Prof. Dr. M. Grohe E. Fluck, A. Riazsadri, J. Feith

Berechenbarkeit und Komplexität

Probeklausur mit Lösungen , **WS22/23**

Nachname: ______________________

Vorname: ______________________

Matrikelnummer: ______________________

Studiengang: ______________________

Hinweise

- Schreiben Sie Ihren Namen und Ihre Matrikelnummer auf **jedes Blatt** (inklusive zusätzliche Blätter).
- Bitte beantworten Sie die Aufgaben auf den Aufgabenblättern. Sprechen Sie uns an, wenn Sie zusätzliches Papier benötigen. Benutzen Sie **kein eigenes Papier** und geben Sie am Ende der Klausur **alle Blätter zusammen mit den Aufgabenblättern** ab.
- Schreiben Sie ausschließlich mit **dokumentenechten Stiften** in **schwarzer** oder **blauer** Farbe. Benutzen Sie insbesondere **keine Bleistifte**.
- Streichen Sie nicht zu wertende Antworten durch. Bei mehreren Antworten für eine Aufgabe wird die schlechteste gewertet.
- Die Bearbeitungszeit beträgt **120 Minuten**. Zum Bestehen der Klausur reichen **60 Punkte**.

Hiermit bestätige ich, dass ich obige Hinweise zur Kenntnis genommen habe und prüfungsfähig bin.

____

(Unterschrift)

Bitte unterhalb dieser Linie nichts eintragen.

	1	2	3	4	5	6	7	8	9
Punkte	/ 12	/ 10	/ 13	/ 23	/ 16	/ 13	/ 17	/ 8	/ 8

$\sum$	/ 120

Aufgabe 1 (Deterministische Turingmaschinen) 2 + 4 + 6 = 12 Punkte

a) Was bedeutet es, dass eine Turingmaschine M eine Sprache L entscheidet?

Lösung:

Eine TM M entscheidet L, wenn M alle $w \in L$ akzeptiert und alle $w \notin L$ verwirft.

b) Definieren Sie den Begriff der Konfiguration einer Turingmaschine $M = (Q, \Sigma, \Gamma, B, q_0, \overline{q}, \delta)$ und geben sie die Bedeutung der Bestandteile einer Konfiguration an.

Lösung:

Eine Konfiguration von M ist ein String $\alpha q \beta$ mit $q \in Q$ und $\alpha, \beta \in \Gamma^*$. Die TM M ist in Konfiguration $\alpha q \beta$, wenn der Inhalt des Bandes von M genau $B^* \alpha \beta B^*$ ist, M sich im Zustand q befindet und der Lese-Schreib-Kopf von M sich auf dem ersten Symbol von β befindet.

c) Geben Sie eine deterministische Turingmaschine über dem Eingabealphabet $\Sigma = \{0, 1\}$ an, welche die folgende Funktion $f\colon \Sigma^* \to \Sigma^*$ berechnet:

$$f(w) = \begin{cases} \varepsilon, & \text{falls } w \in \{0\}^* \text{ oder } \{1\}^*, \\ w, & \text{sonst.} \end{cases}$$

Beschreiben sie kurz die Funktionsweise Ihrer Turingmaschine.

Lösung:

Sei $M = (Q, \Sigma, \Gamma, B, q_{\text{init}}, \overline{q}, \delta)$ mit $Q = \{q_{\text{init}}, \overline{q}, q_0, q_1, q_w\}$ und $\Sigma = \{0, 1\}$ sowie $\Gamma = \{0, 1, B\}$ und δ gegeben durch die folgende Tabelle:

δ	0	1	B
q_{init}	$(q_0, 0, R)$	$(q_1, 1, R)$	$(\overline{q}, B, N)$
q_0	$(q_0, 0, R)$	$(q_w, 1, N)$	$(\overline{q}, B, N)$
q_1	$(q_w, 0, N)$	$(q_1, 1, R)$	$(\overline{q}, B, N)$
q_w	$(q_w, 0, L)$	$(q_w, 1, L)$	$(\overline{q}, B, R)$

