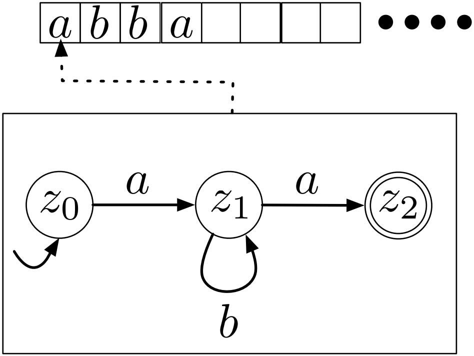
Formale Grundlagen der Informatik 1
Kapitel 8
Turing-Maschinen (Teil 2)

Frank Heitmann
heitmann@informatik.uni-hamburg.de

29. April 2014

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** $\# \in \Gamma \setminus \Sigma$.

Konfiguration einer TM

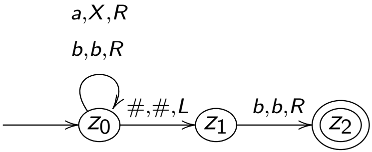
Definition (Konfiguration)

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 $v \in \Gamma^*(\Gamma \setminus \{\#\})$.)

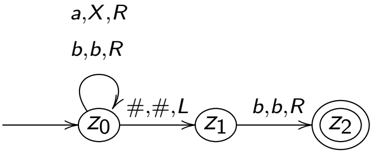
Die Menge aller Konfigurationen der TM M ist KONF_M.

TM: Beispiel



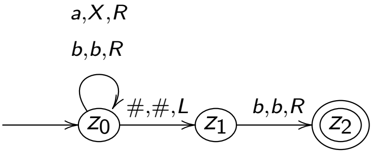
$z_0aab \vdash Xz_0ab \vdash XXz_0b \vdash XXbz_0\# \vdash XXz_1b \vdash XXbz_2\#$

TM: Beispiel



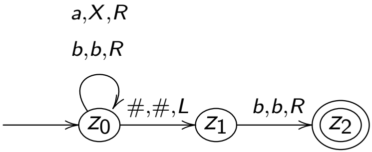
$$z_0 aab \vdash X z_0 ab \vdash XX z_0 b \vdash XXb z_0 \# \vdash XX z_1 b \vdash XXb z_2 \#$$

TM: Beispiel



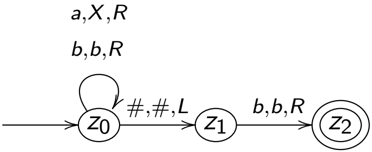
$$z_0 aab \vdash Xz_0 ab \vdash XXz_0 b \vdash XXbz_0\# \vdash XXz_1 b \vdash XXbz_2\#$$

TM: Beispiel



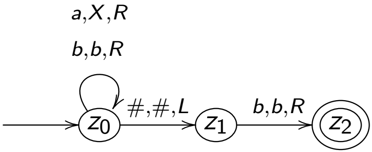
$$z_0aab \vdash Xz_0ab \vdash XXz_0b \vdash XXbz_0\# \vdash XXz_1b \vdash XXbz_2\#$$

TM: Beispiel



$$z_0 aab \vdash Xz_0 ab \vdash XXz_0 b \vdash XXbz_0\# \vdash XXz_1 b \vdash XXbz_2\#$$

TM: Beispiel



$$z_0 aab \vdash Xz_0 ab \vdash XXz_0 b \vdash XXbz_0\# \vdash XXz_1 b \vdash XXbz_2\#$$

Rechnung einer TM

Definition (Schrittrelation einer DTM)

Sei A eine DTM. Die **Schrittrelation** $\vdash_A \subseteq \mathrm{KONF}_A \times \mathrm{KONF}_A$ ist definiert durch: $w \vdash_A w'$ gilt gdw. 1, 2, 3 oder 4 unten gilt.
Es ist $u, v, w \in \Gamma^*$, $x, y, z \in Y$, $p, q \in Z$.

