3. Nedeterminismus

Formální jazyky a automaty

Jiří Balun

Obsah

Simulace běhu více DFA

Součinový DFA

Intuice: součinový automat simuluje běh více automatů najednou, což nám umožní porovnávat jazyky, které rozpoznávají.

Definice

Pro libovolné DFA $A_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1)$ a $A_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$ nad abecedou Σ definujeme **součinový DFA** jako $A_1 \times A_2 = (Q_1 \times Q_2, \Sigma, \delta, \langle q_{0,1}, q_{0,2} \rangle, F)$, kde:

- množina stavů obsahuje všechny **dvojice stavů** $\langle q_1, q_2 \rangle$, kde $q_1 \in Q_1$ a $q_2 \in Q_2$,
- abeceda Σ zůstává stejná (A_1 a A_2 musí být nad stejnou abecedou),
- $\delta(\langle q_1, q_2 \rangle, a) = \langle \delta_1(q_1, a), \delta_2(q_2, a) \rangle$ pro všechna $a \in \Sigma$,
- počáteční stav $\langle q_{0,1}, q_{0,2} \rangle$ je dvojice počátečních stavů z A_1 a A_2,
- množinu koncových stavů $F \subseteq Q_1 \times Q_2$ zvolíme podle účelu $A_1 \times A_2$ (viz. dále).

- konstrukci lze zobecnit pro libovolný konečný součin automatů $A_1 \times \ldots \times A_n$
- pokud A_1 a A_2 jsou nad různými abecedami $\Sigma_1 \neq \Sigma_2$, lze je "rozšířit":
  - oběma nastavíme stejnou abecedu $\Sigma_1 \cup \Sigma_2$, a poté zúplníme jejich přechodové funkce

Průnik regulárních jazyků

Intuice: použijeme součinový automat, který přijme řetězec, pokud jej přijmou oba simulované automaty, tj. zvolíme koncové stavy $F = \{\langle q_1, q_2 \rangle \mid q_1 \in F_1 \wedge q_2 \in F_2\}$.

Věta

Pokud L_1 a L_2 jsou regulární jazyky, pak $L_1 \cap L_2$ je také regulární jazyk.

Důkaz

K jazykům L_1 a L_2 existují automaty A_1 a A_2, takové že:

1. $A_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1)$ a $L(A_1) = L_1$,
2. $A_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$ a $L(A_2) = L_2$.

Zkonstruujeme $A_1 \times A_2$ s množinou koncových stavů $F = \{\langle q_1, q_2 \rangle \mid q_1 \in F_1 \wedge q_2 \in F_2\}$.

Potom platí: $w \in L(A_1 \times A_2)$ právě tehdy, když $\delta(\langle q_{0,1}, q_{0,2} \rangle, w) \in F$, což je ekvivalentní s tím, že $\delta_1(q_{0,1}, w) \in F_1$ a zároveň $\delta_2(q_{0,2}, w) \in F_2$, což je ekvivalentní s $w \in L_1 \cap L_2$. □

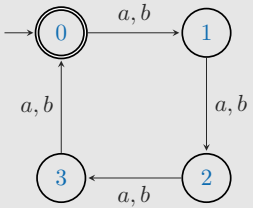
- **Důsledek:** třída regulárních jazyků je uzavřená na průnik

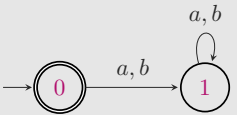
Připomenutí: DFA A_{mod} a A_{ε}

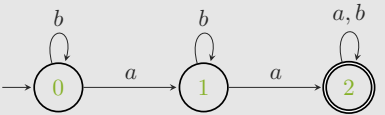
Příklad

Na minulé přednášce jsme měli tyto DFA nad $\Sigma = \{a, b\}$:

1. A_{mod} s jazykem $L_{mod} = \{w \in \Sigma^* \mid \text{pro } w \text{ platí, že } |w| \equiv 0 \,(\text{mod } 4)\}$,
2. A_{ε} s jazykem $L_{\varepsilon} = \{\varepsilon\}$, který obsahuje pouze prázdný řetězec,
3. $A_{\#}$ s jazykem $L_{\#} = \{w \in \{a, b\}^* \mid w \text{ obsahuje alespoň 2 znaky } a, \text{ tj. } \#_a(w) \geq 2\}$.



