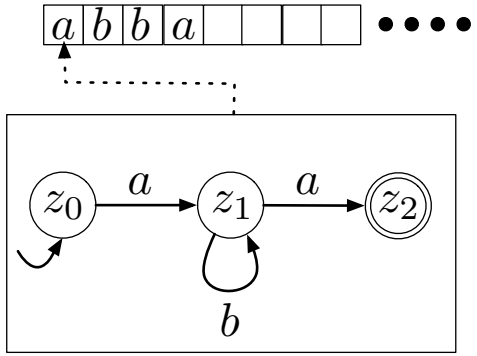
Formale Grundlagen der Informatik 1
Kapitel 9
Turing-Maschinen (Teil 2)

Frank Heitmann
heitmann@informatik.uni-hamburg.de

5. Mai 2015

Vom DFA zur TM



Wir wollen
- auf dem Band nach rechts **und** links gehen können und
- auf dem Band lesen **und** schreiben können.

TM formal

Definition (Deterministische TM)

Eine **deterministische Turing-Maschine** (kurz DTM) ist ein 6-Tupel $A = (Z, \Sigma, \Gamma, \delta, z_0, Z_{end})$ mit
- Der endlichen Menge von **Zuständen** Z.
- Dem endlichen Alphabet Σ von **Eingabesymbolen**.
- Dem endlichen Alphabet Γ von **Bandsymbolen**, wobei $\Gamma \supsetneq \Sigma$ und $\Gamma \cap Z = \emptyset$ gilt.
- Der (partiellen) **Überführungsfunktion** $\delta : (Z \times \Gamma) \to (\Gamma \times \{L, R, H\} \times Z)$.
- Dem **Startzustand** $z_0 \in Z$.
- Der Menge der **Endzustände** $Z_{end} \subseteq Z$.
- Dem **Symbol für das leere Feld** $\# \in \Gamma \setminus \Sigma$.

Die Überführungsfunktion

Mit $\delta(z, x) = (x', B, z')$ ist gemeint, dass, wenn
- die TM im Zustand z ist und
- der LSK gerade über einem Feld mit dem Symbol x ist,
- nun das x durch x' überschrieben wird,
- dann der LSK nach $B \in \{L, R, H\}$ bewegt wird
  - L ist nach links bewegen,
  - R nach rechts und
  - H den LSK an der Stelle halten
- und in den Zustand z' gewechselt wird.

Anmerkung

Ist $B = H$, so bedeutet dies nur, dass der LSK sich nicht bewegt. Es heißt nicht, dass die TM anhält (wichtig für später).

Konfiguration einer TM

Definition (Konfiguration)

Ein Wort $w \in \Gamma^* \cdot Z \cdot \Gamma^*$ heißt **Konfiguration** der TM $A := (Z, \Sigma, \Gamma, \delta, z_0, Z_{end})$. Ist $w = uzv$ mit $z \in Z$ und $u, v \in \Gamma^*$, dann ist

- A im Zustand z,
- die Bandinschrift ist uv (links/rechts davon nur $\#$)
- und das erste Symbol von v ist unter dem LSK. (Ist $v = \lambda$, so ist $\#$ unter dem LSK. Ist $v \neq \lambda$, so ist $v \in \Gamma^*(\Gamma \setminus \{\#\})$.)

Die Menge aller Konfigurationen der TM M ist KONF_M.

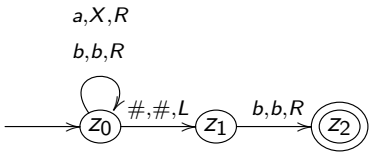
Rechnung einer TM

Definition (Rechnung einer TM)

Sei A eine DTM.

1. Die **Schrittrelation** wird mit $\vdash_A$ bezeichnet (es ist $\vdash_A \subseteq \text{KONF}_A \times \text{KONF}_A$). Der Index A kann auch weggelassen werden. (Die genaue Definition ist etwas langatmig. Siehe bei Bedarf letzten Foliensatz.)
2. Mit $\vdash^*$ wird die reflexive, transitive Hülle von $\vdash$ bezeichnet.
3. Eine Folge $k_1 \vdash k_2 \vdash \ldots \vdash k_i \vdash \ldots$ heißt **Rechnung** der TM.
4. Eine endliche Rechnung $k_1 \vdash k_2 \vdash \ldots \vdash k_n$ heißt **Erfolgsrechnung**, wenn $k_1 \in \{z_0\} \cdot \Sigma^*$ und $k_n \in \Gamma^* \cdot Z_{end} \cdot \Gamma^*$ gilt.

