Teoretická informatika (TIN) - Úkol 2

Šimon Stupinský - xstupi00@stud.fit.vutbr.cz

Hodnotenie:

			0	

1. ÚLOHA

Zadanie: Uvažujte jazyk $L = \{w \in \{a,b\}^* \mid \#_a(w) = \#_b(w)\}$, kde $\#_x(w)$ značí počet výskytov symbolu x v reťazci w. Dokážte, že jazyk L je bezkontextový. Postupujte nasledovne:

(a) Navrhnite gramatiku G, ktorá bude mať za cieľ generovať jazyk L.

(b) Pomocou indukcie k dĺžke slova $w \in L$ dokážte, že $L = L(G)$.

---

a) Navrhnite gramatiku G, ktorá bude mať za cieľ generovať jazyk L:

$$G = (N,\ \Sigma,\ P,\ S)\ =\ (\{S\},\ \{a,b,\varepsilon\},\ \{S \to \varepsilon,\ S \to SS,\ S \to aSb,\ S \to bSa\},\ S)$$

b) Pomocou indukcie k dĺžke slova $w \in L$ dokážte, že $L = L(G)$:

I. $\mathbf{L(G) \subseteq L}$

Najprv ukážeme, že každý reťazec vygenerovaný gramatikou **G** obsahuje rovnaký počet symbolov **a** a **b**, pričom použijeme dôkaz indukciou k dĺžke slova $\mathbf{w} \in \mathbf{L(G)}$: $\forall \mathbf{w} \in \mathbf{L(G)} :\ \#_{\mathbf{a}}(\mathbf{w}) = \#_{\mathbf{b}}(\mathbf{w})$.

Bázový prípad: $\mathbf{n = 0}$. Jediný reťazec ktorý dokážeme vygenerovať v gramatike **G** s dĺžkou $\mathbf{n = 0}$ je ε, pričom tento reťazec dokážeme v gramatike **G** vygenerovať využitím jednej z dvoch nasledujúcich derivácií z počiatočného nonterminálu **S**:

$$a)\ n = 0 :\ w = \varepsilon,\ S \underset{G}{\Rightarrow} \varepsilon \qquad\qquad b)\ n = 0 :\ w = \varepsilon,\ S \underset{G}{\Rightarrow} SS \underset{G}{\Rightarrow} S \underset{G}{\Rightarrow} \varepsilon$$

Je zrejmé, že počet symbolov **a** a **b** je v slove $\mathbf{w} = \varepsilon$ rovnaký ($\#_{\mathbf{a}}(\mathbf{w}) = \#_{\mathbf{b}}(\mathbf{w}) = \mathbf{0}$) a preto $\mathbf{w} \in \mathbf{L}$.

Indukčný predpoklad: $\forall\, w \in L(G)\ :\ (S \underset{G}{\Rightarrow}^* w\ \wedge\ |w| = n) \Rightarrow w \in L$.

Indukčný krok: Ukážeme, že platí: $\forall\, w \in L(G)\ :\ (S \underset{G}{\Rightarrow}^* w\ \wedge\ |w| = n+2) \Rightarrow w \in L$.

Gramatika **G** generuje len reťazce s párnou dĺžkou a preto v tomto indukčnom kroku uvažujeme reťazce s dĺžkou $\mathbf{n+2}$. Pre všetky možné varianty generovania dlhších slov ukážeme, že toto trvdenie bude platné s využitím indukčného prepokladu:

(a) $S \underset{G}{\Rightarrow} aSb \underset{G}{\Rightarrow}^* aw'b = w, \quad |w| = n+2,\ |w'| = n$

- Podľa indukčného predpokladu $\mathbf{w'} \in \mathbf{L}$ a teda $\#_{\mathbf{a}}(\mathbf{w'}) = \#_{\mathbf{b}}(\mathbf{w'})$.
- $\mathbf{w = aw'b}$, je teda zrejmé že $\#_{\mathbf{a}}(\mathbf{w}) = \#_{\mathbf{a}}(\mathbf{w'}) + \mathbf{1}\ \wedge\ \#_{\mathbf{b}}(\mathbf{w}) = \#_{\mathbf{b}}(\mathbf{w'}) + \mathbf{1}$.
- Z toho plynie, že $\#_{\mathbf{a}}(\mathbf{w}) = \#_{\mathbf{b}}(\mathbf{w})$ a teda $\mathbf{w} \in \mathbf{L}$ čím sme dokázali, že indukčný krok platí.

