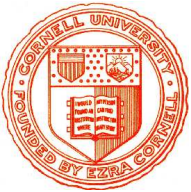


# Theoretische Informatik I



## Einheit 2.6

### Eigenschaften regulärer Sprachen



1. Abschlusseigenschaften
2. Prüfen von Eigenschaften
3. Wann sind Sprachen nicht regulär?

## ABSCHLUSSEIGENSCHAFTEN, WOZU?

**Zeige, daß bestimmte Operationen auf regulären Sprachen wieder zu regulären Sprachen führen**

**Zeige, daß bestimmte Operationen auf regulären Sprachen wieder zu regulären Sprachen führen**

- **Wiederverwendung von “Sprachmodulen”**
  - Schematische Komposition von
    - Grammatiken zur Erzeugung von Sprachen
    - Automaten zur Erkennung von Sprachen
    - Regulären Ausdrücken

## Zeige, daß bestimmte Operationen auf regulären Sprachen wieder zu regulären Sprachen führen

- **Wiederverwendung von “Sprachmodulen”**
  - Schematische Komposition von
    - Grammatiken zur Erzeugung von Sprachen
    - Automaten zur Erkennung von Sprachen
    - Regulären Ausdrücken
- **Schematische Konstruktion ist effektiver**
  - Fehlerfreier Aufbau sehr komplexer Grammatiken / Automaten
  - + Schematische Optimierung / Minimierung
  - Konstruktion “von Hand” oft fehleranfällig

## Zeige, daß bestimmte Operationen auf regulären Sprachen wieder zu regulären Sprachen führen

- **Wiederverwendung von “Sprachmodulen”**
  - Schematische Komposition von
    - Grammatiken zur Erzeugung von Sprachen
    - Automaten zur Erkennung von Sprachen
    - Regulären Ausdrücken
- **Schematische Konstruktion ist effektiver**
  - Fehlerfreier Aufbau sehr komplexer Grammatiken / Automaten
  - + Schematische Optimierung / Minimierung
  - Konstruktion “von Hand” oft fehleranfällig
- **Beispiel: Literale einer Programmiersprache**
  - Bilde Automaten für Tokenklassen: Zahlen, Bezeichner, Schlüsselworte, ...
  - Konstruktion liefert Automaten für alle Arten von Literalen

# ABSCHLUSSEIGENSCHAFTEN, PRÄZISIERT

Zeige:  $L_1, L_2$  regulär  $\Rightarrow L_1 \text{ op } L_2$  regulär

**Zeige:  $L_1, L_2$  regulär  $\Rightarrow L_1 \text{ op } L_2$  regulär**

## ● Es gilt Abgeschlossenheit unter 9 Operationen

- Die **Vereinigung** zweier regulärer Sprachen ist regulär  $L_1 \cup L_2$
- Das **Komplement** einer regulären Sprache ist regulär  $\overline{L}$
- Der **Durchschnitt** zweier regulärer Sprachen ist regulär  $L_1 \cap L_2$
- Die **Differenz** zweier regulärer Sprachen ist regulär  $L_1 - L_2$
- Die **Spiegelung** einer regulären Sprache ist regulär  $L^R$
- Die **Hülle** einer regulären Sprache ist regulär  $L^*$
- Die **Verkettung** zweier regulärer Sprachen ist regulär  $L_1 \circ L_2$
- Jeder **Homomorphismus** einer regulären Sprache ist regulär  $h(L)$
- Jeder **inverse Homomorphismus** einer regulären Sprache ist regulär  $h^{-1}(L)$

**Zeige:  $L_1, L_2$  regulär  $\Rightarrow L_1 \text{ op } L_2$  regulär**

## ● Es gilt Abgeschlossenheit unter 9 Operationen

- Die **Vereinigung** zweier regulärer Sprachen ist regulär  $L_1 \cup L_2$
- Das **Komplement** einer regulären Sprache ist regulär  $\overline{L}$
- Der **Durchschnitt** zweier regulärer Sprachen ist regulär  $L_1 \cap L_2$
- Die **Differenz** zweier regulärer Sprachen ist regulär  $L_1 - L_2$
- Die **Spiegelung** einer regulären Sprache ist regulär  $L^R$
- Die **Hülle** einer regulären Sprache ist regulär  $L^*$
- Die **Verkettung** zweier regulärer Sprachen ist regulär  $L_1 \circ L_2$
- Jeder **Homomorphismus** einer regulären Sprache ist regulär  $h(L)$
- Jeder **inverse Homomorphismus** einer regulären Sprache ist regulär  $h^{-1}(L)$

## ● Nachweis durch Verwendung aller Modelle

- DEA, NEA,  $\epsilon$ -NEA, reguläre Ausdrücke, Typ-3 Grammatiken
- Modelle sind ineinander umwandelbar – **wähle das passendste**



## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1), L_2 = L(E_2)$

## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1), L_2 = L(E_2)$

$\Rightarrow L_1 \cup L_2 = L(E_1) \cup L(E_2) = L(E_1 + E_2)$  regulär

## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1), L_2 = L(E_2)$

$\Rightarrow L_1 \cup L_2 = L(E_1) \cup L(E_2) = L(E_1 + E_2)$  regulär

- $L_1, L_2$  regulär  $\Rightarrow L_1 \circ L_2$  regulär

## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1), L_2 = L(E_2)$

$\Rightarrow L_1 \cup L_2 = L(E_1) \cup L(E_2) = L(E_1 + E_2)$  regulär

- $L_1, L_2$  regulär  $\Rightarrow L_1 \circ L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1), L_2 = L(E_2)$

## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1)$ ,  $L_2 = L(E_2)$

$\Rightarrow L_1 \cup L_2 = L(E_1) \cup L(E_2) = L(E_1 + E_2)$  regulär

- $L_1, L_2$  regulär  $\Rightarrow L_1 \circ L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1)$ ,  $L_2 = L(E_2)$

$\Rightarrow L_1 \circ L_2 = L(E_1) \circ L(E_2) = L(E_1 \circ E_2)$  regulär

## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1)$ ,  $L_2 = L(E_2)$

$\Rightarrow L_1 \cup L_2 = L(E_1) \cup L(E_2) = L(E_1 + E_2)$  regulär

- $L_1, L_2$  regulär  $\Rightarrow L_1 \circ L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1)$ ,  $L_2 = L(E_2)$

$\Rightarrow L_1 \circ L_2 = L(E_1) \circ L(E_2) = L(E_1 \circ E_2)$  regulär

- $L$  regulär  $\Rightarrow L^*$  regulär

## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1)$ ,  $L_2 = L(E_2)$

$\Rightarrow L_1 \cup L_2 = L(E_1) \cup L(E_2) = L(E_1 + E_2)$  regulär

- $L_1, L_2$  regulär  $\Rightarrow L_1 \circ L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1)$ ,  $L_2 = L(E_2)$

$\Rightarrow L_1 \circ L_2 = L(E_1) \circ L(E_2) = L(E_1 \circ E_2)$  regulär

- $L$  regulär  $\Rightarrow L^*$  regulär

$L$  regulär

$\Rightarrow$  Es gibt einen regulären Ausdruck  $E$  mit  $L = L(E)$



## Beweisführung mit regulären Ausdrücken

- $L_1, L_2$  regulär  $\Rightarrow L_1 \cup L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1)$ ,  $L_2 = L(E_2)$

$\Rightarrow L_1 \cup L_2 = L(E_1) \cup L(E_2) = L(E_1 + E_2)$  regulär

- $L_1, L_2$  regulär  $\Rightarrow L_1 \circ L_2$  regulär

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt reguläre Ausdrücke  $E_1, E_2$  mit  $L_1 = L(E_1)$ ,  $L_2 = L(E_2)$

$\Rightarrow L_1 \circ L_2 = L(E_1) \circ L(E_2) = L(E_1 \circ E_2)$  regulär

- $L$  regulär  $\Rightarrow L^*$  regulär

$L$  regulär

$\Rightarrow$  Es gibt einen regulären Ausdruck  $E$  mit  $L = L(E)$

$\Rightarrow L^* = (L(E))^* = L(E^*)$  regulär

## Beweisführung mit endlichen Automaten

- $L$  regulär  $\Rightarrow \overline{L}$  regulär

Komplementiere akzeptierende Zustände des erkennenden Automaten

## Beweisführung mit endlichen Automaten

- $L$  regulär  $\Rightarrow \overline{L}$  regulär

Komplementiere akzeptierende Zustände des erkennenden Automaten

$L$  regulär

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$

## Beweisführung mit endlichen Automaten

- **$L$  regulär  $\Rightarrow \overline{L}$  regulär**

Komplementiere akzeptierende Zustände des erkennenden Automaten

**$L$  regulär**

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$

$\Rightarrow \overline{L} = \overline{L(A)} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \notin F\} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q - F\}$

## Beweisführung mit endlichen Automaten

- **$L$  regulär  $\Rightarrow \overline{L}$  regulär**

Komplementiere akzeptierende Zustände des erkennenden Automaten

**$L$  regulär**

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$

$\Rightarrow \overline{L} = \overline{L(A)} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \notin F\} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q - F\}$   
 $= L(Q, \Sigma, \delta, q_0, Q - F)$  **regulär**

## Beweisführung mit endlichen Automaten

- **$L$  regulär  $\Rightarrow \overline{L}$  regulär**

Komplementiere akzeptierende Zustände des erkennenden Automaten

**$L$  regulär**

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$

$\Rightarrow \overline{L} = \overline{L(A)} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \notin F\} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q - F\}$   
 $= L(Q, \Sigma, \delta, q_0, Q - F)$  **regulär**

- **Beispiel: Komplementierung von  $(0+1)^*01$**

## Beweisführung mit endlichen Automaten

- **$L$  regulär  $\Rightarrow \overline{L}$  regulär**

Komplementiere akzeptierende Zustände des erkennenden Automaten

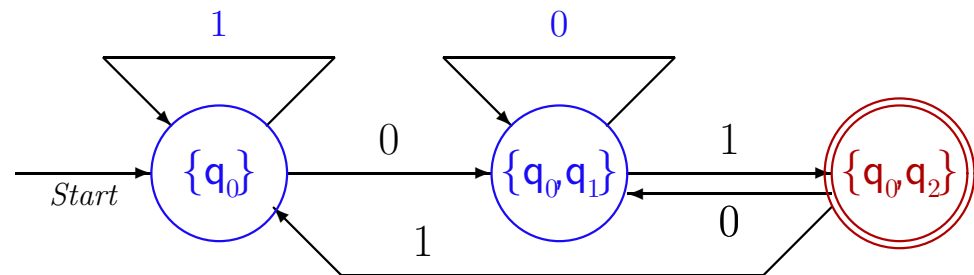
**$L$  regulär**

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$

$\Rightarrow \overline{L} = \overline{L(A)} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \notin F\} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q - F\}$   
 $= L(Q, \Sigma, \delta, q_0, Q - F)$  **regulär**

- **Beispiel: Komplementierung von  $(0+1)^*01$**

– Zugehöriger DEA



## Beweisführung mit endlichen Automaten

- **$L$  regulär  $\Rightarrow \overline{L}$  regulär**

Komplementiere akzeptierende Zustände des erkennenden Automaten

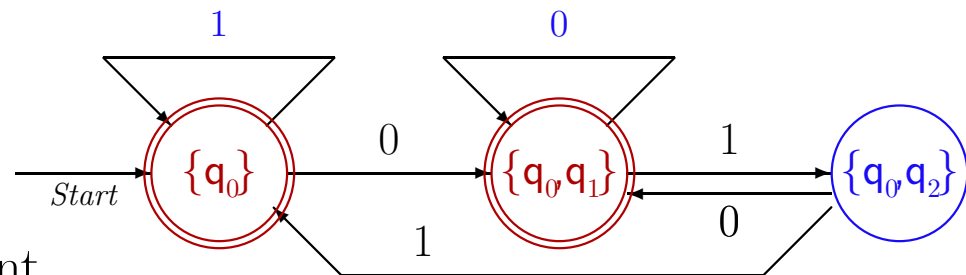
**$L$  regulär**

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$

$\Rightarrow \overline{L} = \overline{L(A)} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \notin F\} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q - F\}$   
 $= L(Q, \Sigma, \delta, q_0, Q - F)$  **regulär**

- **Beispiel: Komplementierung von  $(0+1)^*01$**

– Zugehöriger DEA



– Komplementautomat erkennt  
Worte die **nicht** mit 01 enden



## Beweisführung mit endlichen Automaten

- **$L$  regulär  $\Rightarrow \overline{L}$  regulär**

Komplementiere akzeptierende Zustände des erkennenden Automaten

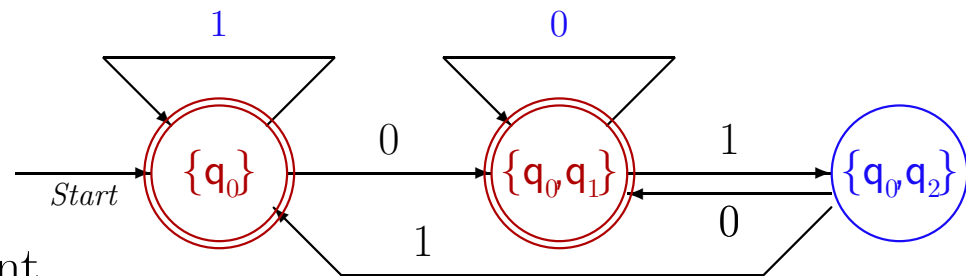
**$L$  regulär**

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$

$\Rightarrow \overline{L} = \overline{L(A)} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \notin F\} = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q - F\}$   
 $= L(Q, \Sigma, \delta, q_0, Q - F)$  **regulär**

- **Beispiel: Komplementierung von  $(0+1)^*01$**

– Zugehöriger DEA



– Komplementautomat erkennt

Worte die **nicht** mit 01 enden

– Regulärer Ausdruck durch Zustandseliminationsverfahren erzeugbar

- **Einfache mathematische Beweise**

- Einfache mathematische Beweise

$L_1, L_2$  regulär

- Einfache mathematische Beweise

$L_1, L_2$  regulär  $\Rightarrow L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$  regulär

- Einfache mathematische Beweise

$L_1, L_2$  regulär  $\Rightarrow L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$  regulär

$L_1, L_2$  regulär

- Einfache mathematische Beweise

$L_1, L_2$  regulär  $\Rightarrow L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$  regulär

$L_1, L_2$  regulär  $\Rightarrow L_1 - L_2 = L_1 \cap \overline{L_2}$  regulär

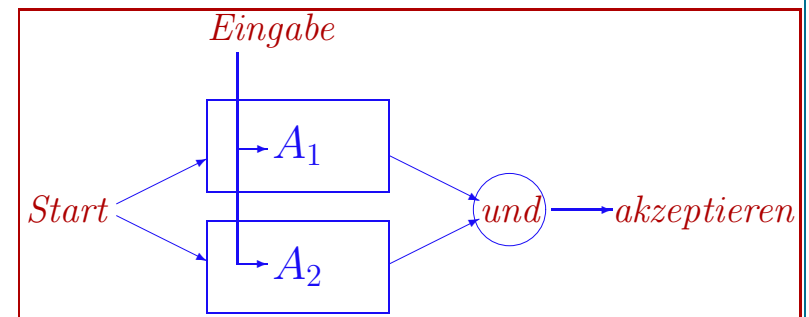
- Einfache mathematische Beweise

$L_1, L_2$  regulär  $\Rightarrow L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$  regulär

$L_1, L_2$  regulär  $\Rightarrow L_1 - L_2 = L_1 \cap \overline{L_2}$  regulär

- **Produktkonstruktion** auf endlichen Automaten

Simultane Abarbeitung von Worten in beiden Automaten



- Einfache mathematische Beweise

$L_1, L_2$  regulär  $\Rightarrow L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$  regulär

$L_1, L_2$  regulär  $\Rightarrow L_1 - L_2 = L_1 \cap \overline{L_2}$  regulär

- **Produktkonstruktion** auf endlichen Automaten

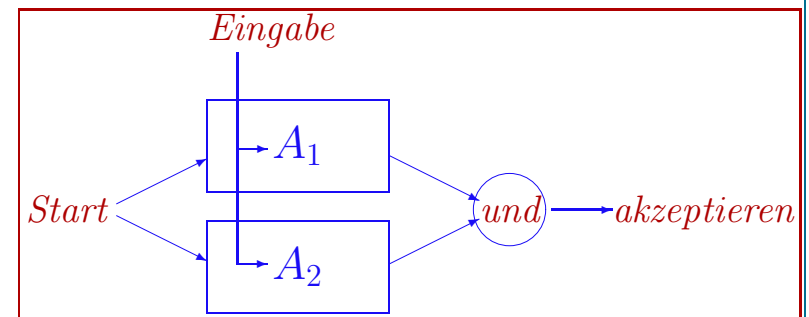
Simultane Abarbeitung von Worten in beiden Automaten