TM: Beispiel



$$z_0aab \vdash Xz_0ab \vdash XXz_0b \vdash XXbz_0\# \vdash XXz_1b \vdash XXbz_2\#$$

Die Kante (a, X, R) bedeutet: Wenn der LSK a liest, dann kann man die Kante nutzen und schreibt dann X (an Stelle von a) und bewegt den LSK nach rechts.

Akzeptierte Sprache einer TM

Definition (Akzeptierte Sprache einer TM)

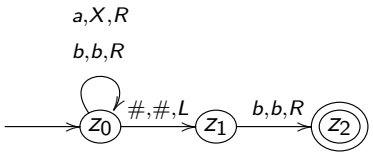
Sei $A = (Z, \Sigma, \Gamma, \delta, z_0, Z_{end})$ eine DTM. Mit $L(A)$ wird die von A akzeptierte Sprache bezeichnet:

$$L(A) := \{w \in \Sigma^* \mid \exists u, v \in \Gamma^* \; \exists z_e \in Z_{end} : \; z_0w \vdash^* uz_ev\}$$

Wichtige Anmerkung

Die TM akzeptiert also sobald sie einen Endzustand erreicht! Sie muss nicht das ganze Eingabewort dafür betrachten, aber sie akzeptiert das ganze Eingabewort (!), sobald sie in einen Endzustand gelangt!

Akzeptierte Sprache einer TM - Beispiel

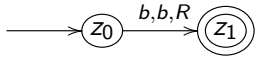


Welche Sprache akzeptiert diese TM?

Die Sprache

$$L = \{w \in \{a, b\}^* \mid w \text{ endet auf } b\}$$

Akzeptierte Sprache einer TM - Beispiel



Welche Sprache akzeptiert diese TM?

Eine TM M für die Sprache

$$L = \{w \in \{a, b\}^* \mid w \text{ beginnt mit } b\}$$

Berechnen von Wortfunktionen

Definition

Sei Σ ein Alphabet und $f : \Sigma^* \to \Sigma^*$ eine (möglicherweise partielle) (Wort-)Funktion. f heißt **Turing-berechenbar** oder kürzer **berechenbar** oder auch **partiell rekursiv** genau dann, wenn

es eine DTM A gibt mit $z_0 w \vdash^* z_e v$ für ein $z_e \in Z_{end}$
genau dann, wenn
$f(w) = v$ ist

Berechnen von Funktionen

Definition

Eine (partielle) Funktion $f : \mathbb{N}^r \to \mathbb{N}^s$ heiß **(Turing-)berechenbar** oder **partiell rekursiv** genau dann, wenn es eine DTM gibt mit

$$z_0 0^{m_1+1} 1 0^{m_2+1} 1 \ldots 1 0^{m_r+1} \vdash^* z_e 0^{n_1+1} 1 0^{n_2+1} 1 \ldots 1 0^{n_s+1}$$

genau dann, wenn

$$f(m_1, m_2, \ldots, m_r) = (n_1, n_2, \ldots, n_s)$$

und der Funktionswert definiert ist.

Meist werden die Zahlen nicht wie oben unär, sondern binär kodiert.

Beispiel

Beispiel

Wir wollen eine TM, die $f(x) = x - 1$ berechnet. Dabei sei x eine natürliche Zahl, die in Binärkodierung gegeben ist. D.h. wir wollen eine Wortfunktion von x nach $f(x)$ berechnen.

Die TM arbeitet wie folgt:

1. Fahre zum am weitesten rechts stehenden Zeichen von x.
2. Ist dies eine 1, schreibe eine 0 und fahre fort bei Schritt 4
3. Ist dies eine 0, schreibe eine 1, gehe ein Feld nach links und mach bei Schritt 2 weiter.
4. Fahre ganz nach links und gehe in einen Endzustand.

