

7c

Turingmaschinen

Prof. Dr. Jasmin Blanchette

Lehr- und Forschungseinheit für
Theoretische Informatik und Theorembeweisen

Stand: 10. Mai 2024

Basiert auf Folien von PD Dr. David Sabel



Wiederholung: Typ 1- und Typ 0-Grammatiken

- ▶ Im Gegensatz zu Typ 2 ist die Linke Seite von Typ 0- und Typ 1-Produktionen $\ell \in (V \cup \Sigma)^+$.
- ▶ Mit Typ 1-Grammatiken gilt $|\ell| \leq |r|$ für alle Produktionen $\ell \rightarrow r$.
- ▶ Typ 1-Grammatiken nennen wir auch **kontextsensitive Grammatiken**.

Hintergrund zu Turingmaschinen

Nun führen wir ein Maschinenmodell ein, das zu Typ 1 und zu Typ 0 passt:
die [Turingmaschinen](#) (für Typ 1: mit Einschränkungen).

Hintergrund zu Turingmaschinen

Nun führen wir ein Maschinenmodell ein, das zu Typ 1 und zu Typ 0 passt: die **Turingmaschinen** (für Typ 1: mit Einschränkungen).

Einschränkungen der Kellerautomaten:

- ▶ PDAs erkennen genau die **kontextfreien Sprachen**, daher müssen Automaten für Typ 1- und Typ 0-Sprachen „mehr können“.
- ▶ Wesentliche Beschränkung bei PDAs:
Der Zugriff auf den Speicher ist **nur von oben möglich**.

Hintergrund zu Turingmaschinen

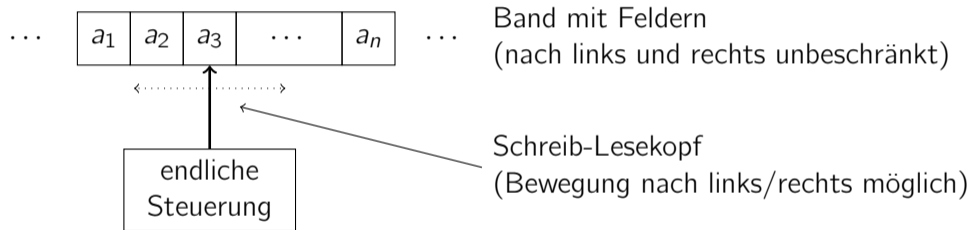
Nun führen wir ein Maschinenmodell ein, das zu Typ 1 und zu Typ 0 passt: die **Turingmaschinen** (für Typ 1: mit Einschränkungen).

Einschränkungen der Kellerautomaten:

- ▶ PDAs erkennen genau die **kontextfreien Sprachen**, daher müssen Automaten für Typ 1- und Typ 0-Sprachen „mehr können“.
- ▶ Wesentliche Beschränkung bei PDAs:
Der Zugriff auf den Speicher ist **nur von oben möglich**.
- ▶ Z.B. kann man $\{a^i b^i c^i \mid i \in \mathbb{N}_{>0}\}$ nicht mit einem PDA erkennen, da man die Anzahl i
 - ▶ beim Lesen der a 's im Keller speichert
 - ▶ beim Lesen der b 's vergleichen muss und das geht nur durch sukzessives Entnehmen aus dem Keller
 - ▶ beim Lesen der c 's nicht mehr hat.

Mit **beliebigem Lesen** des Speichers wäre es kein Problem, $a^i b^i c^i$ zu erkennen.

Illustration einer Turingmaschine



Definition einer Turingmaschine

Definition

Eine **Turingmaschine** (TM) ist ein 7-Tupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$, wobei:

- ▶ Z ist eine endliche Menge von **Zuständen**
- ▶ Σ ist das (endliche) **Eingabealphabet**
- ▶ $\Gamma \supset \Sigma$ ist das (endliche) **Bandalphabet**
- ▶ δ ist die **Überföhrungsfunktion**
 - ▶ für **deterministische TM** (DTM): $\delta : (Z \setminus E) \times \Gamma \rightarrow Z \times \Gamma \times \{L, R, N\}$
 - ▶ für **nichtdeterministische TM** (NTM): $\delta : (Z \setminus E) \times \Gamma \rightarrow \mathcal{P}(Z \times \Gamma \times \{L, R, N\})$
- ▶ $z_0 \in Z$ ist der **Startzustand**
- ▶ $\square \in \Gamma \setminus \Sigma$ ist das **Blank-Symbol**
- ▶ $E \subseteq Z$ ist die Menge der **Endzustände**.

Zustandsübergang

Für deterministische Turingmaschinen:

- ▶ Ein Eintrag $\delta(z, a) = (z', b, x)$ bedeutet: Falls die TM im Zustand z ist und das Zeichen a an der aktuellen Position des Schreib-Lesekopfs ist, dann
 1. Wechsle in Zustand z' .
 2. Ersetze a durch b auf dem Band.
 3. Falls $x = L$: Verschiebe den Schreib-Lesekopf ein Position nach links.
 4. Falls $x = R$: Verschiebe den Schreib-Lesekopf ein Position nach rechts.
 5. Falls $x = N$: Lasse Schreib-Lesekopf unverändert (Neutral).

