Języki, automaty i obliczenia
egzamin – przykładowe rozwiązania zadań
28 czerwca 2017

Rozwiązanie zadania 1. Tak, język $\mathrm{mix}(L)$ jest regularny. Rozważmy dowolny (być może niedeterministyczny) automat A rozpoznający L. Skonstruujemy automat B rozpoznający język $\mathrm{mix}(L)$, opierając się na równoważnej definicji:

$$\mathrm{mix}(L) \;=\; \{\mathrm{rev}(u_1)\ldots\mathrm{rev}(u_k) \mid k \in \mathbb{N},\ u_1,\ldots,u_k \in \Sigma^*,\ u_1\ldots u_k \in L\}.$$

Intuicyjna idea konstrukcji opisanej poniżej jest taka, że każdy bieg B jest otrzymany przez pocięcie biegu automatu A na pewną liczbę k „kawałków" (rozpoznających słowa $u_1, \ldots, u_k$), a następnie odwrócenie każdego z nich. Zrealizujmy tę ideę, ustalając jako stany automatu B zbiór wszystkich trójek (q_1, q_2, q_3) stanów automatu A: q_1 to stan na początku bieżącego kawałka, q_2 to stan na jego końcu, a q_3 to bieżący stan automatu wewnątrz kawałka. Stany początkowe automatu B to wszystkie trójki (q_1, q_2, q_3) gdzie q_1 jest stanem początkowym w A oraz $q_3 = q_2$. Stany akceptujące automatu B to wszystkie trójki (q_1, q_2, q_3) gdzie q_2 jest stanem akceptującym w A oraz $q_3 = q_1$. Dla każdego przejścia $q \xrightarrow{a} q'$ w automacie A, automat B ma przejście

$$(q_1, q_2, q') \xrightarrow{a} (q_1, q_2, q).$$

Przejścia te realizują kawałek biegu automatu A, wstecz. Ponadto, dla każdej trójki postaci (q_1, q_2, q_1), i dla dowolnego stanu q_3, automat B ma przejście

$$(q_1, q_2, q_1) \xrightarrow{\varepsilon} (q_2, q_3, q_3).$$

Przejścia te realizują „przeskoki" z jednego kawałka do następnego. Z konstrukcji automatu B wnioskujemy, że $L(B) = \mathrm{mix}(L)$.

Rozwiązanie zadania 2. Niech G będzie gramatyką liniową, a N sumą długości wszystkich prawych stron produkcji. Rozważmy dowolne słowo $w \in L(G)$ długości większej niż N, oraz dowolne jego wyprowadzenie. Zauważmy, że wyprowadzenia w gramatyce liniowej mają szczególną postać, mianowicie po każdym kroku wyprowadzenia słowo zawiera co najwyżej jeden nieterminal:

$$X_0 \to u_1X_1v_1 \to u_1u_2X_2v_2v_1 \to \ldots \to u_1u_2\ldots u_nX_nv_n\ldots v_2v_1 \to \\ u_1u_2\ldots u_nuv_n\ldots v_2v_1 \;=\; w.$$

(X_0 jest nieterminalem startowym.) Skoro słowo w jest dłuższe niż N, któraś z produkcji gramatyki G musiała zostać użyta więcej niż raz w wyprowadzeniu. Dla naszych potrzeb wystarczy, że któryś z nieterminali musiał się powtórzyć; niech $X_i = X_j$ będzie pierwszym powtórzeniem nieterminala. Dokładniej, niech $X_i = X_j$ będzie pierwszym *wydłużającym* powtórzeniem nieterminala, czyli takim, że któreś ze słów $u_{i+1} \dots u_j$ i $v_j \dots v_{i+1}$ jest niepuste. Zdefiniujmy podział słowa w na 5 słów:

$$\begin{aligned}
x &= u_1 \dots u_i \\
s &= u_{i+1} \dots u_j \\
y &= u_{j+1} \dots u_n u v_n \dots v_{j+1} \\
t &= v_j \dots v_{i+1} \\
z &= v_i \dots v_1
\end{aligned}$$

Mamy $w = xsytz$ oraz $st \neq \varepsilon$, ponieważ wybraliśmy wydłużające powtórzenie nieterminala. Skoro wybraliśmy pierwsze wydłużające powtórzenie, suma długości słów xs i tz jest równa co najwyżej N. Wreszcie, $xs^k yt^k z \in L$ dla każdego $k \in \mathbb{N}$.

Nie każdy język bezkontekstowy jest generowany przez gramatykę liniową. Rozważmy język L wszystkich słów $w \in a^+b^+a^+$, w których występuje tyle samo liter a co b. Język L jest bezkontekstowy. Pokażemy teraz, że L nie spełnia własności z pierwszej części zadania. Skoro spełniają ją wszystkie języki generowane gramatyki liniowe, język L nie jest rozpoznawany przez gramatykę liniową.

