Zadanie 1. Czy prawdziwa jest następująca implikacja? Jeśli $L \subseteq A^*$ jest językiem regularnym, to regularnym językiem jest też

$$L' = \{vw : vuw \in L \text{ dla pewnego } u \in A^* \text{ takiego, że } |u| = |v| + |w|\}$$

Rozwiązanie. Niech

$$\mathcal{A} = (A, Q, q_I, F, \delta)$$

automat, który rozpoznaje L. Dla stanów $p, q \in Q$, rozważmy trzy języki.

1. Rozważmy najpierw język:

   $$L_{p,q} = \{w \in A^* : \text{ istnieje bieg po } w \text{ zaczynający w } p \text{ i kończący w } q\}$$

   Jest to oczywiście język regularny, rozpoznaje go automat $\mathcal{A}$ ze zmienionymi stanami początkowymi i końcowymi.

2. Rozważmy następnie język

   $$K_{p,q} = \{w \in A^* : u \in L_{p,q} \text{ dla pewnego } u \text{ tej samej długości co } w\}$$

   Jest to też język regularny, rozpoznaje go ten sam automat co $L_{p,q}$, tyle że każde przejście umożliwiamy po każdej literze.

3. Rozważmy wreszcie język

   $$M_{p,q} = \{w \in A^* : wu \in L_{p,q} \text{ dla pewnego } u \text{ tej samej długości co } w\}$$

   Jest to język regularny, jako suma konkatenacji języków regularnych

   $$M_{p,q} = \bigcup_{r \in Q} L_{p,r} \cap K_{r,q}$$

   Na tych samych zasadach, regularny jest

   $$M'_{p,q} = \{w \in A^* : uw \in L_{p,q} \text{ dla pewnego } u \text{ tej samej długości co } w\}$$

Sam język L' z treści zadania jest regularny, bo przedstawia się jako

$$L' = \bigcup_{p \in I, q \in Q, r \in F} M_{p,q} \cdot M'_{q,r}$$

Zadanie 2. Niech L będzie zbiorem wyrażen regularnych nad alfabetem $\{a\}$, które generują przynajmniej jedno słowo długości parzystej. A więc alfabet dla języka L to

$$(\quad) \quad \emptyset \quad \cdot \quad a \quad * \quad +$$

Aby nie wdawać się w priorytety operacji, załóżmy, że interesują nas wyrażenia, gdzie każde podwyrażenie jest otoczone nawiasami, jak np.

$$((((a) \cdot (a))^*) + (a))$$

Czy język L jest bezkontekstowy?

Rozwiązanie. Tak, język jest bezkontekstowy. Poniżej opisujemy gramatykę.

Dla podzbioru $X \subseteq \{0,1\}$, będziemy mieli nieterminal N_X. Niezmiennik będzie następujący:

- $N_{0,1}$ opisuje wyrażenia regularne generujące słowa zarówno parzystej jak i nieparzystej długości
- N_0 opisuje wyrażenia regularne generujące tylko słowa parzystej długości
- N_1 opisuje wyrażenia regularne generujące tylko słowa nieparzystej długości
- $N_\emptyset$ opisuje wyrażenia regularne nie generujące żadnych słów

Mamy wreszcie nieterminal startowy S, o regule

$$S \quad \to \quad N_{\{0\}} \quad | \quad N_{\{0,1\}}$$

Podstawowe reguły to

$$\begin{aligned} N_\emptyset &\to \emptyset \\ N_{\{1\}} &\to (a) \end{aligned}$$

Rozważmy teraz operację sumy w wyrażeniach regularnych. Jeśli $X, Y, Z \subseteq \{0,1\}$ są zbiorami takimi, że $X = Y \cup Z$, to gramatyka zawiera regułę

$$N_X \to (N_Y + N_Z)$$

Rozważmy teraz operację konkatenacji w wyrażeniach regularnych. Jeśli $X, Y, Z \subseteq \{0,1\}$ są zbiorami takimi, że

$$X = \{i + j \bmod 2 : i \in Y, j \in Z\},$$

to gramatyka zawiera regułę

$$N_X \to (N_Y \cdot N_Z)$$

Rozważmy wreszcie gwiazdkę. Dla niej mamy reguły:

$$\begin{aligned} N_{\{0\}} &\rightarrow (N_{\emptyset})^* \\ N_{\{0\}} &\rightarrow (N_{\{0\}})^* \\ N_{\{0,1\}} &\rightarrow (N_{\{1\}})^* \\ N_{\{0,1\}} &\rightarrow (N_{\{0,1\}})^* \end{aligned}$$

