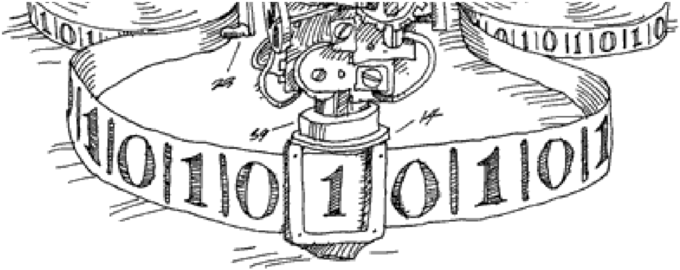
Theoretische Grundlagen der Informatik

Tutorium 1

Institut für Theoretische Informatik



whois tutor

- **Joachim Priesner**
  joachim.priesner@student.kit.edu
  Montag 15:45, SR -108
- **Sebastian Ullrich**
  sebasti@nullrich.de
  Donnerstag 15:45, SR 131
- **Max Wagner**
  max@trollbu.de
  Donnerstag 15:45, SR 301

Organisatorisches – Zum Übungsbetrieb

Karlsruher Institut für Technologie

- **Abgabe:** *Handschriftlich* in Zweiergruppen
- **Schein:**
  - Klausurbonus (1 Notenschritt)
  - Ab 50% der erreichbaren Punkte
- Tutoriumsmaterial und aktueller Punktestand online
  - `http://tinyurl.com/tgi1112`
  - E-Mail-Liste geht rum für
    - Allgemeines Blabla
    - Passwort für Online-Punkteeinsicht

Organisatorisches – Zum Tutorium

- Stoff soll wiederholt werden
- Dabei Fokus auf Übungsbetrieb
- Fragen/Vorschläge/Anmerkungen willkommen!

Kurze Wiederholung: Formale Sprachen

Eine *formale Sprache* L ist eine Teilmenge aller Wörter über einem endlichen Alphabet Σ. Also $L \subseteq \Sigma^*$.

Beispiele:

- $\Sigma = \{0, 1\}, L = \{w11z \mid w, z \in \Sigma^*\}$
  - Die Menge aller Wörter, die ”11” enthalten.

Im Allgemeinen kann man formale Sprachen sehr frei angeben:

- $\Sigma = \{0, 1\}, L = \{w \mid w \in \Sigma^*, w \text{ hat eine gerade Anzahl an 1en}\}$
  - Die Menge aller Wörter, die eine gerade Anzahl an Einsen enthalten.

Kurze Wiederholung: Reguläre Sprachen

Karlsruher Institut für Technologie

Eine Sprache $L \subseteq \Sigma^*$ heißt regulär, wenn für sie einer der folgenden Punkte gilt:

- Verankerung
  - $L = a$ mit $a \in \Sigma^*$ oder
  - $L = \varnothing$
- Induktion: Seien L_1, L_2 reguläre Sprachen.
  - $L = L_1 \cdot L_2$ oder
  - $L = L_1 \cup L_2$ oder
  - $L = L_1^*$

Beispiel ($\Sigma = \{a, b\}$):

- $L_1 = \{w \in \Sigma^* \mid w$ besteht aus einer geraden Anzahl $a\}$
- $L_2 = \{w \in \Sigma^* \mid w$ enthält gleich viele a und $b\}$

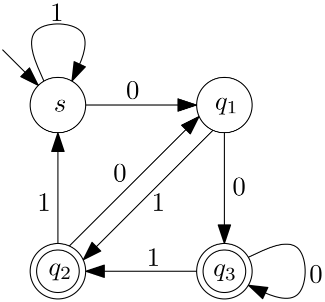
L_1 ist regulär, L_2 nicht.

