Script generated by TTT

Title: Nipkow: Theo (17.06.2019)

Date: Mon Jun 17 14:16:50 CEST 2019

Duration: 88:52 min

Pages: 87

Definition 5.29

$M[w]$ ist Abk. für „Maschine M mit Eingabe w“

$M[w]\downarrow$ bedeutet, dass $M[w]$ terminiert/hält.

Definition 5.30 (Spezielles Halteproblem)

Gegeben: Ein Wort $w \in \{0,1\}^*$.

Problem: Hält M_w bei Eingabe w ?

Als Menge:

$$K := \{w \in \{0,1\}^* \mid M_w[w]\downarrow\}$$

Definition 5.32 ((Allgemeines) Halteproblem)

Gegeben: Wörter $w, x \in \{0,1\}^*$.

Problem: Hält M_w bei Eingabe x ?

Als Menge:

$$H := \{w\#x \mid M_w[x]\downarrow\}$$

Satz 5.33

Das Halteproblem H ist nicht entscheidbar.

Definition 5.34 (Reduktion)

Eine Menge $A \subseteq \Sigma^*$ ist reduzierbar auf eine Menge $B \subseteq \Gamma^*$ gdw es eine totale und berechenbare Funktion $f : \Sigma^* \to \Gamma^*$ gibt mit

$$\forall w \in \Sigma^*.\ w \in A \Leftrightarrow f(w) \in B$$

Wir schreiben dann $A \leq B$.

Definition 5.34 (Reduktion)

Eine Menge $A \subseteq \Sigma^*$ ist reduzierbar auf eine Menge $B \subseteq \Gamma^*$ gdw es eine totale und berechenbare Funktion $f : \Sigma^* \to \Gamma^*$ gibt mit

$$\forall w \in \Sigma^*.\ w \in A \Leftrightarrow f(w) \in B$$

Wir schreiben dann $A \leq B$.

Intuition:

- B ist mindestens so schwer zu lösen wie A.

Lemma 5.35

Falls $A \leq B$ und B ist entscheidbar, so ist auch A entscheidbar.

Beweis:
Es gelte $A \leq B$ mittels f und χ_B sei berechenbar.

Lemma 5.35

Falls $A \leq B$ und B ist entscheidbar, so ist auch A entscheidbar.

Beweis:
Es gelte $A \leq B$ mittels f und χ_B sei berechenbar.
Dann ist $\chi_B \circ f$ berechenbar und $\chi_A = \chi_B \circ f$:

$$\chi_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases} = \begin{cases} 1, & f(x) \in B \\ 0, & f(x) \notin B \end{cases} = \chi_B(f(x)) \qquad \square$$

Korollar 5.36

Falls $A \leq B$ und A ist unentscheidbar, dann ist auch B unentscheidbar.

Beispiel 5.37

Da $K \leq H$ (mit Reduktion $f(w) := w\#w$) und K unentscheidbar ist, ist auch H unentscheidbar.

Satz 5.38

Das Halteproblem auf leerem Band, H_0, ist unentscheidbar.

$$H_0 := \{w \in \{0,1\}^* \mid M_w[\epsilon]\downarrow\}$$

Beweis:
Wir zeigen $K \leq H_0$ mit einer Funktion $f : \{0,1\}^* \to \{0,1\}^*$

Quiz

Ist es entscheidbar, ob eine TM

2. bei Eingabe ϵ mehr als 314 Schritte macht?

Quiz

Ist es entscheidbar, ob eine TM

3. bei *irgendeiner* Eingabe mehr als 314 Schritte macht?

Quiz

Ist es entscheidbar, ob eine TM

4. bei *allen* Eingaben mehr als 314 Schritte macht?

Quiz

Ist es entscheidbar, ob eine TM

5. bei Eingabe ϵ ihren Kopf mehr als 314 Felder von der 0-Position entfernen kann?

Quiz

Ist es entscheidbar, ob eine TM

6. bei Eingabe ϵ einen bestimmten Zustand erreichen kann?

Quiz

Ist es entscheidbar, ob eine TM

7. *irgendeine* Eingabe akzeptieren kann?

$H_0 \leq Q_7 \quad w \in H_0 \iff f(w) \in Q_7$

$f(w) =$ Kodierung der TM „Leiche Eingabe, M_w"

Es müssen nicht immer TM sein:

Satz 5.39 (Matiyasevich 1970, Hilberts 10. Problem)

Es ist unentscheidbar, ob ein Polynom in n Variablen mit ganzzahligen Koeffizienten eine ganzzahlige Nullstelle hat ($\in \mathbb{Z}^n$).

