# Theoretische Informatik I

## Einheit 2.5

## Eigenschaften regulärer Sprachen

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

# Wichtige Eigenschaften formaler Sprachen

- **Abschlusseigenschaften**
  - Wie können Sprachen elegant zusammengesetzt werden?
  - Erlaubt schematische Komposition von Sprachbausteinen

# Wichtige Eigenschaften formaler Sprachen

- **Abschlusseigenschaften**
  - Wie können Sprachen elegant zusammengesetzt werden?
  - Erlaubt schematische Komposition von Sprachbausteinen
- **Entscheidbarkeitsfragen**
  - Kann man bestimmte Eigenschaften automatisch testen?
  - Wortproblem (Zugehörigkeit eines Wortes zur Sprache)
  - Vergleiche zwischen Sprachen (nichtleer, Teilmenge, gleich, ...)

# Wichtige Eigenschaften formaler Sprachen

- **Abschlusseigenschaften**
  - Wie können Sprachen elegant zusammengesetzt werden?
  - Erlaubt schematische Komposition von Sprachbausteinen
- **Entscheidbarkeitsfragen**
  - Kann man bestimmte Eigenschaften automatisch testen?
  - Wortproblem (Zugehörigkeit eines Wortes zur Sprache)
  - Vergleiche zwischen Sprachen (nichtleer, Teilmenge, gleich, ...)
- **Grenzen einer Sprachklasse**
  - Wie einfach strukturiert müssen die Sprachen der Klasse sein?
  - Welche Sprachen gehören nicht zur Klasse?

# Wichtige Eigenschaften formaler Sprachen

- **Abschlusseigenschaften**
  - Wie können Sprachen elegant zusammengesetzt werden?
  - Erlaubt schematische Komposition von Sprachbausteinen
- **Entscheidbarkeitsfragen**
  - Kann man bestimmte Eigenschaften automatisch testen?
  - Wortproblem (Zugehörigkeit eines Wortes zur Sprache)
  - Vergleiche zwischen Sprachen (nichtleer, Teilmenge, gleich, ...)
- **Grenzen einer Sprachklasse**
  - Wie einfach strukturiert müssen die Sprachen der Klasse sein?
  - Welche Sprachen gehören nicht zur Klasse?

**Aus theoretischer Sicht sind das die wirklich interessanten Fragen**

# Abschlusseigenschaften, wozu?

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

# Abschlusseigenschaften, wozu?

**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

# Abschlusseigenschaften, wozu?

**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

# Abschlusseigenschaften, wozu?

**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, Schlüsselwörter, ...
  – Konstruktion liefert Automaten für alle Arten von Literalen

# Abschlusseigenschaften, präzisiert

**Zeige: $L_1, L_2$ regulär $\Rightarrow$ $L_1$ *op* $L_2$ regulär**

# Abschlusseigenschaften, präzisiert

**Zeige: $L_1, L_2$ regulär $\Rightarrow$ $L_1\ op\ L_2$ regulär**

- **Es gilt Abgeschlossenheit unter neun 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 $L_1 \circ L_2$
  - Das Bild einer regulären Sprache unter Homomorphismen ist regulär $h(L)$
  - Das Urbild ... " " ... unter Homomorphismen ist regulär $h^{-1}(L)$

# Abschlusseigenschaften, präzisiert

**Zeige: $L_1, L_2$ regulär $\Rightarrow$ $L_1$ *op* $L_2$ regulär**

- **Es gilt Abgeschlossenheit unter neun 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 $L_1 \circ L_2$
  – Das Bild einer regulären Sprache unter Homomorphismen ist regulär $h(L)$
  – Das Urbild ... " " ... unter Homomorphismen ist regulär $h^{-1}(L)$

- **Nachweis durch Verwendung aller Modelle**
  – DEA, ($\epsilon$-)NEA, reguläre Ausdrücke, Typ-3 Grammatiken
  – Modelle sind ineinander umwandelbar – **wähle das passendste**

# Abschluss unter Vereinigung, Verkettung, Hülle

## Beweisführung mit regulären Ausdrücken

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

# Abschluss unter Vereinigung, Verkettung, Hülle

## Beweisführung mit regulären Ausdrücken

- $\boldsymbol{L_1, L_2}$ **regulär** $\Rightarrow$ $\boldsymbol{L_1 \cup L_2}$ **regulär**

  $\boldsymbol{L_1, L_2}$ **regulär**

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

# Abschluss unter Vereinigung, Verkettung, Hülle

## 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**

# Abschluss unter Vereinigung, Verkettung, Hülle

## Beweisführung mit regulären Ausdrücken

- $\boldsymbol{L_1, L_2}$ **regulär** $\Rightarrow$ $\boldsymbol{L_1 \cup L_2}$ **regulär**

  $\boldsymbol{L_1, L_2}$ **regulär**

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

  $\Rightarrow$ $\boldsymbol{L_1 \cup L_2} = L(E_1) \cup L(E_2) = L(E_1 + E_2)$ **regulär**

- $\boldsymbol{L_1, L_2}$ **regulär** $\Rightarrow$ $\boldsymbol{L_1 \circ L_2}$ **regulär**

# Abschluss unter Vereinigung, Verkettung, Hülle

## 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 $L_1 = L(E_1)$, $L_2 = L(E_2)$

# Abschluss unter Vereinigung, Verkettung, Hülle

## Beweisführung mit regulären Ausdrücken

- $\boldsymbol{L_1, L_2}$ **regulär** $\Rightarrow$ $\boldsymbol{L_1 \cup L_2}$ **regulär**

  $\boldsymbol{L_1, L_2}$ **regulär**

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

  $\Rightarrow$ $\boldsymbol{L_1 \cup L_2} = L(E_1) \cup L(E_2) = L(E_1 + E_2)$ **regulär**

- $\boldsymbol{L_1, L_2}$ **regulär** $\Rightarrow$ $\boldsymbol{L_1 \circ L_2}$ **regulär**

  $\boldsymbol{L_1, L_2}$ **regulär**

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

  $\Rightarrow$ $\boldsymbol{L_1 \circ L_2} = L(E_1) \circ L(E_2) = L(E_1 \circ E_2)$ **regulär**

# Abschluss unter Vereinigung, Verkettung, Hülle

## 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**

# Abschluss unter Vereinigung, Verkettung, Hülle

## 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 $L = L(E)$

# Abschluss unter Vereinigung, Verkettung, Hülle

## 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**

# Abschluss unter Komplementbildung

## Beweisführung mit endlichen Automaten

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

  Komplementiere akzeptierende Zustände des erkennenden Automaten

# Abschluss unter Komplementbildung

## 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 $L = L(A)$

# Abschluss unter Komplementbildung

## 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 $L = L(A)$

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

# Abschluss unter Komplementbildung

## 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**

# Abschluss unter Komplementbildung

## 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**

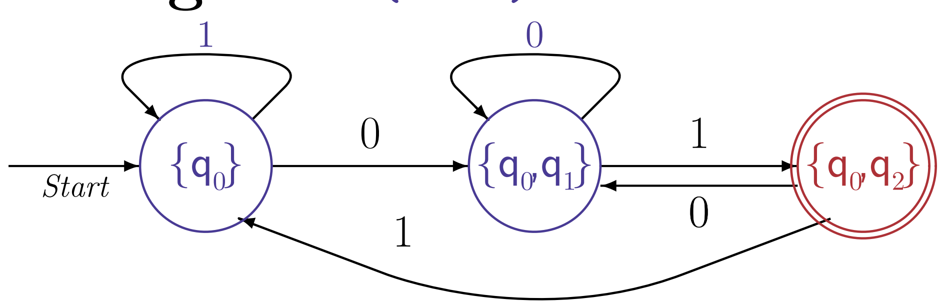
  $\Rightarrow$ Es gibt einen DEA $A = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$ mit $L = L(A)$

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

  $= L((Q,\ \Sigma,\ \delta,\ q_0,\ Q{-}F))$ **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

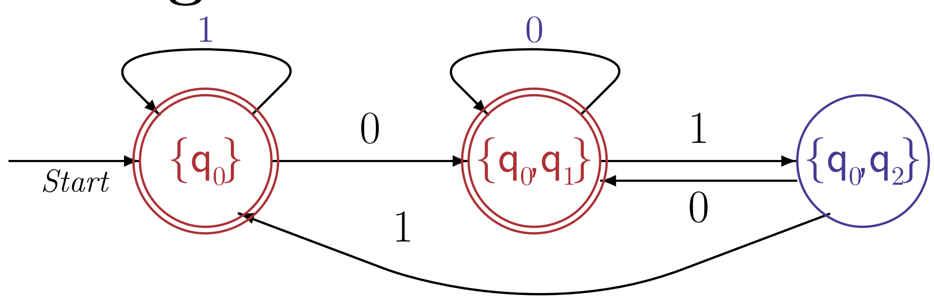
  **$L$ regulär**

  $\Rightarrow$ 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 Wörter die nicht mit 01 enden



## Beweisführung mit endlichen Automaten

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

  Komplementiere akzeptierende Zustände des erkennenden Automaten

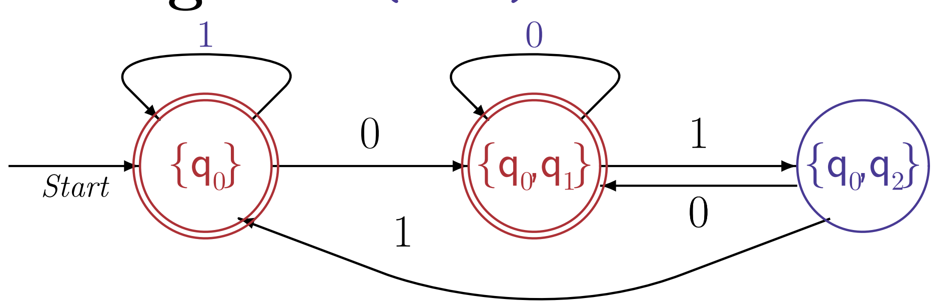
  **$L$ regulär**

  $\Rightarrow$ 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 Wörter die nicht mit 01 enden
  - Regulärer Ausdruck durch Zustandseliminationsverfahren erzeugbar

# Abschluss unter Durchschnitt und Differenz

- **Einfache mathematische Beweise**

# Abschluss unter Durchschnitt und Differenz

- **Einfache mathematische Beweise**

  **$L_1, L_2$ regulär**

# Abschluss unter Durchschnitt und Differenz

- **Einfache mathematische Beweise**

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

## Abschluss unter Durchschnitt und Differenz

- **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**

# Abschluss unter Durchschnitt und Differenz

- **Einfache mathematische Beweise**

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

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

# Abschluss unter Durchschnitt und Differenz

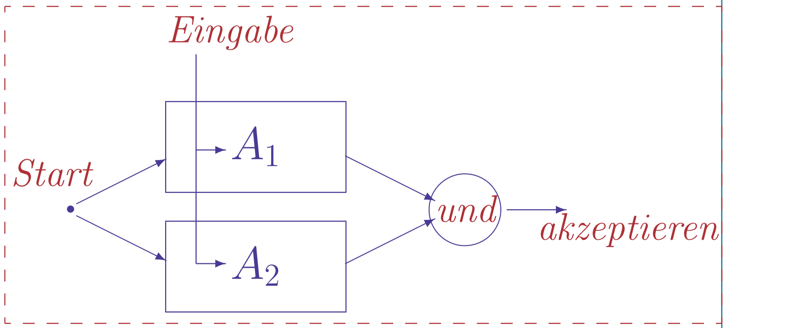
- **Einfache mathematische Beweise**

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

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

- **Produktkonstruktion auf endlichen Automaten**

  Simultane Abarbeitung von Wörtern in beiden Automaten



# Abschluss unter Durchschnitt und Differenz

- **Einfache mathematische Beweise**

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

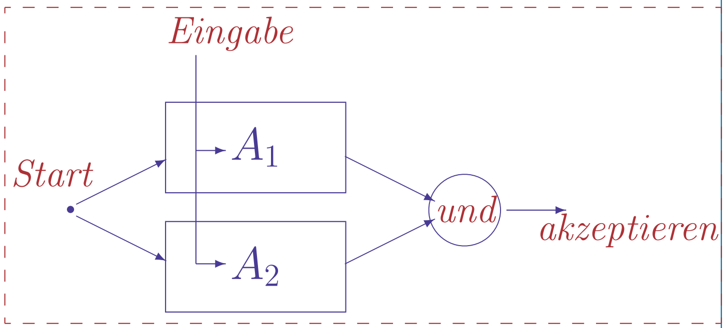
  $\boldsymbol{L_1, L_2}$ **regulär** $\;\Rightarrow\; \boldsymbol{L_1 - L_2} = L_1 \cap \overline{L_2}$ **regulär**

- **Produktkonstruktion auf endlichen Automaten**

  Simultane Abarbeitung von Wörtern in beiden Automaten

  $\boldsymbol{L_1, L_2}$ **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 $L_1 = L(A_1)$, $L_2 = L(A_2)$



# Abschluss unter Durchschnitt und Differenz

- **Einfache mathematische Beweise**

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

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

- **Produktkonstruktion auf endlichen Automaten**

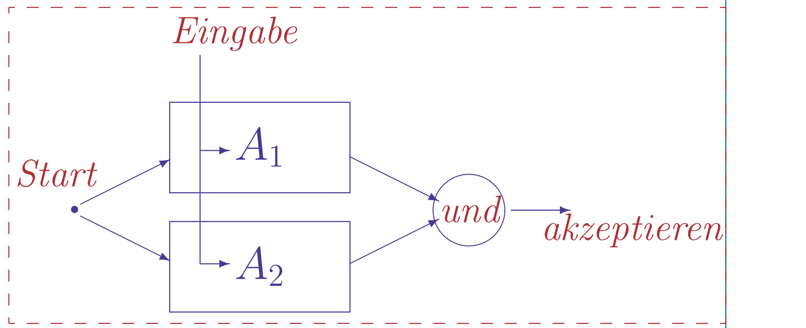
  Simultane Abarbeitung von Wörtern in beiden Automaten

  $\boldsymbol{L_1, L_2}$ **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 $L_1 = L(A_1)$, $L_2 = L(A_2)$




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

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

# Abschluss unter Durchschnitt und Differenz

- **Einfache mathematische Beweise**

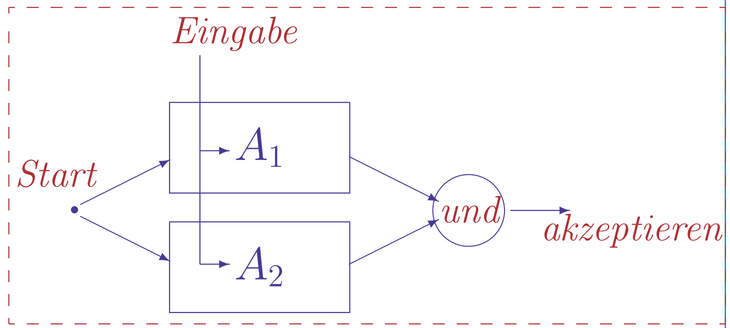
  $L_1, L_2$ **regulär** $\Rightarrow$ $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ **regulär**

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

- **Produktkonstruktion auf endlichen Automaten**

  Simultane Abarbeitung von Wörtern 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 $p \in Q_1$, $q \in Q_2$, $a \in \Sigma$

# Abschluss unter Durchschnitt und Differenz

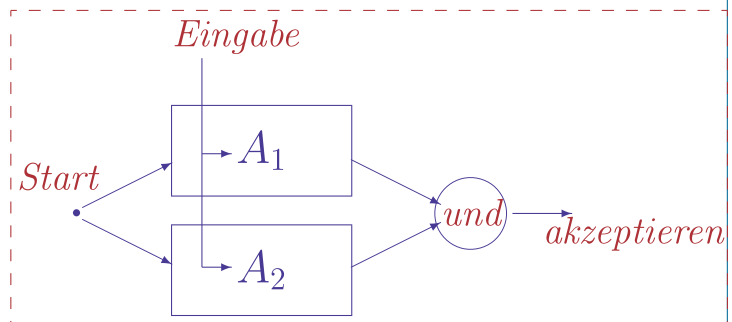
- **Einfache mathematische Beweise**

  $L_1, L_2$ **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 Wörtern 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 $L_1 = L(A_1)$, $L_2 = L(A_2)$

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

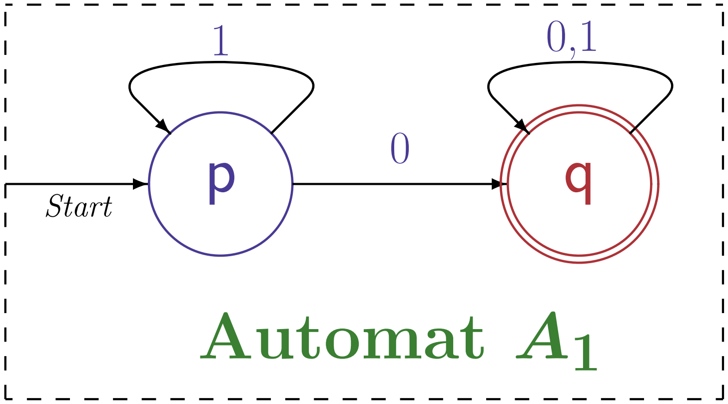
  $$= \{w \in \Sigma^* \mid (\hat{\delta}_1(q_{0,1}, w), \hat{\delta}_2(q_{0,2}, w)) \in F_1 \times F_2\}$$

