Teil VI

Reguläre Sprachen und endliche Automaten
Teil 6: Mehr Nichtdeterminismus

ε-NFAs und Abschlusseigenschaften

Lemma

Die Klasse der regulären Sprachen ist abgeschlossen unter den Operationen

- Vereinigung,
- Konkatenation,
- Kleene-Stern.

ε-NFAs und Abschlusseigenschaften

Lemma

Die Klasse der regulären Sprachen ist abgeschlossen unter den Operationen

- Vereinigung,
- Konkatenation,
- Kleene-Stern.

Geschenkt:

- n-fache Konkatenation,
- Kleene-Plus.

Vereinigung

Seien A_1, A_2 NFAs

$$A_1 := (\Sigma, Q_1, \delta_1, q_{0,1}, F_1)$$
$$A_2 := (\Sigma, Q_2, \delta_2, q_{0,2}, F_2)$$

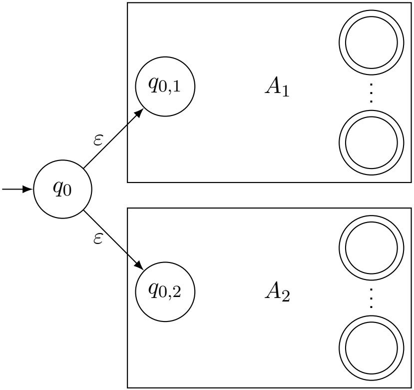
$(Q_1 \cap Q_2) = \emptyset$

Vereinigung

Seien A_1, A_2 NFAs

$$A_1 := (\Sigma, Q_1, \delta_1, q_{0,1}, F_1)$$
$$A_2 := (\Sigma, Q_2, \delta_2, q_{0,2}, F_2)$$

$(Q_1 \cap Q_2) = \emptyset$



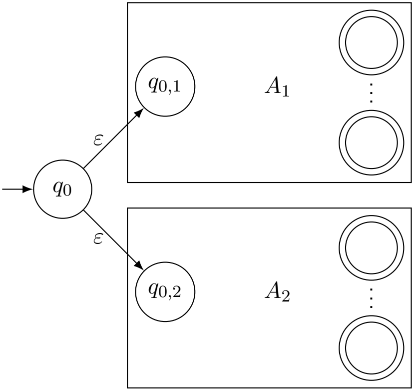
Vereinigung

Seien A_1, A_2 NFAs

$$A_1 := (\Sigma, Q_1, \delta_1, q_{0,1}, F_1)$$
$$A_2 := (\Sigma, Q_2, \delta_2, q_{0,2}, F_2)$$

$(Q_1 \cap Q_2) = \emptyset$

$$A_V := (\Sigma, Q, \delta_V, q_0, F),$$
$$Q := Q_1 \cup Q_2 \cup \{q_0\},$$
$$F := F_1 \cup F_2,$$
$$\delta_V(q,a) = \begin{cases} \delta_1(q,a) & (q \in Q_1), \\ \delta_2(q,a) & (q \in Q_2) \end{cases}$$
$$\delta_V(q_0, \varepsilon) = \{q_{0,1}, q_{0,2}\}.$$



Konkatenation

Seien A_1, A_2 NFAs

$$A_1 := (\Sigma, Q_1, \delta_1, q_{0,1}, F_1)$$
$$A_2 := (\Sigma, Q_2, \delta_2, q_{0,2}, F_2)$$

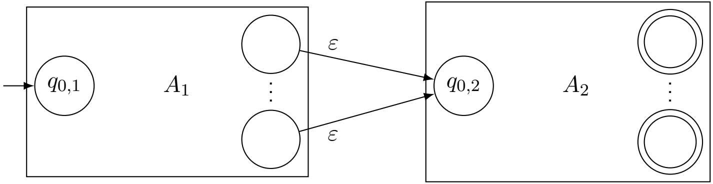
$(Q_1 \cap Q_2) = \emptyset$

Konkatenation

Seien A_1, A_2 NFAs

$$A_1 := (\Sigma, Q_1, \delta_1, q_{0,1}, F_1)$$
$$A_2 := (\Sigma, Q_2, \delta_2, q_{0,2}, F_2)$$

