Prof. Javier Esparza
Philipp Czerner, Kevin Kappelmann, Valentin Krasotin

Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2024 – Übungsblatt 10

- Das Übungsblatt ist in zwei Teile gegliedert: den Vorbereitungsteil, den Sie vor der Übung selbstständig bearbeiten sollen, und den Übungs-/Nachbereitungsteil, der Aufgaben enthält, die in der Übung besprochen werden und von Ihnen anschließend zur Nachbereitung verwendet werden können.

Vorbereitung (vor der Übung selbständig zu bearbeiten)

Vorbereitungsaufgabe Ü10.1. (*Wichtige Begriffe & Kahoot*)

Überprüfen Sie, dass Sie die folgenden Begriffe oder Notationen korrekt definieren können.

- WHILE-Programm
- GOTO-Programm
- Konvertierung: WHILE → TM → GOTO → WHILE
- entscheidbar/unentscheidbar
- charakteristische Funktion einer Menge A: χ_A
- spezielles Halteproblem
- M_w, $M_w[x]$, $M_w[x]\downarrow$
- allgemeines und spezielles Halteproblem, Halteproblem auf leerem Band
- Reduktion (Beispiel: $A \leq B$, sprich: "A ist reduzierbar auf B")

Vorbereitungsaufgabe Ü10.2. (*Zweierpotenz: TM, WHILE, GOTO*)

Sei $f : \{0,1\}^* \to \{0,1\}$ die Funktion, die angibt, ob ein Wort die Binärdarstellung einer Zweierpotenz ist, wobei führende Nullen erlaubt sind $(0100)_2 = (100)_2 = 4$:

$$f(w) = \begin{cases} 1, & \text{wenn } (w)_2 = 2^n \text{ für ein } n \in \mathbb{N} \\ 0, & \text{sonst} \end{cases}$$

(a) Geben Sie graphisch eine TM an, die f berechnet.

Erinnerung: Damit eine TM eine Funktion berechnet, muss das Band nach der Berechnung nur noch die Ausgabe enthalten und der Kopf der TM muss auf das erste Zeichen der Ausgabe zeigen.

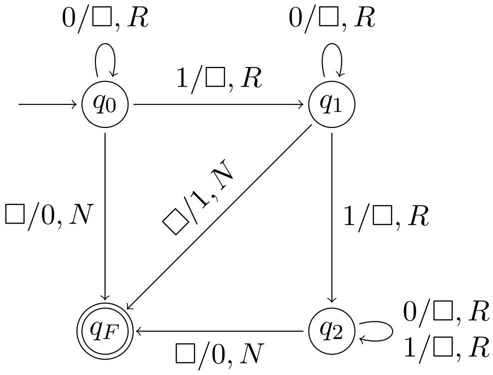
(b) Geben Sie ein WHILE-Programm an, das berechnet, ob $x_1 \in \mathbb{N}$ (die Eingabe) eine Zweierpotenz ist. Die Eingabe liegt, anders als in Aufgabe (a), in Dezimaldarstellung vor. Sie dürfen die Funktionen DIV und MOD benutzen. Geben Sie bei Eingabe einer Zweierpotenz 1 und 0 sonst zurück.

Erinnerung: Die Syntax von WHILE-Programmen findet man auf den Folien. Es gibt bei WHILE-Programmen (und GOTO-Programmen) kein Band und keinen direkten Zugriff auf die Binärrepräsentation der Eingabe. Nach dem Terminieren des Programms muss die Ausgabe in x_0 stehen.

(c) Geben Sie ein GOTO-Programm an, das berechnet, ob x_1 (die Eingabe) eine Zweierpotenz ist. Sie dürfen die Funktionen DIV und MOD benutzen.

Lösungsskizze.

