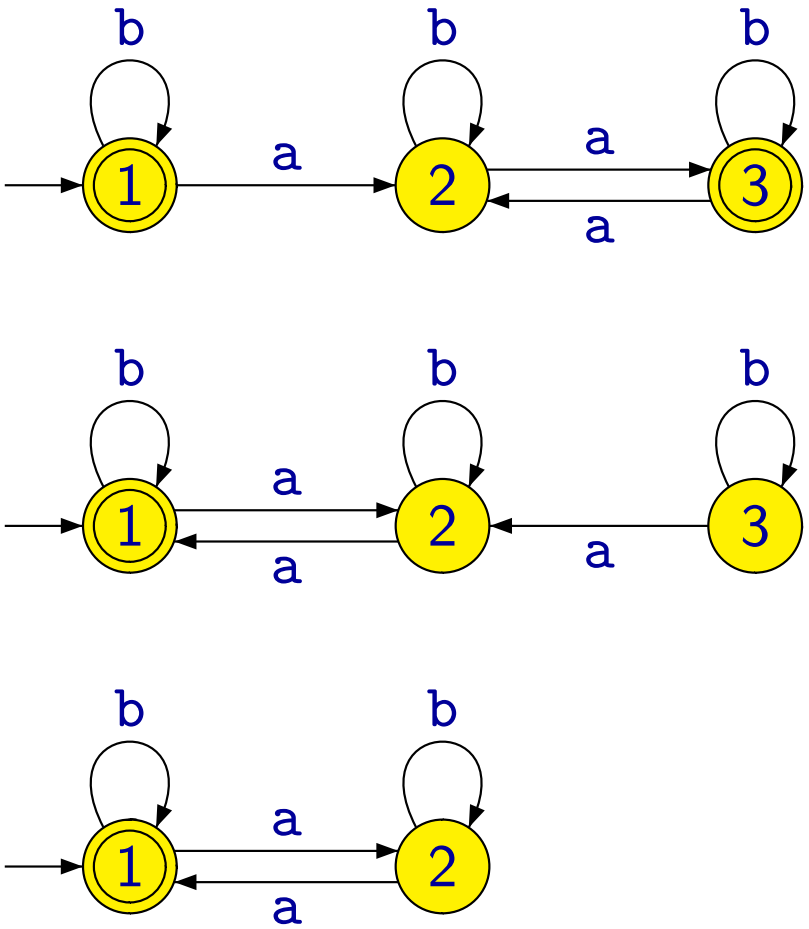
Minimalizace automatů

Ekvivalence automatů



- Všechny 3 automaty přijímají jazyk všech slov se sudým počtem *a*-ček
- Nejvýhodnější je pro nás poslední z nich - má nejméně stavů

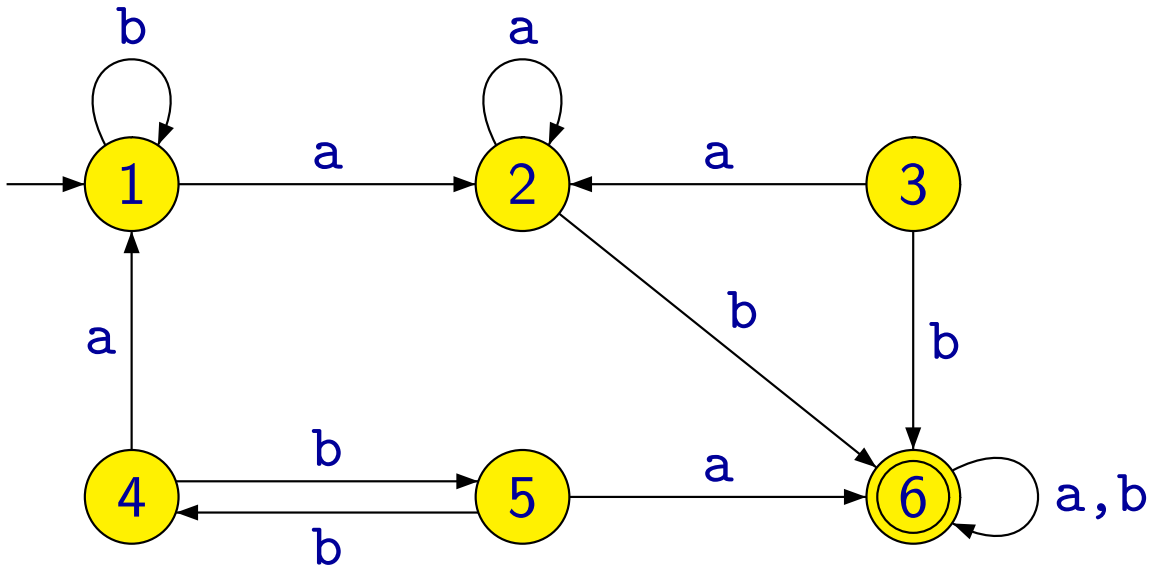
Ekvivalence automatů

Definice

O konečných automatech A_1, A_2 řekneme, že jsou **ekvivalentní**, jestliže $L(A_1) = L(A_2)$.

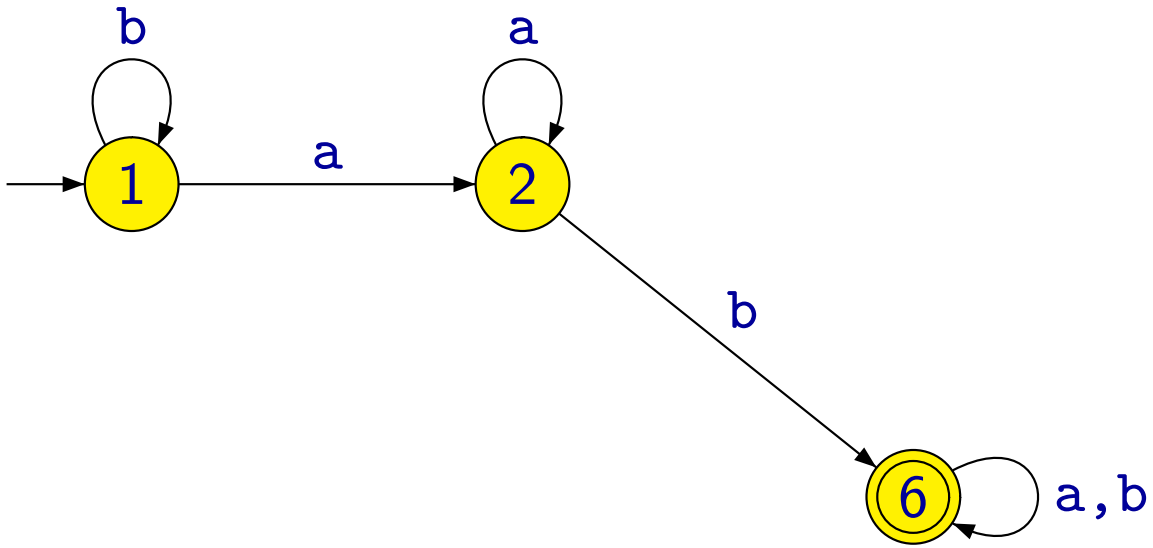
- Existuje nějaký vhodný algoritmus zjišťující, jestli jsou dva automaty ekvivalentní?
- Můžeme zjistit, jestli k danému automatu neexistuje nějaký ekvivalentní menší (s menším počtem stavů)?
- Když o nějakém automatu víme, že menší ekvivalentní už neexistuje, může existovat jiný stejně velký ekvivalentní automat?

Nedosažitelné stavy automatu



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

Nedosažitelné stavy automatu



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

Nedosažitelné stavy automatu

Definice

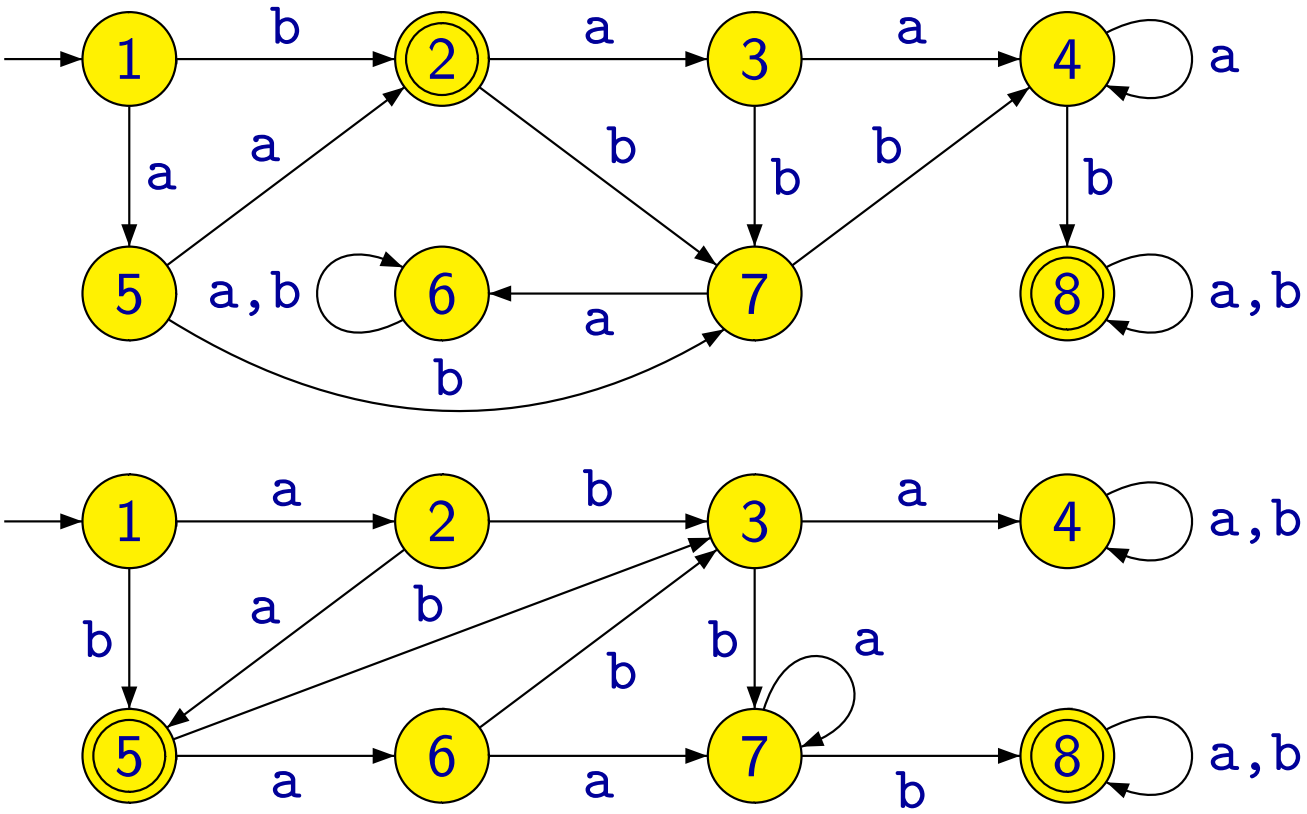
Stav q deterministického konečného automatu $\mathcal{A}$ je **dosažitelný slovem** w, pokud se výpočet $\mathcal{A}$ po přečtení celého slova w zastaví ve stavu q.

Definice

Stav q deterministického konečného automatu $\mathcal{A}$ je **dosažitelný** pokud existuje nějaké slovo, kterým je stav dosažitelný. V opačném případě stav nazýváme **nedosažitelný**.

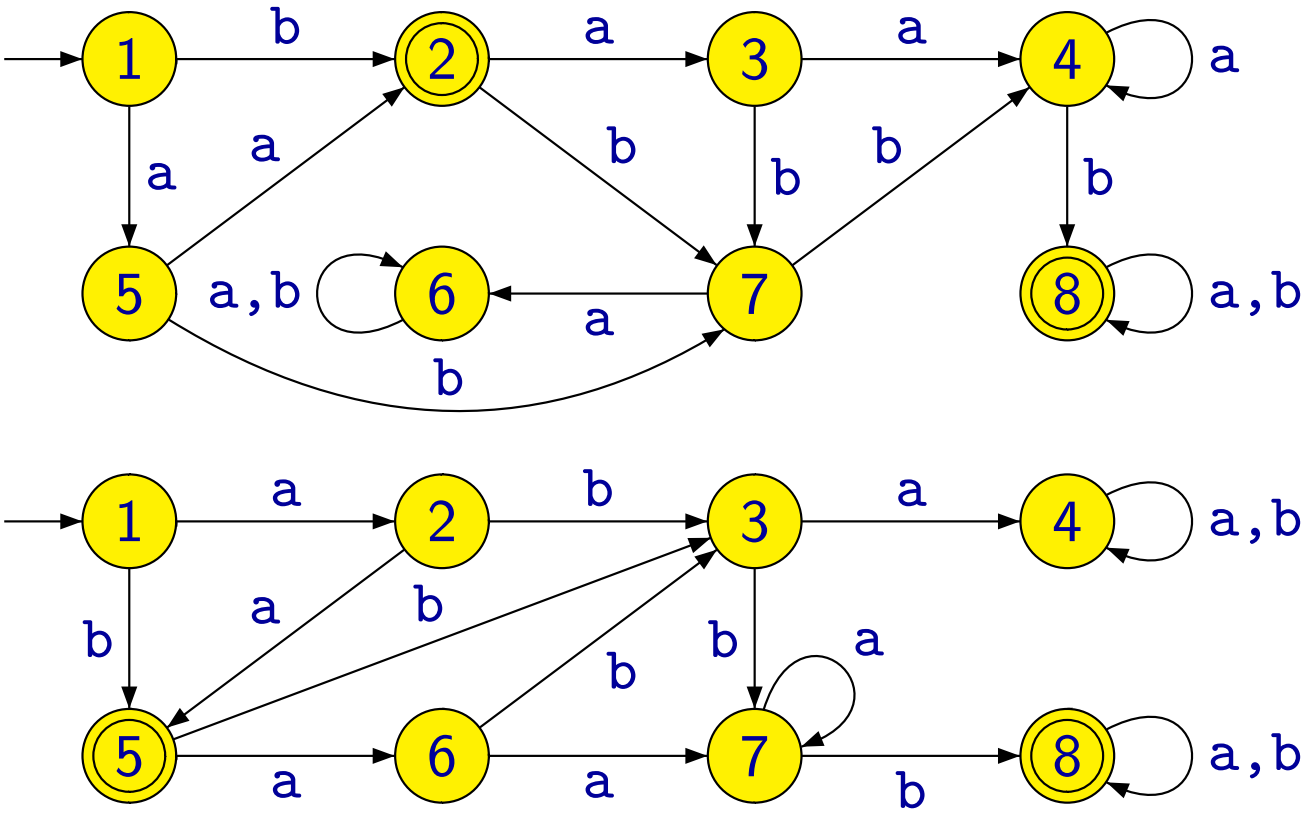
- Do nedosažitelných stavů nevede v grafu automatu žádná orientovaná cesta z počátečního stavu
- Nedosažitelné stavy můžeme z automatu odstranit (spolu se všemi přechody vedoucími z nich). Jazyk přijímaný automatem se nezmění.

