

# Rozwiązania niektórych z około dwustu łatwych zadań z języków formalnych i złożoności obliczeniowej (i jednego nie aż tak trudnego, jak się o nim mówi)

Numeracja zadań jak w zbiorze z 2017 roku

Wrocław, 16 kwietnia 2017

## 1 Wskazówki

Patrząc perspektywicznie w stronę zbliżających się egzaminów, ta sekcja wydaje mi się ważniejsza i zachęcam do spędzania z nią czasu podczas samodzielnych wieczornych rozmyślań.

### 1.1 Języki i automaty

**Zadanie 63.** *Warto zastanowić się, jak miałby wyglądać niedeterministyczny automat ze stosem rozpoznający taki język. Pomocny może się też okazać pewien lemat.*

#### 1.1.1 Języki rodzynekowe

**Zadanie 71.** *Problem sprowadza się do rozpoznawania liter języka  $L^*$ . Gdyby miał istnieć taki język  $L$ , to pewnie musiałby być nieskończony (dlaczego?). Może istnieje jakaś prosta, nieskończona rodzina słów postaci  $a^*ba^*$ , którą automat ze stosem może łatwo rozpoznawać, ale taki bez stosu będzie miał trudniej? Formalnie, jak to bywa, warto używać lematów.*

### 1.2 Obliczalność

**Zadanie 92.** *Jak coś ma nie być r.e. to na pewno chodzi o redukcję z  $\overline{\mathbb{K}}$ . Delikatna modyfikacja 89.*

**Zadanie 99.** *i) Jak coś ma nie być r.e. to na pewno chodzi o redukcję z  $\overline{\mathbb{K}}$ . Delikatna modyfikacja 89.*

*ii) Czy  $\Phi_m \neq \Phi_n \iff \exists k \Phi_m(k) \neq \Phi_n(k)$ ?*

**Zadanie 100.** To zadanie jest podobne do zadania 102. Zaś z zadania 82 wiemy, czym jest  $\Sigma_1$ .

**Zadanie 101.** Trzeba wprost zdefiniować redukcję z  $B$  do  $A$ .

**Zadanie 102.** a. założenie o nietrywialności zadania b. implikuje odpowiedź w zadaniu a.. Pewna użyteczna redukcja i w tym zadaniu okaże się pomocna.  
b. formalne sformułowanie tego, co to znaczy że  $n \in B$ , używająca 3 zmiennych, wprost prowadzi do co-r.e. zbioru  $A$ .

**Zadanie 103.** Należałoby się zastanowić, co wspólnego mają te warunki z monotonicznością  $f$ .

**Zadanie 110.** To jest wgl. ważne zadanie i warto mu poświęcić dłuższą chwilę.

a) Chcemy zakodować Maszynę Turinga (z ustalonym wejściem) jako automat deterministyczny z dwoma stosami.

a<sub>0</sub>) Zamienić MT z ustalonym wejściem na równoważną MT z pustym wejściem.

a<sub>1</sub>) Trzeba się przyjrzeć dokładniej temu, jak Maszyna wygląda, porysować coś. W szczególności spostrzec, że działa ona bardzo lokalnie. Trochę podobne do zadania 130, może być ono pewną inspiracją.

b) Zamienić stos na licznik.

b<sub>0</sub>) Jak zamienić potencjalnie szeroki wybór sweterków (jak u blondynki) na sweterki tylko dwóch kolorów (jak u blondyna)?

b<sub>1</sub>) Popatrzeć na taki zamieniony stos i zobaczyć, że w rzeczywistości wystarczy nam pamiętać jedną liczbę zamiast jednego stosu.

b<sub>2</sub>) Poświęcić chwilę (!) na względnie dopracowanie wymaganych od stosów operacji w licznikowej implementacji stosów. Przydatne mogą okazać się dwa dodatkowe liczniki.

c) Zachwycić się bogactwem liczb naturalnych - w szczególności można skorzystać ze znanego wszem i wobec kodowania skończonych ciągów liczbowych jako pojedyncze liczby. Np. ciągi 4-elementowe też się da. Zaimplementować wymagane operacje przy użyciu drugiego licznika.