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt DEAs  $A_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1)$

und  $A_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$

mit  $L_1 = L(A_1)$ ,  $L_2 = L(A_2)$





- Einfache mathematische Beweise

$L_1, L_2$  regulär  $\Rightarrow L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$  regulär

$L_1, L_2$  regulär  $\Rightarrow L_1 - L_2 = L_1 \cap \overline{L_2}$  regulär

- **Produktkonstruktion** auf endlichen Automaten

Simultane Abarbeitung von Worten in beiden Automaten

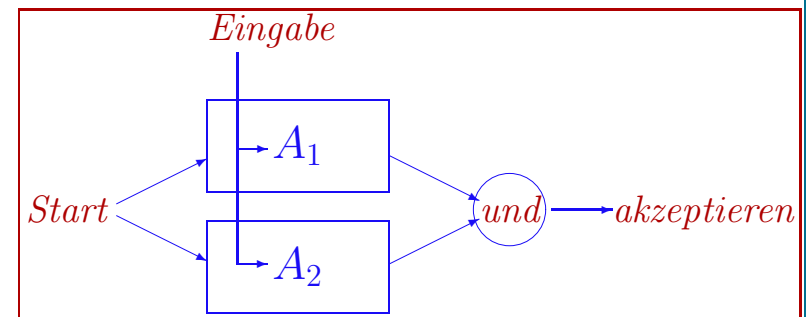
$L_1, L_2$  regulär

$\Rightarrow$  Es gibt DEAs  $A_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1)$

und  $A_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$

mit  $L_1 = L(A_1), L_2 = L(A_2)$

$\Rightarrow L_1 \cap L_2 = \{w \in \Sigma^* \mid \hat{\delta}_1(q_{0,1}, w) \in F_1 \wedge \hat{\delta}_2(q_{0,2}, w) \in F_2\}$   
 $= \{w \in \Sigma^* \mid (\hat{\delta}_1(q_{0,1}, w), \hat{\delta}_2(q_{0,2}, w)) \in F_1 \times F_2\}$



## ● Einfache mathematische Beweise

$L_1, L_2$  regulär  $\Rightarrow L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$  regulär

$L_1, L_2$  regulär  $\Rightarrow L_1 - L_2 = L_1 \cap \overline{L_2}$  regulär

## ● Produktkonstruktion auf endlichen Automaten

Simultane Abarbeitung von Worten in beiden Automaten

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt DEAs  $A_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1)$

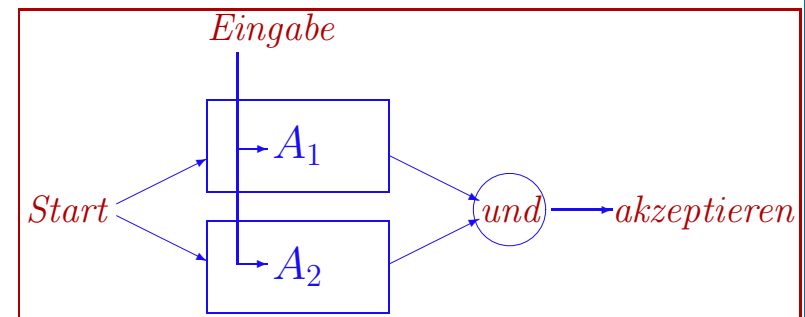
und  $A_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$

mit  $L_1 = L(A_1)$ ,  $L_2 = L(A_2)$

$\Rightarrow L_1 \cap L_2 = \{w \in \Sigma^* \mid \hat{\delta}_1(q_{0,1}, w) \in F_1 \wedge \hat{\delta}_2(q_{0,2}, w) \in F_2\}$   
 $= \{w \in \Sigma^* \mid (\hat{\delta}_1(q_{0,1}, w), \hat{\delta}_2(q_{0,2}, w)) \in F_1 \times F_2\}$

Konstruiere  $A = (Q_1 \times Q_2, \Sigma, \delta, (q_{0,1}, q_{0,2}), F_1 \times F_2)$

mit  $\delta((p, q), a) = (\delta_1(p, a), \delta_2(q, a))$  für  $p \in Q_1, q \in Q_2, a \in \Sigma$



## ● Einfache mathematische Beweise

$L_1, L_2$  regulär  $\Rightarrow L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$  regulär

$L_1, L_2$  regulär  $\Rightarrow L_1 - L_2 = L_1 \cap \overline{L_2}$  regulär

## ● Produktkonstruktion auf endlichen Automaten

Simultane Abarbeitung von Worten in beiden Automaten

$L_1, L_2$  regulär

$\Rightarrow$  Es gibt DEAs  $A_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1)$

und  $A_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$

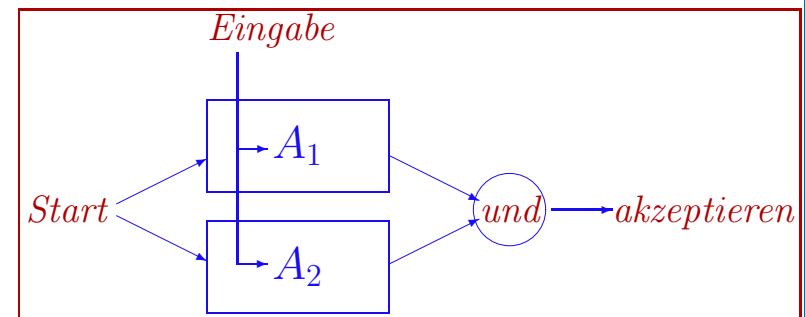
mit  $L_1 = L(A_1)$ ,  $L_2 = L(A_2)$

$\Rightarrow L_1 \cap L_2 = \{w \in \Sigma^* \mid \hat{\delta}_1(q_{0,1}, w) \in F_1 \wedge \hat{\delta}_2(q_{0,2}, w) \in F_2\}$   
 $= \{w \in \Sigma^* \mid (\hat{\delta}_1(q_{0,1}, w), \hat{\delta}_2(q_{0,2}, w)) \in F_1 \times F_2\}$

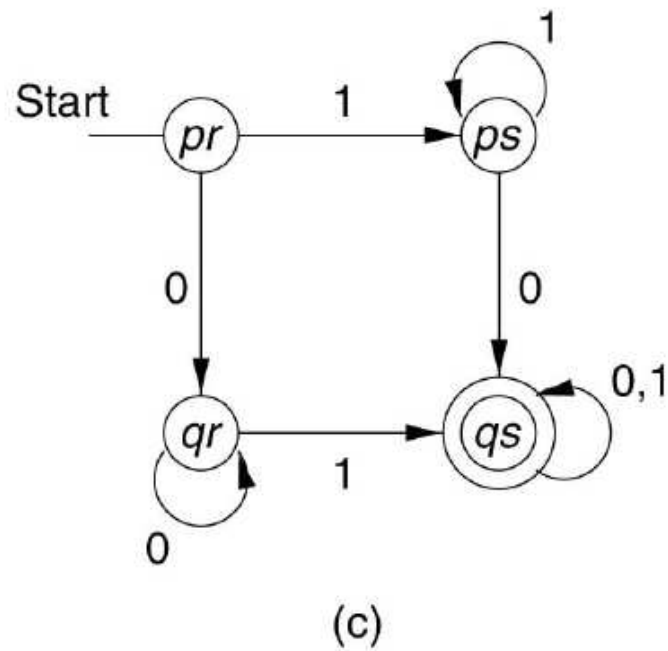
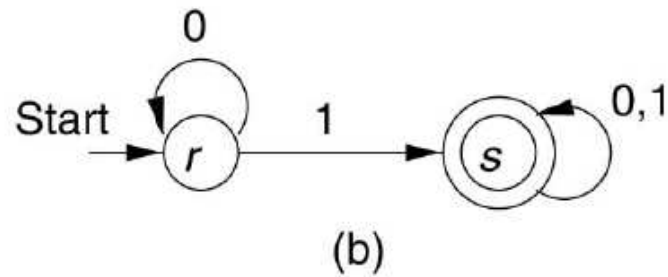
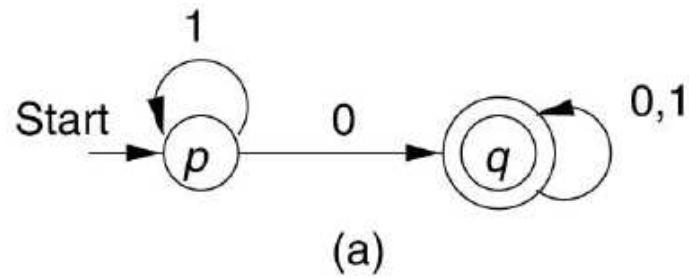
Konstruiere  $A = (Q_1 \times Q_2, \Sigma, \delta, (q_{0,1}, q_{0,2}), F_1 \times F_2)$

mit  $\delta((p, q), a) = (\delta_1(p, a), \delta_2(q, a))$  für  $p \in Q_1, q \in Q_2, a \in \Sigma$

$\Rightarrow L_1 \cap L_2 = L(A)$  regulär



# PRODUKTKONSTRUKTION AM BEISPIEL



## ABSCHLUSS UNTER SPIEGELUNG

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ABSCHLUSS UNTER SPIEGELUNG

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

- Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L=L(A)$

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ● Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$ 
  - Umkehrung der Pfeile im Diagramm:  $\delta^R(q, a) = q'$  g.d.w.  $\delta(q', a) = q$
  - $q_0$  wird zum akzeptierenden Zustand:  $F^R = \{q_0\}$
  - Neuer Startzustand  $q_0^R$  mit  $\epsilon$ -Übergängen zu allen  $q \in F$

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ● Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$ 
  - Umkehrung der Pfeile im Diagramm:  $\delta^R(q, a) = q'$  g.d.w.  $\delta(q', a) = q$
  - $q_0$  wird zum akzeptierenden Zustand:  $F^R = \{q_0\}$
  - Neuer Startzustand  $q_0^R$  mit  $\epsilon$ -Übergängen zu allen  $q \in F$

## ● Induktiver Beweis mit regulären Ausdrücken

Sei  $L = L(E)$  für einen regulären Ausdruck



$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ● Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$ 
  - Umkehrung der Pfeile im Diagramm:  $\delta^R(q, a) = q'$  g.d.w.  $\delta(q', a) = q$
  - $q_0$  wird zum akzeptierenden Zustand:  $F^R = \{q_0\}$
  - Neuer Startzustand  $q_0^R$  mit  $\epsilon$ -Übergängen zu allen  $q \in F$

## ● Induktiver Beweis mit regulären Ausdrücken

Sei  $L = L(E)$  für einen regulären Ausdruck

- Für  $E \in \{\emptyset, \epsilon, a\}$  ist  $L^R = L = L(E)$  regulär

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ● Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$ 
  - Umkehrung der Pfeile im Diagramm:  $\delta^R(q, a) = q'$  g.d.w.  $\delta(q', a) = q$
  - $q_0$  wird zum akzeptierenden Zustand:  $F^R = \{q_0\}$
  - Neuer Startzustand  $q_0^R$  mit  $\epsilon$ -Übergängen zu allen  $q \in F$

## ● Induktiver Beweis mit regulären Ausdrücken

Sei  $L = L(E)$  für einen regulären Ausdruck

- Für  $E \in \{\emptyset, \epsilon, a\}$  ist  $L^R = L = L(E)$  regulär
- Für  $E = E_1 + E_2$  ist  $L^R = (L(E_1) \cup L(E_2))^R = L(E_1)^R \cup L(E_2)^R$  regulär

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ● Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$ 
  - Umkehrung der Pfeile im Diagramm:  $\delta^R(q, a) = q'$  g.d.w.  $\delta(q', a) = q$
  - $q_0$  wird zum akzeptierenden Zustand:  $F^R = \{q_0\}$
  - Neuer Startzustand  $q_0^R$  mit  $\epsilon$ -Übergängen zu allen  $q \in F$

## ● Induktiver Beweis mit regulären Ausdrücken

Sei  $L = L(E)$  für einen regulären Ausdruck

- Für  $E \in \{\emptyset, \epsilon, a\}$  ist  $L^R = L = L(E)$  regulär
- Für  $E = E_1 + E_2$  ist  $L^R = (L(E_1) \cup L(E_2))^R = L(E_1)^R \cup L(E_2)^R$  regulär
- Für  $E = E_1 \circ E_2$  ist  $L^R = (L(E_1) \circ L(E_2))^R = L(E_2)^R \circ L(E_1)^R$  regulär

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ● Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$ 
  - Umkehrung der Pfeile im Diagramm:  $\delta^R(q, a) = q'$  g.d.w.  $\delta(q', a) = q$
  - $q_0$  wird zum akzeptierenden Zustand:  $F^R = \{q_0\}$
  - Neuer Startzustand  $q_0^R$  mit  $\epsilon$ -Übergängen zu allen  $q \in F$

## ● Induktiver Beweis mit regulären Ausdrücken

Sei  $L = L(E)$  für einen regulären Ausdruck

- Für  $E \in \{\emptyset, \epsilon, a\}$  ist  $L^R = L = L(E)$  regulär
- Für  $E = E_1 + E_2$  ist  $L^R = (L(E_1) \cup L(E_2))^R = L(E_1)^R \cup L(E_2)^R$  regulär
- Für  $E = E_1 \circ E_2$  ist  $L^R = (L(E_1) \circ L(E_2))^R = L(E_2)^R \circ L(E_1)^R$  regulär
- Für  $E = E_1^*$  ist  $L^R = L(E_1^*)^R = (L(E_1)^R)^*$  regulär

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ● Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$ 
  - Umkehrung der Pfeile im Diagramm:  $\delta^R(q, a) = q'$  g.d.w.  $\delta(q', a) = q$
  - $q_0$  wird zum akzeptierenden Zustand:  $F^R = \{q_0\}$
  - Neuer Startzustand  $q_0^R$  mit  $\epsilon$ -Übergängen zu allen  $q \in F$

## ● Induktiver Beweis mit regulären Ausdrücken

Sei  $L = L(E)$  für einen regulären Ausdruck

- Für  $E \in \{\emptyset, \epsilon, a\}$  ist  $L^R = L = L(E)$  regulär
- Für  $E = E_1 + E_2$  ist  $L^R = (L(E_1) \cup L(E_2))^R = L(E_1)^R \cup L(E_2)^R$  regulär
- Für  $E = E_1 \circ E_2$  ist  $L^R = (L(E_1) \circ L(E_2))^R = L(E_2)^R \circ L(E_1)^R$  regulär
- Für  $E = E_1^*$  ist  $L^R = L(E_1^*)^R = (L(E_1)^R)^*$  regulär

## ● Beispiel: Spiegelung von $L((0+1)0^*)$

$L$  regulär  $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$  regulär

## ● Beweisführung mit Automaten

- Bilde Umkehrautomaten zu  $A = (Q, \Sigma, \delta, q_0, F)$  mit  $L = L(A)$ 
  - Umkehrung der Pfeile im Diagramm:  $\delta^R(q, a) = q'$  g.d.w.  $\delta(q', a) = q$
  - $q_0$  wird zum akzeptierenden Zustand:  $F^R = \{q_0\}$
  - Neuer Startzustand  $q_0^R$  mit  $\epsilon$ -Übergängen zu allen  $q \in F$

## ● Induktiver Beweis mit regulären Ausdrücken

Sei  $L = L(E)$  für einen regulären Ausdruck

- Für  $E \in \{\emptyset, \epsilon, a\}$  ist  $L^R = L = L(E)$  regulär
- Für  $E = E_1 + E_2$  ist  $L^R = (L(E_1) \cup L(E_2))^R = L(E_1)^R \cup L(E_2)^R$  regulär
- Für  $E = E_1 \circ E_2$  ist  $L^R = (L(E_1) \circ L(E_2))^R = L(E_2)^R \circ L(E_1)^R$  regulär
- Für  $E = E_1^*$  ist  $L^R = L(E_1^*)^R = (L(E_1)^R)^*$  regulär

## ● Beispiel: Spiegelung von $L((0+1)0^*)$

- $L^R = L((0^*)^R(0+1)^R) = L((0^R)^*(0^R+1^R)) = L(0^*(0+1))$

## ABSCHLUSS UNTER HOMOMORPHISMEN

*$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h(L)$  regulär*

# ABSCHLUSS UNTER HOMOMORPHISMEN

**$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h(L)$  regulär**

$h: \Sigma \rightarrow \Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– *Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar*



# ABSCHLUSS UNTER HOMOMORPHISMEN

**$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h(L)$  regulär**

$h:\Sigma\rightarrow\Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– *Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar*

**$h(L)$**  =  $\{h(w) \mid w \in L\} \subseteq \Sigma'^*$  ist das Abbild der Worte von  $L$  unter  $h$

**$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h(L)$  regulär**

$h:\Sigma\rightarrow\Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– *Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar*