Normovaný tvar automatu



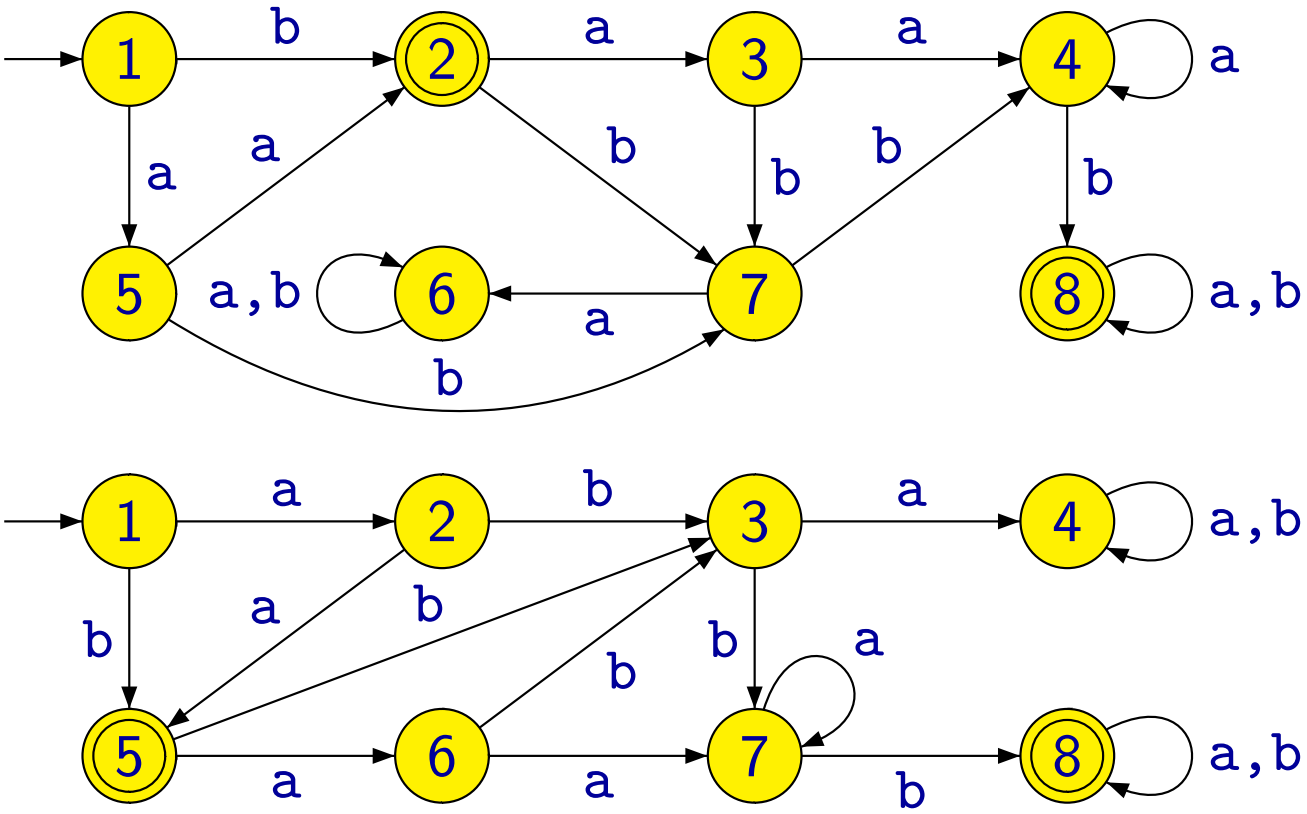
$\mathcal{A}_1$	a	b
$\rightarrow 1$	5	2
$\leftarrow 2$	3	7
3	4	7
4	4	8
5	2	7
6	6	6
7	6	4
$\leftarrow 8$	8	8

Normovaný tvar automatu



$\mathcal{A}_2$	a	b
$\rightarrow 1$	2	5
2	5	3
3	4	7
4	4	4
$\leftarrow 5$	6	3
6	7	3
7	7	8
$\leftarrow 8$	8	8

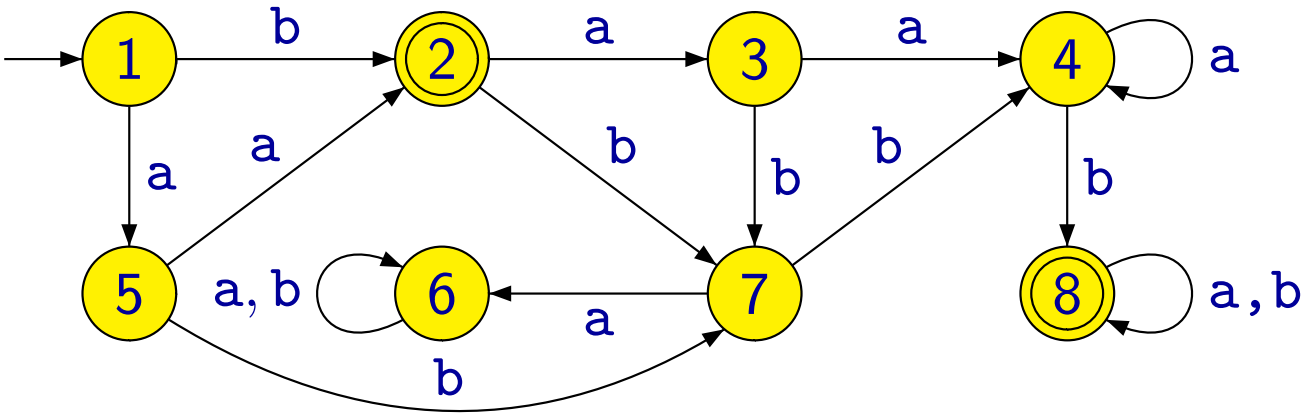
Normovaný tvar automatu



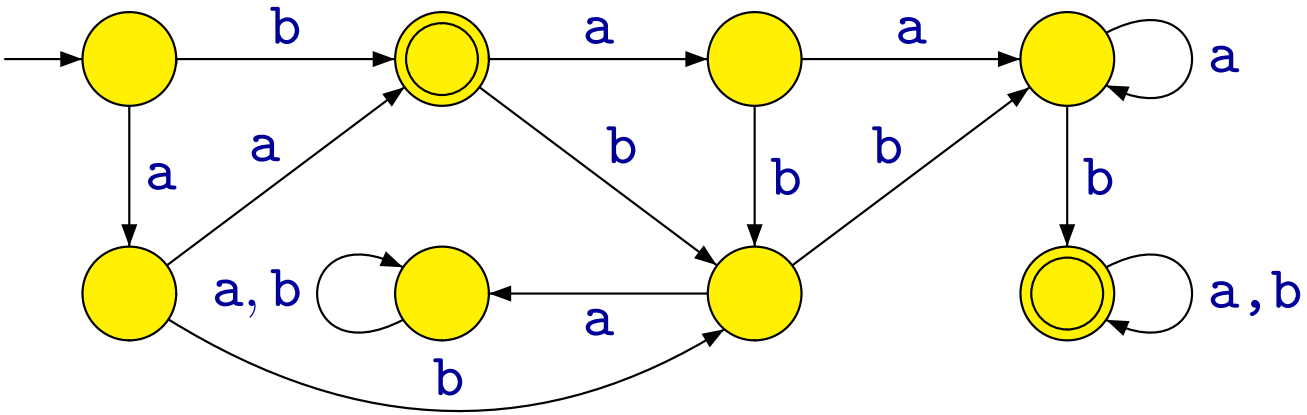
$\mathcal{A}_2$	a	b
$\rightarrow 1$	2	5
2	5	3
3	4	7
4	4	4
$\leftarrow 5$	6	3
6	7	3
7	7	8
$\leftarrow 8$	8	8

- Jak poznáme, že jsou automaty ekvivalentní, když se liší přechodová funkce?
- Automaty na obrázku můžu čísly $1-8$ pojmenovat $8!$ způsoby
- Chtěli bychom jednoznačně vybrat jedno z možných pojmenování

