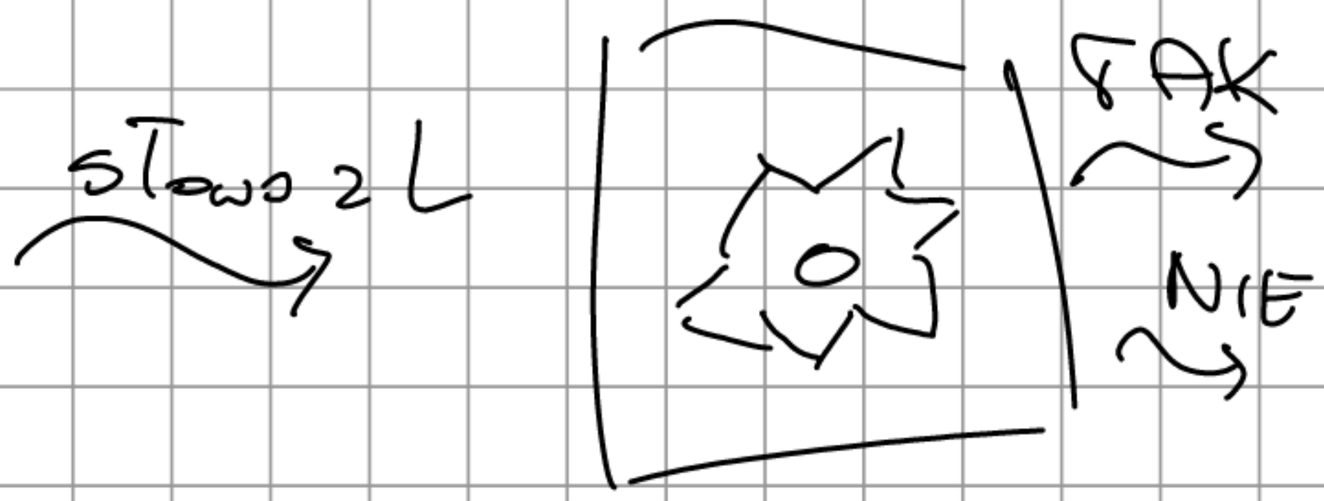
28.08.2022

PROBLEM DECYZYJNY

- Σ – skończony alfabet
- Σ^* – zbiór skończonych słów nad alfabetem Σ
- $\Sigma^* \supseteq L$ – język / problem
- Pytamy o "zasoby obliczeniowe" potrzebne do rozstrzygania tych problemów.

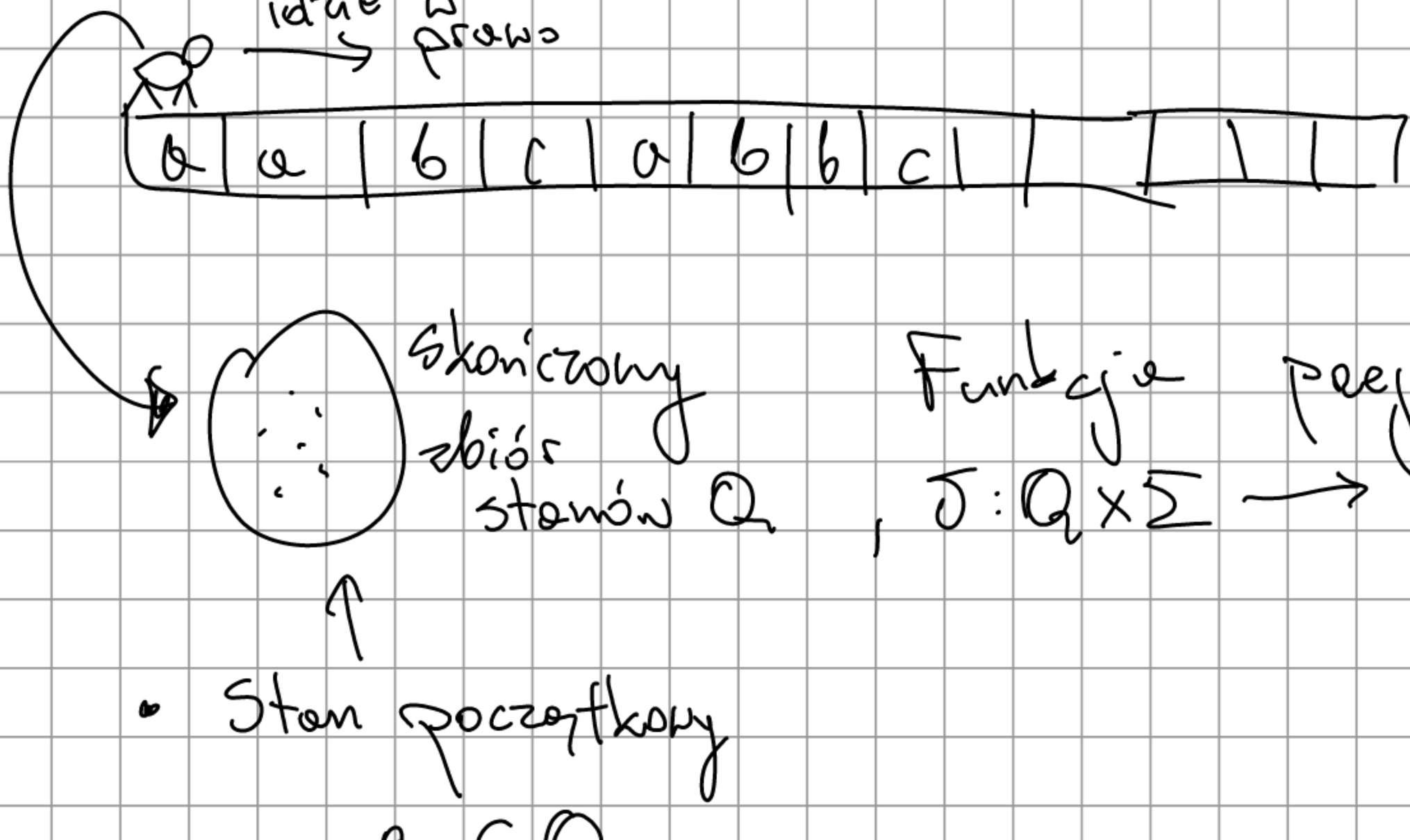


- Klasyfikujemy problemy ze względu na te zasoby

Bla bla...

CZĘŚĆ I

Automaty skończone



Funkcja przejścia $\delta : Q \times \Sigma \to Q$

$q_0 \in Q$

- zbiór stanów akceptujących $F \subseteq Q$

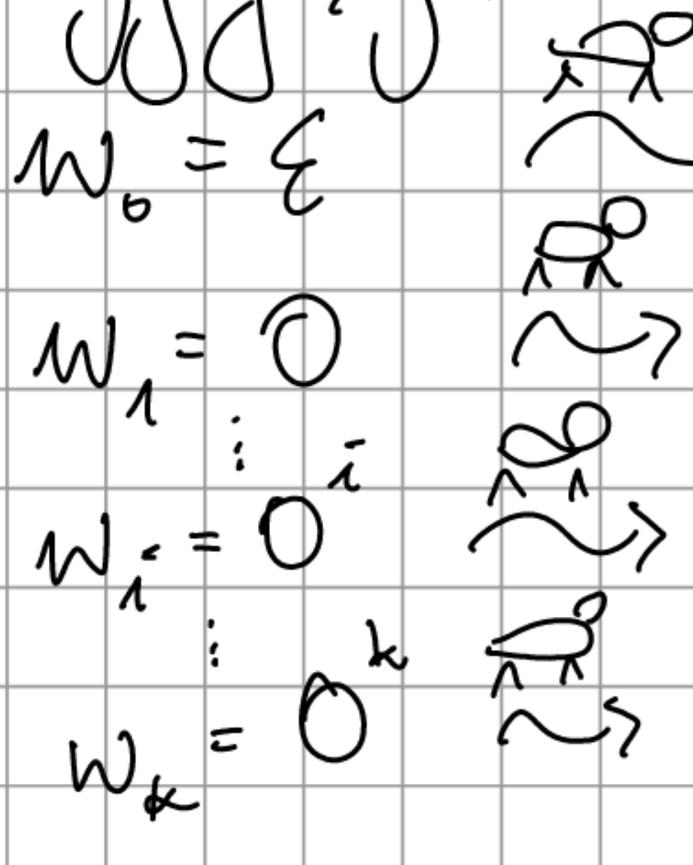
Ćw. Skonstruuj δ, Q dla $\Sigma = \{0,1\}$,

$L = \{ w \in \{0,1\}^* : |w|_1$ jest parzystą$\}$

Ale dla $L = \{ w \in \{0,1\}^* : |w|_1 = |w|_0 \}$

się nie da!

D-d. (A.a.) Niech Zenon będzie zwierzęciem rozstrzygającym L. Zał. że $|Q| = k$.



Jest i, j t. że $s_i = s_j$

Spójrzmy na

$a = w_i 1^i \rightsquigarrow s \in F$

$b = w_j 1^i \rightsquigarrow s \notin F$ ↯

Deterministyczny automat skończony (DFA) to krotka $(\Sigma, Q, q_0, \delta, F)$.

Mamy $\delta: Q \times \Sigma \to Q$, definiujemy

$\hat{\delta}: Q \times \Sigma^* \to Q$:

$$\begin{cases} \hat{\delta}(q, \varepsilon) = q \\ \hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a) \end{cases}$$

Równoważnie: $\hat{\delta}(q, aw) \overset{\text{ZUB}}{=} \hat{\delta}(\delta(q, a), w)$

Dla DFA $A = \langle \Sigma, Q, q_0, \delta, F \rangle$ przez L_A oznaczamy $\{w \in \Sigma^* : \hat{\delta}(q_0, w) \in F\}$.

Def. $L \subseteq \Sigma^*$ nazywamy regularnym gdy istnieje DFA A t.że $L = L_A$.

Lemat (o pompowaniu dla języków regularnych)

Dla każdego j. reg. L istnieje $n \in \mathbb{N}$ t.że dla każdego $w \in L$ t.że $|w| \geq n$ istnieją słowa x, y, z t.że $xyz = w$

oraz $y \neq \varepsilon$, $|xy| \leq n$ takie że dla każdego $k \in \mathbb{N}$ $xy^k z \in L$.

Przykład Weźmy $L = \{w \in \{0,1\}^* : |w|_0 = |w|_1\}$.
Zał. że L regularny. Weźmy n jak z lematu. Niech $w = 0^n 1^n \in L$. Weźmy x, y, z jak w lemacie. Ale $|xy| \leq n$.
więc $|y|_1 = 0$. Dla $k=0$: $xz \in L$
(nie ma "1" w y) ale $|xz|_0 < |xz|_1$ ↯

Dowód lematu

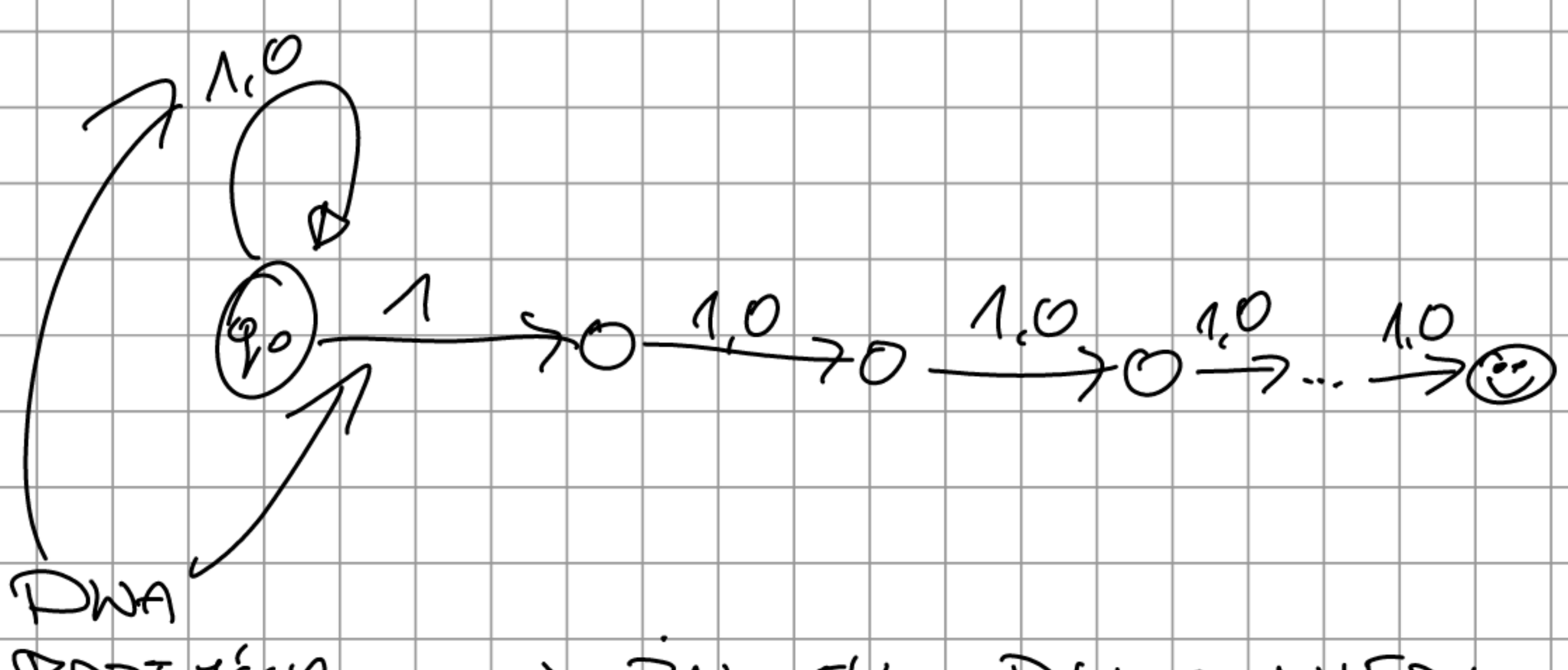
Weźmy $L = L_A$ regularny $(A = \langle \Sigma, Q, q_0, \delta, F \rangle)$.
Niech $n = |Q| + 1$. Weźmy $w \in L$ t.że $|w| \geqslant n$.
Niech $w = a_1 a_2 \dots a_l$, niech $s_i = \hat{\delta}(q_0, a_1 \dots a_i)$.
Wtedy $s_i = s_j$ dla pewnych $i < j \leq n$.
Niech $x = a_1 \dots a_i$, $y = a_{i+1} \dots a_j$,
$z = a_{j+1} \dots a_l$. Wtedy dla $k \in \mathbb{N}$ ... ∎

1.03.2022

NIEDETERMINISTYCZNE AUTOMATY SKOŃCZONE (NFA)

Przykład

$L = \{ w1v \in \{0,1\}^* : |w| = 9 \}$



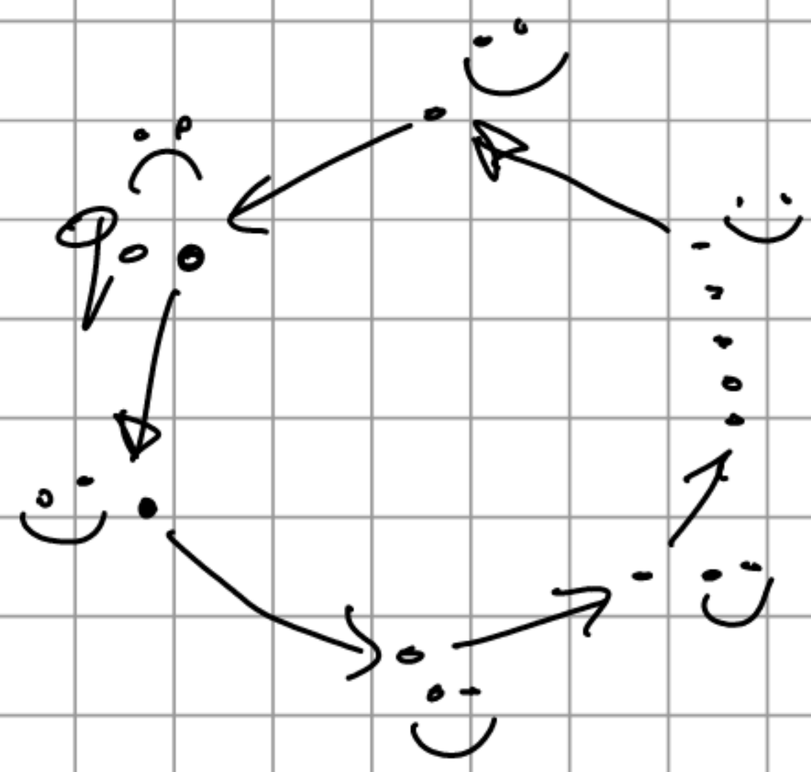
PRZEJŚCIA Z Λ → ŻUCZEK PYTA NIEBIOS "CO ROBIĆ? JAK ŻYĆ"

A NIEBIOSA ODPOWIADAJĄ

Taki automat daje gwarancję, że na pewno nie trafimy w stan akceptujący, jeśli słowo nie jest z języka (nie ma "false-positive" są "false-negative")

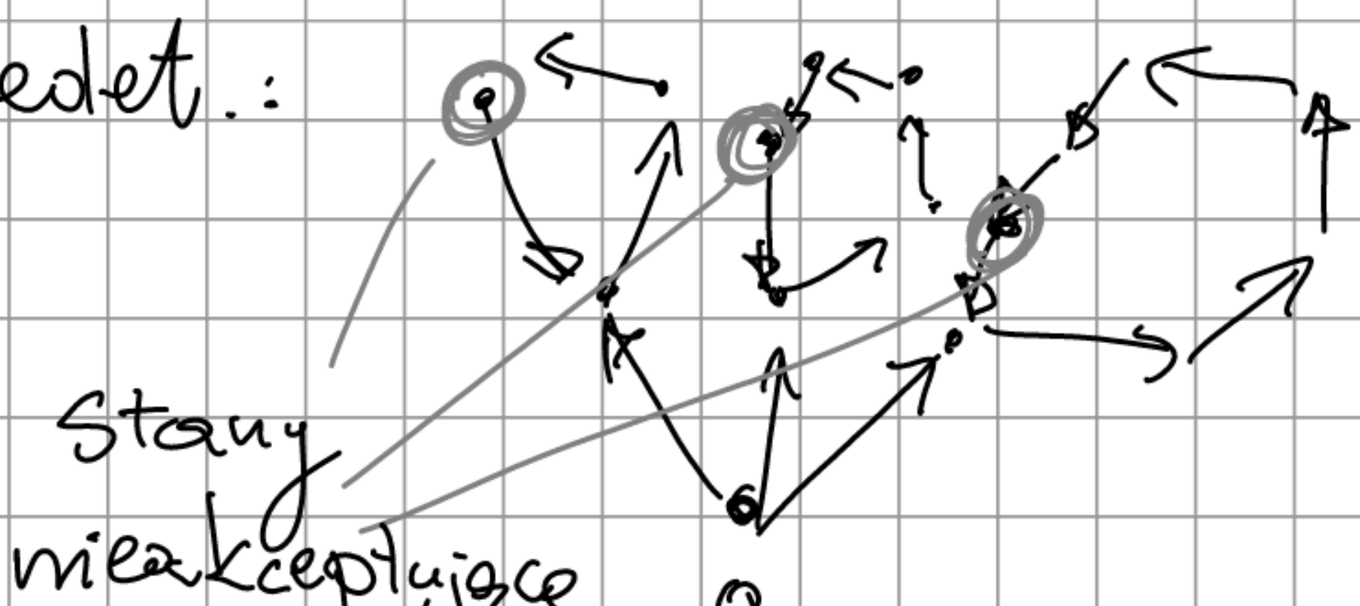
Przykład 2 $L = \{0^i : 105 \mid i\}$

Det. aut.:



105 stanów

Niedet.:



Znowu: na pewno jeśli $105 \mid i$,

to trafimy w stan nieakcept.

Def. Niedeterministyczny automat skończony to krotka $\langle \Sigma, Q, q_0, \delta, F \rangle$ jak w DFA poza δ, gdzie

$$\delta \subseteq Q \times \Sigma \times Q.$$

$\delta(q, a, q')$ – z q do q' jest strzałka z etykietą a.

Teraz $\hat{\delta} \subseteq Q \times \Sigma^* \times Q$:

$$\begin{cases} \hat{\delta}(q, \varepsilon, q') \iff q = q' \\ \hat{\delta}(q, wa, q') \iff \exists p \in Q \ \ \hat{\delta}(q, w, p) \\ \qquad\qquad \wedge \delta(p, a, q') \end{cases}.$$

Alternatywna wersja wg JMa.

Dla każdego $w \in \Sigma^*$ definiujemy

$\delta_w \subseteq Q \times Q$:

$$\begin{cases} \delta_\varepsilon = id_Q \\ \text{jeśli } a \in \Sigma \text{ to } \delta_a(q, q') \iff \delta(q, a, q') \\ \delta_{wa} = \delta_w \circ \delta_a \end{cases}$$

Wtedy $\hat{\delta}(q, w, q') \Leftrightarrow \delta_w(q, q')$.

Def. A: NFA. Wtedy

$$L_A = \{ w \in \Sigma^* : \exists q \in F \; \hat{\delta}(q_0, w, q) \}$$

Tw. Niech A: NFA. Wtedy $\exists A'$: DFA t.że $L_A = L_{A'}$.

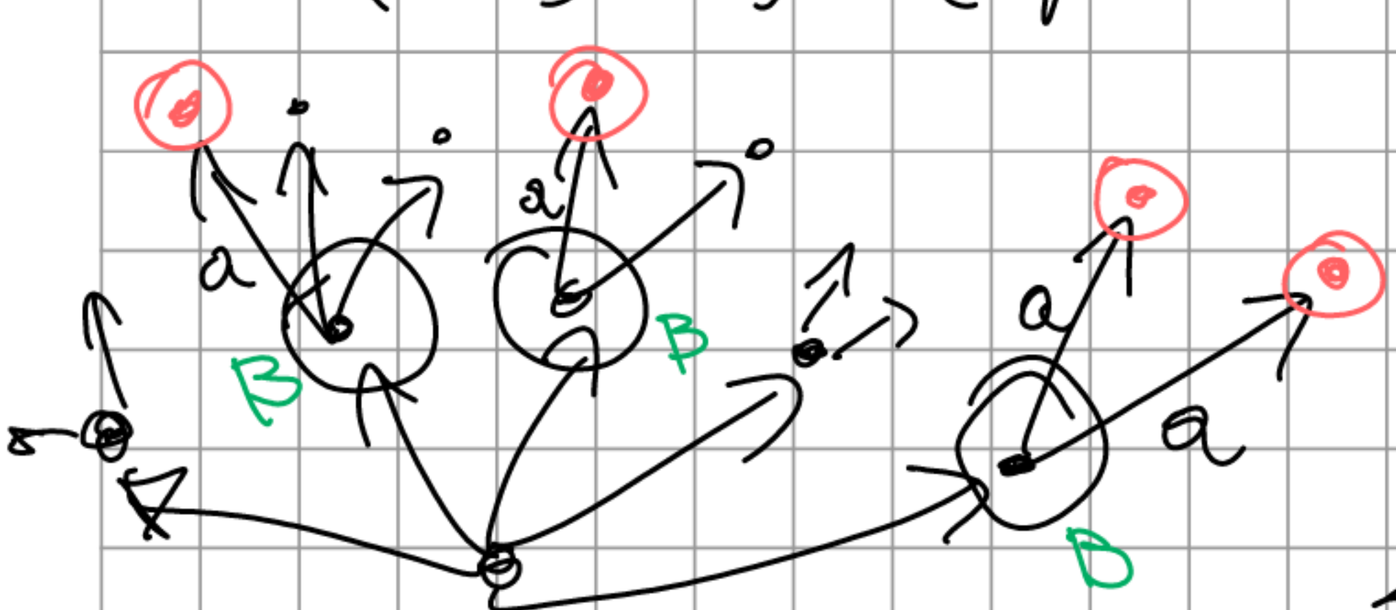
D-d. Weźmy dowolne NFA $A = \langle \Sigma, Q, q_0, \delta, F \rangle$.

Zbudujemy $A' = \langle \Sigma, Q', q_0', \delta', F' \rangle$.

Niech $Q' = \mathcal{P}(Q)$, $q_0' = \{q_0\}$,

$F' = \{ B \subseteq Q : B \cap F \neq \emptyset \}$,

$\delta'(B, a) = \{ q \in Q : \exists p \in B \; \delta(p, a, q) \}$.



Te stany, do których można dojść z któregoś stanu z B po kraw. z etykietą a.

NFA Z ε-PRZEJŚCIAMI

Takie coś, że możemy czasem sobie przejść ze stanu do stanu bez wczytania znaków. Je też można zdeterminizować.

7.03.2022

WYRAŻENIA REGULARNE $(\text{nad } \Sigma)$

- $\emptyset$ jest wyrażeniem regularnym i $L_\emptyset = \emptyset$
- ε jest wyr. reg. i $L_\varepsilon = \{\varepsilon\}$
- jeśli $a \in \Sigma$ to a jest wyr. reg. oraz $L_a = \{a\}$
- jeśli φ, ψ są wyr. reg. to $\varphi + \psi$ jest wyrażeniem regularnym i $L_{\varphi+\psi} = L_\varphi \cup L_\psi$)

$\varphi\psi$ jest wyrażeniem regularnym i $L_{\varphi\psi} = L_\varphi L_\psi$.

