



## Einführung in die Theoretische Informatik

David Drexel      Alexander Maringele  
Julian Fodor      David Obwaller  
Alexander Lochmann    Jonas Schöpf

**Georg Moser**

cbr.uibk.ac.at

## Zusammenfassung

## Zusammenfassung der letzten LVA

### Definition

Eine **Grammatik**  $G$  ist ein Quadrupel  $G = (V, \Sigma, R, S)$ , wobei

- 1  $V$  eine endliche Menge von **Variablen** (oder **Nichtterminale**)
- 2  $\Sigma$  ein Alphabet, die **Terminale**,  $V \cap \Sigma = \emptyset$
- 3  $R$  eine endliche Menge von **Regeln**
- 4  $S \in V$  das **Startsymbol** von  $G$

### Definition

Eine formale Sprache  $L$  heißt

- **beschränkt** wenn  $\exists$  beschränkte Grammatik  $G$ ,  $L = L(G)$
- **rekursiv aufzählbar** (vom **Typ 0**)  
wenn  $\exists$  Grammatik  $G$ ,  $L = L(G)$

## Feedback

*Frage zu Aufgabe 3) von SL-Blatt 5: Wie sollte man denn die Sprachen beschreiben? In natürlicher Sprache ist es unexakt. Und wenn man auf eine Menge der Sprache kommt: Wie kann man dann sicher sein, dass es sich bei der erzeugten Sprache tatsächlich um diese Menge handelt? Müsste man das beweisen? Wenn ja: Wie?*

*Zudem ist es mir schwer gefallen, „leichtere“ Grammatiken für gegebene Sprachen zu finden. Auch hier: Wie kann man sicher sein, dass eine gegebene Grammatik tatsächlich die gesamte Sprache erzeugt? Und wie kann man sicher sein, dass die gefundene Grammatik tatsächlich die „minimalste“ ist?*



## Reguläre Sprachen

### Beispiel

Wir untersuchen Protokolle, die den Gebrauch elektronischen „Geldes“ ermöglichen; dabei handeln: der **Kunde**, die **Bank** und das **Geschäft**:

- Der Kunde kann **zahlen**
- Der Kunde kann das Geld **löschen**
- Das Geschäft kann dem Kunden Waren **zusenden**
- Das Geschäft kann Geld **einlösen**
- Die Bank kann Geld **überweisen**

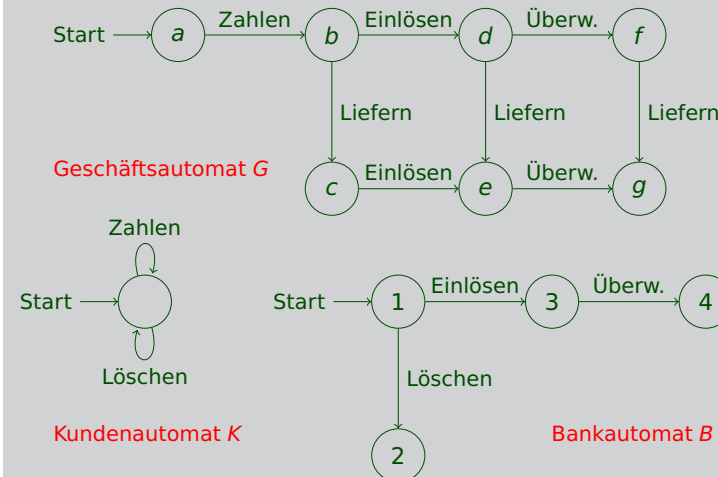
Wir treffen die folgenden **Grundannahmen**:

- Der Kunde ist **unverantwortlich**
- Das Geschäft ist **verantwortlich**, aber **gutgläubig**
- Die Bank ist **strikt**

