3. Determinizace a regulární výrazy

Formální jazyky a automaty

Jiří Balun

Obsah

1 Determinizace
- Podmnožinová konstrukce
- Ekvivalence NFA a DFA

2 NFA s ε-přechody (ε-NFA)
- Popis ε-NFA
- Odstranění ε-přechodů

3 Regulární výrazy (RE)
- Popis RE
- Převod RE na ε-NFA
- Převod DFA na RE

Motivace

Na minulé přednášce

- ukázali jsme, že regulární jazyky:
  - jsou uzavřené na operace jako doplněk, průnik a sjednocení
  - lze porovnávat na rovnost a inkluzi
- rozšířili jsme konečné automaty o nedeterminismus

Na této přednášce

- ukážeme, že DFA a NFA jsou výpočetně ekvivalentní
- dále rozšíříme model NFA o ε-přechody (usnadní nám další konstrukce a důkazy)
- zavedeme nový formalismus, takzvané **regulární výrazy**
  - ukážeme jeho ekvivalenci s konečnými automaty
  - tím také dokážeme uzavřenost regulárních jazyků na zřetězení a Kleeneho uzávěr

Podmnožinová konstrukce

Intuice: vytvoříme DFA, jehož stavy odpovídají podmnožinám stavů původního NFA a přechodovou funkcí simulujeme výpočet rozšířené přechodové funkce.

Definice

K libovolnému NFA $A = (Q, \Sigma, \delta_N, q_0, F)$ sestavíme pomocí **podmnožinové konstrukce** DFA $A^{det} = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$:

1. $Q_D = 2^Q$, tj. stavy A^{det} jsou podmnožiny Q,
2. počáteční stav je $\{q_0\}$, množina obsahující pouze původní počáteční stav q_0,
3. koncové stavy jsou podmnožiny Q obsahující aspoň jeden původní koncový stav, tedy $F_D = \{S \subseteq Q \mid S \cap F \neq \emptyset\}$,
4. $\delta_D \colon 2^Q \times \Sigma \to 2^Q$ je determinisitcká přechodová funkce definovaná jako:

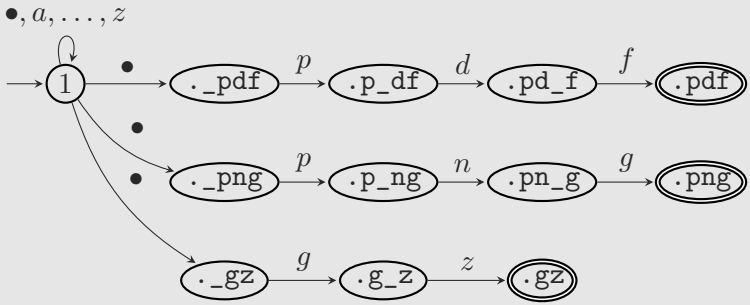
$$\delta_D(\{q_1, \ldots, q_k\}, a) = \bigcup_{i=1}^{k} \delta_N(q_i, a)$$

- při konstrukci A^{det} stačí vypočítat jen podmnožinu jeho dosažitelných stavů

Příklad: NFA rozpoznávající přípony souborů

Příklad

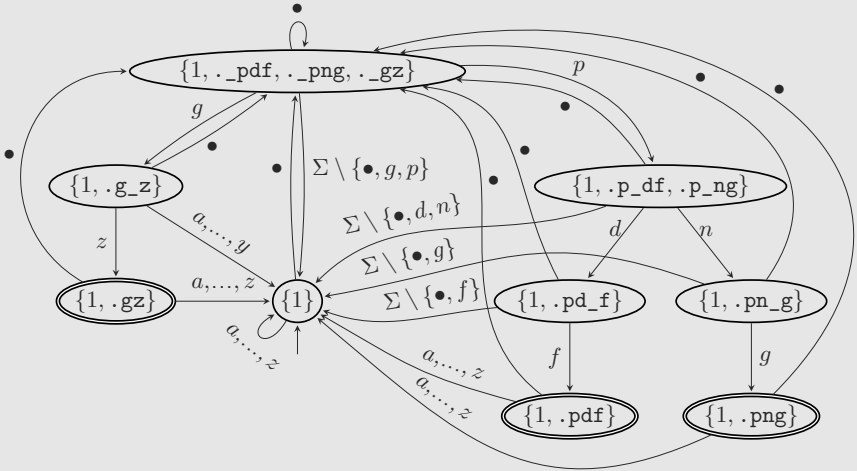
NFA $A_{sfx} = (Q, \{\bullet, \mathtt{a}, \ldots, \mathtt{z}\}, \delta, 1, \{\mathtt{.pdf}, \mathtt{.png}, \mathtt{.gz}\})$ z minulé přednášky, který rozpoznává jazyk přípon souborů $L_{sfx} = \{w \mid w$ má příponu `.pdf`, `.png`, nebo `.gz`$\}$.



