

Logika a regulární jazyky

Václav Brožek

10. listopad 2010

Meta-poznámky

- dotazy a poznámky během přednášky vítány
- po přednášce rovněž vítány, např. na bleble@mail.muni.cz

Meta-poznámky

- dotazy a poznámky během přednášky vítány
- po přednášce rovněž vítány, např. na bleble@mail.muni.cz
- literatura:
Handbook of Theoretical Computer Science, Volume B:
Formal Models and Semantics, 1990 (signatura R269 v
knihovně FI)
... a mnoho dalších

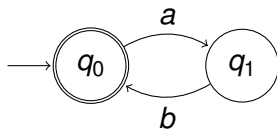
Osnova

- 1 Konečná slova
 - Logický popis jazyků
 - Souvislost s automaty
 - Složitost
 - Důsledky
 - Nekonečná slova
 - Rozdíly a podobnosti
 - Souvislost s automaty
 - Důsledky

Různé způsoby popisu jazyků

Popis operací

- 1 Jsi-li na konci, akceptuj.
- 2 Čti a .
- 3 Čti b .
- 4 Jdi na 1.

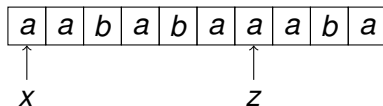


Popis výsledku

- První je a .
- Poslední je b .
- a a b se střídají.

?

Logika prvního řádu



$x, y, z \dots$ pozice

$P_a(x)$ na pozici x je písmeno a

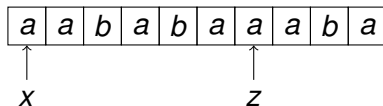
$x < y$ pozice x je před y

\wedge, \vee, \neg logické spojky (poznámka: negace víc podob)

$\forall x, \exists x$ kvantifikace

+ navíc rovnost $x = y$

Logika prvního řádu



$x, y, z \dots$ pozice

$P_a(x)$ na pozici x je písmeno a

$x < y$ pozice x je před y

\wedge, \vee, \neg logické spojky (poznámka: negace víc podob)

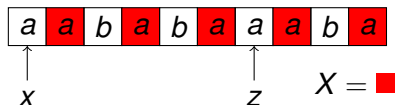
$\forall x, \exists x$ kvantifikace

+ navíc rovnost $x = y$

Cvičení

Co znamená $\exists x : P_a(x) \wedge (\forall z : z \neq x)$?

Monadická logika druhého řádu (MSO)



$x, y, z \dots$ pozice

$P_a(x)$ na pozici x je písmeno a

$x < y$ pozice x je před y

\wedge, \vee, \neg logické spojky

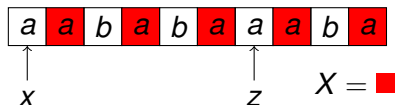
$\forall x, \exists x$ kvantifikace

$X, Y, Z \dots$ množiny pozic

$x \in X$ x leží v X

$\forall X, \exists X$

Monadická logika druhého řádu (MSO)



$x, y, z \dots$ pozice

$P_a(x)$ na pozici x je písmeno a

$x < y$ pozice x je před y

\wedge, \vee, \neg logické spojky

$\forall x, \exists x$ kvantifikace

$X, Y, Z \dots$ množiny pozic

$x \in X$ x leží v X

$\forall X, \exists X$

Cvičení

$$\begin{aligned}
 \exists X : & \quad [\forall x, y : ((x < y) \wedge (\forall z : x \not\prec z \vee z \not\prec y)) \\
 & \quad \implies (x \in X \iff y \notin X)] \\
 & \quad \wedge [\forall x : x \in X \implies (P_a(x) \wedge \exists y : y < x)]
 \end{aligned}$$

Sentence a formule MSO

Proměnné ve formuli dvojího typu:

- **vázané** pomocí kvantifikátorů
- **volné**

$$\phi(y, Y) \equiv \exists X : \forall x : y \in X \vee x \in Y$$

Volné se zpravidla píší za jméno formule.

sentence = uzavřená formule = formule bez volných proměnných

Jazyky zadané sentencí MSO

Sentence MSO ϕ = vlastnost slov.

Slovo $w \in \Sigma^*$ buď vlastnost ϕ má ($w \models \phi$), nebo nemá ($w \not\models \phi$).

Například:

$$aba \models \exists x : P_a(x) \wedge (\forall z : z \neq x)$$

$$bab \not\models \exists x : P_a(x) \wedge (\forall z : z \neq x)$$

$$\varepsilon \models \exists X : \forall x : \neg(\exists y : (x \in X \wedge y \notin X) \vee P_b(y))$$

Definice

$$L(\phi) := \{w \in \Sigma^* \mid w \models \phi\}$$

Pravdivost formulí

Kdy má slovo w vlastnost $\phi(X, Y, \dots, x, y, \dots)$?

Potřebujeme **valuaci**.

Pravdivost formulí

Kdy má slovo w vlastnost $\phi(X, Y, \dots, x, y, \dots)$?

Potřebujeme **valuaci**.

Valuace přiřadí hodnoty volným proměnným. Například pro:

$$\nu_1(x) = 1, \nu_1(y) = 1, \nu_1(X) = \{1, 3\}$$

$$\nu_2(x) = 3, \nu_2(y) = 1, \nu_2(X) = \emptyset$$

$$aba, \nu_1 \models P_a(x) \wedge (\forall z : z \notin x)$$

$$aba, \nu_2 \not\models P_a(x) \wedge (\forall z : z \notin x)$$

$$bab, \nu_1 \models \exists x, y : P_b(x) \wedge P_b(y) \wedge x < y \wedge x \in X \wedge y \in X$$

$$bab, \nu_2 \not\models \exists x, y : P_b(x) \wedge P_b(y) \wedge x < y \wedge x \in X \wedge y \in X$$

Osnova

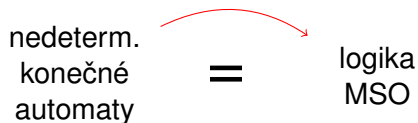
- 1 Konečná slova
 - Logický popis jazyků
 - Souvislost s automaty
 - Složitost
 - Důsledky
- Nekonečná slova
 - Rozdíly a podobnosti
 - Souvislost s automaty
 - Důsledky

Automaty = logika

Věta

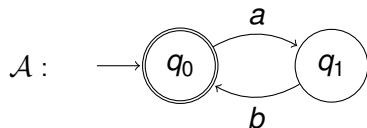
Regulární jazyky = jazyky definovatelné MSO.

Automaty = logika



Věta

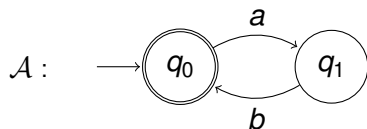
Regulární jazyky = jazyky definovatelné MSO.

Automat \rightarrow sentence MSO

$$L(\mathcal{A}) = L((ab)^*)$$

$w \in L(\mathcal{A})$ právě když

existuje akceptující běh \mathcal{A} pod w .

Automat \rightarrow sentence MSO

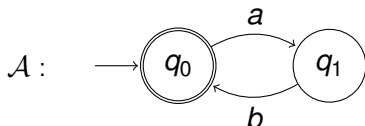
$$L(\mathcal{A}) = L((ab)^*)$$

$w \in L(\mathcal{A})$ právě když

$\exists \varrho : \{1, \dots, |w|\} \rightarrow \{q_0, q_1\}$ tak, že

- $q_0 \xrightarrow{w(1)} \varrho(1)$
- pro všechna i , $1 < i \leq |w|$: $\varrho(i-1) \xrightarrow{w(i)} \varrho(i)$
- $\varrho(|w|) \in F = \{q_0\}$

(nebo $w = \varepsilon$.)

