Dnešní téma

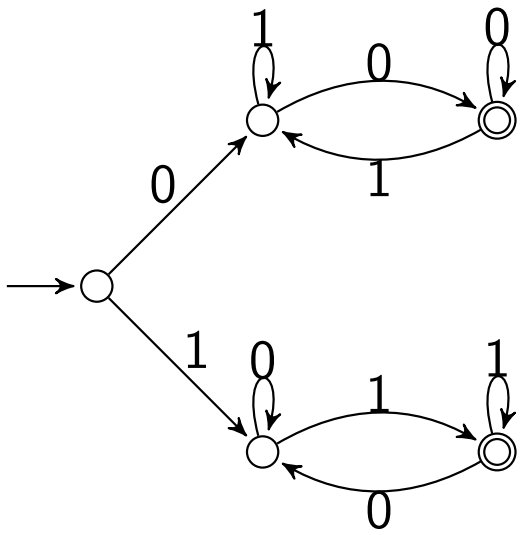
Iterační lemma, Redukovaný DFA, ϵ–NFA

Marta Vomlelová

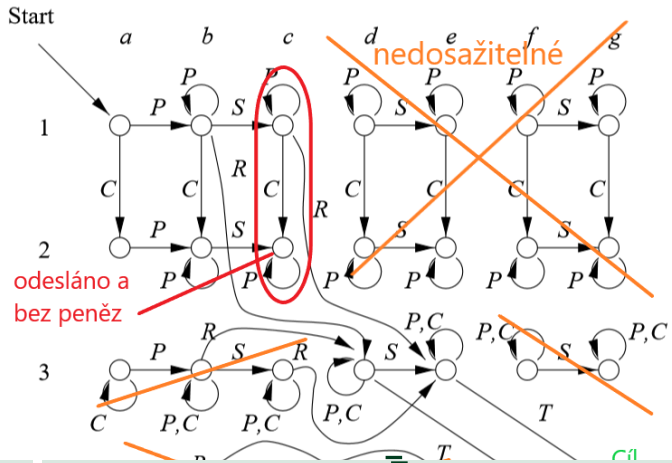
MS 303

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 $L(A) = \{w | w \in \Sigma^* \& \delta^*(q_0, w) \in F\}$.
- Jazyk L je **rozpoznatelný** konečným automatem, jestliže existuje konečný automat A takový, že $L = L(A)$.
- Třídu jazyků rozpoznatelných deterministickými konečnými automaty označíme $\mathcal{F}$, nazveme **regulární jazyky**.



- Dnes budeme zkoumat:
  - Dá se do každého stavu dojít?
  - Je v přechodové funkci cyklus?
  - Jak automat redukovat?
  - Může být nedeterministický?



Definition 2.7 (Dosažitelné stavy)

Mějme DFA $A = (Q, \Sigma, \delta, q_0, F)$ a $q \in Q$. Řekneme, že stav q je **dosažitelný**, jestliže existuje $w \in \Sigma^*$ takové, že $\delta^*(q_0, w) = q$.

Algorithm: Hledání dosažitelných stavů

Dosažitelné stavy hledáme iterativně.

- Začátek: $M_0 = \{q_0\}$.
- Opakuj: $M_{i+1} = M_i \cup \{q | q \in Q, (\exists p \in M_i,\ \exists x \in \Sigma)\ \delta(p, x) = q\}$
- opakuj dokud $M_{i+1} \neq M_i$.

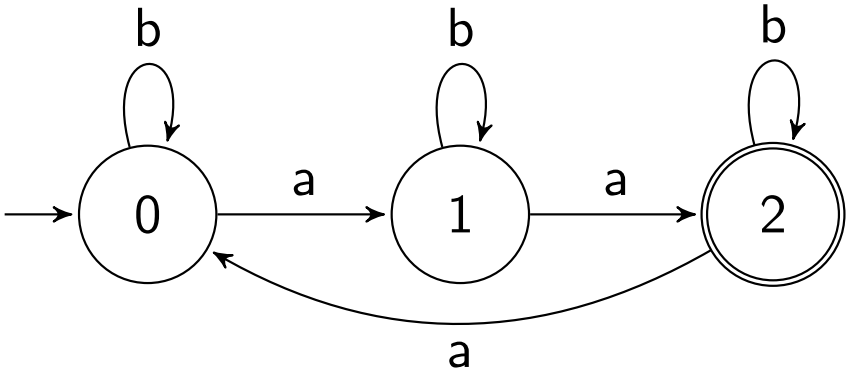
Proof: Korektnost a úplnost

- Korektnost: $M_0 \subseteq M_1 \subseteq \ldots \subseteq Q$ a každé M_i obsahuje pouze dosažitelné stavy.
- Úplnost:
  - nechť q je dosažitelný, tj. $(\exists w \in \Sigma^*)\delta^*(q_0, w) = q$
  - vezměme nejkratší takové $w = x_1 \ldots x_n$ tž. $\delta^*(q_0, x_1 \ldots x_n) = q$
  - zřejmě $\delta^*(q_0, x_1 \ldots x_i) \in M_i$ (dokonce $M_i \setminus M_{i-1}$)
  - tedy $\delta^*(q_0, x_1 \ldots x_n) \in M_n$, tedy $q \in M_n$.

□

Projde automat cyklus?

Automat A



Example 2.11

Projde automat výše cyklus při čtení slova:

- $abbbba$?
- $abbbba = a(b)bbba$; $\forall i \geq 0; a(b)^i bbba \in L(A)$.
- $aaaaba$?
- $aaaaba = (aaa)aba$; $\forall i \geq 0; (aaa)^i aba \in L(A)$.
- aa ?
- aa neprojde cyklus, ale délka $|aa| <$ počet stavů.

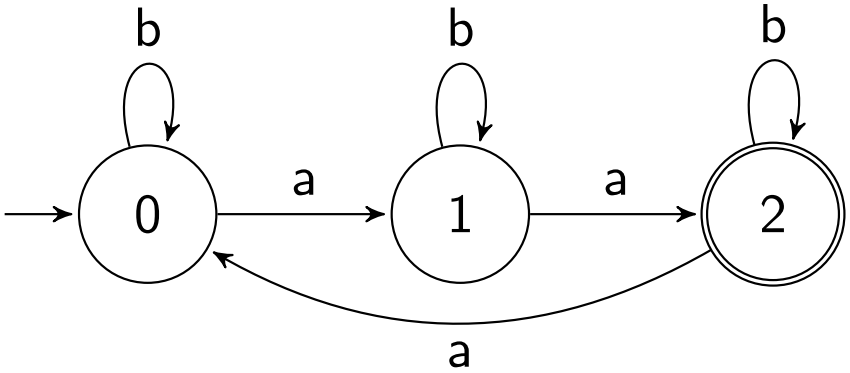
Iterační (pumping) lemma pro regulární jazyky

Theorem 2.2 (!Iterační (pumping) lemma pro regulární jazyky)

Mějme regulární jazyk L. Pak existuje konstanta $n \in \mathbb{N}$ (závislá na L) tak že každé $w \in L; |w| \geq n$ můžeme rozdělit na tři části, $w = xyz$, že:

- $y \neq \epsilon$
- $|xy| \leq n$
- $\forall k \in \mathbb{N}_0$, *slovo* $xy^k z$ *je také v* L.

