Notación para este segmento de la materia:

- las letras $M, N, O, P, \ldots$ denotan términos.
- las letras $V, W, \ldots$ denotan valores.
- las letras griegas $\rho, \sigma, \tau, \ldots$ denotan tipos.

Gramáticas a tener en cuenta:

- Términos
  $M ::= x \mid \lambda x\colon \tau.\ M \mid M\ M \mid \mathsf{true} \mid \mathsf{false} \mid \mathsf{if}\ M\ \mathsf{then}\ M\ \mathsf{else}\ M \mid \mathsf{zero} \mid \mathsf{succ}(M) \mid \mathsf{pred}(M) \mid \mathsf{isZero}(M)$

  Donde la letra x representa un *nombre de variable* arbitrario. Dichos nombres se toman de un conjunto infinito numerable dado $\mathfrak{X} = \{w, w_1, w_2, \ldots, x, x_1, x_2, \ldots, y, y_1, y_2, \ldots, z, z_1, z_2, \ldots\}$

- Tipos
  $\tau ::= \mathsf{Bool} \mid \mathsf{Nat} \mid \tau \to \tau$

Sintaxis

Ejercicio 1 ★

Determinar qué expresiones son sintácticamente válidas (es decir, pueden ser generadas con las gramáticas presentadas) y determinar a qué categoría pertenecen (expresiones de términos o expresiones de tipos):

a) x ✓
b) $x\ x$ ✓
c) M ✗
d) $M\ M$ ✗
e) $\mathsf{true\ false}$ ✓
f) $\mathsf{true\ succ(false\ true)}$ ✗
g) $\lambda x.\mathsf{isZero}(x)$ ✓
h) $\lambda x\colon \sigma.\ \mathsf{succ}(x)$
i) $\lambda x\colon \mathsf{Bool}.\ \mathsf{succ}(x)$ ✓
j) $\lambda x\colon \mathsf{if\ true\ then\ Bool\ else\ Nat}.\ x$ ✗
k) σ
l) Bool ✓
m) $\mathsf{Bool} \to \mathsf{Bool}$ ✓
n) $\mathsf{Bool} \to \mathsf{Bool} \to \mathsf{Nat}$ ✓
ñ) $(\mathsf{Bool} \to \mathsf{Bool}) \to \mathsf{Nat}$ ✓
o) $\mathsf{succ\ true}$ ✗
p) $\lambda x\colon \mathsf{Bool}.\ \mathsf{if\ zero\ then\ true\ else\ zero\ succ(true)}$ ✗

a) TÉRMINO (x)

b) TÉRMINO (M M)

c) No es VÁLIDO.

d) No es válido. No debe ser genérico.

e) TÉRMINO

f) NO ES TERM VÁLIDO SUCC(M N) no existe.

g) NO ES TERM VÁLIDO. FALTA TIPO X.

h) TÉRMINO. VALE SII σ:NAT

i) TÉRMINO. NO TIPA.

j) INVÁLIDO. NO EXISTE EL TIPO IF...

k) EXP. TIPO

l) EXP. TIPO

m) EXP. TIPO

n) EXP. TIPO. ASOCIA DER. $\underbrace{\text{Bool}}_{}^{T} \to \overbrace{(\text{Bool} \to \text{Nat})}^{\sigma}$

ñ) EXP TIPO, T: (Bool->Bool) σ: NAT

o) INVÁLIDO. SUCC ESPERA ARG. SUCC TRUE ES APLICACIÓN SUCC A TRUE.

p) TÉRMINO. NO TIPA.

Ejercicio 3 ★

- a) Marcar las ocurrencias del término x como subtérmino en $\lambda x\colon \mathsf{Nat}.\ \mathsf{succ}((\lambda x\colon \mathsf{Nat}.\ x)\ x)$.
- b) ¿Ocurre x_1 como subtérmino en $\lambda x_1\colon \mathsf{Nat}.\ \mathsf{succ}(x_2)$?
- c) ¿Ocurre $x\ (y\ z)$ como subtérmino en $u\ x\ (y\ z)$?

a) $\lambda x\colon NAT.(SUCC(\lambda x'\colon NAT.x')\ X)$

b) Sí. En verde.

c) No. $U\ X\ (Y\ Z) \equiv ((U\ X)(Y\ Z))$ lo que es DIFERENTE A $X\ (Y\ Z)$

Ejercicio 4 ★

Para los siguientes términos:

- a) $u\ x\ (y\ z)\ (\lambda v\colon \mathsf{Bool}.\ v\ y)$
- b) $(\lambda x\colon \mathsf{Bool} \to \mathsf{Nat} \to \mathsf{Bool}.\ \lambda y\colon \mathsf{Bool} \to \mathsf{Nat}.\ \lambda z\colon \mathsf{Bool}.\ x\ z\ (y\ z))\ u\ v\ w$
- c) $w\ (\lambda x\colon \mathsf{Bool} \to \mathsf{Nat} \to \mathsf{Bool}.\ \lambda y\colon \mathsf{Bool} \to \mathsf{Nat}.\ \lambda z\colon \mathsf{Bool}.\ x\ z\ (y\ z))\ u\ v$

Se pide:

I Insertar todos los paréntesis de acuerdo a la convención usual.

II Dibujar el árbol sintáctico de cada una de las expresiones.

III Indicar en el árbol cuáles ocurrencias de variables aparecen ligadas y cuáles libres.

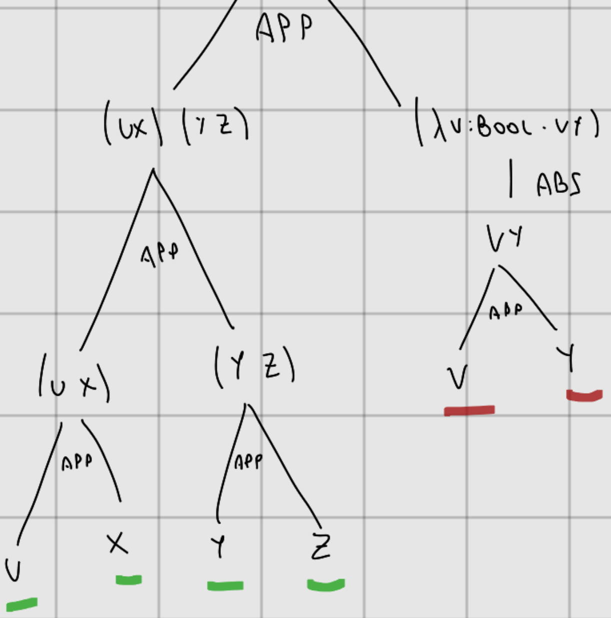
IV ¿En cuál o cuáles de los términos anteriores ocurre la siguiente expresión como subtérmino?
$(\lambda x\colon \mathsf{Bool} \to \mathsf{Nat} \to \mathsf{Bool}.\ \lambda y\colon \mathsf{Bool} \to \mathsf{Nat}.\ \lambda z\colon \mathsf{Bool}.\ x\ z\ (y\ z))\ u$

