Grundlagen der Theoretischen Informatik

Turingmaschinen und rekursiv aufzählbare Sprachen (III)

29.06.2017

Viorica Sofronie-Stokkermans

e-mail: sofronie@uni-koblenz.de

Übersicht

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP

Turing-Maschine

Definition (Turing Machine (DTM))
Eine **determinierte Turing-Maschine (DTM)** $\mathcal{M}$ ist ein Tupel

$$\mathcal{M} = (\ K, \Sigma, \delta, s\ )$$

Dabei ist

- K eine endliche Menge von Zuständen mit $h \notin K$,
  (h ist der **Haltezustand**)
- Σ ein Alphabet mit $L, R \notin \Sigma$, $\# \in \Sigma$,
- $\delta : K \times \Sigma \to (K \cup \{h\}) \times (\Sigma \cup \{L, R\})$ eine Übergangs**funktion**
- $s \in K$ ein Startzustand.

Wir erlauben auch, dass δ nicht überall definiert ist.

Falls die DTM dann in einen solchen nichtdefinierten Zustand kommt, sagen wir **die DTM hängt**. Sie **hält** also nicht.

Dies wird z.T. in der Literatur anders gehandhabt.

Turing-Maschine

Definition **Konfiguration** einer DTM $\mathcal{M}$=(K, Σ, δ, s): Wort der Form C=$q, w\underline{a}u$, wobei:

- $q \in K \cup \{h\}$ der aktuelle Zustand,
- $w \in \Sigma^*$ der Bandinhalt links des Kopfes,
- $a \in \Sigma$ das Bandzeichen unter der Schreib-/Lesekopf.
- $u \in \Sigma^*(\Sigma - \{\#\}) \cup \{\epsilon\}$ der Bandinhalt rechts des Kopfes.

Definition $C_1 \vdash_{\mathcal{M}} C_2$ (C_2 ist **Nachfolgekonfiguration** von C_1) falls:

- $C_i = q_i, w_i\underline{a_i}u_i$ für $i \in \{1, 2\}$, und
- es gibt einen Übergang $\delta(q_1, a_1) = (q_2, b)$ wie folgt:

  Fall 1: $b \in \Sigma$. Dann ist $w_1 = w_2$, $u_1 = u_2$, $a_2 = b$.

  Fall 2: $b = L$. Dann gilt für w_2 und a_2: $w_1 = w_2 a_2$.

  Für u_2: Wenn $a_1 = \#$ und $u_1 = \epsilon$ ist, so ist $u_2 = \epsilon$, sonst ist $u_2 = a_1 u_1$.

  Fall 3: $b = R$. Dann ist $w_2 = w_1 a_1$.

  Für a_2 und u_2 gilt: Wenn $u_1 = \epsilon$ ist, dann ist $u_2 = \epsilon$ und $a_2 = \#$, ansonsten ist $u_1 = a_2 u_2$.

Turing-Maschine

Definition (Eingabe) w heißt **Eingabe** (*input*) für $\mathcal{M}$, falls $\mathcal{M}$ mit der **Startkonfiguration** $C_0 = s, \#w\underline{\#}$ startet.
$(w_1, \ldots, w_n)$ heißt **Eingabe** für $\mathcal{M}$, falls $\mathcal{M}$ mit der **Startkonfiguration**

$$C_0 = s, \#w_1\# \ldots \#w_n\underline{\#}$$

startet.

Definition (Halten, Hängen) Sei $\mathcal{M}$ eine Turing-Maschine.

- $\mathcal{M}$ **hält** in $C = q, w\underline{a}u$ **gdw.** $q = h$.
- $\mathcal{M}$ **hängt** in $C = q, w\underline{a}u$ **gdw.** es keine Nachfolgekonfiguration gibt
  Insbesondere: wenn $w = \epsilon \wedge \exists q' \;\; \delta(q, a) = (q', L)$.

Turing-Maschine

Definition (Rechnung) Sei $\mathcal{M}$ eine Turing-Maschine. Man schreibt $C \vdash_{\mathcal{M}}^{*} C'$ **gdw.**:
es gibt eine Reihe von Konfigurationen $C_0, C_1, \ldots, C_n \quad (n \geq 0)$ so dass
$C = C_0,\ C' = C_n$ und für alle $i < n$ gilt: $C_i \vdash_{\mathcal{M}} C_{i+1}$

Dann heißt $C_0, C_1, \ldots, C_n$ eine **Rechnung** der Länge n von C_0 nach C_n.

