Script generated by TTT

Title: seidl: Theoretische_Informatik (25.06.2012)

Date: Mon Jun 25 10:22:02 CEST 2012

Duration: 82:58 min

Pages: 112

Lemma 4.70
Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:
Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:
Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

input(x);
for $i := 0, 1, 2, \ldots$ **do**
  if $f(i) = x$ **then** output(1); **halt** **fi**

„$\Leftarrow$“: Wir nehmen an $A \subseteq \mathbb{N}$.
Sei A semi-entscheidbar durch (zB) GOTO-Programm P.
Problem: $P[i]$ muss nicht halten und darf daher nur „zeitbeschränkt“ ausgeführt weden.
Gesucht: Paare (i, j) so dass $P[i]$ nach j Schritten hält.

Lemma 4.70
Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:
Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:
Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

input(x);
for $i := 0, 1, 2, \ldots$ **do**
  if $f(i) = x$ **then** output(1); **halt** **fi**

„$\Leftarrow$“: Wir nehmen an $A \subseteq \mathbb{N}$.
Sei A semi-entscheidbar durch (zB) GOTO-Programm P.
Problem: $P[i]$ muss nicht halten und darf daher nur „zeitbeschränkt“ ausgeführt weden.

Lemma 4.70
Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Lemma 4.70

Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:

Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

Lemma 4.70

Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:

Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:

Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

Lemma 4.70

Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:

Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:

Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

input(x);
for $i := 0, 1, 2, \ldots$ **do**
  if $f(i) = x$ **then** output(1); **halt** **fi**

Lemma 4.70

Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:

Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:

Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

input(x);
for $i := 0, 1, 2, \ldots$ **do**
  if $f(i) = x$ **then** output(1); **halt** **fi**

„$\Leftarrow$“: Wir nehmen an $A \subseteq \mathbb{N}$.

Lemma 4.70
Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:
Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:
Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

input(x);
for $i := 0, 1, 2, \ldots$ **do**
  if $f(i) = x$ **then** output(1); **halt** **fi**

„$\Leftarrow$“: Wir nehmen an $A \subseteq \mathbb{N}$.
Sei A semi-entscheidbar durch (zB) GOTO-Programm P.

Lemma 4.70
Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:
Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:
Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

input(x);
for $i := 0, 1, 2, \ldots$ **do**
  if $f(i) = x$ **then** output(1); **halt** **fi**

„$\Leftarrow$“: Wir nehmen an $A \subseteq \mathbb{N}$.
Sei A semi-entscheidbar durch (zB) GOTO-Programm P.
Problem: $P[i]$ muss nicht halten und darf daher nur „zeitbeschränkt“ ausgeführt weden.

Lemma 4.70
Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:
Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:
Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

input(x);
for $i := 0, 1, 2, \ldots$ **do**
  if $f(i) = x$ **then** output(1); **halt** **fi**

„$\Leftarrow$“: Wir nehmen an $A \subseteq \mathbb{N}$.
Sei A semi-entscheidbar durch (zB) GOTO-Programm P.
Problem: $P[i]$ muss nicht halten und darf daher nur „zeitbeschränkt“ ausgeführt weden.
Gesucht: Paare (i, j) so dass $P[i]$ nach j Schritten hält.

Lemma 4.70
Eine Menge A ist rekursiv aufzählbar gdw sie semi-entscheidbar ist.

Beweis:
Der Fall $A = \emptyset$ ist trivial. Sei $A \neq \emptyset$.

„$\Rightarrow$“:
Sei A rekursive aufzählbar mit f. Dann ist A semi-entscheidbar:

input(x);
for $i := 0, 1, 2, \ldots$ **do**
  if $f(i) = x$ **then** output(1); **halt** **fi**

„$\Leftarrow$“: Wir nehmen an $A \subseteq \mathbb{N}$.
Sei A semi-entscheidbar durch (zB) GOTO-Programm P.
Problem: $P[i]$ muss nicht halten und darf daher nur „zeitbeschränkt“ ausgeführt weden.
Gesucht: Paare (i, j) so dass $P[i]$ nach j Schritten hält.
Idee: Bijektion $\mathbb{N} \to \mathbb{N} \times \mathbb{N}$, dh Umkehrung von c.

Beweis (Forts.):

Sei $d \in A$ beliebig.

Beweis (Forts.):

Sei $d \in A$ beliebig.
Folgender Algorithmus berechnet eine Aufzählung von A:

input(n);

Beweis (Forts.):

Sei $d \in A$ beliebig.
Folgender Algorithmus berechnet eine Aufzählung von A:

input(n);
if $P[p_1(n)]$ hält nach $p_2(n)$ Schritten **then** output($p_1(n)$)
else output(d) **fi**

Beweis (Forts.):

Sei $d \in A$ beliebig.
Folgender Algorithmus berechnet eine Aufzählung von A:

input(n);
if $P[p_1(n)]$ hält nach $p_2(n)$ Schritten **then** output($p_1(n)$)
else output(d) **fi**

Korrektheit:
Der Algorithmus hält immer und liefert immer ein Element aus A.