**$h(L)$**  =  $\{h(w) \mid w \in L\} \subseteq \Sigma'^*$  ist das Abbild der Worte von  $L$  unter  $h$

## ● Beweis mit Grammatiken

**$L$  regulär**

$\Rightarrow$  Es gibt eine Typ-3 Grammatik  $G = (V, \Sigma, P, S)$  mit  $L = L(G)$

## $L$ regulär, $h$ Homomorphismus $\Rightarrow h(L)$ regulär

$h:\Sigma\rightarrow\Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– *Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar*

$h(L)=\{h(w) \mid w \in L\} \subseteq \Sigma'^*$  ist das Abbild der Worte von  $L$  unter  $h$

### ● Beweis mit Grammatiken

$L$  regulär

$\Rightarrow$  Es gibt eine Typ-3 Grammatik  $G = (V, \Sigma, P, S)$  mit  $L = L(G)$

$\Rightarrow h(L) = h(L(G)) = \{h(v_1)..h(v_n) \in \Sigma'^* \mid S \xrightarrow{*} v_1..v_n\}$

## $L$ regulär, $h$ Homomorphismus $\Rightarrow h(L)$ regulär

$h:\Sigma\rightarrow\Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– *Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar*

$h(L)=\{h(w) \mid w \in L\} \subseteq \Sigma'^*$  ist das Abbild der Worte von  $L$  unter  $h$

### ● Beweis mit Grammatiken

$L$  regulär

$\Rightarrow$  Es gibt eine Typ-3 Grammatik  $G = (V, \Sigma, P, S)$  mit  $L = L(G)$

$\Rightarrow h(L) = h(L(G)) = \{h(v_1)..h(v_n) \in \Sigma'^* \mid S \xrightarrow{*} v_1..v_n\}$

Für  $A \rightarrow v B \in P$  erzeuge Regeln  $A \rightarrow a_1 B_1, B_1 \rightarrow a_2 B_2, \dots, B_{k-1} \rightarrow a_k B$ ,  
wobei  $h(v) = a_1..a_k$  und alle  $B_i$  neue Hilfsvariablen

## $L$ regulär, $h$ Homomorphismus $\Rightarrow h(L)$ regulär

$h:\Sigma\rightarrow\Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– *Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar*

$h(L)=\{h(w) \mid w \in L\} \subseteq \Sigma'^*$  ist das Abbild der Worte von  $L$  unter  $h$

### ● Beweis mit Grammatiken

$L$  regulär

$\Rightarrow$  Es gibt eine Typ-3 Grammatik  $G = (V, \Sigma, P, S)$  mit  $L = L(G)$

$\Rightarrow h(L) = h(L(G)) = \{h(v_1)..h(v_n) \in \Sigma'^* \mid S \xrightarrow{*} v_1..v_n\}$

Für  $A \rightarrow v B \in P$  erzeuge Regeln  $A \rightarrow a_1 B_1, B_1 \rightarrow a_2 B_2, \dots, B_{k-1} \rightarrow a_k B$ ,  
wobei  $h(v) = a_1..a_k$  und alle  $B_i$  neue Hilfsvariablen

Sei  $P_h$  die Menge dieser Regeln und  $V_h$  die Menge ihrer Hilfsvariablen

## $L$ regulär, $h$ Homomorphismus $\Rightarrow h(L)$ regulär

$h:\Sigma\rightarrow\Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– *Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar*

$h(L)=\{h(w) \mid w \in L\} \subseteq \Sigma'^*$  ist das Abbild der Worte von  $L$  unter  $h$

### ● Beweis mit Grammatiken

$L$  regulär

$\Rightarrow$  Es gibt eine Typ-3 Grammatik  $G = (V, \Sigma, P, S)$  mit  $L = L(G)$

$\Rightarrow h(L) = h(L(G)) = \{h(v_1)..h(v_n) \in \Sigma'^* \mid S \xrightarrow{*} v_1..v_n\}$

Für  $A \rightarrow v B \in P$  erzeuge Regeln  $A \rightarrow a_1 B_1, B_1 \rightarrow a_2 B_2, \dots, B_{k-1} \rightarrow a_k B$ ,  
wobei  $h(v) = a_1..a_k$  und alle  $B_i$  neue Hilfsvariablen

Sei  $P_h$  die Menge dieser Regeln und  $V_h$  die Menge ihrer Hilfsvariablen

Für  $G_h = (V_h, \Sigma', P_h, S)$  gilt  $A \rightarrow v B \in P \Leftrightarrow A \xrightarrow{*}_{G_h} h(v) B$

und  $S \xrightarrow{*}_G v_1..v_n \Leftrightarrow S \xrightarrow{*}_{G_h} h(v_1)..h(v_n)$

## $L$ regulär, $h$ Homomorphismus $\Rightarrow h(L)$ regulär

$h:\Sigma\rightarrow\Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar

$h(L)=\{h(w) \mid w \in L\} \subseteq \Sigma'^*$  ist das Abbild der Worte von  $L$  unter  $h$

### ● Beweis mit Grammatiken

$L$  regulär

$\Rightarrow$  Es gibt eine Typ-3 Grammatik  $G = (V, \Sigma, P, S)$  mit  $L = L(G)$

$\Rightarrow h(L) = h(L(G)) = \{h(v_1)..h(v_n) \in \Sigma'^* \mid S \xrightarrow{*} v_1..v_n\}$

Für  $A \rightarrow v B \in P$  erzeuge Regeln  $A \rightarrow a_1 B_1, B_1 \rightarrow a_2 B_2, \dots, B_{k-1} \rightarrow a_k B$ ,  
wobei  $h(v) = a_1..a_k$  und alle  $B_i$  neue Hilfsvariablen

Sei  $P_h$  die Menge dieser Regeln und  $V_h$  die Menge ihrer Hilfsvariablen

Für  $G_h = (V_h, \Sigma', P_h, S)$  gilt  $A \rightarrow v B \in P \Leftrightarrow A \xrightarrow{*}_{G_h} h(v) B$

und  $S \xrightarrow{*}_G v_1..v_n \Leftrightarrow S \xrightarrow{*}_{G_h} h(v_1)..h(v_n)$

$\Rightarrow h(L) = \{h(v_1)..h(v_n) \in \Sigma'^* \mid S \xrightarrow{*}_{G_h} h(v_1)..h(v_n)\} = L(G_h)$  **regulär**

## $L$ regulär, $h$ Homomorphismus $\Rightarrow h(L)$ regulär

$h:\Sigma\rightarrow\Sigma'$  ist **Homomorphismus**, wenn  $h(v_1..v_n) = h(v_1)..h(v_n)$

– Homomorphismen sind mit endlichen (Ein-/Ausgabe) Automaten berechenbar

$h(L)=\{h(w) \mid w \in L\} \subseteq \Sigma'^*$  ist das Abbild der Worte von  $L$  unter  $h$

### ● Beweis mit Grammatiken

$L$  regulär

$\Rightarrow$  Es gibt eine Typ-3 Grammatik  $G = (V, \Sigma, P, S)$  mit  $L = L(G)$

$\Rightarrow h(L) = h(L(G)) = \{h(v_1)..h(v_n) \in \Sigma'^* \mid S \xrightarrow{*} v_1..v_n\}$

Für  $A \rightarrow v B \in P$  erzeuge Regeln  $A \rightarrow a_1 B_1, B_1 \rightarrow a_2 B_2, \dots B_{k-1} \rightarrow a_k B$ ,  
wobei  $h(v) = a_1..a_k$  und alle  $B_i$  neue Hilfsvariablen

Sei  $P_h$  die Menge dieser Regeln und  $V_h$  die Menge ihrer Hilfsvariablen

Für  $G_h = (V_h, \Sigma', P_h, S)$  gilt  $A \rightarrow v B \in P \Leftrightarrow A \xrightarrow{*}_{G_h} h(v) B$

und  $S \xrightarrow{*}_G v_1..v_n \Leftrightarrow S \xrightarrow{*}_{G_h} h(v_1)..h(v_n)$

$\Rightarrow h(L) = \{h(v_1)..h(v_n) \in \Sigma'^* \mid S \xrightarrow{*}_{G_h} h(v_1)..h(v_n)\} = L(G_h)$  **regulär**

Beweis mit regulären Ausdrücken in Hopcroft, Motwani, Ullman §4.2.3



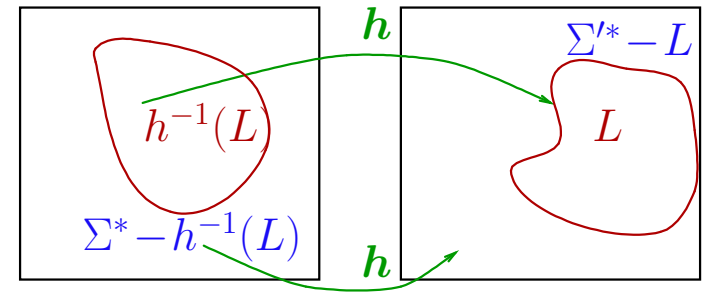
## ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär

# ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär

$h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$  ist das  
Urbild der Worte von  $L$  unter  $h$



# ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

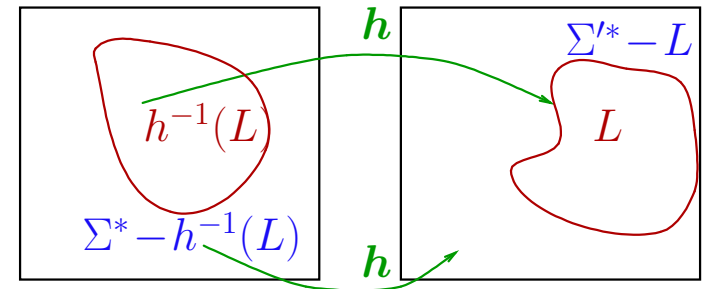
$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär

$h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$  ist das

Urbild der Worte von  $L$  unter  $h$

– z.B. Für  $L = L((01+10)^*)$ ,

$h(a) = 01, h(b) = 10$  ist  $h^{-1}(L) = L((a+b)^*)$



# ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

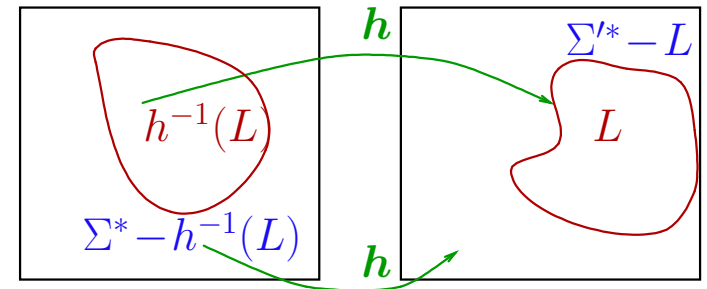
$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär

$h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$  ist das

Urbild der Worte von  $L$  unter  $h$

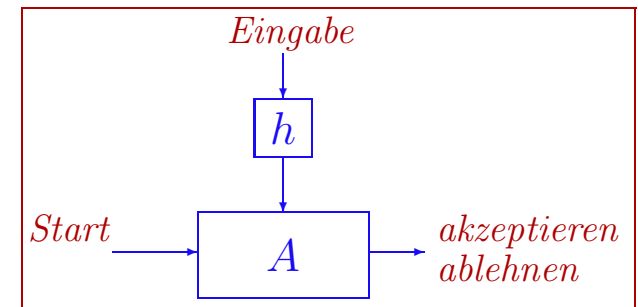
– z.B. Für  $L = L((01+10)^*)$ ,

$h(a) = 01, h(b) = 10$  ist  $h^{-1}(L) = L((a+b)^*)$



## ● Beweis mit endlichen Automaten

Berechnung von  $h$  vor Abarbeitung der Worte im Automaten



# ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

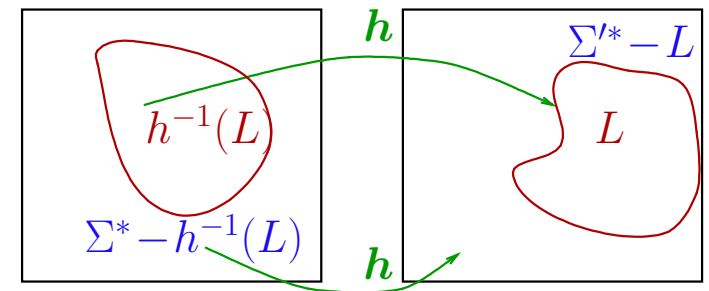
$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär

$h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$  ist das

Urbild der Worte von  $L$  unter  $h$

– z.B. Für  $L = L((01+10)^*)$ ,

$h(a) = 01, h(b) = 10$  ist  $h^{-1}(L) = L((a+b)^*)$



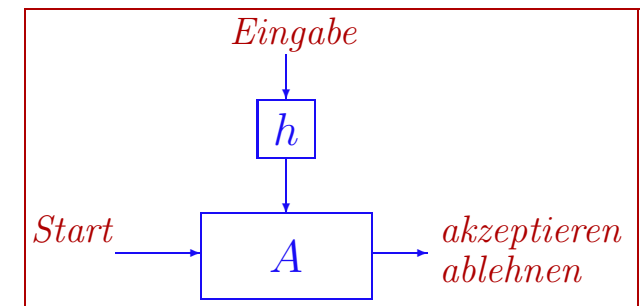
## ● Beweis mit endlichen Automaten

Berechnung von  $h$  vor Abarbeitung der Worte im Automaten

$L$  regulär

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma', \delta, q_0, F)$

mit  $L = L(A) = \{w \in \Sigma'^* \mid \hat{\delta}(q_0, w) \in F\}$



# ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

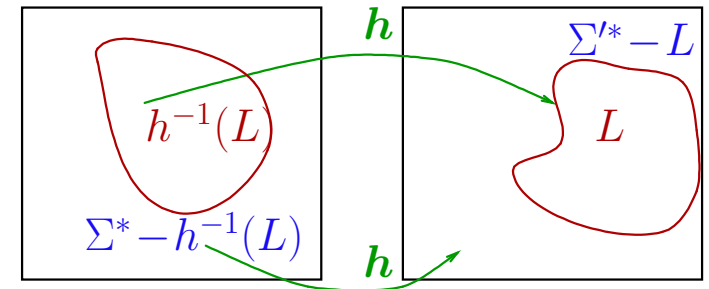
**$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär**

$h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$  ist das

Urbild der Worte von  $L$  unter  $h$

– z.B. Für  $L = L((01+10)^*)$ ,

$h(a) = 01, h(b) = 10$  ist  $h^{-1}(L) = L((a+b)^*)$



## ● Beweis mit endlichen Automaten

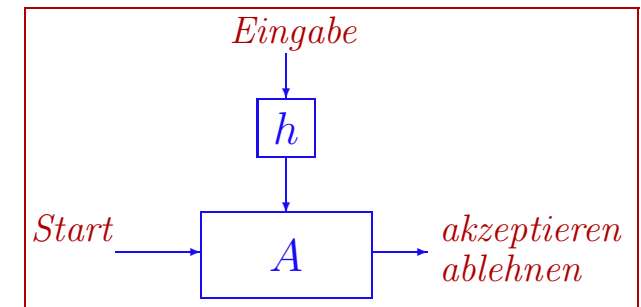
Berechnung von  $h$  vor Abarbeitung der Worte im Automaten

**$L$  regulär**

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma', \delta, q_0, F)$

mit  $L = L(A) = \{w \in \Sigma'^* \mid \hat{\delta}(q_0, w) \in F\}$

$\Rightarrow h^{-1}(L) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, h(w)) \in F\}$



# ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

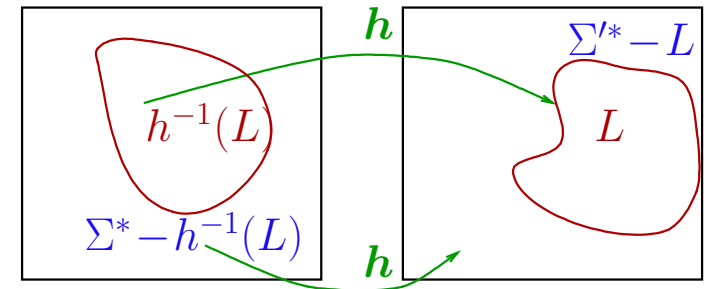
$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär

$h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$  ist das

Urbild der Worte von  $L$  unter  $h$

– z.B. Für  $L = L((01+10)^*)$ ,

$h(a) = 01, h(b) = 10$  ist  $h^{-1}(L) = L((a+b)^*)$



## ● Beweis mit endlichen Automaten

Berechnung von  $h$  vor Abarbeitung der Worte im Automaten

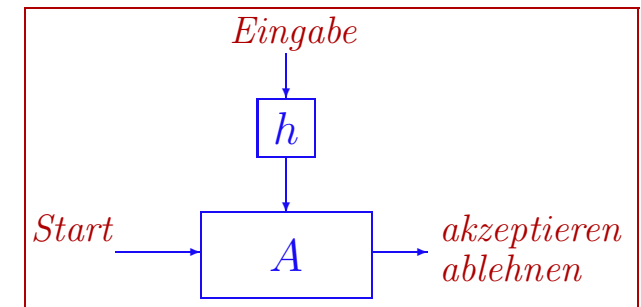
$L$  regulär

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma', \delta, q_0, F)$

mit  $L = L(A) = \{w \in \Sigma'^* \mid \hat{\delta}(q_0, w) \in F\}$

$\Rightarrow h^{-1}(L) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, h(w)) \in F\}$

Konstruiere  $A_h = (Q, \Sigma, \delta_h, q_0, F)$  mit  $\delta_h(q, a) = \hat{\delta}(q, h(a))$



# ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

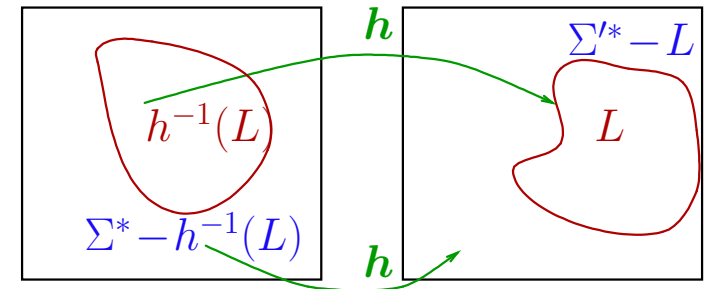
**$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär**

$h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$  ist das

Urbild der Worte von  $L$  unter  $h$

– z.B. Für  $L = L((01+10)^*)$ ,

$h(a) = 01, h(b) = 10$  ist  $h^{-1}(L) = L((a+b)^*)$



## ● Beweis mit endlichen Automaten

Berechnung von  $h$  vor Abarbeitung der Worte im Automaten

**$L$  regulär**

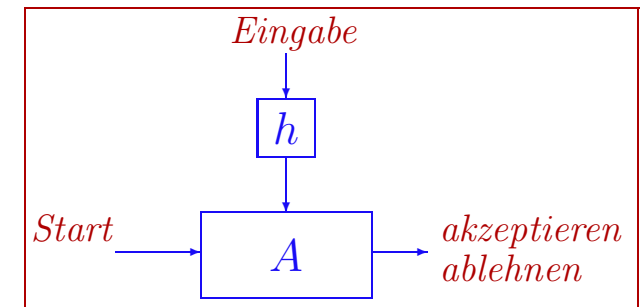
$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma', \delta, q_0, F)$

mit  $L = L(A) = \{w \in \Sigma'^* \mid \hat{\delta}(q_0, w) \in F\}$

$\Rightarrow h^{-1}(L) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, h(w)) \in F\}$

Konstruiere  $A_h = (Q, \Sigma, \delta_h, q_0, F)$  mit  $\delta_h(q, a) = \hat{\delta}(q, h(a))$

Dann gilt  $\hat{\delta}_h(q, w) = \hat{\delta}(q, h(w))$  für alle  $q \in Q$  und  $w \in \Sigma^*$





# ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

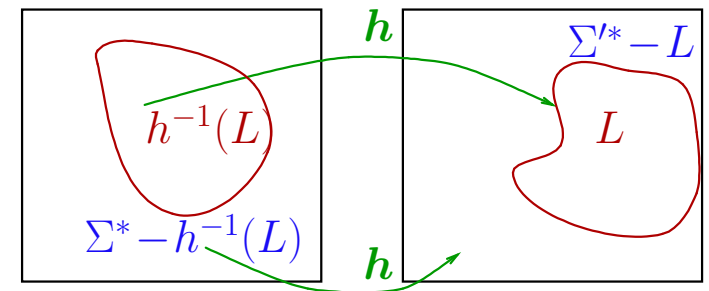
**$L$  regulär,  $h$  Homomorphismus  $\Rightarrow h^{-1}(L)$  regulär**

$h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$  ist das

Urbild der Worte von  $L$  unter  $h$

– z.B. Für  $L = L((01+10)^*)$ ,

$h(a) = 01, h(b) = 10$  ist  $h^{-1}(L) = L((a+b)^*)$



## ● Beweis mit endlichen Automaten

Berechnung von  $h$  vor Abarbeitung der Worte im Automaten

**$L$  regulär**

$\Rightarrow$  Es gibt einen DEA  $A = (Q, \Sigma', \delta, q_0, F)$

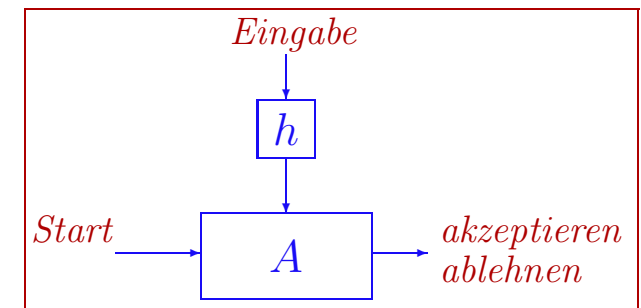
mit  $L = L(A) = \{w \in \Sigma'^* \mid \hat{\delta}(q_0, w) \in F\}$

$\Rightarrow h^{-1}(L) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, h(w)) \in F\}$

Konstruiere  $A_h = (Q, \Sigma, \delta_h, q_0, F)$  mit  $\delta_h(q, a) = \hat{\delta}(q, h(a))$

Dann gilt  $\hat{\delta}_h(q, w) = \hat{\delta}(q, h(w))$  für alle  $q \in Q$  und  $w \in \Sigma^*$

$\Rightarrow h^{-1}(L) = \{w \in \Sigma^* \mid \hat{\delta}_h(q_0, h(w)) \in F\} = L(A_h)$  **regulär**



- Welche Eigenschaften sind **automatisch prüfbar**?

- Ist die Sprache eines Automaten **leer**?
- **Zugehörigkeit**: Ist ein Wort  $w$  **Element der Sprache** eines Automaten?
- **Äquivalenz**: Beschreiben zwei Automaten **dieselbe Sprache**?

- Welche Eigenschaften sind **automatisch prüfbar**?

- Ist die Sprache eines Automaten **leer**?
- **Zugehörigkeit**: Ist ein Wort  $w$  **Element der Sprache** eines Automaten?
- **Äquivalenz**: Beschreiben zwei Automaten **dieselbe Sprache**?

Gleiche Fragestellung für Grammatiken und reguläre Ausdrücke

- **Welche Eigenschaften sind automatisch prüfbar?**

- Ist die Sprache eines Automaten **leer**?
- **Zugehörigkeit**: Ist ein Wort  $w$  **Element der Sprache** eines Automaten?
- **Äquivalenz**: Beschreiben zwei Automaten **dieselbe Sprache**?

Gleiche Fragestellung für Grammatiken und reguläre Ausdrücke

- **Wechsel der Repräsentation ist effektiv**

- **NEA  $\mapsto$  DEA**: Teilmengenkonstruktion (exponentielle Aufblähung möglich)
- **$\epsilon$ -NEA  $\mapsto$  DEA**: Hüllenbildung + Teilmengenkonstruktion
- **DEA  $\mapsto$   $\epsilon$ -NEA/NEA**: Modifikation der Präsentation (Mengenklammern)

## ● Welche Eigenschaften sind **automatisch prüfbar**?

- Ist die Sprache eines Automaten **leer**?
- **Zugehörigkeit**: Ist ein Wort  $w$  **Element der Sprache** eines Automaten?
- **Äquivalenz**: Beschreiben zwei Automaten **dieselbe Sprache**?

Gleiche Fragestellung für Grammatiken und reguläre Ausdrücke

## ● **Wechsel der Repräsentation ist effektiv**

- **NEA**  $\mapsto$  **DEA**: Teilmengenkonstruktion (exponentielle Aufblähung möglich)
- **$\epsilon$ -NEA**  $\mapsto$  **DEA**: Hüllenbildung + Teilmengenkonstruktion
- **DEA**  $\mapsto$   **$\epsilon$ -NEA/NEA**: Modifikation der Präsentation (Mengenklammern)
- **DEA**  $\mapsto$  **RA**:  $R_{ij}^k$ -Methode oder Zustandselimination
- **RA**  $\mapsto$   **$\epsilon$ -NEA**: induktive Konstruktion von Automaten

## ● Welche Eigenschaften sind **automatisch prüfbar**?

- Ist die Sprache eines Automaten **leer**?
- **Zugehörigkeit**: Ist ein Wort  $w$  **Element der Sprache** eines Automaten?
- **Äquivalenz**: Beschreiben zwei Automaten **dieselbe Sprache**?

Gleiche Fragestellung für Grammatiken und reguläre Ausdrücke

## ● **Wechsel der Repräsentation ist effektiv**

- **NEA  $\mapsto$  DEA**: Teilmengenkonstruktion (exponentielle Aufblähung möglich)
- **$\epsilon$ -NEA  $\mapsto$  DEA**: Hüllenbildung + Teilmengenkonstruktion
- **DEA  $\mapsto$   $\epsilon$ -NEA/NEA**: Modifikation der Präsentation (Mengenklammern)
- **DEA  $\mapsto$  RA**:  $R_{ij}^k$ -Methode oder Zustandselimination
- **RA  $\mapsto$   $\epsilon$ -NEA**: induktive Konstruktion von Automaten
- **DEA  $\mapsto$  Typ-3 Grammatik**: Regeln für Überführungsschritte einführen
- **Typ-3 Grammatik  $\mapsto$  NEA**: Überführungstabelle codiert Regeln

## ● Welche Eigenschaften sind **automatisch prüfbar**?

- Ist die Sprache eines Automaten **leer**?
- **Zugehörigkeit**: Ist ein Wort  $w$  **Element der Sprache** eines Automaten?
- **Äquivalenz**: Beschreiben zwei Automaten **dieselbe Sprache**?

Gleiche Fragestellung für Grammatiken und reguläre Ausdrücke

## ● **Wechsel der Repräsentation ist effektiv**

- **NEA  $\mapsto$  DEA**: Teilmengenkonstruktion (exponentielle Aufblähung möglich)
- **$\epsilon$ -NEA  $\mapsto$  DEA**: Hüllenbildung + Teilmengenkonstruktion
- **DEA  $\mapsto$   $\epsilon$ -NEA/NEA**: Modifikation der Präsentation (Mengenklammern)
- **DEA  $\mapsto$  RA**:  $R_{ij}^k$ -Methode oder Zustandselimination
- **RA  $\mapsto$   $\epsilon$ -NEA**: induktive Konstruktion von Automaten
- **DEA  $\mapsto$  Typ-3 Grammatik**: Regeln für Überführungsschritte einführen
- **Typ-3 Grammatik  $\mapsto$  NEA**: Überführungstabelle codiert Regeln

**Es reicht, Tests für ein Modell zu beschreiben**

## ● Nichttriviales Problem

- Automaten: Gibt es überhaupt einen akzeptierenden Pfad?
- Reguläre Ausdrücke: Wird mindestens ein einziges Wort charakterisiert?
- Grammatiken: Wird überhaupt ein Wort aus dem Startzustand erzeugt?



## ● Nichttriviales Problem

- Automaten: Gibt es überhaupt einen akzeptierenden Pfad?
- Reguläre Ausdrücke: Wird mindestens ein einziges Wort charakterisiert?
- Grammatiken: Wird überhaupt ein Wort aus dem Startzustand erzeugt?

## ● Erreichbarkeitstest für DEA $A = (Q, \Sigma, \delta, q_0, F)$

- Wegen  $\hat{\delta}(q_0, \epsilon) = q_0$  ist  $q_0$  in 0 Schritten erreichbar
- $q$  in  $k$  Schritten erreichbar,  $\delta(q, a) = q' \Rightarrow q'$  in  $k+1$  Schritten erreichbar
- $L(A) = \emptyset \Leftrightarrow$  kein  $q \in F$  in  $|Q|$  Schritten erreichbar

# PRÜFE OB EINE REGULÄRE SPRACHE LEER IST

## ● Nichttriviales Problem

- Automaten: Gibt es überhaupt einen akzeptierenden Pfad?
- Reguläre Ausdrücke: Wird mindestens ein einziges Wort charakterisiert?
- Grammatiken: Wird überhaupt ein Wort aus dem Startzustand erzeugt?

## ● Erreichbarkeitstest für DEA $A = (Q, \Sigma, \delta, q_0, F)$

- Wegen  $\hat{\delta}(q_0, \epsilon) = q_0$  ist  $q_0$  in 0 Schritten erreichbar
- $q$  in  $k$  Schritten erreichbar,  $\delta(q, a) = q' \Rightarrow q'$  in  $k+1$  Schritten erreichbar
- $L(A) = \emptyset \Leftrightarrow$  kein  $q \in F$  in  $|Q|$  Schritten erreichbar

## ● Induktive Analyse für reguläre Ausdrücke

- $L(\emptyset) = \emptyset$ ,  $L(\epsilon) \neq \emptyset$ ,  $L(a) \neq \emptyset$
- $L((E)) = \emptyset \Leftrightarrow L(E) = \emptyset$  keine Änderung
- $L(E+F) = \emptyset \Leftrightarrow L(E) = \emptyset \wedge L(F) = \emptyset$  Vereinigung von Elementen
- $L(E \circ F) = \emptyset \Leftrightarrow L(E) = \emptyset \vee L(F) = \emptyset$  Elemente beider Sprachen nötig
- $L(E^*) \neq \emptyset$ ,  $\epsilon$  gehört immer zu  $L(E^*)$

- **Unterschiedlich schwierig je nach Repräsentation**

- Automaten: Gibt es einen akzeptierenden Pfad für  $w$ ?
- Reguläre Ausdrücke: Wird das Wort  $w$  von der Charakterisierung erfasst?
- Grammatiken: Kann  $w$  aus dem Startzustand erzeugt werden?

- Unterschiedlich schwierig je nach Repräsentation

- Automaten: Gibt es einen akzeptierenden Pfad für  $w$ ?
- Reguläre Ausdrücke: Wird das Wort  $w$  von der Charakterisierung erfasst?
- Grammatiken: Kann  $w$  aus dem Startzustand erzeugt werden?

- **Abarbeitung** durch DEA  $A = (Q, \Sigma, \delta, q_0, F)$

- Bestimme  $q := \hat{\delta}(q_0, w)$  und teste  $q \in F$
- Maximal  $|w| + |F|$  Arbeitsschritte

# TEST AUF ZUGEHÖRIGKEIT

- Unterschiedlich schwierig je nach Repräsentation

- Automaten: Gibt es einen akzeptierenden Pfad für  $w$ ?
- Reguläre Ausdrücke: Wird das Wort  $w$  von der Charakterisierung erfasst?
- Grammatiken: Kann  $w$  aus dem Startzustand erzeugt werden?

- Abarbeitung durch DEA  $A = (Q, \Sigma, \delta, q_0, F)$

- Bestimme  $q := \hat{\delta}(q_0, w)$  und teste  $q \in F$
- Maximal  $|w| + |F|$  Arbeitsschritte

Test für andere Repräsentationen durch  
Umwandlung in DEA

# TEST AUF ÄQUIVALENZ VON SPRACHEN

- Wann sind **zwei reguläre Sprachen gleich**?

- Nichttrivial, da **Beschreibungsformen** sehr verschieden sein können
  - Verschiedene Automaten, Grammatiken, Ausdrücke, Mischformen, ...

# TEST AUF ÄQUIVALENZ VON SPRACHEN

- Wann sind **zwei reguläre Sprachen gleich**?
  - Nichttrivial, da **Beschreibungsformen** sehr verschieden sein können
    - Verschiedene Automaten, Grammatiken, Ausdrücke, Mischformen, ...
- Gibt es eine **“kanonische” Repräsentation**?
  - z.B. · **Transformiere** alles in deterministische endliche Automaten
    - Erzeuge **Standardversion** mit kleinstmöglicher Anzahl von Zuständen
  - Äquivalenztest prüft dann, ob der **gleiche Standardautomat erzeugt** wird

# TEST AUF ÄQUIVALENZ VON SPRACHEN

- Wann sind **zwei reguläre Sprachen gleich**?

- Nichttrivial, da **Beschreibungsformen** sehr verschieden sein können
  - Verschiedene Automaten, Grammatiken, Ausdrücke, Mischformen, ...

- Gibt es eine **“kanonische” Repräsentation**?

