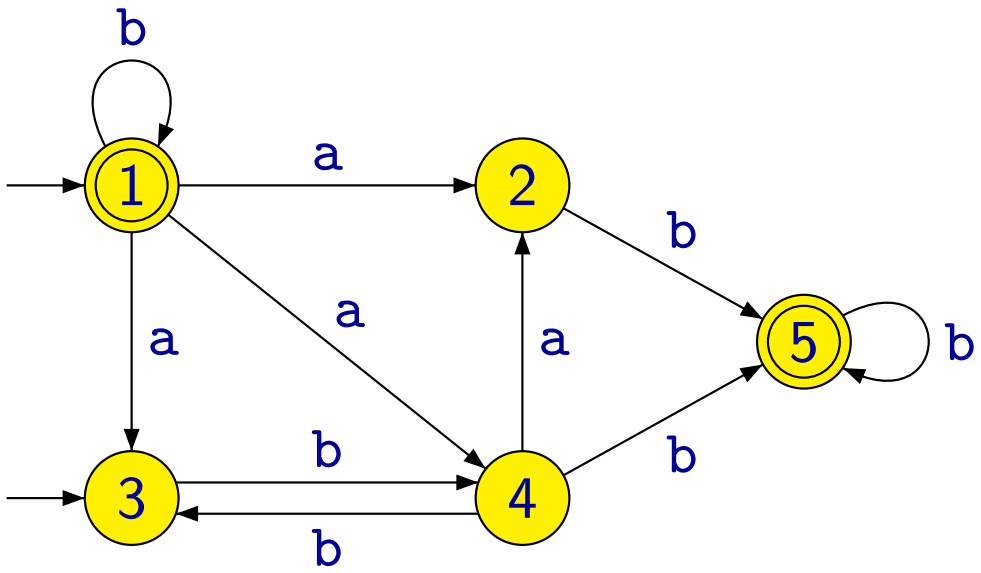
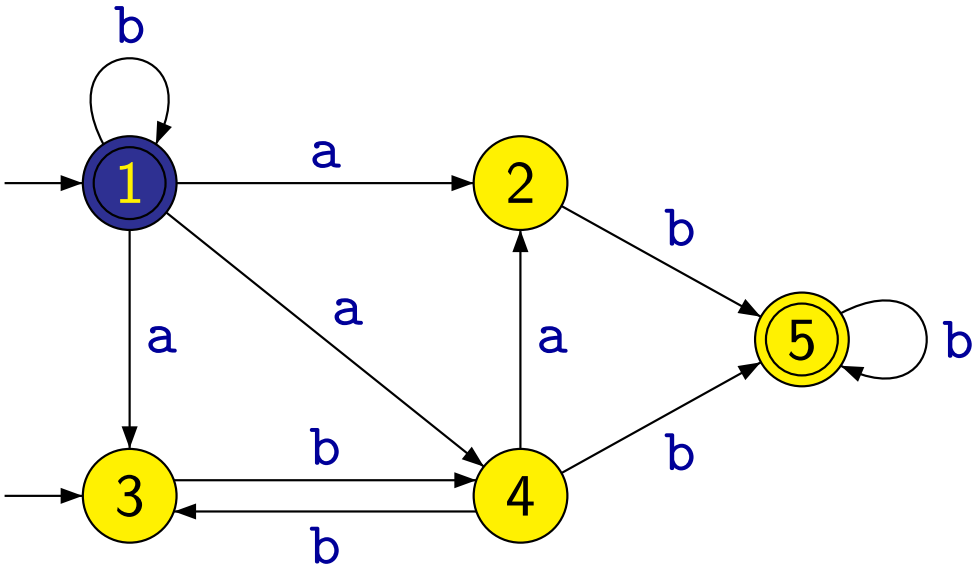
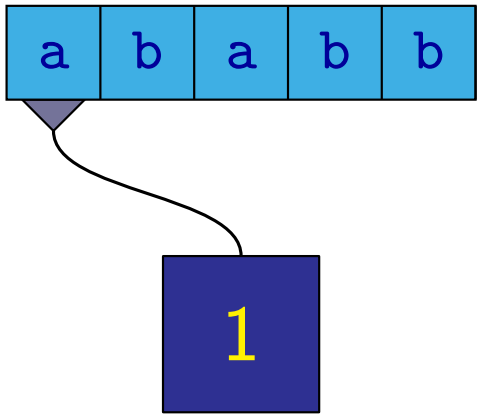
Nedeterministický konečný automat



- Z jednoho stavu může vézt libovolný (i nulový) počet přechodů označených stejným symbolem.
- V automatu může být víc než jeden počáteční stav.

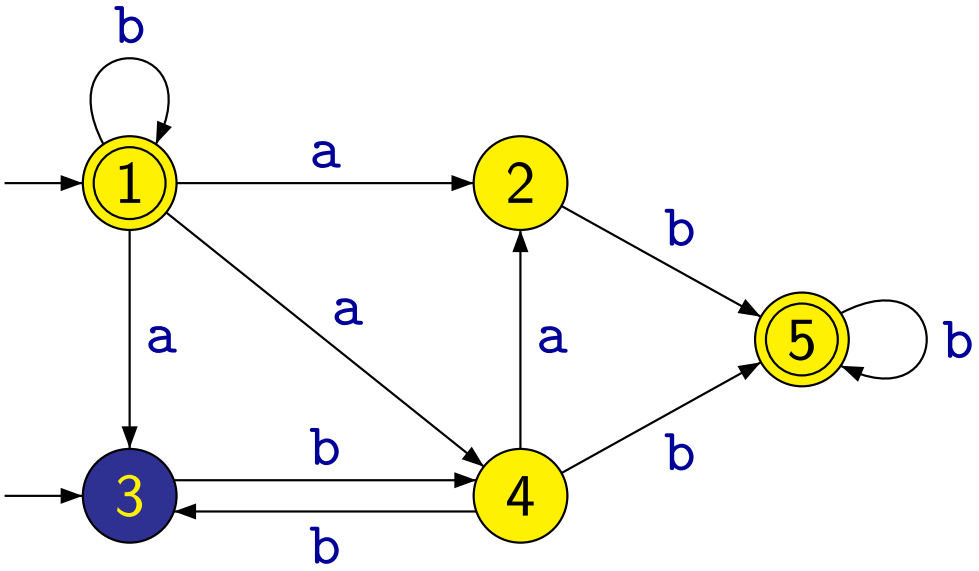
Nedeterministický konečný automat

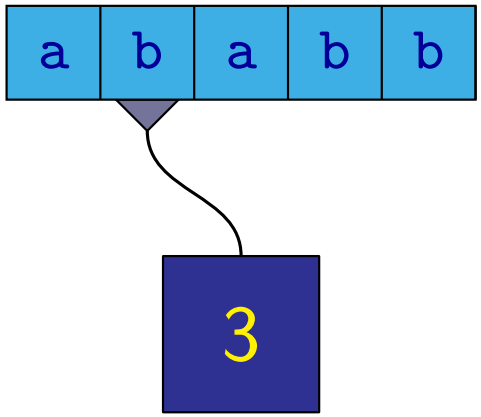




1

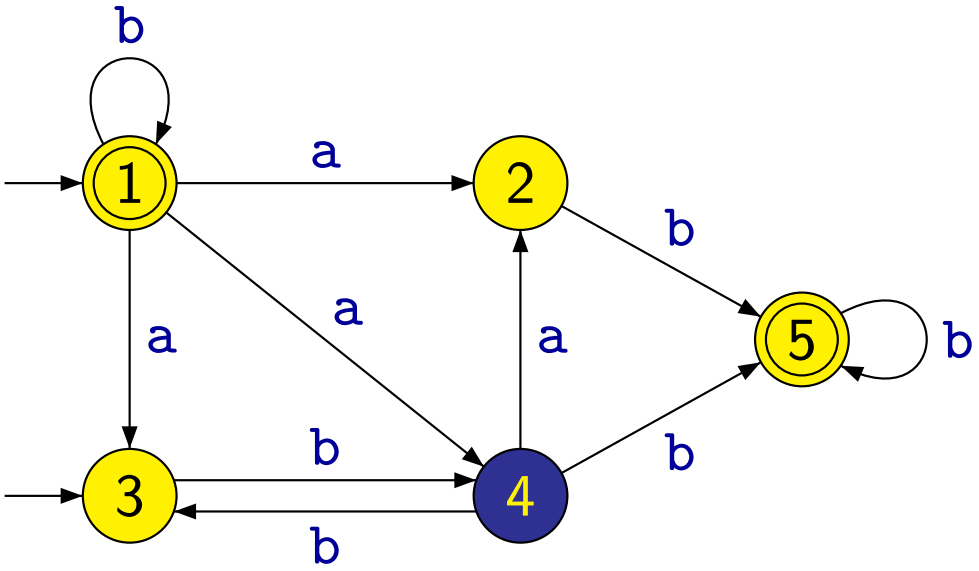
Nedeterministický konečný automat

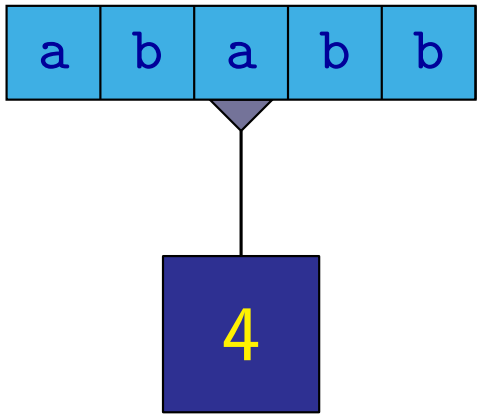




$1 \xrightarrow{a} 3$

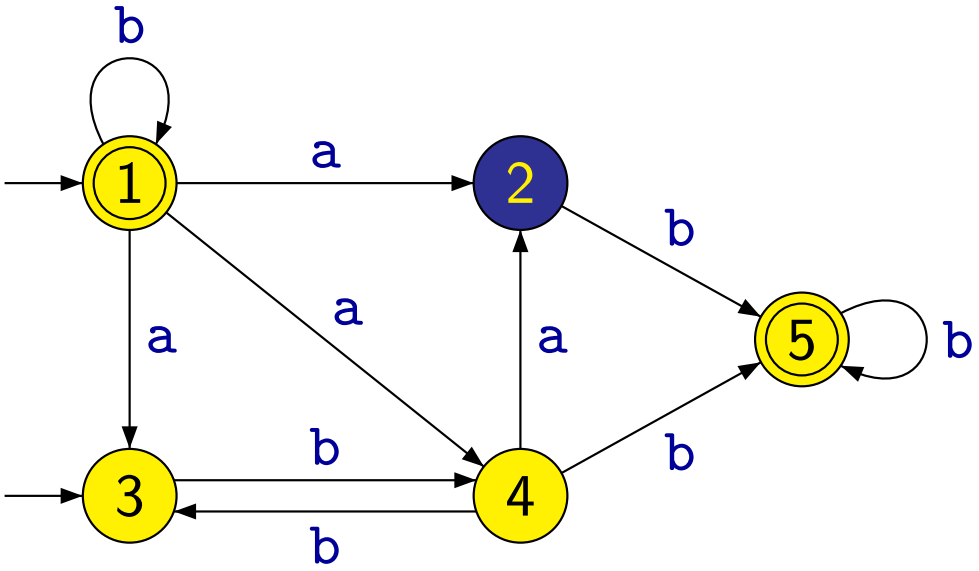
Nedeterministický konečný automat

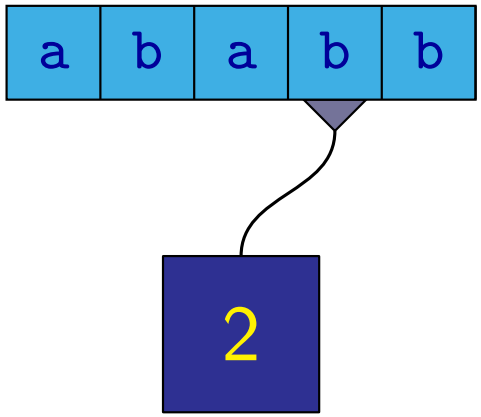




$$1 \xrightarrow{a} 3 \xrightarrow{b} 4$$

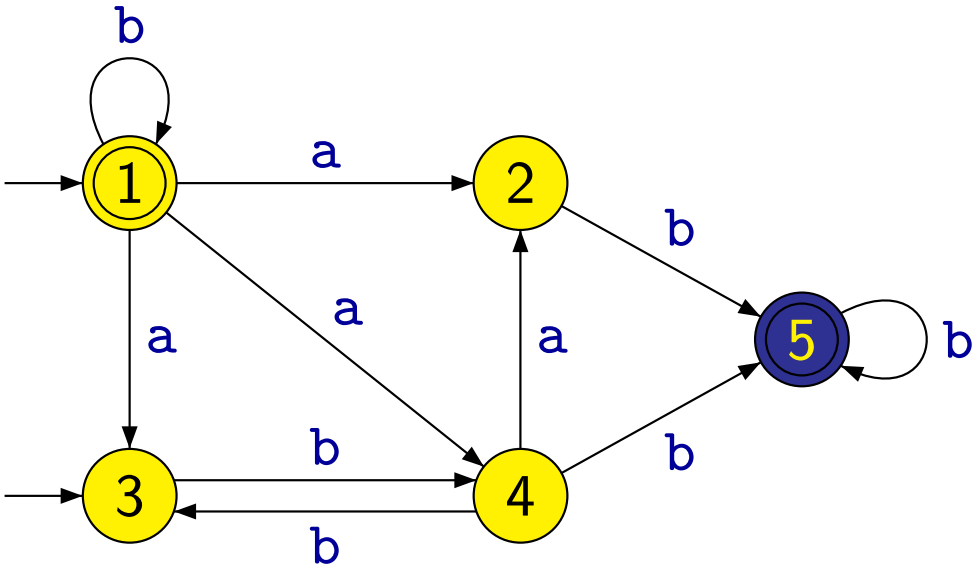
Nedeterministický konečný automat

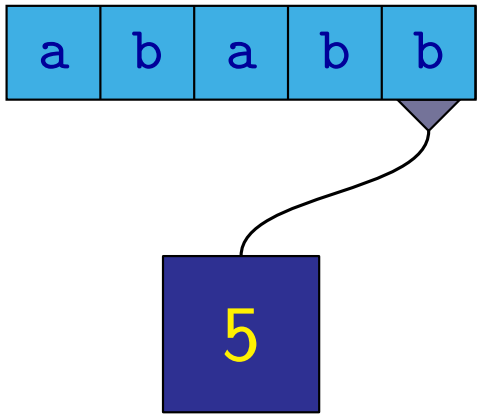




$$1 \xrightarrow{a} 3 \xrightarrow{b} 4 \xrightarrow{a} 2$$

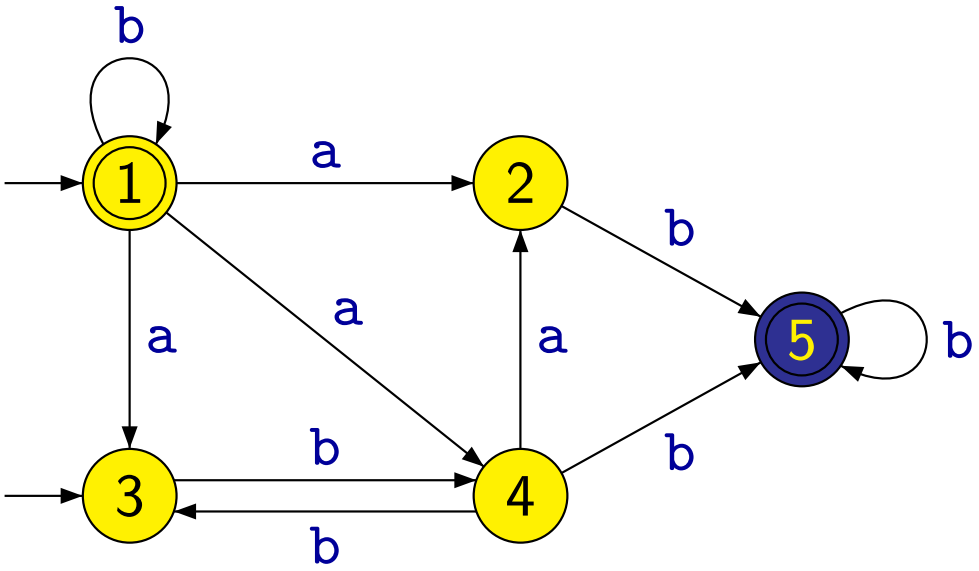
Nedeterministický konečný automat

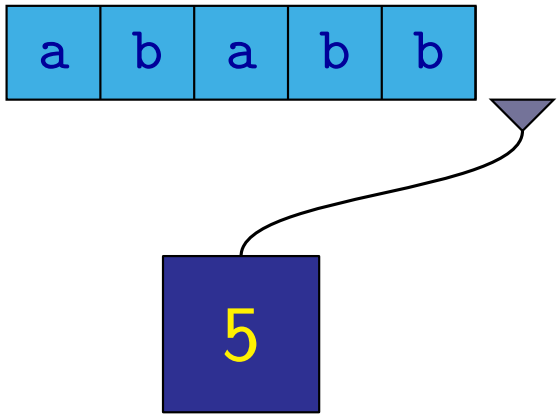




$$1 \xrightarrow{a} 3 \xrightarrow{b} 4 \xrightarrow{a} 2 \xrightarrow{b} 5$$

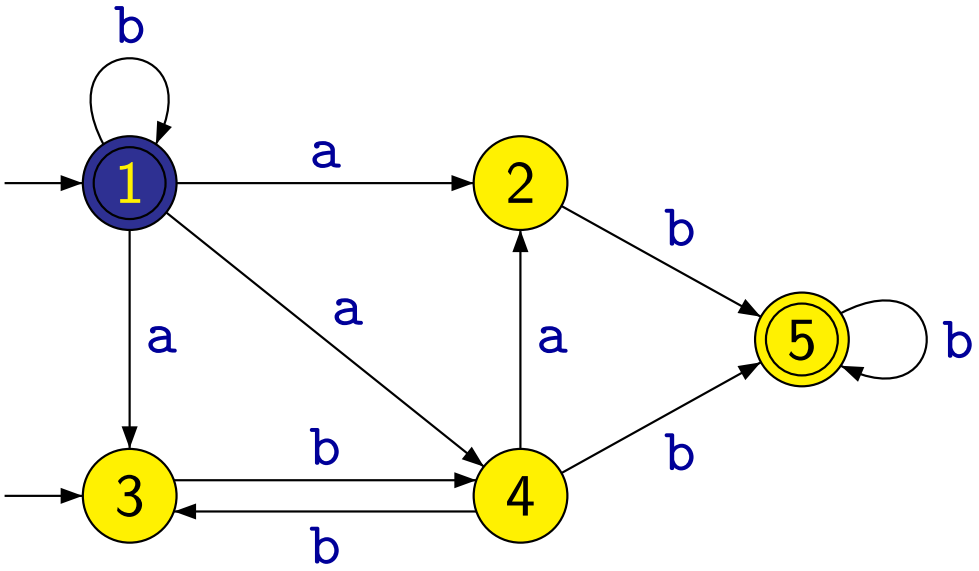
Nedeterministický konečný automat

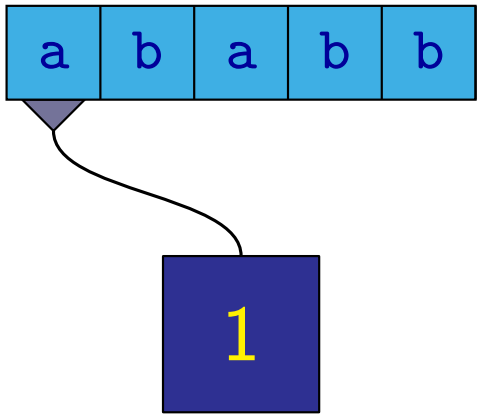




$$1 \xrightarrow{a} 3 \xrightarrow{b} 4 \xrightarrow{a} 2 \xrightarrow{b} 5 \xrightarrow{b} 5$$

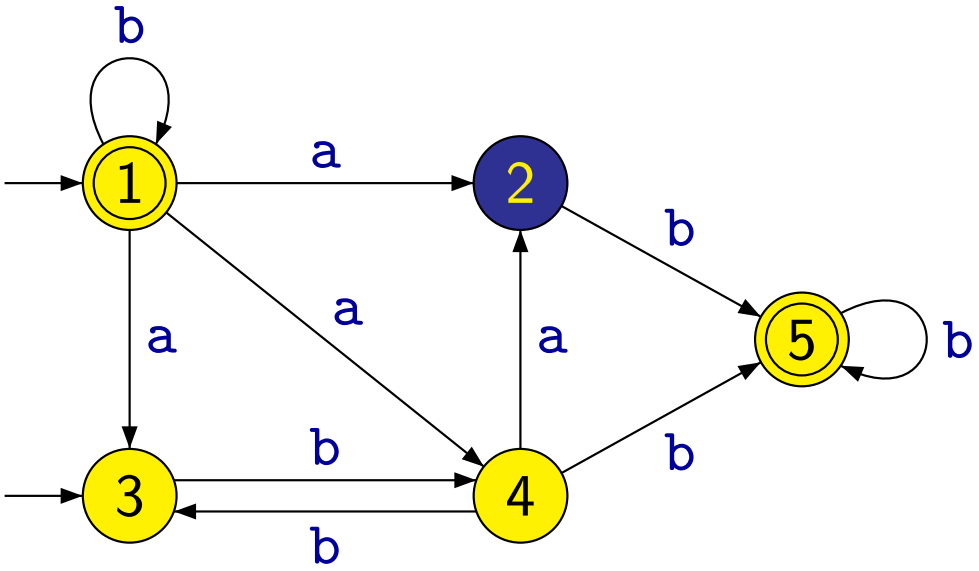
Nedeterministický konečný automat

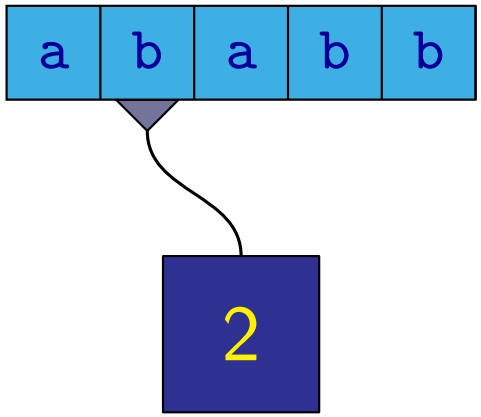




1

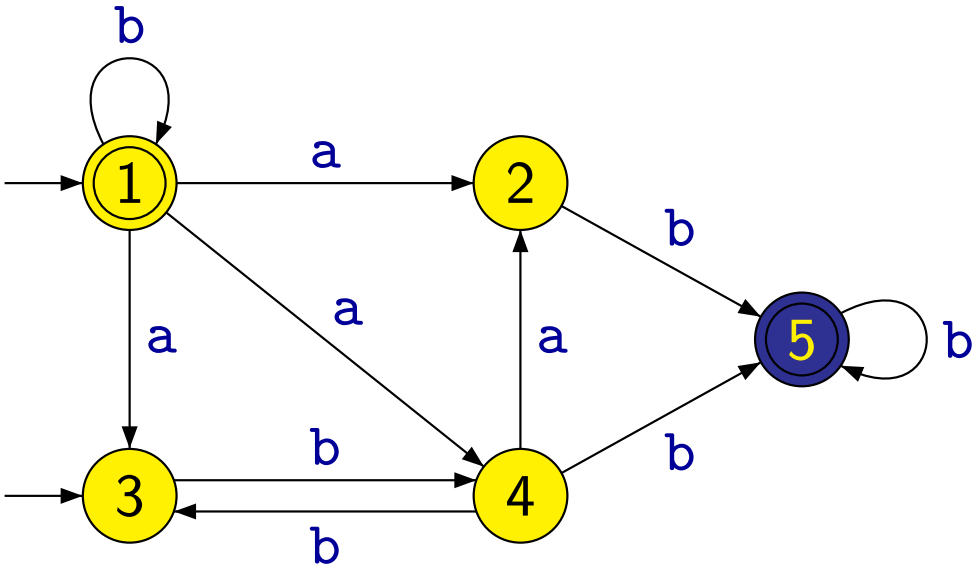
Nedeterministický konečný automat

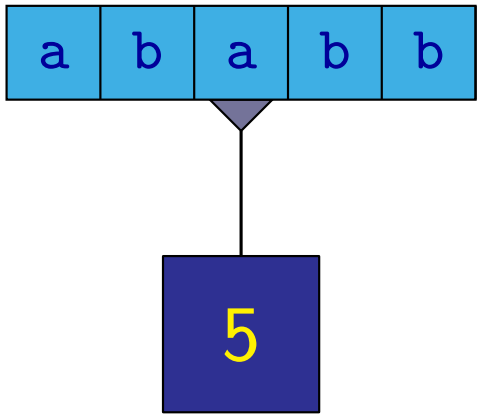




$1 \xrightarrow{a} 2$

Nedeterministický konečný automat

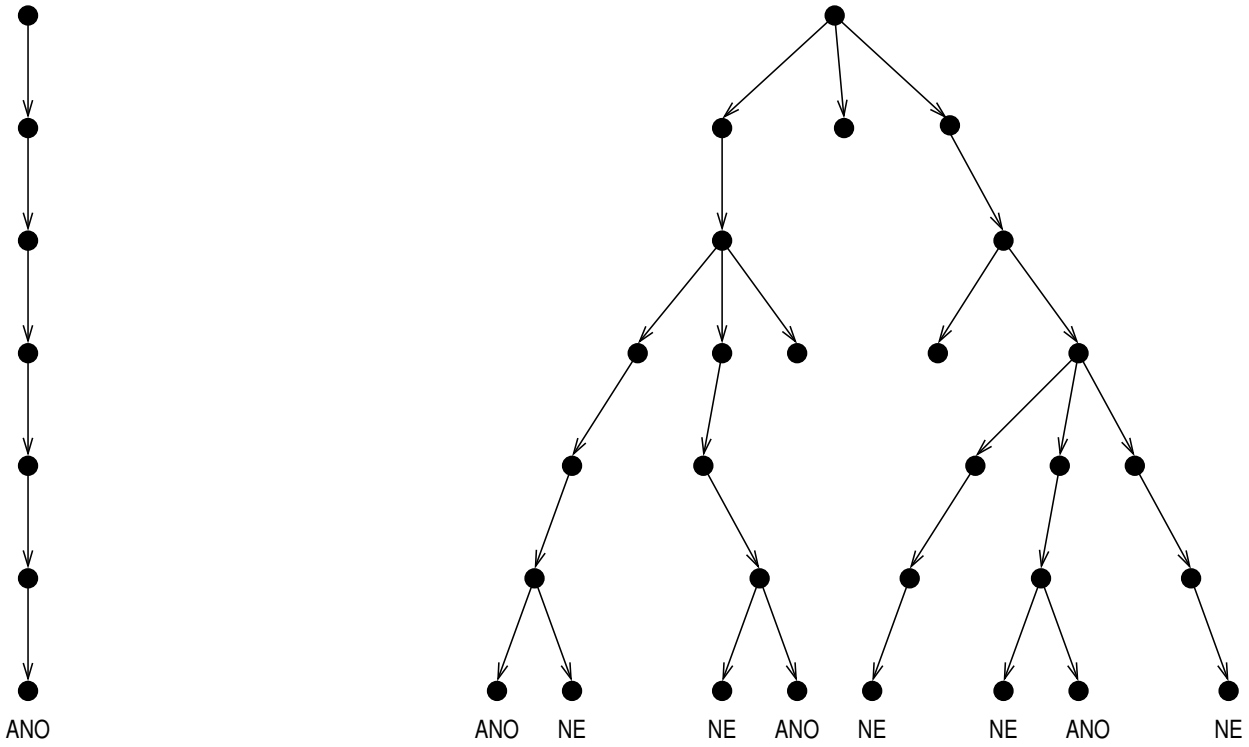




$$1 \xrightarrow{a} 2 \xrightarrow{b} 5$$

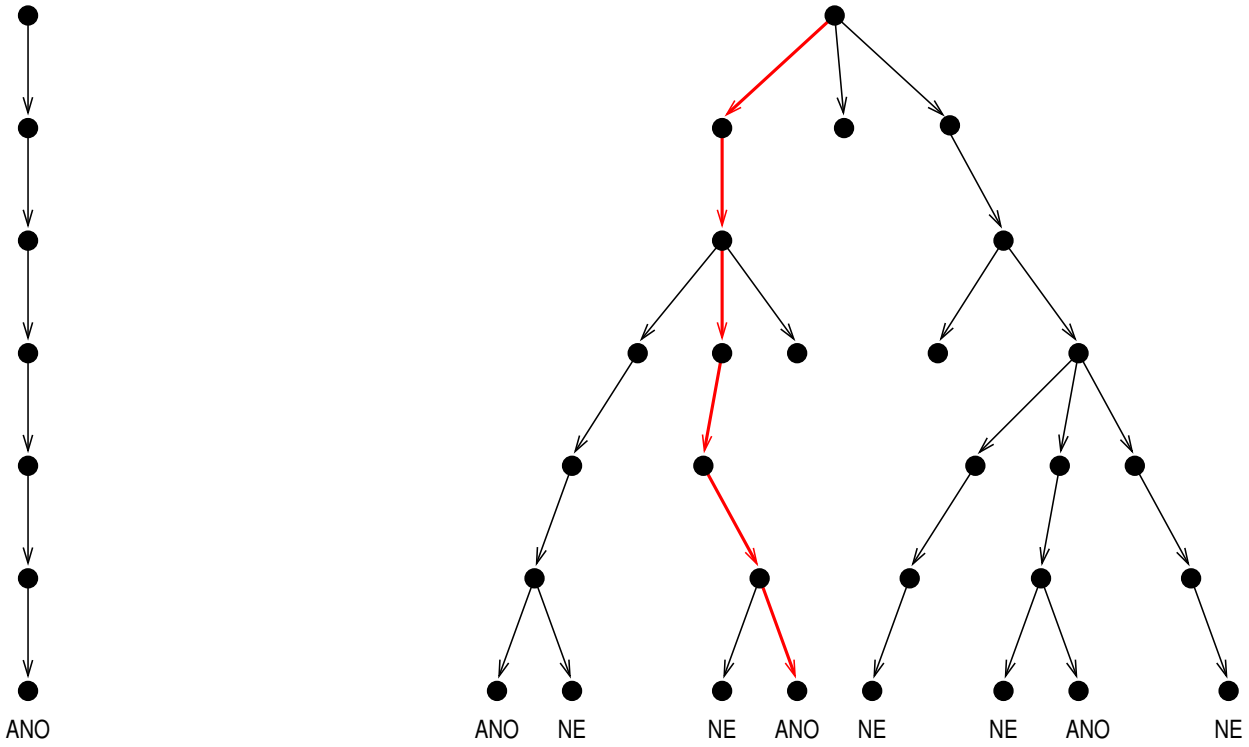
Nedeterministický konečný automat

Nedeterministický konečný automat přijímá dané slovo, jestliže **existuje** alespoň jeden jeho výpočet, který vede k přijetí tohoto slova.



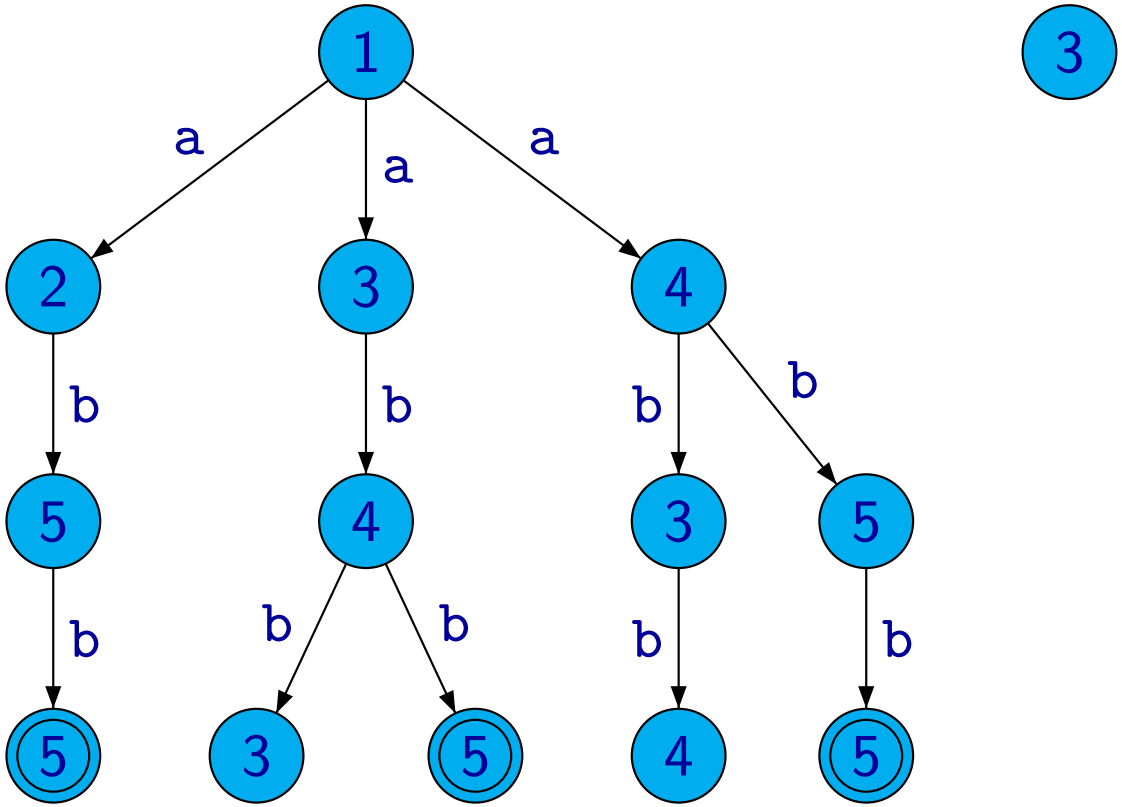
Nedeterministický konečný automat

Nedeterministický konečný automat přijímá dané slovo, jestliže **existuje** alespoň jeden jeho výpočet, který vede k přijetí tohoto slova.



Nedeterministický konečný automat

	a	b
$\leftrightarrow 1$	$2, 3, 4$	1
2	$-$	5
$\rightarrow 3$	$-$	4
4	2	$3, 5$
$\leftarrow 5$	$-$	5



Příklad: Les reprezentující všechny možné výpočty nad slovem abb.