Zustandsübergang

Für deterministische Turingmaschinen:

- ▶ Ein Eintrag $\delta(z, a) = (z', b, x)$ bedeutet: Falls die TM im Zustand z ist und das Zeichen a an der aktuellen Position des Schreib-Lesekopfs ist, dann
 1. Wechsle in Zustand z' .
 2. Ersetze a durch b auf dem Band.
 3. Falls $x = L$: Verschiebe den Schreib-Lesekopf ein Position nach links.
 4. Falls $x = R$: Verschiebe den Schreib-Lesekopf ein Position nach rechts.
 5. Falls $x = N$: Lasse Schreib-Lesekopf unverändert (Neutral).

Für nichtdeterministische Turingmaschinen:

- ▶ $\delta(z, a)$ ist eine Menge solcher möglichen Schritte und die NTM macht in einem Lauf irgendeinen davon (nichtdeterministisch).

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine Turingmaschine.

Eine **Konfiguration** von M ist ein Wort $wzw' \in \Gamma^*Z\Gamma^*$, wobei:

- ▶ z ist der aktuelle Zustand von M
- ▶ $\dots \square \square ww' \square \square \dots$ steht auf dem Band
- ▶ der Schreib-Lesekopf steht auf dem ersten Symbol von w' .

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine Turingmaschine.

Eine **Konfiguration** von M ist ein Wort $wzw' \in \Gamma^*Z\Gamma^*$, wobei:

- ▶ z ist der aktuelle Zustand von M
- ▶ $\dots \square \square ww' \square \square \dots$ steht auf dem Band
- ▶ der Schreib-Lesekopf steht auf dem ersten Symbol von w' .

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine Turingmaschine.

Für ein Eingabewort w ist die **Startkonfiguration** $Start_M(w)$ von M das Wort z_0w .

Im Spezialfall $w = \varepsilon$ ist die Startkonfiguration $z_0\square$.

D.h. am Anfang steht der Schreib-Lesekopf auf dem ersten Symbol der Eingabe.

Übergangsrelation einer deterministische Turingmaschine

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine deterministische Turingmaschine.

Die Relation $\vdash_M \subseteq \Gamma^* Z \Gamma^* \times \Gamma^* Z \Gamma^*$ ist definiert durch

- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_m z' c a_2 \cdots a_n$,
wenn $\delta(z, a_1) = (z', c, N)$, $m \geq 0$, $n \geq 1$ und $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_{m-1} z' b_m c a_2 \cdots a_n$,
wenn $\delta(z, a_1) = (z', c, L)$, $m \geq 1$, $n \geq 1$ und $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_m c z' a_2 \cdots a_n$,
wenn $\delta(z, a_1) = (z', c, R)$, $m \geq 0$, $n \geq 2$ und $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \vdash_M b_1 \cdots b_m c z' \square$, wenn $\delta(z, a_1) = (z', c, R)$, $m \geq 0$ und $z \notin E$
- ▶ $z a_1 \cdots a_n \vdash_M z' \square c a_2 \cdots a_n$, wenn $\delta(z, a_1) = (z', c, L)$, $n \geq 1$ und $z \notin E$.

Übergangsrelation einer nichtdeterministische Turingmaschine

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine nichtdeterministische Turingmaschine.

Die Relation $\vdash_M \subseteq \Gamma^* Z \Gamma^* \times \Gamma^* Z \Gamma^*$ ist definiert durch

- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_m z' c a_2 \cdots a_n$,
wenn $(z', c, N) \in \delta(z, a_1)$, $m \geq 0$, $n \geq 1$ und $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_{m-1} z' b_m c a_2 \cdots a_n$,
wenn $(z', c, L) \in \delta(z, a_1)$, $m \geq 1$, $n \geq 1$ und $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_m c z' a_2 \cdots a_n$,
wenn $(z', c, R) \in \delta(z, a_1)$, $m \geq 0$, $n \geq 2$ und $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \vdash_M b_1 \cdots b_m c z' \square$, wenn $(z', c, R) \in \delta(z, a_1)$, $m \geq 0$ und $z \notin E$
- ▶ $z a_1 \cdots a_n \vdash_M z' \square c a_2 \cdots a_n$, wenn $(z', c, L) \in \delta(z, a_1)$, $n \geq 1$ und $z \notin E$.

Übergangsrelation einer Turingmaschine

Weitere Notationen:

- ▶ \vdash_M^* ist die reflexiv-transitive Hülle von \vdash_M .
- ▶ \vdash_M^i ist die i -fache Anwendung von \vdash_M .
- ▶ Wenn M eindeutig ist, schreiben wir \vdash statt \vdash_M .

Übergangsrelation einer Turingmaschine

Weitere Notationen:

- ▶ \vdash_M^* ist die reflexiv-transitive Hülle von \vdash_M .
- ▶ \vdash_M^i ist die i -fache Anwendung von \vdash_M .
- ▶ Wenn M eindeutig ist, schreiben wir \vdash statt \vdash_M .

Bemerkung:

- ▶ Mit unserer Definition **hält** die Turingmaschine **an**, sobald sie einen Endzustand erreicht hat.
(Das Buch von Schöning erlaubt weiterrechnen.)

