Logika i teoria typów

Wykład 7

23 listopada 2022

Tydzień temu: Monotonic automata

Monotonic automaton: $\mathcal{M} = \langle Q, R, f, \mathcal{I} \rangle$,

- Q is a finite set of states, $f \in Q$ the final state.
- R is a finite set of registers;
- $\mathcal{I}$ is a finite set of instructions of the form:
  (1) q : check S_1; set S_2; jmp p,
  (2) q : jmp p_1 and p_2,
  where $q, p, p_1, p_2 \in Q$ and $S_1, S_2 \subseteq R$.

Lemma

The halting problem for monotonic automata
Is a given configuration accepting?
is PSPACE*-complete.*

Część trudna: Monotonic encoding of an ATM

Alternating time $p(n)$.

Registers: $stan_s^t$, lit_{ia}^t, poz_i^t,

Initial configuration:

$C_0 = \langle loop_1^0, \{lit_{1a_1}^0, ..., lit_{na_n}^0, lit_{n+1B}^0, ..., lit_{p(n)B}^0, stan_0^0, poz_1^0\}\rangle$.

represents initial ID of TM for input $w = a_1...a_n$.

Later:

$C = \langle loop_1^t, S\rangle$ encodes the first t steps of TM computation,

e.g., $lit_{6a}^5 \in S$ iff after 5 steps, the 6th cell contains a.

Co trzeba udowodnić?

Lemat: *Załóżmy, że do zbioru rejestrów S należy dokładnie jeden rejestr $stan_s^t$, dokładnie jeden rejestr poz_i^t i dokładnie po jednym lit_{ia}^t dla każdego i. Ale nie ma żadnego rejestru postaci $stan_s^u$, lit_{ia}^u, poz_i^u, dla $u > t$. Wówczas:*

Konfiguracja $C = \langle loop_1^t, S\rangle$ automatu monotonicznego jest akceptująca wtedy i tylko wtedy, gdy maszyna Turinga akceptuje odpowiednie ID *w alternującym czasie $p(n) - t$.*

Dowód: Indukcja ze względu na $p(n) - t$.

Wniosek: *Konfiguracja C_0 automatu monotonicznego jest akceptująca wtedy i tylko wtedy, gdy maszyna Turinga akceptuje słowo w w alternującym czasie $p(n)$.*

Example instructions for $(s, a) \Rightarrow (u, b, +\varepsilon)$

When $i < p(n)$:

$loop_i^t$:
check $stan_s^t, poz_i^t, lit_{ia}^t$; set $stan_u^{t+1}, poz_{i+\varepsilon}^{t+1}, lit_{ib}^{t+1}$; jmp $loop_{i+1}^t$;
check $stan_s^t, poz_j^t, lit_{ia}^t$; set lit_{ia}^{t+1}; jmp $loop_{i+1}^t$;

When $i = p(n)$:

$loop_i^t$:
check $stan_s^t, poz_i^t, lit_{ia}^t$; set $stan_u^{t+1}, poz_{i+\varepsilon}^{t+1}, lit_{ib}^{t+1}$; jmp $loop_1^{t+1}$;
check $stan_s^t, poz_j^t, lit_{ia}^t$; set lit_{ia}^{t+1}; jmp $loop_1^{t+1}$.

Example instructions: universal split in state s

Assume that:
– TM does not write nor move in universal states.
– The universal step does not depend on symbol scanned.

When $i < p(n)$:

$loop_i^t$: check $stan_s^t, lit_{ia}^t$; set lit_{ia}^{t+1}; jmp $loop_{i+1}^t$;

When $i = p(n)$:

$loop_i^t$: check $stan_s^t, lit_{ia}^t, poz_j^t$; set $lit_{ia}^{t+1}, poz_j^{t+1}$; jmp $split_{s_1s_2}^{t+1}$;

$split_{s_1s_2}^{t+1}$: jmp $go_{s_1}^{t+1}$ and $go_{s_2}^{t+1}$;

$go_{s_1}^{t+1}$: check $\varnothing$; set $stan_{s_1}^{t+1}$; jmp $loop_1^{t+1}$;

Instrukcja końcowa

Dla czasu $p(n)$ i stanu końcowego s maszyny Turinga:

$loop_1^{p(n)}$: check $stan_s^{p(n)}$; set $\varnothing$; jmp f

Złożoność intuicjonistycznej logiki zdaniowej

Twierdzenie (Statman)

