

Rozwiązania zadań ze zbioru zadań edycji 2020

z Języków Formalnych i Złożoności Obliczeniowej

Bartłomiej Grochowski

Część druga:

Problemy nierostrzygalne, \mathcal{K} , redukcje
zadania 82-137

Zamieszczone tu rozwiązania mają służyć
złapaniu pewnych idei i mogą być pomocą przy
rozpoczęciu przygody z JFiZO. Spisywanie rozwiązań,
niestety, nie ułatwi zdania egzaminu.

Uniwersytet Wrocławski
18 czerwca 2023

JF120 zad 82

$A \subseteq \mathbb{N} \times \mathbb{N}$ rekurencyjny $\Rightarrow \exists \varphi \text{ t.z. } \forall n, m \in \mathbb{N} \left\{ \begin{array}{l} \langle n, m \rangle \in A \Leftrightarrow \varphi(n, m) = 1 \\ \langle n, m \rangle \notin A \Leftrightarrow \varphi(n, m) = 0 \end{array} \right.$

A rek $\Rightarrow B = \{n \mid \exists m \langle n, m \rangle \in A\}$ r.e.

$\varphi_B(n)$ zwraca 1, jeśli da się dobrąć m , iż $\varphi_A(n, m) = 1$

$\varphi_B(n) = \text{for } m=0 \text{ to } \infty$
 $\quad \text{if } (\varphi_A(n, m) == 1) \text{ return } 1$ } ta pętla może
trwać w nieskończoność
 $\quad \text{if nie skończy, to}$
 $\quad \quad A \text{ jest rek}$

Czyli B jest r.e.

JF120 zad 83

$B = \{n \mid \exists m \langle n, m \rangle \in A\}$ r.e. $\Rightarrow \exists A \subseteq \mathbb{N}^2$ rek

B jest r.e., czyli $\exists \varphi_B \text{ s.t. } n \in B \Leftrightarrow \varphi_B(n) = 1$

Pokażemy A , który jest rekurencyjny (czyli program dla niego)

$A = \{\langle n, m \rangle \mid \underbrace{\varphi_B(n) = 1} \wedge \varphi_B(n) \text{ działa w jednej sekundzie, } m \text{ sekund}\}$
konceny się

$\varphi_A(n, m) = \text{WAIT } (\overset{\text{time}}{\downarrow} m, \overset{\text{wait for operation}}{\downarrow} \varphi_B(n))$
 return 1 // udało się w mniejszej niż m sekundach
 return 0 // nie udało - działałem oganiczony rekurencyjnie

ZF120 zad 84 : Nierozstrzygalność problemu stoper

Numerujemy wszystkie funkcje

	1	2	3	4	...
φ_1	7				
φ_2		5			
φ_3			1		
φ_4				3	
:					
K					

$$K = \{n \mid \varphi_n(n) \in \mathbb{N}\}$$

→ czyli nie ma pionów na przekątnej.

\bar{K} - cała reszta \mathbb{N}

TABELKA: Wszystkie programy o wszystkich możliwych argumentach.

Zat. iż K jest REk. Wtedy istnieje program ψ rozstrzygający K .

(zwraże 1, jeśli $w \in K$ else 0)

$\psi^1(m) = \begin{cases} \text{if } \underline{\psi(m)} = 1 \text{ then loop forever} \\ \text{else return 1} \end{cases}$

↑
do swego obliczenia, bo zat., iż K jest REk.

ψ^1 jest programem i ma swój numer. ($\psi^1 = \varphi_k$)

$$\begin{array}{ccc} k \in K & \Leftrightarrow & \varphi_k(k) \text{ się zatrzymuje} \Leftrightarrow \psi^1(k) \text{ się zatrzymuje} \\ \uparrow & & \uparrow \\ \text{z def. zbioru } K & \text{bo } \varphi_k = \psi^1 & \text{z dziedziną,} \\ & & \text{funkcji } \psi \\ & & \Downarrow \\ & & \psi(k) = 0 \\ & & \uparrow \\ & & \text{do } \psi \text{ rozstrzyga} \\ & & \text{problem } K \\ k \notin K & \Leftrightarrow & k \notin K \end{array}$$

K nie jest rekurencyjny.

K jest r.e.: pytamy o x , odnajdujemy $\varphi_x(x)$ i jeśli się zatrzymie to OK.

↓ wykładek: A r.e. \wedge \bar{A} r.e. \Rightarrow A rek i \bar{A} rek.

Mamy K r.e. \wedge $K \notin$ rek. Czyli \bar{K} nie jest r.e.

Czyli cała tabelka z funkcjami NIE WIEMT, czy się zatrzyma.

JFIZO zad 85

$$A = \{n \mid |\text{Dom}(\varphi_n)| \geq 7\}$$

moeżliwości argumentów, dla których φ_n są obliczony

CEL: zbudować program t.z. $\text{KEA} \Leftrightarrow \Psi_A(k) = 1$

czyli program zwróci 1 jeśli φ_k jest obliczalna dla co najmniej 7 liczb naturalnych (φ_k - funkcje z tabelki)

φ_i	1	2	3	4	5
$\varphi_0(0)$					
$\varphi_0(1)$					
$\varphi_0(2)$					
$\varphi_0(3)$					
$\varphi_0(4)$					

POMYSŁ: Tak w Cantorze.

Po kolejnych przekształceniach

- * na każdym argumentacie

- * na każdym czasie

→ nie wie oznaczenia

→ nie zauważymy wywołania nie zakończonego

NA RYSUNKU $0,1,3 \in \text{Dom}(\varphi_n)$

w ∞ .

Program dla A:

```

Ψ_A(n) = ψ_n ← get-f(n) // z tabelki
while
if domain = φ // domenina
for i=0 to ∞
    for t=1 to i+1
        if (i-t+1) not in domain
            WAIT(t, ψ_n(i-t+1))
            domain.add(i-t+1) // dla argumentu
                                // i-t+1 udostępnione
                                // są obliczalne w orasie
                                // t sekund
        if domain.size() ≥ 7
            return 1
    
```

JFIZO zad 86

A,B,C,D są re. ciągły mamy $\varphi_A, \varphi_B, \varphi_C, \varphi_D$ t.z. zwracają 1 jeśli kierując się zapisem zapisu zwracają 1. Wtedy kierując się do kolejnymi dwóch.

Pokazujemy, że $\exists \varphi_A, \varphi_B, \varphi_C, \varphi_D$ t.z. zwracają 1 lub 0.

$$\varphi_A^1(k) =$$

for m=1 to ∞

: WAIT(m, $\varphi_A(k)$)

: return 1 // jeśli mamy do A to zwrócić 1

: sum=0

: foreach x in {B,C,D}

: WAIT(m, $\varphi_x(k)$)

: sum+= // mamy do jednego z nich zakończony

: if sum == 2

: return 0 // jeśli mamy do dwóch innych
 to we powrocie mamy do A
 więc zwrócić 0.

Analogicznie $\varphi_B^1, \varphi_C^1, \varphi_D^1$.

ZFIZO zad 87

(ZWF)

φ -wennalgebra f całk. rek $\Rightarrow \varphi(N)$ jest zbiorem rekurencyjnym
(czyli die sie sieć napisów programu Ψ zw. 0 lub 1)

1° Jeśli φ jest stała od pewnego momentu to ZWF jest skończony.

Ψ : pierwszy sieć po całk. zbiór szukając tam n.

2° Ψ : weryfikuj n

for $m=0$ to ∞

if $\varphi(m) = n$ return 1 // $n \in \text{ZWF}$

if $\varphi(m) > n$ return 0 // f jest wennalgebra czyli już
na pewno nie zakończony.

 ↓

φ całk. rek - zawsze się dolicza

φ wennalgebra ciągłaowa funkcja rek \Rightarrow ZWF jest rek?

NIE, pokazany kontrapozycją.

$$\varphi(n) = \begin{cases} n & \text{jeli } \varphi_n(n) \in N \\ 1 & \text{wpp.} \end{cases}$$

φ -funkcja rekurencyjna to istnieje program w MUFP ją obliczająca.

monotoniczna, bo działa jak identyczność $a \leq b \Rightarrow \varphi(a) \leq \varphi(b)$
(czyli się dolicza)

PRZECIWDELIGRZINA: K, czyli zbiór, który nie jest rekurencyjny.

88 A niepusty i r.e

Znalezienie f całkowita rekurencyjna t.z. $f(N) = A$, czyli:

$n \in A \Rightarrow \exists x \ f(x) = n$

$n \notin A \Rightarrow \forall x \ f(x) \neq n$

Ψ : - weryfikuj n

- cnt := 0

- for $k = 0$ to ∞

 * $i, j = b(k)$ // kolejnego po precedencych

 * odpal $\varphi_A(i)$ na j sekund

 * jeśli zwróciło wynik:

 > cnt += 1

 > if cnt = n

 return i

f jest całkowita i rekurencyjna, bo program Ψ zawsze się zatrzyma, bo skoro A jest niepuste, to $\exists x \varphi(x)$ odpalone na y sekund się zatrzyma, więc w końcu f zwróci chwilią x.

$f(N) = A$?

$n \notin A \Rightarrow \forall x \ f(x) \neq n$, bo funkcja narasta, i" tylko jeśli $\varphi_A(i)$ się zatrzymie czyli jeśli $\varphi_A(i) = 1$ to napisy nie zakończą i".

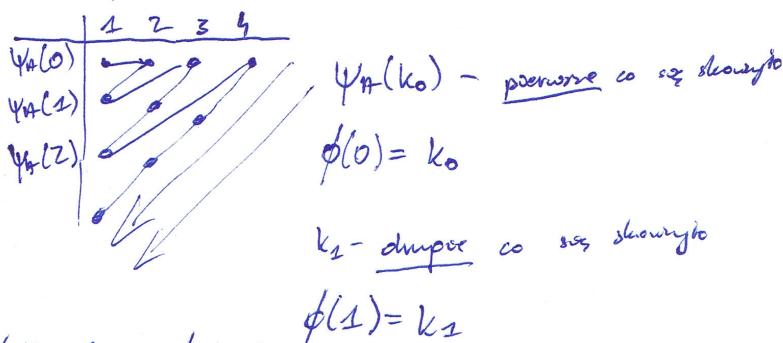
$n \in A \Rightarrow \exists x \ f(x) = n$, wtedy np $x = 3, 3 \in A$, $\varphi_A(3)$ zatrzymuje się po 2 sekundach

$\varphi_A(1)$	1	2	3	4
$\varphi_A(2)$	•	•	•	•
$\varphi_A(3)$	•	•	•	•
$\varphi_A(4)$	•	•	•	•

$\varphi_A(3)$ się zatrzymuje odpalać dla $k=3$, więc $k \neq 2$
dla $j \geq 3, j = b(k)$ i $j \geq 2$ wynik będzie 3 ✓
czyli $\exists x \ f(x) = n$ ■

* $\forall i \in \text{wzestchny}, \exists n \in \text{wzestchny} \text{ taki, } \text{ZWF do } A.$

Ψ_A - program seme-vrozstnyjacy A.



for $i = 2$ to ∞

dowolny do losy,
jedna losa ma n elementow so
zawierajacy n -ty element

MFIZO zad 90

$$A = \{n \mid \text{Dom}(\varphi_n) = \mathbb{N}\} \quad \text{nde jest r.e.}$$

A - indeksy funkcji określających wszystkie losy

Zad. dc A jest r.e.

