Formale Grundlagen der Informatik 1
Kapitel 1
Endliche Automaten

Frank Heitmann
heitmann@informatik.uni-hamburg.de

4. April 2016

Organisatorisches...

Eine typische Woche:

- Mo+Di: Vorlesung
- Di-Fr: Übungsgruppen
  - Neuer Zettel und Bearbeiten der Präsenzaufgaben
  - Abgabe der Übungsaufgaben (der Vorwoche)
  - Rückgabe der Übungsaufgaben (der Vorvorwoche)
- Fr: Neue PDFs:
  - Präsenzlösungen der Aufgaben dieser Woche
  - Musterlösung der gerade abgegebenen Aufgaben
  - Neuer Aufgabenzettel
- Fr: Tutorium

Organisatorisches

Zur **Vorlesung**:

- Mo, 12-14 und Di, 8-10 in ErzWiss H
- Die Vorlesung wird aufgezeichnet!
  - Livestream:
    `https://lecture2go.uni-hamburg.de/livemin`
  - und kurz nach der Vorlesung bei Lecture2Go
- Die Vorlesung ist inhaltlich zweigeteilt:
  - Automatentheorie (bis ca. Mitte Mai)
  - Logik (ab ca. Mitte Mai)

Organisatorisches

Zu den **Übungen**:

- In den Übungen wird der neue Aufgabenzettel verteilt (gibt es i.A. ab Freitag schon online)
- In den Übungen werden i.A. die *Präsenzaufgaben* behandelt:
  - Ü-LeiterIn teilt euch zu Anfang in 3er-Gruppen ein.
  - In diesen ca. 10 Minuten Aufgaben bearbeiten.
  - Ü-LeiterIn geht rum und hilft.
  - Nach 10 Minuten wird "rotiert". Eine Person pro Gruppe wandert im Uhrzeigersinn, eine Person pro Gruppe entgegen des Uhrzeigersinns.
  - Dies wird wiederholt.
  - Besprechen der Präsenzaufgabe im Plenum
  - Wiederholung mit der zweiten Präsenzaufgabe.
- Am Ende der Ü-Gruppe:
  - Abgabe eurer Lösungen des Aufgabenzettels der Vorwoche.
  - Rückgabe eurer Lösungen der Vorvorwoche.

Organisatorisches

Zu den **Abgaben**:

- Die Aufgaben sollen in Arbeitsgruppen bearbeitet werden.
- Eine Arbeitsgruppe soll i.A. aus zwei bis drei Studierenden bestehen.
- Arbeitsgruppen geben nur einen Lösungszettel ab, auf dem alle Namen der Teilnehmer(innen) lesbar notiert sein müssen. Die erreichte Punktzahl wird jedem Gruppenmitglied gutgeschrieben.
- Jedes Arbeitsgruppenmitglied muss die Lösungen seiner Gruppe an der Tafel präsentieren können.
- Es wird voraussichtlich 12 Aufgabenzettel mit Hausaufgaben geben.

Organisatorisches

Die **Kriterien**:

- Übungen:
  - Anwesend sein.
  - Aktiv mitarbeiten.
  - 50% der insgesamt vergebenen Punkte und
  - jeweils $\geq 20\%$ der Punkte auf 10 der 12 Aufgabenzettel.
  - Nicht mogeln ...
    - Kleingruppen geben selbst Geschriebenes ab
    - Quellen müssen angegeben werden
  - Weiteres siehe Webseiten und Modulhandbuch.
- Klausur:
  - 50% der Punkte

  $\Rightarrow$ Am **Samstag, 06.08.2016**, 10:30-12:30 Uhr

Organisatorisches

Zum **Tutorium**:

- Hier werden die in der Woche abgegebenen Übungsaufgaben besprochen.
- Gut zur Klausurvorbereitung.
- Freitag, 16-18 Uhr, B-201 (**ab dem 15.4**).

Organisatorisches

Es wird vor den Klausuren voraussichtlich ein **Repetitorium** geben.

- Termine stehen noch nicht fest.

Organisatorisches

Zur **Literatur**:

- Zwei sehr gute **Bücher**:
  - Hopcroft, John E., Motwani, Rajeev und Ullman, Jeffrey D. (2007) **Introduction to Automata Theory, Languages, and Computation**, 3ed, Pearson/Addison-Wesley (auch auf Deutsch erhltlich).
  - Schöning, Uwe (2000). **Logik für Informatiker**. Spektrum, Akademischer Verlag.

Zum Skript

Auf

`http://fgi1-skript.de`

findet ihr ein **Web-Skript** zur Vorlesung. Dieses Web-Skript

- behandelt ausgewählte Inhalte (insb. die als schwierig geltenden; für anderes: siehe Folien, Bücher, ...)
- ist ein Experiment. Es wird nicht perfekt sein (z.B. bzgl. verschiedener Hardware!) und wir wollen es mit euch gestalten.
- Bitte lest die Abschnitte *Das Skript* und *Annotieren* und beachtet das dort notierte (insb. bzgl. des Annotierungswerkzeuges).

Die einzelnen dort geschriebenen Teile werden auch als PDF-Skript unter `http://www.informatik.uni-hamburg.de/TGI/lehre/vl/SS16/FGI1` veröffentlicht.