$= \{ w_1 w_2 : w_1 \in L_\varphi, w_2 \in L_\psi \}$.

$L_1 L_2 = \{ w_1 w_2 : w_1 \in L_1, w_2 \in L_2 \}$

$L \subseteq \Sigma^*$, wtedy $L^0 = L_\varepsilon$, $L^1 = L$, $L^{i+1} = L^i L$

$$L^* = \bigcup_{n=1}^{\infty} L_n$$

- jeśli φ jest wyr. reg. to φ^* też jest oraz $L_{\varphi^*} = (L_\varphi)^*$

Przykład $\Sigma = \{0,1\}$, $0^*(10^*10^*)^*$

Tw. Niech $L \subseteq \Sigma^*$. Wtedy NWSR:

(1) L jest regularny, tj. istnieje DFA A t.że $L = L_A$.

(2) Istnieje wyrażenie regularne φ t.że $L = L_\varphi$.

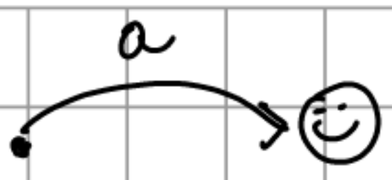
(3) Istnieje NFA z ε-przejściami A' t.że $L = L_{A'}$.

D-d. (3) $\Rightarrow$ (1) Było. ∎

(2) $\Rightarrow$ (3) Indukcja względem długości wyrażenia. Każdy NFA który zbudujemy będzie miał jeden stan wejściowy niebędący akceptującym i jeden akceptujący.

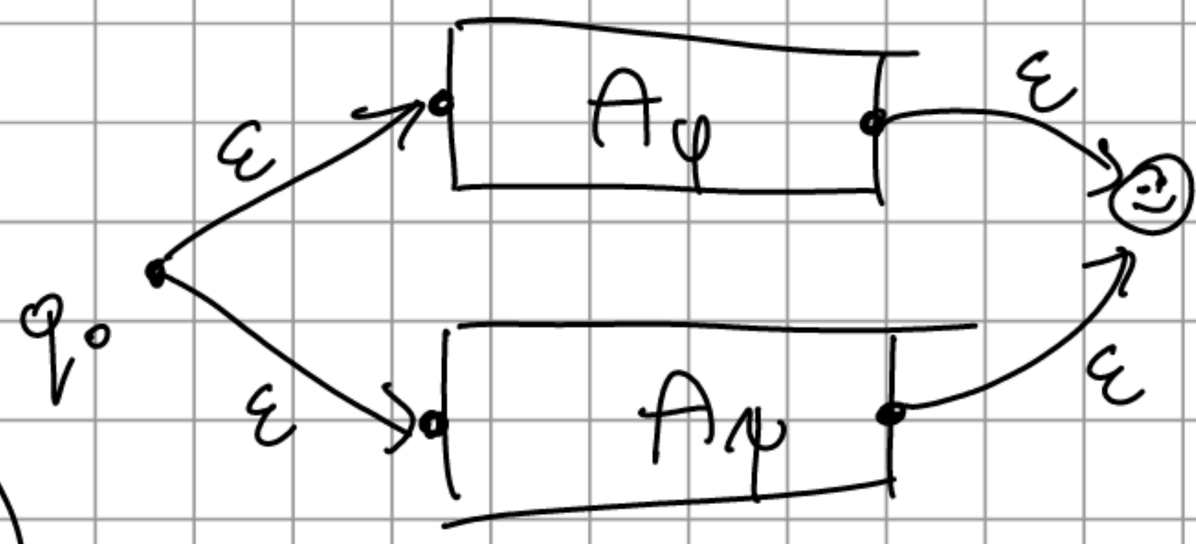
- $\emptyset$ $\rightsquigarrow$ 
- ε $\rightsquigarrow$ 

- $a \rightsquigarrow$



- $\varphi, \psi \rightsquigarrow$

(automat dla $\varphi + \psi$)

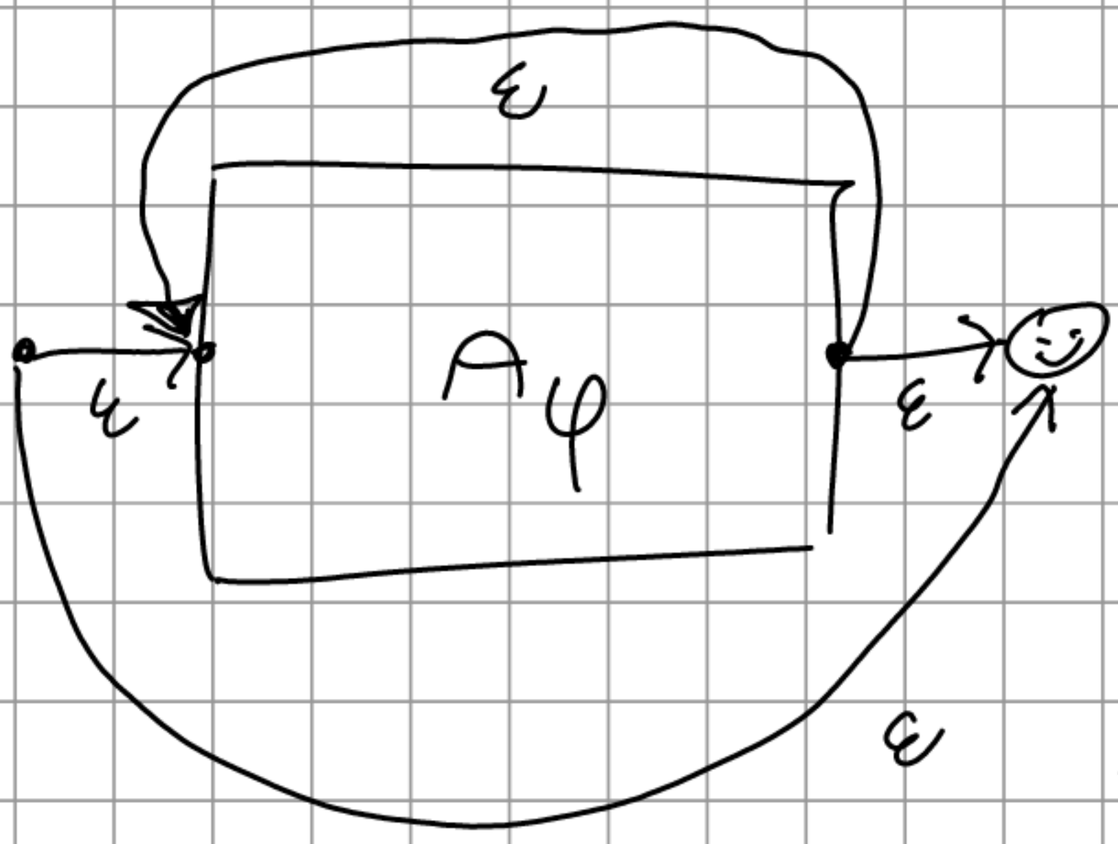


- $\varphi, \psi \rightsquigarrow$

(Automat dla $\varphi\psi$)

- $\varphi \rightsquigarrow$

(Automat dla φ^*)



(1) $\Rightarrow$ (2) Mamy DFA $A = \langle \Sigma, Q, q_0, \delta, F \rangle$

$Q = \{q_1, \ldots, q_n\}$ (gdzie $q_0 = q_1$).

Dla $1 \leq i, j \leq n$, $0 \leq k \leq n$ napiszemy wyrażenie regularne φ_{ij}^k wyrażające język

$\{ w \in \Sigma^* : \hat{\delta}(q_i, w) = q_j \wedge$

$\forall v \in \Sigma^*$ (jeśli v jest niepustym właściwym prefiksem w oraz $\hat{\delta}(q_i, w) = q_l$, to $l < k$) $\}$

niepusty oraz $\neq \varepsilon$

Po drodze z q_i do q_j chodzimy tylko po stanach o indeksach $< k$.

Indukcja względem k:

- $\varphi^0_{i,i} = \varepsilon + \bigoplus_{a \in \Sigma,\ \delta(q_i, a) = q_i} a$ (Suma po literach spełniających warunek)

- $\varphi^0_{i,j} = \bigoplus_{a \in \Sigma,\ \delta(q_i, a) = q_j} a$

- $\varphi^{k+1}_{i,j} = \varphi^k_{i,j} + \varphi^k_{i,k+1} \left(\varphi^k_{k+1,k+1}\right)^* \varphi^k_{k+1,j}$

Mając te wyrażenia regularne piszemy ψ t. że $L_A = L_\psi$:

$$\psi = \bigoplus_{q_i \in F} \varphi^{n}_{1,i}$$

UOGÓLNIENIA

Są dwa możliwe kierunki:

- słowa nieskończone
- drzewa

} można iść w obu tych kierunkach jednocześnie

Automat skończony na słowach nieskończonych.

$\langle \Sigma, Q, q_0, \delta,$ COŚTAM? $\rangle$

Σ ← skończony

Dużo warunków akceptacji Büchiego, Robina...

Interpretacja Büchiego: $F \subseteq Q$.

Słowo jest akceptowane, gdy automat nieskończenie wiele razy odwiedza stany z F.

Pytanie: czy deterministyczne automaty Büchiego robią to samo co niedeterministyczne?

Odpowiedź: nie w ten sposób co poprzednio. Przykład: $|\Sigma| = 1$,

NFA: 

Lepsza odpowiedź: L - zbiór słów nieskończonych nad $\Sigma = \{0,1\}$, w których jest tylko skończenie wiele zer.

L jest rozstrzygany przez 

Deterministycznym się nie da: ćwiczenie.

14.03.2022

Uwaga Klasa języków regularnych nad Σ jest najmniejszą klasą języków nad Σ, która:

- zawiera wszystkie języki skończone $(*)$
- jest zamknięta na sumę (i), konkatenację (ii) i gwiazdkę Kleene'go (iii) $(L \mapsto L^*)$

Istnienie: proste, Najmniejszość: pokazać, że dowolna klasa języków spełniająca powyższe warunki zawiera języki regularne.

GRAMATYKI BEZKONTEKSTOWE

Def. Gramatyka bezkontekstowa (CFG) to krotka $\langle N, \Sigma, S, \Pi \rangle$, $S \in N$, $\Pi \subseteq_{\text{skończone}} N \times (N \cup \Sigma)^*$, $N \cap \Sigma = \emptyset$

CFG - Context Free Grammar

Dygresja A: alfabet, $\Pi \subseteq A^* \times A^*$ skończone

Dla $w, v \in A^*$ definiujemy $w \to_\Pi v$ gdy istnieją słowa $x, y \in A^*$ i para $\langle l, r \rangle \in \Pi$ t.że $w = xly$ oraz $v = xry$

Po ludzku: bierzemy w, znajdujemy w nim infiks l, zamieniamy go na r i dostajemy v.

Relacje $\xrightarrow{*}_\Pi$ i $\xleftrightarrow{*}_\Pi$ definiujemy (odpowiednio) jako tranzytywne i równoważnościowe domknięcie relacji $\to_\Pi$.

- $\xrightarrow{*}_\Pi$ osiągalności w grafie
- $\xleftrightarrow{*}_\Pi$ bycie w jednej spójnej składowej (osiągalność w grafie bez skierowania)

KONIEC DYGRESJI

Dla danej CFG $G = \langle N, \Sigma, S, \Pi \rangle$ przez L_G oznaczamy $\{ w \in (N \cup \Sigma)^* : S \xrightarrow{*}_{\Pi} w \}$, przez $\overline{L_G}$ oznaczamy $\{ w \in (N \cup \Sigma)^* : S \xrightarrow{*}_{\Pi} w \}$, przez $L_G = \overline{L_G} \cap \Sigma^*$

Def. $L \subseteq \Sigma^*$ jest **bezkontekstowy** (CFL), gdy istnieje CFG G t.że $L = L_G$

Obserwacja Każdy język regularny jest bezkontekstowy.

Dowód Klasa CFL spełnia warunki z uwagi (*), (i), (ii), (iii):

(*): proste, dodajemy reguły $S \to w$ dla w z języka

(i): bierzemy sumę rozłączną dwóch gramatyk i dodajemy reguły $\langle S, S' \rangle$, $\langle S, S'' \rangle$ ← nowy S

(ii): $G' = \langle N', \Sigma, S', \Pi' \rangle$, $G'' = \langle N'', \Sigma, S'', \Pi'' \rangle$,

konstruujemy $G = \langle N' \sqcup N'' \sqcup \{S\}, \Sigma, S, \Pi' \sqcup \Pi'' \sqcup \langle S, S'S'' \rangle \rangle$

(iii) Podobnie gwiazdka.

Przykład

- Notacja: $S \to aSa \mid bSb \mid \varepsilon$ oznacza, że $\Pi = \{\langle S, aSa \rangle, \langle S, bSb \rangle, \langle S, \varepsilon \rangle\}$

(to konkretnie daje język palindromów parzystej długości).

- $S \to SS \mid \varepsilon \mid aSb \mid bSa$

$\rightsquigarrow$ języka słów, które mają tyle samo liter a co b.

- $S \to (S) \mid [S] \mid SS \mid \varepsilon$

poprawne nawiasowanie z $(,), [,]$.

Konwencja notacyjna (niemoralna Marcinkowskiego):

Dla języków L, L' piszemy $L = L'$ gdy $L \dot{-} L' \subseteq \{\varepsilon\}$

($\dot{-}$ — różnica symetryczna)

Def CFG $G = \langle N, \Sigma, S, \Pi \rangle$ jest postaci **normalnej Chomskiego**, gdy każda produkcja $\in \Pi$ jest postaci $A \to BC$ dla $A, B, C \in N$ lub $A \to a$ dla $A \in N$, $a \in \Sigma$.

TW. (Chomskyego o postaci normalnej)

Dla każdego CFL L istnieje CFG G w postaci Chomsky'ego t.że $L = L_G$.

Dowód: Nudny :(

Lemat (o pompowaniu dla CFG)

Dla każdego CFL L istnieje $n \in \mathbb{N}$ t.że dla każdego $w \in L$, $|w| \geq n$, istnieją $\leq$ Towa

s, z, t, y, x t. że $|zty| \leq n$, $|zy| > 0$, $w = szty x$
dla każdego $k \in \mathbb{N}$ $sz^k t y^k x \in L$.

Uwaga Czy język $\{a^i b^i c^j : i, j \in \mathbb{N}\}$ jest CFG?

Odp.: TAK (proste)

A co z $\{a^i b^j c^j : i, j \in \mathbb{N}\}$?

Odp.: TAK (to samo)

A czym jest przekrój tych języków?

Odp.: $\{a^i b^i c^i : i \in \mathbb{N}\}$

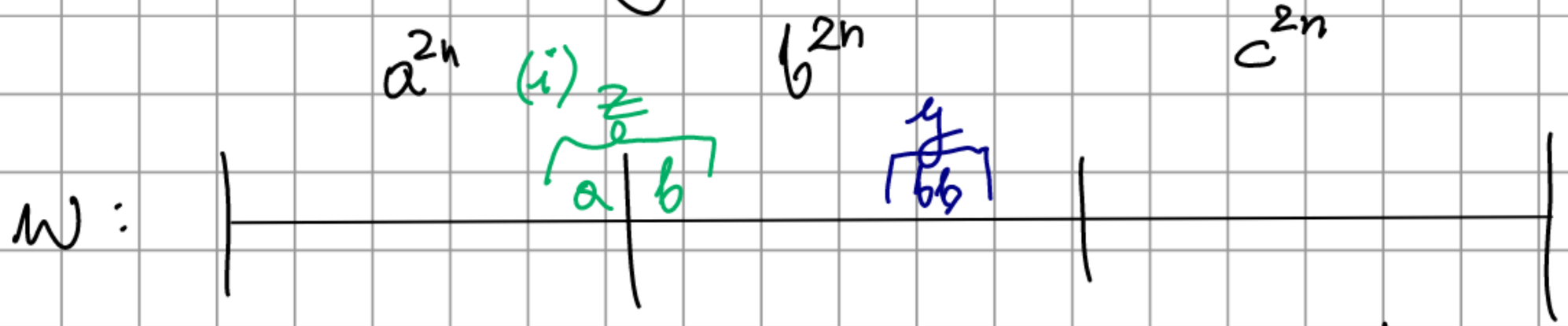
Ten język jednak nie jest CFL!

Dowód Załóżmy, że L jest CF. Niech

n: stała z lematu o pompowaniu.

Niech $w = a^{2n} b^{2n} c^{2n}$. Wtedy są

stałe s, z, t, y, x z lematu.



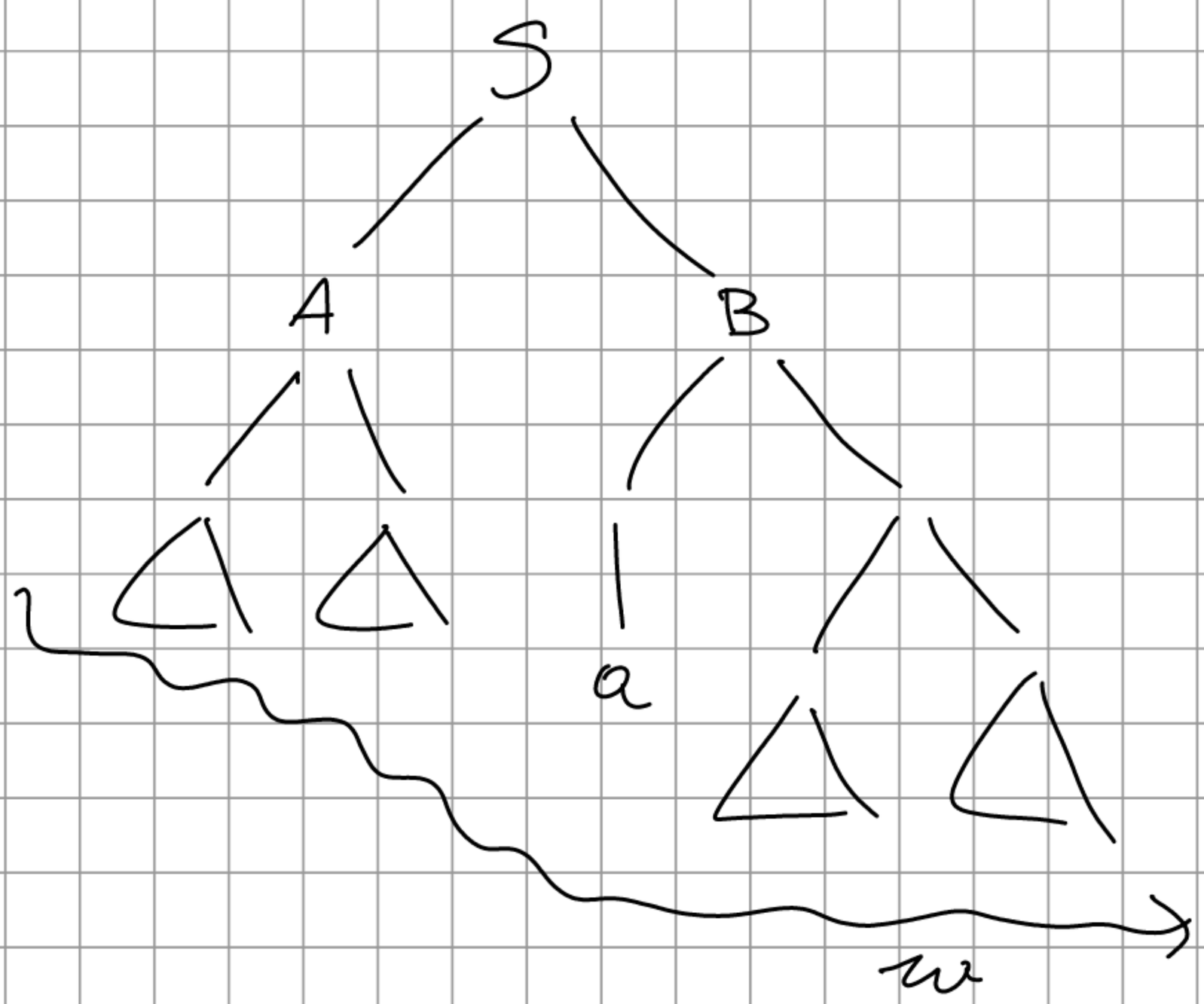
Przypadki: (i) z lub y zawiera dwie różne literki, to dla $k=2$ wygrywamy

(ii) jeżeli x, y mają tylko po jednej literce, to $k=0$ wygrywamy bo tej trzeciej literki jest więcej.

21.03.2022

D-d. (lematu o pompowaniu)

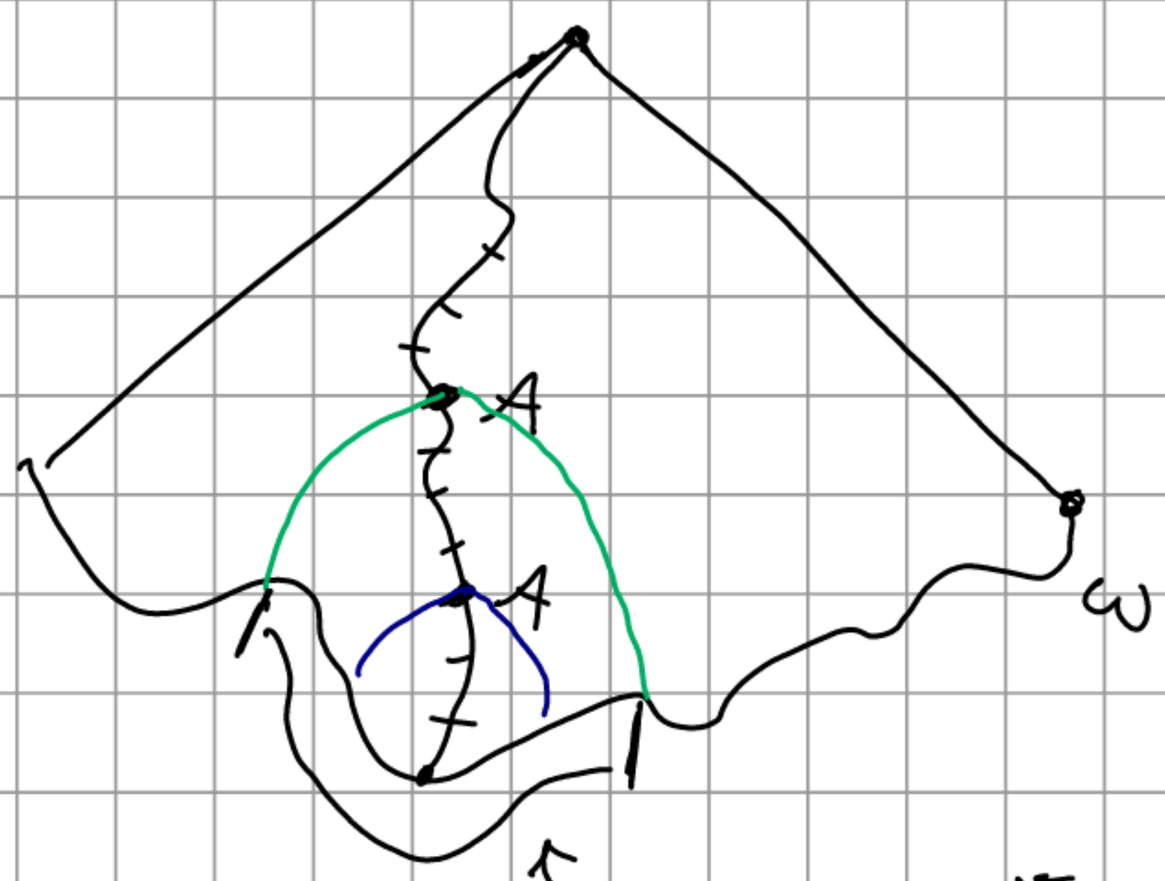
Weźmy $L \in CFL$ oraz CFG $G = \langle N, \Sigma, S, \Pi \rangle$ w postaci normalnej Chomsky'ego i niech $n = 2^{|N|+3}$. Niech $w \in L$, $|w| \geq n$.



Drzewo słowa w.

Jaka jest najdłuższa ścieżka od korzenia do liścia? Co najmniej tyle ma długość $|N|+3$ ($\log n$).

