Automatentheorie

Eduard Mehofer

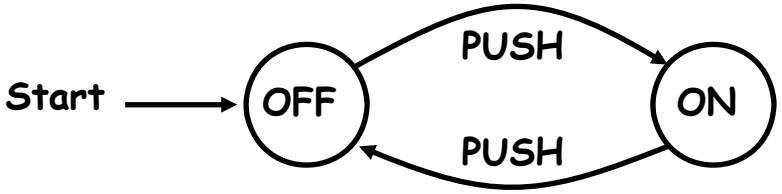
Fakultät für Informatik

Währinger Straße 29

Universität Wien

Motivation

- Automaten sind wichtiges Modell für Textanalysen, Kommunikationsprotokolle, Steuerungstechnik, etc.
- Zentrales Element eines Automaten: Zustand.
- Zustand repräsentiert die Historie eines Systems.



- Insbesondere interessiert uns folgende Fragestellung:

  Ist ein gegebenes Wort $w \in \Sigma^*$ Teil einer Sprache L oder nicht?

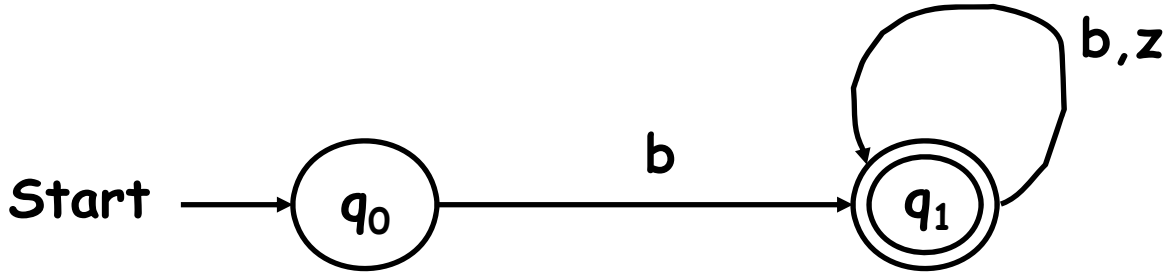
  wobei:

  Eine Sprache L, die durch einen Automaten A definiert wird, ist die Menge aller Worte, die von A akzeptiert werden.

Beispiel intuitiv:

Gesucht ist ein deterministischer endlicher Automat A, der die Sprache **b (b | z)*** akzeptiert (d.h. wenn **b** für Buchstabe steht und **z** für Ziffer, werden Identifier akzeptiert).

Skizze (Übergangsdiagramm, gerichteter Graph, δ):



Beschreibung:

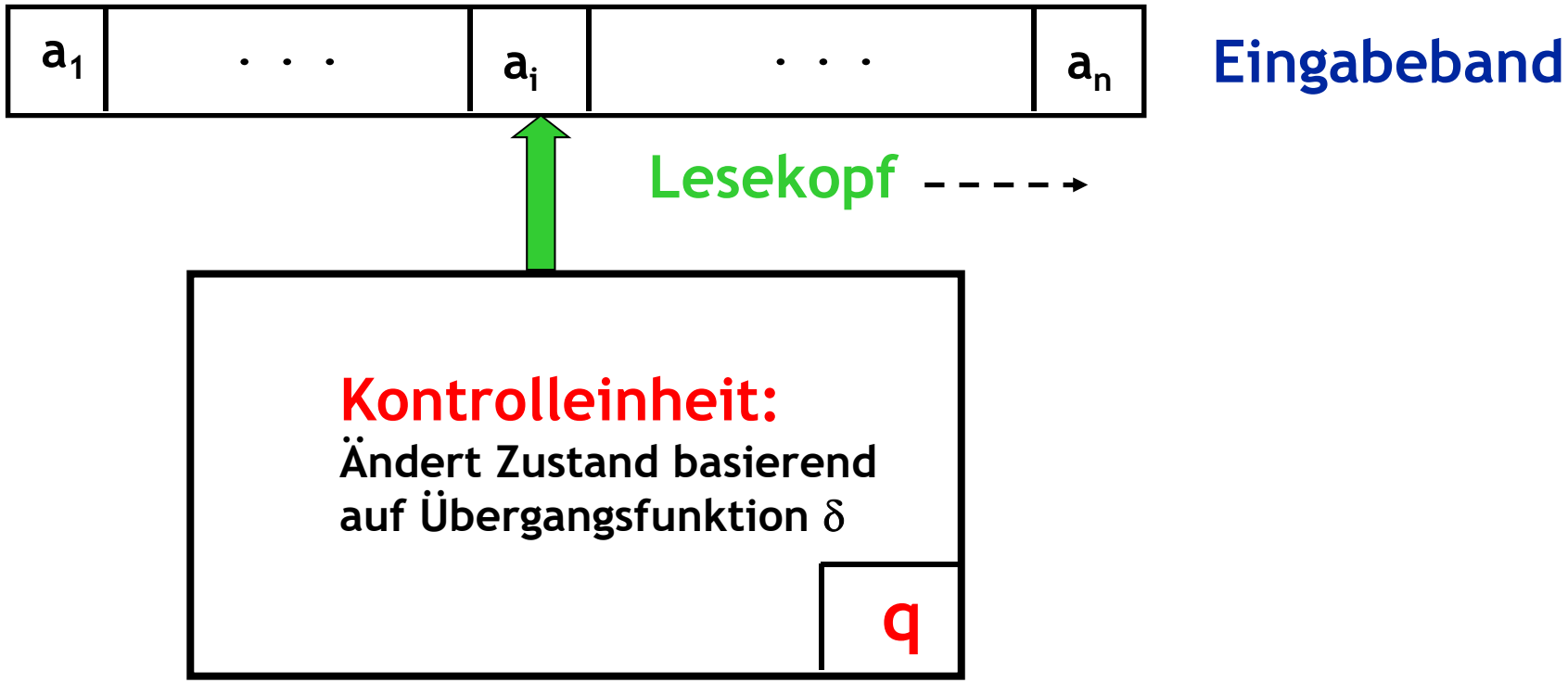
Zustände = $\{q_0, q_1\}$,
Anfangszustand = q_0, Endzustand = $\{q_1\}$,
Eingabealphabet = {b,z},
Graph stellt die Übergänge des Automaten dar.

Die Komponenten eines endlichen Automaten EA

Man kann einen EA $A = (Q, \Sigma, \delta, q_0, F)$ als eine Maschine mit drei Komponenten auffassen: Eingabeband, Lesekopf und Kontrolleinheit:

- Auf dem Eingabeband steht das zu analysierende Wort w.
- Der Lesekopf zeigt zu Beginn auf das erste Zeichen von w und wird bei jedem Schritt um 1 nach rechts gesetzt.
- Die Kontrolleinheit speichert zu jedem Zeitpunkt den Zustand der aktuellen Konfiguration und enthält alle möglichen Folgezustände.

