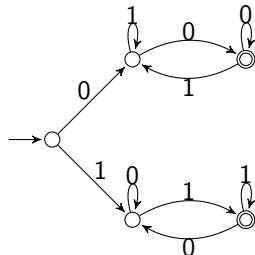


Konečné automaty, Regulární jazyky

- **Deterministický konečný automat (DFA)**
 $A = (Q, \Sigma, \delta, q_0, F)$.
- **Jazykem rozpoznávaným (akceptovaným, přijímaným)** konečným automatem $A = (Q, \Sigma, \delta, q_0, F)$ nazveme jazyk $L(A) = \{w \mid w \in \Sigma^* \& \delta^*(q_0, w) \in F\}$.
- Jazyk L je **rozpoznatelný** konečným automatem, jestliže existuje konečný automat A takový, že $L = L(A)$.
- Třidu jazyků rozpoznatelných deterministickými konečnými automaty označíme \mathcal{F} , nazveme **regulární jazyky**.
- Typická otázka na cvičeních, občas i u zkoušky:
 Je daný jazyk regulární (později i CFL, ...)?



ANO Sestrojíte automat (deterministický či nedeterministický).

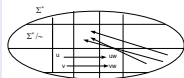
NE Najdete spor s Myhill–Nerodovou větou nebo s Pumping lemmatem.

Kongruence, Myhill–Nerodova věta

Definition 2.1 (kongruence)

Mějme konečnou abecedu Σ a relaci ekvivalence \sim na Σ^* (reflexivní, symetrická, tranzitivní). Potom:

- \sim je **pravá kongruence**, jestliže
 $(\forall u, v, w \in \Sigma^*) u \sim v \Rightarrow uw \sim vw$.
- je **konečného indexu**, jestliže rozklad Σ^* / \sim má konečný počet tříd.
- Třidu kongruence \sim obsahující slovo u značíme $[u]_{\sim}$, resp. $[u]$.



Example 2.1

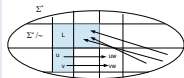
- Relace \sim_{end} 'končí stejným písmenem' je pravá kongruence,
 - pokud $ux \sim_{end} vx$, pak i $uxw \sim_{end} vxw$.
- Relace \sim_{fl} 'končí stejně jako začíná' je ekvivalence, $aa \sim_{fl} bb$, ale $aaa \not\sim_{fl} bba$, tedy není pravá kongruence.
- Relace $\sim_{||}$ 'mají stejný počet znaků' není konečného indexu.

Myhill–Nerodova věta

Theorem 2.1 (Myhill–Nerodova věta)

Nechť L je jazyk nad konečnou abecedou Σ . Potom následující tvrzení jsou ekvivalentní:

- L je rozpoznatelný konečným automatem,
- existuje pravá kongruence \sim konečného indexu nad Σ^* tak, že L je sjednocením jistých tříd rozkladu Σ^* / \sim .



Proof: Důkaz Myhill–Nerodovy věty \Rightarrow

a) \Rightarrow b); tj. automat \Rightarrow pravá kongruence konečného indexu

- definujeme $u \sim v \equiv \delta^*(q_0, u) = \delta^*(q_0, v)$.
- je to ekvivalence (reflexivní, symetrická, transitivní)
- je to pravá kongruence (z definice δ^*)
- má konečný index (konečně mnoho stavů)

$$L = \{w \mid \delta^*(q_0, w) \in F\} = \bigcup_{q \in F} \{w \mid \delta^*(q_0, w) = q\} = \bigcup_{q \in F} [w \mid \delta^*(q_0, w) = q]_{\sim}.$$

Theorem (Myhill–Nerodova věta - 'kopie')

Nechť L je jazyk nad konečnou abecedou Σ . Potom následující tvrzení jsou ekvivalentní:

- L je rozpoznatelný konečným automatem,
- existuje pravá kongruence \sim konečného indexu nad Σ^* tak, že L je sjednocením jistých tříd rozkladu Σ^* / \sim .

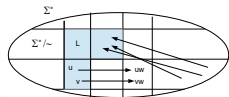
Proof: Důkaz Myhill–Nerodovy věty \Leftarrow

b) \Rightarrow a); tj. pravá kongruence konečného indexu \Rightarrow automat

- abeceda automatu vezmeme Σ
- za stavy Q volíme třídy rozkladu Σ^* / \sim
- počáteční stav $q_0 \equiv [\epsilon]$
- koncové stavy $F = \{c_1, \dots, c_n\}$, kde $L = \bigcup_{i=1, \dots, n} c_i$
- přechodová funkce $\delta([u], x) = [ux]$ (je korektní z def. pravé kongruence).
- $L(A) = L$

$$w \in L \Leftrightarrow w \in \bigcup_{i=1, \dots, n} c_i \Leftrightarrow w \in c_1 \vee \dots \vee w \in c_n \Leftrightarrow [w] = c_1 \vee \dots \vee [w] = c_n \Leftrightarrow [w] \in F \Leftrightarrow w \in L(A)$$

$$\delta^*([\epsilon], w) = [w]$$



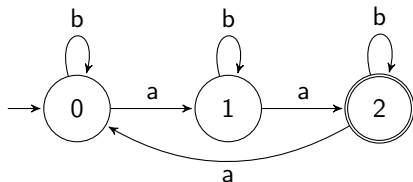
Použití Myhill–Nerodovy věty: Konstrukce automatů

Example 2.2

Sestrojte automat přijímající jazyk

$L = \{w \mid w \in \{a, b\}^* \& |w|_a = 3k + 2\}$, tj. obsahuje $3k + 2$ symbolů a .

- $|u|_x$ značí počet symbolů x ve slově u
- definujme $u \sim v \equiv (|u|_a \bmod 3 = |v|_a \bmod 3)$
- třídy ekvivalence 0,1,2
- L odpovídá třídě 2
- a – přechody do následující třídy
- b – přechody zachovávají třídu.



'Pumpovatelný' ne-regulární jazyk

Example 2.3 (Ne-regulární jazyk, který lze pumpovat)

Jazyk $L = \{u \mid u = a^+ b^i c^i \vee u = b^i c^j, + \in N_{>0}, i, j \in N\}$ není regulární (Myhill–Nerodova věta), ale vždy lze pumpovat první písmeno.

