Univerzita Komenského Bratislava, Fakulta Matematiky Fyziky a Informatiky

Formálne Jazyky a Automaty 2, 2000/2001
Úloha 7.5

Meno: Martin Homola
Krúžok: 2i2
Dátum: 9.4.2001

foja7.5, martin homola, 2i2

Zadanie:
Nech je daný TM M. Dokážte, že jazyk neplatných výpočtov $L_{NPV}(M)$ je bezkontextový.

Definície:

Zásobníkový automat (Push Down Automaton, PDA) je 7-ica $(K, \Sigma, \Gamma, \delta, q_0, z_0, F)$, kde K je končná množina stavov, Σ je vstupná abeceda, Γ je pracovná abeceda, δ je prechodová funkcia, q_0 je počiatočný stav, z_0 je počiatočny znak na zásobníku a F je množina akceptačných stavov.

Turingov Stroj (Turing's Machine, TM) je 6-tica $(K, \Sigma, \Gamma, \delta, q_0, F)$, kde K je končná množina stavov, Σ je vstupná abeceda, Γ je pracovná abeceda, δ je prechodová funkcia, q_0 je počiatočný stav a F je množina akceptačných stavov.

Konfigurácia TM M je slovo z $K\{B\}(\Gamma - \{B\})^* \cup (\Gamma - \{B\})^* K (\Gamma - \{B\})^*$ (predpokladáme, že $K \cap \Gamma = \emptyset$)

Jazyk platných výpočtov TM M
$L_{PV}(M) = \{u_0 \# u_1^R \# u_2 \# u_3^R \# ... \# u_n^{(R)} \mid n \geq 1, \forall i \in \{0, ..., n-1\}\ u_i \vdash_M u_{i+1}, u_0$ je počiatičná konfigurácia M, u_n je akceptačná konfigurácia $M\}$

Jazyk neplatných výpočtov TM M $L_{NPV}(M) = L_{PV}(M)^C$

Riešenie:

Nech je daný TM $M = (K, \Sigma, \Gamma, \delta, q_0, F)$. Zostrojíme PDA A ktorý akceptuje $L_{NPV}(M)$.
PDA A sa skladá troch nedeterministických vetiev.
V prvej vetve A overí, či je prvá konfigurácia na páske akceptačnou konfiguráciou M, ako áno tak sa výpočet v tejto vetve zasekne, ak nie tak A akceptuje.
V druhej vetve A nedeterministicky nájde poslednú konfiguráciu a overí, či obsahuje znak $q \in K - F$ ak ano - akceptuje, ak nie - výpočet v tejto vetve sa zasekne.
Tretia vetva postupne prečíta celé vstupné slovo (jednotlivé znaky pritom ukladá do zasobníku), pričom sa na každom znaku # nedeterministicky rozhodne, či bude čítať dalej, alebo pre poslednú konfigutráciu v_1, ktorú má v zásobníku a pre konfiguráciu v_2 nasledujúcu za # overí, či sú legálne, a či $v_1 \vdash_M v_2$. vďaka nedeterminizmu je toto overené pre každé 2 po sebe idúce konfigurácie. A akceptuje hneď ako nájde 2 po sebe idúce konfigurácie v_1, v_2, z ktorých aspoň 1 nie je legálna alebo $v_1 \nvdash_M v_2$.

Koštrukcia:
$A = (K', \{\Gamma \cup K \cup \{\#\}, \Sigma' \cup \{z_0\}, \delta', q_0', z_0, F')$,
$K' = \{A, AI, R, q_0', p_i, r_j, s_0, s_0\#, s_3, s_m[^{a1a2a3}_{b1b2b3}], t_0, t_0\#, t_3, t_m[^{a1a2a3}_{b1b2b3}] \mid \forall i, m, n \in \{0,1\}, \forall j \in \{0,1,2,3\}$,
$\forall a_k, b_l \in \Gamma' \cup -\}$,
$F' = \{A, AI, p_0, r_3, s0\#, t0\#, s1*, s2*, t1*, t2*\} - (S \cup T)$,
$S = \{s2[^{-a_1a_2}_{-b_1b_2}] \mid \forall a_1, b_1 \in K \forall b_2 \in \Gamma - \{B\} \forall a_2 \in \{\#, z_0\}\}$
$\cup \{s2[^{-a_1a_2}_{-b_1b_2}] \mid \forall a_1, b_2 \in K \forall b_1 \in \Gamma - \{B\} \forall a_2 \in \{\#, z_0\}\}$
$\cup \{s2[^{a_1a_2a_3}_{b_1b_2b_3}] \mid \forall a_2, b_1 \in K \forall a_1, b_2, c_1, c_2 \in \Gamma - \{B\}\}$,
T je analogická mnozina ako S - niektoré neakceptačné stavy odvodené od t_2*

Funkcia δ':
A sa skladá z troch vetiev. Na vstupe dostane slovo $w \in \Sigma'$ a nedeterministicky spustívýpočet vo všetkých troch vetvách. Z počiatočného stavu q_0' prejde automat bez načítania znaku zo vstupu (ε-ový krok) do jedného zo stavov p_0, r_0, s_0. Tri vetvy začínajúce v týchto stavoch sú disjunktné.