Intuicjonistyczny rachunek zdań jest PSPACE*-zupełny.*

Konstrukcja dowodu jako obliczenie

Osąd $\Gamma \vdash p$ (gdzie p – atom), można interpretować jako konfigurację automatu:

- Cel atomowy p to stan maszyny;
- Otoczenie Γ to zawartość pamięci, w tym program.
- Założenie $p \to q$ umożliwia zmianę stanu z q na p.
- Założenie $(b \to p) \to q$ umożliwia zmianę stanu z q na p, z jednoczesnym zapisaniem b w pamięci.
- Założenie $a \to p \to q$ umożliwia zmianę stanu z q na p, pod warunkiem, że w pamięci jest zapisane a.

Uwaga: (1) nie można usunąć danej z pamięci;
(2) nie można sprawdzić, że danej w pamięci nie ma.

Programs into proofs!

Given $\mathcal{M} = \langle Q, R, f, \mathcal{I} \rangle$ and $C_0 = \langle q_0, S_0 \rangle$,
define a set of axioms Γ so that

$$\Gamma \vdash q_0 \quad \text{iff} \quad C_0 \text{ is accepting.}$$

Propositional variables: states and registers of $\mathcal{M}$.

Put all of S_0 into Γ, also $f \in \Gamma$.

Other axioms represent instructions of $\mathcal{M}$.

Instructions seen as axioms

Axiom for q : `check` S_1; `set` S_2; `jmp` p,
where $S_1 = \{s_1^1, ..., s_1^k\}$ and $S_2 = \{s_2^1, ..., s_2^\ell\}$:

(1) $\quad s_1^1 \to \cdots \to s_1^k \to (s_2^1 \to \cdots \to s_2^\ell \to p) \to q.$

Axiom for q : `jmp` p_1 `and` p_2:

(2) $\quad p_1 \to p_2 \to q.$

Proof construction as computation

To prove $\Gamma, S \vdash q$, one can:

- Note that $q = f$ and observe $f \in \Gamma$;
- Use axiom $p_1 \to p_2 \to q$ and prove in parallel: $\Gamma, S \vdash p_1$ and $\Gamma, S \vdash p_2$
- Use axiom $s_1^1 \to \cdots \to s_1^k \to (s_2^1 \to \cdots \to s_2^\ell \to p) \to q$, provided $S_1 = \{s_1^1, ..., s_1^k\} \subseteq S$, and prove that $\Gamma, S, s_2^1, ..., s_2^\ell \vdash p$

The technical lemma we need

Lemma
$\Gamma, S \vdash q$ *iff* $\langle q, S_0 \cup S \rangle$ *is accepting.*

Proof.
($\Rightarrow$) Induction wrt the size of normal proof.
($\Leftarrow$) Induction wrt the definition of acceptance. □

Wniosek
Problem stopu dla automatów monotonicznych redukuje się (w pamięci logarytmicznej) do intuicjonistycznej logiki zdaniowej z samą implikacją.

Rząd formuły implikacyjnej (typu)

Rząd zero = atomy;
Rząd $n+1$ = formuły $\alpha_1 \to \cdots \alpha_k \to p$,
gdzie α_i są rzędu co najwyżej n.

Np. ta formuła jest rzędu 3:

$((p \to q) \to r) \to ((r \to p \to s) \to r) \to p$

Automaty monotoniczne są reprezentowane formułami rzędu 3.

Twierdzenie:

Intuicjonistyczny rachunek zdań jest PSPACE*-zupełny.*

To mało powiedziane.

W istocie PSPACE-zupełny jest fragment implikacyjny i to ograniczony do formuł rzędu 3.

Twierdzenie

Intuicjonistyczny rachunek zdań redukuje się w pamięci logarytmicznej do fragmentu implikacyjnego rzędu 3.

Dowód:

Dowolna formuła $\to$ automat $\to$ formuła implikacyjna rzędu 3.

Przykład:

Formuła $(((e \to d) \to c) \to b) \to a$ jest rzędu 4

i nie jest *równoważna* żadnej formule rzędu 3.

O wyższości Wajsberga i Ben-Yellesa nad Davisem i Putnamem

Klasyczny rachunek zdań:

– reprezentuje problemy obliczeniowe o złożoności NP;

– jako koniunkcje statycznych więzów (SAT);

– dla których trzeba szukać rozwiązań ad hoc.