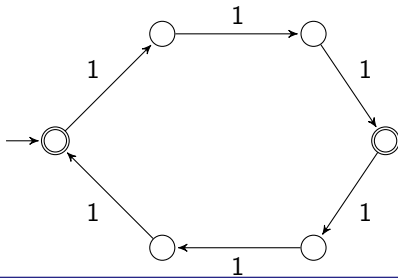
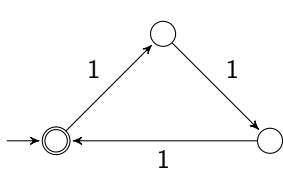
- Předpokládejme, že L je regulární
- ⇒ pak existuje pravá kongruence \sim_L konečného indexu m , L je sjednocení některých tříd Σ^* / \sim_L
- vezmeme množinu řetězců $S = \{ab, abb, abbb, \dots, ab^{m+1}\}$
- existují dvě slova $i \neq j$, která padnou do stejné třídy

$i \neq j$	$ab^i \sim ab^j$	
přidáme c^i	$ab^i c^i \sim ab^j c^i$	\sim je kongruence
spor	$ab^i c^i \in L \ \& \ ab^j c^i \notin L$	s 'L je sjednocení některých tříd Σ^* / \sim_L

Nejednoznačnost

Automat přijímající daný jazyk není určen jednoznačně.

- Jazyk $L = \{w \mid w \in \{1\}^* \& |w| = 3k\}$.



Definition 2.2 (automatový homomorfismus)

Nechť A_1, A_2 jsou DFA. Řekneme, že zobrazení $h : Q_1 \rightarrow Q_2$ na Q_2 je **(automatovým) homomorfismem**, jestliže:

- | | |
|------------------------------------------|----------------------------|
| $h(q_{0_1}) = q_{0_2}$ | 'stejně' počáteční stavy |
| $h(\delta_1(q, x)) = \delta_2(h(q), x)$ | 'stejně' přechodové funkce |
| $q \in F_1 \Leftrightarrow h(q) \in F_2$ | 'stejně' koncové stavy. |

Homomorfismus prostý a na nazýváme **isomorfismus**.

Ekvivalence automatů a homomorfismus

Definition 2.3 (Ekvivalence automatů!)

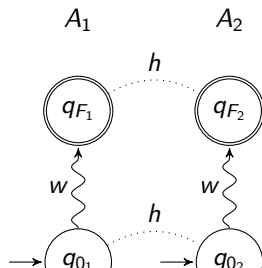
Dva konečné automaty A, B nad stejnou abecedou Σ jsou **ekvivalentní**, jestli že rozpoznávají stejný jazyk, tj. $L(A) = L(B)$.

Theorem 2.2 (Věta o ekvivalenci automatů)

Existuje-li homomorfismus konečných automatů A_1 do A_2 , pak jsou A_1 a A_2 ekvivalentní.

Proof:

- Pro libovolný řetězec $w \in \Sigma^*$ konečnou iterací
 - $h(\delta_1^*(q, w)) = \delta_2^*(h(q), w)$
- dále: $w \in L(A_1) \Leftrightarrow \delta_1^*(q_{0_1}, w) \in F_1$
 - $\Leftrightarrow h(\delta_1^*(q_{0_1}, w)) \in F_2$
 - $\Leftrightarrow \delta_2^*(h(q_{0_1}), w) \in F_2$
 - $\Leftrightarrow \delta_2^*(q_{0_2}, w) \in F_2$



Redukce a ekvivalence automatů, Tranzitivita

Definition 2.4 (Ekvivalence stavů)

Říkáme, že stavy $p, q \in Q$ konečného automatu A jsou **ekvivalentní** pokud:

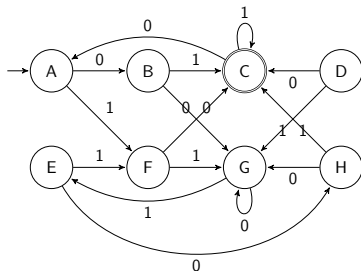
- Pro všechny řetězce $w \in \Sigma^*$; $\delta^*(p, w) \in F$ iff $\delta^*(q, w) \in F$.

Pokud dva stavy nejsou ekvivalentní, říkáme, že jsou **rozlišitelné**.

Example 2.4

Automat na obrázku:

- C a G nejsou ekvivalentní, $\delta^*(C, \epsilon) \in F$ a $\delta^*(G, \epsilon) \notin F$.
- A, G: $\delta^*(A, 01) = C$ je přijímající, $\delta^*(G, 01) = E$ není.
- A, E jsou ekvivalentní – ϵ , 1^* zřejmě, 0 vede do ne-přijímajících stavů, 01 a 00 se sejdou ve stejném stavu.



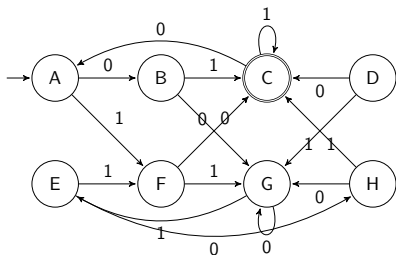
Lemma

Ekvivalence na stavech je tranzitivní.

Algorithm: Algoritmus hledání rozlišitelných stavů v DFA

Následující algoritmus nalezne rozlišitelné stavy:

- Základ: Pokud $p \in F$ (přijímající) a $q \notin F$, pak je dvojice $\{p, q\}$ rozlišitelná.
- Indukce: Nechtě $p, q \in Q$, $a \in \Sigma$ a o dvojici r, s ; $r = \delta(p, a)$ a $s = \delta(q, a)$ víme, že jsou rozlišitelné. Pak i $\{p, q\}$ jsou rozlišitelné.
 - opakuj dokud existuje nová trojice $p, q \in Q$, $a \in \Sigma$.



