Lenguaje Aplicativo, eager y normal

2017

Estructura de la materia a grandes rasgos:

Primera Parte: Lenguaje imperativo

Segunda Parte: Lenguaje aplicativo puro, y lenguaje aplicativo con referencias y asignación

Ejes de contenidos de la segunda parte

Lenguajes Aplicativos

$$
\begin{array}{rcl}
\langle exp \rangle & ::= & \langle var \rangle \mid \langle exp \rangle \langle exp \rangle \mid \lambda \langle var \rangle . \langle exp \rangle \\
& \mid & \langle natconst \rangle \mid \langle boolconst \rangle \\
& \mid & - \langle exp \rangle \mid \langle exp \rangle + \langle exp \rangle \mid \langle exp \rangle * \langle exp \rangle \mid \ldots \\
& \mid & \langle exp \rangle \geq \langle exp \rangle \mid \langle exp \rangle \leq \langle exp \rangle \mid \langle exp \rangle < \langle exp \rangle \mid \ldots \\
& \mid & \langle exp \rangle \wedge \langle exp \rangle \mid \langle exp \rangle \vee \langle exp \rangle \mid \neg \langle exp \rangle \\
& \mid & \mathbf{if}\ \langle exp \rangle\ \mathbf{then}\ \langle exp \rangle\ \mathbf{else}\ \langle exp \rangle \\
& \mid & \textsf{error} \mid \textsf{typeerror} \\
\\
\langle natconst \rangle & ::= & 0 \mid 1 \mid 2 \mid \ldots . \\
\langle boolconst \rangle & ::= & \textsf{true} \mid \textsf{false}
\end{array}
$$

Lenguajes Aplicativos

$$
\begin{array}{rcl}
\langle exp \rangle & ::= & \langle var \rangle \mid \langle exp \rangle \, \langle exp \rangle \mid \lambda \, \langle var \rangle \, . \, \langle exp \rangle \\
& \mid & \langle natconst \rangle \mid \langle boolconst \rangle \\
& \mid & - \langle exp \rangle \mid \langle exp \rangle + \langle exp \rangle \mid \langle exp \rangle * \langle exp \rangle \mid \ldots \\
& \mid & \langle exp \rangle \geq \langle exp \rangle \mid \langle exp \rangle \leq \langle exp \rangle \mid \langle exp \rangle < \langle exp \rangle \mid \ldots \\
& \mid & \langle exp \rangle \wedge \langle exp \rangle \mid \langle exp \rangle \vee \langle exp \rangle \mid \neg \langle exp \rangle \\
& \mid & \mathbf{if} \; \langle exp \rangle \; \mathbf{then} \; \langle exp \rangle \; \mathbf{else} \; \langle exp \rangle \\
& \mid & \textsf{\textbf{error}} \mid \textsf{\textbf{typeerror}}
\end{array}
$$

$$
\begin{array}{rcl}
\langle natconst \rangle & ::= & 0 \mid 1 \mid 2 \mid \ldots . \\
\langle boolconst \rangle & ::= & \textsf{\textbf{true}} \mid \textsf{\textbf{false}}
\end{array}
$$

Lenguajes Aplicativos

$$
\begin{array}{rcl}
\langle exp \rangle & ::= & \langle var \rangle \mid \langle exp \rangle \, \langle exp \rangle \mid \lambda \langle var \rangle \, . \, \langle exp \rangle \\
& \mid & \langle natconst \rangle \mid \langle boolconst \rangle \\
& \mid & - \langle exp \rangle \mid \langle exp \rangle + \langle exp \rangle \mid \langle exp \rangle * \langle exp \rangle \mid \ldots \\
& \mid & \langle exp \rangle \geq \langle exp \rangle \mid \langle exp \rangle \leq \langle exp \rangle \mid \langle exp \rangle < \langle exp \rangle \mid \ldots \\
& \mid & \langle exp \rangle \wedge \langle exp \rangle \mid \langle exp \rangle \vee \langle exp \rangle \mid \neg \langle exp \rangle \\
& \mid & \textbf{if} \ \langle exp \rangle \ \textbf{then} \ \langle exp \rangle \ \textbf{else} \ \langle exp \rangle \\
& \mid & \textbf{error} \mid \textbf{typeerror} \\
\\
\langle natconst \rangle & ::= & 0 \mid 1 \mid 2 \mid \ldots. \\
\langle boolconst \rangle & ::= & \textbf{true} \mid \textbf{false}
\end{array}
$$

Lenguajes Aplicativos

$$
\begin{array}{rcl}
\langle exp \rangle & ::= & \langle var \rangle \mid \langle exp \rangle \, \langle exp \rangle \mid \lambda \, \langle var \rangle \, . \, \langle exp \rangle \\
& \mid & \langle natconst \rangle \mid \langle boolconst \rangle \\
& \mid & - \langle exp \rangle \mid \langle exp \rangle + \langle exp \rangle \mid \langle exp \rangle * \langle exp \rangle \mid \ldots \\
& \mid & \langle exp \rangle \geq \langle exp \rangle \mid \langle exp \rangle \leq \langle exp \rangle \mid \langle exp \rangle < \langle exp \rangle \mid \ldots \\
& \mid & \langle exp \rangle \wedge \langle exp \rangle \mid \langle exp \rangle \vee \langle exp \rangle \mid \neg \langle exp \rangle \\
& \mid & \mathbf{if} \ \langle exp \rangle \ \mathbf{then} \ \langle exp \rangle \ \mathbf{else} \ \langle exp \rangle \\
& \mid & \textbf{error} \mid \textbf{typeerror} \\
\\
\langle natconst \rangle & ::= & 0 \mid 1 \mid 2 \mid \ldots . \\
\langle boolconst \rangle & ::= & \textbf{true} \mid \textbf{false}
\end{array}
$$

Evaluación Eager

Formas canónicas

$$
\begin{array}{lcl}
\langle cnf\rangle & ::= & \langle intcnf\rangle \mid \langle boolcnf\rangle \mid \langle funcnf\rangle \\
\\
\langle intcnf\rangle & ::= & ... \mid -2 \mid -1 \mid 0 \mid 1 \mid 2 \mid ... \\
\langle boolcnf\rangle & ::= & \langle boolconst\rangle \\
\langle funcnf\rangle & ::= & \lambda \, \langle var\rangle \, . \, \langle exp\rangle
\end{array}
$$

No hay formas canónicas para los errores.

Evaluación Eager

Formas canónicas

$$
\begin{array}{lcl}
\langle cnf \rangle & ::= & \langle intcnf \rangle \mid \langle boolcnf \rangle \mid \langle funcnf \rangle \\
\\
\langle intcnf \rangle & ::= & ... \mid -2 \mid -1 \mid 0 \mid 1 \mid 2 \mid ... \\
\langle boolcnf \rangle & ::= & \langle boolconst \rangle \\
\langle funcnf \rangle & ::= & \lambda \langle var \rangle . \langle exp \rangle
\end{array}
$$

No hay formas canónicas para los errores.