1. $w = uypxv$ und

$$w' = \begin{cases} uqyzv, & \text{falls } (v \neq \lambda \text{ oder } z \neq \#) \text{ und } \delta(p, x) = (z, L, q) \\ uqy, & \text{falls } v = \lambda, y \neq \# \text{ und } \delta(p, x) = (\#, L, q) \\ uq, & \text{falls } v = \lambda, y = \# \text{ und } \delta(p, x) = (\#, L, q) \\ uyzqv, & \text{falls } \delta(p, x) = (z, R, q) \end{cases}$$

Bemerkung

Häufig (z.B. in dem Beispiel vorhin) notiert man noch ein $\#$ wenn der Zustand ganz rechts steht. Z.B. im dritten Fall oben $uq\#$ statt uq. Entspricht nicht ganz der Definition, ist aber ok.

Rechnung einer TM

Definition (Schrittrelation einer DTM (Forts.))

Es ist $u, v, w \in \Gamma^*$, $x, y, z \in \Gamma$, $p, q \in Z$.

1. $w = uyp$ und

$$w' = \begin{cases} uqyz, \text{ falls } z \neq \# \text{ und } \delta(p, \#) = (z, L, q) \\ uqy, \text{ falls } y \neq \# \text{ und } d(p, \#) = (\#, L, q) \\ uq, \text{ falls } y = \# \text{ und } \delta(p, \#) = (\#, L, q) \\ uyzq, \text{ falls } \delta(p, \#) = (z, R, q) \end{cases}$$

2. $w = pxv$ und

$$w' = zqv, \text{ falls } \delta(p, x) = (z, R, q)$$

3. $w = p$ und

$$w' = zq, \text{ falls } \delta(p, \#) = (z, R, q)$$

Rechnung einer TM

Definition (Rechnung einer TM)

1. Mit $\vdash_A^*$ wird die reflexive, transitive Hülle von $\vdash_A$ bezeichnet. Der Index A kann auch weggelassen werden.
2. Eine Folge $k_1 \vdash k_2 \vdash \ldots \vdash k_i \vdash \ldots$ heißt **Rechnung** der TM.
3. 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.

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:

$$L(A) := \{w \in \Sigma^* \mid \exists u, v \in \Gamma^* \ \exists z_e \in Z_{end} : \ z_0 w \vdash^* u z_e v\}$$

Wichtige Anmerkung

Nochmal: Die TM akzeptiert, 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!

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

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.

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.

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.

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.

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.

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 $\chi_M(x) = 0$.)

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 $\chi_M(x) = 0$.)

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

Fragen?!

Fragen bis hierhin?!

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.

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:

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.

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.

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.

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.

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:

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

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

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

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

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.

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

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?

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:

- 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:

- 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:

- 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+1} 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?

Mit einer Mehrband-NTM:

- Die NTM ist jetzt nichtdeterministisch in 2^{n+1} 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 nötigen Kanten zwischen den vermuteten Knoten vorhanden sind.
- Falls ja, akzeptiert sie, sonst nicht.

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 nötigen Kanten zwischen den vermuteten Knoten vorhanden sind.
- Falls ja, akzeptiert sie, sonst nicht.

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 nötigen Kanten zwischen den vermuteten Knoten vorhanden sind.
- Falls ja, akzeptiert sie, sonst nicht.

NTM $\rightarrow$ DTM

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 $\rightarrow$ DTM

Die Idee:

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

NTM $\to$ DTM

Die Idee:

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

NTM $\rightarrow$ 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 $\rightarrow$ DTM

Die Idee:

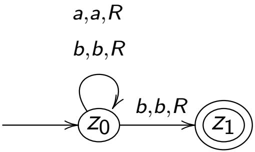
- Ist c_1 eine Konfiguration einer NTM, so kann diese mehrere (nichtdeterministische) Nachfolgekonfiguration haben.
- Dies sind aber endlich viele! Seien dies $c_{1,1}, c_{1,2}, \ldots, c_{1,r}$.
- Man kann sich dies als Baum vorstellen. c_1 als Wurzel. $c_{1,1}, \ldots, c_{1,r}$ als Kinder von c_1.
- 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

Die Idee:

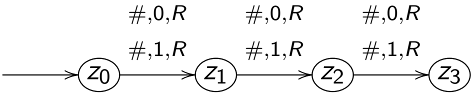
- Ist c_1 eine Konfiguration einer NTM, so kann diese mehrere (nichtdeterministische) Nachfolgekonfiguration haben.
- Dies sind aber endlich viele! Seien dies $c_{1,1}, c_{1,2}, \ldots, c_{1,r}$.
- Man kann sich dies als Baum vorstellen. c_1 als Wurzel. $c_{1,1}, \ldots, c_{1,r}$ als Kinder von c_1.
- 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