B	x						
C	x	x					
D	x	x	x				
E		x	x	x			
F	x	x	x		x		
G	x	x	x	x	x	x	
H	x		x	x	x	x	x
	A	B	C	D	E	F	G

Křížek značí rozlišitelné dvojice. C je rozlišitelné hned, ostatní kromě $\{A, G\}$, $\{E, G\}$ také. Vidíme tři ekvivalentní dvojice stavů.

Algoritmus hledání rozlišitelných stavů

Přijímající vs. nepřijímající stavy

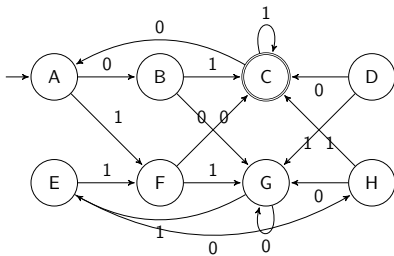
B							
C	x	x					
D			x				
E			x	x			
F			x	x	x		
G			x	x	x	x	
H			x	x	x	x	x
	A	B	C	D	E	F	G

1.krok1: $\delta(q, 1) \in F$ pro $q \in \{B, C, H\}$

B	x						
C	x	x					
D		x	x				
E		x	x	x			
F		x	x	x	x		
G		x	x	x	x	x	
H	x		x	x	x	x	x
	A	B	C	D	E	F	G

1.krok0: $\delta(q, 0) \in F$ pro $q \in \{D, F\}$

B	x						
C	x	x					
D	x	x	x				
E		x	x	x			
F	x	x	x	x	x		
G		x	x	x	x	x	
H	x		x	x	x	x	x
	A	B	C	D	E	F	G



B a G jsou rozlišitelné, $\delta(A, 0) = B$, $\delta(G, 0) = G$, tj. A, G jsou rozlišitelné.
 Podobně pro E, G vedoucí $\delta(*, 0)$ do rozlišitelných stavů H, G.

B	x						
C	x	x					
D	x	x	x				
E		x	x	x			
F	x	x	x	x	x		
G	x	x	x	x	x	x	
H	x		x	x	x	x	x
	A	B	C	D	E	F	G

Zůstávají tři ekvivalentní dvojice stavů.

Theorem 2.3

Pokud dva stavy nejsou odlišeny předchozím algoritmem, pak jsou tyto stavy ekvivalentní.

Proof: Korektnost algoritmu

- Uvažujme špatné páry stavů, které jsou rozlišitelné a algoritmus je nerozlišil.
- Vezměme z nich pár p, q rozlišitelný nejkratším slovem $w = a_1 \dots a_n$.
- Stavy $r = \delta(p, a_1)$ a $s = \delta(q, a_1)$ jsou rozlišitelné kratším slovem $a_2 \dots a_n$ takže pár není mezi špatnými.
Tedy jsou 'vykřížkované' algoritmem.
- Tedy v příštím kroku algoritmus rozliší i p, q . □

Čas výpočtu je polynomiální vzhledem k počtu stavů.

- V jednom kole uvažujeme všechny páry, tj. $O(n^2)$.
- Kol je maximálně $O(n^2)$, protože pokud nepřidáme křížek, končíme.
- Dohromady $O(n^4)$.

Algoritmus lze zrychlit na $O(n^2)$ pamatováním stavů, které závisí na páru $\{r, s\}$ a následováním těchto seznamů 'zpatky'.

Testování ekvivalence regulárních jazyků

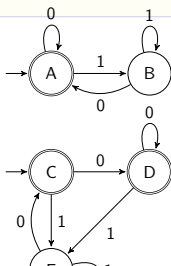
Algorithm: **!** Testování ekvivalence regulárních jazyků

Ekvivalenci regulárních jazyků L, M testujeme následovně:

- Najdeme DFA A_L, A_M rozpoznávající $L(A_L) = L, L(A_M) = M$, $Q_L \cap Q_M = \emptyset$.
- Vytvoříme DFA sjednocením stavů a přechodů $(Q_L \cup Q_M, \Sigma, \delta_L \cup \delta_M, q_L, F_L \cup F_M)$; zvolíme jeden z počátečních stavů.
- Jazyky jsou ekvivalentní právě když počáteční stavy původních DFA jsou ekvivalentní.

Example 2.5

Uvažujme jazyk $\{\epsilon\} \cup \{0, 1\}^*0$ přijímající prázdné slovo a slova končící 0. Vpravo obrázek dvou DFA a tabulku rozlišitelných stavů.



B	x			
C		x		
D		x		
E	x		x	x
	A	B	C	D

Minimalizace DFA

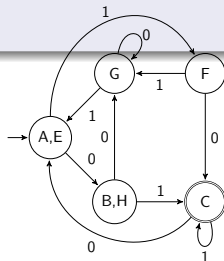
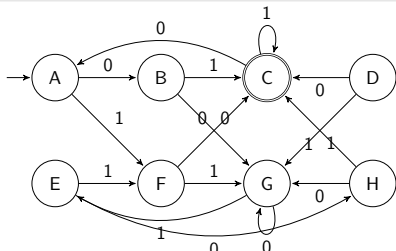
Definition 2.5 (redukovaný DFA, redukt)

Deterministický konečný automat je **redukovaný**, pokud

- nemá nedosažitelné stavy a
- žádné dva stavy nejsou ekvivalentní
- δ je totální.

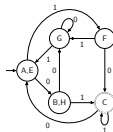
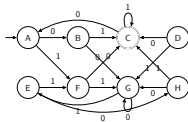
Konečný automat B je **reduktem** automatu A , jestliže:

- B je redukovaný a
- A a B jsou ekvivalentní.

