Języki, Automaty i Obliczenia

Zadania

Damian Niwiński i Wojciech Rytter

31 stycznia 2009

Podziękowania. Autorzy dziękują Wszystkim, w szczególności Studentom, którzy sygnalizowali błędy, sugerowali nowe zadania lub podali ciekawe (i poprawne :) rozwiazania.

1 Słowa, liczby, grafy

Do tej partii zadań może podejść Czytelnik bez żadnej znajomości teorii automatów.

1. *Słowa pierwotne.* Słowo $w \in \Sigma^*$ nazywamy *pierwotnym* (ang. *primitive*), jeśli nie da się go przedstawić $w = v^n$, inaczej niż dla $n = 1$ i $v = w$.

   (a) Dowieść, że dla każdego słowa niepustego w, istnieje *dokładnie jedno* słowo pierwotne v, takie że $w = v^n$, dla pewnego $n \geq 1$.

   Liczbę n nazywamy *wykładnikiem* słowa w.

   (b) O słowach wv i vw mówimy, że są w *relacji koniugacji.* Dowieść, że jest to relacja równoważności.

   Dowieść, że dwa słowa będące w relacji koniugacji mają ten sam wykładnik. Jaka jest moc klasy abstrakcji relacji koniugacji dla słowa o długości m i wykładniku n?

2. *Język nawiasowy.* Wykazać, że zbiór poprawnie uformowanych ciągów nawiasów może być zdefiniowany na dwa *równoważne* sposoby:

   - Jako najmniejszy zbiór L zawierający ciąg pusty oraz taki, że jeśli $w, v \in L$, to również $(w), wv \in L$.
   - Zbiór słów nad alfabetem $\{(,)\}$, w których ilość ")" jest taka sama jak ilość "(", a w każdym prefiksie ilość ")" jest niewiększa niż ilość "(" .

3. *Zbiory semi–liniowe.* Zbiór liczb naturalnych postaci $\{a + bn : n \in \mathbf{N}\}$, dla ustalonych $a, b \in \mathbf{N}$ nazywamy *liniowym.* Zbiór bedący sumą skończonej liczby zbiorów liniowych nazywamy *semiliniowym.* (Gdy sumowana rodzina jest pusta, otrzymujemy zbiór pusty.)

(a) Dowieść, że każdy zbiór postaci $\{a + b_1 n_1 + \ldots + b_k n_k \,:\, n_1, \ldots, n_k \in \mathbf{N}\}$, dla ustalonych k i $a, b_1, \ldots, b_k \in \mathbf{N}$, jest semi–liniowy.

Wskazówka. Wykorzystać podziały zbioru liczb naturalnych na klasy abstrakcji relacji przystawania *modulo* m, dla odpowiednio dobranych liczb m.

(b) Dowieść, że zbiór liczb naturalnych A jest semi–liniowy wtedy i tylko wtedy, gdy jest prawie periodyczny tzn. gdy istnieją $c \in \mathbf{N}$ i $d \in \mathbf{N} - \{0\}$ takie, że dla każdego $x > c$, $x \in A \Leftrightarrow x + d \in A$.

(c) W grafie zorientowanym ustalamy dwa wierzchołki. Interesuje nas zbiór długości wszystkich możliwych ścieżek pomiędzy tymi wierzchołkami. Dowieść, że jest to zbiór semi–liniowy.

(d) Dowieść, że rodzina zbiorów semi–liniowych jest zamknięta na skończone sumy, przecięcia oraz na uzupełnienie względem N.

4. *Graf gry (J. P. Jouannaud).* Rozważamy następującą grę pomiędzy Barmanem i Klientem. Przed Barmanem na obrotowym talerzu stoją cztery szklanki tworząc wierzchołki kwadratu. Szklanka może być ustawiona normalnie lub dnem do góry, jednak Barman ma przepaskę na oczach i rękawiczki na rękach, tak że nie może tego zobaczyć ani wyczuć. Ruch Barmana polega na odwróceniu jednej lub dwóch dowolnie wybranych szklanek. Ruch Klienta polega na obróceniu talerza (o wielokrotność ćwierć-obrotu). Barman wygrywa, jeśli w jakimś momencie gry wszystkie szklanki ustawione są w tej samej pozycji (zostanie o tym lojalnie poinformowany).

   Czy Barman noże wygrać startując z nieznanej konfiguracji początkowej, a jeśli tak, to w jakiej liczbie ruchów?

   Czy graliby Państwo o pieniądze z Barmanem ? A gdyby zamiast kwadratu były 3 szklanki ustawione w trójkąt lub 5 w pięciokąt ?

5. *Kody.* Zbiór $C \subseteq \Sigma^+$ nazywamy *kodem*, jeśli każde słowo $w \in \Sigma^*$ dopuszcza co najwyżej jedną faktoryzację względem C (tzn., da się “odkodować”).

   Niech $\Sigma = \{a, b\}$. Dowieść, że zbiór $\{aa, baa, ba\}$ jest kodem, a zbiór $\{a, ab, ba\}$ nie jest.

   Jeśli skończony zbiór C nie jest kodem, oszacować z góry długość najkrótszego słowa, które o tym świadczy (tzn. dopuszcza dwie różne faktoryzacje).

   Patrz też rozdz. 2.7, zadanie 14.

2 Języki regularne

2.1 Automaty skończone i wyrażenia regularne

1. Dowieść, że dla dowolnych języków L, M, $(L^* M^*)^* = (L \cup M)^*$.

2. Dowieść, że wyrażenie regularne $(00+11+(01+10)(00+11)^*(10+10))^*$ reprezentuje zbiór wszystkich słów nad alfabetem $\{0, 1\}$, w których zarówno liczba wystąpień 0 jak i liczba wystąpień 1 są parzyste.

Jak najkrócej reprezentować zbiór słów, w których te liczby są tej samej parzystości?

3. Zbudować automat nad alfabetem $\{0, 1\}$, rozpoznający słowa, w których liczba jedynek na pozycjach parzystych jest parzysta, a liczba jedynek na pozycjach nieparzystych jest nieparzysta.

4. *Dodawanie.* Rozważamy słowa nad alfabetem $\{0, 1\}^3$. Powiemy, że słowo $\langle a_1, b_1, c_1\rangle \ldots \langle a_n, b_n, c_n\rangle$ długości n *reprezentuje dodawanie*, jeśli liczba reprezentowana binarnie przez słowo $c_1 \ldots c_n$ jest sumą liczb reprezentowanych przez $a_1 \ldots a_n$ i $b_1 \ldots b_n$. Na przykład, ciąg $\langle 0, 0, 1\rangle\langle 1, 1, 1\rangle\langle 0, 1, 0\rangle\langle 1, 1, 0\rangle$ reprezentuje dodawanie (5+7=12). Podać wyrażenie regularne opisujące zbiór wszystkich słów nad alfabetem $\{0, 1\}^3$ reprezentujących dodawanie w powyższym sensie.

5. *Podzielność.*

   (a) Skonstruować skończony automat deterministyczny nad alfabetem $\{0, 1, \ldots, 8, 9\}$ rozpoznający zbiór dziesiętnych reprezentacji wielokrotności liczby 7.

   (b) Jak wyżej, ale przy reprezentacji odwrotnej, tj. poczynając od cyfr najmniej znaczących.

   (c) Uogólnić niniejsze zadanie.

6. *Alfabet jednoliterowy.* Dowieść, że język $L \subseteq \{a\}^*$ jest regularny wtedy i tylko wtedy, gdy zbiór liczb naturalnych $\{n : a^n \in L\}$ jest semi–liniowy w sensie rozdziału 1. Dowieść, że dla dowolnego zbioru $X \subseteq \{a\}^*$, język X^* jest regularny.

7. *Zbiory semi–liniowe* (por. zadanie 3 z rozdziału 1).

   (a) Dowieść, że dla dowolnego języka regularnego L zbiór $\{|w| : w \in L\}$ jest semi–liniowy.

   W szczególności, języki regularne nad *jednoliterowym* alfabetem mogą być utożsamiane ze zbiorami semi–liniowymi poprzez bijekcję $w \mapsto |w|$.

   (b) Dowieść, że jeśli $M \subseteq \mathbf{N}$ jest zbiorem semi–liniowym, to język $\{\mathrm{bin}(m) : m \in M\}$ jest regularny, gdzie $\mathrm{bin}(m)$ oznacza binarną reprezentację liczby m.

   Uwaga. Fakt analogiczny do 7b można udowodnić dla dowolnej reprezentacji, a zatem zbiór semi–liniowy jest regularny w reprezentacji o podstawie k, dla $k \geq 1$. W przeciwnym kierunku wiadomo, że jeśli zbiór jest regularny w reprezentacjach o względnie pierwszych podstawach p i q, to jest semi–liniowy — jest to wniosek z głębokiego Twierdzenia Cobhama.

8. Dowieść, że dla danych liczb $a, b, p, r \in \mathbf{N}$, język

$$L = \{\,\mathrm{bin}(x)\,\$\,\mathrm{bin}(y) \,:\, (a \cdot x + b \cdot y) \equiv r \pmod{p}\}$$

   jest regularny.

Przykłady automatów związanych z rozpoznawaniem wzorca Czytelnik znajdzie poniżej w punkcie 2.5.

2.2 Lemat o pompowaniu

1. Dowieść, że następujące języki nie są regularne:

   - $\{a^n b^n \,:\, n \in N\}$
   - $\{a^{2^n} \,:\, n \in N\}$
   - $\{a^p \,:\, p$ jest liczbą pierwszą$\}$
   - $\{a^i b^j \,:\, \mathrm{nwd}(i, j) = 1\}$
   - $\{a^m b^n \,:\, m \neq n\}$
   - $\{\mathrm{bin}(p) \,:\, p$ jest liczbą pierwszą$\}$.

2. Dowieść, że zbiór palindromów nad alfabetem o co najmniej dwóch elementach nie jest regularny.

3. Dowieść, że zbiór wyrażeń regularnych nie jest regularny.

4. Dowieść, że jeśli w Zadaniu 4 z punktu 2.1 zamienimy *dodawanie* przez *mnożenie*, to otrzymany język (nad alfabetem $\{0, 1\}^3$) jest wprawdzie dobrze określony, ale nie jest regularny.

5. Udowodnić nieco silniejszy wariant Lematu o pompowaniu:

   Jeśli L jest regularny, to

   > **(*)** Istnieje stała n_0, że dla każdych słów v, w, u takich że $|w| \geq n_0$ i $vwu \in L$, istnieją x, y, z takie, że $w = xyz$, $0 < |y| \leq n_0$, oraz $\forall n \in \mathbf{N}$, $vxy^n zu \in L$.

   Podać przykład języka L, który spełnia (*), choć nie jest regularny.

   Wskazówka. $\sum_p \underbrace{cb^*cb^* \ldots cb^*}_{p} + (b + c)^* cc(b + c)^*$, gdzie p przebiega liczby pierwsze.