Es müssen nicht immer TM sein:

Satz 5.39 (Matiyasevich 1970, Hilberts 10. Problem)

Es ist unentscheidbar, ob ein Polynom in n Variablen mit ganzzahligen Koeffizienten eine ganzzahlige Nullstelle hat ($\in \mathbb{Z}^n$).

Beweis:

$$H \leq H_{10}$$

□

Bemerkungen

Bemerkungen

- Nicht alle unentscheidbaren Probleme sind gleich schwer
- ZB gilt: Das Äquivalenzproblem

$$Eq := \{u\#v \mid M_u \text{ berechnet die gleiche Funktion wie } M_v\}$$

Bemerkungen

- Nicht alle unentscheidbaren Probleme sind gleich schwer
- ZB gilt: Das Äquivalenzproblem

$$Eq := \{u\#v \mid M_u \text{ berechnet die gleiche Funktion wie } M_v\}$$

ist schwerer als das Halteproblem:

$$H \leq Eq \qquad \text{aber} \qquad Eq \not\leq H$$

5.6 Semi-Entscheidbarkeit

5.6 Semi-Entscheidbarkeit

Definition 5.40
Eine Menge A ($\subseteq \mathbb{N}$ oder Σ^*) heißt semi-entscheidbar (s-e) gdw

$$\chi'_A(x) := \begin{cases} 1 & \text{falls } x \in A \\ \perp & \text{falls } x \notin A \end{cases}$$

berechenbar ist.

Satz 5.41
Eine Menge A ist entscheidbar gdw sowohl A als auch $\overline{A}$ s-e sind.

Satz 5.41
Eine Menge A ist entscheidbar gdw sowohl A als auch $\overline{A}$ s-e sind.

Satz 5.41
Eine Menge A ist entscheidbar gdw sowohl A als auch $\overline{A}$ s-e sind.

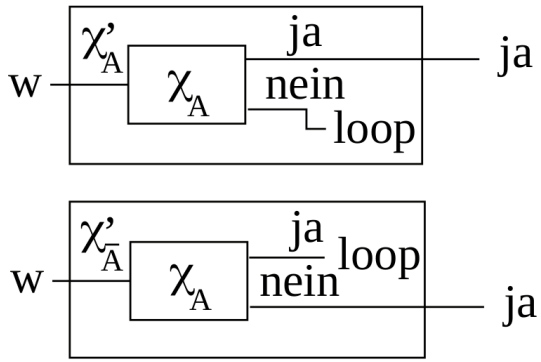
Beweis:
„$\Rightarrow$“: Wandle TM für χ_A in TM für χ'_A und $\chi'_{\overline{A}}$ um:

Satz 5.41

Eine Menge A ist entscheidbar gdw sowohl A als auch $\overline{A}$ s-e sind.

Beweis:

„$\Rightarrow$“: Wandle TM für χ_A in TM für χ'_A und $\chi'_{\overline{A}}$ um:



Beweis (Forts.):

„$\Leftarrow$“:
Wandle TM M_1 für χ'_A und TM M_2 für $\chi'_{\overline{A}}$ in TM für χ_A um:

Beweis (Forts.):

„$\Leftarrow$“:
Wandle TM M_1 für χ'_A und TM M_2 für $\chi'_{\overline{A}}$ in TM für χ_A um:

input(x);
for $s := 0, 1, 2, \ldots$ **do**
  if $M_1[x]$ hält in s Schritten **then** output(1); **halt fi** ;
  if $M_2[x]$ hält in s Schritten **then** output(0); **halt fi**

Beweis (Forts.):

„$\Leftarrow$“:
Wandle TM M_1 für χ'_A und TM M_2 für $\chi'_{\overline{A}}$ in TM für χ_A um:

input(x);
for $s := 0, 1, 2, \ldots$ **do**
  if $M_1[x]$ hält in s Schritten **then** output(1); **halt fi** ;
  if $M_2[x]$ hält in s Schritten **then** output(0); **halt fi**

Formulierung mit Parallelismus:

input(x);
führe $M_1[x]$ und $M_2[x]$ parallel aus;
hält M_1, gib 1 aus, hält M_2, gib 0 aus. □

Beweis (Forts.):

„⇐“:
Wandle TM M_1 für χ'_A und TM M_2 für $\chi'_{\overline{A}}$ in TM für χ_A um:

input(x);
for $s := 0, 1, 2, \ldots$ **do**
  if $M_1[x]$ hält in s Schritten **then** output(1); **halt** **fi** ;
  if $M_2[x]$ hält in s Schritten **then** output(0); **halt** **fi**

Formulierung mit Parallelismus:

input(x);
führe $M_1[x]$ und $M_2[x]$ parallel aus;
hält M_1, gib 1 aus, hält M_2, gib 0 aus. □

Lemma 5.42

Ist $A \leq B$ und ist B s-e, so ist auch A s-e.

