HOCHSCHULE
FÜR TECHNIK
ZÜRICH

Kurs: Informatik 2

Modul: Informatik I

Der Modulgruppe „Grundlagen der Informatik I“

Teil 2
Theoretische Informatik: Endliche Automaten,
Reguläre Sprachen

MITGLIED DER
ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Lernziele: (Allgemein)

- **Die Studierenden kennen die *Grundlagen***
  - der ***Formalen Sprachen*** und ***Automatentheorie***
  - der ***Berechenbarkeit*** und
  - der ***Komplexitätstheorie***
- **Sie beherrschen die *grundlegenden Konzepte*, *Begriffe* und *Definitionen* der theoretischen Informatik und können diese in praktischen Beispielen anwenden**

HOCHSCHULE FÜR TECHNIK ZÜRICH

Lernziele (Allgemein)

Kurs: Informatik 2 – Teil 2

Lernziele: [Zusatz]

- **Die Kurse der Module Informatik I und Informatik II (der Modulgruppen "Grundlagen der Informatik I+II") vermitteln den Studierenden die *Grundlagen der Informatik, die jede / jeder Studierende unabhängig von der Wahl der Wahlpflichtmodule im Fachstudium erlangen sollte***
- **Die vermittelten Grundlagen *werden in den Modulen im Fachstudium vorausgesetzt***

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE
FÜR TECHNIK
ZÜRICH

Kurs: Informatik 2 – Teil 2

Lernziele
(Teil 2)

Lernziele – Spezifisch Teil 2:

- **Die Studierenden können *deterministische* und *nichtdeterministische endliche Automaten* an Beispielen erklären**
- **Sie kennen die *Äquivalenz regulärer Ausdrücke* und *endlicher Automaten***
- **Die Studierenden kennen die *Eigenschaften* der *regulären Sprachen***

MITGLIED DER
ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Themenüberblick:

- **Theoretische Informatik**
  [Olaf Stern, 26 Lektionen - 21. Februar - 6. Juni 2011]
  - Einführung / Übersicht
  - Formale Sprachen / Automatentheorie
    - **Reguläre Sprachen, endliche Automaten**
    - Kontextfreie Sprachen, Kellerautomaten
    - [Kontextsensitive Sprachen, lineare Automaten]
    - Rekursive Sprachen und Turingmaschinen
  - Berechenbarkeit
  - Komplexitätstheorie

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Kurs: Informatik 2 – Teil 2

Lerninhalte:

- **Endliche Automaten**
  - Deterministische endliche Automaten
  - Nichtdeterministische endliche Automaten
  - ε-Nichtdeterministische endliche Automaten
  - Äquivalenz von RA und endlichen Automaten
- **Reguläre Sprachen**
  - Eigenschaften
  - Entscheidbarkeit
  - Pumping-Lema für reguläre Sprachen

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Deterministische endliche Automaten (DEA)**
  - **Def.:** ***Ein deterministischer endlicher Automat A wird (häufig) als 5-Tupel definiert:***

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

    und besteht aus

    - einer endlichen Menge von **Zuständen** ***Q =***
    - einer endlichen Menge von **Eingabesymbolen** $\Sigma =$
    - einer **Übergangsfunktion** δ
    - einem **Startzustand**
    - und einer Menge von finalen bzw. akzeptierenden Zuständen

    [mit $n, m \in IN^+$]

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

DEA

Endliche Automaten

- **Deterministische endliche Automaten (DEA)**
  - **Def.:** *Ein deterministischer endlicher Automat A wird (häufig) als 5-Tupel definiert:*

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

    und besteht aus

    - einer endlichen Menge von **Zuständen** $Q = \{q_0, \ldots, q_n\}$
    - einer endlichen Menge von **Eingabesymbolen** $\Sigma = \{a_0, \ldots, a_m\}$
    - einer **Übergangsfunktion** $\delta : Q \times \Sigma \rightarrow Q$
    - einem **Startzustand** q_0, $q_0 \in Q$
    - und einer Menge von finalen bzw. akzeptierenden Zuständen F, $F \subseteq Q$

    [mit $n, m \in IN^+$]

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Deterministische endliche Automaten (DEA)**
  - Der Begriff ***deterministisch*** bedeutet hierbei, dass der Automat auf Grund einer Eingabe von einem Zustand in **genau einen Zustand** übergehen kann

Vergl. auch Definition und Einführung in Informatik 1, Teil 6:

Dort wurden die deutschen Buchstaben für die Definition verwendet, hier die in der Fachliteratur[1] üblichen: statt Z nun Q, statt E nun Σ, statt f nun δ, statt z_0 nun q_0 und statt Z' nun F

[[1]Quelle: *Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie* von Hopcroft et al.]

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Def.:** *Gegeben seien ein DEA $A = (Q, \Sigma, \delta, q_0, F)$, der sich im Startzustand q_0 befindet, und eine Zeichenreihe $w = a_1a_2 \ldots a_n$ (mit den Symbolen $a_1, a_2, \ldots, a_n \in \Sigma$)*

    A verarbeitet nun die Eingabesymbole $a_1, a_2, \ldots, a_n$, indem mit Hilfe der Übergangsfunktion δ der jeweilige Folgezustand ermittelt wird:

    $$\delta(q_0, a_1) = q_1 \text{ , } \delta(q_1, a_2) = q_2 \text{ , } \ldots \text{ , } \delta(q_{n-1}, a_n) = q_n$$

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Def.:** ***Gegeben seien ein DEA $A = (Q, \Sigma, \delta, q_0, F)$, der sich im Startzustand q_0 befindet, und eine Zeichenreihe $w = a_1 a_2 \ldots a_n$ (mit den Symbolen $a_1, a_2, \ldots, a_n \in \Sigma$)***

    A **verarbeitet nun die Eingabesymbole *$a_1, a_2, \ldots, a_n$*, indem mit Hilfe der Übergangsfunktion *δ* der jeweilige Folgezu-stand ermittelt wird:**

    $$\delta(q_0, a_1) = q_1,\ \delta(q_1, a_2) = q_2, \ldots, \delta(q_{n-1}, a_n) = q_n$$

    Dieses wird auch über die *erweiterte Übergangsfunktion $\hat{\delta}$* definiert:

    $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$ **[d. h. $\hat{\delta}(q_0, w) = q_n$]**

    (Dabei wird vorausgesetzt, dass alle Übergänge definiert sind)

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Def.:** *Gegeben seien ein DEA $A = (Q, \Sigma, \delta, q_0, F)$, der sich im Startzustand q_0 befindet, und eine Zeichenreihe $w = a_1 a_2 \ldots a_n$ (mit den Symbolen $a_1, a_2, \ldots, a_n \in \Sigma$)*

    A „akzeptiert" die Zeichenreihe w genau dann, wenn die erweiterte Übergangsfunktion $\hat{\delta}$ für w in einen akzeptierenden Zustand q_n führt, d. h. $q_n \in F$ gilt

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Def.:** *Die „Sprache" L(A) eines DEA A = (Q, Σ, δ, q_0, F), besteht aus der Menge aller Zeichenreihen aus Σ^*, die ein DEA akzeptiert:*

    $L(A) = \{w \mid \hat{\delta}(q_0, w) \text{ ist in } F \text{ enthalten}\}$

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A'*, der die Sprache**
    L = {***x01y*** | ***x***, ***y*** sind beliebige Zeichenreihen über ***Σ= {0,1}***}
    akzeptiert

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A'*, der die Sprache**
    L **=** *{**x01y** | **x**, **y*** sind beliebige Zeichenreihen über $\Sigma = \{0,1\}$*}*
    akzeptiert

    Beispiele für Zeichenreihen, die

    in ***L*** **enthalten** sind: ***01***, ***01**01*, *0**01***, *111**01**1*, …

    in ***L*** **nicht enthalten** sind: ε, *1*, *00*, *1111*, *111100*, …

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A‘*, der die Sprache**
    ***L* = {*x01y* | *x*, *y* sind beliebige Zeichenreihen über *Σ= {0,1}*}**
    akzeptiert

    Konstruktion des *DEA A‘*=>

    Überlegungen zu *Σ*: *Σ= {0,1}*

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A'*, der die Sprache**

    L* = {*x01y* | *x*, *y sind beliebige Zeichenreihen über ***Σ* = {0,1}**}

    akzeptiert

    Überlegungen zu δ:

    a) ***A'* verbleibt im Startzustand, bis er eine *0* liesst** („überliesst" jede *1*), **und geht dann in einen neuen Zustand über** („merkt" sich die *0*)

    b) **Hat *A'* bereits eine *0* gelesen, verbleibt er in diesem Zustand, bis er eine *1* liesst, und geht dann in einen neuen Zustand über** („merkt" sich so *01*)

    c) **Hat *A'* bereits eine *01* gelesen, verbleibt er in diesem Zustand und akzeptiert jede weitere *0* oder 1 als Eingabe**