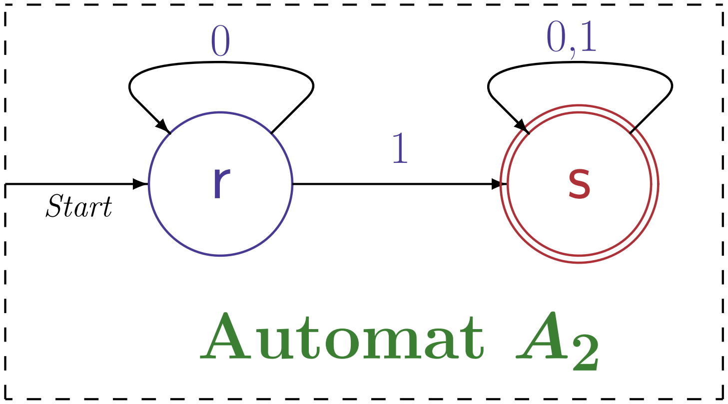
  Konstruiere $A = (Q_1 \times Q_2, \Sigma, \delta, (q_{0,1}, q_{0,2}), F_1 \times F_2)$

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

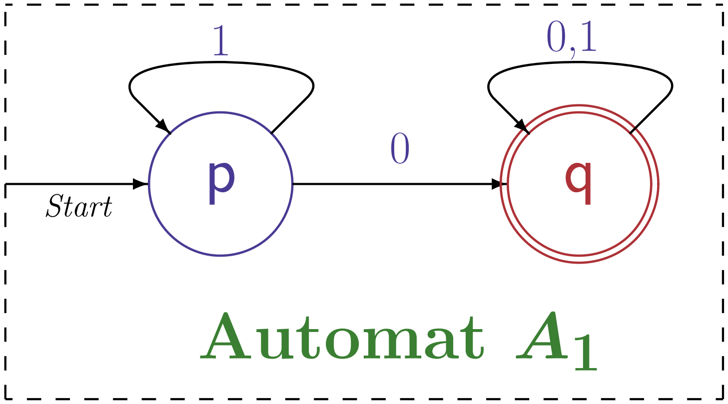
  $\Rightarrow$ $L_1 \cap L_2 = L(A)$ **regulär**

