PLP - Práctica 3: Inferencia de tipos

Zamboni, Gianfranco

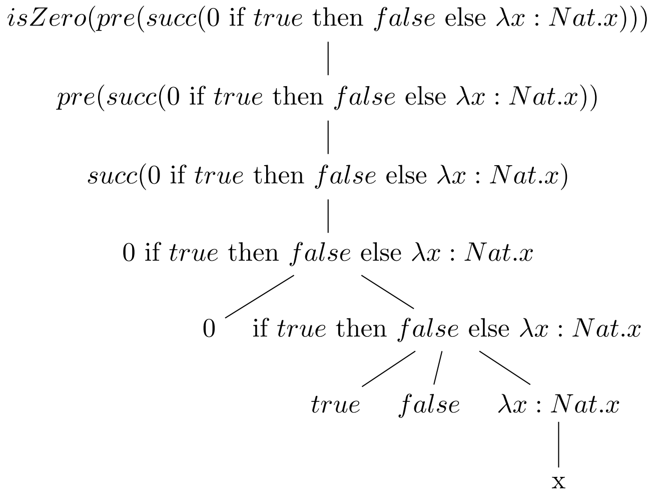
19 de febrero de 2018

Sintaxis

2.1. Ejercicio 1

Expresiones de términos	Expresiones de tipo	No válidas
x	$Bool$	M
$x\ x$	$Bool \to Bool$	$M\ M$
$true\ false$	$Bool \to Bool \to Nat$	$\lambda x.isZero(x)$
$true\ succ(true\ false)$	$(Bool \to Bool) \to Nat$	$\lambda x : \sigma.succ(x)$
$\lambda x : Bool.succ(x)$		$\lambda x : \text{if } true \text{ then } Bool \text{ else } Nat.x$
$\lambda x : Bool.\text{if } 0 \text{ then } true \text{ else } 0\ succ(true)$		σ
		$succ\ true$

2.2. Ejercicio 2

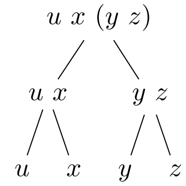


2.3. Ejercicio 3

1. $\lambda x : Nat.succ((\lambda x : Nat.x)\ x)$

2. En el término $\lambda x_1 : Nat.succ(x_2)$, x_1 no aparece como subtérmino.

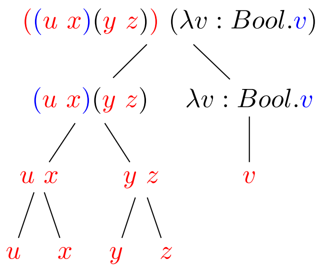
3. La expresión $x\ (y\ z)$ no sucede en la expresión $u\ x\ (y\ z)$



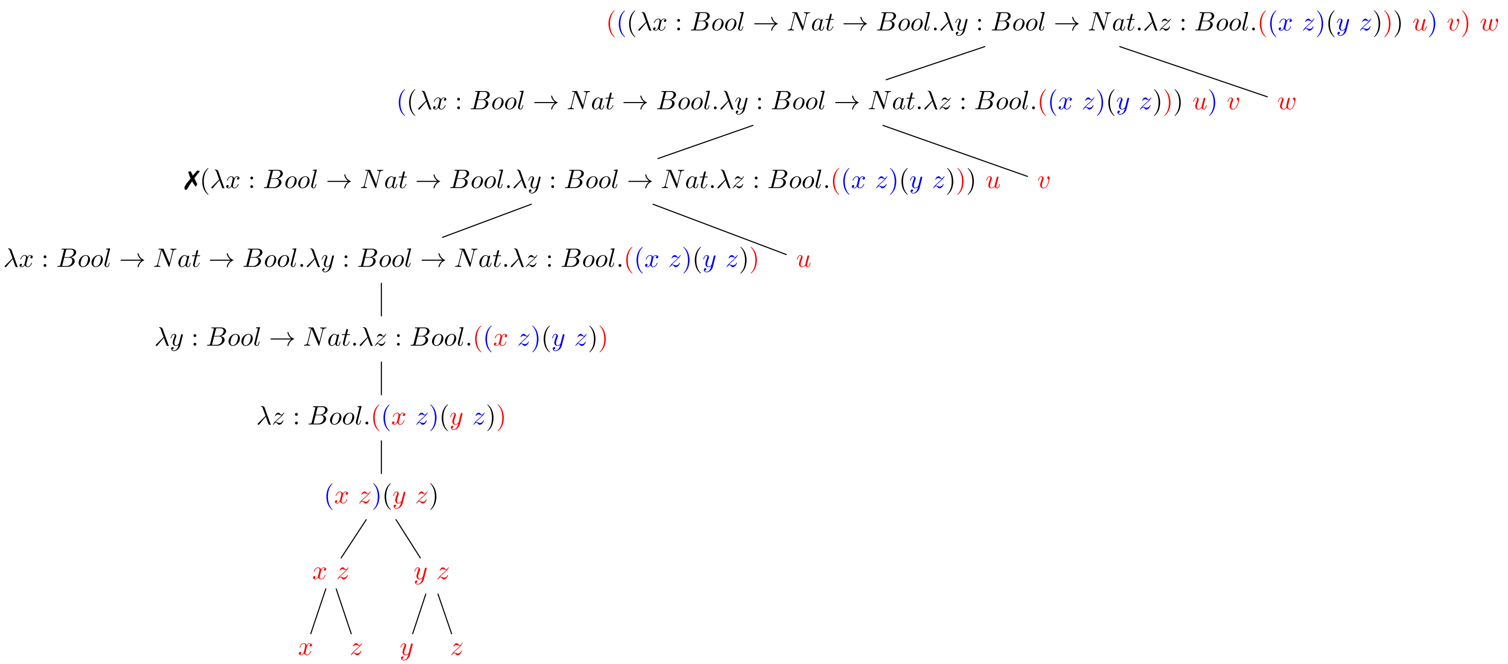
2.4. Ejercicio 4

Marcamos con azul las variables ligadas y con rojo las variables libres

a)



b) En esta expresión aparece $(\lambda x : Bool \to Nat \to Bool.\lambda y : Bool \to Nat.\lambda z : Bool.x\ z\ (y\ z))\ u$, la marco con una ✗ cuando aparece.