Organisatorisches

- Vorlesung und Folien
- Übungsgruppe (Präsenzaufgaben und Übungsaufgaben)
- Skript und Bücher
- Tutorium
- Repetiorium

Bemerkung 1

Anwesenheit in der Vorlesung oder bewusstes, zeitnahes Video schauen ist oft für den Erfolg in der Veranstaltung sehr wichtig! Ebenso ist es wichtig zeitnah in die Literatur zu schauen!

Organisatorisches

Bemerkung 2

Nehmt den Stift in die Hand! Notiert euch Dinge selbst (und handschriftlich). Geht z.B. Folien und Literatur durch und notiert euch, was die wichtigen Dinge der Woche waren. Rechnet Aufgaben (auch dazu: Stift in die Hand nehmen!), redet mit anderen, arbeitet zusammen und investiert Zeit.

Die Boxen...

Wichtige Anmerkung

Wichtige Hinweise, Anmerkungen zur Klausur, ...

Anmerkung

Standard. Definitionen, Sätze, Beweise, “normale” Anmerkungen

Bemerkung

Hinweise, Beispiele, Nebenbemerkungen, ...

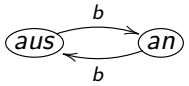
Nebenbemerkung

Literaturhinweis für Interessierte, ...

Motivation ...

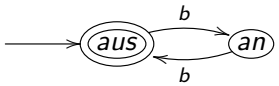
Ein (Licht-)Schalter

Ein einfacher (Licht-)Schalter kann *an* oder *aus* sein. Zwischen diesen *Zuständen* kann man wechseln.

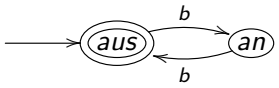


Ein (Licht-)Schalter

Ein einfacher (Licht-)Schalter kann *an* oder *aus* sein. Zwischen diesen *Zuständen* kann man wechseln.

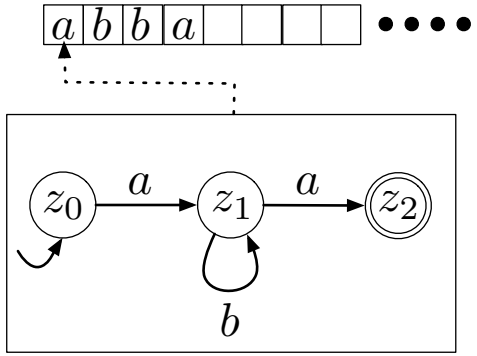


Ein (Licht-)Schalter



- *endlichen Anzahl von Zuständen*;
- davon *einer* als *Startzustand* und
- eine *beliebige Teilmenge* als *Endzustände* hervorgehoben;
- Zustands*übergänge* sind möglich;
- das System ist stets in *genau einem Zustand*.

Endliche Automaten



Um dieses Modell und Erweiterungen davon wird es gehen...
... wobei *a* und *b* hier nur Abstraktionen sind.

Und wozu das Ganze?

Und wozu nun das Ganze? Was kann man damit machen?

- Wir beginnen mit einem recht einfachen Modell (DFA). Da denkt man sich "Was soll ich damit?!".
  - Erstmal lernen! Damit gehen dann später nützliche und spannende Dinge! Mit + und = in einer Programmiersprache geht auch nicht viel, trotzdem muss man die erstmal lernen.
- Grundlage vieler weiterer (Automaten-)Modelle

⇒ Schnelle Lösung des (Wort-)Suchproblems

⇒ Entwicklung von lexikographischen Analysierern, Parsern und Compilern

⇒ Modellierung von Systemen; am Modell können dann Eigenschaften überprüft werden.
- Kommt insb. auch in FGI2.

- Automaten als grundlegende Datenstruktur (für Mengen und Relationen auf denen bestimmte Operationen nötig sind)

Und wozu das Ganze?

Bedeutende Anwendungen

- bei der Modellierung und Verifikation von Systemen
- in der technischen Informatik
- bei Protokollen
- ...

und Erkenntnisse

- Probleme, die von einem Computer **nicht** gelöst werden können
- Probleme, die von einem Computer **praktisch nicht** gelöst werden können

Und wozu das Ganze?

Und ganz wichtig:

- Einüben des abstrakten Denkens und des exakten Argumentierens!
- ... sonst klappt es mit obigem nicht
- ... und dann kann man drüber nachdenken, wie man die Grenzen doch umgeht... ;-)

Und beginnen tun wir jetzt ganz grundlegend ...

Mengen

Definition (Mengen)

- Endliche Menge: $M_1 = \{1, 2, 3, 4\}$
- Unendliche Menge: $M_2 = \{1, 2, 3, \ldots\}$
- Enthalten sein: $3 \in M_1$, $5 \notin M_1$
- Kardinalität: $|M_1| = 4$ ist die Anzahl der Elemente in M_1.

Oft werden die Elemente einer Menge durch eine sie *charakterisierende Eigenschaft* beschrieben:

$$M_3 = \{x \in \mathbb{N} \mid \exists i \in \mathbb{N} : x = 2 \cdot i\} = \{0, 2, 4, 6, \ldots\}$$

Mengen

Definition (Mengen II)

1. **Teilmenge**: $A \subseteq B$ gdw. für jedes $a \in A$ auch $a \in B$ gilt
   - $\{1, 4, 5\} \subseteq \{1, 2, 3, \ldots\}$
   - $\{1, 4, 5\} \subseteq \{1, 4, 5\}$
   - $\{1, 4, 5\} \not\subseteq \{1, 3, 5, 7, \ldots\}$
   - $\{1, 4, 5\} \not\subseteq \{1, 4\}$
2. **Mengengleichheit**: $A = B$ gdw. $A \subseteq B$ **und** $B \subseteq A$ gilt
3. **Echte Teilmenge**: $A \subsetneq B$ gdw. $A \subseteq B$ aber nicht $A = B$

Wichtige Anmerkung

Bei der Mengengleichheit sind **zwei Richtungen** zu zeigen. Einmal $A \subseteq B$ (für jedes Element aus A zeigen, dass es auch in B ist) und einmal $B \subseteq A$ (für jedes Element aus B zeigen, dass es auch in A ist).