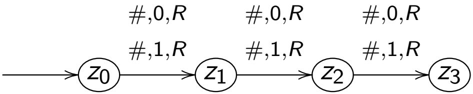
$$
\begin{array}{ccc}
z_0abab & & \\
\downarrow & & \\
az_0bab & \searrow & \\
\downarrow & & abz_1ab \\
abz_0ab & & \\
\downarrow & & \\
abaz_0b & \searrow & \\
\downarrow & & \\
ababz_0 & & ababz_1
\end{array}
$$

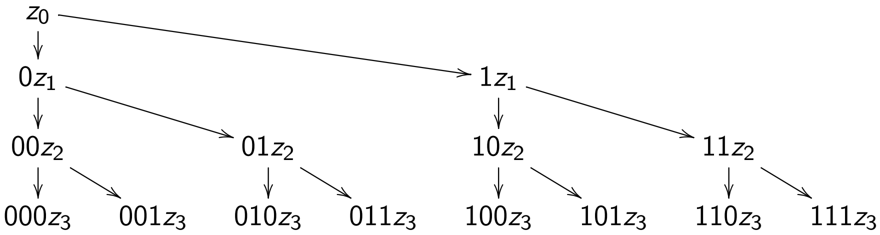
NTM → DTM


#,0,R
#,1,R
#,0,R
#,1,R
#,0,R
#,1,R
z_0
z_1
z_2
z_3

z_0
$0z_1$
$1z_1$
$00z_2$
$01z_2$
$10z_2$
$11z_2$
$000z_3$
$001z_3$
$010z_3$
$011z_3$
$100z_3$
$101z_3$
$110z_3$
$111z_3$

NTM → DTM


#,0,R
#,1,R
#,0,R
#,1,R
#,0,R
#,1,R
z0
z1
z2
z3


z0
0z1
1z1
00z2
01z2
10z2
11z2
000z3
001z3
010z3
011z3
100z3
101z3
110z3
111z3

NTM $\to$ 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 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 $\to$ 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

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 $c_1, \dots, c_r$ 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 $\rightarrow$ 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 $\rightarrow$ 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

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 $\rightarrow$ 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 $c_1, \dots, c_r$ 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 $\rightarrow$ 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!

NTM $\rightarrow$ 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!

NTM $\rightarrow$ DTM

Anmerkung

Wichtig ist, dass nachdem z.B. die Konfiguration c_1 durch ihre Nachfolgekonfigurationen $c_{1,1}, \dots, c_{1,r}$ 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!

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

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.*

Definition (Alternative Definition von RE)

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

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 $\chi_M(x) = 0$.)

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!

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!

Fragen

Sind alle regulären Sprachen entscheidbar?

1. Ja!
2. Nein!
3. Keine Teilmengenbeziehung!

Fragen

Sind alle kontextfreien Sprachen entscheidbar?

1. Ja!
2. Nein!
3. Keine Teilmengenbeziehung!

Fragen

Sind alle aufzählbaren Sprachen entscheidbar?

1. Ja!
2. Nein!
3. Keine Teilmengenbeziehung!

Fragen

Wenn eine Sprache entscheidbar ist, ist dann auch das Komplement entscheidbar?

1. Ja!
2. Nein!

Fragen

Wenn zwei Sprachen L_1 und L_2 entscheidbar sind, ist dann auch $L_1 \cap L_2$ entscheidbar?

1. Ja!
2. Nein!

Zur Nachbereitung

Zur Nachbereitung

1. Ja!
2. Ja! (Sehen wir noch genauer...)
3. Nein! (Sehen wir noch genauer...)
4. Ja!
5. Ja!

Fragen

Wenn eine Sprache aufzählbar ist, ist dann auch das Komplement aufzählbar?

1. Ja!
2. Nein!

Entscheidbar und Aufzählbar

Satz

Eine Sprache L ist entscheidbar genau dann, wenn L und $\overline{L}$ (das Komplement von L) aufzählbar sind.

Beweis.