- z.B. · **Transformiere** alles in deterministische endliche Automaten
  - Erzeuge **Standardversion** mit kleinstmöglicher Anzahl von Zuständen
- Äquivalenztest prüft dann, ob der **gleiche Standardautomat erzeugt** wird

- Wie **standardisiert man Automaten**?

- **Entferne Zustände**, die vom Startzustand **unerreichbar** sind
- **Fasse Zustände zusammen**, die für alle Worte **“äquivalent”** sind
  - Es führen exakt dieselben Worte zu akzeptierenden Zuständen
- Ergibt **minimalen äquivalenten Automaten**



# ÄQUIVALENZTEST FÜR ZUSTÄNDE

- **Äquivalenz** der Zustände  $p$  und  $q$  ( $p \cong q$ )
  - Für alle Worte  $w \in \Sigma^*$  gilt  $\hat{\delta}(p, w) \in F \Leftrightarrow \hat{\delta}(q, w) \in F$
  - Die Worte müssen nicht zum gleichen Zustand führen

# ÄQUIVALENZTEST FÜR ZUSTÄNDE

- **Äquivalenz** der Zustände  $p$  und  $q$  ( $p \cong q$ )
  - Für alle Worte  $w \in \Sigma^*$  gilt  $\hat{\delta}(p, w) \in F \Leftrightarrow \hat{\delta}(q, w) \in F$
  - Die Worte müssen nicht zum gleichen Zustand führen
- **Positives Prüfverfahren schwierig**
  - Man muß **alle** Worte überprüfen, die von einem Zustand ausgehen
  - Man kann sich auf Worte der maximalen Länge  $|Q|$  beschränken
  - Besser: Nichtäquivalente (**unterscheidbare**) Zustände identifizieren

# ÄQUIVALENZTEST FÜR ZUSTÄNDE

- **Äquivalenz** der Zustände  $p$  und  $q$  ( $p \cong q$ )

- Für alle Worte  $w \in \Sigma^*$  gilt  $\hat{\delta}(p, w) \in F \Leftrightarrow \hat{\delta}(q, w) \in F$
- Die Worte müssen nicht zum gleichen Zustand führen

- **Positives Prüfverfahren schwierig**

- Man muß **alle** Worte überprüfen, die von einem Zustand ausgehen
- Man kann sich auf Worte der maximalen Länge  $|Q|$  beschränken
- Besser: Nichtäquivalente (**unterscheidbare**) Zustände identifizieren

- **Table-Filling Algorithmus**

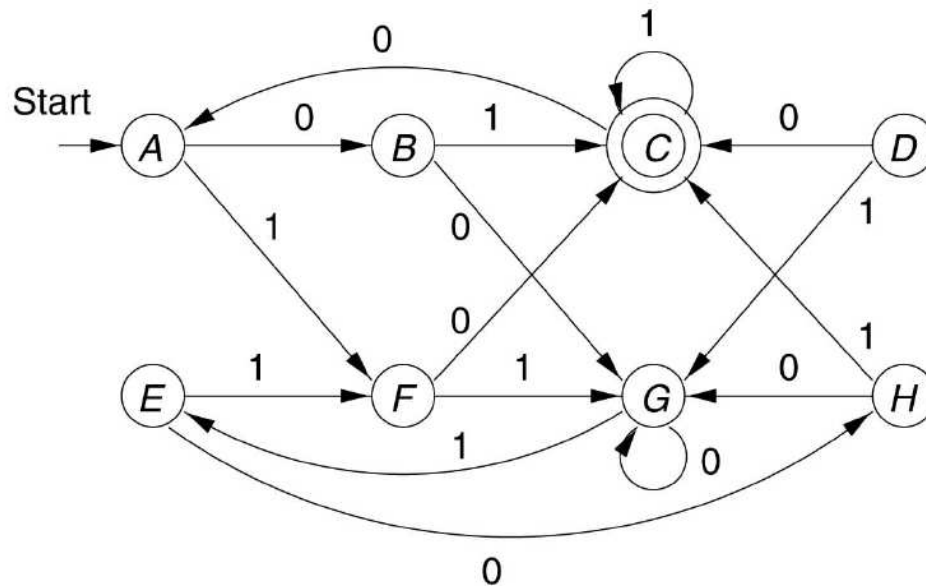
Markiere Unterscheidbarkeit von Zuständen in Tabelle

- Start:  $p \not\cong q$ , falls  $p \in F$  und  $q \notin F$
- Iteration:  $p \not\cong q$ , falls  $\delta(p, a) \not\cong \delta(q, a)$  für ein  $a \in \Sigma$

In jeder Iteration werden nur noch ungeklärte Paare überprüft

Nach **maximal  $|Q|$  Iterationen** sind alle Unterschiede bestimmt

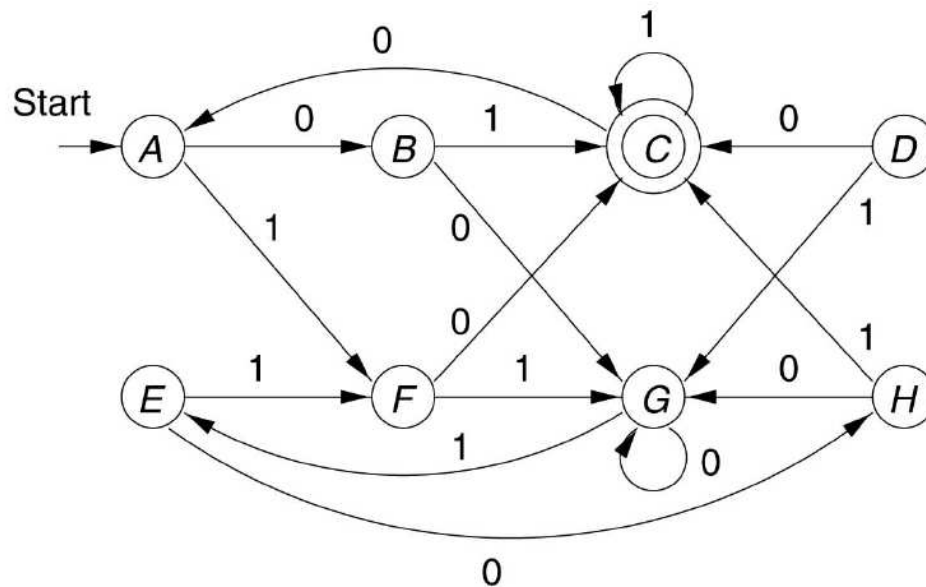
# ÄQUIVALENZTEST AM BEISPIEL



	A	B	C	D	E	F	G	H
A	\							
B		\						
C			\					
D				\				
E					\			
F						\		
G							\	
H								\

*Tabelle der Unterschiede*

# ÄQUIVALENZTEST AM BEISPIEL

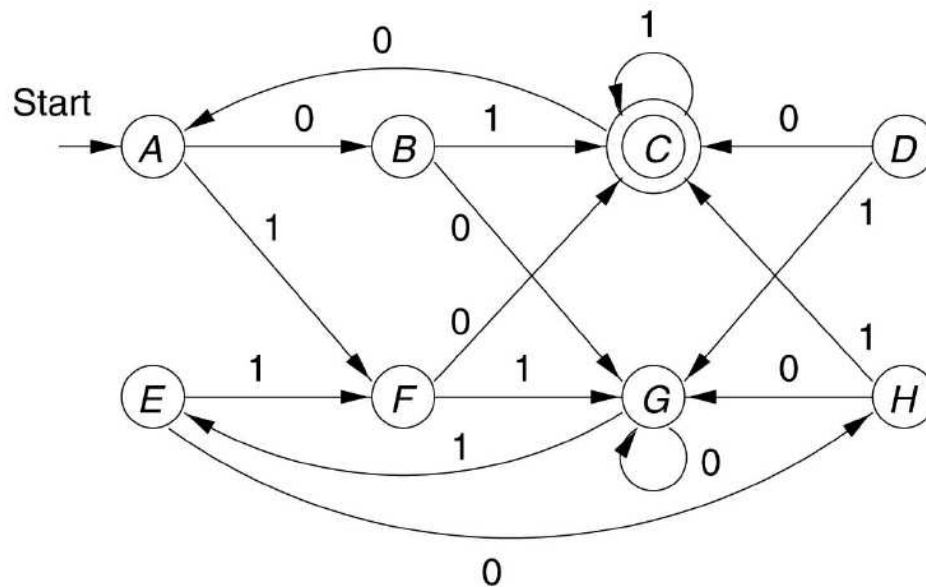


	A	B	C	D	E	F	G	H
A	\		×					
B		\	×					
C	×	×	\	×	×	×	×	×
D			×	\				
E			×		\			
F			×			\		
G			×				\	
H			×					\

*Tabelle der Unterschiede*

1. Unterscheide akzeptierende Zustände von anderen

# ÄQUIVALENZTEST AM BEISPIEL

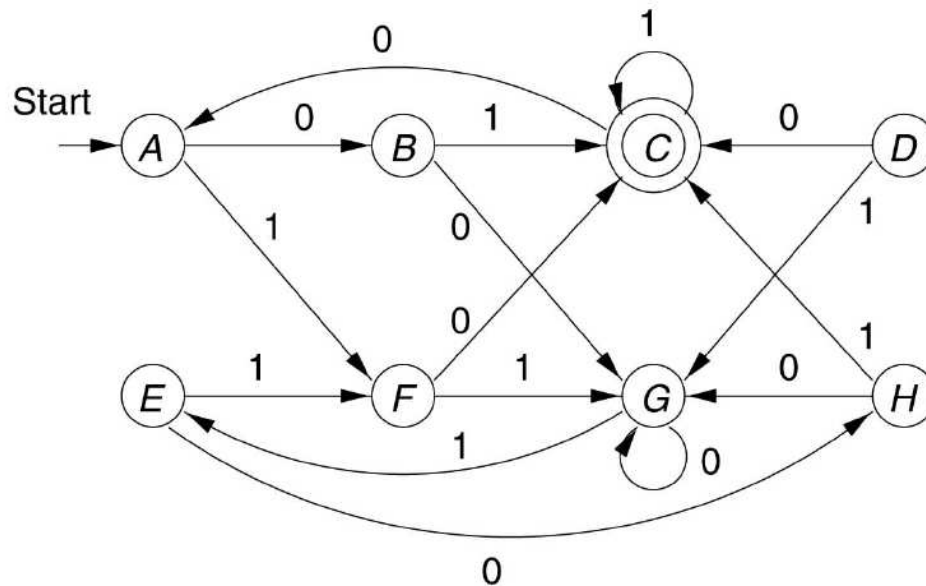


	A	B	C	D	E	F	G	H
A	\		×	×		×		
B		\	×	×		×		
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E			×	×	\	×		
F	×	×	×		×	\	×	×
G			×	×		×	\	
H			×	×		×		\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen

# ÄQUIVALENZTEST AM BEISPIEL

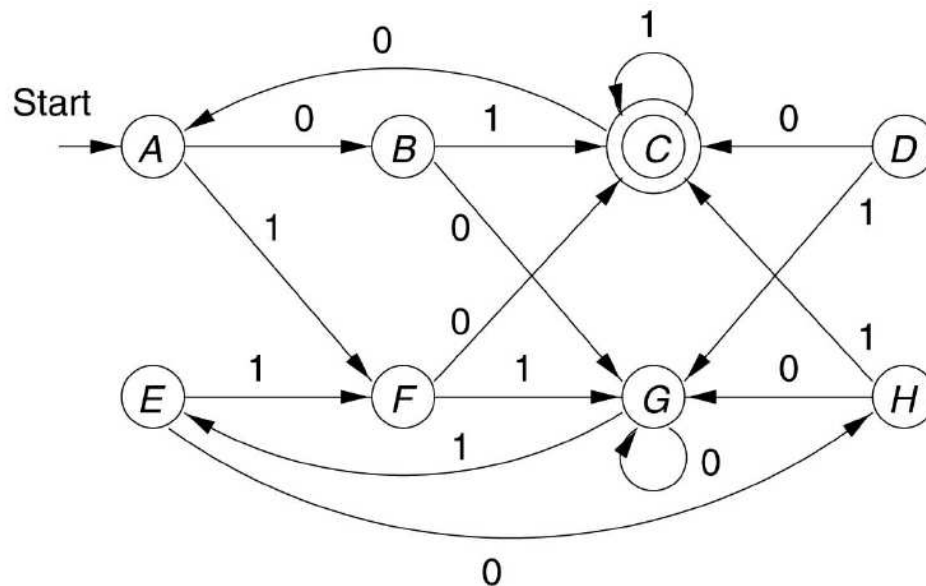


	A	B	C	D	E	F	G	H
A	\	×	×	×		×		×
B	×	\	×	×	×	×	×	
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E		×	×	×	\	×		×
F	×	×	×		×	\	×	×
G		×	×	×		×	\	×
H	×		×	×	×	×	×	\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen
- 2b. **Eingabesymbol 1**: Nur *B* und *H* führen zu akzeptierenden Zuständen

# ÄQUIVALENZTEST AM BEISPIEL



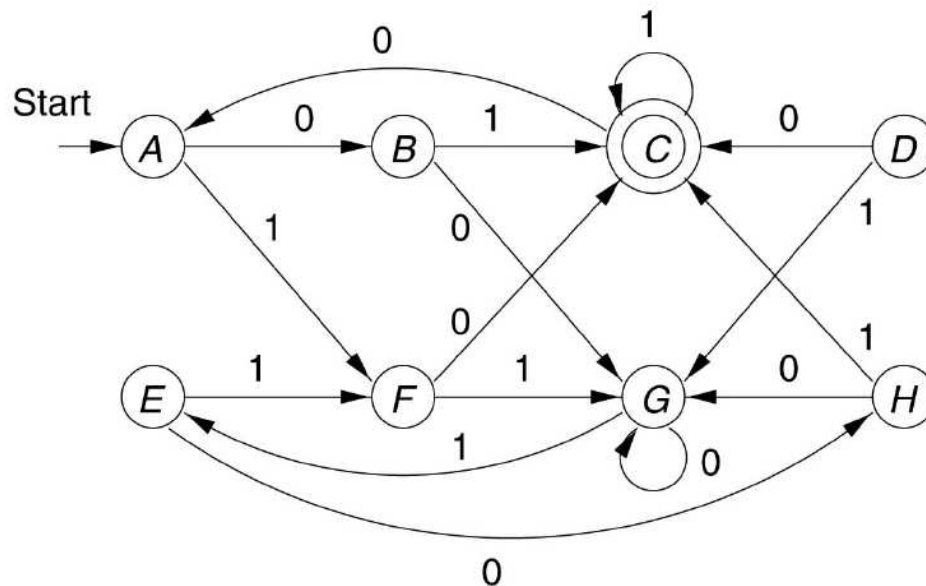
	A	B	C	D	E	F	G	H
A	\	×	×	×		×		×
B	×	\	×	×	×	×	×	
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E		×	×	×	\	×		×
F	×	×	×		×	\	×	×
G		×	×	×		×	\	×
H	×		×	×	×	×	×	\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen
- 2b. **Eingabesymbol 1**: Nur *B* und *H* führen zu akzeptierenden Zuständen
3. Überprüfe Nachfolger von  $\{A, E\}$



# ÄQUIVALENZTEST AM BEISPIEL

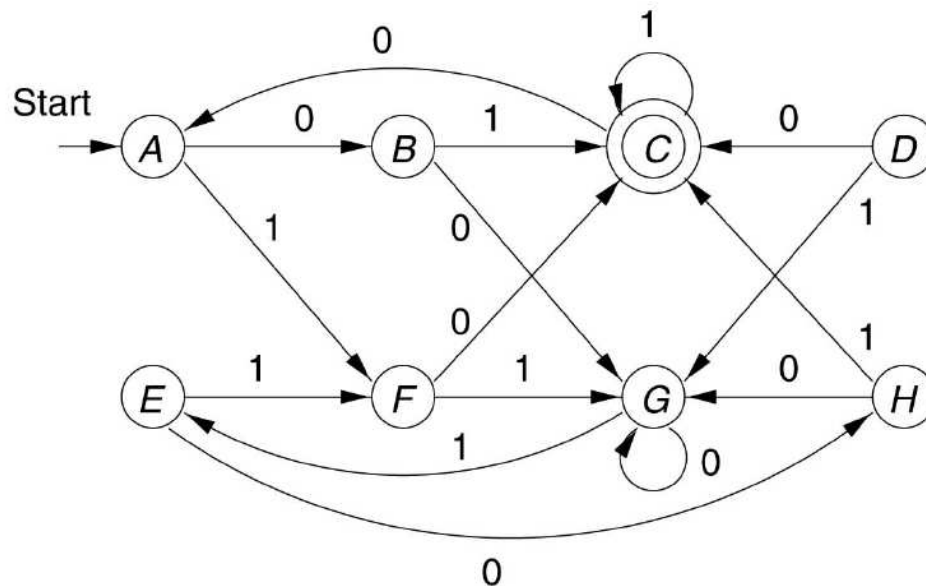


	A	B	C	D	E	F	G	H
A	\	×	×	×		×	×	×
B	×	\	×	×	×	×	×	
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E		×	×	×	\	×		×
F	×	×	×		×	\	×	×
G	×	×	×	×		×	\	×
H	×		×	×	×	×	×	\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen
- 2b. **Eingabesymbol 1**: Nur *B* und *H* führen zu akzeptierenden Zuständen
3. Überprüfe Nachfolger von  $\{A, E\}$ ,  $\{A, G\}$