Deterministische endliche Automaten

Ein deterministischer endlicher Automat M ist ein 5-Tupel

$$M = (Q, \Sigma, \delta, s, F).$$

- Q: endliche Zustandsmenge
- Σ: endliches Alphabet
- δ: Zustandsübergangsfunktion $Q \times \Sigma \to Q$
- s: Startzustand $\in Q$
- F: Endzustandsmenge $\subseteq Q$



Nichtdeterministische endliche Automaten

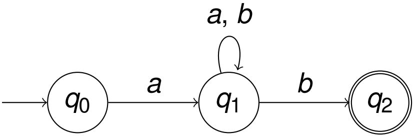
Ein nichtdeterministischer endlicher Automat M ist ein 5-Tupel

$$M = (Q, \Sigma, \delta, s, F).$$

- Q: endliche Zustandsmenge
- Σ: endliches Alphabet
- δ: Zustandsübergangsfunktion $Q \times (\Sigma \cup \varepsilon) \rightarrow 2^Q$
- s: Startzustand $\in Q$
- F: Endzustandsmenge $\subseteq Q$

Damit der NEA ein Wort akzeptiert, muss es *einen* akzeptierenden Weg geben.

NEA: Beispiel

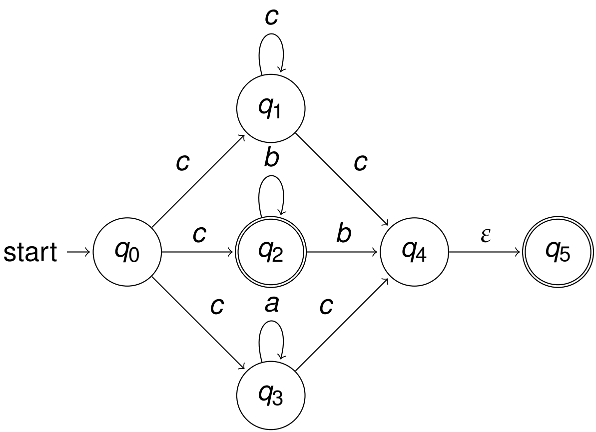


Bei Eingabe von b im Zustand q_1 gibt es mehrere Möglichkeiten.

(siehe Berechnungsbaum an der Tafel).

NEA: Aufgabe

Welche Sprache akzeptiert der nichtdeterministische endliche Automat zu dem folgenden Zustandsgraphen?

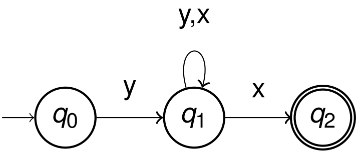


NEA: Aufgabe

Über dem Alphabet $\Sigma = \{a, b\}$ sei der reguläre Ausdruck $r := (a \cup (ab(b)^*ba))^*$ gegeben.
Geben Sie einen NEA an, der $L(r)$ erkennt. Begründen Sie kurz die Korrektheit Ihres Automaten, ein formaler Korrektheitsbeweis ist jedoch nicht erforderlich.
(Hinweis: Es gibt einen NEA mit 3 Zuständen.)

Potenzmengenkonstruktion

Zu jedem nichtdeterministischen endlichen Automaten existiert ein äquivalenter deterministischer endlicher Automat.



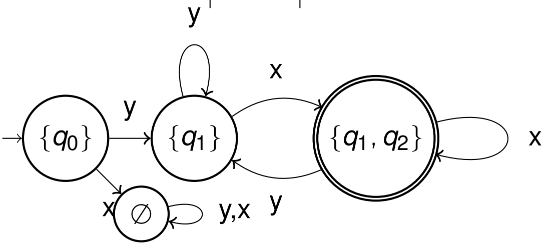
In eine Tabelle werden die Automatenzustände und ihre Folgezustände bei jeweiliger Eingabe eingetragen.