Reglas para $\Rightarrow_E$

Notación: z es una metavariable para $\langle cnf \rangle$, c para $\langle intcnf \rangle$ ó para $\langle boolcnf \rangle$, e i para $\langle intcnf \rangle$.

Regla para las formas canónicas

$$\frac{}{z \ \Rightarrow_E \ z}$$

Regla para la aplicación

$$\frac{e \ \Rightarrow_E \ \lambda v.e_0 \qquad e' \ \Rightarrow_E \ z' \qquad (e_0/v \mapsto z') \ \Rightarrow_E \ z}{ee' \ \Rightarrow_E \ z}$$

Reglas para $\Rightarrow_E$

Regla para las operaciones enteras y booleanas:

$$\frac{e \;\Rightarrow_E\; \lfloor i \rfloor \qquad e' \;\Rightarrow_E\; \lfloor i' \rfloor}{e + e' \;\Rightarrow_E\; \lfloor i + i' \rfloor}$$

Generalizando:

$$\frac{e \;\Rightarrow_E\; \lfloor c \rfloor \qquad e' \;\Rightarrow_E\; \lfloor c' \rfloor}{e \oplus e' \;\Rightarrow_E\; \lfloor c \oplus c' \rfloor}$$

donde $\oplus \in \{+, -, *, =, \neq, \leq, \geq, ...\}$

Reglas para $\Rightarrow_E$

Regla para las operaciones enteras y booleanas:

$$\frac{e \ \Rightarrow_E \ \lfloor i \rfloor \qquad e' \ \Rightarrow_E \ \lfloor i' \rfloor}{e + e' \ \Rightarrow_E \ \lfloor i + i' \rfloor}$$

Generalizando:

$$\frac{e \ \Rightarrow_E \ \lfloor c \rfloor \qquad e' \ \Rightarrow_E \ \lfloor c' \rfloor}{e \oplus e' \ \Rightarrow_E \ \lfloor c \oplus c' \rfloor}$$

donde $\oplus \in \{+, -, *, =, \neq, \leq, \geq, ...\}$

Reglas para $\Rightarrow_E$

Regla para las operaciones enteras y booleanas:

$$\frac{e \;\Rightarrow_E\; \lfloor c \rfloor}{\sim e \;\Rightarrow_E\; \lfloor \sim c \rfloor} \quad \text{donde } \sim \in \{-, \neg\}$$

$$\frac{e \;\Rightarrow_E\; \lfloor i \rfloor \qquad e' \;\Rightarrow_E\; \lfloor i' \rfloor}{e \oplus e' \;\Rightarrow_E\; \lfloor i \oplus i' \rfloor} \quad (i' \neq 0)$$

donde $\oplus \in \{/,\ \div\}$

Reglas para $\Rightarrow_E$

Regla para las operaciones enteras y booleanas:

$$\frac{e \;\Rightarrow_E\; \lfloor c \rfloor}{\sim e \;\Rightarrow_E\; \lfloor \sim c \rfloor} \quad \text{donde } \sim \in \{-, \neg\}$$

$$\frac{e \;\Rightarrow_E\; \lfloor i \rfloor \qquad e' \;\Rightarrow_E\; \lfloor i' \rfloor}{e \oplus e' \;\Rightarrow_E\; \lfloor i \oplus i' \rfloor} \quad (i' \neq 0)$$

donde $\oplus \in \{/,\ \div\}$

Reglas para $\Rightarrow_E$

$$\frac{e \;\Rightarrow_E\; \textbf{true} \qquad e_0 \;\Rightarrow_E\; z}{\textbf{if } e \textbf{ then } e_0 \textbf{ else } e_1 \;\Rightarrow_E\; z}$$

$$\frac{e \;\Rightarrow_E\; \textbf{false} \qquad e_1 \;\Rightarrow_E\; z}{\textbf{if } e \textbf{ then } e_0 \textbf{ else } e_1 \;\Rightarrow_E\; z}$$

Reglas para $\Rightarrow_E$

$$\frac{e \;\Rightarrow_E\; \textbf{true} \qquad e_0 \;\Rightarrow_E\; z}{\textbf{if } e \textbf{ then } e_0 \textbf{ else } e_1 \;\Rightarrow_E\; z}$$

$$\frac{e \;\Rightarrow_E\; \textbf{false} \qquad e_1 \;\Rightarrow_E\; z}{\textbf{if } e \textbf{ then } e_0 \textbf{ else } e_1 \;\Rightarrow_E\; z}$$

Evaluación Normal

Formas Canónicas: Las mismas (por ahora).

Regla para las formas canónicas

$$\frac{}{z \Rightarrow_N z}$$

Regla para la aplicación

$$\frac{e \Rightarrow_N \lambda v.e_0 \qquad (e_0/v \mapsto e') \Rightarrow_N z}{ee' \Rightarrow_N z}$$

Más reglas para $\Rightarrow_N$

Regla "lazy" para las operaciones booleanas:

$$\frac{e \ \Rightarrow_N \ \textbf{false}}{e \wedge e' \ \Rightarrow_N \ \textbf{false}}$$

Alternativa: uso de abreviaturas

$$\begin{aligned} e \wedge e' &=_{def} \ \textbf{if } e \textbf{ then } e' \textbf{ else false} \\ e \vee e' &=_{def} \ \textbf{if } e \textbf{ then true else } e' \\ &\vdots \end{aligned}$$

Más reglas para $\Rightarrow_N$

Regla "lazy" para las operaciones booleanas:

$$\frac{e \ \Rightarrow_N \ \textbf{false}}{e \wedge e' \ \Rightarrow_N \ \textbf{false}}$$

Alternativa: uso de abreviaturas

$$\begin{array}{lll} e \wedge e' & =_{def} & \textbf{if } e \textbf{ then } e' \textbf{ else false} \\ e \vee e' & =_{def} & \textbf{if } e \textbf{ then true else } e' \\ & \vdots & \end{array}$$

Semántica Denotacional Eager

Para el CLE:

$$D = V_\perp$$

$$V \ \simeq\ V_{fun} \qquad \text{con } V_{fun} = [V \to D]$$

Para el Lenguaje Aplicativo Eager:

$$D \ =\ (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_\perp$$

$$V \ \simeq\ V_{int} \ +\ V_{bool} \ +\ V_{fun}$$

Notar que V es un predominio: no hay un elemento mínimo!

¿Cómo son las cadenas en V?

Semántica Denotacional Eager

Para el CLE:

$$D = V_\perp$$

$$V \;\simeq\; V_{fun} \qquad \text{con } V_{fun} = [V \to D]$$

Para el Lenguaje Aplicativo Eager:

$$D \;=\; (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_\perp$$

$$V \;\simeq\; V_{int} \;+\; V_{bool} \;+\; V_{fun}$$

Notar que V es un predominio: no hay un elemento mínimo!

