Theoretische Informatik
Kapitel 7 – Turing-Berechenbarkeit

Sommersemester 2024

Dozentin: Mareike Mutz

im Wechsel mit

Prof. Dr. M. Leuschel

Prof. Dr. J. Rothe

Turing-Berechenbarkeit

Bisher wurden Turingmaschinen als Akzeptoren definiert.

Nun wollen wir Turingmaschinen als Maschinen zur Berechnung von Funktionen auffassen.

Turing-Berechenbarkeit

Bisher wurden Turingmaschinen als Akzeptoren definiert.

Nun wollen wir Turingmaschinen als Maschinen zur Berechnung von Funktionen auffassen.

Definition

Eine Funktion $f : \mathbb{N}^k \to \mathbb{N}$ heißt *Turing-berechenbar*, falls es eine deterministische Turingmaschine $M = (\Sigma, \Gamma, Z, \delta, z_0, \Box, F)$ mit $\Sigma = \{0, 1\}$ für k=1 und $\Sigma = \{0, 1, \#\}$ für $k > 1$ gibt, so dass für alle $n_1, n_2, \ldots, n_k, m \in \mathbb{N}$:

$$f(n_1, n_2, \ldots, n_k) = m \iff$$

$$\exists z \in F. z_0 \mathrm{bin}(n_1)\#\mathrm{bin}(n_2)\# \cdots \#\mathrm{bin}(n_k) \vdash_M^* z\mathrm{bin}(m)$$

Turing-Berechenbarkeit

Bisher wurden Turingmaschinen als Akzeptoren definiert.

Nun wollen wir Turingmaschinen als Maschinen zur Berechnung von Funktionen auffassen.

Definition

Eine Funktion $f : \mathbb{N}^k \to \mathbb{N}$ heißt *Turing-berechenbar*, falls es eine deterministische Turingmaschine $M = (\Sigma, \Gamma, Z, \delta, z_0, \Box, F)$ mit $\Sigma = \{0, 1\}$ für k=1 und $\Sigma = \{0, 1, \#\}$ für $k > 1$ gibt, so dass für alle $n_1, n_2, \ldots, n_k, m \in \mathbb{N}$:

$$f(n_1, n_2, \ldots, n_k) = m \iff$$

$$\exists z \in F . z_0 \mathrm{bin}(n_1)\#\mathrm{bin}(n_2)\# \cdots \#\mathrm{bin}(n_k) \vdash_M^* z\mathrm{bin}(m)$$

Falls $f(n_1, n_2, \ldots, n_k)$ nicht definiert ist, läuft M in eine unendliche Schleife oder stoppt in einem Zustand $z \notin F$.

Turing-Berechenbarkeit: Nachfolgerfunktion

Beispiel: Die (total definierte) *Nachfolgerfunktion* $f : \mathbb{N} \to \mathbb{N}$ mit

$$f : n \to n + 1$$

ist Turing-berechenbar.

Turing-Berechenbarkeit: Nachfolgerfunktion

Beispiel: Die (total definierte) *Nachfolgerfunktion* $f : \mathbb{N} \to \mathbb{N}$ mit

$$f : n \to n + 1$$

ist Turing-berechenbar.

Eine Turingmaschine, die f berechnet, ist definiert durch

$$M = (\{0, 1\}, \{0, 1, \Box\}, \{z_0, z_1, z_2, z_e\}, \delta, z_0, \Box, \{z_e\}),$$

mit der folgenden Liste von Turingbefehlen gemäß δ:

$(z_0, 0) \mapsto (z_0, 0, R)$	$(z_1, 0) \mapsto (z_2, 1, L)$	$(z_2, 0) \mapsto (z_2, 0, L)$
$(z_0, 1) \mapsto (z_0, 1, R)$	$(z_1, 1) \mapsto (z_1, 0, L)$	$(z_2, 1) \mapsto (z_2, 1, L)$
$(z_0, \Box) \mapsto (z_1, \Box, L)$	$(z_1, \Box) \mapsto (z_e, 1, N)$	$(z_2, \Box) \mapsto (z_e, \Box, R)$

Tabelle: Liste δ der Turingbefehle von M für die Funktion $f(n) = n + 1$

Turing-Berechenbarkeit: Nachfolgerfunktion

Z	Bedeutung	Absicht
z_0	Anfangszustand	gehe bis zum Wortende und wechsle in Zustand z_1
z_1	addiere 1	mache eine 0 zu 1 bzw. alle 1en zu 0en
z_2	nach links laufen	gehe an den Wortanfang und wechsle in Zustand z_e
z_e	Endzustand	Akzeptieren

Tabelle: Interpretation der Zustände von M

Turing-Berechenbarkeit: Nachfolgerfunktion

Für $n = 5$ ergibt sich:

$$
\begin{aligned}
z_0 101 & \vdash_M 1 z_0 01 \\
& \vdash_M 10 z_0 1 \\
& \vdash_M 101 z_0 \square \\
& \vdash_M 10 z_1 1 \\
& \vdash_M 1 z_1 00 \\
& \vdash_M z_2 110 \\
& \vdash_M z_2 \square 110 \\
& \vdash_M z_e 110
\end{aligned}
$$

Turing-Berechenbarkeit: Nachfolgerfunktion

Für $n = 3$ ergibt sich:

$$
\begin{aligned}
z_0 11 &\vdash_M 1 z_0 1 \\
&\vdash_M 11 z_0 \square \\
&\vdash_M 1 z_1 1 \\
&\vdash_M z_1 10 \\
&\vdash_M z_1 \square 00 \\
&\vdash_M z_e 100
\end{aligned}
$$

Turing-Berechenbarkeit: $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

Beispiel:

- Die (total definierte) Funktion $\mathrm{div}_2 : \mathbb{N} \to \mathbb{N}$ mit

$$\mathrm{div}_2(n) = \left\lfloor \frac{n}{2} \right\rfloor$$

ist Turing-berechenbar.

