

# Języki, automaty i obliczenia

Wykład 12: Gramatyki i inne modele równoważne maszynom Turinga.  
Wstęp do złożoności obliczeniowej

Sławomir Lasota

**Uniwersytet Warszawski**

20 maja 2015

- 1 Gramatyki
- 2 Języki kontekstowe
- 3 Automaty z licznikami
- 4 Złożoność czasowa i pamięciowa

typ	języki	automaty	gramatyki
typ 0	rekurencyjnie przeliczalne	maszyny Turinga	?
typ 1	kontekstowe	?	?
typ 2	bezkontekstowe	automaty ze stosem	bezkontekstowe
typ 3	regularne	automaty skończone	regularne

$$\mathcal{G} = (A, N, S, \alpha)$$

- $A$  to skończony zbiór symboli końcowych (*terminalnych*)
- $N$  to skończony zbiór symboli niekończących (*nieterminalnych*),  $A \cap N = \emptyset$
- $S \in N$  to symbol początkowy
- $\alpha \subseteq (A \cup N)^+ \times (A \cup N)^*$  to skończony zbiór reguł przepisywania (*produkcji*)

### Notacja

Reguły  $(v, v') \in \alpha$  będziemy zapisywać  $v \rightarrow_{\mathcal{G}} v'$  albo  $v \rightarrow v'$ .

Reguły rozszerzamy do relacji  $\rightarrow_{\mathcal{G}} \subseteq (A \cup N)^* \times (A \cup N)^*$ :

$$w \rightarrow_{\mathcal{G}} w' \iff \exists (v \rightarrow_{\mathcal{G}} v') \in \alpha, \exists u, t \in (A \cup N)^*, w = uvt, v = uv't$$

i domykamy zwrotno-tranzytywnie:  $w \twoheadrightarrow_{\mathcal{G}} w'$  wtw. gdy istnieje ciąg

$$w = v_0 \rightarrow_{\mathcal{G}} v_1 \rightarrow_{\mathcal{G}} \dots \rightarrow_{\mathcal{G}} v_n = w', \quad n \geq 0$$

zwany *wyprowadzeniem* (wywodem)  $w'$  z  $w$  (w gramatyce  $\mathcal{G}$ ).

Język generowany przez gramatykę:

$$L(\mathcal{G}) = \{w \in A^* : S \xrightarrow{\text{G}} w\}$$

## Przykład

Gramatyka  $\mathcal{G}$ :

$$A = \{a, b, c\}$$

$$N = \{S, X, B\}$$

$$S \rightarrow \varepsilon$$

$$S \rightarrow X$$

$$X \rightarrow aBXc$$

$$X \rightarrow abc$$

$$Ba \rightarrow aB$$

$$Bb \rightarrow bb$$

$$L(\mathcal{G}) = ?$$

## Twierdzenie

Języki częściowo rozstrzygalne to języki generowane przez gramatyki.

Dowód:

- gramatyka  $\mathcal{G} \rightsquigarrow$  maszyna Turinga  $\mathcal{M}$ : maszyna symuluje gramatykę „wstecz”

$$v \rightsquigarrow_{\mathcal{G}} w \iff qw \xrightarrow{*}_{\mathcal{M}} qv,$$

zaczyna w konfiguracji  $qw$ , akceptuje „w konfiguracji  $qS$ ”. Więc  $L(\mathcal{G}) = L(\mathcal{M})$ .