2.3 Własności domknięcia języków regularnych

1. Dowieść, że dla języka regularnego $L \subseteq \Sigma^*$ i dowolnego zbioru $X \subseteq \Sigma^*$ języki

   $$X^{-1}L = \{w : (\exists v \in X)\, vw \in L\}$$

   i

   $$LX^{-1} = \{w : (\exists u \in X)\, wu \in L\}$$

   są regularne.

2. Dowieść, że język L jest regularny wtedy i tylko wtedy, gdy język L^R słów stanowiących lustrzane odbicia słów z L jest regularny.

   Uwaga. Lustrzane odbicie w^R słowa w możemy w ścisły sposób zdefiniować rekurencyjnie: $\epsilon^R = \epsilon$ i $(w\sigma)^R = \sigma w^R$, dla $w \in \Sigma^*$, $\sigma \in \Sigma$.

3. Dla danego automatu A rozpoznającego język L, skonstruować automat rozpoznający język
$$Cycle(L) = \{vu \,:\, uv \in L\}$$
Czy z faktu, że język $Cycle(L) = \{vu \,:\, uv \in L\}$ jest regularny, można wnioskować, że język L jest regularny?

4. Niech L bedzie językiem regularnym nad alfabetem $\{0,1\}$. Dowieść, że następujący język jest regularny:
$$\{w \,:\, w \in L \wedge \text{ spośród słów o dlugości } |w|,\ w \text{ jest najmniejsze w porządku leksykograficznym}\}$$

5. Przyjmujemy, że każde niepuste słowo binarne $w \in \{0,1\}^*$, $w = w_1 \dots w_k$, reprezentuje pewien ułamek w przedziale $[0,1)$,
$$\mathrm{bin}(w) = w_1\frac{1}{2} + w_2\frac{1}{2^2} + \ldots + w_k\frac{1}{2^k}$$
Dla liczby rzeczywistej $r \in [0,1]$, niech
$$L_r = \{w \,:\, \mathrm{bin}(w) \le r\}$$
Dowieść, że język L_r jest regularny wtedy i tylko wtedy, gdy liczba r jest wymierna.

6. Podać przykład nieskończonego języka zamkniętego na branie podsłów, który nie zawiera nieskończonego języka regularnego.

7. Niech L będzie językiem regularnym. Dowieść, że następujące języki są również regularne:
   - $\frac{1}{2}L =_{df} \{w \,:\, (\exists u)\, |u| = |w| \wedge wu \in L\}$
   - $\sqrt{L} =_{df} \{w \,:\, ww \in L\}$

8. Niech L będzie językiem regularnym. Dowieść, że następujące języki są również regularne:

   (a) $\mathrm{Root}(L) = \{w \,:\, (\exists n \in N)\, w^n \in L\}$

   (b) $\mathrm{Sqrt}(L) = \{w \,:\, (\exists u)|u| = |w|^2 \wedge wu \in L\}$
   Wskazówka. $n^2 = 1 + 3 + \ldots (2n-1)$.

   (c) $\mathrm{Log}(L) = \{w \,:\, (\exists u)|u| = 2^{|w|} \wedge wu \in L\}$
   Wskazówka. $2^n = 1 + 2 + 2^2 + 2^{n-1} + 1$.

   (d) $\mathrm{Fibb}(L) = \{w \,:\, (\exists u)|u| = F_{|w|} \wedge wu \in L\}$
   gdzie F_n jest n-tą liczbą Fibonacciego tzn.
   $$F_1 = F_2 = 1$$
   $$F_{n+2} = F_n + F_{n+1}$$

9. Dowieść, że dla dowolnego języka regularnego L, regularny jest również język
$$\{w \,:\, w^{|w|} \in L\}.$$

10. Niech będzie dany język regularny L i dowolne, niekoniecznie regularne, języki L_1, $\ldots$, L_m. Skonstruować skończony automat deterministyczny rozpoznający język nad alfabetem $\{1, \ldots, m\}$

$$L = \{i_1 i_2 \ldots i_k : L_{i_1} L_{i_2} \ldots L_{i_k} \subseteq L\}$$

11. *Nietrywialnym palindromem* nazwiemy każdy palindrom o długości co najmniej 2. Niech $\mathrm{Pal}_\Sigma^{\geq 1}$ oznacza zbiór nietrywialnych palindromów nad alfabetem Σ. Dowieść, że język $(\mathrm{Pal}_\Sigma^{\geq 1})^*$ jest regularny wtedy i tylko wtedy, gdy $|\Sigma| = 1$.

    Czy język $(\{ww^R : w \in (0+1)^*\})^*$ jest regularny?

12. Które z następujących języków nad alfabetem $\{0, 1\}$ są regularne?
    (a) Zbiór słów posiadających nietrywialny palindrom jako prefiks.
    (b) Zbiór słów posiadających palindrom długości parzystej jako prefiks.
    (c) Zbiór słów posiadających nietrywialny palindrom długości nieparzystej jako prefiks.

13. Oznaczmy przez $Ile(w, x)$ liczbę wystąpień słowa w jako podsłowo x (wystąpienia nie muszą być rozłączne). Czy następujący język nad alfabetem $\{a, b\}$ jest regularny? Jeśli tak, to podać wyrażenie regularne. Jeśli nie, to udowodnić, że nie jest.
    (a) $L_1 = \{x \;:\; Ile(ab, x) = Ile(ba, x) + 1\}$
    (b) $L_2 = \{x \;:\; Ile(aba, x) = Ile(bab, x)\}$

14. Niech L będzie językiem regularnym. Dowieść, że języki:
    - $L_{+--} = \{w \,:\, (\exists u)\, |u| = 2|w| \wedge wu \in L\}$
    - $L_{++-} = \{w \,:\, (\exists u)\, 2|u| = |w| \wedge wu \in L\}$
    - $L_{-+-} = \{w \,:\, (\exists u, v)\, |u| = |v| = |w| \wedge uwv \in L\}$

    są językami regularnymi, a język
    - $L_{+-+} = \{uv \,:\, (\exists w)\, |u| = |v| = |w| \wedge uwv \in L\}$

    może nie być regularny.

15. Czy zachodzi fakt: jeśli L jest regularny to istnieja dwa niepuste słowa u, v takie, że zachodzi równość ilorazów $L\{uv\}^{-1} = L\{vu\}^{-1}$?

16. Podać przykład języka nieregularnego L, takiego że L^2 jest językiem regularnym.

17. Dla słów u, v o tej samej długości, określamy odległość Hamminga

$$d(u, w) = |\{i : w_i \neq v_i\}|.$$

    Udowodnić, że dla dowolnego języka regularnego L i stałej K, następujący język jest również regularny:

$$\{\ w \;:\; (\exists\ u \in L)\ |u| = |w| \text{ oraz } d(u, w) \leq K\ \}$$

18. Przeplotem słów w i v nazwiemy dowolne słowo długości $|w| + |v|$, które można rozbić na rozłączne podciągi w i v. Na przykład, przeplotami słów ab i acb są słowa $abacb$, $aabcb$, $acabb$, $acbab$, $aacbb$. Przeplotem języków L i M jest zbiór wszystkich możliwych przeplotów słów $w \in L$, $v \in M$. Język ten oznaczamy $L \parallel M$.

    Dowieść, że jeśli L i M są językami regularnymi, to $L \parallel M$ jest również regularny.

19. Określamy $L^\sharp = L \cup (L \parallel L) \cup (L \parallel L \parallel L) \cup \ldots$. Podać przykład języka regularnego L nad dwuelementowym alfabetem, dla którego $L^\sharp$ nie jest językiem regularnym.

2.4 Automaty minimalne.

1. Niech $w \in \Sigma^*$ będzie słowem o długości $|w| = n > 0$. Dowieść, że minimalny automat deterministyczny rozpoznający wszystkie słowa nad Σ, których sufiksem jest w, ma dokładnie $n+1$ stanów.

2. Niech $w \in \Sigma^*$ będzie słowem o długości $|w| = n > 0$. Dowieść, że minimalny automat deterministyczny rozpoznający wszystkie podsłowa słowa w ma nie więcej niż $2n$ stanów.

3. Narysować diagramy automatów minimalnych rozpoznających odpowiednio

    (a) wszystkie podsłowa

    (b) wszystkie sufiksy

    słowa *abbababa* (dla przejrzystości rysunków pominąć stan „śmietnik").

    Ile stanów maja analogiczne automaty dla $w_n = ab(ba)^n$ (licząc stan „śmietnik") ?

4. Skonstruować minimalny deterministyczny automat skończony dla języka

    $$L = \{a^i bb^* a^j \ : \ 2 \mid (i+j)\}.$$

5. Skonstruować minimalny deterministyczny automat skończony dla języka

    $$L = \{\ a_1 a_2 \ldots a_n \ : \ n > 0,\ a_i \in \{0,1,2,3,4\},\ MAX_{i,j}(a_i - a_j) \le 2\}.$$

6. Opisać strukturę i narysować diagram minimalnego deterministycznego automatu skończonego akceptującego wszystkie skończone ciągi zerojedynkowe takie, że każde k kolejnych symboli zawiera dokładnie 2 jedynki i każde kolejnych $j < k$ symboli zawiera co najwyżej dwie jedynki:

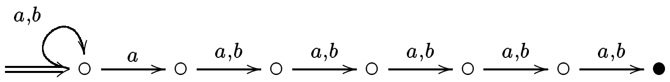
    (a) Dla $k = 3$;

    (b) dla $k = 4$.

    (W szczególności słowo puste należy do obu języków, a słowo 111 do żadnego z nich.)

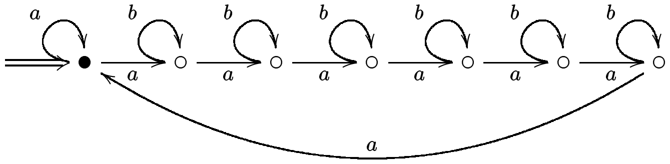
7. Dowieść, że minimalny automat deterministyczny równoważny podanemu automatowi niedeterministycznemu o n stanach ($n \geq 2$) ma dokładnie

   (a)

   2^{n-1} stanów,

   (b) (*)

   2^n stanów.

   Na powyższych rysunkach $n = 7$, stan początkowy jest wskazany przez $\Rightarrow$, a stan akceptujący jest oznaczony przez $\bullet$.

   Uwaga. Własność słów rozpoznawaną przez automat z punktu 7a można łatwo opisać w języku polskim, w przypadku punktu 7b jest to trudniejsze.