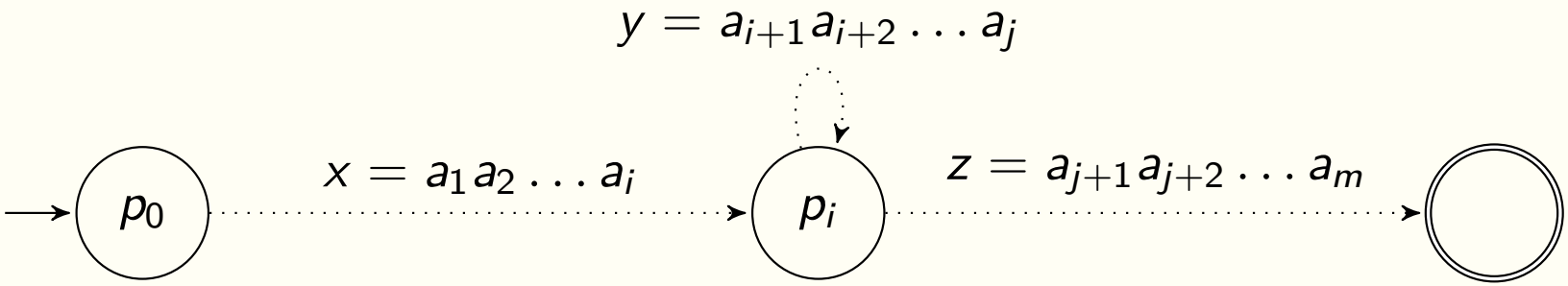
Automat A



Důkaz iteračního lematu pro regulární jazyky

Proof: iteračního lematu pro regulární jazyky

- Mějme regulární jazyk L, pak existuje DFA A s n stavy, že $L = L(A)$.
- Vezměme libovolné slovo $a_1 a_2 \dots a_m = w \in L$ délky $m \geq n$, $a_i \in \Sigma$.
- Definujme: $\forall i \ p_i = \delta^*(q_0, a_1 a_2 \dots a_i)$. Platí $p_0 = q_0$.
- Máme $n+1$ p_i a n stavů, některý se opakuje, vezměme první takový, tj. $(\exists i, j)(0 \leq i < j \leq n \ \& \ p_i = p_j)$.
- Definujme: $x = a_1 a_2 \dots a_i$, $y = a_{i+1} a_{i+2} \dots a_j$, $z = a_{j+1} a_{j+2} \dots a_m$, tj. $w = xyz$, $y \neq \epsilon$, $|xy| \leq n$.



- Smyčka nad p_i se může opakovat libovolně krát a vstup je také akceptovaný.

□

Použití pumping lemmatu

Example 2.12 (Pumping lemma jako hra s oponentem)

Jazyk $L_{eq} = \{w; |w|_0 = |w|_1\}$ slov se stejným počtem 0 a 1 není regulární.

Proof: Jazyk L_{eq} není regulární.

- Předpokládejme že L_{eq} je regulární. Vezměme n z pumping lemmatu.
- Zvolme $w = 0^n 1^n \in L_{eq}$.
- Rozdělme $w = xyz$ dle pumping lemmatu, $y \neq \epsilon$, $|xy| \leq n$.
- Protože $|xy| \leq n$ je na začátku w, obsahuje jen 0.
- Z pumping lemmatu: $xz \in L_{eq}$ (pro $k = 0$). To má ale méně 0 než 1, takže nemůže být v L_{eq}. □

Example 2.13

Jazyk $L = \{0^i 1^i; i \geq 0\}$ není regulární.

Aplikace pumping lemmatu 2

Example 2.14

Jazyk L_{pr} slov 1^p kde p je prvočíslo není regulární.

Proof: L_{pr} slov 1^p kde p je prvočíslo není regulární.

- Předpokládejme že L_{pr} je regulární. Vezměme n z pumping lemmatu. Zvolme prvočíslo $p \geq n+2$, označme $w = 1^p$.
- Rozložme $w = xyz$ dle pumping lemmatu, nechť $|y| = m$. Pak $|xz| = p - m$.
- $xy^{p-m}z \in L_{pr}$ z pumping lemmatu, ale $|xy^{p-m}z| = |xz| + (p-m)|y| = p - m + (p-m)m = (m+1)(p-m)$ není prvočíslo (žádný z činitelů není 1). □

'Pumpovatelný' ne-regulární jazyk

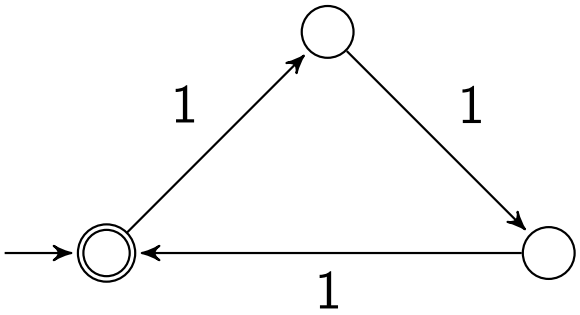
Example 2.15 (Ne–regulární jazyk, který lze pumpovat)

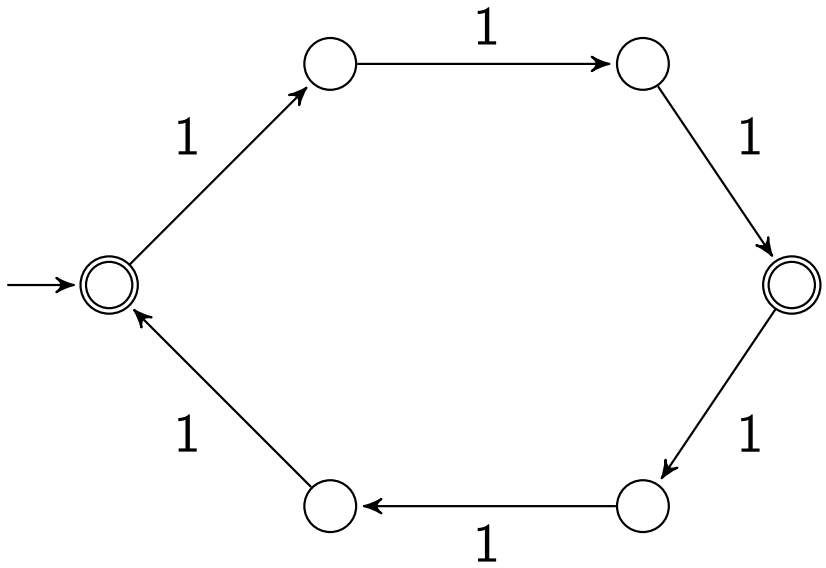
Jazyk $L = \{u | u = a^+ b^i c^i \vee u = b^i c^j, + \in \mathbb{N}_{>0}, i, j \in \mathbb{N}_0\}$ není regulární (Myhill–Nerodova věta), ale vždy lze pumpovat první písmeno.

Nejednoznačnost

Automat přijímající daný jazyk není určen jednoznačně.

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





Definition 2.8 (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 $\delta^*(q, w) \in F$.

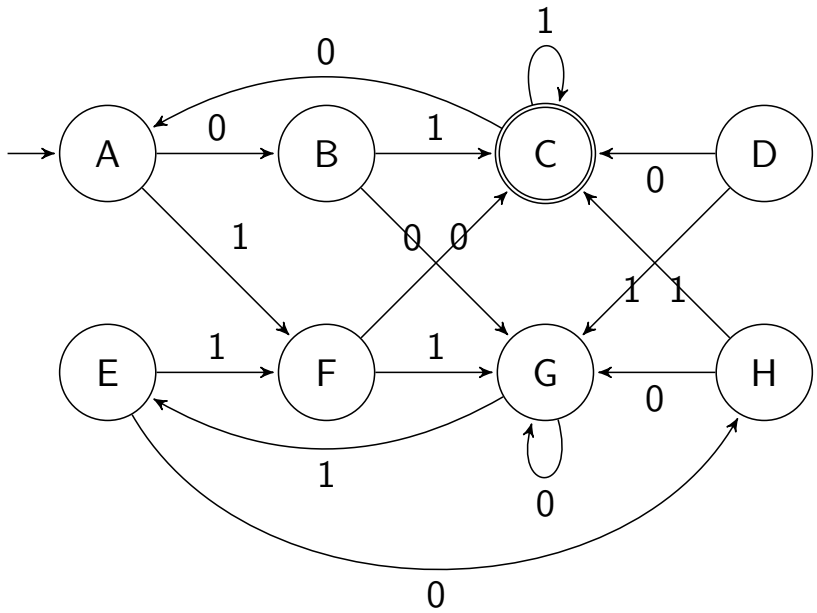
Pokud dva stavy nejsou ekvivalentní, říkáme, že jsou **rozlišitelné**.