(b) $S \underset{G}{\Rightarrow} bSa \underset{G}{\Rightarrow}^* bw'a = w, \quad |w| = n+2,\ |w'| = n$

- Podľa indukčného predpokladu $\mathbf{w'} \in \mathbf{L}$ a teda $\#_{\mathbf{a}}(\mathbf{w'}) = \#_{\mathbf{b}}(\mathbf{w'})$.
- $\mathbf{w = bw'a}$, je teda zrejmé že $\#_{\mathbf{a}}(\mathbf{w}) = \#_{\mathbf{a}}(\mathbf{w'}) + \mathbf{1}\ \wedge\ \#_{\mathbf{b}}(\mathbf{w}) = \#_{\mathbf{b}}(\mathbf{w'}) + \mathbf{1}$.
- Z toho plynie, že $\#_{\mathbf{a}}(\mathbf{w}) = \#_{\mathbf{b}}(\mathbf{w})$ a teda $\mathbf{w} \in \mathbf{L}$ čím sme dokázali, že indukčný krok platí.

(c) $S \underset{G}{\Rightarrow} SS \underset{G}{\Rightarrow}^{*} w_1 w_2 = w, \quad w_1 \in \{aw_1'b, bw_1'a, \varepsilon\},\ w_2 \in \{aw_2'b, bw_2'a, \varepsilon\},\ |w_1'| = n_1,\ |w_2'| = n_2$

- Podľa indukčného predpokladu $\mathbf{w_1'}, \mathbf{w_2'} \in \mathbf{L}$ a teda $\#_{\mathbf{a}}(\mathbf{w_1'}) = \#_{\mathbf{b}}(\mathbf{w_1'}) \wedge \#_{\mathbf{a}}(\mathbf{w_2'}) = \#_{\mathbf{b}}(\mathbf{w_2'})$.
- $\forall \mathbf{i} \in \{\mathbf{1}, \mathbf{2}\} : \mathbf{w_i} \in \{\mathbf{aw_i'b}, \mathbf{bw_i'a}\}$, je teda zrejmé, že $\#_{\mathbf{a}}(\mathbf{w_i}) = \#_{\mathbf{a}}(\mathbf{w_i'}) + \mathbf{1} \wedge \#_{\mathbf{b}}(\mathbf{w_i}) = \#_{\mathbf{b}}(\mathbf{w_i'}) + \mathbf{1}$.
- $\forall \mathbf{i} \in \{\mathbf{1}, \mathbf{2}\} : \mathbf{w_i} = \varepsilon$, potom $\#_{\mathbf{a}}(\mathbf{w_i}) = \#_{\mathbf{a}}(\mathbf{w_i'}) \wedge \#_{\mathbf{b}}(\mathbf{w_i}) = \#_{\mathbf{b}}(\mathbf{w_i'})$.
- Z toho plynie, že $\#_{\mathbf{a}}(\mathbf{w_1}) = \#_{\mathbf{b}}(\mathbf{w_1}) \wedge \#_{\mathbf{a}}(\mathbf{w_2}) = \#_{\mathbf{b}}(\mathbf{w_2})$ a teda $\mathbf{w_1}, \mathbf{w_2} \in \mathbf{L}$.
- Keď že $\mathbf{w_1}, \mathbf{w_2} \in L$, potom platí taktiež $\mathbf{w} \in \mathbf{L}$, pretože $\#_{\mathbf{a}}(\mathbf{w_1}) + \#_{\mathbf{a}}(\mathbf{w_2}) = \#_{\mathbf{b}}(\mathbf{w_1}) + \#_{\mathbf{b}}(\mathbf{w_2})$.

II. $\mathbf{L} \subseteq \mathbf{L(G)}$

V tomto prípade ukážeme, že každé slovo $\mathbf{w}$ z jazyka $\mathbf{L}$ ($\mathbf{w} \in \mathbf{L}$) je možné vygenerovať v gramatike $\mathbf{G}$ aplikovaním relevantných pravidiel. teda ukážeme že: $\forall \mathbf{w} \in \mathbf{L} : \mathbf{S} \underset{\mathbf{G}}{\Rightarrow}^{*} \mathbf{w}$.

