

Nichtdeterministische Turingmaschine

$$(K, \Sigma, \Delta, s, H)$$

⋮

⋮

Δ Übergangsrelation

$$\Delta : (K - H) \times \Sigma \times K \times (\Sigma \cup \{\leftarrow, \rightarrow\})$$

⋮

⋮

Relation \vdash_M wird entsprechend erweitert, ebenso \vdash_M^*

$$M = (K, \Sigma, \Delta, s, H) \quad w \in (\Sigma - \{\sqcup, \triangleright\})^*$$

***M* akzeptiert** Eingabe w , falls $(s, \triangleright \sqcup w) \vdash_M^* (h, \underline{u} \underline{a} \underline{v})$ für ein $h \in H$, $a \in \Sigma$ und $u, v \in \Sigma^*$.

***M* akzeptiert** eine Sprache $L \subseteq \Sigma_0^*$ mit $\Sigma_0 \subseteq \Sigma - \{\sqcup, \triangleright\}$, falls für alle $w \in \Sigma_0^*$ gilt: *M* akzeptiert w genau dann wenn $w \in L$.

$$M = (K, \Sigma, \Delta, s, H = \{y, n\}) \quad w \in (\Sigma - \{\sqcup, \triangleright\})^*$$

M **entscheidet** eine Sprache $L \subseteq \Sigma_0^*$ mit $\Sigma_0 \subseteq \Sigma - \{\sqcup, \triangleright\}$, falls gilt

1. Es gibt ein $N \in \mathbb{N}$, das von M und w abhängt, so dass es keine Konfiguration C gibt mit

$$(s, \triangleright \sqcup w) \vdash_M^N C$$

2. $w \in L$ genau dann wenn $(s, \triangleright \sqcup w) \vdash_M^* (y, uav)$ für $a \in \Sigma$ und $u, v \in \Sigma^*$.

Sei $\Sigma_0 \subseteq \Sigma - \{\sqcup, \triangleright\}$. Eine nichtdeterministische Turingmaschine $M = (K, \Sigma, \Delta, s, \{h\})$ **berechnet** eine Funktion $f : \Sigma_0^* \mapsto \Sigma_0^*$, falls

1. Es gibt ein $N \in \mathbb{N}$, das von M und w abhängt, so dass es keine Konfiguration C gibt mit

$$(s, \triangleright \underline{\sqcup} w) \vdash_M^N C$$

2. $(s, \triangleright \underline{\sqcup} w) \vdash_M^* (h, \underline{uav})$ genau dann wenn $ua = \triangleright \sqcup$ und $v = f(w)$.

Satz: Falls eine nichtdeterministische Turingmaschine M eine Sprache akzeptiert oder entscheidet oder eine Funktion berechnet, so gibt es auch eine deterministische Turingmaschine M' , die die Sprache akzeptiert beziehungsweise entscheidet beziehungsweise die Funktion berechnet.

Beweisidee:

Alle möglichen Berechnungen durchprobieren.

Ein paar Details: Es gibt $r \in \mathbb{N}$, so dass für alle Paare (q, a) gilt

$$|\{(p, b) : (q, a, p, b) \in \Delta\}| \leq r$$

Für jedes Paar (q, a) 4-Tupel (q, a, p_i, b_i) für $i = 1, 2, \dots, r$.

Sei Σ_1 Menge von r Symbolen. Wort aus Σ_1^d kodiert eine Berechnung bestehend aus d Schritten.

3-Band Turingmaschine M :

Band 1: Eingabe w

Band 2: bearbeitete Kopie von w

Band 3: $i_1 i_2 \dots i_d \in \Sigma_1^*$.

Das Symbol auf der j -ten Position von Band 3 bestimmt, welche der r möglichen Aktionen auf Band 2 ausgeführt wird. Wird auf Band 3 ein \square erreicht, so wird die aktuelle deterministische Simulation gestoppt, ein neues Wort auf Band 3 generiert und die zum neu erzeugten Wort gehörige Simulation wird gestartet. Auf Band 3 werden der Länge nach alle Wörter aus Σ_1^* generiert.