Beweis: Übung

Definition 5.43

Eine Menge A heißt rekursiv aufzählbar (*recursively enumerable*) gdw $A = \emptyset$ oder es eine berechenbare totale Funktion $f : \mathbb{N} \to A$ gibt, so dass

$$A = \{f(0), f(1), f(2), \ldots\}$$

Definition 5.43

Eine Menge A heißt rekursiv aufzählbar (*recursively enumerable*) gdw $A = \emptyset$ oder es eine berechenbare totale Funktion $f : \mathbb{N} \to A$ gibt, so dass

$$A = \{f(0), f(1), f(2), \ldots\}$$

Bemerkung:

- Es dürfen Elemente doppelt auftreten ($f(i) = f(j)$ für $i \neq j$)
- Die Reihenfolge ist beliebig.

Definition 5.43

Eine Menge A heißt rekursiv aufzählbar (*recursively enumerable*) gdw $A = \emptyset$ oder es eine berechenbare totale Funktion $f : \mathbb{N} \to A$ gibt, so dass

$$A = \{f(0), f(1), f(2), \ldots\}$$

Bemerkung:

- Es dürfen Elemente doppelt auftreten ($f(i) = f(j)$ für $i \neq j$)
- Die Reihenfolge ist beliebig.

Warnung: Rekursiv aufzählbar $\neq$ abzählbar!

Definition 5.43

Eine Menge A heißt rekursiv aufzählbar (*recursively enumerable*) gdw $A = \emptyset$ oder es eine berechenbare totale Funktion $f : \mathbb{N} \to A$ gibt, so dass

$$A = \{f(0), f(1), f(2), \ldots\}$$

Bemerkung:

- Es dürfen Elemente doppelt auftreten ($f(i) = f(j)$ für $i \neq j$)
- Die Reihenfolge ist beliebig.

Warnung: Rekursiv aufzählbar $\neq$ abzählbar!

- Rekursiv aufzählbar $\Longrightarrow$ abzählbar

Definition 5.43

Eine Menge A heißt rekursiv aufzählbar (*recursively enumerable*) gdw $A = \emptyset$ oder es eine berechenbare totale Funktion $f : \mathbb{N} \to A$ gibt, so dass

$$A = \{f(0), f(1), f(2), \ldots\}$$

Bemerkung:

- Es dürfen Elemente doppelt auftreten ($f(i) = f(j)$ für $i \neq j$)
- Die Reihenfolge ist beliebig.

Warnung: Rekursiv aufzählbar $\neq$ abzählbar!

- Rekursiv aufzählbar $\Longrightarrow$ abzählbar

- Aber nicht umgekehrt:
  jede Sprache ist abzählbar aber nicht jede Sprache ist rekursiv aufzählbar (s.u.)

Lemma 5.44

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

Lemma 5.44

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

Beweis:

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

Lemma 5.44

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 rekursiv aufzählbar mit f. Dann ist A semi-entscheidbar:

Lemma 5.44

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 rekursiv 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 5.44

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 rekursiv 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**

„$\Leftarrow$“: O.B.d.A. nehmen wir $A \subseteq \mathbb{N}$ an.

Lemma 5.44

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 rekursiv 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**

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

Lemma 5.44

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 rekursiv 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**

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

Lemma 5.44

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 rekursiv 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**

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

Beweis (Forts.):

Idee: Wir benutzen eine geeignete Bijektion $c\colon \mathbb{N} \times \mathbb{N} \leftrightarrow \mathbb{N}$.

Cantorsche Bijektion

$c(x,y) = \frac{1}{2}(x+y)(x+y+1) + x$

Umkehrungen auch berechenbar!

Lemma 5.44

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 rekursiv 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**

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

Beweis (Forts.):

Idee: Wir benutzen eine geeignete Bijektion $c\colon \mathbb{N} \times \mathbb{N} \leftrightarrow \mathbb{N}$.

