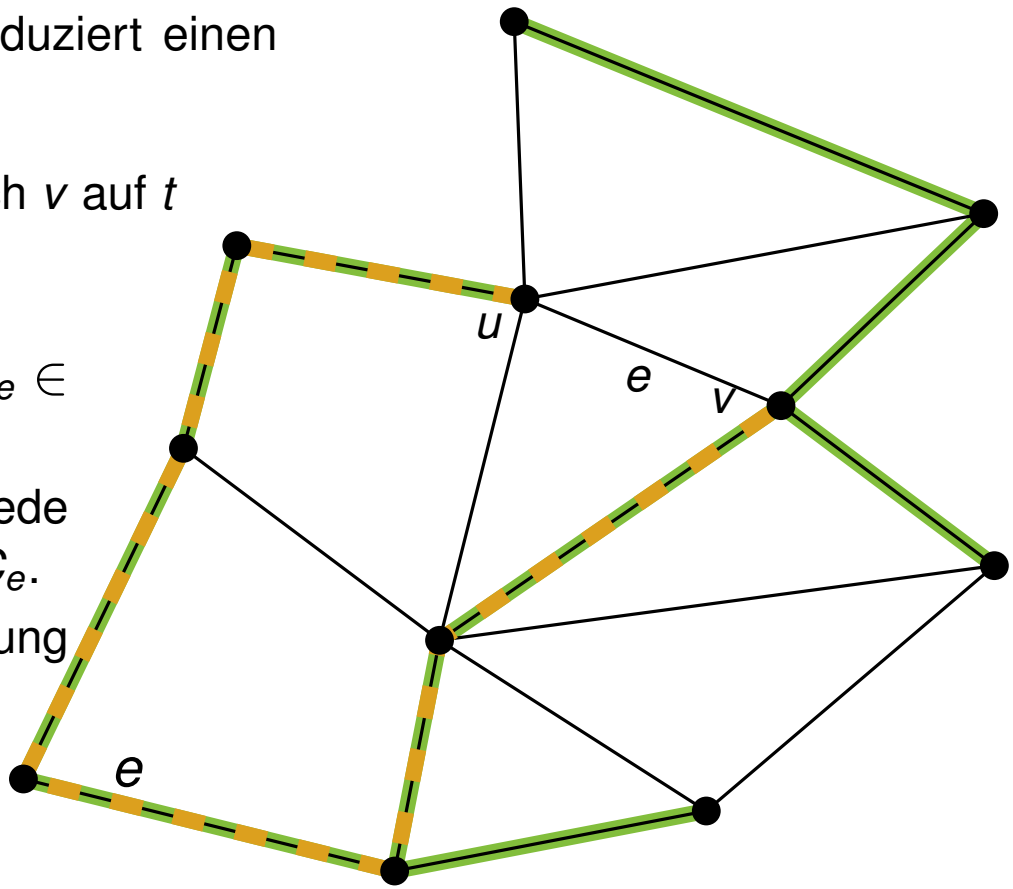
3. Zeigen Sie, dass $|B_T| = m - n + 1$ gilt.

Beob.: Jede Nichtbaumkante $e = \{u, v\}$ induziert einen eindeutigen Kreis C_e:

$\{u, v\}$ + einfacher Weg von u nach v auf t

- Wir zeigen zuerst: $|B_T| = |E \setminus E_T|$
  - Nach Konstruktion enthält jeder Kreis $C_e \in B_T$ genau eine Nichtbaumkante.
  - Wegen obiger Beobachtung induziert jede Nichtbaumkante e genau einen Kreis C_e.

  $\Rightarrow$ wegen bijektiver Mengenentsprechung folgt $|B_T| = |E \setminus E_T|$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

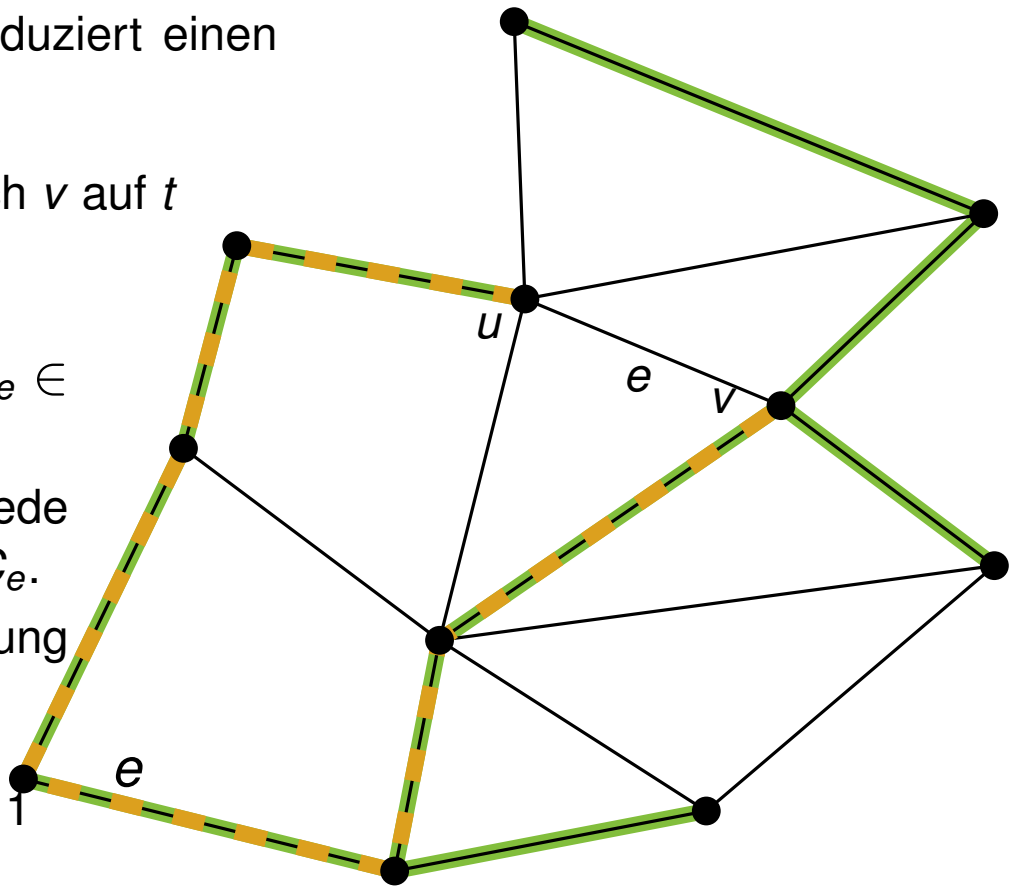
2. Problem

E_C= Kanten im Kreis C.
E_T= Kanten im aufspannenden Baum T.

