Sprachen sind Mengen

$$L_{public} = \{\text{public}\}$$
$$L_{private} = \{\text{private}\}$$
$$L_{protected} = \{\text{protected}\}$$
$$L_{package} = \{\varepsilon\}$$

$$L_{modifier} = L_{public} \cup L_{private} \cup L_{protected} \cup L_{package}$$

$$L_{whitespace} = \{\text{␣}, \backslash\text{t}, \backslash\text{r}, \backslash\text{n}\}$$
$$L_{class} = \{\text{class}\}$$

$$L_{java} = L_{modifier} \circ L_{whitespace} \circ L_{class} \circ \cdots = \{\text{public class}, \text{private class}, \text{protected class}\}$$

$$L_{identifier} = (\{a, \ldots, z, A, \ldots, Z\}) \circ (\{a, \ldots, z, A, \ldots, Z, 0, \ldots, 9\})^*$$

Sprachen sind Mengen

$$L_{public} = \{\mathrm{p}\} \circ \{\mathrm{u}\} \circ \{\mathrm{b}\} \circ \{\mathrm{l}\} \circ \{\mathrm{i}\} \circ \{\mathrm{c}\}$$
$$L_{private} = \{\mathrm{p}\} \circ \{\mathrm{r}\} \circ \{\mathrm{i}\} \circ \{\mathrm{v}\} \circ \{\mathrm{a}\} \circ \{\mathrm{t}\} \circ \{\mathrm{e}\}$$
$$L_{protected} = \{\mathrm{p}\} \circ \{\mathrm{r}\} \circ \{\mathrm{o}\} \circ \{\mathrm{t}\} \circ \{\mathrm{e}\} \circ \{\mathrm{c}\} \circ \{\mathrm{t}\} \circ \{\mathrm{e}\} \circ \{\mathrm{d}\}$$
$$L_{package} = \{\varepsilon\}$$

$$L_{modifier} = L_{public} \cup L_{private} \cup L_{protected} \cup L_{package}$$

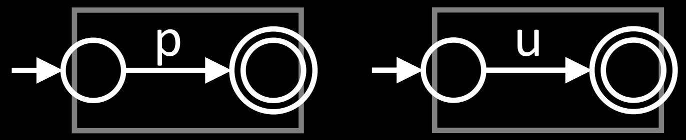
$$L_{whitespace} = \{\text{␣}\} \cup \{\backslash\mathrm{t}\} \cup \{\backslash\mathrm{r}\} \cup \{\backslash\mathrm{n}\}$$
$$L_{class} = \{\mathrm{c}\} \circ \{\mathrm{l}\} \circ \{\mathrm{a}\} \circ \{\mathrm{s}\} \circ \{\mathrm{s}\}$$

$$L_{java} = L_{modifier} \circ L_{whitespace} \circ L_{class} \circ \cdots = \{\text{public class}, \text{private class}, \text{protected class}\}$$

$$L_{identifier} = (\{a\}| \dots |\{z\}|\{A\}| \dots |\{Z\}) \circ (\{a\}| \dots |\{z\}|\{A\}| \dots |\{Z\}|\{0\}| \dots |\{9\})^*$$

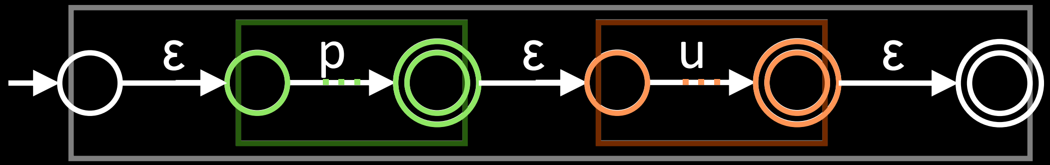
Sprachen sind Mengen

$$L_{public} = \{p\} \circ \{u\} \circ \{b\} \circ \{l\} \circ \{i\} \circ \{c\}$$