Automat \rightarrow sentence MSO

$$L(\mathcal{A}) = L((ab)^*)$$

$w \in L(\mathcal{A})$ právě když

- $\exists X_0, X_1 : \forall x : x \in X_0 \iff x \notin X_1 \quad (\exists \varrho : \{1, \dots, |w|\} \rightarrow \{q_0, q_1\})$
- $\wedge \forall x : \mathit{First}(x) \implies (P_a(x) \wedge x \in X_1) \quad q_0 \xrightarrow{w(1)} \varrho(1)$
 - $\wedge \forall x, y : \mathit{Succ}(x, y) \implies$
 $(x \in X_0 \wedge P_a(y) \wedge y \in X_1) \vee (x \in X_1 \wedge P_b(y) \wedge y \in X_0)$
 pro všechna $i, 1 < i \leq |w| : \varrho(i-1) \xrightarrow{w(i)} \varrho(i)$
 - $\wedge \forall x : \mathit{Last}(x) \implies x \in X_0 \quad \varrho(|w|) \in F = \{q_0\}$

Cvičení

Zdefinujte následující formule (s volnými proměnnými) tak, aby pro každé slovo w a valuaci ν platilo

- $w, \nu \models \text{First}(x)$ právě když $\nu(x)$ je první pozice
- $w, \nu \models \text{Last}(x)$ právě když $\nu(x)$ je poslední pozice
- $w, \nu \models \text{Succ}(x, y)$ právě když $\nu(y) = \nu(x) + 1$

Automat \rightarrow sentence MSO – obecný postup

NKA $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F) \rightarrow$ sentence MSO ϕ t.ž. $L(\mathcal{A}) = L(\phi)$.

$$\phi := \exists \{X_q \mid q \in Q\} : \left(\forall x : \bigvee_{q \in Q} x \in X_q \wedge \bigwedge_{q \neq r \in Q} (x \in X_q \implies x \notin X_r) \right)$$

$$\wedge \forall x : \text{First}(x) \implies \bigvee_{a \in \Sigma, q_0 \xrightarrow{a} q} P_a(x) \wedge x \in X_q$$

$$\wedge \forall x, y : \text{Succ}(x, y) \implies \bigvee_{a \in \Sigma, q \xrightarrow{a} r} x \in X_q \wedge P_a(x) \wedge y \in X_r$$

$$\wedge \forall x : \text{Last}(x) \implies \bigvee_{q \in F} x \in X_q$$

Automat \rightarrow sentence MSO – obecný postup

NKA $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F) \rightarrow$ sentence MSO ϕ t.ž. $L(\mathcal{A}) = L(\phi)$.

$$\phi := \exists\{X_q \mid q \in Q\} : \left(\forall x : \bigvee_{q \in Q} x \in X_q \wedge \bigwedge_{q \neq r \in Q} (x \in X_q \implies x \notin X_r) \right)$$

$$\wedge \forall x : \text{First}(x) \implies \bigvee_{a \in \Sigma, q_0 \xrightarrow{a} q} P_a(x) \wedge x \in X_q$$

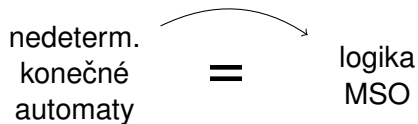
$$\wedge \forall x, y : \text{Succ}(x, y) \implies \bigvee_{a \in \Sigma, q \xrightarrow{a} r} x \in X_q \wedge P_a(x) \wedge y \in X_r$$

$$\wedge \forall x : \text{Last}(x) \implies \bigvee_{q \in F} x \in X_q$$

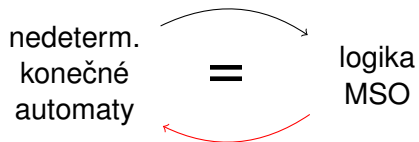
Cvičení

Najděte chybu.

Automaty = logika



Automaty = logika



Sentence MSO \rightarrow automat

Chceme zkonstruovat pro každou

sentenci MSO ϕ

nedeterministický KA \mathcal{A}

tak, že $L(\phi) = L(\mathcal{A})$, indukcí ke struktuře ϕ .

Sentence MSO \rightarrow automat

Chceme zkonstruovat pro každou

sentenci MSO ϕ

nedeterministický KA \mathcal{A}

tak, že $L(\phi) = L(\mathcal{A})$, indukcí ke struktuře ϕ .

Ale podformule ϕ nemusí být sentence!

$\exists x : x < x$ má podformuli $x < x$.

Formule MSO \rightarrow automat

Chceme zkonstruovat pro každou

formuli MSO $\phi(X, \dots, x, \dots)$

nedeterministický KA \mathcal{A}

tak, že $L(\phi(X, \dots, x, \dots)) = L(\mathcal{A})$, indukcí ke struktuře ϕ .

„Čistá” MSO – logika bez prvního řádu

Pro zjednodušení konstrukce odstraníme proměnné 1. řádu:

MSO	„Čistá” MSO
$x, X \dots$ proměnné	$X \dots$ proměnné (jen 2. řádu)
$P_a(x)$ a na pozici x	$P_a(X)$ a na všech pozicích z X
$x \in X$ x leží v X	$X \subseteq Y$ X podmnožina Y
$x < y$ pozice x je před y	$X < Y$ něco z X je před Y
\wedge, \vee, \neg logické spojky	\wedge, \vee, \neg logické spojky
$\forall x, \exists X$ kvantifikace	$\exists X$ kvantifikace (jen 2. řád)
	$Sng(X)$ $ X = 1$

Ekvivalence MSO a čisté MSO

Zápis čisté MSO v MSO:

$$P_a(X) \equiv \forall x : x \in X \implies P_a(x)$$

$$X \subseteq Y \equiv \forall x : x \in X \implies x \in Y$$

$$\text{Sng}(X) \equiv \exists x : x \in X \wedge \forall y : y = x \vee y \notin X$$

$$X < Y \equiv \exists x : x \in X \wedge (\forall y : y \in Y \implies x < y)$$

Zápis MSO v čisté MSO:

Každou x nahradíme S_x a přidáme $\wedge \text{Sng}(S_x)$.

$$P_a(x) \equiv P_a(S_x)$$

$$x \in X \equiv S_x \subseteq X$$

$$x < y \equiv S_x < S_y$$

Formule čisté MSO \rightarrow automat

Chceme zkonstruovat pro každou

formuli čisté MSO $\phi(X, \dots)$

nedeterministický KA \mathcal{A}

tak, že $L(\phi(X, \dots)) = L(\mathcal{A})$, indukcí ke struktuře ϕ .

Co je $L(\phi(X, \dots))$? Aneb kódování valuací

Pro formuli $\phi(X_1, \dots, X_k)$ a slovo $w \in \Sigma^*$ zakódujeme valuaci ν jako slovo $w_\nu \in (\Sigma \times \{0, 1\}^k)^*$. Například:

$k = 2$, $w = abc$, $\nu : X_1 \mapsto \{1, 3\}$, $X_2 \mapsto \{2\}$ dá

$$w_\nu = \begin{bmatrix} a \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} b \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} c \\ 1 \\ 0 \end{bmatrix}$$

Co je $L(\phi(X, \dots))$? Aneb kódování valuací

Pro formuli $\phi(X_1, \dots, X_k)$ a slovo $w \in \Sigma^*$ zakódujeme valuaci ν jako slovo $w_\nu \in (\Sigma \times \{0, 1\}^k)^*$. Například:

$k = 2$, $w = abc$, $\nu : X_1 \mapsto \{1, 3\}$, $X_2 \mapsto \{2\}$ dá

$$w_\nu = \begin{bmatrix} a \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} b \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} c \\ 1 \\ 0 \end{bmatrix}$$

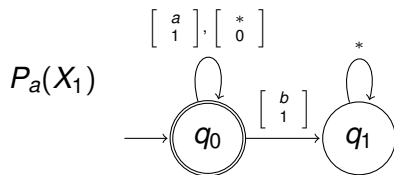
Definice

$$L(\phi(X_1, \dots, X_k)) := \{w_\nu \in (\Sigma \times \{0, 1\}^k)^* \mid w, \nu \models \phi(X_1, \dots, X_k)\}$$

Formule čisté MSO \rightarrow automat

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F)$.

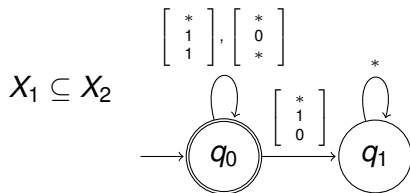
Bázové kroky:



Formule čisté MSO \rightarrow automat

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F)$.

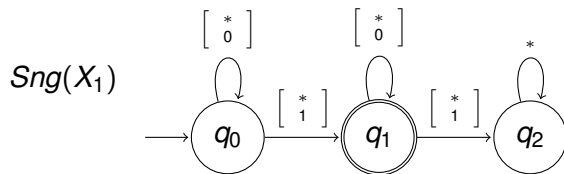
Bázové kroky:



Formule čisté MSO \rightarrow automat

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \rightarrow, F)$.

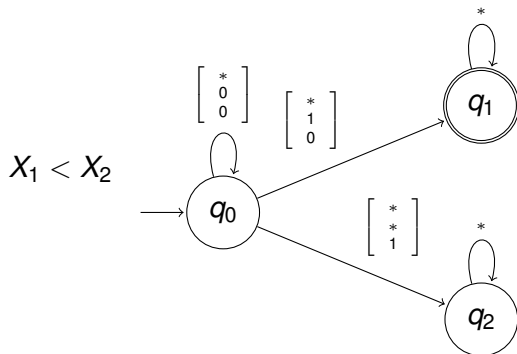
Bázové kroky:



Formule čisté MSO \rightarrow automat

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \rightarrow, F)$.

