Minimalizace DKA

Eliminace nedosažitelných stavů

Definice 2.1 Nechť $M = (Q, \Sigma, \delta, q_0, F)$ je konečný automat. Stav $q \in Q$ nazveme dosažitelný, pokud existuje $w \in \Sigma^*$ takové, že $(q_0, w) \overset{*}{\underset{M}{\vdash}} (q, \varepsilon)$. Stav je nedosažitelný, pokud není dosažitelný.

Algoritmus 2.1 Eliminace nedosažitelných stavů
Vstup: DKA $M = (Q, \Sigma, \delta, q_0, F)$.
Výstup: DKA M' bez nedosažitelných stavů, $L(M) = L(M')$.
Metoda:

1. i := 0
2. S_i := $\{q_0\}$
3. `repeat`
4. $\quad S_{i+1}$:= $S_i \cup \{q \mid \exists p \in S_i \; \exists a \in \Sigma : \delta(p, a) = q\}$
5. $\quad i$:= i + 1
6. `until` $S_i = S_{i-1}$
7. M' := $(S_i, \Sigma, \delta_{|S_i}, q_0, F \cap S_i)$

Jazykově nerozlišitelné stavy

Definice 2.2

- Nechť $M = (Q, \Sigma, \delta, q_0, F)$ je úplně definovaný DKA. Říkáme, že řetězec $w \in \Sigma^*$ rozlišuje q_1, q_2, jestliže $(q_1, w) \overset{*}{\underset{M}{\vdash}} (q_3, \varepsilon) \wedge (q_2, w) \overset{*}{\underset{M}{\vdash}} (q_4, \varepsilon)$ pro nějaké q_3, q_4 a *právě jeden* ze stavů q_3, q_4 je v F.
- Říkáme, že stavy $q_1, q_2 \in Q$ jsou k-nerozlišitelné a píšeme $q_1 \overset{k}{\equiv} q_2$, právě když neexistuje $w \in \Sigma^*$, $|w| \leq k$, který rozlišuje q_1 a q_2.
- Stavy q_1, q_2 jsou nerozlišitelné, značíme $q_1 \equiv q_2$, jsou-li pro každé $k \geq 0$ k-nerozlišitelné.

❖ Poznámka: Dá se snadno dokázat, že $\equiv$ je relací ekvivalence na Q, tj. relací, která je reflexivní, symetrickou a tranzitivní.

Definice 2.3 Úplně definovaný DKA M nazýváme redukovaný, jestliže žádný stav z Q není nedostupný a žádné dva stavy nerozlišitelné.

Věta 2.1 Nechť $M = (Q, \Sigma, \delta, q_0, F)$ je úplně definovaný DKA a $|Q| = n$, $n \geq 2$. Platí $\forall q_1, q_2 \in Q : q_1 \equiv q_2 \Leftrightarrow q_1 \overset{n-2}{\equiv} q_2$.

Důkaz. „$\Rightarrow$“ triviální, ukážeme „$\Leftarrow$“:

1. Jestliže $|F| = 0$ nebo $|F| = n$, pak platí $q_1 \overset{n-2}{\equiv} q_2 \;\Rightarrow\; q_1 \equiv q_2$.
2. Nechť $|F| > 0 \wedge |F| < n$. Ukážeme, že platí $\equiv\, = \overset{n-2}{\equiv} \subseteq \overset{n-3}{\equiv} \subseteq \ldots \subseteq \overset{1}{\equiv} \subseteq \overset{0}{\equiv}$:
   - Zřejmě platí:
     (a) $\forall q_1, q_2 \in Q : q_1 \overset{0}{\equiv} q_2 \Leftrightarrow (q_1 \in F \wedge q_2 \in F) \vee (q_1 \notin F \wedge q_2 \notin F)$, tj.
     $q_1 \overset{0}{\equiv} q_2 \Leftrightarrow (q_1 \in F \Leftrightarrow q_2 \in F)$.
     (b) $\forall q_1, q_2 \in Q\ \forall k \geq 1 : q_1 \overset{k}{\equiv} q_2 \Leftrightarrow (q_1 \overset{k-1}{\equiv} q_2 \wedge \forall a \in \Sigma : \delta(q_1, a) \overset{k-1}{\equiv} \delta(q_2,$
   - Relace $\overset{0}{\equiv}$ je ekvivalencí určující rozklad $\{F, Q \setminus F\}$.
   - Je-li $\overset{k+1}{\equiv} \neq \overset{k}{\equiv}$, pak $\overset{k+1}{\equiv}$ je vlastním zjemněním $\overset{k}{\equiv}$, tj. obsahuje alespoň o jednu třídu více než rozklad $\overset{k}{\equiv}$.
   - Jestliže pro nějaké k platí $\overset{k+1}{\equiv} = \overset{k}{\equiv}$, pak také $\overset{k+1}{\equiv} = \overset{k+2}{\equiv} = \overset{k+3}{\equiv} = \ldots$ podle (b) a tedy $\overset{k}{\equiv}$ je hledaná ekvivalence.
   - Protože F nebo $Q \setminus F$ obsahuje nejvýše $n - 1$ prvků, získáme relaci $\equiv$ po nejvýše $n - 2$ zjemněních $\overset{0}{\equiv}$.

$\square$

Převod na redukovaný DKA

Algoritmus 2.2 Převod na redukovaný DKA

Vstup: Úplně definovaný DKA $M = (Q, \Sigma, \delta, q_0, F)$.

Výstup: Redukovaný DKA $M' = (Q', \Sigma, \delta', q_0', F')$, $L(M) = L(M')$.

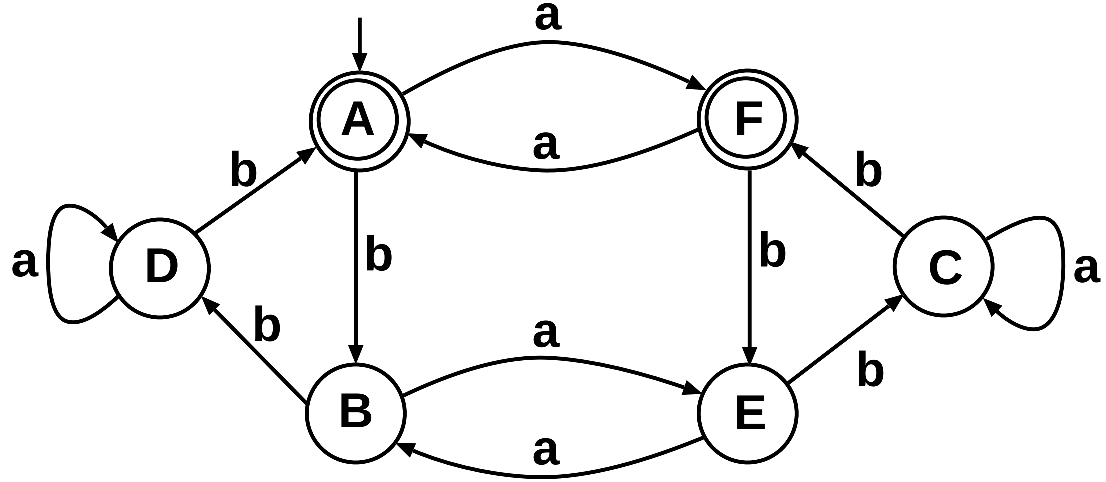
Metoda:

1. Odstraň nedostupné stavy s využitím alg. 2.1.
2. i := 0
3. $\overset{0}{\equiv}$:= $\{(p, q) \mid p \in F \Longleftrightarrow q \in F\}$
4. `repeat`
5. $\quad\overset{i+1}{\equiv}$:= $\{(p, q) \mid p \overset{i}{\equiv} q \wedge \forall a \in \Sigma : \delta(p, a) \overset{i}{\equiv} \delta(q, a)\}$
6. $\quad i$:= i + 1
7. `until` $\overset{i}{\equiv} = \overset{i-1}{\equiv}$
8. Q' := $Q / \overset{i}{\equiv}$
9. $\forall p, q \in Q \ \forall a \in \Sigma : \delta'([p], a) = [q] \Leftrightarrow \delta(p, a) = q$
10. $q_0' = [q_0]$
11. $F' = \{[q] \mid q \in F\}$

❖ *Poznámka:* Výraz $[x]$ značí ekvivalenční třídu určenou prvkem x.

Příklad minimalizace DKA

Příklad 2.1 Převeďte níže uvedený DKA (zadaný diagram přechodů) na odpovídající redukovaný DKA.



$\overset{0}{\equiv}$	δ	a	b
I:	A	F_I	B_{II}
	F	A_I	E_{II}
II:	B	E_{II}	D_{II}
	C	C_{II}	F_I
	D	D_{II}	A_I
	E	B_{II}	C_{II}

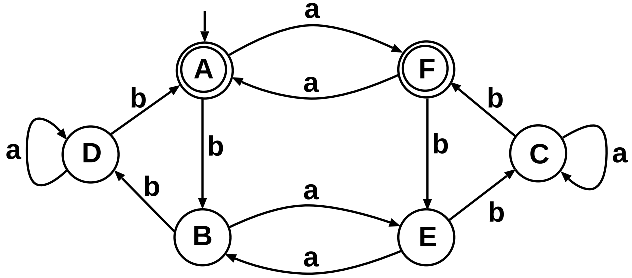
1. Neobsahuje nedostupné stavy.

3. $\overset{0}{\equiv} = \{\{A, F\}, \{B, C, D, E\}\}$

5.1. $\overset{1}{\equiv} = \{\{A, F\}, \{B, E\}, \{C, D\}\}$

Pokračuje na druhé straně...

Pro zopakování automat z předchozího slajdu, v jehož minimalizaci níže pokračujeme:

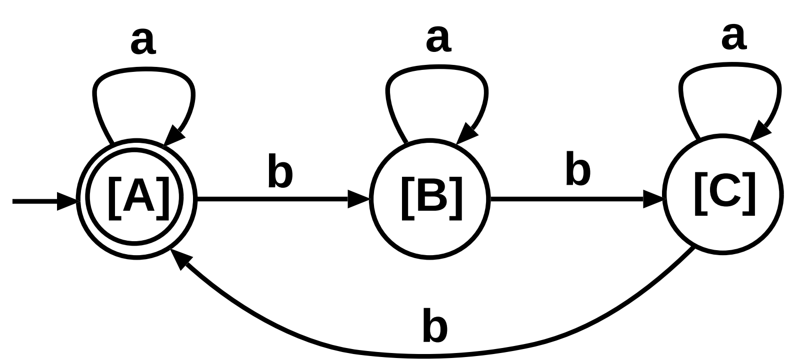


$\overset{1}{\equiv}$	δ	a	b
I:	A	F_I	B_{II}
	F	A_I	E_{II}
II:	B	E_{II}	D_{III}
	E	B_{II}	C_{III}
III:	C	C_{III}	F_I
	D	D_{III}	A_I

5.2. $\overset{2}{\equiv} = \{\{A, F\}, \{B, E\}, \{C, D\}\} = \overset{1}{\equiv} = \equiv$

8. $Q' = \{[A], [B], [C]\}$, kde $[A] = \{A, F\}$, $[B] = \{B, E\}$, $[C] = \{C, D\}$

9–11.



Regulární množiny a výrazy

Regulární množiny

Definice 2.4 Nechť Σ je konečná abeceda. Regulární množinu nad Σ definujeme rekurzívně takto:

1. $\emptyset$ (tj. prázdná množina) je regulární množina nad Σ,
2. $\{\varepsilon\}$ je regulární množina nad Σ,
3. $\{a\}$ je regulární množina nad Σ pro všechny $a \in \Sigma$,
4. jsou-li P a Q regulární množiny nad Σ, pak také
   (a) $P \cup Q$,
   (b) $P.Q$,
   (c) P^*

   jsou regulární množiny nad Σ.
5. Žádné jiné množiny, než ty, které lze získat pomocí výše uvedených pravidel, nejsou regulárními množinami.

Příklad 2.2 $L = (\{a\} \cup \{d\}).(\{b\}^*).\{c\}$ je regulární množina nad $\Sigma = \{a, b, c, d\}$.

Regulární výrazy

Definice 2.5 Regulární výrazy nad Σ a regulární množiny, které označují, jsou rekurzívně definovány takto:

1. $\emptyset$ je regulární výraz označující regulární množinu $\emptyset$,
2. ε je regulární výraz označující regulární množinu $\{\varepsilon\}$,
3. a je regulární výraz označující regulární množinu $\{a\}$ pro všechny $a \in \Sigma$,
4. jsou-li p, q regulární výrazy označující regulární množiny P a Q, pak
   (a) $(p + q)$ je regulární výraz označující regulární množinu $P \cup Q$,
   (b) (pq) je regulární výraz označující regulární množinu $P.Q$,
   (c) (p^*) je regulární výraz označující regulární množinu P^*.
5. Žádné jiné regulární výrazy nad Σ neexistují.

❖ Konvence:

1. Regulární výraz p^+ značí regulární výraz pp^*.
2. Abychom minimalizovali počet používaných závorek, stanovujeme priority operátorů:
    1. $*, +$ (iterace – nejvyšší priorita),
    2. $.$ (konkatenace),
    3. $+$ (alternativa).

Příklad 2.3

1. 01 odpovídá $\{01\}$.
2. 0^* odpovídá $\{0\}^*$.
3. $(0+1)^*$ odpovídá $\{0,1\}^*$.
4. $(0+1)^*011$ značí množinu řetězců nad $\{0,1\}$ končících 011.
5. $(a+b)(a+b+0+1)^*(0+1)$ značí množinu řetězců nad $\{a,b,0,1\}$, které začínají symbolem a nebo b a končí symbolem 0 nebo 1.

