Prof. Dr. Jasmin Blanchette
Jannis Limperg
Lydia Kondylidou

Ludwig-Maximilians-Universität München
Institut für Informatik
Abgabe bis 12. Juli 2023, 10:00 Uhr

Lösungsvorschlag zur Übung 12 zur Vorlesung

Formale Sprachen und Komplexität

FSK12-1 ***PCP-Varianten*** (2 Punkte)

a) Wir betrachten das LPCP-Problem, eine Variante von PCP, bei der die ‚Spielsteine' auch das leere Wort enthalten können. Eine Instanz von LPCP mit Alphabet Σ ist also eine endliche Folge von Paaren $(x_1, y_1), \ldots, (x_n, y_n)$ mit $x_i, y_i \in \Sigma^*$ für $i = 1, \ldots, n$ (wohingegen bei PCP gilt: $x_i, y_i \in \Sigma^+$). Eine Lösung der Instanz K ist wie bei PCP eine endliche Folge von Indices $i_1, \ldots, i_m \in \mathbb{N}$ sodass $x_{i_1} \cdots x_{i_m} = y_{i_1} \cdots y_{i_m}$.
Zeigen Sie durch Reduktion von PCP auf LPCP, dass LPCP für $|\Sigma| \geq 2$ unentscheidbar ist.

LÖSUNGSVORSCHLAG:

Wir zeigen PCP $\leq$ LPCP. Da PCP unentscheidbar ist, ist damit auch LPCP unentscheidbar.

Eine Instanz K von PCP ist auch eine Instanz von LPCP, wir können als Reduktionsfunktion f also $f(K) = K$ wählen. Offensichtlich ist f total und berechenbar und es ist auch K lösbar g.d.w. $f(K)$ lösbar ist.

b) Sind die folgenden Instanzen K_1, K_2 von LPCP lösbar? Wenn ja, geben Sie eine Lösung (also eine geeignete Folge von Indizes) an. Wenn nein, beweisen Sie, dass die Instanz keine Lösung hat.

$$K_1 = \left(\begin{bmatrix} ab \\ aba \end{bmatrix}, \begin{bmatrix} bab \\ ba \end{bmatrix}, \begin{bmatrix} aa \\ \varepsilon \end{bmatrix}, \begin{bmatrix} \varepsilon \\ bb \end{bmatrix} \right)$$
$$K_2 = \left(\begin{bmatrix} a \\ ba \end{bmatrix}, \begin{bmatrix} bc \\ cbaa \end{bmatrix}, \begin{bmatrix} baa \\ \varepsilon \end{bmatrix}, \begin{bmatrix} \varepsilon \\ aab \end{bmatrix} \right)$$

LÖSUNGSVORSCHLAG:

K_1 ist nicht lösbar. Wir bemerken zunächst, dass die Instanz symmetrisch ist: wenn man das obere und untere Wort vertauscht sowie alle a's durch b's ersetzt

und alle b's durch a's, wird der erste Stein zum zweiten und der dritte zum vierten.

Nun betrachten wir die Steine, mit denen eine Lösung anfangen könnte:

- $\begin{bmatrix} aa \\ \varepsilon \end{bmatrix}$. Die einzige Möglichkeit, die Folge fortzusetzen, ohne dass die Wörter unterschiedlich werden, ist wieder $\begin{bmatrix} aa \\ \varepsilon \end{bmatrix}$. Dadurch kommen wir aber nie zu einer Folge mit gleichen Wörtern oben und unten.
- $\begin{bmatrix} ab \\ aba \end{bmatrix}$. Hier haben wir mehrere Möglichkeiten, die Folge fortzusetzen:
  - $\begin{bmatrix} ab \\ aba \end{bmatrix}$. Dadurch werden die Wörter unterschiedlich.
  - $\begin{bmatrix} bab \\ ba \end{bmatrix}$. Analog.
  - Ein oder mehr $\begin{bmatrix} aa \\ \varepsilon \end{bmatrix}$. Dann kann man wiederum die Folge nur fortsetzen, indem man diesen Stein wiederholt, kommt aber dadurch nie zu einer Lösung.
  - Ein oder mehr $\begin{bmatrix} \varepsilon \\ bb \end{bmatrix}$. Analog.
- $\begin{bmatrix} \varepsilon \\ bb \end{bmatrix}$. Symmetrisch zu $\begin{bmatrix} aa \\ \varepsilon \end{bmatrix}$.
- $\begin{bmatrix} bab \\ ba \end{bmatrix}$. Symmetrisch zu $\begin{bmatrix} ab \\ aba \end{bmatrix}$.

