# Rekurencja w logice wyższego rzędu

Łukasz Czajka

Wydział Matematyki, Informatyki i Mechaniki
Uniwersytet Warszawski

12.04.2013

Ta prezentacja dotyczy

- pewnego połączenia logiki wyższego rzędu z beztypowym rachunkiem lambda,

Ta prezentacja dotyczy

- pewnego połączenia logiki wyższego rzędu z beztypowym rachunkiem lambda,
- czyli jak dodać do logiki wyższego rzędu możliwość **dowolnych definicji rekurencyjnych**.

# Logika wyższego rzędu

## Składnia

- Typy: $\mathcal{T} \quad ::= \quad \mathrm{Prop} \mid \mathcal{B} \mid \mathcal{T} \to \mathcal{T}$

# Logika wyższego rzędu

## Składnia

- Typy: $\mathcal{T} \quad ::= \quad \mathrm{Prop} \mid \mathcal{B} \mid \mathcal{T} \to \mathcal{T}$
- Stałe:
  - $\forall_\tau \in \Sigma_{(\tau \to \mathrm{Prop}) \to \mathrm{Prop}}$
  - $\supset \in \Sigma_{\mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}}$

# Logika wyższego rzędu

## Składnia

- Typy: $\mathcal{T} \quad ::= \quad \mathrm{Prop} \mid \mathcal{B} \mid \mathcal{T} \to \mathcal{T}$
- Stałe:
  - $\forall_\tau \in \Sigma_{(\tau \to \mathrm{Prop}) \to \mathrm{Prop}}$
  - $\supset \in \Sigma_{\mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}}$
- Zbiór termów $T_\tau$ typu $\tau$ w rachunku lambda z typami prostymi:
  - $\Sigma_\tau, V_\tau \subseteq T_\tau$,
  - jeśli $t \in T_{\tau_1 \to \tau_2}$ i $s \in T_{\tau_1}$ to $ts \in T_{\tau_2}$,
  - jeśli $x \in V_{\tau_1}$ i $t \in T_{\tau_2}$ to $\lambda x : \tau \,.\, t \in T_{\tau_1 \to \tau_2}$.

# Logika wyższego rzędu

## Składnia

- Typy: $\mathcal{T} \quad ::= \quad \mathrm{Prop} \mid \mathcal{B} \mid \mathcal{T} \to \mathcal{T}$
- Stałe:
  - $\forall_\tau \in \Sigma_{(\tau \to \mathrm{Prop}) \to \mathrm{Prop}}$
  - $\supset \in \Sigma_{\mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}}$
- Zbiór termów $T_\tau$ typu $\tau$ w rachunku lambda z typami prostymi:
  - $\Sigma_\tau, V_\tau \subseteq T_\tau$,
  - jeśli $t \in T_{\tau_1 \to \tau_2}$ i $s \in T_{\tau_1}$ to $ts \in T_{\tau_2}$,
  - jeśli $x \in V_{\tau_1}$ i $t \in T_{\tau_2}$ to $\lambda x : \tau \,.\, t \in T_{\tau_1 \to \tau_2}$.
- Konwencje:
  - $\varphi$, $\psi$, itp. – termy typu $\mathrm{Prop}$,
  - $\varphi \supset \psi \equiv \supset \varphi \psi$,
  - $\forall x : \tau \,.\, \varphi \equiv \forall_\tau (\lambda x : \tau \,.\, \varphi)$.

# Logika wyższego rzędu

## Reguły naturalnej dedukcji (prosty intuicjonistyczny intensjonalny wariant)

$$\frac{}{\Delta, \varphi \vdash \varphi}$$

$$\supset_i: \frac{\Delta, \varphi \vdash \psi}{\Delta \vdash \varphi \supset \psi} \qquad \supset_e: \frac{\Delta \vdash \varphi \supset \psi \quad \Delta \vdash \varphi}{\Delta \vdash \psi}$$

$$\forall_i : \frac{\Delta \vdash \varphi}{\Delta \vdash \forall x : \tau \,.\, \varphi} \; x \in V_\tau, x \notin FV(\Delta)$$

$$\forall_e : \frac{\Delta \vdash \forall x : \tau \,.\, \varphi}{\Delta \vdash \varphi[x/t]} \; t \in T_\tau$$

$$\mathrm{conv} : \frac{\Delta \vdash \varphi \qquad \varphi =_\beta \psi}{\Delta \vdash \psi}$$

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Naiwne podejście

Opuszczamy typy.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Składnia

- Brak typów.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Składnia

- Brak typów.
  - Dziedziny mogą być reprezentowane przez stałe.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Składnia

- Brak typów.
  - Dziedziny mogą być reprezentowane przez stałe.
- Stałe: $\forall, \supset \in \Sigma$.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Składnia

- Brak typów.
  - Dziedziny mogą być reprezentowane przez stałe.
- Stałe: $\forall, \supset \in \Sigma$.
- Zbiór termów $T$ *beztypowego* rachunku lambda:
  - $\Sigma, V \subseteq T$,
  - jeśli $t \in T$ i $s \in T$ to $ts \in T$,
  - jeśli $x \in V$ i $t \in T$ to $\lambda x \,.\, t \in T$.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Składnia

- Brak typów.
  - Dziedziny mogą być reprezentowane przez stałe.
- Stałe: $\forall, \supset \in \Sigma$.
- Zbiór termów $T$ *beztypowego* rachunku lambda:
  - $\Sigma, V \subseteq T$,
  - jeśli $t \in T$ i $s \in T$ to $ts \in T$,
  - jeśli $x \in V$ i $t \in T$ to $\lambda x \,.\, t \in T$.
- Konwencje:
  - $\varphi \supset \psi \equiv \supset \varphi \psi$,
  - $\forall x \,.\, \varphi \equiv \forall(\lambda x \,.\, \varphi)$.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Rekurencja