# ÄQUIVALENZTEST AM BEISPIEL

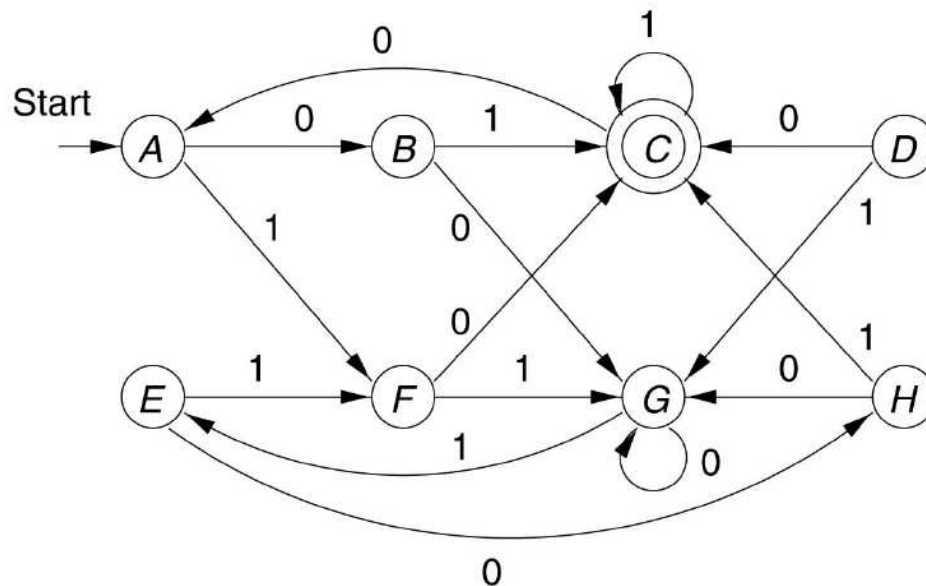


	A	B	C	D	E	F	G	H
A	\	×	×	×		×	×	×
B	×	\	×	×	×	×	×	
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E		×	×	×	\	×		×
F	×	×	×		×	\	×	×
G	×	×	×	×		×	\	×
H	×		×	×	×	×	×	\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen
- 2b. **Eingabesymbol 1**: Nur *B* und *H* führen zu akzeptierenden Zuständen
3. Überprüfe Nachfolger von  $\{A,E\}$ ,  $\{A,G\}$ ,  $\{B,H\}$

# ÄQUIVALENZTEST AM BEISPIEL

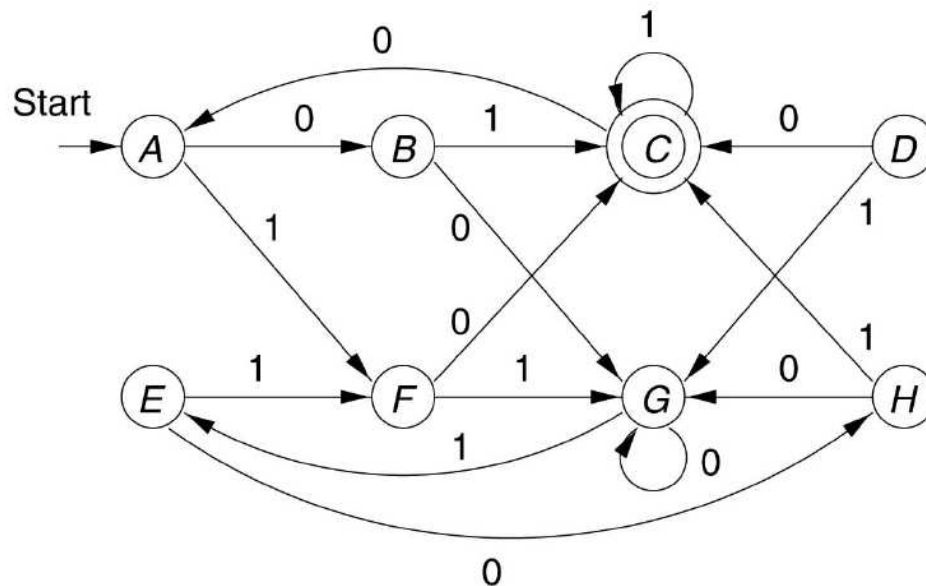


	A	B	C	D	E	F	G	H
A	\	×	×	×		×	×	×
B	×	\	×	×	×	×	×	
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E		×	×	×	\	×		×
F	×	×	×		×	\	×	×
G	×	×	×	×		×	\	×
H	×		×	×	×	×	×	\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen
- 2b. **Eingabesymbol 1**: Nur *B* und *H* führen zu akzeptierenden Zuständen
3. Überprüfe Nachfolger von  $\{A,E\}$ ,  $\{A,G\}$ ,  $\{B,H\}$  ,  $\{D,F\}$

# ÄQUIVALENZTEST AM BEISPIEL

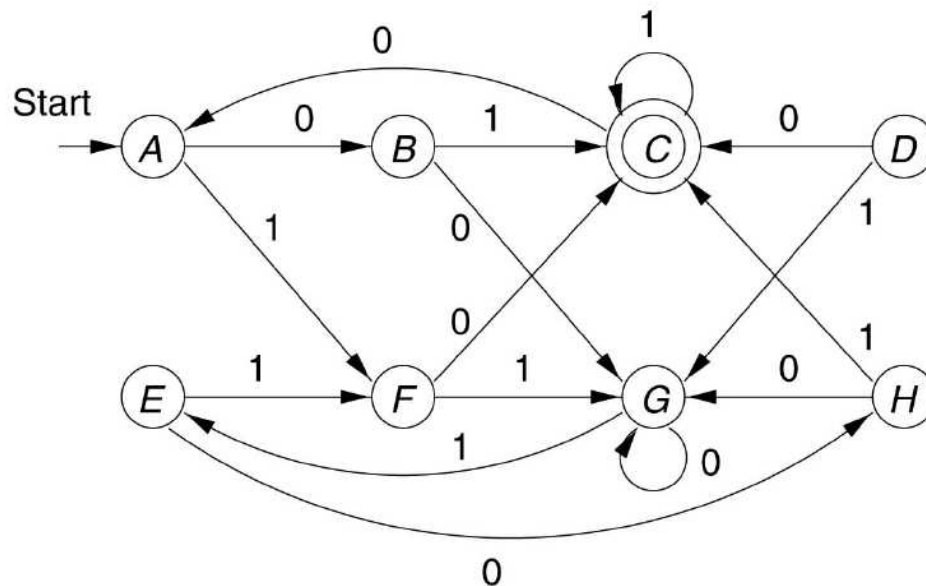


	A	B	C	D	E	F	G	H
A	\	×	×	×		×	×	×
B	×	\	×	×	×	×	×	
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E		×	×	×	\	×	×	×
F	×	×	×		×	\	×	×
G	×	×	×	×	×	×	\	×
H	×		×	×	×	×	×	\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen
- 2b. **Eingabesymbol 1**: Nur *B* und *H* führen zu akzeptierenden Zuständen
3. Überprüfe Nachfolger von  $\{A,E\}$ ,  $\{A,G\}$ ,  $\{B,H\}$ ,  $\{D,F\}$  und  $\{E,G\}$ .

# ÄQUIVALENZTEST AM BEISPIEL

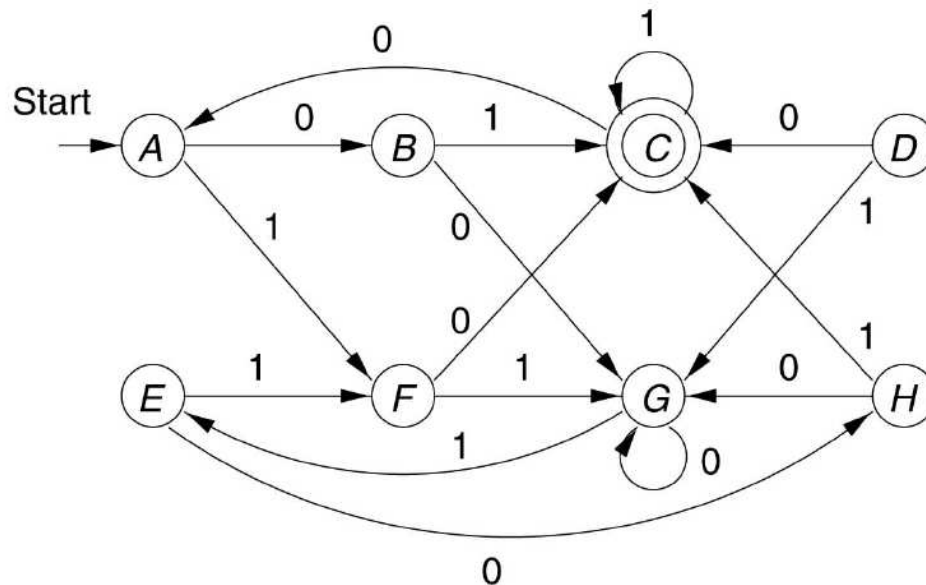


	A	B	C	D	E	F	G	H
A	\	×	×	×		×	×	×
B	×	\	×	×	×	×	×	
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E		×	×	×	\	×	×	×
F	×	×	×		×	\	×	×
G	×	×	×	×	×	×	\	×
H	×		×	×	×	×	×	\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen
- 2b. **Eingabesymbol 1**: Nur *B* und *H* führen zu akzeptierenden Zuständen
3. Überprüfe Nachfolger von  $\{A,E\}$ ,  $\{A,G\}$ ,  $\{B,H\}$ ,  $\{D,F\}$  und  $\{E,G\}$ .
4. Überprüfung von  $\{A,E\}$ ,  $\{B,H\}$  und  $\{D,F\}$  gibt keine Unterschiede

# ÄQUIVALENZTEST AM BEISPIEL



	A	B	C	D	E	F	G	H
A	\	×	×	×		×	×	×
B	×	\	×	×	×	×	×	
C	×	×	\	×	×	×	×	×
D	×	×	×	\	×		×	×
E		×	×	×	\	×	×	×
F	×	×	×		×	\	×	×
G	×	×	×	×	×	×	\	×
H	×		×	×	×	×	×	\

*Tabelle der Unterschiede*

1. Unterscheide **akzeptierende Zustände** von anderen
- 2a. **Eingabesymbol 0**: Nur *D* und *F* führen zu akzeptierenden Zuständen
- 2b. **Eingabesymbol 1**: Nur *B* und *H* führen zu akzeptierenden Zuständen
3. Überprüfe Nachfolger von  $\{A,E\}$ ,  $\{A,G\}$ ,  $\{B,H\}$ ,  $\{D,F\}$  und  $\{E,G\}$ .
4. Überprüfung von  $\{A,E\}$ ,  $\{B,H\}$  und  $\{D,F\}$  gibt keine Unterschiede

**Äquivalenzklassen sind  $\{A,E\}$ ,  $\{B,H\}$ ,  $\{D,F\}$ ,  $\{C\}$  und  $\{G\}$**

# ÄQUIVALENZTEST FÜR SPRACHEN

- **Prüfverfahren**

- Standardisiere Beschreibungsform in zwei disjunkte DEAs  $A_1$  und  $A_2$  um

# ÄQUIVALENZTEST FÜR SPRACHEN

## ● Prüfverfahren

- Standardisiere Beschreibungsform in zwei disjunkte DEAs  $A_1$  und  $A_2$  um
- Vereinige Automaten zu  $A = (Q_1 \cup Q_2, \Sigma, \delta_1 \cup \delta_2, q', F_1 \cup F_2)$



# ÄQUIVALENZTEST FÜR SPRACHEN

## ● Prüfverfahren

- Standardisiere Beschreibungsform in zwei disjunkte DEAs  $A_1$  und  $A_2$  um
- Vereinige Automaten zu  $A = (Q_1 \cup Q_2, \Sigma, \delta_1 \cup \delta_2, q', F_1 \cup F_2)$
- Bilde Äquivalenzklassen von  $A$  und teste ob  $q_{0,1}$  und  $q_{0,2}$  äquivalent sind

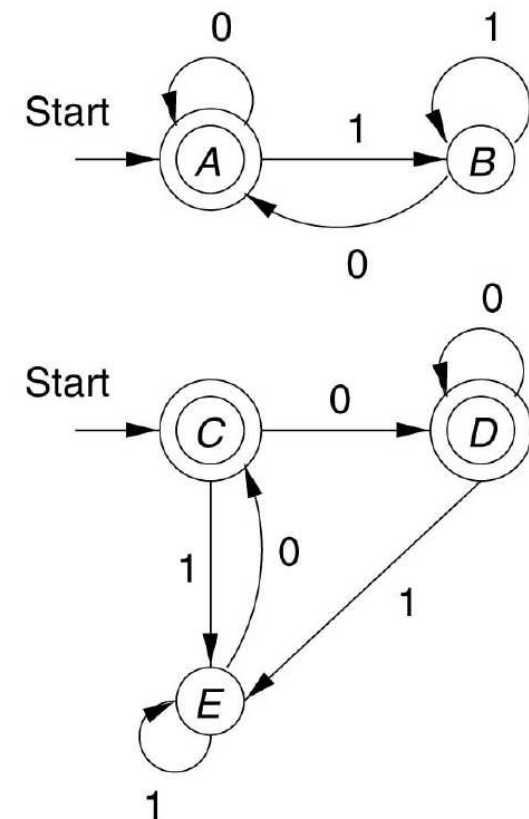
# ÄQUIVALENZTEST FÜR SPRACHEN

## ● Prüfverfahren

- Standardisiere Beschreibungsform in zwei disjunkte DEAs  $A_1$  und  $A_2$  um
- Vereinige Automaten zu  $A = (Q_1 \cup Q_2, \Sigma, \delta_1 \cup \delta_2, q', F_1 \cup F_2)$
- Bilde Äquivalenzklassen von  $A$  und teste ob  $q_{0,1}$  und  $q_{0,2}$  äquivalent sind

## ● Zwei DEAs für $L(\epsilon + (0 + 1)^*0)$

- Äquivalenzklassen sind  $\{A, C, D\}$  und  $\{B, E\}$
- Da  $A$  und  $C$  äquivalent sind,  
sind die Automaten äquivalent



## Konstruiere äquivalenten DEA mit minimaler Menge von Zuständen

- **Entferne überflüssige Zustände**
  - $q$  ist **überflüssig**, wenn  $\hat{\delta}(q_0, w) \neq q$  für alle Worte  $w \in \Sigma^*$
  - Reduziere  $Q$  zu Menge der erreichbaren Zustände (Verfahren auf Folie 11)

## Konstruiere äquivalenten DEA mit minimaler Menge von Zuständen

- **Entferne überflüssige Zustände**
  - $q$  ist **überflüssig**, wenn  $\hat{\delta}(q_0, w) \neq q$  für alle Worte  $w \in \Sigma^*$
  - Reduziere  $Q$  zu Menge der erreichbaren Zustände (Verfahren auf Folie 11)
- **Fasse äquivalente Zustände zusammen**
  - Bestimme Menge der Äquivalenzklassen von  $Q$
  - Setze  $Q'$  als Menge der Äquivalenzklassen von  $Q$
  - Setze  $\delta'(S, a) = \bigcup_{q \in S} \delta(q, a)$   
Wohldefiniert, da alle Nachfolger äquivalenter Zustände äquivalent

## Konstruiere äquivalenten DEA mit minimaler Menge von Zuständen

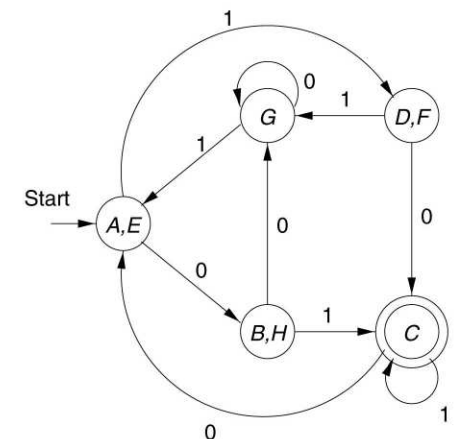
- **Entferne überflüssige Zustände**

- $q$  ist **überflüssig**, wenn  $\hat{\delta}(q_0, w) \neq q$  für alle Worte  $w \in \Sigma^*$
- **Reduziere**  $Q$  zu Menge der erreichbaren Zustände (Verfahren auf Folie 11)

- **Fasse äquivalente Zustände zusammen**

- Bestimme Menge der Äquivalenzklassen von  $Q$
- Setze  $Q'$  als Menge der Äquivalenzklassen von  $Q$
- Setze  $\delta'(S, a) = \bigcup_{a \in S} \delta(q, a)$   
Wohldefiniert, da alle Nachfolger äquivalenter Zustände äquivalent

- **Minimalversion des Beispielautomaten:**



## Konstruiere äquivalenten DEA mit minimaler Menge von Zuständen

- **Entferne überflüssige Zustände**

- $q$  ist **überflüssig**, wenn  $\hat{\delta}(q_0, w) \neq q$  für alle Worte  $w \in \Sigma^*$
- **Reduziere**  $Q$  zu Menge der erreichbaren Zustände (Verfahren auf Folie 11)

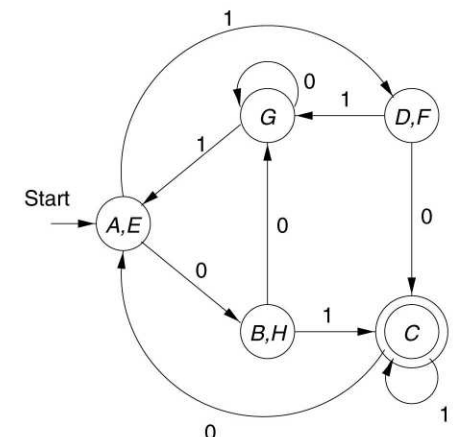
- **Fasse äquivalente Zustände zusammen**

- Bestimme Menge der Äquivalenzklassen von  $Q$
- Setze  $Q'$  als Menge der Äquivalenzklassen von  $Q$
- Setze  $\delta'(S, a) = \bigcup_{a \in S} \delta(q, a)$

Wohldefiniert, da alle Nachfolger äquivalenter Zustände äquivalent

- **Minimalversion des Beispielautomaten:**

- **Resultierender Automat ist minimal**



# GRENZEN REGULÄRER SPRACHEN

Wie zeigt man, daß eine Sprache  $L$  nicht regulär ist?

