Prof. T. Nipkow
M. Eberl, J. Rädle, S. Wimmer
Technische Universität München
KEINE ABGABE

Einführung in die theoretische Informatik

Sommersemester 2019 – Übungsblatt Lösungsskizze 9

AUFGABE 9.1. (*Wichtige Begriffe*)
Überprüfen Sie, dass Sie die Folgenden Begriffe korrekt definieren können. Stufe A

- WHILE-Programme
- GOTO-Programme
- enscheidbar
- charakteristische Funktion
- spezielles Halteproblem
- allgemeines Halteproblem
- reduzierbar
- semi-entscheidbar
- rekursiv-aufzählbar
- Reduktion
- Satz von Rice

AUFGABE 9.2. (*Entscheidbarkeit vs. Berechenbarkeit*)
Ordnen Sie die folgenden Satzanfänge den Satzenden so zu, dass richtige Aussagen entstehen. Sei dazu $\mathsf{A}, \mathsf{B} \subseteq \{0,1\}^*$: Stufe B

(a) Die Funktion χ_{A} ist berechenbar,
(b) Die Funktion χ'_{A} ist berechenbar,
(c) A ist entscheidbar,
(d) B ist nicht entscheidbar,

(i) wenn $\mathsf{A} \leq \mathsf{B}$ gilt und B entscheidbar ist.
(ii) wenn A rekursiv aufzählbar ist.
(iii) wenn A entscheidbar ist.
(iv) wenn $\mathsf{A} \leq \mathsf{B}$ gilt und A nicht entscheidbar ist.

Lösungsskizze
(a) → (i)/(iii), (b) → (ii), (c) → (i)/(iii), (d) → (iv).

AUFGABE 9.3. (*Länge von Wörtern*)
Diskutieren Sie, wie viele Schritte eine Turing-Maschine mindestens machen muss, um zu entscheiden, ob für eine Eingabe w gilt: $|w| \geq 314$. Stufe B

Lösungsskizze
Falls $|w| \geq 314$, dann 314 Schritte. Nach 313 Schritten erreicht man die 314te Zelle des Bandes, nach 314 kann entschieden werden, ob in der 314ten Zelle etwas steht.
Falls $|w| < 314$, dann $|w| + 1$ Schritte, da dann nach $|w| + 1$ Schritten erkannt wird, dass das Band leer ist.

AUFGABE 9.4. (*Reduktionen*)
Entscheiden Sie, ob folgende Aussagen korrekt oder inkorrekt sind, und begründen Sie Ihre Antwort, indem Sie einen Beweis bzw. ein passendes Gegenbeispiel angeben. Sei $\Sigma = \{0,1\}$. Stufe B

(a) $\forall A \subseteq \Sigma^*.\ A \leq \Sigma^*$
(b) $\forall A, B \subseteq \Sigma^*.\ A \leq B \iff \overline{A} \leq \overline{B}$
(c) $\forall A \subseteq \Sigma^*.\ A \neq \emptyset \wedge A \neq \Sigma^* \implies A \leq \overline{A}$
(d) $\forall A, B, C \subseteq \Sigma^*.\ A \leq B \wedge B \leq C \implies A \leq C$

Lösungsskizze

(a) Falsch. Sei $A = \emptyset$. Damit $\overline{A} = \Sigma^*$. Dann muss für eine Reduktionsfunktion f gelten: $\forall x \in \overline{A}.\ f(x) \notin \Sigma^*$. Eine solche Funktion f existiert aber nicht.
(b) Wahr. Gelte $A \leq B$. Dann existiert ein totales und berechenbares f mit: $\forall x \in \Sigma^*. x \in A \Leftrightarrow f(x) \in B$. Somit gilt auch: $\forall x \in \Sigma^*. x \in \overline{A} \Leftrightarrow f(x) \in \overline{B}$. Daraus folgt dann $\overline{A} \leq \overline{B}$. Die Rückrichtung geht analog.
(c) Falsch. Sei $A = \mathsf{H}_0$. Nach Vorlesung wissen wir H_0 ist semi-entscheidbar und $\overline{\mathsf{H}_0}$ ist nicht semi-entscheidbar. Insbesondere sind damit die Mengen nicht trivial. Angenommen (c) gilt, dann haben wir $\mathsf{H}_0 \leq \overline{\mathsf{H}_0}$ und mit (b) auch $\overline{\mathsf{H}_0} \leq \mathsf{H}_0$. Dann sind aber beide Menge semi-entscheidbar und somit H_0 auch entscheidbar. Widerspruch!
(d) Wahr. Seien f und g Reduktionen von $A \leq B$ und $B \leq C$. Sei nun $h(x) = g(f(x))$. h ist total und berechenbar, da f und g total und berechenbar sind. Sei $x \in A$ beliebig. Dann gilt $f(x) \in B$ und $h(x) = g(f(x)) \in C$. Sei nun $x \notin A$ beliebig. Dann gilt $f(x) \notin B$ und damit $h(x) = g(f(x)) \notin C$. Somit ist h eine geeignete Reduktion.