Bázové kroky:



Formule čisté MSO \rightarrow automat

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \rightarrow, F)$.

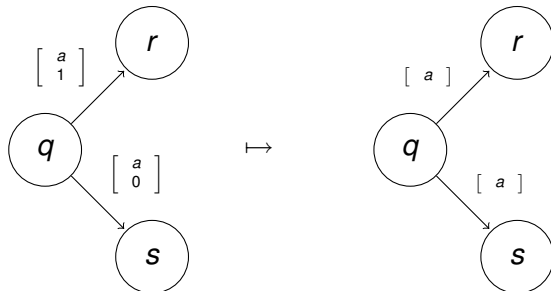
Logické operace \vee, \wedge, \neg přejdou na \cup, \cap a komplement.

Formule čisté MSO \rightarrow automat

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F)$.

Kvantifikace:

$\exists X_1 : \phi(X_1)$

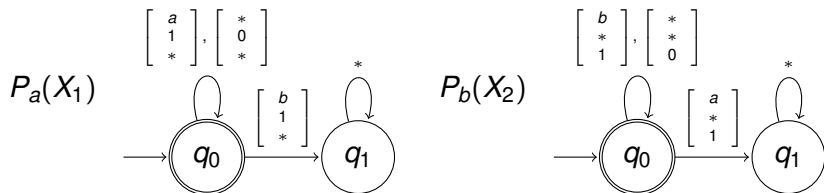


Příklad

$$\neg(\exists X_1, X_2 : P_a(X_1) \wedge P_b(X_2))$$

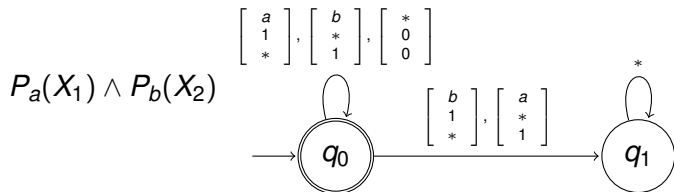
Příklad

$$\neg(\exists X_1, X_2 : P_a(X_1) \wedge P_b(X_2))$$



Příklad

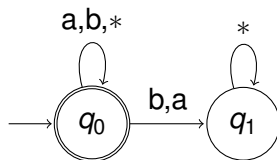
$$\neg(\exists X_1, X_2 : P_a(X_1) \wedge P_b(X_2))$$



Příklad

$$\neg(\exists X_1, X_2 : P_a(X_1) \wedge P_b(X_2))$$

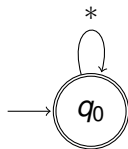
$$\exists X_1, X_2 : P_a(X_1) \wedge P_b(X_2)$$



Příklad

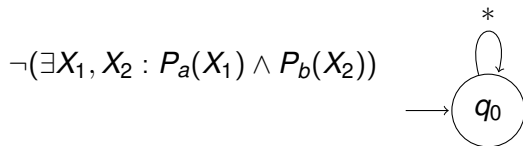
$$\neg(\exists X_1, X_2 : P_a(X_1) \wedge P_b(X_2))$$

$$\exists X_1, X_2 : P_a(X_1) \wedge P_b(X_2)$$



Příklad

$$\neg(\exists X_1, X_2 : P_a(X_1) \wedge P_b(X_2))$$



Poznámky

Existuje i alternativní důkaz přes *rank* formulí.

- Rank formule = maximální počet zanoření kvantifikátorů.
- Existuje jen konečně mnoho formulí ranku $\leq k$ až na ekvivalenci – třída ekvivalentních formulí = *typ*.
- Existuje automat počítající typy pro fixní rank.

Tento důkaz se vyhne indukci na formulích, ale nedává vzhled do detailů převodu.

Poznámky

- Aplikací převodů: sentence \rightarrow automat \rightarrow sentence dostaneme pro každou ϕ její normální formu:

$$\phi \equiv \exists X_1, \dots, X_k : \psi(X_1, \dots, X_k)$$

kde ψ je bez kvantifikátorů druhého řádu.

Poznámky

- Aplikací převodů: sentence \rightarrow automat \rightarrow sentence dostaneme pro každou ϕ její normální formu:

$$\phi \equiv \exists X_1, \dots, X_k : \psi(X_1, \dots, X_k)$$

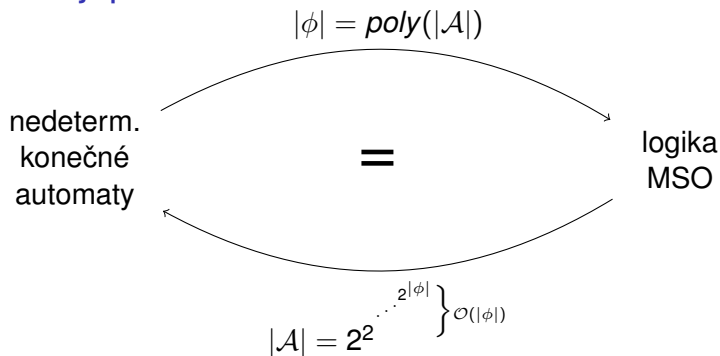
kde ψ je bez kvantifikátorů druhého řádu.

- splnitelnost pro MSO rozhodnutelná nejen nad slovy, ale i nad stromy. Ovšem už pro acyklické grafy je nerozhodnutelná.

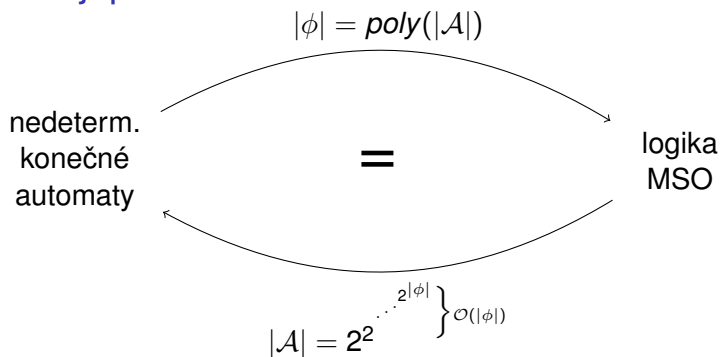
Osnova

- 1 Konečná slova
 - Logický popis jazyků
 - Souvislost s automaty
 - Složitost
 - Důsledky
- Nekonečná slova
 - Rozdíly a podobnosti
 - Souvislost s automaty
 - Důsledky

Kolik stojí převod



Kolik stojí převod



$$\neg(\exists X_1 : \neg(\exists X_2 : \neg(\dots \neg(\exists X_k : \phi(X_1, X_2, \dots, X_k)) \dots)))$$

- \exists vytváří nedeterminismus
- \neg = komplementace \rightarrow exponenciální nárůst

Drahý převod: formule \rightarrow automat

Definice

$G(1) := 1$, a pro všechna $n > 1$: $G(n) := G(n - 1) \cdot 2^{G(n-1)}$

Drahý převod: formule \rightarrow automat

Definice

$G(1) := 1$, a pro všechna $n > 1$: $G(n) := G(n-1) \cdot 2^{G(n-1)}$

Cvičení

Dokažte: $\forall n \geq 1 : 2^{\left. 2^{\dots^2} \right\}^{n+1}} > G(n) \geq 2^{\left. 2^{\dots^2} \right\}^n$.

Drahý převod: formule \rightarrow automat

Definice

$G(1) := 1$, a pro všechna $n > 1$: $G(n) := G(n-1) \cdot 2^{G(n-1)}$

Cvičení

Dokažte: $\forall n \geq 1 : 2^{\left. 2^{\dots^2} \right\}^{n+1}} > G(n) \geq 2^{\left. 2^{\dots^2} \right\}^n$.

Věta (Stockmeyer & Meyer, 1974)

Existuje posloupnost MSO sentencí $\{\phi_n\}_{n=1}^{\infty}$ tak, že:

- $|\phi_n| \in \mathcal{O}(2^n)$
- pro každý nedeterministický KA \mathcal{A}_n :
 $L(\mathcal{A}_n) = L(\phi_n) \implies |\mathcal{A}_n| \geq G(n)$

Důkaz

Definujeme:

$$L_n := \{\overbrace{aaa \cdots a}^{G(n)}\}$$

Důkaz

Definujeme:

$$L_n := \{\overbrace{aaa \cdots a}^{G(n)}\}$$

Cvičení

Dokažte:

- 1 L_n je regulární.