W beztypowym rachunku lambda każdy układ równań postaci

$$\begin{aligned} z_1 x_1 \ldots x_m &=_\beta \Phi_1(z_1, \ldots, z_n, x_1, \ldots, x_m) \\ &\vdots \\ z_n x_1 \ldots x_m &=_\beta \Phi_n(z_1, \ldots, z_n, x_1, \ldots, x_m) \end{aligned}$$

ma rozwiązanie dla $z_1, \ldots, z_n$, gdzie wyrażenia $\Phi_i(z_1, \ldots, z_n, x_1, \ldots, x_m)$ są dowolnymi termami, których wszystkie zmienne wolne są wyszczególnione w nawiasie.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Rekurencja

W beztypowym rachunku lambda każdy układ równań postaci

$$\begin{aligned} z_1 x_1 \ldots x_m &=_\beta \Phi_1(z_1, \ldots, z_n, x_1, \ldots, x_m) \\ &\vdots \\ z_n x_1 \ldots x_m &=_\beta \Phi_n(z_1, \ldots, z_n, x_1, \ldots, x_m) \end{aligned}$$

ma rozwiązanie dla $z_1, \ldots, z_n$, gdzie wyrażenia $\Phi_i(z_1, \ldots, z_n, x_1, \ldots, x_m)$ są dowolnymi termami, których wszystkie zmienne wolne są wyszczególnione w nawiasie.

Innymi słowami, dla każdego zbioru równań powyższej postaci istnieją takie termy $t_1, \ldots, t_n$, że dla dowolnych termów $s_1, \ldots, s_m$ zachodzi $t_i s_1 \ldots s_m =_\beta \Phi_i(t_1, \ldots, t_n, s_1, \ldots, s_m)$ dla każdego $i = 1, \ldots, n$.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Reguły (logika minimalna)

$$\frac{}{\Delta, \varphi \vdash \varphi}$$

$$\supset_i: \frac{\Delta, \varphi \vdash \psi}{\Delta \vdash \varphi \supset \psi} \qquad \supset_e: \frac{\Delta \vdash \varphi \supset \psi \qquad \Delta \vdash \varphi}{\Delta \vdash \psi}$$

$$\forall_i : \frac{\Delta \vdash \varphi}{\Delta \vdash \forall x \,.\, \varphi}\ x \notin FV(\Delta) \qquad \forall_e : \frac{\Delta \vdash \forall x \,.\, \varphi}{\Delta \vdash \varphi[x/t]}\ t \in T$$

$$\mathrm{conv} : \frac{\Delta \vdash \varphi \qquad \varphi =_\beta \psi}{\Delta \vdash \psi}$$

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Reguły (logika minimalna)

$$\frac{}{\Delta, \varphi \vdash \varphi}$$

$$\supset_i: \frac{\Delta, \varphi \vdash \psi}{\Delta \vdash \varphi \supset \psi} \qquad \supset_e: \frac{\Delta \vdash \varphi \supset \psi \qquad \Delta \vdash \varphi}{\Delta \vdash \psi}$$

$$\forall_i : \frac{\Delta \vdash \varphi}{\Delta \vdash \forall x \,.\, \varphi}\ x \notin FV(\Delta) \qquad \forall_e : \frac{\Delta \vdash \forall x \,.\, \varphi}{\Delta \vdash \varphi[x/t]}\ t \in T$$

$$\mathrm{conv} : \frac{\Delta \vdash \varphi \qquad \varphi =_\beta \psi}{\Delta \vdash \psi}$$

Taki system (mniej więcej) był początkowo rozważany w latach 30-tych (przez Churcha, Curry'ego, ...), ...

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Paradoks Curry'ego

. . . i oczywiście okazał się sprzeczny.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Paradoks Curry'ego

. . . i oczywiście okazał się sprzeczny.

Dla dowolnego termu $\psi$, definiujemy $\varphi$ przez $\varphi =_\beta \varphi \supset \psi$.

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Paradoks Curry'ego

. . . i oczywiście okazał się sprzeczny.

Dla dowolnego termu $\psi$, definiujemy $\varphi$ przez $\varphi =_\beta \varphi \supset \psi$.

(1) $\varphi \vdash \varphi$ z aksjomatu,

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Paradoks Curry'ego

. . . i oczywiście okazał się sprzeczny.

Dla dowolnego termu $\psi$, definiujemy $\varphi$ przez $\varphi =_\beta \varphi \supset \psi$.

(1) $\varphi \vdash \varphi$ z aksjomatu,

(2) $\varphi \vdash \varphi \supset \psi$ z (1) za pomocą $\mathrm{conv}$,

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Paradoks Curry'ego

. . . i oczywiście okazał się sprzeczny.

Dla dowolnego termu $\psi$, definiujemy $\varphi$ przez $\varphi =_\beta \varphi \supset \psi$.

(1) $\varphi \vdash \varphi$ z aksjomatu,

(2) $\varphi \vdash \varphi \supset \psi$ z (1) za pomocą $\mathrm{conv}$,

(3) $\varphi \vdash \psi$ z (1) i (2) za pomocą $\supset_e$,

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Paradoks Curry'ego

. . . i oczywiście okazał się sprzeczny.

Dla dowolnego termu $\psi$, definiujemy $\varphi$ przez $\varphi =_\beta \varphi \supset \psi$.

