

1. Proponé una expresión e que tenga como única variable libre a n tal que $\llbracket e \rrbracket[\eta \mid n : \iota_{int} k] \rrbracket$ sea un estado con $|k|$ referencias todas accesibles a partir del valor devuelto. Por ejemplo, con $k = 1$ la semántica podría ser $\iota_{norm} \langle [r_0 : \iota_{tuple} \langle \rangle], \iota_{ref} r_0 \rangle$.

letrec $f \equiv \lambda x . (\text{if } x = 0 \text{ then } () \text{ else } (\text{ref } 0, f(x - 1)))$ in $f(\text{if } n \geq 0 \text{ then } n \text{ else } -n)$

2. Recordemos que en el lenguaje imperativo no había asignaciones múltiples del tipo $x, y := e, e'$. En este ejercicio nos preguntamos por asignaciones múltiples en Iswim.

- Sean $\eta \in Env$ y $\sigma \in \Sigma$ tales que $\eta u = \iota_{ref} r_0$, $\eta v = \iota_{ref} r_1$ y $r_0, r_1 \in dom(\sigma)$. ¿La semántica de $\langle u, v \rangle := \langle 3, 4 \rangle$ corresponde con la semántica de una asignación múltiple? Justificá tu respuesta.
- Suponé que la expresión $\langle e_l, e'_l \rangle := \langle e_0, e_1 \rangle$ es azúcar sintáctico por $(e_l := e_0); (e'_l := e_1)$. ¿Dirías que esta traducción es correcta? Justificá tu respuesta.
- Proponé una construcción sintáctica que permita una asignación de dos referencias simultáneamente, junto con su regla de evaluación y su ecuación semántica.

a)
 $\langle u, v \rangle := \langle 3, 4 \rangle$

La semántica denotacional da $tyerr$

$$\begin{aligned} & \llbracket \langle u, v \rangle := \langle 3, 4 \rangle \rrbracket \eta \sigma \\ &= \left(\lambda \langle \sigma', r \rangle . \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r : z], \iota_{tuple} \langle \rangle \right) \right) \langle 3, 4 \rangle \eta \bar{p}' \right)_{ref^*} \langle \bar{u}, \bar{v} \rangle \bar{p} \\ &= \left(\lambda \langle \sigma', r \rangle . \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r : z], \iota_{tuple} \langle \rangle \right) \right) \langle 3, 4 \rangle \eta \bar{p}' \right)_{ref^*} \left(\iota_{norm} \left\langle \sigma, \iota_{tuple} \langle \iota_{ref} r_0, \iota_{ref} r_1 \rangle \right\rangle \right) \\ &= tyerr \end{aligned}$$

No es lo se esperaría de la asignación múltiple

b)
 Sea:

$$e = \langle x, y \rangle := \langle \text{val } y, \text{val } x \rangle = (\lambda v. y := \text{val } x)(x := \text{val } y)$$

Si suponemos $\eta x = \iota_{ref} r_0, \eta y = \iota_{ref} r_1, r_0 \neq r_1, r_0, r_1 \in Dom(\sigma)$

$$\begin{aligned} & \llbracket e \rrbracket \eta \sigma \\ &= \llbracket \lambda v. y := \text{val } x(x := \text{val } y) \rrbracket \bar{p} \\ &= \lambda \langle \sigma', f \rangle . f_* \{ \text{val } y \bar{p}' \}_{fun^*} \{ \text{val } x \bar{p} \} \\ &= \lambda \langle \sigma', f \rangle . f_* \{ \text{val } y \bar{p}' \}_{fun^*} \iota_{fun} \left(\sigma, \iota_{fun} \left(\lambda \langle \sigma', z \rangle . y \llbracket := \text{val } z \rrbracket \mid v : z \right] \sigma' \right) \\ &= (\lambda \langle \sigma', z \rangle . y \llbracket := \text{val } z \rrbracket \mid v : z] \sigma')_* \{ \text{val } y \bar{p} \} \\ &= (\\ &\quad x \llbracket := \text{val } y \bar{p} \\ &\quad = \left(\lambda \langle \sigma', r \rangle . \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r : z], \iota_{tuple} \langle \rangle \right) \right) \{ \text{val } y \bar{p} \} \right)_{ref^*} \langle \llbracket x \rrbracket \eta \sigma \rangle \\ &\quad = \left(\lambda \langle \sigma', r \rangle . \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r : z], \iota_{tuple} \langle \rangle \right) \right) \{ \text{val } y \bar{p} \} \right)_{ref^*} \iota_{norm} \left\langle \sigma, \iota_{ref} r_0 \right\rangle \end{aligned}$$