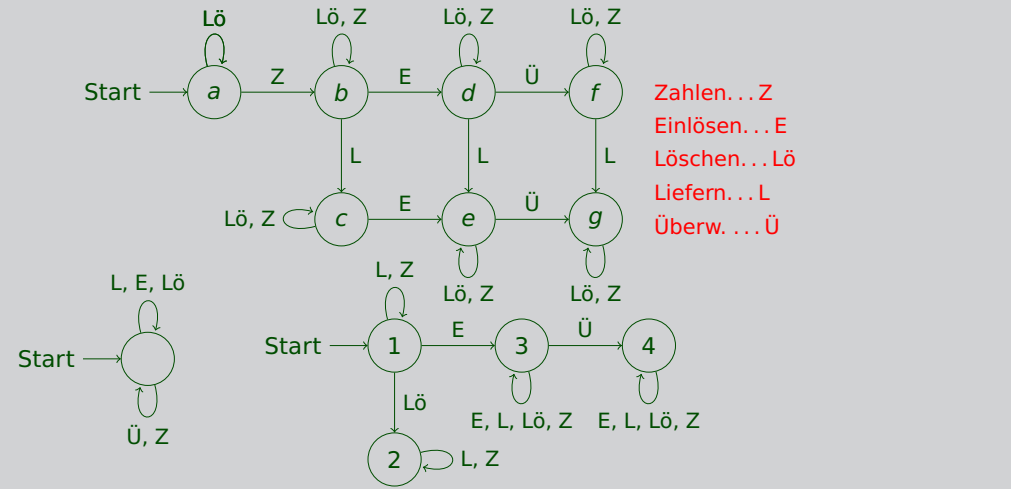
## Anwendungen von Endlichen Automaten

- Software zum Entwurf und Testen von digitalen Schaltkreisen (Mealy/Moore Automaten)
- Softwarebausteine eines Compilers, etwa in der lexikalischen Analyse:
  - 1 lexikalische Scanner („Lexer“) wird mit endlichen Automaten implementiert
  - 2 Der lexikalische Scanner dient zur Aufteilung des Eingabetextes in logische Einheiten, wie Bezeichner oder Schlüsselwörter
- Software zum Durchsuchen umfangreicher Texte
- **Software zur Verifizierung aller Arten von Systemen, die eine endliche Anzahl verschiedener Zustände besitzen**
- Softwarebausteine eines Computerspiels:
  - 1 Kontrolle von Spielfiguren kann mit Hilfe eines endlichen Automaten implementiert werden
  - 2 erlaubt eine bessere Modularisierung des Codes

### Beispiel (Fortsetzung)



### Beispiel (Fortsetzung)



### Beispiel (Fortsetzung)

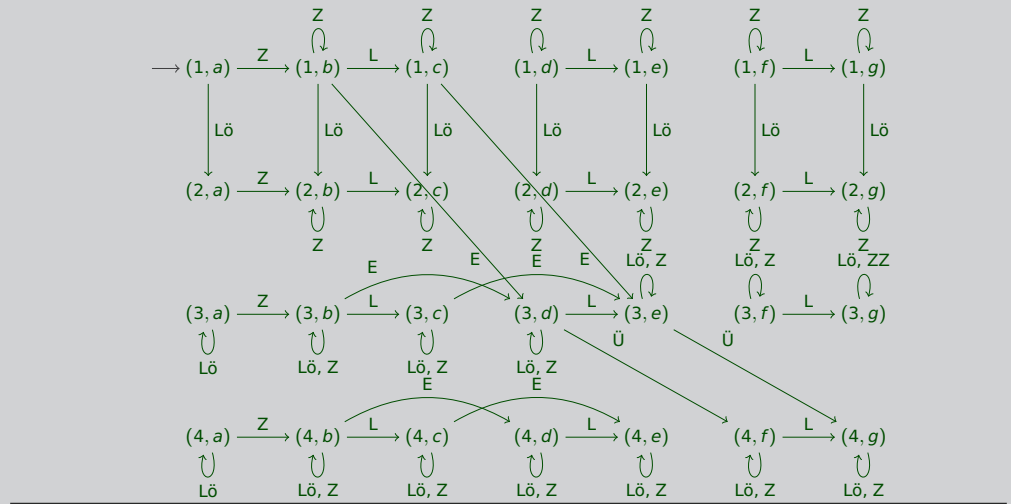
Wir definieren den **Produktautomaten**  $B \times G$  aus  $B$  und  $G$ :

- 1 Die **Zustände** dieses Automaten sind:  
 $(i, x)$  wobei  $i \in \{1, 2, 3, 4\}, x \in \{a, b, c, d, e, f, g\}$
- 2 Die **Übergänge** werden durch **paralleles** Ausführen von  $B$  und  $G$  definiert:  
wenn  $i \xrightarrow{\text{Aktion}} i'$  und  $x \xrightarrow{\text{Aktion}} x'$  dann  $(i, x) \xrightarrow{\text{Aktion}} (i', x')$