(1) $\varphi \vdash \varphi$ z aksjomatu,

(2) $\varphi \vdash \varphi \supset \psi$ z (1) za pomocą $\mathrm{conv}$,

(3) $\varphi \vdash \psi$ z (1) i (2) za pomocą $\supset_e$,

(4) $\vdash \varphi \supset \psi$ z (3) za pomocą $\supset_i$,

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Paradoks Curry'ego

. . . i oczywiście okazał się sprzeczny.

Dla dowolnego termu $\psi$, definiujemy $\varphi$ przez $\varphi =_\beta \varphi \supset \psi$.

(1) $\varphi \vdash \varphi$ z aksjomatu,

(2) $\varphi \vdash \varphi \supset \psi$ z (1) za pomocą $\mathrm{conv}$,

(3) $\varphi \vdash \psi$ z (1) i (2) za pomocą $\supset_e$,

(4) $\vdash \varphi \supset \psi$ z (3) za pomocą $\supset_i$,

(5) $\vdash \varphi$ z (4) za pomocą $\mathrm{conv}$,

# Logika wyższego rzędu z rekurencją (sprzeczna)

## Paradoks Curry'ego

. . . i oczywiście okazał się sprzeczny.

Dla dowolnego termu $\psi$, definiujemy $\varphi$ przez $\varphi =_\beta \varphi \supset \psi$.

(1) $\varphi \vdash \varphi$ z aksjomatu,

(2) $\varphi \vdash \varphi \supset \psi$ z (1) za pomocą $\mathrm{conv}$,

(3) $\varphi \vdash \psi$ z (1) i (2) za pomocą $\supset_e$,

(4) $\vdash \varphi \supset \psi$ z (3) za pomocą $\supset_i$,

(5) $\vdash \varphi$ z (4) za pomocą $\mathrm{conv}$,

(6) $\vdash \psi$ z (4) i (5) za pomocą $\supset_e$.

# Logika wyższego rzędu

## Dygresja

W zwykłej logice wyższego rzędu dozwolone są stałe typu $\mathrm{Prop} \to \tau$, $(\alpha \to \mathrm{Prop}) \to \tau$, itd. dla $\tau \neq \mathrm{Prop}$.

# Logika wyższego rzędu

## Dygresja

W zwykłej logice wyższego rzędu dozwolone są stałe typu $\mathrm{Prop} \to \tau$, $(\alpha \to \mathrm{Prop}) \to \tau$, itd. dla $\tau \neq \mathrm{Prop}$.

Jest oczywiste, że można stworzyć system logiki wyższego rzędu zezwalający na nieograniczoną rekurencję w *programach*, gdzie programy i logika nie będą pomieszane.

# Logika wyższego rzędu

## Dygresja

W zwykłej logice wyższego rzędu dozwolone są stałe typu $\mathrm{Prop} \to \tau$, $(\alpha \to \mathrm{Prop}) \to \tau$, itd. dla $\tau \neq \mathrm{Prop}$.

Jest oczywiste, że można stworzyć system logiki wyższego rzędu zezwalający na nieograniczoną rekurencję w *programach*, gdzie programy i logika nie będą pomieszane.

- Np. można mieć dziedzinę programów w zwykłej logice wyższego rzędu.

# Logika wyższego rzędu

## Dygresja

W zwykłej logice wyższego rzędu dozwolone są stałe typu $\mathrm{Prop} \to \tau$, $(\alpha \to \mathrm{Prop}) \to \tau$, itd. dla $\tau \neq \mathrm{Prop}$.

Jest oczywiste, że można stworzyć system logiki wyższego rzędu zezwalający na nieograniczoną rekurencję w *programach*, gdzie programy i logika nie będą pomieszane.

- Np. można mieć dziedzinę programów w zwykłej logice wyższego rzędu.
- Ale nas interesuje rekurencja *w* **logice wyższego rzędu**.

# Logika wyższego rzędu z rekurencją

## Poprawna wersja

- Jednak potrzebujemy typów.

# Logika wyższego rzędu z rekurencją

## Poprawna wersja

- Jednak potrzebujemy typów.
- Ale będą one obiektami *wewnątrz* systemu.

# Logika wyższego rzędu z rekurencją

## Poprawna wersja

- Jednak potrzebujemy typów.
- Ale będą one obiektami *wewnątrz* systemu.
- W tym kontekście lepiej myśleć o typach jako *zbiorach*.

# Logika wyższego rzędu z rekurencją

## Składnia

- Stałe: $\forall, \supset, \to, \mathrm{Prop}, \mathrm{Type} \in \Sigma$ oraz po jednej stałej dla każdego typu bazowego ($\mathcal{B} \subseteq \Sigma$).

# Logika wyższego rzędu z rekurencją

## Składnia

- Stałe: $\forall, \supset, \to, \mathrm{Prop}, \mathrm{Type} \in \Sigma$ oraz po jednej stałej dla każdego typu bazowego ($\mathcal{B} \subseteq \Sigma$).
- Zbiór termów $T$ beztypowego rachunku lambda ze stałymi z $\Sigma$.

# Logika wyższego rzędu z rekurencją

## Składnia

- Stałe: $\forall, \supset, \to, \mathrm{Prop}, \mathrm{Type} \in \Sigma$ oraz po jednej stałej dla każdego typu bazowego ($\mathcal{B} \subseteq \Sigma$).
- Zbiór termów $T$ beztypowego rachunku lambda ze stałymi z $\Sigma$.
- Konwencje:
  - $\varphi \supset \psi \equiv \supset \varphi \psi$,
  - $\forall x : \tau \,.\, \varphi \equiv \forall \tau (\lambda x \,.\, \varphi)$,
  - $\alpha \to \beta \equiv \to \alpha \beta$,
  - $\alpha : \beta \equiv \beta \alpha$.