Definition (TM-berechenbare Funktion) Sei Σ_0 ein Alphabet mit $\# \notin \Sigma_0$.
Eine (partielle) Funktion $f : ({\Sigma_0}^*)^m \to ({\Sigma_0}^*)^n$ heißt **DTM-berechenbar**, falls:
Es existiert eine determinierte Turing-Maschine $\mathcal{M} = (\ K, \Sigma, \delta, s\ )$

- mit $\Sigma_0 \subseteq \Sigma$,
- so dass für alle $w_1, \ldots, w_m,\ u_1, \ldots, u_n \in {\Sigma_0}^*$ gilt:
  $-f(w_1, \ldots, w_m) = (u_1, \ldots, u_n)$ $\underline{\text{gdw}}$ $s, \#w_1\# \ldots \#w_m\underline{\#} \vdash_{\mathcal{M}}^{*} h, \#u_1\# \ldots \#u_n\underline{\#}$
  $-f(w_1, \ldots, w_m)$ ist undefiniert $\underline{\text{gdw}}$

  $\mathcal{M}$ gestartet mit $s, \#w_1\# \ldots \#w_m\underline{\#}$ hält nicht (läuft unendlich oder hängt)

Berechnete Funktion/Akzeptierte Sprache

Bei Turing-Maschinen untersuchen wir,

- **welche Sprachen sie akzeptieren** und
- **welche Funktionen sie berechnen**.

Akzeptieren ist Spezialfall von Berechnen

Definition (Von einer DTM akzeptierte Sprache)
Ein Wort w wird **akzeptiert von einer DTM $\mathcal{M}$**,
falls $\mathcal{M}$ bei Eingabe von w hält
(wobei am Ende der Kopf auf dem ersten Blank rechts von w steht).

Eine Sprache $L \subseteq \Sigma^*$ **wird akzeptiert von einer DTM $\mathcal{M}$**, wenn genau die Wörter aus L von $\mathcal{M}$ und keine anderen akzeptiert werden.

Achtung

Bei nicht akzeptierten Wörtern muss die DTM nicht halten.

Sie darf es sogar nicht!

TM-Flussdiagramme

Graphische Darstellung der Übergangsfunktion einer DTM: mit einem Flußdiagramm.

- Die Zustandsnamen werden nicht genannt.
- Nur die Schritte und die Ausführungsreihenfolge werden beschrieben.

TM-Flussdiagramme

Folgende Elemente stehen zur Verfügung:

- L: eine DTM, die nach dem Starten ein Feld nach links geht und danach hält.
- R: eine DTM, die nach dem Starten ein Feld nach rechts geht und danach hält.
- a: TM, die a auf das Band schreibt und danach hält.

- Startschritt: mit einer Pfeilspitze $>$ bezeichnet

TM-Flussdiagramme

- $\mathcal{M}_1 \longrightarrow \mathcal{M}_2$ oder abgekürzt $\mathcal{M}_1\mathcal{M}_2$ (falls $\mathcal{M}_1$, $\mathcal{M}_2$ die Flußdiagramme zweier DTM sind):
  eine DTM die zuerst wie $\mathcal{M}_1$ arbeitet und dann, falls $\mathcal{M}_1$ hält, wie $\mathcal{M}_2$ weiterarbeitet.

  Direkt aufeinanderfolgende Schritte werden also
  - entweder direkt nebeneinander notiert
  - oder durch einen Pfeil verbunden.

  Im Gegensatz zu der Maschine $\mathcal{M}_1$ gilt also für $\mathcal{M}_1\mathcal{M}_2$: **Nachdem $\mathcal{M}_1$ seine Arbeit beendet hat, ist $\mathcal{M}_1\mathcal{M}_2$ nicht im Haltezustand, sondern im Startzustand von $\mathcal{M}_2$.**

TM-Flussdiagramme

- $\mathcal{M}_1 \xrightarrow{a} \mathcal{M}_2$: $\mathcal{M}_2$ wird nur dann aufgeführt, wenn nach der Beendigung von $\mathcal{M}_1$ der aktuelle Bandbuchstabe a ist.

Sind $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_n$ Turing-Maschinen, $a_i \in \Sigma$ für $1 \leq i \leq n$, so ist

$$\begin{array}{cll} & \mathcal{M}_1 & \\ & \uparrow^{a_1} & \\ > & \mathcal{M}_0 \stackrel{a_2}{\to} \mathcal{M}_2 & \\ & \downarrow^{a_n} & \ddots \\ & \mathcal{M}_n & \end{array}$$

die Turing-Maschine, die zuerst wie $\mathcal{M}_0$ arbeitet und dann, falls $\mathcal{M}_0$ mit dem Buchstaben a_i auf dem Arbeitsfeld hält, wie $\mathcal{M}_i$ weiterarbeitet.

