AUTOMATY A GRAMATIKY

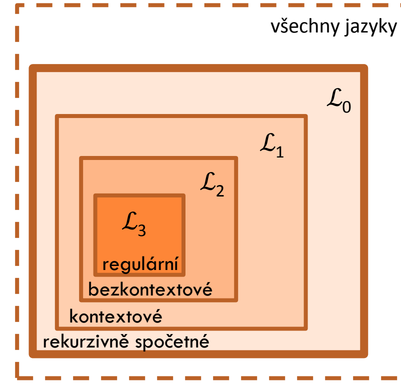
13

Pavel Surynek

Univerzita Karlova v Praze

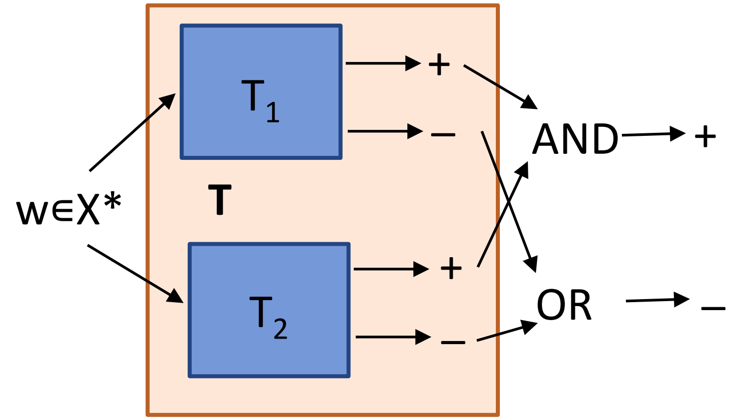
Matematicko-fyzikální fakulta
Katedra teoretické informatiky a matematické logiky

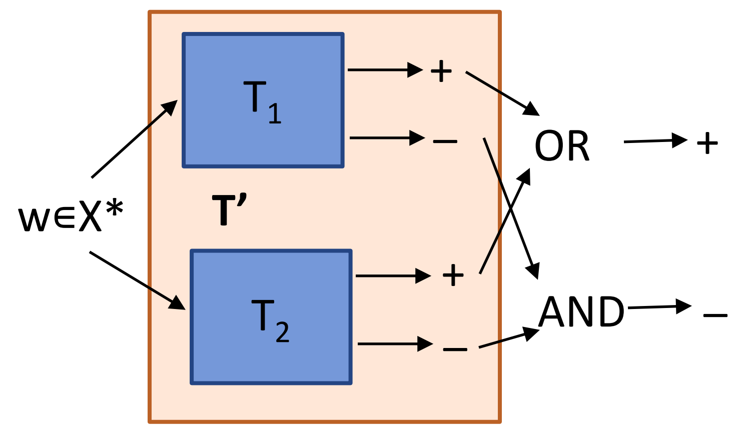
Uzávěrové vlastnosti
Nerozhodnutelné problémy
Ricova věta
Postův korespondenční problém
Nerozhodnutelnost u gramatik



Uzávěrové vlastnosti (1)

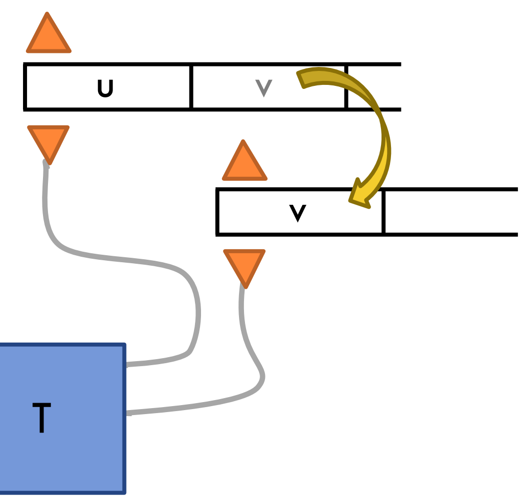
- **rekurzivní** a **rekurzivně spočetné** jazyky jsou uzavřené na **konečný průnik** a **konečné sjednocení**
  - jsou-li L_1 a L_2 rekurzivní resp. rekurzivně spočetné jazyky, pak $L_1 \cap L_2$ i $L_1 \cup L_2$ jsou rekurzivní resp. rekurzivně spočetné jazyky
  - mějme TS T_1 a T_2 1-páskové, že $L(T_1)=L_1$ a $L(T_2)=L_2$
    - zkonstruujeme dvoupáskové TS T a T', že $L(T) = L_1 \cap L_2$, $L(T') = L_1 \cup L_2$
      - na druhou pásku zkopíruje vstup
      - na první pásce simuluje T_1 } paralelně
      - na druhé pásce simuluje T_2 }
    - T přijme, když obě simulace přijmou
    - T' přijme, když aspoň jedna simulace přijme
      - v **rekurzivním** případě vždy oba simulované TS zastaví
  - v **rekurzivně spočetném** může jeden či oba simulované TS běžet navždy





