

Musterlösung zur Klausur „Algorithmtheorie“ vom 31.01.2005

Aufgabe 1 (Fragenkatalog)

[30 Punkte]

(a) Korrekt oder nicht?

JA NEIN

Wenn $L = L_M$ für eine Turingmaschine M , die nicht auf allen Eingaben hält, dann ist L nicht rekursiv.

NEIN Grob gesprochen: Zu jeder rekursiven Sprache gibt es viele TMn, und viele davon halten nicht auf allen Eingaben. Genauer: Nach Lemma 1.1.18 kann man zu jeder TM M' eine TM M'' konstruieren, die genau auf den Eingaben hält (und akzeptiert), die von M' akzeptiert werden, und auf den anderen nicht hält. Wenn nun $L_{M'} \neq \Sigma^*$ rekursiv ist, dann ist $L_{M'} = L_{M''}$ und M'' hält nicht auf allen Eingaben.

Wenn $K \leq L$, dann ist L nicht rekursiv.

JA Korollar 1.9.9.(a) und K ist nicht rekursiv (Satz 1.9.4(a)) und Reduktionsmethode (Korollar 1.9.9.(a)).

Wenn $L \leq \overline{H}$, dann ist L nicht rekursiv aufzählbar.

NEIN Die Reduktion geht in die falsche Richtung. Konkretes Gegenbeispiel: Die rekursive Sprache $L = \Sigma^*$ erfüllt $L \leq \overline{H}$, z.B. mit der Reduktionsfunktion $f: x \mapsto \varepsilon \notin H$ für $x \in \Sigma^*$.

(b) Korrekt oder nicht?

JA NEIN

Zu jeder TM M gibt es eine Konstante d , so dass $t_M(x) \leq 2^{|x|^d}$ für jedes $x \in \Sigma^*$.

NEIN Erstens kann es sein, dass M auf x nicht hält — dann ist $t_M(x) = \infty$. Selbst wenn M auf allen Eingaben hält, gibt es keine solche exponentielle Zeitbeschränkung. Es ist leicht, eine TM zu bauen, die zu Eingabe x eine Ausgabe berechnet, die aus $2^{2^{|x|}}$ Einsen besteht. Offenbar ist dann $2^{2^{|x|}} \leq t_M(x) < \infty$, also $t_M(x) > 2^{|x|^d}$ für $|x|$ genügend groß.

Zu jeder TM M gibt es eine Registermaschine M' , die das Ein-/Ausgabeverhalten von M simuliert.

JA (Satz 1.5.1)

M sei eine TM. Wenn es kein Polynom $p(n)$ gibt mit $t_M(x) \leq p(|x|)$ für jedes $x \in \Sigma^*$, dann ist $L_M \in \mathbf{NP}$.

NEIN Für die Definition von **NP** siehe Def. 2.2.2. — Ein Beispiel: K ist nicht rekursiv, also nicht in **NP**. Sei nun M eine TM mit $L_M = K$. Weil M auf manchen Eingaben nicht hält, gilt für kein Polynom $p(n)$, dass $t_M(x) \leq p(|x|)$.

(c) Korrekt oder nicht?

JA NEIN

Wenn L rekursiv ist, dann ist $\Sigma^* - L^*$ rekursiv.

JA Wenn L rekursiv ist, dann auch L^* (Satz 1.4.1(e)), und dann auch $\Sigma^* - L^*$ (Satz 1.4.1(a)).

Wenn L_1 und L_2 rekursiv aufzählbar sind, dann ist auch $\overline{L_1 \cup L_2}$ rekursiv aufzählbar.

NEIN Grob: Die r.a. Sprachen sind nicht unter Komplement abgeschlossen. — Konkret wählt man $L_1 = L_2 = K$. Diese Sprache ist r.a., aber $\overline{K} = \overline{L_1 \cup L_2}$ nicht.

Wenn L rekursiv aufzählbar ist, dann gibt es eine 1-Band-TM M mit $L = H_M$.

JA (Satz 1.2.6 und Lemma 1.1.18)

(d) Korrekt oder nicht?

JA NEIN

[] [] Wenn $L = L(G)$ für eine Chomsky-0-Grammatik G ist, dann gibt es eine deterministische Turingmaschine M mit $L = L_M$.

JA (Satz 1.3.16)

[] [] Es gibt eine Chomsky-0-Grammatik G mit $H = L(G)$.

JA H ist r.a. Dann wende Satz 1.3.16. an.

