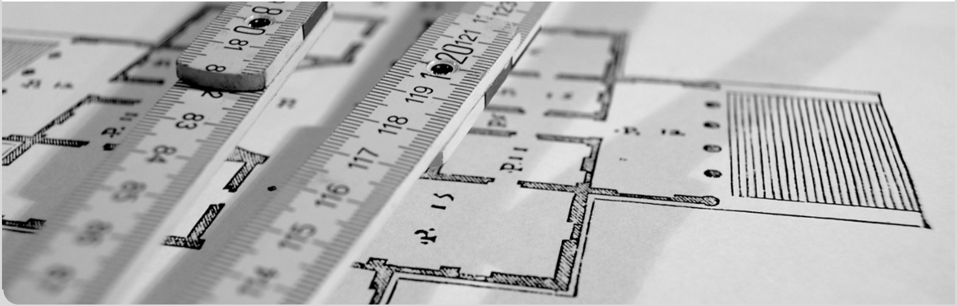
Karlsruher Institut für Technologie

Grundbegriffe der Informatik
Reguläre Ausdrücke und Turing Maschinen

Maria Schmid | 08.02.2016

KIT - KARLSRUHER INSTITUT FÜR TECHNOLOGIE



Inhaltsverzeichnis

Informationen zum letzten Blatt

Karlsruher Institut für Technologie

Blatt Nr.12

Abgabetermin	05.02.2016 um 12:30 Uhr
Abgabeort	Briefkasten
Themen	Automaten
Maximale Punkte	18
Gesamtpunkte	212

Motivation

Warum reguläre Ausdrücke?

- Schnelle Suche / Vergleich usw.

Motivation

Warum reguläre Ausdrücke?

- Schnelle Suche / Vergleich usw.
- Ausdruck einer Sprache

Motivation

Warum reguläre Ausdrücke?

- Schnelle Suche / Vergleich usw.
- Ausdruck einer Sprache
- Compilerbau

Motivation

Warum reguläre Ausdrücke?

- Schnelle Suche / Vergleich usw.
- Ausdruck einer Sprache
- Compilerbau
- uvm.

Definition

Definition

Die Grammatik

$$G = (\{R\}, \{|, (,), *, \emptyset\} \cup A, R, P)$$

mit

$$P = \{R \to \emptyset, R \to x\ (x \in A), R \to (R \mid R), R \to (RR), R \to (R*)\}$$

erzeugt die so genannten **regulären Ausdrücke** über dem Alphabet A.

Reguläre Ausdrücke

Wir können uns also reguläre Ausdrücke zusammenbauen aus

- den einzelnen Symbolen x aus A

Reguläre Ausdrücke

Wir können uns also reguläre Ausdrücke zusammenbauen aus

- den einzelnen Symbolen x aus A
- zwei regulären Ausdrücken R_1 und R_2 mit

$$(R_1 R_2) \qquad \text{oder} \qquad (R_1 \mid R_2)$$

Reguläre Ausdrücke

Wir können uns also reguläre Ausdrücke zusammenbauen aus

- den einzelnen Symbolen x aus A
- zwei regulären Ausdrücken R_1 und R_2 mit

$$(R_1 R_2) \qquad \text{oder} \qquad (R_1 \mid R_2)$$

- einem Stern $R*$

Reguläre Ausdrücke

Wir können uns also reguläre Ausdrücke zusammenbauen aus

- den einzelnen Symbolen x aus A
- zwei regulären Ausdrücken R_1 und R_2 mit

$$(R_1 R_2) \qquad \text{oder} \qquad (R_1 \mid R_2)$$

- einem Stern $R*$
- oder dem leeren Ausdruck

Reguläre Ausdrücke

Wir können uns also reguläre Ausdrücke zusammenbauen aus

- den einzelnen Symbolen x aus A
- zwei regulären Ausdrücken R_1 und R_2 mit

$$(R_1 R_2) \qquad \text{oder} \qquad (R_1 \mid R_2)$$

- einem Stern $R*$
- oder dem leeren Ausdruck

Reguläre Ausdrücke

Wir können uns also reguläre Ausdrücke zusammenbauen aus

- den einzelnen Symbolen x aus A
- zwei regulären Ausdrücken R_1 und R_2 mit

$$(R_1 R_2) \qquad \text{oder} \qquad (R_1 \mid R_2)$$

- einem Stern $R*$
- oder dem leeren Ausdruck

Klammern dürfen nach den Klammerregeln weggelassen werden!

Sprache eines Ausdruckes

Satz

Sei R ein regulärer Ausdruck. Dann gibt es eine Sprache $L = \langle R \rangle$ von R die alle Wörter enthält, die der reguläre Ausdruck beschreibt.

Beispiele

- $R = a$

Beispiele

Karlsruher Institut für Technologie

- $R = a$ und $\langle R \rangle = \{a\}$
- $R = (a \mid b)$

Beispiele

- $R = a$ und $\langle R \rangle = \{a\}$
- $R = (a \mid b)$ und $\langle R \rangle = \{a, b\}$
- $R = (ab)$

Beispiele

- $R = a$ und $\langle R \rangle = \{a\}$
- $R = (a \mid b)$ und $\langle R \rangle = \{a, b\}$
- $R = (ab)$ und $\langle R \rangle = \{ab\}$
- $R = (a \mid b)*$

Beispiele

- $R = a$ und $\langle R \rangle = \{a\}$
- $R = (a \mid b)$ und $\langle R \rangle = \{a, b\}$
- $R = (ab)$ und $\langle R \rangle = \{ab\}$
- $R = (a \mid b)*$ und $\langle R \rangle = \{a, b\}^*$
- $R = (a * b*)*$