Uzávěrové vlastnosti (2)

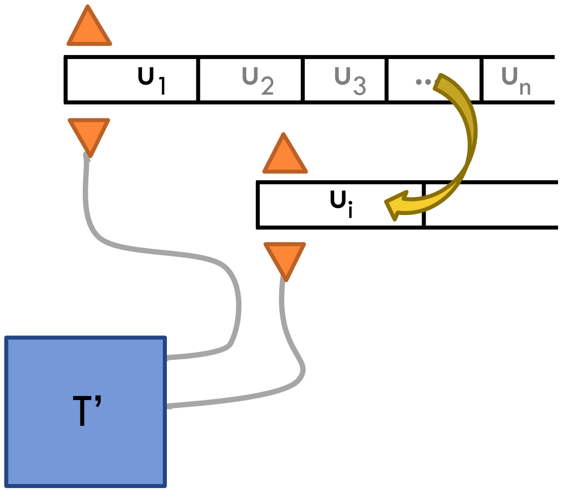
- **rekurzivní** a **rekurzivně spočetné** jazyky jsou uzavřené na **konkatenaci**
  - jsou-li L_1 a L_2 rekurzivní resp. rekurzivně spočetné jazyky, pak $L_1.L_2$ je rekurzivní resp. rekurzivně spočetný jazyk
  - mějme TS T_1 a T_2 1-páskové, že $L(T_1)=L_1$ a $L(T_2)=L_2$
    - rekurzivně spočetný případ
      - zkonstruujeme nedeterministický 2-páskový TS T, že $L(T) = L_1.L_2$
      - T nedeterministicky uhádne rozdělení vstupního slova w = u.v
      - přesune v na druhou pásku
        - na první pásce (tedy nad u) simuluje T_1
        - na druhé pásce (tedy nad v) simuluje T_2
      - když oba simulované TS přijmou, přijme i T



    - rekurzivní případ
      - nedeterminismus nelze použít, protože převod na deterministický případ nezachovává zastavení při nepřijímání
      - zkonstruujeme deterministický (více-páskový) TS T', že $L(T') = L_1.L_2$
      - T' otestuje všechna rozdělení vstupního slova w = u.v
      - simulace stejně jako pro rekurzivně spočetný případ

Uzávěrové vlastnosti (3)

- **rekurzivní** a **rekurzivně spočetné** jazyky jsou uzavřené na **iteraci**
  - je-li L rekurzivní resp. rekurzivně spočetný jazyk, pak L* je rekurzivní resp. rekurzivně spočetný jazyk
  - mějme TS T 1-páskový, že L(T)=L
    - rekurzivně spočetný případ
      - zkonstruujeme nedeterministický 2-páskový TS T, že L(T') = L*
      - T' nedeterministicky uhádne počet dělení a samo dělení $w = u_1.u_2...u_n$
      - na druhé pásce postupně simuluje práci T nad $u_1,u_2,...,u_n$
      - T' přijme, pokud všechny simulace přijmou
    - rekurzivní případ
      - opět je nutno nahradit nedeterministické uhádnutí dělení vstupního slova
      - zkonstruujeme deterministický (více-páskový) TS T'', že L(T'') = L*
      - T'' otestuje všechna možná dělení $w = u_1.u_2...u_n$
      - simulace stejně jako v rekurzivně spočetném případě



Uzávěrové vlastnosti (4)