Nedeterministický konečný automat

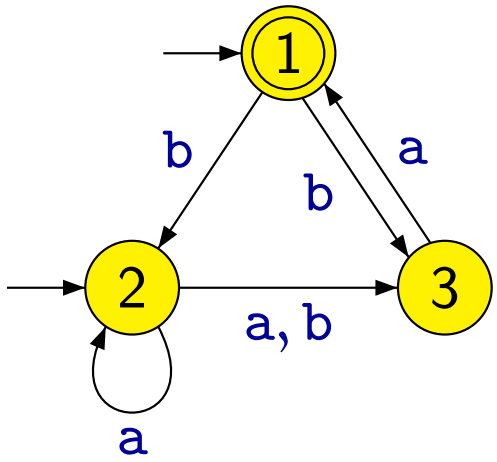
Formálně je **nedeterministický konečný automat** (**NKA**) definován jako pětice

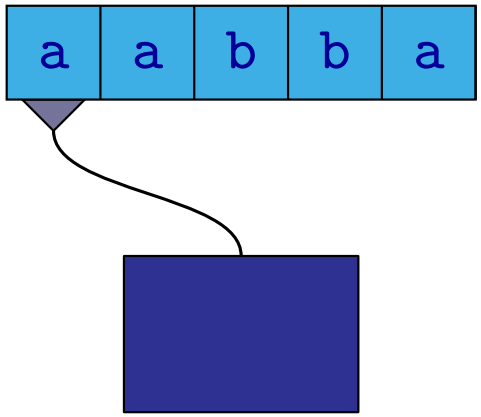
$$(Q, \Sigma, \delta, I, F)$$

kde:

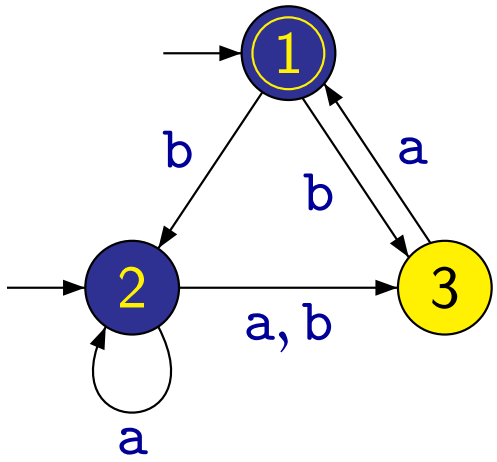
- Q je konečná množina **stavů**
- Σ je konečná **abeceda**
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ je **přechodová funkce**
- $I \subseteq Q$ je množina **počátečních stavů**
- $F \subseteq Q$ je množina **přijímajících stavů**

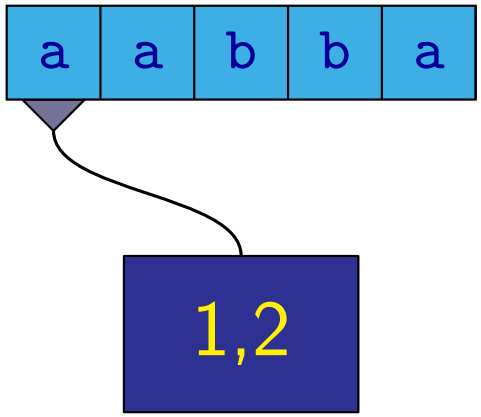
Převod NKA na DKA



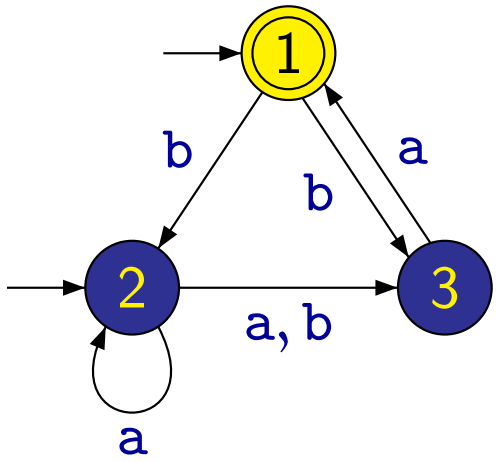


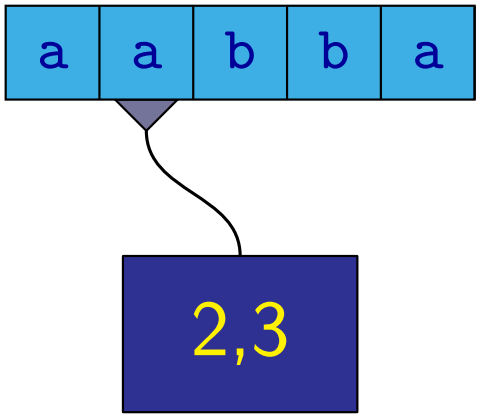
Převod NKA na DKA



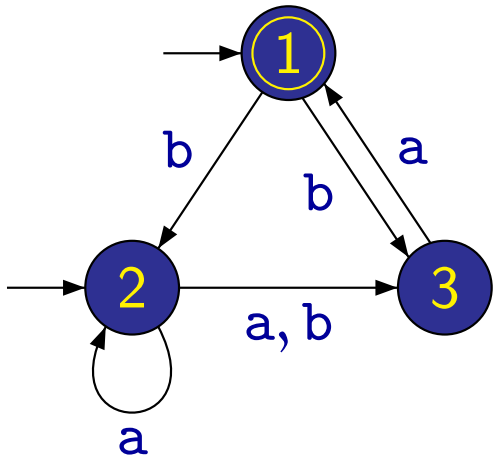


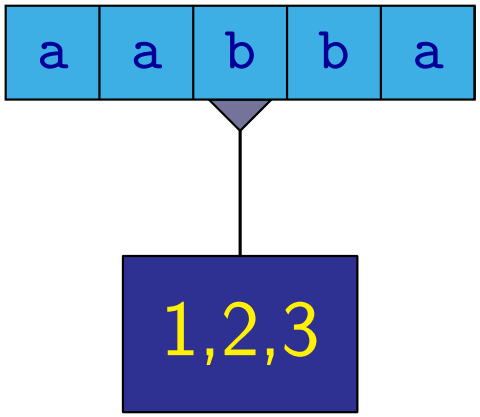
Převod NKA na DKA



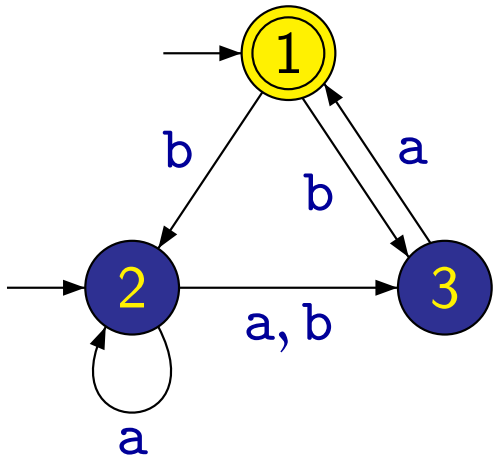


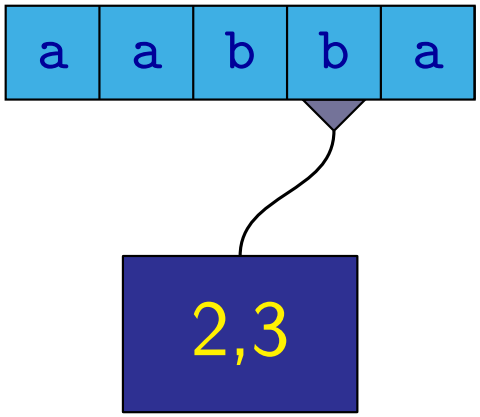
Převod NKA na DKA



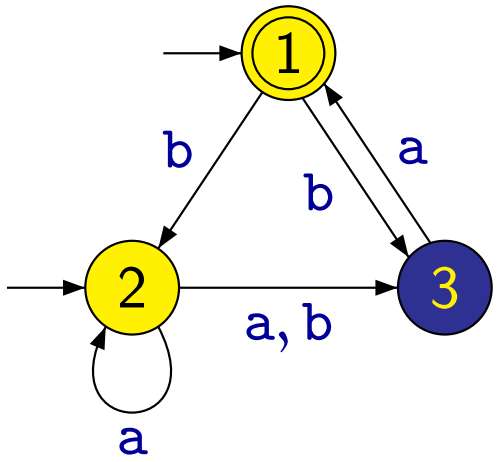


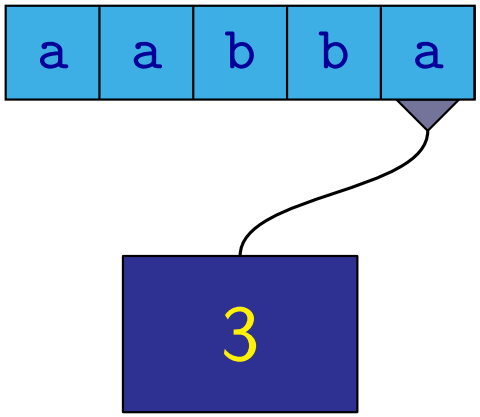
Převod NKA na DKA



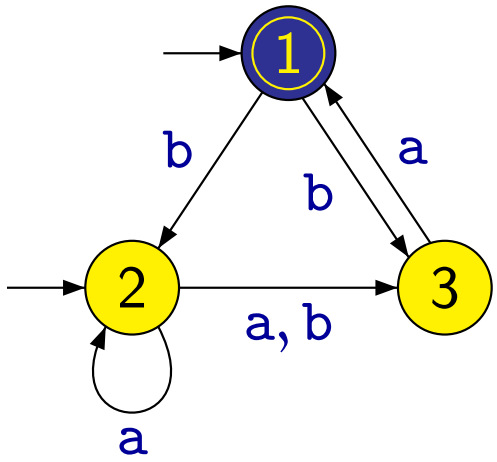


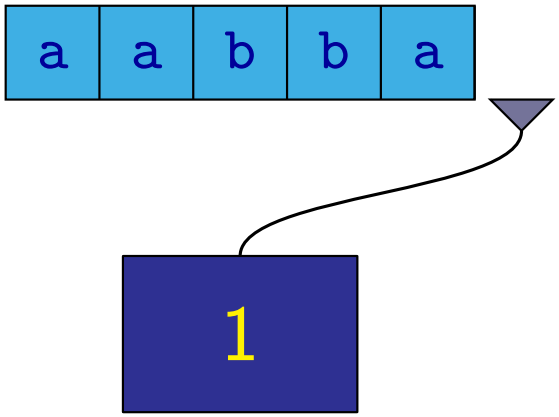
Převod NKA na DKA



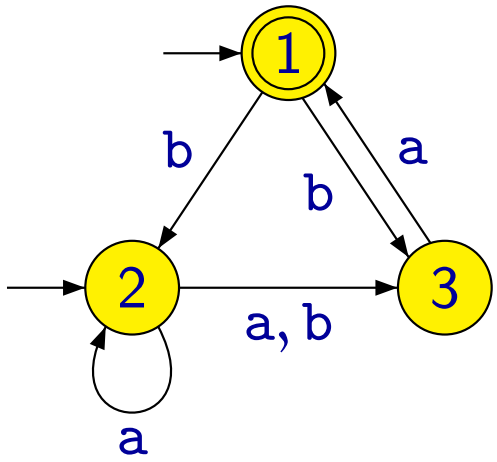


Převod NKA na DKA

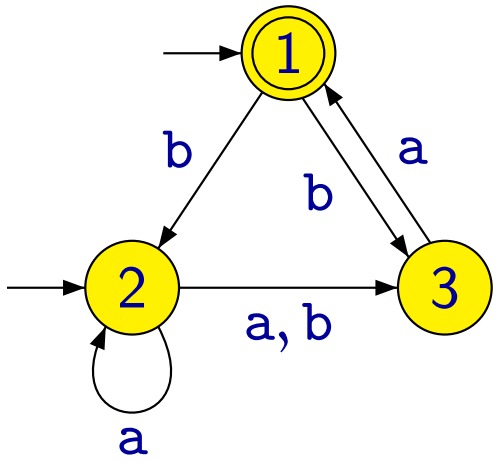


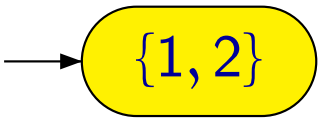


Převod NKA na DKA

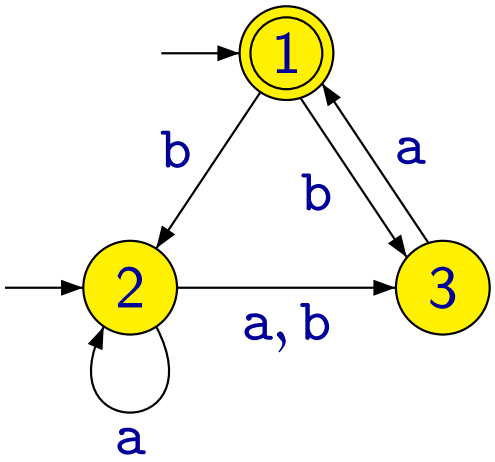


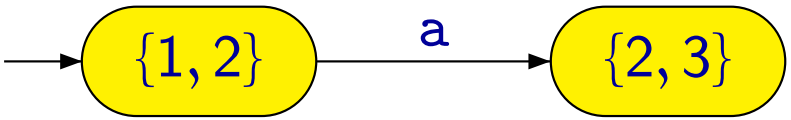
Převod NKA na DKA



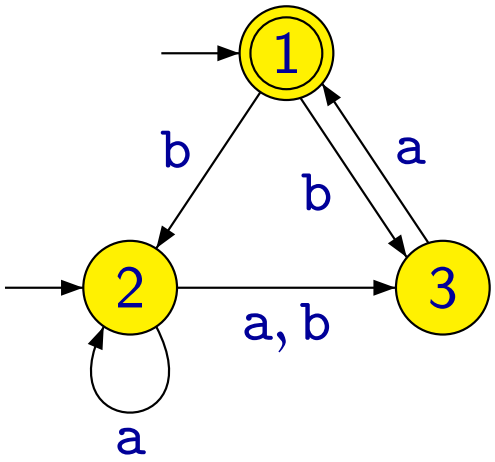


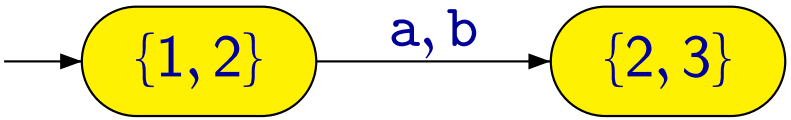
Převod NKA na DKA



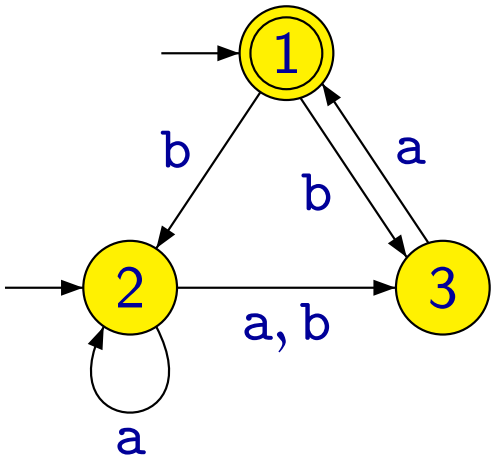


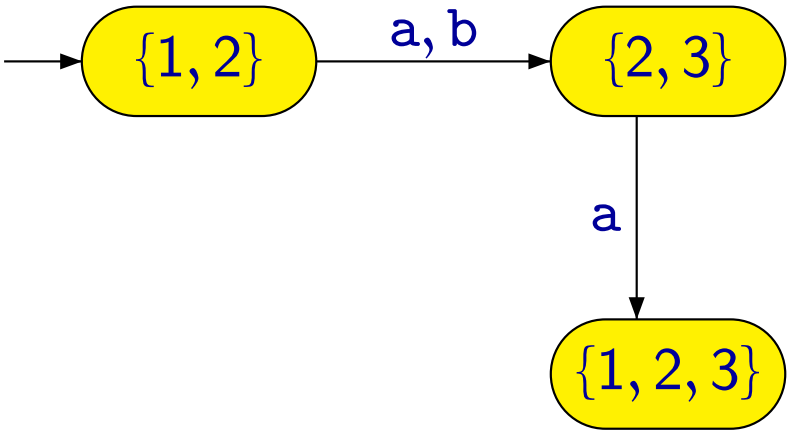
Převod NKA na DKA



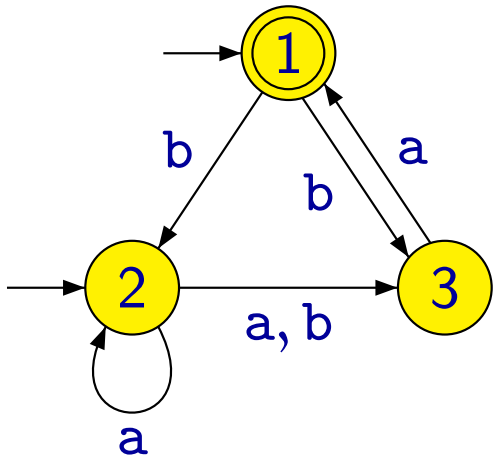


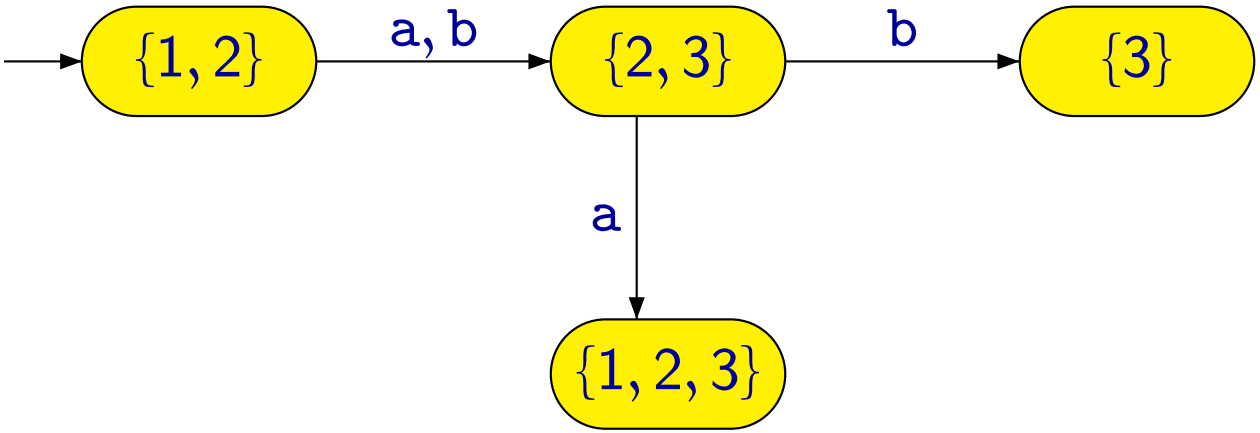
Převod NKA na DKA



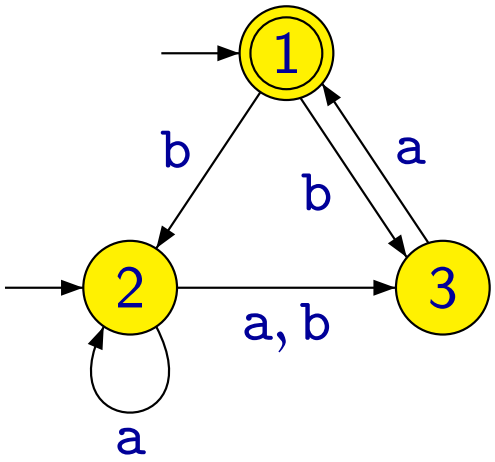


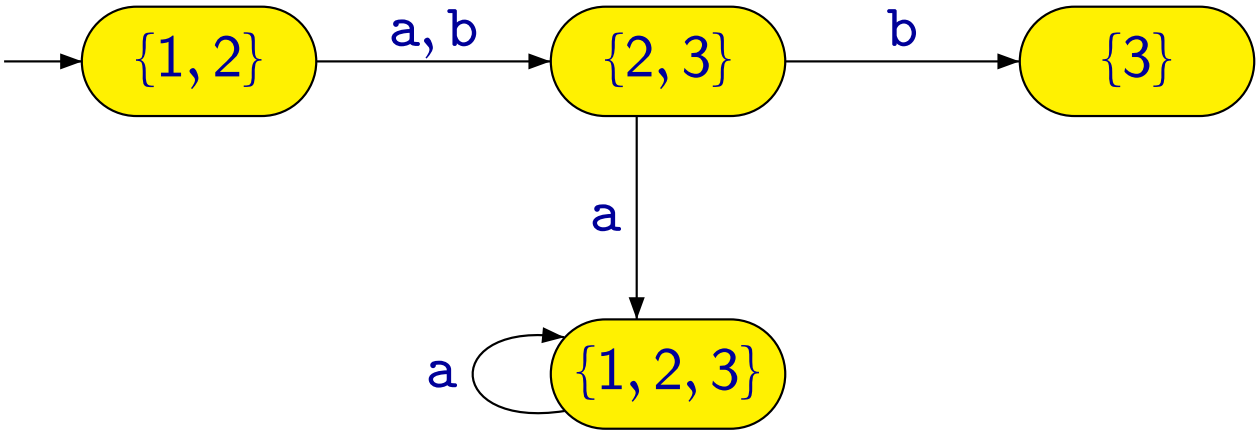
Převod NKA na DKA



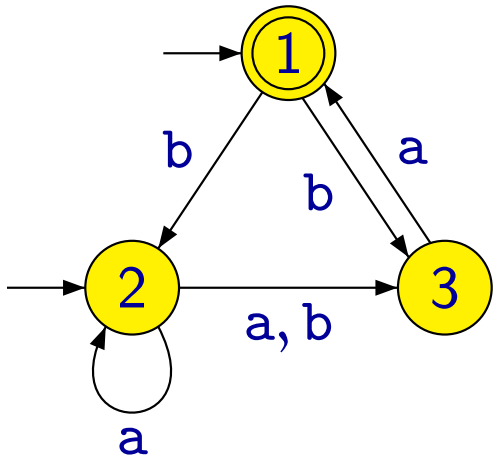


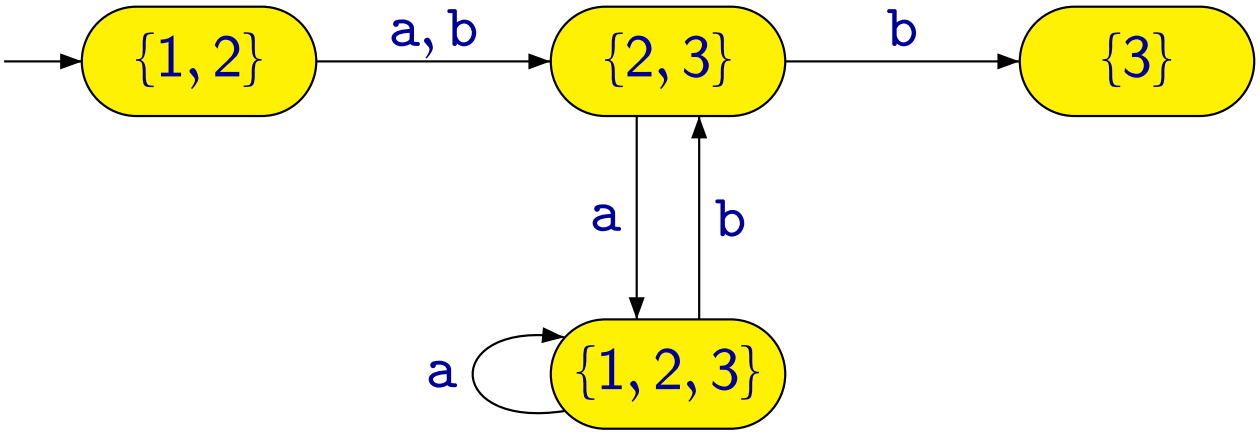
Převod NKA na DKA



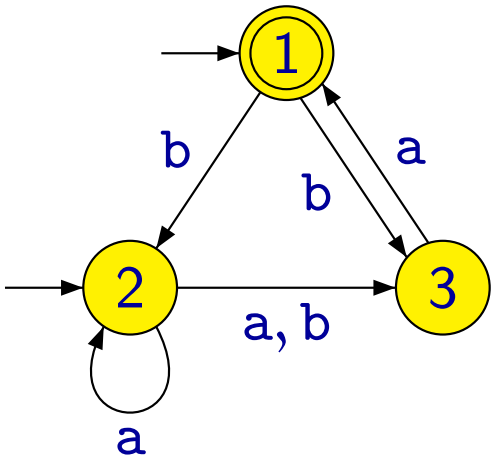


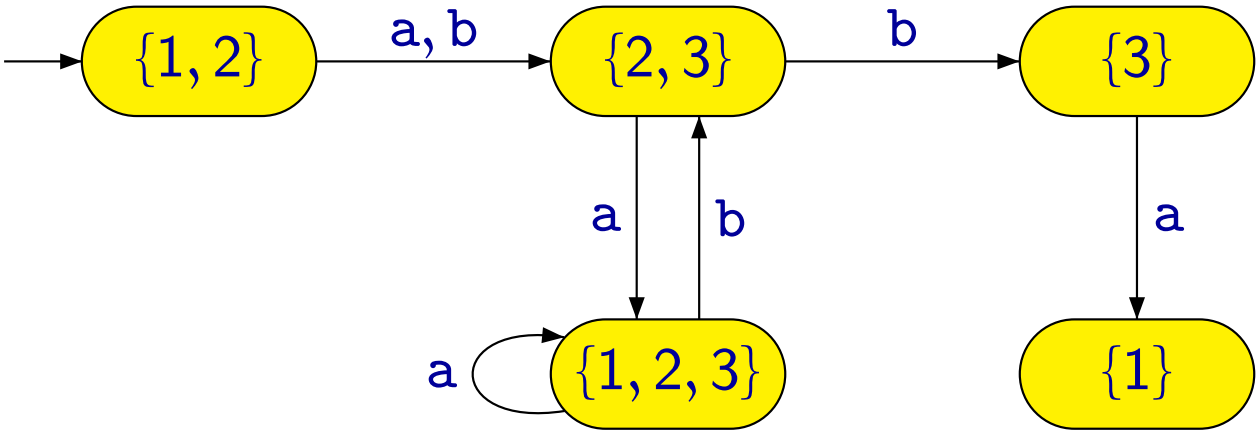
Převod NKA na DKA



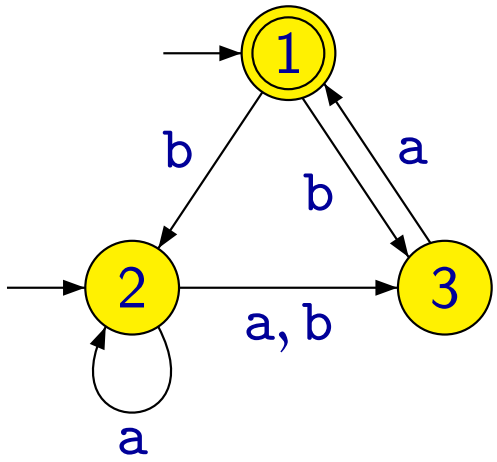


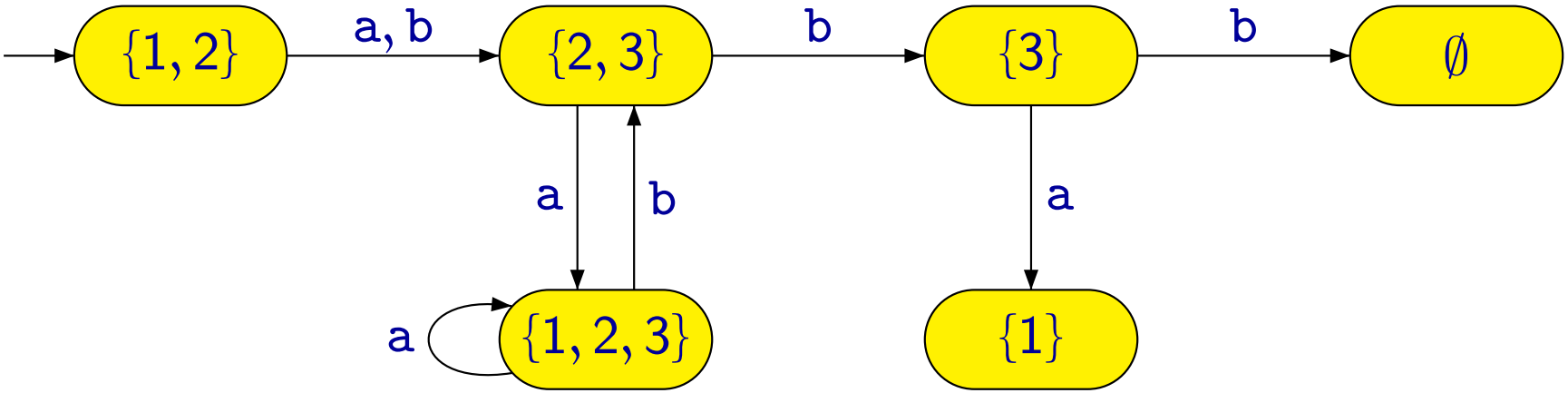
Převod NKA na DKA



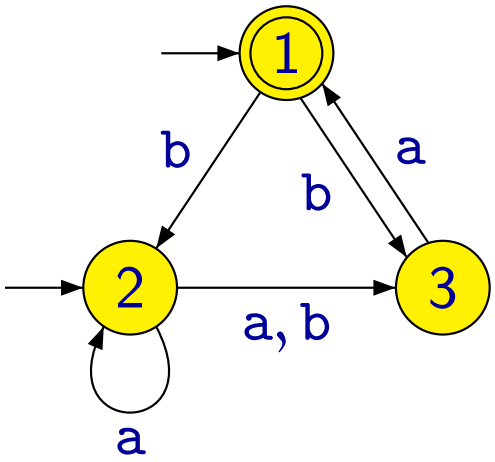


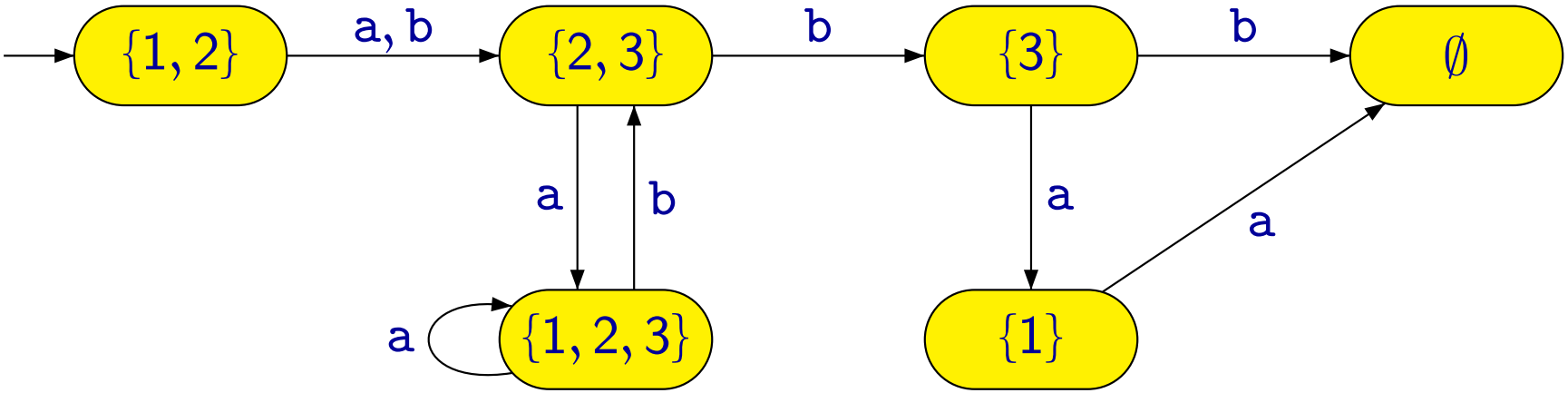
Převod NKA na DKA



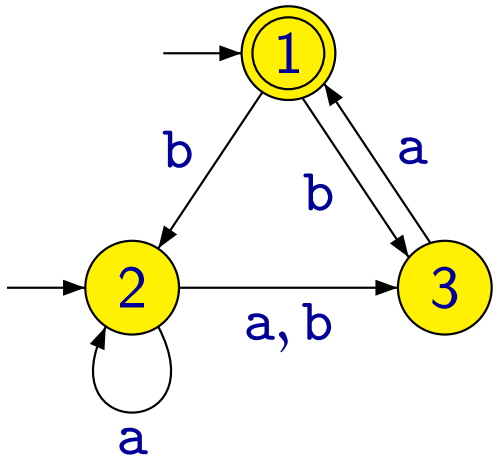


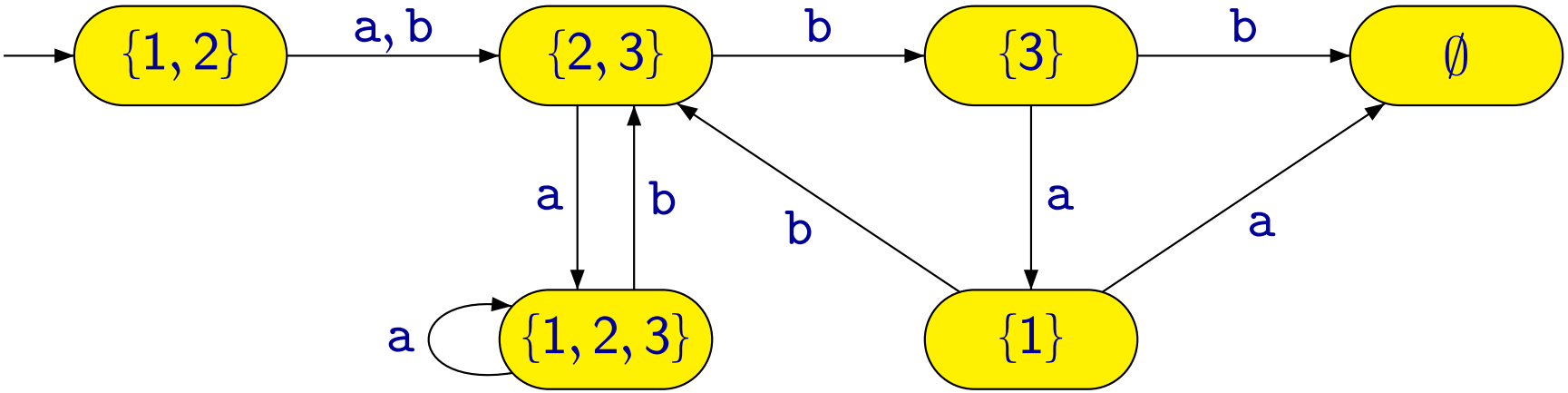
Převod NKA na DKA



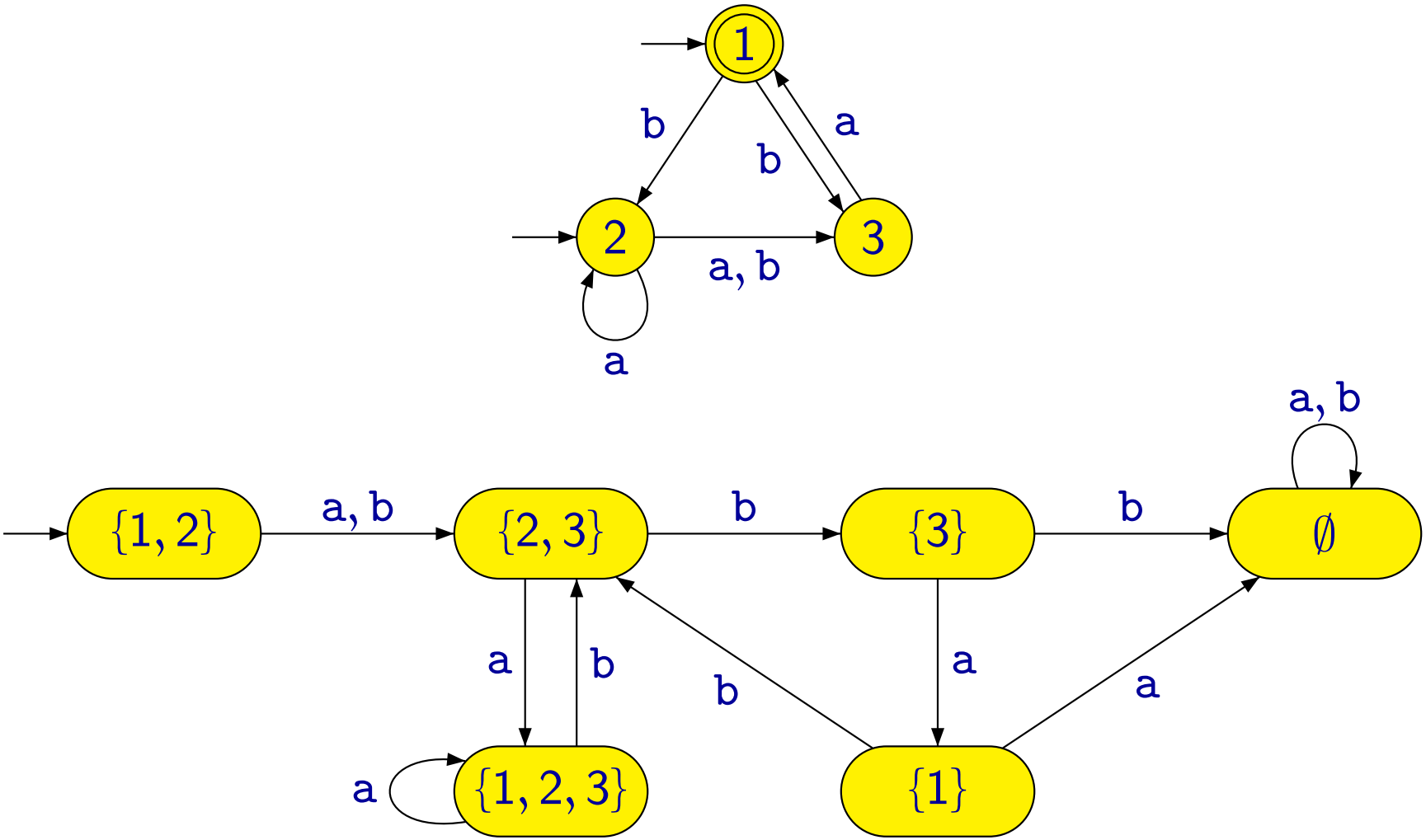


Převod NKA na DKA

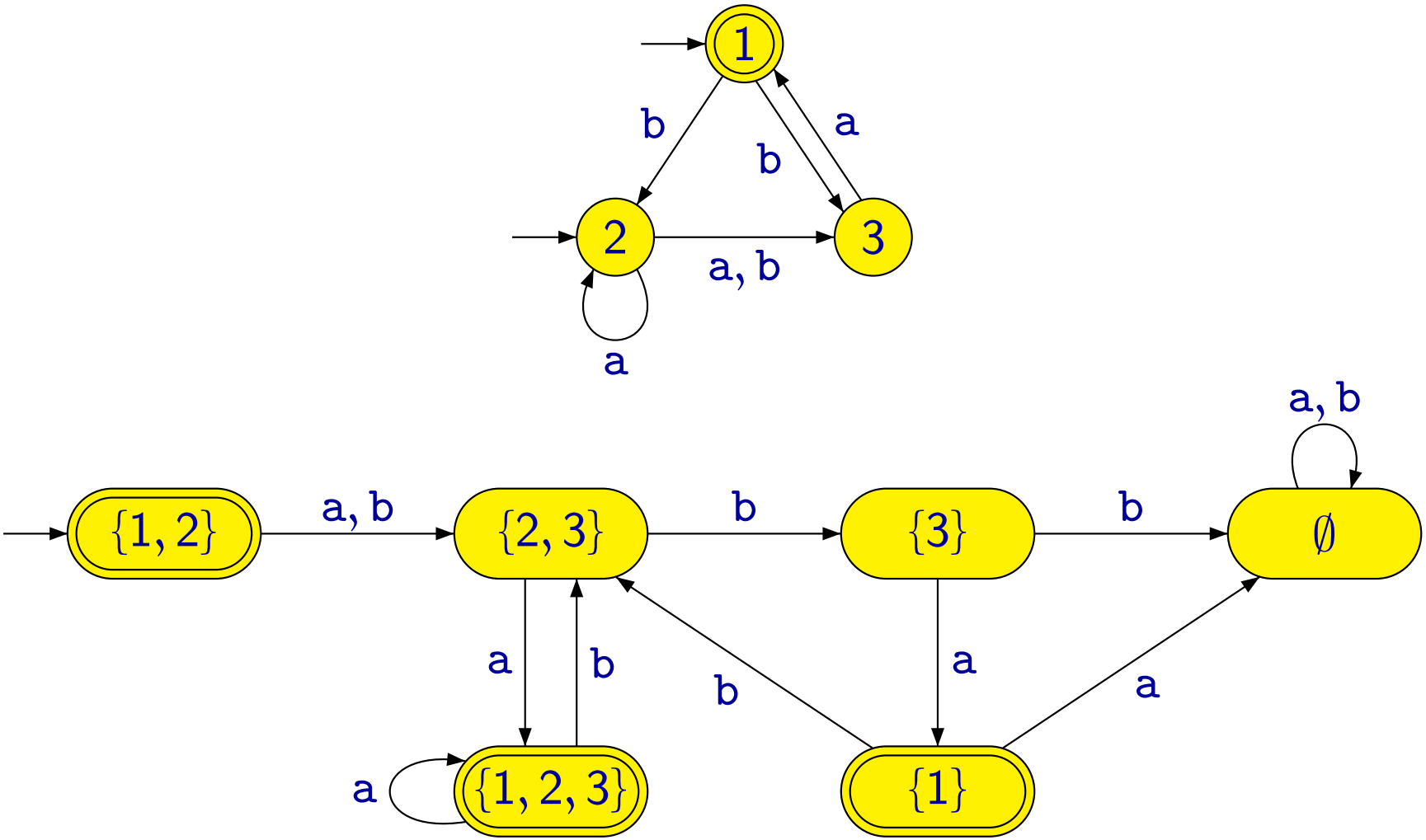




Převod NKA na DKA



Převod NKA na DKA



Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2, 3$
$\rightarrow 2$	$2, 3$	3
3	1	—