AUFGABE 9.5. (*Entscheidbarkeit*)
Entscheiden Sie, ob die folgenden Behauptungen korrekt oder inkorrekt sind. Begründen Sie dann Ihre Antworten wie folgt: Wenn L entscheidbar bzw. semi-entscheidbar ist, beschreiben Sie einen Algorithmus, der die charakteristische Funktion χ_{L} bzw. die Funktion χ'_{L} berechnet. Wenn L unentscheidbar ist, leiten Sie einen Widerspruch zu einem Ergebnis der Vorlesung ab. Stufe B/C

(a) Wenn A und B entscheidbare Sprachen sind, dann ist $\mathsf{A} \cap \mathsf{B}$ entscheidbar.
(b) Wenn A und $\mathsf{A} \cup \mathsf{B}$ entscheidbar sind, dann ist B entscheidbar.
(c) Das Problem, ob $\mathsf{L}_H(\mathsf{M}) \neq \emptyset$ für eine gegebene Turingmaschine M gilt, ist semi-entscheidbar.
(d) Das Problem, ob $\mathsf{L}_H(\mathsf{M}) = \emptyset$ für eine gegebene Turingmaschine M gilt, ist semi-entscheidbar.

Lösungsskizze

(a) Korrekt.
Sei T_A DTM, die A entscheidet, T_B DTM, die B entscheidet.
DTM zu $A \cap B$: Gegeben x, berechne $T_A(x)$. Falls $T_A(x) = 0$, lehne x ab, ansonsten berechne $T_B(x)$. Gilt $T_B(x) = 0$, lehne x ab, sonst akzeptiere x. Da T_A, T_B die jeweiligen Mengen entscheiden, terminieren beide DTM immer, womit auch die DTM zu $A \cap B$ stets mit dem korrekten Ergebnis terminiert, womit $A \cap B$ entscheidbar ist.
(b) Inkorrekt.
Sei $B = H_0 \subseteq \{0,1\}^*$ das Halteproblem auf leere Eingabe und $A = \{0,1\}^*$. Dann sind A und $A \cup B$ entscheidbar, aber B nicht.
(c) Korrekt.
Eine NTM, kann einfach ein Wort aus $\mathsf{L}_H(M)$ raten, soweit es existiert die TM M ausführen und dann halten. Diese NTM kann von einer DTM simuliert werden und terminiert gdw. $\mathsf{L}(M)$ nicht leer ist. Andernfalls terminiert die Simulation nie.
Direkte DTM Konstruktion: Verwende einen wachsenden Zähler $i = 0, 1, \ldots$ und simuliere $\mathsf{M}[w]$ für i Schritte und für alle $w \in \Sigma^*$ mit $|w| \leq i$. Falls irgendeine $\mathsf{M}[w]$ hält, dann stoppe Simulation und gib "1" aus.
(d) Inkorrekt.
Kann nicht semi-entscheidbar sein, sonst wäre (zusammen mit der vorigen Teilaufgabe) $\mathsf{L}_H(\mathsf{M}) \neq \emptyset$ entscheidbar. Dann wäre auch das Halteproblem auf leerer Eingabe entscheidbar mit folgender Reduktion:
Für gegebene eine TM M, bilde ihre Codierung w auf die Codierung w' der TM:

"Falls $x \neq \varepsilon$, lehne x ab, sonst simuliere $M_w(\varepsilon)$ bis sie terminiert hat, und akzeptiere dann ε"

Somit akzeptiert $M_{w'}$ höchstens ε und das genau dann, wenn M_w auf ε terminiert.
Wenn jetzt entscheidbar wäre, ob $\mathsf{L}_H(M_{w'}) \neq \emptyset$, dann wäre entscheidbar, ob M_w auf ε terminiert.