- maszyna Turinga z taśmą jednostronnie nieograniczoną  $\rightsquigarrow$  gramatyka:

$$\text{Faza 1: } S \rightsquigarrow_{\mathcal{G}} \$ w [q_f, a] v \mathbb{B} \quad w, v \in (T - \{\mathbb{B}\})^*, a \in T - \{\mathbb{B}\}$$

$$\begin{array}{ll} \text{Faza 2:} & b [q', c] \longrightarrow [q, a] c & \text{jeśli } (q, a, q', b, \rightarrow) \in \delta \\ & [q', c] b \longrightarrow c [q, a] & \text{jeśli } (q, a, q', b, \leftarrow) \in \delta \\ & b [q', \mathbb{B}] \longrightarrow [q, \mathbb{B}] & \text{jeśli } (q, \mathbb{B}, q', b, \rightarrow) \in \delta \\ & [q', c] b \mathbb{B} \longrightarrow c [q, \mathbb{B}] & \text{jeśli } (q, \mathbb{B}, q', b, \leftarrow) \in \delta \\ & \$ [q_0, a] \longrightarrow \$ a \\ & a \mathbb{B} \longrightarrow a \end{array}$$

## Problem słów

Dane: gramatyk  $\mathcal{G}$  i dwa słowa  $v, w$

Wynik: czy  $v \xrightarrow{\text{***}}_{\mathcal{G}} w$  ?

## Wniosek

*Problem słów jest nierozstrzygalny.*

*Dowód:*

Redukujemy problem stopu do problemu słów:

funkcja obliczalna:	maszyna $\mathcal{M}$ , słowo $w$	$\mapsto$	$\mathcal{G}$ i słowa $S, w$
poprawność:	$w \in L(\mathcal{M})$	$\iff$	$S \xrightarrow{\text{***}}_{\mathcal{G}} w$

## Pytanie

Czy problem słów jest częściowo rozstrzygalny?

- 1 Gramatyki
- 2 Języki kontekstowe
- 3 Automaty z licznikami
- 4 Złożoność czasowa i pamięciowa



typ	języki	automaty	gramatyki
typ 0	rekurencyjnie przeliczalne	maszyny Turinga	?
typ 1	kontekstowe	?	?
typ 2	bezkontekstowe	automaty ze stosem	bezkontekstowe
typ 3	regularne	automaty skończone	regularne

Gramatyka  $\mathcal{G} = (A, N, S, \alpha)$  jest *kontekstowa* jeśli produkcje są postaci:

- $S \rightarrow \varepsilon$
- $uXv \rightarrow uwv$       $X \in N, \quad u, v, w \in (A \cup N)^*, \quad w \neq \varepsilon, \quad S \notin w$

## Pytanie

Gramatyka  $\mathcal{G}$ :

$$A = \{a, b, c\}$$

$$N = \{S, X, B\}$$

$$S \rightarrow \varepsilon$$

$$S \rightarrow X$$

$$X \rightarrow aBXc$$

$$X \rightarrow abc$$

$$Ba \rightarrow aB$$

$$Bb \rightarrow bb$$

$$L(\mathcal{G}) = \{a^n b^n c^n : n \in \mathbb{N}\}$$

Jak przerobić tę gramatykę na gramatykę kontekstową?

Gramatyka  $\mathcal{G} = (A, N, S, \alpha)$  jest *kontekstowa* jeśli produkcje są postaci:

- $S \rightarrow \varepsilon$
- $uXv \rightarrow uwv$       $X \in N, \quad u, v, w \in (A \cup N)^*, \quad w \neq \varepsilon, \quad S \notin w$

## Pytanie

Gramatyka  $\mathcal{G}$ :

$$\begin{array}{llll} A = \{a, b, c\} & S \rightarrow \varepsilon & X \rightarrow aBXc & Ba \rightarrow aB \\ N = \{S, X, B\} & S \rightarrow X & X \rightarrow abc & Bb \rightarrow bb \end{array}$$

$$L(\mathcal{G}) = \{a^n b^n c^n : n \in \mathbb{N}\}$$

Jak przerobić tę gramatykę na gramatykę kontekstową?