Betrachte  $B$  und  $G$ :

- in  $B$  gilt, dass aus Zustand 1 durch die Aktion „Einlösen“ Zustand 3 wird, konzise:  $1 \xrightarrow{\text{Einlösen}} 3$
- in  $G$  gilt, dass aus Zustand  $b$  mit Aktion „Einlösen“  $d$  wird  
Konzise:  $b \xrightarrow{\text{Einlösen}} d$
- also gilt in  $B \times G$ , dass aus Zustand  $(1, b)$  mit Aktion „Einlösen“ Zustand  $(3, d)$  wird, konzise:  $(1, b) \xrightarrow{\text{Einlösen}} (3, d)$

### Beispiel (Fortsetzung)



### Beispiel (Fortsetzung)

Als Schlussfolgerung ergibt sich, dass das Protokoll nicht sicher ist:  
*Der Automat  $B \times G$  kann in den Zustand  $(2, c)$  gelangen, in welchem die Waren geschickt wurden und trotzdem nie eine Überweisung an das Geschäft erfolgen wird*

### Definition (Deterministischer endlicher Automat (kurz: DEA))

Ein **DEA** ist ein 5-Tupel  $A = (Q, \Sigma, \delta, q_0, F)$  sodass

- 1  $Q$  eine endliche Menge von **Zuständen**
- 2  $\Sigma$  eine endliche Menge von **Eingabesymbolen**, ( $\Sigma$  wird auch **Eingabealphabet** genannt)
- 3  $\delta: Q \times \Sigma \rightarrow Q$  die **Übergangsfunktion**
- 4  $q_0 \in Q$  der **Startzustand**
- 5  $F \subseteq Q$  eine endliche Menge von **akzeptierenden Zuständen**

$\delta$  muss für alle möglichen Argumente definiert sein; Vgl. auch Mealy-Moore Automaten

## Zustandstabelle

	$a_1 \in \Sigma$	$a_2 \in \Sigma$	$\dots$
$q_1 \in Q$	$\delta(q_1, a_1)$	$\delta(q_1, a_2)$	$\dots$
$q_2 \in Q$	$\delta(q_2, a_1)$		
$\vdots$	$\vdots$		

## Zustandsgraph

Sei  $A = (Q, \Sigma, \delta, q_0, F)$  ein DEA, der **Zustandsgraph** ist definiert, sodass

- 1 die Ecken die Zustände sind,
- 2 für Zustände  $p, q \in Q$  sind die Kanten von  $p$  nach  $q$  alle Tripel  $(p, a, q)$  mit  $a \in \Sigma$  und  $\delta(p, a) = q$

## Konvention

Zu jeder Kante  $(p, a, q)$  schreibt man die Eingabe  $a$ . Den Startzustand markiert man mit einem Pfeil; die akzeptierenden Zustände werden mit einem doppelten Kreis gekennzeichnet.

## Definition (erweiterte Übergangsfunktion)

Sei  $\delta$  eine Übergangsfunktion, wir definieren die **erweiterte Übergangsfunktion**  $\hat{\delta}: Q \times \Sigma^* \rightarrow Q$  induktiv:

$$\begin{aligned} \hat{\delta}(q, \epsilon) &:= q \\ \hat{\delta}(q, xa) &:= \delta(\hat{\delta}(q, x), a) \quad x \in \Sigma^*, a \in \Sigma \end{aligned}$$

## Definition

Sei  $A = (Q, \Sigma, \delta, q_0, F)$  ein DEA; die **Sprache**  $L(A)$  von  $A$ :

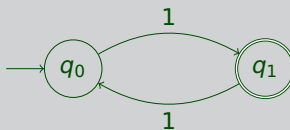
$$L(A) := \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in F\}$$

## Satz

Sei  $A$  ein DEA, dann ist  $L(A)$  regulär und umgekehrt existiert zu jeder regulären Sprache  $L$  ein DEA  $A$ , sodass  $L = L(A)$

## Beispiel

Betrachte den Automaten  $B$ :