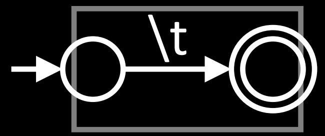


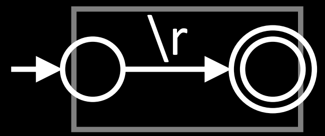
Sprachen sind Mengen

$$L_{public} = \{p\} \circ \{u\} \circ \{b\} \circ \{l\} \circ \{i\} \circ \{c\}$$



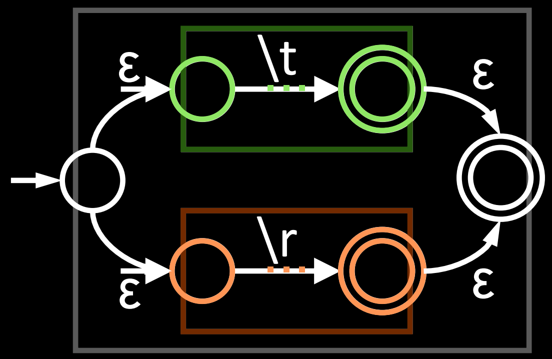
Sprachen sind Mengen





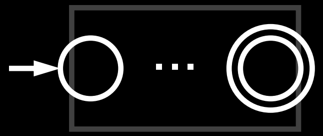
$$L_{whitespace} = \{\text{␣}\} \cup \{\backslash t\} \cup \{\backslash r\} \cup \{\backslash n\}$$

Sprachen sind Mengen



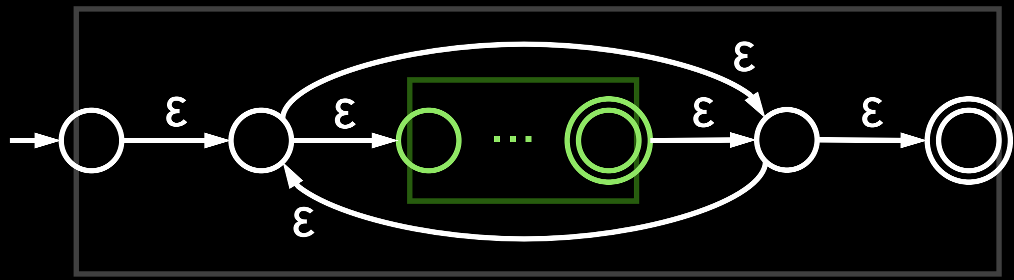
$$L_{whitespace} = \{␣\} \cup \{\backslash t\} \cup \{\backslash r\} \cup \{\backslash n\}$$

Sprachen sind Mengen



$$L_{identifier} = (\{a\}| \dots |\{z\}|\{A\}| \dots |\{Z\}) \circ (\{a\}| \dots |\{z\}|\{A\}| \dots |\{Z\}|\{0\}| \dots |\{9\})^*$$

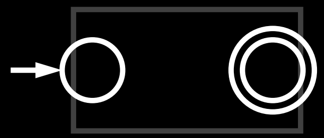
Sprachen sind Mengen



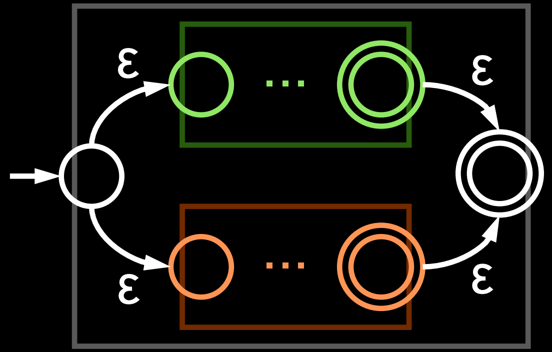
$$L_{identifier} = (\{a\}|\ldots|\{z\}|\{A\}|\ldots|\{Z\}) \circ (\{a\}|\ldots|\{z\}|\{A\}|\ldots|\{Z\}|\{0\}|\ldots|\{9\})^*$$

