



# Theoretische Informatik: Formale Sprachen und Automaten

Basisinformationstechnologie I – Wintersemester 2021 / 2022

Dr. Jan Wieners | Institut für Digital Humanities | Universität zu Köln

# Themenüberblick „Theoretische Informatik“

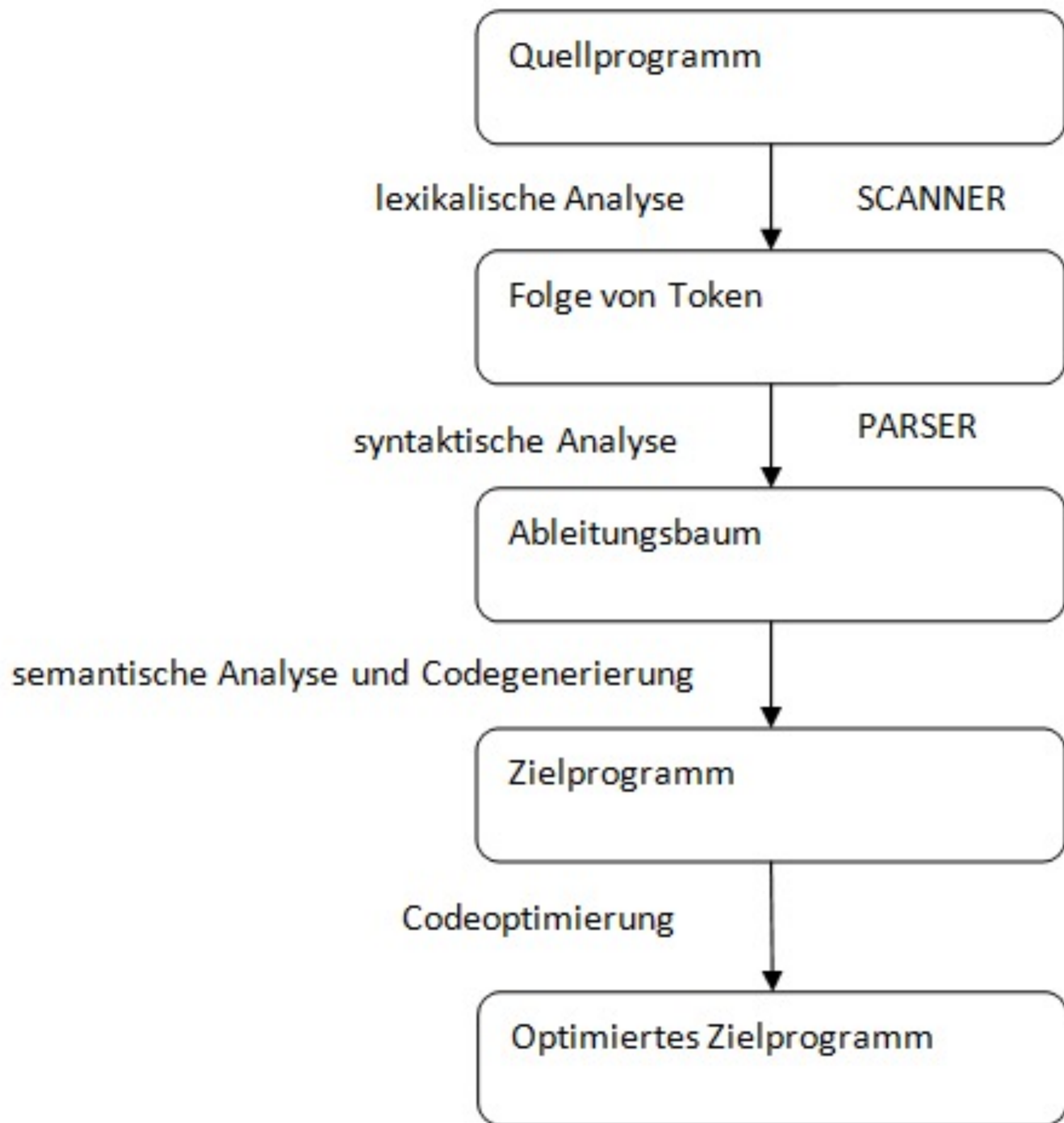
- Formale Sprachen
  - Alphabet
  - Buchstabe
  - Wort
- Automaten
  - Deterministische endliche Automaten
  - (Nichtdeterministische endliche Automaten)
  - Anwendung endlicher Automaten
- Grammatiken
- (Kellerautomat)
- (Turingmaschine)

# Formale Sprachen (& Co.)

```
mov al, 61h
```

vs.

```
SELECT uid, name, pass, mail FROM users  
WHERE uid < 2
```



Wer reitet so spät durch Nacht und Wind?  
Es ist der Vater mit seinem Kind;  
er hat den Knaben wohl in dem Arm,  
er fasst ihn sicher, er hält ihn warm.

Mein Sohn, was birgst du so bang dein Gesicht? -  
Siehst Vater, du den Erlkönig nicht?  
Den Erlenkönig mit Kron' und Schweif? -  
Mein Sohn, es ist ein Nebelstreif.



**Die Möglichkeiten der deutschen Grammatik können einen, wenn man sich darauf, was man ruhig, wenn man möchte, sollte, einlässt, überraschen.**

[www.twitterperlen.de](http://www.twitterperlen.de)

Quelle: [twitter.com/Gedankenbalsam](https://twitter.com/Gedankenbalsam)



**Halbblutlehrerin (🏠)** @MelsGedanken · 16. März 2019

Gestern wurde mir ein Antrag auf Unterrichtsbefreiung vorgelegt:

„Bitte befreien Sie meinen Sohn Justin am nächsten Freitag vom Unterricht, weil Justin seine Tante heiratet.“

Gratulier ich dem Bub jetzt? Meld ich das dem Jugendamt? Oder schenke ich der Familie einen Duden?

🗨️ 632

↻ 3.845

❤️ 21.771





**Wir essen jetzt, Opa!**

**Wir essen jetzt Opa!**

**Dieser Bereich wird zur  
Verhütung von Straftaten  
durch die Polizei videoüberwacht**

Polizeipräsidium Westhessen  
-Polizeidirektion Limburg-Weilburg-

# Grammatiken

## Grammatikbeispiel natürliche Sprache: deutsche Sprache

<Satz> → <Nominalsatz> <Verb>  
<Nominalsatz> → <Artikel><Substantiv>  
ODER  
<Artikel><Adjektiv><Substantiv>  
<Artikel> → das | die  
<Adjektiv> → große | kleine  
<Substantiv> → Klavier | Katze  
<Verb> → schläft

Definition:

- Ausdrücke in spitzen Klammern, z.B. <Verb>: **Variablen** oder **Nichtterminalsymbole**
- Andere Ausdrücke, z.B. „schläft“: **Terminalsymbole**

# Grammatiken

Grammatikalisch richtig und sinnvoll:

*„Die kleine Katze schläft.“*

Grammatikalisch falsch

(zwar richtig nach dem vorhergehenden Beispiel, jedoch falsch in der Grammatik des Deutschen):

*„Das große Katze schläft.“*

Grammatikalisch richtig, jedoch sinnlos:

*„Das große Klavier schläft.“*

# Formale Sprachen

## Formale Sprache

- z.B. Programmiersprache
- auf Basis einer verhältnismäßig kleinen Menge von Regeln werden Programme formuliert, die der Rechner verstehen kann

## Zu klären (natürliche und formale Sprache):

- Alphabet
- Buchstabe
- Wort
- ...