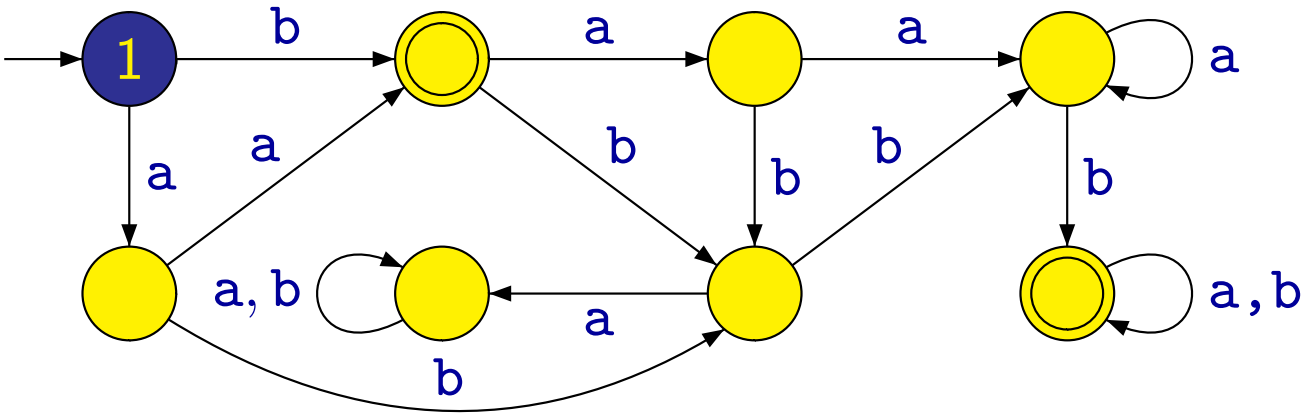
Normovaný tvar automatu



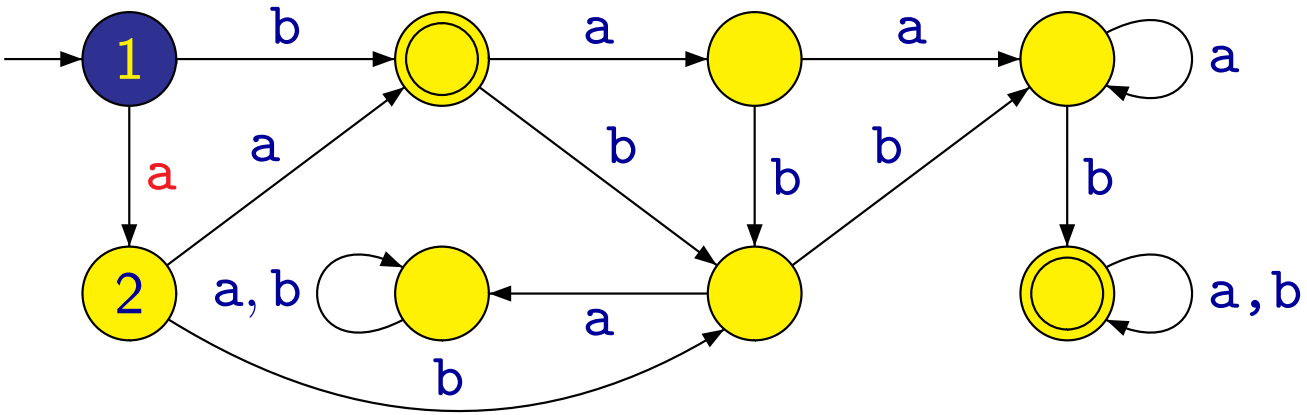
Normovaný tvar automatu



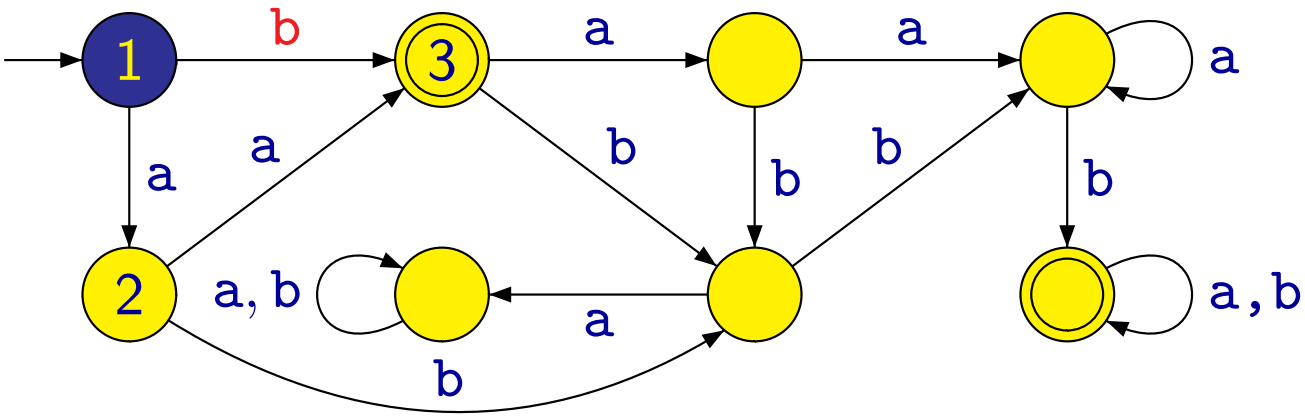
Normovaný tvar automatu



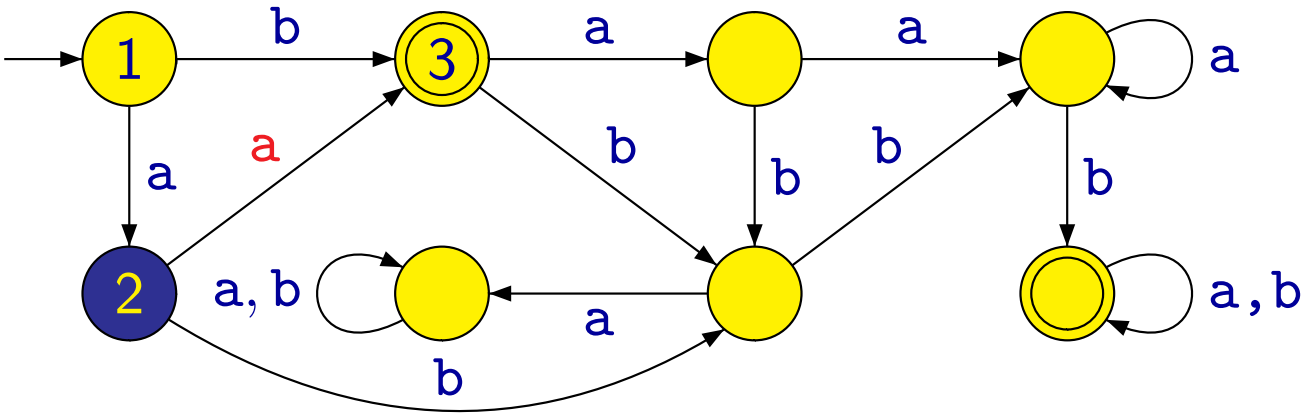
Normovaný tvar automatu



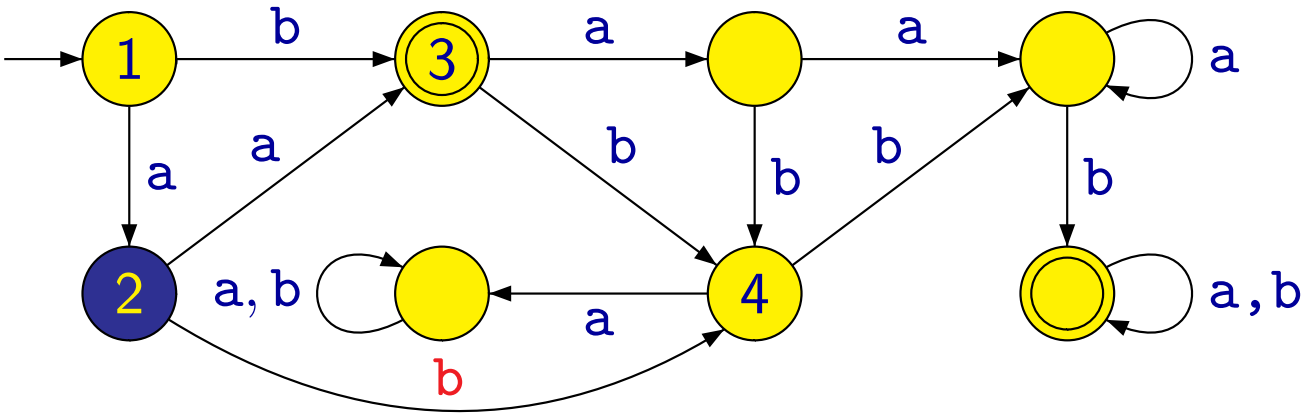
Normovaný tvar automatu



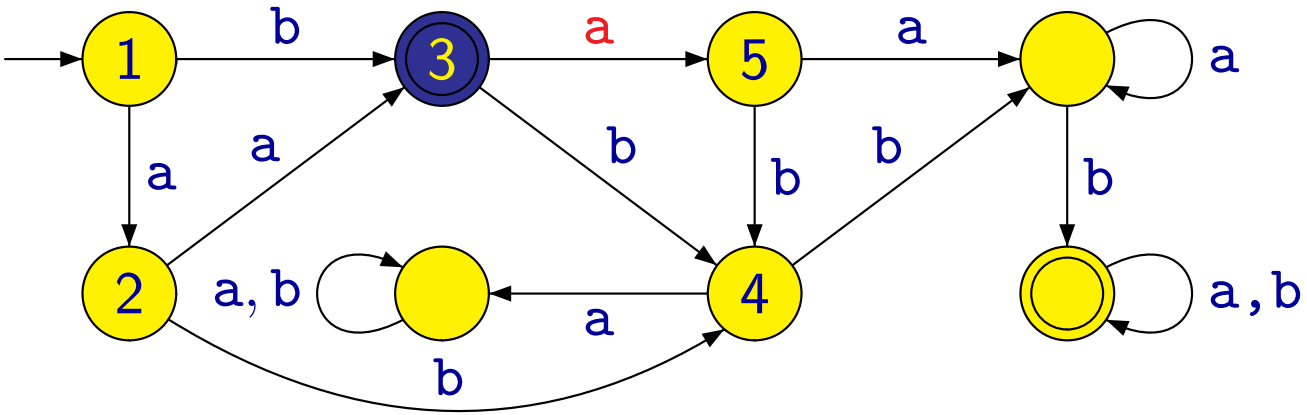
Normovaný tvar automatu



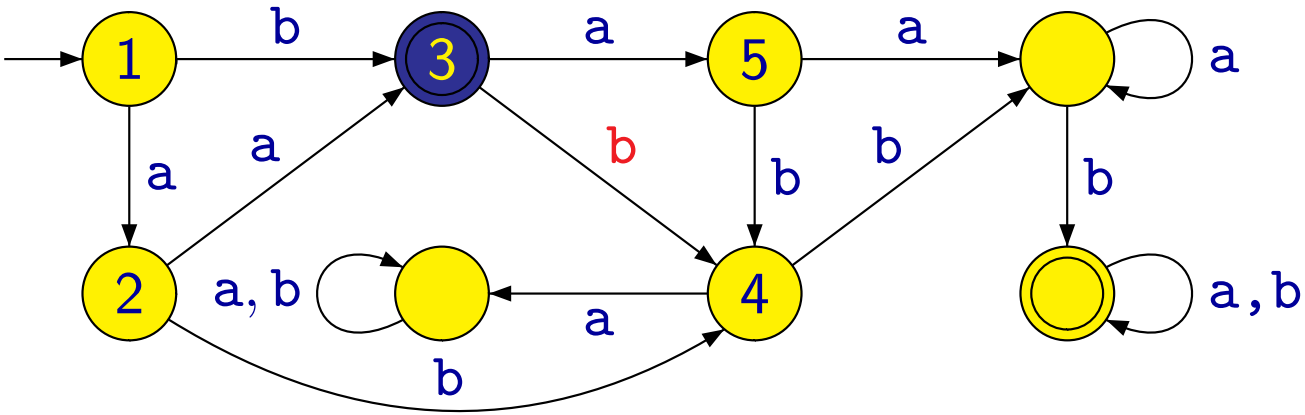
Normovaný tvar automatu



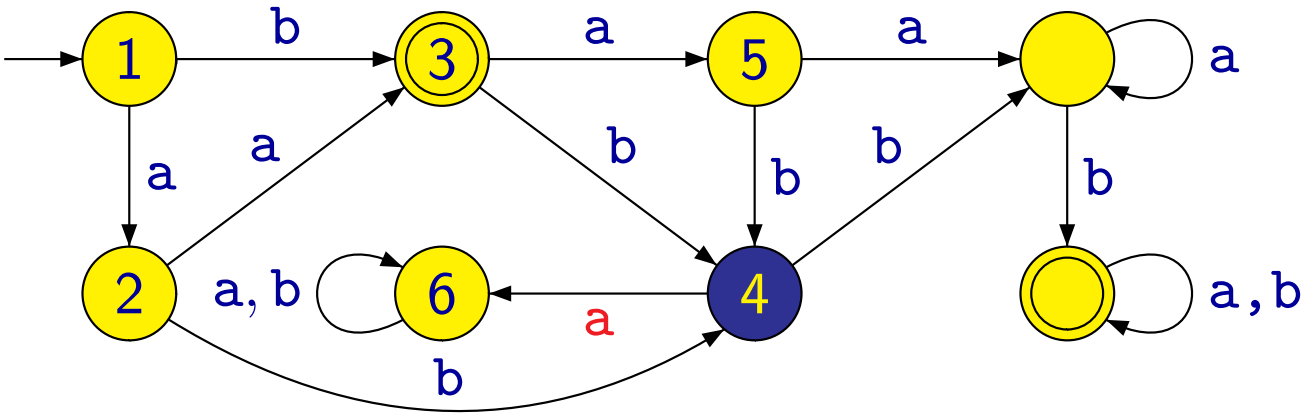
Normovaný tvar automatu



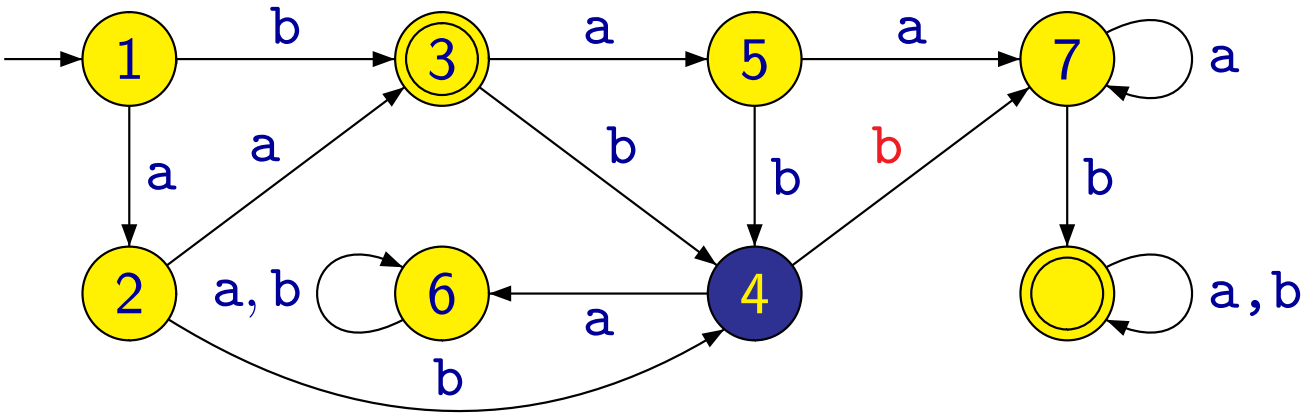
Normovaný tvar automatu



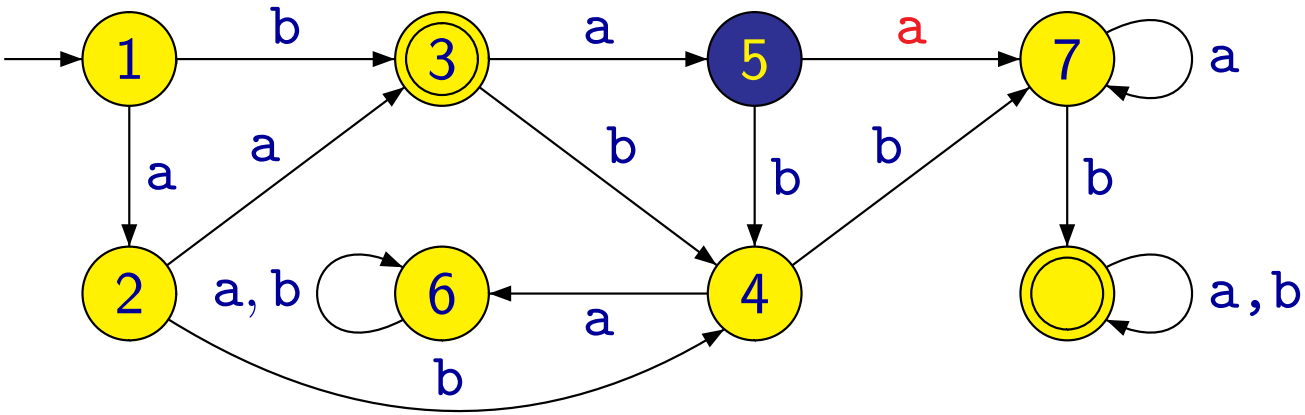
Normovaný tvar automatu



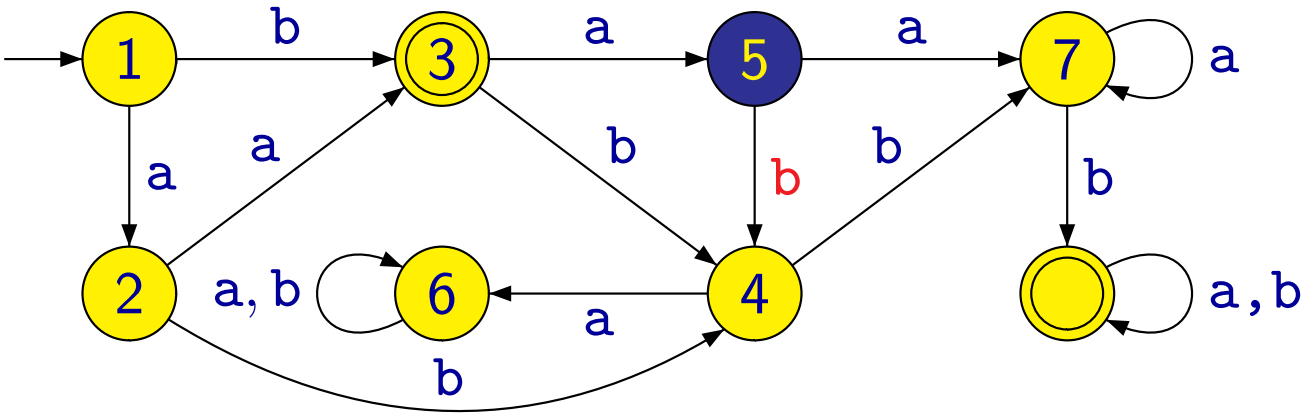
Normovaný tvar automatu



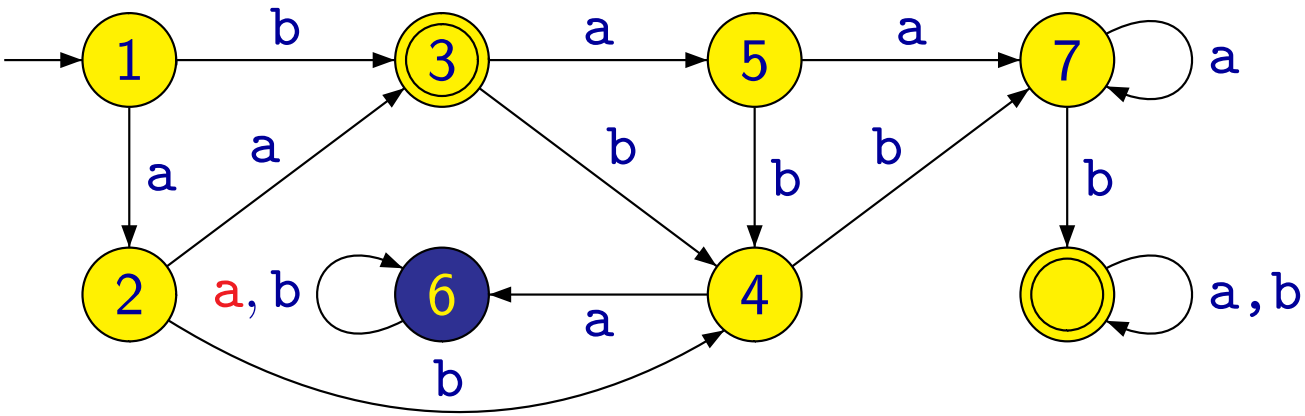
Normovaný tvar automatu



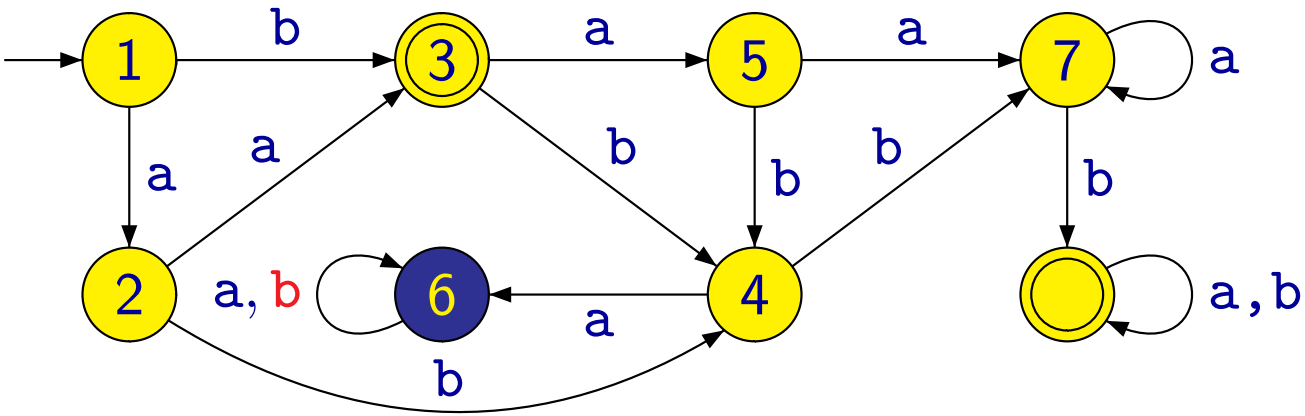
Normovaný tvar automatu



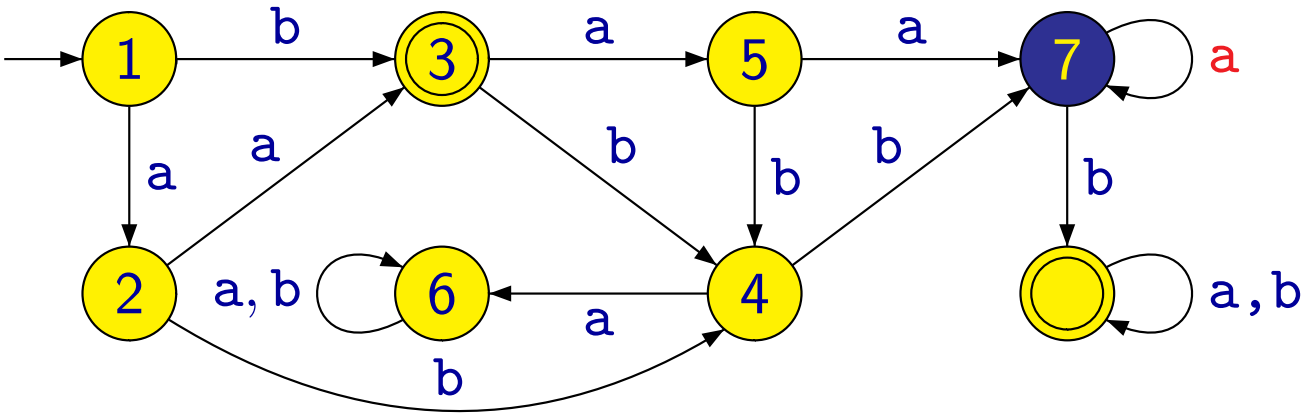
Normovaný tvar automatu



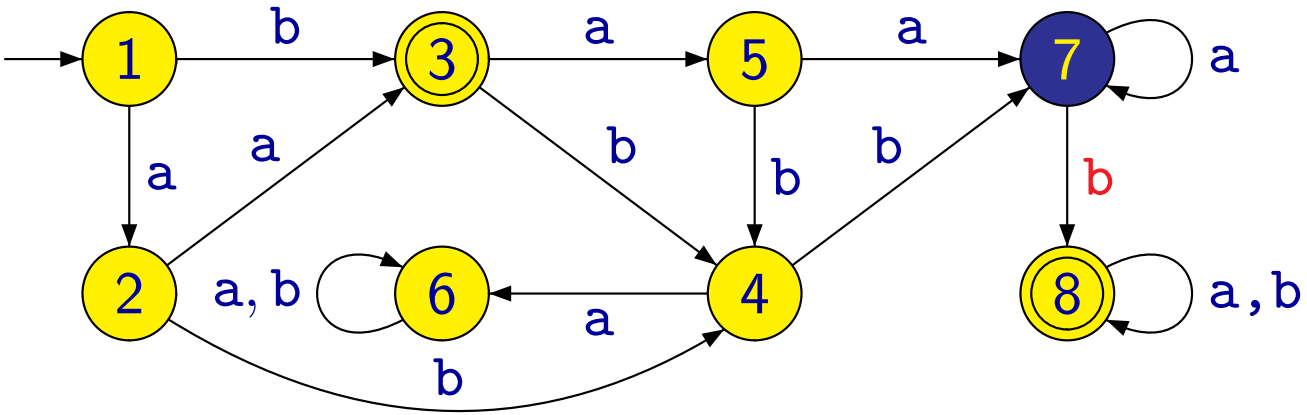
Normovaný tvar automatu



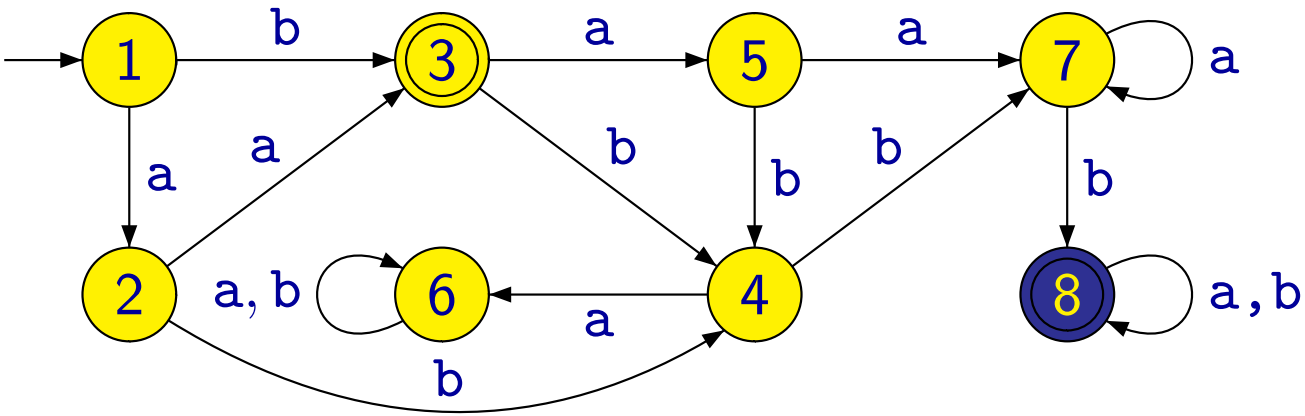
Normovaný tvar automatu



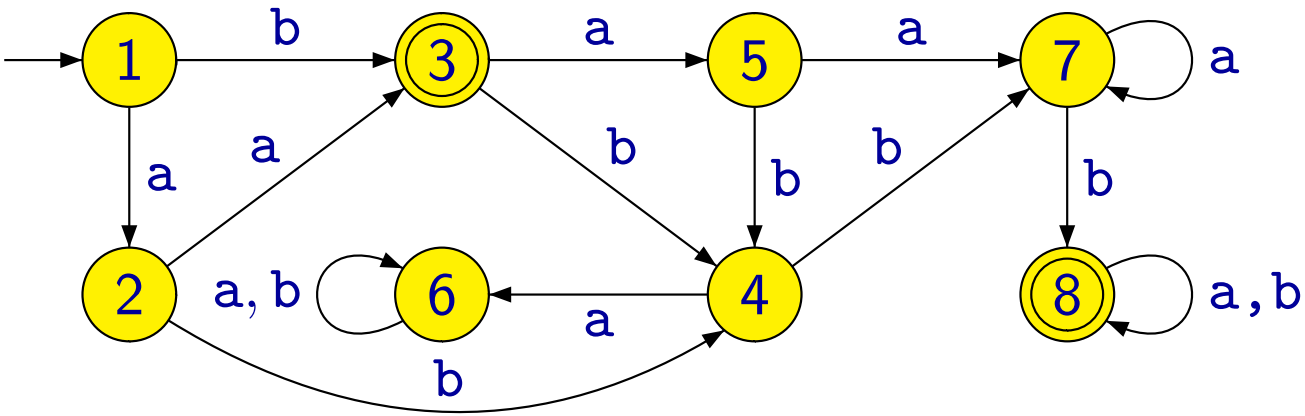
Normovaný tvar automatu



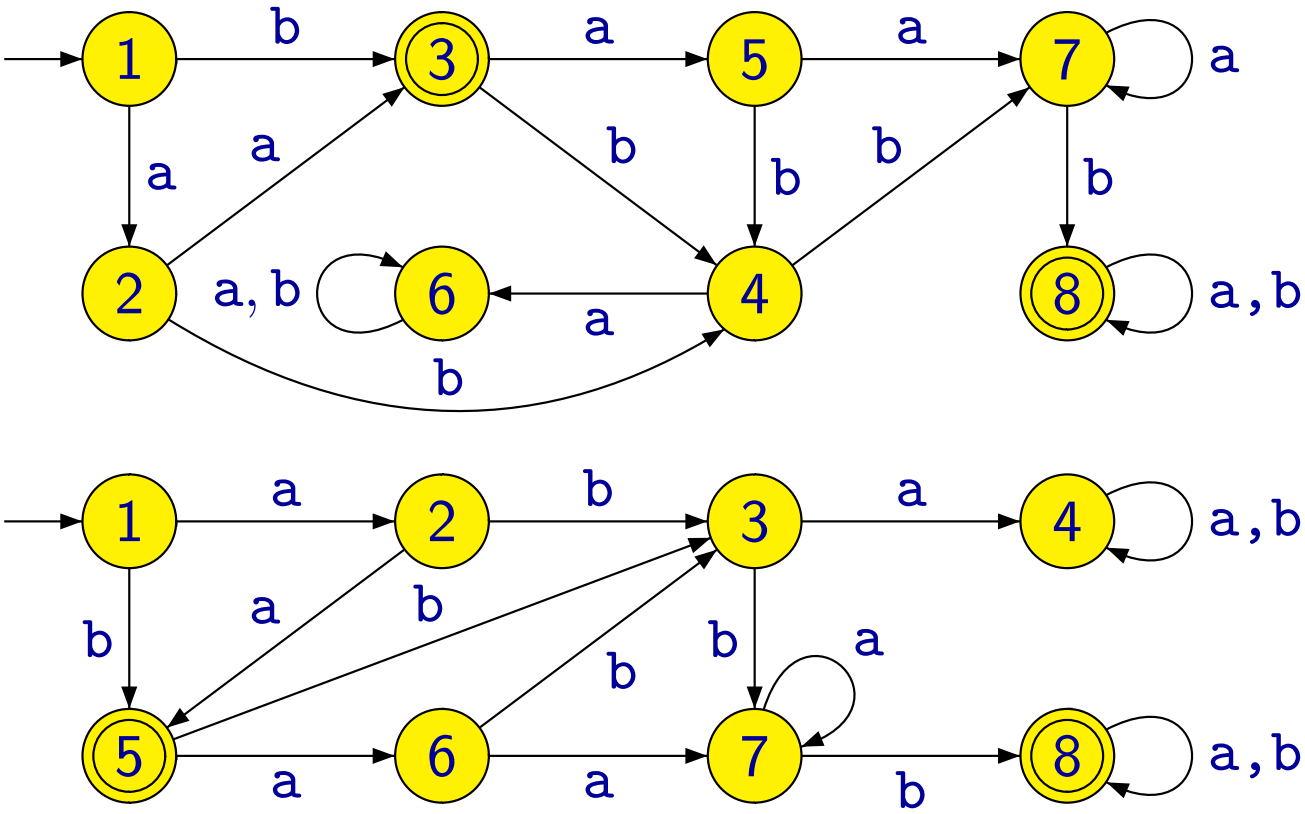
Normovaný tvar automatu



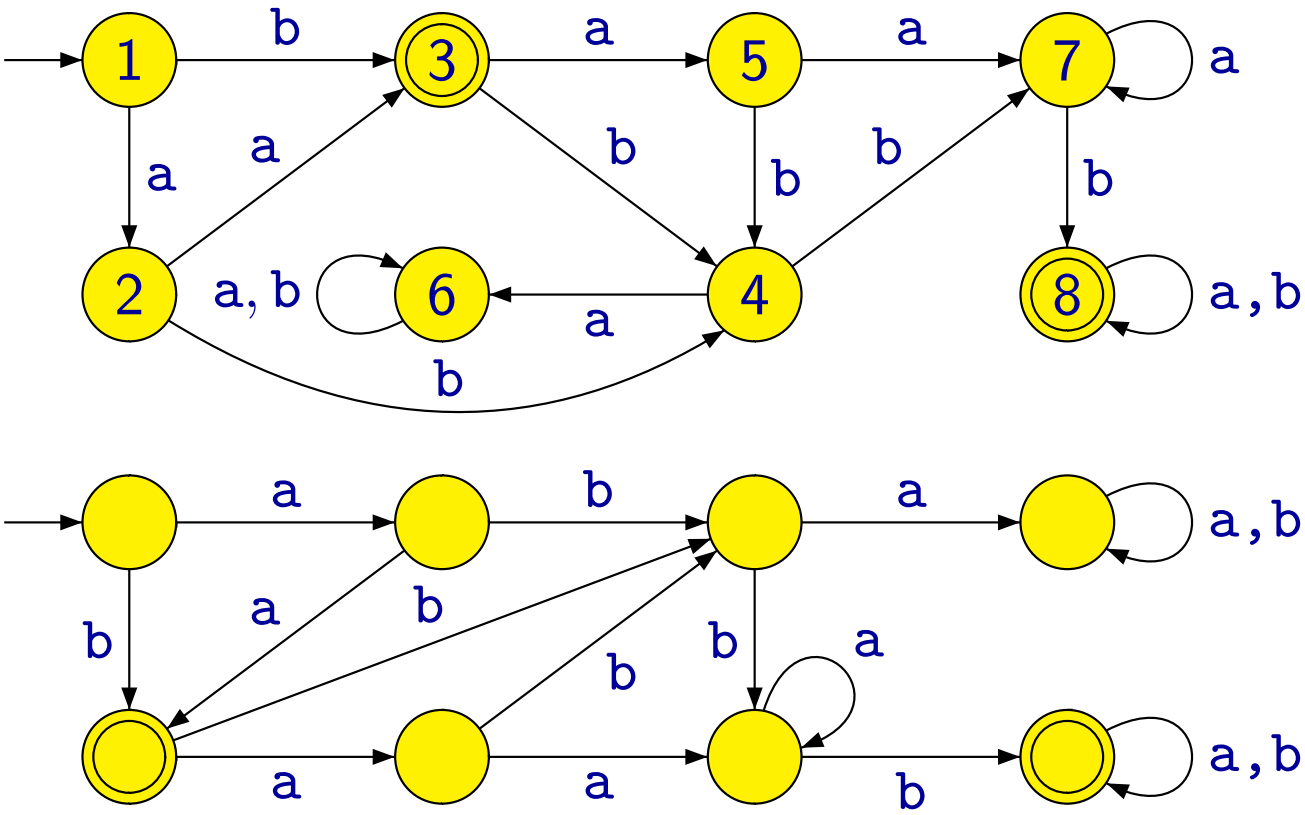
Normovaný tvar automatu



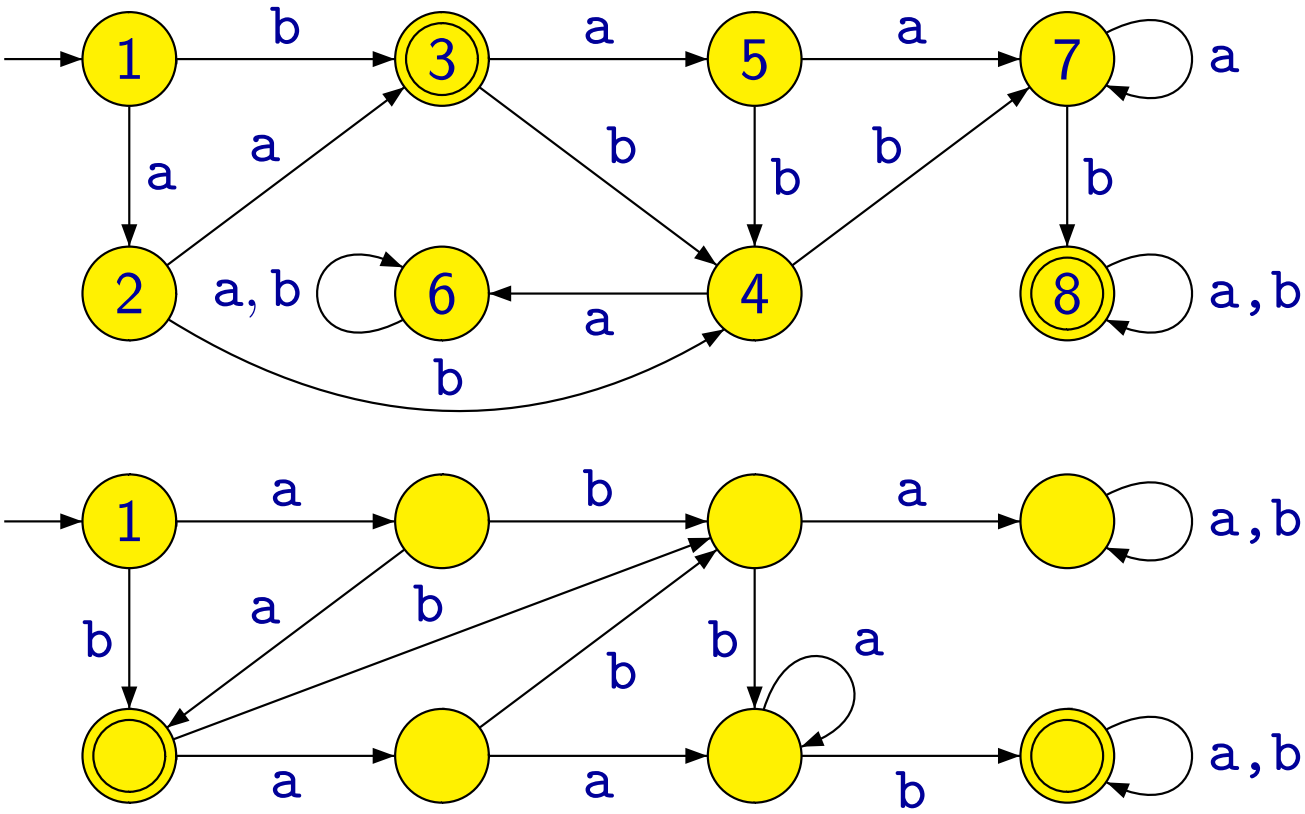
Normovaný tvar automatu