¿Cómo son las cadenas en V?

Semántica Denotacional Eager: Valores

$$V \;\simeq\; V_{int} + V_{bool} + V_{fun} \qquad D \;=\; (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_\perp$$

$$\begin{aligned} V_{int} &= \mathbb{Z} \\ V_{bool} &= \{T, F\} \\ V_{fun} &= [V \to D] \end{aligned}$$

Ahora el isomorfismo ϕ y su inversa ψ tienen tipos:

$\phi \in V \to \; V_{int} + V_{bool} + V_{fun}$

$\psi \in V_{int} + V_{bool} + V_{fun} \to V$

Semántica Denotacional Eager: Valores

$$V \;\simeq\; V_{int} + V_{bool} + V_{fun} \qquad D \;=\; (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_{\perp}$$

$$\begin{aligned} V_{int} &= \mathbb{Z} \\ V_{bool} &= \{T, F\} \\ V_{fun} &= [V \to D] \end{aligned}$$

Ahora el isomorfismo ϕ y su inversa ψ tienen tipos:

$$\phi \in V \to \; V_{int} + V_{bool} + V_{fun}$$

$$\psi \in V_{int} + V_{bool} + V_{fun} \to V$$

Funciones auxiliares

$\iota_{int} \in V_{int} \to V$ se define:

$$\iota_{int} = \psi \circ \iota_0$$

donde

$$\iota_0 \in V_{int} \to V_{int} + V_{bool} + V_{fun}$$

Lo mismo para booleanos y funciones:

$\iota_{int} \in V_{int} \to V$

$\iota_{bool} \in V_{bool} \to V$

$\iota_{fun} \in V_{fun} \to V$

Funciones auxiliares

$\iota_{int} \in V_{int} \to V$ se define:

$$\iota_{int} = \psi \circ \iota_0$$

donde

$$\iota_0 \in V_{int} \to V_{int} + V_{bool} + V_{fun}$$

Lo mismo para booleanos y funciones:

$\iota_{int} \in V_{int} \to V$

$\iota_{bool} \in V_{bool} \to V$

$\iota_{fun} \in V_{fun} \to V$

Constructores de D

$$D \;=\; (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_\perp$$

$err = (\iota_\perp \circ \iota_1)(\textbf{error}) = \iota_\perp(\iota_1\ (\textbf{error}))$

$tyerr = (\iota_\perp \circ \iota_2)(\textbf{typeerror})$

$\iota_{norm} \in V \to D \qquad\qquad \iota_{norm} = \iota_\perp \circ \iota_0$

Uso habitual: $\iota_{norm} \circ \iota_{int} \in V_{int} \to D$

Abreviatura: $\iota_{\underline{int}} = \iota_{norm} \circ \iota_{int} \in V_{int} \to D$

En general: $\iota_{\underline{\theta}} = \iota_{norm} \circ \iota_\theta \in V_\theta \to D$ con $\theta \in \{int, bool, fun\}$.

Constructores de D

$$D \;=\; (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_\perp$$

$err = (\iota_\perp \circ \iota_1)(\textbf{error}) = \iota_\perp(\iota_1\ (\textbf{error}))$

$tyerr = (\iota_\perp \circ \iota_2)(\textbf{typeerror})$

$\iota_{norm} \in V \to D \qquad\qquad \iota_{norm} = \iota_\perp \circ \iota_0$

Uso habitual: $\iota_{norm} \circ \iota_{int} \in V_{int} \to D$

Abreviatura: $\iota_{\underline{int}} = \iota_{norm} \circ \iota_{int} \in V_{int} \to D$

En general: $\iota_{\underline{\theta}} = \iota_{norm} \circ \iota_\theta \in V_\theta \to D$ con $\theta \in \{int, bool, fun\}$.

Constructores de D

$$D \ = \ (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_\perp$$

$err = (\iota_\perp \circ \iota_1)(\textbf{error}) = \iota_\perp(\iota_1 \ (\textbf{error}))$

$tyerr = (\iota_\perp \circ \iota_2)(\textbf{typeerror})$

$\iota_{norm} \in V \to D$ $\qquad\qquad \iota_{norm} = \iota_\perp \circ \iota_0$

Uso habitual: $\iota_{norm} \circ \iota_{int} \in V_{int} \to D$

Abreviatura: $\iota_{\underline{int}} = \iota_{norm} \circ \iota_{int} \in V_{int} \to D$

En general: $\iota_{\underline{\theta}} = \iota_{norm} \circ \iota_\theta \in V_\theta \to D$ con $\theta \in \{int, bool, fun\}$.

Más funciones auxiliares

Si $f \in V \to D$ entonces $f_* \in D \to D$ se define:

$$
\begin{aligned}
f_*(\iota_{norm}\ z) &= f\ z \\
f_*(err) &= err \\
f_*(tyerr) &= tyerr \\
f_*(\bot) &= \bot
\end{aligned}
$$

Más funciones auxiliares

Si $f \in V_{int} \to D$ entonces $f_{int} \in V \to D$ se define:

$$f_{int}(\iota_{int}\ i) = f\ i$$
$$f_{int}(\iota_{bool}\ b) = tyerr$$
$$f_{int}(\iota_{fun}\ f) = tyerr$$

Si $f \in V_{int} \to D$, entonces podemos componer las dos transformaciones

$$(f_{int})_* \in D \to D$$

que lo escribimos

$$f_{int*} \in D \to D$$

Más funciones auxiliares

Si $f \in V_{int} \to D$ entonces $f_{int} \in V \to D$ se define:

$$f_{int}(\iota_{int}\ i) = f\ i$$
$$f_{int}(\iota_{bool}\ b) = tyerr$$
$$f_{int}(\iota_{fun}\ f) = tyerr$$

Si $f \in V_{int} \to D$, entonces podemos componer las dos transformaciones

$$(f_{int})_* \in D \to D$$

que lo escribimos

$$f_{int*} \in D \to D$$

Más funciones auxiliares

Si $f \in V_\theta \to D$ entonces $f_\theta \in V \to D$ se define:

$$\begin{array}{rcll} f_\theta(\iota_\theta\, x) & = & f\ x & \\ f_\theta(\iota_{\theta'}\, y) & = & tyerr & \quad \text{si } \theta \neq \theta' \end{array}$$

En general: si $f \in V_\theta \to D$, entonces

$$\begin{array}{rcll} f_{\theta *} & \in & D \to D & \quad \text{satisfaciendo} \\ f_{\theta *}(\iota_{\underline{\theta}}\, x) & = & f\ x & \\ f_{\theta *}(\iota_{\underline{\theta'}}\, y) & = & tyerr & \quad \text{si } \theta \neq \theta' \\ f_{\theta *}(err) & = & err & \\ f_{\theta *}(tyerr) & = & tyerr & \\ f_{\theta *}(\perp) & = & \perp & \end{array}$$