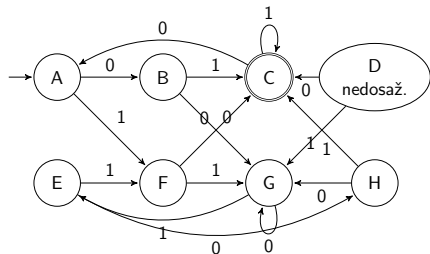


Algorithm: Algoritmus nalezení reduktu DFA A

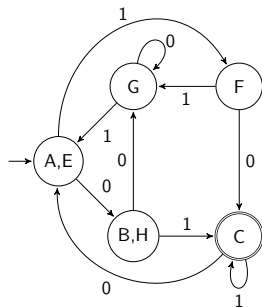
- Ze vstupního DFA A eliminujeme stavy nedosažitelné z počátečního stavu.
- Najdeme rozklad zbylých stavů na třídy ekvivalence.
- Konstruujeme DFA B na třídách ekvivalence jakožto stavech. Přejchodovou funkci B označíme γ , mějme $S \in Q_B$. Pro libovolné $q \in S$, označíme T třídu ekvivalence $\delta(q, a)$ a definujeme $\gamma(S, a) = T$. Tato třída musí být stejná pro všechny $q \in S$.
- Počáteční stav B je třída obsahující počáteční stav A .
- Množina přijímajících stavů B jsou bloky odpovídající přijímajícím stavům A .



Příklad redukovaného DFA



B	x					
C	x	x				
E		x	x			
F	x	x	x	x		
G	x	x	x	x	x	
H	x		x	x	x	x
	A	B	C	E	F	G



Třídy ekvivalence:

$\{A, E\}, \{B, H\}, \{C\}, \{F\}, \{G\}$

Redukty a jejich ekvivalence

Lemma

Každé dva ekvivalentní redukované automaty jsou izomorfní.

Proof.

- Každý stav $q \in Q_1$ je dosažitelný. Najdeme pro něj slovo $q = \delta_1^*(q_{0_1}, w)$
- a definujeme $h(q) = \delta_2^*(q_{0_2}, w)$.
- Lze dokázat, že je h korektně definovaná funkce, zachovává vlastnosti homomorfizmu (q_0, F, δ) a jde o bijekci, tj. je to isomorfismus.



Lemma

Pro každý deterministický konečný automat A , který přijímá alespoň jedno slovo, existuje redukovaný DFA, který je s ním ekvivalentní.

Definition 2.6 (Redukt)

Reduktem DFA A nazýváme redukovaný automat s A ekvivalentní.

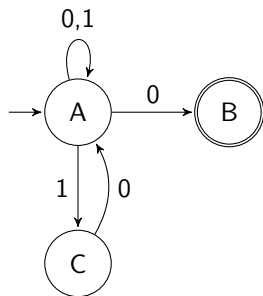
Z předchozí věty plyne, že všechny redukty automatu A jsou izomorfní.

Pro nedeterministické FA to tak snadné není

Example 2.6

Nedeterministický FA na obrázku můžeme redukovat vpuštěním stavu C . Stavů $\{A, C\}$ jsou rozlišitelné vstupem 0 , takže algoritmus pro DFA redukci nenajde.

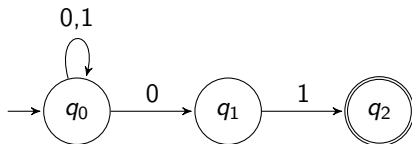
Mohli bychom hledat exhaustivním výpočtem.



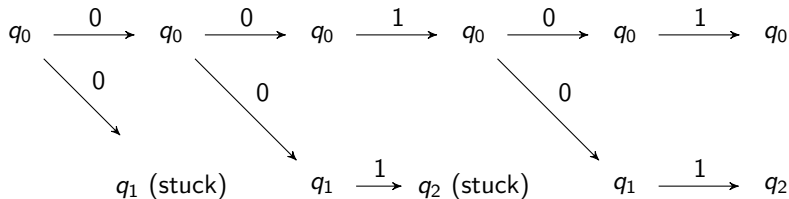
Nedeterministické konečné automaty s ϵ přechody (ϵ -NFA)

Nedeterministický automat může být ve více stavech paralelně. Má schopnost 'uhodnout' něco o vstupu.

NFA přijímající všechna slova končící 01.



NFA zpracovává vstup 00101.



Definition 3.1 (Nedeterministický konečný automat s ϵ přechody (ϵ -NFA))Nedeterministický konečný automat s ϵ přechody (ϵ -NFA)

$A = (Q, \Sigma, \delta, q_0, F)$ sestává z:

- ① konečné množiny **stavů**, zpravidla značíme Q
- ② konečné množiny **vstupních symbolů**, značíme Σ
- ③ **přechodové funkce**, zobrazení $\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}(Q)$ vracející podmnožinu Q .
- ④ **počáteční ho stavu**^a $q_0 \in Q$,
- ⑤ a **množiny koncových (přijímajících) stavů** $F \subseteq Q$.

^aalternativa: množiny počátečních stavů $S_0 \subseteq Q$

Example 3.1

Tabulka pro ϵ -NFA z předchozího slajdu $A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$ je:

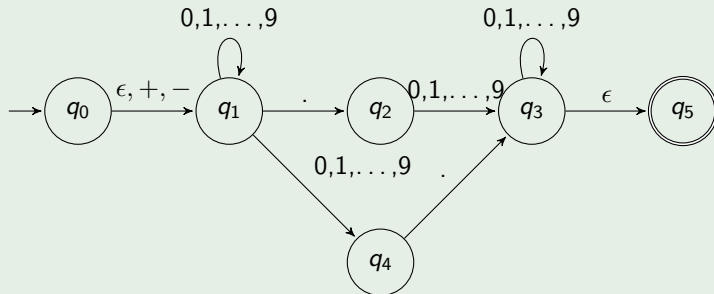
δ	ϵ	0	1
$\rightarrow q_0$	\emptyset	$\{q_0, q_1\}$	$\{q_0\}$
q_1	\emptyset	\emptyset	$\{q_2\}$
$*q_2$	\emptyset	\emptyset	\emptyset

Konečné automaty s ϵ přechody

- Nově dovolíme přechody na ϵ , prázdné slovo, tj. bez přečtení vstupního symbolu.