3. Zeigen Sie, dass $|B_T| = m - n + 1$ gilt.

Beob.: Jede Nichtbaumkante $e = \{u, v\}$ induziert einen eindeutigen Kreis C_e:

$\{u, v\}$ + einfacher Weg von u nach v auf t



- Wir zeigen zuerst: $|B_T| = |E \setminus E_T|$
  - Nach Konstruktion enthält jeder Kreis $C_e \in B_T$ genau eine Nichtbaumkante.
  - Wegen obiger Beobachtung induziert jede Nichtbaumkante e genau einen Kreis C_e.

  $\Rightarrow$ wegen bijektiver Mengenentsprechung folgt $|B_T| = |E \setminus E_T|$

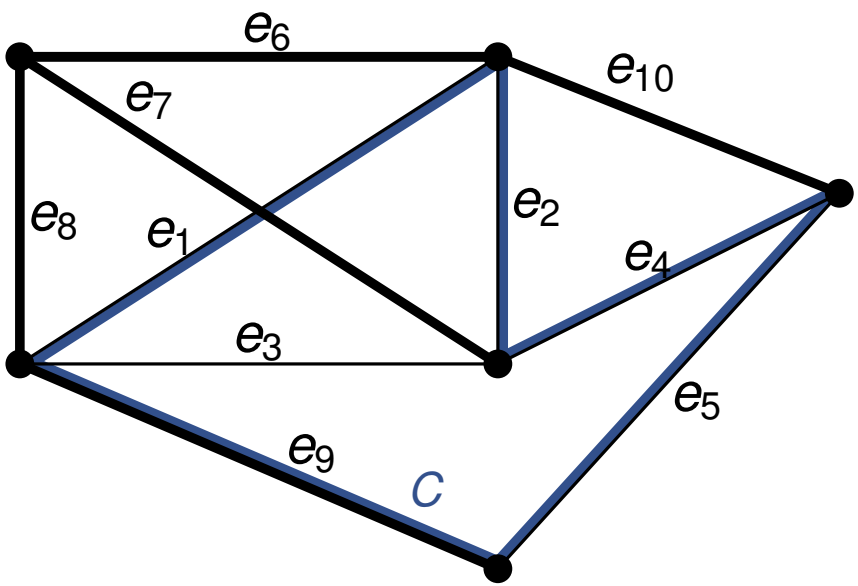
- Aufspannender Baum in Graph hat $n - 1$ Kanten. Damit gilt

$$|B_T| = |E \setminus E_T| = m - n + 1$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von De Pina

Betrachte Kreise als Inzidenzvektoren über E mit Einschränkung auf die Nichtbaumkanten $\{e_1, \ldots, e_N\}$.



Beispiel: Kreise werden mithilfe der Nichtbaumkanten e_1, e_2, e_3, e_4 und e_5 beschrieben.

$$C = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} \begin{matrix} e_1 \\ \\ \vdots \\ \\ e_5 \end{matrix}$$

Kreis C kann mithilfe der Fundamentalkreise C_i (C_i=Fundamentalkreis der Nichtbaumkante e_i) rekonstruiert werden.

$$C = C_1 \oplus C_2 \oplus C_4 \oplus C_5$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von de Pina

Eingabe: Graph $G = (V, E)$, aufspannenden Baum T mit Nichtbaumkanten $\{e_1, \ldots, e_N\}$

Ausgabe: MCB von G

for $i = 1$ *bis* N **do**
  $S_i \leftarrow \{e_i\}$

for $k = 1$ *bis* N **do**
  Finde einen kürzesten Kreis C_k mit $\langle C_k, S_k \rangle = 1$
  for $i = k + 1$ *bis* N **do**
    if $\langle C_k, S_i \rangle = 1$ **then**
      $S_i \leftarrow S_i \oplus S_k$