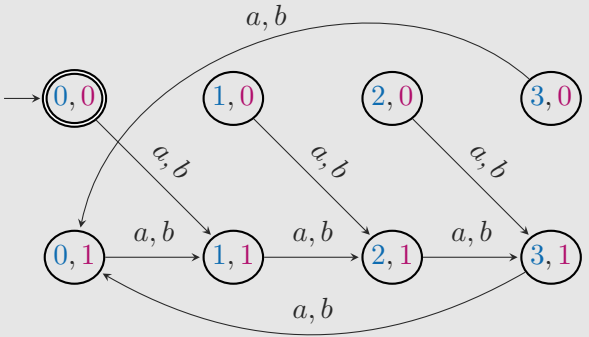




Příklad: průnik regulárních jazyků

Příklad

Součinový DFA $A_{mod} \times A_{\varepsilon}$ (včetně nedosažitelných stavů) s množinou koncových stavů $F = \{\langle 0, 0\rangle\}$ rozpoznává jazyk $L(A_{mod}) \cap L(A_{\varepsilon}) = L_{\varepsilon}$:



Sjednocení regulárních jazyků

Intuice: použijeme součinový automat, který přijme řetězec, pokud jej přijme alespoň jeden simulovaný automat, tj. zvolíme koncové stavy $F = \{\langle q_1, q_2 \rangle \mid q_1 \in F_1 \vee q_2 \in F_2\}$.

Věta

Pokud L_1 a L_2 jsou regulární jazyky, pak $L_1 \cup L_2$ je také regulární jazyk.

Důkaz

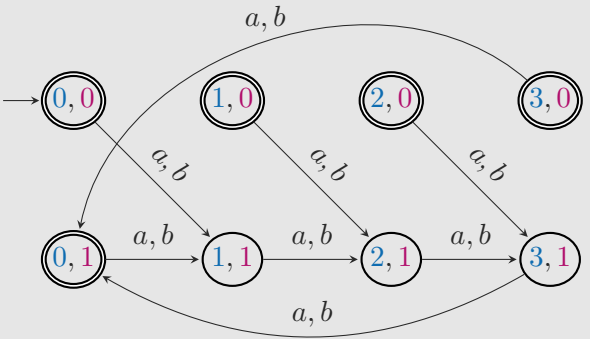
Analogicky jako u průniku. □

- **Důsledek:** třída regulárních jazyků je uzavřená na sjednocení

Příklad: sjednocení regulárních jazyků

Příklad

Součinový DFA $A_{mod} \times A_{\varepsilon}$ (včetně nedosažitelných stavů) s množinou koncových stavů $F = \{\langle 0,0\rangle, \langle 1,0\rangle, \langle 2,0\rangle, \langle 3,0\rangle, \langle 0,1\rangle\}$ rozpoznává jazyk $L(A_{mod}) \cup L(A_{\varepsilon}) = L_{mod}$:



Rozhodování rovnosti regulárních jazyků

Intuice: rozhodneme neprázdnost jazyka součinového automatu, který přijímá řetězec v případě, že jej přijme pouze jeden simulovaný automat (nalezne protipříklad), tj. zvolíme:

$$F = \{\langle q_1, q_2 \rangle \mid (q_1 \in F_1 \wedge q_2 \notin F_2) \vee (q_1 \notin F_1 \wedge q_2 \in F_2)\}$$

Věta

Problém rovnosti $L_1 = L_2$ dvou regulárních jazyků L_1 a L_2 je rozhodnutelný.

Důkaz

K jazykům L_1 a L_2 existují automaty A_1 a A_2, takové že:

- $A_i = (Q_i, \Sigma, \delta_i, q_{0,i}, F_i)$ a $L(A_i) = L_i$ pro $i \in \{1, 2\}$.

