Vorkurs Theoretische Informatik

Einführung in reguläre Sprachen

Arbeitskreis Theo-Vorkurs

Donnerstag, 13. Oktober 2022

Fachgruppe Informatik Universität Stuttgart

Übersicht

1. Chomsky-Hierarchie

2. Automaten

    NEA

    DEA

3. Wiederholung

Chomsky-Hierarchie

Manche Sprachen sind schwerer zu beschreiben als andere

Wenn wir unsere Grammatiken einschränken, können wir weniger Sprachen beschreiben.

Beispiel

Mit der Einschränkung

Alle Produktionsregeln müssen der Form $A \longrightarrow a$ *oder* $A \longrightarrow aB$ *entsprechen, wobei* $A, B \in V$ *und* $a \in \Sigma$.

können wir Sprachen wie $L_1 = \{a^n \mid n \in \mathbb{N}\}$ beschreiben,

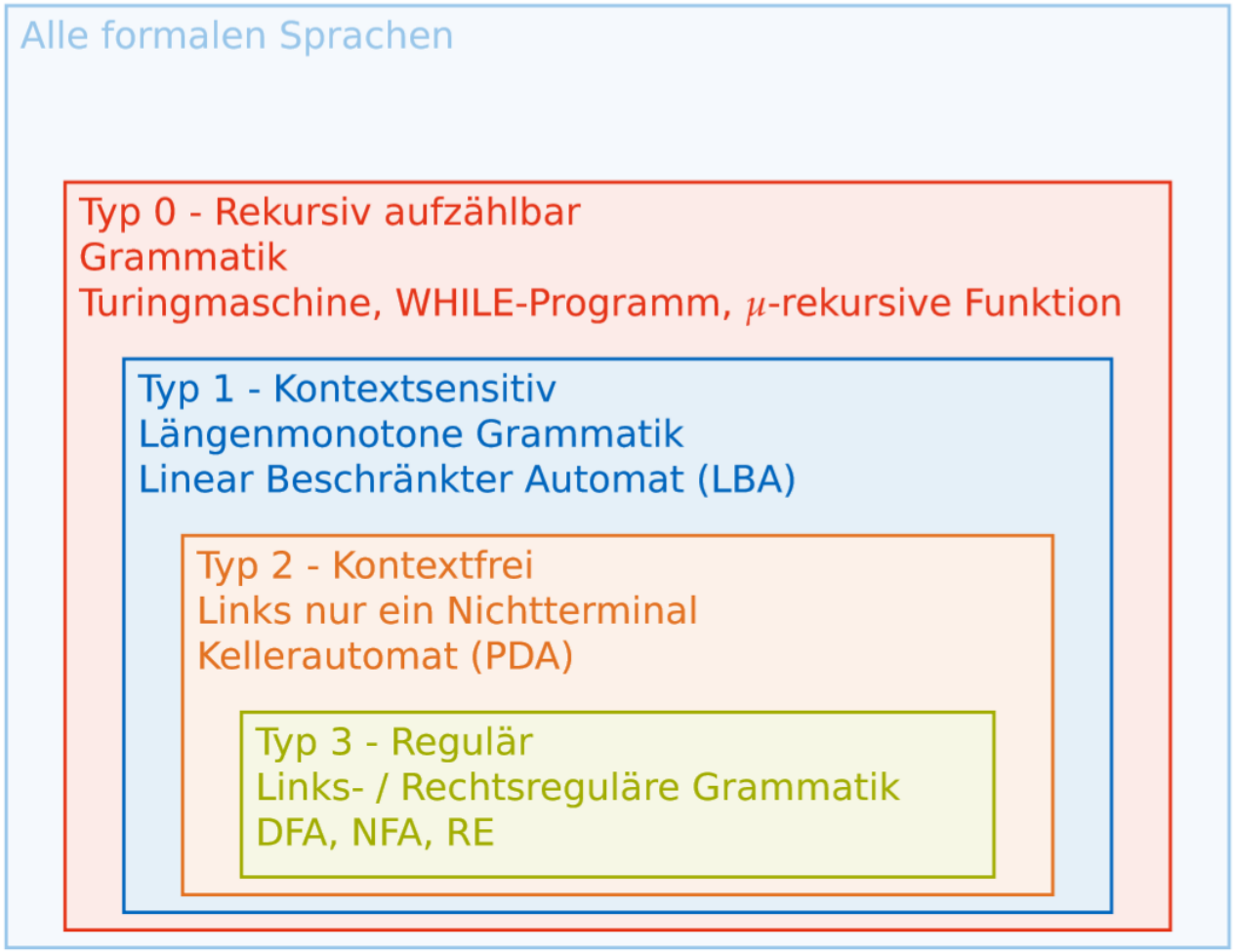
aber nicht mehr Sprachen wie $L_1 = \{a^n b^n \mid n \in \mathbb{N}\}$.