a) APP > ABS > COND

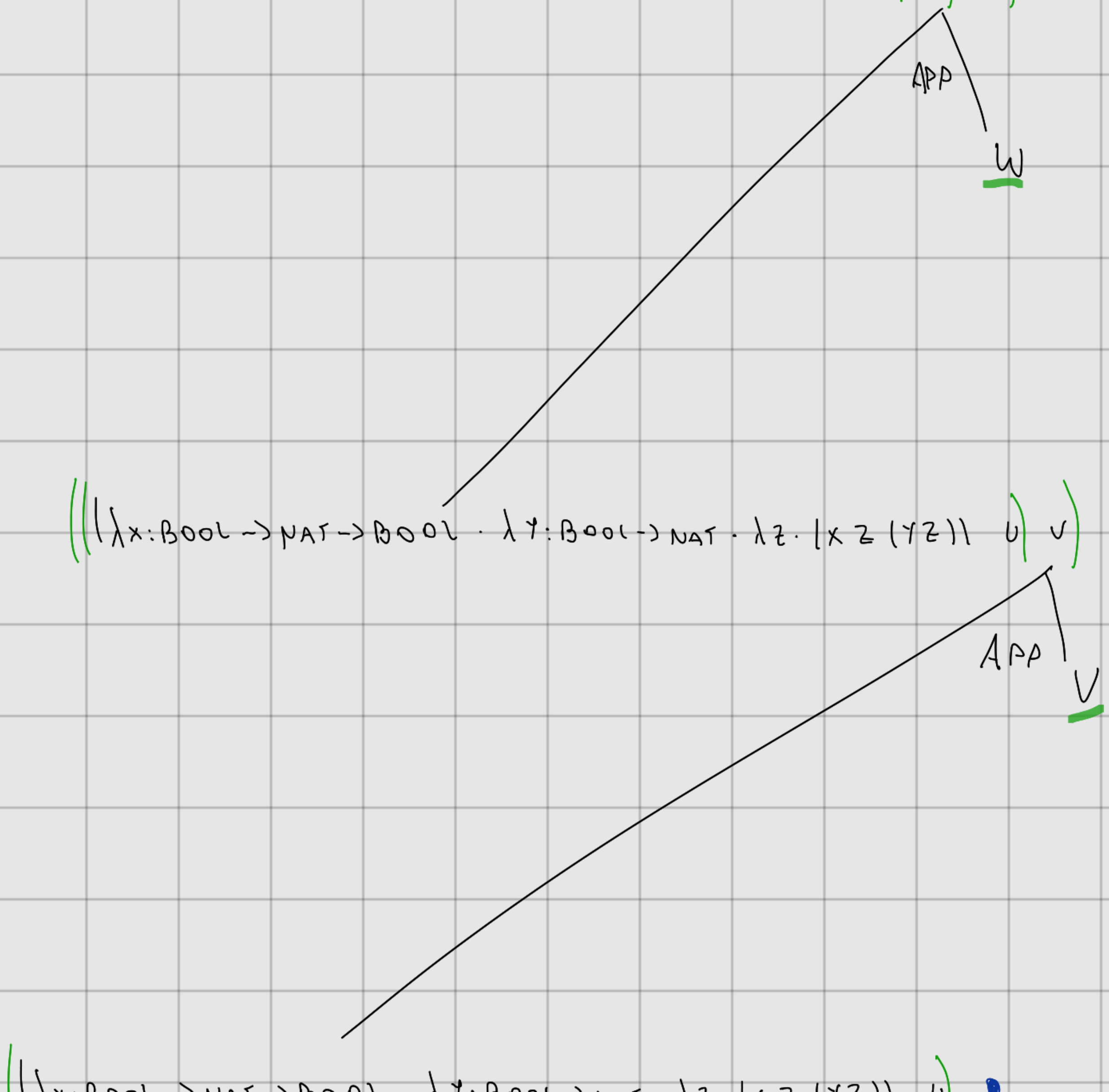
· L

· Li

$((U\ X)\ (Y\ Z))(\lambda V\colon BOOL.\ V\ Y)$



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



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

u

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

ABS

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

ABS

$\lambda z\,(x\ z\ (y\ z))$

ABS

$x\ z\ (y\ z)$

APP

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

APP

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

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

APP

V

$((W\,(\lambda x{:}BOOL \to NAT \to BOOL.\ \lambda y{:}BOOL \to NAT.\ \lambda z{:}BOOL.\ x\,z\,(y\,z)))\ U)$

APP

U

$W\,(\lambda x{:}BOOL \to NAT \to BOOL.\ \lambda y{:}BOOL \to NAT.\ \lambda z{:}BOOL.\ x\,z\,(y\,z))$

APP

W

$\lambda x{:}BOOL \to NAT \to BOOL.\ \lambda y{:}BOOL \to NAT.\ \lambda z{:}BOOL.\ x\,z\,(y\,z)$

ABS

$\lambda y{:}BOOL \to NAT.\ \lambda z{:}BOOL.\ x\,z\,(y\,z)$

ABS

$\lambda z{:}BOOL.\ x\,z\,(y\,z)$

ABS

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

APP

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

iv) En ii).

Tipado

Ejercicio 5

Mostrar un término que no sea tipable y que no tenga variables libres ni abstracciones.

SUCC(TRUE)

Ejercicio 6 *(Derivaciones ★)*

Dar una derivación –o explicar por qué no es posible dar una derivación– para cada uno de los siguientes juicios de tipado:

- a) $\vdash$ if true then zero else succ(zero) : Nat
- b) x : Nat, y : Bool $\vdash$ if true then false else $(\lambda z\colon$ Bool. $z)$ true : Bool
- c) $\vdash$ if $\lambda x\colon$ Bool. x then zero else succ(zero) : Nat
- d) x : Bool $\to$ Nat, y : Bool $\vdash$ $x\ y$: Nat

a)

$$\dfrac{\dfrac{}{\vdash \text{TRUE}:\text{BOOL}}\,\text{T-TRUE} \qquad \dfrac{}{\vdash \text{ZERO}:\text{NAT}}\,\text{T-ZERO} \qquad \dfrac{\dfrac{}{\vdash \text{ZERO}:\text{NAT}}\,\text{T-ZERO}}{\vdash \text{SUCC(ZERO)}:\text{NAT}}\,\text{T-SUCC}}{\vdash \text{ if TRUE THEN ZERO ELSE SUCC(ZERO)}:\text{NAT}}\,\text{T-IF}$$

ESTA VALE.

b) Esta vale.

$$\dfrac{\dfrac{}{X:\text{NAT}, Y:\text{BOOL} \vdash \text{TRUE}:\text{BOOL}}\,\text{T-TRUE} \qquad \dfrac{}{X:\text{NAT}, Y:\text{BOOL} \vdash \text{FALSE}:\text{BOOL}}\,\text{T-FALSE} \qquad \dfrac{\dfrac{\dfrac{}{X:\text{NAT}, Y:\text{BOOL}, Z:\text{BOOL} \vdash Z:\text{BOOL}}\,\text{T-VAR}}{}\,\text{T-ABS}}{X:\text{NAT}, Y:\text{BOOL} \vdash \underbrace{(\lambda Z:\text{BOOL}.Z)}_{\text{BOOL}\to\text{BOOL}}\,\text{TRUE}:\text{BOOL}}}{X:\text{NAT}, Y:\text{BOOL} \vdash \text{ IF TRUE THEN FALSE ELSE } (\lambda Z:\text{BOOL}.Z)\,\text{TRUE}:\text{BOOL}}\,\text{T-IF}$$

c) No tipa. No hay data de Z.

$$\dfrac{\vdash \lambda X:\text{BOOL}.X:\text{BOOL} \qquad \dfrac{}{\vdash Z:\text{NAT}} \qquad \vdash \text{SUCC(ZERO)}:\text{NAT}}{\vdash \text{ IF } \lambda X:\text{BOOL}.X \text{ THEN } Z \text{ ELSE SUCC(ZERO)}:\text{NAT}}\,\text{T-IF}$$

d) No vale. Y tiene 2 tipos dif.

En Γ tengo Y:BOOL.

$$\dfrac{\Gamma \vdash X:\text{BOOL}\to\text{NAT} \qquad \Gamma \vdash Y:\text{NAT}}{\Gamma = X:\text{BOOL}\to\text{NAT}, Y:\text{BOOL} \vdash X\,Y:\text{NAT}}\,\text{T-APP}$$

Ejercicio 7 ★

Se modifica la regla de tipado de la abstracción $(\to_i)$ y se la cambia por la siguiente regla:

$$\dfrac{\Gamma \vdash M\ :\ \tau}{\Gamma \vdash \lambda x\colon \sigma.\ M\ :\ \sigma \to \tau}\ \to_{i2}$$

Exhibir un juicio de tipado que sea derivable en el sistema original pero que no lo sea en el sistema actual.