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	
$\{2,3\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	
$\leftarrow \{1,2,3\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$		
$\{3\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	
$\{3\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$	$\{1\}$	

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$	$\{1\}$	
$\leftarrow \{1\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$	$\{1\}$	$\emptyset$
$\leftarrow \{1\}$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$	$\{1\}$	$\emptyset$
$\leftarrow \{1\}$		
$\emptyset$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$	$\{1\}$	$\emptyset$
$\leftarrow \{1\}$	$\emptyset$	
$\emptyset$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$	$\{1\}$	$\emptyset$
$\leftarrow \{1\}$	$\emptyset$	$\{2,3\}$
$\emptyset$		

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$	$\{1\}$	$\emptyset$
$\leftarrow \{1\}$	$\emptyset$	$\{2,3\}$
$\emptyset$	$\emptyset$	$\emptyset$

Převod NKA na DKA

	a	b
$\leftrightarrow 1$	—	$2,3$
$\rightarrow 2$	$2,3$	3
3	1	—

	a	b
$\leftrightarrow \{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\{3\}$	$\{1\}$	$\emptyset$
$\leftarrow \{1\}$	$\emptyset$	$\{2,3\}$
$\emptyset$	$\emptyset$	$\emptyset$

	a	b
$\leftrightarrow 1$	2	2
2	3	4
$\leftarrow 3$	3	2
4	5	6
$\leftarrow 5$	6	2
6	6	6

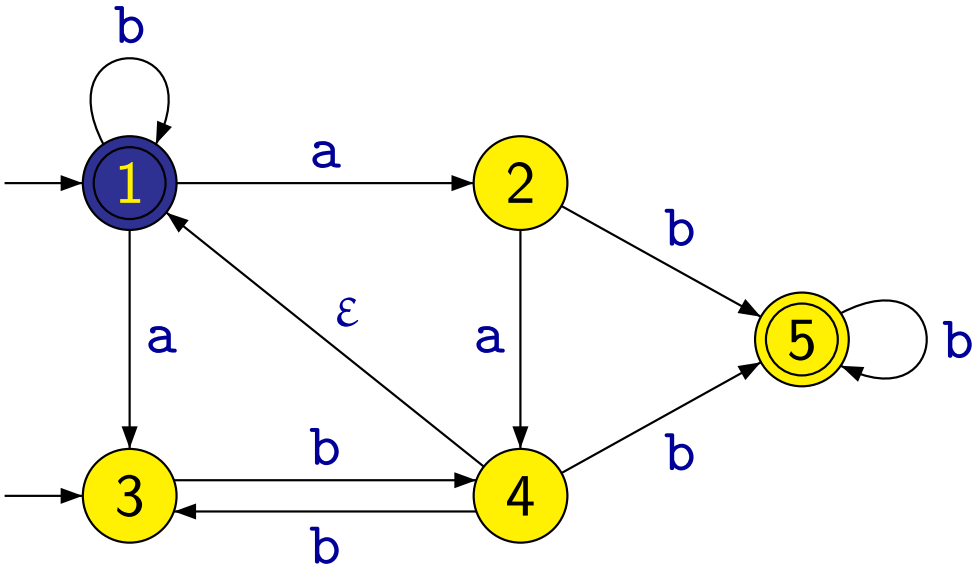
Převod NKA na DKA

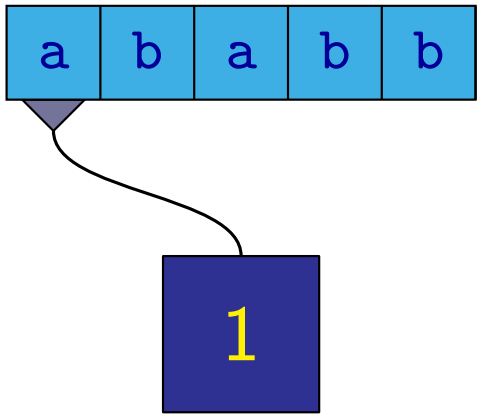
Poznámka: Při převodu nedeterministického automatu, který má n stavů, může mít výsledný deterministický automat až 2^n stavů.

Například při převodu automatu, který má 20 stavů, může vzniknout automat, který má $2^{20} = 1048576$ stavů.

Často má sice výsledný automat podstatně méně než 2^n stavů, nicméně tyto nejhorší případy občas nastávají.

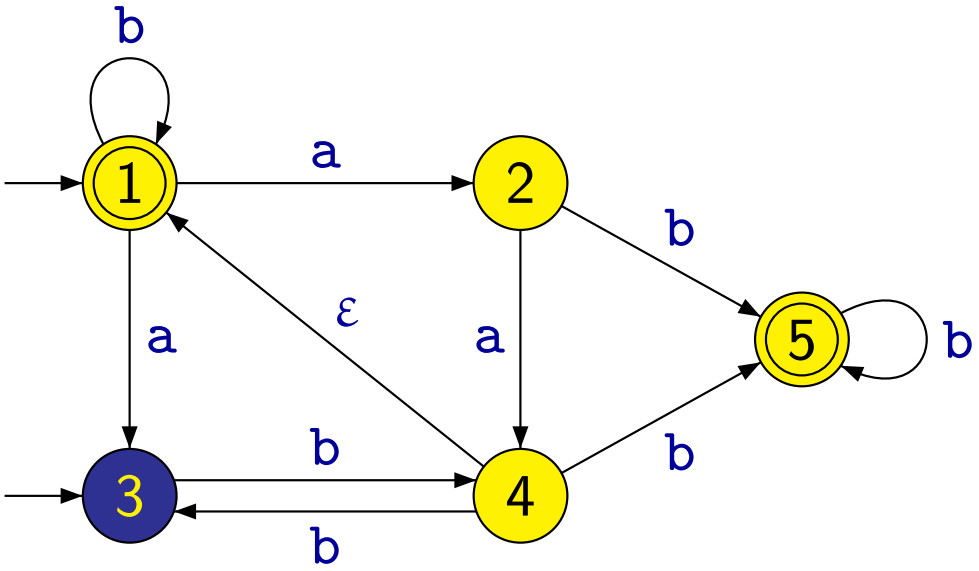
Zobecněný nedeterministický konečný automat

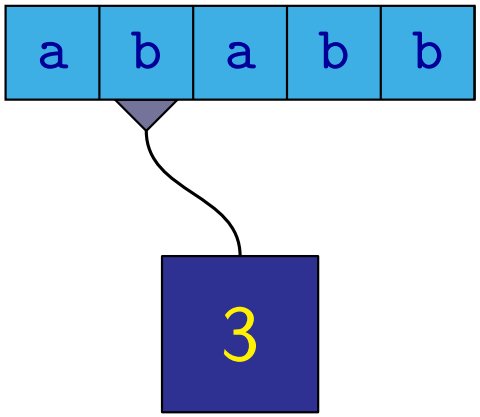




1

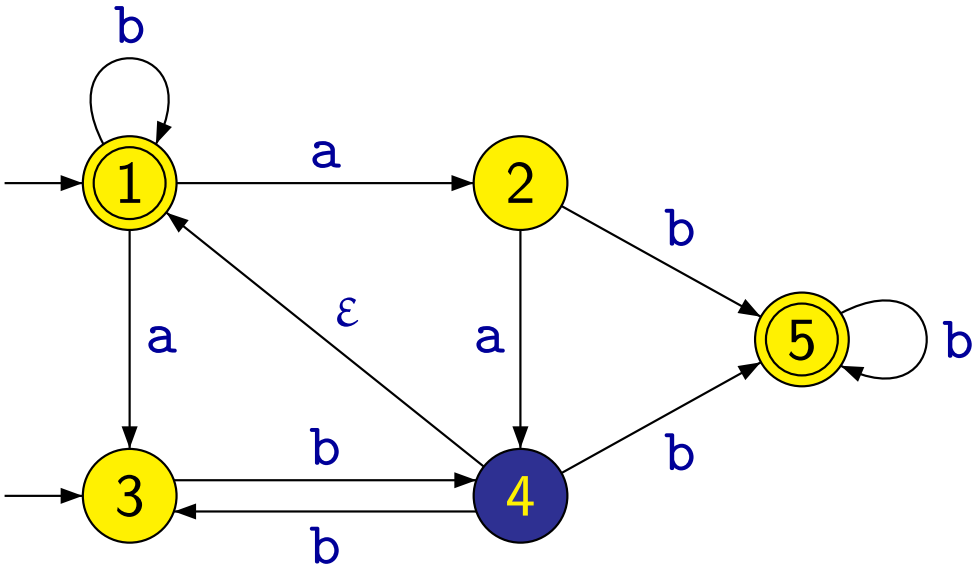
Zobecněný nedeterministický konečný automat

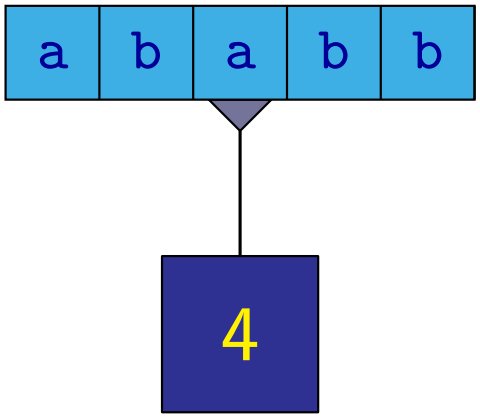




$1 \xrightarrow{a} 3$

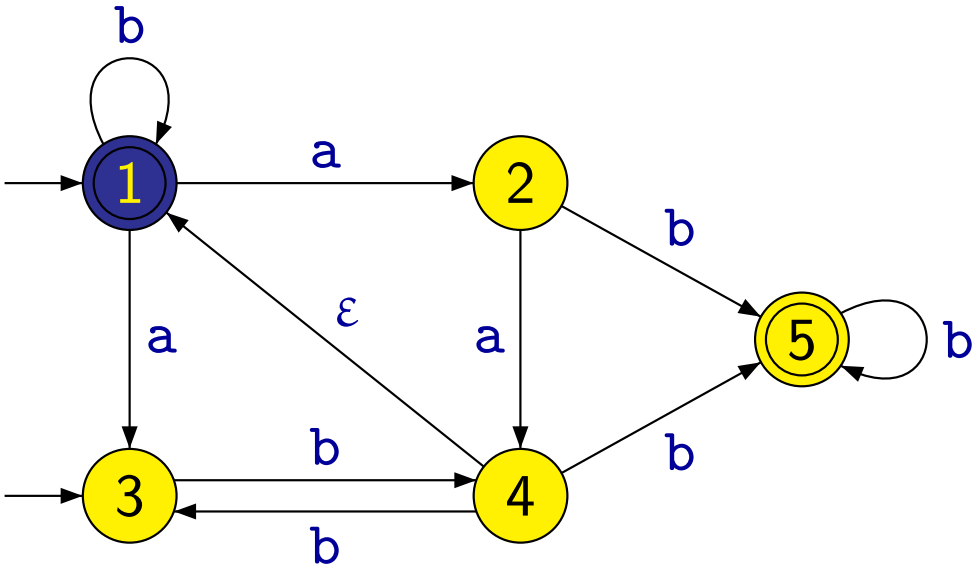
Zobecněný nedeterministický konečný automat

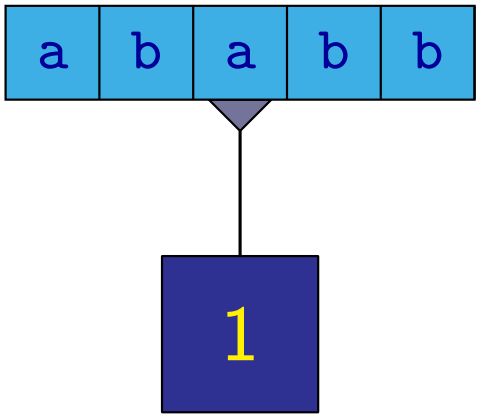




$1 \xrightarrow{a} 3 \xrightarrow{b} 4$

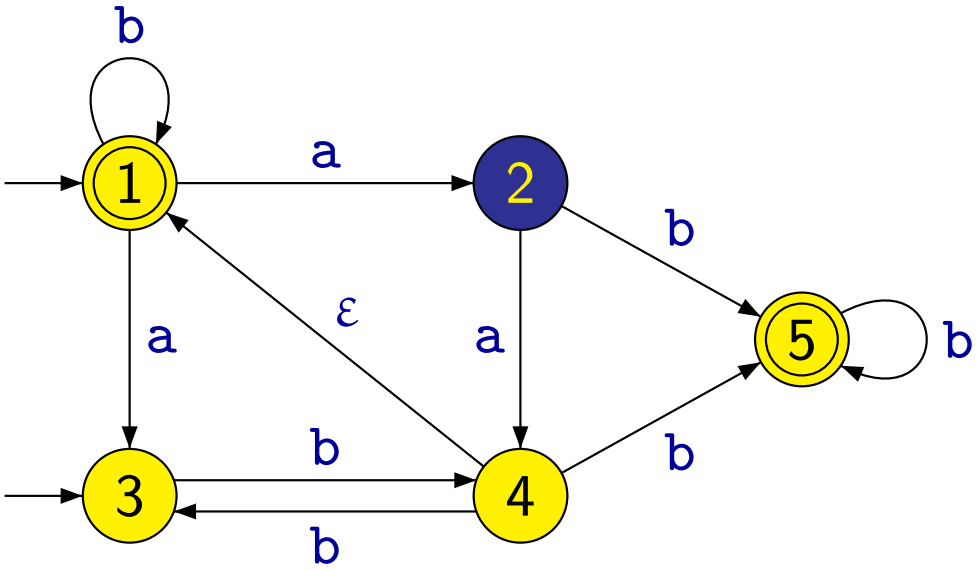
Zobecněný nedeterministický konečný automat

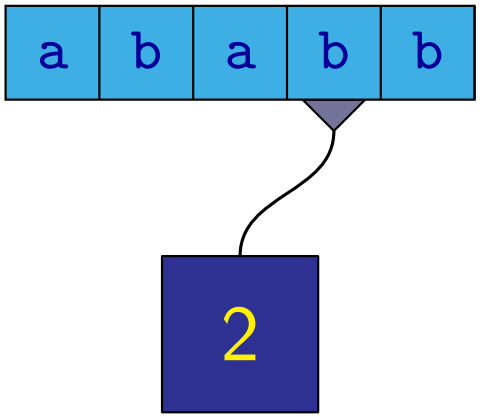




$$1 \xrightarrow{a} 3 \xrightarrow{b} 4 \xrightarrow{\varepsilon} 1$$

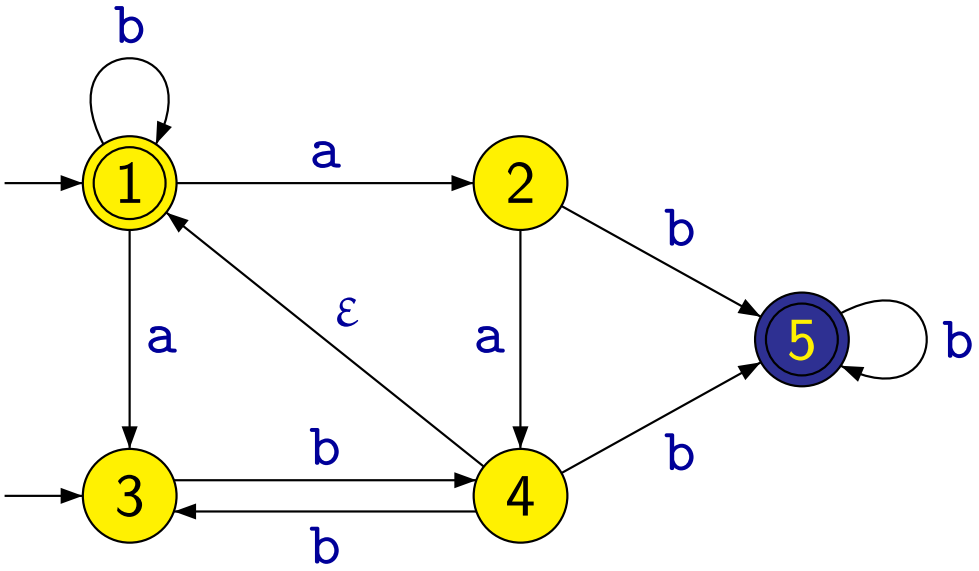
Zobecněný nedeterministický konečný automat

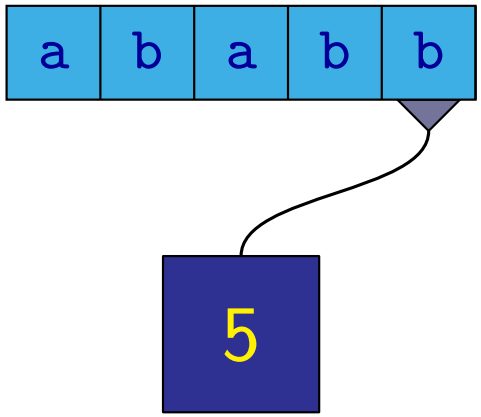




$$1 \xrightarrow{a} 3 \xrightarrow{b} 4 \xrightarrow{\varepsilon} 1 \xrightarrow{a} 2$$

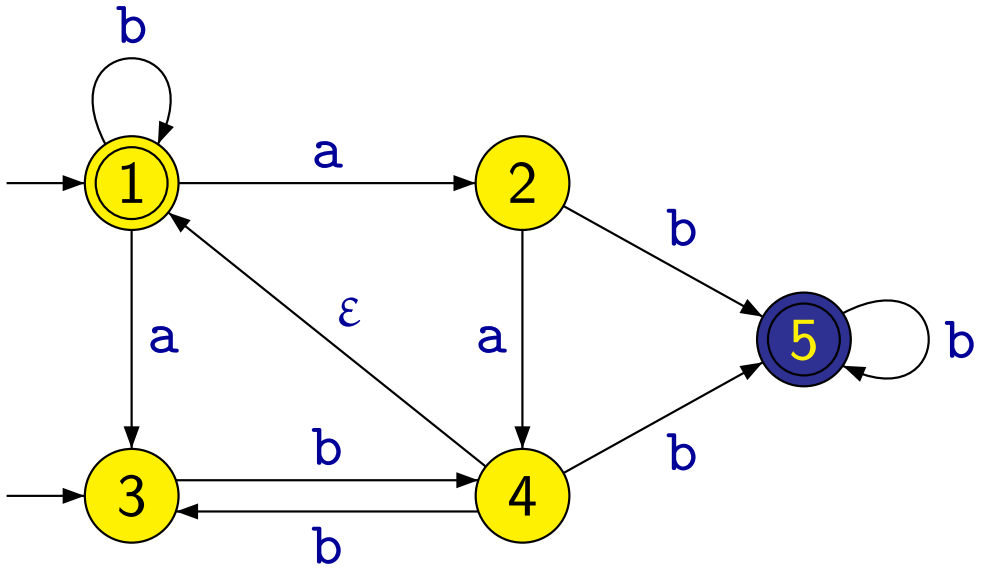
Zobecněný nedeterministický konečný automat

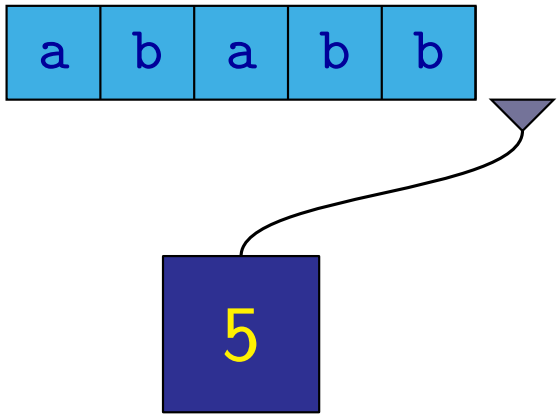




$$1 \xrightarrow{a} 3 \xrightarrow{b} 4 \xrightarrow{\varepsilon} 1 \xrightarrow{a} 2 \xrightarrow{b} 5$$

Zobecněný nedeterministický konečný automat





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

Zobecněný nedeterministický konečný automat

Oproti nedeterministickému konečnému automatu má **zobecněný nedeterministický konečný automat** tzv. ε**-přechody**, tj. přechody označené symbolem ε.

Při provádění ε-přechodu se mění pouze stav řídící jednotky, ale hlava na pásce se neposouvá.

Poznámka: Výpočty zobecněného nedeterministického automatu mohou být libovolně dlouhé a dokonce i nekonečné (pokud graf obsahuje cyklus tvořený ε-přechody) bez ohledu na délku slova na pásce.

Zobecněný nedeterministický konečný automat

Formálně je **zobecněný nedeterministický konečný automat** (**ZNKA**) definován jako pětice

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

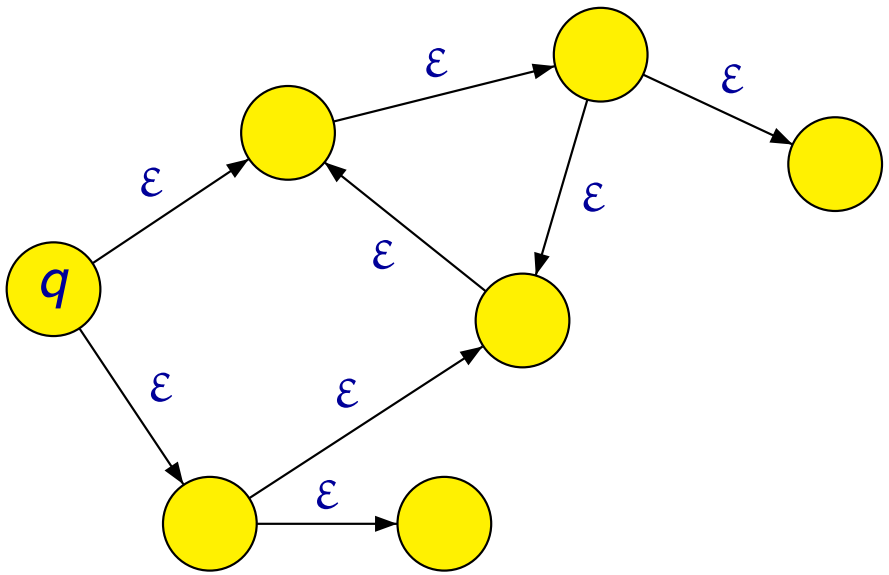
kde:

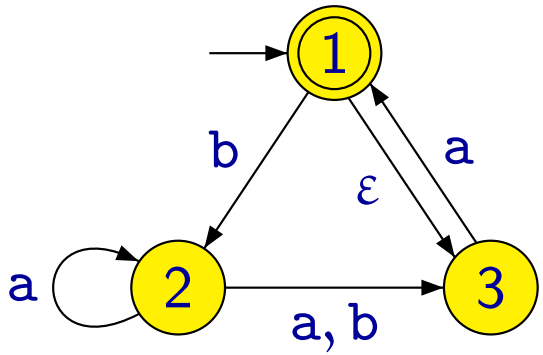
- Q je konečná množina **stavů**
- Σ je konečná **abeceda**
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ je **přechodová funkce**
- $I \subseteq Q$ je množina **počátečních stavů**
- $F \subseteq Q$ je množina **přijímajících stavů**

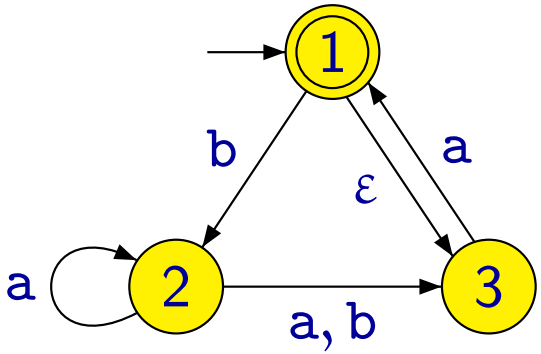
Poznámka: Na NKA můžeme nahlížet jako na speciální případ ZNKA, kde $\delta(q, \varepsilon) = \emptyset$ pro všechna $q \in Q$.

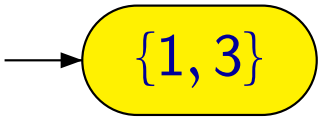
Převod na deterministický konečný automat

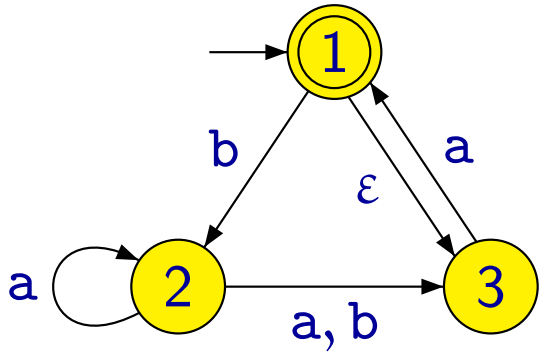
Zobecněný nedeterministický konečný automat je možné převést na deterministický podobnou konstrukcí jako nedeterministický konečný automat, s tím rozdílem, že do množin stavů musíme vždy přidat navíc i všechny stavy dosažitelné z již přidaných stavů nějakou sekvencí ε-přechodů.

