

Turingmaschinen und Typ 1- und Typ 0-Sprachen

Prof. Dr. Jasmin Blanchette

Lehr- und Forschungseinheit für
Theoretische Informatik

Stand: 6. Juni 2023

Folien ursprünglich von PD Dr. David Sabel



Erinnerung: Typ 1- und Typ 0-Sprachen

- ▶ Typ 1: $|\ell| \leq |r|$ für alle Produktionen $\ell \rightarrow r$
- ▶ Typ 1-Grammatik = kontextsensitive Grammatik
- ▶ im Gegensatz zu Typ 2: $\ell \in (\Sigma \cup V)^+$
- ▶ Typ 0: alles erlaubt

Erinnerung: Typ 1- und Typ 0-Sprachen

- ▶ Typ 1: $|\ell| \leq |r|$ für alle Produktionen $\ell \rightarrow r$
- ▶ Typ 1-Grammatik = kontextsensitive Grammatik
- ▶ im Gegensatz zu Typ 2: $\ell \in (\Sigma \cup V)^+$
- ▶ Typ 0: alles erlaubt

- ▶ In manchen Büchern werden unsere Typ 1-Grammatiken 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 $\alpha_3 \neq \varepsilon$.

Erinnerung: Typ 1- und Typ 0-Sprachen

- ▶ Typ 1: $|\ell| \leq |r|$ für alle Produktionen $\ell \rightarrow r$
- ▶ Typ 1-Grammatik = kontextsensitive Grammatik
- ▶ im Gegensatz zu Typ 2: $\ell \in (\Sigma \cup V)^+$
- ▶ Typ 0: alles erlaubt

- ▶ In manchen Büchern werden unsere Typ 1-Grammatiken 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).

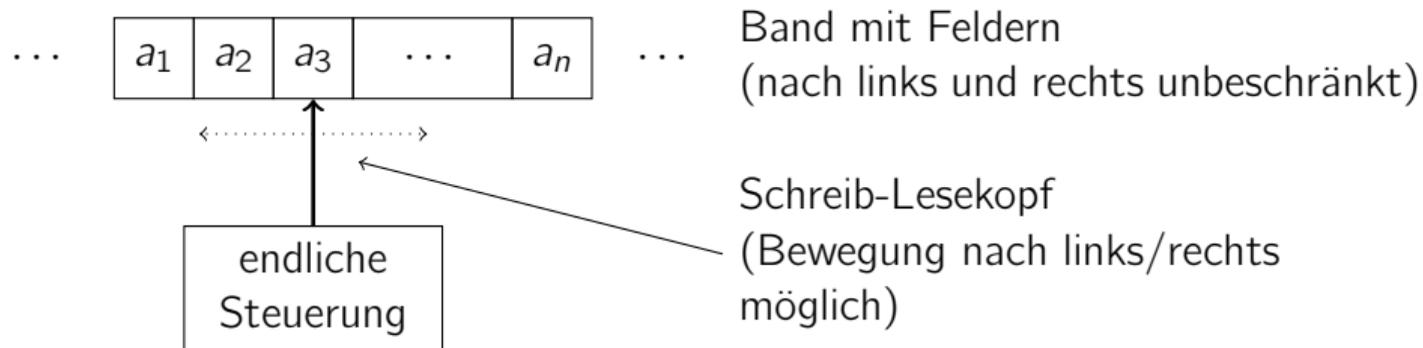
Hintergrund zu Turingmaschinen

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 $\{a^i b^i c^i \mid i \in \mathbb{N}_{>0}\}$ 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



Formale Definition der Turingmaschine

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** und
- ▶ $E \subseteq Z$ ist die Menge der **Endzustände**.

Zustandswechsel, informell

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)

Zustandswechsel, informell

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 $Start_M(w)$ 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.