Ausgabe ist: $\{C_1, \ldots, C_N\}$

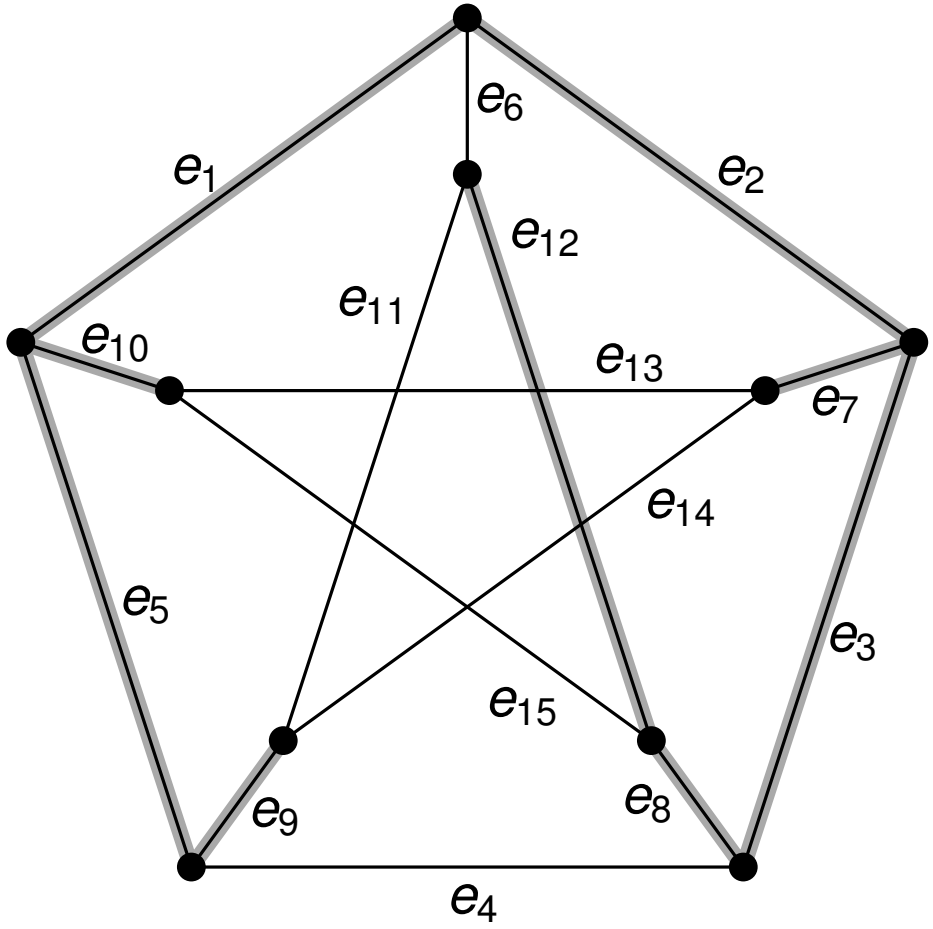
Definiere Bilinearform zweier Vektoren C und S: $\langle C, S \rangle := \sum_{i=1}^{N} (c_i \cdot s_i)$

Produkt und Summe sind über GF(2) definiert.

C und S sind *orthogonal* zueinander genau dann, wenn $\langle C, S \rangle = 0$.

$\langle C, S \rangle = 1$ genau dann, wenn C eine ungerade Anzahl Einträge (Kreise) mit S gemeinsam hat.