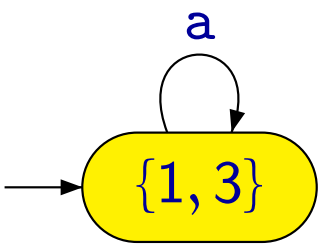


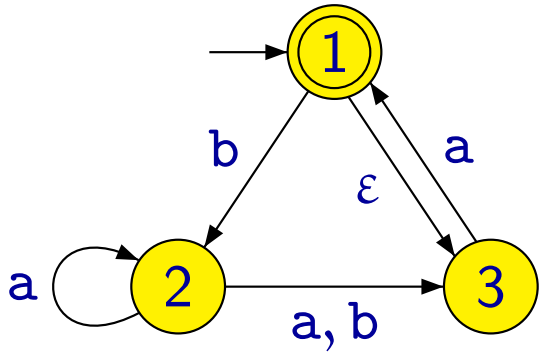

1
b
ε
a
a
2
3
a, b

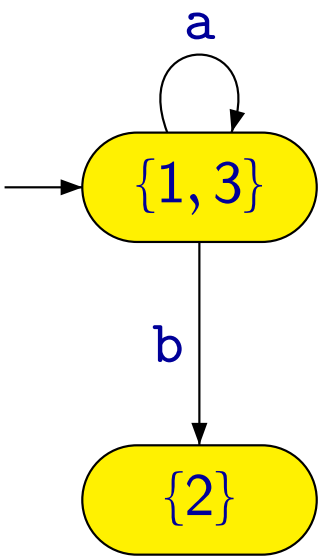

1
b
a
ε
a
2
3
a, b

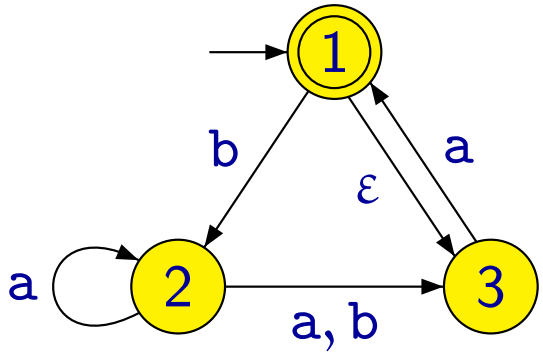

{1, 3}

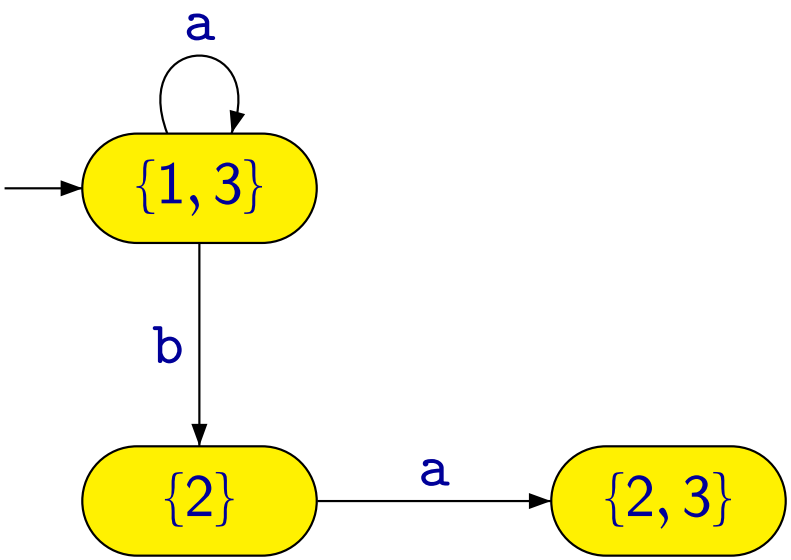

1
b
a
ε
a
2
3
a, b

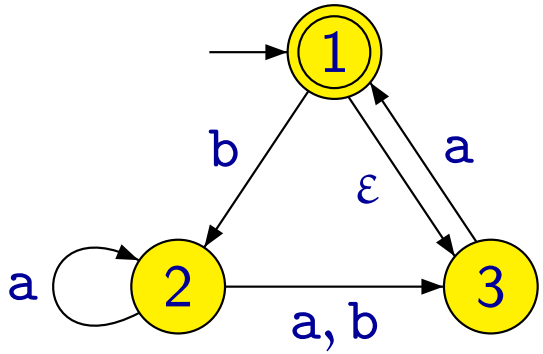

a
{1, 3}

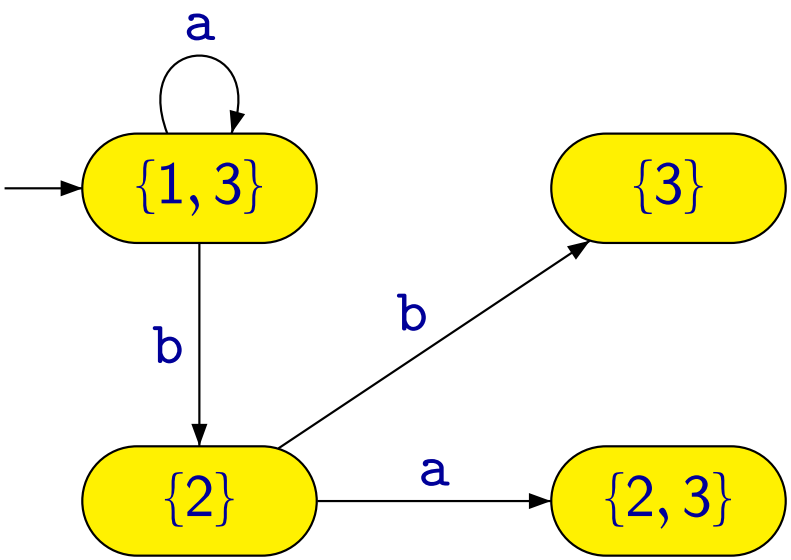

1
b
a
ε
a
2
3
a, b

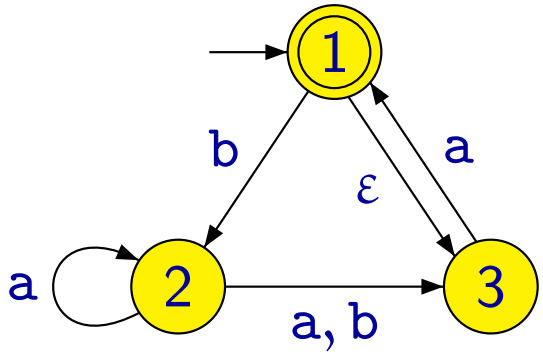

a
{1, 3}
b
{2}

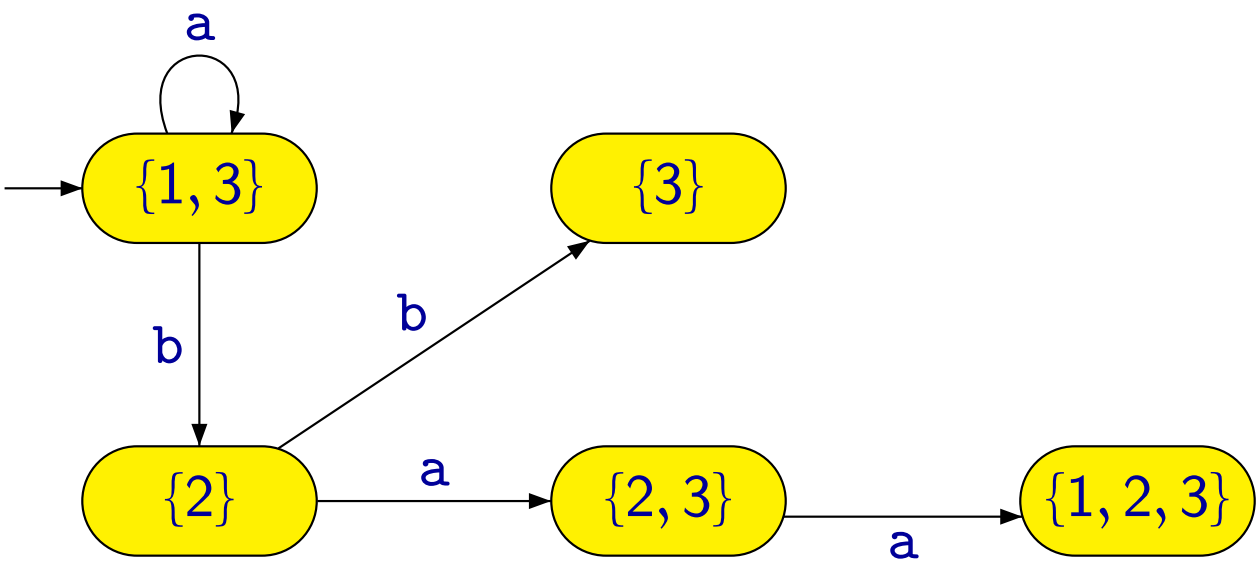

1
b
a
ε
a
2
3
a, b

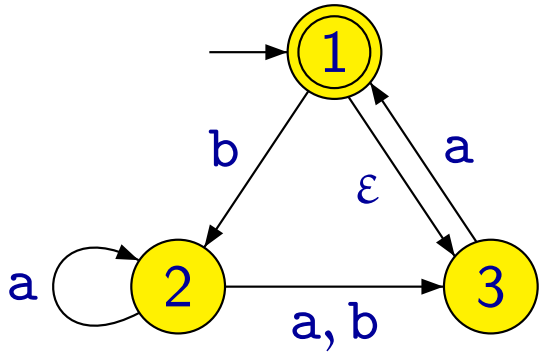

a
{1, 3}
b
{2}
a
{2, 3}

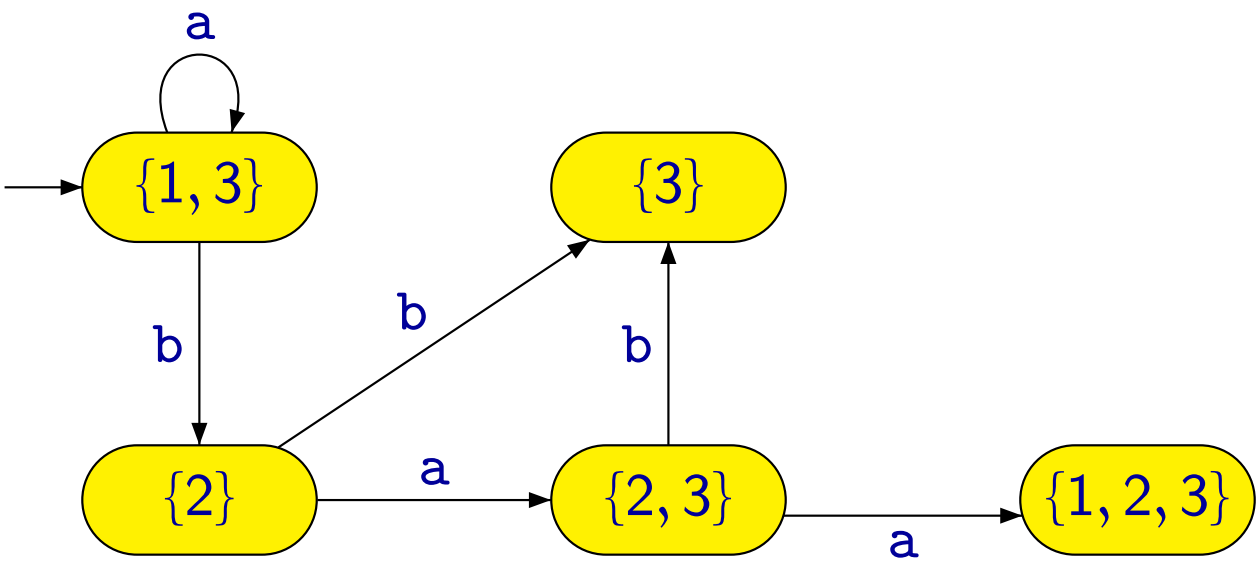

1
b
ε
a
a
2
3
a, b

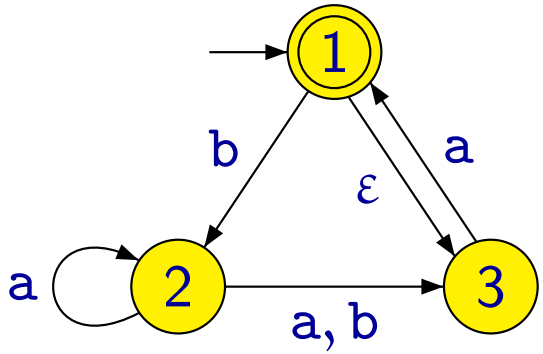

a
{1, 3}
{3}
b
b
{2}
a
{2, 3}

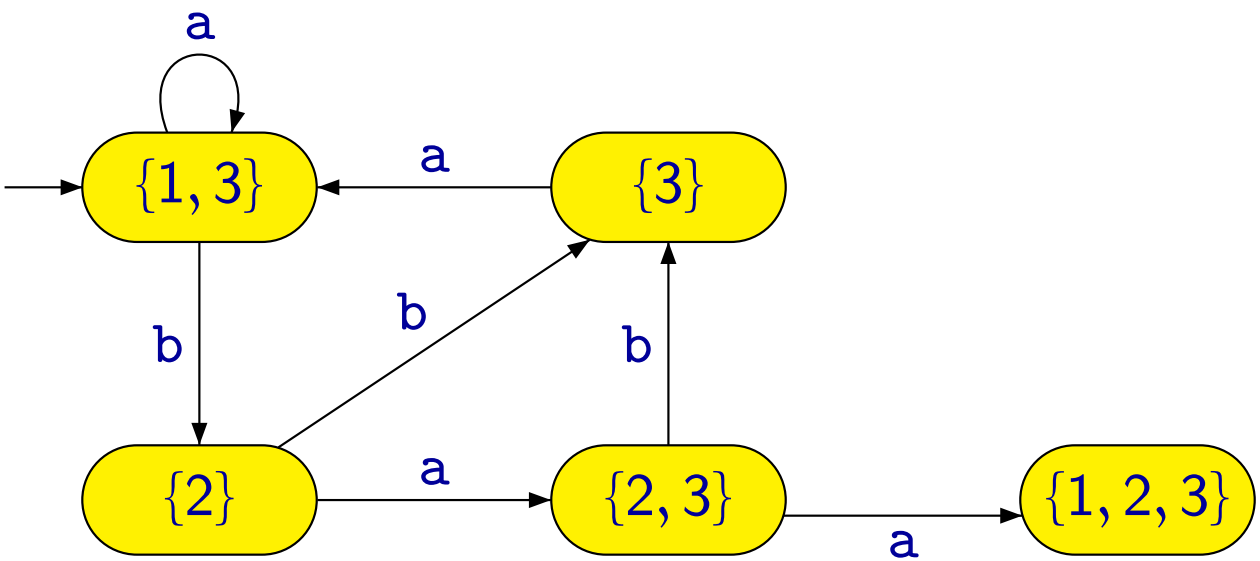

1
b
ε
a
a
2
3
a, b

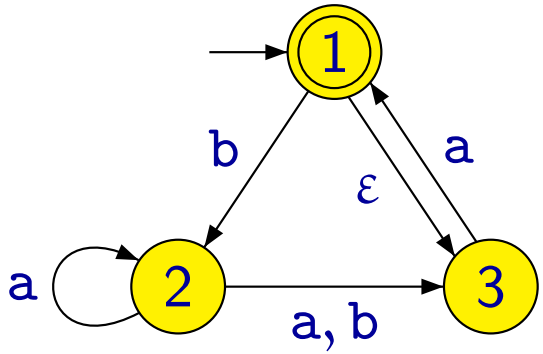

a
{1, 3}
{3}
b
b
{2}
a
{2, 3}
{1, 2, 3}
a

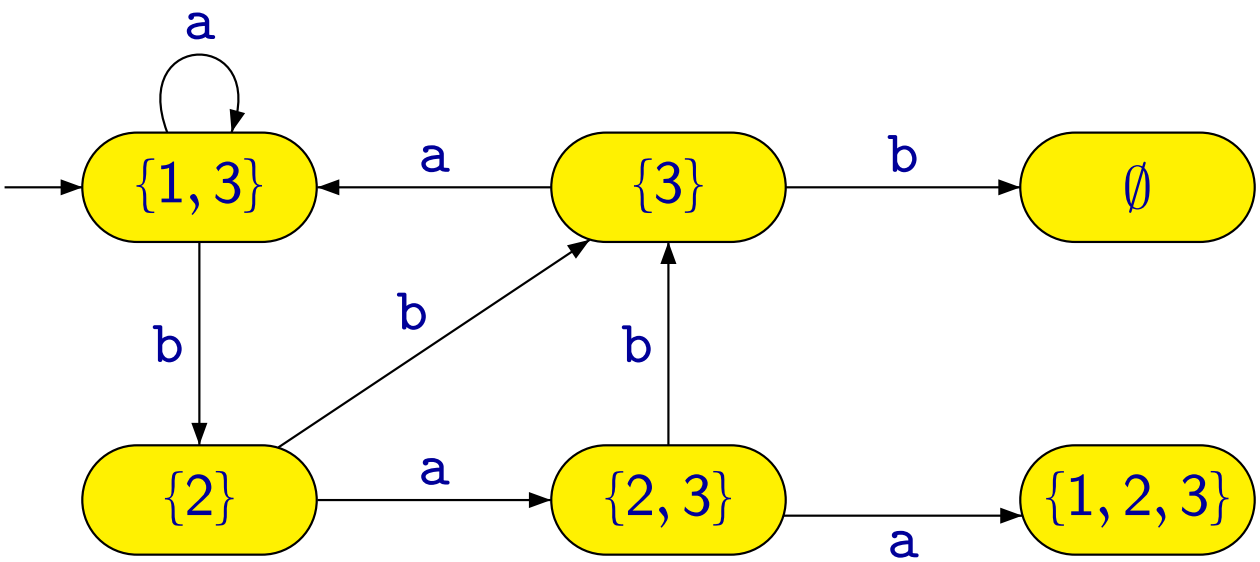

1
b
ε
a
a
2
3
a, b

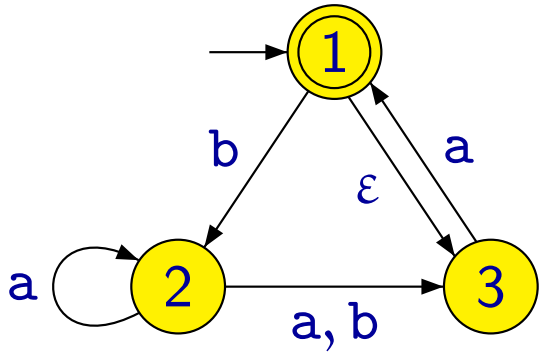

a
{1, 3}
{3}
b
b
b
{2}
a
{2, 3}
{1, 2, 3}
a

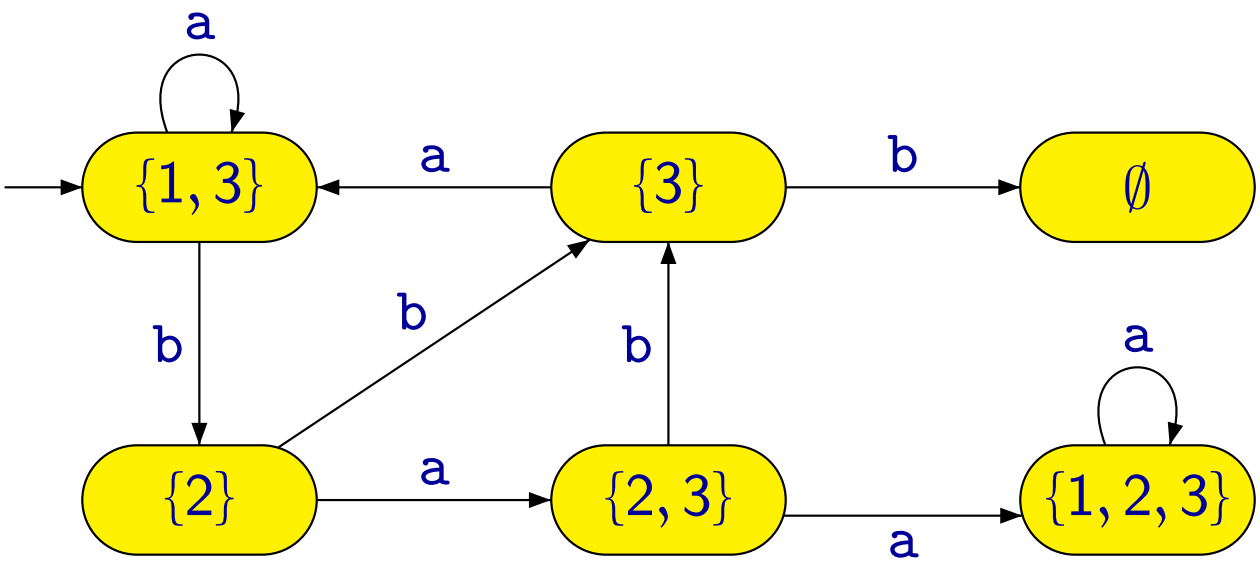

1
b
ε
a
a
2
3
a, b

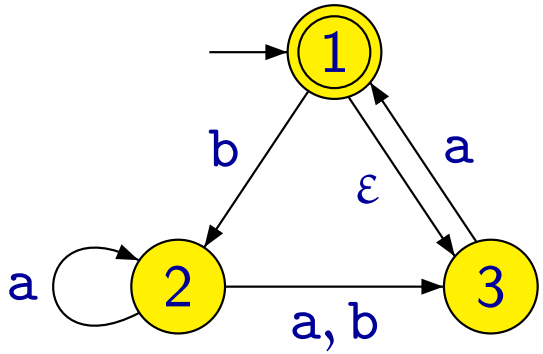

a
{1, 3}
a
{3}
b
b
b
{2}
a
{2, 3}
a
{1, 2, 3}

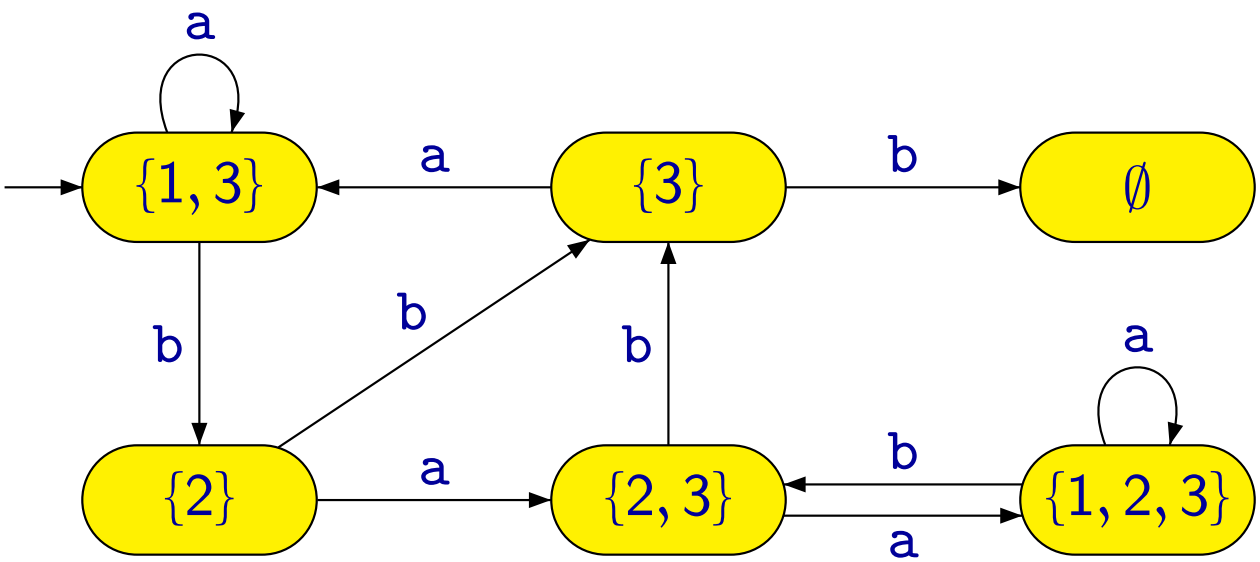

1
b
ε
a
a
2
3
a, b

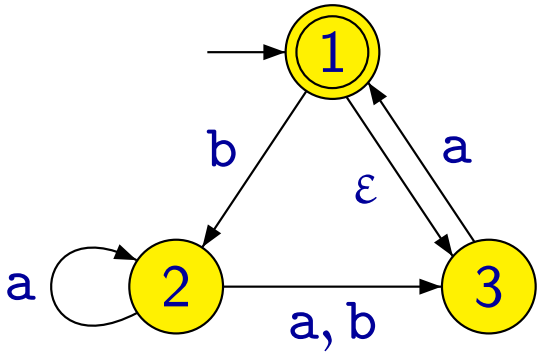

a
{1, 3}
a
{3}
b
∅
b
b
b
{2}
a
{2, 3}
{1, 2, 3}
a

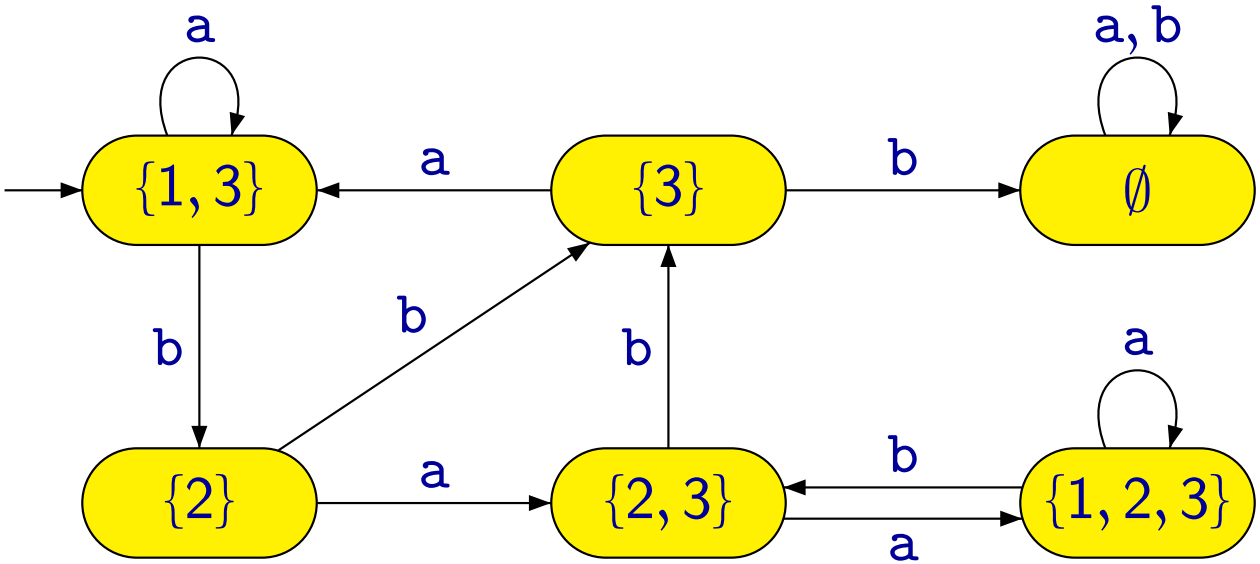

1
b
a
ε
a
2
3
a, b

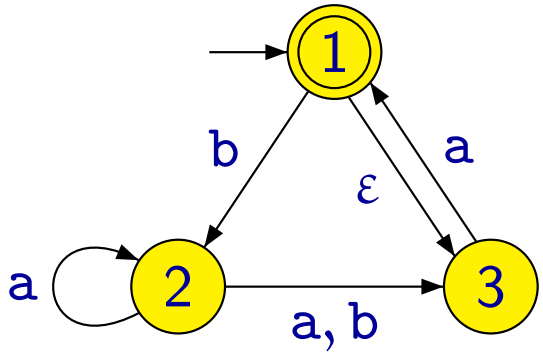

a
{1, 3}
a
{3}
b
∅
b
b
b
a
{2}
a
{2, 3}
{1, 2, 3}
a

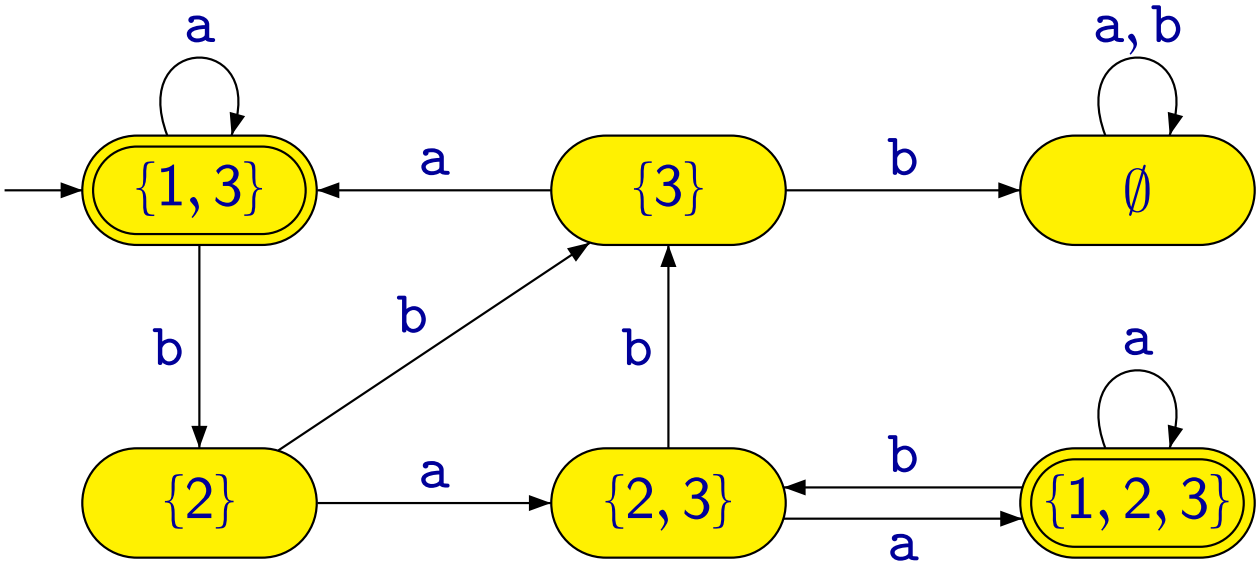

1
b
a
ε
a
2
3
a, b


a
{1, 3}
a
{3}
b
∅
b
b
b
a
{2}
a
{2, 3}
b
{1, 2, 3}
a


1
b
a
ε
a
2
3
a, b


a
a, b
{1, 3}
a
{3}
b
∅
b
b
b
a
{2}
a
{2, 3}
b
{1, 2, 3}
a


1
b
ε
a
a
2
3
a, b


a
a, b
{1, 3}
a
{3}
b
∅
b
b
b
a
{2}
a
{2, 3}
b
{1, 2, 3}
a

Převod ZNKA na DKA

Předtím, než formálně popíšeme převod ZNKA na DKA, zaveďme si několik pomocných definic.

Předpokládejme nějaký daný ZNKA $A = (Q, \Sigma, \delta, I, F)$.

Definujme funkci $\hat{\delta} : \mathcal{P}(Q) \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ tak, že pro $K \subseteq Q$ a $a \in \Sigma \cup \{\varepsilon\}$ je

$$\hat{\delta}(K, a) = \bigcup_{q \in K} \delta(q, a)$$

Převod ZNKA na DKA

Pro $K \subseteq Q$ označme $Cl_\varepsilon(K)$ množinu všech stavů dosažitelných ze stavů z množiny K nějakou libovolnou sekvencí ε-přechodů.

To znamená, že funkce $Cl_\varepsilon : \mathcal{P}(Q) \to \mathcal{P}(Q)$ je definována tak, že pro $K \subseteq Q$ je $Cl_\varepsilon(K)$ nejmenší (vzledem k inkluzi) množina splňující následující dvě podmínky:

- $K \subseteq Cl_\varepsilon(K)$
- Pro každé $q \in Cl_\varepsilon(K)$ platí, že $\delta(q, \varepsilon) \subseteq Cl_\varepsilon(K)$.

Poznámka: Všimněme si, že pro libovolné K je $Cl_\varepsilon(Cl_\varepsilon(K)) = Cl_\varepsilon(K)$.

Všimněme si také, že v případě NKA (kde $\delta(q, \varepsilon) = \emptyset$ pro každé $q \in Q$) je $Cl_\varepsilon(K) = K$.

Převod ZNKA na DKA

K danému ZNKA $A = (Q, \Sigma, \delta, I, F)$ nyní můžeme sestrojit DKA $A' = (Q', \Sigma, \delta', q_0', F')$, kde:

- $Q' = \mathcal{P}(Q)$ ($K \in Q'$ tedy znamená, že $K \subseteq Q$)
- $\delta' : Q' \times \Sigma \to Q'$ je definová tak, že pro $K \in Q'$ a $a \in \Sigma$ je

$$\delta'(K, a) = Cl_\varepsilon(\hat{\delta}(Cl_\varepsilon(K), a))$$

- $q_0' = Cl_\varepsilon(I)$
- $F' = \{K \in Q' \mid Cl_\varepsilon(K) \cap F \neq \emptyset\}$

Není težké ověřit, že $L(A) = L(A')$.

Uzavřenost třídy regulárních jazyků

Množina (všech) regulárních jazyků je uzavřená vůči celé řadě různých operací:

- sjednocení
- průnik
- doplněk
- zřetězení
- iterace
- ...

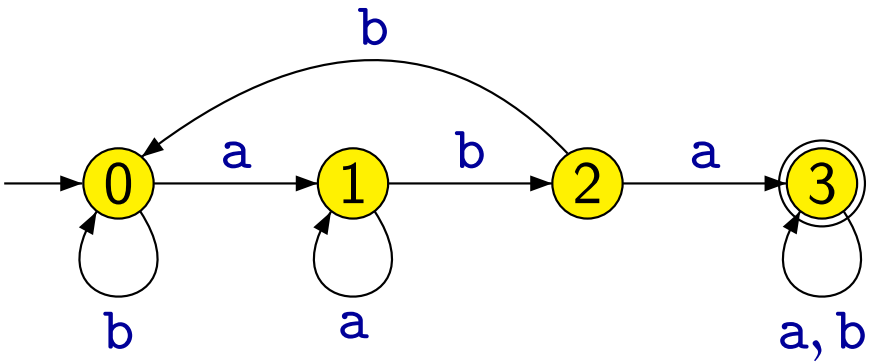
Například uzavřenost vůči sjednocení znamená toto:

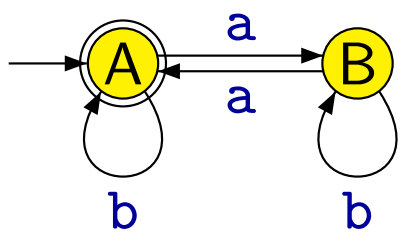
> *Jestliže jsou jazyky L_1 a L_2 regulární, pak také jazyk $L_1 \cup L_2$ je regulární.*

Poznámka: Jazyk je regulární, jestliže existuje konečný automat, který ho rozpoznává.

Automat pro průnik jazyků

Máme následující dva automaty:

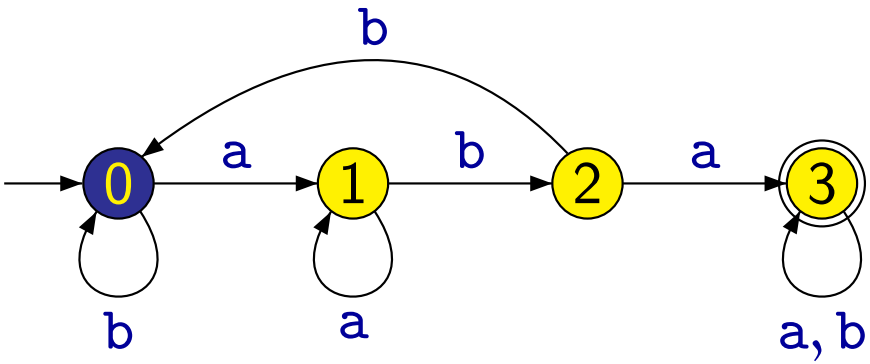


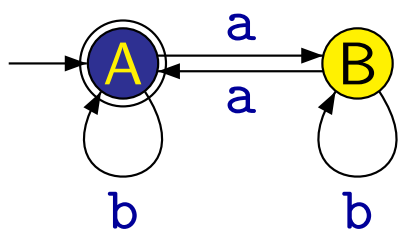


Příjmou oba slovo ababb?

Automat pro průnik jazyků

Máme následující dva automaty:

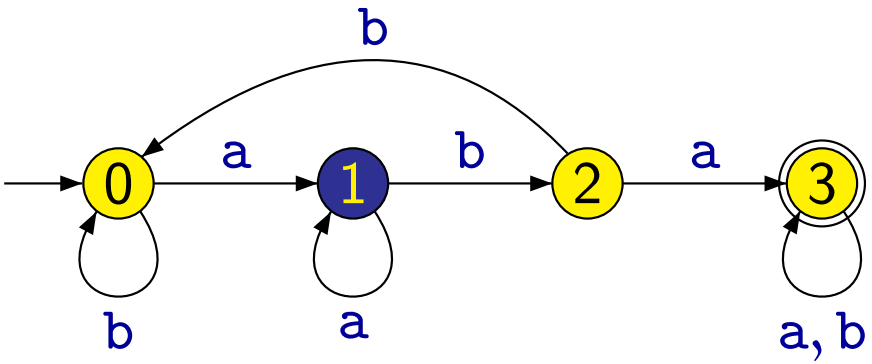


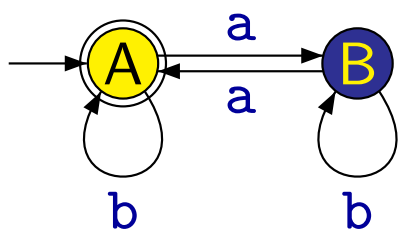


Příjmou oba slovo ababb?

Automat pro průnik jazyků

Máme následující dva automaty:

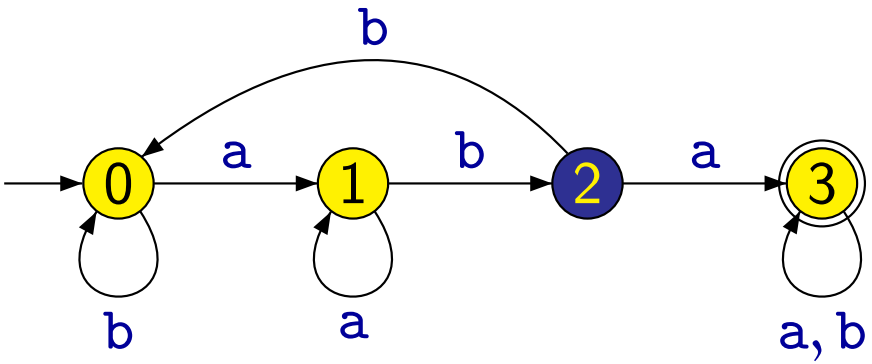


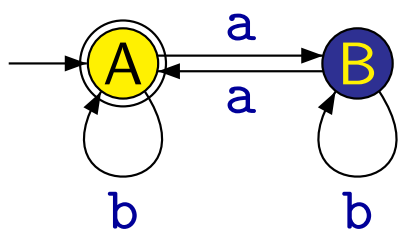


Příjmou oba slovo ababb?

Automat pro průnik jazyků

Máme následující dva automaty:

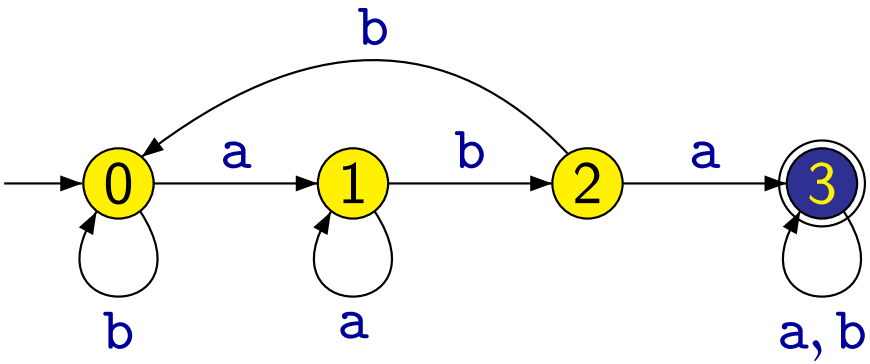


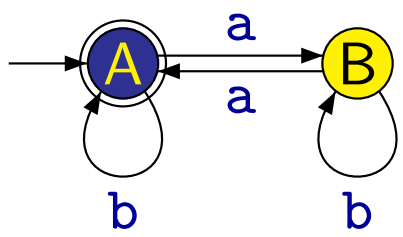


Příjmou oba slovo ababb?

Automat pro průnik jazyků

Máme následující dva automaty:

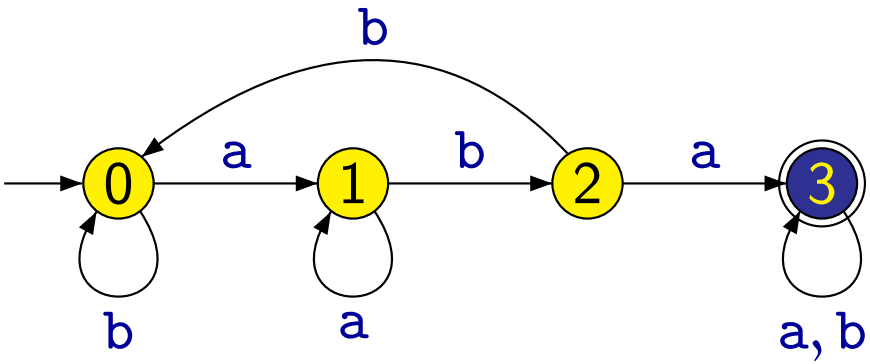


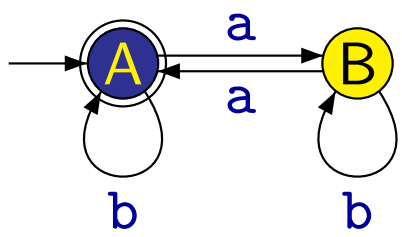


Příjmou oba slovo ababb?

Automat pro průnik jazyků

Máme následující dva automaty:

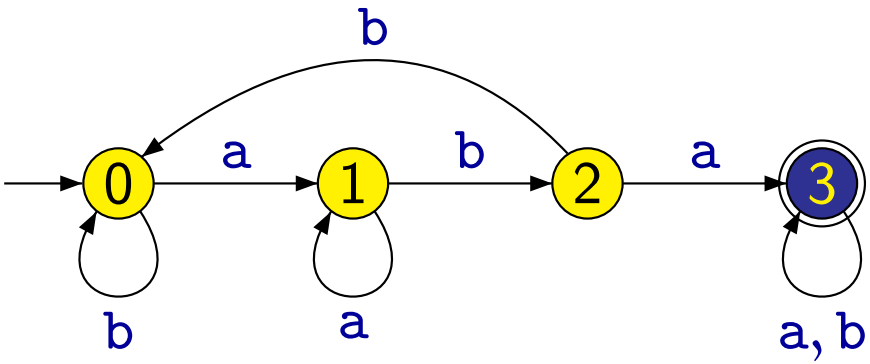


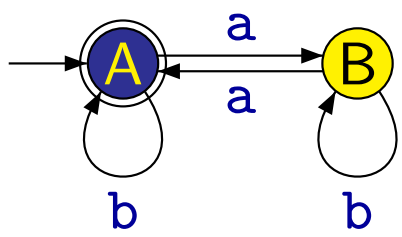


Příjmou oba slovo ababb?

