3.0 VU Formale Modellierung

Gernot Salzer

Forschungsbereich Theory and Logic
Institut für Logic and Computation

2.4.2019

Was Sie letztes Mal hörten

1. Organisatorisches
2. Was bedeutet Modellierung?
3. Aussagenlogik
4. Endliche Automaten
   - 4.1. Beispiele
   - 4.2. Klassifikation
   - 4.3. Grundlagen formaler Sprachen
   - 4.4. Deterministische endliche Automaten
   - 4.5. Nichtdeterministische endliche Automaten
   - 4.6. Determinisierung
   - 4.7. Transducer
     - 4.7.1. Endlicher Transducer
     - 4.7.2. Mealy-Automaten
     - 4.7.3. Moore-Automaten
   - 4.8. Büchi-Automaten

Endliche Automaten

Bezeichnungen

Q	endliche Menge von Zuständen
q_0	Anfangszustand
I	Menge von Anfangszuständen
F	Menge von Endzuständen

Endliche Automaten

Bezeichnungen

Q	endliche Menge von Zuständen
q_0	Anfangszustand
I	Menge von Anfangszuständen
F	Menge von Endzuständen
Σ	Eingabealphabet
Γ	Ausgabealphabet

Endliche Automaten

Bezeichnungen

Q	endliche Menge von Zuständen
q_0	Anfangszustand
I	Menge von Anfangszuständen
F	Menge von Endzuständen
Σ	Eingabealphabet
Γ	Ausgabealphabet
δ	Übergangsfunktion/-relation
δ^*	erweiterte Übergangsfunktion/-relation
γ	Ausgabefunktion
γ^*	erweiterte Ausgabefunktion

Endliche Automaten

Bezeichnungen

Q	endliche Menge von Zuständen
q_0	Anfangszustand
I	Menge von Anfangszuständen
F	Menge von Endzuständen
Σ	Eingabealphabet
Γ	Ausgabealphabet
δ	Übergangsfunktion/-relation
δ^*	erweiterte Übergangsfunktion/-relation
γ	Ausgabefunktion
γ^*	erweiterte Ausgabefunktion
$\mathcal{A}$	Automat
$\mathcal{L}(\mathcal{A})$	die von $\mathcal{A}$ akzeptierte Sprache
$[\mathcal{A}]$	die von $\mathcal{A}$ berechnete Übersetzungsfunktion/-relation

Übergangsfunktionen und -relationen

$\delta\colon Q \times \Sigma \mapsto Q$	det. endlicher Automat (DEA) det. Büchi-Automat

Übergangsfunktionen und -relationen

$\delta\colon Q \times \Sigma \mapsto Q$	det. endlicher Automat (DEA)
	det. Büchi-Automat
$\delta \subseteq Q \times \Sigma \times Q$	nichtdet. endl. Aut. (NEA) ohne ε-Ü.
$\delta\colon Q \times \Sigma \mapsto 2^Q$	nichtdet. Büchi-Automat

Übergangsfunktionen und -relationen

$\delta\colon Q \times \Sigma \mapsto Q$	det. endlicher Automat (DEA) det. Büchi-Automat
$\delta \subseteq Q \times \Sigma \times Q$ $\delta\colon Q \times \Sigma \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) ohne ε-Ü. nichtdet. Büchi-Automat
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) mit ε-Ü.

Übergangsfunktionen und -relationen

$\delta\colon Q \times \Sigma \mapsto Q$	det. endlicher Automat (DEA) det. Büchi-Automat
$\delta \subseteq Q \times \Sigma \times Q$ $\delta\colon Q \times \Sigma \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) ohne ε-Ü. nichtdet. Büchi-Automat
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) mit ε-Ü.
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^{(\Gamma \cup \{\varepsilon\}) \times Q}$	Transducer (indet. mit ε-Ü.)

Übergangsfunktionen und -relationen

$\delta\colon Q \times \Sigma \mapsto Q$	det. endlicher Automat (DEA) det. Büchi-Automat
$\delta \subseteq Q \times \Sigma \times Q$ $\delta\colon Q \times \Sigma \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) ohne ε-Ü. nichtdet. Büchi-Automat
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) mit ε-Ü.
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^{(\Gamma \cup \{\varepsilon\}) \times Q}$	Transducer (indet. mit ε-Ü.)
$\delta\colon Q \times \Sigma \mapsto \Gamma \times Q$ $\delta\colon Q \times \Sigma \mapsto Q \quad \gamma\colon Q \times \Sigma \mapsto \Gamma$	Mealy-Automat

Übergangsfunktionen und -relationen

$\delta\colon Q \times \Sigma \mapsto Q$	det. endlicher Automat (DEA) det. Büchi-Automat
$\delta \subseteq Q \times \Sigma \times Q$ $\delta\colon Q \times \Sigma \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) ohne ε-Ü. nichtdet. Büchi-Automat
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) mit ε-Ü.
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^{(\Gamma \cup \{\varepsilon\}) \times Q}$	Transducer (indet. mit ε-Ü.)
$\delta\colon Q \times \Sigma \mapsto \Gamma \times Q$ $\delta\colon Q \times \Sigma \mapsto Q \quad \gamma\colon Q \times \Sigma \mapsto \Gamma$	Mealy-Automat
$\delta\colon Q \times \Sigma \mapsto Q \quad \gamma\colon Q \mapsto \Gamma$	Moore-Automat

Übergangsfunktionen und -relationen

$\delta\colon Q \times \Sigma \mapsto Q$	det. endlicher Automat (DEA) det. Büchi-Automat
$\delta \subseteq Q \times \Sigma \times Q$ $\delta\colon Q \times \Sigma \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) ohne ε-Ü. nichtdet. Büchi-Automat
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^Q$	nichtdet. endl. Aut. (NEA) mit ε-Ü.
$\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$ $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^{(\Gamma \cup \{\varepsilon\}) \times Q}$	Transducer (indet. mit ε-Ü.)
$\delta\colon Q \times \Sigma \mapsto \Gamma \times Q$ $\delta\colon Q \times \Sigma \mapsto Q \quad \gamma\colon Q \times \Sigma \mapsto \Gamma$	Mealy-Automat
$\delta\colon Q \times \Sigma \mapsto Q \quad \gamma\colon Q \mapsto \Gamma$	Moore-Automat

ε-Übergänge, Relation oder Potenzmenge $\Longrightarrow$ nichtdeterm. Automat

Akzeptierte Wörter bzw. Wortpaare

- (Nicht-)Deterministischer Automat:
  Alle Wörter, mit denen man vom Anfangszustand aus einen der Endzustände erreicht.

Akzeptierte Wörter bzw. Wortpaare

- (Nicht-)Deterministischer Automat:
  Alle Wörter, mit denen man vom Anfangszustand aus einen der Endzustände erreicht.
- Büchi-Automat:
  Alle unendlichen Wörter, mit denen man vom Anfangszustand aus unendlich oft an einem Endzustand vorbei kommt.

Akzeptierte Wörter bzw. Wortpaare

- (Nicht-)Deterministischer Automat:
  Alle Wörter, mit denen man vom Anfangszustand aus einen der Endzustände erreicht.
- Büchi-Automat:
  Alle unendlichen Wörter, mit denen man vom Anfangszustand aus unendlich oft an einem Endzustand vorbei kommt.
- Transducer:
  Alle Paare (u, v) von Ein-/Ausgabewörtern, bei denen man mit u vom Anfangszustand (von einem der Anfangszustände) aus in einen Endzustand gelangt und dabei v ausgibt.

Akzeptierte Wörter bzw. Wortpaare

- (Nicht-)Deterministischer Automat:
  Alle Wörter, mit denen man vom Anfangszustand aus einen der Endzustände erreicht.
- Büchi-Automat:
  Alle unendlichen Wörter, mit denen man vom Anfangszustand aus unendlich oft an einem Endzustand vorbei kommt.
- Transducer:
  Alle Paare (u, v) von Ein-/Ausgabewörtern, bei denen man mit u vom Anfangszustand (von einem der Anfangszustände) aus in einen Endzustand gelangt und dabei v ausgibt.
  - Mealy: Pro Übergang wird genau ein Symbol gelesen und eines geschrieben. Die Ausgabe ist an den Übergang gebunden.

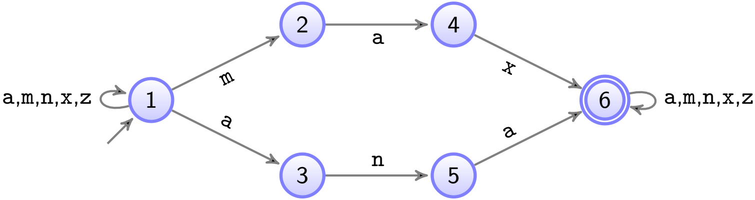
Akzeptierte Wörter bzw. Wortpaare

- (Nicht-)Deterministischer Automat:
  Alle Wörter, mit denen man vom Anfangszustand aus einen der Endzustände erreicht.
- Büchi-Automat:
  Alle unendlichen Wörter, mit denen man vom Anfangszustand aus unendlich oft an einem Endzustand vorbei kommt.
- Transducer:
  Alle Paare (u, v) von Ein-/Ausgabewörtern, bei denen man mit u vom Anfangszustand (von einem der Anfangszustände) aus in einen Endzustand gelangt und dabei v ausgibt.
  - Mealy: Pro Übergang wird genau ein Symbol gelesen und eines geschrieben. Die Ausgabe ist an den Übergang gebunden.
  - Moore: Pro Übergang wird genau ein Symbol gelesen und eines geschrieben. Die Ausgabe ist an den Zielzustand gebunden.

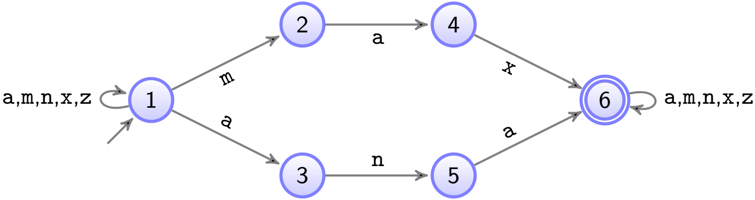
Was Sie letztes Mal hörten

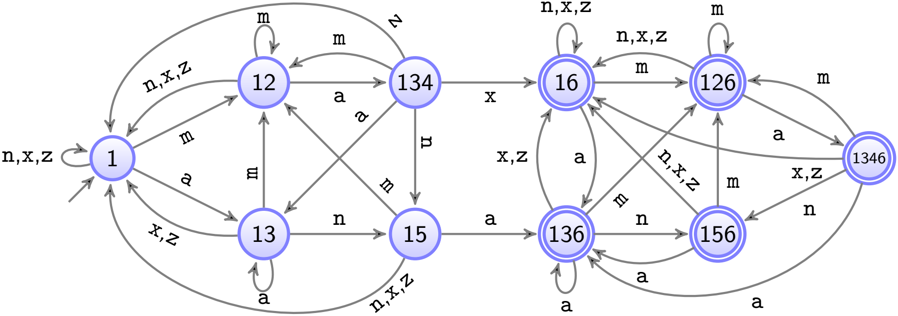
1. Organisatorisches
2. Was bedeutet Modellierung?
3. Aussagenlogik
4. Endliche Automaten
   4.1. Beispiele
   4.2. Klassifikation
   4.3. Grundlagen formaler Sprachen
   4.4. Deterministische endliche Automaten
   4.5. Nichtdeterministische endliche Automaten
   4.6. Determinisierung
   4.7. Transducer

Beispiel: Suche nach Max und Ana



Beispiel: Suche nach Max und Ana





Modellierung mit endlichen Automaten

Vorgangsweise:

1. Was sind die Zustände des Systems? Wieviele sind notwendig? Zustandsbezeichnungen?
2. Startzustand? Endzustände?
3. Was sind die Aktionen/Eingaben, die zu Zustandsübergängen führen? Bezeichnung?
4. Was sind die Aktionen/Ausgaben, die bei Zustandsübergängen stattfinden? Bezeichnung?
5. Lege für jeden Zustand und jede Eingabe die Folgezustände und die Ausgaben fest.

Was Sie heute erwartet

1. Organisatorisches
2. Was bedeutet Modellierung?
3. Aussagenlogik
4. Endliche Automaten
5. Reguläre Sprachen
   5.1. Operationen auf formalen Sprachen
   5.2. Definition regulärer Sprachen
   5.3. Reguläre Ausdrücke
   5.4. Eigenschaften regulärer Sprachen
   5.5. Vom regulären Ausdruck zum Automaten
   5.6. Vom Automaten zum regulären Ausdruck

Operationen auf formalen Sprachen

Σ Alphabet, d.h., endliche, nicht-leere Menge atomarer Symbole

w Wort über Σ, (endliche) Folge von Zeichen aus dem Alphabet Σ

ε Leerwort

Σ^* Menge aller endlichen Wörter über Σ (inklusive Leerwort)

$w \cdot w' = ww'$ Verkettung der Wörter $w, w' \in \Sigma^*$

Operationen auf formalen Sprachen

Σ Alphabet, d.h., endliche, nicht-leere Menge atomarer Symbole
w Wort über Σ, (endliche) Folge von Zeichen aus dem Alphabet Σ
ε Leerwort
Σ^* Menge aller endlichen Wörter über Σ (inklusive Leerwort)
$w \cdot w' = ww'$ Verkettung der Wörter $w, w' \in \Sigma^*$

Seien $L, L' \subseteq \Sigma^*$ zwei Sprachen.

$$L \cup L' = \{\, w \mid w \in L \text{ oder } w \in L' \,\} \qquad \text{Vereinigung}$$

Operationen auf formalen Sprachen

Σ	Alphabet, d.h., endliche, nicht-leere Menge atomarer Symbole
w	Wort über Σ, (endliche) Folge von Zeichen aus dem Alphabet Σ
ε	Leerwort
Σ^*	Menge aller endlichen Wörter über Σ (inklusive Leerwort)
$w \cdot w' = ww'$	Verkettung der Wörter $w, w' \in \Sigma^*$

Seien $L, L' \subseteq \Sigma^*$ zwei Sprachen.

$$L \cup L' = \{\, w \mid w \in L \text{ oder } w \in L' \,\} \qquad \text{Vereinigung}$$

$$L \cdot L' = \{\, w \cdot w' \mid w \in L, w' \in L' \,\} \qquad \text{Verkettung}$$

Operationen auf formalen Sprachen

Σ Alphabet, d.h., endliche, nicht-leere Menge atomarer Symbole
w Wort über Σ, (endliche) Folge von Zeichen aus dem Alphabet Σ
ε Leerwort
Σ^* Menge aller endlichen Wörter über Σ (inklusive Leerwort)
$w \cdot w' = ww'$ Verkettung der Wörter $w, w' \in \Sigma^*$

Seien $L, L' \subseteq \Sigma^*$ zwei Sprachen.

$$L \cup L' = \{ w \mid w \in L \text{ oder } w \in L' \} \qquad \text{Vereinigung}$$

$$L \cdot L' = \{ w \cdot w' \mid w \in L, w' \in L' \} \qquad \text{Verkettung}$$

$$\begin{aligned} L^0 &= \{\varepsilon\} \\ L^{n+1} &= L \cdot L^n \quad (n \geq 0) \end{aligned} \qquad \text{Potenzen}$$

Operationen auf formalen Sprachen

Σ Alphabet, d.h., endliche, nicht-leere Menge atomarer Symbole
w Wort über Σ, (endliche) Folge von Zeichen aus dem Alphabet Σ
ε Leerwort
Σ^* Menge aller endlichen Wörter über Σ (inklusive Leerwort)
$w \cdot w' = ww'$ Verkettung der Wörter $w, w' \in \Sigma^*$

Seien $L, L' \subseteq \Sigma^*$ zwei Sprachen.

$$L \cup L' = \{\, w \mid w \in L \text{ oder } w \in L' \,\} \qquad \text{Vereinigung}$$

$$L \cdot L' = \{\, w \cdot w' \mid w \in L, w' \in L' \,\} \qquad \text{Verkettung}$$

$$L^0 = \{\varepsilon\} \qquad \text{Potenzen}$$
$$L^{n+1} = L \cdot L^n \quad (n \geq 0)$$

$$L^+ = \bigcup_{n \geq 1} L^n$$
$$L^* = \bigcup_{n \geq 0} L^n = L^0 \cup L^+ = \{\varepsilon\} \cup L^+ \qquad \text{Kleene-Stern}$$

Verkettung

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\mathtt{b}, \mathtt{c}, \mathtt{d}\} = \{\mathtt{ab}, \mathtt{ac}, \mathtt{ad}, \mathtt{bb}, \mathtt{bc}, \mathtt{bd}\}$
$\{\mathtt{b}, \mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\mathtt{ba}, \mathtt{bb}, \mathtt{ca}, \mathtt{cb}, \mathtt{da}, \mathtt{db}\}$