## Odpowiedź

Pomysł jest następujący:

$$\begin{array}{ll} Ba \rightarrow B\bar{a} & \bar{B}\bar{a} \rightarrow \bar{B}B \\ B\bar{a} \rightarrow \bar{B}\bar{a} & \bar{B}B \rightarrow aB \end{array}$$

Gramatyka  $\mathcal{G} = (A, N, S, \alpha)$  jest *kontekstowa* jeśli produkcje są postaci:

- $S \rightarrow \varepsilon$
- $uXv \rightarrow uwv$      $X \in N, \quad u, v, w \in (A \cup N)^*, \quad w \neq \varepsilon, \quad S \notin w$

Gramatyka  $\mathcal{G} = (A, N, S, \alpha)$  jest *monotoniczna* jeśli produkcje są postaci:

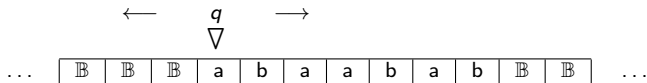
- $S \rightarrow \varepsilon$
- $v \rightarrow w$      $|v| \leq |w|, \quad S \notin w$

## Twierdzenie

*Gramatyki kontekstowe i monotoniczne rozpoznają te same języki.*

*Dowód:*

j.w.



Maszyna Turinga jest *liniowo ograniczona* jeśli nigdy nie pisze na pozycji taśmy zajętej przez symbol  $\mathbb{B}$ . Czyli zabraniamy przejść postaci:

$$(q, \mathbb{B}, q', a, x) \quad a \neq \mathbb{B}$$

## Twierdzenie

*Języki kontekstowe to języki rozpoznawane przez niedet. maszyny liniowo ograniczone.*

*Dowód:*

- gramatyka  $\mathcal{G} \rightsquigarrow$  maszyna Turinga  $\mathcal{M}$ : maszyna symuluje gramatykę „wstecz”

$$v \rightsquigarrow_{\mathcal{G}} w \iff qw \xrightarrow{*}_{\mathcal{M}} qv,$$

zaczyna w konfiguracji  $qw$ , akceptuje „w konfiguracji  $qS$ ”. Więc  $L(\mathcal{G}) = L(\mathcal{M})$ .

- maszyna Turinga z taśmą jednostronnie nieograniczoną  $\rightsquigarrow$  gramatyka:

$$\text{Faza 1: } S \rightsquigarrow_{\mathcal{G}} \$w [q_f, a] v \mathbb{B} \quad w, v \in (T - \{\mathbb{B}\})^*, a \in T - \{\mathbb{B}\}$$

$$\text{Faza 2: } b [q', c] \longrightarrow [q, a] c \quad \text{jeśli } (q, a, q', b, \rightarrow) \in \delta$$

$$[q', c] b \longrightarrow c [q, a] \quad \text{jeśli } (q, a, q', b, \leftarrow) \in \delta$$

$$b [q', \mathbb{B}] \longrightarrow [q, \mathbb{B}] \quad \text{jeśli } (q, \mathbb{B}, q', b, \rightarrow) \in \delta$$

$$[q', c] b \mathbb{B} \longrightarrow c [q, \mathbb{B}] \quad \text{jeśli } (q, \mathbb{B}, q', b, \leftarrow) \in \delta$$

$$\$ [q_0, a] \longrightarrow \$ a$$

$$a \mathbb{B} \longrightarrow a$$

## Fakt

*Problem słów jest rozstrzygalny dla gramatyk monotonicznych.*

*Więc problem stopu jest rozstrzygalny dla liniowo ograniczonych maszyn Turinga.*

## Twierdzenie

*Problem (nie)pustości jest nierozstrzygalny dla języków kontekstowych.*

*Dowód:*

Dla ustalonej instancji PCP, zbiór rozwiązań jest językiem kontekstowym.

funkcja obliczalna:	instancja PCP $x$	$\mapsto$	$\mathcal{G}_x$
poprawność:	$x$ ma rozwiązanie	$\iff$	$L(\mathcal{G}_x) \neq \emptyset$

## Pytanie

Czy problem (nie)pustości jest częściowo rozstrzygalny?