$$\begin{aligned}
&= \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r_0 : z], \iota_{tuple} \langle \rangle \right) \right) (\llbracket y \rrbracket p) \\
&= \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r_0 : z], \iota_{tuple} \langle \rangle \right) \right) (\iota_{norm} \langle \sigma, \sigma r_1 \rangle) \\
&= \iota_{norm} \left([\sigma \mid r_0 : \sigma r_1], \iota_{tuple} \langle \rangle \right) \\
) \\
&(\lambda \langle \sigma', z \rangle y ::= \text{val } x \mid v : z] \sigma') \left(\iota_{norm} \left([\sigma \mid r_0 : \sigma r_1], \iota_{tuple} \langle \rangle \right) \right) \\
&= y ::= \text{val } x \mid v : \iota_{tuple} \langle \rangle [\sigma \mid r_0 : \sigma r_1] \\
&= y ::= \text{val } x \llbracket \sigma \mid r_0 : \sigma r_1 \rrbracket \\
&= \left(\lambda \langle \sigma', r \rangle . \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r : z], \iota_{tuple} \langle \rangle \right) \right) (\llbracket x \rrbracket p') \right)_{ref^*} (\llbracket y \rrbracket \eta[\sigma \mid r_0 : \sigma r_1]) \\
&= \left(\lambda \langle \sigma', r \rangle . \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r : z], \iota_{tuple} \langle \rangle \right) \right) (\llbracket x \rrbracket p') \right)_{ref^*} \iota_{norm} \langle [\sigma \mid r_0 : \sigma r_1], \iota_{ref} r_1 \rangle \\
&= \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r_1 : z], \iota_{tuple} \langle \rangle \right) \right) (\llbracket x \rrbracket \sigma \mid r_0 : \sigma r_1) \\
&= \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r_1 : z], \iota_{tuple} \langle \rangle \right) \right) \iota_{norm} \langle [\sigma \mid r_0 : \sigma r_1], [\sigma \mid r_0 : \sigma r_1] r_0 \rangle \\
&= \left(\lambda \langle \hat{\sigma}, z \rangle . \iota_{norm} \left([\hat{\sigma} \mid r_1 : z], \iota_{tuple} \langle \rangle \right) \right) \iota_{norm} \langle [\sigma \mid r_0 : \sigma r_1], \sigma r_1 \rangle \\
&= \iota_{norm} \langle [\sigma \mid r_0 : \sigma r_1, r_1 : \sigma r_1], \iota_{tuple} \langle \rangle \rangle
\end{aligned}$$

Lo cual no es lo que esperaríamos de una asignación múltiple

c)
 $\langle exp \rangle = \dots \mid \langle exp \rangle, \langle exp \rangle \coloneqq \langle exp \rangle, \langle exp \rangle$

$$e \llbracket, e_1 := e'_0, e'_1 \rrbracket p = \\
\left(\lambda \langle \sigma_0, r_0 \rangle . \left(\lambda \langle \sigma_1, r_1 \rangle . \left(\lambda \langle \sigma_2, z_0 \rangle . \left(\lambda \langle \sigma_3, z_1 \rangle . \iota_{norm} \left([\sigma_2 \mid r_0 : z_0, r_1 : z_1], \iota_{tuple} \langle \rangle \right) \right) \llbracket p_2 \rrbracket \right) \ast \right) \iota_{\llbracket p_1 \rrbracket \sigma_1} \right)_{ref^*} \iota_{\llbracket p_0 \rrbracket \sigma_0} \iota_{\llbracket p \rrbracket}$$

Si $\sigma, e_0 \Rightarrow r_0, \sigma_1 \wedge \sigma_1, e_1 \Rightarrow r_1, \sigma_2 \wedge \sigma_2, e'_0 \Rightarrow z_0, \sigma_3 \wedge \sigma_3, e'_1 \Rightarrow z_1, \sigma_4$
Entonces $\sigma, (e_0, e_1 := e'_0, e'_1) \Rightarrow \langle \rangle, [\sigma_4 \mid r_0 : z_0, r_1 : z_1]$