AUFGABE 9.6. (*Reduktionen und Unentscheidbarkeit*) Stufe C
In dieser Aufgabe sollen Sie zeigen, dass das angegebene Problem unentscheidbar ist, indem Sie eine passende Reduktion von einem unentscheidbaren Problem angeben.

Beispiel:

Es ist unentscheidbar, ob eine Turing-Maschine für alle Eingaben 0 *ausgibt.* Formal heißt dies, dass wir zeigen wollen, dass $V_0 := \{\mathsf{enc}(\mathsf{M}) \mid \forall \mathsf{x} \in \Sigma^*.\ \varphi_{\mathsf{enc}(\mathsf{M})}(\mathsf{x}) = 0\}$ unentscheidbar ist. Wir tun dies, indem wir das Komplement des speziellen Halteproblems auf V_0 reduzieren, wobei das Halteproblem so definiert ist: $\mathsf{K} := \{w \mid \exists \mathsf{M}.\mathsf{M} = \mathsf{dec}(w) \wedge \mathsf{M}[w]\downarrow\}$. dec ist dabei eine Funktion, die ein Encoding einer TM auf die entsprechende TM abbildet. Gegeben ein Wort, das keine TM encodiert, ist dec undefiniert.

Reduktion von $\overline{\mathsf{K}}$*:* Wir konstruieren für ein gegebenes Encoding $\mathsf{enc}(\mathsf{M})$ eine Turing-Maschine M' mit Eingabe $\mathsf{x} \in \{0,1\}^*$ wie folgt: Wir simulieren die Turing-Maschine M auf $\mathsf{enc}(\mathsf{M})$ für $|\mathsf{x}|$ viele Schritte. Falls M nicht innerhalb von $|\mathsf{x}|$ vielen Schritten hält, dann geben wir 0 aus, ansonsten 1.

Die Reduktion ist berechenbar, weil wir das Encoding von Turing-Maschinen berechnen können und Turing-Maschinen anhand ihres Encodings für eine begrenzte Anzahl an Schritten simulieren können.

Die Reduktion ist korrekt, weil... Angenommen, $\mathsf{enc}(\mathsf{M}) \in \overline{\mathsf{K}}$. Dann hält die Simulation der Turing-Maschine M auf Eingabe $\mathsf{enc}(\mathsf{M})$ während der Ausführung der Turing-Maschine M' für keine Eingabe x. Das heißt, die Turing-Maschine M' gibt immer 0 aus.

Angenommen, $\mathsf{enc}(\mathsf{M}) \in \mathsf{K}$. Dann gibt es eine natürliche Zahl t, so dass die Simulation von M in t Schritten auf $\mathsf{enc}(\mathsf{M})$ hält. Dann gibt die Turing-Maschine M' den Wert 1 für alle Eingaben der Länge größer oder gleich t aus.

Damit ist die folgende Funktion eine Reduktion von $\overline{\mathsf{K}}$ auf V_o: Sei $y \in V_0$.

$$i \mapsto \begin{cases} \mathsf{enc}(\mathsf{M}') & \text{falls } i = \mathsf{enc}(\mathsf{M}) \text{ für eine Turing-Maschine } \mathsf{M} \\ y & \text{ansonsten} \end{cases}$$

Der Fall falls i kein Encoding einer TM ist, ist wichtig, damit die Funktion total ist. Da K nur TMs enthält, sind alle Worte, die keine TM encodieren in $\overline{\mathsf{K}}$, und müssen folglich auf ein Element von V_0 abgebildet werden.

Zeigen Sie: Es ist unentscheidbar zu prüfen, ob für zwei als Eingabe gegebene Turing-Maschinen die eine auf allen Eingaben genau dann hält, wenn die andere nicht hält.
Zeigen Sie auch, dass dieses Problem nicht semi-entscheidbar ist.

Lösungsskizze
Wir bezeichnen die Menge des Entscheidungsproblems: $\mathsf{H}_{\mathsf{NEQ}}$
Das Halteproblem auf dem leeren Band ist die folgende Menge: $\mathsf{H}_0 := \{w \mid \exists \mathsf{M}.\mathsf{enc}(\mathsf{M}) = w \wedge \mathsf{M}[\varepsilon]\downarrow\}$. (Wir verwenden für die Definition die Encoding-Funktion enc, da dadurch der Randfall eines Wortes, das keine TM encodiert, offensichtlicher wird, als wenn man M_w verwendet.)