8. Oznaczmy przez Σ_k alfabet $\{0, 1, \ldots k\}$. Niech $NPAL_k$ oznacza zbiór słów nad alfabetem Σ_k nie zawierających, jako podsłowa, palindromu (symetrycznego słowa długości co najmniej 2). Ile stanów ma minimalny automat deterministyczny akceptujący $NPAL_{99}$? Podać rozwiązanie ogólne dla $k = 0, 1, 2, \ldots$

9. Trzy drużyny piłkarskie A, B i C rozgrywają między sobą serię spotkań towarzyskich, przy czym umówiły się, że każdy kolejny mecz jest rozgrywany pomiędzy drużyną, która wygrała poprzedni mecz i drużynę, która nie brała udziału w tym spotkaniu (tzn. jeśli drużyna X wygrała z drużyną Y, to następny mecz rozegrają X i Z). Zakładając, że *nie ma remisów*, rozważamy zbiór słów nad alfabetem $\{A, B, C\}$, stanowiących ciągi możliwych wyników spotkań. Dowieść, że jest to język regularny. Zbudować minimalny automat deterministyczny, który go rozpoznaje.

10. Pan X zakupił pakiety trzech różnych akcji: A, B i C. Każdego dnia bada wzajemne położenie wartości swoich akcji, które może być reprezentowane jako *uporządkowanie* zbioru $\{A, B, C\}$, w kierunku od najmniej do najbardziej wartościowej. (Przyjmujemy założenie, że dwie różne akcje mają zawsze *różne* wartości.) Pan X postanowił, że jeśli w ciągu dwóch kolejnych dni któraś z akcji dwukrotnie spadnie na niższą pozycję, to sprzeda tę akcję. Np. jeśli kolejne notowania (w tym wypadku: uporządkowania) są $(A, B, C), (B, C, A), (C, B, A)$ to pan X sprzeda pakiet akcji C. Rozważamy zbiór G wszystkich uporządkowań liniowych zbioru $\{A, B, C\}$. Dowieść, że zbiór tych ciągów nad alfabetem G, które opisują takie wyniki notowań giełdowych, przy których pan X nie pozbędzie się żadnego pakietu akcji, jest językiem regularnym. Znaleźć liczbę stanów automatu minimalnego akceptującego ten język.

11. Pan X postanowił, że każdego dnia będzie pracował lub nie, przestrzegając przy tym zasady, by na żadne siedem kolejnych dni nie przypadało więcej niż cztery dni pracy. Możliwy rozkład dni pracy pana X w ciągu n kolejnych dni możemy przedstawić jako n-bitowe słowo, gdzie 1 odpowiada dniowi pracy, a 0 dniowi odpoczynku. Dowieść, że zbiór wszystkich tak otrzymanych słów jest językiem regularnym (nad alfabetem $\{0, 1\}$). Znaleźć minimalny automat deterministyczny.

12. Dowieść, że istnieje nieskończenie wiele słów $w \in (a+b)^*$ takich, że jeśli zastosujemy homomorfizm $a \mapsto 0$, $b \mapsto 1$, to otrzymamy zapis binarny liczby podzielnej przez 3, a kiedy zastosujemy homomorfizm $a \mapsto 1$, $b \mapsto 0$, to również otrzymamy zapis binarny liczby podzielnej przez 3 (oczywiście ignorujemy początkowe zera).

13. Rozważamy automat wydający napoje, działający na następujących zasadach.

    — Każdy napój kosztuje 1 złoty.

    — W chwili początkowej automat nie zawiera żadnych monet.

    — Automat przyjmuje monety: 1 złoty lub 1 euro; w tym ostatnim przypadku wydaje 3 złote reszty po warunkiem, oczywiście, że ma dostateczną ilość monet jednozłotowych (zakładamy, że €1 = 4 zł).

    — Jeśli automat nie jest w stanie wydać reszty — sygnalizuje **błąd**.

    — Jeśli po wrzuceniu monety i ewentualnym wydaniu reszty wartość wszystkich monet zgromadzonych w automacie osiągnie równowartość 8 zł, wszystkie monety są wyjmowane (*reset*).

    Historią działania jest ciąg wrzuconych monet (złoty lub euro). Historia jest *udana*, jeśli w trakcie jej realizacji ani razu nie wystąpił błąd.

    Skonstruować minimalny automat deterministyczny rozpoznajacy zbiór wszystkich udanych historii.

2.5 Rozpoznawanie wzorca w tekście

1. Deterministyczny automat z zadania 1 z punktu 2.4 rozpoznający język $\Sigma^* w$ można skonstruować biorąc za stany wszystkie prefiksy słowa w (a zatem $|w| + 1$ stanów) oraz przejścia $v \stackrel{a}{\to} u$, gdzie u jest maksymalnym sufiksem słowa va będącym jednocześnie prefiksem w. (Efektywna konstrukcja tegoż automatu, patrz zadanie 3 w punkcie 2.7.) Dowieść, że liczba nietrywialnych przejść „wstecznych”, tj. przejść postaci $v \stackrel{a}{\to} u$, gdzie $|v| > |u| > 0$, jest nie większa niż $|w|$. Zauważmy, że daje to możliwość reprezentacji automatu o rozmiarze proporcjonalnym do $|w|$ (przejścia postaci $v \stackrel{a}{\to} \epsilon$ możemy pominąć).

   Wskazówka. Wykazać, że dla każdej liczby k istnieje co najwyżej jedno nietrywialne przejście wsteczne, takie, że $|va| - |u| = k$.

2. *Rozpoznawanie podsłów.*

   (a) Dla danego słowa w o długości $|w| = n$ skonstruować automat deterministyczny o $\leq 2n + 1$ stanach rozpoznający dokładnie zbiór sufiksów słowa w.

Wskazówka. Jako stany automatu można wziąć zbiory pozycji S_x ($S_x \subseteq \{0, 1, \ldots, n\}$) kończących wystąpienie x jako podsłowa słowa w. Oszacowanie na liczbę stanów wynika ze spostrzeżenia, że różne zbiory S_x, S_y są albo w relacji inkluzji albo rozłączne, a zatem, ze względu na inkluzję, tworzą drzewo o nie więcej niż n liściach.

(b) Powyższy automat zawiera stan typu „czarna dziura”, mianowicie stan $\emptyset = S_x$, gdzie x nie jest podsłowem w. Dowieść, że liczbę przejść nie prowadzących do stanu $\emptyset$ można oszacować z góry przez $3n$.

Wskazówka. Wziąć drzewo rozpinające grafu automatu i oszacować liczbę pozostałych krawędzi przez liczbę sufiksów w.

(c) Opisany wyżej automat jest minimalny dla zbioru sufiksów. Tę samą konstrukcję można zastosować również do rozpoznawania zbioru wszystkich podsłów słowa w, ale otrzymany automat nie musi być minimalny. Podać przykład.

Wskazówka: 4 grudnia.

2.6 Warianty automatów skończonych

1. *Automaty z ϵ-przejściami.* Niedeterministyczny automat z ϵ-przejściami $A = (\Sigma, Q, I, \delta, F)$ jest określony jak zwykły automat skończony z tym, że $\delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Q$. Relacja $q \stackrel{w}{\to} p$ (gdzie $p, q \in Q$ i $w \in \Sigma^*$) jest określona jak poprzednio, tzn. $q \stackrel{\epsilon}{\to} q$ i $q \stackrel{va}{\to} p$ o ile $q \stackrel{v}{\to} q_1$ i $(q_1, a, p) \in \delta$ z tym, że teraz a może być literą w Σ lub ϵ.

   Dowieść, że dla dowolnego automatu z ϵ-przejściami istnieje zwykły automat skończony, który akceptuje ten sam język.

2. *Automat z wyjściem wg Mealy'ego.* Automat Mealy'ego można przedstawić jako deterministyczny automata skończony, powiedzmy $A = (\Sigma, Q, q_I, \delta, F)$ dany razem z funkcją $\gamma : Q \times \Sigma \to \Delta$. Intuicyjnie, dany stan q i litera $\sigma \in \Sigma$ determinują nie tylko kolejny stan, powiedzmy p, ale także sygnał wyjściowy (*output*), $\gamma(q, \sigma)$. Dokładniej, słowo $w = w_1 w_2 \ldots w_n$ nad Σ, takie, że $\hat{\delta}(q_I, w_1 \ldots w_{i-1}) = p_i$, dla $i = 1, \ldots, n$, wyznacza n-literowe słowo nad alfabetem Δ,

$$\hat{\gamma}(w) =_{def} \gamma(q_I, w_1)\gamma(p_2, w_2) \ldots \gamma(p_n, w_n)$$

   Powiemy, że automat Mealy'ego redukuje język L_1 do L_2 jeśli $(\forall w)\ w \in L_1 \Leftrightarrow \hat{\gamma}(w) \in L_2$. Skonstruować automat, który redukuje $a^*b(a^*ba^*ba^*)^*$ do $(a^*ba^*ba^*)^*$.

3. *Automat z wyjściem wg Moore'a.* Automat Moore'a można przedstawić jako deterministyczny automata skończony wraz z funkcją $\gamma : Q \to \Delta$. Tym razem, słowo $w = w_1 w_2 \ldots w_n$ wyznacza

$$\hat{\gamma}(w) =_{def} \gamma(\hat{\delta}(q_I, w_1))\gamma(\hat{\delta}(q_I, w_1 w_2)) \ldots \gamma(\hat{\delta}(q_I, w_1 w_2 \ldots w_n))$$

   Dowieść, że automaty Mealy'ego i Moore'a są równoważne w tym sensie, że dla każdego automatu jednego typu można znaleźć automat drugiego typu realizujący tę samą funkcję $\hat{\gamma}$.

4. *Automaty dwukierunkowe.* Funkcja przejścia deterministycznego automatu dwukierunkowego jest postaci $\delta : Q \times \Sigma \to Q \times \{L, R\}$, co interpretujemy, że automat może przesunąć czytnik zarówno w prawo jak i w lewo. Dowieść, że deterministyczne automaty dwukierunkowe mogą być symulowane przez zwykłe automaty skończone[1]. (Rezultat zachodzi również dla automatów niedeterministycznych.)

   Uwaga. W dalszej części wykładu spotkamy się z mocniejszym stwierdzeniem, a mianowicie, że maszyny Turinga pracujące na jednej taśmie w czasie liniowym akceptują jedynie języki regularne.

5. Dla ustalonego k, rozważmy język nad alfabetem $\{a, b, c\}$ opisany wyrażeniem

$$((a+b)^*c)^{k-1}\ (a+b)^*a(a+b)^{k-1}c\ ((a+b)^*c)^*$$

   Intuicyjnie: język składa się z „bloków" liter a, b, zakończonych „znacznikiem" c, przy czym k-ty (od początku) blok ma tę własność, że jego k-ty symbol od końca jest a.

   (a) Wykazać, że deterministyczny „jednokierunkowy" automat rozpoznający ten język ma $\geq 2^k$ stanów.

   (b) Skonstruować 2-kierunkowy automat o liniowej ($\mathcal{O}(k)$) liczbie stanów, rozpoznający ten język.

   (c) (*) Jak wyżej, ale automat ma prawo wykonać tylko *jeden* nawrót. (Bez zmniejszenia ogólności, możemy założyć, że automat idzie do końca słowa, po czym wraca i kończy obliczenie na początku słowa.)