Kleeneho algebra

Definice 2.6 Kleeneho algebra sestává z neprázdné množiny se dvěma význačnými konstantami 0 a 1, dvěma binárními operacemi $+$ a . a unární operací $*$, které splňují následující axiomy:

$a + (b + c) = (a + b) + c$	asociativita $+$	[A.1]
$a + b = b + a$	komutativita $+$	[A.2]
$a + a = a$	idempotence $+$	[A.3]
$a + 0 = a$	0 je identitou pro $+$	[A.4]
$a(bc) = (ab)c$	asociativita .	[A.5]
$a1 = 1a = a$	1 je identitou pro .	[A.6]
$a0 = 0a = 0$	0 je anihilátorem pro .	[A.7]
$a(b + c) = ab + ac$	distributivita zleva	[A.8]
$(a + b)c = ac + bc$	distributivita zprava	[A.9]
$1 + aa^* = a^*$		[A.10]
$1 + a^*a = a^*$		[A.11]
$b + ac \leq c \Rightarrow a^*b \leq c$		[A.12]
$b + ca \leq c \Rightarrow ba^* \leq c$		[A.13]

V A.12 a A13 reprezentuje $\leq$ uspořádání definované takto: $a \leq b \stackrel{def}{\Longleftrightarrow} a + b = b$.

❖ Příklady Kleeneho algeber:

- Třída 2^{Σ^*} všech podmnožin Σ^* s konstantami $\emptyset$ a $\{\varepsilon\}$ a operacemi $\cup$, . a $*$.
- Třída všech regulárních podmnožin Σ^* s konstantami $\emptyset$ a $\{\varepsilon\}$ a operacemi $\cup$, . a $*$.
- Třída všech binárních relací nad množinou X s konstantami v podobě prázdné relace a identity a $\cup$, kompozicí (součinem) binárních relací a reflexivním tranzitivním uzávěrem binární relace jako operacemi.
- Matice nad Kleeneho algebrami.

❖ Ukázka platnosti A.2 pro regulární množiny:

- Nechť p, resp. q, označují reg. množiny P, resp. Q.
- Pak $p + q$ označuje $P \cup Q$ a $q + p$ označuje $Q \cup P$.
- $P \cup Q = Q \cup P$ (komutativita množinového sjednocení) $\Rightarrow p + q = q + p$.

❖ Poznámka: Axiomy A.12 a A.13 lze nahradit následujícími ekvivalentními vztahy:

$ac \leq c \Rightarrow a^*c \leq c$ [A.14]

$ca \leq c \Rightarrow ca^* \leq c$ [A.15]

Důkaz. Viz D. Kozen. *A Completeness Theorem for Kleene Algebras and the Algebra of Regular Events*. Technical Report TR 90-1123, Dept. of Comp. Sci., Cornell University, Ithaca, NY, USA, 1990. Dostupné na Internetu: odkaz viz stránky kurzu. □

❖ Některé užitečné teorémy Kleeneho algebry, které lze odvodit z jejich axiómů:

$$0^* = 1$$

$$1 + a^* = a^*$$

$$a^* = a + a^*$$

$$a^*a^* = a^*$$

$$a^{**} = a^*$$

$(a^*b)^*a^* = (a+b)^*$ pravidlo „vynořování“ [R.16]

$a(ba)^* = (ab)^*a$ pravidlo posuvu [R.17]

$$a^* = (aa)^* + a(aa)^*$$

❖ Další vlastnosti Kleeneho algeber, které lze odvodit z uvedených axiómů:

- $\leq$ je neostrým částečným uspořádáním:
  - $\leq$ je reflexivní ($a \leq a$),
  - $\leq$ je tranzitivní ($a \leq b \wedge b \leq c \Rightarrow a \leq c$),
  - $\leq$ je antisymetrické ($a \leq b \wedge b \leq a \Rightarrow a = b$). [a]

- $a + b$ je supremum (nejmenší horní omezení – least upper bound) a a b vůči $\leq$.

- $\leq$ je monotónní vůči všem operátorům:
  - $a \leq b \Rightarrow ac \leq bc \wedge ca \leq cb$,
  - $a \leq b \Rightarrow a + c \leq b + c$,
  - $a \leq b \Rightarrow a^* \leq b^*$.

---

[a] Snadno se samozřejmě také ukáže, že $a = b \Rightarrow a \leq b \wedge b \leq a$.

Důkaz. Příklady důkazů uvedených vlastností – ostatní viz např. D. Kozen. *A Completeness Theorem for Kleene Algebras and the Algebra of Regular Events* nebo *Automata and Computability*:

❖ $0^* = 1$: $0^* \stackrel{A.10}{=} 1 + 00^* \stackrel{A.7}{=} 1 + 0 \stackrel{A.4}{=} 1$.

❖ $1 + a^* = a^*$ – pro stručnost neuvádíme použití A.1, A.2:

- $1 + a^* \leq a^*$:
  - A.10: $a^* = 1 + aa^*$
  - A.3: $a^* + a^* = 1 + aa^*$
  - A.10: $1 + aa^* + a^* = 1 + aa^*$
  - A.3: $1 + 1 + aa^* + a^* = 1 + aa^*$, neboli $1 + a^* + 1 + aa^* = 1 + aa^*$
  - def. $\leq$: $1 + a^* \leq 1 + aa^*$
  - A.10: $1 + a^* \leq a^*$
- $a^* \leq 1 + a^*$:
  - $1 + a^* = 1 + a^*$
  - A.3: $1 + a^* + a^* = 1 + a^*$
  - def. $\leq$: $a^* \leq 1 + a^*$
- antisymetrie $\leq$.

- Poznámka: Různé vlastnosti Kleeneho algeber se někdy snáze dokazují pro jednotlivé konkrétní příklady těchto algeber, např. pro Kleeneho algebru regulárních výrazů, kde lze např. využít vazby na teorii množin.

- Vlastnosti Kleeneho algeber umožňují snadno řešit systémy lineárních rovnic nad těmito algebrami. V další části budeme s těmito rovnicemi pracovat již přímo nad Kleeneho algebrou regulárních výrazů.

Rovnice nad regulárními výrazy

Definice 2.7 Rovnice, jejímiž složkami jsou koeficienty a neznámé, které reprezentují (dané a hledané) regulární výrazy, nazýváme rovnicemi nad regulárními výrazy.

Příklad 2.4 Uvažujme rovnici nad regulárními výrazy nad abecedou $\{a, b\}$

$$X = aX + b$$

Jejím řešením je regulární výraz $X = a^*b$.

Důkaz.

- $LS = a^*b$
- $PS = a(a^*b) + b = a^+b + b = (a^+ + \varepsilon)b = a^*b$.