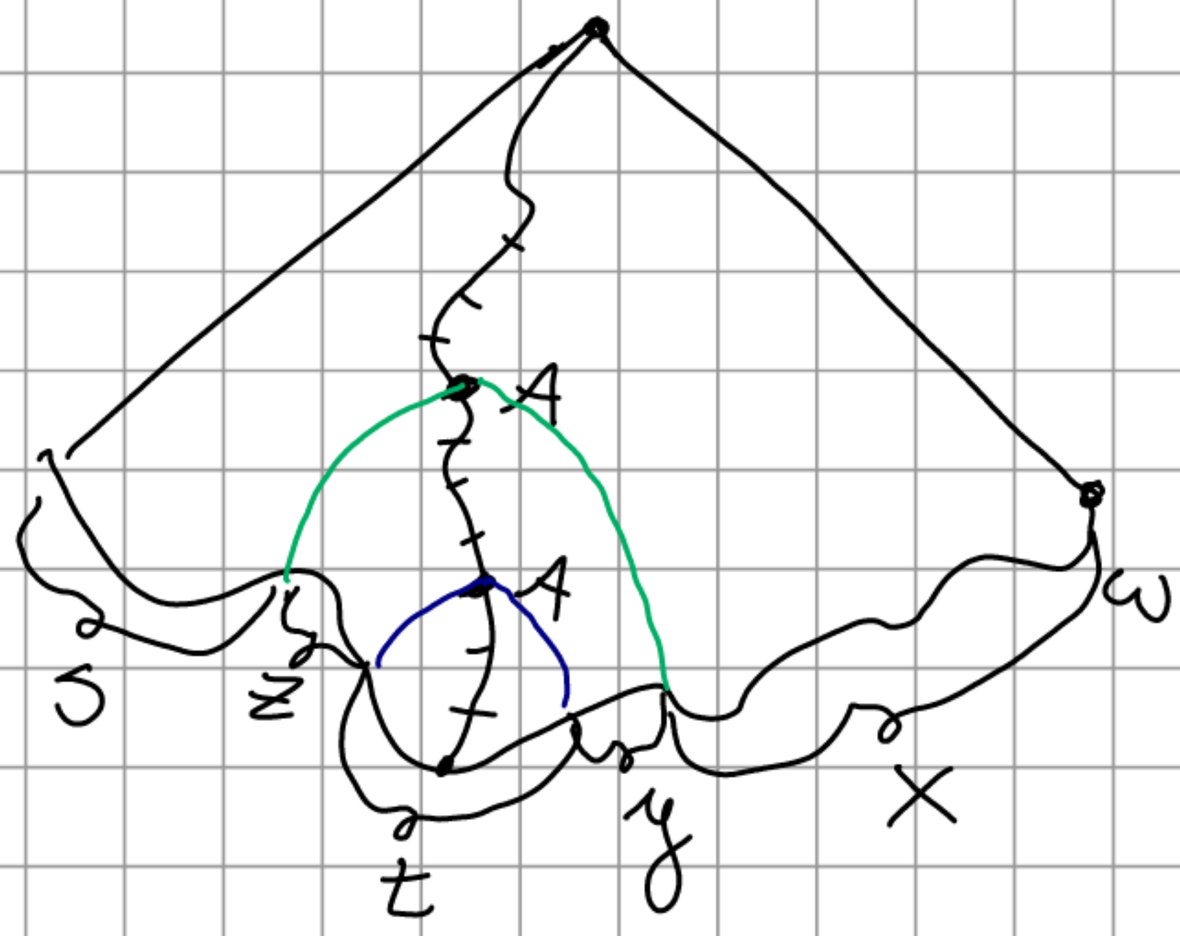
Na tej ścieżce są same nieterminale (poza liściem), zatem musi być jakiś nieterminal, który się powtarza. Niech A będzie takim nieterminalem który jest najniżej na tej ścieżce.



↖ to ma dł. $\leq 2^{|N|+1}$

(ma bo tylko A się może powtórzyć na tej ścieżce)

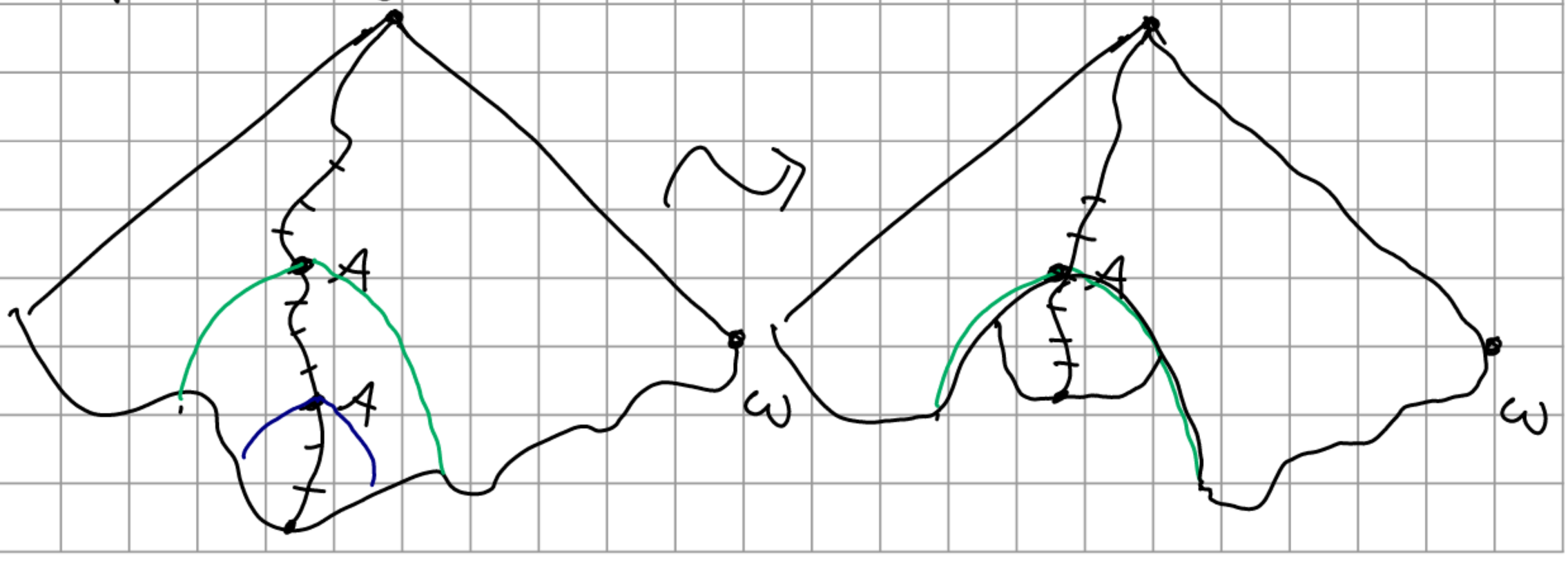
Podział jest naturalny:



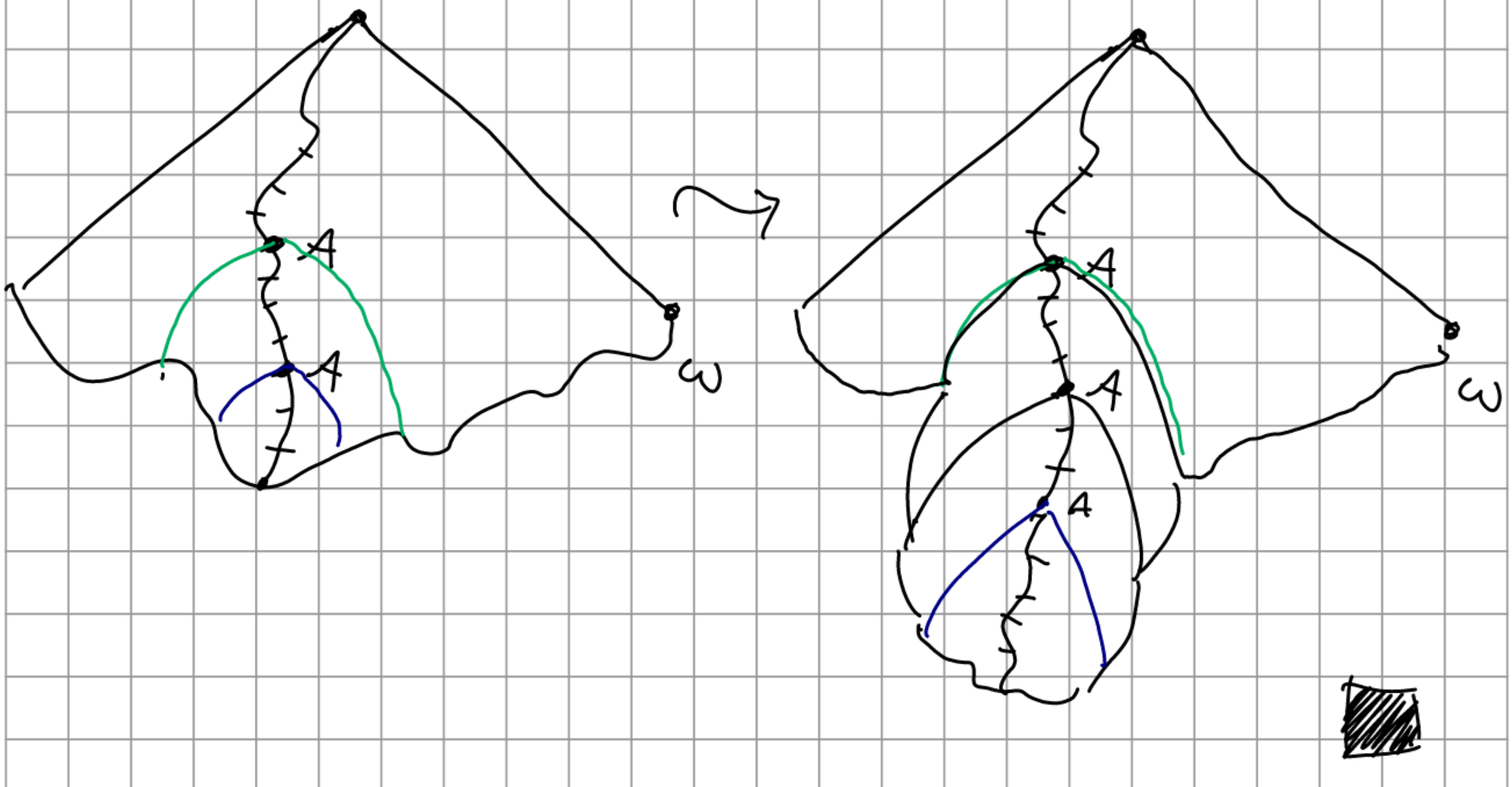
Wtedy $|ztuy| \leq 2^{|N|+1} \leq n$.

Ponadto $|zuy| > 0$ (to niższe wystąpienie A jest potomkiem tylko jednego dziecka tego wystąpienia wyżej).

- gdy $k=0$: "przeszczepiamy" tę niższą produkcję A pod tę większą



- gdy $k \geq 1$: "replikujemy" tę wyższą produkcją tyle ile trzeba.



AUTOMATY ZE STOSEM

Def. **Automat ze stosem** (NPDA: Nondeterministic Push Down Automaton) to krotka

$$A = \langle \Sigma, Q, q_0, S, Z_{\varepsilon S}, \delta \rangle,$$

- Σ — alfabet skończony
- Q — zbiór stanów
- $q_0 \in Q$
- S — zbiór kolorów sweterków (skończony)
- $Z_{\varepsilon S}$ — sweterek dna stosu
- δ — relacja przejścia

gdzie $\delta \subseteq (Q \times (\Sigma \cup \{\varepsilon\}) \times S) \times (Q \times S^*)$

oraz Z ma takie własności:

- $\delta(q, a, Z, q', w) \Rightarrow w = Zv$ gdzie v nie zawiera Z.
- $\delta(q, a, A, q', w) \wedge A \neq Z \Rightarrow w$ nie zawiera Z.

Wtedy $\hat{\delta} \subseteq \Sigma^* \times (Q \times S^*)$ jest najmniejszą relacją spełniającą:

- $\hat{\delta}(\varepsilon, q, Z)$
- $\hat{\delta}(w, q, VA) \wedge \delta(q, a, A, q', W) \Rightarrow \hat{\delta}(wa, q, VW)$ $(a \in \Sigma \cup \{\varepsilon\})$.

Wtedy $L_A = \{w : \exists q \ \hat{\delta}(w, q, Z)\}$.

Przykład Palindromy parzyste:

$$\delta(q_1, B, A, q_1, AB)$$
$$\delta(q_1, \varepsilon, A, q_2, A)$$
$$\delta(q_2, A, A, q_2, \varepsilon)$$

Przykład Słowa w t. że $|w|_0 = |w|_1$.

Tw. Klasa języków bezkontekstowych jest równa klasie języków wyrażanych przez NPDA.

Uwaga (bardzo ważna) Klasy rozstrzygnane przez maszyny deterministyczne są zamknięte na dopełnienia.

Pytanie Czy w sformułowaniu tw. nie można zmienić NPDA na PDA:

$\{a^i b^i c^i : i \in \mathbb{N}\}$ nie jest CFL

ale dopełnienie jest.

Def. Zanim dd, wprowadzimy pojęcie postaci normalnej Greibach: CFG G jest w tej postaci gdy dla każdego $\langle A, w\rangle \in \Pi$ $w \in \Sigma N^*$.

Lemat Dla każdego L: CFL istnieje G w postaci normalnej Greibach t.że $L = L_G$

Dowód tw. "⇒" Weźmy CFL L oraz CFG G w postaci normalnej Greibach t.że $L = L_G$.

Zbudujemy NPDA $\langle \Sigma, Q, q_0, S, Z, \delta \rangle$ t.że $L = L_A$.

- Jeśli w Π jest produkcja $A \to a w$, $w \in N^*$, to $\delta(q, a, A, q, w^R)$,
- $\delta(q_0, \varepsilon, Z, q, ZS)$,

gdzie $Q = \{q_0, q\}$, $S = N$

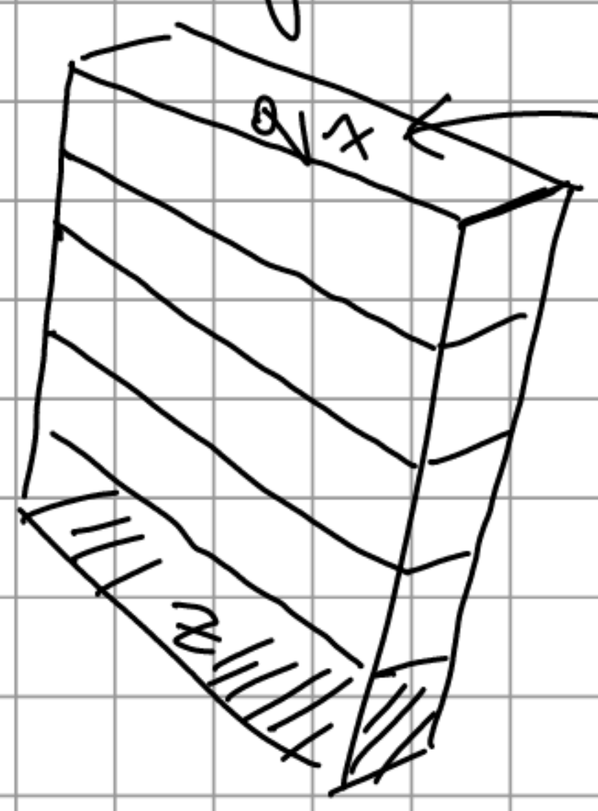
28.03.2022

Kontynuacja dowodu NPDA → CFG

NPDA → NPDA z JEDNYM STANEM → CFG

tą część już potrafimy analogicznie do poprzedniej części dowodu.

Teraz stos to kolorowe klocki, na wierzchu których napisany jest jakiś stan



"wyobraźmy sobie, że oglądając ten klocek jesteś w stanie q_7"

Problem pojawił się przy ściąganiu klocków. Ten napisany stan może być już niedeterministyczny.

stary automat

Np.: $\langle q_7, \text{czerwony}, a, q_3, \text{ziel-nieb-ziel-czerw} \rangle$

nowy automat

$\langle -, \langle q_7, \text{czerw} \rangle, a, \langle q_3, \text{ziel} \rangle, \langle ?, \text{nieb} \rangle, \langle ?, \text{ziel} \rangle, \langle ?, \text{czerw} \rangle \rangle$

co tutaj?

no nie wiadomo

Poprawa: teraz klocki będą takie:

Trzpieni jest tyle co stanów.

Nasze stany: $\langle$ trzpień, dziura, kolor $\rangle$

Instrukcja starego automatu:

$\langle q, K, a; q', A_1 A_2 \dots A_l \rangle$, $l \geq 1$

wierzch

zastępujemy zbiorem wszystkich instrukcji postaci

(pomijamy stan: jest tylko jeden)

$$\langle [d_0, k, q], a ; [d_0, A_1, d_1]\,[d_1, A_2, d_2] \ldots [d_{L-1}, A_L, q'] \rangle$$

$[d_0, k, q]$: d_0 — dziurka, k — kolor, q — trzpień

(kwantyfikujemy po $d_0, d_1, \ldots, d_{L-1}$)

Instrukcje starego automatu:

$\langle q, k, a ; q', \varepsilon \rangle$

$\hookrightarrow \langle [q', k, q], a ; \varepsilon \rangle$

Poprawność Jasne jest, że każdy przebieg starego automatu można zasymulować nowym. W drugą stronę: nie wprost w mierę łatwo.

Kilka szczegółów o których nie chcemy rozmawiać: co z dnem stosu? Co z akceptowaniem?

DRUGA CZĘŚĆ KURSU

Wcześniej: Język $\subseteq \Sigma^*$

Teraz: Problem $\subseteq \mathbb{N}$

Będziemy pisać programy, które wczytują jedną liczbę naturalną, a jeśli zwrócą wynik, to on też będzie liczbą naturalną.

Def. $A \subseteq \mathbb{N}$ nazywamy **rekurencyjnym** (obliczalnym, rozstrzygalnym) jeśli istnieje program φ t.że dla każdej $n \in \mathbb{N}$:

$$\begin{cases} \varphi(n) = 0 \iff n \notin A \\ \varphi(n) = 1 \iff n \in A \end{cases}.$$

Obserwacja Każdy zbiór skończony jest rekurencyjny. Klasa zbiorów rekurencyjnych jest zamknięta na sumę, przecięcie i dopełnienie.

Obserwacja Istnieją nierekurencyjne podzbiory $\mathbb{N}$.

Def. Podzbiór $A \subseteq \mathbb{N}$ jest **rekurencyjnie przeliczalny** (r.e.: recursively enumerable) gdy istnieje program φ t.że dla każdego $n \in \mathbb{N}$:

(i) $\varphi(n) = 1 \iff n \in A$ (równoważnie definicje)

(ii) $\varphi(n) \in \mathbb{N} \iff n \in A$ $\wedge$

$\varphi(n) = \perp \iff n \notin A$

φ się zapętla na n.

Obserwacja Klasa r.e. jest zamknięta na przecięcie i sumę.

Obserwacja Jeżeli A jest r.e. i $\mathbb{N} \setminus A$ jest r.e., to A i $\mathbb{N} \setminus A$ są rekurencyjne.

Numerujemy wszystkie programy EFEKTYWNIE, tzn. mamy inny program, który może wczytać program i zwrócić jego numer oraz dla numera zwraca program.

	1	2	3	4	5	...
φ_1	1	1	1			
φ_2	4	2137	28			
φ_3	1	1	1			
φ_4	0	7	1			
⋮						

$K = \{ n : \varphi_n(n) \in \mathbb{N} \}$ / zbiór tych miejsc na przekątnej

Obserwacja K jest r.e. gdzie jest liczbą, czyli te programy

Umiemy policzyć φ_n które się zatrzymują

i uruchamiamy go na n. dla swojego numeru

TW. (Turinga o nierozstrzygalności problemu stopu)
K jest nierozstrzygalny.

D-d. Dowód nie wprost. Załóżmy że K jest rozstrzygany przez pewien program ψ.
Niech φ będzie programem, który:

- wczytuje n,
- jeśli $\psi(n)=1$, to się zapętl,
- w p.p. zwróć 1.

Wtedy φ ma pewien numer n.

- Jeśli $\psi(n)=1$, to znaczy, że φ nie zapętla się na n, ale z definicji φ powinien się zapętlić na n.
- Jeśli $\psi(n)=0$, to znaczy, że φ nie powinien się zatrzymać, ale z jego definicji φ się zatrzyma na n.

$n \in K \iff \varphi_n(n) \in \mathbb{N} \iff \psi(n)=0 \iff n \notin K$ ↯

4.04.2022

Def. Funkcja rekurencyjna to relacja wejście - wyjście dla programu w MUJP.

Obserwacja Funkcje rekurencyjne to częściowe funkcje $\varphi : \mathbb{N} \to \mathbb{N}$. Jak weźmy podklasa: funkcje rekurencyjne całkowite.

Uwaga Zbiór $A \subseteq \mathbb{N}$ jest r.e. $\iff$ istnieje f rek. t.że $A = dom(f)$

Def. Niech $A, B \subseteq \mathbb{N}$. Wtedy $A \leq_{rek} B$ (czytaj "A jest nie trudniejszy od B ze względu na redukcje całkowite rekurencyjne") gdy istnieje całkowita funkcja rek. f (zwana redukcją) t.że $\forall n \in \mathbb{N}$ $(n \in A \iff f(n) \in B)$

Obserwacja (1) $A \leq_{rek} B$ i $B \leq_{rek} C$, to $A \leq_{rek} C$

(2) $A \leq_{rek} B$ i B jest rekurencyjny, to A też.

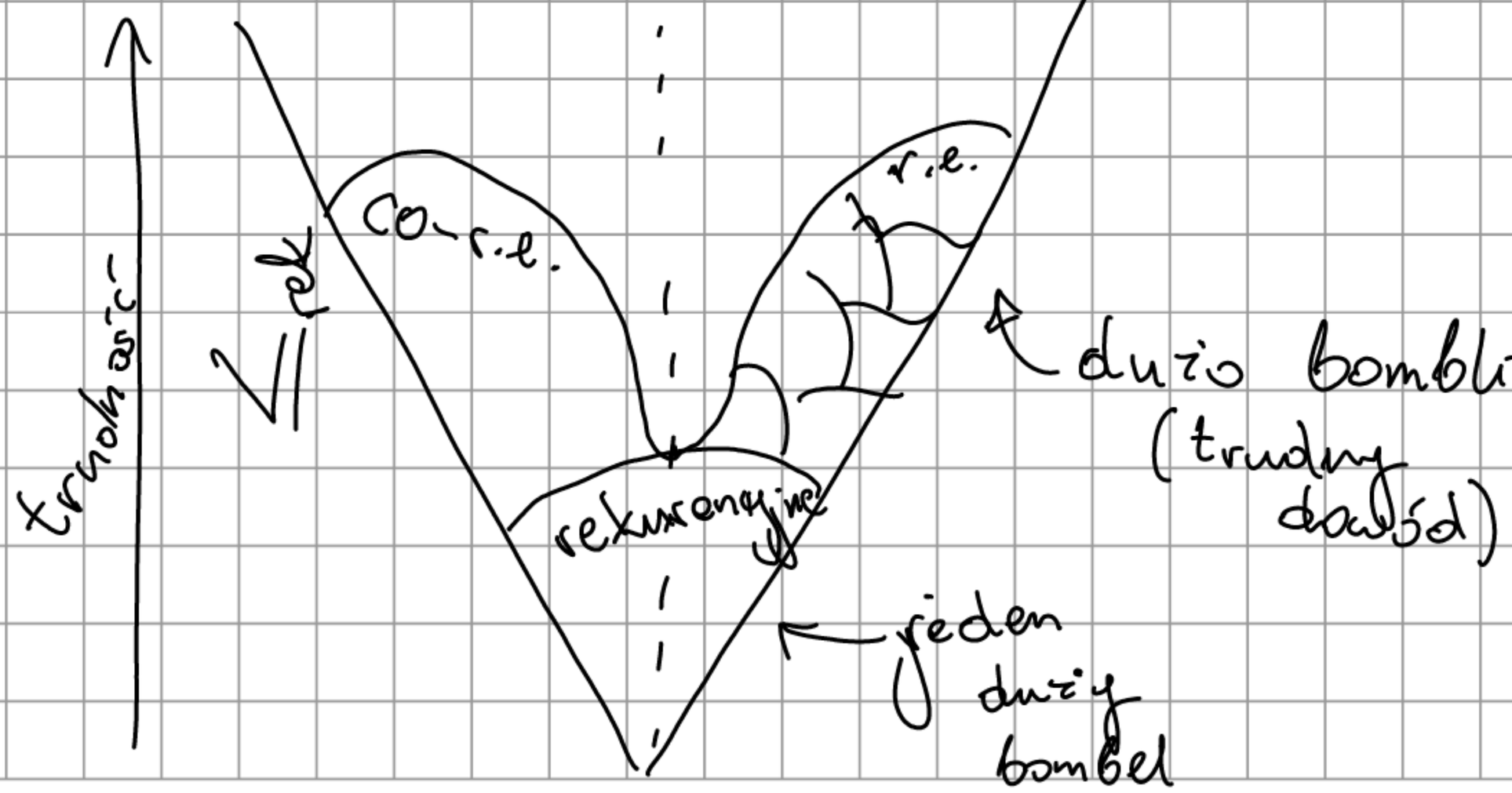
(3) $A \leq_{rek} B$ i B jest r.e., to A też jest r.e.

(4) Jeżeli A, B rekurencyjne i nietrywialne, ($\notin \{\emptyset, \mathbb{N}\}$), to $A \leq_{rek} B$ i $B \leq_{rek} A$

D-d. (4) Weźmy $b \in B$, $b' \in \mathbb{N} \setminus B$. Niech f_A: rozstrzyga A. Zbudujemy redukcję f:

$f(n) =$ "wczytaj n, uruchom $f_A(n)$, jeśli wyszło 0, to zwróć b', jeśli wyszło 1, to b"

$\leq_{rek}$ dzieli $P(\mathbb{N})$ na "bomble"



Tw. K (z popr. wykładu) jest zupełny w klasie r.e. ze względu na całkowite redukcje rekurencyjne, tj. dla każdego $B \in$ r.e. zachodzi $B \leq_{rek} K$

Przykład $A_7 = \{n : \varphi_n(7) = 77\}$ jest r.e. Pokażemy, że $K \leq_{rek} A_7$. Budowa redukcji:

- przynosząc nam n,
- piszemy program:
  wczytaj m
  uruchom $\varphi_n(n)$
  zwróć 77
- zwracamy numer tego programu

To jest funkcja całkowita i do tego dzieła. Gdy $n \in K$, to $\varphi_{f(n)}$ się kończy i zwraca 77 dla dowolnego wejścia, więc $f(n) \in A$. Gdy $n \notin K$, to $\varphi_{f(n)}$ się nie skończy dla każdego

wejście, więc $f(n) \notin A_z$.