Zkonstruujeme $A_1 \times A_2$ s množinou koncových stavů F (viz výše), pro který platí, že $L_1 = L_2$ právě tehdy, když $L(A_1 \times A_2) = \emptyset$ (což jak víme, je rozhodnutelný problém).

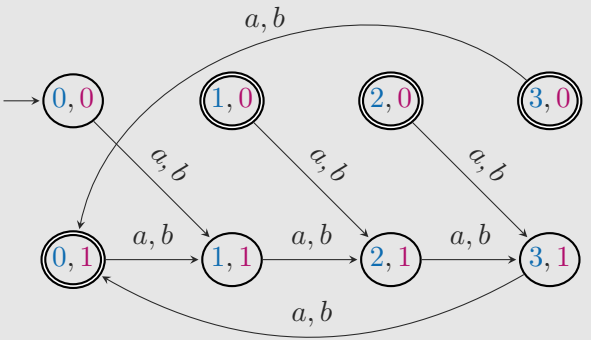
Platí: $w \in L(A_1 \times A_2) \neq \emptyset$ právě tehdy, když $\delta(\langle q_{0,1}, q_{0,2} \rangle, w) \in F$, což je ekvivalentní s:

1. $\delta_1(q_{0,1}, w) \in F_1$ a zároveň $\delta_2(q_{0,2}, w) \notin F_2$, a proto $w \in L_1 \wedge w \notin L_2$, nebo
2. $\delta_1(q_{0,1}, w) \notin F_1$ a zároveň $\delta_2(q_{0,2}, w) \in F_2$, a proto $w \in \bar{L}_1 \cap L_2$. □

Příklad: rovnost regulárních jazyků

Příklad

DFA $A_{mod} \times A_{\varepsilon}$ s množinou koncových stavů $F = \{\langle 0,1\rangle, \langle 1,0\rangle, \langle 2,0\rangle, \langle 3,0\rangle\}$ rozpoznává jazyk $(L_{mod} \cap \bar{L}_{\varepsilon}) \cup (\bar{L}_{mod} \cap L_{\varepsilon}) \neq \emptyset$, a proto $L_{mod} \neq L_{\varepsilon}$ (protipříklad $w = aaaa$):



Inkluze regulárních jazyků

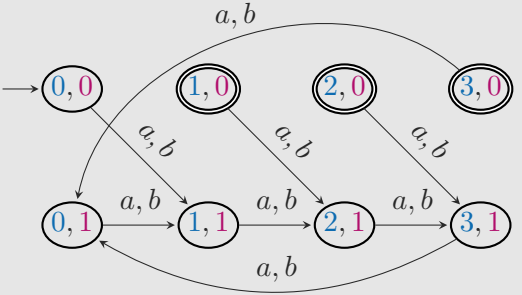
Intuice: analogicky jako u rovnosti, jen zvolíme $F = \{\langle q_1, q_2 \rangle \mid q_1 \in F_1 \wedge q_2 \notin F_2\}$.

Věta

Problém inkluze $L_1 \subseteq L_2$ dvou regulárních jazyků L_1 a L_2 je rozhodnutelný. □

Příklad

Součinový DFA $A_{mod} \times A_{\varepsilon}$ s množinou koncových stavů $F = \{\langle 1, 0 \rangle, \langle 2, 0 \rangle, \langle 3, 0 \rangle\}$ rozpoznává jazyk $\bar{L}_{mod} \cap L_{\varepsilon} = \emptyset$, a proto $L_{\varepsilon} \subseteq L_{mod}$ (tedy $L_2 \subseteq L_1$ dle Věty výše):

