

Theoretische Informatik I



Einheit 2.2

Nichtdeterministische Automaten



1. Arbeitsweise
2. Akzeptierte Sprache
3. Äquivalenz zu deterministischen Automaten

WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)

WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage

WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

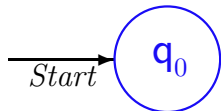
- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage
 - Mühsame Beschreibung eines deterministischen Erkennungsverfahrens

WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage
 - Mühsame Beschreibung eines deterministischen Erkennungsverfahrens
- Idee: **Simultane Verarbeitung** von Alternativen

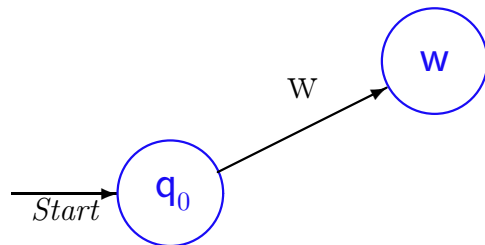
WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage
 - Mühsame Beschreibung eines deterministischen Erkennungsverfahrens
- **Idee: Simultane Verarbeitung** von Alternativen
 - z.B. Suche nach den Worten **web** und **ebay**



WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

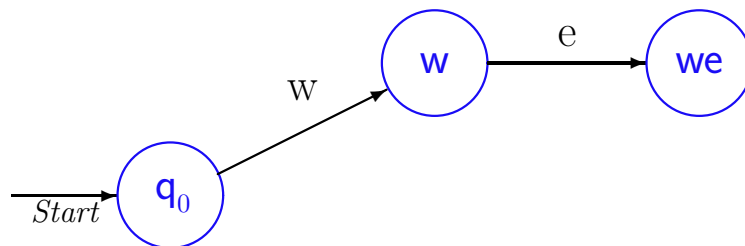
- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage
 - Mühsame Beschreibung eines deterministischen Erkennungsverfahrens
- **Idee: Simultane Verarbeitung** von Alternativen
 - z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein

WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

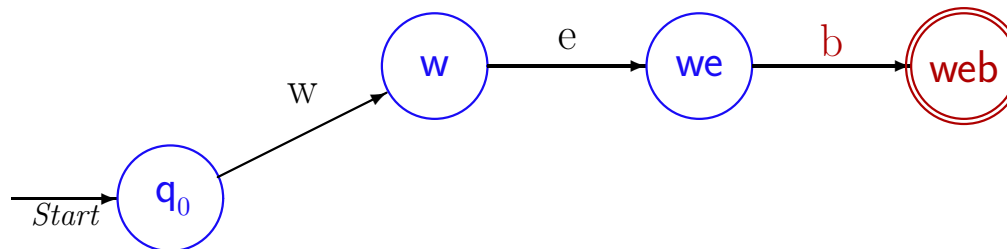
- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage
 - Mühsame Beschreibung eines deterministischen Erkennungsverfahrens
- **Idee: Simultane Verarbeitung** von Alternativen
 - z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein

WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage
 - Mühsame Beschreibung eines deterministischen Erkennungsverfahrens
- **Idee: Simultane Verarbeitung** von Alternativen
 - z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein

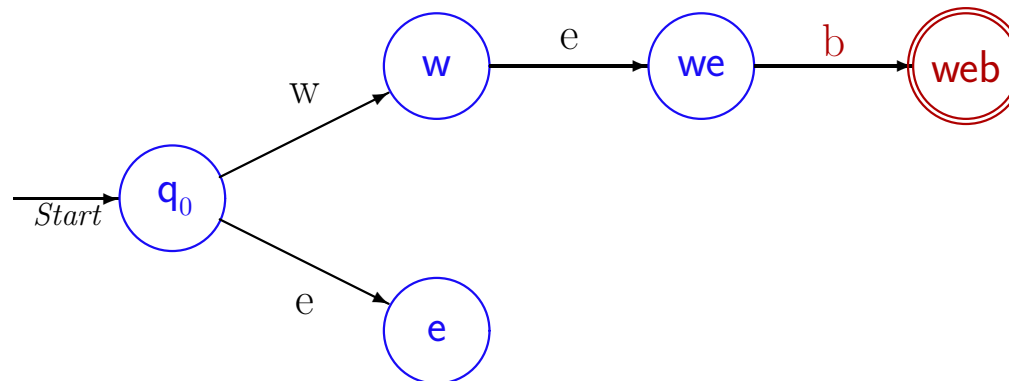
WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten

- Viele verschiedene Worte in großen Textsammlungen (Internet)
- Leichte Beschreibung der Suchanfrage
- Mühsame Beschreibung eines deterministischen Erkennungsverfahrens

- **Idee: Simultane Verarbeitung** von Alternativen

- z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein
- Ein **e** könnte der Anfang von **ebay** sein

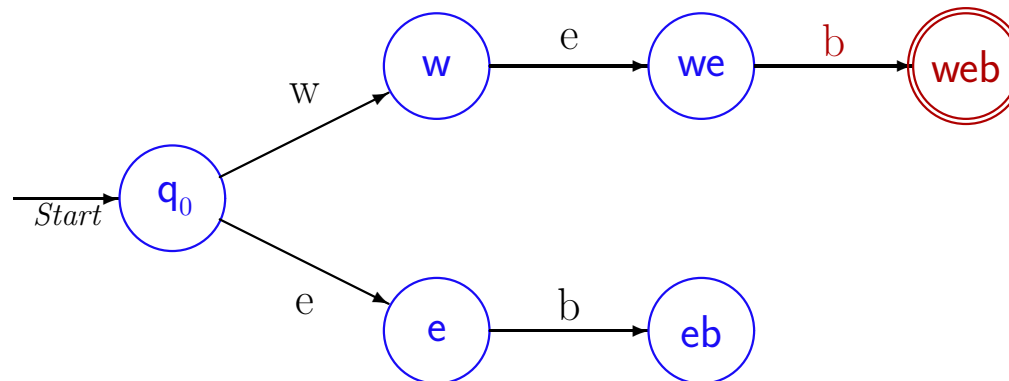
WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten

- Viele verschiedene Worte in großen Textsammlungen (Internet)
- Leichte Beschreibung der Suchanfrage
- Mühsame Beschreibung eines deterministischen Erkennungsverfahrens

- **Idee: Simultane Verarbeitung** von Alternativen

- z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein
- Ein **e** könnte der Anfang von **ebay** sein

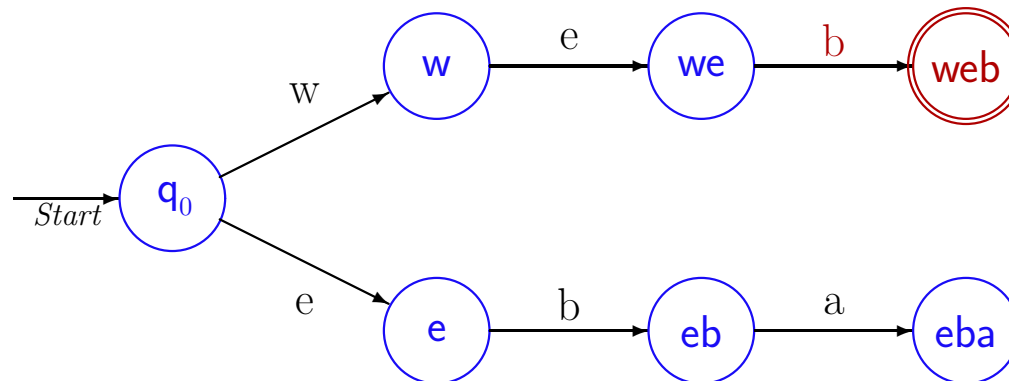
WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten

- Viele verschiedene Worte in großen Textsammlungen (Internet)
- Leichte Beschreibung der Suchanfrage
- Mühsame Beschreibung eines deterministischen Erkennungsverfahrens

- **Idee: Simultane Verarbeitung** von Alternativen

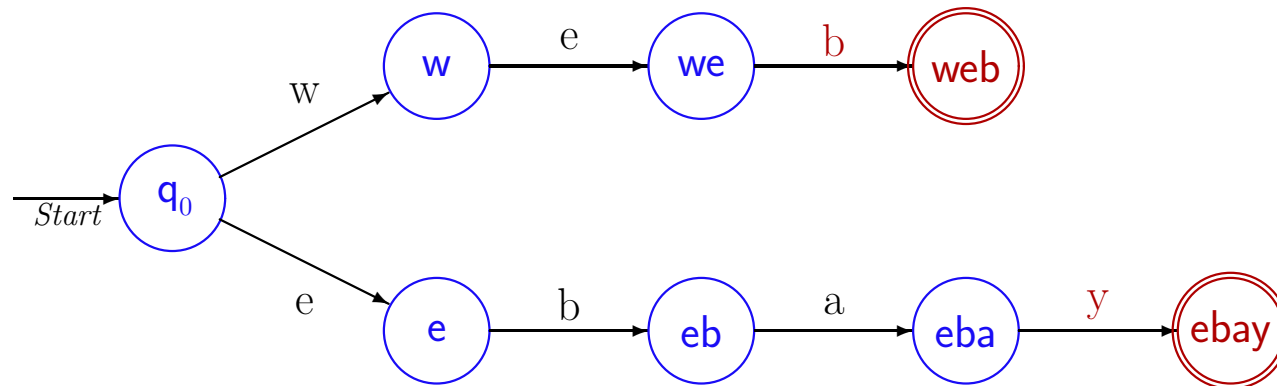
- z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein
- Ein **e** könnte der Anfang von **ebay** sein

WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage
 - Mühsame Beschreibung eines deterministischen Erkennungsverfahrens
- **Idee: Simultane Verarbeitung** von Alternativen
 - z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein
- Ein **e** könnte der Anfang von **ebay** sein

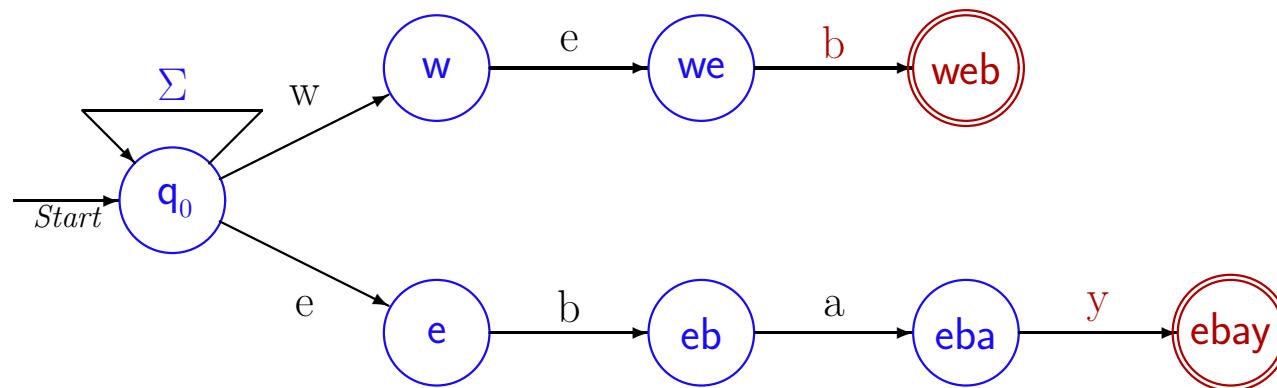
WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

- **Elegante Form** der Textsuche in Dokumenten

- Viele verschiedene Worte in großen Textsammlungen (Internet)
- Leichte Beschreibung der Suchanfrage
- Mühsame Beschreibung eines deterministischen Erkennungsverfahrens

