Universität Siegen
Lehrstuhl Theoretische Informatik
Markus Lohrey

Übungsblatt 7

Aufgabe 1
Für zwei Sprachen $A, B \subseteq \{0,1\}^*$ sei die markierte Vereinigung $A \oplus B$ definiert durch

$$A \oplus B = \{0x \mid x \in A\} \cup \{1x \mid x \in B\}.$$

Zeigen Sie folgende Aussagen für beliebige Sprachen $A, B, C \subseteq \{0,1\}^*$:

(a) Es gelten $A \leq A \oplus B$ und $B \leq A \oplus B$.

(b) $A \oplus B$ ist genau dann entscheidbar, wenn A und B entscheidbar sind.

(c) $A \oplus B$ ist genau dann semi-entscheidbar, wenn A und B semi-entscheidbar sind.

(d) Es gilt genau dann $A \oplus B \leq C$, wenn $A \leq C$ und $B \leq C$ gelten.

Lösung

(a) Es gilt $A \leq A \oplus B$: Eine Reduktion erhalten wir mit $f(w) = 0w$ für $w \in \{0,1\}^*$. Denn: Es ist $w \in A$ genau dann, wenn $0w \in \{0x \mid x \in A\}$. Aber da Wörter aus der zweiten Menge mit einer 1 beginnen, ist auch $0w \in A \oplus B$, also $f(w) \in A \oplus B$. Wir erhalten analog $B \leq A \oplus B$ mit der Reduktion $g(w) = 1w$.

(b) Wegen Teil (a) ist klar, dass aus der Entscheidbarkeit von $A \oplus B$ folgt, dass A und B entscheidbar sein müssen. Seien nun also A und B entscheidbar. Aus der Entscheidbarkeit von A folgt per Definition, dass die charakteristische Funktion χ_A berechenbar ist. Somit ist aber die charakteristische Funktion $\chi_{A\oplus B}$ eingeschränkt auf $\{0x \mid x \in A\}$, also $\chi_{A\oplus B}|_{\{0x|x\in A\}}$, berechenbar. Ebenso folgt aus der Entscheidbarkeit von B die Berechenbarkeit von $\chi_{A\oplus B}|_{\{1x|x\in B\}}$. Also ist jeder Wert der Funktion $\chi_{A\oplus B}$ berechenbar und es folgt, dass $A \oplus B$ entscheidbar ist.

(c) Die Hinrichtung gilt analog für Semi-Entscheidbarkeit wegen Teil (a) und für die Rückrichtung kann man mit $\chi'_{A\oplus B}$ genau wie bei (b) argumentieren.

(d) Es gelte zunächst $A \leq C$ und $B \leq C$. Es gibt also totale und berechenbare Funktionen f_A und f_B, die A auf C bzw. B auf C reduzieren. Somit erhalten wir eine Reduktion f von $A \oplus B$ auf C via

$$f(w) = \begin{cases} f_A(v), & w = 0v, \\ f_B(v), & w = 1v, \end{cases}$$

wobei f als zusammengesetzte Funktion wieder total und berechenbar ist. Sei nun umgekehrt $A \oplus B \leq C$. Es gibt also eine Reduktion $f(w) = w'$ mit $w \in A \oplus B$ genau dann, wenn $w' \in C$ ist. Sei $w = 0x$, $x \in \{0,1\}^*$. Dann ist $w \in A \oplus B$ genau dann, wenn $x \in A$ ist. Und $f(w) = f(0x) = w' \in C$ bedeutet, wir erhalten eine Funktion f_A via $f_A(x) = f(0x)$. Denn: $x \in A$ gilt genau dann, wenn $f_A(x) = f(0x) = w' \in C$ ist. Also ist f_A eine Reduktion von A auf C. Analog erhalten wir eine Reduktion von B auf C via $f_B(x) = f(1x)$.

Aufgabe 2
Betrachten Sie die Sprache

$$EQ = \{u\#v \mid L(M_u) = L(M_v)\}.$$

Zeigen Sie, dass weder EQ noch $\overline{EQ}$ semi-entscheidbar sind mit Hilfe des Halteproblems.