Intuicjonistyczny rachunek zdań:

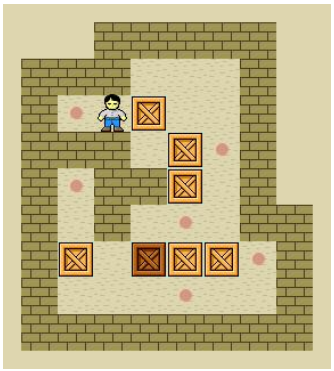
– reprezentuje problemy obliczeniowe o złożoności PSPACE;

– jako dynamiczne zadania dowodowe,

– których rozwiązania bezpośrednio realizują obliczenie.

Przykład

Do tego zadania SAT-solvery się nie nadają.



Normalizacja

Twierdzenie: *Jeśli istnieje dowód osądu $\Gamma \vdash \varphi$, to istnieje dowód normalny.*

Lepsze twierdzenie: *Każdy dowód osądu $\Gamma \vdash \varphi$ można przekształcić w dowód normalny.*

Jeszcze lepsze twierdzenie: *Każdy ciąg redukcji prowadzi do dowodu normalnego.*

Przypadek implikacyjny

Term w postaci β-normalnej: term, którego nie można redukować.

To jest to samo co implikacyjny dowód normalny.

Normalizacja:
Jeśli $\Gamma \vdash M : \tau$, to $M \twoheadrightarrow_\beta N$, gdzie N jest normalny.

Silna normalizacja:
Jeśli $\Gamma \vdash M : \tau$, to każdy ciąg redukcji termu M jest skończony.

Silna normalizacja

Twierdzenie: *Rachunek lambda z typami prostymi ma własność silnej normalizacji: każdy poprawnie typowany term jest silnie $\beta\eta$-normalizowalny.*

Wstęp do dowodu

(tu na razie nie ma typów)

Lemat 0:

(1) Podterm termu silnie normalizowalnego jest silnie normalizowalny.

(2) Jeśli $A[x := B] \in SN$, to $A \in SN$.

(3) Jeśli $A \in SN$ i $A \to B$, to $B \in SN$.

Wstęp do dowodu (tu na razie nie ma typów)

Definicja: Określamy klasę termów $\mathcal{S}$ przez indukcję:

1. Jeśli $N_1, \ldots, N_k \in \mathcal{S}$, to $xN_1 \ldots N_k \in \mathcal{S}$;
2. Jeśli $N \in \mathcal{S}$, to $\lambda x\, N \in \mathcal{S}$;
3. Jeśli $Q \in \mathcal{S}$ oraz $P[x := Q]N_1 \ldots N_k \in \mathcal{S}$, to $(\lambda x\, P)QN_1 \ldots N_k \in \mathcal{S}$.

Lemat 1: Jeśli $M \in \mathcal{S}$, to $M \in \mathrm{SN}_{\beta\eta}$.

Dowód: Indukcja ze względu na **definicję** $\mathcal{S}$.

(1) Wszystkie redukcje w termie $xN_1 \ldots N_k$ muszą być „wewnętrzne". Teza wynika od razu z założenia indukcyjnego.

Lemat 1: *Jeśli $M \in \mathcal{S}$, to $M \in \mathrm{SN}_{\beta\eta}$.*

Dowód: Przypadek (2) Jeśli $N \in \mathcal{S}$, to $\lambda x\, N \in \mathcal{S}$.

Jeśli wszystkie redukcje w $\lambda x\, N$ są wewnątrz N, to teza wynika z założenia indukcyjnego. Jedyna inna możliwość jest taka:

$$\lambda x\, N \twoheadrightarrow_{\beta\eta} \lambda x\, Px \to_\eta P \twoheadrightarrow_{\beta\eta} \cdots$$

gdzie $N \twoheadrightarrow_{\beta\eta} Px$. Ale wtedy $Px \in \mathrm{SN}_{\beta\eta}$ z założenia indukcyjnego dla N, więc także $P \in \mathrm{SN}_{\beta\eta}$.

Lemat 1: *Jeśli* $M \in \mathcal{S}$, *to* $M \in \mathrm{SN}_{\beta\eta}$.

Dowód: (3) Jeśli $Q \in \mathcal{S}$ oraz $P[x := Q]N_1 \dots N_k \in \mathcal{S}$, to $(\lambda x\, P)QN_1 \dots N_k \in \mathcal{S}$.