- **rekurzivní** a **rekurzivně spočetné** jazyky jsou uzavřené na **zrcadlový obraz**
  - L rekurzivní resp. rekurzivně spočetný jazyk, pak L^R je rekurzivní resp. rekurzivně spočetný jazyk
  - mějme TS T 1-páskový, že L(T)=L
    - zkonstruujeme TS T', že L(T')=L
      - T' bude skoro stejný jako T
      - ale nejprve zrcadlově otočí vstupní slovo
    - konstrukce funguje pro rekurzivně spočetný i rekurzivní případ
- **rekurzivní** a **rekurzivně spočetné** jsou uzavřené na **inverzní homomorfismus**
  - zkonstruujeme TS T'', že $L(T'')=h^{-1}(L)$, kde $h: Y \rightarrow X^*$ je homomorfismus
    - T'' na vstup $w \in Y^*$ aplikuje h
    - na h(w) simuluje T
      - když simulovaný T přijme, T'' také přijme
    - konstrukce opět funguje pro rekurzivně spočetný i rekurzivní případ
- **rekurzivně spočetné** jazyky jsou uzavřené na **homomorfismus**
  - zkonstruujeme nedeterministický TS T''', že L(T''')=h(L)
    - pro vstup $w \in Y^*$ T''' nedeterministicky uhádne $u \in X^*$, že $h(u) = w$; T''' přijme w, jestliže T přijme x

Problémy formálně

- **rozhodovací problém** (rozhodovací úloha)
  - intuitivně
    - otázka typu ano/ne o nekonečně mnoha (spočetně mnoha) <u>instancích</u>

Př.: Otázka: má daný neorientovaný graf G Hamiltonovskou kružnici?

Instance: <u>všechny neorientované grafy</u>.

  - formálně
    - problém je jazyk L
      - slovo w kóduje instanci
      - w∈L, jestliže je odpověď na instanci kódovanou w **„ANO“**
    - přirozeně máme pojmy (algoritmicky) **rozhodnutelný** a (algoritmicky) **nerozhodnutelný** problém
      - odpovídá rekurzivnímu resp. nerekurzivnímu jazyku

Př.: L_h = { w | w kóduje neorientovaný graf s Hamiltonovskou kružnicí }
je **rozhodnutelný**

L_u = { u111v | TS s kódem u přijímá v }
je **nerozhodnutelný**

Nerozhodnutelnost a Ricova věta

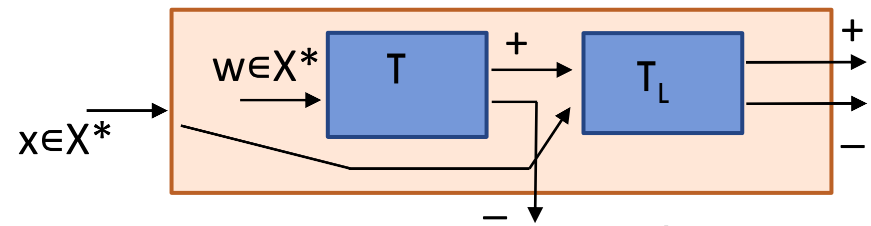
- existují i jiné (praktické) nerozhodnutelné problémy než L_u
  - nechť P je nějaká vlastnost jazyka L (L je nekonečný, L je regulární, bezkontextový, ...)
  - L_P = { kód(T) | L(T) má vlastnost P }
- **Ricova věta** (Rice's theorem)
  - L_P je rozhodnutelný pouze pro dvě triviální vlastnosti P
    - a sice pro vlastnost splňenou všemi rekurzivně spočetnými jazyky (*always true*) a pro vlastnost, kterou nesplňuje žádný rekurzivně spočetný jazyk (*always false*)
  - jinak je L_P nerozhodnutelný
- **redukce** jazyka L na jazyk K, kde $L,K \subseteq X^*$
  - je TS, který vždy zastaví a libovolné $w \in X^*$ převede na $v \in X^*$ tak, že $w \in L \Leftrightarrow v \in K$
    - výstup je realizován na výstupní pásce
      - TS s výstupem ... *transducer*
- když najdeme redukci jazyka L na rozhodnutelný jazyk K, pak je L rozhodnutelný
  - TS rozhodující K a TS provádějící převod dohromady ukazují rozhodnutelnost L
  - **obměna:** když L není rozhodnutelný, nemůže být ani K rozhodnutelný

Ricova věta a převody (1)