Verkettung

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\mathtt{b}, \mathtt{c}, \mathtt{d}\} = \{\mathtt{ab}, \mathtt{ac}, \mathtt{ad}, \mathtt{bb}, \mathtt{bc}, \mathtt{bd}\}$
$\{\mathtt{b}, \mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\mathtt{ba}, \mathtt{bb}, \mathtt{ca}, \mathtt{cb}, \mathtt{da}, \mathtt{db}\}$

$(\{\mathtt{a}, \mathtt{b}\} \cdot \{1, 2\}) \cdot \{\#, \$\} = \{\mathtt{a1\#}, \mathtt{a1\$}, \mathtt{a2\#}, \mathtt{a2\$}, \mathtt{b1\#}, \mathtt{b1\$}, \mathtt{b2\#}, \mathtt{b2\$}\}$
$= \{\mathtt{a}, \mathtt{b}\} \cdot (\{1, 2\} \cdot \{\#, \$\})$

Verkettung

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\mathtt{b}, \mathtt{c}, \mathtt{d}\} = \{\mathtt{ab}, \mathtt{ac}, \mathtt{ad}, \mathtt{bb}, \mathtt{bc}, \mathtt{bd}\}$
$\{\mathtt{b}, \mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\mathtt{ba}, \mathtt{bb}, \mathtt{ca}, \mathtt{cb}, \mathtt{da}, \mathtt{db}\}$

$(\{\mathtt{a}, \mathtt{b}\} \cdot \{1, 2\}) \cdot \{\#, \$\} = \{\mathtt{a1\#}, \mathtt{a1\$}, \mathtt{a2\#}, \mathtt{a2\$}, \mathtt{b1\#}, \mathtt{b1\$}, \mathtt{b2\#}, \mathtt{b2\$}\}$
$= \{\mathtt{a}, \mathtt{b}\} \cdot (\{1, 2\} \cdot \{\#, \$\})$

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\varepsilon\} = \{\mathtt{a} \cdot \varepsilon, \mathtt{b} \cdot \varepsilon\} = \{\mathtt{a}, \mathtt{b}\} = \{\varepsilon \cdot \mathtt{a}, \varepsilon \cdot \mathtt{b}\} = \{\varepsilon\} \cdot \{\mathtt{a}, \mathtt{b}\}$

Verkettung

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\mathtt{b}, \mathtt{c}, \mathtt{d}\} = \{\mathtt{ab}, \mathtt{ac}, \mathtt{ad}, \mathtt{bb}, \mathtt{bc}, \mathtt{bd}\}$
$\{\mathtt{b}, \mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\mathtt{ba}, \mathtt{bb}, \mathtt{ca}, \mathtt{cb}, \mathtt{da}, \mathtt{db}\}$

$(\{\mathtt{a}, \mathtt{b}\} \cdot \{1, 2\}) \cdot \{\#, \$\} = \{\mathtt{a1\#}, \mathtt{a1\$}, \mathtt{a2\#}, \mathtt{a2\$}, \mathtt{b1\#}, \mathtt{b1\$}, \mathtt{b2\#}, \mathtt{b2\$}\}$
$= \{\mathtt{a}, \mathtt{b}\} \cdot (\{1, 2\} \cdot \{\#, \$\})$

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\varepsilon\} = \{\mathtt{a} \cdot \varepsilon, \mathtt{b} \cdot \varepsilon\} = \{\mathtt{a}, \mathtt{b}\} = \{\varepsilon \cdot \mathtt{a}, \varepsilon \cdot \mathtt{b}\} = \{\varepsilon\} \cdot \{\mathtt{a}, \mathtt{b}\}$

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\} = \{\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\}$

Verkettung

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\mathtt{b}, \mathtt{c}, \mathtt{d}\} = \{\mathtt{ab}, \mathtt{ac}, \mathtt{ad}, \mathtt{bb}, \mathtt{bc}, \mathtt{bd}\}$
$\{\mathtt{b}, \mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\mathtt{ba}, \mathtt{bb}, \mathtt{ca}, \mathtt{cb}, \mathtt{da}, \mathtt{db}\}$

$(\{\mathtt{a}, \mathtt{b}\} \cdot \{\mathtt{1}, \mathtt{2}\}) \cdot \{\mathtt{\#}, \mathtt{\$}\} = \{\mathtt{a1\#}, \mathtt{a1\$}, \mathtt{a2\#}, \mathtt{a2\$}, \mathtt{b1\#}, \mathtt{b1\$}, \mathtt{b2\#}, \mathtt{b2\$}\}$
$= \{\mathtt{a}, \mathtt{b}\} \cdot (\{\mathtt{1}, \mathtt{2}\} \cdot \{\mathtt{\#}, \mathtt{\$}\})$

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\varepsilon\} = \{\mathtt{a} \cdot \varepsilon, \mathtt{b} \cdot \varepsilon\} = \{\mathtt{a}, \mathtt{b}\} = \{\varepsilon \cdot \mathtt{a}, \varepsilon \cdot \mathtt{b}\} = \{\varepsilon\} \cdot \{\mathtt{a}, \mathtt{b}\}$

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\} = \{\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\}$

$\{\varepsilon\} \cdot \{\varepsilon\} = \{\varepsilon\}$
$\{\} \cdot \{\} = \{\varepsilon\} \cdot \{\} = \{\} \cdot \{\varepsilon\} = \{\}$

Verkettung

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\mathtt{b}, \mathtt{c}, \mathtt{d}\} = \{\mathtt{ab}, \mathtt{ac}, \mathtt{ad}, \mathtt{bb}, \mathtt{bc}, \mathtt{bd}\}$
$\{\mathtt{b}, \mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\mathtt{ba}, \mathtt{bb}, \mathtt{ca}, \mathtt{cb}, \mathtt{da}, \mathtt{db}\}$

$(\{\mathtt{a}, \mathtt{b}\} \cdot \{\mathtt{1}, \mathtt{2}\}) \cdot \{\mathtt{\#}, \mathtt{\$}\} = \{\mathtt{a1\#}, \mathtt{a1\$}, \mathtt{a2\#}, \mathtt{a2\$}, \mathtt{b1\#}, \mathtt{b1\$}, \mathtt{b2\#}, \mathtt{b2\$}\}$
$= \{\mathtt{a}, \mathtt{b}\} \cdot (\{\mathtt{1}, \mathtt{2}\} \cdot \{\mathtt{\#}, \mathtt{\$}\})$

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\varepsilon\} = \{\mathtt{a} \cdot \varepsilon, \mathtt{b} \cdot \varepsilon\} = \{\mathtt{a}, \mathtt{b}\} = \{\varepsilon \cdot \mathtt{a}, \varepsilon \cdot \mathtt{b}\} = \{\varepsilon\} \cdot \{\mathtt{a}, \mathtt{b}\}$

$\{\mathtt{a}, \mathtt{b}\} \cdot \{\} = \{\} \cdot \{\mathtt{a}, \mathtt{b}\} = \{\}$

$\{\varepsilon\} \cdot \{\varepsilon\} = \{\varepsilon\}$
$\{\} \cdot \{\} = \{\varepsilon\} \cdot \{\} = \{\} \cdot \{\varepsilon\} = \{\}$

Beobachtungen:

- Sprachverkettung ist nicht kommutativ.
- Sprachverkettung ist assoziativ.
- $\{\varepsilon\}$ ist neutrales Element bzgl. Sprachverkettung.
- $\{\}$ ist Nullelement bzgl. Sprachverkettung.

Potenzen von $\{\mathtt{a}, \mathtt{42}\}$

$$
\begin{aligned}
L &= \{\mathtt{a}, \mathtt{42}\} \\
L^0 &= \{\varepsilon\} \\
L^1 &= L \cdot L^0 = L \cdot \{\varepsilon\} = L = \{\mathtt{a}, \mathtt{42}\} \\
L^2 &= L \cdot L^1 = L \cdot L = \{\mathtt{aa}, \mathtt{a42}, \mathtt{42a}, \mathtt{4242}\} \\
L^3 &= L \cdot L^2 = \{\mathtt{aaa}, \mathtt{aa42}, \mathtt{a42a}, \mathtt{a4242}, \mathtt{42aa}, \mathtt{42a42}, \mathtt{4242a}, \mathtt{424242}\} \\
&\vdots \\
L^+ &= \textstyle\bigcup_{n \geq 1} L^n = L^1 \cup L^2 \cup L^3 \cup \cdots \\
&= \{\mathtt{a}, \mathtt{42}, \mathtt{aa}, \mathtt{a42}, \mathtt{42a}, \mathtt{4242}, \mathtt{aaa}, \mathtt{aa42}, \mathtt{a42a}, \mathtt{a4242}, \mathtt{42aa}, \dots\} \\
L^* &= \textstyle\bigcup_{n \geq 0} L^n = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \cdots \\
&= \{\varepsilon, \mathtt{a}, \mathtt{42}, \mathtt{aa}, \mathtt{a42}, \mathtt{42a}, \mathtt{4242}, \mathtt{aaa}, \mathtt{aa42}, \mathtt{a42a}, \mathtt{a4242}, \mathtt{42aa}, \dots\}
\end{aligned}
$$

Potenzen von $\{\mathtt{a}, 42\}$

$$
\begin{aligned}
L &= \{\mathtt{a}, 42\} \\
L^0 &= \{\varepsilon\} \\
L^1 &= L \cdot L^0 = L \cdot \{\varepsilon\} = L = \{\mathtt{a}, 42\} \\
L^2 &= L \cdot L^1 = L \cdot L = \{\mathtt{aa}, \mathtt{a42}, \mathtt{42a}, 4242\} \\
L^3 &= L \cdot L^2 = \{\mathtt{aaa}, \mathtt{aa42}, \mathtt{a42a}, \mathtt{a4242}, \mathtt{42aa}, \mathtt{42a42}, \mathtt{4242a}, 424242\} \\
&\vdots \\
L^+ &= \textstyle\bigcup_{n \geq 1} L^n = L^1 \cup L^2 \cup L^3 \cup \cdots \\
&= \{\mathtt{a}, 42, \mathtt{aa}, \mathtt{a42}, \mathtt{42a}, 4242, \mathtt{aaa}, \mathtt{aa42}, \mathtt{a42a}, \mathtt{a4242}, \mathtt{42aa}, \dots\} \\
L^* &= \textstyle\bigcup_{n \geq 0} L^n = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \cdots \\
&= \{\varepsilon, \mathtt{a}, 42, \mathtt{aa}, \mathtt{a42}, \mathtt{42a}, 4242, \mathtt{aaa}, \mathtt{aa42}, \mathtt{a42a}, \mathtt{a4242}, \mathtt{42aa}, \dots\}
\end{aligned}
$$

Potenzen eines Alphabets Σ

$\Sigma^0 = \{\varepsilon\}$	$\Sigma^1 = \Sigma$
Σ^n	alle Σ-Wörter der Länge n (d.h., mit n Symbolen)
$\Sigma^+ = \bigcup_{n \geq 1} \Sigma^n$	alle Σ-Wörter ohne Leerwort
$\Sigma^* = \bigcup_{n \geq 0} \Sigma^n$	alle Σ-Wörter mit Leerwort

$\langle 2^{\Sigma^*}, \cup, \cdot, \{\}, \{\varepsilon\}\rangle$ bildet einen idempotenten Halbring

Das heißt, es gelten folgende Gleichungen.

$\langle 2^{\Sigma^*}, \cup, \{\}\rangle$ ... idemp.komm.Monoid

$(A \cup B) \cup C = A \cup (B \cup C)$
$\{\} \cup A = A \cup \{\} = A$
$A \cup B = B \cup A$
$A \cup A = A$

$\langle 2^{\Sigma^*}, \cup, \cdot, \{\}, \{\varepsilon\} \rangle$ bildet einen idempotenten Halbring

Das heißt, es gelten folgende Gleichungen.

$\langle 2^{\Sigma^*}, \cup, \{\} \rangle$ … idemp.komm.Monoid

$(A \cup B) \cup C = A \cup (B \cup C)$
$\{\} \cup A = A \cup \{\} = A$
$A \cup B = B \cup A$
$A \cup A = A$

$\langle 2^{\Sigma^*}, \cdot, \{\varepsilon\} \rangle$ … Monoid

$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
$\{\varepsilon\} \cdot A = A \cdot \{\varepsilon\} = A$

$\langle 2^{\Sigma^*}, \cup, \cdot, \{\}, \{\varepsilon\} \rangle$ bildet einen idempotenten Halbring

Das heißt, es gelten folgende Gleichungen.

$\langle 2^{\Sigma^*}, \cup, \{\} \rangle$ … idemp.komm.Monoid

$(A \cup B) \cup C = A \cup (B \cup C)$
$\{\} \cup A = A \cup \{\} = A$
$A \cup B = B \cup A$
$A \cup A = A$

$\langle 2^{\Sigma^*}, \cdot, \{\varepsilon\} \rangle$ … Monoid

$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
$\{\varepsilon\} \cdot A = A \cdot \{\varepsilon\} = A$

Verkettung distribuiert über Vereinigung.

$A \cdot (B \cup C) = (A \cdot B) \cup (A \cdot C) \qquad (B \cup C) \cdot A = (B \cdot A) \cup (C \cdot A)$

$\langle 2^{\Sigma^*}, \cup, \cdot, \{\}, \{\varepsilon\}\rangle$ bildet einen idempotenten Halbring

Das heißt, es gelten folgende Gleichungen.

$\langle 2^{\Sigma^*}, \cup, \{\}\rangle$ ... idemp.komm.Monoid

$(A \cup B) \cup C = A \cup (B \cup C)$
$\{\} \cup A = A \cup \{\} = A$
$A \cup B = B \cup A$
$A \cup A = A$

$\langle 2^{\Sigma^*}, \cdot, \{\varepsilon\}\rangle$ ... Monoid

$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
$\{\varepsilon\} \cdot A = A \cdot \{\varepsilon\} = A$

Verkettung distribuiert über Vereinigung.

$A \cdot (B \cup C) = (A \cdot B) \cup (A \cdot C) \qquad (B \cup C) \cdot A = (B \cdot A) \cup (C \cdot A)$

$\{\}$ ist Nullelement bzgl. Verkettung.

$\{\} \cdot A = A \cdot \{\} = \{\}$

$\langle 2^{\Sigma^*}, \cup, \cdot, \{\}, \{\varepsilon\} \rangle$ bildet einen idempotenten Halbring

Das heißt, es gelten folgende Gleichungen.

$\langle 2^{\Sigma^*}, \cup, \{\} \rangle$ … idemp.komm.Monoid

$(A \cup B) \cup C = A \cup (B \cup C)$
$\{\} \cup A = A \cup \{\} = A$
$A \cup B = B \cup A$
$A \cup A = A$

$\langle 2^{\Sigma^*}, \cdot, \{\varepsilon\} \rangle$ … Monoid

$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
$\{\varepsilon\} \cdot A = A \cdot \{\varepsilon\} = A$

Verkettung distribuiert über Vereinigung.

$A \cdot (B \cup C) = (A \cdot B) \cup (A \cdot C) \qquad (B \cup C) \cdot A = (B \cdot A) \cup (C \cdot A)$

$\{\}$ ist Nullelement bzgl. Verkettung.

$\{\} \cdot A = A \cdot \{\} = \{\}$

Weitere Identitäten für $^+$ und *:

$(A^*)^* = A^* \qquad (A \cup \{\varepsilon\})^* = A^* \qquad A^* \cdot A = A^+ \qquad A^+ \cup \{\varepsilon\} = A^*$

Was Sie heute erwartet

1. Organisatorisches
2. Was bedeutet Modellierung?
3. Aussagenlogik
4. Endliche Automaten
5. Reguläre Sprachen
   5.1. Operationen auf formalen Sprachen
   5.2. Definition regulärer Sprachen
   5.3. Reguläre Ausdrücke
   5.4. Eigenschaften regulärer Sprachen
   5.5. Vom regulären Ausdruck zum Automaten
   5.6. Vom Automaten zum regulären Ausdruck

Reguläre Sprachen

Alle Sprachen, die aus einem Alphabet mit Hilfe von Vereinigung, Verkettung und Stern gebildet werden können.

Reguläre Sprachen

Alle Sprachen, die aus einem Alphabet mit Hilfe von Vereinigung, Verkettung und Stern gebildet werden können.

Anwendungen:

- Betriebssytem-Shells: DOS („Wildcards“), UNIX-Shells (wie `sh`, `csh`, `ash`, `bash`, `zsh`), ...
- UNIX Kommandozeilenprogramme: `grep`, `awk`, `ed`, `sed`, ...
- Editoren: `vi`, `emacs`, ...

Reguläre Sprachen

Alle Sprachen, die aus einem Alphabet mit Hilfe von Vereinigung, Verkettung und Stern gebildet werden können.