CUALQUIER ABS QUE EN CUERPO DE M USE LA VARIABLE DE LAMBDA

$$\dfrac{\dfrac{\to_{i2}}{\vdash x : Bool}\ ?}{\vdash \lambda x : Bool . x : \underbrace{Bool}_{\sigma} \to \underbrace{Bool}_{\tau}}$$

$$\dfrac{\dfrac{\to_{i1}}{x : Bool \vdash x : Bool}\ \text{T-VAR}}{\vdash \lambda x : Bool . x : \underbrace{Bool}_{\tau} \to \underbrace{Bool}_{\sigma}}$$

Ejercicio 8

Determinar qué tipo representa σ en cada uno de los siguientes juicios de tipado.

a) $\vdash$ succ(zero) $:\ \sigma$

b) $\vdash$ isZero(succ(zero)) $:\ \sigma$

c) $\vdash$ if (if true then false else false) then zero else succ(zero) $:\ \sigma$

a) NAT   b) BOOL   c) NAT

Ejercicio 9 *(Tipos habitados)* ★

Decimos que un tipo τ está *habitado* si existe un término M tal que el juicio $\vdash M : \tau$ es derivable. En ese caso, decimos que M es un *habitante* de τ. Por ejemplo, dado un tipo σ, la identidad $\lambda x : \sigma . x$ es un habitante del tipo $\sigma \to \sigma$. Demostrar que los siguientes tipos están habitados (para cualquier σ, τ y ρ):

a) $\sigma \to \tau \to \sigma$

b) $(\sigma \to \tau \to \rho) \to (\sigma \to \tau) \to \sigma \to \rho$

c) $(\sigma \to \tau \to \rho) \to (\tau \to (\sigma \to \rho))$ FLIP

d) $(\tau \to \rho) \to (\sigma \to \tau) \to (\sigma \to \rho)$ COMPOSICIÓN

Para pensar: el tipo b es el de la función conocida como *Combinador S*. ¿Con qué función ya conocida de Haskell se corresponden los habitantes de los otros tipos? ¿Hay tipos que no estén habitados? ¿Si se reemplaza $\to$ por $\Rightarrow$, las fórmulas habitadas son siempre tautologías? ¿Las tautologías son siempre fórmulas habitadas?

a)

$$\dfrac{\dfrac{\dfrac{}{x : \sigma, y : \tau \vdash x : \sigma}\ \text{T-VAR}}{x : \sigma \vdash \lambda y : \tau . x : \tau \to \sigma}\ \text{T-ABS}}{\vdash \lambda x : \sigma . \lambda y : \tau . x : \sigma \to (\tau \to \sigma)}\ \text{T-ABS}$$

b)

$$\dfrac{\dfrac{\dfrac{}{\Gamma' \vdash f : \sigma \to (\tau \to \rho)}\ \text{T-VAR} \quad \dfrac{}{\Gamma' \vdash x : \sigma}\ \text{T-VAR}}{\Gamma' \vdash f x : \tau \to \rho}\ \text{T-APP} \qquad \dfrac{\dfrac{}{\Gamma' \vdash g : \sigma \to \tau}\ \text{T-VAR} \quad \dfrac{}{\Gamma' \vdash x : \sigma}\ \text{T-VAR}}{\Gamma' \vdash (g x) : \tau}\ \text{T-APP}}{\Gamma' = f : \sigma \to \tau \to \rho,\ g : \sigma \to \tau,\ x : \sigma \vdash f x (g x) : \rho}\ \text{T-APP}$$

$f : (\sigma \to \tau \to \rho)$ APP PARCIAL $\rightsquigarrow$ $f x : (\tau \to \rho)$

$$\dfrac{\dfrac{\dfrac{\Gamma' = f : \sigma \to \tau \to \rho,\ g : \sigma \to \tau,\ x : \sigma \vdash f x (g x) : \rho}{f : \sigma \to \tau \to \rho,\ g : \sigma \to \tau \vdash \lambda x : \sigma . (f x (g x)) : \sigma \to \rho}\ \text{T-ABS}}{f : \sigma \to \tau \to \rho \vdash \lambda g : \sigma \to \tau . \lambda x : \sigma . (f x (g x)) : (\sigma \to \tau) \to (\sigma \to \rho)}\ \text{T-ABS}}{\vdash \lambda f : \sigma \to (\tau \to \rho) . \lambda g : \sigma \to \tau . \lambda x : \sigma . (f x (g x)) : (\sigma \to (\tau \to \rho)) \to (\sigma \to \tau) \to \sigma \to \rho}\ \text{T-ABS}$$

c) FLIP

$$\dfrac{\dfrac{\dfrac{}{\Gamma' \vdash f : (\sigma \to (\tau \to \rho))}\ \text{T-VAR} \quad \dfrac{}{\Gamma' \vdash y : \sigma}\ \text{T-VAR}}{\Gamma' \vdash f y : (\tau \to \rho)}\ \text{T-APP} \qquad \dfrac{}{\Gamma' \vdash x : \tau}\ \text{T-VAR}}{\Gamma' = f : (\sigma \to \tau \to \rho),\ x : \tau,\ y : \sigma \vdash f y x : \rho}\ \text{T-APP}$$

$$\dfrac{\dfrac{\Gamma' = f : (\sigma \to \tau \to \rho),\ x : \tau,\ y : \sigma \vdash f y x : \rho}{f : (\sigma \to \tau \to \rho),\ x : \tau \vdash \lambda y : \sigma . (f y x) : \sigma \to \rho}\ \text{T-ABS}}{f : (\sigma \to \tau \to \rho) \vdash \lambda x : \tau . \lambda y : \sigma . (f y x) : (\tau \to (\sigma \to \rho))}\ \text{T-ABS}$$

T-ABS

$\vdash \lambda f:(\sigma\to\tau\to\rho).\lambda x:\tau.\lambda y:\sigma.(f\ y\ x) : (\sigma\to\tau\to\rho)\to\tau\to\sigma\to(\rho)$

d) Composición

$$\dfrac{\dfrac{}{\Gamma' \vdash f:\tau\to\rho}\text{T-VAR} \qquad \dfrac{\dfrac{}{\Gamma'\vdash g:\sigma\to\tau}\text{T-VAR} \quad \dfrac{}{\Gamma'\vdash x:\sigma}\text{T-VAR}}{\Gamma'\vdash g\ x:\tau}\text{T-APP}}{\Gamma' = f:\tau\to\rho,\ g:\sigma\to\tau,\ x:\sigma \ \vdash f\$g(x) : \rho}\text{T-APP}$$

$$\dfrac{\Gamma' = f:\tau\to\rho,\ g:\sigma\to\tau,\ x:\sigma \ \vdash f\$g(x) : \rho}{f:\tau\to\rho,\ g:\sigma\to\tau \vdash \lambda x:\sigma.f(g(x)) : \sigma\to\rho}\text{T-ABS}$$

$$\dfrac{f:\tau\to\rho,\ g:\sigma\to\tau \vdash \lambda x:\sigma.f(g(x)) : \sigma\to\rho}{f:(\tau\to\rho) \vdash \lambda g:(\sigma\to\tau).\lambda x:\sigma.f(g(x)) : (\sigma\to\tau)\to(\sigma\to\rho)}\text{T-ABS}$$

$$\dfrac{f:(\tau\to\rho) \vdash \lambda g:(\sigma\to\tau).\lambda x:\sigma.f(g(x)) : (\sigma\to\tau)\to(\sigma\to\rho)}{\vdash \lambda f:(\tau\to\rho).\lambda g:(\sigma\to\tau).\lambda x:\sigma.(f(g\ x)) : (\tau\to\rho)\to((\sigma\to\tau)\to(\sigma\to\rho))}\text{T-ABS}$$

Ejercicio 10 ★

Determinar qué tipos representan σ y τ en cada uno de los siguientes juicios de tipado. Si hay más de una solución, o si no hay ninguna, indicarlo.

a) $x:\sigma \vdash \mathsf{isZero}(\mathsf{succ}(x)) \ : \ \tau$

b) $\vdash (\lambda x:\sigma.\ x)(\lambda y:\mathsf{Bool}.\ \mathsf{zero}) \ : \ \sigma$

c) $y:\tau \vdash \mathsf{if}\ (\lambda x:\sigma.\ x)\ \mathsf{then}\ y\ \mathsf{else}\ \mathsf{succ}(\mathsf{zero}) \ : \ \sigma$

d) $x:\sigma \vdash x\ y \ : \ \tau$

e) $x:\sigma, y:\tau \vdash x\ y \ : \ \tau$

f) $x:\sigma \vdash x\ \mathsf{true} \ : \ \tau$

g) $x:\sigma \vdash x\ \mathsf{true} \ : \ \sigma$ $\rightsquigarrow \sigma:(\mathsf{Bool}\to\sigma)$ OCCURS-CHECK

h) $x:\sigma \vdash x\ x \ : \ \tau$

a) $\sigma:\mathsf{NAT}$, $\tau:\mathsf{BOOL}$

b) $\sigma:\mathsf{NAT}$

c) GUARDA $\sigma:\mathsf{BOOL}$ No hay solución.