2.7 Algorytmy i złożoność

1. Zaprojektować algorytm, który dla wyrażenia regularnego β i słowa $w \in \Sigma^*$ odpowiada na pytanie, czy w należy do języka opisywanego przez β

   (a) w czasie $\mathcal{O}(|\Sigma| \cdot |\beta|^2 \cdot |w|)$,

   (b) w czasie $\mathcal{O}(|\beta| \cdot |w|)$,

   Wskazówka. Skonstruować automat niedeterministyczny równoważny wyrażeniu β i obliczyć zbiór stanów osiągalnych po słowie w. Dla oszacowania w punkcie 1b użyć automatu z ϵ-przejściami (por. zadanie 2.6. 1), w grafie którego z każdego stanu wychodzą co najwyżej dwie krawędzie.

---

[1] *Wskazówka.* Znane rozwiązanie oparte na idei ciągów skrzyżowań (ang. *crossing sequences*) można znaleźć w rozdziale 2.6 książki J.E.Hopcroft, J.D.Ullman, *Wprowadzenie do teorii automatów, języków i obliczeń*, Wydawnictwo Naukowe PWN, Warszawa 1994. Innym, być może prostszym rozwiązaniem – zaproponowanym przez Mikołaja Bojańczyka – jest uwzględnienie w stanie symulującego automatu jednokierunkowego funkcji $h : Q \to Q \cup \{\perp\}$ o następującej interpretacji: jeśli automat (dwukierunkowy) pójdzie w lewo w stanie q, to wróci w stanie $h(q)$ (być może wcale nie wróci, gdy $h(q) = \perp$).

2. Rozważamy pytanie „$w \in L[\beta]$?” jak w zadaniu 1, ale dla *uogólnionego* wyrażenia regularnego, tj. takiego, w którym dopuszczamy również operacje teorio–mnogościowe $\cap$ i $-$. Zaprojektować algorytm, który rozstrzyga to pytanie w czasie wielomianowym od $|\beta| + |w|$.

   Wskazówka. Zastosować metodę programowania dynamicznego: dla $1 \leq i \leq j \leq |w|$ sukcesywnie obliczać zbiór podwyrażeń α wyrażenia β takich, że $w[i..j] \in L[\alpha]$.

3. Konstrukcję automatu z zadań 2.4. 1 i 2.5. 1, który dla danego w rozpoznaje język $\Sigma^* w$, można przeprowadzić za pomocą następującego algorytmu. Niech $m = |w|$.

   Najpierw obliczamy pomocniczą tablicę $F[0..m]$, taką, że $F[0] = 0$ oraz $F[i]$ jest długością najdłuższego prefiksu właściwego $w[1..i]$ będącego jednocześnie sufiksem $w[1..i]$, dla $i = 1, \ldots, m$.

   $F[0] := F[1] := 0$; $i := 0$;

   <u>for</u> $j := 2$ <u>to</u> m <u>do</u>

   <u>begin</u> $(* \, i = F[j-1] \, *)$

       <u>while</u> $w_j \neq w_{i+1} \wedge i > 0$ <u>do</u> $i := F[i]$;

       <u>if</u> $w_j = w_{i+1}$ <u>then</u> $i := i + 1$;

       $F[j] := i$

   <u>end</u>

   W konstrukcji automatu przyjmujemy, że stany są liczbami $0, 1, \ldots, m$ (które można utożsamić z prefiksami w o odpowiednich długościach). Funkcję przejścia δ określamy przez

   - $\delta(0, w_1) = 1$, $\delta(0, a) = 0$, dla $a \neq w_1$,
   - $\delta(j, w_{j+1}) = j + 1$, $\delta(j, a) = \delta(F[j], a)$, dla $a \neq w_{j+1}$

   Dowieść, że powyższy algorytm oblicza żądany automat w czasie liniowym względem długości w (zależnym od alafabetu).

4. Niech A będzie ustalonym automatem deterministycznym. Zaprojektować algorytm, który dla danej liczby n znajduje w czasie $\mathcal{O}(n)$ liczbę słów długości n akceptowanych przez A.

5. Podać przykład świadczący o tym, że najkrótsze słowo, jakiego nie akceptuje automat niedeterministyczny o n stanach może mieć długość $2^{\Omega(n)}$.

   Wskazówka. Rozważyć automat akceptujący wszystko za wyjątkiem słowa

   $$(\ldots((a_0^2 a_1)^2 a_2)^2 a_3 \ldots a_{n-1})^2 a_n.$$

6. Dowieść, że jeśli dwa stany n–stanowego automatu deterministycznego nie są równoważne (tzn. $L(A, q) \neq L(A, p)$), to istnieje słowo długości nie większej niż n akceptowane z dokładnie jednego z nich.

   Wskazówka. Rozważyć zstępujący ciąg relacji równoważności na zbiorze stanów: $R_i(p, q)$ o ile stany p i q są nierozróżnialne słowami długości $\leq i$.

7. Podać algorytm rozstrzygający równoważność dwóch automatów deterministycznych.

   Uwaga. Zastosowanie idei zadania 6 prowadzi do efektywniejszego algorytmu niż testowanie niepustości automatu produktowego.

8. Czy dla słów u, v takich, że $|u| < |v| = n$ istnieje automat deterministyczny spełniający: $u \in L(A) \ \wedge v \notin L(A)$ i mający $\mathcal{O}(\log n)$ stanów?

9. (a) Wykazać, że dla dowolnego automatu niedeterministycznego o n stanach można skonstruować równoważne wyrażenie regularne długości $2^{\mathcal{O}(n)}$.

   (b) (**) Podać przykład świadczący o tym, że najkrótsze wyrażenie regularne równoważne danemu automatowi deterministycznemu o n stanach może mieć długość $2^{\Omega(n)}$.

   Wskazówka. Rozważyć automat, którego graf jest pełnym grafem o n wierzchołkach, a każda krawędź ma inną etykietę. Stosując indukcję po n, skonstruować pętlę, która „wymusza” eksponencjalną długość wyrażenia regularnego.

10. *Eksplozja stanów w produkcie automatów.* Niech $\Sigma = \{0, 1, \ldots, k\}$ i niech dla $i = 1, \ldots, k$, L_i będzie zbiorem słów v nad Σ o następującej własności:

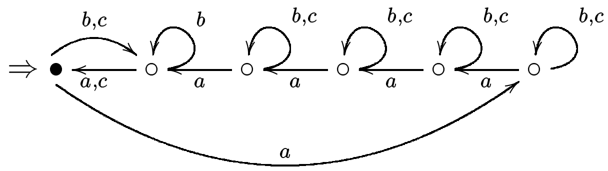
    > i występuje w v, ale przed pierwszym i pomiędzy każdymi dwoma kolejnymi wystąpieniami i, $i-1$ występuje co najmniej dwa razy.

    Nietrudno jest skonstruować deterministyczny automat o 4 stanach rozpoznający L_i. Dowieść, że każdy (nawet niedeterministyczny) automat rozpoznający język $L_1 \cap \ldots \cap L_k$ musi mieć co najmniej $2^{k+1} - 1$ stanów.

    Wskazówka. Oszacować od dołu długość najkrótszego słowa w tym języku.

11. Dowieść, że jakikolwiek automat niedeterministyczny rozpoznający język $\{xcy : x, y \in \{a, b\}^* \ \wedge \ x[1..k] = y[1..k]\}$ ma $2^{\Omega(k)}$ stanów.

12. Zakładając, że następujący automat deterministyczny A_n ma n stanów ($n \geq 2$, na rysunku $n = 6$), dowieść, że jakikolwiek automat *deterministyczny* rozpoznający lustrzane odbicie języka $L(A_n)$ ma co najmniej 2^n stanów (por. zadanie 7 na str. 8).



13. *Słowo synchronizujące.* Mówimy, że słowo w synchronizuje stany automatu deterministycznego, jeśli istnieje taki stan q_0, że startując z dowolnego stanu i czytając słowo w, automat dojdzie zawsze do stanu q_0, tzn. $(\forall q \in Q)\, q \xrightarrow{w} q_0$.

    (a) Znaleźć słowo synchronizujące dla automatu nad alfabetem $\{a, b\}$ o zbiorze stanów $\{0, 1, \ldots, k-1\}$ i funkcji przejścia

$$\begin{array}{ll} i \xrightarrow{a} i+1 \pmod{k} & \text{dla } i = 0, 1, \ldots, k-1 \\ i \xrightarrow{b} i & \text{dla i = 0,1, \ldots, k-2} \\ k-1 \xrightarrow{b} 0 & . \end{array}$$

(b) Zaprojektować algorytm, który dla automatu o n stanach rozstrzyga w czasie $\mathcal{O}(n^3)$, czy istnieje słowo synchronizujące i w pozytywnym przypadku znajduje takie słowo (długości $\leq (n-1)^3$).

(c) (*) Znaleźć *najkrótsze* słowo synchronizujące dla automatu z punktu (13a).

Uwaga. Licząca już 40 lat *hipoteza Černego*[2] głosi, że najkrótsze słowo synchronizujące, o ile istnieje, ma długość $(n-1)^2$.

14. Podać wielomianowy algorytm sprawdzania, czy dany skończony zbiór słów $C \subseteq \Sigma^*$ jest kodem (por. rozdz. 1, zadanie 5).

    Wskazówka. Skonstruować automat skończony, rozpoznający hipotetyczne słowa posiadające dwie różne faktoryzacje.

3 Języki bezkontekstowe

3.1 Gramatyki bezkontekstowe

1. Podać gramatyki bezkontekstowe generujące następujące języki:

   (a) zbiór słów nad alfabetem $\{a, b\}$, które zawierają tyle samo a co b;

   (b) zbiór słów nad alfabetem $\{a, b\}$, które zawierają dwa razy więcej a niż b;

   (c) zbiór słów nad alfabetem $\{a, b\}$ o długości parzystej, w których liczba wystąpień litery b na pozycjach parzystych jest równa liczbie wystąpień tej litery na pozycjach nieparzystych;

   (d) zbiór wyrażeń arytmetycznych nad alfabetem $\{0, 1, (,), +, \cdot\}$, które, przy zwykłej interpretacji działań dla liczb naturalnych, mają wartość 3;

   (e) zbiór wyrażeń arytmetycznych w notacji polskiej (nad alfabetem $\{0, 1, +, \cdot\}$) o wartości 4;

   (f) zbiór poprawnie zbudowanych formuł rachunku zdań ze zmienną zdaniową p i stałymi logicznymi **true**, **false** (alfabet: $\{p, \mathbf{true}, \mathbf{false}, \wedge, \vee, \neg, (,)\}$);

   (g) zbiór tych formuł z poprzedniego punktu, które przy każdym wartościowaniu zmiennej p mają wartość logiczną *prawda* (tzn. tautologii);

   (h) $\{a^i b^j c^k \;:\; i \neq j \vee j \neq k\}$;

   (i) $\{a^i b^j a^k \;:\; i + k = j\}$.