**Zadanie 114.** Mocno podobne do zadania 127 (gramatyk ze znikaniem). Tym razem trzeba zrobić dwie gramatyki, osobną dla słów  $l_i$  i  $r_i$ .

**Zadanie 115.** Warto skorzystać z zadania 114. Udowodnić, że  $(L_G)^c$  dla CFG  $G$  z zad. 114 jest CFL. Porachować, przypomnieć sobie prawa de Morgana.

**Zadanie 116.** Warto skorzystać z zadania 115.

**Zadanie 118.** Czy da się, być może dodając specyficzne zamiany, zasymulować pełny semiproces Thuego, ograniczając się w **istotnych** zamianach tylko do zamiany początku pierwszego słowa na koniec drugiego?

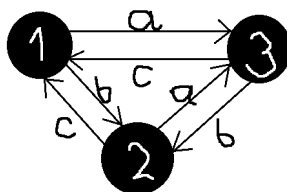
## 2 Szkice rozwiązań

### 2.1 Języki i automaty

#### 2.1.1 Synchronizacja automatów częściowych

**Zadanie 40.**

Odp: NIE.



Rysunek 1:  $csync(\{1, 2\}) \supset \{a\}$ ;  $csync(\{1, 3\}) \supset \{b\}$ ;  $csync(\{1, 2\}) \supset \{c\}$ .  
Natomiast  $csync(Q) = \emptyset$ .

**Zadanie 41.**

W obydwu podpunktach wystarczy zbadać funkcję

$$F : 2^Q \times \Sigma \longrightarrow 2^Q$$

$$F(S, a) = \{\delta(q, a) : q \in S\}.$$

Oczywiście  $s \in csync(S) \iff |\hat{F}(S, s)| = 1$ , gdy zdefiniujemy  $\hat{F}$  w naturalny sposób. Ponadto  $1 \leq |F(A, a)| \leq |A|$ , zatem  $1 \leq |\hat{F}(S, p)| \leq |S|$  dla dowolnego prefiksu  $p$  słowa  $s$ , czyli  $\hat{F}(S, p)$  może przyjmować co najwyżej  $\sum_{k=1}^{|S|} \binom{|Q|}{k}$  różnych wartości. Oznaczmy tę liczbę jako  $M$ .

Dla  $|s| > M$  istnieją prefiksy  $p_1$  i  $p_2$  słowa  $s = p_1 s_1 = p_2 s_2$ ,  $|p_1| < |p_2|$ , takie że  $\hat{F}(S, s_1) = \hat{F}(S, s_2)$  (Zasada Szufladkowa). Wtedy oczywiście  $\hat{F}(S, s) = \hat{F}(S, p_1 s_2)$ , przy czym  $|p_1 s_2| < |s|$ .

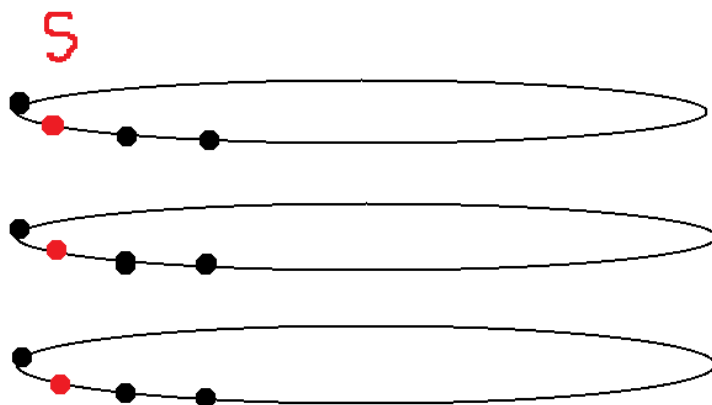
W związku z powyższym  $csync(Q) \neq \emptyset \iff \exists s \in S |s| \leq M$ .

Dokładne odpowiedzi wynikają z tego wprost, po podstawieniu za  $S$  a) dowolnego trzelementowego zbioru stanów b)  $Q$ .

### Zadanie 42.

Wystarczy rozwiązać M, L i XL wynikają w prosty sposób. Wskazówka jest myląca.

Zbudujmy automat (częściowy) z trzech cykli, ułożonych jeden nad drugim, każdy długości  $m$ . Trzy stany, ułożone jeden nad drugim, będą stanowiły nasz początkowy zbiór  $S$



Naszym celem jest, aby co jedną literę zmieniał się stan na górnym cyklu, co  $m$  liter stan na drugim, a co  $m^2$  na trzecim. Zapenimy też, że synchronizacja będzie mogła nastąpić dopiero po przejściu przez dolny stan całego cyklu (czyli  $m^3$  krokach).

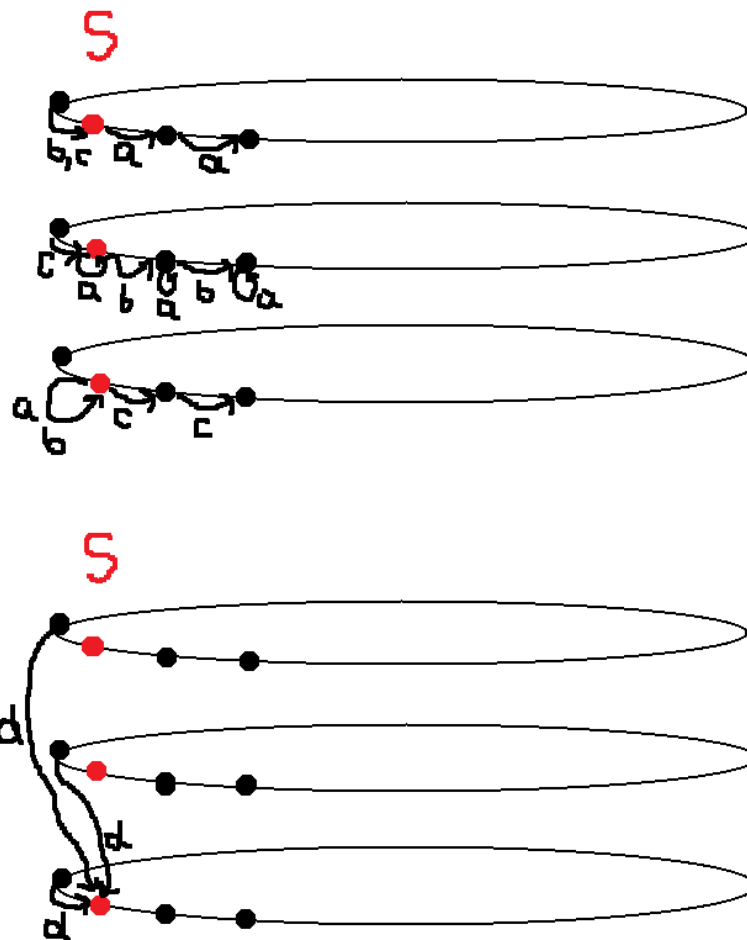
Możemy to wymusić w następujący sposób:

przy czym pętelki z literą  $a$  są przy każdym stanie na drugim dysku, a pętelki z literami  $a, b$  są przy każdym stanie na trzecim dysku.

Możliwość synchronizacji zapeniamy przez dodanie przejść z przedostatnich stanów na każdym dysku do pierwszego stanu dolnego dysku. Oznaczenie ich specjalną “literką synchronizacji”, jak  $d$ , zapewni nam, że skorzystać z niej będzie można dopiero gdy dolny stan dojdzie do przedostatniego miejsca na dolnym dysku (co następuje dopiero po  $m^3$  krokach:

Zauważmy teraz, że ten automat (zanim dojedzie do synchronizacji) jednoznacznie wyznacza słowo, dla którego funkcja przejścia jest określona dla wszystkich stanów z  $S$ :

$$s = ((a^{m-1}b)^{m-1}c)^{m-1}d$$



Takie  $s$  synchronizuje  $S$  i nie istnieje żadne krótsze od niego. Nietrudno wyliczyć, że jest ono odpowiednio długie.

### Zadanie 63.

Przypuśćmy nie wprost, że ten język jest bezkontekstowy. Niech  $N$  będzie stałą z lematu o pompowaniu dla tego języka. Wystarczy rozważyć słowo  $0^{2N}1^{2N}0^{2N}1^{2N}$  i pamiętać, że przy podziale z lematu (ozn.  $vxyz$ ) zachodzi  $|wxy| \leq N$ . Odpowiednio cierpliwe rozpatrywanie przypadków (gdzie w oryginalnym słowie łąduje podśłowo  $wxy$ ) prowadzi nas do wniosku, że zawsze można odpowiedni fragment podpompować (lub spompować) i uzyskać słowo spoza języka.

### 2.1.2 Języki rodzyńkowe

#### Zadanie 71.

Rozważmy  $L = \{a^n b a^n : n \in \mathbb{N}\}$ . Łatwo sprawdzić, że nawet deterministyczny automat ze stosem daje sobie radę z językiem  $L^*$ , bo rozpoznawanie liter tego języka (czyli słów języka  $L$ ) jest bardzo łatwe.

Z lematu o pompowaniu bardzo łatwo pokazać, że  $L^*$  nie jest regularny ( $L$  też nie jest).

### 2.1.3 Transducery

#### Zadanie 77.

Podpunkt 1: definiujemy  $\sigma_{Mealy} = \sigma_{Moore} \circ \delta$ . Reszta zostaje.

Podpunkt 2: definiujemy (dla transducera Mealy'ego  $\langle \Sigma, \Sigma_1, Q, q_0, \delta, \sigma_{Mealy} \rangle$ )

1.  $Q' = Q \times \Sigma \cup q'_0$ . Stan  $(q, a)$  = stan do którego doszlibyśmy w starym automacie ze stanu  $q$  wczytując literę  $a$ . Stan  $q'_0$  - dodatkowy stan początkowy.
2.  $\delta'((q, a), b) = (\delta(q, a), b)$  dla pary  $(q, a) \in Q \times \Sigma$   
 $\delta'(q'_0, a) = (q_0, a)$
3.  $\sigma_{Moore}((q, a)) = \sigma_{Mealy}(q, a)$   
 $\sigma_{Moore}(q'_0) = \varepsilon$

Otrzymujemy transducer Moore'a  $\langle \Sigma, \Sigma_1, Q', q'_0, \delta', \sigma_{Moore} \rangle$  równoważny z pierwotnym t. Mealy'ego.

Dowód w obu przypadkach zapewne angażuje Zasadę Indukcji Matematycznej względem długości słowa.

#### Zadanie 78.

bso. (77) zajmijmy się transducerem Mealy'ego  $\langle \Sigma, \Sigma_1, Q, q_0, \delta, \sigma_{Mealy} \rangle$ , który świadczy że  $A \leq_{reg} B$ . Niech przy okazji  $A_B = \langle \Sigma_1, Q^B, q_0^B, F^B, \delta^B \rangle$  będzie DFA rozpoznającym  $B$ . Definiujemy  $\delta'(q, a) = \widehat{\delta^B}(q, \sigma_{Mealy}(q, a))$ . Udajemy, że jesteśmy słowem z języka  $B$  i chodzimy po automacie  $A_B$ . Wtedy  $\langle \Sigma, Q^B, q_0^B, F^B, \delta' \rangle$  jest DFA rozpoznającym  $A$  (d-d. indukcyjny względem długości słowa).

#### Zadanie 79.

Definiujemy transducer Mealy'ego  $T_{Mealy}$ :

1.  $\Sigma = \{1, 2, 3, \dots, n\}$

2.  $Q = \Sigma$

3.  $q_0 = 1$

4.  $\delta(q, a) = a$

5.  $\Sigma_1 = Q \times \Sigma$

6.  $\sigma_{Mealy} = Id$

Niech  $T_{Moore} = \langle \Sigma, \Sigma_1, Q', q'_0, \delta', \sigma_{Moore} \rangle$  będzie transducerem Moore'a równoważnym z  $T_{Mealy}$ .

**Obserwacja 1.** Możemy założyć, że każdy stan z  $Q'$  jest osiągany przez DFA stowarzyszony z  $T_{Moore}$ . W przeciwnym razie możemy usunąć te stany, a powstały  $T'_{Moore}$  wciąż będzie równoważny z  $T_{Mealy}$ .

**Obserwacja 2.**  $s \in Im(\sigma_{Mealy}) \Rightarrow |s| = 1$ . Zatem  $s \in Im(\sigma_{Moore}) \Rightarrow |s| = 1$ .

Gdyby było  $|Q'| < n^2$ , to  $Im(\sigma_{Mealy}) \not\subseteq Im(\sigma_{Moore})$ . Niech  $s \in Im(\sigma_{Mealy}) \setminus Im(\sigma_{Moore})$ .  $s = (k, l)$  dla pewnych  $k, l \in \Sigma$ . Rozważmy słowo  $t = kl$ . Wtedy  $f_{T_{Mealy}}(t) = \sigma_{Mealy}(1, k)\sigma_{Mealy}(k, l) = (1, k)(k, l)$ . Załóżmy nie wprost, że  $f_{T_{Moore}}(t) = f_{T_{Mealy}}(t)$ . Jest to równoważne (obs. 2) z tym, że  $\sigma_{Moore}(\delta'(q'_0, k)) = \sigma_{Mealy}(1, k)$  oraz  $\sigma_{Moore}(\widehat{\delta'}(q'_0, kl)) = \sigma_{Mealy}(k, l) = (k, l)$ . Druga równość stoi w jawnej sprzeczności z naszym założeniem, że  $(k, l) \notin Im(\sigma_{Moore})$ .

### Zadanie 80.

Zdaje się, że świadczy o tym następujący transducer Mealy'ego:

1.  $\Sigma = \{(\cdot), [\cdot], \langle \cdot \rangle\}$

2.  $Q = \{q_0\}$

3.  $q_0 = q_0$

4.  $\delta \equiv q_0$

5.  $\Sigma_1 = \{(\cdot), [\cdot], \langle \cdot \rangle\}$

6.  $\sigma_{Mealy}(q_0, (\cdot)) = ((\cdot))$   
 $\sigma_{Mealy}(q_0, [\cdot]) = [[\cdot]]$   
 $\sigma_{Mealy}(q_0, \langle \cdot \rangle) = [\langle \cdot \rangle]$

Dowód pozostawiamy Czytelnikowi jako ćwiczenie.

## 2.2 Obliczalność

### Zadanie 89.

Niech  $B = \{n : \text{Dom}(\Phi_n) = \mathbb{N}\}$ . Robimy redukcję  $f_{89}$  z  $\overline{\mathbb{K}}$ .  
Definiujemy  $f_{89}(n)$  jako numer następującego programu:

```
wczytaj k
odpal  $\Phi_n(n)$  na  $k$  kroków
jeżeli się skończył : zapętl się
w p.p. : zwróć 1
```

Sprawdzenie, że zachodzi  $n \in \overline{\mathbb{K}} \iff f_{89}(n) \in B$  pozostawiamy jako proste ćwiczenie.

### Zadanie 92.

Robimy redukcję  $f$  z  $\overline{\mathbb{K}}$ .  
Definiujemy  $f(n)$  jako numer następującego programu:

```
wczytaj k
jeżeli  $k$  jest parzyste : zapętl się
w p.p. :
odpal  $\Phi_n(n)$  na  $k$  kroków
jeżeli się skończył : zapętl się
w p.p. : zwróć 1
```

Oczywiście  $\text{Dom}(\Phi_{f(n)}) = (2\mathbb{N} + \{1\}) \cap \{t \in \mathbb{N} : t < \text{czas wykonania się } \Phi_n(n)\}$ .  
Sprawdzenie, że zachodzi  $n \in \overline{\mathbb{K}} \iff f(n) \in B$  pozostawiamy jako proste ćwiczenie.

### Zadanie 99.

i) Robimy redukcję  $f$  z  $\overline{\mathbb{K}}$ , pamiętając o redukcji  $f_{89}$ .  
Niech  $c$  będzie numerem następującego programu:

```
wczytaj k
zwróć 1
```

Definiujemy  $f(n) = \langle c, f_{89}(n) \rangle$ . Jest oczywiste (po zrobieniu zadania 89), że zachodzi

$$n \in \overline{\mathbb{K}} \iff f(n) \in T.$$

ii) Nie. Robimy redukcję  $f$  z  $\overline{\mathbb{K}}$ .

Niech  $f_a(n)$  będzie numerem następującego programu:

```
wczytaj k
jeżeli  $k > 1$  : zapętl się
w p.p. :
odpal  $\Phi_n(n)$ 
zwróć 1
```

Niech  $f_b(n)$  będzie numerem następującego programu:

```
wczytaj k
```



jeżeli  $k > 1$  : zapętl się

w p.p. :

zwróć 1

Definiujemy  $f(n) = \langle f_a(n), f_b(n) \rangle$ . Oczywiście zachodzi

$$n \in \overline{\mathbb{K}} \iff f(n) \in \overline{T}.$$

### Zadanie 101.

Definiujemy  $f^{-1}$  w następujący sposób:

wczytaj  $n$

Dla  $k = 1, 2, 3, \dots$  :

jeżeli  $f(k) = n$  : zwróć  $k$

Oczywiście  $f^{-1}$  jest całkowita, bo  $f$  jest "na". Łatwo sprawdzić, że zachodzi również  $f \circ f^{-1} = Id$  (a czy  $f^{-1} \circ f = Id$ ?).

Pokażemy, że  $f^{-1}$  jest redukcją z  $B$  do  $A$  :

$$n \in B \iff Id(n) \in B \iff f(f^{-1}(n)) \in B \iff f^{-1}(n) \in A,$$

przy czym ostatnia równoważność wynika z tego, że  $f$  jest redukcją z  $A$  do  $B$ . Oczywiście powyższe oznacza dokładnie to, że  $f^{-1}$  jest redukcją z  $B$  do  $A$ .

### Zadanie 102.

a) Nie. Wystarczy pokazać redukcję z  $\mathbb{K}$  (dlaczego?). O dziwo, jest to dokładnie  $f_{89}$ .

b) Definiujemy  $A = \{(n, k, t) : \Phi_n(i) \text{ kończy się po co najwyżej } t \text{ krokach dla } i = 1, 2, \dots, k \text{ oraz nie kończy się dla } i > k\}$ . Ten zbiór jest dobry, a sprawdzenie pozostawiamy jako ćwiczenie.

### Zadanie 103.

a) Tak. Bo istnieją takie duże  $N$ , że na zbiorze  $\{N, N+1, \dots\}$   $f$  jest nie-malejąca. Z zadania 86 wynika więc, że  $f(\{N, N+1, \dots\})$  jest rekurencyjny. Zatem  $f(\mathbb{N}) = f(\{1, 2, \dots, N-1\}) \cup f(\{N, N+1, \dots\})$  jest rekurencyjny, jako suma dwóch zbiorów rekurencyjnych.

b) Nie. Niech  $A = \{1\}$ . Niech  $g$  będzie całkowitą funkcją rekurencyjną, taką że  $g(\mathbb{N}) = \mathbb{K}$  (skądinąd wiemy, że taka istnieje - np. 87). Rozważmy następującą funkcję  $f$ :

wczytaj  $n$

jeżeli  $n$  jest parzyste : zwróć  $g(\frac{n}{2})$

w p.p. : zwróć 1

$f$  spełnia warunek z zadania, ale  $f(\mathbb{N}) = \mathbb{K} \cup \{1\}$ , a to nie jest zbiór rekurencyjny.

### Zadanie 105.

Uwaga:  $N_{emp}$  jest r.e. Semi-rozstrzyga go następujący program  $\Phi_{N_{emp}}$ :

```
wczytaj n
dla  $k = 1, 2, 3, \dots$  :
  odpal  $\Phi_n(i)$  na  $k$  kroków dla  $i = 1, 2, \dots, k$ 
  jeżeli na którymś z argumentów  $\Phi_n$  się zatrzymał : zwróć 1
```

a) Tak. Świadczy o tym następująca redukcja  $f$ :

Jako  $f(n)$  przyjmijmy numer następującego programu:

```
wczytaj k
odpal  $\Phi_{N_{emp}}(n)$ 
zwróć 1
```

Dowód poprawności pozostawiamy jako ćwiczenie.

b) Nie, bo  $Tot$  nie jest r.e. (zadanie 89), a  $N_{emp}$  jest.