Mengen

Definition (Mengen III)

1. **Vereinigung**: $A \cup B = \{x \mid x \in A \text{ oder } x \in B\}$
   Beispiel: $\{1, 2, 4\} \cup \{1, 2, 3\} = \{1, 2, 4, 1, 2, 3\} = \{1, 1, 2, 2, 3, 4\} = \{1, 2, 3, 4\}$
2. **Schnitt**: $A \cap B = \{x \mid x \in A \text{ und } x \in B\}$
   Beispiel: $\{1, 2, 4\} \cap \{1, 2, 3\} = \{1, 2\}$
3. **Mengendifferenz**: $A \setminus B = \{x \mid x \in A \text{ und } x \notin B\}$
   Beispiel: $\{1, 2, 4\} \setminus \{1, 2, 3\} = \{4\}$
4. **Potenzmenge**: $\mathcal{P}(A) = 2^A = \{B \mid B \subseteq A\}$
   Beispiel: $\mathcal{P}(\{1, 2\}) = \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$
5. **Kartesisches Produkt**: $A \times B = \{(a, b) \mid a \in A, b \in B\}$
   Beispiel:
   $\{1, 2\} \times \{a, b, c\} = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c)\}$

Relationen

Definition (Kartesisches Produkt)

Seien $A_1, A_2, \ldots, A_n$ Mengen. Dann ist

$$A_1 \times A_2 \times \ldots \times A_n := \{(x_1, x_2, \ldots, x_n) \mid x_1 \in A_1, \ldots, x_n \in A_n\}$$

das **kartesische Produkt** dieser Mengen. $(x_1, \ldots, x_n)$ ist ein *n-Tupel*. Eine *Teilmenge* $R \subseteq A_1 \times \ldots \times A_n$ heißt ***n*-stellige Relation**.

Beispiel

Sei $Z = \{1, 2, 3\}$, $A = \{a, b, c\}$, dann ist

$$R = \{(1, a, 1), (1, b, 2), (2, c, 3), (3, c, 3)\}$$

eine 3-stellige Relation über $Z \times A \times Z$.

Funktionen

Definition (Funktion)

Eine **totale Funktion**

$$f : A \to B$$

weist jedem Element aus ihrem Definitionsbereich A ein Element aus ihrem Bildbereich B zu. Z.B.

- $f : \{0, 1\} \to \{0, 1\}$ mit $0 \mapsto 1$ und $1 \mapsto 0$
- $f : \mathbb{N} \to \mathbb{N}$ mit $x \mapsto 2 \cdot x$ (alternativ $f(x) = 2x$)

Funktionen

Anmerkung

$$f : A \to B$$

Bemerkung

1. Gibt es ein $a \in A$ ohne Bild, d.h. ist $f(a)$ nicht definiert, nennt man f eine *partielle* Funktion.
2. Jedes $x \in A$ hat höchstens ein Bild. ($f(x) = 5$ und $f(x) = 4$ ist nicht gleichzeitig möglich.)
3. Zwei $x_1, x_2 \in A$ können aber das gleiche Bild haben, d.h. $f(x_1) = f(x_2)$ ist möglich.

Alphabete und Wörter

Definition

1. Ein **Alphabet** ist eine (total geordnete) endliche Menge von unterschiedlichen **Zeichen** (alternativ: Buchstaben oder Symbole).
2. Die **Konkatenation** $\cdot$ ist die Operation zum Hintereinanderschreiben von Buchstaben. So ergeben sich Worte auf die die Operation dann erweitert wird. Z.B. ist $a \cdot b = ab$ und dann $ab \cdot c = abc$.
3. Das **leere Wort** (Wort mit 0 Buchstaben) wird mit λ oder ϵ bezeichnet (nie in einem Alphabet enthalten!).
4. Wir schreiben $w^1 = w, w^2 = w \cdot w$ und $w^k = w^{k-1} \cdot w$ ($k \geq 2$) und als Spezialfall $w^0 = \lambda$.

Alphabete und Wörter

Die Konkatenation wird auf Mengen von Wörtern erweitert:

$$\begin{aligned}\{ab, aba\} \cdot \{ab, b\} &= \{ab \cdot ab, ab \cdot b, aba \cdot ab, aba \cdot b\} \\ &= \{abab, abb, abaab\} \\ \{a, ab, \lambda\} \cdot \{b, \lambda\} &= \{ab, a, abb, b, \lambda\}\end{aligned}$$

Ausserdem definieren wir für eine Menge von Worten R

$$\begin{aligned}R^1 &= R \\ R^2 &= R \cdot R \\ R^3 &= R^2 \cdot R = R \cdot R \cdot R \\ &\dots \\ R^n &= R^{n-1} \cdot R\end{aligned}$$

und als Spezialfall $R^0 = \{\lambda\}$.

