

Semânticas de Linguagens de Programação - CC3033

Exercícios 9

Linguagem funcional FUN

1. Escreve um programa em **FUN** para calcular o máximo divisor comum entre dois inteiros x e y .
2. Considera o programa **FUN**

$$f_1 \ x = \text{let } y = x \times x \text{ in } y \times y$$
$$f_2 \ x = \text{let } x = x \times x \text{ in } x \times x$$

com $\Gamma(f_1) = \Gamma(f_2) = \text{int} \rightarrow \text{int}$.

- (a) Mostra que admite tipo no ambiente Γ .
- (b) Verifica se o termo $\text{let } x = 2 \text{ in } f_1 \ (f_2 \ x)$ admite tipo em Γ
- (c) Avalia o termo

$$\text{let } x = 2 \text{ in } f_1 \ (f_2 \ x)$$

na semântica de passagem de argumentos por valor $\Downarrow_P^v$ (dinâmica).

3. Considera o programa **FUN**

$$g_1 \ x = \text{let } x_2 = 1 \text{ in let fun } f \ x_1 : \text{int} = (x_1 + x_2) : \text{int} \text{ in } (f \ x \times x)$$

- (a) Mostra que admite tipo no ambiente $\Gamma(g_1) = \text{int} \rightarrow \text{int}$.
- (b) Avalia o termo $g_1 \ 3$ na semântica de passagem de argumentos por valor $\Downarrow_P^v$.
- (c) Usando uma das linguagens do Programming zoo implementa e testa a função g_1 .
- (d)

4. Seja o termo

$$(\text{let fun } \text{sqr } x : \text{int} = (x \times x) : \text{int} \text{ in } (\text{sqr } 3) > (\text{sqr } 2))$$

- (a) Mostra que admite tipo.
- (b) Avalia-o na semântica de passagem de argumentos por valor $\Downarrow_P^v$.

5. Seja o termo

$$\text{let } x = 25 \text{ in let fun } f \ x : \text{int} = (x + 3) : \text{int} \text{ in } (f \ 7) + x$$

- (a) Mostra que admite tipo no ambiente vazio.
- (b) Avalia-o na semântica de passagem de argumentos por valor $\Downarrow_P^v$.

6. Modifica a semântica de passagem de argumentos por valor $\Downarrow_P^v$ para que o âmbito dos identificadores seja estático. Sugestão: Na regra *letfun* inclui o ambiente na definição da função f .

Indica qual o valor obtido para o termo seguinte usando o âmbito dinâmico e o usando o âmbito estático (para o ambiente vazio):

$$\text{let } x = 3 \text{ in let fun } f \ y : \text{int} = (y + x) : \text{int} \text{ in let } x = 8 \text{ in } (f \ 2)$$

7. Considera o programa programa **FUN**

$$f_1 \ x \ y = \text{let } z = x \times y \text{ in } z \times z$$

- (a) Mostra que o programa admite tipo considerando um contexto Γ tal que $\Gamma(f_1) = \text{int} \rightarrow \text{int} \rightarrow \text{int}$.
- (b) Mostra que o termo t_1 dado por `let w = f1 2 in (w 3)`, admite tipo no contexto Γ .
- (c) Avalia o termo t_1 na semântica de passagem de argumentos por valor $\Downarrow_P^v$ (âmbito dinâmico), isto é, indica se existe um v tal que $\vdash t_1 \Downarrow_P^v v$. Indica também as regras usadas.
- (d) Avalia o termo t_1 usando o âmbito estático.
8. Define uma semântica de passagem de argumentos por nome para **FUN**.
9. Implementa em Haskell a linguagem **FUN** e a semântica de passagem de argumentos por valor $\Downarrow_P^v$ (supondo que os termos e programas admitem tipo).

Regras do sistema de tipos para FUN

$$\begin{array}{c}
\frac{}{\Gamma \vdash b : \text{bool}} \text{ (b)} \qquad \frac{\Gamma \vdash t_1 : \text{bool} \quad \Gamma \vdash t_2 : \text{bool}}{\Gamma \vdash t_1 \wedge t_2 : \text{bool}} \text{ (and)} \\
\\
\frac{}{\Gamma \vdash n : \text{int}} \text{ (n)} \qquad \frac{\Gamma \vdash t : \text{bool}}{\Gamma \vdash \neg t : \text{bool}} \text{ (not)} \\
\\
\frac{\Gamma(id) = \sigma}{\Gamma \vdash id : \sigma} \text{ (id)} \qquad \frac{\Gamma \vdash t_0 : \text{bool} \quad \Gamma \vdash t_1 : \sigma \quad \Gamma \vdash t_2 : \sigma}{\Gamma \vdash \text{if } t_0 \text{ then } t_1 \text{ else } t_2 : \sigma} \text{ (if)} \\
\\
\frac{\Gamma \vdash t_1 : \text{int} \quad \Gamma \vdash t_2 : \text{int}}{\Gamma \vdash t_1 \text{ op } t_2 : \text{int}} \text{ (op)} \qquad \frac{\Gamma \vdash t_1 : \sigma \rightarrow \tau \quad \Gamma \vdash t_2 : \sigma}{\Gamma \vdash (t_1 t_2) : \tau} \text{ (Ap)} \\
\\
\frac{\Gamma \vdash t_1 : \text{int} \quad \Gamma \vdash t_2 : \text{int}}{\Gamma \vdash t_1 \text{ bop } t_2 : \text{bool}} \text{ (bop)} \qquad \frac{\Gamma \vdash t_1 : \sigma \quad \Gamma, x : \sigma \vdash t_2 : \tau}{\Gamma \vdash \text{let } x = t_1 \text{ in } t_2 : \tau} \text{ (let)} \\
\\
\text{ (letfun)} \frac{\Gamma, x_1 : \sigma_1, \dots, x_n : \sigma_n, f : \rho \vdash t_1 : \sigma \quad \Gamma, f : \rho \vdash t_2 : \tau}{\Gamma \vdash \text{let fun } f x_1 : \sigma_1 x_2 : \sigma_2 \dots x_n : \sigma_n = t_1 : \sigma \text{ in } t_2 : \tau}
\end{array}$$