Akzeptierte Sprache einer Turingmaschine

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine Turingmaschine.

Die von M akzeptierte Sprache $L(M)$ ist definiert als

$$L(M) := \{w \in \Sigma^* \mid \text{Start}_M(w) \vdash_M^* uzv \text{ f\"ur } u, v \in \Gamma^*, z \in E\}$$

Akzeptierte Sprache einer Turingmaschine

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine Turingmaschine.

Die von M **akzeptierte Sprache** $L(M)$ ist definiert als

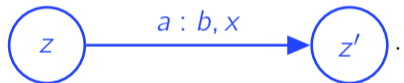
$$L(M) := \{w \in \Sigma^* \mid \text{Start}_M(w) \vdash_M^* uzv \text{ f\"ur } u, v \in \Gamma^*, z \in E\}$$

Weitere Begriffe:

- ▶ Die TM **akzeptiert** heißt, die TM erreicht einen Endzustand.
- ▶ Die TM **verwirft** heißt, die TM erreicht keinen Endzustand.

Notation als Zustandsgraph

- ▶ Die Darstellung ist ähnlich der von DFAs, NFAs und PDAs.
- ▶ Für $\delta(z, a) = (z', b, x)$ bzw. $(z', b, x) \in \delta(z, a)$ zeichnen wir



- ▶ Beachte, dass das Blank-Symbol bekannt sein muss (üblicherweise \square).

Beispiel für eine deterministische Turingmaschine

DTM $M = (\{z_0, z_1, z_2, z_3\}, \{0, 1\}, \{0, 1, \square\}, \delta, z_0, \square, \{z_3\})$ mit

$$\begin{array}{lll} \delta(z_0, 0) = (z_0, 0, R) & \delta(z_0, 1) = (z_0, 1, R) & \delta(z_0, \square) = (z_1, \square, L) \\ \delta(z_1, 0) = (z_2, 1, L) & \delta(z_1, 1) = (z_1, 0, L) & \delta(z_1, \square) = (z_3, 1, N) \\ \delta(z_2, 0) = (z_2, 0, L) & \delta(z_2, 1) = (z_2, 1, L) & \delta(z_2, \square) = (z_3, \square, R) \end{array}$$

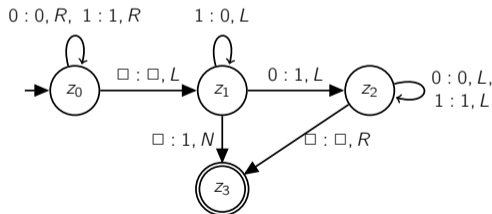
turingmachinesimulator.com/shared/istktezldr

Beispiel für eine deterministische Turingmaschine

DTM $M = (\{z_0, z_1, z_2, z_3\}, \{0, 1\}, \{0, 1, \square\}, \delta, z_0, \square, \{z_3\})$ mit

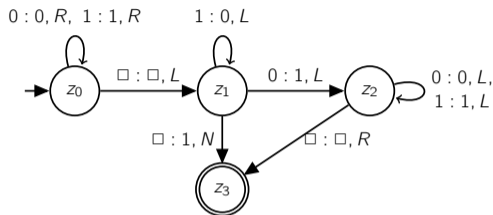
$$\begin{array}{lll} \delta(z_0, 0) = (z_0, 0, R) & \delta(z_0, 1) = (z_0, 1, R) & \delta(z_0, \square) = (z_1, \square, L) \\ \delta(z_1, 0) = (z_2, 1, L) & \delta(z_1, 1) = (z_1, 0, L) & \delta(z_1, \square) = (z_3, 1, N) \\ \delta(z_2, 0) = (z_2, 0, L) & \delta(z_2, 1) = (z_2, 1, L) & \delta(z_2, \square) = (z_3, \square, R) \end{array}$$

Zustandsgraph zu M :



turingmachinesimulator.com/shared/istktezldr

Beispiel für eine deterministische Turingmaschine



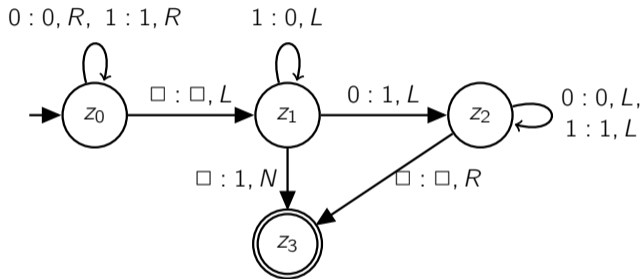
M interpretiert die Eingabe $w \in \{0, 1\}^*$ als Binärzahl und addiert 1:

- ▶ In z_0 wird das rechte Ende gesucht, dann in z_1 gewechselt.
- ▶ In z_1 wird versucht 1 zur aktuellen Ziffer hinzu zu addieren:
Gelingt das ohne Übertrag, dann wird in z_2 gewechselt.
Bei Übertrag: Weitermachen in z_1 und +1 zur nächsten Ziffer links.
- ▶ In z_2 wird bis zum Anfang links gelaufen, dann in z_3 gewechselt.
- ▶ In z_3 wird akzeptiert.