Wie zeigt man, daß eine Sprache  $L$  nicht regulär ist?

- **Direkter Nachweis**

- Zeige, daß kein endlicher Automat genau die Worte von  $L$  erkennt
- Sprache muß **unendlich** sein und **komplizierte Struktur** haben
- Technisches Hilfsmittel: **Pumping Lemma**



## Wie zeigt man, daß eine Sprache $L$ nicht regulär ist?

- **Direkter Nachweis**

- Zeige, daß kein endlicher Automat genau die Worte von  $L$  erkennt
- Sprache muß **unendlich** sein und **komplizierte Struktur** haben
- Technisches Hilfsmittel: **Pumping Lemma**

- **Verwendung der **Abschlußeigenschaften****

- Zeige daß Regularität von  $L$  dazu führen würde, daß eine als nichtregulär bekannte Sprache regulär sein müsste
- Häufige Technik: **(inverse) Homomorphismen**

# DAS PUMPING LEMMA FÜR REGULÄRE SPRACHEN

- Warum ist  $\{0^n 1^n \mid n \in \mathbb{N}\}$  nicht regulär?

# DAS PUMPING LEMMA FÜR REGULÄRE SPRACHEN

- Warum ist  $\{0^n 1^n \mid n \in \mathbb{N}\}$  nicht regulär?
  - Ein DFA muß alle Nullen beim Abarbeiten zählen und dann vergleichen

# DAS PUMPING LEMMA FÜR REGULÄRE SPRACHEN

- Warum ist  $\{0^n 1^n \mid n \in \mathbb{N}\}$  nicht regulär?
  - Ein DFA muß alle Nullen beim Abarbeiten zählen und dann vergleichen
  - Für  $n > |Q|$  muß ein Zustand von  $A$  doppelt benutzt worden sein

# DAS PUMPING LEMMA FÜR REGULÄRE SPRACHEN

- Warum ist  $\{0^n 1^n \mid n \in \mathbb{N}\}$  nicht regulär?
  - Ein DFA muß alle Nullen beim Abarbeiten zählen und dann vergleichen
  - Für  $n > |Q|$  muß ein Zustand von  $A$  doppelt benutzt worden sein
  - Eine  $\delta$ -Schleife mit  $k$  Zuständen bedeutet, daß  $A$  auch  $0^{n+k} 1^n$  akzeptiert

# DAS PUMPING LEMMA FÜR REGULÄRE SPRACHEN

- Warum ist  $\{0^n 1^n \mid n \in \mathbb{N}\}$  nicht regulär?

- Ein DFA muß alle Nullen beim Abarbeiten zählen und dann vergleichen
- Für  $n > |Q|$  muß ein Zustand von  $A$  doppelt benutzt worden sein
- Eine  $\delta$ -Schleife mit  $k$  Zuständen bedeutet, daß  $A$  auch  $0^{n+k} 1^n$  akzeptiert

- Allgemeine Version: **Pumping Lemma**

Für jede reguläre Sprache  $L \in \mathcal{L}_3$  gibt es eine Zahl  $n \in \mathbb{N}$ , so daß jedes Wort  $w \in L$  mit Länge  $|w| \geq n$  zerlegt werden kann in  $w = x y z$  mit den Eigenschaften

- (1)  $y \neq \epsilon$ ,
- (2)  $|x y| \leq n$  und
- (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

# DAS PUMPING LEMMA FÜR REGULÄRE SPRACHEN

- Warum ist  $\{0^n 1^n \mid n \in \mathbb{N}\}$  nicht regulär?

- Ein DFA muß alle Nullen beim Abarbeiten zählen und dann vergleichen
- Für  $n > |Q|$  muß ein Zustand von  $A$  doppelt benutzt worden sein
- Eine  $\delta$ -Schleife mit  $k$  Zuständen bedeutet, daß  $A$  auch  $0^{n+k} 1^n$  akzeptiert

- Allgemeine Version: **Pumping Lemma**

Für jede reguläre Sprache  $L \in \mathcal{L}_3$  gibt es eine Zahl  $n \in \mathbb{N}$ , so daß jedes Wort  $w \in L$  mit Länge  $|w| \geq n$  zerlegt werden kann in  $w = x y z$  mit den Eigenschaften

- (1)  $y \neq \epsilon$ ,
- (2)  $|x y| \leq n$  und
- (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

- Aussage ist wechselseitig konstruktiv

- Die Zahl  $n$  kann zu jeder regulären Sprache  $L$  bestimmt werden
- Die Zerlegung  $w = x y z$  kann zu jedem Wort  $w \in L$  bestimmt werden

# BEWEIS DES PUMPING LEMMAS

Für jede Sprache  $L \in \mathcal{L}_3$  gibt es ein  $n \in \mathbb{N}$ , so daß jedes  $w \in L$  mit  $|w| \geq n$  zerlegbar ist in  $w = x y z$  mit den Eigenschaften

(1)  $y \neq \epsilon$ , (2)  $|x y| \leq n$  und (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

- Beweis mit Automaten



# BEWEIS DES PUMPING LEMMAS

Für jede Sprache  $L \in \mathcal{L}_3$  gibt es ein  $n \in \mathbb{N}$ , so daß jedes  $w \in L$  mit  $|w| \geq n$  zerlegbar ist in  $w = x y z$  mit den Eigenschaften

(1)  $y \neq \epsilon$ , (2)  $|x y| \leq n$  und (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

## ● Beweis mit Automaten

– Sei  $L$  regulär und  $A = (Q, \Sigma, \delta, q_0, F)$  ein DEA mit  $L = L(A)$

# BEWEIS DES PUMPING LEMMAS

Für jede Sprache  $L \in \mathcal{L}_3$  gibt es ein  $n \in \mathbb{N}$ , so daß jedes  $w \in L$  mit  $|w| \geq n$  zerlegbar ist in  $w = x y z$  mit den Eigenschaften  
(1)  $y \neq \epsilon$ , (2)  $|x y| \leq n$  und (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

## ● Beweis mit Automaten

- Sei  $L$  regulär und  $A = (Q, \Sigma, \delta, q_0, F)$  ein DEA mit  $L = L(A)$
- Wähle  $n = |Q|$ . Betrachte  $w = a_1 .. a_m$  mit  $|w| \geq n$  und  $p_i := \hat{\delta}(q_0, a_1 .. a_i)$

# BEWEIS DES PUMPING LEMMAS

Für jede Sprache  $L \in \mathcal{L}_3$  gibt es ein  $n \in \mathbb{N}$ , so daß jedes  $w \in L$  mit  $|w| \geq n$  zerlegbar ist in  $w = x y z$  mit den Eigenschaften  
(1)  $y \neq \epsilon$ , (2)  $|x y| \leq n$  und (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

## ● Beweis mit Automaten

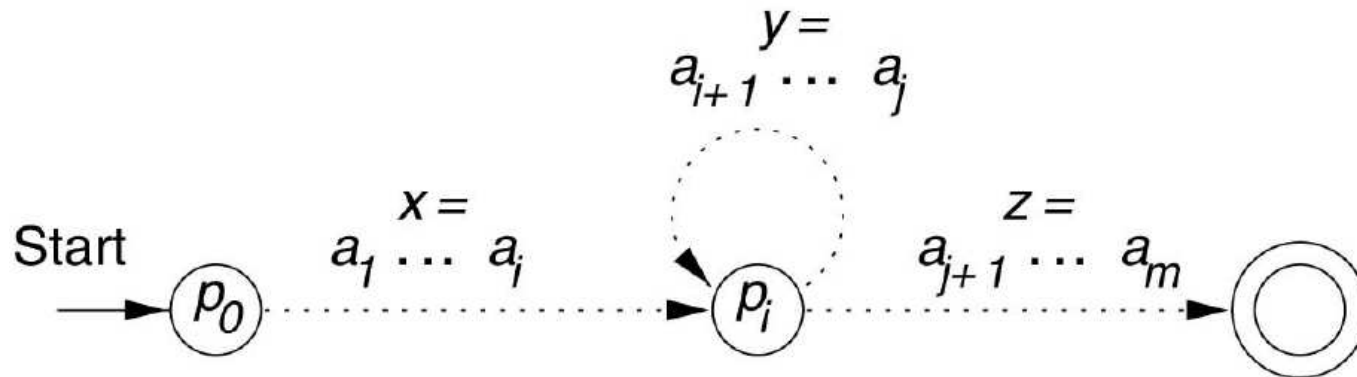
- Sei  $L$  regulär und  $A = (Q, \Sigma, \delta, q_0, F)$  ein DEA mit  $L = L(A)$
- Wähle  $n = |Q|$ . Betrachte  $w = a_1..a_m$  mit  $|w| \geq n$  und  $p_i := \hat{\delta}(q_0, a_1..a_i)$
- Dann gibt es  $i, j$  mit  $0 \leq i < j \leq n$  und  $p_i = p_j$  (Schubfachprinzip)

# BEWEIS DES PUMPING LEMMAS

Für jede Sprache  $L \in \mathcal{L}_3$  gibt es ein  $n \in \mathbb{N}$ , so daß jedes  $w \in L$  mit  $|w| \geq n$  zerlegbar ist in  $w = x y z$  mit den Eigenschaften  
(1)  $y \neq \epsilon$ , (2)  $|x y| \leq n$  und (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

## ● Beweis mit Automaten

- Sei  $L$  regulär und  $A = (Q, \Sigma, \delta, q_0, F)$  ein DEA mit  $L = L(A)$
- Wähle  $n = |Q|$ . Betrachte  $w = a_1 \dots a_m$  mit  $|w| \geq n$  und  $p_i := \hat{\delta}(q_0, a_1 \dots a_i)$
- Dann gibt es  $i, j$  mit  $0 \leq i < j \leq n$  und  $p_i = p_j$  (Schubfachprinzip)
- Zerlege  $w$  in  $w = x y z$  mit  $x = a_1 \dots a_i$ ,  $y = a_{i+1} \dots a_j$  und  $z = a_{j+1} \dots a_m$

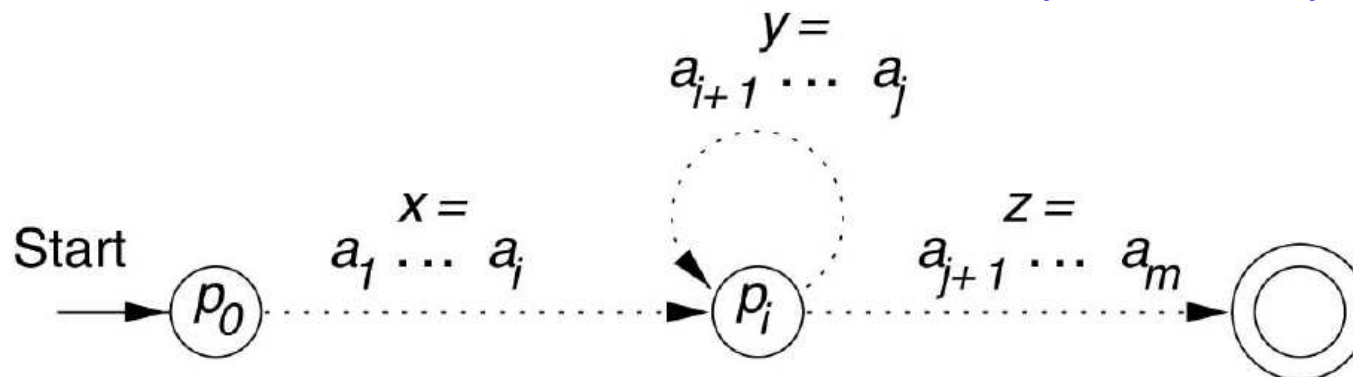


# BEWEIS DES PUMPING LEMMAS

Für jede Sprache  $L \in \mathcal{L}_3$  gibt es ein  $n \in \mathbb{N}$ , so daß jedes  $w \in L$  mit  $|w| \geq n$  zerlegbar ist in  $w = x y z$  mit den Eigenschaften  
(1)  $y \neq \epsilon$ , (2)  $|x y| \leq n$  und (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

## ● Beweis mit Automaten

- Sei  $L$  regulär und  $A = (Q, \Sigma, \delta, q_0, F)$  ein DEA mit  $L = L(A)$
- Wähle  $n = |Q|$ . Betrachte  $w = a_1 \dots a_m$  mit  $|w| \geq n$  und  $p_i := \hat{\delta}(q_0, a_1 \dots a_i)$
- Dann gibt es  $i, j$  mit  $0 \leq i < j \leq n$  und  $p_i = p_j$  (Schubfachprinzip)
- Zerlege  $w$  in  $w = x y z$  mit  $x = a_1 \dots a_i$ ,  $y = a_{i+1} \dots a_j$  und  $z = a_{j+1} \dots a_m$



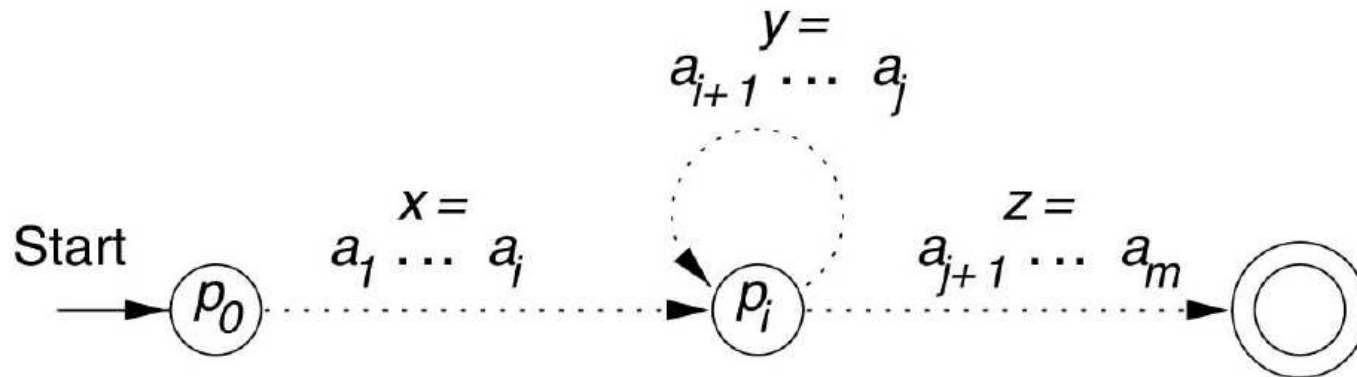
- Per Konstruktion gilt  $y \neq \epsilon$ ,  $|x y| \leq n$  und  $\hat{\delta}(p_i, y^k) = p_i$  für alle  $k \in \mathbb{N}$