Důkaz

Definujeme:

$$L_n := \{\overbrace{aaa \cdots a}^{G(n)}\}$$

Cvičení

Dokažte:

- 1 L_n je regulární.
- 2 pro každý *nedeterministický* KA \mathcal{A}_n :
 $L(\mathcal{A}_n) = L_n \implies |\mathcal{A}_n| \geq G(n)$

Důkaz

Definujeme:

$$L_n := \{\overbrace{aaa \cdots a}^{G(n)}\}$$

Cvičení

Dokažte:

- 1 L_n je regulární.
- 2 pro každý *nedeterministický* KA \mathcal{A}_n :
 $L(\mathcal{A}_n) = L_n \implies |\mathcal{A}_n| \geq G(n)$

Zbývá najít formuli ϕ_n tak, aby

- $|\phi_n| \in \mathcal{O}(2^n)$
- $L_n = L(\phi_n)$

- Indukcí na n sestrojíme formule $\theta_n(x, y)$ tak, že

$$w, \nu \models \theta_n(x, y) \quad \text{právě když} \quad \nu(y) - (\nu(x) - 1) = G(n).$$

- Pak $\phi_n := \exists x, y : \text{First}(x) \wedge \text{Last}(y) \wedge \theta_n(x, y)$.

- Indukcí na n sestrojíme formule $\theta_n(x, y)$ tak, že

$$w, \nu \models \theta_n(x, y) \quad \text{právě když} \quad \nu(y) - (\nu(x) - 1) = G(n).$$

- Pak $\phi_n := \exists x, y : \text{First}(x) \wedge \text{Last}(y) \wedge \theta_n(x, y)$.

- indukční báze je jednoduchá:

$$\theta_1(x, y) := x = y$$



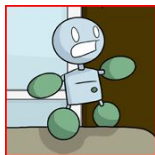
- Indukcí na n sestrojíme formule $\theta_n(x, y)$ tak, že

$$w, \nu \models \theta_n(x, y) \quad \text{právě když} \quad \nu(y) - (\nu(x) - 1) = G(n).$$

- Pak $\phi_n := \exists x, y : \text{First}(x) \wedge \text{Last}(y) \wedge \theta_n(x, y)$.

- indukční báze je jednoduchá:

$$\theta_1(x, y) := x = y$$



questionablecontent.net

indukční krok je složitější...

$\theta_{n+1}(x, y) := \exists Y : Y$ zadává intervaly na $\{x, \dots, y\}$
 \wedge tyto intervaly jsou délky $G(n)$
 \wedge intervalů je $2^{G(n)}$

$$\begin{aligned} \theta_{n+1}(x, y) := & \exists Y : Y \text{ zadává intervaly na } \{x, \dots, y\} \\ & \wedge \text{ tyto intervaly jsou délky } G(n) \\ & \wedge \text{ intervalů je } 2^{G(n)} \end{aligned}$$

Kódování intervalů pomocí množiny pozic Y :

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

$$Y = \blacktriangledown$$

zadané intervaly: $\{1, 2, 3\}$, $\{4\}$, $\{5, 6, 7, 8, 9, 10\}$.

Formule $Int(r, s, Y)$ testuje, že $\{r, \dots, s\}$ je interval dle Y :

$$\begin{aligned} Int(r, s, Y) := & r \in Y \wedge r \leq s \\ & \wedge (\forall z : Succ(s, z) \implies z \in Y) \\ & \wedge (\forall z : r < z \leq s \implies z \notin Y) \end{aligned}$$

Y zadává intervaly na $\{x, \dots, y\}$

\equiv

$$x \in Y \wedge \forall z : \text{Succ}(y, z) \implies z \in Y$$

tyto intervaly jsou délky $G(n)$

\equiv

$$\forall r, s : (x \leq r, s \leq y \wedge \text{Int}(r, s, Y)) \implies \theta_n(r, s)$$

intervalů je $2^{G(n)}$

≡

∃ J : první interval kóduje 0

∧ poslední interval kóduje $2^{G(n)} - 1$

∧ po sobě jdoucí intervaly kódují přičtení 1

intervalů je $2^{G(n)}$

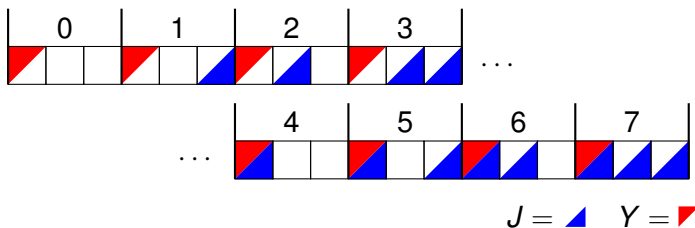
\equiv

$\exists J$: první interval kóduje 0

\wedge poslední interval kóduje $2^{G(n)} - 1$

\wedge po sobě jdoucí intervaly kódují přičtení 1

Kódování čísel pomocí množiny J :



první interval kóduje 0

≡

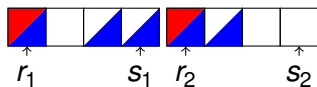
$$\forall s, t : (\text{Int}(x, s, Y) \wedge x \leq t \leq s) \implies t \notin J$$

poslední interval kóduje $2^{G(n)} - 1$

≡

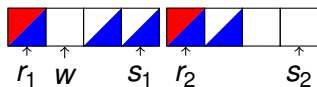
$$\forall r, t : (\text{Int}(r, y, Y) \wedge r \leq t \leq y) \implies t \in J$$

po sobě jdoucí intervaly kódují přičtení 1



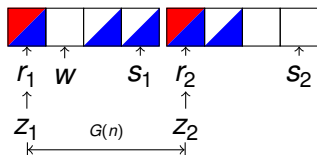
$$\forall r_1, s_1, r_2, s_2 : \left(x \leq r_1, s_1, r_2, s_2 \leq y \right. \\ \left. \wedge \text{Int}(r_1, s_1, Y) \wedge \text{Int}(r_2, s_2, Y) \wedge \text{Succ}(s_1, r_2) \right)$$

po sobě jdoucí intervaly **kódují přičtení 1**



$$\forall r_1, s_1, r_2, s_2 : \left(x \leq r_1, s_1, r_2, s_2 \leq y \right. \\ \left. \wedge \text{Int}(r_1, s_1, Y) \wedge \text{Int}(r_2, s_2, Y) \wedge \text{Succ}(s_1, r_2) \right) \\ \implies \exists w : \left[r_1 \leq w \leq s_1 \wedge w \notin J \wedge (\forall z : w < z \leq s_1 \implies z \in J) \right]$$

po sobě jdoucí intervaly **kódují přičtení 1**



$$\forall r_1, s_1, r_2, s_2 : (x \leq r_1, s_1, r_2, s_2 \leq y$$

$$\wedge \text{Int}(r_1, s_1, Y) \wedge \text{Int}(r_2, s_2, Y) \wedge \text{Succ}(s_1, r_2))$$

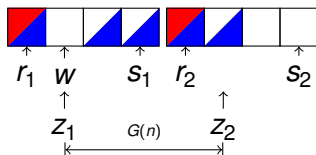
$$\implies \exists w : \left[r_1 \leq w \leq s_1 \wedge w \notin J \wedge (\forall z : w < z \leq s_1 \implies z \in J) \right]$$

$$\wedge \forall z_1, z_2 : \left((r_1 \leq z_1 \leq s_1 \wedge \exists t : (\theta_n(z_1, t) \wedge \text{Succ}(t, z_2))) \right)$$

$$\implies ((z_1 < w \wedge (z_1 \in J \iff z_2 \in J))$$

$$\vee (z_1 = w \wedge z_2 \in J) \vee (z_1 > w \wedge z_2 \notin J)) \Big]$$

po sobě jdoucí intervaly **kódují přičtení 1**



$$\forall r_1, s_1, r_2, s_2 : (x \leq r_1, s_1, r_2, s_2 \leq y$$

$$\wedge \text{Int}(r_1, s_1, Y) \wedge \text{Int}(r_2, s_2, Y) \wedge \text{Succ}(s_1, r_2))$$

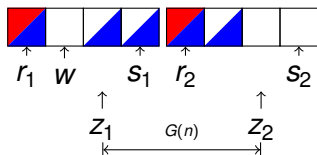
$$\implies \exists w : \left[r_1 \leq w \leq s_1 \wedge w \notin J \wedge (\forall z : w < z \leq s_1 \implies z \in J) \right]$$

$$\wedge \forall z_1, z_2 : \left((r_1 \leq z_1 \leq s_1 \wedge \exists t : (\theta_n(z_1, t) \wedge \text{Succ}(t, z_2))) \right)$$

$$\implies ((z_1 < w \wedge (z_1 \in J \iff z_2 \in J))$$

$$\vee (z_1 = w \wedge z_2 \in J) \vee (z_1 > w \wedge z_2 \notin J)) \Big]$$

po sobě jdoucí intervaly **kódují přičtení 1**



$$\forall r_1, s_1, r_2, s_2 : \left(x \leq r_1, s_1, r_2, s_2 \leq y \right.$$

$$\wedge \text{Int}(r_1, s_1, Y) \wedge \text{Int}(r_2, s_2, Y) \wedge \text{Succ}(s_1, r_2) \Big)$$

$$\implies \exists w : \left[r_1 \leq w \leq s_1 \wedge w \notin J \wedge (\forall z : w < z \leq s_1 \implies z \in J) \right.$$

$$\wedge \forall z_1, z_2 : \left((r_1 \leq z_1 \leq s_1 \wedge \exists t : (\theta_n(z_1, t) \wedge \text{Succ}(t, z_2))) \right.$$

$$\implies ((z_1 < w \wedge (z_1 \in J \iff z_2 \in J))$$

$$\left. \left. \vee (z_1 = w \wedge z_2 \in J) \vee (z_1 > w \wedge z_2 \notin J) \right) \right]$$