Auf der Eingabe ε hält M sofort, womit $\varepsilon \in \{0\}^*$ ausgegeben wird. Auf einer Eingabe $w \neq \varepsilon$ speichert M das erste Zeichen von w im Zustand (q_0 bei 0, q_1 bei 1) und prüft dann, ob alle Zeichen der Eingabe dem gespeicherten Zeichen entsprechen. Ist dies der Fall, so erreicht M das erste Blank und hält, womit die Ausgabe ε ist. Andernfalls begibt M sich mittels q_w wieder an den Anfang des Wortes und hält, womit die Ausgabe w ist.

Aufgabe 2 (Entscheidbare Sprachen) 3 + 7 = 10 Punkte

a) Beweisen oder widerlegen Sie die folgende Behauptung:

"Die Menge der entscheidbaren Sprachen über dem Alphabet $\{0,1\}$ ist unter Teilmengenbildung abgeschlossen. Das heißt: wenn $L \subseteq \{0,1\}^*$ entscheidbar ist, dann ist auch jede Sprache L' mit $L' \subseteq L$ entscheidbar."

Lösung:

Die Behauptung ist falsch. Die Sprache $L = \{0,1\}^*$ ist entscheidbar, aber das Halteproblem H ist unentscheidbar und es gilt $H \subseteq L$.

b) Zeigen Sie, dass die Sprache

$$L = \{\langle M\rangle w \mid M \text{ liest auf Eingabe } w \text{ niemals ein Blank-Symbol}\}$$

entscheidbar ist.

Lösung:

Sei $M = (Q, \Sigma, \Gamma, B, q_0, \overline{q}, \delta)$ eine TM, welche auf Eingabe w nie ein Blank-Symbol liest. Dann kann M insbesondere nur die Bandpositionen besuchen, auf denen zu Anfang w steht. Somit kann M auf Eingabe w höchstens

$$c(M, w) = (|\Gamma| - 1)^{|w|} \cdot |w| \cdot |Q|$$

verschiedene Konfigurationen erreichen.

Wir können also wie folgt L entscheiden:

- Falls die Eingabe nicht die Form $\langle M\rangle w$ hat, verwerfe.
- Ansonsten berechne $k = c(M, w)$.
- Simuliere M für k Schritte und prüfe in jedem Schritt, ob M ein Blank liest. Falls dies passiert, verwerfe; ansonsten akzeptiere.

Da $c(M, w)$ stets endlich ist, terminiert das Verfahren.

Aufgabe 3 (Satz von Rice) 4 + 9 = 13 Punkte

a) Was besagt der Satz von Rice? Benennen Sie insbesondere auch die Voraussetzungen, die erfüllt sein müssen, damit der Satz von Rice angewendet werden kann.

Lösung:

Sei $\mathcal{R}$ die Menge der berechenbaren, partiellen Funktionen und $\mathcal{S}$ eine Teilmenge von $\mathcal{R}$ mit $\emptyset \subsetneq \mathcal{S} \subsetneq \mathcal{R}$. Dann ist die Sprache

$$L(\mathcal{S}) = \{\langle M \rangle \mid M \text{ berechnet eine Funktion aus } \mathcal{S}\}$$

unentscheidbar.

b) Zeigen Sie mit Hilfe des Satzes von Rice, dass die Sprache

$$L' = \{\langle M \rangle \mid \text{ die Sprache } L(M) \text{ ist entscheidbar}\}$$

nicht entscheidbar ist. Zeigen Sie insbesondere, dass die Voraussetzungen des Satzes von Rice erfüllt sind.

Lösung:

Sei $\mathcal{S} = \{f_M \mid \exists f \in \mathcal{R} \text{ so dass } \forall x \in \Sigma^*, f(x) \neq \perp \text{ und } f(x) = 1 \Leftrightarrow f_M(x) = 1w \text{ für ein } w \in \Sigma^*\}$.

Es gilt $\mathcal{S} \neq \emptyset$, da die Funktion, die alle $x \in \Sigma^*$ auf 1 abbildet, in $\mathcal{S}$ liegt.