(a) Eine mögliche TM:



(b) Lösungsvorschlag:

while $x_1 \neq 0$ **do**
    $x_2 := x_1$ MOD 2
    $x_1 := x_1$ DIV 2
    if $x_1 = 0$ **then**
        $x_0 := 1$
    else
        if $x_2 = 0$ **then**
            $x_0 := x_0$ [1]
        else
            $x_0 := 0$
            $x_1 := 0$
        end
    end
end

(c)

```
START: IF x1 = 0 GOTO STOP;
x2 := x1 MOD 2;
x1 := x1 DIV 2;
IF x1 = 0 GOTO SUCCESS;
IF x2 = 0 GOTO START;
x0 := 0;
x1 := 0;
GOTO START;
SUCCESS: x0 := 1;
STOP: HALT
```

[1] Beachten Sie, dass wir nur if-Anweisungen der Form $x_k = 0$ definiert haben und nicht der Form $x_k \neq 0$. Wenn $x_2 = 0$ gilt, wollen wir nichts tun, aber müssen eine Anweisung angeben. Wir haben $x_0 = x_0$ gewählt, die offensichtlich keinen Effekt hat.

Vorbereitungsaufgabe Ü10.3. (*Quiz*)

Ordnen Sie die folgenden Satzanfänge allen Satzenden zu, sodass richtige Aussagen entstehen. Sei dazu $A, B \subseteq \{0,1\}^*$:

(a) Die Funktion χ_A ist berechenbar,

(b) A ist entscheidbar,

(c) B ist nicht entscheidbar,

(i) wenn $A \leq B$ gilt und B entscheidbar ist.

(ii) wenn A entscheidbar ist.

(iii) wenn $A \leq B$ gilt und A nicht entscheidbar ist.

Lösungsskizze. (a) $\to$ (i)/(ii), (b) $\to$ (i)/(ii), (c) $\to$ (iii).

Vorbereitungsaufgabe Ü10.4. (*Reductio ad absurdum*)

Sei $A := \{w \in \Sigma^* \mid \exists i \in \mathbb{N}_0.\ |w| = 5i + 3\}$ mit $\Sigma = \{a, b\}$.

Entscheiden und erklären Sie, ob die folgenden Funktionen eine korrekte Reduktionen gemäß Vorlesungsdefinition sind.

(a) Behauptung: Sei L_1 kontextfrei mit Grammatik G und $\emptyset \neq L_2 \neq \Sigma^*$ regulär mit Grammatik G' und $L_1, L_2 \subseteq \Sigma^*$. Dann $L_1 \leq L_2$.

Reduktion: Wähle $u \in L_2$ und $v \in \Sigma^* \setminus L_2$. Definiere $f : \Sigma^* \to \Sigma^*$ mit

$$f(w) := \begin{cases} u, & \text{falls } w \in L_1 \\ v, & \text{sonst} \end{cases} \quad .$$

(b) Behauptung: $A \leq A \cup \{x\}$.

Reduktion: $f(w) = w$.

(c) Behauptung: $\mathcal{H}_0 \leq A$

Reduktion: Definiere $f : \mathcal{H}_0 \to A$ mit $f(w) := aaa$.

(d) Behauptung: $\mathcal{H}_0 \leq A$

Reduktion:

$$f(w) = \begin{cases} aaa & \text{falls } w \in \mathcal{H}_0 \\ b & \text{sonst} \end{cases}$$

(e) Behauptung: $A \leq \mathcal{H}_0$

Reduktion: f bildet jedes Element $x \in \Sigma^*$ auf die Kodierung w einer TM ab, die wie folgt definiert ist: Die TM M_w löscht die Eingabe und schreibt x aufs Band, bestimmt dann die Länge von x, zieht 3 ab und prüft anschließend, ob das Ergebnis durch 5 teilbar ist. Dementsprechend gibt die Maschine „Ja“(1) oder „Nein“(0) aus.

(f) Behauptung: $\overline{\mathcal{H}_0} \leq \mathcal{H}_0$

Reduktion: f bildet jedes $w \in \{0,1\}^*$ auf die Kodierung $f(w)$ einer TM $M_{f(w)}$ ab, die $M_w[\epsilon]$ simuliert. Falls $M_w[\epsilon]$ hält, geht $M_{f(w)}$ in eine Endlosschleife. Falls $M_w[\epsilon]$ nicht hält, hält $M_{f(w)}$.

Lösungsskizze.

(a) Richtig: die Funktion ist total, korrekt und berechenbar, da das Wortproblem für kontextfreie Grammatiken berechenbar ist (z.B. mit CYK).

