

# Theoretische Grundlagen der Informatik

## Vorlesung am 16.10.2018

INSTITUT FÜR THEORETISCHE INFORMATIK



## Organisatorisches

- Team, Termine
- Vorlesungsaufzeichnungen
- Homepage, Literaturempfehlungen
- Übungsblätter, Klausur
- Tutorien, Einteilung

## Wiederholung aus GBI

- Wörter, formale Sprachen
- Reguläre Sprachen, reguläre Ausdrücke
- Endliche Automaten
- Kontextfreie Grammatiken

## Vorlesung:

Torsten Ueckerdt

## Übung:

Guido Brückner  
[brueckner@kit.edu](mailto:brueckner@kit.edu)

Jonas Sauer  
[jonas.sauer2@kit.edu](mailto:jonas.sauer2@kit.edu)

## Tutorium:

Mo	8:00	Jasper Marianczuk	Mi	11:30	Miriam Goetze
Mo	15:45	Jan Ellmers	Mi	15:45	Jean-Pierre von der Heydt
Mo	15:45	René-Maurice Hanne	Mi	15:45	Florian Leiser
Di	8:00	Markus Himmel	Do	8:00	Eric Hamann
Di	8:00	Lukas Hübner	Do	9:45	Moritz Potthoff
Di	14:00	Liran Dattner	Do	9:45	Tina Strößner
Di	14:00	Jan Ellmers	Do	14:00	Vasil Papanchev
Di	17:30	Vera Chekan	Do	15:45	Nils Froleyks
Mi	8:00	Michael Schrempp	Do	15:45	Laura Merker
Mi	8:00	Laurin Benz	Do	17:30	Nils Froleyks
Mi	9:45	Laurin Benz	Fr	9:45	Florian Lanzinger
Mi	11:30	Michael Schrempp	Fr	11:30	Max Ruff

# Termine (Änderungen vorbehalten)

## Dienstags

16.10. Vorlesung  
23.10. Vorlesung  
30.10. Vorlesung  
06.11. Vorlesung  
13.11. Übung  
20.11. Vorlesung  
27.11. Übung  
04.12. Vorlesung  
11.12. Übung  
18.12. Vorlesung  
08.01. Übung  
15.01. Vorlesung  
22.01. Übung  
29.01. Vorlesung  
05.02. Übung

## Donnerstags

18.10. Vorlesung  
25.10. Übung  
01.11. **Feiertag**  
08.11. Vorlesung  
15.11. Vorlesung  
22.11. Vorlesung  
29.11. Vorlesung  
06.12. Vorlesung  
13.12. Vorlesung  
20.12. Vorlesung  
10.01. Vorlesung  
17.01. Vorlesung  
24.01. Vorlesung  
31.01. Vorlesung  
07.02. Vorlesung

- Ton und Folien der Vorlesungen werden aufgezeichnet.
- Aufnahmen sind abrufbar über YouTube Kanal [KIT | WEBCAST](#):
  - <https://www.youtube.com/channel/UC6AqaL6fH91U5YhyUCIOZ0Q>
  - Aufnahmen werden nach den Vorlesungen online gestellt (mit gewisser Verzögerung).
  - Direkte Links auf der Vorlesungshomepage

- <https://i11www.iti.kit.edu/teaching/winter2018/tgi/index>
- Aktuelle Informationen / Termine
- Alte Klausuren
- Skript
- Folien
- Übungsblätter
- ILIAS-Forum
  - Für Fragen an die Übungsleiter
  - Für Austausch untereinander
  - [https://ilias.studium.kit.edu/goto\\_produkativ\\_crs\\_876431.html](https://ilias.studium.kit.edu/goto_produkativ_crs_876431.html)
- Links zu Vorlesungsaufzeichnungen
- Literaturempfehlungen