Anwendungen:

- Betriebssytem-Shells: Dos („Wildcards“), Unix-Shells (wie `sh`, `csh`, `ash`, `bash`, `zsh`), ...
- Unix Kommandozeilenprogramme: `grep`, `awk`, `ed`, `sed`, ...
- Editoren: `vi`, `emacs`, ...
- Compilerbau: *Tokens* bilden reguläre Sprache, die durch sog. Scanner (Lexer) wie `lex` oder `flex` verarbeitet werden.

Reguläre Sprachen

Alle Sprachen, die aus einem Alphabet mit Hilfe von Vereinigung, Verkettung und Stern gebildet werden können.

Anwendungen:

- Betriebssytem-Shells: DOS („Wildcards“), UNIX-Shells (wie `sh`, `csh`, `ash`, `bash`, `zsh`), ...
- UNIX Kommandozeilenprogramme: `grep`, `awk`, `ed`, `sed`, ...
- Editoren: `vi`, `emacs`, ...
- Compilerbau: *Tokens* bilden reguläre Sprache, die durch sog. Scanner (Lexer) wie `lex` oder `flex` verarbeitet werden.
- Programmiersprachen: PERL, TCL, PHP, PYTHON, RUBY, R, JAVA, JAVASCRIPT, .NET-Sprachen, ...
- Websprachen: XML Schema, XQuery, XPath, DTDs, ...
- Datenbanken: MySQL, Oracle, PostgreSQL, ...
- ...

Regulären Sprachen über einem Alphabet

Die Menge der regulären Sprachen über Σ, $\mathcal{L}_{\mathrm{reg}}(\Sigma)$, ist die kleinste Menge, sodass gilt:

- $\{\}$, $\{\varepsilon\}$ und $\{s\}$ sind reguläre Sprachen (für alle $s \in \Sigma$).
- Wenn L und L' reguläre Sprachen sind, dann auch $L \cup L'$, $L \cdot L'$ und L^*.

Regulären Sprachen über einem Alphabet

Die Menge der regulären Sprachen über Σ, $\mathcal{L}_{\mathrm{reg}}(\Sigma)$, ist die kleinste Menge, sodass gilt:

- $\{\}$, $\{\varepsilon\}$ und $\{s\}$ sind reguläre Sprachen (für alle $s \in \Sigma$).
- Wenn L und L' reguläre Sprachen sind, dann auch $L \cup L'$, $L \cdot L'$ und L^*.

Reellen Numerale: reguläre Sprache über $\Sigma = \{0, \ldots, 9, \texttt{.}, \texttt{E}, \texttt{+}, \texttt{-}\}$

$$
\begin{aligned}
\mathit{real} &= \mathit{digit} \cdot \mathit{digit}^* \cdot \{\texttt{.}\} \cdot \mathit{digit}^* \cdot (\{\varepsilon\} \cup \mathit{scale}) \\
\mathit{scale} &= \{\texttt{E}\} \cdot \{\texttt{+}, \texttt{-}, \varepsilon\} \cdot \mathit{digit} \cdot \mathit{digit}^* \\
\mathit{digit} &= \{0, \ldots, 9\} = \{0\} \cup \cdots \cup \{9\}
\end{aligned}
$$

Regulären Sprachen über einem Alphabet

Die Menge der regulären Sprachen über Σ, $\mathcal{L}_{\text{reg}}(\Sigma)$, ist die kleinste Menge, sodass gilt:

- $\{\}$, $\{\varepsilon\}$ und $\{s\}$ sind reguläre Sprachen (für alle $s \in \Sigma$).
- Wenn L und L' reguläre Sprachen sind, dann auch $L \cup L'$, $L \cdot L'$ und L^*.

Reellen Numerale: reguläre Sprache über $\Sigma = \{0, \ldots, 9, \texttt{.}, \texttt{E}, \texttt{+}, \texttt{-}\}$

$$\begin{aligned}
\mathit{real} &= \mathit{digit} \cdot \mathit{digit}^* \cdot \{\texttt{.}\} \cdot \mathit{digit}^* \cdot (\{\varepsilon\} \cup \mathit{scale}) \\
\mathit{scale} &= \{\texttt{E}\} \cdot \{\texttt{+}, \texttt{-}, \varepsilon\} \cdot \mathit{digit} \cdot \mathit{digit}^* \\
\mathit{digit} &= \{0, \ldots, 9\} = \{0\} \cup \cdots \cup \{9\}
\end{aligned}$$

Wichtig: Unterscheide Symbole des Alphabets von Meta-Symbolen!

$0, \ldots, 9, \texttt{.}, \texttt{E}, \texttt{+}, \texttt{-}$ ... Symbole des Alphabets
ε, *real*, *scale*, *digit* ... Meta-Symbole, Abkürzungen

Was Sie heute erwartet

1. Organisatorisches
2. Was bedeutet Modellierung?
3. Aussagenlogik
4. Endliche Automaten
5. Reguläre Sprachen
   5.1. Operationen auf formalen Sprachen
   5.2. Definition regulärer Sprachen
   5.3. Reguläre Ausdrücke
   5.4. Eigenschaften regulärer Sprachen
   5.5. Vom regulären Ausdruck zum Automaten
   5.6. Vom Automaten zum regulären Ausdruck
6. Kontextfreie Grammatiken

Reguläre Ausdrücke

Ausdrücke wie $\mathit{digit} \cdot \mathit{digit}^*$ und digit^+ sind ununterscheidbar: Beides sind semantische Beschreibungen der Menge aller Ziffernfolgen. Um Aussagen über ihre Form treffen zu können, benötigen wir eine formale Sprache.

Reguläre Ausdrücke (algebraische Notation)

Die regulären Ausdrücke über Σ sind die kleinste Menge, für die gilt:

- $\emptyset$, ε und s sind reguläre Ausdrücke (für alle Symbole $s \in \Sigma$).
- Sind r und r' reguläre Ausdrücke, dann auch $(r + r')$, (rr') und r^*.

Vereinfachte Klammerung: $+$ bindet am schwächsten, $*$ am stärksten.
Keine Klammern bei gleichartigen Operatoren (wegen Assoziativität).

Reguläre Ausdrücke

Ausdrücke wie $\mathit{digit} \cdot \mathit{digit}^*$ und digit^+ sind ununterscheidbar: Beides sind semantische Beschreibungen der Menge aller Ziffernfolgen. Um Aussagen über ihre Form treffen zu können, benötigen wir eine formale Sprache.

Reguläre Ausdrücke (algebraische Notation)

Die regulären Ausdrücke über Σ sind die kleinste Menge, für die gilt:

- $\emptyset$, ε und s sind reguläre Ausdrücke (für alle Symbole $s \in \Sigma$).
- Sind r und r' reguläre Ausdrücke, dann auch $(r + r')$, (rr') und r^*.

Vereinfachte Klammerung: $+$ bindet am schwächsten, $*$ am stärksten. Keine Klammern bei gleichartigen Operatoren (wegen Assoziativität).

Die Sprache $\mathcal{L}(r)$ zu einem regulären Ausdruck r ist definiert durch:

$$
\begin{aligned}
\mathcal{L}(\emptyset) &= \{\} & \mathcal{L}(r + r') &= \mathcal{L}(r) \cup \mathcal{L}(r') \\
\mathcal{L}(\varepsilon) &= \{\varepsilon\} & \mathcal{L}(rr') &= \mathcal{L}(r) \cdot \mathcal{L}(r') \\
\mathcal{L}(s) &= \{s\} \quad \text{für } s \in \Sigma & \mathcal{L}(r^*) &= (\mathcal{L}(r))^*
\end{aligned}
$$

Regulärer Ausdruck für die reellen Numerale

$R = DD^* \mathtt{.} D^*(\varepsilon + S)$
$S = \mathtt{E}(\mathtt{+} + \mathtt{-} + \varepsilon)DD^*$
$D = \mathtt{0} + \mathtt{1} + \cdots + \mathtt{9}$

(R, S und D sind Abkürzungen für die jeweiligen regulären Ausdrücke.)

Regulärer Ausdruck für die reellen Numerale

$R = DD^* \texttt{.} D^*(\varepsilon + S)$
$S = \texttt{E}(\texttt{+} + \texttt{-} + \varepsilon)DD^*$
$D = \texttt{0} + \texttt{1} + \cdots + \texttt{9}$

(R, S und D sind Abkürzungen für die jeweiligen regulären Ausdrücke.)

Die zugehörigen Sprachen:

$$\begin{aligned}\mathcal{L}(D) &= \mathcal{L}(\texttt{0} + \texttt{1} + \cdots + \texttt{9}) \\ &= \mathcal{L}(\texttt{0}) \cup \mathcal{L}(\texttt{1}) \cup \cdots \cup \mathcal{L}(\texttt{9}) \\ &= \{\texttt{0}\} \cup \{\texttt{1}\} \cup \cdots \cup \{\texttt{9}\} \\ &= \mathit{digits}\end{aligned}$$

Regulärer Ausdruck für die reellen Numerale

$R = DD^*\,\mathtt{.}\,D^*(\varepsilon + S)$
$S = \mathtt{E}(\mathtt{+} + \mathtt{-} + \varepsilon)DD^*$
$D = \mathtt{0} + \mathtt{1} + \cdots + \mathtt{9}$

(R, S und D sind Abkürzungen für die jeweiligen regulären Ausdrücke.)

Die zugehörigen Sprachen:

$$\begin{aligned}
\mathcal{L}(D) &= \mathcal{L}(\mathtt{0} + \mathtt{1} + \cdots + \mathtt{9}) \\
&= \mathcal{L}(\mathtt{0}) \cup \mathcal{L}(\mathtt{1}) \cup \cdots \cup \mathcal{L}(\mathtt{9}) \\
&= \{\mathtt{0}\} \cup \{\mathtt{1}\} \cup \cdots \cup \{\mathtt{9}\} \\
&= \mathit{digits}
\end{aligned}$$

$$\begin{aligned}
\mathcal{L}(S) &= \mathcal{L}(\mathtt{E}(\mathtt{+} + \mathtt{-} + \varepsilon)DD^*) \\
&= \mathcal{L}(\mathtt{E}) \cdot \mathcal{L}(\mathtt{+} + \mathtt{-} + \varepsilon) \cdot \mathcal{L}(D) \cdot \mathcal{L}(D^*) \\
&= \{\mathtt{E}\} \cdot (\mathcal{L}(\mathtt{+}) \cup \mathcal{L}(\mathtt{-}) \cup \mathcal{L}(\varepsilon)) \cdot \mathit{digits} \cdot \mathcal{L}(D)^* \\
&= \{\mathtt{E}\} \cdot (\{\mathtt{+}\} \cup \{\mathtt{-}\} \cup \{\varepsilon\}) \cdot \mathit{digits} \cdot \mathit{digits}^* \\
&= \mathit{scale}
\end{aligned}$$

Regulärer Ausdruck für die reellen Numerale

$R = DD^*\texttt{.}D^*(\varepsilon + S)$
$S = \texttt{E}(\texttt{+} + \texttt{-} + \varepsilon)DD^*$
$D = \texttt{0} + \texttt{1} + \cdots + \texttt{9}$

(R, S und D sind Abkürzungen für die jeweiligen regulären Ausdrücke.)

Die zugehörigen Sprachen:

$$\begin{aligned}\mathcal{L}(D) &= \mathcal{L}(\texttt{0} + \texttt{1} + \cdots + \texttt{9})\\ &= \mathcal{L}(\texttt{0}) \cup \mathcal{L}(\texttt{1}) \cup \cdots \cup \mathcal{L}(\texttt{9})\\ &= \{\texttt{0}\} \cup \{\texttt{1}\} \cup \cdots \cup \{\texttt{9}\}\\ &= \mathit{digits}\end{aligned}$$

$$\begin{aligned}\mathcal{L}(S) &= \mathcal{L}(\texttt{E}(\texttt{+} + \texttt{-} + \varepsilon)DD^*)\\ &= \mathcal{L}(\texttt{E}) \cdot \mathcal{L}(\texttt{+} + \texttt{-} + \varepsilon) \cdot \mathcal{L}(D) \cdot \mathcal{L}(D^*)\\ &= \{\texttt{E}\} \cdot (\mathcal{L}(\texttt{+}) \cup \mathcal{L}(\texttt{-}) \cup \mathcal{L}(\varepsilon)) \cdot \mathit{digits} \cdot \mathcal{L}(D)^*\\ &= \{\texttt{E}\} \cdot (\{\texttt{+}\} \cup \{\texttt{-}\} \cup \{\varepsilon\}) \cdot \mathit{digits} \cdot \mathit{digits}^*\\ &= \mathit{scale}\end{aligned}$$

$\mathcal{L}(R) = \cdots = \mathit{real}$

Zwei reguläre Ausdrücke r und r' heißen äquivalent, geschrieben $r = r'$, wenn $\mathcal{L}(r) = \mathcal{L}(r')$ gilt.

Zwei reguläre Ausdrücke r und r' heißen äquivalent, geschrieben $r = r'$, wenn $\mathcal{L}(r) = \mathcal{L}(r')$ gilt.

$((\mathtt{a} + \mathtt{b})^* + \varepsilon)^* = (\mathtt{a} + \mathtt{b})^*$

$\mathcal{L}(((\mathtt{a} + \mathtt{b})^* + \varepsilon)^*)$

Zwei reguläre Ausdrücke r und r' heißen äquivalent, geschrieben $r = r'$, wenn $\mathcal{L}(r) = \mathcal{L}(r')$ gilt.

$((\mathtt{a}+\mathtt{b})^* + \varepsilon)^* = (\mathtt{a}+\mathtt{b})^*$

$$\begin{aligned} \mathcal{L}(((\mathtt{a}+\mathtt{b})^* + \varepsilon)^*) &= \cdots \\ &= (\{\mathtt{a},\mathtt{b}\}^* \cup \{\varepsilon\})^* \end{aligned}$$

Zwei reguläre Ausdrücke r und r' heißen äquivalent, geschrieben $r = r'$, wenn $\mathcal{L}(r) = \mathcal{L}(r')$ gilt.

$((\mathtt{a} + \mathtt{b})^* + \varepsilon)^* = (\mathtt{a} + \mathtt{b})^*$

$$\begin{aligned} \mathcal{L}(((\mathtt{a} + \mathtt{b})^* + \varepsilon)^*) &= \cdots \\ &= (\{\mathtt{a}, \mathtt{b}\}^* \cup \{\varepsilon\})^* \\ &= (\{\mathtt{a}, \mathtt{b}\}^*)^* \qquad \text{da } \varepsilon \in L^* \text{ für alle } L \end{aligned}$$

Zwei reguläre Ausdrücke r und r' heißen äquivalent, geschrieben $r = r'$, wenn $\mathcal{L}(r) = \mathcal{L}(r')$ gilt.

$((\mathtt{a} + \mathtt{b})^* + \varepsilon)^* = (\mathtt{a} + \mathtt{b})^*$

$$
\begin{aligned}
\mathcal{L}(((\mathtt{a} + \mathtt{b})^* + \varepsilon)^*) &= \cdots \\
&= (\{\mathtt{a}, \mathtt{b}\}^* \cup \{\varepsilon\})^* \\
&= (\{\mathtt{a}, \mathtt{b}\}^*)^* \qquad \text{da } \varepsilon \in L^* \text{ für alle } L \\
&= (\{\mathtt{a}, \mathtt{b}\}^*)^0 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^1 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^2 \cup \cdots
\end{aligned}
$$

Zwei reguläre Ausdrücke r und r' heißen äquivalent, geschrieben $r = r'$, wenn $\mathcal{L}(r) = \mathcal{L}(r')$ gilt.

$((\mathtt{a} + \mathtt{b})^* + \varepsilon)^* = (\mathtt{a} + \mathtt{b})^*$

$$\begin{aligned}
\mathcal{L}(((\mathtt{a} + \mathtt{b})^* + \varepsilon)^*) &= \cdots \\
&= (\{\mathtt{a}, \mathtt{b}\}^* \cup \{\varepsilon\})^* \\
&= (\{\mathtt{a}, \mathtt{b}\}^*)^* \qquad \text{da } \varepsilon \in L^* \text{ für alle } L \\
&= (\{\mathtt{a}, \mathtt{b}\}^*)^0 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^1 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^2 \cup \cdots \\
&= \{\mathtt{a}, \mathtt{b}\}^* \cup (\{\mathtt{a}, \mathtt{b}\}^*)^0 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^2 \cup \cdots
\end{aligned}$$

Zwei reguläre Ausdrücke r und r' heißen äquivalent, geschrieben $r = r'$, wenn $\mathcal{L}(r) = \mathcal{L}(r')$ gilt.