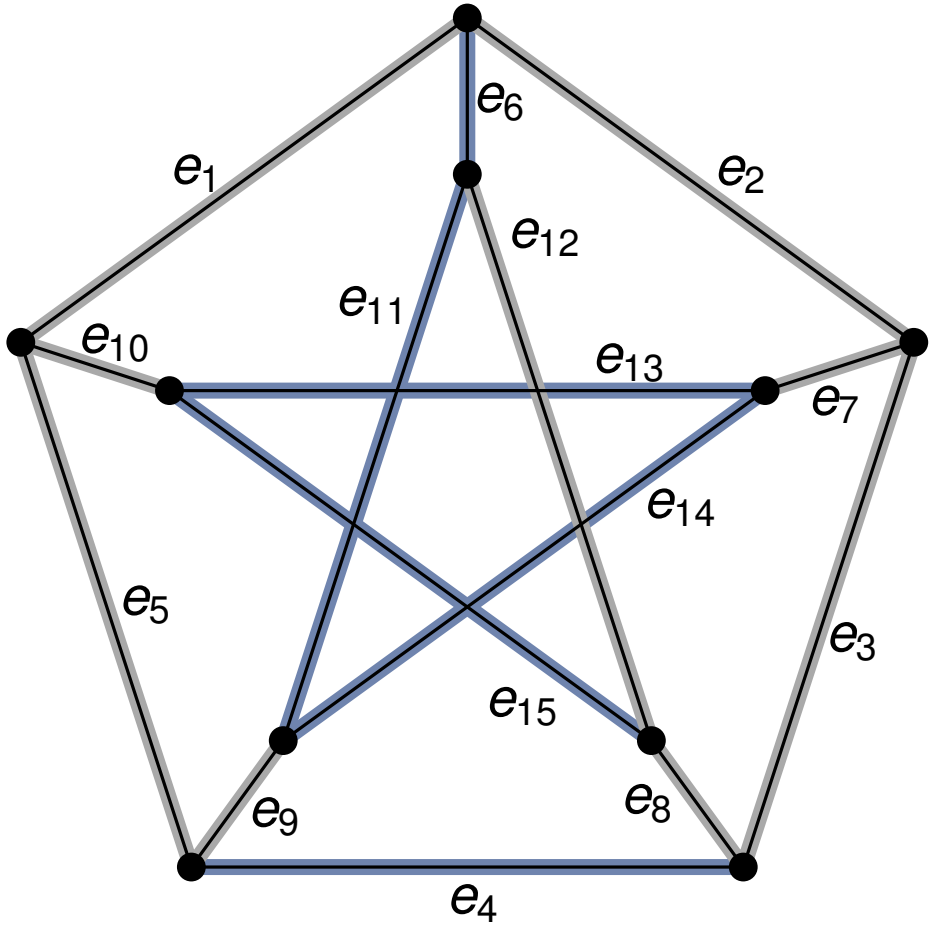
Problem 2



Peterson-Graph

Gewicht pro Kante: 3

Problem 2



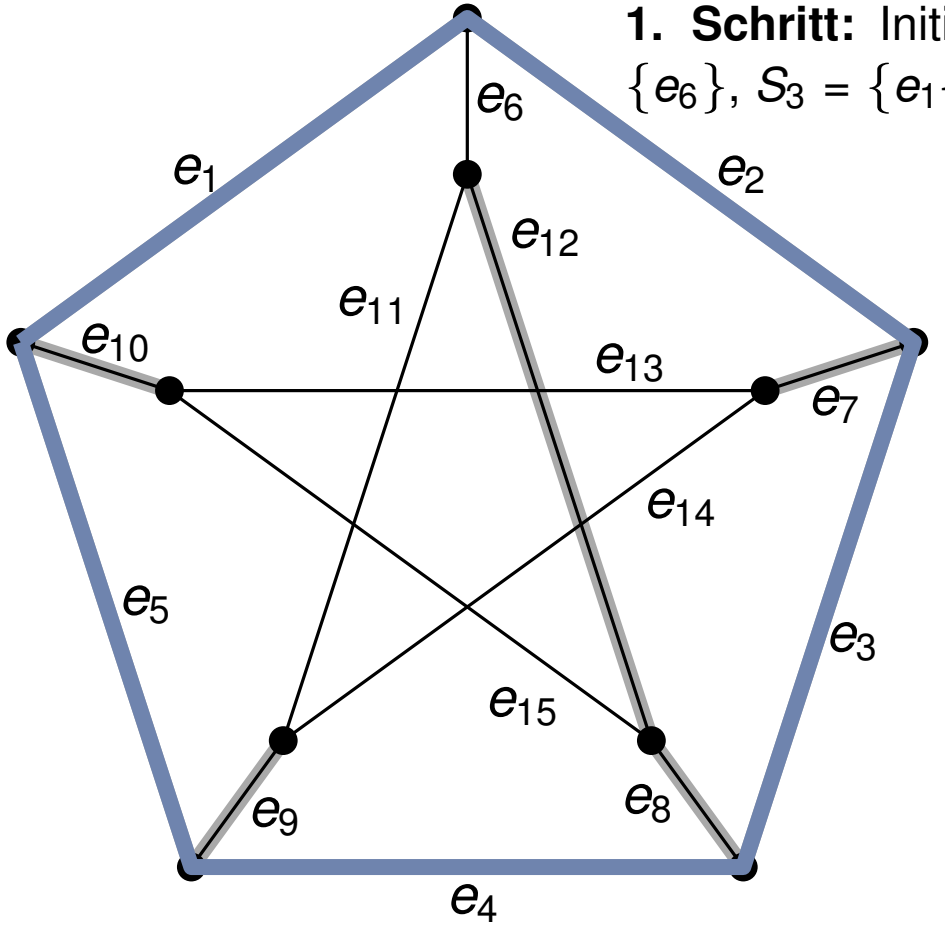
Peterson-Graph

Gewicht pro Kante: 3

1. Schritt: Initialisierung mit Nichtbaumkanten

$S_1 = \{e_4\}$
$S_2 = \{e_6\}$
$S_3 = \{e_{11}\}$
$S_4 = \{e_{13}\}$
$S_5 = \{e_{14}\}$
$S_6 = \{e_{15}\}$

Problem 2



Peterson-Graph

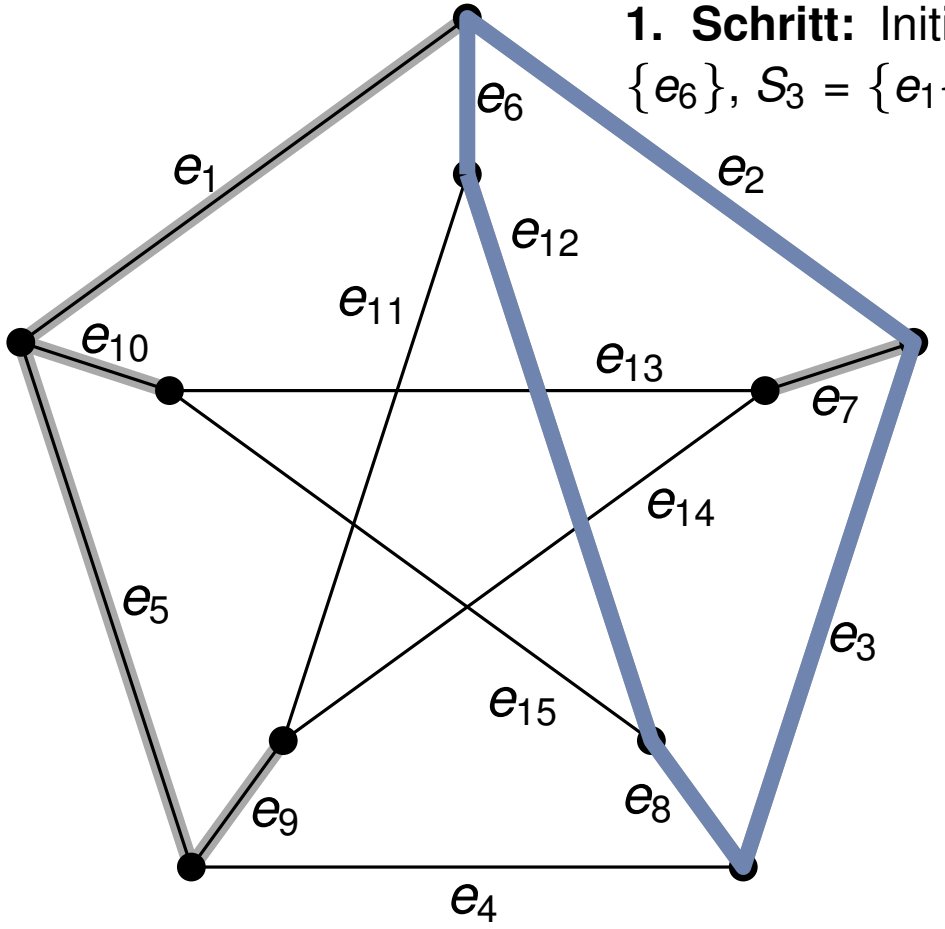
1. Schritt: Initialisierung mit Nichtbaumkanten $S_1 = \{e_4\}$, $S_2 = \{e_6\}$, $S_3 = \{e_{11}\}$, $S_4 = \{e_{13}\}$, $S_5 = \{e_{14}\}$, $S_6 = \{e_{15}\}$

k = 1: Wähle $C_1 = \{e_1, e_2, e_3, e_4, e_5\}$, $w(C_1)=15$
Für i=2...6 gibt es kein S_i mit $\langle C_1, S_i \rangle = 1$

Gewicht pro Kante: 3

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 2



Peterson-Graph

1. Schritt: Initialisierung mit Nichtbaumkanten $S_1 = \{e_4\}$, $S_2 = \{e_6\}$, $S_3 = \{e_{11}\}$, $S_4 = \{e_{13}\}$, $S_5 = \{e_{14}\}$, $S_6 = \{e_{15}\}$