Wtedy istnieje program Ψ_A .

$$\varphi_A(n) = \begin{cases} 1 & \Leftrightarrow n \in A \text{ oryg } \forall k \varphi_n(k) \text{ sie oblicza} \\ 1 & \Leftrightarrow n \notin A \text{ oryg } \exists k \varphi_n(k) = 1 \end{cases}$$

Program Ψ :

- * wczytaj n
- * napisz sobie taki program
 - > wczytaj m
 - φ_K {
 - if $\varphi_n(m)$ zwraca wynik po m sekundach
 - zapetl sie
 - > else return 1
- * oblicz $\varphi_A(k)$ i zwroc wynik

1° $n \in \bar{K} \Rightarrow n \notin K \Rightarrow \varphi_n(m) = 1 \Rightarrow \exists m \text{ ie } \varphi_n(m) \text{ sie zliczaj po m sec}$

$\Rightarrow \forall m \varphi_n(m) = 1 \Rightarrow \text{Dom}(\varphi_K) = \mathbb{N} \Rightarrow k \in A \Rightarrow \varphi_A(k) \text{ sie oblicza}$

$\Rightarrow \Psi$ sie zliczaj

2° $n \notin \bar{K} \Rightarrow n \in K \Rightarrow \varphi_n(m) \text{ zliczaj sie so } \Rightarrow \exists m \text{ ie } \varphi_n(m) \text{ zliczaj sie po m sec}$

$\Rightarrow \forall m \varphi_n(m) \text{ sie zapetla} \Rightarrow \text{Dom}(\varphi_K) \neq \mathbb{N} \Rightarrow k \notin A \Rightarrow$

$\varphi_A(k) \text{ sie zapetla} \Rightarrow \Psi = 1$

GZPLI: Ψ seme-vrozstnyjacy \bar{K} , a \bar{K} nie jest r.e. SPRZECZNOST.

(\bar{K} nie r.e., bo K r.e. i K nie relk)

- 91
- 1) A role to $f(A)$ role? other
2) A role to $f^{-1}(A)$ role? preimage
3) $x \in$ \vdash \vdash
4) $x \in$ \vdash \vdash
- a) esteosuite
b) joneses

1a) nie, ziel 88
2a) $f^{-1}(A) \subseteq$ role A TAK (f jest biżekcja)

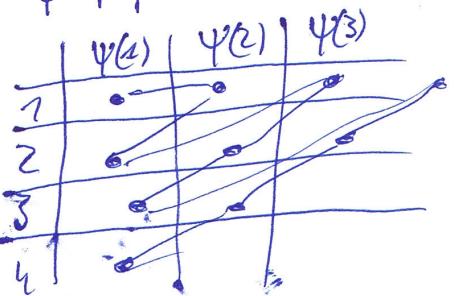
2b) $f(x) = \psi_x(x)$, $f^{-1}(N) = K$ cyklo NIE
 $f_{\emptyset} \in N \Leftrightarrow x \in K$
 $f_{\emptyset} = \perp \Leftrightarrow x \notin K$

3b) $A = \psi(N)$, $f(A) = f \circ \psi(N)$
 ↑ funkcja wklasowana
 relacyjne wklasowanie

czy $x \in f(A)$?

TAK wtedy $\exists a$ taki

4b) $x \in f^{-1}(A) \Leftrightarrow (x \in f^{-1}(N) \wedge f(x) \in A)$
 - obliczamy $f(x)$ // jeśli się wejdzie do konicznej
 fukcji $\psi(1)$ i powstanie $\geq f(x)$
 iż $\psi(1)$ jest funkcją
 o starym



92 $B = \{n \mid \text{Dom}(\varphi_n) \subset \mathbb{N} \setminus \text{Dom}(\varphi_m) \text{ i } \varphi_n \text{ nie skonczona}\}$

\rightarrow dla n wówczas φ_n zdefiniowana

\rightarrow dla n wówczas \perp

B nie jest rekurencyjny.

$K \subseteq_{\text{rek}} B$

f: - wczytaj n

- napisz program:

* wczytaj m
 * jeśli $m \% 2 = 1$ return 1
 * jeśli $\varphi_n(m)$ skończone po m sekundach
 ↳ zapamiętaj skończone
 * else return 1

- zwróć numer TEGO programu.

$n \in K \Rightarrow \varphi_n(n)$ skończone po \times krokach $\Rightarrow \exists m > \times \varphi_{f(n)}(m)$ skończone po \times krokach

i $\forall m$ nieparzyste $\varphi_{f(n)}(m) = 1 \Rightarrow f(n) \in B$

$n \notin K \Rightarrow \varphi_n(n)$ nieskończona $\Rightarrow \varphi_{f(n)}$ zawsze zwraca 1

czyli $\text{Dom}(\varphi_{f(n)}) = \mathbb{N}$ oraz $\mathbb{N} \setminus \text{Dom} = \emptyset$ - skontraryjnie $\Rightarrow f(n) \notin B$

93 $B = \{n \mid \text{Dom}(\varphi_n) \subset \mathbb{N} \setminus \text{Dom}(\varphi_m) \text{ i } \varphi_n \text{ nie skonczona}\}$ nie jest r.e.

$K \subseteq_{\text{rek}} B$

f: - wczytaj n

- napisz program:

* wczytaj n
 * jeśli $n \% 2 = 1$ return 1
 * jeśli $\varphi_n(n)$ zakończenie skończone po n sekundach
 ↳ return 1
 * else zapamiętaj skończone

- zwróć numer TEGO programu

$n \in K \Rightarrow \varphi_n(n)$ skończone $\Rightarrow \varphi_{f(n)}$ dla parzystych n zapamiętała
 a dla nieparzystych zwraca 1 $\Rightarrow n \in B$

$n \notin K \Rightarrow \varphi_n(n)$ nieskończona $\Rightarrow \forall m > n \varphi_{f(n)}(m) = 1$
 $\Rightarrow \mathbb{N} \setminus \text{Dom}(\varphi_{f(n)})$ jest skończone $\Rightarrow f(n) \notin B$.

Kad 94 $B = \{n \mid \varphi_n \text{ zatrzymuje się dla wszystkich arg opierających się skończonych liczb}\}$

Dla wejścia do post r.c. \Rightarrow wtedy zatrzymuje się semi-rozstypiąca B .

Piszę tego programu Ψ , który semi-rozstypia \overline{K} .

Ψ : - wczytał n

- napisz program:

- * wczytaj n
- * odpal $\varphi_n(n)$ na n sekund
 - \rightarrow ustaw do 0
 - \rightarrow zapisz do 0
 - \rightarrow nie ustaw \rightarrow return 1

} Ψ_K

- k -numer tego programu

- zwrot $\Psi_B(k)$

1° $n \in \overline{K} \Rightarrow n \notin K \Rightarrow \varphi_n(n)$ nigdy się nie zatrzymuje $\Rightarrow \varphi_n$ zatrzymie zawsze 1

$$\Rightarrow \varphi_B(k) = 1 \Rightarrow \psi(n) = 1$$

2° $n \notin \overline{K} \Rightarrow n \in K \Rightarrow \varphi_n(n)$ się zatrzymuje po x sekundach $\Rightarrow \exists m \in \mathbb{N} \times \varphi_n(m) = 1$

$$\Rightarrow \text{dla innych arg } \varphi_K \text{ się nie zatrzymuje} \Rightarrow \varphi_B(k) = 0 \Rightarrow \psi(n) = 1$$

Ψ semi-rozstypia \overline{K} , a \overline{K} nie jest r.c!

] F120 zad 95

$$\exists C \text{ i.e. } A \subseteq_{\text{relk}} C \wedge B \subseteq_{\text{relk}} C \wedge \forall D. A \subseteq_{\text{relk}} D \wedge B \subseteq_{\text{relk}} D \Rightarrow C \subseteq_{\text{relk}} D$$

1

2

$C = A \times \{0\} \cup B \times \{1\}$ - bieżący wiersz, z którego zakończy pasek
element: $0 \rightarrow A, 1 \rightarrow B$

1) $A \subseteq C \wedge B \subseteq C$

czyli istnieje funkcja

$$f_{AC}(n) = \langle n, 0 \rangle$$

$$f_{BC}(n) = \langle n, 1 \rangle$$

$$n \in A \Leftrightarrow f_{AC}(n) \in A \times \{0\} \cup B \times \{1\} = C$$

$$\Leftrightarrow n \in A \Rightarrow f_{AC}(n) \stackrel{\text{def}}{=} \langle n, 0 \rangle \in A \times \{0\} \subseteq C \quad \checkmark$$

$$\Leftrightarrow f_{AC}(n) \in C \Rightarrow f_{AC}(n) \in A \times \{0\} \Rightarrow n \in A \quad \checkmark$$

dla f_{BC} analogicznie

2) Dla dowolnego D i.e. $A \subseteq D : B \subseteq D$ chcemy zbudować f_{CD}

i.e. f_{CD} jest zbiorem $C \subseteq D$. Należy f_{CD} jest prop. oznaczającym f_{CD} .

f_{CD} : wczytaj (n, i)

if $i = 0$ return $f_{AD}(n)$ // istnieje bo $A \subseteq D : B \subseteq D$

if $i = 1$ return $f_{BD}(n)$ // istnieje bo $A \subseteq D : B \subseteq D$

$$\langle n, i \rangle \in C \Leftrightarrow f_{CD}(\langle n, i \rangle) \in D$$

1° $i = 0$

$$\langle n, 0 \rangle \in C \Leftrightarrow f_{CD}(\langle n, 0 \rangle) \in D$$

bo dropując \downarrow
 \downarrow do D \Downarrow z ief. fukcji oblicz. \Downarrow z ief. fukcji oblicz. \Downarrow
 $n \in A \Leftrightarrow f_{AD}(n) \in D$ program f_{CD}

2° $i = 1$ - ANALOGICZNIE

$$\overline{K} \leq_{rek} K$$

NIE. Zat z teb, wtedy istnieje funkcja f t.j. $\forall n \in \overline{K} \Leftrightarrow f(n) \in K$

Napiszmy program Ψ , który semirozstępuje \overline{K}

Ψ : * wczytaj n
 * $\varphi_K(f(n))$ // φ_K -program semirozstępujący K
 * ret 1

$$1^{\circ} n \in \overline{K} \Rightarrow f(n) \in K \Rightarrow \varphi_K(f(n)) = 1 \Rightarrow \Psi(n) = 1$$

$$2^{\circ} n \notin \overline{K} \Rightarrow f(n) \notin K \Rightarrow \varphi_K(f(n)) = \perp \Rightarrow \Psi(n) = \perp$$

↳ bo \overline{K} nie jest r.e.

$$K \leq_{rek} \overline{K}$$

NIE. Zat, ze teb, wtedy istnieje funkcja f t.j. $\forall n \in K \Leftrightarrow f(n) \in \overline{K}$

$$n \in \overline{K} \Leftrightarrow n \notin K \Leftrightarrow f(n) \notin K \Leftrightarrow f(n) \in K$$

$$n \in \overline{K} \Leftrightarrow f(n) \in K$$

a w tym pojęciu poznaliśmy, że teka funkcja nie istnieje.