Die Church-Turing-These

Wie obige -1-Funktion kann man auch die allgemeine Subtraktion, die Addition, die Multiplikation und viele weitere Berechnungen in eine TM kodieren.

Church-Turing-These

Alles was intuitiv berechenbar ist, d.h. alles, was von einem Menschen berechnet werden kann, das kann auch von einer Turing-Maschine berechnet werden. Ebenso ist alles, was eine andere Maschine berechnen kann, auch von einer Turing-Maschine berechenbar.
Umkehrschluss: Was eine Turing-Maschine nicht kann, kann auch kein Mensch berechnen!

Berechnen und Akzeptieren

Berechnen von Funktionen und Akzeptieren von Sprachen ist ähnlich:

- Akzeptieren einer Sprache M ist Berechnen der charakteristischen Funktion von M.
- Berechnen einer Funktionen $f : M \to N$ ist Akzeptieren der Sprache $L = \{(x, f(x)) \mid x \in M\}$ (ggf. müssen M und N geeignet kodiert werden).

Bemerkung

Die charakteristische Funktion einer Menge M tut folgendes:

$$\chi_M(x) = 1 \text{ gdw. } x \in M$$

(Ist also $x \notin M$, so ist $\chi_M(x) = 0$.)

Zusammenfassung

Bisher:

- Definition der TM
- Definition der Arbeitsweise der TM
  - Konfiguration
  - Schrittrelation
  - Rechnung, Erfolgsrechnung
- Akzeptieren von Sprachen
- Berechnen von Funktionen
- Church-Turing-These

Fragen?!

Fragen bis hierhin?!

Weiterer Ablauf

Der weitere Ablauf heute:

1. Varianten von Turing-Maschinen:
   1. Ein-Band TM mit beidseitig unendlichem Band
   2. Mehrband-TM
   3. Nichtdeterministische TM
2. Aufzählbarkeit und Entscheidbarkeit

Äquivalenz

Definition

Zwei Turing-Maschinen A und B sind äquivalent, wenn $L(A) = L(B)$ gilt.

Ein-Band TMs

Definition

Die bisherige TM hat ein einseitig unendliches Band. Man kann eine TM definieren, bei der das Band in beide Richtungen (**beidseitig**) unendlich ist. Die Startkonfiguration ist dann weiterhin $z_0 w$ bei Eingabe von w, aber man kann jetzt mit dem LSK auch beliebig weit nach links wandern.

Nebenbemerkung

Die Definition der Schrittrelation wäre anzupassen. Bei der beidseitigen TM ist das noch recht einfach. Wir wollen hier dennoch darauf verzichten, dies im Detail zu machen. Ein intuitives Verständnis dieser Maschinen genügt uns hier.

Ein-Band TMs

Satz

Zu jeder DTM A mit einseitig unendlichem Band gibt es eine äquivalente DTM B mit beidseitig unendlichem Band und umgekehrt.

Beweisidee

Die Idee:

- Das Band von A hat ein 0tes, 1tes, 2tes usw. Feld
- Das Band von B hat ein
  - 0tes Feld (dort ist zu Anfang der LSK) und
  - ein 1tes und -1tes Feld (rechts/links vom LSK),
  - ein 2tes und -2tes Feld (zwei Felder rechts/links vom LSK)
  - usw.

Ein-Band TMs

Beweisidee

- A merkt sich nun auf dem n-ten Feld des Bandes was auf dem n-ten und $-n$-ten Feld von B steht. Dazu ist das $\Gamma \times \Gamma$ das Bandalphabet von A (Γ ist das Bandalphabet von B). Man sagt, man hat das Band in *Spuren* eingeteilt.
- A merkt sich im Zustand, ob B gerade im positiven oder negativen Bereich ist. Zustände von A sind also $[z, P]$ und $[z, N]$ (wobei $z \in Z_B$ ist).
- Dann arbeitet A wie B und ändert bei $(x, y) \in \Gamma \times \Gamma$ nur das Element was durch das P bzw. N im Zustand vorgegeben wird.
- Nur bei dem 0-Feld muss man aufpassen, da hier ein Wechsel von P zu N oder andersherum auftritt.