	y	x
$\{q_0\}$	$\{q_1\}$	$\emptyset$
$\{q_1\}$	$\{q_1\}$	$\{q_1, q2\}$

Potenzmengenkonstruktion

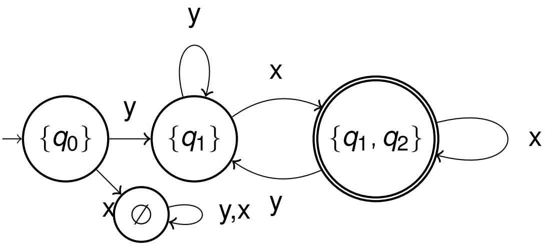
Ein neuer Zustand entsteht, wenn man von einem alten Zustand durch eine Eingabe in mehrere Zustände kommt.

	y	x
$\{q_0\}$	$\{q_1\}$	$\emptyset$
$\{q_1\}$	$\{q_1\}$	$\{q_1, q_2\}$
$\{q_1, q_2\}$	$\{q_1\}$	$\{q_1, q_2\}$
$\emptyset$	$\emptyset$	$\emptyset$



Potenzmengenkonstruktion

Die Einträge der ersten Spalte sind die neuen Zustände. Alle Mengen, die einen Endzustand enthalten, sind wiederum im neuen Automaten Endzustände.



Eliminierung von ε-Übergängen

Karlsruher Institut für Technologie

Satz 2.13 (Skript)

- Zu jedem nichtdeterministischen endlichen Automaten mit ε-Übergängen gibt es einen äquivalenten nichtdeterministischen endlichen Automaten ohne ε-Übergänge, der nicht mehr Zustände hat.
- äquivalent = akzeptiert dieselbe Sprache.

Erinnerung

Der ε-Abschluss $E(q)$ eines Zustandes q ist definiert als die Menge aller Zustände, die von q aus durch lediglich ε-Übergänge erreichbar sind (q selbst zählt auch dazu).

Eliminierung von ε-Übergängen

Karlsruher Institut für Technologie

Konstruktion

Zu einem NEA $A := (Q, \Sigma, \delta, s, F)$ mit ε-Übergängen konstruieren wir einen äquivalenten NEA $\tilde{A} := (\tilde{Q}, \Sigma, \tilde{\delta}, \tilde{s}, \tilde{F})$ mit

- $\tilde{Q} := (Q \backslash F) \cup \tilde{F}$
- $\tilde{s} := s$
- $\tilde{F} := \{q | E(q) \cap F \neq \emptyset\}$
- $\tilde{\delta}(q, a) := \begin{cases} \{q\} & \text{falls } a = \varepsilon \\ \delta(E(q), a) & \text{sonst} \end{cases}$

Eigenschaften von $\tilde{A}$

$L(\tilde{A}) = L(A)$, und $|\tilde{Q}| = |Q|$.

NEA2DEA: Aufgabe

Gegeben sei der NEA $\mathcal{A} = (\{s, q, f\}, \{a, b, c\}, \delta, s, \{f\})$, wobei die Übergangsfunktion δ gegeben ist durch:

	ε	a	b	c
s	$\{q, f\}$	$\emptyset$	$\{q\}$	$\{f\}$
q	$\emptyset$	$\{s\}$	$\{f\}$	$\{s, q\}$
f	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

1. Geben Sie zu dem Automaten $\mathcal{A}$ den Übergangsgraphen an und eliminieren Sie die ε-Übergänge.
2. Ermitteln Sie mittels Potenzmengenkonstruktion den zu $\mathcal{A}$ äquivalenten DEA. Geben Sie hierbei die Übergangsfunktion tabellarisch an.