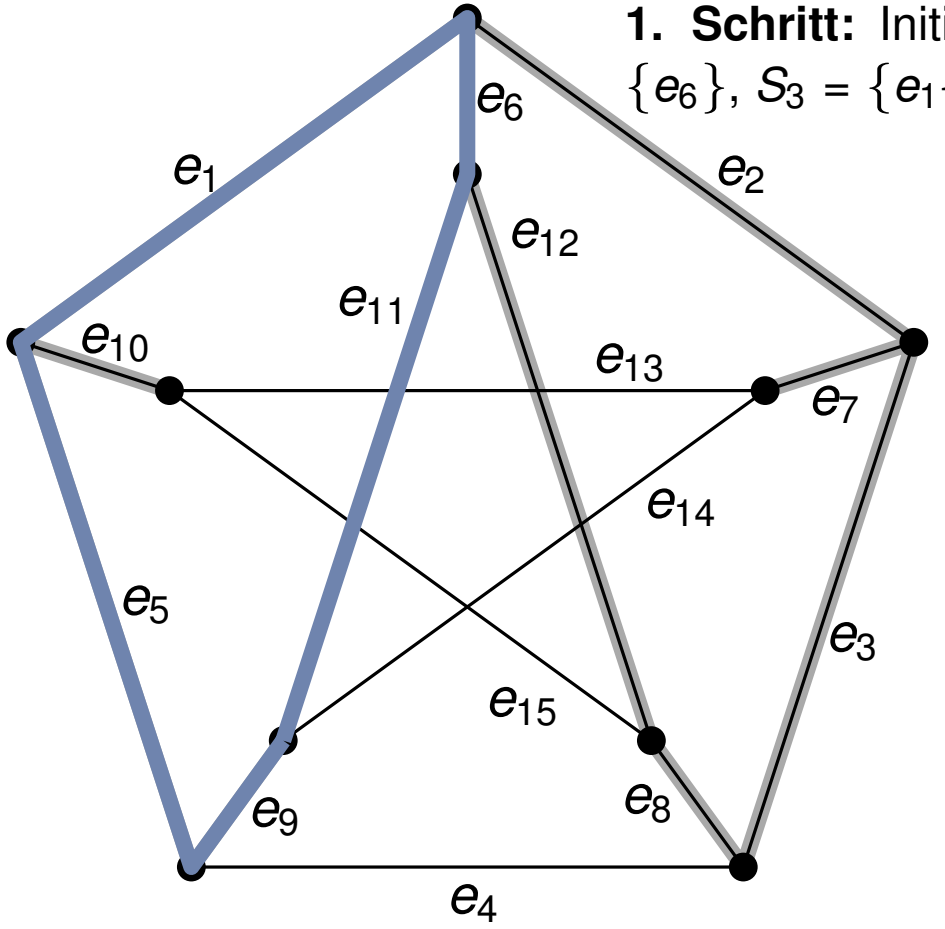
k = 1: Wähle $C_1 = \{e_1, e_2, e_3, e_4, e_5\}$, $w(C_1)=15$
Für i=2...6 gibt es kein S_i mit $\langle C_1, S_i \rangle = 1$

k = 2: Wähle $C_2 = \{e_6, e_2, e_3, e_8, e_{12}\}$, $w(C_2)=15$
Für i=3...6 gibt es kein S_i mit $\langle C_2, S_i \rangle = 1$

Gewicht pro Kante: 3

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 2



Peterson-Graph

1. Schritt: Initialisierung mit Nichtbaumkanten $S_1 = \{e_4\}$, $S_2 = \{e_6\}$, $S_3 = \{e_{11}\}$, $S_4 = \{e_{13}\}$, $S_5 = \{e_{14}\}$, $S_6 = \{e_{15}\}$

k = 1: Wähle $C_1 = \{e_1, e_2, e_3, e_4, e_5\}$, $w(C_1)$=15
Für i=2...6 gibt es kein S_i mit $\langle C_1, S_i \rangle = 1$

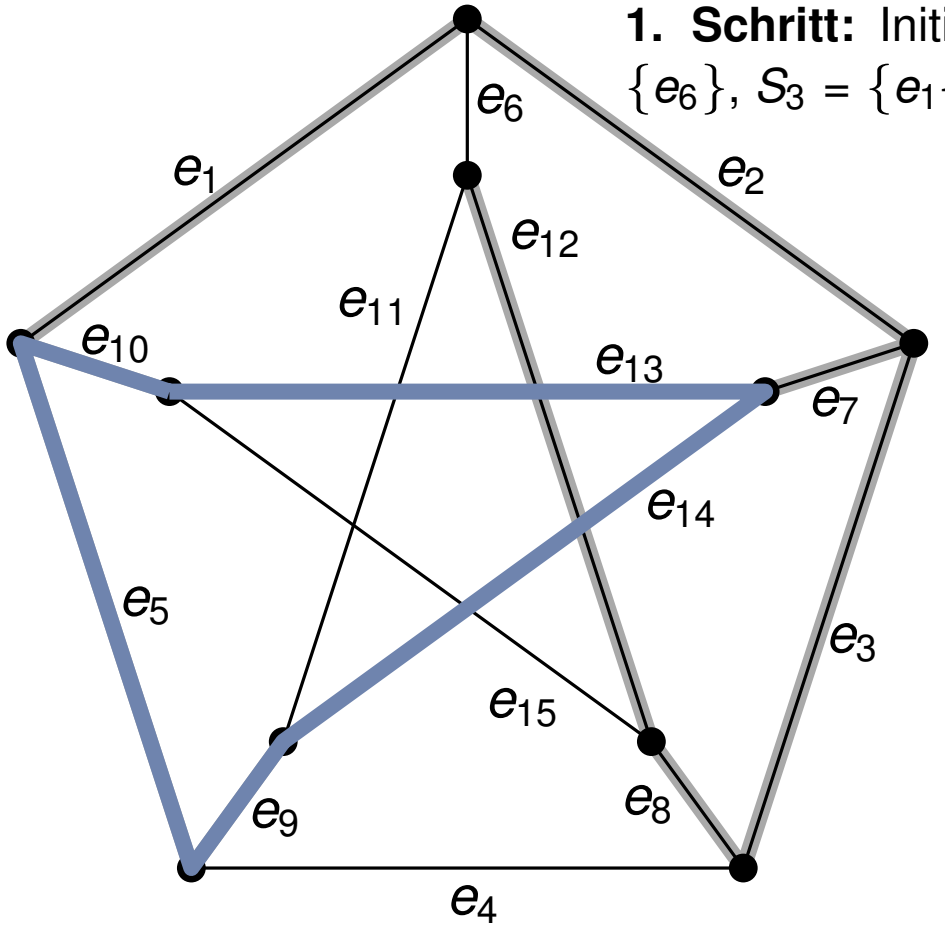
k = 2: Wähle $C_2 = \{e_6, e_2, e_3, e_8, e_{12}\}$, $w(C_2)$=15
Für i=3...6 gibt es kein S_i mit $\langle C_2, S_i \rangle = 1$