Příklad rozlišitelných stavů

Example 2.16

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, $1*$ zřejmé, 0 vede do ne–přijímajících stavů, 01 a 00 se sejdou ve stejném stavu.



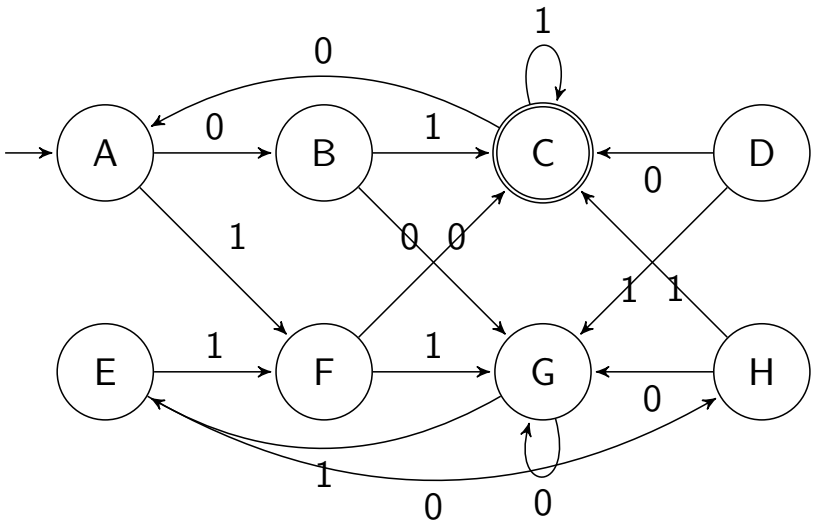
Lemma

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: Nechť $p, q \in Q$, $a \in \Sigma$ 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 $p, q \in Q$, $a \in \Sigma$.



	A	B	C	D	E	F	G
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

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

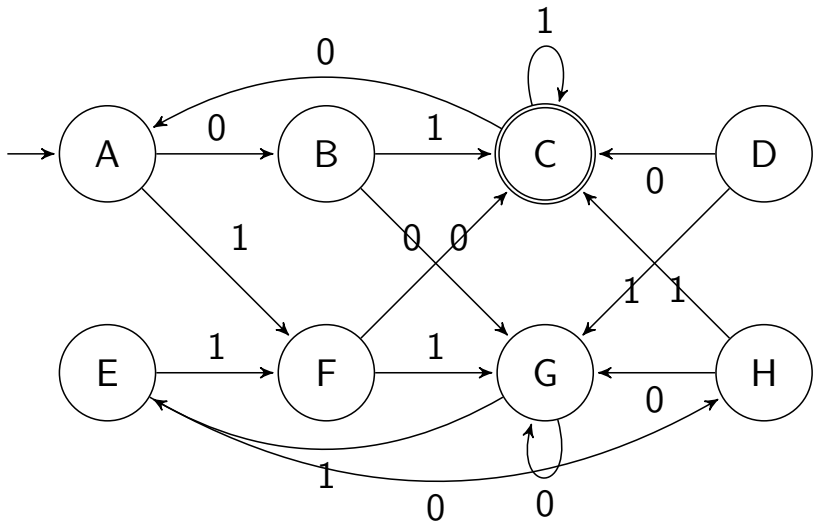
	A	B	C	D	E	F	G
B							
C	x	x					
D			x				
E			x				
F			x				
G			x				
H			x				

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

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

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

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



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

	A	B	C	D	E	F	G
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

Zůstávají tři ekvivalentní dvojice stavů.

Theorem 2.3

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

Proof: Koreknost 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 'zpátky'.

Testování rovnosti regulárních jazyků

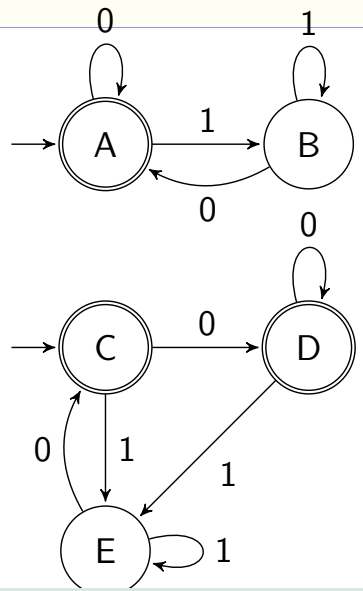
Algorithm: !Testování rovnost regulárních jazyků

Rovnost (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 stejné právě když počáteční stavy původních DFA jsou ekvivalentní.

Example 2.17

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

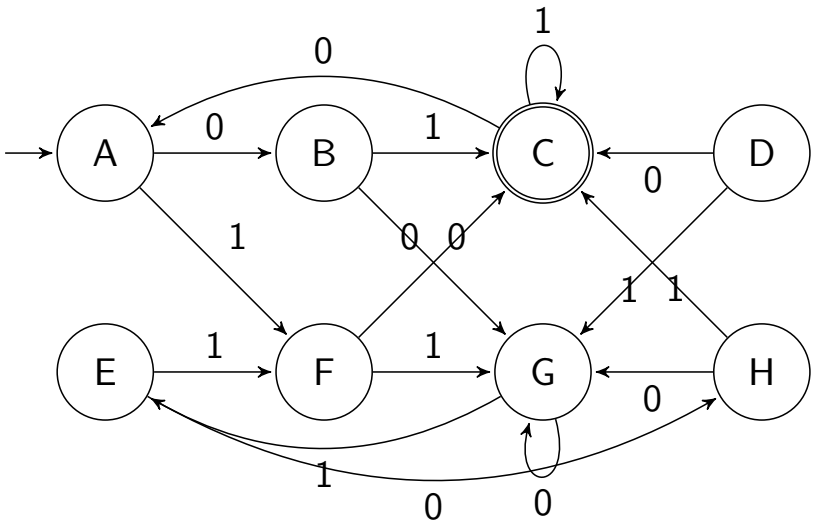
Definition 2.9 (redukovaný DFA, redukt)

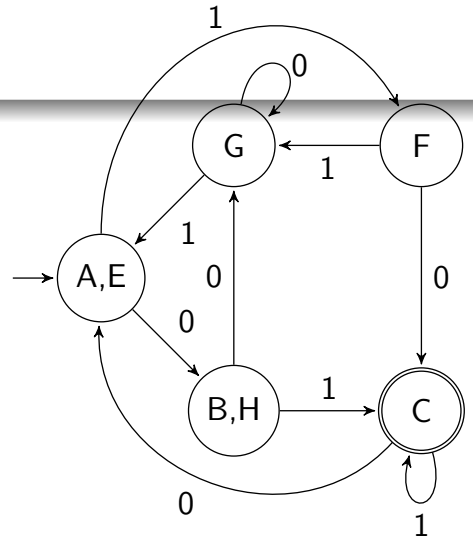
Deterministický konečný automat je **redukovaný**, pokud

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

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

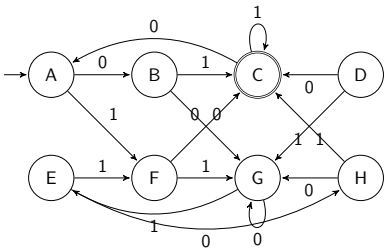
- B je redukovaný a
- A a B jsou ekvivalentní.