Achtung: ist $\varepsilon \in L$, ist auch $S \longrightarrow \varepsilon$ erlaubt, sofern S nicht auf der rechten Seite einer Produktion vorkommt.

$\rightsquigarrow$ Sprachen, die wir mit dieser starken Einschränkung beschreiben können, nennen wir *regulär* oder vom *Typ 3*.

Es gibt weitere Typen $\rightsquigarrow$ Mehr dazu in der Vorlesung

Manche Sprachen sind schwerer zu beschreiben als andere



Reguläre Grammatik

Aufgaben

Finde Produktionsregeln, die in regulären Grammatiken erlaubt wären, für die folgenden Sprachen

Normal

- $L_1 = \{a^{2n} \mid n \in \mathbb{N}\}$
- $L_2 = \{a^n b^m \mid n, m \in \mathbb{N}\}$
- $L_3 = \{uv \mid u \in \{a, b\}^*,\ v \in \{c, d\}\}$
- $L_4 = \{w \mid |w| = 3, w \in \{a, b, c\}^*\}$

Etwas Schwerer

- $L_5 = \{a^n \mid n \equiv 1 \pmod 3\}$
- $L_6 = \{uv \mid u \in \{◀, ▲, ▶, ▼\}^*,\ v \in \{\text{STOP}\}\}$
- $L_7 = \{w \in \{a, b, c\}^* \mid |w|_a = 3, |w|_b = 1\}$

Automaten

Reguläre Sprachen anders beschreiben

Wir können reguläre Sprachen auch graphisch beschreiben.
Dafür nutzen wir (nicht deterministische) endliche Automaten.
Ein Automat prüft Wörter und entscheidet, ob sie Teil der Sprache sind oder nicht.
$\rightsquigarrow$ Wir nennen das *akzeptieren*, bzw. nicht akzeptieren.

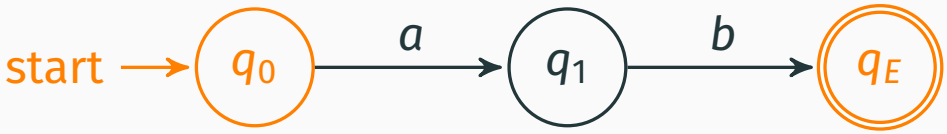
Funktionsweise

1. Ein Wort wird in den Automat eingegeben
2. Wort wird zeichenweise abgearbeitet
3. Nach jedem Zeichen wird der Automat in einen Zustand überführt, der bestimmt, wie fortgefahren wird
4. Befindet sich der Automat in einem *Endzustand*, sobald das Wort abgearbeitet wurde, akzeptiert der Automat das Wort.

Bestandteile eines endlichen Automaten

Ein Automat kann als gerichteter Graph notiert werden.

Wir konstruieren ihn aus den folgenden Komponenten:



Startzustand

Im Startzustand wird das Wort eingegeben.