Zadanie 3. Niech $bin(n)$ oznacza dwójkowy zapis liczby n, np. $bin(12) = 1100$. Niech w^R oznacza słowo w napisane do tyłu. Czy bezkontekstowy jest język

$$L = \{bin(n)\#(bin(n^2))^R : n \in \mathbb{N}\}$$

Rozwiązanie. Język nie jest bezkontekstowy. Trudność występuje już przy kwadratach liczb, których rozwinięcie dwójkowe jest postaci 10^n1 dla $n \geqslant 1$. Rozważmy przecięcie języka L z regularnym językiem

$$K = 100^*1\#(0+1)^*$$

Poniżej pokażemy, że przecięcie $L \cap K$ nie jest bezkontekstowe, z czego wynika, że sam język L nie jest bezkontekstowy.

Jak wygląda kwadrat liczby, której rozwinięcie dwójkowe jest zgodne z wyrażeniem 100^*1? Jeśli rozwinięciem jest 10^n1, dla $n \geqslant 1$, to kwadrat wynosi

$$(2^{n+1}+1)^2 = (2^{n+1})^2 + 2 \cdot 2^{n+1} + 1 = 2^{2n+2} + 2^{n+2} + 1$$

a więc, przy założeniu $n \geqslant 1$, rozwinięcie dwójkowe kwadaratu to

$$10^{n-1}10^{n+1}1.$$

Czyli język $L \cap K$ to

$$\{10^n1\#10^{n+1}10^{n-1}1 : n \geqslant 1\}.$$

Teraz pokażemy, że powyższy język nie jest bezkontekstowy. Dowód jest ten sam co dla języka $\{a^nb^nc^n : n \in \mathbb{N}\}$. Zastosujmy do języka $L \cap K$ lemat o pompowaniu dla języków bezkontekstowych (wystarczy zwykły lemat, niepotrzebny jest silniejszy lemat Ogdena). Niech $M \in \mathbb{N}$ stała z lematu o pompowaniu. Rozważmy słowo

$$w = 10^M1\#10^{M+1}10^{M-1}1 \in L \cap K.$$

Na mocy lematu istnieje podział

$$w = w_1w_2w_3w_4w_5$$

taki, że słowo w_2w_4 zawiera przynajmniej jedną literę, oraz zachodzi następujący warunek pompowania

$$w_1w_2^iw_3w_4^iw_5 \in L \cap K \qquad \text{dla dowolnego } i \in \mathbb{N}.$$

Pokażemy, że takiego podziału być nie może, co dowodzi, że język $L \cap K$ nie mógł być bezkontekstowy. Po pierwsze, słowa w_2, w_4 nie mogą zawierać ani 1 ani #, bo inaczej przy pompowaniu dla $i = 2$ byśmy dostali w języku $L \cap K$ słowo, które ma więcej niż pięć jedynek, albo więcej niż jeden #. A więc każde ze słów w_2 i w_4 musi się mieścić w całości w jednym z trzech bloków zer, które występują w słowie w. Wówczas pompowanie dla $i = 2$ zmieni długość przynajmiej jednego ale co najwyżej dwóch bloków, bez zmiany długości trzeciego bloku, co spowoduje wypadnięcie z języka $L \cap K$.

Zadanie *. Rozważmy automat ze stosem, który oprócz normalnych instrukcji, może też wykonać instrukcję "usuń wszystkie wystąpienia litery b ze stosu". Inaczej mówiąc, dozwolone są też przejścia postaci:

$$(p, a, usun(b), q)$$

gdzie p, q są stanami, a jest literą alfabetu wejściowego (może być też $a = \epsilon$), oraz b jest literą alfabetu stosowego.

Czy opisany powyżej model automatu potrafi rozpoznawać tylko języki bezkontekstowe?

Rozwiązanie. Tak. Niech $\mathcal{A}$ będzie automatem w nowym modelu. Pokażemy automat $\mathcal{A}'$, w zwykłym modelu automatu ze stosem, który rozpoznaje ten sam język co $\mathcal{A}$.