TM-Flussdiagramme

$\boldsymbol{\sigma}$ — eine Schreibabkürzung für einen beliebigen Buchstaben aus Σ.

- Die Maschine $\mathcal{M} \xrightarrow{\sigma} \ldots \sigma$ zum Beispiel ist eine Abkürzung für

$$
\begin{array}{lll}
 & \ldots a_1 & \\
 & \uparrow^{a_1} & \\
> & \mathcal{M} \xrightarrow{a_2} \ldots a_2 & \\
 & \downarrow^{a_n} \quad \ddots & \\
 & \ldots a_n &
\end{array}
$$

falls $\Sigma = \{a_1, \ldots, a_n\}$ ist.

TM-Flussdiagramme

Die Maschine $> L \stackrel{\sigma}{\to} R\sigma R$ für $\Sigma = \{\#, |\}$ macht also zuerst einen Schritt nach links; steht hier ein $\#$ (bzw. ein $|$), so geht sie einen Schritt nach rechts, druckt $\#$ (bzw. $|$) und geht ein weiteres Feld nach rechts.

- Weitere Schreibabkürzungen sind:

  $\stackrel{\boldsymbol{\sigma}\neq a}{\to}$ für $\sigma \in \Sigma - \{a\}$

  $\mathcal{M}_1 \xrightarrow{\sigma \neq a} \mathcal{M}_2$: $\mathcal{M}_2$ wird nur dann aufgeführt, wenn $\mathcal{M}_1$ hält und Lesekopf auf einem Buchstaben, der nicht a ist positioniert ist.

  $\boldsymbol{\mathcal{M}}_1 \stackrel{a,b}{\to} \boldsymbol{\mathcal{M}}_2$ falls nach der Ausführung von $\mathcal{M}_1$ sowohl für den Bandbuchstaben a als auch für b nach $\mathcal{M}_2$ verzweigt werden soll.

TM-Flussdiagramme

Beispiel:

Die DTM $\mathcal{M}^+ = (\{s, q_1, q_2, q_3, q_4\}, \{|, \#\}, \delta, s)$ addiert zwei natürliche Zahlen in Unärdarstellung.

Sie rechnet

$$s, \#|^n\#|^m\underline{\#} \ \vdash^*_{\mathcal{M}^+} \ h, \#|^{n+m}\underline{\#}$$

Der Trick: Sie löscht den letzten Strich von $|^m$ und schreibt ihn in den Zwischenraum zwischen $|^n$ und $|^m$.

TM-Flussdiagramme

Beispiel: Hier ist zunächst die δ-Funktion:

$s, \#$	$\mapsto$	q_1, L	$q_2, \#$	$\mapsto$	$q_3, \vert$	$q_3, \#$	$\mapsto$	q_4, L
$q_1, \#$	$\mapsto$	$h, \#$	$q_2, \vert$	$\mapsto$	q_2, L	$q_4, \vert$	$\mapsto$	$h, \#$
$q_1, \vert$	$\mapsto$	q_2, L	$q_3, \vert$	$\mapsto$	q_3, R			

Für $\delta(s, |)$ und $\delta(q_4, \#)$ haben wir keine Werte angegeben; sie sind beliebig, weil $\mathcal{M}^+$ sie nie benötigt.

Das Flußdiagramm zur gleichen DTM ist erheblich leichter zu lesen:

$$\begin{tikzcd} > L \arrow[r, "|"] \arrow[d, "\#"] & L \arrow[loop above, "|"] \arrow[r, "\#"] & {|\,R} \arrow[loop above, "|"] \arrow[r, "\#"] & L\# \\ \# \end{tikzcd}$$

TM-Flussdiagramme

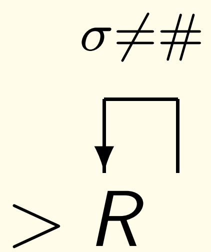
Die folgenden Turing-Maschinen werden wir als Bestandteile von komplexeren Turing-Maschinen später noch häufig benutzen.

Beispiel: ($R_\#$)

$R_\#$ bewegt nur den Kopf, ohne zu schreiben:

- Sie geht mindestens einmal nach rechts.
- Dann geht sie solange weiter nach rechts, bis sie ein $\#$ liest.

Die folgende Turing-Maschine $R_\#$ läuft zum ersten Blank rechts von der momentanen Position.