THEN: $y:\tau$
ELSE: $\mathsf{SUCC}(\mathsf{ZERO})$ $\to \tau:\mathsf{NAT}$

PERO SALIDA ES σ (BOOL)

d) $x:\sigma \to$ FUNCIÓN Q TOMA ALGO Y DEVUELVE ALGO.

$x:\mathsf{BOOL}\to\mathsf{BOOL} \vdash x\ y : \mathsf{BOOL}$

NO OBSTANTE, y NO TIENE TIPO, ERROR. No hay solución.

e) $\sigma = \tau\to\tau$. Hay más de una solución

Ej: $x:\mathsf{Bool}\to\mathsf{Bool}, y:\mathsf{Bool} \vdash x\ y : \mathsf{Bool}$

f) $\sigma = \mathsf{BOOL}\to\tau$. Hay más de una solución

$x:\sigma \vdash x\ \mathsf{TRUE} : \tau$

Ej: $x:(\mathsf{BOOL}\to\mathsf{NAT}) \vdash x\ \mathsf{TRUE} : \mathsf{NAT}$

g) FALLA

$\sigma:(\mathsf{BOOL}\to\sigma)$

Ejercicio 12 *(Lema de sustitución ★)*

Demostrar que si valen $\Gamma, x:\sigma \vdash M \ : \ \tau$ y $\Gamma \vdash N \ : \ \sigma$ entonces vale $\Gamma \vdash M\{x := N\} \ : \ \tau$.
Sugerencia: proceder por inducción en la estructura del término M.

?

Ejercicio 13 ★

Sean σ, τ, ρ tipos. Según la definición de sustitución, calcular:

1. Renombramos las ligadas si hay alguna libre que se

a) $(\lambda y\colon \sigma.\ x\ (\lambda x\colon \tau.\ x))\{x := (\lambda y\colon \rho.\ x\ y)\}$

b) $(y\ (\lambda v\colon \sigma.\ x\ v))\{x := (\lambda y\colon \tau.\ v\ y)\}$

llame igual.

POR α-EQUIVALENCIA RENOMBRO LAS LIGADAS.

2. Sustituyo los libres por la variable que lleva.

a) $(\lambda y:\sigma . x (\lambda x : \tau . x)) \{x := (\lambda y : \rho . (x\ y))\}$

$(\lambda z : \sigma ((\lambda y : \rho . (\underline{x}\ y)) (\lambda x : \tau . x)))$ ✓

queda libre.

RENOMBRO

b) $(y (\lambda v : \sigma . x\ v)) \{x := (\lambda y : \tau . v\ y)\}$

$(y (\lambda v : \sigma . ((\lambda z : \tau . v\ z)\ v)))$ ✓

Ejercicio 15 *(Valores)* ★

Dado el conjunto de valores visto en clase: ⇝ Dice qué términos si es VALOR o no por sintaxis

$$V := \lambda x\colon \tau.\ M \mid \mathsf{true} \mid \mathsf{false} \mid \mathsf{zero} \mid \mathsf{succ}(V)$$

(xq el CONCEPTO de VALOR sólo si es PROGRAMA)

Determinar si cada una de las siguientes expresiones es o no un valor:

- a) $(\lambda x\colon \mathsf{Bool}.\ x)\ \mathsf{true}$
- b) $\lambda x\colon \mathsf{Bool}.\ \underline{2}$
- c) $\lambda x\colon \mathsf{Bool}.\ \mathsf{pred}(\underline{2})$
- d) $\lambda y\colon \mathsf{Nat}. ((\lambda x\colon \mathsf{Bool}.\ \mathsf{pred}(\underline{2}))\ \mathsf{true})$
- e) x
- f) $\mathsf{succ}(\mathsf{succ}(\mathsf{zero}))$

a) $(\lambda x : \text{BOOL} . x)$ TRUE NO. ES TÉRMINO

e) no.

b) SÍ

f) SÍ $= \underline{2}$

c) SÍ, es VALOR. No me importa q CUERPO LAMBDA no tenga valores.

d) NO. ES UNA APLICACIÓN

Si tengo $(\lambda x : T . \vec{M})$ PERO ESTO ES LAMBDA O TÉRMINO (NO VALOR) ES CONSIDERADO COMO VALOR? sí.

Ejercicio 16 *(Programa, Forma Normal)* ★

Para el siguiente ejercicio, considerar el cálculo **sin** la regla $\mathsf{pred}(\mathsf{zero}) \to \mathsf{zero}$

Un *programa* es un término que tipa en el contexto vacío (es decir, no puede contener variables libres).

Para cada una de las siguientes expresiones,

a) Determinar si puede ser considerada un **programa**.

b) Si es un programa, ¿Cuál es el resultado de su evaluación? Determinar si se trata de una forma normal, y en caso de serlo, si es un **valor** o un **error**.

(REDUZCO TODO AL MÁXIMO (SI SE PUEDE) Y DSP ME FIJO VALOR?

- I $(\lambda x\colon \mathsf{Bool}.\ x)\ \mathsf{true}$
- II $\lambda x\colon \mathsf{Nat}.\ \mathsf{pred}(\mathsf{succ}(x))$
- III $\lambda x\colon \mathsf{Nat}.\ \mathsf{pred}(\mathsf{succ}(y))$
- IV $(\lambda x\colon \mathsf{Bool}.\ \mathsf{pred}(\mathsf{isZero}(x)))\ \mathsf{true}$
- V $(\lambda f\colon \mathsf{Nat} \to \mathsf{Bool}.\ f\ \mathsf{zero})\ (\lambda x\colon \mathsf{Nat}.\ \mathsf{isZero}(x))$
- VI $(\lambda f\colon \mathsf{Nat} \to \mathsf{Bool}.\ x)\ (\lambda x\colon \mathsf{Nat}.\ \mathsf{isZero}(x))$
- VII $(\lambda f\colon \mathsf{Nat} \to \mathsf{Bool}.\ f\ \mathsf{pred}(\mathsf{zero}))\ (\lambda x\colon \mathsf{Nat}.\ \mathsf{isZero}(x))$
- VIII $\mu y\colon \mathsf{Nat}.\ \mathsf{succ}(y)$

μ ⇝ $(\lambda y : \text{NAT}) (\text{SUCC}(y))$ ES VALOR

i) TIPA. F.N: TRUE

ii) $(\lambda x : \text{NAT} . (\text{PRED}(\text{SUCC}(x))))$ x como mucho es ZERO. $\text{SUCC}(\text{ZERO}) \rightsquigarrow \text{PRED}(\overbrace{\text{SUCC}(\text{ZERO})}^{1}) \rightsquigarrow \text{PRED}(1) \rightsquigarrow \text{ZERO}$

TIPA. F.N: SÍ. AL REDUCIR TERMINA SIENDO VALOR.

iii) No es programa. Y está libre

iv) Error. No tipa

v) Tipa. F.N: Sí. True

vi) No es programa. Y libre.

Ejercicio 17 *(Determinismo)*

Sí, por def.

a) ¿Es cierto que la relación definida $\to$ está determinada (es una función parcial)?
Más precisamente, ¿pasa que si $M \to N$ y $M \to N'$ entonces también vale $N = N'$?

b) ¿Vale lo mismo con muchos pasos? Es decir, ¿es cierto que si $M \twoheadrightarrow M'$ y $M \twoheadrightarrow M''$ entonces $M' = M''$?

c) ¿Acaso es cierto que si $M \to M'$ y $M \twoheadrightarrow M''$ entonces $M' = M''$?

a) No es un "también". Por el teorema de determinismo si $M \to N$ y $M \to N'$ entonces $N = N'$. Esto es porque solo podemos aplicar una y única regla de reducción semántica en cada paso.