↳

INNE ROZW:

$K \leq_{rek} \overline{K}$? zat z teb, wtedy istnieje rozstępy f , t.j.
 $n \in K \Leftrightarrow f(n) \in \overline{K}$

Ψ : - wczytaj n
 - otwórz $k = f(n)$
 - odpal rozstępy $\varphi_n(u) : \varphi_k(k)$ // który zawsze zwroti
 \hookrightarrow jeśli φ_n zawsze zwroti wynik ret 1
 \hookrightarrow jeśli φ_k zawsze zwroti wynik ret 0

Ψ rozstępuje \overline{K} , ↳

$$n \in K \Leftrightarrow k \in \overline{K} \text{ zycie } \varphi_n(u) \text{ stycie} \Leftrightarrow \varphi_n(k) \text{ nie stycie}$$

$$\varphi_n(u) \text{ stycie} \Rightarrow \varphi_n(k) \text{ nie stycie}$$

$$\varphi_n(u) \text{ nie stycie} \Rightarrow \varphi_n(k) \text{ stycie}$$

$\overline{K} \leq_{rek} K$ - anektogram konstrukcja Ψ i rozumowanie.

JF120 and 97 (my own program dealing the same)

a) $T = \{ \langle n, m \rangle \mid \varphi_n \vdash \varphi_m \text{ in the same function signature} \}$
 (crys $\vdash \varphi_n(x) = \varphi_m(x)$ lub \perp)

The first v.e.

Zat we wprost,że T jest n.e. Wtedy istnieje φ_T semi-rozstrzygający T.

Program Ψ semi- ψ -string $\bar{\psi}$

Ψ : wingen n

* napoří program

q_0 {
 > $w_{\text{copy}} m$
 > if $m = 100$ loop forever
 > wait m sec for $q_n(n)$
 if stop \rightarrow loop forever
 else return 1

* regular program

Ψ_6 } \rightarrow wrong by m
 \rightarrow if $m = 100$ wrong power
 \rightarrow between 1

* return $\Psi_{\pm}(a, b)$

1° $n \in \mathbb{K} \Rightarrow \varphi_{\alpha(n)}$ many sige wie zetnig nje $\Rightarrow \varphi_n$ i ψ_b zwisch 1

jeśli wartości m upierzą 100, to gdy są równe $\Rightarrow \varphi_i(a,b) = 1 \Rightarrow \psi = 1$

$\exists^{\infty} n \notin K \Rightarrow n \in \mathbb{N} \Rightarrow \varphi_n(n)$ są zatem niewiększe niż w sec $\Rightarrow \varphi_n(x) = 1, \varphi_0(x) = 1$

$$\Rightarrow \varphi_+(a, b) = 1 \Rightarrow \psi = 1$$

↳ crypt T was just r.e.

b) $\overline{T} = \{ \langle n, m \rangle \mid q_n \text{ is } q_m \text{ to some finite extension} \}$ we just r.e.

Każda wypowiedź \bar{T} jest r.e. Wtedy istnieje yf semi-rozstugojęza \bar{T} .
 Program y semi-rozstugojęza \bar{K} .

$\psi = \star w \circ \phi \circ \gamma$

* napisz program

> wavy
> wavy

$\varphi_2 \left\{ \begin{array}{l} > \text{if } m = 100 \\ > \text{else ret 1} \end{array} \right.$

* major proposal

{ > wortelj n

$\psi_6 \left\{ \begin{array}{l} > \text{if } m = 100 \text{ loop forever} \\ > \text{else ret 1} \end{array} \right.$

* return $\varphi_T(a, b)$

1° $n \in \mathbb{K} \Rightarrow \psi_n(n) \text{ sige we zehnigwe} \Rightarrow \psi_n \text{ sige we zehnigwe, } n^{\varphi_0} = 1$

$$\Rightarrow \psi_{\bar{F}}(a, b) = 1 \Rightarrow \psi = 1$$

$$2^{\circ} n \notin K \Rightarrow \varphi_n(x) \text{ sie zwingt } \Rightarrow \forall x \varphi_n(x) = \varphi_b(x) \geq 1 \Rightarrow \text{sie minne}$$

$$\Rightarrow \psi_F(a, b) = 1 \Rightarrow \psi = 1$$

↳ be \overline{K} we jest r.e., only \overline{F} we jest r.e.

$\Pi_0 = \Sigma_0$ bo dopełnianie rek jest rek

$\Sigma_1 = \text{r.e.} - \text{to wypadek } z \text{ zad 83 : 82}$

$\Pi_1 = \text{co-r.e.}$

$L = \{ n \mid |\text{Dom}(\varphi_n)| \in \mathbb{N}^+ \}$ - skończona niepusta skupisko

$1^{\circ} L \in \Sigma_2$

$B = \{ \langle n, k \rangle \mid k = f(i, j) \wedge \varphi_n(\cdot) \text{ zatrzymuje się po } j \text{ sekundach} \}$

$\forall m \exists i \exists j \varphi_n(i) \text{ się nie zatrzymuje}$

$\Pi_1(B) = L$ //nit we pozwolił os

B jest core

$\rightarrow \bar{B}$ jest r.e., da się napisać program

$2^{\circ} L \notin \Sigma_1$ (nie jest r.e.)

program: * wstęp' n

* napisać program

wstęp' k

if $k=1$

ret 1

else ret $\varphi_n(n)$

* ret k

f_n - rekurencja

$\bar{K} \subseteq_{\text{rek}} L \rightarrow x \in \bar{K} \Leftrightarrow f(x) \in L$

$1^{\circ} x \in \bar{K} \rightarrow$ dwoistwe siedzenie {1} $\rightarrow x \in L$

$2^{\circ} x \notin \bar{K} \rightarrow$ dwoistwe IN \rightarrow niekoniecznie $\Rightarrow x \notin L$

$L = \{ n \mid \text{Dom}(\varphi_n) \text{ jest niepusta i skończona} \}$
 $= \{ M \mid \exists n \exists k \exists t_k \forall h \forall t \ k \leq_m M(h) \text{ staje po } t_k$
 \uparrow
 zbiór maszyn
 $\wedge M(n) \text{ nie staje po } t \}$

$\exists x \exists y \exists z \forall y_1 \forall y_2 \exists x \forall y \quad \text{odpowiedniemu bijekcjiom.}$

$L \in \Sigma_2$ bo rozszerzony jako $\exists A$.

$L \notin \Sigma_1$ bo $\bar{K} \in \Pi_1$ i $\bar{K} \not\subseteq L$

$\bar{K} \subseteq_{\text{rek}} L$

- wstęp' n

- napisać program:

* wstęp' m
* if $m=0$ return 1 // niepotrzebne
* odpat $\varphi_n(n)$ na m kroki
 ↳ jeśli stanie do return 1
 ↳ jeśli nie stanis do się zapeł.

- zwróci numer TBO programu.

$n \in \bar{K} \Rightarrow \varphi_n(n) \text{ nie staje} \Rightarrow \text{Dom}(\varphi_{f(n)}) = \{0\} \Rightarrow f(n) \in L$

$n \notin \bar{K} \Rightarrow \varphi_n(n) \text{ staje po } x \text{ krokach} \Rightarrow \forall j \in \mathbb{N} \times \varphi_{f(n)} \text{ staje}$

czyli $\text{Dom}(\varphi_{f(n)})$ jest niekoniecznie $\Rightarrow f(n) \notin L$

JFIZO zad 9g

$A \leq_{rel} B$, $f: A \rightarrow B$ jest "na" cykliczny $\forall y \in B \exists x \in A f(x) = y$

Pokazać $B \leq_{rel} A \rightarrow$ skonstruowanie g + zr $\forall n \in B \Leftrightarrow g(n) \in A$

φ_g : wypisz y

for $x=0$ to ∞

if $f(x) = y$ ret x // takie x istnieje, bo f jest "na"

Dowód, że g jest rekurencyjny:

1° $y \in B \Rightarrow f(x) = y \Rightarrow x \in A$
// bo f jest "na" $\forall n \in B \Leftrightarrow f(n) \in A \wedge$ wypisz $\varphi_g = f(n) \in B$

2° $y \notin B \Rightarrow f(x) = y \Rightarrow x \notin A$

JF120 zad 100

$$\mathcal{B} = \{n \mid \text{Dom}(\varphi_n) = \{1, 2, \dots, m\} \text{ albo } \text{dom}(\varphi_n) = \emptyset\}$$

a) \mathcal{B} core?

NIE. \mathcal{B} core $\Leftrightarrow \overline{\mathcal{B}}$ r.e.. Pokażemy, że $\overline{\mathcal{B}} \leq \text{ren } \overline{\mathcal{B}}$, a $\overline{\mathcal{B}}$ nie jest r.e.

$\psi_{f \in \overline{\mathcal{B}}}^*$: * wczytaj n

* napisz siebie program:

$\varphi_k \left\{ \begin{array}{l} > \text{wczytaj } m \\ > \text{if } m=2 \text{ ret } 1 \\ > \text{if } m=1 \text{ ret } \varphi_n(n) \\ > \text{loop forever} \\ > \text{return } k // \text{jako f(n)} \end{array} \right\}$

Zauważ, że obiektami φ_k są $\{1, 2\}$ albo $\{2, 3\}$ w zależności od n.

1° $n \in \overline{\mathcal{B}} \Rightarrow f(n) \in \overline{\mathcal{B}}$ bo $\text{Dom} = \{2, 3\}$ // dla 1 są rozpatrywane pary $\varphi_n(n)$

2° $n \notin \overline{\mathcal{B}} \Rightarrow f(n) \in \mathcal{B}$ bo $\text{Dom} = \{1, 2\} \Rightarrow f(n) \notin \overline{\mathcal{B}}$

Zatem $\overline{\mathcal{B}}$ nie jest r.e. ergo \mathcal{B} nie jest core.

b) A jest core $\Leftrightarrow \exists \psi \text{ t.j. } \psi(\langle n, m, t \rangle) = 1 \Leftrightarrow \langle n, m, t \rangle \in A$

PRZYKŁAD: $A = \{\langle n, m, t \rangle \mid \varphi_n(1), \varphi_n(2), \dots, \varphi_n(m) zaliczają się po max t sekundach i \forall i > m \varphi_n(i) = 1\}$
nagdyż nie zwaca!