- Ingo Wegener: **Theoretische Informatik**  
B.G. Teubner Verlag Stuttgart, 1993
- Uwe Schöning: **Theoretische Informatik - kurzgefasst**  
Hochschultaschenbuch, Spektrum Akademischer Verlag, 1997
- R. Garey und D. S. Johnson:  
**Computers and Intractability: A Guide to the Theory of NP-Completeness**  
W. H. Freeman, New York, 1979
- T. Cormen, C. Leiserson, R. Rivest: **Introduction to Algorithms**  
The MIT press, 1997, 2001.
- A. Asteroth, C. Baier:  
**Theoretische Informatik: eine Einführung in Berechenbarkeit, Komplexität und formale Sprachen mit 101 Beispielen**  
Pearson Studium, 2002

## In der Regel:

- Ausgabe etwa jeden zweiten Dienstag
- In derselben Übung: Besprechung von ähnlichen Aufgaben
- Bearbeitungszeit: 1-2 Wochen (siehe Abgabedatum auf Übungsblatt!)
- Abgabe: Kasten im Untergeschoss des Informatik-Hauptgebäudes (50.34)
- Abgabe bis Dienstag, 11.00 Uhr im Kasten
- Rückgabe und Besprechung der korrigierten Blätter in den Tutorien
- Ausgabe des ersten Übungsblatts: [Donnerstag 18.10.2018](#)

- Doppelabgabe erlaubt, aber nur, wenn beide für das gleiche Tutorium eingeteilt sind
- Deckblatt bitte mit [WebInScribe Deckblattgenerator](#)
  - <https://webinscribe.ira.uka.de/deckblatt/index.php?course=10696>
- Keine losen Blätter, bitte zusammenheften
- Übungsblätter müssen handschriftlich sein, also nicht gedruckt oder kopiert
- Bei Abschreiben: Keine Punkte
- 25% der möglichen Gesamtpunktzahl  $\rightsquigarrow$  **1 Bonuspunkt**
- 50% der möglichen Gesamtpunktzahl  $\rightsquigarrow$  **2 Bonuspunkte**
- 75% der möglichen Gesamtpunktzahl  $\rightsquigarrow$  **3 Bonuspunkte**
- Der Klausurbonus wird nur auf bestandene Klausuren angerechnet

- Hauptklausur: **25. März 2019** um 12:00 Uhr.
- Dauer: 2 Stunden
- Orientierung: Altklausuren auf der Homepage
- Termin für Nachklausur wird noch bekannt gegeben.

- Rückgabe und Besprechung der Übungsblätter
- Zusätzliche Aufgaben und Beispiele zum Stoff der Vorlesung
- Start: Nächste Woche (ab 22.10.2018)
- Einteilung über WebInScribe
  - <http://webinscribe.informatik.kit.edu/>
- Dort verlinkt: Merkblatt und Termine der Tutorien
- Anmeldebeginn: Dienstag, 16.10., 18:00 Uhr  $\rightsquigarrow$  heute!
- Anmeldeschluss: Donnerstag, 18.10., **18:00 Uhr**

Welche Fragestellungen werden in TGI behandelt?

- Theoretische Grundlagen zu Algorithmen- und Programmentwurf:  
Wie kann man allgemeingültige (rechner- und programmiersprachenunabhängige) Aussagen zu gegebenen Problemstellungen machen?
- Gibt es Probleme die nicht von Computern gelöst werden können?
- Gibt es Probleme, für die Ausprobieren die beste Lösungsstrategie ist?
- Wie kann man konkrete Computer abstrakt betrachten oder modellieren?
- Wie kann man konkrete Programmiersprachen abstrakt betrachten oder modellieren?

- Vertiefter Einblick in die Grundlagen der Theoretischen Informatik
- Beherrschen der Berechnungsmodelle und Beweistechniken der TI
- Verständnis für Grenzen und Möglichkeiten der Informatik in Bezug auf die Lösung von definierbaren aber nur bedingt berechenbaren Problemen
- Abstraktionsvermögen für grundlegende Aspekte der Informatik von konkreten Gegebenheiten wie konkreten Rechnern oder Programmiersprachen
- Anwendung der erlernten Beweistechniken bei der Spezifikation von Systemen der Informatik und für den systematischen Entwurf von Programmen und Algorithmen

- Wörter
- Formale Sprachen
- Reguläre Ausdrücke
- Endliche Automaten
- Kontextfreie Grammatiken

- Ein *Alphabet*  $\Sigma$  ist eine endliche Menge von Zeichen.