Def. $A \subseteq \mathbb{N}$ jest **ekstensjonalny** jeśli

$\forall i \in A, j \notin A \quad \exists n \quad \varphi_i(n) \neq \varphi_j(n)$

↑ może być pinezka

Przykład $A = \{n : \varphi_n$ jest całkowita$\}$

Alt. definicja A ekstensjonalny gdy $\forall i,j \in \mathbb{N}$

jeśli $\varphi_i = \varphi_j$ (jako funkcje), to $i \in A \Leftrightarrow j \in A$

Tw. (Rice'a) Żaden nietrywialny zbiór ekstensjonalny nie jest rekurencyjny.

D-d. Weźmy $A \subseteq \mathbb{N}$: nietrywialny i ekstensjonalny. BSO przyjmijmy, że żaden "numer funkcji pustej" (same pinezki) nie należy do A (zawsze możemy wziąć A^c).

Pokażemy, że $K \leq_{rek} A$. Niech $k \in A$.

Konstruujemy redukcję f:

- dają nam n
- piszemy program:

```
wczytaj m
oblicz φ_n(n)
oblicz φ_k(m)
i zwróć wynik
```

- zwróć jako $f(n)$ numer tego programu.

Σ - alfabet skończony, $\Pi \subseteq \Sigma^* \times \Sigma^*$, skończony

$w \to_\Pi v$: relacja na Σ^*

$\overset{def}{\Leftrightarrow}$ $w = w_1 l w_2$, $v = w_1 r w_2$, $\langle l, r \rangle \in \Pi$,

$w \xrightarrow{*}_\Pi v$: przechodnie domknięcie $\to_\Pi$

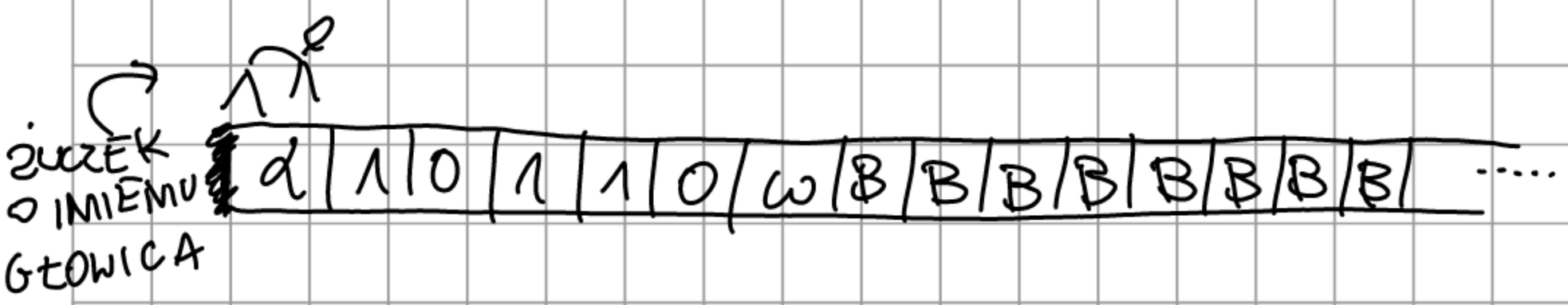
$w \xleftrightarrow{*}_\Pi v$: równoważnościowe domknięcie $\to_\Pi$.

Tw. (o nierozstrzygalności problemu słów Thuego)

Problemy $SemiThue = \{\langle w, v, \Pi \rangle : w \xrightarrow[\Pi]{*} v\}$,

$Thue = \{\langle w, v, \Pi \rangle : w \xleftrightarrow{*}_\Pi v\}$ są nierozstrzygalne.

11.04.2022 MASZYNA TURINGA



$A = \{\alpha, \omega, 0, 1, B\}$

↑ blank, początkowo tym wypełniona taśma

Q : skończony zbiór stanów, $q_0 \in Q$: stan początkowy,
$q_F \in Q$: stan końcowy

$\delta : Q \times A \longrightarrow Q \times A \times \{L, R\}$

↑ aktualny stan; ↑ litera pod głowicą; ↑ nowy stan; ↑ na co zmienić literę pod głowicą; ↑ przesunąć głowicę w lewo lub prawo.

Własności:

- α jest znakiem końca taśmy, nie można go napisać, zmazać, ani pójść na lewo stojąc na α
- widząc B należy coś napisać
- δ jest f. częściową

- wynikiem jest ciąg zmian na prawo od w.
- nie ma ruchów z q_F.

Teraz maszyna Turinga to nasz MUJP, dalej mamy K "z tym głowy".

<u>Teza Churcha</u> Każdy algorytm daje się przedstawić jako maszyna Turinga.

<u>D-d.</u> (tw. o nierozstrzygalności zbiorów SemiThm)

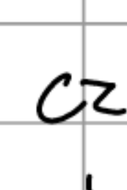
(Dają π, v, w i pytają czy $w \xrightarrow{*}_{\pi} v$)

Pokażemy, że $K \leq_{rek} SemiThm$.

n-ta maszyna Turinga

Konstrukcja redukcji: dają nam $M_{n,n}$

mamy zwrócić w, v, π takie, że $n \in K \Leftrightarrow w \xrightarrow[\pi]{*} v$

Niech $\Sigma = Q \cup A \cup \{$  $\}$,

↖ pozostałe

$w = q_0 \underbrace{1 \ldots 1}_{n} \omega B$

↑ n zapisane binarnie

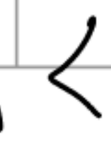
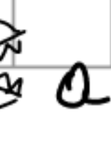
Konstrukcja Π:

- dla każdej "instrukcji" z δ, która ma postać $\delta(q, a) = \langle q', a', L \rangle$, gdzie $a \neq B$, dajemy do Π parę $\langle aq, q'a' \rangle$

- dla każdej instrukcji z δ, która ma postać $\delta(q, B) = \langle q', a', L \rangle$, do Π dajemy $\langle Bq, q'a'B \rangle$ (powiększenie taśmy o blanka)

- dla każdej instrukcji z δ, która ma postać $\delta(q, a) = \langle q', a', R \rangle$, $a \neq B$ do Π dajemy $\langle aqb, a'bq' \rangle$ dla każdego $b \in A$.

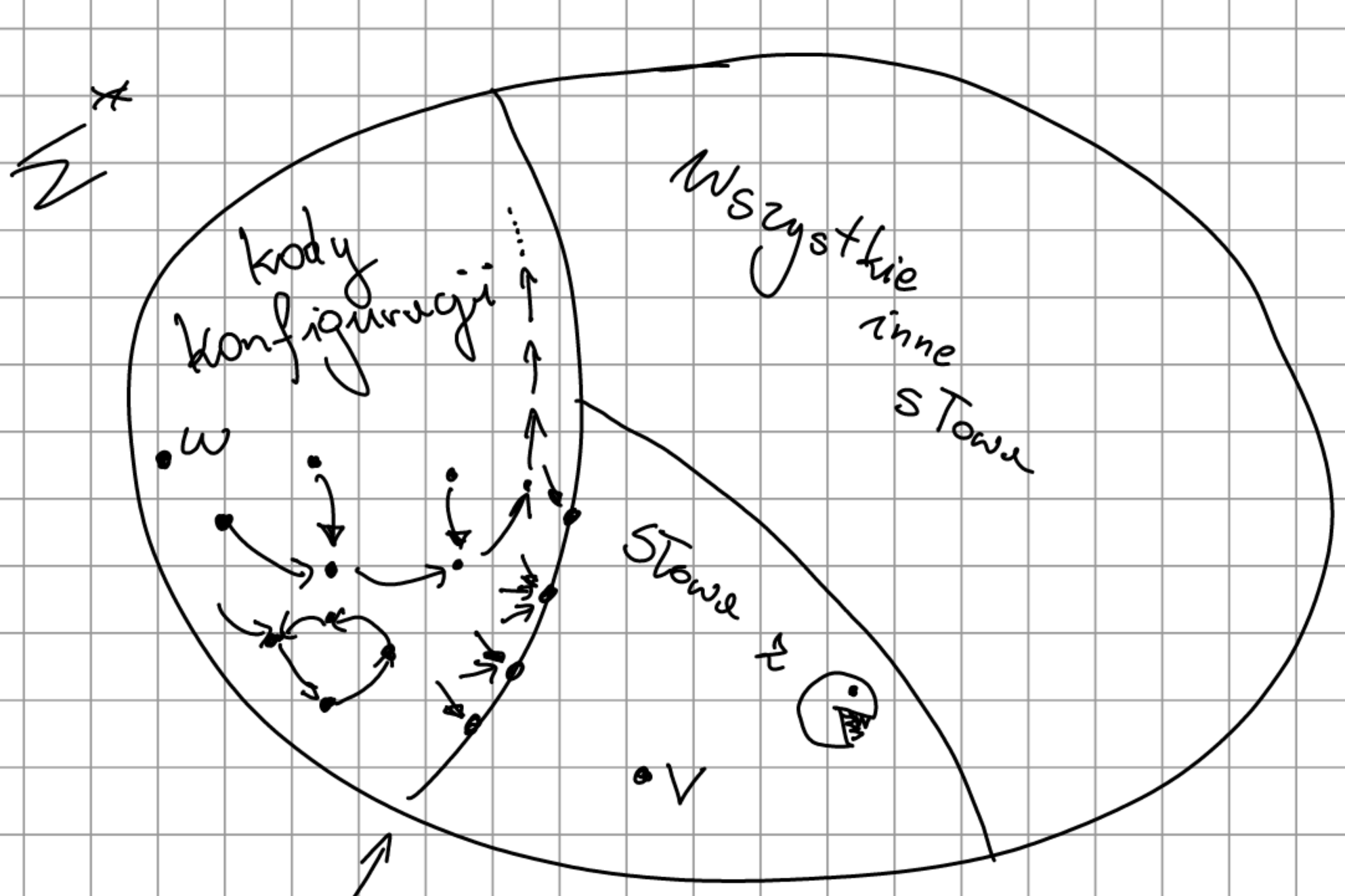
- dla każdej instrukcji z δ, która ma postać $\delta(q, B) = \langle q', a, R \rangle$, do Π dajemy $\langle Bq, aBq' \rangle$

- dla każdego $a \in A$ w Π są takie pary: $\langle a$  $, $  $ \rangle, \langle$$, a, $  $ \rangle$

• dodajemy do Π parę $\langle q_F, \text{🐊} \rangle$

Teraz $V = \text{🐊}$.

kod: konfiguracje $M_n \longrightarrow$ słowo nad Σ



D-d. (tw. o nierozstrzygalności problemu Thue)

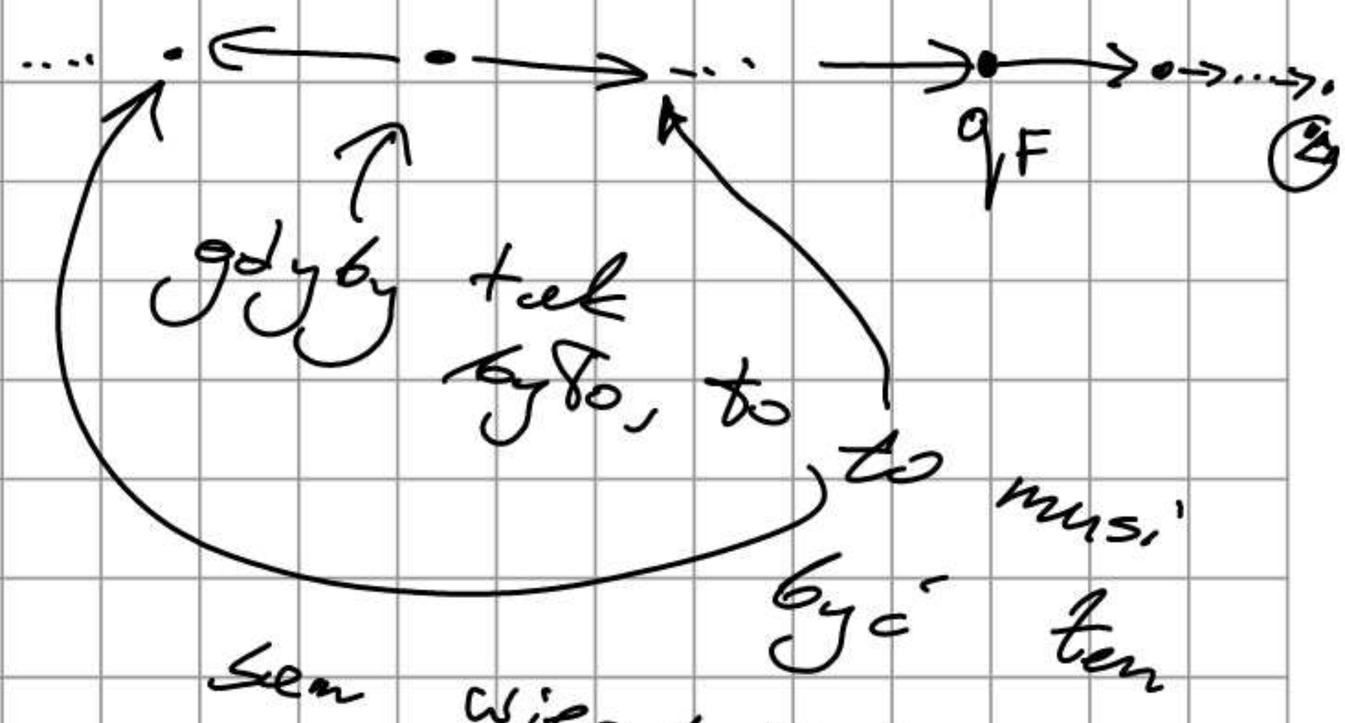
$\{\langle w, v, \Pi \rangle : w \overset{*}{\longleftrightarrow}_\Pi v\}$

W zasadzie to samo, co poprzednio.

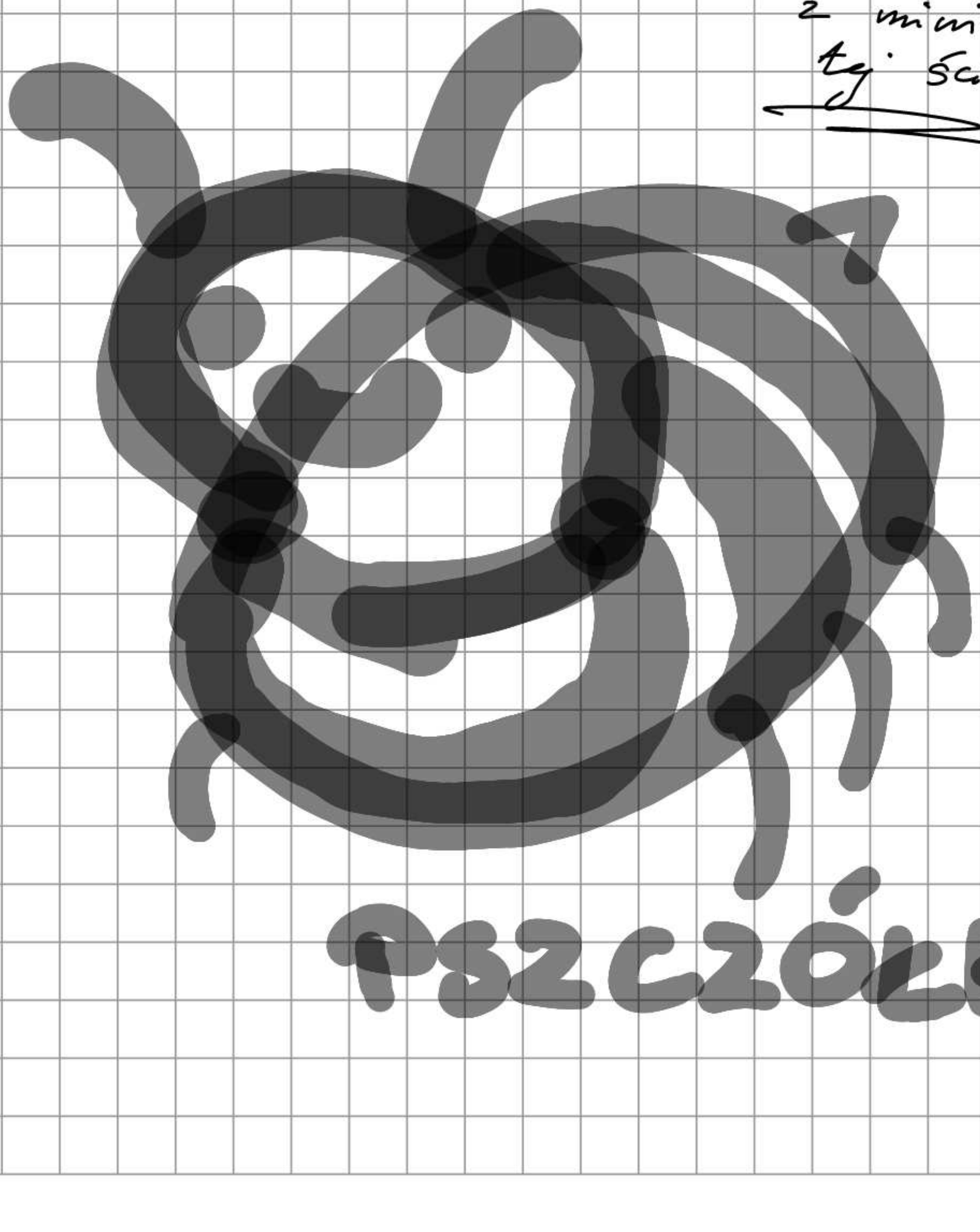
Trzeba tylko pokazać, że gdy $n \notin K$, to $w \not\overset{*}{\leftrightarrow}_\Pi v$

Widać z tego obrazka wyżej!

Nr



$\Rightarrow$
sprzeczność
z minimalnością
tej ścieżki.



PSZCZÓŁKA

Dwukierunkowe Automaty Skończone

Q: zbiór stanów, $q_0, q_F \in Q$

$\Sigma' = \Sigma \cup \{\alpha, \omega\}$

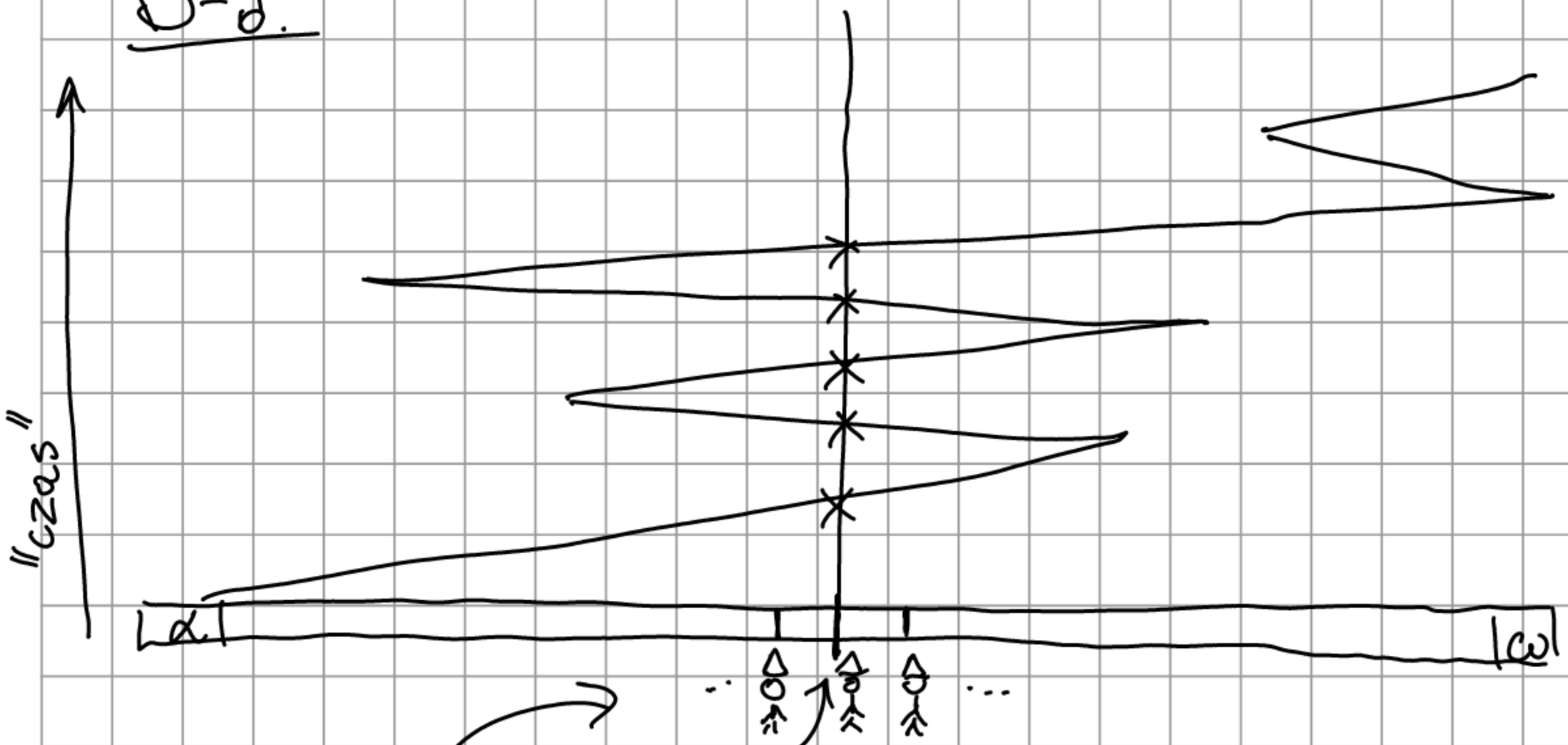
$\delta: Q \times \Sigma' \longrightarrow Q \times \{L, R\}$, musi kończyć w ω

Tw. To model wyraża tylko języki regularne

(Chcemy z pszczółki zrobić żuczka)

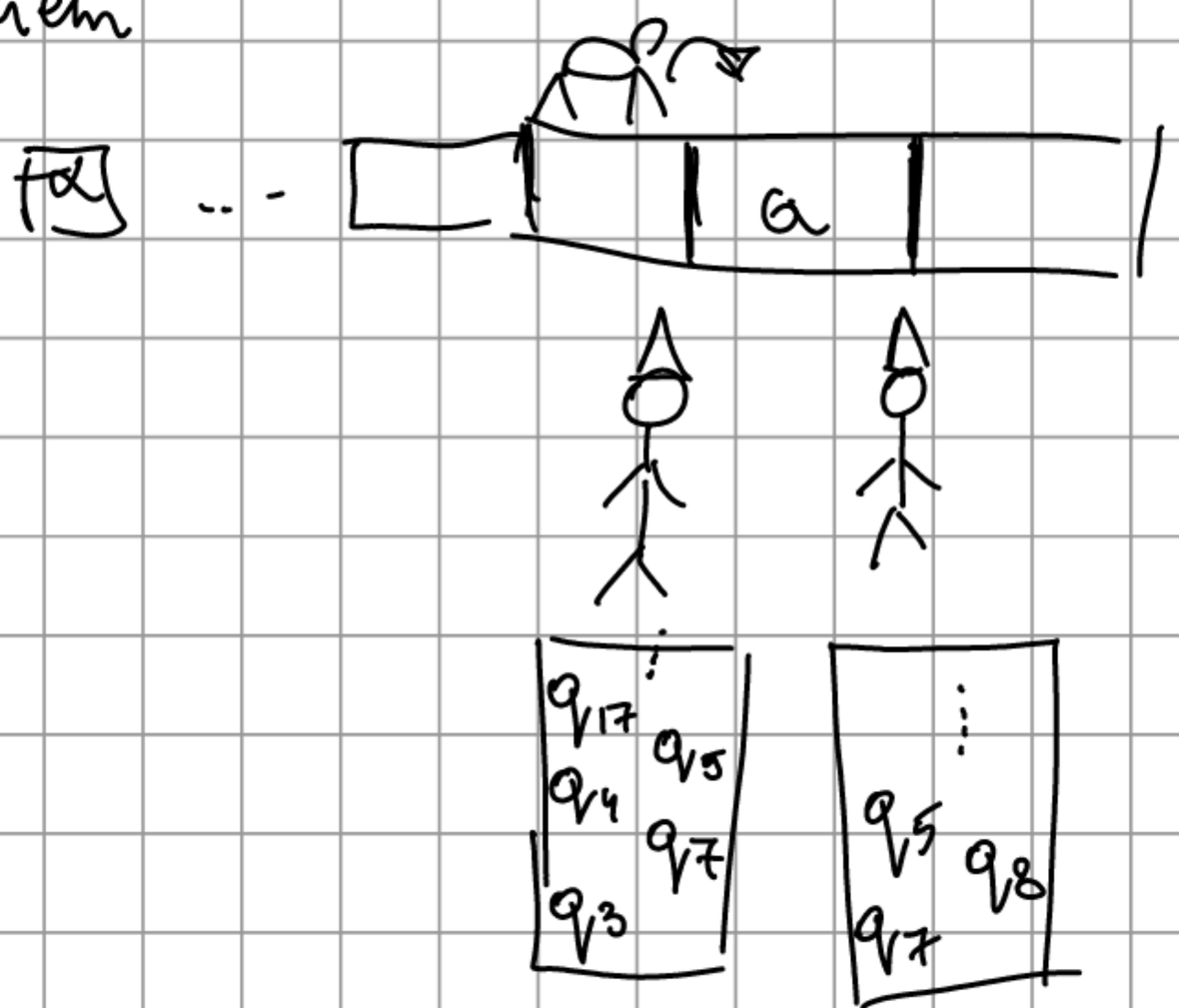
(NFA)