Lösung
Wenn wir zeigen könnten, dass sich das Halteproblem H sowohl auf EQ, als auch auf $\overline{EQ}$ reduzieren ließe, dann wüssten wir auch, dass beide Sprachen nicht semi-entscheidbar sein können. Denn das Komplement des Halteproblems $\overline{H}$ ist nicht semi-entscheidbar, womit aus $H \leq EQ$ folgt $\overline{H} \leq \overline{EQ}$, woraus wiederum die nicht Semi-Entscheidbarkeit von $\overline{EQ}$ folgt. Das gleiche gilt also auch für EQ.
Zur Erinnerung: Das (allgemeine) Halteproblem ist die Sprache

$$H = \{w\#x \mid M_w \text{ hält auf } x\}.$$

Für die Reduktion $H \leq EQ$ definieren wir eine Reduktionsfunktion f durch $f(w\#x) = w'\#v$ wie folgt: M_v terminiert auf jeder Eingabe, also gilt $L(M_v) = \Sigma^*$. Die Turingmaschine $M_{w'}$ ignoriert seinen Input und überschreibt ihn mit x. Anschließend simuliert $M_{w'}$ die TM M_w (auf x). Mit anderen Worten: Falls M_w auf x hält, gilt $L(M_{w'}) = \Sigma^*$ und falls M_w auf x nicht hält, gilt $L(M_{w'}) = \emptyset$. Es gilt also $L(M_{w'}) = L(M_v)$ genau dann, wenn M_w auf x hält. Die Reduktion $H \leq \overline{EQ}$ funktioniert analog mit $L(M_v) = \emptyset$.

Aufgaben zum Postschen Korrespondenzproblem

Die Wortpaare, die als Eingabe für das Postsche Korrespondenzproblem (PCP) benötigt werden, wollen wir im Folgenden als Matrix darstellen und bezeichnen diese als *PCP-Instanz*. Beispiel: $I = ((01, 10), (1, 11), (000, 1))$ stellen wir dar als

$$\begin{pmatrix} 01 & 1 & 000 \\ 10 & 11 & 1 \end{pmatrix}.$$

Aufgabe 3

(a) Entscheiden Sie die beiden folgenden PCP-Instanzen:

$$\begin{pmatrix} a & ba & abb & bab \\ ab & ab & bb & abb \end{pmatrix} \qquad \begin{pmatrix} aaaa & aa \\ aaa & aaaaa \end{pmatrix}$$

Geben Sie im positiven Fall eine PCP-Lösung an und beweisen Sie im negativen Fall, dass keine PCP-Lösung existiert.

(b) Zeigen Sie, dass $\text{PCP}_{m,1}$ entscheidbar ist, also PCP eingeschränkt auf unäre Alphabete.

(c) Überlegen Sie sich weitere Spezialfälle bzw. Einschränkungen, wo PCP entscheidbar ist. Denken Sie vor allem an Fälle, wo man leicht sieht, dass es *keine* Lösung gibt.

Lösung

(a) Bei der ersten PCP-Instanz sehen wir, dass es nur ein Paar (x_i, y_i) gibt, bei welchem beide Wörter mit dem gleichen Zeichen beginnen, nämlich das erste (a, ab). Wir sehen, dass jetzt das zweite Wort ein b mehr enthält als das erste. Es gibt jedoch kein Paar der PCP-Instanz, wo das erste Wort (mindestens) ein b mehr enthält als das zweite. Mit anderen Worten: Jedes Wort $a\, x_{i_2} \cdots x_{i_m}$ wird stets weniger b enthalten als $ab\, y_{i_2} \cdots y_{i_m}$. Somit existiert keine PCP-Lösung für die erste Instanz.

Für die zweite PCP-Instanz finden wir leicht eine Lösung, indem wir uns anschauen, wann oben und unten (in der ersten und zweiten Komponente) die gleiche Anzahl an a steht. Eine (minimale) Lösung ist zum Beispiel $(1, 1, 1, 2)$, wo beide Wörter gleich a^{14} sind.

(b) Wir können die Idee zur Lösung der zweiten Instanz aus Teil a verallgemeinern. Dazu müssen wir allerdings erstmal zwei Spezialfälle analysieren:

Sei I eine beliebige PCP-Instanz der Länge $k \leq m$ über $\text{PCP}_{m,1}$. Falls es ein Paar (x_i, y_i) in I gibt mit $x_i = y_i$, ist I lösbar und eine Lösung ist (i). Falls stets $|x_i| < |y_i|$ oder $|x_i| > |y_i|$ für alle $1 \leq i \leq k$ gilt, so gibt es keine Lösung, da jedes zusammengesetzte Wort aus den x_i stets kürzer bzw. stets länger ist als das Gegenstück mit den y_i.