Reguläre Ausdrücke

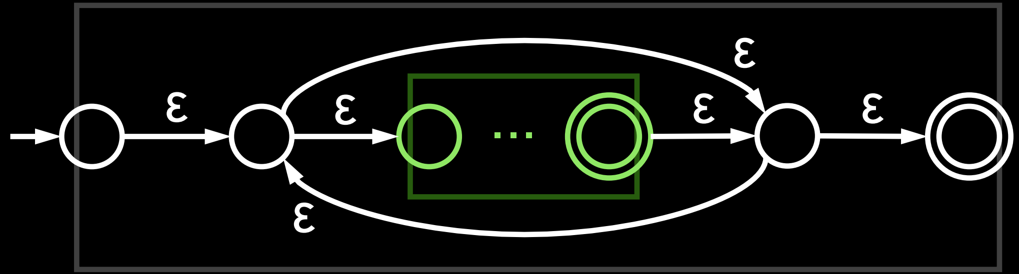
r = ∅



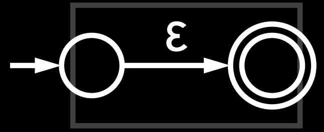
r = s|t



r = s*

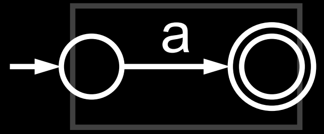


r = ε

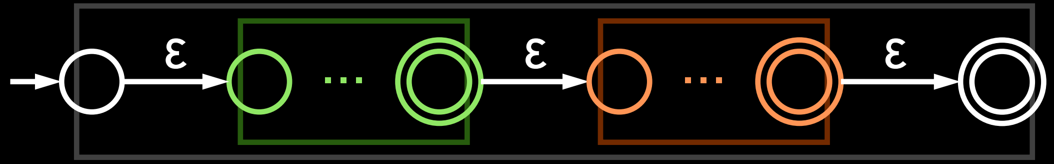


Thompson Konstruktion

r = a



r = s·t



Find the first match

`input.match( 'a' );`	b	b	a	a	b	a	b	b	a	a	a	a	b
`input.match( 'ab' );`	b	b	a	a	b	a	b	b	a	a	a	a	b
`input.match( 'a*' );`	b	b	a	a	b	a	b	b	a	a	a	a	b
`input.match( 'a*b' );`	b	b	a	a	b	a	b	b	a	a	a	a	b
`input.match( 'a*bb' );`	b	b	a	a	b	a	b	b	a	a	a	a	b
`input.match( 'ba*' );`	b	b	a	a	b	a	b	b	a	a	a	a	b
`input.match( '(a|b)' );`	b	b	a	a	b	a	b	b	a	a	a	a	b
`input.match( '(a|b)*' );`	b	b	a	a	b	a	b	b	a	a	a	a	b

Find the first match

<table>
<tr><td><code>input.match('a');</code></td><td>b</td><td>b</td><td>a</td><td>a</td><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>a</td><td>a</td><td>a</td><td>b</td></tr>
<tr><td><code>input.match('ab');</code></td><td>b</td><td>b</td><td>a</td><td>a</td><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>a</td><td>a</td><td>a</td><td>b</td></tr>
<tr><td><code>input.match('a*');</code></td><td>b</td><td>b</td><td>a</td><td>a</td><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>a</td><td>a</td><td>a</td><td>b</td></tr>
<tr><td><code>input.match('a*b');</code></td><td>b</td><td>b</td><td>a</td><td>a</td><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>a</td><td>a</td><td>a</td><td>b</td></tr>
<tr><td><code>input.match('a*bb');</code></td><td>b</td><td>b</td><td>a</td><td>a</td><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>a</td><td>a</td><td>a</td><td>b</td></tr>
<tr><td><code>input.match('ba*');</code></td><td>b</td><td>b</td><td>a</td><td>a</td><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>a</td><td>a</td><td>a</td><td>b</td></tr>
<tr><td><code>input.match('(a|b)');</code></td><td>b</td><td>b</td><td>a</td><td>a</td><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>a</td><td>a</td><td>a</td><td>b</td></tr>
<tr><td><code>input.match('(a|b)*');</code></td><td>b</td><td>b</td><td>a</td><td>a</td><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>a</td><td>a</td><td>a</td><td>b</td></tr>
</table>