Mehr-Band TMs

Definition

Eine k-**Band off-line Turing-Maschine** hat

- k beidseitig unendliche **Arbeitsbänder** mit jeweils einem eigenen LSK,
- ein **Eingabeband**, auf dem sie ausschließlich lesen kann, aber den Lese-Kopf dabei in beide Richtungen bewegen darf und
- ein **Ausgabeband**, auf dem sie nur schreiben und den Schreibkopf ausschließlich von links nach rechts bewegen darf.

Nebenbemerkung

Hier wäre die Definition der Konfiguration und der Schrittrelation anzupassen. Dies ist etwas aufwändiger als bei der vorherigen TM, aber dennoch recht schnell möglich. Wir wollen dennoch darauf verzichten, dies im Detail zu machen. Ein intuitives Verständnis genügt auch hier.

Mehr-Band TMs

Satz

Zu jeder k-Band off-line Turing-Maschine A mit $k \geq 1$ gibt es eine äquivalente DTM B mit nur einem Band.

Beweisidee

Die Idee: Im Grund genommen so wie eben. Nur mit mehr Spuren. Da A außerdem mehr Lese-/Schreibköpfe hat, muss man bei jeder Spur markieren, wo der jeweilige LSK ist und diese einzeln bearbeiten.

Varianten der TM. Ergebnis

Satz

Zu jeder DTM A mit einseitig unendlichem Band gibt es eine äquivalente DTM B mit beidseitig unendlichem Band und umgekehrt.

Satz

Zu jeder k-Band off-line Turing-Maschine A mit $k \geq 1$ gibt es eine äquivalente DTM B mit nur einem Band.

Wichtige Anmerkung

Diese Varianten sind also äquivalent und man kann stets die TM-"Art" nehmen, die einem gerade mehr zusagt!

Literaturhinweis

Genaue Beweise zu den obigen Aussagen findet man teilweise im Skript und sonst in [HMU].

Varianten der TM

Anmerkung

Wir benutzen im Allgemeinen:

- Eine TM mit einem Band, das in beide Richtungen unendlich ist, wenn wir eine TM konstruieren wollen und das Zustandsübergangsdiagramm angeben wollen.
- Eine k-Band off-line TM, wenn wir nur die Funktionsweise einer TM erläutern wollen.

Fragen

Angenommen wir statten eine TM statt mit einem Band mit einem zweidimensionalen Feld aus, dass nach rechts und nach unten unendlich weit wachsen kann. Ist dies auch äquivalent zur ursprünglichen TM?

1. Ja - und ich habe auch eine Idee, wie das geht...
2. Ja - aber ich habe keine Ahnung warum ...
3. Nein, das sind quasi unendlich viele unendlich lange Bänder, das ist dann doch zu viel ...
4. Bin mir nicht sicher ...

Fragen

Die Konfigurationen einer TM sind definiert als Wörter aus...

1. $\Gamma \cdot Z \cdot \Gamma$
2. $\Gamma^* \cdot Z \cdot \Gamma^*$
3. $\Gamma^+ \cdot Z \cdot \Gamma^+$
4. $\Gamma^* \cdot Z \cdot \Gamma^+$
5. $\Gamma^+ \cdot Z \cdot \Gamma^*$
6. Uffff ....

Fragen

Wie definieren Sie die Übergangsfunktion einer *nichtdeterministischen* TM?

1. $\delta : Z \to 2^{\Gamma \times \Gamma \times \{L,R,H\} \times Z}$
2. $\delta : Z \times \Gamma \to 2^{\Gamma \times \{L,R,H\} \times Z}$
3. $\delta : Z \times \Gamma \times \Gamma \to 2^{\{L,R,H\} \times Z}$
4. $\delta : Z \times \Gamma \times \Gamma \times \{L,R,H\} \to 2^{Z}$
5. Anders!
6. Weiß ich nicht...