Außerdem gilt $\mathcal{S} \neq \mathcal{R}$, da die Funktion

$$f(x) = \begin{cases} 1 & \text{wenn } x \in H_\varepsilon, \\ \perp & \text{sonst,} \end{cases}$$

nicht in $\mathcal{S}$ liegt (da H_ε semi-entscheidbar, aber nicht entscheidbar ist).

Somit gilt nach Satz von Rice, dass

$$\begin{aligned} L(\mathcal{S}) &= \{\langle M \rangle \mid M \text{ berechnet eine Funktion aus } \mathcal{S}\} \\ &= \{\langle M \rangle \mid \exists f \in \mathcal{R} \text{ so dass } \forall x \in \Sigma^*, f(x) \neq \perp \text{ und } f(x) = 1 \Leftrightarrow x \in L(M)\} \\ &= \{\langle M \rangle \mid \text{ die Sprache } L(M) \text{ ist entscheidbar}\} = L' \end{aligned}$$

unentscheidbar ist.

Aufgabe 4 (Reduktion) 4 + 5 + 4 + 10 = 23 Punkte

a) Was bedeutet es formal, dass eine Sprache $L_1 \subseteq \Sigma^*$ auf eine Sprache $L_2 \subseteq \Sigma^*$ reduzierbar ist ($L_1 \leq L_2$)?

Lösung:

Eine Sprache L_1 ist auf L_2 reduzierbar, falls es eine berechenbare Funktion $f\colon \Sigma^* \to \Sigma^*$ gibt, sodass für alle $x \in \Sigma^*$ genau dann $x \in L_1$ gilt, wenn auch $f(x) \in L_2$ gilt.

b) Seien $L_1, L_2 \subseteq \{0,1\}^*$ zwei Sprachen so, dass $L_1 \leq L_2$ und $L_2 \leq \overline{L_1}$ gilt. Beweisen oder widerlegen Sie die folgende Behauptung:

"Wenn L_2 semi-entscheidbar ist, dann ist L_1 entscheidbar."

Lösung:

Aus $L_1 \leq L_2$ folgt nach Übung auch $\overline{L_1} \leq \overline{L_2}$ und ähnlich $\overline{L_2} \leq L_1$ aus $L_2 \leq \overline{L_1}$. Somit haben wir

$$L_1 \leq L_2 \leq \overline{L_1} \leq \overline{L_2} \leq L_1$$

und aus der Transitivität der Reduktion folgt damit $\overline{L_1} \leq L_2$. Da L_2 semi-entscheidbar ist, sind also L_1 und $\overline{L_1}$ semi-entscheidbar, weswegen L_1 entscheidbar ist.

c) Definieren Sie das spezielle Halteproblem H_ε und sein Komplement $\overline{H_\varepsilon}$:

$$H_\varepsilon = \{ \qquad\qquad\qquad\qquad\qquad\qquad \}$$

$$\overline{H_\varepsilon} = \{ \qquad\qquad\qquad\qquad\qquad\qquad \}$$

Lösung:

$$H_\varepsilon = \{\langle M\rangle \mid M \text{ hält auf } \varepsilon\}$$

$$\overline{H_\varepsilon} = \{0,1\}^* \setminus H_\varepsilon = \{w \in \{0,1\}^* \mid w \text{ keine Gödelnummer oder } w = \langle M\rangle \text{ und } M \text{ hält nicht auf } \varepsilon\}$$

---

d) Wir betrachten die Sprache

$$L_{11} = \{\langle M\rangle \mid M \text{ akzeptiert mindestens eine Eingabe, die mit 11 beginnt}\}.$$

Zeigen Sie, dass das spezielle Halteproblem auf L_{11} reduzierbar ist, d.h., dass $H_\varepsilon \leq L_{11}$ gilt. Beweisen Sie insbesondere die Korrektheit Ihrer Reduktion.

Lösung:

Wir zeigen $H_\varepsilon \leq L_{11}$.