Ejemplo:

$$\begin{aligned}
&[], (\lambda xy(\lambda v. \langle \text{val } x, \text{val } y \rangle)(x, y ::= \text{val } y, \text{val } x)) \text{ref } 0 \text{ (ref 1)} \\
&[], (\lambda xy(\lambda v. \langle \text{val } x, \text{val } y \rangle)(x, y ::= \text{val } y, \text{val } x)) \text{ref } 0 \\
&[], \lambda xy(\lambda v. \langle \text{val } x, \text{val } y \rangle)(x, y ::= \text{val } y, \text{val } x) \not\Rightarrow \lambda xy(\lambda v. \langle \text{val } x, \text{val } y \rangle)(x, y ::= \text{val } y, \text{val } x) \\
&[], \text{ref } 0 \Rightarrow r_0, [r_0 : 0] \\
&[r_0 : 0], \lambda y(\lambda v. \langle \text{val } r_0, \text{val } y \rangle)(x, y ::= \text{val } y, \text{val } r_0) \not\Rightarrow \lambda y(\lambda v. \langle \text{val } r_0, \text{val } y \rangle)(x, y ::= \text{val } y, \text{val } r_0) \\
&\Rightarrow \lambda y(\lambda v. \langle \text{val } r_0, \text{val } y \rangle)(r_0, y ::= \text{val } y, \text{val } r_0) \\
&[r_0 : 0], \text{ref } 1 \Rightarrow r_1, [r_0 : 0, r_1 : 1] \\
&[r_0 : 0, r_1 : 1], (\lambda v. \langle \text{val } r_0, \text{val } r_1 \rangle)(r_0, r_1 ::= \text{val } r_1, \text{val } r_0) \\
&[r_0 : 0, r_1 : 1], \lambda v. \langle \text{val } r_0, \text{val } r_1 \rangle \Rightarrow \lambda v. \langle \text{val } r_0, \text{val } r_1 \rangle, [r_0 : 0, r_1 : 1]
\end{aligned}$$

$[r_0 : 0, r_1 : 1], r_0, r_1 := \text{val } r_1, \text{val } r_0$
 $[r_0 : 0, r_1 : 1], r_0 \Rightarrow r_0, [r_0 : 0, r_1 : 1]$
 $[r_0 : 0, r_1 : 1], r_1 \Rightarrow r_1, [r_0 : 0, r_1 : 1]$
 $[r_0 : 0, r_1 : 1], \text{val } r_1 \Rightarrow 1, [r_0 : 0, r_1 : 1]$
 $[r_0 : 0, r_1 : 1], \text{val } r_0 \Rightarrow 0, [r_0 : 0, r_1 : 1]$
 $\Rightarrow \langle \rangle, [[r_0 : 0, r_1 : 1] \mid r_0 : 1, r_1 : 0] = [r_0 : 1, r_1 : 0]$
 $[r_0 : 1, r_1 : 0], \langle \text{val } r_0, \text{val } r_1 \rangle$
 $[r_0 : 1, r_1 : 0], \text{val } r_0 \Rightarrow 1, [r_0 : 1, r_1 : 0]$
 $[r_0 : 1, r_1 : 0], \text{val } r_1 \Rightarrow 0, [r_0 : 1, r_1 : 0]$
 $\Rightarrow \langle 1, 0 \rangle, [r_0 : 1, r_1 : 0]$
 $\Rightarrow \langle 1, 0 \rangle, [r_0 : 1, r_1 : 0]$
 $\Rightarrow \langle 1, 0 \rangle, [r_0 : 1, r_1 : 0]$

3. Ejercicio 13.3 del Reynolds: implementar referencias que permitan deshacer asignaciones.

Extensión de las expresiones y formas canónicas de lenguaje aplicativo eager

$$\langle \text{exp} \rangle = \dots | \text{ref } \langle \text{exp} \rangle | \text{val } \langle \text{exp} \rangle | \langle \text{exp} \rangle := \langle \text{exp} \rangle | \langle \text{exp} \rangle =_{\text{ref}} \langle \text{exp} \rangle | \text{undo } \langle \text{exp} \rangle$$

$$\langle \text{cnf} \rangle = \dots | \langle Rf \rangle$$

Semántica operacional:

$$\Sigma = \bigcup_{D \subset_{fin} \langle Rf \rangle} D \rightarrow \langle \text{cnf} \rangle^*$$

