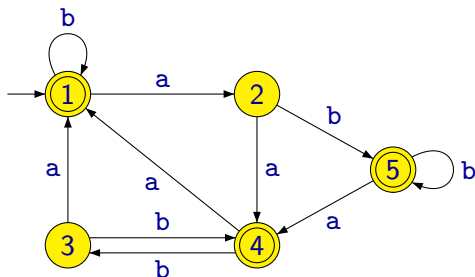


Jazyk rozpoznávaný automatem

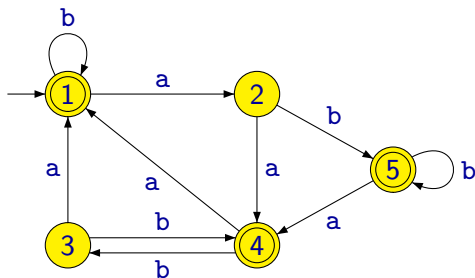


Uvažujme libovolný **tah** v grafu takový, že:

- Začíná v počátečním stavu.
- Končí v některém z přijímajících stavů.

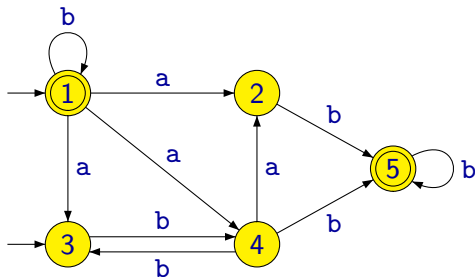
Symbole, jimiž jsou ohodnoceny hrany (tj. přechody) v tomto tahu, tvoří **slovo**, které je přijímáno daným automatem.

Jazyk rozpoznávaný automatem



Jazyk rozpoznávaný automatem je množina všech slov, pro které v grafu existuje takovýto tah.

Nedeterministický konečný automat



Je očividné, že pokud jazyk definujeme tímto způsobem, nemusíme se omezovat na grafy, kde:

- Z každého stavu vede právě jedna hrana označená daným symbolem abecedy.
- Máme právě jeden počáteční stav.

Nedeterministický konečný automat

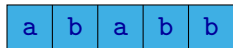
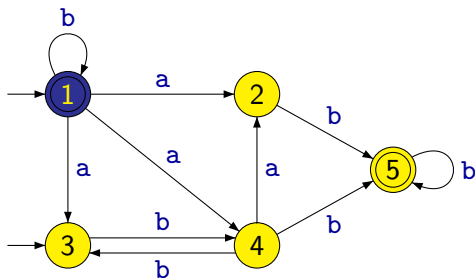
Takto obecněji definovaný automat se nazývá **nedeterministický konečný automat**.

Rozdíly oproti deterministickým konečným automatům:

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

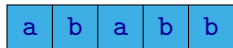
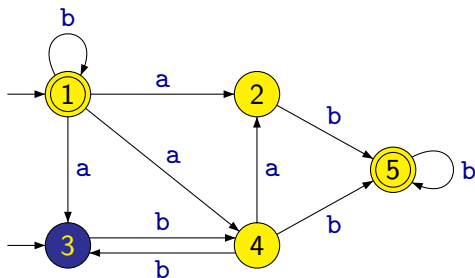
Pokud se na automat díváme jako na zařízení čtoucí slovo, vidíme, že jednomu slovu může odpovídat více než jeden výpočet (nebo naopak žádný).

Nedeterministický konečný automat



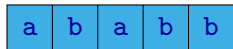
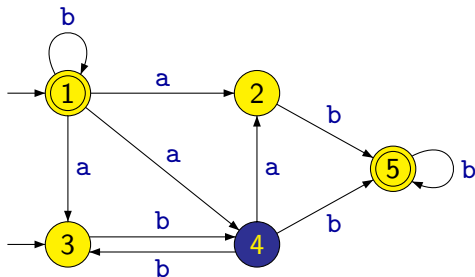
(1, ababb)

Nedeterministický konečný automat



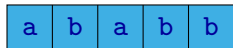
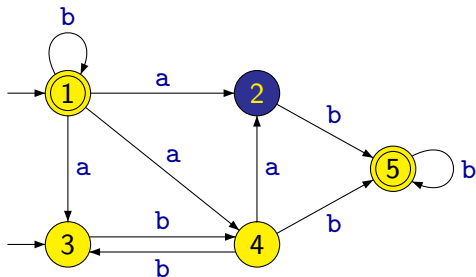
(1, ababb)
⊢ (3, babb)

Nedeterministický konečný automat



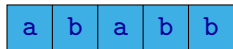
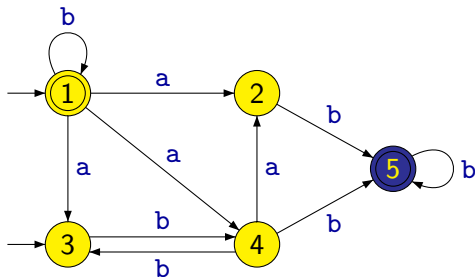
- (1, ababb)
- ┆ (3, babb)
- ┆ (4, abb)

Nedeterministický konečný automat



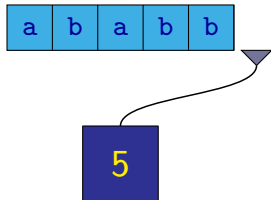
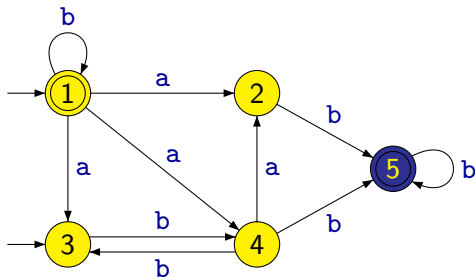
- (1, ababb)
- ┆ (3, babb)
- ┆ (4, abb)
- ┆ (2, bb)

Nedeterministický konečný automat



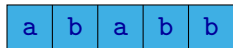
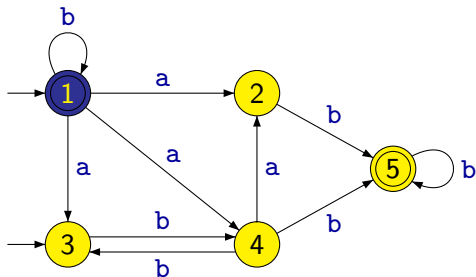
- (1, ababb)
- ⊢ (3, babb)
- ⊢ (4, abb)
- ⊢ (2, bb)
- ⊢ (5, b)

Nedeterministický konečný automat



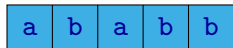
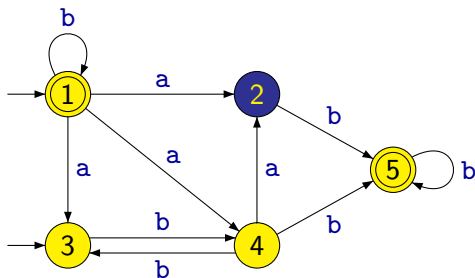
- (1, ababb)
- ⊢ (3, babb)
- ⊢ (4, abb)
- ⊢ (2, bb)
- ⊢ (5, b)
- ⊢ (5, ε)