Příklad: podmnožinová konstrukce

Příklad

DFA $A^{det}_{sfx} = (Q_D, \{\bullet, \mathtt{a}, \ldots, \mathtt{z}\}, \delta_D, \{1\}, F_D\})$ s jazykem $L(A_{sfx}) = L(A^{det}_{sfx})$.



Ekvivalence NFA a DFA ($\Rightarrow$)

Věta

Ke každému NFA A existuje DFA A^{det} takový, že $L(A) = L(A^{det})$.

Důkaz

Nechť $A^{det} = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ je DFA získaný pomocí podmnožinové konstrukce z $A = (Q, \Sigma, \delta_N, q_0, F)$.

Indukcí přes délku slova $w \in \Sigma^*$ ukážeme, že:

$$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$$

δ_D interpretuje podmnožiny Q jako stavy A^{det}, proto je tato rovnost korektně definovaná.

Indukční předpoklad: nechť $|w| = 0$, a tedy musí platit $w = \varepsilon$.

- Dle rozšířené přechodové funkce pro DFA $\hat{\delta}_D(\{q_0\}, \varepsilon) = \{q_0\}$,
- dle rozšířené přechodové funkce pro NFA $\hat{\delta}_N(q_0, \varepsilon) = \{q_0\}$,
- dohromady dostáváme $\hat{\delta}_D(\{q_0\}, \varepsilon) = \hat{\delta}_N(q_0, \varepsilon)$.

Ekvivalence NFA a DFA ($\Rightarrow$)

Důkaz – pokračování

Indukční krok: předpokládejme, že rovnost platí pro slova délky n, a nechť $|w| = n + 1$.

- Řetězec w si rozdělíme na $w = xa$, kde a je poslední znak w, tj. $|x| = n$,
- z předpokladu plyne $\hat{\delta}_D(\{q_0\}, x) = \hat{\delta}_N(q_0, x) = R$, kde $R \subseteq Q$,
- z definice rozšířené přechodové funkce pro NFA platí:

$$\hat{\delta}_N(q_0, w) = \hat{\delta}_N(\hat{\delta}_N(q_0, x), a) = \hat{\delta}_N(R, a) = \bigcup_{r \in R} \delta_N(r, a)$$

- z definice podmnožinové konstrukce platí:

$$\hat{\delta}_D(\{q_0\}, w) = \delta_D(\hat{\delta}_D(q_0, x), a) = \delta_D(R, a) = \bigcup_{r \in R} \delta_N(r, a)$$

- dohromady tedy dostáváme $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$.

Vzhledem k tomu, že A i A^{det} přijmou w právě když $\hat{\delta}_D(\{q_0\}, w)$ a $\hat{\delta}_N(q_0, w)$ obsahují alespoň jeden koncový stav, pak platí $L(A) = L(A^{det})$. $\square$

Ekvivalence NFA a DFA ($\Leftarrow$)

Intuice: V DFA stačí "přetypovat" výstup přechodové funkce, abychom získali NFA.

Věta

Ke každému DFA A existuje NFA A' takový, že $L(A) = L(A')$.