□

Weitere Berechenbarkeitsmodelle

Grammatik (V, Σ, R, S)

V Alphabet

Σ $\Sigma \subseteq V$; Terminalalphabet

R $R \subseteq (V^+ - \Sigma^+) \times V^*$; endliche Menge von
Regeln oder auch Produktionen

S $S \in V - \Sigma$; Startsymbol

$V - \Sigma$ Nichtterminal(symbol)e oder Variablen

Sei $G = (V, \Sigma, R, S)$ eine Grammatik.

Falls $(u, v) \in R$, so schreiben wir $u \rightarrow_G v$.

Relation \Rightarrow_G auf $V^* \times V^*$:

$x \Rightarrow_G y$ genau dann wenn es $u, v, w, z \in V^*$ gibt, so dass $x = uvw$, $y = uzv$ und $v \rightarrow_G z$

Ableitung von y in G aus x :

$x = w_0 \Rightarrow_G w_1 \Rightarrow_G \dots \Rightarrow_G w_n = y$

Erzeugte Sprache

$$L(G) = \{w \in \Sigma^* : S \Rightarrow_G^* w\}$$

Beispiel:

$$G = (V, \Sigma, S, R)$$

$$V = \{S, a, b, c, A, B, C, T_a, T_b, T_c\}, \quad \Sigma = \{a, b, c\},$$

$$R = \{S \rightarrow ABCS, S \rightarrow T_c, \\ CA \rightarrow AC, BA \rightarrow AB, CB \rightarrow BC, \\ CT_c \rightarrow T_cc, CT_c \rightarrow T_bc, \\ BT_b \rightarrow T_b b, BT_b \rightarrow T_a b, \\ AT_a \rightarrow T_a a, T_a \rightarrow \varepsilon\}$$

Beispiel:

$$G = (V, \Sigma, S, R)$$

$$V = \{S, a, b, c, A, B, C, T_a, T_b, T_c\}, \quad \Sigma = \{a, b, c\},$$

$$R = \{S \rightarrow ABCS, S \rightarrow T_c, \\ CA \rightarrow AC, BA \rightarrow AB, CB \rightarrow BC, \\ CT_c \rightarrow T_c c, CT_c \rightarrow T_b c, \\ BT_b \rightarrow T_b b, BT_b \rightarrow T_a b, \\ AT_a \rightarrow T_a a, T_a \rightarrow \varepsilon\}$$

$$L(G) = \{a^n b^n c^n : n \geq 1\}$$

Beispiel:

$$G = (V, \Sigma, S, R)$$

$$V = \{S, a, A, D, \diamond, \heartsuit\}, \quad \Sigma = \{a\},$$

$$R = \{S \rightarrow \diamond A \heartsuit, \diamond \rightarrow \diamond D, \\ D \heartsuit \rightarrow \heartsuit, DA \rightarrow AAD, \\ \diamond \rightarrow \varepsilon, \heartsuit \rightarrow \varepsilon, A \rightarrow a\}$$

Beispiel:

$$G = (V, \Sigma, S, R)$$

$$V = \{S, a, A, D, \diamond, \heartsuit\}, \quad \Sigma = \{a\},$$

$$R = \{S \rightarrow \diamond A \heartsuit, \diamond \rightarrow \diamond D, \\ D \heartsuit \rightarrow \heartsuit, DA \rightarrow AAD, \\ \diamond \rightarrow \varepsilon, \heartsuit \rightarrow \varepsilon, A \rightarrow a\}$$

$$L(G) = \{a^{2^n} : n \geq 0\}$$

Grammatiken und Turingmaschinen

Satz: *Eine Sprache ist eine von einer Grammatik erzeugte Sprache genau dann wenn sie rekursiv aufzählbar ist.*

Beweisskizze:

\Rightarrow : Sei $G = (V, \Sigma, R, S)$ eine Grammatik.