$(Q_1 \cap Q_2) = \emptyset$



Konkatenation

Seien A_1, A_2 NFAs

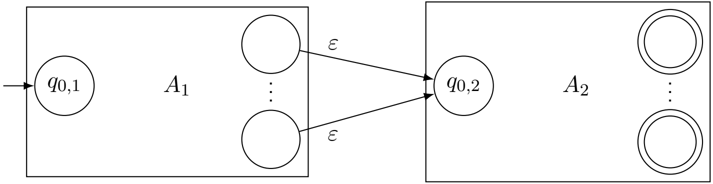
$$A_1 := (\Sigma, Q_1, \delta_1, q_{0,1}, F_1)$$
$$A_2 := (\Sigma, Q_2, \delta_2, q_{0,2}, F_2)$$

$(Q_1 \cap Q_2) = \emptyset$

$$A_K := (\Sigma, Q_1 \cup Q_2, \delta_K, q_{0,1}, F_2),$$

$$\delta_K(q,a) := \begin{cases} \delta_1(q,a) & (q \in Q_1), \\ \delta_2(q,a) & (q \in Q_2) \end{cases}$$

$$\delta_K(q,\varepsilon) := \{q_{0,2}\}$$



Kleene-Stern

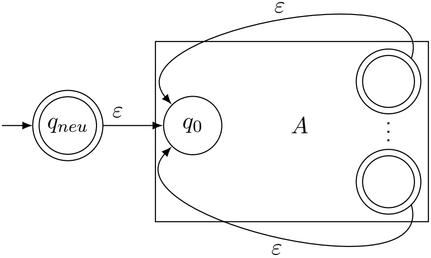
Sei A ein NFA

$$A := (\Sigma, Q, \delta, q_0, F)$$

Kleene-Stern

Sei A ein NFA

$$A := (\Sigma, Q, \delta, q_0, F)$$

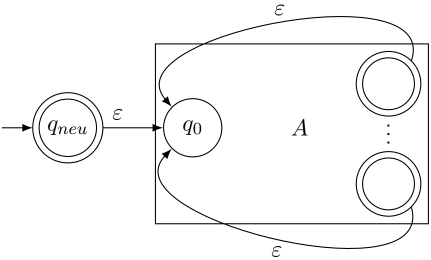


Kleene-Stern

Sei A ein NFA

$$A := (\Sigma, Q, \delta, q_0, F)$$

$$\begin{aligned} A_S &:= (\Sigma, Q_S, \delta_S, q_{neu}, F_S), \\ Q_S &:= Q \cup \{q_{neu}\}, \\ F_S &:= F \cup \{q_{neu}\}, \\ \delta_S(q, a) &:= \delta(q, a) \text{ für } q \in Q \\ \delta_S(q, \varepsilon) &:= \{q_0\} \text{für } q \in (F \cup \{q_{neu}\}) \end{aligned}$$



Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- jedes $a \in \Sigma$ ist ein regulärer Ausdruck

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$
- $\mathcal{L}(a) := \{a\}$

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- jedes $a \in \Sigma$ ist ein regulärer Ausdruck

Angenommen, α, β sind reguläre Ausdrücke. Dann auch:

- $(\alpha \cdot \beta)$

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$
- $\mathcal{L}(a) := \{a\}$
- $\mathcal{L}(\alpha \cdot \beta) := \mathcal{L}(\alpha) \cdot \mathcal{L}(\beta)$

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- jedes $a \in \Sigma$ ist ein regulärer Ausdruck

Angenommen, α, β sind reguläre Ausdrücke. Dann auch:

- $(\alpha \cdot \beta)$
- $(\alpha \mid \beta)$

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$
- $\mathcal{L}(a) := \{a\}$

- $\mathcal{L}(\alpha \cdot \beta) := \mathcal{L}(\alpha) \cdot \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha \mid \beta) := \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- jedes $a \in \Sigma$ ist ein regulärer Ausdruck

Angenommen, α, β sind reguläre Ausdrücke. Dann auch:

- $(\alpha \cdot \beta)$
- $(\alpha \mid \beta)$
- α^*

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$
- $\mathcal{L}(a) := \{a\}$