Example 3.2 (NFA s ϵ přechody)

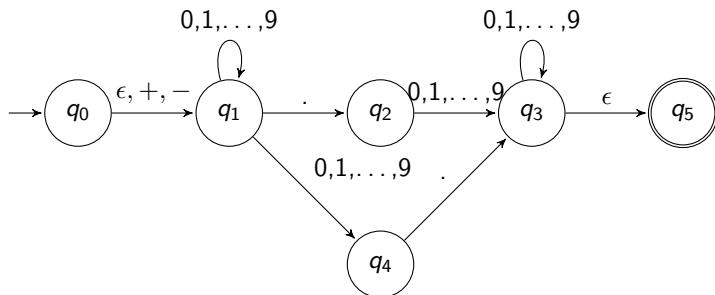
- Volitelně znaménko + nebo - ,
- řetězec číslic,
- desetinná tečka a
- další řetězec číslic. Nejméně jeden z řetězců (2) a (4) musí být neprázdný.



Example 3.3 (Přechodová funkce v tabulce)

Předešlý ϵ -NFA je: $E = (\{q_0, q_1, \dots, q_5\}, \{., +, -, 0, 1, \dots, 9\}, \delta, q_0, \{q_5\})$ s δ :

δ	ϵ	$+, -$	$.$	$0, 1, \dots, 9$
q_0	$\{q_1\}$	$\{q_1\}$	\emptyset	\emptyset
q_1	\emptyset	\emptyset	$\{q_2\}$	$\{q_1, q_4\}$
q_2	\emptyset	\emptyset	\emptyset	$\{q_3\}$
q_3	$\{q_5\}$	\emptyset	\emptyset	$\{q_3\}$
q_4	\emptyset	\emptyset	$\{q_3\}$	\emptyset
$*q_5$	\emptyset	\emptyset	\emptyset	\emptyset



ϵ -uzávěrDefinition 3.2 (ϵ -uzávěr)

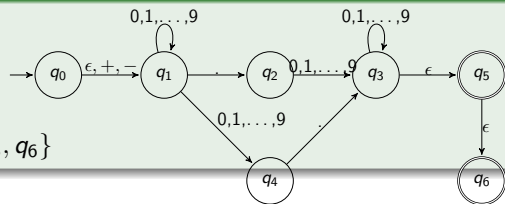
Pro $q \in Q$ definujeme ϵ -uzávěr $\epsilon\text{closure}(q)$ rekurzivně:

- Stav q je v $\epsilon\text{closure}(q)$.
- Je-li $p \in \epsilon\text{closure}(q)$ a $r \in \delta(p, \epsilon)$ pak i $r \in \epsilon\text{closure}(q)$.

Pro množinu stavů $S \subseteq Q$ definujeme $\epsilon\text{closure}(S) = \bigcup_{q \in S} \epsilon\text{closure}(q)$.

Example 3.4 (ϵ uzavěr)

- $\epsilon\text{closure}(q_0) = \{q_0, q_1\}$
- $\epsilon\text{closure}(q_1) = \{q_1\}$
- $\epsilon\text{closure}(q_3) = \{q_3, q_5, q_6\}$
- $\epsilon\text{closure}(\{q_3, q_4\}) = \{q_3, q_4, q_5, q_6\}$



Rozšířená přechodová funkce a jazyk přijímaný ϵ -NFA

Definition 3.3

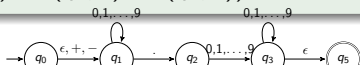
Nechť $E = (Q, \Sigma, \delta, q_0, F)$ je ϵ -NFA. Rozšířenou přechodovou funkci δ^* definujeme následovně:

- $\delta^*(q, \epsilon) = \epsilon\text{closure}(q)$.
- Indukční krok: $v = wa$, kde $w \in \Sigma^*$, $a \in \Sigma$.

$$\delta^*(q, wa) = \epsilon\text{closure} \left(\bigcup_{p \in \delta^*(q, w)} \delta(p, a) \right).$$

Example 3.5

$$\begin{aligned} \delta^*(q_0, \epsilon) &= \epsilon\text{closure}(q_0) &= \{q_0, q_1\} \\ \delta^*(q_0, 5) &= \epsilon\text{closure}(\bigcup_{q \in \delta^*(q_0, \epsilon)} \delta(q, 5)) = \epsilon\text{closure}(\delta(q_0, 5) \cup \delta(q_1, 5)) &= \{q_1, q_4\} \\ \delta^*(q_0, 5.) &= \epsilon\text{closure}(\delta(q_1, .) \cup \delta(q_4, .)) &= \{q_2, q_3, q_5, q_6\} \\ \delta^*(q_0, 5.6) &= \epsilon\text{closure}(\delta(q_2, 6) \cup \delta(q_3, 6) \cup \delta(q_5, 6)) &= \{q_3, q_5, q_6\} \end{aligned}$$



Jazyk přijímaný ϵ NFA

Definition 3.4 (Jazyk přijímaný nedeterministickým konečným automatem s přechody)

Mějme NFA $A = (Q, \Sigma, \delta, q_0, F)$, pak

$$L(A) = \{w \mid \delta^*(q_0, w) \cap F \neq \emptyset\}$$

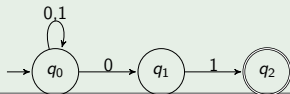
je jazyk přijímaný automatem A .

Tedy $L(A)$ je množina slov $w \in \Sigma^*$ takových, že $\delta^*(q_0, w)$ obsahuje alespoň jeden přijímající stav.

Example 3.6

Automat z předchozího slajdu přijímá jazyk $L = \{w \mid w \text{ končí na } 01, w \in \{0, 1\}^*\}$.
Důkaz indukcí konjunkce tvrzení:

- $\delta^*(q_0, w)$ obsahuje q_0 pro každé slovo w .
- $\delta^*(q_0, w)$ obsahuje q_1 iff w končí 0.
- $\delta^*(q_0, w)$ obsahuje q_2 iff w končí 01.