Graphische Darstellung der Komponenten



Endliche Automaten

Definition: Ein (nichtdeterministischer) endlicher Automat (EA) ist ein Quintupel

$$A = (Q, \Sigma, \delta, q_0, F)$$

wobei

- Q: Menge der Zustände (endlich)
- Σ: Eingabealphabet (endlich)
- $q_0 \in Q$: Anfangszustand
- $F \subseteq Q$: Menge der Endzustände (endlich)
- $\delta : Q \times \Sigma \to P(Q)$: Übergangsfunktion ($P$ … *Potenzmenge*)

Definition: Für einen deterministischen endlichen Automaten (DEA) gilt:

$$\text{für alle } q \in Q,\ a \in \Sigma : |\ \delta(q,a)\ | \leq 1.$$

(Für einen DEA läßt sich die Übergangsfunktion als partielle Funktion $\delta : Q \times \Sigma \to Q$ interpretieren.)

Arbeitsweise des deterministischen endlichen Automaten

Gegeben sei ein **DEA** $A = (Q, \Sigma, \delta, q_0, F)$ **mit** $\delta : Q \times \Sigma \rightarrow Q$. Weiters sei $w \in \Sigma^*$ das zu analysierende Wort.

Falls $w = \varepsilon$, dann wird w von A genau dann akzeptiert, wenn $q_0 \in F$. Andernfalls gelte $w = a_1 \ldots a_n$, mit $n \geq 1$ und $a_i \in \Sigma$ $(1 \leq i \leq n)$.

Initialisierung: Der Automat startet in der **Anfangskonfiguration** (q_0, w) und befindet sich nach i Schritten $(1 \leq i \leq n-1)$ in der **Konfiguration** $(q_i, a_{i+1} \ldots a_n)$. **Schritt** i + 1 wird nun wie folgt durchgeführt:

- falls $\delta(q_i, a_{i+1})$ undefiniert:
  - der Automat stoppt und Wort w ist **nicht akzeptiert**.
- falls $\delta(q_i, a_{i+1}) = q_{i+1}$, sind 2 Fälle zu unterscheiden:
  - falls i + 1 = n: (d.h. Wort w bereits vollständig abgearbeitet)
    Der Automat **akzeptiert** w genau dann, wenn $q_{i+1} \in F$.
  - falls i + 1 < n, dann geht der Automat in die Konfiguration $(q_{i+1}, a_{i+2} \ldots a_n)$ über und führt nächsten Schritt i+2 aus.

Darstellung EA: Übergangstabelle

Gegeben sei ein **EA A = (Q, Σ, δ, q_0, F)** mit $Q=\{q_0,\ldots,q_m\}$ und $\Sigma=\{a_1,\ldots,a_n\}$. Die Übergangsfunktion δ kann durch eine **Übergangstabelle** dargestellt werden:

δ	a_1	a_2	...	a_j	...	a_n
q_0						
q_1						
⋮						
q_i				$\delta(q_i,a_j)$		
⋮						
q_m						

Übergangstabelle erstellt basierend auf Übergangsfunktion $\delta(q_i,a_j) = \{\ldots\}$.

Darstellung EA: Übergangsdiagramm

Definition: Das Übergangsdiagramm für einen EA ist ein gerichteter Graph, der wie folgt definiert ist:

- Die Knoten des Graphen entsprechen den Zuständen des EA.
- Bewirkt Symbol **a** einen Übergang von Zustand **q** nach Zustand **p**, dann existiert eine mit **a** markierte Kante vom Knoten **q** nach Knoten **p**.

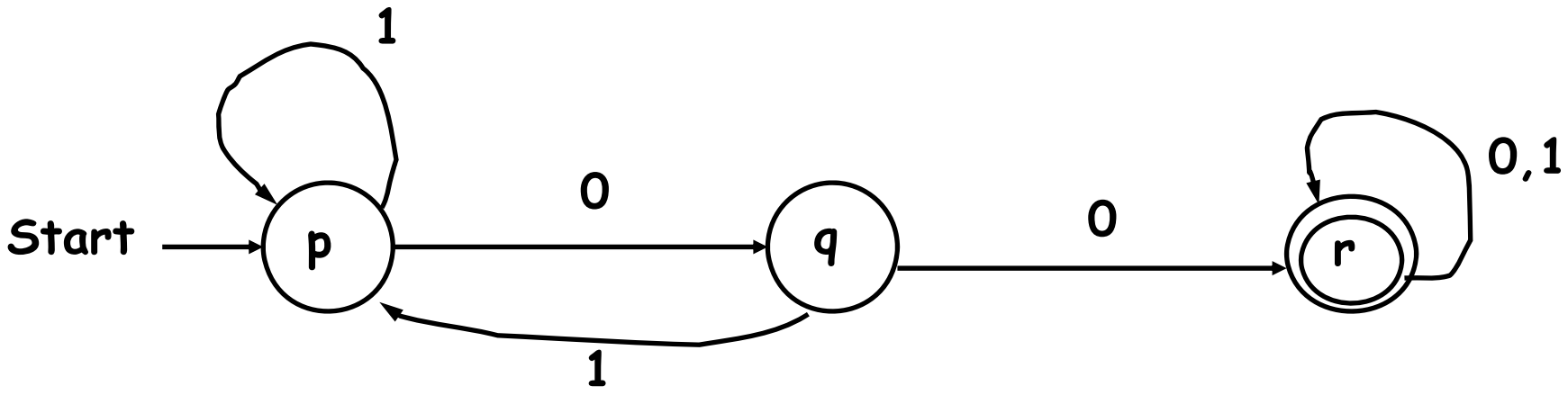
Der EA akzeptiert eine Zeichenkette **x**, wenn das Wort zur Gänze gelesen wurde und ein Endzustand erreicht wurde.

Der Anfangszustand wird durch einen mit ‘**Start**‘ markierten Pfeil angegeben und die Endzustände werden mit einem **doppelten Kreis** markiert.

Beispiel für DEA

A = ({p, q, r}, {0,1}, δ, p, {r})

δ	0	1
p	{q}	{p}
q	{r}	{p}
r	{r}	{r}



L(A) ist die Menge aller Worte, die mindestens zwei aufeinanderfolgende Nullen enthalten:

L(A) = { w | w = x00y mit x, y ∈ Σ* }.