A core $\Leftrightarrow \overline{A}$ re $\Leftrightarrow \exists \varphi_{\overline{A}}$ sans rozstępy pomiędzy

Zbudujmy $\varphi_{\overline{A}}$:

* wczytaj n, m, t

* uruchom $\varphi_n(1), \varphi_n(2), \dots, \varphi_n(m)$ na t sekund

	1	2	3	*
$\varphi_n(m+1)$	1	2	3	4
$\varphi_n(m+2)$	2	3	4	
$\varphi_n(m+3)$	3	4		
$\varphi_n(m+4)$	4			

* jeśli któryś nie zwroci wyniku to ret 1 // $\notin A$

* odczytaj po kolejnych sekundach jeśli któryś zwroci to ret 1 // $\notin A$

(tac nie zwieracmy)

RZUT: $\{\langle n, m, t \rangle \mid \dots\} = A$

$\{n \mid \exists m, t. \langle n, m, t \rangle \in A\} = B$

TAK, bo są to numery programów, których domeny to $\{1, \dots, m\}$

te $\varphi_n(1), \varphi_n(2), \dots, \varphi_n(m)$ są OK

or $\varphi_n(m+1) = \varphi_n(m+2) = \dots = 1$

Zad 102 Gdy $g(n) = \min\{k \mid f(k) = n\}$ jest w:

a) funkcja rek i jest "na"

TAK. Napiszemy program dla g:

φ_g - wynik g m

- for $n=0 \rightarrow \infty$

if $f(n)=m$ return s

jeśli $w_{\text{f}} > 0$ zwracaj m

i f jest "na" zwracaj zgodnie z kolejnością n

$\rightarrow g$ jest funkcja rek.

b) f ciągowa rek i jest "na"

NIE. Komentarz:

$$f(n) = \begin{cases} \frac{n}{2} \text{ jeśli } n \text{ parzyste} & \varphi_{\frac{n}{2}}(\frac{n}{2}) = 1 \\ \frac{n+1}{2} \text{ jeśli } n \text{ nieparzyste} & \varphi_{\frac{n+1}{2}}(\frac{n+1}{2}) = 1 \end{cases}$$

$\rightarrow f$ jest ciągowa (może się powtarzać)

$\rightarrow f$ jest "na" \rightarrow zwraca $0, \frac{1}{2}, \frac{1}{4}, \dots$ dla nieparzystych

Zad. dc f jest rekurencja \rightarrow stwierdz ψ

ψ - wynik g n

- odpal g(n)

\rightarrow jeśli wynik parzysty, to zwracaj 1

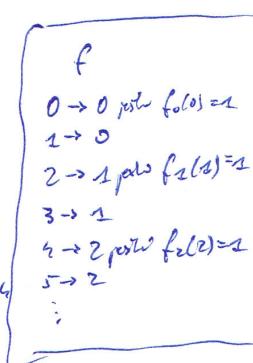
\rightarrow jeśli nieparzysty, to zwracaj 0

ψ jest stała K

$n \in K \Rightarrow \varphi_n(n) = 1 \Rightarrow g(2n) = \text{parzyste} \Rightarrow \psi(n) = 1$

$n \notin K \Rightarrow \varphi_n(n) = 1 \Rightarrow g(2n) = \text{nieparzyste} \Rightarrow \psi(n) = 0$

\therefore g nie jest n.e.



Zad 103

D pełnorozłączny $\Leftrightarrow \exists B \in \mathbb{N} \forall A \in D \quad A \not\subseteq B$

(\Rightarrow) D pełnorozłączny $\Rightarrow D = \{A_1, A_2, A_3, \dots\}$ (mamy numerowane A_i)

Pocharemy B, konstrukcja jak w zad 95

\rightarrow index, z ktorego otrzymamy pochodzący element

Dla A_i mamy f_i : $f_i(n) = \langle n, i \rangle \in A_i$

φ - bijekcja $\mathbb{N}^2 \rightarrow \mathbb{N}$

$$B = \bigcup_{i \in \mathbb{N}} \{\varphi(f_i(n)) \mid n \in A_i, A_i \in D\}$$

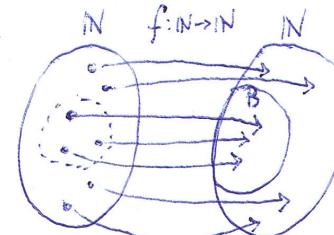
$\forall A_i \quad A_i \not\subseteq B \rightarrow$ redukcja: ~~$\varphi \circ f_i$~~ $\varphi \circ f_i$

$\forall x \quad x \in A_i \Leftrightarrow \varphi(f_i(x)) \in B$

\rightarrow to wynika wprost z konstrukcji zbiuru B.

(\Leftarrow) Wskazówka: Dla danego B i f jest tylko 1 zbiór A.

Czyli?



jak mamy $f: B \rightarrow A$ i f jest taka że $\forall x \quad x \in A \Leftrightarrow f(x) \in B$
to mamy określone które $x \in A$.

$$A = f^{-1}(B) \quad // \text{przeciwstaż}$$

Mamy B i jakieś funkcje f, które są redukcjami więc całkowitymi funkcjami rek, a ich jest PIZZELCZALNIE WIELE.

Każdej f odpowiadają 1 A, czyli zbiory A tąż jest PIZZELCZALNIE WIELE

Czyli D jest pełnorozłączny.

Zad 104 $Tot = \{n \mid \text{Dom}(\varphi_n) = N\}$ - zbiór liczb zdefiniowanych
 $Nump = \{n \mid \text{Dom}(\varphi_n) \neq \emptyset\}$ - zbiór liczb nie zdefiniowanych

a) $Nump \subseteq_{rek} Tot$ - TAK

Skonstruujemy rekurencję:

$f = \{ \text{wyszczególnij } n \}$

- napisać program:

* wyróżnić n

* umówić $\varphi_N(n)$

jakość rekurencyjną do zapisu - 1.

} φ_k

- k - numer tego programu

- zapisz k

1° $n \in Nump \Rightarrow \varphi_n(n)$ jest zdefiniowany $\Rightarrow \varphi_k$ zawsze zwraca 1
 $\Rightarrow \text{Dom}(\varphi_k) = N \Rightarrow k = f(N) \in Tot$

2° $n \notin Nump \Rightarrow \varphi_n(n)$ nie jest zdefiniowany $\Rightarrow \text{Dom}(\varphi_k) = \emptyset$
 $\Rightarrow k = f(N) \notin Tot$

b) $Tot \subseteq_{rek} Nump$ - NIE

Pokażemy, że Tot nie jest r.e. (zad 80)

$Nump$ jest r.e.

ZF120 zad 105

Które f rekurencyjne zapisować $\subseteq f'$ rekurencyjna całkowite?

NIE

$f = \{(x, y)\}$ - zbiór par

$f \subseteq f' \Leftrightarrow \forall (x, y) \quad (x, y) \in f \Rightarrow (x, y) \in f'$

Niech $f(n) = m \Leftrightarrow \varphi_n(n)$ kończy się po m sek.

f jest rekurencyjne zapisaną w istnieje program obliczający f

$\varphi_f =$

* wyróżnić n
* for $i = 0$ to ∞
if φ_n skończyła się po i sek
return \boxed{i} ;

Zauważ, że $\exists f'$ rekurencyjna całkowite - INTUICJA: oznacza, że f nie zakonczy działania i zwraca coś swego typu - to niemożliwe!

Napiszemy program, który rozstępuje K

$\psi =$ + wyróżnić n

* odpali $\varphi_n(n)$ na $f'(n)$ krokach

* jeśli zwróciło wynik to ret 1
else ret 0

1° $n \in K \Rightarrow \varphi_n(n)$ się kończy $\Rightarrow f'(n)$ zwraca wynik K $\Rightarrow f'(n) = K \Rightarrow \psi(n) = 1$

2° $n \notin K \Rightarrow \varphi_n(n)$ się nie kończy $\Rightarrow f'(n) = K$ to jest zdefiniowane dla $n \in K$

$\Rightarrow \varphi_n(n)$ się nie skończy po K krokach $\Rightarrow \psi(n) = 0$

a K nie jest rek.

Zad 111 MT stop się 2 MSkana stop

Identyczne jak w poprzednim zadaniu.

1) $\langle q_0, \alpha, q_1, \alpha, R \rangle \rightarrow$ maszyna nie idzie w przód, robimy ją w MT

2) $\langle q_0, \alpha, q_1, \alpha, L \rangle \rightarrow$ * dojedź do końca

* odjeżdżaj (wstecznie, bo będzie 'blank')

* dojedź do początku (zatrzymaj, al')

→ idea ta sama jak w poprzednim zadaniu
z przesunięciem koniunktury

(także algorytm jest taki sam jak dojedź do końca
i do początku)



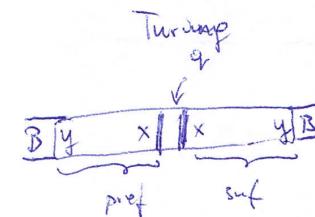
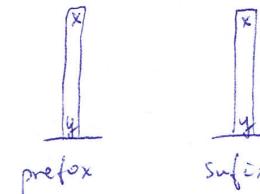
Zad 112 Turing → Stosy

a)



dwie stosy

Minsky



Równoważne maszyny Turinga ✓

b)	stosy → głoszki	system dwójkowy
$\Sigma = \{0, 1\}$	$q_0, q_1, q_2, \dots, q_n$	$2^{\Sigma} \rightarrow \text{system N-ary } N$

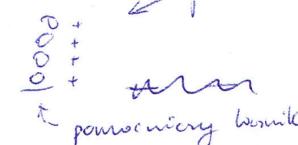
system dwójkowy $2^{\Sigma} \rightarrow$ system N-ary N
 $\Sigma = \{0, 1\}$ → stos jako licznik

$$|\Sigma| = N$$

indukcyjne:



→ jeden stos - 2 liczniki



$$2^{\Sigma^{\Sigma^{\Sigma^{\Sigma}}}} \mid 5^{\Sigma^{\Sigma^{\Sigma^{\Sigma}}}}$$

$$2^{\Sigma^{\Sigma^{\Sigma^{\Sigma}}}} \mid 3^{\Sigma^{\Sigma^{\Sigma^{\Sigma}}}}$$

Licznik → indeksowe, odkrywacze;
spojery, pushy

c) 4 berulek \rightarrow 2 berulek



$$2^{l_1} 3^{l_2} 5^{l_3} 7^{l_4}$$

$$\sum^N 2^{l_i} \alpha_i = 2 + \dots$$



Czy l_1 jest parzysty \rightarrow skróć 2 berulek po 2 elementy
nie równocześnie

odd

$$\begin{array}{r} 0 \\ 0 \\ 0 \\ 0 \\ \hline 0 \end{array}$$

$$\times \{2, 3, 5, 7\}$$

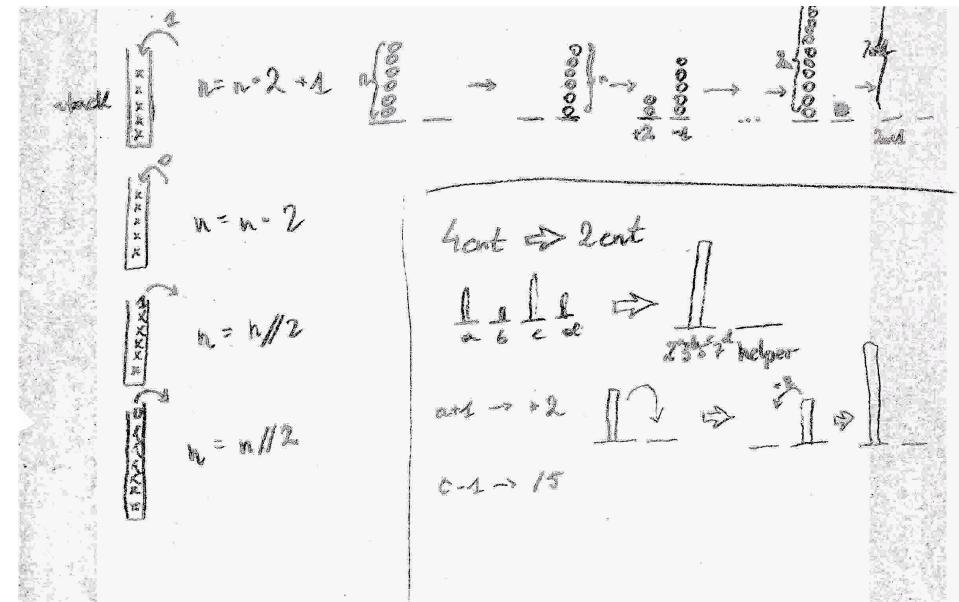
even

$$\begin{array}{r} 0 \\ 0 \\ 0 \\ 0 \\ \hline 0 \end{array}$$

$$\div \{2, 3, 5, 7\}$$

Wnioski: LaTeX jest Turing zupełny dla nie berulek

Czyli nie są zdecydowane Maszyny Minskiego



JFIZO zad 14

$$\text{PCP} \subseteq_{\text{rec}} [L_G \cap L_H \neq \emptyset]$$

Wyraźnie, czyli istnieje wspólna slowo

Mamy $P = \{< l_1, r_1 >, \dots, < l_k, r_k >\}$. Rozważmy redukcję:

$$G = S \rightarrow \# l_1 S^1 a_1 \quad H = S \rightarrow r_1 S^1 a_1 \\ S^1 \rightarrow \epsilon \mid l_1 S^1 a_1 \quad \& \quad S^1 \rightarrow \epsilon \mid r_1 S^1 a_1$$

Po co nam specjalny symbol, a_i^1 ?