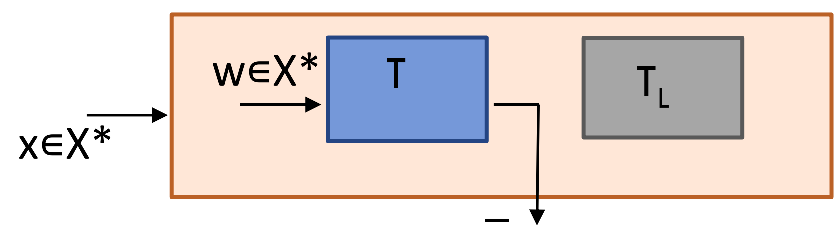
- pro **netriviální** vlastnost P ($\neq$ always true, always false)
  - najdeme redukci L_u na L_P
    - pak, jelikož je L_u není rozhodnutelný, nemůže být ani L_P
  - předpoklady
    - jazyk Ø nemá vlastnost P
      - pokud tomu tak není, vezmeme místo P doplněk P (P^c)
        - kdyby byl L_P rozhodnutelný, je i L_{Pc} rozhodnutelný
    - nechť L je libovolný rekurzivně spočetný jazyk, který má vlastnost P; T_L je TS, že $L(T_L) = L$
- sestrojíme TS, který pro vstup kód(T)111w vytvoří kód(T'), kde L(T') bude mít vlastnost P $\Leftrightarrow$ T přijímá w
  - T' vznikne přeprogramováním T
    - T' bude mít dvě virtuální pásky
    - na 2. pásku zapíše w a simuluje T na 2. pásce
      - když T přijme w, T' simuluje T_L na svém vstupu x z 1.pásky
        - když T_L přijme x, T' přijme
    - obě virtuální pásky budou simulovány v jedné skutečné pásce
      - kódujeme jednopáskové TS

Ricova věta a převody (2)

- když T **přijímá** w, pak
  - zkonstruovaný TS T' přijímá L, protože v tomto případě T' simuluje T_L, pro který $L(T_L)=L$
  - jelikož L má vlastnost P, má rovněž L(T') vlastnost P
    - $kód(T')\in L_P$



- když T **nepřijímá** w, pak
  - zkonstruovaný TS T' nepřijímá žádné slovo, tedy $L(T')=\emptyset$, o němž jsme předpokládali, že vlastnost P nemá
    - $kód(T')\notin L_P$



- **celkem**
  - T přijímá w ⇔ zkonstruovaný TS T' má vlastnost P
  - konstruování T' z T a w (pomocí TS) je redukce L_u na L_P

Důsledky Ricovy věty

- máme nepřeberné množství nerozhodnutelných jazyků
  - pro každdou netriviální vlastnost P, je L_P nerozhodnutelný
    - $L_P = \{ \text{kód}(T) \mid L(T) \text{ je regulární} \}$
      - P je regularita
    - $L_P = \{ \text{kód}(T) \mid L(T) \text{ je bezkontextový} \}$
      - P je bezkontextovost
    - $L_P = \{ \text{kód}(T) \mid L(T) \text{ obsahuje palindrom} \}$
      - P je palindromovitost
    - $L_P = \{ \text{kód}(T) \mid L(T) = \emptyset \}$
      - P je prázdnost
    - $L_P = \{ \text{kód}(T) \mid L(T) = X^* \}$
    - $L_P = \{ \text{kód}(T) \mid |L(T)|>3 \}$
    - atd...
  - kód(T) lze nahlížet jako program
    - o tom, co dělají programy, nelze programem téměř nic rozhodnout

Postův korespondenční problém (1)

- instance **Postova korespondenčního problému** (*PKP*) je konečná posloupnost dvojic neprázdných slov nad nějakou abecedou X
  - $(w_1,x_1), (w_2,x_2), \ldots, (w_n,x_n)$ $n \in \mathbb{N}$, $w_i, x_i \in X^*$ pro $i=1,2,\ldots,n$
  - instance PKP má řešení, jestliže existují indexy $i_1,i_2,\ldots,i_k$ s $k \in \mathbb{N}$ kde $i_j \in \{1,2,\ldots,n\}$ pro $j = 1,2,\ldots,k$, že
    - $w_{i1}.w_{i2}\ldots w_{ik} = x_{i1}.x_{i2}\ldots x_{ik}$
- **modifikovaný PKP** (*mPKP*)
  - skoro stejný jako PKP, ale řešení musí začít první dvojicí, tj.
    - $w_1 w_{i1}.w_{i2}\ldots w_{ik} = x_1 x_{i1}.x_{i2}\ldots x_{ik}$

