Linguaggi di Programmazione

Roberta Gori

HOFL Sintassi e Tipi - 7.1

HOFL pre-termini
(linguaggio funzionale di ordine superiore)

Sintassi HOFL

$n \in \mathbb{Z}$

op $\in \{+, -, \times\}$ $\quad$ $\textbf{if } t = 0 \textbf{ then } t_0 \textbf{ else } t_1$

$x \in Ide$

$$
\begin{array}{rll}
t & ::= & x \mid n \mid t_0 \ \mathsf{op}\ t_1 \mid \textbf{if } t \textbf{ then } t_0 \textbf{ else } t_1 \\
& \mid & (t_0, t_1) \mid \textbf{fst}(t) \mid \textbf{snd}(t) \\
& \mid & \lambda x.\ t \mid t_0\ t_1 \\
& \mid & \textbf{rec } x.\ t
\end{array}
$$

standard

coppie proiezioni

abstrazione applicazione

ricorsione

let $x = t$ in x

can contain x

Esercizio

$$\mathbf{rec}\ f.\ \lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

indovinate il significato del precedente pre-termine

fattoriale

Esercizio

$$\begin{aligned}\textbf{rec}\ rep.\ \lambda n.\ \lambda f.\ \lambda x.\ &\textbf{if}\ n\ \textbf{then}\ x\\ &\textbf{else}\ f\ (rep\ (n-1)\ f\ x)\end{aligned}$$

indovinate il significato del precedente pre-termine

$$rep\ n\ f\ x = f^n\ x$$

Esercizio

$$\lambda x. \left(\left(\begin{array}{l} \textbf{rec } f.\ \lambda y.\ \textbf{if } (x-y) \textbf{ then } 0 \\ \qquad\qquad\quad \textbf{else if } (x+y) \textbf{ then } 1 \\ \qquad\qquad\quad \textbf{else } f\ (y+1) \end{array} \right) 0 \right)$$

indovinate il significato del precedente pre-termine

maggiore o uguale a 0

Esercizio (da consegnare)

assumiamo $true = 0$

$false =$ qls $n \neq 0$

riempire al posto dei puntini (in HOFL)

$$
\begin{aligned}
or &\triangleq \lambda n.\ \lambda m.\ \cdots \\
and &\triangleq \lambda n.\ \lambda m.\ \cdots \\
not &\triangleq \lambda m.\ \cdots \\
implies &\triangleq \lambda n.\ \lambda m.\ \cdots \\
iff &\triangleq \lambda n.\ \lambda m.\ \cdots
\end{aligned}
$$

Pre-termini

$$
\begin{array}{rcl}
t & ::= & x \mid n \mid t_0 \ \mathsf{op}\ t_1 \mid \mathbf{if}\ t\ \mathbf{then}\ t_0\ \mathbf{else}\ t_1 \\
 & \mid & (t_0, t_1) \mid \mathbf{fst}(t) \mid \mathbf{snd}(t) \\
 & \mid & \lambda x.\ t \mid t_0\ t_1 \\
 & \mid & \mathbf{rec}\ x.\ t
\end{array}
$$

Perche sono chiamati pre-termini?

$x + 1$ ✓	✗ $1 + (0, 5)$
$\mathbf{if}\ x\ \mathbf{then}\ x + 1\ \mathbf{else}\ x - 1$ ✓	✗ $2 \times \lambda x.\ x$
$(0, \lambda x.\ x)$ ✓	✗ $3\ \lambda x.\ x + 1$
$\mathbf{fst}(0, \lambda x.\ x)$ ✓	✗ $\mathbf{fst}(3)$
$(\lambda x.\ x + 1)\ 3$ ✓	✗ $\mathbf{if}\ x\ \mathbf{then}\ \lambda x.\ x\ \mathbf{else}\ (x, x)$
$\mathbf{rec}\ f.\ \lambda x.\ x + (f\ 0)$ ✓	✗ $\mathbf{rec}\ f.\ \lambda x.\ f + x$

abbiamo bisogno di un sistema di tipi

tipi HOFL

Tipi

$$\begin{array}{rcl} t & ::= & x \mid n \mid t_0 \ \mathsf{op}\ t_1 \mid \mathbf{if}\ t\ \mathbf{then}\ t_0\ \mathbf{else}\ t_1 \\ & \mid & (t_0, t_1) \mid \mathbf{fst}(t) \mid \mathbf{snd}(t) \\ & \mid & \lambda x.\ t \mid t_0\ t_1 \\ & \mid & \mathbf{rec}\ x.\ t \end{array}$$

quali tipi?