Folgende Aussagen sind äquivalent:

- A ist semi-entscheidbar
- A ist rekursiv aufzählbar
- χ'_A ist berechenbar

Folgende Aussagen sind äquivalent:

- A ist semi-entscheidbar
- A ist rekursiv aufzählbar
- χ'_A ist berechenbar
- $A = L(M)$ für eine TM M

Folgende Aussagen sind äquivalent:

- A ist semi-entscheidbar
- A ist rekursiv aufzählbar
- χ'_A ist berechenbar
- $A = L(M)$ für eine TM M
- A ist Definitionsbereich einer berechenbaren Funktion

Folgende Aussagen sind äquivalent:

- A ist semi-entscheidbar
- A ist rekursiv aufzählbar
- χ'_A ist berechenbar
- $A = L(M)$ für eine TM M
- A ist Definitionsbereich einer berechenbaren Funktion
- A ist Wertebereich einer berechenbaren Funktion

Folgende Aussagen sind äquivalent:

- A ist semi-entscheidbar
- A ist rekursiv aufzählbar
- χ'_A ist berechenbar
- $A = L(M)$ für eine TM M
- A ist Definitionsbereich einer berechenbaren Funktion
- A ist Wertebereich einer berechenbaren Funktion

Satz 4.71
Die Menge $K = \{w \mid M_w[w]\downarrow\}$ ist semi-entscheidbar.

Satz 4.71
Die Menge $K = \{w \mid M_w[w]\downarrow\}$ ist semi-entscheidbar.

Beweis:
Die Funktion χ'_K ist wie folgt Turing-berechenbar:

Satz 4.71
Die Menge $K = \{w \mid M_w[w]\downarrow\}$ ist semi-entscheidbar.

Beweis:
Die Funktion χ'_K ist wie folgt Turing-berechenbar:

Bei Eingabe w simuliere die Ausführung von $M_w[w]$;
gib 1 aus. □

Satz 4.71
Die Menge $K = \{w \mid M_w[w]\downarrow\}$ ist semi-entscheidbar.

Beweis:
Die Funktion χ'_K ist wie folgt Turing-berechenbar:

Bei Eingabe w simuliere die Ausführung von $M_w[w]$;
gib 1 aus. □

- Hier haben wir benutzt, dass man einen Interpreter/Simulator für Turingmaschinen als Turingmaschine programmieren kann.

Satz 4.71
Die Menge $K = \{w \mid M_w[w]\downarrow\}$ ist semi-entscheidbar.

Beweis:
Die Funktion χ'_K ist wie folgt Turing-berechenbar:

Bei Eingabe w simuliere die Ausführung von $M_w[w]$;
gib 1 aus. □

- Hier haben wir benutzt, dass man einen Interpreter/Simulator für Turingmaschinen als Turingmaschine programmieren kann.
- Ein solcher Interpreter wird oft eine Universelle Turingmaschine (U) genannt.

Satz 4.71
Die Menge $K = \{w \mid M_w[w]\downarrow\}$ ist semi-entscheidbar.

Beweis:
Die Funktion χ'_K ist wie folgt Turing-berechenbar:

Bei Eingabe w simuliere die Ausführung von $M_w[w]$;
gib 1 aus. □

- Hier haben wir benutzt, dass man einen Interpreter/Simulator für Turingmaschinen als Turingmaschine programmieren kann.
- Ein solcher Interpreter wird oft eine Universelle Turingmaschine (U) genannt.

Korollar 4.72
$\overline{K}$ ist nicht semi-entscheidbar.

Satz 4.71
Die Menge $K = \{w \mid M_w[w]\downarrow\}$ ist semi-entscheidbar.

Beweis:
Die Funktion χ'_K ist wie folgt Turing-berechenbar:

Bei Eingabe w simuliere die Ausführung von $M_w[w]$;
gib 1 aus. □

- Hier haben wir benutzt, dass man einen Interpreter/Simulator für Turingmaschinen als Turingmaschine programmieren kann.
- Ein solcher Interpreter wird oft eine Universelle Turingmaschine (U) genannt.

Korollar 4.72
$\overline{K}$ ist nicht semi-entscheidbar.

Semi-Entscheidbarkeit ist nicht abgeschlossen unter Komplement.

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.
Wir betrachten implizit nur einstellige Funktionen.

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.
Wir betrachten implizit nur einstellige Funktionen.

Satz 4.73 (Rice)

Sei F eine Menge berechenbarer Funktionen.

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.
Wir betrachten implizit nur einstellige Funktionen.

Satz 4.73 (Rice)

Sei F eine Menge berechenbarer Funktionen.
Es gelte weder $F = \emptyset$ noch $F =$ alle ber. Funkt.

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.
Wir betrachten implizit nur einstellige Funktionen.

Satz 4.73 (Rice)

Sei F eine Menge berechenbarer Funktionen.
Es gelte weder $F = \emptyset$ noch $F =$ alle ber. Funkt. („F nicht trivial“)
Dann ist unentscheidbar, ob die von einer gegebenen TM M_w
berechnete Funktion Element F ist

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.
Wir betrachten implizit nur einstellige Funktionen.