(b) Falsch: Falls $x \notin A$ gilt $f(x) \in A \cup \{x\}$.

(c) Falsch: f ist undefiniert auf $\{0,1\}^* \setminus \mathcal{H}_0 \neq \emptyset$ und somit nicht total.

(d) Falsch: f ist unberechenbar, da $\mathcal{H}_0$ unentscheidbar ist und somit $\chi_{\mathcal{H}_0}$ unberechenbar ist.

(e) Falsch: f bildet auf Kodierungen von Turingmaschinen ab, die immer terminieren. Da $a \notin A$, aber $f(a) \in \mathcal{H}_0$, erfüllt die Funktion f nicht die Definition einer Reduktion.

(f) Falsch: f ist nicht wohldefiniert. Wenn $M_{f(w)}$ die Berechnung von $M_w[\epsilon]$ simuliert und $M_w[\epsilon]$ nicht hält, dann hält definitiv $M_{f(w)}$ auch nicht.

Vorbereitungsaufgabe Ü10.5. (H_{NEQ})

Sei Σ ein beliebiges Alphabet. Zeigen Sie: Es ist unentscheidbar zu prüfen, ob für zwei als Eingabe gegebene Turingmaschinen die eine auf allen Eingaben genau dann hält, wenn die andere nicht hält. Formaler: Zeigen Sie, die Menge

$$\mathcal{H}_{NEQ} \coloneqq \{w_1\#w_2 \mid w_1, w_2 \in \{0,1\}^* \text{ und } \forall x \in \Sigma^*.\ M_{w_1}[x]\downarrow \iff \neg M_{w_2}[x]\downarrow\}$$

ist unentscheidbar.

Lösungsskizze. Für diese Aufgabe gibt es eine Videolösung: Link

Wir reduzieren das Halteproblem auf leerem Band $\mathcal{H}_0$ auf $\mathcal{H}_{NEQ}$. Dabei gilt zu beachten, dass die Reduktion total, berechenbar und korrekt sein muss.

Reduktion von $\mathcal{H}_0$: Sei $w_\perp$ die Kodierung einer Turingmaschine, die auf keiner Eingabe hält. Sei $w \in \{0,1\}^*$ beliebig. Wir berechnen zunächst die Kodierung w' einer Turingmaschine, die bei jeder Eingabe das Band löscht und dann $M_w[\epsilon]$ ausführt. Anschließend geben wir $w'\#w_\perp$ zurück.

Die Reduktion ist total: Für jede Eingabe w wird die Ausgabe $w'\#w_\perp$ erzeugt.

Die Reduktion ist berechenbar: Die En- und Dekodierungsfunktionen für Turingmaschinen sind berechenbar. Außerdem kann eine TM alle Zeichen auf dem Band, die nicht dem Leerzeichen entsprechen, durch geeignete Übergänge anfangs überschreiben.

Die Reduktion ist korrekt:

$$\begin{aligned}
w \in \mathcal{H}_0 &\iff M_w[\epsilon]\downarrow && \text{(Def. } \mathcal{H}_0\text{)}\\
&\iff \forall x \in \Sigma^*.\ M_{w'}[x]\downarrow && (M_{w'} \text{ führt immer } M_w[\epsilon] \text{ aus})\\
&\iff \forall x \in \Sigma^*.(M_{w'}[x]\downarrow \text{ und } \neg M_{w_\perp}[x]\downarrow) && (M_{w_\perp} \text{ hält nie})\\
&\iff w'\#w_\perp \in \mathcal{H}_{NEQ} && \text{(Def. } \mathcal{H}_{NEQ}\text{)}
\end{aligned}$$

□

Übung und Nachbereitung

Übungsaufgabe Ü10.6. (*Entscheidbarkeit*)

Überprüfen Sie, ob die folgenden Behauptungen korrekt oder inkorrekt sind. Begründen

Sie dann Ihre Antworten wie folgt: Wenn L entscheidbar ist, beschreiben Sie einen Algorithmus, der die charakteristische Funktion χ_L berechnet. Wenn L unentscheidbar ist, leiten Sie einen Widerspruch zu einem Ergebnis der Vorlesung ab.