$$\Rightarrow \subseteq (\Sigma \times \langle \text{exp} \rangle) \times (\langle \text{cnf} \rangle \times \Sigma)$$

r representa un elemento de $\langle Rf \rangle$

$$\langle \sigma, z \rangle \Rightarrow \langle z, \sigma \rangle$$

Si $\langle \sigma_0, e_0 \rangle \Rightarrow \langle \sigma_1, \lambda v. e'_0 \rangle$, $\langle \sigma_1, e_1 \rangle \Rightarrow \langle z, \sigma_2 \rangle$ y $\langle \sigma_2, e'_0 / [v : z] \rangle \Rightarrow \langle z', \sigma_3 \rangle$
entonces $\langle \sigma_0, e_0 e_1 \rangle \Rightarrow \langle z', \sigma_3 \rangle$



Si $\langle \sigma, e_0 \rangle \Rightarrow \langle r, \sigma' \rangle$ y $\langle \sigma', e_1 \rangle \Rightarrow \langle z', \sigma'' \rangle$

entonces $\langle \sigma, e_0 := e_1 \rangle \Rightarrow \langle \langle \rangle, [\sigma'' | r : z' \cdot (\sigma'' r)] \rangle$

Si $\langle \sigma, e \rangle \Rightarrow \langle z, \sigma' \rangle$ entonces $\langle \sigma, \text{ref } e \rangle \Rightarrow \langle r, [\sigma' | r : \langle z \rangle] \rangle$

Donde $r \notin \text{Dom}(\sigma')$

Si $\langle \sigma, e \rangle \Rightarrow \langle r, \sigma' \rangle$ y $\sigma' r \neq \langle \rangle$ entonces $\langle \sigma, \text{val } e \rangle \Rightarrow \langle \sigma' r, (\sigma' r)_0 \rangle$

Si $\langle \sigma, e_0 \rangle \Rightarrow \langle r_0, \sigma' \rangle$ y $\langle \sigma', e_1 \rangle \Rightarrow \langle r_1, \sigma'' \rangle$ entonces $\langle \sigma, e_0 =_{\text{ref}} e_1 \rangle \Rightarrow \langle [r = r'], \sigma'' \rangle$

Si $\langle \sigma, e \rangle \Rightarrow \langle r, \sigma' \rangle$, $r \in \text{Dom}(\sigma')$ y $\sigma' r \neq \langle \rangle$ entonces $\langle \sigma, \text{undo } e \rangle \Rightarrow \langle \langle \rangle, [\sigma | r : (\sigma r)_{1\dots}] \rangle$

Semántica denotacional:

$$V_{int} = \mathbb{Z}$$

$$V_{bool} = \{T, F\}$$

$$V_{fun} = [V \rightarrow D]$$

$$V_{tuple} = V^*$$

$$V = V_{int} + V_{bool} + V_{fun} + V_{tuple} + V_{ref}$$

$$\Sigma = \{\sigma \in V_{ref} \times V^*: \sigma \text{ es función}\}$$

$$D = (\Sigma \times V + \{\text{error}\} + \{\text{typeerror}\})_{\perp}$$

$$Env = \langle var \rangle \rightarrow V$$

$$\begin{aligned}err &= \iota_{\perp} \iota_{\text{error}} \\tyerr &= \iota_{\perp} \iota_{\text{typeerror}} \\\iota_{\text{norm}} : V &\rightarrow D\end{aligned}$$

Si $f : V \rightarrow D$:

$$\begin{aligned}f_*(\iota_{\text{norm}} v) &= f v \\f_* x &= x\end{aligned}$$

Para $\theta \in \{\text{inf}, \text{bool}, \text{fun}, \text{tuple}, \text{ref}\}$:

$$\begin{aligned}\iota_\theta : V_\theta &\rightarrow V \\\underline{\iota_\theta} &= \iota_{\text{norm}} \circ \iota_\theta\end{aligned}$$