Satz 4.73 (Rice)

Sei F eine Menge berechenbarer Funktionen.
Es gelte weder $F = \emptyset$ noch $F =$ alle ber. Funkt. („F nicht trivial“)
Dann ist unentscheidbar, ob die von einer gegebenen TM M_w
berechnete Funktion Element F ist, dh ob $\varphi_w \in F$.

Nicht-triviale semantische Eigenschaften von Programmen sind unentscheidbar.

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.
Wir betrachten implizit nur einstellige Funktionen.

Satz 4.73 (Rice)

Sei F eine Menge berechenbarer Funktionen.
Es gelte weder $F = \emptyset$ noch $F =$ alle ber. Funkt. („F nicht trivial“)
Dann ist unentscheidbar, ob die von einer gegebenen TM M_w
berechnete Funktion Element F ist, dh ob $\varphi_w \in F$.

Nicht-triviale semantische Eigenschaften von Programmen sind unentscheidbar.

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.
Wir betrachten implizit nur einstellige Funktionen.

Satz 4.73 (Rice)

Sei F eine Menge berechenbarer Funktionen.
Es gelte weder $F = \emptyset$ noch $F =$ alle ber. Funkt. („F nicht trivial“)
Dann ist unentscheidbar, ob die von einer gegebenen TM M_w berechnete Funktion Element F ist, dh ob $\varphi_w \in F$.

Nicht-triviale semantische Eigenschaften von Programmen sind unentscheidbar.

Beispiel 4.74

Es ist unentscheidbar ob ein Programm

- irgendwo hält. $(F = \{\varphi_w \mid \exists x.\ M_w[x]\downarrow\})$

4.11 Die Sätze von Rice und Shapiro

Die von der TM M_w berechnete Funktion bezeichnen wir mit φ_w.
Wir betrachten implizit nur einstellige Funktionen.

Satz 4.73 (Rice)

Sei F eine Menge berechenbarer Funktionen.
Es gelte weder $F = \emptyset$ noch $F =$ alle ber. Funkt. („F nicht trivial“)
Dann ist unentscheidbar, ob die von einer gegebenen TM M_w berechnete Funktion Element F ist, dh ob $\varphi_w \in F$.

Nicht-triviale semantische Eigenschaften von Programmen sind unentscheidbar.

Beispiel 4.74

Es ist unentscheidbar ob ein Programm

- irgendwo hält. $(F = \{\varphi_w \mid \exists x.\ M_w[x]\downarrow\})$
- überall hält. $(F = \{\varphi_w \mid \forall x.\ M_w[x]\downarrow\})$
- bei Eingabe 42 Ausgabe 42 produziert.

Warnung

Es ist entscheidbar ob ein Programm

- länger als 5 Zeilen ist.
- eine Zuweisung an die Variable x_{17} enhält.

Warnung

Es ist entscheidbar ob ein Programm

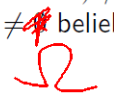
- länger als 5 Zeilen ist.
- eine Zuweisung an die Variable x_{17} enhält.

Im Satz von Rice geht es um die von einem Programm berechnete Funktion (Semantik),

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig;



Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$:

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$:
$f(w)$ ist die Kodierung folgender TM:

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$:
$f(w)$ ist die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;
schreibe $w\#w$ auf die Eingabe und führe U aus;
führe M_u auf x aus.

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$:
$f(w)$ ist die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;
schreibe $w\#w$ auf die Eingabe und führe U aus;
führe M_u auf x aus.

Damit gilt $\varphi_{f(w)} = \begin{cases} h & \text{falls } M_w[w]\downarrow \\ \Omega & \text{sonst} \end{cases}$

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$:
$f(w)$ ist die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;
schreibe $w\#w$ auf die Eingabe und führe U aus;
führe M_u auf x aus.

Damit gilt $\varphi_{f(w)} = \begin{cases} h & \text{falls } M_w[w]\downarrow \\ \Omega & \text{sonst} \end{cases}$ und damit

$$w \in K \;\Leftrightarrow\; M_w[w]\downarrow \overset{(*)}{\Leftrightarrow} \varphi_{f(w)} \in F \;\Leftrightarrow\; f(w) \in C_F$$

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$:
$f(w)$ ist die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;
schreibe $w\#w$ auf die Eingabe und führe U aus;
führe M_u auf x aus.

Damit gilt $\varphi_{f(w)} = \begin{cases} h & \text{falls } M_w[w]\downarrow \\ \Omega & \text{sonst} \end{cases}$ und damit

$$w \in K \;\Leftrightarrow\; M_w[w]\downarrow \overset{(*)}{\Leftrightarrow} \varphi_{f(w)} \in F \;\Leftrightarrow\; f(w) \in C_F$$

$$(*) : \begin{cases} M_w[w]\downarrow \Rightarrow \varphi_{f(w)} = h \in F \\ \varphi_{f(w)} \in F \Rightarrow \varphi_{f(w)} = h \Rightarrow M_w[w]\downarrow \end{cases}$$

Beweis (Forts.):
Fall 2: $\Omega \in F$.
Wähle berechenbares $h \notin F$.
Zeige analog, dass $\overline{K} \leq C_F$. □

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$:
$f(w)$ ist die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;
schreibe $w\#w$ auf die Eingabe und führe U aus;
führe M_u auf x aus.