Beispiele

- $R = a$ und $\langle R \rangle = \{a\}$
- $R = (a \mid b)$ und $\langle R \rangle = \{a, b\}$
- $R = (ab)$ und $\langle R \rangle = \{ab\}$
- $R = (a \mid b)*$ und $\langle R \rangle = \{a, b\}^*$
- $R = (a * b*)*$ und $\langle R \rangle = \{a, b\}^*$
- $R = (a \mid b) * abb(a \mid b)*$

Beispiele

- $R = a$ und $\langle R \rangle = \{a\}$
- $R = (a \mid b)$ und $\langle R \rangle = \{a, b\}$
- $R = (ab)$ und $\langle R \rangle = \{ab\}$
- $R = (a \mid b)*$ und $\langle R \rangle = \{a, b\}^*$
- $R = (a * b*)*$ und $\langle R \rangle = \{a, b\}^*$
- $R = (a \mid b) * abb(a \mid b)*$ und $\langle R \rangle = \{w_1 abb w_2 \mid w_1, w_2 \in \{a, b\}^*\}$

Ausdruck einer Sprache

Definition

Eine Grammatik G nennt man **rechtslinear** wenn bei jeder Ersetzung der Produktion auf der rechten Seite höchstens ein Nichtterminalsymbol und dieses nur als letztes Symbol steht.

Satz

Sei $L(G)$ eine Sprache einer rechtslinearen Grammatik (also eine Typ-3-Sprache). Dann existiert ein regulärer Ausdruck R mit $\langle R \rangle = L(G)$.

DER Satz

Satz

Für jede formale Sprache L sind die folgenden drei Aussagen äquivalent:

- *L kann von einem endlichen Akzeptor erkannt werden.*
- *L kann durch einen regulären Ausdruck beschrieben werden.*
- *L kann von einer rechtslinearen Grammatik erzeugt werden.*

DER Satz

Satz

Für jede formale Sprache L sind die folgenden drei Aussagen äquivalent:

- *L kann von einem endlichen Akzeptor erkannt werden.*
- *L kann durch einen regulären Ausdruck beschrieben werden.*
- *L kann von einer rechtslinearen Grammatik erzeugt werden.*

Wir können also zu regulären Ausdrücken Grammatiken definieren, Automaten zeichnen und auch umgekehrt.

Beispiele

$G = (\{X, Y, Z\}, \{a, b\}, X, P)$ mit

$$P = \{X \to aX \mid bY \mid \varepsilon, Y \to aX \mid bZ \mid \varepsilon, Z \to aZ \mid bZ\}$$

ist eine rechtslineare Grammatik. Die Sprache ist

Beispiele

Karlsruher Institut für Technologie

$G = (\{X, Y, Z\}, \{a, b\}, X, P)$ mit

$$P = \{X \to aX \mid bY \mid \varepsilon, Y \to aX \mid bZ \mid \varepsilon, Z \to aZ \mid bZ\}$$

ist eine rechtslineare Grammatik. Die Sprache ist

$$L(G) = \{w \mid \forall v_1, v_2 \in \{a, b\}^* : w \neq v_1 bb v_2\}$$

Der reguläre Ausdruck ist

Beispiele

$G = (\{X, Y, Z\}, \{a, b\}, X, P)$ mit

$$P = \{X \to aX \mid bY \mid \varepsilon, Y \to aX \mid bZ \mid \varepsilon, Z \to aZ \mid bZ\}$$

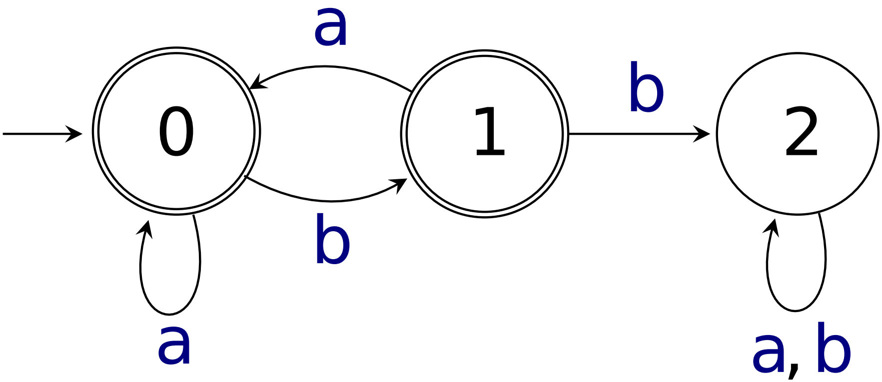
ist eine rechtslineare Grammatik. Die Sprache ist

$$L(G) = \{w \mid \forall v_1, v_2 \in \{a, b\}^* : w \neq v_1 bbv_2\}$$

Der reguläre Ausdruck ist

$$R = (a \mid ba) * (b \mid \emptyset)$$

der Automat ist


a
b
0
1
b
2
a
a, b

Jetzt sieht man vielleicht auch

$$G = (\{X\}, \{a, b\}, X, P)$$

mit

$$P = \{X \to aX \mid baX \mid b \mid \varepsilon\}$$

Noch mehr Beispiele

- $G = (\{X\}, \{a, b\}, X, \{X \to abX \mid bbaX \mid \varepsilon\}$ wird beschrieben durch

Noch mehr Beispiele

- $G = (\{X\}, \{a, b\}, X, \{X \to abX \mid bbaX \mid \varepsilon\}$ wird beschrieben durch $L(G) = \langle (ab \mid bba)* \rangle$
- $G = (\{X, Y\}, \{a, b\}, X, \{X \to aX \mid bX \mid ababbY, Y \to aY \mid bY \mid \varepsilon\}$