$((\mathtt{a} + \mathtt{b})^* + \varepsilon)^* = (\mathtt{a} + \mathtt{b})^*$

$$\begin{aligned}
\mathcal{L}(((\mathtt{a} + \mathtt{b})^* + \varepsilon)^*) &= \cdots \\
&= (\{\mathtt{a}, \mathtt{b}\}^* \cup \{\varepsilon\})^* \\
&= (\{\mathtt{a}, \mathtt{b}\}^*)^* \qquad \text{da } \varepsilon \in L^* \text{ für alle } L \\
&= (\{\mathtt{a}, \mathtt{b}\}^*)^0 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^1 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^2 \cup \cdots \\
&= \{\mathtt{a}, \mathtt{b}\}^* \cup (\{\mathtt{a}, \mathtt{b}\}^*)^0 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^2 \cup \cdots \\
&= \{\mathtt{a}, \mathtt{b}\}^* \qquad \text{da } L^* \text{ alle Wörter über } L \text{ enthält}
\end{aligned}$$

Zwei reguläre Ausdrücke r und r' heißen äquivalent, geschrieben $r = r'$, wenn $\mathcal{L}(r) = \mathcal{L}(r')$ gilt.

$((\mathtt{a} + \mathtt{b})^* + \varepsilon)^* = (\mathtt{a} + \mathtt{b})^*$

$$\begin{aligned}
\mathcal{L}(((\mathtt{a} + \mathtt{b})^* + \varepsilon)^*) &= \cdots \\
&= (\{\mathtt{a}, \mathtt{b}\}^* \cup \{\varepsilon\})^* \\
&= (\{\mathtt{a}, \mathtt{b}\}^*)^* \qquad \text{da } \varepsilon \in L^* \text{ für alle } L \\
&= (\{\mathtt{a}, \mathtt{b}\}^*)^0 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^1 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^2 \cup \cdots \\
&= \{\mathtt{a}, \mathtt{b}\}^* \cup (\{\mathtt{a}, \mathtt{b}\}^*)^0 \cup (\{\mathtt{a}, \mathtt{b}\}^*)^2 \cup \cdots \\
&= \{\mathtt{a}, \mathtt{b}\}^* \qquad \text{da } L^* \text{ alle Wörter über } L \text{ enthält} \\
&= \cdots \\
&= \mathcal{L}((\mathtt{a} + \mathtt{b})^*)
\end{aligned}$$

Reguläre Ausdrücke in EBNF-Notation

EBNF ... Erweiterte Backus-Naur-Form (Formalismus zur Beschreibung der Syntax von Programmiersprachen, die reguläre Ausdrücke zulässt)

AB	$A \cdot B$	Aufeinanderfolge
$A \mid B$	$A \cup B$	Alternativen
$[A]$	$\{\varepsilon\} \cup A$	Option
$\{A\}$	A^*	Wiederholung
(A)	(A)	Gruppierung
"s"	$\{s\}$	Symbol

Reguläre Ausdrücke in EBNF-Notation

EBNF ... Erweiterte Backus-Naur-Form (Formalismus zur Beschreibung der Syntax von Programmiersprachen, die reguläre Ausdrücke zulässt)

AB	$A \cdot B$	Aufeinanderfolge
$A \mid B$	$A \cup B$	Alternativen
$[A]$	$\{\varepsilon\} \cup A$	Option
$\{A\}$	A^*	Wiederholung
(A)	(A)	Gruppierung
"s"	$\{s\}$	Symbol

Reelle Numerale in EBNF-Notation

```
real  = digit { digit } "." { digit } [ scale ]
scale = "E" [ "+" | "-" ] digit { digit }
digit = "0" | "1" | "2" | ··· | "9"
```

Reguläre Ausdrücke als Syntaxdiagramme

Syntaxdiagramme . . . graphische Form der EBNF

A Abkürzung

Reguläre Ausdrücke als Syntaxdiagramme

Syntaxdiagramme ... graphische Form der EBNF

	A	Abkürzung
	$\{s\}$	Symbol

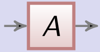
Reguläre Ausdrücke als Syntaxdiagramme

Syntaxdiagramme ... graphische Form der EBNF

	A	Abkürzung
	$\{s\}$	Symbol
	X^+	Wiederholung ≥ 1

Reguläre Ausdrücke als Syntaxdiagramme

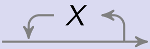
Syntaxdiagramme . . . graphische Form der EBNF



A Abkürzung

$\{s\}$ Symbol

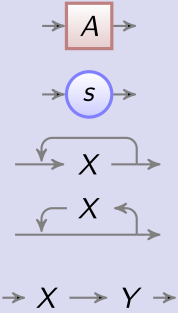
X^+ Wiederholung ≥ 1



X^* Wiederholung ≥ 0

Reguläre Ausdrücke als Syntaxdiagramme

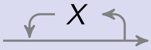
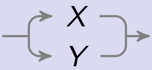
Syntaxdiagramme ... graphische Form der EBNF



A	Abkürzung
$\{s\}$	Symbol
X^+	Wiederholung ≥ 1
X^*	Wiederholung ≥ 0
$X \cdot Y$	Aufeinanderfolge

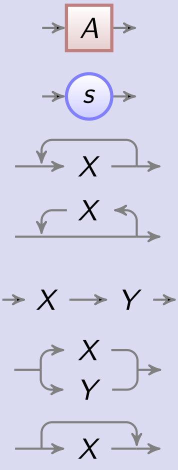
Reguläre Ausdrücke als Syntaxdiagramme

Syntaxdiagramme . . . graphische Form der EBNF

	A	Abkürzung
	$\{s\}$	Symbol
	X^+	Wiederholung ≥ 1
	X^*	Wiederholung ≥ 0
	$X \cdot Y$	Aufeinanderfolge
	$X \cup Y$	Alternativen

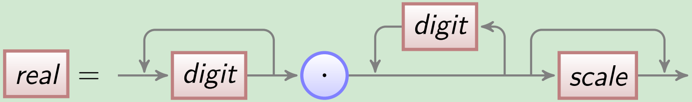
Reguläre Ausdrücke als Syntaxdiagramme

Syntaxdiagramme . . . graphische Form der EBNF

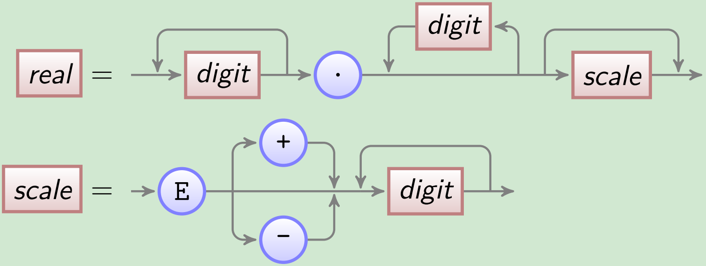


A	Abkürzung
$\{s\}$	Symbol
X^+	Wiederholung ≥ 1
X^*	Wiederholung ≥ 0
$X \cdot Y$	Aufeinanderfolge
$X \cup Y$	Alternativen
$\{\varepsilon\} \cup X$	Option

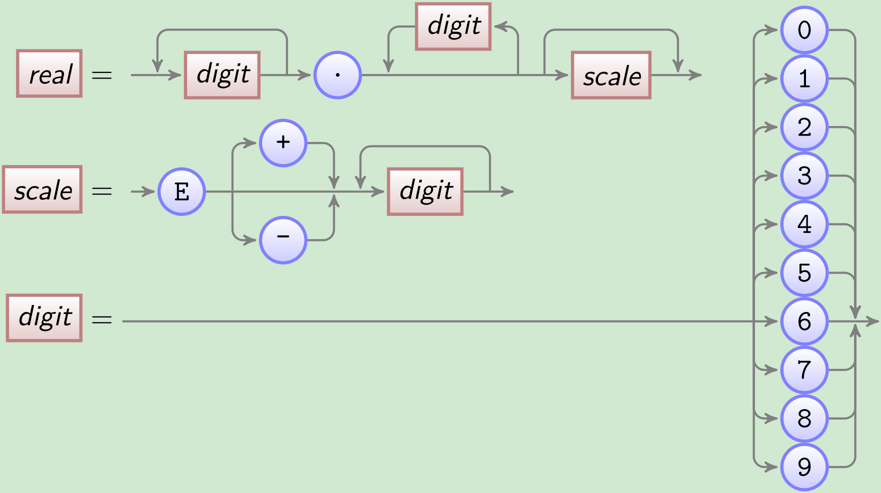
Reelle Numerale als Syntaxdiagramm



Reelle Numerale als Syntaxdiagramm



Reelle Numerale als Syntaxdiagramm



Reguläre Ausdrücke im informatischen Alltag

UNIX := *„30 definitions of regular expressions living under one roof“*

(Donald E. Knuth)

`grep -E -e "`*regexp*`"` *file*

liefert alle Zeilen der Datei *file*, die eine Zeichenkette enthalten, die dem regulären Ausdruck *regexp* (in POSIX ERE Syntax) entspricht.

Verschiedene „Standards“:

- POSIX Basic Regular Expressions
- POSIX Extended Regular Expressions
- PERL Regular Expressions
- ...

POSIX Extended Regular Expressions (ERE)

regexp	trifft zu auf
$\backslash s$	Zeichen s
s	s, falls kein Sonderzeichen
.	alle Zeichen
^	Zeilenanfang
$	Zeilenende
$[s_1 \cdots s_n]$	ein Zeichen aus $\{s_1, \ldots, s_n\}$
$[\hat{}\,s_1 \cdots s_n]$	alle Zeichen außer $s_1, \ldots, s_n$
(r)	r

POSIX Extended Regular Expressions (ERE)

regexp	trifft zu auf	*regexp*	trifft zu auf
\\s	Zeichen s	rr'	r gefolgt von r'
s	s, falls kein Sonderzeichen	$r\,\|\,r'$	r oder r'
.	alle Zeichen	$r*$	≥ 0 Mal r
ˆ	Zeilenanfang	r+	≥ 1 Mal r
\$	Zeilenende	r?	≤ 1 Mal r
[$s_1 \cdots s_n$]	ein Zeichen aus $\{s_1, \ldots, s_n\}$	r{i}	i Mal r
[ˆ$s_1 \cdots s_n$]	alle Zeichen außer $s_1, \ldots, s_n$	r{i,}	$\geq i$ Mal r
(r)	r	r{i,j}	i bis j Mal r

POSIX Extended Regular Expressions (ERE)

regexp	trifft zu auf	*regexp*	trifft zu auf
`\`s	Zeichen s	rr'	r gefolgt von r'
s	s, falls kein Sonderzeichen	$r \mid r'$	r oder r'
`.`	alle Zeichen	r`*`	≥ 0 Mal r
`^`	Zeilenanfang	r`+`	≥ 1 Mal r
`$`	Zeilenende	r`?`	≤ 1 Mal r
`[`$s_1 \cdots s_n$`]`	ein Zeichen aus $\{s_1, \ldots, s_n\}$	r`{`i`}`	i Mal r
`[^`$s_1 \cdots s_n$`]`	alle Zeichen außer $s_1, \ldots, s_n$	r`{`i`,}`	$\geq i$ Mal r
`(`r`)`	r	r`{`i`,`j`}`	i bis j Mal r

Reelle Numerale als ERE

$$digit = \{\texttt{0}, \ldots, \texttt{9}\}$$
$$scale = \{\texttt{E}\} \cdot \{\texttt{+}, \texttt{-}, \varepsilon\} \cdot digit \cdot digit^*$$
$$real = digit \cdot digit^* \cdot \{\texttt{.}\} \cdot digit^* \cdot (\{\varepsilon\} \cup scale)$$

POSIX Extended Regular Expressions (ERE)

regexp	trifft zu auf	*regexp*	trifft zu auf
`\`s	Zeichen s	rr'	r gefolgt von r'
s	s, falls kein Sonderzeichen	$r \mid r'$	r oder r'
`.`	alle Zeichen	r`*`	≥ 0 Mal r
`^`	Zeilenanfang	r`+`	≥ 1 Mal r
`$`	Zeilenende	r`?`	≤ 1 Mal r
`[`$s_1 \cdots s_n$`]`	ein Zeichen aus $\{s_1, \ldots, s_n\}$	r`{`i`}`	i Mal r
`[^`$s_1 \cdots s_n$`]`	alle Zeichen außer $s_1, \ldots, s_n$	r`{`i`,}`	$\geq i$ Mal r
`(`r`)`	r	r`{`i`,`j`}`	i bis j Mal r

Reelle Numerale als ERE

$$digit = \{\texttt{0}, \ldots, \texttt{9}\} \qquad \texttt{[0-9]}$$

$$scale = \{\texttt{E}\} \cdot \{\texttt{+}, \texttt{-}, \varepsilon\} \cdot digit \cdot digit^*$$

$$real = digit \cdot digit^* \cdot \{\texttt{.}\} \cdot digit^* \cdot (\{\varepsilon\} \cup scale)$$

`[0-9]` ... Kurzform von `[0123456789]`; analog `[a-zA-Z]` für Buchstaben

POSIX Extended Regular Expressions (ERE)

regexp	trifft zu auf	*regexp*	trifft zu auf
`\`s	Zeichen s	rr'	r gefolgt von r'
s	s, falls kein Sonderzeichen	$r \mid r'$	r oder r'
`.`	alle Zeichen	r`*`	≥ 0 Mal r
`^`	Zeilenanfang	r`+`	≥ 1 Mal r
`$`	Zeilenende	r`?`	≤ 1 Mal r
`[`$s_1 \cdots s_n$`]`	ein Zeichen aus $\{s_1, \ldots, s_n\}$	r`{`i`}`	i Mal r
`[^`$s_1 \cdots s_n$`]`	alle Zeichen außer $s_1, \ldots, s_n$	r`{`i`,}`	$\geq i$ Mal r
`(`r`)`	r	r`{`i`,`j`}`	i bis j Mal r

Reelle Numerale als ERE

$$digit = \{0, \ldots, 9\} \qquad \texttt{[0-9]}$$

$$scale = \{\texttt{E}\} \cdot \{\texttt{+}, \texttt{-}, \varepsilon\} \cdot digit \cdot digit^* \qquad \texttt{E[+-]?[0-9]+}$$

$$real = digit \cdot digit^* \cdot \{\texttt{.}\} \cdot digit^* \cdot (\{\varepsilon\} \cup scale)$$

`[0-9]` … Kurzform von `[0123456789]`; analog `[a-zA-Z]` für Buchstaben

POSIX Extended Regular Expressions (ERE)

regexp	trifft zu auf	*regexp*	trifft zu auf
`\`s	Zeichen s	rr'	r gefolgt von r'
s	s, falls kein Sonderzeichen	$r\,\|\,r'$	r oder r'
`.`	alle Zeichen	r`*`	≥ 0 Mal r
`^`	Zeilenanfang	r`+`	≥ 1 Mal r
`$`	Zeilenende	r`?`	≤ 1 Mal r
`[`$s_1 \cdots s_n$`]`	ein Zeichen aus $\{s_1, \ldots, s_n\}$	r`{`i`}`	i Mal r
`[^`$s_1 \cdots s_n$`]`	alle Zeichen außer $s_1, \ldots, s_n$	r`{`i`,}`	$\geq i$ Mal r
`(`r`)`	r	r`{`i`,`j`}`	i bis j Mal r

Reelle Numerale als ERE

$digit = \{0, \ldots, 9\}$ `[0-9]`

$scale = \{\mathtt{E}\} \cdot \{\mathtt{+}, \mathtt{-}, \varepsilon\} \cdot digit \cdot digit^*$ `E[+-]?[0-9]+`

$real = digit \cdot digit^* \cdot \{\mathtt{.}\} \cdot digit^* \cdot (\{\varepsilon\} \cup scale)$

`^[0-9]+\.[0-9]*(E[+-]?[0-9]+)?$`

`[0-9]` ... Kurzform von `[0123456789]`; analog `[a-zA-Z]` für Buchstaben

Was Sie heute erwartet

1. Organisatorisches
2. Was bedeutet Modellierung?
3. Aussagenlogik
4. Endliche Automaten
5. Reguläre Sprachen
    5.1. Operationen auf formalen Sprachen
    5.2. Definition regulärer Sprachen
    5.3. Reguläre Ausdrücke
    5.4. Eigenschaften regulärer Sprachen
    5.5. Vom regulären Ausdruck zum Automaten
    5.6. Vom Automaten zum regulären Ausdruck

Eigenschaften regulärer Sprachen

Reguläre Sprachen sind abgeschlossen gegenüber

- Vereinigung, Verkettung und Stern (warum?)

Eigenschaften regulärer Sprachen

Reguläre Sprachen sind abgeschlossen gegenüber

- Vereinigung, Verkettung und Stern (warum?)
- Durchschnitt, Komplement, Differenz, Homomorphismen, Quotientenbildung, ...

Eigenschaften regulärer Sprachen

Reguläre Sprachen sind abgeschlossen gegenüber

- Vereinigung, Verkettung und Stern (warum?)
- Durchschnitt, Komplement, Differenz, Homomorphismen, Quotientenbildung, ...