Bázový prípad: $\mathbf{n = 0}$. Najkratšie slovo $\mathbf{w}$, ktoré sa nachádza v jazyku $\mathbf{L}$ a má dĺžku $\mathbf{n = 0}$ je slovo $\mathbf{w} = \varepsilon$. Toto slovo dokážeme v gramatike $\mathbf{G}$ vygenerovať použitím pravidla $\mathbf{S} \to \varepsilon$, čím je zrejmé, že $\mathbf{w} \in \mathbf{L(G)}$. Ukázali sme, že pre všetky slová $\mathbf{w} \in \mathbf{L}$, ktoré majú dĺžku $\mathbf{n = 0}$, platí že $\mathbf{L} \subseteq \mathbf{L(G)}$.

Indukčný predpoklad: $\forall w \in L\ :\ |w| = n - 2 \Rightarrow w \in L(G)$.

Indukčný krok: Ukážeme, že platí: $\forall\ w \in L\ :\ |w| = n \Rightarrow w \in L(G)$.
Postupne budeme uvažovať štyri rôzne prípady, ktoré môžu nastať pri slove $\mathbf{w} : \mathbf{w} \in \mathbf{L} \wedge |\mathbf{w}| \geq \mathbf{2}$. V prípadoch **c)** a **d)** budeme používať symbol $\alpha_{\mathbf{i}}$, ktorého sémantika je nasledovná. Majme slovo $\mathbf{w} = \mathbf{x_1 x_2 \ldots x_n} \in \mathbf{L}$, potom $\alpha_{\mathbf{i}} = \#_{\mathbf{a}}(\mathbf{x_1 \ldots x_i}) - \#_{\mathbf{b}}(\mathbf{x_1 \ldots x_i})$. Možme si všimnúť, že v špecifických hraničných prípadoch je $\alpha_{\mathbf{0}} = \alpha_{\mathbf{n}} = \mathbf{0}$, pretože slovo $\mathbf{w} \in \mathbf{L}$.

a) $\mathbf{w = aw'b},\ |\mathbf{w}| = \mathbf{n},\ |\mathbf{w'}| = \mathbf{n - 2}$

- Podľa indukčného predpokladu $\mathbf{w'} \in \mathbf{L(G)}$ a teda v gramatike $\mathbf{G}$ nutne existuje derivácia $\mathbf{S} \underset{\mathbf{G}}{\Rightarrow}^{*} \mathbf{w'}$.
- V gramatike $\mathbf{G}$ existuje nasledujúca derivácia k vygenerovaniu slova $\mathbf{w}$: $\mathbf{S} \underset{\mathbf{G}}{\Rightarrow} \mathbf{aSb} \underset{\mathbf{G}}{\Rightarrow}^{*} \mathbf{aw'b} = \mathbf{w}$.
- Z toho zjavne plynie, že slovo $\mathbf{w} \in \mathbf{L(G)}$, čím sme dokázali, že indukčný krok pre tento prípad platí.

b) $\mathbf{w = bw'a},\ |\mathbf{w}| = \mathbf{n},\ |\mathbf{w'}| = \mathbf{n - 2}$

- Podľa indukčného predpokladu $\mathbf{w'} \in \mathbf{L(G)}$ a teda v gramatike $\mathbf{G}$ nutne existuje derivácia $\mathbf{S} \underset{\mathbf{G}}{\Rightarrow}^{*} \mathbf{w'}$.
- V gramatike $\mathbf{G}$ existuje nasledujúca derivácia k vygenerovaniu slova $\mathbf{w}$: $\mathbf{S} \underset{\mathbf{G}}{\Rightarrow} \mathbf{bSa} \underset{\mathbf{G}}{\Rightarrow}^{*} \mathbf{bw'a} = \mathbf{w}$.
- Z toho zjavne plynie, že slovo $\mathbf{w} \in \mathbf{L(G)}$, čím sme dokázali, že indukčný krok pre tento prípad platí.

c) $\mathbf{w = axa = w'w''},\ |\mathbf{w}| = \mathbf{n},\ |\mathbf{w'}| + |\mathbf{w''}| = \mathbf{n}$