- **Idee: Simultane Verarbeitung** von Alternativen

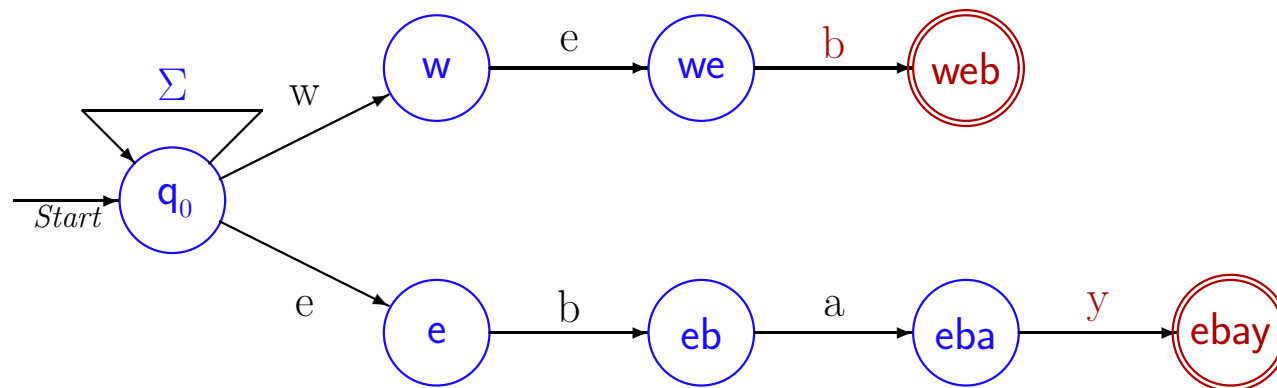
- z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein
- Ein **e** könnte der Anfang von **ebay** sein
- Aber vor den Worten könnte noch etwas anderes stehen

WARUM NICHTDETERMINISTISCHE ENDLICHE AUTOMATEN?

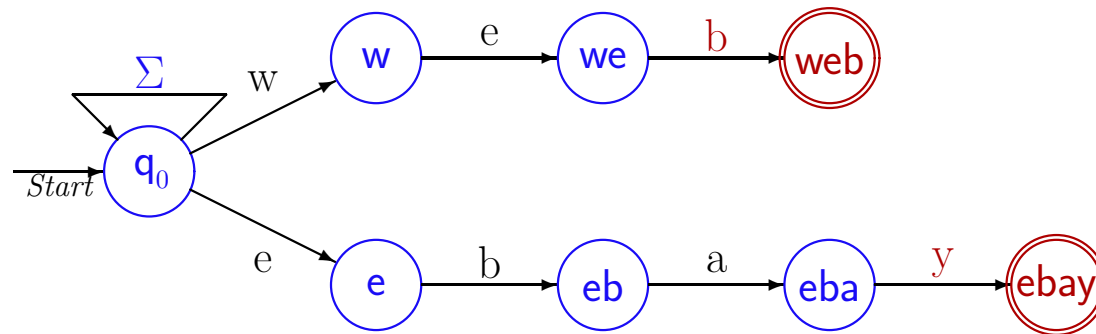
- **Elegante Form** der Textsuche in Dokumenten
 - Viele verschiedene Worte in großen Textsammlungen (Internet)
 - Leichte Beschreibung der Suchanfrage
 - Mühsame Beschreibung eines deterministischen Erkennungsverfahrens
- **Idee: Simultane Verarbeitung** von Alternativen
 - z.B. Suche nach den Worten **web** und **ebay**



- Ein **w** könnte der Anfang von **web** sein
- Ein **e** könnte der Anfang von **ebay** sein
- Aber vor den Worten könnte noch etwas anderes stehen

Nichtdeterministische Modelle verfolgen alle Möglichkeiten simultan

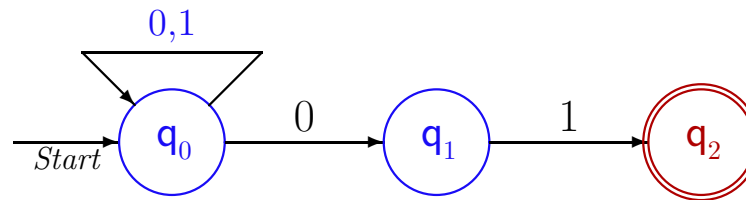
NICHTDETERMINISTISCHE AUTOMATEN – PRÄZISIERT



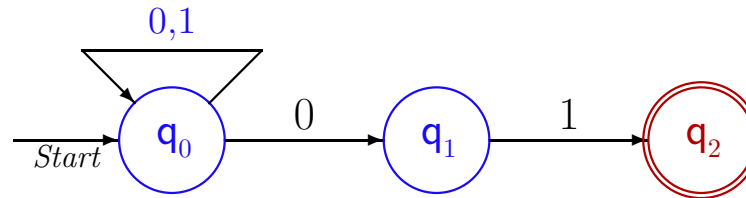
Ein **Nichtdeterministischer Endlicher Automat (NEA)** ist ein 5-Tupel $A = (Q, \Sigma, \delta, q_0, F)$ mit

- Q nichtleere endliche **Zustandsmenge**
- Σ **Eingabealphabet**
- $\delta: Q \times \Sigma \rightarrow \mathcal{P}(Q)$ **Zustandsüberföhrungsfunktion**
 - $\delta(q', a)$ ist (möglicherweise leere) Menge von Zuständen
 - $(\mathcal{P}(Q) = \{S \mid S \subseteq Q\}$ Potenzmenge von Q)
- $q_0 \in Q$ **Startzustand**
- $F \subseteq Q$ Menge von **akzeptierenden** (finalen) **Zuständen**

Erkenne Strings, die mit 01 enden

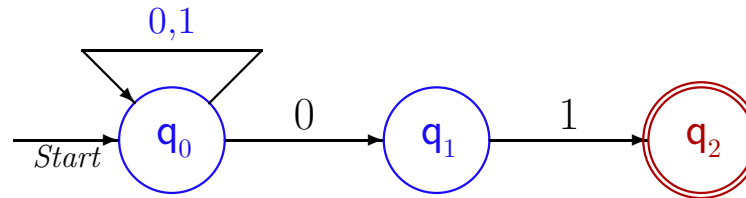


Erkenne Strings, die mit 01 enden



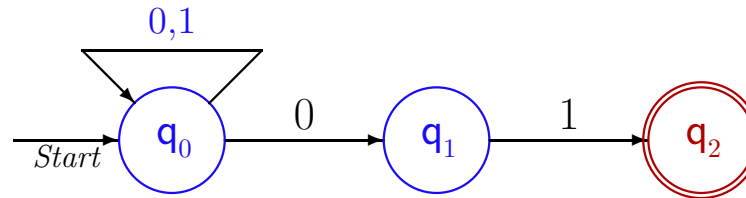
- Jedes Teilwort kann in q_0 bleiben

Erkenne Strings, die mit 01 enden



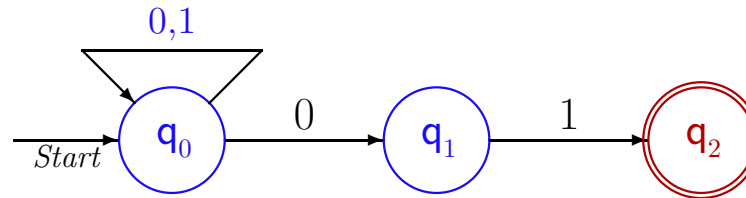
- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen

Erkenne Strings, die mit 01 enden



- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
- In q_2 muß das Wort abgearbeitet sein

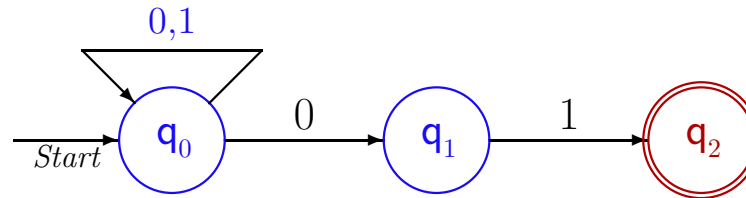
Erkenne Strings, die mit 01 enden



- Jedes Teilwort kann in q_0 bleiben
 - Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
 - Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
 - In q_2 muß das Wort abgearbeitet sein
- Abarbeitung von 00101

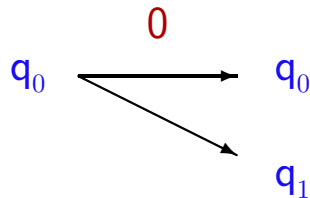
q_0

Erkenne Strings, die mit 01 enden

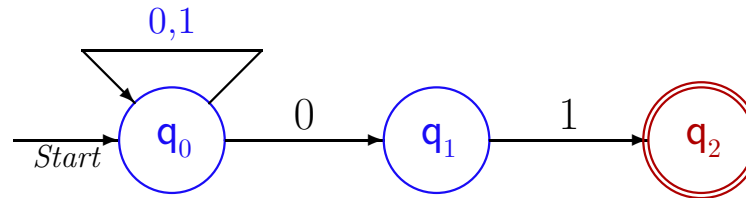


- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
- In q_2 muß das Wort abgearbeitet sein

• Abarbeitung von 00101

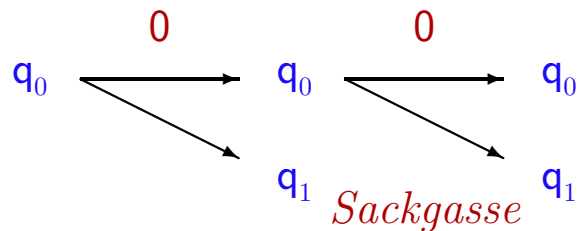


Erkenne Strings, die mit 01 enden

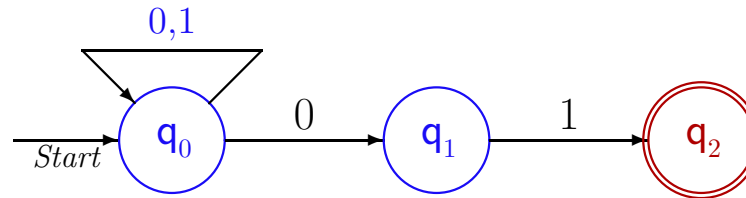


- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
- In q_2 muß das Wort abgearbeitet sein

• Abarbeitung von 00101

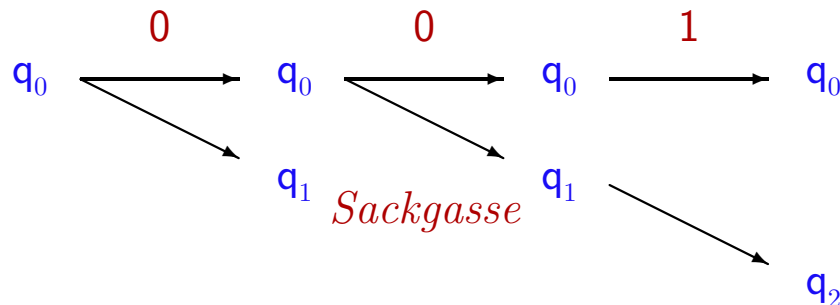


Erkenne Strings, die mit 01 enden

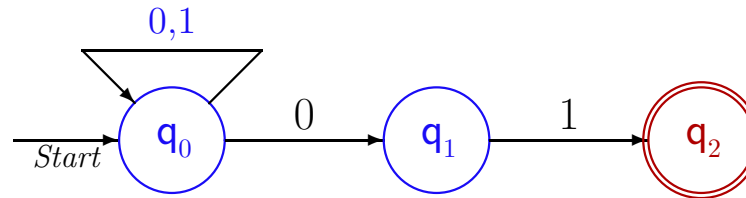


- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
- In q_2 muß das Wort abgearbeitet sein