c)

$$((w\ (\lambda x : Bool \to Nat \to Bool.\lambda y : Bool \to Nat.\lambda z : Bool.((x\ z)\ (y\ z))))\ u)\ v$$

$$(w\ (\lambda x : Bool \to Nat \to Bool.\lambda y : Bool \to Nat.\lambda z : Bool.((x\ z)\ (y\ z))))\ u \qquad v$$

$$w\ (\lambda x : Bool \to Nat \to Bool.\lambda y : Bool \to Nat.\lambda z : Bool.((x\ z)\ (y\ z))) \qquad u$$

$$w \qquad \lambda x : Bool \to Nat \to Bool.\lambda y : Bool \to Nat.\lambda z : Bool.((x\ z)\ (y\ z))$$

$$\lambda y : Bool \to Nat.\lambda z : Bool.((x\ z)\ (y\ z))$$

$$\lambda z : Bool.((x\ z)\ (y\ z))$$

$$(x\ z)\ (y\ z)$$

$$x\ z \qquad y\ z$$

$$x \quad z \quad y \quad z$$

Tipado

2.5. Ejercicio 5

1)

$$\dfrac{\dfrac{}{\emptyset \triangleright true : Bool}\ \text{T-True} \qquad \dfrac{}{\emptyset \triangleright 0 : Nat}\ \text{T-Zero} \qquad \dfrac{\dfrac{}{\emptyset \triangleright 0 : Nat}\ \text{T-Zero}}{\emptyset \triangleright succ(0) : Nat}\ \text{T-Succ}}{\emptyset \triangleright \text{if } true \text{ then } 0 \text{ else } succ(0) : Nat}$$

2) En la siguiente demostración $\Gamma = \{x : Nat,\ y : Bool\}$

$$\dfrac{\dfrac{}{\Gamma \triangleright true : Bool}\ \text{T-True} \qquad \dfrac{}{\Gamma \triangleright false : Bool}\ \text{T-False} \qquad \dfrac{\dfrac{\dfrac{z : Bool \in \Gamma, z : Bool}{\Gamma, z : Bool \triangleright z : Bool}\ \text{T-Var}}{\Gamma \triangleright \lambda z : Bool.z : Bool \to Bool}\ \text{T-Abs} \qquad \dfrac{}{\Gamma \triangleright true : Bool}\ \text{T-True}}{\Gamma \triangleright (\lambda z : Bool.z)\ true : Bool}\ \text{T-App}}{\Gamma \triangleright \text{if } true \text{ then } false \text{ else } (\lambda z : Bool.z)\ true : Bool}\ \text{T-If}$$

3)

$$\dfrac{\dfrac{\color{red}{\emptyset, x : Bool \triangleright x : Bool \Rightarrow \emptyset \triangleright \lambda x : Bool.x : Bool \to \tau}}{\emptyset \triangleright \lambda x : Bool.x : Bool}\ \text{T-Abs} \qquad \emptyset \triangleright 0 : Nat \qquad \emptyset \triangleright succ(0) : Nat}{\emptyset \triangleright \text{if } \lambda x : Bool.x \text{ then } 0 \text{ else } succ(0) : Nat}\ \text{T-If}$$

4) En la próxima demostración $\Gamma = \{x : Bool \to Nat, y : Bool\}$

$$\dfrac{\dfrac{x : Bool \to Nat \in \Gamma}{\Gamma \triangleright x : Bool \to Nat}\ \text{T-Var} \qquad \dfrac{y : Bool \in \Gamma}{\Gamma \triangleright y : Bool}\ \text{T-Var}}{\Gamma \triangleright \text{if } \lambda x : Bool.x \text{ then } 0 \text{ else } succ(0) : Nat}\ \text{T-App}$$

2.6. Ejercicio 6

1)

$$\dfrac{\dfrac{\sigma = Nat}{\emptyset \triangleright 0 : \sigma}\,\text{T-Zero}}{\emptyset \triangleright succ(0) : \sigma}\,\text{T-Succ} \quad \Rightarrow \sigma = Nat$$

2)

$$\dfrac{\dfrac{}{\emptyset \triangleright succ(0) : Nat \Rightarrow \sigma = Bool}\,\text{Ejercicio 2.6}}{\emptyset \triangleright isZero(succ(0)) : \sigma}\,\text{T-IsZero} \quad \Rightarrow \sigma = Bool$$

3)

$$\dfrac{\dfrac{\dfrac{}{\emptyset \triangleright true : Bool}\,\text{T-True} \quad \dfrac{}{\emptyset \triangleright false : Bool}\,\text{T-True} \quad \dfrac{}{\emptyset \triangleright false : Bool}\,\text{T-True}}{\Gamma \triangleright \text{if } true \text{ then } false \text{ else } false : Bool}\,\text{T-If} \quad \dfrac{\sigma = Nat}{\emptyset \triangleright 0 : \sigma}\,\text{T-Zero} \quad \dfrac{\sigma = Nat}{\emptyset \triangleright succ(0) : \sigma}\,\text{Ejercicio 2.6}}{\emptyset \triangleright \text{if if } true \text{ then } false \text{ else } false \text{ then } 0 \text{ else } succ(0) : \sigma}\,\text{T-If} \quad \Rightarrow \sigma = Nat$$