k = 3: Wähle $C_3 = \{e_{11}, e_6, e_1, e_5, e_9\}$, $w(C_3)$=15
Für i=4...6 gibt es kein S_i mit $\langle C_3, S_i \rangle = 1$

Gewicht pro Kante: 3

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 2



Peterson-Graph

Gewicht pro Kante: 3

1. Schritt: Initialisierung mit Nichtbaumkanten $S_1 = \{e_4\}$, $S_2 = \{e_6\}$, $S_3 = \{e_{11}\}$, $S_4 = \{e_{13}\}$, $S_5 = \{e_{14}\}$, $S_6 = \{e_{15}\}$

k = 1: Wähle $C_1 = \{e_1, e_2, e_3, e_4, e_5\}$, $w(C_1)$=15
Für i=2...6 gibt es kein S_i mit $\langle C_1, S_i \rangle = 1$

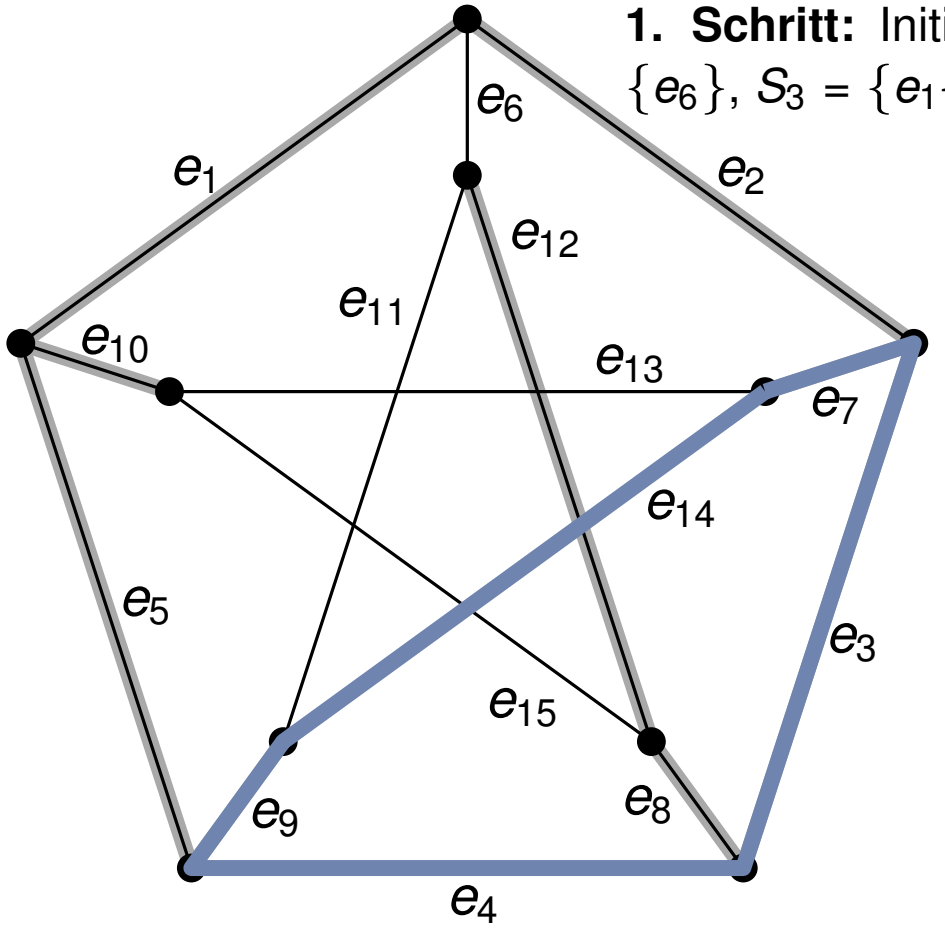
k = 2: Wähle $C_2 = \{e_6, e_2, e_3, e_8, e_{12}\}$, $w(C_2)$=15
Für i=3...6 gibt es kein S_i mit $\langle C_2, S_i \rangle = 1$

k = 3: Wähle $C_3 = \{e_{11}, e_6, e_1, e_5, e_9\}$, $w(C_3)$=15
Für i=4...6 gibt es kein S_i mit $\langle C_3, S_i \rangle = 1$

k = 4: Wähle $C_4 = \{e_{13}, e_{14}, e_9, e_5, e_{10}\}$, $w(C_4)$=15
$\langle C_4, S_5 \rangle = 1$, $S_5 := S_5 \oplus S_4 = \{e_{13}, e_{14}\}$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 2