(lo que vemos en materia)

b) Sí, también. Llegamos al mismo resultado.

c) Sí

Ejercicio 18

a) ¿Da lo mismo evaluar $\mathsf{succ}(\mathsf{pred}(M))$ que $\mathsf{pred}(\mathsf{succ}(M))$? ¿Por qué?

b) ¿Es verdad que para todo término M vale $\mathsf{isZero}(\mathsf{succ}(M)) \twoheadrightarrow \mathsf{false}$? Si no lo es, ¿para qué términos vale?

c) ¿Para qué términos M vale $\mathsf{isZero}(\mathsf{pred}(M)) \twoheadrightarrow \mathsf{true}$? (Hay infinitos). ✓

a) No. succ(pred(zero)) explota. pred(M) con M = zero no está definido.

pred(succ(zero)) siempre vale. succ(zero) > zero ⇒ pred está definido

b) Sí. Olvidé que pred(zero) no vale. succ(M) > zero

c) Cualquier término que haga llegar a 0 para que isZero(0)

Extensiones

En esta sección puede asumirse, siempre que sea necesario, que el cálculo ha sido extendido con la suma de números naturales $(M + N)$, con las siguiente reglas de tipado y semántica:

$$\frac{\Gamma \vdash M : \mathsf{Nat} \quad \Gamma \vdash N : \mathsf{Nat}}{\Gamma \vdash M + N : \mathsf{Nat}} +$$

Si $M \to M'$, **entonces:** $M + N \to M' + N$	$(+_{c1})$
Si $N \to N'$, **entonces:** $V + N \to V + N'$	$(+_{c2})$
$V + \mathsf{zero} \to V$	$(+_0)$
$V_1 + \mathsf{succ}(V_2) \to \mathsf{succ}(V_1) + V_2$	$(+_{\mathsf{succ}})$

Ejercicio 20 *(Pares, o productos)* ★

Este ejercicio extiende el cálculo-λ tipado con *pares*. Las gramáticas de los tipos y los términos se extienden de la siguiente manera:

$$\begin{array}{lcl} \tau & ::= & \ldots \mid \tau \times \tau \\ M & ::= & \ldots \mid \langle M, M\rangle \mid \pi_1(M) \mid \pi_2(M) \end{array}$$

donde $\sigma \times \tau$ es el tipo de los pares cuya primera componente es de tipo σ y cuya segunda componente es de tipo τ, $\langle M, N\rangle$ construye un par y $\pi_1(M)$ y $\pi_2(M)$ proyectan la primera y la segunda componente de un par, respectivamente.

a) Definir reglas de tipado para los nuevos constructores de términos.

b) Usando las reglas de tipado anteriores, y dados los tipos σ, τ y ρ, exhibir habitantes de los siguientes tipos:

- I) Constructor de pares: $\sigma \to \tau \to (\sigma \times \tau)$
- II) Proyecciones: $(\sigma \times \tau) \to \sigma$ y $(\sigma \times \tau) \to \tau$
- III) Conmutatividad: $(\sigma \times \tau) \to (\tau \times \sigma)$,
- IV) Asociatividad: $((\sigma \times \tau) \times \rho) \to (\sigma \times (\tau \times \rho))$ y $(\sigma \times (\tau \times \rho)) \to ((\sigma \times \tau) \times \rho)$.
- V) Currificación: $((\sigma \times \tau) \to \rho) \to (\sigma \to \tau \to \rho)$ y $(\sigma \to \tau \to \rho) \to ((\sigma \times \tau) \to \rho)$.

↳ Funciones a tipar

- c) ¿Cómo se extiende el conjunto de los valores?
- d) Definir reglas de semántica operacional manteniendo el determinismo y la preservación de tipos. **Importante:** no olvidar las reglas de congruencia.
- e) Demostrar el determinismo de la relación de reducción definida. ¿Se verifica la propiedad de preservación de tipos? ¿Se verifica la propiedad de progreso?

a) $M ::= \ldots \mid \langle M, N \rangle \mid \pi_1(M) \mid \pi_2(M)$ Tipo nuevo: $\tau \times \tau$

$$\frac{\Gamma \vdash M : \sigma \qquad \Gamma \vdash N : \tau}{\Gamma \vdash \langle M, N \rangle : \sigma \times \tau}\ \text{T-PAR} \qquad \frac{\Gamma \vdash M : \sigma \times \tau}{\Gamma \vdash \pi_1(M) : \sigma}\ \text{T-}\pi_1$$

$$\frac{\Gamma \vdash M : \sigma \times \tau}{\Gamma \vdash \pi_2(M) : \tau}\ \text{T-}\pi_2$$

b)

i)

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma'' \vdash x : \sigma}\ \text{T-VAR} \qquad \dfrac{}{\Gamma'' \vdash y : \tau}\ \text{T-VAR}}{\Gamma'' = \Gamma, x:\sigma, y:\tau \vdash \langle x, y \rangle : \sigma \times \tau}\ \text{T-PAR}}{\Gamma, x:\sigma \vdash \lambda y:\tau . \langle x, y \rangle : \tau \to (\sigma \times \tau)}\ \text{T-ABS}}{\vdash \lambda x:\sigma . \lambda y:\tau . \langle x, y \rangle : \sigma \to \tau \to (\sigma \times \tau)}\ \text{T-ABS}$$

ii)

$$\dfrac{\dfrac{\dfrac{}{p:(\sigma \times \tau) \vdash p:(\sigma \times \tau)}\ \text{T-VAR}}{p:(\sigma \times \tau) \vdash \pi_1(p) : \sigma}\ \text{T-}\pi_1}{\vdash \lambda p:(\sigma \times \tau) . \pi_1(p) : (\sigma \times \tau) \to \sigma}\ \text{T-ABS}$$

El otro es análogo pero con $\pi_2(p)$

iii)

$$\dfrac{\dfrac{}{p:(\sigma \times \tau) \vdash p : \sigma \times \tau}\ \text{T-VAR}}{\ldots}\ \text{T-}\pi_1 \qquad \dfrac{\dfrac{}{p:(\sigma \times \tau) \vdash p : \sigma \times \tau}\ \text{T-VAR}}{\ldots}\ \text{T-}\pi_1$$

$$\frac{P:(\sigma\times\tau)\vdash\pi_2(P):\tau \qquad P:(\sigma\times\tau)\vdash\pi_1(P):\sigma}{P:(\sigma\times\tau)\vdash\langle\pi_2(P),\pi_1(P)\rangle:(\tau\times\sigma)}\ \text{T-PAR}$$

$$\frac{P:(\sigma\times\tau)\vdash\langle\pi_2(P),\pi_1(P)\rangle:(\tau\times\sigma)}{\vdash\lambda P:(\sigma\times\tau).\langle\pi_2(P),\pi_1(P)\rangle:(\sigma\times\tau)\to(\tau\times\sigma)}\ \text{T-ABS}$$

iv)

$$\frac{}{\Gamma'\vdash P:(\sigma\times\tau)\times P}\ \text{T-VAR} \qquad \frac{\Gamma'\vdash P:(\sigma\times\tau)\times P}{\Gamma'\vdash\pi_1(P):\sigma\times\tau}\ \text{T-C1} \qquad \frac{\Gamma'\vdash\pi_1(P):\sigma\times\tau}{\Gamma'\vdash\pi_1(\pi_1(A)):\sigma}$$

$$\frac{}{\Gamma'\vdash A:(\sigma\times\tau)\times P}\ \text{T-VAR} \qquad \frac{\Gamma'\vdash A:(\sigma\times\tau)\times P}{\Gamma'\vdash\pi_1(A):\sigma\times\tau}\ \text{T-C1} \qquad \frac{\Gamma'\vdash\pi_1(A):\sigma\times\tau}{\Gamma'\vdash\pi_2(\pi_1(A)):\tau}\ \text{T-C}_2$$

$$\frac{}{\Gamma'\vdash A:(\sigma\times\tau)\times P}\ \text{T-VAR} \qquad \frac{\Gamma'\vdash A:(\sigma\times\tau)\times P}{\Gamma'\vdash\pi_2(A):P}\ \text{T-C}_2$$

$$\frac{\Gamma'\vdash\pi_2(\pi_1(A)):\tau \qquad \Gamma'\vdash\pi_2(A):P}{\Gamma'\vdash\langle\pi_2(\pi_1(A)),\pi_2(A)\rangle:(\tau\times P)}\ \text{T-PAR}$$

$$\frac{\Gamma'\vdash\pi_1(\pi_1(A)):\sigma \qquad \Gamma'\vdash\langle\pi_2(\pi_1(A)),\pi_2(A)\rangle:(\tau\times P)}{\Gamma'=A:((\sigma\times\tau)\times P)\vdash\langle\pi_1(\pi_1(A)),\langle\pi_2(\pi_1(A)),\pi_2(A)\rangle\rangle:(\sigma\times(\tau\times P))}\ \text{T-PAR}$$

$$\frac{A:((\sigma\times\tau)\times P)\vdash\langle\pi_1(\pi_1(A)),\langle\pi_2(\pi_1(A)),\pi_2(A)\rangle\rangle:(\sigma\times(\tau\times P))}{\lambda A:((\sigma\times\tau)\times P).\langle\pi_1(\pi_1(A)),\langle\pi_2(\pi_1(A)),\pi_2(A)\rangle\rangle:((\sigma\times\tau)\times P)\to(\sigma\times(\tau\times P))}\ \text{T-ABS}$$