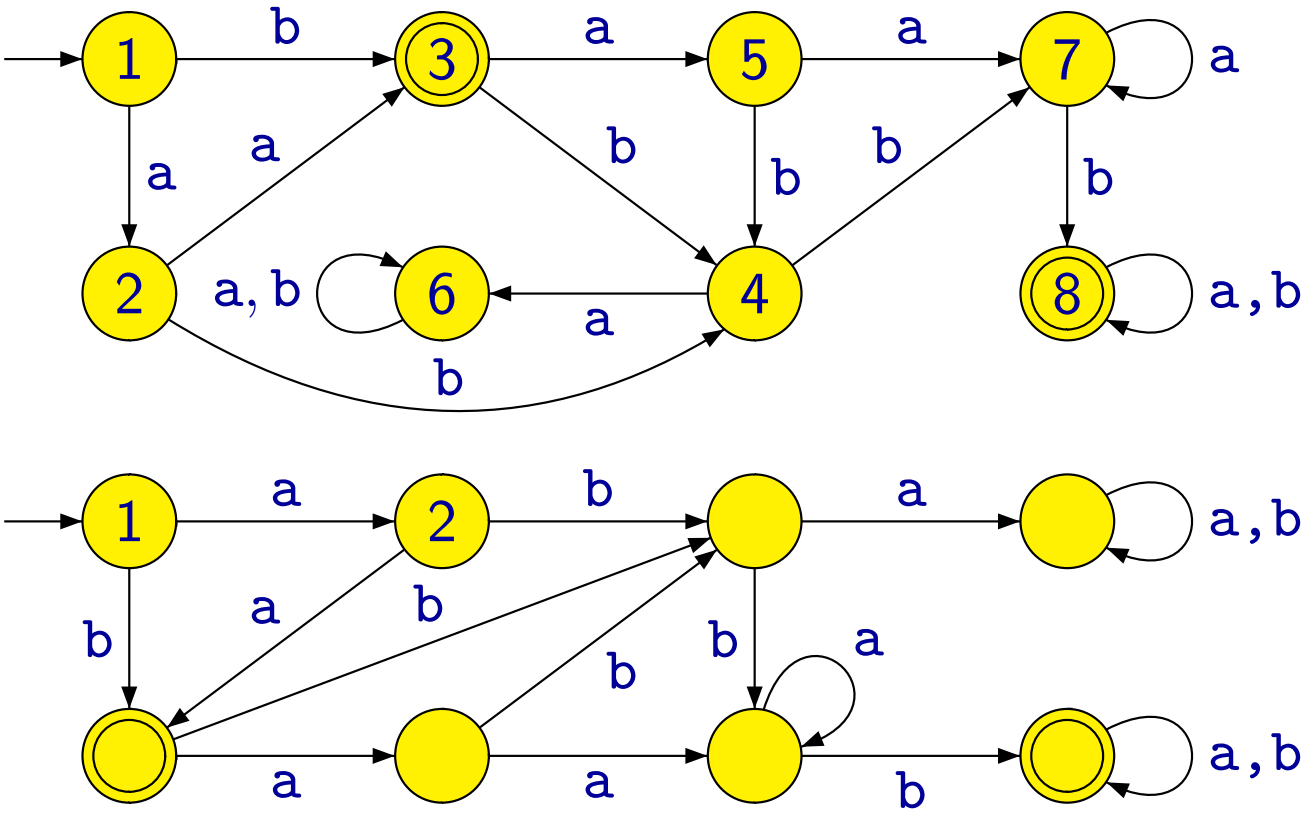
Normovaný tvar automatu



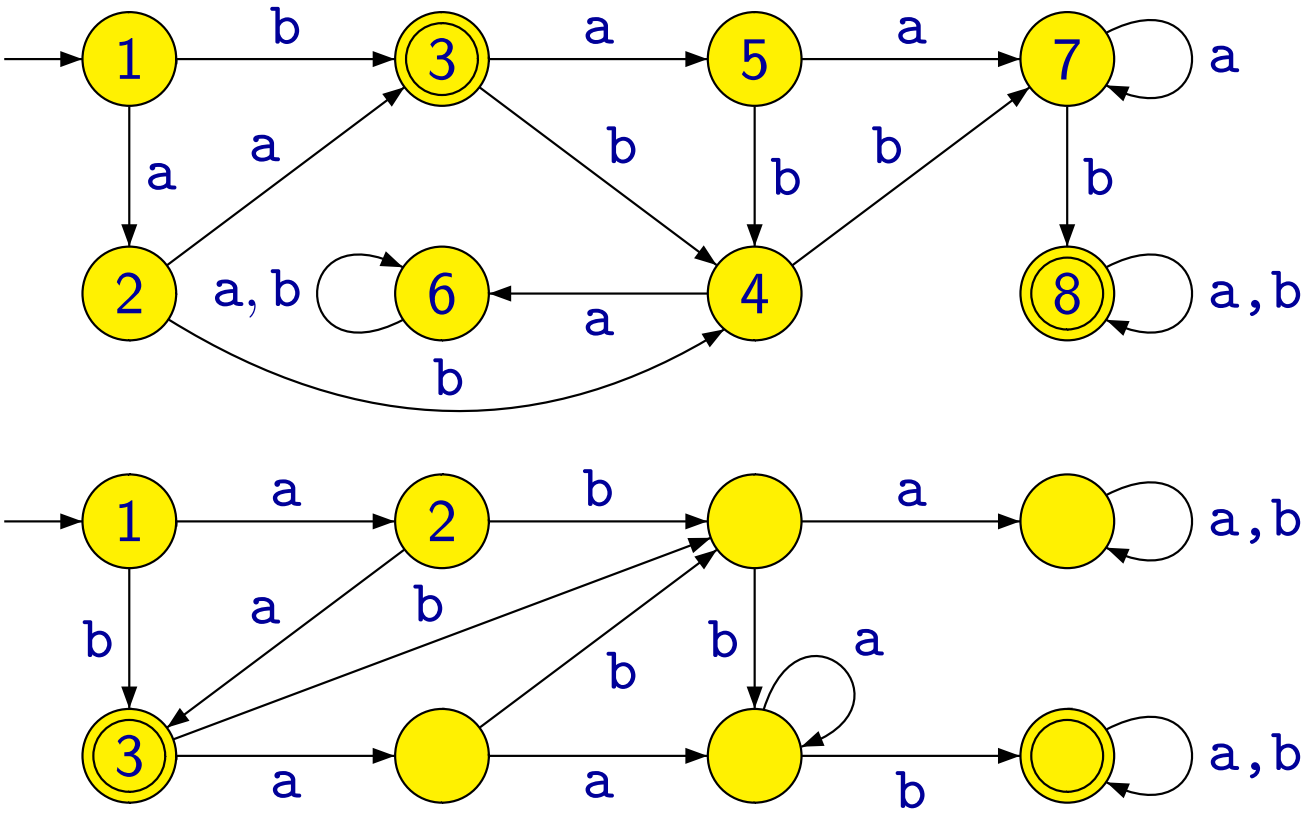
Normovaný tvar automatu



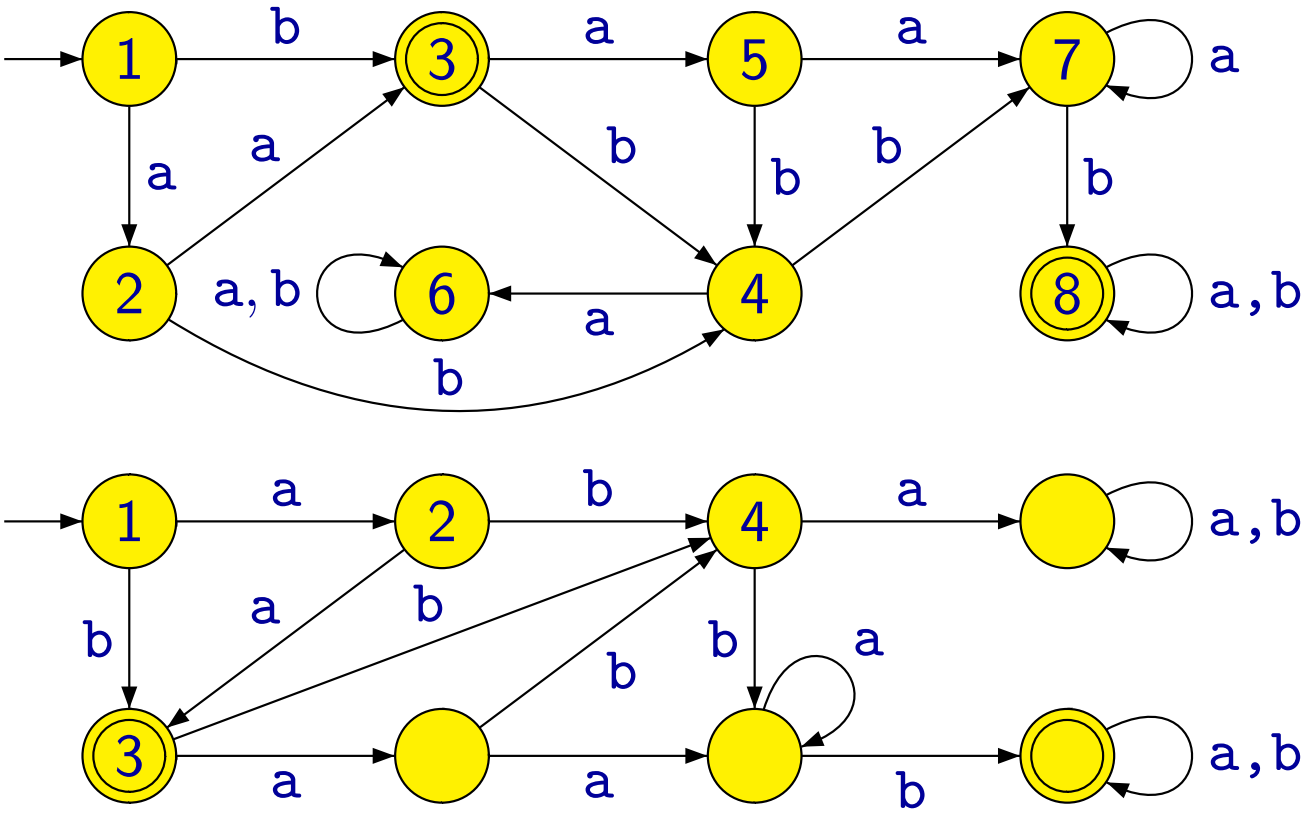
Normovaný tvar automatu



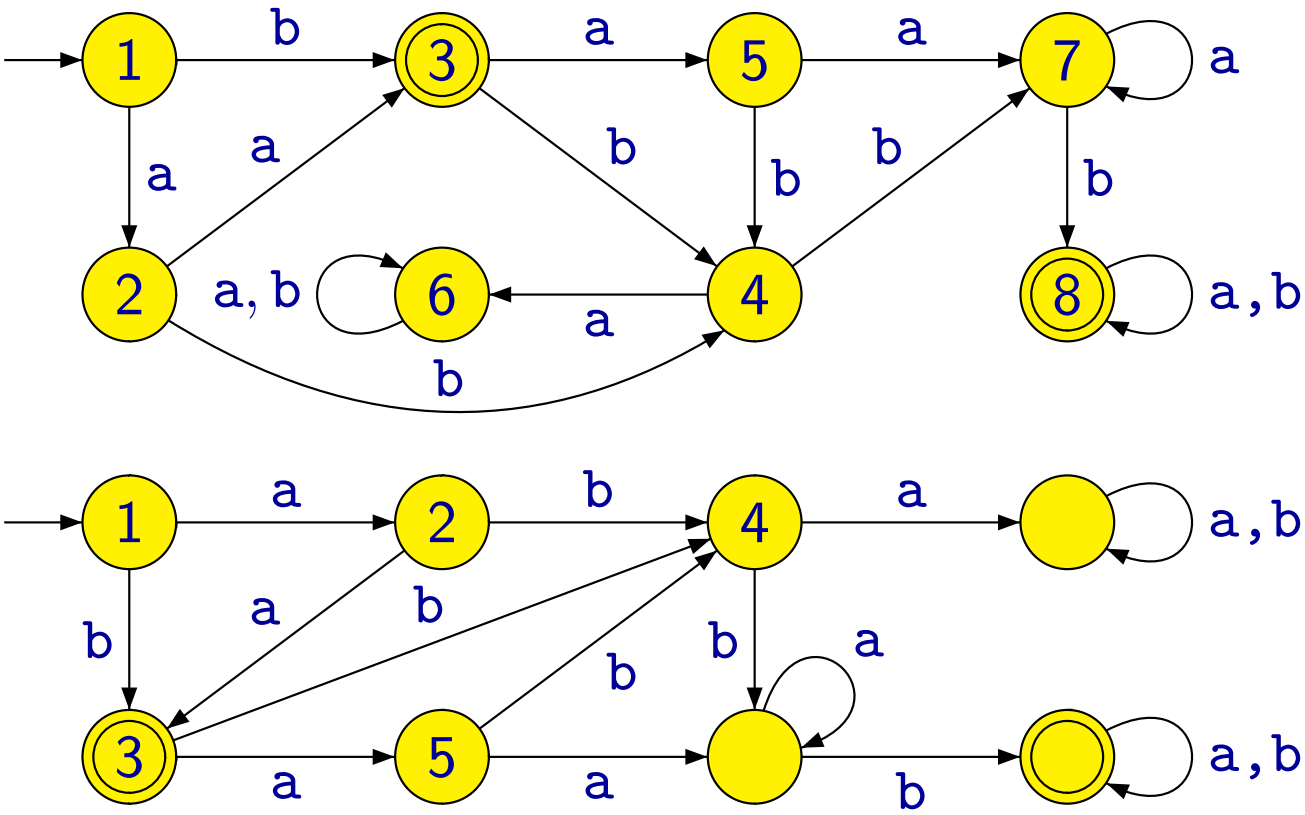
Normovaný tvar automatu



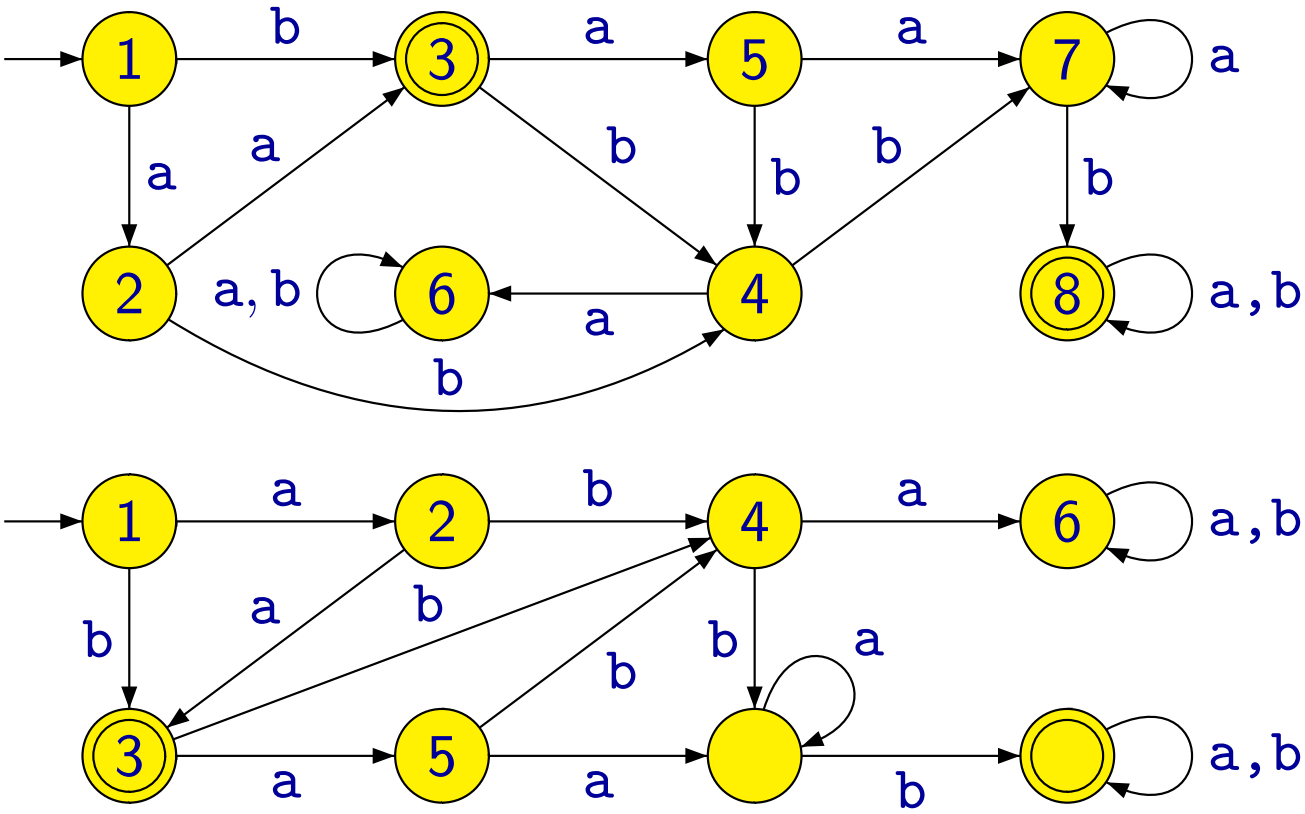
Normovaný tvar automatu



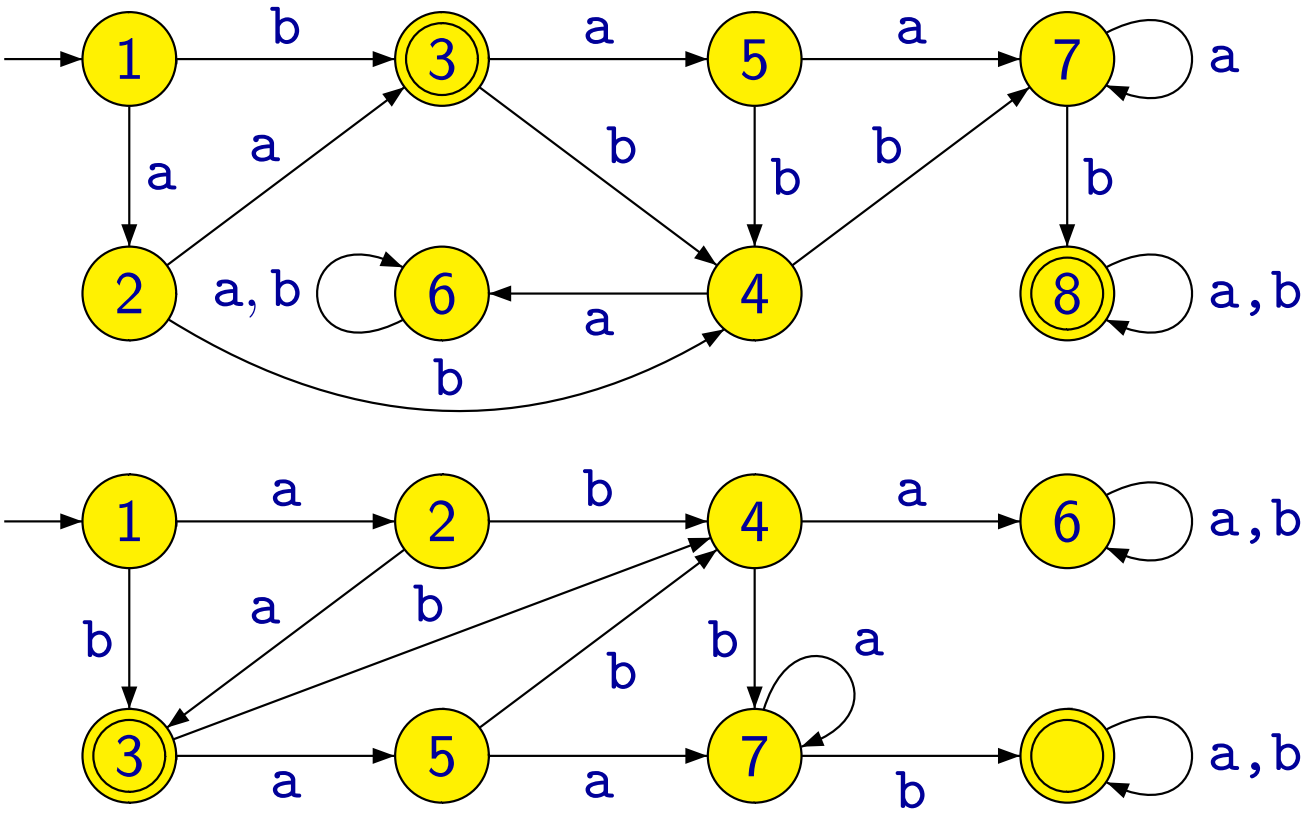
Normovaný tvar automatu



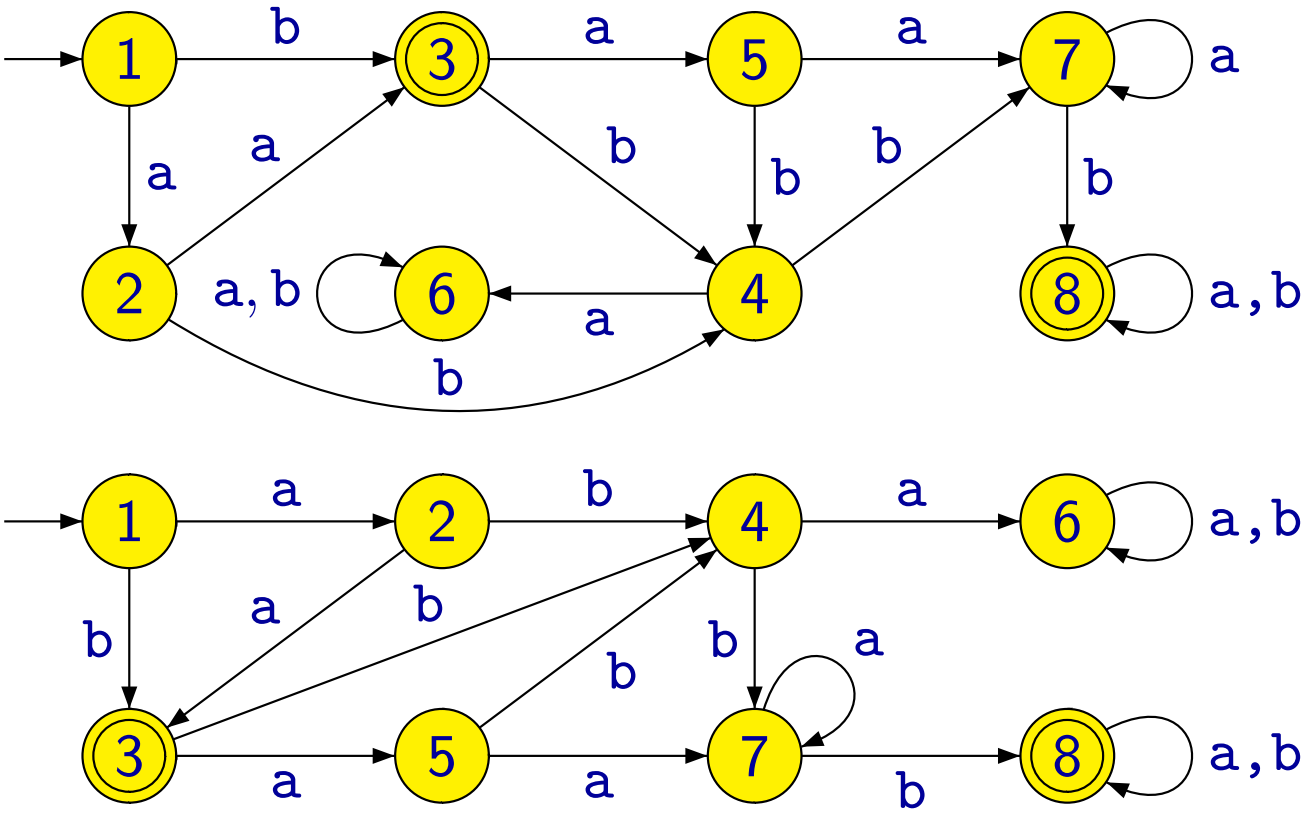
Normovaný tvar automatu



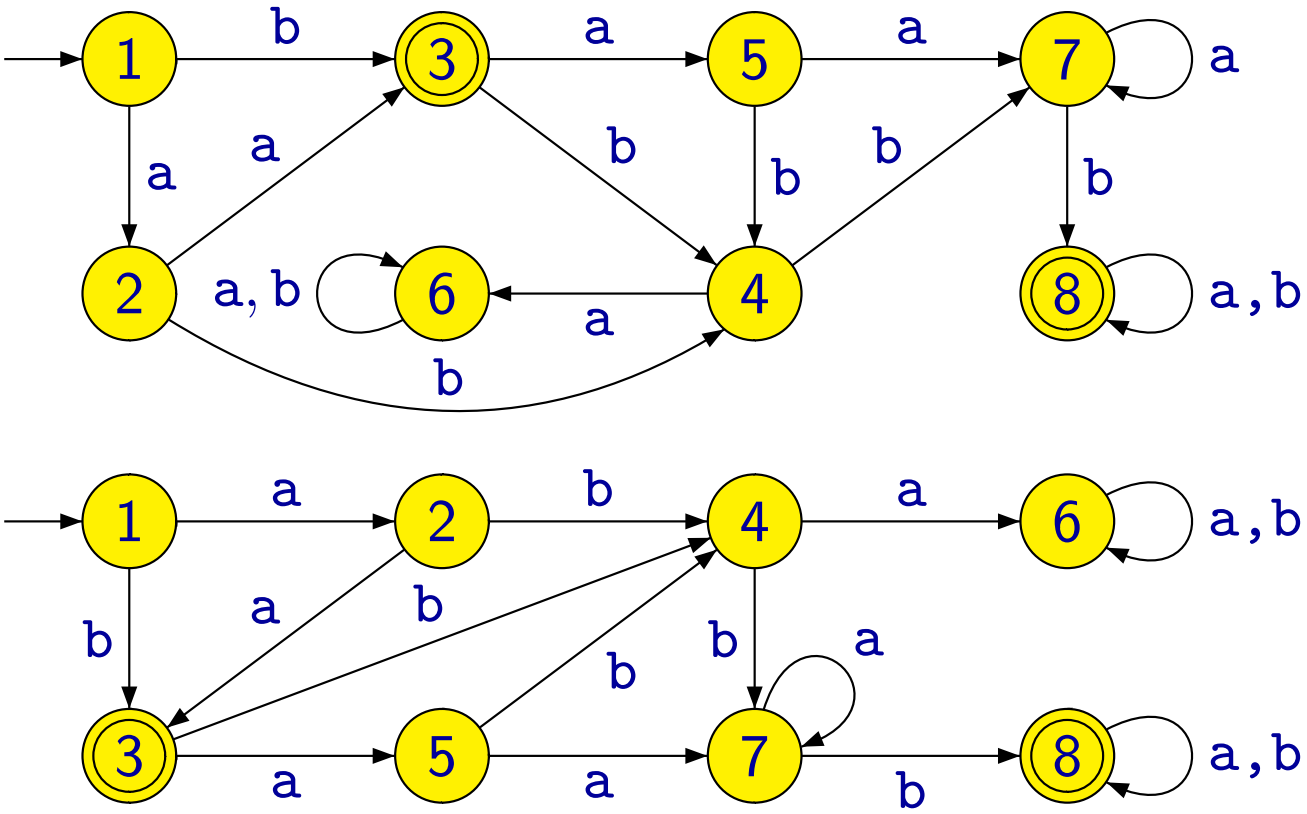
Normovaný tvar automatu



Normovaný tvar automatu

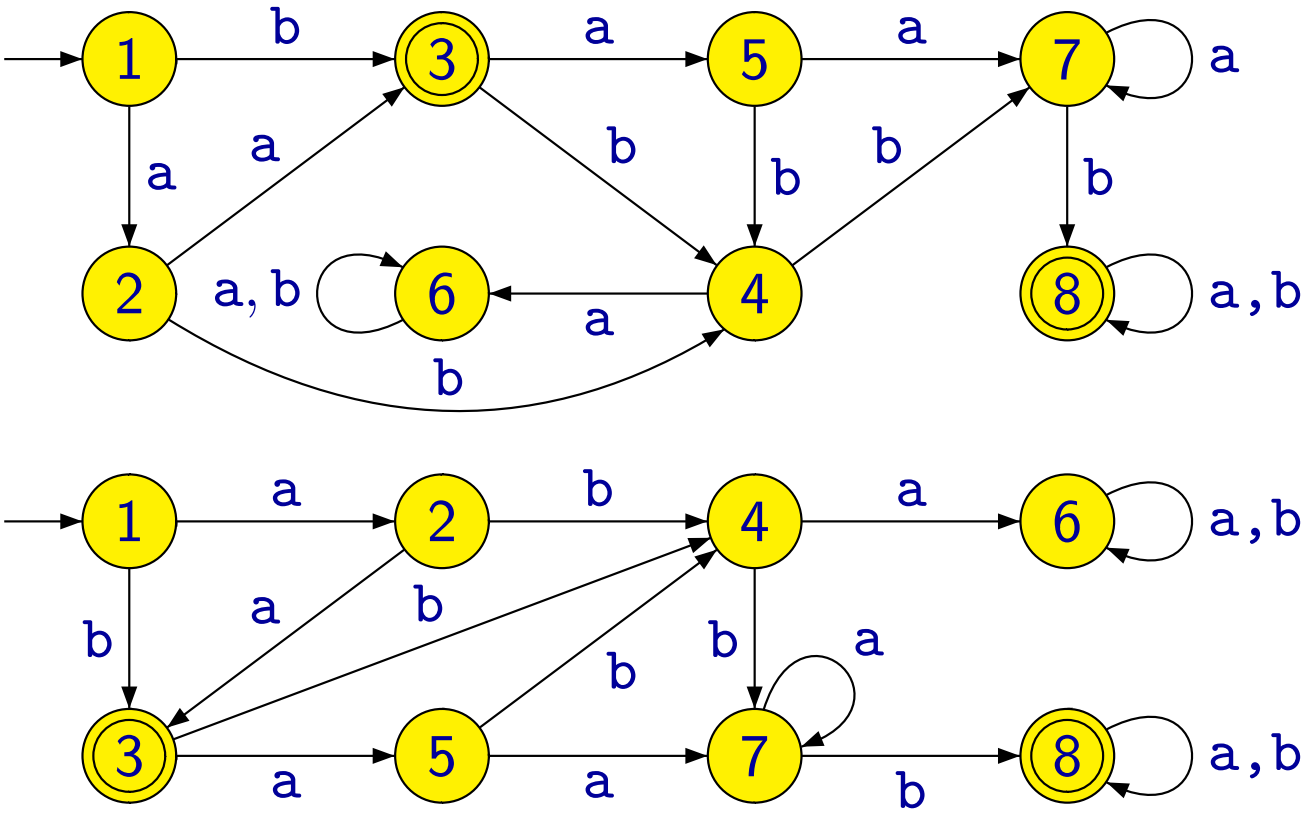


Normovaný tvar automatu



	a	b
$\rightarrow$ 1	2	3
2	3	4
$\leftarrow$ 3	5	4
4	6	7
5	7	4
6	6	6
7	7	8
$\leftarrow$ 8	8	8

Normovaný tvar automatu



	a	b
$\rightarrow 1$	2	3
2	3	4
$\leftarrow 3$	5	4
4	6	7
5	7	4
6	6	6
7	7	8
$\leftarrow 8$	8	8

- Nyní už je přechodová funkce stejná pro oba automaty, stejné jsou i koncové a počáteční stav
- Automaty jsou tedy nejen ekvivalentní, ale dokonce stejné
- Říkáme, že automaty jsou v normovaném tvaru

Normovaný tvar automatu

Definice

Mějme abecedu Σ spolu s úplný ostrým uspořádáním $<$ nad znaky abecedy. Pro slova nad abecedou Σ definujeme úplné ostré uspořádání $<_L$ následovně (pro všechna možná slova $u, v \in \Sigma^*$):

- Pokud $|u| < |v|$, potom $u <_L v$