# Logika wyższego rzędu z rekurencją

## Nieformalna intuicyjna semantyka

# Logika wyższego rzędu z rekurencją

## Nieformalna intuicyjna semantyka

- Uniwersum zawiera wszystko: programy, formuły, wartości logiczne, typy, liczby, wartości danych, “niezdefiniowane” wartości, programy pomieszane z formułami, . . . .

# Logika wyższego rzędu z rekurencją

## Nieformalna intuicyjna semantyka

- Uniwersum zawiera wszystko: programy, formuły, wartości logiczne, typy, liczby, wartości danych, "niezdefiniowane" wartości, programy pomieszane z formułami, ....
- Termy oznaczają elementy tego wielkiego uniwersum.

# Logika wyższego rzędu z rekurencją

## Nieformalna intuicyjna semantyka

- Uniwersum zawiera wszystko: programy, formuły, wartości logiczne, typy, liczby, wartości danych, "niezdefiniowane" wartości, programy pomieszane z formułami, ....
- Termy oznaczają elementy tego wielkiego uniwersum.
- Nie wiemy *a priori* do jakiej kategorii należy dany term.

# Logika wyższego rzędu z rekurencją

## Nieformalna intuicyjna semantyka

- Uniwersum zawiera wszystko: programy, formuły, wartości logiczne, typy, liczby, wartości danych, "niezdefiniowane" wartości, programy pomieszane z formułami, ....
- Termy oznaczają elementy tego wielkiego uniwersum.
- Nie wiemy *a priori* do jakiej kategorii należy dany term.
- Reguły wnioskowania pozwalają na wnioskowanie o typach termów.

# Logika wyższego rzędu z rekurencją

## Nieformalna intuicyjna semantyka

- $t : \mathrm{Prop}$ jest prawdą wtw $t$ jest prawdą lub fałszem,
- $\alpha : \mathrm{Type}$ jest prawdą wtw $\alpha$ jest typem,
- $t : \alpha$ jest prawdą wtw $t$ ma typ $\alpha$, zakładając, że $\alpha$ jest typem,
- $\forall x : \alpha.\varphi$ jest prawdą wtw $\alpha$ jest typem oraz dla dowolnego $t$ typu $\alpha$, $\varphi[x/t]$ jest prawdą,
- $\forall x : \alpha.\varphi$ jest fałszem wtw $\alpha$ jest typem oraz istnieje takie $t$ typu $\alpha$, że $\varphi[x/t]$ jest fałszem,
- $t_1 \vee t_2$ jest prawdą wtw $t_1$ jest prawdą lub $t_2$ jest prawdą,
- $t_1 \vee t_2$ jest fałszem wtw $t_1$ jest fałszem oraz $t_2$ jest fałszem,
- $t_1 \supset t_2$ jest prawdą wtw $t_1$ jest fałszem lub zarówno $t_1$ jak i $t_2$ są prawdą,
- $t_1 \supset t_2$ jest fałszem wtw $t_1$ jest prawdą oraz $t_2$ jest fałszem,
- $\neg t$ jest prawdą wtw $t$ jest fałszem,
- $\neg t$ jest fałszem wtw $t$ jest prawdą.

# Logika wyższego rzędu z rekurencją

## Reguły typowania

$$\frac{}{\Delta \vdash \bot : \mathrm{Prop}} \qquad \frac{}{\Delta \vdash \mathrm{Prop} : \mathrm{Type}}$$

$$\frac{\alpha \in \mathcal{B}}{\Delta \vdash \alpha : \mathrm{Type}} \qquad \frac{\Delta \vdash \alpha : \mathrm{Type}}{\Delta \vdash (\forall \alpha) : (\alpha \to \mathrm{Prop}) \to \mathrm{Prop}}$$

$$\frac{\Delta \vdash \varphi : \mathrm{Prop} \qquad \Delta, \varphi \vdash \psi : \mathrm{Prop}}{\Delta \vdash (\varphi \supset \psi) : \mathrm{Prop}}$$

$$\frac{\Delta \vdash (\varphi \supset \psi) : \mathrm{Prop}}{\Delta \vdash \varphi : \mathrm{Prop}}$$

$$\frac{\Delta \vdash \varphi}{\Delta \vdash \varphi : \mathrm{Prop}}$$

# Logika wyższego rzędu z rekurencją

## Reguły typowania

$$\to_i: \frac{\Delta \vdash \alpha : \mathrm{Type} \qquad \Delta, x : \alpha \vdash t : \beta \qquad x \notin FV(\Delta, \alpha, \beta)}{\Delta \vdash (\lambda x \,.\, t) : \alpha \to \beta}$$

$$\to_e: \frac{\Delta \vdash t_1 : \alpha \to \beta \qquad \Delta \vdash t_2 : \alpha}{\Delta \vdash t_1\, t_2 : \beta}$$

$$\to_t: \frac{\Delta \vdash \alpha : \mathrm{Type} \qquad \Delta \vdash \beta : \mathrm{Type}}{\Delta \vdash (\alpha \to \beta) : \mathrm{Type}}$$

