Vysoké učení technické v Brně
Fakulta informačních technologií

TIN

Úloha 2

24. listopadu 2023

Adam Zvara (xzvara01)

1. Rozhodnite či jazyk $L_{prime} = \{w \in \Sigma^* : |w| \text{ je prvočíslo}\}$ nad abecedou $\Sigma = \{a, b\}$ je bezkontextový.

Dôkaz sporom:

Predpokládame, že jazyk L_{prime} je bezkontextový. Potom pre neho musí platiť Pumping Lemma pre bezkontextové jazyky:

$$\begin{aligned} L \in \mathcal{L}_2 \implies \exists k > 0 : \forall z \in \Sigma^* : z \in L \wedge |z| \geq k \implies \\ (\exists uvwxy \in \Sigma^* : z = uvwxy \wedge vx \neq \epsilon \wedge |vwx| \leq k \wedge \forall i \geq 0 : uv^i wx^i y \in L) \end{aligned}$$

Uvažujme ľubovoľné $k > 0$. Potom si vyberieme slovo $z = a^p$, kde p je ľubovoľné prvočíslo také, že $p \geq k$ (určite existuje, keďže prvočísel je nekonečne veľa). Platí, že $z \in L_{prime}$ a zároveň že $|z| = p \geq k$. Potom všetky možné rozloženia slova z môžeme zapísať v tvare:

$$\begin{aligned} vx &= a^r \\ uwy &= a^q \\ r + q &= p \\ r &\geq 1 \end{aligned}$$

Výsledné slovo (po "vypumpovaní") bude v tvare $uv^i wx^i y = a^{ir} a^q \to |uv^i wx^i y| = ir + q$. Keďže je naším cieľom dosiahnuť aby slovo po vypumpovaní nepatrilo do jazyka L_{prime}, musíme zabezpečiť, aby $ir + q$ bolo zložené číslo - teda násobok dvoch čísel, pričom ani jedno z nich nie je 1:

$$ir + q = (X)(Y)$$

Určite budeme musieť do X a Y nejakým spôsobom zakomponovať čísla r a q. Keďže vieme, že $r \geq 1$, môžeme ho využiť na to, aby sme z člena X vytvorili prvý člen, ktorý nebude rovný 1 (ak $r \geq 1 \Rightarrow r + 1 \geq 2$):

$$ir + q = (r + 1)(Y)$$

S členom Y potrebujeme spraviť to isté - zabezpečiť, aby bol ≥ 2 a zároveň aby obsahoval číslo q. Na číslo q sa však nevzťahujú žiadne podmienky (teda môže byť aj nulové, ale nemusí). Môžeme však opäť využiť fakt, že $r \geq 1$. Rovnaký trik nám však pri tomto člene nepomôže, pretože by sme neboli schopní vyjadriť číslo i:

$$\begin{aligned} ir + q &= (r + 1)(q + r + 1) \\ ir + q &= rq + r^2 + r + q + r + 1 \\ ir &= r(q + r + 2) + 1 \end{aligned}$$

Môžeme však použiť iný trik, a to že $r \geq 1 \Rightarrow 2r \geq 2$. Potom môžeme vyjadriť člen i:

$$\begin{aligned} ir + q &= (r + 1)(q + 2r) \\ ir + q &= rq + 2r^2 + q + 2r \\ ir &= r(q + 2r + 2) \\ i &= q + 2r + 2 \end{aligned}$$

Určite platí, že $(r + 1)(q + 2r)$ je zložené číslo $(r \geq 1)$. Teda pre ľubovoľné $k > 0$ sme našli slovo $z = a^p$, ktoré spĺňa podmienky PL a zároveň sme pre všetky platné rozdelenia daného slova $uvwxy$ ukázali, že existuje $i = q + 2r + 2 \geq 0$ také, že $uv^i wx^i y \notin L_{prime}$. Jazyk L_{prime} teda nie je bezkontextový.

2. Rozhodnite, či je jazyk $L_{BKG} = \{\langle M\rangle\#\langle G\rangle : M$ je TS, G je BKG, $L(G) \subseteq L(M)\}$ je (čiastočne) rozhodnuteľný alebo nerozhodnuteľný.