Más funciones auxiliares

Si $f \in V_\theta \to D$ entonces $f_\theta \in V \to D$ se define:

$$\begin{array}{rcll} f_\theta(\iota_\theta\, x) & = & f\ x & \\ f_\theta(\iota_{\theta'}\, y) & = & tyerr & \quad \text{si } \theta \neq \theta' \end{array}$$

En general: si $f \in V_\theta \to D$, entonces

$$\begin{array}{rcll} f_{\theta *} & \in & D \to D & \qquad \text{satisfaciendo} \\ f_{\theta *}(\iota_{\underline{\theta}}\, x) & = & f\ x & \\ f_{\theta *}(\iota_{\underline{\theta'}}\, y) & = & tyerr & \quad \text{si } \theta \neq \theta' \\ f_{\theta *}(err) & = & err & \\ f_{\theta *}(tyerr) & = & tyerr & \\ f_{\theta *}(\perp) & = & \perp & \end{array}$$

Ecuaciones semánticas

$$Env = \langle var \rangle \to V$$
$$[\![\_]\!] \in \langle exp \rangle \to Env \to D$$

$$
\begin{array}{lcl}
[\![0]\!]\eta & = & \iota_{\underline{int}} 0 \\
[\![\textbf{true}]\!]\eta & = & \iota_{\underline{bool}} T \\
[\![-e]\!]\eta & = & (\lambda i \in V_{int}.\ \iota_{\underline{int}} \lfloor -i \rfloor)_{int*}([\![e]\!]\eta) \\
[\![e + e']\!]\eta & = & (\lambda i \in V_{int}.\ (\lambda i' \in V_{int}.\ \iota_{\underline{int}} \lfloor i + i' \rfloor)_{int*}([\![e']\!]\eta))_{int*}([\![e]\!]\eta) \\
 & \vdots & \\
[\![e/e']\!]\eta & = & (\lambda i \in V_{int}.\ (\lambda i' \in V_{int}. \\
 & & \quad \textbf{if } i' = 0 \textbf{ then } err \textbf{ else } \iota_{\underline{int}} \lfloor i/i' \rfloor)_{int*}([\![e']\!]\eta))_{int*}([\![e]\!]\eta)
\end{array}
$$

Ecuaciones semánticas

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

$[\![\_]\!] \in \langle exp \rangle \to Env \to D$

$$
\begin{array}{lcl}
[\![0]\!]\eta & = & \iota_{\underline{int}} 0 \\
[\![\textbf{true}]\!]\eta & = & \iota_{\underline{bool}} T \\
[\![-e]\!]\eta & = & (\lambda i \in V_{int}.\ \iota_{\underline{int}} \lfloor -i \rfloor)_{int*}([\![e]\!]\eta) \\
[\![e + e']\!]\eta & = & (\lambda i \in V_{int}.\ (\lambda i' \in V_{int}.\ \iota_{\underline{int}} \lfloor i + i' \rfloor)_{int*}([\![e']\!]\eta))_{int*}([\![e]\!]\eta) \\
 & \vdots & \\
[\![e / e']\!]\eta & = & (\lambda i \in V_{int}.\ (\lambda i' \in V_{int}. \\
 & & \quad \textbf{if } i' = 0 \textbf{ then } err \textbf{ else } \iota_{\underline{int}} \lfloor i / i' \rfloor)_{int*}([\![e']\!]\eta))_{int*}([\![e]\!]\eta)
\end{array}
$$

Ecuaciones semánticas

$$Env = \langle var \rangle \to V$$
$$[\![\_]\!] \in \langle exp \rangle \to Env \to D$$

$$
\begin{array}{lcl}
[\![0]\!]\eta & = & \iota_{\underline{int}} 0 \\
[\![\textbf{true}]\!]\eta & = & \iota_{\underline{bool}} T \\
[\![-e]\!]\eta & = & (\lambda i \in V_{int}.\ \iota_{\underline{int}} \lfloor -i \rfloor)_{int*}([\![e]\!]\eta) \\
[\![e + e']\!]\eta & = & (\lambda i \in V_{int}.\ (\lambda i' \in V_{int}.\ \iota_{\underline{int}} \lfloor i + i' \rfloor)_{int*}([\![e']\!]\eta))_{int*}([\![e]\!]\eta) \\
 & \vdots & \\
[\![e/e']\!]\eta & = & (\lambda i \in V_{int}.\ (\lambda i' \in V_{int}. \\
 & & \quad \textbf{if } i' = 0 \textbf{ then } err \textbf{ else } \iota_{\underline{int}} \lfloor i/i' \rfloor)_{int*}([\![e']\!]\eta))_{int*}([\![e]\!]\eta)
\end{array}
$$

Ecuaciones semánticas

$$Env = \langle var \rangle \to V$$
$$[\![\_]\!] \in \langle exp \rangle \to Env \to D$$

$$
\begin{array}{lcl}
[\![0]\!]\eta & = & \iota_{\underline{int}} 0 \\
[\![\textbf{true}]\!]\eta & = & \iota_{\underline{bool}} T \\
[\![-e]\!]\eta & = & (\lambda i \in V_{int}.\ \iota_{\underline{int}} \lfloor -i \rfloor)_{int*}([\![e]\!]\eta) \\
[\![e + e']\!]\eta & = & (\lambda i \in V_{int}.\ (\lambda i' \in V_{int}.\ \iota_{\underline{int}} \lfloor i + i' \rfloor)_{int*}([\![e']\!]\eta))_{int*}([\![e]\!]\eta) \\
 & \vdots & \\
[\![e/e']\!]\eta & = & (\lambda i \in V_{int}.\ (\lambda i' \in V_{int}. \\
 & & \quad \textbf{if } i' = 0 \textbf{ then } err \textbf{ else } \iota_{\underline{int}} \lfloor i/i' \rfloor)_{int*}([\![e']\!]\eta))_{int*}([\![e]\!]\eta)
\end{array}
$$

Ecuaciones semánticas

$$Env = \langle var \rangle \to V$$
$$[\![\_]\!] \in \langle exp \rangle \to Env \to D$$

$$
\begin{array}{lcl}
[\![v]\!]\eta & = & \iota_{norm}(\eta\, v) \\
[\![ee']\!]\eta & = & (\lambda f \in V_{fun}.\ (\lambda z \in V.\ f\ \ z)_*([\![e']\!]\eta))_{fun*}([\![e]\!]\eta) \\
 & = & (\lambda f \in V_{fun}.\ f_*([\![e']\!]\eta))_{fun*}([\![e]\!]\eta) \\
[\![\lambda v.e]\!]\eta & = & \iota_{\underline{fun}}(\lambda z \in V.\ [\![e]\!][\eta|v : z]) \\
 & & \\
[\![\mathbf{error}]\!]\eta & = & err \\
[\![\mathbf{typeerror}]\!]\eta & = & tyerr
\end{array}
$$