Reduktion von H_0*:* Sei $\mathsf{M}_\bot$ die Turingmaschine, die auf keiner Eingabe hält. Sei nun M eine beliebige TM. Wir konstruieren nun folgende Maschine M': Ignoriere Eingabe w und lösche das Band. Führe dann $\mathsf{M}[\varepsilon]$ aus.

Die Reduktion ist berechenbar, weil wir das Encoding von Turing-Maschinen decodieren und eine Turing-Maschine dann simulieren können. Außerdem können wir alle Zeichen auf dem Band, die nicht dem Leerzeichen entsprechen, überschreiben.

Die Reduktion ist korrekt: Angenommen, $\mathsf{enc}(\mathsf{M}) \in \mathsf{H}_0$. Dann hält M' auf alle Eingaben und da $\mathsf{M}_\bot$ auf alle Eingaben divergiert, ist $\mathsf{enc}(\mathsf{M}')\#\mathsf{enc}(\mathsf{M}_\bot) \in \mathsf{H}_{\mathsf{NEQ}}$.

Umgekehrt: Angenommen $\mathsf{enc}(\mathsf{M}) \notin \mathsf{H}_0$. Dann divergiert die Turing-Maschine für alle Eingaben, genauso auch die Turing-Maschine $\mathsf{M}_\bot$, also ist $\mathsf{enc}(\mathsf{M}')\#\mathsf{enc}(\mathsf{M}_\bot) \notin \mathsf{H}_{\mathsf{NEQ}}$.

Bei Eingabe eines Wortes i, das keine TM encodiert, gilt $i \notin \mathsf{H}_0$, und folglich müssen wir ein beliebiges $y \notin \mathsf{H}_{\mathsf{NEQ}}$ zurück geben. So ein y existiert, zum Beispiel ein Wort, das keine TM encodiert.

Damit ist die folgende Funktion eine Reduktion von H_0 auf $\mathsf{H}_{\mathsf{NEQ}}$. Sei dafür $y \in \overline{\mathsf{H}_{\mathsf{NEQ}}}$ beliebig:

$$f(w) = \begin{cases} \mathsf{enc}(\mathsf{M}')\#\mathsf{enc}(\mathsf{M}_\bot) & \text{falls } w = \mathsf{enc}(\mathsf{M}) \text{ für eine Turing-Maschine } \mathsf{M} \\ y & \text{ansonsten} \end{cases}$$

Bis zu diesem Punkt haben wir nur gezeigt, dass $\mathsf{H}_{\mathsf{NEQ}}$ nicht entscheidbar (= unentscheidbar) ist. Mit dieser Reduktion von H_0 können wir aber nichts darüber aussagen, ob das Problem semi-entscheidbar ist oder nicht. Die folgende Reduktion zeigt nun, dass das Problem *nicht* semi-entscheidbar ist, weil $\overline{\mathsf{H}_0}$ nicht semi-entscheidbar ist. (Wir hätten uns die erste Reduktion auch sparen können, da jedes nicht semi-entscheidbare Problem auch nicht entscheidbar ist.)

Reduktion von $\overline{\mathsf{H}_0}$*:* Sei $\mathsf{M}_\top$ die Turingmaschine, die auf allen Eingaben hält. Sei nun M eine beliebige TM. Wir konstruieren nun folgende Maschine M': Ignoriere Eingabe w und lösche das Band. Führe dann $\mathsf{M}[\varepsilon]$ aus.

Die Reduktion ist berechenbar, weil wir das Encoding von Turing-Maschinen decodieren und eine Turing-Maschine dann simulieren. können. Außerdem können wir alle Zeichen auf dem Band, die nicht dem Leerzeichen entsprechen, überschreiben.

Die Reduktion ist korrekt: Angenommen, $\mathsf{enc}(\mathsf{M}) \in \overline{\mathsf{H}_0}$. Dann divergiert M' auf alle Eingaben und da $\mathsf{M}_\top$ auf alle Eingaben hält, ist $\mathsf{enc}(\mathsf{M}')\#\mathsf{enc}(\mathsf{M}_\top) \in \mathsf{H}_{\mathsf{NEQ}}$.

Umgekehrt: Angenommen $\mathsf{enc}(\mathsf{M}) \notin \overline{\mathsf{H}_0}$. Dann hält die Turing-Maschine für alle Eingaben, genauso auch die Turing-Maschine $\mathsf{M}_\top$, also ist $\mathsf{enc}(\mathsf{M}')\#\mathsf{enc}(\mathsf{M}_\top) \notin \mathsf{H}_{\mathsf{NEQ}}$.