Kein Anfangsstein führt zu einer Lösung, also ist die Instanz unlösbar.

K_2 hat die Lösung $3, 1, 4, 2, 3$, also

$$\begin{bmatrix} baa \\ \varepsilon \end{bmatrix}, \begin{bmatrix} a \\ ba \end{bmatrix}, \begin{bmatrix} \varepsilon \\ aab \end{bmatrix}, \begin{bmatrix} bc \\ cbaa \end{bmatrix}, \begin{bmatrix} baa \\ \varepsilon \end{bmatrix}$$

Diese Steine ergeben oben und unten das Wort $baaabcbaa$.

c) Wir betrachten das EVENPCP-Problem, eine Variante von LPCP, bei der die Wörter auf den Spielsteinen gerade Länge haben müssen. Eine Instanz von EVENPCP ist also eine endliche Folge von Paaren $(x_1, y_1), \ldots, (x_n, y_n)$ mit $x_i, y_i \in \{w \mid w \in \Sigma^* \text{ und } |w| \text{ gerade}\}$ für $i = 1, \ldots, n$. Beispielsweise ist $(ab, aaba)$ ein erlaubter Spielstein; (ab, aab) aber nicht.

Zeigen Sie durch Reduktion von PCP auf EVENPCP, dass EVENPCP für $|\Sigma| \geq 2$

unentscheidbar ist.

LÖSUNGSVORSCHLAG:

Wir zeigen PCP $\leq$ EVENPCP. Da PCP unentscheidbar ist, folgt daraus, dass auch EVENPCP unentscheidbar ist.

Sei $K = ((x_1, y_1), \ldots, (x_n, y_n))$ eine Instanz von PCP mit Alphabet Σ. Wir definieren für jedes Wort $w = z_1 \cdots z_p$ mit $z_i \in \Sigma$ das Wort $f(w) = z_1 z_1 \cdots z_p z_p$. Damit ist $|f(w)|$ gerade. Definiere nun die EVENPCP-Instanz $f(K) = ((f(x_1), f(y_1)), \ldots, (f(x_n), f(y_n)))$. Offensichtlich ist f total und berechenbar. Außerdem ist f kompatibel mit Konkatenation: $f(u \circ v) = f(u) \circ f(v)$ für alle $u, v \in \Sigma^*$. Es gilt somit:

- Wenn $K \in$ PCP ist, dann hat K eine Lösung $i_1, \ldots, i_m$ mit $x_{i_1} \cdots x_{i_m} = y_{i_1} \cdots y_{i_m}$. Dann ist auch

  $$f(x_{i_1}) \cdots f(x_{i_m}) = f(x_{i_1} \cdots x_{i_m}) = f(y_{i_1} \cdots y_{i_m}) = f(y_{i_1}) \cdots f(y_{i_m})$$

  und es ist also $i_1, \ldots, i_m$ eine Lösung für $f(K)$, somit $f(K) \in$ EVENPCP.
- Wenn $f(K) \in$ EVENPCP ist, dann hat $f(K)$ eine Lösung $i_1, \ldots, i_m$ mit $f(x_{i_1}) \cdots f(x_{i_m}) = f(y_{i_1}) \cdots f(y_{i_m})$. Wir definieren die Funktion g, die aus einem Wort mit gerader Länge p jeden zweiten Buchstaben entfernt: $g(z_1 z_2 z_3 \cdots z_{p-1} z_p) = z_1 z_3 \cdots z_{p-1}$. Damit ist g linksinvers zu f, d.h. $g(f(w)) = w$ für alle $w \in \Sigma^*$. Es gilt also:

  $$x_{i_1} \cdots x_{i_m} = g(f(x_{i_1} \cdots x_{i_m})) = g(f(x_{i_1}) \cdots f(x_{i_m})) = g(f(y_{i_1}) \cdots g(y_{i_m})) = g(f(y_{i_1} \cdots y_{i_m})) = y_{i_1} \cdots y_{i_m}$$

  Somit ist $i_1, \ldots, i_m$ auch eine Lösung für K und $K \in$ PCP.