Pumping Lemma

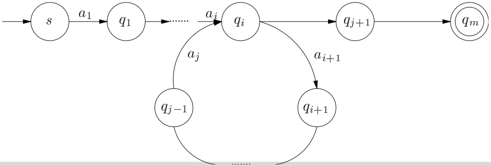
Pumping Lemma

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, sodass für jedes Wort $w \in L$ mit $|w| > n$ eine Darstellung

$$w = uvx$$

existiert, so dass folgende Eigenschaften erfüllt sind:

1. $v \neq \varepsilon$
2. $|uv| \leq n$
3. Für alle $i \in \mathbb{N}_0$ gilt: $uv^i x \in L$



Pumping Lemma: Übersicht

- Jede reguläre Sprache erfüllt das Pumping Lemma. Aber: Nicht jede Sprache, die das Pumping Lemma erfüllt, ist regulär!
- In der Übung wird üblicherweise die Kontraposition des Pumping-Lemmas verwendet: Man zeigt für eine Sprache, dass das Pumping-Lemma *nicht* erfüllt ist, woraus folgt, dass diese Sprache *nicht* regulär sein kann.
- $\neg\left[\exists n \in \mathbb{N} : \forall w \in L, |w| > n : \exists uvx = w : \ldots \forall i \in \mathbb{N} : uv^i x \in L\right]$
  $\Leftrightarrow \forall n \in \mathbb{N} : \exists w \in L, |w| > n : \forall uvx = w : \ldots \exists i \in \mathbb{N} : uv^i x \notin L$
  - Finden wir für *jedes n ein w* mit $|w| > n$, so dass für *jede* Darstellung $w = uvx$ mit $v \neq \varepsilon$ sowie $|uv| \leq n$ ein $i \in \mathbb{N}_0$ existiert mit $uv^i x \notin L$, dann ist L nicht regulär.

Pumping Lemma: Übersicht

- Jede reguläre Sprache erfüllt das Pumping Lemma. Aber: Nicht jede Sprache, die das Pumping Lemma erfüllt, ist regulär!
- In der Übung wird üblicherweise die Kontraposition des Pumping-Lemmas verwendet: Man zeigt für eine Sprache, dass das Pumping-Lemma *nicht* erfüllt ist, woraus folgt, dass diese Sprache *nicht* regulär sein kann.
- $\neg \left[\exists n \in \mathbb{N} : \forall w \in L, |w| > n : \exists uvx = w : \ldots \forall i \in \mathbb{N} : uv^i x \in L\right]$
  $\Leftrightarrow \forall n \in \mathbb{N} : \exists w \in L, |w| > n : \forall uvx = w : \ldots \exists i \in \mathbb{N} : uv^i x \notin L$
  - Finden wir für *jedes n ein w* mit $|w| > n$, so dass für *jede* Darstellung $w = uvx$ mit $v \neq \varepsilon$ sowie $|uv| \leq n$ ein $i \in \mathbb{N}_0$ existiert mit $uv^i x \notin L$, dann ist L nicht regulär.

Pumping Lemma: Übersicht

- Jede reguläre Sprache erfüllt das Pumping Lemma. Aber: Nicht jede Sprache, die das Pumping Lemma erfüllt, ist regulär!
- In der Übung wird üblicherweise die Kontraposition des Pumping-Lemmas verwendet: Man zeigt für eine Sprache, dass das Pumping-Lemma *nicht* erfüllt ist, woraus folgt, dass diese Sprache *nicht* regulär sein kann.
- $\neg \left[\exists n \in \mathbb{N} : \forall w \in L, |w| > n : \exists uvx = w : \ldots \forall i \in \mathbb{N} : uv^i x \in L\right]$
  $\Leftrightarrow \forall n \in \mathbb{N} : \exists w \in L, |w| > n : \forall uvx = w : \ldots \exists i \in \mathbb{N} : uv^i x \notin L$
  - Finden wir für *jedes n ein w* mit $|w| > n$, so dass für *jede* Darstellung $w = uvx$ mit $v \neq \varepsilon$ sowie $|uv| \leq n$ ein $i \in \mathbb{N}_0$ existiert mit $uv^i x \notin L$, dann ist L nicht regulär.