Damit gilt $\varphi_{f(w)} = \begin{cases} h & \text{falls } M_w[w]{\downarrow} \\ \Omega & \text{sonst} \end{cases}$ und damit

$$w \in K \;\Leftrightarrow\; M_w[w]{\downarrow} \overset{(*)}{\Leftrightarrow} \varphi_{f(w)} \in F \;\Leftrightarrow\; f(w) \in C_F$$

$$(*) : \begin{cases} M_w[w]{\downarrow} \Rightarrow \varphi_{f(w)} = h \in F \\ \varphi_{f(w)} \in F \Rightarrow \varphi_{f(w)} = h \Rightarrow \end{cases}$$

Beweis:
Wir zeigen $C_F := \{w \in \{0,1\}^* \mid \varphi_w \in F\}$ ist unentscheidbar.
Fall 1: $\Omega := (x \mapsto \bot) \notin F$.
Wähle $h \in F \neq \emptyset$ beliebig; sei u Kodierung einer TM mit $\varphi_u = h$.
Reduziere K auf C_F ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$:
$f(w)$ ist die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;
schreibe $w\#w$ auf die Eingabe und führe U aus;
führe M_u auf x aus.

Damit gilt $\varphi_{f(w)} = \begin{cases} h & \text{falls } M_w[w]{\downarrow} \\ \Omega & \text{sonst} \end{cases}$

Beweis (Forts.):
Fall 2: $\Omega \in F$.
Wähle berechenbares $h \notin F$.
Zeige analog, dass $\overline{K} \leq C_F$. □

Satz 4.75 (Rice-Shapiro)
Sei F eine Menge berechenbarer Funktionen.

Satz 4.75 (Rice-Shapiro)

Sei F eine Menge berechenbarer Funktionen.
Ist $C_F := \{w \mid \varphi_w \in F\}$ semi-entscheidbar,

Satz 4.75 (Rice-Shapiro)

Sei F eine Menge berechenbarer Funktionen.
Ist $C_F := \{w \mid \varphi_w \in F\}$ semi-entscheidbar,
so gilt für alle berechenbaren f:
$f \in F \Leftrightarrow$ es gibt eine endlich Teilfunktion $g \subseteq f$ mit $g \in F$.

Satz 4.75 (Rice-Shapiro)

Sei F eine Menge berechenbarer Funktionen.
Ist $C_F := \{w \mid \varphi_w \in F\}$ semi-entscheidbar,
so gilt für alle berechenbaren f:
$f \in F \Leftrightarrow$ es gibt eine endlich Teilfunktion $g \subseteq f$ mit $g \in F$.

Beweis:
„$\Rightarrow$“ mit Widerspruch.

Satz 4.75 (Rice-Shapiro)

Sei F eine Menge berechenbarer Funktionen.
Ist $C_F := \{w \mid \varphi_w \in F\}$ semi-entscheidbar,
so gilt für alle berechenbaren f:
$f \in F \Leftrightarrow$ es gibt eine endlich Teilfunktion $g \subseteq f$ mit $g \in F$.

Beweis:
„$\Rightarrow$“ mit Widerspruch.
Sei $f \in F$, so dass für alle endlichen $g \subseteq f$ gilt $g \notin F$.

Satz 4.75 (Rice-Shapiro)

Sei F eine Menge berechenbarer Funktionen.
Ist $C_F := \{w \mid \varphi_w \in F\}$ semi-entscheidbar,
so gilt für alle berechenbaren f:
$f \in F \Leftrightarrow$ es gibt eine endlich Teilfunktion $g \subseteq f$ mit $g \in F$.

Beweis:
„$\Rightarrow$“ mit Widerspruch.
Sei $f \in F$, so dass für alle endlichen $g \subseteq f$ gilt $g \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist.

Satz 4.75 (Rice-Shapiro)

Sei F eine Menge berechenbarer Funktionen.
Ist $C_F := \{w \mid \varphi_w \in F\}$ semi-entscheidbar,
so gilt für alle berechenbaren f:
$f \in F \Leftrightarrow$ es gibt eine endlich Teilfunktion $g \subseteq f$ mit $g \in F$.

Beweis:
„$\Rightarrow$“ mit Widerspruch.
Sei $f \in F$, so dass für alle endlichen $g \subseteq f$ gilt $g \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. ↯

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t simuliere t Schritte von $M_w[w]$.
Hält diese Berechnung in $\leq t$ Schritten, gehe in eine endlos Schleife, sonst berechne $f(t)$.

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t simuliere t Schritte von $M_w[w]$.
Hält diese Berechnung in $\leq t$ Schritten, gehe in eine endlos Schleife, sonst berechne $f(t)$.

Wir zeigen

$$w \in \overline{K} \Leftrightarrow h(w) \in C_F$$

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t simuliere t Schritte von $M_w[w]$.
Hält diese Berechnung in $\leq t$ Schritten, gehe in eine endlos Schleife, sonst berechne $f(t)$.