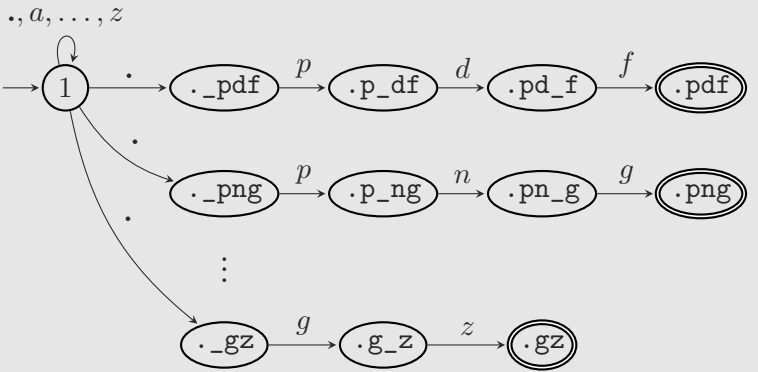


Nedeterministický konečný automat (NFA)

Motivační příklad pro nedeterminisus

Příklad

Chceme filtrovat soubory různých typů s odpovádající příponou, jako například `.pdf`, `.png`, `.gz`, `.tar.gz` atd., přičemž i samotné jméno souboru může obsahovat tečku.



Nedeterministický konečný automat

Intuice: zobecníme definici DFA o možnost být ve více stavech najednou.

Definice

Nedeterministický konečný automat (zkráceně NFA z *nondeterministic finite automaton*) je reprezentován uspořádanou pěticí $A = (Q, \Sigma, \delta, q_0, F)$, kde:

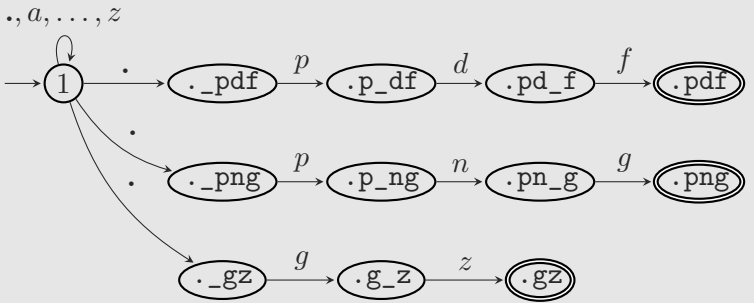
1. Q je konečná množina stavů,
2. Σ je abeceda,
3. δ je přechodová funkce ve tvaru $\delta : Q \times \Sigma \to 2^Q$, která stavu a symbolu přiřadí množinu stavů,
4. q_0 je počáteční stav (platí $q_0 \in Q$),
5. F je množina koncových/akceptujících stavů (platí $F \subseteq Q$).

- jediná změna oproti definici DFA je v přechodové funkci:
  - přechodová funkce nemusí být uplná, může tedy nastat $\delta(q, a) = \emptyset$ (protože $\emptyset \in 2^Q$)
  - z jednoho stavu může vést více přechodů pod stejným symbolem, například $\delta(q, a) = \{r, s\}$

Příklad: NFA pro jazyk přípon souborů

Příklad

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



- Znak `.` přesune A_{sfx} ze stavu 1 do čtyřech stavů $\delta(1, .) = \{1, \texttt{._pdf}, \texttt{._png}, \texttt{._gz}\}$,
- například ve stavu `._pdf` pro znak g není definován přechod, tj. $\delta(\texttt{._pdf}, g) = \emptyset$.

Rozšířená přechodová funkce

Motivace: vzhledem k tomu, že NFA se může nacházet ve více stavech najednou, musíme umět vypočítat přechod i z libovolné podmnožiny stavů.

Definice

Rozšířenou přechodovou funkci $\hat{\delta}\colon 2^Q \times \Sigma^* \to 2^Q$ definujeme pro dvojici množiny stavů $R \subseteq Q$ a řetězce $w = w_1 \dots w_n \in \Sigma^*$ následovně:

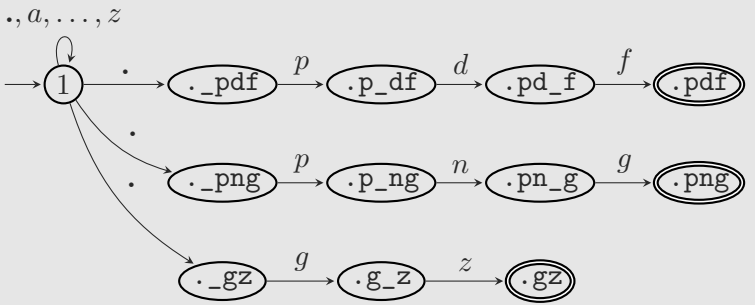
$$\hat{\delta}(R, w) = \begin{cases} R & \text{pro } w = \varepsilon \\ \hat{\delta}(\bigcup_{r \in R} \delta(r, w_1), w_2 \dots w_n) & \text{pro } w \neq \varepsilon \end{cases}$$

- $\hat{\delta}$ nám dává návod jak vypočítat množinu stavů dosažitelných nějakým řetězcem
- pro jednoprvkovou množinu $\{q\}$ budeme zápis zkracovat na $\hat{\delta}(q, w)$
- opět nebudeme rozlišovat mezi δ a $\hat{\delta}$, tj. budeme psát rovnou $\delta(q, w_1 w_2 w_3) = \{r, s\}$

Příklad: výpočet NFA

Příklad

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



- A_{sfx} přijímá řetězec $.gz$, protože $\hat{\delta}(1, .gz) \cap F \neq \emptyset$:
  $\hat{\delta}(1, .gz) = \hat{\delta}(\{1, \texttt{._pdf}, \texttt{._png}, \texttt{._gz}\}, gz) = \hat{\delta}(\{1, \texttt{.g\_z}\}, z) = \{1, \texttt{.gz}\}$,
- A_{sfx} zamítá řetězec $.c$, protože $\hat{\delta}(1, .c) = \{1\}$ a množina $\{1\}$ neobsahuje koncový stav.

Jazyk NFA

Intuice: NFA přijímá daný řetězec, pokud po jeho přečtení skončí alespoň v jednom koncovém stavu. Takto definované NFA rozhodují třídu regulárních jazyků (více později).

Definice

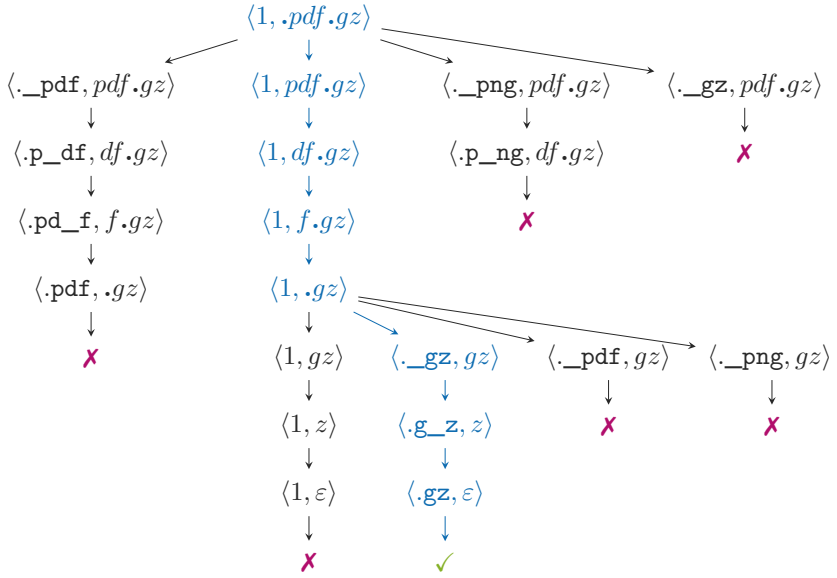
Jazyk NFA $A = (Q, \Sigma, \delta, q_0, F)$ definujeme jako $L(A) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$.