# Logika wyższego rzędu z rekurencją

## Reguły dla spójników logicznych

$$\frac{}{\Delta, \varphi \vdash \varphi}$$

$$\supset_i: \frac{\Delta, \varphi \vdash \psi \qquad \Delta \vdash \varphi : \mathrm{Prop}}{\Delta \vdash \varphi \supset \psi} \qquad \supset_e: \frac{\Delta \vdash \varphi \supset \psi \quad \Delta \vdash \varphi}{\Delta \vdash \psi}$$

$$\forall_i : \frac{\Delta, x : \tau \vdash \varphi \qquad \Delta \vdash \tau : \mathrm{Type}}{\Delta \vdash \forall x : \tau \,.\, \varphi} \; x \notin FV(\Delta)$$

$$\forall_e : \frac{\Delta \vdash \forall x : \tau \,.\, \varphi \qquad \Delta \vdash t : \tau}{\Delta \vdash \varphi[x/t]}$$

$$\mathrm{conv} : \frac{\Delta \vdash \varphi \qquad \varphi =_\beta \psi}{\Delta \vdash \psi}$$

# Logika wyższego rzędu z rekurencją

## Reguły dla spójników logicznych

$$\frac{}{\Delta, \varphi \vdash \varphi}$$

$$\supset_i: \frac{\Delta, \varphi \vdash \psi \qquad \Delta \vdash \varphi : \mathrm{Prop}}{\Delta \vdash \varphi \supset \psi} \qquad \supset_e: \frac{\Delta \vdash \varphi \supset \psi \quad \Delta \vdash \varphi}{\Delta \vdash \psi}$$

$$\forall_i : \frac{\Delta, x : \tau \vdash \varphi \qquad \Delta \vdash \tau : \mathrm{Type}}{\Delta \vdash \forall x : \tau \,.\, \varphi} \; x \notin FV(\Delta)$$

$$\forall_e : \frac{\Delta \vdash \forall x : \tau \,.\, \varphi \qquad \Delta \vdash t : \tau}{\Delta \vdash \varphi[x/t]}$$

$$\mathrm{conv} : \frac{\Delta \vdash \varphi \qquad \varphi =_\beta \psi}{\Delta \vdash \psi}$$

Ten system jest niesprzeczny.

# Logika wyższego rzędu z rekurencją

Taki system (mniej więcej) został zaproponowany przez szkołę Haskella Curry'ego aby uratować program (illatywnej) logiki kombinatorycznej po odkryciu paradoksów

# Logika wyższego rzędu z rekurencją

Taki system (mniej więcej) został zaproponowany przez szkołę Haskella Curry'ego aby uratować program (illatywnej) logiki kombinatorycznej po odkryciu paradoksów, ale:

- do stosunkowo niedawna znano niewiele dowodów niesprzeczności,

# Logika wyższego rzędu z rekurencją

Taki system (mniej więcej) został zaproponowany przez szkołę Haskella Curry'ego aby uratować program (illatywnej) logiki kombinatorycznej po odkryciu paradoksów, ale:

- do stosunkowo niedawna znano niewiele dowodów niesprzeczności,
  - do lat 90-tych nie były znane żadne dowody niesprzeczności dla systemów wystarczająco silnych aby zinterpretować zwykłą logikę *pierwszego* rzędu,

# Logika wyższego rzędu z rekurencją

Taki system (mniej więcej) został zaproponowany przez szkołę Haskella Curry'ego aby uratować program (illatywnej) logiki kombinatorycznej po odkryciu paradoksów, ale:

- do stosunkowo niedawna znano niewiele dowodów niesprzeczności,
  - do lat 90-tych nie były znane żadne dowody niesprzeczności dla systemów wystarczająco silnych aby zinterpretować zwykłą logikę *pierwszego* rzędu,
- znane dowody niesprzeczności dla silnych systemów są dosyć skomplikowane.

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),
- $\lceil \mathrm{Prop} \rceil = \mathrm{Prop}$,

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),
- $\lceil \mathrm{Prop} \rceil = \mathrm{Prop}$,
- $\lceil \tau_1 \to \tau_2 \rceil = \lceil \tau_1 \rceil \to \lceil \tau_2 \rceil$ dla $\tau_1, \tau_2 \in \mathcal{T}$,

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),
- $\lceil \mathrm{Prop} \rceil = \mathrm{Prop}$,
- $\lceil \tau_1 \to \tau_2 \rceil = \lceil \tau_1 \rceil \to \lceil \tau_2 \rceil$ dla $\tau_1, \tau_2 \in \mathcal{T}$,
- $\lceil c \rceil = c$ jeśli $c$ jest stałą,

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),
- $\lceil \mathrm{Prop} \rceil = \mathrm{Prop}$,
- $\lceil \tau_1 \to \tau_2 \rceil = \lceil \tau_1 \rceil \to \lceil \tau_2 \rceil$ dla $\tau_1, \tau_2 \in \mathcal{T}$,
- $\lceil c \rceil = c$ jeśli $c$ jest stałą,
- $\lceil x \rceil = x$ jeśli $x$ jest zmienną,

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),
- $\lceil \mathrm{Prop} \rceil = \mathrm{Prop}$,
- $\lceil \tau_1 \to \tau_2 \rceil = \lceil \tau_1 \rceil \to \lceil \tau_2 \rceil$ dla $\tau_1, \tau_2 \in \mathcal{T}$,
- $\lceil c \rceil = c$ jeśli $c$ jest stałą,
- $\lceil x \rceil = x$ jeśli $x$ jest zmienną,
- $\lceil t_1 t_2 \rceil = \lceil t_1 \rceil \lceil t_2 \rceil$,