(a) Wenn A und B entscheidbare Sprachen sind, dann ist $A \cap B$ entscheidbar.

(b) Wenn A und $A \cup B$ entscheidbar sind, dann ist B entscheidbar.

Lösungsskizze. Für diese Aufgabe gibt es eine Videolösung: Link

(a) Korrekt. Wenn A und B entscheidbar sind, dann sind die charakteristischen Funktionen χ_A und χ_B berechenbar. Wir definieren für alle Wörter w

$$\chi_{A\cap B}(w) = \begin{cases} 1, & \text{wenn } \chi_A(w) = 1 \land \chi_B(w) = 1, \\ 0 & \text{sonst.} \end{cases}$$

Diese Funktion ist berechenbar (da χ_A und χ_B berechenbar sind) und eine charakteristische Funktion von $A \cap B$, da

$$\chi_{A\cap B}(w) = 1 \Leftrightarrow \chi_A(w) = 1 \land \chi_B(w) = 1 \Leftrightarrow w \in A \land w \in B \Leftrightarrow w \in A \cap B$$

Folglich ist die Menge $A \cap B$ entscheidbar.

Alternativ mit Turingmaschinen:
Sei T_A eine DTM, die A entscheidet, T_B eine DTM, die B entscheidet. Wir konstruieren eine DTM zu $A \cap B$: Gegeben x, berechne $T_A[x]$. Falls $T_A[x]$ mit 0 auf dem Band hält, lehne x ab, ansonsten berechne $T_B[x]$. Hält $T_B[x]$ mit 0 auf dem Band, lehne x ab, sonst akzeptiere x. Da T_A, T_B die jeweiligen Mengen entscheiden, terminieren beide DTM, womit auch die DTM zu $A \cap B$ stets mit dem korrekten Ergebnis terminiert. Damit ist $A \cap B$ entscheidbar.

(b) Inkorrekt. Sei $B \coloneqq K = \{w \in \{0,1\}^* \mid M_w[w]\downarrow\} \subsetneq \{0,1\}^*$ das spezielle Halteproblem und $A = \{0,1\}^*$. Dann sind A und $A \cup B = \{0,1\}^*$ entscheidbar, aber B nicht.

Übungsaufgabe Ü10.7. (*Reduktion*)

Erinnerung: $\mathcal{H} \coloneqq \{w\#x : w, x \in \{0,1\}^*, M_w[x]\downarrow\}$ (*allgemeines Halteproblem*) und $\mathcal{H}_0 \coloneqq \{w \in \{0,1\}^* \mid M_w[\epsilon]\downarrow\}$ (*Halteproblem auf leerem Band*)

(a) Sei $\mathcal{H}_{uvu} \coloneqq \{w\#u\#v : w, u, v \in \{0,1\}^*, M_w[uvu]\downarrow\}$.
Zeigen Sie: $\mathcal{H}_{uvu}$ ist unentscheidbar, da $\mathcal{H} \leq \mathcal{H}_{uvu}$.

(b) Sei $\mathcal{H}_{\Sigma^*} \coloneqq \{w \in \{0,1\}^* \mid \forall x \in \Sigma^* : \ M_w[x]\downarrow\}$.
Zeigen Sie: $\mathcal{H}_{\Sigma^*}$ ist unentscheidbar, da $\mathcal{H}_0 \leq \mathcal{H}_{\Sigma^*}$.

Lösungsskizze.

(a) Wir reduzieren das allgemeine Halteproblem $\mathcal{H}$ auf $\mathcal{H}_{uvu}$. Dabei gilt zu beachten, dass die Reduktion total, berechenbar und korrekt sein muss.

Reduktion von $\mathcal{H}$:
Für $w\#x$ geben wir $w\#\#x$ zurück und für andere Eingaben ein beliebiges $v \notin \mathcal{H}$, z.B. ϵ.

Die Reduktion ist total:
Für jede Eingabe wird eine Ausgabe erzeugt (insbesondere, wenn die Eingabe nicht

von der Form $w\#x$ ist!).