Př.: **a)** instance *PKP*
(0,01), (100,001)
nemá řešení

b) instance *mPKP*
(0,01), (100,001), (110,10)
má řešení

c) instance *mPKP*
(110,10), (0,01), (100,001)
nemá řešení

Postův korespondenční problém (2)

- **PKP pomocí mPKP**
  - vyzkoušíme všechny možné dvojice jako počáteční v mPKP
    - pokud aspoň jeden vytvořený mPKP má řešení, má řešení PKP
- **mPKP pomocí PKP**
  - použijeme nové symboly # a $
    - **za** každý symbol **prvního** z každé dvojice slov přidat #
    - **před** každý symbol **druhého** z každé dvojice slov přidat #
    - přidat dvojici ($,#$)
      - slouží k zakončení
    - přidat další kopii první dvojice slov, kde bude přidán # na **začátek prvního** z dvojice slov
      - vynuceno použití na začátku výsledné posloupnosti

Př.: instance *mPKP*

(110,10)
(0,01),
(100,001)

ekvivalentní instance *PKP*

(1#1#0#, #1#0)
(0#, #0#1),
(1#0#0#, #0#0#1)
($, #$)
(#1#1#0#, #1#0)

Nerozhodnutelnost PKP (1)

- L_{PKP} = { kód(I) | I je instance PKP, která má řešení }
- L_{mPKP} = { kód(I) | I je instance mPKP, která má řešení }
- ukážeme, že L_{mPKP} je nerozhodnutelný
  - tím pádem ani L_{PKP} nebude rozhodnutelný
    - popsali jsme redukční algoritmus pro převod L_{mPKP} na L_{PKP}
      - návod na vytvoření TS, který z kódu instance mPKP vytvoří kód instance PKP při zachování řešitelnosti
  - redukce L_u na L_{mPKP}
    - pro daný TS $T = (Q, \{0,1\}, \delta, q_0, b, F)$ a w (zadané jako kód(T)111w) vytvoříme instanci I mPKP, že I má řešení $\Leftrightarrow (\lambda, q_0, w) \vdash_T^* (\lambda, f, b)$, kde $f \in F$
    - existuje posloupnost konfigurací $K_1, K_2, \ldots, K_m$ s $m \in \mathbb{N}_0$, že $(\lambda, q_0, w) \vdash_T K_1 \vdash_T K_2 \vdash_T \ldots \vdash_T K_m \vdash_T (\lambda, f, b)$
    - posloupnost konfigurací sestavíme jako výsledné slovo v mPKP
      - nový symbol @ bude oddělovat konfigurace

Nerozhodnutelnost PKP (2)

- konstrukce mPKP
  - 1. dvojice
    - $(@,@q_0w@)$
  - další dvojice
    - (x,x) pro $x \in X$
      - pro kopírování
    - $(@,@)$
      - pro zakončení kroku výpočtu
    - pro každý $q \in Q$ a $x \in (X-\{b\})$
      - (qx, yp) kdykoli $\delta(q, x) = (p, y, +1)$
      - (qx, py) kdykoli $\delta(q, x) = (p, y, 0)$
      - (zqx, pzy) kdykoli $\delta(q, x) = (p, y, -1)$ a $z \in X$
    - technické opatření pro zpracování b (narazíme na oddělovač @)
      - $(q@, yp@)$ kdykoli $\delta(q, b) = (p, y, +1)$
      - $(q@, py@)$ kdykoli $\delta(q, b) = (p, y, 0)$
      - $(zq@, pzy@)$ kdykoli $\delta(q, b) = (p, y, -1)$ a $z \in X$
    - přijímání a mazání pásky; pro $f \in F$ a $x, y \in X$
      - (xfy,f)
      - $(@fy, @f)$
      - $(xf@,f@)$
      - $(f@@,@)$