Alternativní pohled na nedeterminismus

- NFA se nachází pouze v jednom stavu (jako DFA), ale navíc umí správně "hádat":
  - pokud má NFA na výběr z více přechodů pod stejným symbolem, vybere si vždy správně ten, který jej nakonec dovede ke koncovému stav (pokud takový přechod existuje)
  - takový výpočet lze vizualizovat jako strom, kde uzly jsou konfigurace, a pokud existuje cesta od kořene stromu (počáteční konfigurace) do přijímací konfigurace, pak NFA řetězec přijímá
- toto chápání nedeterminismu je častější u složitějších modelů (zásobníkový automat, Turingův stroj atd.), které mezi stavy i modifikují svojí paměť

Příklad: alternativní pohled na nedeterminismus

- NFA A_{sfx} uhádne cestu z počáteční konfigurace $\langle 1, .pdf.gz \rangle$ do přijímací konfigurace

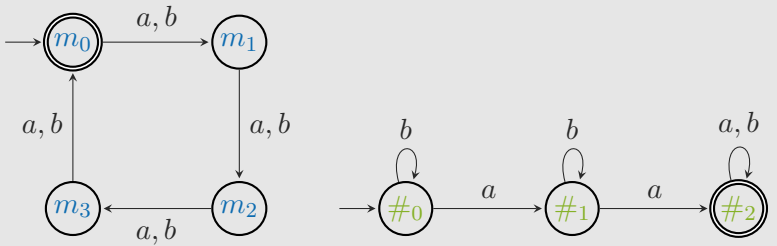


Příklad: NFA s množinou počátečních stavů

Intuice: nedeterministické automaty jsou v literatuře často definovány obecněji, a to s množinou počátečních stavů, tj. mějme NFA $A = (Q, \Sigma, \delta, I, F)$, kde $I \subseteq Q$.

Příklad

NFA $A_1 = (Q_{mod} \cup Q_{\#}, \{a, b\}, \delta_{mod} \cup \delta_{\#}, \{m_0, \#_0\}, \{m_0, \#_2\})$ s množinou počátečních stavů, který rozpoznává jazyk $L_{mod} \cup L_{\#}$.



- stavy původních automatů si nejdříve přejmenujeme tak, aby nedocházelo ke kolizím

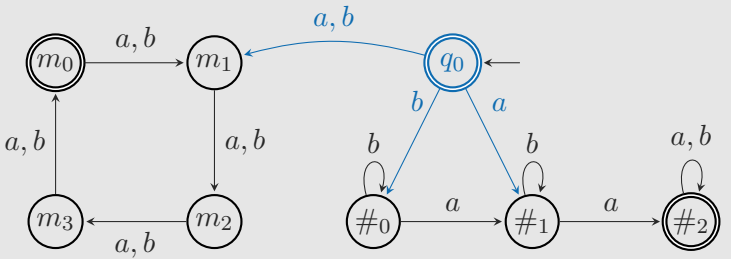
Příklad: NFA po redukci počátečních stavů

Více počátečních stavů NFA A lze lehce zredukovat na jeden

- přidáme nový počáteční stav q_0 (je zároveň i koncový pokud $\varepsilon \in L(A)$)
- pro každý přechod $\langle i, a, q \rangle \in \delta$, kde $i \in I$ je jeden z původních počátečních stavů, přidáme do δ nový přechod $\langle q_0, a, q \rangle$

Příklad

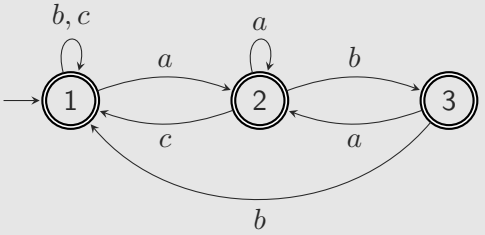
NFA $A_1' = (Q_{mod} \cup Q_{\#} \cup \{q_0\}, \{a, b\}, \delta', q_0, \{q_0, m_0, \#_2\})$ s jedním počátečním stavem, který rozpoznává jazyk $L_{mod} \cup L_{\#}$:



Příklad: jazyk bez konkrétního podřetězce

Příklad

NFA $A_{\not{abc}}$ pro jazyk $L_{\not{abc}} = \{w \in \{a,b,c\}^* \mid w \text{ neobsahuje podřetězec } abc\}$:



Princip fungovaní $A_{\not{abc}}$ na řetězci $w \in \{a,b,c\}^*$:

- pokud je řetězec abc sufixem řetězec w, pak $A_{\not{abc}}$ navždy "vypadne" ze všech stavů,
- v opačném případě je $A_{\not{abc}}$ v jednom z koncových stavů 1, 2 nebo 3.