Wir zeigen

$$w \in \overline{K} \Leftrightarrow h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]\downarrow$

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t simuliere t Schritte von $M_w[w]$.
Hält diese Berechnung in $\leq t$ Schritten, gehe in eine endlos Schleife, sonst berechne $f(t)$.

Wir zeigen

$$w \in \overline{K} \Leftrightarrow h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]\downarrow \implies \varphi_{h(w)} = f$

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t simuliere t Schritte von $M_w[w]$.
Hält diese Berechnung in $\leq t$ Schritten, gehe in eine endlos Schleife, sonst berechne $f(t)$.

Wir zeigen

$$w \in \overline{K} \Leftrightarrow h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]\downarrow \implies \varphi_{h(w)} = f \in F \implies h(w) \in C_F$
- Falls $w \notin \overline{K}$ dann hält $M_w[w]$ nach t Schritten.

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t simuliere t Schritte von $M_w[w]$.
Hält diese Berechnung in $\leq t$ Schritten, gehe in eine endlos Schleife, sonst berechne $f(t)$.

Wir zeigen

$$w \in \overline{K} \Leftrightarrow h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]\downarrow \implies \varphi_{h(w)} = f \in F \implies h(w) \in C_F$

- Falls $w \notin \overline{K}$ dann hält $M_w[w]$ nach t Schritten.
  Damit gilt: $\varphi_{h(w)}$ ist f eingeschränkt auf $\{0, \ldots, t-1\}$.

Beweis (Forts.):

Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t simuliere t Schritte von $M_w[w]$.
Hält diese Berechnung in $\leq t$ Schritten, gehe in eine endlos Schleife, sonst berechne $f(t)$.

Wir zeigen

$$w \in \overline{K} \Leftrightarrow h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]\downarrow \implies \varphi_{h(w)} = f \in F \implies h(w) \in C_F$
- Falls $w \notin \overline{K}$ dann hält $M_w[w]$ nach t Schritten.
  Damit gilt: $\varphi_{h(w)}$ ist f eingeschränkt auf $\{0, \ldots, t-1\}$.

  Nach Annahme folgt $\varphi_{h(w)} \notin F$, dh $h(w) \notin C_F$.

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist.

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. ↯
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. ↯
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t, teste ob t im *endlichen* Def.ber. von g ist.
Wenn ja, berechne $f(t)$,
sonst simuliere $M_w[w]$ und berechne dann $f(t)$.

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. $\lightning$
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t, teste ob t im *endlichen* Def.ber. von g ist.
Wenn ja, berechne $f(t)$,
sonst simuliere $M_w[w]$ und berechne dann $f(t)$.

Wir zeigen

$$w \in \overline{K} \;\Leftrightarrow\; h(w) \in C_F$$

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. $\lightning$
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t, teste ob t im *endlichen* Def.ber. von g ist.
Wenn ja, berechne $f(t)$,
sonst simuliere $M_w[w]$ und berechne dann $f(t)$.

Wir zeigen

$$w \in \overline{K} \;\Leftrightarrow\; h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]{\downarrow}$

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. $\lightning$
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t, teste ob t im *endlichen* Def.ber. von g ist.
Wenn ja, berechne $f(t)$,
sonst simuliere $M_w[w]$ und berechne dann $f(t)$.

Wir zeigen

$$w \in \overline{K} \;\Leftrightarrow\; h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]{\downarrow} \implies \varphi_{h(w)} = g$

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. $\lightning$
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t, teste ob t im *endlichen* Def.ber. von g ist.
Wenn ja, berechne $f(t)$,
sonst simuliere $M_w[w]$ und berechne dann $f(t)$.

Wir zeigen

$$w \in \overline{K} \;\Leftrightarrow\; h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]{\downarrow} \implies \varphi_{h(w)} = g \in F \implies h(w) \in C_F$

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. $\lightning$
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t, teste ob t im *endlichen* Def.ber. von g ist.
Wenn ja, berechne $f(t)$,
sonst simuliere $M_w[w]$ und berechne dann $f(t)$.

Wir zeigen

$$w \in \overline{K} \;\Leftrightarrow\; h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]\!\downarrow \implies \varphi_{h(w)} = g \in F \implies h(w) \in C_F$
- $w \notin \overline{K} \implies M_w[w]\!\downarrow$

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. $\lightning$
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t, teste ob t im *endlichen* Def.ber. von g ist.
Wenn ja, berechne $f(t)$,
sonst simuliere $M_w[w]$ und berechne dann $f(t)$.

Wir zeigen

$$w \in \overline{K} \;\Leftrightarrow\; h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]\!\downarrow \implies \varphi_{h(w)} = g \in F \implies h(w) \in C_F$
- $w \notin \overline{K} \implies M_w[w]\!\downarrow \implies \varphi_{h(w)} = f$

Beweis (Forts.):

„$\Leftarrow$“ mit Widerspruch.
Sei f berechenbar, sei $g \subseteq f$ endlich mit $g \in F$, aber sei $f \notin F$.
Wir zeigen $\overline{K} \leq C_F$ womit C_F nicht semi-entscheidbar ist. $\lightning$
Reduktion $\overline{K} \leq C_F$ mit $h : \{0,1\}^* \to \{0,1\}^*$:
$h(w)$ ist die Kodierung folgender TM:

Bei Eingabe t, teste ob t im *endlichen* Def.ber. von g ist.
Wenn ja, berechne $f(t)$,
sonst simuliere $M_w[w]$ und berechne dann $f(t)$.