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A'*, der die Sprache**

    L = {***x01y*** | ***x***, ***y*** sind beliebige Zeichenreihen über $\Sigma = \{0,1\}$}

    akzeptiert

    Überlegungen zu δ:

    a) ***A'*** **verbleibt im Startzustand, bis er eine *0* liesst** („überliesst" jede *1*), **und geht dann in einen neuen Zustand über** („merkt" sich die *0*)

    Daraus lässt sich ableiten:

    $\delta(q_0, 1) = q_0$ und $\delta(q_0, 0) = q_1$

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

DEA

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A'*, der die Sprache**

    $L = \{x01y \mid x, y$ sind beliebige Zeichenreihen über $\Sigma = \{0,1\}\}$

    akzeptiert

    Überlegungen zu δ:

    b) **Hat *A'* bereits eine *0* gelesen, verbleibt er in diesem Zustand, bis er eine *1* liesst, und geht dann in einen neuen Zustand über** („merkt" sich so *01*)

    Daraus lässt sich ableiten:

    $\delta(q_1, 0) = q_1$ und $\delta(q_1, 1) = q_2$

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A'*, der die Sprache**

    $L = \{x01y \mid x, y$ sind beliebige Zeichenreihen über $\Sigma = \{0,1\}\}$

    akzeptiert

    Überlegungen zu δ:

    c) **Hat *A'* bereits eine *01* gelesen, verbleibt er in diesem Zustand und akzeptiert jede weitere *0* oder 1 als Eingabe**

    $\delta(q_2, 0) = q_2$ und $\delta(q_2, 1) = q_2$

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A'*, der die Sprache**

    $L = \{x01y \mid x, y$ sind beliebige Zeichenreihen über $\Sigma = \{0,1\}\}$

    akzeptiert

    Überlegungen zu δ:

    Damit gilt für δ:

    $\delta(q_0, 1) = q_0$, $\delta(q_0, 0) = q_1$, $\delta(q_1, 0) = q_1$, $\delta(q_1, 1) = q_2$,
    $\delta(q_2, 0) = q_2$ und $\delta(q_2, 1) = q_2$

    Mit

    q_0 als **Startzustand** und
    q_2 als einzigem **finalen Zustand**

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines DEA**
  - **Beispiel: Gesucht wird der *DEA A'*, der die Sprache**

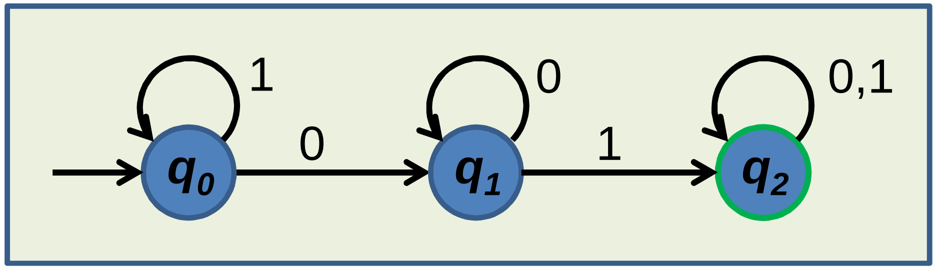
    $L = \{x01y \mid x, y$ sind beliebige Zeichenreihen über $\Sigma = \{0,1\}\}$

    akzeptiert

    Mögliche Lösung für den *EDA A'*:

    $A' = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$

    mit $\delta = \{\delta(q_0, 1) = q_0, \delta(q_0, 0) = q_1, \delta(q_1, 0) = q_1, \delta(q_1, 1) = q_2, \delta(q_2, 0) = q_2, \delta(q_2, 1) = q_2\}$



Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Nichtdeterministische endliche Automaten (NEA)**
  - **Def.:** ***Ein nichtdeterministischer endlicher Automat A wird (häufig) als 5-Tupel definiert:***

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

    Er unterscheidet sich von einem *DEA* lediglich in der Übergangsfunktion δ, die einen Zustand aus *Q* und ein Eingabesymbol aus Σ in eine *Teilmenge von Q* abbildet:

    $$\delta : Q \times \Sigma \rightarrow P(Q)$$

    [P steht für Potenzmenge; „Menge aller Teilmengen“]

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

Endliche Automaten

- **Nichtdeterministische endliche Automaten (NEA)**
  - Der Begriff ***nichtdeterministisch*** bedeutet hierbei, dass der Automat auf Grund einer Eingabe von einem Zustand in **mehrere Zustände gleichzeitig** übergehen kann

    Dieses wird so interpretiert, dass der NEA bei **mehreren möglichen Folgezuständen** den „***korrekten***“ **Folgezustand errät**

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

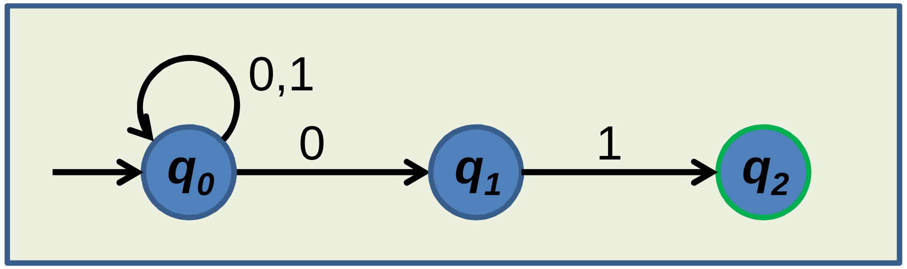
Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

Endliche Automaten

- **Nichtdeterministische endliche Automaten (NEA)**
  - **Beispiel:**



MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

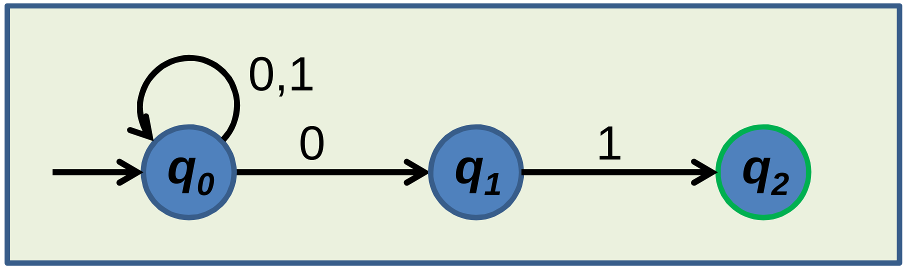
NEA

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Nichtdeterministische endliche Automaten (NEA)**
  - **Beispiel:**



NEA, der alle auf *„01"* endende Zeichenreihen akzeptiert

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

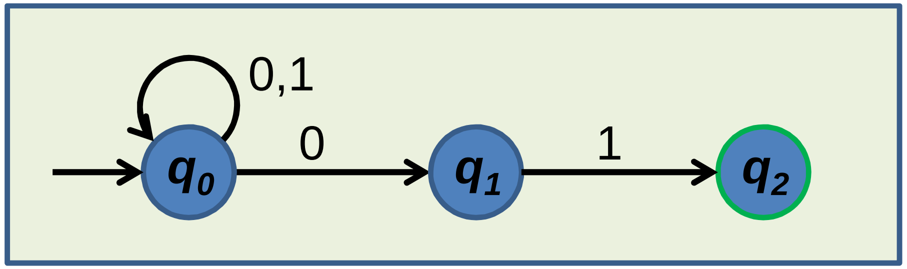
Endliche Automaten

NEA

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Nichtdeterministische endliche Automaten (NEA)**
  - **Beispiel:**



NEA, der alle auf „*01*“ endende Zeichenreihen akzeptiert

mit: Q = Σ =

q_0 = F =

δ =

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE
FÜR TECHNIK
ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

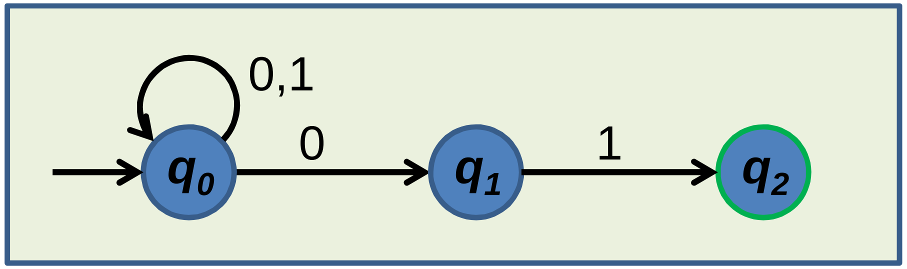
Endliche Automaten

NEA

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Nichtdeterministische endliche Automaten (NEA)**
  - **Beispiel:**