$\langle a, ab \rangle$

$$\langle b, ab \rangle \Rightarrow l_1 l_2 = r_1 \quad (ab = ab) \rightarrow ab x_2 x_1 \neq ab x_1$$

WNIĘTEK - kiedyś miałem pod
 $a_i^1 S^1 r_i \rightarrow$ to nieporządku
 indeksy, więc zauważet
 że slowem po tym iżelki,
 ciągle to jest DCFL
 ale potem zmienią reakcje
 to co zwrócić

Dowód, że to redukcja:

$$(\Rightarrow) \text{PCP}(P) \Rightarrow \exists s. l_{s_1} \dots l_{s_l} = r_{s_1} \dots r_{s_k} \Rightarrow l_{s_1} \dots l_{s_l} a_1 \dots a_l = r_{s_1} \dots r_{s_k} a_1 \dots a_k$$

$$\Rightarrow L_G \cap L_H \neq \emptyset \Rightarrow \text{PMA}(f(P))$$

$$(\Leftarrow) \text{PMA}(f(P)) \Rightarrow \text{istnieje jakieś wspólnie slowo} \xrightarrow[\text{z konstrukcją}]{} \underbrace{a_{s_1} \dots a_{s_l}}_w = \underbrace{a_{s_1} \dots a_{s_k}}_v$$

$$\Rightarrow w = v \Rightarrow l_{s_1} \dots l_{s_l} = r_{s_1} \dots r_{s_k} = \text{f co gęps} \xrightarrow[\text{z gęstością}]{} \overline{IwI} = \overline{IvI} \Rightarrow \text{PCP}(P)$$

Czy jest r.c.? TAK

* Generuj po kolei wszystkie permutacje slow $\Sigma = \Sigma_G \cup \Sigma_H$
 co najmniej dwa różnych

* Sprawdź, czy oba slowa należą do $L_G \cap L_H$ (CPLK)

Imię rozw:

$$\Pi = \{(u_1 v_1), \dots, (u_n v_n)\}$$

$$L_1 = \{ w \# w^R \mid |w| \geq 1\}$$

$$L_2 = L \left(\begin{array}{l} S \rightarrow \# \\ S \rightarrow u_i S v_i \end{array} \mid i \leq n \right)$$

$$\exists w \in L_1 \cap L_2 \Rightarrow w = u_{i_1} \dots u_{i_k} \# v_{i_1}^R \dots v_{i_k}^R \wedge (u_{i_1} \dots u_{i_k}) \cdot (v_{i_1}^R \dots v_{i_k}^R)^R \\ = v_{i_1} \dots v_{i_k}$$

$$u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k} \quad \text{nowa dłuż.}$$

$$u_{i_1} \dots u_{i_k} \# v_{i_1}^R \dots v_{i_k}^R$$

↑ swążaj produktywnie oż. skróceniu do #

JF120 zad 115

Nie istnieje Ψ rozszygrywający, aby $L_G = \Sigma^*$

Pokażemy rozkładając $\overline{P114} \leq_{\text{ek}} P115$

$$\forall G, H \in \underline{\text{DCFL}} \quad \overline{P114(G, H)} \Leftrightarrow P115(K)$$

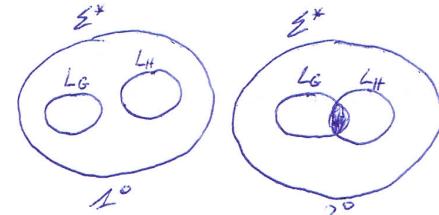
\uparrow
jest grammarka

DCFL - deterministyczne CFL, gramatyki CFL zamknięte m.in. na dopełnienie.
 \rightarrow W szeregu rozkładu dla nich $P114$, bo $\text{DCFL} \subseteq \text{CFL}$

$$L_G \cap L_H = \emptyset \Leftrightarrow L_K = \Sigma^*$$

\uparrow
to jest core., wiemy ze M4.

$$L_K = \overline{(L_G \cap L_H)}$$



$$1^\circ L_G \cap L_H = \emptyset \Rightarrow \overline{(L_G \cap L_H)} = \Sigma^*$$

$$2^\circ L_G \cap L_H \neq \emptyset \Rightarrow \overline{(L_G \cap L_H)} \neq \Sigma^*$$

Czy K jest grammarką bezkontekstową?

$$L_K = \overline{L_G \cap L_H} = \overline{L_G} \cup \overline{L_H}$$

\downarrow
 \downarrow
dopełnienie DCFL sq DCFL czyli teori CFL

suma CFL jest CFL ✓

UNIWERTY: tworzą wartościowe postacie w M4 poleżane, że to zakończenie dla alfa DCFL. Push-down automaton
 \rightarrow ze sposobem \uparrow
 determinator

DETERMINISTYCZNY \rightarrow przykłady są rozporządzane automatem ze M4

Wprowadzenie o gramatyce?

Jednorzędowa

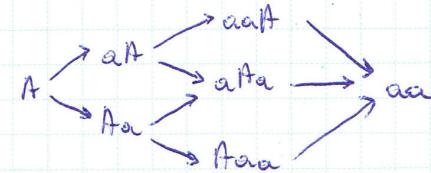
Deterministyczna - istnieje jedno wyprawianowe słowa, czyli wiersz, który produktycznie wygrywa.

$$1^\circ A \rightarrow aA \mid \varepsilon \quad \leftarrow \text{deterministyczna jednorzędowa}$$

$$2^\circ A \rightarrow aA \mid Aa \mid \varepsilon \quad \leftarrow \text{nie deterministyczna jednorzędowa}$$

$$\text{Ad 1}^\circ \quad A \rightarrow aA \rightarrow aaA \rightarrow aa$$

$$\text{Ad 2}^\circ$$



Gramatyki G i H z zad 114 są deterministyczne.

Czymu? Pierwsze symbole a_1 , które jednoznacznie wskazuje na wyłączną produkcję, więc automat ze sposobem to rozpozna



Gdy $L_G = L_H$ jest rek? NIE

$M \leq_{rek} M'$

Dostępnego G i H . Niech H będzie trywialna gramatyka dla alfabetu A , tzn: $S \rightarrow \epsilon \mid aS, a \in A$

$$L_G = A^* \Leftrightarrow L_G = L_H$$

→ trywialne, bo H w任何时候 sprawia generuje A^* .

17 M stop $\leq_{rek} H_w$

Daję nam maszynę Turinga z konkretem. Skonstruujmy $\langle w, \Pi \rangle$

żeby M stop $\Leftrightarrow \{v \mid w \xrightarrow[\Pi]{} v\}$ jest skonczone.

$w =$ konf. maszyny M

Π : * jeśli było $s(a, q) = \langle a', q', L \rangle$ to $aq \rightarrow q'a'$

* jeśli było $s(a, q) = \langle a', q', R \rangle$ to $aqb \rightarrow a'bq'$

// WHTGHT: to, że M jest z konkretem nie ma znaczenia dla konstrukcji:

Π . Rozum się wtedy Q i S maszyny.

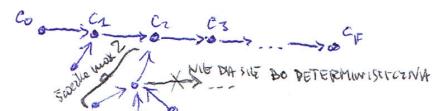
→ my już dostępnego M' , która działa jak M z konkretem kroków.

M nie stop \Rightarrow maszyna się zatrzymała \Rightarrow konkrek robiąc $w \infty$

\Rightarrow mamy co najmniej dwa słowa $\Rightarrow H_w$ jest nieskończony

M stop \Rightarrow istnieje skończony ciąg konfiguracji $c_0 \rightarrow \dots \rightarrow c_F$ wtedy $\{v \mid w \xrightarrow[\Pi]{} v\}$

jest skończony. Ale co z $\xrightarrow[\Pi]^*$ (możemy chodzić w obie strony)?



1° jeśli z jednego c_i zaczynamy "wracać lung swojego" to skoro na konkretem c_i mamy " i ", to ta siatka jest nieskończona nie " i "

2° Wroga pod przed i znowu pojawiły się "z problemem" ale go nie inicjuję
→ tak się wie ale, bo maszyna jest DETERMINISTYCZNA

Gdy $w \{v \mid w \xrightarrow[\Pi]{} v\}$ tei jest skończony. ■

118

occurring possible states \rightarrow clockency clo. changes

$$w \rightarrow v, \quad \alpha x \rightarrow x^b$$

sensitive $\leq_{\text{rel}} 118$

$$\begin{cases} x = w\# & \Sigma^1 = \Sigma \cup \{\#\} \\ y = v\# \\ p = \Pi \cup \{a_i a > |a \in \Sigma^1\} \end{cases}$$

$$x \rightsquigarrow y$$

$$x \longrightarrow w_1\# \longrightarrow w_2\# \longrightarrow \underbrace{w_n\#}_y$$

$$x \rightarrow y \Leftrightarrow \text{sensitive } w \rightarrow v$$

$$w\# \rightarrow w_n\#$$

$$a_1 \dots a_n\# \rightarrow b_1 \dots b_n\#$$

$w\# a_1 a_2 \dots a_n\#$
 $\downarrow \Pi \quad \downarrow \quad \dots \quad \downarrow$
 $w_n\# b_1 b_2 \dots b_n\#$ \rightarrow same energy $\in \Pi$, only
 changing \in sensitive in
 wtw seq due.