• Abarbeitung von 00101

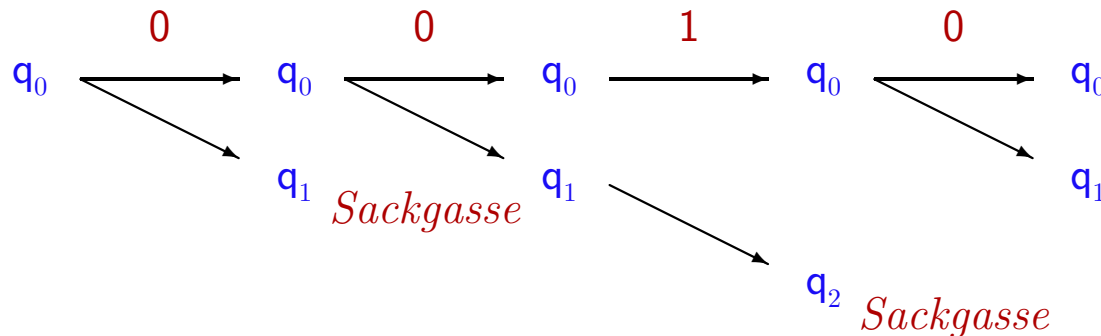


Erkenne Strings, die mit 01 enden

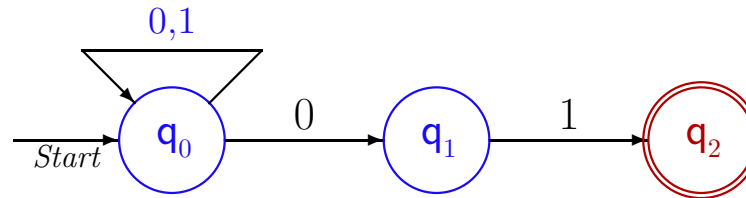


- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
- In q_2 muß das Wort abgearbeitet sein

• Abarbeitung von 00101

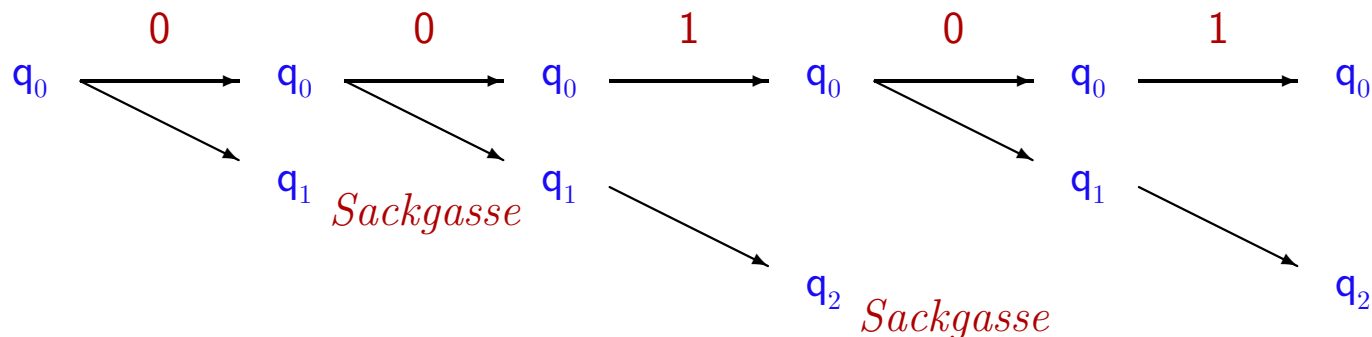


Erkenne Strings, die mit 01 enden

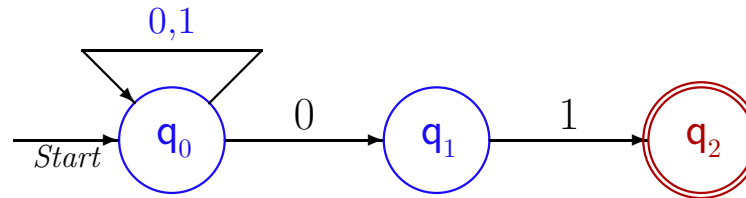


- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
- In q_2 muß das Wort abgearbeitet sein

• Abarbeitung von 00101

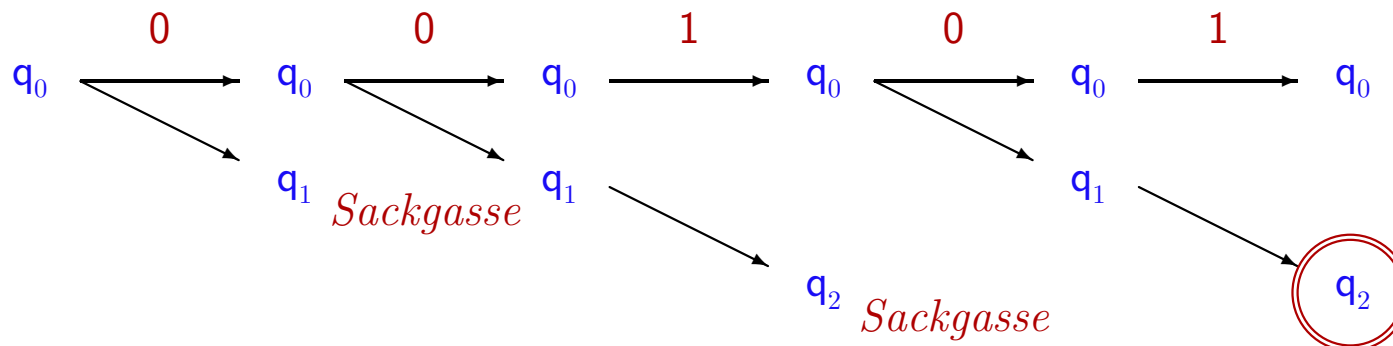


Erkenne Strings, die mit 01 enden



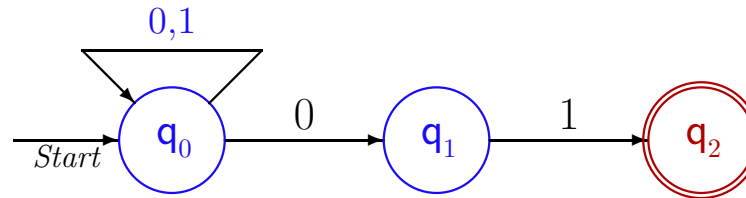
- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
- In q_2 muß das Wort abgearbeitet sein

• Abarbeitung von 00101



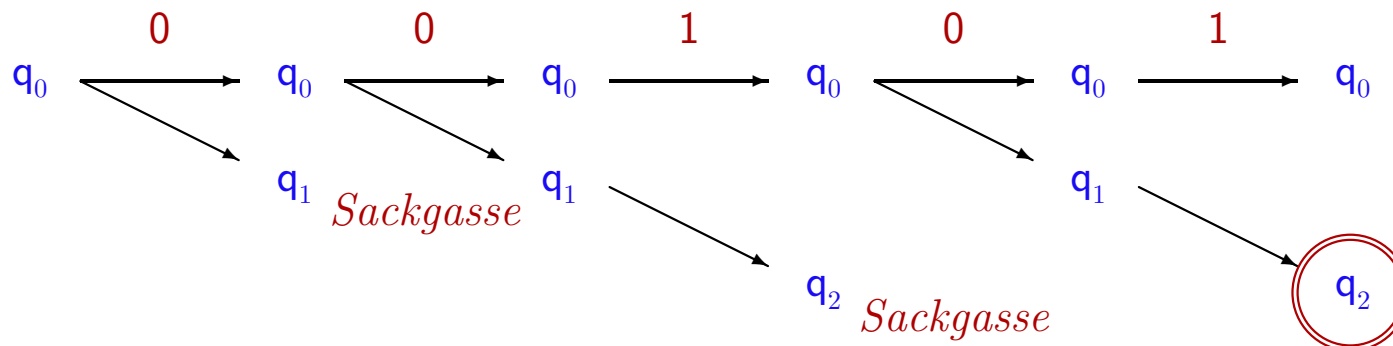
ARBEITSWEISE VON NEAs

Erkenne Strings, die mit 01 enden



- Jedes Teilwort kann in q_0 bleiben
- Ein Teilwort muß mit 0 enden, um nach q_1 zu führen
- Ein Teilwort muß mit 01 enden, um nach q_2 zu führen
- In q_2 muß das Wort abgearbeitet sein

• Abarbeitung von 00101



Ein Abarbeitungsweg führt zu einem akzeptierenden Zustand

- **Erweiterte Überföhrungsfunktion $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$**
 - Schrittweise Abarbeitung der Eingabe mit δ von links nach rechts
 - Aufsammeln **aller** bei der Abarbeitung erreichbarer Zustände

- **Erweiterte Überföhrungsfunktion $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$**
 - Schrittweise Abarbeitung der Eingabe mit δ von links nach rechts
 - Aufsammeln **aller** bei der Abarbeitung erreichbarer Zustände
 - Induktive Definition

$$\hat{\delta}(q, w) = \begin{cases} \{q\} & \text{falls } w = \epsilon, \\ \bigcup_{q' \in \hat{\delta}(q, v)} \delta(q', a) & \text{falls } w = v a \text{ für } v \in \Sigma^*, a \in \Sigma \end{cases}$$

- **Erweiterte Überföhrungsfunktion $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$**

- Schrittweise Abarbeitung der Eingabe mit δ von links nach rechts
- Aufsammeln **aller** bei der Abarbeitung erreichbarer Zustände
- Induktive Definition

$$\hat{\delta}(q, w) = \begin{cases} \{q\} & \text{falls } w = \epsilon, \\ \bigcup_{q' \in \hat{\delta}(q, v)} \delta(q', a) & \text{falls } w = v a \text{ für } v \in \Sigma^*, a \in \Sigma \end{cases}$$

- **Von A akzeptierte Sprache**

- Menge der Eingaben w für die $\hat{\delta}(q_0, w)$ einen akzeptierenden Zustand enthält

- **Erweiterte Überföhrungsfunktion $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$**

- Schrittweise Abarbeitung der Eingabe mit δ von links nach rechts
- Aufsammeln **aller** bei der Abarbeitung erreichbarer Zustände
- Induktive Definition

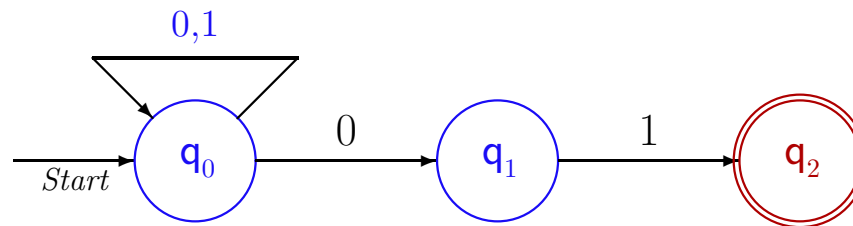
$$\hat{\delta}(q, w) = \begin{cases} \{q\} & \text{falls } w = \epsilon, \\ \bigcup_{q' \in \hat{\delta}(q, v)} \delta(q', a) & \text{falls } w = v a \text{ f\u00fcr } v \in \Sigma^*, a \in \Sigma \end{cases}$$