Entscheidbarkeit eines Problems: Es gibt ein Verfahren (einen Algorithmus), der für jede Eingabe die richtige Antwort ja/nein liefert.

Nicht entscheidbar: Halteproblem Ihrer Lieblingsprogrammiersprache

- Gegeben ein Programm mit einer Eingabe, wird es anhalten?

Eigenschaften regulärer Sprachen

Reguläre Sprachen sind abgeschlossen gegenüber

- Vereinigung, Verkettung und Stern (warum?)
- Durchschnitt, Komplement, Differenz, Homomorphismen, Quotientenbildung, ...

Entscheidbarkeit eines Problems: Es gibt ein Verfahren (einen Algorithmus), der für jede Eingabe die richtige Antwort ja/nein liefert.

Nicht entscheidbar: Halteproblem Ihrer Lieblingsprogrammiersprache

- Gegeben ein Programm mit einer Eingabe, wird es anhalten?

Folgende Probleme regulärer Sprachen sind entscheidbar:

- Gegeben ein Wort w und einen regulären Ausdruck r, gilt $w \in \mathcal{L}(r)$?
(Wortproblem)

Eigenschaften regulärer Sprachen

Reguläre Sprachen sind abgeschlossen gegenüber

- Vereinigung, Verkettung und Stern (warum?)
- Durchschnitt, Komplement, Differenz, Homomorphismen, Quotientenbildung, ...

Entscheidbarkeit eines Problems: Es gibt ein Verfahren (einen Algorithmus), der für jede Eingabe die richtige Antwort ja/nein liefert.

Nicht entscheidbar: Halteproblem Ihrer Lieblingsprogrammiersprache

- Gegeben ein Programm mit einer Eingabe, wird es anhalten?

Folgende Probleme regulärer Sprachen sind entscheidbar:

- Gegeben ein Wort w und einen regulären Ausdruck r, gilt $w \in \mathcal{L}(r)$? (Wortproblem)
- Gegeben zwei reguläre Ausdrücke r und r', gilt $\mathcal{L}(r) = \mathcal{L}(r')$? (Äquivalenzproblem)

Eigenschaften regulärer Sprachen

Reguläre Sprachen sind abgeschlossen gegenüber

- Vereinigung, Verkettung und Stern (warum?)
- Durchschnitt, Komplement, Differenz, Homomorphismen, Quotientenbildung, ...

Entscheidbarkeit eines Problems: Es gibt ein Verfahren (einen Algorithmus), der für jede Eingabe die richtige Antwort ja/nein liefert.

Nicht entscheidbar: Halteproblem Ihrer Lieblingsprogrammiersprache

- Gegeben ein Programm mit einer Eingabe, wird es anhalten?

Folgende Probleme regulärer Sprachen sind entscheidbar:

- Gegeben ein Wort w und einen regulären Ausdruck r, gilt $w \in \mathcal{L}(r)$? (Wortproblem)
- Gegeben zwei reguläre Ausdrücke r und r', gilt $\mathcal{L}(r) = \mathcal{L}(r')$? (Äquivalenzproblem)
- Gegeben einen regulären Ausdruck r, ist $\mathcal{L}(r)$ leer/endlich/unendlich?

Ausdruckskraft regulärer Sprachen

Die regulären Sprachen sind genau jene, die von endlichen Automaten akeptiert werden, d.h.:

- Zu jedem regulären Ausdruck r gibt es einen endlichen Automaten $\mathcal{A}$, sodass $\mathcal{L}(\mathcal{A}) = \mathcal{L}(r)$ gilt.
- Zu jedem endlichen Automaten $\mathcal{A}$ gibt es einen regulären Ausdruck r, sodass $\mathcal{L}(r) = \mathcal{L}(\mathcal{A})$ gilt.

Ausdruckskraft regulärer Sprachen

Die regulären Sprachen sind genau jene, die von endlichen Automaten akeptiert werden, d.h.:

- Zu jedem regulären Ausdruck r gibt es einen endlichen Automaten $\mathcal{A}$, sodass $\mathcal{L}(\mathcal{A}) = \mathcal{L}(r)$ gilt.
- Zu jedem endlichen Automaten $\mathcal{A}$ gibt es einen regulären Ausdruck r, sodass $\mathcal{L}(r) = \mathcal{L}(\mathcal{A})$ gilt.

Nicht regulär sind Sprachen, deren Analyse ein unbegrenztes Gedächtnis erfordert:

- Klammerausdrücke: $\{\texttt{()},\ \texttt{(())},\ \texttt{()()},\ \texttt{((()))},\ \texttt{(())()},\ \texttt{()()()},\ \ldots\}$

Ausdruckskraft regulärer Sprachen

Die regulären Sprachen sind genau jene, die von endlichen Automaten akeptiert werden, d.h.:

- Zu jedem regulären Ausdruck r gibt es einen endlichen Automaten $\mathcal{A}$, sodass $\mathcal{L}(\mathcal{A}) = \mathcal{L}(r)$ gilt.
- Zu jedem endlichen Automaten $\mathcal{A}$ gibt es einen regulären Ausdruck r, sodass $\mathcal{L}(r) = \mathcal{L}(\mathcal{A})$ gilt.

Nicht regulär sind Sprachen, deren Analyse ein unbegrenztes Gedächtnis erfordert:

- Klammerausdrücke: $\{\texttt{()}, \texttt{(())}, \texttt{()()}, \texttt{((()))}, \texttt{(())()}, \texttt{()()()}, \ldots\}$
- $\{\, \texttt{a}^n\texttt{b}^n \mid n \geq 0 \,\} = \{\varepsilon, \texttt{ab}, \texttt{aabb}, \texttt{aaabbb}, \ldots\}$
- $\{\, \texttt{a}^n\texttt{b}^n\texttt{c}^n \mid n \geq 0 \,\} = \{\varepsilon, \texttt{abc}, \texttt{aabbcc}, \texttt{aaabbbccc}, \ldots\}$

Ausdruckskraft regulärer Sprachen

Die regulären Sprachen sind genau jene, die von endlichen Automaten akeptiert werden, d.h.:

- Zu jedem regulären Ausdruck r gibt es einen endlichen Automaten $\mathcal{A}$, sodass $\mathcal{L}(\mathcal{A}) = \mathcal{L}(r)$ gilt.
- Zu jedem endlichen Automaten $\mathcal{A}$ gibt es einen regulären Ausdruck r, sodass $\mathcal{L}(r) = \mathcal{L}(\mathcal{A})$ gilt.

Nicht regulär sind Sprachen, deren Analyse ein unbegrenztes Gedächtnis erfordert:

- Klammerausdrücke: $\{\texttt{()}, \texttt{(())}, \texttt{()()}, \texttt{((()))}, \texttt{(())()}, \texttt{()()()}, \dots\}$
- $\{\, \texttt{a}^n\texttt{b}^n \mid n \geq 0 \,\} = \{\varepsilon, \texttt{ab}, \texttt{aabb}, \texttt{aaabbb}, \dots\}$
- $\{\, \texttt{a}^n\texttt{b}^n\texttt{c}^n \mid n \geq 0 \,\} = \{\varepsilon, \texttt{abc}, \texttt{aabbcc}, \texttt{aaabbbccc}, \dots\}$
- Palindrome: Wörter, die identisch mit ihrem Spiegelbild sind. $\{\texttt{otto}, \texttt{anna}, \texttt{reliefpfeiler}, \texttt{o genie der herr ehre dein ego}, \dots\}$

Ausdruckskraft regulärer Sprachen

Die regulären Sprachen sind genau jene, die von endlichen Automaten akeptiert werden, d.h.:

- Zu jedem regulären Ausdruck r gibt es einen endlichen Automaten $\mathcal{A}$, sodass $\mathcal{L}(\mathcal{A}) = \mathcal{L}(r)$ gilt.
- Zu jedem endlichen Automaten $\mathcal{A}$ gibt es einen regulären Ausdruck r, sodass $\mathcal{L}(r) = \mathcal{L}(\mathcal{A})$ gilt.

Nicht regulär sind Sprachen, deren Analyse ein unbegrenztes Gedächtnis erfordert:

- Klammerausdrücke: $\{\texttt{()}, \texttt{(())}, \texttt{()()}, \texttt{((()))}, \texttt{(())()}, \texttt{()()()}, \dots\}$
- $\{\, \mathtt{a}^n\mathtt{b}^n \mid n \geq 0 \,\} = \{\varepsilon, \mathtt{ab}, \mathtt{aabb}, \mathtt{aaabbb}, \dots\}$
- $\{\, \mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \mid n \geq 0 \,\} = \{\varepsilon, \mathtt{abc}, \mathtt{aabbcc}, \mathtt{aaabbbccc}, \dots\}$
- Palindrome: Wörter, die identisch mit ihrem Spiegelbild sind.
  $\{\mathtt{otto}, \mathtt{anna}, \mathtt{reliefpfeiler}, \ \mathtt{o\,genie\,der\,herr\,ehre\,dein\,ego}, \dots\}$
- Doppelwörter: $\{\, ww \mid w \in \Sigma^* \,\}$ (falls $|\Sigma| > 1$)

Was Sie heute erwartet

1. Organisatorisches
2. Was bedeutet Modellierung?
3. Aussagenlogik
4. Endliche Automaten
5. Reguläre Sprachen
   - 5.1. Operationen auf formalen Sprachen
   - 5.2. Definition regulärer Sprachen
   - 5.3. Reguläre Ausdrücke
   - 5.4. Eigenschaften regulärer Sprachen
   - 5.5. Vom regulären Ausdruck zum Automaten
   - 5.6. Vom Automaten zum regulären Ausdruck

Vom regulären Ausdruck zum Automaten

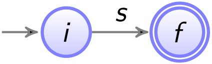
Vom regulären Ausdruck zum Automaten

Automat für $\emptyset$:

Vom regulären Ausdruck zum Automaten

Automat für $\emptyset$:

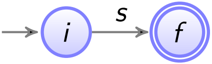
Automat für $s \in \Sigma \cup \{\varepsilon\}$:



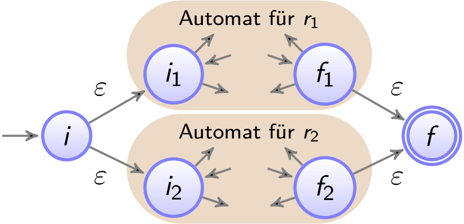
Vom regulären Ausdruck zum Automaten

Automat für $\emptyset$:

Automat für $s \in \Sigma \cup \{\varepsilon\}$:



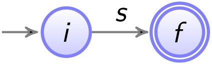
Automat für $r_1 + r_2$:



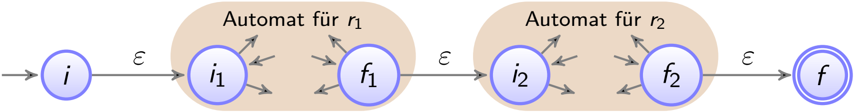
Vom regulären Ausdruck zum Automaten

Automat für $\emptyset$:

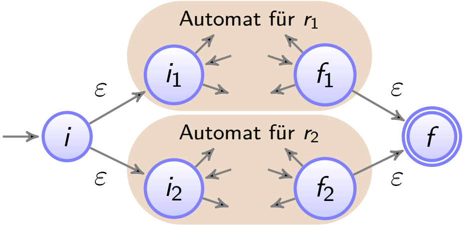
Automat für $s \in \Sigma \cup \{\varepsilon\}$:



Automat für $r_1 r_2$:



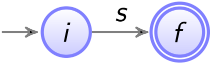
Automat für $r_1 + r_2$:



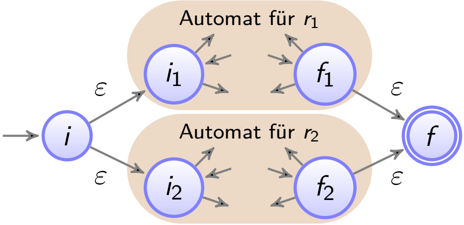
Vom regulären Ausdruck zum Automaten

Automat für $\emptyset$:

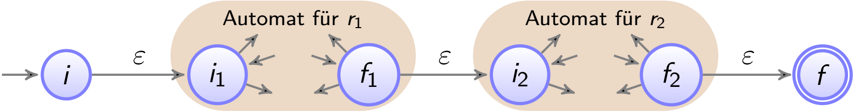
Automat für $s \in \Sigma \cup \{\varepsilon\}$:



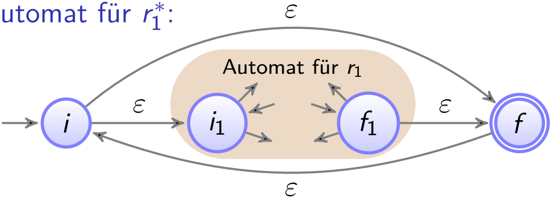
Automat für $r_1 + r_2$:



Automat für $r_1 r_2$:



Automat für r_1^*:

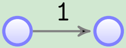


Automat für Binärnumerale ohne führende Null

Regulärer Ausdruck: $1(0+1)^*$

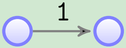
Automat für Binärnumerale ohne führende Null

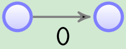
Regulärer Ausdruck: $1(0 + 1)^*$



Automat für Binärnumerale ohne führende Null

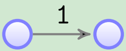
Regulärer Ausdruck: $1(0 + 1)^*$

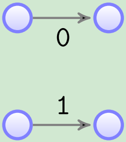




Automat für Binärnumerale ohne führende Null

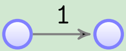
Regulärer Ausdruck: $1(0+1)^*$

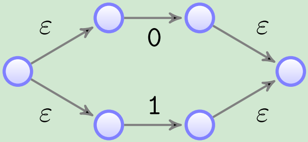




Automat für Binärnumerale ohne führende Null

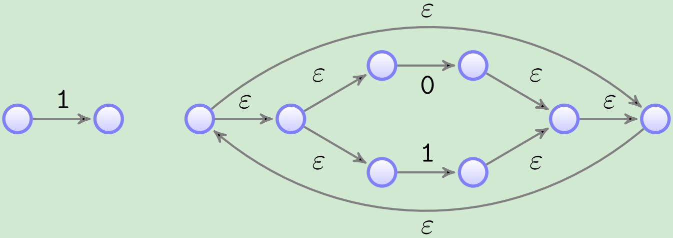
Regulärer Ausdruck: $1(0 + 1)^*$





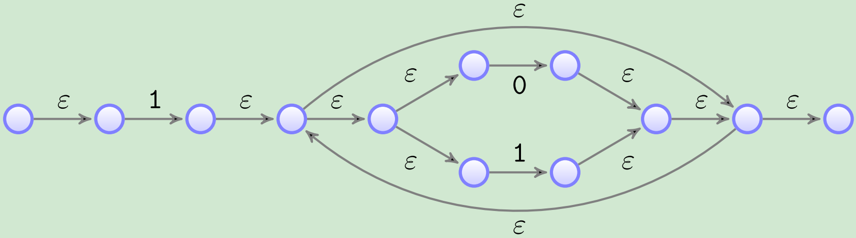
Automat für Binärnumerale ohne führende Null

Regulärer Ausdruck: $1(0+1)^{*}$



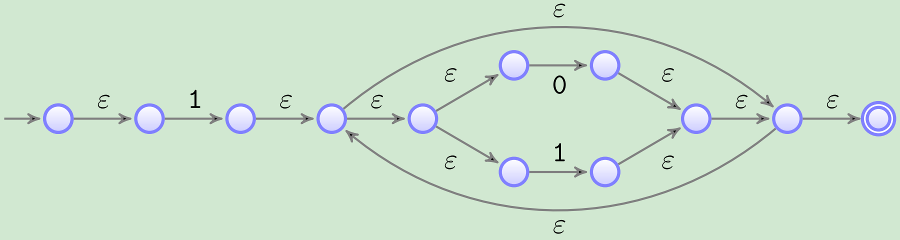
Automat für Binärnumerale ohne führende Null

Regulärer Ausdruck: $1(0+1)^*$



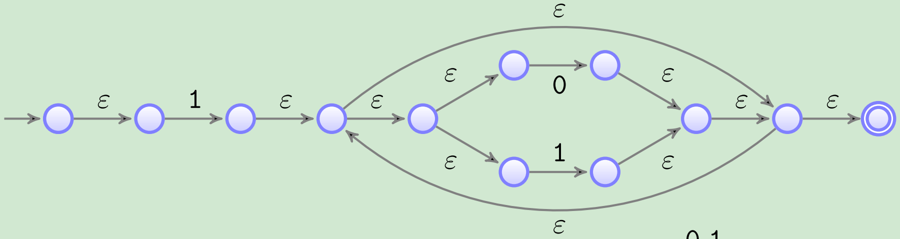
Automat für Binärnumerale ohne führende Null

Regulärer Ausdruck: $\mathtt{1}(\mathtt{0}+\mathtt{1})^*$

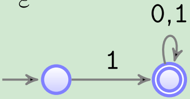


Automat für Binärnumerale ohne führende Null

Regulärer Ausdruck: $1(0+1)^*$

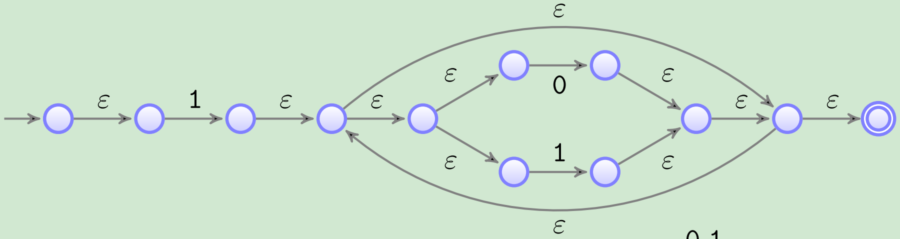


Minimaler deterministischer Automat:

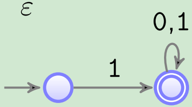


Automat für Binärnumerale ohne führende Null

Regulärer Ausdruck: $1(0+1)^*$



Minimaler deterministischer Automat:



Manuelle Konstruktion von Automaten:

1. Konstruiere die Automaten zu einfachen Teilsprachen „durch Hinschauen".
2. Verwende die allgemeine Konstruktion mit ε-Übergängen für undurchsichtige Situationen.