Nedeterministický konečný automat



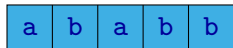
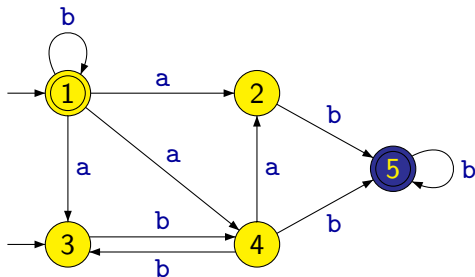
(1, ababb)

Nedeterministický konečný automat



(1, ababb)
└ (2, babb)

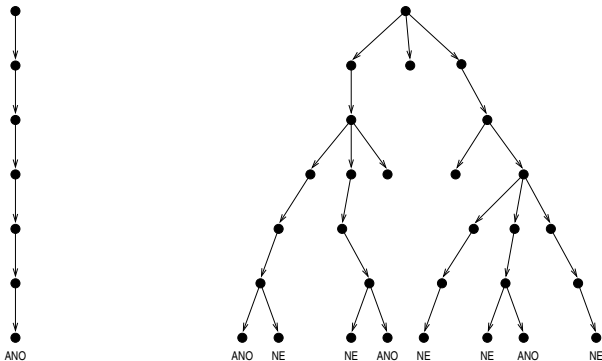
Nedeterministický konečný automat



(1, ababb)
└ (2, babb)
└ (5, abb)

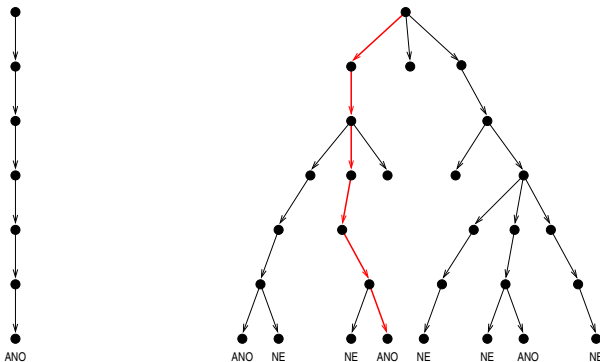
Nedeterministický konečný automat

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



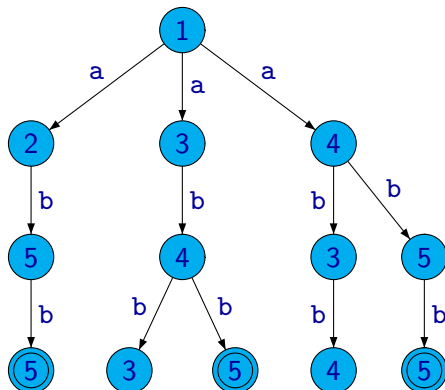
Nedeterministický konečný automat

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



Nedeterministický konečný automat

	a	b
↔ 1	2, 3, 4	1
2	—	5
→ 3	—	4
4	2	3, 5
← 5	—	5



3

Příklad: Les reprezentující všechny možné výpočty nad slovem **abb**.

Nedeterministický konečný automat

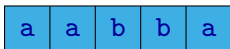
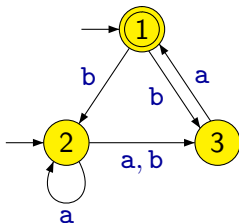
Formálně je **nedeterministický konečný automat** definován jako pětice

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

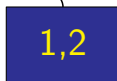
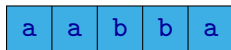
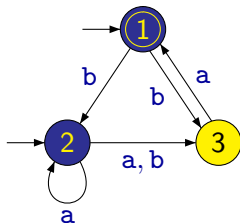
kde:

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

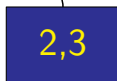
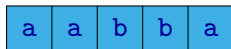
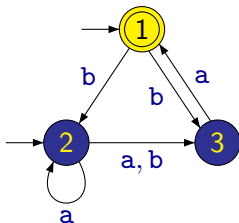
Převod nedeterministického automatu na deterministický



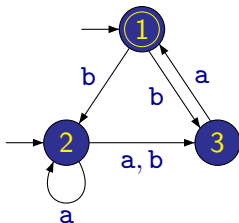
Převod nedeterministického automatu na deterministický



Převod nedeterministického automatu na deterministický



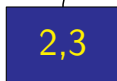
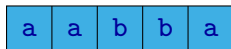
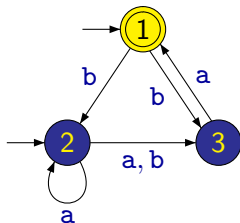
Převod nedeterministického automatu na deterministický



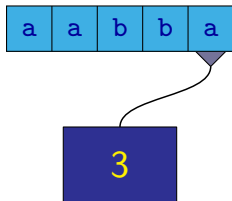
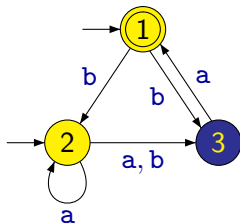
a a b b a

1,2,3

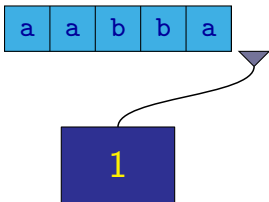
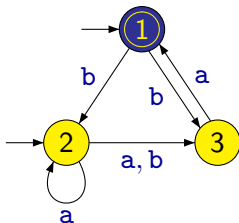
Převod nedeterministického automatu na deterministický



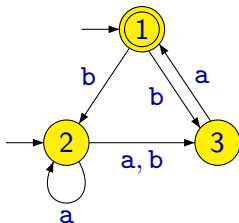
Převod nedeterministického automatu na deterministický



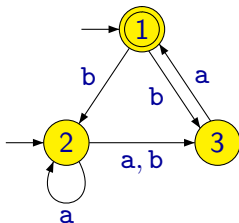
Převod nedeterministického automatu na deterministický



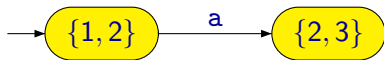
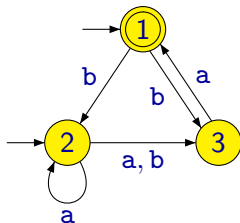
Převod nedeterministického automatu na deterministický



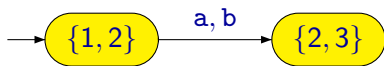
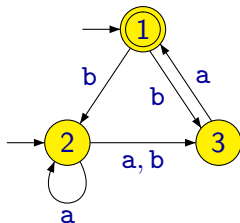
Převod nedeterministického automatu na deterministický