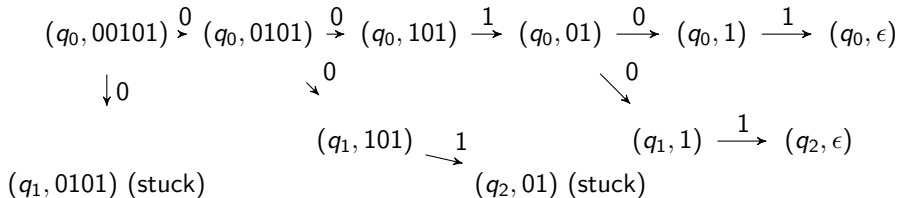
Konfigurace automatu, Výpočetní graf

Definition 3.5 (Konfigurace DFA, ϵ NFA)

Mějme ϵ NFA $A = (Q, \Sigma, \delta, q_0, F)$, $q \in Q$, $v \in \Sigma^*$, pak dvojice (q, v) označuje **konfiguraci** konečného automatu, nacházejícího se ve stavu q s nepřčteným vstupem v .

Definition 3.6 (Výpočetní strom, graf ϵ NFA)

Mějme NFA $A = (Q, \Sigma, \delta, q_0, F)$ a vstupní slovo $w \in \Sigma^*$. Uzly **výpočetního grafu** jsou konfigurace A nad slovem w , orientované hrany značí možný přechod mezi konfiguracemi, tj. z (p, au) vede hrana do (q, u) právě když $q \in \delta(p, a)$.



Podmnožinová konstrukce (s ϵ -přechody)

Theorem 3.1 (Podmnožinová konstrukce (s ϵ -přechody))

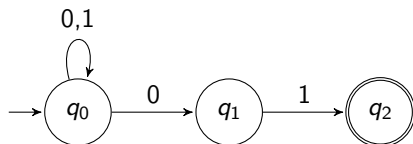
Jazyk L je rozpoznatelný ϵ -NFA právě když je L regulární.

Algorithm: **!** Podmnožinová konstrukce (s ϵ -přechody)

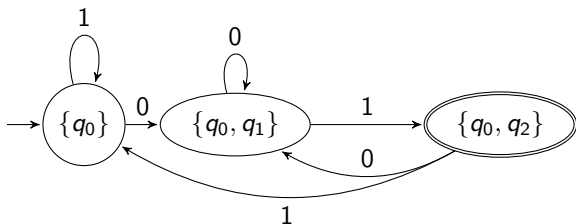
Pro libovolný ϵ -NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ zkonstruujeme DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$ přijímající stejný jazyk jako N .

- Nové stavy jsou ϵ uzavřené podmnožiny Q_N .
 $Q_D \subseteq \mathcal{P}(Q_N), \forall S \subseteq Q_N : \epsilon\text{closure}(S) \in Q_D$. V Q_D může být i \emptyset .
- Počáteční stav je ϵ uzávěr q_0 .
 $q_D = \epsilon\text{closure}(q_0)$.
- Přijímací stavy jsou všechny množiny obsahující nějaký přijímací stav.
 $F_D = \{S \mid S \in Q_D \ \& \ S \cap F_N \neq \emptyset\}$.
- Přechodová funkce sjednotí předchody z jednotlivých stavů a uzavře $\epsilon\text{closure}$.

Pro $S \in Q_D, a \in \Sigma$ definujeme $\delta_D(S, a) = \epsilon\text{closure}(\bigcup_{p \in S} \delta(p, a))$.

Příklad podmnožinové konstrukce pro $\{w.01 \mid w \in \{0, 1\}^*\}$ 

	0	1
\emptyset	\emptyset	\emptyset
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	\emptyset	$\{q_2\}$
$*\{q_2\}$	\emptyset	\emptyset
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*\{q_1, q_2\}$	\emptyset	$\{q_2\}$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$



Theorem 3.2 (Převod NFA na DFA)

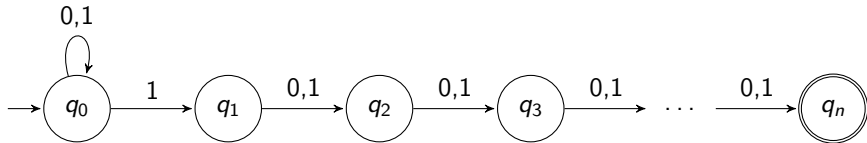
Pro DFA $D = (Q_D, \Sigma, \delta_D, q_{D_0}, F_D)$ vytvořený podmnožinovou konstrukcí z NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ platí $L(N) = L(D)$.

Proof.

Indukcí dokážeme: $\delta_D^*(q_0, w) = \delta_N^*(q_{D_0}, w)$. □

Example 3.7 ('Těžký' případ pro podmnožinovou konstrukci)

Jazyk $L(N)$ slov 0's a 1's takových, že n -tý symbol od konce je 1. Intuitivně si DFA musí pamatovat n posledních přečtených symbolů.



- Aplikace hledání v textu

Regulární jazyky

- **Iterační (pumping) lemma** pro regulární jazyky,
- **Mihyll–Nerodova věta**,
 - užití pro důkaz ne-regulárnosti jazyka
 - příklad ne-regulárního jazyka, který lze pumpovat
- **dosažitelné stavy**, algoritmus nalezení,
- **ekvivalentní automaty, stavy**,
- **rozlišitelné stavy**, algoritmus nalezení,
- **redukovaný DFA**, redukt, **algoritmus nalezení reduktu**.
- **Nedeterministický FA**, **podmnožinová konstrukce**.
- ϵ **nedeterministický FA**, ϵ **uzávěr**.

Množinové operace nad jazyky

Definition 3.7 (Množinové operace nad jazyky)

Mějme dva jazyky L, M . Definujme následující operace:

- (1) binární **sjednocení** $L \cup M = \{w \mid w \in L \vee w \in M\}$
 - Příklad: jazyk obsahuje slova začínající a^i nebo tvaru $b^j c^j$.
- (2) **průnik** $L \cap M = \{w \mid w \in L \ \& \ w \in M\}$
 - Příklad: jazyk obsahuje slova sudé délky končící na 'baa'.
- (3) **rozdíl** $L - M = \{w \mid w \in L \ \& \ w \notin M\}$
 - Příklad: jazyk obsahuje slova sudé délky nekončící na 'baa'.
- (4) **doplňěk (komplement)** $\bar{L} = -L = \{w \mid w \notin L\} = \Sigma^* - L$
 - Příklad: jazyk obsahuje slova nekončící na 'a'.