Definition (Übergangsrelation für Konfigurationen einer DTM)

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine DTM. Die Relation \vdash_M 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 $\delta(z, a_1) = (z', c, N)$, $m \geq 0, n \geq 1, z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_{m-1} z' b_m c a_2 \cdots a_n$,
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 $n \geq 1, z \notin E$

Definition (Übergangsrelation für Konfigurationen einer NTM)

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine NTM. Die Relation \vdash_M 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 $(z', c, N) \in \delta(z, a_1)$, $m \geq 0$, $n \geq 1$, $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_{m-1} z' b_m c a_2 \cdots a_n$,
wenn $(z', c, L) \in \delta(z, a_1)$, $m \geq 1$, $n \geq 1$, $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \cdots a_n \vdash_M b_1 \cdots b_m c z' a_2 \cdots a_n$,
wenn $(z', c, R) \in \delta(z, a_1)$, $m \geq 0$, $n \geq 2$, $z \notin E$
- ▶ $b_1 \cdots b_m z a_1 \vdash_M b_1 \cdots b_m c z' \square$,
wenn $(z', c, R) \in \delta(z, a_1)$ und $m \geq 0$, $z \notin E$
- ▶ $z a_1 \cdots a_n \vdash_M z' \square c a_2 \cdots a_n$,
wenn $(z', c, L) \in \delta(z, a_1)$ und $n \geq 1$, $z \notin E$

Übergangsrelation einer TM

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^* .

Übergangsrelation einer TM

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.)

Akzeptierte Sprache einer TM

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

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

Akzeptierte Sprache einer TM

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

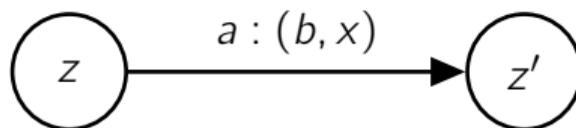
$$L(M) := \{w \in \Sigma^* \mid \exists u, v \in \Gamma^*, z \in E : Start_M(w) \vdash_M^* uzv\}$$

Triviale Beispiele:

- ▶ Für Turingmaschinen von 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 von der Form $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, \emptyset)$ gilt $L(M) = \emptyset$, denn sie akzeptieren nie.

Notation als Zustandsgraph

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



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

$$\delta(z_0, 0) = (z_0, 0, R) \quad \delta(z_0, 1) = (z_0, 1, R) \quad \delta(z_0, \square) = (z_1, \square, L)$$

$$\delta(z_1, 0) = (z_2, 1, L) \quad \delta(z_1, 1) = (z_1, 0, L) \quad \delta(z_1, \square) = (z_3, 1, N)$$

$$\delta(z_2, 0) = (z_2, 0, L) \quad \delta(z_2, 1) = (z_2, 1, L) \quad \delta(z_2, \square) = (z_3, \square, R)$$

$$\delta(z_3, 0) = (z_3, 0, N) \quad \delta(z_3, 1) = (z_3, 1, N) \quad \delta(z_3, \square) = (z_3, \square, N)$$

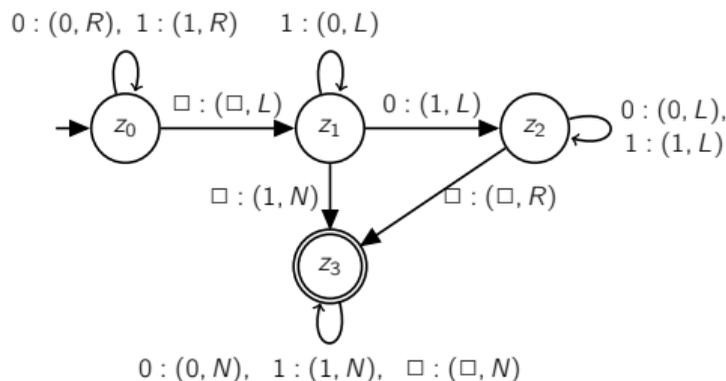
Beispiel (aus Schöning-Buch)