2.7. Ejercicio 7

1) En la próxima demostración $\Gamma = \{x : \sigma\}$

$$\dfrac{\dfrac{\dfrac{x : Nat \in \Gamma}{\Gamma \triangleright x : Nat}\,\text{T-Var}}{\Gamma \triangleright succ(x) : Nat}\,\text{T-Succ}}{\Gamma \triangleright isZero(succ(x)) : \tau}\,\text{T-Succ}$$

Entonces, $\sigma = Nat$ y $\tau = Bool$

2)

$$\dfrac{\dfrac{\{x : \sigma\} \triangleright x : \sigma}{\emptyset \triangleright \lambda x : \sigma.x : \sigma \to \sigma}\,\text{T-Abs} \quad \dfrac{\{y : Bool\} \triangleright 0 : Nat}{\emptyset \triangleright \lambda y : Bool.0 : \sigma}\,\text{T-Abs}}{\emptyset \triangleright (\lambda x : \sigma.x)\ (\lambda y : Bool.0) : \sigma}\,\text{T-App}$$

El árbol de la segunda abstracción nos dice que $\sigma = Bool \to Nat$. Por lo que la primera abstracción seria la función identidad de funciones del tipo $Bool \to Nat$.

3) $\Gamma = \{y : \tau\}$

$$\dfrac{\dfrac{\Gamma, x : \sigma \triangleright x : \sigma \Rightarrow \Gamma, x : \sigma \triangleright \lambda x : \sigma.x : \sigma \to Bool}{\Gamma \triangleright \lambda x : \sigma.x : Bool}\,\text{T-Abs} \quad \Gamma \triangleright y : \sigma \quad \Gamma \triangleright succ(0) : \sigma}{\Gamma \triangleright \text{if } (\lambda x : \sigma.x) \text{ then } y \text{ else } succ(0) : \sigma}\,\text{T-If}$$

Entonces, la expresión no es tipable.

4) $\Gamma = \{x : \sigma\}$

$$\dfrac{\dfrac{x : \sigma_1 \to \tau \in \Gamma \Rightarrow \sigma = \sigma_1 \to \tau}{\Gamma \triangleright x : \sigma_1 \to \tau}\text{ T-Var} \qquad \dfrac{y : \sigma_1 \in \Gamma}{\Gamma \triangleright y : \sigma_1}\text{ T-Var}}{\Gamma \triangleright x\ y : \tau}\text{ T-App}$$

En este caso, $\sigma = \sigma_1 \to \tau$ para cualquier τ y cualquier σ_1. Sin embargo, el juicio de tipado no es derivable ya que el sistema de tipado no nos permite asegurar nada sobre el tipo de y. Acá no podemos hacer juicio de tipado porque

no podemos asegurar que y sea de tipo σ_1

5) $\Gamma = \{x : \sigma, y : \tau\}$

$$\dfrac{\dfrac{x : \sigma_1 \to \tau \in \Gamma \Rightarrow \sigma = \sigma_1 \to \tau}{\Gamma \triangleright x : \sigma_1 \to \tau}\text{ T-Var} \qquad \dfrac{y : \sigma_1 \in \Gamma \Rightarrow \sigma_1 = \tau}{\Gamma \triangleright y : \sigma_1}\text{ T-Var}}{\Gamma \triangleright x\ y : \tau}\text{ T-App}$$

Entonces $\sigma = \tau \to \tau$ para cualquier tipo τ

2.8. Ejercicio 8

La expresión más simple que se me ocurre es $true\ false$.

6) $\Gamma = \{x : \sigma\}$

$$\dfrac{\dfrac{x : Bool \to \tau \in \Gamma \Rightarrow \sigma = Bool \to \tau}{\Gamma \triangleright x : Bool \to \tau}\text{ T-Var} \qquad \Gamma \triangleright true : Bool}{\Gamma \triangleright x\ true : \tau}\text{ T-App}$$

Entonces $\sigma = Bool \to \tau$ para cualquier tipo τ

7) $\Gamma = \{x : \sigma\}$

$$\dfrac{\dfrac{x : Bool \to \sigma \in \Gamma}{\Gamma \triangleright x : Bool \to \sigma}\text{ T-Var} \qquad \Gamma \triangleright true : Bool}{\Gamma \triangleright x\ true : \sigma}\text{ T-App}$$

Tenemos que x debe ser de tipo σ y $Bool \to \sigma$ al mismo tiempo, por lo que no es posible dar un tipo a esta expresión

8) $\Gamma = \{x : \sigma\}$

$$\dfrac{\dfrac{x : \sigma_1 \to \tau \in \Gamma}{\Gamma \triangleright x : \sigma_1 \to \tau}\text{ T-Var (1)} \qquad \dfrac{x : \sigma_1 \in \Gamma}{\Gamma \triangleright x : \sigma_1}\text{ T-Var (2)}}{\Gamma \triangleright x\ x : \tau}\text{ T-App}$$

Tenemos que x debe ser de tipo σ y $Bool \to \sigma$ al mismo tiempo, por lo que no es posible dar un tipo a esta expresión

2.9. Ejercicio 9

El juicio de tipado puede ser $\Gamma \triangleright \lambda x : \sigma . x : \sigma \to \sigma$