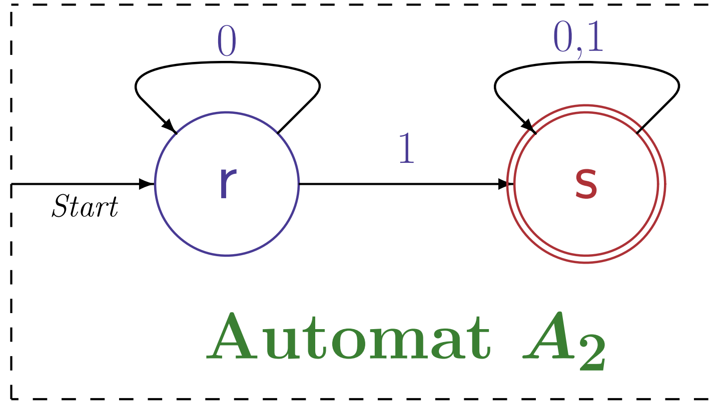
# Produktkonstruktion am Beispiel

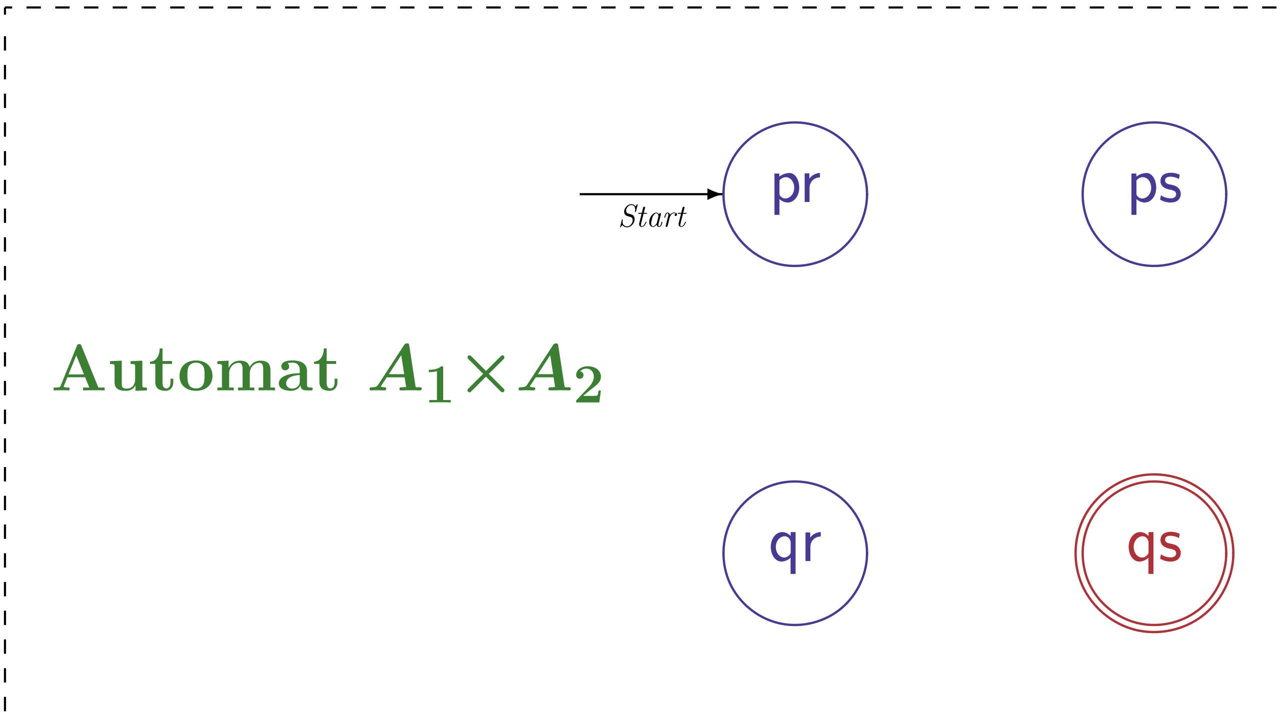


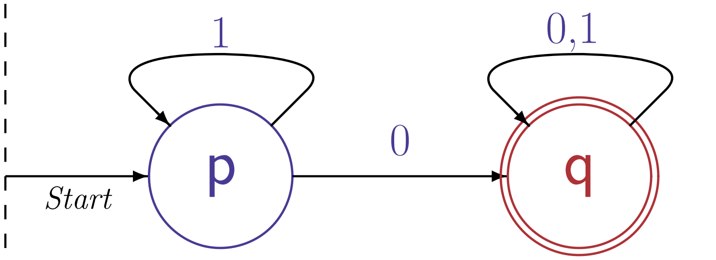
# Produktkonstruktion am Beispiel



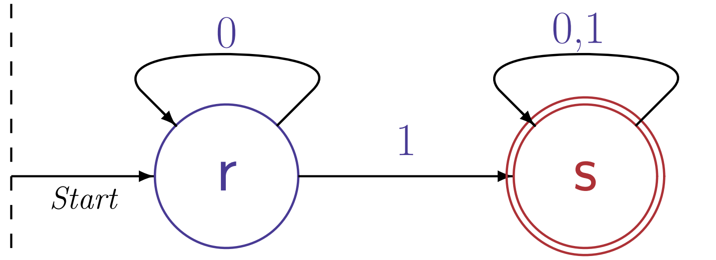



# Produktkonstruktion am Beispiel

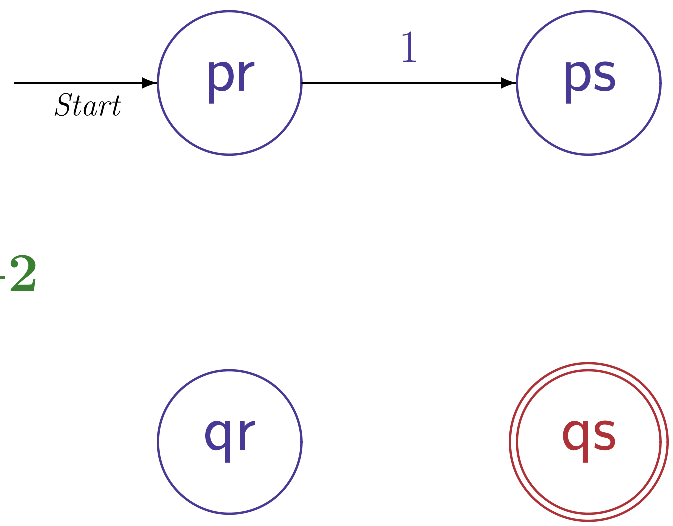



**Automat $A_1$**

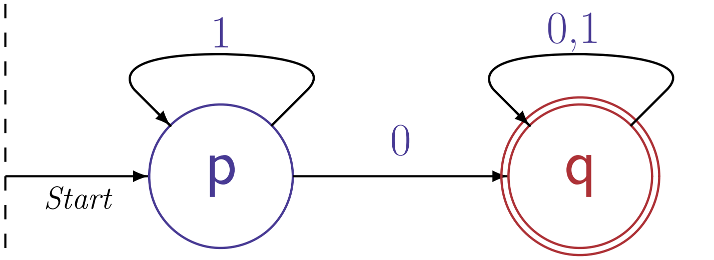



**Automat $A_2$**

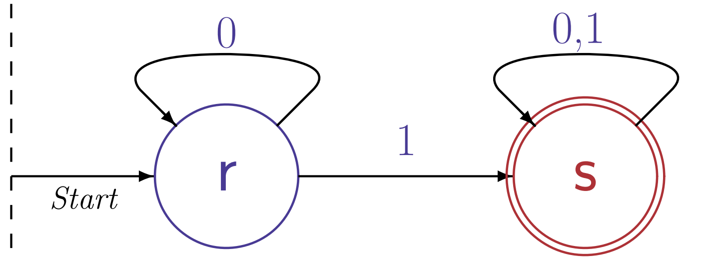
**Automat $A_1 \times A_2$**



# Produktkonstruktion am Beispiel

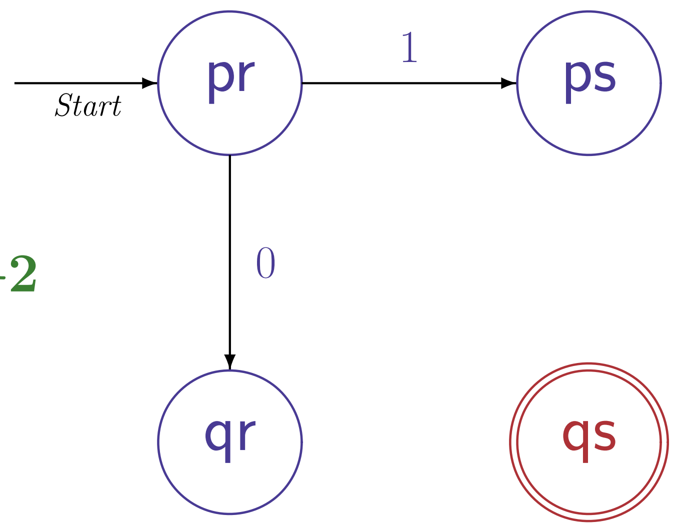



**Automat $A_1$**

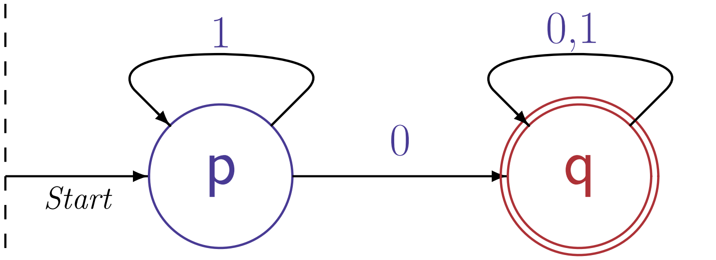



**Automat $A_2$**

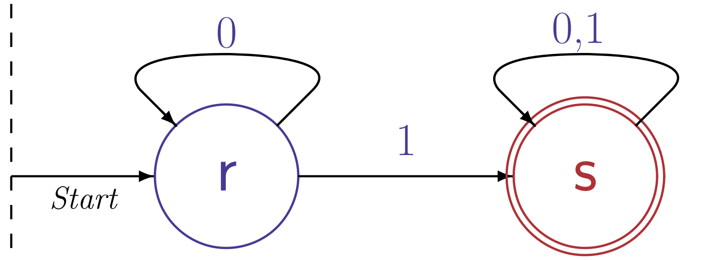
**Automat $A_1 \times A_2$**



# Produktkonstruktion am Beispiel

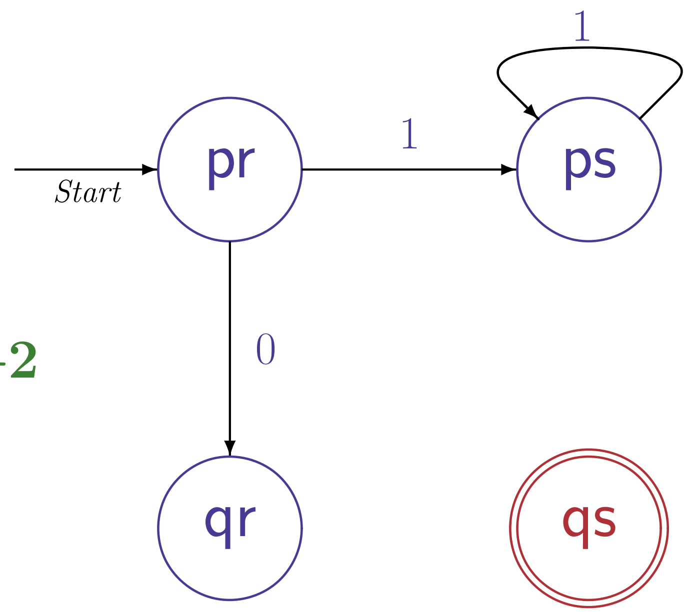



**Automat $A_1$**

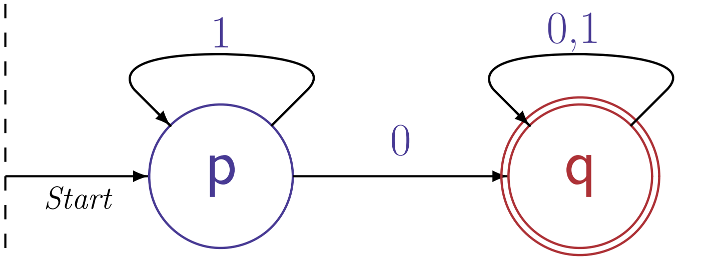



**Automat $A_2$**

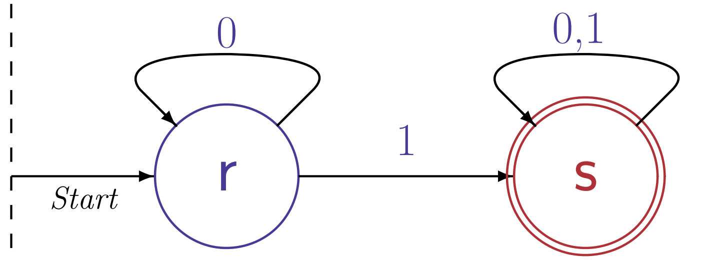
**Automat $A_1 \times A_2$**



# Produktkonstruktion am Beispiel

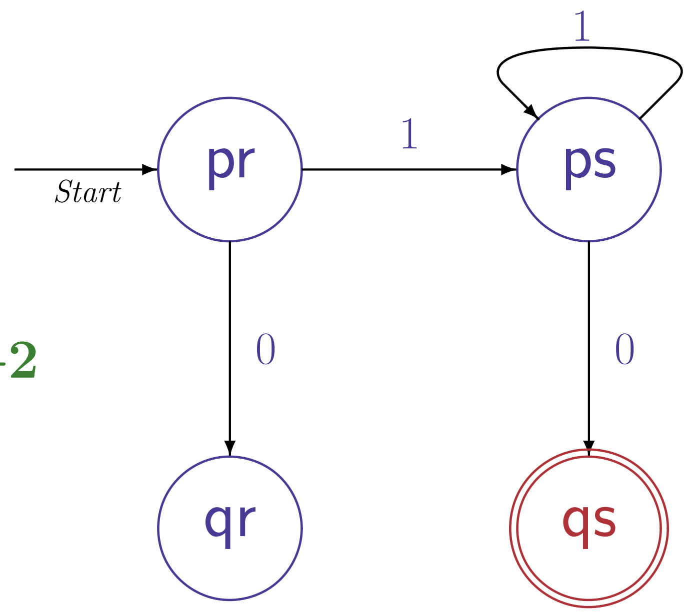



**Automat $A_1$**

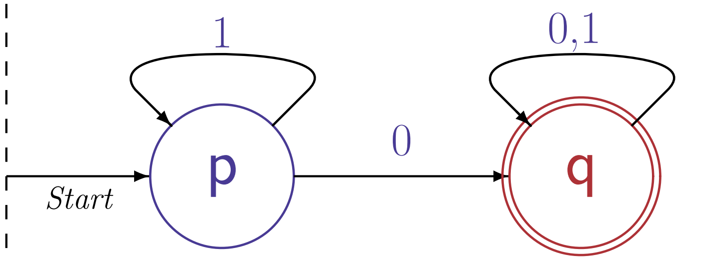



**Automat $A_2$**

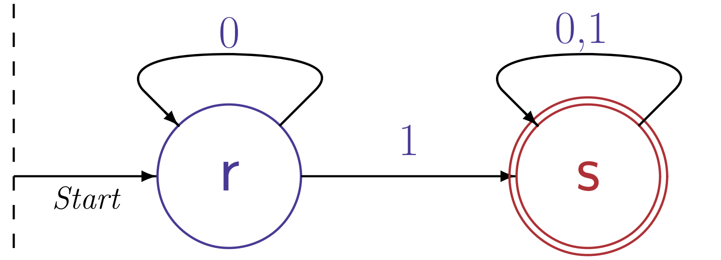
**Automat $A_1 \times A_2$**



# Produktkonstruktion am Beispiel

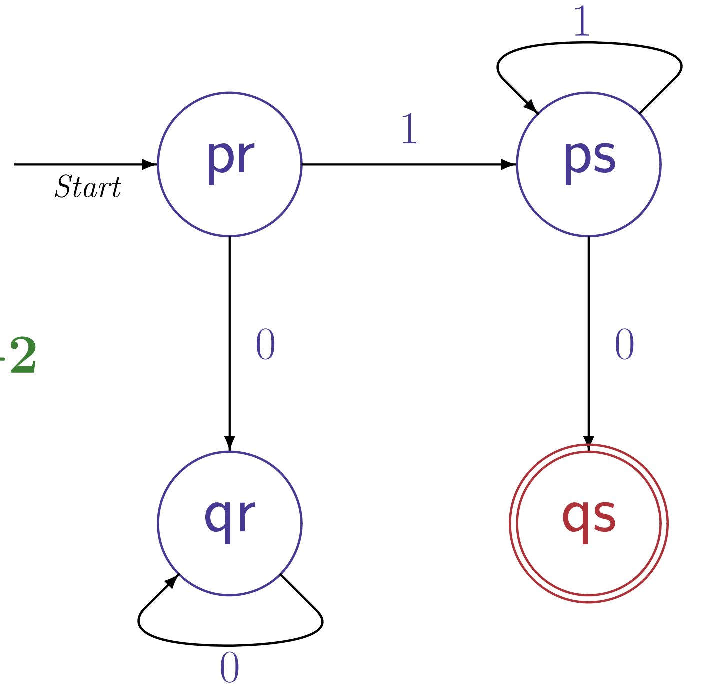



**Automat $A_1$**

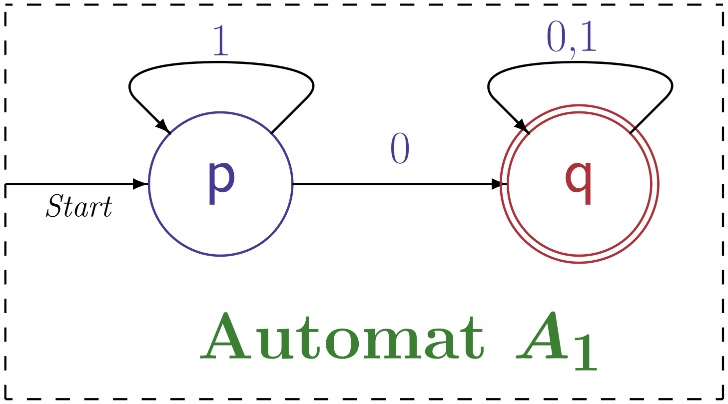


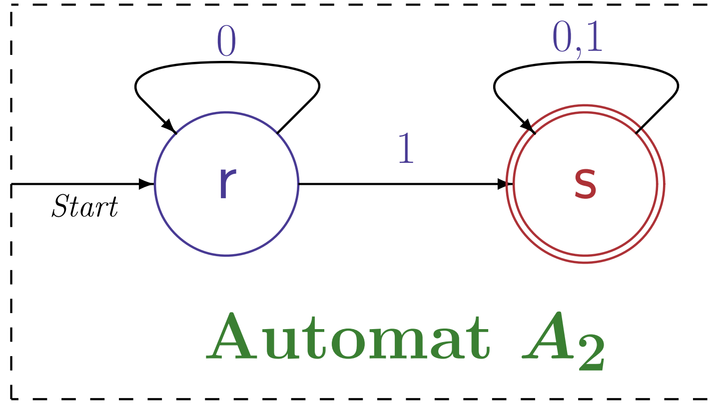

**Automat $A_2$**

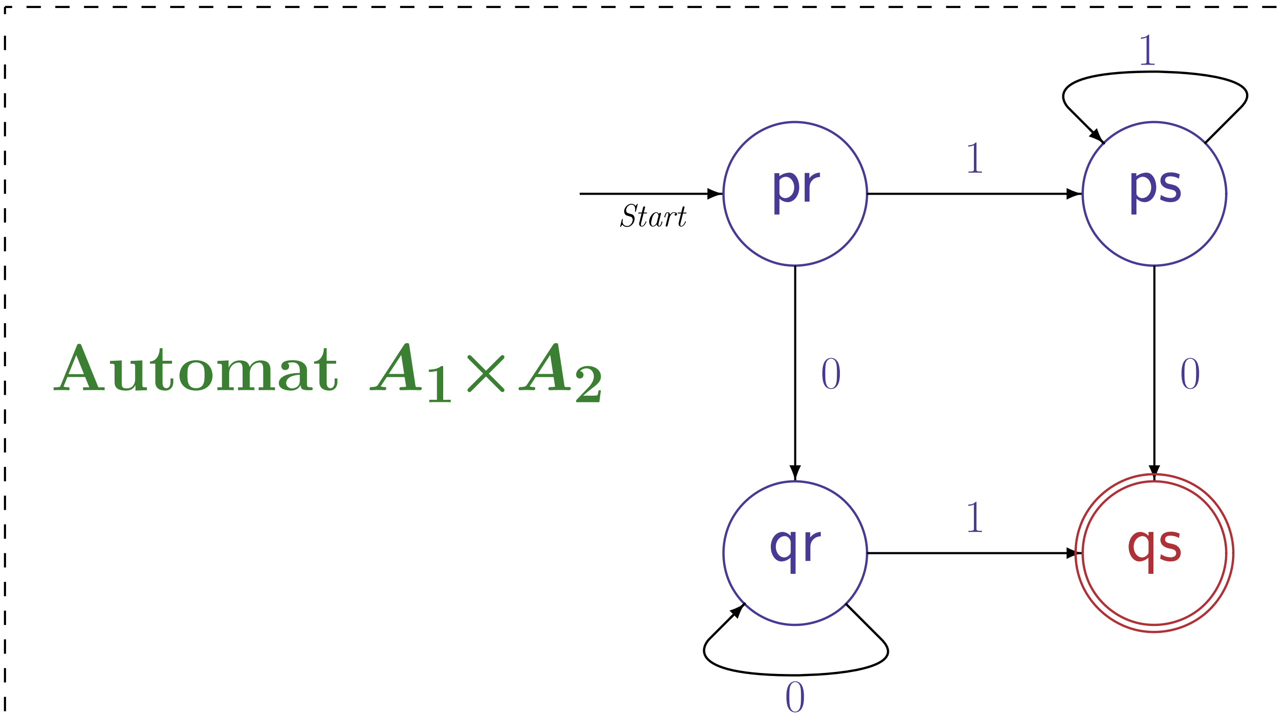
**Automat $A_1 \times A_2$**



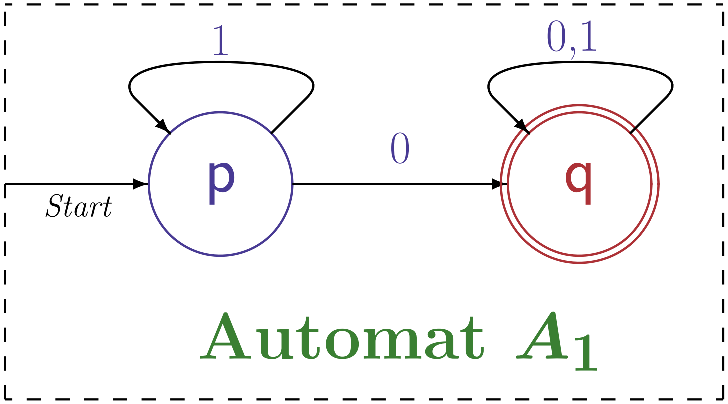
# Produktkonstruktion am Beispiel

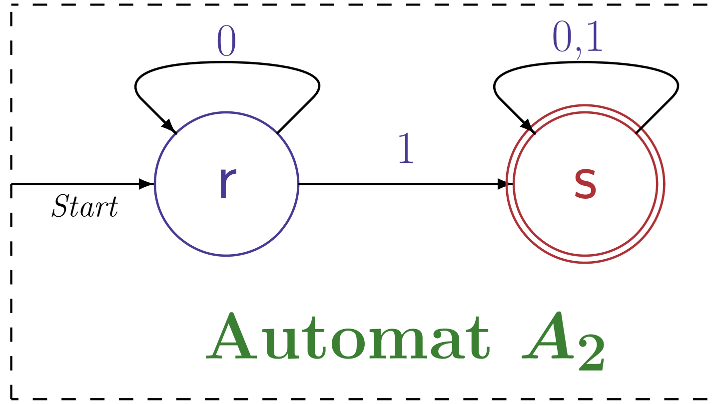


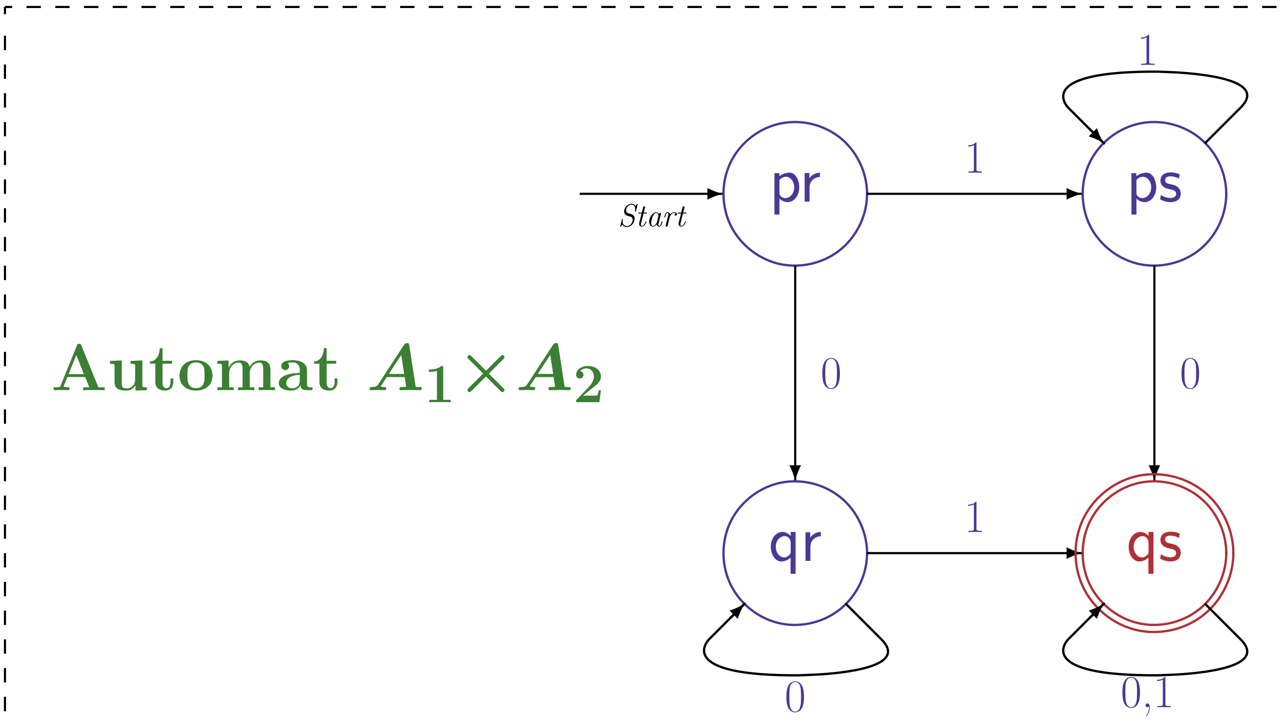



# Produktkonstruktion am Beispiel



# Abschluss unter Spiegelung

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

# Abschluss unter Spiegelung

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

- **Beweisführung mit Automaten**
  – Bilde Umkehrautomaten zu $A = (Q, \Sigma, \delta, q_0, F)$ mit $L{=}L(A)$

# Abschluss unter Spiegelung

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

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

# Abschluss unter Spiegelung

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

- **Beweisführung mit Automaten**
  – Bilde Umkehrautomaten zu $A = (Q, \Sigma, \delta, q_0, F)$ mit $L{=}L(A)$
    - Umkehrung der Pfeile im Diagramm: $\delta^R(q, a) = \{q' | \delta(q', a) = q\}$
    - $q_0$ 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

# Abschluss unter Spiegelung

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

- **Beweisführung mit Automaten**
  - Bilde Umkehrautomaten zu $A = (Q, \Sigma, \delta, q_0, F)$ mit $L{=}L(A)$
    - Umkehrung der Pfeile im Diagramm: $\delta^R(q,a) = \{q' | \delta(q',a) = q\}$
    - $q_0$ 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 $E \in \{\emptyset, \epsilon, \mathtt{a}\}$ ist $L^R = L = L(E)$ regulär

# Abschluss unter Spiegelung

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

- **Beweisführung mit Automaten**
  - Bilde Umkehrautomaten zu $A = (Q, \Sigma, \delta, q_0, F)$ mit $L=L(A)$
    - Umkehrung der Pfeile im Diagramm: $\delta^R(q,a) = \{q' | \delta(q',a) = q\}$
    - $q_0$ 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 $E \in \{\emptyset, \epsilon, \mathtt{a}\}$ ist $L^R = L = L(E)$ regulär
  - Für $E = E_1\mathtt{+}E_2$ ist $L^R = (L(E_1) \cup L(E_2))^R = L(E_1)^R \cup L(E_2)^R$ regulär

# Abschluss unter Spiegelung

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

- **Beweisführung mit Automaten**
  - Bilde Umkehrautomaten zu $A = (Q, \Sigma, \delta, q_0, F)$ mit $L{=}L(A)$
    - Umkehrung der Pfeile im Diagramm: $\delta^R(q, a) = \{q' | \delta(q', a) = q\}$
    - $q_0$ 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 $E \in \{\emptyset, \epsilon, \mathtt{a}\}$ 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

# Abschluss unter Spiegelung

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

- **Beweisführung mit Automaten**
  - Bilde Umkehrautomaten zu $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$
    - Umkehrung der Pfeile im Diagramm: $\delta^R(q, a) = \{q' | \delta(q', a) = q\}$
    - $q_0$ 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 $E \in \{\emptyset, \epsilon, \mathtt{a}\}$ 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

# Abschluss unter Spiegelung

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

- **Beweisführung mit Automaten**
  - Bilde Umkehrautomaten zu $A = (Q, \Sigma, \delta, q_0, F)$ mit $L{=}L(A)$
    - Umkehrung der Pfeile im Diagramm: $\delta^R(q,a) = \{q' | \delta(q',a) = q\}$
    - $q_0$ 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 $E \in \{\emptyset, \epsilon, \mathtt{a}\}$ ist $L^R = L = L(E)$ regulär
  - Für $E = E_1\mathtt{+}E_2$ 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 $L(\mathtt{(0+1)0^*})$**

# 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 $L = L(A)$
    - Umkehrung der Pfeile im Diagramm: $\delta^R(q, a) = \{q' | \delta(q', a) = q\}$
    - $q_0$ 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 $E \in \{\emptyset, \epsilon, \mathtt{a}\}$ 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 $L(\mathtt{(0+1)0^*})$**
  - $L^R = L(\mathtt{(0^*)}^R\mathtt{(0+1)}^R) = L(\mathtt{(0}^R\mathtt{)}^*\mathtt{(0}^R\mathtt{+1}^R\mathtt{)}) = L(\mathtt{0^*(0+1)})$

# 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*

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

# Abschluss unter Homomorphismen

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

$h{:}\Sigma^*{\to}\Sigma'^*$ ist **Homomorphismus**, wenn $h(v_1..v_n) = h(v_1)..h(v_n)$

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

$\boldsymbol{h(L)}{=}\{h(w)\,|\,w\in L\} \subseteq \Sigma'^*$ ist das Abbild der Wörter von $L$ unter $h$

- **Beweis mit Grammatiken**

  **$L$ regulär**

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

# Abschluss unter Homomorphismen

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

$h{:}\Sigma^*\to\Sigma'^*$ ist **Homomorphismus**, wenn $h(v_1..v_n) = h(v_1)..h(v_n)$

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

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

- **Beweis mit Grammatiken**

  **$L$ regulär**

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

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

# 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*

$\boldsymbol{h(L)}{=}\{h(w) \mid w \in L\} \subseteq \Sigma'^*$ ist das Abbild der Wörter 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 $A{\rightarrow}a\,B \in P$ erzeuge Regeln $A{\rightarrow}a_1B_1,\ B_1{\rightarrow}a_2B_2,...,B_{k-1}{\rightarrow}a_kB$, wobei $h(a) = a_1..a_k$ und alle $B_i$ neue Hilfsvariablen

# Abschluss unter Homomorphismen

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

$h{:}\Sigma^*{\to}\Sigma'^*$ ist **Homomorphismus**, wenn $h(v_1..v_n) = h(v_1)..h(v_n)$

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

$\boldsymbol{h(L)}{=}\{h(w)\,|\,w\in L\} \subseteq \Sigma'^*$ ist das Abbild der Wörter 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 $A{\to}a\,B \in P$ erzeuge Regeln $A{\to}a_1B_1,\ B_1{\to}a_2B_2,...,B_{k-1}{\to}a_kB$, wobei $h(a) = a_1..a_k$ und alle $B_i$ neue Hilfsvariablen