$\delta'(q_0', \varepsilon, z_0) = \{(p_0, z_0), (r_0, z_0), (s_0, z_0)\}$

1. vetva: prieskum prvej konfigurácie vstupu w:
PDA A najprv skontroluje prvý znak. Ak to nie je q_0 tak akceptuje. Ak to je q_0, tak prekontroluje zvyšok pásky až po najbližšiu # (t.j. ak $\# \notin w$ tak celú pásku) a akceptuje iba ak narazína znak $a \notin \Sigma$. Na prvej # alebo na konci pásky sa zasekne. Stav p_0 je akceptačný, teda ak $w = \varepsilon$, tak A akceptuje.

$\delta'(p_0, a, z_0) = \{(A, z_0)\} \forall a \in \Gamma' \wedge a \neq q_0$
$\delta'(p_0, q_0, z_0) = \{(p_1, z_0)\}$
$\delta'(p_1, a, z_0) = \{(A, z_0)\} \forall a \in (\Gamma \cup K) - \Sigma$
$\delta'(p_1, a, z_0) = \{(p_1, z_0) \forall a \in \Sigma\}$

2. vetva: prieskum poslednej konfigurácie vstupu w:
A nedeterministicky skontroluje všetky konfigurácie (t.j. číta vstup w a na každej # (ako aj na začiatku vstupu) sa rozhodne, či skontroluje nasledujúcu konfiguráciu alebo alebo číta ďalej). Pri kontrole A hľadá znak $a \in K - F$. Ak ho nenájde - zasekne sa. Ak ho nájde, preverí, či skutočne kontroluje poslednú konfiguráciu - ak áno akceptuje - ak nie zasekne sa.

$\delta'(r_0, \varepsilon, z_0) = \{(r_1, z_0), (r_2, z_0)\}$
$\delta'(r_1, a, z_0) = \{(r_1, z_0)\} \forall a \in \Sigma' - \{\#\}$
$\delta'(r_1, \#, z_0) = \{(r_1, z_0), (r_2, z_0)\}$
$\delta'(r_2, a, z_0) = \{(r_2, z_0)\} \forall a \in \Gamma \cup F$
$\delta'(r_2, a, z_0) = \{(r_3, z_0)\} \forall a \in K - F$
$\delta'(r_2, \#, z_0) = \{(R, z_0)\}$
$\delta'(r_3, a, z_0) = \{(r_3, z_0)\} \forall a \in \Sigma' - \{\#\}$
$\delta'(r_3, \#, z_0) = \{(R, z_0)\}$

3. vetva: kontrola po sebe idúcich konfigurácií:
V tejto vetve A postupne číta vstup w a ukladá si ho do zásobníka. Na každej # sa nedeterministicky rozhodne a buď pokračuje v kopírovanívstupu do zásobníku, alebo pre poslednú konfiguráciu v zásobníku v_1 a na páske bezprostredne nasledujúcu konciguráciu v_2 preverí, či sú korektné, a či $v_1 \vdash_M v_2$. Ak toto neplatíA akceptuje. Ak je vstup naopak korektný tak sa A zasekne.

Riadiaca časť tretej vetvy:
$\delta'(s_0, a, z) = \{(s_0, az)\} \forall a \in \Sigma' - \{\#\}$
$\delta'(s_0\#, a, z) = \delta'(s_0, a, z) \forall a \in \Sigma' - \{\#\}$
$\delta'(s_0, \#, z) = \{(t_1, z), (t_0\#, \#z)\}$
$\delta'(t_0, a, z) = \{(t_0, az)\} \forall a \in \Sigma' - \{\#\}$
$\delta'(t_0\#, a, z) = \delta'(t_0, a, z) \forall a \in \Sigma' - \{\#\}$
$\delta'(t_0, \#, z) = \{(s_1, z), (s_0\#, \#z)\}$