Noch mehr Beispiele

- $G = (\{X\}, \{a, b\}, X, \{X \to abX \mid bbaX \mid \varepsilon\}$ wird beschrieben durch $L(G) = \langle (ab \mid bba)* \rangle$
- $G = (\{X, Y\}, \{a, b\}, X, \{X \to aX \mid bX \mid ababbY, Y \to aY \mid bY \mid \varepsilon\}$ wird beschrieben durch $L(G) = \langle (a \mid b) * ababb(a \mid b)* \rangle$

Aufgabe

Gegeben ist im folgenden jeweils eine Beschreibung einer formalen Sprache L und ein dazugehöriges Alphabet. Schreiben Sie jeweils den regulären Ausdruck R auf, für den $L(R) = L$ gilt:

- Die Menge aller Worte über dem Alphabet $A = \{a, b, c\}$, die genau ein c enthalten.
- Die Menge aller Worte über dem Alphabet $A = \{a, b\}$, bei denen die Anzahl der b durch 3 teilbar ist.

Aufgabe

Gegeben ist im folgenden jeweils eine Beschreibung einer formalen Sprache L und ein dazugehöriges Alphabet. Schreiben Sie jeweils den regulären Ausdruck R auf, für den $L(R) = L$ gilt:

- Die Menge aller Worte über dem Alphabet $A = \{a, b, c\}$, die genau ein c enthalten.
- Die Menge aller Worte über dem Alphabet $A = \{a, b\}$, bei denen die Anzahl der b durch 3 teilbar ist.

Aufgabe

Gegeben ist im folgenden jeweils eine Beschreibung einer formalen Sprache L und ein dazugehöriges Alphabet. Schreiben Sie jeweils den regulären Ausdruck R auf, für den $L(R) = L$ gilt:

- Die Menge aller Worte über dem Alphabet $A = \{a, b, c\}$, die genau ein c enthalten.
  Lösung: $(a|b) * c(a|b)*$
- Die Menge aller Worte über dem Alphabet $A = \{a, b\}$, bei denen die Anzahl der b durch 3 teilbar ist.

Aufgabe

Gegeben ist im folgenden jeweils eine Beschreibung einer formalen Sprache L und ein dazugehöriges Alphabet. Schreiben Sie jeweils den regulären Ausdruck R auf, für den $L(R) = L$ gilt:

- Die Menge aller Worte über dem Alphabet $A = \{a, b, c\}$, die genau ein c enthalten.
  Lösung: $(a|b) * c(a|b)*$
- Die Menge aller Worte über dem Alphabet $A = \{a, b\}$, bei denen die Anzahl der b durch 3 teilbar ist.
  Lösung: $a * (ba * ba * ba*)*$

Aufgabe

Gegeben sei die rechtslineare Grammatik

$$G = (\{S\}, \{a, b\}, S, P) \qquad P = \{S \to baaS|baS|aaS|\varepsilon\}$$

(a) Geben Sie einen endlichen Akzeptor A an, so dass $L(A) = L(G)$ gilt

(b) Geben Sie einen regulären Ausdruck R an, so dass $\langle R \rangle = L(G)$ gilt

(c) Geben Sie einen regulären Ausdruck R an, der nicht das Zeichen $|$ enthält, und für den $\langle R \rangle = L(G)$ gilt.

Aufgabe

Gegeben sei die rechtslineare Grammatik

$$G = (\{S\}, \{a, b\}, S, P) \qquad P = \{S \rightarrow baaS|baS|aaS|\varepsilon\}$$

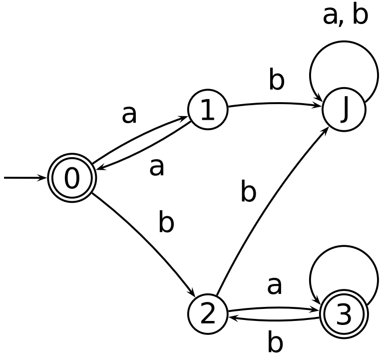
(a) Geben Sie einen endlichen Akzeptor A an, so dass $L(A) = L(G)$ gilt

Aufgabe

Gegeben sei die rechtslineare Grammatik

$$G = (\{S\}, \{a, b\}, S, P) \qquad P = \{S \rightarrow baaS | baS | aaS | \varepsilon\}$$

(a) Geben Sie einen endlichen Akzeptor A an, so dass $L(A) = L(G)$ gilt



Aufgabe

Gegeben sei die rechtslineare Grammatik

$$G = (\{S\}, \{a, b\}, S, P) \qquad P = \{S \to baaS|baS|aaS|\varepsilon\}$$

(a) Geben Sie einen regulären Ausdruck R an, so dass $\langle R \rangle = L(G)$ gilt

Aufgabe

Gegeben sei die rechtslineare Grammatik

$$G = (\{S\}, \{a, b\}, S, P) \qquad P = \{S \to baaS | baS | aaS | \varepsilon\}$$

(a) Geben Sie einen regulären Ausdruck R an, so dass $\langle R \rangle = L(G)$ gilt

$$(baa|ba|aa)*$$

Aufgabe

Gegeben sei die rechtslineare Grammatik

$$G = (\{S\}, \{a, b\}, S, P) \qquad P = \{S \to baaS|baS|aaS|\varepsilon\}$$

(a) Geben Sie einen regulären Ausdruck R an, der nicht das Zeichen $|$ enthält, und für den $\langle R \rangle = L(G)$ gilt.