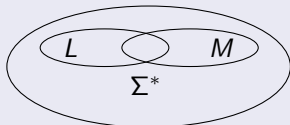
Theorem (de Morganova pravidla)

$$L \cap M = \overline{\bar{L} \cup \bar{M}}$$

Platí: $L \cup M = \overline{\bar{L} \cap \bar{M}}$

$$L - M = L \cap \bar{M}.$$

Důkaz ze vztahů $\&$, \vee , \neg .



Uzávěrové vlastnosti regulárních jazyků

Theorem 3.3 (Uzavřenost na množinové operace)

Mějme regulární jazyky L, M . Pak jsou následující jazyky také regulární:

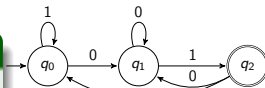
- (1) sjednocení $L \cup M$
- (2) průnik $L \cap M$
- (3) rozdíl $L - M$
- (4) doplněk $\bar{L} = \Sigma^* - L$.

Proof: Uzavřenost RJ na doplněk

- Pokud δ není pro některé dvojice q, a definovaná, přidáme nový nepřijímající stav q_{fail} a do něj přechod pro vše dříve nedefinované plus $\forall a \in \Sigma: \delta(q_{fail}, a) = q_{fail}$.
- Pak stačí prohodit koncové a nekoncové stavy přijímajícího deterministického FA $F = Q_A - F_A$. □

Example 3.8

Jazyk $\{w \mid w \in \{0, 1\}^*01\}$



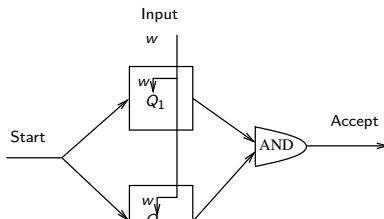
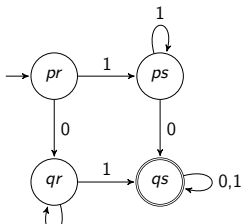
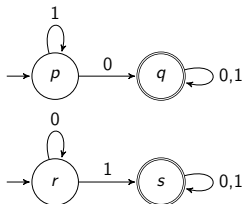
Konstrukce součinu automatů

Proof: Průnik, sjednocení, rozdíl

- Pro sjednocení a rozdíl doplníme funkci δ na totální.
- Zkonstruujeme součinnový automat,
 $Q = (Q_1 \times Q_2, \Sigma, \delta((p_1, p_2), x) = (\delta_1(p_1, x), \delta_2(p_2, x)), (q_{0_1}, q_{0_2}), F)$
- průnik: $F = F_1 \times F_2$
- sjednocení: $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$
- rozdíl: $F = F_1 \times (Q_2 - F_2)$.



Příklad součinu automatů (průnik jazyků). Slova obsahující 0,1, oboje.



Příklady na uzávěrové vlastnosti

Example 3.9

Konstruujeme konečný automat přijímající slova, která obsahují $3k + 2$ symbolů 1 a neobsahují posloupnost 11.

- Přímá konstrukce je komplikovaná.
- $L_1 = \{w \mid w \in \{0, 1\}^* \& |w|_1 = 3k + 2\}$
- $L_2 = \{w \mid u, v \in \{0, 1\}^* \& w = u11v\}$
- $L = L_1 - L_2$.

Example 3.10

Jazyk M slov s různým počtem 0 a 1 není regulární.

- Je-li M regulární, $\overline{M} = \Sigma^* - M$ je také regulární.
- O \overline{M} víme, že regulární není (pumping lemma).

Ještě jeden příklad

Example 3.11

Jazyk $L_{0 \neq 1} = \{0^i 1^j : i \neq j, i, j \in \mathbb{N}_0\}$ není regulární.

- Jazyk $L_{01} = \{0^i 1^j : i, j \in \mathbb{N}_0\}$ je regulární, umíme sestavit konečný automat.
- $L_{01} - L_{0 \neq 1} = \{0^i 1^i : i \in \mathbb{N}_0\}$
- Z uzávěrových vlastností víme, že rozdíl regulárních jazyků je regulární.
- Jazyk L_{01} regulární je.
- Předpokládejme, že $L_{0 \neq 1}$ je regulární. Pak by i $\{0^i 1^i : i \in \mathbb{N}_0\}$ musel být regulární, což není - SPOR.

Řetězcové operace nad jazyky

Definition 3.8 (Řetězcové operace nad jazyky)

Nad jazyky L, M definujeme následující operace:

zřetězení jazyků

$$L.M = \{uv \mid u \in L \ \& \ v \in M\}$$

$$L.x = L.\{x\} \text{ a } x.L = \{x\}.L \text{ pro } x \in \Sigma$$

mocniny jazyka

$$L^0 = \{\epsilon\}$$

$$L^{i+1} = L^i.L$$

pozitivní iterace

$$L^+ = L^1 \cup L^2 \dots = \bigcup_{i \geq 1} L^i$$

obecná iterace

$$L^* = L^0 \cup L^1 \cup L^2 \dots = \bigcup_{i \geq 0} L^i$$

$$\text{tedy } L^* = L^+ \cup \{\epsilon\}$$

otočení jazyka

$$L^R = \{u^R \mid u \in L\}$$

(=zrcadlový obraz, reverze)

$$(x_1x_2 \dots x_n)^R = x_nx \dots x_2x_1$$

levý kvocient L podle M

$$M \setminus L = \{v \mid uv \in L \ \& \ u \in M\}$$

levá derivace L podle w

$$\partial_w L = \{w\} \setminus L \text{ (pozn. derivace bude i v jiném významu)}$$

pravý kvocient L podle M

$$L/M = \{u \mid uv \in L \ \& \ v \in M\}$$

pravá derivace L podle w

$$\partial_w^R L = L/\{w\}.$$

Theorem 3.4 (Uzavřenost reg. jazyků na řetězcové operace)

Jsou-li L, M regulární jazyky, je regulární i $L.M, L^, L^+, L^R, M \setminus L$ a L/M .*

Lemma ($L.M$)

Jsou-li L, M regulární jazyky, je regulární i $L.M$.

