Shrnutí minulé přednášky (+dva řádky a sloupce navíc)

- Podmnožinová konstrukce DFA z ϵNFA
- Regulární výrazy
- Kleeneho věta
  - Jazyk je přijímaný konečným automatem právě když lze napsat jako regulární výraz,
  - tj. z $\emptyset$ a $\{\mathbf{a}\}$ pro $a \in \Sigma$
  - a konečného počtu aplikací iterace, zřetězení a sjednocení.
- Uzávěrové vlastnosti
  - dnes jen 'regulární' sloupec.

jazyk	regulární (RL)	bezkontextové	deterministické CFL
sjednocení	ANO	ANO	NE
průnik	ANO	NE	NE
$\cap$ s RL	ANO	ANO	ANO
doplněk	ANO	NE	ANO
homomorfizmus	ANO	ANO	NE
inverzní hom.	ANO	ANO	ANO

Substituce jazyků

Definition 4.1 (Substituce jazyků)

Mějme konečnou abecedu Σ. Pro každé $x \in \Sigma$ budiž $\sigma(x)$ jazyk v nějaké abecedě Y_x. Dále položme

$$\sigma(\epsilon) = \{\epsilon\}$$

$$\sigma(u.v) = \sigma(u).\sigma(v)$$

- Zobrazení $\sigma : \Sigma^* \rightarrow P(Y^*)$, kde $Y = \bigcup_{x \in \Sigma} Y_x$ se nazývá **substituce**.
- Pro jazyk L definujeme: $\sigma(L) = \bigcup_{w \in L} \sigma(w)$, podobně sjednocení.
- **nevypouštějící substituce** je substituce, kde žádné $\sigma(x)$ neobstahuje ϵ.

Example 4.1 (substituce)

1) $\Sigma = \{k, p, m, c, t\}$, $L = (kmp)(ckmp)^* t$,
   k slovník křestních jmen, p slovník příjmení, m mezera, c čárka, t tečka.
2) Pokud $\sigma(0) = \{a^i b^j, i, j \geq 0\}, \sigma(1) = \{cd\}$
   tak $\sigma(010) = \{a^i b^j cda^k b^l, i, j, k, l \geq 0\}$.

Homomorfizmus a inverzní homomorfizmus jazyků

Definition 4.2 (homomorfizmus (jazyků), inverzní homomorfizmus)

Homomorfizmus h je speciální případ substituce, kde obraz je vždy jen jednoslovný jazyk (vynecháváme u něj závorky), tj. $(\forall x \in \Sigma)\ h(x) = w_x$.
Pokud $\forall x : w_x \neq \epsilon$, jde o **nevypouštějící homomorfizmus**.
Inverzní homomorfizmus $h^{-1}(L) = \{w | h(w) \in L\}$.

Example 4.2 (homomorfizmus)

- Znaky nahradíme TEXzápisem, $h(\mu) = \backslash mu$ a podobně.
- Homomorfizmus h definujeme: $h(0) = ab$, a $h(1) = \epsilon$. Pak $h(0011) = abab$. Pro $L = \mathbf{10^*1}$ je $h(L) = (ab)^*$.

Theorem 4.1 (uzavřenost na homomorfizmus)

Je–li jazyk L i $\forall x \in \Sigma$ jazyk $\sigma(x), h(x)$ regulární, pak je regulární i $\sigma(L), h(L)$.

Uzavřenost na substituci, homomorfizmus.