# BEWEIS DES PUMPING LEMMAS

Für jede Sprache  $L \in \mathcal{L}_3$  gibt es ein  $n \in \mathbb{N}$ , so daß jedes  $w \in L$  mit  $|w| \geq n$  zerlegbar ist in  $w = x y z$  mit den Eigenschaften  
 (1)  $y \neq \epsilon$ , (2)  $|x y| \leq n$  und (3) für alle  $k \in \mathbb{N}$  ist  $x y^k z \in L$

## ● Beweis mit Automaten

- Sei  $L$  regulär und  $A = (Q, \Sigma, \delta, q_0, F)$  ein DEA mit  $L = L(A)$
- Wähle  $n = |Q|$ . Betrachte  $w = a_1 \dots a_m$  mit  $|w| \geq n$  und  $p_i := \hat{\delta}(q_0, a_1 \dots a_i)$
- Dann gibt es  $i, j$  mit  $0 \leq i < j \leq n$  und  $p_i = p_j$  (Schubfachprinzip)
- Zerlege  $w$  in  $w = x y z$  mit  $x = a_1 \dots a_i$ ,  $y = a_{i+1} \dots a_j$  und  $z = a_{j+1} \dots a_m$



- Per Konstruktion gilt  $y \neq \epsilon$ ,  $|x y| \leq n$  und  $\hat{\delta}(p_i, y^k) = p_i$  für alle  $k \in \mathbb{N}$
- Also  $\hat{\delta}(q_0, x y^k z) = \hat{\delta}(p_i, y^k z) = \hat{\delta}(p_i, y z) = \hat{\delta}(q_0, x y z) = \hat{\delta}(q_0, w) \in F$

## ANWENDUNGEN DES PUMPING LEMMAS

- $L_1 = \{0^m 1^m \mid m \in \mathbb{N}\}$  ist nicht regulär
  - Wir nehmen an  $L_1$  sei regulär

## ANWENDUNGEN DES PUMPING LEMMAS

- $L_1 = \{0^m 1^m \mid m \in \mathbb{N}\}$  ist nicht regulär
  - Wir nehmen an  $L_1$  sei regulär
  - Wähle  $n$  entsprechend des Pumping Lemmas und  $m > n$



## ANWENDUNGEN DES PUMPING LEMMAS

- $L_1 = \{0^m 1^m \mid m \in \mathbb{N}\}$  ist nicht regulär
  - Wir nehmen an  $L_1$  sei regulär
  - Wähle  $n$  entsprechend des Pumping Lemmas und  $m > n$
  - Dann kann  $w = 0^m 1^m$  zerlegt werden in  $x=0^i$ ,  $y=0^j$   $z=0^{m-i-j}1^m$   
mit  $j \neq 0$  und  $i+j \leq n$  und  $x y^k z \in L_1$  für alle  $k \in \mathbb{N}$

## ANWENDUNGEN DES PUMPING LEMMAS

- $L_1 = \{0^m 1^m \mid m \in \mathbb{N}\}$  ist nicht regulär
  - Wir nehmen an  $L_1$  sei regulär
  - Wähle  $n$  entsprechend des Pumping Lemmas und  $m > n$
  - Dann kann  $w = 0^m 1^m$  zerlegt werden in  $x=0^i$ ,  $y=0^j$   $z=0^{m-i-j}1^m$   
mit  $j \neq 0$  und  $i+j \leq n$  und  $x y^k z \in L_1$  für alle  $k \in \mathbb{N}$
  - Aber für  $k=0$  ist  $x y^0 z = 0^{m-j} 1^m \notin L_1$
  - Dies ist ein Widerspruch, also ist  $L_1$  nicht regulär

## ANWENDUNGEN DES PUMPING LEMMAS

- $L_1 = \{0^m 1^m \mid m \in \mathbb{N}\}$  ist nicht regulär
  - Wir nehmen an  $L_1$  sei regulär
  - Wähle  $n$  entsprechend des Pumping Lemmas und  $m > n$
  - Dann kann  $w = 0^m 1^m$  zerlegt werden in  $x=0^i$ ,  $y=0^j$   $z=0^{m-i-j}1^m$  mit  $j \neq 0$  und  $i+j \leq n$  und  $x y^k z \in L_1$  für alle  $k \in \mathbb{N}$
  - Aber für  $k=0$  ist  $x y^0 z = 0^{m-j} 1^m \notin L_1$
  - Dies ist ein Widerspruch, also ist  $L_1$  nicht regulär
- $L_2 = \{w \in \{1\}^* \mid |w| \text{ ist Primzahl}\} \notin \mathcal{L}_3$

## ANWENDUNGEN DES PUMPING LEMMAS

- $L_1 = \{0^m 1^m \mid m \in \mathbb{N}\}$  ist nicht regulär
  - Wir nehmen an  $L_1$  sei regulär
  - Wähle  $n$  entsprechend des Pumping Lemmas und  $m > n$
  - Dann kann  $w = 0^m 1^m$  zerlegt werden in  $x=0^i$ ,  $y=0^j$   $z=0^{m-i-j}1^m$  mit  $j \neq 0$  und  $i+j \leq n$  und  $x y^k z \in L_1$  für alle  $k \in \mathbb{N}$
  - Aber für  $k=0$  ist  $x y^0 z = 0^{m-j} 1^m \notin L_1$
  - Dies ist ein Widerspruch, also ist  $L_1$  nicht regulär
- $L_2 = \{w \in \{1\}^* \mid |w| \text{ ist Primzahl}\} \notin \mathcal{L}_3$ 
  - Wir nehmen an  $L_2$  sei regulär
  - Wähle  $n$  entsprechend des Pumping Lemmas und eine Primzahl  $p > n + 1$

# ANWENDUNGEN DES PUMPING LEMMAS

- $L_1 = \{0^m 1^m \mid m \in \mathbb{N}\}$  ist nicht regulär

- Wir nehmen an  $L_1$  sei regulär
- Wähle  $n$  entsprechend des Pumping Lemmas und  $m > n$
- Dann kann  $w = 0^m 1^m$  zerlegt werden in  $x = 0^i$ ,  $y = 0^j$ ,  $z = 0^{m-i-j} 1^m$  mit  $j \neq 0$  und  $i+j \leq n$  und  $x y^k z \in L_1$  für alle  $k \in \mathbb{N}$
- Aber für  $k=0$  ist  $x y^0 z = 0^{m-j} 1^m \notin L_1$
- Dies ist ein Widerspruch, also ist  $L_1$  nicht regulär

- $L_2 = \{w \in \{1\}^* \mid |w| \text{ ist Primzahl}\} \notin \mathcal{L}_3$

- Wir nehmen an  $L_2$  sei regulär
- Wähle  $n$  entsprechend des Pumping Lemmas und eine Primzahl  $p > n + 1$
- Dann kann  $w$  zerlegt werden in  $x = 1^i$ ,  $y = 1^j$ ,  $z = 1^{p-i-j}$  mit  $j \neq 0$  und  $i+j \leq n$  und  $x y^k z \in L_2$  für alle  $k \in \mathbb{N}$

# ANWENDUNGEN DES PUMPING LEMMAS

- $L_1 = \{0^m 1^m \mid m \in \mathbb{N}\}$  ist nicht regulär

- Wir nehmen an  $L_1$  sei regulär
- Wähle  $n$  entsprechend des Pumping Lemmas und  $m > n$
- Dann kann  $w = 0^m 1^m$  zerlegt werden in  $x=0^i$ ,  $y=0^j$   $z=0^{m-i-j}1^m$   
mit  $j \neq 0$  und  $i+j \leq n$  und  $x y^k z \in L_1$  für alle  $k \in \mathbb{N}$
- Aber für  $k=0$  ist  $x y^0 z = 0^{m-j} 1^m \notin L_1$
- Dies ist ein Widerspruch, also ist  $L_1$  nicht regulär

- $L_2 = \{w \in \{1\}^* \mid |w| \text{ ist Primzahl}\} \notin \mathcal{L}_3$

- Wir nehmen an  $L_2$  sei regulär
- Wähle  $n$  entsprechend des Pumping Lemmas und eine Primzahl  $p > n + 1$
- Dann kann  $w$  zerlegt werden in  $x=1^i$ ,  $y=1^j$   $z=1^{p-i-j}$   
mit  $j \neq 0$  und  $i+j \leq n$  und  $x y^k z \in L_2$  für alle  $k \in \mathbb{N}$
- Aber für  $k=p-j$  ist  $|x y^k z| = i + m(p-j) + p-i-j = (m+1)(p-j)$   
Da dies keine Primzahl ist ( $m+1 \geq 2$ ,  $p-j \geq 2$ ), ist  $x y^k z \notin L_2$
- Dies ist ein Widerspruch, also ist  $L_2$  nicht regulär

# NACHWEIS VON $L \notin \mathcal{L}_3$ MIT ABSCHLUSSEIGENSCHAFTEN

- Anwendung des Pumping Lemmas ist oft mühsam
  - Beweis für  $L_3 = \{(^m)^m \mid m \in \mathbb{N}\} \notin \mathcal{L}_3$  identisch mit dem von  $L_1$
  - Beweis für  $L_4 = \{w \in \{0, 1\}^* \mid \#_0(w) = \#_1(w)\} \notin \mathcal{L}_3$  ähnlich  
( $\#_1(w)$  ist die Anzahl der Einsen in  $w$ )

# NACHWEIS VON $L \notin \mathcal{L}_3$ MIT ABSCHLUSSEIGENSCHAFTEN

- Anwendung des Pumping Lemmas ist oft mühsam

- Beweis für  $L_3 = \{(^m)^m \mid m \in \mathbb{N}\} \notin \mathcal{L}_3$  identisch mit dem von  $L_1$
- Beweis für  $L_4 = \{w \in \{0, 1\}^* \mid \#_0(w) = \#_1(w)\} \notin \mathcal{L}_3$  ähnlich ( $\#_1(w)$  ist die Anzahl der Einsen in  $w$ )

- Verwende Umkehrung der Abschlußeigenschaften

$$\overline{L} \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L^R \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$h(L) \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$h^{-1}(L) \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \cup L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \cap L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \circ L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L' \circ L \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$\vdots$$
$$\vdots$$



# NACHWEIS VON $L \notin \mathcal{L}_3$ MIT ABSCHLUSSEIGENSCHAFTEN

- Anwendung des Pumping Lemmas ist oft mühsam

- Beweis für  $L_3 = \{(^m)^m \mid m \in \mathbb{N}\} \notin \mathcal{L}_3$  identisch mit dem von  $L_1$
- Beweis für  $L_4 = \{w \in \{0,1\}^* \mid \#_0(w) = \#_1(w)\} \notin \mathcal{L}_3$  ähnlich ( $\#_1(w)$  ist die Anzahl der Einsen in  $w$ )

- Verwende Umkehrung der Abschlußeigenschaften

$$\overline{L} \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L^R \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$h(L) \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$h^{-1}(L) \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \cup L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \cap L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \circ L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L' \circ L \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

⋮

⋮

- Anwendungsbeispiele

$L_3 \notin \mathcal{L}_3$ : Wähle Homomorphismus  $h: \{(,)\} \rightarrow \{0,1\}$  mit  $h(() = 0, h()) = 1$

Dann ist  $h(L_3) = \{0^m 1^m \mid m \in \mathbb{N}\} = L_1 \notin \mathcal{L}_3$

# NACHWEIS VON $L \notin \mathcal{L}_3$ MIT ABSCHLUSSEIGENSCHAFTEN

- Anwendung des Pumping Lemmas ist oft mühsam

- Beweis für  $L_3 = \{(^m)^m \mid m \in \mathbb{N}\} \notin \mathcal{L}_3$  identisch mit dem von  $L_1$
- Beweis für  $L_4 = \{w \in \{0, 1\}^* \mid \#_0(w) = \#_1(w)\} \notin \mathcal{L}_3$  ähnlich ( $\#_1(w)$  ist die Anzahl der Einsen in  $w$ )

- Verwende Umkehrung der Abschlußeigenschaften

$$\overline{L} \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L^R \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$h(L) \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$h^{-1}(L) \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \cup L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \cap L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \circ L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L' \circ L \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

⋮

⋮

- Anwendungsbeispiele

$L_3 \notin \mathcal{L}_3$ : Wähle Homomorphismus  $h: \{(,)\} \rightarrow \{0,1\}$  mit  $h(() = 0, h()) = 1$

Dann ist  $h(L_3) = \{0^m 1^m \mid m \in \mathbb{N}\} = L_1 \notin \mathcal{L}_3$

$L_4 \notin \mathcal{L}_3$ : Es gilt  $L_4 \cap L(0^* + 1^*) = L_1 \notin \mathcal{L}_3$

# NACHWEIS VON $L \notin \mathcal{L}_3$ MIT ABSCHLUSSEIGENSCHAFTEN

- Anwendung des Pumping Lemmas ist oft mühsam

- Beweis für  $L_3 = \{(^m)^m \mid m \in \mathbb{N}\} \notin \mathcal{L}_3$  identisch mit dem von  $L_1$
- Beweis für  $L_4 = \{w \in \{0, 1\}^* \mid \#_0(w) = \#_1(w)\} \notin \mathcal{L}_3$  ähnlich ( $\#_1(w)$  ist die Anzahl der Einsen in  $w$ )

- Verwende Umkehrung der Abschlußeigenschaften

$$\overline{L} \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L^R \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$h(L) \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$h^{-1}(L) \notin \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \cup L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \cap L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L \circ L' \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

$$L' \circ L \notin \mathcal{L}_3 \wedge L' \in \mathcal{L}_3 \Rightarrow L \notin \mathcal{L}_3$$

⋮

⋮

- Anwendungsbeispiele

$L_3 \notin \mathcal{L}_3$ : Wähle Homomorphismus  $h: \{(,)\} \rightarrow \{0,1\}$  mit  $h(( ) = 0, h( ) = 1$

Dann ist  $h(L_3) = \{0^m 1^m \mid m \in \mathbb{N}\} = L_1 \notin \mathcal{L}_3$

$L_4 \notin \mathcal{L}_3$ : Es gilt  $L_4 \cap L(0^* + 1^*) = L_1 \notin \mathcal{L}_3$

**DEAs können korrekte Klammerausdrücke nicht erkennen!**

## ● Abschlußeigenschaften

- Operationen  $\cup, \cap, \overline{\phantom{x}}, -, \overset{R}{\phantom{x}}, \circ, *, h, h^{-1}$  erhalten Regularität von Sprachen
- Verwendbar zum Nachweis von Regularität oder zur Widerlegung

## ● Abschlußeigenschaften

- Operationen  $\cup, \cap, \overline{\phantom{x}}, -, \overset{R}{\phantom{x}}, \circ, *, h, h^{-1}$  erhalten Regularität von Sprachen
- Verwendbar zum Nachweis von Regularität oder zur Widerlegung

## ● Automatische Prüfungen

- Man kann testen ob eine reguläre Sprache leer ist
- Man kann testen ob ein Wort zu einer regulären Sprache gehört
- Man kann testen ob zwei reguläre Sprachen gleich sind

## ● **Abschlußeigenschaften**

- Operationen  $\cup, \cap, \overline{\phantom{x}}, -, ^R, \circ, *, h, h^{-1}$  erhalten Regularität von Sprachen
- Verwendbar zum Nachweis von Regularität oder zur Widerlegung

## ● **Automatische Prüfungen**

- Man kann testen ob eine reguläre Sprache leer ist
- Man kann testen ob ein Wort zu einer regulären Sprache gehört
- Man kann testen ob zwei reguläre Sprachen gleich sind

## ● **Minimierung von Automaten**

- Ein Automat kann minimiert werden indem man äquivalente Zustände zusammenlegt und unerreichbare Zustände entfernt

## ● Abschlußeigenschaften

- Operationen  $\cup$ ,  $\cap$ ,  $\bar{\phantom{x}}$ ,  $-$ ,  $^R$ ,  $\circ$ ,  $*$ ,  $h$ ,  $h^{-1}$  erhalten Regularität von Sprachen
- Verwendbar zum Nachweis von Regularität oder zur Widerlegung

## ● Automatische Prüfungen

- Man kann testen ob eine reguläre Sprache leer ist
- Man kann testen ob ein Wort zu einer regulären Sprache gehört
- Man kann testen ob zwei reguläre Sprachen gleich sind

## ● Minimierung von Automaten

- Ein Automat kann minimiert werden indem man äquivalente Zustände zusammenlegt und unerreichbare Zustände entfernt

## ● Pumping Lemma

- Wiederholt man einen bestimmten Teil ausreichend großer Worte einer regulären Sprache beliebig oft, so erhält man immer ein Wort der Sprache
- Verwendbar zur Widerlegung von Regularität