Wir zeigen

$$w \in \overline{K} \;\Leftrightarrow\; h(w) \in C_F$$

- $w \in \overline{K} \implies \neg M_w[w]\!\downarrow \implies \varphi_{h(w)} = g \in F \implies h(w) \in C_F$
- $w \notin \overline{K} \implies M_w[w]\!\downarrow \implies \varphi_{h(w)} = f \notin F \implies h(w) \notin C_F$

□

Rice-Shapiro (in Kurzform): $C_F := \{w \mid \varphi_w \in F\}$ s-e $\implies$
$f \in F \Leftrightarrow$ es gibt endliche Funkt. $g \subseteq f$ mit $g \in F$.

Rice-Shapiro (in Kurzform): $C_F := \{w \mid \varphi_w \in F\}$ s-e $\Longrightarrow$

$f \in F \Leftrightarrow$ es gibt endliche Funkt. $g \subseteq f$ mit $g \in F$.

Ein Programm heißt terminierend gdw es für alle Eingaben hält.

Rice-Shapiro (in Kurzform): $C_F := \{w \mid \varphi_w \in F\}$ s-e $\Longrightarrow$

$f \in F \Leftrightarrow$ es gibt endliche Funkt. $g \subseteq f$ mit $g \in F$.

Ein Programm heißt terminierend gdw es für alle Eingaben hält.

Korollar 4.76

- *Die Menge der terminierenden Programme ist nicht semi-entscheidbar.*

Rice-Shapiro (in Kurzform): $C_F := \{w \mid \varphi_w \in F\}$ s-e $\Longrightarrow$

$f \in F \Leftrightarrow$ es gibt endliche Funkt. $g \subseteq f$ mit $g \in F$.

Ein Programm heißt terminierend gdw es für alle Eingaben hält.

Korollar 4.76

- *Die Menge der terminierenden Programme ist nicht semi-entscheidbar.*
- *Die Menge der nicht-terminierenden Programme ist nicht semi-entscheidbar.*

Rice-Shapiro (in Kurzform): $C_F := \{w \mid \varphi_w \in F\}$ s-e $\Longrightarrow$

$f \in F \Leftrightarrow$ es gibt endliche Funkt. $g \subseteq f$ mit $g \in F$.

Ein Programm heißt terminierend gdw es für alle Eingaben hält.

Korollar 4.76

- *Die Menge der terminierenden Programme ist nicht semi-entscheidbar.*
- *Die Menge der nicht-terminierenden Programme ist nicht semi-entscheidbar.*

Beweis:

- $F :=$ Menge aller berechenbaren totalen Funktionen.

Rice-Shapiro (in Kurzform): $C_F := \{w \mid \varphi_w \in F\}$ s-e $\Longrightarrow$

$f \in F \Leftrightarrow$ es gibt endliche Funkt. $g \subseteq f$ mit $g \in F$.

Ein Programm heißt terminierend gdw es für alle Eingaben hält.

Korollar 4.76

- *Die Menge der terminierenden Programme ist nicht semi-entscheidbar.*
- *Die Menge der nicht-terminierenden Programme ist nicht semi-entscheidbar.*

Beweis:

- $F :=$ Menge aller berechenbaren totalen Funktionen.
  Sei $f \in F$. Jede endliche $g \subseteq f$ ist echt partiell, dh $g \notin F$.

Rice-Shapiro (in Kurzform): $C_F := \{w \mid \varphi_w \in F\}$ s-e $\Longrightarrow$

$f \in F \Leftrightarrow$ es gibt endliche Funkt. $g \subseteq f$ mit $g \in F$.

Ein Programm heißt terminierend gdw es für alle Eingaben hält.

Korollar 4.76

- *Die Menge der terminierenden Programme ist nicht semi-entscheidbar.*
- *Die Menge der nicht-terminierenden Programme ist nicht semi-entscheidbar.*

Beweis:

- $F :=$ Menge aller berechenbaren totalen Funktionen.
  Sei $f \in F$. Jede endliche $g \subseteq f$ ist echt partiell, dh $g \notin F$.
  Also kann C_F nicht semi-entscheidbar sein.
- $F :=$ Menge aller berechenbaren nicht-totalen Funktionen.
  Sei f total und berechenbar. Damit $f \notin F$.

Rice-Shapiro (in Kurzform): $C_F := \{w \mid \varphi_w \in F\}$ s-e $\Longrightarrow$

$f \in F \Leftrightarrow$ es gibt endliche Funkt. $g \subseteq f$ mit $g \in F$.

Ein Programm heißt terminierend gdw es für alle Eingaben hält.