Poznámky k důkazu

$$\begin{aligned}\theta_{n+1}(x, y) &:= \exists Y : Y \text{ zadává intervaly na } \{x, \dots, y\} \\ &\wedge \text{ tyto intervaly jsou délky } G(n) (\dots \theta_n \dots) \\ &\wedge \text{ intervalů je } 2^{G(n)} (\dots \theta_n \implies \dots)\end{aligned}$$

- velikost formule exponenciální v n
- vnořená negace $((\theta_n \implies \alpha) \equiv (\neg\theta_n \vee \alpha))$

Přehled složitostí

nedeterministické KA		MSO	
$L(\mathcal{A}) \stackrel{?}{=} \emptyset$	NLOG-úplné	$L(\phi) \stackrel{?}{=} \emptyset$	
$L(\mathcal{A}) \stackrel{?}{=} \Sigma^*$	PSPACE-úplné	$L(\phi) \stackrel{?}{=} \Sigma^*$	non-elementary
$L(\mathcal{A}) \stackrel{?}{=} L(\mathcal{B})$	PSPACE-úplné	$L(\phi) \stackrel{?}{=} L(\psi)$	
$L(\mathcal{A}) \stackrel{?}{\subseteq} L(\mathcal{B})$	PSPACE-úplné	$L(\phi) \stackrel{?}{\subseteq} L(\psi)$	

Osnova

- 1 Konečná slova
 - Logický popis jazyků
 - Souvislost s automaty
 - Složitost
 - Důsledky
- Nekonečná slova
 - Rozdíly a podobnosti
 - Souvislost s automaty
 - Důsledky

Podtřídy regulárních jazyků

Jazyky **definované** logikou prvního řádu (**FO**) tvoří podtřídu MSO jazyků uzavřenou na Booleovské operace.

Jazyky definované regulárními výrazy bez Kleeneho *, ale s komplementem, tvoří podtřídu MSO jazyků uzavřenou na Booleovské operace, **star-free jazyky**.

Podtřídy regulárních jazyků

Jazyky **definované** logikou prvního řádu (**FO**) tvoří podtřídu MSO jazyků uzavřenou na Booleovské operace.

Jazyky definované regulárními výrazy bez Kleeneho *, ale s komplementem, tvoří podtřídu MSO jazyků uzavřenou na Booleovské operace, **star-free jazyky**.

Věta (McNaughton-Pappert)

Jazyk L je star-free právě když je FO definovatelný.

Podtřídy regulárních jazyků

Jazyky **definované** logikou prvního řádu (**FO**) tvoří podtřídu MSO jazyků uzavřenou na Booleovské operace.

Jazyky definované regulárními výrazy bez Kleeneho *, ale s komplementem, tvoří podtřídu MSO jazyků uzavřenou na Booleovské operace, **star-free jazyky**.

Věta (McNaughton-Pappert)

Jazyk L je star-free právě když je FO definovatelný.

Věta (Schützenberger)

Jazyk L je star-free právě když jeho syntaktický monoid neobsahuje podgrupu velikosti ≥ 2 .

Syntaktický monoid – definice přes Google

Monoid podle Google Image search:



Syntaktický monoid – definice pro matematiky

Syntaktická ekvivalence pro jazyk $L \subseteq \Sigma^*$:

$$u \equiv_L v \stackrel{\text{def}}{\iff} \forall p, q : p \cdot u \cdot q \in L \iff p \cdot v \cdot q \in L$$

Syntaktický monoid – definice pro matematiky

Syntaktická ekvivalence pro jazyk $L \subseteq \Sigma^*$:

$$u \equiv_L v \stackrel{\text{def}}{\iff} \forall p, q : p \cdot u \cdot q \in L \iff p \cdot v \cdot q \in L$$

Monoid \mathcal{M}_L pro L :

- prvky = Σ^* / \equiv_L
- operace: $[u]_{\equiv_L} \cdot [v]_{\equiv_L} = [u \cdot v]_{\equiv_L}$
- neutrální prvek: $[\varepsilon]_{\equiv_L}$

Syntaktický monoid – definice pro matematiky

Syntaktická ekvivalence pro jazyk $L \subseteq \Sigma^*$:

$$u \equiv_L v \stackrel{\text{def}}{\iff} \forall p, q : p \cdot u \cdot q \in L \iff p \cdot v \cdot q \in L$$

Monoid \mathcal{M}_L pro L :

- prvky = Σ^* / \equiv_L
- operace: $[u]_{\equiv_L} \cdot [v]_{\equiv_L} = [u \cdot v]_{\equiv_L}$
- neutrální prvek: $[\varepsilon]_{\equiv_L}$

\mathcal{M}_L je konečný, právě když L je regulární.

Syntaktický monoid – definice přes sousední balkon

Prefixová ekvivalence pro jazyk $L \subseteq \Sigma^*$:

$$u \approx_L v \stackrel{\text{def}}{\iff} \forall w : u \cdot w \in L \iff v \cdot w \in L$$

Syntaktický monoid – definice přes sousední balkon

Prefixová ekvivalence pro jazyk $L \subseteq \Sigma^*$:

$$u \approx_L v \stackrel{\text{def}}{\iff} \forall w : u \cdot w \in L \iff v \cdot w \in L$$

Automat \mathcal{A}_L pro L :

- stavy: $Q := \Sigma^* / \approx_L$
- přechody: $[u]_{\approx_L} \xrightarrow{a} [u \cdot a]_{\approx_L}$
- iniciální stav: $[\varepsilon]_{\approx_L}$
- koncové stavy: $[u]_{\approx_L}, u \in L$

Syntaktický monoid – definice přes sousední balkon

Prefixová ekvivalence pro jazyk $L \subseteq \Sigma^*$:

$$u \approx_L v \stackrel{\text{def}}{\iff} \forall w : u \cdot w \in L \iff v \cdot w \in L$$

Automat \mathcal{A}_L pro L :

- stavy: $Q := \Sigma^* / \approx_L$
- přechody: $[u]_{\approx_L} \xrightarrow{a} [u \cdot a]_{\approx_L}$
- iniciální stav: $[\varepsilon]_{\approx_L}$
- koncové stavy: $[u]_{\approx_L}, u \in L$

\mathcal{A}_L je deterministický automat, a je konečný, právě když L je regulární (pak je \mathcal{A}_L minimální).

Syntaktický monoid – definice přes sousední balkon

Prefixová ekvivalence pro jazyk $L \subseteq \Sigma^*$:

$$u \approx_L v \stackrel{\text{def}}{\iff} \forall w : u \cdot w \in L \iff v \cdot w \in L$$

Automat \mathcal{A}_L pro L :

- stavy: $Q := \Sigma^* / \approx_L$
- přechody: $[u]_{\approx_L} \xrightarrow{a} [u \cdot a]_{\approx_L}$
- iniciální stav: $[\varepsilon]_{\approx_L}$
- koncové stavy: $[u]_{\approx_L}, u \in L$

\mathcal{A}_L je deterministický automat, a je konečný, právě když L je regulární (pak je \mathcal{A}_L minimální).

$\mathcal{M}_L \cong T(\mathcal{A}_L)$, což je transformační monoid \mathcal{A}_L

- prvky: $z Q \rightarrow Q$, tvaru f_w , kde $f_w(q) = r$ tak, že $q \xrightarrow{w} r$
- operace: $f_v \circ f_u = f_{u \cdot v}$
- neutrální prvek: f_ε

Cvičení

Najděte MSO formuli pro $L((aa)^*)$.

Popište jeho syntaktický monoid.

Rozhodněte, zda je tento jazyk definovatelný v FO.

Cvičení

Najděte MSO formuli pro $L((ab)^*)$.

Popište jeho syntaktický monoid.

Rozhodněte, zda je tento jazyk definovatelný v FO.