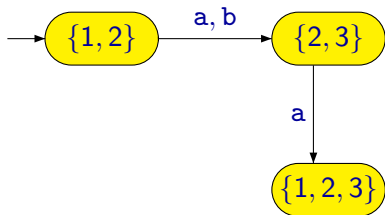
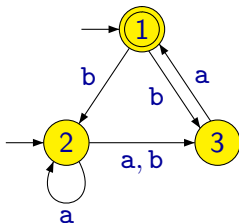
Převod nedeterministického automatu na deterministický



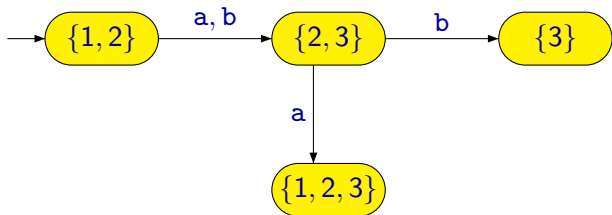
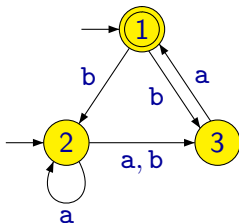
Převod nedeterministického automatu na deterministický



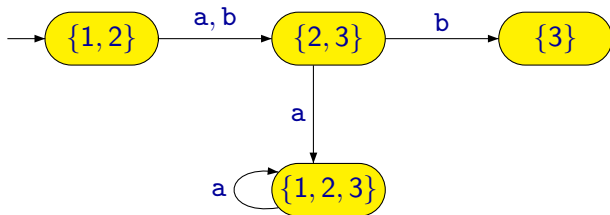
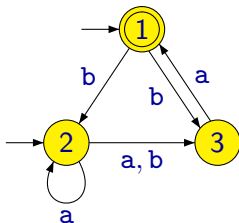
Převod nedeterministického automatu na deterministický



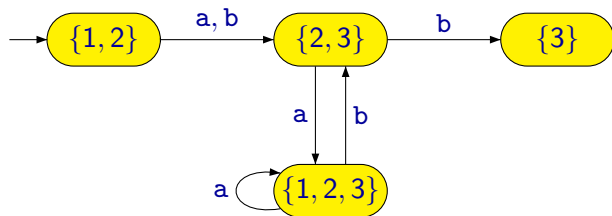
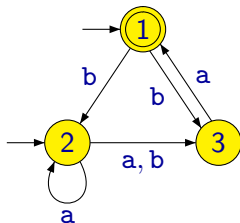
Převod nedeterministického automatu na deterministický



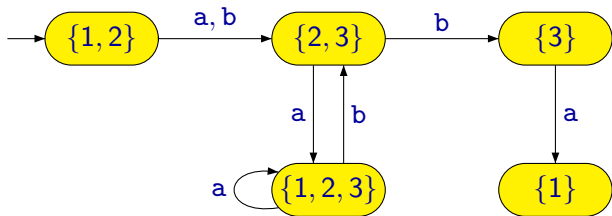
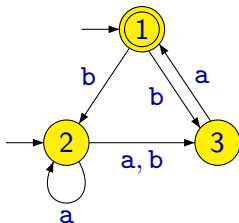
Převod nedeterministického automatu na deterministický



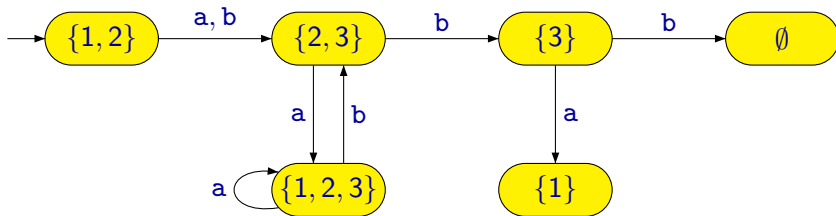
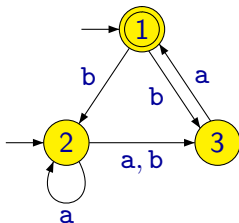
Převod nedeterministického automatu na deterministický



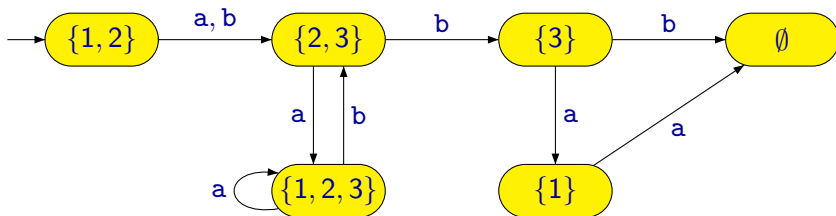
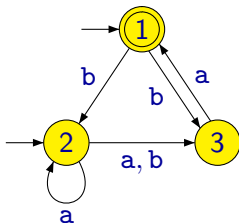
Převod nedeterministického automatu na deterministický



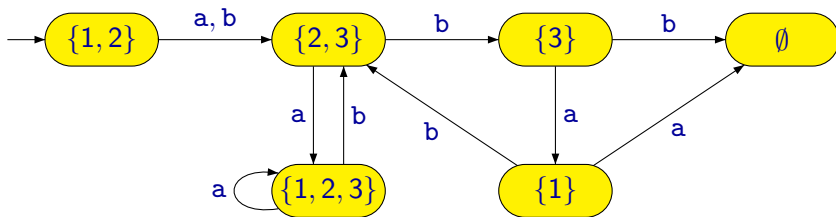
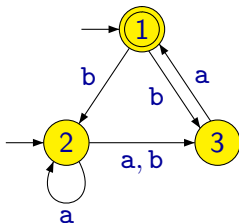
Převod nedeterministického automatu na deterministický



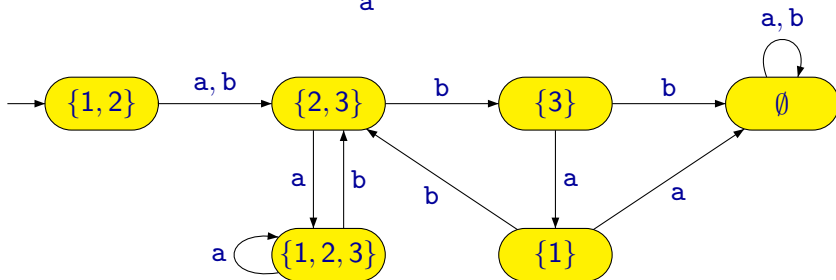
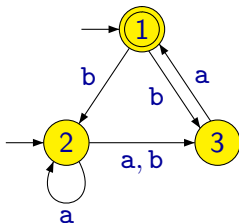
Převod nedeterministického automatu na deterministický