Strukturální indukcí 'probublááváním' algebraickým popisem jazyka základních, sjednocení, zřetězení a iterace. Pro sjednocení a zřetězení z definice substituce a uzavřenosti regulárních jazyků na sjednocení a zřetězení.

$$\begin{aligned} \underline{\sigma}(\alpha+\beta) &= \sigma(L(\alpha)) \cup \sigma(L(\beta)) \\ \underline{\sigma}(\alpha\beta) &= \{w | \exists u \in L(\alpha) \exists v \in L(\beta) : \sigma(u)\sigma(v) = w\} \end{aligned}$$

Pro iteraci rozložíme na nekonečné sjednocení, pro každý konkrétní počet iterací σ aplikované na konečné zřetězení.

$$\begin{aligned} \sigma(L(\alpha)^*) &= \sigma(L(\alpha)^0) \cup \sigma(L(\alpha)^1) \cup \ldots \cup \sigma(L(\alpha)^n) \cup \ldots \\ &= \underline{\sigma}(\alpha)^0 \cup \underline{\sigma}(\alpha)^1 \cup \ldots \cup \underline{\sigma}(\alpha)^n \cup \ldots \\ &= L(\underline{\sigma}(\alpha)^*). \end{aligned}$$

□

Inverzní homomorfizmus

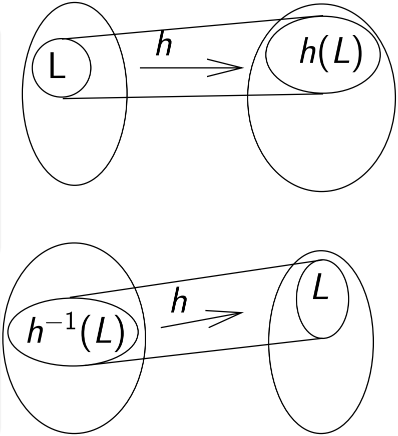
Definition ((4.2) Inverzní homomorfizmus)

Nechť h je homomorfizmus abecedy T do slov nad abecedou Σ. Pak $h^{-1}(L)$ *'h inverze L'* je množina řetězců
$h^{-1}(L) = \{w | w \in T^*;\ h(w) \in L\}$.

Example 4.3

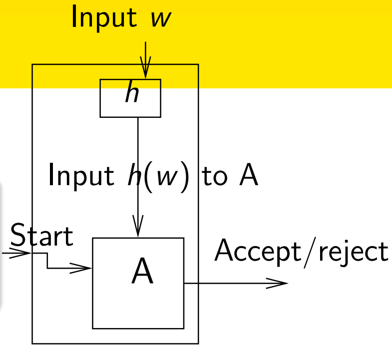
Nechť $L = (\{00\} \cup \{1\})^*$, $h(a) = 01$ a $h(b) = 10$.
Pak $h^{-1}(L) = (\{ba\})^*$.

Důkaz: $h((\{ba\})^*) \in L$ snadno. Ostatní w generují izolované 0 (rozbor případů).



Homomorfizmus aplikovaný dopředně a zpětně.

Inverzní homomorfizmus DFA



Theorem 4.2

Je–li h homomorfizmus abecedy T do abecedy Σ a L je regulární jazyk abecedy Σ, pak $h^{-1}(L)$ je také regulární jazyk.

Proof:

- pro L máme DFA $A = (Q, \Sigma, \delta, q_0, F)$
- $h : T \to \Sigma^*$
- definujeme ϵ-NFA $B = (Q', T, \delta', [q_0, \epsilon], F \times \{\epsilon\})$ kde

$Q' = \{[q, u] | q \in Q, u \in \Sigma^*, \exists(a \in T)\exists(v \in \Sigma^*) h(a) = vu\}$ — u je buffer

$\delta'([q, \epsilon], a) = [q, h(a)]$ — naplňuje buffer

$\delta'([q, bv], \epsilon) = [p, v]$ kde $\delta(q, b) = p$ — čte buffer.

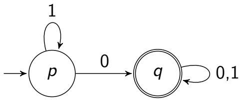
□

Příklad: Navštiv všechny stavy

Example 4.4

Nechť $A = (Q, \Sigma, \delta, q_0, F)$ je DFA. Definujme jazyk $L = \{w \in \Sigma^*;\ \delta^*(q_0, w) \in F$ a pro každý stav $q \in Q$ existuje prefix x_q slova w tak, že $\delta^*(q_0, x_q) = q\}$.
Tento jazyk L je regulání.

