

Automatentheorie und Formale Sprachen

für die Studiengänge

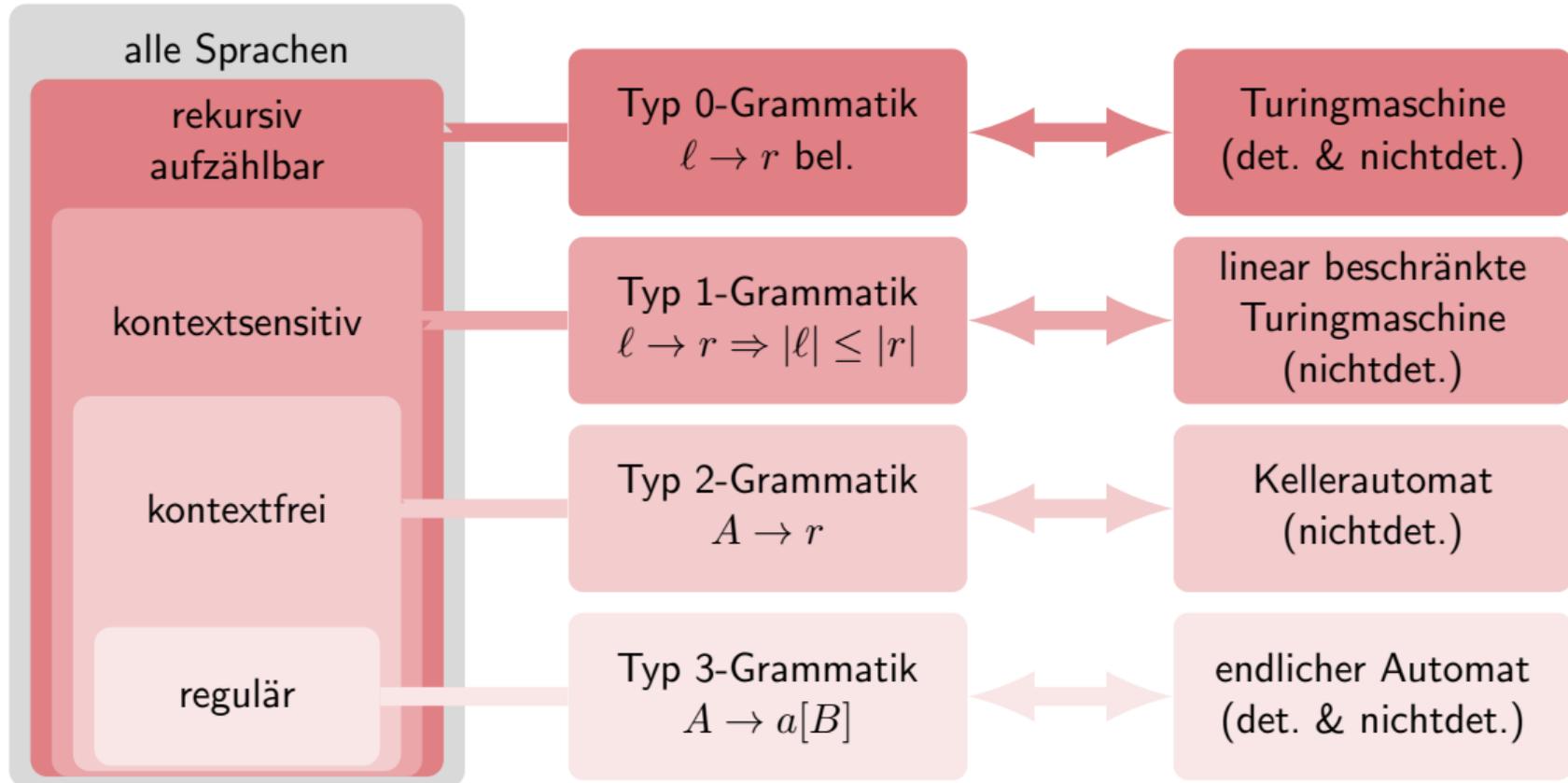
- Angewandte Informatik
- Informatik - Technische Systeme