Z założenia indukcyjnego $P, Q, N_1, \dots, N_k \in \mathrm{SN}_{\beta\eta}$. Zatem redukcje wewnętrzne termu $M = (\lambda x\, P)QN_1 \dots N_k$ muszą się skończyć. Pozostaje jeszcze redukcja czołowa:

$M \twoheadrightarrow_{\beta\eta} (\lambda x\, P')Q'N_1' \dots N_k' \to_\beta P'[x := Q']N_1' \dots N_k' \twoheadrightarrow_{\beta\eta} \cdots$

Ale $\mathrm{SN}_{\beta\eta} \ni P[x := Q]N_1 \dots N_k \twoheadrightarrow_{\beta\eta} P'[x := Q']N_1' \dots N_k'$, więc też $P'[x := Q']N_1' \dots N_k' \in \mathrm{SN}_{\beta\eta}$.

Oczywiście: *Jeśli* $M \in \mathrm{SN}_{\beta\eta}$, *to* $M \in \mathrm{SN}_\beta$.

1. $N_1, \dots, N_k \in \mathcal{S} \quad \Rightarrow \quad xN_1 \dots N_k \in \mathcal{S}$;
2. $N \in \mathcal{S} \quad \Rightarrow \quad \lambda x\, N \in \mathcal{S}$;
3. $Q \in \mathcal{S}, \quad P[x := Q]N_1 \dots N_k \in \mathcal{S} \quad \Rightarrow \quad (\lambda x\, P)QN_1 \dots N_k \in \mathcal{S}$.

Lemat 2: Jeśli $M \in \mathrm{SN}_\beta$, to $M \in \mathcal{S}$.

Dowód: Indukcja ze względu na dwa parametry:

– pierwszy to maksymalna długość redukcji termu M,
– drugi to długość termu M.

Przypadek 1: $M = (\lambda x\, P)QN_1 \dots N_k$.
Wtedy $Q \in \mathcal{S}$ z założenia indukcyjnego (pierwszy parametr mniejszy).

Także $P[x := Q]N_1 \dots N_k \in \mathcal{S}$ (pierwszy parametr mniejszy).
Zatem oba te termy są w $\mathcal{S}$, skąd $M \in \mathcal{S}$.

1. $N_1, \dots, N_k \in \mathcal{S} \quad \Rightarrow \quad xN_1 \dots N_k \in \mathcal{S}$;
2. $N \in \mathcal{S} \quad \Rightarrow \quad \lambda x\, N \in \mathcal{S}$;
3. $Q \in \mathcal{S}, \quad P[x := Q]N_1 \dots N_k \in \mathcal{S} \quad \Rightarrow \quad (\lambda x\, P)QN_1 \dots N_k \in \mathcal{S}$.

Lemat 2: Jeśli $M \in \mathrm{SN}_\beta$, to $M \in \mathcal{S}$.

Dowód: Indukcja ze względu na dwa parametry:

– pierwszy to maksymalna długość redukcji termu M,
– drugi to długość termu M.

Przypadek 2: Term M jest postaci $xN_1 \dots N_k$ lub jest abstrakcją. Wtedy teza wynika od razu z założenia indukcyjnego ze względu na pierwszy lub drugi parametr.

Wniosek

- Klasa $\mathcal{S}$ to to samo co $\mathrm{SN}_{\beta\eta}$ i to samo co SN_β.
- Produkt uboczny: β-SN i $\beta\eta$-SN to to samo.

Dowód silnej normalizacji

Konwencja:

- termy (w tym zmienne) mają ustalone typy;
- nie piszemy tych typów.

Główny lemat

Lemat 3: *Jeśli* $M, P \in \mathcal{S}$, *to* $M[x := P] \in \mathcal{S}$.

Dowód twierdzenia: Z lematów 1 i 2 wynika, że $\mathrm{SN}_\beta = \mathrm{SN}_{\beta\eta} = \mathcal{S}$. Wystarczy więc udowodnić, że każdy (poprawny) term M jest w $\mathcal{S}$.

– Jeśli M jest zmienną, to oczywiste.

– Jeśli M jest abstrakcją, to teza wynika z założenia indukcyjnego.

– Jeśli M jest aplikacją PQ to stosujemy lemat 3 do podstawienia $(xQ)[x := P]$.