Aufgabe

Gegeben sei die rechtslineare Grammatik

$$G = (\{S\}, \{a, b\}, S, P) \qquad P = \{S \rightarrow baaS|baS|aaS|\varepsilon\}$$

(a) Geben Sie einen regulären Ausdruck R an, der nicht das Zeichen $|$ enthält, und für den $\langle R \rangle = L(G)$ gilt.

$$(aa) * (baa*)*$$

In dieser Aufgabe geht um die formalen Sprachen

$$L_1 = \{a^k b^m \mid k, m \in \mathbb{N}_0\} \qquad L_2 = \{b^k a^m \mid k, m \in \mathbb{N}_0\}$$

Geben Sie für jede der folgenden formalen Sprachen L je einen regulären Ausdruck R an mit $\langle R \rangle = L$.

- $L = L_1 \cup L_2$
- $L = L_1 \cap L_2$
- $L = L_1 \cdot L_2$
- $L = L_1*$

In dieser Aufgabe geht um die formalen Sprachen

$$L_1 = \{a^k b^m \mid k, m \in \mathbb{N}_0\} \qquad L_2 = \{b^k a^m \mid k, m \in \mathbb{N}_0\}$$

Geben Sie für jede der folgenden formalen Sprachen L je einen regulären Ausdruck R an mit $\langle R \rangle = L$.

- $L = L_1 \cup L_2$ Lösungsvorschlag: $a * b* \mid b * a*$
- $L = L_1 \cap L_2$
- $L = L_1 \cdot L_2$
- $L = L_1*$

In dieser Aufgabe geht um die formalen Sprachen

$$L_1 = \{a^k b^m \mid k, m \in \mathbb{N}_0\} \qquad L_2 = \{b^k a^m \mid k, m \in \mathbb{N}_0\}$$

Geben Sie für jede der folgenden formalen Sprachen L je einen regulären Ausdruck R an mit $\langle R \rangle = L$.

- $L = L_1 \cup L_2$ Lösungsvorschlag: $a * b* \mid b * a*$
- $L = L_1 \cap L_2$ Lösungsvorschlag: $a* \mid b*$
- $L = L_1 \cdot L_2$
- $L = L_1*$

In dieser Aufgabe geht um die formalen Sprachen

$$L_1 = \{a^k b^m \mid k, m \in \mathbb{N}_0\} \qquad L_2 = \{b^k a^m \mid k, m \in \mathbb{N}_0\}$$

Geben Sie für jede der folgenden formalen Sprachen L je einen regulären Ausdruck R an mit $\langle R \rangle = L$.

- $L = L_1 \cup L_2$ Lösungsvorschlag: $a * b * \mid b * a *$
- $L = L_1 \cap L_2$ Lösungsvorschlag: $a * \mid b *$
- $L = L_1 \cdot L_2$ Lösungsvorschlag: $a * b * b * a *$ oder $a * b * a *$
- $L = L_1 *$

In dieser Aufgabe geht um die formalen Sprachen

$$L_1 = \{a^k b^m \mid k, m \in \mathbb{N}_0\} \qquad L_2 = \{b^k a^m \mid k, m \in \mathbb{N}_0\}$$

Geben Sie für jede der folgenden formalen Sprachen L je einen regulären Ausdruck R an mit $\langle R \rangle = L$.

- $L = L_1 \cup L_2$ Lösungsvorschlag: $a * b* \mid b * a*$
- $L = L_1 \cap L_2$ Lösungsvorschlag: $a* \mid b*$
- $L = L_1 \cdot L_2$ Lösungsvorschlag: $a * b * b * a*$ oder $a * b * a*$
- $L = L_1*$ Lösungsvorschlag: $(a * b*)*$ oder $(a \mid b)*$

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren:

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, | ist Verzweigung, Akzeptierende Zustände wählen

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, | ist Verzweigung, Akzeptierende Zustände wählen
- den regulären Ausdrücken:

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, | ist Verzweigung, Akzeptierende Zustände wählen
- den regulären Ausdrücken: Schwierig!

..endlichen Akzeptoren.. zu

- den Grammatiken:

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, | ist Verzweigung, Akzeptierende Zustände wählen
- den regulären Ausdrücken: Schwierig!

..endlichen Akzeptoren.. zu

- den Grammatiken: Zustandsübergang ist eine Produktion

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, | ist Verzweigung, Akzeptierende Zustände wählen
- den regulären Ausdrücken: Schwierig!

..endlichen Akzeptoren.. zu

- den Grammatiken: Zustandsübergang ist eine Produktion
- den regulären Ausdrücken:

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, | ist Verzweigung, Akzeptierende Zustände wählen
- den regulären Ausdrücken: Schwierig!

..endlichen Akzeptoren.. zu

- den Grammatiken: Zustandsübergang ist eine Produktion
- den regulären Ausdrücken: Einzelne Wege abgehen

..regulären Ausdrücken.. zu

- den Akzeptoren:

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, $|$ ist Verzweigung, Akzeptierende Zustände wählen
- den regulären Ausdrücken: Schwierig!

..endlichen Akzeptoren.. zu

- den Grammatiken: Zustandsübergang ist eine Produktion
- den regulären Ausdrücken: Einzelne Wege abgehen

..regulären Ausdrücken.. zu

- den Akzeptoren: in Abschnitte teilen, $*$ ist Schleife, $|$ ist Verzweigung

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, $|$ ist Verzweigung, Akzeptierende Zustände wählen
- den regulären Ausdrücken: Schwierig!

..endlichen Akzeptoren.. zu

- den Grammatiken: Zustandsübergang ist eine Produktion
- den regulären Ausdrücken: Einzelne Wege abgehen

..regulären Ausdrücken.. zu

- den Akzeptoren: in Abschnitte teilen, $*$ ist Schleife, $|$ ist Verzweigung
- den rechtslinearen Grammatiken:

Was wir können:

Von..

..rechtslinearen Gammatiken.. zu

- den Akzeptoren: (mind.) jedes Nichtterminalsymbol ein Zustand, | ist Verzweigung, Akzeptierende Zustände wählen
- den regulären Ausdrücken: Schwierig!