119mousz Mousley'spo $\leq_{\text{rel}} \text{MS}$ $\langle q_1, 0, 0 \rangle$ - star polygon $\langle q_0, 0, 0 \rangle$ - star polygon

$$\langle q_i, a, b \rangle \rightarrow \langle q_j, a^{\pm \frac{1}{2}}, b^{\pm \frac{1}{2}} \rangle \quad n = |Q|$$

$$\langle q_i, a, b \rangle \rightarrow 2^i \cdot 3^a \cdot 5^b$$

$$p = 2^n \cdot 3^a \cdot 5^b \equiv r \pmod{p}$$

$$1 \leq r \leq p \quad i < n$$

$$\begin{aligned} \text{GCD}(p, 2^i \cdot 3^a \cdot 5^b) &= 2^i \cdot 3^{\min(a, 1)} \cdot 5^{\min(b, 1)} \\ \text{II} \quad \text{GCD}(p, r) &\rightarrow \text{potrebne jake w alg. Euklidesa} \end{aligned}$$

Skoro:

$$2^0 \cdot 3^5 \cdot 5^7 \rightarrow \underbrace{2^0 \dots 2}_n \quad \text{same konfiguracje ale rekur}$$

$$2^i \cdot 3^6 \cdot 5^3 \rightarrow \underbrace{2^i \dots 2}_n \quad \text{same stary, bo stary, } i^{\text{th}}$$

$$f(m) = \frac{n \cdot ar}{br} \Leftrightarrow \text{Mousley: } \langle q_i, \min(a, 1), \min(b, 1) \rangle$$

119Mousley \leq Conway

and start

$$Q = \{q_0, q_1, \dots, q_{n-1}\}$$

$$\langle q_i, a, b \rangle \rightarrow 2^i 3^a 5^b$$

$$\langle q_1, 0, 0 \rangle \rightarrow 2$$

$$\langle q_0, 0, 0 \rangle \rightarrow 1$$

$$\begin{array}{ll} a_1, \dots, a_m \\ b_1, \dots, b_m \end{array} \quad f(k) = k \cdot \frac{ar}{br} \text{ gdzie } K \equiv r \pmod{m}$$

$$m = 2^n \cdot 3 \cdot 5$$

$$f(k) \not\equiv K \pmod{m}$$

$$\begin{aligned} \text{NWD}(r, m) &= \text{NWD}(2^i \cdot 3^a \cdot 5^b, m) \\ &= 2^i \cdot 3^{\min(a, 1)} \cdot 5^{\min(b, 1)} \quad \begin{array}{l} \text{// cykle w drugim pot. pot.} \\ \text{stan, czy co pot. zera} \\ \text{czy co, co zero} \end{array} \\ &\quad \begin{array}{c} q \\ \text{zero lub 1} \end{array} \quad \begin{array}{c} 5^b \\ \text{zero lub 2} \end{array} \end{aligned}$$

$$\langle q_i, u_1, u_2 \rangle \rightarrow \langle q_j, x_1, x_2 \rangle \quad \begin{array}{l} 2^j \cdot 3^{a+x_1} \cdot 5^{b+x_2} \\ u_1, u_2 \in \{0, 1\} \end{array}$$

$$\text{Czy istnieje } N \text{ t. dt. } f^N(z) = 1 ?$$

wzorzysty punkt.

$$br = 5$$

$$\begin{array}{c} 2^j \cdot 3^{a+x_1} \cdot 5^{b+x_2} \\ x_1 = 1, x_2 = -1 \\ ar = 2^{j+i} \cdot 3 \end{array}$$

120 $M_T \leq_{rek} \text{Kafelkowanie}$

Daje nam $M_T M$, skonstruowany $N \in C$ t.j. $M \text{ stop} \Leftrightarrow \text{istnieje kwadrat}$

Pokazujemy "dobry kafelk" \overline{M} .

$$C = \{G_2, B_2\} \cup \{\langle a^i q^j \rangle \mid a \in \Sigma, q \in Q \cup L, R \}\}$$

jesterzy
na lewo
jesterzy
na prawo od planicy

Idea:	$C_2 \quad B_2 \quad B_2 \quad B_2 \dots \quad B_2 \quad B_2 \dots$
$C_0 \rightarrow$	$A_1 \quad A_2 \quad A_3 \quad A_4 \dots \quad A_n \quad A_{n+1} \quad B \dots$
\vdots	$a_1^i \quad a_2^j \quad a_3^k \quad a_4^l \dots \quad a_n^m \quad B \dots$
$C_F \rightarrow$	$a \dots \quad q_F \dots \quad B \dots$
\vdots	$q_F \quad B' \quad B' \dots \quad B \dots$
	$C_2 \quad B_2 \quad B_2 \dots$

Over

$$\overline{M} = * \frac{C_2 | B_2}{A_2 | A_2}, \frac{B_2 | B_2}{A_1 | A_{1+2}}, \frac{B_2 | B_2}{B}, \frac{A_1 | A_{1+2}}{a^i q^j | a^i R} \text{ itp (perowane razy)}$$

* j.w. $\delta(a^i q^j) = \langle a^i q^j, R \rangle$: (maszyna nie posz)

$$\frac{a^i q^j | b, R}{a^i, L | b, q^j} \text{ oraz } \frac{a, R | b, R}{a^i q^j | b, R} \text{ oraz } \frac{a, b | b, q^j}{a, L | b, L}$$

* aneloponiczny "L", ale nie mówiąca, bo maszyna skanująca ten ~~jest~~ jest strelk.

$$\frac{a, L | b, L}{a, L | b, L} \text{ i } \frac{a, R | b, R}{a, R | b, R} \text{ (maszyna nie nie posz)}$$

* zjadacz: * przesuwa od zjadania do lewego końca npw:

$$\frac{b, L | a, q_F}{b, q_F | NULL} \text{ itp.} \quad \frac{q_F | NULL}{G_2 | B_2}$$

$M \text{ stop} \Rightarrow \text{istnieje Kafelkowanie}$

LATWE \rightarrow wygryska konstrukcji kafelków

| istnieje kafelkowanie $\Rightarrow M \text{ stop}$

Skoro kafelkowanie jest dobrze, to $\overline{M} : L$ to zgodno z konstrukcją kafelków naużymy je wstawiac t.j. dołączając do co i ic prędkowy pierz a^i i zjadacem dodawany do L . Skoro $C_0 \rightarrow C_F$ to maszyna się zatrzymuje.

JF120 zad 121

Paradox: Wyświetlajemy cyfry n , które nie są wykorzystane w mniejszej niż 20 siliach.

My mówimy: Najmniejsze liczbę, której nie da się wykorzystać w mniejszej niż 20 siliach
 $\begin{array}{ccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ + & & & & & & 10 & 11 & 12 \end{array}$

→ Wystarczyłyby 12 sili, napisana liczba.

Komp NIE jest rekurencyjna.

Zat, że jest, wtedy istnieje program Ψ_{komp} , który ją odtwarza (zawiera się rekurencja)

Czyli zwraca $\text{komp}(n)$.

Napiszmy program:

```

 $\Psi$ : for n=0 to  $\infty$ 
    if  $\Psi_{\text{komp}}(n) > \text{len} + 10$ 
        wypisz n
    
```

gdzie len' to ilość instrukcji Ψ_{komp} - kiedy program ma skończoną dлиność. 10 - Ψ ma mniej niż 10 instrukcji.

Ψ wypisze, n' t.ż. Komp twierdzi, że potrafi $> \text{len}+10$ instrukcji
 a Ψ ma len+3 instrukcje - czyli Ψ_{komp} się rozwija! ↴

UNAGA: Komp(n) musi być dość małe, w MUGP wypisujemy liczbę po cyfrach, więc 102356783 to 8 cyfr.

JF120 zad 122

a) $H1O_Z \leq_{\text{rek}} H1O_N$

Mamy: $\left\{ \begin{array}{l} \alpha_1 x_1^{\beta_1} + \alpha_2 x_2^{\beta_2} + \dots + \alpha_n x_n^{\beta_n} = 0 \\ \vdots \end{array} \right.$

rozwi. jest istnieje do $(x_1, x_2, \dots, x_n) \in Z^n$

Konstruujemy:

$$\left\{ \begin{array}{l} \alpha_1 \underbrace{(x_1^1 - x_1^{\prime\prime})}_{\in Z x_1}^{\beta_1} + \alpha_2 \underbrace{(x_2^1 - x_2^{\prime\prime})}_{\in Z x_2}^{\beta_2} + \dots + \alpha_n \underbrace{(x_n^1 - x_n^{\prime\prime})}_{\in Z x_n}^{\beta_n} = 0 \\ \vdots \end{array} \right.$$

rozwi. jest istnieje $\rightarrow (x_1^1, x_2^1, x_3^1, \dots, x_n^1, x_n^{\prime\prime}) \in N^{2n}$

→ każda liczba całkowita da się przedstawić jako rozw. naturalny.

b) $H1O_N \leq_{\text{rek}} H1O_Z$

Wskazówka: $\forall n \in N \exists a, b, c, d \in Z \quad n = a^2 + b^2 + c^2 + d^2$

Mamy: $\left\{ \begin{array}{l} \alpha_1 x_1^{\beta_1} + \alpha_2 x_2^{\beta_2} + \dots + \alpha_n x_n^{\beta_n} = 0 \\ \vdots \end{array} \right.$

rozwi. $(x_1, x_2, \dots, x_n) \in N^n$

Konstruujemy:

$$\left\{ \begin{array}{l} \alpha_1 \underbrace{(A_1^2 + B_1^2 + C_1^2 + D_1^2)}_{= x_1}^{\beta_1} + \dots + \alpha_n \underbrace{(A_n^2 + B_n^2 + C_n^2 + D_n^2)}_{= x_n}^{\beta_n} = 0 \\ \vdots \end{array} \right.$$

rozwi. $(A_1, B_1, C_1, D_1, \dots, A_n, B_n, C_n, D_n) \in Z^{4n}$

x^n - st. wielomianu: n

$$x^n \cdot y^m - \text{st weitemann} = n+m$$

H10Z ≤ H10 bis

Kaide równanie to wobudzenie st. max 2.

Dostosujesz układy równań, zawierające równania które mają st > 2 na kilka równoważnych st ≤ 2.

just many values x^n to randomly use:

$$\left\{ \begin{array}{l} x_1 = x \\ x_2 = x \circ x_1 \\ x_3 = x \circ x_2 \\ \vdots \\ x_n = x \circ x_{n-1} \end{array} \right.$$

diese ~~abfolge~~ ~~abfolge~~ zunehmend ist 1
eigentlich ich denke jetzt ist 2.

just many polynomials $x^n y^m$ to remember me:

$$\left\{ \begin{array}{l} x_n = x \circ x_{n-1} \\ \vdots \\ x_1 = x \\ y_m = y \circ y_{m-1} \\ \vdots \\ y_2 = y \end{array} \right.$$

$$\text{Gal} 124 \quad H_1O_Z \leq_{\text{rk}} H_1O_{Z_{2k+1}}$$

Many woolousian w , stopper $\deg(w) = d$

$W1 = 2^d \cdot W$, wtedy dla każdego występującego x_i mamy:

$$2^d \cdot x_i = 2^{d-\beta_i} \cdot \underbrace{(2x_i)}_{\beta_i}$$

→ ozyw. W1 nie tylko rozwijającą pełnię

w_2 = podstawa wej. $e_i - 1$ od w_1

$$\rightarrow \text{cycle} \quad \underbrace{2x_i^*}_{\text{odd}} = z_i^* = \underbrace{c_i^* - 1}_{\text{even}}$$

↳ we have $\text{val } \leftarrow \text{ we have now independent.}$

Inne resurssenser:

Zentrale H_2O -Zentren \rightarrow isturige organische Verbindungen mit hydrophilen Gruppen
 \hookrightarrow Werbeschmuckstoffe.

Napíšeme propozici, kterou využívá HCl_3 :

fólk 4: sprawdzić czy istnieje konstrukcja $w_1^2 = 10, w_2 \oplus 13, \dots, w_{10} \oplus 9$
 $E(x_1 + w_1, x_2 + w_2, \dots, x_n + w_n)$

$\rightarrow \Sigma^n$ posst, alle jette & wege geliebte Wörter wieder (jedes ist möglich)

Cough weakly H_2O_2 post resp \rightarrow sputocrust.

Wad 126 PCP $\leq_{\text{rel}} \text{EqStack}$, NIE istwuje takie udowodnienie.