08 Typ 1 und Typ 0 Sprachen

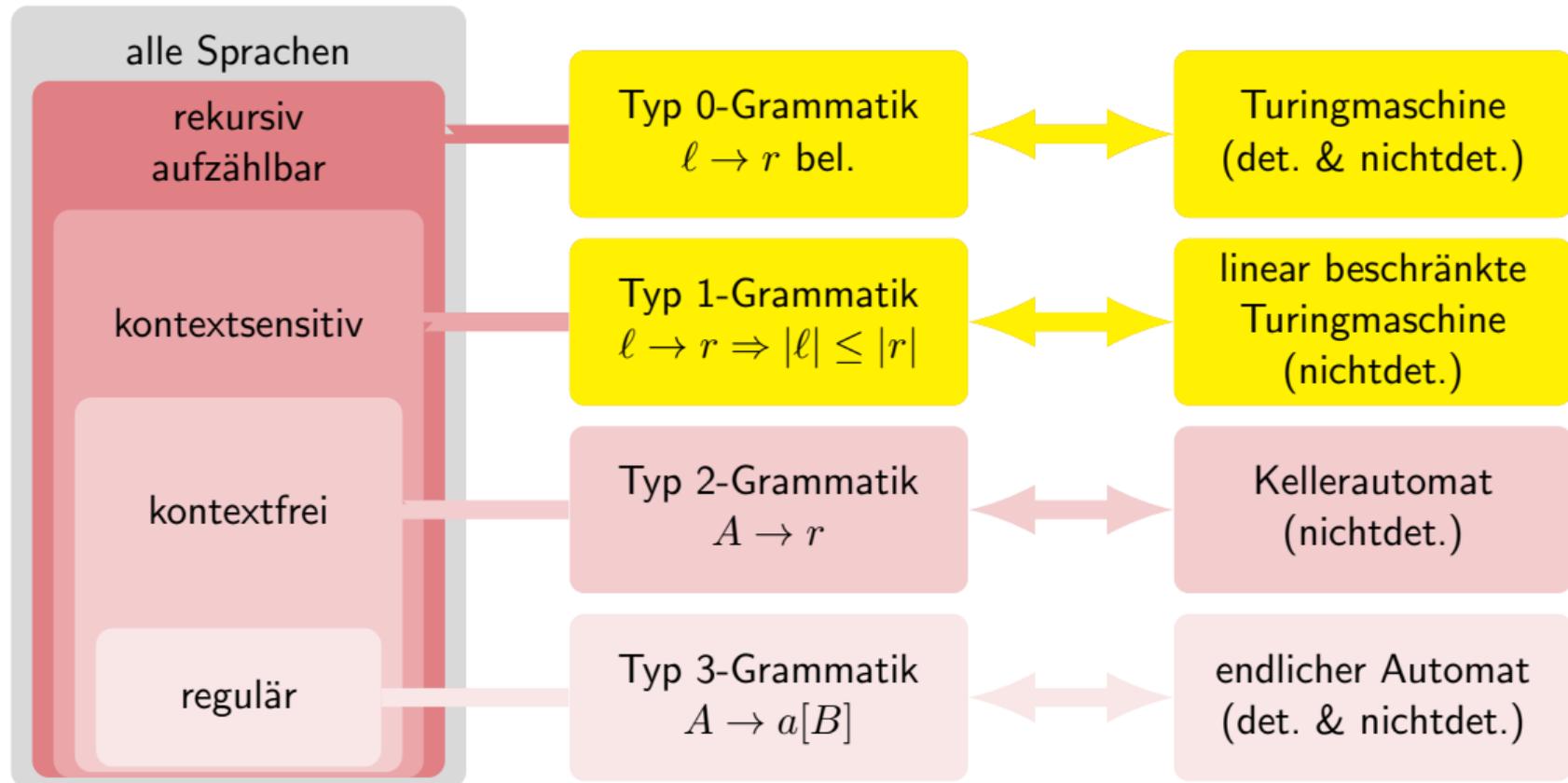
Prof. Dr. David Sabel
Sommersemester 2025

