Zadanie 1

Przypuśćmy, że jakiś automat rozpoznaje język L. Weźmy minimalny taki automat, tj. każda litera stosowa jest przez niego wykorzystywana.

Dla każdego $s \in S_1 \sqcup S_2$ możemy określić $L(s)$ – zbiór słów, które zaakceptowałby automat, gdyby zaczynał z konfiguracji (ε, s) lub (s, ε) – w zależności czy $s \in S_2$ czy $s \in S_1$.

Jeśli $L(s) = \varnothing$, to oczywiście automat nie jest w stanie z konfiguracji zawierającej s gdziekolwiek na stosie dojść do konfiguracji z pustymi stosami. Istotnie, zaznaczając symbol s, symbole wrzucone na stos w wyniku zdjęcia s, symbole wrzucone w wyniku zdjęcia tych symboli, etc., a także litery w słowie, nad którymi automat wykonał wyżej wymienione przejścia, widzimy, że możemy zupełnie usunąć z biegu automatu, z konfiguracji i ze słowa wszystkie symbole i litery niezaznaczone i nadal uzyskamy poprawny bieg. Tym razem zacznie się on w konfiguracji, w której jeden ze stosów jest pusty, a drugi zawiera jedynie s. Poprawność biegu wynika z tego, że automat nie ma stanów, a zatem nie jest w stanie rozpoznać, czy między zaznaczonymi operacjami cokolwiek się działo. Oczywiście, w pełni formalny dowód jest przez indukcję po długości biegu.

Teraz niech $L(u, w)$ będzie zbiorem słów, które akceptuje automat zaczynający z konfiguracji (u, w). Ponieważ automat nie ma ε-przejść, to $\epsilon \notin L(u, w)$ dla $(u, w) \neq (\varepsilon, \varepsilon)$. Ponieważ automat nie ma stanów, łatwo widać, że $L(s)L(u, w) \subseteq L(su, w)$ dla $s \in S_1$, $L(s)L(u, w) \subseteq L(u, sw)$ dla $s \in S_2$ – istotnie, automat może najpierw wczytać dowolne słowo powodujące, że uda mu się zdjąć s bez ruszania reszty stosu, a następnie zdjąć resztę stosu. Przez indukcję widzimy, że można to uogólnić na $L(r, \varepsilon)L(u, w) \subseteq L(ru, w)$, $L(\varepsilon, r)L(u, w) \subseteq L(u, rw)$. Ponadto argument z zeszłego paragrafu pokazuje, że jeśli s występuje w którymkolwiek ze słów u, w, to każdy element $L(u, w)$ zawiera jako podciąg element $L(s)$. Co więcej, każdy element $L(u, w)$ jest przeplotem języków $L(s)$ dla $s \in u, w$ (aczkolwiek oczywiście nie każdy taki przeplot jest elementem $L(u, w)$).

W szczególności: jeśli w pewnym momencie akceptującego biegu oba stosy są niepuste: $(u, w) \in S_1^+ \times S_2^+$, to $L(u, \varepsilon)$ i $L(\varepsilon, w)$ są niepuste (bo na pewno zawierają chociażby konkatenację L dla poszczególnych symboli odpowiednio u lub w, a te języki są niepuste, gdyż wszystkie symbole w u i w są zdejmowalne). Ponadto słowo puste nie należy do tych języków. Jednakże zarówno $L(u, \varepsilon)L(\varepsilon, w)$ jak i $L(\varepsilon, w)L(u, \varepsilon)$ są podzbiorami $L(u, w)$. Jednak z założenia $L(u, w)$ jest ilorazem L przez już wczytany prefiks słowa, gdyż rozważany automat ma akceptować wyłącznie język L. Zatem wszystkie słowa z $L(u, \varepsilon)L(\varepsilon, w)$ jak i z $L(\varepsilon w)L(u, \varepsilon)$ są sufiksami $a^n b^n c^n$ dla jakichś n (być może różnych dla różnych słów). W szczególności wszystkie słowa z $L(\varepsilon, w)$ muszą kończyć się na c, skoro w $L(u, \varepsilon)L(\varepsilon, w)$ mają być same (niepuste) sufiksy $a^n b^n c^n$. Ale to oznacza, że ponieważ w $L(\varepsilon, w)L(u, \varepsilon)$ są same sufiksy $a^n b^n c^n$, to wszystkie słowa w $L(u, \varepsilon)$ mogą się składać jedynie z samych liter c. Analogiczny argument pokazuje, że także $L(\varepsilon, w) \subseteq c^*$. Zatem uzyskujemy, że dla każdego symbolu s występującego w u lub w mamy $L(s) \subseteq c^*$, a zatem na mocy obserwacji, że $L(u, w)$ składa się z pewnych przeplotów $L(s)$ łatwo widzimy, że $L(u, w) \subseteq c^*$.

Łatwo można teraz zakończyć mówiąc, że przecież konfiguracja początkowa ma oba stosy niepuste, zatem uzyskalibyśmy, że każdy element L należy do c^*, co jest oczywistym absurdem.

Jednakże zastanówmy się, czy wymóg, ażeby oba stosy początkowe były niepuste jest w ogóle konieczny. Powyższy dowód mówi, że jeśli tylko w jakimś momencie oba stosy są niepuste, to od razu wiemy, że wszystkie symbole na tych stosach generują wyłącznie słowa złożone z samych liter c. W szczególności – zupełnie nie jest istotna kolejność w jakiej będziemy przetwrzać symbole z tych stosów. Zatem nie ma konieczności operowania na dwóch stosach – rozpatrzmy automat z jednym stosem nad alfabetem $S_1 \sqcup S_2$. który symuluje automat z zadania, tj. bierze symbol z góry stosu (tylko to wolno automatowi stosowemu), patrzy na odpowiednie przejście automatu z zadania, wczytuje odpowiednią literę z wejścia, po czym zamiast wkładać w_1, w_2 na dwa stosy, to wkłada go na jeden – łącząc te słowa dowolnym przeplotem, np. konkatenacją.