$\square$

❖ Ne vždy existuje jediné řešení rovnice nad reg. výrazy.

Věta 2.2 Nechť $X = pX + q$ je rovnice nad reg. výrazy, kde p, q jsou reg. výrazy a p označuje regulární množinu P takovou, že $\varepsilon \in P$. Pak

$$X = p^*(q + r)$$

je řešením této rovnice pro libovolné r (kterému nemusí ani odpovídat regulární množina, ale případně i obecnější jazyk).

Důkaz.

- $PS = p^*(q + r)$
- $LS = p(p^*(q + r)) + q = pp^*(q + r) + q = p^*(q + r) + q = p^*(q + r)$ (Uvědomme si, že $\varepsilon \in P$.)

$\square$

❖ Obvykle ale hledáme „nejmenší řešení“, tzv. nejmenší pevný bod, dané rovnice.

Věta 2.3 Nejmenším pevným bodem rovnice $X = pX + q$ je:

$$X = p^*q$$

Důkaz.

- $PS = p^*q$
- $LS = pp^*q + q = (pp^* + \varepsilon)q = p^*q$
- Minimalita plyne přímo z A.12.

$\square$

Soustavy rovnic nad regulárními výrazy

Příklad 2.5 Budiž dána soustava rovnic

$$\begin{aligned} X &= a_1X + a_2Y + a_3 \\ Y &= b_1X + b_2Y + b_3 \end{aligned}$$

Její řešení je:

$$\begin{aligned} X &= (a_1 + a_2b_2^*b_1)^*(a_3 + a_2b_2^*b_3) \\ Y &= (b_2 + b_1a_1^*a_2)^*(b_3 + b_1a_1^*a_3) \end{aligned}$$

Důkaz. Ponecháno na čtenáře. □

Definice 2.8 Soustava rovnic nad reg. výrazy je ve standardním tvaru vzhledem k neznámým $\Delta = \{X_1, X_2, ..., X_n\}$, má-li soustava tvar

$$\bigwedge_{i \in \{1,\dots,n\}} X_i = \alpha_{i0} + \alpha_{i1} X_1 + \alpha_{i2} X_2 + ... + \alpha_{in} X_n$$

kde α_{ij} jsou reg. výrazy nad nějakou abecedou Σ, $\Sigma \cap \Delta = \emptyset$.

Věta 2.4 Je-li soustava rovnic nad reg. výrazy ve std. tvaru, pak existuje její minimální pevný bod a algoritmus jeho nalezení.

Důkaz. Vyjadřujeme hodnotu jednotlivých proměnných pomocí řešení rovnice $X = pX + q$ jako regulární výraz s proměnnými, jejichž počet se postupně snižuje: Z rovnice pro X_n vyjádříme např. X_n jako regulární výraz nad Σ a $X_1, ..., X_{n-1}$. Dosadíme za X_n do rovnice pro X_{n-1} a postup opakujeme. Jsou přitom možné (ale ne nutné) různé optimalizace tohoto pořadí. $\square$

Příklad 2.6 Řešme soustavu rovnic nad reg. výrazy:

$$\begin{array}{llll}(1) & X_1 & = & (01^*+1)X_1+X_2 \\ (2) & X_2 & = & 11+1X_1+00X_3 \\ (3) & X_3 & = & \varepsilon+X_1+X_2\end{array}$$

- Výraz pro X_3 dosadíme z (3) do (2). Dostaneme soustavu:

$$\begin{array}{llll}(4) & X_1 & = & (01^*+1)X_1+X_2 \\ (5) & X_2 & = & 11+1X_1+00(\varepsilon+X_1+X_2)=00+11+(1+00)X_1+00X_2\end{array}$$

- Ze (4) vyjádříme X_1 s využitím řešení rovnice $X=pX+q$ (věta 2.3):

$$(6) \quad X_1 = (01^*+1)^*X_2=(0+1)^*X_2$$

- Dosazením do (5):

$$(7) \quad X_2 = 00+11+(1+00)(0+1)^*X_2+00X_2=00+11+(1+00)(0+1)^*X$$

- Vypočtením X_2 jako řešení rovnice $X=pX+q$ dostaneme:

$$(8) \quad X_2 = ((1+00)(0+1)^*)^*(00+11)$$

- Dosazením do (6) dostaneme:

$$(9) \quad X_1 = (0+1)^*((1+00)(0+1)^*)^*(00+11)=(0+1)^*(00+11)$$

- Dosazením do (3) dostaneme:

$$\begin{array}{lll}(10) \quad X_3 & = & \varepsilon+(0+1)^*(00+11)+((1+00)(0+1)^*)^*(00+11)= \\ & = & \varepsilon+((0+1)^*+((1+00)(0+1)^*)^*)(00+11)= \\ & = & \varepsilon+(0+1)^*(00+11)\end{array}$$

Regulární množiny a jazyky typu 3

Věta 2.5 Jazyk L je regulární množinou právě tehdy, je-li L jazykem typu 3. Označíme-li $\mathcal{L}_R$ třídu všech regulárních množin, pak:

$$\mathcal{L}_R = \mathcal{L}_3$$

Důkaz. I. $\mathcal{L}_R \subseteq \mathcal{L}_3$, tj. každou regulární množinu lze generovat gramatikou typu 3.

	regulární množina	*gramatika typu 3*
(1)	$\emptyset$	$G_\emptyset = (\{S\}, \Sigma, \emptyset, S)$
(2)	$\{\varepsilon\}$	$G_\varepsilon = (\{S\}, \Sigma, \{S \rightarrow \varepsilon\}, S)$
(3)	$\{a\}$ pro každé $a \in \Sigma$	$G_a = (\{S\}, \Sigma, \{S \rightarrow a\}, S)$

Nyní ukážeme, že sjednocení, konkatenaci a iteraci reg. množin lze generovat rovněž gramatikou typu 3. Nechť tedy

- $L_1 = L(G_1)$, kde $G_1 = (N_1, \Sigma_1, P_1, S_1)$,
- $L_2 = L(G_2)$, kde $G_2 = (N_2, \Sigma_2, P_2, S_2)$

a G_1, G_2 jsou gramatiky typu 3, $N_1 \cap N_2 = \emptyset$ (nonterminály je vždy možno takto odlišit).

Důkaz pokračuje dále.