Endzustand

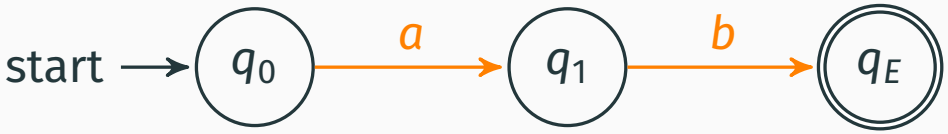
Falls sich der Automat in diesem Zustand befindet, und das Wort abgearbeitet ist, wird das Wort akzeptiert.

Anmerkung: Unter Umständen sind mehrere hiervon nötig

Bestandteile eines endlichen Automaten

Ein Automat kann als gerichteter Graph notiert werden.

Wir konstruieren ihn aus den folgenden Komponenten:

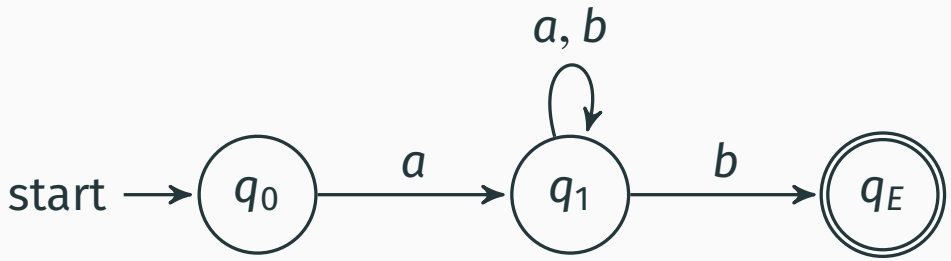


Zustandsübergang

Wird das Zeichen auf dem Übergang *gelesen*, geht der Automat in den folgenden Zustand über.

Beispiel

$L = \{axb \mid x \in \{a, b\}^*\}$



Worteingabe:

$aababb \in L? \rightsquigarrow$ akzeptiert

Zu Beachten

Anmerkungen zu Übergängen bei Automaten:

- ε-Übergänge sind *nicht* möglich:

- Mehrere Endzustände sind möglich:

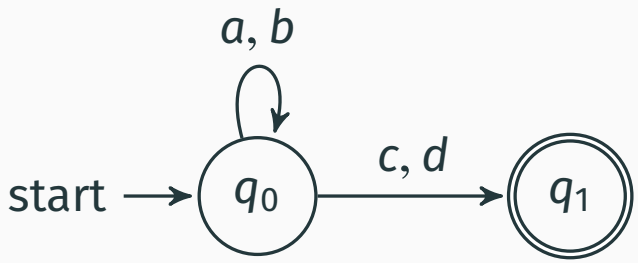
- Verlassen eines Endzustands ist möglich:

- Nicht erreichbare Zustände/Teile sind möglich:

Beispiel: Automat

Gegeben ist eine Sprache L. Gesucht ist ein Automat M, der **genau** die Wörter aus L akzeptiert.

$L_1 = \{uv \mid u \in \{a, b\}^*,\ v \in \{c, d\}\}$



Denkpause

knifflige Aufgabe

Wir entwerfen einen Automat zur Aufzugskontrolle. Der Aufzug hat folgende Möglichkeiten:

$$\Sigma = \{ \boxed{\text{EG}\nearrow\text{1.OG}}, \boxed{\text{EG}\nearrow\text{2.OG}}, \boxed{\text{1.OG}\searrow\text{EG}}, \boxed{\text{1.OG}\nearrow\text{2.OG}}, \boxed{\text{2.OG}\searrow\text{EG}}, \boxed{\text{2.OG}\searrow\text{1.OG}}, \boxed{\text{OFF}} \}$$

- Der Aufzug startet vom Erdgeschoss und darf sich nur aus Stockwerken bewegen, in denen er sich befindet.
- Der Aufzug kann nur im Erdgeschoss ausgeschaltet werden. Er kann dann keine Bewegung durchführen.
- Der Aufzug muss ausgeschaltet werden.

Zeichne einen Automaten an, dessen akzeptierte Sprache genau die Menge der korrekten Abläufe ist.

Denkpause

Aufgaben

Finde Automaten, die **genau** folgende Sprachen erkennen.

