Reguläre Ausdrücke

Reguläre Ausdrücke

- Beschreibung regulärer Sprachen
- Anwendungsgebiete:
  - Compilerbau: Lexikalische Analyse, Eingabe für Scannergeneratoren
  - Softwaretechnik: Spezifikation zulässiger Eingaben für Dialogschnittstellen
  - Suchbefehle: Unix, Webbrowser
  - Kontrollbedingungen, Ablaufspezifikationen
  - ....

Definition

- I : endliches Alphabet mit $lambda, empty, +, \circ, *, (,) \notin I$

Menge $REX(I)$ der *regulären Ausdrücke über I*

1. $empty, lambda \in REX(I)$,
2. $I \subseteq REX(I)$,
3. für alle $r, r_1, r_2 \in REX(I)$ sind auch $(r_1 + r_2)$, $(r_1 \circ r_2)$ und (r^*) in $REX(I)$.

Sprache regulärer Ausdrücke

Jedem regulären Ausdruck r über I wird wie folgt eine Sprache $L(r)$ zugeordnet:

1. $L(empty) = \emptyset$, $L(lambda) = \{\lambda\}$ und $L(x) = \{x\}$ für alle $x \in I$,
2. für alle $r, r_1, r_2 \in REX(I)$
   (a) $L((r_1 + r_2)) = L(r_1) \cup L(r_2)$,
   (b) $L((r_1 \circ r_2)) = L(r_1)L(r_2)$,
   (c) $L((r^*)) = L(r)^*$.

Klammerregeln/Vereinfachung

1. Der Operator $*$ bindet stärker als $\circ$, entsprechende Klammern dürfen entfallen
2. Der Operator $\circ$ bindet stärker als $+$, entsprechende Klammern dürfen entfallen
3. Der Operator $\circ$ und $+$ sind assoziativ, entsprechende Klammern dürfen entfallen
4. Der Operator $\circ$ darf entfallen.

Rechenregeln

1. $empty \circ r = empty = r \circ empty$
2. $empty^* = lambda = lambda^*$
3. $lambda + r^* = r^*$, $lambda + r \circ r^* = r^*$
4. Kommutativgesetz: $r + t = t + r$
5. Neutralität: $r + empty = r = lambda \circ r = r \circ lambda$
6. Assoziativgesetze: $(r + s) + t = r + (s + t) = r + s + t$, $(r \circ s) \circ t = r \circ (s \circ t) = r \circ s \circ t = rst$
7. Distributivgesetze: $r(s + t) = rs + rt$, $(s + t)r = sr + tr$
8. Idempotenz: $(r^*)^* = r^*$, $r + r = r$

Vereinfachung eines regulären Ausdrucks

Beispiel

$((b^*) + ((b^*) \circ (a \circ (a^*)))) = (b^*) + ((b^*) \circ (a \circ (a^*))) = b^* + (b^* \circ (a \circ a^*)) = b^* + (b^* \circ a \circ a^*) = b^* + b^* \circ a \circ a^* = b^* + b^*aa^* = b^* \circ lambda + b^*aa^* = b^* \circ (lambda + aa^*) = b^*(a^*) = b^*a^*$

Sprache eines regulären Ausdrucks

Beispiel

$$
\begin{aligned}
L(b^* + b^*aa^*) &= L(b^*) \cup L(b^*aa^*) \\
&= L(b)^* \cup L(b^*)L(aa^*) \\
&= \{b\}^* \cup L(b)^*L(a)L(a^*) \\
&= \{b\}^* \cup \{b\}^*\{a\}L(a)^* \\
&= \{b\}^* \cup \{b\}^*\{a\}\{a\}^* \\
&= \{b\}^*(\{\lambda\} \cup \{a\}\{a\}^*) \\
&= \{b\}^*\{a\}^* = \{b^m a^n \mid m, n \in \mathbb{N}\} \\
&= \{w \in \{a, b, \}^* \mid \; ab \text{ ist kein Teilwort von } w\}
\end{aligned}
$$

Satz

1. Jeder reguläre Ausdruck r definiert eine reguläre Sprache, d.h. $L(r) \in \mathcal{L}_{REG(I)}$.
2. Jede reguläre Sprache $L \in \mathcal{L}_{REG(I)}$ lässt sich durch einen regulären Ausdruck beschreiben, d.h. es existiert ein regulärer Ausdruck r mit $L(r) = L$.

Schlussfolgerung

- Endliche Automaten erkennen genau die Klasse der regulären Sprachen.
- Reguläre Audrücke beschreiben genau die Klasse der regulären Sprachen.

$$\boxed{\mathcal{L}_{REG(I)} = \mathcal{L}_{REX(I)} = \mathcal{L}_{AUT(I)}}$$

mit $\mathcal{L}_{REX(I)} = \{L(r) \mid r \in REX(I)\}$
und $\mathcal{L}_{AUT(I)} = \{L(A) \mid A \text{ ist ein endlicher Automat}\}$