Institut für Theoretische Informatik
Lehrstuhl Prof. Dr. D. Wagner

# Übungsblatt 5

Vorlesung Theoretische Grundlagen der Informatik im WS 21/22

**Ausgabe:** 17. Dezember 2021
**Abgabe:** 14. Januar 2022 (digital im ILIAS)

Bitte bearbeiten Sie die Aufgaben **handschriftlich** und laden Sie eine eingescannte PDF-Version im Übungsmodul Ihrer ILIAS-Tutoriumsgruppe hoch! Beschriften Sie Ihren handschriftlichen Aufschrieb gut sichtbar mit Name und Matrikelnummer. Nicht handschriftliche oder unbeschriftete Abgaben werden nicht akzeptiert!

## Aufgabe 1

(3 (+1) Punkte)

Die Mensa von Dr. Meta war ein solcher Erfolg, dass er sich dazu entschieden hat, zwei neue Linien $L_1$ und $L_2$ zu eröffnen. Dafür stellt er neue Minions $M$ ein, die an den neuen Linien arbeiten sollen. Sie auf die Linien aufzuteilen ist aber gar nicht so einfach, denn es gibt einige Minions, die sich nicht leiden können. Dabei gibt die Funktion $f\colon M \to 2^M$ für jeden Minion $m \in M$ an, welche Minions $f(m)$ Minion $m$ nicht leiden kann. Sich nicht leiden können beruht immer auf Gegenseitigkeit, also gilt für je zwei Minions $m, m' \in M$: $m \in f(m') \Leftrightarrow m' \in f(m)$. Dr. Meta möchte die Minions nun so auf die beiden Linien aufteilen, dass die Konflikte minimiert werden, also möglichst wenige Minions, die sich nicht leiden können, zusammenarbeiten müssen. Dabei müssen nicht gleich viele Minions an jeder Linie arbeiten.

Wir betrachten nun das äquivalente Maximierungsproblem. Dabei maximieren wir die Anzahl der Minions, die sich nicht leiden können und an verschiedenen Linien arbeiten.

Betrachten Sie den folgenden Greedy-Algorithmus $\mathcal{A}$: Die zwei Linien $L_1, L_2$ sind zunächst leer und wir gehen die Minions $m_1, \ldots, m_n$ in beliebiger Reihenfolge durch. Einen Minion $m_i \in M$ fügen wir dann zu der Linie $L_k$ ($k \in \{1, 2\}$) hinzu, die zum Zeitpunkt der Betrachtung von $m_i$ weniger Minions enthält, die $m_i$ nicht leiden kann.

Zeigen Sie, dass $\mathcal{A}$ ein Approximationsalgorithmus mit relativer Güte 2 ist.

*Hinweis:* Verwenden Sie eine triviale obere Schranke, um den optimalen Wert abzuschätzen.

*Bonusaufgabe:* In einer früheren Version der Aufgabe haben wir das Minimierungsproblem betrachtet. Eine Minimierung der Konflikte innerhalb der Linien ist äquivalent dazu, die Konflikte zwischen den Linien zu maximieren. Warum eignet sich der angegebene Algorithmus trotzdem nicht für eine Approximation des Minimierungsproblems? Zeigen Sie, dass $\mathcal{A}$ keine 2-Approximation für das Minimierungsproblem ist.

## Aufgabe 2

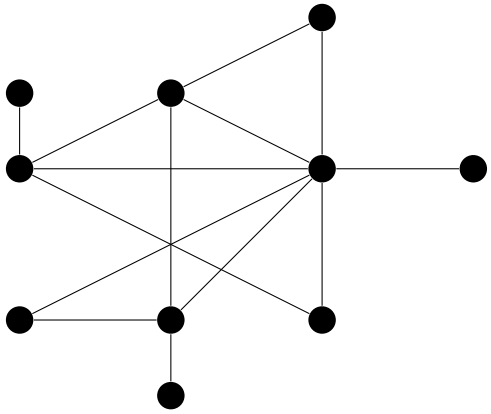
(1 (+ 3) + 3 + 1 = 5 (8) Punkte)

Für einen ungerichteten Graphen $G$ ist ein Subgraph $T$ ein Spannbaum von $G$, wenn $T$ ein Baum ist und alle Knoten von $G$ enthält.