Unser Adventskalender

Typ	Name	Erlaubte Produktionen	Akzeptierender Automat	(Trennendes) Beispiel
3	regulär	$P \subseteq N \times (\{\varepsilon\} \cup T^* \cup T^*N)$ oder $P \subseteq N \times (\{\varepsilon\} \cup T^* \cup NT^*)$	endlicher Automat	$L = \{a^k b^l \mid k, l \in \mathbb{N}_0\}$
2	kontextfrei	$P \subseteq N \times (N \cup T)^*$		$L = \{a^k b^k \mid k \in \mathbb{N}_0\}$

Sprache eines regulären Ausdrucks

Die von einem regulären Ausdruck R beschriebene formale Sprache $L(R)$ ist wie folgt definiert:

- $L(\emptyset) = \emptyset$ (Die Sprache des leeren Ausdrucks ist die leere Menge).
- Für jedes $a \in \Sigma$ ist $\mathrm{L}(a) = \{a\}$.
- Sind R_1 und R_2 reguläre Ausdrücke, dann ist $\mathrm{L}(R_1|R_2) = L(R_1) \cup L(R_2)$.
- Sind R_1 und R_2 reguläre Ausdrücke, dann ist $\mathrm{L}(R_1 R_2) = L(R_1) \cdot L(R_2)$.
- Ist R ein regulärer Ausdruck, dann ist $\mathrm{L}(R\,*) = L(R)^*$.

Ankündigung

Regex-Bibliotheken verwenden meist erweiterte reguläre Ausdrücke:

- Quantifizierer: Stern, Plus, Fragezeichen, geschweifte Klammer
- Zeichenklassen: eckige Klammer, Negation, von-bis-Schreibweise
- Metazeichen: Punkt
- Escaping: über Zeichenklasse oder Backslash
- Anker: ^ und $
- Rückwärtsverweise: \1

Abschlusseigenschaften

	reguläre Sprachen	kontextfrei	???	???
Konkatenation	Ja			
Kleene-Stern	Ja			
Vereinigung	Ja			
Schnitt				
Komplement				

Das Komplement regulärer Sprachen

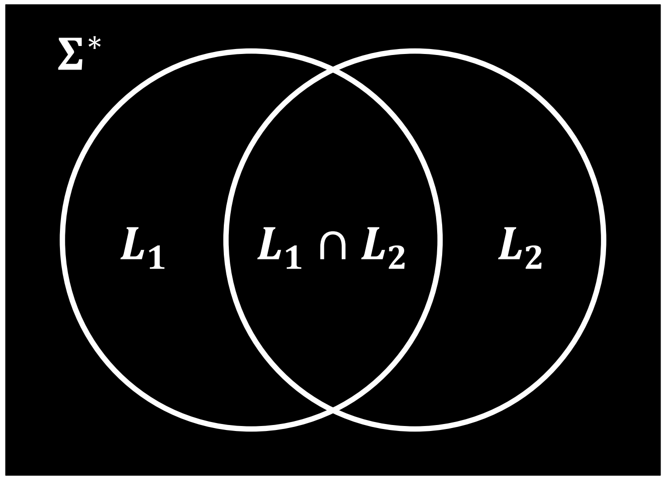
Sei $M = (Q, q_0, \Sigma, \delta, F)$ ein deterministischer endlicher Akzeptor mit $L(M) = L$.