Automat pro průnik jazyků

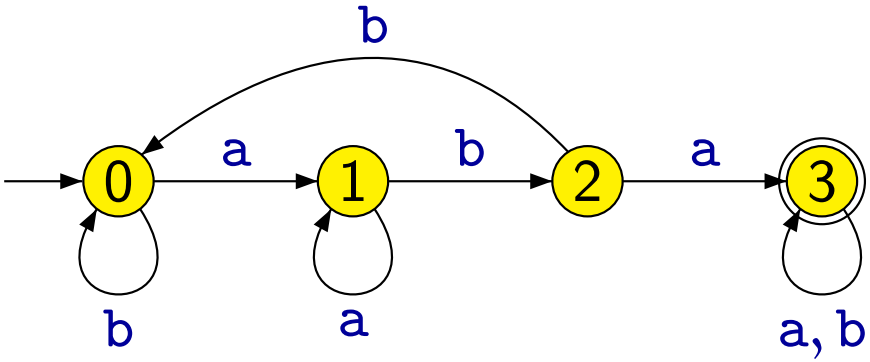
Máme následující dva automaty:

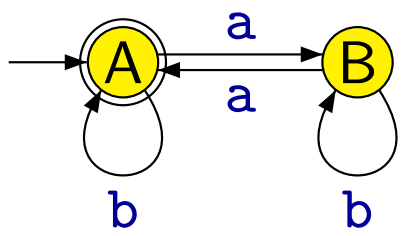




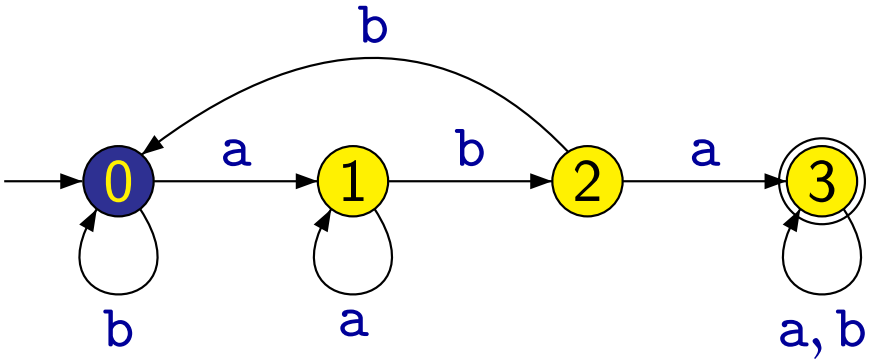
Příjmou oba slovo ababb?