Die Reduktion ist berechenbar:
Es gibt eine TM, die nach dem Trennzeichen (#) ein weiteres Trennzeichen einfügt und alle anderen Zeichen danach um eins nach rechts verschiebt.

Die Reduktion ist korrekt:

$$\begin{aligned} w\#x \in \mathcal{H} &\iff M_w[x]\downarrow && \text{(Def. } \mathcal{H}\text{)} \\ &\iff M_w[\epsilon x \epsilon]\downarrow \\ &\iff w\#\#x \in \mathcal{H}_{uvu} && \text{(Def. } \mathcal{H}_{uvu}\text{)} \end{aligned}$$

□

(b) Wir reduzieren das allgemeine Halteproblem auf leerem Band $\mathcal{H}_0$ auf $\mathcal{H}_{\Sigma^*}$. Dabei gilt zu beachten, dass die Reduktion total, berechenbar und korrekt sein muss.

Reduktion von $\mathcal{H}$:
Wir berechnen die Kodierung w' einer TM, die zuerst die Eingabe löscht und dann M_w simuliert. Wir geben w' zurück.

Die Reduktion ist total:
Für jede Eingabe w wird die Ausgabe w' erzeugt.

Die Reduktion ist berechenbar:
Die En- und Dekodierungsfunktionen für Turingmaschinen sind berechenbar. Außerdem kann eine TM alle Zeichen auf dem Band, die nicht dem Leerzeichen entsprechen, durch geeignete Übergänge anfangs überschreiben.

Die Reduktion ist korrekt:

$$\begin{aligned} w \in \mathcal{H}_0 &\iff M_w[\epsilon]\downarrow && \text{(Def. } \mathcal{H}_0\text{)} \\ &\iff \forall x \in \Sigma^*.\ M_{w'}[x]\downarrow && (M_{w'} \text{ führt stets } M_w \text{ auf leerem Band aus}) \\ &\iff w' \in \mathcal{H}_{\Sigma^*} && \text{(Def. } \mathcal{H}_{\Sigma^*}\text{)} \end{aligned}$$

□

Übungsaufgabe Ü10.8. (*Allgemeine Reduktion*)

Entscheiden Sie, ob folgende Aussagen korrekt oder inkorrekt sind, und begründen Sie Ihre Antwort, indem Sie einen Beweis bzw. ein passendes Gegenbeispiel angeben. Sei $\Sigma := \{0, 1\}$.

(a) $\forall A \subseteq \Sigma^* : A \leq \Sigma^*$

(b) $\forall A, B \subseteq \Sigma^* : A \leq B \iff \overline{A} \leq \overline{B}$

(c) $\forall A, B, C \subseteq \Sigma^* : A \leq B \land B \leq C \implies A \leq C$

Lösungsskizze.

(a) Falsch. Sei $A = \emptyset$. Dann muss für eine Reduktionsfunktion f gelten: $\forall x \notin A.\ f(x) \notin \Sigma^*$. Eine solche Funktion f existiert aber nicht, da stets $f(x) \in \Sigma^*$ gilt.

(b) Wahr. Es gelte $A \leq B$. Dann existiert ein totales und berechenbares f mit: $x \in A$ genau dann wenn $f(x) \in B$ für alle $x \in \Sigma^*$. Somit gilt auch: $x \in \overline{A}$ genau dann wenn $f(x) \in \overline{B}$ für alle $x \in \Sigma^*$. Daraus folgt dann $\overline{A} \leq \overline{B}$. Die Rückrichtung folgt analog.

(c) 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.

Übungsaufgabe Ü10.9. (*Collatz-Vermutung*)

Zu einem Startwert $a_0 \in \mathbb{N}_+$ definieren wir eine Folge $(a_n)_{n\in\mathbb{N}}$ 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 Startwerte $a_0 \in \mathbb{N}_+$, gibt es einen Index $i \in \mathbb{N}$, sodass $a_i = 1$.

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:

```
x := x - 1;
WHILE x ≠ 0 DO
  x := x + 1;
  z := x MOD 2;
  IF z = 0 DO
    x := x DIV 2
  ELSE
    z := x + x;
    x := x + z;
    x := x + 1
  END;
  x := x - 1
END;
y := 0
```