Ecuaciones semánticas

$$Env = \langle var \rangle \to V$$
$$[\![\_]\!] \in \langle exp \rangle \to Env \to D$$

$$\begin{array}{lcl}
[\![v]\!]\eta & = & \iota_{norm}(\eta\, v) \\
[\![ee']\!]\eta & = & (\lambda f \in V_{fun}.\ (\lambda z \in V.\ f\ \ z)_*([\![e']\!]\eta))_{fun*}([\![e]\!]\eta) \\
 & = & (\lambda f \in V_{fun}.\ f_*([\![e']\!]\eta))_{fun*}([\![e]\!]\eta) \\
[\![\lambda v.e]\!]\eta & = & \iota_{\underline{fun}}(\lambda z \in V.\ [\![e]\!][\eta|v:z]) \\
 & & \\
[\![\textbf{error}]\!]\eta & = & err \\
[\![\textbf{typeerror}]\!]\eta & = & tyerr
\end{array}$$

Ecuaciones semánticas

$$Env = \langle var \rangle \to V$$
$$[\![\_]\!] \in \langle exp \rangle \to Env \to D$$

$$\begin{array}{lcl}
[\![v]\!]\eta & = & \iota_{norm}(\eta\, v) \\
[\![ee']\!]\eta & = & (\lambda f \in V_{fun}.\ (\lambda z \in V.\ f\ \ z)_*([\![e']\!]\eta))_{fun*}([\![e]\!]\eta) \\
 & = & (\lambda f \in V_{fun}.\ f_*([\![e']\!]\eta))_{fun*}([\![e]\!]\eta) \\
[\![\lambda v.e]\!]\eta & = & \iota_{\underline{fun}}(\lambda z \in V.\ [\![e]\!][\eta|v : z])
\end{array}$$

$$\begin{array}{lcl}
[\![\textbf{error}]\!]\eta & = & err \\
[\![\textbf{typeerror}]\!]\eta & = & tyerr
\end{array}$$

Ecuaciones semánticas

$$Env = \langle var \rangle \to V$$
$$[\![\_]\!] \in \langle exp \rangle \to Env \to D$$

$$
\begin{array}{lcl}
[\![v]\!]\eta & = & \iota_{norm}(\eta\, v) \\
[\![ee']\!]\eta & = & (\lambda f \in V_{fun}.\ (\lambda z \in V.\ f\ \ z)_*([\![e']\!]\eta))_{fun*}([\![e]\!]\eta) \\
 & = & (\lambda f \in V_{fun}.\ f_*([\![e']\!]\eta))_{fun*}([\![e]\!]\eta) \\
[\![\lambda v.e]\!]\eta & = & \iota_{\underline{fun}}(\lambda z \in V.\ [\![e]\!][\eta|v:z])
\end{array}
$$

$$
\begin{array}{lcl}
[\![\textbf{error}]\!]\eta & = & err \\
[\![\textbf{typeerror}]\!]\eta & = & tyerr
\end{array}
$$

Semántica Denotacional Normal

Para el CLN:

$$D = V_{\perp}$$

$$V \;\simeq\; V_{fun} \qquad \text{con } V_{fun} = [D \to D]$$

Para el Lenguaje Aplicativo Normal (lo mismo que el eager):

$$D \;=\; (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_{\perp}$$

$$V \;\simeq\; V_{int} \;+\; V_{bool} \;+\; V_{fun}$$

Semántica Denotacional Normal

Para el CLN:

$$D = V_\perp$$

$$V \;\simeq\; V_{fun} \qquad \text{con } V_{fun} = [D \to D]$$

Para el Lenguaje Aplicativo Normal (lo mismo que el eager):

$$D \;=\; (V + \{\textbf{error}\} + \{\textbf{typeerror}\})_\perp$$

$$V \;\simeq\; V_{int} \;+\; V_{bool} \;+\; V_{fun}$$

Semántica Denotacional Normal: Valores

$$V \;\simeq\; V_{int} \;+\; V_{bool} \;+\; V_{fun}$$

$$V_{int} = \mathbf{Z}$$
$$V_{bool} = \{V, F\}$$
$$V_{fun} = [D \to D]$$

Los isomorfismos ϕ y ψ, lo mismo que las funciones auxiliares $\iota_\theta \in V_\theta \to V$, los resultados err y $tyerr$, la función $\iota_{norm} \in V \to D$, y las extensiones $f_* \in D \to D$ y $f_\theta \in V \to D$ se definen de la misma manera.

Semántica Denotacional Normal: Valores

$$V \;\simeq\; V_{int} \;+\; V_{bool} \;+\; V_{fun}$$

$$V_{int} = \mathbf{Z}$$
$$V_{bool} = \{V, F\}$$
$$V_{fun} = [D \to D]$$

Los isomorfismos ϕ y ψ, lo mismo que las funciones auxiliares $\iota_\theta \in V_\theta \to V$, los resultados err y $tyerr$, la función $\iota_{norm} \in V \to D$, y las extensiones $f_* \in D \to D$ y $f_\theta \in V \to D$ se definen de la misma manera.

Funciones semánticas

Entornos normales:

$Env = \langle var \rangle \to D$

Función semántica:

$[\![\_]\!]^{exp} \in \langle exp \rangle \to Env \to D$

Funciones semánticas

Entornos normales:

$Env = \langle var \rangle \to D$

Función semántica:

$[\![\_]\!]^{exp} \in \langle exp \rangle \to Env \to D$

Ecuaciones semánticas

La mayoría de las ecuaciones son las mismas. Vamos a señalar aquellas que presentan algún cambio.

Evaluación lazy de las expresiones booleanas:

¿Cuál es el problema con :
$[\![e \vee e']\!]\eta =$

$$(\lambda b \in V_{bool}.\ \textbf{if}\ b\ \textbf{then}\ \iota_{\underline{bool}}T$$
$$\textbf{else}\ (\lambda b' \in V_{bool}.\iota_{\underline{bool}}b')_{bool*}([\![e']\!]\eta)$$
$$)_{bool*}([\![e]\!]\eta)\ ?$$

$[\![e \vee e']\!]\eta = (\lambda b \in V_{bool}.\ \textbf{if}\ b\ \textbf{then}\ \iota_{\underline{bool}}T\ \textbf{else}\ [\![e']\!]\eta)_{bool*}([\![e]\!]\eta)$?

Ecuaciones semánticas

La mayoría de las ecuaciones son las mismas. Vamos a señalar aquellas que presentan algún cambio.