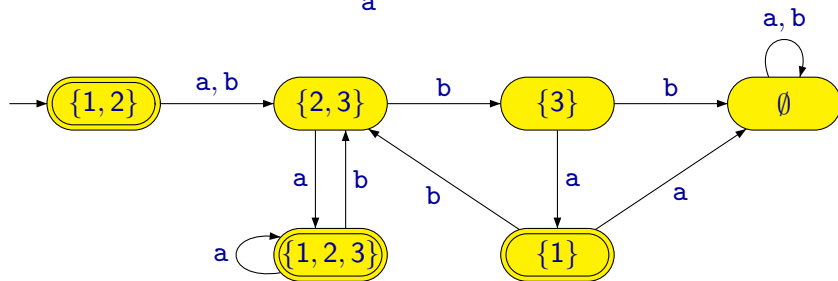
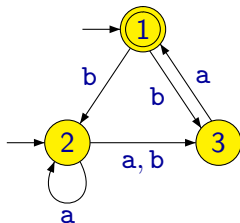
Převod nedeterministického automatu na deterministický



Převod nedeterministického automatu na deterministický



Převod nedeterministického automatu na deterministický



Převod nedeterministického automatu na deterministický

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

Převod nedeterministického automatu na deterministický

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

	a	b

Převod nedeterministického automatu na deterministický

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

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

Převod nedeterministického automatu na deterministický

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

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

Převod nedeterministického automatu na deterministický

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

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

Převod nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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 nedeterministického automatu na deterministický

	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

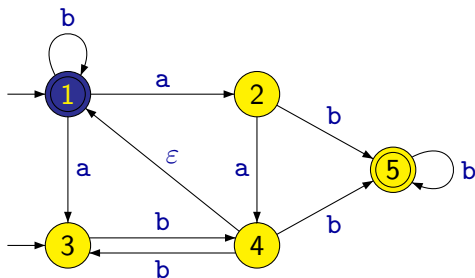
	a	b
$\leftrightarrow 1$	2	2
2	3	4
$\leftarrow 3$	3	2
4	5	6
$\leftarrow 5$	6	2
6	6	6

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

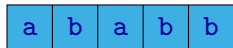
Například při převodu automatu, který má 20 stavů, může vzniknout automat, který má $2^{20} = 1048576$ stavů.

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

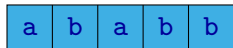
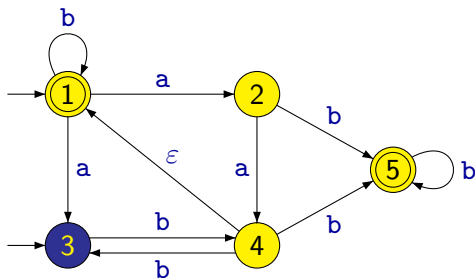
Zobecněný nedeterministický konečný automat



(1, ababb)

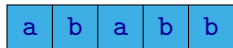
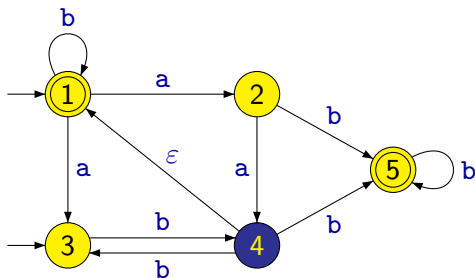


Zobecněný nedeterministický konečný automat



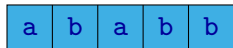
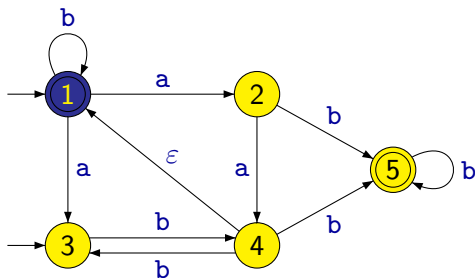
(1, ababb)
⊢ (3, babb)

Zobecněný nedeterministický konečný automat



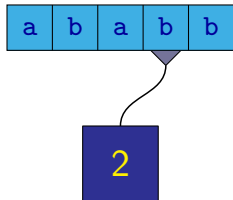
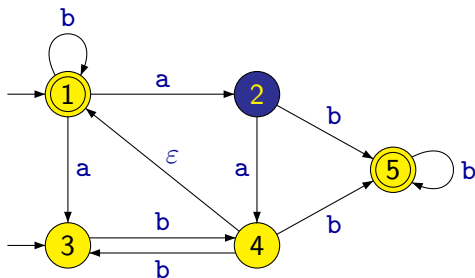
(1, ababb)
└ (3, babb)
└ (4, abb)

Zobecněný nedeterministický konečný automat



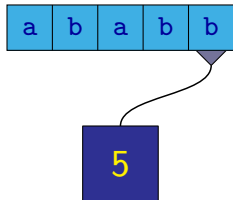
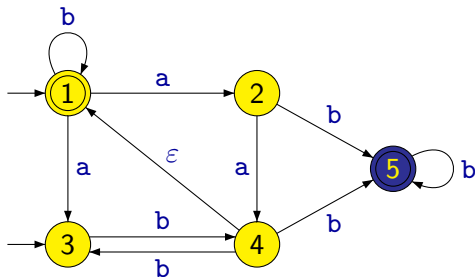
- (1, ababb)
- ⊢ (3, babb)
- ⊢ (4, abb)
- ⊢ (1, abb)

Zobecněný nedeterministický konečný automat



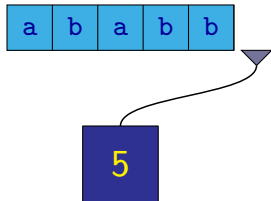
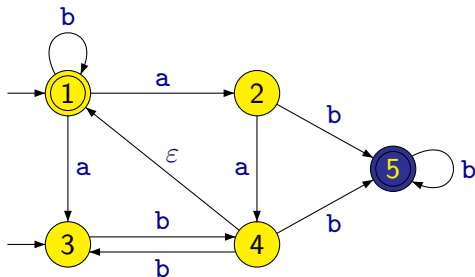
- (1, ababb)
- ⊢ (3, babb)
- ⊢ (4, abb)
- ⊢ (1, abb)
- ⊢ (2, bb)

Zobecněný nedeterministický konečný automat



- (1, ababb)
- ⊢ (3, babb)
- ⊢ (4, abb)
- ⊢ (1, abb)
- ⊢ (2, bb)
- ⊢ (5, b)

Zobecněný nedeterministický konečný automat



- (1, ababb)
- ⊢ (3, babb)
- ⊢ (4, abb)
- ⊢ (1, abb)
- ⊢ (2, bb)
- ⊢ (5, b)
- ⊢ (5, ε)

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 řídicí 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** definován jako pětice

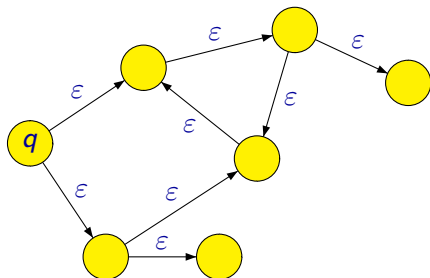
$$(Q, \Sigma, \delta, I, F)$$

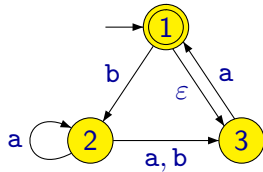
kde:

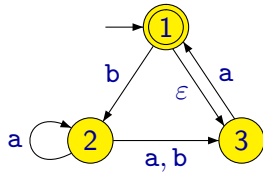
- Q je konečná množina **stavů**
- Σ je konečná **abeceda**
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$ je **přechodová funkce**
- $I \subseteq Q$ je množina **počátečních stavů**
- $F \subseteq Q$ je množina **přijímajících stavů**

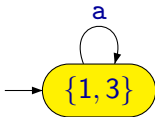
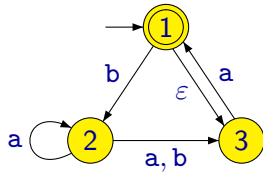
Převod na deterministický konečný automat

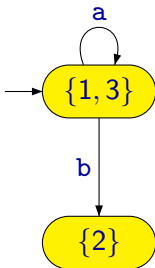
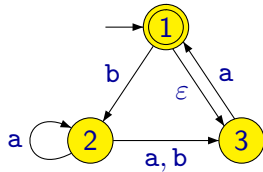
Zobecněný nedeterministický konečný automat je možné převést na deterministický podobnou konstrukcí jako nedeterministický konečný automat, s tím rozdílem, že do množin stavů musíme vždy přidat navíc i všechny stavy dosažitelné daných stavů nějakou sekvencí ϵ -přechodů.

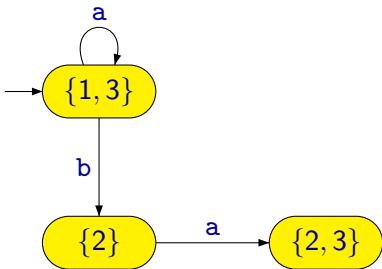
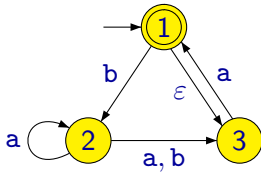


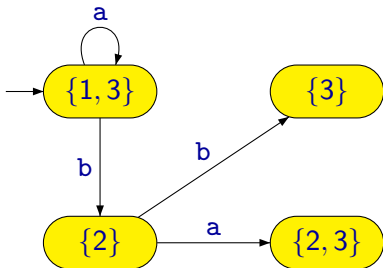
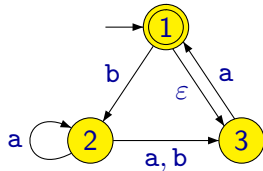


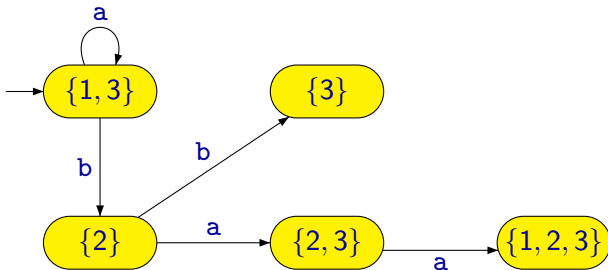
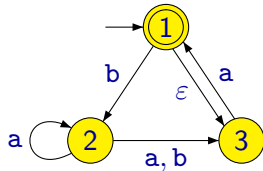


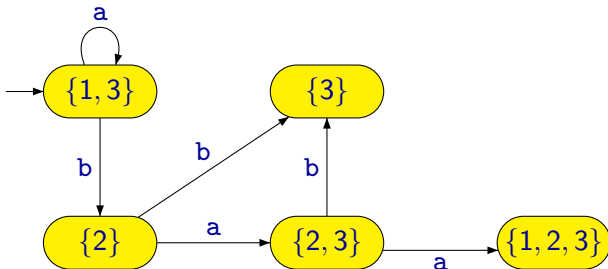
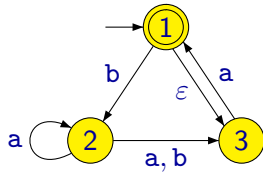


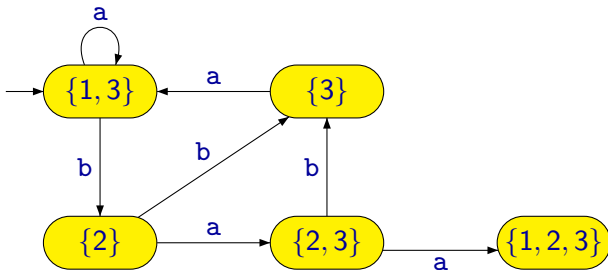
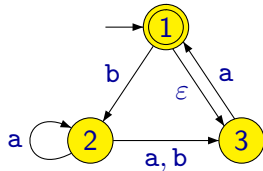


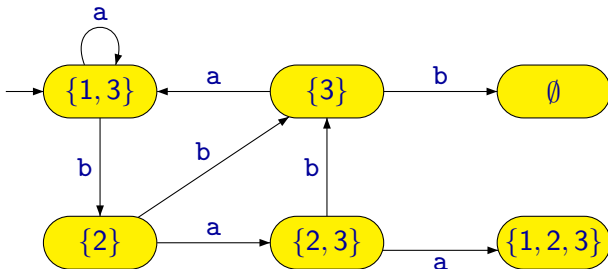
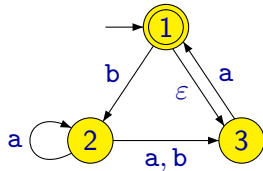


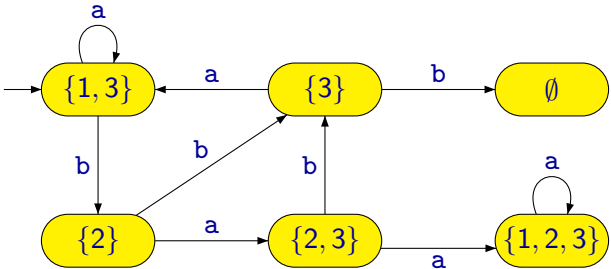
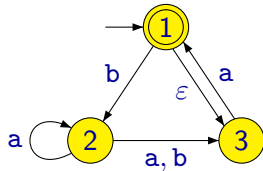


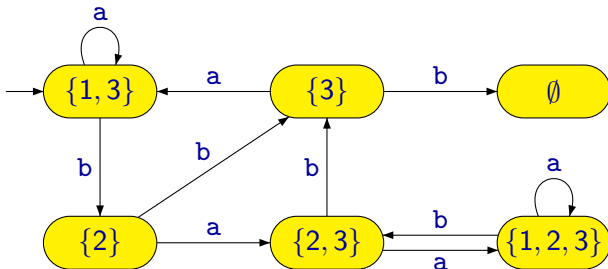
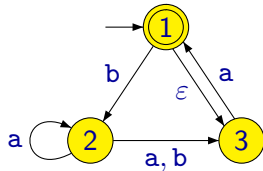