Korollar 4.76

- *Die Menge der terminierenden Programme ist nicht semi-entscheidbar.*
- *Die Menge der nicht-terminierenden Programme ist nicht semi-entscheidbar.*

Beweis:

- $F :=$ Menge aller berechenbaren totalen Funktionen.
  Sei $f \in F$. Jede endliche $g \subseteq f$ ist echt partiell, dh $g \notin F$.
  Also kann C_F nicht semi-entscheidbar sein.
- $F :=$ Menge aller berechenbaren nicht-totalen Funktionen.
  Sei f total und berechenbar. Damit $f \notin F$.
  Aber jede endliche $g \subseteq f$ ist in F.
  Also kann C_F nicht semi-entscheidbar sein. $\square$

Grenzen automatischer Terminationsanalyse von Programmen

- Termination ist unentscheidbar (Rice):

Grenzen automatischer Terminationsanalyse von Programmen

- Termination ist unentscheidbar (Rice):
  Klare Ja/Nein Antwort unmöglich.
- Termination ist nicht semi-entscheidbar (Rice-Shapiro):

Grenzen automatischer Terminationsanalyse von Programmen

- Termination ist unentscheidbar (Rice):
  Klare Ja/Nein Antwort unmöglich.
- Termination ist nicht semi-entscheidbar (Rice-Shapiro):
  Es gibt kein Zertifizierungs-Programm,
  das alle terminierenden Programme erkennt.
- Nicht-Termination ist nicht semi-entscheidbar (Rice-Shapiro):

Grenzen automatischer Terminationsanalyse von Programmen

- Termination ist unentscheidbar (Rice):
  Klare Ja/Nein Antwort unmöglich.
- Termination ist nicht semi-entscheidbar (Rice-Shapiro):
  Es gibt kein Zertifizierungs-Programm,
  das alle terminierenden Programme erkennt.
- Nicht-Termination ist nicht semi-entscheidbar (Rice-Shapiro):
  Es gibt keinen perfekten *Bug Finder*,
  der alle nicht-terminierenden Programme erkennt.

Grenzen automatischer Terminationsanalyse von Programmen

- Termination ist unentscheidbar (Rice):
  Klare Ja/Nein Antwort unmöglich.
- Termination ist nicht semi-entscheidbar (Rice-Shapiro):
  Es gibt kein Zertifizierungs-Programm,
  das alle terminierenden Programme erkennt.
- Nicht-Termination ist nicht semi-entscheidbar (Rice-Shapiro):
  Es gibt keinen perfekten *Bug Finder*,
  der alle nicht-terminierenden Programme erkennt.

Aber es gibt mächtige heuristische Verfahren, die für
relativ viele Programme aus der Praxis (Gerätetreiber)

- Termination beweisen können, oder
- Gegenbeispiele finden können.

4.12 Das Postsche Korrespondenzproblem

4.12 Das Postsche Korrespondenzproblem

Gegeben beliebig viele Kopien der 3 „Spielkarten“

001	10	0
00	11	010

4.12 Das Postsche Korrespondenzproblem

Gegeben beliebig viele Kopien der 3 „Spielkarten“

001	10	0
00	11	010

gibt es dann eine Folge dieser Karten

...	...	...
...		...

so dass oben und unten das gleiche Wort steht?

4.12 Das Postsche Korrespondenzproblem

Gegeben beliebig viele Kopien der 3 „Spielkarten“

001	10	0
00	11	010

gibt es dann eine Folge dieser Karten

...	...	...
...		...

so dass oben und unten das gleiche Wort steht?

001	10	001	0
00	11	00	010

Kurz: 1,2,1,3.

Definition 4.77 (Postsche Korrespondenzproblem, *Post's Correspondence Problem*, PCP)

Gegeben: Eine endliche Folge $(x_1, y_1), \ldots, (x_k, y_k)$, wobei $x_i, y_i \in \Sigma^+$.

Definition 4.77 (Postsche Korrespondenzproblem, *Post's Correspondence Problem*, PCP)

Gegeben: Eine endliche Folge $(x_1, y_1), \ldots, (x_k, y_k)$, wobei $x_i, y_i \in \Sigma^+$.

Problem: Gibt es eine Folge von Indizes $i_1, \ldots, i_n \in \{1, \ldots, k\}$, $n > 0$, mit $x_{i_1} \ldots x_{i_n} = y_{i_1} \ldots y_{i_n}$?

Definition 4.77 (Postsche Korrespondenzproblem, *Post's Correspondence Problem*, PCP)

Gegeben: Eine endliche Folge $(x_1, y_1), \ldots, (x_k, y_k)$, wobei $x_i, y_i \in \Sigma^+$.

Problem: Gibt es eine Folge von Indizes $i_1, \ldots, i_n \in \{1, \ldots, k\}$, $n > 0$, mit $x_{i_1} \ldots x_{i_n} = y_{i_1} \ldots y_{i_n}$?

Dann nennen wir $i_1, \ldots, i_n$ eine Lösung des Problems $(x_1, y_1), \ldots, (x_k, y_k)$.

Definition 4.77 (Postsche Korrespondenzproblem, *Post's Correspondence Problem*, PCP)

Gegeben: Eine endliche Folge $(x_1, y_1), \ldots, (x_k, y_k)$, wobei $x_i, y_i \in \Sigma^+$.

