Regulární výrazy

Regulární výrazy

Regulární výrazy popisující jazyky nad abecedou Σ:

- $\emptyset$, ε, a (kde $a \in \Sigma$) jsou regulární výrazy:

  $\emptyset$ ... označuje prázdný jazyk
  ε ... označuje jazyk $\{\varepsilon\}$
  a ... označuje jazyk $\{a\}$

- Jestliže α, β jsou regulární výrazy, pak i $(\alpha + \beta)$, $(\alpha \cdot \beta)$, (α^*) jsou regulární výrazy:

  $(\alpha + \beta)$ ... označuje sjednocení jazyků označených α a β
  $(\alpha \cdot \beta)$ ... označuje zřetězení jazyků označených α a β
  (α^*) ... označuje iteraci jazyka označeného α

- Neexistují žádné další regulární výrazy než ty definované podle předchozích dvou bodů.

Regulární výrazy

Příklad: abeceda $\Sigma = \{0, 1\}$

- Podle definice jsou `0` i `1` regulární výrazy.

Regulární výrazy

Příklad: abeceda $\Sigma = \{0, 1\}$

- Podle definice jsou $\texttt{0}$ i $\texttt{1}$ regulární výrazy.
- Protože $\texttt{0}$ i $\texttt{1}$ jsou regulární výrazy, je i $(\texttt{0} + \texttt{1})$ regulární výraz.

Regulární výrazy

Příklad: abeceda $\Sigma = \{0, 1\}$

- Podle definice jsou 0 i 1 regulární výrazy.
- Protože 0 i 1 jsou regulární výrazy, je i $(0 + 1)$ regulární výraz.
- Protože 0 je regulární výraz, je i (0^*) regulární výraz.

Regulární výrazy

Příklad: abeceda $\Sigma = \{0, 1\}$

- Podle definice jsou 0 i 1 regulární výrazy.
- Protože 0 i 1 jsou regulární výrazy, je i $(0 + 1)$ regulární výraz.
- Protože 0 je regulární výraz, je i (0^*) regulární výraz.
- Protože $(0 + 1)$ i (0^*) jsou regulární výrazy, je i $((0 + 1) \cdot (0^*))$ regulární výraz.

Regulární výrazy

Příklad: abeceda $\Sigma = \{0, 1\}$

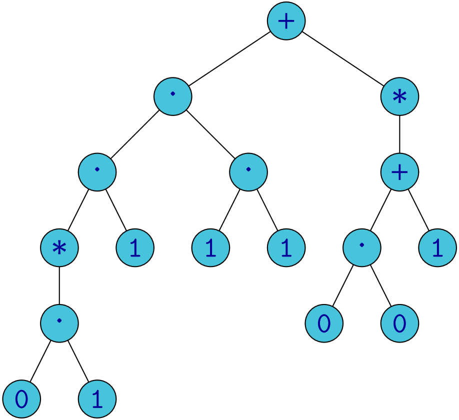
- Podle definice jsou 0 i 1 regulární výrazy.
- Protože 0 i 1 jsou regulární výrazy, je i $(0 + 1)$ regulární výraz.
- Protože 0 je regulární výraz, je i (0^*) regulární výraz.
- Protože $(0 + 1)$ i (0^*) jsou regulární výrazy, je i $((0 + 1) \cdot (0^*))$ regulární výraz.

Poznámka: Jestliže α je regulární výraz, zápisem $\mathcal{L}(\alpha)$ označujeme jazyk definovaný regulárním výrazem α.

$$\mathcal{L}(((0 + 1) \cdot (0^*))) = \{0,\ 1,\ 00,\ 10,\ 000,\ 100,\ 0000,\ 1000,\ 00000,\ \ldots\}$$

Regulární výrazy

Strukturu regulárního výrazu si můžeme znázornit abstraktním syntaktickým stromem:



$$(((((0 \cdot 1)^*) \cdot 1) \cdot (1 \cdot 1)) + (((0 \cdot 0) + 1)^*))$$

Regulární výrazy

Formální definice sémantiky regulárních výrazů:

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

Regulární výrazy

Aby byl zápis regulárních výrazů přehlednější a stručnější, používáme následují pravidla:

- Vynecháváme vnější pár závorek.
- Vynecháváme závorky, které jsou zbytečné vzhledem k asociativitě operací sjednocení ($+$) a zřetězení ($\cdot$).
- Vynecháváme závorky, které jsou zbytečné vzhledem k prioritě operací (nejvyšší prioritu má iterace (*), menší zřetězení ($\cdot$) a nejmenší sjednocení ($+$)).
- Nepíšeme tečku pro zřetězení.

Příklad: Místo

$$(((((0 \cdot 1)^*) \cdot 1) \cdot (1 \cdot 1)) + (((0 \cdot 0) + 1)^*))$$

obvykle píšeme

$$(01)^* 111 + (00 + 1)^*$$

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

$\mathtt{a}$ ... jazyk tvořený jediným slovem $\mathtt{a}$

Příklady: Ve všech případech $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

$\mathtt{a}$ . . . jazyk tvořený jediným slovem $\mathtt{a}$

$\mathtt{ab}$ . . . jazyk tvořený jediným slovem $\mathtt{ab}$

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

$\mathtt{a}$ ... jazyk tvořený jediným slovem $\mathtt{a}$

$\mathtt{ab}$ ... jazyk tvořený jediným slovem $\mathtt{ab}$

$\mathtt{a} + \mathtt{b}$ ... jazyk tvořený dvěma slovy $\mathtt{a}$ a $\mathtt{b}$

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

$\mathtt{a}$ ... jazyk tvořený jediným slovem $\mathtt{a}$

$\mathtt{ab}$ ... jazyk tvořený jediným slovem $\mathtt{ab}$

$\mathtt{a} + \mathtt{b}$ ... jazyk tvořený dvěma slovy $\mathtt{a}$ a $\mathtt{b}$

$\mathtt{a}^*$ ... jazyk tvořený slovy ε, $\mathtt{a}$, $\mathtt{aa}$, $\mathtt{aaa}$, ...

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

$\mathtt{a}$ . . . jazyk tvořený jediným slovem $\mathtt{a}$

$\mathtt{ab}$ . . . jazyk tvořený jediným slovem $\mathtt{ab}$

$\mathtt{a} + \mathtt{b}$ . . . jazyk tvořený dvěma slovy $\mathtt{a}$ a $\mathtt{b}$

$\mathtt{a}^*$ . . . jazyk tvořený slovy ε, $\mathtt{a}$, $\mathtt{aa}$, $\mathtt{aaa}$, . . .

$(\mathtt{ab})^*$ . . . jazyk tvořený slovy ε, $\mathtt{ab}$, $\mathtt{abab}$, $\mathtt{ababab}$, . . .

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

$\mathtt{a}$	... jazyk tvořený jediným slovem $\mathtt{a}$
$\mathtt{ab}$	... jazyk tvořený jediným slovem $\mathtt{ab}$
$\mathtt{a} + \mathtt{b}$	... jazyk tvořený dvěma slovy $\mathtt{a}$ a $\mathtt{b}$
$\mathtt{a}^*$	... jazyk tvořený slovy ε, $\mathtt{a}$, $\mathtt{aa}$, $\mathtt{aaa}$, ...
$(\mathtt{ab})^*$	... jazyk tvořený slovy ε, $\mathtt{ab}$, $\mathtt{abab}$, $\mathtt{ababab}$, ...
$(\mathtt{a} + \mathtt{b})^*$	... jazyk tvořený všemi slovy nad abecedou $\{\mathtt{a}, \mathtt{b}\}$

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

$\mathtt{a}$	... jazyk tvořený jediným slovem $\mathtt{a}$
$\mathtt{ab}$	... jazyk tvořený jediným slovem $\mathtt{ab}$
$\mathtt{a}+\mathtt{b}$	... jazyk tvořený dvěma slovy $\mathtt{a}$ a $\mathtt{b}$
$\mathtt{a}^*$	... jazyk tvořený slovy ε, $\mathtt{a}$, $\mathtt{aa}$, $\mathtt{aaa}$, ...
$(\mathtt{ab})^*$	... jazyk tvořený slovy ε, $\mathtt{ab}$, $\mathtt{abab}$, $\mathtt{ababab}$, ...
$(\mathtt{a}+\mathtt{b})^*$	... jazyk tvořený všemi slovy nad abecedou $\{\mathtt{a}, \mathtt{b}\}$
$(\mathtt{a}+\mathtt{b})^*\mathtt{aa}$	... jazyk tvořený všemi slovy končícími $\mathtt{aa}$

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

$\mathtt{a}$ ... jazyk tvořený jediným slovem $\mathtt{a}$

$\mathtt{ab}$ ... jazyk tvořený jediným slovem $\mathtt{ab}$

$\mathtt{a} + \mathtt{b}$ ... jazyk tvořený dvěma slovy $\mathtt{a}$ a $\mathtt{b}$

$\mathtt{a}^*$ ... jazyk tvořený slovy ε, $\mathtt{a}$, $\mathtt{aa}$, $\mathtt{aaa}$, ...

$(\mathtt{ab})^*$ ... jazyk tvořený slovy ε, $\mathtt{ab}$, $\mathtt{abab}$, $\mathtt{ababab}$, ...

$(\mathtt{a} + \mathtt{b})^*$ ... jazyk tvořený všemi slovy nad abecedou $\{\mathtt{a}, \mathtt{b}\}$

$(\mathtt{a} + \mathtt{b})^*\mathtt{aa}$ ... jazyk tvořený všemi slovy končícími $\mathtt{aa}$

$(\mathtt{ab})^*\mathtt{bbb}(\mathtt{ab})^*$ ... jazyk tvořený všemi slovy obsahujícími podslovo $\mathtt{bbb}$ předcházené i následované libovolným počtem slov $\mathtt{ab}$

Regulární výrazy

$(\mathtt{a}+\mathtt{b})^*\mathtt{aa}+(\mathtt{ab})^*\mathtt{bbb}(\mathtt{ab})^*$ ... jazyk tvořený všemi slovy, která buď končí aa nebo obsahují podslovo bbb předcházené i následované libovolným počtem slov ab

Regulární výrazy

$(\mathtt{a}+\mathtt{b})^*\mathtt{aa}+(\mathtt{ab})^*\mathtt{bbb}(\mathtt{ab})^*$ ... jazyk tvořený všemi slovy, která buď končí $\mathtt{aa}$ nebo obsahují podslovo $\mathtt{bbb}$ předcházené i následované libovolným počtem slov $\mathtt{ab}$

$(\mathtt{a}+\mathtt{b})^*\mathtt{b}(\mathtt{a}+\mathtt{b})^*$ ... jazyk tvořený všemi slovy obsahujícími alespoň jeden symbol $\mathtt{b}$

Regulární výrazy

$(\mathtt{a}+\mathtt{b})^*\mathtt{aa}+(\mathtt{ab})^*\mathtt{bbb}(\mathtt{ab})^*$ ... jazyk tvořený všemi slovy, která buď končí aa nebo obsahují podslovo bbb předcházené i následované libovolným počtem slov ab

$(\mathtt{a}+\mathtt{b})^*\mathtt{b}(\mathtt{a}+\mathtt{b})^*$ ... jazyk tvořený všemi slovy obsahujícími alespoň jeden symbol b

$\mathtt{a}^*(\mathtt{ba}^*\mathtt{ba}^*)^*$ ... jazyk tvořený všemi slovy obsahujícími sudý počet symbolů b

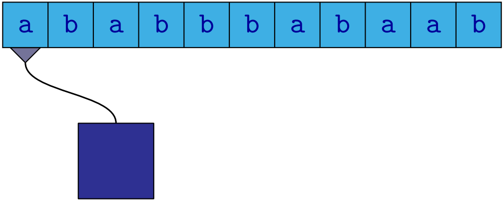
Konečné automaty

Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

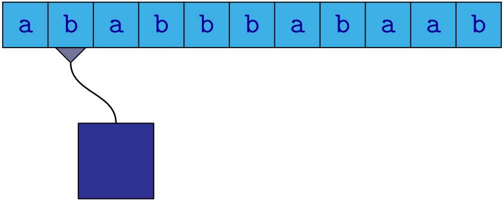


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

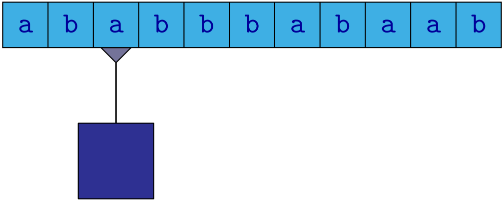


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů b.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