Dajżej nam $\langle l_1, r_1, \dots, l_k, r_k \rangle$ zbadujemy dla deterministycznego PDA, t.j. PCP ma rozszerzenie $\Delta_2(w) = D_2(w)$

↑ istwy teore same po przyjęciu

$$Q = \{q\} - 1 \text{ stan}$$

$$\Sigma = \{s_1, s_2, \dots, s_k\} - \text{współczesne indeksy}$$

$$\Gamma = \Sigma \cup A$$

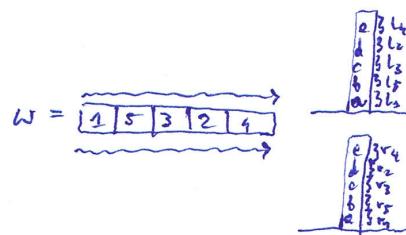
↑ alfabet stow $l_1, r_1, \dots, l_k, r_k$
symbol dane stow

$$S_2: \langle q, s_i, w \rangle \rightarrow \langle q, l_{s_i} \rangle$$

↑ na stow indeks (indeks w)

na stow dokladnie ktorego kowetka
o odpowiednim indeksie

$$\Delta_2: \langle q, s_i, w \rangle \rightarrow \langle q, \lambda, s_i \rangle$$



$$P \in \text{PCP} \Leftrightarrow f(P) = \langle D_2, \Delta_2 \rangle \in \text{EqStack}$$

(\Rightarrow) $P \in \text{PCP} \Rightarrow$ istwiste współczesne indeksy $s_1 \dots s_k \Rightarrow$ jest istny deterministyczny PDA $w = s_1 s_2 \dots s_k$ taki ze przekształcaniu go po przyjętym bieżącym indeksie wnoszą istwy

(\Leftarrow) istwuje istwy stow \Rightarrow one losujac rozszerzenie PCP \Rightarrow PCP ma rozszerzenie.

Wad 127 PCP $\leq_{\text{rel}} \langle G, S \xrightarrow{\pi} \epsilon \rangle$

Dajżej nam stow par $l_1, r_1, \dots, l_k, r_k$. Skonstruujemy gramatykę \in kontekstowej zadaną, t.e. $\exists S \quad l_{s_1} \dots l_{s_k} = r_{s_1} \dots r_{s_k} \Leftrightarrow S \xrightarrow{\pi} \epsilon$

$$\Pi: S \xrightarrow{\text{def.}} \underbrace{L_1 S^1 R^R_1 | \dots | L_k S^1 R^R_k}_{\text{zakladejemy}} // \text{aby nie bylo slasku juz tego}$$

$$\cancel{S^1 \rightarrow L_1 S^1 R^R_1 | \dots | L_k S^1 R^R_k}$$

$$L_1 = \underbrace{A_1^L \dots A_5^L}_{\text{takie wskazujace}}$$

$$R^R_1 = \underbrace{A_T^R \dots A_1^R}_{\text{takie wskazujace}}$$

L i R to znakow, my poszukujemy
 $= l_i \text{ i } r_j = r_i$

$$\left. \begin{array}{l} \text{zakladejemy: } \\ \text{czyli: } A_i^L A_i^R \rightarrow \epsilon \end{array} \right\} (\text{czyli } \epsilon \text{ to mamy "przygotowanego"})$$

$$T = \{ \epsilon \}$$

$\epsilon = \text{bez ... liter}$

$$N = \{ S, S^1, A_i^L \text{ dla kredago } a_i \in \Sigma, A_i^R \text{ dla kredago } a_i \in \Sigma \}$$

(\Rightarrow) PCP ma rozszerzenie \Rightarrow istnieje algorytm indeksow t.j. $l_{s_1} \dots l_{s_k} = r_{s_1} \dots r_{s_k}$

\Rightarrow wygenerowujemy stow $w = l_{s_1} l_{s_2} \dots l_{s_k} R_{s_1}^R \dots R_{s_k}^R$ i to pełnowartowne

\Rightarrow budujemy moźliwosc po wygenerowaniu dla ϵ , czyli $S \xrightarrow{\pi} \epsilon$ ✓

(\Leftarrow) $S \xrightarrow{\pi} \epsilon$ czyli pozwolilo coś, co mogło to być w pełnowartownym pełnowartownem

\Rightarrow wygenerowaliśmy po takim produktyjnym, t.e. istnieje $l_{s_1} \dots l_{s_k} = r_{s_1} \dots r_{s_k}$
czyli istwuje algorytm indeksow.

Many semiTorus $\langle \Pi, w, v \rangle$ $\forall \langle w, v \rangle \in \Pi$ $|w| = 1$

Rozstrzygalny ✓

Die dagegs problem P: $\langle T, w, v \rangle$ konstruerej gneus by kg CFG

$$G = \langle T, N, S, \Pi^l \rangle$$

$N = \{S\} \cup \Sigma$ // skoro $\text{pot} = 1$, to kolejne litery muszą być połączone z jednym z liter alfabetu
 \leftarrow deterministyczny

$$T = \{a^i : a \in \Sigma\}$$

$$S = \omega = \omega_1 \omega_2 \dots \omega_n$$

$$\Pi^1 = \Pi \cup \{ s \rightarrow w \} \cup \{ a \rightarrow a' : a \in \Sigma \}$$

→
the major big konstrukteure
 $\alpha \rightarrow \varepsilon$

→ listek z częstotlą się jesi terminalna

→ Sprowadzamy do postaci Chomskiego

→ Algoritmu CTK alpawde on de soj puerobit $w \xrightarrow{*} v$

only one $v \in L_G$, $v = v_1^l v_2^l v_3^l \dots v_n^l$
 \uparrow
 prime to terminal

JF120 zad 12.3

semiThue $|w| \cdot |v| = 2$

NIE ROZSTRZYGALNY

\mathcal{L}_{rk} ALMOST THUE

Dowód: dowodzące że nie rozstrzygawne dla semiThue

Mamy maszynę Turinga M_n , skonstruującą \mathbb{P} (almost Thue) t.j.:

$M_n(n)$ się redukuje $\Leftrightarrow \omega \xrightarrow{*} v$ gdzie $\mathbb{P}(\omega, v, \Pi)$

$\mathbb{P} = \Sigma = Q \cup \{\alpha, 0, 1, \omega, B\}$

Π : * jeśli w s. jest instrukcja $\langle q_i a; q'_j a', L \rangle$ to

$w \Pi$ będzie produktye: $\overbrace{aq} \rightarrow \overbrace{q'a'}$ // cykl: $\overbrace{a} \xrightarrow{q} \overbrace{a'}$

zauważ tyleż produktye: $\overbrace{2-2=4}$

1° $\overbrace{aq} \rightarrow \overbrace{A_{aq}}$

2° $\overbrace{A_{aq}} \rightarrow \overbrace{q'a'}$ A - rozwinięta zredukowana alfabetu, aby nie mówić konwersji: $00 \rightarrow \overbrace{00}$

blotek

* jeśli w s. jest $\langle q_i a; q'_j a', R \rangle$ to $w \Pi$:

$\overbrace{ab} \rightarrow \overbrace{ab}$ $\overbrace{ab} \rightarrow \overbrace{a'b'}$ $\overbrace{ab} \rightarrow \overbrace{a'b}$ $\overbrace{ab} \rightarrow \overbrace{a'b'}$
3-3=9
 $\overbrace{ab} \rightarrow \overbrace{ab}$ $\overbrace{ab} \rightarrow \overbrace{a'b'}$ $\overbrace{ab} \rightarrow \overbrace{a'b}$ $\overbrace{ab} \rightarrow \overbrace{a'b'}$

xanidz tyleż produktye

1° $\overbrace{aq} \rightarrow \overbrace{B_{aq}}$
2° $\overbrace{B_{aq} b} \rightarrow \overbrace{C_{aqb}}$
3° $\overbrace{C_{aqb}} \rightarrow \overbrace{D_{ab} q'}$
4° $\overbrace{D_{ab}} \rightarrow \overbrace{a'b}$

blotek

blotek

blotek

blotek

* $B \rightarrow BB$
zadanie {
* $q_F a \rightarrow q_F$
* $a q_F \rightarrow q_F$

(konfiguracja po rozpoczęcie uruch.)

weflate: $w = \alpha q_0$ w konstrukcji, $w B$ ---
 $v = q_F$

$w \xrightarrow{*} v$ - wynika z konstrukcji, jeśli pośrodku są pojawiać się q_F to oznacza "zakończony".

JF120 zad 130

fajny Thue $\rightarrow \langle l, r \rangle \in \Pi \Rightarrow |M_l| = |M_r|$

P: $1111 \xrightarrow{*}_{\Pi} w$ urozszypalny

$K \leq_{rek} 2CMM \leq_{rek} \varPhi$
two counters Thue's machine

Pokazanie, że dla instancji problemu P, φ $111 \xrightarrow{*}_{\Pi} w \Leftrightarrow \psi_n(n)$ są rozwiązywane

INTERPRETACJA:

↑ masyng
dla każdego z dwóch
kombinacji

1,00...00,1
00...00,1
liczba A
uwartość
liczba B
uwartość

Π : $\rightarrow 0 = \text{party male}$ $\rightarrow 0 = \text{party female}$

- * $01 \xrightarrow{q} 10 \rightarrow 001 \xrightarrow{q} 10 // \text{inc } A$
- * $001 \xrightarrow{q} 10 \rightarrow 01 \xrightarrow{q} 10 // \text{dec } A$
- * $01 \xrightarrow{q} 10 \rightarrow 01 \xrightarrow{q} 100 // \text{inc } B$
- * $01 \xrightarrow{q} 100 \rightarrow 01 \xrightarrow{q} 10 // \text{dec } B$

} $\forall q \in Q$

\rightarrow faktyk interpretująca masyng endowac wartości kombinacji
aby manipulować $2^a 3^b 5^c 7^d$.

- * ~~$01 \xrightarrow{q} 10 \rightarrow 01 \xrightarrow{q'} 10 // \text{zwiększenie stanu}$~~ } $\forall q, q'$
 \uparrow
 \uparrow
biorekli masyng by paste

$w = 101 \xrightarrow{q_F} 101$
 \uparrow
puste konsoli

131 y-koduje ciąg liczb pierwszych

$$\psi \exists x, y \quad y \geq 1 \quad (\exists z) \\ \wedge (\exists p \text{ MinPierw}(p, y) \wedge a_1 = 1) \\ \wedge \text{Mod}(x, p, 1)$$

$$\wedge (\exists p \text{ MaxPierw}(p, y) \wedge a_2 = 2) \\ \wedge \text{Mod}(x, p, 2)$$

$$\wedge \left(\forall p_1' \in \{p_1\} \left(\text{Kolejne Pierwsze}(p_1, p_1', y) \wedge \text{Mod}(x, p_1, a) \right) \wedge \right. \\ \left. \text{Mod}(x, p_1', a') \right) \Rightarrow \varphi(a, a')$$

Kolejne liczby pierwsze
 w tym węzeł
 a do so, elementy
 co gęste

x - związek się w zakodowanych ciągach

$$\text{Kolejne Pierwsze}(x, y, z) = \text{Pierwsze}(x, z) \wedge \text{Pierwsze}(y, z) \\ \wedge \neg \exists k (k < y \wedge k > x \wedge \text{Pierwsze}(k, z))$$

$$\text{Pierwsze}(x, y) = \underbrace{\text{Pierwsze}(x)}_{\text{zwykła definicja}} \wedge \text{Dobr}(x, y)$$

~~x - rozw. kolejnych liczb pierwszych~~

y - kodowane liczby pierwszych

\rightarrow y skryta lista liczb pierwszych

aby je złożyć opatrzyć

$\wedge p \notin \text{MinPierw}$
 $\wedge p \notin \text{Max Pierwsze}$

\wedge Chodzi o resztach

Chodzące Tw o resztach

$$m \equiv a_1 \pmod{m_1}$$

$$m \equiv a_2 \pmod{m_2}$$

gdzie m_1, m_2 - m to względne pierwiastki

Wtedy \exists jakaś rozmaitość tworząca ułamek.

$$0 \leq m \leq m_1 \dots m_2$$

Znaczący znak jest przy powyższym tworzącym ułamekiem.