onde $\rho = \sigma_1 \rightarrow \sigma_2 \rightarrow \dots \rightarrow \sigma_n \rightarrow \sigma$.

Semântica com passagem de argumentos por valor para FUN (âmbito dinâmico),
dado um programa P , $f_i x_1 \dots x_{a_i} = d_i$, $1 \leq i \leq k$.

$$\begin{array}{l}
\text{(n)} \frac{}{\Gamma \vdash n \Downarrow_P^v n} \\
\text{(b)} \frac{}{\Gamma \vdash b \Downarrow_P^v b} \\
\text{(f)} \frac{ar(f) > 0}{\Gamma \vdash f \Downarrow_P^v f} \\
\text{(id)} \frac{\Gamma(x) = v}{\Gamma \vdash x \Downarrow_P^v v} \\
\text{(op)} \frac{\Gamma \vdash t_1 \Downarrow_P^v n_1 \quad \Gamma \vdash t_2 \Downarrow_P^v n_2}{\Gamma \vdash t_1 \text{ op } t_2 \Downarrow_P^v n_1 \text{ op } n_2} \\
\text{(bop)} \frac{\Gamma \vdash t_1 \Downarrow_P^v n_1 \quad \Gamma \vdash t_2 \Downarrow_P^v n_2}{\Gamma \vdash t_1 \text{ bop } t_2 \Downarrow_P^v n_1 \text{ bop } n_2} \\
\text{(and)} \frac{\Gamma \vdash t_1 \Downarrow_P^v b_1 \quad \Gamma \vdash t_2 \Downarrow_P^v b_2}{\Gamma \vdash t_1 \wedge t_2 \Downarrow_P^v b_1 \wedge b_2} \\
\frac{\Gamma \vdash t \Downarrow_P^v b}{\Gamma \vdash \neg t \Downarrow_P^v \neg b} \text{ (not)} \\
\frac{\Gamma \vdash t_0 \Downarrow_P^v \text{true} \quad \Gamma \vdash t_1 \Downarrow_P^v v_1}{\Gamma \vdash \text{if } t_0 \text{ then } t_1 \text{ else } t_2 \Downarrow_P^v v_1} \text{ (ifT)} \\
\frac{\Gamma \vdash t_0 \Downarrow_P^v \text{false} \quad \Gamma \vdash t_2 \Downarrow_P^v v_2}{\Gamma \vdash \text{if } t_0 \text{ then } t_1 \text{ else } t_2 \Downarrow_P^v v_2} \text{ (ifF)} \\
\frac{\Gamma \vdash t_1 \Downarrow_P^v v_1 \quad \Gamma, x = v_1 \vdash t_2 \Downarrow_P^v v}{\Gamma \vdash \text{let } x = t_1 \text{ in } t_2 \Downarrow_P^v v} \text{ (let)} \\
\frac{\Gamma, f = \lambda x_1 \dots x_n. t_1 \vdash t_2 \Downarrow_P^v v}{\Gamma \vdash \text{let fun } f x_1 \dots x_n = t_1 \text{ in } t_2 \Downarrow_P^v v} \text{ (letfun)} \\
\frac{\Gamma \vdash d_i \Downarrow_P^v v}{\Gamma \vdash f_i \Downarrow_P^v v \quad \text{se } def(f_i) = d_i} \text{ (fn0)} \\
\text{(Ap)} \frac{\Gamma \vdash s \Downarrow_P^v f t_1 \dots t_i \quad \Gamma \vdash t \Downarrow_P^v v \quad i < ar(f) - 1}{\Gamma \vdash (s t) \Downarrow_P^v (f t_1 \dots t_i v)} \\
\text{(fn)} \frac{\Gamma \vdash s \Downarrow_P^v f_i v_1 \dots v_{a_i-1} \quad \Gamma \vdash t \Downarrow_P^v v_{a_i} \quad \Gamma \vdash d_i[\vec{x} \mapsto \vec{v}] \Downarrow_P^v v}{\Gamma \vdash (s t) \Downarrow_P^v v \quad \text{se } def(f_i) = \lambda x_1 \dots x_{a_i}. d_i} \\
\\
def(f_i) = \begin{cases} \Gamma(f_i) & \text{se } f_i \in dom(\Gamma) \\ \lambda x_1 \dots x_{a_i}. d_i & \text{se } f_i x_1 \dots x_{a_i} = d_i \in P \\ \lambda x_1 x_2. x_1 f_i x_2 & \text{se } f_i \text{ é um operador.} \end{cases}
\end{array}$$