Beispiel

Sei $\Sigma = \{a, b\}$ und $L = \{a^n b^n \mid n \geq 0\}$. (Also $L = \{\varepsilon, ab, aabb, aaabbb, \ldots\}$)

1. Angenommen, L sei regulär. Sei dann n wie im Pumping Lemma.
2. Wähle das Wort $w = a^n b^n$.
3. Es ist also $|w| > n$.
4. Nun ist aber für *jede* Darstellung $w = uvx$ mit $|uv| \leq n$ und $v \neq \varepsilon$ $v = a^m$ mit $m \geq 1$. Demnach ist $uv^0x = a^l b^n \neq L$, da $l < n$.
5. Daher kann L nicht regulär sein.

Pumping Lemma: Aufgaben

Welche der folgenden Sprachen sind regulär? Begründen Sie Ihre Antwort.

1. Die Menge aller Wörter über $\{0, 1\}$, sodass auf jede Null eine Eins folgt
2. $L_1 = \{ww^R | w \in \{a, b\}^*\}$, wobei w^R das "Spiegelwort" zu w ist (Sprache der Palindrome gerader Länge)
3. $L_2 = \{a^i b^j c^k \mid i < j < k\}$.

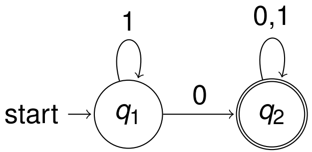
Konstruktion eines RA aus einem DEA

Wir wissen: Zu jedem DEA gibt es einen regulären Ausdruck, der genau die Sprache beschreibt, die der Automat akzeptiert. Wie konstruiert man nun diesen RA aus dem DEA?

Idee: Betrachte die Sprachen L_{q_r,i,q_t}, definiert als $w \in \Sigma^*$ mit w überführt q_r in q_t unter Benutzung der Zwischenzustände $\{q_1, \ldots, q_i\}$

- Es ist $L = \cup_{f \in F} L_{s,n,f}$
- Es ist weiterhin $L_{q_r,i+1,q_t} = L_{q_r,i,q_t} \cup (L_{q_r,i,q_{i+1}}(L_{q_{i+1},i,q_{i+1}})^* L_{q_{i+1},i,q_t})$
- Letztlich ist $L_{q_r,0,q_t}$ immer regulär, denn das sind die Zeichen, mit denen man von q_r nach q_t kommt, ohne weitere Zustände zu verwenden (sowie ε falls $r = t$).
- Unter Benutzung dieser Punkte kann man nun zu einem DEA einen Regulären Ausdruck konstruieren.

Beispiele zum Verständnis

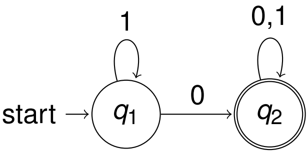


Was sind hier jeweils:

- $L_{q_1,0,q_1}$

Beispiele zum Verständnis

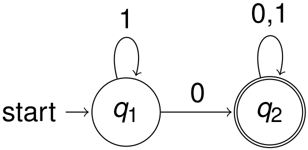
Karlsruher Institut für Technologie



Was sind hier jeweils:

- $L_{q_1,0,q_1} = (1 \cup \varepsilon)$
- $L_{q_1,0,q_2}$

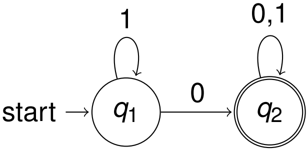
Beispiele zum Verständnis



Was sind hier jeweils:

- $L_{q_1,0,q_1} = (1 \cup \varepsilon)$
- $L_{q_1,0,q_2} = (0)$
- $L_{q_1,1,q_1}$

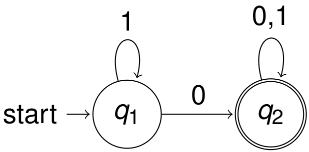
Beispiele zum Verständnis



Was sind hier jeweils:

- $L_{q_1,0,q_1} = (1 \cup \varepsilon)$
- $L_{q_1,0,q_2} = (0)$
- $L_{q_1,1,q_1} = (1^*)$
- $L_{q_2,1,q_2}$