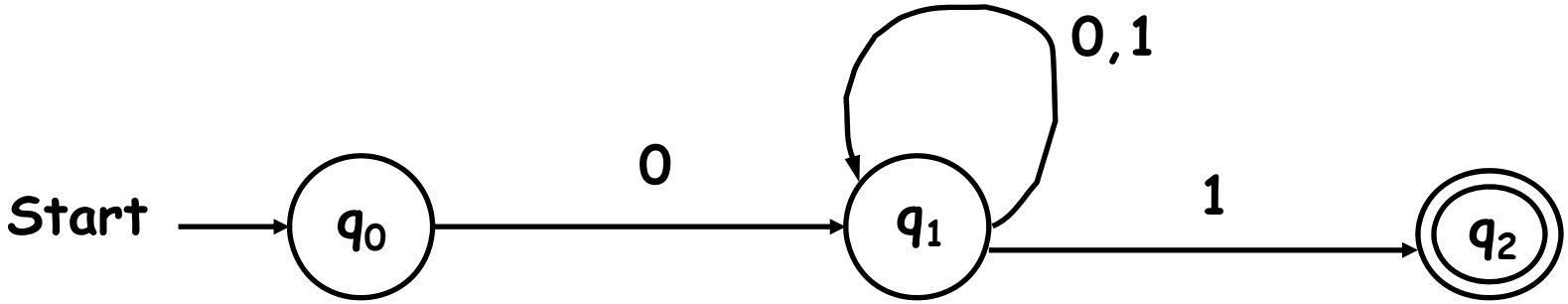
Beispiel für NEA

Sei A = ($\{q_0, q_1, q_2\}$, $\{0, 1\}$, δ, q_0, $\{q_2\}$), mit

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

$\delta(q_1,1) = \{q_1, q_2\}$

$\delta(q,a) = \emptyset$ für alle anderen Paare $(q, a) \in Q \times \Sigma$.



A ist ein **nichtdeterministischer** endlicher Automat und akzeptiert die Sprache **L(A) = { w | w = 0x1, x ∈ {0, 1}* }**.

Erweiterung der Übergangsfunktion von δ zu δ^*

Sei $A = (Q, \Sigma, \delta, q_0, F)$ ein beliebiger endlicher Automat. Die Übergangsfunktion δ ist definiert als totale Funktion

$$\delta : Q \times \Sigma \to \mathcal{P}(Q).$$

In der folgenden Definition wird die Übergangsfunktion δ auf den Definitionsbereich $Q \times \Sigma^*$ erweitert. Die resultierende Funktion δ^* kann auf Worte beliebiger Länge ≥ 0 angewendet werden.

Auf dieser Basis läßt sich dann die von einem Automaten **akzeptierte Sprache** definieren.

Akzeptierte Sprache

Definition:
Sei $A = (Q, \Sigma, \delta, q_0, F)$ ein endlicher Automat mit der Übergangsfunktion $\delta : Q \times \Sigma \to P(Q)$.

Wir definieren die **erweiterte Übergangsfunktion** $\delta^* : Q \times \Sigma^* \to P(Q)$ total auf der Basis von δ wie folgt:

1. Für alle $q \in Q$:

$$\delta^*(q, \varepsilon) \in \{q\}$$ i.e. man verbleibt im Zustand

2. Für alle $w \in \Sigma^*$ und $a \in \Sigma$:

$$\delta^*(q, wa) = \bigcup_{p \in \delta^*(q,w)} \delta(p, a)$$

Bemerkung: $\delta^*(q, wa) = \delta(\delta^*(q, w), a)$ **für deterministische Automaten.**

Definition:
Sei $A = (Q, \Sigma, \delta, q_0, F)$. Dann ist

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

die von A **akzeptierte Sprache**.

Der Potenzautomat

Definition:
Sei $A = (Q, \Sigma, \delta, q_0, F)$ ein beliebiger **nichtdeterministischer** endlicher Automat. Wir konstruieren einen **deterministischen** endlichen Automaten, $A' = (Q', \Sigma, \delta', q_0', F')$ mittels folgender Regeln:

1. $Q' \subseteq P(Q) \setminus \varnothing$
2. $q_0' = [q_0]$
3. $F' \subseteq Q'$ sei die Menge aller Zustände in Q', die mindestens ein Element aus F enthalten.
4. $\delta' : Q' \times \Sigma \rightarrow Q'$ ist eine deterministische Übergangsfunktion für A', die wie folgt definiert wird:

$$\delta'([q_1, \ldots, q_i], a) = [p_1, \ldots, p_j],$$

wobei

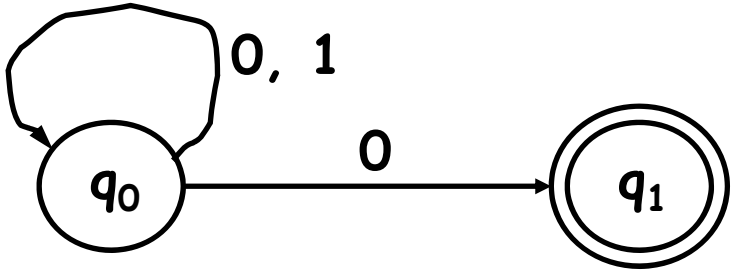
$$\cup_{1 \le k \le i} \delta(q_k, a) = \{ p_1, \ldots, p_j \} \qquad \text{für } [q_1, \ldots, q_i] \in Q', a \in \Sigma.$$

A' heißt der **Potenzautomat** zu A. Man kann zeigen, es gilt $L(A) = L(A')$.

Beispiel 1: $A = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_1\})$ (Sprache: {0,1}*0)

δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\varnothing$	$\varnothing$

NEA

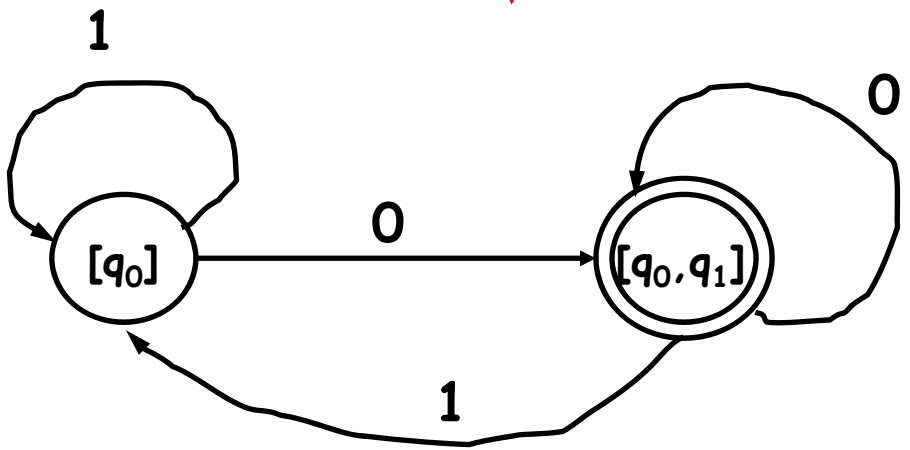


Transformation

$A' = (\{[q_0], [q_0,q_1]\}, \{0, 1\}, \delta', [q_0], \{[q_0,q_1]\})$

δ'	0	1
$[q_0]$	$[q_0,q_1]$	$[q_0]$
$[q_0,q_1]$	$[q_0,q_1]$	$[q_0]$

DEA



Beispiel 2:

Sei $A = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_1\})$ ein NEA mit

$\delta(q_0,0) = \{q_0, q_1\}$, $\delta(q_0,1) = \{q_1\}$,

$\delta(q_1,0) = \varnothing$, $\delta(q_1,1) = \{q_0, q_1\}$

A:

δ	0	1
q_0	$\{q_0,q_1\}$	$\{q_1\}$
q_1	$\varnothing$	$\{q_0,q_1\}$

A':

δ'	0	1
$[q_0]$	$[q_0,q_1]$	$[q_1]$
$[q_0,q_1]$	$[q_0,q_1]$	$[q_0,q_1]$
$[q_1]$	$\varnothing$	$[q_0,q_1]$

Potenzautomat A' = ($\{[q_0],[q_0,q_1],[q_1]\},\{0,1\}, \delta', [q_0], \{[q_0,q_1],[q_1]\}$)

Aufgabe: Zeichnen sie die Übergangsdiagramme für beide Automaten.

Beispiel 2: Fortsetzung

Konstruktionsschritte:
Wir können einen DEA $A' = (Q', \{0, 1\}, \delta', [q_0], F')$ konstruieren, der L(A) akzeptiert: Q´ besteht aus den Teilmengen von $\{q_0, q_1\}$, und die Zustände werden mit $[q_0]$, $[q_1]$, $[q_0, q_1]$ bezeichnet.

Da $\delta(q_0,0) = \{q_0, q_1\}$ gilt, folgt $\delta'([q_0],0) = [q_0, q_1]$

Weiters gilt
$\delta'([q_0],1) = [q_1]$ $\delta'([q_1],0) = \varnothing$ und $\delta'([q_1],1) = [q_0, q_1]$

Schließlich gilt noch

$$\delta'([q_0, q_1],0) = [q_0, q_1]$$

da $\delta'([q_0, q_1],0) = \cup_{0 \le k \le 1}\delta(q_k, 0) = \delta(q_0,0) \cup \delta(q_1,0) = \{q_0, q_1\} \cup \varnothing = \{q_0, q_1\}$

und es gilt

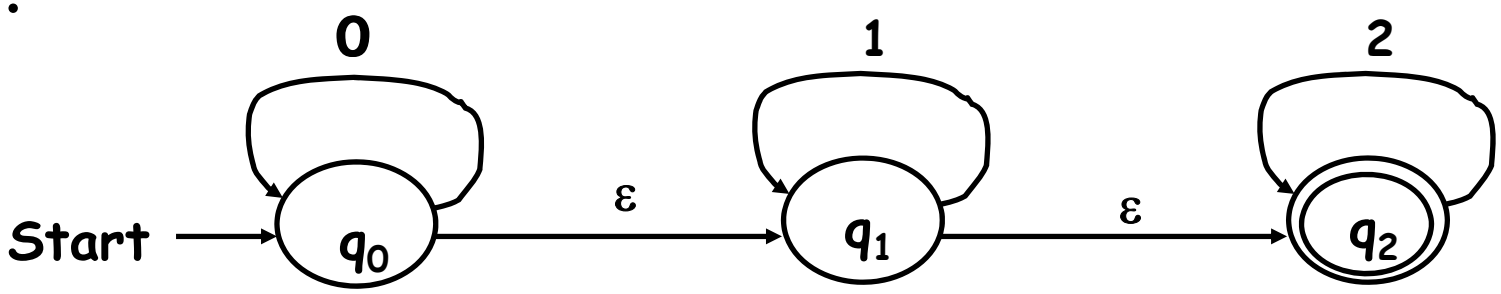
$$\delta'([q_0, q_1],1) = [q_0, q_1]$$

da $\delta'([q_0, q_1],1) = \cup_{0 \le k \le 1}\delta(q_k, 1) = \delta(q_0,1) \cup \delta(q_1,1) = \{q_1\} \cup \{q_0,q_1\} = \{q_0, q_1\}$

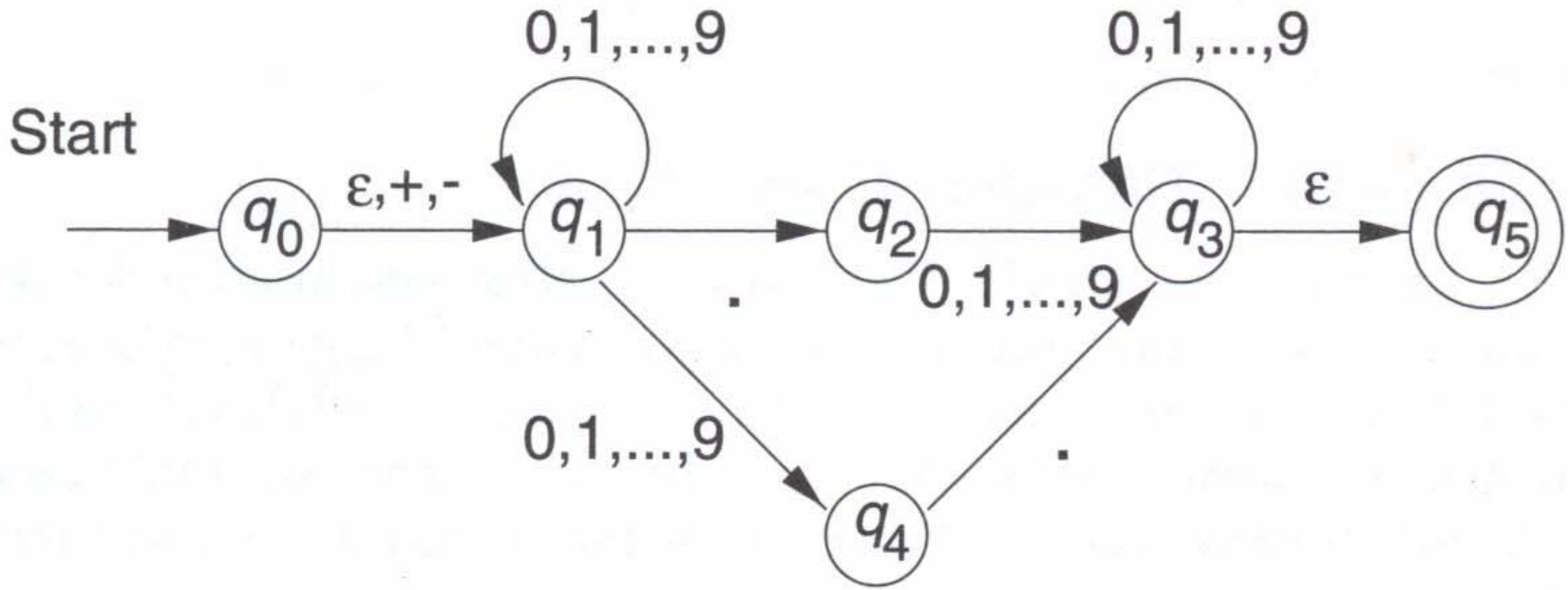
Erweiterung: Endliche Automaten mit ε-Übergängen

Wir können unser Modell des NEA durch Hinzunahme von ε-Übergängen erweitern. Das bedeutet, dass ein NEA in einen anderen Zustand übergehen kann, ohne ein Eingabesymbol zu konsumieren. Solche NEA bezeichnet man als ε–NEA $A=(Q,\Sigma,\delta,q_0,F)$ wobei δ zu $Q \times (\Sigma \cup \{\varepsilon\})$ erweitert wird, d.h. zusätzlich zum Eingabealphabet ist bei Übergängen auch ε erlaubt, sonst bleibt alles gleich (weiterhin gilt, dass ε kein Element aus Σ ist).

In der Abbildung wird das Übergangsdiagramm eines ε-NEA gezeigt, der die Sprache akzeptiert, die aus einer beliebigen Anzahl **0**er, gefolgt von einer beliebigen Anzahl **1**er, gefolgt von einer beliebigen Anzahl **2**er besteht (wobei beliebig auch kein Element inkludiert), i.e. entspricht dem regulären Ausdruck **0*1*2***.



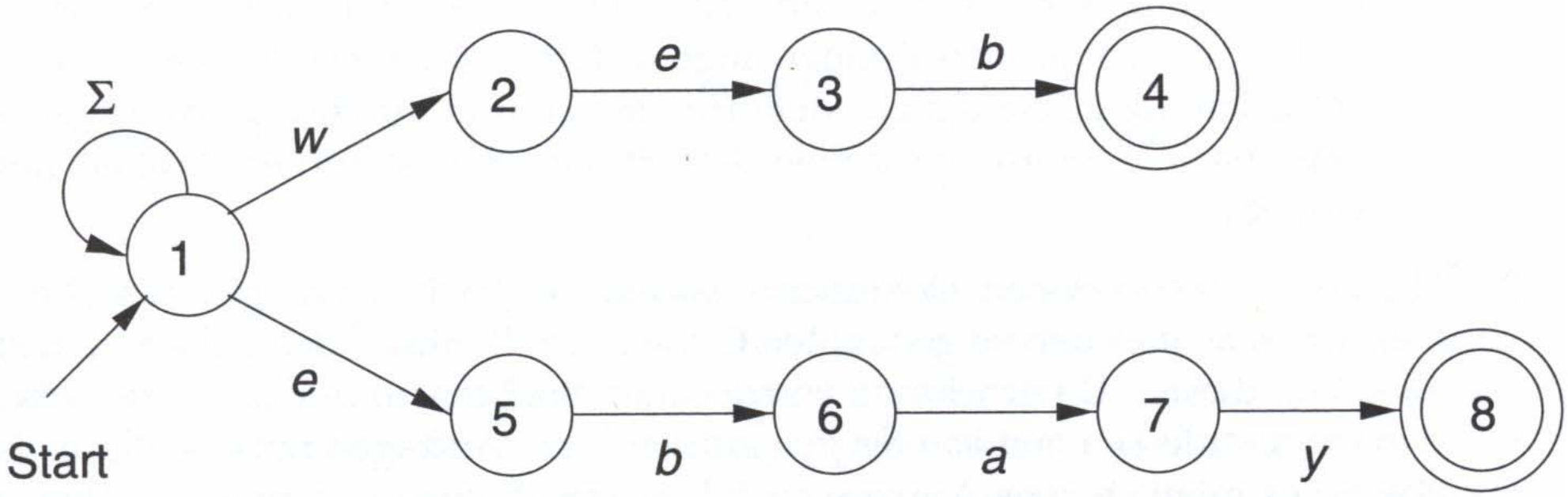
Beispiel: Ein ε-NEA, der Dezimalzahlen akzeptiert



Beispiel:
Textsuche nach den Wörtern web und ebay (1)

NEA

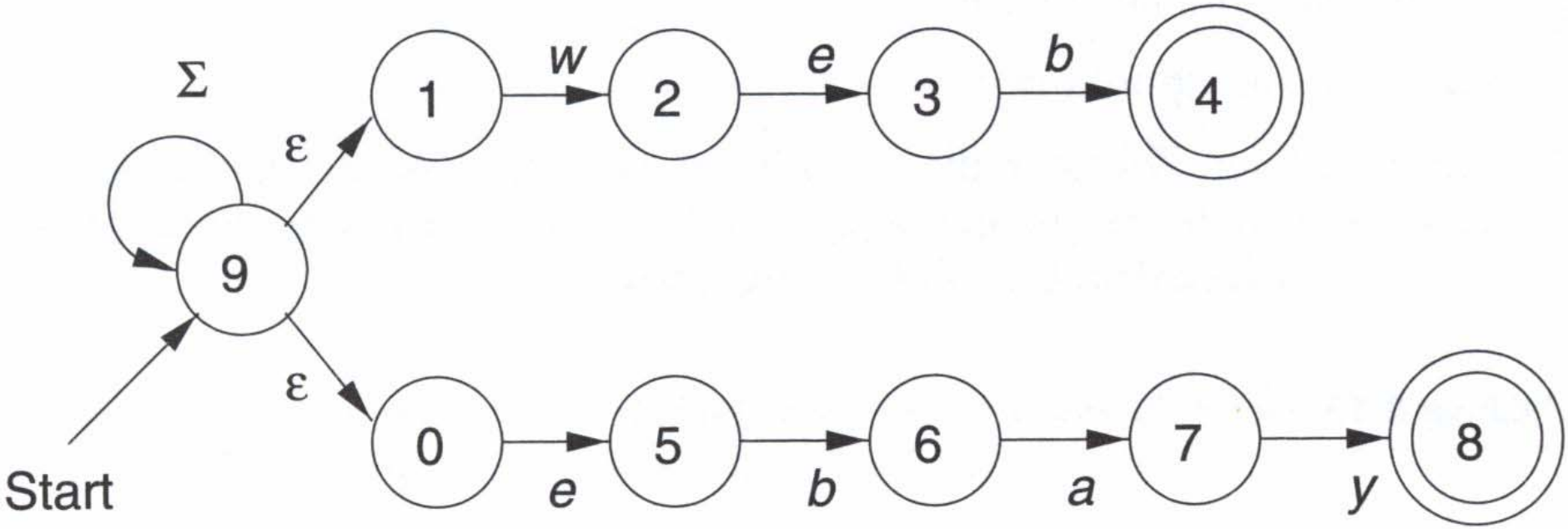
$A = (\{1,\ldots,8\}, \{a,\ldots,z\}, \delta, 1, \{4,8\})$



Textsuche ... (2)

ε-NEA

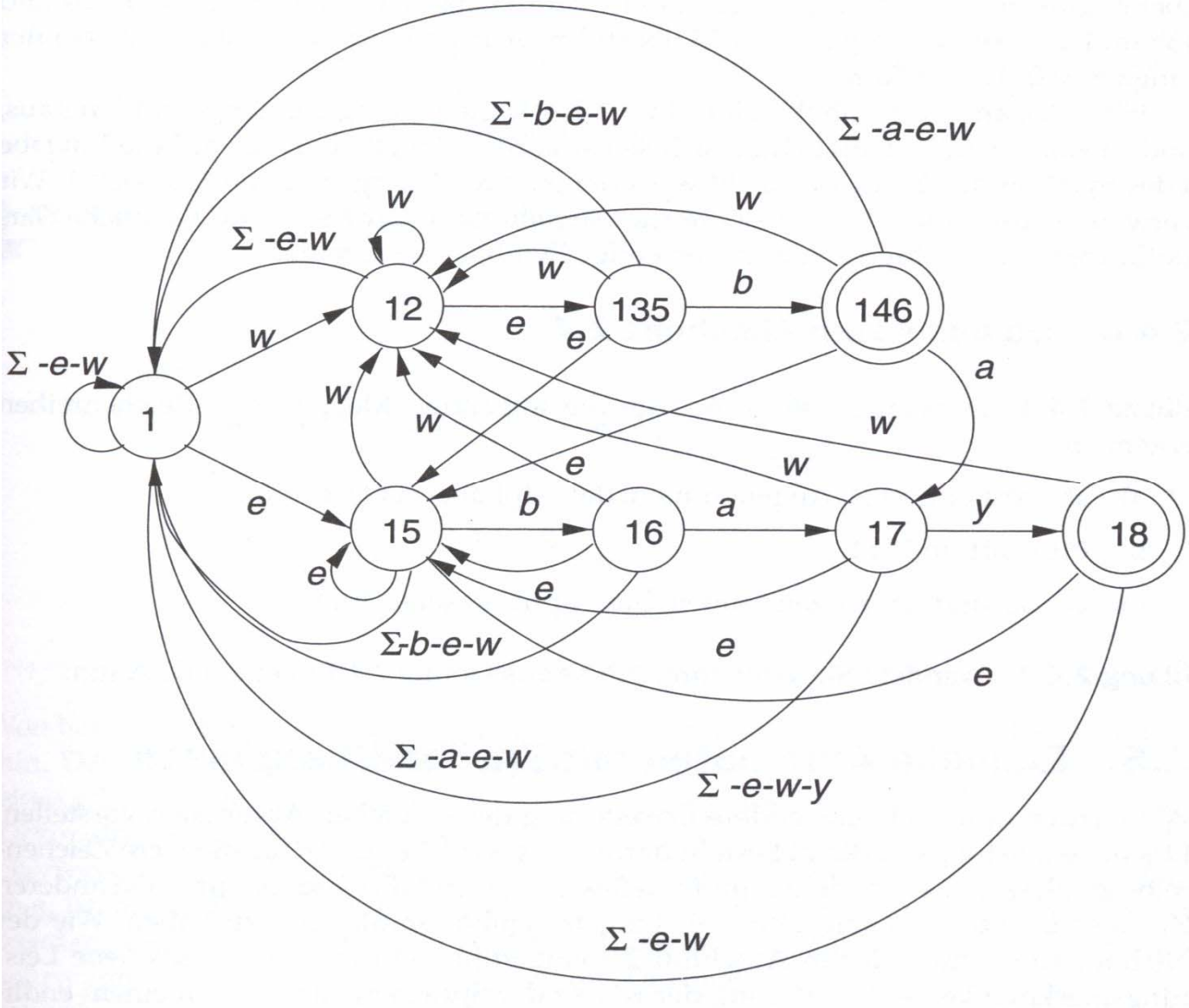
A = ({0,...,9}, {a,...,z}, δ, 9, {4,8})



Textsuche ... (3)

DEA

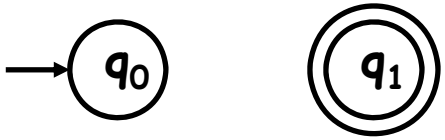
Umwandlung des NEA in einen DEA



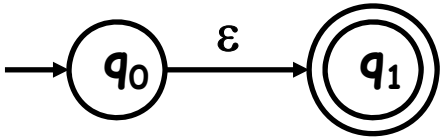
Systematische Konstruktion eines ε-NEAs aus einem regulären Ausdruck

Wir zeigen jetzt, dass aus einem regulären Ausdruck r ein ε-NEA konstruiert werden kann, der L(r) akzeptiert. Diese Konstruktion basiert auf der rekursiven Definition von L(r).

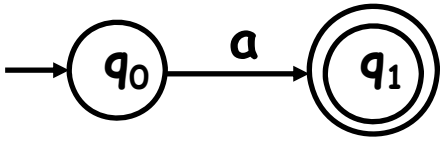
Am Anfang konstruieren wir einfache Automaten für Teile (1), (2), (3) unserer Definition und dann können wir zeigen, wie sie kombiniert werden, um die komplizierteren Teile (4) zu realisieren.



(NEA akzeptiert ∅)



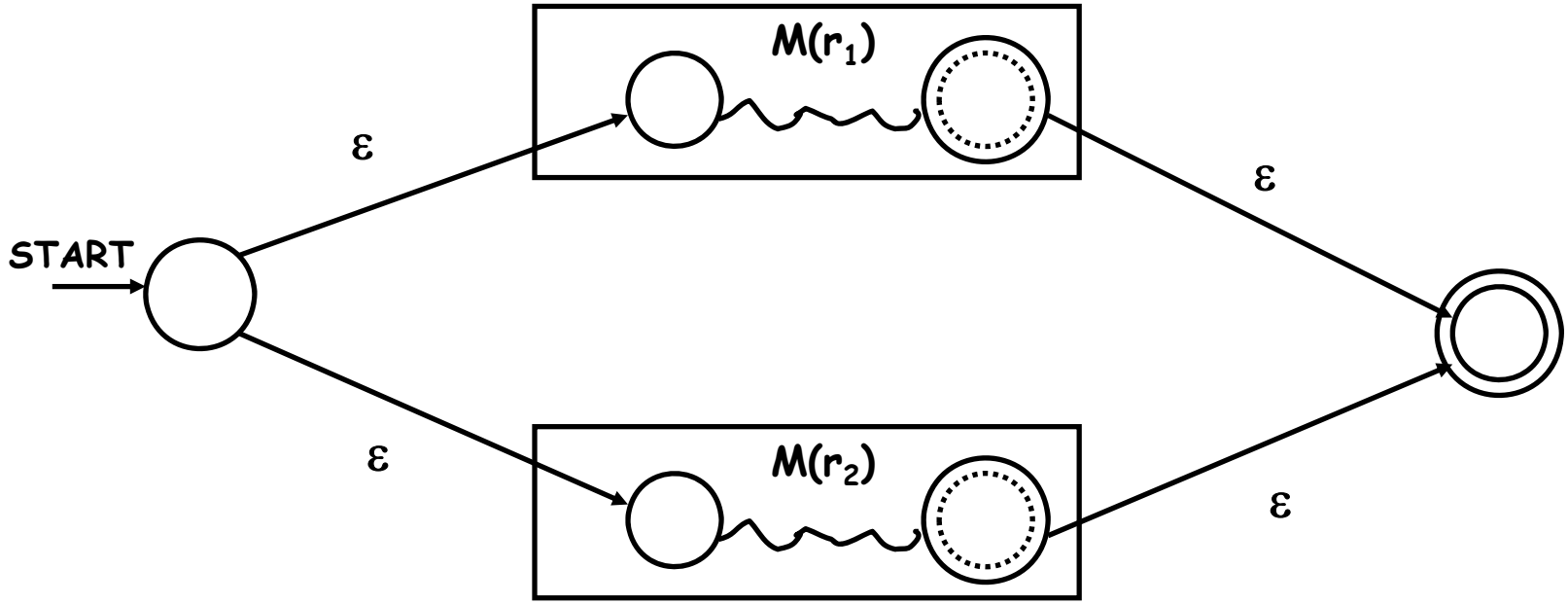
(NEA akzeptiert {ε})



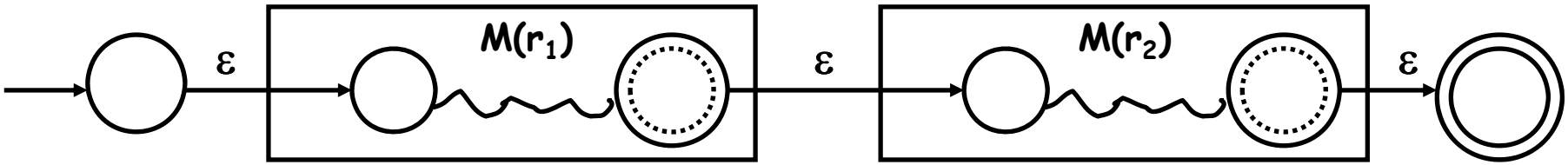
(NEA akzeptiert {a})

M(r) - ein NEA Automat, der L(r) akzeptiert

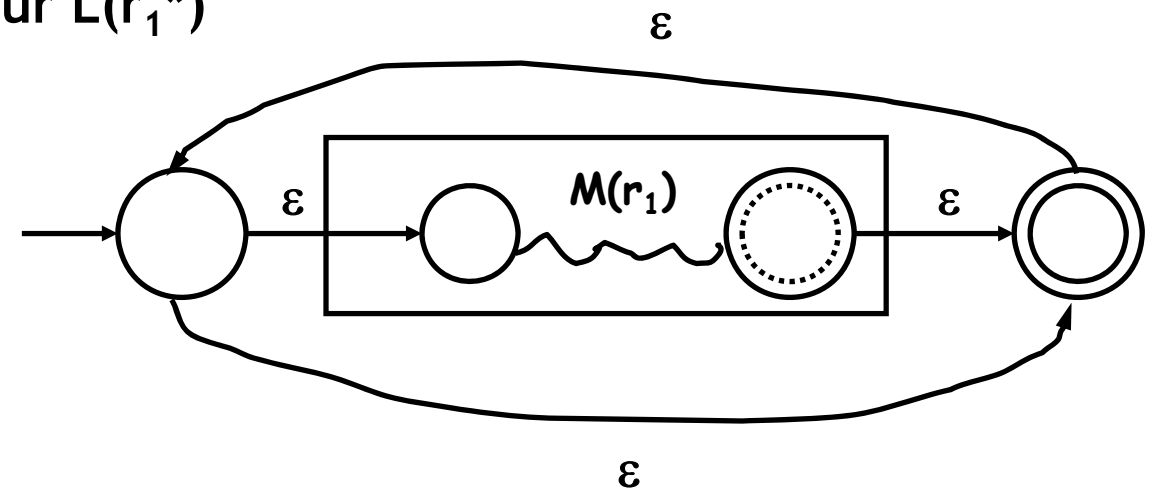
Automat für $L(r_1 \mid r_2)$



Automat für $L(r_1 r_2)$



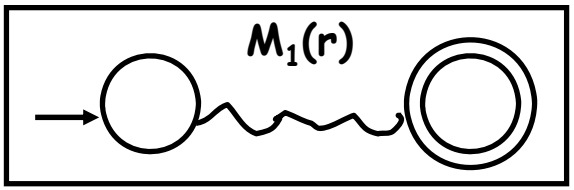
Automat für $L(r_1^*)$



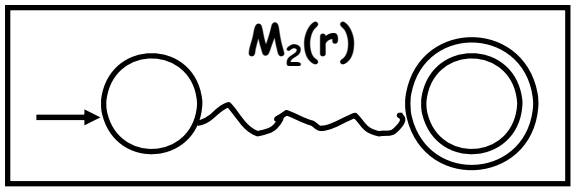
Automat für $L(r_1^+)$: der +-Operator wurde als vereinfachende Notation für $r_1 r_1^*$ eingeführt und wird hier nicht gesondert behandelt; als Automat kann er einfach direkt realisiert werden, indem die umgehende ε-Vorwärtskante weggelassen wird.

Beispiel 1: Konstruktion eines ε-NEA, der L(r) akzeptiert mit $r = (a|bb)^*(ba^*|\ \varepsilon)$.