Normal

- $L_1 = \{a^n b^m \mid n, m \in \mathbb{N}\}$
- $L_2 = \{w \mid |w| = 3, w \in \{a, b, c\}^*\}$
- $L_3 = \{uv \mid u \in \{◀, ▲, ▶, ▼\}^*,\ v \in \{\text{STOP}\}\}$

Etwas Schwerer

- $L_4 = \{a^n \mid n \equiv 1 \pmod 3\}$
- $L_5 = \{w \in \{a, b, c\}^* \mid |w|_a = 3, |w|_b = 1\}$
- $L_6 = \{w \in \{a, b\}^* \mid |w|_a \equiv |w|_b \pmod 3\}$

Verschiedene endliche Automaten

NEA

Beim Lesen eines Wortes ist es manchmal unklar, welchen Übergang der Automat nehmen soll.

$\rightsquigarrow$ Der Automat ist *nichtdeterministisch*.

DEA

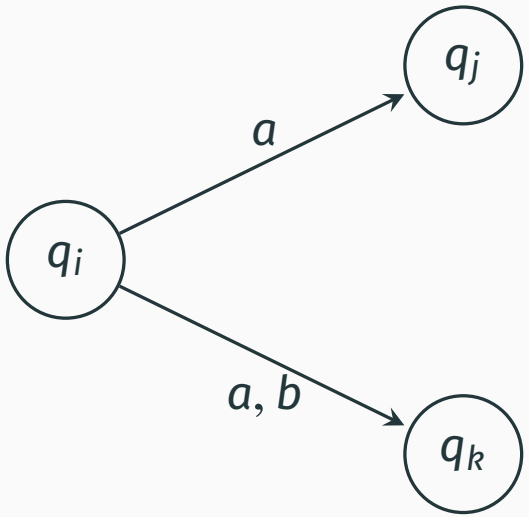
Wir können unsere Möglichkeiten so einschränken, dass bei jedem Zeichen eindeutig ist, welcher Übergang genutzt wird.

$\rightsquigarrow$ Der Automat ist *deterministisch*.

Unterschied NEA und DEA

Bildlich kann man den Unterschied zwischen NEA und DEA an verschiedenen Merkmalen erkennen:

Mehrere Möglichkeiten bei Übergängen:

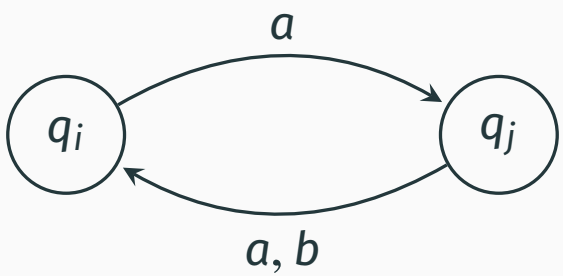


Bei einem DEA wäre nur ein Weg mit einem a von dem Zustand q_i aus erlaubt!

Unterschied NEA und DEA

Bildlich kann man den Unterschied zwischen NEA und DEA an verschiedenen Merkmalen erkennen:

Nicht definierte Übergänge

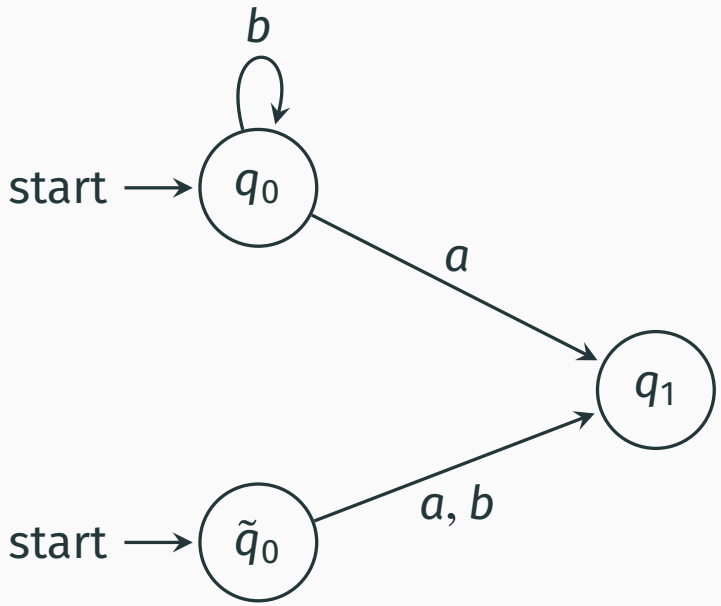


Bei einem DEA müsste ein Übergang mit b für den Zustand q_i definiert werden!