[] [] Es gibt eine Chomsky-0-Grammatik G mit $\overline{H} = L(G)$.

NEIN \overline{H} ist nicht r.a. (Satz 1.9.5.(b)). Nach Satz 1.3.16. gilt dann auch $\overline{H} \notin \mathcal{L}_0$.

(e) An dem Berechnungsbaum $CT(M, x)$ einer nichtdeterministischen TM M zu einem Input x kann man erkennen, ...

JA NEIN

[] [] ... ob L_M rekursiv ist.

NEIN Der Berechnungsbaum gibt nur Auskunft darüber, wie M auf der einen Eingabe x rechnet. Über andere Inputs sagt er nichts.

[] [] ... ob x von M akzeptiert wird oder nicht.

JA M akzeptiert x genau dann wenn $CT(M, x)$ ein Blatt hat, das einer akzeptierenden Haltekonfiguration entspricht.

[] [] ... ob jede Berechnung von M auf x maximal $50|x|^2 + 10$ Schritte macht.

JA Jede Berechnung entspricht einem Weg von der Wurzel zu einem Blatt; die Schrittzahl entspricht der Tiefe des Blattes.

(f) Korrekt oder nicht?

JA NEIN

[] [] Wenn $L \in \mathbf{NP}$ und $L \leq_p L'$, dann ist L' **NP**-vollständig.

NEIN Zwei Gründe: Erstens ist nicht klar, dass L' in **NP** ist; zweitens muss für die Reduktionsmethode L nicht nur in **NP**, sondern **NP**-vollständig sein. Gegenbeispiel: Wähle $L = L' = \emptyset \in \mathbf{NP}$. Dann $L \leq_p L$, aber \emptyset ist nicht **NP**-vollständig.

[] [] Wenn $L_{\text{SAT}} \leq L$, dann ist L **NP**-vollständig.

NEIN Es ist nicht klar, dass L in **NP** ist. Dies muss man für die Eigenschaft „**NP**-vollständig“ aber haben. — Es gibt auch ein konkretes Gegenbeispiel (das man für die korrekte Beantwortung der Frage aber nicht kennen musste). Setze $L = H$. Wähle eine Turingmaschine M mit $L_{\text{SAT}} = H_M$. Dann ist $x \mapsto \langle M \rangle x$ eine polynomialzeit-berechenbare Reduktionsfunktion von L_{SAT} auf H . Aber H ist nicht rekursiv, also nicht in **NP**, also nicht **NP**-vollständig.

[] [] Wenn L **NP**-vollständig ist, dann gilt $L_{\text{SAT}} \leq L$.

JA $L_{\text{SAT}} \in \mathbf{NP}$ nach Satz 2.5.9 und Teil (i) des Beweises zu diesem Satz. Nach Definition 2.4.5(ii) gilt $L_{\text{SAT}} \leq L$.

(g) Korrekt oder nicht?

JA NEIN

[] [] Wenn M eine $p(n)$ -zeitbeschränkte k -Band TM ist, wobei $p(n)$ ein Polynom ist, dann existiert ein Polynom $q(n)$ und eine $q(n)$ -zeitbeschränkte 1-Band TM M' mit $f_M = f_{M'}$.

JA Siehe Satz 1.2.8(c). Die 1-Band-Maschine M' ist $O(p(n)^2)$ -zeitbeschränkt, und $p(n)^2$ ist auch ein Polynom.

[] [] Jede TM M mit $L_M = L_{\text{Clique}}$ ist nichtdeterministisch.

NEIN Es gibt deterministische TM für die Sprache L_{Clique} , die man zum Beispiel aus dem Algorithmus erhält, einfach jede der 2^m Teilmengen der Knotenmenge $\{1, \dots, m\}$ darauf zu testen, ob sie eine Clique bilden. Die Laufzeit einer solchen TM ist natürlich exponentiell. (Hinweis: In Bemerkung 1.3.9 wurde gesagt, dass man aus jeder deterministischen TM eine *äquivalente* nichtdeterministische erhalten kann. Das heißt nicht, dass die Menge der TM eine Teilmenge der NTM ist.)

[] [] Wenn $L \in \mathbf{P}$, dann ist jede TM M mit $L = L_M$ polynomiell zeitbeschränkt.

NEIN Zum Beispiel ist $\emptyset \in \mathbf{P}$. Betrachte die TM M , die auf Input x folgendes tut: Schreibe $1^{2^{|x|}}$ auf das Band, dann halte verwerfend. Natürlich ist $L_M = \emptyset$, aber M ist nicht polynomiell zeitbeschränkt.