- Pokud $|u| = |v|$ a u je lexikograficky menší než v, potom $u <_L v$

Normovaný tvar automatu

Definice

Mějme abecedu Σ spolu s úplný ostrým uspořádáním $<$ nad znaky abecedy. Pro slova nad abecedou Σ definujeme úplné ostré uspořádání $<_L$ následovně (pro všechna možná slova $u, v \in \Sigma^*$):

- Pokud $|u| < |v|$, potom $u <_L v$
- Pokud $|u| = |v|$ a u je lexikograficky menší než v, potom $u <_L v$

Příklad: Nad abecedou $\Sigma = \{a, b\}$ uvažujeme běžné uspořádání $a < b$. Potom platí $\varepsilon <_L a <_L b <_L aa <_L bb <_L aba <_L baa <_L aaaa <_L bbbbb$.

Normovaný tvar automatu

Definice

Mějme abecedu Σ spolu s úplný ostrým uspořádáním $<$ nad znaky abecedy. Pro slova nad abecedou Σ definujeme úplné ostré uspořádání $<_L$ následovně (pro všechna možná slova $u, v \in \Sigma^*$):

- Pokud $|u| < |v|$, potom $u <_L v$
- Pokud $|u| = |v|$ a u je lexikograficky menší než v, potom $u <_L v$

Příklad: Nad abecedou $\Sigma = \{a, b\}$ uvažujeme běžné uspořádání $a < b$. Potom platí $\varepsilon <_L a <_L b <_L aa <_L bb <_L aba <_L baa <_L aaaa <_L bbbbb$.
Příklad: Nad abecedou $\Sigma = \{1, 2, 3\}$ uvažujeme běžné uspořádání $1 < 2 < 3$. Potom platí $\varepsilon <_L 1 <_L 3 <_L 31 <_L 33 <_L 111 <_L 321$.

Normovaný tvar automatu

Definice

Mějme abecedu Σ spolu s úplný ostrým uspořádáním $<$ nad znaky abecedy. Pro slova nad abecedou Σ definujeme úplné ostré uspořádání $<_L$ následovně (pro všechna možná slova $u, v \in \Sigma^*$):