Dla dowolnie wybranego N, rozważmy słowo

$$a^{N+1} b^{2N+2} a^{N+1} \in L.$$

Rozważmy dowolny podział $w = xsytz$ taki, że słowa x, z są długości co najwyżej N oraz $st \neq \varepsilon$. Mamy $x \in a^*$ oraz $z \in a^*$. Jeśli słowa s, t zawierają tylko litery a, to słowo xs^2yt^2z nie należy do L ponieważ wzrosła liczba liter a a nie wzrosła liczba liter b. Jeśli któreś ze słów s, t zawiera choć jedno b, to zawiera też przynajmniej jedno a, więc słowo xs^2yt^2z znów nie należy do L ponieważ wzrosła liczba alternacji między literami a i b.

Rozwiązanie zadania 3. Problem jest nierozstrzygalny, co pokażemy za pomocą redukcji z innego problemu nierozstrzygalnego, mianowicie z problemu niepustości przecięcia języków bezkontekstowych, w którym dla danych gramatyk G_1, G_2 pytamy, czy przecięcie $L(G_1) \cap L(G_2)$ jest niepuste.

Zdefiniujmy funkcję przekształcającą instancję tego ostatniego problemu, czyli gramatyki G_1, G_2, na instancję naszego problemu, czyli gramatykę G generującą język[1]

$$L(G) \;=\; L(G_1)\#L(G_1)\#L(G_2)\#L(G_2)\#$$

(litera $\#$ jest świeża, tzn. nie występuje w G_1 i G_2). Funkcja ta jest łatwo obliczalna: przy założeniu (bez utraty ogólności) że gramatyki G_1, G_2 mają rozłączne zbiory symboli nieterminalnych, gramatyka G powstaje jako suma gramatyk G_1 i G_2, plus nowy symbol startowy S z jedną produkcją

$$S \;\to\; S_1\#S_1\#S_2\#S_2\#$$

(gdzie S_1 i S_2 to symbole startowe gramatyk G_1 i G_2).

Dla poprawności redukcji powinniśmy pokazać, że $L(G_1) \cap L(G_2)$ jest niepusty wtedy, i tylko wtedy gdy $L(G)$ zawiera słowo postaci $wwww$. W jedną stronę: jeśli $v \in L(G_1) \cap L(G_2)$, to $v\#v\#v\#v\# \in L(G)$. W drugą stronę: jeśli $L(G)$ zawiera słowo postaci $wwww$, to w kończy się literą $\#$ (bo $\#$ występuje dokładnie 4 razy w każdym słowie generowanym przez G), czyli $w = v\#$, i v należy tak do $L(G_1)$ jak i $L(G_2)$.

Rozwiązanie zadania 4. Problem należy do NP: wystarczy zgadnąć właściwe wartościowanie zmiennych a następnie sprawdzić, że spełnia ono dokładnie 2 z formuł.

Dla pokazania NP-trudności, zdefiniujemy redukcję z problemu spełnialności formuł boolowskich (SAT) do problemu z zadania. Niech $\top$ będzie dowolną formułą boolowską spełnioną przez wszystkie wartościowania, np.

$$\top \;\equiv\; x \vee \neg x.$$

Podobnie, niech $\bot$ będzie dowolną formułą boolowską nie spełnioną przez żadne wartościowanie, np.

$$\bot \;\equiv\; x \wedge \neg x.$$

Zdefiniujmy funkcję, która przekształca instancję problemu SAT, czyli formułę boolowską ϕ, na instancję naszego problemu złożoną z następujących 3 formuł:

$$\top,\; \bot,\; \phi. \tag{1}$$

[1] Równie dobrze można tu użyć innych kombinacji języków $L(G_1)$ i $L(G_2)$, np. $L(G_1)\#L(G_2)\#L(G_2)\#L(G_2)\#$ albo $L(G_1)\#L(G_2)\#L(G_1)\#L(G_2)\#$.

Funkcja ta jest oczywiście obliczalna w czasie wielomianowym. Dla poprawności redukcji należy pokazać, że ϕ jest spełnialna wtedy, i tylko wtedy gdy istnieje wartościowanie spełniające dokładnie 2 z formuł (1). W jedną stronę: jeśli jakieś wartościowanie spełnia ϕ, to to samo wartościowanie spełnia też $\top$, ale oczywiście nie $\perp$, więc spełnia ono dokładnie 2 z formuł (1). W przeciwną stronę: jeśli jakieś wartościowanie spełnia dokładnie 2 z formuł (1), to z pewnością są to formuły $\top$ i ϕ, więc ϕ jest spełnialna.