Alphabete und Wörter

Definition

Für eine Menge von Worten R definieren wir

$$\begin{aligned}R^+ &:= \cup_{i \geq 1} R^i \text{ mit } R^1 := R \text{ und } R^{i+1} = R^i \cdot R \\ R^* &:= R^+ \cup \{\lambda\}\end{aligned}$$

Definition

1. Betrachten wir Σ^* für ein Alphabet Σ, so ist Σ^* die **Menge aller endlichen Wörter** (über dem Alphabet Σ).
2. Jede (Teil-)Menge $L \subseteq \Sigma^*$ heißt **formale Sprache**.

Formaler...

Formaler kann man $(\Sigma^*, \cdot, \lambda)$ als ein (freies) Monoid auffassen.

Alphabet und Wörter - Zusammengefasst

Die wichtigsten Dinge:

- Σ für eine Menge von Symbolen (ein Alphabet), z.B. $\Sigma = \{a, b, c\}$ oder $\Sigma = \{0, 1\}$.
- λ oder ϵ für das leere Wort.
- Das leere Wort ist *nie in einem Alphabet* Σ!
- Konkatenation:
  - Von Symbolen oder Worten: $a \cdot b = ab$, $ab \cdot cd = abcd$
  - Von Wortmengen: $\{a, ac\} \cdot \{\lambda, c\} = \{a, ac, acc\}$
- R^2, R^3 oder auch Σ^2, Σ^3 und w^2, w^3 usw. für mehrfach Hinterinanderausführen von $\cdot$.
- Sonderfall: $R^0 = \{\lambda\}$ und auch $w^0 = \lambda$ (w ein Wort)
- R^+ für alle Worte, die sich durch beliebiges Aneinanderreihen von Worten aus R bilden lassen.
- R^* wie R^+, aber λ kommt noch hinzu.

Alphabete und Wörter - Noch zwei Notationen

Noch zwei Notationen:

Definition/Notation

Sei Σ ein Alphabet, $x \in \Sigma$ ein Symbol und $w \in \Sigma^*$ ein Wort.

1. Mit $|w|$ ist die Länge von w gemeint, also z.B. $|001| = 3$ und $|00111| = 5$. Außerdem ist $|\lambda| = 0$.
2. Mit $|w|_x$ ist die Anzahl der x in w gemeint. Z.B. ist mit $\Sigma = \{0, 1, 2\}$

$$|2202|_2 = 3, \quad |2202|_1 = 0 \quad \text{und } |2202|_0 = 1.$$

Fragen...

$\{a, bc\} \cdot \{de, fg\}$

1. $\{abc, defg\}$
2. $\{abcde, abcfg\}$
3. $\{abcde, abcfg, bcade, bcafg\}$
4. $\{ade, afg, bcde, bcfg\}$

Fragen...

$\{abc, a\} \cdot \{bc, \lambda\}$

1. $\{abca, bc\}$
2. $\{abcbc, abc\}$
3. $\{abcbc, abc\lambda, abc, a\lambda\}$
4. $\{abcbc, abc, a\}$

Fragen...

$R = \{a, b\}$ was ist R^2?

1. $\{\lambda, a, b, aa, ab, ba, bb\}$
2. $\{a, b, aa, ab, ba, bb\}$
3. $\{aa, ab, ba, bb\}$
4. $\{\{aa\}, \{ab\}, \{ba\}, \{bb\}\}$

Fragen...

$R = \{ab, ba, a, b\}$ was ist R^*?

1. Menge aller endlichen Worte aus a und b
2. Menge aller *ungeraden* Worte aus a und b
3. Menge aller *geraden* Worte aus a und b
4. Menge aller unendlichen Worte aus a und b

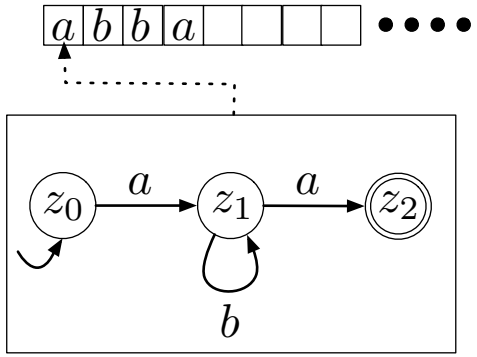
Zur Nachbereitung

Zur Nachbereitung

Richtige Antworten sind:

1. 4
2. 3 oder 4, aber 4 ist schöner
3. 3
4. 1

Endliche Automaten



Wie definieren wir das?

Der deterministische, endliche Automat

Definition (DFA)

Ein **deterministischer, endlicher Automat** (DFA) ist ein 5-Tupel

$$A = (Z, \Sigma, \delta, z_0, Z_{end})$$

mit:

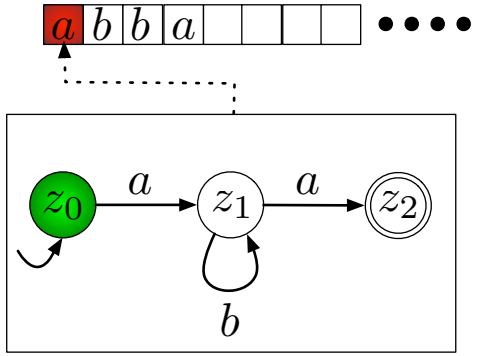
- Der endlichen Menge von *Zuständen* Z.
- Dem endlichen Alphabet Σ von *Eingabesymbolen*.
- Der *Überführungsfunktion* $\delta : Z \times \Sigma \to Z$.
- Dem *Startzustand* $z_0 \in Z$.
- Der Menge der *Endzustände* $Z_{end} \subseteq Z$.