- Zo slova $\mathbf{w = axa = x_1 x_2 \ldots x_n}$ zjavne plynie, že $\alpha_{\mathbf{1}} = \mathbf{1} \wedge \alpha_{\mathbf{n}} = \mathbf{0}$, keď že $\mathbf{w} \in \mathbf{L}$.
- $(\alpha_n = 0 \wedge x_n = a) \Rightarrow (\alpha_{n-1} = -1)$.
- $(w = axa) \Rightarrow (\exists 1 < i < n - 1 : \alpha_i = 0) \Rightarrow (w = w'w'' \wedge w' = x_1 \ldots x_i \wedge w'' = x_{i+1} \ldots x_n \wedge w', w'' \in L)$.
- Keď že $\mathbf{w'}, \mathbf{w''} \in \mathbf{L}$, podľa indukčného prepokladu v gramatike $\mathbf{G}$ nutne existujú derivácie $\mathbf{S} \underset{\mathbf{G}}{\Rightarrow}^{*} \mathbf{w'} \wedge \mathbf{S} \underset{\mathbf{G}}{\Rightarrow}^{*} \mathbf{w''}$.
- V gramatike $\mathbf{G}$ potom existuje nasledujúca derivácia k vygenerovaniu slova $\mathbf{w}$: $\mathbf{S} \underset{\mathbf{G}}{\Rightarrow} \mathbf{SS} \underset{\mathbf{G}}{\Rightarrow}^{*} \mathbf{w'w''} = \mathbf{w}$.
- Z toho už jasne plynie, že slovo $\mathbf{w} \in \mathbf{L(G)}$, čím sme dokázali, že indukčný krok pre tento prípad platí.

d) $\mathbf{w} = \mathbf{bxb} = \mathbf{w'w''}$, $|\mathbf{w}| = \mathbf{n}$, $|\mathbf{w'}| + |\mathbf{w''}| = \mathbf{n}$

- Zo slova $\mathbf{w} = \mathbf{bxb} = \mathbf{x_1 x_2 \ldots x_n}$ zjavne plynie, že $\alpha_\mathbf{1} = -\mathbf{1} \wedge \alpha_\mathbf{n} = \mathbf{0}$, keďže $\mathbf{w} \in \mathbf{L}$.
- $(\alpha_n = 0 \wedge x_n = b) \Rightarrow (\alpha_{n-1} = 1)$.
- $(w = bxb) \Rightarrow (\exists\, 1 < i < n-1 : \alpha_i = 0) \Rightarrow (w = w'w'' \wedge w' = x_1 \ldots x_i \wedge w'' = x_{i+1} \ldots x_n \wedge w', w'' \in L)$.
- Keďže $\mathbf{w'}, \mathbf{w''} \in \mathbf{L}$, podľa indukčného prepokladu v gramatike $\mathbf{G}$ nutne existujú derivácie $\mathbf{S} \underset{\mathbf{G}}{\Rightarrow}^* \mathbf{w'} \wedge \mathbf{S} \underset{\mathbf{G}}{\Rightarrow}^* \mathbf{w''}$.
- V gramatike $\mathbf{G}$ potom existuje nasledujúca derivácia k vygenerovaniu slova $\mathbf{w}$: $\mathbf{S} \underset{\mathbf{G}}{\Rightarrow} \mathbf{SS} \underset{\mathbf{G}}{\Rightarrow}^* \mathbf{w'w''} = \mathbf{w}$.
- Z toho už jasne plynie, že slovo $\mathbf{w} \in \mathbf{L(G)}$, čím sme dokázali, že indukčný krok pre tento prípad platí.

Na základe preukazaných vzťahov teda platí: $(\mathbf{L} \subseteq \mathbf{L(G)} \wedge \mathbf{L(G)} \subseteq \mathbf{L}) \Rightarrow \mathbf{L} = \mathbf{L(G)}$. □

---

2. ÚLOHA

Zadanie: Uvažujte doprava čítaný jazyk TS M, značený ako $L^P(M)$, ktorý je definovaný ako množina reťazcov, ktoré M príjme v behu, pri ktorom nikdy nepohne hlavou doľava a nikdy neprepíše žiadny symbol na páske za iný. Dokážte, či je probém prázdnosti doprava čítaného jazyka TS M, t.j. či $L^P(M) = \emptyset$, rozhodnuteľný.

- Ak áno, napíšte algoritmus v pseudokóde, ktorý bude daný problém rozhodovať.
- Ak nie, dokážte nerozhodnuteľnosť redukciou z jazyka HP.

Algoritmus

Odstránenie operácií posuvu doľava a operácií prepisu symbolu pod aktuálnou pozicíou hlavy na iný symbol.

Meno: RemoveRulesFromTS

Vstup: TS M $= (Q, \Sigma, \Gamma, \delta, q_0, q_f)$.

Výstup: TS M' $= (Q, \Sigma, \Gamma, \delta', q_0, q_f)$.