- $\mathcal{L}(\alpha \cdot \beta) := \mathcal{L}(\alpha) \cdot \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha \mid \beta) := \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha^*) := \mathcal{L}(\alpha)^*$

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- jedes $a \in \Sigma$ ist ein regulärer Ausdruck

Angenommen, α, β sind reguläre Ausdrücke. Dann auch:

- $(\alpha \cdot \beta)$
- $(\alpha \mid \beta)$
- α^*

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$
- $\mathcal{L}(a) := \{a\}$

- $\mathcal{L}(\alpha \cdot \beta) :=$ $\mathcal{L}(\alpha) \cdot \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha \mid \beta) :=$ $\mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha^*) := \mathcal{L}(\alpha)^*$

- α^+ statt $(\alpha \cdot \alpha^*)$

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- jedes $a \in \Sigma$ ist ein regulärer Ausdruck

Angenommen, α, β sind reguläre Ausdrücke. Dann auch:

- $(\alpha \cdot \beta)$
- $(\alpha \mid \beta)$
- α^*

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$
- $\mathcal{L}(a) := \{a\}$

- $\mathcal{L}(\alpha \cdot \beta) := \mathcal{L}(\alpha) \cdot \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha \mid \beta) := \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha^*) := \mathcal{L}(\alpha)^*$

- α^+ statt $(\alpha \cdot \alpha^*)$
- $\cdot$ kann weggelassen werden

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- jedes $a \in \Sigma$ ist ein regulärer Ausdruck

Angenommen, α, β sind reguläre Ausdrücke. Dann auch:

- $(\alpha \cdot \beta)$
- $(\alpha \mid \beta)$
- α^*

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$
- $\mathcal{L}(a) := \{a\}$
- $\mathcal{L}(\alpha \cdot \beta) := \mathcal{L}(\alpha) \cdot \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha \mid \beta) := \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha^*) := \mathcal{L}(\alpha)^*$

- α^+ statt $(\alpha \cdot \alpha^*)$
- $\cdot$ kann weggelassen werden
- bestimmte Klammern können weggelassen werden

Reguläre Ausdrücke

Syntax

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- jedes $a \in \Sigma$ ist ein regulärer Ausdruck

Angenommen, α, β sind reguläre Ausdrücke. Dann auch:

- $(\alpha \cdot \beta)$
- $(\alpha \mid \beta)$
- α^*

Semantik

- $\mathcal{L}(\emptyset) := \emptyset$
- $\mathcal{L}(\varepsilon) := \{\varepsilon\}$
- $\mathcal{L}(a) := \{a\}$

- $\mathcal{L}(\alpha \cdot \beta) :=$ $\mathcal{L}(\alpha) \cdot \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha \mid \beta) :=$ $\mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$
- $\mathcal{L}(\alpha^*) := \mathcal{L}(\alpha)^*$

- α^+ statt $(\alpha \cdot \alpha^*)$
- $\cdot$ kann weggelassen werden
- bestimmte Klammern können weggelassen werden
- „Stern vor Punkt vor Strich“

Reguläre Ausdrücke und reguläre Sprachen

Satz

Sei Σ ein Alphabet. Dann gilt für jede Sprache $L \subseteq \Sigma^*$:

$$L \text{ regulär} \Leftrightarrow \text{ex. regulärer Ausdruck } \alpha \text{ mit } \mathcal{L}(A) = L.$$

Satz

Sei Σ ein Alphabet. Dann gilt für jede Sprache $L \subseteq \Sigma^*$:

L regulär $\Leftrightarrow$ ex. regulärer Ausdruck α mit $\mathcal{L}(A) = L$.

Beweisidee $\Leftarrow$

Folgt direkt aus Definition und Abschluss von REG unter Konkatenation, Vereinigung und Kleene-Stern.

Reguläre Ausdrücke und reguläre Sprachen

Satz

Sei Σ ein Alphabet. Dann gilt für jede Sprache $L \subseteq \Sigma^*$:

$$L \text{ regulär} \Leftrightarrow \text{ex. regulärer Ausdruck } \alpha \text{ mit } \mathcal{L}(A) = L.$$

Beweisidee $\Rightarrow$

- konvertiere DFA in regulären Ausdruck

Beweisidee $\Leftarrow$

Folgt direkt aus Definition und Abschluss von REG unter Konkatenation, Vereinigung und Kleene-Stern.