Con T-Abs tenemos:
$$\dfrac{\dfrac{x : \sigma \in \Gamma, x : \sigma}{\Gamma, x : \sigma \triangleright x : \sigma}\text{T-Var}}{\Gamma \triangleright \lambda x : \sigma . x : \sigma \to \sigma}\text{T-Abs}$$

Con T-Abs2:
$$\dfrac{\dfrac{x : \sigma \in \Gamma}{\Gamma \triangleright x : \sigma}\text{T-Var}}{\Gamma \triangleright \lambda x : \sigma . x : \sigma \to \sigma}\text{T-Abs2}$$

No hay nada que nos asegure que $x : \sigma$ pertenece a Γ

Semántica

2.10. Ejercicio 10

1) $(\lambda y : \sigma . x\ (\lambda x : \tau . x))\{x \leftarrow (\lambda y : \rho . x\ y)\} \underset{y=w;\ x=z}{\overset{\alpha-eq}{\rightsquigarrow}} (\lambda w : \sigma . x\ (\lambda z : \tau . z))\{x \leftarrow (\lambda y : \rho . x\ y)\} \overset{def}{=} \lambda w : \sigma . (\lambda y : \rho . x\ y)\ (\lambda z : \tau . z)$

2)

$$(y\ (\lambda v : \sigma . x\ v))\{x \leftarrow (\lambda y : \tau . v\ y)\} \overset{def}{=} y\{x \leftarrow (\lambda y : \tau . v\ y)\}\ (\lambda v : \sigma . x\ v)\{x \leftarrow (\lambda y : \tau . v\ y)\} \overset{def}{=} y\ (\lambda v : \sigma . x\ v)\{x \leftarrow (\lambda y : \tau . v\ y)\}$$
$$\underset{v=z}{\overset{\alpha-eq}{\rightsquigarrow}} y\ (\lambda z : \sigma . x\ z)\{x \leftarrow (\lambda y : \tau . v\ y\}) \overset{def}{=} y\ (\lambda z : \sigma . (\lambda y : \tau . v\ y)\ z)$$

2.11. Ejercicio 11

Es un valor	**No es un valor**
$\lambda x : Bool . \underline{2}$	$(\lambda x : Bool . x)\ true$
$\lambda x : Bool . pred(\underline{2})$	x
$\lambda y : Nat . (\lambda x : Bool . pred(\underline{2}))\ true$	
$succ(succ(0))$	

2.12. Ejercicio 12

Los programas en rojo son los que dan error y los programas en azul los que son valores, el resto no están en forma normal.

Programas	No Programas
$(\lambda x : Bool.x)\ true$	$\lambda x : Nat.pred(succ(y))$
$\lambda x : Nat.pred(succ(x))$	$(\lambda x : Bool.pred(isZero(x)))\ true$
$(\lambda f : Nat \to Bool.f\ 0)\ (\lambda x : Nat.isZero(x))$	$(\lambda f : Nat \to Bool.x)\ (\lambda x : Nat.isZero(x))$
$(\lambda f : Nat \to Bool.f\ pred(0))\ (\lambda x : Nat.isZero(x))$	
fix $(\lambda y : Nat.succ(y))$	
letrec $f = \lambda x : Nat.succ(f\ x)$ in $f\ 0$	

2.13. Ejercicio 13

1) Cuando definimos $\to$, lo hacemos como una función deterministica que dada una expresión, siempre hay una única que regla que aplica para dar un paso en la reducción. Por lo que una expresión siempre reduce a la misma expresión, en un paso.

2) Al haber, en cada paso, un único camino por el que reducir, si reducimos $M \twoheadrightarrow N$ y $M \twoheadrightarrow N'$, ambas reducciones en n pasos, entonces $N = N'$

3) No es cierto, con la mayoría de las expresiones seguimos un camino de reducción que nos llevará a una forma normal. En estos casos, M' y M'' no pueden ser iguales, sin embargo, hay expresiones recursivas como fix $\lambda x : Bool- > Bool.x$ que cuando se las trata de reducir siempre evalúan a si mismas y, entonces vale siempre que $M' = M''$.

2.14. Ejercicio 14

1) Cuando $M \twoheadrightarrow 0$, tenemos que $succ(pred(0)) = 1$ y que $pred(succ(0)) = 0$, por lo que no siempre es lo mismo evaluar estas dos expresiones.

2) $isZero(succ(M)) \twoheadrightarrow false$ solo cuando M es de tipo Nat y su forma normal es 0.

3) Como dice el enunciado, hay infinitas expresiones que evalúan a cero, por lo tanto $isZero(pred(M))$ vale cuando $M \twoheadrightarrow 0$.

2.15. Ejercicio 15

1) Con la nueva regla, las abstracciones $\lambda x : \sigma.M$ pasan a ser expresiones reducibles mientras M sea una expresión reducible, por lo que debemos considerarla un valor solo cuando su expresión interna es un valor.

2) Además, esta nueva regla nos da un nuevo camino para reducirla cuando es aplicada a un valor, es decir, en una expresión del tipo $\lambda x : \sigma.M\ N$, podemos elegir entre usar esta regla o la regla E-Abs/β, por lo que hay que modificar esta regla para que solo sea apicable cuando M está en su forma normal F, o sea no es posible seguir reduciendola. Entonces:

$$V \ ::= \ ... \mid \lambda x : \sigma.V$$

$$\frac{}{\lambda x : \sigma.F\ V \to F\{x \leftarrow V\}}\text{(E-Abs/}\beta\text{)}$$

3) Ahora, si tratamos de reducir $(\lambda x : Nat \to Nat.x\ \underline{23})\ (\lambda x : Nat.0)$ nos encontraremos con que la expresión dentro de la primer abstracción es reducible pues $x\ \underline{23}$ no es un valor. Sin embargo, x puede ser cualquier función de tipo $Nat \to Nat$, o sea que tiene una variable libre, pero no sabemos como ni cuanta veces ocurre dentro de ella. Esto evita que podamos reducirla, trabando la computación.