- Pokud $|u| < |v|$, potom $u <_L v$
- Pokud $|u| = |v|$ a u je lexikograficky menší než v, potom $u <_L v$

Příklad: Nad abecedou $\Sigma = \{a, b\}$ uvažujeme běžné uspořádání $a < b$. Potom platí $\varepsilon <_L a <_L b <_L aa <_L bb <_L aba <_L baa <_L aaaa <_L bbbbb$.

Příklad: Nad abecedou $\Sigma = \{1, 2, 3\}$ uvažujeme běžné uspořádání $1 < 2 < 3$. Potom platí $\varepsilon <_L 1 <_L 3 <_L 31 <_L 33 <_L 111 <_L 321$.

Příklad: Nad abecedou $\Sigma = \{0, 1, 2\}$ uvažujeme běžné uspořádání $0 < 1 < 2$. Potom platí $\varepsilon <_L 1 <_L 2 <_L 01 <_L 03 <_L 111 <_L 0021$.

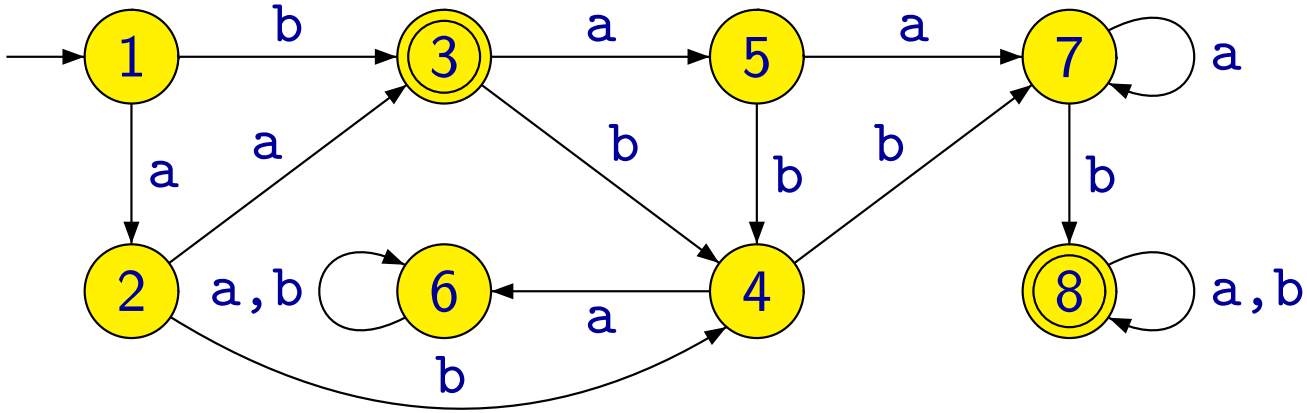
Normovaný tvar automatu

Definice

Mějme deterministický konečný automat $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ bez nedosažitelných stavů a uspořádání prvků abecedy Σ (které indukuje uspořádání $<_L$ na Σ^*).
Označme pro každý stav $i \in \{1, 2, \ldots, n\}$ symbolem u_i nejmenší slovo podle uspořádání $<_L$ takové, že pro něj platí $\delta(1, u_i) = i$.
Potom řekneme, že $\mathcal{A}$ je v normovaném tvaru, pokud

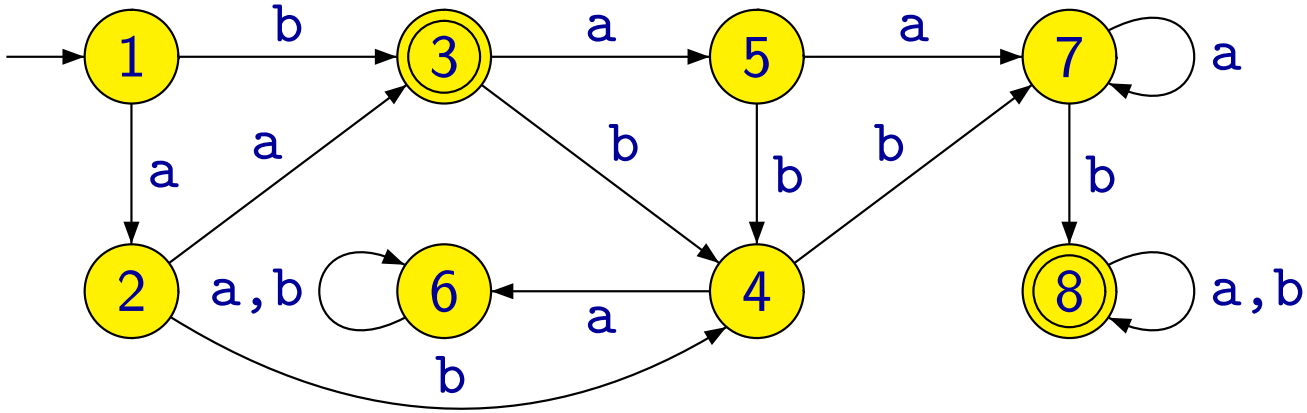
- $Q = \{1, 2, 3, \ldots, n\}$ pro nějaké $n \geq 1$
- 1 je počáteční stav
- Pro každou dvojici stavů $i, j \in \{1, 2, \ldots, n\}$ takovou, že $i < j$, platí $u_i <_L u_j$.