Beispiel für einen akzeptierenden Lauf

DTM M :

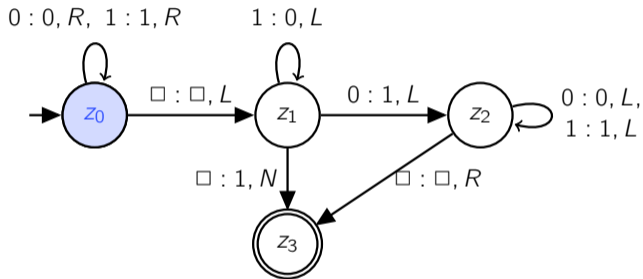
Eingabe: 0011



Beispiel für einen akzeptierenden Lauf

DTM M :

Eingabe: 0011

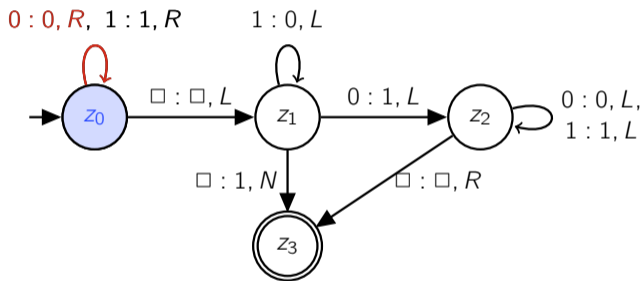


z_0 0011

Beispiel für einen akzeptierenden Lauf

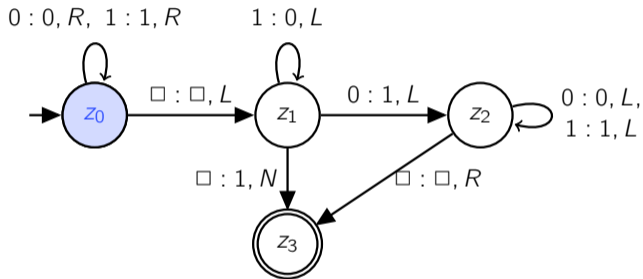
DTM M :

Eingabe: 0011



Beispiel für einen akzeptierenden Lauf

DTM M :

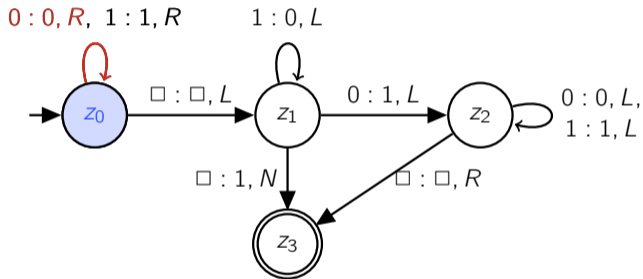


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$

Beispiel für einen akzeptierenden Lauf

DTM M :

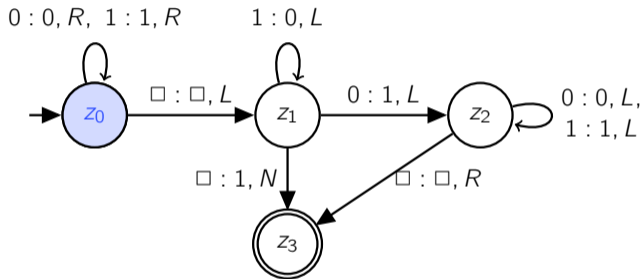


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$

Beispiel für einen akzeptierenden Lauf

DTM M :

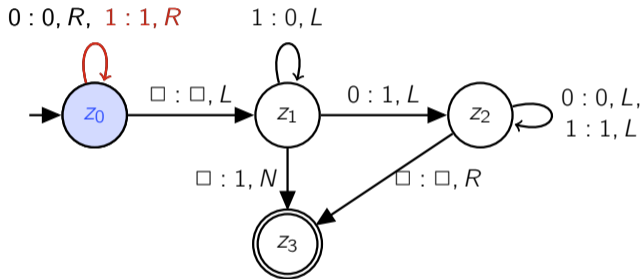


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$

Beispiel für einen akzeptierenden Lauf

DTM M :

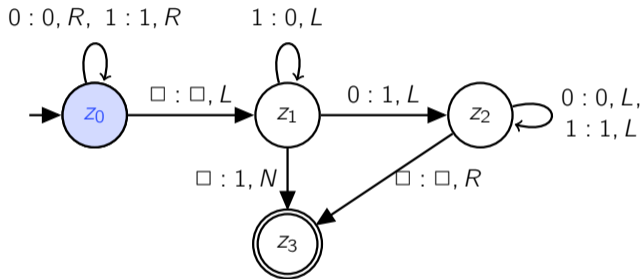


Eingabe: 0011

z_0 0011
 \vdash 0 z_0 011
 \vdash 00 z_0 11

Beispiel für einen akzeptierenden Lauf

DTM M :