Reguläre Ausdrücke und reguläre Sprachen

Satz

Sei Σ ein Alphabet. Dann gilt für jede Sprache $L \subseteq \Sigma^*$:

$$L \text{ regulär} \Leftrightarrow \text{ex. regulärer Ausdruck } \alpha \text{ mit } \mathcal{L}(A) = L.$$

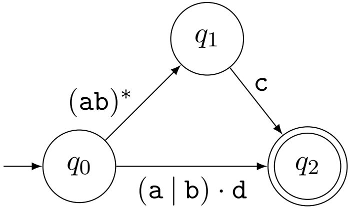
Beweisidee $\Rightarrow$

- konvertiere DFA in regulären Ausdruck
- Zwischenstufe: erweiterter NFA

Beweisidee $\Leftarrow$

Folgt direkt aus Definition und Abschluss von REG unter Konkatenation, Vereinigung und Kleene-Stern.

Erweiterter NFA



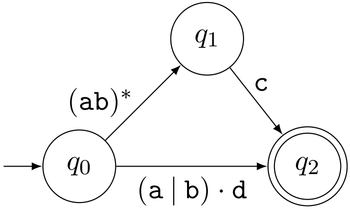
ENFA

- NFA mit regulären Ausdrücken an den Kanten
- Formal: Wie NFA; aber

$$\delta : Q \times Q \to \mathrm{RX}_\Sigma$$

- RX_Σ: reguläre Ausdrücke über Σ

Erweiterter NFA



ENFA

- NFA mit regulären Ausdrücken an den Kanten
- Formal: Wie NFA; aber

$$\delta : Q \times Q \to \mathrm{RX}_\Sigma$$

- RX_Σ: reguläre Ausdrücke über Σ

- Keine Kante zwischen p und q entspricht $\delta(p, q) = \emptyset$.
- Lässt sich direkt in ε-NFA konvertieren.

Der Konvertierungsalgorithmus

ENFA2RegEx

Eingabe: Ein ENFA $A := (\Sigma, Q, \delta, q_0, F)$.
Ausgabe: Ein regulärer Ausdruck α mit $\mathcal{L}(\alpha) = \mathcal{L}(A)$.

Der Konvertierungsalgorithmus

ENFA2RegEx

Eingabe: Ein ENFA $A := (\Sigma, Q, \delta, q_0, F)$.
Ausgabe: Ein regulärer Ausdruck α mit $\mathcal{L}(\alpha) = \mathcal{L}(A)$.

1. Stelle sicher, dass q_0 keine eingehenden Kanten hat.

Der Konvertierungsalgorithmus

ENFA2RegEx

Eingabe: Ein ENFA $A := (\Sigma, Q, \delta, q_0, F)$.
Ausgabe: Ein regulärer Ausdruck α mit $\mathcal{L}(\alpha) = \mathcal{L}(A)$.

1. Stelle sicher, dass q_0 keine eingehenden Kanten hat.
2. Stelle sicher, dass es genau einen akzeptierenden Zustand $q_F \neq q_0$ gibt, und dass dieser keine ausgehenden Kanten hat.
3. Falls $|Q| = 2$: Gib $\delta(q_0, q_F)$ aus.

Der Konvertierungsalgorithmus

ENFA2RegEx

Eingabe: Ein ENFA $A := (\Sigma, Q, \delta, q_0, F)$.
Ausgabe: Ein regulärer Ausdruck α mit $\mathcal{L}(\alpha) = \mathcal{L}(A)$.

1. Stelle sicher, dass q_0 keine eingehenden Kanten hat.
2. Stelle sicher, dass es genau einen akzeptierenden Zustand $q_F \neq q_0$ gibt, und dass dieser keine ausgehenden Kanten hat.
3. Falls $|Q| = 2$: Gib $\delta(q_0, q_F)$ aus.
4. Wähle einen Zustand $q_X \in Q - \{q_0, q_F\}$.