El otro es análogo

v)

$$\frac{}{\Gamma'\vdash f:((\sigma\times\tau)\to P)}\ \text{T-VAR} \qquad \frac{}{\Gamma'\vdash a:\sigma}\ \text{T-VAR} \qquad \frac{}{\Gamma'\vdash b:\tau}\ \text{T-VAR}$$

$$\frac{\Gamma'\vdash a:\sigma \qquad \Gamma'\vdash b:\tau}{\Gamma'\vdash\langle a,b\rangle:(\sigma\times\tau)}\ \text{T-PAR}$$

$$\frac{\Gamma'\vdash f:((\sigma\times\tau)\to P) \qquad \Gamma'\vdash\langle a,b\rangle:(\sigma\times\tau)}{\Gamma'=f:((\sigma\times\tau)\to P),a:\sigma,b:\tau\vdash f\langle a,b\rangle:P}\ \text{T-APP}$$

$$\frac{\Gamma'=f:((\sigma\times\tau)\to P),a:\sigma,b:\tau\vdash f\langle a,b\rangle:P}{f:((\sigma\times\tau)\to P),a:\sigma\vdash\lambda b:\tau.f\langle a,b\rangle:\tau\to P}\ \text{T-ABS}$$

$$\frac{f:((\sigma\times\tau)\to P),a:\sigma\vdash\lambda b:\tau.f\langle a,b\rangle:\tau\to P}{f:((\sigma\times\tau)\to P)\vdash\lambda a:\sigma.\lambda b:\tau.f\langle a,b\rangle:(\sigma\to(\tau\to P))}\ \text{T-ABS}$$

$$\frac{f:((\sigma\times\tau)\to P)\vdash\lambda a:\sigma.\lambda b:\tau.f\langle a,b\rangle:(\sigma\to(\tau\to P))}{\vdash\lambda f:((\sigma\times\tau)\to P).\lambda a:\sigma.\lambda b:\tau.f\langle a,b\rangle:((\sigma\times\tau)\to P)\to(\sigma\to\tau\to P)}\ \text{T-ABS}$$

$$\frac{}{\Gamma'\vdash f:\sigma\to\tau\to P}\ \text{T-VAR} \qquad \frac{}{\Gamma'\vdash P:\sigma\times\tau}\ \text{T-VAR} \qquad \frac{\Gamma'\vdash P:\sigma\times\tau}{\Gamma'\vdash\pi_1(P):\sigma}\ \text{T-C1}$$

$$\frac{\Gamma'\vdash f:\sigma\to\tau\to P \qquad \Gamma'\vdash\pi_1(P):\sigma}{\Gamma'\vdash f(\pi_1(P)):\tau\to P}\ \text{T-APP}$$

$$\frac{}{\Gamma'\vdash P:\sigma\times\tau}\ \text{T-VAR} \qquad \frac{\Gamma'\vdash P:\sigma\times\tau}{\Gamma'\vdash\pi_2(P):\tau}\ \text{T-C}_2$$

$$\frac{\Gamma'\vdash f(\pi_1(P)):\tau\to P \qquad \Gamma'\vdash\pi_2(P):\tau}{\Gamma'=f:(\sigma\to\tau\to P),P:\sigma\times\tau\vdash f(\pi_1(P))(\pi_2(P)):P}\ \text{T-APP}$$

$$\dfrac{f:(G\to T\to P) \vdash \lambda p:G\times T\,.\,f\,((\pi_1(p))(\pi_2(p)) : ((G\times T)\to P)}{\vdash \lambda f:(G\to T\to P)\,.\,\lambda p:(G\times T)\,.\,f\,(\pi_1(p))(\pi_2(p)) : (G\to T\to P)\to((G\times T)\to P)}\ \text{T-ABS}$$

$\pi_1(v)$ y $\pi_2(v)$ no tq el v justamente es del TIPO $\langle v,v\rangle$.

c) $V ::= \ldots \mid \langle V,V\rangle$ -> PARA CÓMPUTO, VEO TÉRMINOS Y DEFINO CASOS PARA $\langle V,V\rangle$

CONGRUENCIA

$$\dfrac{M\to M'}{\langle M,N\rangle \to \langle M',N\rangle}\ \text{C-1} \qquad \dfrac{N\to N'}{\langle V,N\rangle \to \langle V,N'\rangle}\ \text{C-2}$$

$$\dfrac{M\to M'}{\pi_1(M)\to\pi_1(M')}\ \text{C-}\pi_1 \qquad \dfrac{M\to M'}{\pi_2(M)\to\pi_2(M')}\ \text{C-}\pi_2$$

CÓMPUTO

$$\dfrac{}{\pi_1(\langle V_1,V_2\rangle)\to V_1}\ \text{CO-}\pi_1 \qquad \dfrac{}{\pi_2(\langle V_1,V_2\rangle)\to V_2}\ \text{CO-}\pi_2$$

e) DETERMINISMO: si $\Gamma\vdash M:P$, $M\to N_1$ y $M\to N_2$ entonces $N_1=N_2$ y esto sucede porque sólo $\exists$ una regla posible para ir reduciendo los términos.

PRESERVACIÓN DE TIPOS: si $\Gamma\vdash M:P$ y $M\to N$ entonces $\Gamma\vdash N:P$.

PROGRESO: si M está en forma NORMAL no reduce, sino, si $M\to N$ entonces reduce.

Esto sí sucede pues nuestras reglas o su forma finalizan computando un valor VÁLIDO (en F.N).

Ejercicio 21 *(Uniones disjuntas, también conocidas como co-productos o sumas)*

Este ejercicio extiende el cálculo-λ tipado con *uniones disjuntas*. Las gramáticas de los tipos y los términos se extienden de la siguiente manera:

$\tau ::= \ldots \mid \tau+\tau$

$M \quad ::= \quad \ldots \mid \mathsf{left}_\tau(M) \mid \mathsf{right}_\tau(M) \mid \mathsf{case}\ M\ \mathsf{of}\ \mathsf{left}(x) \rightsquigarrow M \parallel \mathsf{right}(y) \rightsquigarrow M$

donde $\sigma + \tau$ representa el tipo de la unión disjunta entre σ y τ, similar al tipo Either σ τ de Haskell, $\mathsf{left}_\sigma(M)$ y $\mathsf{right}_\tau(M)$ inyectan un valor en la unión y $\mathsf{case}\ M\ \mathsf{of}\ \mathsf{left}(x) \rightsquigarrow M_1 \parallel \mathsf{right}(y) \rightsquigarrow M_2$ efectúa un análisis de casos del término M comparándolo con los patrones $\mathsf{left}_\sigma(x)$ y $\mathsf{right}_\tau(y)$.

a) Definir reglas de tipado para los nuevos constructores de términos.

b) Usando las reglas de tipado anteriores, y dados los tipos σ, τ y ρ, exhibir habitantes de los siguientes tipos:

- I) Inyecciones: $\sigma \to (\sigma + \tau)$ y $\tau \to (\sigma + \tau)$.
- II) Análisis de casos: $(\sigma + \tau) \to (\sigma \to \rho) \to (\tau \to \rho) \to \rho$.
- III) Conmutatividad: $(\sigma + \tau) \to (\tau + \sigma)$.
- IV) Asociatividad: $((\sigma + \tau) + \rho) \to (\sigma + (\tau + \rho))$ y $(\sigma + (\tau + \rho)) \to ((\sigma + \tau) + \rho)$.
- V) Distributividad del producto sobre la suma: $(\sigma \times (\tau + \rho)) \to ((\sigma \times \tau) + (\sigma \times \rho))$ y $((\sigma \times \tau) + (\sigma \times \rho)) \to (\sigma \times (\tau + \rho))$.
- VI) Ley de los exponentes[1]: $((\sigma + \tau) \to \rho) \to ((\sigma \to \rho) \times (\tau \to \rho))$ y $((\sigma \to \rho) \times (\tau \to \rho)) \to ((\sigma + \tau) \to \rho)$.

c) ¿Cómo se extiende el conjunto de los valores?

d) Definir reglas de semántica operacional manteniendo el determinismo y la preservación de tipos. ¿Se verifica la propiedad de progreso?

e) Demostrar que la relación de reducción definida tiene la propiedad de preservación de tipos.