Definitorisches:  
Alphabet, Wort, Sprache

Alphabet?

# Alphabet.

- endliche, nichtleere Menge von Zeichen (auch: Buchstaben oder Symbole → formale Einheit, die nicht weiter definiert wird)
- **V** oder **Σ** (Sigma) als Abkürzung für Alphabete



$\Sigma$  Binärcode = {???

$$\Sigma \text{ Binärcode} = \{0,1\}$$

$$\Sigma \text{ Hexadezimalcode} = \{???\}$$

**$\Sigma$**  Binärcode = { 0,1 }

**$\Sigma$**  Hexadezimalcode = {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}

# Wort.

- Ein Wort über einem Alphabet  $\Sigma$  ist eine endliche Folge von Zeichen aus  $\Sigma$ .
- Es existiert genau ein Wort über  $\Sigma$  der Länge 0, das Leerwort, das wir mit  $\varepsilon$  (Epsilon) bezeichnen
- Beispiel: Aus den Zeichen „a“, „b“, „c“ lassen sich Wörter wie „cba“ oder „aab“ bilden.

w = Haus

$|w| = 4$

Alphabet? Wort?

$\Sigma = \{ \text{Anweisung, Bedingung, if, then, else, begin, end, while} \}$

$w = \text{if Bedingung then Anweisung end}$

$|w| = 5$

Die **Menge aller Wörter**, die mit Zeichen aus  $\Sigma$  gebildet werden können, wird mit  $\Sigma^*$  bezeichnet

→ Jede beliebige Teilmenge von  $\Sigma^*$  wird als **formale Sprache** bezeichnet.

# Sprache

Die wichtigste Operation für Wörter besteht aus dem **Verketteten** einzelner Wörter

→ **Komposita** (z.B. Dampfschiffahrt).

Beispiel:  $\Sigma = \{0, 1\}$

Die Wörter über  $\Sigma$  (Sigma) sind alle endlichen 0-1-Folgen inklusive des Leerwortes  $\epsilon$  (Epsilon), d.h.:

$\Sigma^* = \{ \text{Epsilon}, 0, 1, 00, 01, 10, 11, \dots \}$

→ Wir bezeichnen Sprachen mit dem Buchstaben **L** (Language)

Eine typische Sprache L über  $\Sigma$  ist die Menge aller vorzeichenlosen Binärzahlen ohne führende Nullen:  $L = \{0, 1, 10, 11\}$

## Zwischenstand

- Formale- vs. natürliche Sprachen
  - Grammatik
  - Alphabet
  - Buchstabe
  - Wort
- 
- ToDo → Konzepte: Automaten & Co.

# Automaten





24h  
**Times**

タイムズ代々木駅前

クレジット  
ポイント  
最大  
現金

**HOT:** 暑気候があったら  
ドリンク  
**POKKA P**

お買得コーナー  
あります。

乗ってる？  
知ってる？  
Times Car  
PUSH

ぽっかサッポロ

Asahi  
SOFT DRINKS

Asahi CALPIS

Asahi SOFT DRINKS

乗ってる？  
知ってる？  
Times Car  
PUSH



HOSTED ON

**LiveLeak**







CIGARETTEN

Hall  
TABAKWAREN

40

40  
CIG.

DM

[www.schlauchOmat.de](http://www.schlauchOmat.de)



Nie mehr ohne...

SFR 10.-

**Continental**



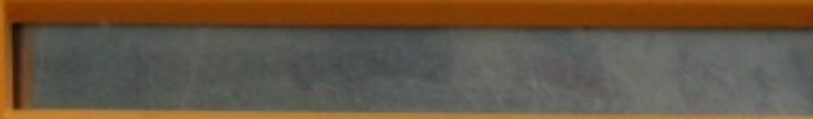
Fahrradschläuche

[www.conti-fahrradreifen.de](http://www.conti-fahrradreifen.de)

Anwahl gesperrt? Bitte kontaktieren Sie:



Taste ganz eindrücken



**FAIL**

**PEPSI**



**failblog.org**





Gemeinsamkeiten?  
Automatencharakteristika?