2. Dla danych gramatyk bezkontekstowych G, H, skonstruować gramatyki generujące języki $L(G) \cup L(H)$, $L(G)L(H)$, $(L(G))^*$, $(L(G))^R$ (=lustrzane odbicie).

[2]Zob. http://www.liafa.jussieu.fr/~jep/Problemes/Cerny.html.

3. Wykazać, że zbiór palindromów nad ustalonym alfabetem jak również jego dopełnienie są językami bezkontekstowymi.

4. Napisać gramatykę bezkontekstową (jak najkrótszą) generującą język:
$$L \;=\; \{\; a^i b^j \;:\; i,j \geq 1;\; i < 2j-1 \;\}$$

5. Skonstruować gramatykę bezkontekstową z jednym symbolem nieterminalnym generującą zbiór $\{x \in (a+b)^* \;:\; \#_a(x) = \#_b(x)\;\}$, gdzie $\#_s(w)$ oznacza liczbę wystąpień symbolu s w słowie w.

6. Skonstruować gramatykę bezkontekstową jednoznaczną generującą język $\mathbf{D_1}$ poprawnie uformowanych wyrażeń nawiasowych (por. Rozdział 1, Zadanie 2).

7. Napisać gramatyki bezkontekstowe generujace zbiór tych słów z języka $\mathbf{D_1}$, które

   (a) zawierają parzystą liczbę (np. 0) nawiasów otwierających,

   (b) nie zawierają podsłowa (()).

8. Skonstruować gramatykę bezkontekstową jednoznaczną generującą język z Zadania 5.

   Wskazówka.[3] Rozważyć gramatykę
$$\begin{array}{rcl} Z & \to & ZaZ^+b \,|\, ZbZ^-a \,|\, \varepsilon \\ Z^+ & \to & Z^+aZ^+b \,|\, \varepsilon \\ Z^- & \to & Z^-bZ^-a \,|\, \varepsilon \end{array}$$

   Zauważyć związek miedzy produkcjami dla Z^+ a poprzednim zadaniem (podobnie dla Z^-).

9. Dowieść, że następujące warunki są równoważne dla języka $L \subseteq \Sigma^*$:

   (a) L jest regularny,

   (b) L jest generowany przez gramatykę bezkontekstową, w której każda reguła jest postaci $X \to \epsilon$, $X \to Y$, lub $X \to \sigma Y$, $\sigma \in \Sigma$,

   (c) L jest generowany przez gramatykę bezkontekstową, w której każda reguła jest postaci $X \to \epsilon$, $X \to Y$, lub $X \to Y\sigma$, $\sigma \in \Sigma$,

   (d) L jest generowany przez gramatykę bezkontekstową, w której każda reguła jest postaci $X \to \alpha$ lub $X \to \beta Y$, $\alpha, \beta \in \Sigma^*$.

10. Podać przykład gramatyki bezkontekstowej w której każda reguła jest postaci $X \to \epsilon$, $X \to Y$, $X \to \sigma Y$ lub $X \to Y\sigma$, $\sigma \in \Sigma$, ale język generowany przez gramatykę nie jest regularny.

[3] Rozwiązanie podane przez studenta, pana Dariusza Leniowskiego.

11. Czy każdy język bezkontekstowy jest generowany przez gramatykę w postaci z poprzedniego zadania?

12. Powiemy, że gramatyka G ma własność *właściwego samozapętlenia*, jeśli dla pewnej zmiennej X zachodzi $X \stackrel{G}{\to}^{*} \alpha X \beta$, gdzie $\alpha, \beta \neq \epsilon$. Udowodnij, że gramatyka bezkontekstowa nie mająca własności właściwego samozapętlenia generuje język regularny.

13. Udowodnij, że każdy język bezkontekstowy nad jednoliterowym alfabetem jest regularny.

14. Udowodnij, że jeśli L jest bezkontekstowy to język $\{a^{|w|} \ : \ w \in L\}$ jest regularny.

15. Niech G będzie gramatyką bezkontekstową, z m zmiennymi i niech, dla każdej reguły $Y \stackrel{G}{\to} w$, $|w| \leq \ell$. Dowieść, że jeśli $X^I \stackrel{G*}{\to} \epsilon$, to istnieje wyprowadzenie o długości $1 + \ell + \ell^2 + \ldots + \ell^{m-1}$. Czy to oszacowanie jest optymalne?

16. Dowieść, że dla każdej gramatyki G istnieje stała C, taka, że dla dowolnego $w \neq \epsilon$, jeśli $X_I \stackrel{G*}{\to} w$, to istnieje wyprowadzenie o długości $\leq C \cdot |w|$.

17. Załóżmy, że mamy pewną skończoną liczbę reguł wymazujących postaci $\alpha \to \epsilon$. Stosując takie reguły możemy w danym słowie zastępować słowo α przez słowo puste. Niech L będzie zbiorem słów, które możemy przekształcić na słowo puste stosując reguły wymazywania.

    Czy zawsze istnieje gramatyka bezkontekstowa generująca L i mająca tylko jeden symbol nieterminalny ?

18. Zaprojektować algorytm, który, dla danej gramatyki G, odpowiada na pytanie, czy język $L(G)$ jest nieskończony.

19. Udowodnij, że każdy język bezkontekstowy może być generowany przez gramatykę w której każdy symbol nieterminalny (poza być może symbolem początkowym) generuje nieskończenie wiele słów terminalnych.

3.2 Bezkontekstowy czy nie? — lematy o pompowaniu

1. Udowodnić, że żaden nieskończony podzbiór języka $L = \{a^n b^n c^n \ : \ n \geq 1\}$ nie jest językiem bezkontekstowym.

2. Udowodnić, że dopełnienie języka z poprzedniego zadania jest językiem bezkontekstowym.

3. Udowodnić, że jeśli alfabet Σ ma co najmniej dwie litery, to język $L = \{ww : w \in \Sigma^*\}$ nie jest bezkontekstowy, natomiast jego dopełnienie $\Sigma^* - L$ jest językiem bezkontekstowym.

4. Dowieść, że dla każdego ustalonego k dopełnienie języka $\{w^k : w \in \Sigma^*\}$ jest językiem bezkontekstowym.

5. Udowodnić, że język $L = \{xcx : x \in (a+b)^* \}$ nie jest językiem bezkontekstowym.

6. Udowodnić, że dopełnienie języka z poprzedniego zadania jest językiem bezkontekstowym.

7. Niech $Subwords(x)$ oznacza zbiór wszystkich podsłów słowa x, $Subwords(L) = \bigcup_{x \in L} Subwords(x)$.

   (a) Udowodnić, że dla języka bezkontekstowego L, $Subwords(L)$ jest językiem bezkontekstowym.

   (b) Podać przykład języka bezkontekstowego nieregularnego L, dla którego $Subwords(L)$ jest językiem regularnym.

   (c) Podać przykład języka bezkontekstowego L, dla którego $Subwords(L)$ nie jest językiem regularnym.

8. Pokazać, że język dopasowywania wzorca $L = \{xcy : x, y \in (a+b)^*,\ y \in Subwords(x)\}$ nie jest bezkontekstowy. Czy dopelnienie tego języka jest bezkontekstowe ?

9. Pokazać, że język $L = \{xcy^R : x, y \in (a+b)^*,\ y \in Subwords(x)\}$ jest bezkontekstowy.

10. (*) Pokazać, że dopełnienie języka z poprzedniego zadania nie jest językiem bezkontekstowym.

11. Rozstrzygnać, czy następujący język jest bezkontekstowy:

$$L = \{u\$w^R : u, w \in \{a,b\}^+,\ w \text{ jest prefiksem i sufiksem } u \}$$

    To samo pytanie dla dopełnienia tego jezyka.

12. Udowodnić, że język $L = \{a^i b^j c^k : i \neq j,\ i \neq k,\ j \neq k \}$ nie jest bezkontekstowy.

    Czy jego dopełnienie jest bezkontekstowe ?

13. Czy język $L = \{a^i b^j a^i b^j : i, j \geq 1 \}$ jest bezkontekstowy ?

    Czy jego dopełnienie jest językiem bezkontekstowym ?

14. Udowodnić, że język $L = \{ww^R w : w \in (a+b)^*\}$ nie jest bezkontekstowy.

    Czy jego dopełnienie jest bezkontekstowe ?

15. Pokaż, że język $\{x\#y^R : x, y \in \{0,1\}^+,\ [x]_2 + 1 = [y]_2\}$ jest bezkontekstowy.

16. Pokaż, że język $\{x\#y : x, y \in \{0,1\}^+,\ [x]_2 + 1 = [y]_2\}$ nie jest bezkontekstowy.

17. Które z następujących języków są bezkontekstowe ?

    (a) $\{a^m b^n : m < n < 2m\}$

    (b) $(a+b)^* - \{(a^n b^n)^n : n \geq 1\}$

(c) $\{ww^Rw : w \in (a+b)^*\}$

(d) $\{a^xba^yba^z : x+y=z\}$

(e) $\{a^xba^yba^z : x \cdot y = z\}$

18. Dowiedź, że następujące języki nie są bezkontekstowe:

(a) $\{a^ib^ja^k : j = \max\{i,k\}\}$

(b) $\{a^ib^ic^k : k \neq i\}$.

19. Czy język

$$\{\mathrm{bin}(n)\,\$\,\mathrm{bin}(n^2)^R \,:\, n \in N\},$$

gdzie $\mathrm{bin}(n) \in \{0,1\}^*$ jest binarnym przedstawieniem liczby n, jest bezkontekstowy?

20. Niech $[n]_2$ oznacza zapis binarny liczby naturalnej $n \geq 1$, pierwszą cyfrą jest jedynka (najbardziej znacząca). Czy następujący język jest bezkontekstowy:

$$L_4 \;=\; \{\; [n]_2\,[2n]_2 \;:\; n \geq 1 \;\}$$

21. (a) Udowodnij, że zbiór tautologii nad ustalonym skończonym zbiorem zmiennych jest bezkontekstowy (stanowi to uogólnienie zadania (1g) z sekcji 3.1).

(b) Formuły nad przeliczalnym zbiorem zmiennych można przedstawić jako język nad skończonym alfabetem, przyjmując indeksowanie zmiennych. Dokładniej, przyjmijmy, że zbiór wszystkich formuł jest generowany przez gramatykę

$$\begin{aligned} F &\to \mathbf{true} \mid \mathbf{false} \mid V \mid (F \vee F) \mid (F \wedge F) \mid (\neg F) \\ V &\to xI \\ I &\to 0 \mid 1J \\ J &\to J0 \mid J1 \mid \epsilon \end{aligned}$$

Np. $((x101 \vee (\neg x0)) \wedge (\neg(\mathbf{false} \vee x101)))$ jest formułą.