infinite combinazioni!

coppie

funzioni

int

$int * int$

$int \to int$

$int * (int \to int)$

$(int * int) \to int$

$int * (int * int)$

$(int \to int) \to int$

$(int \to int) * (int \to int)$

$(int \to int) \to (int \to (int * int))$

Sintassi dei tipi

$$\tau \ ::= \ int \ \mid \ \tau_0 * \tau_1 \ \mid \ \tau_0 \to \tau_1$$

$\mathcal{T}$

insieme di tutti i tipi

perche' non le liste?

per la stessa ragione per cui evitiamo la divisione
testa e coda non sono funzioni totali

assumiamo variabili tipate

$$Ide = \{Ide_\tau\}_{\tau \in \mathcal{T}}$$

$$\widehat{\cdot} : Ide \to \mathcal{T}$$

$\widehat{x}$ denota il tipo di x

Type judgements

si legge “ha tipo”

formula: $t : \tau$

sono assegnati ai pre-termini
usando un insieme di regole di inferenza
(induzione strutturale della sintassi HOFL)

Sistema di tipi

$$\frac{}{x : \widehat{x}} \qquad \frac{}{n : int} \qquad \frac{t_0 : int \quad t_1 : int}{t_0 \ \mathsf{op}\ t_1 : int} \qquad \frac{t : int \quad t_0 : \tau \quad t_1 : \tau}{\mathbf{if}\ t\ \mathbf{then}\ t_0\ \mathbf{else}\ t_1 : \tau}$$

$$\frac{t_0 : \tau_0 \quad t_1 : \tau_1}{(t_0, t_1) : \tau_0 * \tau_1} \qquad \frac{t : \tau_0 * \tau_1}{\mathbf{fst}(t) : \tau_0} \qquad \frac{t : \tau_0 * \tau_1}{\mathbf{snd}(t) : \tau_1}$$

$$\frac{x : \tau_0 \quad t : \tau_1}{\lambda x.\ t : \tau_0 \to \tau_1} \qquad \frac{t_1 : \tau_0 \to \tau_1 \quad t_0 : \tau_0}{t_1\ t_0 : \tau_1}$$

$$\frac{x : \tau \quad t : \tau}{\mathbf{rec}\ x.\ t : \tau}$$

Termini ben formati

$$
\begin{array}{rcl}
t & ::= & x \mid n \mid t_0 \ \mathsf{op}\ t_1 \mid \mathbf{if}\ t\ \mathbf{then}\ t_0\ \mathbf{else}\ t_1 \\
& \mid & (t_0, t_1) \mid \mathbf{fst}(t) \mid \mathbf{snd}(t) \\
& \mid & \lambda x.\ t \mid t_0\ t_1 \\
& \mid & \mathbf{rec}\ x.\ t
\end{array}
$$

$$\tau \ ::= \ int \mid \tau_0 * \tau_1 \mid \tau_0 \to \tau_1$$

$\mathcal{T}$ insieme di tutti i tipi

un pre-termine t e' ben formato se $\exists \tau \in \mathcal{T}.\ t : \tau$

cioe' se possiamo assegnargli un tipo
anche detto ben tipato *o tipabile*

T_τ insieme di tutti i termini ben formati di tipo τ

Controllo dei tipi

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

$$\frac{}{fact : int \to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati dal programmatore)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

$$\frac{f : int \to int \qquad \lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int \to int}{fact : int \to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

$$\dfrac{\dfrac{\widehat{f} = int \to int}{f : int \to int} \qquad \dfrac{x : int \qquad \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int}{\lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int \to int}}{fact : int \to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

$$\dfrac{\dfrac{\widehat{f} = int \to int}{f : int \to int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{x : int \qquad 1 : int \qquad (x \times (f(x-1))) : int}{\mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int}}{\lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int \to int}}{fact : int \to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