@ K_0 @ K_1 @ ... @ K_m @ ... @ f @@
@q_0w@ K_1 @ ... @ K_m @ ufv @ ... @ f @@

Př.: $\delta(q,C) = (p, E, +1)$
... @AB
... @ABqCD@AB

... @ABqCD@
... @ABqCD@ABEpD@

Př.: ... @ABfCDE@AfDE@fE@f@@
... @ABfCDE@AfDE@fE@f@@

Nerozhodnutelnost u **gramatik** (1)

- pro bezkontextové gramatiky G_1 a G_2 je nerozhodnutelné, zda $L(G_1) \cap L(G_2) = \emptyset$
  - přesněji $\{kód(G_1)\#kód(G_2) \mid G_1$ a G_2 jsou bezkontextové gramatiky a $L(G_1) \cap L(G_2) = \emptyset\}$ je nerozhodnutelný jazyk (není rekurzivní)
  - mějme instanci PKP $(w_1,x_1), (w_2,x_2), \dots, (w_n,x_n)$ nad abecedou X
    - položíme $V_T = X \cup \{a_1, a_2, \dots, a_n\}$
      - $G_1 = (\{S\}, V_T, S, \{ S \to w_i S a_i \mid w_i a_i \mid i=1,2,\dots,n\})$
        - generuje slova $w_{i1}.w_{i2} \dots w_{ik} a_{ik}.a_{ik-1} \dots a_{i1}$
      - $G_2 = (\{S\}, V_T, S, \{ S \to x_i S a_i \mid x_i a_i \mid i=1,2,\dots,n\})$
        - generuje slova $x_{i1}.x_{i2} \dots x_{ik} a_{ik}.a_{ik-1} \dots a_{i1}$
    - instance PKP má řešení $\Leftrightarrow$ $L(G_1) \cap L(G_2) \neq \emptyset$
- pro bezkontextovou gramatiku G je nerozhodnutelné, zda G je jednoznačná
  - $G = (\{S, S_1, S_2\}, V_T, S,$ $\{S \to S_1 \mid S_2\} \cup$ $\{S_1 \to w_i S_1 a_i \mid w_i a_i \mid i=1,2,\dots,n\} \cup$ $\{S_2 \to x_i S_2 a_i \mid x_i a_i \mid i=1,2,\dots,n\})$
  - instance PKP má řešení $\Leftrightarrow$ G je víceznačná

Nerozhodnutelnost u **gramatik** (2)

- pro bezkontextovou gramatiku G je nerozhodnutelné, zda L(G)=X*
  - $G_1 = (\{S\}, V_T, S, \{ S\rightarrow w_iSa_i |\ w_ia_i \mid i=1,2,\ldots,n\})$
    - generuje slova $w_{i_1}.w_{i_2}\ldots w_{i_k}a_{i_k}.a_{i_{k-1}}\ldots a_{i_1}$
  - $G_2 = (\{S\}, V_T, S, \{ S\rightarrow x_iSa_i |\ x_ia_i \mid i=1,2,\ldots,n\})$
    - generuje slova $x_{i_1}.x_{i_2}\ldots x_{i_k}a_{i_k}.a_{i_{k-1}}\ldots a_{i_1}$
  - $L(G_1)$ a $L(G_2)$ jsou deterministické bezkontextové jazyky
    - z uzavřenosti na doplněk jsou deterministické bezkontextové i $-L(G_1)$ a $-L(G_2)$
    - z uzavřenosti bezkontextových jazyků na konečná sjednocení existuje bezkontextová gramatika G, že $L(G) = -L(G_1) \cup -L(G_2)$
    - instance PKP má řešení $\Leftrightarrow L(G_1)\cap L(G_2)\neq\emptyset \Leftrightarrow -L(G_1) \cup -L(G_2) \neq X^* \Leftrightarrow L(G) \neq X^*$
- následující problémy jsou rovněž nerozhodnutelné:
  - L(G) = R pro bezkontextovou gramatiku G a regulární jazyk R
    - za R zvolme X*
  - $R \subseteq L(G)$ pro bezkontextovou gramatiku G a regulární jazyk R
    - za R zvolme X*
  - $L(G_1)=L(G_2)$ pro bezkontextové gramatiky G_1 a G_2
    - G_1 taková, že $L(G_1) = X^*$
  - $L(G_1)\subseteq L(G_2)$ pro bezkontextové gramatiky G_1 a G_2
    - G_1 taková, že $L(G_1) = X^*$