Teoretyczne podstawy informatyki
zadanie 32

Jarosław Socha

27 kwietnia 2024

1 Treść zadania

Zadanie 32
Pokaż, że dla gramatyki bezkontekstowej G nie jest rozstrzygalne, czy $L(G) = \Sigma^*$.

2 Rozwiązanie

2.1 Problem przekroju

Weźmy instancję problemu Posta nad alfabetem Σ.

$$A = (a_1, ..., a_k) \qquad B = (b_1, ..., b_j) \qquad , \qquad a_i, b_i \in \Sigma^+$$

Następnie stworzymy dwie gramatyki bezkontekstowe nad alfabetem Σ', za pomocą których rozwiążemy problem Posta.

$$\Sigma' = \Sigma \dot{\cup} \{1, 2, ..., k\}$$

$$G_A : S_A \to a_i S_A i \mid a_i i \qquad 1 \leqslant i \leqslant k$$

$$G_B : S_B \to b_i S_B i \mid b_i i \qquad 1 \leqslant i \leqslant k$$

Gramatyka G_A wygeneruje słowa postaci $a_{i_1} a_{i_2} ... a_{i_j} i_j i_{j-1} ... i_1$, gramatyka G_B wygeneruje słowa analogiczne.

Jeżeli słowo w jest wyprowadzalne w obu gramatykach, to jest ono rozwiązaniem problemu Posta. Innymi słowy, rozwiązanie problemu Posta istnieje wtedy i tylko wtedy, gdy $L(G_A) \cap L(G_B) \neq \emptyset$.

Zatem problem "Czy dwie gramatyki bezkontekstowe G_A i G_B mają słowo wspólne?" jest nierozstrzygalny, ponieważ rozwiązywałby nierozstrzygalny problem Posta.

2.2 Deterministyczny automat ze stosem

Teraz stworzymy deterministyczny automat ze stosem (DPDA) rozpoznający gramatykę G_A. Działanie automatu będzie wyglądało następująco:

- dopóki widzimy symbole z Σ, to wrzucamy je na stos
- gdy zobaczymy symbol $i_j \in \{1, ..., k\}$
  - za pomocą stanów sprawdzamy, czy na szczycie stosu mamy słowo $a_{i_j}^R$. Jeśli tak, to przechodzimy dalej, do i_{j-1}
  - jeśli nie, to przechodzimy do stanu śmietnikowego
  - do stanu śmietnikowego przechodzimy także gdy na tym etapie zobaczymy symbol z Σ, bądź opróżniliśmy stos mając wciąż symbole w słowie do przeczytania
- jeśli stos jest pusty i nie jesteśmy w stanie śmietnikowym to akceptujemy, w przeciwnym przypadku odrzucamy słowo.

Automat ma skończoną liczbę stanów (związane ze skończonym ciągiem skończonych długości a_i) oraz jest deterministyczny, każdy krok jest jednoznaczny.

Determinizm automatu $PDA(G_A)$ pozwala nam stworzyć $PDA^C(G_A)$, wyznaczający dopełnienie języka $L(G_A)$, wystarczy zamienić wynik zwracany przez automat (tak na nie i nie na tak). Dowód dla $PDA(G_B)$ i $PDA^C(G_B)$ przebiega analogicznie.

To, że języki $L(G_A)^C$ i $L(G_B)^C$ są rozpoznawalne przez automaty ze stosem oznacza, że istnieją gramatyki bezkontekstowe, które generują te języki.

2.3 Problem ostateczny

Weźmy gramatykę G, taką że $L(G) = L(G_A)^C \cup L(G_B)^C$. Obydwie składniowe gramatyki są bezkontekstowe, a gramatyki bezkontekstowe są zamknięte na sumę, więc gramatyka G też jest bezkontekstowa. Znając odpowiedź na pytanie $L(G) \stackrel{?}{=} \Sigma^*$ Dowiadujemy się:

- Jeśli $L(G) = \Sigma^*$

$$L(G_A)^C \cap L(G_B)^C = \Sigma^*$$

$$L(G_A) \cup L(G_B) = \emptyset$$

- Jeśli $L(G) \neq \Sigma^*$

$$L(G_A)^C \cap L(G_B)^C \neq \Sigma^*$$

$$L(G_A) \cup L(G_B) \neq \emptyset$$

Czyli znając odpowiedź na to pytanie jesteśmy w stanie stwierdzić czy dwie gramatyki generują wspólne słowo, co jest problemem nierozstrzygalnym, zatem pytanie $L(G) \stackrel{?}{=} \Sigma^*$ dla bezkontekstowych G jest nierozstrzygalne.