D-d.



w przejściach między komórkami jest krzyżówek punktów przecięcia jest max. $2|Q|$

Każdy krasnoludek sporządza listę stanów, z których pszczółka przekupstwa może krasnoludkiem



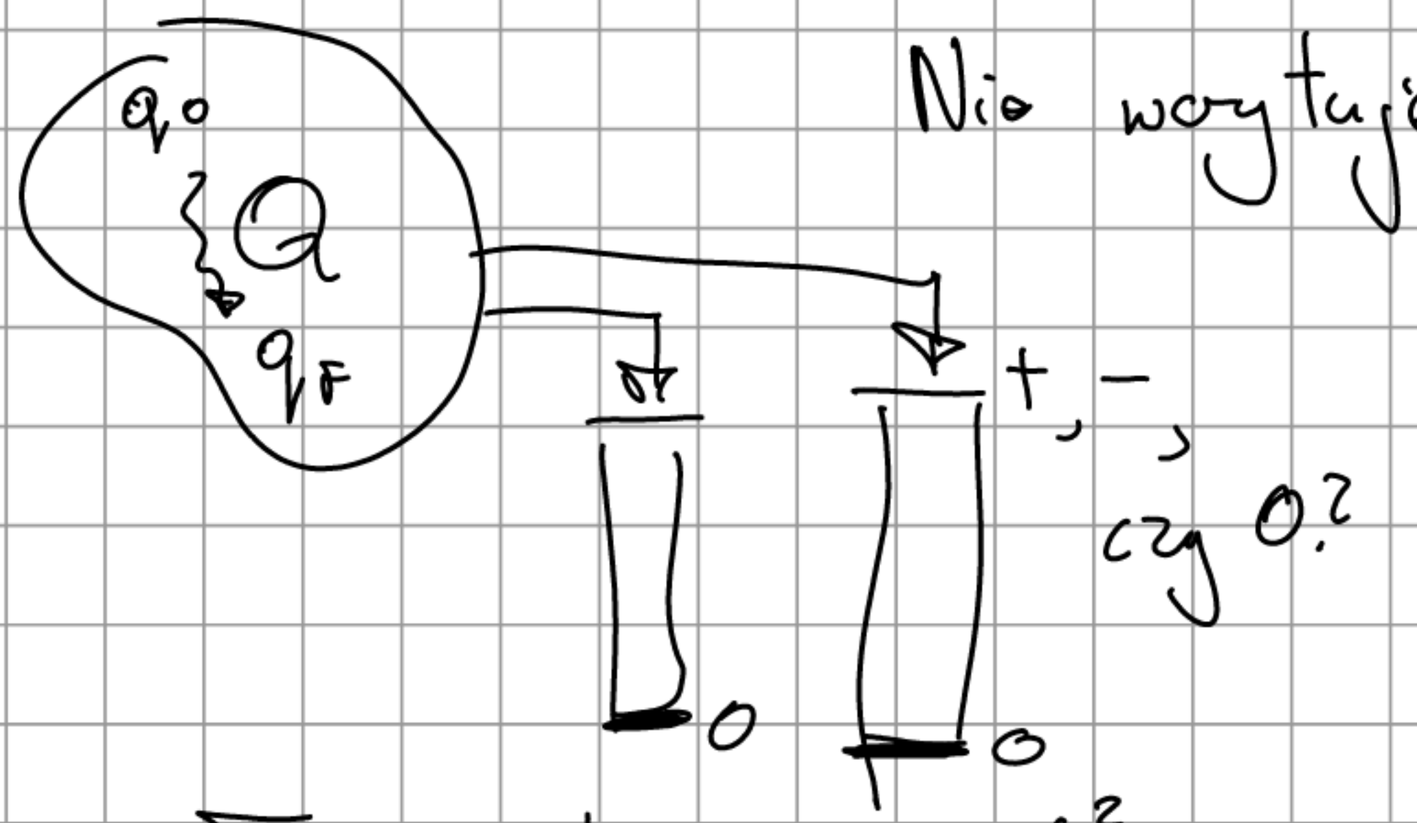
sprawdza, czy te listy są kompatybilne

Fakt Pszczółka nie istnieje: są tylko krasnoludki opowiadające bajki

Obserwacja Niedeterministyczna pszczółka jest Tobie samo.

9.05.2022 NIEROZSTRZYGALNOŚĆ ARYTMETYKI

Maszyny Minsky'ego : $\langle Q, q_0, q_F, \delta \rangle$



Nie wszystkie mic — bawi się sam ze sobą.

czy 0?

$$\delta : Q \times \{\text{zero}, \text{nie-zero}\}^2 \longrightarrow Q \times \{+1, 0, -1\}^2$$

TW Problem rozstrzygania czy dana MM zakończy działanie jest nierozstrzygalny. (Problem MM)

- Logika I rzędu w $+, \cdot, \uparrow$

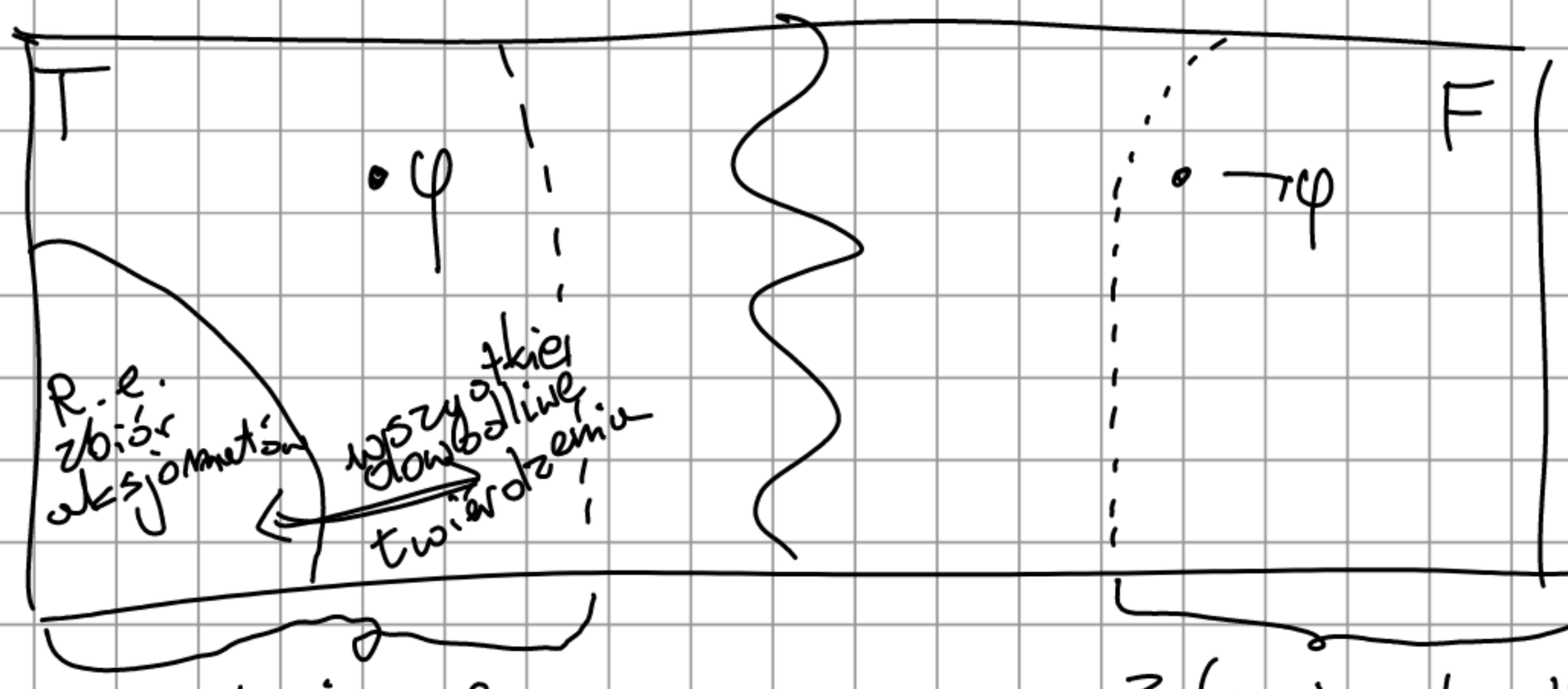
$$\forall n \exists m \forall k, l \left(m > n \wedge \left(k \cdot l = m \rightarrow \left(k = 1 \vee l = 1 \right) \right) \right)$$

- Aksjomaty arytmetyki Peano $\langle N, +, \cdot \rangle$

Pytanie: czy każde prawdziwe zdanie $Th(N, +, \cdot)$ jest dowodliwe w arytmetyce Peano?

ODP: NIE.

WSZYSTKIE ZDANIA ARYTMETYKI



Gdyby wszystkie zdanie arytmetyki dało się udowodnić, to $Th(\mathbb{N},+,\cdot)$ byłby rozstrzygalny.

IW. $Th(\mathbb{N},+,\cdot,\uparrow)$ nie jest rozstrzygalny.

D-d. Pokażemy, że $MM \leq_{rek} Th(\mathbb{N},\cdot,+,\uparrow)$

Czyli dlaj nam program M dla MM.

Mamy napisać zdanie Φ_M t.że

M się zatrzymuje $\iff (\mathbb{N},\cdot,+,\uparrow) \models \Phi_M$

Potrzebujemy zapamiętać konfigurację M jako trójkę liczb.

Niech $k = |Q|$, $Q = \{q_1, \dots, q_k\}$ gdzie $q_1 = q_0$, $q_2 = q_F$. Konfiguracja C: Maszyna jest w stanie q_i, a na dziennikach jest m_1 oraz m_2, będzie oznaczona przez trójkę $t(C) = \langle i, k+1+m_1, k+1+m_2 \rangle$

<u>Def.</u> <u>Fajna szóstka</u> to szóstka liczb $\langle a_1, a_2, a_3, a_4, a_5, a_6 \rangle$ która:

- $a_1 > k$ lub
- $a_1 \leq k$; $a_2, a_3 > k$; $a_4 \leq k$; $a_5, a_6 > k$

oraz istnieją konfiguracje C oraz C' maszyny M t.że C' wynika z C w jednym kroku obliczenia M oraz że $t(C) = \langle a_1, a_2, a_3 \rangle$ i $t(C') = \langle a_4, a_5, a_6 \rangle$

Def Ciąg $a_1, \dots, a_N$ jest ~~fajny~~ gdy

- $\langle a_1, a_2, a_3 \rangle = t(C_0)$, ← początkowa konfiguracja M
- $\langle a_{n-2}, a_{n-1}, a_n \rangle = t(C_F)$, ← końcowa konfiguracja M
- $\langle a_l, a_{l+1}, \dots, a_{l+5} \rangle$ jest fajną szóstką dla $1 \leq l \leq N-5$

Obserwacja Istnieje formuła arytmetyki ψ z 6 zm. wolnymi taka, że

$(\mathbb{N}, +, \cdot, \uparrow) \models \psi(x_1, \dots, x_6)$ w.t.w. gdy $x_1, \dots, x_6$ są fajne.

$\psi(x_1, \dots, x_6)$: $x_1 > k \ \vee$

$$\bigwedge_{i \in \{1,\dots,k\}} \Big[\Big(x_1 = i \wedge \chi(x_2, i) \wedge \chi(x_3, i) \Big) \longrightarrow$$

$$\bigwedge_{j, j' \in \{0,1\}} \chi_4 = \Pi_1\big(\delta(q_i, j, j')\big) \wedge x_5 = x_2 + \Pi_2\big(\delta(q_i, j, j')\big)$$

$$\wedge \ x_6 = x_3 + \Pi_3\big(\delta(q_i, j, j')\big) \Big]$$

sprawdza czy x_2 jest w stanie opisanym przez j

gdzie Π_1, Π_2, Π_3 działają jak trzeba

Teraz chcemy formułę, która sprawdza czy ciąg jest fajny. Musimy kodować ciągi jako liczby: $a_1, \dots, a_N \longrightarrow 2^{a_1} 3^{a_2} \dots p_N^{a_N}$

"Makra":

- $\text{pierwsza}(x)$: $\forall y,z \; (yz = x \to y = 1 \vee z = 1) \wedge x > 1$
- $\text{kolejne\_pierwsze}(x,y)$: $\forall z \; (x < z < y \to \neg \text{pierwsza}(z)) \wedge \text{pierwsza}(x) \wedge \text{pierwsza}(y) \wedge x < y$
- $\text{w\_rozkładzie}(x,y,z)$:

  $\exists m \; x^y m = z \wedge \forall m \; x^{y+1} m \neq z \wedge \text{pierwsza}(x)$
- $\text{największa\_pierwsza}(x,y)$:

  $\exists s \forall z,t \; \text{pierwsza}(x) \wedge sx = y \wedge (\text{pierwsza}(z) \wedge z > x \to zt \neq y)$

Φ_M: $\exists m \; \forall x, x', y, y' \; (\text{kolejne\_pierwsze}(x,x') \wedge \text{w\_rozkładzie}(x,y,m) \wedge y' > 0 \wedge \text{w\_rozkładzie}(x',y',m) \longrightarrow y > 0)$

$$\wedge\ w\_rozkladzie(2,1,m) \wedge w\_rozkladzie(3,k+1,m)$$

$$\wedge\ w\_rozkladzie(5,k+1,m)$$

$$\wedge\ \forall p p' p''\Big(najw\_pierwsza(p'',m) \wedge kolejne\_pierwsze(p,p')$$

$$\wedge\ kolejne\_pierwsze(p',p'') \rightarrow w\_rozkladzie(p,2,m)$$

$$\wedge\ w\_rozkladzie(p',k+1,m) \wedge w\_rozkladzie(p'',k+1,m)\Big)$$

$$\wedge\ \forall_{y_1,\dots,y_6} x\Big[\bigwedge_{i=1}^{5} kolejne\_pierwsze(x_i,x_{i+1}) \wedge najw\_pierwsza(x,m)$$

$$\wedge\ x_6 \leqslant x \wedge \bigwedge_{i=1}^{6} w\_rozkladzie(x_i,y_i,m) \rightarrow \psi(y_1,\dots,y_6)\Big]$$

■

X PROBLEM HILBERTA

Czy jest algorytm rozwiązujący układ równań diofantycznych?

ODP: Nie, to jest nierozstrzygalne (trudne)

16.05.2022 PROBLEM ODPOWIEDNOŚCI POSTA (PCP)

Instancja jest skończony zbiór par słów $\{\langle l_i, r_i\rangle, i \in \{1,\dots,k\}, l_i, r_i \in \Sigma^*\}$ nad pewnym alfabetem Σ.

Czy istnieje $s \in \{1,\dots,k\}^*$ t.że $s \neq \varepsilon$ oraz $l_{s_1} l_{s_2} \dots l_{s_{|s|}} = r_{s_1} r_{s_2} \dots r_{s_{|s|}}$?

$l_7 \; l_2 \; l_3 \; \dots \; l_7$

$r_7 \; r_2 \; r_8 \; \dots \; r_7$

Tw. Problem odpowiedności Posta jest nierozstrzygalny.

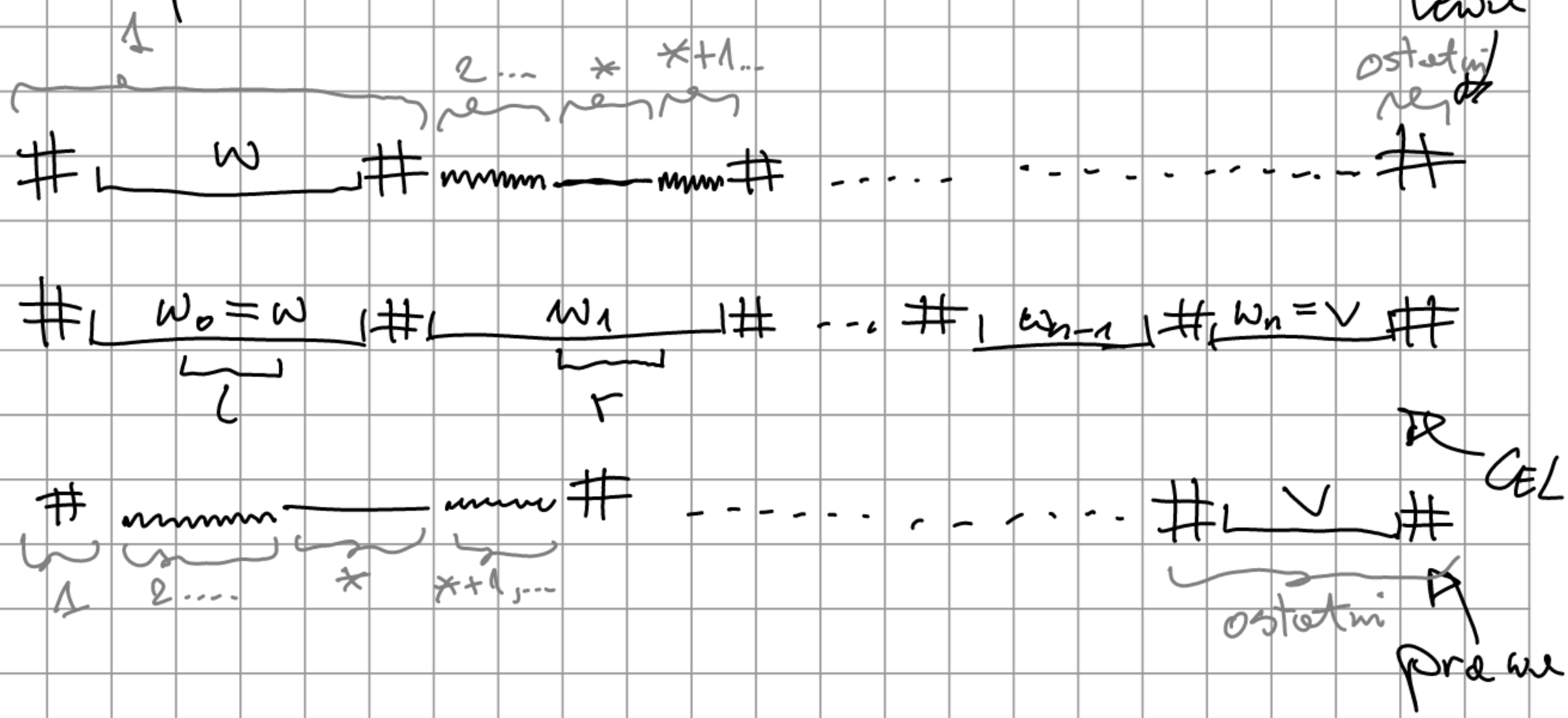
D-d. I przymierzka dowodu: Semi Thue $\leq_{rek}$ PCP.

Dana mam $\langle w, v, \Pi\rangle$. Mamy wyprodukować instancję PCP P t.że $w \xrightarrow[\Pi]{*} v \Leftrightarrow P$ ma rozwiązanie.

Niech A: alfabet Π. Wtedy $\Sigma = A \sqcup \{\#\}$.

$P = \{\langle \#w\#, \# \rangle, \langle \#, \#v\# \rangle\} \cup \Pi^R \cup \{\langle a, a\rangle : a \in \Sigma\}$

Takie P zawsze działa, ale pokażemy tylko $\Rightarrow$ w "fajny i zadowalający" sposób.



II podejście: $\Sigma = \{\#, \overline{\#}\} \cup \{a, \overline{a} : a \in A\}$.

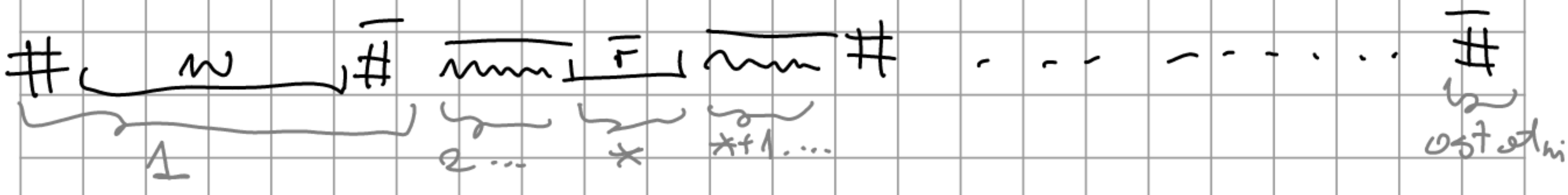
$\overline{a} \notin A$

$P = \{\langle \#, \#w\# \rangle, \langle \#v\overline{\#}, \overline{\#} \rangle\}$

$\cup \{\langle \overline{l}, r \rangle, \langle l, \overline{r} \rangle : \langle l, r \rangle \in \Pi\}$

$\cup \{\langle \#, \overline{\#} \rangle, \langle \overline{\#}, \# \rangle\} \cup \{\langle a, \overline{a} \rangle, \langle \overline{a}, a \rangle : a \in A\}$.

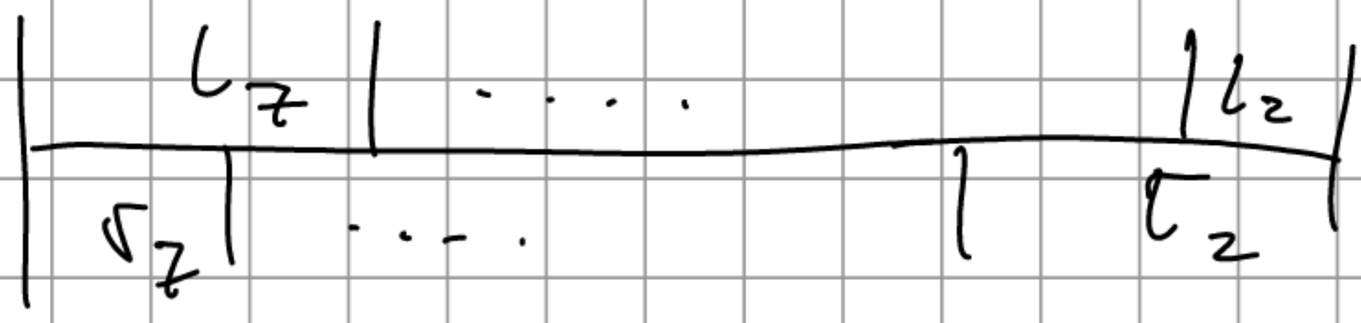
Wtedy:



(jeśli pary stosować się nie zgodzą, to mogę jakieś słowo zdublować)

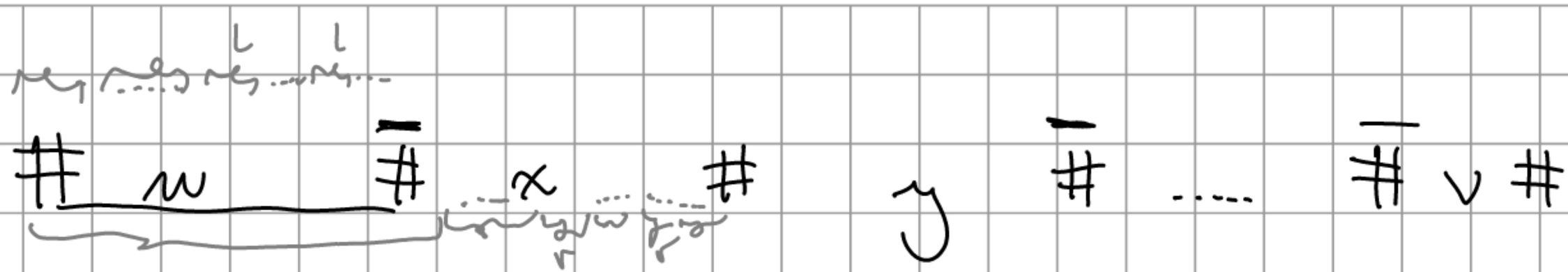
wtedy $\Rightarrow$ prosto. Teraz $\Leftarrow$:

$\{\langle l_i, r_i\rangle : i \in \{1,\dots,k\}\}$: pewna instancja PCP.



r_z jest prefiksem l $\neg$.
l_2 jest sufiksem r_2 $\neg$

Zatem musimy zacząć od $\langle \#, \#w\#\rangle$
i skończyć na $\langle \#v\#, \#\rangle$.



Lemat $x \xrightarrow{*}_{\Pi} y$

D-d. Na początek: $w \xrightarrow{*}_{\Pi} x$. Łatwe.

Dalej indukcja i elo!

z dobrymi założeniami.