Załóżmy, że automat $\mathcal{A}$ korzysta tylko z przejść typu push i pop, oraz że akceptuje przez stan akceptujący. Nowy automat $\mathcal{A}'$ będzie miał te same stany co automat $\mathcal{A}$, ten sam stan początkowy, te same stany akceptujące (no i oczywiście ten sam alfabet wejściowy). Zmienią się tylko alfabet stosowy i przejścia. (Uważny czytelnik zauważy, że wszystkie przejścia typu push i pop oryginalnego automatu są zachowane w nowym automacie, dochodzi jednak kilka nowych przejść.)

Niech B będzie alfabetem stosowym oryginalnego automatu $\mathcal{A}$. Alfabet stosowy nowego automatu $\mathcal{A}'$ zawiera wszystkie symbole z oryginalnego alfabetu B, oraz dodatkowe symbole

$$\{usun(C) : C \subseteq B\}.$$

Idea jest taka, że jeśli stos zawiera sybmol $usun(C)$, to wszystkie wystąpienia liter ze zbioru C pod tym symbolem należy uznać za niebyłe.

Poniżej opisane są przejścia nowego automatu $\mathcal{A}'$.

1. Przejście postaci $(p, a, usun(b), q)$ w automacie $\mathcal{A}$ zastępujemy w automacie $\mathcal{A}'$ przez przejście, które czytając literę a zmienia stan z p na q, oraz kładzie na stosie symbol $usun(\{b\})$.

2. Chcemy, żeby na stosie symbole typu $usun$ nie sąsiadowały ze sobą. Aby to osiągnąć, sąsiednie wystąpienia łączymy ze sobą. Dla każdego stanu q, dodajemy ϵ-przejście,

$$q, usun(C_1) \cdot usun(C_2) \xrightarrow{\epsilon} q, usun(C_1 \cup C_2).$$

3. Załóżmy, że oryginalny automat $\mathcal{A}$ miał przejście postaci pop, czyli

$$q, b \xrightarrow{a} p, \epsilon.$$

   Przejście takie może być wykonane jeśli symbol b jest przykryty symbolem usuwającym litery nie zawierające b, czyli za pomocą przejść postaci:

$$q, usun(C) \cdot b \xrightarrow{a} p, usun(C) \qquad \text{dla } C \not\ni b.$$

(W powyższym napisie, czubek stosu jest z lewej strony, a więc b jest przykryte symbolem $usun(C)$.) Może się też zdarzyć, że b nie jest przykryte żadnym symbolem usuwającym, wówczas możemy zastosować oryginalne przejście

$$q, b \xrightarrow{a} p, \epsilon.$$

4. Przejścia push pozostają bez zmiany.

To kończy opis automatu $\mathcal{A}'$. Poniżej opisany jest niezmiennik, który dowodzi, że automat $\mathcal{A}'$ akceptuje te same słowa co automat $\mathcal{A}$.

Dla stosu nad nowym alfabetem stosowym, definiujemy jego redukt, który jest stosem nad oryginalnym alfabetem stosowym, w następujacy sposób:

- usuwamy wszystkie wystąpienia symbolu b, które są przykryte wystąpieniem symbolu $usun(C)$ takiego, że $b \in C$.
- następnie usuwamy wszystkie symbole typu $usun(C)$.

Na przykład redukt stosu

$$usun(\{b_1\}) \cdot b_2 \cdot usun(\{b_2\}) \cdot b_1 \cdot b_2 \cdot b_3 \cdot usun(\{b_1, b_2\}) \cdot b_3$$

to stos $b_2 \cdot b_3 \cdot b_3$.

Korzystając z pojęcia reduktu, możemy sformułować niezmiennik. Niezmiennik mówi, że dla każdego słowa w i każdej zawartości stosu γ nad oryginalnym alfabetem stosowym, następujące warunki są równoważne:

1. Istnieje bieg ρ automatu $\mathcal{A}$ po słowie w, który kończy się w stanie q z zawartością stosu γ. γ.
2. Istnieje stos γ', którego redukt to γ, oraz bieg ρ' automatu $\mathcal{A}'$ po słowie w, który kończy się w stanie q z zawartością stosu γ'.

Implikację z góry na dół dowodzi się przez indukcję po ilości przejść w biegu ρ, a implikację z dołu do góry dowodzi się przez indukcję po ilości przejść w biegu ρ'.