Języki, automaty i obliczenia

Wykład 10: Obliczalność

Sławomir Lasota

Uniwersytet Warszawski

15 maja 2024

Plan

1. Teza Churcha-Turinga
2. Problem stopu
3. Rozstrzygalność
4. Obliczalność

Języki jako problemy decyzyjne

Języki będziemy utożsamiać z *problemami decyzyjnymi*.

Cykl Hamiltona

Dane:	graf skierowany G
Wynik:	rozstrzygnąć, czy G ma cykl Hamiltona ?

Graf można opisać słowem nad $A = \{0, 1, \#\}$, np. 110#101#001

- Dane wejściowe nazywamy *instancją* problemu decyzyjnego.
- Dane wejściowe można opisać („kodować”) na różne sposoby słowem nad A.
- A co ze słowami, które nie opisują żadnej instancji?

Teza Churcha-Turinga

języki rozpoznawane przez maszyny Turinga	=	problemy decyzyjne, dla których istnieje efektywny algorytm, przy założeniu nieograniczonych zasobów

albo:

maszyny Turinga	=	komputery

Wątpliwości:

- nieskończoność taśmy?
- alfabet taśmowy większy niż rozmiar dysku?
- liczba stanów większa niż rozmiar dysku?
- niedeterminizm?

Modele obliczeń równoważne maszynom Turinga

- gramatyki typu 0
- automaty z wieloma stosami
- automaty z licznikami
- automaty z kolejką
- maszyny RAM (ang. *random access machines*) albo maszyny rejestrowe
- while-programy
- rachunek λ
- ...

Plan

Alfabet binarny

Bez u.o. możemy ograniczyć się do maszyn, w których

$$A = \{0, 1\} \qquad\qquad T = \{0, 1, \mathbb{B}\}.$$

Faktycznie, np. słowa nad alfabetem

$$T = \{a, b, c, d, \mathbb{B}\}$$

można zapisać jako słowa nad $\{0, 1, \mathbb{B}\}$,

$$a\,b\,c\,a\,\mathbb{B}\,d \;=\; 1000 \quad 0100 \quad 0010 \quad 1000 \quad \mathbb{B}\mathbb{B}\mathbb{B}\mathbb{B} \quad 0001$$

Problem stopu

Dane:	maszyna Turinga $\mathcal{M}$ nad alfabetem A i słowo $w \in A^*$
Wynik:	czy $w \in L(\mathcal{M})$?

Maszynę $\mathcal{M} = (A, Q, q_0, q_{\mathsf{tak}}, T, \mathbb{B}, \delta)$ można opisać słowem $w_{\mathcal{M}} \in \{0, 1, \#\}^*$, np.

$$000000\#001000\#000001\#\#000100\#001\#010000\#100\#001\#\# \ldots$$

albo słowem $w_{\mathcal{M}} \in \{0, 1\}^*$, np.

$$000000\ 1\ 001000\ \ 000001\ \ 000100\ \ 001\ \ 010000\ \ 100\ \ 001\ \ldots$$

Program może stanowić dane wejściowe...

Język uniwersalny i maszyna uniwersalna

Język uniwersalny (problem stopu):

$$\{\ w_{\mathcal{M}}\,\$\,w \ \ :\ w \in L(\mathcal{M})\ \} \ \subseteq\ \{0,1,\$\}^*$$

$$\{\ (\mathcal{M}, w) \ \ :\ w \in L(\mathcal{M})\ \}$$

Twierdzenie

Problem stopu jest częściowo rozstrzygalny.

Dowód: Uniwersalna maszyna Turinga:

$000100\,\$$

$000000\#001000\#000001\#\#000100\#001\#010000\#100\#001\#\#\ldots\$$

$001010101010\overline{1}111101011$

Poza częściową rozstrzygalność

W pewnym mieście żyje fryzjer, który strzyże tych mieszkańców, którzy nie strzygą się sami. Czy ten fryzjer strzyże się sam?

Twierdzenie

Język „przekątniowy”:

$$L_p = \{\, w_{\mathcal{M}} \;:\; w_{\mathcal{M}} \notin L(\mathcal{M}) \,\}$$

$$L_p = \{\, \mathcal{M} \;:\; \mathcal{M} \notin L(\mathcal{M}) \,\}$$

nie jest częściowo rozstrzygalny.