- M Označme $M = L(A)$.
- T Definujeme novou abecedu T trojic $\{[paq]; p, q \in Q, a \in \Sigma, \delta(p, a) = q\}$.
- h Definujeme homomorfizmus $(\forall\ p, q, a)\ h([paq]) = a$.
- L_1 Jazyk $L_1 = h^{-1}(M)$ je regulární, protože M je regulární (DFA inverzní homomorfizmus).
  - $h^{-1}(101)$ obsahuje $2^3 = 8$ řetězců, např. $[p1p][q0q][p1p] \in \{[p1p], [q1q]\}\{[p0q], [q0q]\}\{[p1p], [q1q]\}$.
  - Dále zkonstruujeme L z L_1 (další slide).



- L_2 Vynutíme začátek q_0. Definujeme $E_1 = \bigcup_{a\in\Sigma, q\in Q}\{[q_0 a q]\} =$ $E_1 = \{[q_0 a_1 q_0], [q_0 a_2 q_1], \ldots, [q_0 a_m q_n]\}$. Pak $L_2 = L_1 \cap L(E_1.T^*)$.
- L_3 Vynutíme stejné sousedící stavy. Definujeme ne–odpovídající dvojice $E_2 = \bigcup_{q\neq r, p,q,r,s\in Q, a,b\in\Sigma}\{[paq][rbs]\}$. Definujeme $L_3 = L_2 - L(T^*.E_2.T^*)$,
  - Končí v přijímajícím stavu, protože jsme začali z jazyku M přijímaném DFA A.
- L_4 Všechny stavy. $\forall\ q \in Q$ definujeme E_q jako regulární výraz sjednocení všech symbolů T takových, že q není ani na první, ani na poslední pozici. Odečteme $L(E_q^*)$ od L_3. $L_4 = L_3 - \bigcup_{q\in Q}\{E_q^*\}$.
- L Odstraníme stavy, necháme symboly. $L = h(L_4)$. Tedy L je regulární.

Přehled:

- M $= L(A)$
  Inverzní homom.
- L_1 $h^{-1}(M) \subseteq \{[qap]\}^*$
  průnik RJ
- L_2 $+\ q_0$
  rozdíl RJ
- L_3 + sousední stavy rovny
  rozdíl RJ
- L_4 + všechny stavy
  homomorfismus
- L $h([qap]) = a$

Rozhodovací problémy pro regulární jazyky!

Lemma (Test ne–prázdnosti regulárního jazyka)

Lze algoritmicky rozhodnout, zda jazyk přijímaný DFA, NFA, ϵ-NFA je prázdný.

Jazyk je prázdný právě když žádný z koncových stavů není dosažitelný. Dosažitelnost lze testovat $O(|Q|^2)$.

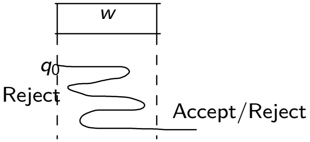
Lemma (Test náležení do regulárního jazyka)

Pro daný řetězec w; $|w| = n$ a regulární jazyk L. Lze algoritmicky rozhodnout, zda je $w \in L$.

- DFA: Spusť automat; pokud $|w| = n$, při dobré reprezentaci a konstatním čase přechodu $O(n)$.
- NFA o s stavech: čas $O(ns^2)$. Každý vstupní symbol aplikujeme na všechny stavy předchozího kroku, kterých je nejvýš s.
- $\epsilon-$NFA - nejdříve určíme ϵ–uzávěr. Pak aplikujeme přechodovou funkci a ϵ–uzávěr na výsledek.

Dvousměrné (dvoucestné) konečné automaty

- Konečný automat provádí následující činnosti:
  - přečte písmeno
  - změní stav vnitřní jednotky
  - posune čtecí hlavu doprava
- Čtecí hlava se nesmí vracet.