Evaluación lazy de las expresiones booleanas:

¿Cuál es el problema con :
$[\![e \vee e']\!] \eta =$

$$\begin{array}{l} (\lambda b \in V_{bool}.\ \textbf{if}\ b\ \textbf{then}\ \iota_{\underline{bool}} T \\ \qquad\qquad\qquad \textbf{else}\ (\lambda b' \in V_{bool}. \iota_{\underline{bool}} b')_{bool*}([\![e']\!] \eta) \\ )_{bool*}([\![e]\!] \eta) \end{array}$$?

$[\![e \vee e']\!] \eta = (\lambda b \in V_{bool}.\ \textbf{if}\ b\ \textbf{then}\ \iota_{\underline{bool}} T\ \textbf{else}\ [\![e']\!] \eta)_{bool*}([\![e]\!] \eta)$?

Ecuaciones semánticas

Cálculo lambda:

$$
\begin{array}{lcl}
[\![v]\!]\eta & = & \eta\ v \\
[\![ee']\!]\eta & = & (\lambda f \in V_{fun}.\ f\ [\![e']\!]\eta)_{fun*}([\![e]\!]\eta) \\
[\![\lambda v.e]\!]\eta & = & \iota_{\underline{fun}}(\lambda d \in D.\ [\![e]\!][\eta|v:d])
\end{array}
$$

Ecuaciones semánticas

Cálculo lambda:

$$
\begin{array}{lcl}
[\![v]\!]\eta & = & \eta\ v \\
[\![ee']\!]\eta & = & (\lambda f \in V_{fun}.\ f\ [\![e']\!]\eta)_{fun*}([\![e]\!]\eta) \\
[\![\lambda v.e]\!]\eta & = & \iota_{\underline{fun}}(\lambda d \in D.\ [\![e]\!][\eta|v:d])
\end{array}
$$

Ecuaciones semánticas

Cálculo lambda:

$$
\begin{array}{lcl}
[\![v]\!]\eta & = & \eta\ v \\
[\![ee']\!]\eta & = & (\lambda f \in V_{fun}.\ f\ [\![e']\!]\eta)_{fun*}([\![e]\!]\eta) \\
[\![\lambda v.e]\!]\eta & = & \iota_{\underline{fun}}(\lambda d \in D.\ [\![e]\!][\eta|v:d])
\end{array}
$$

Tuplas

$$\begin{array}{rcl} \langle exp \rangle & ::= & \langle \langle exp \rangle , ..., \langle exp \rangle \rangle \\ & & \langle exp \rangle \, . \, \langle tag \rangle \end{array}$$

$$\langle tag \rangle \quad ::= \quad \langle natconst \rangle$$

Los componentes de la tupla pueden ser de distintos tipos:

Ejemplo $M : \langle 2, \lambda\ x.\Delta\, \Delta, \lambda\ f\ x.f\ (f\ x) \rangle$

y las usamos accediendo a sus campos:
$(M.0\, M.1)\, M.2$

Tuplas

$$
\begin{array}{rcl}
\langle exp \rangle & ::= & \langle \langle exp \rangle, ..., \langle exp \rangle \rangle \\
 & & \langle exp \rangle . \langle tag \rangle
\end{array}
$$

$$
\langle tag \rangle \ ::= \ \langle natconst \rangle
$$

Los componentes de la tupla pueden ser de distintos tipos:

Ejemplo $M : \langle 2, \lambda\ x.\Delta\,\Delta, \lambda\ f\ x.f\,(f\,x) \rangle$

y las usamos accediendo a sus campos:
$(M.0\ M.1)\ M.2$

Tuplas

$$\begin{array}{rcl} \langle exp \rangle & ::= & \langle \langle exp \rangle, ..., \langle exp \rangle \rangle \\ & & \langle exp \rangle . \langle tag \rangle \end{array}$$

$$\langle tag \rangle \quad ::= \quad \langle natconst \rangle$$

Los componentes de la tupla pueden ser de distintos tipos:

Ejemplo M : $\langle 2, \lambda\ x.\Delta\,\Delta, \lambda\ f\ x.f\ (f\ x) \rangle$

y las usamos accediendo a sus campos:
$(M.0\ M.1)\ M.2$

Evaluación de las Tuplas

Formas canónicas:

$$\langle cnf \rangle \ ::= \ \langle intcnf \rangle \mid \langle boolcnf \rangle \mid \langle funcnf \rangle \mid \langle tuplecnf \rangle$$

Formas canónicas de las tuplas para la evaluación Eager:

$$\langle tuplecnf \rangle \ ::= \ \langle \langle cnf \rangle, ..., \langle cnf \rangle \rangle$$

Formas canónicas de tuplas para la evaluación Normal:

$$\langle tuplecnf \rangle \ ::= \ \langle \langle exp \rangle, ..., \langle exp \rangle \rangle$$

Evaluación de las Tuplas

Formas canónicas:

$$\langle cnf \rangle \quad ::= \quad \langle intcnf \rangle \mid \langle boolcnf \rangle \mid \langle funcnf \rangle \mid \langle tuplecnf \rangle$$

Formas canónicas de las tuplas para la evaluación Eager:

$$\langle tuplecnf \rangle \quad ::= \quad \langle \langle cnf \rangle , ..., \langle cnf \rangle \rangle$$

Formas canónicas de tuplas para la evaluación Normal:

$$\langle tuplecnf \rangle \quad ::= \quad \langle \langle exp \rangle , ..., \langle exp \rangle \rangle$$

Evaluación de las Tuplas

Formas canónicas:

$$\langle cnf \rangle \quad ::= \quad \langle intcnf \rangle \mid \langle boolcnf \rangle \mid \langle funcnf \rangle \mid \langle tuplecnf \rangle$$

Formas canónicas de las tuplas para la evaluación Eager:

$$\langle tuplecnf \rangle \quad ::= \quad \langle \langle cnf \rangle, ..., \langle cnf \rangle \rangle$$

Formas canónicas de tuplas para la evaluación Normal:

$$\langle tuplecnf \rangle \quad ::= \quad \langle \langle exp \rangle, ..., \langle exp \rangle \rangle$$

Reglas para la evaluación Eager (Tuplas)

$$\frac{e_0 \Rightarrow_E z_0 \quad \dots \quad e_{n-1} \Rightarrow_E z_{n-1}}{\langle e_0, \dots, e_{n-1} \rangle \ \Rightarrow_E \ \langle z_0, \dots, z_{n-1} \rangle}$$

$$\frac{e \Rightarrow_E \langle z_0, \dots, z_{n-1} \rangle}{e.\lfloor k \rfloor \Rightarrow_E \ z_k} \quad (k < n)$$

Reglas para la evaluación Eager (Tuplas)

$$\frac{e_0 \Rightarrow_E z_0 \quad \ldots \quad e_{n-1} \Rightarrow_E z_{n-1}}{\langle e_0, \ldots, e_{n-1} \rangle \;\Rightarrow_E\; \langle z_0, \ldots, z_{n-1} \rangle}$$

$$\frac{e \Rightarrow_E \langle z_0, \ldots, z_{n-1} \rangle}{e.\lfloor k \rfloor \Rightarrow_E \; z_k} \quad (k < n)$$