Důkaz

K DFA $A = (Q, \Sigma, \delta, q_0, F)$ sestrojíme NFA $A' = (Q, \Sigma, \delta', q_0, F)$ tak, že:

- v δ' definujeme $\delta'(q, a) = \{p\}$ pro každý přechod $\delta(q, a) = p$, který je v δ.

Evidentně platí $L(A) = L(A')$. $\square$

- na DFA lze pohlížet jako na speciální případ NFA, kde pro každou dvojici stavu $q \in Q$ a symbolu $a \in \Sigma$ platí $|\delta(q, a)| = 1$
- **Důsledek:** DFA a NFA rozpoznávají stejné (regulární) jazyky

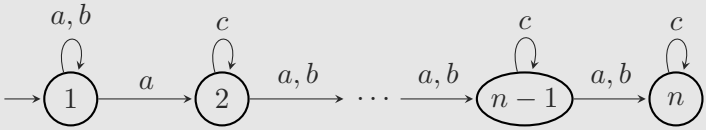
Mezní případ

Věta

Pro každé $n \in \mathbb{N}$ existuje NFA s n stavy takový, že ekvivalentní DFA sestrojený pomocí podmnožinové konstrukce má 2^n dosažitelných stavů.

Idea důkazu

Pro zadané n sestrojíme NFA $A_n = (\{1, \ldots, n\}, \{a, b, c\}, \delta, q_0, \emptyset)$, který má po determinizaci 2^n dosažitelných stavů:



- A_n nemá koncové stavy, proto lze lehce najít menší automat se stejným jazykem
- **přirozená otázka:** existuje způsob jak vypočítat nejmenší deterministický automat?
  - tento problém vyřešíme později pomocí minimalizace automatu

NFA s ε-přechody

Intuice: rozšíříme model NFA o možnost provádět přechody i bez čtení znaku ze vstupu.

Definice

NFA s ε-přechody (ε-NFA) je NFA $A = (Q, \Sigma, \delta_E, q_0, F)$ s přechodovou funkcí ve tvaru $\delta_E \colon Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$.

- jedná se o další konzervativní rozšíření modelu konečných automatů:
  - každý NFA je speciální případ ε-NFA (jen nemá žádný ε-přechod)
  - každý ε-NFA lze převést na NFA bez ε-přechodů
  - proto ε-NFA stejně jako DFA a NFA rozpoznávají **regulární jazyky**
- definujeme **ε-uzávěr** $E(S)$ jako množinu dosažitelných stavů pomocí ε-přechodů ze stavů v $S \subseteq Q$ (případně i pro jeden stav $E(q) = E(\{q\})$, kde $q \in Q$)
- **Rozšířenou přechodovou funkci** $\hat{\delta}_E \colon 2^Q \times \Sigma^* \to 2^Q$ definujeme pro dvojici množiny stavů $R \subseteq Q$ a řetězce $s = s_1 \dots s_n \in \Sigma^*$ následovně:

$$\hat{\delta}_E(R, s) = \begin{cases} E(R) & \text{pro } s = \varepsilon \\ \hat{\delta}_E\big(E(\bigcup_{r \in E(R)} \delta_E(r, s_1)),\ s_2 \dots s_n\big) & \text{pro } s \neq \varepsilon \end{cases}$$

Odstranění ε-přechodů

Intuice: přepočítáme přechodovou funkci δ_E aby vracela stejné stavy i bez ε-přechodů.

Věta

Ke každému ε-NFA A existuje NFA A' takový, že $L(A) = L(A')$.

Idea důkazu

Nechť $A = (Q, \Sigma, \delta_E, q_0, F)$ je ε-NFA, pak k němu vytvoříme NFA $A' = (Q, \Sigma, \delta_N, q_0, F')$:

1. Pro každý stav $q \in Q$ vypočítáme ε-uzávěr $E_q = E(\{q\})$,
2. výpočítáme δ_N z δ_E pro každý stav $q \in Q$ a symbol $a \in \Sigma$:

$$\delta_N(q, a) = E\left(\bigcup_{p \in E_q} \delta_E(p, a)\right)$$

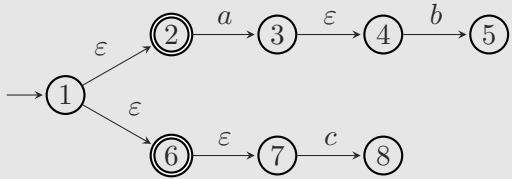
3. označ jako koncový stav q_0 pokud platí $E(\{q_0\}) \cap F \neq \emptyset$.