2.16. Ejercicio 16

Para poder utilizar la reducción *call-by-name* debemos eliminar la regla E-App/v que nos dice que debemos reducir la expresión usada como parámetro hasta obtener un valor y remplazar E-App2/β por:

$$\frac{}{(\lambda x : \sigma.M)\ N \to M\{x \leftarrow N\}}(\text{E-App}/\beta)$$

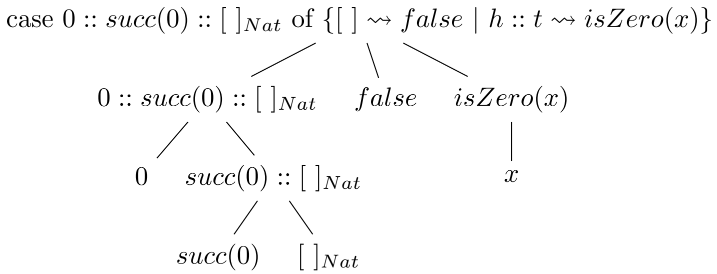
Esta nueva regla, nos permite remplazar x por N, sin importar si N está en forma normal o no. Esto evitará que si la expresión M no usa a x, evaluemos N inncesariamente. La próxima reducción se hizo utilizando los cambios mencionados:

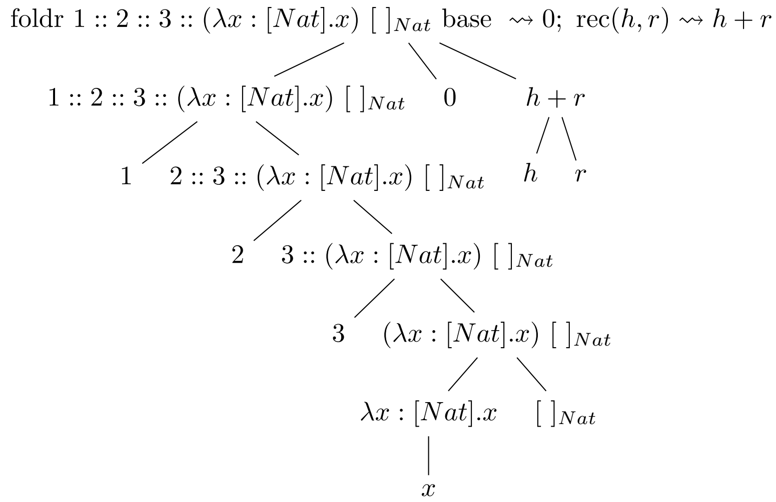
$$(\lambda f : Nat \to Nat.\lambda g : Nat \to Nat.\lambda x : Nat.f\ (g\ x))\ (\lambda x : Nat.succ(x))\ (\lambda x : Nat.succ(x))\ \underline{5}$$
$$\underset{\text{E-App}/\beta}{\to} ((\lambda g : Nat \to Nat.\lambda x : Nat.(\lambda x : Nat.succ(x))\ (g\ x))\ (\lambda x : Nat.succ(x))\ \underline{5} \underset{\text{E-App}/\beta}{\to} (\lambda x : Nat.(\lambda x : Nat.succ(x))\ ((\lambda x : Nat.succ(x))\ x))\ \underline{5}$$
$$\underset{\text{E-App}/\beta}{\to} (\lambda x : Nat.(\lambda x : Nat.succ(x))\ ((\lambda x : Nat.succ(x))\ x))\ \underline{5} \underset{\text{E-App}/\beta}{\to} (\lambda x : Nat.succ(x))\ ((\lambda x : Nat.succ(x))\ \underline{5}) \underset{\text{E-App}/\beta}{\to} succ((\lambda x : Nat.succ(x))\ \underline{5})$$
$$\underset{\text{E-Succ; E-App}/\beta}{\to} succ(succ(\underline{5}))$$

Extensiones

2.17. Ejercicio 17

1)





2)

$$\frac{}{\Gamma \rhd [\,]_\sigma : [\sigma]}\text{(T-Vacio)}$$

$$\frac{\Gamma \rhd M : [\sigma] \qquad \Gamma \rhd N : \tau \qquad \Gamma, h : \sigma, t : [\sigma] \rhd O : \tau}{\Gamma \rhd \text{case } M \text{ of } \{[\,] \leadsto N \mid h :: t \leadsto O\} : \tau}(T-Case)$$

$$\frac{\Gamma \rhd M : \sigma \qquad \Gamma \rhd N : [\sigma]}{\Gamma \rhd M :: N : [\sigma]}\text{(T- ::)}$$

$$\frac{\Gamma \rhd M : [\sigma] \qquad \Gamma \rhd N : \tau \qquad \Gamma, h : \sigma, r : \tau \rhd O : \tau}{\Gamma \rhd \text{foldr } M \text{ base } \leadsto N; \ \text{rec}(h, r) \leadsto O : \tau}(T-Fold)$$