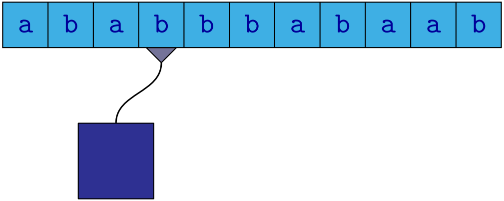


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

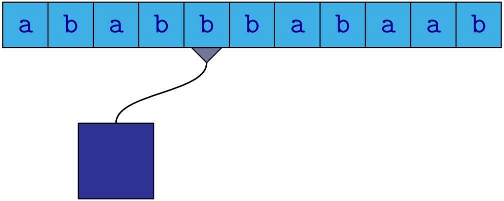


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

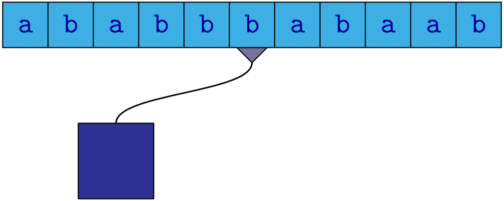


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

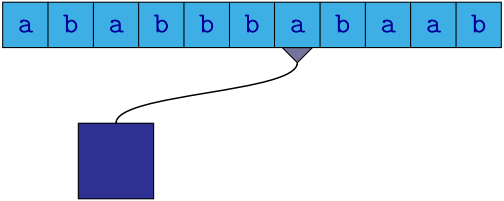


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

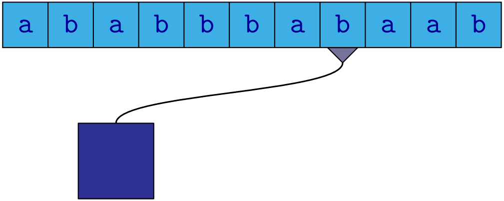


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{a, b\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů b.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

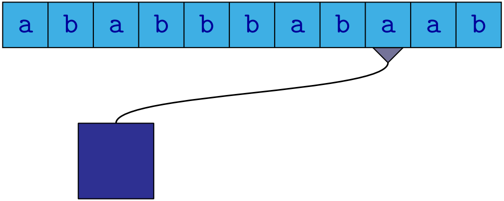


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

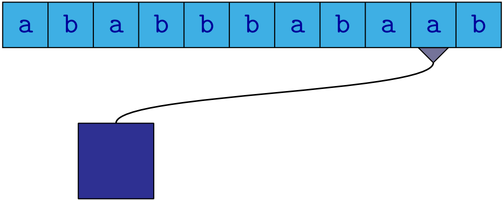


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

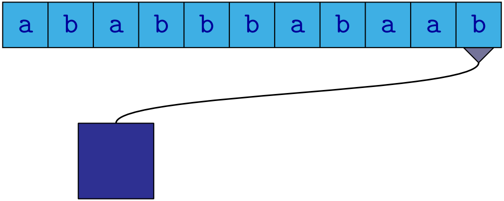


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.

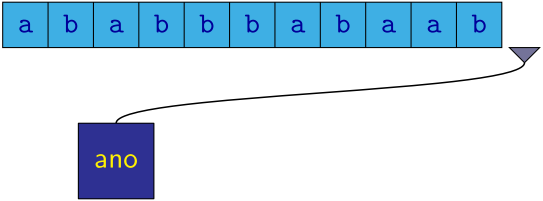


Rozpoznávání jazyka

Příklad: Uvažujme slova nad abecedou $\{\mathtt{a}, \mathtt{b}\}$.

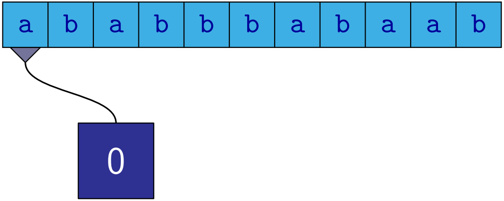
Chtěli bychom rozpoznávat jazyk L, který je tvořen slovy, ve kterých se vyskytuje sudý počet symbolů $\mathtt{b}$.

Chceme navrhnout zařízení, které přečte slovo, a sdělí nám, zda toto slovo patří do jazyka L či ne.



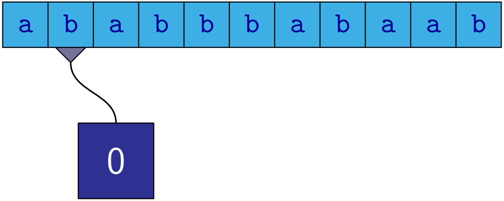
Rozpoznávání jazyka

První nápad: Počítat počet výskytů symbolů b.



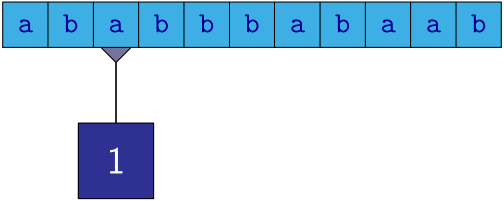
Rozpoznávání jazyka

První nápad: Počítat počet výskytů symbolů `b`.



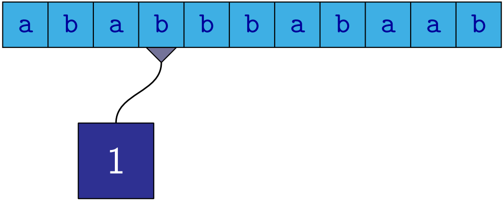
Rozpoznávání jazyka

První nápad: Počítat počet výskytů symbolů `b`.



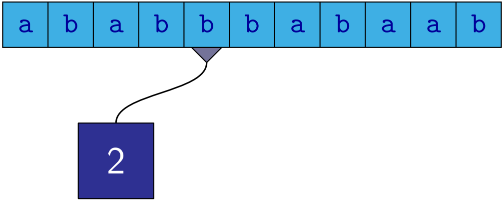
Rozpoznávání jazyka

První nápad: Počítat počet výskytů symbolů `b`.



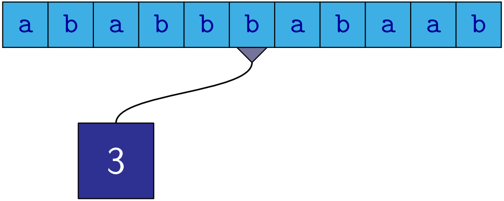
Rozpoznávání jazyka

První nápad: Počítat počet výskytů symbolů `b`.

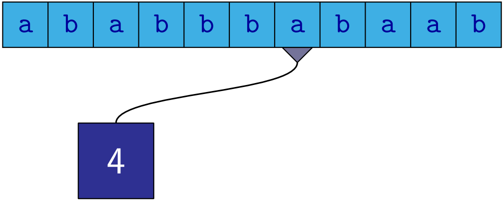


Rozpoznávání jazyka

První nápad: Počítat počet výskytů symbolů b.

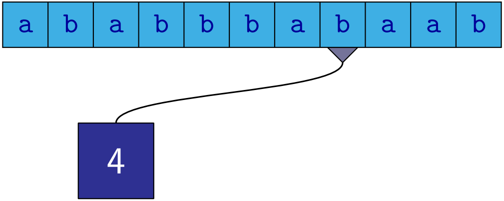


První nápad: Počítat počet výskytů symbolů b.



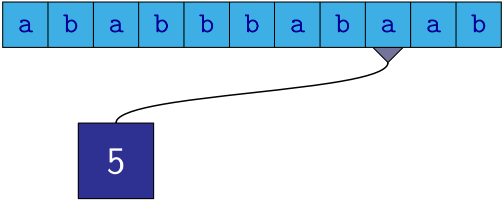
Rozpoznávání jazyka

První nápad: Počítat počet výskytů symbolů b.



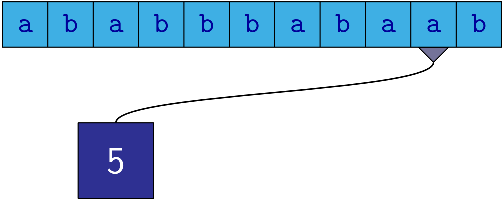
Rozpoznávání jazyka

První nápad: Počítat počet výskytů symbolů `b`.

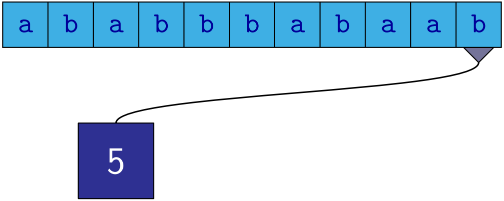


Rozpoznávání jazyka

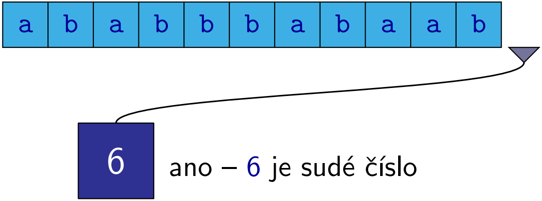
První nápad: Počítat počet výskytů symbolů `b`.



První nápad: Počítat počet výskytů symbolů `b`.



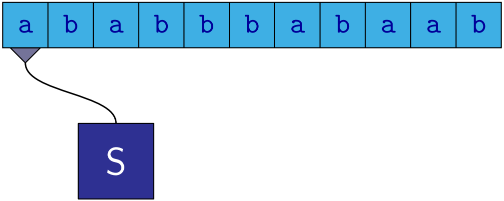
První nápad: Počítat počet výskytů symbolů b.



ano – 6 je sudé číslo

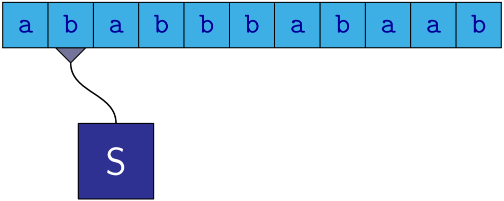
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů `b` je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



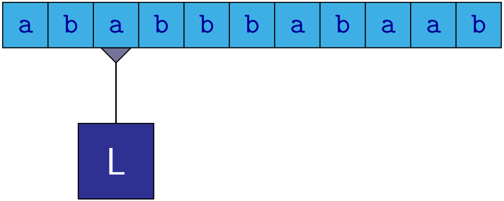
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů b je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



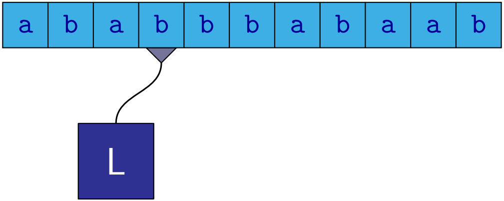
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů `b` je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



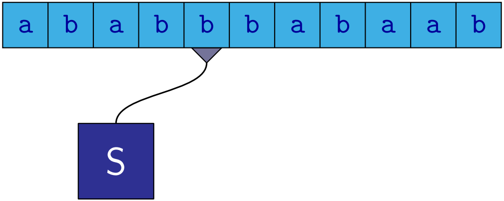
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů `b` je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



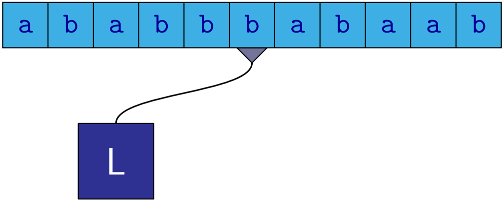
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů `b` je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



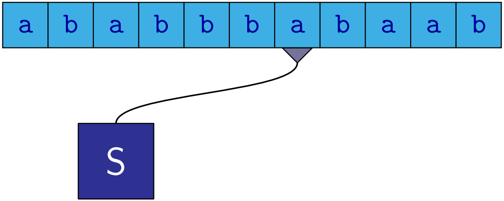
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů b je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).

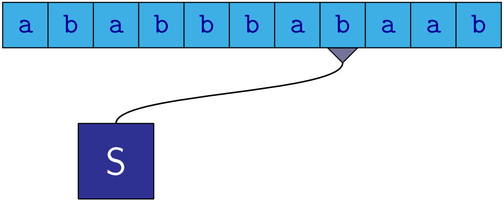


Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů `b` je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).

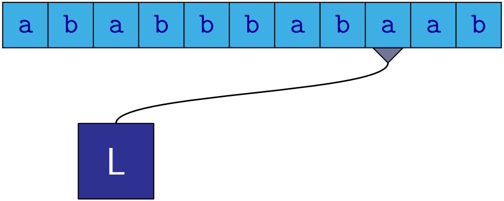


Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů `b` je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



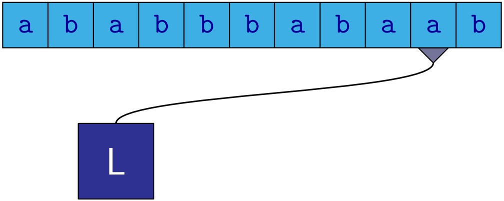
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů `b` je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



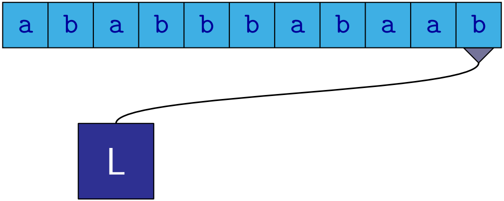
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů b je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



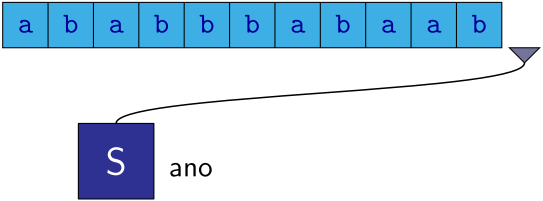
Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů `b` je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



Rozpoznávání jazyka

Druhý nápad: Ve skutečnosti nás zajímá pouze, zda počet dosud přečtených symbolů b je sudý nebo lichý (tj. místo čísla si stačí pamatovat jen jeho poslední bit).



Rozpoznávání jazyka

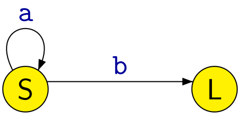
Chovaní tohoto zařízení můžeme popsat grafem:

Rozpoznávání jazyka

Chovaní tohoto zařízení můžeme popsat grafem:

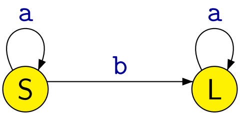
Rozpoznávání jazyka

Chovaní tohoto zařízení můžeme popsat grafem:



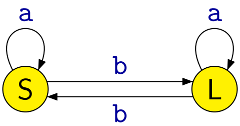
Rozpoznávání jazyka

Chovaní tohoto zařízení můžeme popsat grafem:

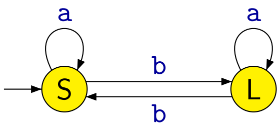


Rozpoznávání jazyka

Chovaní tohoto zařízení můžeme popsat grafem:

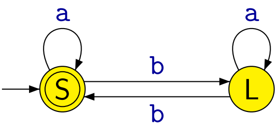


Chovaní tohoto zařízení můžeme popsat grafem:



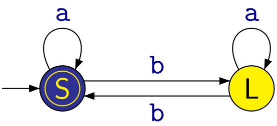
Rozpoznávání jazyka

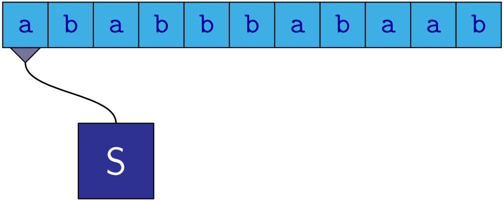
Chovaní tohoto zařízení můžeme popsat grafem:



Rozpoznávání jazyka

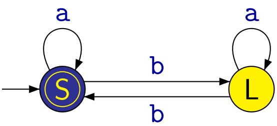
Chovaní tohoto zařízení můžeme popsat grafem:

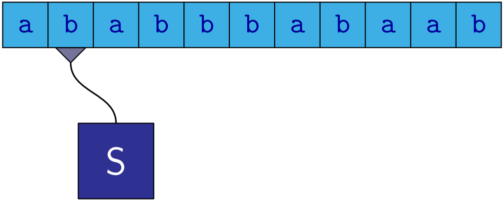




Rozpoznávání jazyka

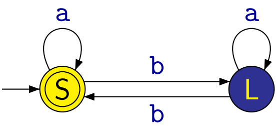
Chovaní tohoto zařízení můžeme popsat grafem:

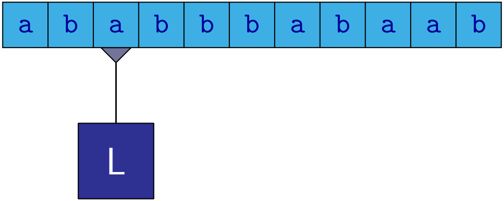




Rozpoznávání jazyka

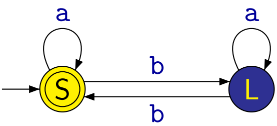
Chovaní tohoto zařízení můžeme popsat grafem:

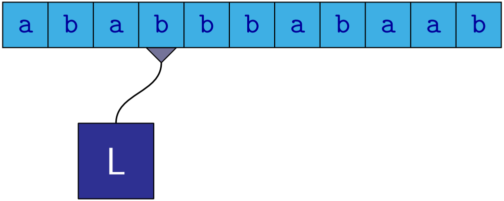




Rozpoznávání jazyka

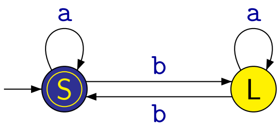
Chovaní tohoto zařízení můžeme popsat grafem:

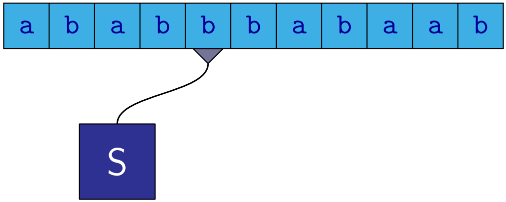




Rozpoznávání jazyka

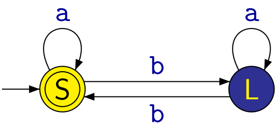
Chovaní tohoto zařízení můžeme popsat grafem:

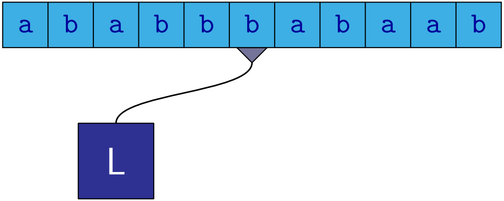




Rozpoznávání jazyka

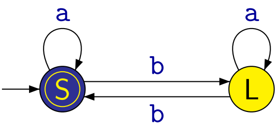
Chovaní tohoto zařízení můžeme popsat grafem:

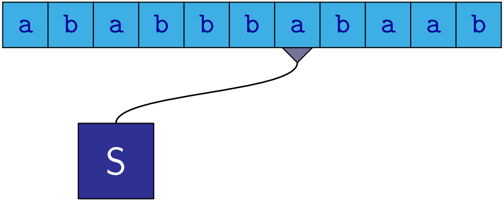




Rozpoznávání jazyka

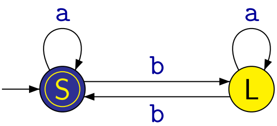
Chovaní tohoto zařízení můžeme popsat grafem:

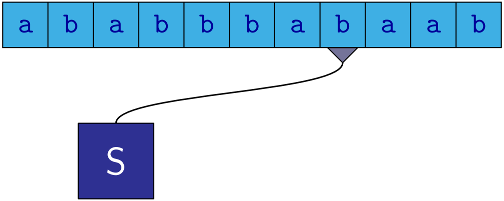




Rozpoznávání jazyka

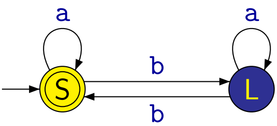
Chovaní tohoto zařízení můžeme popsat grafem:

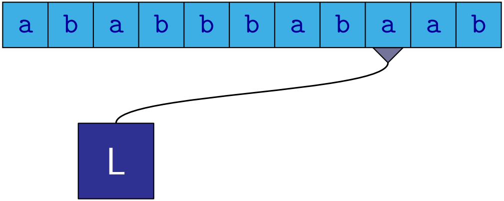




Rozpoznávání jazyka

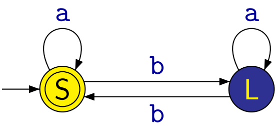
Chovaní tohoto zařízení můžeme popsat grafem:

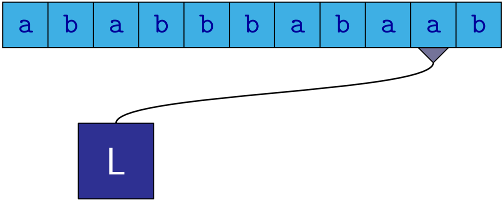




Rozpoznávání jazyka

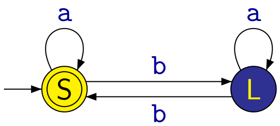
Chovaní tohoto zařízení můžeme popsat grafem:

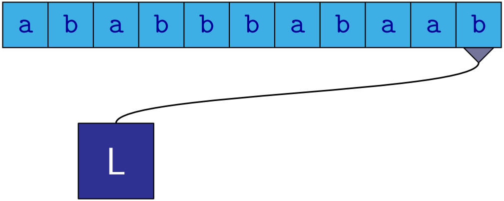




Rozpoznávání jazyka

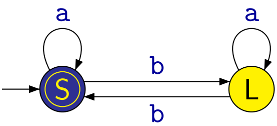
Chovaní tohoto zařízení můžeme popsat grafem:

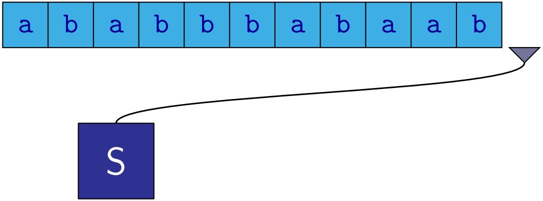




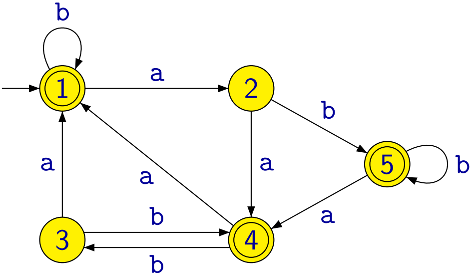
Rozpoznávání jazyka

Chovaní tohoto zařízení můžeme popsat grafem:





Deterministický konečný automat



Deterministický konečný automat se skládá ze **stavů** a **přechodů**. Jeden ze stavů je označen jako **počáteční stav** a některé ze stavů jsou označeny jako **přijímající**.

Deterministický konečný automat

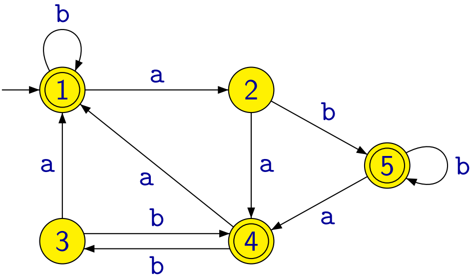
Formálně je **deterministický konečný automat** (**DKA**) definován jako pětice

$$(Q, \Sigma, \delta, q_0, F)$$

kde:

- Q je neprázdná konečná množina **stavů**
- Σ je **abeceda** (neprázdná konečná množina symbolů)
- $\delta : Q \times \Sigma \to Q$ je **přechodová funkce**
- $q_0 \in Q$ je **počáteční stav**
- $F \subseteq Q$ je množina **přijímajících stavů**

Deterministický konečný automat



- $Q = \{1, 2, 3, 4, 5\}$
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$
- $q_0 = 1$
- $F = \{1, 4, 5\}$

$$
\begin{array}{ll}
\delta(1, \mathtt{a}) = 2 & \delta(1, \mathtt{b}) = 1 \\
\delta(2, \mathtt{a}) = 4 & \delta(2, \mathtt{b}) = 5 \\
\delta(3, \mathtt{a}) = 1 & \delta(3, \mathtt{b}) = 4 \\
\delta(4, \mathtt{a}) = 1 & \delta(4, \mathtt{b}) = 3 \\
\delta(5, \mathtt{a}) = 4 & \delta(5, \mathtt{b}) = 5
\end{array}
$$

Deterministický konečný automat

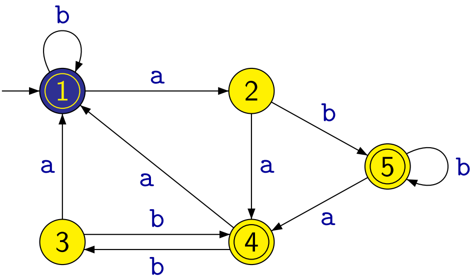
Místo zápisu

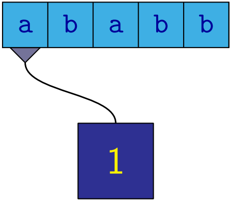
$$
\begin{array}{ll}
\delta(1, \mathtt{a}) = 2 & \delta(1, \mathtt{b}) = 1 \\
\delta(2, \mathtt{a}) = 4 & \delta(2, \mathtt{b}) = 5 \\
\delta(3, \mathtt{a}) = 1 & \delta(3, \mathtt{b}) = 4 \\
\delta(4, \mathtt{a}) = 1 & \delta(4, \mathtt{b}) = 3 \\
\delta(5, \mathtt{a}) = 4 & \delta(5, \mathtt{b}) = 5
\end{array}
$$

budeme raději používat stručnější tabulku nebo grafické znázornění:

δ	a	b
$\leftrightarrow 1$	2	1
2	4	5
3	1	4
$\leftarrow 4$	1	3
$\leftarrow 5$	4	5

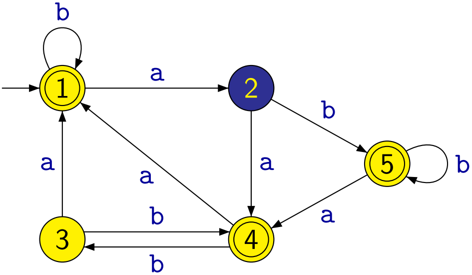
Deterministický konečný automat

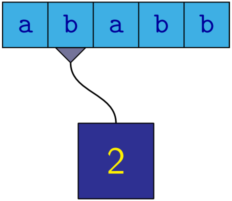




1

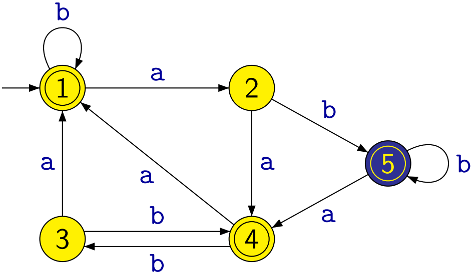
Deterministický konečný automat

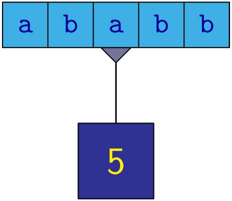




$$1 \xrightarrow{a} 2$$

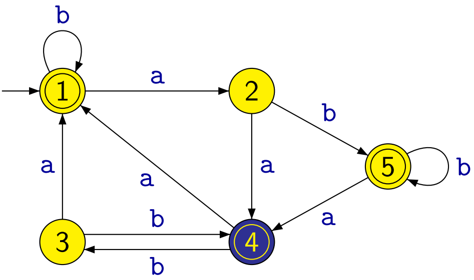
Deterministický konečný automat

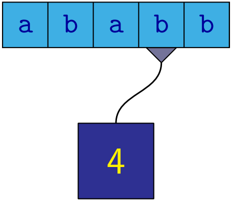




$$1 \xrightarrow{a} 2 \xrightarrow{b} 5$$

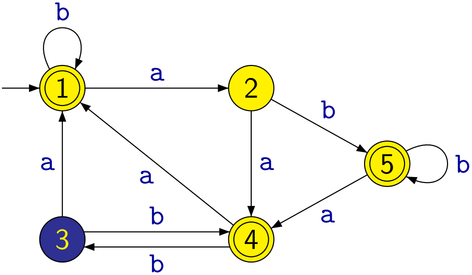
Deterministický konečný automat

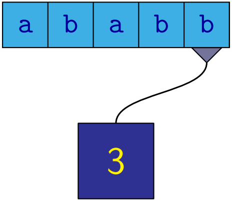




$$1 \xrightarrow{a} 2 \xrightarrow{b} 5 \xrightarrow{a} 4$$

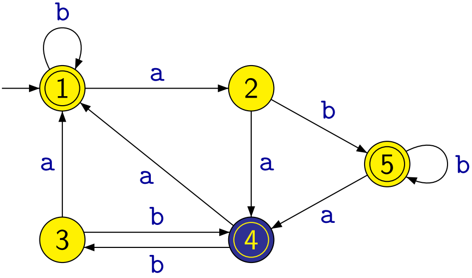
Deterministický konečný automat

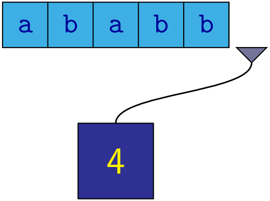




$$1 \xrightarrow{a} 2 \xrightarrow{b} 5 \xrightarrow{a} 4 \xrightarrow{b} 3$$

Deterministický konečný automat





$$1 \xrightarrow{a} 2 \xrightarrow{b} 5 \xrightarrow{a} 4 \xrightarrow{b} 3 \xrightarrow{b} 4$$

Deterministický konečný automat

Definice

Mějme DKA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$.

Zápisem $q \xrightarrow{w} q'$, kde $q, q' \in Q$ a $w \in \Sigma^*$, budeme označovat to, že pokud je automat ve stavu q, tak přečtením slova w přejde do stavu q'.

Poznámka: $\longrightarrow \subseteq Q \times \Sigma^* \times Q$ je ternární relace.

Místo $(q, w, q') \in \longrightarrow$ píšeme $q \xrightarrow{w} q'$.

Pro DKA platí, že pro libovolný stav q a libovolné slovo w existuje právě jeden stav q' takový, že $q \xrightarrow{w} q'$.

Deterministický konečný automat

Relaci $\longrightarrow$ můžeme formálně definovat následující induktivní definicí:

- $q \xrightarrow{\varepsilon} q$ pro libovolné $q \in Q$
- Pro $w \in \Sigma^*$ a $a \in \Sigma$:

  $q \xrightarrow{wa} q'$ právě tehdy, když existuje $q'' \in Q$ takové, že

  $$q \xrightarrow{w} q'' \text{ a } \delta(q'', a) = q'$$

Deterministický konečný automat

	a	b
$\leftrightarrow 1$	2	1
2	4	5
3	1	4
$\leftarrow 4$	1	3
$\leftarrow 5$	4	5

$$1 \xrightarrow{\varepsilon} 1 \qquad \delta(1, \mathtt{a}) = 2$$

$$1 \xrightarrow{\mathtt{a}} 2 \qquad \delta(2, \mathtt{b}) = 5$$

$$1 \xrightarrow{\mathtt{ab}} 5 \qquad \delta(5, \mathtt{a}) = 4$$

$$1 \xrightarrow{\mathtt{aba}} 4 \qquad \delta(4, \mathtt{b}) = 3$$

$$1 \xrightarrow{\mathtt{abab}} 3 \qquad \delta(3, \mathtt{b}) = 4$$

$$1 \xrightarrow{\mathtt{ababb}} 4$$

Deterministický konečný automat

Slovo $w \in \Sigma^*$ je **přijímáno** deterministickým konečným automatem $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ právě tehdy, když existuje stav $q \in F$ takový, že $q_0 \xrightarrow{w} q$.

Definice

Jazyk rozpoznávaný (přijímaný) daným deterministickým konečným automatem $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, označovaný $\mathcal{L}(\mathcal{A})$, je množina všech slov přijímaných tímto automatem, tj.

$$\mathcal{L}(\mathcal{A}) = \{w \in \Sigma^* \mid \exists q \in F : q_0 \xrightarrow{w} q\}$$

Regulární jazyky

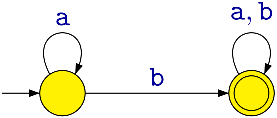
Definice

Jazyk L je **regulární** právě tehdy, když existuje nějaký deterministický konečný automat $\mathcal{A}$, který jej přijímá, tj. DKA $\mathcal{A}$ takový, že $\mathcal{L}(\mathcal{A}) = L$.

Příklady deterministických konečných automatů

Příklad: Automat rozpoznávající jazyk L nad abecedou $\{\mathtt{a}, \mathtt{b}\}$ tvořený slovy, která obsahují alespoň jeden výskyt symbolu $\mathtt{b}$, tj.

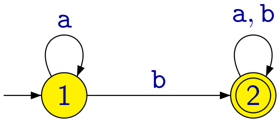
$$L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* \mid |w|_\mathtt{b} \geq 1\}$$



Příklady deterministických konečných automatů

Příklad: Automat rozpoznávající jazyk L nad abecedou $\{\mathtt{a}, \mathtt{b}\}$ tvořený slovy, která obsahují alespoň jeden výskyt symbolu $\mathtt{b}$, tj.

$$L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* \mid |w|_{\mathtt{b}} \geq 1\}$$

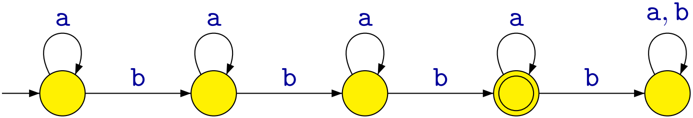


	a	b
$\rightarrow 1$	1	2
$\leftarrow 2$	2	2

Příklady deterministických konečných automatů

Příklad: Automat rozpoznávající jazyk L nad abecedou $\{\mathtt{a}, \mathtt{b}\}$ tvořený slovy, která obsahují právě tři výskyty symbolu $\mathtt{b}$, tj.

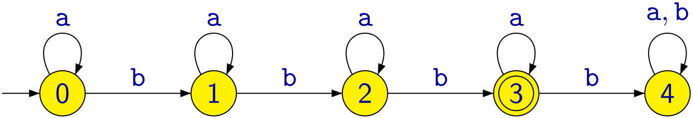
$$L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* \mid |w|_{\mathtt{b}} = 3\}$$



Příklady deterministických konečných automatů

Příklad: Automat rozpoznávající jazyk L nad abecedou $\{\mathtt{a}, \mathtt{b}\}$ tvořený slovy, která obsahují právě tři výskyty symbolu $\mathtt{b}$, tj.

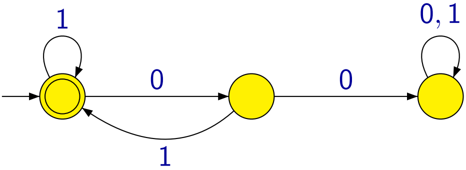
$$L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* \mid |w|_{\mathtt{b}} = 3\}$$



	a	b
$\rightarrow$ 0	0	1
1	1	2
2	2	3
$\leftarrow$ 3	3	4
4	4	4

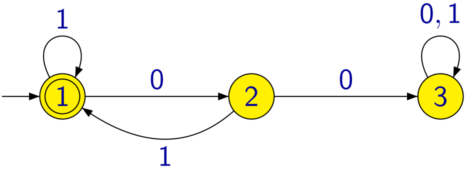
Příklady deterministických konečných automatů

Příklad: Automat rozpoznávající jazyk nad abecedou $\{0, 1\}$ tvořený slovy, kde každý výskyt symbolu 0 je bezprostředně následován symbolem 1.



Příklady deterministických konečných automatů

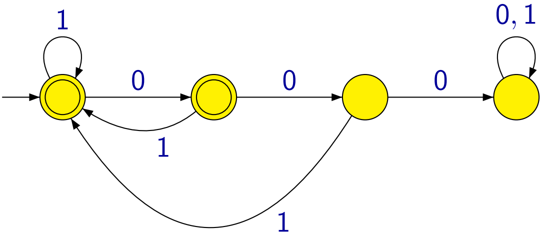
Příklad: Automat rozpoznávající jazyk nad abecedou $\{0, 1\}$ tvořený slovy, kde každý výskyt symbolu 0 je bezprostředně následován symbolem 1.



	0	1
$\leftrightarrow 1$	2	1
2	3	1
3	3	3

Příklady deterministických konečných automatů

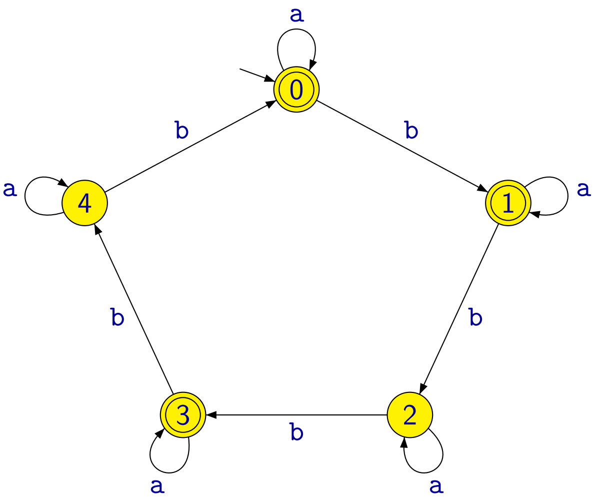
Příklad: Automat rozpoznávající jazyk nad abecedou $\{0, 1\}$ tvořený slovy, kde každá dvojice symbolů 0 je bezprostředně následována symbolem 1.



Příklady deterministických konečných automatů

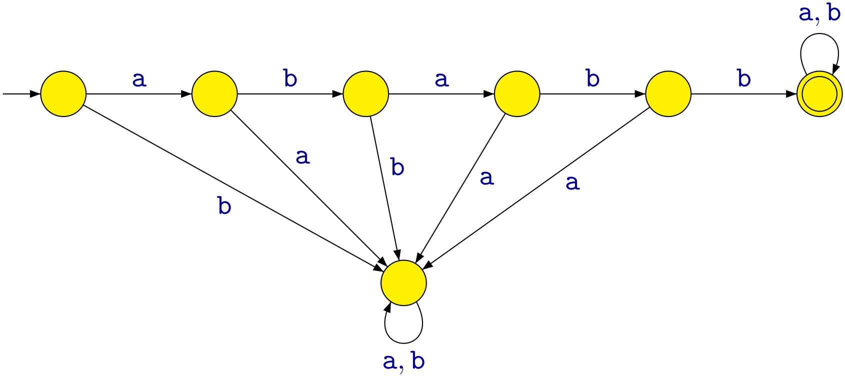
Příklad: Automat rozpoznávající jazyk

$$L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* \mid (|w|_{\mathtt{b}} \bmod 5) \in \{0, 1, 3\}\}$$



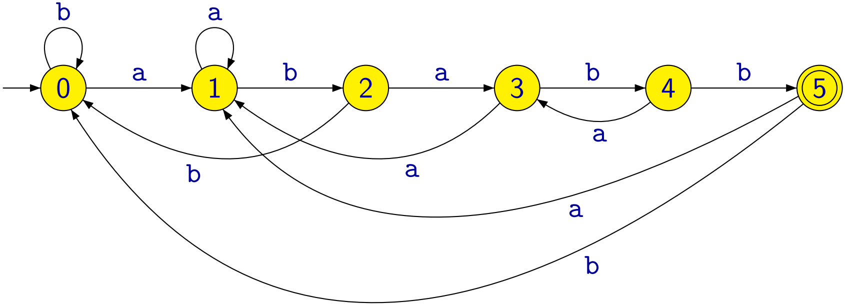
Příklady deterministických konečných automatů

Příklad: Automat rozpoznávající jazyk nad abecedou $\{\mathtt{a},\mathtt{b}\}$ tvořený slovy, která začínají **prefixem** $\mathtt{ababb}$.



Příklady deterministických konečných automatů

Příklad: Automat rozpoznávající jazyk nad abecedou $\{\mathtt{a}, \mathtt{b}\}$ tvořený slovy, která končí **sufixem** `ababb`.



Příklady deterministických konečných automatů

Konstrukce tohoto automatu je založena na následující myšlence:

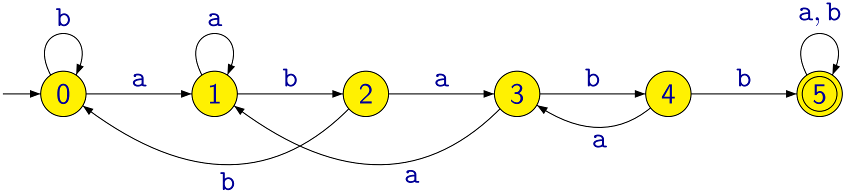
- Předpokládejme, že chceme vyhledávat slovo u délky n (tj. $|u| = n$). Stavy automatu jsou označeny čísly $0, 1, \ldots, n$.
- Stav s číslem i odpovídá situaci, kdy i je délka nejdelšího slova, které je zároveň:
  - prefixem hledaného vzorku u
  - sufixem té části vstupního slova, kterou automat zatím přečetl

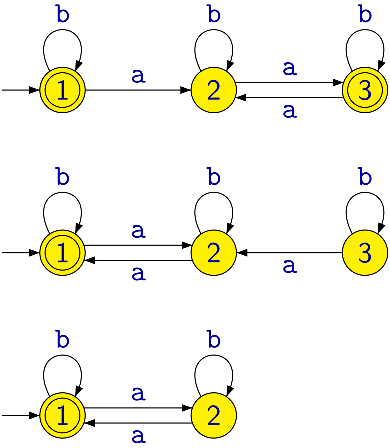
Například pro slovo `ababb` stavy automatu odpovídají následujícím slovům:

- Stav 0 … ε
- Stav 1 … `a`
- Stav 2 … `ab`
- Stav 3 … `aba`
- Stav 4 … `abab`
- Stav 5 … `ababb`

Příklady deterministických konečných automatů

Příklad: Automat rozpoznávající jazyk nad abecedou $\{\mathtt{a}, \mathtt{b}\}$ tvořený slovy, která obsahují **podslovo** `ababb`.





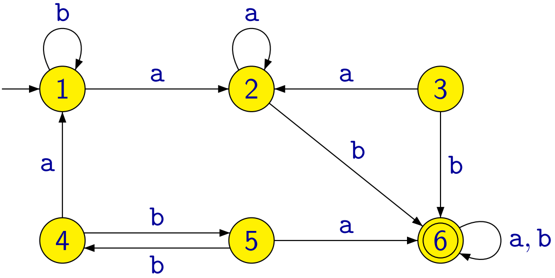
Všechny tři automaty přijímají jazyk všech slov se sudým počtem a.

Ekvivalence automatů

Definice

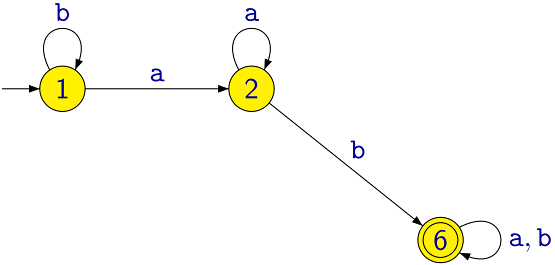
O konečných automatech $\mathcal{A}_1$, $\mathcal{A}_2$ řekneme, že jsou **ekvivalentní**, jestliže $\mathcal{L}(\mathcal{A}_1) = \mathcal{L}(\mathcal{A}_2)$.

Nedosažitelné stavy automatu



- Automat přijímá jazyk $L = \{w \in \{\texttt{a}, \texttt{b}\}^* \mid w \text{ obsahuje podslovo } \texttt{ab}\}$
- Pro žádnou posloupnost vstupních symbolů se automat nedostane do stavů 3, 4 nebo 5.

Nedosažitelné stavy automatu



- Automat přijímá jazyk $L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* \mid w \text{ obsahuje podslovo } \mathtt{ab}\}$
- Pro žádnou posloupnost vstupních symbolů se automat nedostane do stavů 3, 4 nebo 5.
- Pokud tyto stavy odstraníme, pořád automat přijímá stejný jazyk L.

Nedosažitelné stavy automatu

Definice

Stav q konečného automatu $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ je **dosažitelný** pokud existuje nějaké slovo w takové, že $q_0 \xrightarrow{w} q$.

V opačném případě stav nazýváme **nedosažitelný**.

- Do nedosažitelných stavů nevede v grafu automatu žádná orientovaná cesta z počátečního stavu.
- Nedosažitelné stavy můžeme z automatu odstranit (spolu se všemi přechody vedoucími do nich a z nich). Jazyk přijímaný automatem se nezmění.

Automaty a operace na jazycích

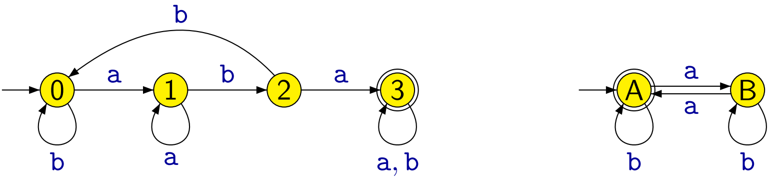
Při konstrukci automatů může být obtížné přímo zkonstruovat automat pro daný jazyk L.

Pokud je možné jazyk L popsat jako výsledek nějakých jazykových operací (průnik, sjednocení, doplněk, zřetězení, iterace, ...) aplikovaných na nějaké jednodušší jazyky L_1 a L_2, může být výhodné postupovat modulárním způsobem:

- Nejprve zkonstruovat automaty pro jazyky L_1 a L_2.
- Poté použít některou z obecných konstrukcí, které umožňují k daným automatům rozpoznávajícím jazyky L_1 a L_2 algoritmicky zkonstruovat automat pro jazyk L, který je výsledkem aplikace dané jazykové operace na jazyky L_1 a L_2.

Automat pro průnik jazyků

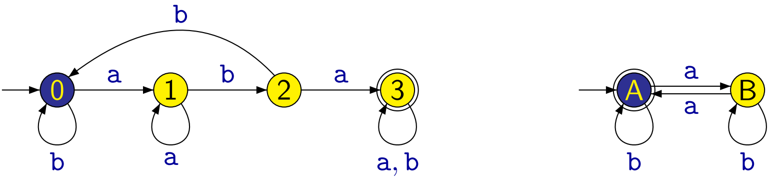
Máme následující dva automaty:



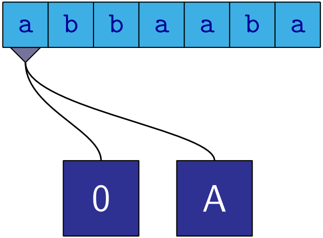
Příjmou oba slovo abbaaba?

Automat pro průnik jazyků

Máme následující dva automaty:

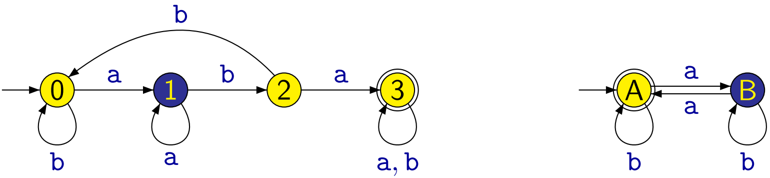


Příjmou oba slovo abbaaba?

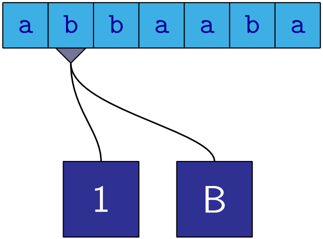


Automat pro průnik jazyků

Máme následující dva automaty:

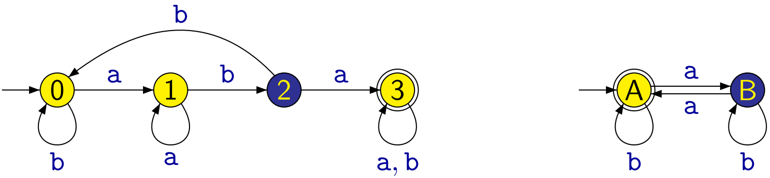


Příjmou oba slovo abbaaba?

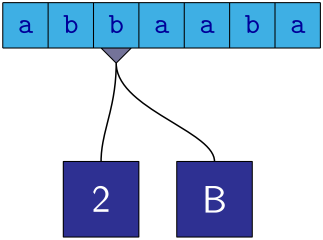


Automat pro průnik jazyků

Máme následující dva automaty:

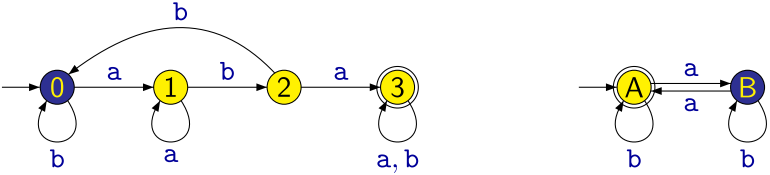


Příjmou oba slovo abbaaba?

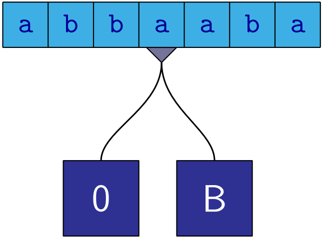


Automat pro průnik jazyků

Máme následující dva automaty:

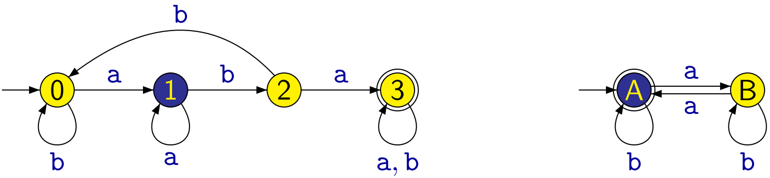


Příjmou oba slovo abbaaba?

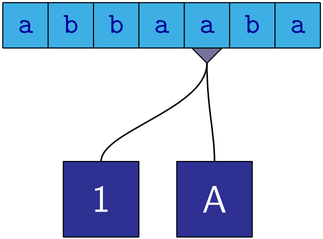


Automat pro průnik jazyků

Máme následující dva automaty:

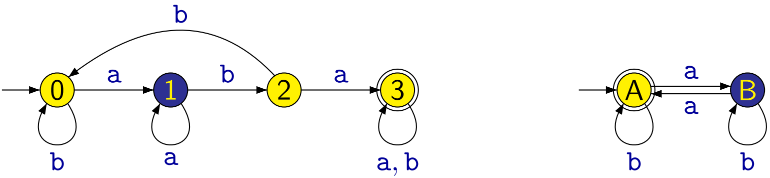


Příjmou oba slovo abbaaba?

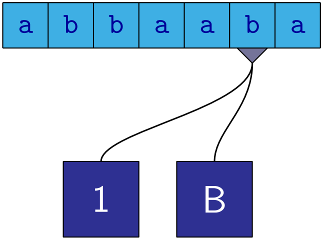


Automat pro průnik jazyků

Máme následující dva automaty:

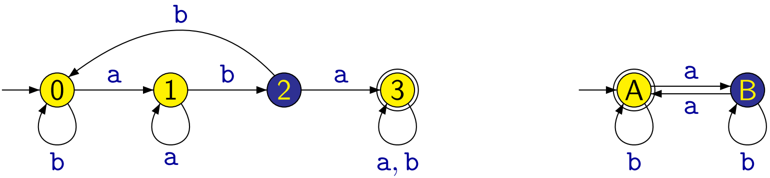


Příjmou oba slovo abbaaba?

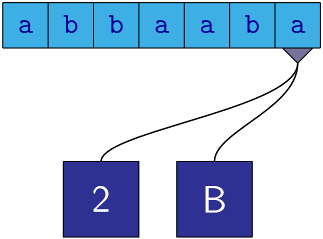


Automat pro průnik jazyků

Máme následující dva automaty:

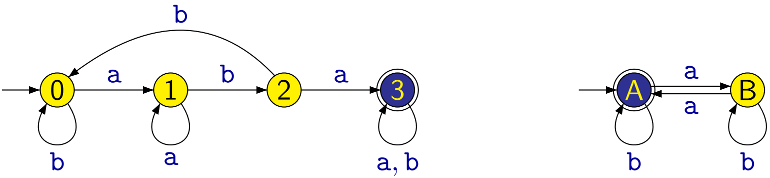


Příjmou oba slovo abbaaba?

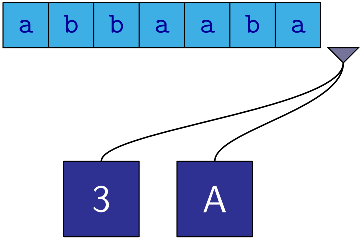


Automat pro průnik jazyků

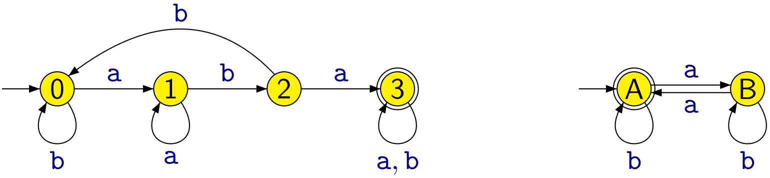
Máme následující dva automaty:



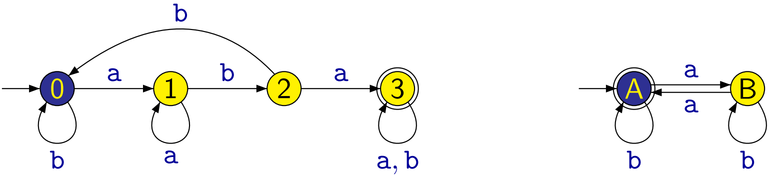
Příjmou oba slovo `abbaaba`?

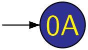


Automat pro průnik jazyků

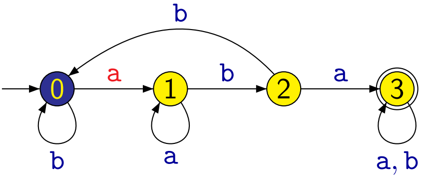


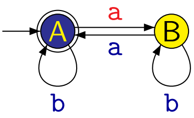
Automat pro průnik jazyků

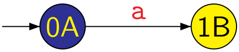




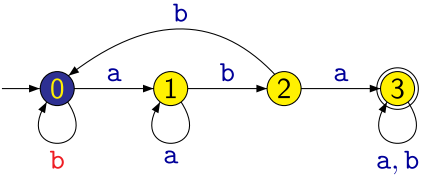
Automat pro průnik jazyků

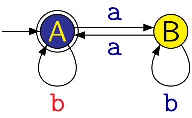


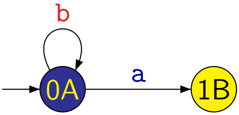




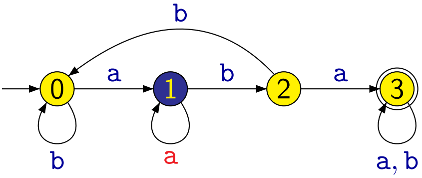
Automat pro průnik jazyků

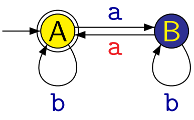


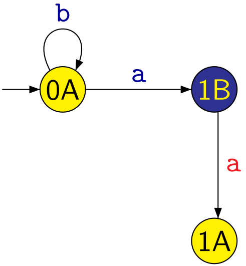




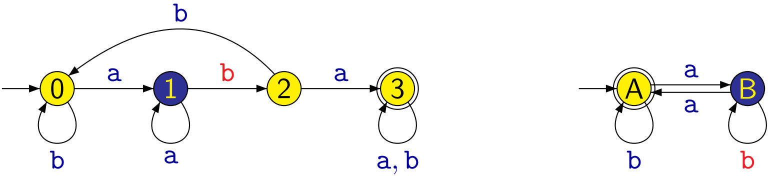
Automat pro průnik jazyků

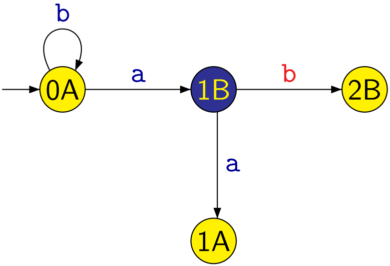




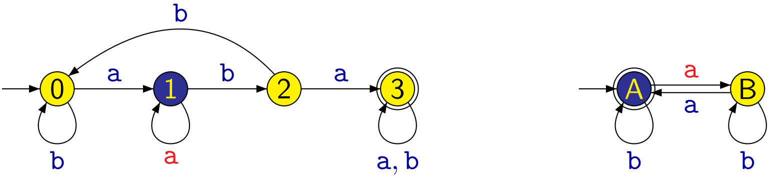


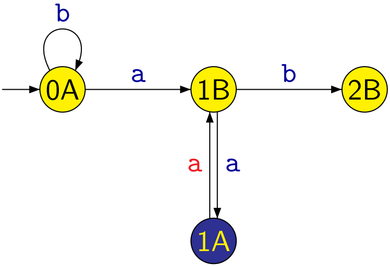
Automat pro průnik jazyků



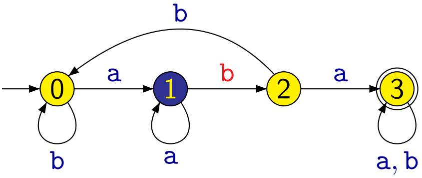


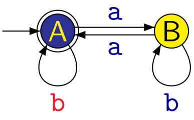
Automat pro průnik jazyků

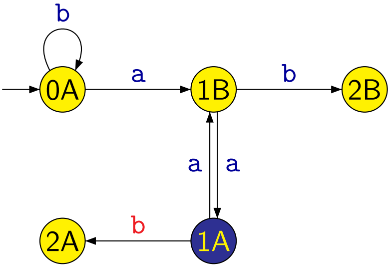




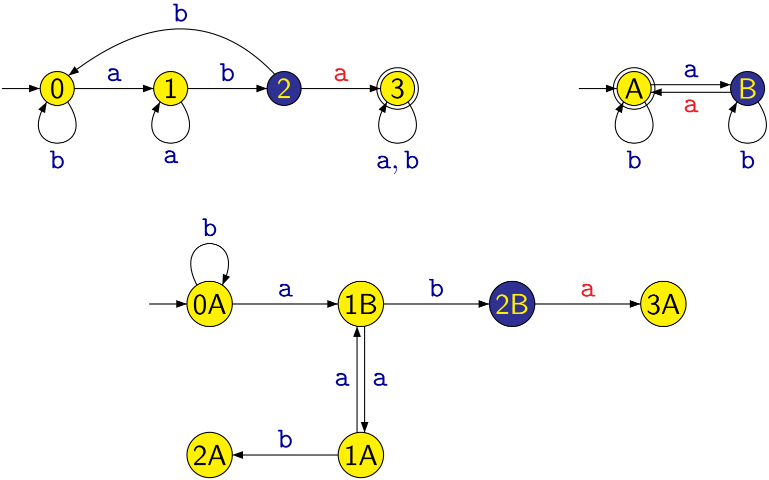
Automat pro průnik jazyků



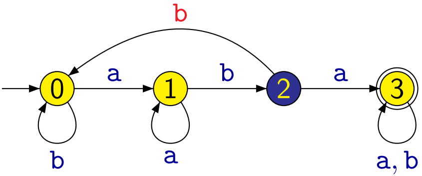


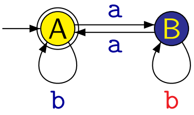


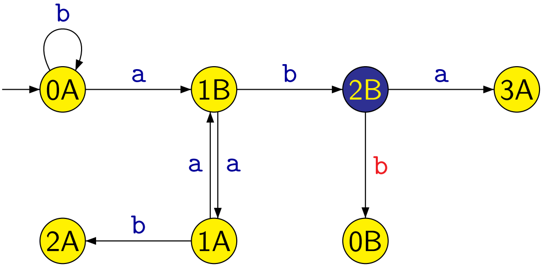
Automat pro průnik jazyků



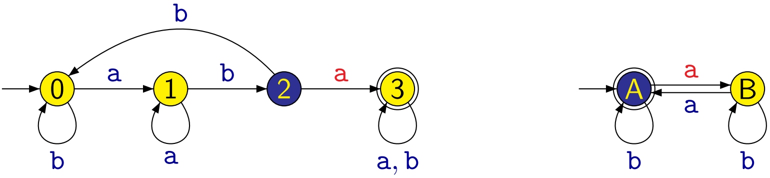
Automat pro průnik jazyků

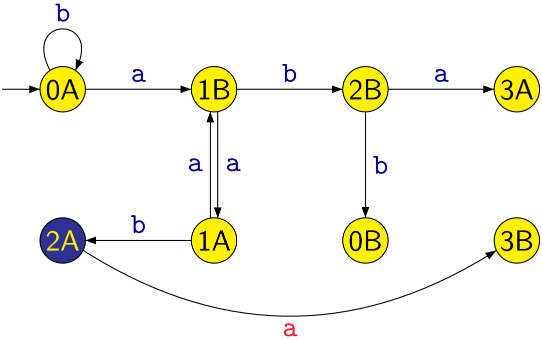




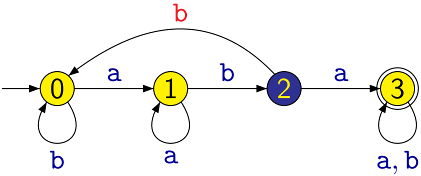


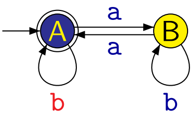
Automat pro průnik jazyků

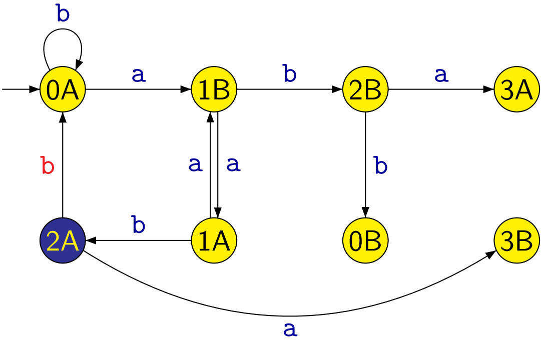




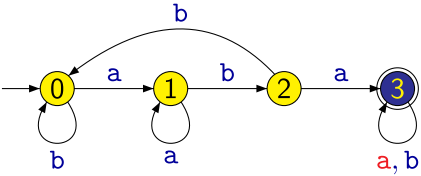
Automat pro průnik jazyků

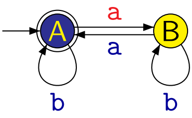


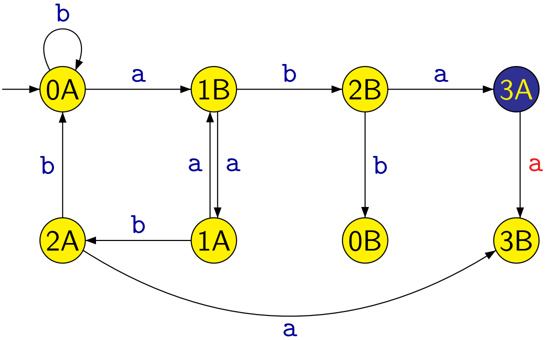




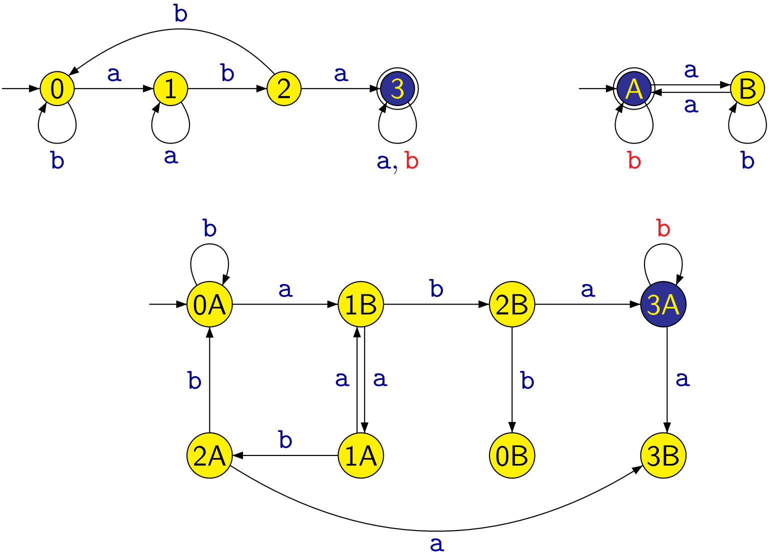
Automat pro průnik jazyků



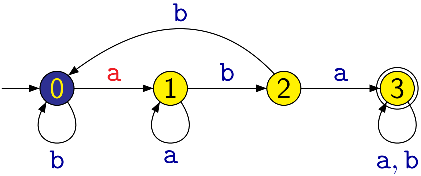


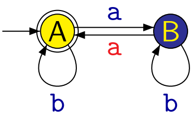


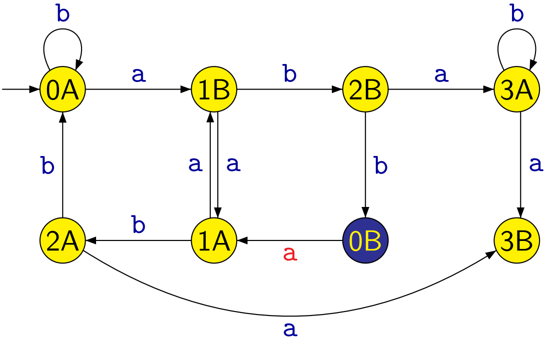
Automat pro průnik jazyků



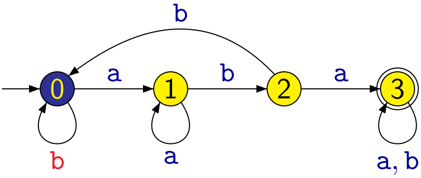
Automat pro průnik jazyků

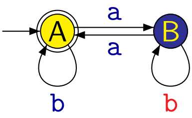


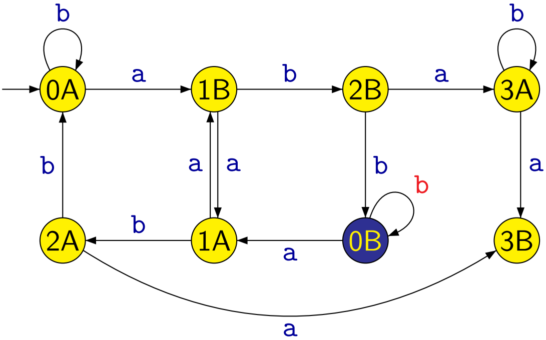




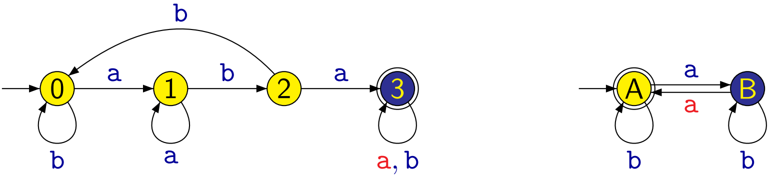
Automat pro průnik jazyků

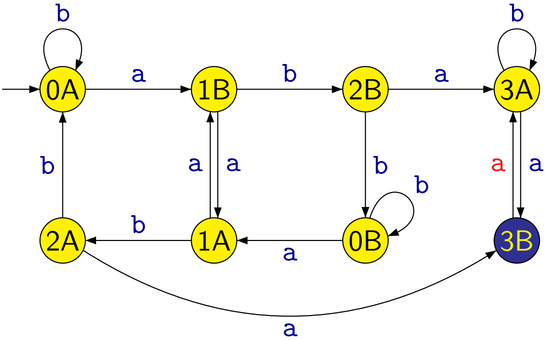




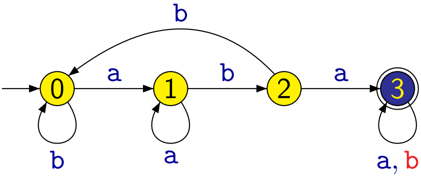


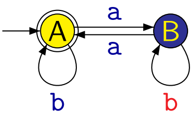
Automat pro průnik jazyků

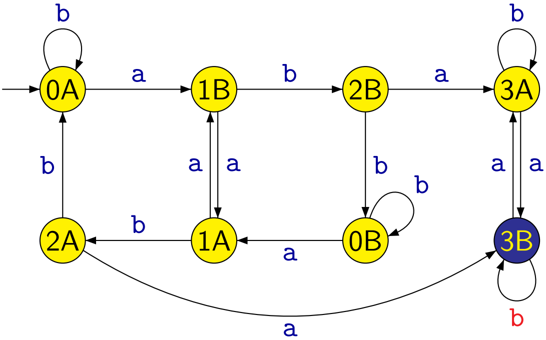




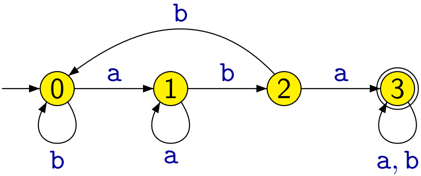
Automat pro průnik jazyků

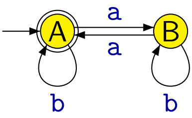


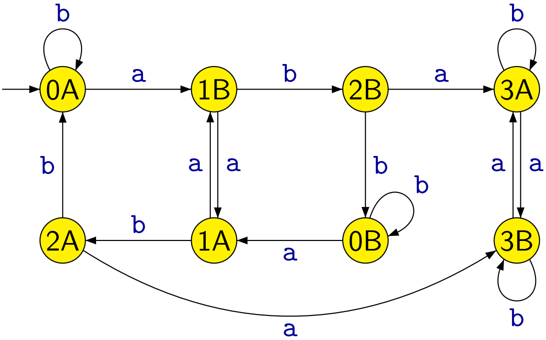




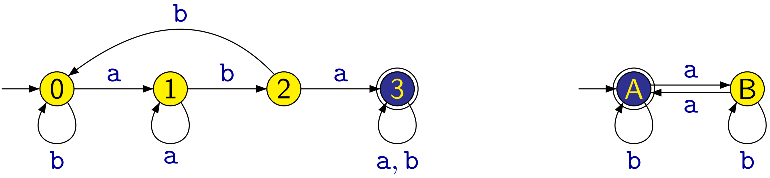
Automat pro průnik jazyků

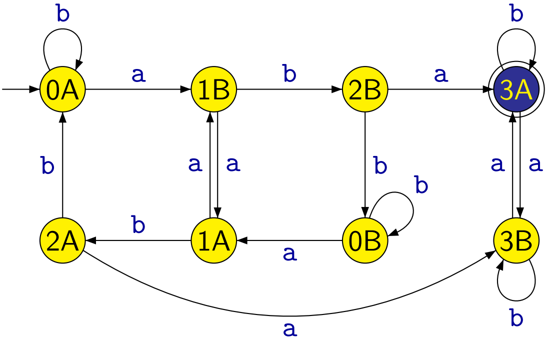




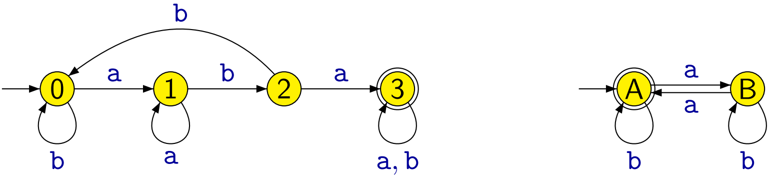


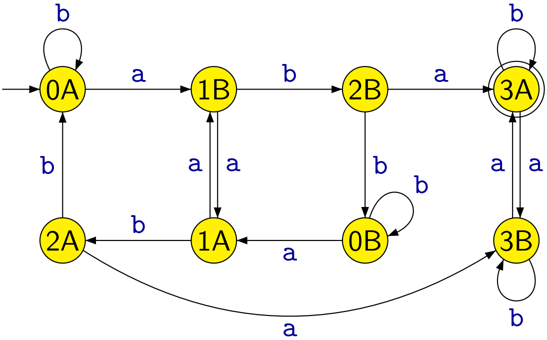
Automat pro průnik jazyků





Automat pro průnik jazyků





Automat pro průnik jazyků

Formálně můžeme popsat tuto konstrukci následovně:

Předpokládáme, že máme dva deterministické konečné automaty $\mathcal{A}_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$ a $\mathcal{A}_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$.