Sei M eine TM. Wir definieren $f(\langle M\rangle) = \langle M'\rangle$, wobei M' eine TM mit folgendem Verhalten ist:

- Falls die Eingabe 11 ist, löscht M' den Bandinhalt und simuliert M auf ε. Falls (die Simulation von) M auf ε hält, so akzeptiert M'.
- Andernfalls verwirft M' sofort.

Weiterhin definieren wir $f(w) = w$ für alle Eingaben w, welche keine Gödelnummern sind. Dann ist f berechenbar und es gilt:

- Falls w keine Gödelnummer ist, so gilt $w \notin H_\varepsilon$ und $f(w) = w \notin L_{11}$.
- Falls $w = \langle M\rangle$ für eine TM M ist, so hält M auf ε genau dann, wenn M' die Eingabe 11 akzeptiert und alle anderen Eingaben verwirft. Folglich gilt $\langle M\rangle \in H_\varepsilon$ genau dann, wenn $f(w) = \langle M'\rangle \in L_{11}$ gilt.

Somit gilt $H_\varepsilon \leq L_{11}$.

Aufgabe 5 (Programmiersprachen) 4 + 2 + 8 + 2 = 16 Punkte

a) Geben Sie die drei syntaktischen Regeln zur Bildung von WHILE-Programmen an:

-
-
-

Lösung:

- Für $c \in \{-1, 0, 1\}$ ist $x_i := x_j + c$ ein WHILE-Programm.
- Wenn P_1, P_2 WHILE-Programme sind, dann ist auch $P_1; P_2$ ein WHILE-Programm.
- Wenn P ein WHILE-Programm ist, dann ist auch

  $$\text{WHILE } x_k \neq 0 \text{ DO } P \text{ END}$$

  ein WHILE-Programm.

b) Was ist der Unterschied zwischen den in WHILE- und LOOP-Programmen verwendeten Schleifenarten? Warum ist dieser Unterschied wichtig?

Lösung:

In WHILE-Schleifen darf die Schleifenvariable beliebig verändert werden, während in LOOP-Schleifen die Schleifenvariable nicht vorkommen darf; sie wird nach jedem Schleifendurchlauf dekrementiert. Insbesondere terminieren LOOP-Schleifen (und damit LOOP-Programme) stets, was bei WHILE-Schleifen (und damit WHILE-Programmen) nicht der Fall ist. Weiterhin ist dadurch das Wachstum der von LOOP-berechenbaren beschränkt, weswegen z.B. die Ackermannfunktion nicht LOOP-berechenbar ist (wohl aber WHILE-berechenbar).

c) Geben Sie ein LOOP-Programm an, welches das Minimum zweier Zahlen berechnet.

Die Eingabe sei dabei in den Variablen x_1 und x_2 gegeben; die Ausgabe soll in der Variable x_0 gespeichert werden.

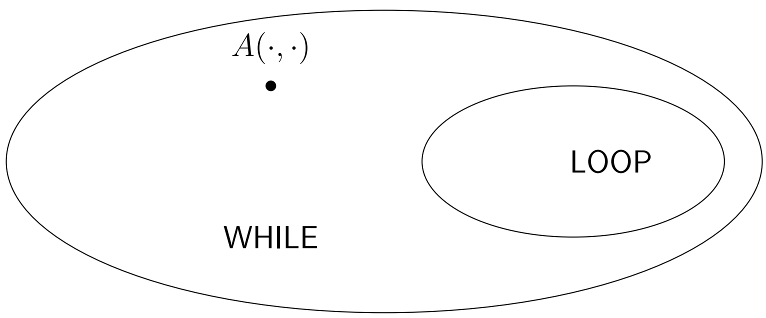
Lösung:

$$x_3 = x_2; \tag{1}$$
$$x_0 = x_2; \tag{2}$$
$$\text{LOOP } x_1 \text{ DO} \tag{3}$$
$$\quad x_3 = x_3 - 1 \tag{4}$$
$$\text{END}; \tag{5}$$
$$\text{LOOP } x_3 \text{ DO} \tag{6}$$
$$\quad x_0 = x_1 \tag{7}$$
$$\text{END} \tag{8}$$