→ Zustände, Aktionen,  
Zustandsübergänge / Transitionen



Sortie

Sortie

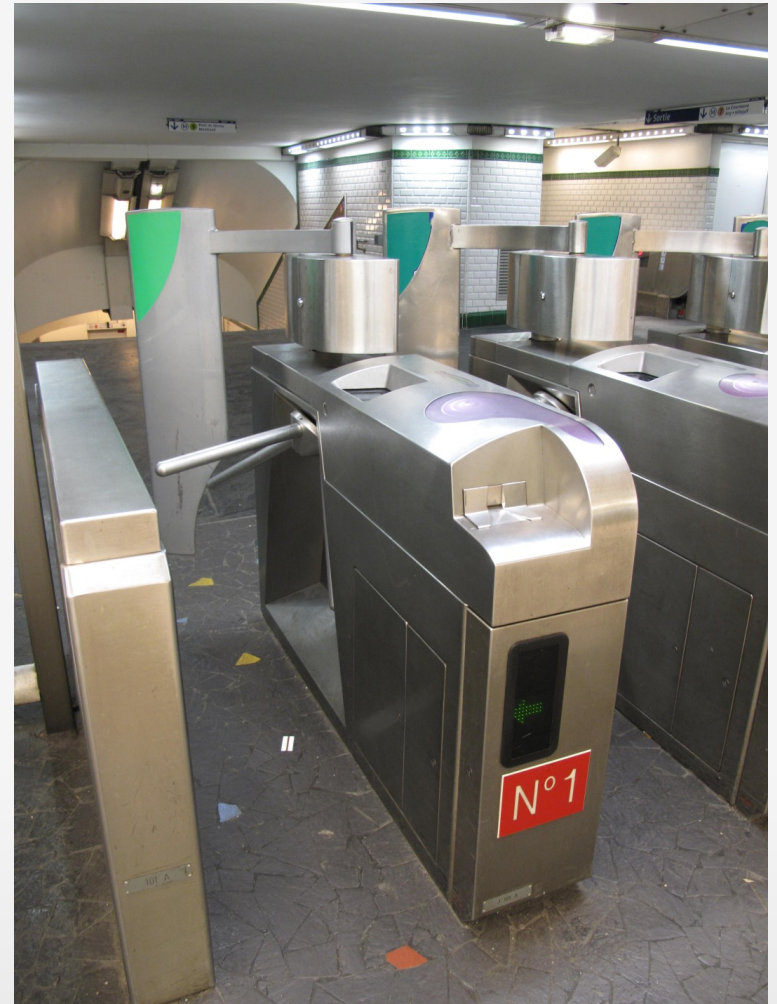
N°1

101 A

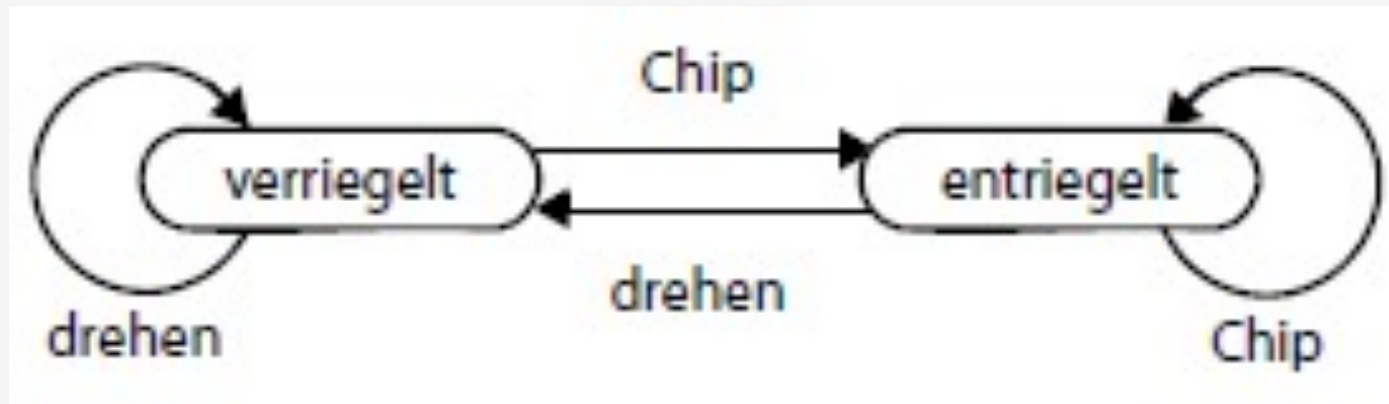
# Endliche Automaten

## Beispiel Drehkreuzautomat:

- Verwendet wird das Alphabet  $\Sigma = \{ \text{„Chip“}, \text{„drehen“} \}$



# Endliche Automaten



## Zustände / Zustandsübergänge:

- Verriegelt: Im verriegelten Zustand lassen sich die Arme nicht drehen
- Chipeinwurf: Einwurf entriegelt Sperre
- Im entriegelten Zustand: Chipeinwurf wirkungslos
- Im entriegelten Zustand: Werden die Arme gedreht, geht der Automat in den Zustand „verriegelt“ zurück

# Endliche Automaten

## Beispiel:

```
<html>  
  <head>  
    <title>Ein HTML-Dokument</title>  
  </head>  
  <body>Inhalt des HTML-Dokumentes</boby>  
</html>
```

## Formale Definition DEA:

Ein deterministischer endlicher Automat [...] besteht [...] aus

- einem Alphabet  $\Sigma$ , das die zulässigen Eingabezeichen auflistet,
- einer endlichen Menge  $Z$  von Zuständen
- und einer Vorschrift  $\delta$  (Delta), die die Zustandsübergänge beschreibt.

Zusätzlich verfügt ein Automat über einen speziellen **Startzustand** sowie einen oder mehrere akzeptierende Zustände (auch **Endezustände** genannt).

Vgl.: Socher, Rolf: Theoretische Grundlagen der Informatik. München 2008, Carl Hanser Verlag, S. 18.

Formale Definition DEA, kurz und knackig:

Ein DEA besteht aus den fünf Komponenten („fünf-Tupel“)  $Z, \Sigma, \delta, z_0, E$

- $Z$  ist eine endliche Menge von **Zuständen**
- $\Sigma$  ist ein Alphabet, das **Eingabealphabet**
- $\delta : Z \times \Sigma \rightarrow Z$  ist die **Übergangsfunktion**
- $z_0 \in Z$  ist der **Startzustand**
- $E \subseteq Z$  ist die Menge der **Endezustände** (E ist Teilmenge der Menge von Zuständen Z)

## Beispiel: Automat zur lexikalischen Analyse

Der Automat  $A_{LA} = (\mathbf{Z}, \Sigma, \delta, z_0, \mathbf{E})$  prüft, ob die Eingabe eine korrekt geschriebene ganze Zahl ohne führende Nullen darstellt.

Gültige Eingabezeichen:

$$\Sigma = \{-, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$



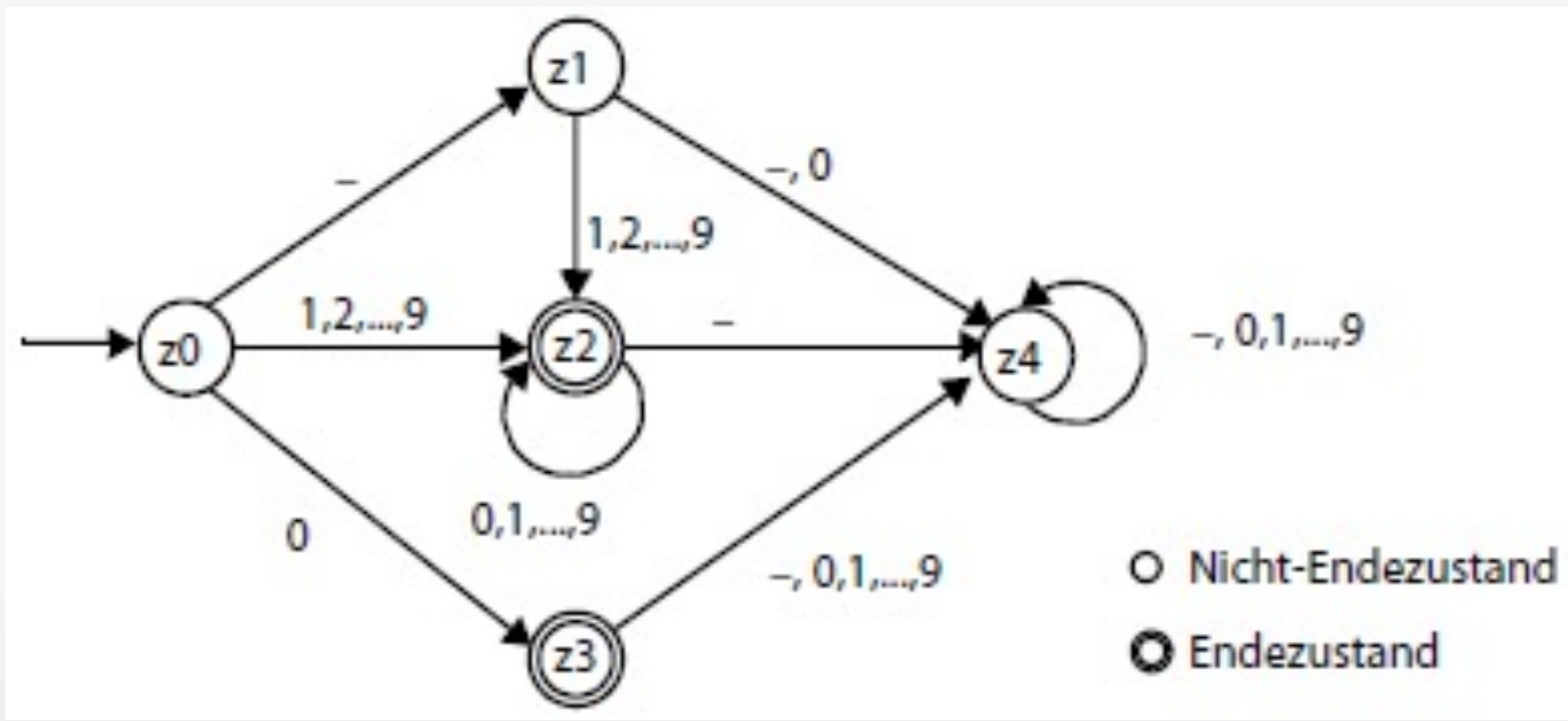
# Deterministischer endlicher Automat

## Übergangsdigramm:

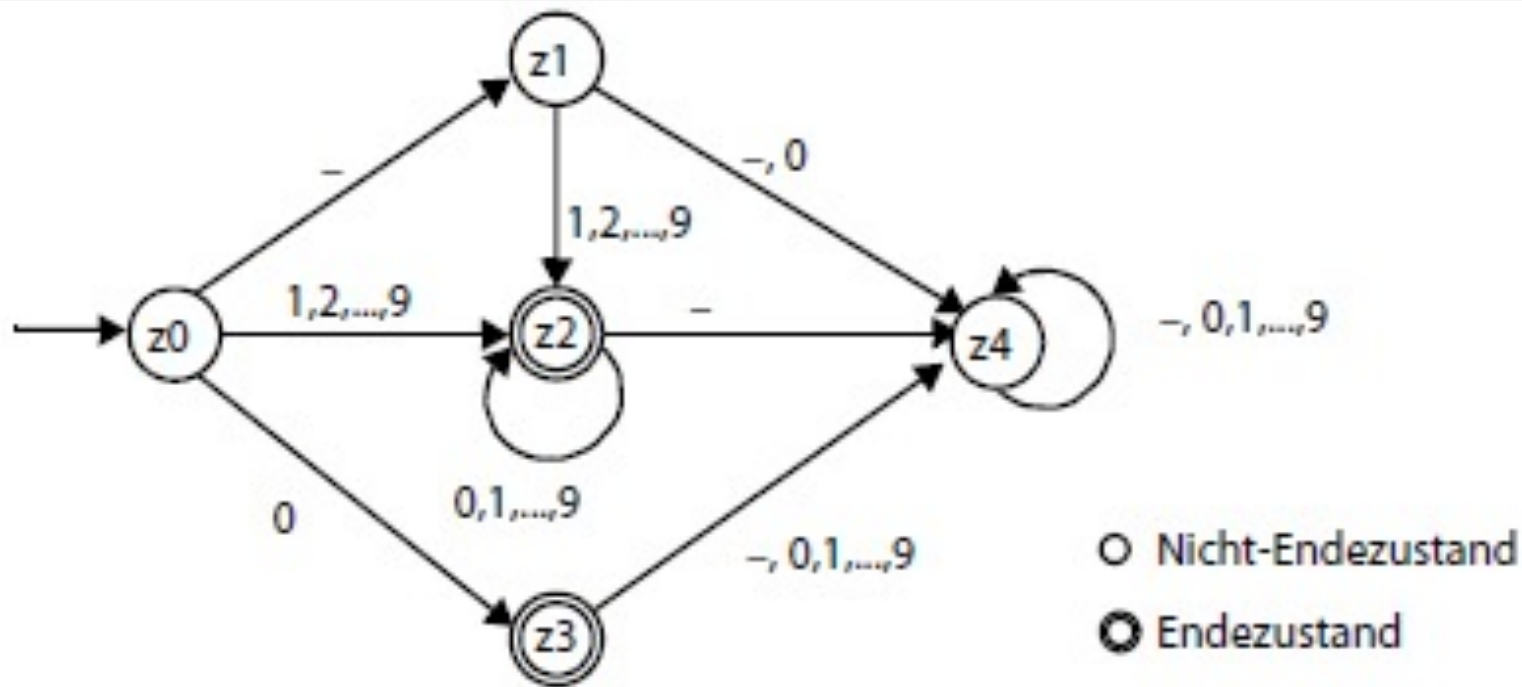
- Jeder in  $Z$  enthaltene Zustand wird durch einen **Knoten** dargestellt
- Für jeden Zustand  $z_0$  aus  $Z$  und für jedes Eingabesymbol  $a$  aus  $\Sigma$  sei  $\delta(z_0, a)=p$ 
  - Das Übergangsdigramm enthält dann einen Pfeil, der von Knoten  $z_0$  zu Knoten  $p$  führt und mit  $a$  beschriftet ist
- Knoten, die **akzeptierenden Zuständen** entsprechen, werden durch eine doppelte Kreislinie gekennzeichnet.

# Deterministischer endlicher Automat

## Zustands-/Übergangsdiagramm:



Vgl.: Socher, Rolf: Theoretische Grundlagen der Informatik. München 2008, Carl Hanser Verlag. S. 19 f.

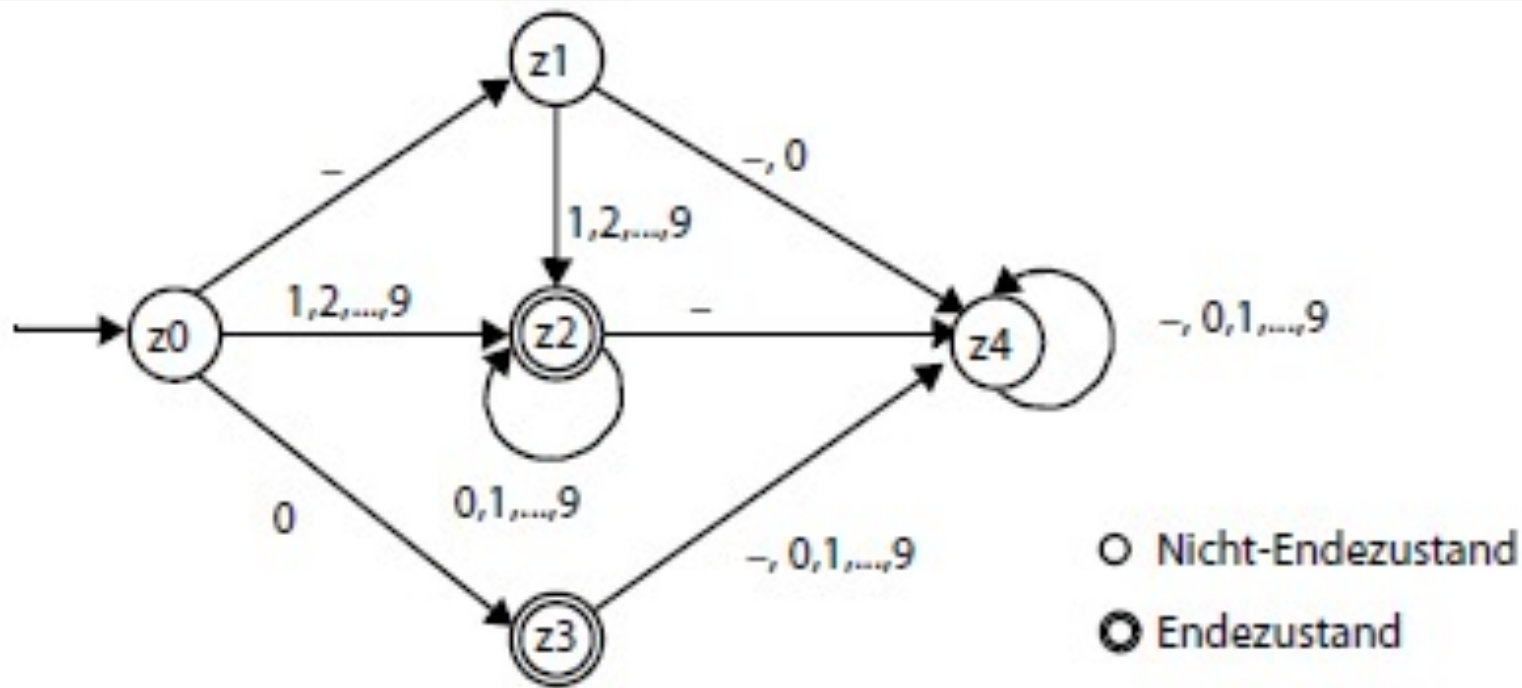


Fall I:

Zu Beginn, d.h. im Startzustand  $z_0$ , können drei Fälle unterschieden werden:

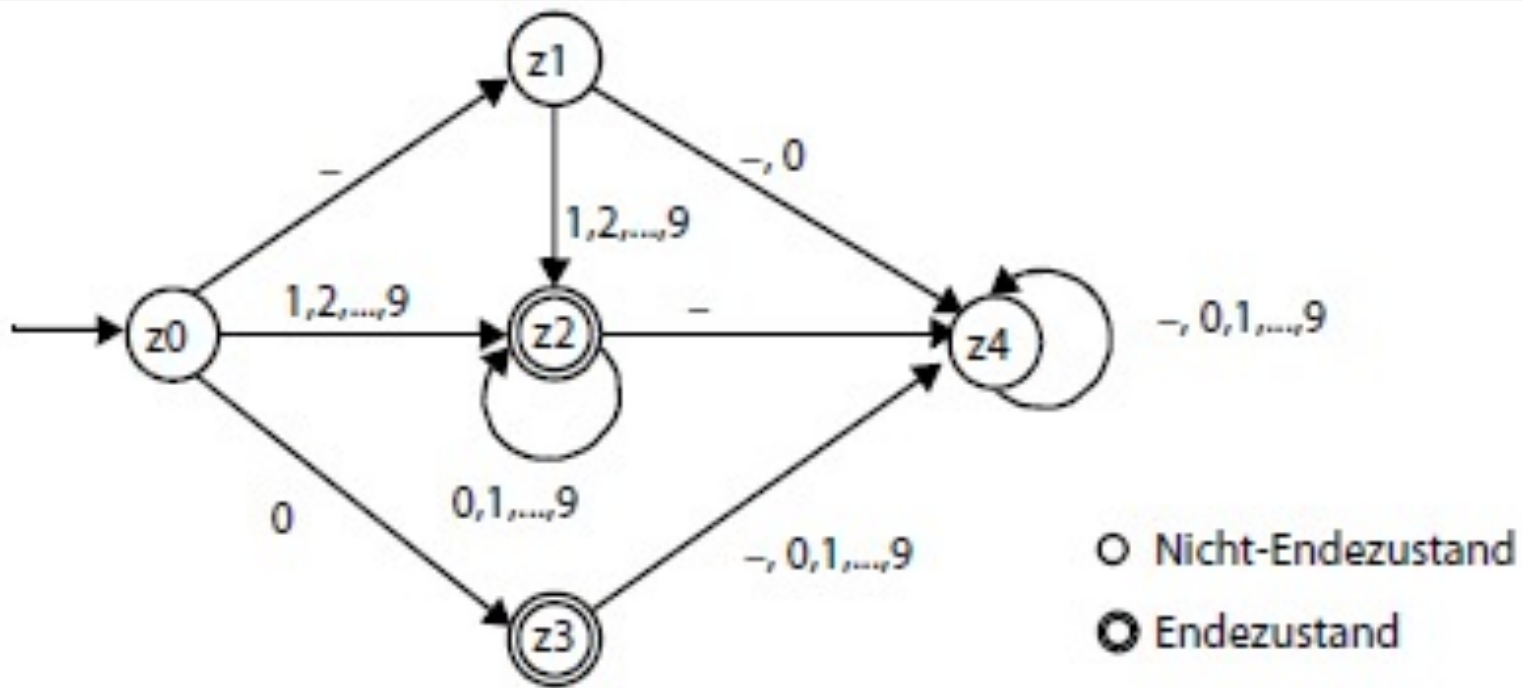
- Ist das erste Eingabezeichen eine 0, so springt der Automat in den Zustand  $z_3$ :  $(z_0, 0) \rightarrow (z_3, \varepsilon)$
- $z_3$  muss ein akzeptierender Zustand sein, da 0 ein gültiges Eingabewort ist.
- Anschließend darf kein weiteres Zeichen mehr kommen, weil führende Nullen nicht erlaubt sind.

Z.B.:  $(z_3, 0) \rightarrow (z_4, \varepsilon)$



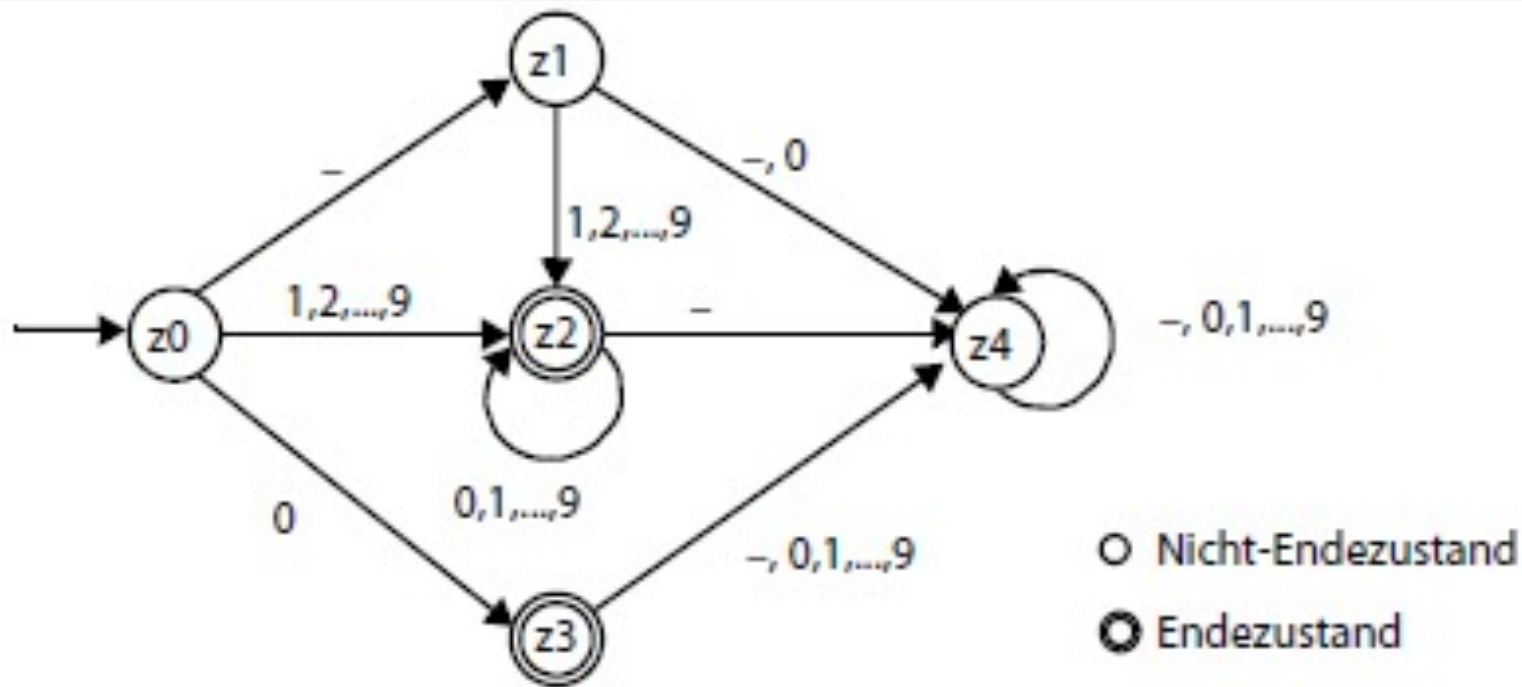
## Fall II:

- Ist das erste Eingabezeichen nicht 0 und auch nicht das Minuszeichen, so springt der Automat in den Zustand  $z_2$ .  
 Z.B.:  $(z_0, 3) \rightarrow (z_2, \varepsilon)$
- Anschließend dürfen beliebig viele Ziffern, jedoch kein Vorzeichen kommen.  
 Z.B.:  $(z_2, 5) \rightarrow (z_2, \varepsilon)$   
 oder (kein gültiger Endzustand)  $(z_2, -) \rightarrow (z_4, \varepsilon)$



Fall III:

- ???