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),
- $\lceil \mathrm{Prop} \rceil = \mathrm{Prop}$,
- $\lceil \tau_1 \to \tau_2 \rceil = \lceil \tau_1 \rceil \to \lceil \tau_2 \rceil$ dla $\tau_1, \tau_2 \in \mathcal{T}$,
- $\lceil c \rceil = c$ jeśli $c$ jest stałą,
- $\lceil x \rceil = x$ jeśli $x$ jest zmienną,
- $\lceil t_1 t_2 \rceil = \lceil t_1 \rceil \lceil t_2 \rceil$,
- $\lceil \lambda x : \tau \,.\, t \rceil = \lambda x \,.\, \lceil t \rceil$,

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),
- $\lceil \mathrm{Prop} \rceil = \mathrm{Prop}$,
- $\lceil \tau_1 \to \tau_2 \rceil = \lceil \tau_1 \rceil \to \lceil \tau_2 \rceil$ dla $\tau_1, \tau_2 \in \mathcal{T}$,
- $\lceil c \rceil = c$ jeśli $c$ jest stałą,
- $\lceil x \rceil = x$ jeśli $x$ jest zmienną,
- $\lceil t_1 t_2 \rceil = \lceil t_1 \rceil \lceil t_2 \rceil$,
- $\lceil \lambda x : \tau \,.\, t \rceil = \lambda x \,.\, \lceil t \rceil$,
- $\lceil \varphi \supset \psi \rceil = \lceil \varphi \rceil \supset \lceil \psi \rceil$,

# Związek ze zwykłą logiką

## Translacja

Definiujemy translację $\lceil - \rceil$ z języka zwykłej logiki wyższego rzędu do języka powyższego systemu.

- $\lceil \tau \rceil = \tau$ jeśli $\tau$ jest typem bazowym ($\tau \in \mathcal{B}$),
- $\lceil \mathrm{Prop} \rceil = \mathrm{Prop}$,
- $\lceil \tau_1 \to \tau_2 \rceil = \lceil \tau_1 \rceil \to \lceil \tau_2 \rceil$ dla $\tau_1, \tau_2 \in \mathcal{T}$,
- $\lceil c \rceil = c$ jeśli $c$ jest stałą,
- $\lceil x \rceil = x$ jeśli $x$ jest zmienną,
- $\lceil t_1 t_2 \rceil = \lceil t_1 \rceil \lceil t_2 \rceil$,
- $\lceil \lambda x : \tau \,.\, t \rceil = \lambda x \,.\, \lceil t \rceil$,
- $\lceil \varphi \supset \psi \rceil = \lceil \varphi \rceil \supset \lceil \psi \rceil$,
- $\lceil \forall x : \tau \,.\, \varphi \rceil = \forall x : \lceil \tau \rceil \,.\, \lceil \varphi \rceil$.

# Związek ze zwykłą logiką

## Funkcja kontekstu

Funkcja $\Gamma$ ze zbiorów termów zwykłej logiki wyższego rzędu do zbiorów termów w naszym systemie.

# Związek ze zwykłą logiką

## Funkcja kontekstu

Funkcja $\Gamma$ ze zbiorów termów zwykłej logiki wyższego rzędu do zbiorów termów w naszym systemie.
Dla zbioru termów $\Delta$, zbiór $\Gamma(\Delta)$ składa się z:

- $x : \lceil \tau \rceil$ dla wszystkich typów $\tau$ i zmiennych $x \in FV(\Delta)$ typu $\tau$,

# Związek ze zwykłą logiką

## Funkcja kontekstu

Funkcja $\Gamma$ ze zbiorów termów zwykłej logiki wyższego rzędu do zbiorów termów w naszym systemie.
Dla zbioru termów $\Delta$, zbiór $\Gamma(\Delta)$ składa się z:

- $x : \lceil \tau \rceil$ dla wszystkich typów $\tau$ i zmiennych $x \in FV(\Delta)$ typu $\tau$,
- $c : \lceil \tau \rceil$ dla wszystkich typów $\tau$ i stałych $c$ typu $\tau$,

# Związek ze zwykłą logiką

## Funkcja kontekstu

Funkcja $\Gamma$ ze zbiorów termów zwykłej logiki wyższego rzędu do zbiorów termów w naszym systemie.
Dla zbioru termów $\Delta$, zbiór $\Gamma(\Delta)$ składa się z:

- $x : \lceil \tau \rceil$ dla wszystkich typów $\tau$ i zmiennych $x \in FV(\Delta)$ typu $\tau$,
- $c : \lceil \tau \rceil$ dla wszystkich typów $\tau$ i stałych $c$ typu $\tau$,
- $\tau : \mathrm{Type}$ dla wszystkich $\tau \in \mathcal{B}$,

# Związek ze zwykłą logiką

## Funkcja kontekstu

Funkcja $\Gamma$ ze zbiorów termów zwykłej logiki wyższego rzędu do zbiorów termów w naszym systemie.
Dla zbioru termów $\Delta$, zbiór $\Gamma(\Delta)$ składa się z:

- $x : \lceil \tau \rceil$ dla wszystkich typów $\tau$ i zmiennych $x \in FV(\Delta)$ typu $\tau$,
- $c : \lceil \tau \rceil$ dla wszystkich typów $\tau$ i stałych $c$ typu $\tau$,
- $\tau : \mathrm{Type}$ dla wszystkich $\tau \in \mathcal{B}$,
- $y : \tau$ dla wszystkich $\tau \in \mathcal{B}$ i pewnej zmiennej $y$ typu $\tau$ takiej, że $y \notin FV(\Delta)$.

# Związek ze zwykłą logiką

## Poprawność i pełność translacji

### Twierdzenie

*Jeśli* $\Delta \vdash_{\mathrm{PRED}\omega} \varphi$ *to* $\Gamma(\Delta \cup \{\varphi\}), \lceil\Delta\rceil \vdash_{\mathcal{I}} \lceil\varphi\rceil$.