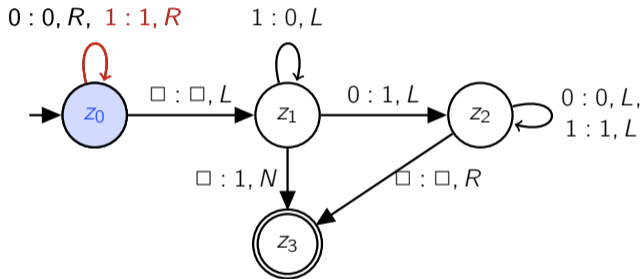


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$

Beispiel für einen akzeptierenden Lauf

DTM M :

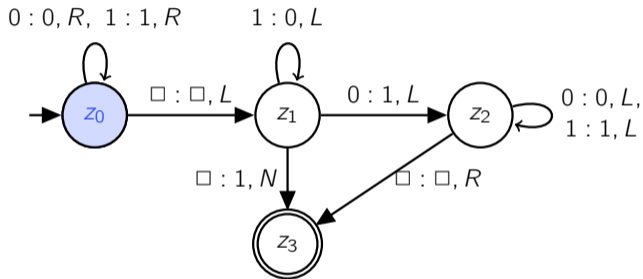


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$

Beispiel für einen akzeptierenden Lauf

DTM M :

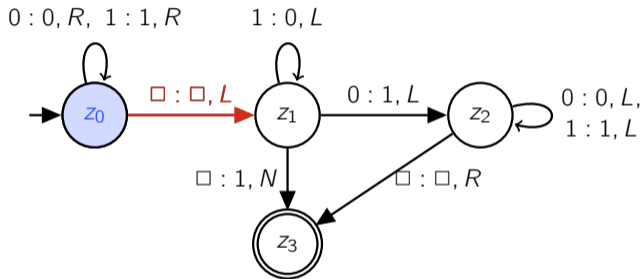


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$
 $\vdash 0011z_0 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

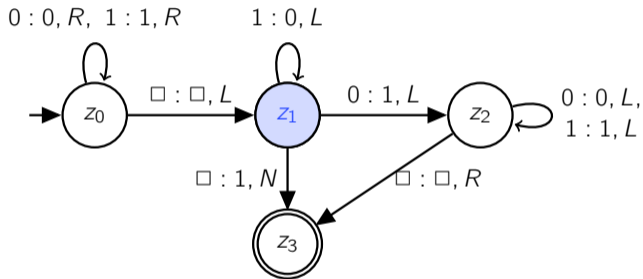


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$
 $\vdash 0011z_0 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

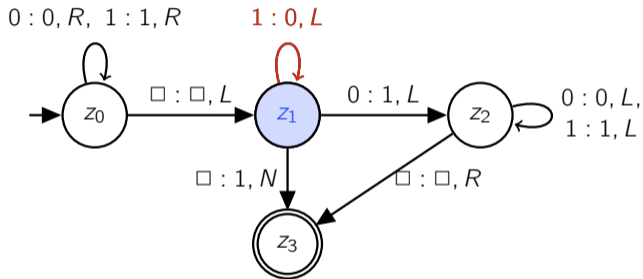


Eingabe: 0011

- $z_0 0011$
- $\vdash 0z_0 011$
- $\vdash 00z_0 11$
- $\vdash 001z_0 1$
- $\vdash 0011z_0 \square$
- $\vdash 001z_1 1 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

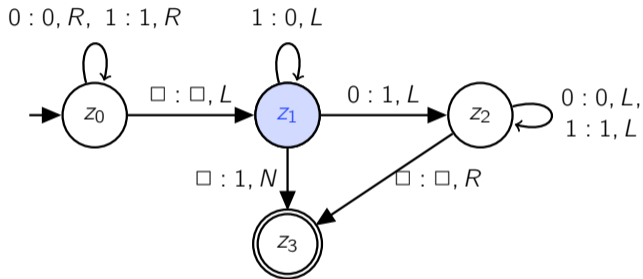


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$
 $\vdash 0011z_0 \square$
 $\vdash 001z_1 1 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

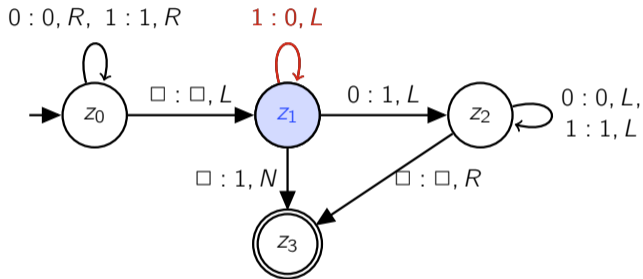


Eingabe: 0011

z₀0011
⊢ 0z₀011
⊢ 00z₀11
⊢ 001z₀1
⊢ 0011z₀□
⊢ 001z₁1□
⊢ 00z₁10□

Beispiel für einen akzeptierenden Lauf

DTM M :

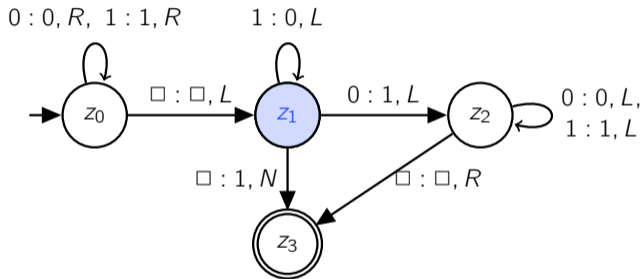


Eingabe: 0011

z_00011
 $\vdash 0z_0011$
 $\vdash 00z_011$
 $\vdash 001z_01$
 $\vdash 0011z_0\square$
 $\vdash 001z_11\square$
 $\vdash 00z_110\square$