Beispiele:  $\Sigma = \{0, 1\}$  oder  $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \cup \{, \} \cup \{.\}$ .

- Ein *Wort*  $w$  über einem Alphabet  $\Sigma$  ist eine (möglicherweise leere) Folge von Zeichen aus  $\Sigma$ .

Beispiel:  $\Sigma = \{0, 1\}$  und  $w = 0010010$ .

$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \cup \{, \} \cup \{.\}$  und  $w = 1.024, 48$

- Das *leere Wort* wird mit  $\varepsilon$  symbolisiert.
- Die *Menge aller Wörter* über einem Alphabet  $\Sigma$  wird mit  $\Sigma^*$  abgekürzt.

Beispiel:  $\Sigma = \{0, 1\}$  und  $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \dots\}$

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

$\rightsquigarrow \varepsilon$  und  $1..0.4$  und  $.,.7$  etc. in  $\Sigma^*$

- Die *Menge aller Wörter* mit Länge  $n$  über einem Alphabet  $\Sigma$  wird mit  $\Sigma^n$  abgekürzt.

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

$$\Sigma^0 = \{\varepsilon\}$$

$$\Sigma^1 = \{0, 1\}$$

$$\Sigma^2 = \{00, 01, 10, 11\}$$

- Die *Konkatenation* (Aneinanderhängen) zweier Wörter  $w_1$  und  $w_2$  wird mit  $w_1 \cdot w_2$  oder  $w_1 w_2$  abgekürzt.

Beispiel:  $w_1 = 001$  und  $w_2 = 110$  dann  $w_1 \cdot w_2 = 001110$ .

$w_1 = 001$  und  $w_2 = \varepsilon$  dann  $w_1 \cdot w_2 = 001$ .

- Die *iterierte Konkatenation* eines Wortes  $w$  ist  $w^k := \underbrace{w \cdot \dots \cdot w}_{k\text{-mal}}$

Beispiel:  $w = 110$ .

$$w^0 = \varepsilon$$

$$w^1 = 110$$

$$w^2 = 110 \cdot 110 = 110110$$

$$w^3 = 110 \cdot 110 \cdot 110 = 110110110$$

- Eine *formale Sprache*  $L$  über einem Alphabet  $\Sigma$  ist eine Teilmenge  $L \subseteq \Sigma^*$ .

Beispiel: Sprache  $L'$  aller Wörter deren vorletztes Zeichen 0 ist

$$L' = \{w0z \mid w \in \Sigma^*, z \in \Sigma\}$$

- Das Produkt  $L_1 \cdot L_2$  der Sprachen  $L_1$  und  $L_2$  ist definiert als

$$L_1 \cdot L_2 := \{w_1 w_2 \mid w_1 \in L_1 \text{ und } w_2 \in L_2\}$$

Beispiel:

$$L' = \Sigma^* \cdot \{0\} \cdot \Sigma$$

- Produkte einer Sprache  $L$  mit sich selbst werden abgekürzt als

$$L^k := \underbrace{L \cdot \dots \cdot L}_{k\text{-mal}}$$

Beispiel  $L = \{00, 1\}$ :

$$L^0 = \{\varepsilon\}$$

$$L^1 = \{00, 1\}$$

$$L^2 = \{00, 1\} \cdot \{00, 1\} = \{0000, 001, 100, 11\}$$

$$\begin{aligned} L^3 &= \{0000, 001, 100, 11\} \cdot \{00, 1\} \\ &= \{000000, 00100, 10000, 1100, 00001, 0011, 1001, 111\} \end{aligned}$$

Läßt sich ein Wort  $w$  schreiben als  $w = u \cdot v \cdot x$ , wobei  $u, v, x$  beliebige Wörter sind, so heißt:

$u$	Präfix	} von $w$
$v$	Teilwort	
$x$	Suffix	

Beispiel  $w = \text{TAL}$

Präfixe	$P = \{\varepsilon, T, TA, TAL\}$
Suffixe	$S = \{\varepsilon, L, AL, TAL\}$
Teilworte	$\{A\} \cup P \cup S$

Seien  $L, L_1, L_2 \subseteq \Sigma^*$  Sprachen.