Proof:

Vezmeme DFA $A_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$, pak $A_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ tak že $L = L(A_1)$ a $M = L(A_2)$.

Definujeme nedeterministický automat $B = (Q \cup \{q_0\}, \Sigma, \delta, \{q_0\}, F_2)$ kde:

$Q = Q_1 \cup Q_2$ předpokládáme různá jména stavů, jinak přejmenujeme končíme až po přečtení slova z L_2

$\delta(q_0, \epsilon) = \{q_1, q_2\}$	pro $q_1 \in F_1$	tj. $\epsilon \in L(A_1)$
$\delta(q_0, \epsilon) = \{q_1\}$	pro $q_1 \notin F_1$	tj. $\epsilon \notin L(A_1)$
$\delta(q_0, x) = \emptyset$	pro $x \in \Sigma$	
$\delta(q, x) = \{\delta_1(q, x)\}$	pro $q \in Q_1$ & $\delta_1(q, x) \notin F_1$	počítáme v A_1
$\delta(q, x) = \{\delta_1(q, x), q_2\}$	pro $q \in Q_1$ & $\delta_1(q, x) \in F_1$	nedet. přechod z A_1
$\delta(q, x) = \{\delta_2(q, x)\}$	pro $q \in Q_2$	počítáme v A_2 .

Pak $L(B) = L(A_1).L(A_2)$. □

Uzavřenost iterace

Lemma (L^* , L^+)

Je-li L regulární jazyk, je regulární i L^* , L^+ .

- Idea: opakovaný výpočet automatu $A = (Q, \Sigma, \delta, q_0, F)$
- realizace: nedeterministické rozhodnutí, zda pokračovat nebo restart
- speciální stav pro příjem $\epsilon \in L^0$ (pro L^+ vynecháme či $\notin F$).

Proof: Důkaz pro L^*

Vezmeme DFA $A = (Q, \Sigma, \delta, q_0, F)$, tak že $L = L(A)$.

Definujeme NFA automat $B = (Q \cup \{q_B\}, \Sigma, \delta_B, \{q_B\}, F \cup \{q_B\})$ kde:

$\delta_B(q_B, \epsilon) = \{q_0\}$ nový stav q_B pro příjem ϵ , přejdeme do q_0

$\delta_B(q_B, x) = \emptyset$ pro $x \in \Sigma$

$\delta_B(q, x) = \{\delta(q, x)\}$ pokud $q \in Q$ & $\delta(q, x) \notin F$ uvnitř A

$= \{\delta(q, x), q_0\}$ pokud $q \in Q$ & $\delta(q, x) \in F$ možný restart

Pak $L(B) = L(A)^*$ ($q_B \in F_B$), $L(B) = L(A)^+$ ($q_B \notin F_B$). □

Uzavřenost reverze

Lemma (L^R)

Je-li L regulární jazyk, je regulární i L^R .

- Zřejmě $(L^R)^R = L$ a tedy stačí ukázat jeden směr.
- idea: obrátíme šipky ve stavovém diagramu; nedeterministický FA

Proof:

Vezmeme DFA $A = (Q, \Sigma, \delta, q_0, F)$, tak že $L = L(A)$.

Definujeme nedeterministický automat $B = (Q \cup \{q_B\}, \Sigma, \delta_B, \{q_B\}, \{q_0\})$ kde:

- $\delta_B(q, x) = \{p \mid \delta(p, x) = q\}$ pro $q \in Q$
- $\delta_B(q_B, \epsilon) = F$, $\delta_B(q_B, x) = \emptyset$.
- Pro libovolné slovo $w = x_1x_2 \dots x_n$
 - $q_0, q_1, q_2, \dots, q_n$ je přijímající výpočet pro w v A

\Leftrightarrow

- $q_B, q_n, q_{n-1}, \dots, q_2, q_1, q_0$ je přijímající výpočet pro w^R v B . □

Uzavřenost kvocientu

Lemma ($M \setminus L$ a L/M)

Jsou-li L, M regulární jazyky, je regulární i $M \setminus L$ a L/M .

- Idea: A_L , budeme startovat ve stavech, do kterých se lze dostat slovem z M

Proof:

- $v \in M \setminus L$
 - $\Leftrightarrow (\exists u \in M) uv \in L$
 - $\Leftrightarrow (\exists u \in M, \exists q \in Q) \delta(q_0, u) = q \ \& \ \delta(q, v) \in F$
 - $\Leftrightarrow \exists q \in S_0 \ \& \ \delta(q, v) \in F$
 - $\Leftrightarrow v \in L(B)$

Vezmeme DFA $A = (Q, \Sigma, \delta, q_0, F)$, tak že $L = L(A)$.

Definujeme nedeterministický NFA $B = (Q, \Sigma, \delta, S_0, F)$ kde:

- definujeme $S_0 = \{q \mid q \in Q \ \& \ (\exists u \in M) q = \delta(q_0, u)\}$
 - Ize nalézt algoritmicky
 $(\{q; L(A_q) \cap M \neq \emptyset \text{ kde } A_q = (Q, \Sigma, \delta, q_0, \{q\})\})$

$$L/M = (M^R \setminus L^R)^R$$



Přehled kapitol

- 1 Úvod, Iterační lemma pro reg. jazyky
- 2 Redukovaný DFA a ekvivalence automatů, stavů
- 3 Nedeterministické ϵ -NFA, Operace zachovávající regularitu
- 4 Regulární výrazy, Kleeneova věta, Substitute, Homomorfizmus
- 5 Dvousměrné FA, Mealy a Moore stroje
- 6 Gramatiky, Chomského hierarchie, víceznačnost
- 7 Chomského NF, Pumping Lemma pro CFL
- 8 CYK – náležení do CFL
- 9 Zásobníkové automaty, Deterministické PDA
- 10 Uzávěrové vlastnosti, Dykovy jazyky
- 11 Turingův stroj, rozšíření
- 12 Lineárně omezené automaty, Univerzální TM, Diagonální jazyk
- 13 Nerozhodnutelné problémy, Postův korespondenční p.
- 14 Časová složitost