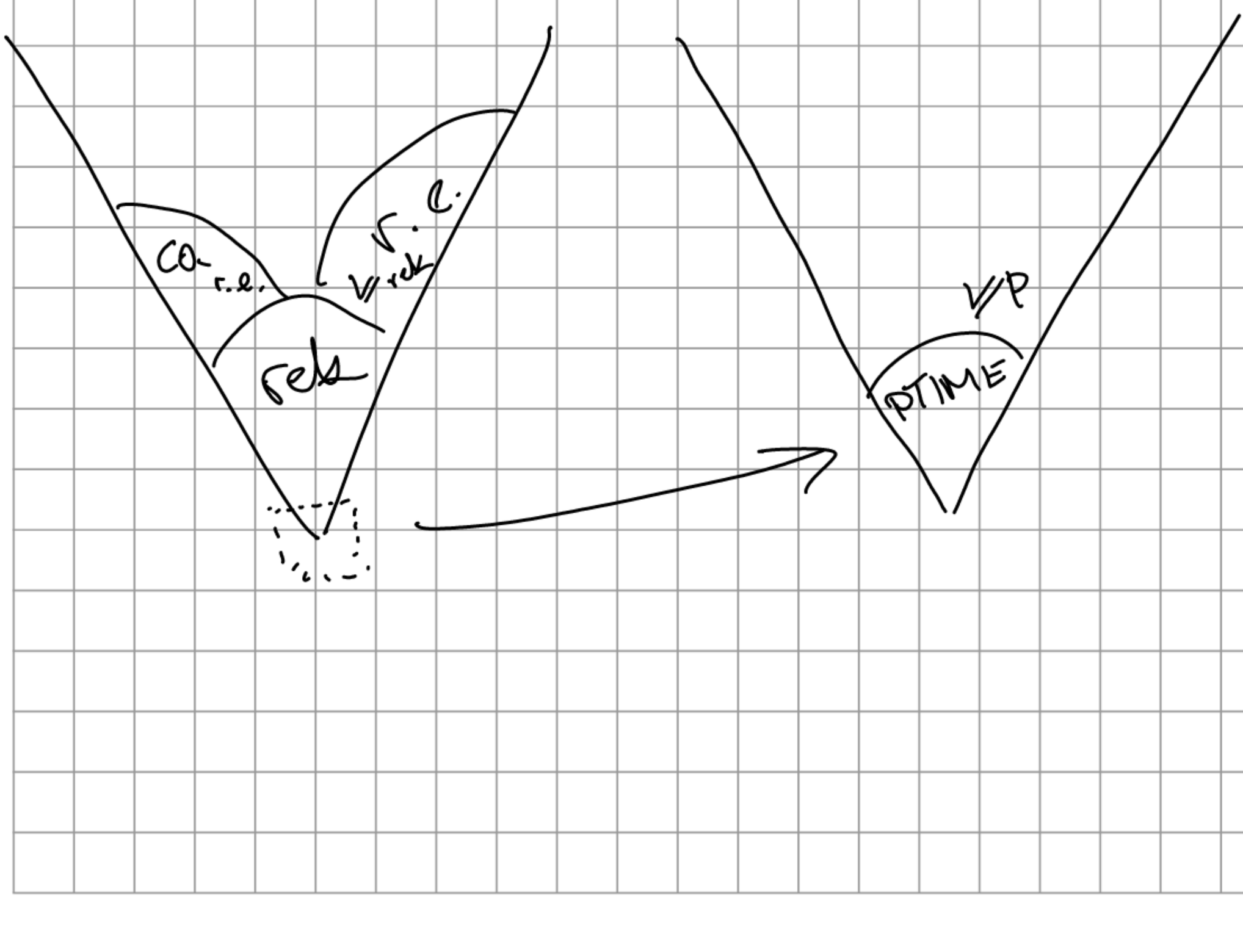
∎

Koniec tercji 2.

III CZĘŚĆ KURSU

Def. $A \subseteq \Sigma^*$ jest w **PTIME** (dla przyjaciół w "P") gdy istnieje maszyna Turinga M oraz wielomian p takie, że:

- M rozstrzyga A
- dla każdego $x \in \Sigma^*$ M uruchomiona na x zatrzymuje się po najwyżej $p(|x|)$ krokach.



Σ_A^* UI Σ_B^* UI

Def. $A \leq_p B$ (czytamy: „A jest nietrudniejsze od B w sensie redukcji wielomianowych") gdy istnieje $f: \Sigma_A^* \to \Sigma_B^*$ (zwana redukcją wielomianową) o następujących własnościach:

- istnieje wielomian p oraz MT M t.że $\forall a \in \Sigma_A^*$ $M(a)$ zatrzymuje się po $\leq p(|a|)$ krokach i zwraca wartość $f(a)$
- $\forall a \in \Sigma_A^* \left(a \in A \iff f(a) \in B\right)$

Obserwacja Jeśli $A \leq_p B$ i $B \in PTIME$, to $A \in PTIME$.

Obserwacja Jeśli $A, B \in PTIME$ i oba są nietrywialne, to $A \leq_p B$ oraz $B \leq_p A$.

Problem 3-SAT Instancja jest formuła rachunku zdań postaci 3CNF (koniunkcje klauzul o co najwyżej 3 literałach).

Pytanie czy formuła jest spełnialna.

Problem 3-kolorowanie Graf nieskierowany.

Pytanie czy da się pokolorować wierzchołki grafu tak żeby żadne dwa sąsiednie były różnych kolorów.

Tw. $3COL \leq_p 3SAT$.

D-d. Dany mi graf $G = \langle V, E \rangle$.

Mamy SZYBKO napisać formułę ψ_G w postaci 3CNF t.że G da się pokolorować $\iff \psi_G$ jest spełnialna.

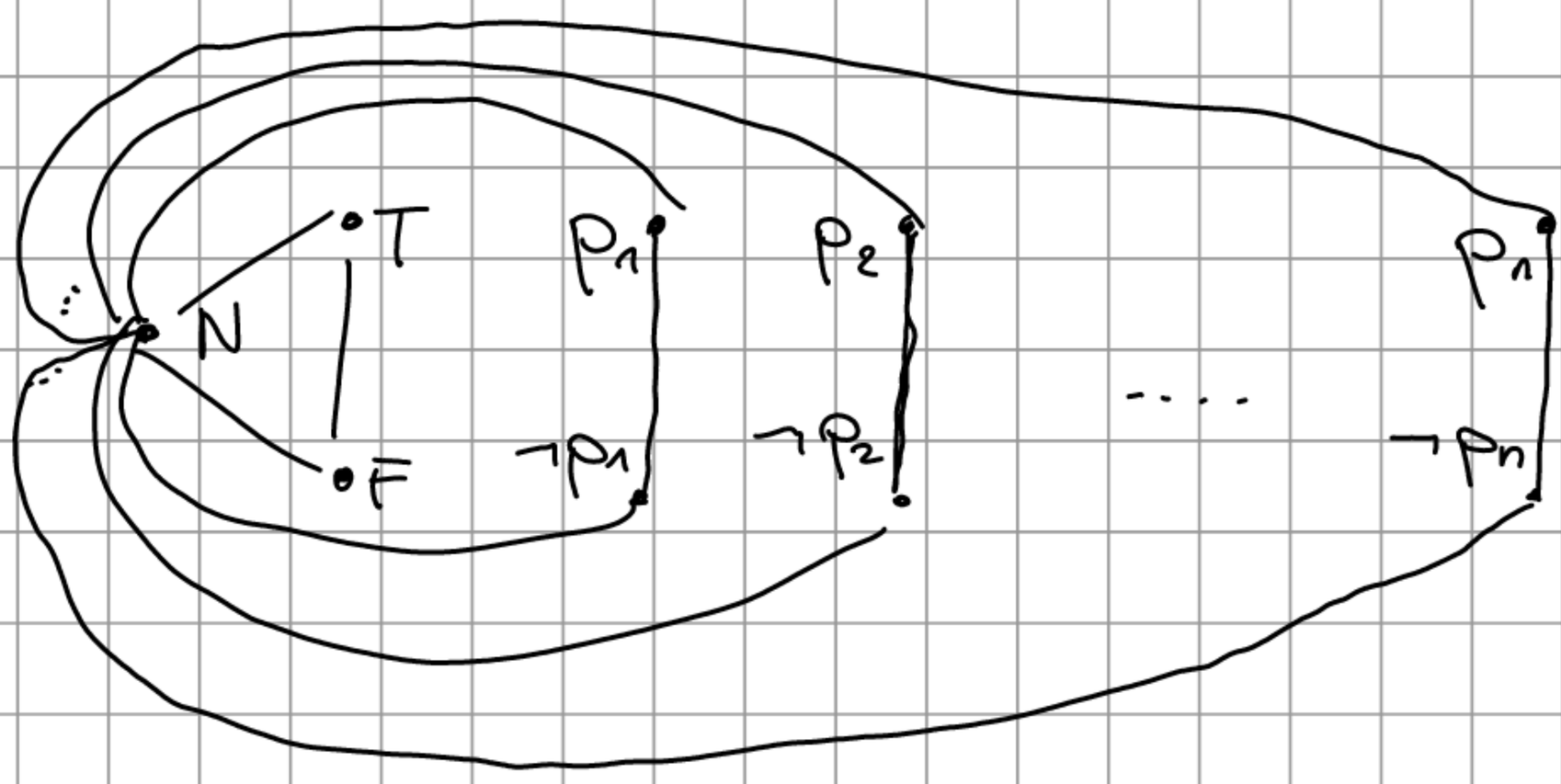
Konstrukcja ψ_G: zmiennymi będą r_u, g_u, b_u dla $u \in V$. Klauzule:

- $\forall_{u \in V} \quad (r_u \vee g_u \vee b_u)$
- $\forall_{(u,w) \in E} \quad (\neg r_u \vee \neg r_w), (\neg g_u \vee \neg g_w), (\neg b_u \vee \neg b_w)$

∎

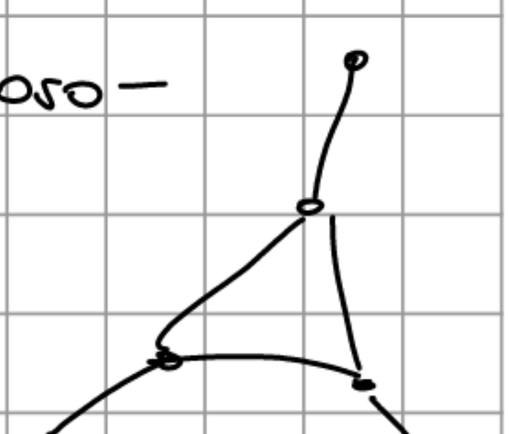
Tw $3SAT \leq_p 3COL$.

D-d. Dana nam formułę φ w postaci 3CNF o zmiennych $p_1, \dots, p_n$. Mamy szybko zbudować $G_\varphi = (V, E)$ t.że G_φ jest 3-kolorowalny $\Longleftrightarrow$ φ jest spełnialny.



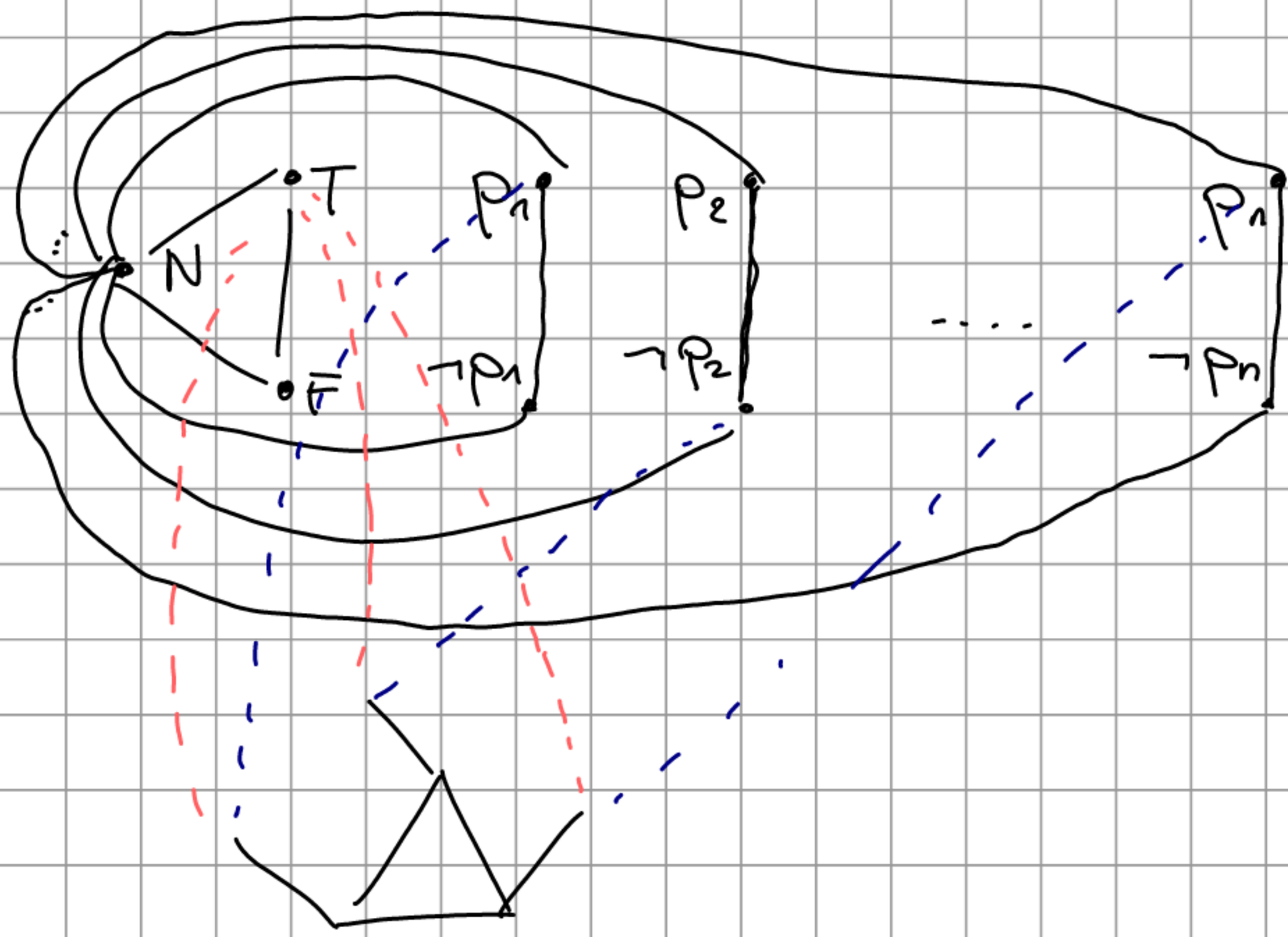
Gadżet 3 wierzchołki zewnętrzne i 3 wewnętrzne.

Obserwacja: kol: Zewn $\to \{N, F, T\}$ rozszerza się do poprawnego kolorowania całości, gdy nie jest stała.



D-d. prosty.

Podemie gadżetu dla klauzuli $p_1 \vee \neg p_2 \vee p_n$:



No i to tyle ...

23.05.2022

KLASA NP I TWIERDZENIE COOKA

Rozważamy niedeterministyczną (wielomianową) maszynę Turinga.

Teraz $\delta \subseteq (\Sigma \times Q) \times (\Sigma \times Q \times \{L, R\})$

"Wielomianowość" oznacza, że maszyna przychodzi z wielomianem p oraz gwarantuje, że jeżeli istnieje ścieżka akceptująca dla słowa w, to istnieje ścieżka akceptująca długości $p(|w|)$.

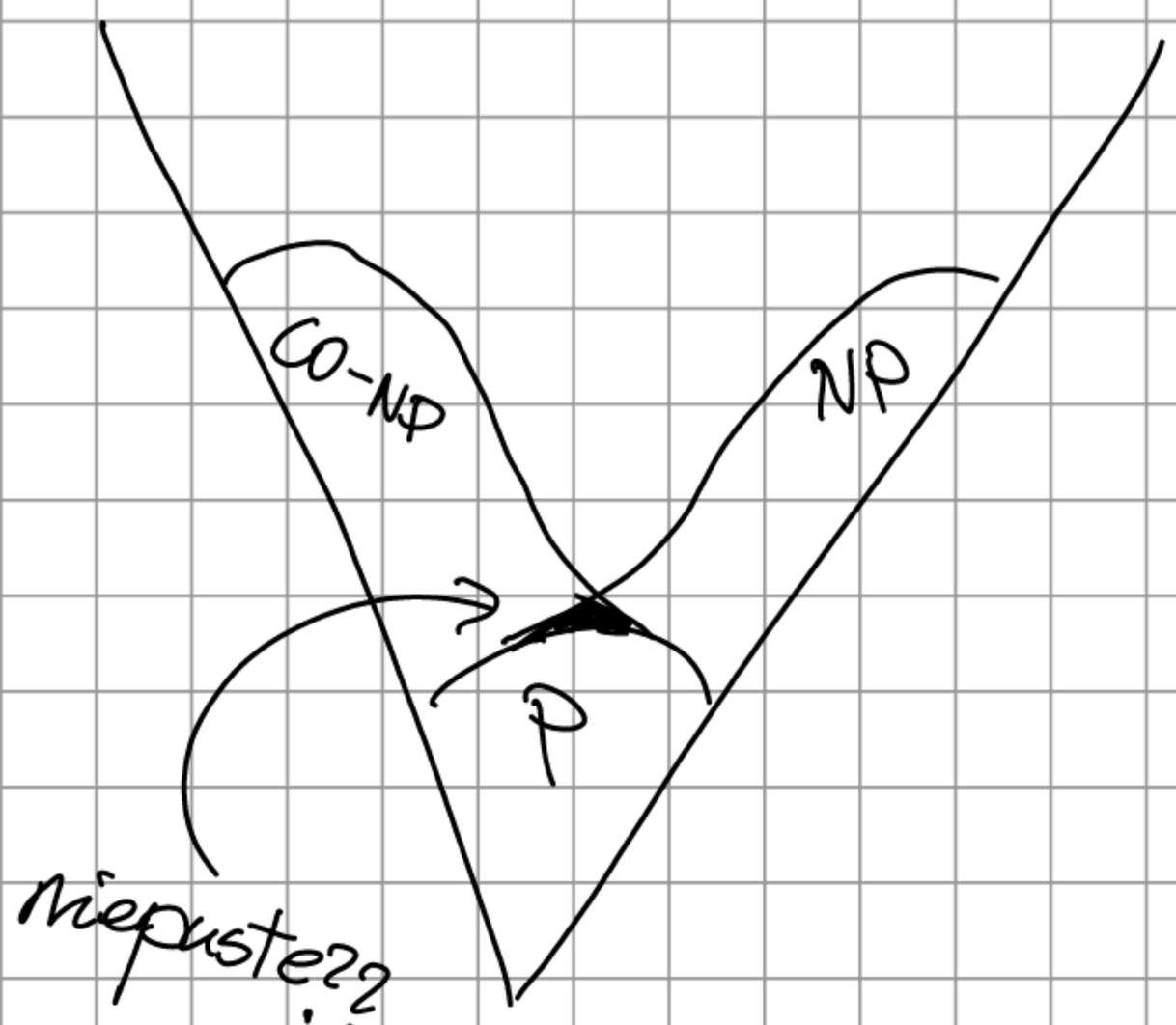
Def $A \in NP$ gdy istnieje niedeterministyczna wielomianowa maszyna Turinga rozstrzygająca A.

Przykład $3COL \in NP$. Można zgadywać kolory wierzchołków i sprawdzić, czy jest poprawne.

MICHAŁ TO PAŁA

Obserwacja $A \leq_p B \wedge B \in NP \Rightarrow A \in NP$

Przykład Sprawdzenie czy formuła zdaniowa jest tautologią jest CO-NP

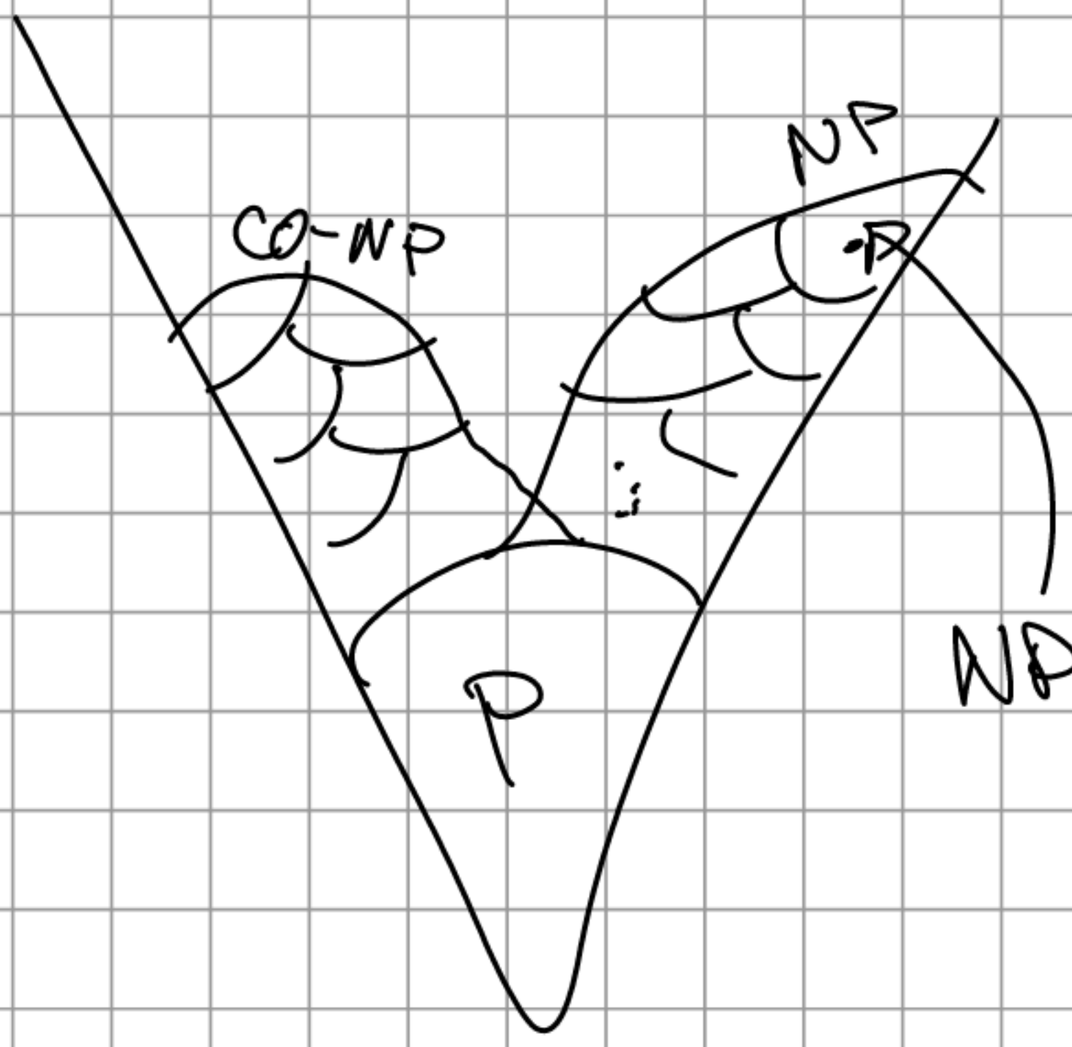


Nie wiemy, czy $P = NP$ (ludzie raczej myślą, że $NP \neq P$)

Nie wiemy, czy $NP \cap CO\text{-}NP = P$
(nawet, jeśli założymy, że $P \neq NP$)

Od teraz zakładamy, że $P \neq NP$.

Wtedy w NP są "banki":



NP - zupełne

Def. A jest NP-trudny jeżeli dla każdego $B \in NP$ $B \leq_p A$.

A jest NP-zupełny gdy $A \in NP$ oraz A jest NP-trudny.

Obserwacja "NP to zbiór P ma pierwszą oś".

Dokładniej: $\forall A \in NP \; \exists B \in P \; \exists p(x)$

$$A = \{ x : \exists y \left(|y| \leq p(|x|) \wedge [x,y] \in B \right) \}$$

Tw (Cooka) 3SAT jest NP-zupełny.