Produktsprache  $L_1 \cdot L_2 := \{w_1 \cdot w_2 \mid w_1 \in L_1, w_2 \in L_2\}$

$k$ -faches Produkt  $L^k := \{w_1 \cdot \dots \cdot w_k \mid w_i \in L \text{ für } 1 \leq i \leq k\}$

$$L^1 = L, \quad L^0 := \{\varepsilon\}$$

Kleene'scher Abschluss  $L^* := \bigcup_{i \geq 0} L^i$

Positiver Abschluss  $L^+ := \bigcup_{i \geq 1} L^i$

Quotientensprache  $L_1 / L_2 := \{w \in \Sigma^* \mid \exists z \in L_2 \text{ mit } w \cdot z \in L_1\}$

Komplementsprache  $L^c := \Sigma^* \setminus L$

Eine Sprache  $L \subseteq \Sigma^*$  heißt *regulär*, wenn für sie einer der folgenden Punkte gilt: (induktive Definition)

- Verankerung:

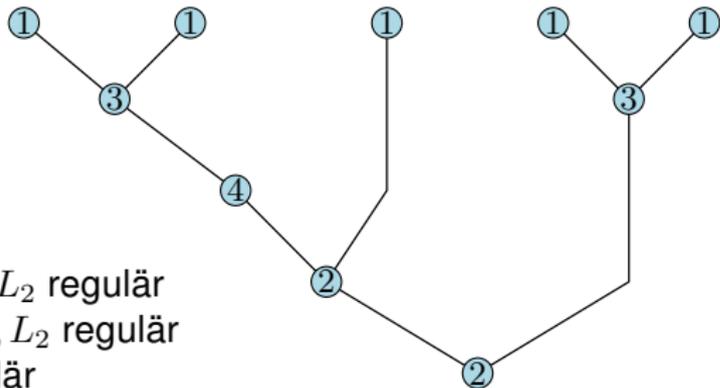
- $L = \{a\}$  mit  $a \in \Sigma$  oder
- $L = \{\varepsilon\}$  oder
- $L = \emptyset$

- Induktion: Es gibt reguläre Sprachen  $L_1, L_2$  sodass

- $L = L_1 \cdot L_2$  oder
- $L = L_1 \cup L_2$  oder
- $L = L_1^*$

Beispiel: Die Sprache  $L'$  aller Wörter über  $\{0, 1\}$ , die als vorletztes Zeichen eine 0 haben

$$L' := (\{0\} \cup \{1\})^* \cdot \{0\} \cdot (\{0\} \cup \{1\})$$



Verankerung:

①  $L = \{a\}$  mit  $a \in \Sigma$

Induktion:

②  $L = L_1 \cdot L_2$  mit  $L_1, L_2$  regulär

③  $L = L_1 \cup L_2$  mit  $L_1, L_2$  regulär

④  $L = L_1^*$  mit  $L_1$  regulär

- Wir benutzen eine leicht andere Schreibweise für reguläre Ausdrücke als in GBI.
- Sei  $\Sigma$  eine Alphabet. Eine reguläre Sprache über  $\Sigma$  kann durch einen *regulären Ausdruck* beschrieben werden.

Dabei bezeichnet

- $\emptyset$  den regulären Ausdruck, der die leere Menge beschreibt.

$$\rightsquigarrow L(\emptyset) = \{\}$$

- $\varepsilon$  den regulären Ausdruck, der die Menge  $\{\varepsilon\}$  beschreibt.

$$\rightsquigarrow L(\varepsilon) = \{\varepsilon\}$$

- $a$  den regulären Ausdruck, der die Menge  $\{a\}$  beschreibt.

$$\rightsquigarrow L(a) = \{a\}$$

Wenn  $\alpha, \beta$  reguläre Ausdrücke sind, die die Sprachen  $L(\alpha), L(\beta)$  beschreiben, so schreiben wir

- $(\alpha) \cup (\beta)$  für  $L(\alpha) \cup L(\beta)$

- $(\alpha) \cdot (\beta)$  für  $L(\alpha) \cdot L(\beta)$

- $(\alpha)^+$  für  $L(\alpha)^+$

- $(\alpha)^*$  für  $L(\alpha)^*$

## Notation

- Wir schreiben auch  $\alpha$  statt  $L(\alpha)$  und  $w \in \alpha$  statt  $w \in L(\alpha)$ .
- \* bindet stärker als  $\cdot$  und  $\cdot$  stärker als  $\cup$
- Das heißt zum Beispiel:

$$a \cup b \cdot c = a \cup (b \cdot c) = a \cup bc$$

- Wir lassen Konkatenationspunkte und unnötige Klammern weg.