Normovaný tvar automatu



- Do stavu 4 se dostaneme slovy ab, bb, aab, bab tedy $u_4 = ab$
- Do stavu 5 se dostaneme slovy ba, aaa tedy $u_5 = ba$
- $ab <_L ba$ proto jsou stavy $4, 5$ označeny v tomto pořadí

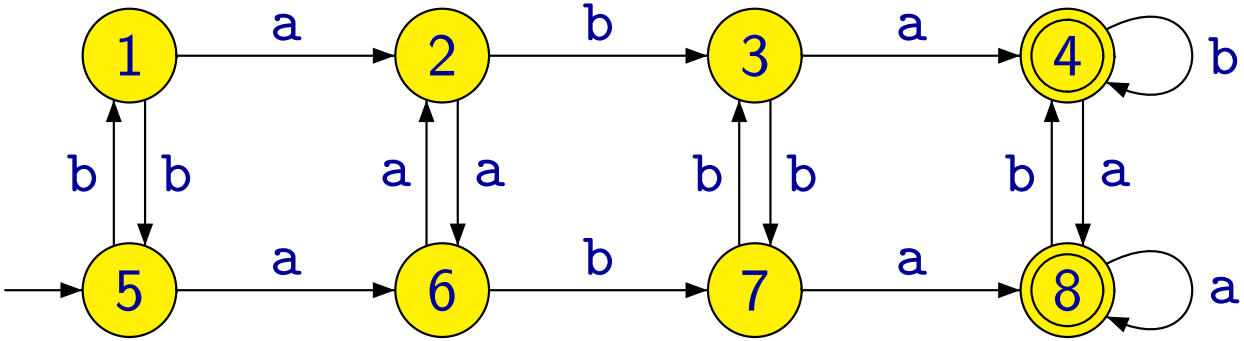
Normovaný tvar automatu



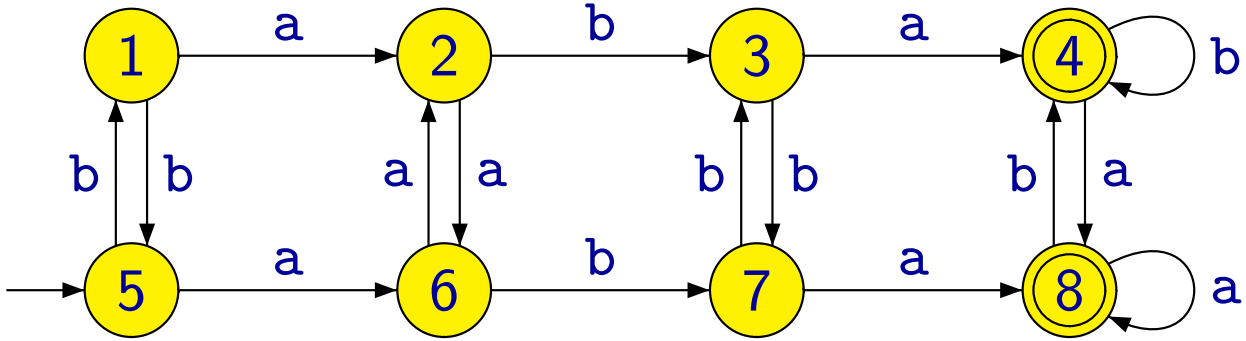
- Do stavu 4 se dostaneme slovy ab, bb, aab, bab tedy $u_4 = ab$
- Do stavu 5 se dostaneme slovy ba, aaa tedy $u_5 = ba$
- $ab <_L ba$ proto jsou stavy $4, 5$ označeny v tomto pořadí
- Pro ostatní stavy jsou nejmenší slova následující:

$$\begin{aligned} u_1 &= \varepsilon & u_5 &= ba \\ u_2 &= a & u_6 &= aba \\ u_3 &= b & u_7 &= abb \\ u_4 &= ab & u_8 &= abbb \end{aligned}$$

Minimalizace konečného automatu

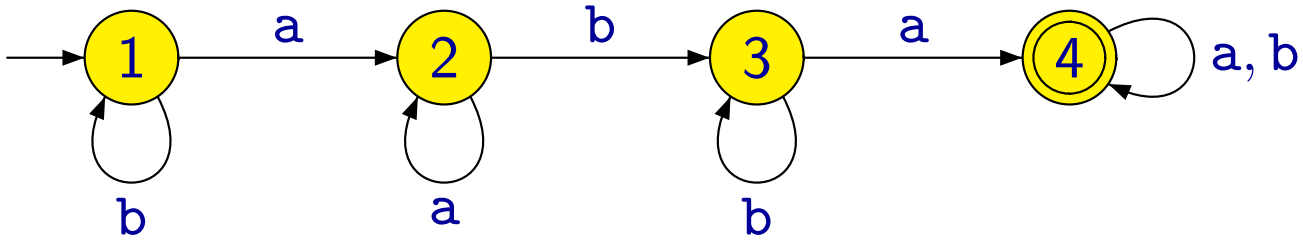


Minimalizace konečného automatu



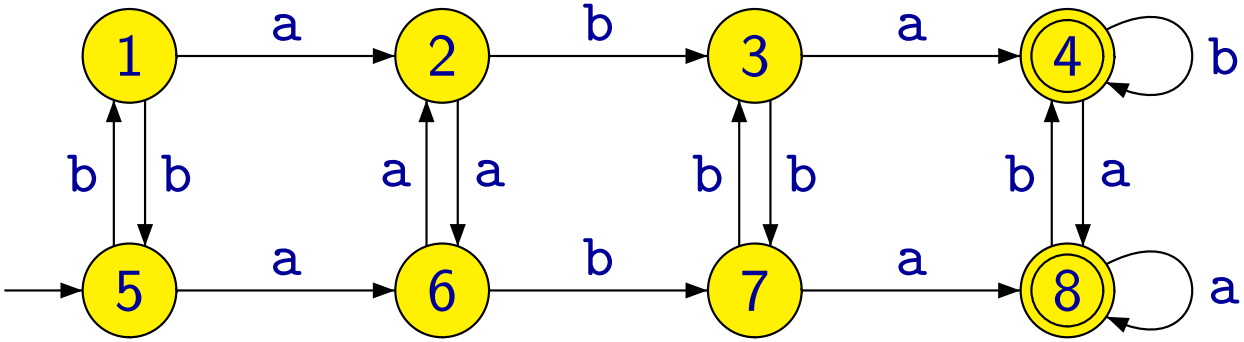
- Automat přijímá taková slova, z nichž by vypuštěním některých znaků zůstalo *aba*
- Existuje automat s méně stavy, který přijímá stejný jazyk?

Minimalizace konečného automatu



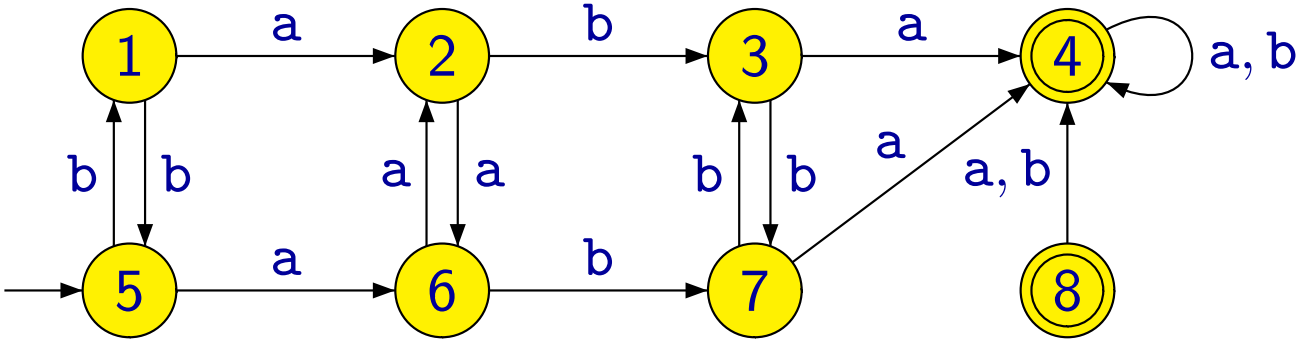
- Automat přijímá taková slova, z nichž by vypuštěním některých znaků zůstalo *aba*
- Existuje automat s méně stavy, který přijímá stejný jazyk?
- Existuje nějaký algoritmus, který by našel ekvivalentní automat s nejmenším možným počtem stavů?

Minimalizace konečného automatu



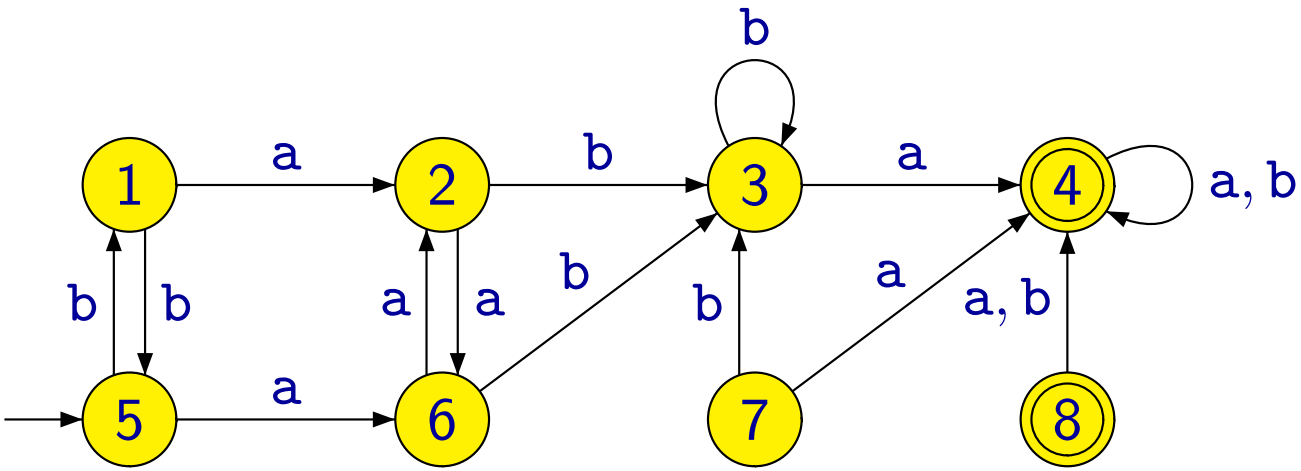
- Když se automat při běhu dostane do stavu 4, příjme už jistě dané slovo, ať už bude pokračovat jakkoliv
- Když se automat při běhu dostane do stavu 8, příjme také už jistě dané slovo
- Je tedy jedno, jestli jde automat některým přechodem do 4 nebo 8

Minimalizace konečného automatu



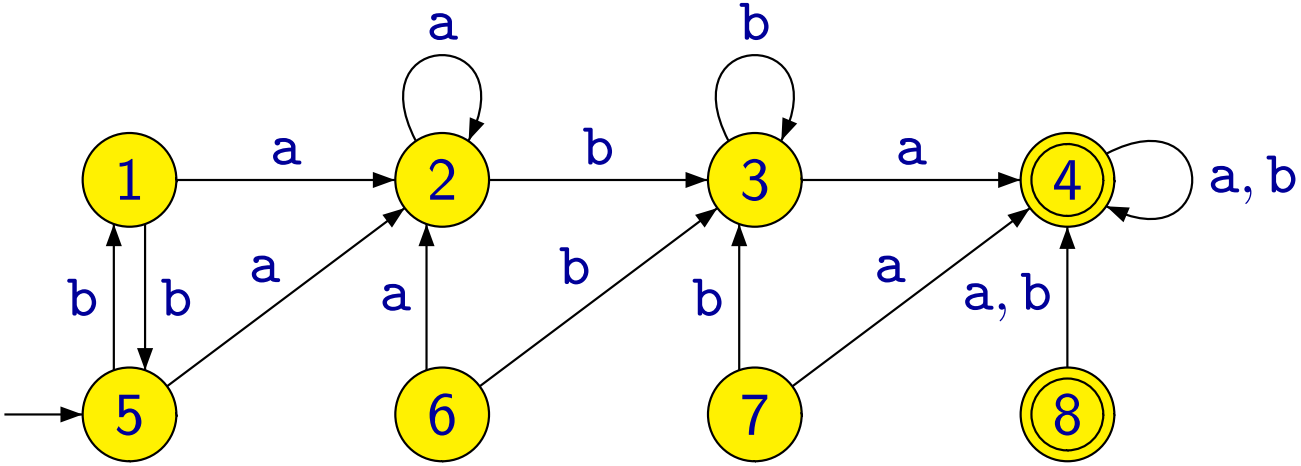
- Když se automat při běhu dostane do stavu 3, příjme každé slovo, které bude v pokračování obsahovat alespoň jedno *a*
- Když se automat při běhu dostane do stavu 7, příjme každé slovo, které bude v pokračování obsahovat alespoň jedno *a*
- Je tedy jedno, jestli jde automat některým přechodem do 3 nebo 7

Minimalizace konečného automatu



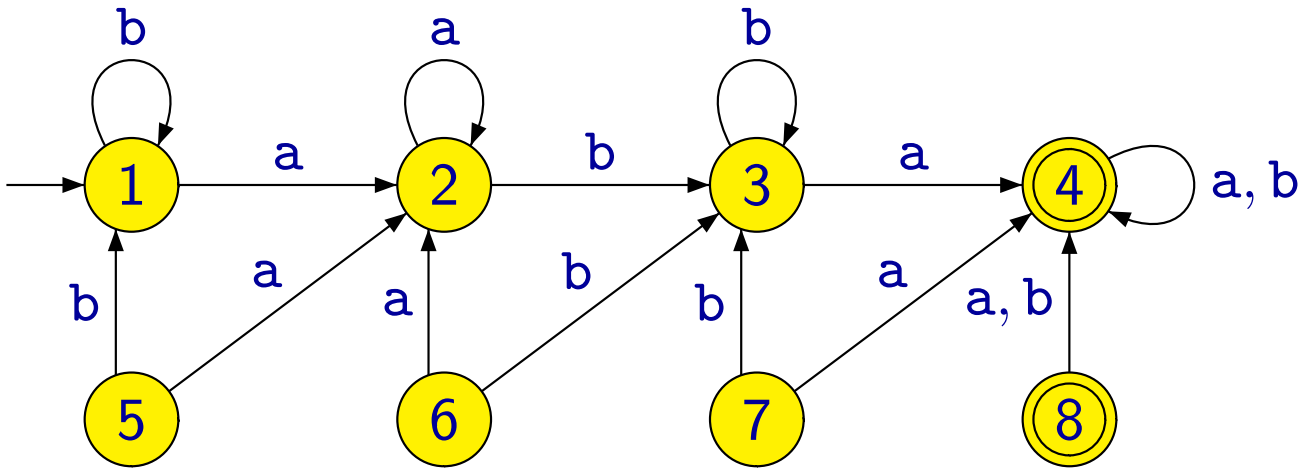
- Když se automat při běhu dostane do stavu 2, příjme každé slovo, které bude v pokračování obsahovat alespoň jedno *b* a jedno *a* v tomto pořadí
- Když se automat při běhu dostane do stavu 6, příjme stejná slova
- Je tedy jedno, jestli jde automat některým přechodem do 2 nebo 6

Minimalizace konečného automatu



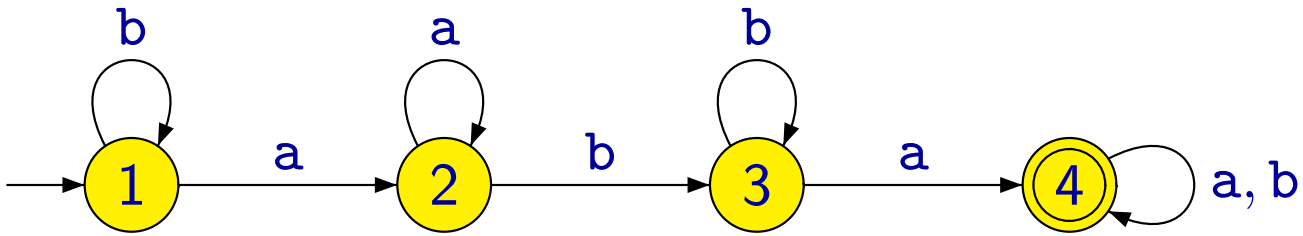
- Ve stavu 1 i 5 automat příjme jakékoliv slovo, které patří do jazyka takových slov, z nichž by vypuštěním některých znaků zůstalo *aba*
- Je tedy jedno, jestli jde automat některým přechodem do 1 nebo 5, a je jedno, ve kterém z nich začne

Minimalizace konečného automatu



- Stavy $5, 6, 7, 8$ jsou teď už nedosažitelné a můžeme je odstranit

Minimalizace konečného automatu



Minimalizace konečného automatu

Definice

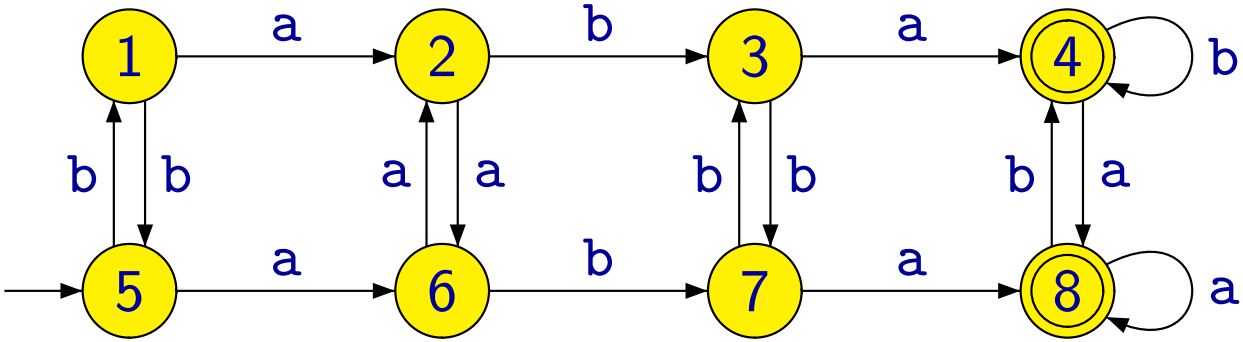
Pro každý stav q automatu $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ definujeme $L(q) = L(\mathcal{A}_q)$, kde $\mathcal{A}_q = (Q, \Sigma, \delta, q, F)$

Definice

Stavy q_1, q_2 automatu $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ nazýváme **jazykově ekvivalentní** nebo zkráceně **ekvivalentní**, jestliže $L(q_1) = L(q_2)$.

- Jsou-li dva stavy q_1, q_2 automatu ekvivalentní, můžeme jeden vypustit a všechny šipky do něj směřující přesměrovat do druhého. Pokud byl vypouštěný stav počáteční, bude počáteční ten druhý.
- Dvojice vzájemně ekvivalentních stavů můžeme hledat rychlým (polynomiálním) algoritmem

Minimalizace konečného automatu



	a	b
1	2	5
2	6	3
3	4	7
← 4	8	4
→ 5	6	1
6	2	7
7	8	3
← 8	8	4

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
$\leftarrow$ 4	8	4
$\rightarrow$ 5	6	1
6	2	7
7	8	3
$\leftarrow$ 8	8	4

- Rozdělíme stavy do dvou skupin na přijímající a nepřijímající. Je jisté, že přijímající stav nikdy není ekvivalentní s nepřijímajícím.