Das Optimierungsproblem MIN-DEGREE-SPANNING-TREE ist wie folgt definiert: Gegeben ist ein zusammenhängender, ungerichteter Graph $G = (V, E)$. Gesucht ist ein Spannbaum $T$ von $G$ mit dem kleinsten Maximalgrad unter allen Spannbäumen von $G$. Der Maximalgrad eines Graphen $G$ ist $\max\{\deg(v) \mid v \in V(G)\}$.

Betrachten Sie außerdem das $\mathcal{NP}$-vollständige Entscheidungsproblem HAMILTON-PFAD: Gegeben ein ungerichteter Graph $G$, gibt es einen Pfad $P$, der jeden Knoten in $G$ genau einmal enthält?

(a) Geben Sie die Lösung des MIN-DEGREE-SPANNING-TREE für die folgende Instanz an und begründen Sie die Optimalität Ihrer Lösung:



(b) *Bonusaufgabe:* Formulieren Sie das zu MIN-DEGREE-SPANNING-TREE zugehörige Entscheidungsproblem und zeigen Sie, dass es $\mathcal{NP}$-vollständig ist. Reduzieren Sie dazu von dem Problem HAMILTON-PFAD.

(c) Zeigen Sie: Falls $\mathcal{P} \neq \mathcal{NP}$, dann gibt es keinen Approximationsalgorithmus für MIN-DEGREE-SPANNING-TREE mit relativer Güte $\alpha < \frac{3}{2}$.

(d) Es gibt aber einen Approximationsalgorithmus $\mathcal{A}$ für MIN-DEGREE-SPANNING-TREE mit absoluter Güte 1. Geben Sie $\mathcal{R}_{\mathcal{A}}^{\infty}$ an. Warum ist das kein Widerspruch zu c)? Überlegen Sie sich dazu, was der Optimalwert einer Instanz sein könnte, für die $\mathcal{A}$ mindestens relative Güte $\frac{3}{2}$ hat.

**Aufgabe 3** (3 + 1 = 4 Punkte)

Sei $\Pi$ ein Minimierungsproblem mit ganzzahliger Optimierungsfunktion. Bezeichne das Entscheidungsproblem, ob $\mathrm{OPT}_{\Pi}(I) \leq K$ für eine gegebene Instanz $I$, als $\Pi_K$. Sei $\Pi_K$ $\mathcal{NP}$-schwer für ein $K \in \mathbb{N}$.

(a) Zeigen Sie, dass unter der Annahme $\mathcal{P} \neq \mathcal{NP}$ kein Approximationsalgorithmus $\mathcal{A}$ für $\Pi$ mit relativer Güte $\mathcal{R}_{\mathcal{A}} < 1 + \frac{1}{K}$ existiert.

(b) Folgern Sie, dass es unter der Annahme $\mathcal{P} \neq \mathcal{NP}$ kein PTAS für $\Pi$ geben kann.

**Aufgabe 4** (1 + 1 + 2 + 1 + 1 = 6 Punkte)

Das Maximierungsproblem MaxCover ist wie folgt definiert: Gegeben seien ein $k \in \mathbb{N}$, ein Universum $U$ mit $|U| = n$ Elementen und eine Menge von $m \geq k$ Teilmengen $\mathcal{S} = \{S_1, \dots S_m\}$ mit $S_i \subseteq U$. Gesucht ist eine Teilmenge $\mathcal{X} \subseteq \mathcal{S}$ mit $|\mathcal{X}| = k$, sodass $\bigcup_{S_i \in \mathcal{X}} S_i$ maximale Kardinalität hat. D.h. wir wollen $k$ Mengen aus $\mathcal{S}$ so wählen, dass diese Mengen die maximale Anzahl an Elementen in $U$ abdecken.

(a) Betrachten Sie die folgende MaxCover-Instanz mit $k = 3$:

$$U = \{0, 1, 2, 3, 4, 5, 6, 7, 8\},\ \mathcal{S} = \{S_1, S_2, S_3, S_4, S_5, S_6\}$$

mit $S_1 = \{2, 3, 6\}$, $S_2 = \{1, 2, 3, 4\}$, $S_3 = \{1, 4, 8\}$, $S_4 = \{0, 4, 5\}$, $S_5 = \{0, 1, 2\}$ und $S_6 = \{5, 6\}$. Geben Sie $k$ Mengen in $\mathcal{S}$ an, die ausgewählt werden, um eine optimale Lösung zu erreichen.