Pokračování důkazu.

regulární množina		*gramatika typu 3*
		$G_4 = (N_4, \Sigma_1 \cup \Sigma_2, P_4, S_4)$, kde
(4)	$L_1 \cup L_2$	• $N_4 = N_1 \cup N_2 \cup \{S_4\}$, $S_4 \notin N_1 \cup N_2$, • $P = \{S \rightarrow S_1 \mid S_2\} \cup P_1 \cup P_2$
		$G_5 = (N_1 \cup N_2, \Sigma_1 \cup \Sigma_2, P_5, S_1)$ a P_5 je nejmenší množina takov že:
(5)	$L_1.L_2$	• je-li $(A \rightarrow xB) \in P_1$, pak $(A \rightarrow xB) \in P_5$, • je-li $(A \rightarrow x) \in P_1$, pak $(A \rightarrow xS_2) \in P_5$, • $\forall (A \rightarrow \alpha) \in P_2 : (A \rightarrow \alpha) \in P_5$.
		$G_6 = (N_1 \cup \{S_6\}, \Sigma_1, P_6, S_6)$, $S_6 \notin N_1$ a P_6 je nejmenší množ taková, že:
(6)	L_1^*	• je-li $(A \rightarrow xB) \in P_1$, pak $(A \rightarrow xB) \in P_6$, • je-li $(A \rightarrow x) \in P_1$, pak $(A \rightarrow xS_6) \in P_6$, • $(S_6 \rightarrow S_1 \mid \varepsilon) \in P_6$.

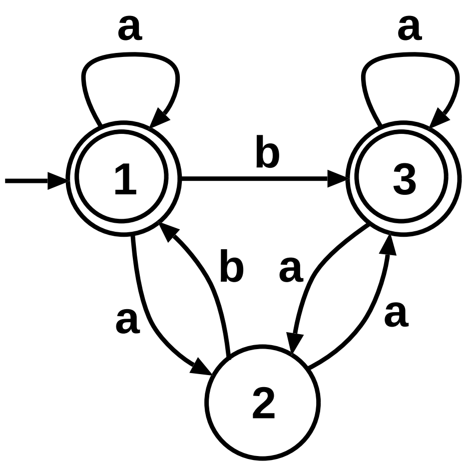
Důkaz pokračuje dále.

Pokračování důkazu. II. $\mathcal{L}_3 \subseteq \mathcal{L}_R$, tj. každý jazyk generovaný gramatikou typu 3 je regulární množinou.

- Nechť $L \in \mathcal{L}_3$ je libovolný jazyk typu 3. Již vím, že ho můžeme popsat KA $M = (Q, \Sigma, \delta, q_0, F)$. Nechť $Q = \{q_0, q_1, ..., q_n\}$.
- Vytvoříme soustavu rovnic na reg. výrazy s proměnnými $X_0, X_1, ..., X_n$ ve standardním tvaru. Rovnice pro X_i popisuje množinu řetězců přijímaných ze stavu Q_i.
- Řešením této soustavy získáme reg. výraz pro proměnnou X_0, který reprezentuje jazyk L.

$\square$

Příklad 2.7



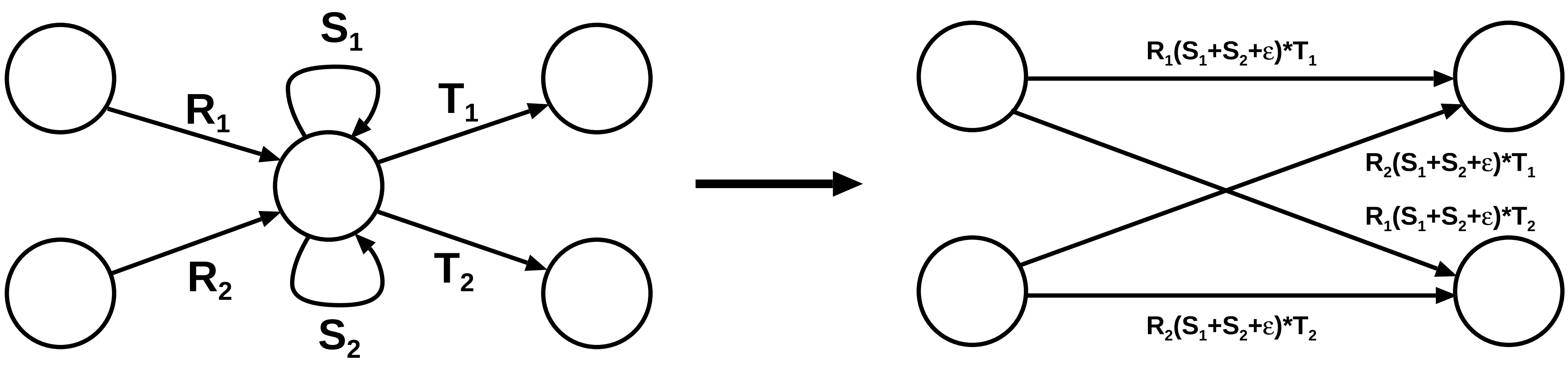
$$
\begin{array}{rcllll}
X_1 & = & \varepsilon + & aX_1 + & aX_2 + & bX_3 \\
X_2 & = & & bX_1 + & & aX_3 \\
X_3 & = & \varepsilon + & & aX_2 + & aX_3
\end{array}
$$

Jazyk L popisuje reg. výraz, který je řešením této soustavy pro proměnnou X_1.

Poznámka: jiný převod KA na RV

❖ Regulární přechodový graf je zobecnění KA, které umožňuje množinu počátečních stavů a regulární výrazy na hranách.

❖ Každý RPG je možné převést na RPG s jediným přechodem, ze kterého odečteme hledaný RV. Zavedeme nový počáteční a koncový stav, které propojíme s původními počátečními a koncovými stavy ε přechody. Pak postupně odstraňujeme všechny původní stavy následujícím způsobem:



Přímý převod RV na DKA

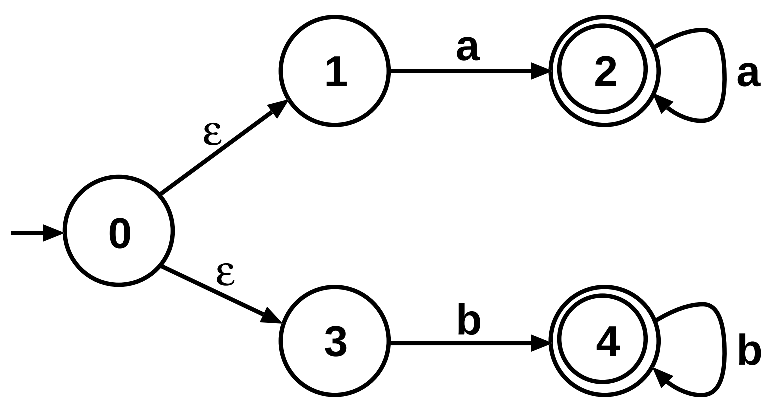
Rozšířené konečné automaty

❖ RV budeme převádět nejprve na tzv. rozšířené KA a ty pak na DKA.

Definice 2.9 Rozšířený konečný automat (RKA) je pětice $M = (Q, \Sigma, \delta, q_0, F)$, kde

- Q je konečná množina stavů,
- Σ je konečná vstupní abeceda,
- δ je zobrazení $Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$,
- $q_0 \in Q$ je počáteční stav,
- $F \subseteq Q$ je množina koncových stavů.