..endlichen Akzeptoren.. zu

- den Grammatiken: Zustandsübergang ist eine Produktion
- den regulären Ausdrücken: Einzelne Wege abgehen

..regulären Ausdrücken.. zu

- den Akzeptoren: in Abschnitte teilen, $*$ ist Schleife, | ist Verzweigung
- den rechtslinearen Grammatiken: genauso wie Akzeptor

Definition

Eine Turingmaschine T ist definiert als

$$T = (Z, z_0, X, f, g, m)$$

wobei

- Z die Zustandsmenge
- $z_0 \in Z$ den Startzustand
- X das Bandalphabet
- $f : Z \times X \dashrightarrow Z$ eine partielle Übergangsfunktion
- $g : Z \to X \dashrightarrow X$ eine partielle Ausgabefunktion
- $m : Z \to X \dashrightarrow \{-1, 0, 1\}$ eine partielle Bewegungsfunktion (alternativ $m : Z \to X \dashrightarrow \{\mathrm{L}, \mathrm{N}, \mathrm{R}\}$)

bezeichnet.

Wichtig: Es gibt weiterhin noch das Blanket $\Box \in X$, welches das leere Wort auf dem Band darstellen soll.

Definition

Eine Turingmaschine T ist definiert als

$$T = (Z, z_0, X, f, g, m)$$

wobei

- Z die Zustandsmenge
- $z_0 \in Z$ den Startzustand
- X das Bandalphabet
- $f : Z \times X \dashrightarrow Z$ eine partielle Übergangsfunktion
- $g : Z \to X \dashrightarrow X$ eine partielle Ausgabefunktion
- $m : Z \to X \dashrightarrow \{-1, 0, 1\}$ eine partielle Bewegungsfunktion (alternativ $m : Z \to X \dashrightarrow \{\mathrm{L}, \mathrm{N}, \mathrm{R}\}$)

bezeichnet.

Wichtig: Es gibt weiterhin noch das Blanket $\Box \in X$, welches das leere Wort auf dem Band darstellen soll.

partielle Funktionen

Eine partielle Funktion, ist eine rechtseindeutige Relation, die nicht notwendigerweise linkstotal ist. Es gibt also Werte, für die die Funktion nicht definiert ist.
Beispiel: $f : \mathbb{R} \to \mathbb{R}, x \mapsto \frac{1}{x}$ ist eine partielle Funktion, da sie für $x = 0$ nicht definiert ist.

partielle Funktionen

Eine partielle Funktion, ist eine rechtseindeutige Relation, die nicht notwendigerweise linkstotal ist. Es gibt also Werte, für die die Funktion nicht definiert ist.
Beispiel: $f : \mathbb{R} \to \mathbb{R}, x \mapsto \frac{1}{x}$ ist eine partielle Funktion, da sie für $x = 0$ nicht definiert ist.

Übergangs- und Ausgabefunktion

Genauso wie bei Mealy-Automaten, nur partiell.

Bewegungsfunktion und Funktionsweise der TM

- Eingabewort steht auf einem Band

Bewegungsfunktion und Funktionsweise der TM

- Eingabewort steht auf einem Band
- TM liest das Symbol, auf dem der Kopf steht

Bewegungsfunktion und Funktionsweise der TM

- Eingabewort steht auf einem Band
- TM liest das Symbol, auf dem der Kopf steht
- Über f und g definierter Mealy-Automat entscheidet über die Ausgabe

Bewegungsfunktion und Funktionsweise der TM

- Eingabewort steht auf einem Band
- TM liest das Symbol, auf dem der Kopf steht
- Über f und g definierter Mealy-Automat entscheidet über die Ausgabe
- Ausgabe wird anstelle der gelesenen Eingabe auf das Band geschrieben

Bewegungsfunktion und Funktionsweise der TM

- Eingabewort steht auf einem Band
- TM liest das Symbol, auf dem der Kopf steht
- Über f und g definierter Mealy-Automat entscheidet über die Ausgabe
- Ausgabe wird anstelle der gelesenen Eingabe auf das Band geschrieben
- Bewegungsfunktion entscheidet darüber, ob der Kopf sich um eins nach links oder rechts bewegt oder stehen bleibt.

Bewegungsfunktion und Funktionsweise der TM

Karlsruher Institut für Technologie

- Eingabewort steht auf einem Band
- TM liest das Symbol, auf dem der Kopf steht
- Über f und g definierter Mealy-Automat entscheidet über die Ausgabe
- Ausgabe wird anstelle der gelesenen Eingabe auf das Band geschrieben
- Bewegungsfunktion entscheidet darüber, ob der Kopf sich um eins nach links oder rechts bewegt oder stehen bleibt.
- Prozess beginnt von vorne

Konfiguration

Definition

Als **Konfiguration** $c = (z, b, p)$ beschreiben wir die Konfiguration einer Turing-Maschine zu einem beliebigen Zeitpunkt. Dabei ist

- $z \in Z$ der aktuelle Zustand
- $b \in X^*$ die aktuelle Bandbeschriftung
- $p \in \mathbb{Z}$ die aktuelle Position des Zeigers.

Definition

Geht eine TM in Konfiguration c durch ein Bandsymbol in eine andere Konfiguration c' über, so bezeichnen wir

$$c' = \Delta_1(c)$$

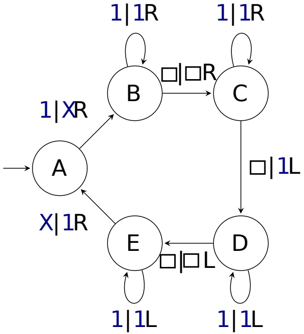
Eine Konfiguration c_E ist eine *Endkonfiguration*, wenn es kein Bandsymbol gibt, so dass $\Delta_1(c_E)$ existiert.
Analog bezeichnet $\Delta_t(c), t \in \mathbb{N}_0$ die Konfiguration, die nach t Schritten erreicht werden kann.