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

# Abschluss unter Homomorphismen

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

$h{:}\Sigma^*{\to}\Sigma'^*$ ist **Homomorphismus**, wenn $h(v_1..v_n) = h(v_1)..h(v_n)$

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

$\boldsymbol{h(L)}{=}\{h(w)\,|\,w\in L\} \subseteq \Sigma'^*$ ist das Abbild der Wörter 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 $A{\to}a\,B \in P$ erzeuge Regeln $A{\to}a_1B_1,\ B_1{\to}a_2B_2,...,B_{k-1}{\to}a_kB$, wobei $h(a) = a_1..a_k$ 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 $A{\to}a\,B \in P \Leftrightarrow A \xrightarrow{*}_{G_h} h(a)\,B$

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

# Abschluss unter Homomorphismen

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

$h{:}\Sigma^*\longrightarrow\Sigma'^*$ ist **Homomorphismus**, wenn $h(v_1..v_n) = h(v_1)..h(v_n)$

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

$\boldsymbol{h(L)}=\{h(w)\mid w\in L\}\subseteq\Sigma'^*$ ist das Abbild der Wörter 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 $A{\longrightarrow}a\,B\in P$ erzeuge Regeln $A{\longrightarrow}a_1B_1$, $B_1{\longrightarrow}a_2B_2$,...,$B_{k-1}{\longrightarrow}a_kB$, wobei $h(a)=a_1..a_k$ 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 $A{\longrightarrow}a\,B\in P\;\Leftrightarrow\;A\xrightarrow{*}_{G_h}h(a)\,B$

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

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

# Abschluss unter Homomorphismen

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

$h{:}\Sigma^*{\to}\Sigma'^*$ ist **Homomorphismus**, wenn $h(v_1..v_n) = h(v_1)..h(v_n)$

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

$\boldsymbol{h(L)}{=}\{h(w)\,|\,w\in L\} \subseteq \Sigma'^*$ ist das Abbild der Wörter 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 $A{\to}a\,B \in P$ erzeuge Regeln $A{\to}a_1B_1,\ B_1{\to}a_2B_2,...,B_{k-1}{\to}a_kB$, wobei $h(a) = a_1..a_k$ 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 $A{\to}a\,B \in P \;\Leftrightarrow\; A \xrightarrow{*}_{G_h} h(a)\,B$

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

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

---

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

# Abschluss unter inversen Homomorphismen

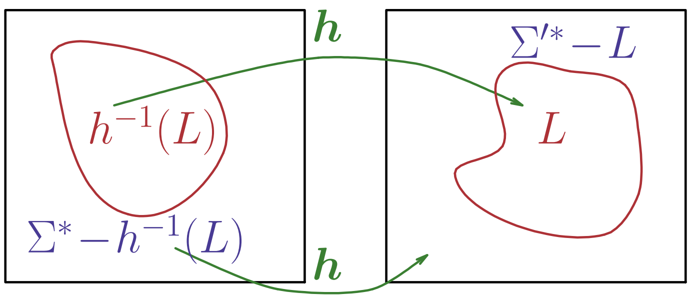
$$\boldsymbol{L} \textbf{ regulär, } \boldsymbol{h} \textbf{ Homomorphismus } \Rightarrow \boldsymbol{h^{-1}(L)} \textbf{ regulär}$$

# Abschluss unter inversen Homomorphismen

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

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

Urbild der Wörter von $L$ unter $h$



# Abschluss unter inversen Homomorphismen

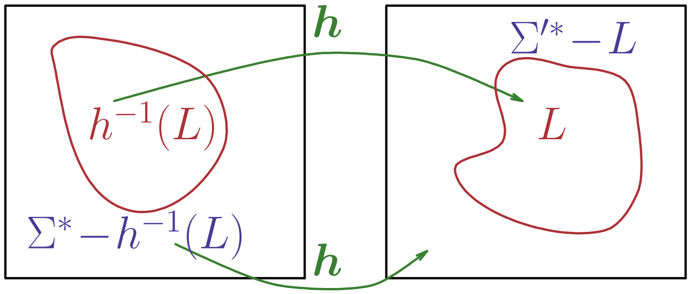
**$L$ regulär, $h$ Homomorphismus $\Rightarrow$ $h^{-1}(L)$ regulär**

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

Urbild der Wörter von $L$ unter $h$

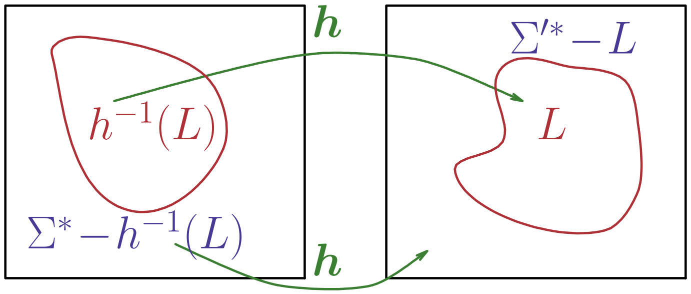
– z.B. Für $L = L(\texttt{(01+10)}^*)$,

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



## 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

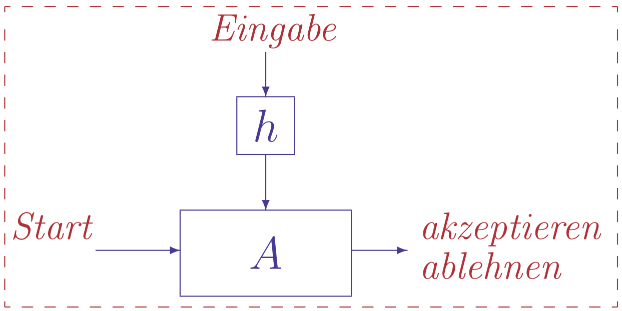
Urbild der Wörter von $L$ unter $h$

– z.B. Für $L = L(\texttt{(01+10)}^*)$,

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

- **Beweis mit endlichen Automaten**

  Berechnung von $h$ vor Abarbeitung der Wörter im Automaten



# Abschluss unter inversen Homomorphismen

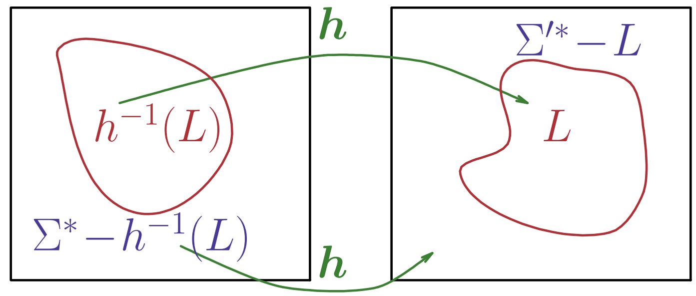
**$L$ regulär, $h$ Homomorphismus $\Rightarrow$ $h^{-1}(L)$ regulär**

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

Urbild der Wörter von $L$ unter $h$

– z.B. Für $L = L(\texttt{(01+10)}^*)$,

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



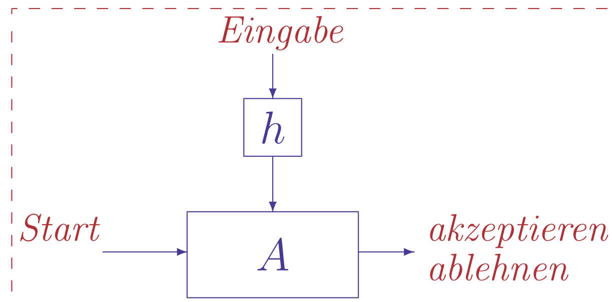

- **Beweis mit endlichen Automaten**

  Berechnung von $h$ vor Abarbeitung der Wörter im Automaten

  **$L$ regulär**

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

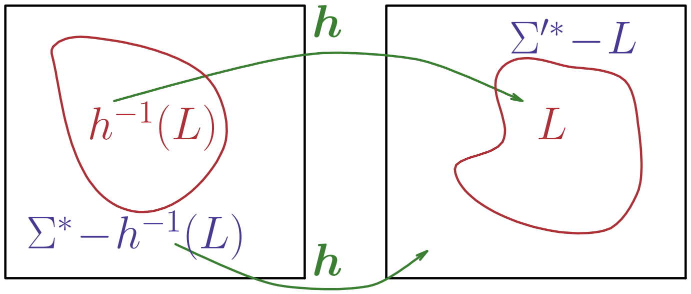
  mit $L = L(A) = \{w \in \Sigma'^* \mid \hat{\delta}(q_0, w) \in F\}$



# 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 Wörter von $L$ unter $h$
– z.B. Für $L = L(\texttt{(01+10)}^*)$,
$h(a) = 01$, $h(b) = 10$ ist $h^{-1}(L) = L(\texttt{(a+b)}^*)$



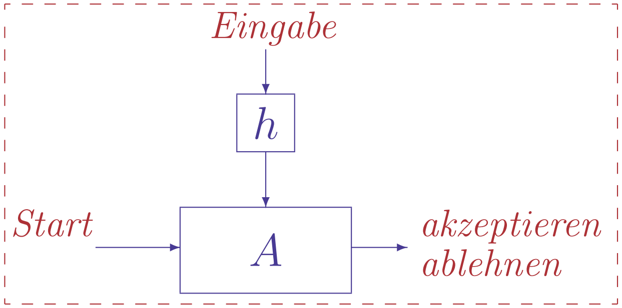

- **Beweis mit endlichen Automaten**

Berechnung von $h$ vor Abarbeitung der Wörter 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

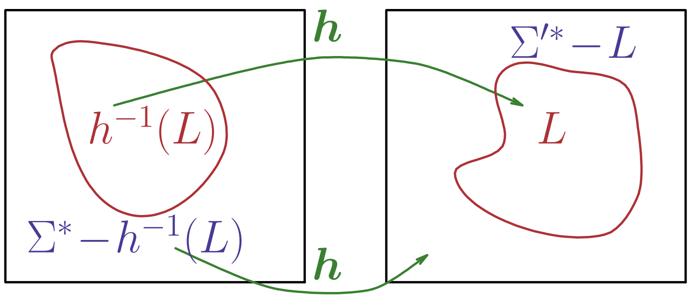
**$L$ regulär, $h$ Homomorphismus $\Rightarrow$ $h^{-1}(L)$ regulär**

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

Urbild der Wörter von $L$ unter $h$

– z.B. Für $L = L(\texttt{(01+10)}^*)$,

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




- **Beweis mit endlichen Automaten**

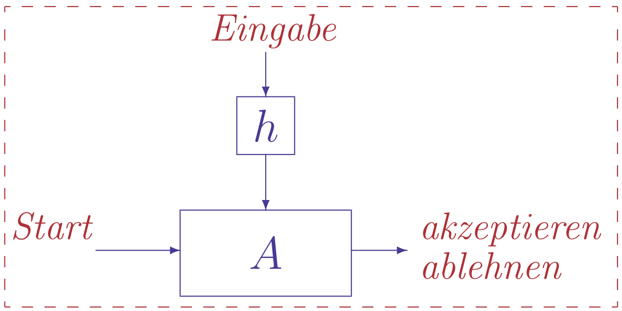
Berechnung von $h$ vor Abarbeitung der Wörter im Automaten

**$L$ 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

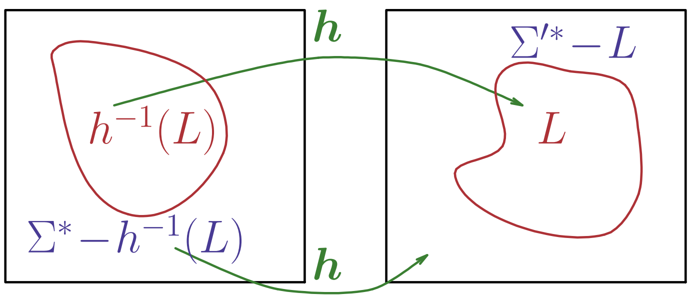
## $L$ regulär, $h$ Homomorphismus $\Rightarrow$ $h^{-1}(L)$ regulär

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

Urbild der Wörter von $L$ unter $h$

– z.B. Für $L = L(\texttt{(01+10)}^*)$,

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




- **Beweis mit endlichen Automaten**

  Berechnung von $h$ vor Abarbeitung der Wörter im Automaten

  **$L$ regulär**

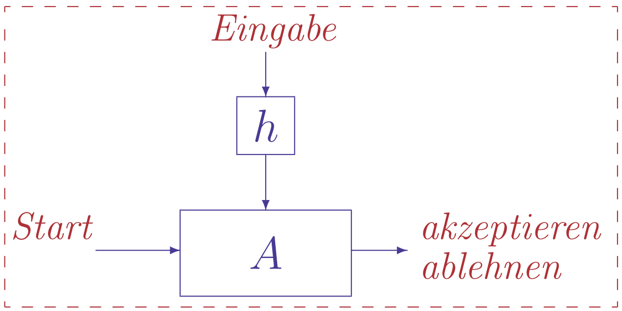
  $\Rightarrow$ Es gibt einen DEA $A = (Q, \Sigma', \delta, q_0, F)$

  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

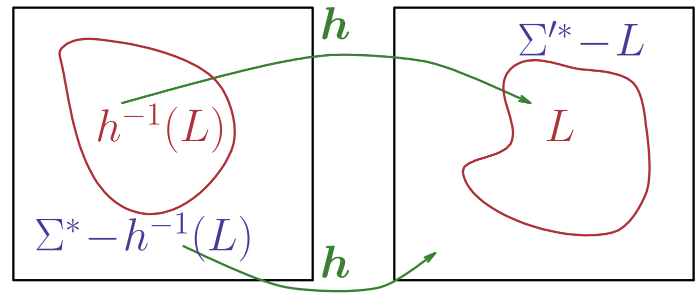
**$L$ regulär, $h$ Homomorphismus $\Rightarrow$ $h^{-1}(L)$ regulär**

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

Urbild der Wörter von $L$ unter $h$

– z.B. Für $L = L(\texttt{(01+10)}^*)$,

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




- **Beweis mit endlichen Automaten**

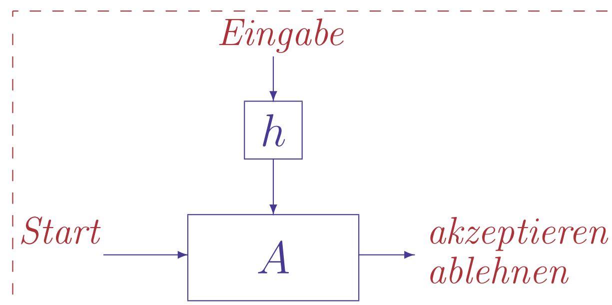
Berechnung von $h$ vor Abarbeitung der Wörter im Automaten

**$L$ 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 $w \in \Sigma^*$

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

# Tests für Eigenschaften regulärer Sprachen

- **Welche Eigenschaften sind automatisch prüfbar?**

# Tests für Eigenschaften regulärer Sprachen

- **Welche Eigenschaften sind automatisch prüfbar?**
  - Ist die Sprache eines Automaten leer?

# Tests für Eigenschaften regulärer Sprachen

- **Welche Eigenschaften sind automatisch prüfbar?**
  - Ist die Sprache eines Automaten leer?
  - Zugehörigkeit: Ist ein Wort $w$ Element der Sprache eines Automaten?

# Tests für Eigenschaften regulärer Sprachen

- **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?