$$\dfrac{\dfrac{\widehat{f} = int \to int}{f : int \to int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{}{1 : int} \qquad \dfrac{x : int \qquad f(x-1) : int}{(x \times (f(x-1))) : int}}{\mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int}}{\lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int \to int}}{fact : int \to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

$$\dfrac{\dfrac{\widehat{f} = int \to int}{f : int \to int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{}{1 : int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{f : int \to int \qquad x - 1 : int}{f(x-1) : int}}{(x \times (f(x-1))) : int}}{\mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int}}{\lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int \to int}}{fact : int \to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

$$\dfrac{\dfrac{\widehat{f} = int \to int}{f : int \to int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{}{1 : int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{\dfrac{\widehat{f} = int \to int}{f : int \to int} \qquad \dfrac{x : int \qquad 1 : int}{x - 1 : int}}{f(x-1) : int}}{(x \times (f(x-1))) : int}}{\mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int}}{\lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int \to int}}{fact : int \to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

$$\dfrac{\dfrac{\widehat{f} = int \to int}{f : int \to int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{}{1 : int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{\dfrac{\widehat{f} = int \to int}{f : int \to int} \qquad \dfrac{\dfrac{\widehat{x} = int}{x : int} \qquad \dfrac{}{1 : int}}{x - 1 : int}}{f(x-1) : int}}{(x \times (f(x-1))) : int}}{\mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int}}{\lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ (x \times (f(x-1))) : int \to int}}{fact : int \to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati dal programmatore)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int \to int} .\lambda \underbrace{x}_{int} .\mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (\ f\ (x-1))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int\to int} .\lambda \underbrace{x}_{int} .\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (\ f\ (x - 1))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int \to int}.\lambda \underbrace{x}_{int}.\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ x \times (\ f\ (x-1))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int \to int}\ .\lambda \underbrace{x}_{int}\ .\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{x}_{int} \times (\ f\ \ (\ x - 1\ ))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int \to int} .\lambda \underbrace{x}_{int} .\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{x}_{int} \times (\underbrace{f}_{int \to int} \ (x-1))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int \to int} .\lambda \underbrace{x}_{int} .\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{x}_{int} \times (\underbrace{f}_{int \to int} (\underbrace{x}_{int} - 1))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int \to int} .\lambda \underbrace{x}_{int} .\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{x}_{int} \times (\underbrace{f}_{int \to int} (\underbrace{x}_{int} - \underbrace{1}_{int}))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int\to int}.\ \lambda \underbrace{x}_{int}.\ \mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{x}_{int} \times (\underbrace{f}_{int\to int}\ \underbrace{(\underbrace{x}_{int} - \underbrace{1}_{int})}_{int}))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int\to int} .\lambda \underbrace{x}_{int} .\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{x}_{int} \times (\underbrace{\underbrace{f}_{int\to int}\ (\underbrace{\underbrace{x}_{int} - \underbrace{1}_{int}}_{int}}_{int}))$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int \to int} .\lambda \underbrace{x}_{int} .\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{x \times (\underbrace{\underbrace{f}_{int\to int}\ (\underbrace{\underbrace{x}_{int} - \underbrace{1}_{int}}_{int})}_{int})}_{int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x - 1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int \to int} .\lambda \underbrace{x}_{int} . \underbrace{\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{\underbrace{x}_{int} \times (\underbrace{\underbrace{f}_{int \to int}\ (\underbrace{\underbrace{x}_{int} - \underbrace{1}_{int}}_{int})}_{int})}_{int}}_{int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int\to int} .\underbrace{\lambda \underbrace{x}_{int} .\underbrace{\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{\underbrace{x}_{int} \times (\underbrace{\underbrace{f}_{int\to int}\ (\underbrace{\underbrace{x}_{int} - \underbrace{1}_{int}}_{int})}_{int})}_{int}}_{int}}_{int\to int}$$