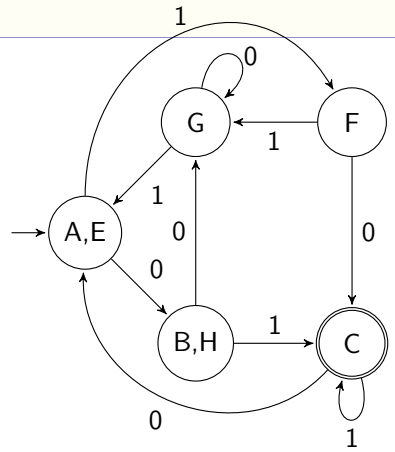




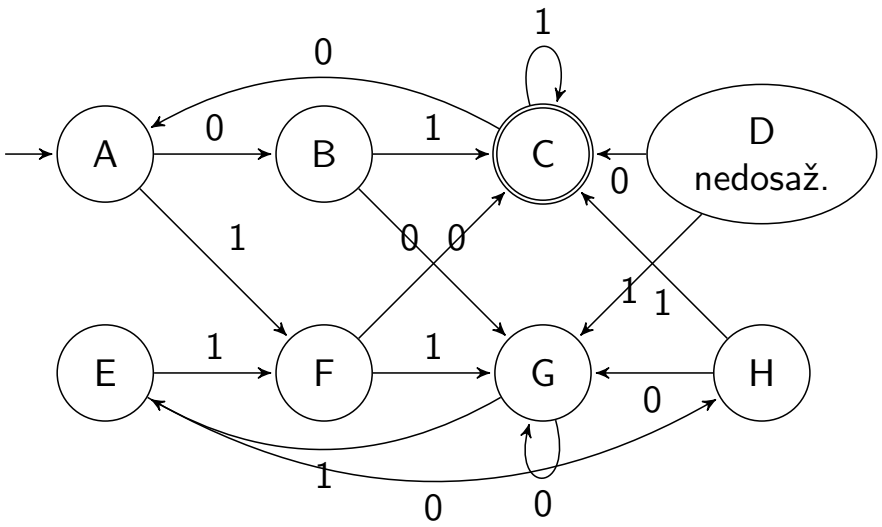
Algorithm: !Algoritmus nalezení reduktu DFA A

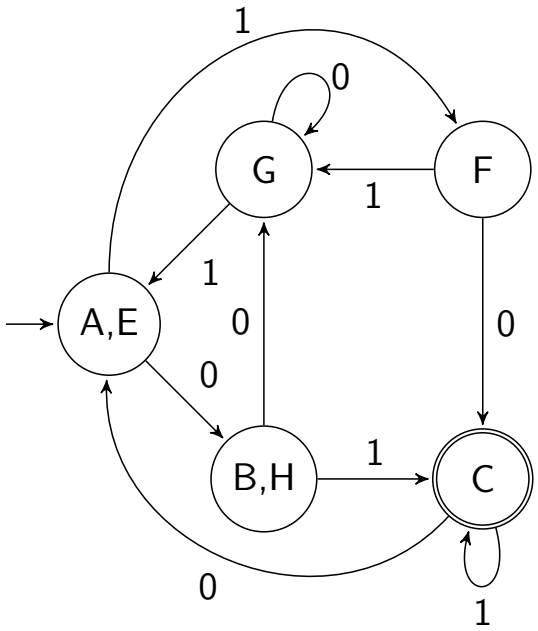
- Ze vstupního DFA A eliminujeme stavy nedosažitelné z počátečního stavu.
- Najdeme rozklad zbylých stavů na třídy ekvivalence.
- Konstruujeme DFA B na třídách ekvivalence jakožto stavech. Přechodovou funkci B označíme γ, mějme $S \in Q_B$. Pro libovolné $q \in S$, označíme T třídu ekvivalence $\delta(q, a)$ 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:

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

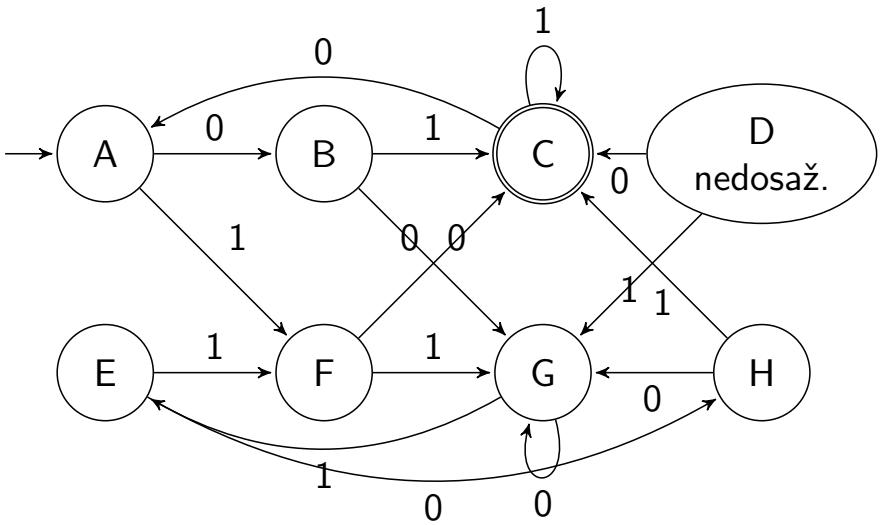
Automatový homomorfizmus

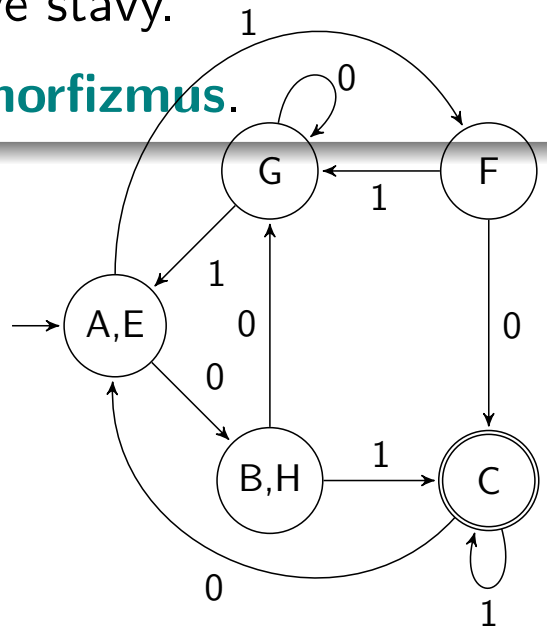
Definition 2.10 (automatový homomorfizmus)

Nechť A_1, A_2 jsou DFA. Řekneme, že zobrazení $h : Q_1 \to Q_2$ Q_1 na Q_2 je **automatovým homomorfizmem**, jestliže:

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

Homomorfizmus prostý a na nazýváme **isomorfizmus**.





Ekvivalence automatů a homomorfizmus

Definition 2.11 (Ekvivalence automatů!)

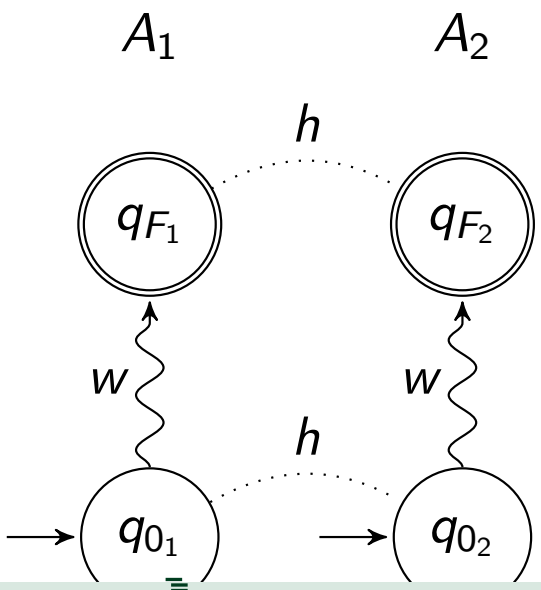
Dva konečné automaty A, B nad stejnou abecedou Σ jsou **ekvivalentní**, jestli že rozpoznávají stejný jazyk, tj. $L(A) = L(B)$.

Theorem 2.4 (Věta o ekvivalenci automatů)

Existuje-li homomorfizmus 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: $w \in L(A_1) \Leftrightarrow \delta_1^*(q_{0_1}, w) \in F_1$
  $\Leftrightarrow h(\delta_1^*(q_{0_1}, w)) \in F_2$
  $\Leftrightarrow \delta_2^*(h(q_{0_1}), w) \in F_2$
  $\Leftrightarrow \delta_2^*(q_{0_2}, w) \in F_2$
  $\Leftrightarrow w \in L(A_2)$



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 isomorfizmus.