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
$\leftarrow$ 4	8	4
$\rightarrow$ 5	6	1
6	2	7
7	8	3
$\leftarrow$ 8	8	4

		a	b
I	1	2	5
	2	6	3
	3	4	7
	$\rightarrow$ 5	6	1
	6	2	7
	7	8	3
II	$\leftarrow$ 4	8	4
	$\leftarrow$ 8	8	4

- Rozdělíme stavy do dvou skupin na přijímající a nepřijímající. Je jisté, že přijímající stav nikdy není ekvivalentní s nepřijímajícím.
- Do tabulky si, místo přechodů do konkrétních stavů, vyznačíme skupinu, do které přecházíme.

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
← 4	8	4
→ 5	6	1
6	2	7
7	8	3
← 8	8	4

		a	b
I	1	2	5
	2	6	3
	3	4	7
	→ 5	6	1
	6	2	7
	7	8	3
II	← 4	8	4
	← 8	8	4

		a	b
I	1	I	I
	2	I	I
	3	II	I
	→ 5	I	I
	6	I	I
	7	II	I
II	← 4	II	II
	← 8	II	II

- Rozdělíme stavy do dvou skupin na přijímající a nepřijímající. Je jisté, že přijímající stav nikdy není ekvivalentní s nepřijímajícím.
- Do tabulky si, místo přechodů do konkrétních stavů, vyznačíme skupinu, do které přecházíme.

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
← 4	8	4
→ 5	6	1
6	2	7
7	8	3
← 8	8	4

		a	b
I	1	I	I
	2	I	I
	3	II	I
	→ 5	I	I
	6	I	I
	7	II	I
II	← 4	II	II
	← 8	II	II

- Pokud se liší řádky ve stejné skupině, skupinu rozdělíme na menší, kde se lišit nebudou
- Např. 3 se od 1 liší, protože z 1 přejde a-čkem do skupiny, která následně nepřijímá prázdné slovo, z 3 přejde a-čkem do skupiny, která následně přijímá prázdné slovo.

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
← 4	8	4
→ 5	6	1
6	2	7
7	8	3
← 8	8	4

		a	b
I	1	I	I
	2	I	I
	3	II	I
	→ 5	I	I
	6	I	I
	7	II	I
II	← 4	II	II
	← 8	II	II

		a	b
I	1	I	I
	2	I	I
	→ 5	I	I
	6	I	I
II	3	II	I
	7	II	I
III	← 4	II	II
	← 8	II	II

- Pokud se liší řádky ve stejné skupině, skupinu rozdělíme na menší, kde se lišit nebudou
- Např. 3 se od 1 liší, protože z 1 přejde a-čkem do skupiny, která následně nepřijímá prázdné slovo, z 3 přejde a-čkem do skupiny, která následně přijímá prázdné slovo.
- Musíme znovu vyplnit tabulku, protože se změnily skupiny

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
$\leftarrow$ 4	8	4
$\rightarrow$ 5	6	1
6	2	7
7	8	3
$\leftarrow$ 8	8	4

		a	b
I	1	I	I
	2	I	II
	$\rightarrow$ 5	I	I
	6	I	II
II	3	III	II
	7	III	II
III	$\leftarrow$ 4	III	III
	$\leftarrow$ 8	III	III

- Teď se 2 se od 1 liší, protože z 1 přejde b-čkem do jiné skupiny než z 2. Tedy z každého příjme jiná slova začínající b-čkem a nemohou být ekvivalentní.
- Takže zase skupinu I rozdělíme

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
← 4	8	4
→ 5	6	1
6	2	7
7	8	3
← 8	8	4

		a	b
I	1	I	I
	2	I	II
	→ 5	I	I
	6	I	II
II	3	III	II
	7	III	II
III	← 4	III	III
	← 8	III	III

		a	b
I	1	I	I
	→ 5	I	I
II	2	I	II
	6	I	II
III	3	III	II
	7	III	II
IV	← 4	III	III
	← 8	III	III

- Teď se 2 se od 1 liší, protože z 1 přejde b-čkem do jiné skupiny než z 2. Tedy z každého příjme jiná slova začínající b-čkem a nemohou být ekvivalentní.
- Takže zase skupinu I rozdělíme
- Musíme znovu vyplnit tabulku, protože se změnily skupiny

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
$\leftarrow$ 4	8	4
$\rightarrow$ 5	6	1
6	2	7
7	8	3
$\leftarrow$ 8	8	4

		a	b
I	1	II	I
	$\rightarrow$ 5	II	I
II	2	II	III
	6	II	III
III	3	IV	III
	7	IV	III
IV	$\leftarrow$ 4	IV	IV
	$\leftarrow$ 8	IV	IV

- Teď už se žádná skupina nerozdělí, takže algoritmus končí
- Stavy, které jsou v jedné skupině, jsou ekvivalentní. Můžeme nechat kterýkoliv z nich a ostatní odstranit
- Šipky vedoucí do odstraňovaných stavů povedou do toho, který ze skupiny necháme
- Pokud skupina obsahovala počáteční stav, bude ponechaný zástupce počáteční

Minimalizace konečného automatu

	a	b
1	2	5
2	6	3
3	4	7
← 4	8	4
→ 5	6	1
6	2	7
7	8	3
← 8	8	4

		a	b
I	1	II	I
	→ 5	II	I
II	2	II	III
	6	II	III
III	3	IV	III
	7	IV	III
IV	← 4	IV	IV
	← 8	IV	IV

	a	b
→ I	II	I
II	II	III
III	IV	III
← IV	IV	IV

- Teď už se žádná skupina nerozdělí, takže algoritmus končí
- Stavy, které jsou v jedné skupině, jsou ekvivalentní. Můžeme nechat kterýkoliv z nich a ostatní odstranit
- Šipky vedoucí do odstraňovaných stavů povedou do toho, který ze skupiny necháme
- Pokud skupina obsahovala počáteční stav, bude ponechaný zástupce počáteční

Minimální konečný automat

Definice

Deterministický konečný automat nazveme **minimálním automatem**, jestliže neexistuje automat, který by s ním byl ekvivalentní a měl by menší počet stavů.

Minimální konečný automat

Definice

Deterministický konečný automat nazveme **minimálním automatem**, jestliže neexistuje automat, který by s ním byl ekvivalentní a měl by menší počet stavů.

Lemma

Pro každý regulární jazyk L existuje právě jeden minimální automat A v normovaném tvaru takový, že $L = L(A)$.

Minimální konečný automat

Definice

Deterministický konečný automat nazveme **minimálním automatem**, jestliže neexistuje automat, který by s ním byl ekvivalentní a měl by menší počet stavů.

Lemma

Pro každý regulární jazyk L existuje právě jeden minimální automat A v normovaném tvaru takový, že $L = L(A)$.

Lemma

V každém konečném automatu $A = \{Q, \Sigma, \delta, q_0, F\}$, který není minimální, najdeme alespoň jednu dvojici stavů q_1, q_2 $(q_1 \neq q_2)$ takovou, že $L(q_1) \neq L(q_2)$.

Porovnání konečných automatů

Lemma

Existuje algoritmus, který pro zadané konečné automaty A_1, A_2 rozhodne, zda jsou ekvivalentní, tj. zda $L(A_1) = L(A_2)$.

Důkaz:

- Je-li A_1 nebo A_2 nedeterministický, převedeme jej na deterministický
- Automaty A_1 a A_2 převedeme na minimální
- Minimální automaty převedeme do normovaného tvaru
- Pokud jsou vzniklé minimální automaty v normovaném tvaru stejné, byly původní automaty ekvivalentní. Pokud výsledné automaty nejsou stejné, nebyly původní automaty ekvivalentní.

Neregulární jazyky

Neregulární jazyky

Ne všechny jazyky jsou regulární.

Existují jazyky, pro které neexistuje žádný konečný automat, který by je rozpoznával.

Příklady neregulárních jazyků:

- $L_1 = \{a^n b^n \mid n \geq 0\}$
- $L_2 = \{ww \mid w \in \{a, b\}^*\}$
- $L_3 = \{ww^R \mid w \in \{a, b\}^*\}$

Poznámka: Existence neregulárních jazyků vyplývá již z faktu, že automatů pracujících nad nějakou abecedou Σ je jen spočetně mnoho, zatímco jazyků nad abecedou Σ je nespočetně mnoho.

Neregulární jazyky

Jak dokázat o nějakém jazyce L, že není regulární?

Jazyk není regulární, jestliže neexistuje (tj. není možné sestrojit) konečný automat, který by ho rozpoznával.

Jak ale dokázat, že něco neexistuje?

Neregulární jazyky

Jak dokázat o nějakém jazyce L, že není regulární?

Jazyk není regulární, jestliže neexistuje (tj. není možné sestrojit) konečný automat, který by ho rozpoznával.

Jak ale dokázat, že něco neexistuje?

Odpověď: Stačí ukázat, že jazyk L nemá nějakou vlastnost, kterou má každý jazyk rozpoznávaný nějakým konečným automatem.

Neregulární jazyky

Jak dokázat o nějakém jazyce L, že není regulární?

Jazyk není regulární, jestliže neexistuje (tj. není možné sestrojit) konečný automat, který by ho rozpoznával.

Jak ale dokázat, že něco neexistuje?

Odpověď: Stačí ukázat, že jazyk L nemá nějakou vlastnost, kterou má každý jazyk rozpoznávaný nějakým konečným automatem.

Dva základní postupy dokazování neregularity jazyků:

- využití tzv. pumping lemmatu
- využití Myhillovy-Nerodovy věty (nebudeme se jí dále zabývat)

Neregulární jazyky

Tvrzení

Každý konečný jazyk je regulární.

Důkaz: Konečný jazyk s n slovy můžeme popsat regulárním výrazem $w_1 + w_2 + \ldots + w_n$. Jazyk je tedy regulární.

Neregulární jazyky

Tvrzení

Každý konečný jazyk je regulární.

Důkaz: Konečný jazyk s n slovy můžeme popsat regulárním výrazem $w_1 + w_2 + \ldots + w_n$. Jazyk je tedy regulární.

Tvrzení

Je-li jazyk nekonečný, obsahuje pro každou konstantu $k \in \mathbb{N}$ nějaké slovo w takové, že $|w| > k$

Důkaz: Uvažujeme jen konečnou abecedu. Pro každou konstantu je tedy jen konečně mnoho slov kratších než tato konstanta. Pokud má být jazyk nekonečný, musí být i slovo delší, než jakákoliv konstanta. Ve skutečnosti je slov delších než jakákoliv konstanta nekonečně mnoho.

Pumping Lemma

Předpokládejme, že jazyk L je rozpoznáván nějakým konkrétním deterministickým automatem A, tj. $L = L(A)$.

Vezměme nyní nějaké libovolné slovo $z \in L$, kde $z = a_1 a_2 \cdots a_k$.

Protože automat A slovo z přijímá, musí tomuto slovu odpovídat určitý přijímající výpočet automatu, tj. posloupnost stavů:

$$q_0,\ q_1,\ q_2,\ \ldots,\ q_{k-1},\ q_k$$

délky $k+1$, kde

- q_0 je počáteční stav
- $\delta(q_{i-1}, a_i) = q_i$ pro $\forall i \in \{1, 2, \ldots, k\}$
- q_k je přijímající stav

Pumping Lemma

Předpokládejme, že A má n stavů a že $|z| \geq n$.

Pak máme posloupnost délky minimálně $n+1$, ve které se může vyskytovat maximálně n různých stavů.

Z toho plyne, že musí existovat alespoň jeden stav q, který se v této posloupnosti vyskytuje alespoň dvakrát.

Jde o aplikaci tzv. **holubníkového principu (pigeonhole principle)**.

Holubníkový princip

Jestliže mám $n+1$ holubů rozmístěných do n klecí, pak jsou alespoň v jedné kleci minimálně dva holubi.

Pumping Lemma

Řekněme, že opakující stav se vyskytuje na pozicích i, j, tj. $q_i = q_j$, kde $i < j$.

$$q_0, \cdots, q_i, \cdots, q_j, \cdots, q_k$$

Poznámka: Zjevně můžeme najít taková i, j, že $i < j \leq n$

Slovo z můžeme rozdělit na tři části:

$$\underbrace{a_1 \cdots a_i}_{u} \quad \underbrace{a_{i+1} \cdots a_j}_{v} \quad \underbrace{a_{j+1} \cdots a_k}_{w}$$

- $\delta^*(q_0, u) = q_i$
- $\delta^*(q_i, v) = q_j = q_i$
- $\delta^*(q_j, w) = q_k$

Pumping Lemma

Vezměme nyní slova:

$$\underbrace{a_1 \cdots a_i}_{u} \quad \underbrace{a_{j+1} \cdots a_k}_{w}$$

$$\underbrace{a_1 \cdots a_i}_{u} \quad \underbrace{a_{i+1} \cdots a_j}_{v} \quad \underbrace{a_{i+1} \cdots a_j}_{v} \quad \underbrace{a_{j+1} \cdots a_k}_{w}$$

$$\underbrace{a_1 \cdots a_i}_{u} \quad \underbrace{a_{i+1} \cdots a_j}_{v} \quad \underbrace{a_{i+1} \cdots a_j}_{v} \quad \underbrace{a_{i+1} \cdots a_j}_{v} \quad \underbrace{a_{j+1} \cdots a_k}_{w}$$

$$\cdots$$

Je zřejmé, že A přijme každé z nich, vzhledem k tomu, že

- $\delta^*(q_0, u) = q_i$
- $\delta^*(q_i, v) = q_j = q_i$
- $\delta^*(q_j, w) = q_k$, $q_k \in F$

Pumping Lemma

Pumping Lemma

Jestliže jazyk L je regulární, pak existuje n takové, že každé slovo $z \in L$ takové, že $|z| \geq n$, je možné rozdělit na podslova u, v, w taková, že $z = uvw$, $|uv| \leq n$, $|v| \geq 1$ a pro všechna $i \geq 0$ platí $uv^i w \in L$.

Formálně zapsáno:

Jestliže L je regulární, pak

$(\exists n)(\forall z \text{ tž. } z \in L, |z| \geq n)(\exists u, v, w \text{ tž. } z = uvw, |uv| \leq n, |v| \geq 1)$
$(\forall i \geq 0) : uv^i w \in L$

Pumping Lemma

Tvrzení je možné obrátit. ($A \Rightarrow B$ je totéž, co $\neg B \Rightarrow \neg A$.)

Jestliže

$(\forall n)(\exists z \text{ tž. } z \in L, |z| \geq n)(\forall u, v, w \text{ tž. } z = uvw, |uv| \leq n, |v| \geq 1)$
$(\exists i \geq 0) : uv^i w \notin L,$

pak L není regulární.

Pokud tedy chceme ukázat, že jazyk L není regulární, stačí ukázat, že splňuje výše uvedenou podmínku.

Pumping Lemma

Příklad: Uvažujme jazyk $L = \{a^i b^i \mid i \geq 0\}$.

- Předpokládáme, že L je rozpoznáván nějakým automatem s n stavy.
- Zvolme slovo $z = a^n b^n$.
- Uvažujme všechny možnosti, jak může být z rozděleno na podslova u, v, w splňující podmínky $|uv| \leq n$ a $|v| \geq 1$.
  Zjevně slova u a v obsahují pouze symboly a. Pro každé konkrétní rozdělení existují nějaká j a k taková, že $j + k \leq n$, $k \geq 1$ a
  - $u = a^j$
  - $v = a^k$
  - $w = a^{n-(j+k)} b^n$
- Pokud nyní zvolíme $i = 0$, dostáváme $uv^i w = uw = a^{n-k} b^n$. Protože $n - k < n$, zjevně platí $uv^i w \notin L$.

Pumping Lemma

Poznámka: Dokazování platnosti či neplatnosti formule prvého řádu, kde se střídají universální a existenční kvantifikátory, si můžeme představovat jako hru dvou hráčů, hráče A a hráče B, kde se hráč A snaží dokázat, že formule platí, a hráč B, že formule neplatí.

Hráč A vždy volí hodnotu proměnné vázané existenčním kvantifikátorem a naopak hráč B vždy volí hodnotu proměnné vázané universálním kvantifikátorem.

Pokud například chceme platnost formule vyvrátit, stačí nám najít vítěznou strategii hráče B.

Pumping Lemma

$$(\exists n)(\forall z \text{ tž. } z \in L, |z| \geq n)(\exists u, v, w \text{ tž. } z = uvw, |uv| \leq n, |v| \geq 1)$$
$$(\forall i \geq 0) : uv^i w \in L$$

Konkrétně pro Pumping Lemma vypadá hra následovně:

1. A zvolí $n > 0$.
2. B zvolí slovo z takové, že $z \in L$ a $|z| \geq n$.
3. A zvolí slova u, v, w taková, že $z = uvw, |uv| \leq n, |v| \geq 1$.
4. B zvolí $i \geq 0$.
5. Je-li $uv^i w \in L$, vyhrává hráč A. Je-li $uv^i w \notin L$, vyhrává hráč B.

Pumping Lemma

Příklad: $L = \{a^i b^i \mid i \geq 0\}$

1. A zvolí $n > 0$.
2. B zvolí $z = a^n b^n$
3. A zvolí slova u, v, w tž. $z = uvw, |uv| \leq n, |v| \geq 1$
4. B zvolí $i = 0$
5. Vyhrál hráč B, protože bez ohledu na to, co volil hráč A, vždy platí $uv^i w \notin L$, protože neprázdné slovo v se nachází v části slova z tvořené pouze symboly a, a jeho vypuštěním dostaneme slovo tvaru $a^k b^n$, kde $k < n$, a tedy nepatřící do L.