<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

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

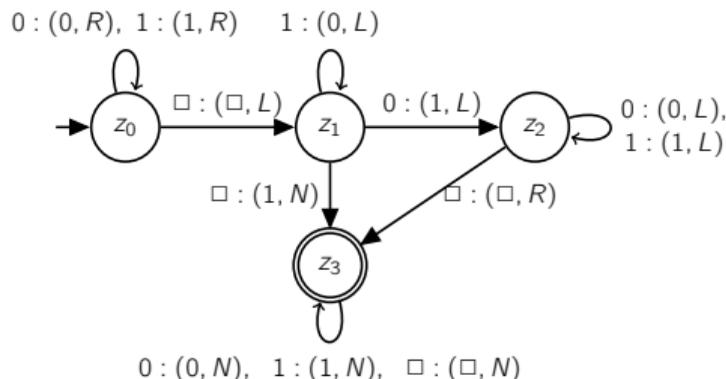
Zustandsgraph:



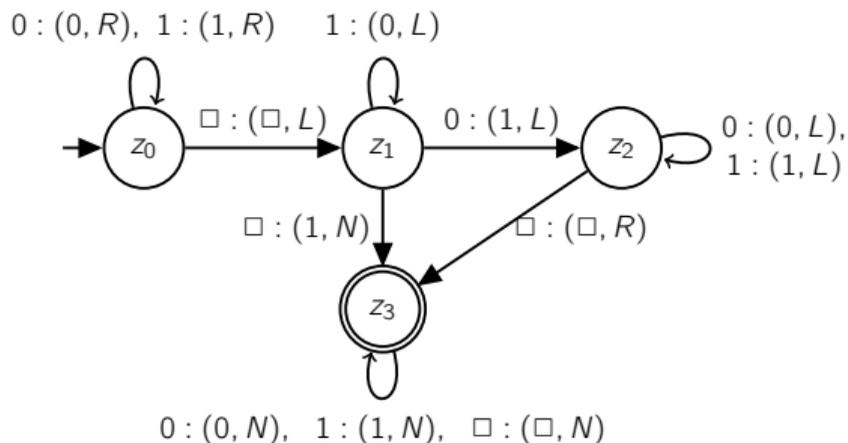
Beispiel (Fortsetzung)

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.



Beispiellauf



z₀0011
⊢ 0z₀011
⊢ 00z₀11
⊢ 001z₀1
⊢ 0011z₀□
⊢ 001z₁1□
⊢ 00z₁10□
⊢ 0z₁000□
⊢ z₂0100□
⊢ z₂□0100□
⊢ □z₃0100□

LBAs: Spezielle Turingmaschinen

Ideen und Notationen:

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

Die **akzeptierte Sprache eines LBA** M ist

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

Beachte: LBAs sind NTMs.

Theorem (Satz von Kuroda)

Kontextsensitive Sprachen werden genau von den LBAs erkannt.