- $L'$  ist der reguläre Ausdruck für die Sprache aller Wörter über  $\{0, 1\}$ , die als vorletztes Zeichen eine 0 haben

$$L' = (0 \cup 1)^* 0 (0 \cup 1)$$

- $L := \{w \in \{0, 1\}^* \mid w \text{ enthält } 10 \text{ als Teilwort}\}$

$$L = (0 \cup 1)^* 10 (0 \cup 1)^*$$

- $L := \{w \in \{0, 1\}^* \mid w \text{ enthält } 101 \text{ als Teilwort}\}$

$$L = (0 \cup 1)^* 101 (0 \cup 1)^*$$

- $L := \{w \in \{0, 1\}^* \mid w \text{ enthält } 10 \text{ nicht als Teilwort}\} = 0^*1^*$ ,  
denn
  - Sei  $w \in 0^*1^*$ . Dann kommt in  $w$  nach keiner 1 eine 0.  
Also ist  $w \in L$  und damit  $0^*1^* \subseteq L$ .
  - Sei  $w \in L$ . Dann kommen nach der ersten Eins keine Nullen mehr vor,  
d.h.  $w = w'1 \dots 1$  wobei  $w'$  keine 1 enthält.  
Also ist  $w \in 0^*1^*$  und damit  $L \subseteq 0^*1^*$ .

- $L := \{w \in \{0, 1\}^* \mid w \text{ enthält 10 nicht als Teilwort}\} = 0^*1^*$ ,  
denn
  - Sei  $w \in 0^*1^*$ . Dann kommt in  $w$  nach keiner 1 eine 0.  
Also ist  $w \in L$  und damit  $0^*1^* \subseteq L$ .
  - Sei  $w \in L$ . Dann kommen nach der ersten Eins keine Nullen mehr vor,  
d.h.  $w = w'1 \dots 1$  wobei  $w'$  keine 1 enthält.  
Also ist  $w \in 0^*1^*$  und damit  $L \subseteq 0^*1^*$ .

## Testen Sie sich:

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

Sei  $L_{\mathbb{R}} \subseteq \Sigma^*$  die Sprache aller endlichen Dezimaldarstellungen von positiven reellen Zahlen.

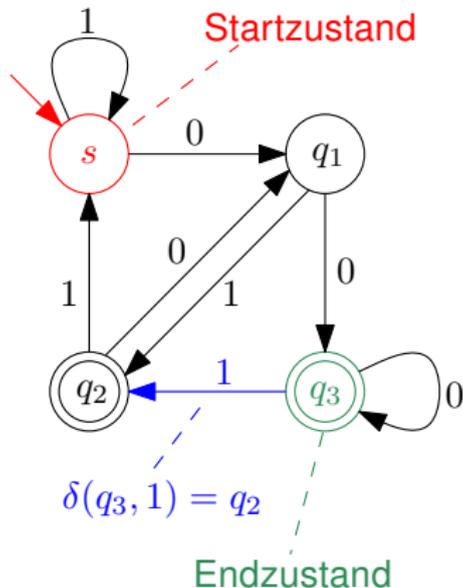
Beispiele:  $1.024, 48 \in L_{\mathbb{R}}$  und  $42 \in L_{\mathbb{R}}$   
aber  $1024, 48 \notin L_{\mathbb{R}}$  und  $042, 00 \notin L_{\mathbb{R}}$  und  $., .123 \notin L_{\mathbb{R}}$

$\rightsquigarrow$  Finden Sie einen regulären Ausdruck für  $L_{\mathbb{R}}$ ?