Zur Nachbereitung

Zur Nachbereitung

1. 1. und 2. - nach der Diskussion hoffentlich 1. ;)
2. 2
3. 2. (siehe nachfolgende Folien)

Nichtdeterministische Turing-Maschinen

Definition (Nichtdeterministische TM)

Eine TM $M = (Z, \Sigma, \Gamma, K, z_0, Z_{end})$ heißt **nichtdeterministische Turing-Maschine** (kurz NTM), wenn es zu jedem Paar $(z, x) \in Z \times \Gamma$ eine endliche Anzahl von Übergängen gibt. D.h. es ist

$$K \subseteq Z \times \Gamma \times \Gamma \times \{L, R, H\} \times Z$$

bzw.

$$\delta : Z \times \Gamma \to 2^{\Gamma \times \{L,R,H\} \times Z}$$

alles andere ist wie bei der DTM definiert.

Eine NTM akzeptiert ein Eingabewort w dann wieder genau dann, wenn es *mindestens eine* Erfolgsrechnung auf w gibt.

NTM: Beispiel

Das Problem

Eingabe: Ein ungerichteter Graph $G = (V, E)$ und ein $k \in \mathbb{N}$.
Frage: Hat G eine k-Clique?

Eine k-Clique sind k Knoten, die alle paarweise miteinander verbunden sind.

Wie löst ihr das mit einer DTM? Wie mit einer NTM?

Die Eingabe ist dabei die Zahl k, gefolgt von der Liste der Knoten, gefolgt von der Liste der Kanten (als Tupel).

NTM: Beispiel

Das Problem

Eingabe: Ein ungerichteter Graph $G = (V, E)$ und ein $k \in \mathbb{N}$.
Frage: Hat G eine k-Clique?

Mit einer Mehrband-NTM:

- Zähle zunächst die Anzahl der Knoten. Sei das n.
- Auf dem Arbeitsband rate nichtdeterministisch 0 oder 1 (d.h. es gibt eine Kante, die eine 0 auf das Arbeitsband schreibt und eine, die bei gleichen Voraussetzungen eine 1 auf das Arbeitsband schreibt)
- Wiederhole das n-mal.

NTM: Beispiel

Das Problem

Eingabe: Ein ungerichteter Graph $G = (V, E)$ und ein $k \in \mathbb{N}$.
Frage: Hat G eine k-Clique?

Mit einer Mehrband-NTM:

- Die NTM ist jetzt nichtdeterministisch in 2^n Konfigurationen.
- Die Konfigurationen unterscheiden sich nur in der Bandinschrift des Arbeitsbandes. Auf diesem stehen die Zahlen zwischen 0^n und 1^n.
- In jeder diese Konfigurationen geht es jetzt deterministisch weiter...

NTM: Beispiel

Das Problem

Eingabe: Ein ungerichteter Graph $G = (V, E)$ und ein $k \in \mathbb{N}$.
Frage: Hat G eine k-Clique?

- Ist die i-te Zahl auf dem Arbeitsband eine 1 bedeutet, dass das die NTM (in dieser Rechnung!) vermutet, dass der i-te Knoten zur Clique gehört.
- Sie überprüft nun deterministisch, ob die 1 gerade k-mal vorkommt und
- ob die nötigen Kanten zwischen den vermuteten Knoten vorhanden sind.
- Falls ja, akzeptiert sie, sonst nicht.

NTM: Beispiel

Bemerkung

Man kann an obigen Beispiel auch gut sehen, wie man die Arbeitsweise einer TM beschreiben kann, ohne das Zustandsübergangsdiagramm anzugeben. Die Beschreibung wird dann abstrakter, muss aber genau genug bleiben, um nachvollzogen werden zu können und es dürfen nur Operationen vorkommen, die man tatsächlich auch mit einer (ggf. sehr komplizierten TM) machen kann!

NTM $\rightarrow$ DTM

Damit ist die NTM scheinbar viel schneller als die DTM! Dazu später mehr, jetzt die Frage: Kann die NTM generell Dinge, die die DTM nicht kann?

Satz

Jede von einer NTM akzeptierte Sprache kann auch von einer DTM akzeptiert werden und umgekehrt.

Die Richtung von DTM zu NTM ist wieder klar (da man die zusätzliche Eigenschaft der NTM nicht auszunutzen braucht).

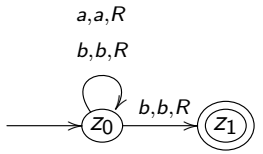
Von NTM zu DTM?