(h) Welche(s) der folgenden Probleme ist/sind entscheidbar?

JA NEIN

[] [] Zu einem gegebenen ASCII-Text P entscheide, ob P ein syntaktisch korrektes C-Programm ist.

JA Jeder C-Compiler enthält ein Verfahren, das dies entscheidet.

[] [] Zu zwei gegebenen C-Programmen P und P' entscheide, ob P und P' das gleiche Ein-/Ausgabeverhalten besitzen.

NEIN Siehe Abschnitt 1.9.2 im Skript, Beispiel (iii).

[] [] Zu einer (als Programm) gegebenen Registermaschine M entscheide, ob M die Additionsfunktion $\mathbb{N}^2 \rightarrow \mathbb{N}$, $(a_1, a_2) \mapsto a_1 + a_2$, berechnet.

NEIN Siehe Beispiel 1.9.3(iii). Setze $g := +$. Wenn es ein Entscheidungsverfahren für Registermaschinen gäbe, dann auch für Turingmaschinen, weil man Turingmaschinen auf Registermaschinen simulieren kann.

(i) Korrekt oder nicht?

JA NEIN

[] [] Die Sprache L_{PKP} zum Postschen Korrespondenzproblem ist rekursiv aufzählbar.

JA Satz 1.11.6.

[] [] Die Sprache $\{\langle M \rangle \mid H_M = \Sigma^*\}$ ist rekursiv aufzählbar.

NEIN Satz 1.9.6.

[] [] Die Sprache L_{TAUT} aller (aussagenlogischen) Tautologien ist rekursiv.

JA Algorithmus: Gegeben φ , erstelle eine Wahrheitstafel für alle 2^n Belegungen für die n Variablen, die in φ vorkommen. Akzeptiere genau dann, wenn das Resultat in allen Fällen „wahr“ ist. Nach der Churchschen These erhält man eine TM, die diesen Algorithmus ausführt, und auf jeder Eingabe hält.

(j) Sind die folgenden Fragen entscheidbar?

JA NEIN

[] [] Gegeben eine kontextfreie Grammatik G , ist $L(G) \neq \emptyset$?

JA Skript, Seite 132 unten, und AFS-Vorlesung.

[] [] Gegeben kontextfreie Grammatiken G_1, G_2 , ist $L(G_1) \cap L(G_2) \neq \emptyset$?

NEIN Satz 1.12.1.

[] [] Gegeben eine kontextfreie Grammatik G , ist $L(G)$ regulär?

NEIN Satz 1.12.4.(d).

Aufgabe 2 (Definitionen von Begriffen und Konzepten aus der Vorlesung)

[12 Punkte]

- (a) Definieren Sie: Eine Sprache $L \subseteq \Sigma^*$ ist rekursiv. Siehe Definition 1.1.16(b).
- (b) Definieren Sie die Haltesprache H und die Diagonalsprache K . Siehe Definition 1.9.2(b)+(c).
- (c) Definieren Sie: $L_1 \leq L_2$. Siehe Definition 1.9.6.
- (d) Definieren Sie, wann eine nichtdeterministische TM polynomiell zeitbeschränkt heißt. Siehe Definition 2.2.1(b).
- (e) Definieren Sie: L ist **NP**-vollständig. Siehe Definition 2.4.5(a).
- (f) Formulieren Sie die Reduktionsmethode zum Beweis der Behauptung “ L ist **NP**-vollständig”. Siehe Lemma 2.6.1.

Aufgabe 3 (Sätze aus der Vorlesung)

[12 Punkte]

Zeigen Sie:

- (a) \bar{K} ist nicht rekursiv aufzählbar.
- (b) Wenn $\mathbf{P} \cap \mathbf{NPC} \neq \emptyset$, dann gilt $\mathbf{NP} \subseteq \mathbf{P}$.

Lösung: Für (a) siehe Beweis von Satz 1.9.4. und für (b) siehe Beweis von Satz 2.4.7.