- In GBI wurden Mealy- und Moore-Automaten behandelt.
- In TGI werden nur endliche Akzeptoren benötigt.

Ein (deterministischer) endlicher Automat DEA  $(Q, \Sigma, \delta, s, F)$  besteht aus:

- $Q$ , einer endlichen Menge von *Zuständen*
- $\Sigma$ , einer endlichen Menge von *Eingabesymbolen*
- $\delta: Q \times \Sigma \rightarrow Q$ , einer *Übergangsfunktion*
- $s \in Q$ , einem *Startzustand*;
- $F \subseteq Q$ , einer Menge von *Endzuständen*.



Der Automat heißt

- endlich, da die Zustandsmenge (vgl. mit Speicher, Gedächtnis) endlich ist.
- deterministisch, da  $\delta$  eine Funktion ist und der Automat somit in jedem Schritt eindeutig arbeitet. Es gibt keine Zufälligkeiten oder Wahlmöglichkeiten.

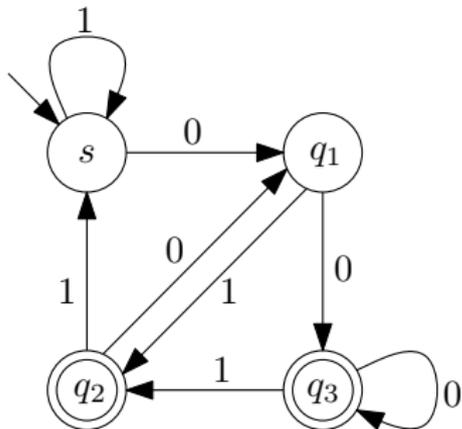
Was kann ein endlicher Automat?

- Gegeben ist eine Eingabe als endliche Folge von Eingabesymbolen. Der Automat entscheidet, ob die Eingabe zulässig ist oder nicht, indem er in einem Endzustand endet oder nicht.

- Ein endlicher Automat **erkennt** oder **akzeptiert** eine Sprache  $L$ , d.h. eine Menge von Wörtern über dem Alphabet  $\Sigma$  des Automaten, wenn er nach Abarbeitung eines Wortes  $w$  genau dann in einem Endzustand ist, wenn das Wort  $w$  in der Sprache  $L$  ist ( $w \in L$ ).
- Eine formale Sprache heißt *endliche Automatensprache*, wenn es einen endlichen Automaten gibt, der sie erkennt.

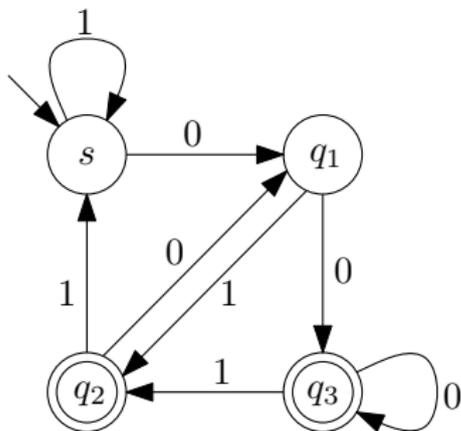
Der Automat erkennt  
 $w = 001$ .

Der Automat erkennt nicht  
 $w = 110$ .



Der Automat erkennt  
 $w = 001$ .

Der Automat erkennt nicht  
 $w = 110$ .



Der Automat erkennt die Sprache  $L'$  aller Wörter, deren vorletztes Zeichen 0 ist:

$$L' = (0 \cup 1)^* 0 (0 \cup 1)$$

$\rightsquigarrow L'$  ist eine endliche Automatensprache.

Eine kontextfreie Grammatik  $G = (\Sigma, V, S, R)$  ist gegeben durch

- ein endliches Alphabet  $\Sigma$  (auch Terminalalphabet genannt)
- eine endlichen Menge  $V$  mit  $V \cap \Sigma = \emptyset$  von Variablen (auch Nichtterminale genannt)
- ein Startsymbol  $S \in V$
- eine endlichen Menge von Ableitungsregeln  $R$ ,  
d.h. durch eine Menge von Tupeln  $(l, r) \in V \times (\Sigma \cup V)^*$   
(auch Produktionen genannt)

Wir schreiben Produktionen auch in der Form  $l \rightarrow r$ .