# Związek ze zwykłą logiką

## Poprawność i pełność translacji

### Twierdzenie

*Jeśli* $\Delta \vdash_{\mathrm{PRED}\omega} \varphi$ *to* $\Gamma(\Delta \cup \{\varphi\}), \lceil\Delta\rceil \vdash_{\mathcal{I}} \lceil\varphi\rceil$.

### Hipoteza

*Jeśli* $\Gamma(\Delta \cup \{\varphi\}), \lceil\Delta\rceil \vdash_{\mathcal{I}} \lceil\varphi\rceil$ *to* $\Delta \vdash_{\mathrm{PRED}\omega} \varphi$.

# Związek ze zwykłą logiką

## Poprawność i pełność translacji

### Twierdzenie

*Jeśli* $\Delta \vdash_{\mathrm{PRED}\omega} \varphi$ *to* $\Gamma(\Delta \cup \{\varphi\}), \lceil\Delta\rceil \vdash_{\mathcal{I}} \lceil\varphi\rceil$.

### Hipoteza

*Jeśli* $\Gamma(\Delta \cup \{\varphi\}), \lceil\Delta\rceil \vdash_{\mathcal{I}} \lceil\varphi\rceil$ *to* $\Delta \vdash_{\mathrm{PRED}\omega} \varphi$.

### Twierdzenie

$$\Delta \vdash_{\mathrm{FOL}} \varphi \quad \text{wtw} \quad \Gamma(\Delta \cup \{\varphi\}), \lceil\Delta\rceil \vdash_{\mathcal{I}} \lceil\varphi\rceil$$

# Rozszerzenia

- Logika klasyczna.

$$\frac{}{\Delta \vdash \forall p : \mathrm{Prop} \,.\, ((p \supset \bot) \supset \bot) \supset p}$$

# Rozszerzenia

- Logika klasyczna.

$$\frac{}{\Delta \vdash \forall p : \mathrm{Prop} \,.\, ((p \supset \bot) \supset \bot) \supset p}$$

- Operator wyboru.

# Rozszerzenia

- Logika klasyczna.

$$\frac{}{\Delta \vdash \forall p : \mathrm{Prop} \,.\, ((p \supset \bot) \supset \bot) \supset p}$$

- Operator wyboru.
- Ekstensjonalność zwn. równość Leibniza.

# Rozszerzenia

- Logika klasyczna.

$$\frac{}{\Delta \vdash \forall p : \mathrm{Prop}\,.\,((p \supset \bot) \supset \bot) \supset p}$$

- Operator wyboru.
- Ekstensjonalność zwn. równość Leibniza.
- Funkcja “If”.

# Rozszerzenia

- Logika klasyczna.

$$\frac{}{\Delta \vdash \forall p : \mathrm{Prop} \,.\, ((p \supset \bot) \supset \bot) \supset p}$$

- Operator wyboru.
- Ekstensjonalność zwn. równość Leibniza.
- Funkcja “If”.
- Podtypy predykatowe.

# Rozszerzenia

- Logika klasyczna.

$$\frac{}{\Delta \vdash \forall p : \mathrm{Prop}\,.\,((p \supset \bot) \supset \bot) \supset p}$$

- Operator wyboru.
- Ekstensjonalność zwn. równość Leibniza.
- Funkcja “If”.
- Podtypy predykatowe.
- Pewna klasa **typów indukcyjnych**.

# Indukcja

Mamy teraz logikę wyższego rzędu z rekurencją

# Indukcja

Mamy teraz logikę wyższego rzędu z rekurencją, jednak kluczowym pytaniem jest:

Jak wiele możemy wnioskować o termach, których typ nie został jeszcze ustalony?

# Indukcja

Mamy teraz logikę wyższego rzędu z rekurencją, jednak kluczowym pytaniem jest:

Jak wiele możemy wnioskować o termach, których typ nie został jeszcze ustalony?

- Dla podstawowego systemu odpowiedź brzmi: niewiele.

# Indukcja

Mamy teraz logikę wyższego rzędu z rekurencją, jednak kluczowym pytaniem jest:

Jak wiele możemy wnioskować o termach, których typ nie został jeszcze ustalony?

- Dla podstawowego systemu odpowiedź brzmi: niewiele.
- Potrzebujemy zasady indukcji którą można stosować do *dowolnych* termów.

# Indukcja

Mamy teraz logikę wyższego rzędu z rekurencją, jednak kluczowym pytaniem jest:

Jak wiele możemy wnioskować o termach, których typ nie został jeszcze ustalony?

- Dla podstawowego systemu odpowiedź brzmi: niewiele.
- Potrzebujemy zasady indukcji którą można stosować do *dowolnych* termów.
- Nasz system pozwala na dowolne typy indukcyjne bez argumentów funkcyjnych, ale tutaj dla prostoty ograniczmy się do liczb naturalnych.

# Indukcja

## Dla liczb naturalnych

Potrzebujemy zasady indukcji, którą można stosować do **dowolnych** termów.

# Indukcja

## Dla liczb naturalnych

Potrzebujemy zasady indukcji, którą można stosować do **dowolnych** termów.

- Taka jest zła:

$$\forall f : \mathrm{Nat} \to \mathrm{Prop}\,.\,((f\,0 \wedge (\forall x : \mathrm{Nat}\,.\,f\,x \supset f\,(s\,x))) \supset \forall x : \mathrm{Nat}\,.\,f\,x)$$

# Indukcja

## Dla liczb naturalnych

Potrzebujemy zasady indukcji, którą można stosować do **dowolnych** termów.