NEA, der alle auf „*01*" endende Zeichenreihen akzeptiert

mit: $Q = \{q_0, q_1, q_2\}$ $\Sigma = \{0,1\}$

$q_0 = q_0$ $F = \{q_2\}$

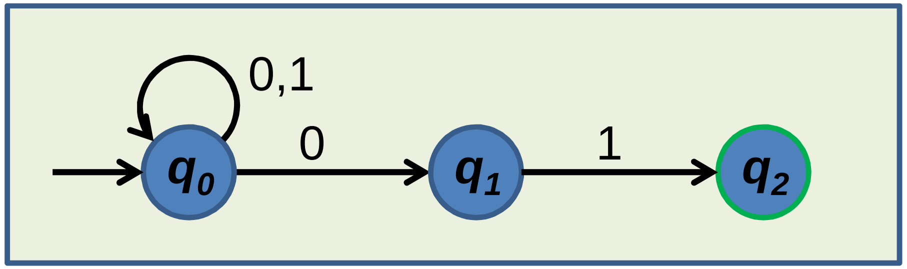
$\delta =$

MITGLIED DER
ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Nichtdeterministische endliche Automaten (NEA)**
  - **Beispiel:**



NEA, der alle auf „*01*" endende Zeichenreihen akzeptiert

mit: $Q = \{q_0, q_1, q_2\}$ $\quad\quad \Sigma = \{0,1\}$

$q_0 = q_0$ $\quad\quad F = \{q_2\}$

$\delta = \{\delta(q_0,1) = \{q_0\},\ \delta(q_0,0) = \{q_0, q_1\},\ \delta(q_1,1) = \{q_2\}$

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines NEA**
  - **Def.:** ***Gegeben seien ein NEA $A = (Q, \Sigma, \delta, q_0, F)$, der sich im Startzustand q_0 befindet, und eine Zeichenreihe $w = a_1a_2 \ldots a_n$ (mit den Symbolen $a_1, a_2, \ldots, a_n \in \Sigma$)***

    A **verarbeitet nun die Eingabesymbole *$a_1, a_2, \ldots, a_n$*, indem mit Hilfe der Übergangsfunktion *δ* die jeweiligen Folgezu-stände ermittelt werden:**

    1. Schritt: $\delta(q_0, a_1) = \{q_x, \ldots, q_y\}$ **mit:** $\{q_x, \ldots, q_y\} \subseteq Q$

    Im nächsten Schritt wird nun für alle im ersten Schritt erreichten Zustände die Übergangsfunktion *δ* angewen-det. Die Vereinigung aller erreichbaren Zustände ist das Resultat dieses Schrittes. Dieses wird für die weiteren Schritte so fortgesetzt

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

Endliche Automaten

- **Sprache eines NEA**
  - **Def.:** *Gegeben seien ein NEA $A = (Q, \Sigma, \delta, q_0, F)$, der sich im Startzustand q_0 befindet, und eine Zeichenreihe $w = a_1 a_2 \ldots a_n$ (mit den Symbolen $a_1, a_2, \ldots, a_n \in \Sigma$)*

    A **verarbeitet nun die Eingabesymbole** $a_1, a_2, \ldots, a_n$**, indem mit Hilfe der Übergangsfunktion** δ **die jeweiligen Folgezu-stände ermittelt werden:**

    1. Schritt: $\delta(q_0, a_1) = \{q_x, \ldots, q_y\}$ **mit:** $\{q_x, \ldots, q_y\} \subseteq Q$

    2. Schritt:

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Endliche Automaten

- **Sprache eines NEA**
  - **Def.:** *Gegeben seien ein NEA $A = (Q, \Sigma, \delta, q_0, F)$, der sich im Startzustand q_0 befindet, und eine Zeichenreihe $w = a_1 a_2 \ldots a_n$ (mit den Symbolen $a_1, a_2, \ldots, a_n \in \Sigma$)*

    A verarbeitet nun die Eingabesymbole $a_1, a_2, \ldots, a_n$, indem mit Hilfe der Übergangsfunktion δ die jeweiligen Folgezustände ermittelt werden:

    1. Schritt: $\delta(q_0, a_1) = \{q_x, \ldots, q_y\}$ **mit:** $\{q_x, \ldots, q_y\} \subseteq Q$

    2. Schritt: $\delta(q_x, a_2) \cup \ldots \cup \delta(q_y, a_2) = \{q_{x1}, \ldots, q_{y1}\} \cup \ldots$

    $\cup \{q_{xm}, \ldots, q_{ym}\} = \{q_i, \ldots, q_j\}$ **mit:** $\{q_i, \ldots, q_j\} \subseteq Q$

    . . .

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines NEA**
  - **Def.:** *Gegeben seien ein NEA $A = (Q, \Sigma, \delta, q_0, F)$, der sich im Startzustand q_0 befindet, und eine Zeichenreihe $w = a_1a_2 \ldots a_n$ (mit den Symbolen $a_1, a_2, \ldots, a_n \in \Sigma$)*

    A **verarbeitet nun die Eingabesymbole $a_1, a_2, \ldots, a_n$, indem mit Hilfe der Übergangsfunktion δ die jeweiligen Folgezu-stände ermittelt werden:**

    Entsprechend wird die erweiterte Übergangsfunktion $\hat{\delta}$ definiert.

    $\hat{\delta} : Q \times \Sigma^* \rightarrow P(Q)$ **[d. h.** $\hat{\delta}(q_0, w) = \{q_x,\ldots,q_y\}$ **]**

    mit: $\{q_x,\ldots,q_y\} \subseteq Q$

    (Nicht definierte Übergänge bilden implizit auf die leere Menge $\varnothing$ von Zuständen ab)

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines NEA**
  - **Def.:** *Gegeben seien ein NEA $A = (Q, \Sigma, \delta, q_0, F)$, der sich im Startzustand q_0 befindet, und eine Zeichenreihe $w = a_1 a_2 \dots a_n$ (mit den Symbolen $a_1, a_2, \dots, a_n \in \Sigma$)*

    A „akzeptiert" die Zeichenreihe w genau dann, wenn die erweiterte Übergangsfunktion $\hat{\delta}$

    $\hat{\delta}(q_0, w) = \{q_x, \dots, q_y\}$ **mit:** $\{q_x, \dots, q_y\} \subseteq Q$

    für w in mindestens einen akzeptierenden Zustand führt, d. h.:

    $\{q_x, \dots, q_y\} \cap F \neq \varnothing$

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines NEA**
  - **Def.:** *Die „Sprache" L(A) eines NEA A = (Q, Σ, δ, q_0, F), besteht aus der Menge aller Zeichenreihen aus Σ^*, die ein NEA akzeptiert:*

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

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

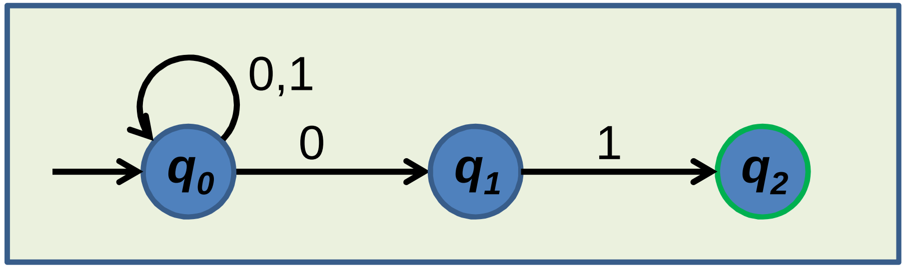
NEA

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines NEA**
  - **Beispiel: Zur *erweiterten Übergangsfunktion* $\hat{\delta}$ eines *NEA***

    Gegeben sei der bereits verwendete NEA und die Zeichen-reihe „*01001*":



1:

2:

3:

4:

5:

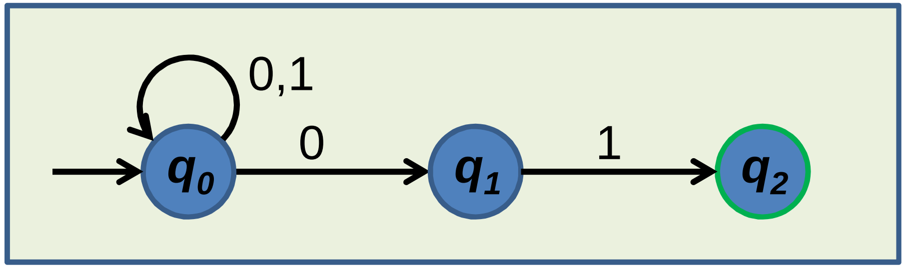
MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines NEA**
  - **Beispiel: Zur *erweiterten Übergangsfunktion* $\hat{\delta}$ eines *NEA***

    Gegeben sei der bereits verwendete NEA und die Zeichen-reihe „*01001*“:



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

2:

3:

4:

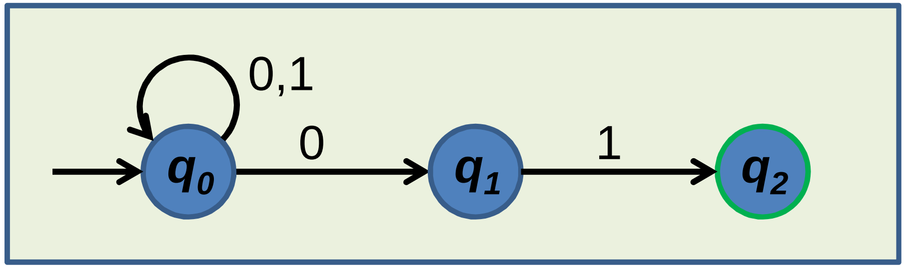
5:

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines NEA**
  - **Beispiel: Zur *erweiterten Übergangsfunktion* $\hat{\delta}$ eines *NEA***

    Gegeben sei der bereits verwendete NEA und die Zeichenreihe „*01001*":



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

2: $\hat{\delta}(q_0,01) = \delta(q_0,1) \cup \delta(q_1,1) = \{q_0\} \cup \{q_2\} = \{q_0,q_2\}$

3:

4:

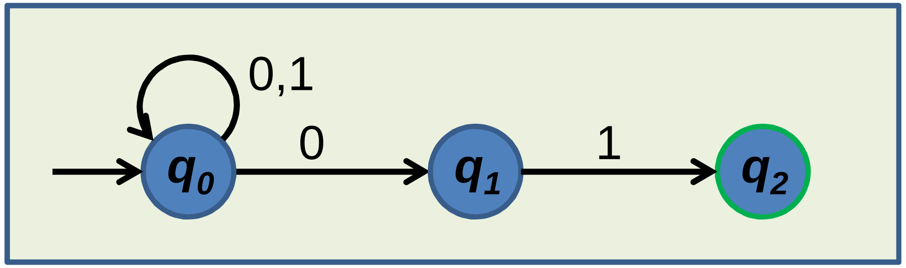
5:

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Sprache eines NEA**
  - **Beispiel: Zur *erweiterten Übergangsfunktion* $\hat{\delta}$ eines *NEA***

Gegeben sei der bereits verwendete NEA und die Zeichenreihe „*01001*“:



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

2: $\hat{\delta}(q_0,01) = \delta(q_0,1) \cup \delta(q_1,1) = \{q_0\} \cup \{q_2\} = \{q_0,q_2\}$

3: $\hat{\delta}(q_0,010) = \delta(q_0,0) \cup \delta(q_2,0) = \{q_0,q_1\} \cup \emptyset = \{q_0,q_1\}$

4: $\hat{\delta}(q_0,0100) = \delta(q_0,0) \cup \delta(q_1,0) = \{q_0,q_1\} \cup \emptyset = \{q_0,q_1\}$

5: $\hat{\delta}(q_0,01001) = \delta(q_0,1) \cup \delta(q_1,1) = \{q_0,q_1\} \cup \{q_2\} = \{q_0,q_2\}$

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Äquivalenz von DEA und NEA**
  - **Satz:** ***Eine Sprache L wird von einem DEA genau dann erkannt, wenn L von einem NEA erkannt wird***

    =>„*DEA* und *NEA* sind gleich mächtig!“
  - **Beweiskonstruktion:**
    a) **Zeigen, dass zu jedem *DEA* ein *NEA* konstruiert werden kann, der die dieselbe Sprache akzeptiert**

    b) **Zeigen, dass zu jedem *NEA* ein *DEA* konstruiert werden kann, der die dieselbe Sprache akzeptiert**

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Äquivalenz von DEA und NEA**
  - **Satz:** ***Eine Sprache L wird von einem DEA genau dann erkannt, wenn L von einem NEA erkannt wird***

    =>„*DEA* und *NEA* sind gleich mächtig!"
  - **Beweiskonstruktion:**

    a) **Zeigen, dass zu jedem *DEA* ein *NEA* konstruiert werden kann, der die dieselbe Sprache akzeptiert**

    => Einfach: konstruieren die Übergangsfunktion δ für den ***NEA*** so, dass sie genau nur einen Übergang besitzt:

    wenn: $\delta_D(q,a) = p$, dann $\delta_N(q,a) = \{p\}$

    und entsprechend $\hat{\delta}$

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Äquivalenz von DEA und NEA**
  - **Satz:** ***Eine Sprache L wird von einem DEA genau dann erkannt, wenn L von einem NEA erkannt wird***

    =>„*DEA* und *NEA* sind gleich mächtig!“
  - **Beweiskonstruktion:**

    b) Zeigen, dass zu jedem *NEA* ein *DEA* konstruiert werden kann, der die dieselbe Sprache akzeptiert

    => Über eine „*Teilmengenkonstruktion*“ und Induktion

[Beweis: siehe Hopcroft, et al. S. 70 ff.]

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Äquivalenz von DEA und NEA**
  - **Satz:** ***Eine Sprache L wird von einem DEA genau dann erkannt, wenn L von einem NEA erkannt wird***

    =>„*DEA* und *NEA* sind gleich mächtig!“
  - **Anmerkungen:**
    - In der Praxis hat der zu einem ***NEA*** äquivalente ***DEA*** häufig in etwa (nur) genauso viele Zustände (Worst Case 2^n Zustände) …
    - … aber deutlich mehr Zustandsübergänge

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **NEA mit ε-Übergängen**
  - **Def.:** ***Ein nichtdeterministischer endlicher Automat A mit ε-Übergängen wird (häufig) als 5-Tupel definiert:***

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

Er unterscheidet sich von einem *NEA* lediglich in der Übergangsfunktion δ, die einen Zustand aus Q und ein Eingabesymbol aus $\Sigma \cup \{\varepsilon\}$ in eine *Teilmenge von Q* abbildet:

$$\delta : Q \times \Sigma \cup \{\varepsilon\} \rightarrow \{q_i, q_{i+1}, \ldots, q_{i+m}\}, \quad \{q_i, q_{i+1}, \ldots, q_{i+m}\} \subseteq Q$$

[ε, das Symbol für die leere Zeichenreihe, darf kein Element von Σ sein, um Verwechslungen zu vermeiden – dieses stellt aber keine grundsätzliche Einschränkung dar]

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **NEA mit ε-Übergängen**
  - Dass ein **Eingabesymbol aus** $\Sigma \cup \{\varepsilon\}$ für einen Zustands-übergang zulässig ist, bedeutet, dass der Automat auch auf Grund von **keiner Eingabe** (entspricht der Eingabe einer leeren Zeichenreihe) **von einem Zustand** in einen **anderen Zustände übergehen kann**

    Man sagt auch, der NEA wechselt „***spontan***“ in einen anderen Zustand

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

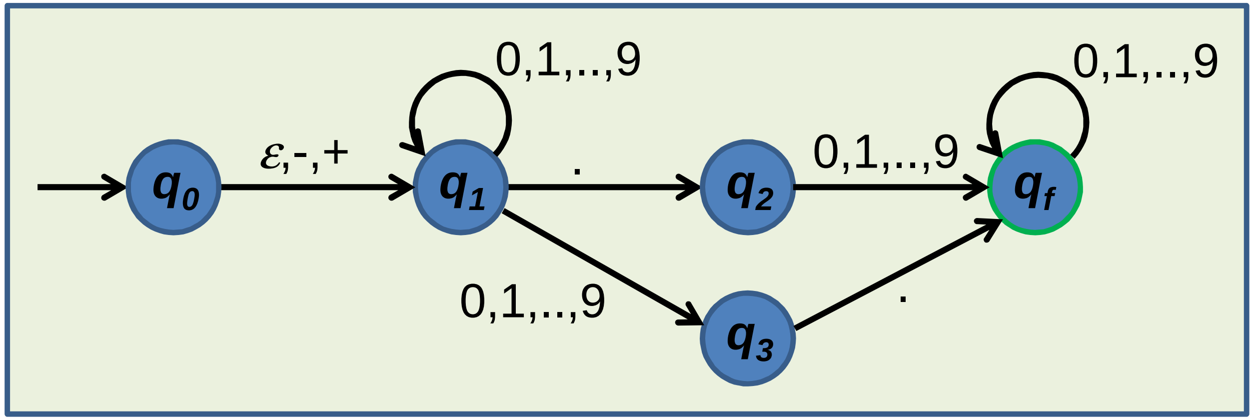
Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

Endliche Automaten

- **NEA mit ε-Übergängen**
  - **Beispiel:**



MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

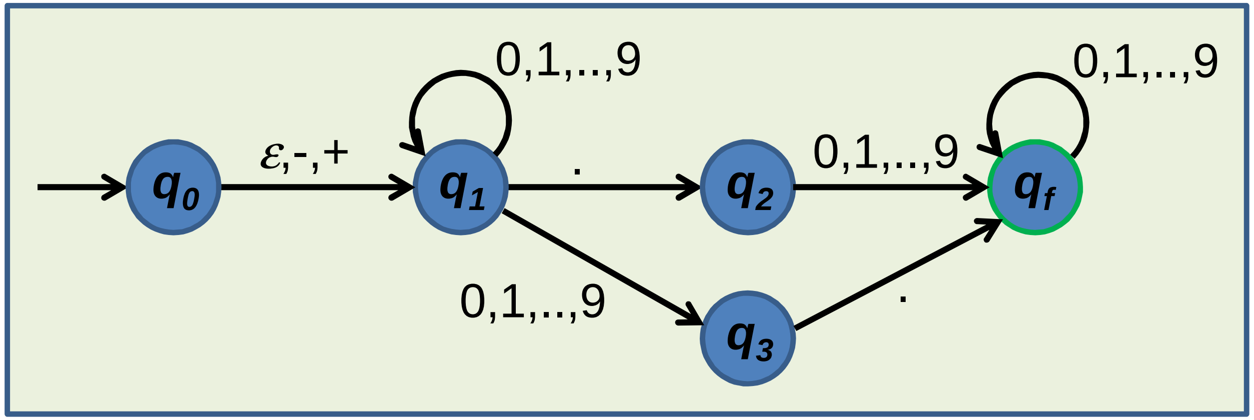
Endliche Automaten

NEA

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **NEA mit ε-Übergängen**
  - **Beispiel: Ein *ε-NEA*, der allen Dezimalzahlen akzeptiert**

