TheGI2: Turing Maschinen

31.05.11

Alan Turing

= Endl. Kontrolle

Beachte:

- kein Indexzugriff
- keine Arithmetic

Def.: Eine Turingmaschine ($\overset{D}{N}TM$) ist ein Siebentupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$

- Z: endl. Menge von Zuständen
- Σ: Eingabealphabet
- $\Gamma \supseteq \Sigma$ Arbeits-/Band-Alphabet, $\Gamma \cap Z = \emptyset$
- z_0: Startzustand
- $\square \in \Gamma \setminus \Sigma$ Blank-Symbol
- $E \subseteq Z$: Endzustände

det: $\delta: Z \times \Gamma \rightarrow Z \times \Gamma \times \{L, R, N\}$

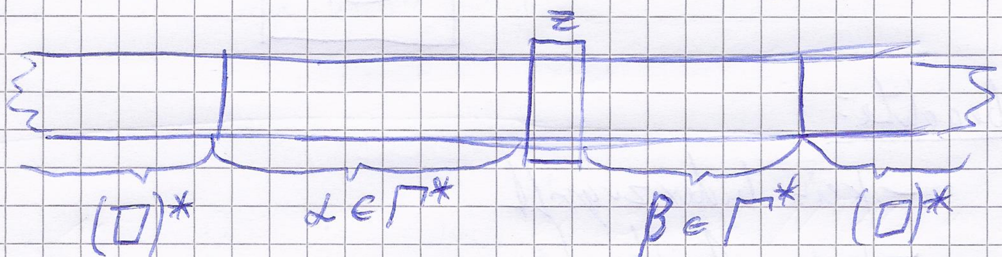
bzw. nicht det: $\delta \subseteq Z \times \Gamma \times Z \times \Gamma \times \{L, R, N\}$

Wenn $\delta(z, a) = (z', b, p)$ bzw. wenn $(z, a, z', b, p) \in \delta$ ist, so muss bzw. kann M im Zustand z und bei Lesen von a in den Zustand z' übergehen, das a durch ein b ersetzen und den Lese-Schreib-Kopf gemäß p bewegen (L: 1 nach links, R: 1 nach rechts; N: keine Bewegung)

Def.

Eine Konfiguration einer TM
$M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ ist
ein Wort $k \in \Gamma^* Z \Gamma^*$

Interpretation: Konf: $\alpha z \beta$



Startkonf zu einem Wort $x \in \Sigma^*$ $z_0 x$

geschwungenes Grundschal-k

Def. Für eine Konfiguration $k = a_1 \dots a_m z b_1 \dots b_n$

$(n=0 \cdot b_1 = \square)$

setzen wir:

Fall 1: $k \vdash_M^1 a_1 \dots a_m z' c b_2 \dots b_n$

wenn $\delta(z, b_1) = (z', c, N)$

Fall 2: $k \vdash_M^1 a_1 \dots a_m c z' b_2 \dots b_n$

wenn $\delta(z, b_1) = (z', c, R)$

Fall 3: a) $k \vdash_M^1 a_1 \dots a_{m-1} z' a_n c b_2 \dots b_n$ falls $m>0$

b) $k \vdash_M^1 z' \square c b_2 \dots b_n$ wenn $\delta($

wenn $\delta(z, b_1) = (z', c, L)$

„Wie immer": Betrachte die reflexive, transitive Hülle von $\overset{1}{\underset{M}{\vdash}}$:

$\overset{*}{\underset{M}{\vdash}}$ $\hat{=}$ Folge von Einschritt übergängen

$\hat{=}$ „Berechnung"

<u>Beispiel:</u> Inkrementieren einer Binärzahl

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

δ	0	1	$\square$
z_0	$(z_0, 0, R)$	$(z_0, 1, R)$	$(z_1, \square, L)$
z_1	$(z_2, 1, L)$	$(z_1, 0, L)$	$(z_e\ 1, N)$
z_2	$(z_2, 0, L)$	$(z_2, 1, L)$	$(z_e, \square, R)$
z_e	—	—	—

Bei Eingabe von 101 ergibt sich folgende Konf.-Folge:

$z_0 \underbrace{101}_{x} \overset{1}{\underset{M}{\vdash}} 1z_0 01 \vdash 10z_0 1 \vdash 101z_0$

$\vdash 10z_1 1 \vdash 1z_1 00 \vdash z_2 110 \vdash z_2 \square 110$

$\vdash \square z_e 110$

<u>Def.</u> Die von einer TM M akzeptierte Sprache ist

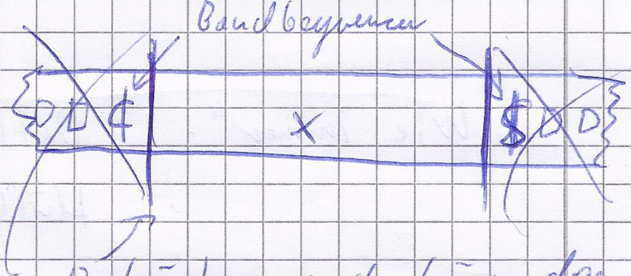
$T(M) := \{ x \in \Sigma^* \mid \exists \alpha, \beta \in \Gamma^* \ \exists z \in E :$

$z_0 x \overset{*}{\vdash} \alpha z \beta$

Def. Eine NTM:

$M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$

heißt <u>linear beschränkt</u> wenn das Eingabewort x durch einen Linken und einen Rechten „Bandbegrenzer" eingerahmt wird und die TM keine Bandpositionen links vom linken und rechts vom rechten Bandbegrenzer betreten darf.



Beschränkung auf Länge des Eingabewortes.

$\hat{=}$ Linear Beschränkter Automat

~~[illegible]~~

Mitteilung 1: Die von lin. beschränkten NTM akzeptierten Sprachen sind genau die Typ-1-Sprachen (kontextsensitive Sprachen)

$\alpha \to \beta$

$|\alpha| \leq |\beta|$

Mitteilung 2:

Die von TM akzeptierten Sprachen sind genau die Typ-0-Sprachen.

<u>Berechenbarkeit</u> (Motivation)

<u>Der intuitive Begriff der Berechenbarkeit und die Church'sche These</u>

→ Berechenbarkeit von (partiellen) Funktionen:

~~Vor-Def.: part. Fktn. $f: \mathbb{N}^k \to \mathbb{N}$ heißt berechenbar, wenn $\exists$ Algorithmus, der bei Eingabe ... $\in \mathbb{N}^k$ genau dann nach endlicher Zeit~~

siehe nächste Seite

$\boxed{\text{Eingabe}} \longrightarrow \boxed{\text{Algorithmus}} \longrightarrow \boxed{\text{Ausgabe}}$

Keine <u>endl.</u> Automaten, da zu schwach.

↳ wir brauchen <u>unendl.</u> Modelle

Speicher ↑

Vor-Def. ~~Wenn es einen~~

part. Fktn: $f: \mathbb{N}^k \to \mathbb{N}$ heißt

berechenbar, wenn es einen Alg. gibt, der bei Eingabe $(n_1, \ldots, n_k) \in \mathbb{N}^k$ genau dann nach endlicher Zeit mit einer Ausgabe $m \in \mathbb{N}$ zum Halten kommt, wenn f an der Stelle $(n_1, \ldots n_k)$ def. ist und $f(n_1, \ldots n_k) = m$ gilt.

Bsp. 1: Der Alg.:

```
INPUT (n)
WHILE TRUE DO
```

berechnet die nirgends def. Fktn. mit leeren Def. Bereich.

$$\Omega: \mathbb{N} \to \mathbb{N}$$
$$n \mapsto \bot$$