Makro:

$$1) x/y = \exists k \cdot y = k \cdot x$$

$$2) x \equiv z \pmod{y} \equiv y \mid z \wedge \exists k \cdot x = k \cdot z + y$$

$$3) \text{Pierwsze}(x) \# \text{ - względny}$$

$$4) \text{Kolejne Pierwsze}(x, y) \text{ - względny}$$

$$5) \text{Najmniejsze Pierwsze}(n, p) \equiv \text{Pierwsze}(p) \wedge p \mid n \wedge \forall q, \text{Pierwsze}(q) \wedge q \mid n \Rightarrow q = p$$

$$6) \text{Największe Pierwsze}(n, p) \text{ - analogicznie}$$

$$7) \text{Dobre}(n) \equiv \exists p, q \text{ Najmniejsze Pierwsze}(p, q) \wedge \text{Największe Pierwsze}(n, p)$$

$$\wedge \forall p' (\text{Pierwsze}(p') \wedge p \leq p' \leq q) \Rightarrow p' \mid n$$

// skryty kolejnych liczb pierwszych od p do q
 // określone indeksy się powtarzają.

$$\varphi: \exists \overbrace{a_0, a_1, \dots, a_l}^{\text{to pasty wszystkie}}, \overbrace{z_{n,m}}^{\text{także z kótką}}, \# a_0=1, a_1=2 \wedge \forall i \ 1 \leq i \leq l-1 \ \#(a_i, a_{i+1})$$

$$\varphi: \exists n, m \text{ Dobre}(n) \wedge \exists p, q \text{ Najmniejsze Pierwsze}(p, q) \wedge \text{Największe Pierwsze}(n, q) \wedge m \equiv 1 \pmod{p} \\ \wedge m \equiv q \pmod{p} \wedge \forall p' \text{ Pierwsze}(p') \wedge \forall q' \text{ Pierwsze}(q') \wedge p \leq p' \leq q \wedge q \leq q' \\ \wedge \exists r_1, r_2 \text{ m} \equiv r_1 \pmod{p} \wedge m \equiv r_2 \pmod{q} \Rightarrow \#(r_1, r_2)$$

131

$$\text{Zad 134} \quad [L_G \cap L_H \neq \emptyset] \Leftrightarrow [L_{G'} \cap L_{G'} \cap L_H \neq \emptyset]$$

Dlaż nam jest $G \in H$, skonstruujmy G' tzn

$$L_G \cap L_H \neq \emptyset \Leftrightarrow L_{G'} \cap L_{G'} \cap L_H \neq \emptyset$$

$$G' : \begin{cases} T_{G'} = T_G \cup T_H \\ N_{G'} = N_G \cup N_H \cup \{\#\$, \$\} \\ S_{G'} \\ \Pi : S_{G'} \rightarrow \# | \# S_G \$ | \# \# S_H \$ \end{cases}$$

$$\begin{array}{ccc} L_{G'} & L_G L_{G'} \\ \downarrow & \downarrow \downarrow \\ S_{G'} & S_G' S_{G'} \\ \downarrow & \downarrow \downarrow \\ \# \# S_H \$ & \# \# S_G \$ \end{array}$$

(\Rightarrow) $L_G \cap L_H$ niepuste oznacza istnienie wspólnego słowa „ w ”.

Dla wyciągnąć wspólnego słowa „ w' ” = $\# \# w \$$

(\Leftarrow) $L_{G'} \cap L_{G'} \cap L_H$ niepuste. Zauważmy, że każde słowo z $L_{G'}$ ma 1# lub 2# na początku. Zbyt wspólnego słowa się zgodziło we prefiksie musieli być generowane

$$L_G L_{G'} : S_{G'} S_{G'} \rightarrow \# S_{G'} \rightarrow \# \# S_G \$$$

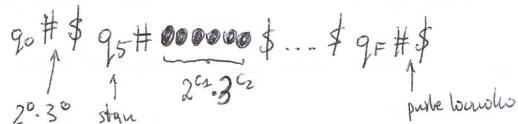
$$L_{G'} : S_{G'} \rightarrow \# \# S_H \$$$

Słowo jest jedno - wspólnie ~~#~~ słowa, to aby się go wygenerować musi być wyciągnięte S_H i S_G , zatem $L_G \cap L_H$ dla samego wspólnego słowa (w').

$\overline{MM} \leq_{rel} TOTAL_{NFA \text{ 2 states}}$

Daje main mamyg^z Minsky'ego, skonstruujemy automat = dwanya stepenem
 t. j. ΣM soj uznaczalnym \Leftrightarrow Automat akceptuje wszystkie słowa nad Σ ,

Staw jest TAKIE jest kochaję przede wszystkim Murskym -



Cryk spodnie garnki:

- 1) jest postaw $(Q \# \otimes^* \$)^*$, gdzie $Q = (q_0 + q_1 + \dots + q_{n-1})$
 - 2) Każda kolejna konfiguracja jest zgodne z S masyw.

$$\text{Cayley's rule: } s(q_i, N2, N2) = s(q_{i+1}, +1, +1)$$

$$q_i \# \underbrace{\dots}_{2^{c_2} 3^{c_2}} \$ q_j \# \underbrace{\dots}_{2^{c_2+1} 3^{c_2+1}} \$$$

Automat akceptuje wszystkie słowa, jeli w wypisie ze BĘDĄCIE.

→ często nie zauważa których z warunków.

~~Automet~~ Automet zapada powód brydłkowym stanem

- warunki "1" mniej sprecyfikowane, bo mniej mocy DFT
 - warunki "2": jeśli mniej \rightarrow zgodnie, które dwie są jednocześnie konfiguracjami nieprawidłowymi (niezgodne z §) odnoszące się i co i co

Pokazany do trzech na przykładzie:

jestwo bytu w maszynie: $\delta(q_i, z, NZ) = \langle q_j, +1, +1 \rangle$

$$q_i \# \underbrace{00 \dots 0}_{c^x} \$ q_j \# \underbrace{00 \dots 0}_{c^{x+1}} \$$$

- automat ustalony się na poziomie conf_x , gdzie uchwadzają się $\text{conf}_x \rightarrow \text{conf}_{x+1}$
 - odchylyje stan, będzie był daleki biermele jest niepoprawne.
 - C_1 ma być zero \rightarrow dla siego stanu, biermele ma być niepodzielne przez 2
 - C_2 ma być parzyste \rightarrow dla siego stanu, — 11 — podzielne przez 3
 - Skoro biermele w conf_{x+1} ma być 5 razy większy, to
 - automat będzie po każdym kroku robić kolejno 5 ε -przejścia
 - czyli na jeden kamyk zrobimy robić 0 5.
 - = po skończeniu zadania zróbi 1 stoper
 - prawdopodobnie do conf_{x+1} , ponieważ stan i odpowiada drugiemu stoperowi
 - teraz po każdym kroku będzie posiadać normalne, tzn na jeden kamyk zrobimy jeden krok.
 - jeśli dojdziemy do końca to zadajemy stoper 2.
 - jesteśmy stopery NIE są równie jak jest BRZYDKO czyli całkowitym sposobem.

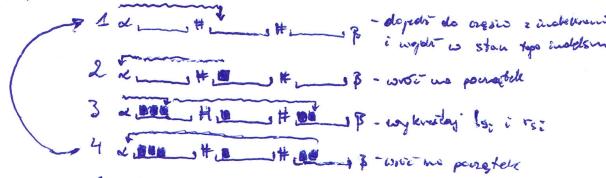
Zad 136 $\text{PCP} \leq_{\text{rel}} \text{ZK}$ niepop

Dlaż nam instancja PCP: $l_1 \dots l_k r_1 \dots r_k$. Zbudujmy taki automat ZK, że istnieje jakieś słowo w ZK akceptujące $\Leftrightarrow \exists s \quad l_{s_1} \dots l_{s_l} = r_{s_1}^R \dots r_{s_k}^R$

Intuicja: * słowo musi być rowne PCP.

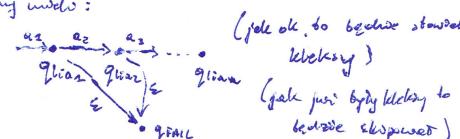
$$M = \langle l_1 l_2 \dots l_k \# s_1 s_2 \dots s_l \# r_1 r_2 \dots r_k \rangle$$

DFA ZK:



Wróć na samą parętek, poszukaj słowa i popchnij do stanu gr: tylko gęstość sz sama kleksy.

Stan gr → mamy k par słów $l_i r_i$, gdzie k stanów średnich. Kolejne słwo w każdej parze jest skonczone wstępnie przez stanów budujących mówiąc:



- (\Rightarrow) jeśli słowo nowe jest w ZK zaakceptowane to słowo.
- (\Leftarrow) jeśli słowo zaakceptowane, to poświęcić wszystko, co będzie jednak istnieć jakaś rozwijająca PCP.

137 $\overline{\text{PCP}} \leq_{\text{rel}} \text{TOTAL}(\text{ZK})$

Dlaż nam instancja PCP $l_1 \dots l_k r_1 \dots r_k$. Skonstruujmy ZK t.ż. akceptując wszystkie słowa \Leftrightarrow PCP nie ma rozwijania.

Słowo jest ~~ŁADNE~~ jest kandyduje rozwijanemu PCP:

$$l_{s_1} l_{s_2} \dots l_{s_l} \# r_{s_1}^R r_{s_2}^R \dots r_{s_k}^R$$

- 1) ma dokładnie jeden symbol #
- 2) zaczyna się symbolem „indeksowym” (i kończy również)
- 3) przed #: po każdym indeksie i_k jest odpowiadające słowo l_k . po #: przed każdym indeksem i_k jest odpowiadające słowo r_k^R .
- 4) patrząc na same indeksy mamy palindrom
- 5) patrząc na same litery $\geq \Sigma$ mamy palindrom.

ŁADNE $\Leftrightarrow 1 \wedge 2 \wedge 3 \wedge 4 \wedge 5$

$\text{BRZDZIKI} \Leftrightarrow (\neg 1) \vee (\neg 2) \vee (\neg 3) \vee (\neg 4) \vee (\neg 5)$ // mówiąc sobie rozwijanie

Wszystkie BRZDZIKI $\equiv \text{TOTAL}$, kiedy istnieje przedeleg akceptujący. (pokazujący ie słowo w' jest brygadka).

ZK jest ŁNFA i mówiąc rozwijać przedrost. Wybiera niedeterministyczne przedrost, dla którego to słowo jest BRZDZIKI. $\neg 1, \neg 2, \neg 3$ potrafi sprawdzić, bo ta mówiąc rozwijająca DFA. $\neg 4$ i $\neg 5$ to sprawdzają co to mówiąc palindrom. Gdyby mówiąc słwo, to by mówiąc. Ale mówiąc rozwijając przedrost α mówiąc co to mówiąc. Te dwa sprawdzają co to mówiąc palindrom co mówiąc, co mówiąc $\neg 4 \wedge \neg 5$.

Katem ZK mówiąc rozwijając BRZDZIKI, totalność alle mówiąc jest rozwijalna.