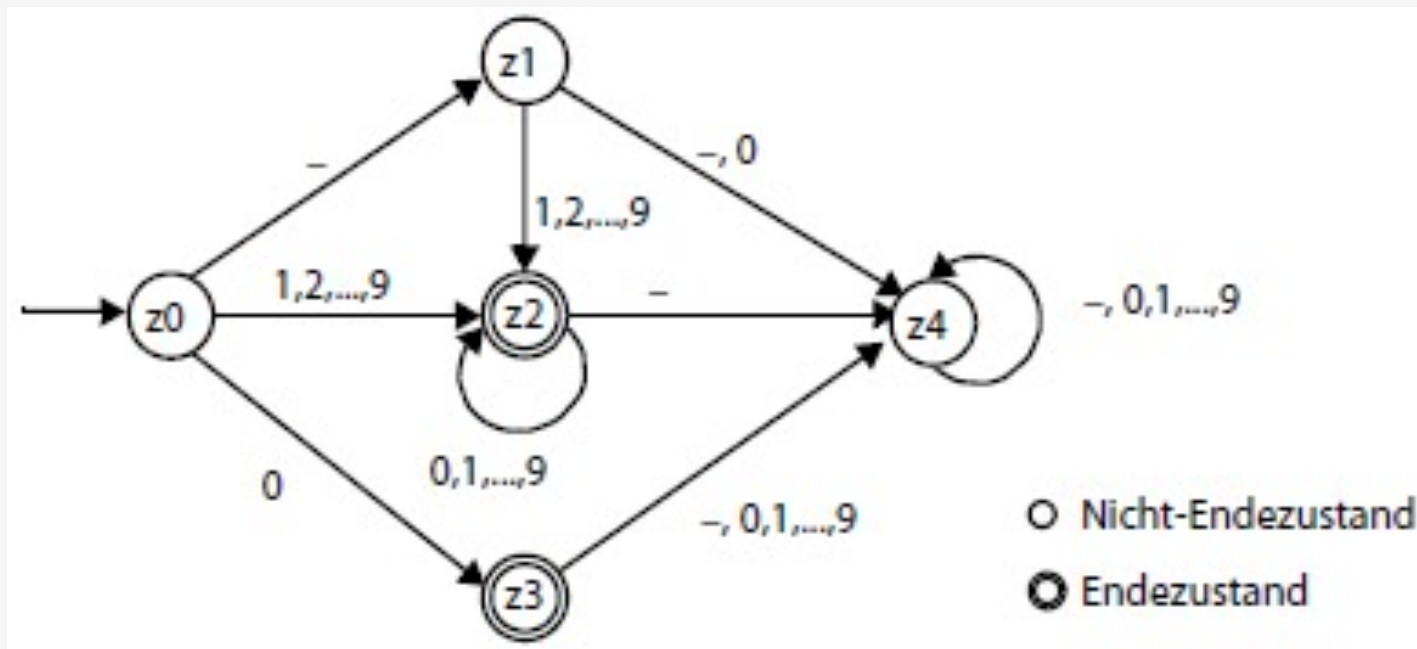


Fall III:

- Ist das erste Eingabezeichen das Vorzeichen, so springt der Automat in den Zustand  $z_1$ .
- Folgt nun eine weitere Ziffer ungleich 0, so springt der Automat in den Zustand  $z_2$ .

# Übung

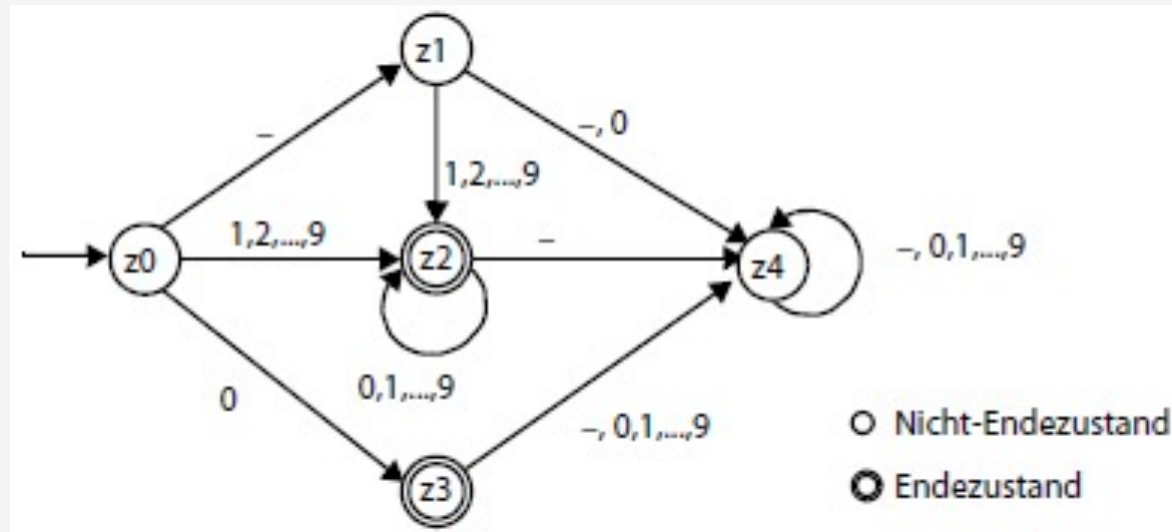
Beispiel: Eingabe „-153“



Ist das Eingabewort „-153“ eine korrekt geschriebene ganze Zahl?

# Übung

Beispiel: Eingabe „-153“

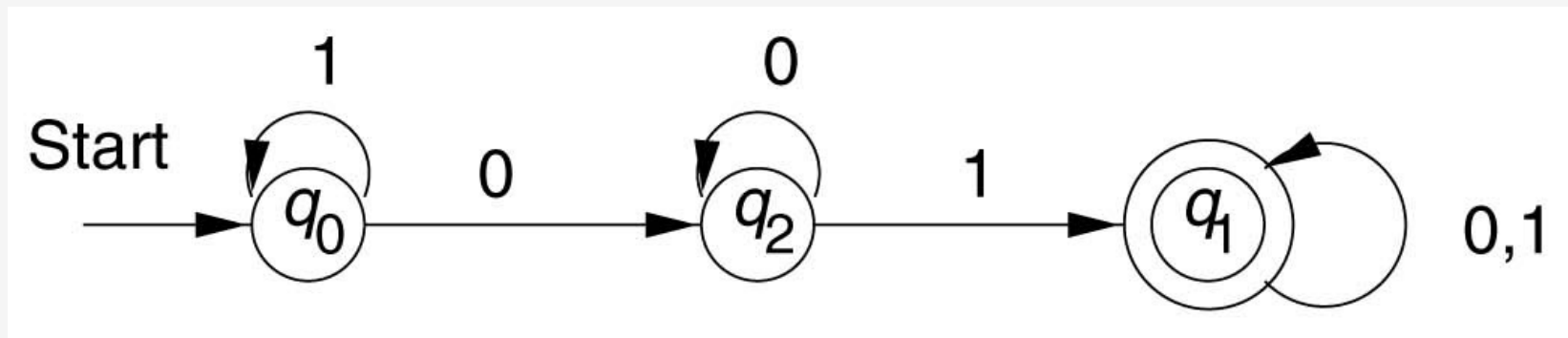


- Verlauf:  $(z_0, -153) \rightarrow (z_1, 153) \rightarrow (z_2, 53) \rightarrow (z_2, 3) \rightarrow (z_2, \varepsilon)$
- Fazit: Die Verarbeitung endet im Zustand  $z_2$ , der ein Endezustand ist. Das Eingabewort „-153“ ist folglich eine korrekt geschriebene ganze Zahl.
- Die Menge aller Eingabewörter, die vom Automaten A akzeptiert werden, wird als **die von A akzeptierte Sprache** bezeichnet



# Übung

Wie reagiert der folgende **DEA** auf die Eingabe „111001“? Verarbeitet der Automat die Eingabe?