Determinizace

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 $\{q_0\}$ je množina obsahující pouze původní počáteční stav q_0,
3. koncové stavy z $F_D = \{S \subseteq Q \mid S \cap F \neq \emptyset\}$ jsou podmnožiny Q, které obsahují alespoň jeden původní koncový stav,
4. $\delta_D \colon 2^Q \times \Sigma \to 2^Q$ je deterministická přechodová funkce definovaná jako:

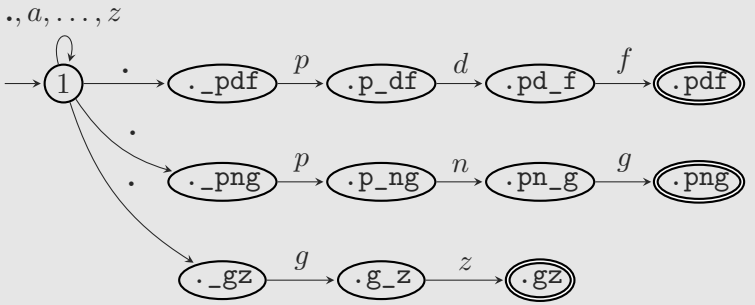
$$\delta_D(\{q_1, \ldots, q_k\}, a) = \bigcup_{i=1}^{k} \delta_N(q_i, a)$$

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

Příklad: NFA rozpoznávající přípony souborů

Příklad

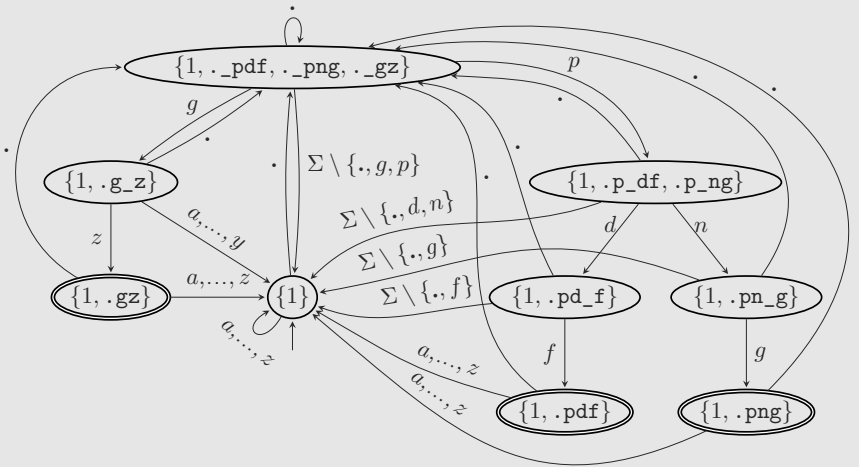
NFA $A_{sfx} = (Q, \{\texttt{.}, \texttt{a}, \ldots, \texttt{z}\}, \delta, 1, \{\texttt{.pdf}, \texttt{.png}, \texttt{.gz}\})$, který rozpoznává jazyk přípon souborů $L_{sfx} = \{w \mid w$ má příponu `.pdf`, `.png`, nebo `.gz`$\}$.



Příklad: podmnožinová konstrukce

Příklad

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

Základní krok: 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 indukčního 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 tehdy, když množiny $\hat{\delta}_D(\{q_0\}, w)$ a $\hat{\delta}_N(q_0, w)$ obsahují alespoň jeden koncový stav, pak platí $L(A) = L(A^{det})$. □

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 $L(A) = L(A')$. □

- 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

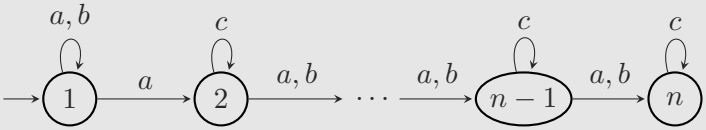
Mezní případ

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