d) Für Mengen Σ und Δ nennen wir eine Funktion $f : \Sigma^* \to \Delta^*$ einen *Homomorphismus* (siehe auch Aufgabe FSK6-4), wenn gilt:

$$\begin{aligned} f(\varepsilon) &= \varepsilon \\ f(u \circ v) &= f(u) \circ f(v) \quad \forall u, v \in \Sigma^* \end{aligned}$$

Wir definieren das Problem HOMPCP. Eine Instanz dieses Problem ist ein 4-Tupel (Σ, Δ, f, g), wobei Σ und Δ endliche Mengen sind und $f, g : \Sigma^* \to \Delta^*$ Homomorphismen. Eine Lösung der Instanz ist ein Wort $w \in \Sigma^+$ sodass gilt: $f(w) = g(w)$.

Zeigen Sie durch Reduktion von LPCP auf HOMPCP, dass HOMPCP für $|\Delta| \geq 2$ unentscheidbar ist.

LÖSUNGSVORSCHLAG:

Wir zeigen LPCP $\leq$ HOMPCP. Da LPCP unentscheidbar ist, folgt daraus, dass auch HOMPCP unentscheidbar ist.

Sei $K = ((x_1, y_1), \ldots, (x_n, y_n))$ eine Instanz von LPCP mit Alphabet Γ. Wir definieren eine Instanz $F(K) = (\Sigma, \Delta, f, g)$ von HOMPCP wie folgt:

$$\Sigma = \{1, \ldots, n\}$$
$$\Delta = \Gamma$$
$$f(i) = \begin{cases} \varepsilon & \text{für } i = \varepsilon \\ x_i & \text{für } i \in \Sigma \\ f(j) \circ f(k) & \text{für } i = j \circ k;\ j, k \in \Sigma^+ \end{cases}$$
$$g(i) = \begin{cases} \varepsilon & \text{für } i = \varepsilon \\ y_i & \text{für } i \in \Sigma \\ g(j) \circ g(k) & \text{für } i = j \circ k;\ j, k \in \Sigma^+ \end{cases}$$

Wörter aus Σ^+ sind dabei endliche, nicht leere Folgen $i_1 \cdots i_m$ von Indizes aus der Indexmenge Σ. Offensichtlich sind f und g per Definition Homomorphismen.

Nun gilt:

- Eine Lösung für K ist eine endliche, nicht leere Folge von Indizes $i_1, \ldots, i_m \in \Sigma$. Sei $w = i_1 \cdots i_m$. Dann gilt:

  $$f(w) = f(i_1) \cdots f(i_m) = x_{i_1} \cdots x_{i_m} = y_{i_1} \cdots y_{i_m} = g(i_1) \cdots g(i_m) = g(w)$$

  Somit ist w eine Lösung von $F(K)$.
- Umgekehrt ist eine Lösung von $F(K)$ ein nicht leeres Wort $w = i_1 \cdots i_m$ sodass

  $$x_{i_1} \cdots x_{i_m} = f(i_1) \cdots f(i_m) = f(w) = g(w) = g(i_1) \cdots g(i_m) = y_{i_1} \cdots y_{i_m}$$

  Somit ist $i_1, \ldots, i_m$ eine Lösung von K.

F ist außerdem total und berechenbar und somit eine valide Reduktionsfunktion.

Übrigens kann man analog auch HOMPCP auf LPCP reduzieren – HOMPCP und LPCP sind letztlich nur zwei Formulierungen desselben Problems. Die Formulierung mit Homomorphismen ist manchmal nützlich, weil Homomorphismen viele hübsche Eigenschaften haben.

FSK12-2 ***Beweise prüfen*** (2 Punkte)

In den folgenden Teilaufgaben betrachten wir jeweils einen Beweis, der einen Fehler enthält. Identifizieren Sie diesen Fehler (mit kurzer Begründung).

a) Beweisen oder widerlegen Sie: Die Sprache

$$D = \{w \in \{0,1\}^* \mid M_w \text{ akzeptiert } w \text{ nicht}\}$$

ist semi-entscheidbar.

Beweis:
D ist semi-entscheidbar. Um das zu zeigen, konstruieren wir eine DTM M, die D semi-entscheidet. Das heißt, dass M für alle Eingaben $w \in D$ hält und für alle Eingaben $w \notin D$ nicht hält.

Angenommen, es gäbe so eine Turingmaschine M. Betrachte ein Wort $w \in \{0,1\}^*$.