Poznámka

Logika nabízí ještě jemnější klasifikaci:
hloubka alternací FO kvantifikátorů odpovídá
hloubce alternací booleovských operací a řetězení v
regulárních výrazech (s negací a bez *).

Viz: W. Thomas, Classifying regular events in symbolic logic, J.
Comput. System Sci., 1982

wS1S

Změníme trochu uvažovanou logiku:

Syntax: místo relace $<$ (následník) máme relaci *Succ* (bezprostřední následník).

Sémantika:

- vyhodnocování nad \mathbb{N} místo nad konečnými slovy
- kvantifikace jen přes **konečné** množiny

wS1S

Změníme trochu uvažovanou logiku:

Syntax: místo relace $<$ (následník) máme relaci *Succ* (bezprostřední následník).

Sémantika:

- vyhodnocování nad \mathbb{N} místo nad konečnými slovy
- kvantifikace jen přes **konečné** množiny

wS1S (Weak Second-order + 1 Successor)
=
množina všech sentencí pravdivých na $(\mathbb{N}, +1)$

Příklady

- $\exists x : \forall y : \exists z : y = x \vee Succ(z, y) \in \text{wS1S}$
Existuje číslo x tak, že každé číslo jiné než x má předchůdce.

Příklady

- $\exists x : \forall y : \exists z : y = x \vee Succ(z, y) \in \text{wS1S}$
Existuje číslo x tak, že každé číslo jiné než x má předchůdce.
- $\exists X : \forall x, y : (x \in X \wedge Succ(x, y)) \implies y \in X \in \text{wS1S}$
Existuje nahoru uzavřená *konečná* podmnožina \mathbb{N} .

Příklady

- $\bullet \exists x : \forall y : \exists z : y = x \vee Succ(z, y) \in \text{wS1S}$
 Existuje číslo x tak, že každé číslo jiné než x má předchůdce.
- $\bullet \exists X : \forall x, y : (x \in X \wedge Succ(x, y)) \implies y \in X \in \text{wS1S}$
 Existuje nahoru uzavřená *konečná* podmnožina \mathbb{N} .
- $\bullet \exists X : ((\exists x : x \in X) \wedge \forall x, y : (x \in X \wedge Succ(x, y)) \implies y \in X) \notin \text{wS1S}$
 Existuje neprázdňá nahoru uzavřená *konečná* podmnožina \mathbb{N} .

Příklady

- $\bullet \exists x : \forall y : \exists z : y = x \vee Succ(z, y) \in \text{wS1S}$
 Existuje číslo x tak, že každé číslo jiné než x má předchůdce.
- $\bullet \exists X : \forall x, y : (x \in X \wedge Succ(x, y)) \implies y \in X \in \text{wS1S}$
 Existuje nahoru uzavřená *konečná* podmnožina \mathbb{N} .
- $\bullet \exists X : ((\exists x : x \in X) \wedge \forall x, y : (x \in X \wedge Succ(x, y)) \implies y \in X) \notin \text{wS1S}$
 Existuje neprázdná nahoru uzavřená *konečná* podmnožina \mathbb{N} .

Cvičení

Zadefinujte $<$ pomocí *Succ*.

Příklady

- $\exists x : \forall y : \exists z : y = x \vee Succ(z, y) \in \text{wS1S}$
Existuje číslo x tak, že každé číslo jiné než x má předchůdce.
- $\exists X : \forall x, y : (x \in X \wedge Succ(x, y)) \implies y \in X \in \text{wS1S}$
Existuje nahoru uzavřená *konečná* podmnožina \mathbb{N} .
- $\exists X : ((\exists x : x \in X) \wedge \forall x, y : (x \in X \wedge Succ(x, y)) \implies y \in X) \notin \text{wS1S}$
Existuje neprázdna nahoru uzavřená *konečná* podmnožina \mathbb{N} .

Cvičení

Zadefinujte $<$ pomocí *Succ*.

- $\forall X : \exists t : \forall x : x \in X \implies x < t \in \text{wS1S}$
Každá (*konečná*) množina je konečná. :-)

Rozhodnutelnost wS1S

Věta (Büchi)

Lze rozhodnout, zda $\phi \in wS1S$.

Rozhodnutelnost wS1S

Věta (Büchi)

Lze rozhodnout, zda $\phi \in wS1S$.

Důkaz:

- 1 nahradíme *Succ* ekvivalentní definicí pomocí $<$
+ protože nepotřebujeme predikáty P_a , máme $|\Sigma| = 1$
- 2 uvědomíme si, že pro všechny formule $\psi(X_1, \dots, X_k)$ a
valuace ν přiřazující jen konečné množiny:
 $\exists w_\nu \in (\Sigma \times \{0, 1\}^k)^* : w_\nu \in L(\mathcal{A}_\psi)$
právě když
 $\mathbb{N}, \nu \models \psi(X_1, \dots, X_k)$
- 3 otestujeme $\varepsilon \in L(\mathcal{A}_\phi)$

Presburgerova aritmetika – $(\mathbb{N}, +)$

Jak rozhodovat následující problém?

Vstup: prvořádomá formule ϕ se sčítáním

Výstup: Platí ϕ na $(\mathbb{N}, +)$?

Presburgerova aritmetika – $(\mathbb{N}, +)$

Jak rozhodovat následující problém?

Vstup: prvořádková formule ϕ se sčítáním

Výstup: Platí ϕ na $(\mathbb{N}, +)$?

Redukcí na wS1S.

- Čísla z P.A. však kódujeme množinami čísel – jejich binárními zápisy.
- Příklady: $6 \mapsto \{2, 3\}$, $11 \mapsto \{1, 2, 4\}$.
- Binární sčítání pak lze vyjádřit v MSO.

Poznámky

- Rozhodování $wS1S$ je non-elementary.
- Pressburgerova aritmetika je v $3EXPTIME$.

Poznámky

- Rozhodování $wS1S$ je non-elementary.
- Pressburgerova aritmetika je v $3EXPTIME$.
- $(\mathbb{N}, .)$ (Skolemova a.) je rozhodnutelná, ale $(\mathbb{N}, ., +1)$, $(\mathbb{N}, ., <)$, $(\mathbb{N}, ., +)$ (Peanova a.) už jsou nerozhodnutelné (a ve skutečnosti ekvivalentní)
viz např. [A. Bes: A Survey of Arithmetical Definability, 2002]

Osnova

- Konečná slova

- Logický popis jazyků
 - Souvislost s automaty
 - Složitost
 - Důsledky

- ② Nekonečná slova

- Rozdíly a podobnosti
 - Souvislost s automaty
 - Důsledky

Konečná slova

$$\Sigma^* = \bigcup_{n \geq 0} \Sigma^n = \bigcup_{n \geq 0} (\{1, \dots, n\} \rightarrow \Sigma)$$

abc, ε , abbbaaababba...

Nekonečná slova

$$\Sigma^\omega = \mathbb{N} \rightarrow \Sigma$$

abababbbabaaababbabaabbababbababbaabbababaabbabbabb...

Konečná slova

Definice regulárních jazyků:

- nedeterministickými KA
- deterministickými KA
- regulárními výrazy
- algebraicky
- logikou MSO

Nekonečná slova

Definice ω -regulárních jazyků:

- nedet. KA více druhů
- deterministické ne vždy
- ω -regulární výrazy
- algeb. popis složitější
- logika MSO funguje

Konečná slova

Definice regulárních jazyků:

- nedeterministickými KA
- deterministickými KA
- regulárními výrazy
- algebraicky
- logikou MSO

Nekonečná slova

Definice ω -regulárních jazyků:

- nedet. KA více druhů
- deterministické ne vždy
- ω -regulární výrazy
- algeb. popis složitější
- logika MSO funguje

Cvičení

- Je každý $L \subseteq \Sigma^*$, $|L| = 1$ regulární?
- Je každý $L \subseteq \Sigma^\omega$, $|L| = 1$ ω -regulární?
- Kolik je jazyků nad abecedou $\Sigma = \{a\}$?
- Kolik je ω -jazyků nad abecedou $\Sigma = \{a\}$?