Esempio

le variabili sono etichettate con tipi (dichiarati)
deduciamo il tipo dei termini per ricorsione strutturale

$$fact \triangleq \mathbf{rec}\ f : int \to int.\ \lambda x : int.\ \mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ x \times (f\ (x-1))$$

scritto in maniera più semplice

$$fact \stackrel{\text{def}}{=} \mathbf{rec}\ \underbrace{f}_{int\to int} \underbrace{.\lambda \underbrace{x}_{int}.\underbrace{\mathbf{if}\ \underbrace{x}_{int}\ \mathbf{then}\ \underbrace{1}_{int}\ \mathbf{else}\ \underbrace{\underbrace{x}_{int}\times(\underbrace{\underbrace{f}_{int\to int}\ (\underbrace{\underbrace{x}_{int}-\underbrace{1}_{int}}_{int})}_{int})}_{int}}_{int}}_{int\to int} \qquad : int \to int$$

Inferenza di tipi

Esempio

i tipi delle variabili non sono dati
le regole di tipo sono usate per derivare vincoli di tipo (equazioni di tipo)
le cui soluzioni (tramite unificazione) definiscono il tipo principale

$$t \stackrel{\text{def}}{=} \mathbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

intuivamente

$$t\ 0 \equiv (0, (t\ 2)) \equiv (0, (2, (t\ 4))) \equiv \cdots \equiv (0, (2, (4, \ldots)))$$

sequenza di tutti i numeri pari

possiamo digitare sequenze di interi di lunghezza fissa
non abbiamo un tipo per sequenze di lunghezza qualsiasi/infinita

Esempio

i tipi delle variabili non sono dati
le regole di tipo sono usate per derivare vincoli di tipo (equazioni di tipo)
le cui soluzioni (tramite unificazione) definiscono il tipo principale

$$t \stackrel{\text{def}}{=} \mathbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

Haskell

```
Prelude> let p x = (x, p (x+2))

<interactive>:…:5: error:
    • Occurs check: cannot construct the infinite type: b ~ (t, b)
      Expected type: t -> b
        Actual type: t -> (t, b)
    • Relevant bindings include
        p :: t -> b (bound at <interactive>:…:5)
```

Esempio

$$t \stackrel{\text{def}}{=} \textbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

$$t = \textbf{rec}\ p.\ \lambda x.\ (x,(p\ (x+2))) : \tau \quad \nwarrow_{\widehat{p}=\tau} \quad \lambda x.\ (x,(p\ (x+2))) : \tau$$

$$\nwarrow_{\tau=\tau_1\to\tau_2,\ \widehat{x}=\tau_1} \quad (x,(p\ (x+2))) : \tau_2$$

$$\nwarrow_{\tau_2=\tau_3*\tau_4} \quad x : \tau_3,\ (p\ (x+2)) : \tau_4$$

$$\nwarrow_{\widehat{x}=\tau_3} \quad (p\ (x+2)) : \tau_4$$

$$\nwarrow \quad p : \tau_5 \to \tau_4,\ (x+2) : \tau_5$$

$$\nwarrow_{\widehat{p}=\tau_5\to\tau_4} \quad (x+2) : \tau_5$$

$$\nwarrow_{\tau_5=int} \quad x : int,\ 2 : int$$

$$\nwarrow^{*}_{\widehat{x}=int} \quad \square$$

$$\left.\begin{array}{l} \widehat{x} = \tau_1 \\ \widehat{x} = \tau_3 \\ \widehat{x} = int \end{array}\right\} \tau_1 = \tau_3 = int$$

$$\left.\begin{array}{l} \widehat{p} = \tau = \tau_1 \to \tau_2 \\ \widehat{p} = \tau_5 \to \tau_4 \end{array}\right\} \begin{array}{l} \tau_1 = \tau_5 = int \\ \tau_2 = \tau_4 \end{array}$$

$$\left.\begin{array}{l} \tau_2 = \tau_4 \\ \tau_2 = \tau_3 * \tau_4 \end{array}\right\} \text{fail! (occur check)}$$

Esempio

$$t \stackrel{\text{def}}{=} \mathbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

scritto in maniera più semplice

$$t = \mathbf{rec}\ p.\quad \lambda\ x\ .\ (\ x\ , (\quad p \quad (\ x + \underbrace{2}_{int})))$$

Esempio

$$t \stackrel{\text{def}}{=} \textbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

scritto in maniera più semplice

$$t = \textbf{rec}\ p.\quad \lambda\ x\ .\ (\ x\ , (\quad p \quad (\underbrace{x}_{int} + \underbrace{2}_{int})))$$