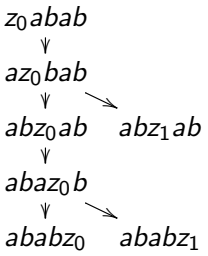
NTM → DTM

Die Idee:

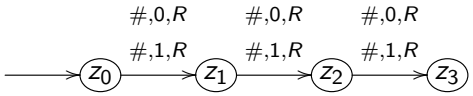
- Ist c_1 eine Konfiguration einer NTM, so kann diese mehrere (nichtdeterministische) Nachfolgekonfiguration haben.
- Dies sind aber endlich viele! Seien dies $c_{1,1}, c_{1,2}, \ldots, c_{1,r}$.
- Man kann sich dies als Baum vorstellen. c_1 als Wurzel. $c_{1,1}, \ldots, c_{1,r}$ als Kinder von c_1.
- Die $c_{1,i}$ haben wieder Kinder $c_{1,1,1}, \ldots, c_{1,1,j_1}$ und $c_{1,2,1}, \ldots, c_{1,2,j_2}$ usw. die die dritte Ebene im Baum bilden.

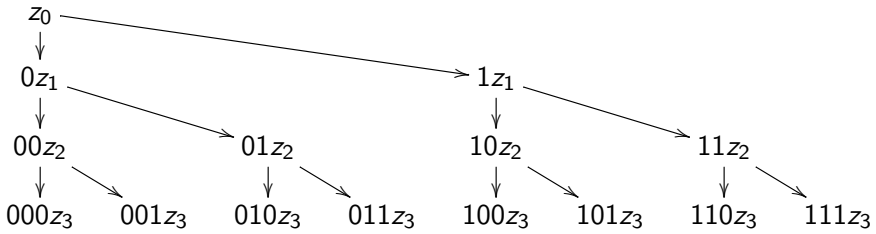
NTM → DTM





NTM → DTM





NTM → DTM

Die Idee:

- Ist c_1 eine Konfiguration einer NTM, so kann diese mehrere (nichtdeterministische) Nachfolgekonfiguration haben.
- Dies sind aber endlich viele! Seien dies $c_{1,1}, c_{1,2}, \ldots, c_{1,r}$.
- Man kann sich dies als Baum vorstellen. c_1 als Wurzel. $c_{1,1}, \ldots, c_{1,r}$ als Kinder von c_1.
- Die $c_{1,i}$ haben wieder Kinder $c_{1,1,1}, \ldots, c_{1,1,j_1}$ und $c_{1,2,1}, \ldots, c_{1,2,j_2}$ usw. die die dritte Ebene im Baum bilden.

Die Idee

Die Idee ist nun, dass die DTM eine Breitensuche in diesem Baum macht!

NTM → DTM

Die Ausführung (grob):

- Die DTM hat ein weiteres Arbeitsband. Auf diesem notiert sie sich, wenn sie das Eingabewort w erhält, die Startkonfiguration der NTM $z_0 w$.
- Sind auf dem Arbeitsband die Konfigurationen $c_1, \ldots, c_r$ notiert (durch ein Extrasymbol \$ getrennt), so
  - werden diese von links nach rechts durchgearbeitet.
  - Jede wird durch ihre Nachfolgekonfigurationen ersetzt.
  - Dabei ist es vermutlich mehrmals nötig, die aktuelle Inschrift auf dem Arbeitsband nach rechts zu verschieben.
- Wird dabei eine Konfiguration erzeugt, die einen Endzustand enthält, akzeptiert die DTM.

NTM → DTM

Anmerkung

Wichtig ist, dass nachdem z.B. die Konfiguration c_1 durch ihre Nachfolgekonfigurationen $c_{1,1}, \ldots, c_{1,r}$ ersetzt wurde, als nächstes mit c_2 weitergemacht wird (nicht mit $c_{1,1}$). Sonst hat man eine Tiefensuche statt einer Breitensuche und würde dann evtl. eine unendlich langen Rechnung der NTM simulieren und dabei eine Erfolgsrechnung auf einem anderen Pfad verpassen!

Damit haben wir:

Satz

Jede von einer NTM akzeptierte Sprache kann auch von einer DTM akzeptiert werden und umgekehrt.