- Wenn $w \in D$ ist, dann akzeptiert M_w die Eingabe w. Somit akzeptiert auch M das Wort w.
- Wenn $w \notin D$ ist, dann hält M_w mit Eingabe w nicht. Somit akzeptiert auch M das Wort w nicht.

M semi-entscheidet also D.

LÖSUNGSVORSCHLAG:

Der Beweis enthält zwei Fehler:

- $w \in D$ bedeutet, dass M_w die Eingabe w *nicht* akzeptiert. Die zwei Stichpunkte in der Lösung sind also vertauscht.
- Selbst wenn wir diesen Fehler beheben, nimmt die Lösung an, dass ein M mit der gewünschten Eigenschaft existiert, aber wir haben das nie gezeigt. Der Beweis ist also zirkulär: „Unter der Annahme, dass M existiert, existiert M."

Tatsächlich ist D nicht semi-entscheidbar, d.h. die Aussage ist falsch. Intuition: Die einzige Möglichkeit, zu testen, ob M_w das Wort w nicht akzeptiert, ist, M_w auf w auszuführen. Wenn M_w dann beliebig lange läuft, kann man nie sagen, ob M_w noch halten (und damit akzeptieren) wird oder nicht.

b) Sei $L_u = \{w\#x \mid w, x \in \{0,1\}^* \text{ und } x \in L(M_w)\}$. Diese Sprache ist semi-entscheidbar, aber nicht entscheidbar.

Zeigen Sie: Die Sprache $L_r = \{w \in \{0,1\}^* \mid L(M_w) \text{ ist regulär}\}$ ist unentscheidbar.

Beweis:
Wir reduzieren L_u auf L_r. Da L_u unentscheidbar ist, folgt daraus, dass L_r unentscheidbar ist.

Sei $v \in \{0,1,\#\}^*$. Wir definieren die Reduktionsfunktion f durch

$$f(v) = \begin{cases} \langle M_3 \rangle & \text{falls } v = w\#x \text{ und } M_w \text{ akzeptiert } x \\ \langle M_4 \rangle & \text{sonst} \end{cases}$$

Dabei ist M_3 eine Turingmaschine, die die reguläre Sprache $\{0,1\}^*$ akzeptiert. M_4 ist eine Turingmaschine, die die nicht-reguläre Sprache $\{0^n1^n \mid n \in \mathbb{N}\}$ akzeptiert. $\langle M_i \rangle$ ist die Binärkodierung der jeweiligen Turingmaschine.

M_3, M_4 und die Binärkodierungen sind offensichtlich berechenbar, also ist f berechenbar (und offensichtlich total). Weiterhin gilt:

$$\begin{array}{ll} & v \in L_u \\ \text{g.d.w.} & v = w\#x \text{ und } x \in L(M_w) \\ \text{g.d.w.} & M_{f(v)} \text{ akzeptiert eine reguläre Sprache} \\ \text{g.d.w.} & f(v) \in L_r \end{array}$$

Somit ist f eine valide Reduktionsfunktion und $L_u \leq L_r$.

LÖSUNGSVORSCHLAG:

Die Reduktionsfunktion f ist nicht berechenbar. Um zu entscheiden, ob $f(v) = M_3$ oder $f(v) = M_4$, müssen wir entscheiden, ob M_w das Wort x akzeptiert. Das ist aber bekanntermaßen unentscheidbar.

c) Sei $A = \{w \in \{0,1\}^* \mid M_w \text{ hält bei Eingabe 36 mit Ausgabe 42 an}\}$. Zeigen Sie durch Reduktion von H_0 auf A, dass A unentscheidbar ist.

Beweis:
Wir zeigen $H_0 \leq A$. Da H_0 unentscheidbar ist, folgt daraus, dass auch A unentscheidbar ist.

Wir definieren die Reduktionsfunktion $f : \{0,1\}^* \to \{0,1\}^*$ wie folgt. Für $w \in \{0,1\}^*$ berechnet f zunächst die Turingmaschine M_w, erstellt daraus eine Turingmaschine T und berechnet anschließend deren Binärcodierung $\langle T \rangle$. Dabei verhält sich T wie folgt:

- Prüfe, ob auf dem Band die Zahl 36 (in Binärdarstellung) steht. Falls nein, gehe in eine Endlosschleife über.
- Führe M_w aus.

- Falls M_w anhält, schreibe die Zahl 42 (in Binärdarstellung) auf das Band und akzeptiere.