Wie kann ich mir nun also eine Turingmaschine vorstellen?

Wie kann ich mir nun also eine Turingmaschine vorstellen?
Ein endlicher Automat, der durch ein Eingabeband und einem beweglichem Kopf die Möglichkeit hat, sich Dinge zu merken und zu zählen.

Beispiel

Eingabe $w = 11$



	A	B	C	D	E
□		C,□,R	D, 1, L	E,□, L	
1	B,X, R	B,1, R	C,1, R	D,1, L	E,1, L
X					A,1,R

1

$$\begin{array}{ccccccc} & \overset{A}{\downarrow} & & & & & \\ \square & 1 & 1 & \square & \square & \square & \square \end{array}$$

1

$$\begin{array}{ccccccc} & A & & & & & \\ & \downarrow & & & & & \\ \square & 1 & 1 & \square & \square & \square & \square \end{array}$$

2

$$\begin{array}{ccccccc} & & B & & & & \\ & & \downarrow & & & & \\ \square & X & 1 & \square & \square & \square & \square \end{array}$$

1

$$\begin{array}{ccccccc} & A & & & & & \\ & \downarrow & & & & & \\ \square & 1 & 1 & \square & \square & \square & \square \end{array}$$

2

$$\begin{array}{ccccccc} & & B & & & & \\ & & \downarrow & & & & \\ \square & X & 1 & \square & \square & \square & \square \end{array}$$

3

$$\begin{array}{ccccccc} & & & & C & & \\ & & & & \downarrow & & \\ \square & X & 1 & \square & \square & \square & \square \end{array}$$

			B					
			↓					
2	□	X	1	□	□	□	□
					C			
					↓			
3	□	X	1	□	□	□	□
				D				
				↓				
4	□	X	1	□	1	□	□

3
$$\begin{array}{ccccccc} & & & & C & & \\ & & & & \downarrow & & \\ \square & X & 1 & \square & \square & \square & \square \end{array}$$

4
$$\begin{array}{ccccccc} & & & D & & & \\ & & & \downarrow & & & \\ \square & X & 1 & \square & 1 & \square & \square \end{array}$$

5
$$\begin{array}{ccccccc} & & E & & & & \\ & & \downarrow & & & & \\ \square & X & 1 & \square & 1 & \square & \square \end{array}$$

4

D
↓
□ X 1 □ 1 □ □

5

E
↓
□ X 1 □ 1 □ □

6

A
↓
□ 1 1 □ 1 □ □

			E				
			↓				
5	□	X	1	□	1	□	□
			A				
			↓				
6	□	1	1	□	1	□	□
				B			
				↓			
7	□	1	X	□	1	□	□

6

$$\begin{array}{ccccccc} & & \text{A} & & & & \\ & & \downarrow & & & & \\ \square & 1 & 1 & \square & 1 & \square & \square \end{array}$$

7

$$\begin{array}{ccccccc} & & & \text{B} & & & \\ & & & \downarrow & & & \\ \square & 1 & \text{X} & \square & 1 & \square & \square \end{array}$$

8

$$\begin{array}{ccccccc} & & & & & \text{C} & \\ & & & & & \downarrow & \\ \square & 1 & \text{X} & \square & 1 & \square & \square \end{array}$$

				B			
				$\downarrow$			
7	$\square$	1	X	$\square$	1	$\square$	$\square$
						C	
						$\downarrow$	
8	$\square$	1	X	$\square$	1	$\square$	$\square$
					D		
					$\downarrow$		
9	$\square$	1	X	$\square$	1	1	$\square$

8

C
$\downarrow$
□ 1 X □ 1 □ □

9

D
$\downarrow$
□ 1 X □ 1 1 □

10

E
$\downarrow$
□ 1 X □ 1 1 □

9

$$\begin{array}{ccccccc} & & & & \text{D} & & \\ & & & & \downarrow & & \\ \square & 1 & \text{X} & \square & 1 & 1 & \square \end{array}$$

10

$$\begin{array}{ccccccc} & & \text{E} & & & & \\ & & \downarrow & & & & \\ \square & 1 & \text{X} & \square & 1 & 1 & \square \end{array}$$

11

$$\begin{array}{ccccccc} & & & \text{A} & & & \\ & & & \downarrow & & & \\ \square & 1 & 1 & \square & 1 & 1 & \square \end{array}$$

10 $\begin{array}{ccccccc} & & E & & & & \\ & & \downarrow & & & & \\ \square & 1 & X & \square & 1 & 1 & \square \end{array}$

11 $\begin{array}{ccccccc} & & & A & & & \\ & & & \downarrow & & & \\ \square & 1 & 1 & \square & 1 & 1 & \square \end{array}$

Also allgemein : Eingabe von 1^k wird zu $\square 1^k \square 1^k \square$

TM als Akzeptor

Ist eine Turingmasche ein Akzeptor, so ist ein Eingabewort akzeptiert, wenn der Endzustand ein akzeptierter Zustand ist.

Definition

Eine Sprache L ist eine

- *aufzählbare Sprache*, wenn es eine Turingmaschine gibt, die L akzeptiert.
- *entscheidbare Sprache*, wenn es eine Turingmasche gibt, die L akzeptiert und *immer* hält.

TM als Akzeptor

Ist eine Turingmasche ein Akzeptor, so ist ein Eingabewort akzeptiert, wenn der Endzustand ein akzeptierter Zustand ist.