Beispiel für einen akzeptierenden Lauf

DTM M :

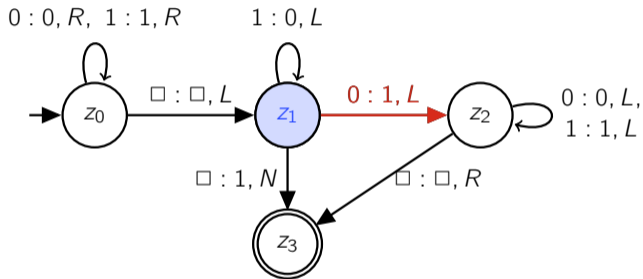


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$
 $\vdash 0011z_0 \square$
 $\vdash 001z_1 1 \square$
 $\vdash 00z_1 10 \square$
 $\vdash 0z_1 000 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

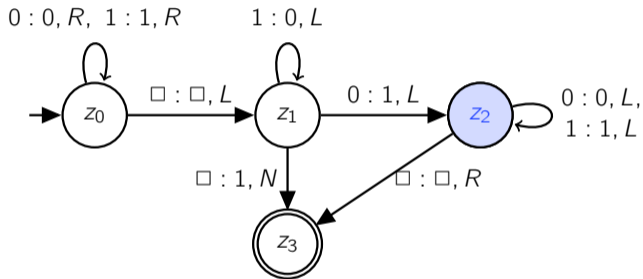


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$
 $\vdash 0011z_0 \square$
 $\vdash 001z_1 1 \square$
 $\vdash 00z_1 10 \square$
 $\vdash 0z_1 000 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

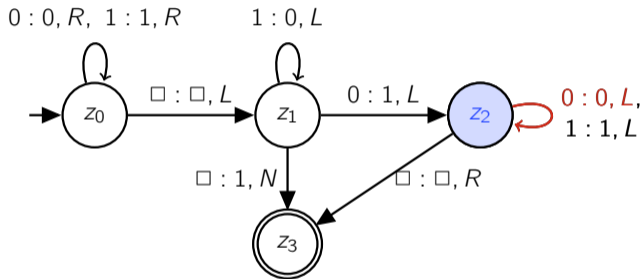


Eingabe: 0011

$z_0 0011$
⊢ $0z_0 011$
⊢ $00z_0 11$
⊢ $001z_0 1$
⊢ $0011z_0 \square$
⊢ $001z_1 1 \square$
⊢ $00z_1 10 \square$
⊢ $0z_1 000 \square$
⊢ $z_2 0100 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

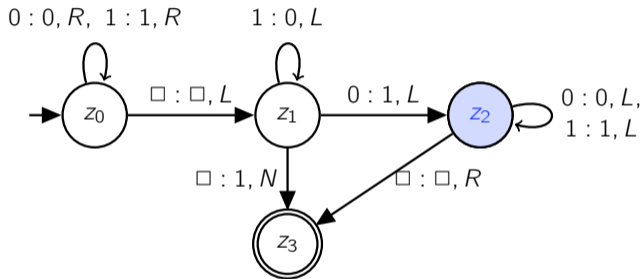


Eingabe: 0011

$z_0 0011$
⊢ $0z_0 011$
⊢ $00z_0 11$
⊢ $001z_0 1$
⊢ $0011z_0 \square$
⊢ $001z_1 1 \square$
⊢ $00z_1 10 \square$
⊢ $0z_1 000 \square$
⊢ $z_2 0100 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

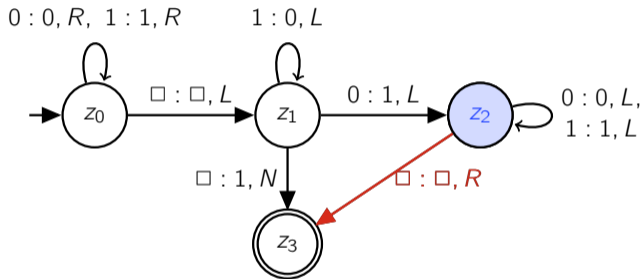


Eingabe: 0011

$z_0 0011$
⊢ $0z_0 011$
⊢ $00z_0 11$
⊢ $001z_0 1$
⊢ $0011z_0 \square$
⊢ $001z_1 1 \square$
⊢ $00z_1 10 \square$
⊢ $0z_1 000 \square$
⊢ $z_2 0100 \square$
⊢ $z_2 \square 0100 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