Definition 5.1 (Deterministické Dvousměrné konečné automaty)

Deterministickým dvousměrným konečným automatem nazýváme pětici $A = (Q, \Sigma, \delta, q_0, F)$, kde

1. Q je konečná množina stavů,
2. Σ je konečná množina vstupních symbolů
3. přechodové funkce δ je zobrazení $Q \times \Sigma \rightarrow Q \times \{-1, 1\}$ rozšířené o pohyb hlavy
4. $q_0 \in Q$ počáteční stav
5. množina přijímajících stavů $F \subseteq Q$.

Pozn.: Je deterministický, nedeterministický $\delta_N : Q \times \Sigma \rightarrow \mathcal{P}(Q \times \{-1, 1\})$.
Pozn.2: Nulový pohyb hlavy lze, jen trochu zkomplikuje důkaz dále.

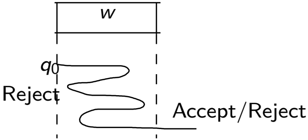
- Reprezentujeme opět stavovým diagramem, tabulkou

Výpočet dvousměrného automatu

Definition 5.2 (Výpočet dvousměrného automatu)

Slovo w je **přijato dvousměrným konečným automatem**, pokud:

- výpočet začal na prvním písmenu slova w vlevo v počátečním stavu
- čtecí hlava poprvé opustila slovo w vpravo v některém přijímajícím stavu
- mimo čtené slovo není výpočet definován (výpočet zde končí a slovo není přijato).



- Ke slovům si můžeme přidat speciální koncové znaky $\# \notin \Sigma$
- funkce $\partial_\#$ odstraní $\#$ zleva, $\partial_\#^R$ zprava.

Lemma

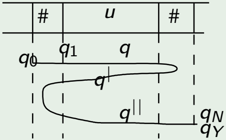
Je–li $L(A) = \{\#w\# | w \in L \subseteq \Sigma^*\}$ regulární, potom je regulární i $L = \partial_\# \partial_\#^R (L(A) \cap \#\Sigma^*\#)$.

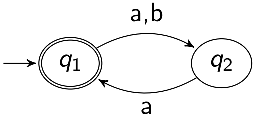
Příklad dvousměrného automatu

Example 5.1 (Příklad dvousměrného automatu)

Nechť $A = (Q, \Sigma, \delta, q_1, F)$. Dvousměrný konečný automat $B = (Q \cup Q^{|} \cup Q^{||} \cup \{q_0, q_N, q_Y\}, \Sigma \cup \{\#\}, \delta^{|}, q_0, \{q_Y\})$ přijímající jazyk $L(B) = \{\#u\# | uu \in L(A)\}$ (toto NENÍ levý ani pravý kvocient!) definujeme následovně:

$\delta^{|}$	$x \in \Sigma$	$\#$	poznámka
q_0	$q_N, -1$	$q_1, +1$	q_1 je počátek A
q	$p, +1$	$q^{|}, -1$	$p = \delta(q, x)$
$q^{|}$	$q^{|}, -1$	$q^{||}, +1$	
$q^{||}$	$p^{||}, +1$	$q_Y, +1$	$q \in F, p = \delta(q, x)$
$q^{||}$	$p^{||}, +1$	$q_N, +1$	$q \notin F, p = \delta(q, x)$
q_N	$q_N, +1$	$q_N, +1$	
q_Y	$q_N, +1$	$q_N, +1$	





Dvousměrné a jednosměrné konečné automaty

Theorem 5.1

Jazyky přijímané dvousměrnými konečnými automaty jsou právě regulární jazyky.

Proof: konečný automat → dvousměrný automat

- Konečný automat předeveme na dvousměrný přidáním posunu hlavy vpravo
- $A = (Q, \Sigma, \delta, q_0, F) \rightarrow 2A = (Q, \Sigma, \delta^|, q_0, F)$, kde $\delta^|(q, x) = (\delta(q, x), +1)$. $\square$

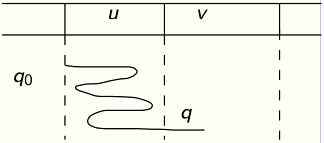
- Možnost pohybovat čtecí hlavou po pásce nezvětšila sílu konečného automatu (dokud na pásku nic nepíšeme!).
- Pro důkaz potřebujeme přípravu.