Was Sie heute erwartet

1. Organisatorisches
2. Was bedeutet Modellierung?
3. Aussagenlogik
4. Endliche Automaten
5. Reguläre Sprachen
    5.1. Operationen auf formalen Sprachen
    5.2. Definition regulärer Sprachen
    5.3. Reguläre Ausdrücke
    5.4. Eigenschaften regulärer Sprachen
    5.5. Vom regulären Ausdruck zum Automaten
    5.6. Vom Automaten zum regulären Ausdruck

Vom Automaten zum regulären Ausdruck

R ... Menge der regulären Ausdrücke über Σ

Verallgemeinerter endlicher Automat

... wird beschrieben durch ein 5-Tupel $\mathcal{A} = \langle Q, \Sigma, \delta, i, f \rangle$, wobei

- Q ... endliche Zustandsmenge
- Σ ... Eingabealphabet
- $\delta\colon (Q - \{f\}) \times (Q - \{i\}) \mapsto R$ ... Übergangsfunktion
- $i \in Q$ ... Anfangszustand
- $f \in Q$, $f \neq i$ ... Endzustand

Vom Automaten zum regulären Ausdruck

R ... Menge der regulären Ausdrücke über Σ

Verallgemeinerter endlicher Automat

... wird beschrieben durch ein 5-Tupel $\mathcal{A} = \langle Q, \Sigma, \delta, i, f \rangle$, wobei

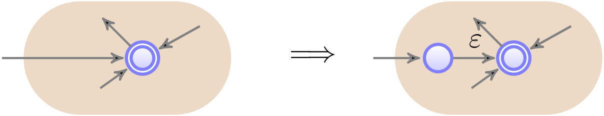
- Q ... endliche Zustandsmenge
- Σ ... Eingabealphabet
- $\delta\colon (Q - \{f\}) \times (Q - \{i\}) \mapsto R$ ... Übergangsfunktion
- $i \in Q$ ... Anfangszustand
- $f \in Q$, $f \neq i$ ... Endzustand

Unterschiede zu „normalen“ Automaten:

- keine Übergänge in den Anfangszustand;
- nur ein Endzustand, der nicht Anfangszustand ist;
- keine Übergänge weg vom Endzustand;
- nur ein Übergang zwischen je zwei Zuständen;
- Übergänge beschriftet mit regulären Ausdrücken.

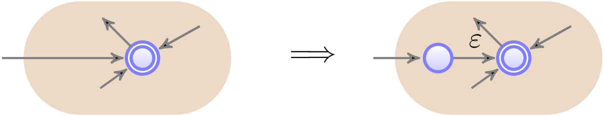
Umwandlung eines endlichen Automaten in einen verallgemeinerten

- Übergänge in den Anfangszustand oder Anfangszustand ist Endzustand:

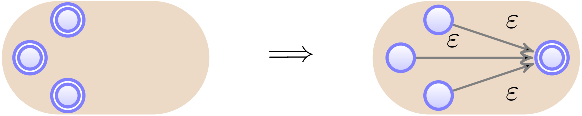


Umwandlung eines endlichen Automaten in einen verallgemeinerten

- Übergänge in den Anfangszustand oder Anfangszustand ist Endzustand:

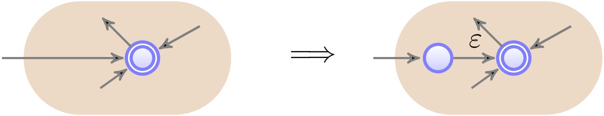


- Mehrere Endzustände:

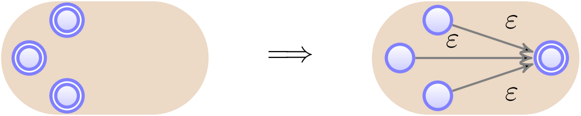


Umwandlung eines endlichen Automaten in einen verallgemeinerten

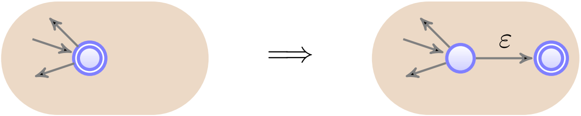
- Übergänge in den Anfangszustand oder Anfangszustand ist Endzustand:



- Mehrere Endzustände:

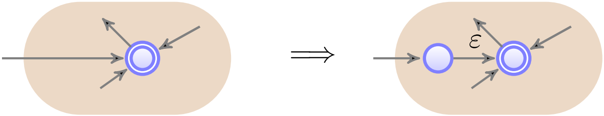


- Übergänge weg vom Endzustand:

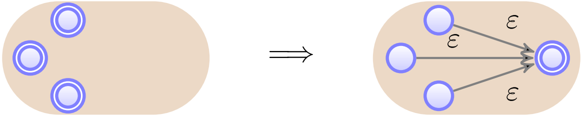


Umwandlung eines endlichen Automaten in einen verallgemeinerten

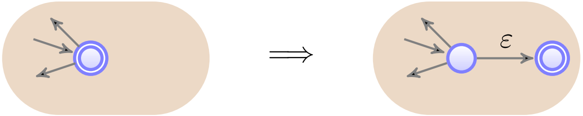
- Übergänge in den Anfangszustand oder Anfangszustand ist Endzustand:



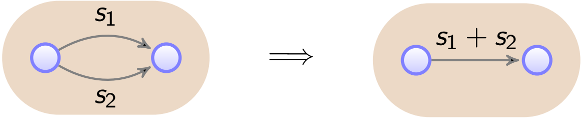
- Mehrere Endzustände:



- Übergänge weg vom Endzustand:



- Mehrere Übergänge zwischen zwei Zuständen:



Vom verallgemeinerten Automaten zum regulären Ausdruck

Gegeben: Verallgemeinerter Automat $\mathcal{A} = \langle Q, \Sigma, \delta, i, f \rangle$

Vom verallgemeinerten Automaten zum regulären Ausdruck

Gegeben: Verallgemeinerter Automat $\mathcal{A} = \langle Q, \Sigma, \delta, i, f \rangle$

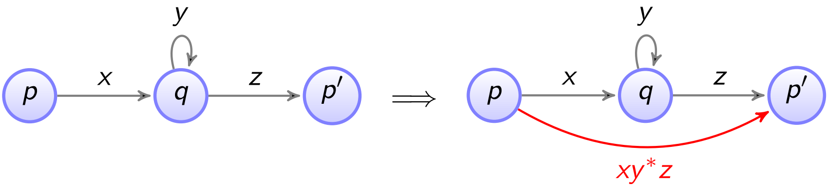
Für jeden Zustand $q \in Q - \{i, f\}$ führe folgende Schritte durch:

Vom verallgemeinerten Automaten zum regulären Ausdruck

Gegeben: Verallgemeinerter Automat $\mathcal{A} = \langle Q, \Sigma, \delta, i, f \rangle$

Für jeden Zustand $q \in Q - \{i, f\}$ führe folgende Schritte durch:

1. Füge zwischen allen Nachbarn p, p' von q neue Übergänge hinzu:



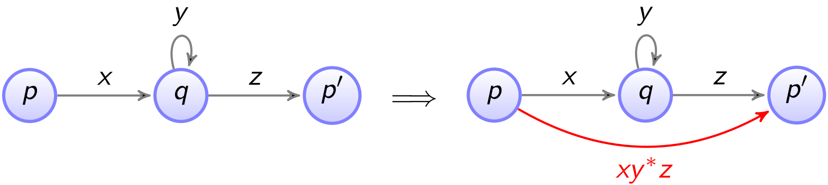
(x, y und z bezeichnen reguläre Ausdrücke.)

Vom verallgemeinerten Automaten zum regulären Ausdruck

Gegeben: Verallgemeinerter Automat $\mathcal{A} = \langle Q, \Sigma, \delta, i, f \rangle$

Für jeden Zustand $q \in Q - \{i, f\}$ führe folgende Schritte durch:

1. Füge zwischen allen Nachbarn p, p' von q neue Übergänge hinzu:



   (x, y und z bezeichnen reguläre Ausdrücke.)

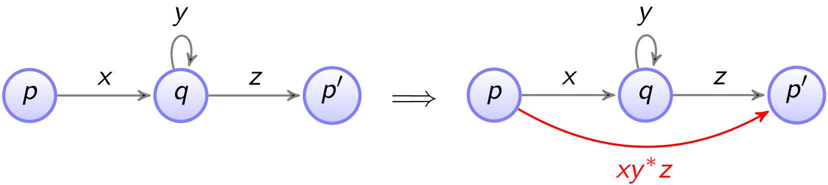
2. Entferne q und alle Kanten von und nach q aus dem Automaten.

Vom verallgemeinerten Automaten zum regulären Ausdruck

Gegeben: Verallgemeinerter Automat $\mathcal{A} = \langle Q, \Sigma, \delta, i, f \rangle$

Für jeden Zustand $q \in Q - \{i, f\}$ führe folgende Schritte durch:

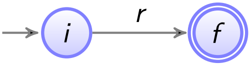
1. Füge zwischen allen Nachbarn p, p' von q neue Übergänge hinzu:



   (x, y und z bezeichnen reguläre Ausdrücke.)

2. Entferne q und alle Kanten von und nach q aus dem Automaten.

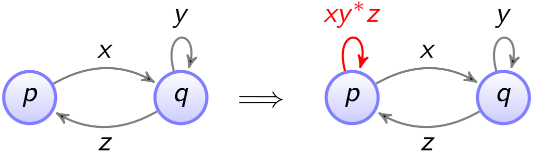
Restautomat:



Ergebnis: r ist ein regulärer Ausdruck mit $\mathcal{L}(r) = \mathcal{L}(\mathcal{A})$.

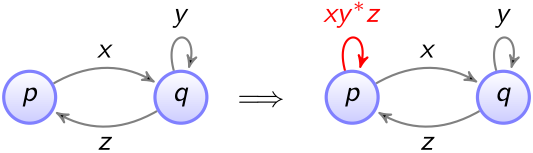
Anmerkungen:

- Falls p und p' derselbe Knoten sind, erhält man:



Anmerkungen:

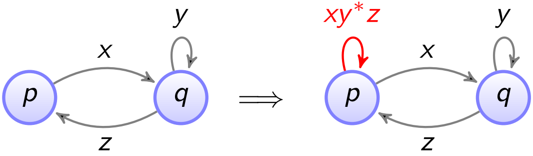
- Falls p und p' derselbe Knoten sind, erhält man:



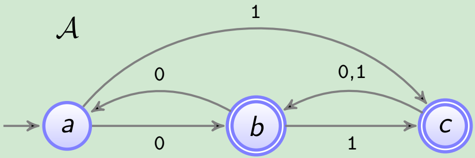
- Falls die Schleife mit dem Ausdruck y nicht existiert, entfällt y^*.

Anmerkungen:

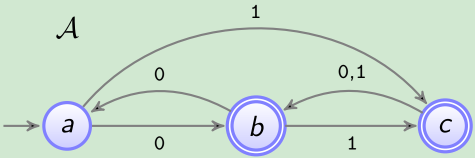
- Falls p und p' derselbe Knoten sind, erhält man:

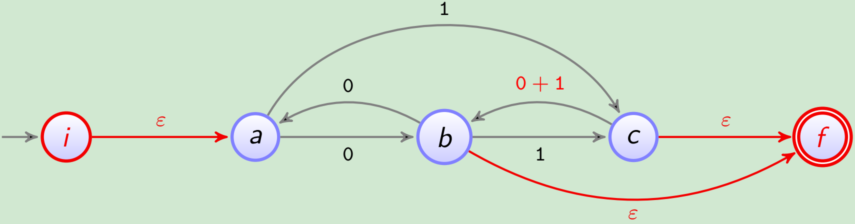


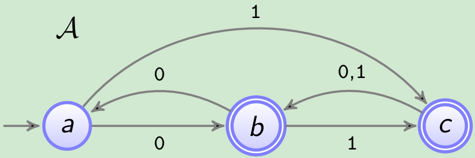
- Falls die Schleife mit dem Ausdruck y nicht existiert, entfällt y^*.
- Falls der Übergang p–p' bereits existiert, wird xy^*z addiert.

