Prof. Javier Esparza
Philipp Czerner, Kevin Kappelmann, Valentin Krasotin
Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2024 – Übungsblatt 11

- 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 Ü11.1. (*Wichtige Begriffe*)

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

- semi-entscheidbar
- rekursiv aufzählbar
- Satz von Rice
- Postsches Korrespondenzproblem (PCP)

Übung und Nachbereitung

Übungsaufgabe Ü11.2. (*All will be 1*)

Sei $\Sigma := \{a\}$ ein *unäres* Alphabet.

(a) Welche der folgenden PCP Instanzen ist erfüllbar?

(1) $\binom{aaaa}{a}, \binom{aa}{aaa}$

(2) $\binom{aa}{a}, \binom{aaa}{aa}$

(b) Zeigen Sie, dass a-PCP (also PCP auf dem Alphabat $\Sigma = \{a\}$) entscheidbar ist.

Lösungsskizze.

(a) (1) ist erfüllbar, mit Lösung $\binom{aaaa}{a}, \binom{aa}{aaa}, \binom{aa}{aaa}, \binom{aa}{aaa}$.

(2) ist nicht erfüllbar, da die obere Reihe stets länger ist.

(b) Sei $(x_1, y_1), ..., (x_n, y_n)$ die Eingabe, mit $x_1, ..., x_n, y_1, ..., y_n \in \{a\}^*$. Wir definieren $d_i := |x_i| - |y_i|$ für $i \in \{1, ..., n\}$. Wenn $d_i = 0$ für ein i, ist die Instanz offensichtlich erfüllbar, wir nehmen also an, dass dies nicht der Fall ist. Nun zeigen wir, dass die Instanz genau dann erfüllbar ist, wenn es i, j gibt, sodass $d_i > 0 > d_j$.

„$\Leftarrow$“: Wir wählen entsprechende i, j. Dann ist $(i)^{-d_j}(j)^{d_i}$ eine Lösung, wobei $(i)^{-d_j}$ die Folge bezeichnet, die aus $-d_j$ Kopien von i bezeichnet. Wir begründen dies kurz: sei $x := x_i^{-d_j} x_j^{d_i}$ und $y := y_i^{-d_j} y_j^{d_i}$. Dann gilt

$$\begin{aligned}
&|x| - |y| \\
&= (-d_j)|x_i| + d_i|x_j| - (-d_j)|y_i| - d_i|y_j| \\
&= -d_j(|x_i| - |y_i|) + d_i(|x_j| - |y_j|) \\
&= -d_j d_i + d_i d_j = 0
\end{aligned}$$

und somit $x = y$.

„$\Rightarrow$“: Der Beweis ist über Kontraposition. Wir nehmen oBdA an, dass $d_i > 0$ für alle $i \in \{1, ..., n\}$ (falls stattdessen $d_i < 0$ für alle i, können wir die beiden Komponenten tauschen). Dann ist aber die erste Komponente stets größer als die zweite, und somit gibt es keine Lösung.

Übungsaufgabe Ü11.3. (*Chomsky Lebt*)

In der Vorlesung haben wir, beispielsweise, folgende Probleme für kontextfreie Grammatiken G_1, G_2 über einem Alphabet Σ kennengelernt:

$\langle 1 \rangle$ Ist $L(G_1) \cap L(G_2) = \emptyset$?

$\langle 2 \rangle$ Ist $|L(G_1) \cap L(G_2)| < \infty$?

$\langle 3 \rangle$ Ist $L(G_1) \subseteq L(G_2)$?

$\langle 4 \rangle$ Ist $L(G_1) = L(G_2)$?

Von $\langle 1 \rangle$ wurde in der Vorlesung gezeigt, dass es unentscheidbar ist. Zeigen Sie nun:

(a) $\langle 3 \rangle \le \langle 4 \rangle$

(b) $\langle 1 \rangle \le \langle 2 \rangle$

Zur Vereinfachung dürfen Sie $\Sigma = \{a, b\}$ für das zu reduzierende Problem annehmen.

Lösungsskizze. In beiden Teilaufgaben bildet unsere Reduktion die Eingabe G_1, G_2 auf G_1', G_2' ab.

(a) Hier konstruieren wir G_1' so, dass $L(G_1') = L(G_1) \cup L(G_2)$ und $G_2' := G_2$. Dann gilt

$$L(G_1') = L(G_2') \Leftrightarrow L(G_1) \cup L(G_2) = L(G_2) \Leftrightarrow L(G_1) \subseteq L(G_2).$$

Für die Konstruktion von G_1' (vgl. Satz 4.31 VL) vereinigen wir die Produktionen von G_1, G_2 und führen ein neues Startsymbol S' mit Produktionen $S' \to S_1 \mid S_2$ hinzu, wobei S_i das Startsymbol der Grammatik G_i ist. Um Überschneidungen zu vermeiden, benennen wir die Nichtterminale in G_1, G_2 vor diesem Schritt in disjunkte Mengen um. Diese Schritte sind offensichtlich berechenbar und G_1' ist weiterhin kontextfrei, da G_1, G_2 kontextfrei sind.

(b) Für diese Aufgabe verwenden wir die Aussage

$$L(c^*)L_1 \cap L(c^*)L_2 = L(c^*)(L_1 \cap L_2)$$

für Sprachen L_1, L_2, die c nicht enthalten. Beweis: Übung.