Das Programm berechnet zunächst die mod. Subtraktion $x_3 = \max\{0, x_2 - x_1\}$. Ist in Zeile (6) $x_3 > 0$, so ist $x_2 > x_1$ und die Ausgabe wird auf x_1 gesetzt. Ansonsten gilt $x_1 \geq x_2$ und Zeile (7) wird nicht ausgeführt, womit das Programm x_2 ausgibt.

d) Stellen Sie durch ein Mengendiagramm die Beziehung zwischen der Klasse der LOOP-berechenbaren Funktionen und der Klasse der WHILE-berechenbaren Funktionen dar. Zeichnen Sie die Ackermannfunktion $A(\cdot,\cdot)$ in Ihr Diagramm ein.

Lösung:



Aufgabe 6 (Komplexitätstheorie) $3 + 2 + 3 + 3 + 2 = 13$ Punkte

a) Definieren Sie die Klasse EXPTIME.

Lösung:

EXPTIME ist die Klasse aller Entscheidungsprobleme, welche durch eine deterministische TM mit Laufzeit $2^{p(n)}$ für ein Polynom p entschieden werden können.

b) Definieren Sie, wann ein Entscheidungsproblem NP-vollständig ist.

Lösung:

Ein Entscheidungsproblem A ist NP-vollständig, wenn gilt:

- $A \in \mathsf{NP}$ und
- A ist NP-schwer, dass bedeutet für alle $B \in \mathsf{NP}$ gilt $B \leq_p A$.

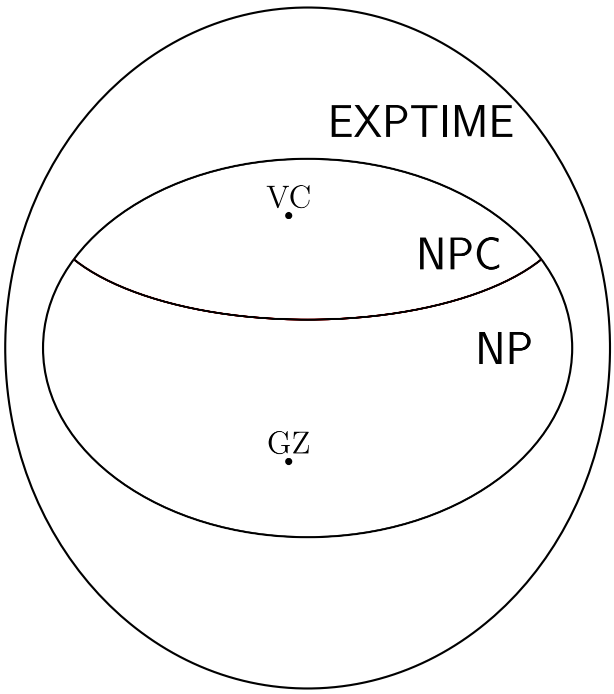
c) Zeigen oder widerlegen Sie: Das Halteproblem ist NP-vollständig.

Lösung:

Das Halteproblem ist nicht NP-vollständig, da das Halteproblem nicht berechenbar ist und somit nicht in NP liegt.