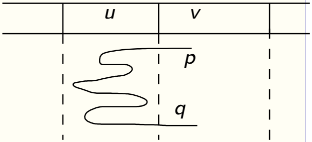
Funkce f_u popisující výpočet 2DFA nad slovem u

Algorithm: Funkce f_u popisující výpočet 2DFA nad slovem u

Definujeme funkci $f_u : Q \cup \{q_0^{|}\} \to Q \cup \{0\}$

- $f_u(q_0^{|})$ popisuje v jakém stavu poprvé odejdeme vpravo, pokud začneme výpočet vlevo v počátečním stavu q_0,
- symbol 0 značí, že daná situace nenastane (odejdeme vlevo nebo cyklus),
- $f_u(p); p \in Q$ v jakém stavu opět odejdeme vpravo, pokud začneme výpočet vpravo v p





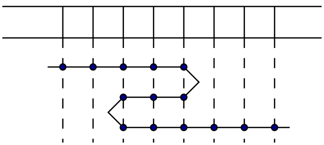
- Definujeme ekvivalenci slov následovně: $u \sim w \Leftrightarrow_{def} f_u = f_w$,
  - tj. slova jsou ekvivalentní pokud mají stejné 'výpočtové' funkce.

Regulárnost 2DFA

Ekvivalence $\sim$ je ekvivalence, má konečný index, je to pravá kongruence, jazyk 2DFA odpovídá sjednocení tříd $f_w(q_0^{|}) \in F$.

Konstruktivní důkaz věty o 2DFA

- Potřebujeme převést návraty na lineární výpočet.
- Zajímají nás jen přijímající výpočty
- Díváme se na řezy mezi symboly (v jakém stavu přechází na další políčko)



Pozorování:

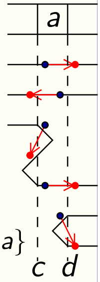
- stavy se v přechodu řezu střídají (doprava, doleva)
- první stav jde doprava, poslední také doprava
- v deterministických přijímajících výpočtech nejsou cykly
- první a poslední řez obsahují jediný stav.

Algorithm: 2DFA $\rightarrow$ NFA

- Najdeme všechny možné řezy – posloupnosti stavů (je jich konečně mnoho).
- Mezi řezy definujeme nedeterministické přechody podle čteného symbolu.
- Rekonstruujeme výpočet skládáním řezů jako puzzle.

Algorithm: Formální převod 2DFA na NFA

Nechť $A = (Q, \Sigma, \delta, q_0, F)$ je dvousměrný (deterministický) konečný automat. Definujeme ekvivalentní nedeterministický automat $B = (Q^{|}, \Sigma, \delta^{|}, (q_0), F^{|})$ kde:



- $Q^{|}$ jsou všechny korektní přechodové posloupnosti
  - posloupnosti stavů $(q^1, \ldots, q^k)$; $q^i \in Q$
  - délka posloupnosti je lichá ($k = 2m + 1$)
  - žádný stav se neopakuje na liché ani na sudé pozici $(\forall i \neq j)\ (q^{2i} \neq q^{2j})$ & $(\forall i \neq j)(q^{2i+1} \neq q^{2j+1})$
- $F^{|} = \{(q) | q \in F\}$ posloupnosti délky 1
- $\delta^{|}(c, a) = \{d | d \in Q^{|} \& c \xrightarrow{a} d$ je lokálně konzistentní přechod pro $a\}$
  - existuje bijekce: $h : c_{odd} \cup d_{even} \rightarrow c_{even} \cup d_{odd}$, tak, že:
  - zachovává uspořádání
  - pro $h(q) \in c_{even}$ je $(h(q), -1) = \delta(q, a)$
  - pro $h(q) \in d_{odd}$ je $(h(q), +1) = \delta(q, a)$.