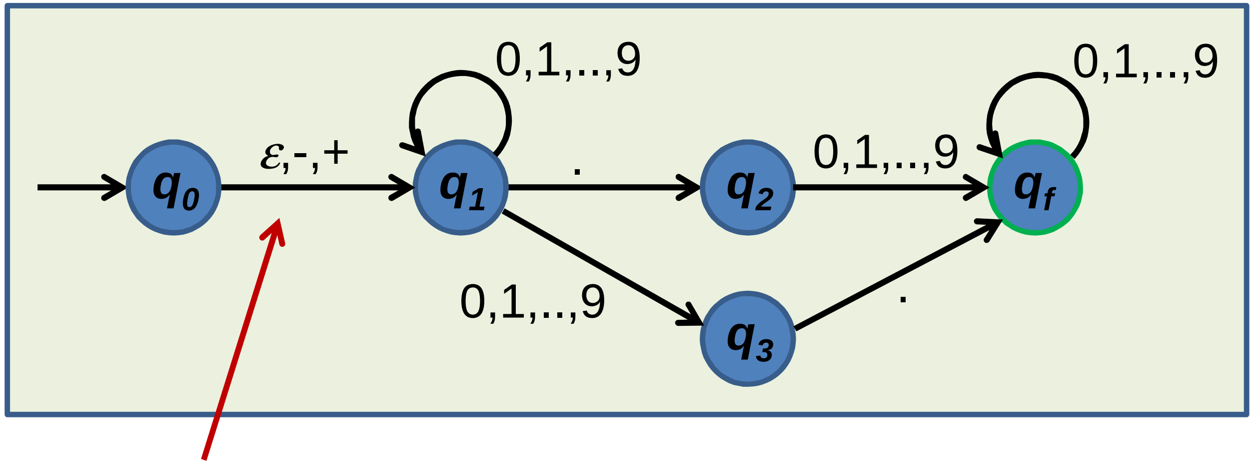


MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **NEA mit ε-Übergängen**
  - **Beispiel: Ein *ε-NEA*, der allen Dezimalzahlen akzeptiert**



Der ε**-Übergang** wird im wesentlichen zur **Vereinfachung** verwendet …

… daher verwundert es auch nicht, dass auch gilt:

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Äquivalenz von NEA und NEA mit ε-Übergängen**
  - **Satz:** ***Eine Sprache L wird von einem NEA genau dann erkannt, wenn L von einem ε-NEA erkannt wird***

    =>„*ε-NEA* und *NEA* sind gleich mächtig!“
  - **Beweiskonstruktion:**
    - a) **Zeigen, dass zu jedem *NEA* ein *ε-NEA* konstruiert werden kann, der die dieselbe Sprache akzeptiert**
    - b) **Zeigen, dass zu jedem *ε-NEA* ein *NEA* konstruiert werden kann, der die dieselbe Sprache akzeptiert**

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

Endliche Automaten

- **Äquivalenz von NEA und NEA mit ε-Übergängen**
  - **Satz:** ***Eine Sprache L wird von einem NEA genau dann erkannt, wenn L von einem ε-NEA erkannt wird***

    =>„*ε-NEA* und *NEA* sind gleich mächtig!“
  - **Beweiskonstruktion:**

    a) **Zeigen, dass zu jedem *NEA* ein *ε-NEA* konstruiert werden kann, der die dieselbe Sprache akzeptiert**

    => Einfach: die Übergangsfunktion δ für den ***ε-NEA*** so konstruieren, dass für alle Übergänge der Übergang $\delta_N(q,\varepsilon) = \{\emptyset\}$ hinzugefügt wird

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Äquivalenz von NEA und NEA mit ε-Übergängen**
  - **Satz:** ***Eine Sprache L wird von einem NEA genau dann erkannt, wenn L von einem ε-NEA erkannt wird***

    =>„*ε-NEA* und *NEA* sind gleich mächtig!“
  - **Beweiskonstruktion:**

    b) Zeigen, dass zu jedem *ε-NEA* ein *NEA* konstruiert werden kann, der dieselbe Sprache akzeptiert

    => Über die Konstruktion der „***ε-Hülle***“ und Induktion

[Beweis: siehe Hopcroft, et al. S. 89 ff.]

Kurs: Informatik 2 – Teil 2

Endliche Automaten

- **Äquivalenz von NEA und NEA mit ε-Übergängen**
  - **Satz:** ***Eine Sprache L wird von einem NEA genau dann erkannt, wenn L von einem ε-NEA erkannt wird***

    =>„*ε-NEA* und *NEA* sind gleich mächtig!“
  - **Anmerkungen:**
    - In der Praxis ist es häufig einfacher, zunächst einen ***ε-NEA*** zu entwerfen und diesen dann in einen äquivalenten ***NEA*** bzw. schlussendlich äquivalenten ***DEA*** zu überführen (inkl. Optimierung)
    - Häufig weist der äquivalente ***NEA / DEA*** wiederum in etwa (nur) genauso viele Zustände wie ***ε-NEA*** auf …
    - … aber wiederum deutlich mehr Zustandsübergänge

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

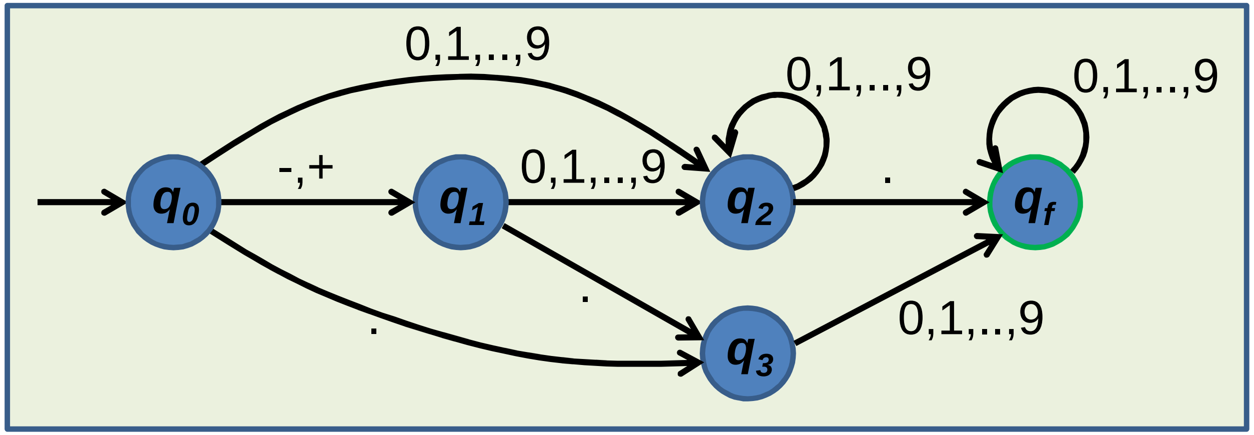
Formale Sprachen / Automaten-theorie

Endliche Automaten

NEA

Endliche Automaten

- **NEA mit ε-Übergängen**
  - **Beispiel: Ein äquivalenter (zum *ε-NEA* vom vorherigen Beispiel) *DEA*, der alle Dezimalzahlen akzeptiert**



MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Endliche Automaten – Reguläre Sprachen

- **Äquivalenz von RA und endlichen Automaten**
  - **Satz:** ***Die von regulären Ausdrücken definierten Sprachen entsprechen den Sprachen, die durch einen DEA definiert werden***

=>„*RA* und *DEA* sind gleich mächtig!“

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

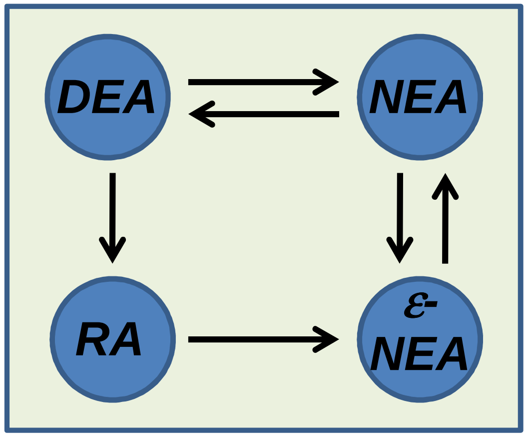
Endliche Automaten – Reguläre Sprachen

Endliche Automaten – Reguläre Sprachen

- **Äquivalenz von RA und endlichen Automaten**
  - **Satz:** ***Die von regulären Ausdrücken definierten Sprachen entsprechen den Sprachen, die durch einen DEA definiert werden***

=>„*RA* und *DEA* sind gleich mächtig!"