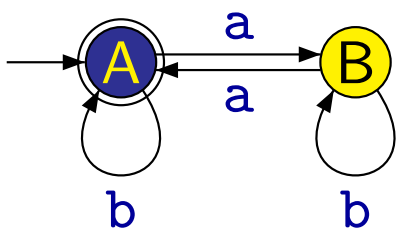
Automat pro průnik jazyků

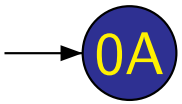




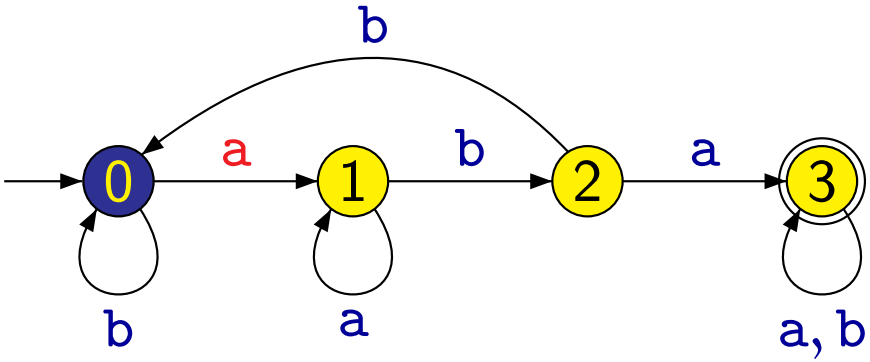
Automat pro průnik jazyků

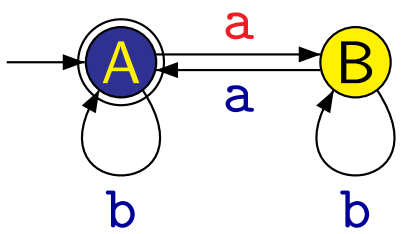




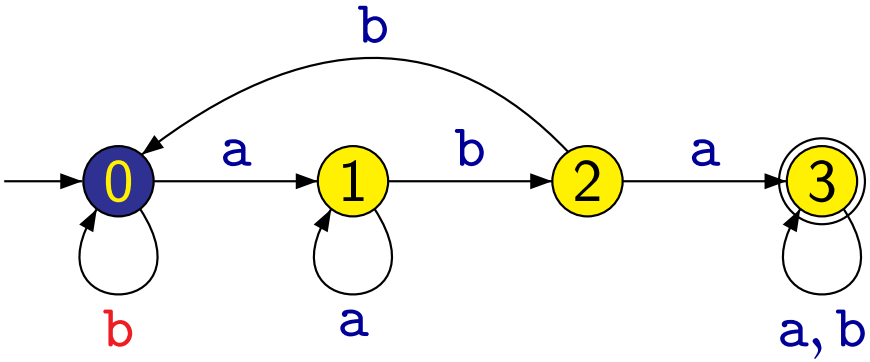


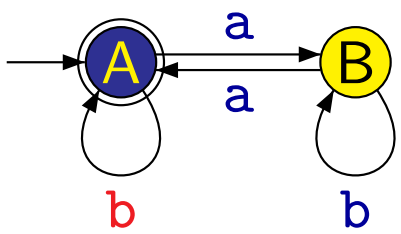
Automat pro průnik jazyků

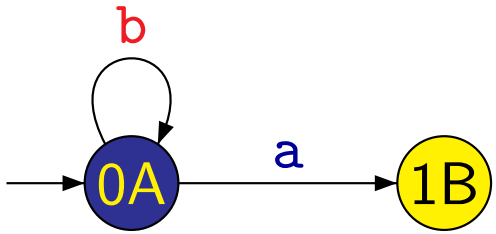




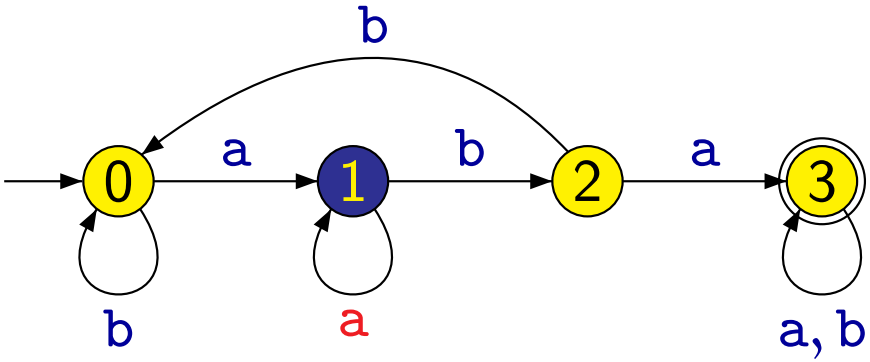
Automat pro průnik jazyků

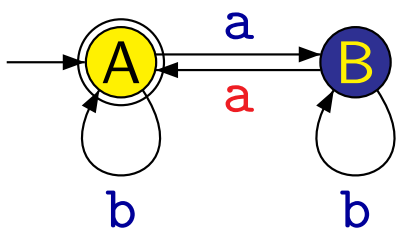


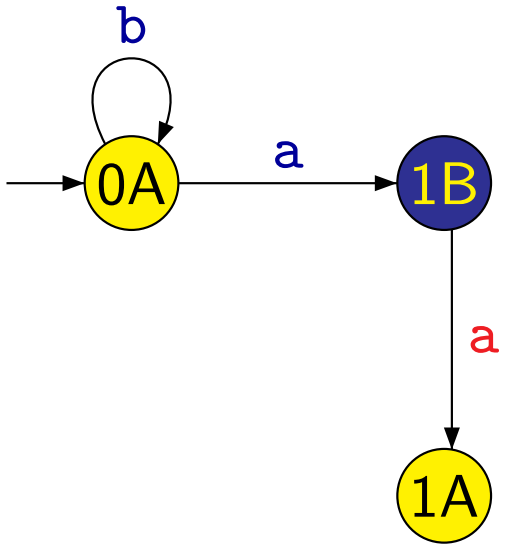




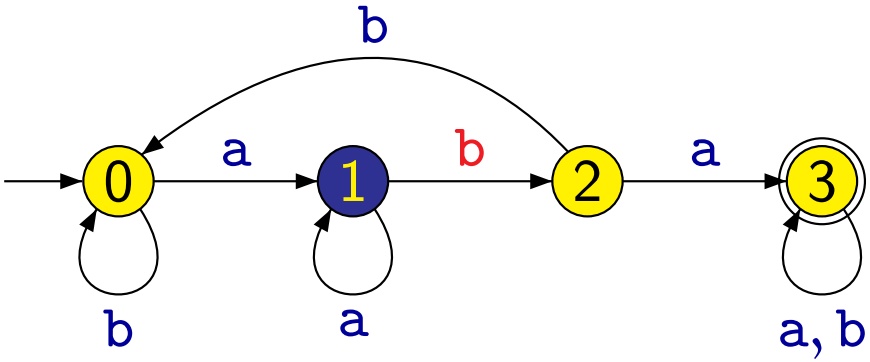
Automat pro průnik jazyků

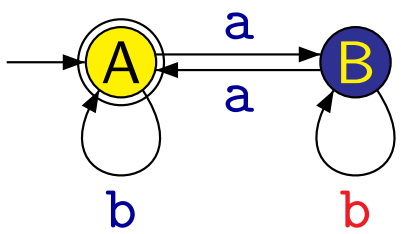


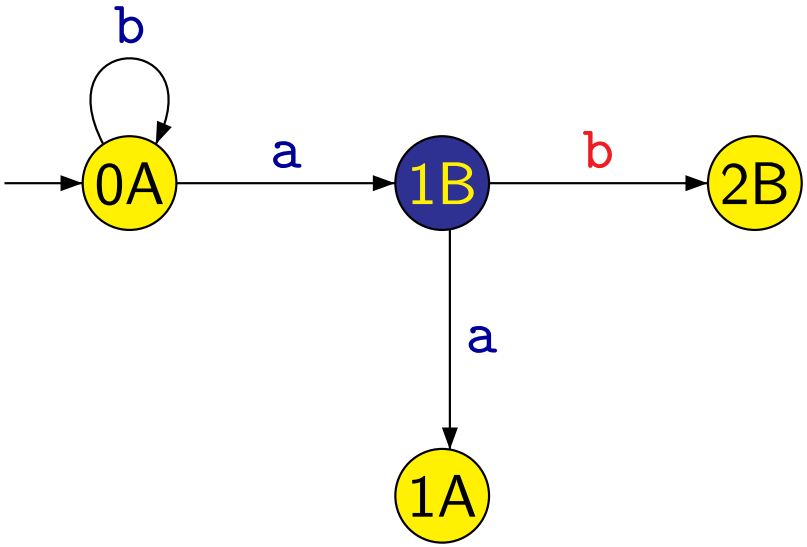




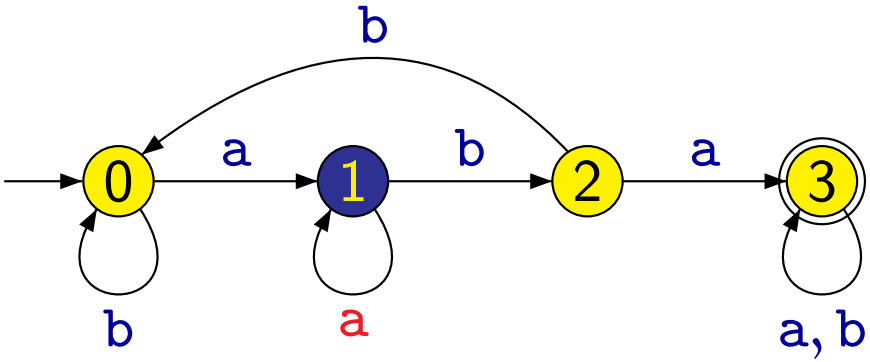
Automat pro průnik jazyků

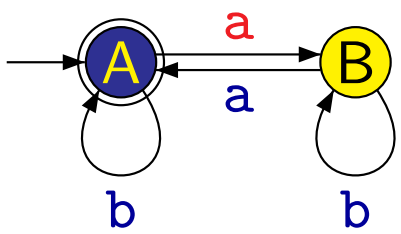


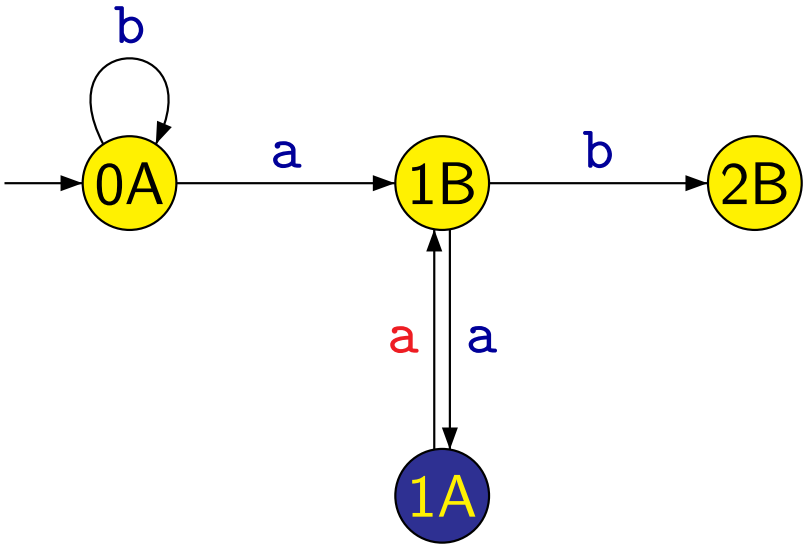




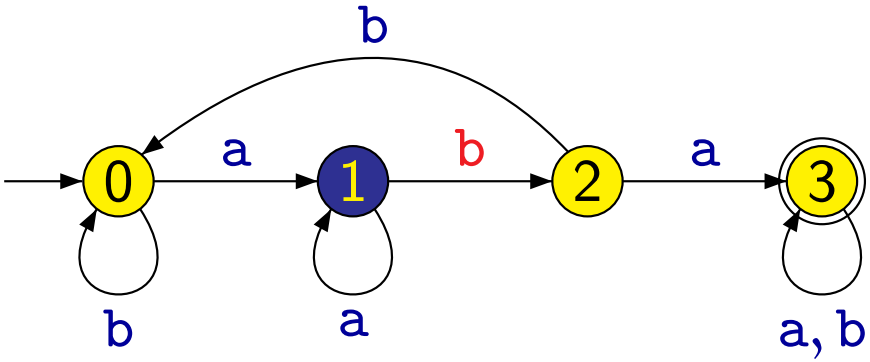
Automat pro průnik jazyků

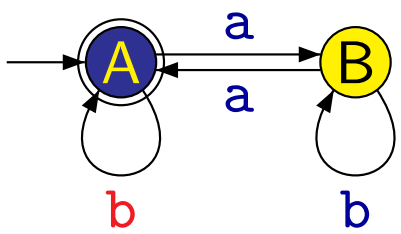


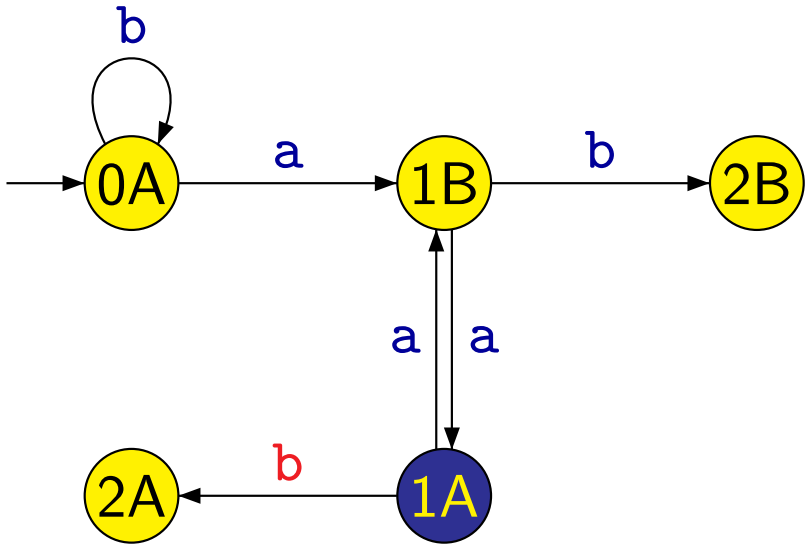




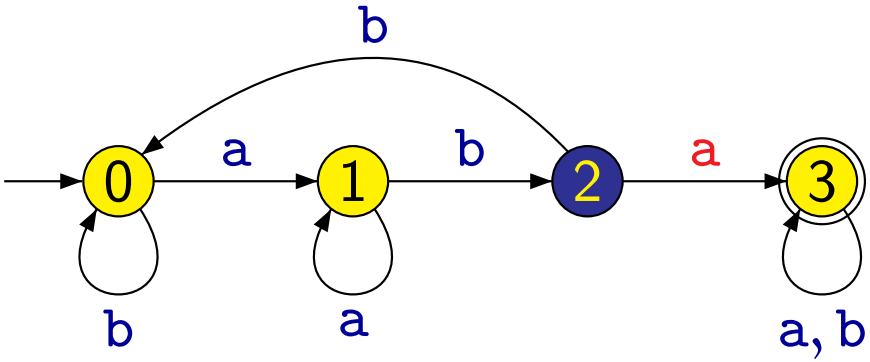
Automat pro průnik jazyků

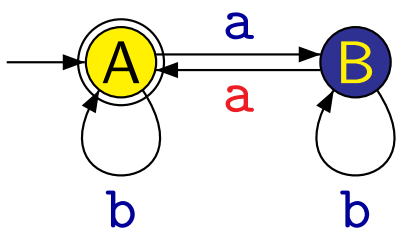


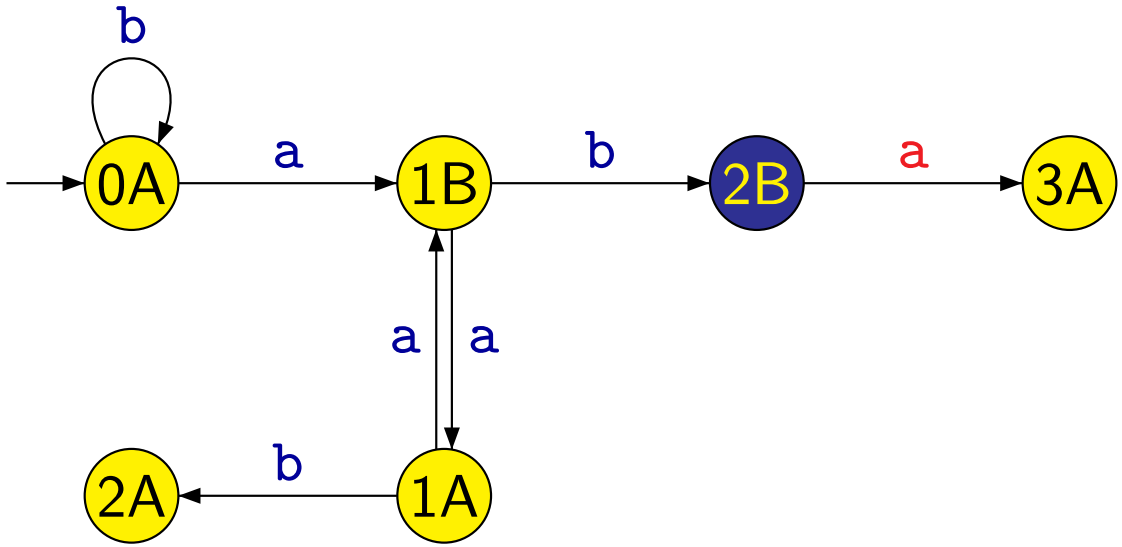




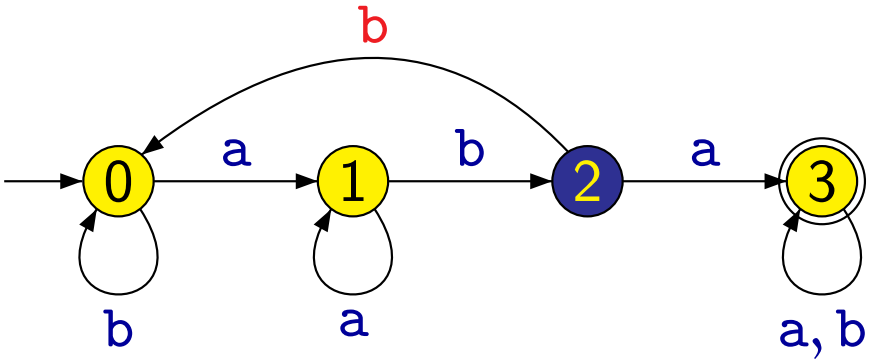
Automat pro průnik jazyků

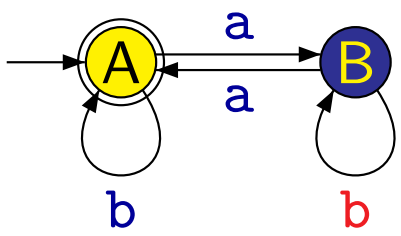


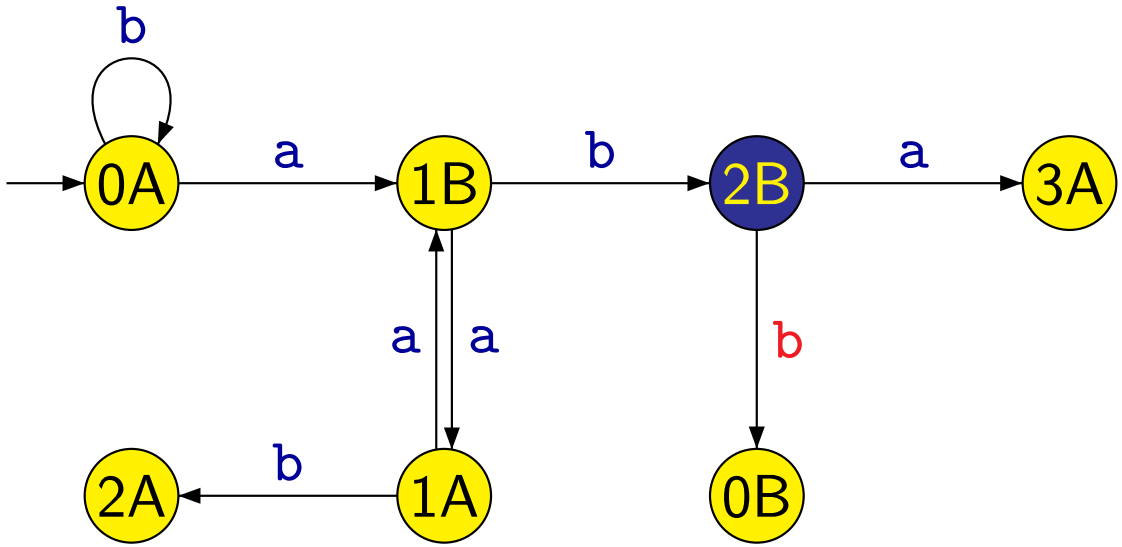




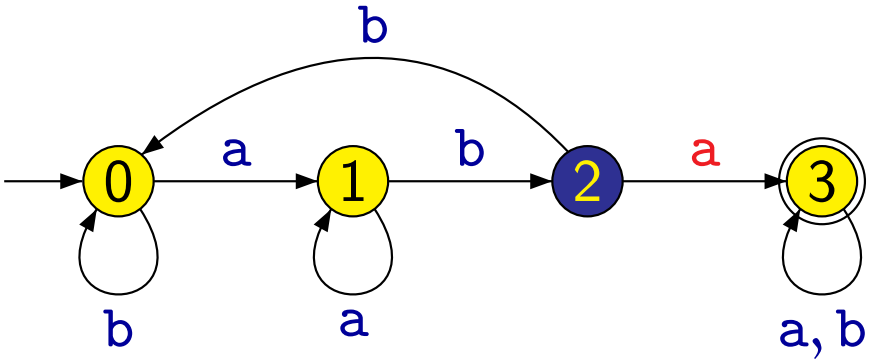
Automat pro průnik jazyků

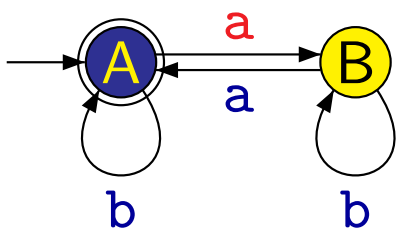


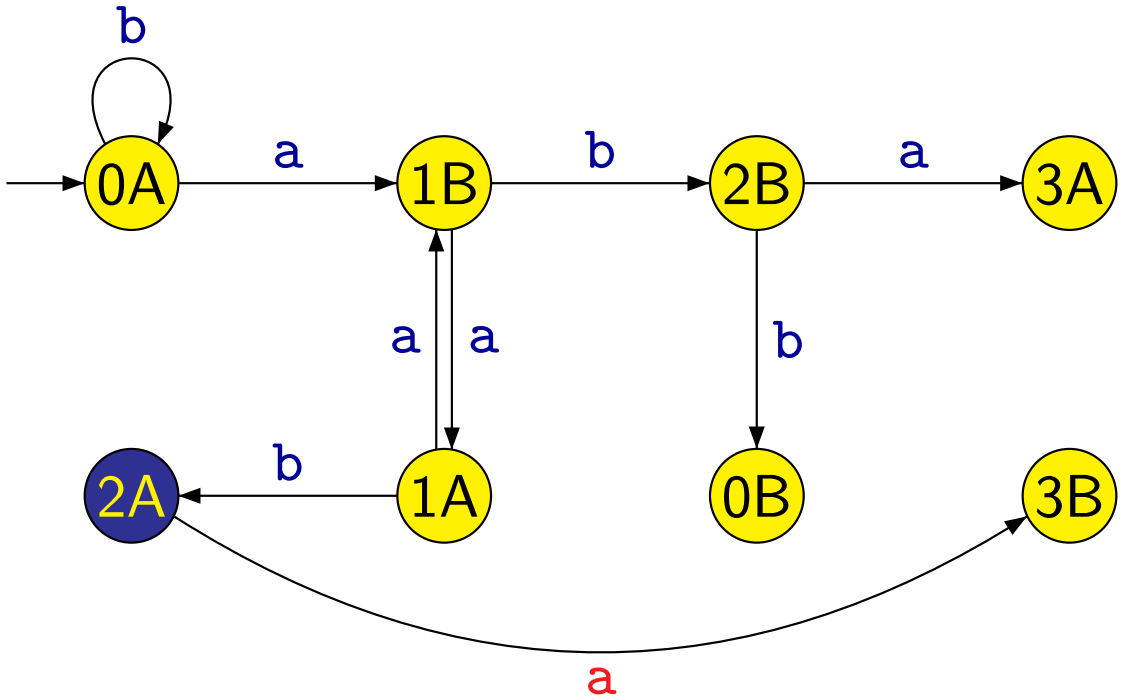




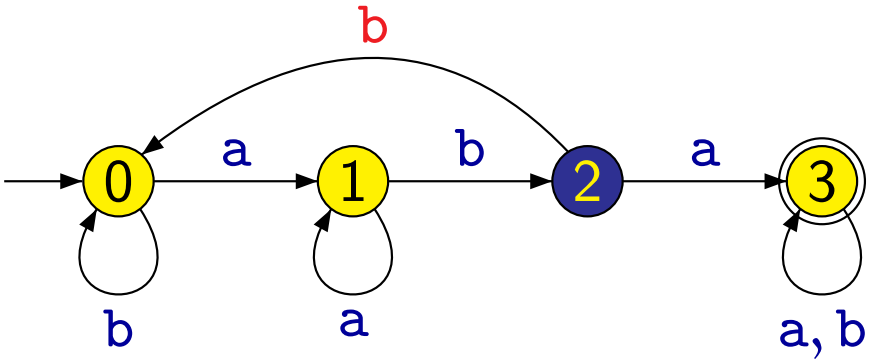
Automat pro průnik jazyků

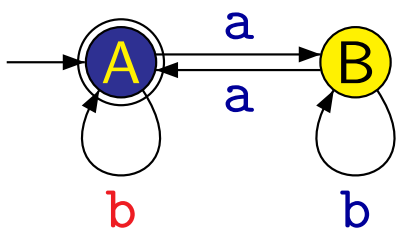


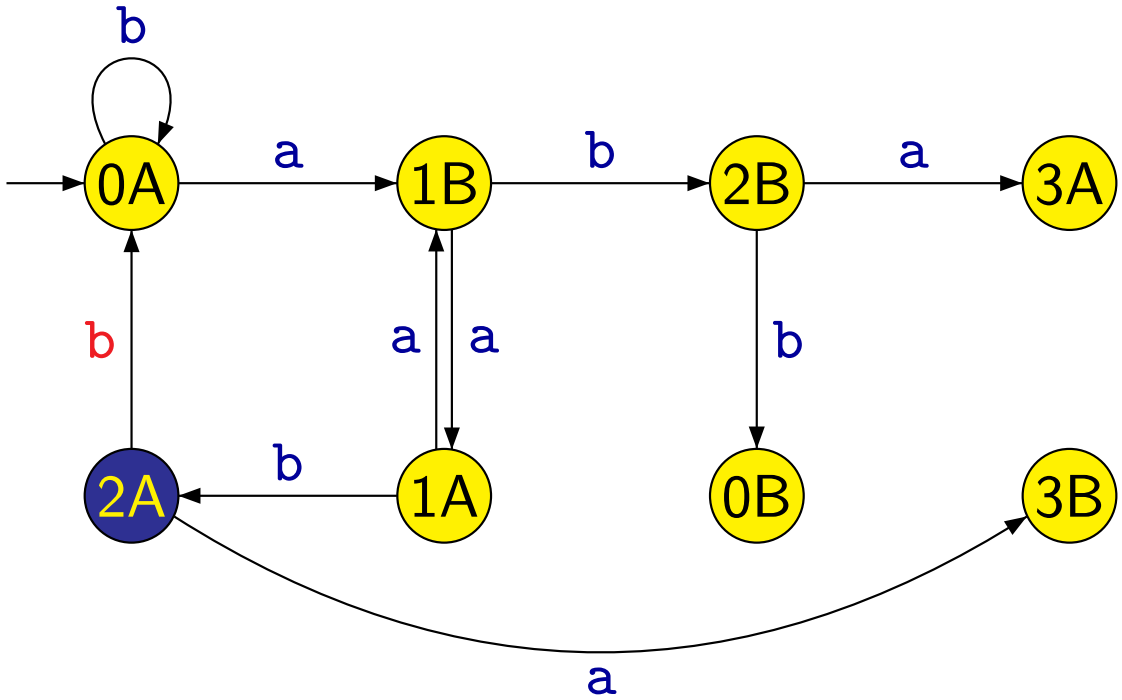




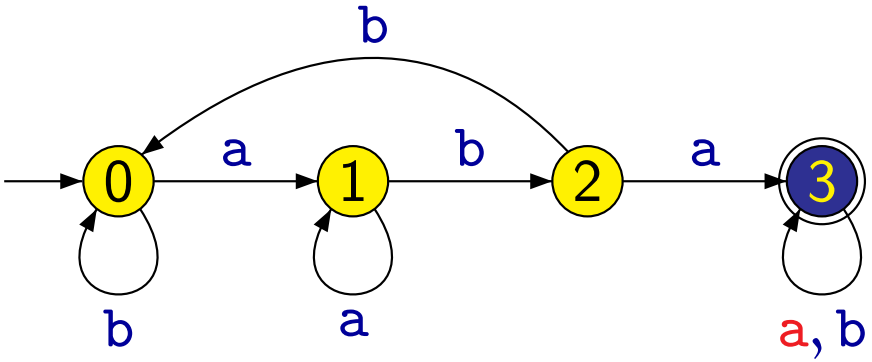
Automat pro průnik jazyků

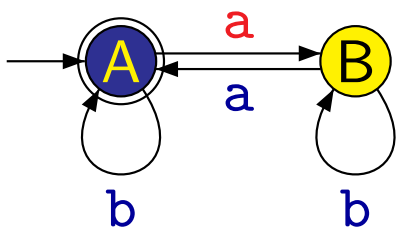


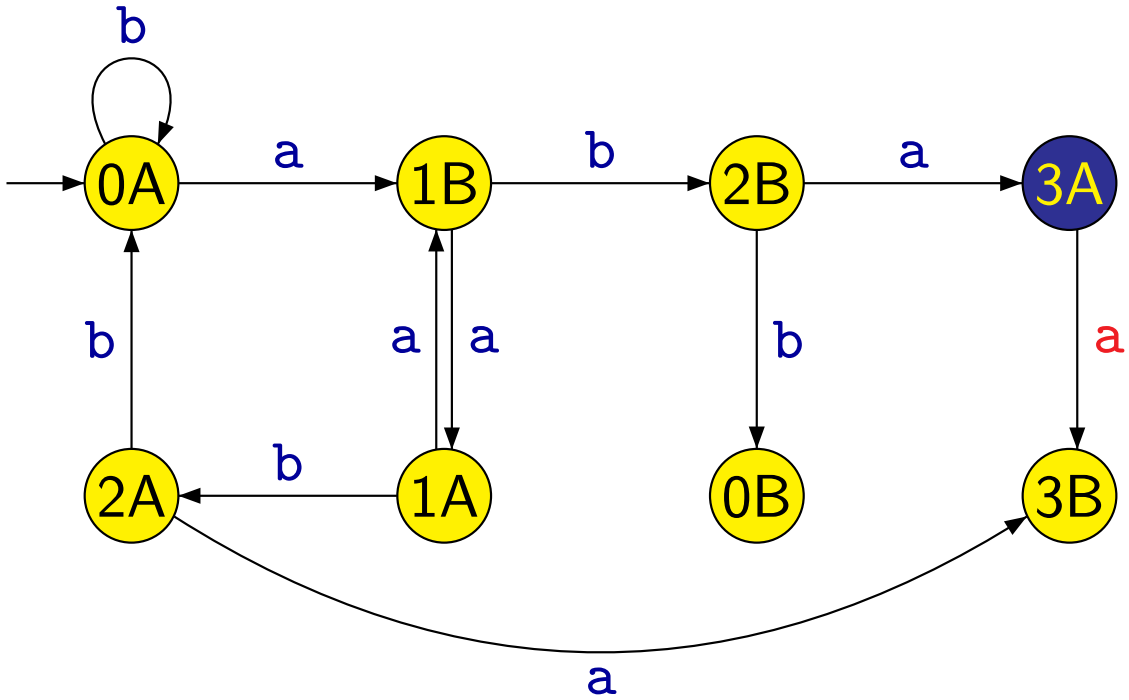




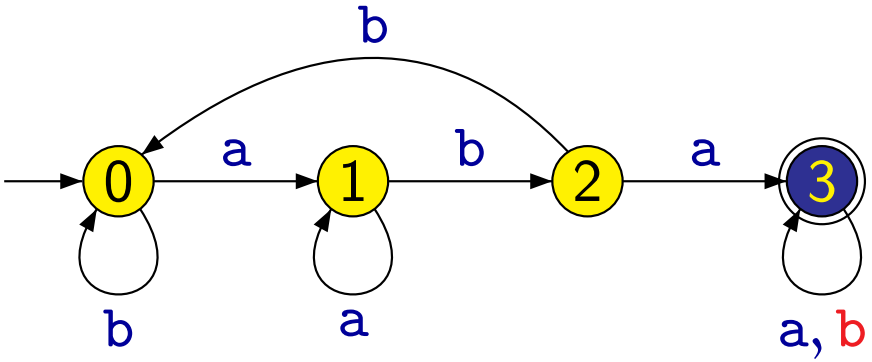
Automat pro průnik jazyků

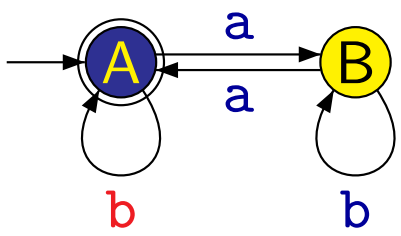


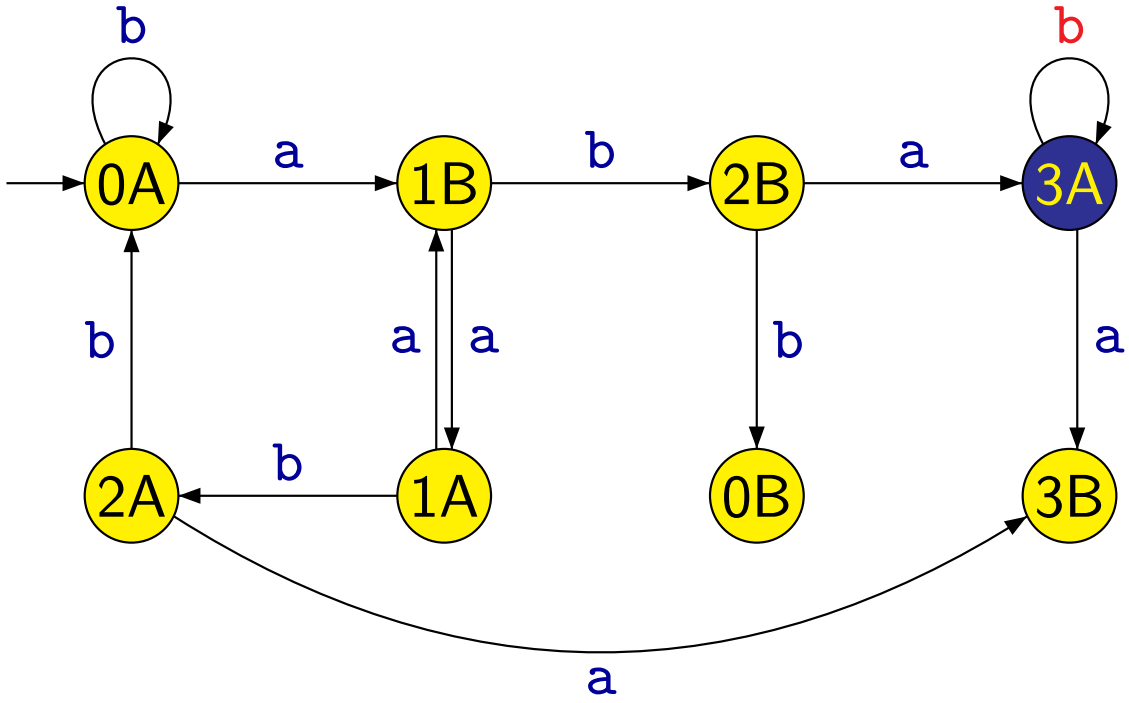




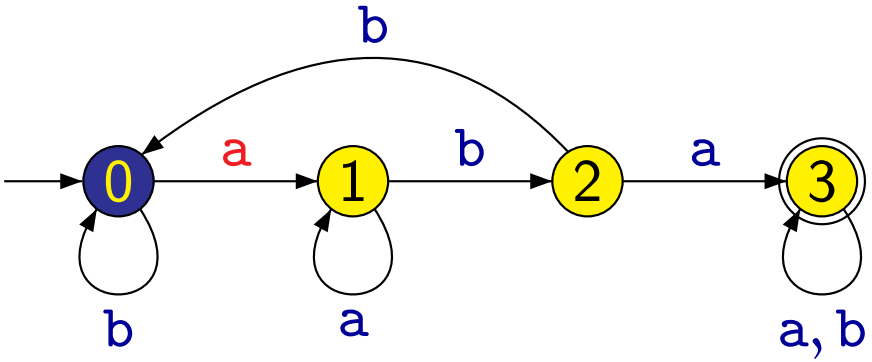
Automat pro průnik jazyků

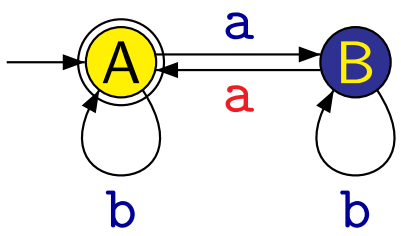


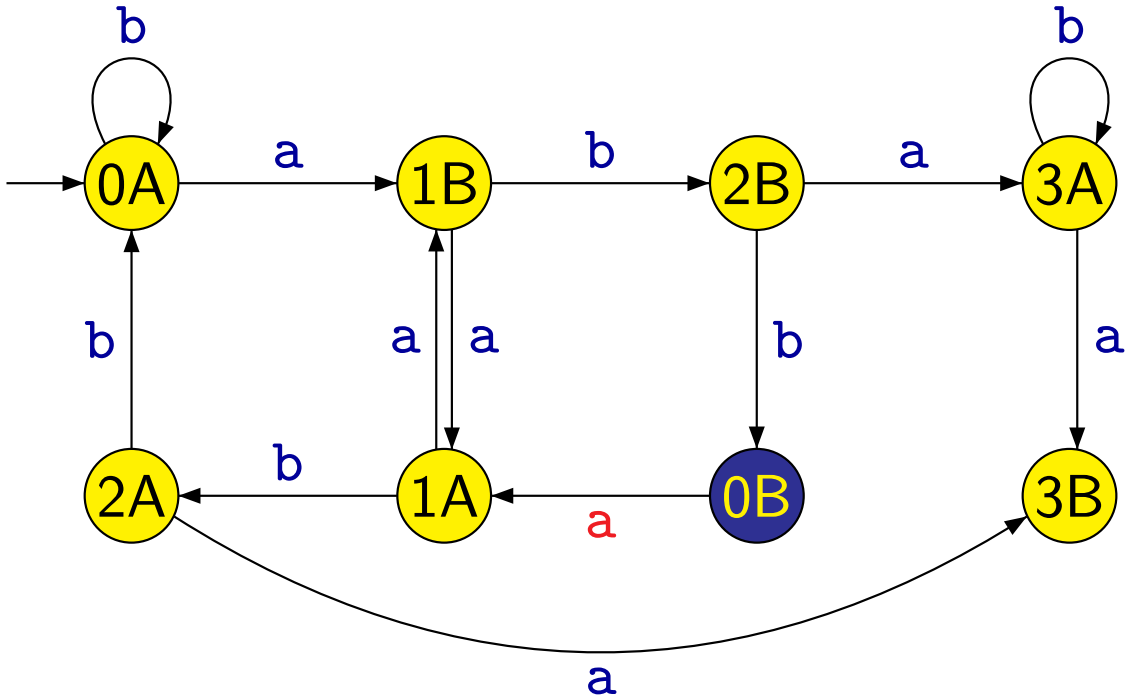




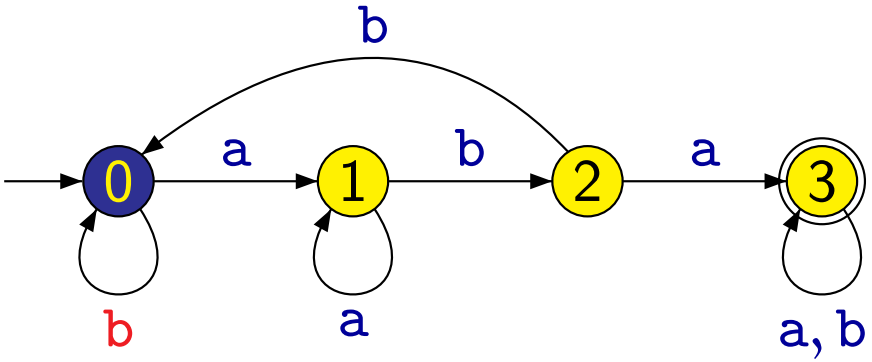
Automat pro průnik jazyků

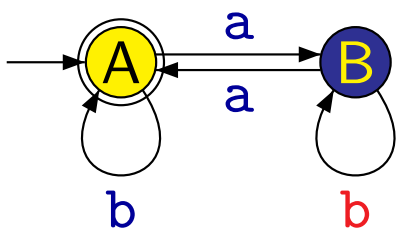


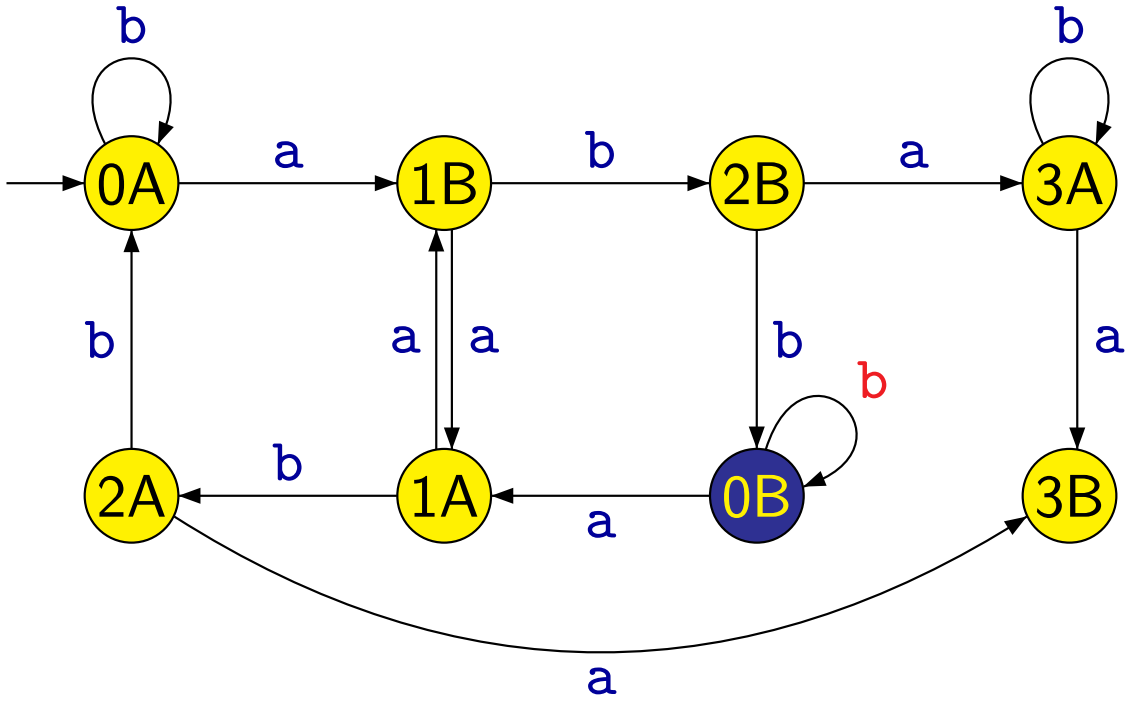




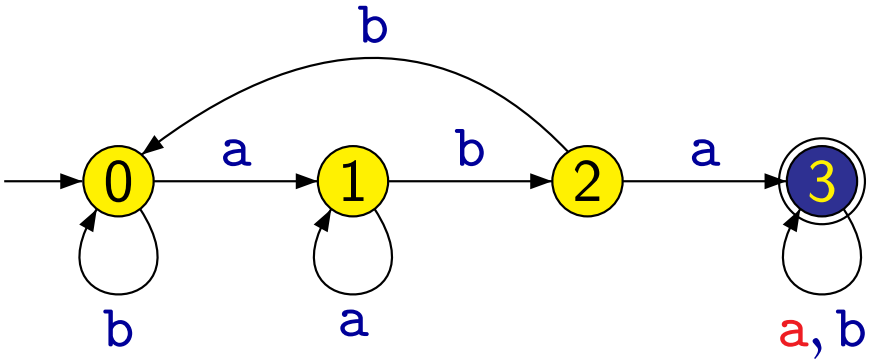
Automat pro průnik jazyků

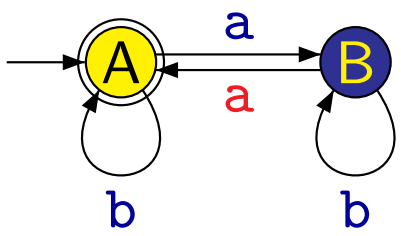


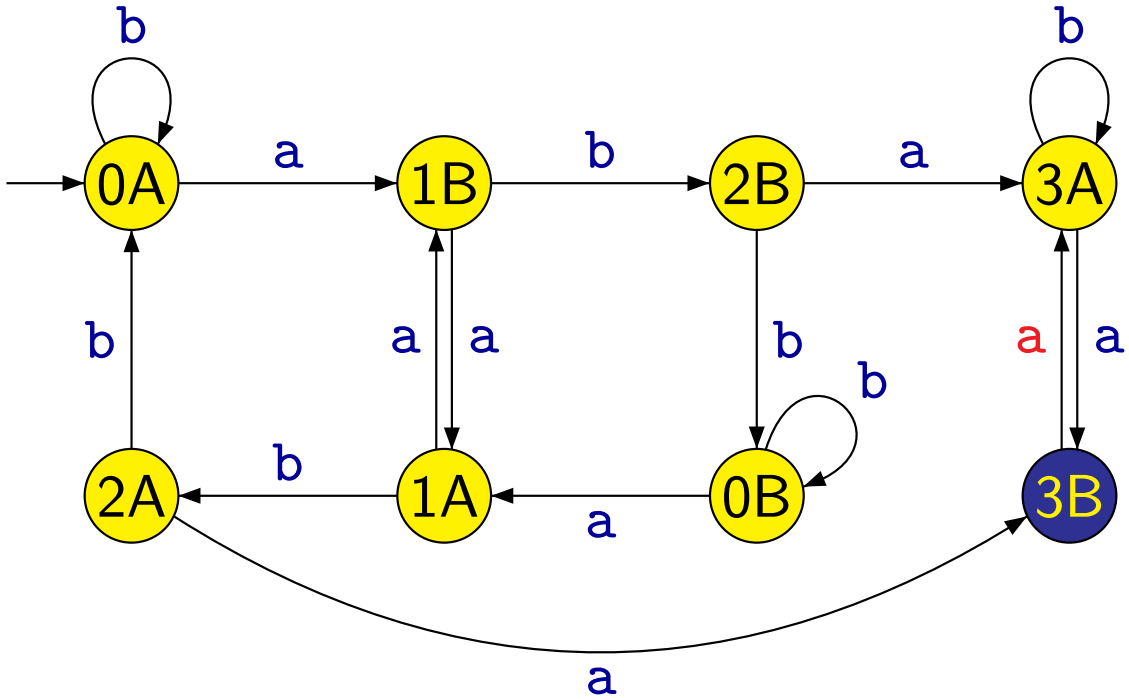




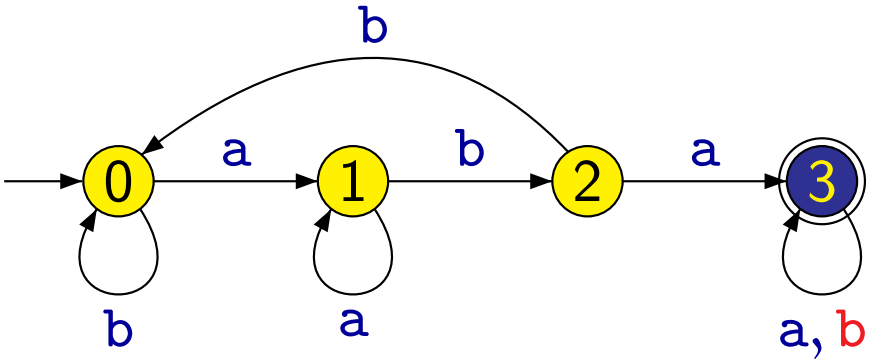
Automat pro průnik jazyků

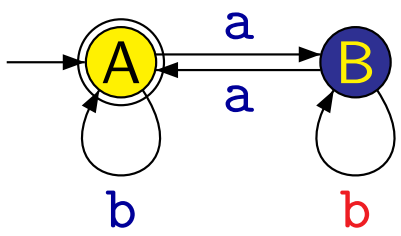


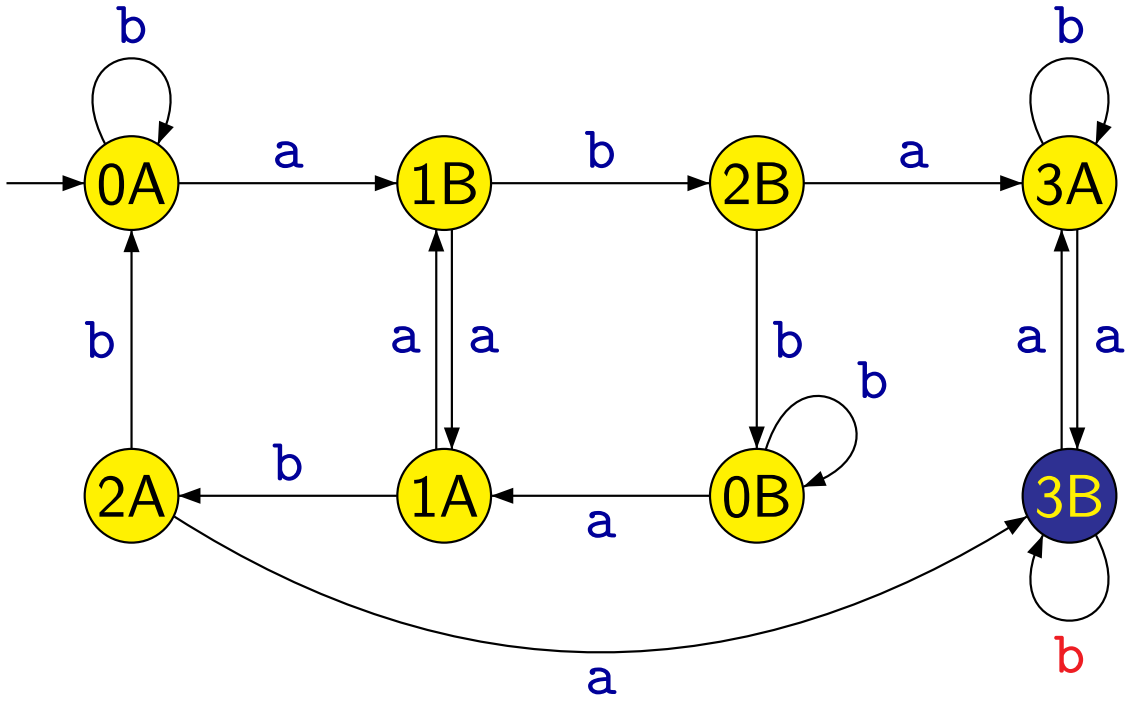




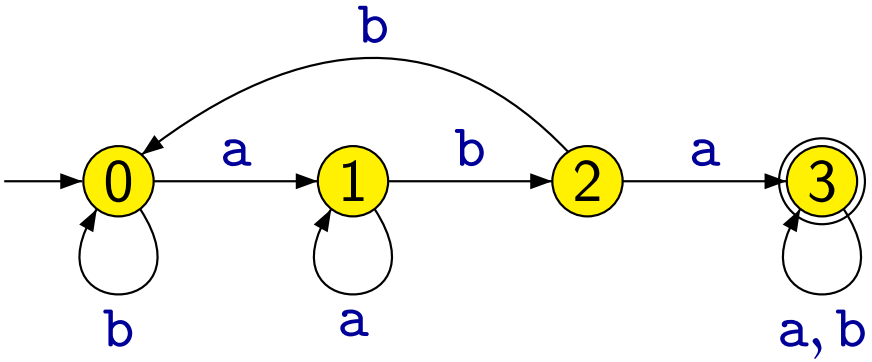
Automat pro průnik jazyků

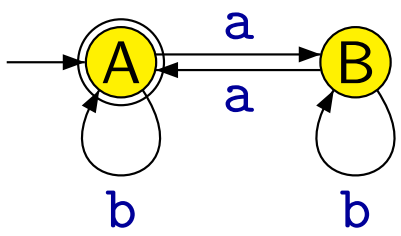


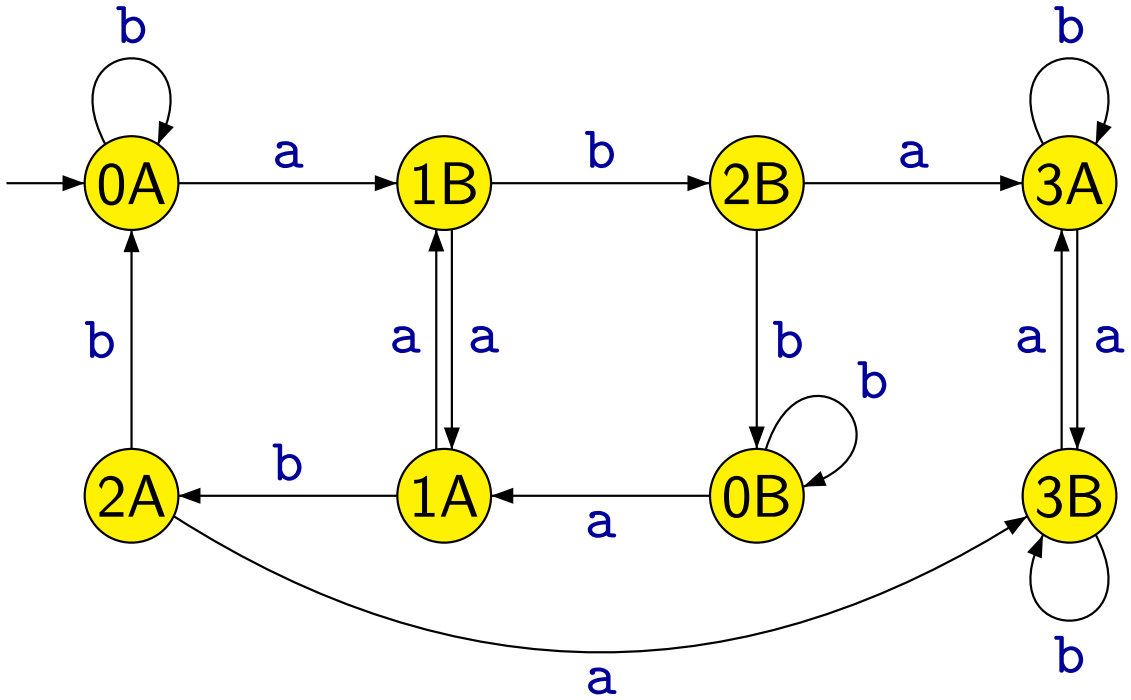




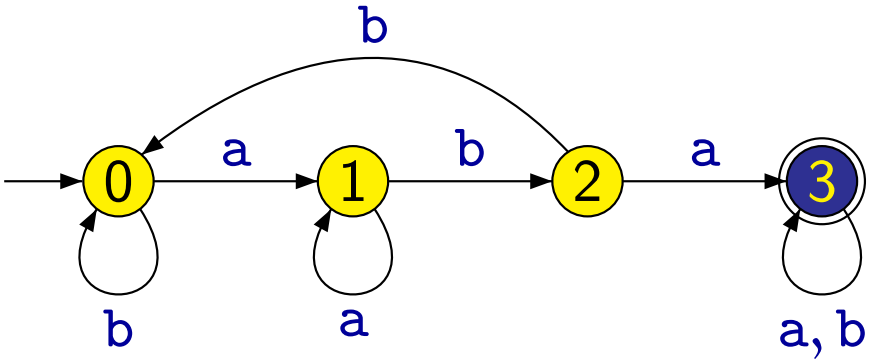
Automat pro průnik jazyků

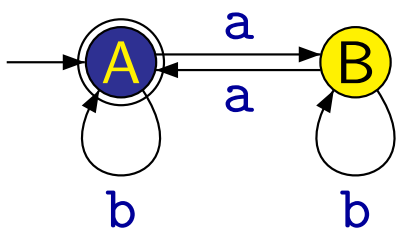


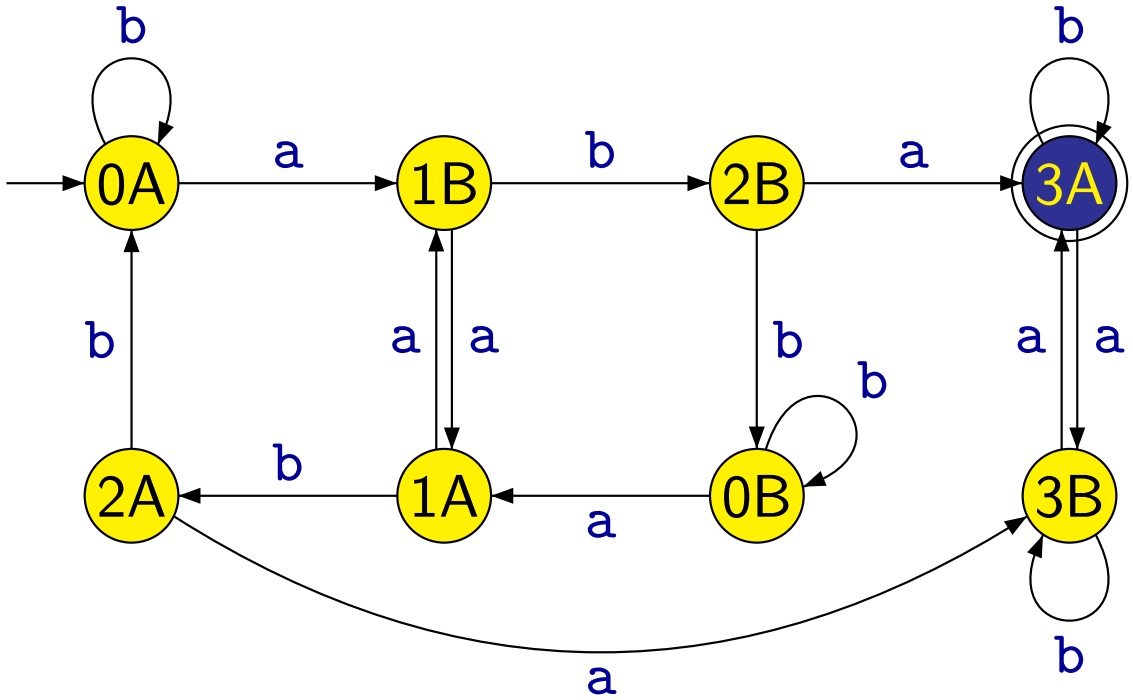




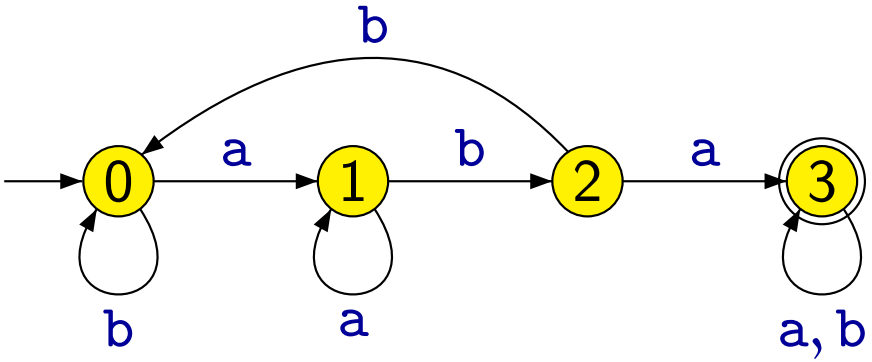
Automat pro průnik jazyků

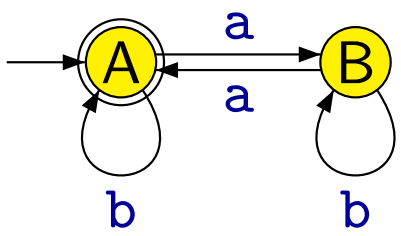


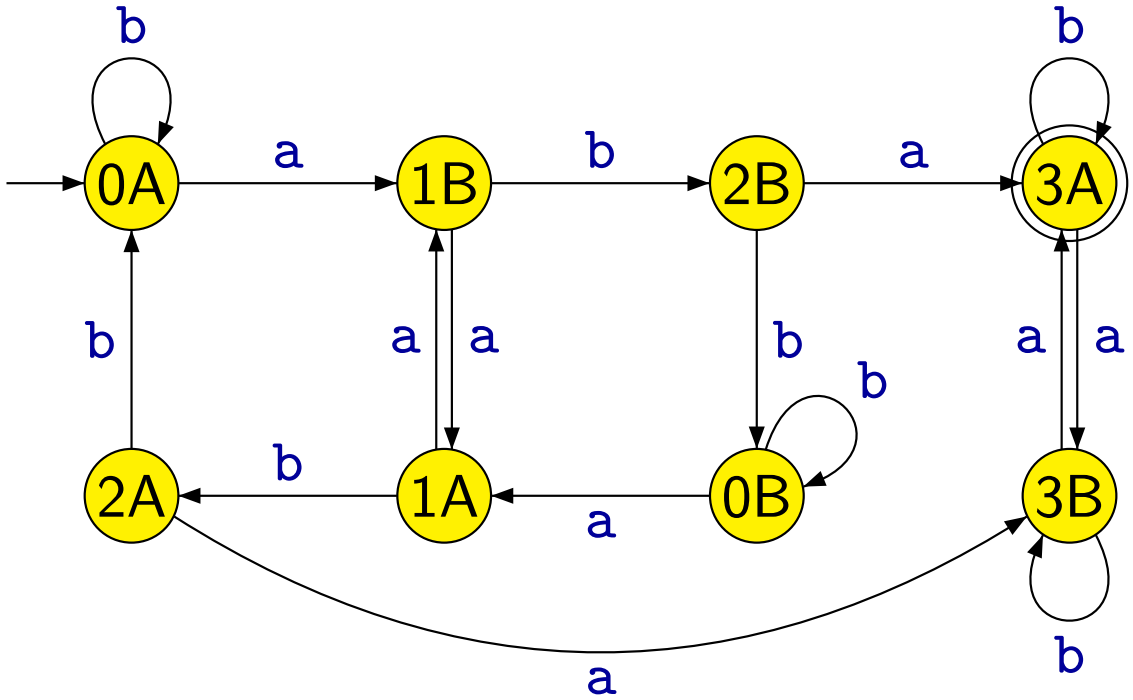




Automat pro průnik jazyků







Automat pro průnik jazyků

Formálně můžeme popsat tuto konstrukci následovně:

Předpokládáme, že máme dva deterministické konečné automaty $A_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$ a $A_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$.

K nim setrojíme DKA $A = (Q, \Sigma, \delta, q_0, F)$ 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 $w \in L(A)$ právě tehdy, když $w \in L(A_1)$ a $w \in L(A_2)$, tj.

$$L(A) = L(A_1) \cap L(A_2)$$

Průnik regulárních jazyků

Věta

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

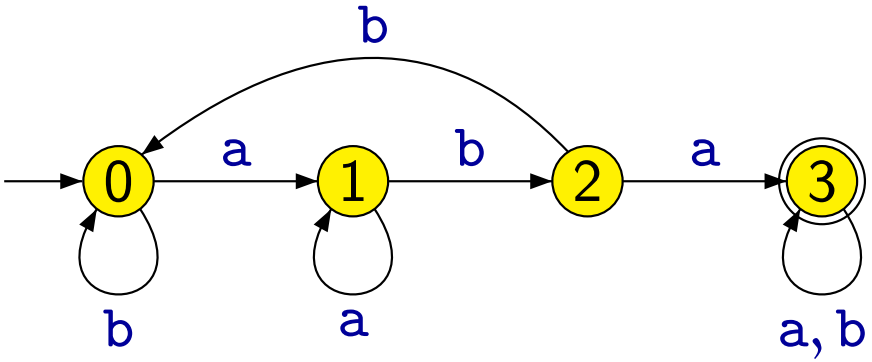
Důkaz: Předpokládejme, že A_1 a A_2 jsou deterministické konečné automaty takové, že

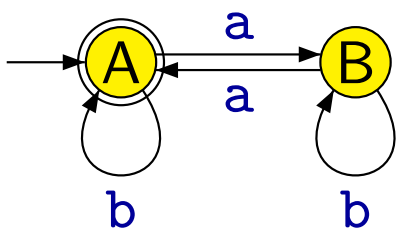
$$L_1 = L(A_1) \qquad L_2 = L(A_2)$$

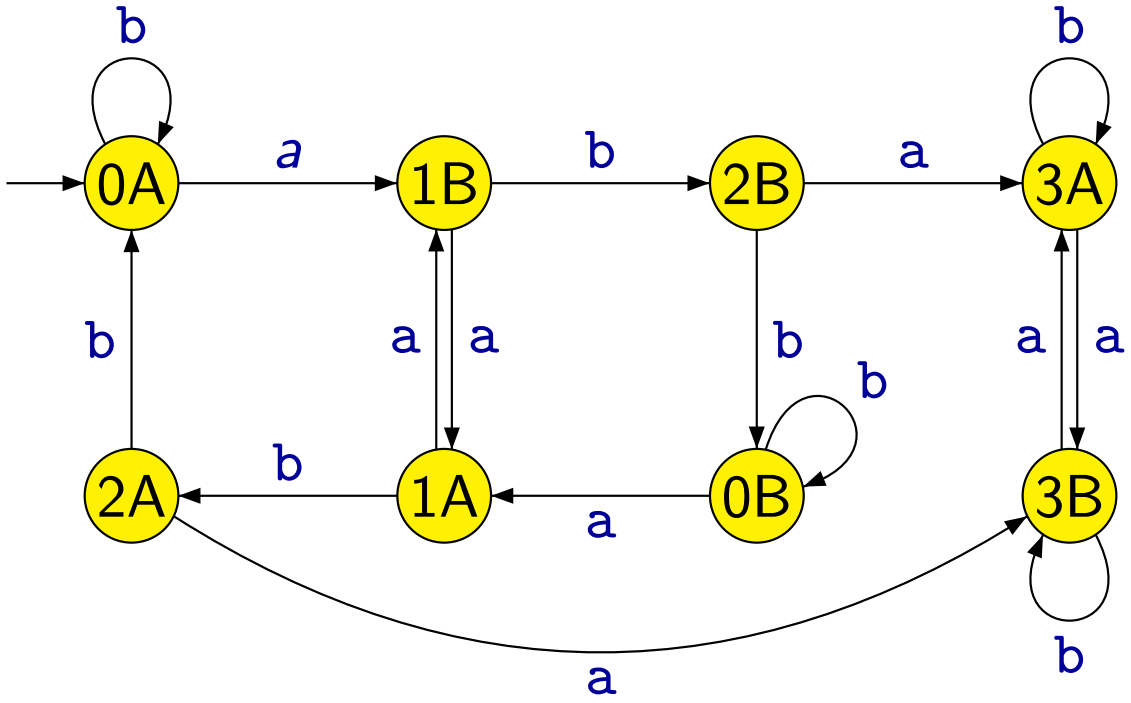
Popsanou konstrukcí k nim můžeme sestrojit deterministický konečný automat A takový, že