Jeśli tylko w biegu automatu z zadania kiedykolwiek oba stosy były niepuste, to wiemy, że już do końca będziemy mieli jedynie symbole odpowiedzialne za słowa złożone z liter c, zatem możemy operować na nich na jednym stosie, bo symbole te są przemienne. Jeśli jednak cały czas któryś ze stosów jest niepusty, to oczywiście można operować na tym drugim, bo pusty stos nic nie zmienia w przejściach (dopóki się czegoś na niego nie włoży). Jeśli jeden stos się opróżni, a w tym samym czasie drugi stanie się niepusty, nasz nowy automat także poprawnie to symuluje. Oczywiście formalny dowód równoważności jest przez indukcję po długości słowa.

Uzyskaliśmy więc automat z jednym stosem akceptujący ten sam język, to automat z zadania, czyli język L. Taki automat jednak nie istnieje.

Zadanie 2

Nie, nie musi to być prawda.

Wiemy z wykładu, że klasa języków deterministycznych bezkontekstowych nie jest domknięta na branie sumy. Niech więc L_1, L_2 będą takimi językami deterministycznymi bezkontekstowymi, że $L_1 \cup L_2$ nie jest deterministyczny bezkontekstowy. Niech Σ będzie alfabetem, nad którym są L_1, L_2. Niech ponadto $A = \Sigma \sqcup \{\clubsuit, \spadesuit\}$.

Określmy $L = \{\clubsuit w : w \in L_1\} \sqcup \{\spadesuit w : w \in L_2\}$. Oczywiście L jest deterministyczny bezkontekstowy – automat czyta pierwszą literę, po czym wie, czy ma parsować słowo z L_1 czy z L_2, a języki te umie rozpoznawać deterministycznie.

Niech więc $M = \{\clubsuit, \spadesuit\}$. Oczywiście M jest deterministyczny bezkontekstowy, gdyż jest skończony, zatem regularny. Jednakże $\mathrm{cl}(L, M) = L_1 \cup L_2$, gdyż jedyne wystąpienia słów z M w słowach z L znajdują się na początku słów. Zatem $\mathrm{cl}(L, M)$ nie jest deterministyczny bezkontekstowy.

Zadanie 3

Tak, ten język jest rozpoznawany przez taki automat.

Dla ustalenia uwagi: $F_0 = a, F_1 = b, F_n = F_{n-1}F_{n-2}$ jest ciągiem słów Fibonacciego.

Fragmenty stosu oddzielone symbolami początkowymi ρ będę nazywał ministosami, tj. stos $\varpi\varpi\varpi\rho\varpi\varpi\rho\rho$ składa się z trzech ministosów: $\varpi\varpi\varpi\rho$, $\varpi\varpi\rho$, ρ. Skonstruuję automat kopiujący nad alfabetem stosowym $\{\varpi, \rho\}$, który rozpozna język słów Fibonacciego. W pierwszej fazie automat niedeterministycznie dokłada ileś symboli ϖ na stos, uzyskując stos: $\varpi^n\rho$.

Od tej pory automat będzie postępował deterministycznie, a ponadto chcę, aby ministos $\varpi^k\rho$ odpowiadał w jakiś sposób słowu F_k. Mianowcie, działanie automatu jest od teraz podzielone na rundy. Jeśli na początku rundy stos składa się z ministosów $\varpi^{k_1}\rho, \ldots, \varpi^{k_m}\rho$, to automat ma zaakceptować wtedy i tylko wtedy, gdy pozostałe wejście to $F_{k_1}F_{k_2}\ldots F_{k_m}$.

Na początku każdej rundy automat może sobie oglądnąć dwa górne symbole z ministosu znajdującego się na szczycie (lub stwierdzić, że ministos jest krótszy niż 2 znaki) – ε-przejściami zdejmuje je ze stosu, zapisując w skończonym stanie, po czym odczytawszy te conajwyżej dwa symbole, odkłada je na stos (pamięta w stanie, jakie one były, więc nie ma z tym problemu).

Jeśli automat stwierdził, że aktualny ministos to ρ, to znaczy, że powinien wczytać słowo F_0. Zatem wykonuje przejście, w którym zdejmuje ρ ze stosu (usuwając obecny ministos), wczytując literę a.

Jeśli stwierdził, że aktualny ministos to $\varpi\rho$, to powinien wczytać słowo F_1. Zatem ε-przejściem zdejmuje ϖ, po czym zdejmuje ρ czytając literę b.

W przeciwnym wypadku wie, że aktualny mikrostos to $\varpi^n\rho$ dla pewnego $n \geqslant 2$ i ma on odpowiadać słowu $F_n = F_{n-1}F_{n-2}$. Zatem automat zdejmuje dwa ϖ ze stosu, wykonuje operację „kopiuj”, po czym wkłada jedno ϖ na stos. Oczywiście uzyskał wtedy dwa mikrostosy: $\varpi^{n-1}\rho$, $\varpi^{n-2}\rho$, które z założenia indukcyjnego odpowiadają słowom F_{n-1} i F_{n-2}.

Formalny dowód poprawności przebiega oczywiście przez prostą indukcję po n.