Arbeitsweise des DFA (informal)

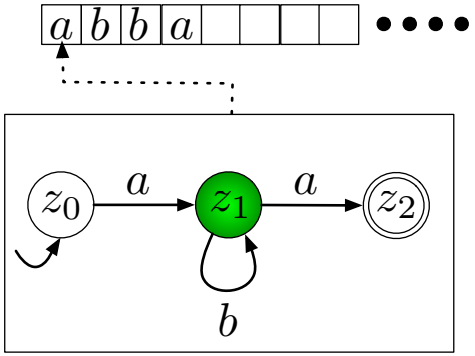
Erhält ein DFA ein Eingabewort $w \in \Sigma^*$, so

- beginnt er im Startzustand z_0.
- Beginnt w mit dem Symbol $x \in \Sigma$, so
- wird der Nachfolgezustand nun durch $\delta(z_0, x)$ bestimmt.
- Dies wird dann in dem nun aktuellen Zustand und mit dem Restwort fortgeführt.
- Das Wort w wird akzeptiert, wenn
  - w bis zum Ende gelesen werden kann **und**
  - der Automat dann in einem Endzustand ist.

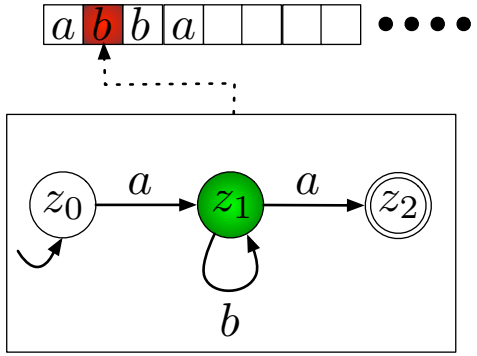
Ein Beispiel



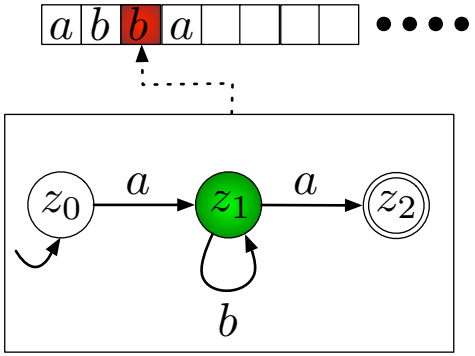
Ein Beispiel



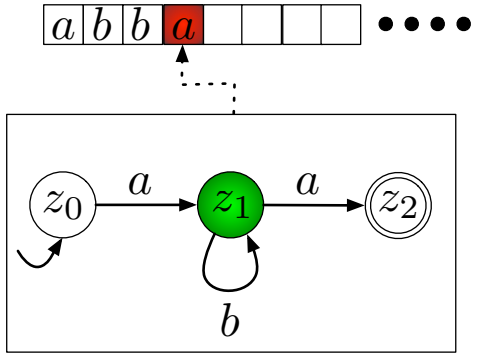
Ein Beispiel



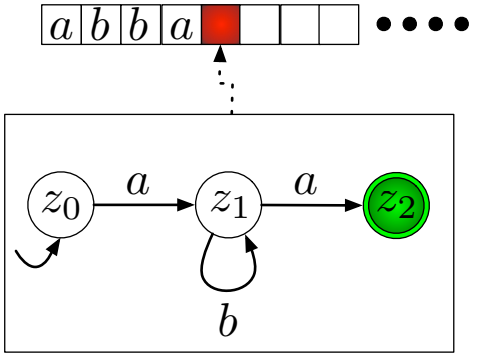
Ein Beispiel



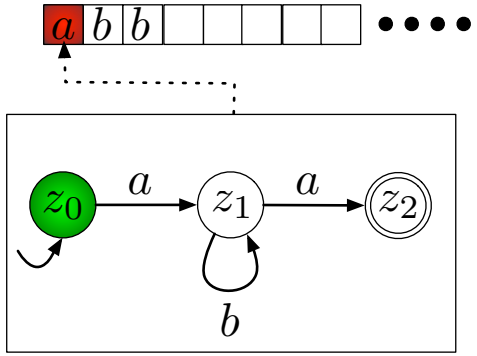
Ein Beispiel



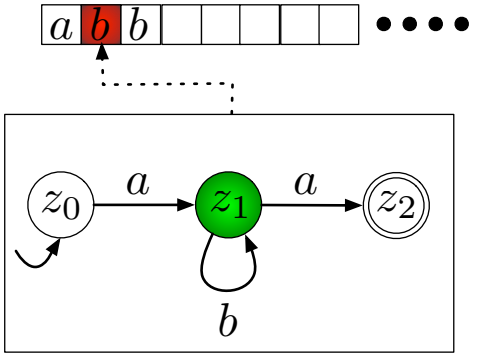
Ein Beispiel



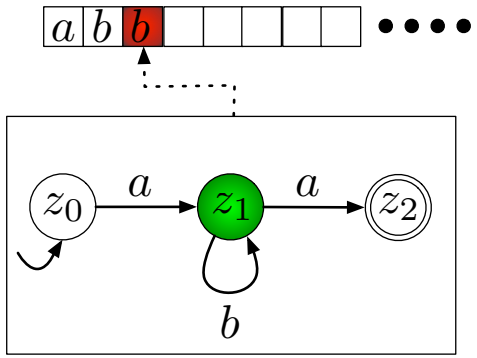
Ein Beispiel



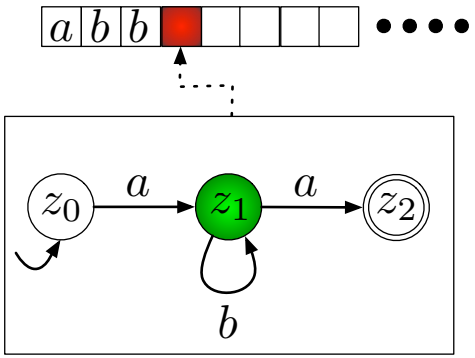
Ein Beispiel



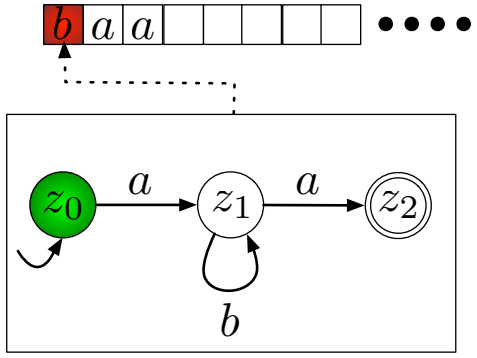
Ein Beispiel



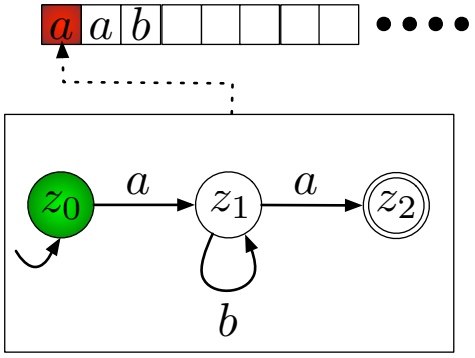
Ein Beispiel



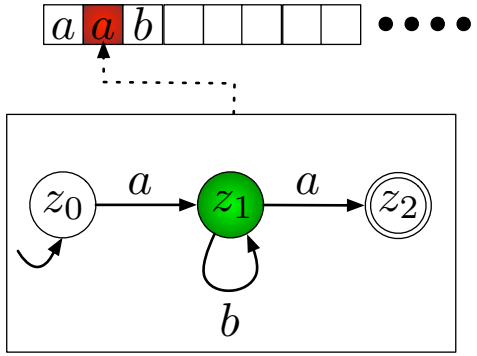
Ein Beispiel



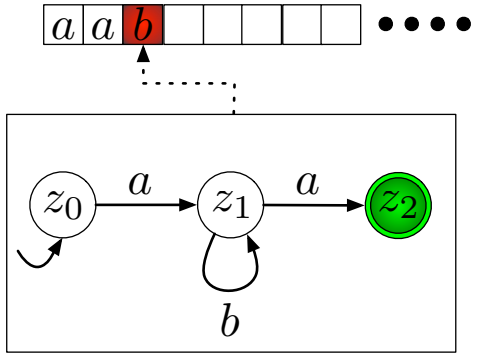