Stand der Folien: 11. Juni 2025

Grammatiken & Maschinen für die Chomsky-Hierarchie

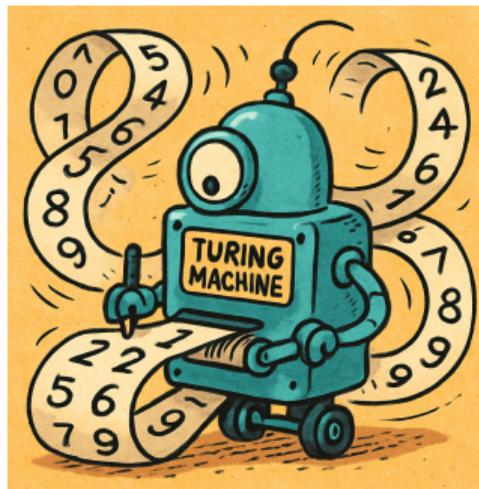


Grammatiken & Maschinen für die Chomsky-Hierarchie



Erinnerung: Typ 1- und Typ 0-Sprachen

- Typ 1: $|\ell| \leq |r|$ für alle Produktionen $\ell \rightarrow r$
- Typ 1-Grammatik = kontextsensitive Grammatik
- aber (im Gegensatz zu Typ 2): $\ell \in (\Sigma \cup V)^+$
- Typ 0: alles erlaubt
- In manchen Büchern werden unsere Typ 1-Grammatik auch **monotone Grammatiken** genannt
- In manchen Büchern wird für kontextsensitive Grammatiken gefordert: Produktionen von der Form $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \alpha_3 \alpha_2$ mit $\alpha_3 \neq \varepsilon$
- Nun: Maschinenmodell, das zu Typ 1 und zu Typ 0 passt: Turingmaschinen (für Typ 1: mit Einschränkungen).

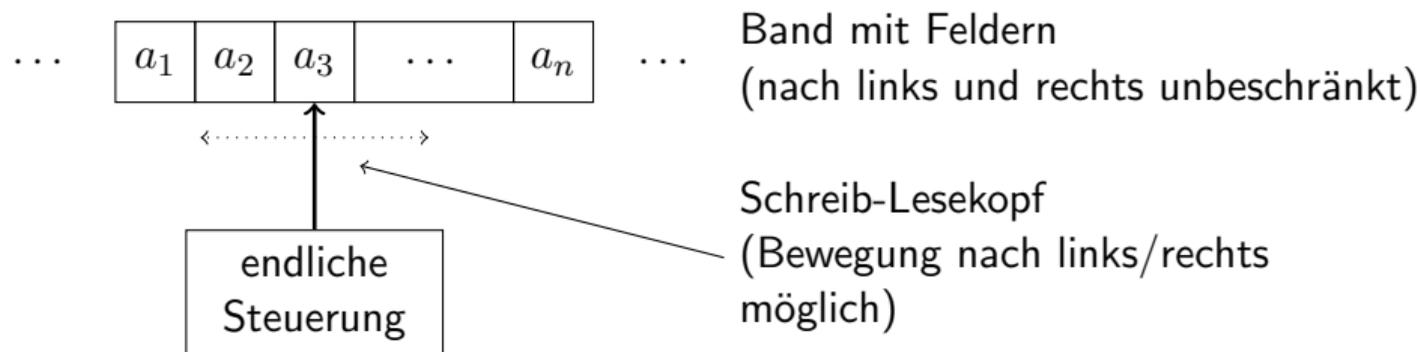


Einschränkungen der Kellerautomaten

- PDAs erkennen genau die CFLs, daher müssen Automaten für Typ 1- und Typ 0-Sprachen „mehr können“
- Wesentliche Beschränkung bei PDAs:
Zugriff auf Speicher nur von oben möglich
- Z.B. kann man $\{a^i b^i c^i \mid i \in \mathbb{N}\}$ nicht mit 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.

Turingmaschine: Illustration



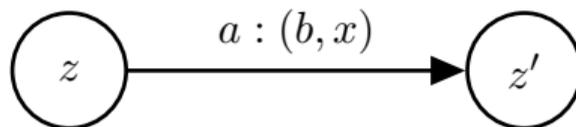
Definition (Turingmaschine)

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

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

Notation als Zustandsgraph

- Darstellung analog zu DFA / NFA / PDA
- Für $(z', b, x) \in \delta(z, a)$ zeichnen wir



DTM: 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

- Wechsle in Zustand z'
- Ersetze a durch b auf dem Band
- Falls $x = L$: Verschiebe den Schreib-Lesekopf ein Position nach links
- Falls $x = R$: Verschiebe den Schreib-Lesekopf ein Position nach rechts
- Falls $x = N$: Lasse Schreib-Lesekopf unverändert (Neutral)