# Tests für Eigenschaften regulärer Sprachen

- **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

# Tests für Eigenschaften regulärer Sprachen

- **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)

# Tests für Eigenschaften regulärer Sprachen

- **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

# Tests für Eigenschaften regulärer Sprachen

- **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

# Tests für Eigenschaften regulärer Sprachen

- **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**

# 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?

# 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**

# 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**
  - $L(\emptyset)=\emptyset$, $L(\epsilon)\neq\emptyset$, $L(\mathtt{a})\neq\emptyset$
  - $L((E))=\emptyset \Leftrightarrow L(E)=\emptyset$ — 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 $L(E^*)$

## Test auf Zugehörigkeit

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

# Test auf Zugehörigkeit

- **Unterschiedlich schwierig je nach Repräsentation**
  - Automaten: Gibt es einen akzeptierenden Pfad für das Wort $w$?
  - Reguläre Ausdrücke: Wird $w$ 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 das Wort $w$?
  - Reguläre Ausdrücke: Wird $w$ 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 Wörter "äquivalent" sind
    - Es führen exakt dieselben Wörter zu akzeptierenden Zuständen
  - Ergibt **minimalen äquivalenten Automaten**

# Äquivalenztest für Zustände

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

# Äquivalenztest für Zustände

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

# Äquivalenztest für Zustände

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

# Äquivalenztest für Zustände

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

  Markiere Unterscheidbarkeit von Zuständen in Tabelle

# Äquivalenztest für Zustände

- **Äquivalenz der Zustände $p$ und $q$ ($p \cong q$)**
  - Für alle Wörter $w \in \Sigma^*$ gilt $\hat{\delta}(p, w) \in F \Leftrightarrow \hat{\delta}(q, w) \in F$
  - Die Wörter müssen nicht zum gleichen Zustand führen
- **Positives Prüfverfahren schwierig**
  - Man muss alle Wörter überprüfen, die von einem Zustand ausgehen
  - Man kann sich auf Wörter 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 $q \notin F$

# Äquivalenztest für Zustände

- **Äquivalenz der Zustände $p$ und $q$ ($p \cong q$)**
  - Für alle Wörter $w \in \Sigma^*$ gilt $\hat{\delta}(p,w) \in F \Leftrightarrow \hat{\delta}(q,w) \in F$
  - Die Wörter müssen nicht zum gleichen Zustand führen
- **Positives Prüfverfahren schwierig**
  - Man muss alle Wörter überprüfen, die von einem Zustand ausgehen
  - Man kann sich auf Wörter 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

# Äquivalenztest für Zustände

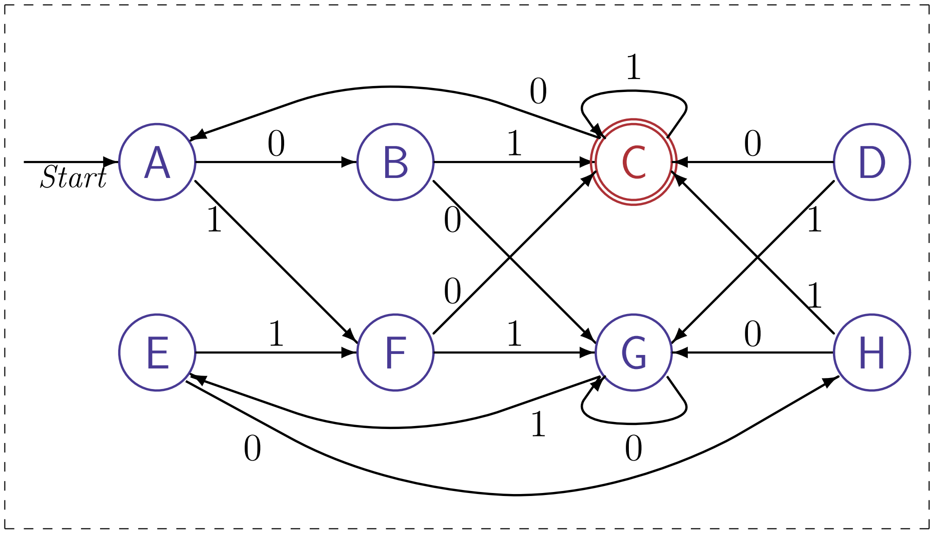
- **Äquivalenz der Zustände $p$ und $q$ ($p \cong q$)**
  - Für alle Wörter $w \in \Sigma^*$ gilt $\hat{\delta}(p, w) \in F \Leftrightarrow \hat{\delta}(q, w) \in F$
  - Die Wörter müssen nicht zum gleichen Zustand führen
- **Positives Prüfverfahren schwierig**
  - Man muss alle Wörter überprüfen, die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

  Markiere Unterscheidbarkeit von Zuständen in Tabelle
  - Start: $p \ncong q$, falls $p \in F$ und $q \notin F$
  - Iteration: $p \ncong q$, falls $\delta(p, a) \ncong \delta(q, a)$ für ein $a \in \Sigma$

  In jeder Iteration werden nur noch ungeklärte Paare überprüft

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

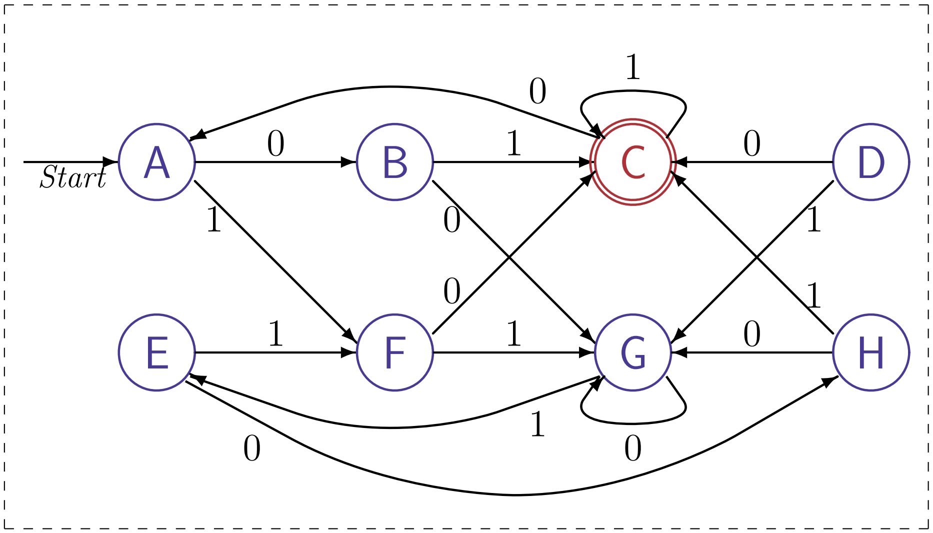
# Äquivalenztest am Beispiel




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

*Tabelle der Unterschiede*

# Äquivalenztest am Beispiel

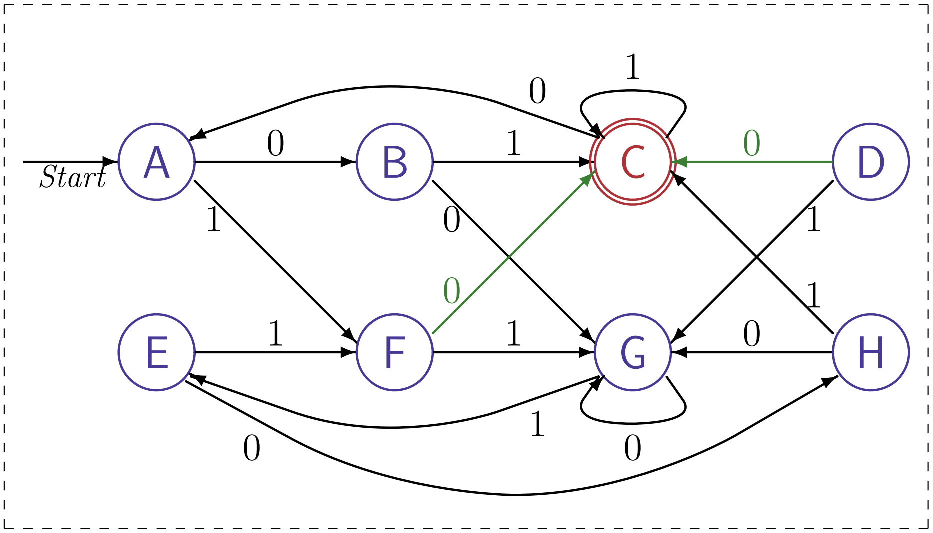



| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| A | \ | | × | | | | | |
| B | | \ | × | | | | | |
| C | × | × | \ | × | × | × | × | × |
| D | | | × | \ | | | | |
| E | | | × | | \ | | | |
| F | | | × | | | \ | | |
| G | | | × | | | | \ | |
| H | | | × | | | | | \ |

*Tabelle der Unterschiede*

**1.** Unterscheide akzeptierende Zustände (C) von allen anderen

# Äquivalenztest am Beispiel



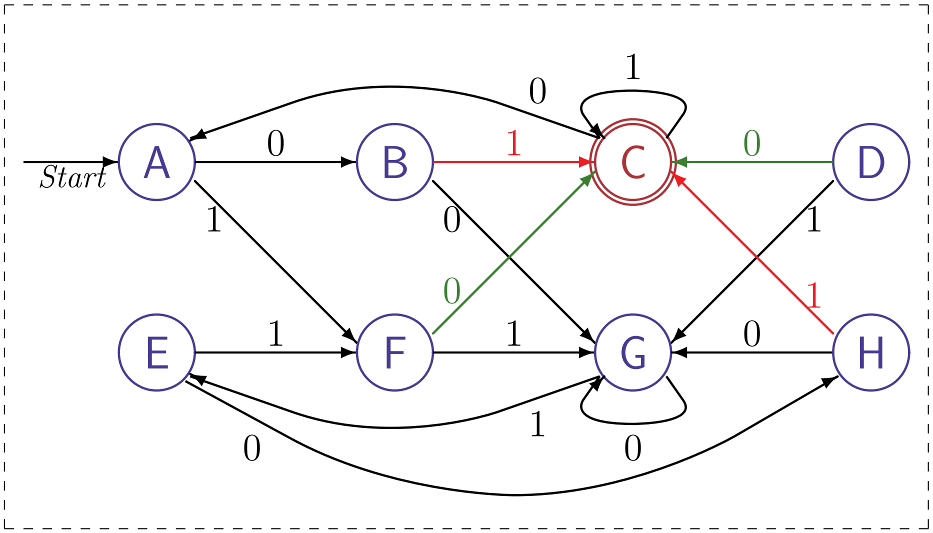

| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| A | \ | | × | × | | × | | |
| B | | \ | × | × | | × | | |
| C | × | × | \ | × | × | × | × | × |
| D | × | × | × | \ | × | | × | × |
| E | | | × | × | \ | × | | |
| F | × | × | × | | × | \ | × | × |
| G | | | × | × | | × | \ |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

*Tabelle der Unterschiede*

**1.** Unterscheide akzeptierende Zustände (C) von allen 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 | × | | × | × | × | × | × | \ |

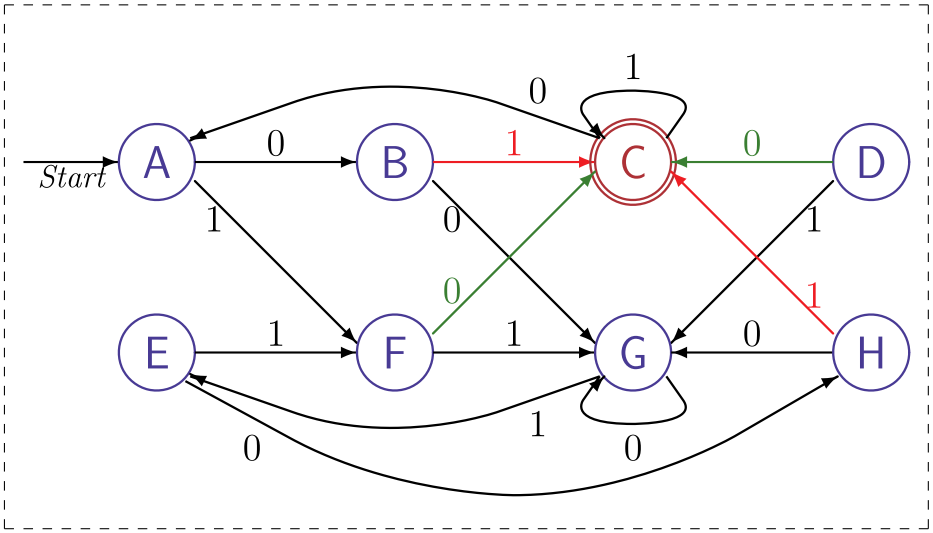
*Tabelle der Unterschiede*

**1.** Unterscheide akzeptierende Zustände (C) von allen 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*

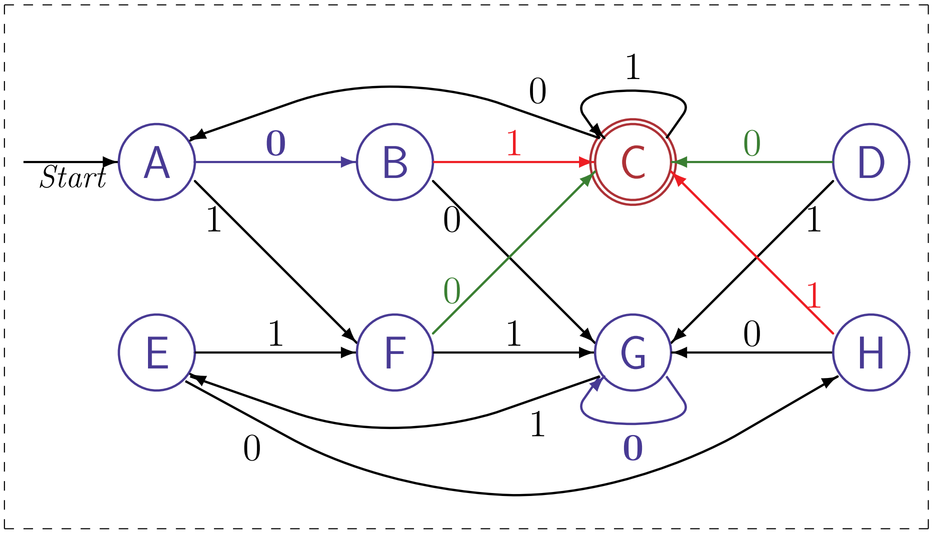
**1.** Unterscheide akzeptierende Zustände (C) von allen anderen

**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*

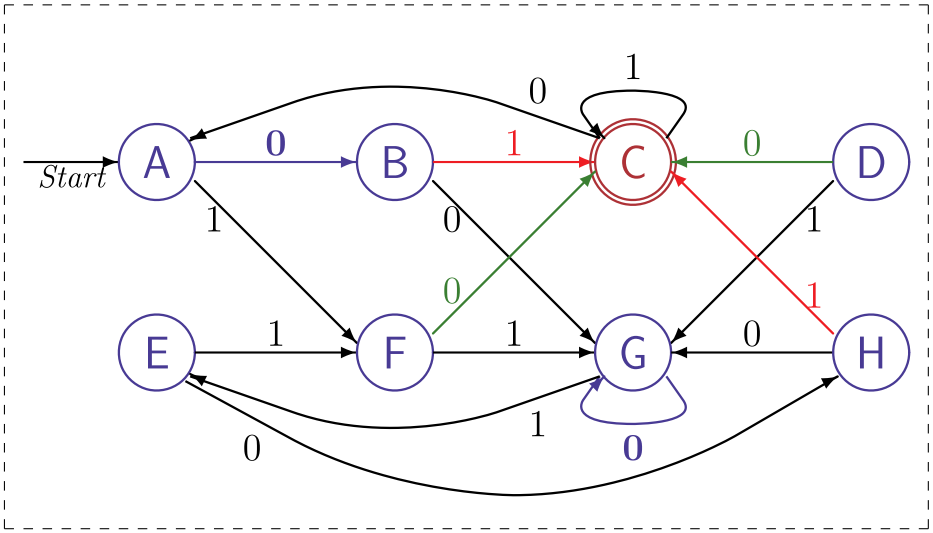
**1.** Unterscheide akzeptierende Zustände (C) von allen anderen

**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*

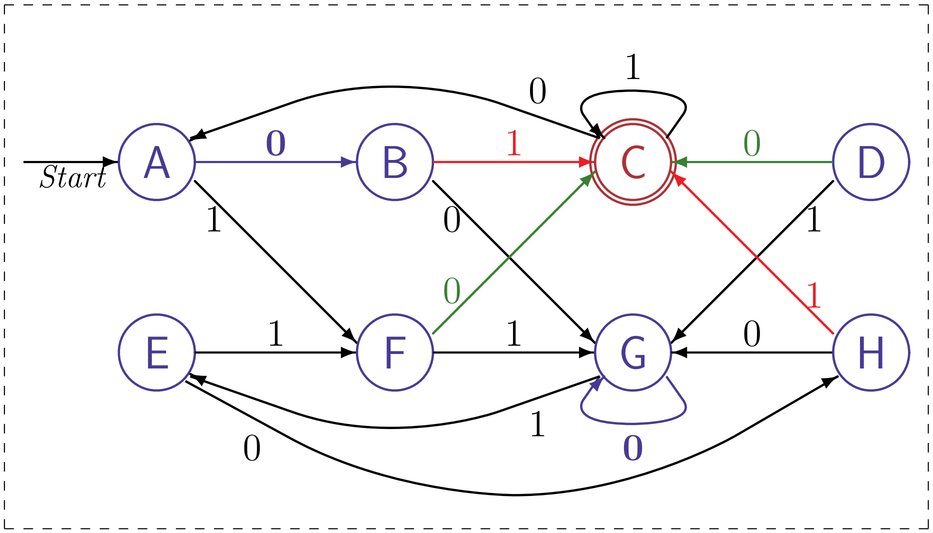
**1.** Unterscheide akzeptierende Zustände (C) von allen anderen

**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*

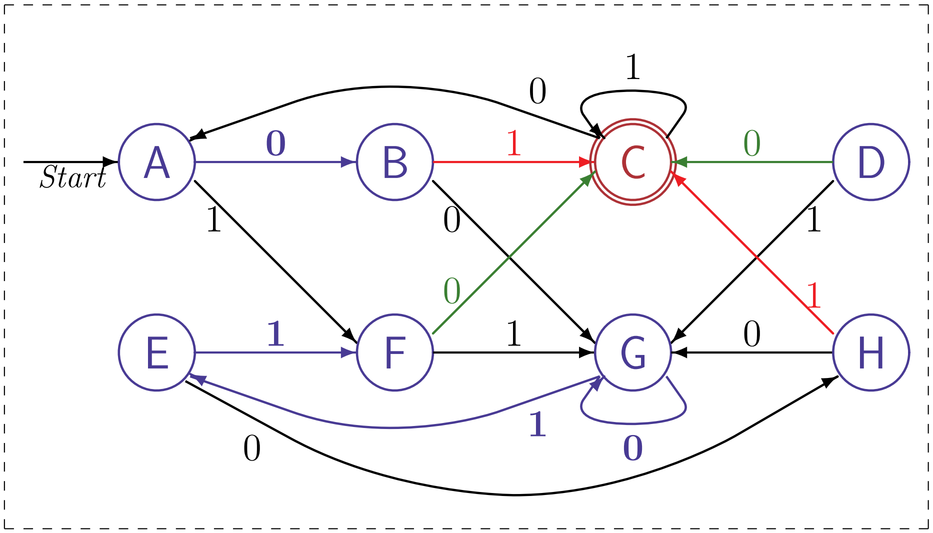
**1.** Unterscheide akzeptierende Zustände (C) von allen anderen