Peterson-Graph

1. Schritt: Initialisierung mit Nichtbaumkanten $S_1 = \{e_4\}$, $S_2 = \{e_6\}$, $S_3 = \{e_{11}\}$, $S_4 = \{e_{13}\}$, $S_5 = \{e_{14}\}$, $S_6 = \{e_{15}\}$

k = 1: Wähle $C_1 = \{e_1, e_2, e_3, e_4, e_5\}$, $w(C_1)$=15
Für i=2...6 gibt es kein S_i mit $\langle C_1, S_i \rangle = 1$

k = 2: Wähle $C_2 = \{e_6, e_2, e_3, e_8, e_{12}\}$, $w(C_2)$=15
Für i=3...6 gibt es kein S_i mit $\langle C_2, S_i \rangle = 1$

k = 3: Wähle $C_3 = \{e_{11}, e_6, e_1, e_5, e_9\}$, $w(C_3)$=15
Für i=4...6 gibt es kein S_i mit $\langle C_3, S_i \rangle = 1$

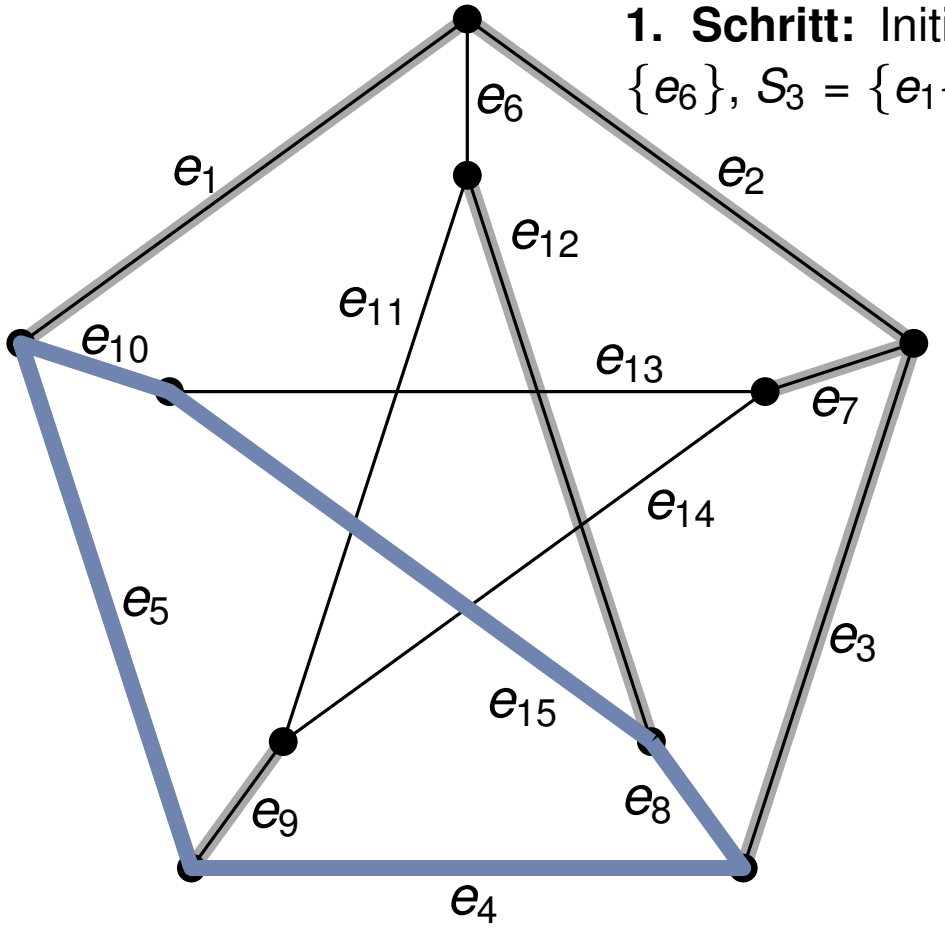
k = 4: Wähle $C_4 = \{e_{13}, e_{14}, e_9, e_5, e_{10}\}$, $w(C_4)$=15
$\langle C_4, S_5 \rangle = 1$, $S_5 := S_5 \oplus S_4 = \{e_{13}, e_{14}\}$

k = 5: Wähle $C_5 = \{e_{14}, e_9, e_4, e_3, e_7\}$, $w(C_5)$=15
$\langle C_5, S_6 \rangle = 0$

Gewicht pro Kante: 3

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 2



Peterson-Graph

Gewicht pro Kante: 3

1. Schritt: Initialisierung mit Nichtbaumkanten $S_1 = \{e_4\}$, $S_2 = \{e_6\}$, $S_3 = \{e_{11}\}$, $S_4 = \{e_{13}\}$, $S_5 = \{e_{14}\}$, $S_6 = \{e_{15}\}$

k = 1: Wähle $C_1 = \{e_1, e_2, e_3, e_4, e_5\}$, $w(C_1)$=15
Für i=2...6 gibt es kein S_i mit $\langle C_1, S_i \rangle = 1$

k = 2: Wähle $C_2 = \{e_6, e_2, e_3, e_8, e_{12}\}$, $w(C_2)$=15
Für i=3...6 gibt es kein S_i mit $\langle C_2, S_i \rangle = 1$

k = 3: Wähle $C_3 = \{e_{11}, e_6, e_1, e_5, e_9\}$, $w(C_3)$=15
Für i=4...6 gibt es kein S_i mit $\langle C_3, S_i \rangle = 1$

k = 4: Wähle $C_4 = \{e_{13}, e_{14}, e_9, e_5, e_{10}\}$, $w(C_4)$=15
$\langle C_4, S_5 \rangle = 1$, $S_5 := S_5 \oplus S_4 = \{e_{13}, e_{14}\}$