Udowodnij, że zbiór wszystkich tautologii nie jest bezkontekstowy[4].

22. Niech alfabetem będzie $\{0,1\}$. *Słowo Lyndona* to słowo pierwotne (nie będące potęgą mniejszego słowa, por. rozdz. 1, zad. 1), które jest leksykograficznie najmniesze spośród swoich przesunięć cyklicznych.

Czy zbiór słów Lyndona jest bezkontekstowy ?

Wskazówka: weźmy $a^{n+1}ba^nba^n$ i zastosujmy lemat Ogdena, „markując" środkową grupę a^n.

23. Niech PAL onacza zbiór palindromów. Czy zachodzi implikacja:

$$\text{L bezkontekstowy} \quad\Longrightarrow\quad L \cap PAL \text{ bezkontekstowy ?}$$

[4]Stwierdzenie to wynika łatwo z hipotezy P $\neq$ NP, ale należy je dowieść bez tej hipotezy.

24. Sformuluj $uvwxy$-lemat o pompowaniu dla języków liniowych w którym $|uvxy| = O(1)$.

    Dowiedź, że język $\{a^i b^i c^j d^j : i,j \in \mathbf{N}\}$ nie jest liniowy.

25. Dowiedź, że $L = \{x \in (a+b)^*, \#_a(x) = \#_b(x)\}$ nie jest liniowym językiem bezkontekstowym.

26. Udowodnij, że zbiór słów w nad alfabetem $\{a, b\}$ mających tę samą liczbę liter a co b nie jest liniowym językiem bezkontekstowym.

27. Który z następujących języków jest bezkontekstowy. W przypadku gdy język jest bezkontekstowy wypisać gramatykę bezkontekstową. W zadaniu $[x]_2$ oznacza binarny zapis liczby x, oraz w^R oznacza odwrócenie słowa w.

    (a) $L_1 = \{[x]_2 \;\&\; [y]_2 \;:\; 1 \le x \le y\}$.
    (b) $L_2 = \{[x]_2 \;\&\; [y]_2^R \;:\; 1 \le x \le y\}$.

28. Niech $\mathbf{D_1}, \mathbf{D_2}$ oznacza zbiór poprawnych ciągów nawiasowych jednego typu (*okrągłe*) i dwóch typów nawiasów (*okrągłe* i *kwadratowe*), odpowiednio, włącznie ze słowem pustym. Czy następujące języki są bezkontekstowe

    (a) $\{\, u\#v^R \;:\; uv \in \mathbf{D_1} \,\}$,
    (b) $\{\, u\#v^R \;:\; uv \in \mathbf{D_2} \,\}$?

29. Niech L będzie zbiorem słów w alfabecie trzyelementowym, które nie zawierają słowa postaci xx, dla $x \neq \epsilon$. Udowodnić, że L nie jest bezkontekstowy.

30. Podać przykład języka bezkontekstowego nad alfabetem trzyelemnentowym, którego dopełnienie jest nieskończone i nie zawiera słów postaci xx, dla $x \neq \epsilon$.

3.3 Automaty ze stosem

1. Skonstruować automaty ze stosem rozpoznające poznane wcześniej języki bezkontekstowe: zbiór palindromów, zbiór poprawnie uformowanych ciągów nawiasów, zbiór słów, które mają dwa razy więcej b niż a, zbiór ciągów, które nie są postaci ww.

2. Dla liczby naturalnej n, niech $\mathrm{bin}(n) \in \{0,1\}^*$ będzie binarnym przedstawieniem liczby n. Skonstruować automat ze stosem rozpoznający język

$$\{\mathrm{bin}(n)\,\$\,\mathrm{bin}(n+1)^R \;:\; n \in N\}$$

3. Skonstruować automat ze stosem rozpoznający język

$$\{\mathrm{bin}(n)\,\$\,\mathrm{bin}(3*n)^R \;:\; n \in N\}$$

   Uogólnić tezę zadania[5].

[5] Można zacząć od przykładu $\{\mathrm{dec}(n)\,\$\,\mathrm{dec}(2006*n)^R \;:\; n \in N\}$, gdzie $\mathrm{dec}(n) \in \{0,1,\ldots,9\}^*$ jest dziesiętnym przedstawieniem liczby n.

4. Dowieść, że dla każdego automatu ze stosem A, można skonstruować automat ze stosem *o dwóch stanach* A', taki że $L(A) = L(A')$.

5. Dowieść, że automatowi A' z poprzedniego zadania można postawić dalszy wymóg, że każde przejście jest postaci

$$q, a, Z \rightarrow_{A'} q', \alpha$$

gdzie $|\alpha| \leq 2$ (q, q' dowolne).

6. Dowieść, że dla każdego automatu ze stosem A, można skonstruować równoważny mu automat ze stosem A'', w którym każde przejście jest postaci *push* lub *pop*, tzn.

$$q, a, Z \rightarrow_{A''} q', YZ$$

lub

$$q, a, Z \rightarrow_{A''} q', \epsilon$$

Czy dla takich automatów można nadal ograniczyć liczbę stanów?

7. Mając dany automat ze stosem akceptujący język L, skonstruować automaty ze stosem akceptujące następujące języki:

   - $\text{Prefix}(L) = \{w \,:\, (\exists v) wv \in L\}$
   - $\text{Suffix}(L) = \{w \,:\, (\exists u) uw \in L\}$
   - $\text{Subword}(L) = \{w \,:\, (\exists u, v) uwv \in L\}$
   - $L^R = \{w^R \,:\, w \in L\}$
   - (*) $\text{Cycle}(L) = \{vw \,:\, wv \in L\}$

8. Mając dany automat ze stosem akceptujący język L i automat skończony akceptujący język R, skonstruować automaty ze stosem akceptujące następujące języki:

   - LR^{-1}
   - $R^{-1}L$
   - $L \cap R$

9. Niech $\#_s(w)$ oznacza liczbę symboli s w słowie w, oraz $PREF(u)$ oznacza zbiór prefiksów słowa u. Ponadto oznaczmy przez $max(w)$, $min(w)$, $med(w)$ opowiednio maksimum, minimum i medianę liczb $\#_a(w), \#_b(w), \#_c(w)$.

   Który z następujących języków jest regularny, a który bezkontekstowy?

   (a) $L_1 = \{u \in (a \cup b \cup c)^* \,:\, \forall\, w \in PREF(u)\ max(w) - min(w) \leq 13\ \}$.

   (b) $L_2 = \{u \in (a \cup b \cup c)^* \,:\, \forall\, w \in PREF(u)\ max(w) - med(w) \leq 13\ \}$.

10. Dla danych językow regularnych L i M, skonstruować automat ze stosem rozpoznający język $\bigcup_{i \in N}(L^i \cap M^i)$. Uwaga: ten zbiór nie musi być regularny.

11. Dowieść, że dla każdego automatu ze stosem A istnieje stała C (zależna od automatu), taka, że dla każdego słowa $w \in Z(A)$, istnieje obliczenie akceptujące (przez pusty stos) o długości $\leq C|w|$. Wskazówka: oszacować wysokość stosu w obliczeniu akceptującym.

12. Niech A będzie automatem ze stosem. Dowieść, że zbiór słów, które są możliwymi zawartościami stosu automatu A, jest językiem regularnym. Formalnie, mamy na myśli zbiór

$$\{\alpha \in \Gamma^* \,:\, (\exists w, v \in \Sigma^*)(\exists q \in Q)\, q_I, w, Z_I \vdash_A q, v, \alpha\}$$

Wywnioskować stąd, że zbiór słów, które są możliwymi zawartościami stosu automatu A w jakimś obliczeniu *akceptującym*, jest językiem bezkontekstowym.

13. Automat ze stosem $A = (\Sigma, \Gamma, Q, q_I, Z_I, \delta, F)$ nazywamy *deterministycznym*, jeśli w każdej sytuacji możliwy jest co najwyżej jeden ruch. Dokładniej:

    - jeśli, dla pewnej pary q, Z, zachodzi $q, \epsilon, Z \rightarrow_A p, \alpha$ przy pewnych p, α, to dla żadnego $\sigma \in \Sigma$, nie zachodzi $q, \sigma, Z \rightarrow_A p', \alpha'$, dla żadnych p', α';
    - dla każdych q, σ, Z, istnieje co najwyżej jedna para p, α, taka że $q, \sigma, Z \rightarrow_A p, \alpha$.

    Język bezkontekstowy jest *deterministyczny* jeśli jest rozpoznawany przez pewien deterministyczny automat ze stosem (w sensie stanów akceptujących, tzn. $L = L(A)$).

    Dowieść, że język $\{a^n b^n : n \in \mathbf{N}\} \cup \{a^n b^{2n} : n \in \mathbf{N}\}$ nie jest deterministycznym językiem bezkontekstowym.

    Wskazówka.[6] Zakładając, że istnieje deterministyczny automat A rozpoznający ten język, rozważyć automat A', który różni się od A tylko tym, że zamienia rolami a i b. Przy pomocy tych dwóch automatów można łatwo skonstruować automat ze stosem rozpoznający język $\{a^n b^n a^n : n \in \mathbf{N}\}$.

14. Wykazać, że zbiór palindromów nad alfabetem dwuelementowym nie jest deterministycznym językiem bezkontekstowym.

    Wskazówka. Zastosować metodę z poprzedniego zadania. Przy pomocy hipotetycznego automatu deterministycznego rozpoznającego palindromy można np. skonstruować automat rozpoznający język $\{ca^n ba^n ca^n ba^n c : n \in \mathbf{N}\}$.

3.4 Własności języków bezkontekstowych

1. Podaj przykład języka bezkontekstowego L t.że język $L = \{x : (\exists y)\, |x| = |y| \wedge xy \in L\}$ nie jest bezkontekstowy.

2. Udowodnić, że przecięcie języka (deterministycznego) bezkontekstowego z regularnym jest też językiem (deterministycznym) bezkontekstowym.

---
[6] Rozwiązanie podane przez studenta, pana Adama Wilczyńskiego.

3. Przeplotem słów w i v nazwiemy dowolne słowo długości $|w|+|v|$, które można rozbić na rozłączne podciągi w i v. L i M jest zbiór wszystkich możliwych przeplotów słów $w \in L$, $v \in M$. Język ten oznaczamy $L \parallel M$. Udowodnij, ze przeplot języka bezkontekstowego i regularnego jest językiem bezkontekstowym. Podaj przykład języków bezkontekstowych L i M, dla których $L \parallel M$ nie jest językiem bezkontekstowym.

4. Domknięcie przeplotne języka L określamy przez $L^\sharp = L \cup (L \parallel L) \cup (L \parallel L \parallel L) \cup \ldots$. Wykazać, że operacja domknięcia przeplotnego języka skończonego może dać w wyniku język który nie jest bezkontekstowy.