$L(A) = L(B)$

Trajektorie 2DFA A odpovídá řezům v FA B, odtud $L(A) = L(B)$.

Příklad převodu 2DFA na NKA

Možné řezy a jejich přechody

- Mějme následující dvousměrný konečný automat:

	a	b
$\rightarrow p$	p,+1	q,+1
$*q$	q,+1	r,-1
r	p,+1	r,-1

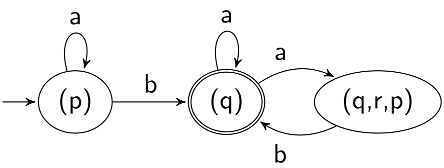
- Doleva jedině r – všechny sudé pozice r, tj. jediná sudá
- možné řezy: $(p), (q), (p, r, q), (q, r, p)$.

	a	b
$\rightarrow (p)$	(p)	(q)
$*(q)$	(q),(q,r,p)	
(p,r,q)		
(q,r,p)		(q)

Ukázka (zacykleného, nepřijímajícího) výpočtu:

a	a	b	a	a	b	a	a	b	b
p	p	p	q	q	q				
				r					
					p	q	q	q	
							r		
								p	q
							r	r	

Výsledný NFA:



Automaty s výstupem (motivace)

- Dosud jediná zpráva z automatu: 'Jsme v přijímajícím stavu'.
- Můžeme z FA získat více informací? Můžeme zaznamenat trasu výpočtu?

Moore: indikace stavů (všech, nejen koncových)

- v každé chvíli víme, kde se automat nachází
- Příklad: různé (regulární) čítače

Mealy: indikace přechodů

- po přečtení každého symbolu víme, co automat dělal
- Příklad: regulární překlad slov

Automat už není tak docela černá skříňka.

Mooreův stroj

Definition 5.3 (Mooreův stroj)

Mooreovým (sekvenčním) strojem nazýváme šestici $A = (Q, \Sigma, Y, \delta, \mu, q_0)$ resp. pětici $A = (Q, \Sigma, Y, \delta, \mu)$, kde

Q je konečná neprázdná množina stavů

Σ je konečná neprázdná množina symbolů (vstupní abeceda)

Y je konečná neprázdná množina symbolů (**výstupní abeceda**)

δ je zobrazení $Q \times \Sigma \rightarrow Q$ (přechodová funkce)

μ je zobrazení $Q \rightarrow Y$ (**značkovací funkce**)

$q_0 \in Q$ (počáteční stav)

- Někdy nás nezajímá počáteční stav, ale jen práce automatu
- značkovací funkce umožňuje suplovat roli koncových stavů
  - $F \subseteq Q$ nahradíme značkovací funkcí $\mu : Q \rightarrow \{0, 1\}$ takto:

    $\mu(q) = 0$ pokud $q \notin F$,
    $\mu(q) = 1$ pokud $q \in F$.

Příklad Mooreova stroje

Example 5.2 (Mooreův stroj pro tenis)

Mooreův stroj pro počítání tenisového skóre.

- Vstupní abeceda: ID hráče, který uhrál bod
- Výstupní abeceda & stavy: skóre (tj. $Q = Y$ a $\mu(q) = q$)

Stav/výstup	A	B
00:00	15:00	00:15
15:00	30:00	15:15
15:15	30:15	15:30
00:15	15:15	00:30
30:00	40:00	30:15
30:15	40:15	30:30
30:30	40:30	30:40
15:30	30:30	15:40
00:30	15:30	00:40
40:00	A	40:15
40:15	A	40:30
40:30	A	shoda
30:40	shoda	B
15:40	30:40	B
00:40	15:00	B
shoda	A:40	40:B
A:40	A	shoda
40:B	shoda	B
A	15:00	00:15
B	15:00	00:15

Mealyho stroj

Definition 5.4 (Mealyho stroj)

Mealyho (sekvenčním) strojem nazýváme šestici $A = (Q, \Sigma, Y, \delta, \lambda_M, q_0)$ resp. pětici $A = (Q, \Sigma, Y, \delta, \lambda_M)$, kde

Q je konečná neprázdná množina stavů

Σ je konečná neprázdná množina symbolů (vstupní abeceda)

Y je konečná neprázdná množina symbolů (výstupní abeceda)

δ je zobrazení $Q \times \Sigma \to Q$ (přechodová funkce)

λ_M je zobrazení $Q \times \Sigma \to Y$ (**výstupní funkce**)

$q_0 \in Q$ (počáteční stav)

- Výstup je určen stavem a vstupním symbolem
  - Mealyho stroj je obecnějším prostředkem než stroj Mooreův
  - Značkovací funkci $\mu : Q \to Y$ lze nahradit výstupní funkcí $\lambda_M : Q \times \Sigma \to Y$ například takto:

$$\forall x \in \Sigma \ \lambda_M(q, x) = \mu(q)$$

nebo $\forall x \in \Sigma \ \lambda_M(q, x) = \mu(\delta(q, x))$

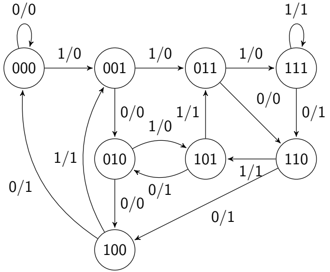
Příklad Mealyho stroje

Example 5.3 (Mealyho stroj)

Automat, který dělí vstupní slovo v bináním tvaru číslem 8 (celočíselně).

- Posun o tři bity doprava
- potřebujeme si pamatovat poslední trojici bitů
- vlastně tříbitová dynamická paměť

Stav\symbol	0	1
000	000/0	001/0
001	010/0	011/0
010	100/0	101/0
011	110/0	111/0
100	000/1	001/1
101	010/1	011/1
110	100/1	101/1
111	110/1	111/1

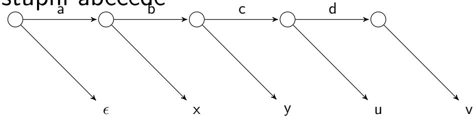


- I když nevíme, kde automat startuje, po třech symbolech začne počítat správně.

Výstup sekvenčních strojů

slovo ve vstupní abecedě $\rightarrow$ slovo ve výstupní abecedě

Mooreův stroj



značkovací funkce $\mu : Q \rightarrow Y$

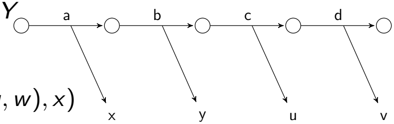
$\mu^* : Q \times \Sigma^* \rightarrow Y^*$

$\mu^*(q, \epsilon) = \epsilon$ (někdy $\mu^*(q, \epsilon) = q$)

$\mu^*(q, wx) = \mu^*(q, w).\mu(\delta^*(q, wx))$

Příklad: μ^*(00:00,AABA)=(00:00 .) 15:00 . 30:00 . 30:15 . 40:15

Mealyho stroj



výstupní funkce $\lambda_M : Q \times \Sigma \rightarrow Y$

$\lambda_M^* : Q \times \Sigma^* \rightarrow Y^*$

$\lambda_M^*(q, \epsilon) = \epsilon$

$\lambda_M^*(q, wx) = \lambda_M^*(q, w).\lambda_M(\delta^*(q, w), x)$

Příklad: λ_M^*(000,1101010)=0001101

Lemma (Převod Mooreova stroje na Mealyho)

Pro každý Moorův stroj existuje Mealyho stroj převádějící každé vstupní slovo na stejné výstupní slovo.

Proof.