- **Von A akzeptierte Sprache**

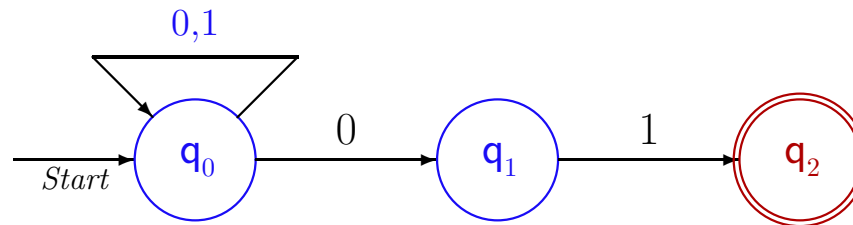
- Menge der Eingaben w f\u00fcr die $\hat{\delta}(q_0, w)$ einen akzeptierenden Zustand enth\u00e4lt

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

ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL

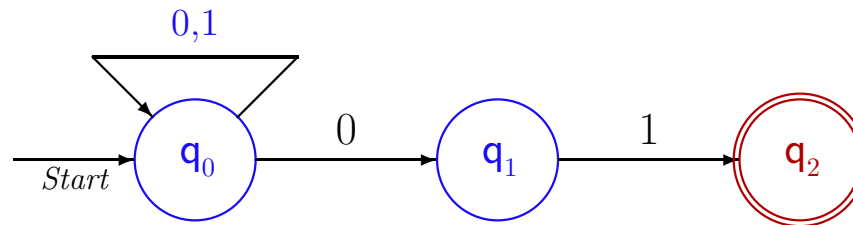


ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL



- Abarbeitung von 00101

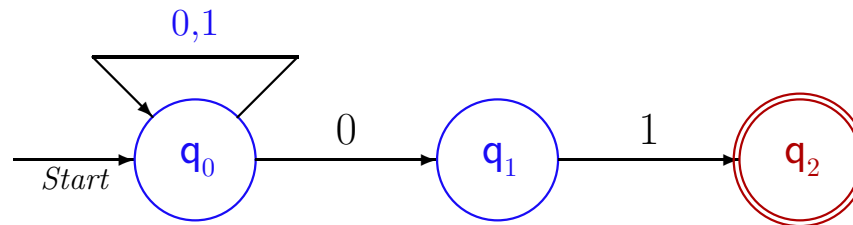
ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL



- Abarbeitung von 00101

- $\hat{\delta}(q_0, \epsilon) = \{q_0\}$

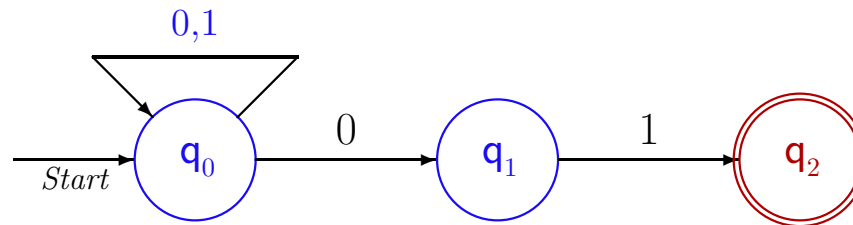
ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL



- Abarbeitung von 00101

- $\hat{\delta}(q_0, \epsilon) = \{q_0\}$
- $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$

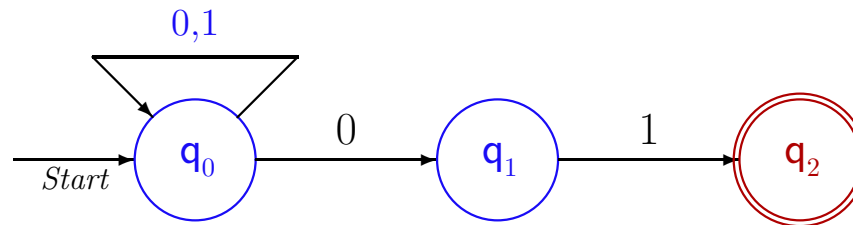
ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL



● Abarbeitung von 00101

- $\hat{\delta}(q_0, \epsilon) = \{q_0\}$
- $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

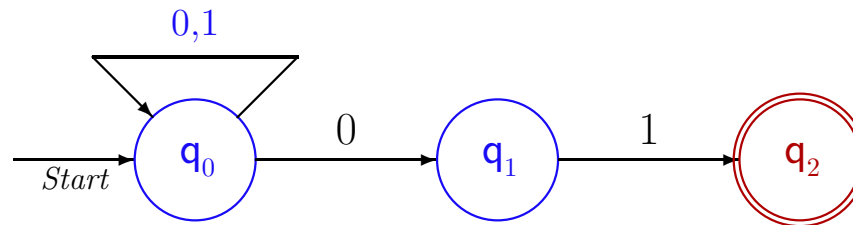
ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL



● Abarbeitung von 00101

- $\hat{\delta}(q_0, \epsilon) = \{q_0\}$
- $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

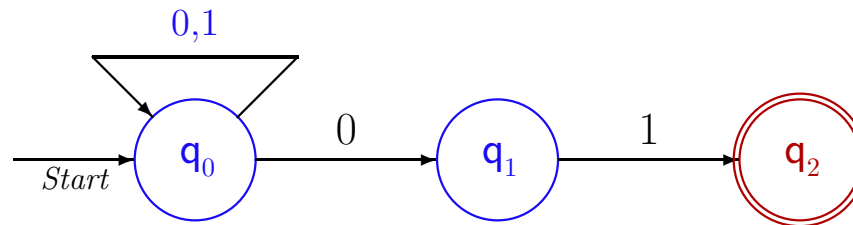
ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL



● Abarbeitung von 00101

- $\hat{\delta}(q_0, \epsilon) = \{q_0\}$
- $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$
- $\hat{\delta}(q_0, 0010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

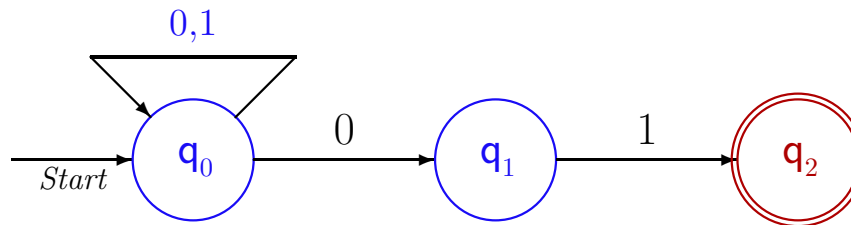
ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL



● Abarbeitung von 00101

- $\hat{\delta}(q_0, \epsilon) = \{q_0\}$
- $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$
- $\hat{\delta}(q_0, 0010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 00101) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

ERWEITERTE ÜBERFÜHRUNGSFUNKTION AM BEISPIEL

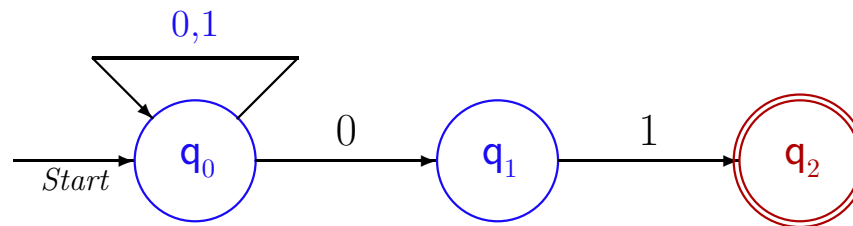


● Abarbeitung von 00101

- $\hat{\delta}(q_0, \epsilon) = \{q_0\}$
- $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$
- $\hat{\delta}(q_0, 0010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
- $\hat{\delta}(q_0, 00101) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

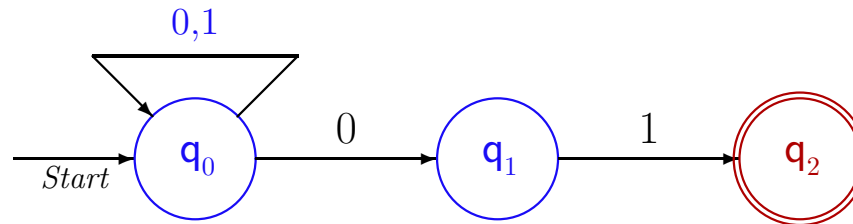
$$\hat{\delta}(q_0, 00101) \cap F = \{q_2\}$$

NACHWEIS DER ERKANNTEN SPRACHE



$$L(A) = \{w \in \{0,1\}^* \mid w \text{ endet mit } 01\}$$

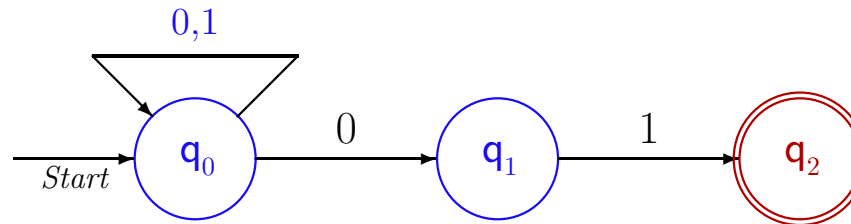
NACHWEIS DER ERKANNTEN SPRACHE



$$L(A) = \{w \in \{0, 1\}^* \mid w \text{ endet mit } 01\}$$

- Zeige durch **simultane Induktion** für alle $w \in \{0, 1\}^*$
 - a) $q_0 \in \hat{\delta}(q_0, w)$
 - b) $q_1 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 0 endet
 - c) $q_2 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 01 endet

NACHWEIS DER ERKANNTEN SPRACHE



$$L(A) = \{w \in \{0, 1\}^* \mid w \text{ endet mit } 01\}$$

● Zeige durch **simultane Induktion** für alle $w \in \{0, 1\}^*$

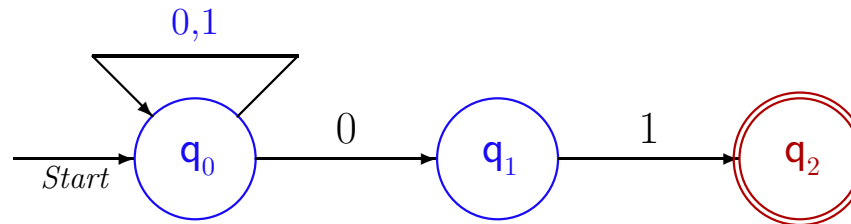
a) $q_0 \in \hat{\delta}(q_0, w)$

b) $q_1 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 0 endet

c) $q_2 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 01 endet

Es folgt $w \in L(A) \Leftrightarrow \hat{\delta}(q_0, w) \cap \{q_2\} \neq \emptyset \Leftrightarrow w$ endet mit 01

NACHWEIS DER ERKANNTEN SPRACHE



$$L(A) = \{w \in \{0, 1\}^* \mid w \text{ endet mit } 01\}$$

● Zeige durch **simultane Induktion** für alle $w \in \{0, 1\}^*$

a) $q_0 \in \hat{\delta}(q_0, w)$

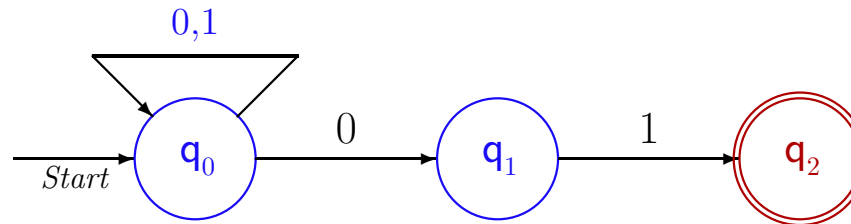
b) $q_1 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 0 endet

c) $q_2 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 01 endet

Es folgt $w \in L(A) \Leftrightarrow \hat{\delta}(q_0, w) \cap \{q_2\} \neq \emptyset \Leftrightarrow w$ endet mit 01