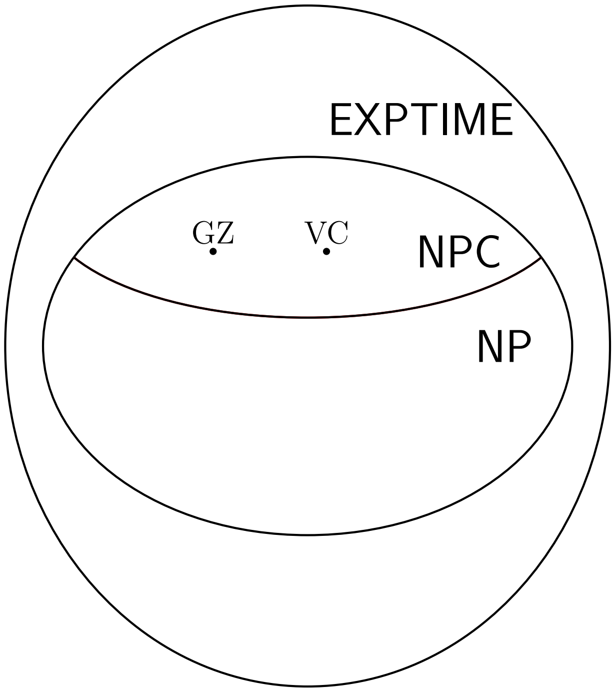
d) Nehmen Sie für diesen Aufgabenteil an, dass $\mathsf{P} \neq \mathsf{NP} \neq \mathsf{EXPTIME}$ gilt. Stellen Sie durch ein Mengendiagramm die Beziehunen zwischen den Klassen NP, EXPTIME und der Klasse NPC der NP-vollständigen Probleme dar. Zeichnen Sie die Probleme Vertex Cover (VC) und Graphzusammenhang (GZ) in Ihr Diagramm ein.

Lösung:



e) Nehmen Sie nun an, dass $\mathsf{P} = \mathsf{NP}$ und $\mathsf{NP} \neq \mathsf{EXPTIME}$ gelten. Stellen Sie durch ein Mengendiagramm die Beziehunen zwischen den Klassen NP, EXPTIME und der Klasse NPC der NP-vollständigen Probleme dar. Zeichnen Sie die Probleme VERTEX COVER (VC) und GRAPHZUSAMMENHANG (GZ) in Ihr Diagramm ein.

Lösung:


EXPTIME
GZ
VC
NPC
NP

Aufgabe 7 (NP-Vollständigkeit) 4 + 13 = 17 Punkte

a) Wir betrachten das Entscheidungsproblem Set Cover:

> Set Cover
>
> Eingabe : Eine endliche Menge U, eine Menge $\mathcal{S} \subseteq \mathrm{Pot}(U)$ und $k \in \mathbb{N}$.
>
> Frage : Gibt es eine Teilmenge $\mathcal{C} \subseteq \mathcal{S}$ so, dass $|\mathcal{C}| \leq k$ und jedes Element von U durch $\mathcal{C}$ überdeckt ist, d.h.
>
> $$U = \bigcup_{V \in \mathcal{C}} V?$$

Lösen Sie die folgenden Set Cover-Instanzen. Falls es sich um eine Ja-Instanz handelt, geben Sie eine entsprechende Menge $\mathcal{C}$ an; andernfalls begründen Sie kurz, warum keine solche Menge existiert.

i) $U = \{1, 2, 3, 4, 5, 6\}$,

$$\mathcal{S} = \{\{1, 2, 4\}, \{1, 2, 5\}, \{2, 3, 5\}, \{2, 4, 5\}, \{1, 3\}, \{2, 6\}\},$$

und $k = 3$.

ii) $U = \{1, 2, 3, 4, 5, 6\}$,

$$\mathcal{S} = \{\{1, 2, 4\}, \{1, 2, 5\}, \{2, 3, 5\}, \{2, 4, 5\}, \{1, 3\}, \{2, 6\}\},$$

und $k = 2$.

Lösung:

i) $U = \{1, 2, 3, 4, 5, 6\}$,

$$\mathcal{S} = \{\{1, 2, 4\}, \{1, 2, 5\}, \{2, 3, 5\}, \{2, 4, 5\}, \{1, 3\}, \{2, 6\}\},$$

und $k = 3$.

Ja-Instanz

$\mathcal{C} = \{\{1, 2, 4\}, \{2, 3, 5\}, \{2, 6\}\}$

ii) $U = \{1, 2, 3, 4, 5, 6\}$,

$$\mathcal{S} = \{\{1, 2, 4\}, \{1, 2, 5\}, \{2, 3, 5\}, \{2, 4, 5\}, \{1, 3\}, \{2, 6\}\},$$

und $k = 2$.