Indukcí přes délku slova w lze ukázat, že $\hat{\delta}_E(q_0, w) = \hat{\delta}_N(q_0, w)$, a tedy $L(A) = L(A')$. $\square$

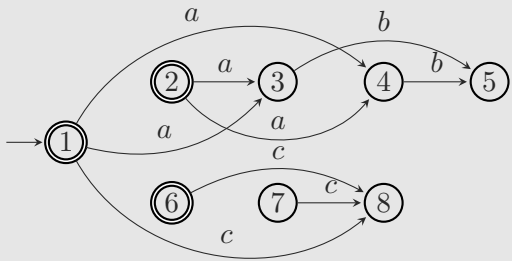
Příklad: odstranění ε-přechodů

Příklad

Pro ε-NFA $A = (Q, \Sigma, \delta_E, q_0, \{5, 6\})$ sestrojíme NFA bez ε-přechodů. Vypočítáme $E(1) = \{1, 2, 6, 7\}$, $E(3) = \{3, 4\}$, $E(6) = \{6, 7\}$ (zbylé uzávěry jsou jednoprvkové).



Výsledný NFA A':



Motivační příklad pro regulární výrazy

- definovat automat (jeho jednotlivé stavy a přechody) v kódu by bylo nepraktické
- proto zavedeme regulární výrazy, které lze jednoduše zapisovat
- mnoho aplikací: `grep`, `sed`, ověřování formulářů, nahrazování řetězce v kódu atd.
- regulární výrazy nedávají návod jak řetězce verifikovat, musíme je převést na automat

Příklad

Chceme úsporně zapsat jazyk $L_{sfx} = \{w \mid w$ má příponu `.pdf`, `.png`, nebo `.gz`$\}$ nad abecedou $\Sigma = \{., a, \ldots, z\}$. Nejprve jej vyjádříme pomocí zřetězení a Kleeneho uzávěru:

$$\{., a, \ldots, z\}^* \cdot \{.pdf, .png, .gz\}$$

což je ekvivalentní s výrazem:

$$(\{.\} \cup \{a\} \cup \ldots \cup \{z\})^* \cdot (\{.pdf\} \cup \{.png\} \cup \{.gz\})$$

na kterém lze vidět podobnost s ekvivalentním regulárním výrazem:

$$(\boldsymbol{.} + \boldsymbol{a} + \ldots + \boldsymbol{z})^*(\boldsymbol{.pdf} + \boldsymbol{.png} + \boldsymbol{.gz})$$

Regulární výrazy (RE)

Intuice: regulární výraz je formalismus pro čitelný a úsporný zápis regulárních jazyků.

Definice

Regulární výraz (RE z *regular expression*) nad abecedou Σ definujeme jako:

1. každý symbol $a \in \Sigma$ je $\boldsymbol{a}$ RE s jazykem $L(\boldsymbol{a}) = \{a\}$,
2. $\boldsymbol{\varepsilon}$ je RE s jazykem $L(\boldsymbol{\varepsilon}) = \{\varepsilon\}$,
3. $\boldsymbol{\emptyset}$ je RE s jazykem $L(\boldsymbol{\emptyset}) = \emptyset$,
4. nechť $\boldsymbol{E}_1$ a $\boldsymbol{E}_2$ jsou RE, pak $\boldsymbol{E}_1 + \boldsymbol{E}_2$ je RE s jazykem $L(\boldsymbol{E}_1 + \boldsymbol{E}_2) = L(\boldsymbol{E}_1) \cup L(\boldsymbol{E}_2)$,
5. nechť $\boldsymbol{E}_1$ a $\boldsymbol{E}_2$ jsou RE, pak $\boldsymbol{E}_1\boldsymbol{E}_2$ je RE s jazykem $L(\boldsymbol{E}_1\boldsymbol{E}_2) = L(\boldsymbol{E}_1) \cdot L(\boldsymbol{E}_2)$,
6. nechť $\boldsymbol{E}$ je RE, pak $\boldsymbol{E}^*$ je RE s jazykem $L(\boldsymbol{E}^*) = (L(\boldsymbol{E}))^*$.