Definition

Eine Sprache L ist eine

- *aufzählbare Sprache*, wenn es eine Turingmaschine gibt, die L akzeptiert.
- *entscheidbare Sprache*, wenn es eine Turingmasche gibt, die L akzeptiert und *immer* hält.

Was ist der Unterschied?

TM als Akzeptor

Ist eine Turingmasche ein Akzeptor, so ist ein Eingabewort akzeptiert, wenn der Endzustand ein akzeptierter Zustand ist.

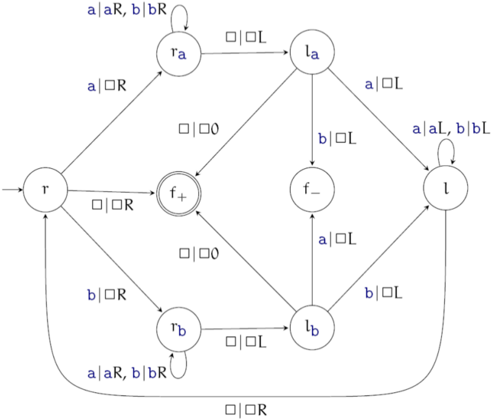
Definition

Eine Sprache L ist eine

- *aufzählbare Sprache*, wenn es eine Turingmaschine gibt, die L akzeptiert.
- *entscheidbare Sprache*, wenn es eine Turingmasche gibt, die L akzeptiert und *immer* hält.

Was ist der Unterschied? Die TM kann jedes Wort $w \in L$ akzeptieren, jedoch muss nicht zwangsläufig für ein Wort $w \notin L$ halten.

Was kann diese Turingmaschine?



Geben sei folgende Turingmaschine T

- Zustandsmenge $Z = \{r, s, u, d_b, d_a\}$
- Anfangszustand ist r
- Bandalphabet $X = \{\square, , a, b, 0, 1\}$
- Die Arbeitsweise ist wie folgt festgelegt

	r	s	u	d_b	d_a
0	(r,0,1)	(s,1,-1)	(r,0,1)	-	-
1	(r,1,1)	(r,0,1)	(r,1,1)	$(d_b, \square, 1)$	-
a	(s,b,-1)	-	-	-	$(d_a, \square, 1)$
b	(r,b,1)	(s,b,-1)	(u,a,-1)	$(d_a, \square, 1)$	$(d_a, b, 1)$
$\square$	(u,$\square$,-1)	$(d_b, \square, 1)$	-	-	-

Die Turingmaschine wird im folgenden benutzt für Bandbeschriftungen, bei denen zu Beginn der Berechnung auf dem Band ein Wort $v \in \{0, 1\}^+ \circ \{a\}^+$ steht, das von Blanksymbolen umgeben ist.
Der Kopf der Turingmaschine stehe anfangs auf dem ersten Symbol des Eingabewortes.

Punkte: 4/9/47

Geben Sie für die Eingabe $0100aaa$ folgende Konfiguration an

- die Anfangskonfiguration
- die Endkonfiguration
- jede Konfiguration, die in einem Zeitschritt vorliegt, nachdem die Turingmaschine von einem Zustand ungleich r in den Zustand r wechselt.

Punkte: 4/9/47

Geben Sie für die Eingabe $0100aaa$ folgende Konfiguration an

- die Anfangskonfiguration
- die Endkonfiguration
- jede Konfiguration, die in einem Zeitschritt vorliegt, nachdem die Turingmaschine von einem Zustand ungleich r in den Zustand r wechselt.

Zustand vor das Symbol auf dem sich der Zeiger befindet

Anfangskonfiguration $r0100aaa$

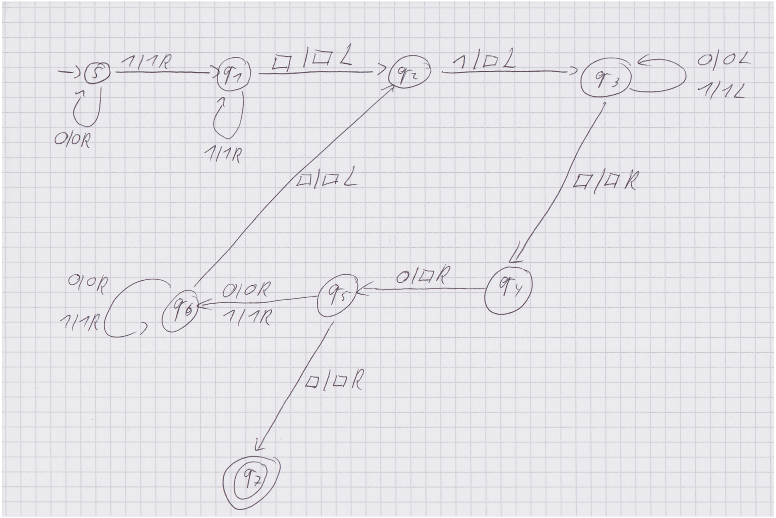
Zwischenkonfigurationen

- 00r11baa
- 0010rbba
- 000r1bbb
- 0001raaa
- 0000rbaa

Endkonfiguration $b\square d_a\square$

Entwerfen Sie eine Turingmaschine, die $w \in \{0^k 1^k | k \in \mathbb{N}_+\}$ akzeptiert.

Entwerfen Sie eine Turingmaschine, die $w \in \{0^k 1^k | k \in \mathbb{N}_+\}$ akzeptiert.