3) $\Gamma = \{x : Bool,\ y : [Bool]\}$

$$\dfrac{\dfrac{\text{Demostración de abajo (1)}}{\Gamma \triangleright x :: x :: y : [Bool]} \quad \dfrac{y : [Bool] \in \Gamma}{\Gamma \triangleright y : [Bool]}\text{T-Var} \quad \dfrac{\text{Demostración abajo (2)}}{\Gamma, h : Bool, r : [Bool] \triangleright \text{if } h \text{ then } r \text{ else } [\,]_{Bool} : [Bool]}}{\Gamma \triangleright \text{foldr } x :: x :: y \text{ base } \rightsquigarrow y;\ \text{rec}(h, r) \rightsquigarrow \text{if } h \text{ then } r \text{ else } [\,]_{Bool} : [Bool]}\text{T-Fold}$$

$$(1) \qquad \dfrac{\dfrac{x : Bool \in \Gamma}{\Gamma \triangleright x : Bool}\text{T-Var} \quad \dfrac{\dfrac{x : Bool \in \Gamma}{\Gamma \triangleright x : Bool}\text{T-Var} \quad \dfrac{y : [Bool] \in \Gamma}{\Gamma \triangleright y : [Bool]}\text{T-Var}}{\Gamma \triangleright x :: y : [Bool]}\text{T-::}}{\Gamma \triangleright x :: x :: y : [Bool]}\text{T-::}$$

$$(2) \qquad \dfrac{\dfrac{h : Bool \in \Gamma, h : Bool, r : [Bool]}{\Gamma, h : Bool, r : [Bool] \triangleright h : Bool}\text{T-Var} \quad \dfrac{r : [Bool] \in \Gamma, h : Bool, r : [Bool]}{\Gamma, h : Bool, r : [Bool] \triangleright r : [Bool]}\text{T-Var} \quad \dfrac{}{\Gamma, h : Bool, r : [Bool] \triangleright [\,]_{Bool} : [Bool]}\text{T-Vacio}}{\Gamma, h : Bool, r : [Bool] \triangleright \text{if } h \text{ then } r \text{ else } [\,]_{Bool} : [Bool]}$$

4) Los nuevos valores son $V\ ::= \ldots \mid [\,]_\sigma \mid V :: V$

5)

$$\dfrac{M_1 \to M_1'}{M_1 :: M_2 \to M_1' :: M_2}(\text{E- :: 1}) \qquad \dfrac{}{\text{case } [\,]_\sigma \text{ of } \{[\,] \rightsquigarrow N \mid h :: t \rightsquigarrow O\} \to N}(\text{E-Case}[\,])$$

$$\dfrac{M_2 \to M_2'}{V :: M_2 \to V :: M_2'}(\text{E- :: 2}) \qquad \dfrac{}{\text{case } V_1 :: V_2 \text{ of } \{[\,] \rightsquigarrow N \mid h :: t \rightsquigarrow O\} \to O\{h \leftarrow V_1,\ t \leftarrow V_2\}}(\text{E-Case ::})$$

$$\frac{M \to M'}{\text{case } M \text{ of } \{[\,] \leadsto N \mid h :: t \leadsto O\} \to \text{case } M' \text{ of } \{[\,] \leadsto N \mid h :: t \leadsto O\}}\text{(E-Case1)}$$

$$\frac{}{\text{foldr } [\,]_\sigma \text{ base } \leadsto N;\ \text{rec}(h, r) \leadsto O \to N}\text{(E-Fold[ ])}$$

$$\frac{}{\text{foldr } V_1 :: V_2 \text{ base } \leadsto N;\ \text{rec}(h, r) \leadsto O \to O\{h \leftarrow V_1,\ r \leftarrow (\text{foldr } V_2 \text{ base } \leadsto N;\ \text{rec}(h, r) \leadsto O)\}}\text{(E-Fold ::)}$$

$$\frac{M \to M'}{\text{foldr } M \text{ base } \leadsto N;\ \text{rec}(h, r) \leadsto O \to \text{foldr } M' \text{ base } \leadsto N;\ \text{rec}(h, r) \leadsto O}\text{(E-Fold1)}$$

2.18. Ejercicio 18

$M \ := \ ... \mid map(N, M)$

$$\frac{\Gamma \triangleright N : [\sigma] \qquad \Gamma \triangleright M : \sigma \to \tau}{\Gamma \triangleright map(N, M) : [\tau]}(T - Map)$$

$$\frac{}{map(V_1,\ V_2 :: V_3) \to (V_1\ V_2) :: map(V_1, V_3)}\text{(E-Map ::)}$$

Los valores del lenguaje no cambian.

$$\frac{M \to M'}{map(V, M) \to map(V, M')}\text{(E-Map1)}$$

$$\frac{}{mapV[\,]_\sigma \to [\,]_\tau}\text{(E-Map[ ])}$$

$$\frac{M_1 \to M_1'}{map(M_1, M_2) \to map(M_1', M_2)}\text{(E-Map2)}$$

2.19. Ejercicio 19

1)

$$\frac{\Gamma, x_1 : \sigma_1, \ldots, x_n : \sigma_n \triangleright M : \tau}{\Gamma \triangleright \mu x_1 : \sigma_1, \ldots, x_n : \sigma_n.M : \tau}(\text{T-}\mu)$$

$$\frac{\Gamma \triangleright M : \{x_1 : \sigma_1\} \to \tau \qquad \Gamma \triangleright N : \sigma_1}{\Gamma \triangleright M \ \#_1 \ N : \tau}(\text{T-}\#_1)$$

$$\frac{\Gamma \triangleright M : \{x_1 : \sigma_1, \ldots, x_n : \sigma_n\} \to \tau \qquad \Gamma \triangleright N : \sigma_i \qquad \text{para } n > 1}{\Gamma \triangleright M \ \#_i \ N : \{x_1 : \sigma_1, \ldots, x_{i-1} : \sigma_{i-1}, x_{i+1} : \sigma_{i+1}, \ldots, x_n : \sigma_n\} \to \tau}(\text{T-}\#_i)$$