D-d. Niech $A \in NP$. Istnieją wielomiany p, q oraz deterministyczne MT M_B działające

w czasie $q(|w|)$ t.że $*(w)$

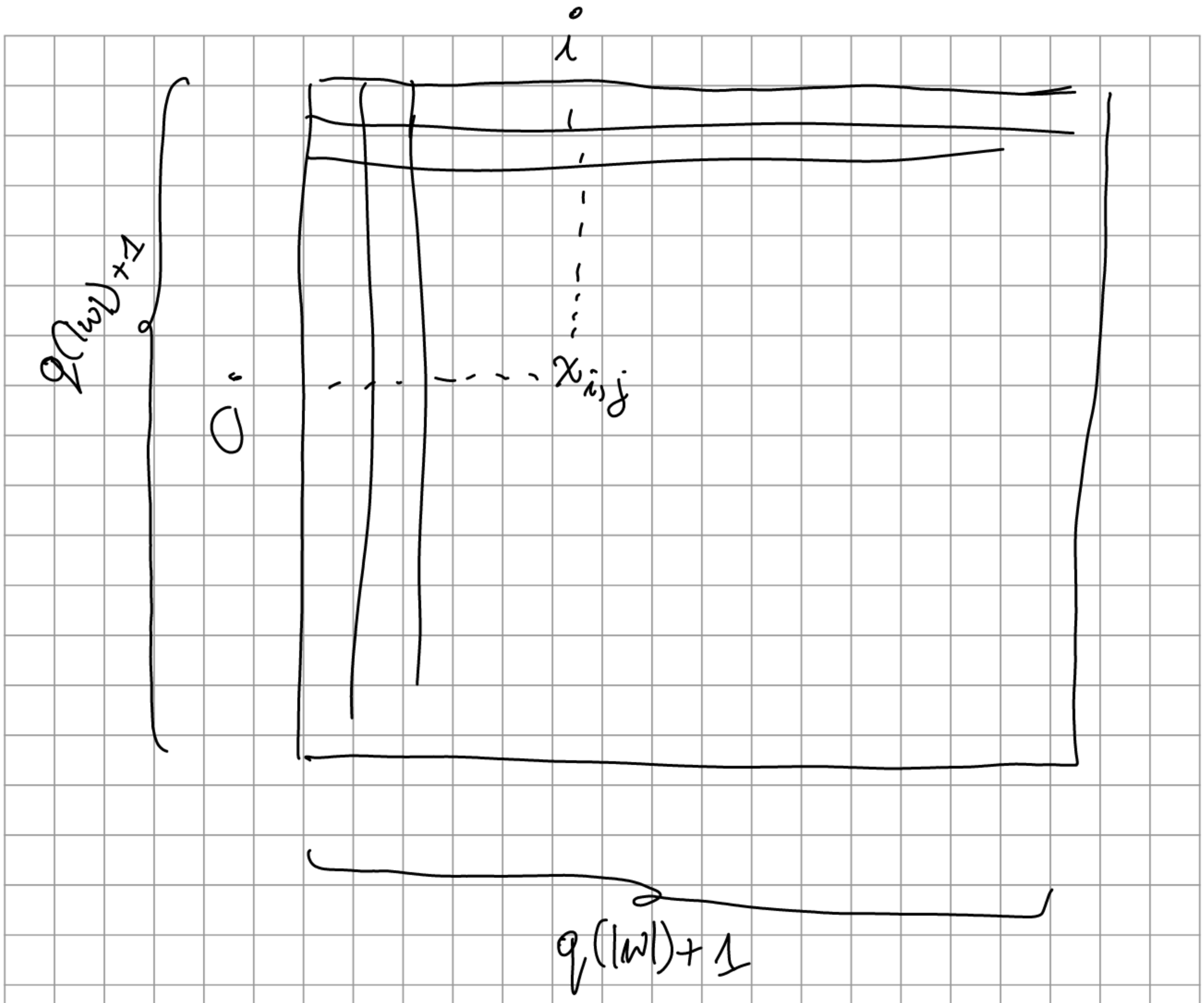
$\forall w \quad w \in A \Leftrightarrow \exists y (|y| \leq p(|w|) \wedge M_B(w,y)$ akceptuje)

Konstrukcja redukcji $A \leq_p 3SAT$:

- dając w: instancję problemu A. Mam szybko zwrócić formułę $\varphi_w \in 3CNF$ t.że

$$*(w) \Leftrightarrow \varphi_w \in 3SAT$$

- W każdym polu i,j poniższej tabelki mieszkają zmienne $a_{i,j}$, $\omega_{i,j}$, $0_{i,j}$, $1_{i,j}$, $B_{i,j}$ oraz $\forall q \in Q_{M_B}$ zmienna $q_{i,j}$ oraz $L_{i,j}$, $R_{i,j}$
- Piszemy formułę $Conf(i)$ będzie koniunkcją mnóstwa klauzul:
  - dla $\beta \neq \gamma \in \Sigma$ oraz dla $0 \leq j \leq q(|w|)$ w $Conf(i)$ będzie klauzula $(\neg B_{i,j} \vee \neg \gamma_{i,j})$
  - dla każdego $0 \leq j \leq q(|w|)$ i każdych $\beta \neq \gamma \in Q_{M_B} \cup \{L,R\}$ w $Conf(i)$ będzie $(\neg B_{i,j} \vee \neg \gamma_{i,j})$



- dla każdego j oraz $q \in Q_{M_B}$ w $Conf(i)$ będą kolejno

$L_{i,j} \to L_{i,j-1}$, $R_{i,j} \to R_{i,j+1}$, $(a \Rightarrow b \iff \neg a \vee b)$

$q_{i,j} \to R_{i,j+1}$, $q_{i,j} \to L_{i,j-1}$

- Formuła Pierwszy(i) składa się z

klauzul $q^0_{i,0}$, $\alpha_{i,0}$, dla każdego $1 \leq j \leq |w|$ $w[j]_{i,j}$, $\omega_{i,|w|+1}$

$w[j]$ – 0 albo 1

dla każdego $|w|+2 \leq j \leq |w|+1+p(|w|)$ $(O_{i,j} \vee 1_{i,j})$, dla każdego $|w|+1+p(|w|) < j \leq q(|w|)$ $B_{i,j}$

- Formuła Ostatni(i) składa się z klauzuli $q^F_{i,0}$
- Formuła Krok(i) składa się z klauzul:
  - dla każdego j, każdego $p \in \Sigma$ mamy klauzulę $L_{i,j} \wedge p_{i,j} \rightarrow p_{i+1,j}$, $R_{i,j} \wedge p_{i,j} \rightarrow p_{i+1,j}$
  - dla każdej instrukcji w δ_{M_D} postaci $\langle q, p, q', p', L' \rangle$ i dla każdego j $q_{i,j} \wedge p_{i,j} \rightarrow p'_{i+1,j} \wedge q'_{i+1,j-1}$

- analogicznie dla instrukcji w prawo

- Wtedy $\varphi_w = \bigwedge_i Conf(i) \wedge Pierwszy(0) \wedge Ostatni(q(|w|))$
$\wedge \bigwedge_i Krok(i)$

30.05.2022

O NIEROZSTRZYGALNOŚCI RACHUNKU PREDYKATÓW

$$\underbrace{\forall x \exists y \left(E(x,y) \wedge \dots \wedge \dots \right)}$$

formuła logiki I rzędu

Pytania:

a) czy jest spełnialna?

b) czy jest tautologią?

1) dopuszczamy dowolne struktury

2) tylko skończone formuły

2a) $\rightsquigarrow$ R.E.

2b) $\rightsquigarrow$ CO-R.E.

Przykład

$$\left(\exists x \, \forall z \neg E(z,x)\right) \wedge \left(\forall x \exists y \; E(x,y)\right) \wedge$$
$$\left(\forall x,y,y' \; E(y,x) \wedge E(y',x) \rightarrow y = y'\right)$$

↑ Ma model nieskończony, nie ma skończonego.

TW. Problem 1b jest nierozstrzygalny

D-d. Pokażemy $MM \leq_{rek}$ TAUT-FOL

problem stopu maszyny Turinga

Dają nam maszynę M, ona ma stany Q_M, w tym q_0, q_F i δ_M.

Napiszemy formułę φ_M t.że φ_M jest tautologią $\Leftrightarrow$ M się zatrzymuje.

Będę miał 3 predykaty binarne E, P, D oraz tyle predykatów unarnych ile jest stanów w Q_M i może jeszcze jeden: Z.

φ_M będzie koniunkcją następujących formuł:

- $\forall x \exists y \left[\left(\bigvee_{R \in Q_M \setminus \{q_F\}} R(x) \right) \rightarrow E(x,y) \right]$
- $\exists x \exists z \quad Q_0(x) \wedge P(x,z) \wedge D(x,z) \wedge Z(z)$
- $\exists z \forall t \quad Z(z) \wedge \neg P(z,t) \wedge \neg D(z,t)$
  $\wedge \left(Z(t) \rightarrow t = z \right)$

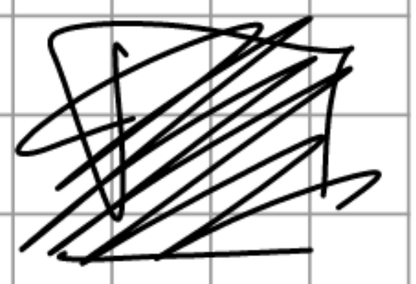
- Dla każdej instrukcji z $\mathcal{I}_M$ dejemy formaTę. Np. jeżeli $\mathcal{I}_M$ ma instrukcję postaci $\langle Q, Z, NZ; Q', +1, -1\rangle$, to dodajemy formułę:

$$\forall x,y,t,t',t'' \exists r \Big[\big(Q(x) \wedge P(x,t) \wedge D(x,t') \wedge Z(t) \wedge \neg Z(t') \wedge D(t',t'') \big) \longrightarrow \big(Q'(y) \wedge D(y,t'') \wedge P(y,r) \wedge P(r,t) \big) \Big]$$

Lemat Jeśli po n krokach obliczań M jest w konfiguracji $\langle Q, m, m' \rangle$ i struktura S spełnia φ_M, to istnieje w niej wierzchołek x t.że $Q(x)$ i z x jest P-ścieżką do Z dł m i D-ścieżką długości m'.

Teraz konstruujemy ψ_M.

$\varphi_M \longrightarrow \exists x\, Q_f(x)$

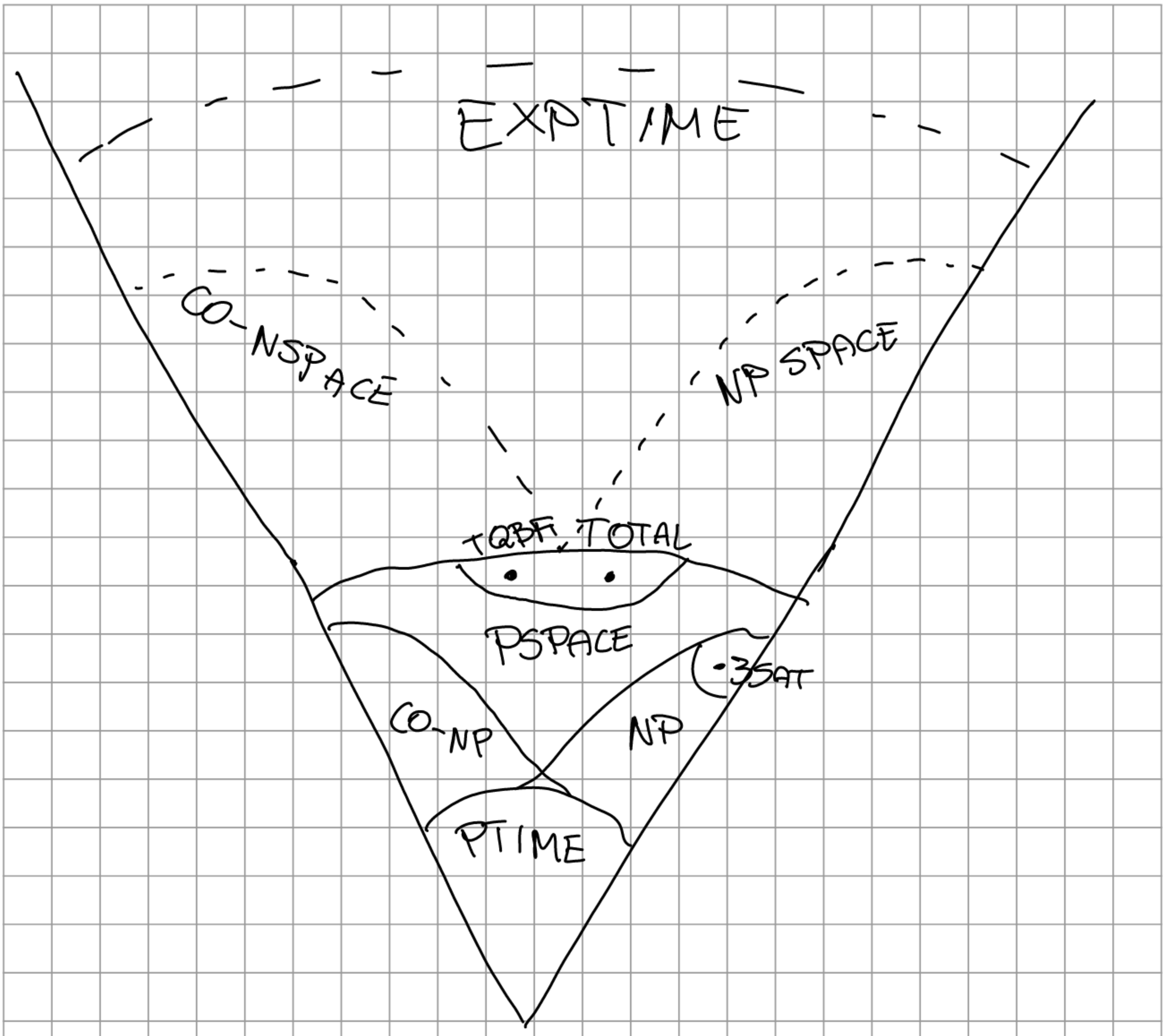


Klasa PSPACE

Def $A \in PSPACE$, gdy A daje się rozstrzygać przy pomocy Maszyny Turinga z wielomianową pamięcią.

Obserwacja
- $PTIME \subseteq PSPACE$
- $NP \subseteq PSPACE$
- $CO\text{-}NP \subseteq PSPACE$ (PSPACE jest zamknięty na dopełnienia)

Nie umiemy rozwiązać NIC. Nawet nie umiemy powiedzieć, czy $PTIME \neq PSPACE$.



Tw. PSPACE = NPSPACE

Dajmy mam niedeterministyczną MT M i wielomian $p(x)$ t. że dla każdego wejścia w maszyna M na każdej "ścieżce obliczeń" budzi $\leq p(|w|)$.

Mamy skonstruować deterministyczny algorytm zużywający wielomianowo dużo pamięci rozstrzygający czy M akceptuje dane wejście.

(BSO M zanim się zatrzyma "zeruje" taśmę i wraca nad a)

M ma zbiór stanów Q t.że $|Q| = k$ i 5 symboli taśmowych. Jesteśmy teraz tym wielomianowym algorytmem. Dają nam słowo w. Wyobrażamy sobie graf konfiguracji M dla w.

Chcemy się dowiedzieć czy istnieje ścieżka z C_0 do C_F.

W tym grafie jest $5^{p(|w|)} \cdot |Q| \cdot p(|w|) \leq$

$\leq 8^{p(|w|)} = 2^{3p(|w|)}$

$5^{p(|w|)}$ – ilość stanów

$|Q|$ – gdzie jest głowica

$p(|w|)$

Procedura$_i$ odpowie czy dla danych dwóch wierzchołków S, t grafu istnieje ścieżka z S do t dł. $\leq 2^i$

Procedura$_0$ – sprawdza czy jest krawędź

Procedura$_{i+1}$ – dla każdego x

– sprawdź czy Procedura$_i(S, x)$,

jeśli tak, to posprzątaj i sprawdź

Procedura$_i(x, t)$

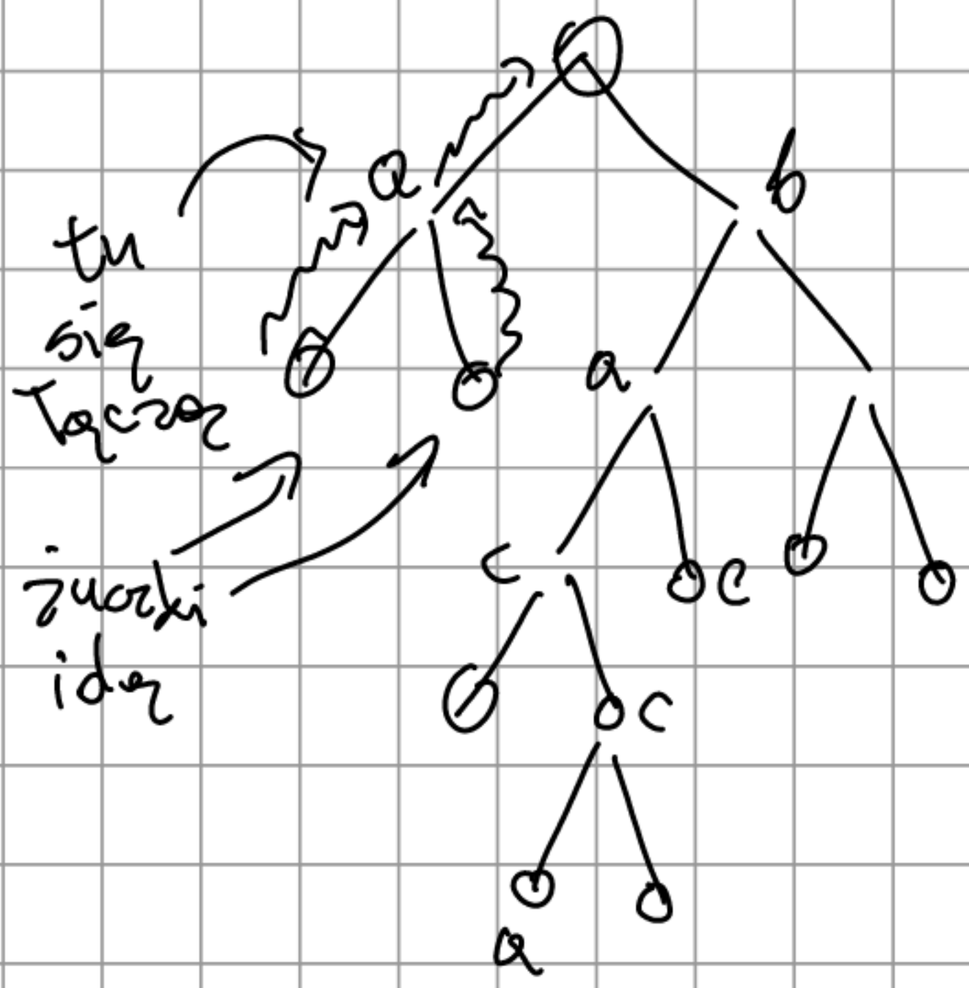
Teraz uruchamiam Procedura$_{3p(|w|)}(C_0, C_F)$

i zwróć wynik.

Pamięć ulega $\sim 10 p^2 (|w|)$

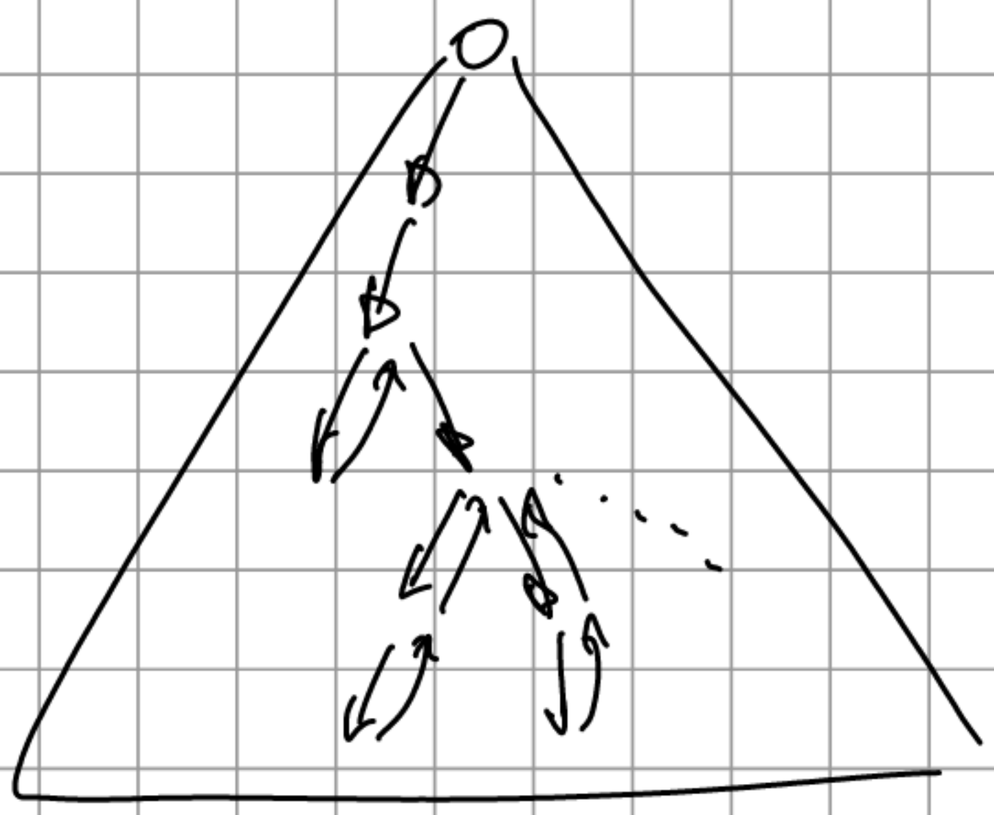
6.06.2022 Cofamy się na chwilę do I części kursu.

REGULARNE JĘZYKI DRZEW



$\delta : Q \times Q \times \Sigma \longrightarrow Q$

TREE WALKING AUTOMATA



Pokazano, że TWA są słabsze niż RJD

Ale! TWA umie zwartościować formułę.

W każdym węźle wewnętrznym mamy koniunkcję, alternatywę lub negację, a w liściach 0 lub 1.

$\underbrace{\exists p \forall q \; p \leftrightarrow q}$ (kwantyfikujemy po $\{\bot, \top\}$)

quantified boolean formula (QBF)

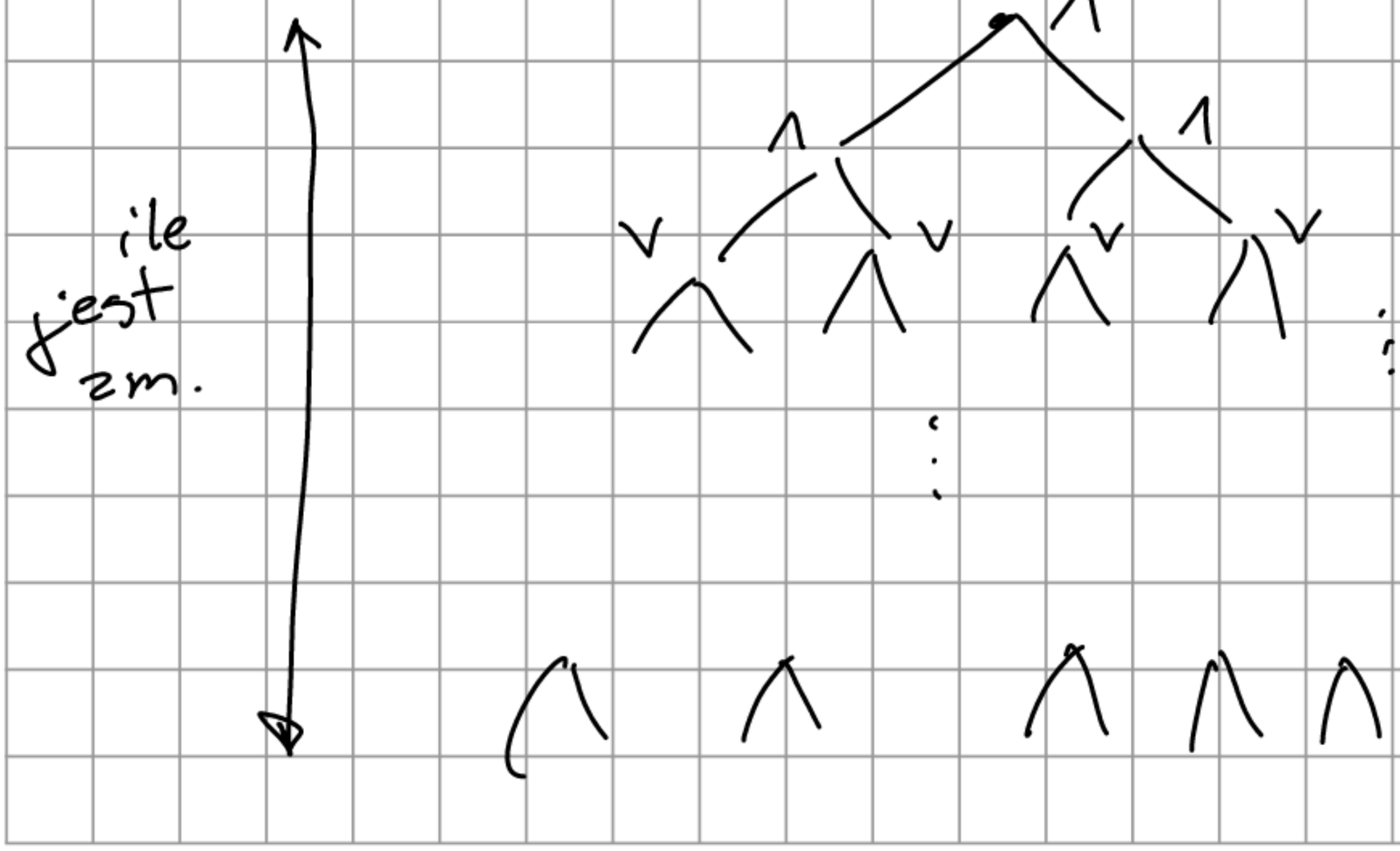
Tw. TQBF jest problemem PSPACE-zupełnym.