Reglas para la evaluación Normal (Tuplas)

No hay regla para la tupla $\langle e_0, \ldots, e_{n-1} \rangle$ porque es una forma canónica.

$$\frac{e \Rightarrow_N \langle e_0, \ldots, e_{n-1} \rangle \qquad e_k \Rightarrow_N z}{e.\lfloor k \rfloor \Rightarrow_N \; z} \quad (k < n)$$

Reglas para la evaluación Normal (Tuplas)

No hay regla para la tupla $\langle e_0, ..., e_{n-1} \rangle$ porque es una forma canónica.

$$\frac{e \Rightarrow_N \langle e_0, ..., e_{n-1} \rangle \qquad e_k \Rightarrow_N z}{e.\lfloor k \rfloor \Rightarrow_N \ z} \quad (k < n)$$

Semántica denotacional de las tuplas

$$V \;\simeq\; V_{int} \;+\; V_{bool} \;+\; V_{fun} \;+\; V_{tuple}$$

Semántica denotacional Eager:

$$\begin{aligned} V_{tuple} = V^* = & \bigcup_{n \in \mathbb{N}} V^n \\ = & \{\langle\rangle\} \cup \{\langle v\rangle | v \in V\} \cup \ldots \{\langle v_1, \ldots, v_n\rangle | v_i \in V\} \end{aligned}$$

Notar que $\langle \iota_{tuple} \langle \iota_{int} 2, \iota_{bool} T\rangle\rangle \in V_{tuple}$

Semántica denotacional Normal:

$V_{tuple} = D^*$

Semántica denotacional de las tuplas

$V \;\simeq\; V_{int} \;+\; V_{bool} \;+\; V_{fun} \;+\; V_{tuple}$

Semántica denotacional Eager:

$$\begin{aligned} V_{tuple} = V^* = & \bigcup_{n \in \mathbb{N}} V^n \\ = & \{\langle\rangle\} \cup \{\langle v \rangle | v \in V\} \cup \ldots \{\langle v_1, \ldots, v_n \rangle | v_i \in V\} \end{aligned}$$

Notar que $\langle \iota_{tuple} \langle \iota_{int} 2, \iota_{bool} T \rangle \rangle \in V_{tuple}$

Semántica denotacional Normal:

$V_{tuple} = D^*$

Semántica denotacional de las tuplas

$V \;\simeq\; V_{int} \;+\; V_{bool} \;+\; V_{fun} \;+\; V_{tuple}$

Semántica denotacional Eager:

$$
\begin{aligned}
V_{tuple} = V^* = & \bigcup_{n \in \mathbb{N}} V^n \\
= & \{\langle\rangle\} \cup \{\langle v \rangle | v \in V\} \cup \ldots \{\langle v_1, \ldots, v_n \rangle | v_i \in V\}
\end{aligned}
$$

Notar que $\langle \iota_{tuple} \langle \iota_{int} 2, \iota_{bool} T \rangle \rangle \in V_{tuple}$

Semántica denotacional Normal:

$V_{tuple} = D^*$

Ecuaciones semánticas Eager

$$[\![\langle e_0, ..., e_{n-1}\rangle]\!]\eta =$$

$$(\lambda z_0 \in V.$$

$$(\lambda z_1 \in V.$$

$$...(\lambda z_{n-1} \in V.\ \iota_{\underline{tuple}}\langle z_0, ..., z_{n-1}\rangle)_*([\![e_{n-1}]\!])...$$

$$)_*([\![e_1]\!]\eta)$$

$$)_*([\![e_0]\!]\eta)$$

Ecuaciones semánticas Eager

$$[\![e.\lfloor k \rfloor]\!]\eta =$$

$$(\lambda t \in V_{tuple}.$$

$$\quad \text{if } k \geq |t| \text{ then } tyerr \text{ else } \iota_{norm} t_k$$

$$)_{tuple*}([\![e]\!]\eta)$$

Ecuaciones semánticas Normales

$$[\![\langle e_0, ..., e_{n-1}\rangle]\!]\eta = \langle [\![e_0]\!]\eta, ..., [\![e_{n-1}]\!]\eta\rangle$$

$$[\![e.\lfloor k\rfloor]\!]\eta =$$

$$(\lambda t \in V_{tuple}.$$

$$\text{if } k \geq |t| \text{ then } tyerr \text{ else } t_k$$

$$)_{tuple*}([\![e]\!]\eta)$$

Ecuaciones semánticas Normales

$$[\![\langle e_0, ..., e_{n-1}\rangle]\!]\eta = \langle [\![e_0]\!]\eta, ..., [\![e_{n-1}]\!]\eta\rangle$$

$$[\![e.\lfloor k\rfloor]\!]\eta =$$

$$(\lambda t \in V_{tuple}.$$

$$\text{if } k \geq |t| \text{ then } tyerr \text{ else } t_k$$

$$)_{tuple*}([\![e]\!]\eta)$$

Definiciones locales y patrones

$$
\begin{array}{rcl}
\langle exp \rangle & ::= & \textbf{let}\ \langle pat \rangle \equiv \langle exp \rangle, \ldots, \langle pat \rangle \equiv \langle exp \rangle\ \textbf{in}\ \langle exp \rangle \\
& & \lambda \langle pat \rangle . \langle exp \rangle
\end{array}
$$

$$
\langle pat \rangle \quad ::= \quad \langle var \rangle \,|\, \langle\, \langle pat \rangle, \ldots, \langle pat \rangle \,\rangle
$$

Por ejemplo, podemos escribir

$$\lambda\langle\, u, \langle\, v, w \rangle\rangle . uwv$$

en vez de

$$\lambda t.(t.0)(t.1.1)(t.1.0)$$

$$(\lambda\langle\, u, \langle\, v, w \rangle\rangle . uwv)\, \langle M, \langle N, P \rangle\rangle \text{"}{\Rightarrow}\text{"}\, M\, P\, N$$

Definiciones locales y patrones

$$
\begin{array}{lcl}
\langle exp \rangle & ::= & \textbf{let} \ \langle pat \rangle \equiv \langle exp \rangle , \ldots , \langle pat \rangle \equiv \langle exp \rangle \ \textbf{in} \ \langle exp \rangle \\
 & & \lambda \langle pat \rangle . \langle exp \rangle
\end{array}
$$

$$
\langle pat \rangle \quad ::= \quad \langle var \rangle \,|\langle \, \langle pat \rangle , \ldots , \langle pat \rangle \, \rangle
$$

Por ejemplo, podemos escribir

$$\lambda\langle\, u, \langle\, v, w\rangle\rangle.uwv$$

en vez de

$$\lambda t.(t.0)(t.1.1)(t.1.0)$$

$$(\lambda\langle\, u, \langle\, v, w\rangle\rangle.uwv)\,\langle M, \langle N, P\rangle\rangle\text{"}\Rightarrow\text{"}M\,P\,N$$