1. $\delta' : (Q \setminus \{q_f\}) \times \Gamma \to Q \times (\Gamma \cup \{R\})$, definovaná tak, že:
   $\forall q_1, q_2 \in Q, \forall a \in \Gamma :$
   $((q_2, R) \in \delta'(q_1, a) \Leftrightarrow (q_2, R) \in \delta(q_1, a)) \vee ((q_2, a) \in \delta'(q_1, a) \Leftrightarrow (q_2, a) \in \delta(q_1, a))$
2. **return** $M' = (Q, \Sigma, \Gamma, \delta', q_0, q_f)$

Výpočet Δ-uzáveru

Zavedieme reláciu $\xrightarrow{\Delta}$ v množine $\mathbf{Q}$ takto:

$$\forall q_1, q_2 \in Q : q_1 \xrightarrow{\Delta} q_2 \overset{\text{def}}{\iff} (q_2, \Delta) \in \delta(q_1, \Delta)$$

Následne môžme definovať Δ-uzáver nasledovne:

$$\Delta\text{-uzáver}(q_1) = \{q_2 \in Q \mid q_1 \xrightarrow{\Delta}{}^* q_2\}$$

K výpočtu Δ-uzáveru môžme použiť *Warshalluv algoritmus*, doplníme diagonálu jednotkami a z príslušného riadku matice výslednej relácie vyčítame Δ-uzáver.

Výpočet Δ_R-uzáveru

Zavedieme reláciu $\xrightarrow{\Delta_R}$ v množine $\mathbf{Q}$ takto:

$$\forall q_1, q_2 \in Q : q_1 \xrightarrow{\Delta_R} q_2 \overset{\text{def}}{\iff} ((q_2, \Delta) \in \delta(q_1, \Delta) \vee (q_2, R) \in \delta(q_1, \Delta))$$

Následne môžme definovať Δ_R-uzáver nasledovne:

$$\Delta_R\text{-uzáver}(q_1) = \{q_2 \in Q \mid q_1 \xrightarrow{\Delta_R}{}^{*} q_2\}$$

K výpočtu Δ_R-uzáveru môžme použiť *Warshalluv algoritmus*, doplníme diagonálu jednotkami a z príslušného riadku matice výslednej relácie vyčítame Δ_R-uzáver.

Algoritmus

Transformácia **TS M**, obsahujúceho len operácie posuvu doprava a operácie prepisu symbolu pod aktuálnou pozíciou hlavy na páske na rovnaký symbol na ekvivaletný, **RKA K**.

Meno: TransformTStoRKA

Vstup: TS M $= (Q, \Sigma, \Gamma, \delta, q_0, q_f)$.

Výstup: RKA K $= (Q, \Sigma \cup \{\varepsilon\}, \delta_K, q_0, \{q_f\})$.

1. $\delta_K : Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$, definovaná tak, že:
   $\forall q_1, q_2 \in Q, \forall a \in \Sigma :$
   $q_2 \in \delta_K(q_1, a) \Leftrightarrow ((q_2, R) \in \delta(q_1, a))$
   $q_2 \in \delta_K(q_1, \varepsilon) \Leftrightarrow (((q_2, a) \in \delta(q_1, a)) \wedge (\delta(q_2, a) \neq \emptyset \vee q_2 = q_f))$
   $q_2 \in \delta_K(q_1, \varepsilon) \Leftrightarrow (((q_2, \Delta) \in \delta(q_1, \Delta)) \wedge (q_2 \in \Delta\text{-uzáver}(q_0)))$
   $q_2 \in \delta_K(q_1, \varepsilon) \Leftrightarrow (((q_2, R) \in \delta(q_1, \Delta)) \wedge (q_1 \in \Delta\text{-uzáver}(q_0)))$
   $q_f \in \delta_K(q_1, \varepsilon) \Leftrightarrow (((q_2, \Delta | R) \in \delta(q_1, \Delta)) \wedge (q_f \in \Delta_R\text{-uzáver}(q_1)))$
2. **return** K $= (Q, \Sigma \cup \{\varepsilon\}, \delta_K, q_0, \{q_f\})$

Algoritmus

Overeniu faktu či v danom **TS M** je doprava čítaný jazyk týmto strojom, označovaný ako $\mathbf{L^P(M)}$, prázdny alebo nie, t.j. či $\mathbf{L^P(M)} = \emptyset$.

Vstup: DTS M $= (Q, \Sigma, \Gamma, \delta, q_0, q_f)$.