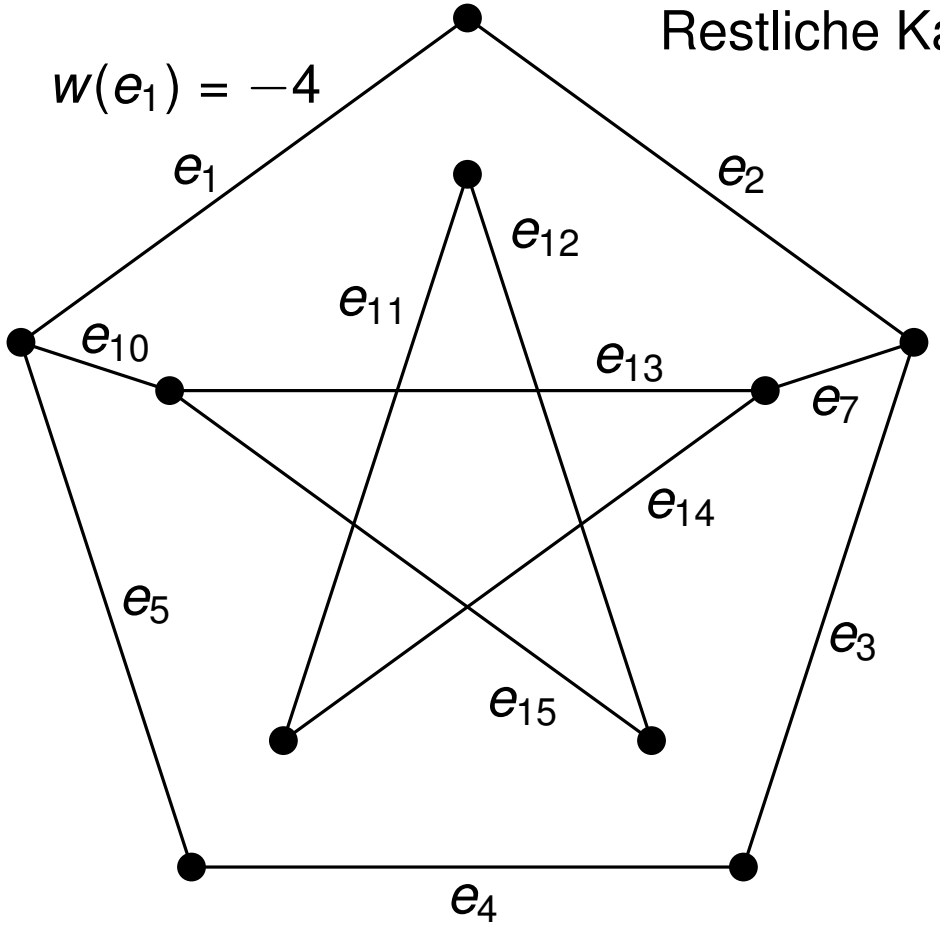
k = 5: Wähle $C_5 = \{e_{14}, e_9, e_4, e_3, e_7\}$, $w(C_5)$=15
$\langle C_5, S_6 \rangle = 0$

k = 6: Wähle $C_6 = \{e_{15}, e_{10}, e_5, e_4, e_8\}$, $w(C_5)$=15

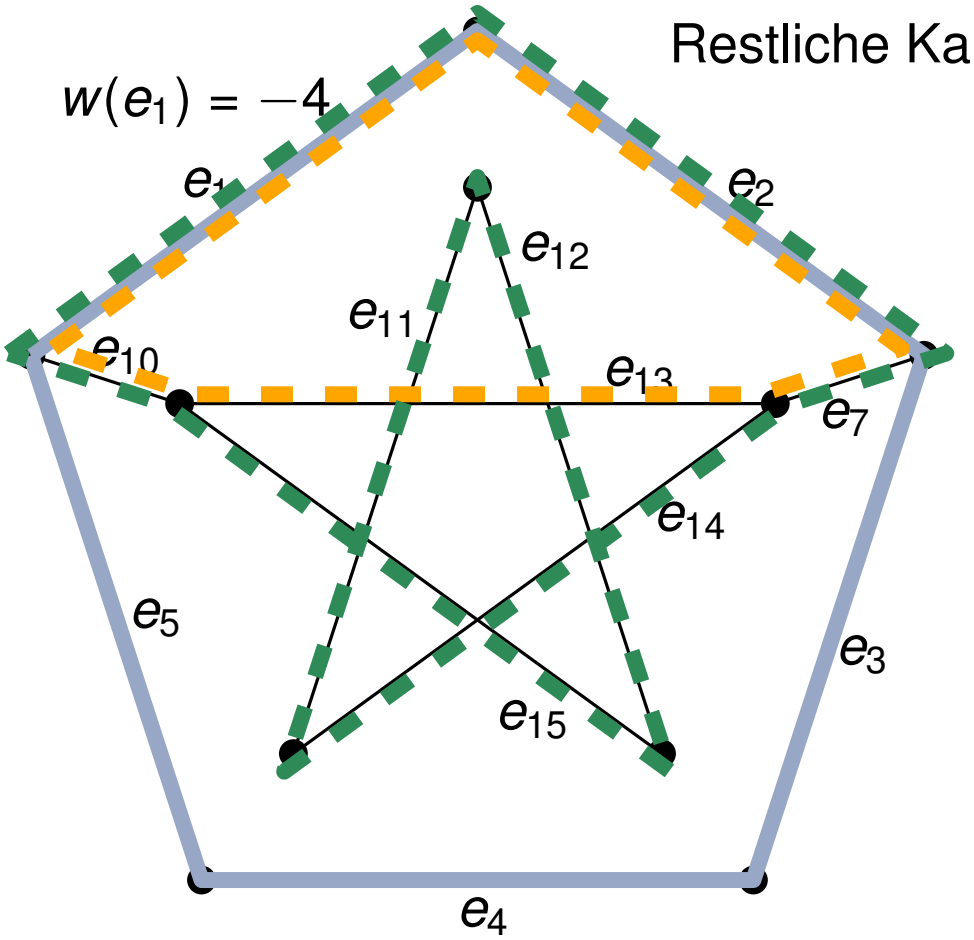
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem 2

Restliche Kanten haben Gewicht 1.



Problem 2



Restliche Kanten haben Gewicht 1.

$MCB = \{C_1, C_2, C_3\}$ mit

$C_1 = \{e_1, e_2, e_3, e_4, e_5\}$

$C_2 = \{e_1, e_2, e_7, e_{13}, e_{10}\}$

$C_3 = \{e_1, e_2, e_7, e_{14}, e_{11}, e_{12}, e_{15}, e_{10}\}$

$w(MCB) = 3$