Esempio

$$t \stackrel{\text{def}}{=} \textbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

scritto in maniera più semplice

$$t = \textbf{rec}\ p. \quad \lambda \underbrace{x}_{int}.\ (\underbrace{x}_{int}, (\quad p \quad (\underbrace{x}_{int} + \underbrace{2}_{int})))$$

Esempio

$$t \stackrel{\text{def}}{=} \textbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

scritto in maniera più semplice

$$t = \textbf{rec}\ p.\quad \lambda \underbrace{x}_{int}.\ (\underbrace{x}_{int}, (\quad p \quad \underbrace{(\underbrace{x}_{int} + \underbrace{2}_{int})}_{int})))$$

Esempio

$$t \stackrel{\text{def}}{=} \mathbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

scritto in maniera più semplice

$$t = \mathbf{rec}\ p.\quad \lambda \underbrace{x}_{int} .\ (\underbrace{x}_{int}, (\underbrace{p}_{int \to \tau_4}\ \underbrace{(\underbrace{x}_{int} + \underbrace{2}_{int})}_{int})))$$

Esempio

$$t \stackrel{\text{def}}{=} \mathbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

scritto in maniera più semplice

$$t = \mathbf{rec}\ \underbrace{p}_{int \to \tau_4}.\quad \lambda\ \underbrace{x}_{int}\ .\ (\ \underbrace{x}_{int}, (\ \underbrace{p}_{int \to \tau_4}\quad \underbrace{(\ \underbrace{x}_{int} + \underbrace{2}_{int}\ )}_{int}))$$

Esempio

$$t \stackrel{\text{def}}{=} \mathbf{rec}\ p.\ \lambda x.\ (x,(p\ (x+2)))$$

scritto in maniera più semplice

$$t = \mathbf{rec}\ \underbrace{p}_{int\to\tau_4}.\ \ \lambda\ \underbrace{x}_{int}\ .\ (\underbrace{x}_{int},\underbrace{(\ \underbrace{p}_{int\to\tau_4}\ \ \underbrace{(\underbrace{x}_{int}+\underbrace{2}_{int})}_{int}}_{\tau_4})))$$

Esempio

$$t \stackrel{\text{def}}{=} \textbf{rec}\ p.\ \lambda x.\ (x, (p\ (x+2)))$$

scritto in maniera più semplice

$$t = \textbf{rec}\ \underbrace{p}_{int\to\tau_4}.\ \ \lambda\ \underbrace{x}_{int}\ .\ \underbrace{(\underbrace{x}_{int}, \underbrace{(\ \underbrace{p}_{int\to\tau_4}\ \ \underbrace{(\underbrace{x}_{int} + \underbrace{2}_{int})}_{int})}_{\tau_4})}_{int*\tau_4}$$

Esempio

$$t \stackrel{\text{def}}{=} \mathbf{rec}\ p.\ \lambda x.\ (x,(p\ (x+2)))$$

scritto in maniera più semplice

$$t = \mathbf{rec}\ \underbrace{p}_{int\to\tau_4}.\ \underbrace{\lambda\ \underbrace{x}_{int}\ .\ \underbrace{(\underbrace{x}_{int},\underbrace{(\ \underbrace{p}_{int\to\tau_4}\ \ \underbrace{(\underbrace{x}_{int}+\underbrace{2}_{int})}_{int}))}_{\tau_4})}_{int*\tau_4}}_{(int\to(int*\tau_4))=(int\to\tau_4)\ \Rightarrow\ \tau_4=(int*\tau_4)}$$

fallisce (occur check)

Esercizio

$$\begin{aligned}\textbf{rec } rep.\ \lambda n.\ \lambda f.\ \lambda x.\ \textbf{if } n \textbf{ then } x \\ \textbf{else } f\ (rep\ (n-1)\ f\ x)\end{aligned}$$

inferire il tipo del termine sopra

Esercizio

$$\lambda x.\ \left(\left(\begin{array}{l} \mathbf{rec}\ f.\ \lambda y.\ \mathbf{if}\ (x - y)\ \mathbf{then}\ 0 \\ \qquad\qquad\quad \mathbf{else\ if}\ (x + y)\ \mathbf{then}\ 1 \\ \qquad\qquad\quad \mathbf{else}\ f\ (y + 1) \end{array} \right)\ 0 \right)$$

inferire il tipo del termine sopra

Substituzioni capture-avoiding (di nuovo)