Der Konvertierungsalgorithmus

ENFA2RegEx

Eingabe: Ein ENFA $A := (\Sigma, Q, \delta, q_0, F)$.
Ausgabe: Ein regulärer Ausdruck α mit $\mathcal{L}(\alpha) = \mathcal{L}(A)$.

1. Stelle sicher, dass q_0 keine eingehenden Kanten hat.
2. Stelle sicher, dass es genau einen akzeptierenden Zustand $q_F \neq q_0$ gibt, und dass dieser keine ausgehenden Kanten hat.
3. Falls $|Q| = 2$: Gib $\delta(q_0, q_F)$ aus.
4. Wähle einen Zustand $q_X \in Q - \{q_0, q_F\}$.
5. Definiere ENFA $A' := (\Sigma, Q', \delta', q_0, F)$ durch:

Der Konvertierungsalgorithmus

ENFA2RegEx

Eingabe: Ein ENFA $A := (\Sigma, Q, \delta, q_0, F)$.
Ausgabe: Ein regulärer Ausdruck α mit $\mathcal{L}(\alpha) = \mathcal{L}(A)$.

1. Stelle sicher, dass q_0 keine eingehenden Kanten hat.
2. Stelle sicher, dass es genau einen akzeptierenden Zustand $q_F \neq q_0$ gibt, und dass dieser keine ausgehenden Kanten hat.
3. Falls $|Q| = 2$: Gib $\delta(q_0, q_F)$ aus.
4. Wähle einen Zustand $q_X \in Q - \{q_0, q_F\}$.
5. Definiere ENFA $A' := (\Sigma, Q', \delta', q_0, F)$ durch:
    1. $Q' := Q - \{q_X\}$,

Der Konvertierungsalgorithmus

ENFA2RegEx

Eingabe: Ein ENFA $A := (\Sigma, Q, \delta, q_0, F)$.
Ausgabe: Ein regulärer Ausdruck α mit $\mathcal{L}(\alpha) = \mathcal{L}(A)$.

1. Stelle sicher, dass q_0 keine eingehenden Kanten hat.
2. Stelle sicher, dass es genau einen akzeptierenden Zustand $q_F \neq q_0$ gibt, und dass dieser keine ausgehenden Kanten hat.
3. Falls $|Q| = 2$: Gib $\delta(q_0, q_F)$ aus.
4. Wähle einen Zustand $q_X \in Q - \{q_0, q_F\}$.
5. Definiere ENFA $A' := (\Sigma, Q', \delta', q_0, F)$ durch:
    1. $Q' := Q - \{q_X\}$,
    2. Für alle $q_V \in Q' - \{q_F\}$ und alle $q_N \in Q' - \{q_0\}$ ist

$$\delta'(q_V, q_N) := \left(\left(\delta(q_V, q_X) \cdot \delta(q_X, q_X)^* \cdot \delta(q_X, q_N)\right) \mid \delta(q_V, q_N)\right).$$

Der Konvertierungsalgorithmus

ENFA2RegEx

Eingabe: Ein ENFA $A := (\Sigma, Q, \delta, q_0, F)$.
Ausgabe: Ein regulärer Ausdruck α mit $\mathcal{L}(\alpha) = \mathcal{L}(A)$.

1. Stelle sicher, dass q_0 keine eingehenden Kanten hat.
2. Stelle sicher, dass es genau einen akzeptierenden Zustand $q_F \neq q_0$ gibt, und dass dieser keine ausgehenden Kanten hat.
3. Falls $|Q| = 2$: Gib $\delta(q_0, q_F)$ aus.
4. Wähle einen Zustand $q_X \in Q - \{q_0, q_F\}$.
5. Definiere ENFA $A' := (\Sigma, Q', \delta', q_0, F)$ durch:
   1. $Q' := Q - \{q_X\}$,
   2. Für alle $q_V \in Q' - \{q_F\}$ und alle $q_N \in Q' - \{q_0\}$ ist

$$\delta'(q_V, q_N) := \left(\left(\delta(q_V, q_X) \cdot \delta(q_X, q_X)^* \cdot \delta(q_X, q_N)\right) \mid \delta(q_V, q_N)\right).$$

6. Rufe `ENFA2RegEx` rekursiv auf A' auf und gib das Resultat zurück.

Substitution

Substitution von Σ_1 nach Σ_2

- Funktion $s : (\Sigma_1) \to \mathcal{P}(\Sigma_2^*)$,
- ordnet jedem Buchstaben $a \in \Sigma_1$ eine Sprache über Σ_2 zu.