Aufgabe 4 (Abschlußigenschaften von r.a. Sprachen mittels Determinismus)

[10 Punkte]

Sei $M = (Q, \{0, 1\}, \Gamma, B, q_0, F, \delta)$ eine 1-Band-Turingmaschine. Betrachten Sie die Sprache

$$L_{\text{Flip}(M)} := \{w \in \{0, 1\}^* \mid \exists \text{ Zerlegung } w = uv \text{ mit } u \neq \varepsilon \text{ und } \text{Flip}(u)v \in L_M\}$$

wobei $\text{Flip}(b_0 \dots b_n) := \bar{b}_0 \dots \bar{b}_n$ mit $\bar{1} := 0$ und $\bar{0} := 1$.

Beschreiben Sie präzise die Arbeitsweise und Organisation einer **deterministischen** Mehrband-Turingmaschine N mit:

$$L_N = L_{\text{Flip}(M)}$$

Lösung: Die Mehrband-TM arbeite auf Eingabe $w \in \{0, 1\}^*$ wie folgt:

- 1: **Falls** $w = \varepsilon$, so verwerfe.
- 2: **Markiere** das erste Zeichen der Eingabe mit $*$. \triangleright Interpretation: Das *aktuelle* u entspricht dem Präfix von w , das mit Marke $*$ endet, und das *aktuelle* v dem Restwort.
- 3: Initialisiere Band 2 mit dem *Zähler* $t := 1 \equiv \text{bin}(1)$.
- 4: **Wiederhole**
- 5: Erzeuge $\text{Flip}(u)$ auf Band 2 aus dem aktuellen u und kopiere das aktuelle v rechts daneben.
- 6: Simuliere M für $(t)_2$ Schritte auf Band 2.
- 7: **Falls** M (innerhalb 6:) akzeptiert,
- 8: so *akzeptiere*.
- 9: Lösche Band 2.
- 10: Verschiebe Markierung $*$ auf Band 1 um ein Feld nach rechts.
- 11: **Falls** Markierung unter B steht, so
- 12: stelle die initiale Markierung (siehe 2:) wieder her,
- 13: erhöhe den Zähler t binär um eins.

Es gilt nun:

$$\begin{aligned}
 w \in L_{\text{Flip}(M)} &\iff \exists \text{ Zeitschranke } t \text{ und Zerlegung } w = uv, u \neq \varepsilon \text{ mit:} \\
 &\quad \text{Die Simulation von } M \text{ auf } \text{Flip}(u)v \text{ f\u00fcr } t \text{ Schritte ist erfolgreich} \\
 &\iff w \in L_N
 \end{aligned}$$

Aufgabe 5 (Rekursive Aufz\u00e4hlbarkeit und Nichtrekursivit\u00e4t von Sprachen)

[12 Punkte]

Zeigen Sie:

- (a) $L_{\text{Union}} := \{\langle M \rangle \langle M' \rangle \mid H_M \cup H_{M'} \neq \emptyset\}$ ist rekursiv aufz\u00e4hlbar.
- (b) $L_{2005} := \{\langle M \rangle \mid \forall x \in \{1\}^+ : |f_M(x)| \leq |x|^{2005}\}$ ist nicht rekursiv.

L\u00f6sung: Zu (a) Die Mehrband-TM N mit $L_N = L_{\text{Union}}$ arbeite auf Eingabe $x \in \{0, 1\}^*$ wie folgt:

- 1: **Falls** $x \neq \langle M \rangle \langle M' \rangle$ f\u00fcr normierte Turingmaschinen M, M' , so *verwerfe*.
- 2: **F\u00fcr** jede Zeitschranke $t = 1, 2, \dots$ (verwaltet auf Band 2) **tue**
- 3: Erzeuge t Eingaben $x_1 \# \dots \# x_t$ mittels Enum_{kan} (vgl. \u00dcbungsblatt 3, 1(c)) auf Band 3.
- 4: **F\u00fcr** jede solche Eingabe x_i **tue**
- 5: Erzeuge $\langle M \rangle x_i$ auf Band 4.
- 6: Simuliere die Universelle-TM U auf $\langle M \rangle x_i$ f\u00fcr t Schritte.
- 7: **Falls** U (innerhalb 6:) akzeptiert,
- 8: so *akzeptiere*.
- 9: L\u00f6sche Band 4.
- 10: Erzeuge $\langle M' \rangle x_i$ auf Band 4.
- 11: Simuliere die Universelle-TM U auf $\langle M' \rangle x_i$ f\u00fcr t Schritte.
- 12: **Falls** U (innerhalb 11:) akzeptiert,
- 13: so *akzeptiere*.
- 14: L\u00f6sche Band 4.
- 15: L\u00f6sche Band 3.