↑ true

D-d. Po pierwsze pokażemy, że TQBF $\in$ PSPACE

Rysujemy drzewo:

$\exists \exists \forall \exists \forall$



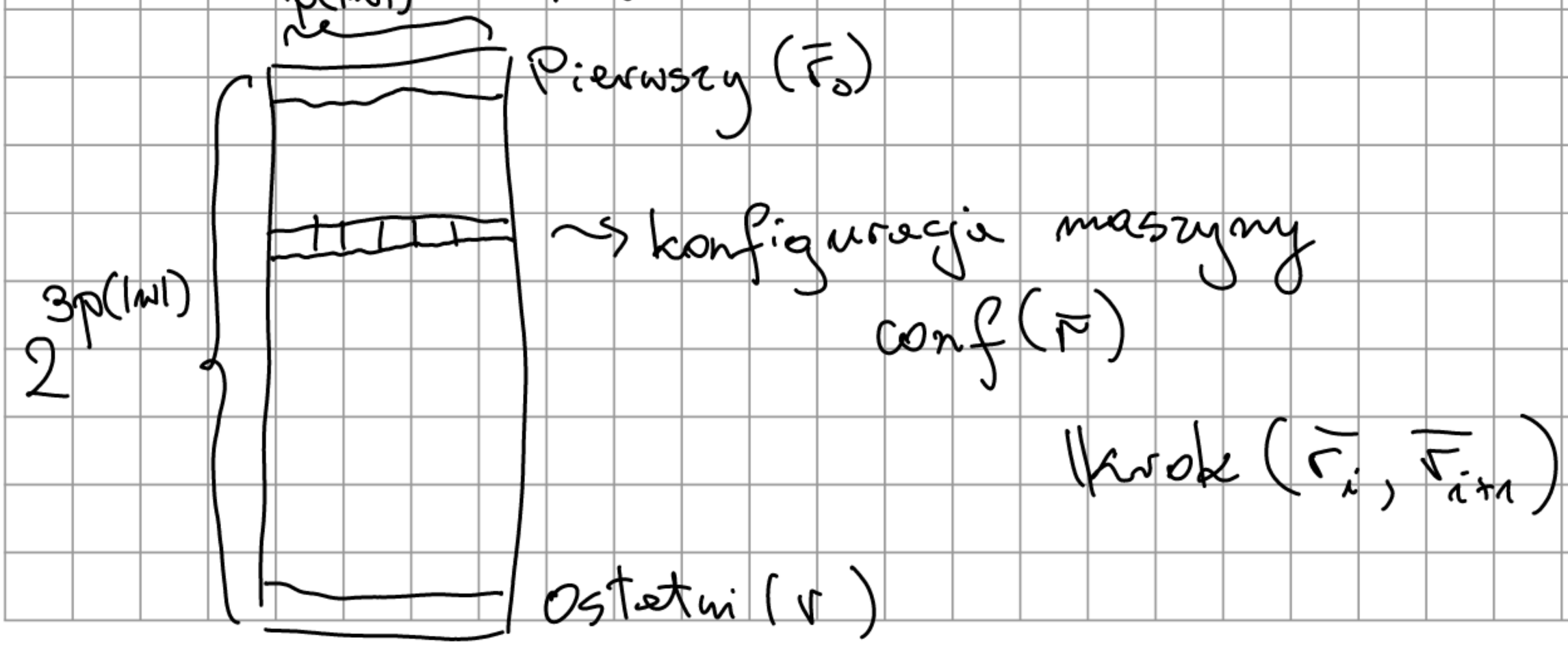
Znowu robimy żuczka, ale teraz ma wielomianowo wiele pamięci.

Teraz pokażemy, że dla każdego $B \in PSPACE$ $B \leq_p TQBF$.

Weźmy $B \in PSPACE$ (czyli jest MT M_B i wielomian t.że M_B rozstrzyga B działając w przestrzeni $p(\text{wielkość wejścia})$.

Od teraz jesteśmy redukcją. Dajemy mam w: instancję B, mamy SZYBKO wyprodukować formułę ψ_w w postaci QBF t.że

$M_B(w)$ akceptuje $\iff$ ψ_w jest prawdziwe



Problem: nie możemy mieć zmiennych na każdą komórkę!

Najbardziej by nam pasowało napisać:

$$\begin{aligned}
&Pierwszy(r_0) \wedge \\
&Ostatni(\bar{r}_{2^{3p(|w|)}}) \wedge \\
&Krok(\bar{r}_0, \bar{r}_1) \wedge \\
&Krok(\bar{r}_1, \bar{r}_2) \wedge \\
&\vdots
\end{aligned}$$

$$Konf(\bar{t}) \wedge$$

Napiszemy $Krok_{i+1}(\bar{r}, \bar{s}) = \exists \bar{t} \Big(Krok_i(\bar{r}, \bar{t}) \wedge Krok_i(\bar{t}, \bar{s}) \Big)$. $Krok_{i+1}(\bar{r}, \bar{s})$: z $\bar{r}$ do $\bar{s}$ da się dojść w 2^{i+1} kroków.

Ostateczna formuła:

$$Pierwszy(\bar{r}) \wedge Ostatni(\bar{s}) \wedge Krok_{3p(|w|)}(\bar{r}, \bar{s})$$

Ale nadal coś jest nie tak z Krok: jest za duża. Naprawa:

$$Krok_{in}(\bar{r},\bar{s}) = \exists \bar{t}\ \forall \bar{x},\bar{y}\ \Big[(\bar{x}=\bar{r} \wedge \bar{y}=\bar{t}) \vee (\bar{x}=\bar{t} \vee \bar{y}=\bar{s}) \longrightarrow Krok_i(\bar{x},\bar{y})\Big]$$

∎

GRA W GEOGRAFIĘ

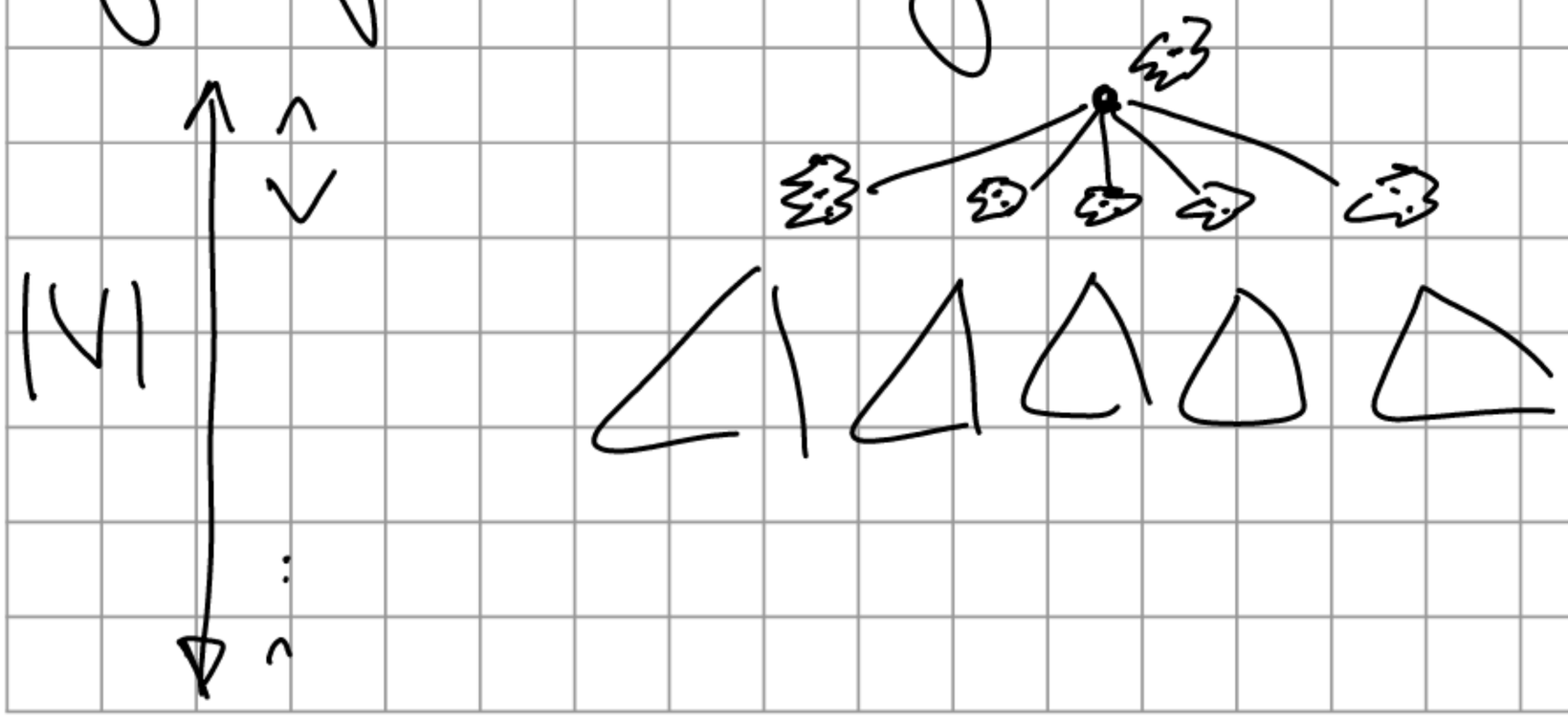
Instancja jest graf skierowany (V, E, v_0), gdzie v_0 jest wierzchołkiem początkowym.

Dwóch graczy: Staś i Nel. Mają pieczątkę, która stoi w v_0. Następnie Staś bierze pieczątkę i przesuwa po jakiejś krawędzi do jeszcze nieodwiedzonego wierzchołka, potem Nel. Przegrywa ten, kto nie może zrobić ruchu.

Pytają czy Staś ma strategię wygrywającą.

Tw. TQBF $\leq_p$ Gra w geografię

D-d. Na początku pokażemy, że ta gra jest w ogóle w PSPACE



Reszta to samo co w TWA.

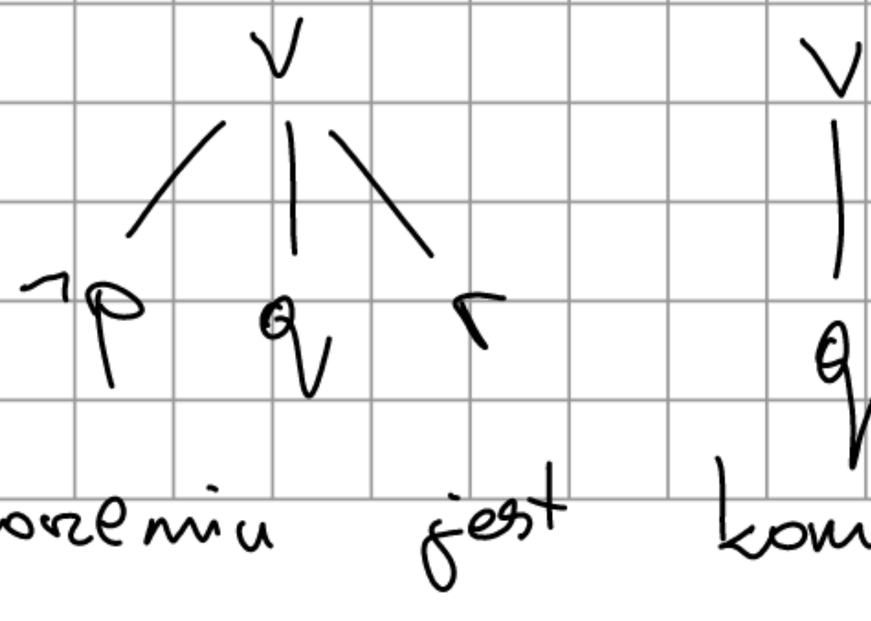
Teraz dajemy nam formułę φ w postaci QBF.

BSO φ ma prefiks dwuntyfikatorowy

$\exists p_1 \forall q_1 \exists p_2 \forall q_2 \dots \exists p_k \forall q_k$ >

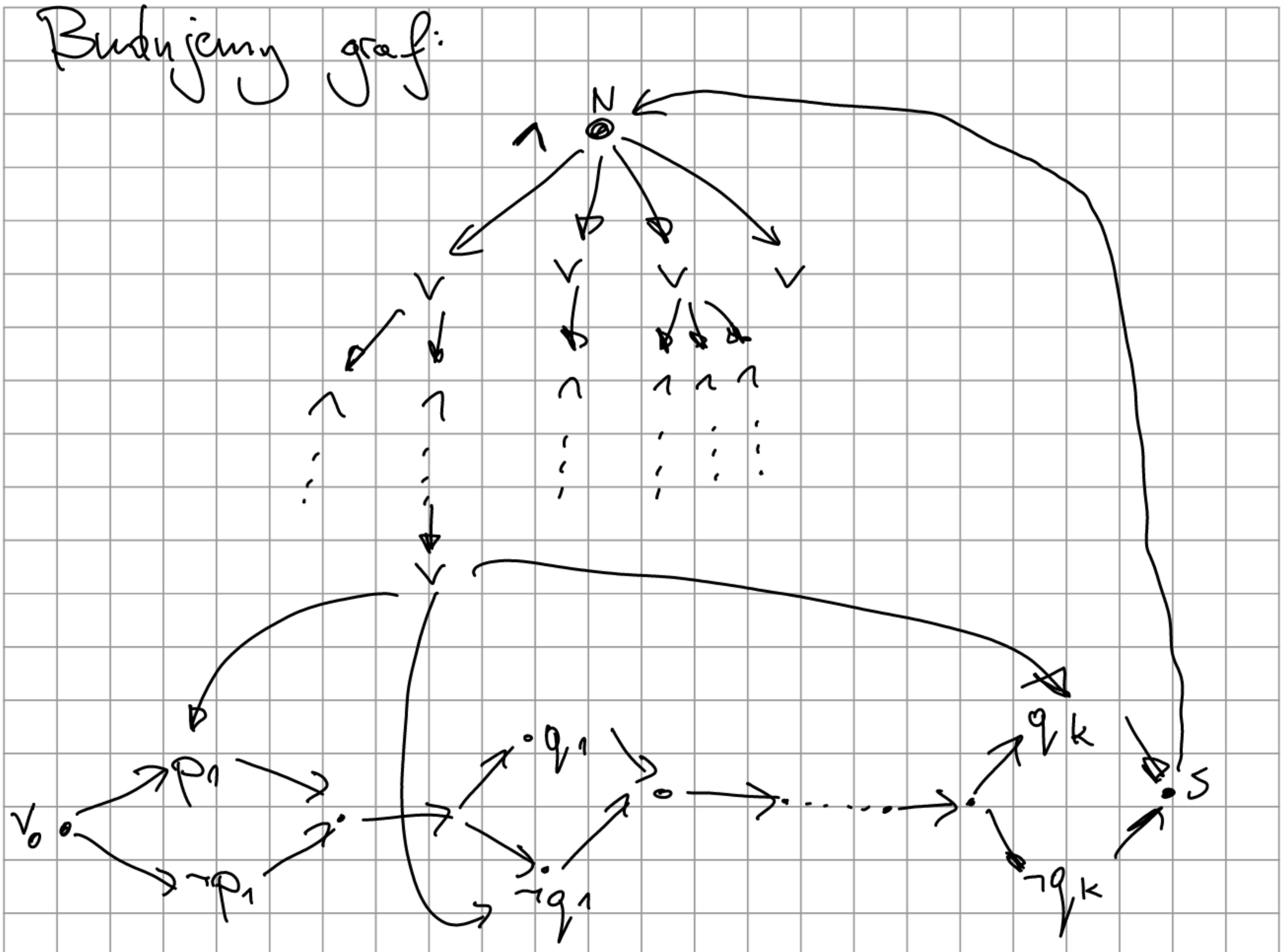
negacja jest tylko w liściach.

Każdy literał ma bezpośrednio nad sobą alternatywę



a w korzeniu jest koniunkcja

Budujemy graf:



Nieostemplowane literały to te wybrane.
Nela chce pokazać, że formuła nie jest prawdziwa.

13.06.2022

Rozważamy wyrażenia regularne z

- dodatkowym symbolem podnoszenia do kwadratu: e^2. Wtedy $|e^2| = |e| + 1$,
  $L(e^2) = L(e) \cdot L(e)$.
- dodatkowym symbolem dopełnienia: e^c.
  Wtedy $L(e^c) = \Sigma^* \setminus L(e)$,
  $|e^c| = |e| + 1$.

PROBLEM TOTALNOŚCI REx

Okazuje się, że problem totalności:

1) dla RE jest PSPACE-zupełny
2) dla RE(2) jest EXPSPACE-zupełny
3) dla RE(^{c}) jest nieelementarny.

Rozważamy rodziny funkcji:

- EXP: $f = \mathcal{O}(2^g)$, g: wielomian
- k-EXP: $f = \mathcal{O}(2^g)$, $g \in (k-1)$-EXP

Klasa funkcji elementarnych: $\bigcup_{k>0}^{\infty} k\text{-EXPTIME}$

$=$

$\bigcup_{k>0}^{\infty} k\text{-EXPSPACE}$

bo $k\text{-EXPSPACE} \subsetneq (k+1)\text{-EXPTIME}$ $\longrightarrow$

Rodzina $\mathcal{F}$. Komowiczny problem $\mathcal{F}$-TIME (SPACE)-zupełny

Dajemy MT M, $w \in \Sigma^*$, $p \in \mathcal{F}$ i pytamy czy M akceptuje w w owsie (pamięci) $p(|w|)$.

D-d.(1) Dajemy MT M, $w \in \Sigma^*$, $p \in POLY$.

Najpierw zmieniamy (redukujemy) M w M', ε, p.

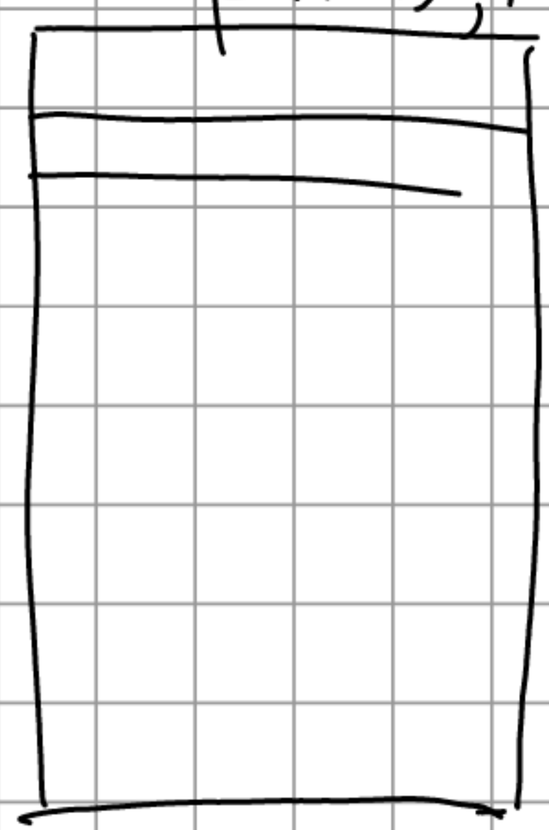
To zmieniam. $p(|w|)$, $|w| = n$

w problem kafel-

kowania dla

tej maszyny:

$K, E, n, K^4, p(|w|)$

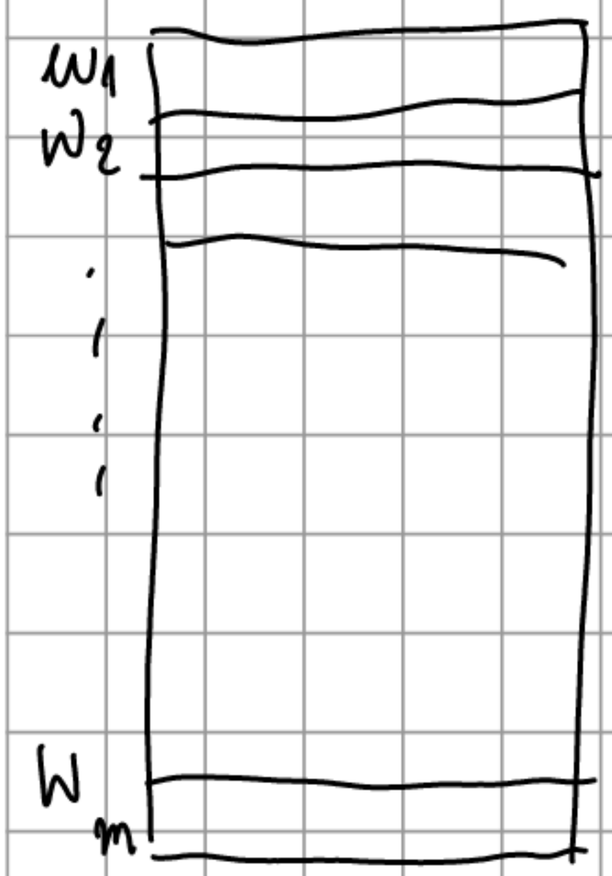


Kaf 1) każdy kafelek 2×2 należy do E

Kaf 2) Pierwszy i ostatni rząd są postaci $N \subset B^* N$

Kaf 3) wymiary są $m \times p(|w|)$

$(k^2)^*$ $\quad \Sigma = k^2 \cup \{\#\}$



$\cup$

$$\binom{w_1}{w_2}\#\binom{w_2}{w_3}\#\ldots\#\binom{w_{m-1}}{w_m}$$

kodowanie kolorowanie

Napiszemy e dopasowujące się do stów w, które NIE kodują poprawnych kolorowań

K1) $\Sigma^*\binom{a_1\ b_1}{a_2\ b_2}\Sigma^*$, gdzie $\binom{a_1\ b_1}{a_2\ b_2} \notin E$

K2) Początek i koniec NIE są postaci NCB^*N

K3) $\Sigma^*\binom{a_1}{a_2}\Sigma^{p(n)}\binom{b_1}{b_2}\Sigma^*$, $a_2 \neq b_1$

$\Sigma^*\binom{a_1}{a_2}\Sigma^{p(n)}\#\Sigma^*$ i symetrycznie

K4) $(\neg\#)^{p(n)+1}\Sigma^* + (\neg\#)^{<p(n)-1}\#\Sigma^*$

e : alternatywa wyrażeń K1-4

Wniosek 1) Dla danej MT M, słowa $w \in \Sigma^*$, $p(x) \in POLY$ istnieje $e_{M,w}$ t.że $\overline{L}(e_{M,w}) = \{H^p_{M,w}\}$

Problem EXPTIME-zupełności możemy rozwiązać tak samo, ale w K3 i K4 wyrażamy $\sum^{p(n)} = \sum^{2^{q(n)}} = (\underbrace{\sum 2^{2^{\cdots^{2}}}}_{q(n)})$, gdzie $p(n) = 2^{q(n)}$, $q \in$ POLY

Dla problemu z dopełnieniem pokażemy, że problem RE(c) jest k-EKPSPACE trudny dla $k \in \mathbb{N}$.

Pokażemy, że jest 1-EXPSPACE-trudny, 2-EXPSPACE-trudny, a reszta przez indukcję.

Wybacz czytelniku, ale nie chcem tego sensownie zamotować.

20.06.2022

Pokażemy, że PSPACE $\subsetneq$ EXPSPACE.

Wskażemy $L \in EXPSPACE \setminus PSPACE$

Przypomnienie Każde $w \in \Sigma_{0,1}^*$ możemy czytać jako kod pewnej maszyny Turinga, którą będziemy nazywać M_w.

Przez g oznaczamy pewną niemalejącą funkcję $\mathbb{N} \to \mathbb{N}$ BARDZO POWOLI dążącą do ∞, np. $\log^*$ (ale $\log \log \log n$ chyba wystarczy).

Przez p_n oznaczamy wielomian $g(n)\left(1 + x + x^2 + \ldots + x^{g(n)}\right)$

Definiujemy maszynę ML:

- wczytaj n
- zaznacz obszar składający się z $p_{|w|}(|w|)$ komórek taśmy.

- w tym obszarze symuluj działanie $M_w(w)$.
  - Jeśli w trakcie działania M_w chce opuścić przydzielony obszar, to zaakceptuj.
  - Jeśli $M_w(w)$ akceptuje, to odrzuć
  - Jeśli $M_w(w)$ odrzuci lub się zapętla, to zaakceptuj

Pokażemy, że $L \notin PSPACE$

Zał. nie wprost, że $L \in PSPACE$. Wtedy istnieje M i wielomian q t.że M rozstrzyga L i działa w przestrzeni ograniczonej przez q.

Niech $n \in \mathbb{N}$ t.że $\forall m \in \mathbb{N}\ p_n(m) \geq q(m)$.

Niech w będzie t.że $|w| \geq n$ i $M_w = M$

Czy $w \in L$?

- gdyby $w \in L$, to $ML(w)$ akceptuje,

ponadto $p_n(|w|) > q_1(|w|)$, więc $M_w(w)$ nie opuści obszaru, więc musi odrzucić lub się zapętlać ↯

$w \in L \Leftrightarrow ML(w)$ akceptuje $\Leftrightarrow M_w(w)$ odrzuca $\Leftrightarrow w \notin L$

Teraz pokażemy, że $L \in EXPSPACE$.

Pokażemy, że ML działa w przestrzeni EXPSPACE. No ale to wystarczy pokazać, że $c \cdot p_{|w|}(|w|) \leq 2^{|w|}$

$$g(n) \cdot n^{g(n)} \leq 2^n \quad | \log$$

$$\log(g(n)) + g(n) \cdot \log(n)$$

$$\leq g(n)(\log(n) + 1)$$

$$\leq \log^2(n) + \log(n) \leq n$$

No, coś tam, TUTAJ...