Unterschied NEA und DEA

Bildlich kann man den Unterschied zwischen NEA und DEA an verschiedenen Merkmalen erkennen:

Mehrere Startzustände



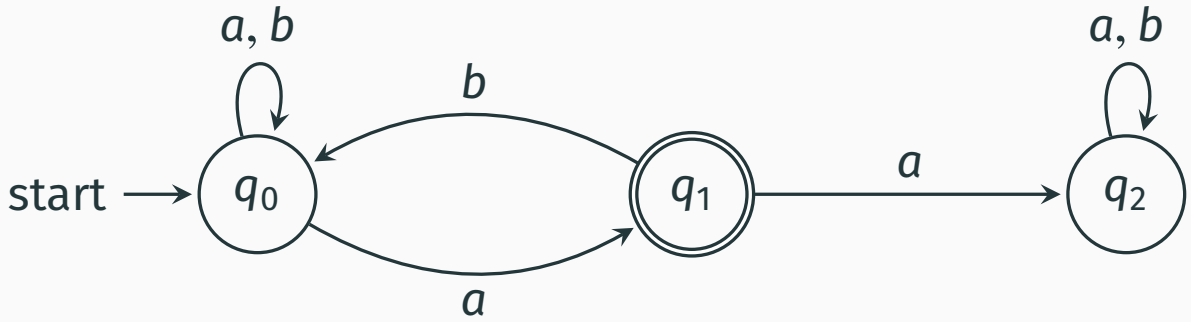
Ein DEA muss genau einen Startzustand besitzen!

Wann wird ein Wort akzeptiert?

Ein Wort $w \in \Sigma^*$ wird von einem NEA akzeptiert, wenn es mindestens einen akzeptierenden Pfad gibt.

Beispiel

Welche Zustände können erreicht werden, wenn $w = aaba$ eingelesen wird?



i) $q_0 \to q_0 \to q_0 \to q_0 \to q_0$

ii) $q_0 \to q_0 \to q_0 \to q_0 \to q_1$ (Endzustand erreicht)

iii) $q_0 \to q_0 \to q_1 \to q_0 \to q_1$ (Endzustand erreicht)

iv) $q_0 \to q_1 \to q_2 \to q_2 \to q_2$

Wir beschränken unseren Automaten folgendermaßen:

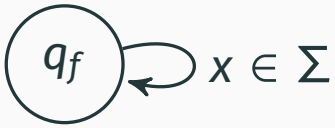
Von jedem Zustand muss genau ein Übergang für jedes $a \in \Sigma$ ausgehen.

Um dies zu ermöglichen führen wir eine neue Komponente ein:

Fangzustand

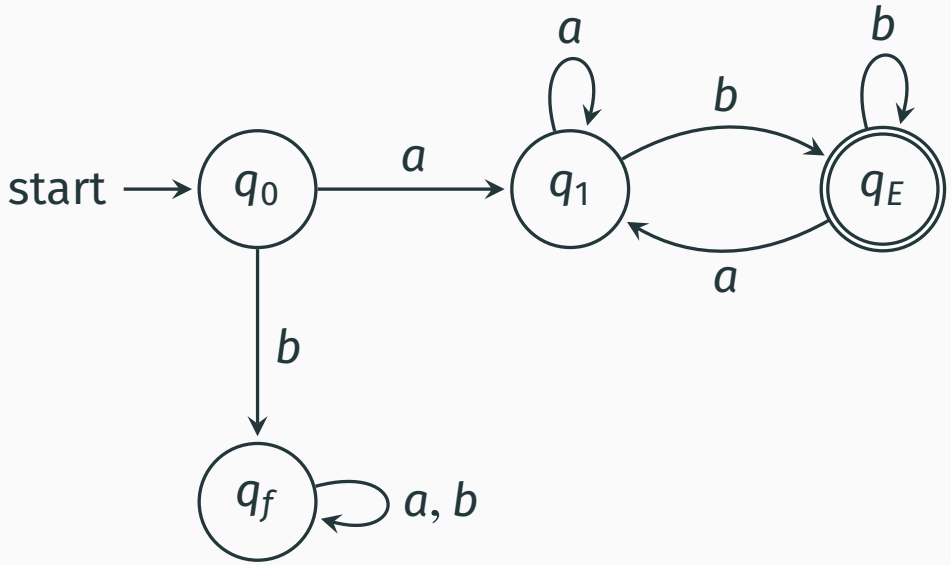
Dieser Zustand kann nicht verlassen werden.

Falls der Automat in diesem Zustand landet, kommt er nicht mehr raus. Das Wort kann nicht akzeptiert werden.



Beispiel

$L = \{axb \mid x \in \{a, b\}^*\}$



Worteingabe:

$aababb \in L? \rightsquigarrow$ akzeptiert

Denkpause

Aufgaben

Finde deterministische endliche Automaten (DEAs) für die folgenden Sprachen.

Normal

- $L_1 = \{a^{2n} \mid n \in \mathbb{N}\}$ über $\Sigma = \{a\}$
- $L_2 = \{w \in \{a,b\}^* \mid |w|_a \geq 2\}$ über $\Sigma = \{a,b\}$

Prüfungsaufgabe: Etwas Schwerer

- $L_3 = \{w \in \{a,b,c\}^* \mid |w|_b \geq 1$ und $aaca$ ist Suffix von w$\}$ über $\Sigma = \{a,b,c\}$

Prüfungsaufgabe: Schwer

- $L_4 = \{bin(n) \mid \exists k \in \mathbb{N} : n = 4k, bin(n)$ ist Binärdarstellung von $n\}$ über $\Sigma = \{1,0\}$
  Achtung: Keine führenden Nullen.

Wiederholung

Das können wir jetzt beantworten

Reguläre Grammatiken

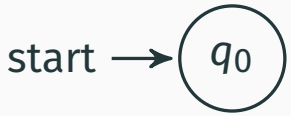
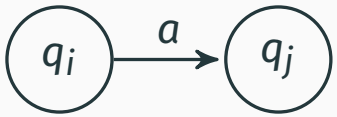
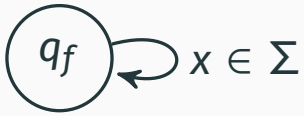
- Wie sehen Produktionsregeln für reguläre Grammatiken aus?
- Bilden einer regulären Grammatik für gegebene, reguläre Sprache

Das können wir jetzt beantworten

Automaten

- Was sind Automaten?
- Wie wandelt man Automaten zu einer äquivalenten Grammatik um?
- Was macht einen deterministischen Automaten aus?
- Finden eines (deterministischen) Automaten für gegebene Sprache

Glossar

Abk.	Bedeutung	Was?!
	Startzustand	Hier fängt der Automat beim Lesen eines Wortes an
	Zustandsübergang	gibt an, welches Symbol eingelesen werden kann, um in den Folgezustand zu übergehen.
	Endzustand	Hier kann ein fertig gelesenes Wort akzeptiert werden.
	Fangzustand	wird benötigt, um Determinismus zu gewährleisten. In Graphiken oft nicht eingezeichnet, ist aber da. Malt den hin.

Lizenz

- Unsere Folien sind frei!
- Jeder darf die Folien unter den Bedingungen der **GNU General Public License v3** (oder jeder späteren Version) weiterverwenden.
- Ihr findet den Quelltext unter `https://www.github.com/FIUS/theo-vorkurs-folien`