Die Funktion f ist offensichtlich total. Sie ist auch berechenbar, da T konstruierbar ist. Weiterhin gilt:

$$\begin{array}{ll} & w \in H_0 \\ \text{g.d.w.} & M_w \text{ hält auf leerem Band} \\ \text{g.d.w.} & M_{f(w)} \text{ hält für Eingabe 36 mit Ausgabe 42} \\ \text{g.d.w.} & f(w) \in A \end{array}$$

Somit ist f eine valide Reduktionsfunktion und $H_0 \leq A$.

LÖSUNGSVORSCHLAG:

Die von f für w konstruierte Turingmaschine löscht nicht das Band, bevor sie M_w ausführt. Somit testet sie nicht, ob M_w auf dem leeren Band hält. Die Aussage „M_w hält auf dem leeren Band g.d.w. $M_{f(w)}$ für Eingabe 36 mit Ausgabe 42 hält“ ist also falsch.

FSK12-3 $\mathcal{P}$ ***und*** $\mathcal{NP}$ (0 Punkte)

a) Zeigen Sie, dass $\mathcal{P}$ unter Vereinigung abgeschlossen ist.

LÖSUNGSVORSCHLAG:

Seien $L, M \in \mathcal{P}$. Wir müssen zeigen: $L \cup M \in \mathcal{P}$.

Da $L, M \in \mathcal{P}$, gibt es deterministische 1-Band-Turingmaschinen A_L, A_M mit $L(A_L) = L, L(A_M) = M$, die für jede Eingabe nur polynomiell lange brauchen.

Wir konstruieren nun eine 2-Band-Turingmaschine B mit $L(B) = L \cup M$ wie folgt:

- Zuerst wird der Inhalt vom Eingabeband (Band 1) unverändert auf Band 2 kopiert und beide Köpfe wieder auf die Startposition gefahren.
  Das braucht pro Buchstabe 4 Schritte:
  - Buchstaben auf Band 1 lesen und Kopf nach rechts fahren
  - Buchstaben auf Band 2 schreiben und Kopf nach rechts fahren (der Buchstabe muss im Zustand der Turingmaschine gemerkt werden)
  - Am Ende nochmal 1 Schritt pro Buchstabe und Band: Kopf wieder nach links fahren

Es kommen insgesamt noch konstant viele Schritte dazu, um den Kopf wieder genau auf den Startbuchstaben zu stellen.

Insgesamt bei Eingabelänge n also $O(n)$ Schritte.

- Danach wird die Turingmaschine A_L auf Band 1 simuliert:

  Jede Operation von A_L bis zur Akzeptanz wird durchgeführt, aber alle Lese- und Schreibeoperationen beziehen sich stattdessen auf Band 1.

  Im Akzeptanzfall wird statt wirklich zu akzeptieren das Symbol $\top$ auf die Kopfposition auf Band 1 geschrieben, sonst $\bot$; der Kopf wird dabei nicht bewegt.

  Dies braucht $O(p(n))$ viele Schritte für ein geeignetes Polynom p, da A_L nur polynomiell viele Schritte braucht.

  Auch das Schreiben von $\top$ bzw. $\bot$ braucht nur konstant viele Extraschritte.

- Danach wird die Turingmaschine A_M auf Band 2 simuliert:

  Jede Operation von A_M bis zur Akzeptanz wird durchgeführt, aber alle Lese- und Schreibeoperationen beziehen sich stattdessen auf Band 2.

  Im Akzeptanzfall wird statt wirklich zu akzeptieren das Symbol $\top$ auf die Kopfposition auf Band 2 geschrieben, sonst $\bot$; der Kopf wird dabei nicht bewegt.

  Dies braucht $O(q(n))$ viele Schritte für ein geeignetes Polynom q, da A_M nur polynomiell viele Schritte braucht.

  Auch das Schreiben von $\top$ bzw. $\bot$ braucht nur konstant viele Extraschritte.

- Danach wird geschaut, ob auf Band 1 oder auf Band 2 unter dem Kopf ein $\top$ steht. Falls dies der Fall ist, akzeptiert die Turingmaschine B.

  Dies kann im Zustandsraum von B geschehen, da es nur endliche viele (4) Kombinationsmöglichkeiten von $\top$ und $\bot$ auf den beiden Kopfpositionen gibt, es braucht also nur $O(c)$ viele Schritte mit einer Konstanten c.