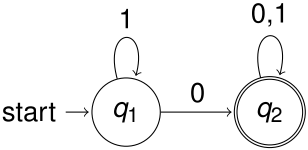
Beispiele zum Verständnis



Was sind hier jeweils:

- $L_{q_1,0,q_1} = (1 \cup \varepsilon)$
- $L_{q_1,0,q_2} = (0)$
- $L_{q_1,1,q_1} = (1^*)$
- $L_{q_2,1,q_2} = (0 \cup 1 \cup \varepsilon)$
- $L_{q_2,2,q_2}$

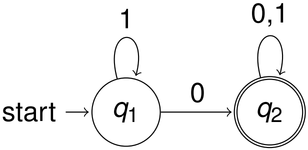
Beispiele zum Verständnis



Was sind hier jeweils:

- $L_{q_1,0,q_1} = (1 \cup \varepsilon)$
- $L_{q_1,0,q_2} = (0)$
- $L_{q_1,1,q_1} = (1^*)$
- $L_{q_2,1,q_2} = (0 \cup 1 \cup \varepsilon)$
- $L_{q_2,2,q_2} = (0 \cup 1)^*$
- $L_{q_1,2,q_2}$

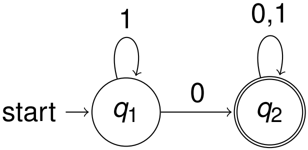
Beispiele zum Verständnis



Was sind hier jeweils:

- $L_{q_1,0,q_1} = (1 \cup \varepsilon)$
- $L_{q_1,0,q_2} = (0)$
- $L_{q_1,1,q_1} = (1^*)$
- $L_{q_2,1,q_2} = (0 \cup 1 \cup \varepsilon)$
- $L_{q_2,2,q_2} = (0 \cup 1)^*$
- $L_{q_1,2,q_2} = 1^*0(0 \cup 1)^*$

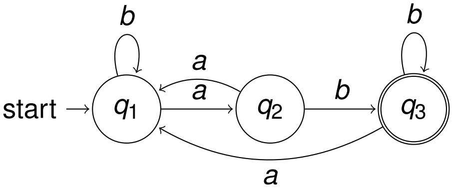
Ausführliche Konstruktion



1. $L_{q_1,2,q_2} = L_{q_1,1,q_2} \cup (L_{q_1,1,q_2}(L_{q_2,1,q_2})^* L_{q_2,1,q_2})$
2. $L_{q_1,1,q_2} = L_{q_1,0,q_2} \cup (L_{q_1,0,q_1}(L_{q_1,0,q_1})^* L_{q_1,0,q_2}) = 0 \cup ((1 \cup \varepsilon)(1 \cup \varepsilon)^* 0)$
3. $L_{q_2,1,q_2} = L_{q_2,0,q_2} \cup (L_{q_2,0,q_1}(L_{q_1,0,q_1})^* L_{q_1,0,q_2}) = (0 \cup 1 \cup \varepsilon)$
4. Also: $L_{q_1,2,q_2} =$
   $0 \cup (1 \cup \varepsilon)(1 \cup \varepsilon)^* 0 \cup ((0 \cup ((1 \cup \varepsilon)(1 \cup \varepsilon)^* 0))(0 \cup 1 \cup \varepsilon)^* (0 \cup 1 \cup \varepsilon))$
5. Vereinfacht: $1^* 0 (0 \cup 1)^*$

Aufgabe

Bestimmen Sie mit dem im Beweis von Satz 2.14 verwendeten Verfahren die reguläre Sprache, die folgender deterministische endliche Automat erkennt:



Bis zum nächsten Mal!

Some people, when confronted with a problem, think "I know, I'll use regular expressions." Now they have two problems. – Jamie Zawinski

Some people, when confronted with a problem, think "I know, I'll quote Jamie Zawinski." Now they have two problems. – Mark Pilgrim

Lizenzen