Výstup: `True` ak $L^P(M) = \emptyset$ alebo `False` ak $L^P(M) \neq \emptyset$.

Metóda:

1. $M_1 = RemoveRulesFromTS(M)$
2. $RKA = TransformTStoRKA(M_1)$
3. $K = TransformRKAtoDKA(RKA)$
4. **return False if** $\exists w \in \Sigma^* : (q_0, w) \vdash^* (q_f, \varepsilon)$ **else True**

Popis algoritmu. Algoritmus je rozdelený do niekoľkých čiastkových algoritmov, ktoré vykonávajú jednotlivé potrebné kroky k rozhodnutiu daného problému prázdnosti doprava čítaného jazyka TS M, teda či $\mathbf{L^P(M)} = \emptyset$. Vstupom algoritmu je *deterministický* **TS M**, ktorý v prvom kroku putuje do funkcie `RemoveRulesFromTS`. Keďže daný problém overuje prázdnosť doprava čítaného jazyka, táto funkcia ponechá v TS len nasledujúce pravidlá z prechodovej funkcie δ **TS M**: operácie posuvu hlavy doprava a operácie prepisu rovnakého symbolu, ktorý sa nachádza pod aktuálnou pozíciou hlavy na páske. Transformovaný **TS**, označme $\mathbf{TS\,M_1}$, reprezentuje tzv. *right moving TS*, ktorý akceptuje regulárny jazyk ($\mathbf{L(M_1)} \in \mathbf{L_3}$). Vďaka tejto vlastnosti nutne existuje nejaký **KA K**, u ktorého bude platiť, že $\mathbf{L(K)} = \mathbf{L(M_1)}$. V nasledujúcom kroku preto prevedieme $\mathbf{TS\,M_1}$ na rožšíreny **KA RKA** pomocou algoritmu `TransformTStoRKA`. K overeniu neprázdnosti daného regulárneho jazyka, však potrebujeme zostrojiť ekvivaletný **DKA** a preto ziskaný **RKA** prevedieme na takýto **DKA** pomocou algoritmu `TransformRKAtoDKA`, ktorý je možné

nájsť v skriptách ako **Algoritmus 3.6.**, na strane 43. V získanom **DKA K** už je triviálne overiť problém neprázdnosti jazyka **L(K)**: $L(K) \neq \emptyset \iff \exists q \in Q : (q \in F \ \wedge$ q je dostupný z $q_0)$. Problém prázdnosti doprava čítaného jazyka **TS M** môže byť rozhodnutý na základe nasledujúceho predikátu: $\exists w \in \Sigma^* : (q_0, w) \vdash^* (q_f, \varepsilon) \Leftrightarrow L^P(M) \neq \emptyset$.

---

3. ÚLOHA

Zadanie: Uvažujte jazyk $L_{42} = \{\langle M \rangle \mid$ TS M zastaví na niektorom vstupe tak, že páska bude obsahovať práve 42 neblankových symbolov $\}$. Dokážte pomocou redukcie, že L_{42} je nerozhodnuteľný. Uveďte ideu dôkazu čiastočnej rozhoduteľnosti L_{42}.

Postup riešenia. Dôkaz nerozhodnuteľnosti jazyka L_{42} budeme realizovať redukciou z *problému zastavenia* (*Halting Problem*) TS. Tento problém sa zaoberá otázkou, či daný TSM pre danú vstupnú vetu w zastaví a zároveň platí, že tento problém je *nerozhodnuteľný*, avšak je čiastočne *rozhodnuteľný*. Budeme teda realizovať redukciu $HP \leq L_{42}$, ktorou ukážeme, že daný problém L_{42} je aspoň "tak ťažký" ako problém zastavenia TS a teda, že problém L_{42} je nerozhodnuteľný.

Redukcia.

Jazyk charakterizujúci problém HP. $HP = \{\langle M \rangle \# \langle w \rangle \mid$ M zastaví pri w$\}$, kde $\langle M \rangle$ je kód TS M a $\langle w \rangle$ je kód w.

Jazyk charakterizujúci problém 42. $L_{42} = \{\langle M \rangle \mid$ TS M zastaví na niektorom vstupe tak, že páska bude obsahovať práve 42 neblankových symbolov$\}$.