- Taka jest zła:

$$\forall f : \mathrm{Nat} \to \mathrm{Prop} \,.\, ((f\,0 \wedge (\forall x : \mathrm{Nat} \,.\, f\,x \supset f\,(s\,x))) \supset \forall x : \mathrm{Nat} \,.\, f\,x)$$

- Taka jest dobra:

$$n_i : \frac{\Delta \vdash t\,0 \qquad \Delta, x : \mathrm{Nat}, t\,x \vdash t\,(s\,x) \qquad x \notin FV(\Delta, t)}{\Delta \vdash \forall x : \mathrm{Nat} \,.\, t\,x}$$

# Wnioskowanie o typach przez indukcję

## Twierdzenie (nieformalne sformułowanie)

*Niech $f$ będzie taką funkcją, że istnieje miara (w liczbach naturalnych) na jej argumentach dla której można pokazać, że w skończonej liczbie wyczerpujących przypadków maleje ona z każdym wywołaniem rekurencyjnym $f$. Wtedy jeśli przy założeniu, że $f$ ma typ $\beta$ można pokazać, że ciało $f$ ma typ $\beta$, to $f$ ma typ $\beta$.*

# Wnioskowanie o typach przez indukcję

## Twierdzenie (nieformalne sformułowanie)

*Niech $f$ będzie taką funkcją, że istnieje miara (w liczbach naturalnych) na jej argumentach dla której można pokazać, że w skończonej liczbie wyczerpujących przypadków maleje ona z każdym wywołaniem rekurencyjnym $f$. Wtedy jeśli przy założeniu, że $f$ ma typ $\beta$ można pokazać, że ciało $f$ ma typ $\beta$, to $f$ ma typ $\beta$.*

## Dowód.

Proste użycie zasady indukcji. □

# Wniosek

- Jedyną własnością naszego systemu istotnie różną od zwykłej logiki jest konieczność jawnego wyprowadzania osądów o typach jako dodatkowych przesłanek w kilku regułach wnioskowania.

# Wniosek

- Jedyną własnością naszego systemu istotnie różną od zwykłej logiki jest konieczność jawnego wyprowadzania osądów o typach jako dodatkowych przesłanek w kilku regułach wnioskowania.
- Z poprzednich twierdzeń wynika, że w implementacji naszej logiki ręczne wyprowadzanie typów może być konieczne jedynie gdy rozważana funkcja:

# Wniosek

- Jedyną własnością naszego systemu istotnie różną od zwykłej logiki jest konieczność jawnego wyprowadzania osądów o typach jako dodatkowych przesłanek w kilku regułach wnioskowania.
- Z poprzednich twierdzeń wynika, że w implementacji naszej logiki ręczne wyprowadzanie typów może być konieczne jedynie gdy rozważana funkcja:
  - nie jest typowalna w typach prostych,

# Wniosek

- Jedyną własnością naszego systemu istotnie różną od zwykłej logiki jest konieczność jawnego wyprowadzania osądów o typach jako dodatkowych przesłanek w kilku regułach wnioskowania.
- Z poprzednich twierdzeń wynika, że w implementacji naszej logiki ręczne wyprowadzanie typów może być konieczne jedynie gdy rozważana funkcja:
  - nie jest typowalna w typach prostych,
  - nie jest zdefiniowana przez rekursję strukturalną,

# Wniosek

- Jedyną własnością naszego systemu istotnie różną od zwykłej logiki jest konieczność jawnego wyprowadzania osądów o typach jako dodatkowych przesłanek w kilku regułach wnioskowania.
- Z poprzednich twierdzeń wynika, że w implementacji naszej logiki ręczne wyprowadzanie typów może być konieczne jedynie gdy rozważana funkcja:
  - nie jest typowalna w typach prostych,
  - nie jest zdefiniowana przez rekursję strukturalną,
  - nie możemy łatwo znaleźć malejącej miary,

# Wniosek

- Jedyną własnością naszego systemu istotnie różną od zwykłej logiki jest konieczność jawnego wyprowadzania osądów o typach jako dodatkowych przesłanek w kilku regułach wnioskowania.
- Z poprzednich twierdzeń wynika, że w implementacji naszej logiki ręczne wyprowadzanie typów może być konieczne jedynie gdy rozważana funkcja:
  - nie jest typowalna w typach prostych,
  - nie jest zdefiniowana przez rekursję strukturalną,
  - nie możemy łatwo znaleźć malejącej miary,
  - zatem nie może być (w bezpośredni sposób) zdefiniowana w prostych rozszerzeniach logiki wyższego rzędu o terminujące definicje rekurencyjne.

# Wyjaśnienie

Zasygnalizowane wcześniej twierdzenia mają konstruktywne dowody (oprócz twierdzenia o niesprzeczności i pełności translacji FOL). Zatem istnieje algorytm, który przekształca odpowiednie adnotacje o typach i/lub mierze na wyprowadzenia w naszej logice.

# Bibliografia



Czajka, Ł.:
Higher-order illative combinatory logic.
Journal of Symbolic Logic (2013) Zaakceptowane.


Czajka, Ł.:
Partiality and recursion in higher-order logic.
W: Foundations of Software Science and Computation Structures, FOSSACS 2013, Proceedings. Volume 7794 of LNCS, Springer (2013) 177-192


Czajka, Ł.:
A semantic approach to illative combinatory logic.
W: Computer Science Logic, CSL 2011, Proceedings. Volume 12 of LIPIcs, Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2011) 174–188