Ein Beispiel



Ein Beispiel



Ein Beispiel



Zusammenfassung

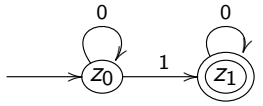
- Das Wort *abba* wird akzeptiert!
  (Zu Ende gelesen **und** im Endzustand!)
- Das Wort *abb* wird *nicht* akzeptiert!
  (Zu Ende gelesen, aber nicht im Endzustand!)
- Das Wort *baa* wird *nicht* akzeptiert!
  (Kann nicht zu Ende gelesen werden!)
- Das Wort *aab* wird *nicht* akzeptiert!
  (Kann nicht zu Ende gelesen werden! Wir sind zwar in einem Endzustand, aber wir können das Wort nicht zu Ende lesen!)

Überführungsfunktion und Rechnung

Definition (Erweiterte Überführungsfunktion)

Sei $A = (Z, \Sigma, \delta, z_0, Z_{end})$ ein DFA. Die **erweiterte Überführungsfunktion** $\hat{\delta} : Z \times \Sigma^* \to Z$ wird für alle $z \in Z$, $x \in \Sigma$ und $w \in \Sigma^*$ rekursiv definiert durch

$$\begin{aligned} \hat{\delta}(z, \lambda) &:= z \\ \hat{\delta}(z, xw) &:= \hat{\delta}(\delta(z, x), w) \end{aligned}$$



$$\hat{\delta}(z_0, 001) = \hat{\delta}(z_0, 01) = \hat{\delta}(z_0, 1) = z_1$$

Überführungsfunktion und Rechnung

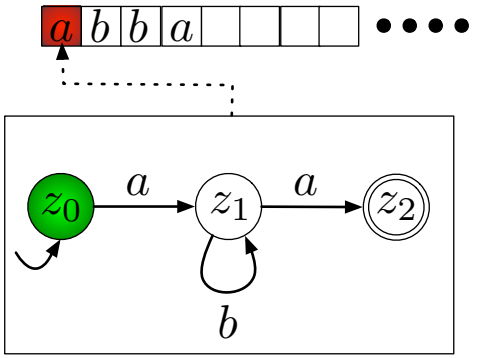
Definition (Konfiguration und Rechnung)