**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*

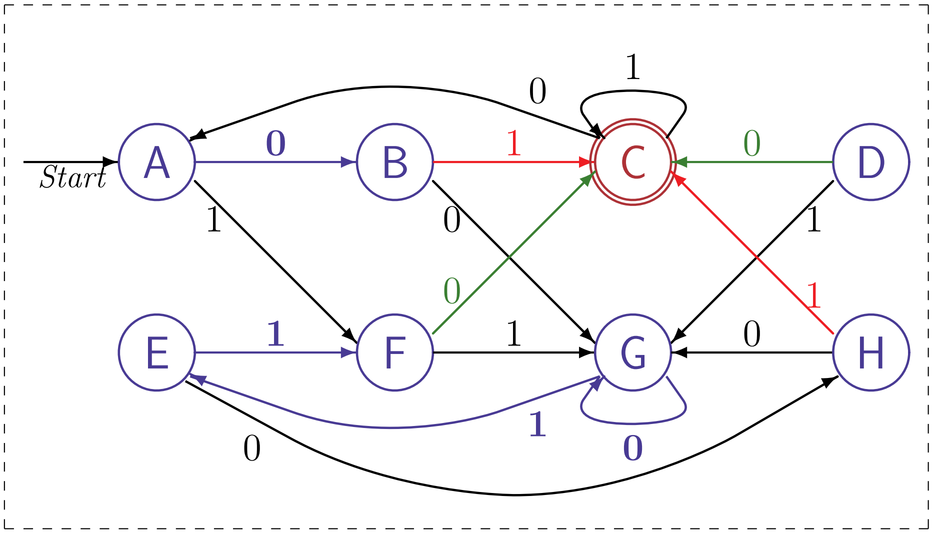
**1.** Unterscheide akzeptierende Zustände (C) von allen anderen

**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 (C) von allen anderen

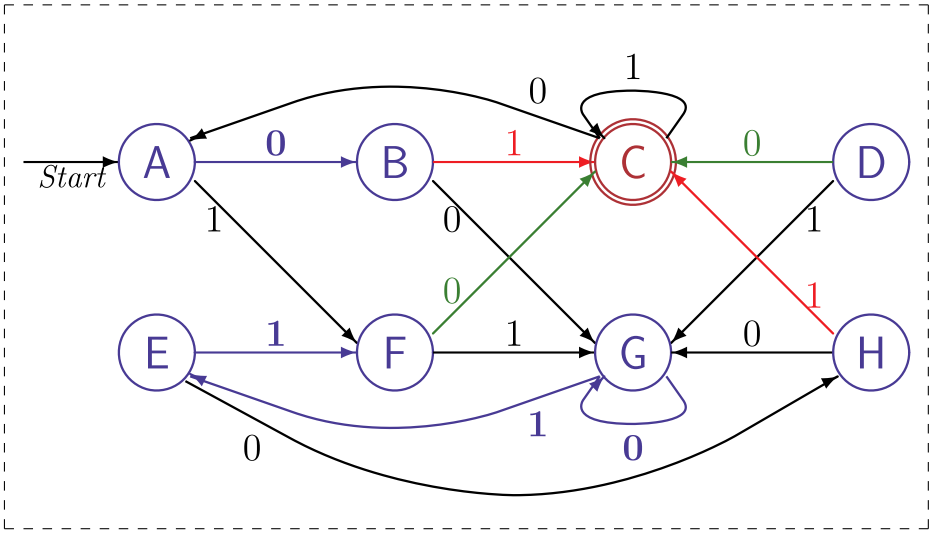
**2a.** Eingabesymbol 0: Nur D und F führen zu akzeptierenden Zuständen

**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 (C) von allen 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

**Äquivalenklassen sind {A,E}, {B,H}, {D,F}, {C} und {G}**

# Äquivalenztest für Sprachen

- **Prüfverfahren**
  - Standardisiere Beschreibungsform in zwei disjunkte DEAs $A_1$ und $A_2$

# Äquivalenztest für Sprachen

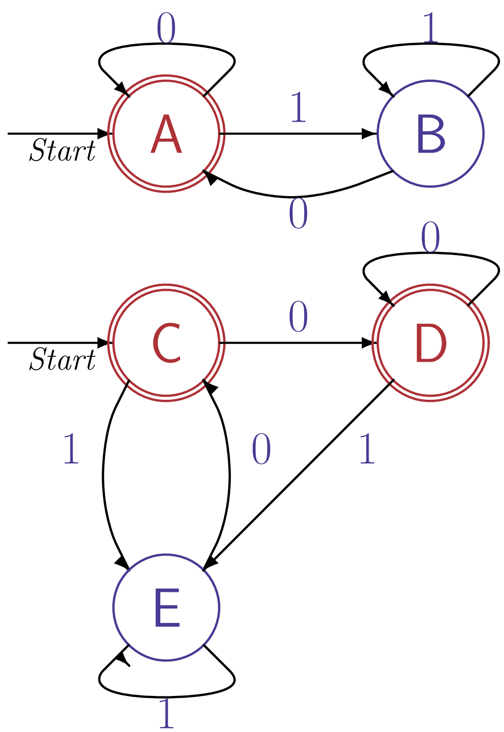
- **Prüfverfahren**
  - Standardisiere Beschreibungsform in zwei disjunkte DEAs $A_1$ und $A_2$
  - Vereinige Automaten zu $A = (Q_1 \cup Q_2 \cup \{q'\},\, \Sigma,\, \delta_1 \cup \delta_2,\, q',\, F_1 \cup F_2)$
    $A$ enthält $A_1$ und $A_2$ als unabhängige Teile

# Äquivalenztest für Sprachen

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

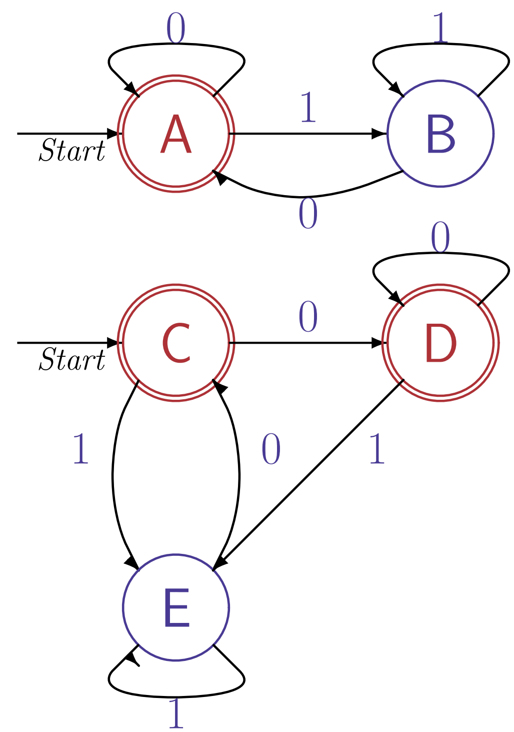
# Äquivalenztest für Sprachen

- **Prüfverfahren**
  - Standardisiere Beschreibungsform in zwei disjunkte DEAs $A_1$ und $A_2$
  - Vereinige Automaten zu $A = (Q_1 \cup Q_2 \cup \{q'\},\ \Sigma,\ \delta_1 \cup \delta_2,\ q',\ F_1 \cup F_2)$
    $A$ enthält $A_1$ und $A_2$ als unabhängige Teile
  - Bilde Äquivalenzklassen von $A$
    und teste ob $q_{0,1}$ und $q_{0,2}$ äquivalent sind
- **Zwei DEAs für $L(\epsilon + (0+1)^*0)$**



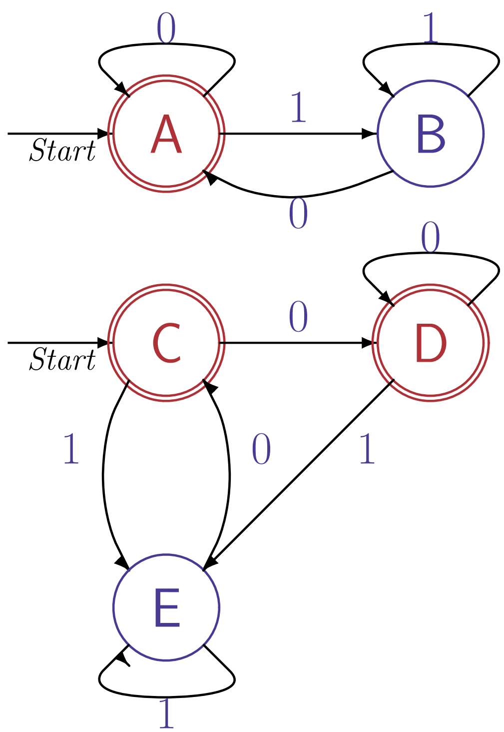
# Äquivalenztest für Sprachen

- **Prüfverfahren**
  - Standardisiere Beschreibungsform in zwei disjunkte DEAs $A_1$ und $A_2$
  - Vereinige Automaten zu $A = (Q_1 \cup Q_2 \cup \{q'\},\ \Sigma,\ \delta_1 \cup \delta_2,\ q',\ F_1 \cup F_2)$
    $A$ enthält $A_1$ und $A_2$ als unabhängige Teile
  - Bilde Äquivalenzklassen von $A$
    und teste ob $q_{0,1}$ und $q_{0,2}$ äquivalent sind
- **Zwei DEAs für $L(\epsilon + (0+1)^*0))$**
  - Äquivalenklassen sind {A,C,D} (alle Endzustände)
    und {B,E} (alle Nicht-Endzustände)



# Äquivalenztest für Sprachen

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




- **Zwei DEAs für $L(\epsilon + (0+1)^*0)$**
  - Äquivalenklassen sind {A,C,D} (alle Endzustände)
    und {B,E} (alle Nicht-Endzustände)
  - **Da A und C äquivalent sind,**
    **sind die Automaten äquivalent**

# Minimierung endlicher Automaten

**Konstruiere äquivalenten DEA mit minimaler Menge von Zuständen**

## Konstruiere äquivalenten DEA mit minimaler Menge von Zuständen

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

# Minimierung endlicher Automaten

**Konstruiere äquivalenten DEA**
**mit minimaler Menge von Zuständen**

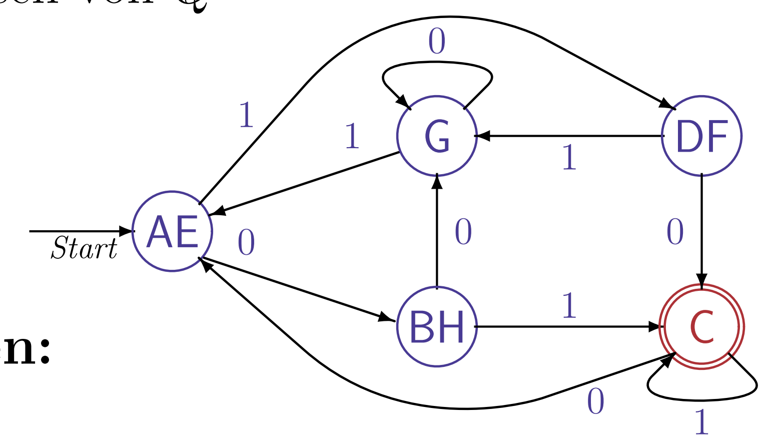
- **Entferne überflüssige Zustände**
  - $q$ ist überflüssig, wenn $\hat{\delta}(q_0, w) \neq q$ für alle Wörter $w \in \Sigma^*$
  - Reduziere $Q$ zu Menge der erreichbaren Zustände (Verfahren auf Folie 11)
- **Fasse äquivalente Zustände zusammen**
  - Bestimme Menge der Äquivalenzklassen von $Q$
 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

## Konstruiere äquivalenten DEA mit minimaler Menge von Zuständen

- **Entferne überflüssige Zustände**
  - $q$ ist überflüssig, wenn $\hat{\delta}(q_0, w) \neq q$ für alle Wörter $w \in \Sigma^*$
  - 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 $\delta'(S, a)$ als Äquivalenzklasse von $\delta(q, a)$ für ein beliebiges $q \in S$

    Wohldefiniert, da alle Nachfolger äquivalenter Zustände äquivalent sind

**Anwendung auf Beispielautomaten:**

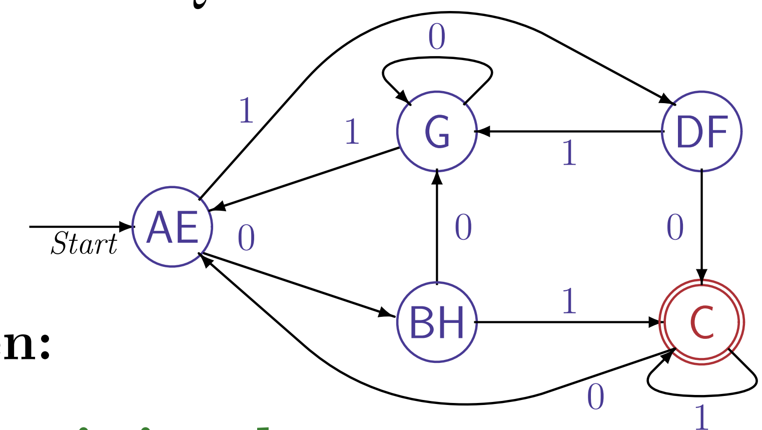


**Konstruiere äquivalenten DEA
mit minimaler Menge von Zuständen**

- **Entferne überflüssige Zustände**
  - $q$ ist überflüssig, wenn $\hat{\delta}(q_0, w) \neq q$ für alle Wörter $w \in \Sigma^*$
  - 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 $\delta'(S, a)$ als Äquivalenzklasse von $\delta(q, a)$ für ein beliebiges $q \in S$

    Wohldefiniert, da alle Nachfolger äquivalenter Zustände äquivalent sind

**Anwendung auf Beispielautomaten:**




- **Resultierender Automat ist minimal**

# Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  – Wörter, die zum gleichen Zustand führen, sind ununterscheidbar

# Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar

# Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar

    Jede Fortsetzung der Wörter führt zum "gleichen" Ergebnis

    $\hat{\delta}(q_0, u) \cong \hat{\delta}(q_0, v)$ bedeutet $\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F$ für alle $w \in \Sigma^*$

# Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar

    Jede Fortsetzung der Wörter führt zum “gleichen” Ergebnis

    $\hat{\delta}(q_0, u) \cong \hat{\delta}(q_0, v)$ bedeutet $\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F$ für alle $w \in \Sigma^*$

- **Äquivalenzklassen hängen nur von der Sprache ab**

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar
    Jede Fortsetzung der Wörter führt zum "gleichen" Ergebnis
    $\hat{\delta}(q_0,u)\cong\hat{\delta}(q_0,v)$ bedeutet $\hat{\delta}(q_0,u\,w)\in F \;\Leftrightarrow\; \hat{\delta}(q_0,v\,w)\in F$ für alle $w\in\Sigma^*$

- **Äquivalenzklassen hängen nur von der Sprache ab**
  - Für $L\subseteq\Sigma^*$ definiere Äquivalenzrelation $\sim_L$ auf $\Sigma^*$:
    - $\boldsymbol{u \sim_L v} \;\equiv\; u\,w\in L \Leftrightarrow v\,w\in L$ gilt für alle $w\in\Sigma^*$

# Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar

    Jede Fortsetzung der Wörter führt zum "gleichen" Ergebnis

    $\hat{\delta}(q_0, u) \cong \hat{\delta}(q_0, v)$ bedeutet $\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F$ für alle $w \in \Sigma^*$

- **Äquivalenzklassen hängen nur von der Sprache ab**
  - Für $L \subseteq \Sigma^*$ definiere Äquivalenzrelation $\sim_L$ auf $\Sigma^*$:
    - $\boldsymbol{u \sim_L v} \;\equiv\; u\,w \in L \Leftrightarrow v\,w \in L$ gilt für alle $w \in \Sigma^*$
  - Die Äquivalenzklasse eines Wortes $v$ ist $[\boldsymbol{v}]_{\boldsymbol{L}} = \{\boldsymbol{u} \in \boldsymbol{\Sigma}^* \mid \boldsymbol{u} \sim_L \boldsymbol{v}\}$

## Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar

    Jede Fortsetzung der Wörter führt zum "gleichen" Ergebnis

    $\hat{\delta}(q_0,u)\cong\hat{\delta}(q_0,v)$ bedeutet $\hat{\delta}(q_0,u\,w)\in F \;\Leftrightarrow\; \hat{\delta}(q_0,v\,w)\in F$ für alle $w\in\Sigma^*$

- **Äquivalenzklassen hängen nur von der Sprache ab**
  - Für $L\subseteq\Sigma^*$ definiere Äquivalenzrelation $\sim_L$ auf $\Sigma^*$:
    - $\boldsymbol{u \sim_L v} \;\equiv\; u\,w\in L \Leftrightarrow v\,w\in L$ gilt für alle $w\in\Sigma^*$
  - Die Äquivalenzklasse eines Wortes $v$ ist $[\boldsymbol{v}]_{\boldsymbol{L}} = \{\boldsymbol{u}\in\boldsymbol{\Sigma^*} \mid \boldsymbol{u \sim_L v}\}$
  - $\boldsymbol{\Sigma^*/L}$ bezeichnet die Menge der Äquivalenzklassen modulo $\sim_L$

# Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar
    Jede Fortsetzung der Wörter führt zum "gleichen" Ergebnis
    $\hat{\delta}(q_0, u) \cong \hat{\delta}(q_0, v)$ bedeutet $\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F$ für alle $w \in \Sigma^*$