Substitution

Substitution von Σ_1 nach Σ_2

- Funktion $s : (\Sigma_1) \to \mathcal{P}(\Sigma_2^*)$,
- ordnet jedem Buchstaben $a \in \Sigma_1$ eine Sprache über Σ_2 zu.
- Erweiterbar auf Wörter und Sprachen:

$$\begin{aligned} s(\varepsilon) &:= \varepsilon, \\ s(w \cdot a) &:= s(w) \cdot s(a), \\ s(L) &:= \bigcup_{w \in L} s(w) \end{aligned}$$

Substitution

Substitution von Σ_1 nach Σ_2

- Funktion $s : (\Sigma_1) \to \mathcal{P}(\Sigma_2^*)$,
- ordnet jedem Buchstaben $a \in \Sigma_1$ eine Sprache über Σ_2 zu.
- Erweiterbar auf Wörter und Sprachen:

$$\begin{aligned} s(\varepsilon) &:= \varepsilon, \\ s(w \cdot a) &:= s(w) \cdot s(a), \\ s(L) &:= \bigcup_{w \in L} s(w) \end{aligned}$$

- Heißt regulär, wenn alle $s(a)$ regulär.

Substitution

Substitution von Σ_1 nach Σ_2

- Funktion $s : (\Sigma_1) \to \mathcal{P}(\Sigma_2^*)$,
- ordnet jedem Buchstaben $a \in \Sigma_1$ eine Sprache über Σ_2 zu.
- Erweiterbar auf Wörter und Sprachen:

$$s(\varepsilon) := \varepsilon,$$
$$s(w \cdot a) := s(w) \cdot s(a),$$
$$s(L) := \bigcup_{w \in L} s(w)$$

- Heißt regulär, wenn alle $s(a)$ regulär.

Satz

Die Klasse der regulären Sprachen ist abgeschlossen unter regulärer Substitution.

Substitution

Substitution von Σ_1 nach Σ_2

- Funktion $s : (\Sigma_1) \to \mathcal{P}(\Sigma_2^*)$,
- ordnet jedem Buchstaben $a \in \Sigma_1$ eine Sprache über Σ_2 zu.
- Erweiterbar auf Wörter und Sprachen:

$$s(\varepsilon) := \varepsilon,$$
$$s(w \cdot a) := s(w) \cdot s(a),$$
$$s(L) := \bigcup_{w \in L} s(w)$$

- Heißt regulär, wenn alle $s(a)$ regulär.

Satz

Die Klasse der regulären Sprachen ist abgeschlossen unter regulärer Substitution.

Beweisidee

Nimm DFA für L, ersetze jede Kante a durch regulären Ausdruck für $s(a)$. Erhalte ENFA für $s(L)$.

Homomorphismus

Homomorphismus von Σ_1 nach Σ_2

- Funktion $h : (\Sigma_1)^* \to (\Sigma_2)^*$ mit
$$h(uv) = h(u)h(v) \text{ für alle } u, v \in \Sigma_1^*$$
- Definiert durch $h(a)$ für alle $a \in \Sigma$.
- Anschaulich: Ersetzt Buchstaben durch Wörter.
- $h(L) := \{h(w) \mid w \in L\}$.

Homomorphismus

Homomorphismus von Σ_1 nach Σ_2

- Funktion $h : (\Sigma_1)^* \to (\Sigma_2)^*$ mit
  $$h(uv) = h(u)h(v) \text{ für alle } u, v \in \Sigma_1^*$$
- Definiert durch $h(a)$ für alle $a \in \Sigma$.
- Anschaulich: Ersetzt Buchstaben durch Wörter.
- $h(L) := \{h(w) \mid w \in L\}$.

Lemma

REG ist abgeschlossen unter Homomorphismus.