Automaty

Automaty – jako u konečných slov

nedeterministický KA $\mathcal{A} = (Q, q_0, \longrightarrow, \text{Akc})$

čte vstupní slovo písmeno po písmenu
(akorát to hrozně dlouho trvá :-))

Akceptační podmínky – odlišné

$\text{Inf}(\rho)$ = právě ty stavy, které běh ρ navštíví ∞ -krát

Büchi $\text{Akc} = F \subseteq Q$,
cíl: $\text{Inf}(\rho) \cap F \neq \emptyset$

Muller $\text{Akc} = \mathcal{F} \subseteq 2^Q$,
cíl: $\text{Inf}(\rho) \in \mathcal{F}$

Rabin $\text{Akc} = \mathcal{T} \subseteq 2^Q \times 2^Q$,
cíl: $\exists (N, K) \in \mathcal{T} : N \cap \text{Inf}(\rho) \neq \emptyset \wedge K \cap \text{Inf}(\rho) = \emptyset$

ω -regulární jazyky

Jazyky akceptované

- nedeterministickými Büchiho automaty (BA)
- (ne)deterministickými Mullerovy automaty
- (ne)deterministickými Rabinovy automaty

ω -regulární jazyky

Jazyky akceptované

- nedeterministickými Büchiho automaty (BA)
- (ne)deterministickými Mullerovy automaty
- (ne)deterministickými Rabinovy automaty

Uzavřeny na \cup , \cap . Důkaz stejný jako pro konečná slova.

ω -regulární jazyky

Jazyky akceptované

- nedeterministickými Büchiho automaty (BA)
- (ne)deterministickými Mullerovy automaty
- (ne)deterministickými Rabinovy automaty

Uzavřeny na \cup , \cap . Důkaz stejný jako pro konečná slova.

Uzavřeny na doplněk. Ovšem. . .

Konečná slova: $NKA \rightarrow DKA \rightarrow \overline{DKA}$.

ω -slova: $NBA \not\rightarrow DBA \not\rightarrow \overline{DBA}$.

Jazyky akceptované determ. BA nejsou uzavřeny na doplněk, proto det. BA nepopisují celou třídu ω -reg. jazyků.

Logika

Dvě varianty MSO:

- MSO
- wMSO: „weak” kvantifikace – jen přes konečné množiny

Logika

Dvě varianty MSO:

- MSO
- wMSO: „weak” kvantifikace – jen přes konečné množiny

Hloupá otázka

Která z výše uvedených je „MSO tak, jak ji známe”?

Osnova

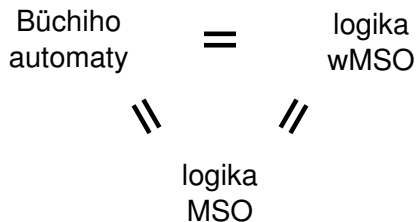
- Konečná slova

- Logický popis jazyků
 - Souvislost s automaty
 - Složitost
 - Důsledky

- ② Nekonečná slova

- Rozdíly a podobnosti
 - Souvislost s automaty
 - Důsledky

Automaty = (w)MSO



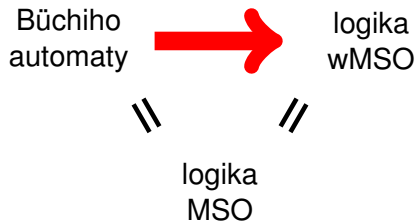
Věta

ω -regulární jazyky = jazyky definovatelné wMSO.

Věta (Büchi)

ω -regulární jazyky = jazyky definovatelné MSO

Automaty = (w)MSO

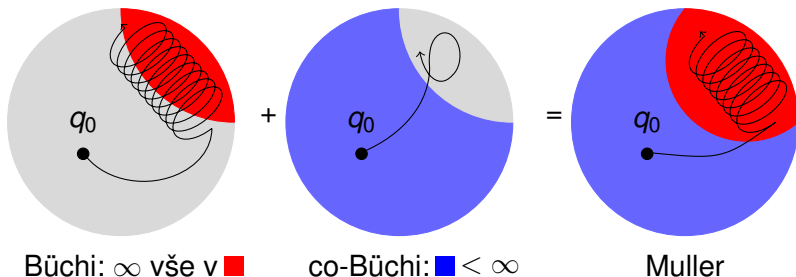


Věta

ω -regulární jazyky = jazyky definovatelné wMSO.

Věta (Büchi)

ω -regulární jazyky = jazyky definovatelné MSO

Büchi \rightarrow Muller \rightarrow wMSO

Pro každý determ. Mullerův \mathcal{A}

existují determ. Büchiho $\mathcal{B}_1, \dots, \mathcal{B}_n, \mathcal{B}'_1, \dots, \mathcal{B}'_n$ tak, že:

$$L(\mathcal{A}) = (L(\mathcal{B}_1) \cap \text{co-}L(\mathcal{B}'_1)) \cup \dots \cup (L(\mathcal{B}_n) \cap \text{co-}L(\mathcal{B}'_n))$$

Büchi \rightarrow Muller \rightarrow wMSO

Věta (McNaughton)

Deterministické Mullerovy automaty a nedeterministické Büchiho automaty akceptují stejnou třídu jazyků.

Důsledek

L je ω -regulární, právě když existují determ. Büchiho $\mathcal{B}_1, \dots, \mathcal{B}_n, \mathcal{B}'_1, \dots, \mathcal{B}'_n$ tak, že:

$$L = (L(\mathcal{B}_1) \cap \text{co-}L(\mathcal{B}'_1)) \cup \dots \cup (L(\mathcal{B}_n) \cap \text{co-}L(\mathcal{B}'_n))$$

Büchi \rightarrow Muller \rightarrow wMSO

Věta (McNaughton)

Deterministické Mullerovy automaty a nedeterministické Büchiho automaty akceptují stejnou třídu jazyků.

Důsledek

L je ω -regulární, právě když existují determ. Büchiho $\mathcal{B}_1, \dots, \mathcal{B}_n, \mathcal{B}'_1, \dots, \mathcal{B}'_n$ tak, že:

$$L = (L(\mathcal{B}_1) \cap \text{co-}L(\mathcal{B}'_1)) \cup \dots \cup (L(\mathcal{B}_n) \cap \text{co-}L(\mathcal{B}'_n))$$

Důsledek

ω -regulární jazyky jsou uzavřeny na doplněk.

detBA \rightarrow wMSO

Protože wMSO je uzavřená na \wedge , \vee a \neg , stačí dokázat:

Pro každý determ. Büchiho $\mathcal{A} = (Q, q_0, \xrightarrow{\cdot}, F)$ existuje sentence wMSO ϕ tak, že $L(\mathcal{A}) = L(\phi)$.

Pro každé $w \in \Sigma^\omega$ platí:

$$w \in L(\mathcal{A}) \\ \iff$$

Existuje běh ρ nad w , tak, že $\rho(n) \in F$ pro ∞ mnoho n .

detBA \rightarrow wMSO

Protože wMSO je uzavřená na \wedge , \vee a \neg , stačí dokázat:

Pro každý determ. Büchiho $\mathcal{A} = (Q, q_0, \xrightarrow{\cdot}, F)$ existuje sentence wMSO ϕ tak, že $L(\mathcal{A}) = L(\phi)$.

Pro každé $w \in \Sigma^\omega$ platí:

$$w \in L(\mathcal{A})$$

$$\iff$$

Existuje běh ρ nad w , tak, že $\rho(n) \in F$ pro ∞ mnoho n .

$$\iff$$

Existuje **jediný** běh ρ nad w , a splňuje $\rho(n) \in F$ pro ∞ mnoho n .

detBA \rightarrow wMSO

Protože wMSO je uzavřená na \wedge , \vee a \neg , stačí dokázat:

Pro každý determ. Büchiho $\mathcal{A} = (Q, q_0, \xrightarrow{\cdot}, F)$ existuje sentence wMSO ϕ tak, že $L(\mathcal{A}) = L(\phi)$.

Pro každé $w \in \Sigma^\omega$ platí:

$$w \in L(\mathcal{A})$$

$$\iff$$

Existuje běh ρ nad w , tak, že $\rho(n) \in F$ pro ∞ mnoho n .

$$\iff$$

Existuje **jediný** běh ρ nad w , a splňuje $\rho(n) \in F$ pro ∞ mnoho n .

$$\iff$$

Jako KA, \mathcal{A} akceptuje $w(1) \cdots w(n)$ pro ∞ mnoho n .