K nim setrojíme DKA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ kde:

- $Q = Q_1 \times Q_2$
- $\delta(\,(q_1, q_2), a\,) = (\,\delta_1(q_1, a), \delta_2(q_2, a)\,)$ pro všechna $q_1 \in Q_1$, $q_2 \in Q_2$, $a \in \Sigma$
- $q_0 = (q_{01}, q_{02})$
- $F = F_1 \times F_2$

Není težké ověřit, že pro libovolné slovo $w \in \Sigma^*$ platí, že $w \in \mathcal{L}(\mathcal{A})$ právě tehdy, když $w \in \mathcal{L}(\mathcal{A}_1)$ a $w \in \mathcal{L}(\mathcal{A}_2)$, tj.

$$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$$

Průnik regulárních jazyků

Věta

Jestliže jazyky $L_1, L_2 \subseteq \Sigma^*$ jsou regulární, pak také jazyk $L_1 \cap L_2$ je regulární.

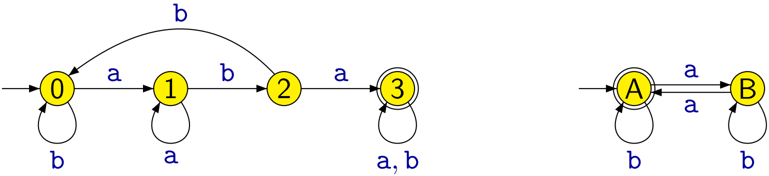
Důkaz: Předpokládejme, že $\mathcal{A}_1$ a $\mathcal{A}_2$ jsou deterministické konečné automaty takové, že

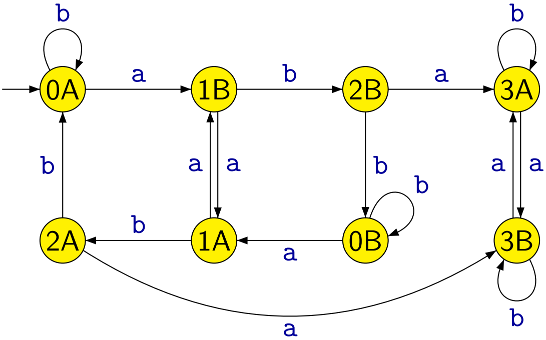
$$L_1 = \mathcal{L}(\mathcal{A}_1) \qquad\qquad L_2 = \mathcal{L}(\mathcal{A}_2)$$

Popsanou konstrukcí k nim můžeme sestrojit deterministický konečný automat $\mathcal{A}$ takový, že

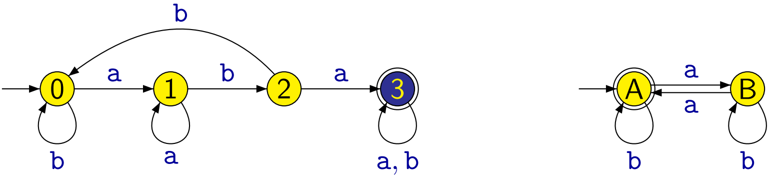
$$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2) = L_1 \cap L_2$$

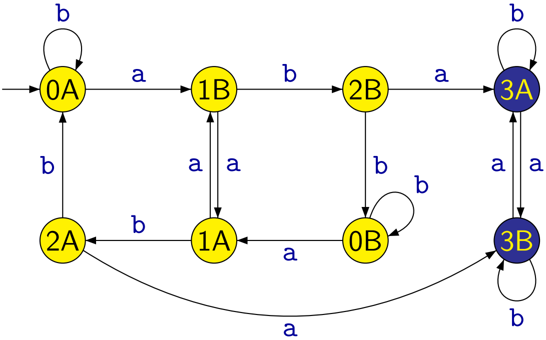
Automat pro sjednocení jazyků



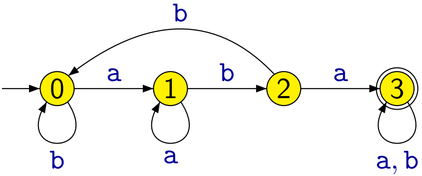


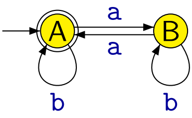
Automat pro sjednocení jazyků

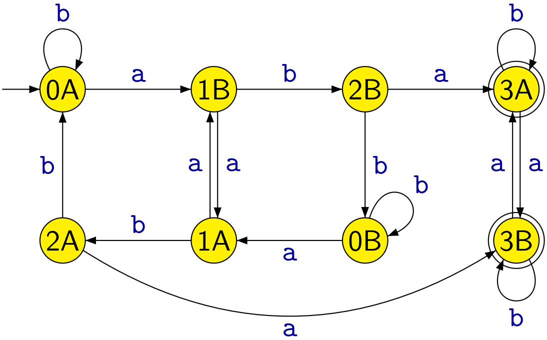




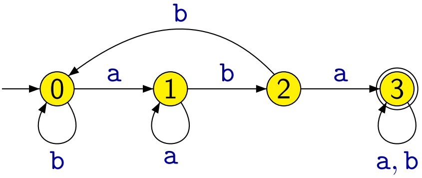
Automat pro sjednocení jazyků

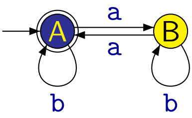


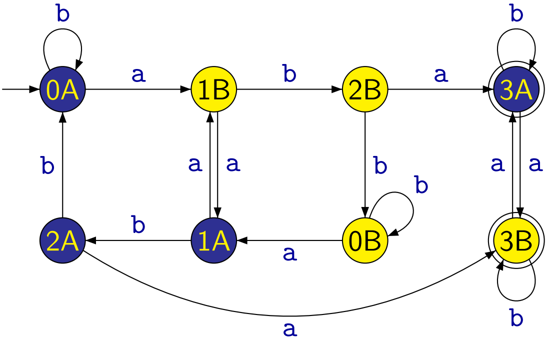




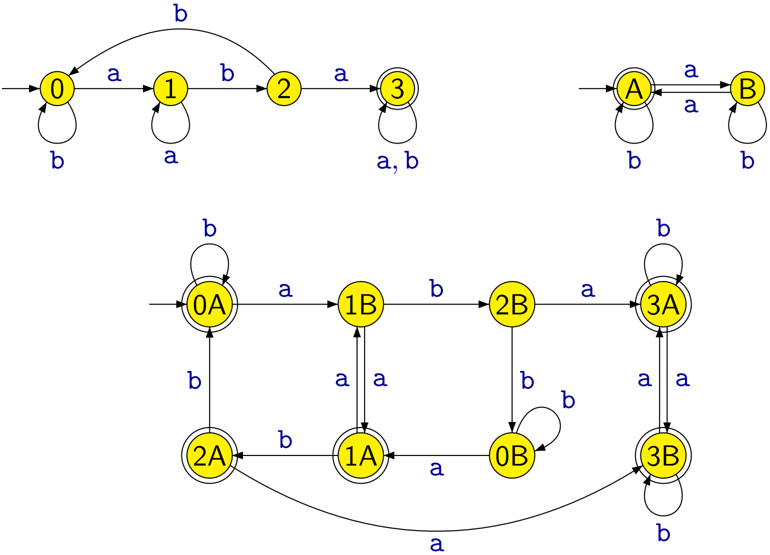
Automat pro sjednocení jazyků







Automat pro sjednocení jazyků



Sjednocení regulárních jazyků

Konstukce automatu $\mathcal{A}$, který přijímá **sjednocení** jazyků přijímaných automaty $\mathcal{A}_1$ a $\mathcal{A}_2$, tj. jazyk

$$\mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_1)$$

je téměř stejná jako v případě automatu přijímajícího $\mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$.

Jediný rozdíl je v definici množiny přijímajících stavů:

- $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$

Sjednocení regulárních jazyků

Konstukce automatu $\mathcal{A}$, který přijímá **sjednocení** jazyků přijímaných automaty $\mathcal{A}_1$ a $\mathcal{A}_2$, tj. jazyk

$$\mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_1)$$

je téměř stejná jako v případě automatu přijímajícího $\mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$.

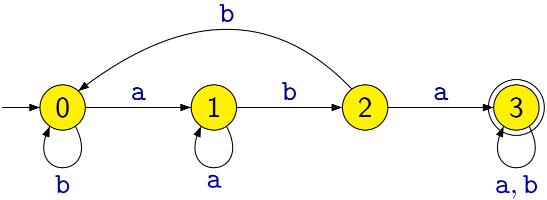
Jediný rozdíl je v definici množiny přijímajících stavů:

- $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$

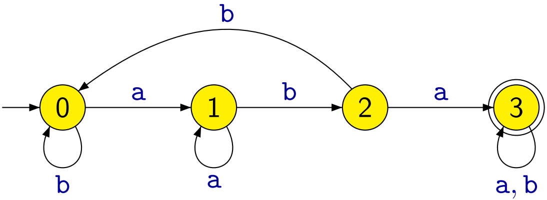
Věta

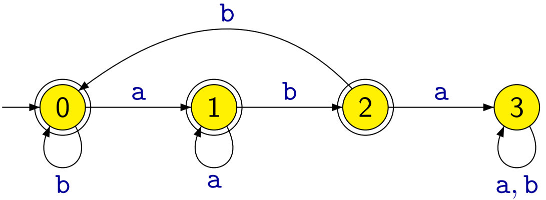
Jestliže jazyky $L_1, L_2 \subseteq \Sigma^*$ jsou regulární, pak také jazyk $L_1 \cup L_2$ je regulární.

Automat pro doplněk jazyka



Automat pro doplněk jazyka





Doplněk regulárního jazyka

K DKA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ sestrojíme DKA $\mathcal{A}' = (Q, \Sigma, \delta, q_0, Q - F)$.

Je očividné, že pro každé slovo $w \in \Sigma^*$ platí, že $w \in \mathcal{L}(\mathcal{A}')$ právě tehdy, když $w \notin \mathcal{L}(\mathcal{A})$, tj.

$$\mathcal{L}(\mathcal{A}') = \overline{\mathcal{L}(\mathcal{A})}$$

Doplněk regulárního jazyka

K DKA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ sestrojíme DKA $\mathcal{A}' = (Q, \Sigma, \delta, q_0, Q - F)$.

Je očividné, že pro každé slovo $w \in \Sigma^*$ platí, že $w \in \mathcal{L}(\mathcal{A}')$ právě tehdy, když $w \notin \mathcal{L}(\mathcal{A})$, tj.

$$\mathcal{L}(\mathcal{A}') = \overline{\mathcal{L}(\mathcal{A})}$$

Věta

Jestliže jazyk L je regulární, pak také jeho doplněk $\overline{L}$ je regulární.