Homomorphismus

Homomorphismus von Σ_1 nach Σ_2

- Funktion $h : (\Sigma_1)^* \to (\Sigma_2)^*$ mit
$$h(uv) = h(u)h(v) \text{ für alle } u, v \in \Sigma_1^*$$
- Definiert durch $h(a)$ für alle $a \in \Sigma$.
- Anschaulich: Ersetzt Buchstaben durch Wörter.
- $h(L) := \{h(w) \mid w \in L\}$.

Lemma

REG ist abgeschlossen unter Homomorphismus.

Inverser Homomorphismus

- Sei h Homomorphismus von Σ_1 nach Σ_2.
- $h^{-1}(w) := \{v \in (\Sigma_1)^* \mid h(v) = w\}$
- $h^{-1}(L) := \{v \in (\Sigma_1)^* \mid h(v) \in L\}$

Homomorphismus

Homomorphismus von Σ_1 nach Σ_2

- Funktion $h : (\Sigma_1)^* \to (\Sigma_2)^*$ mit
$$h(uv) = h(u)h(v) \text{ für alle } u, v \in \Sigma_1^*$$
- Definiert durch $h(a)$ für alle $a \in \Sigma$.
- Anschaulich: Ersetzt Buchstaben durch Wörter.
- $h(L) := \{h(w) \mid w \in L\}$.

Lemma

REG ist abgeschlossen unter Homomorphismus.

Inverser Homomorphismus

- Sei h Homomorphismus von Σ_1 nach Σ_2.
- $h^{-1}(w) := \{v \in (\Sigma_1)^* \mid h(v) = w\}$
- $h^{-1}(L) := \{v \in (\Sigma_1)^* \mid h(v) \in L\}$

Lemma

REG ist abgeschlossen unter inversem Homomorphismus.

Anwendungen

L ist regulär

Konstruktiver Beweis:

1. Nimm eine reguläre Sprache R (oder mehrere),

Anwendungen

L ist regulär

Konstruktiver Beweis:

1. Nimm eine reguläre Sprache R (oder mehrere),
2. benutze Abschlusseigenschaften auf R,

Anwendungen

L ist regulär

Konstruktiver Beweis:

1. Nimm eine reguläre Sprache R (oder mehrere),
2. benutze Abschlusseigenschaften auf R,
3. erhalte L.

Anwendungen

L ist regulär

Konstruktiver Beweis:

1. Nimm eine reguläre Sprache R (oder mehrere),
2. benutze Abschlusseigenschaften auf R,
3. erhalte L.
4. Also ist L regulär.

Anwendungen

L ist regulär

Konstruktiver Beweis:

1. Nimm eine reguläre Sprache R (oder mehrere),
2. benutze Abschlusseigenschaften auf R,
3. erhalte L.
4. Also ist L regulär.

L ist nicht regulär

Indirekter Beweis:

1. Angenommen, L ist regulär...

Anwendungen

L ist regulär

Konstruktiver Beweis:

1. Nimm eine reguläre Sprache R (oder mehrere),
2. benutze Abschlusseigenschaften auf R,
3. erhalte L.
4. Also ist L regulär.

L ist nicht regulär

Indirekter Beweis:

1. Angenommen, L ist regulär. . .
2. benutze Abschlusseigenschaften auf L. . .

Anwendungen

L ist regulär

Konstruktiver Beweis:

1. Nimm eine reguläre Sprache R (oder mehrere),
2. benutze Abschlusseigenschaften auf R,
3. erhalte L.
4. Also ist L regulär.

L ist nicht regulär

Indirekter Beweis:

1. Angenommen, L ist regulär. . .
2. benutze Abschlusseigenschaften auf L. . .
3. erhalte Sprache, von der wir wissen, dass sie nicht regulär ist.

Anwendungen

L ist regulär

Konstruktiver Beweis:

1. Nimm eine reguläre Sprache R (oder mehrere),
2. benutze Abschlusseigenschaften auf R,
3. erhalte L.
4. Also ist L regulär.

L ist nicht regulär

Indirekter Beweis:

1. Angenommen, L ist regulär...
2. benutze Abschlusseigenschaften auf L...
3. erhalte Sprache, von der wir wissen, dass sie nicht regulär ist.
4. Widerspruch! Also ist L nicht regulär.