2) $V \ ::= \ \ldots \mid \mu x_1 : \sigma_1, \ldots, x_n : \sigma_n.M$

$$\frac{M \to M'}{M \ \#_i \ N \to M' \ \#_i \ N}(\text{E-}\#1)$$

$$\frac{}{(\mu x_1 : \sigma_1.M) \ \#_1 \ V \to M\{x_1 \leftarrow V\}}(\text{E-}\#3)$$

$$\frac{N \to N'}{V \ \#_i \ N \to V \ \#_i \ N'}(\text{E-}\#2)$$

$$\frac{}{(\mu x_1 : \sigma_1, \ldots, x_n : \sigma_n.M) \ \#_i \ V \to \mu x_1 : \sigma_1, \ldots, x_{i-1} : \sigma_{i-1}, x_{i+1} : \sigma_{i+1}, \ldots, x_n : \sigma_n.(M\{x_i \leftarrow V\})}(\text{E-}\#4) \text{ para todo } n > 1$$

3) $\lambda x : \sigma.M \overset{def}{=} \mu x : \sigma.M$ y $M_1 \ M_2 \overset{def}{=} M_1 \ \#_1 \ M_2$

2.20. Ejercicio 20

Acá usamos la extensión con registros vista en la teórica.
$M \ := \ \ldots \mid unionReg(M, M)$
Los valores no cambian.

$$\frac{\Gamma \triangleright M : \{l_i : \sigma_i^{i \in 1..n}\} \quad \Gamma \triangleright N : \{l_i : \sigma_i^{i \in n+1..m}\} \quad l_i = l_j \Rightarrow i = j}{\Gamma \triangleright unionReg(M, N) : \{l_i : \sigma_i^{i \in 1..m}\}} (T - UnionReg)$$

$$\frac{M \to M'}{unionReg(M, N) \to unionReg(M', N)} (E - UnionReg1)$$

$$\frac{}{unionReg(\{l_i : V_i^{i \in 1..n}\}, \{l_i : V_i^{i \in n+1..m}\}) \to \{l_i : V_i^{i \in 1..m}\}} (E - UnionReg3)$$

$$\frac{N \to N'}{unionReg(V, N) \to unionReg(V, N')} (E - UnionReg2)$$

2.21. Ejercicio 21

Not $\stackrel{def}{=} \lambda x : Bool.$if x then $false$ else $true$

Or $\stackrel{def}{=} \lambda x : Bool.\lambda y : Bool.$if x then $true$ else y

And $\stackrel{def}{=} \lambda x : Bool.\lambda y : Bool.$if x then y else $false$

Xor $\stackrel{def}{=} \lambda x : Bool.$**Or** (**And** x (**Not** y)) (**And** (**Not** x) y)

Ejercicio 22

1)

$$\frac{\Gamma \triangleright M_i : \sigma_i \to \tau \quad i \in 1..n \quad \sigma_i = \sigma_j \Rightarrow i = j}{\Gamma \triangleright [(M_1, \ldots, M_n)] : Union(\sigma_1, \ldots, \sigma_n)_\tau} (T - Union)$$

$$\frac{\Gamma \triangleright N : \sigma_i \quad \Gamma \triangleright M : Union(\sigma_1, \ldots, \sigma_n)_\tau \quad \text{para algún } i \in 1..n}{\Gamma \triangleright M \ N : \tau} (T{-}UnionApp)$$

2) $\Gamma = \{y : Nat\}$

$$\dfrac{\dfrac{y : Nat \in \Gamma}{\Gamma \triangleright y : Nat}\text{T-Var} \qquad \dfrac{\dfrac{\text{Demostrado en (1)}}{\Gamma \triangleright \lambda x : Bool.y : Bool \to Nat} \qquad \dfrac{\text{Demostrado en (2)}}{\Gamma \triangleright \lambda x : Nat.x : Nat \to Nat} \qquad Bool \neq Nat}{\Gamma \triangleright [(\lambda x : Bool.y, \lambda x : Nat.x)] : Union(Nat, Bool)_{Nat}}\text{T-Union} \qquad Nat \in \{Bool, Nat\}}{\Gamma \triangleright [(\lambda x : Bool.y, \lambda x : Nat.x)]\ y : Nat}\text{T-UnionApp}$$

(1) $$\dfrac{\dfrac{y : Nat \in \Gamma, x : Bool}{\Gamma, x : Bool \triangleright y : Nat}\text{T-Var}}{\Gamma \triangleright \lambda x : Bool.y : Bool \to Nat}\text{T-Abs}$$

(2) $$\dfrac{\dfrac{x : Nat \in \Gamma, x : Nat}{\Gamma, x : Nat \triangleright x : Nat}\text{T-Var}}{\Gamma \triangleright \lambda x : Nat.x : Nat \to Nat}\text{T-Abs}$$

3) $V \ := \ \dots \mid [(V_1, \dots, V_n)]$. Cada valor, sera una expresión lamda que podremos reducir cuando la apliquemos a otra expresión.