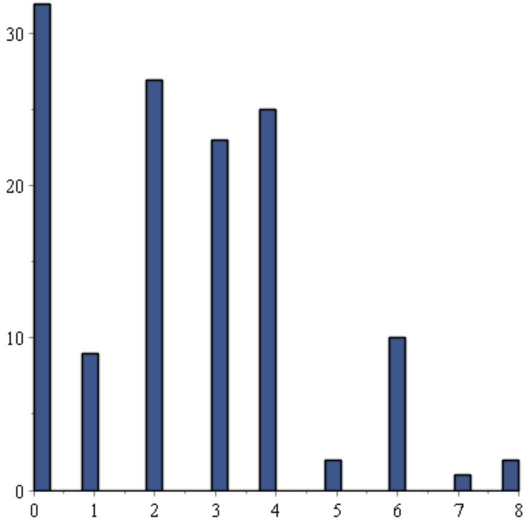


Probeklausur Statistika

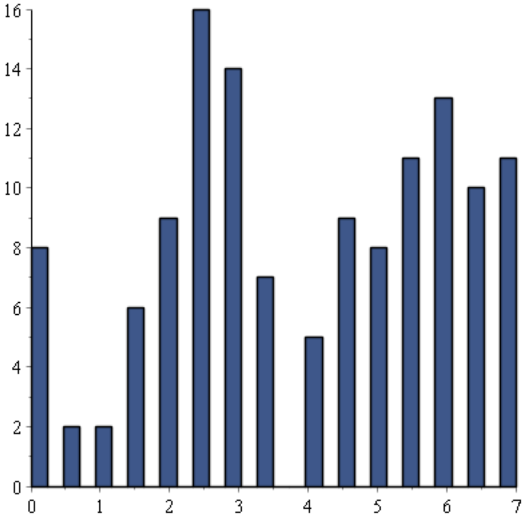
Notentabelle

0.00 - 17.0	5.0
17.5 - 19.0	4.0
19.5 - 21.0	3.7
21.5 - 23.0	3.3
23.5 - 25.0	3.0
25.5 - 27.0	2.7
27.5 - 29.0	2.3
29.5 - 31.0	2.0
31.5 - 33.0	1.7
33.5 - 35.0	1.3
35.5 - 37.0	1.0

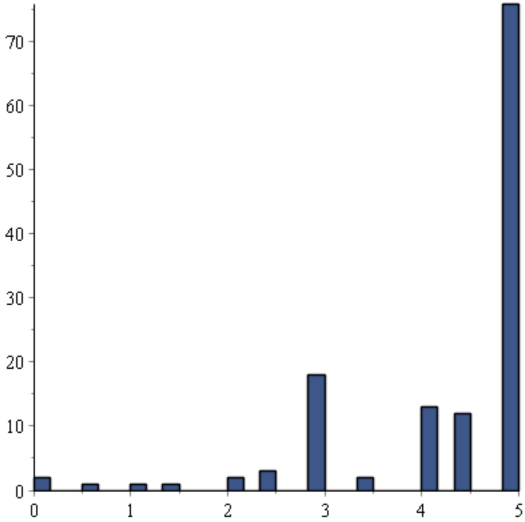
Aufgabe 1



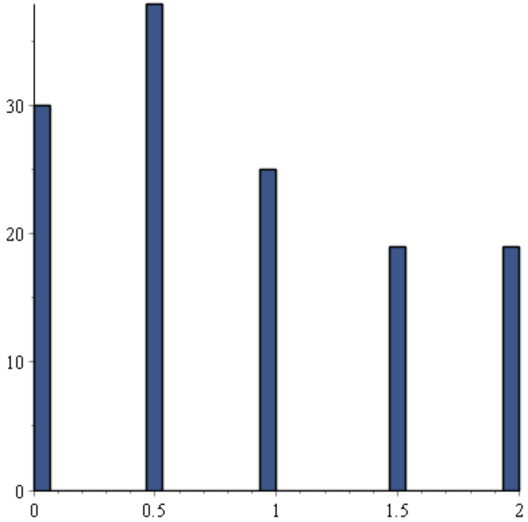
Aufgabe 2



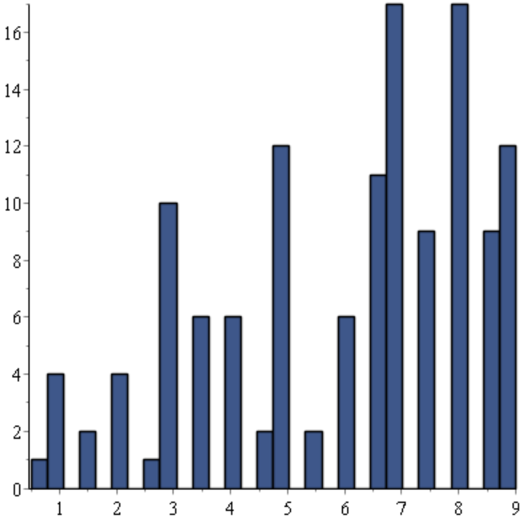
Aufgabe 3



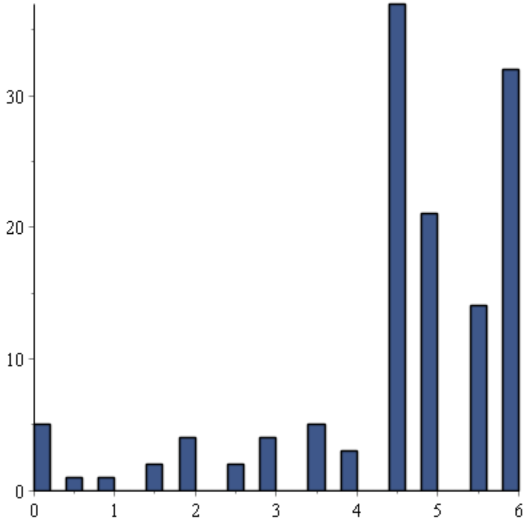
Aufgabe 4



Aufgabe 5



Aufgabe 6



Summe