Návrh redukcie. $\sigma : HP \to L_{42} = (\langle M \rangle \# \langle w \rangle) \to \langle M_x \rangle = \{0, 1, \#\}^* \to \{0, 1\}^*$.
σ priradí každému reťazcu $x \in \{0, 1, \#\}^*$ reťazec $\langle M_x \rangle$, kde $M_x \in \{0, 1\}^*$ je TS, ktorý pracuje nasledovne:

- $\mathbf{M_x}$ zmaže svoj vstup **w** na vstupnej páske.
- $\mathbf{M_x}$ zapíše na svoju vstupnú pásku reťazec **x**, ktorý má uložený v konečnom stavovom riadení.
- $\mathbf{M_x}$ overí, či **x** má očakávanú štruktúru $\mathbf{x_1 \# x_2}$, kde $\mathbf{x_1}$ je kód **TS** $\mathbf{M_{x_1}}$ a $\mathbf{x_2}$ je kód jeho vstupu $\mathbf{w_{x_2}}$.
- $\mathbf{M_x}$ zmaže obsah vstupnej pásky a odmietne, ak **x** nemá patričnú štruktúru.
- $\mathbf{M_x}$ odsimuluje beh **TS** $\mathbf{M_{x_1}}$ s kódom $\mathbf{x_1}$ na reťazci $\mathbf{w_{x_2}}$ s kódom $\mathbf{x_2}$ s využitím *univerzálneho* TS.
- $\mathbf{M_x}$ zmaže obsah vstupnej pásky, zapíše **42** neblankových symbolov na vstupnú pásku a príjme, ak **univerzálny** TS zastaví, inak $\mathbf{M_x}$ cyklí.

Realizácia redukcie. σ sa dá implementovať *úplným* **TS** $\mathbf{M_\sigma}$, ktorý pre vstup **x** na vstupnej páske vygeneruje kód **TS** $\mathbf{M_x}$, ktorý bude pozostávať z nasledujúcich komponent:

- **TS** $\mathbf{M_c}$, ktorý maže svoju vstupnú pásku a kód tohto **TS** $\mathbf{M_c}$ následne vypíše **TS** $\mathbf{M_\sigma}$.
- **TS** $\mathbf{M_w}$, ktorý zapíše reťazec na svoju vstupnú pásku a kód tohto **TS** $\mathbf{M_w}$ následne vypíše **TS** $\mathbf{M_\sigma}$. Vstupný reťazec v tvare $\mathbf{x = a_1 a_2 \ldots a_n}$ možme zapísať sekvenciou v tvare: $\mathbf{R \mid a_1 \mid R \mid a_2 \mid \ldots}$, kde **R** značí operáciu posuvu hlavy doprava a $\mathbf{a_n}$ značí zápis tohto symbolu na pásku pod aktuálnou pozíciou hlavy.
- **TS** $\mathbf{M_v}$, ktorý overí, čí vstupný reťazec na jeho páske je platnou inštanciou jazyka **HP** a odmietne ak nie (test na členstvo v zafixovanom regulárnom jazyku). **TS** $\mathbf{M_\sigma}$ následne vypíše kód **TS** $\mathbf{M_v}$.
- **Univerzálny TS** $\mathbf{M_s}$, ktorý má na svojej vstupnej páske inštanciu **HP** problému, teda kód $\langle M \rangle \# \langle w \rangle$. Tento TS simuluje beh **TS** s kódom **M** na reťazci s kódom **w**. **TS** $\mathbf{M_\sigma}$ následne vypíše kód **univerzálneho TS** $\mathbf{M_s}$.

Môžme pozorovať, že **TS** $\mathbf{M_s}$, $\mathbf{M_v}$ a $\mathbf{M_c}$ sú konštantné, teda nezávisia na vstupe reťazca **x** z **TS** $\mathbf{M_\sigma}$. Tieto **TS** s popísanou funkcionalitou, ktoré očividne vieme zostrojiť, možme vďaka tejto vlastnosti zkonštruovať vopred a ako bolo povedané vyššie, **TS** $\mathbf{M_\sigma}$ následne vypíše kódy týchto **TS** na svoju pásku. V poslednom kroku sa vygeneruje kód pre **TS** $\mathbf{M_w}$, ktorý zapíše na pásku daný reťazec **x**. **TS** $\mathbf{M_\sigma}$ zaistí sekvenčné predávanie riadenia medzi uvedenými komponentami.

Jazyk TS $\mathbf{M_x}$.