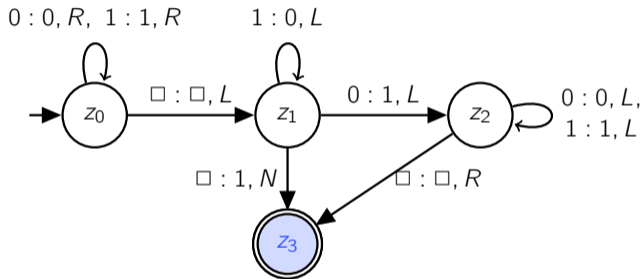


Eingabe: 0011

$z_0 0011$
⊢ $0z_0 011$
⊢ $00z_0 11$
⊢ $001z_0 1$
⊢ $0011z_0 \square$
⊢ $001z_1 1 \square$
⊢ $00z_1 10 \square$
⊢ $0z_1 000 \square$
⊢ $z_2 0100 \square$
⊢ $z_2 \square 0100 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :

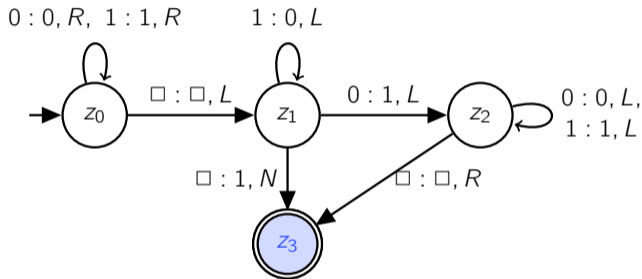


Eingabe: 0011

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$
 $\vdash 0011z_0 \square$
 $\vdash 001z_1 1 \square$
 $\vdash 00z_1 10 \square$
 $\vdash 0z_1 000 \square$
 $\vdash z_2 0100 \square$
 $\vdash z_2 \square 0100 \square$
 $\vdash \square z_3 0100 \square$

Beispiel für einen akzeptierenden Lauf

DTM M :



Eingabe: 0011
 $\in L(M)$

$z_0 0011$
 $\vdash 0z_0 011$
 $\vdash 00z_0 11$
 $\vdash 001z_0 1$
 $\vdash 0011z_0 \square$
 $\vdash 001z_1 1 \square$
 $\vdash 00z_1 10 \square$
 $\vdash 0z_1 000 \square$
 $\vdash z_2 0100 \square$
 $\vdash z_2 \square 0100 \square$
 $\vdash \square z_3 0100 \square$

Linear beschränkte Turingmaschinen

Grundgedanke:

- ▶ **Linear beschränkte Turingmaschinen** (*linear bounded automata*, **LBAs**) sind spezielle Turingmaschinen.
- ▶ Der Schreib-Lesekopf darf den **Bereich der Eingabe** auf dem Band **nicht verlassen**.
- ▶ Zum Erkennen des Anfangs und des Endes wird die Eingabe in **spitzen Klammern** gesetzt: Statt w ist die Eingabe nun $\langle w \rangle$.

Definition eines LBA

Definition

Eine **linear beschränkte Turingmaschine** (*linear bounded automaton*, **LBA**) ist ein 8-Tupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \langle, \rangle, E)$, wobei:

- ▶ $Z, \Sigma, \Gamma, \delta, z_0$ und E sind wie bei nichtdeterministischen Turingmaschinen
- ▶ $\langle, \rangle \in \Sigma$ sind **Start-** bzw. **Endmarker**
- ▶ δ überschreibt keinen der Marker
- ▶ bei \langle gibt δ nie L aus und bei \rangle gibt δ nie R aus.

Definition eines LBA

Definition

Eine **linear beschränkte Turingmaschine** (*linear bounded automaton*, **LBA**) ist ein 8-Tupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \langle, \rangle, E)$, wobei:

- ▶ $Z, \Sigma, \Gamma, \delta, z_0$ und E sind wie bei nichtdeterministischen Turingmaschinen
- ▶ $\langle, \rangle \in \Sigma$ sind **Start-** bzw. **Endmarker**
- ▶ δ überschreibt keinen der Marker
- ▶ bei \langle gibt δ nie L aus und bei \rangle gibt δ nie R aus.

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \langle, \rangle, E)$ ein LBA.

Die von M **akzeptierte Sprache** $L(M)$ ist definiert als

$$L(M) := \{w \in (\Sigma - \{\langle, \rangle\})^* \mid z_0 \langle w \rangle \vdash_M^* uzv \text{ für } u, v \in \Gamma^*, z \in E\}$$

Theorem (Satz von Kuroda)

Die LBAs akzeptieren genau die kontextsensitiven Sprachen.

Satz

Die nichtdeterministischen Turingmaschinen akzeptierten genau die Typ 0-Sprachen.