□

Lemma

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

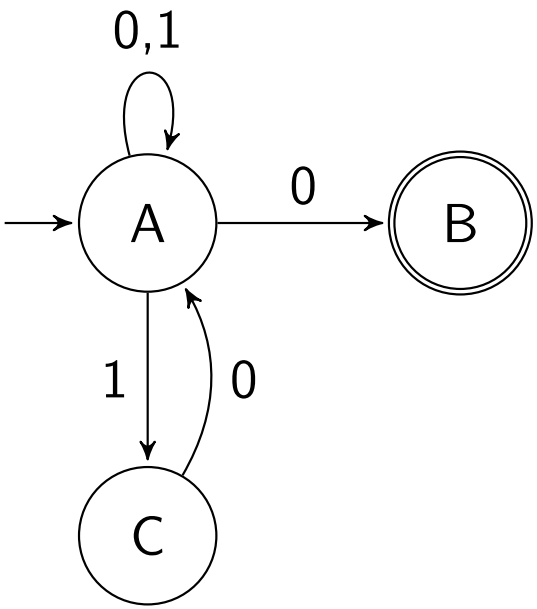
Definition 2.12 (Redukt)

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 2.18

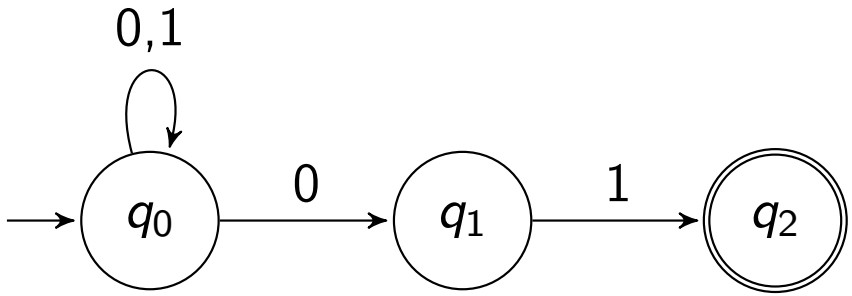
Nedeterministický FA na obrázku můžeme redukovat vypuštěním stavu C.
Stavy $\{A, C\}$ jsou rozlišitelné vstupem 0, takže algoritmus pro DFA redukci nenajde.
Mohli bychom hledat exhauzivním výpočtem.



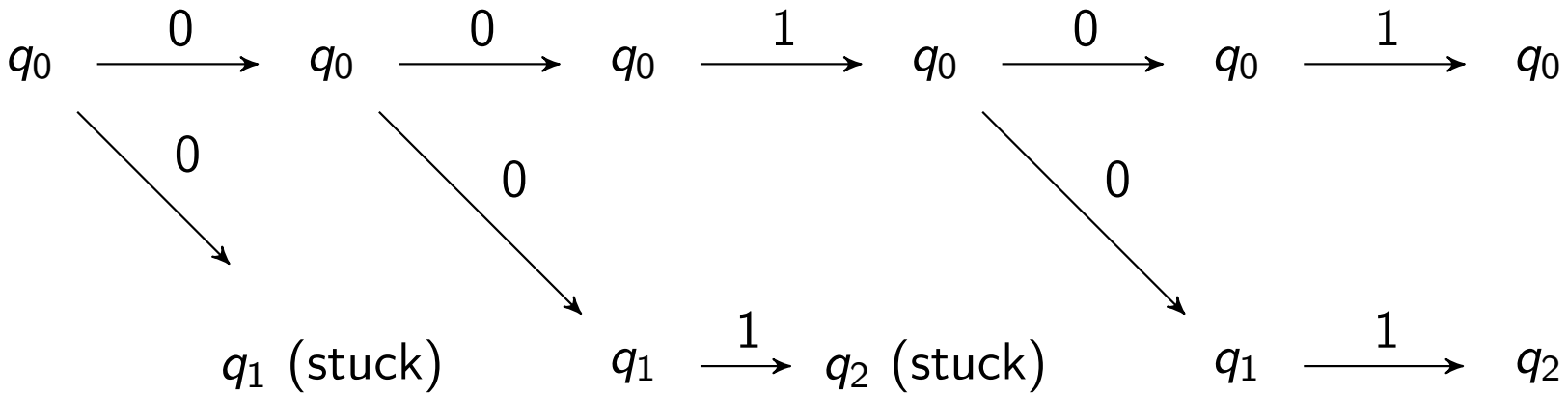
Nedeterministické konečné automaty s ϵ přechody (ϵ-NFA)

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 $\delta : Q \times (\Sigma \cup \{\epsilon\}) \to \mathcal{P}(Q)$ vracející podmnožinu Q.
4. **počáteční ho stavu**[a] $q_0 \in Q$,
5. a **množiny koncových (přijímajících) stavů** $F \subseteq Q$.

[a] alternativa: 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:

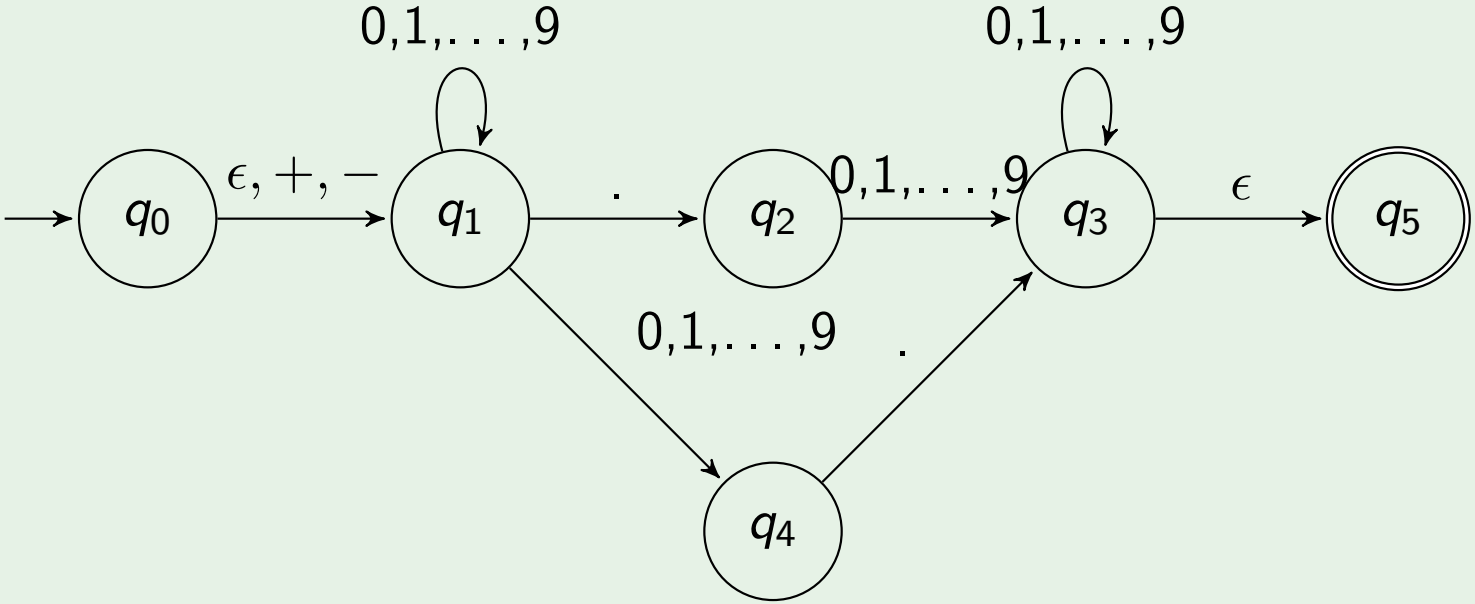
δ	ϵ	0	1
$\to q_0$	$\emptyset$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\emptyset$	$\{q_2\}$
$*q_2$	$\emptyset$	$\emptyset$	$\emptyset$

