Języki, Automaty i Obliczenia

Zadania

Część pierwsza

Damian Niwiński i Wojciech Rytter

4 października 2006

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 $\{a+b_1n_1+\ldots+b_kn_k \,:\, n_1, \ldots, n_k \in \mathbf{N}\}$, 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 gralibу 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).

   Podać wielomianowy algorytm sprawdzania, czy dany skończony zbiór C jest kodem.

   Wskazówka. Można rozważyć graf, którego wierzchołkami są sufiksy słów z C, a krawędź z v do u prowadzi wtedy, gdy $(\exists w \in C)\, w = vu$. Wtedy rozwiązanie naszego problemu można wydedukować z przeszukania tego grafu.

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

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

   i

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

   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

$$\mathit{Cycle}(L) = \{vu \ : \ uv \in L\}$$

   Czy z faktu, że język $\mathit{Cycle}(L) = \{vu \ : \ uv \in L\}$ 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:

   $\{w \ : \ w \in L \wedge$ spośród słów o dlugości $|w|$, w jest najmniejsze w porządku leksykograficznym$\}$

5. Przyjmujemy, że każde niepuste słowo binarne $w \in \{0,1\}^*$, $w = w_1 \ldots w_k$, 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

$$L_r = \{w : \mathrm{bin}(w) \leq r\}$$

Dowieść, że język L_r jest regularny wtedy i tylko wtedy, gdy liczba r jest wymierna.

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

- $\frac{1}{2}L =_{df} \{w : (\exists u)\, |u| = |w| \wedge wu \in L\}$
- $\sqrt{L} =_{df} \{w : ww \in L\}$

7. 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.

$$F_1 = F_2 = 1$$
$$F_{n+2} = F_n + F_{n+1}$$

8. Dowieść, że dla dowolnego języka regularnego L, język

$$\{w : w^{|w|} \in L\}$$

jest również regularny.

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

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

10. *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?

11. 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.

12. 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.

13. 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.

14. 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. 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.

4. 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.

5. 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.

6. 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).

7. 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 $v \xrightarrow{a} u$, 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 $v \xrightarrow{a} u$, gdzie $|v| > |u| > 0$, jest nie większa niż $|w|$. Zauważmy, że daje to możliwość reprezentacji automatu o rozmiarze proporcjonalnym do $|w|$ (przejścia postaci $v \xrightarrow{a} \epsilon$ 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 $q \xrightarrow{w} p$ (gdzie $p, q \in Q$ i $w \in \Sigma^*$) jest określona jak poprzednio, tzn. $q \xrightarrow{\epsilon} q$ i $q \xrightarrow{va} p$ o ile $q \xrightarrow{v} q_1$ 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. (Rezultat zachodzi również dla automatów niedeterministycznych.)

   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[1] 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 $h(q) = \perp$).

[1] Zaproponowanym przez Mikołaja Bojańczyka.

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.

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.

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. (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.

9. *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.

10. 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.

11. *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. $(\forall q \in Q)\, q \stackrel{w}{\to} q_0$.

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

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

(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 (11a).

Uwaga. Licząca już 40 lat *hipoteza Černego* głosi, że najkrótsze słowo synchronizujące, o ile istnieje, ma długość $(n-1)^2$ (zob. http://www.liafa.jussieu.fr/~jep/Problemes/Cerny.html).