(a) $\mathbf{L}(\mathbf{M_x}) = \emptyset \Longleftrightarrow$ ((**x** nie je validná inštancia jazyka **HP**) $\vee$ (**x** je validná inštancia jazyka **HP**, teda $\mathbf{x} = \mathbf{x_1}\#\mathbf{x_2}$, kde $\mathbf{x_1}$ je kód **TS** a $\mathbf{x_2}$ je kód vstupu, avšak **TS** s kódom $\mathbf{x_1}$ nezastaví na vstupe s kódom $\mathbf{x_2}$)).

(b) $\mathbf{L}(\mathbf{M_x}) = \Sigma^* \Longleftrightarrow$ (**x** je validná inštancia jazyka **HP**, teda $\mathbf{x} = \mathbf{x_1}\#\mathbf{x_2}$, kde $\mathbf{x_1}$ je kód **TS** a $\mathbf{x_2}$ je kód vstupu a **TS** s kódom $\mathbf{x_1}$ zastaví na vstupe s kódom $\mathbf{x_2}$).

Korektnosť redukcie σ. Ukážeme, že σ zachovává členstvo v jazyku:

$$\forall x \in \{0,1,\#\}^* : \sigma(x) = \langle M_x \rangle \in L_{42} \Leftrightarrow L(M_x) = \Sigma^* \Leftrightarrow \text{ (x je validná inštancia jazyka HP, teda } x = x_1\#x_2$$
$$\text{kde } x_1 \text{ je kód TS a } x_2 \text{ je kód vstupu a TS s kódom } x_1 \text{ zastaví na vstupe s kódom } x_2) \Leftrightarrow x \in HP.$$

Dokázali sme teda, že existuje redukcia z problému **HP**, ktorý je nerozhodnuteľný, na problém $\mathbf{L_{42}}$. Z toho teda jasne plynie, že daný problém $\mathbf{L_{42}}$ je taktiež nutne nerozhodnuteľný.

Idea čiastočnej nerozhodnuteľnosti

- Môžme zostrojiť **TS** $\mathbf{M_{SIM}}$, ktorý na svojej vstupnej páske simuluje výpočet vstupného **TS** $\mathbf{M_{42}}$, ktorý je inštanciou jazyka $\mathbf{L_{42}}$, pre jednotlivé možné vstupné reťazce.
- **TS** $\mathbf{M_{SIM}}$ nemôže len systematicky vygenerovať jednotlivé vstupné reťazce, v napríklad lexikografickom usporiadaní, a na každom z nich spustiť neobmedzenú simuláciu **TS** $\mathbf{M_{42}}$. V takomto prípade by mohlo dôjsť k zacykleniu **TS** $\mathbf{M_{42}}$ a následne by sa celý výpočet zacyklil bez toho, aby bolo garantované nájdenie reťazca, na ktorom **TS** $\mathbf{M_{42}}$ zastaví.
- Aby sme sa vyhli možnému vzniku tohto problému, tak **TS** $\mathbf{M_{SIM}}$ na svojej vstupnej páske postupne rozbieha viaceré simulácie **TS** $\mathbf{M_{42}}$ pre jednotlivé možné vstupné reťazce.
- **TS** $\mathbf{M_{SIM}}$ si u každej rozbehnutej simulácie pamätá navyše ešte aj vnútorný stav riadenia **TS** $\mathbf{M_{42}}$ pri spracovaní daného vstupu, napríklad uložením na páske za príslušným vstupom.
- Jednotlivé rozbehnuté simulácie má teda **TS** $\mathbf{M_{SIM}}$ uložené na vstupnej páske a sú oddelené vhodným oddeľovačom. V prípade nedostatku miesta im **TS** $\mathbf{M_{SIM}}$ môže zväčšiť potrebný priestor realizovaním relevantných akcií posunov.
- Samotná simulácia prebieha tak, že **TS** $\mathbf{M_{SIM}}$ vždy prevedie **1** krok na každej rozbehnutej simulácií. V prípade, že jedna z týchto rozbehnutých simulácií povedie k prijatiu vstupného reťazca, tak **TS** $\mathbf{M_{SIM}}$ taktiež príjme. Inak **TS** $\mathbf{M_{SIM}}$ rozbehne ďalšiu simuláciu pre nasledujúci vygenerovaný vstupný reťazec a tento proces znova opakuje.
- Je zrejmé, že **TS** $\mathbf{M_{SIM}}$ príjme, ak $\mathbf{L}(\mathbf{M_{42}}) \neq \emptyset$, v opačnom prípade nikdy neskončí.