Dowód:

Przypuśćmy, że $L(\mathcal{M}_p) = L_p$. Wtedy

$$w_{\mathcal{M}_p} \in L(\mathcal{M}_p) \iff w_{\mathcal{M}_p} \notin L(\mathcal{M}_p)$$

$$\mathcal{M}_p \in L(\mathcal{M}_p) \iff \mathcal{M}_p \notin L(\mathcal{M}_p)$$

Plan

Całkowite deterministyczne maszyny Turinga

Deterministyczna maszyna Turinga $\mathcal{M} = (A, Q, q_0, q_{\mathsf{tak}}, q_{\mathsf{nie}}, T, \mathbb{B}, \delta)$

- $q_{\mathsf{tak}} \in Q$ – stan akceptujący
- $q_{\mathsf{nie}} \in Q$ – stan odrzucający
- $\delta : (Q - \{q_{\mathsf{tak}}, q_{\mathsf{nie}}\}) \times T \to Q \times T \times \{\leftarrow, \circlearrowright, \rightarrow\}$

Konfiguracje *końcowe*:

$$T^*\{q_{\mathsf{tak}}, q_{\mathsf{nie}}\}\, T^* = \{\, w\, q\, w' \;:\; q \in \{q_{\mathsf{tak}}, q_{\mathsf{nie}}\};\; w, w' \in T^* \,\}$$

Deterministyczna maszyna Turinga $\mathcal{M}$ *zatrzymuje się* dla słowa wejściowego w jeśli

$$q_0\, w \longrightarrow^*_{\mathcal{M}} c$$

dla jakiejś konfiguracji końcowej c.

Maszyna $\mathcal{M}$ jest *całkowita*, jeśli zatrzymuje się dla każdego słowa wejściowego.

Przykład (całkowita maszyna deterministyczna)

- $q_{\mathsf{tak}} = \mathsf{ok}$
- $q_{\mathsf{nie}} = \mathsf{nok}$

	a	b	$\mathbb{B}$	$\#$
$\mathsf{start}_{\rightarrow}$	$(\mathsf{cont}_{\rightarrow}, \#, \rightarrow)$	$(\mathsf{check}, b, \rightarrow)$	$(\mathsf{nok}, \mathbb{B}, \circlearrowright)$	$(\mathsf{nok}, \#, \circlearrowright)$
$\mathsf{cont}_{\rightarrow}$	$(\mathsf{cont}_{\rightarrow}, a, \rightarrow)$	$(\mathsf{cont}_{\rightarrow}, b, \rightarrow)$	$(\mathsf{start}_{\leftarrow}, \mathbb{B}, \leftarrow)$	$(\mathsf{start}_{\leftarrow}, \#, \leftarrow)$
$\mathsf{start}_{\leftarrow}$	$(\mathsf{cont}_{\leftarrow}, \#, \leftarrow)$	$(\mathsf{nok}, b, \circlearrowright)$	$(\mathsf{nok}, \mathbb{B}, \circlearrowright)$	$(\mathsf{nok}, \#, \circlearrowright)$
$\mathsf{cont}_{\leftarrow}$	$(\mathsf{cont}_{\leftarrow}, a, \leftarrow)$	$(\mathsf{cont}_{\leftarrow}, b, \leftarrow)$	$(\mathsf{nok}, \mathbb{B}, \circlearrowright)$	$(\mathsf{start}_{\rightarrow}, \#, \rightarrow)$
check	$(\mathsf{nok}, a, \circlearrowright)$	$(\mathsf{nok}, b, \circlearrowright)$	$(\mathsf{ok}, \mathbb{B}, \circlearrowright)$	$(\mathsf{ok}, \#, \circlearrowright)$

$$L(\mathcal{M}) = \{\, a^n b\, a^n \;:\; n \in \mathbb{N} \,\}$$

Problemy rozstrzygalne

Język (problem) $L \subseteq A^*$ nazywamy

częściowo rozstrzygalnym, albo *rekurencyjnie przeliczalnym*,

jeśli $L = L(\mathcal{M})$ dla pewnej deterministycznej maszyny Turinga $\mathcal{M}$.

Język (problem) $L \subseteq A^*$ nazywamy

rozstrzygalnym, albo *rekurencyjnym*,

jeśli $L = L(\mathcal{M})$ dla pewnej całkowitej deterministycznej maszyny Turinga $\mathcal{M}$.

Pytanie

Czy języki bezkontekstowe są rozstrzygalne? A ich dopełnienia?

Fakt

Jeśli $L \subseteq A^*$ jest rozstrzygalny to $\overline{L} = A^* - L$ też.

Dopełnienie

Fakt

Jeśli L i $\overline{L}$ są częściowo rozstrzygalne, to są rozstrzygalne.

Dowód:

Z dwóch deterministycznych maszyn Turinga dla języków L i $\overline{L}$ skonstruujemy *całkowitą* deterministyczną maszynę dla L...

rysunek

Wniosek

Dla każdego języka L zachodzi dokładnie jeden z warunków:

- *L jest rozstrzygalny (i $\overline{L}$ też)*
- *L jest częściowo rozstrzygalny, $\overline{L}$ nie jest częściowo rozstrzygalny*
- *$\overline{L}$ jest częściowo rozstrzygalny, L nie jest częściowo rozstrzygalny*
- *L i $\overline{L}$ nie są częściowo rozstrzygalne*

rysunek

Problem stopu jest nierozstrzygalny

Twierdzenie

Problem stopu jest częściowo rozstrzygalny, ale nie jest rozstrzygalny.