Si $f : \Sigma \times V_\theta \rightarrow D$

$$\begin{aligned}f_\theta(\langle \sigma, \iota_\theta x \rangle) &= f \langle \sigma, x \rangle \\f_\theta(\langle \sigma, \iota_{\theta'} x \rangle) &= tyerr \\f_\theta x &= x\end{aligned}$$

$$f_{\theta*} = (f_\theta)_*$$

$$\boxed{\boxed{\langle \text{exp} \rangle \rightarrow \text{Env} \rightarrow \Sigma \rightarrow D}}$$

$$v[\![\eta]\!] = \iota_{\text{norm}}(\sigma, \eta v)$$

$$e[\![\eta]\!] = \lambda(\langle \sigma', f \rangle . f_* e[\![\eta]\!])_{\text{fun}*} e[\![\eta]\!]$$

$$\lambda[\![\eta]\!] = \iota_{\text{norm}}(\sigma, \iota_{\text{fun}}(\lambda(\langle \sigma', z \rangle e[\![\eta]\!] | v : z] \sigma'))$$



$$\text{val } e[\![\eta]\!] = \left(\lambda(\sigma', r) . \begin{cases} r \in \text{Dom}(\sigma') \wedge \sigma' r \neq \langle \rangle & \rightarrow \iota_{\text{norm}}(\sigma', (\sigma' r)_0) \\ \text{si no} & \rightarrow \text{err} \end{cases} \right)_{\text{ref}*} ([\![e]\!] \eta \sigma)$$

$$\text{ref } e[\![\eta]\!] = \lambda(\langle \sigma', z \rangle . \iota_{\text{norm}}([\sigma' | r : \langle z \rangle], \iota_{\text{ref}} z)) [\![e]\!] \eta \sigma$$

Donde $r \in \text{new}(\sigma')$

$$e[\![\eta]\!] := e[\![\eta]\!] = \left(\lambda(\sigma', r) . \left(\lambda(\hat{\sigma}, z) . \iota_{\text{norm}} \left([\hat{\sigma} | r : (z \triangleright \hat{\sigma} r)], \iota_{\text{tuple}} \langle \rangle \right) \right)_{\text{ref}*} e[\![\eta]\!] \right)_{\text{ref}*} ([\![e_0]\!] \eta \sigma)$$

$$[\![e_0 =_{\text{ref}} e]\!] = \left(\lambda(\sigma', r_0) . \left(\lambda(\hat{\sigma}, r_1) . \iota_{\text{norm}} \langle \hat{\sigma}, \iota_{\text{bool}}(r_0 = r_1) \rangle \right)_{\text{ref}*} e[\![\eta]\!] \right)_{\text{ref}*} ([\![e_0]\!] \eta \sigma)$$

$$\text{undo } e[\![\eta]\!] = \left(\lambda(\sigma', r) . \begin{cases} r \in \text{Dom}(\sigma') \wedge \sigma' r \neq \langle \rangle & \rightarrow \iota_{\text{norm}} \langle [\sigma' | r : (\sigma' r)_1 \dots], \iota_{\text{tuple}} \langle \rangle \rangle \\ \text{si no} & \rightarrow \text{err} \end{cases} \right)_{\text{ref}*} ([\![e]\!] \eta \sigma)$$

4. Ahora nos centramos en la composición secuencial (nombre raro para el punto-y-coma).

- a) Proponé una traducción que no descarte el valor de e en $e; e'$, devolviendo la tupla $\langle z, z' \rangle$, asumiendo que z y z' son los valores de e y e' respectivamente.
- b) Indicá cuál es el valor de $\text{ref } 1 := \langle 2, 3 \rangle; \text{ref } 5 := 4$ usando la traducción del ítem anterior. Contrastá ese valor con el valor si usamos la traducción del teórico.
- c) Considerá de nuevo $e; e'$. ¿Esperarías que si e evalúa a una referencia r , esa referencia pueda ser usada en e' ? ¿Te parece que la nueva traducción logra eso?

a)

$$e_0; e_1 = \langle e_0, e_1 \rangle$$

b)

$$\text{Ahora tenemos } \sigma, \text{ref } 1 := \langle 2, 3 \rangle; \text{ref } 5 := 4 \Rightarrow \langle \langle \rangle, \langle \rangle \rangle, [\sigma \mid r_0: \langle 2, 3 \rangle, r_1: 4]$$

$$\text{Antes teníamos } \sigma, \text{ref } 1 := \langle 2, 3 \rangle; \text{ref } 5 := 4 \Rightarrow \langle \rangle, [\sigma \mid r_0: \langle 2, 3 \rangle, r_1: 4]$$

c)

e' no debería poder usar el resultado de e y mi traducción efectivamente lo impide