5. Udowodnić, że jeśli X, Y są językami regularnymi to język
$$L \;=\; \sum_{n\geq 1} X^n \cap Y^n$$
jest bezkontekstowy.

6. Podać przykład języków regularnych X, Y t.że
$$\sum_{n\geq 1}(X^n \cap Y^n) \;=\; \{w \in (a^+b^+)^+ : \#_a(w) = \#_b(w)\}$$

7. Podaj języki regularne X, Y, Z t.że język
$$\sum_{n\geq 1}(X^n \cap Y^n \cap Z^n)$$
nie jest bezkontekstowy.

8. Wskaż język bezkontekstowy L, dla którego $\sqrt{L} = \{w : ww \in L\}$ nie jest językiem bezkontekstowym.

9. Podaj przykład języka bezkontekstowego takiego. że $\{x \;:\; x^k \in L$ dla pewnego $k\}$ nie jest bezkontekstowy.

10. Rozważmy następujący morfizm:
$$h(a) = a,\; h(b) = b,\; h(a') = a,\; h(b') = b$$
oraz $h(x) = \epsilon$ dla symboli $x \in \{a, b, a', b'\}$ dla których morfizm nie został zdefiniowany powyżej. Wykazać, że język $EQ(h, g) = \{w : h(w) = g(w)\}$ nie jest bezkontekstowy.

11. Udowodnij, że jeśli U jest regularny to następujący język jest bezkontekstowy
$$\{xy^R \;:\; x \neq y,\; xy \in U\}$$

12. Język ma własność prefiksową, gdy dla każdych dwóch słow z tego języka jedno z nich jest prefiksem drugiego. Wykaż, że jeśli język bezkontekstowy ma własność prefiksową to jest on regularny.

13. Niech $L \subseteq \{a, b\}^*$ będzie językiem regularnym oraz h, g będą morfizmami. Udowodnić, że następujący język jest liniowym językiem bezkontekstowym
$$\{h(u)c(g(u))^R \;:\; u \in L\}$$

14. Udowodnij, że przecięcie liniowego języka bezkontekstowego z regularnym jest językiem liniowym.

15. Dla języka L określamy Min(L) jako zbiór słów w L minimalnych ze względu na porządek bycia prefiksem. (Zatem $u \in \text{Min}(L) \Leftrightarrow$ nie istnieją $v \in L$ oraz $w \neq \epsilon$ takie, że $vw \in L$.) Udowodnij, że dla deterministycznego języka bezkontekstowego L język Min(L) jest również deterministycznym językiem bezkontekstowym.

16. Niech $L = \{a^i b^j c^k \ : \ k \geq i \ lub \ k \geq j\}$. Pokaż, że $Min(L)$ nie jest językiem bezkontekstowym.

17. Zbiór Max(L) określamy analogicznie jak zbiór Min(L) w zadaniu 15. Podaj przykład języka bezkontekstowego L, dla którego Max(L) nie jest językiem bezkontekstowym.

18. Niech h_1, h_2 będą morfizmami takimi, że alfabet wyjściowy nie zawiera \$. Wykaż, że języki $\{x\$y^R \ : \ h_1(x) = h_2(x)\}$ i $\{x\$y^R \ : \ h_1(x) \neq h_2(x)\}$ są liniowymi językami bezkontekstowymi.

19. Podzbiór $M \subseteq \mathbf{N}^k$ nazywamy *liniowym* jeśli można go przedstawić $M = \{\vec{a} + n \cdot \vec{b} : n \in \mathbf{N}\}$, dla pewnych wektorów $\vec{a}, \vec{b} \in \mathbf{N}^k$, a *semi-liniowym*, jeśli jest sumą skończenie wielu zbiorów semi-lniowych.

    (a) Udowodnij, że zbiór długości słów języka bezkontekstowego jest semi-liniowy.

    (b) (*) Udowodnij *twierdzenie Parikha* głoszące, że dla języka bezkontekstowego $L \subseteq \Sigma^*$ zbiór ($\subseteq \mathbf{N}^{|\Sigma|}$) wektorów liczby wystąpień liter z Σ w słowach z L jest semiliniowy.

20. Przypomnijmy pojęcie odległości Hamminga, określonej dla słów tej samej długości (zadanie 2.3.17)
$$d(u, w) = |\{i : w_i \neq v_i\}|.$$
Udowodnić, że dla języka regularnego L, język
$$\{ \ w \ : \ (\exists \ u \in L) \ |u| = |w| \text{ oraz } d(u, w) \leq \frac{|u|}{2} \ \}$$
jest bezkontekstowy. Czy język ten jest zawsze regularny ?

21. Czy następujące stwierdzenia są prawdziwe:

    (a) Istnieje nieskończony zbiór słów L nad skończonym alfabetem taki, że ani L ani dopełnienie L nie zawierają nieskończonego języka regularnego.

    (b) To samo, ale wymagamy dodatkowo, aby L był bezkontekstowy.

4 Teoria obliczeń

4.1 Maszyny Turinga

1. Skonstruować maszynę Turinga obliczającą funkcję 2^n reprezentowaną *unarnie*. Bardziej dokładnie, zakładamy, że alfabet wejściowy składa się z jednego symbolu, powiedzmy $\{1\}$. Jeśli na wejściu dany jest ciąg n jedynek, (tzn. konfiguracja wejściowa jest $q_0 1^n$), to po wykonaniu obliczenia konfiguracja powinna być $q_f 1^{2^n}$.

2. Skonstruować maszynę Turinga obliczającą funkcję $\lceil \log n \rceil$ reprezentowaną *unarnie.*

3. Przyjmijmy $\Sigma = \{0, 1\}$. Skonstruować maszyny Turinga rozpoznające następujące języki:

   (a) zbiór palindromów

   (b) $\{w\$w \,:\, w \in \Sigma^*\}$

   (c) $\{ww \,:\, w \in \Sigma^*\}$

   (d) zbiór ciągów reprezentujących binarnie liczby pierwsze.

4. Graf zorientowany o n wierzchołkach ponumerowanych $0, 1, \ldots, n-1$, reprezentujemy ciągiem zer i jedynek długości n^2, takim, że k-ty bit jest 1 o ile istnieje krawędź z wierzchołka o numerze i do wierzchołka o numerze j, gdzie $k = i \cdot n + j + 1$.

   (a) Skonstruować *niedeterministyczną* maszynę Turinga rozpoznającą zbiór słów zero-jedynkowych, które w powyższy sposób reprezentują te grafy, w których istnieje ścieżka z wierzchołka o numerze 0 do wierzchołka o numerze $n-1$.

   (b) Skonstruować maszynę *deterministyczną* realizującą to samo zadanie.

5. Przypomnijmy, że dwie maszyny Turinga o tym samym alfabecie wejściowym uważamy za *równoważne* o ile akceptują ten sam język. Udowodnić, że dla każdej maszyny Turinga istnieje równoważna jej maszyna posiadająca dokładnie jeden stan akceptujący i taka, że w konfiguracji akceptującej głowica znajduje się nad pierwszą komórką taśmy (tzn. konfiguracja akceptująca jest postaci q_f coś).

6. Dla danej *niedeterministycznej* maszyny Turinga skonstruować równoważną maszynę *deterministyczną.*

7. Dla danych deterministycznych maszyn Turinga M_1 i M_2 skonstruować deterministyczne maszyny rozpoznające języki

   - $L(M_1) \cup L(M_2)$
   - $L(M_1) \cap L(M_2)$
   - $L(M_1)L(M_2)$
   - $L(M_1)^*$.

8. Udowodnić, że dla każdej maszyny Turinga nad alfabetem wejściowym $\{0, 1\}$, istnieje równoważna jej maszyna, której alfabet *wszystkich* symboli roboczych obejmuje jedynie symbole 0,1,B.

9. Powiemy, że maszyna Turinga jest *write-once* jeśli może pisać tylko w pustych komórkach taśmy, a symbol raz napisany nie może być zastąpiony żadnym innym symbolem (w szczególności nie może być "zmazany" tj. zastąpiony przez "blank"). Dla dowolnej maszyny Turinga z jedną taśmą, skonstruować maszynę *write-once* z dwiema taśmami, która akceptuje ten sam język.

   (*) Dowieść, że maszyny typu *write-once* z *jedną* taśmą akceptują jedynie języki regularne.

10. Dla dowolnej maszyny Turinga (nad dowolnym alfabetem), skonstruować równoważną maszyne o *jednej taśmie* i *czterech* stanach. (Wolno powiększyć alfabet roboczy.)

11. Pojęcie automatu ze stosem można rozszerzyć do automatu z k stosami, $k \geq 2$. Dowieść, że dla każdej maszyny Turinga istnieje równoważny jej deterministyczny automat z dwoma stosami. Wywnioskować stąd, że automat z k stosami, $k > 2$ może być symulowany przez automat z 2 stosami (symulację tę można również opisać bezpośrednio).

12. Udowodnić, że jeśli automat skończony wyposażyć dodatkowo w kolejkę, to otrzymany model ma siłe obliczeniową maszyny Turinga, tzn. dla dowolnej maszyny istnieje automat z kolejką, który rozpoznaje ten sam język.

13. Automat z k licznikami $c_1, \ldots, c_k$, określamy podobnie jak automat z k stosami, z tym, że liczniki zawierają liczby naturalne, a dostępne operacje na licznikach są postaci $c_i := c_i + 1$, $c_i := c_i \dot{-} 1$ (gdzie $0 \dot{-} 1 = 0$), oraz test $c_i \stackrel{?}{=} 0$. Tzn. przejścia takiego automatu są postaci $q \to p$, $q, c_i \stackrel{?}{=} 0 \to p$, $q, c_i \stackrel{?}{\neq} 0 \to p$, $q \to p, c_i := c_i + 1$, $q \to p, c_i := c_i \dot{-} 1$. Nie ma taśmy, lecz zakładamy, że w chwili początkowej dana wejściowa jest wartością licznika c_1, a pozostałe liczniki mają wartość 0. Dowieść najpierw, że dla każdej maszyny Turinga nad alfabetem $\{1\}$ istnieje równoważny jej automat z 4 licznikami. Następnie wykazać, że liczba liczników może być dalej zredukowana do trzech. (Ograniczenie alfabetu maszyny Turinga nie jest istotne, bo słowa nad alfabetem n–literowym można również łatwo "przerobić" na liczby naturalne w systemie unarnym.)

4.2 Obliczalność i nierozstrzygalność

1. Udowodnić równoważność następujących warunków

   - język L jest częściowo obliczalny,
   - język L jest dziedziną pewnej funkcji częściowo obliczalnej,
   - język L jest zbiorem wartości pewnej funkcji częściowo obliczalnej,
   - język L jest zbiorem wartości pewnej funkcji obliczalnej.

2. Udowodnić, że język L jest obliczalny, wtedy i tylko wtedy, gdy jest skończony lub jest zbiorem wartości pewnej funkcji obliczalnej rosnącej.