DTM: 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

- Wechsle in Zustand z'
- Ersetze a durch b auf dem Band
- Falls $x = L$: Verschiebe den Schreib-Lesekopf ein Position nach links
- Falls $x = R$: Verschiebe den Schreib-Lesekopf ein Position nach rechts
- Falls $x = N$: Lasse Schreib-Lesekopf unverändert (Neutral)

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

Definition (Konfiguration einer Turingmaschine)

Eine Konfiguration einer Turingmaschine ist ein Wort $k \in \Gamma^* Z \Gamma^*$

D.h. eine Konfiguration ist ein Wort wzw' , sodass:

- die TM ist im Zustand z ,
- auf dem Band steht $\dots \square \square ww' \square \square \dots$ und
- der Schreib-Lesekopf steht auf dem ersten Symbol von w'

Definition (Konfiguration einer Turingmaschine)

Eine Konfiguration einer Turingmaschine ist ein Wort $k \in \Gamma^* Z \Gamma^*$

D.h. eine Konfiguration ist ein Wort wzw' , sodass:

- die TM ist im Zustand z ,
- auf dem Band steht $\dots \square \square ww' \square \square \dots$ und
- der Schreib-Lesekopf steht auf dem ersten Symbol von w'

Definition (Startkonfiguration einer TM)

Für ein Eingabewort w ist die Startkonfiguration einer TM $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ das Wort $z_0 w$.

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

D.h. am Anfang steht der Kopf auf dem ersten Symbol der Eingabe.

Quiz 1

Sei $aaabbb$ der Bandinhalt und der Schreib-Lesekopf der Turingmaschine steht auf dem ersten b von links. Die Turingmaschine ist in Zustand q .

Welches Wort beschreibt diese Konfiguration?

arsnova.hs-rm.de

6750 1376



Quiz 2

Sie $abcabcqc$ die Konfiguration einer Turingmaschine mit Zustandsmenge $\{z, q\}$.

Welches Symbol ist unter dem Schreib-/Lesekopf der Turingmaschine?

- A a
- B b
- C Das Blank-Symbol
- D c

arsnova.hs-rm.de

6750 1376

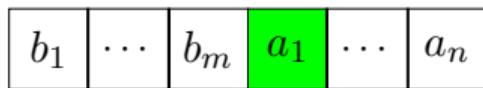
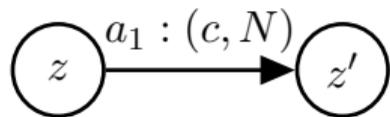


Definition (Transitionsrelation für Konfigurationen einer TM)

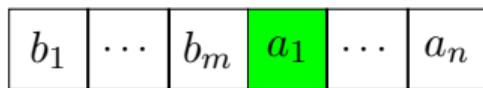
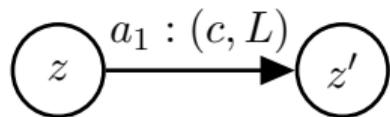
Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine TM. Die Relation \vdash_M ist definiert durch (wobei $\delta(z, a) = (z', c, x)$ für eine NTM $(z', c, x) \in \delta(z, a)$ meint):

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

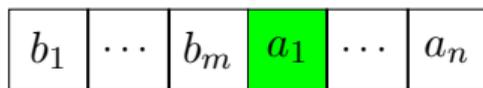
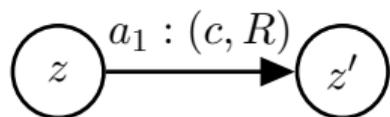
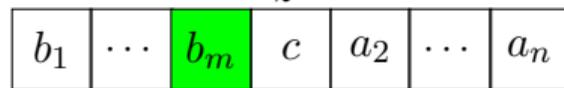
Illustrationen dazu



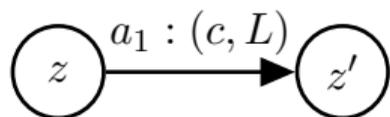
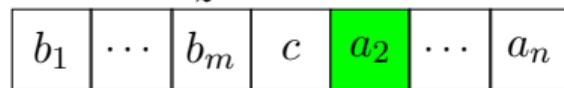
\vdash_M