Es gilt nun:

$$\begin{aligned}
 x \in L_{\text{Union}} &\iff x = \langle M \rangle \langle M' \rangle \text{ f\u00fcr normierte TMn } M, M' \text{ und } \exists \text{ Zeitschranke } t \text{ und Input } x: \\
 &\quad \text{Die Simulation von } U \text{ auf } \langle M \rangle x \text{ oder } \langle M' \rangle x \text{ f\u00fcr } t \text{ Schritte ist erfolgreich} \\
 &\iff x \in L_N
 \end{aligned}$$

Zu (b) Wir argumentieren mittels Satz von Rice und betrachten dazu die folgende Eigenschaft:

$$\mathcal{F} := \{f : D \mid D \subseteq \{0, 1\}^*, f \text{ ist partiell rekursiv, } \forall x \in \{1\}^+ : |f(x)| \leq |x|^{2005}\}.$$

Diese Eigenschaft ist *nichttrivial*, denn z.B. liegt die Funktion $x \mapsto \varepsilon$ in \mathcal{F} , w\u00e4hrend z.B. die (total) rekursive Funktion $x \mapsto 1^{2006}$ nicht in \mathcal{F} liegt. Nach Konstruktion gilt nun

$$L_{\mathcal{F}} = \{\langle M \rangle \mid f_M \in \mathcal{F}\} = L_{2005}$$

und somit ist L_{2005} nicht rekursiv nach Satz von Rice (Satz 1.10.4(b)).

Aufgabe 6 (Unbeschränkte Reduktionen)

[12 Punkte]

Betrachten Sie die folgende Sprache:

$$L_{\text{Prim}} := \{ \langle M \rangle \mid M \text{ hält auf allen } y \in \{0, 1\}^* \text{ mit } |y| \in \text{Prim} \}$$

Zeigen Sie:

- (a) $H \leq L_{\text{Prim}}$
- (b) $\overline{H} \leq L_{\text{Prim}}$
- (c) Weder L_{Prim} noch $\overline{L_{\text{Prim}}}$ ist rekursiv aufzählbar.

Lösung: Zu (a) Wir konstruieren eine rekursive Funktion $f: \{0, 1\}^* \rightarrow \{0, 1\}^*$, $x \mapsto \langle M_x \rangle$ mit:

$$(*) \quad H_{M_x} = \begin{cases} \{0, 1\}^* & \text{falls } x \in H \\ \emptyset & \text{sonst} \end{cases}$$

Wegen (*) gilt dann für ein beliebiges $x \in \{0, 1\}^*$ wie gewünscht:

$$\begin{aligned} x \in H &\iff H_{M_x} = \{0, 1\}^* \\ &\iff \langle M_x \rangle = f(x) \in L_{\text{Prim}} \end{aligned}$$

Für $x \in \{0, 1\}^*$ arbeite M_x auf Eingabe $y \in \{0, 1\}^*$ wie folgt:

- 1: Überschreibe y mit x .
- 2: Starte die Universelle-TM U auf x .

Klar: f ist rekursiv und wegen $H = H_U$ gilt auch (*) nach Konstruktion von M_x .Zu (b) Wir konstruieren eine rekursive Funktion $g: \{0, 1\}^* \rightarrow \{0, 1\}^*$, $x \mapsto \langle M_x \rangle$ mit:

$$(**) \quad H_{M_x} = \begin{cases} \{0, 1\}^* & \text{falls } x \notin H \\ \{y \mid t_{\text{Sim}(U)}(x) > |t|\} & \text{sonst} \end{cases}$$

Man beachte, daß $\{y \mid t_{\text{Sim}(U)}(x) > |t|\}$ im Fall $x \in H$ eine endliche Menge ist. Wegen (**) gilt daher für beliebiges $x \in \{0, 1\}^*$ wie gewünscht:

$$\begin{aligned} x \notin H &\iff H_{M_x} = \{0, 1\}^* \\ &\iff \langle M_x \rangle = g(x) \in L_{\text{Prim}} \end{aligned}$$

Für $x \in \{0, 1\}^*$ arbeite M_x auf Eingabe $y \in \{0, 1\}^*$ wie folgt:

- 1: Schreibe x auf Band 2.
- 2: Simuliere die Universelle-TM U auf x für $|y|$ Schritte.
- 3: **Falls** U (innerhalb 2:) nicht hält,
- 4: so *halte*
- 5: **andernfalls**
- 6: gehe in eine *Endlosschleife*.