Weiterer Ablauf

Der weitere Ablauf heute:

1. Aufzählbarkeit und Entscheidbarkeit

Aufzählbarkeit

Definition

1. Eine Menge $M \subseteq \Sigma^*$ heißt **(rekursiv) aufzählbar** genau dann, wenn $M = \emptyset$ ist oder eine totale, berechenbare Funktion $g : \mathbb{N} \to \Sigma^*$ existiert mit $g(\mathbb{N}) = M$.
2. Die Familie aller aufzählbaren Mengen wird mit RE (recursively enumerable) bezeichnet.

Satz

Eine Menge M ist aufzählbar genau dann, wenn

1. *eine k-Band off-line TM existiert, die jedes Wort der Menge M genau einmal auf ihr Ausgabeband schreibt.*
2. *$M = L(A)$ für eine DTM A gilt.*

Aufzählbarkeit

Definition (Alternative Definition von RE)

Die von Turing-Maschinen akzeptierten Sprachen bilden die Sprachfamilie RE.

Wichtige Anmerkung

- Man überlegt sich schnell, dass die TM alles kann, was ein DFA kann.
- In den Aufgaben zeigt ihr, dass sie auch alles kann, was ein PDA kann.
- Zusäzlich kann sie z.B. $\{a^n b^n c^n \mid n \in \mathbb{N}\}$ akzeptieren.

Damit kann sie alles, was die uns bisher bekannten Automatenmodelle konnten - und mehr.

Aufzählbarkeit

Definition (Alternative Definition von RE)

Die von Turing-Maschinen akzeptierten Sprachen bilden die Sprachfamilie RE.

Bemerkung

Die Sprachfamilie RE wird auch von den wenig eingeschränkten Grammatiken erfasst, d.h. von denen, bei denen für die Produktionen

$$P \subseteq V^* \cdot V_N \cdot V^* \times V^*$$

gilt (sogenannte Typ-0-Grammatiken).

Entscheidbarkeit

Definition

1. Eine Menge $M \subseteq \Sigma^*$ heißt **entscheidbar** genau dann, wenn die charakteristische Funktion $\chi_M : \Sigma^* \to \{0,1\}$ berechenbar ist.
2. Die Familie aller entscheidbaren Mengen wird mit REC (recursive sets) bezeichnet.

Bemerkung

Die charakteristische Funktion einer Menge M tut folgendes:

$$\chi_M(x) = 1 \text{ gdw. } x \in M$$

(Ist also $x \notin M$, so ist $\chi_M(x) = 0$.)

Entscheidbarkeit

Satz

Eine Menge M ist genau dann entscheidbar, wenn es eine TM A gibt mit $L(A) = M$ und derart, dass A auf jeder Eingabe anhält.

Anmerkung

Man könnte dies auch als alternative Definition benutzen.

Entscheidbarkeit vs. Aufzählbarkeit

Entscheidbarkeit von M:

- Es gibt eine TM A mit $L(A) = M$.
- **und** A hält auf jeder Eingabe (in einem Endzustand, wenn das vorgelegte Wort in M ist, sonst in einem Nicht-Endzustand).

Aufzählbarkeit von M:

- Es gibt eine TM A mit $L(A) = M$.

Insb. muss die TM für eine aufzählbare Menge nicht bei jeder Eingabe anhalten! Bei Worten in M tut sie es (da die Worte ja auch in $L(A)$ sind). Bei Worten, die nicht in M sind, tut sie es nicht zwingend. Rechnet eine TM also lange, dann kann das daran liegen, dass das Wort nicht in M ist oder dass sie noch nicht fertig ist!

Zusammenfassung

Wir haben heute und gestern:

- **Turing-Maschinen** eingeführt.
- Diese **akzeptieren Sprachen**
- und **berechnen Funktionen**.
- Es gibt **Varianten** wie die Mehrband-TM usw. und insb. die **nichtdeterministische TM**.
- Die DTM kann die NTM simulieren!
- Wir haben die Begriffe
  - **entscheidbar** und
  - **aufzählbar**

  eingeführt

Nächste Woche zeigen wir, dass es nicht-entscheidbare Mengen gibt! Und machen uns dann noch Gedanken über den Aufwand, den wir bei entscheidbaren Mengen betreiben müssen.