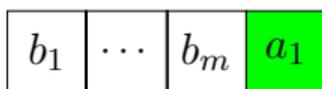
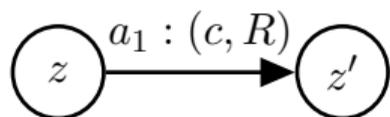
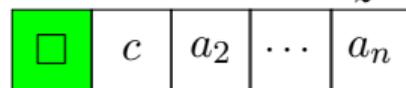
\vdash_M



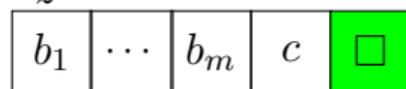
\vdash_M



\vdash_M



\vdash_M



Transitionsrelation einer TM (2)

Weitere Notation dazu:

- \vdash_M^i : die i -fache Anwendung von \vdash_M
- \vdash_M^* die reflexiv-transitive Hülle von \vdash_M
- Wenn M klar ist, schreiben wir nur \vdash, \vdash^i , bzw. \vdash^* .

Bemerkung:

Wir nehmen an, dass die TM anhält,
sobald sie einen Endzustand erreicht.

(Schöning-Buch erlaubt weiterrechnen)

Definition (Akzeptierte Sprache einer TM)

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

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

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

Definition (Akzeptierte Sprache einer TM)

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

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

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

Triviale Beispiele:

- Für Turingmaschinen der Form $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ mit $z_0 \in E$ gilt $L(M) = \Sigma^*$, denn diese Turingmaschinen akzeptieren jede Eingabe sofort.
- Für Turingmaschinen der Form $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, \emptyset)$ gilt $L(M) = \emptyset$, denn sie akzeptieren nie.

- Eigene Syntax, z.B. $\langle, -, \rangle$ statt L, N, R .
Übergänge als 2 Zeilen: z.B. $\delta(q_0, a) = (q_1, b, R)$ wird eingegeben als
q0, a
q1, b, >
- Leicht anderes Verhalten als unsere Turingmaschinen:
 - In Endzuständen wird weiter gerechnet
 - Daher können Endzustände auch wieder verlassen werden
 - Partielle Übergangsfunktion erlaubt
- Unsere Turingmaschinen für Simulator anpassen:
 - Übergänge für Endzustände entfernen
- Deren Turingmaschinen für uns anpassen:
 - Neuen Endzustand und neuen Müllzustand anlegen
 - Deren Endzustände zu Nicht-Endzuständen machen
 - Fehlende Übergänge von deren Endzustände in neuen Endzustand
 - Alle anderen fehlenden Übergänge in Müllzustand gehen lassen.

Beispiel (aus Schöning-Buch)

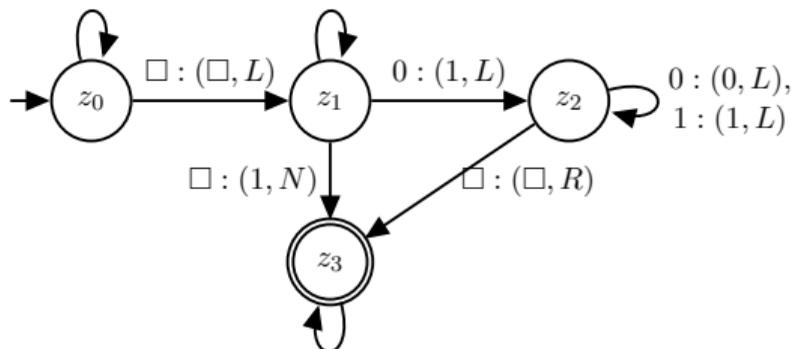
<http://turingmachinesimulator.com/shared/istktezldr>

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

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

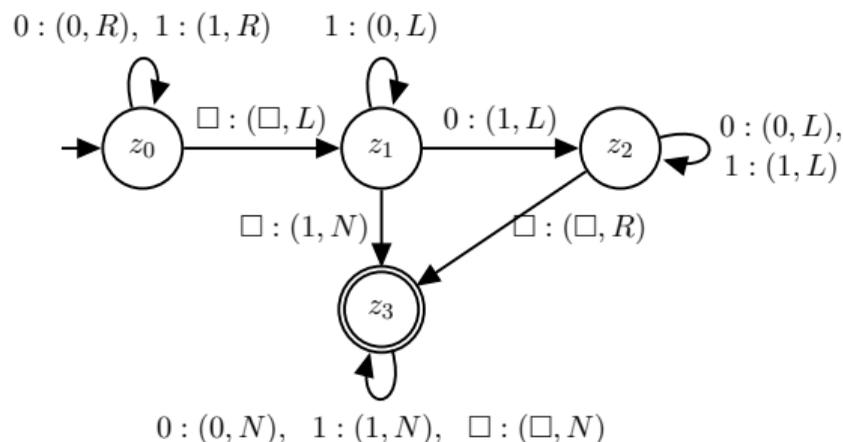
Zustandsgraph:

0 : (0, R), 1 : (1, R) 1 : (0, L)



0 : (0, N), 1 : (1, N), \square : (\square , N)

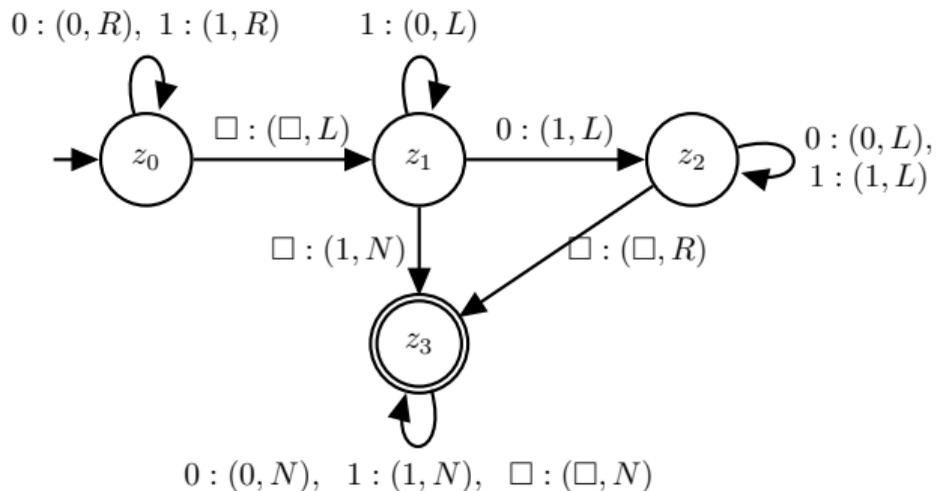
Beispiellauf



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

Quiz 3

Führen Sie den Lauf für Eingabe 10 auf folgender TM aus.



Welches Wort steht am Ende auf dem Band (ohne Blank-Symbole)?

arsnova.hs-rm.de

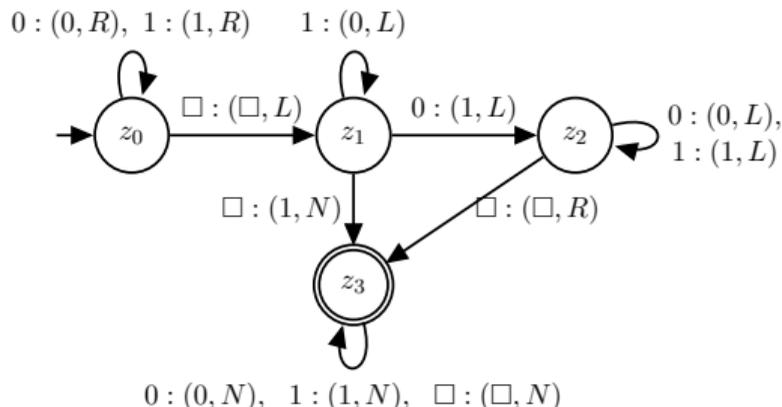
6750 1376



Beispiel (Forts.)

TM interpretiert 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 in z_2
Bei Übertrag: Weitermachen in z_1 und +1 zur nächsten Ziffer links
- In z_2 : bis zum Anfang links laufen, dann in z_3 .
- In z_3 wird akzeptiert.



Ideen und Notationen:

- **Linear beschränkte Turingmaschinen:**
Schreib-Lesekopf darf den **Bereich der Eingabe** auf dem Band **nicht verlassen**
- Zum Erkennen des Endes:
Letztes Symbol der Eingabe wird markiert
- Kopie des Alphabets:
Für Alphabet $\Sigma = \{a_1, \dots, a_n\}$ bezeichne $\hat{\Sigma} = \{\hat{a}_1, \dots, \hat{a}_n\}$.
- Eingabe bei LBAs: Statt $a_1 \cdots a_m$ nun $a_1 \cdots a_{m-1} \hat{a}_m$
- TM arbeitet auf $\Sigma' = \Sigma \cup \hat{\Sigma}$
- Linkes Ende muss die Maschine selbst markieren!