- **Äquivalenzklassen hängen nur von der Sprache ab**
  - Für $L \subseteq \Sigma^*$ definiere Äquivalenzrelation $\sim_L$ auf $\Sigma^*$:
    - $\boldsymbol{u \sim_L v} \;\equiv\; u\,w \in L \Leftrightarrow v\,w \in L$ gilt für alle $w \in \Sigma^*$
  - Die Äquivalenzklasse eines Wortes $v$ ist $[\boldsymbol{v}]_{\boldsymbol{L}} = \boldsymbol{\{u \in \Sigma^* \mid u \sim_L v\}}$
  - $\boldsymbol{\Sigma^*/L}$ bezeichnet die Menge der Äquivalenzklassen modulo $\sim_L$
    - Für $L = \{0^n 1^m \mid n, m \in \mathbb{N}\}$ ist $\Sigma^*/L = \{[\epsilon]_L, [1]_L, [10]_L\}$

# Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar
    Jede Fortsetzung der Wörter führt zum "gleichen" Ergebnis
    $\hat{\delta}(q_0, u) \cong \hat{\delta}(q_0, v)$ bedeutet $\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F$ für alle $w \in \Sigma^*$

- **Äquivalenzklassen hängen nur von der Sprache ab**
  - Für $L \subseteq \Sigma^*$ definiere Äquivalenzrelation $\sim_L$ auf $\Sigma^*$:
    - $\boldsymbol{u \sim_L v} \;\equiv\; u\,w \in L \Leftrightarrow v\,w \in L$ gilt für alle $w \in \Sigma^*$
  - Die Äquivalenzklasse eines Wortes $v$ ist $[\boldsymbol{v}]_{\boldsymbol{L}} = \boldsymbol{\{u \in \Sigma^* \mid u \sim_L v\}}$
  - $\boldsymbol{\Sigma^*/L}$ bezeichnet die Menge der Äquivalenzklassen modulo $\sim_L$
    - Für $L = \{0^n 1^m \mid n, m \in \mathbb{N}\}$ ist $\Sigma^*/L = \{[\epsilon]_L, [1]_L, [10]_L\}$
    - Für $L = \{0^n 1^n \mid n \in \mathbb{N}\}$
      ist $\Sigma^*/L = \{[\epsilon]_L, [0]_L, [1]_L, [00]_L, [01]_L, [000]_L, [001]_L, \ldots\}$

## Eine Algebraische Charakterisierung regulärer Sprachen

- **Automaten teilen Sprachen in Äquivalenzklassen**
  - Wörter, die zum gleichen Zustand führen, sind ununterscheidbar
  - Wörter, die zu äquivalenten Zuständen führen, sind ununterscheidbar
    Jede Fortsetzung der Wörter führt zum "gleichen" Ergebnis
    $\hat\delta(q_0,u)\cong\hat\delta(q_0,v)$ bedeutet $\hat\delta(q_0,u\,w)\in F \;\Leftrightarrow\; \hat\delta(q_0,v\,w)\in F$ für alle $w\in\Sigma^*$

- **Äquivalenzklassen hängen nur von der Sprache ab**
  - Für $L\subseteq\Sigma^*$ definiere Äquivalenzrelation $\sim_L$ auf $\Sigma^*$:
    - $\boldsymbol{u \sim_L v} \;\equiv\; u\,w\in L \Leftrightarrow v\,w\in L$ gilt für alle $w\in\Sigma^*$
  - Die Äquivalenzklasse eines Wortes $v$ ist $[\boldsymbol{v}]_{\boldsymbol{L}} = \{\boldsymbol{u}\in\boldsymbol{\Sigma}^* \mid \boldsymbol{u} \sim_L \boldsymbol{v}\}$
  - $\boldsymbol{\Sigma^*/L}$ bezeichnet die Menge der Äquivalenzklassen modulo $\sim_L$
    - Für $L=\{0^n1^m \mid n,m\in\mathbb{N}\}$ ist $\Sigma^*/L = \{[\epsilon]_L, [1]_L, [10]_L\}$
    - Für $L=\{0^n1^n \mid n\in\mathbb{N}\}$
      ist $\Sigma^*/L = \{[\epsilon]_L, [0]_L, [1]_L, [00]_L, [01]_L, [000]_L, [001]_L, ...\}$

**Reguläre Sprachen haben nur endlich viele Äquivalenzklassen**

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

**Beweis**

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

### Beweis

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

**Beweis**

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

**Beweis**

$\Rightarrow$ : Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

**Beweis**

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v)$

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

### Beweis

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \;\Leftrightarrow\; (\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F)$ für alle $w \in \Sigma^*$

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

**Beweis**

$\Rightarrow$ : Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \Leftrightarrow (\hat{\delta}(q_0, u\,w) \in F \Leftrightarrow \hat{\delta}(q_0, v\,w) \in F) \text{ für alle } w \in \Sigma^*$$

$$\Leftrightarrow (u\,w \in L \Leftrightarrow v\,w \in L) \text{ für alle } w \in \Sigma^*$$

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

**Beweis**

$\Rightarrow$ : Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \Leftrightarrow (\hat{\delta}(q_0, u\,w) \in F \Leftrightarrow \hat{\delta}(q_0, v\,w) \in F) \text{ für alle } w \in \Sigma^*$$

$$\Leftrightarrow (u\,w \in L \Leftrightarrow v\,w \in L) \text{ für alle } w \in \Sigma^* \quad \Leftrightarrow \quad u \sim_L v$$

# Der Satz von Myhill/Nerode

## Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist

**Beweis**

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \Leftrightarrow (\hat{\delta}(q_0, u\,w) \in F \Leftrightarrow \hat{\delta}(q_0, v\,w) \in F) \text{ für alle } w \in \Sigma^*$$

$$\Leftrightarrow (u\,w \in L \Leftrightarrow v\,w \in L) \text{ für alle } w \in \Sigma^* \quad \Leftrightarrow \quad u \sim_L v$$

Damit ist $|\Sigma^*/L|$ (der Index von $L$) gleich der Anzahl der Zustände in $A$

# Der Satz von Myhill/Nerode

## Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist

### Beweis

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \;\Leftrightarrow\; (\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F) \;\text{ für alle } w \in \Sigma^*$$
$$\Leftrightarrow\; (u\,w \in L \;\Leftrightarrow\; v\,w \in L) \;\text{ für alle } w \in \Sigma^* \quad\Leftrightarrow\quad u \sim_L v$$

Damit ist $|\Sigma^*/L|$ (der Index von $L$) gleich der Anzahl der Zustände in $A$

$\Leftarrow$: Es sei $\Sigma^*/L$ endlich.

# Der Satz von Myhill/Nerode

## Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist

**Beweis**

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \;\Leftrightarrow\; (\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F) \;\text{ für alle } w \in \Sigma^*$$

$$\Leftrightarrow\; (u\,w \in L \;\Leftrightarrow\; v\,w \in L) \;\text{ für alle } w \in \Sigma^* \quad \Leftrightarrow\; u \sim_L v$$

Damit ist $|\Sigma^*/L|$ (der Index von $L$) gleich der Anzahl der Zustände in $A$

$\Leftarrow$: Es sei $\Sigma^*/L$ endlich.

Konstruiere einen DEA $A = (\Sigma^*/L\,, \Sigma, \delta, [\epsilon]_L, F)$

mit $\delta([u]_L, a) = [u\,a]_L$ für alle $a \in \Sigma$ und $F = \{[v]_L \mid v \in L\}$

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

**Beweis**

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \;\Leftrightarrow\; (\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F)$ für alle $w \in \Sigma^*$

$\Leftrightarrow\; (u\,w \in L \;\Leftrightarrow\; v\,w \in L)$ für alle $w \in \Sigma^*$ $\;\Leftrightarrow\; u \sim_L v$

Damit ist $|\Sigma^*/L|$ (der Index von $L$) gleich der Anzahl der Zustände in $A$

$\Leftarrow$: Es sei $\Sigma^*/L$ endlich.

Konstruiere einen DEA $A = (\Sigma^*/L\,, \Sigma, \delta, [\epsilon]_L, F)$

mit $\delta([u]_L, a) = [u\,a]_L$ für alle $a \in \Sigma$ und $F = \{[v]_L \mid v \in L\}$

$\delta$ ist wohldefiniert, weil $u\,a \sim_L v\,a$ für alle $a \in \Sigma$ gilt, wenn $u \sim_L v$

# Der Satz von Myhill/Nerode

## Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist

**Beweis**

$\Rightarrow$ : Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \;\Leftrightarrow\; (\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F) \text{ für alle } w \in \Sigma^*$$
$$\Leftrightarrow\; (u\,w \in L \;\Leftrightarrow\; v\,w \in L) \text{ für alle } w \in \Sigma^* \quad\Leftrightarrow\; u \sim_L v$$

Damit ist $|\Sigma^*/L|$ (der Index von $L$) gleich der Anzahl der Zustände in $A$

$\Leftarrow$ : Es sei $\Sigma^*/L$ endlich.

Konstruiere einen DEA $A = (\Sigma^*/L\;, \Sigma, \delta, [\epsilon]_L, F)$

mit $\delta([u]_L, a) = [u\,a]_L$ für alle $a \in \Sigma$ und $F = \{[v]_L \mid v \in L\}$

$\delta$ ist wohldefiniert, weil $u\,a \sim_L v\,a$ für alle $a \in \Sigma$ gilt, wenn $u \sim_L v$

und es gilt $w \in L(A) \;\Leftrightarrow\; \hat{\delta}([\epsilon]_L, w) \in F$

# Der Satz von Myhill/Nerode

**Eine Sprache $L$ ist regulär, g.d.w $\Sigma^*/L$ endlich ist**

**Beweis**

$\Rightarrow$: Es sei $L$ eine reguläre Sprache

Dann gibt es einen minimalen DEA $A = (Q, \Sigma, \delta, q_0, F)$ mit $L = L(A)$

Da $A$ minimal ist, gilt für beliebige Wörter $u, v \in \Sigma^*$

$$\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v) \;\Leftrightarrow\; (\hat{\delta}(q_0, u\,w) \in F \;\Leftrightarrow\; \hat{\delta}(q_0, v\,w) \in F) \text{ für alle } w \in \Sigma^*$$

$$\Leftrightarrow\; (u\,w \in L \;\Leftrightarrow\; v\,w \in L) \text{ für alle } w \in \Sigma^* \;\Leftrightarrow\; u \sim_L v$$

Damit ist $|\Sigma^*/L|$ (der Index von $L$) gleich der Anzahl der Zustände in $A$

$\Leftarrow$: Es sei $\Sigma^*/L$ endlich.

Konstruiere einen DEA $A = (\Sigma^*/L\ , \Sigma, \delta, [\epsilon]_L, F)$

mit $\delta([u]_L, a) = [u\,a]_L$ für alle $a \in \Sigma$ und $F = \{[v]_L \mid v \in L\}$

$\delta$ ist wohldefiniert, weil $u\,a \sim_L v\,a$ für alle $a \in \Sigma$ gilt, wenn $u \sim_L v$

und es gilt $w \in L(A) \;\Leftrightarrow\; \hat{\delta}([\epsilon]_L, w) \in F \;\Leftrightarrow\; [w]_L \in F \;\Leftrightarrow\; w \in L$

# Grenzen regulärer Sprachen

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

# Grenzen regulärer Sprachen

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

- **Direkter Nachweis**
  – Zeige, dass kein endlicher Automat genau die Wörter von $L$ erkennt

# Grenzen regulärer Sprachen

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

- **Direkter Nachweis**
  - Zeige, dass kein endlicher Automat genau die Wörter von $L$ erkennt
  - Sprache muss unendlich sein und komplizierte Struktur haben
    (Anzahl der Äquivalenzklassen muss unendlich sein)

# Grenzen regulärer Sprachen

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

- **Direkter Nachweis**
  - Zeige, dass kein endlicher Automat genau die Wörter von $L$ erkennt
  - Sprache muss unendlich sein und komplizierte Struktur haben (Anzahl der Äquivalenzklassen muss unendlich sein)
  - Technisches Hilfsmittel: Pumping Lemma

# Grenzen regulärer Sprachen

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

- **Direkter Nachweis**
  - Zeige, dass kein endlicher Automat genau die Wörter von $L$ erkennt
  - Sprache muss unendlich sein und komplizierte Struktur haben (Anzahl der Äquivalenzklassen muss unendlich sein)
  - Technisches Hilfsmittel: Pumping Lemma
- **Verwendung der Abschlusseigenschaften**
  - Zeige, dass Regularität von $L$ dazu führen würde, dass eine als nichtregulär bekannte Sprache regulär sein müsste

# Grenzen regulärer Sprachen

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

- **Direkter Nachweis**
  - Zeige, dass kein endlicher Automat genau die Wörter von $L$ erkennt
  - Sprache muss unendlich sein und komplizierte Struktur haben (Anzahl der Äquivalenzklassen muss unendlich sein)
  - 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 DEA muss alle Nullen beim Abarbeiten zählen und dann vergleichen

# Das Pumping Lemma für reguläre Sprachen

- **Warum ist $\{0^n1^n \mid n\in\mathbb{N}\}$ nicht regulär?**
  - Ein DEA 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 $\{0^n1^n \mid n \in \mathbb{N}\}$ nicht regulär?**
  - Ein DEA 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 DEA 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 $\{0^n1^n \mid n \in \mathbb{N}\}$ nicht regulär?**
  - Ein DEA 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

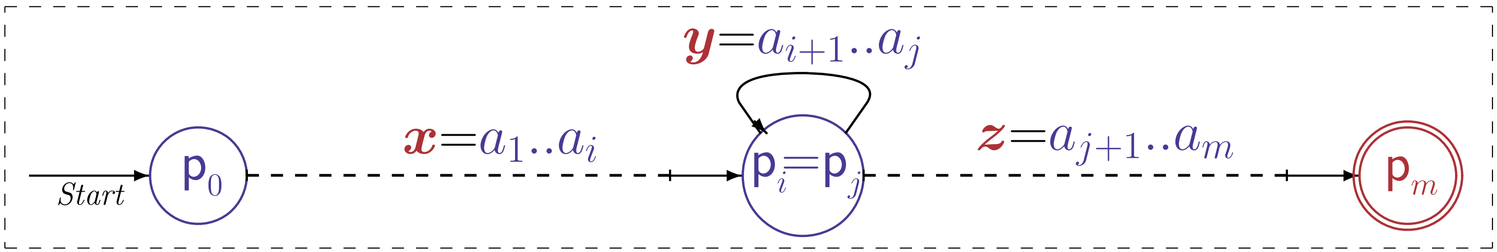
**Für jede Sprache $L \in \mathcal{L}_3$ gibt es ein $n \in \mathbb{N}$, so dass jedes $w \in L$ mit $|w| \geq n$ zerlegbar ist in $w = x\,y\,z$ mit den Eigenschaften (1) $y \neq \epsilon$, (2) $|x\,y| \leq n$ und (3) für alle $k \in \mathbb{N}$ ist $x\,y^k\,z \in L$**

- **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 $x\,y^k\,z \in L$**

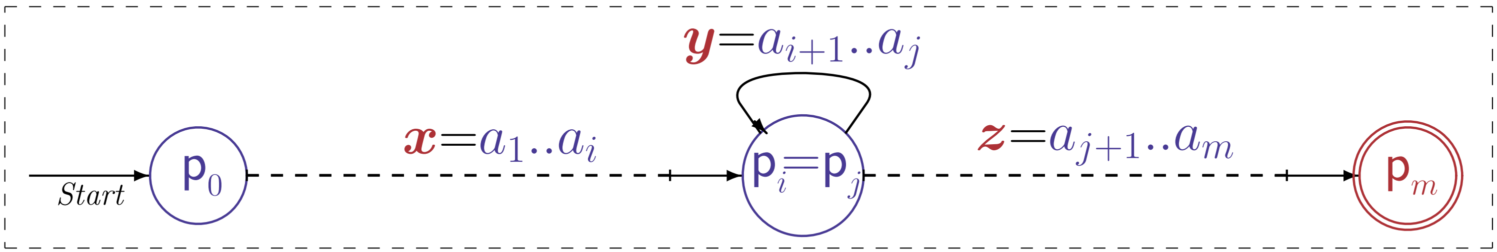
- **Beweis mit Automaten**
  - Sei $L$ regulär und $A = (Q, \Sigma, \delta, q_0, F)$ ein DEA mit $L = L(A)$
  - Wähle $n{=}|Q|$. Betrachte $w{=}a_1..a_m$ mit $|w| \geq n$ und $p_i := \hat{\delta}(q_0, a_1..a_i)$
  - Dann gibt es $i, j$ mit $0 \leq i < j \leq n$ und $p_i = p_j$ (Schubfachprinzip)
  -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



# 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 $w = a_1..a_m$ 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)
  - Zerlege $w$ in $w = x\,y\,z$ mit $x = a_1..a_i$, $y = a_{i+1}..a_j$ und $z = a_{j+1}..a_m$

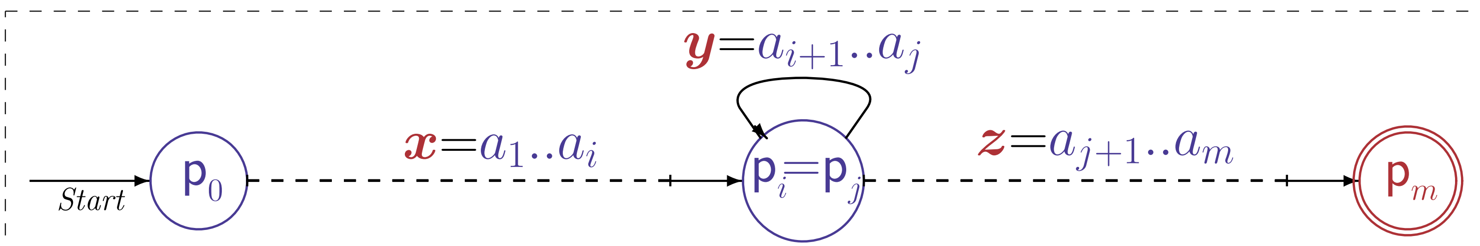



  - Per Konstruktion gilt $y \neq \epsilon$, $|x\,y| \leq n$ und $\hat{\delta}(p_i, y^k) = p_i$ für alle $k \in \mathbb{N}$

# 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 $w{=}a_1..a_m$ 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)
  - Zerlege $w$ in $w = x\,y\,z$ mit $x{=}a_1..a_i$, $y{=}a_{i+1}..a_j$ und $z{=}a_{j+1}..a_m$




  - 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

# Anwendungen des Pumping Lemmas

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

- **Verwende Umkehrung des Pumping Lemmas**

  *Eine Sprache $L$ ist nicht regulär, wenn es kein $n \in \mathbb{N}$ gibt, 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 $x\,y^k\,z \in L$*

# Anwendungen des Pumping Lemmas

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

- **Verwende Umkehrung des Pumping Lemmas**

  *Eine Sprache $L$ ist nicht regulär, wenn es kein $n \in \mathbb{N}$ gibt, 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 $x\,y^k\,z \in L$*

  **Umformulierung: Ziehe Negation in die Bedingungen hinein**

  *$L$ ist nicht regulär, wenn es für jedes $n \in \mathbb{N}$ ein $w \in L$ mit $|w| \geq n$ gibt so dass für jede Zerlegung $w = x\,y\,z$ mit den Eigenschaften*

  *(1) $y \neq \epsilon$ und (2) $|x\,y| \leq n$ ein $k \in \mathbb{N}$ existiert mit $x\,y^k\,z \notin L$*

# Anwendungen des Pumping Lemmas

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

- **Verwende Umkehrung des Pumping Lemmas**

  *Eine Sprache $L$ ist nicht regulär, wenn es kein $n \in \mathbb{N}$ gibt, 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 $x\,y^k\,z \in L$*

  **Umformulierung: Ziehe Negation in die Bedingungen hinein**

  *$L$ ist nicht regulär, wenn es für jedes $n \in \mathbb{N}$ ein $w \in L$ mit $|w| \geq n$ gibt so dass für jede Zerlegung $w = x\,y\,z$ mit den Eigenschaften*

  *(1) $y \neq \epsilon$ und (2) $|x\,y| \leq n$ ein $k \in \mathbb{N}$ existiert mit $x\,y^k\,z \notin L$*

- **Kontrapositionsbeweis für $L_1 \notin \mathcal{L}_3$**
  – Sei $n \in \mathbb{N}$ beliebig.