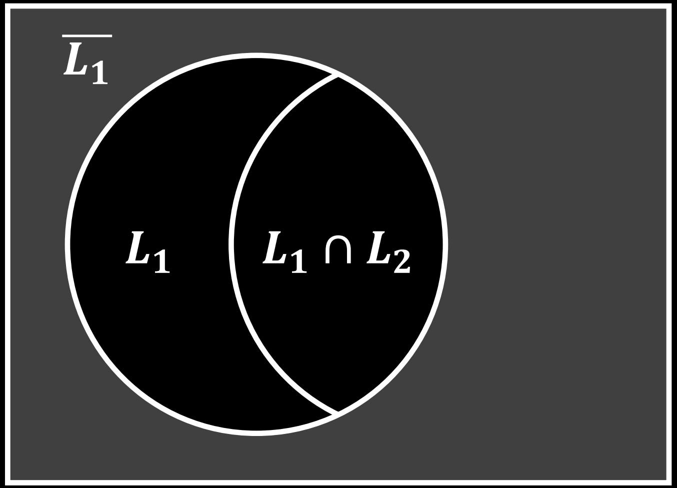
Für den endlicher Akzeptor M', der die Sprache $\Sigma^*\backslash \mathrm{L}$ akzeptiert, gilt folgender Zusammenhang:

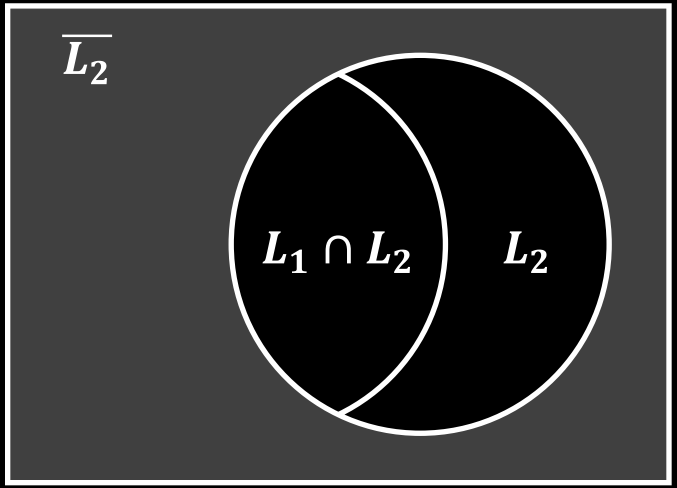
M' akzeptiert genau dann, wenn M nicht akzeptiert.

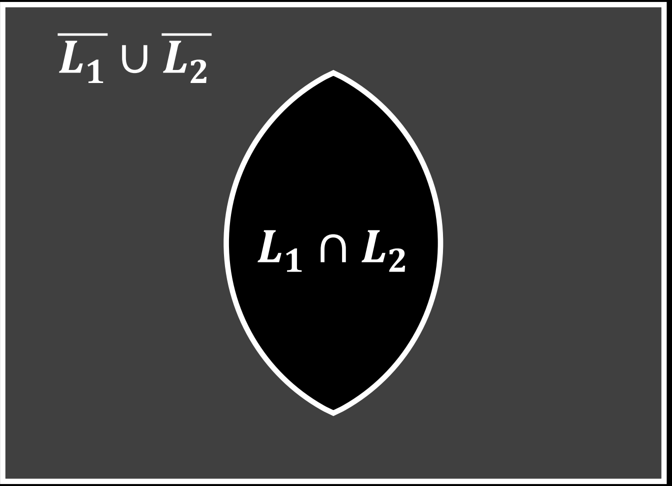
Also leistet $M' = (Q, q_0, \Sigma, \delta, Z\backslash F)$ das Gewünschte.

Durchschnitt regulärer Sprachen









Fazit: $\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$

Abschlusseigenschaften

	reguläre Sprachen	kontextfrei	???	???
Konkatenation	Ja			
Kleene-Stern	Ja			
Vereinigung	Ja			
Schnitt	Ja			
Komplement	Ja			