Jayzk L_{BKG} je nerozhodnuteľný. Nerozhodnuteľnosť ukážeme pomocou redukcie z $co - HP$, teda $co - HP \leq L_{BKG}$. Zostrojíme redukciu $\sigma : \{0, 1, \#\} \rightarrow \{0, 1, \#\}$, ktorá každému vstupu $x \in \{0, 1, \#\}$ priradí kód TS M' a kód bezkontextovej gramatiky G':

$$\sigma(\langle M\rangle\#\langle w\rangle) = \langle M'\rangle\#\langle G'\rangle$$

Kód bezkontextovej gramatiky G' bude v takom tvare, aby gramatika G' prijímala všetky jej vstupy, teda $L(G') = \Sigma^*$. Určite platí, že G' je bezkontextová gramatika (keďže je G' regulárna). TS M' pracuje nasledujúcim spôsobom:

- M' overí, že $\langle M\rangle\#\langle w\rangle$ sú validné kódy pre TS M a jeho vstup w, ak nie, tak M' zamietne ($L(M') = \emptyset$). Kódy M a w sú zakódované v stavovom zariadení TS M'.
- M' si zapamätá dĺžku svojho vstupu w', teda hodnotu $|w'|$.
- M' vykoná najviac $|w'|$ krokov simulácie stroja M na reťazci w.
- Ak simulácia M na w skončí v menšom alebo rovnakom počte krokov ako $|w'|$, TS M' odmietne svoj vstup ($L(M') = \emptyset$).
- Ak simulácia neskončí ani do $|w'|$ krokov, M' akceptuje svoj vstup ($L(M') = \Sigma^*$).

Vytvorená redukcia σ zachováva členstvo:

$$\langle M\rangle\#\langle w\rangle \in co - HP \Rightarrow L(G') = \Sigma^* \wedge L(M') = \Sigma^* \Rightarrow L(G') \subseteq L(M') \Rightarrow \langle M'\rangle\#\langle G'\rangle \in L_{BKG}$$

$$\langle M\rangle\#\langle w\rangle \notin co - HP \Rightarrow L(G') = \Sigma^* \wedge L(M') = \emptyset \Rightarrow L(G') \nsubseteq L(M') \Rightarrow \langle M'\rangle\#\langle G'\rangle \notin L_{BKG}$$

$$\langle M\rangle\#\langle w\rangle \in co - HP \iff \sigma(\langle M\rangle\#\langle w\rangle) \in L_{BKG}$$

Redukcia σ je totálna a rekurzívne vyčísliteľná funkcia, ktorú vieme implementovať úplným TS M_σ, ktorý realizuje následujúce operácie:

- vytvorenie gramatiky generujúcej (prijímajúcej) Σ^* (gramatika určite existuje a zapísanie jej kódu je realizovateľné pomocou úplného TS)
- test správneho kódovania ⟨M⟩#⟨w⟩ (syntaktická analýza je realizovateľná úplným TS)
- zapamätanie dĺžky vstupu na páske - elementárna operácia
- výpis kódu univerzálneho TS (konštantná operácia)

Dokázali sme, že redukcia σ je totálna rekurzívne vyčísliteľná funkcia, ktorá zachováva členstvo reťazcov v daných jazykoch a teda jazyk L_{BKG} je nerozhodnuteľný.

3. Plný binárny strom a dokazovanie tvrdení:

(a) Množina V je spočítateľne nekonečná.

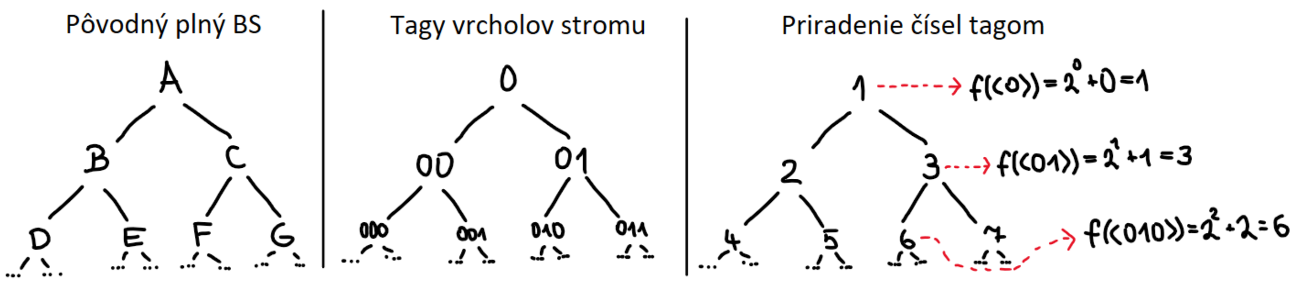
Dôkaz spočítateľnosti realizujeme pomocou ukázania, že existuje bijekcia medzi vrcholmi stromu a množinou prirodzených čísel. Na základe podmienky zo zadania $img(E_L) \cap img(E_R) = \emptyset$ vieme povedať, že množina V je určite nekonečná (pre každý vrchol vieme povedať, že vrcholy v pravom a ľavom podstrome sa určite neopakujú - množina V by mohla byť konečná v prípade, ak by sa mohli opakovať vrcholy v ľavom a pravom podstrome). Bijekciu môžeme realizovať pomenovaním (*otagovaním*) jednotlivých vrcholov nasledujúcim spôsobom:

- ak je uzol koreň, tak $\langle \text{tag} \rangle = 0$
- tag uzlu je $\langle$tag prechodcu ·[1] $\{0/1\}\rangle$ - 0 značí, že sa jedná o ľavého potomka a 1 pravého potomka

Je zrejmé, že každému vrcholu je pridelený unikátny tag. Potom môžeme vytvoriť funkciu, ktorá každému tagu jednoznačne pridelí jedno číslo z množiny prirodzených čísel:

$$f(\langle \text{tag} \rangle) = 2^{w-1} + bin(\text{tag})$$

kde w je dĺžka tagu (predstavuje hĺbku vrcholu v danom strome) a bin je funkcia realizujúca prevod tagu z binárnej sústavy do desiatkovej. Potom pre každý tag (vrchol stromu) existuje práve jedno priradenie - teda mapovacia funkcia je surjektívna a zároveň injektívna. Príklad otagovania stromu:



Dokázali sme nájsť bijeciu, ktorá realizuje mapovanie jednotlivých vrcholov na množinu prirodzených čísel, čím sme dokázali, že množina vrcholov V je spočítateľne nekonečná.

(b) Nech ofarbenie stromu T je funkcia, ktorá priraďuje každému vrcholu $v \in V$ farbu $c(v) \in \{red,\ black\}$. Množina všetkých ofarbení stromu je nespočítateľne nekonečná.

Dôkaz sporom:

Predpokladajme, že množina všetkých ofarbení stromu je spočítateľne konečná. Teda existuje bijekcia z množiny prirodzených čísel na množinu všetkých ofarbení uzlov stromu. Ak si usporiadáme vrcholy stromu do postupnosti (napríklad pomocou funkcie z predchádzajúceho príkladu), môžeme túto bijekciu zobraziť v nekonečnej matici, kde stĺpce predstavujú vrcholy, riadky jednotlivé ofarbenia daného stromu a buňky matice predstavujú farbu daného vrcholu v danom strome ($f_{ij} = \{red,\ black\}$).

$$\begin{array}{cc} & \begin{array}{ccccc} \text{V1} & \text{V2} & \text{V3} & \text{V4} & \end{array} \\ \begin{array}{c} \text{T0} \\ \text{T1} \\ \text{T2} \\ \text{T3} \\ \vdots \end{array} & \begin{bmatrix} f_{00} & f_{01} & f_{02} & f_{03} & \cdots \\ f_{10} & f_{11} & f_{12} & f_{13} & \cdots \\ f_{20} & f_{21} & f_{22} & f_{23} & \cdots \\ f_{30} & f_{31} & f_{32} & f_{33} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \end{array}$$

Ak by množina všetkých ofarbení bola skutočne spočítateľná, táto matica by obsahovala všetky možné ofarbenia vrcholov daného stromu. Ak si však vytvoríme nový strom obrátením farieb (*red* $\rightarrow$ *black*, *black* $\rightarrow$ *red*) vrcholov na hlavnej diagonále matice bijekcie tak dostaneme strom, ktorý sa určite v tejto matici nenachádza (líši sa s každým ofarbením stromu aspoň v jednom prvku). Z toho plynie, že táto matica neobsahuje všetky možné ofarbenia vrcholov stromu, čo je ale spor, pretože sme predpokladali, že obsahuje. Predpoklad, že množina všetkých ofarbení je spočítateľná neplatí a musí platiť jej opak.