Die Reduktion konvertiert G_i zu einer CFG G_i' mit $L(G_i') = L(c^*)L(G_i)$, für $i = 1, 2$. Dies lässt sich erreichen, indem man ein neues Startsymbol S' hinzufügt, und die Produktionen $S' \to cS' \mid S$, wobei S das alte Startsymbol der Grammatik ist.

Wir müssen nun $L(G_1) \cap L(G_2) = \emptyset$ genau dann wenn $|L(G_1') \cap L(G_2')| < \infty$ zeigen. Es gilt

$$L(G_1') \cap L(G_2') = L(c^*)L(G_1) \cap L(c^*)L(G_2) = L(c^*)(L(G_1) \cap L(G_2)).$$

Falls $L(G_1) \cap L(G_2) = \emptyset$, gilt $|L(G_1') \cap L(G_2')| = 0 < \infty$. Ansonsten ist $|L(G_1') \cap L(G_2')|$ unendlich, da $|L(c^*)| = \infty$.

Übungsaufgabe Ü11.4. (*Ein Sack Rice*)

Sei $\Sigma := \{0, 1\}$. Entscheiden Sie für (a–d), ob die folgenden Mengen unentscheidbar sind und begründen Sie Ihre Antworten mit dem Satz von Rice (falls anwendbar). Geben Sie dabei die Menge $\mathcal{F}$ genau an und argumentieren Sie, warum die Menge nicht trivial ist (d.h. weder alle berechenbare Funktionen enthält noch leer ist).

(a) $L_1 = \{w \in \Sigma^* \mid \{u \in \Sigma^* \mid \varphi_w(u) = 1\} \text{ ist regulär}\}$

(b) $L_2 = \{w \in \Sigma^* \mid \forall n \in \mathbb{N} : \ \varphi_w(n) = n \cdot (n + 23) + 42\}$

(c) $L_3 = \{w \in \Sigma^* \mid \forall x \in \mathbb{N} : \varphi_w(x) \neq |w|\}$

(d) $L_4 = \{w \in \Sigma^* \mid \forall p \in \mathbb{N}. \ (|w| > p \wedge p \text{ ist prim}) \implies w_p = 0\}$. Erinnerung: $w_p \in \Sigma$ bezeichnet den Buchstaben an der p-ten Stelle im Wort w.

Lösungsskizze.

(a) Sei $\mathcal{F} = \{f \mid f \text{ ist berechenbar } \wedge f^{-1}(1) \text{ ist regulär}\}$. Sei nun g, h mit $g(w) := 1$ und

$$h(w) := \begin{cases} 1 & \text{falls } \exists i \geq 0. \ w = 0^i 1^i \\ 0 & \text{sonst} \end{cases}$$

zwei berechenbare Funktionen. Dann gilt $g \in \mathcal{F}$ und $h \notin \mathcal{F}$. Somit ist $\mathcal{F}$ nicht die Menge aller berechenbarer Funktionen. Damit folgt aus dem Satz von Rice, dass L_1 unentscheidbar ist.

(b) Sei $\mathcal{F} = \{f \mid f \text{ ist berechenbar } \wedge \forall n \in \mathbb{N} : \ f(n) = n \cdot (n + 23) + 42\}$. Dann gilt für die konstante Nullfunktion g, dass $g \notin \mathcal{F}$. Somit ist $\mathcal{F}$ nicht die Menge aller berechenbarer Funktionen. Weiterhin ist $\mathcal{F}$ auch nicht leer, da das Polynom in der Definition berechenbar ist. Somit ist nach Satz von Rice L_2 unentscheidbar.

(c) Die Menge ist unentscheidbar, jedoch ist der Satz von Rice nicht anwendbar. Für den Satz von Rice müsste es eine nicht-triviale Menge $\mathcal{F}$ an berechenbaren Funktionen geben, sodass $L_3 = \{w \in \Sigma^* \mid \varphi_w \in \mathcal{F}\}$. Das Problem ist nun, das zwei Turingmaschinenkodierungen verschiedener Längen existieren können, sodass die Maschinen dieselbe Funktion berechnen, die eine Kodierung die Bedingung erfüllt, die andere jedoch nicht. Formaler:

Je nach Kodierungsfunktion können $v, w \in \Sigma^*$ mit $|v| \neq |w|$ existieren, sodass $\varphi_v = \varphi_w =: g$, $g(x) = |w|$ für ein $x \in \mathbb{N}$ und $g(x) \neq |v|$ für alle $x \in \mathbb{N}$. Somit ist zugleich $g = \varphi_w \notin \mathcal{F}$ und $g = \varphi_v \in \mathcal{F}$, und somit ist $\mathcal{F}$ nicht definierbar.

Der Beweis für die Unentscheidbarkeit kann durch Reduktion von $\overline{\mathcal{H}_0}$ erfolgen. Idee: Länge der TM ($|w|$) auf zweitem Band speichern; Eingabe löschen; TM $M_w[\epsilon]$ simulieren; erstes Band löschen; $|w|$ ausgeben

(d) L_4 ist entscheidbar. Eine TM kann alle Primzahlen kleiner $|w|$ berechnen und an diesen Stellen in w prüfen, ob $w_p = 0$ gilt.