● Induktionsanfang $w = \epsilon$

NACHWEIS DER ERKANNTEN SPRACHE



$$L(A) = \{w \in \{0, 1\}^* \mid w \text{ endet mit } 01\}$$

● Zeige durch **simultane Induktion** für alle $w \in \{0, 1\}^*$

a) $q_0 \in \hat{\delta}(q_0, w)$

b) $q_1 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 0 endet

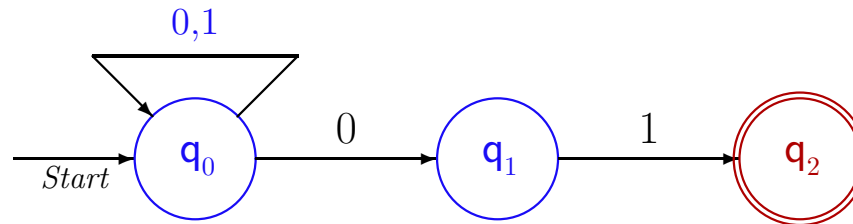
c) $q_2 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 01 endet

Es folgt $w \in L(A) \Leftrightarrow \hat{\delta}(q_0, w) \cap \{q_2\} \neq \emptyset \Leftrightarrow w$ endet mit 01

● Induktionsanfang $w = \epsilon$

– Per Definition ist $\hat{\delta}(q_0, \epsilon) = \{q_0\}$. Also gilt Aussage a)

NACHWEIS DER ERKANNTEN SPRACHE



$$L(A) = \{w \in \{0, 1\}^* \mid w \text{ endet mit } 01\}$$

● Zeige durch **simultane Induktion** für alle $w \in \{0, 1\}^*$

a) $q_0 \in \hat{\delta}(q_0, w)$

b) $q_1 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 0 endet

c) $q_2 \in \hat{\delta}(q_0, w)$ genau dann, wenn w mit 01 endet

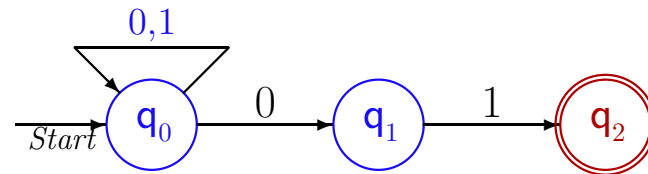
Es folgt $w \in L(A) \Leftrightarrow \hat{\delta}(q_0, w) \cap \{q_2\} \neq \emptyset \Leftrightarrow w$ endet mit 01

● **Induktionsanfang** $w = \epsilon$

– Per Definition ist $\hat{\delta}(q_0, \epsilon) = \{q_0\}$. Also gilt Aussage a)

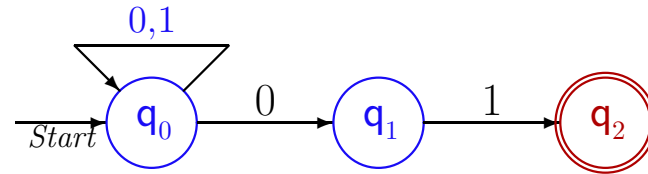
– w endet weder mit 0 noch mit 01. Aussagen b) und c) gelten trivialerweise

NACHWEIS DER ERKANNTEN SPRACHE II



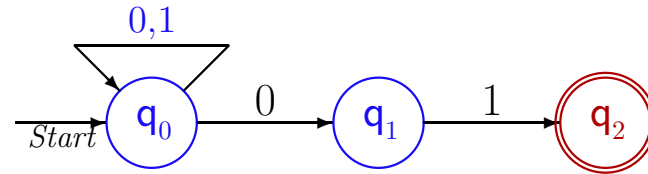
- Induktionsschritt: $w = va$ für $v \in \{0, 1\}^*$, $a \in \{0, 1\}$

NACHWEIS DER ERKANNTEN SPRACHE II



- **Induktionsschritt:** $w = va$ für $v \in \{0, 1\}^*$, $a \in \{0, 1\}$
 - Die Aussagen a), b), und c) seien für v gültig

NACHWEIS DER ERKANNTEN SPRACHE II

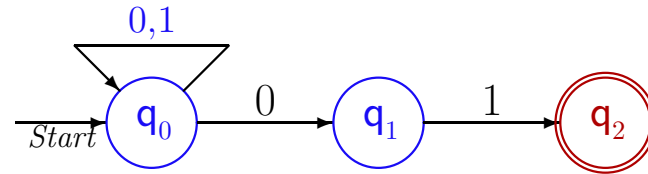


- **Induktionsschritt:** $w = va$ für $v \in \{0, 1\}^*$, $a \in \{0, 1\}$

- Die Aussagen a), b), und c) seien für v gültig

- a) Wegen $q_0 \in \hat{\delta}(q_0, v)$ und $q_0 \in \delta(q_0, a)$ für $a \in \{0, 1\}$ folgt $q_0 \in \hat{\delta}(q_0, w)$

NACHWEIS DER ERKANNTEN SPRACHE II



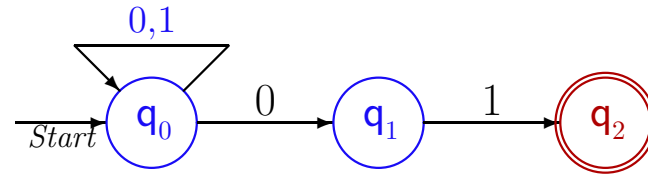
- **Induktionsschritt:** $w = va$ für $v \in \{0, 1\}^*$, $a \in \{0, 1\}$

- Die Aussagen a), b), und c) seien für v gültig

- a) Wegen $q_0 \in \hat{\delta}(q_0, v)$ und $q_0 \in \delta(q_0, a)$ für $a \in \{0, 1\}$ folgt $q_0 \in \hat{\delta}(q_0, w)$

- b) Sei $q_1 \in \hat{\delta}(q_0, w)$. Wegen $q_1 \in \delta(q, a) \Leftrightarrow q = q_0 \wedge a = 0$ muß w mit 0 enden

NACHWEIS DER ERKANNTEN SPRACHE II



- **Induktionsschritt:** $w = va$ für $v \in \{0, 1\}^*$, $a \in \{0, 1\}$

- Die Aussagen a), b), und c) seien für v gültig

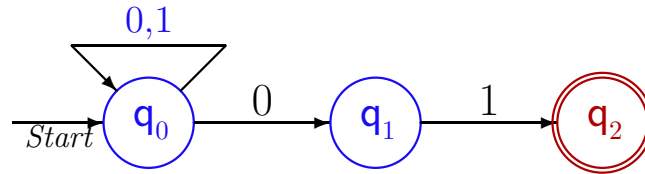
- a) Wegen $q_0 \in \hat{\delta}(q_0, v)$ und $q_0 \in \delta(q_0, a)$ für $a \in \{0, 1\}$ folgt $q_0 \in \hat{\delta}(q_0, w)$

- b) Sei $q_1 \in \hat{\delta}(q_0, w)$. Wegen $q_1 \in \delta(q, a) \Leftrightarrow q=q_0 \wedge a=0$ muß w mit 0 enden

Wenn umgekehrt w mit 0 endet, dann ist $a=0$.

Wegen $q_0 \in \hat{\delta}(q_0, v)$ und $q_1 \in \delta(q_0, a)$ folgt $q_1 \in \hat{\delta}(q_0, w)$

NACHWEIS DER ERKANNTEN SPRACHE II



● Induktionsschritt: $w = va$ für $v \in \{0, 1\}^*$, $a \in \{0, 1\}$

– Die Aussagen a), b), und c) seien für v gültig

a) Wegen $q_0 \in \hat{\delta}(q_0, v)$ und $q_0 \in \delta(q_0, a)$ für $a \in \{0, 1\}$ folgt $q_0 \in \hat{\delta}(q_0, w)$

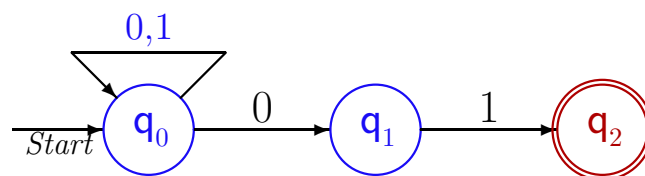
b) Sei $q_1 \in \hat{\delta}(q_0, w)$. Wegen $q_1 \in \delta(q, a) \Leftrightarrow q = q_0 \wedge a = 0$ muß w mit 0 enden

Wenn umgekehrt w mit 0 endet, dann ist $a = 0$.

Wegen $q_0 \in \hat{\delta}(q_0, v)$ und $q_1 \in \delta(q_0, a)$ folgt $q_1 \in \hat{\delta}(q_0, w)$

c) Sei $q_2 \in \hat{\delta}(q_0, w)$. Wegen $q_2 \in \delta(q, a) \Leftrightarrow q = q_1 \wedge a = 1$ muß w mit 1 enden und $q_1 \in \hat{\delta}(q_0, v)$ gelten. Wegen b) für v endet v mit 0, also w mit 01

NACHWEIS DER ERKANNTEN SPRACHE II



● Induktionsschritt: $w = va$ für $v \in \{0, 1\}^*$, $a \in \{0, 1\}$

– Die Aussagen a), b), und c) seien für v gültig

a) Wegen $q_0 \in \hat{\delta}(q_0, v)$ und $q_0 \in \delta(q_0, a)$ für $a \in \{0, 1\}$ folgt $q_0 \in \hat{\delta}(q_0, w)$

b) Sei $q_1 \in \hat{\delta}(q_0, w)$. Wegen $q_1 \in \delta(q, a) \Leftrightarrow q = q_0 \wedge a = 0$ muß w mit 0 enden

Wenn umgekehrt w mit 0 endet, dann ist $a = 0$.

Wegen $q_0 \in \hat{\delta}(q_0, v)$ und $q_1 \in \delta(q_0, a)$ folgt $q_1 \in \hat{\delta}(q_0, w)$

c) Sei $q_2 \in \hat{\delta}(q_0, w)$. Wegen $q_2 \in \delta(q, a) \Leftrightarrow q = q_1 \wedge a = 1$ muß w mit 1 enden und $q_1 \in \hat{\delta}(q_0, v)$ gelten. Wegen b) für v endet v mit 0, also w mit 01

Wenn umgekehrt w mit 01 endet, dann ist $a = 1$ und v endet mit 0.

Wegen $q_1 \in \hat{\delta}(q_0, v)$ nach b) und $q_2 \in \delta(q_1, a)$ folgt $q_2 \in \hat{\delta}(q_0, w)$

- **Nichtdeterministische Automaten sind flexibler**
 - Man muß sich nicht auf eine genaue Verarbeitungsfolge festlegen

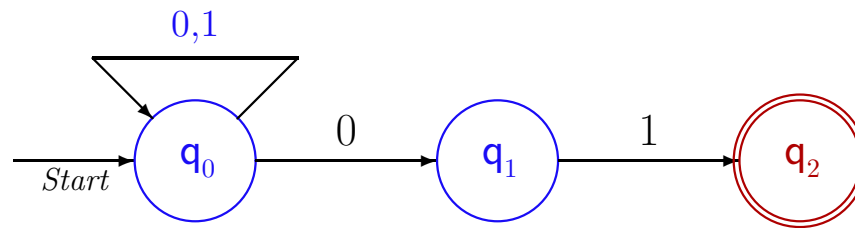
- **Nichtdeterministische Automaten sind flexibler**
 - Man muß sich nicht auf eine genaue Verarbeitungsfolge festlegen
- **DEAs sind genauso ausdruckstark wie NEAs**
 - Man kann Mengen von NEA-Zuständen als DEA Zustände codieren
 - Man kann mengenwertige Zustandsüberföhrungsfunktionen codieren

- **Nichtdeterministische Automaten sind flexibler**
 - Man muß sich nicht auf eine genaue Verarbeitungsfolge festlegen
- **DEAs sind genauso ausdruckstark wie NEAs**
 - Man kann Mengen von NEA-Zuständen als DEA Zustände codieren
 - Man kann mengenwertige Zustandsüberföhrungsfunktionen codieren
- **(Potenzmengen- oder) Teilmengenkonstruktion**
 - Sei $A_N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ ein nichtdeterministischer Automat
 - Konstruiere äquivalenten DEA $A_D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ mit
 - $Q_D = \mathcal{P}(Q_N)$
 - $F_D = \{S \in Q_D \mid S \cap F_N \neq \emptyset\}$
 - $\delta_D(S, a) = \bigcup_{q \in S} \delta_N(q, a)$

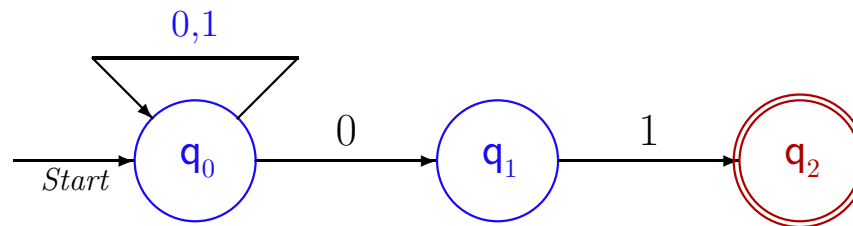
- **Nichtdeterministische Automaten sind flexibler**
 - Man muß sich nicht auf eine genaue Verarbeitungsfolge festlegen
- **DEAs sind genauso ausdruckstark wie NEAs**
 - Man kann Mengen von NEA-Zuständen als DEA Zustände codieren
 - Man kann mengenwertige Zustandsüberföhrungsfunktionen codieren
- **(Potenzmengen- oder) Teilmengenkonstruktion**
 - Sei $A_N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ ein nichtdeterministischer Automat
 - Konstruiere äquivalenten DEA $A_D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ mit
 - $Q_D = \mathcal{P}(Q_N)$
 - $F_D = \{S \in Q_D \mid S \cap F_N \neq \emptyset\}$
 - $\delta_D(S, a) = \bigcup_{q \in S} \delta_N(q, a)$
 - Dann gilt $L(A_D) = L(A_N)$

- **Nichtdeterministische Automaten sind flexibler**
 - Man muß sich nicht auf eine genaue Verarbeitungsfolge festlegen
- **DEAs sind genauso ausdruckstark wie NEAs**
 - Man kann Mengen von NEA-Zuständen als DEA Zustände codieren
 - Man kann mengenwertige Zustandsüberföhrungsfunktionen codieren
- **(Potenzmengen- oder) Teilmengenkonstruktion**
 - Sei $A_N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ ein nichtdeterministischer Automat
 - Konstruiere äquivalenten DEA $A_D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ mit
 - $Q_D = \mathcal{P}(Q_N)$
 - $F_D = \{S \in Q_D \mid S \cap F_N \neq \emptyset\}$
 - $\delta_D(S, a) = \bigcup_{q \in S} \delta_N(q, a)$
 - Dann gilt $L(A_D) = L(A_N)$
 - Konstruktion benötigt $2^{|Q_N|}$ Zustände (Optimierung möglich)

TEILMENGENKONSTRUKTION AM BEISPIEL



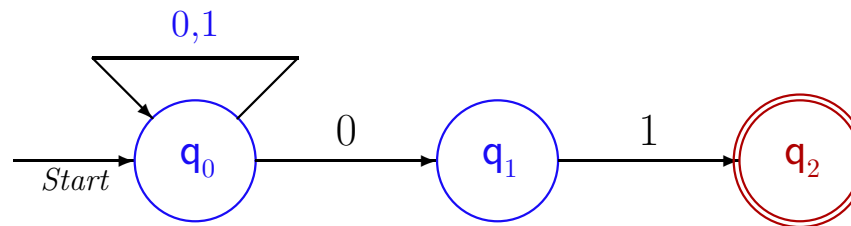
TEILMENGENKONSTRUKTION AM BEISPIEL



Konstruierter deterministischer Automat

	0	1
\emptyset	\emptyset	\emptyset
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	\emptyset	$\{q_2\}$
$* \{q_2\}$	\emptyset	\emptyset
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$* \{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$* \{q_1, q_2\}$	\emptyset	$\{q_2\}$
$* \{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

TEILMENGENKONSTRUKTION AM BEISPIEL



Konstruierter deterministischer Automat

	0	1
\emptyset	\emptyset	\emptyset
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	\emptyset	$\{q_2\}$
$* \{q_2\}$	\emptyset	\emptyset
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$* \{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$* \{q_1, q_2\}$	\emptyset	$\{q_2\}$
$* \{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

Viele überflüssige Zustände (nur 3 von $\{q_0\}$ erreichbar)

- **Iterative Konstruktion** ohne überflüssige Zustände

- **Iterative Konstruktion** ohne überflüssige Zustände
 - Start: $Q_0 := \{\{q_0\}\}$
 - Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$

- **Iterative Konstruktion** ohne überflüssige Zustände
 - Start: $Q_0 := \{\{q_0\}\}$
 - Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
 - Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

- **Iterative Konstruktion** ohne überflüssige Zustände
 - Start: $Q_0 := \{\{q_0\}\}$
 - Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
 - Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$
- **Korrektheit:** $L(A_D) = L(A_N)$

- **Iterative Konstruktion** ohne überflüssige Zustände
 - Start: $Q_0 := \{\{q_0\}\}$
 - Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
 - Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$
- **Korrektheit:** $L(A_D) = L(A_N)$
 - Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$

- **Iterative Konstruktion** ohne überflüssige Zustände

- Start: $Q_0 := \{\{q_0\}\}$
- Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
- Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

- **Korrektheit:** $L(A_D) = L(A_N)$

- Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$
- $w = \epsilon$: $\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}_N(q_0, w)$

- **Iterative Konstruktion** ohne überflüssige Zustände

- Start: $Q_0 := \{\{q_0\}\}$
- Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
- Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

- **Korrektheit:** $L(A_D) = L(A_N)$

- Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$
- $w = \epsilon$: $\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}_N(q_0, w)$
- $w = va$: Es gelte $\hat{\delta}_D(\{q_0\}, v) = \hat{\delta}_N(q_0, v) = \{p_1, \dots, p_k\}$

- **Iterative Konstruktion** ohne überflüssige Zustände

- Start: $Q_0 := \{\{q_0\}\}$
- Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
- Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

- **Korrektheit:** $L(A_D) = L(A_N)$

- Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$
- $w = \epsilon$: $\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}_N(q_0, w)$
- $w = va$: Es gelte $\hat{\delta}_D(\{q_0\}, v) = \hat{\delta}_N(q_0, v) = \{p_1, \dots, p_k\}$
Dann gilt $\hat{\delta}_D(\{q_0\}, w)$

- **Iterative Konstruktion** ohne überflüssige Zustände

- Start: $Q_0 := \{\{q_0\}\}$
- Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
- Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

- **Korrektheit:** $L(A_D) = L(A_N)$

- Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$
- $w = \epsilon$: $\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}_N(q_0, w)$
- $w = va$: Es gelte $\hat{\delta}_D(\{q_0\}, v) = \hat{\delta}_N(q_0, v) = \{p_1, \dots, p_k\}$

Dann gilt $\hat{\delta}_D(\{q_0\}, w)$

$$= \delta_D(\hat{\delta}_D(\{q_0\}, v), a) \quad (\text{Definition } \hat{\delta}_D)$$

- **Iterative Konstruktion** ohne überflüssige Zustände

- Start: $Q_0 := \{\{q_0\}\}$
- Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
- Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

- **Korrektheit:** $L(A_D) = L(A_N)$

- Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$
- $w = \epsilon$: $\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}_N(q_0, w)$
- $w = va$: Es gelte $\hat{\delta}_D(\{q_0\}, v) = \hat{\delta}_N(q_0, v) = \{p_1, \dots, p_k\}$

Dann gilt $\hat{\delta}_D(\{q_0\}, w)$

$$= \delta_D(\hat{\delta}_D(\{q_0\}, v), a)$$

(Definition $\hat{\delta}_D$)

$$= \delta_D(\hat{\delta}_N(q_0, v), a)$$

(Induktionsannahme)

● **Iterative Konstruktion** ohne überflüssige Zustände

- Start: $Q_0 := \{\{q_0\}\}$
- Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
- Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

● **Korrektheit:** $L(A_D) = L(A_N)$

- Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$
- $w = \epsilon$: $\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}_N(q_0, w)$
- $w = va$: Es gelte $\hat{\delta}_D(\{q_0\}, v) = \hat{\delta}_N(q_0, v) = \{p_1, \dots, p_k\}$

Dann gilt $\hat{\delta}_D(\{q_0\}, w)$

$$= \delta_D(\hat{\delta}_D(\{q_0\}, v), a) \quad (\text{Definition } \hat{\delta}_D)$$

$$= \delta_D(\hat{\delta}_N(q_0, v), a) \quad (\text{Induktionsannahme})$$

$$= \bigcup_{q' \in \hat{\delta}_N(q_0, v)} \delta_N(q', a) \quad (\text{Konstruktion von } \delta_D)$$

● **Iterative Konstruktion** ohne überflüssige Zustände

- Start: $Q_0 := \{\{q_0\}\}$
- Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
- Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

● **Korrektheit:** $L(A_D) = L(A_N)$

- Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$
- $w = \epsilon$: $\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}_N(q_0, w)$
- $w = va$: Es gelte $\hat{\delta}_D(\{q_0\}, v) = \hat{\delta}_N(q_0, v) = \{p_1, \dots, p_k\}$

Dann gilt $\hat{\delta}_D(\{q_0\}, w)$

$$= \delta_D(\hat{\delta}_D(\{q_0\}, v), a) \quad (\text{Definition } \hat{\delta}_D)$$

$$= \delta_D(\hat{\delta}_N(q_0, v), a) \quad (\text{Induktionsannahme})$$

$$= \bigcup_{q' \in \hat{\delta}_N(q_0, v)} \delta_N(q', a) \quad (\text{Konstruktion von } \delta_D)$$

$$= \hat{\delta}_N(q_0, w) \quad (\text{Definition } \hat{\delta}_N)$$

- **Iterative Konstruktion** ohne überflüssige Zustände

- Start: $Q_0 := \{\{q_0\}\}$
- Schritt: $Q_{i+1} := Q_i \cup \{\delta_D(S, a) \mid S \in Q_i, a \in \Sigma\}$
- Abschluß: Wenn $Q_{i+1} = Q_i$, dann halte an und setze $Q_D := Q_i$

- **Korrektheit:** $L(A_D) = L(A_N)$

- Zeige durch Induktion: $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ für alle $w \in \Sigma^*$
- $w = \epsilon$: $\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}_N(q_0, w)$
- $w = va$: Es gelte $\hat{\delta}_D(\{q_0\}, v) = \hat{\delta}_N(q_0, v) = \{p_1, \dots, p_k\}$

Dann gilt $\hat{\delta}_D(\{q_0\}, w)$

$$= \delta_D(\hat{\delta}_D(\{q_0\}, v), a) \quad (\text{Definition } \hat{\delta}_D)$$

$$= \delta_D(\hat{\delta}_N(q_0, v), a) \quad (\text{Induktionsannahme})$$

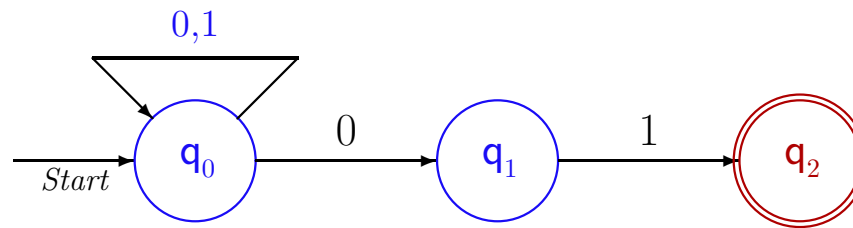
$$= \bigcup_{q' \in \hat{\delta}_N(q_0, v)} \delta_N(q', a) \quad (\text{Konstruktion von } \delta_D)$$

$$= \hat{\delta}_N(q_0, w) \quad (\text{Definition } \hat{\delta}_N)$$

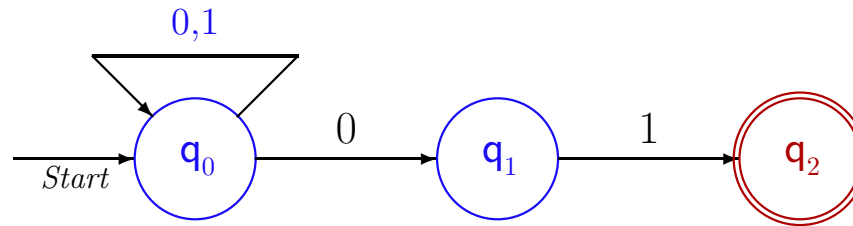
- **NEAs und DEAs akzeptieren dieselben Sprachen**

- Jeder DEA ist als “eindeutiger” NEA beschreibbar

OPTIMIERTE TEILMENGENKONSTRUKTION AM BEISPIEL



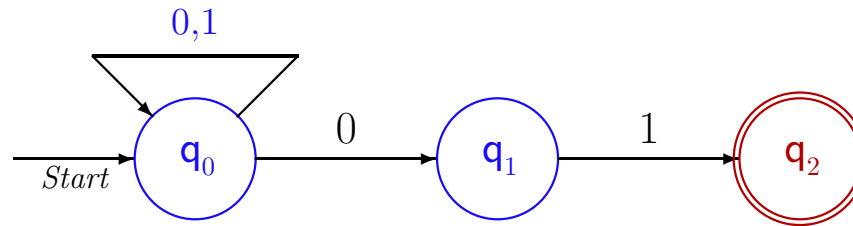
OPTIMIERTE TEILMENGENKONSTRUKTION AM BEISPIEL



- **Konstruktion der Zustandsmengen**

- $Q_0 := \{\{q_0\}\}$

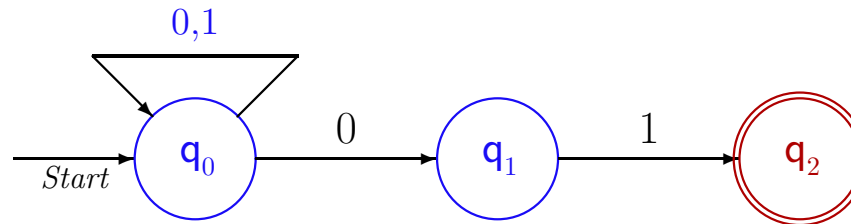
OPTIMIERTE TEILMENGENKONSTRUKTION AM BEISPIEL



● Konstruktion der Zustandsmengen

- $Q_0 := \{\{q_0\}\}$
- $Q_1 := \{\{q_0\}, \{q_0, q_1\}\}$

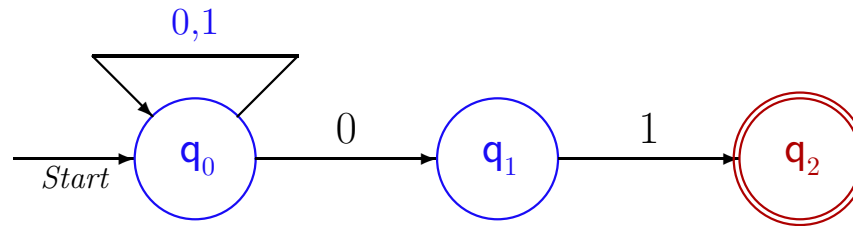
OPTIMIERTE TEILMENGENKONSTRUKTION AM BEISPIEL



● Konstruktion der Zustandsmengen

- $Q_0 := \{\{q_0\}\}$
- $Q_1 := \{\{q_0\}, \{q_0 q_1\}\}$
- $Q_2 := \{\{q_0\}, \{q_0 q_1\}, \{q_0 q_2\}\}$

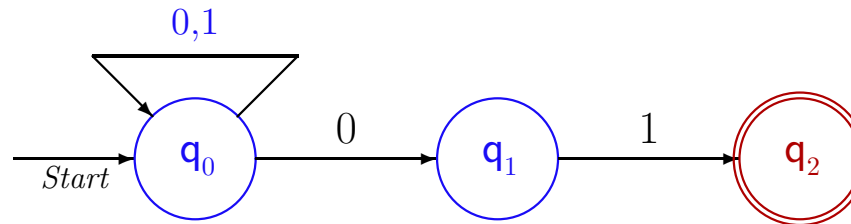
OPTIMIERTE TEILMENGENKONSTRUKTION AM BEISPIEL



● Konstruktion der Zustandsmengen

- $Q_0 := \{\{q_0\}\}$
- $Q_1 := \{\{q_0\}, \{q_0 q_1\}\}$
- $Q_2 := \{\{q_0\}, \{q_0 q_1\}, \{q_0 q_2\}\}$
- $Q_3 = Q_2$

OPTIMIERTE TEILMENGENKONSTRUKTION AM BEISPIEL



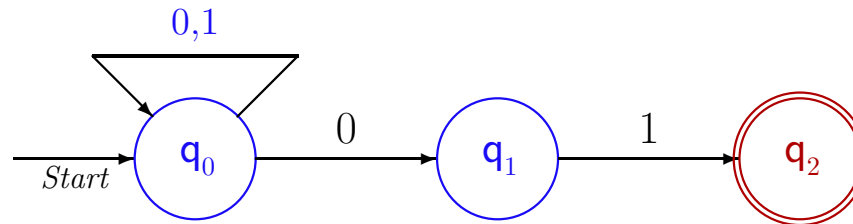
● Konstruktion der Zustandsmengen

- $Q_0 := \{\{q_0\}\}$
- $Q_1 := \{\{q_0\}, \{q_0 q_1\}\}$
- $Q_2 := \{\{q_0\}, \{q_0 q_1\}, \{q_0 q_2\}\}$
- $Q_3 = Q_2$

	0	1
→ $\{q_0\}$	$\{q_0 q_1\}$	$\{q_0\}$
$\{q_0 q_1\}$	$\{q_0 q_1\}$	$\{q_0 q_2\}$
* $\{q_0 q_2\}$	$\{q_0 q_1\}$	$\{q_0\}$

● Reduzierte Überföhrungsfunktion

OPTIMIERTE TEILMENGENKONSTRUKTION AM BEISPIEL



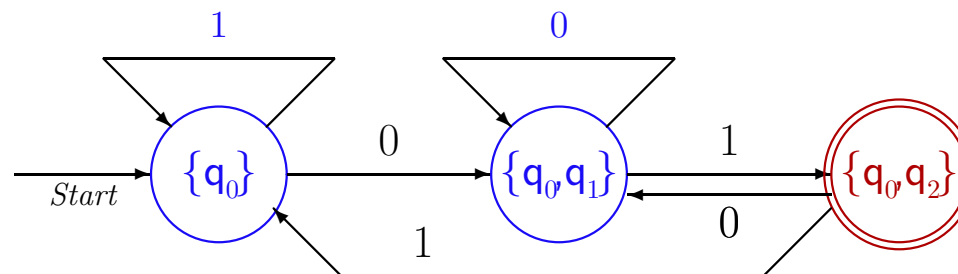
• Konstruktion der Zustandsmengen

- $Q_0 := \{\{q_0\}\}$
- $Q_1 := \{\{q_0\}, \{q_0 q_1\}\}$
- $Q_2 := \{\{q_0\}, \{q_0 q_1\}, \{q_0 q_2\}\}$
- $Q_3 = Q_2$

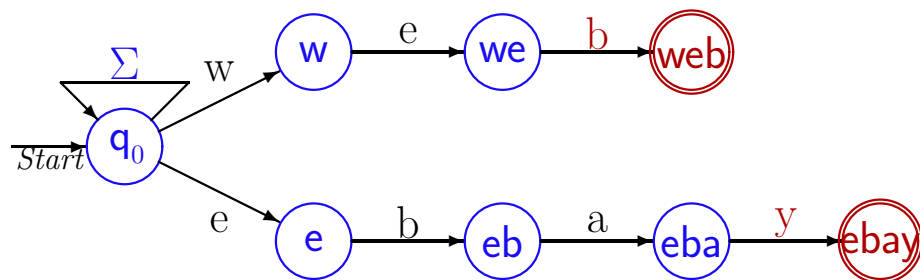
	0	1
→ $\{q_0\}$	$\{q_0 q_1\}$	$\{q_0\}$
$\{q_0 q_1\}$	$\{q_0 q_1\}$	$\{q_0 q_2\}$
* $\{q_0 q_2\}$	$\{q_0 q_1\}$	$\{q_0\}$