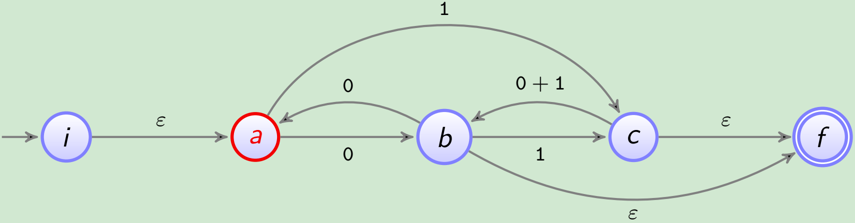


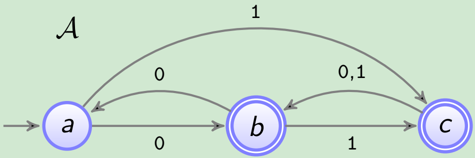
$\mathcal{A}$
1
0
0,1
a
b
c
0
1


$\mathcal{A}$
1
0
0,1
a
b
c
0
1

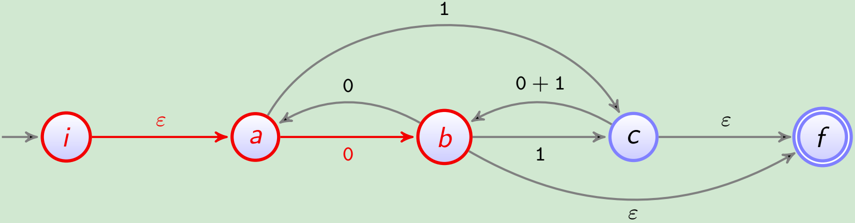

1
0
0 + 1
i
ε
a
b
c
ε
f
0
1
ε


$\mathcal{A}$
1
0
0,1
a
b
c
0
1

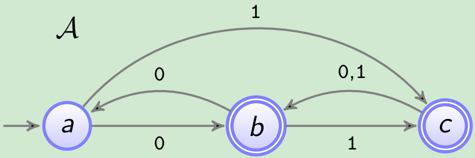

1
0
0 + 1
i
ε
a
b
c
ε
f
0
1
ε

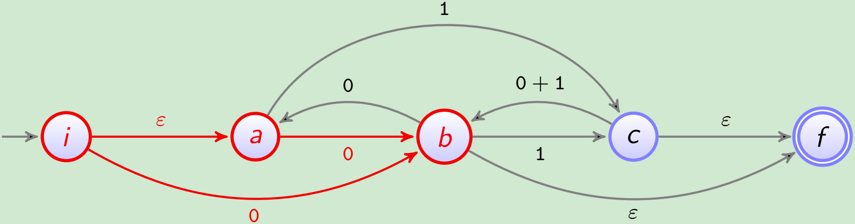


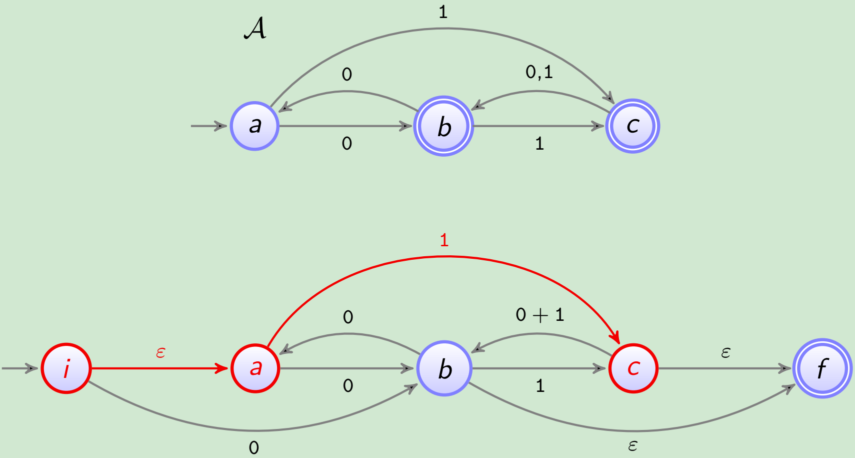
$\mathcal{A}$
1
0
0,1
a
b
c
0
1



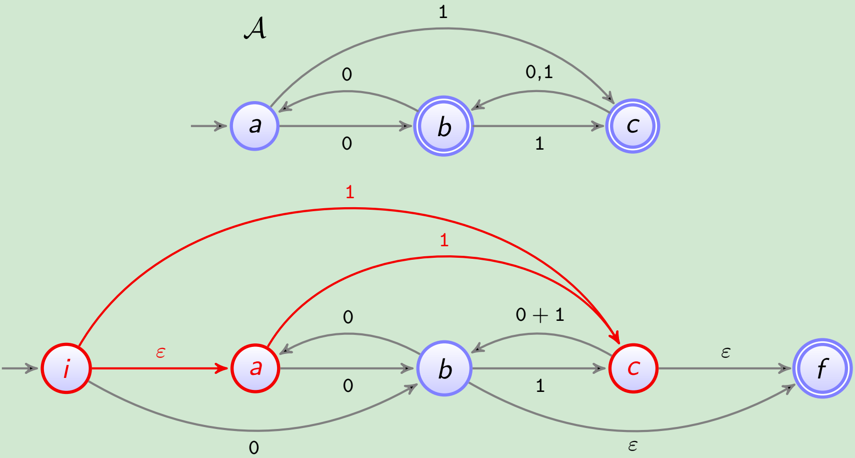
1
0
0 + 1
ε
ε
i
a
b
c
f
0
1
ε


$\mathcal{A}$
1
0
0,1
a
b
c
0
1

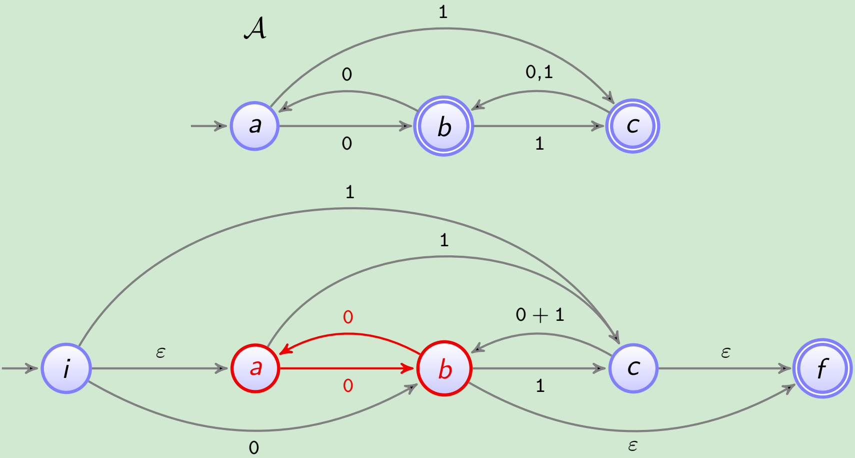

1
0
0 + 1
ε
ε
i
a
b
c
f
0
1
0
ε

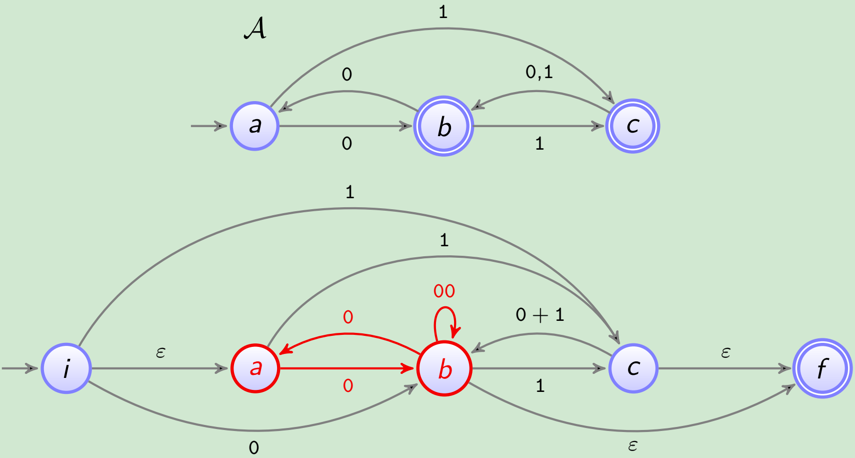


$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
0
0 + 1
i
ε
a
b
c
ε
f
0
1
0
ε

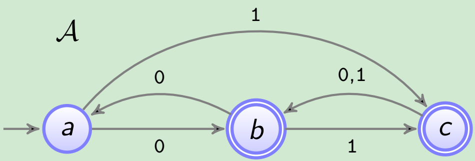


$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
1
0
0 + 1
i
ε
a
b
c
ε
f
0
1
0
ε

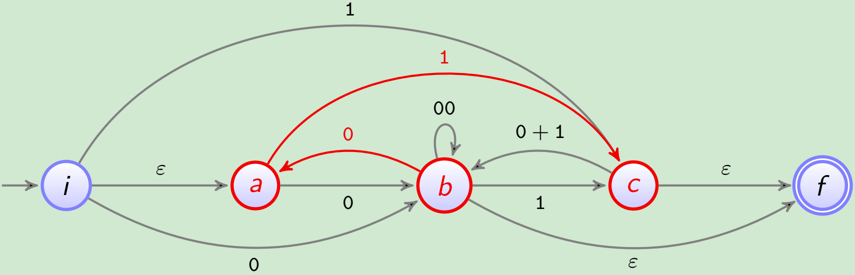

$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
1
0
0 + 1
i
ε
a
b
c
ε
f
0
1
0
ε



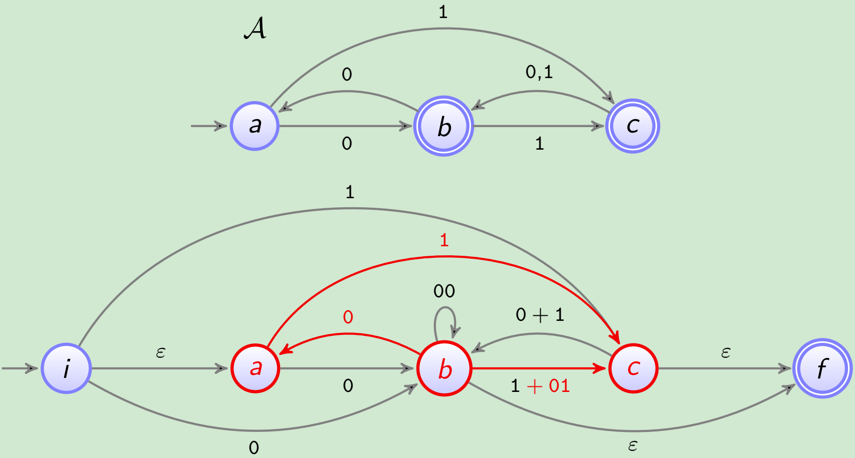
$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
1
00
0
0 + 1
ε
ε
i
a
b
c
f
0
1
0
ε



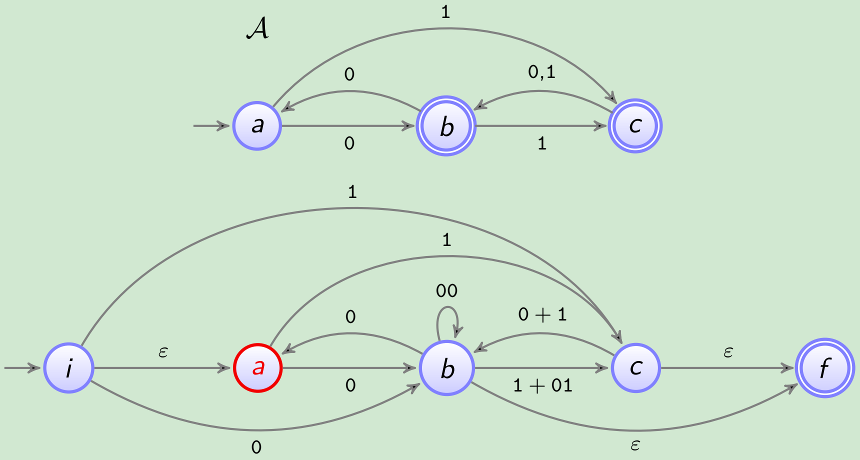
𝒜
1
0
0,1
a
b
c
0
1



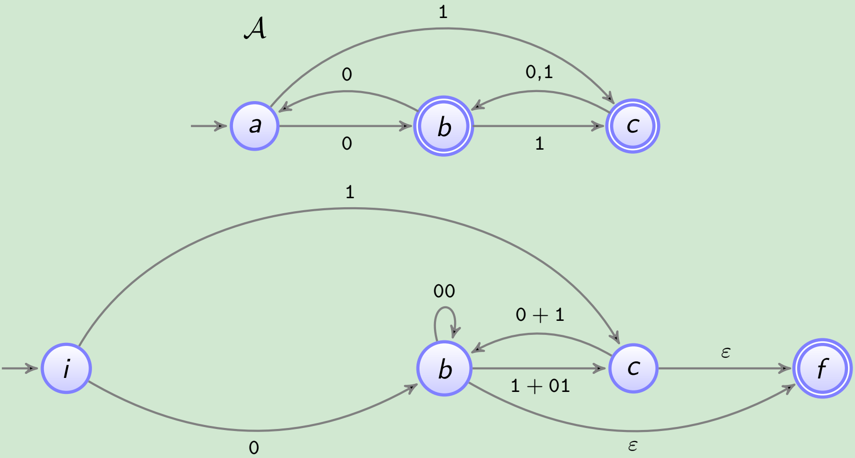
1
1
00
0
0 + 1
ε
ε
i
a
b
c
f
0
1
0
ε



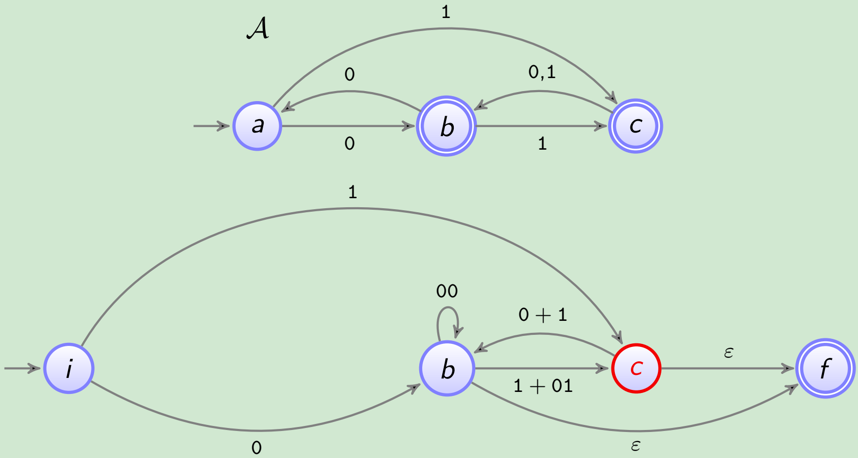
$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
1
00
0
0 + 1
ε
ε
i
a
b
c
f
0
1 + 01
0
ε



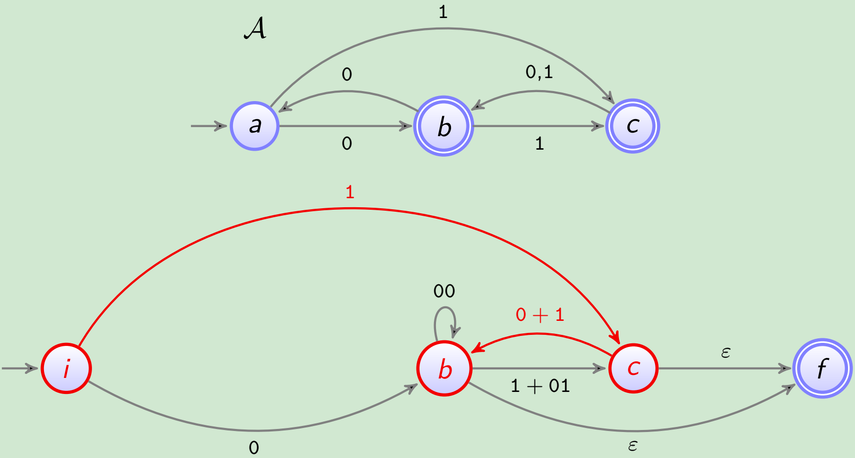
$\mathcal{A}$
a
b
c
0
0
0,1
1
1
i
a
b
c
f
ε
0
0
0
1
1
00
0 + 1
1 + 01
ε
ε

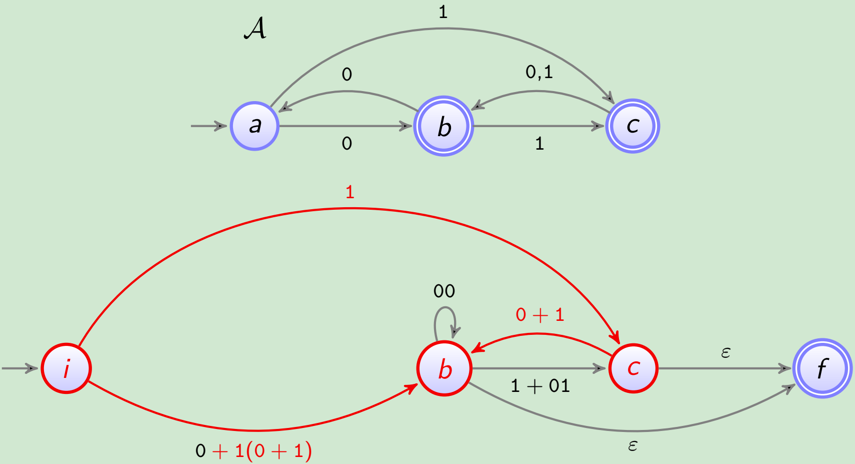


$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
00
0 + 1
i
b
c
f
ε
1 + 01
0
ε

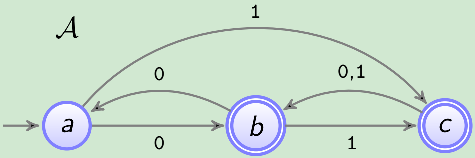


$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
00
0 + 1
i
b
c
f
ε
1 + 01
0
ε

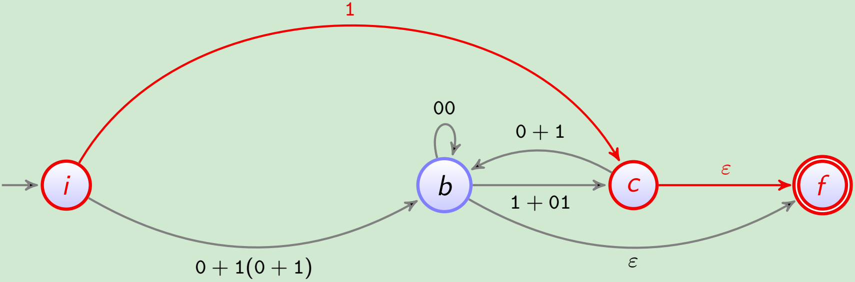

$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
00
0 + 1
i
b
c
f
ε
1 + 01
0
ε



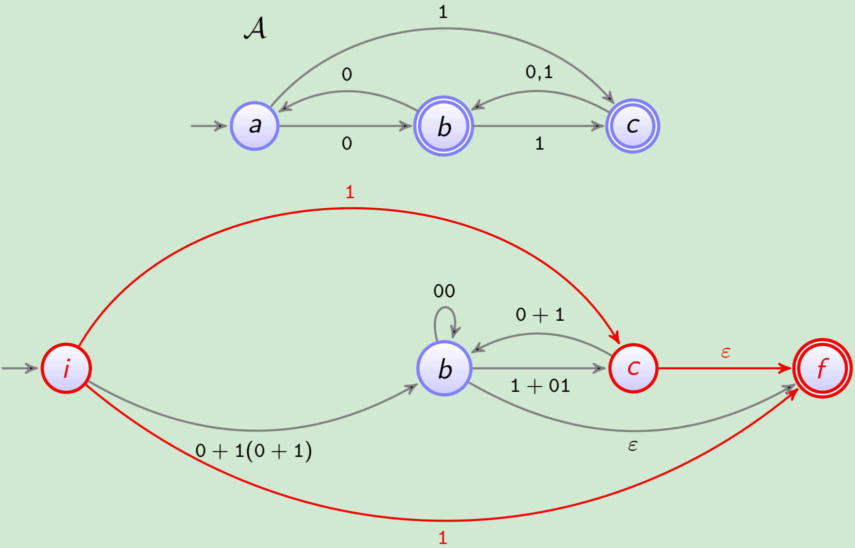
$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
00
0 + 1
i
b
c
f
ε
1 + 01
0 + 1(0 + 1)
ε