Turing-Berechenbarkeit: $\text{div}_2 = \lfloor \frac{n}{2} \rfloor$

Beispiel:

- Die (total definierte) Funktion $\text{div}_2 : \mathbb{N} \to \mathbb{N}$ mit

$$\text{div}_2(n) = \left\lfloor \frac{n}{2} \right\rfloor$$

  ist Turing-berechenbar.

- Idee: Falls $n \geq 2$, letzte Ziffer in $\text{bin}(n)$ löschen.

Turing-Berechenbarkeit: $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

Beispiel:

- Die (total definierte) Funktion $\mathrm{div}_2 : \mathbb{N} \to \mathbb{N}$ mit

$$\mathrm{div}_2(n) = \left\lfloor \frac{n}{2} \right\rfloor$$

  ist Turing-berechenbar.

- Idee: Falls $n \geq 2$, letzte Ziffer in $\mathrm{bin}(n)$ löschen.

- Für $n = 0$ ist nichts zu tun, für $n = 1$ muss eine 0 aufs Band geschrieben werden.

Turing-Berechenbarkeit: $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

Eine Turingmaschine, die div_2 berechnet, ist definiert durch

$$M = (\{0,1\}, \{0,1,\square\}, \{z_0, z_1, z_2, z_3, z_4, z_5, z_e\}, \delta, z_0, \square, \{z_e\}),$$

mit dieser Liste der Turingbefehle gemäß der Überführungsfunktion δ:

$(z_0, 0) \mapsto (z_e, 0, N)$ $(z_0, 1) \mapsto (z_1, 1, R)$	$(z_1, 0) \mapsto (z_3, 0, R)$ $(z_1, 1) \mapsto (z_3, 1, R)$ $(z_1, \square) \mapsto (z_2, \square, L)$	$(z_2, 1) \mapsto (z_e, 0, N)$
$(z_3, 0) \mapsto (z_3, 0, R)$ $(z_3, 1) \mapsto (z_3, 1, R)$ $(z_3, \square) \mapsto (z_4, \square, L)$	$(z_4, 0) \mapsto (z_5, \square, L)$ $(z_4, 1) \mapsto (z_5, \square, L)$	$(z_5, 0) \mapsto (z_5, 0, L)$ $(z_5, 1) \mapsto (z_5, 1, L)$ $(z_5, \square) \mapsto (z_e, \square, R)$

Tabelle: Liste δ der Turingbefehle von M für die Funktion $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

Turing-Berechenbarkeit: $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

Z	Bedeutung	Absicht
z_0	Anfangszustand	falls Eingabe = 0, dann wechsle in z_e, sonst in z_1
z_1	Eingabe = 1 testen	falls Eingabe = 1 (Symbol=$\square$), dann wechsle in z_2, sonst z_3
z_2	Eingabe 1 zu 0 machen	(hier konnte nichts gelöscht werden)
z_3	nach rechts laufen	Wortende suchen und in z_4 wechseln
z_4	letztes Zeichen löschen	letztes Zeichen durch $\square$ ersetzen und in z_5 wechseln
z_5	nach links laufen	Wortanfang suchen und in z_e wechseln
z_e	Endzustand	Akzeptieren

Turing-Berechenbarkeit: $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

- Für $n = 0$ ergibt sich:

$$z_0 0 \quad \vdash_M \quad z_e 0$$

Turing-Berechenbarkeit: $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

- Für $n = 0$ ergibt sich:

$$z_0 0 \quad \vdash_M \quad z_e 0$$

- Für $n = 1$ ergibt sich:

$$\begin{aligned} z_0 1 \quad & \vdash_M \quad 1 z_1 \Box \\ & \vdash_M \quad z_2 1 \\ & \vdash_M \quad z_e 0 \end{aligned}$$

Turing-Berechenbarkeit: $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

- Für $n = 2$ ergibt sich:

$$
\begin{array}{rcl}
z_0 10 & \vdash_M & 1 z_1 0 \square \\
 & \vdash_M & 10 z_3 \square \\
 & \vdash_M & 1 z_4 0 \\
 & \vdash_M & z_5 1 \square \\
 & \vdash_M & z_5 \square 1 \\
 & \vdash_M & z_e 1
\end{array}
$$

Turing-Berechenbarkeit: $\mathrm{div}_2 = \lfloor \frac{n}{2} \rfloor$

- Für $n = 5$ ergibt sich:

$$
\begin{aligned}
z_0 101 &\vdash_M 1 z_1 01 \\
&\vdash_M 10 z_3 1 \\
&\vdash_M 101 z_3 \square \\
&\vdash_M 10 z_4 1 \\
&\vdash_M 1 z_5 0 \square \\
&\vdash_M z_5 10 \\
&\vdash_M z_5 \square 10 \\
&\vdash_M z_e 10
\end{aligned}
$$

Turing-Berechenbarkeit für Wortfunktionen

Definition

Eine Funktion $f : \Sigma^* \to \Sigma^*$ heißt *Turing-berechenbar*, falls es eine deterministische Turingmaschine

$$M = (\Sigma, \Gamma, Z, \delta, z_0, \Box, F)$$

gibt, so dass für alle $x, y \in \Sigma^*$:

$$f(x) = y \iff \exists z \in F . z_0 x \vdash_M^* zy$$

Falls $f(x)$ nicht definiert ist, dann läuft M in eine unendliche Schleife oder stoppt in einem Zustand $z \notin F$.