typ	języki	automaty	gramatyki
typ 0	rekurencyjnie przeliczalne	maszyny Turinga	typu 0
typ 1	kontekstowe	maszyny liniowo ogr.	kontekstowe/monotoniczne
typ 2	bezkontekstowe	automaty ze stosem	bezkontekstowe
typ 3	regularne	automaty skończone	regularne



- 1 Gramatyki
- 2 Języki kontekstowe
- 3 Automaty z licznikami**
- 4 Złożoność czasowa i pamięciowa

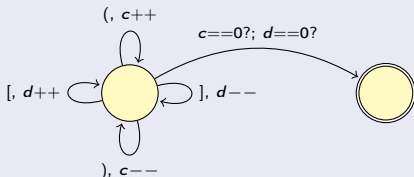
$$\mathcal{A} = (A, Q, I, F, C, \delta)$$

- $A$  – alfabet
- $Q$  – skończony zbiór stanów
- $I \subseteq Q$  – stany początkowe
- $F \subseteq Q$  – stany akceptujące
- $C$  – skończony zbiór liczników
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \{c++, c--, c==0? : c \in C\} \times Q$  – relacja przejścia

## Przykład

$$A = \{ (, ), [, ] \}$$

$$C = \{ c, d \}$$



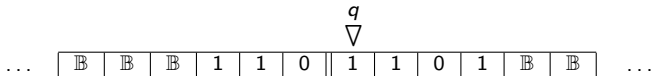
$$L(\mathcal{A}) = ?$$

## Twierdzenie

*Języki częściowo rozstrzygalne to języki rozpoznawane przez automaty z licznikami.*

*Dowód (idea):*

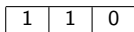
Maszyna Turinga  $\rightsquigarrow$  automat z 3 licznikami:



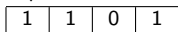
$q$



$q$



$$6 = 4 + 2 + 0$$



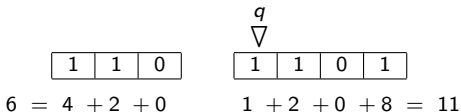
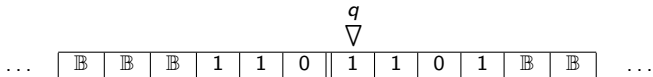
$$1 + 2 + 0 + 8 = 11$$

## Twierdzenie

*Języki częściowo rozstrzygalne to języki rozpoznawane przez automaty z licznikami.*

*Dowód (idea):*

Maszyna Turinga  $\rightsquigarrow$  automat z 3 licznikami:

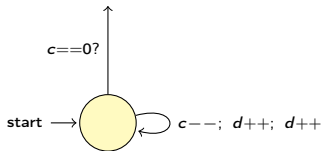
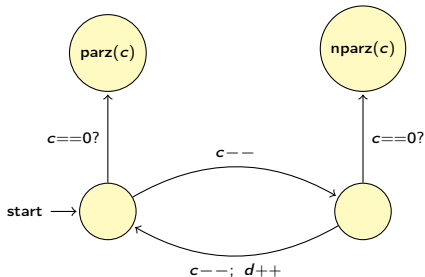


Potrzebne operacje na licznikach:

$$\text{parz}(c)? \quad \text{nparz}(c)? \quad c = c \gg 1 \quad c = c \ll 1$$

Potrzebne operacje na licznikach:

parz(c)?    nparz(c)?     $c = c >> 1$      $c = c << 1$



## Twierdzenie

*Maszyny z 2 licznikami potrafią symulować maszyny z 3 licznikami.*

*Dowód:*

?

## „Najprostszy” problem nierozstrzygalny

Dane:  $\mathcal{A}$  – automat deterministyczny z 2 licznikami  $c_1, c_2$

Wynik: czy  $\mathcal{A}$  zatrzyma się, jeśli rozpocznie w konfiguracji  $(q_0, c_1 = 0, c_2 = 0)$ ?