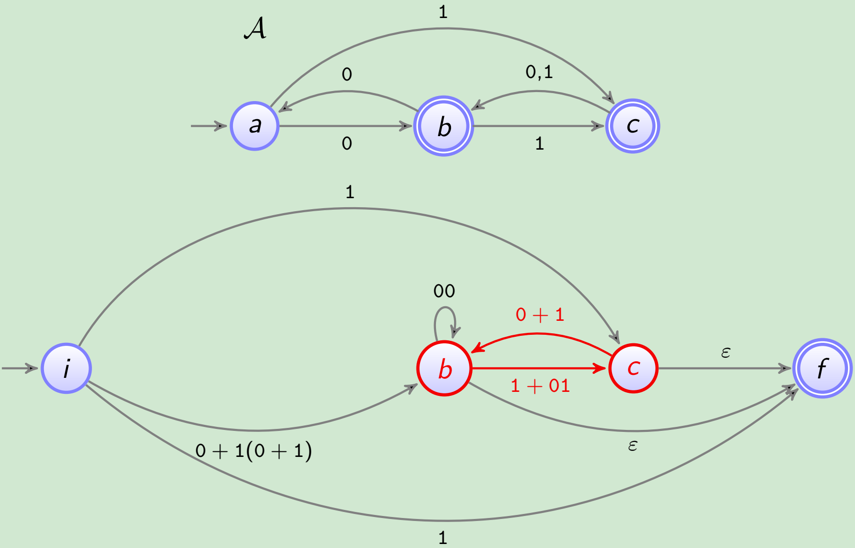
$\mathcal{A}$
1
0
0,1
a
b
c
0
1

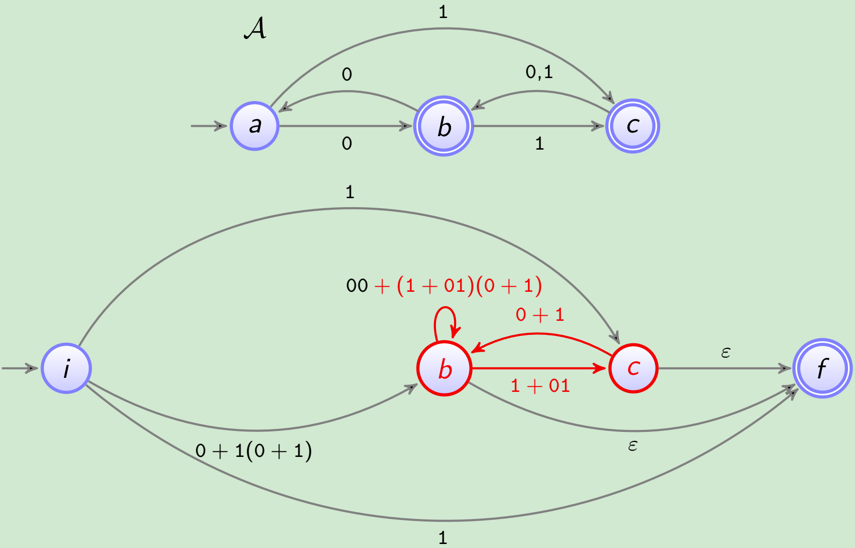


1
00
0 + 1
i
b
c
f
ε
1 + 01
0 + 1(0 + 1)
ε

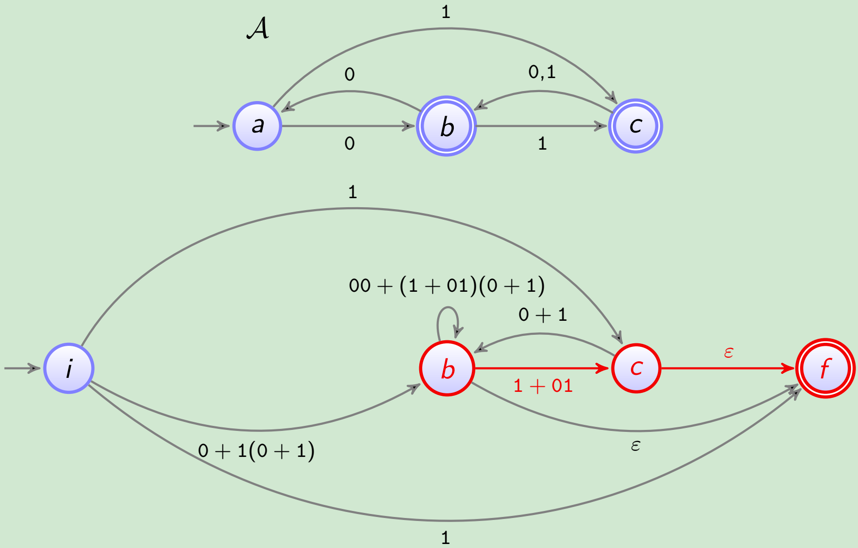


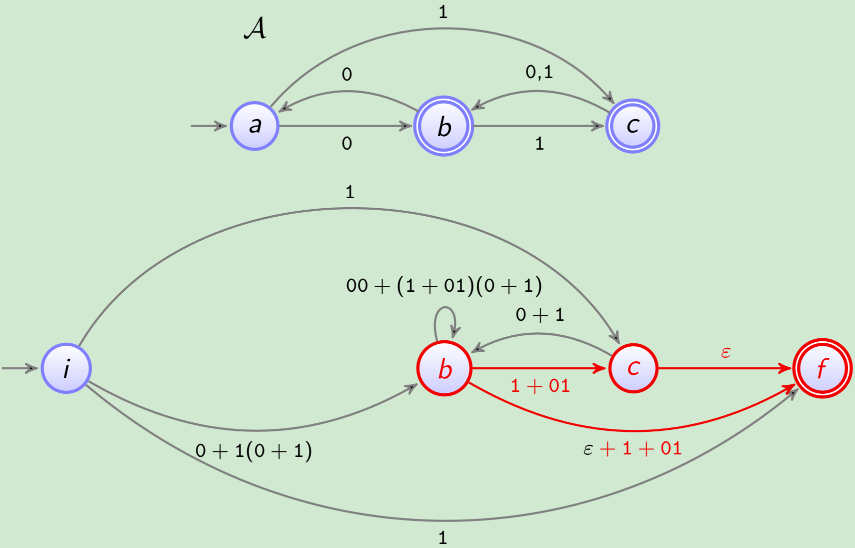
𝒜
1
0
0,1
a
b
c
0
1
1
00
0 + 1
i
b
c
ε
f
1 + 01
ε
0 + 1(0 + 1)
1


𝒜
a
b
c
0
0
0,1
1
1
i
b
c
f
1
00
0 + 1
1 + 01
ε
ε
0 + 1(0 + 1)
1

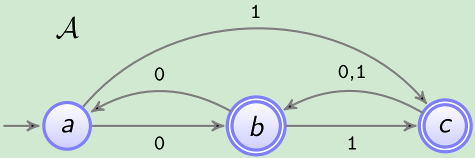


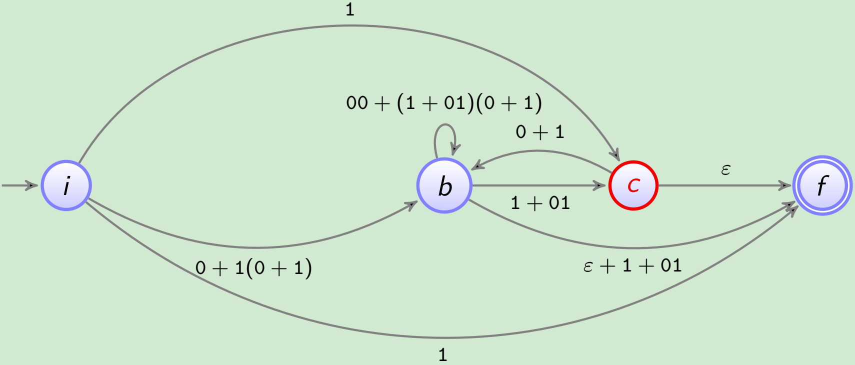
$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
$00 + (1 + 01)(0 + 1)$
$0 + 1$
i
b
c
f
ε
$1 + 01$
ε
$0 + 1(0 + 1)$
1

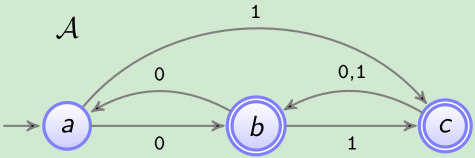

$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
00 + (1 + 01)(0 + 1)
0 + 1
i
b
c
f
ε
1 + 01
ε
0 + 1(0 + 1)
1

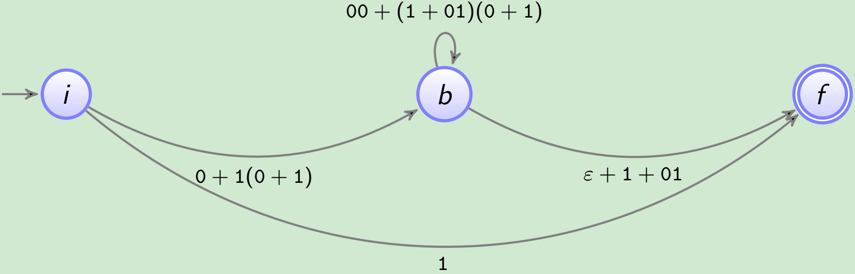


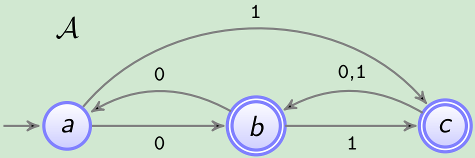
$\mathcal{A}$
1
0
0,1
a
b
c
0
1
1
00 + (1 + 01)(0 + 1)
0 + 1
i
b
c
f
ε
1 + 01
0 + 1(0 + 1)
ε + 1 + 01
1

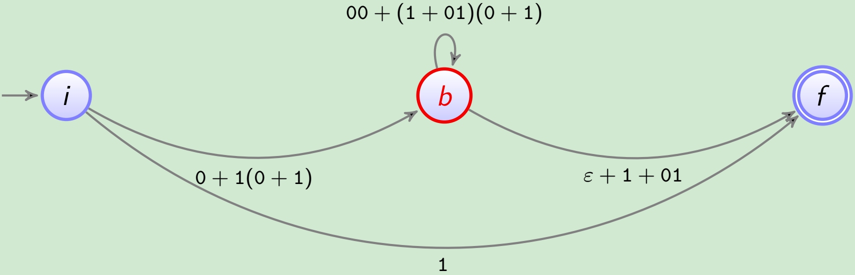

$\mathcal{A}$
1
0
0,1
a
b
c
0
1

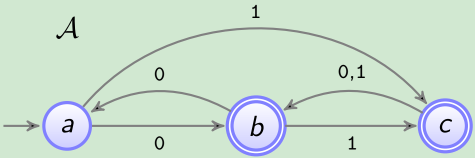

1
00 + (1 + 01)(0 + 1)
0 + 1
i
b
c
ε
f
1 + 01
0 + 1(0 + 1)
ε + 1 + 01
1


$\mathcal{A}$
1
0
0,1
a
b
c
0
1

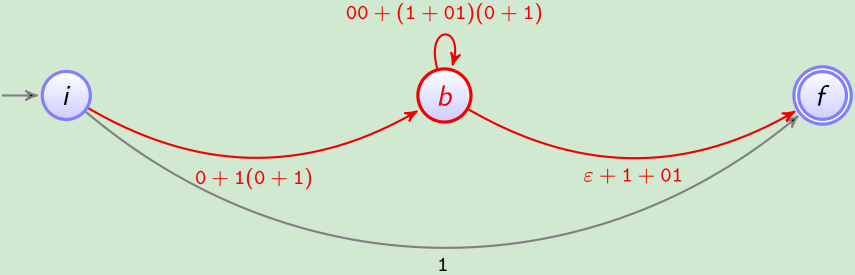

$00 + (1 + 01)(0 + 1)$
i
b
f
$0 + 1(0 + 1)$
$\varepsilon + 1 + 01$
1


$\mathcal{A}$
1
0
0,1
a
b
c
0
1

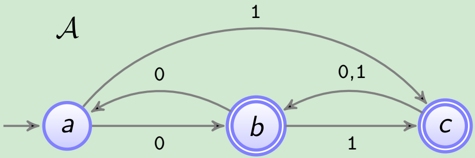

00 + (1 + 01)(0 + 1)
i
b
f
0 + 1(0 + 1)
ε + 1 + 01
1

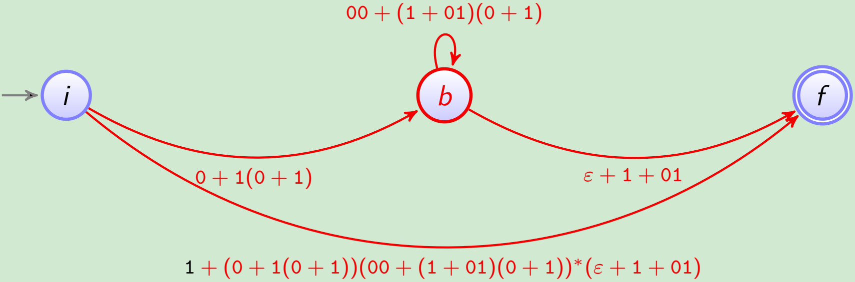


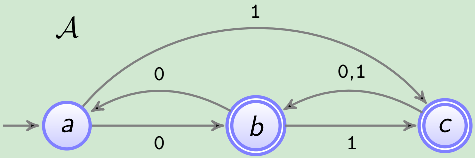
$\mathcal{A}$
1
0
0,1
a
b
c
0
1

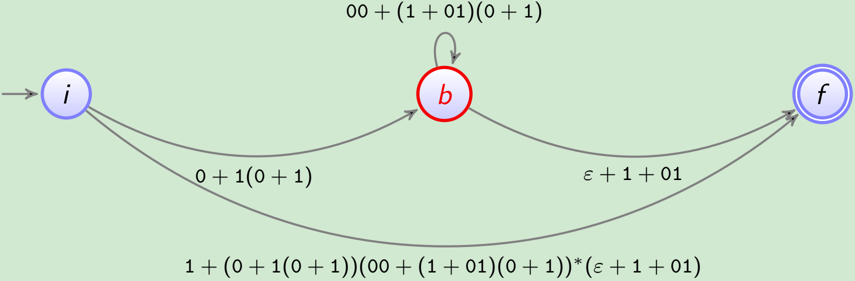


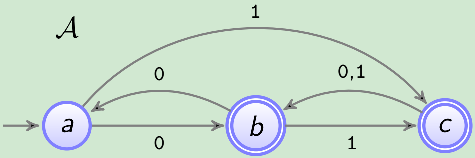
$00 + (1 + 01)(0 + 1)$
i
b
f
$0 + 1(0 + 1)$
$\varepsilon + 1 + 01$
1


$\mathcal{A}$
1
0
0,1
a
b
c
0
1

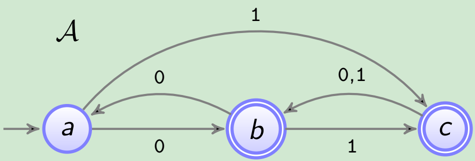

$00 + (1 + 01)(0 + 1)$
i
b
f
$0 + 1(0 + 1)$
$\varepsilon + 1 + 01$
$1 + (0 + 1(0 + 1))(00 + (1 + 01)(0 + 1))^*(\varepsilon + 1 + 01)$


$\mathcal{A}$
1
0
0,1
a
b
c
0
1


00 + (1 + 01)(0 + 1)
i
b
f
0 + 1(0 + 1)
ε + 1 + 01
1 + (0 + 1(0 + 1))(00 + (1 + 01)(0 + 1))*(ε + 1 + 01)



$$1 + (0 + 1(0 + 1))(00 + (1 + 01)(0 + 1))^*(\varepsilon + 1 + 01)$$



$$1 + (0 + 1(0 + 1))(00 + (1 + 01)(0 + 1))^*(\varepsilon + 1 + 01)$$

$r = 1 + (0 + 10 + 11)(00 + 10 + 11 + 010 + 011)^*(\varepsilon + 1 + 01)$

Es gilt: $\mathcal{L}(r) = \mathcal{L}(\mathcal{A})$.