Betrachten Sie den folgenden Greedy-Algorithmus $\mathcal{A}$, der immer die Menge auswählt, die maximal viele der noch nicht abgedeckten Elemente abdeckt.

**Algorithmus 1:** MaxCover-Approximation $\mathcal{A}$

```
X ← ∅                                          // Lösung
Y ← ∅                                  // Bereits abgedeckte Elemente
for i ← 1, ..., k do
    Wähle X_i ∈ S, dass |X_i ∩ (U \ Y)| maximiert wird
    Y ← Y ∪ X_i
    X ← X ∪ {X_i}
```

(b) Wenden Sie $\mathcal{A}$ auf der Instanz aus a) an. Wählen Sie bei mehreren Möglichkeiten die Menge mit dem kleinsten Index. Geben Sie die Reihenfolge an, in der die Mengen gewählt werden.

Im Folgenden zeigen wir, dass $\mathcal{A}$ ein relativer Approximationsalgorithmus mit relativer Güte $(1 - \frac{1}{e})^{-1}$ ist. Sei $Y_{\text{OPT}}$ die Menge der Elemente, die durch eine optimale Lösung abgedeckt wird. Sei $\Delta_i = |Y_{\text{OPT}}| - |Y_i|$, wobei $Y_i$ die Menge der überdeckten Elemente ist nach Auswahl der $i$-ten Menge von $\mathcal{A}$.

(c) Zeigen Sie, dass in jedem Schritt für die Anzahl der neu abgedeckten Elemente

$$|Y_i \setminus Y_{i-1}| \geq \frac{\Delta_{i-1}}{k}$$

gilt.

(d) Zeigen Sie mit Hilfe von vollständiger Induktion, dass

$$\Delta_i \leq \left(1 - \frac{1}{k}\right)^i \cdot |Y_{\text{OPT}}|$$

gilt.

(e) Zeigen Sie, dass $\mathcal{A}$ ein Approximationsalgorithmus mit relativer Güte $(1 - \frac{1}{e})^{-1}$ ist.
*Hinweis: Es gilt* $\left(1 - \frac{1}{n}\right)^n < \frac{1}{e}$ *für alle* $n \in \mathbb{N}$.

**Aufgabe 5** (2 + 4 = 6 Punkte)

Das Maximierungsproblem $d$-WeightedIndSet ist wie folgt definiert: Gegeben sei ein Graph $G = (V, E)$ mit Maximalgrad $d$ und eine Knotengewichtsfunktion $w\colon V \to \mathbb{N}$. Gesucht ist eine unabhängige Menge $S \subseteq V$ mit maximalem Gewicht $w(S) \coloneqq \sum_{v \in S} w(v)$.

Betrachten Sie den folgenden Greedy-Algorithmus $\mathcal{A}$. Dieser beginnt mit $S = \emptyset$ und fügt immer den Knoten mit maximalem Gewicht zu $S$ hinzu. Nachdem ein Knoten $v$ zu $S$ hinzugefügt wird, werden $v$, alle Nachbarn $N(v)$ von $v$ und alle inzidenten Kanten aus dem Graphen gelöscht. Das wird so lange wiederholt, bis der Graph leer ist.

(a) Zeigen Sie, dass $\mathcal{A}$ tatsächlich eine unabhängige Menge ausgibt und in polynomieller Zeit läuft.

(b) Zeigen Sie, dass $\mathcal{A}$ ein Approximationsalgorithmus ist mit relativer Güte $d$ ist.

**Aufgabe 6** (1 + 1 + 2 + 3 = 7 Punkte)

Das Minimierungsproblem DreiecksfreierGraph ist wie folgt definiert: Gegeben ist ein Graph $G = (V, E)$ mit $|V| = n$ und $|E| = m$. Gesucht ist eine kardinalitätsminimale Kantenmenge $E' \subseteq E$, sodass durch Löschen von $E'$ aus Graph $G$ ein Graph $G' = (V, E \setminus E')$ entsteht, der dreiecksfrei ist. Ein Graph ist dreiecksfrei, wenn es keine drei Knoten $u, v, w \in V$ mit $u \neq v$, $v \neq w$, $u \neq w$ gibt, die paarweise zueinander adjazent sind, also ein Dreieck bilden.