$$L(A) = L(A_1) \cap L(A_2) = L_1 \cap L_2$$

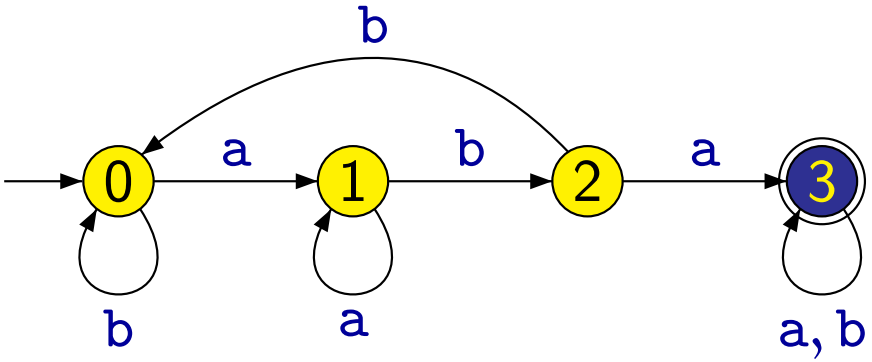
Automat pro sjednocení jazyků

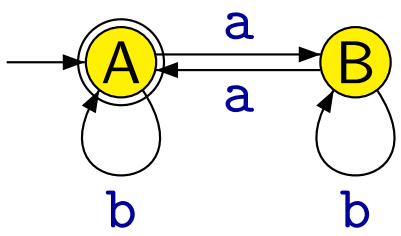


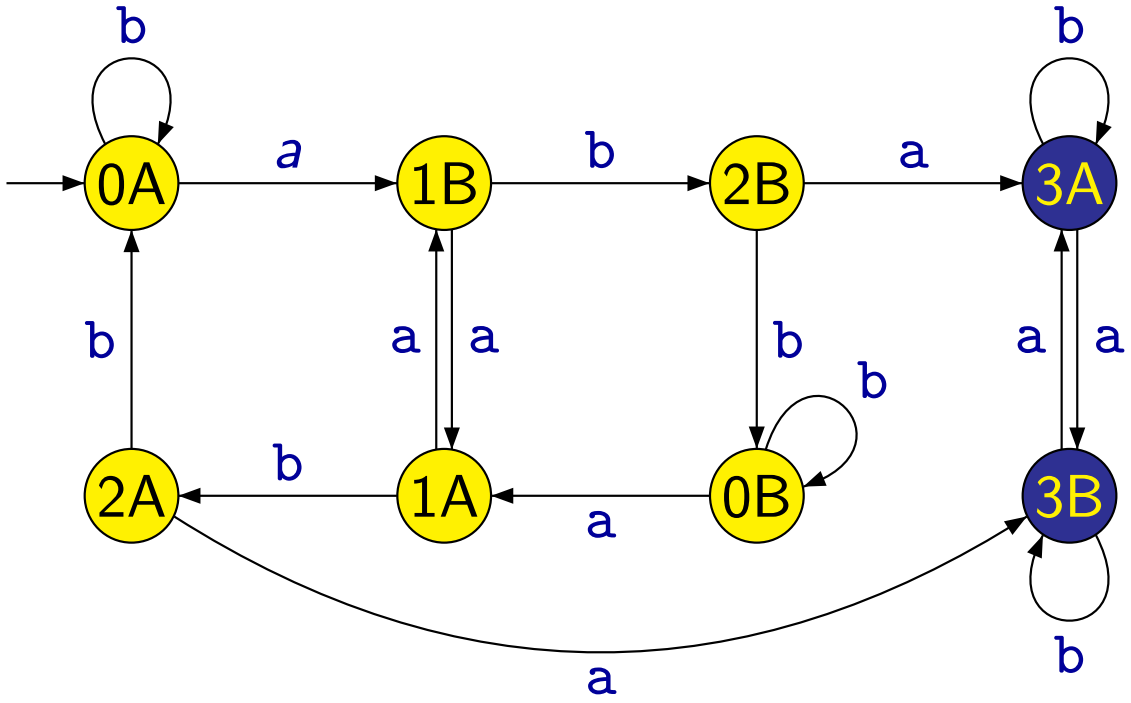




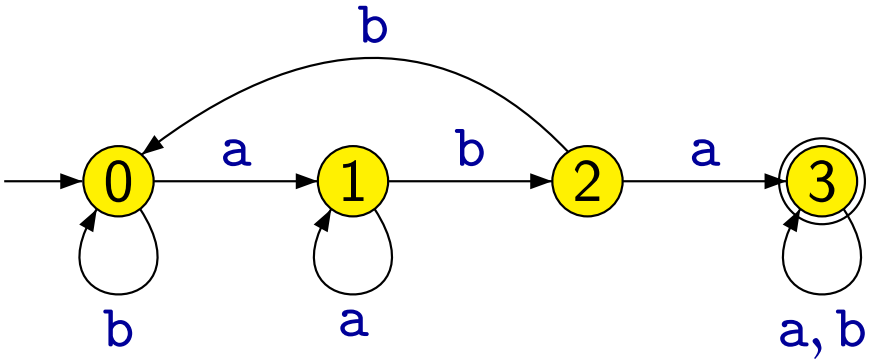
Automat pro sjednocení jazyků

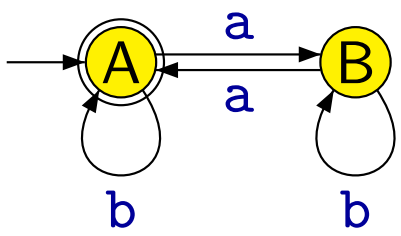


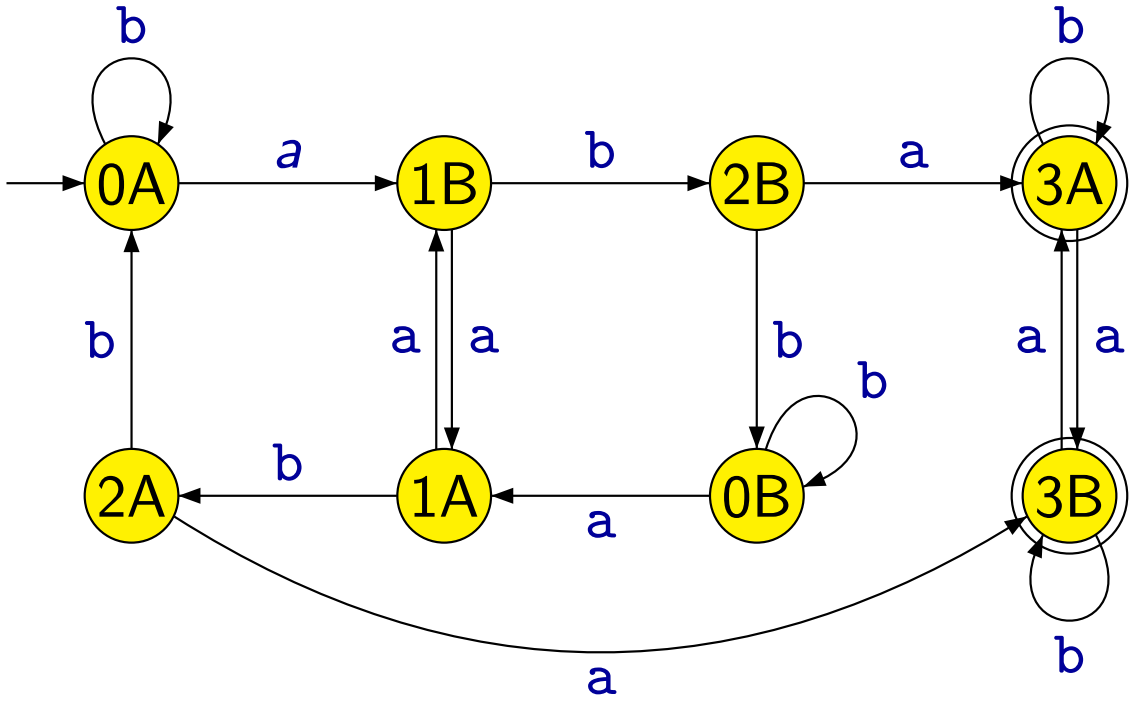




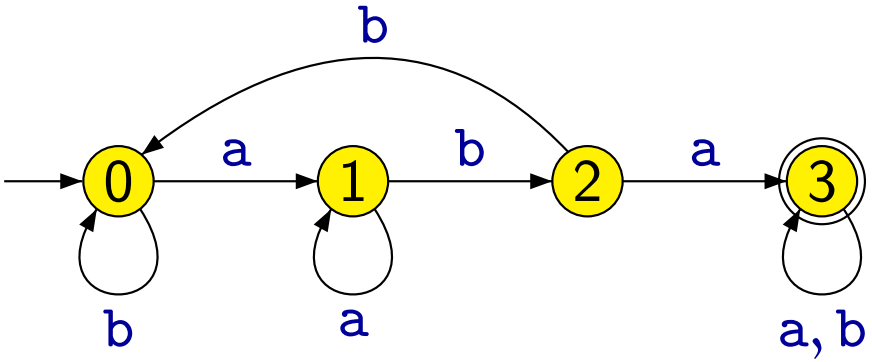
Automat pro sjednocení jazyků

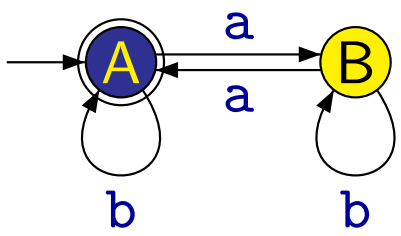


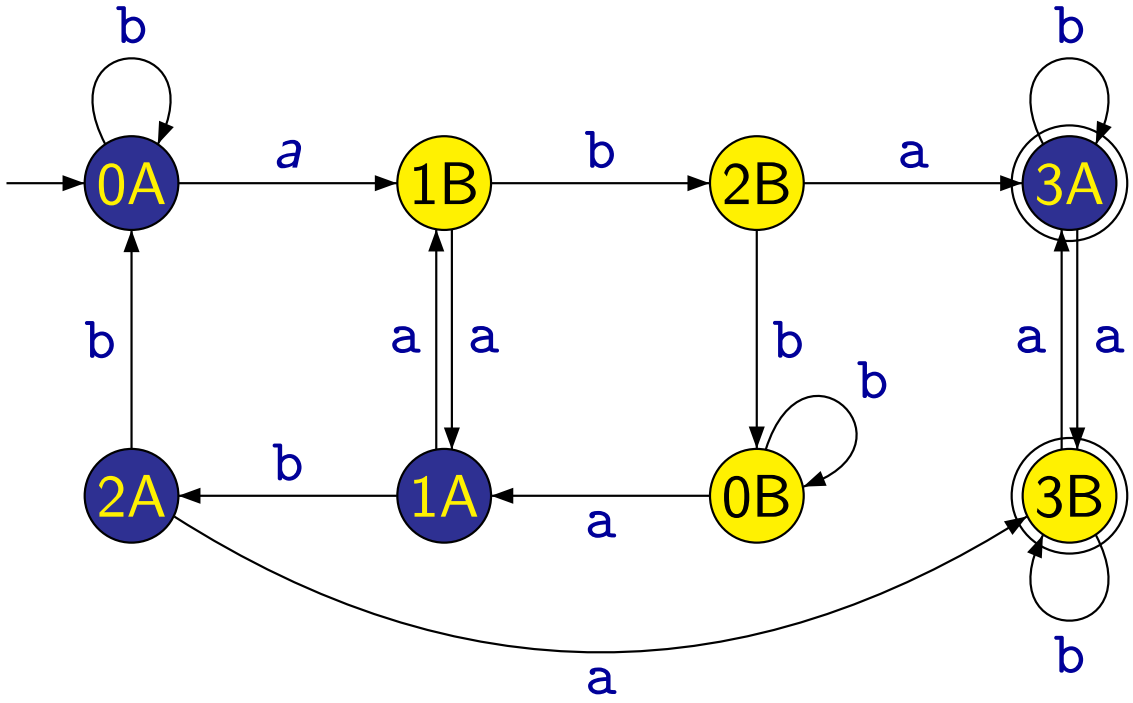




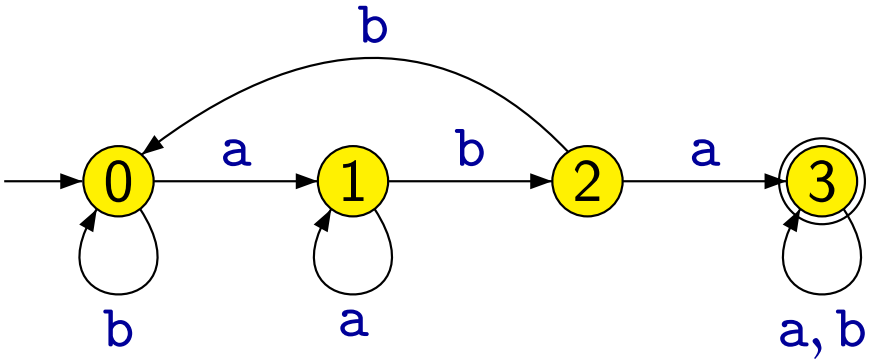
Automat pro sjednocení jazyků

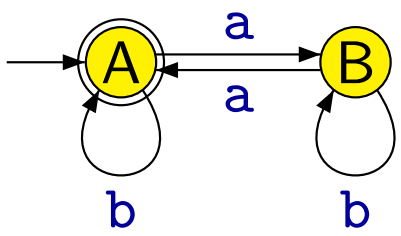


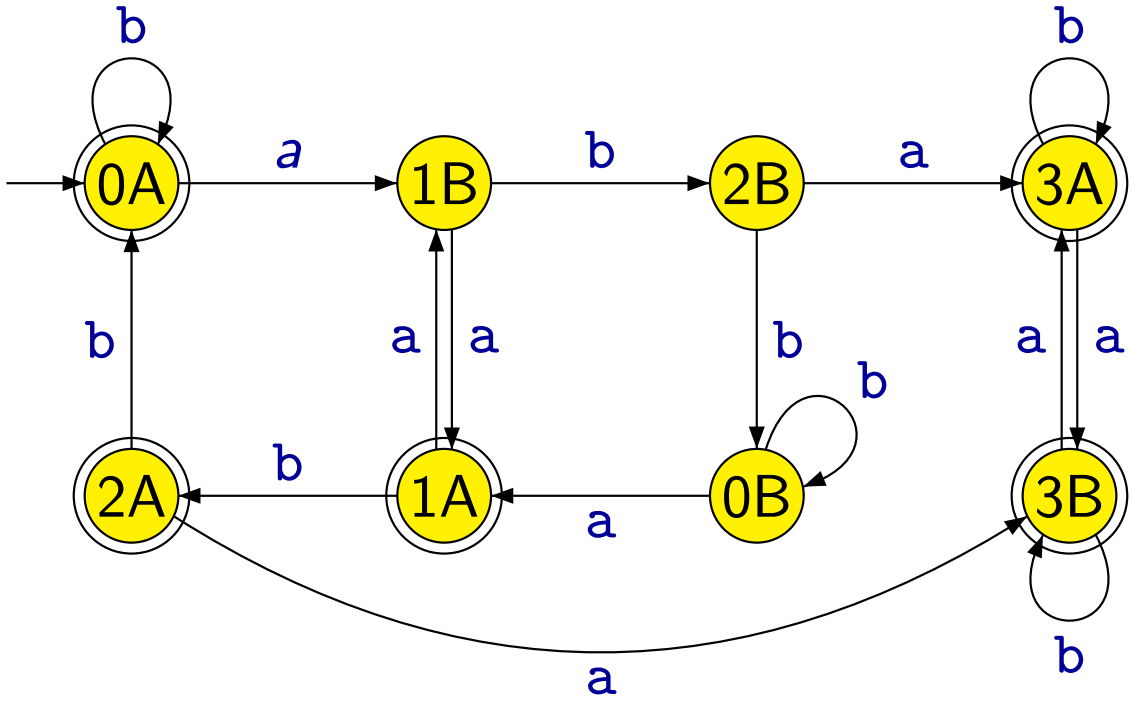




Automat pro sjednocení jazyků







Sjednocení regulárních jazyků

Konstukce automatu A, který přijímá **sjednocení** jazyků přijímaných automaty A_1 a A_2, tj. jazyk

$$L(A_1) \cup L(A_1)$$

je téměř stejná jako v případě automatu přijímajícího $L(A_1) \cap L(A_2)$.

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 A, který přijímá **sjednocení** jazyků přijímaných automaty A_1 a A_2, tj. jazyk

$$L(A_1) \cup L(A_1)$$

je téměř stejná jako v případě automatu přijímajícího $L(A_1) \cap L(A_2)$.

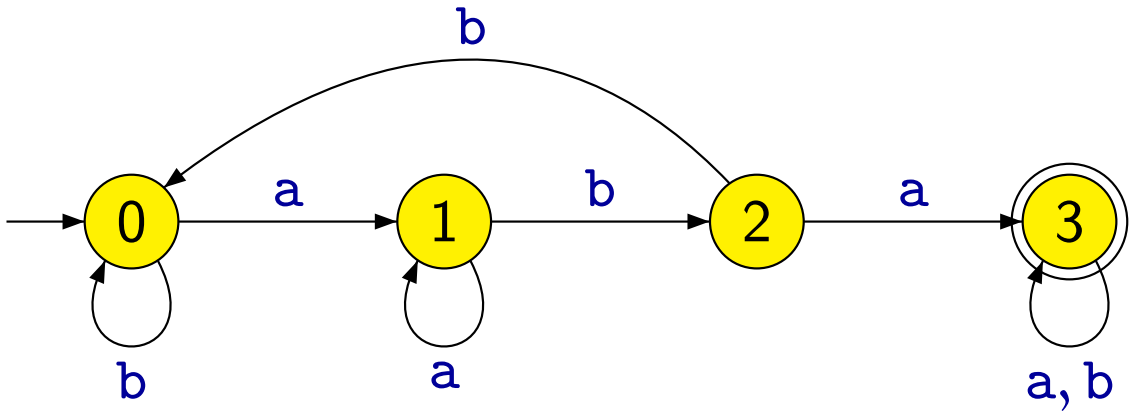
Jediný rozdíl je v definici množiny přijímajících stavů:

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

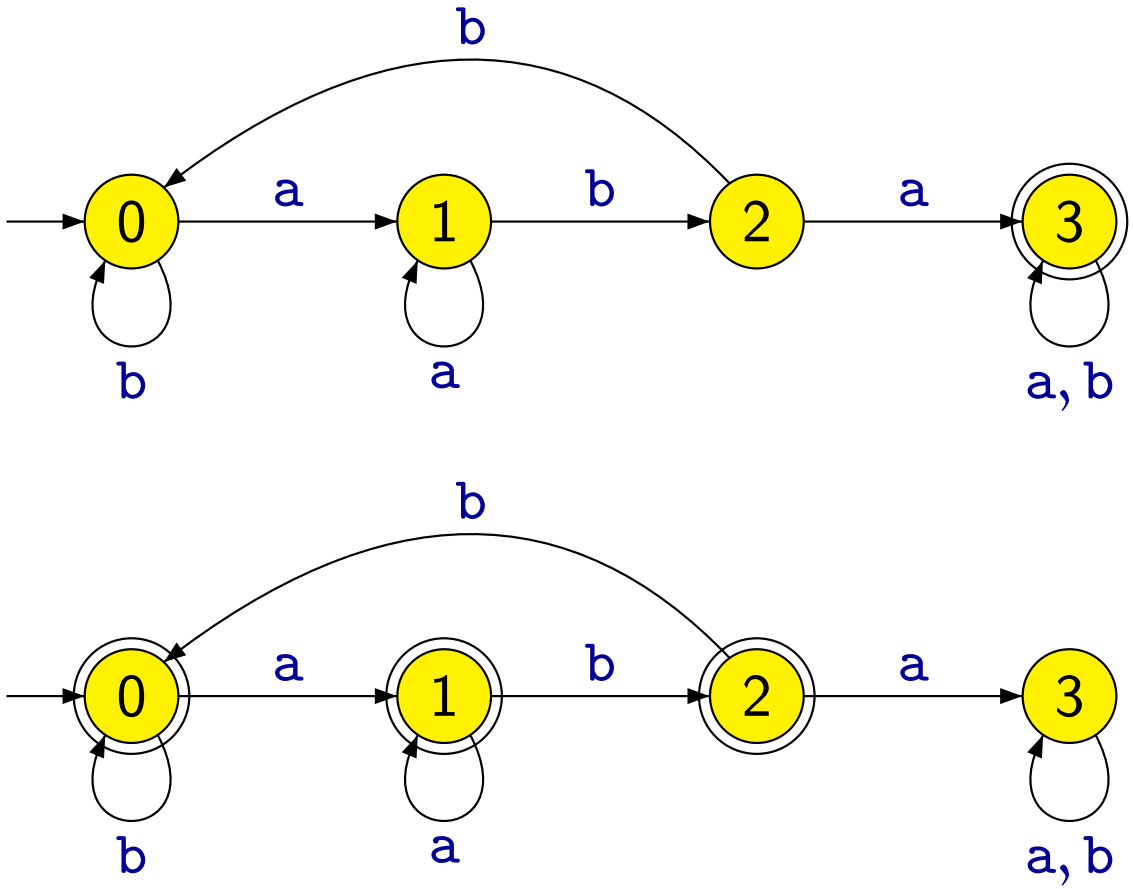
Věta

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

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

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

Doplněk regulárního jazyka

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

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

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

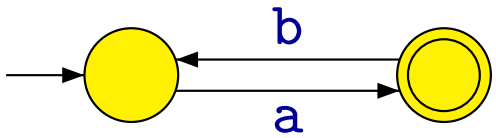
Věta

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

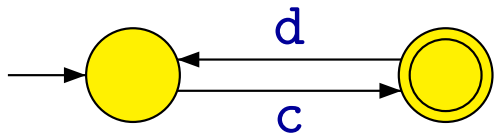
Zřetězení jazyků

$\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$

A_1:



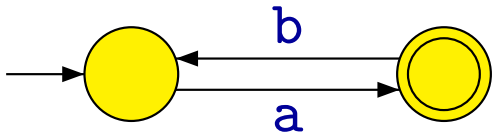
A_2:



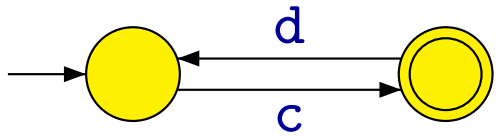
Zřetězení jazyků

$\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$

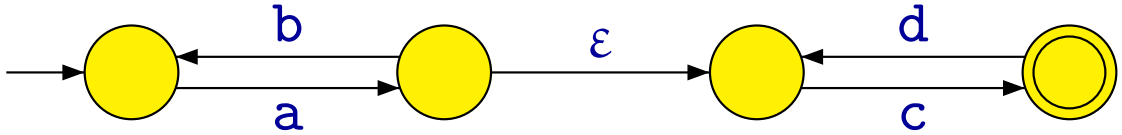
A_1:



A_2:



A:

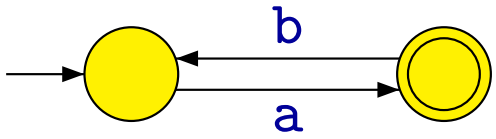


$L(A) = L(A_1) \cdot L(A_2)$

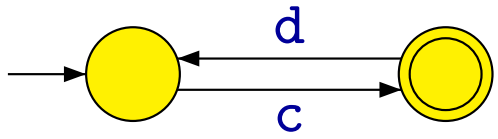
Zřetězení jazyků

$\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$

A_1:

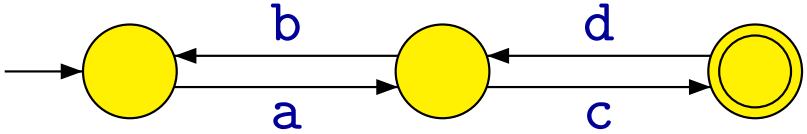


A_2:



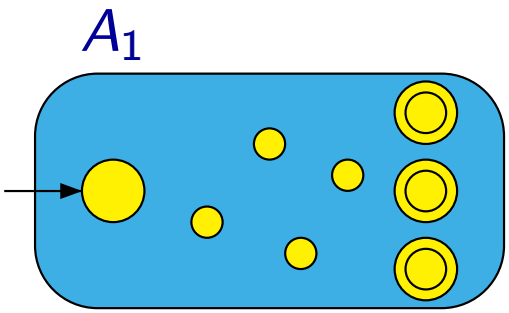
Chybná konstrukce:

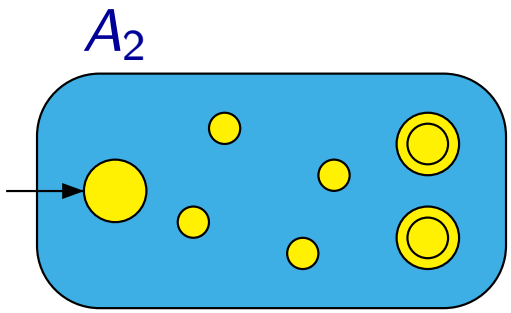
A:



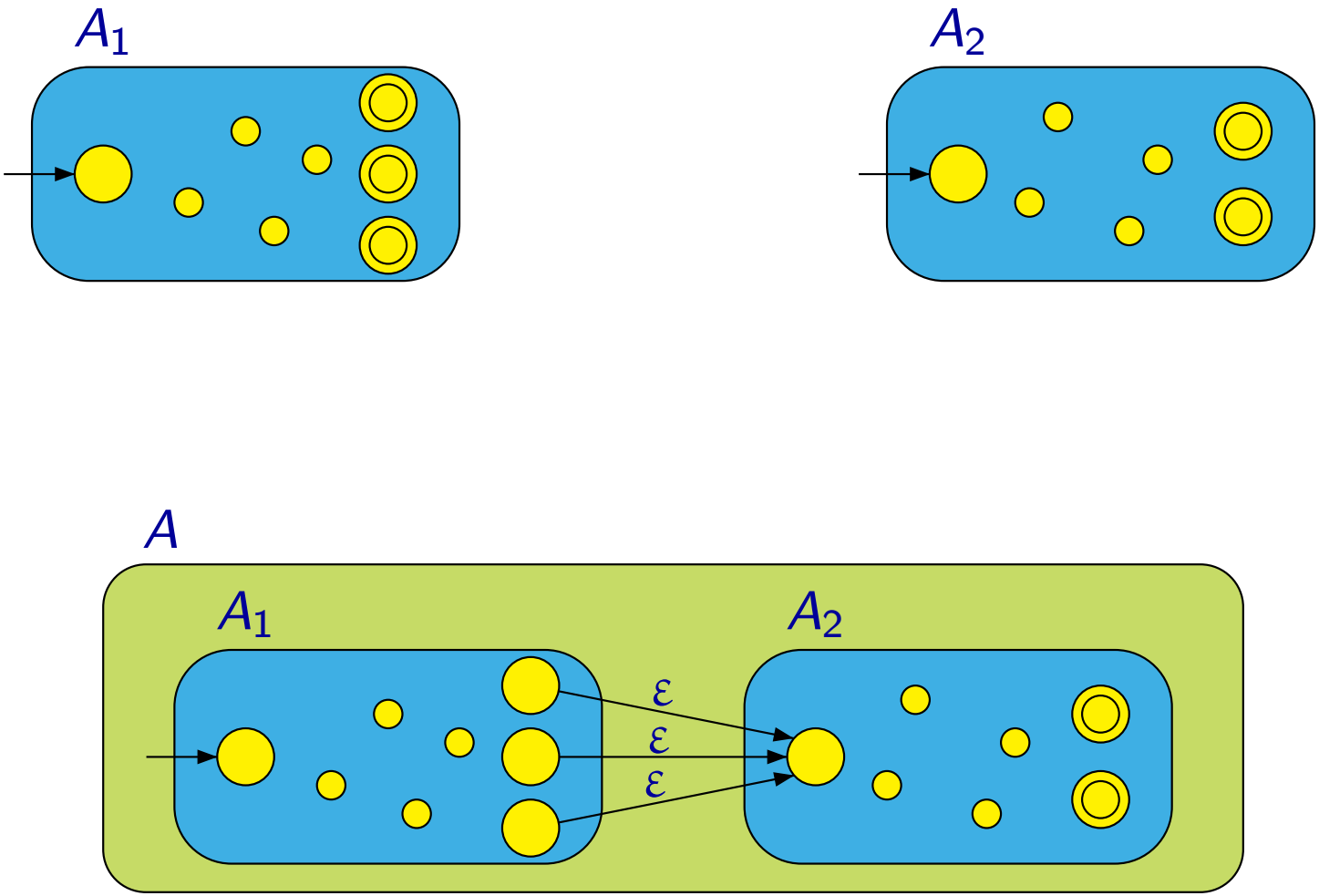
$\mathtt{acdbac} \in L(A)$, ale $\mathtt{acdbac} \notin L(A_1) \cdot L(A_2)$

Zřetězení jazyků

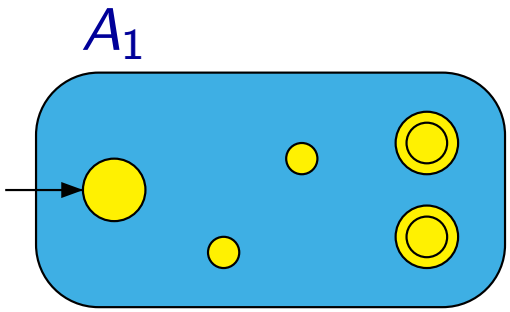




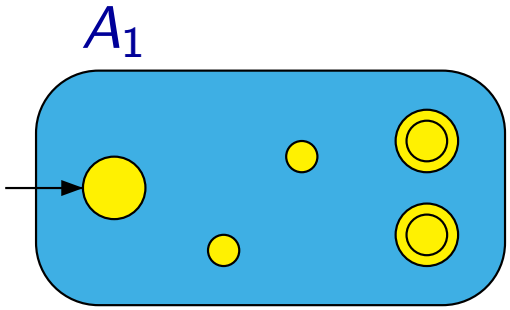
Zřetězení jazyků

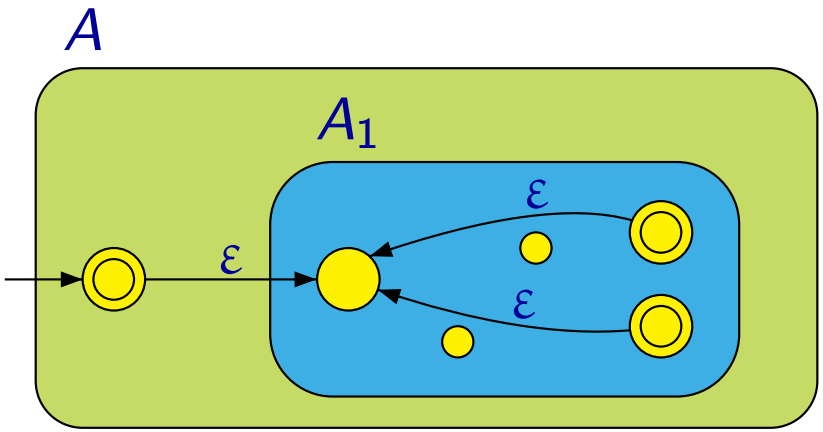


Iterace jazyka



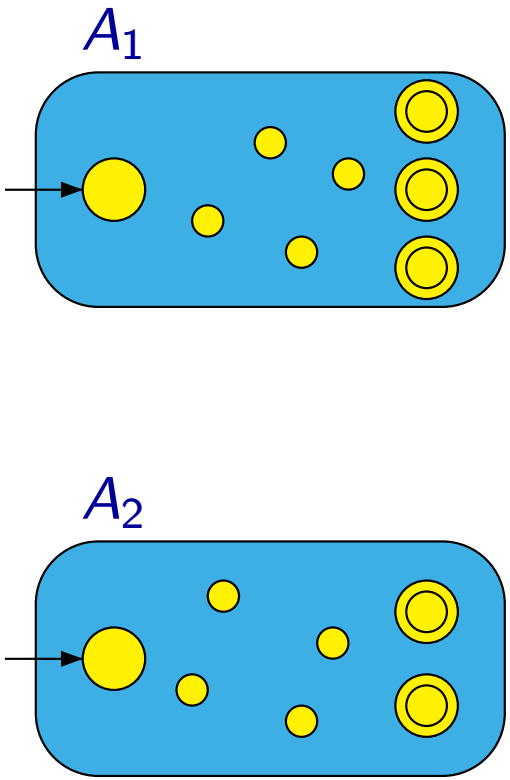
Iterace jazyka





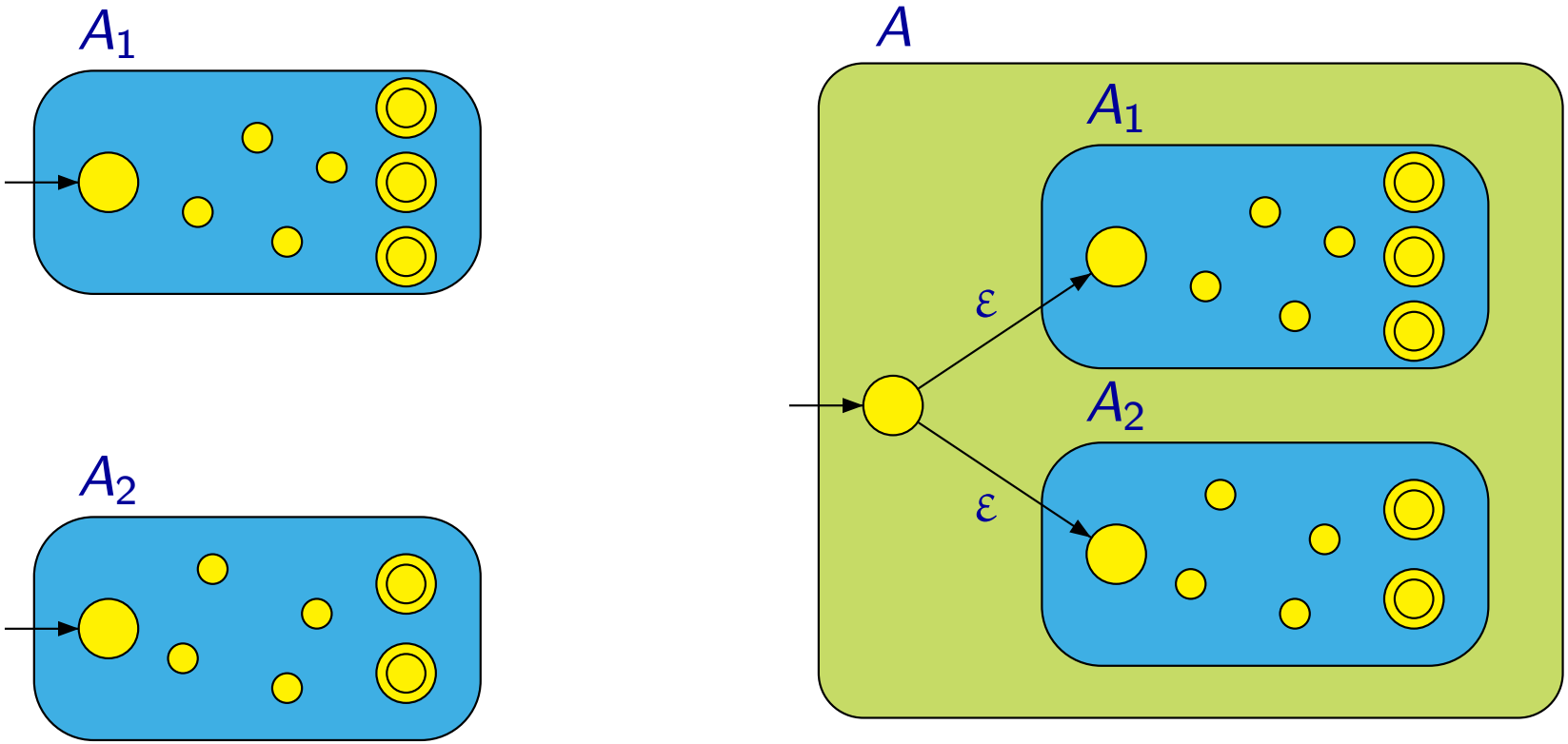
Sjednocení jazyků

Alternativní konstrukce pro sjednocení jazyků:



Sjednocení jazyků

Alternativní konstrukce pro sjednocení jazyků:



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 0 i 1 regulární výrazy.
- Protože 0 i 1 jsou regulární výrazy, je i $(0 + 1)$ 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 $\Sigma = \{0, 1\}$

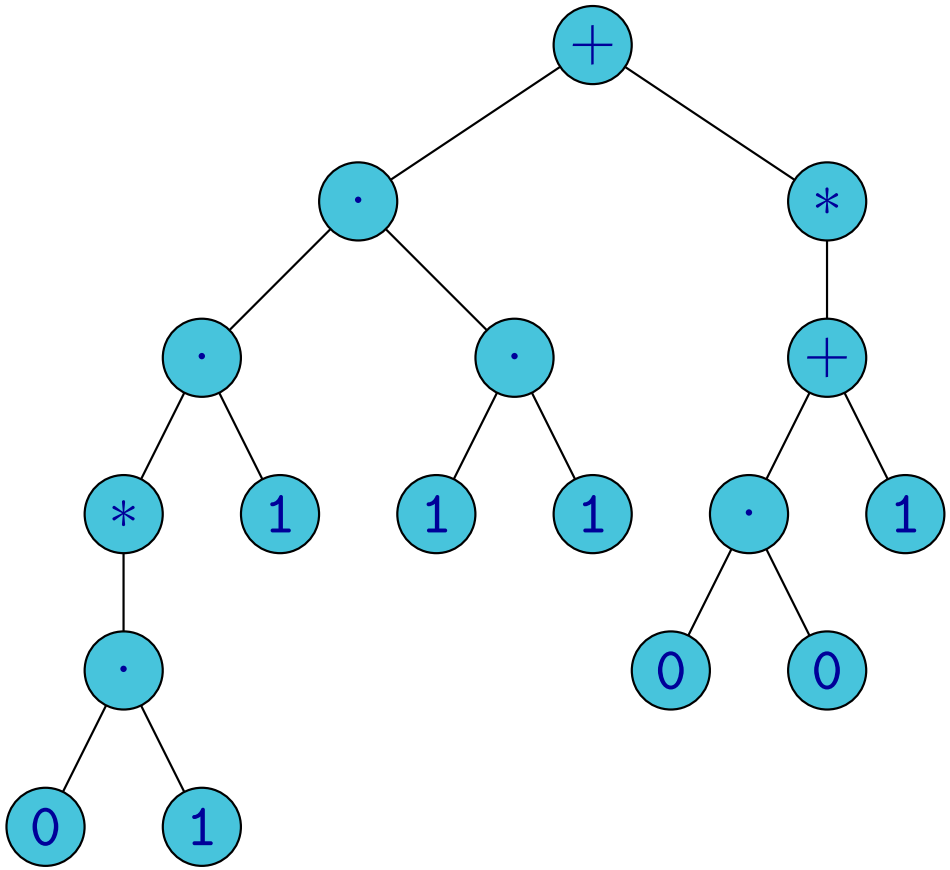
- Podle definice jsou 0 i 1 regulární výrazy.
- Protože 0 i 1 jsou regulární výrazy, je i $(0 + 1)$ regulární výraz.
- Protože 0 je regulární výraz, je i (0^*) regulární výraz.
- Protože $(0 + 1)$ i (0^*) jsou regulární výrazy, je i $((0 + 1) \cdot (0^*))$ regulární výraz.