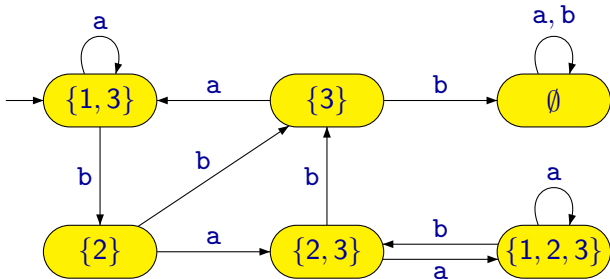
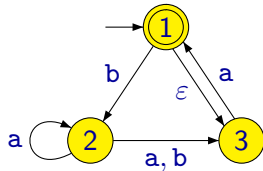


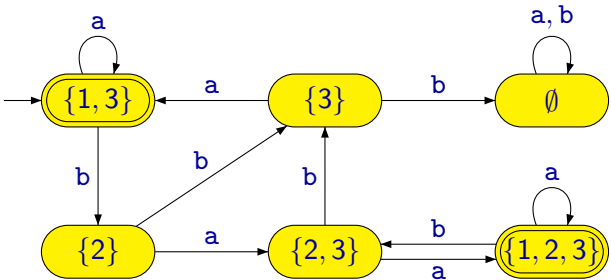
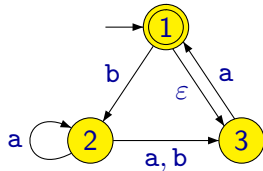






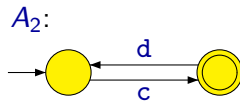
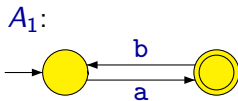






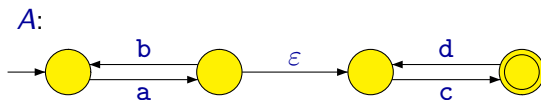
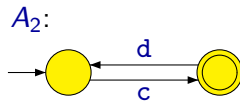
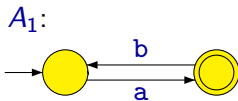
Zřetězení jazyků

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



Zřetězení jazyků

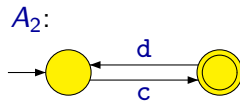
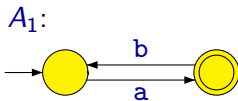
$$\Sigma = \{a, b, c, d\}$$



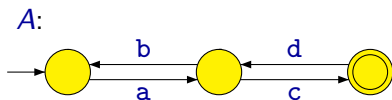
$$L(A) = L(A_1) \cdot L(A_2)$$

Zřetězení jazyků

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

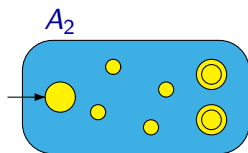
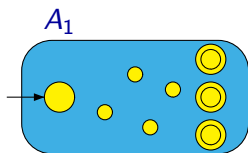


Chybná konstrukce:

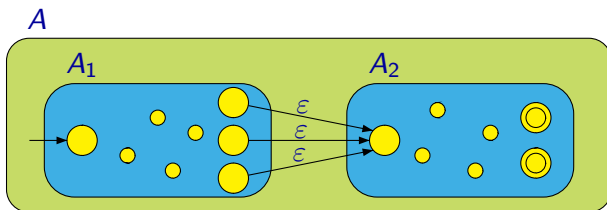
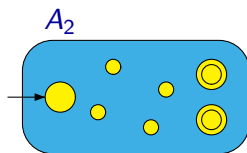
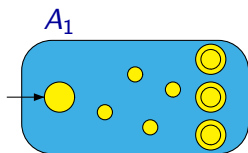


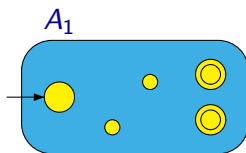
$acdbac \in L(A)$, ale $acdbac \notin (A_1) \cdot L(A_2)$

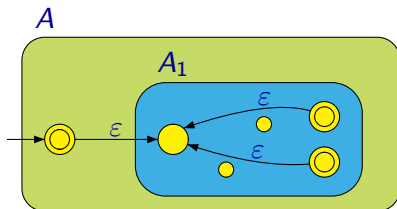
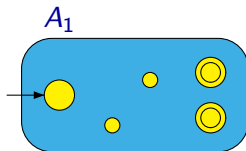
Zřetězení jazyků



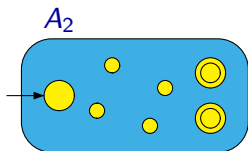
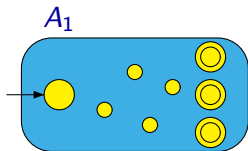
Zřetězení jazyků





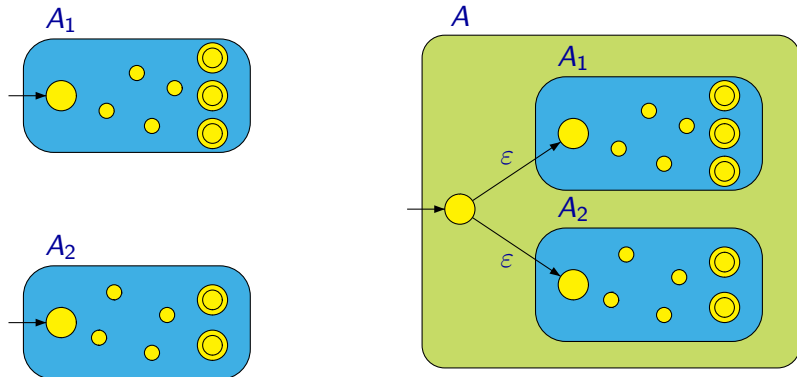


Alternativní konstrukce pro sjednocení jazyků:



Sjednocení jazyků

Alternativní konstrukce pro sjednocení jazyků:



Předpokládejme, že máme dány množiny X a Y , kde $X \subseteq Y$, a dále nějakou operaci $f : Y \times Y \times \cdots \times Y \rightarrow Y$.

O množině X řekneme, že je **uzavřená** vůči operaci f , jestliže platí, že pokud $x_1, x_2, \dots, x_k \in X$, pak $f(x_1, x_2, \dots, x_k) \in X$.

Příklad: Uvažujme množinu přirozených čísel $\mathbb{N} = \{0, 1, 2, 3, \dots\}$ a množinu reálných čísel \mathbb{R} .

- Množina \mathbb{N} je uzavřená vůči operacím $+$ a \times , ale není uzavřená vůči operacím $-$ a $/$.
Například $3 + 5 \in \mathbb{N}$, ale $3 - 5 \notin \mathbb{N}$ a $3/5 \notin \mathbb{N}$
- Množina \mathbb{R} je uzavřená vůči operacím $+$, \times , $-$, $/$.

Množina (všech) regulárních jazyků je uzavřená vůči operacím:

- sjednocení
- průniku
- doplňku
- zřetězení
- iteraci
- ...

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

Existují i jazyky, které nejsou regulární.