(⇒) Ist L entscheidbar, dann ist L auch aufzählbar. Dreht man zudem die Ergebnisse einer TM, die L entscheidet, um, hat man eine TM, die $\overline{L}$ entscheidet. Damit ist $\overline{L}$ dann auch aufzählbar. □

Entscheidbar und Aufzählbar

Satz

Eine Sprache L ist entscheidbar genau dann, wenn L und $\overline{L}$ (das Komplement von L) aufzählbar sind.

Beweis.

(⇒) Ist L entscheidbar, dann ist L auch aufzählbar. Dreht man zudem die Ergebnisse einer TM, die L entscheidet, um, hat man eine TM, die $\overline{L}$ entscheidet. Damit ist $\overline{L}$ dann auch aufzählbar. □

Entscheidbar und Aufzählbar

Satz

Eine Sprache L ist entscheidbar genau dann, wenn L und $\overline{L}$ (das Komplement von L) aufzählbar sind.

Beweis.

($\Leftarrow$) Seien A und $\overline{A}$ TMs, die L bzw. $\overline{L}$ akzeptieren. Eine TM B, die L entscheidet arbeitet wie folgt:

1. Bei Eingabe von w
2. Lasse A einen Schritt arbeiten
3. Dann $\overline{A}$ einen Schritt
4. Wiederhole Schritte 2 und 3 bis eine akzeptiert.
5. Hat A akzeptiert, akzeptiere. Hat $\overline{A}$ akzeptiert, lehne ab.

□

Entscheidbar und Aufzählbar

Satz

Eine Sprache L ist entscheidbar genau dann, wenn L und $\overline{L}$ (das Komplement von L) aufzählbar sind.

Beweis.

($\Leftarrow$) Seien A und $\overline{A}$ TMs, die L bzw. $\overline{L}$ akzeptieren. Eine TM B, die L entscheidet arbeitet wie folgt:

1. Bei Eingabe von w
2. Lasse A einen Schritt arbeiten
3. Dann $\overline{A}$ einen Schritt
4. Wiederhole Schritte 2 und 3 bis eine akzeptiert.
5. Hat A akzeptiert, akzeptiere. Hat $\overline{A}$ akzeptiert, lehne ab.

□

Entscheidbar und Aufzählbar

Satz

Eine Sprache L ist entscheidbar genau dann, wenn L und $\overline{L}$ (das Komplement von L) aufzählbar sind.

Beweis.

($\Leftarrow$) Seien A und $\overline{A}$ TMs, die L bzw. $\overline{L}$ akzeptieren. Eine TM B, die L entscheidet arbeitet wie folgt:

1. Bei Eingabe von w
2. Lasse A einen Schritt arbeiten
3. Dann $\overline{A}$ einen Schritt
4. Wiederhole Schritte 2 und 3 bis eine akzeptiert.
5. Hat A akzeptiert, akzeptiere. Hat $\overline{A}$ akzeptiert, lehne ab.

$\square$

Entscheidbar und Aufzählbar

Satz

Eine Sprache L ist entscheidbar genau dann, wenn L und $\overline{L}$ (das Komplement von L) aufzählbar sind.

Beweis.

($\Leftarrow$) Seien A und $\overline{A}$ TMs, die L bzw. $\overline{L}$ akzeptieren. Eine TM B, die L entscheidet arbeitet wie folgt:

1. Bei Eingabe von w
2. Lasse A einen Schritt arbeiten
3. Dann $\overline{A}$ einen Schritt
4. Wiederhole Schritte 2 und 3 bis eine akzeptiert.
5. Hat A akzeptiert, akzeptiere. Hat $\overline{A}$ akzeptiert, lehne ab.

□

Entscheidbar und Aufzählbar

Satz

Eine Sprache L ist entscheidbar genau dann, wenn L und $\overline{L}$ (das Komplement von L) aufzählbar sind.

Beweis.

($\Leftarrow$) Seien A und $\overline{A}$ TMs, die L bzw. $\overline{L}$ akzeptieren. Eine TM B, die L entscheidet arbeitet wie folgt:

1. Bei Eingabe von w
2. Lasse A einen Schritt arbeiten
3. Dann $\overline{A}$ einen Schritt
4. Wiederhole Schritte 2 und 3 bis eine akzeptiert.
5. Hat A akzeptiert, akzeptiere. Hat $\overline{A}$ akzeptiert, lehne ab.

$\square$

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.