Nichtdeterministische 2-Band Turingmaschine M
simuliert Ableitungen:

Band 1: w

Band 2: $x \in V^*$ mit $S \Rightarrow_G^* x$
(anfangs $x = S$)

$|R| + 1$ Optionen, von denen eine nichtdeterministisch ausgewählt wird: Eine von $|R|$ Regeln auf Band 2 anwenden oder Ableiten beenden.

Regel $u \rightarrow v$ anwenden: Startposition für Suche nach u auf Band 2 nichtdeterministisch auswählen, Bandinhalt ab Startposition mit u vergleichen. Falls Übereinstimmung gefunden wurde, zugehörigen Teil von Band 2 löschen, Platz für v anpassen und v auf Band 2 schreiben.

Ableiten beenden: Bänder 1 und 2 vergleichen.

Falls in einem der Schritte keine Übereinstimmung gefunden wurde, in eine Endlosschleife übergehen.

M hält bei Eingabe w genau dann wenn $w \in L(G)$.

\Leftarrow : Sei $(K, \Sigma, \delta, s, \{h\})$ eine Turingmaschine.

Annahmen o.B.d.A.: Falls M hält, dann in der Konfiguration $(h, \triangleright \underline{\sqcup})$; Ferner $K \cap \Sigma = \emptyset$ und $\triangleleft \notin \Sigma \cup K$.

Wir konstruieren eine Grammatik $G = (V, \Sigma, R, S)$ mit

$$V = \Sigma \cup K \cup \{S, \triangleleft\}$$

Konfiguration wird als Wort aus V^* dargestellt:

$$(q, \triangleright \underline{uaw}) \quad \longleftrightarrow \quad \triangleright uaqw \triangleleft$$

Regeln in R entsprechen rückwärts Übergängen von M :

M	G
$\delta(q, a) = (p, b)$	$bp \rightarrow_G aq$
$\delta(q, a) = (p, \rightarrow)$	$abp \rightarrow_G aqb, b \neq \triangleleft$ $a \sqcup p \triangleleft \rightarrow_G aq \triangleleft$
$\delta(q, a \neq \sqcup) = (p, \leftarrow)$	$pa \rightarrow_G aq$
$\delta(q, \sqcup) = (p, \leftarrow)$	$p \sqcup b \rightarrow_G \sqcup qb, b \neq \triangleleft$ $p \triangleleft \rightarrow_G \sqcup q \triangleleft$ $S \rightarrow_G \triangleright \sqcup h \triangleleft$ $\triangleleft \rightarrow_G \varepsilon, \triangleright \sqcup s \rightarrow_G \varepsilon$

Lemma: $(q_1, u_1 \underline{a_1} w_1) \vdash_M (q_2, u_2 \underline{a_2} w_2)$
 g.d.w.
 $u_2 a_2 q_2 w_2 \Rightarrow_G u_1 a_1 q_1 w_1$

□

Nun gilt

$w \in L(G)$

g.d.w. $S \Rightarrow_G \triangleright \sqcup h \triangleleft \Rightarrow_G^* \triangleright \sqcup s w \triangleleft \Rightarrow_G w \triangleleft \Rightarrow_G w$

g.d.w. $\triangleright \sqcup h \triangleleft \Rightarrow_G^* \triangleright \sqcup s w \triangleleft$

g.d.w. $(s, \triangleright \sqcup w) \vdash_M^* (h, \triangleright \sqcup)$

g.d.w. M hält bei Eingabe w

□

Sei $G = (V, \Sigma, R, S)$ eine Grammatik und sei $f : \Sigma^* \mapsto \Sigma^*$ eine Funktion. G **berechnet** f , falls für alle $v, w \in \Sigma^*$ gilt:

$$SwS \Rightarrow_G^* v \quad \text{g.d.w.} \quad v = f(w)$$

Satz: *Eine Funktion ist durch eine Grammatik berechenbar genau dann wenn sie durch eine Turingmaschine berechenbar ist.*

(Beweis Übungsaufgabe)

□