# Anwendungen des Pumping Lemmas

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

- **Verwende Umkehrung des Pumping Lemmas**

  *Eine Sprache $L$ ist nicht regulär, wenn es kein $n \in \mathbb{N}$ gibt, 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 $x\,y^k\,z \in L$*

  **Umformulierung: Ziehe Negation in die Bedingungen hinein**

  *$L$ ist nicht regulär, wenn es für jedes $n \in \mathbb{N}$ ein $w \in L$ mit $|w| \geq n$ gibt so dass für jede Zerlegung $w = x\,y\,z$ mit den Eigenschaften*

  *(1) $y \neq \epsilon$ und (2) $|x\,y| \leq n$ ein $k \in \mathbb{N}$ existiert mit $x\,y^k\,z \notin L$*

- **Kontrapositionsbeweis für $L_1 \notin \mathcal{L}_3$**

  – Sei $n \in \mathbb{N}$ beliebig. Wir wählen $w = 0^m 1^m$ für ein $m > n$

# Anwendungen des Pumping Lemmas

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

- **Verwende Umkehrung des Pumping Lemmas**

  *Eine Sprache $L$ ist nicht regulär, wenn es kein $n \in \mathbb{N}$ gibt, 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 $x\,y^k\,z \in L$*

  **Umformulierung: Ziehe Negation in die Bedingungen hinein**

  *$L$ ist nicht regulär, wenn es für jedes $n \in \mathbb{N}$ ein $w \in L$ mit $|w| \geq n$ gibt so dass für jede Zerlegung $w = x\,y\,z$ mit den Eigenschaften*

  *(1) $y \neq \epsilon$ und (2) $|x\,y| \leq n$ ein $k \in \mathbb{N}$ existiert mit $x\,y^k\,z \notin L$*

- **Kontrapositionsbeweis für $L_1 \notin \mathcal{L}_3$**
  - Sei $n \in \mathbb{N}$ beliebig. Wir wählen $w = 0^m 1^m$ für ein $m > n$
  - Sei $w = x\,y\,z$ eine beliebige Zerlegung mit $y \neq \epsilon$ und $|x\,y| \leq n$

# Anwendungen des Pumping Lemmas

## $L_1 = \{0^m1^m \mid m \in \mathbb{N}\}$ ist nicht regulär

- **Verwende Umkehrung des Pumping Lemmas**

  *Eine Sprache $L$ ist nicht regulär, wenn es kein $n \in \mathbb{N}$ gibt, 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 $x\,y^k\,z \in L$*

  **Umformulierung: Ziehe Negation in die Bedingungen hinein**

  *$L$ ist nicht regulär, wenn es für jedes $n \in \mathbb{N}$ ein $w \in L$ mit $|w| \geq n$ gibt so dass für jede Zerlegung $w = x\,y\,z$ mit den Eigenschaften*

  *(1) $y \neq \epsilon$ und (2) $|x\,y| \leq n$ ein $k \in \mathbb{N}$ existiert mit $x\,y^k\,z \notin L$*

- **Kontrapositionsbeweis für $L_1 \notin \mathcal{L}_3$**
  - Sei $n \in \mathbb{N}$ beliebig. Wir wählen $w = 0^m1^m$ für ein $m > n$
  - Sei $w = x\,y\,z$ eine beliebige Zerlegung mit $y \neq \epsilon$ und $|x\,y| \leq n$

    Dann gilt $x = 0^i$, $y = 0^j$ $z = 0^{m-i-j}1^m$ für ein $j \neq 0$ und $i+j \leq n$.

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

- **Verwende Umkehrung des Pumping Lemmas**

  *Eine Sprache $L$ ist nicht regulär, wenn es kein $n \in \mathbb{N}$ gibt, 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 $x\,y^k\,z \in L$*

  **Umformulierung: Ziehe Negation in die Bedingungen hinein**

  *$L$ ist nicht regulär, wenn es für jedes $n \in \mathbb{N}$ ein $w \in L$ mit $|w| \geq n$ gibt so dass für jede Zerlegung $w = x\,y\,z$ mit den Eigenschaften*

  *(1) $y \neq \epsilon$ und (2) $|x\,y| \leq n$ ein $k \in \mathbb{N}$ existiert mit $x\,y^k\,z \notin L$*

- **Kontrapositionsbeweis für $L_1 \notin \mathcal{L}_3$**
  - Sei $n \in \mathbb{N}$ beliebig. Wir wählen $w = 0^m 1^m$ für ein $m > n$
  - Sei $w = x\,y\,z$ eine beliebige Zerlegung mit $y \neq \epsilon$ und $|x\,y| \leq n$

    Dann gilt $x = 0^i$, $y = 0^j$ $z = 0^{m-i-j} 1^m$ für ein $j \neq 0$ und $i + j \leq n$.
  - Wir wählen $k = 0$. Dann ist $x\,y^0\,z = 0^{m-j} 1^m \notin L_1$

# Anwendungen des Pumping Lemmas

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

- **Verwende Umkehrung des Pumping Lemmas**

  *Eine Sprache $L$ ist nicht regulär, wenn es kein $n \in \mathbb{N}$ gibt, 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 $x\,y^k\,z \in L$*

  **Umformulierung: Ziehe Negation in die Bedingungen hinein**

  *$L$ ist nicht regulär, wenn es für jedes $n \in \mathbb{N}$ ein $w \in L$ mit $|w| \geq n$ gibt so dass für jede Zerlegung $w = x\,y\,z$ mit den Eigenschaften*

  *(1) $y \neq \epsilon$ und (2) $|x\,y| \leq n$ ein $k \in \mathbb{N}$ existiert mit $x\,y^k\,z \notin L$*

- **Kontrapositionsbeweis für $L_1 \notin \mathcal{L}_3$**
  - Sei $n \in \mathbb{N}$ beliebig. Wir wählen $w = 0^m 1^m$ für ein $m > n$
  - Sei $w = x\,y\,z$ eine beliebige Zerlegung mit $y \neq \epsilon$ und $|x\,y| \leq n$

    Dann gilt $x = 0^i$, $y = 0^j$ $z = 0^{m-i-j} 1^m$ für ein $j \neq 0$ und $i + j \leq n$.
  - Wir wählen $k = 0$. Dann ist $x\,y^0\,z = 0^{m-j} 1^m \notin L_1$
  - Aufgrund des Pumping Lemmas kann $L_1$ also nicht regulär sein.

# Anwendungen des Pumping Lemmas II

$$L_2 = \{w \in \{1\}^* \mid |w| \text{ ist Primzahl}\} \not\in \mathcal{L}_3$$

# Anwendungen des Pumping Lemmas II

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

- **Beweis**
  – Sei $n \in \mathbb{N}$ beliebig.

# Anwendungen des Pumping Lemmas II

$$L_2 = \{w \in \{1\}^* \mid |w| \text{ ist Primzahl}\} \not\in \mathcal{L}_3$$

- **Beweis**
  - Sei $n \in \mathbb{N}$ beliebig.
  - Wir wählen $w = 1^p$ für eine Primzahl $p > n+1$

# Anwendungen des Pumping Lemmas II

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

- **Beweis**
  - Sei $n \in \mathbb{N}$ beliebig.
  - Wir wählen $w = 1^p$ für eine Primzahl $p > n+1$
  - Sei $w = x\,y\,z$ eine beliebige Zerlegung mit $y \neq \epsilon$ und $|x\,y| \leq n$

    Dann gilt $x = 1^i$, $y = 1^j$ $z = 1^{p-i-j}$ für ein $j \neq 0$ und $i+j \leq n$.

# Anwendungen des Pumping Lemmas II

$$L_2 = \{w \in \{1\}^* \mid |w| \text{ ist Primzahl}\} \not\in \mathcal{L}_3$$

- **Beweis**
  - Sei $n \in \mathbb{N}$ beliebig.
  - Wir wählen $w = 1^p$ für eine Primzahl $p > n+1$
  - Sei $w = x\,y\,z$ eine beliebige Zerlegung mit $y \neq \epsilon$ und $|x\,y| \leq n$

    Dann gilt $x = 1^i$, $y = 1^j$ $z = 1^{p-i-j}$ für ein $j \neq 0$ und $i+j \leq n$.
  - Wir wählen $k = p-j$.

# Anwendungen des Pumping Lemmas II

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

- **Beweis**
  - Sei $n \in \mathbb{N}$ beliebig.
  - Wir wählen $w = 1^p$ für eine Primzahl $p > n+1$
  - Sei $w = x\,y\,z$ eine beliebige Zerlegung mit $y \neq \epsilon$ und $|x\,y| \leq n$

    Dann gilt $x = 1^i$, $y = 1^j$ $z = 1^{p-i-j}$ für ein $j \neq 0$ und $i+j \leq n$.
  - Wir wählen $k = p-j$.

    Dann ist $x\,y^k\,z = 1^i\,1^{j(p-j)}\,1^{p-i-j} = 1^{i+j(p-j)+p-i-j} = 1^{(j+1)(p-j)} \notin L_2$

# Anwendungen des Pumping Lemmas II

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

- **Beweis**
  - Sei $n \in \mathbb{N}$ beliebig.
  - Wir wählen $w = 1^p$ für eine Primzahl $p{>}n{+}1$
  - Sei $w = x\,y\,z$ eine beliebige Zerlegung mit $y{\neq}\epsilon$ und $|x\,y|{\leq}n$

    Dann gilt $x{=}1^i$, $y{=}1^j$ $z{=}1^{p-i-j}$ für ein $j{\neq}0$ und $i{+}j{\leq}n$.
  - Wir wählen $k{=}p{-}j$.

    Dann ist $x\,y^k\,z = 1^i\,1^{j(p-j)}\,1^{p-i-j} = 1^{i+j(p-j)+p-i-j} = 1^{(j+1)(p-j)} \notin L_2$
  - Aufgrund des Pumping Lemmas kann $L_2$ also nicht regulär sein.

# Nachweis von $L \notin \mathcal{L}_3$ mit Abschlusseigenschaften

- **Anwendung des Pumping Lemmas ist oft mühsam**

# Nachweis von $L \not\in \mathcal{L}_3$ mit Abschlusseigenschaften

- **Anwendung des Pumping Lemmas ist oft mühsam**
  - Beweis für $\boldsymbol{L_3 = \{(^m)^m \mid m \in \mathbb{N}\} \not\in \mathcal{L}_3}$ identisch mit dem von $L_1$

# Nachweis von $L \notin \mathcal{L}_3$ mit Abschlusseigenschaften

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

- **Verwende Umkehrung der Abschlusseigenschaften**

$$\overline{L} \notin \mathcal{L}_3 \;\Rightarrow\; L \notin \mathcal{L}_3 \qquad L^R \notin \mathcal{L}_3 \;\Rightarrow\; L \notin \mathcal{L}_3$$

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

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

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

$$\vdots \qquad\qquad \vdots$$

# Nachweis von $L \notin \mathcal{L}_3$ mit Abschlusseigenschaften

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

- **Verwende Umkehrung der Abschlusseigenschaften**

$$\overline{L} \notin \mathcal{L}_3 \;\Rightarrow\; L \notin \mathcal{L}_3 \qquad L^R \notin \mathcal{L}_3 \;\Rightarrow\; L \notin \mathcal{L}_3$$

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

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

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

$$\vdots \qquad\qquad\qquad \vdots$$

- **Anwendungsbeispiele**

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

- **Verwende Umkehrung der Abschlusseigenschaften**

$$\overline{L} \notin \mathcal{L}_3 \;\Rightarrow\; L \notin \mathcal{L}_3 \qquad L^R \notin \mathcal{L}_3 \;\Rightarrow\; L \notin \mathcal{L}_3$$

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

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

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

$$\vdots \qquad\qquad \vdots$$

- **Anwendungsbeispiele**

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

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

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

# Nachweis von $L \notin \mathcal{L}_3$ mit Abschlusseigenschaften

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

- **Verwende Umkehrung der Abschlusseigenschaften**

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

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

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

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

$$\vdots \qquad\qquad \vdots$$

- **Anwendungsbeispiele**

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

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

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

**DEAs können korrekte Klammerausdrücke nicht erkennen!**

# Eigenschaften regulärer Sprachen im Rückblick

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

# Eigenschaften regulärer Sprachen im Rückblick

- **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

# Eigenschaften regulärer Sprachen im Rückblick

- **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

# Eigenschaften regulärer Sprachen im Rückblick

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

# Zusammenfassung: reguläre Sprachen

- **Drei Modelle**
  – Endliche Automaten (DEA, NEA, $\epsilon$-NEA) erkennen Wörter einer Sprache
  – Reguläre Ausdrücke beschreiben Struktur der Wörter
  – (Typ 3) Grammatiken erzeugen Wörter einer regulären Sprache

# Zusammenfassung: reguläre Sprachen

- **Drei Modelle**
  - Endliche Automaten (DEA, NEA, $\epsilon$-NEA) erkennen Wörter einer Sprache
  - Reguläre Ausdrücke beschreiben Struktur der Wörter
  - (Typ 3) Grammatiken erzeugen Wörter einer regulären Sprache
- **Alle drei Modelle sind äquivalent**
  - $\epsilon$-NEA $\mapsto$ DEA: Teilmengenkonstruktion
  - DEA $\mapsto$ Typ-3 Grammatik: Verwandle Überführungsfunktion in Regeln
  - Typ-3 Grammatik $\mapsto$ NEA: Verwandle Regeln in Überführungsfunktion
  - DEA $\mapsto$ Reguläre Ausdrücke: Erzeuge Ausdrücke für Verarbeitungspfade oder eliminiere Zustände in RA Automaten
  - Reguläre Ausdrücke $\mapsto$ NEA: Iterative Konstruktion von Automaten

# Zusammenfassung: reguläre Sprachen

- **Drei Modelle**
  - Endliche Automaten (DEA, NEA, $\epsilon$-NEA) erkennen Wörter einer Sprache
  - Reguläre Ausdrücke beschreiben Struktur der Wörter
  - (Typ 3) Grammatiken erzeugen Wörter einer regulären Sprache
- **Alle drei Modelle sind äquivalent**
  - $\epsilon$-NEA $\mapsto$ DEA: Teilmengenkonstruktion
  - DEA $\mapsto$ Typ-3 Grammatik: Verwandle Überführungsfunktion in Regeln
  - Typ-3 Grammatik $\mapsto$ NEA: Verwandle Regeln in Überführungsfunktion
  - DEA $\mapsto$ Reguläre Ausdrücke: Erzeuge Ausdrücke für Verarbeitungspfade oder eliminiere Zustände in RA Automaten
  - Reguläre Ausdrücke $\mapsto$ NEA: Iterative Konstruktion von Automaten
- **Wichtige Eigenschaften von $\mathcal{L}_3$**
  - Abgeschlossen unter $\cup$, $\cap$, $\overline{\phantom{x}}$, -, $^R$, $\circ$, $^*$, $h$, $h^{-1}$
  - Entscheidbarkeit des Wortproblems und Gleichheit von Sprachen
  - Endliche Automaten können automatisch minimiert werden
  - Nachweis der Nichtregularität von Sprachen mit dem Pumping Lemma