Nein-Instanz

Es kann kein solches $\mathcal{C}$ geben, da das Element 6 nur in $\{2, 6\}$ vorkommt und keine der anderen Teilmengen 4 Elemente abdeckt.

---

b) Zeigen Sie, dass Set Cover NP-vollständig ist.

Beachten Sie dabei beide Aspekte der NP-Vollständigkeit und beweisen Sie Korrektheiten und polynomielle Beschränktheiten, sofern diese in Ihrem Beweis wichtig sind.

Lösung:

Set Cover $\in$ NP:

Zertifikat: String $y \in \{0, 1\}^{|\mathcal{S}|}$

Verifizierer:

- überprüfe dass y höchstens k 1en hat
- erzeuge String z aus 0en der Länge $|U|$
- iteriere über y und wenn $y_i = 1$, dann setze alle $z_j = 1$, für die gilt dass $j \in S_i$ (S_i sei das i-te Element von $\mathcal{S}$)
- überprüfe ob z eine 0 enthält, wenn ja verwerfe sonst akzeptiere

Das Zertifikat hat offensichtlich Länge polynomiell in der Eingabe und der Verifizierer hat offensichtlich Laufzeit polynomiell in der Eingabe.

Set Cover ist NP-schwer über Reduktion Vertex Cover $\leq_p$ Set Cover:

Sei $G = (V, E)$ zusammen mit $k \in \mathbb{N}$ eine Eingabe für Vertex Cover. Wir konstruieren eine Set Cover-Instanz mit $U = E$, $\mathcal{S} = \{E_v \mid v \in V\}$, wobei $E_v = \{e \in E \mid v \text{ Endpunkt von } e\}$ und lassen den Parameter k unverändert.

Diese Reduktion ist offensichtlich polynomiell berechenbar.

Korrektheit:

Wenn $G = (V, E)$ ein Vertex Cover C der Größe maximal k hat, so ist $\mathcal{C} = \{E_v \mid v \in C\}$ ein Set Cover von $U = E$ der Größe maximal k, da C mindestens einen Endpunkt jeder Kante enthält.

Wenn es andersrum eine Teilmenge $\mathcal{C} \subseteq \mathcal{S}$ der Größe maximal k gibt, die ein Set Cover von $U = E$ ist, dann ist $C = \{v \in V \mid E_v \in \mathcal{C}\}$ ein Vertex Cover von G der Größe maximal k, da für jede Kante $e \in E$ mindestens ein $v \in V$ existiert, sodass $v \in e \in E_v$.

Aufgabe 8 (Optimierungsprobleme) 8 Punkte

Wir betrachten das Entscheidungsproblem INDEPENDENT SET, was aus den Übungen bekannt ist:

> INDEPENDENT SET
>
> Eingabe : Ein Graph G und $k \in \mathbb{N}$.
>
> Frage : Gibt es eine unabhängige Knotenmenge I der Größe mindestens k in G, d.h. existiert $I \subseteq V(G)$ mit $|I| \geq k$ so, dass für alle $v, w \in I$ auch $vw \notin E(G)$ gilt?

Beweisen Sie die folgende Aussage: Wenn INDEPENDENT SET in P liegt, so kann man auch die zugehörige Optimierungsvariante (gegeben ein Graph G, berechne eine unabhängige Knotenmenge I von G maximaler Größe) in polynomieller Zeit lösen.

Lösung:

Sei A ein Algorithmus für INDEPENDENT SET mit Laufzeitschranke $p(n)$, für ein Polynom p.

Im folgenden verwenden wir folgende Notation: Sei $v \in V$, $G \setminus \{v\}$ ist der Graph wenn man den Knoten v und alle ausgehenden Kanten löscht.

Ein Algorithmus C für die Optimierungsvariante O-IS geht wie folgt vor:

(1) Suche mittels A und linearer Suche den optimalen Wert $k_{\text{opt}} \leq |V|$.

(2) Für alle $v \in V$, rufe A mit Eingabe $G \setminus \{v\}, k_{\text{opt}}$ auf:

- wenn A akzeptiert, setze $G = G \setminus \{v\}$.