- Nechť $A = (Q, \Sigma, Y, \delta, \mu, q_0)$ je Mooreův stroj.
- Definujeme Mealyho stroj $B = (Q, \Sigma, Y, \delta, \lambda_M, q_0)$, kde $\lambda_M(q, x) = \mu(\delta(q, x))$

  tj. λ_M vrací značku stavu, do kterého přejdeme.

$\square$

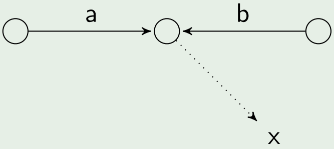
Example 5.4

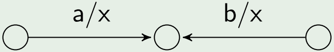
Mooreův stroj

stav	0	1	výstup
a	a	b	0
b	b	c	1
c	c	a	2

Mealyho stroj se stejným výstupem

stav	0	1
a	a/0	b/1
b	b/1	c/2
c	c/2	a/0





Převod Mealyho stroje na Mooreův

Lemma (Převod Mealyho stroje na Mooreův)

Pro každý Mealyho stroj existuje Mooreův stroj převádějící každé vstupní slovo na stejné výstupní slovo.

Nechť $A = (Q, \Sigma, Y, \delta, \lambda_M, q_0)$ je Mealyho stroj.
Sestrojme Mooreův stroj B tak, aby $\forall q, w\ \lambda_M^*(q, w) = \mu^*(q, w)$.

! Rozdělíme stav na více stavů, podle počtu výstupních symbolů.

$B = (Q \times (Y \cup \{\_\}), \Sigma, Y, \delta^{|}, \mu, (q_0, \_))$, kde

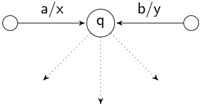
$\delta^{|}((q, y), x) = (\delta(q, x), \lambda_M(q, x))$ a

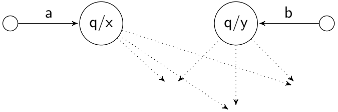
$\mu((q, y)) = y$

Příklad:

stav	0	1
a	a/0	b/0
b	a/1	b/1

stav	0	1	výstup
(a,0)	(a,0)	(b,0)	0
(a,1)	(a,0)	(b,0)	1
(b,0)	(a,1)	(b,1)	0
(b,1)	(a,1)	(b,1)	1





Konečné automaty – shrnutí

Konečný automat

- redukovaný deterministický automat (lze definovat i jednoznačný)
- nedeterminismus ϵ–NFA, 2^n, (dvousměrný FA n^n)

Regulární výrazy

Automaty a jazyky

- regulární jazyky
- uzavřenost na množinové operace
- uzavřenost na řetězcové operace
- uzavřenost na substituci, homomorfizmus a inverzní homomorfizmus,
- automaty výše i regulární výrazy popisují stejnou třídu jazyků.

Charakteristika regulárních jazyků

- Mihyll–Nerodova věta (kongruence)
- Kleeneova věta (elementární jazyky a operace)
- Iterační (pumping) lemma (iterace podslov, jen nutná podmínka).

(Automaty s výstupem)

- (Mooreův stroj)
- (Mealyho stroj)

Přehled kapitol

1. Úvod, Iterační lemma pro reg. jazyky
2. Redukovaný DFA a ekvivalence automatů, stavů
3. Nedeterministické ϵ–NFA, Operace zachovávající regularitu
4. Regulární výrazy, Kleeneova věta, Substituce, Homomorfizmus
5. Dvousměrné FA, Mealy a Moore stroje
6. Gramatiky, Chomského hierarchie, víceznačnost
7. Chomského NF, Pumping Lemma pro CFL
8. CYK – náležení do CFL
9. Zásobníkové automaty, Deterministické PDA
10. Uzávěrové vlastnosti, Dykovy jazyky
11. Turingův stroj, rozšíření
12. Lineárně omezené automaty, Univerzální TM, Diagonální jazyk
13. Nerozhodnutelné problémy, Postův korespondenční p.
14. Časová složitost