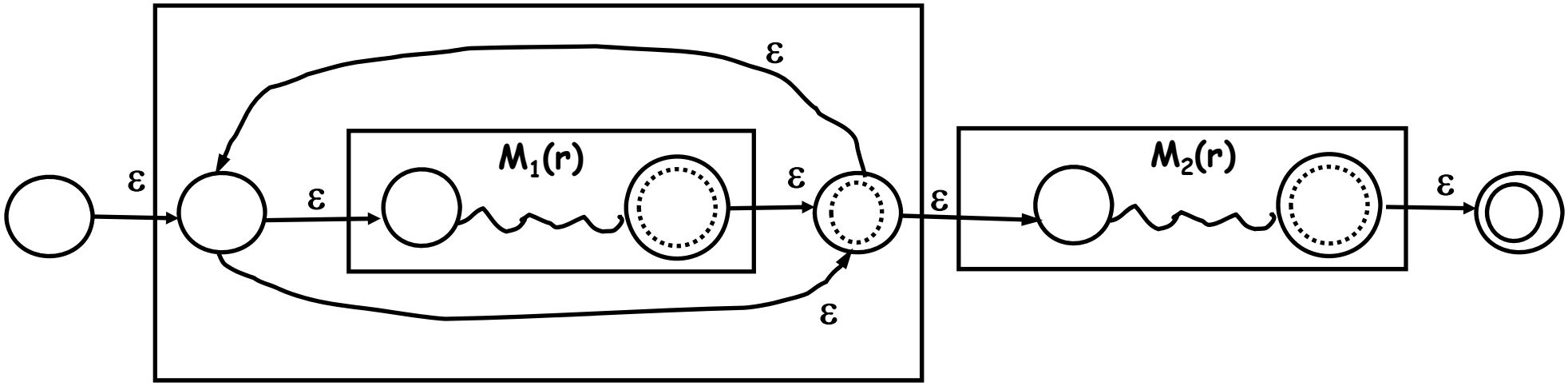
(a) M_1 akzeptiert $L(a|bb)$



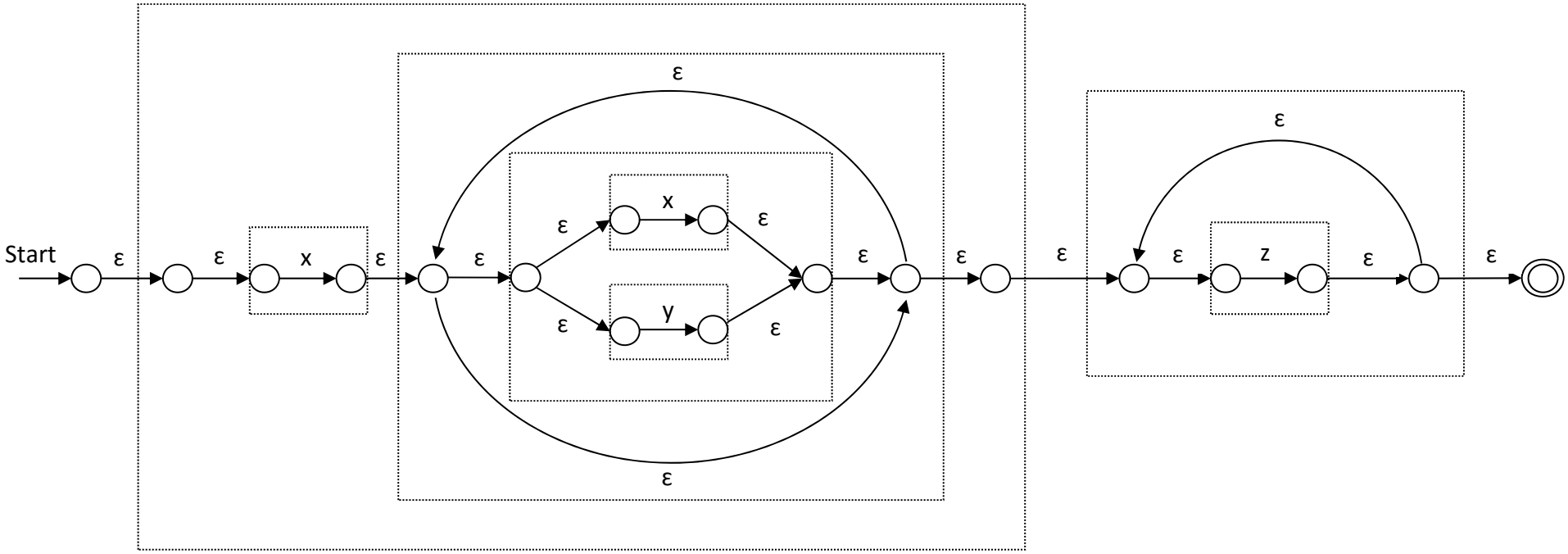
(b) M_2 akzeptiert $L(ba^*\ |\ \varepsilon)$



(c) Automat akzeptiert $L((a|bb)^*(ba^*|\ \varepsilon))$



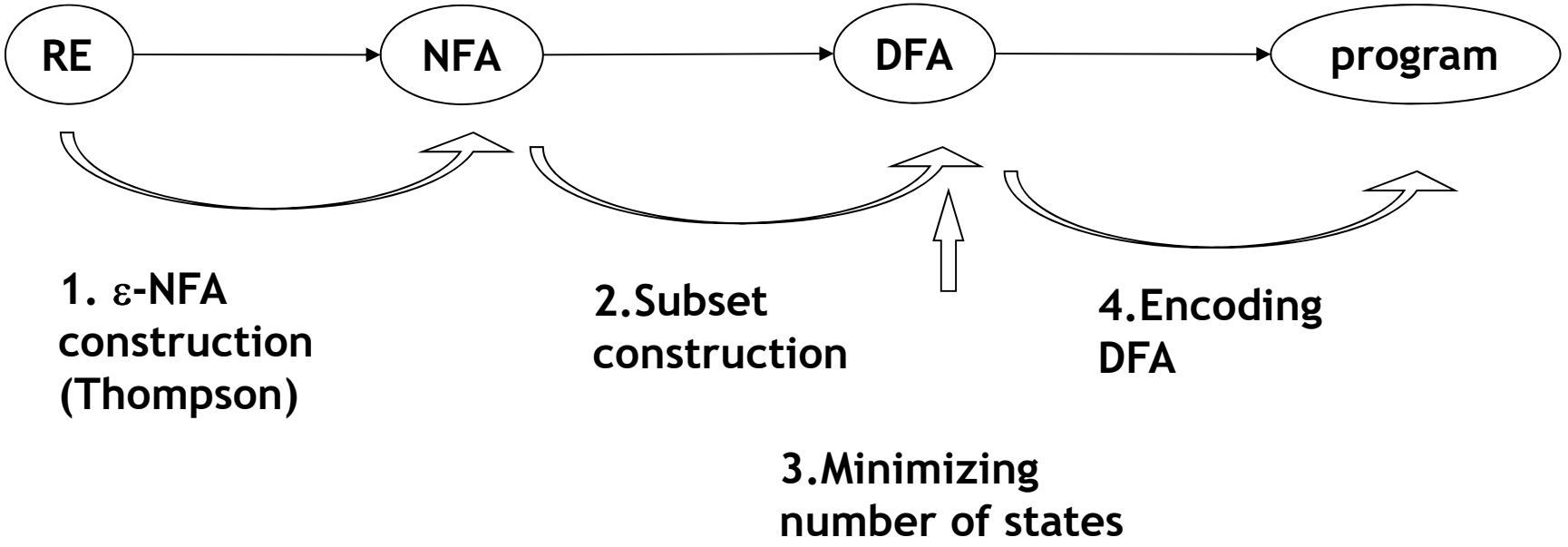
Beispiel 2: Konstruktion eines ε-NEA für den regulären Ausdruck $x(x|y)^*z^+$.



Scanner: From Regular Expressions to Executable

Task: From RE to DFA and encoding DFA.

(Note: RE…regular expression, NFA …nondeterministic finite automaton, DFA…deterministic finite automaton)



- Simplest algorithm: first intermediate construction of an NFA.
- Also possible: from RE directly to DFA.
- For each DFA exists a unique minimum-state DFA.

Encoding DFA: Table-driven Approach

Data structure **'table'**, indexed by state and input character, is derived from the **δ-transition table** and specifies the following actions:

- **go to** new state and read next input character (reading not always done)
- **accept** token and go to start state to wait for a request to analyze a new token
- **error** occurred

Code fragment of a typical scanner driver:

```
state=1;
head=read-next;
while (table[state,head]!=accept and
       table[state,head]!=error) {
  newstate=table[state,head];
  if (table[state,head]==advance) head=read-next();
  state=newstate;
}
if (table[state,head]==accept) {state=1; return ACCEPT;}
if (table[state,head]==error) {error-recovery; state=1;}
```

Endliche Automaten mit Ausgabe

Bis jetzt nur die Aussage: „akzeptiere“ oder „akzeptiere nicht“.

Man kann das bisher eingeführte Konzept eines endlichen Automaten durch Hinzunahme einer Ausgabefunktion erweitern.

Je nachdem, ob die Ausgabefunktion die Form

(1) $\lambda : Q \times \Sigma \rightarrow \Sigma'$ oder

(2) $\lambda : Q \rightarrow \Sigma'$

hat, spricht man von Mealy-Automaten (1) bzw. Moore-Automaten (2).
Hier ist Σ' ein Ausgabealphabet;
d.h. Mealy-Automat: Ausgabewert auf Kante,
Moore-Automat: Ausgabe beim Erreichen eines Zustands.