Bei Eingabe eines Wortes i, das keine TM encodiert, gilt $i \notin \mathsf{H}_0$, also $i \in \overline{\mathsf{H}_0}$, und folglich müssen wir ein beliebiges $y \in \mathsf{H}_{\mathsf{NEQ}}$ zurück geben. So ein y existiert, z.B. $\mathsf{enc}(\mathsf{M}_\top)\#\mathsf{enc}(\mathsf{M}_\bot)$.

Damit ist die folgende Funktion eine Reduktion von $\overline{\mathsf{H}_0}$ auf $\mathsf{H}_{\mathsf{NEQ}}$. Sei dafür $y \in \mathsf{H}_{\mathsf{NEQ}}$ beliebig:

$$f(w) = \begin{cases} \mathsf{enc}(\mathsf{M}')\#\mathsf{enc}(\mathsf{M}_\top) & \text{falls } w = \mathsf{enc}(\mathsf{M}) \text{ für eine Turing-Maschine } \mathsf{M} \\ y & \text{ansonsten} \end{cases}$$

AUFGABE 9.7. (*Collatz-Vermutung*) Stufe D
Zu einem Startwert $a_0 \in \mathbb{N}$ definieren wir eine Folge $(a_n)_{n\in\mathbb{N}_0}$ wie folgt:

$$a_{n+1} := \begin{cases} a_n/2 & a_n \text{ gerade} \\ 3a_n + 1 & a_n \text{ ungerade} \end{cases}$$

Die seit 1937 unbewiesene Collatz-Vermutung besagt:

Für alle positiven Startwerte $a_0 \in \mathbb{N}$ gibt es einen Index $i \in \mathbb{N}_0$, so dass $a_i = 1$.

Nehmen Sie an, es gibt ein Programm N, welches als Eingabe ein WHILE-Programm P mit genau einer Eingabevariable nimmt und zu jedem solchen P angibt, ob P die Nullfunktion berechnet. *Zeigen Sie, dass Sie dann die Collatz-Vermutung beweisen oder widerlegen können.*
Hinweise:
- Geben Sie auch das WHILE-Programm P, das sie für Ihren Beweis verwendet haben, an.
- Sie dürfen jede Syntax, die in den Folien für WHILE-Programme eingeführt worden ist, verwenden.

Lösungsskizze
Das folgende Programm P terminiert, falls es für einen Startwert a_0 (in Variable x) in der Collatz-Folge das Folgeglied $a_i = 1$ findet. Beachte, dass für die Eingabe $x = 0$ die Collatz-Folge nicht definiert ist und das Programm sofort mit 0 terminiert (Achtung: modifizierte Differenz). Zur besseren Lesbarkeit verwenden wir $y = x_0$, $x = x_1$, $z = x_2$, $c = x_3$ als Abkürzungen. Berechnet nun das Programm die Nullfunktion, so ist die Collatz-Vermutung korrekt, da für alle Startwerte das Programm hält und 0 zurückgibt. Berechnet das Programm nicht die Nullfunktion, so gibt es mindestens einen Startwert, für den das Programm nicht terminiert.
Programm P:

```
1  x := x - 1;
2  WHILE x ≠ 0 DO
3    x := x + 1;
4    c := 2;
5    z := x MOD c;
6    IF z = 0 DO
7      x := x DIV c
8    ELSE
9      z := x + x;
10     x := x + z;
11     x := x + 1
12   END;
13   x := x - 1
14 END;
15 y := 0
```

AUFGABE 9.8. (*Semi-Entscheidbarkeit*) Stufe D
Sei $A \subseteq \Sigma^*$. Zeigen Sie die folgende Behauptung:

A ist semi-entscheidbar gdw. A ist Wertebereich einer berechenbaren Funktion

Lösungsskizze
Wenn A semi-entscheidbar: Dann ist A rekursiv aufzählbar und somit A Wertebereich einer berechenbaren Funktion (genau die Funktion aus Definition 5.44, die A aufzählt)

Wenn A Wertebereich einer berechenbaren Funktion f: Sei T eine TM zu f. Wir zeigen die Semi-Entscheidbarkeit von A indem wir eine TM bauen, die für eine Eingabe w hält gdw. w im Wertebereich von f ist: Simuliere T für ansteigende Grenze N auf allen Eingaben der Länge $\leq N$ für N Schritte; falls die Simulation für irgendeine Eingabe terminiert und w zurückgibt, halte an.