Konečné automaty s ϵ přechody

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

Example 3.2 (NFA s ϵ přechody)

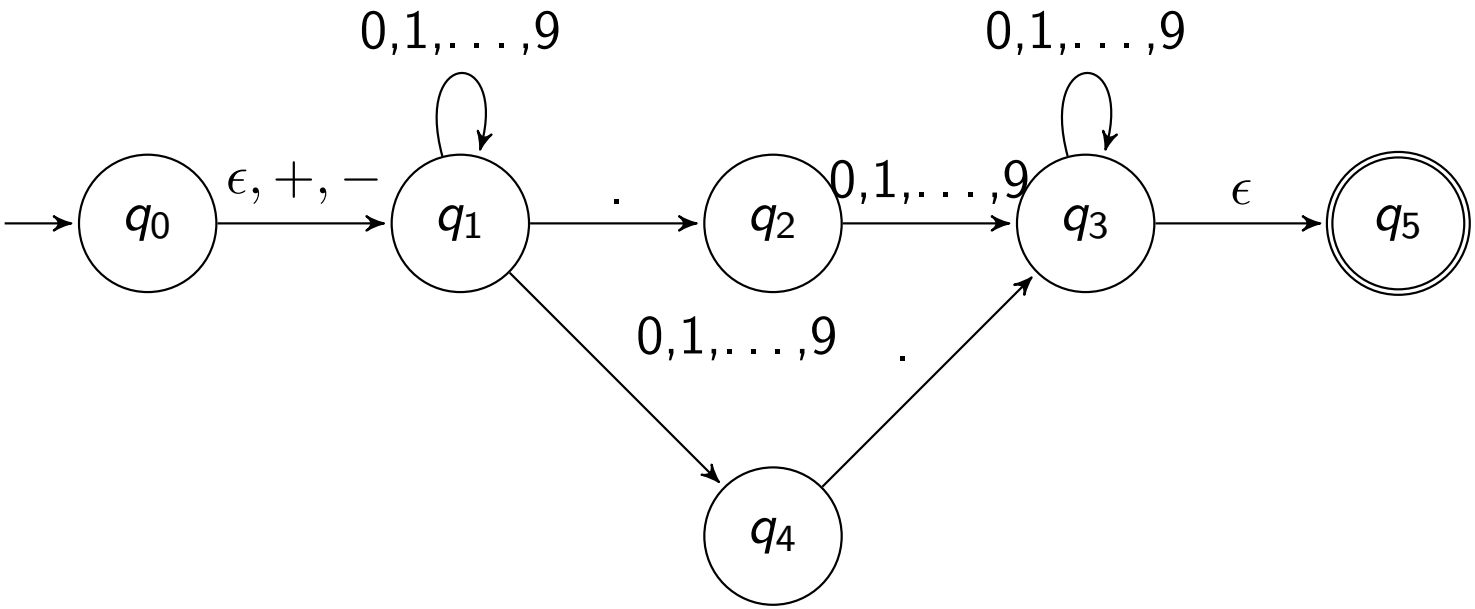
(1) Volitelně znaménko + nebo - ,
(2) řetězec číslic,
(3) desetinná tečka a
(4) další řetězec číslic. Nejméně jeden z řetězců (2) a (4) musí být neprázdný.



Example 3.3 (Přechodová funkce v tabulce)

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

δ	ϵ	+,-	.	0,1,. . . ,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–uzávěr

Definition 3.2 (ϵ–uzávěr)

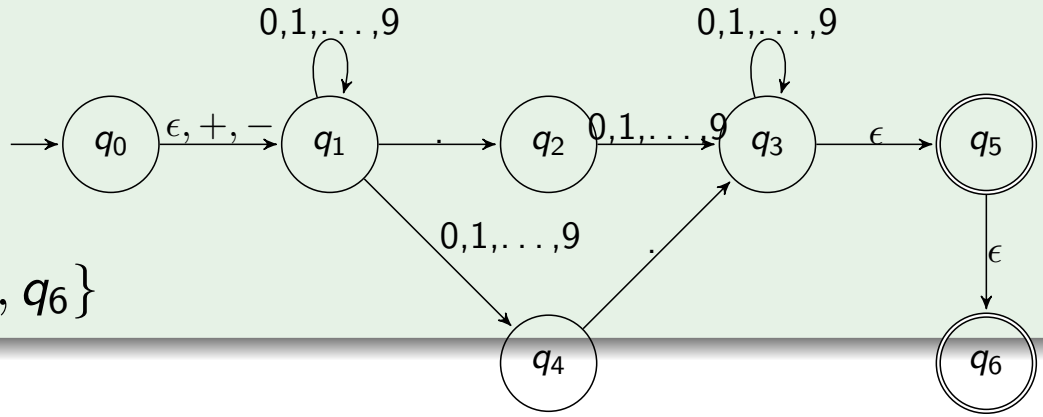
Pro $q \in Q$ definujeme **ϵ–uzávěr** $\epsilon closure(q)$ rekurzivně:

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

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

Example 3.4 (ϵ uzávěr)

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



Rozšířená přechodová funkce a jazyk přijímaný ϵ-NFA

Definition 3.3 (δ^* pro ϵNFA)

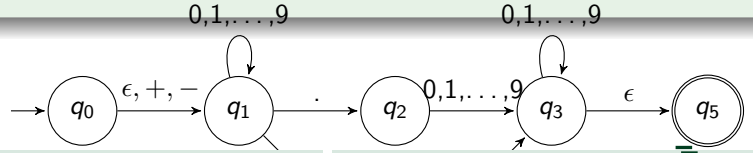
Nechť $E = (Q, \Sigma, \delta, q_0, F)$ je ϵ–NFA. Rozšířenou přechodovou funkci δ^* definujeme následovně:

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

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

Example 3.5

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



Jazyk přijímaný nedeterministickým konečným automatem s ϵ přechody

Definition 3.4 (Jazyk přijímaný ϵNFA)

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

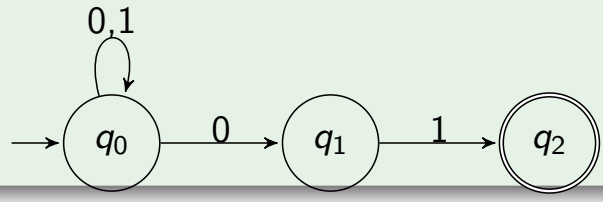
$$L(A) = \{w|\ \delta^*(q_0, w) \cap F \neq \emptyset\}$$

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 $L = \{w|w \text{ končí na } 01, w \in \{0,1\}^*\}$.
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.



Shrnutí druhé přednášky

- Iterační lemma pro regulární jazyky.
- Rekudce deterministického konečného automatu
  - dosažitelné stavy
  - ekvivalentní stavy
  - ekvivalentní automaty
  - redukt
  - automatový homomorfizmus
- ϵ–nedeterministické konečné automaty (úvod)