- priorita operátorů: nejvyšší má operace *, pak zřetězení $\cdot$, a nejnižší má operace $+$
- syntaxe RE v programovacích jazycích se většinou liší (více možností, různé standardy)
- kvůli přehlednost si zavedeme ještě regulární výrazy pro:
  - $\boldsymbol{\Sigma}$ s jazykem $L(\boldsymbol{\Sigma}) = \cup_{a \in \Sigma} \{a\}$
  - $\boldsymbol{E}^+ = \boldsymbol{E}\boldsymbol{E}^*$ s jazykem $L(\boldsymbol{E}^+) = L(\boldsymbol{E}) \cdot (L(\boldsymbol{E}))^*$

Příklady RE

Příklad

RE pro jazyk $L_{mod} = \{w \in \{a,b\}^* \mid$ platí, že $|w| \equiv 0 \pmod 4\}$:

$$((\boldsymbol{a}+\boldsymbol{b})(\boldsymbol{a}+\boldsymbol{b})(\boldsymbol{a}+\boldsymbol{b})(\boldsymbol{a}+\boldsymbol{b}))^*$$

nebo také pomocí zjednodušeného zápisu $(\boldsymbol{\Sigma\Sigma\Sigma\Sigma})^*$.

Příklad

RE pro jazyk $L = \{w \in \{a,b\} \mid w \neq bbb\}$:

$$\boldsymbol{\Sigma}^*\boldsymbol{a}\boldsymbol{\Sigma}^* + \boldsymbol{b} + \boldsymbol{bb} + \boldsymbol{bbbb}^+$$

Příklad

RE pro jazyk emailových adres nad abecedou $\Sigma = \{., @, a, \ldots, z\}$:

$$(\boldsymbol{a}+\ldots+\boldsymbol{z})(\boldsymbol{.}+\boldsymbol{a}+\ldots+\boldsymbol{z})^*\boldsymbol{@}(\boldsymbol{a}+\ldots+\boldsymbol{z})^+\boldsymbol{.}(\boldsymbol{a}+\ldots+\boldsymbol{z})^+$$

Převod RE na ε-NFA

Věta

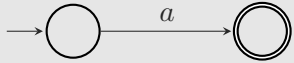
Ke každému RE E existuje ε-NFA A takový, že $L(E) = L(A)$.

Důkaz

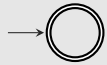
Strukturální indukcí přes složitost regulárního výrazu E popíšeme jak sestrojit ε-NFA A.

Indukční předpoklad: začneme u atomických výrazů.

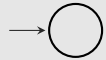
1. Pro RE E ve tvaru $\boldsymbol{a}$:



2. Pro RE E ve tvaru $\boldsymbol{\varepsilon}$:



3. Pro RE E ve tvaru $\boldsymbol{\emptyset}$:

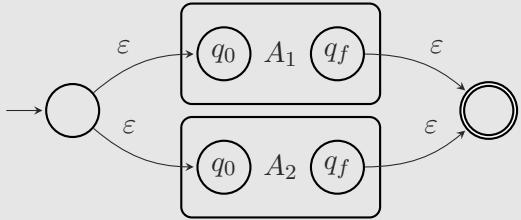


Převod RE na ε-NFA

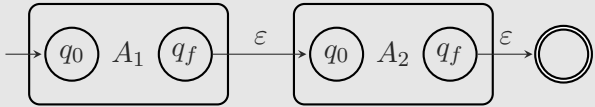
Důkaz – pokračování

Induktivní krok: předpokládáme, že platí 1, 2 a 3 pro jednoduché výrazy. Nyní ukážeme jak převést výrazy složené pomocí operací $+$,$\cdot$ a *.