Es bleibt also noch der dritte, allgemeine Fall, dass es ein i gibt mit $|x_i| < |y_i|$ und ein j gibt mit $|x_j| > |y_j|$. Dann genügen uns die beiden Paare (x_i, y_i) und (x_j, y_j) bereits, um eine Lösung zu finden. Sei $|y_i| - |x_i| = d_i$ und $|x_j| - |y_j| = d_j$. Nach Voraussetzung gilt $d_i, d_j > 0$. Für jedes Paar (d_i, d_j) positiver ganzer Zahlen gibt es Konstanten (c_1, c_2) mit $c_1 d_i = c_2 d_j$, nämlich zum Beispiel $c_1 = d_j$ und $c_2 = d_i$. Somit erhalten wir eine PCP-Lösung durch $(i, i, \ldots, i, j, j, \ldots, j)$ mit d_j vielen i und d_i vielen j. Bei Teil a ist $i = 2$ und $j = 1$ und es gilt $d_i = 3$ und $d_j = 1$. Dies liefert uns die Lösung $(i, j, j, j) = (2, 1, 1, 1)$, die wir in einer anderen Reihenfolge bereits herausgefunden haben.

(c) Wir erhalten auf jeden Fall eine Lösung für eine PCP-Instanz I, wenn es ein paar (x_i, y_i) in I gibt mit $x_i = y_i$. Wir erhalten auf jeden Fall eine unlösbare Instanz I,

(1) falls stets $|x_i| < |y_i|$ oder $|x_i| > |y_i|$ für alle i gilt,

(2) falls stets $|x_i|_a < |y_i|_a$ oder $|x_i|_a > |y_i|_a$ für alle i und für ein Zeichen $a \in \Sigma$ gilt (Spezialfall: Zeichen a taucht *nur* in der ersten oder in der zweiten Komponente auf),

(3) falls es kein Paar (x_i, y_i) gibt, wo x_i und y_i ein gemeinsames erstes oder ein gemeinsames letztes Zeichen haben.

(4) Man kann (1)-(3) auch kombinieren, um die Anzahl der „nutzbaren" Paare (x_i, y_i) zu reduzieren und am Ende zu einer unlösbaren Instanz zu gelangen (ähnlich wie bei Teil a). Dies liefert aber eher einen aufwendigeren Algorithmus, als ein leicht überprüfbares Kriterium.

Aufgabe 4
Zeigen Sie, dass folgende PCP-Variante PCP* entscheidbar ist:
Gegeben: Eine PCP-Instanz $\begin{pmatrix} x_1 & \cdots & x_k \\ y_1 & \cdots & y_k \end{pmatrix}$.
Frage: Gibt es Indexfolgen $i_1, \ldots, i_m$ und $j_1, \ldots, j_n$ mit $m, n \geq 1$ und $x_{i_1} \cdots x_{i_m} = y_{j_1} \cdots y_{j_n}$?
Hinweis: Reduzieren Sie PCP* auf das Schnittproblem für reguläre Sprachen.

Lösung
Man beachte, dass der Unterschied zum normalen PCP recht groß ist, auch wenn die Fragestellung zunächst recht ähnlich aussieht: Wir können hier die Komponenten unabhängig voneinander auswählen und das sowohl unabhängig von der Paarung (x_i, y_i), als auch unabhängig von der Anzahl der genutzten ersten Komponenten vs. der genutzten zweiten Komponenten. Somit ist es uns möglich, aus $x_{i_1} \cdots x_{i_m}$ und $y_{j_1} \cdots y_{j_n}$ jeweils reguläre Ausdrücke zu machen und dann eine Reduktion zum Schnittproblem für reguläre Sprachen anzugeben. Bemerkung: Wäre $n = m$, so ginge die gleiche Argumentation nicht und wir müssten uns mit kontextfreien Sprachen auseinandersetzen!
Sei I die Input-Instanz. Jedes x_i und y_i ist ein String über Σ und kann somit als regulärer Ausdruck aufgefasst werden. Alle Wörter der Form $x_{i_1} \cdots x_{i_m}$ erhalten wir durch den regulären Ausdruck $\alpha = (x_1|x_2|\ldots|x_k)^*$. Analog erhalten wir alle Wörter der Form $y_{j_1} \cdots y_{j_n}$ durch $\beta = (y_1|y_2|\ldots|y_k)^*$. Die Instanz I hat also genau dann eine Lösung, wenn $L(\alpha) \cap L(\beta)$ nicht leer ist, also wenn die beiden regulären Sprachen mindestens ein gemeinsames Wort enthalten. Die Funktion $f(I) = L_I = L(\alpha) \cap L(\beta)$ ist außerdem offenbar total und berechenbar. Das Schnittproblem regulärer Sprachen ist jedoch entscheidbar. Man kann aus α und β leicht einen NFA bzw. einen DFA machen und anschließend den Produktautomaten konstruieren, um das Schnittproblem zu lösen.