Definiciones locales y patrones

$$\begin{array}{rcl}\langle exp\rangle & ::= & \textbf{let}\ \ \langle pat\rangle \equiv \langle exp\rangle\,, \ldots, \langle pat\rangle \equiv \langle exp\rangle\ \ \textbf{in}\ \ \langle exp\rangle \\ & & \lambda\,\langle pat\rangle\,.\,\langle exp\rangle\end{array}$$

$$\langle pat\rangle \quad ::= \quad \langle var\rangle \,|\langle\, \langle pat\rangle\,, \ldots, \langle pat\rangle\,\rangle$$

Por ejemplo, podemos escribir

$$\lambda\langle\, u, \langle\, v, w\rangle\rangle.uwv$$

en vez de

$$\lambda t.(t.0)(t.1.1)(t.1.0)$$

$$(\lambda\langle\, u, \langle\, v, w\rangle\rangle.uwv)\,\langle M, \langle N, P\rangle\rangle\text{"}\Rightarrow\text{"}M\;P\;N$$

Se definen como abreviaturas (azucar sintactico):

$$\lambda\langle\, p_1, \ldots, p_n \,\rangle.e \quad =$$

$$\lambda v.\ \textbf{let}\ p_1 \equiv v.0, \ldots, p_n \equiv v.[n-1]\ \textbf{in}\ e$$

donde v es una variable nueva (no ocurre libre en e ni en ninguno de los patrones).

$$\textbf{let}\ p_1 \equiv e_1, \ldots, p_n \equiv e_n\ \textbf{in}\ e \quad =$$

$$(\lambda p_1 \ldots \lambda p_n.e)e_1 \ldots e_n$$

Se definen como abreviaturas (azucar sintactico):

$$\lambda\langle\, p_1, \ldots, p_n \,\rangle.e \quad =$$

$$\lambda v.\ \textbf{let}\ p_1 \equiv v.0, \ldots, p_n \equiv v.[n-1]\ \textbf{in}\ e$$

donde v es una variable nueva (no ocurre libre en e ni en ninguno de los patrones).

$$\textbf{let}\ p_1 \equiv e_1, \ldots, p_n \equiv e_n\ \textbf{in}\ e \quad =$$

$$(\lambda p_1 \ldots \lambda p_n.e)e_1 \ldots e_n$$

Aplicando repetidamente estas dos transformaciones podemos eliminar los patrones que no sean variables y las definiciones locales (let) obteniendo una expresión cuya semántica ya está definida.

Tener en cuenta que cuando $n = 0$, **let** $p_1 \equiv e_1, \ldots, p_n \equiv e_n$ **in** e quedará **let in** e, esto en realidad es directamente la expresión e

Aplicando repetidamente estas dos transformaciones podemos eliminar los patrones que no sean variables y las definiciones locales (let) obteniendo una expresión cuya semántica ya está definida.

Tener en cuenta que cuando $n = 0$, **let** $p_1 \equiv e_1, \ldots, p_n \equiv e_n$ **in** e quedará **let in** e, esto en realidad es directamente la expresión e

Recursión

$$\begin{array}{rlr} \langle exp \rangle ::= & \textbf{letrec}\, \langle var \rangle \equiv \lambda\, \langle var \rangle \,.\, \langle exp \rangle \ \textbf{ in } \ \langle exp \rangle & \text{(Eval. Eager)} \\ & \textbf{rec} \ \langle exp \rangle & \text{(Eval. Normal)} \end{array}$$

El letrec permite hacer definiciones recursivas como

$$\textbf{letrec}\ \textit{fact} \equiv \lambda n.\ \textbf{if}\ n = 0\ \textbf{then}\ 1\ \textbf{else}\ n * \textit{fact}(n-1)\ \textbf{in}\ \textit{fact}\ 2$$

Regla para la Evaluación Eager

$$\frac{(e/v \mapsto \lambda u.e_0^*) \Rightarrow_E z}{\textbf{letrec } v \equiv \lambda u.e_0 \textbf{ in } e \;\Rightarrow_E\; z}$$

donde

$$e_0^* \;=\; \textbf{letrec } v \equiv \lambda u.e_0 \textbf{ in } e_0$$

Regla para la Evaluación Normal

$$\frac{e\ (\textbf{rec}\ e) \Rightarrow_N z}{\textbf{rec}\ e \Rightarrow_N z}$$

Semántica Denotacional Eager (Recursión)

$$[\![\textbf{letrec } v \equiv \lambda u.e_0 \textbf{ in } e\,]\!]\eta \;=\; [\![e]\!][\eta|v : \iota_{fun} g]$$

Semántica Denotacional Eager (Recursión)

$$[\![\textbf{letrec } v \equiv \lambda u.e_0 \textbf{ in } e\ ]\!]\eta \ = \ [\![e]\!][\eta|v : \iota_{fun} g]$$

donde g es el menor punto fijo de

$$F\ f\ z = [\![e]\!][\eta|v : \iota_{fun} f,\ u : z]$$

Semántica Denotacional Eager (Recursión)

$$[\![\textbf{letrec } v \equiv \lambda u.e_0 \textbf{ in } e\ ]\!]\eta \ = \ [\![e]\!][\eta|v : \iota_{fun} g]$$

donde g es el menor punto fijo de

$$F\ f\ z = [\![e]\!][\eta|v : \iota_{fun} f,\ u : z]$$

o sea

$$g = \mathbf{Y}_{V_{fun}}\ F \qquad\qquad \mathbf{Y}_{V_{fun}} F \ = \ \sqcup_i F^i \bot$$

Semántica Denotacional Eager (Recursión)

$$[\![\textbf{letrec } v \equiv \lambda u.e_0 \textbf{ in } e\,]\!]\eta \;=\; [\![e]\!][\eta|v:\iota_{fun} g]$$

donde g es el menor punto fijo de

$$F\ f\ z = [\![e]\!][\eta|v:\iota_{fun} f,\ u:z]$$

o sea

$$g = \mathbf{Y}_{V_{fun}}\ F \qquad\qquad \mathbf{Y}_{V_{fun}} F \;=\; \sqcup_i F^i \bot$$

$$g = \mathbf{Y}_{V_{fun}}(\lambda f \in V_{fun}.\lambda z \in V.[\![e]\!][\eta|v:\iota_{fun} f,\ u:z])$$

Semántica Denotacional Normal (Recursión)

$$[\![\textbf{rec } e]\!]\eta \;=\; (\lambda f \in V_{fun}.\, \mathbf{Y}_D f)_{fun*}([\![e]\!]\eta)$$

donde $\mathbf{Y}_D$ es el operador de menor punto fijo

$$\mathbf{Y}_D f \;=\; \sqcup_i f^i \bot$$