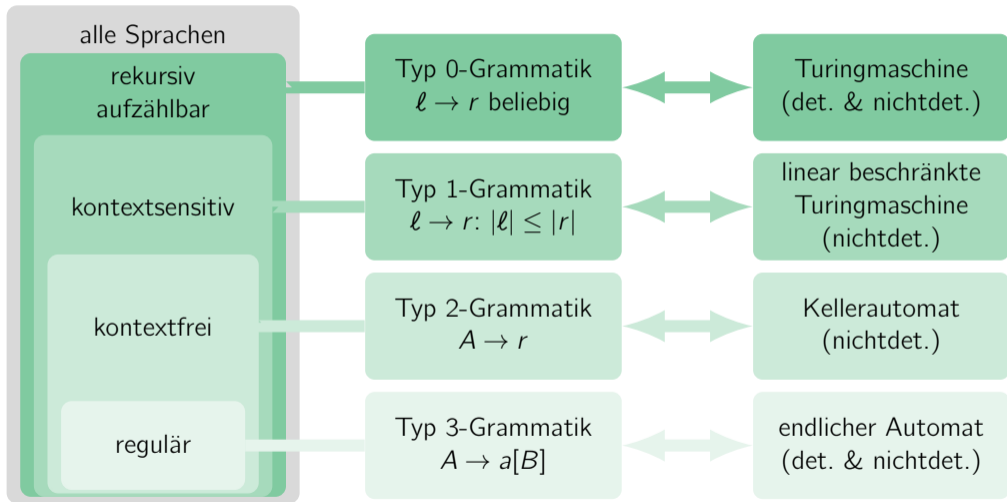
Beweis Nächste Vorlesung (nur FSK).



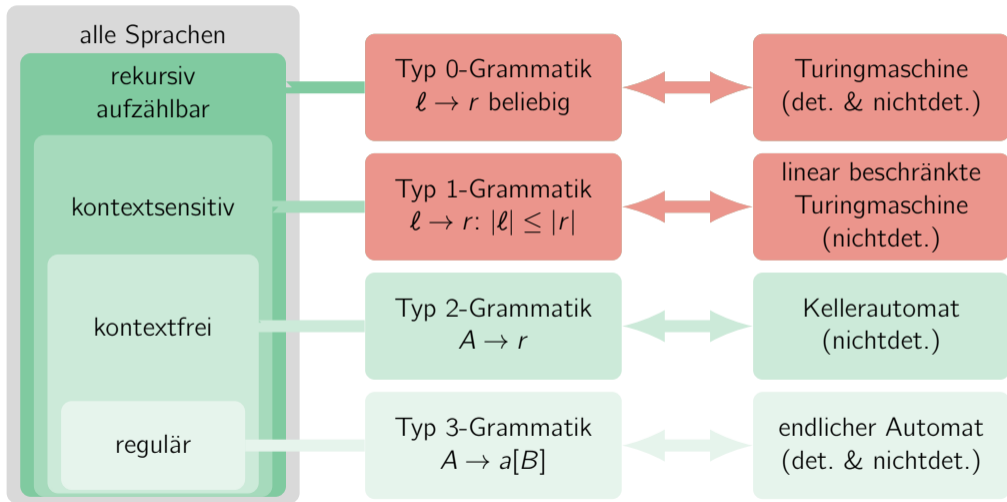
Deterministische vs. nichtdeterministische Turingmaschinen

- ▶ Nichtdeterministische Turingmaschinen können durch deterministische Turingmaschinen simuliert werden (mithilfe von Dovetailing).
- ▶ Daher gilt der letzte Satz auch für deterministische Turingmaschinen.
- ▶ Der Unterschied zwischen NTMs und DTMs kommt erst zum Tragen, wenn wir das Laufzeitverhalten betrachten (im Kursteil zur Komplexitätstheorie).

Überblick über Grammatiken und Maschinenmodelle



Überblick über Grammatiken und Maschinenmodelle



Trennende Beispiele

- ▶ Die Sprache $\{a^n b^n \mid n \in \mathbb{N}\}$ ist vom Typ 2 aber nicht vom Typ 3.
- ▶ Die Sprache $\{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist vom Typ 1 aber nicht vom Typ 2.
- ▶ Die Sprache

$$H = \{M\#w \mid \text{die durch } M \text{ beschriebene} \\ \text{Turingmaschine hält bei Eingabe } w\}$$

ist vom Typ 0 aber nicht vom Typ 1.

(Die Sprache H ist das Halteproblem, welches wir später noch genauer betrachten und erläutern.)

- ▶ Das Komplement von H ist nicht vom Typ 0.

Deterministisch vs. nichtdeterministisch

Deterministischer Automat	Nichtdeterministischer Automat	Äquivalent?
DFA	NFA	ja
DPDA	PDA	nein
DLBA	LBA	unbekannt
DTM	NTM	ja

Abschlusseigenschaften

Sprachklasse	Schnitt	Vereinigung	Komplement	Produkt	Kleenescher Abschluss
Typ 3	ja	ja	ja	ja	ja
Det. kontextfrei	nein	nein	ja	nein	nein
Typ 2	nein	ja	nein	ja	ja
Typ 1	ja	ja	ja	ja	ja
Typ 0	ja	ja	nein	ja	ja

Entscheidbarkeiten

Sprachklasse	Wortproblem	Leerheitsproblem	Äquivalenzproblem	Schnittproblem
Typ 3	ja	ja	ja	ja
Det. kontextfrei	ja	ja	ja	nein
Typ 2	ja	ja	nein	nein
Typ 1	ja	nein	nein	nein
Typ 0	nein	nein	nein	nein

Komplexität des Wortproblems

Sprachklasse	Komplexität
Typ 3, DFA gegeben	lineare Komplexität
deterministisch kontextfrei, DPDA gegeben	lineare Komplexität
Typ 2, Chomsky-Normalform gegeben	$O(n^3)$
Typ 1	exponentiell (nächste Vorlesung, FSK)
Typ 0	unlösbar