1. Eine **Konfiguration** eines DFA A ist ein Tupel $(z, w) \in Z \times \Sigma^*$ mit der Bedeutung, dass A im Zustand z ist und noch das Wort w zu lesen ist.
2. Ein **Konfigurationsübergang** ist dann

$$(z, w) \vdash (z', v)$$

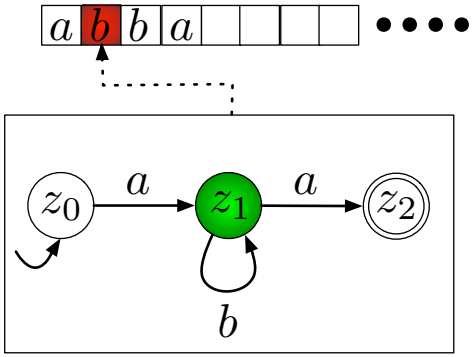
gdw. $w = xv$, $x \in \Sigma$ und $\delta(z, x) = z'$ ist.
3. Eine **Rechnung** auf dem Wort $w \in \Sigma^*$ ist eine Folge von Konfigurationsübergängen, die in (z_0, w) beginnt.
4. Endet die Rechnung in (z', λ) und ist $z' \in Z_{end}$, so ist dies eine **Erfolgsrechnung** und w wird akzeptiert.

Ein Beispiel



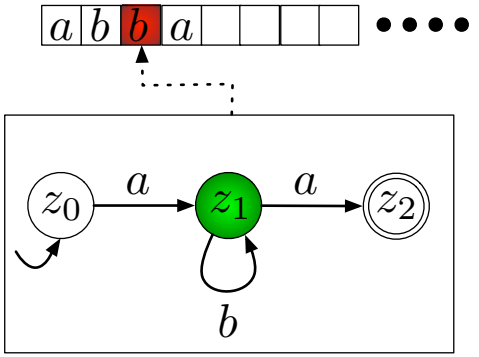
$(z_0, abba)$

Ein Beispiel



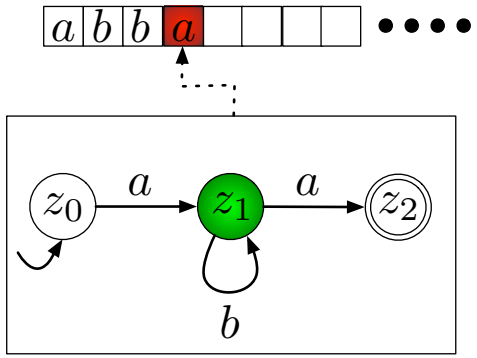
$(z_0, abba) \vdash (z_1, bba)$

Ein Beispiel



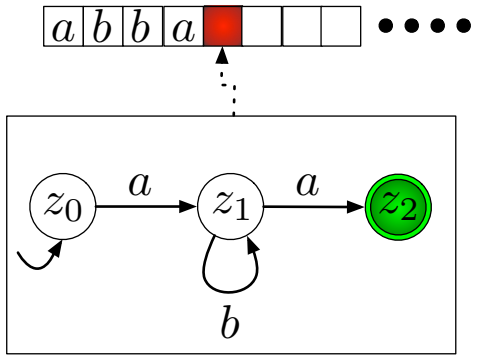
$(z_0, abba) \vdash (z_1, bba) \vdash (z_1, ba)$

Ein Beispiel



$(z_0, abba) \vdash (z_1, bba) \vdash (z_1, ba) \vdash (z_1, a)$

Ein Beispiel



$(z_0, abba) \vdash (z_1, bba) \vdash (z_1, ba) \vdash (z_1, a) \vdash (z_2, \lambda)$

Hinweis

Hinweis

Der Begriff der Rechnung wird oft in der Literatur nicht für DFAs eingeführt (sondern erst später). Es ist aber ganz praktisch, ihn hier zur Übung bereits einzuführen.

Akzeptierte Sprache

Definition (Akzeptierte Sprache)

Die von einem DFA A **akzeptierte Sprache** ist die Menge

$$\begin{aligned} L(A) &:= \{w \in \Sigma^* \mid \hat{\delta}(z_0, w) \in Z_{end}\} \\ &= \{w \in \Sigma^* \mid (z_0, w) \vdash^* (z_e, \lambda), z_e \in Z_{end}\} \end{aligned}$$

Diese Menge wird auch als **reguläre Menge** bezeichnet. Die **Familie aller regulären Mengen** wird mit REG bezeichnet.

Hinweis

Ist A ein DFA, so ist also $L(A)$ eine Menge von Worten (die von A akzeptierte Sprache) und außerdem ist $L(A) \in$ REG (REG enthält also Mengen!).

Noch zwei Begriffe

Definition

Ein DFA $A = (Z, \Sigma, \delta, z_0, Z_{end})$ heißt

1. **vollständig**, wenn zu jedem $(z, x) \in Z \times \Sigma$ ein z' existiert mit $\delta(z, x) = z'$.
2. **initial zusammenhängend**, wenn es zu jedem $z \in Z$ ein Wort w gibt mit $\hat{\delta}(z_0, w) = z$.

Hinweis

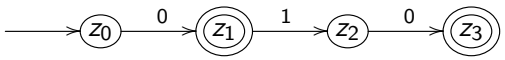
vollständig = jede Eingabe kann gelesen werden
initial zusammenhängend = jeder Zustand erreichbar

Zusammenfassung

Begriffe bisher:

- DFA
- Zustände, Startzustand, Endzustände
- Überführungsfunktion
- erweitere Überführungsfunktion
- vollständig, initial zusammenhängend
- Konfiguration, Konfigurationsübergang
- Rechnung, Erfolgsrechnung
- akzeptierte Sprache
- reguläre Menge, REG

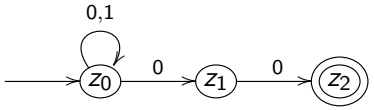
Fragen...