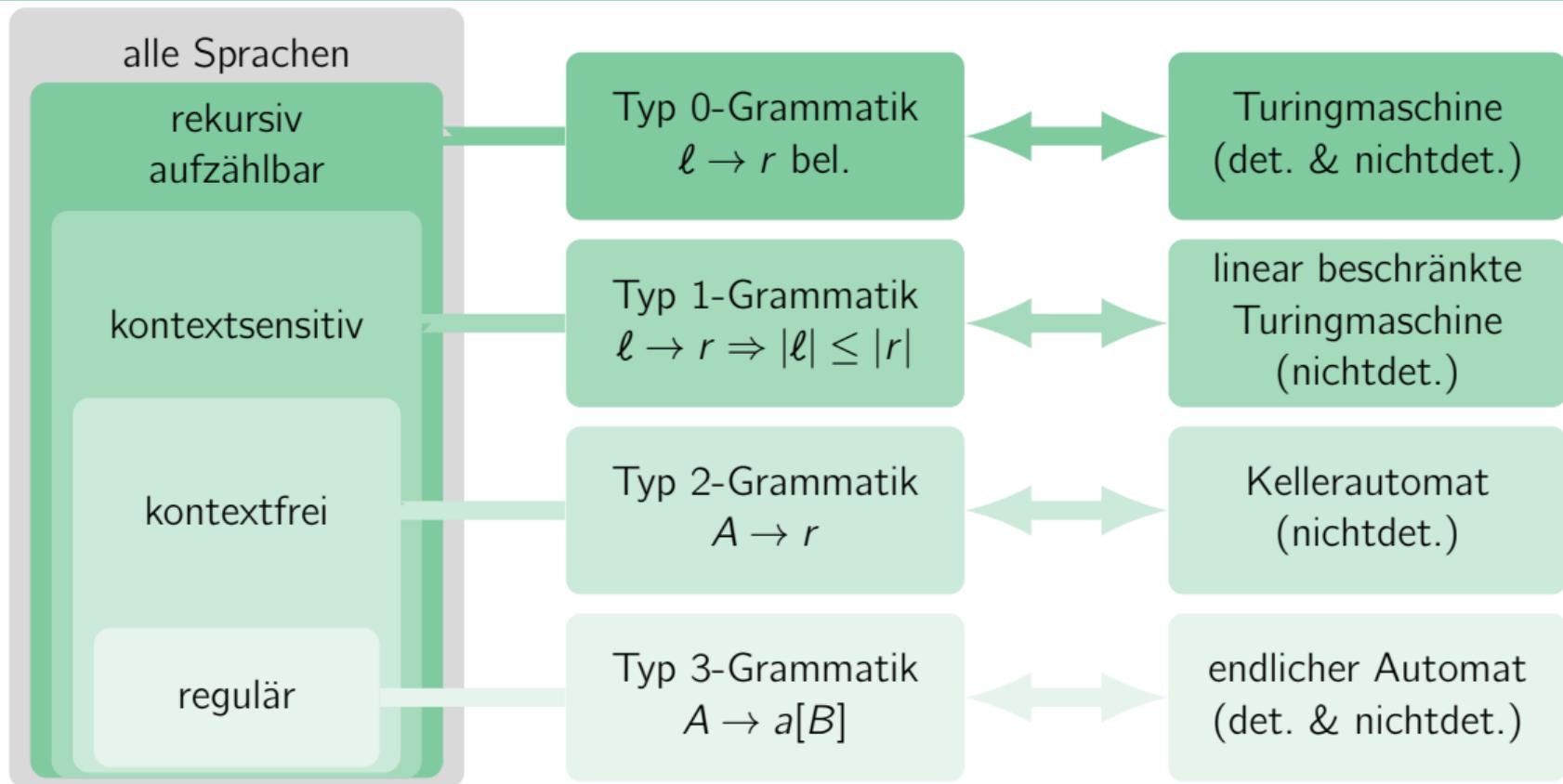
Satz

Typ 0-Sprachen werden genau von den (allgemeinen) nichtdeterministischen Turingmaschinen erkannt.