Dowód:

Przypuśćmy, że

$$L(\mathcal{M}_u) \;=\; L_u \;=\; \{\, (w_{\mathcal{M}} \,\$\, w) \;:\; w \in L(\mathcal{M}) \,\}$$

dla całkowitej deterministycznej maszyny M_u. Skonstruujemy (całkowitą deterministyczną) maszynę M_p dla języka

$$L_p \;=\; \{\, w_{\mathcal{M}} \;:\; w_{\mathcal{M}} \notin L(\mathcal{M}) \,\}$$

rysunek

Wniosek

Istnieje deterministyczna maszyna $\mathcal{M}$ i słowo wejściowe w takie, że $w \notin L(\mathcal{M})$, ale nie da się tego dowieść na gruncie teorii zbiorów.

Inne przykładowe problemy częściowo rozstrzygalne, ale nierozstrzygalne:

Niepustość języka maszyny Turinga

Dane:	maszyna Turinga $\mathcal{M}$
Wynik:	czy $L(\mathcal{M}) \neq \emptyset$?

Nieuniwersalność języka bezkontektowego

Dane:	język bezkontekstowy $L \subseteq A^*$
Wynik:	czy $L \neq A^*$?

Plan

Funkcje częściowo obliczalne I

Relację $R \subseteq (A^*)^n$ możemy utożsamić z językiem

$$L_R \;=\; \{\, w_1\$w_2\$\ldots\$w_n \;:\; (w_1, w_2, \ldots, w_n) \in R \,\}$$

Definicja 1: Funkcję częściową $f : (A^*)^n \to A^*$ nazywamy

częściowo obliczalną, albo *częściowo rekurencyjną*,

jeśli język

$$L_f \;=\; \{\, w_1\$w_2\$\ldots\$w_n\$f(w_1, w_2, \ldots, w_n) \;:\; (w_1, w_2, \ldots, w_n) \in \mathrm{dom}(f) \,\}$$

jest częściowo rozstrzygalny.

Pytanie

No dobrze, ale jak *obliczyć* wartość funkcji?

Deterministyczna maszyna Turinga $\mathcal{M} \;=\; (A, Q, q_0, q_{\text{stop}}, T, \mathbb{B}, \delta)$

- $q_{\text{stop}} \in Q$ – stan końcowy
- $\delta : (Q - \{q_{\text{stop}}\}) \times T \to Q \times T \times \{\leftarrow, \circlearrowright, \rightarrow\}$

Funkcja częściowa

$$F(\mathcal{M}) : (A^*)^n \to A^*$$

obliczana przez maszynę $\mathcal{M}$:

$$F(\mathcal{M})(w_1, \ldots, w_n) \;=\; v \iff q_0\, w_1\$w_2\$\ldots\$w_n \longrightarrow^*_{\mathcal{M}} q_{\text{stop}}\, v$$

Definicja 2: Funkcję częściową $f : (A^*)^n \to A^*$ nazywamy

częściowo obliczalną, albo *częściowo rekurencyjną*,

jeśli $f \;=\; F(\mathcal{M})$ dla jakiejś maszyny $\mathcal{M}$.

Funkcje częściowo obliczalne i obliczalne

Funkcja częściowa $f : (A^*)^n \to A^*$.

Definicja 1: L_f jest częściowo rozstrzygalny.

$$L_f = \{ w_1\$w_2\$\ldots\$w_n\$f(w_1, w_2, \ldots, w_n) \ : \ (w_1, w_2, \ldots, w_n) \in \mathrm{dom}(f) \}$$

Definicja 2: $f = F(\mathcal{M})$ dla jakiejś maszyny $\mathcal{M}$.

$$F(\mathcal{M})(w_1, \ldots, w_n) = v \iff q_0\, w_1\$w_2\$\ldots\$w_n \longrightarrow^*_{\mathcal{M}} q_{\mathbf{stop}}\, v$$

Pytanie

Czy te dwie definicje są równoważne?

Całkowitą, częściowo obliczalną funkcję $f : (A^*)^n \to A^*$ nazywamy

obliczalną, albo *rekurencyjną*.

Podsumowanie

	deterministyczne maszyny Turinga	całkowite deterministyczne maszyny Turinga
języki (problemy decyzyjne)	częściowo rozstrzygalne rekurencyjnie przeliczalne	rozstrzygalne
funkcje (problemy obliczeniowe)	częściowo obliczalne częściowo rekurencyjne	obliczalne rekurencyjne

Teza Churcha-Turinga

funkcje obliczane przez maszyny Turinga	=	problemy obliczeniowe, dla których istnieje efektywny algorytm, przy założeniu nieograniczonych zasobów

albo:

maszyny Turinga	=	komputery

teoria obliczeń	algorytmy
język	problem decyzyjny
funkcja	problem obliczeniowy
maszyna Turinga	algorytm / program / system komputerowy
słowo wejściowe	dane wejściowe / instancja
uniwersalna maszyna Turinga	interpreter programów
częściowo rozstrzygalny / obliczalna	rozwiązywalny algorytmicznie
rozstrzygalny / obliczalna	rozwiązywalny algorytmicznie, własność stopu

W następnym odcinku:

Nieobliczalność