Regulární výrazy

Jako například v aritmetice můžeme pomocí operátorů $+$ a \times vytvářet výrazy jako

$$(5 + 3) \times 4$$

můžeme v teorii formálních jazyků pomocí operátorů $+$, \cdot a $*$ vytvářet tzv. **regulární výrazy**, jako třeba

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

které reprezentují jazyky.

Jako je hodnotou aritmetického výrazu $(5 + 3) \times 4$ číslo 32, je hodnotou regulárního výrazu $(0 + 1) \cdot 0^*$ jazyk

$$(\{0\} \cup \{1\}) \cdot \{0\}^*$$

Induktivní definice regulárních výrazů 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.

Příklad:

- Podle definice jsou 0 i 1 regulární výrazy.

Příklad:

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

Příklad:

- 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.

Příklad:

- 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.

Příklad:

- 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 .

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

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

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

Příklad: Místo

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

obvykle píšeme

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

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

0 ... jazyk tvořený jediným slovem 0

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

0 ... jazyk tvořený jediným slovem 0

01 ... jazyk tvořený jediným slovem 01

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

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, ...

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 $\varepsilon, 0, 00, 000, \dots$

$(01)^*$... jazyk tvořený slovy $\varepsilon, 01, 0101, 010101, \dots$

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 $\varepsilon, 0, 00, 000, \dots$

$(01)^*$... jazyk tvořený slovy $\varepsilon, 01, 0101, 010101, \dots$

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

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 $\varepsilon, 0, 00, 000, \dots$

$(01)^*$... jazyk tvořený slovy $\varepsilon, 01, 0101, 010101, \dots$

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

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

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 $\varepsilon, 0, 00, 000, \dots$

$(01)^*$... jazyk tvořený slovy $\varepsilon, 01, 0101, 010101, \dots$

$(0 + 1)^*$... 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

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

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

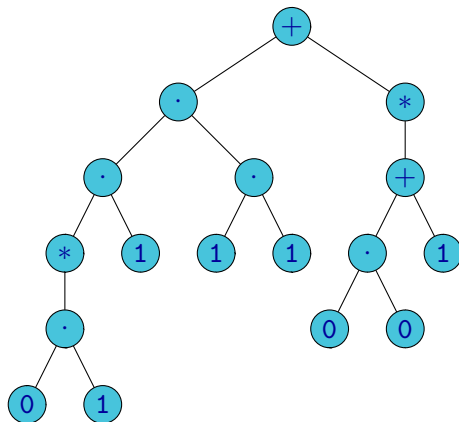
$(0 + 1)^*1(0 + 1)^*$... jazyk tvořený všemi slovy obsahujícími alespoň jeden symbol 1

$(0 + 1)^*00 + (01)^*111(01)^*$... 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

Strukturu regulárního výrazu si můžeme znázornit jako strom:



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

Tvrzení

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

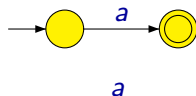
Důkaz: Stačí ukázat, jak k danému regulárnímu výrazu α zkonstruovat konečný automat, který rozpoznává jazyk $[\alpha]$.

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 $[\alpha]$.

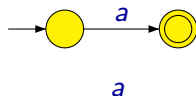
Převod regulárního výrazu na konečný automat

Automaty pro elementární výrazy:

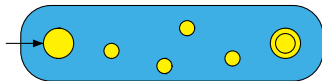
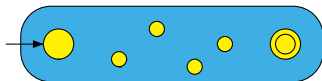


Převod regulárního výrazu na konečný automat

Automaty pro elementární výrazy:

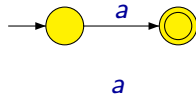
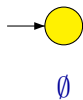


Konstrukce pro sjednocení:

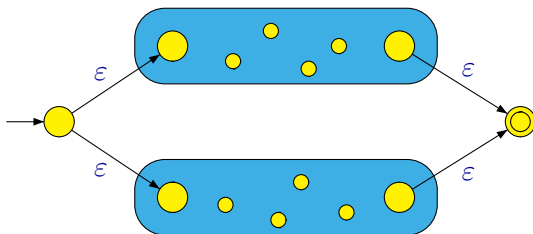


Převod regulárního výrazu na konečný automat

Automaty pro elementární výrazy:

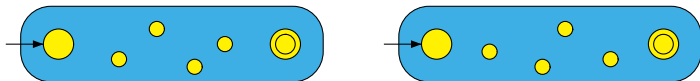


Konstrukce pro sjednocení:



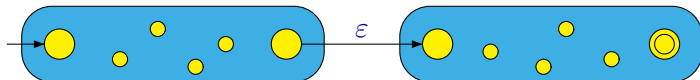
Převod regulárního výrazu na konečný automat

Konstrukce pro zřetězení:



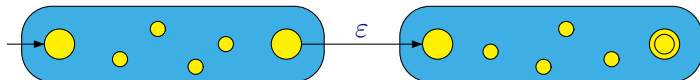
Převod regulárního výrazu na konečný automat

Konstrukce pro zřetězení:

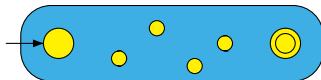


Převod regulárního výrazu na konečný automat

Konstrukce pro zřetězení:

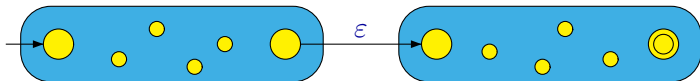


Konstrukce pro iteraci:

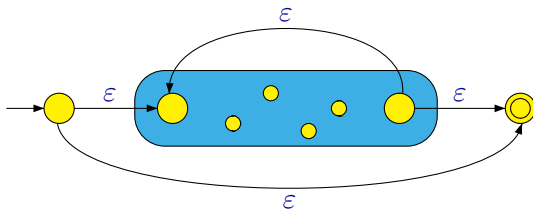


Převod regulárního výrazu na konečný automat

Konstrukce pro zřetězení:



Konstrukce pro iteraci:

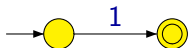


Příklad: Konstrukce automatu pro výraz $((0 + 1) \cdot 1)^*$:

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

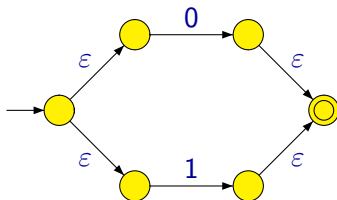


Příklad: Konstrukce automatu pro výraz $((0 + 1) \cdot 1)^*$:



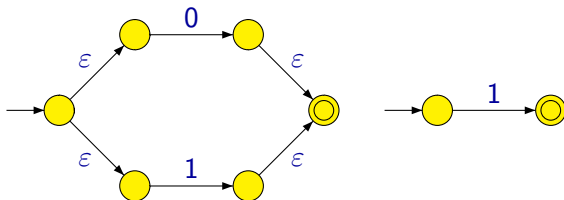
Převod regulárního výrazu na konečný automat

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



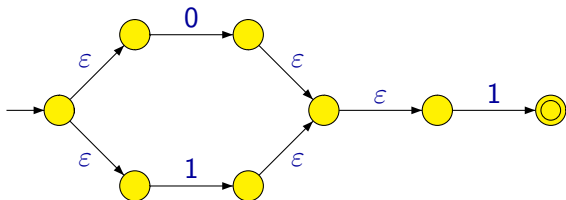
Převod regulárního výrazu na konečný automat