detBA \rightarrow wMSO

- 1 Nad *konečnými* slovy je \mathcal{A} ekvivalentní formulí MSO:
$$\phi = \exists\{X_q \mid q \in Q\} : \phi'.$$

detBA \rightarrow wMSO

- 1 Nad *konečnými* slovy je \mathcal{A} ekvivalentní formulí MSO:
 $\phi = \exists\{X_q \mid q \in Q\} : \phi'$.
- 2 Je-li $w \in \Sigma^\omega$, pak \mathcal{A} akceptuje $w(1) \cdots w(n) \in \Sigma^*$, právě když $w, \nu \models \tau(x)$ pro valuaci $\nu(x) = n$ a formulí

$$\tau(x) := \exists\{X_q \mid q \in Q\} :$$

$$(\phi' \wedge \bigvee_{q \in F} x \in X_q \wedge \bigwedge_{q \in Q} \forall y : (y \in X_q \implies y < x))$$

detBA \rightarrow wMSO

- 1 Nad *konečnými* slovy je \mathcal{A} ekvivalentní formuli MSO:

$$\phi = \exists\{X_q \mid q \in Q\} : \phi'.$$

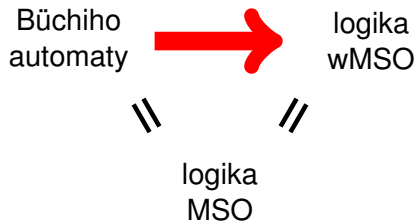
- 2 Je-li $w \in \Sigma^\omega$, pak \mathcal{A} akceptuje $w(1) \cdots w(n) \in \Sigma^*$, právě když $w, \nu \models \tau(x)$ pro valuaci $\nu(x) = n$ a formuli

$$\tau(x) := \exists\{X_q \mid q \in Q\} :$$

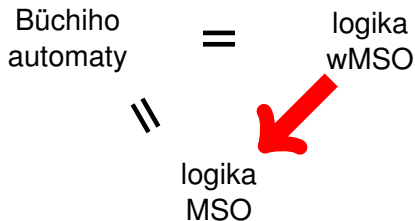
$$(\phi' \wedge \bigvee_{q \in F} x \in X_q \wedge \bigwedge_{q \in Q} \forall y : (y \in X_q \implies y < x))$$

- 3 V MSO i ve wMSO platí: $w \models \forall x : \exists y : (y > x \wedge \tau(y))$, právě když \mathcal{A} akceptuje $w(1) \cdots w(n)$ pro ∞ mnoho n .

Automaty = (w)MSO



Automaty = (w)MSO

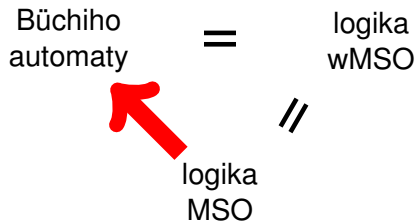


Cvičení

Pro formuli wMSO ϕ najděte formuli MSO ψ tak, že $L(\phi) = L(\psi)$.

Najděte příklad, kdy nelze vzít $\phi = \psi$.

Automaty = (w)MSO



Formule čisté MSO \rightarrow automat

Chceme zkonstruovat pro každou

formuli čisté MSO $\phi(X, \dots)$

Büchiho automat \mathcal{A}

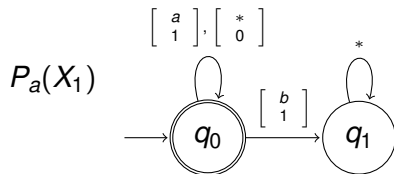
tak, že $L(\phi(X, \dots)) = L(\mathcal{A})$, indukcí ke struktuře ϕ .

Formule čisté MSO \rightarrow automat

Stejně jako u konečných slov.

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F)$.

Bázové kroky:

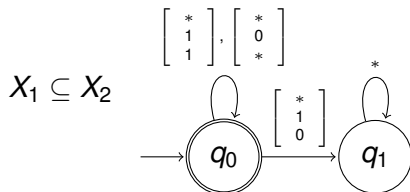


Formule čisté MSO \rightarrow automat

Stejně jako u konečných slov.

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F)$.

Bázové kroky:

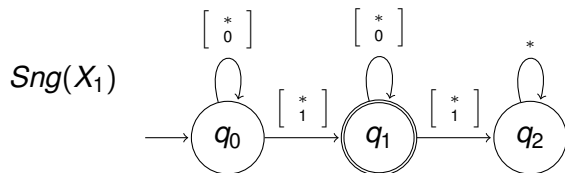


Formule čisté MSO \rightarrow automat

Stejně jako u konečných slov.

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F)$.

Bázové kroky:

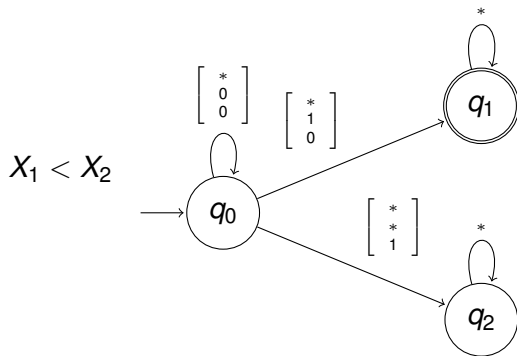


Formule čisté MSO \rightarrow automat

Stejně jako u konečných slov.

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F)$.

Bázové kroky:



Formule čisté MSO \rightarrow automat

Stejně jako u konečných slov.

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\cdot}, F)$.

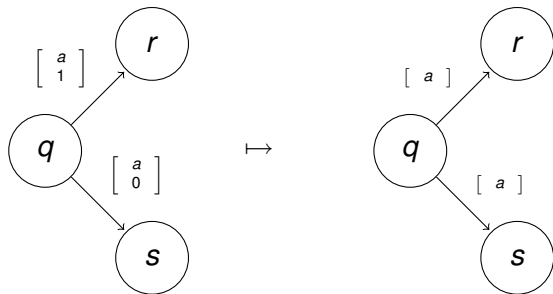
Logické operace \vee, \wedge, \neg přejdou na \cup, \cap a komplement.

Formule čisté MSO \rightarrow automat

Stejně jako u konečných slov.

Formule = $\phi(X_1, \dots, X_k)$, automat = $\mathcal{A} = (Q, q_0, \xrightarrow{\quad}, F)$.

Kvantifikace:

 $\exists X_1 : \phi(X_1)$ 

Osnova

■ Konečná slova

- Logický popis jazyků
- Souvislost s automaty
- Složitost
- Důsledky

② Nekonečná slova

- Rozdíly a podobnosti
- Souvislost s automaty

- Důsledky

$$wS1S = S1S$$

Snadný důsledek toho, že MSO je ekvivalentní wMSO na nekonečných slovech, zejména nad $\Sigma = \{a\}$.

Linear Temporal Logic

Další logika schopná popsat vlastnosti slov:

- $P(a)$ – první písmeno je a
- \vee, \wedge, \neg
- $X\phi$ – ϕ platí od druhého písmena
- $\phi U \psi$ – od nějakého písmena platí ψ , a do té doby ϕ

Linear Temporal Logic

Další logika schopná popsat vlastnosti slov:

- $P(a)$ – první písmeno je a
- \vee, \wedge, \neg
- $X\phi$ – ϕ platí od druhého písmena
- $\phi U \psi$ – od nějakého písmena platí ψ , a do té doby ϕ

Cvičení

Popište

- $true U (\neg (true U \neg P(a)))$ pomocí BA
- $(a^*b)^\omega$ pomocí LTL
- $((aa)^*b)^\omega$ pomocí deterministického BA

FO = LTL

Věta (Kamp; Gabbay, Pnueli, Shelah and Stavi)

Jazyky popsatelné LTL jsou přesně jazyky popsatelné FO.

FO = LTL

Věta (Kamp; Gabbay, Pnueli, Shelah and Stavi)

Jazyky popsateľné LTL jsou přesně jazyky popsateľné FO.

Překvapující, neboť LTL je výrazově mnohem chudší.

Platí se složitostí – splnitelnost je PSPACE-úplná pro LTL, ale non-elementary pro FO.

FO = LTL

Věta (Kamp; Gabbay, Pnueli, Shelah and Stavi)

Jazyky popsatelné LTL jsou přesně jazyky popsatelné FO.

Překvapující, neboť LTL je výrazově mnohem chudší.

Platí se složitostí – splnitelnost je PSPACE-úplná pro LTL, ale non-elementary pro FO.

Cvičení

Nechť \mathcal{L}_{DBA} jsou jazyky akceptované determ. BA, a \mathcal{L}_{LTL} jazyky popsatelné LTL. Dokažte:

- $\mathcal{L}_{LTL} \not\subseteq \mathcal{L}_{DBA}$
- $\mathcal{L}_{DBA} \not\subseteq \mathcal{L}_{LTL}$

FO = star-free

L je ω -regulární, právě když je tvaru

$$L = K_1 \cdot L_1^\omega \cup \dots \cup K_n \cdot L_n^\omega$$

pro K_j, L_j regulární.

FO = star-free

L je ω -regulární, právě když je tvaru

$$L = K_1 \cdot L_1^\omega \cup \dots \cup K_n \cdot L_n^\omega$$

pro K_i, L_i regulární.

Podobně L je definovatelný v FO, právě když je tvaru

$$L = K_1 \cdot L_1^\omega \cup \dots \cup K_n \cdot L_n^\omega$$

pro K_i, L_i star-free.

Další směry

Automaty pro slova už známe. . .

Další směry

Automaty pro slova už známe. . .

. . . přišel čas na automaty pro stromy:

