# Spis treści

**1 Egzamin 2005** **2**
1.1 Część 2 . . . . . 2
1.1.1 Zadanie 5 . . . . . 2

**2 Egzamin 2007 poprawka** **4**
2.1 Część 2 . . . . . 4
2.1.1 Zadanie 4 . . . . . 4

**3 Egzamin 2008** **5**
3.1 Część 2 . . . . . 5
3.1.1 Zadanie 4 . . . . . 5
3.1.2 zadanie 5 . . . . . 5
3.1.3 zadanie 6 . . . . . 6

**4 Egzamin 2009 Poprawka** **8**
4.1 Część 3 . . . . . 8
4.1.1 zadanie 7 . . . . . 8
4.1.2 zadanie 8 . . . . . 8
4.1.3 zadanie 9 . . . . . 8

**5 Egzamin 2011** **10**
5.1 Część 2 . . . . . 10
5.1.1 Zadanie 5 . . . . . 10
5.1.2 Zadanie 6 . . . . . 10

# Rozdział 1

# Egzamin 2005

## 1.1. Część 2

### 1.1.1. Zadanie 5

Udowodnij, że zbiór takich par $< G_1, G_2 >$ gramatyk bezkontekstowych, dla których $L(G_1) \subseteq L(G_2)$ nie jest rekurencyjny. Wskazówka: skorzystaj z nierozstrzygalności PCP, i z tego że język słów które nie sa palidromami jest CFL.

$$A = \{< G_1, G_2 >: L(G_1) \subseteq L(G_2)\}$$

Korzystając ze wskazówki pokażę, że:

*Dowód.* NIEWPROST Załóżmy że $A - -rek \quad \Leftrightarrow \quad \exists_{\phi_A} \forall_x [(\phi_A(x) = 1 \Leftrightarrow x \in A) \wedge (\phi_A(x) = 0 \Leftrightarrow x \notin A)]$. Korzystając z powyższego założenia skonstruujemy program $phi_{PCP}$ rozpoznający przynależność do PCP. Program będzie działać następująco:

1. Wczytaj zbiór par: $P = \{(l_1, r_1), \cdots, (l_k, r_k)\}$. Niech $\Sigma$ będzie alfabetem symboli, z których zbudowane są słowa $l_i$ oraz $r_i$ z jednym dodatkowym symbolem '#' nie występującym w żadnym $l_i$ oraz $r_i$.

2. Korzystając ze wskazówki niech $G_2$ będzie gramatyka generująca język słów nad alfabetem $\Sigma$, które nie są palidromami.

3. Niech:
$$G_1 =< \Sigma, \{S\}, \{S \to \#\} \cup \left\{S \to l_i^R S r_i : 1 \geqslant i \leqslant k\right\}, S >$$
Zaóważmy, że:
$$L(G_1) = \left\{t^R \# v : \exists_{w_1, \cdot w_s \in \{1, \cdots, k\}} t = l_{w_1} \cdots l_{w_s} \wedge v = r_{w_1} \cdot r_{w_s}\right\}$$

4. zwróć 1 jeżeli $\phi_A(< G_1, G_2 >) = 0$, 0 wpp.

Zaóważmy że powyższy program rzeczywiście rozstrzyga przynależność do PCP:

$P \in PCP$ Wtedy istnieje taki ciąg $w_1, \cdots, w_s \in \{1, \cdots, k\}$, że $t = l_{w_1} \cdots l_{w_s}$, $v = r_{w_1} \cdots r_{w_s}$ i $t = v$. Słowo $t^R \# v$ jest palidromem i $t^R \# v \notin A$, wiec $\phi_{PCP}(P) = \phi_A(< G_1, G_2 >) = 0$

$P \notin PCP$ Wtedy nie istnieje taki ciąg by $t = v$. Słowo $t^R \# v$ nie jest palidrome, gdyż:

**a)** jeżeli t i v są rożnej długości to symbol # nie leży po środku,

**b)** jeżeli t i v są równej długości to słowo nie jest palidromem bo $t \neq v$.

Zaóważmy, że wtedy $\forall_{t,v} t^R \# v \in A$ czyli żadne słowo generowane przez $G_1$ nie jest palidromem. Z powyższych rozważań wynika że: $\phi_{PCP}(P) = \phi_A(< G_1, G_2 >) = 1$

Z powyższych rozwań wynika sprzeczność gdyż pokazaliśmy że PCP jest rek, a wiemy że tak nie jest.

$\square$

# Rozdział 2

# Egzamin 2007 poprawka

## 2.1. Część 2

### 2.1.1. Zadanie 4

Niech A będzie zbiorem takich par numerów programów $\langle n, m \rangle$, że dla każdej naturalnej k albo program o numerze n uruchominy dla danej k się zatrzymuje, a program o numerze m uruchominy dla danej k się nie zatrzymuje, albo program o numerze m uruchomiony dla danej k się zatrzymuje, a program o numerze n uruchominy dla danej k się nie zatrzymuje.

Czy A jest rekurencyjnie przeliczalny?

$$A = \{< n, m >: \forall_k [(\phi_n(k) = \perp \wedge \phi_m(k) \neq \perp) \vee (\phi_n(k) \neq \perp \wedge \phi_m(k) = \perp)]\} \tag{2.1}$$

Pokażemy, że A nie jest rekurencyjnie przeliczalny.
Załóżmy niewprost że:

$$A - r.e. \Leftrightarrow \exists_{\phi_A} \forall_{n,m} (\phi_A(< n, m >) = 1 \Leftrightarrow < n, m > \in A) \wedge (\phi_A(< n, m >) = \perp \Leftrightarrow < n, m > \notin A) \tag{2.2}$$

Skonstruujemy następujący program, rozpoznający przynależność do $\overline{K}$:

```
wczytaj n
t <- numer nastepujacego programu:
        wczytaj m
        return 1
t' <- numer nastepujacego programu:
        wczutaj n
        return phi_n(n)
return phi_A(t, t')
```

Program rzeczywiście rozpoznaje przynależność do $\overline{K}$:

$n \in \overline{K}$

$$n \in \overline{K} \;\Rightarrow\; \forall_x \phi_{t'}(x) = \perp \;\Rightarrow\; \phi_A(t, t') = 1 \tag{2.3}$$

$n \notin \overline{K}$

$$n \notin \overline{K} \;\Rightarrow\; \forall_x \phi_{t'}(x) \neq \perp \;\Rightarrow\; \phi_A(t, t') = \perp \tag{2.4}$$

Tutaj mamy sprzeczność gdyż wiemy że $\overline{K}$ nie jest r.e.

# Rozdział 3

# Egzamin 2008

## 3.1. Część 2

### 3.1.1. Zadanie 4

Czy istnieje algorytm rostrzygający dla danych dwóch deterministycznych automatów ze stosem, czy istnieje niepuste słowo, które zostanie zaakceptowane przez oba te automaty?

Przez deterministyczny automat ze stosem rozumiemy tu automat, którego każdy krok polega na wczytaniu jedneo symbulu ze słowa wejściowego, jednego symbolu ze stosu i w zależności od wczytanych danych oraz od aktualnego stanu, na zmianie stanu i odłożeni na stos ciągu ( być może pustego) symboli (w szczególności taki automat nie wykonue $\varepsilon$-przejść). Automat akceptuje słowo, jeśli po wczytaniu tego słowa stos jest pusty (tzn. zaiwera jedynie symbol dna stosu).

Taki algorytm nieistnieje. W celu pokazania tego, stworzymy redukcję, przedstawioną poniżej, problemu PCP (nierostrzygalnego) do naszego problemu z automatami.

$$f(\langle(l_1, r_1), \cdots, (l_k, r_k)\rangle) = (A_1, A_2) \text{ gdzie } A_1 \text{ i } A_2 \text{ to automat opisane w zadaniu} \tag{3.1}$$

Automaty $A_1$ i $A_2$ zbudujemy tak by akceptoway języki opisane poniżej:

$$L(A_1) = L(\{i_{j_1} \cdots i_{j_s} l_{j_s}^R \cdots l_{j_1}^R\}) \tag{3.2}$$

$$L(A_2) = L(\{i_{j_1} \cdots i_{j_s} r_{j_s}^R \cdots r_{j_1}^R\}) \tag{3.3}$$

Idea automaty $A_1$: Automat wczytuje tylke słowa postaci ciąg indeksów, po którym występuje ciąg literek, w przeciwnym przypadku zatrzymuje sie z niepustym stosem.

1. automat wczytuje kolejne indeksy $i_j$ i wrzuca na stos słowa $l_{i_j}^R$
2. gdy zobaczy literke to ściąga ją ze stosu jeżeli taka sama leży na wierzchu lub się zacina

### 3.1.2. zadanie 5

Niech $A \subset N$ (nie jest r.e.) będzie zbiorem tych liczb naturalnych, ktore są numerami programów zatrzymujących się dla wszystkich danych. Niech $E \subset N$ (nie jest r.e.) bedzie zbiorem tych liczb naturalnych, które sa numerami programów nie zatrzymujujących się dla żadnych danych. Udowodnij, że $E \leqslant_{rek} A$.

Wskazówka: To jest łatwe zadanie. Przypomnij sobie co to jest redukjca. Pomyśl jakiego typu muszą byc argumenty redukcji, która masz zbudować i jakiego typu maja być wartości. Nie gap sie w pustą kartkę, napisz sobie równoważność, ktora ma zachodzić.

$$A = \{i : \forall_x \varphi_i(x) \neq \perp\} \tag{3.4}$$

$$E = \{j : \forall_x \varphi_j(x) = \perp\} \tag{3.5}$$

Mamy pokazać że:

$$E \leqslant_{rek} A \ \Leftrightarrow (\exists_f \forall_{x \in N} x \in E \Leftrightarrow f(x) \in A) \tag{3.6}$$

Pokażemy że istnieje takie $f$. W tym celu stworzymy odpowiednią redukcję, której kod przedstiony jest poniżej:

Niech $b : N \rightarrow N \times N$ będzie bijekcją (np taką przekątniową $b(1) = (1,1)$, $b(2) = (1,2)$, $b(3) = (2,1)$, ...).

```
wczytaj n
t = numer takiego programu:
        wczytaj m
        x, k = b(m)
        uruchom fi_n(x) na k krokow
                jesli fi_n(x) zatrzyma sie i zwroci wynik to zapetl sie
                w.p.p. zwroc 1
zwroc t
```

$n \in E$

$$n \in E \ \rightarrow \ \forall_x \varphi_n(x) = \perp \ \rightarrow \ \forall_x \varphi_t(x) = 1 \neq \perp \ \rightarrow t = f(n) \in A$$

$n \notin E$

$$n \notin E \ \rightarrow \ \exists_x \varphi_n(x) \neq \perp \rightarrow \ \exists_m b(m) = (x,k) \wedge \varphi_n(x) \text{ uruchomione na k kroków} \neq \perp \rightarrow \exists_m \varphi_t(m) = \perp \rightarrow n \notin A$$

### 3.1.3. zadanie 6

Niech A i E będą takie, jak w poprzednim zadaniu. Czy zachodzi nierówność $A \leqslant_{rek} E$?
Wskazówka: Co pamiętasz z ćwiczeń na temat nierówności $K \leqslant_{rek} \overline{K}$?

Korzystając ze wskazówki wiemy że $K \nleqslant_{rek} \overline{K}$. Pokażemy, że:

$$K \leqslant_{rek} A \ \wedge \ E \leqslant_{rek} \overline{K} \ \Rightarrow \ A \nleqslant_{rek} E \tag{3.7}$$

Najpierw pokażemy, że:

$$K \leqslant_{rek} A \Leftrightarrow \exists_f \forall_x x \in K \Leftrightarrow f(x) \in A \tag{3.8}$$

W tym celu skonstrujemy następująca redukcje f:

```
wczytaj n
t <- znajdz numer nastepujacego programu:
        wczytaj m
        zwroc fi_n(n)
zwroc t
```

$n \in K$

$$n \in K \Rightarrow \forall_x \varphi_t(x) = \varphi_n(n) \neq \perp \Rightarrow n \in A$$

$n \notin K$

$$n \notin K \Rightarrow \forall_x \varphi_t(x) = \varphi_n(n) = \perp \Rightarrow n \notin A$$

Teraz pokażemy, że:

$$E \leqslant_{rek} \overline{K} \Leftrightarrow \exists_f \forall_x x \in K \Leftrightarrow f(x) \in \overline{K} \tag{3.9}$$

W tym celu skonstrujuemy następującą redukcje f:

```
wczytaj n
t <- znajdz numer nastepujacego programu:
        wczytaj m
        dla kazdego i od 1 do ....
                dla kazdego j od 1 do ...
                        jezeli fi_n(i) zwroci wyniki po j krokach zwroc 1
        zwroc 1
zwroc t
```

$n \in E$

$$n \in E \Rightarrow \forall_x \varphi_n(x) = \perp \Rightarrow \neq \exists_{i,j} \varphi_n(i) \text{ zatrzyma sie po j krokach } \Rightarrow \forall \varphi_t(x) = \perp \Rightarrow \varphi_t(t) = \perp \Rightarrow t = f(n) \in \overline{K}$$

$n \notin E$

$$n \notin E \Rightarrow \exists_{i,j} \varphi_n(i) \text{ zatrzyma sie po j krokach } \Rightarrow \forall_x \varphi_t(x) = 1 \Rightarrow \varphi_t(t) = 1 \Rightarrow t = f(n) \notin \overline{K}$$

# Rozdział 4

# Egzamin 2009 Poprawka

## 4.1. Część 3

### 4.1.1. zadanie 7

> Jaka jest zlożoność problemu 3-kolorowania grafów, jeśli ograniczymy się do grafów o stopniu wierzchołków równym co najwyżej 4?
>
> Wskazówka: Zadania 8 i 9 są być może łatwiejsze.

### 4.1.2. zadanie 8

> W kolejnych zadanich rozważamy akademik, w którym studenci budzą się o godzinie 9 rano, a k kwadransów później wychodza na egzamin. Przed wyjściem każdy z nich musi (w dowolnej kolejności) wykonać k czynności (być może innych dla każdego studenta), z których każda trwa kwadrans i musi byc wykonywana nieprzerwanie. Problem polega na tym, że niektóre czynności sie wykluczają: na przykąłd prasowanie białej bluzeczki wymaga zajęcia całego stołu, więc nie można prasowac jeśli ktos akurat je śniadnie, albo pisze ściągę. Relacja wykluczania nie musi jednak byc przechodnia: można przeciez pisac ściągę, kiedy ktoś je śnaidanie.
>
> Instacją Problemu Zdążenia w k Kwadransów ($PZK_k$) będzie lista studnetów, wraz z czynnościami które moja wykonać, oraz lista par czynności które sie wykluczają. $PZK_k$ będzie zbiorem tych instacji, dla których istnieje harmonogram wykonywania czynności przy których studenci zdążą na egzamin i który nie zakąłda wykonywania jednocześnie wykluczających się czynności.
>
> Jaka jest złożoność problemu $PZK_2$?

Problem ten jest w P. Pokażemy to przez redukcję problemu $PZK_2$ do $2SAT$, który jest w P:

$$PZK_2 \leqslant 2SAT$$

Redukcja $f : PZK_2 \to 2SAT$:

1. Dla każdej czynności a tworzymy nową zmienną $x_a$.
2. Jeżeli czynności a i b nie mogą być wykonywane jednocześnie to tworzymy nowa klauzule $(x_a \vee \not{x}_b) \wedge (\not{x}_a \vee x_b)$.

### 4.1.3. zadanie 9

> Jaka jest złożoność problu $PZK_3$?

Problem ten jest NP-zupełny. Pokażemy to konstruując redukcję z NP-zupełnego proglemu 3COL do $PZK_3$:

$$3COL \leqslant_P PZK_3 \Leftrightarrow \left(\exists_{\text{wielomianowa redukcja f}} \forall_x x \in 3COL \Leftrightarrow f(x) \in PZK_3\right)$$

Redukcja $f : 3COL \to PZK_3$:

1. Redukcja jako argument przyjmuje graf $G =< V, E >$, a zwraca trójkę $< A, \{S_i\}, E' >$ gdzie:

   **A** – zbiór czynności,

   $S_i$ – to czynności do wykonania przez i'tego studenta,

   **E'** – zbiór par czynności, ktore nie moga być wykonywane jednocześnie.

2. Za zbiór czynności A, redukcja przyjmuje wierzchołki grafu oraz dodatkowy element 'x' nie występujący w V.
$$A = V \cup \{x\}$$

3. Tworzymy $n = |V|$ studentów. Niech $g : V \to \{1, \cdots, n\}$ będzie bijekcją odwzorowującą zbiór V w liczby $1, \cdots, n$. Każdemu wierzchołkowi $v \in V$, przyporządkowujemy studenta $S_{g(v)}$ ze zbiorem czynności $\{v, x, x\}$.
$$\forall_v S_{g(v)} = \{v, x, x\}$$

4. Za zbiór par czynności, które nie moga być wykonywane jednocześnie E' przyjmujemy zbiór krawędźi E.

5. Podsumowując:
$$f(G) = f(< V, E >) =< V \cup \{x\}, < S_1, \cdots S_n >, E > \quad \wedge x \notin V \wedge S_i = \left\{g^{-1}(i), x, x\right\}$$

*Dowód.* Redukcja jest wielomianowa gdyż stworzenie zbioru A, $S_i$ oraz E' jest O(n). Kolory używane w kolorowaniu grafu oznaczmy jako: $\{1_c, 2_c, 3_c\}$. Poniżej pokażemy że jeżeli graf G należy do 3COL to f(G) należy do $PZK_3$ i wdrugą stornę.

$\Rightarrow$
1. $f(G) = f(< V, E >) =< A, \{S_i\}, E' >$
2. Z założenia wiemy, że istniеje trój pokolorowanie grafu G.
3. Za kolor studenta i, będziemy rozumieli kolor wierzchołka znajdującego się w zbiorze $S_i$ (z konstruckji f wiemy, że w zbiorze $S_i$ może znajdować się tylko jeden wierzchołek).
4. Podzielmy studentów na trzy grupy X, Y i Z, w taki sposób aby w każdej z grup znajdowali się studenci tego samego koloru i każdy student należał do jednej z grup.
5. Niech studenci z grupy X wykonują czynności inną niż 'x' jako pierwszą z koleji, studenci z Y jako drugą, a studneci z Z jako trzecią.
6. Zaóważmy że w każdym z kwadransów wykonywane są tylko czynności tego samego koloru i czynność 'x', która może być jednocześnie wykonywana z każda inną czynnością.
7. Ponieważ $E = E'$ więc czynności tego samego koloru mogą być wykonywane jednocześnie.
8. Jak pokazaliśmy powyżej: $< A \{S_i\}, E' >\in PZK_3$.

$Leftarrow$

1. Z założenia wiemy, że $f(G) \in PZK_3$.
2. Analogicznie jak powyżej niech X, Y, Z, bedą zbiorami studentów takimi, że:

   **X** – studenci, którzy wykonują czynność inną niż 'x' jako pierwsza,

   **Y** – studenci, którzy wykonują czynność inną niż 'x' jako drugą,

   **Z** – studenci, którzy wykonują czynność inną niż 'x' jako trzecią.
3. Wierzchołki wykonywane przez studentów z X jako czynność pokolorujmy na kolor $1_c$, z Y na kolor $2_c$, a z Z na kolor $3_c$.
4. Zaóważmy że wtedy każdy wierzchołek ma przypisany jeden z trzech kolorów i żaden jego sąsiad w grafie nie ma takiego samego koloru.

□

# Rozdział 5

# Egzamin 2011

## 5.1. Część 2

### 5.1.1. Zadanie 5

$$A_0 = \{\phi_n : Dom(\phi_n) = \emptyset\}$$
$$A_1 = \{\phi_n : |Dom(\phi_n)| = 1\}$$

Czy $A_0 \leqslant_{REK} A_1$ ?

$$A_0 \leqslant_{REK} A_1 \;\Leftrightarrow\; \exists_f \forall_n n \in A_0 \Leftrightarrow f(n) \in A_1$$

REDUKCJA f:

```
wczytaj n
t <- znajdz numer programu:
        wczytaj m
        if m == 1 then zwroc 1
        else zwroc phi_n(m-1)
return t
```

### 5.1.2. Zadanie 6

Czy $A_1 \leqslant A_0$?

Pokażemy, że:

$$K \leqslant A_1 \;\wedge\; A_0 \overline{K} \;\Rightarrow\; A_1 \nleqslant A_0 \tag{5.1}$$

1) Pokażemy, że:

$$K \leqslant A_1 \;\Leftrightarrow\; (\exists_{f_1} \forall_n n \in K \Leftrightarrow f(n) \in A_1)$$

REDUKCA $f_1$:

```
wczytaj n
t <- znajdz numer programu:
        wczytaj m
        if m == 1 then
                uruchom $phi_n(n)$
        else zapetl sie
return t
```

2) Pokażemy, że:

$$A_0 \leqslant \overline{K} \ \Leftrightarrow \ (\exists_{f_2} \forall_n n \in A_0 \Leftrightarrow f(n) \in \overline{K})$$

REDUKCJA $f_2$:

```
wczytaj n
t <- znajdz numer programu:
        wczytaj m
        for i=1 to +inf
                for j=1 to i
                        uruchom phi_n(m) na i krokow
                        jezeli phi_n(m) zatrzyma sie i zwroci wynik to zwroc 1
return t
```

# Bibliografia

[1] test reference