Klar: g ist rekursiv und wegen $H = H_U$ gilt auch (**) nach Konstruktion von M_x .Zu (c) Daß L_{Prim} nicht r.a. ist, folgt direkt aus (b) mittels Reduktionsmethode, denn \overline{H} ist nicht r.a. Wegen (a) gilt auch $\overline{H} \leq \overline{L_{\text{Prim}}}$ (nach Lemma 1.9.7(c)). Also folgt wie eben, daß $\overline{L_{\text{Prim}}}$ nicht r.a. ist.

Aufgabe 7 (Reduktionsmethode für **NP**-vollständige Sprachen)

[12 Punkte]

Ein Element der Sprache L_{Konflikt} repräsentiert einen Graphen $G = (V, E)$ mit Gewichten c_1, \dots, c_m für jeden Knoten von G sowie eine Gewichtsschranke k , so daß eine in G unabhängige Menge $I \subseteq V$, d.h.

$$\forall i, j \in I : i \neq j \implies (i, j) \notin E$$

mit Gesamtgewicht $m(I) := \sum_{i \in I} c_i \geq k$ existiert.

$$L_{\text{Konflikt}} := \{ \langle G \rangle \# \text{bin}(c_1) \# \dots \# \text{bin}(c_m) \# \text{bin}(k) \mid G \text{ ist ein Graph mit } m \text{ Knoten, der eine unabhängige Menge } I \text{ mit } m(I) \geq k \text{ besitzt} \}$$

Zeigen Sie:

- (a) $L_{\text{Konflikt}} \in \mathbf{NP}$
- (b) $L \leq_p L_{\text{Konflikt}}$ für eine geeignet gewählte Sprache $L \in \mathbf{NPC}$
- (c) $L_{\text{Konflikt}} \in \mathbf{NPC}$

Lösung: Zu (a) Wir beschreiben eine polynomialzeitbeschränkte NTM M mit $L_M = L_{\text{Konflikt}}$. Diese arbeite auf Eingabe $x \in \{0, 1, \#\}^*$ wie folgt:

- 1: Falls $x \neq \langle G \rangle \# \text{bin}(c_1) \# \dots \# \text{bin}(c_m) \# \text{bin}(k)$ für einen Graphen $G = (V, E)$ mit $|V| = m$, so *verwerfe*.
- 2: Schreibe 0^m auf Band 2 und starte darauf M_{Ratewort} .
▷ Der Ergebnis-Binärstring $b_1 \dots b_m$ repräsentiert $I = \{i \mid b_i = 1\} \subseteq V = \{1, \dots, m\}$.
- 3: Teste, ob I in G unabhängig (siehe oben) ist. Falls nein, so *verwerfe*.
- 4: Teste, ob $m(I) = \sum_{i \in I} c_i \geq k$ gilt. Falls nein, so *verwerfe*, andernfalls *akzeptiere*.

Man überzeugt sich leicht davon, daß $t_M(x) \in O(|x|^3)$ gilt.

Zu (b) Ruft man sich die Sprache L_{IS} (*Independent Set*) aus der Vorlesung in Erinnerung, so liegt die folgende Polynomialzeitreduktion auf der Hand: Es gilt $L_{\text{IS}} \leq_p L_{\text{Konflikt}}$ via $f \in \mathbf{FP}$, wobei

$$f(x) = \begin{cases} \langle (V, E) \rangle \# \underbrace{1 \dots 1}_{|V|-\text{mal}} \# \text{bin}(k) & \text{falls } x = \langle (V, E) \rangle \# \text{bin}(k) \text{ für einen Graphen } (V, E) \\ \varepsilon & \text{sonst.} \end{cases}$$

Denn für ein beliebiges $x \in \{0, 1, \#\}^*$ gilt wegen *Gewicht 1* für jeden „Knoten in $f(x)$ “:

$$\begin{aligned} x \in L_{\text{IS}} &\iff x = \langle G \rangle \# \text{bin}(k) \text{ und } G \text{ besitzt eine unabhängige Menge } I \text{ mit } |I| \geq k \\ &\iff f(x) = \langle G \rangle \# 1 \# \dots \# 1 \# \text{bin}(k) \in L_{\text{Konflikt}} \end{aligned}$$

Zu (c) Dies folgt aus (a) und (b) mittels Reduktionsmethode für **NP**-vollständige Sprachen (Lemma 2.6.1), denn L_{IS} ist **NP**-vollständig laut Vorlesung.