# Zwischenstand II

## Automaten

- Deterministische endliche A.
- [Nichtdeterministische A.]

## ToDo:

- Grammatiken
- Kellerautomat
- Turingmaschine

# Grammatiken

# Grammatiken

Grundfrage: Wie erkennen wir, ob ein „Satz“ einer Programmiersprache wohlgeformt, d.h. korrekt ist?

Lösung: Grammatik → Regelwerk, das beschreibt, wie syntaktisch korrekte Wörter und Sätze einer Sprache, z.B. einer Programmiersprache gebildet werden

# Grammatiken

## Grammatikbeispiel natürliche Sprache: deutsche Sprache

<Satz>                    →        <Nominalsatz> <Verb>  
<Nominalsatz>         →        <Artikel><Substantiv>  
  ||  
  <Artikel><Adjektiv><Substantiv>  
<Artikel>                 → das | die  
<Adjektiv>                → große | kleine  
<Substantiv> → Klavier | Katze  
<Verb>                    → schläft

Definition:

- Ausdrücke in spitzen Klammern, z.B. <Verb>: **Variablen** oder **Nichtterminalsymbole**
- Andere Ausdrücke, z.B. „schläft“: **Terminalsymbole**

# Grammatiken

Definition: Eine Grammatik  $G$  ist ein **Quadrupel** (d.h. ein System mit **4** charakteristischen Bestandteilen):

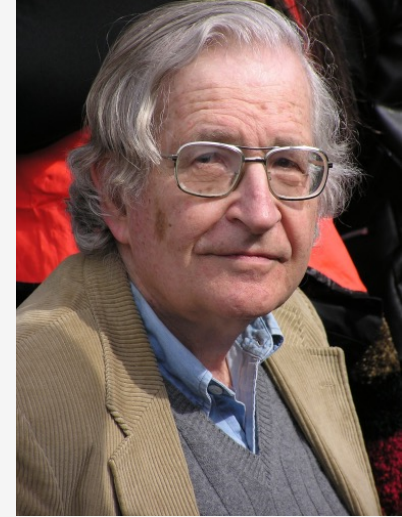
$$G = \{V, \Sigma, P, S\}$$

- $V$  bezeichnet eine endliche Menge von Variablen
- $\Sigma$  ist das Terminalalphabet, d.h. die Menge der Terminalen Symbole. Terminale Symbole bilden das Alphabet der Sprache, sind nicht weiter zerlegbar (notiert als Kleinbuchstaben)
- ▶  $P$  ist eine Menge von **Produktionen**, d.h. eine endliche Menge von Regeln.  
Beispiel:  $\langle \text{Satz} \rangle \rightarrow \langle \text{Nominalsatz} \rangle \langle \text{Verb} \rangle$
- $S$  ist die **Startvariable**

# Grammatiken

**Chomsky-Hierarchie** (sowohl für natürliche, als auch für formale Sprachen):

- Chomsky-**Typ 0**, auch: **allgemeine Grammatik**.
- Chomsky-**Typ 1**, auch: **kontextsensitive Grammatik**.
- Chomsky-**Typ 2**, auch: **kontextfreie Grammatik**.
- Chomsky-**Typ 3**, auch: **reguläre Grammatik**.



# Automaten

- **Typ 0** (allgemein) → **Turing-Maschine**
- **Typ 1** (kontextsensitiv) → **linear beschränkter Automat**
- **Typ 2** (kontextfrei) → **Kellerautomat**
- **Typ 3** (regulär) → **endlicher Automat**

# Grammatiken und die Praxis

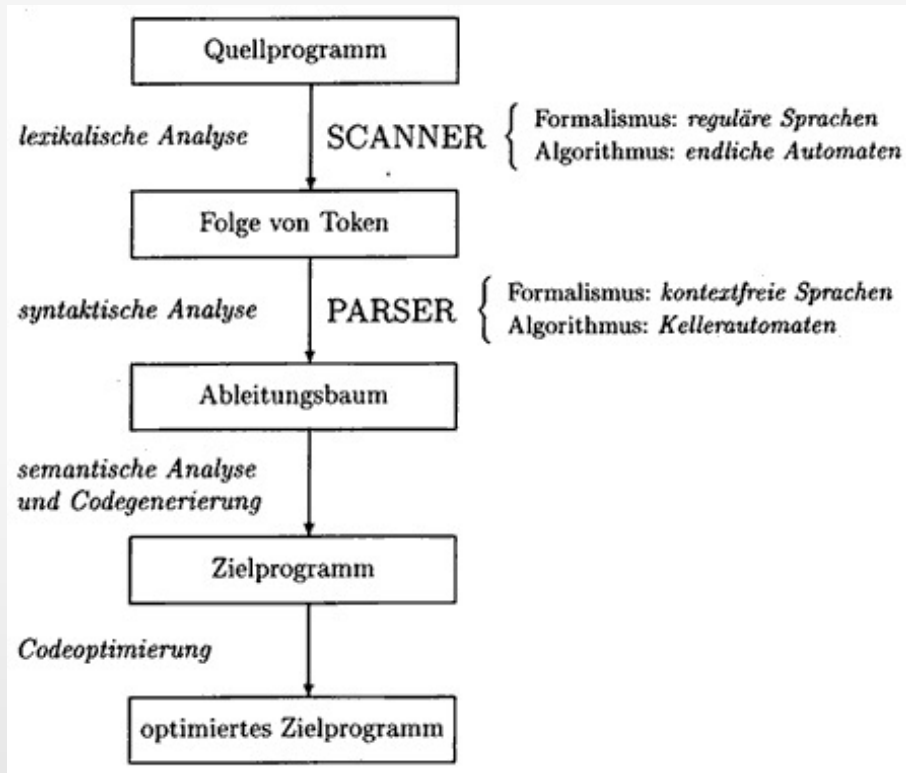
- **Typ 0** (allgemein) → **Turing-Maschine**
- **Typ 1** (kontextsensitiv) → **linear beschränkter Automat**
- **Typ 2** (kontextfrei) → **Kellerautomat**
- **Typ 3** (regulär) → **endlicher Automat**



# Grammatiken und die Praxis

## Wofür der Aufwand?

- **Compilierung:** Stellt die Eingabe ein zu akzeptierendes Programm dar?