Gegeben ist eine Regel  $l \rightarrow r$ .

- Wenn in einem Wort  $w$  das Zeichen  $l$  vorhanden ist, darf  $l$  in  $w$  durch  $r$  ersetzt werden.
- Wir schreiben  $w \rightarrow z$ , wenn  $w$  durch Anwendung *einer* Ableitungsregel in  $z$  umgewandelt werden kann.
- Wir schreiben  $w \xrightarrow{*} z$ , wenn  $w$  durch Anwendung einer, keiner oder *mehrerer* Ableitungsregeln in  $z$  umgewandelt werden kann.

Die von einer Grammatik  $G = (\Sigma, V, S, R)$  erzeugte Sprache  $L(G)$  ist die Menge aller Wörter  $z \in \Sigma^*$  für die gilt  $S \xrightarrow{*} z$ .

# Beispiel Kontextfreie Grammatiken

Gegeben ist die Grammatik  $G = (\Sigma, V, S, R)$  mit

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

$$V := \{S\}$$

$$R := \{S \rightarrow 01, S \rightarrow 0S1\} \text{ wir schreiben dafür kurz } \{S \rightarrow 01|0S1\}$$

# Beispiel Kontextfreie Grammatiken

Gegeben ist die Grammatik  $G = (\Sigma, V, S, R)$  mit

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

$$V := \{S\}$$

$$R := \{S \rightarrow 01, S \rightarrow 0S1\} \text{ wir schreiben dafür kurz } \{S \rightarrow 01|0S1\}$$

Welche Wörter erzeugt  $G$ , d.h. was ist  $L(G)$ ?

# Beispiel Kontextfreie Grammatiken

Gegeben ist die Grammatik  $G = (\Sigma, V, S, R)$  mit

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

$$V := \{S\}$$

$$R := \{S \rightarrow 01, S \rightarrow 0S1\} \text{ wir schreiben dafür kurz } \{S \rightarrow 01|0S1\}$$

Welche Wörter erzeugt  $G$ , d.h. was ist  $L(G)$ ?

$$\begin{aligned} S &\rightarrow 01|0S1 \rightarrow 01|0011|00S11 \\ &\rightarrow 01|0011|000111|000S111 \rightarrow \dots \end{aligned}$$

# Beispiel Kontextfreie Grammatiken

Gegeben ist die Grammatik  $G = (\Sigma, V, S, R)$  mit

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

$$V := \{S\}$$

$$R := \{S \rightarrow 01, S \rightarrow 0S1\} \text{ wir schreiben dafür kurz } \{S \rightarrow 01|0S1\}$$

Welche Wörter erzeugt  $G$ , d.h. was ist  $L(G)$ ?

$$\begin{aligned} S &\rightarrow 01|0S1 \rightarrow 01|0011|00S11 \\ &\rightarrow 01|0011|000111|000S111 \rightarrow \dots \end{aligned}$$

Es ist  $L(G) = \{0^n 1^n \mid n \in \mathbb{N}\}$ .

# Beispiel Kontextfreie Grammatiken

Welche Grammatik erzeugt die Sprache über dem Alphabet  $\Sigma = \{0, 1\}$  deren vorletztes Zeichen 0 ist?

# Beispiel Kontextfreie Grammatiken

Welche Grammatik erzeugt die Sprache über dem Alphabet  $\Sigma = \{0, 1\}$  deren vorletztes Zeichen 0 ist?

Grammatik  $G = (\Sigma, V, S, R)$  mit

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

$$V := \{S, A, B\}$$

$$R := \{S \rightarrow A0|A1, \quad A \rightarrow B0, \quad B \rightarrow \varepsilon|B0|B1\}$$

Ableitung:

$$S \rightarrow A0|A1 \rightarrow B00|B01 \rightarrow \dots$$

In TGI werden wir Grammatiken kennenlernen, deren Produktionen *mächtiger* sind, als die von kontextfreien Grammatiken.

Wir werden auch Sprachen kennenlernen, die nicht von Grammatiken erzeugt werden können.