Variabili libere

$$\mathrm{fv}(n) \stackrel{\text{def}}{=} \varnothing$$

$$\mathrm{fv}(x) \stackrel{\text{def}}{=} \{x\}$$

$$\mathrm{fv}(t_0 \text{ op } t_1) \stackrel{\text{def}}{=} \mathrm{fv}(t_0) \cup \mathrm{fv}(t_1)$$

$$\mathrm{fv}(\textbf{if } t \textbf{ then } t_0 \textbf{ else } t_1) \stackrel{\text{def}}{=} \mathrm{fv}(t) \cup \mathrm{fv}(t_0) \cup \mathrm{fv}(t_1)$$

$$\mathrm{fv}((t_0, t_1)) \stackrel{\text{def}}{=} \mathrm{fv}(t_0) \cup \mathrm{fv}(t_1)$$

$$\mathrm{fv}(\textbf{fst}(t)) \stackrel{\text{def}}{=} \mathrm{fv}(t)$$

$$\mathrm{fv}(\textbf{snd}(t)) \stackrel{\text{def}}{=} \mathrm{fv}(t)$$

$$\mathrm{fv}(\lambda x.\ t) \stackrel{\text{def}}{=} \mathrm{fv}(t) \setminus \{x\}$$

$$\mathrm{fv}((t_0\ t_1)) \stackrel{\text{def}}{=} \mathrm{fv}(t_0) \cup \mathrm{fv}(t_1)$$

$$\mathrm{fv}(\textbf{rec } x.\ t) \stackrel{\text{def}}{=} \mathrm{fv}(t) \setminus \{x\}$$

Substituzioni

$$n[^t/_x] = n$$

$$y[^t/_x] \stackrel{\text{def}}{=} \begin{cases} t & \text{if } y = x \\ y & \text{if } y \neq x \end{cases}$$

$$(t_0 \text{ op } t_1)[^t/_x] \stackrel{\text{def}}{=} t_0[^t/_x] \text{ op } t_1[^t/_x] \quad \text{with op} \in \{+, -, \times\}$$

$$(\mathbf{if}\ t'\ \mathbf{then}\ t_0\ \mathbf{else}\ t_1)[^t/_x] \stackrel{\text{def}}{=} \mathbf{if}\ t'[^t/_x]\ \mathbf{then}\ t_0[^t/_x]\ \mathbf{else}\ t_1[^t/_x]$$

$$(t_0, t_1)[^t/_x] \stackrel{\text{def}}{=} (t_0[^t/_x], t_1[^t/_x])$$

$$\mathbf{fst}(t')[^t/_x] \stackrel{\text{def}}{=} \mathbf{fst}(t'[^t/_x])$$

$$\mathbf{snd}(t')[^t/_x] \stackrel{\text{def}}{=} \mathbf{snd}(t'[^t/_x])$$

$$(t_0\ t_1)[^t/_x] \stackrel{\text{def}}{=} (t_0[^t/_x]\ t_1[^t/_x])$$

$$(\lambda y.\ t')[^t/_x] \stackrel{\text{def}}{=} \lambda z.\ (\ t'[^z/_y][^t/_x]\ ) \quad \text{for } z \notin \text{fv}(\lambda y.\ t') \cup \text{fv}(t) \cup \{x\}$$

$$(\mathbf{rec}\ y.\ t')[^t/_x] \stackrel{\text{def}}{=} \mathbf{rec}\ z.\ (t'[^z/_y][^t/_x]) \quad \text{for } z \notin \text{fv}(\mathbf{rec}\ y.\ t') \cup \text{fv}(t) \cup \{x\}$$

I tipi sono rispettati

TH. $x_0 : \tau_0$

$t_0 : \tau_0 \qquad t : \tau \qquad \Rightarrow \qquad t[^{t_0}/_{x_0}] : \tau$

prova omessa
(per induzione strutturale)