4 Nechť existují ε-NFA A_1 a A_2 (každý se svým počátečním stavem q_0 a koncovými stavy $q_f \in F$) pro výrazy E_1 a E_2, pak pro výraz E ve tvaru $E_1 + E_2$:



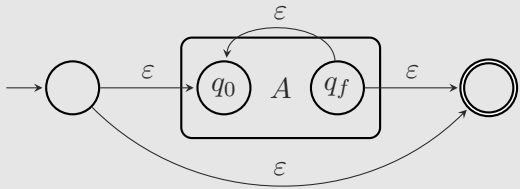
5 Nechť existují ε-NFA A_1 a A_2 pro výrazy E_1 a E_2, pak pro výraz E ve tvaru E_1E_2:



Převod RE na ε-NFA

Důkaz – pokračování

6 Nechť existuje ε-NFA A pro výraz E, pak pro výraz ve tvaru E^*:



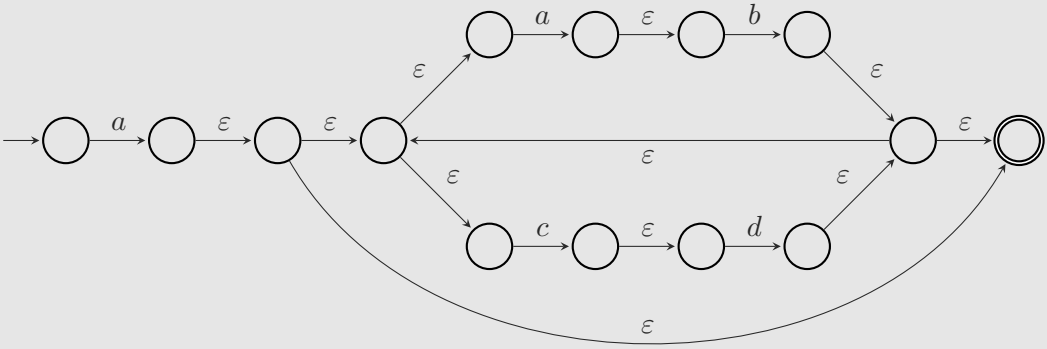
Ke každé části definice RE jsme sestrojili ε-NFA, který rozpoznává stejný jazyk, proto tímto postupem můžeme převést libovolně složitý výraz. □

- **Důsledek:** jazyky, které popisují RE, jsou regulární
- tato konstrukce vytváří velké množství zbytečných stavů a přechodů
  - při převodu lze výsledný automat konstruovat rovnou zjednodušený

Příklad – převod RE na ε-NFA

Příklad

Pro RE $\boldsymbol{a}(\boldsymbol{ab}+\boldsymbol{cd})^*$ sestrojíme ε-NFA:



Konstrukce k-cest

Intuice: ekvivalentní RE sestrojíme pomocí speciálních cest v grafu automatu, které začínají v počátečním stavu a končí v koncovém stavu.

Definice

Nechť $A = (\{1, \ldots, n\}, \Sigma, \delta, 1, F)$ je DFA s n stavy (jsou označené čísly od 1 do n), pak RE R_A, pro který platí $L(R_A) = L(A)$, vypočítáme předpisem:

$$R_A = \sum_{f \in F} R^n_{1,f}$$

kde jednotlivé RE $R^n_{1,f}$ získáme jako:

$$\begin{aligned}
R^k_{ij} &= R^{k-1}_{ij} + R^{k-1}_{ik}(R^{k-1}_{kk})^* R^{k-1}_{kj} \\
R^0_{ij} &= \sum_{a \in \Sigma:\ \delta(i,a)=j} \boldsymbol{a} \qquad (\text{pro } i \neq j) \\
R^0_{ii} &= \boldsymbol{\varepsilon} + \sum_{a \in \Sigma:\ \delta(i,a)=i} \boldsymbol{a}
\end{aligned}$$