Definition (LBA)

Eine NTM $M = (Z, \Sigma \cup \widehat{\Sigma}, \Gamma, \delta, z_0, \square, E)$ heißt **linear beschränkt** (LBA, linear bounded automaton), wenn für alle $a_1 \cdots a_m \in \Sigma^+$ und **alle Konfigurationen** uzv mit $z_0 a_1 \cdots a_{m-1} \widehat{a}_m \vdash_M^* uzv$ gilt: $|uv| \leq m$.

Die **akzeptierte Sprache** eines LBA M ist

$$L(M) := \left\{ a_1 \cdots a_m \in \Sigma^+ \mid \begin{array}{l} z_0 a_1 \cdots a_{m-1} \widehat{a}_m \vdash_M^* uzv, \\ \text{wobei } u, v \in \Gamma^* \text{ und } z \in E \end{array} \right\}$$

Beachte: LBAs sind NTMs

Theorem (Satz von Kuroda)

Kontextsensitive Sprachen werden genau von den LBAs erkannt.

- Der Beweis ist im Anhang des Skripts zu finden.
- Ideen:
 - LBA \rightarrow Grammatik: Erzeuge Grammatik, die folgendes simuliert: Erzeuge ein beliebiges Wort aus Σ^* , und anschließend simuliere Turingmaschine auf diesem Wort. Wenn Turingmaschine akzeptiert beende Herleitung.
 - Grammatik \rightarrow LBA: Simuliere Herleitung rückwärts vom Wort zum Startsymbol: Ersetze Vorkommen von r durch linke Seite l , wenn $l \rightarrow r$ eine Produktion

Satz

Die durch (allgemeine) nichtdeterministischen Turingmaschinen akzeptierten Sprachen sind genau die Typ 0-Sprachen.

Analog, aber: Grammatik \rightarrow TM: $|l| > |r|$ erlaubt, daher Platz nicht beschränkt.

- Nichtdeterministische Turingmaschinen können durch deterministische Turingmaschinen simuliert werden:
Probiere alle Berechnungsmöglichkeiten der NTM nacheinander durch
- Daher gilt der letzte Satz auch für DTM
- Unterschied zwischen NTMs und DTMs kommt erst zum Tragen, wenn wir das Laufzeitverhalten betrachten (s. Kapitel zur Komplexitätstheorie)

Sprache	Grammatik	Automat	sonstiges
Typ 3	reguläre Grammatik	endlicher Automat (DFA und NFA)	regulärer Ausdruck
deterministisch kontextfrei	$LR(k)$ -Grammatik	Deterministischer Kellerautomat (DPDA)	
Typ 2	kontextfreie Grammatik	Kellerautomat (PDA) (nichtdeterministisch)	
Typ 1	kontextsensitive Grammatik	linear beschränkte Turingmaschine (LBA) (nichtdeterministisch)	
Typ 0	Typ 0-Grammatik	Turingmaschine (deterministisch und nichtdeterministisch)	

Beachte: $LR(k)$ -Grammatiken wurden nicht behandelt.

- Die Sprache $\{a^n b^n \mid n \in \mathbb{N}_0\}$ ist Typ 2 aber nicht vom Typ 3.
- Die Sprache $\{w \in \{a, b\}^* \mid w \text{ ist Palindrom}\}$ ist Typ 2 aber nicht deterministisch-kontextfrei.
- Die Sprache $\{a^n b^n c^n \mid n \in \mathbb{N}_0\}$ ist Typ 1 aber nicht vom Typ 2.
- Die Sprache

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

ist Typ 0 aber nicht vom Typ 1.

Anmerkung: H ist das Halteproblem, es wird später genau betrachtet und erläutert

- 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

Sprachklasse	Schnitt	Vereinigung	Komplement	Produkt	Kleenescher Abschluss
Typ 3	✓	✓	✓	✓	✓
det.kontextfrei	×	×	✓	×	×
Typ 2	×	✓	×	✓	✓
Typ 1	✓	✓	✓	✓	✓
Typ 0	✓	✓	×	✓	✓

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

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