TIPO NUEVO: $\sigma + \tau$

a.) OJO: ANTES HABÍA PUESTO $M: \sigma+\tau$ PERO EN REALIDAD $M:\sigma$ Y EL TIPO ESTE INYECTA Y DEVUELVE INSTANCIA DE $\sigma+\tau$. ME DI CUENTA POR TIPOS HAB.

$$\frac{\Gamma \vdash M : \sigma}{\Gamma \vdash \mathrm{LEFT}_\sigma(M) : \sigma+\tau}\ \text{T-LEFT} \qquad \frac{\Gamma \vdash M : \tau}{\Gamma \vdash \mathrm{RIGHT}_\tau(M) : \sigma+\tau}\ \text{T-RIGHT}$$

$$\frac{\Gamma \vdash M : \sigma+\tau \qquad \Gamma, x:\sigma \vdash N : \rho \qquad \Gamma, y:\tau \vdash O : \rho}{\Gamma \vdash \mathrm{CASE}\ M\ \mathrm{OF}\ \mathrm{LEFT}(x) \rightsquigarrow N;\ \mathrm{RIGHT}(y) \rightsquigarrow O : \rho}\ \text{T-CASE}$$

b) i)

$$\cfrac{\cfrac{\cfrac{}{x:\sigma \vdash x:\sigma}\ \text{T-VAR}}{x:\sigma \vdash \mathrm{LEFT}_\sigma(x) : \sigma+\tau}\ \text{T-LEFT}}{\vdash \lambda x:\sigma .\ \mathrm{LEFT}_\sigma(x) : \sigma \to (\sigma+\tau)}\ \text{T-ABS}$$

↳ CAMBIO QUE ES LEFT POR LEFT_σ.

$$\cfrac{\cfrac{\cfrac{}{x:\tau \vdash x:\tau}\ \text{T-VAR}}{x:\tau \vdash \mathrm{RIGHT}_\tau(x) : \sigma+\tau}\ \text{T-RIGHT}}{\vdash \lambda x:\tau .\ \mathrm{RIGHT}_\tau(x) : \tau \to (\sigma+\tau)}\ \text{T-ABS}$$

ii)

$$\cfrac{\cfrac{\cfrac{\cfrac{}{\Gamma' \vdash x : \sigma+\tau}\ \text{T-VAR} \qquad \cfrac{\cfrac{}{\Gamma'' \vdash f:\sigma\to\rho}\ \text{T-VAR} \qquad \cfrac{}{\Gamma'' \vdash z:\sigma}\ \text{T-VAR}}{\Gamma''=\Gamma', z:\sigma \vdash f\,z : \rho}\ \text{T-APP} \qquad \cfrac{\cfrac{}{\Gamma''' \vdash g:\tau\to\rho}\ \text{T-VAR} \qquad \cfrac{}{\Gamma''' \vdash \sigma:\tau}\ \text{T-VAR}}{\Gamma'''=\Gamma', \sigma:\tau \vdash g\,x : \rho}\ \text{T-APP}}{\Gamma' : x:(\sigma+\tau), f:(\sigma\to\rho), g:(\tau\to\rho) \vdash \mathrm{CASE}\ x\ \mathrm{OF}\ \mathrm{LEFT}(z) \rightsquigarrow f\,z;\ \mathrm{RIGHT}(\sigma) \rightsquigarrow g\,\sigma : \rho}\ \text{T-CASE}}{x:(\sigma+\tau) \vdash \lambda f:(\sigma\to\rho).\lambda g:(\tau\to\rho).\ \mathrm{CASE}\ x\ \mathrm{OF}\ \mathrm{LEFT}(z) \rightsquigarrow f\,z;\ \mathrm{RIGHT}(\sigma) \rightsquigarrow g\,\sigma : (\sigma\to\rho)\to(\tau\to\rho)\to\rho}\ \text{T-ABS}\times 2}{\vdash \lambda x:(\sigma+\tau).\lambda f:(\sigma\to\rho).\lambda g:(\tau\to\rho).\ \mathrm{CASE}\ x\ \mathrm{OF}\ \mathrm{LEFT}(z) \rightsquigarrow f\,z;\ \mathrm{RIGHT}(\sigma) \rightsquigarrow g\,\sigma : (\sigma+\tau)\to(\sigma\to\rho)\to(\tau\to\rho)\to\rho}\ \text{T-ABS}$$

PREG.

c) $V ::= \ldots \mid \mathrm{LEFT}_{\sigma}(V) \mid \mathrm{RIGHT}_{\tau}(V)$ devuelven instancias del tipo.

d) Congruencia

$$\frac{M \to M'}{\mathrm{LEFT}(M) \to \mathrm{LEFT}(M')}\ \text{S-LEFT} \qquad \frac{N \to N'}{\mathrm{RIGHT}(N) \to \mathrm{RIGHT}(N')}\ \text{S-RIGHT}$$

$$\frac{M \to M'}{\mathrm{CASE}\ M\ \mathrm{OF}\ \mathrm{LEFT}(x) \rightsquigarrow N;\ \mathrm{RIGHT}(y) \rightsquigarrow O \ \to\ \mathrm{CASE}\ M'\ \mathrm{OF}\ \mathrm{LEFT}(x) \rightsquigarrow N;\ \mathrm{RIGHT}(y) \rightsquigarrow O}$$

No reduzco N ni O xq es lazy y tiene variables libres

Cómputo:

$$\frac{}{\mathrm{CASE}\ \mathrm{LEFT}_{\sigma}(V)\ \mathrm{OF}\ \mathrm{LEFT}(x) \rightsquigarrow N;\ \mathrm{RIGHT}(y) \rightsquigarrow O \ \to\ N\{x := V\}}\ \text{C-CASE-LEFT}$$

$$\frac{}{\mathrm{CASE}\ \mathrm{RIGHT}_{\tau}(V)\ \mathrm{OF}\ \mathrm{LEFT}(x) \rightsquigarrow N;\ \mathrm{RIGHT}(y) \rightsquigarrow O \ \to\ O\{y := V\}}\ \text{C-CASE-RIGHT}$$

Ejercicio 22 ★

Este ejercicio extiende el Cálculo Lambda tipado con listas. Comenzamos ampliando el conjunto de tipos:

$$\tau \quad ::= \quad \ldots \mid [\tau]$$

donde $[\tau]$ representa el tipo de las listas cuyas componentes son de tipo τ. El conjunto de términos ahora incluye:

$$M, N, O \quad ::= \quad \ldots \mid [\,]_{\tau} \mid M :: N \mid \texttt{case}\, M\ \texttt{of}\ \{[\,] \rightsquigarrow N \mid \underline{h :: t \rightsquigarrow O}\} \mid \texttt{foldr}\, M\, \text{base} \rightsquigarrow N; \underline{\text{rec}(h, r) \rightsquigarrow O}$$

No reducimos términos condicionales, solo guardan.

donde

En cómputo, ligo h,t en O.

En cómputo, R y r hay que sustituir en O.