Problem: Gibt es eine Folge von Indizes $i_1, \ldots, i_n \in \{1, \ldots, k\}$, $n > 0$, mit $x_{i_1} \ldots x_{i_n} = y_{i_1} \ldots y_{i_n}$?

Dann nennen wir $i_1, \ldots, i_n$ eine Lösung des Problems $(x_1, y_1), \ldots, (x_k, y_k)$.

Beispiel 4.78

- Hat $(1, 111), (10111, 10), (10, 0)$ eine Lösung?

Definition 4.77 (Postsche Korrespondenzproblem, *Post's Correspondence Problem*, PCP)

Gegeben: Eine endliche Folge $(x_1, y_1), \ldots, (x_k, y_k)$, wobei $x_i, y_i \in \Sigma^+$.

Problem: Gibt es eine Folge von Indizes $i_1, \ldots, i_n \in \{1, \ldots, k\}$, $n > 0$, mit $x_{i_1} \ldots x_{i_n} = y_{i_1} \ldots y_{i_n}$?

Dann nennen wir $i_1, \ldots, i_n$ eine Lösung des Problems $(x_1, y_1), \ldots, (x_k, y_k)$.

Beispiel 4.78

- Hat $(1, 111), (10111, 10), (10, 0)$ eine Lösung? 2,1,1,3

Definition 4.77 (Postsche Korrespondenzproblem, *Post's Correspondence Problem*, PCP)

Gegeben: Eine endliche Folge $(x_1, y_1), \ldots, (x_k, y_k)$, wobei $x_i, y_i \in \Sigma^+$.

Problem: Gibt es eine Folge von Indizes $i_1, \ldots, i_n \in \{1, \ldots, k\}$, $n > 0$, mit $x_{i_1} \ldots x_{i_n} = y_{i_1} \ldots y_{i_n}$?

Dann nennen wir $i_1, \ldots, i_n$ eine Lösung des Problems $(x_1, y_1), \ldots, (x_k, y_k)$.

Beispiel 4.78

- Hat $(1, 111), (10111, 10), (10, 0)$ eine Lösung? 2,1,1,3

- Hat $(b, ca), (a, ab), (ca, a), (abc, c)$ eine Lösung? 2,1,3,2,4

Definition 4.77 (Postsche Korrespondenzproblem, *Post's Correspondence Problem*, PCP)

Gegeben: Eine endliche Folge $(x_1, y_1), \ldots, (x_k, y_k)$, wobei $x_i, y_i \in \Sigma^+$.

Problem: Gibt es eine Folge von Indizes $i_1, \ldots, i_n \in \{1, \ldots, k\}$, $n > 0$, mit $x_{i_1} \ldots x_{i_n} = y_{i_1} \ldots y_{i_n}$?

Dann nennen wir $i_1, \ldots, i_n$ eine Lösung des Problems $(x_1, y_1), \ldots, (x_k, y_k)$.

Beispiel 4.78

- Hat $(1, 111), (10111, 10), (10, 0)$ eine Lösung? 2,1,1,3
- Hat $(b, ca), (a, ab), (ca, a), (abc, c)$ eine Lösung? 2,1,3,2,4

- Hat $(101, 01), (101, 010), (010, 10)$ eine Lösung?

Emil Post.
A Variant of a Recursively Unsolvable Problem.
Bulletin American Mathematical Society, 1946.

Lemma 4.79
Das PCP ist semi-entscheidbar.

Lemma 4.79
Das PCP ist semi-entscheidbar.

Beweis:
Zähle die möglichen Lösungen der Länge nach auf, und probiere jeweils, ob es eine wirkliche Lösung ist. □

Lemma 4.79
Das PCP ist semi-entscheidbar.

Beweis:
Zähle die möglichen Lösungen der Länge nach auf, und probiere jeweils, ob es eine wirkliche Lösung ist. □

Wir zeigen nun:

$$H \leq MPCP \leq PCP$$

Lemma 4.79
Das PCP ist semi-entscheidbar.

Beweis:
Zähle die möglichen Lösungen der Länge nach auf, und probiere jeweils, ob es eine wirkliche Lösung ist. □

Wir zeigen nun:

$$H \leq MPCP \leq PCP$$

wobei

Definition 4.80 (Modifiziertes PCP, MPCP)

Gegeben: wie beim PCP
Problem: Gibt es eine Lösung $i_1, \ldots, i_n$ mit $i_1 = 1$?

Lemma 4.79

Das PCP ist semi-entscheidbar.

Beweis:

Zähle die möglichen Lösungen der Länge nach auf, und probiere jeweils, ob es eine wirkliche Lösung ist. □

Wir zeigen nun:

$$H \leq MPCP \leq PCP$$

wobei

Definition 4.80 (Modifiziertes PCP, MPCP)

Gegeben: wie beim PCP

Problem: Gibt es eine Lösung $i_1, \ldots, i_n$ mit $i_1 = 1$?

Satz 4.81

$$MPCP \leq PCP$$