4)

$$\dfrac{M_j \to M_j'}{[(V_i^{i\in 1..j-1}, M_j, M_i^{i\in j+1..n})] \to [(V_i^{i\in 1..j-1}, M_j', M_i^{i\in j+1..n})]}\text{(E-Un1)}$$

$$\dfrac{\forall\ \Gamma \qquad \Gamma \triangleright V_i : \sigma_i \to \tau \qquad \Gamma \triangleright V : \sigma_i \qquad \text{para algún } i \in 1..n}{[(V_i^{i\in 1..n})]\ V \to V_i\ V}\text{(E-AppUnion)}$$

2.22. Ejercicio 23

Usamos la extensión del ejercicio 2.17

1) $head_\sigma \stackrel{def}{=} \lambda xs : [\sigma].\text{case } xs \text{ of } \{[\ ] \rightsquigarrow \bot_\sigma \mid h :: t \rightsquigarrow h\}$

$tail_\sigma \stackrel{def}{=} \lambda xs : [\sigma].\text{case } xs \text{ of } \{[\ ] \rightsquigarrow [\ ]_\sigma \mid h :: t \rightsquigarrow t\}$

2) $iterate_\sigma \stackrel{def}{=} \text{fix } (\lambda g : (\sigma \to \sigma) \to \sigma \to [\sigma].\lambda f : \sigma \to \sigma.\lambda x : \sigma.x :: (g\ f\ (f\ x)))$

3) $isNull_\sigma \stackrel{def}{=} \lambda xs : [\sigma].\text{case } xs \text{ of } \{[\ ] \rightsquigarrow true \mid h :: t \rightsquigarrow false\}$

$zip_{\sigma,\tau} \stackrel{def}{=} \text{fix } (\lambda g : [\sigma] \to [\tau] \to [\sigma \times \tau].\lambda xs : [\sigma].\lambda ys : [\tau].\text{if } \textbf{Or } (isNull_\tau\ ys)\ (isNull_\sigma\ xs) \text{ then } [\ ]_{\sigma\times\tau} \text{ else } (head_\tau\ ys, head_\sigma\ xs) : (g\ (tail_\sigma\ xs)\ (tail_\tau\ ys))$

4) $take_\sigma \stackrel{def}{=} \text{fix } (\lambda f : Nat \to [\sigma].\lambda n : Nat.\lambda xs : [\sigma].\text{if } \textbf{Or } isZero(n)\ (isNull\ xs) \text{ then } [\ ]_\sigma \text{ else } (head\ xs) :: (f\ pred(n)\ (tail\ xs)))$

2.23. Ejercicio 24

1)

$$\frac{\Gamma \triangleright M : \sigma}{\Gamma \triangleright \mathit{detener}(M) : \det(\sigma)}\text{(T-Det)}$$

$$\frac{\Gamma \triangleright M : \det(\sigma) \to \tau \qquad \Gamma \triangleright N : \sigma}{\Gamma \triangleright M\ N : \tau}\text{(T-DetApp)}$$

$$\frac{\Gamma \triangleright M : \det(\sigma)}{\Gamma \triangleright \mathit{continuar}(M) : \sigma}\text{(T-Cont)}$$

2) $\Gamma = \{y : \mathit{Bool}\}$

$$\dfrac{\dfrac{\text{Demostrado en (1)}}{\Gamma \triangleright \lambda x : \det(\mathit{Bool}).\text{if } y \text{ then } \mathit{continuar}(x) \text{ else } \mathit{false} : \det(\mathit{Bool}) \to \mathit{Bool}} \qquad \dfrac{\dfrac{}{\Gamma \triangleright 0 : \mathit{Nat}}\text{T-Zero}}{\Gamma \triangleright \mathit{isZero}(0) : \mathit{Bool}}\text{T-isZeroZero}}{\Gamma \triangleright (\lambda x : \det(\mathit{Bool}).\text{if } y \text{ then } \mathit{continuar}(x) \text{ else } \mathit{false})\ \mathit{isZero}(0) : \mathit{Bool}}\text{T-DetApp}$$

$$(1) \quad \dfrac{\dfrac{\dfrac{y : \mathit{Bool} \in \Gamma, x : \det(\mathit{Bool})}{\Gamma, x : \det(\mathit{Bool}) \triangleright y : \mathit{Bool}}\text{T-Var} \qquad \dfrac{\dfrac{x : \det(\mathit{Bool}) \in \Gamma, x : \det(\mathit{Bool})}{\Gamma, x : \det(\mathit{Bool}) \triangleright x : \det(\mathit{Bool})}\text{T-Var}}{\Gamma, x : \det(\mathit{Bool}) \triangleright \mathit{continuar}(x) : \mathit{Bool}}\text{T-Cont} \qquad \dfrac{}{\Gamma, x : \det(\mathit{Bool}) \triangleright \mathit{false} : \mathit{Bool}}\text{T-False}}{\Gamma, x : \det(\mathit{Bool}) \triangleright \text{if } y \text{ then } \mathit{continuar}(x) \text{ else } \mathit{false} : \mathit{Bool}}\text{T-If}}{\Gamma \triangleright \lambda x : \det(\mathit{Bool}).\text{if } y \text{ then } \mathit{continuar}(x) \text{ else } \mathit{false} : \det(\mathit{Bool}) \to \mathit{Bool}}\text{T-Abs}$$

3) $V \ ::= \ ... \mid detener(M)$

$$\frac{M \to M'}{continuar(M) \to continuar(M')}\text{(E-Cont)}$$

$$\frac{}{continuar(detener(M)) \to M}\text{(E-Cont1)}$$

$$\frac{\Gamma \triangleright N : \sigma}{(\lambda x : \mathrm{det}(\sigma).M)\ N \to (\lambda x : \mathrm{det}(\sigma).M)\ detener(N)}\text{(E-AppDet)}$$

Debemos modificar la regla E-App2 y E-AppAbs para mantener el determinismo:

$$\frac{\Gamma \triangleright V : \sigma \to \tau \qquad \Gamma \triangleright N : \sigma \qquad M_2 \to M_2'}{V\ M_2 \to V\ M_2'}\text{(T-App2)}$$

$$\frac{\Gamma \triangleright V : \sigma}{(\lambda x : \sigma.M)\ V \to M\{x \leftarrow V\}}\text{(T-Abs2)}$$