(3) Gebe die verbliebenen Knoten aus.

Wenn der Algorithmus A in Schritt (2) akzeptiert, so existiert ein Independent Set in $G \setminus \{v\}$ der Größe k, also enthält der verbliebene Graph ein Independent Set der Größe k. Desweiteren hat der Algorithmus A in Schritt (2) auf allen verbliebenen Knoten verworfen, also müssen alle diese Knoten ein Independent Set bilden (ansonsten hätte man einen weiteren Knoten entfernen können). Alles in allem gibt der Algorithmus ein Independent Set der Größe k_{opt} aus und in Schritt (1) wird der größtmögliche Wert für k_{opt} berechnet.

Sei n die Größe der Eingabe. In Schritt (1) wird A maximal n mal aufgerufen, also ist die Laufzeit beschränkt durch $O(n \cdot p(n))$, wie oben beschrieben. Schritt (2) wird $|V|$ mal aufgerufen und in jeder Iteration wird A aufgerufen auf einer Eingabe kleiner als $O(n)$, also ist die Laufzeit der Schleife in Schritt (2) beschränkt durch

$O(n \cdot p(n))$. Da am Ende maximal alle Knoten von G ausgegeben werden, ist die Laufzeit von Schritt (3) beschränkt durch $O(n)$. Alles in allem ist die Laufzeit von C beschränkt durch $O(n \cdot p(n))$.

Aufgabe 9 (Abschlusseigenschaften von NP) 4 + 4 = 8 Punkte

a) Ist die Klasse NP unter Schnitt abgeschlossen? Begründen Sie Ihre Antwort.

Lösung:

Ja. Seien $L_1, L_2 \in \mathsf{NP}$, dann existieren insbesondere entsprechende Verifizierer V_1, V_2 für diese Sprachen. Wir konstruieren einen NTM N mit polynomiell beschränkter Laufzeit für $L_1 \cap L_2$ mit folgendem Verhalten:

- Zunächst "rät" nichtdeterministisch N Zertifikate c_1 und c_2 für die Eingabe x.
- Dann simuliert N den Verifizierer V_1 auf der Eingabe $c_1\#x$ und den Verifizierer V_2 auf der Eingabe $c_2\#x$.
- Falls beide Verifizierer ihre Eingaben akzeptieren, so akzeptiert N. Andernfalls verwirft N.

Offensichtlich ist die Laufzeit von N polynomiell, da die Länge der Zertifikate und die Laufzeiten der beiden Verifizierer polynomiell in der Eingabelänge sind. Weiterhin akzeptiert N genau dann, wenn es entsprechende Zertifikate gibt, welche die Mitgliedschaft von x in L_1 und L_2 zertifizieren, d.h. wenn $x \in L_1 \cap L_2$ gilt.

Es folgt $L_1 \cap L_2 \in \mathsf{NP}$.

b) Ist die Klasse NP unter Vereinigung abgeschlossen? Begründen Sie Ihre Antwort.

Lösung:

Ja. Seien $L_1, L_2 \in \mathsf{NP}$, dann existieren insbesondere entsprechende Verifizierer V_1, V_2 für diese Sprachen. Wir konstruieren einen entsprechenden Verifizierer V für $L_1 \cup L_2$ wie folgt:

Gegeben einer Eingabe $c\#x$ führt V die beiden Verifizierer V_1 und V_2 auf $c\#x$ aus und akzeptiert, wenn mindestens einer der beiden Verifizierer V_1 und V_2 akzeptiert.

Offensichtlich akzeptiert V, wenn $x \in L_1$ oder $x \in L_2$ gilt. Aus den Eigenschaften von V_1 und V_2 als Polynomialzeit-Verifizierer für L_1 und L_2 ergibt sich sofort, dass auch V in Polynomialzeit in $|x|$ läuft und die Länge des Zertifikates polynomiell in $|x|$ ist.

Es folgt $L_1 \cup L_2 \in \mathsf{NP}$.