Kontrolná časť tretej vetvy je zložitejšia. Nastupuje vždy po # na vstupe. V prvom rade je treba rozlišovať paritu počtu doposiaľ načítaných znakov #. Keďže v slove z $L_{PV}(M)$ je každ'a druhá konfigurácia reverzná - ak sa A nachádza na párnej # v poradí, začína kontrolu v stave $s_0\#$ - ak sa naopak nachádza na nepárnej # v poradí, začína kontrolu v stave $t_0\#$. V stavoch $s*$ spracú va konfigurácie odpredu, v stavoch $t*$ odzadu.
Stavy $s_1[{}^{***}_{***}], s_2[{}^{***}_{***}], t_1[{}^{***}_{***}] a t_2[{}^{***}_{***}]$ slúžia na 'bufferovanie' vstupu - v $*[{}^{a_1 a_2 a_3}_{b_1 b_2 b_3}]$ su a_1, a_2, a_3 posledné tri znaky zo zásobníku a b_1, b_2, b_3 posledné tri znaky z pásky.

Činnost A v stavoch $t*$ je analogická ako v stavoch $s*$, preto sa ďalej budeme zaoberať len činnosťou A v stavoch $s*$. Automat v stave s_1 po znakoch porovnáva obsah pásky s obsahom zásobníka a hľadá nejakú odlišnosť alebo znak $q \in K$. Keď ju nájde spustípomerne zložitú kontrolu, či $v_1 \vdash_M v_2$. Na toto A využíva 'buffer' v stavoch $s1*, s2*$, pretože na to aby vedel rozhodnúť, či je zmena v súlade s δ funkciou M potrebuje porovnať niekoľko znakov z pásky aj zo zásobníka súčasne. Stavy $s1*$ sa pritom využijú len na kontrolu šeciálnych prípadov na zčiatku (v okolíznaku #). Ak A v tejto fáze zistíže konfigurácie nemohli vo výpočte po sebe nasledovať, tak akceptuje. Ak odlišnosť súhlasís δ funkciou, tak A v stave $s3$ preveríkoreknosť zvyšku konfiguráciív_1 a v_2. Kazdá dalšia odlišnosť alebo výskyt znaku $q \in K$ vedie k akceptácii. Ak je všetko v poriadku a A nájde # alebo koniec pásky a v zásobníku tiež # alebo dno (z_0), tak sa zasekne.

Náčrt dôkazu:
Keďže ide o pomerne zložitý automat a tiež vzhľadom na konštrukciu (nedeterminismus, disjunktné vetvy) - je priam žiadúce rozdelit dôkaz na niekoľko samostatných tvrdení.

1. V prvej vetve PDA A akceptuje práve vtedy, keď prvá konfigurácia na vstupe nie je počiatočnou konfiguráciou M, alebo obsahuje nejaký znak $a \notin \Sigma$.

2. V druhej vetve A akceptuje práve vtedy, keď posledná konfigurácia na vstupe obsahuje znak $q \in K - F$ (t.j v žiadnom prípade nemôze byť akceptačná).

3. V tretej vetve A akceptuje práve vtedy, keď vstup w obsahuje dve po sebe idúce konfigurácie v_1 a v_2 také, že v_1 alebo v_2 nie je legálna v M alebo $v_1 \not\vdash_M v_2$

Tvrdenia 1. a 2. sa dokážu ľahko napr. indukciou na dĺžku výpočtu alebo indukciou na dĺžku vstupu. A sa tu správa ako jednoduchý konečný automat (nepoužíva zásobník).

Tvrdenie 3. je zložitejšie. Najprv pomocné tvrdenie:

3.1 Ak je A na stave s_0 (t_0) v zásobníku má $[\ldots\#]v_1$ a na vstupe $v_2[\#\ldots]$, tak akceptuje práve vtedy keď v_1^R (v_1) alebo v_2 (v_2^R) nie je legálnou kofiguráciou v M, alebo $v_1^R \not\vdash_M v_2$ ($v_1 \not\vdash_M v_2^R$).

Tvrdenie 3.1. možno dokázať indukciou na dĺžku vstupu. Z indukčného predpokladu pre ľubovolné slovo na zásobníku a pre dĺzku vstupu $\leq n$ vyvodíme rozlíšením všetkých možných prípadov (indukčného predpokladu a posledných pár znakov) platnosť tvrdenia pre ľubovolný zásobník a vstup dĺžky $n+1$

Tvrdenie 3. potom už ľahko dokážeme indukciou na počet # vo vstupe w.

Tvredenie $L(A) = L_{NPV}(M)$ je dôsledkom tvrdení 1., 2. a 3.