Zadania z gwiazdką - seria III, zadania 2 i 3
szkic rozwiązań

2. Pokaż, że następujący problem jest nierozstrzygalny: dana gramatyka bezkontekstowa G, rozstrzygnij, czy język $L(G)$ jest akceptowany przez pewien deterministyczny automat ze stosem.
Wskazówka: Użyj faktu, że problem uniwersalności języka bezkontekstowego jest nierozstrzygalny.

Szkic rozwiązania Wykonamy redukcję z problemu uniwersalności dla języków bezkontekstowych. Niech $L \subseteq \Sigma^*$ będzie bezkontekstowy, a $K \subseteq \Sigma^*$ będzie językiem bezkontekstowym, który nie jest rozpoznawalny przez deterministyczny automat ze stosem (powiedzmy $K = \overline{\{ww \mid w \in \Sigma^*\}}$). Niech $\# \notin \Sigma$ i niech

$$L' = (L \cdot \{\#\} \cdot \Sigma^*) \cup (\Sigma^* \cdot \{\#\} \cdot K).$$

Pokażemy, że $L = \Sigma^*$ wtedy i tylko wtedy gdy L' jest rozpoznawalny przez deterministyczny automat ze stosem, co zakończy redukcję. Jeśli $L = \Sigma^*$, to $L' = \Sigma^* \cdot \{\#\} \cdot \Sigma^*$ i istotnie jest rozpoznawany przez deterministyczny automat ze stosem. Jeśli natomiast $L \neq \Sigma^*$, to niech $w \notin L$. Gdyby L' był rozpoznawany przez deterministyczny automat ze stosem, to również język $L'' = \{w\#\} \cdot K = L' \cap (\{w\#\} \cdot \Sigma^*)$ byłby tej postaci. Wtedy jednak również K byłby tej postaci (nietrudno zmodyfikować nieco automat dla L''), a to byłaby sprzeczność z założeniem na temat K. Zatem istotnie jeśli $L \neq \Sigma^*$, to L' nie jest rozpoznawalny przez deterministyczny automat ze stosem, co kończy redukcję.

3. Niech Σ to skończony alfabet oraz $T \subseteq \Sigma^* \times \Sigma^*$ będzie zbiorem par słów. *Przepisanie* zgodne z T to ruch postaci

- $uv_1w \mapsto uv_2w$, gdzie $u, w \in \Sigma^*$ oraz $(v_1, v_2) \in T$; lub
- $uav \mapsto uv$, gdzie $u, v \in \Sigma^*$, natomiast $a \in \Sigma$.

Rozstrzygnij, czy następujący problem jest rozstrzygalny: dane T i dwa słowa $w_{\text{init}}, w_{\text{fin}} \in \Sigma^*$, czy startując z w_{init} da się otrzymać w_{fin} po wykonaniu pewnego skończonego ciągu przepisań zgodnych z T?

Szkic rozwiązania Niech $v, w \in \Sigma^*$. Mówimy, że v jest *podciągiem* w jeśli $v = a_1 \cdot \ldots \cdot a_k$, gdzie $a_i \in \Sigma$, natomiast $w = w_0a_1w_1 \cdot \ldots \cdot w_{k-1}a_kw_k$, gdzie $w_i \in \Sigma^*$. Piszemy wtedy $v \preceq w$. Skorzystamy z lematu Higmana, który implikuje, że dowolny $S \subseteq \Sigma^*$, który jest antyłańcuchem w porządku $\preceq$ (czyli każde jego dwa elementy są nieporównywalny) jest skończony. Powiemy, że zbiór $U \subseteq \Sigma^*$ jest *zamknięty w górę* jeśli dla każdego $u \in U$ jeśli $u \preceq u'$, to również $u' \in U$. Zauważmy, że dla zbioru U zamkniętego w górę zbiór elementów minimalnych U w porządku $\preceq$, czyli $\min(U) = \{u \in U \mid \neg\exists_{v \in U, v \neq u} v \preceq u\}$ jest skończony (z lematu Higmana) i opisuje całkowicie zbiór U.

Powiedzmy, że $uv_1w \mapsto_T uv_2w$ gdzie $u, v_1, v_2, w \in \Sigma^*$ jeśli $(v_1, v_2) \in T$, takie przepisanie nazwiemy *tranzycją*. Powiedzmy też, że $w_0 \mapsto_T^* w_k$ jeśli istnieje ciąg

tranzycji od w_0 do w_k, czyli jeśli istnieją w_i dla $i \in \{1, \ldots, k-1\}$ takie, że $w_j \mapsto_T w_{j+1}$ dla $j \in \{0, \ldots, k-1\}$. Zauważmy, że problem w zadaniu jest równoważny pytaniu, czy istnieje słowo w takie, że $w_{\text{init}} \mapsto_T^* w$ oraz $w_{\text{fin}} \preceq w$, nazwijmy je słowem *pokrywającym*.

Zaprojektujemy dwa algorytmy, które niekoniecznie będą się kończyć, ale:

- pierwszy, zwany *pozytywnym*, będzie się kończył o ile słowo pokrywające istnieje i zwracał odpowiedź „TAK";
- drugi, zwany *negatywnym*, będzie się kończył o ile słowo pokrywające nie istnieje i zwracał odpowiedź „NIE".

Zauważmy, że jeśli zaprojektujemy takie dwa algorytmy, to możemy również zaprojektować poprawny algorytm rozwiązujący problem z treści zdania. Mianowicie zapuścimy oba algorytmy równocześnie. W pewnym momencie zakończy się któryś z nich zwracając poprawną odpowiedź, wtedy przerywamy działanie tego drugiego i zwracamy odpowiedź od tego, który się zakończył.

Algorytm pozytywny jest bardzo prosty. Po prostu przeszukuje coraz dłuższe ciągi tranzycji i sprawdza, czy może doszedł do słowa pokrywającego. Jeśli istnieje takie słowo, to w końcu jest znajdzie i odpowie "TAK".

Algorytm negatywny jest bardziej skomplikowany. Oznaczmy przez U zbiór wszystkich słów, do których można dojść ciągiem tranzycji z w_{init}. Oznaczmy $V = \{v \mid \exists_{u \in U} v \preceq u\}$. Zauważmy, że słowo pokrywające nie istnieje wtedy i tylko wtedy, gdy V nie zawiera słowa w_{fin}. Co więcej zbiór V ma następujące własności:

1. jeśli $v \in V$ oraz $v \mapsto_T v'$, to $v' \in V$;
2. $\Sigma^* \setminus V$ jest zamknięty w górę.

Zauważmy więc, że słowo pokrywające nie istnieje wtedy i tylko wtedy, gdy istnieje zbiór V spełniający warunki:

1. $w_{\text{init}} \in V$;
2. jeśli $v \in V$ oraz $v \mapsto_T v'$, to $v' \in V$;
3. $\Sigma^* \setminus V$ jest zamknięty w górę.

Zatem drugi algorytm przegląda coraz większe skończone zbiory S (kandydaci na $\min(\Sigma^* \setminus V)$) i dla każdego sprawdza, czy może $\Sigma^* \setminus \{u \mid \exists_{s \in S} s \preceq u\}$ nie spełnia powyższych trzech warunków. Można pokazać, że każdy z nich da się rozstrzygnąć, co kończy opis drugiego algorytmu i tym samym całego rozwiązania.