Welche Worte akzeptiert dieser Automat?

1. Nur 0
2. 0 und 010
3. Nur 010
4. 0, 01 und 010
5. Gar keine Worte (zwei Endzustände sind nicht möglich!)

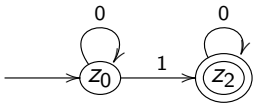
Fragen...



Ist dies ein DFA?

1. Ja!
2. Nein!
3. Weiß ich nicht!
4. Sag ich nicht! ;)

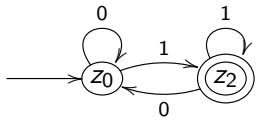
Fragen...



Welche Sprache akzeptiert dieser DFA A?

1. $L(A) = \{1\}$
2. $L(A) = \{0\}^* \cdot \{1\}$
3. $L(A) = \{0\}^* \cdot \{1\} \cdot \{0\}^*$
4. $L(A) = \{0\}^* \cdot \{1\} \cdot \{0,1\}^*$

Fragen...



Welche Sprache akzeptiert dieser DFA A?

1. $L(A) = \{0,1\}^*$
2. $L(A) = \{0\}^* \cdot \{1\} \cdot \{1\}^*$
3. $L(A) = \{0,1\}^* \cdot \{1\}$
4. $L(A) = (\{0\}^* \cdot \{1\} \cdot \{1\}^* \cdot \{0\})^*$

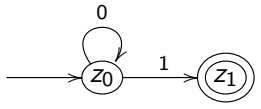
Zur Nachbereitung

Zur Nachbereitung

Richtige Antworten sind:

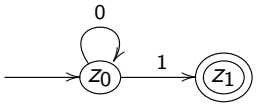
1. 2
2. 2
3. 3
4. 3

Ein Beispiel



Welche Sprache akzeptiert dieser DFA?

Ein Beispiel



Behauptung

$$L(A) = \{0,1\}^* \cdot \{1\} \cdot \{0,1\}^* =: M$$

Wichtige Anmerkung

Dies ist zunächst eine **Behauptung**! Die Richtigkeit ist zu zeigen. Üblicherweise indem man $L(A) \subseteq M$ **und** $M \subseteq L(A)$ zeigt.

Anmerkung

- Wenn ihr auf eine Menge M kommt, von der ihr meint, dass $L(A) = M$ gilt, dann ist das zunächst eine Behauptung!
- Diese ist zu beweisen/zu begründen!
- Hierzu genügt es nicht, nur eine der Teilmengenbeziehungen zu zeigen!
  - $L(A) \subseteq M$. Zeigt nur, dass jedes Wort, das der Automat akzeptiert auch in M ist! Aber in M kann mehr sein! (Würde das genügen, kann man $M = \Sigma^*$ wählen und ist immer gleich fertig!)
  - $M \subseteq L(A)$. Zeigt nur, dass jedes Wort in M vom Automaten akzeptiert wird, aber der Automat kann evtl. mehr! (Insb. vielleicht auch Sachen die er nicht können soll! Und würde dies genügen, kann man stets mit $M = \emptyset$ alles zeigen!
- Also: Sauber die Behauptung aufstellen und sauber die zwei Richtungen trennen und zeigen!

Technik für $L(A) = M$

Wichtiges Vorgehen

Ermittelt man für einen DFA A seine akzeptierte Sprache, so ist $L(A) = M$ zunächst eine Behauptung, die zu zeigen ist!

Wichtiges Vorgehen

Hierzu sind dann **zwei Richtungen** zu zeigen:

- $L(A) \subseteq M$. Jedes Wort, dass der Automat akzeptiert ist tatsächlich in M. Bei der Argumentation geht man von einem Wort w aus, das der Automat akzeptiert ($w \in L(A)$) und zeigt, dass dann auch $w \in M$ gilt.
- $M \subseteq L(A)$. Jedes Wort aus M wird auch von dem Automaten akzeptiert. Man geht von einem (beliebigen!) Wort aus M aus ("Sei $w \in M$, dann ...") und zeigt, dass dieses auch von A akzeptiert wird, also in $L(A)$ ist.

Zusammenfassung - Begriffe

Begriffe bisher (mathematische Grundlagen):

- Mengen, (echte) Teilmenge, Mengengleichheit
- Vereinigung, Schnitt, Komplement, Potenzmenge
- kartesisches Produkt, Relation
- Funktion

Begriffe bisher (Buchstaben, Worte):

- Alphabet, Konkatenation, Wort, leeres Wort, $|w|$, $|w|_x$
- R^+, R^*, Σ^+, Σ^*, R^0, R^1, R^i, w^0, w^1, w^i

Zusammenfassung - Begriffe

Begriffe bisher (endliche Automaten):

- DFA
- Zustände, Startzustand, Endzustände
- Überführungsfunktion
- erweitere Überführungsfunktion
- vollständig, initial zusammenhängend
- Konfiguration, Konfigurationsübergang
- Rechnung, Erfolgsrechnung
- akzeptierte Sprache
- reguläre Menge, REG

Wichtige Anmerkung

Alle diese Begriffe sind wichtig (auf dieser Grundlage bauen wir auf). Daher die fleissig lernen!