

JAO praca domowa

Bartosz Kucypera, bk439964

30 kwietnia 2023

Zadanie 2.1

$$L_{\exists} = \{ab^{n_1}ab^{n_2} \dots ab^{n_k}a \in \{a,b\}^* \mid \exists i \in \mathbb{N}. 1 \leq i \leq k \wedge n_i = k\}$$

Definicja gramatyki

Niech \mathcal{G} będzie gramatyką bezkontekstową opisującą L_{\exists} . $\mathcal{G} = (T, N, P, S)$ gdzie:

T - zbiór symboli terminalnych, $T = \{a, b\}$

N - zbiór symboli nieterminalnych, $N = \{S, L_0, R_0, L_1, R_1\}$

P - zbiór reguł, zdefiniowany poniżej

S - symbol początkowy

Do P należą następujące reguły:

$$S \rightarrow L_0 b R_0 \mid a$$

$$L_0 \rightarrow a L_1 b \mid a$$

$$R_0 \rightarrow b R_1 a \mid a$$

$$L_1 \rightarrow a L_1 b \mid b L_1 \mid a$$

$$R_1 \rightarrow b R_1 a \mid R_1 b \mid a$$

Inkluzja $L_{\exists} \subseteq L(\mathcal{G})$

Niech w będzie dowolnym słowem z L_{\exists} . Pokażemy, że potrafimy zwinąć w do S korzystając z odwrotnych przejść gramatyki \mathcal{G} , czyli, że $w \in L(\mathcal{G})$.

Niech k będzie stałą z definicji w , oraz niech a_l, a_r będą zwykłymi literami a z w , wyróżnionymi dla naszej wygody.

$$w = \text{prefix} \cdot a_l b^k a_r \cdot \text{suffix}$$

(pomijając trywialny przypadek dla $w = a$).

Pokażmy, jak zwinąć prefix , suffix związa się analogicznie.

Jeśli na początku $|\text{prefix}| = 0$, to przekształcamy $a_l \rightarrow L_0$ i kończymy.

Jeśli nie to przekształcamy $a_l \rightarrow L_1$ i wykonujemy następujący algorytm:

Jeśli $\text{prefix} = a$, to przekształcamy $a L_1 b \rightarrow L_0$ i kończymy.

Jeśli $\text{prefix} = \text{prefix}' \cdot a$ to przekształcamy $a L_1 b \rightarrow L_1$, czyli prefix traci ostatnią literkę i "zjadamy" jedno z b pomiędzy a_l i a_r .

Jeśli $\text{prefix} = \text{prefix}' \cdot b$ to przekształcamy $b L_1 \rightarrow L_1$, czyli prefix po prostu traci ostatnią literkę.

Algorytm zawsze się skończy, bo prefix jest skończony i zaczyna się od literki a .

Analogicznie postępujemy dla suffix .

Pozbyliśmy się już suffix i prefix . Teraz zauważmy, że początkowo w w mieliśmy $k+1$ liter a (z definicji w), dwie wyróżniliśmy (a_l i a_r) a pozostałe $k-1$ zjadł nasz algorytm. Skoro każde usunięcie a wiązało się też ze "zjedzeniem" jednej z liter b pomiędzy a_l i a_r których na początku było k , to została nam tylko jedna.

Zwineliśmy, więc w do $L_0 b R_0$. Wystarczy wykonać teraz ostatni krok i przekształcić $L_0 b R_0$ do S .

Skoro umiemy ciągiem operacji odwrotnych zwinąć w do S to niewątpliwie potrafimy je też wygenerować za pomocą gramatyki \mathcal{G} . Z dowolności wyboru w wnioskujemy, że $L_{\exists} \subseteq L(\mathcal{G})$.

Inkluzja $L(\mathcal{G}) \subseteq L_{\exists}$

Pokażmy indukcyjnie, że jesteśmy w stanie generować jedynie słowa należące do L_{\exists} .

Przypadek bazowy: $L_0 b R_0$ (jeśli $S \rightarrow a$ to oczywiście $a \in L_{\exists}$ ok),
kończymy dalsze generowanie, zamieniamy $L_0 \rightarrow a, R_0 \rightarrow a, aba \in L_{\exists}$ ok.

Krok indukcyjny:

Założmy, że wygenerowaliśmy poprawne wyrażenie postaci:

$$prefix \cdot L_x b^k R_y \cdot suffix$$

* L_x to L_0 lub L_1 , R_y to R_0 lub R_1 .

*poprawne - do $prefix \cdot suffix$ należy dokładnie $k-1$ liter a i po zamianie L_x i R_y na a , słowo będzie należało do L_{\exists} .

Możemy wykorzystać jedno z dwóch przekształceń (oprócz tych kończących, zamieniających w a):

$$L_x \rightarrow aL_1b \text{ albo } R_y \rightarrow bR_1a.$$

Pokażmy dla L_x , dla R_y dowód jest analogiczny. Przekształcamy:

$$prefix \cdot L_x b^k R_y \cdot suffix \rightarrow prefix \cdot aL_1 b^{k+1} R_y \cdot suffix$$

Nie dodaliśmy do początku żadnej literki b , wcześniejsze słowo mało $k+1$ liter a (razem z L_x i R_y), nowe ma więc $k+2$, ale ma też blok liter b długości $k+1$, czyli wszystko się zgadza. Nowe wyrażenie też jest poprawne* i też jest postaci:

$$prefix' \cdot L_x b^{k'} R_x \cdot suffix$$

dla $prefix' = prefix \cdot a$ i $k' = k+1$.

Na mocy indukcji matematycznej, gramatyką \mathcal{G} jesteśmy w stanie wygenerować tylko słowa należące do L_{\exists} , czyli zachodzi $L(\mathcal{G}) \subseteq L_{\exists}$.

Konkluzja

Skoro

$$L_{\exists} \subseteq L(\mathcal{G}) \wedge L(\mathcal{G}) \subseteq L_{\exists}$$

to zachodzi

$$L_{\exists} = L(\mathcal{G})$$

czyli gramatyka \mathcal{G} faktycznie opisuje L_{\exists} . Gramatyka \mathcal{G} jest bezkontekstowa, więc oczywiście L_{\exists} jest językiem bezkontekstowym.