[1]Operátor · značí konkatenáciu reťazcov

4. Algoritmus na výpočet množiny ${}_aN_a = \{A \in N \mid \exists w \in \Sigma^* : A \Rightarrow_G^+ w \wedge \exists u \in \Sigma^* : w = aua\}$

Vstup: bezkontextová gramatika $G = (N, \Sigma, P, S)$ nad abecedou Σ.
Výstup: množina neterminálov ${}_aN_a$.
Metóda:

1. Výpočet množiny neterminálov N_ϵ.
2. Výpočet množiny neterminálov $N_t = \{A \in N \mid A \Rightarrow_G^+ w, w \in \Sigma^*\}$.
   (a) $N_t^0 = \emptyset, i = 1$.
   (b) $N_t^i = \{A \in N \mid \exists (A \to \alpha) \in P \wedge \alpha \in (N_t^{i-1} \cup \Sigma)^*\}$.
   (c) Ak $N_t^i \neq N_t^{i-1}$ tak $i = i + 1$ a pokračuj v bode a). Ak $N_t^i = N_t^{i-1}$ tak $N_t = N_t^i$.
3. Výpočet množiny neterminálov $N_a = \{A \in N \mid A \Rightarrow_G^+ wa, w \in \Sigma^*\}$.
   (a) $N_a^0 = \emptyset, i = 1$.
   (b) $N_a^i = \{A \in N \mid \exists (A \to \alpha) \in P \wedge \alpha \in (N_t \cup \Sigma)^*(\{a\} \cup N_a^{i-1})N_\epsilon^*\}$.
   (c) Ak $N_a^i \neq N_a^{i-1}$ tak $i = i + 1$ a pokračuj v bode a). Ak $N_a^i = N_a^{i-1}$ tak $N_a = N_a^i$.
4. Výpočet množiny neterminálov ${}_aN = \{A \in N \mid A \Rightarrow_G^+ aw, w \in \Sigma^*\}$.
   (a) ${}_aN^0 = \emptyset, i = 1$.
   (b) ${}_aN^i = \{A \in N \mid \exists (A \to \alpha) \in P \wedge \alpha \in N_\epsilon^*(\{a\} \cup {}_aN^{i-1})(N_t \cup \Sigma)^*\}$.
   (c) Ak ${}_aN^i \neq {}_aN^{i-1}$ tak $i = i + 1$ a pokračuj v bode a). Ak ${}_aN^i = {}_aN^{i-1}$ tak ${}_aN = {}_aN^i$.
5. Výpočet množiny neterminálov ${}_aN_a = \{A \in N \mid \exists w \in \Sigma^* : A \Rightarrow_G^+ w \wedge \exists u \in \Sigma^* : w = aua\}$.
   (a) ${}_aN_a^0 = \emptyset, i = 1$.
   (b) ${}_aN_a^i = \{A \in N \mid \exists (A \to \alpha) \in P \wedge (\alpha \in N_\epsilon^*(\{a\} \cup {}_aN)(N_t \cup \Sigma)^*(\{a\} \cup N_a)N_\epsilon^* \vee \alpha \in N_\epsilon^*({}_aN_a^{i-1})N_\epsilon^*)$.
   (c) Ak $N_{aw}^i \neq N_{aw}^{i-1}$ tak $i = i + 1$ a pokračuj v bode a). Ak $N_{aw}^i = N_{aw}^{i-1}$ tak $N_{aw} = N_{aw}^i$.

Realizácia algoritmu na gramatike s pravidlami:

$S \to aWU \mid UWa \qquad W \to YacU \mid Xa \qquad Y \to X \mid \epsilon \qquad X \to aXa \qquad U \to bcaYY \mid Ucb$

$N_\epsilon = \{Y\}$
$N_t^0 = \emptyset \quad N_t^1 = \{Y\} \quad N_t^2 = \{Y, U\} \quad N_t^3 = \{Y, U, W\} \quad N_t^4 = \{Y, U, W, S\} = N_t^5 = N_t$
$N_a^0 = \emptyset \quad N_a^1 = \{S, U\} \quad N_a^2 = \{S, U, W\} = N_a^3 = N_a$
${}_aN^0 = \emptyset \quad {}_aN^1 = \{S, W\} = {}_aN^2 = {}_aN$
${}_aN_a^0 = \emptyset \quad {}_aN_a^1 = \{S, W\} = {}_aN_a^2 = {}_aN_a$