Beweis (Forts.):

Idee: Wir benutzen eine geeignete Bijektion $c\colon \mathbb{N} \times \mathbb{N} \leftrightarrow \mathbb{N}$.
Seien $p_1\colon \mathbb{N} \to \mathbb{N}$ und $p_2\colon \mathbb{N} \to \mathbb{N}$ mit

$$p_1(c(n_1, n_2)) = n_1 \qquad \text{und} \qquad p_2(c(n_1, n_2)) = n_2$$

(Umkehrung von c).

Sei $d \in A$ beliebig.

Beweis (Forts.):

Idee: Wir benutzen eine geeignete Bijektion $c\colon \mathbb{N} \times \mathbb{N} \leftrightarrow \mathbb{N}$.
Seien $p_1\colon \mathbb{N} \to \mathbb{N}$ und $p_2\colon \mathbb{N} \to \mathbb{N}$ mit

$$p_1(c(n_1, n_2)) = n_1 \qquad \text{und} \qquad p_2(c(n_1, n_2)) = n_2$$

(Umkehrung von c).

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.

Beweis (Forts.):

Idee: Wir benutzen eine geeignete Bijektion $c\colon \mathbb{N} \times \mathbb{N} \leftrightarrow \mathbb{N}$.
Seien $p_1\colon \mathbb{N} \to \mathbb{N}$ und $p_2\colon \mathbb{N} \to \mathbb{N}$ mit

$$p_1(c(n_1, n_2)) = n_1 \qquad \text{und} \qquad p_2(c(n_1, n_2)) = n_2$$

(Umkehrung von c).

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.

Vollständigkeit: Sei $a \in A \subseteq \mathbb{N}$.
Dann hält $P[a]$ nach einer endlichen Zahl k von Schritten. Dann liefert die Eingabe $n = c(a, k)$ die Ausgabe a. □

Folgende Aussagen sind äquivalent:

- A ist semi-entscheidbar

Folgende Aussagen sind äquivalent:

- A ist semi-entscheidbar
- A ist rekursiv aufzählbar

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

totale

Satz 5.45

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

Satz 5.45

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 5.45

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

Satz 5.45

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 5.45

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 5.46

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

Satz 5.45
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 5.46
$\overline{K}$ ist nicht semi-entscheidbar.

Semi-Entscheidbarkeit ist nicht abgeschlossen unter Komplement.

5.7 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.

5.7 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 5.47 (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 $\varphi_w \in F$.

5.7 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 5.47 (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 $\varphi_w \in F$.

5.7 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 5.47 (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 $\varphi_w \in F$.

Alle nicht-triviale semantische Eigenschaften von Programmen sind unentscheidbar.

Warnung

Es ist entscheidbar, ob ein Programm

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

Im Satz von Rice geht es um die von einem Programm berechnete Funktion (Semantik),
nicht um den Programmtext (Syntax).

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: $\Omega := (x \mapsto \bot) \notin F$.
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 ($K \leq C_F$) mit $f : \{0,1\}^* \to \{0,1\}^*$ und
$f(w)$ die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;

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 $f : \{0,1\}^* \to \{0,1\}^*$ und
$f(w)$ die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;
schreibe $w\#w$ auf die Eingabe und führe die universelle TM U 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 $f : \{0,1\}^* \to \{0,1\}^*$ und
$f(w)$ die Kodierung folgender TM:

Speichere die Eingabe x auf einem getrennten Band;
schreibe $w\#w$ auf die Eingabe und führe die universelle TM 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 $f : \{0,1\}^* \to \{0,1\}^*$ und
$f(w)$ die Kodierung folgender TM:

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

Es 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 $f : \{0,1\}^* \to \{0,1\}^*$ und
$f(w)$ die Kodierung folgender TM:

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

Es 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 $f : \{0,1\}^* \to \{0,1\}^*$ und
$f(w)$ die Kodierung folgender TM:

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

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

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

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

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 $f : \{0,1\}^* \to \{0,1\}^*$ und
$f(w)$ die Kodierung folgender TM:

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

Es 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 $\overline{K} \leq C_F$. □

Satz 5.49 (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 endliche 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.
Widerspruch

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 5.50

- *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 5.50

- *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 5.50

- *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 $f \in F$.

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 5.50

- *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 5.50

- *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.

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 5.50

- *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):
  Klare Ja/Nein Antwort unmöglich.

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.

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.