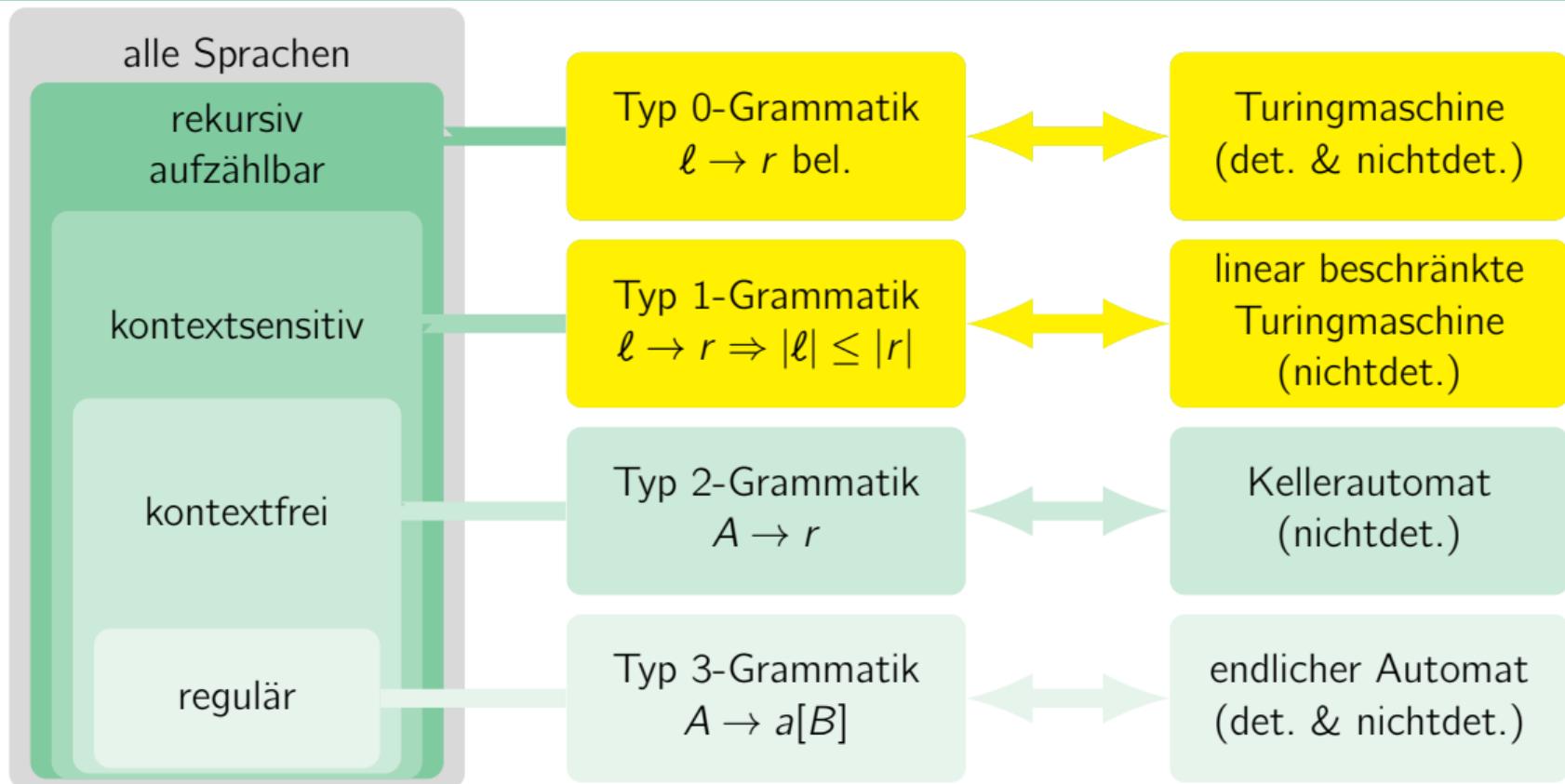
Beweis: Nächste Vorlesung (nur FSK)

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

Überblick: Grammatiken & Maschinenmodelle für die Chomsky-Hierarchie



Überblick: Grammatiken & Maschinenmodelle für die Chomsky-Hierarchie



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

$$H = \{M\#w \mid \text{die durch } M \text{ beschriebene} \\ \text{Turingmaschine h\u00e4lt bei Eingabe } w\}$$

ist vom Typ 0 aber nicht vom Typ 1.

(Die Sprache H ist das Halteproblem, welches wir sp\u00e4ter noch genauer betrachten und erl\u00e4utern.)

- ▶ 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
Typ 0	unlösbar