Konstrukce k-cest – vysvětlení

- výraz $R^k_{i,j}$ je RE, který popisuje množinu všech cest (grafem automatu) ze stavu i do stavu j přes stavy, jež jsou označené maximálně číslem k:

$$R^k_{ij} = R^{k-1}_{ij} + R^{k-1}_{ik}(R^{k-1}_{kk})^* R^{k-1}_{kj}$$

  výraz výpočítáme z jednodušších podvýrazů:
  1. R^{k-1}_{ij} jsou cesty z i do j, které nevedou přes k
  2. $R^{k-1}_{ik}(R^{k-1}_{kk})^* R^{k-1}_{kj}$ jsou cesty z i do j, které vedou přes k

- výrazy, kde $k = 0$, slouží jako podmínka ukončující rekurzi (už jsme na úrovni jednotlivých přechodů mezi stavy i a j):

$$R^0_{ij} = \sum_{a \in \Sigma:\ \delta(i,a)=j} \boldsymbol{a} \qquad (\text{pro } i \neq j)$$

$$R^0_{ii} = \boldsymbol{\varepsilon} + \sum_{a \in \Sigma:\ \delta(i,a)=i} \boldsymbol{a}$$

- výsledný výraz je součet všech cest z počátečního stavu 1 do všech koncových stavů:

$$R_A = \sum_{f \in F} R^n_{1,f}$$

Převod DFA na RE

Věta

Ke každému DFA A existuje RE E takový, že $L(A) = L(E)$. □

- tvrzení dokazovat nebudeme (důkaz je veden indukcí přes složitost výrazů R_A a R_{ij}^k)

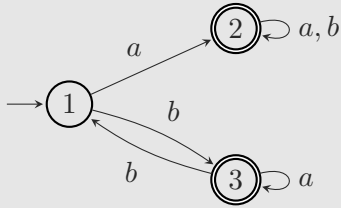
Důsledky

- RE je ekvivalentní s DFA, NFA a ε-NFA
  - všechny tyto modely lze mezi sebou libovolně převádět
- regulární jazyky jsou uzavřené na zřetězení a Kleeneho uzávěr
  - RE jsou uzavřené na zřetězení a Kleeneko uzávěr dle definice
  - proto je na tyto operace uzavřený i libovolný regulární jazyk

Příklad – převod DFA na RE

Příklad

Mějme DFA $A = (\{1,2,3\}, \{a,b\}, \delta, 1, \{2,3\})$ a chceme najít ekvivalentní RE:



$$R_A = R^3_{12} + R^3_{13}$$

$$R^3_{12} = R^2_{12} + R^2_{13}(R^2_{33})^* R^2_{32} = \dots$$

$$R^2_{12} = R^1_{12} + R^1_{12}(R^1_{22})^* R^1_{22} = \boldsymbol{a} + \boldsymbol{a}(\boldsymbol{\varepsilon} + \boldsymbol{a} + \boldsymbol{b})^*(\boldsymbol{\varepsilon} + \boldsymbol{a} + \boldsymbol{b}) = \boldsymbol{a}(\boldsymbol{a} + \boldsymbol{b})^*$$

$$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = \boldsymbol{a} + \boldsymbol{\varepsilon}(\boldsymbol{\varepsilon})^* \boldsymbol{a} = \boldsymbol{a}$$

$$R^1_{22} = R^0_{22} + R^0_{21}(R^0_{11})^* R^0_{12} = \boldsymbol{\varepsilon} + \boldsymbol{a} + \boldsymbol{b} + \boldsymbol{\emptyset}(\boldsymbol{\varepsilon})^* \boldsymbol{a} = \boldsymbol{\varepsilon} + \boldsymbol{a} + \boldsymbol{b}$$

$$R^2_{13} = R^1_{13} + R^1_{12}(R^1_{22})^* R^1_{23} = \boldsymbol{b} + \boldsymbol{a}(\boldsymbol{\varepsilon} + \boldsymbol{a} + \boldsymbol{b})^* \boldsymbol{\emptyset} = \boldsymbol{b}$$

... zbytek na cvičení.