## Pytanie

Kiedy automat z licznikami jest deterministyczny?

## Twierdzenie

*Maszyny z 2 licznikami potrafią symulować maszyny z 3 licznikami.*

*Dowód:*

?

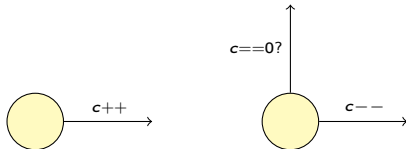
## „Najprostszy” problem nierozstrzygalny

Dane:  $\mathcal{A}$  – automat deterministyczny z 2 licznikami  $c_1, c_2$

Wynik: czy  $\mathcal{A}$  zatrzyma się, jeśli rozpocznie w konfiguracji  $(q_0, c_1 = 0, c_2 = 0)$ ?

## Pytanie

Kiedy automat z licznikami jest deterministyczny?



- 1 Gramatyki
- 2 Języki kontekstowe
- 3 Automaty z licznikami
- 4 Złożoność czasowa i pamięciowa**



$n$  – długość słowa wejściowego. Rozważamy funkcje  $f : \mathbb{N} \rightarrow \mathbb{N}$  t.ż.  $f(n) \geq \log n$ .

klasa języków	języki rozpoznawane przez
$\text{DTIME}(f(n))$	deterministyczną maszynę w czasie $f(n)$
$\text{NTIME}(f(n))$	niedeterministyczną maszynę w czasie $f(n)$
$\text{DSPACE}(f(n))$	deterministyczną maszynę w pamięci $f(n)$
$\text{NSPACE}(f(n))$	niedeterministyczną maszynę w pamięci $f(n)$

### Pytanie

Czy wystarczy ograniczyć się do obliczalnych funkcji  $f$ ?

### Twierdzenie

*Wszystkie klasy są właściwymi podklasami języków rozstrzygalnych.*

Dowód:

$$L_p = \{w : w \notin L_{\leq f}(\mathcal{M}(w))\}$$

## Twierdzenie

- $DTIME(f(n)) \subseteq NTIME(f(n)) \subseteq DSPACE(f(n)) \subseteq NSPACE(f(n))$
- $NSPACE(f(n)) \subseteq \bigcup_c DTIME(2^{c \cdot f(n)}) = DTIME(2^{O(f(n))})$  **założenie!**

...

$$EXPSPACE = \bigcup_c DSPACE(2^{n^c}) = \bigcup_c NSPACE(2^{n^c})$$

$$NEXPTIME = \bigcup_c NTIME(2^{n^c})$$

$$EXPTIME = \bigcup_c DTIME(2^{n^c})$$

$$PSPACE = \bigcup_c DSPACE(n^c) = \bigcup_c NSPACE(n^c)$$

$$NP = NPTIME = \bigcup_c NTIME(n^c)$$

$$P = PTIME = \bigcup_c DTIME(n^c)$$

$$NL = NLOGSPACE = NSPACE(\log(n))$$

$$L = LOGSPACE = DSPACE(\log(n))$$

typ	języki	automaty	gramatyki	klasa złożoności
typ 0	rekurencyjnie przeliczalne	maszyny Turinga	dowolne	
typ 1	kontekstowe	maszyny liniowo ograniczone	kontekstowe/ monotoniczne	$\text{NSPACE}(n)$
typ 2	bezkontekstowe	automaty ze stosem	bezkontekstowe	$\subseteq \text{PTIME}$
typ 3	regularne	automaty skończone	regularne	$\text{NSPACE}(1)/\text{DSPACE}(1)$

Czas wielomianowy i pamięć wielomianowa:

$$\begin{aligned} PSPACE &= \bigcup_c DSPACE(n^c) &= \bigcup_c NSPACE(n^c) \\ NP &= NPTIME &= \bigcup_c NTIME(n^c) \\ P &= PTIME &= \bigcup_c DTIME(n^c) \end{aligned}$$