Insgesamt braucht die Turingmaschine B damit bei einem Wort der Eingabelänge n nur polynomiell viele Schritte: $O(n + p(n) + q(n) + c)$ ist ebenfalls polynomiell, da es sich um die Summe von 4 Polynomen handelt (auch Konstanten sind Polynome).

B berechnet auch das richtige, da B genau dann akzeptiert, wenn A_L oder A_M akzeptieren würden, also ist $L(B) = \{w \mid w \in L(A_L) \vee w \in L(A_M)\} = L \cup M$.

Da B polynomiell ist und das richtige berechnet, ist damit $L \cup M \in \mathcal{P}$.

b) Zeigen Sie, dass $\mathcal{P}$ unter Schnitt abgeschlossen ist.

LÖSUNGSVORSCHLAG: Wie (a), nur dass B akzeptiert, wenn A_L und A_M akzeptieren würden, also wenn am Ende auf beiden Bändern unter dem Kopf ein $\top$ steht. Dies ist aus den gleichen Gründen polynomiell und $L(B) = \{w \mid w \in L(A_L) \wedge w \in L(A_M)\} = L \cap M$

c) Zeigen Sie, dass $\mathcal{NP}$ unter Vereinigung abgeschlossen ist.

LÖSUNGSVORSCHLAG: Wie (a), nur mit einer nichtdeterministischen Turingmaschine.

d) Zeigen Sie, dass $\mathcal{NP}$ unter Schnitt abgeschlossen ist.

LÖSUNGSVORSCHLAG: Wie (b), nur mit einer nichtdeterministischen Turingmaschine.

FSK12-4 ***PCP und DFA*** (0 Punkte)

Betrachten Sie das folgende Verfahren V:

V berechnet für eine PCP-Instanz K über einem Alphabet Σ (d. h. $K = ((x_1, y_1), \ldots, (x_k, y_k))$ mit $x_i, y_i \in \Sigma^+$) und einen deterministischen endlichen Automaten $A = (Z, \Sigma, \delta, z_0, E)$ einen Wahrheitswert.

Dafür wird zunächst die Menge $M \subseteq Z \times Z$ folgendermaßen iterativ bestimmt:

- Anfangs ist $M = \varnothing$
- In jedem Schritt wird für alle $(z_i, z_j) \in M$, sowie für (z_0, z_0) (selbst, wenn es nicht in M enthalten ist) und jedes Wortpaar/Spielstein $T = (x_l, y_l)$ aus K bestimmt, welcher Zustand $z_{i'}$ von z_i aus mit dem oberen Wort von T erreicht wird (d.h. $z_{i'} = \widehat{\delta}(z_i, x_l)$) und welcher Zustand $z_{j'}$ von z_j aus mit dem unteren Wort von T erreicht wird (d.h. $z_{j'} = \widehat{\delta}(z_j, y_l)$).

  Das Tupel $(z_{i'}, z_{j'})$ wird zu M hinzugefügt, wenn es nicht bereits in M vorhanden ist.
- Wenn sich die Menge M nicht mehr ändert, ist dies die engültige Gestalt von M und die Berechnung von M ist beendet.

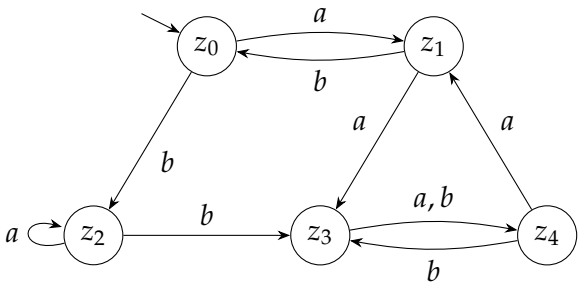
Gibt es nun einen Zustand $z \in Z$ mit $(z, z) \in M$, so wird „wahr“ zurückgegeben, ansonsten „falsch“.

Hinweis: Die Endzustandsmenge E wird vom Verfahren V nicht verwendet.

a) Berechnen Sie die Menge M für die PCP-Instanz K:

$$\left\{ \begin{bmatrix} bba \\ bb \end{bmatrix}, \begin{bmatrix} abba \\ abbab \end{bmatrix}, \begin{bmatrix} aba \\ bab \end{bmatrix}, \begin{bmatrix} abaaba \\ bbb \end{bmatrix}, \begin{bmatrix} ba \\ bbaabbab \end{bmatrix} \right\}$$

und den Automaten A:



Was gibt das Verfahren zurück?