• Reduzierte Überföhrungsfunktion

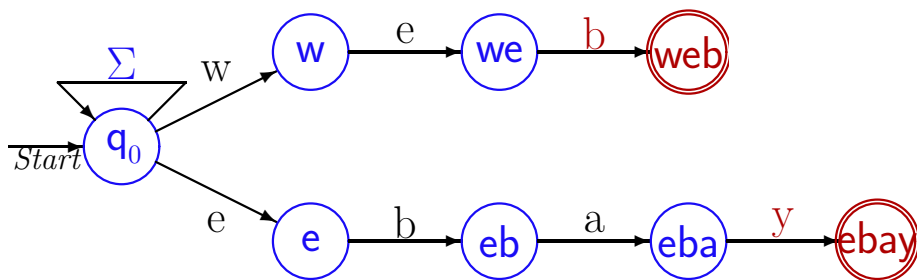
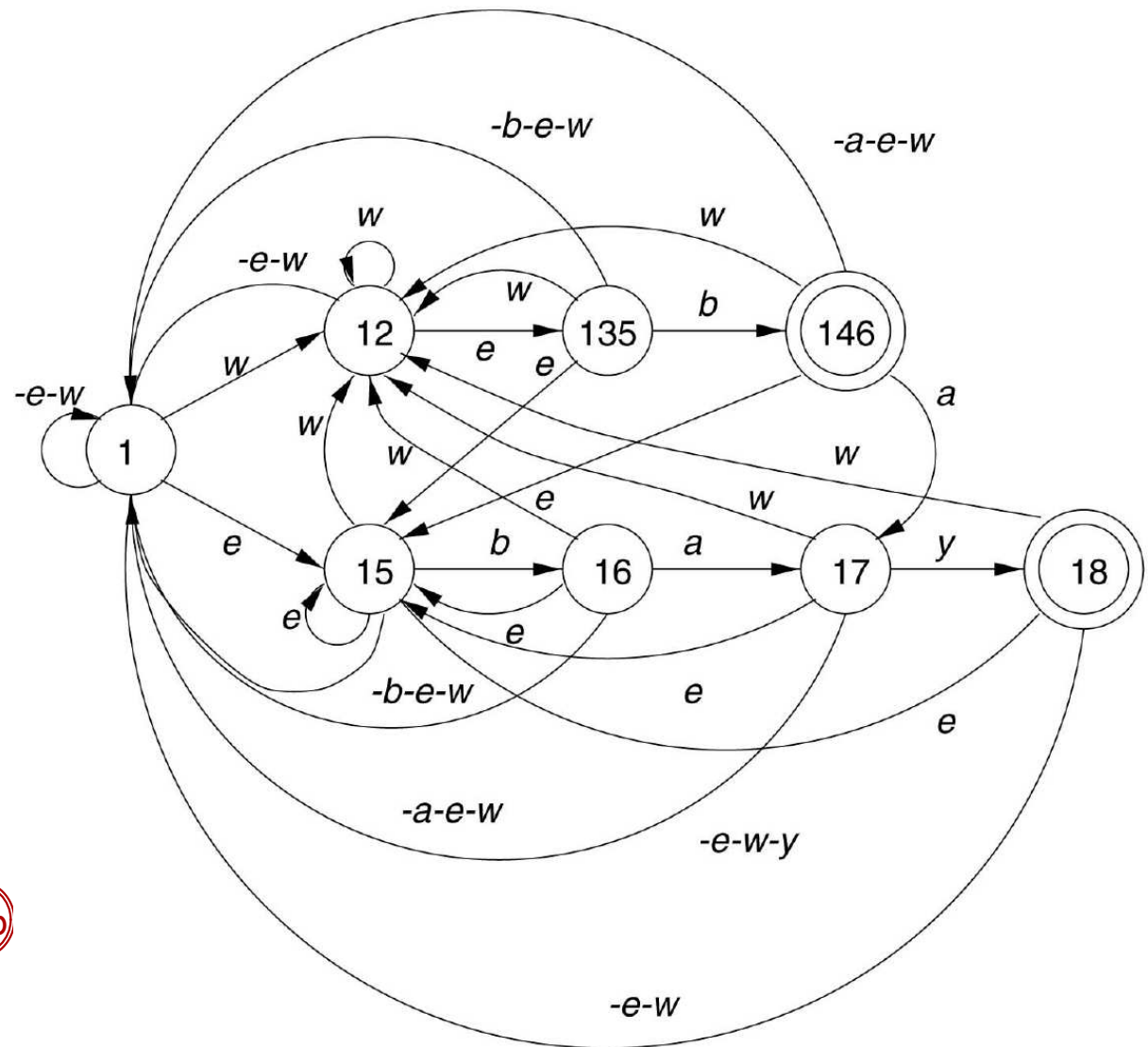
• Resultierender deterministischer Automat



DETERMINISTISCHE AUTOMATEN FÜR TEXTANALYSE



DETERMINISTISCHE AUTOMATEN FÜR TEXTANALYSE



ANALYSE DER OPTIMIERTEN TEILMENGENKONSTRUKTION

- A_D kann so klein sein wie A_N

ANALYSE DER OPTIMIERTEN TEILMENGENKONSTRUKTION

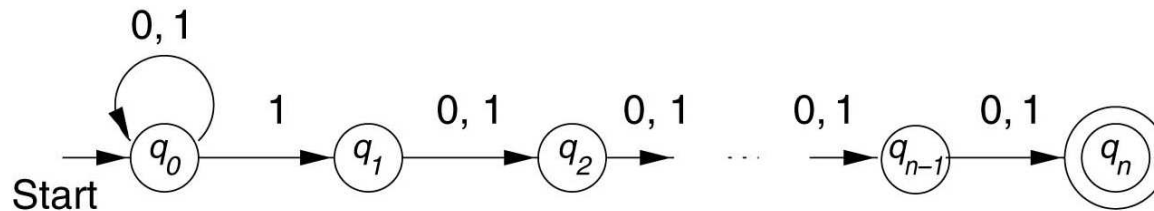
- A_D kann so klein sein wie A_N
 - Nur wenige Teilmengen von Q_N werden wirklich erreicht

ANALYSE DER OPTIMIERTEN TEILMENGENKONSTRUKTION

- A_D kann so klein sein wie A_N
 - Nur wenige Teilmengen von Q_N werden wirklich erreicht
- A_D kann exponentiell größer werden

ANALYSE DER OPTIMIERTEN TEILMENGENKONSTRUKTION

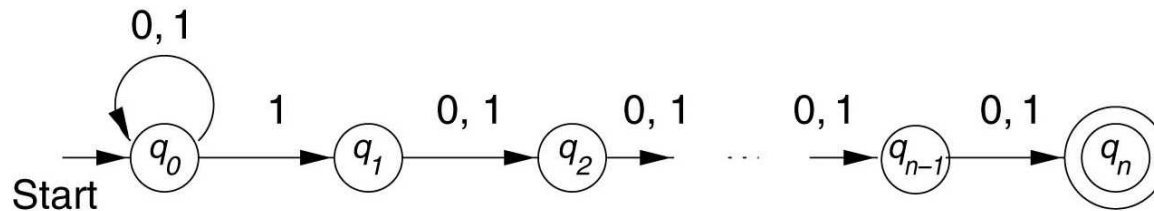
- A_D kann so klein sein wie A_N
 - Nur wenige Teilmengen von Q_N werden wirklich erreicht
- A_D kann exponentiell größer werden



- $L(A_N) = \{w \in \{0, 1\}^* \mid \text{das } n\text{-te Zeichen vor dem Ende ist eine } 1\}$

ANALYSE DER OPTIMIERTEN TEILMENGENKONSTRUKTION

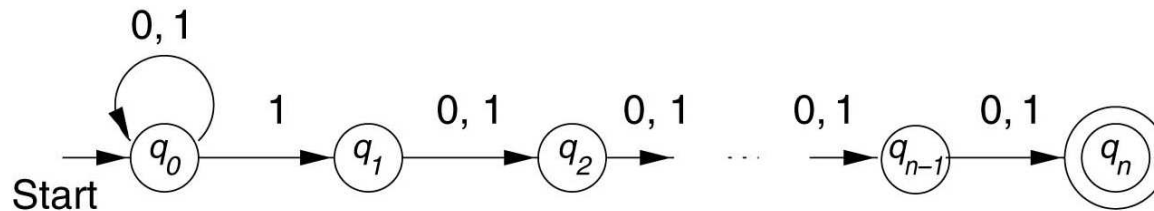
- A_D kann so klein sein wie A_N
 - Nur wenige Teilmengen von Q_N werden wirklich erreicht
- A_D kann exponentiell größer werden



- $L(A_N) = \{w \in \{0, 1\}^* \mid \text{das } n\text{-te Zeichen vor dem Ende ist eine 1}\}$
- Jeder DEA A für $L(A_N)$ benötigt mindestens 2^n Zustände

ANALYSE DER OPTIMIERTEN TEILMENGENKONSTRUKTION

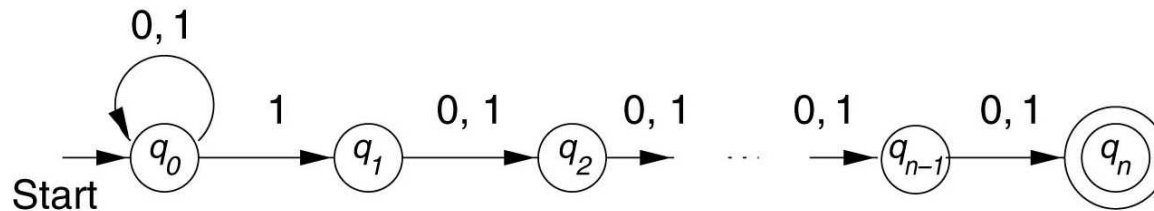
- A_D kann so klein sein wie A_N
 - Nur wenige Teilmengen von Q_N werden wirklich erreicht
- A_D kann exponentiell größer werden



- $L(A_N) = \{w \in \{0, 1\}^* \mid \text{das } n\text{-te Zeichen vor dem Ende ist eine 1}\}$
- **Jeder DEA A für $L(A_N)$ benötigt mindestens 2^n Zustände**
- **Beweis:** Es gibt 2^n Worte der Länge n in $\{0, 1\}^*$

ANALYSE DER OPTIMIERTEN TEILMENGENKONSTRUKTION

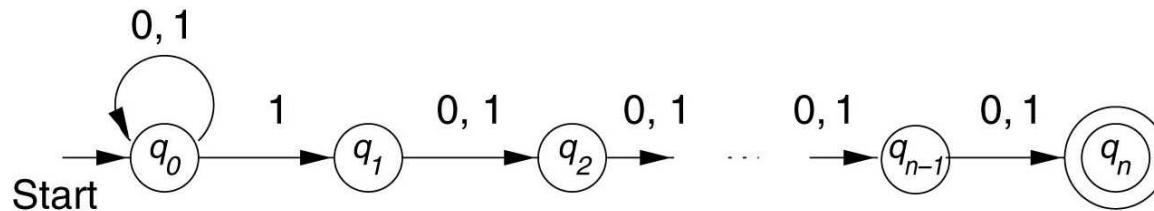
- A_D kann so klein sein wie A_N
 - Nur wenige Teilmengen von Q_N werden wirklich erreicht
- A_D kann exponentiell größer werden



- $L(A_N) = \{w \in \{0, 1\}^* \mid \text{das } n\text{-te Zeichen vor dem Ende ist eine } 1\}$
- **Jeder DEA A für $L(A_N)$ benötigt mindestens 2^n Zustände**
- **Beweis:** Es gibt 2^n Worte der Länge n in $\{0, 1\}^*$
Hat A weniger als 2^n Zustände, so gibt es $w = a_1..a_n$ und $v = b_1..b_n$
mit $w \neq v$ und $\hat{\delta}_A(q_0, w) = \hat{\delta}_A(q_0, v)$ **(Schubfachprinzip)**

ANALYSE DER OPTIMIERTEN TEILMENGENKONSTRUKTION

- A_D kann so klein sein wie A_N
 - Nur wenige Teilmengen von Q_N werden wirklich erreicht
- A_D kann exponentiell größer werden



- $L(A_N) = \{w \in \{0, 1\}^* \mid \text{das } n\text{-te Zeichen vor dem Ende ist eine } 1\}$
- **Jeder DEA A für $L(A_N)$ benötigt mindestens 2^n Zustände**
- **Beweis:** Es gibt 2^n Worte der Länge n in $\{0, 1\}^*$
Hat A weniger als 2^n Zustände, so gibt es $w = a_1..a_n$ und $v = b_1..b_n$
mit $w \neq v$ und $\hat{\delta}_A(q_0, w) = \hat{\delta}_A(q_0, v)$ **(Schubfachprinzip)**
Sei $a_i \neq b_i$. Für $q = \delta_A(q_0, w0^{i-1}) = \delta_A(q_0, v0^{i-1})$ folgt $q \in F$ und $q \notin F$