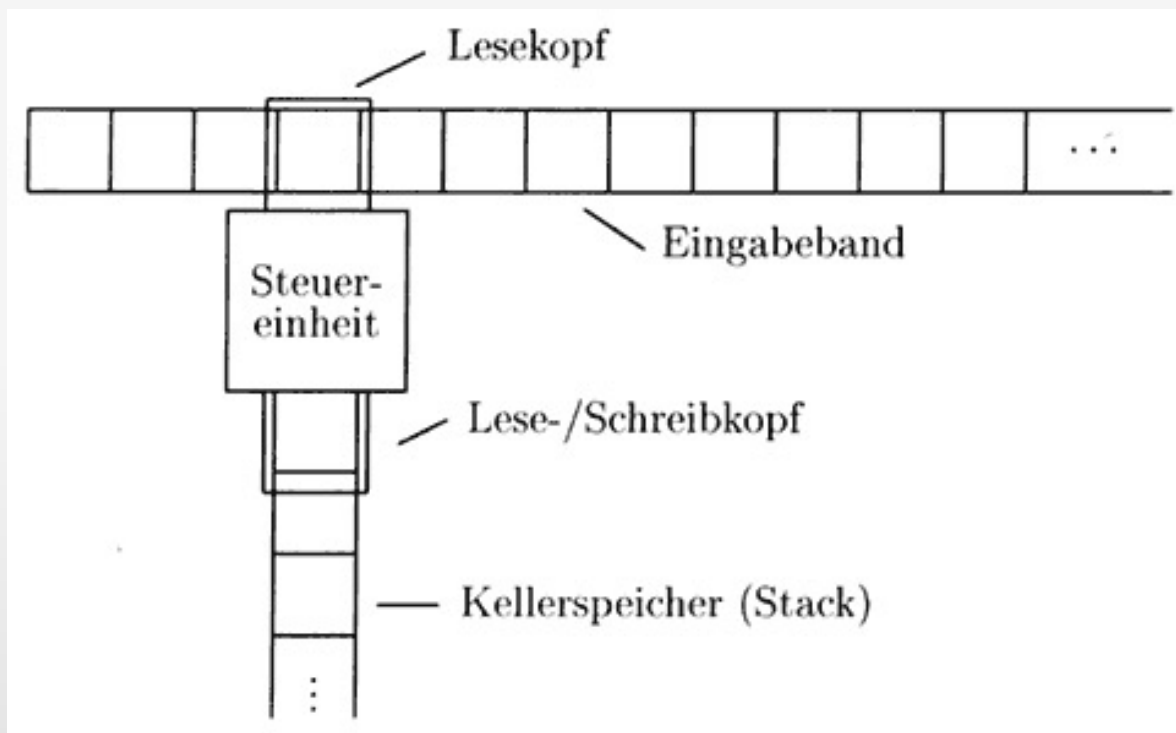


Kellerautomat

# Kellerautomat

Intention: Endlichen Automaten fehlt ein Speicher, der beliebig viele Informationen speichern kann und nicht – wie DEA – von vornherein, d.h. durch die Anzahl ihrer Zustände, beschränkt ist.

Lösung: **Kellerautomat**



# Kellerautomat

Formale Definition: Ein nichtdeterministischer Kellerautomat (mit Endzuständen) ist ein **7-Tupel**):

$$M = (Z, V, U, \delta, q_M, K_M, F)$$

- **Z** ist die Zustandsmenge
- **V** ist das Eingabealphabet
- **U** ist das Kelleralphabet.
- **$\delta$**  (Delta) ist die Überführungsrelation. Die Elemente aus  $\delta$  heißen Anweisungen.
- **$q_M$**  ist der Startzustand. Der Kellerautomat befindet sich in diesem Zustand, bevor ein Zustandsübergang erfolgt ist.
- **$K_M$**  ist das Anfangskellerzeichen. Anfangs enthält der Kellerspeicher nur das Startsymbol.
- **F** ist die Menge der akzeptierenden Zustände oder Endzustände.

# Turing-Maschine

# Turing-Maschine



Anforderung: Entwurf eines allgemeineren Automaten

Auflösung der Beschränkungen des Kellerspeichers, u.a. dass nur an dem einen Ende des Speichers ein- und ausgelesen werden kann

→ Automat, der wahlfreien Zugang (random access) auf alle Speicherplätze erlaubt

Formuliert von Alan M. Turing in „On computable numbers, with an application to the Entscheidungsproblem“ (1937)

# Turing-Maschine

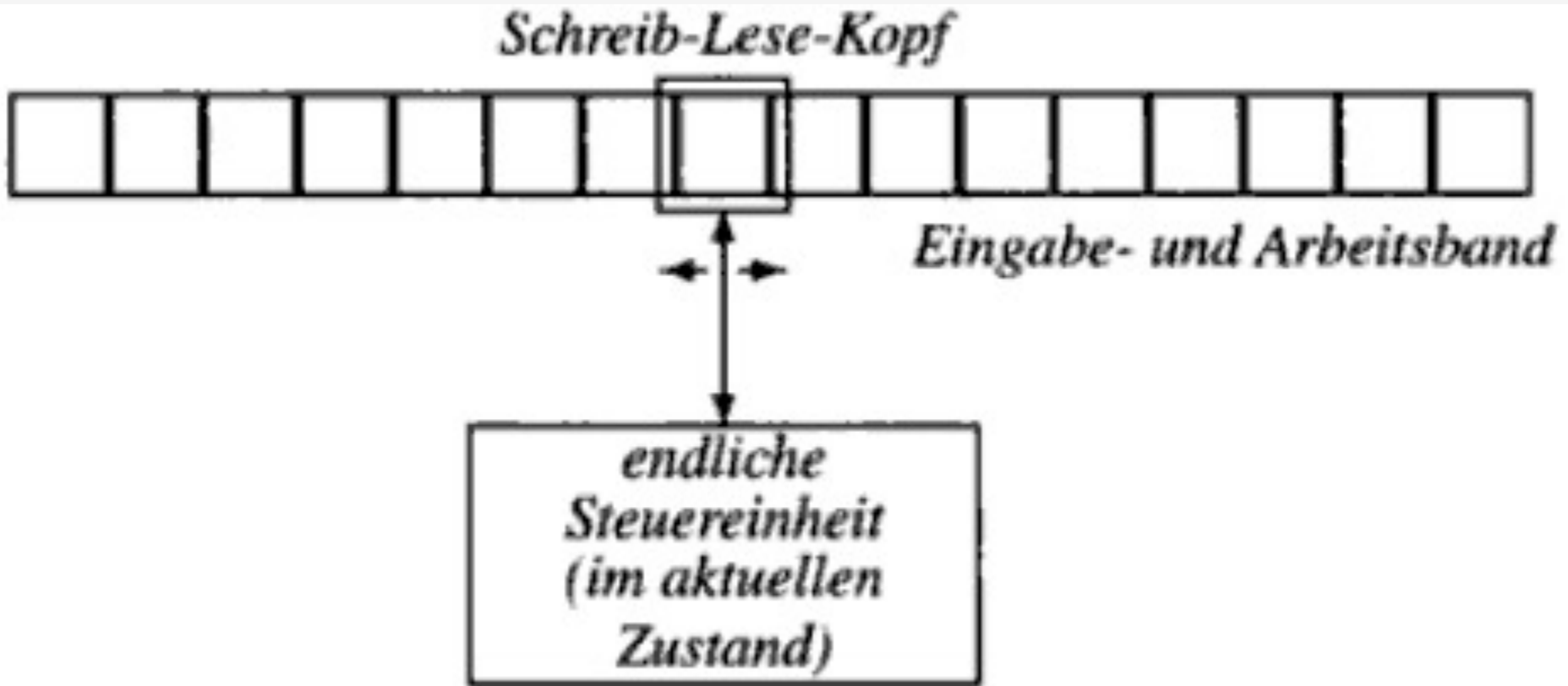


Abb.: Schöning, Uwe: Ideen der Informatik. München 2008, Oldenbourg Wissenschaftsverlag. S. 102.

/