LÖSUNGSVORSCHLAG:

Zur Vereinfachung betrachten wir nur, welche Zustandspaare in jedem Schritt zu M hinzugefügt werden, falls sie noch nicht enthalten sind.

- $M = \emptyset$
- Betrachte (z_0, z_0):

$$\begin{aligned} T = (bba, bb) &: (z_4, z_3) \in M \\ T = (abba, abbab) &: (z_2, z_3) \in M \\ T = (aba, bab) &: (z_1, z_3) \in M \\ T = (abaaba, bbb) &: (z_1, z_4) \in M \\ T = (ba, bbaaabbab) &: (z_2, z_3) \in M \end{aligned}$$

- Betrachte (z_4, z_3):

$$
\begin{aligned}
T = (bba, bb) &: (z_1, z_3) \in M \\
T = (abba, abbab) &: (z_2, z_0) \in M \\
T = (aba, bab) &: (z_1, z_0) \in M \\
T = (abaaba, bbb) &: (z_1, z_4) \in M \\
T = (ba, bbaaabbab) &: (z_4, z_3) \in M
\end{aligned}
$$

- Betrachte (z_2, z_3):

$$
\begin{aligned}
T = (bba, bb) &: (z_1, z_3) \in M \\
T = (abba, abbab) &: (z_1, z_0) \in M \\
T = (aba, bab) &: (z_4, z_0) \in M \\
T = (abaaba, bbb) &: (z_1, z_4) \in M \\
T = (ba, bbaaabbab) &: (z_4, z_3) \in M
\end{aligned}
$$

  An dieser Stelle können wir beobachten, dass die rechten Seiten der *Ergebnis*-Tupel identisch ist zu denen für (z_4, z_3). Dies liegt daran, dass das Verfahren die beiden Seiten der Tupel komplett unabhängig voneinander behandelt. Wir können also, um Arbeit zu sparen, falls wir für eine der Seiten des *Eingabe*-Tupels bereits die *Ergebnis*-Tupel kennen, die entsprechende Seite der *Ergebnis*-Tupel einfach ablesen, statt sie erneut zu berechnen.

- Betrachte (z_1, z_3):

$$
\begin{aligned}
T = (bba, bb) &: (z_2, z_3) \in M \\
T = (abba, abbab) &: (z_4, z_0) \in M \\
T = (aba, bab) &: (z_1, z_0) \in M \\
T = (abaaba, bbb) &: (z_1, z_4) \in M \\
T = (ba, bbaaabbab) &: (z_1, z_3) \in M
\end{aligned}
$$

- Betrachte (z_1, z_4):

$$
\begin{aligned}
T = (bba, bb) &: (z_2, z_4) \in M \\
T = (abba, abbab) &: (z_4, z_3) \in M \\
T = (aba, bab) &: (z_1, z_3) \in M \\
T = (abaaba, bbb) &: (z_1, z_3) \in M \\
T = (ba, bbaaabbab) &: (z_1, z_3) \in M
\end{aligned}
$$

An dieser Stelle sind alle Zustände z_0, z_1, z_2, z_3, z_4 mindestens einmal (auf der jeweils korrekten Seite für die folgenden Schritte) in einem der betrachteten Tupel vorgekommen. Wir können also ab hier immer ablesen statt zu berechnen.

- Betrachte (z_2, z_0):

$$\begin{aligned} T = (bba, bb) &: (z_1, z_3) \in M \\ T = (abba, abbab) &: (z_1, z_3) \in M \\ T = (aba, bab) &: (z_4, z_3) \in M \\ T = (abaaba, bbb) &: (z_1, z_4) \in M \\ T = (ba, bbaaabbab) &: (z_4, z_3) \in M \end{aligned}$$

- Betrachte (z_1, z_0):

$$\begin{aligned} T = (bba, bb) &: (z_2, z_3) \in M \\ T = (abba, abbab) &: (z_4, z_3) \in M \\ T = (aba, bab) &: (z_1, z_3) \in M \\ T = (abaaba, bbb) &: (z_1, z_4) \in M \\ T = (ba, bbaaabbab) &: (z_1, z_3) \in M \end{aligned}$$