  - **Beweiskonstruktion:**



MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten – Reguläre Sprachen

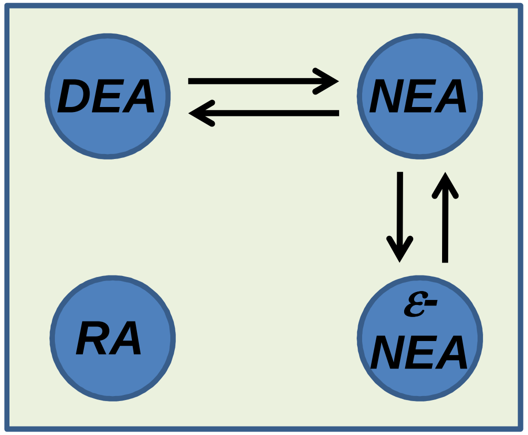
Kurs: Informatik 2 – Teil 2

Endliche Automaten – Reguläre Sprachen

- **Äquivalenz von RA und endlichen Automaten**
  - **Gezeigt wurde bereits:**

    a) **Äquivalenz von *NEA* und *DEA***

    b) **Äquivalenz von *NEA* und *ε-NEA***



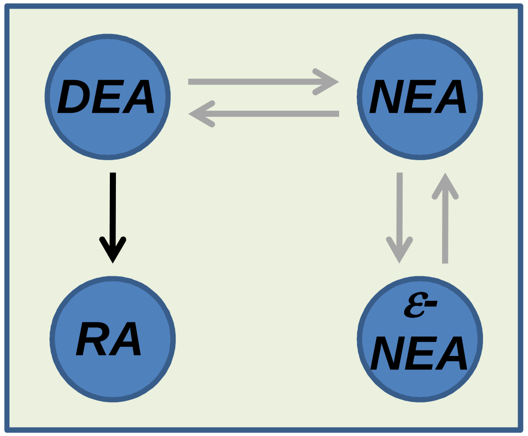
Kurs: Informatik 2 – Teil 2

Endliche Automaten – Reguläre Sprachen

- **Äquivalenz von RA und endlichen Automaten**
  - **Zu zeigen:**

    a) **Für jeden beliebigen *DEA A* mit *L = L(A)* gibt es einen Regulären Ausdruck *R* mit *L = L(R)*** [d. h. ***L(A) = L(R)***]

    Beweiskonstruktion:



Kurs: Informatik 2 – Teil 2

Endliche Automaten – Reguläre Sprachen

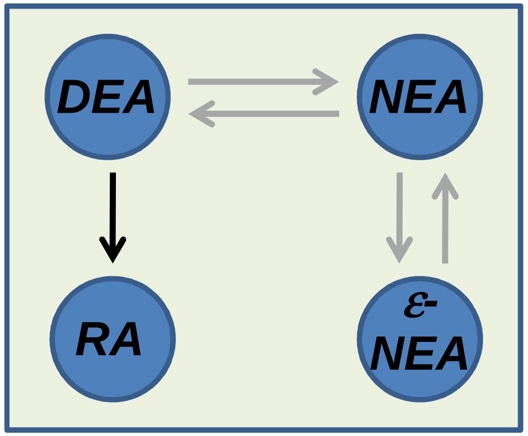
- **Äquivalenz von RA und endlichen Automaten**
  - **Zu zeigen:**

a) **Für jeden beliebigen *DEA A* mit *L = L(A)* gibt es einen Regulären Ausdruck *R* mit *L = L(R)***

Beweiskonstruktion:

=> über Zeichenreihen, die als Beschriftung der Pfade für die Übergänge im ***DEA*** auftreten

=> Induktion über die Länge der Pfade



[Beweis: siehe Hopcroft, et al. S. 101 ff.]

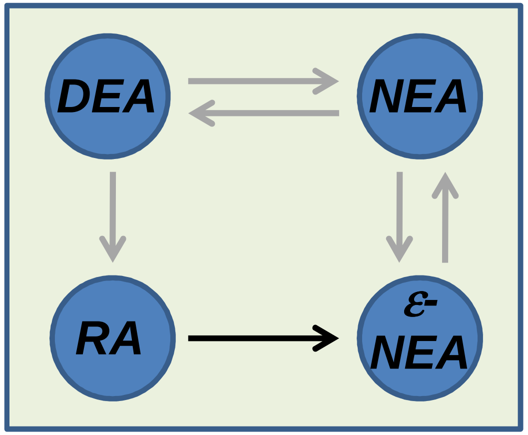
Kurs: Informatik 2 – Teil 2

Endliche Automaten – Reguläre Sprachen

- **Äquivalenz von RA und endlichen Automaten**
  - **Zu zeigen:**

    b) **Für jede Sprache *L = L(R)* eines Regulären Ausdrucks *R* gibt es einen *ε-NEA A* mit *L = L(A)*** [d. h. ***L(A) = L(R)***]

    Beweiskonstruktion:



Kurs: Informatik 2 – Teil 2

Endliche Automaten – Reguläre Sprachen

- **Äquivalenz von RA und endlichen Automaten**
  - **Zu zeigen:**

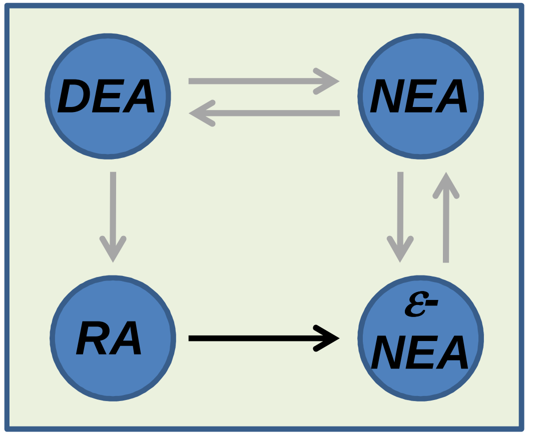
    b) Für jede Sprache *L = L(R)* eines Regulären Ausdrucks *R* gibt es einen *ε-DEA A* mit *L = L(A)* [d. h. ***L(A) = L(R)***]

    Beweiskonstruktion:

    => Über strukturelle Induktion über ***R***, äquivalent dem Aufbau der regulären Ausdrücke:

    => ***DEA*** für ***(ε)***, ***(∅)***, ***(a)*** und der Zusammensetzung über ***Vereinigung***, ***Verkettung*** und der ***Hülle***

    [Beweis: siehe Hopcroft, et al. S. 112 ff.]



Kurs: Informatik 2 – Teil 2

Eigenschaften reguläre Sprachen

- **Abgeschlossenheitseigenschaften**

  L und M seien reguläre Sprachen über Σ. Dann gilt:

  - **Vereinigung: $L \cup M$ ist regulär**
  - **Durchschnitt: $L \cap M$ ist regulär**
  - **Komplement: das Komplement von L ist regulär**
  - **Differenz: $L - M$ ist regulär**
  - **Spiegelung: die Spiegelung von L ist regulär**
    - Spiegelung einer Zeichenreihe $a_1 a_2 \ldots a_n$ ist $a_n \ldots a_2\, a_1$
    - Spiegelung eine Sprache L (mit L_R bezeichnet) beinhaltet alle gespiegelten Zeichenreihen aus L

  Beispiel:

Kurs: Informatik 2 – Teil 2

Eigenschaften reguläre Sprachen

- **Abgeschlossenheitseigenschaften**

  L und M seien reguläre Sprachen über Σ. Dann gilt:

  - **Vereinigung: $L \cup M$ ist regulär**
  - **Durchschnitt: $L \cap M$ ist regulär**
  - **Komplement: das Komplement von L ist regulär**
  - **Differenz: $L - M$ ist regulär**
  - **Spiegelung: die Spiegelung von L ist regulär**
    - Spiegelung einer Zeichenreihe $a_1a_2 \dots a_n$ ist $a_n \dots a_2\, a_1$
    - Spiegelung eine Sprache L (mit L_R bezeichnet) beinhaltet alle gespiegelten Zeichenreihen aus L

  Beispiel: $L = \{01, 1001, 100\}$ => $L_R = \{10, 1001, 001\}$

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Reguläre Sprachen

Eigenschaften reguläre Sprachen

- **Abgeschlossenheitseigenschaften (Forts.)**

  L und M seien reguläre Sprachen über Σ. Dann gilt:

  - **Hülle: L^* ist regulär**
  - **Verkettung: $L + M$ ist regulär**
  - **Homomorphismus: $h(L)$ ist regulär**
    - ***Zeichenreihen-Homomorphismus*:**

      h ist eine Funktion, die jedes Symbol aus Σ auf eine Zeichenreihe abbildet,

      entsprechend wird für $h(w)$ für eine Zeichenreihe w gebildet: $h(w) = h(a_1)h(a_2) \dots h(a_n)$ und

      $h(L) = \{h(w) \mid w \in L\}$
  - **Inverse Homomorphismus: $h^{-1}(L)$ ist regulär**

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Eigenschaften reguläre Sprachen

- **Entscheidbarkeit regulärer Sprachen**

  Interessante Fragestellungen sind die folgenden:

  1) **Gegeben ist eine Sprache *L*** (in beliebiger Repräsentation)

     Frage: ***Ist L leer?***

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Reguläre Sprachen

Eigenschaften reguläre Sprachen

- **Entscheidbarkeit regulärer Sprachen**

  Interessante Fragestellungen sind die folgenden:

  1) **Gegeben ist eine Sprache** ***L*** (in beliebiger Repräsentation)

     Frage: ***Ist L leer?***

  2) **Gegeben ist eine Sprache** ***L*** (in beliebiger Repräsentation) **und eine Zeichenreihe** ***w***

     Frage: ***Ist w ein Wort aus L?*** (d. h. $w \in L$)

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Reguläre Sprachen

Eigenschaften reguläre Sprachen

- **Entscheidbarkeit regulärer Sprachen**

Interessante Fragestellungen sind die folgenden:

1) **Gegeben ist eine Sprache *L*** (in beliebiger Repräsentation)

   Frage: ***Ist L leer?***

2) **Gegeben ist eine Sprache *L*** (in beliebiger Repräsentation) **und eine Zeichenreihe *w***

   Frage: ***Ist w ein Wort aus L?*** (d. h. $w \in L$)

3) **Gegeben sind zwei Sprachen L_1 und L_2** (in beliebiger Repräsentation)

   Frage: ***Gilt $L_1 = L_2$*** (d. h. beschreiben die beiden Repräsentationen dieselbe Sprache?)

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Eigenschaften reguläre Sprachen

- **Entscheidbarkeit regulärer Sprachen**

  1) **Gegeben ist eine Sprache *L*** (in beliebiger Repräsentation)

     Frage: ***Ist L leer?***

     Antwort: ***Ist entscheidbar!***

HOCHSCHULE FÜR TECHNIK ZÜRICH

Einführung

Formale Sprachen / Automaten-theorie

Reguläre Sprachen

Kurs: Informatik 2 – Teil 2

Eigenschaften reguläre Sprachen

- **Entscheidbarkeit regulärer Sprachen**

  1) **Gegeben ist eine Sprache *L*** (in beliebiger Repräsentation)

     Frage: ***Ist L leer?***

     Antwort: ***Ist entscheidbar!***

     - Liegt eine explizite Auflistung der Sprache (der Zeichen-reihen) vor, ist die Beantwortung trivial; in der Regel liegt aber eine Repräsentation als ***RA*** oder ***EA*** vor.
     - Für jeden ***EA*** ist aber die Frage, ob er vom **Startzustand** einen **akzeptierenden Zustand erreichen kann** (nicht leer ist) ebenfalls einfach zu beantworten
     - Neben der Äquivalenz von ***RA*** und ***EA*** kann aber auch für jeden ***RA*** einfach gezeigt werden, dass ein gegebe-ner ***RA*** die leere Sprache repräsentiert oder nicht

Kurs: Informatik 2 – Teil 2

Eigenschaften reguläre Sprachen

- **Entscheidbarkeit regulärer Sprachen**

  2) **Gegeben ist eine Sprache** ***L*** (in beliebiger Repräsentation) **und eine Zeichenreihe** ***w***

  Frage: ***Ist w ein Wort aus L?*** (d. h. $w \in L$)

  Antwort: ***Ist entscheidbar!***

Kurs: Informatik 2 – Teil 2

Eigenschaften reguläre Sprachen

- **Entscheidbarkeit regulärer Sprachen**

  2) **Gegeben ist eine Sprache *L*** (in beliebiger Repräsentation) **und eine Zeichenreihe *w***

  Frage: ***Ist w ein Wort aus L?*** (d. h. $w \in L$)

  Antwort: ***Ist entscheidbar!***

  - Für jeden ***DEA*** ist die Frage, ob er ***w*** akzeptiert, einfach zu beantworten (der ***DEA*** befindet sich nach Eingabe von ***w*** in einem akzeptierenden Zustand)
  - Liegt ***L*** in einer anderen Repräsentation vor, kann ***L*** in einen ***DEA*** überführt werden und anschliessend der Test durchgeführt werden

Kurs: Informatik 2 – Teil 2

Eigenschaften reguläre Sprachen

- **Entscheidbarkeit regulärer Sprachen**

  3) **Gegeben sind zwei Sprachen *L_1* und *L_2*** (in beliebiger Repräsentation)

  Frage: *Gilt $L_1 = L_2$* (d. h. beschreiben die beiden Repräsentationen dieselbe Sprache?)

  Antwort: *Ist entscheidbar!*

  - **Beweiskonstruktion:**

    Schritt 1: beide Repräsentationen in äquivalente ***DEA D1*** und ***D2*** umformen

    Schritt 2: ***D1*** und ***D2*** jeweils minimieren$^{(+)}$

    Schritt 3: Zeigen, dass ***D1*** und ***D2*** gleich sind

    [Beweis: siehe Hopcroft, et al. S. 164 ff.]

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Reguläre Sprachen

Backup(+)

- **Minimierung von EAs**
  - **Mit Hilfe des „*Table-Filling-Algorithums*" kann zu jedem beliebigen *DEA D* ein äquivalenter *DEA* D_m gefunden werden, der eine minimale Anzahl von Zuständen aufweist**

    (d. h. es gibt keinen zu ***D*** **äquivalenten** ***DEA*** D_1, der **weniger Zustände** als D_m hat)
  - Die **Optimierung** von ***EA***s hatte in Zusammenhang mit dem **Entwurf integrierter Schaltkreise** eine grosse Bedeutung

    [Beweis: siehe Hopcroft, et al. S. 164 ff.]

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2

Nicht reguläre Sprachen

- **Existenz nicht regulärer Sprachen**

  Auf Grund der vielen Operationen, unter denen die *regulären Sprachen* abgeschlossen sind, könnte man annehmen, Sprachen sind immer reguläre Sprachen!

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Nicht reguläre Sprachen

Nicht reguläre Sprachen

- **Existenz nicht regulärer Sprachen**

  Auf Grund der vielen Operationen, unter denen die *regulären Sprachen* abgeschlossen sind, könnte man annehmen, Sprachen sind immer reguläre Sprachen!

  1) ***Gibt es Sprachen, die nicht regulär sind?***
  2) ***Wie lässt sich beweisen, dass eine Sprache nicht regulär ist?***

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Nicht reguläre Sprachen

- **Existenz nicht regulärer Sprachen**

1) *Gibt es Sprachen, die nicht regulär sind?*

Beispiel:

$L = \{0^n 1^n \mid n \geq 0\}$ $\quad$ $(L = \{01, 0011, 000111, \ldots\}$

Grundidee:

Kurs: Informatik 2 – Teil 2

Nicht reguläre Sprachen

- **Existenz nicht regulärer Sprachen**

1) ***Gibt es Sprachen, die nicht regulär sind?***

Beispiel:

$L = \{0^n 1^n \mid n \geq 0\}$ $(L = \{01, 0011, 000111, \ldots\}$

Grundidee:

- Ein ***DEA***, der ein Wort aus ***L*** akzeptieren soll, muss sich „*quasi*“ die Anzahl der eingelesenen Nullen „***merken***“
- Da ein ***DEA*** aber nur endlich viele Zustände haben kann, z. B. ***m***, kann er Wörter aus ***L*** mit ***n > m*** nicht akzeptieren
- Folglich kann ***L*** keine reguläre Sprache sein

Kurs: Informatik 2 – Teil 2

Nicht reguläre Sprachen

- **„Pumping-Lemma" für reguläre Sprachen**
  - **Satz:** *L sei eine reguläre Sprache. Dann gibt es eine Konstante n, so dass jede Zeichenreihe w, w $\in$ L, in drei Teilzeichenreihen w = xyz derart zerlegt werden kann, dass gilt:*
    1) $y \neq \varepsilon$
    2) $|xy| \leq n$ *und*
    3) $xy^kz \in L$, *für alle* $k \geq 0$

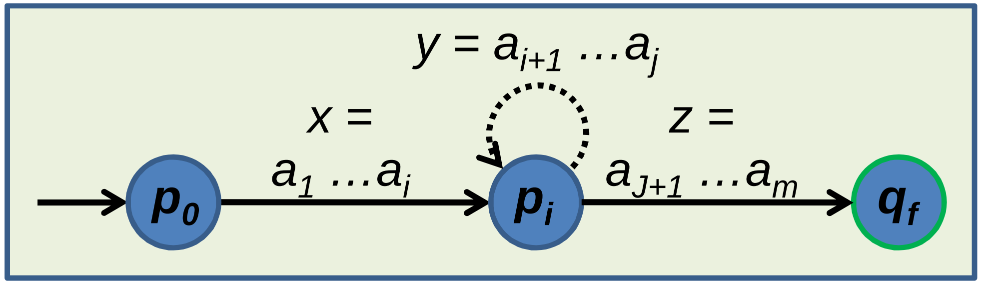
    Beweiskonstruktion:

Kurs: Informatik 2 – Teil 2

Nicht reguläre Sprachen

- **„Pumping-Lemma“ für reguläre Sprachen**
  - **Satz:** *L sei eine reguläre Sprache. Dann gibt es eine Konstante n, so dass jede Zeichenreihe w, $w \in L$, in drei Teilzeichenreihen w = xyz derart zerlegt werden kann, dass gilt:*
    1) $y \neq \varepsilon$
    2) $|xy| \leq n$ *und*
    3) $xy^k z \in L$, *für alle* $k \geq 0$

Beweiskonstruktion:



[Ausführlicher Beweis: siehe Hopcroft, et al. S. 136, 137]

Kurs: Informatik 2 – Teil 2

Nicht reguläre Sprachen

- **„Pumping-Lemma“ für reguläre Sprachen**

  Beispiel: $L = \{0^m 1^m \mid m \geq 0\}$ **und Annahme** L **sei** ***regulär***

Kurs: Informatik 2 – Teil 2

Nicht reguläre Sprachen

- **„Pumping-Lemma" für reguläre Sprachen**

  Beispiel: $L = \{0^m 1^m \mid m \geq 0\}$ **und Annahme** L **sei** ***regulär***

  - Dann müsste es nach dem ***Pumping-Lemma*** die Konstante n geben, so dass die Zeichenreihe $w = 0^n 1^n$ in L ist
  - Nun muss $w = 0^n 1^n$ in xyz zerlegt werden. Da $y \neq \varepsilon$ und $|xy| \leq n$ gilt, können x und y nur aus Nullen bestehen

Kurs: Informatik 2 – Teil 2

Nicht reguläre Sprachen

- **„Pumping-Lemma" für reguläre Sprachen**

  Beispiel: $L = \{0^m 1^m \mid m \geq 0\}$ **und Annahme** L **sei** ***regulär***

  - Dann müsste es nach dem ***Pumping-Lemma*** die Konstante n geben, so dass die Zeichenreihe $w = 0^n 1^n$ in L ist
  - Nun muss $w = 0^n 1^n$ in xyz zerlegt werden. Da $y \neq \varepsilon$ und $|xy| \leq n$ gilt, können x und y nur aus Nullen bestehen
  - Das ***Pumping-Lemma*** besagt, dass (auch) $xz \in L$ (für $k = 0$)

    => xz besteht aber aus **weniger Nullen wie Einsen,** da $y \neq \varepsilon$ gilt und somit y aus **mindestens einer Null** besteht, die der Zeichenreihe xz eben fehlt

    => xz **kann nicht zu** L **gehören, was ein Widerspruch bedeutet, daher kann** L **nicht regulär sein**

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Einführung

Formale Sprachen / Automaten-theorie

Endliche Automaten

Endliche Automaten

- **Fragen ?**

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE FÜR TECHNIK ZÜRICH

Kurs: Informatik 2 – Teil 2

Zusätzliche Übungsaufgabe

MITGLIED DER ZÜRCHER FACHHOCHSCHULE

HOCHSCHULE
FÜR TECHNIK
ZÜRICH

Kurs: Informatik 2 – Teil 2 - Aufgabe

Zusätzliche Übungs-aufgabe

Zusätzliche Übungsaufgabe 1

Gesucht sei der ***EDA***, der die Sprache L_{GZ} der ganzen Zahlen erkennt (typischerweise ganzzahlige Dezimal-zahlen in Programmen).

HOCHSCHULE FÜR TECHNIK ZÜRICH

Zusätzliche Übungs-aufgabe

Kurs: Informatik 2 – Teil 2 - Aufgabe

Zusätzliche Übungsaufgabe 1

Gesucht sei der ***EDA***, der die Sprache L_{GZ} der ganzen Zahlen erkennt (typischerweise ganzzahlige Dezimal-zahlen in Programmen).

Lösung:

L_{GZ} ist eine Sprache über dem Alphabet $\Sigma = \{+,-,0,1,\ldots,9\}$

L_{GZ} muss die „*0*" enthalten, d. h. der EDA einen entsprechenden finalen Zustand beinhalten

Alle Zeichenreihen (ausser der „*0*") können mit einem „**+**" oder „**-**" beginnen (müssen aber nicht)

Alle Zeichenreihen (ausser der „*0*") müssen (nach dem optio-nalen Vorzeichen) mindestens ein Symbol aus dem Alphabet $\{1,\ldots,9\}$ besitzen

Alle Zeichenreihen enden mit einer Zeichenreihe Σ^* über dem Alphabet $\{0,1,\ldots,9\}$ (EDA muss finalen Zustand dafür beinhalten)

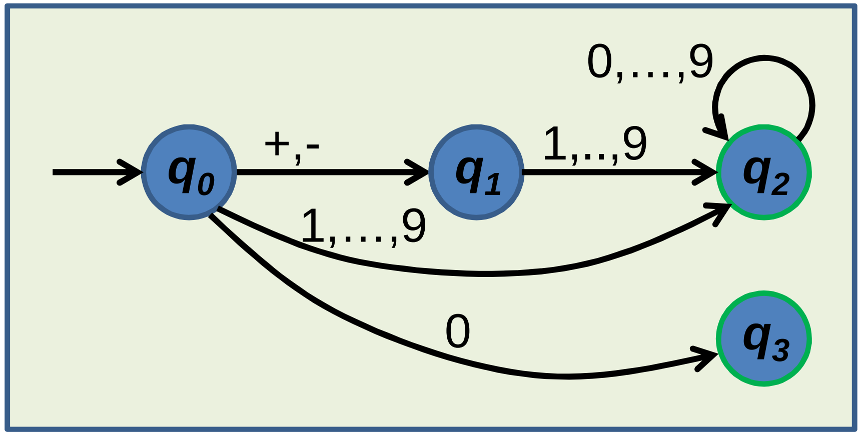
MITGLIED DER ZÜRCHER FACHHOCHSCHULE

Kurs: Informatik 2 – Teil 2 - Aufgabe

Zusätzliche Übungsaufgabe 1

Gesucht sei der ***EDA***, der die Sprache L_{GZ} der ganzen Zahlen erkennt (typischerweise ganzzahlige Dezimal-zahlen in Programmen).

Lösung:



Mögliche Lösung für den *DEA* A_{GZ}:

A_{GZ} =

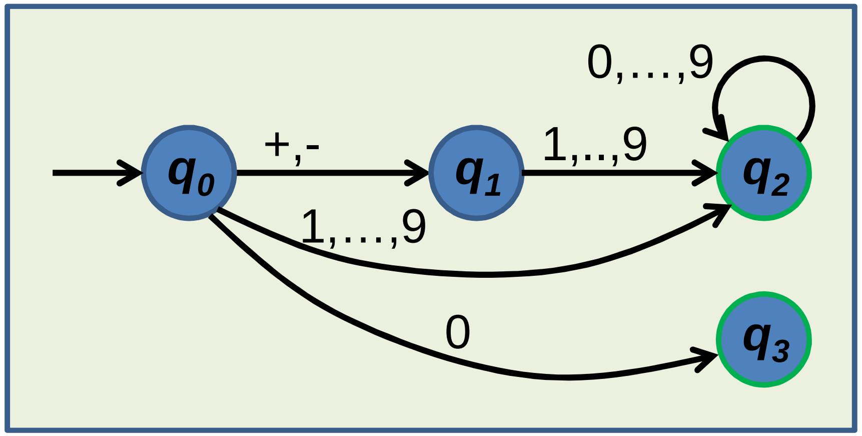
mit δ =

Kurs: Informatik 2 – Teil 2 - Aufgabe

Zusätzliche Übungsaufgabe 1

Gesucht sei der ***EDA***, der die Sprache L_{GZ} der ganzen Zahlen erkennt (typischerweise ganzzahlige Dezimal-zahlen in Programmen).

Lösung:



Mögliche Lösung für den *DEA* A_{GZ}:

$A_{GZ} = (\{q_0, q_1, q_2, q_3\}, \{+,-,0,1,\ldots,9\}, \delta, q_0, \{q_2, q_3\})$

mit $\delta = \{\delta(q_0, 0) = q_3,\ \delta(q_0, -) = q_1,\ (q_0, +) = q_1,\ \delta(q_0, 1) = q_2,\ \ldots,$
$\delta(q_0, 9) = q_2,\ \delta(q_1, 1) = q_2,\ \ldots,\ (q_1, 9) = q_2,\ \delta(q_2, 0) = q_2,\ \ldots,$
$\delta(q_2, 9) = q_2\}$