Příklad 2.8 $M = (\{0, 1, 2, 3, 4\}, \{a, b\}, \delta, 0, \{2, 4\})$



$$L(M) = aa^* + bb^* = a^+ + b^+$$

ε-uzávěr

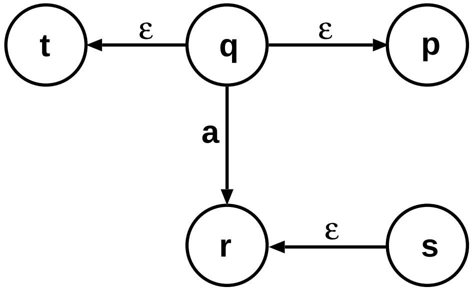
❖ Klíčovou funkci v algoritmu převodu RKA na DKA má výpočet funkce, která k danému stavu určí množinu všech stavů, jež jsou dostupné po ε hranách diagramu přechodů funkce δ. Označme tuto funkci jako ε-uzávěr:

$$\varepsilon\text{-uzávěr}(q) = \{p \mid \exists w \in \Sigma^* : (q, w) \overset{*}{\vdash} (p, w)\}$$

❖ Funkci ε-uzávěr zobecníme tak, aby argumentem mohla být množina $T \subseteq Q$:

$$\varepsilon\text{-uzávěr}(T) = \bigcup_{s \in T} \varepsilon\text{-uzávěr}(s)$$

Příklad 2.9



$$\varepsilon\text{-uzávěr}(\{q, r, s\}) = \{p, q, r, s, t\}$$

Výpočet ε-uzávěru

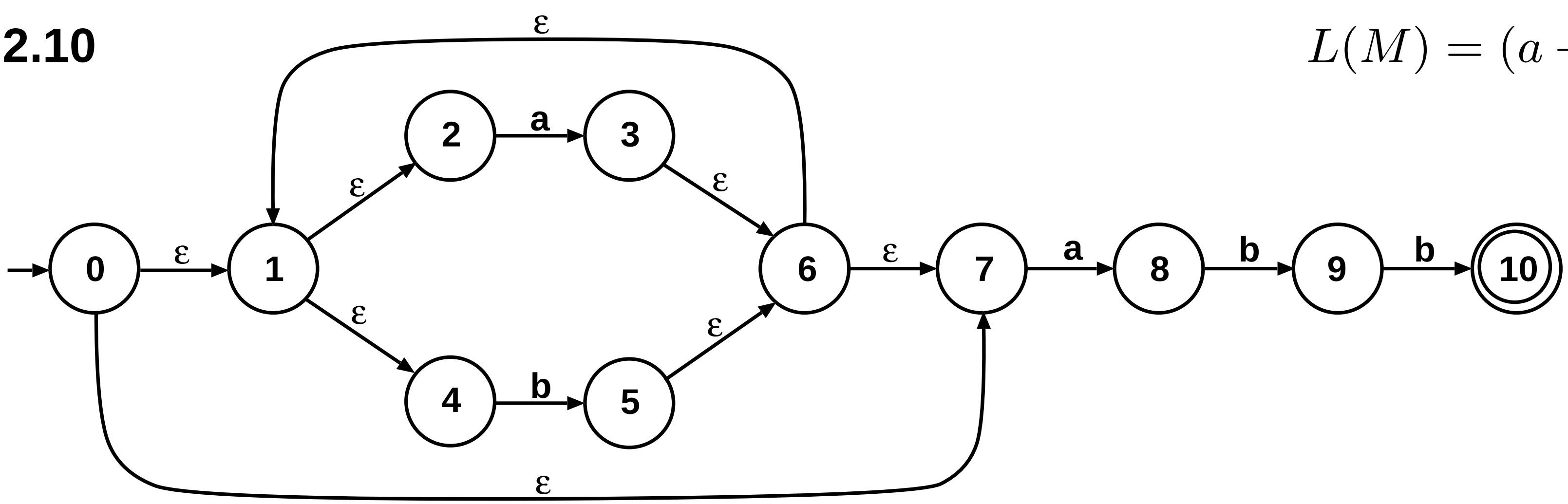
❖ Zavedeme relaci $\xrightarrow{\varepsilon}$ v množině Q takto:

$$\forall q_1, q_2 \in Q : q_1 \xrightarrow{\varepsilon} q_2 \overset{def}{\Longleftrightarrow} q_2 \in \delta(q_1, \varepsilon)$$

Pak ε-uzávěr$(p) = \{q \in Q \mid p \xrightarrow{\varepsilon}{}^{*} q\}$.

❖ K výpočtu ε-uzávěru pak použijeme Warshallův algoritmus, doplníme diagonálu jedničkami a z příslušného řádku matice výsledné relace vyčteme ε-uzávěr.

Příklad 2.10 $L(M) = (a + b)^* abb$



ε-uzávěr$(3) = \{3, 6, 7, 1, 2, 4\}$
ε-uzávěr$(\{1, 0\}) = \{0, 1, 2, 4, 7\}$

Převod RKA na ekvivalentní DKA

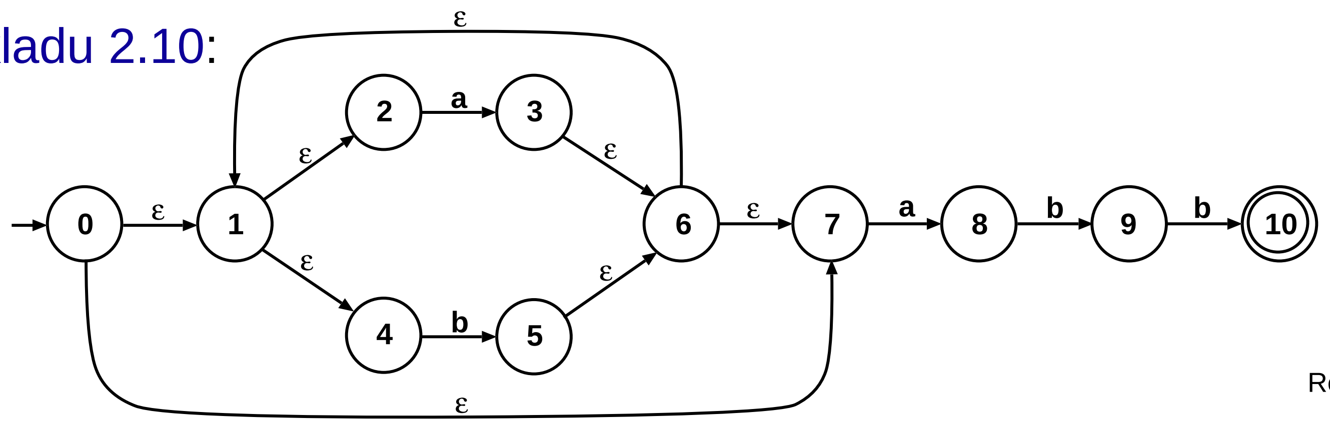
Algoritmus 2.3 Převod RKA na DKA
Vstup: RKA $M = (Q, \Sigma, \delta, q_0, F)$.
Výstup: DKA $M' = (Q', \Sigma, \delta', q_0', F')$, $L(M) = L(M')$.
Metoda:

1. $Q' := 2^Q \setminus \{\emptyset\}$.
2. $q_0' := \varepsilon$-uzávěr(q_0).
3. $\delta' : Q' \times \Sigma \rightarrow Q'$ je vypočtena takto:
   - Nechť $\forall T \in Q', a \in \Sigma : \overline{\delta}(T, a) = \bigcup_{q \in T} \delta(q, a)$.
   - Pak pro každé $T \in Q'$, $a \in \Sigma$:
     (a) pokud $\overline{\delta}(T, a) \neq \emptyset$, pak $\delta'(T, a) = \varepsilon$-uzávěr$(\overline{\delta}(T, a))$,
     (b) jinak $\delta'(T, a)$ není definováno.
4. $F' := \{S \mid S \in Q' \wedge S \cap F \neq \emptyset\}$.

Příklad 2.11 Aplikujeme algoritmus 2.3 na automat z příkladu 2.10:

1. Počáteční stav, označíme ho A, je $A = \varepsilon$-uzávěr$(0) = \{0, 1, 2, 4, 7\}$.
2. $\delta'(A, a) = \varepsilon$-uzávěr$(\{3, 8\}) = \{1, 2, 3, 4, 6, 7, 8\} = B$.
3. $\delta'(A, b) = \varepsilon$-uzávěr$(\{5\}) = \{1, 2, 4, 5, 6, 7\} = C$.
4. $\delta'(B, a) = \varepsilon$-uzávěr$(\{3, 8\}) = B$.
5. $\delta'(B, b) = \varepsilon$-uzávěr$(\{5, 9\} = \{1, 2, 4, 5, 6, 7, 9\} = D$.
6. $\delta'(C, a) = \varepsilon$-uzávěr$(\{3, 8\}) = B$.
7. $\delta'(C, b) = \varepsilon$-uzávěr$(\{5\}) = C$.
8. $\delta'(D, a) = \varepsilon$-uzávěr$(\{3, 8\}) = B$.
9. $\delta'(D, b) = \varepsilon$-uzávěr$(\{5, 10\} = \{1, 2, 4, 5, 6, 7, 10\} = E$.
10. $\delta'(E, a) = \varepsilon$-uzávěr$(\{3, 8\}) = B$.
11. $\delta'(E, b) = \varepsilon$-uzávěr$(\{5\}) = C$.
12. Množina koncových stavů $F = \{E\}$.

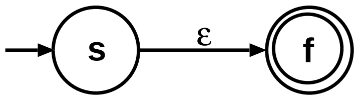
Automat z příkladu 2.10:

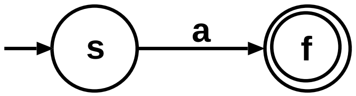


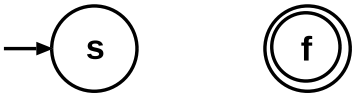
Převod RV na ekvivalentní RKA

Algoritmus 2.4 Převod RV na RKA
Vstup: RV r popisující regulární množinu R nad Σ.
Výstup: RKA M takový, že $L(M) = R$.
Metoda:

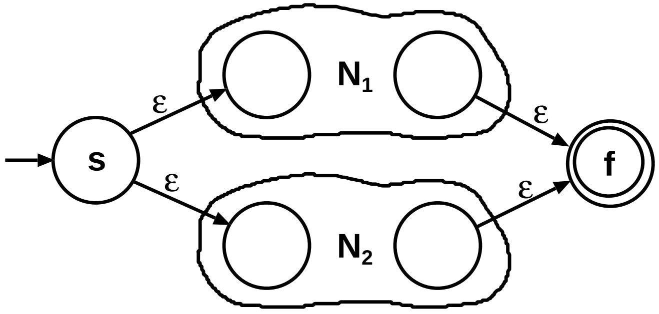
1. Rozložíme r na jeho primitivní složky podle rekurzivní definice reg. množiny/výrazu.

2. (a) Pro výraz ε zkonstruujeme automat: 

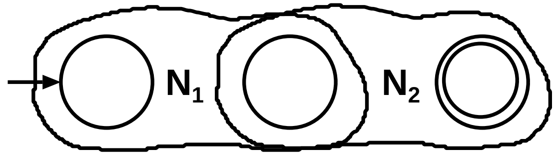
   (b) Pro výraz $a, a \in \Sigma$ zkonstruujeme automat: 

   (c) Pro výraz $\emptyset$ zkonstruujeme automat: 

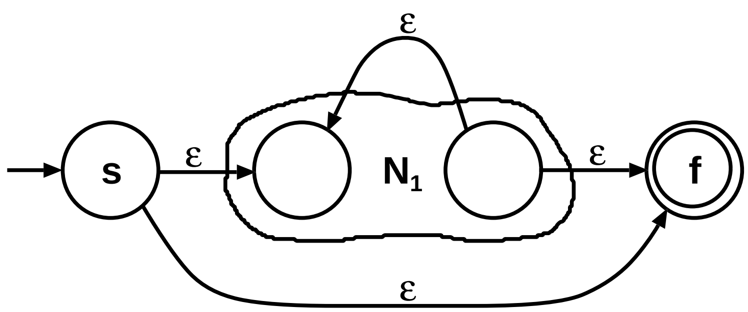
   (d) Nechť N_1 je automat přijímající jazyk specifikovaný výrazem r_1 a nechť N_2 je automat přijímající jazyk specifikovaný výrazem r_2.
      i. Pro výraz $r_1 + r_2$ zkonstruujeme automat:



2. (d) ii. Pro výraz $r_1 r_2$ zkonstruujeme automat:

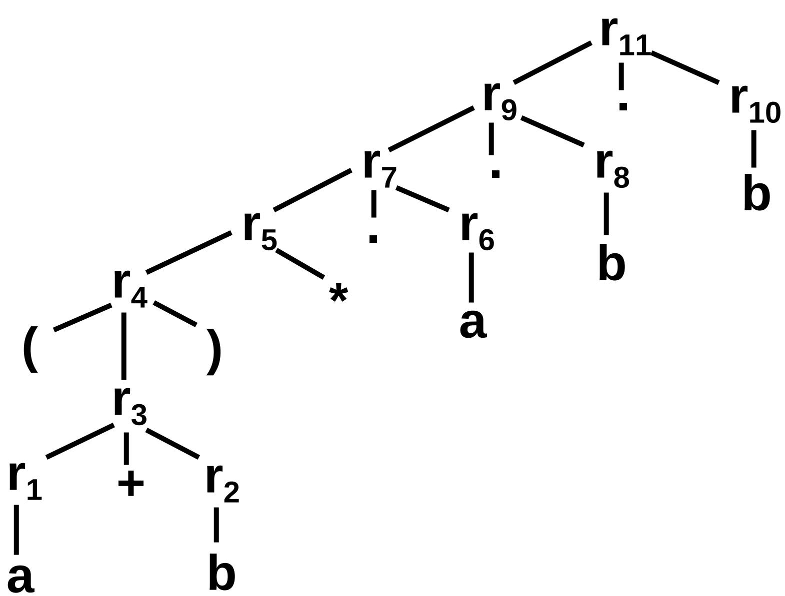


iii. Pro výraz r_1^* zkonstruujeme automat:

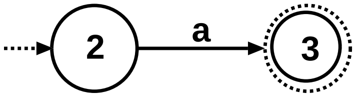


Příklad 2.12 Vytvořme RKA pro RV $(a + b)^* abb$:

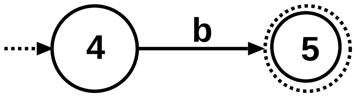
1. Rozklad RV vyjádříme stromem:



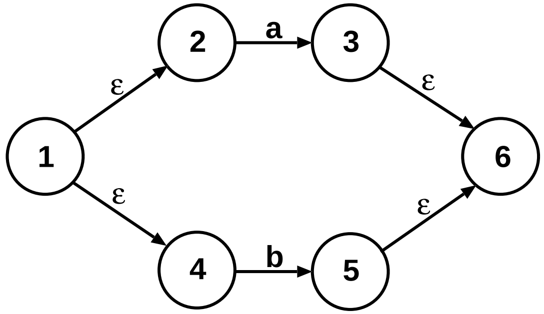
2. (a) Regulárnímu výrazu $r_1 = a$ přísluší automat N_1:



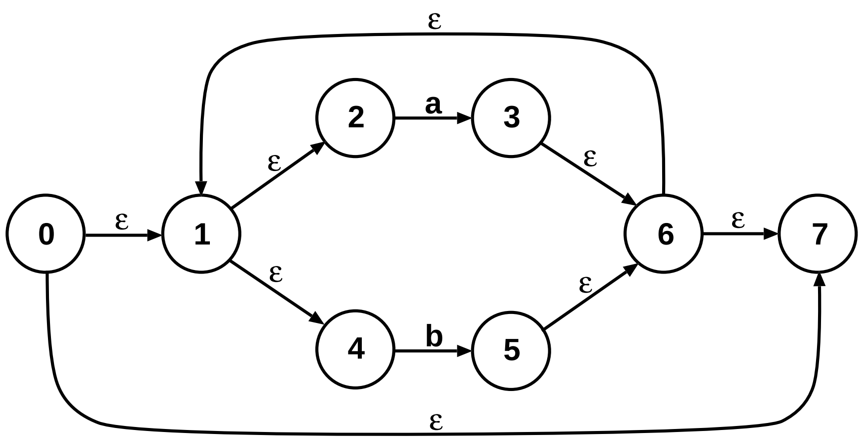
(b) Regulárnímu výrazu $r_2 = b$ přísluší automat N_2:



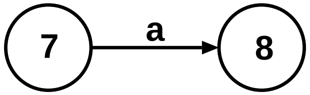
(c) Regulárnímu výrazu $r_1 + r_2$ přísluší automat N_3:



(d) Automat N_4 pro $r_4 = (r_3)$ je stejný jako N_3, zkonstruujeme tedy rovnou N_5 pro výraz $r_5 = r_4^* = (a + b)^*$:

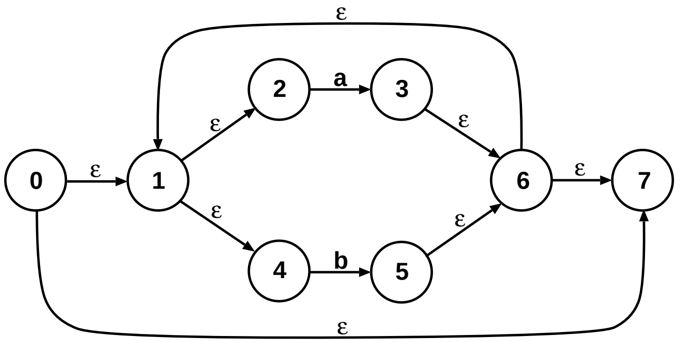


2. (e) Regulárnímu výrazu $r_6 = a$ přísluší automat N_6:

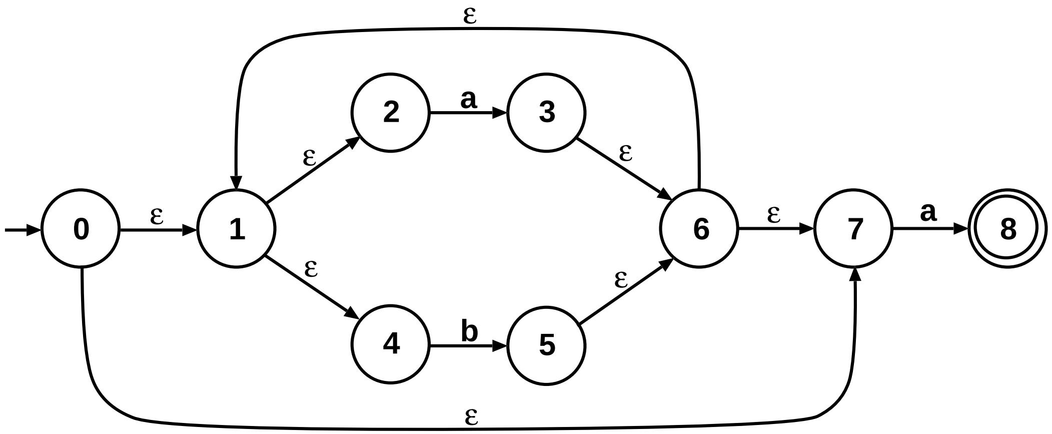


(f) Regulárnímu výrazu $r_7 = r_5 r_6$ přísluší automat N_7:

pro zopakování N_5:



N_7:



(...) Pokračujeme až do získání automatu z příkladu 2.10.

❖ Převod RV na RKA zavádí mnoho vnitřních stavů a je proto obvykle následován použitím algoritmu minimalizace DKA (algoritmus 2.2).

Vztahy regulárních gramatik, KA a RV

❖ Můžeme tedy shrnout, že

- gramatiky typu 3 (pravé/levé regulární gramatiky, pravé/levé lineární gramatiky),
- (rozšířené/nedeterministické/deterministické) konečné automaty a
- regulární výrazy

mají ekvivalentní vyjadřovací sílu.