Poznámka: Jestliže α je regulární výraz, zápisem $[\alpha]$ označujeme jazyk definovaný regulárním výrazem α.

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

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:

- $[\emptyset] = \emptyset$
- $[\varepsilon] = \{\varepsilon\}$
- $[a] = \{a\}$
- $[\alpha^*] = [\alpha]^*$
- $[\alpha \cdot \beta] = [\alpha] \cdot [\beta]$
- $[\alpha + \beta] = [\alpha] \cup [\beta]$

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 $\Sigma = \{0, 1\}$.

0 ... jazyk tvořený jediným slovem 0

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{0, 1\}$.

0 . . . jazyk tvořený jediným slovem 0

01 . . . jazyk tvořený jediným slovem 01

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{0, 1\}$.

0 ... jazyk tvořený jediným slovem 0

01 ... jazyk tvořený jediným slovem 01

$0 + 1$ ... jazyk tvořený dvěma slovy 0 a 1

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{0, 1\}$.

0 ... jazyk tvořený jediným slovem 0

01 ... jazyk tvořený jediným slovem 01

$0 + 1$ ... jazyk tvořený dvěma slovy 0 a 1

0^* ... jazyk tvořený slovy ε, 0, 00, 000, ...

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{0, 1\}$.

0 ... jazyk tvořený jediným slovem 0

01 ... jazyk tvořený jediným slovem 01

$0 + 1$ ... jazyk tvořený dvěma slovy 0 a 1

0^* ... jazyk tvořený slovy ε, 0, 00, 000, ...

$(01)^*$ ... jazyk tvořený slovy ε, 01, 0101, 010101, ...

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{0, 1\}$.

0 ... jazyk tvořený jediným slovem 0

01 ... jazyk tvořený jediným slovem 01

$0 + 1$ ... jazyk tvořený dvěma slovy 0 a 1

0^* ... jazyk tvořený slovy ε, 0, 00, 000, ...

$(01)^*$ ... jazyk tvořený slovy ε, 01, 0101, 010101, ...

$(0 + 1)^*$ ... jazyk tvořený všemi slovy nad abecedou $\{0, 1\}$

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{0, 1\}$.

0 ... jazyk tvořený jediným slovem 0

01 ... jazyk tvořený jediným slovem 01

$0 + 1$ ... jazyk tvořený dvěma slovy 0 a 1

0^* ... jazyk tvořený slovy ε, 0, 00, 000, ...

$(01)^*$ ... jazyk tvořený slovy ε, 01, 0101, 010101, ...

$(0 + 1)^*$ ... jazyk tvořený všemi slovy nad abecedou $\{0, 1\}$

$(0 + 1)^*00$ ... jazyk tvořený všemi slovy končícími 00

Regulární výrazy

Příklady: Ve všech případech $\Sigma = \{0, 1\}$.

0 ... jazyk tvořený jediným slovem 0

01 ... jazyk tvořený jediným slovem 01

$0 + 1$ ... jazyk tvořený dvěma slovy 0 a 1

0^* ... jazyk tvořený slovy ε, 0, 00, 000, ...

$(01)^*$ ... jazyk tvořený slovy ε, 01, 0101, 010101, ...

$(0 + 1)^*$ ... jazyk tvořený všemi slovy nad abecedou $\{0, 1\}$

$(0 + 1)^*00$ ... jazyk tvořený všemi slovy končícími 00

$(01)^*111(01)^*$ ... jazyk tvořený všemi slovy obsahujícími podslovo 111 předcházené i následované libovolným počtem slov 01

Regulární výrazy

$(0+1)^*00 + (01)^*111(01)^*$ ... jazyk tvořený všemi slovy, která buď končí 00 nebo obsahují podslovo 111 předcházené i následované libovolným počtem slov 01

Regulární výrazy

$(0+1)^*00+(01)^*111(01)^*$ ... jazyk tvořený všemi slovy, která buď končí 00 nebo obsahují podslovo 111 předcházené i následované libovolným počtem slov 01

$(0+1)^*1(0+1)^*$ ... jazyk tvořený všemi slovy obsahujícími alespoň jeden symbol 1

Regulární výrazy

$(0+1)^*00+(01)^*111(01)^*$ ... jazyk tvořený všemi slovy, která buď končí 00 nebo obsahují podslovo 111 předcházené i následované libovolným počtem slov 01

$(0+1)^*1(0+1)^*$ ... jazyk tvořený všemi slovy obsahujícími alespoň jeden symbol 1

$0^*(10^*10^*)^*$ ... jazyk tvořený všemi slovy obsahujícími sudý počet symbolů 1

Převod regulárního výrazu na konečný automat

Tvrzení

Každý jazyk, který je možné vyjádřit regulárním výrazem, je regulární (tj. rozpoznávaný nějakým konečným automatem).

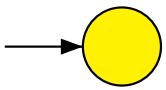
Důkaz: Stačí ukázat, jak k danému regulárnímu výrazu α zkonstruovat konečný automat, který rozpoznává jazyk $[\alpha]$.

Konstrukce je rekurzivní a postupuje podle struktury výrazu α:

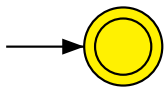
- Pokud je α elementární výraz (tj. $\emptyset$, ε nebo a):
  - Sestrojíme přímo odpovídající automat.
- Pokud je α tvaru $(\beta + \gamma)$, $(\beta \cdot \gamma)$ nebo (β^*):
  - Rekurzivně sestrojíme automaty rozpoznávající jazyky $[\beta]$ a $[\gamma]$.
  - Z nich sestrojíme automat rozpoznávající jazyk $[\alpha]$.

Převod regulárního výrazu na konečný automat

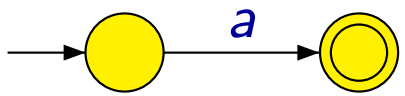
Automaty pro elementární výrazy:



$\emptyset$



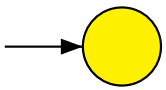
ε



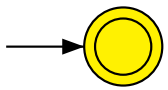
a

Převod regulárního výrazu na konečný automat

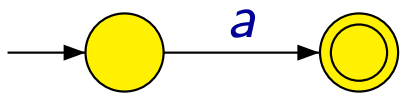
Automaty pro elementární výrazy:



$\emptyset$

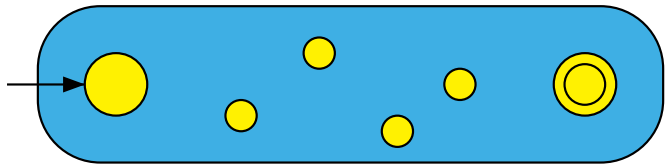


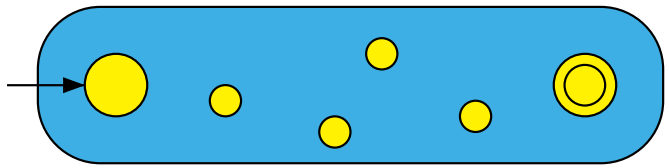
ε



a

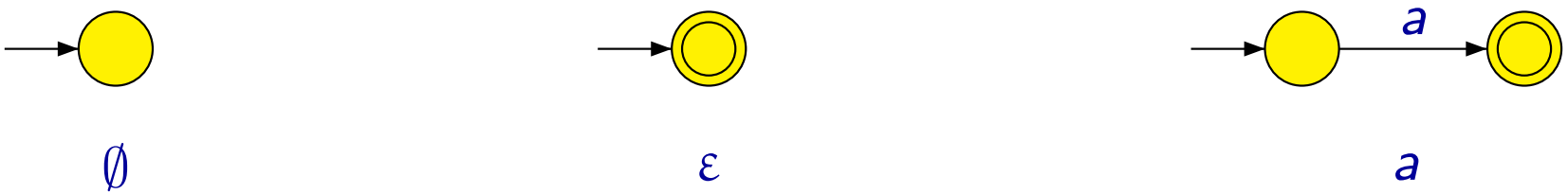
Konstrukce pro sjednocení:



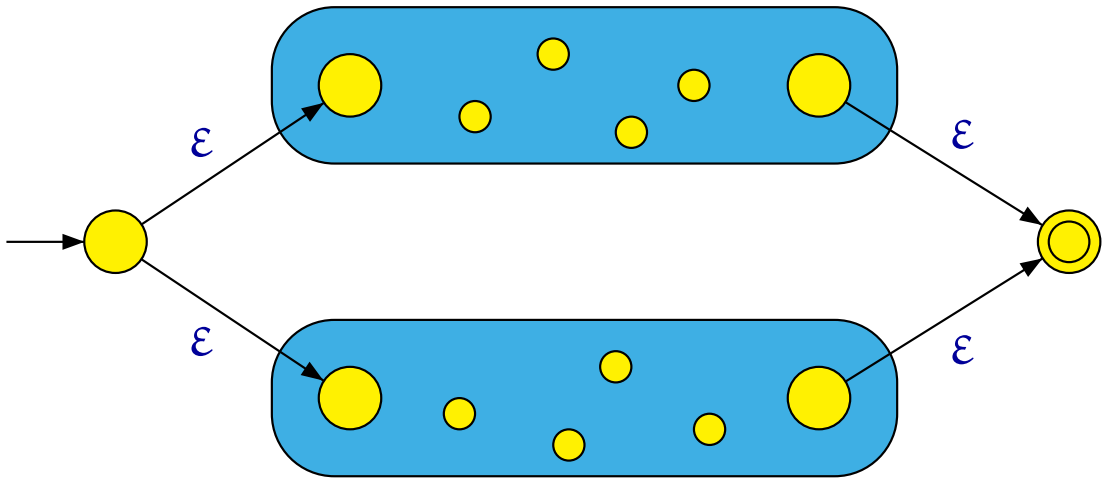


Převod regulárního výrazu na konečný automat

Automaty pro elementární výrazy:

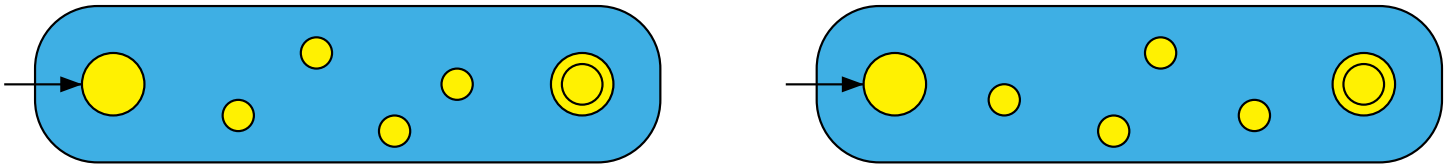


Konstrukce pro sjednocení:



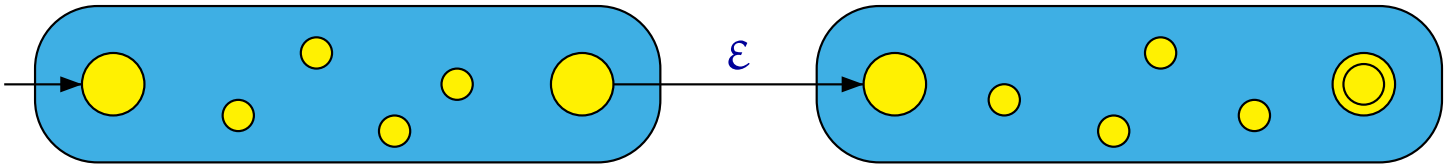
Převod regulárního výrazu na konečný automat

Konstrukce pro zřetězení:



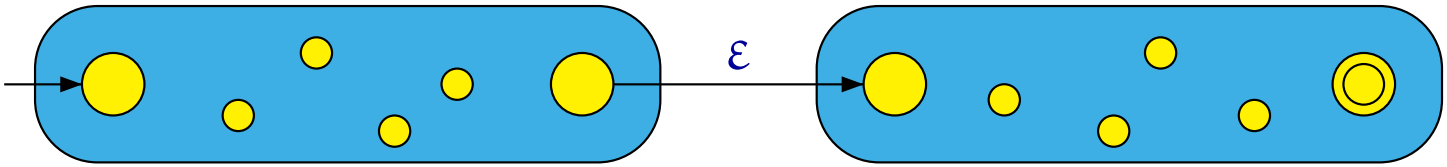
Převod regulárního výrazu na konečný automat

Konstrukce pro zřetězení:

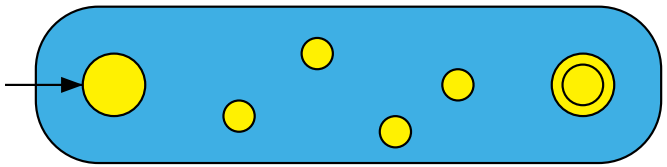


Převod regulárního výrazu na konečný automat

Konstrukce pro zřetězení:

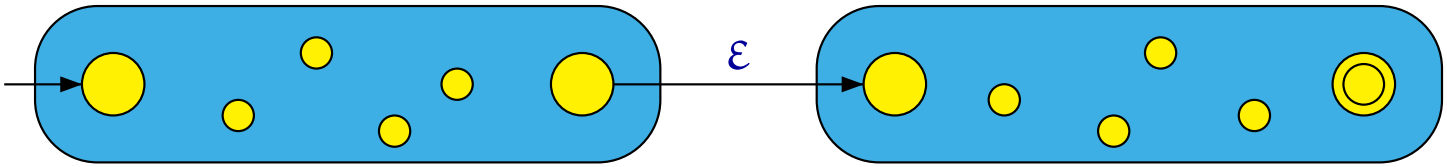


Konstrukce pro iteraci:

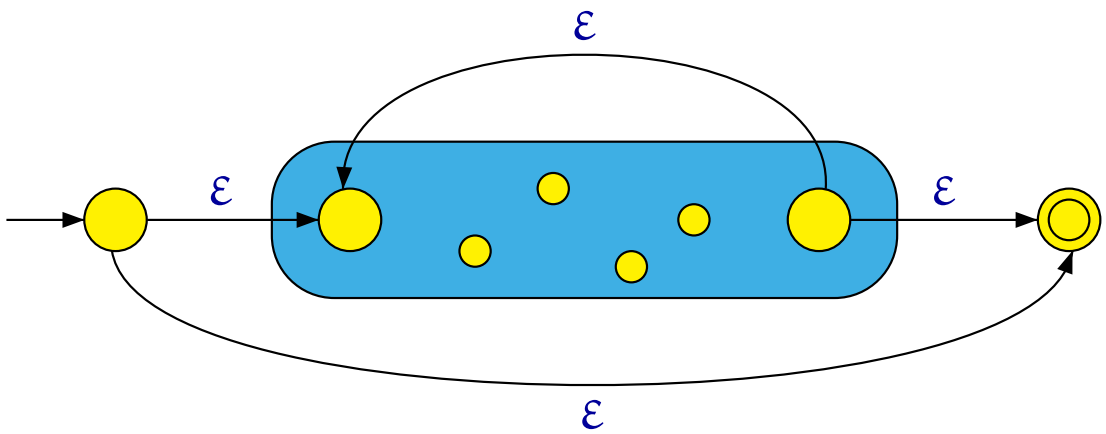


Převod regulárního výrazu na konečný automat

Konstrukce pro zřetězení:



Konstrukce pro iteraci:

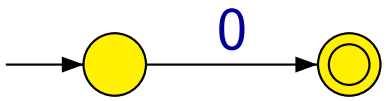


Převod regulárního výrazu na konečný automat

Příklad: Konstrukce automatu pro výraz $((0+1)\cdot 1)^*$:

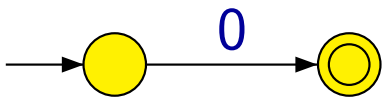
Převod regulárního výrazu na konečný automat

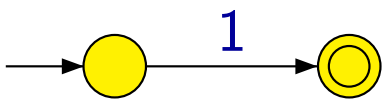
Příklad: Konstrukce automatu pro výraz $((0+1)\cdot 1)^*$:



Převod regulárního výrazu na konečný automat

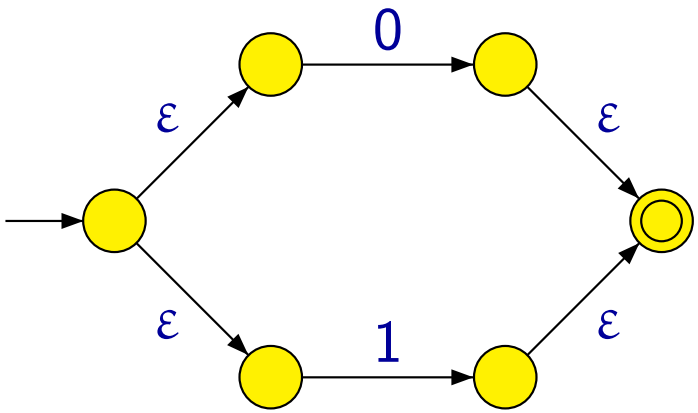
Příklad: Konstrukce automatu pro výraz $((0+1)\cdot 1)^*$:





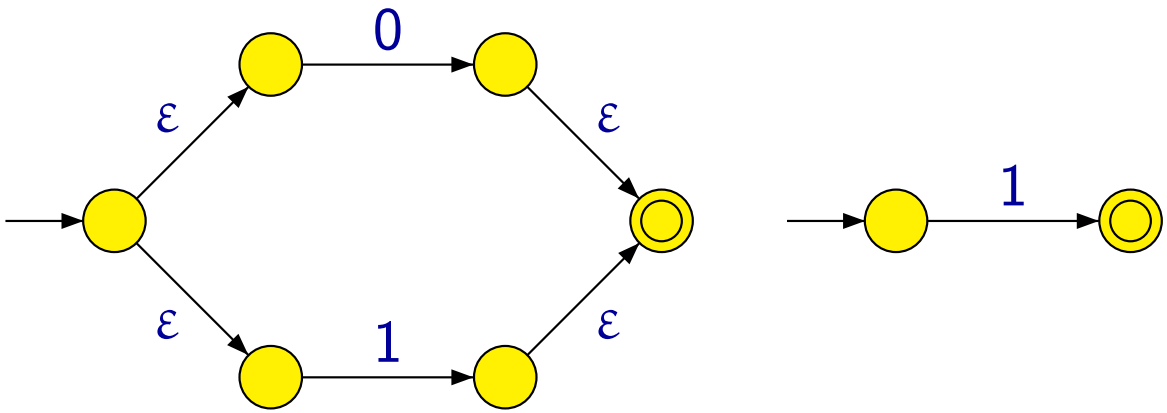
Převod regulárního výrazu na konečný automat

Příklad: Konstrukce automatu pro výraz $((0+1)\cdot 1)^*$:



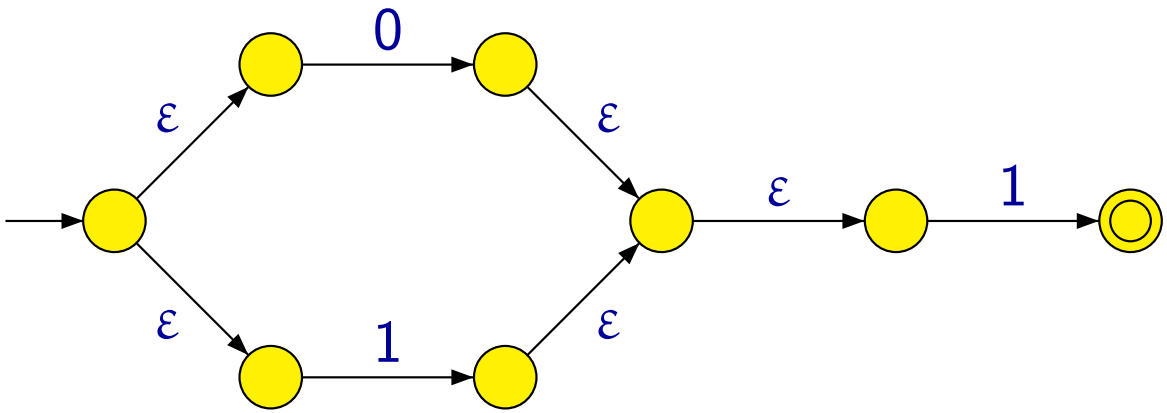
Převod regulárního výrazu na konečný automat

Příklad: Konstrukce automatu pro výraz $((0+1)\cdot 1)^*$:



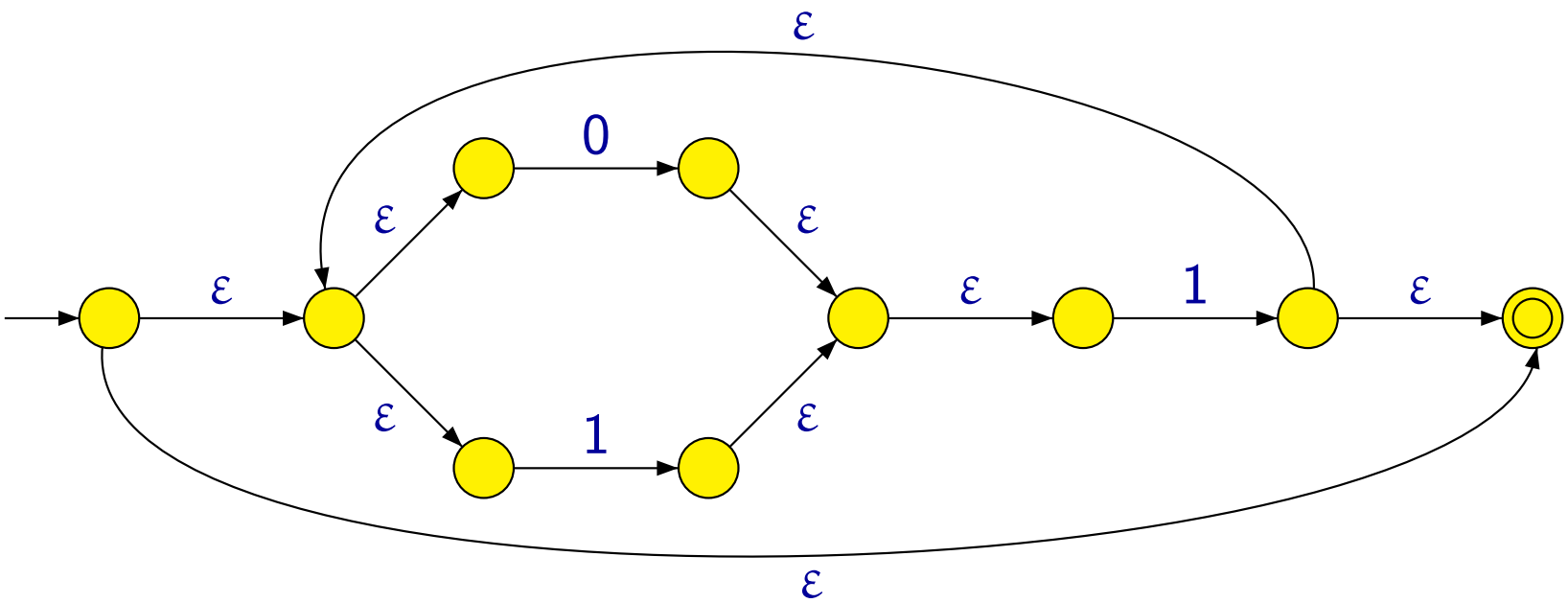
Převod regulárního výrazu na konečný automat

Příklad: Konstrukce automatu pro výraz $((0+1)\cdot 1)^*$:



Převod regulárního výrazu na konečný automat

Příklad: Konstrukce automatu pro výraz $((0+1)\cdot 1)^*$:



Převod regulárního výrazu na konečný automat

Pokud se výraz α skládá z n znaků (nepočítáme-li závorky), má výsledný automat:

- nejvýše $2n$ stavů,
- nejvýše $4n$ přechodů.

Poznámka: Převodem ze zobecněného nedeterministického automatu na deterministický však může počet stavů vzrůst exponenciálně, tj. výsledný automat pak může mít až $2^{2n} = 4^n$ stavů.

Převod konečného automatu na regulární výraz

Tvrzení

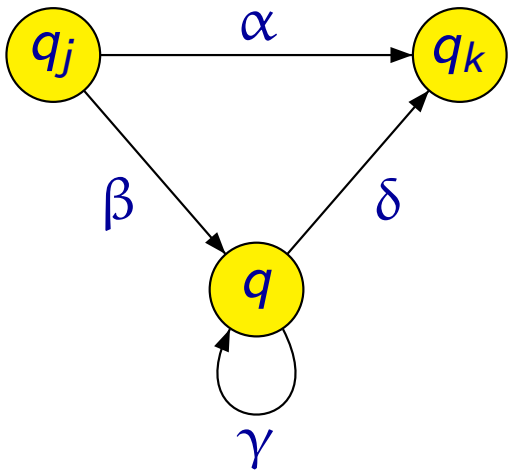
Každý regulární jazyk je možné popsat nějakým regulárním výrazem.

Důkaz: Stačí ukázat, jak pro libovolný konečný automat A zkonstruovat regulární výraz α takový, že $[\alpha] = L(A)$.

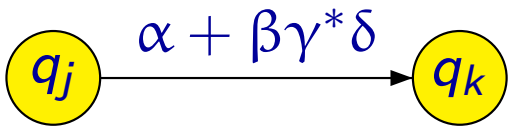
- A upravíme tak, aby měl právě jeden počáteční a právě jeden koncový stav.
- Budeme postupně odebírat jednotlivé stavy.
- Přechody budou označeny regulárními výrazy.
- Zbude automat se dvěma stavy – počátečním a koncovým, a jedním přechodem ohodnoceným výsledným regulárním výrazem.

Převod konečného automatu na regulární výraz

Hlavní myšlenka: Při odstraňování stavu q nahradit pro každou dvojici zbylých stavů q_j, q_k cestu z q_j do q_k vedoucí přes q.

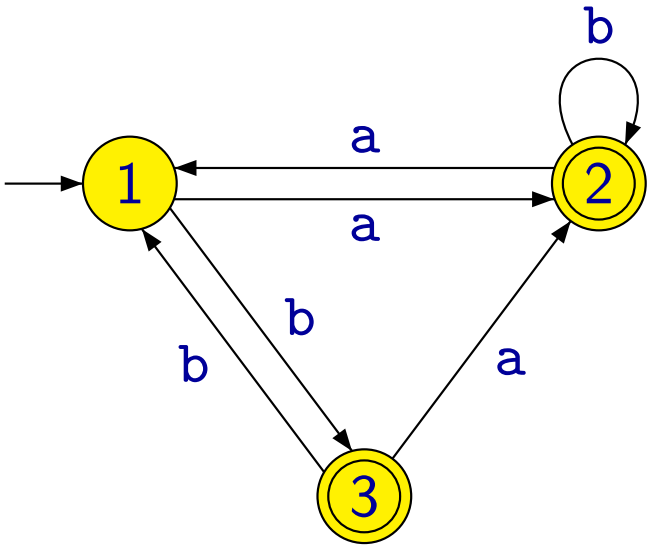


Po odstranění stavu q:



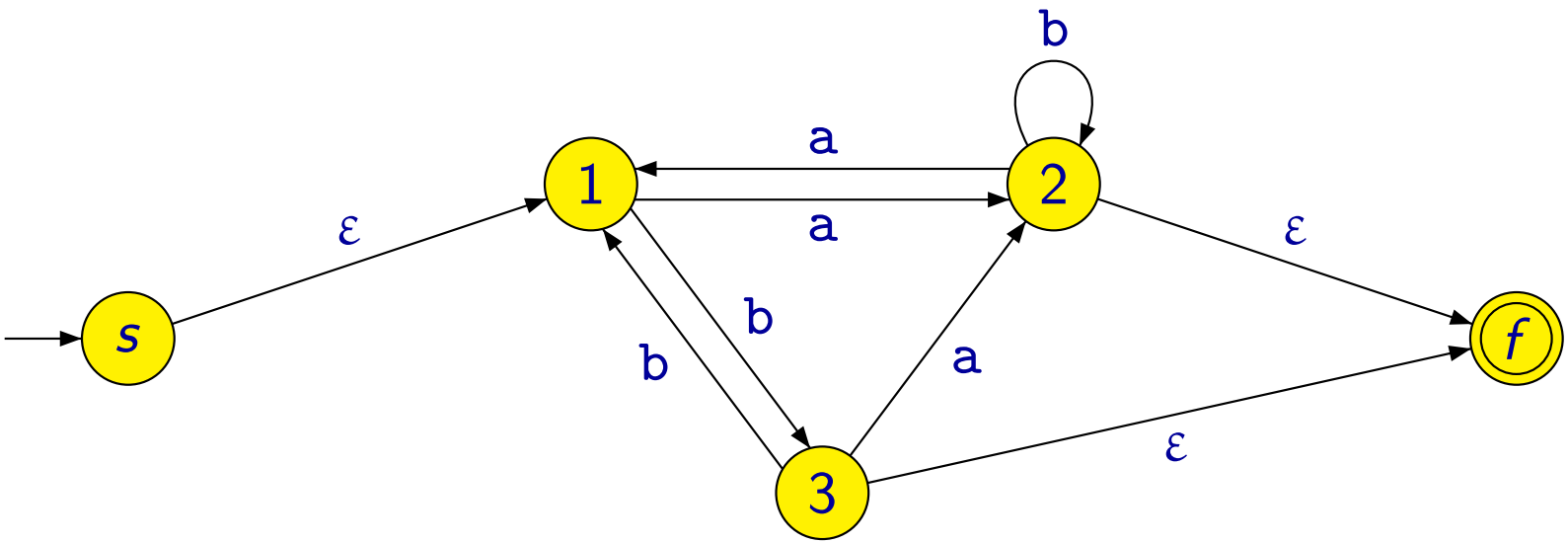
Převod konečného automatu na regulární výraz

Příklad:



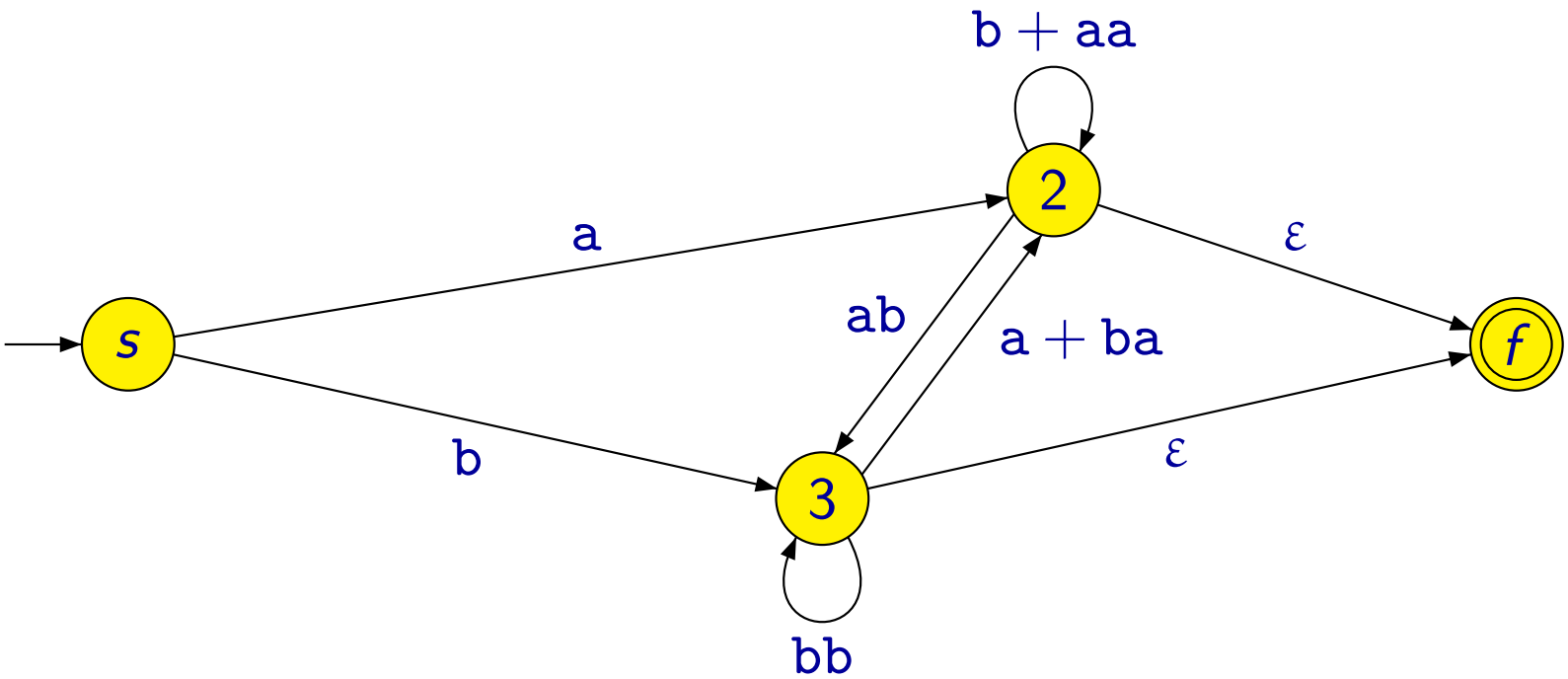
Převod konečného automatu na regulární výraz

Příklad:



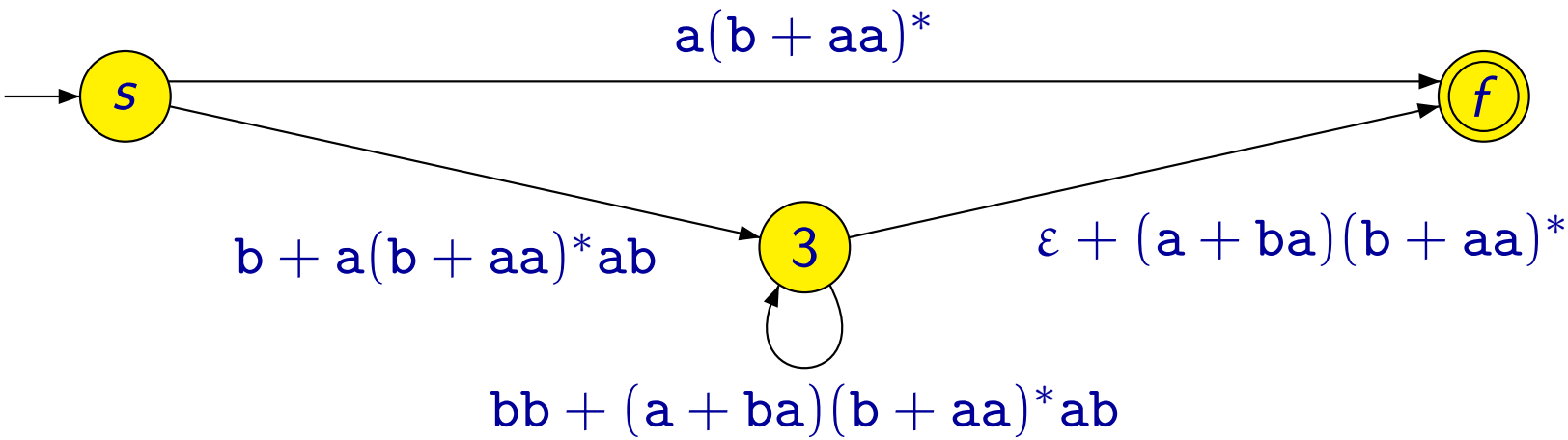
Převod konečného automatu na regulární výraz

Příklad:



Převod konečného automatu na regulární výraz

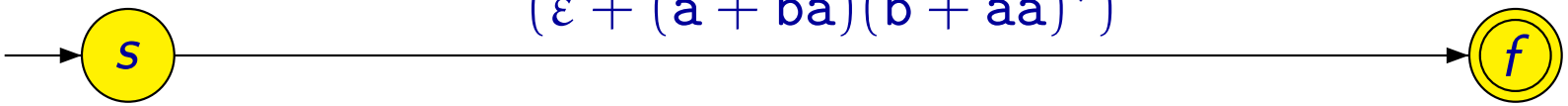
Příklad:



Převod konečného automatu na regulární výraz

Příklad:

$$\mathtt{a}(\mathtt{b}+\mathtt{aa})^*+$$
$$(\mathtt{b}+\mathtt{a}(\mathtt{b}+\mathtt{aa})^*\mathtt{ab})$$
$$(\mathtt{bb}+(\mathtt{a}+\mathtt{ba})(\mathtt{b}+\mathtt{aa})^*\mathtt{ab})^*$$
$$(\varepsilon+(\mathtt{a}+\mathtt{ba})(\mathtt{b}+\mathtt{aa})^*)$$



Ekvivalence konečných automatů a regulárních výrazů

Věta

Jazyk je regulární právě tehdy, když je ho možné popsat regulárním výrazem.