TM-Flussdiagramme

Beispiel: ($L_{\#}$) Analog funktioniert die DTM $L_{\#}$:

- Sie geht mindestens einmal nach links.
- Dann geht sie solange weiter nach links, bis sie ein $\#$ liest.

$$\begin{tikzcd} > L \arrow[loop above, "\sigma\neq\#"] \end{tikzcd}$$

TM-Flussdiagramme

Beispiel: ($\mathscr{C}$) Die folgende DTM $\mathscr{C}$ erhält als Eingabe einen String Einsen.

Sie rechnet so:

- Sie bewegt sich nach links auf das Blank vor das Eingabewort,
- geht das Eingabewort von links nach rechts durch,
- merkt sich jeweils ein Zeichen σ von w, markiert die aktuelle Position, indem sie σ mit $\#$ überschreibt,
- und kopiert das Zeichen σ.
- Sie verwendet dabei die Maschinen $L_\#$ und $R_\#$, die wir schon definiert haben.

$$
\begin{array}{l}
> L_\# R \overset{\sigma \neq \#}{\longrightarrow} \# R_\# R_\# \sigma L_\# L_\# \sigma \\
\quad\;\; \downarrow^{\#} \\
\quad\;\; R_\#
\end{array}
$$

TM-Flussdiagramme

Beispiel: Die DTM $\mathscr{S}_R$ bewirkt eine “Verschiebung nach rechts”, das heißt, wenn $\mathscr{S}_R$ das Alphabet Σ besitzt, rechnet sie

$$s, w_1\underline{\#}w_2\#w_3 \vdash^*_{\mathscr{S}_R} h, w_1\#\underline{\#}w_2w_3$$

für alle Wörter $w_1, w_3 \in \Sigma^*$ und $w_2 \in (\Sigma - \{\#\})^*$. (Entgegen der sonstigen Konvention startet sie zwischen zwei Eingabewörtern.)
Sie arbeitet so:

$$\begin{tikzcd} \mathscr{S}_R: \quad > R_{\#} L \arrow[r, "\sigma \neq \#"] \arrow[d, "\#"] & R\sigma L \arrow[l, bend right] \\ R\# \end{tikzcd}$$

TM-Flussdiagramme

Beispiel: Dazu invers arbeitet die Maschine $\mathscr{S}_L$, die einen “shift nach links” bewirkt. Sie rechnet

$$s, w_1\#w_2\underline{\#}w_3 \vdash^*_{\mathscr{S}_L} h, w_1w_2\underline{\#}\#w_3$$

für alle $w_1, w_3 \in \Sigma^*$, $w_2 \in (\Sigma - \{\#\})^*$. Sie ist definiert als

$$\mathscr{S}_L: \quad \begin{tikzcd} > L_\# R \arrow[r, "\sigma\neq\#"] \arrow[d, "\#"] & L\sigma R \arrow[l, bend right] \\ L\# \end{tikzcd}$$

Variantes von Turing-Maschinen

Variationen von Turing-Maschinen

Standard-DTM

Die Turing-Maschine, die wir bisher kennen, ...

- ist determiniert
- hat ein einseitig unbeschränktes Band (**Halbband**).

Ab jetzt nennen wir sie auch:

Standard-Turing-Maschine (Standard-DTM oder kurz DTM)

Variationen von Turing-Maschinen

Turing-Maschinen, die nie hängen

Gegeben:

Eine Turing-Maschine $\mathcal{M}$, mit Eingabge $\#w\underline{\#}$

Daraus konstruieren wir eine DTM $\mathcal{M}'$, die

- dasselbe berechnet wie $\mathcal{M}$
- **nie hängt**.

Variationen von Turing-Maschinen

Konstruktion der TM, die nie hängt

Das Bandende ist am Anfang ein Zeichen links vom Eingabewort

DTM $\mathcal{M}'$ rechnet so:

- Sie verschiebt die Eingabe ein Zeichen nach rechts.
- Dann druckt sie ganz links ein **Sonderzeichen** α,
  das das **Bandende** anzeigt.
- Ab dann rechnet sie wie $\mathcal{M}$.
- Aber:
  Wenn sie α erreicht, bleibt sie dort stehen und druckt immer wieder α.

Variationen von Turing-Maschinen

Eigenschaften der nicht-hängenden DTM

- $\mathcal{M}'$ hält für Eingabe w gdw $\mathcal{M}$ hält für Eingabe w.
- $\mathcal{M}'$ hängt nie.
  Wenn $\mathcal{M}$ hängt, rechnet $\mathcal{M}'$ unendlich lang.

O.B.d.A. sollen alle Turing-Maschinen, die wir von jetzt an betrachten, nie hängen.