- Betrachte (z_4, z_0):

$$\begin{aligned} T = (bba, bb) &: (z_1, z_3) \in M \\ T = (abba, abbab) &: (z_2, z_3) \in M \\ T = (aba, bab) &: (z_1, z_3) \in M \\ T = (abaaba, bbb) &: (z_1, z_4) \in M \\ T = (ba, bbaaabbab) &: (z_4, z_3) \in M \end{aligned}$$

- Betrachte (z_2, z_4):

$$\begin{aligned} T = (bba, bb) &: (z_1, z_4) \in M \\ T = (abba, abbab) &: (z_1, z_3) \in M \\ T = (aba, bab) &: (z_4, z_3) \in M \\ T = (abaaba, bbb) &: (z_1, z_3) \in M \\ T = (ba, bbaaabbab) &: (z_4, z_3) \in M \end{aligned}$$

Es ist damit: $M = \{(z_4, z_3), (z_2, z_3), (z_1, z_3), (z_1, z_4), (z_2, z_0), (z_1, z_0), (z_4, z_0), (z_2, z_4)\}$

Das Verfahren gibt also „falsch“ zurück.

b) Wenn V auf einer PCP-Instanz K gelaufen ist, was kann man dann aus einer „wahr“- und was aus einer „falsch“-Antwort für die PCP-Instanz schlussfolgern?

Bestimmen Sie mit Begründung getrennt für den „wahr“- und „falsch“-Fall die richtige Antwortkategorie:

- K hat sicher eine Lösung
- K hat sicher keine Lösung
- Man kann damit noch nicht sicher sagen, ob K eine Lösung hat

LÖSUNGSVORSCHLAG:

Das Verfahren V betrachtet für alle Folgen von Indizes $i_1, \ldots, i_n$, welche Zustände von $x_{i_1} \cdots x_{i_n}$ und $y_{i_1} \cdots y_{i_n}$ in A erreicht werden. Dabei wird „wahr“ zurückgegeben, wenn es eine Folge $i_1, \ldots, i_n$ gibt, bei der $x_{i_1} \cdots x_{i_n}$ und $y_{i_1} \cdots y_{i_n}$ den gleichen Zustand erreichen. Andernfalls wird „falsch“ zurückgegeben.

„falsch“ bedeutet deshalb, dass K sicher keine Lösung hat:
Wenn $x_{i_1} \cdots x_{i_n}$ nie den gleichen Zustand wie $y_{i_1} \cdots y_{i_n}$ erreicht, so gilt sicherlich $x_{i_1} \cdots x_{i_n} \neq y_{i_1} \cdots y_{i_n}$ für alle Folgen von Indizes $i_1, \ldots, i_n$.

„wahr“ bedeutet dagegen, dass man noch nicht sicher sagen kann, ob K eine Lösung hat: Hierfür kann z. B. ein Automaten mit nur einem Zustand und eine unlösbare PCP-Instanz betrachtet werden. Es ist leicht zu sehen, dass der Algorithmus hier „wahr“ zurückgibt, obwohl die Instanz nicht lösbar ist. Der Algorithmus gibt aber auch bei einer lösbaren PCP-Instanz „wahr“ zurück.

c) Zeigen Sie, dass es PCP-Instanzen K gibt, die keine Lösung haben, bei denen es aber dennoch keinen Automaten A gibt, sodass $V(K, A)$ mit „falsch“ antwortet.

Hinweis: Die Unentscheidbarkeit von PCP kann Ihnen beim Beweis helfen

LÖSUNGSVORSCHLAG:

Wir führen den Beweis durch Widerspruch und nehmen hierfür an, dass es für jede PCP-Instanz einen Automaten gibt, sodass der Algorithmus mit „falsch“ antwortet, falls die Instanz keine Lösung hat.

Wir können nun sowohl alle Automaten, wie auch alle PCP-Instanzen rekursiv aufzählen. Da der Algorithmus berechenbar ist, könnten wir so die Menge aller PCP-Instanzen ohne Lösung rekursiv aufzählen.

Es wäre somit das Komplement von PCP rekursiv aufzählbar, also semi-entscheidbar. Zusammen damit, dass, wie aus der Vorlesung bekannt, PCP semi-entscheidbar ist, wäre also PCP entscheidbar. Dies steht im Widerspruch dazu,

dass PCP, wie aus der Vorlesung bekannt, durch Reduktion des Halteproblems auf PCP, unentscheidbar ist.