Es kann gezeigt werden, dass die Entscheidungsvariante von DreiecksfreierGraph $\mathcal{NP}$-vollständig ist.

In der Vorlesung haben wir gesehen, dass sich jedes $\mathcal{NP}$-vollständige Problem als *ganzzahliges lineares Programm* (engl. *Integer Linear Program*, ILP) darstellen lässt. Um DreiecksfreierGraph als ILP darzustellen, führen wir für jede Kante $e \in E$ eine Variable $x_e \in \{0, 1\}$ ein. Dabei soll $x_e = 1$ bedeuten, dass wir $e$ aus $G$ löschen (also $e \in E'$), und $x_e = 0$, dass wir $e$ nicht löschen.

Im Folgenden ist ein noch unvollständiges ILP für DreiecksfreierGraph gegeben, das wir mit DG-$\mathbb{N}$ bezeichnen:

**Gegeben:**
DreiecksfreierGraph-Instanz $I = (G = (V, E))$
**Variablen:**
Für jede Kante $e \in E$ eine Variable $x_e$
**Nebenbedingungen:**
(I) Für jede Kante $e \in E$: $x_e \in \{0, 1\}$
(II) [ ]
**Zielfunktion:**
Minimiere [ ]

(a) Vervollständigen Sie die zweite Nebenbedingung, die garantiert, dass $G' = (V, E \setminus E')$ dreiecksfrei ist, und die Minimierungsfunktion.

Eine gängige Methode, ILPs zu approximieren, ist die *LP-Relaxierung*. Dabei wird die Beschränkung aufgehoben, dass die Variablen ganzzahlige Werte haben müssen. Das dadurch entstehende *lineare Programm* (LP) lässt sich in Polynomialzeit lösen. Im Fall von SetCover erhalten wir das Problem DG-$\mathbb{R}$. Der einzige Unterschied gegenüber DG-$\mathbb{N}$ ist die Nebenbedingung (I). Die Variablen $x_e$ dürfen nun beliebige reelle Zahlen aus dem Intervall $[0, 1]$ sein. Dementsprechend ist auch die Zielfunktion reellwertig.

Sei im Folgenden $\mathrm{OPT}_{\mathbb{R}}(I)$ der Wert von $L$ für eine optimale Lösung von DG-$\mathbb{R}$ und $\mathrm{OPT}_{\mathbb{N}}(I)$ der Wert von $L$ einer optimalen Lösung von DG-$\mathbb{N}$.

(b) Zeigen Sie, dass für jede DreiecksfreierGraph-Instanz $I = (G = (V, E))$ gilt: $\mathrm{OPT}_{\mathbb{R}}(I) \leq \mathrm{OPT}_{\mathbb{N}}(I)$.

(c) Geben Sie eine DreiecksfreierGraph-Instanz $I = (G = (V, E))$ an, für die $\mathrm{OPT}_{\mathbb{R}}(I) < \mathrm{OPT}_{\mathbb{N}}(I)$ gilt. Geben Sie jeweils eine optimale Lösung an.

Wir betrachten nun folgenden Algorithmus $\mathcal{A}$, der eine (nicht notwendigerweise optimale) Lösung für DG-$\mathbb{N}$ berechnet:

1. Berechne eine optimale Lösung $X = (x_{e_1}, \ldots, x_{e_m})$ mit $x_{e_i} \in [0, 1]$ für DG-$\mathbb{R}$.
2. Generiere eine Lösung $X' = (x'_{e_1}, \ldots, x'_{e_m})$ mit $x'_{e_i} \in \{0, 1\}$ für DG-$\mathbb{N}$:
   Wähle jedes $x'_e$ wie folgt:
$$x'_S = \begin{cases} 1 & \text{falls } x_S \geq \frac{1}{3} \\ 0 & \text{sonst.} \end{cases}$$

(d) Zeigen Sie, dass $\mathcal{A}$ ein 3-Approximationsalgorithmus für DG-$\mathbb{N}$ ist.