## Satz

Die von Automaten  $B$  akzeptierte Sprache ist gegeben durch

$$L(B) = \{1^n \mid n \text{ ungerade}\}$$

## Beweis.

Beweis mittels verschränkter Induktion der folgenden beiden Aussagen:  $S_1(n)$ : Der Automaten ist nach  $n$ -maligem Drücken im Zustand  $q_0$ , gdw.  $n$  gerade. bzw.  $S_2(n)$ : Der Automaten ist nach  $n$ -maligem Drücken im Zustand  $q_1$ , gdw.  $n$  ungerade. ■



## Beispiel

Induktive Definition von Palindromen über  $\Sigma := \{0, 1\}$ :

- 1  $\epsilon, 0, 1$  sind Palindrome
- 2 Wenn  $x$  ein Palindrom ist, dann sind auch

$0x0 \quad 1x1$

Palindrome

Wir betrachten die folgenden Regeln  $R$ :

$P \rightarrow \epsilon \mid 0 \mid 1$

$P \rightarrow 0P0 \mid 1P1$

Die KFG  $G_1 = (\{P\}, \Sigma, R, P)$  beschreibt die Sprache der Palindrome:

$L(G_1)$  ist genau die Menge der Palindrome über dem Alphabet  $\Sigma$

## Satz (für KFG)

Gelte  $A \Rightarrow X_1 X_2 \dots X_n \xrightarrow{*} x$ , wobei  $X_i \in V \cup \Sigma$ ,  $x \in \Sigma^*$ ; dann können wir  $x$  in die Stücke  $x_1, x_2, \dots, x_n$  brechen, sodass für alle  $i$ :  $X_i \xrightarrow{*} x_i$

## Hilfsüberlegung

- Wir betrachten:  $X_1 X_2 \dots X_n \xrightarrow{*} x$
- Sei  $i < j$ , dann sind in  $x$  alle aus  $X_i$  abgeleiteten Satzformen links von den aus  $X_j$  abgeleiteten zu finden

## Beweis (des Satzes).

- 1 Wenn  $X_i \in \Sigma$ , dann  $X_i = x_i$  und offensichtlich  $X_i \xrightarrow{*} x_i$
- 2 Wenn  $X_i \in V$ , dann erhalten wir  $X_i \xrightarrow{*} x_i$  indem
  - Expansionen von  $X_1, \dots, X_{i-1}$  eliminiert
  - Expansionen von  $X_{i+1}, \dots, X_n$  eliminiert
  - überflüssige Schritte weggelassen werden

## Lemma

Sei  $G = (V, \Sigma, R, S)$  eine Grammatik und

- 1 sei  $A \xrightarrow{*} x$  eine Ableitung in  $G$
- 2 seien  $u, v \in (V \cup \Sigma)^*$

Dann ist auch  $uAv \xrightarrow{*} uxv$  eine Ableitung in  $G$

## Beweis.

Angenommen es existieren Wörter  $w_1, \dots, w_{k-1} \in (V \cup \Sigma)^*$ , sodass

$A \Rightarrow w_1 \Rightarrow \dots \Rightarrow w_{k-1} \Rightarrow x$

Wir argumentieren informell:

- Wir betrachten den Schritt  $w_i \Rightarrow w_{i+1}$  für  $i \in \{1, \dots, k-2\}$
- Nach Definition gilt auch  $uw_i v \Rightarrow uw_{i+1} v$
- Die anderen Fälle werden gleich behandelt

## Korrektheit der Grammatik.

Es ist zu zeigen, dass  $x \in L(G)$  gdw.  $x$  ein Palindrom ist

Wir zeigen nur:  $x \in L(G)$  impliziert  $x$  ist ein Palindrom. Dazu verwenden wir Induktion nach der Länge  $\ell$  der Ableitung  $P \xrightarrow{*} x$

- 1 **Basis**  $\ell = 1$ : Also gilt einer der folgenden 3 Fälle:

- $x = \epsilon$
- $x = 0$
- $x = 1$

In allen Fällen:  $x$  ist Palindrom

- 2 **Schritt**  $\ell > 1$ : Also hat die Ableitung eine der folgenden Gestalten:

- $P \Rightarrow 0P0 \xrightarrow{*} 0y0 = x$
- $P \Rightarrow 1P1 \xrightarrow{*} 1y1 = x$

wobei  $y \in \Sigma^*$

Somit gilt  $P \xrightarrow{*} y$ , woraus mit Induktionshypothese folgt,  $y$  ist ein Palindrom; damit ist aber auch  $x$  ein Palindrom