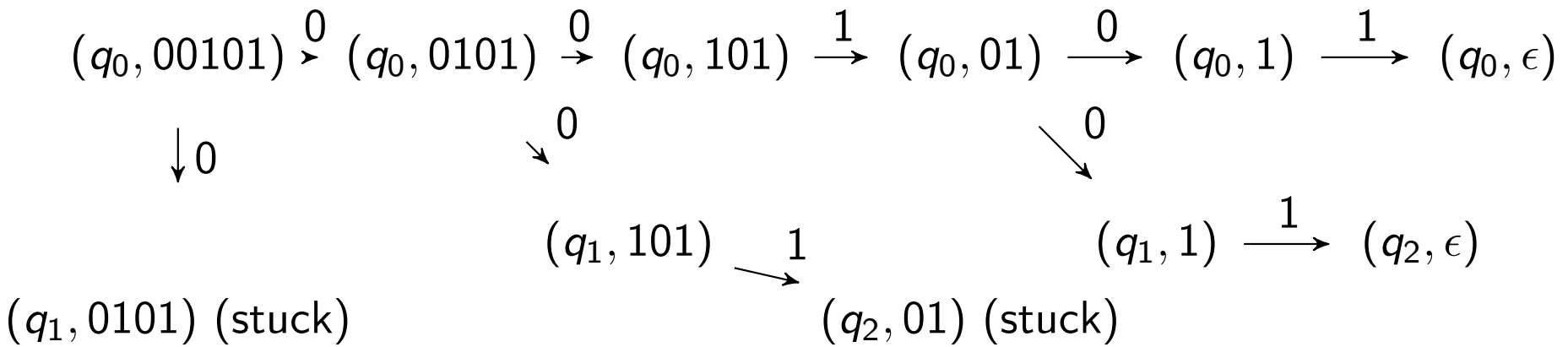
Konfigurace automatu, Výpočetní graf

Definition 3.5 (Konfigurace *DFA*, ϵ*NFA*)

Mějme ϵNFA $A = (Q, \Sigma, \delta, q_0, F)$, $q \in Q$, $v \in \Sigma^*$, pak dvojice (q, v) označuje **konfiguraci** konečného automatu, nacházejícího se ve stavu q s nepřečteným 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 $D = (Q_D, \Sigma, \delta_D, q_{D_0}, F_D)$ přijímající stejný jazyk jako N.

- Nové stavy jsou ϵuzavřené podmnožiny Q_N.

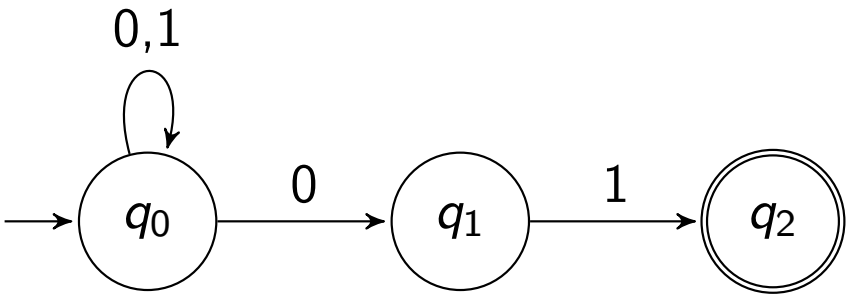
  $Q_D \subseteq \mathcal{P}(Q_N)$, $\forall S \subseteq Q_N : \epsilon closure(S) \in Q_D$. V Q_{D_0} může být i $\emptyset$.
- Počáteční stav je ϵuzávěr q_0.

  $q_D = \epsilon closure(q_0)$.
- Přijímací stavy jsou všechny množiny obsahující nějaký přijímající stav.

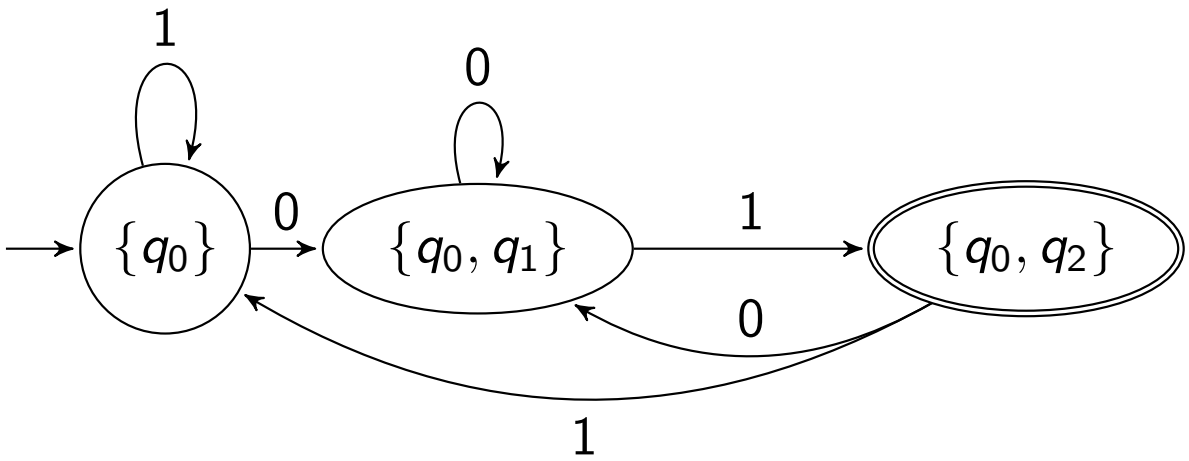
  $F_D = \{S | S \in Q_D \ \& \ S \cap F_N \neq \emptyset\}$.
- Přechodová funkce sjednotí předchody z jednotlivých stavů a uzavře ϵclosure.

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

Příklad podmnožinové konstrukce pro $\{w.01 | w \in \{0,1\}\}$



	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)

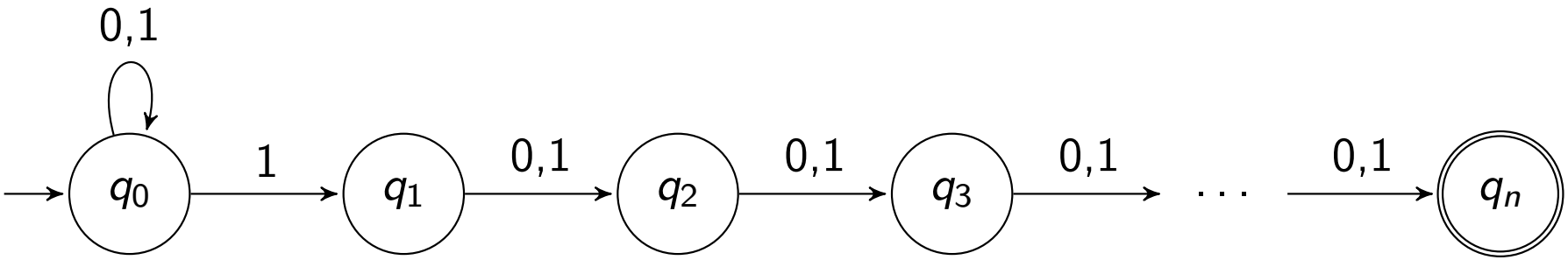
Pro DFA $D = (Q_D, \Sigma, \delta_D, q_{D_0}, F_D)$ *vytvořený podmnožinovou konstrukcí z NFA* $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ *platí* $L(N) = L(D)$.

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

Množinové operace nad jazyky

Definition 3.7 (Množinové operace nad jazyky)

Mějme dva jazyky L, M. Definujme následující operace:

(1) binární **sjednocení** $L \cup M = \{w | w \in L \vee w \in M\}$
- Příklad: jazyk obsahuje slova začínající a^i nebo tvaru $b^j c^j$.

(2) **průnik** $L \cap M = \{w | w \in L \ \& \ w \in M\}$
- Příklad: jazyk obsahuje slova sudé délky končící na 'baa'.

(3) **rozdíl** $L - M = \{w | w \in L \ \& \ w \notin M\}$
- Příklad: jazyk obsahuje slova sudé délky nekončící na 'baa'.

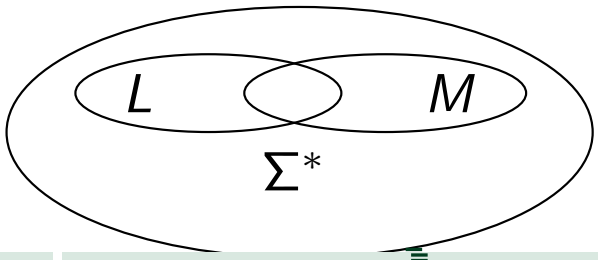
(4) **doplněk (komplement)** $\overline{L} = -L = \{w | w \notin L\} = \Sigma^* - L$
- Příklad: jazyk obsahuje slova nekončící na 'a'.