3. Udowodnić następujące twierdzenie Turinga–Posta : jeśli zarówno język jak i jego uzupełnienie są częściowo obliczalne, to są również obliczalne.

4. Udowodnić, że istnieje zbiór częściowo obliczalny, którego uzupełnienie nie zawiera żadnego nieskończonego zbioru częściowo obliczalnego.

5. Udowodnić, że istnieje automat z dwoma (*sic !*) licznikami A taki, że nie jest rozstrzygalne, czy A zatrzymuje się dla danej wejściowej n. Wskazówka : wykorzystać zadanie 13.

6. Udowodnić nierozstrzygalność następującego *problemu Posta.*

   Dane są dwie listy słów : $u_1, \ldots, u_n \in \Sigma^*$ i $w_1, \ldots, w_n \in \Sigma^*$. Pytanie, czy istnieje ciąg indeksów $i_1, \ldots, i_m \in \{1, \ldots, n\}$, taki, że $u_{i_1} \ldots u_{i_n} = w_{i_1} \ldots w_{i_n}$? Jeśli taki ciąg istnieje, to słowo $u_{i_1} \ldots u_{i_n}$ $(= w_{i_1} \ldots w_{i_n})$ nazywamy *rozwiązaniem* danej instancji problemu Posta.

   Wskazówka : Rozważyć zmodyfikowany problem Posta, w którym dodatkowo wymaga się, by pierwszymi słowami w rozwiązaniu były u_1 i w_1 (tzn. $i_1 = 1$). Redukcja zmodyfikowanego problemu do oryginalnego problemu Posta nie jest trudna. Z kolei przedstawić algoerytm, który dla dowolnej maszyny Turinga M i jej wejścia w konstruuje instancję P zmodyfikowanego problemu Posta w ten sposób, że P ma rozwiązanie wtedy i tylko wtedy, gdy M akceptuje w. Rozwiązaniem P (o ile istnieje) będzie właśnie akceptujące obliczenie M na w.

7. Dowieść, że następujące problemy są nierozstrzygalne :

   - Dana gramatyka bezkontekstowa G nad alfabetem Σ. Pytanie : czy $L(G) = \Sigma^*$? (Tzw. problem *uniwersalności.*) Wskazówka : Dla maszyny Turinga M, *udane obliczenie* jest ciągiem $c_0\#c_1\#\ldots\#c_f$, gdzie c_i są konfiguracjami M, c_0 jest konfiguracją początkową, c_f akceptującą, oraz $c_i \vdash_M c_{i+1}$. Należy przedstawić algorytm, który dla dowolnej maszyny M konstruuje gramatykę G_M generującą wszystkie ciągi, które nie są udanymi obliczeniami. A zatem, gdybyśmy potrafili rozstrzygać problem uniwersalności, moglibyśmy również rozstrzygać problem niepustości ($L(M) \neq \emptyset$?) dla maszyn Turinga, co jest niemożliwe (na mocy twierdzenia Rice'a).
   - Dane dwie gramatyki bezkontekstowe G_1, G_2. Pytanie : czy $L(G_1) \cap L(G_2) \neq \emptyset$? Wskazówka : Podobnie jak poprzednie zadanie, ale należy wykorzystać operację lustrzanego odbicia.
   - Dana gramatyka bezkontekstowa G. Pytanie : czy G jest jednoznaczna (tzn. dla każedego słowa w $L(G)$ istnieje dokładnie jedno drzewo wyprowadzenia).

     Wskazówka : wykorzystać problem Posta.

8. Czy następuący problem X jest rozstrzygalny:
   Dana gramatyka bezkontekstowa G, sprawdzić czy $L(G)$ zawiera palindrom.

9. Załóżmy, że wiemy że jednotasmowa deterministyczna maszyna Turinga M wykonuje zawsze co najwyzej jeden "nawrót" (to znaczy glowica przesuwa sie w prawo a od pewnego momentu już tylko w lewo). Czy problem "$w \in L(M)$" jest rozstrzygalny, gdzie w jest słowem wejściowym a $L(M)$ oznacza zbiór słów akceptowanych przez M stanem akceptującym.

10. Czy następujący problem jest rozstrzygalny. Dane są dwa słowa $u, w \in \Sigma^*$ oraz liczba k, sprawdzić czy

    $$(\exists\ x \in \Sigma^*)\ (\ |x| > k\ \&\ \mathit{Ile}(w,x) = \mathit{Ile}(u,x)\ )$$

    gdzie $\mathit{Ile}(w,x)$ oznacza liczbę wystąpień słowa w jako podsłowo x (wystąpienia nie muszą być rozłączne).

11. Niech $bin(n)$ oznacza zapis binarny liczby naturalnej n. Mamy $|bin(n)| = \lfloor \log_2 n \rfloor + 1$ z wyjątkiem liczby 0, dla której $|bin(0)| = 1$. Określić obliczalną różnowartościową funkcję pary $\{0,1\}^* \times \{0,1\}^*$, tak by

$$|(x,y)| = |x| + |y| + \min\{|bin(|x|)|, |bin(|y|)|\} + \mathcal{O}(1)$$

12. Ustalamy pewne kodowanie maszyn Turinga, które pozwala nam utożsamiać maszynę M z pewnym słowem nad $\{0,1\}$. Rozważmy funkcję C, która parze (M,v) przyporządkowuje minimalną długość słowa x, takiego że

$$M(x) = v$$

lub specjalny symbol ∞, gdy nie ma takiego x.

   (a) Dowieść, że funkcja C nie jest obliczalna.

   Wskazówka. W przeciwnym razie obliczalna byłaby również funkcja, która liczbie n danej w postaci binarnej przyporządkowuje pewne słowo $w \in (0+1)^n$, takie że

$$\min\{|(M,x)| : M(x) = w\} \geq n$$

   (długość pary w rozumieniu za zadania 11). Biorąc za M maszynę obliczajacą tę funkcję oraz $bin(n)$ za x, otrzymujemy sprzeczność.

   (b) Dowieść, że funkcja C z punktu 12a moze być przybliżana w następującym sensie. Istnieje maszyna Turinga, która dla wejścia (M,v) produkuje nieskończony ciąg liczb, który od pewnego miejsca ustala się na wartości $C(M,v)$.

   Uwaga. Nie ma sprzeczności pomiędzy punktami 12a i 12b, ponieważ odbiorca powyższego ciągu nigdy nie wie, czy jest on już ustalony.

4.3 Hierarchia Chomsky'ego

1. Dowieść, że każdą gramatykę *monotoniczną*, czyli o regułach $\alpha \to \beta$, gdzie $|\beta| \geq |\alpha|$, można sprowadzić do postaci, w której każda reguła ma formę

$$\alpha X \beta \to \alpha \gamma \beta$$

gdzie $X \in V$, $\alpha, \beta, \gamma \in (\Sigma \cup V)^*$, $\gamma \neq \varepsilon$ (V – zbiór symboli nieterminalnych, Σ – zbiór symboli terminalnych).

Uwaga. Przy powyższym sprowadzeniu trzeba na ogół powiększyć zbiór symboli nieterminalnych.

2. Dowieść, że języki kontekstowe, to dokładnie języki rozpoznawane przez niedeterministyczne *automaty liniowo ograniczone*.

Uwaga. Automaty liniowo ograniczone są określone podobnie jak maszyny Turinga, z tym, że jedyną dostępną pamięcią są komórki taśmy zajęte na początku przez słowo wejściowe (przy czym koniec słowa jest zaznaczony specjalnym markerem).

3. Dowieść, że iloraz języka kontekstowego przez język regularny może nie być językiem obliczalnym.

   Wskazówka. Wykorzystać język obliczeń maszyny Turinga, która akceptuje język nieobliczalny.

   Wywnioskować dalej, że ani języki kontekstowe, ani języki obliczalne nie są zamknięte na ilorazy przez języki regularne.

4. Dowieść, że języki częściowo obliczalne **są** zamknięte na ilorazy przez języki częściowo obliczalne.

5. Które z czterech klas hierarchii Chomsky'ego są zamknięte na operacje $\cup, \cap, -$?

4.4 Złożoność obliczeniowa

1. *Konstruowalność pamięciowa.* Skonstruować maszynę Turinga *off-line*, która dla wejścia w o długości n zużywa dokładnie $f(n)$ komórek pamięci roboczej, gdzie

$$f(n) = 2n, n^2, n^k, 2^n, \log n, \ldots$$

2. *Konstruowalność czasowa.* Skonstruować maszynę Turinga, która dla wejścia w o długości n wykonuje dokładnie $f(n)$ kroków, gdzie $f(n) = 2n, 3n, n^2, n^k, 2^n, 2^{2^n}, \ldots$.

3. Dowieść, że klasy P, NP i $PSPACE$ są zamknięte na operację $*$.

4. *Problem domina.* Instancja problemu domina obejmuje liczbę naturalną M daną unarnie oraz skończony zbiór *wzorców kostek domina*, tzn. wektorów postaci (*up, down, left, right*), gdzie *up, down, left* i *right* są binarnie zadanymi liczbami, które w tym kontekście nazywamy *kolorami*. (Liczba kolorów nie jest ograniczona, a więc zależy od danej instancji problemu.) Pytanie brzmi: czy kwadrat $M \times M$ można pokryć kostkami domina w ten sposób, że sąsiednie kostki stykają się bokami o tym samym kolorze, a na bokach kwadratu jest kolor 0 ("biały"). Zakładamy przy tym, że każdy wzorzec można wykorzystać dowolnie wiele razy, ale nie można go "obracać" (tzn. *up* jest zawsze górną krawędzią itd.).

   Należy udowodnić, że problem domina jest NP-zupełny. Wskazówka : zastosować redukcję generyczną, tzn. wychodząc od dowolnej niedeterministycznej maszyny Turinga działającej w czasie wielomianowym.

5. Udowodnić, że następujący problem *dokładnego pokrycia* (ang. *exact cover*) jest NP-zupełny. Dana rodzina podzbiorów $\{1, 2, \ldots, n\}$ (liczby dane binarnie). Pytanie: czy istnieje podrodzina podzbiorów rozłącznych, które w sumie dają $\{1, 2, \ldots, n\}$.

   Wskazówka : wykorzystać zadanie 4.

6. Udowodnić, że następujący problem *plecakowy* (ang. *knapsack problem*) jest NP-zupełny. Dane: liczby $n, m_1, \ldots, m_k$ (binarnie). Pytanie: czy istnieje podzbiór zbioru $\{m_1, \ldots, m_k\}$ taki, że $m_1 + \ldots + m_k = n$.

   Wskazówka : wykorzystać zadanie 5.

7. Dowieść, że złożenie dwóch funkcji obliczalnych w pamięci (przestrzeni) logarytmicznej jest funkcją obliczalną w pamięci logarytmicznej.

8. Dowieść, że każdy problem z klasy *NP* redukuje się do problemu *3-CNF SAT* w pamięci logarytmicznej.