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



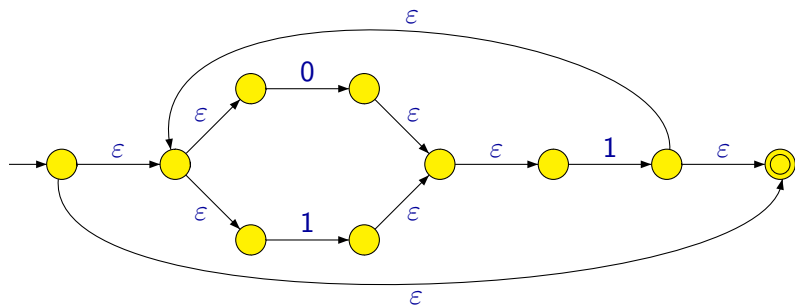
Převod regulárního výrazu na konečný automat

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.

Tvrzení

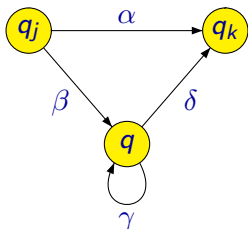
Každý regulární jazyk je možné popsat nějakým regulárním výrazem.

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

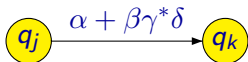
- A upravíme tak, aby měl právě jeden počáteční a právě jeden koncový stav.
- Budeme postupně odebírat jednotlivé stavy.
- Přejchody budou označeny regulárními výrazy.
- Zbude automat se dvěma stavy – počátečním a koncovým, a jedním přechodem ohodnoceným výsledným regulárním výrazem.

Převod konečného automatu na regulární výraz

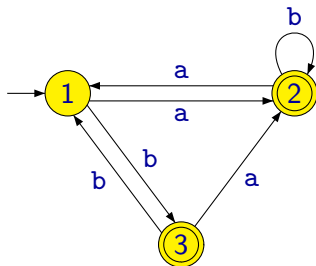
Hlavní myšlenka: Při odstraňování stavu q nahradit pro každou dvojici zbylých stavů q_j , q_k cestu z q_j do q_k vedoucí přes q .



Po odstranění stavu q :

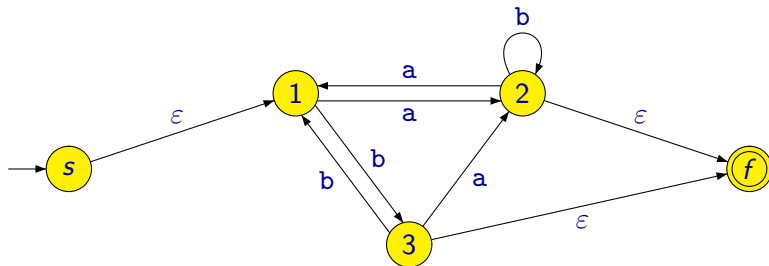


Příklad:



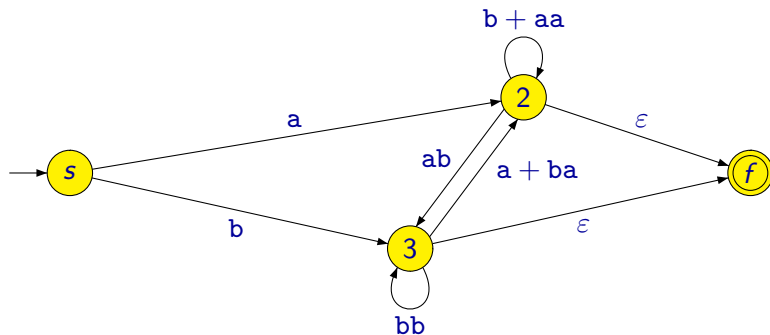
Převod konečného automatu na regulární výraz

Příklad:



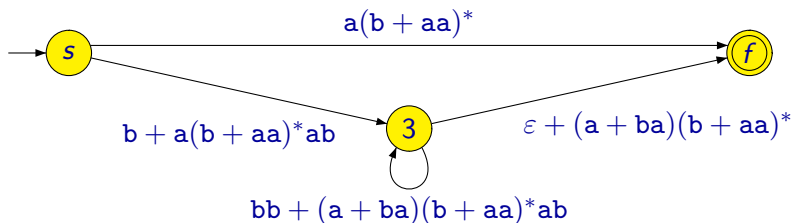
Převod konečného automatu na regulární výraz

Příklad:



Převod konečného automatu na regulární výraz

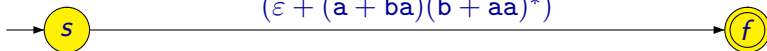
Příklad:



Převod konečného automatu na regulární výraz

Příklad:

$$\begin{aligned} & a(b + aa)^* + \\ & (b + a(b + aa)^* ab) \\ & (bb + (a + ba)(b + aa)^* ab)^* \\ & (\varepsilon + (a + ba)(b + aa)^*) \end{aligned}$$



Věta

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

Regulární výrazy jsou používány v celé řadě různých nástrojů.

Příklady:

- Knihovna `regex` jazyka C.
- Package `java.util.regex` v jazyce Java.
- Modul `re` v jazyce Python.
- Programovací jazyk Perl.
- Unixové utility pro zpracování textových souborů `grep`, `sed` a `awk`.
- Generátory lexikálních analyzátorů `lex` a `flex`.
- Textové editory (`vi`, `vim`, `emacs`, ...).

V praxi používané regulární výrazy

Běžně používaná syntaxe (mezi jednotlivými nástroji jsou však drobné rozdíly):

- \cdot ... zastupuje libovolný znak
- $\alpha\beta$... zřetězení α a β
- $\alpha|\beta$... sjednocení α a β
- α^* ... iterace α
- α^+ ... totéž, co $\alpha\alpha^*$
- $\alpha?$... totéž, co $\alpha + \epsilon$
- $\alpha\{m\}$... totéž co m krát α
- $\alpha\{m,n\}$... α minimálně m krát, maximálně n krát
- (α) ... závorky

V praxi používané regulární výrazy

- [xyz] ... libovolný ze znaků **x**, **y**, **z**
- [^xyz] ... libovolný znak, kromě **x**, **y**, **z**
- [a-f] ... libovolný ze znaků **a**, **b**, **c**, **d**, **e**, **f**
- ^ ... začátek řádku
- \$... konec řádku
- \c ... znak **c**

Příklad: správně vytvořená e-mailová adresa

```
^[a-zA-Z0-9\.\-]+@[a-zA-Z0-9\.\-]+\.[a-zA-Z]{2,4}$
```

Poznámka: Podrobnější informace najdete například v seriálu „Regulární výrazy“ autora Pavla Satrapy na

<http://www.root.cz/serialy/regularni-vyrazy/>