Theorem (de Morganova pravidla)

Platí:

$$L \cap M = \overline{\overline{L} \cup \overline{M}}$$
$$L \cup M = \overline{\overline{L} \cap \overline{M}}$$
$$L - M = L \cap \overline{M}.$$

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:

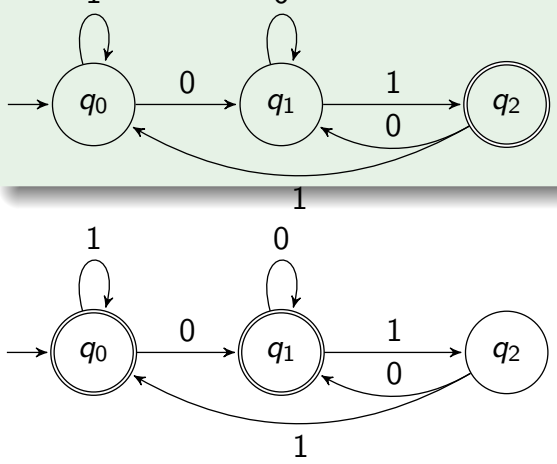
(1) *sjednocení* $L \cup M$,
(2) *průnik* $L \cap M$,
(3) *rozdíl* $L - M$,
(4) *doplněk* $\bar{L} = \Sigma^* - L$.

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, tj. $F_{complement} = Q_A - F_A$. □

Example 3.8

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



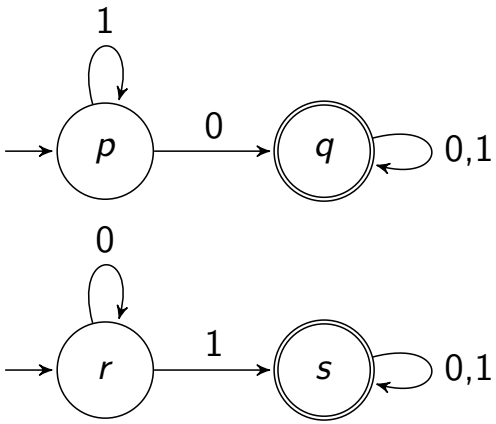
Konstrukce součinu automatů

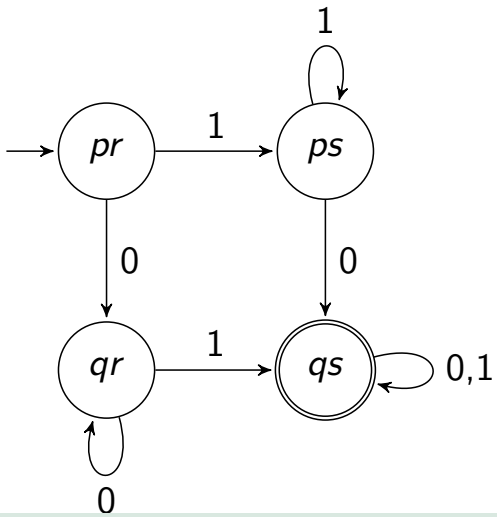
Proof: Průnik, sjednocení, rozdíl

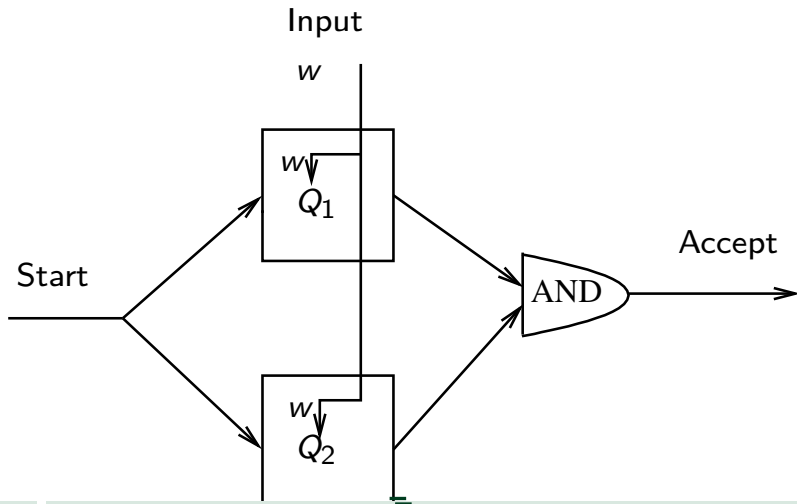
- Pro sjednocení a rozdíl doplníme funkci δ na totální.
- Zkonstruujeme součinový automat,
  $Q = (Q_1 \times Q_2, \Sigma, \delta([p_1, p_2], x) = [\delta_1(p_1, x), \delta_2(p_2, x)], (q_{0_1}, q_{0_2}), F)$
- průnik: $F = F_1 \times F_2$
- sjednocení: $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$
- 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.
- $L_1 = \{w | w \in \{0,1\}^* \& |w|_1 = 3k + 2\}$
- $L_2 = \{w | u, v \in \{0,1\}^* \& w = u11v\}$
- $L = L_1 - L_2$.

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 sestrojit 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	$L.M = \{uv \mid u \in L \ \& \ v \in M\}$
	$L.x = L.\{x\}$ a $x.L = \{x\}.L$ pro $x \in \Sigma$
mocniny jazyka	$L^0 = \{\epsilon\}$
	$L^{i+1} = L^i.L$
pozitivní iterace	$L^+ = L^1 \cup L^2 \ldots = \bigcup_{i \geq 1} L^i$
obecná iterace	$L^* = L^0 \cup L^1 \cup L^2 \ldots = \bigcup_{i \geq 0} L^i$
	tedy $L^* = L^+ \cup \{\epsilon\}$
otočení jazyka	$L^R = \{u^R \mid u \in L\}$
(=zrcadlový obraz,reverze)	$(x_1 x_2 \ldots x_n)^R = x_n x \ldots x_2 x_1$
levý kvocient L podle M	$M \setminus L = \{v \mid uv \in L \ \& \ u \in M\}$
levá derivace L podle w	$\partial_w L = \{w\} \setminus L$ (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
	$= \{\delta_1(q, x), q_2\}$	pro $q \in Q_1$ & $\delta_1(q, x) \in F_1$	nedet. přechod z A_1
	$= \{\delta_2(q, x)\}$	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; nederministický 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:

- $\delta_B(q, x) = \{p | \delta(p, x) = q\}$ pro $q \in Q$
- $\delta_B(q_B, \epsilon) = F$, $\delta_B(q_B, x) = \emptyset$.
- Pro libovolné slovo $w = x_1 x_2 \dots x_n$
  - $q_0, q_1, q_2, \dots, q_n$ 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. □

Pozn. Někdy L nebo L^R má výrazně jednodušší přijímající automat.

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:

- Vezmeme DFA $A = (Q, \Sigma, \delta, q_0, F)$, tak že $L = L(A)$.
  Definujeme nedeterministický NFA $B = (Q \cup \{q_\epsilon\}, \Sigma, \delta, q_\epsilon, F)$ kde:
- a přidáme přechod $\delta(q_\epsilon, \epsilon) = \{q | q \in Q \ \& \ (\exists u \in M)\ q = \delta^*(q_0, u)\}$ do stavů, kam lze dojit slovem z M.
  - lze nalézt algoritmicky
    $(\{q;\ L(A_q) \cap M \neq \emptyset$ kde $A_q = (Q, \Sigma, \delta, q_0, \{q\})\})$
- $v \in M \setminus L$
  - $\Leftrightarrow$ $(\exists u \in M)\ uv \in L$
  - $\Leftrightarrow$ $(\exists u \in M, \exists q \in Q)\ \delta(q_0, u) = q \ \& \ \delta(q, v) \in F$
  - $\Leftrightarrow$ $\exists q \in \delta(q_\epsilon, \epsilon) \ \& \ \delta(q, v) \in F$
  - $\Leftrightarrow$ $v \in L(B)$.

$L/M = (M^R \setminus L^R)^R$. □