- $[\,]_{\sigma}$ es la lista vacía cuyos elementos son de tipo σ;
- $M :: N$ agrega M a la lista N;
- $\texttt{case}\ M\ \texttt{of}\ \{[\,] \rightsquigarrow N \mid h :: t \rightsquigarrow O\}$ es el observador de listas. Por su parte, los nombres de variables que se indiquen luego del | (h y t en este caso) son variables que pueden aparecer libres en O y deberán ligarse con la cabeza y cola de la lista respectivamente;
- $\texttt{foldr}\ M\ \text{base} \rightsquigarrow N;\ \text{rec}(h, r) \rightsquigarrow O$ es el operador de recursión estructural (no currificado). Los nombres de variables indicados entre paréntesis (h y r en este caso) son variables que pueden aparecer libres en O y deberán ser ligadas con la cabeza y el resultado de la recursión respectivamente.

Por ejemplo,

Por ejemplo,

- $\texttt{case}\ \text{zero} :: \text{succ}(\text{zero}) :: [\,]_{\text{Nat}}\ \texttt{of}\ \{[\,] \rightsquigarrow \text{false} \mid x :: xs \rightsquigarrow \text{isZero}(x)\} \twoheadrightarrow \text{true}$
- $\texttt{foldr}\ \underline{1} :: \underline{2} :: \underline{3} :: (\lambda x : [\text{Nat}].\, x)\ [\,]_{\text{Nat}}\ \text{base} \rightsquigarrow \text{zero};\ \text{rec}(head, rec) \rightsquigarrow head + rec \twoheadrightarrow \underline{6}$

ELEM :: LISTA

- a) Mostrar el árbol sintáctico para los dos ejemplos dados.

b) Agregar reglas de tipado para las nuevas expresiones.

c) Demostrar el siguiente juicio de tipado (recomendación: marcar variables libres y ligadas en el término antes de comenzar).

$x : \mathsf{Bool}, y : [\mathsf{Bool}] \vdash \texttt{foldr}\ x :: x :: y\ \ \text{base} \leadsto y;\ \text{rec}(y, x) \leadsto \text{if } y \text{ then } x \text{ else } [\,]_{\mathsf{Bool}}\ :\ [\mathsf{Bool}]$

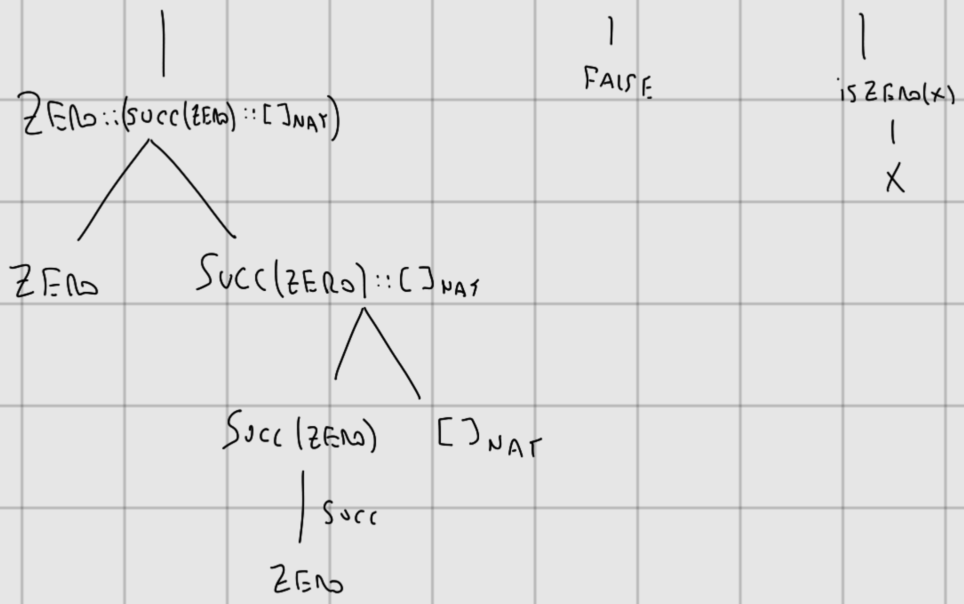
d) Mostrar cómo se extiende el conjunto de valores. Estos deben reflejar la forma de las listas que un programa podría devolver.

e) Agregar los axiomas y reglas de reducción asociados a las nuevas expresiones.

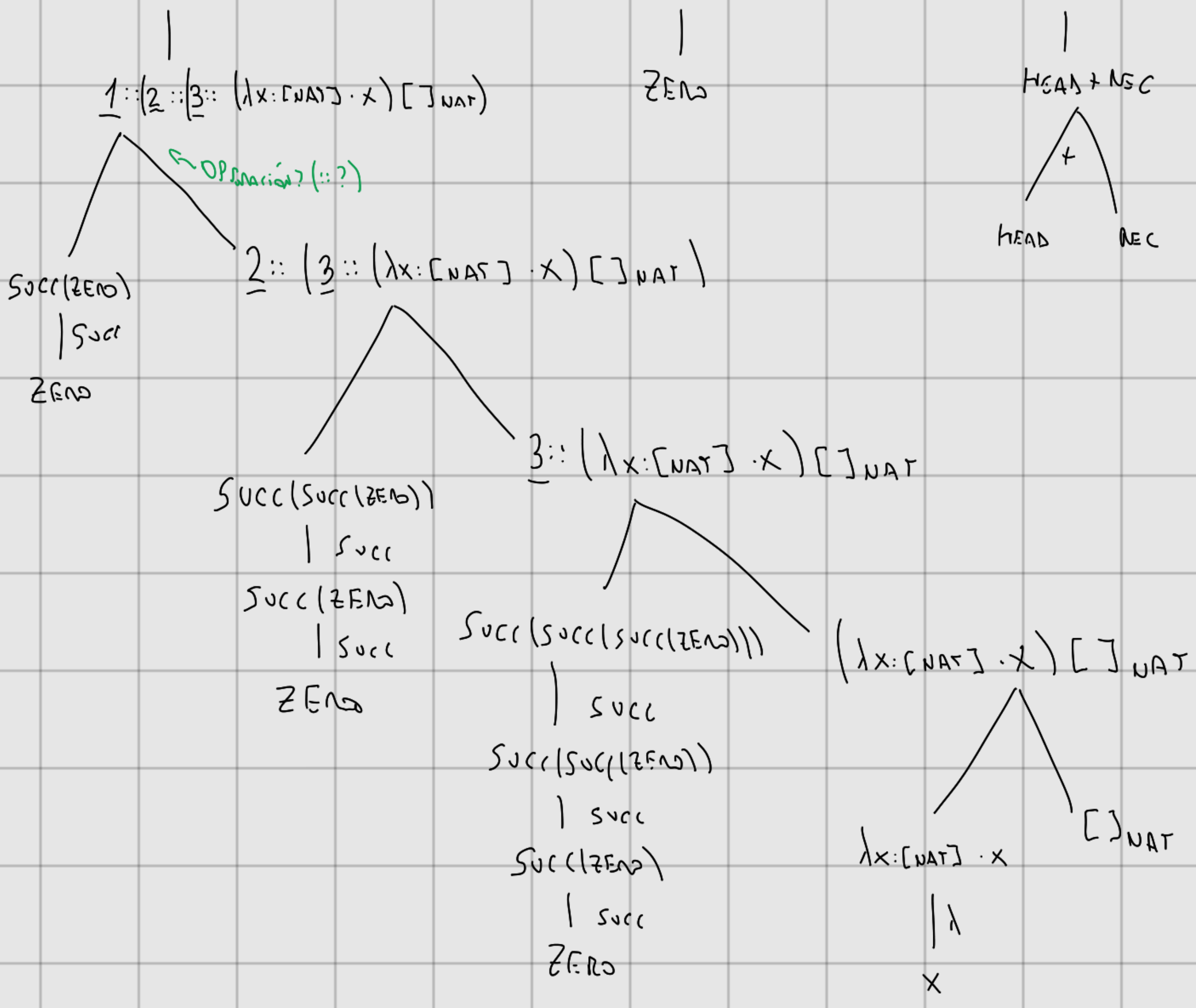
Tip: ¿Cómo sería R::N::n? Si n::N es ELEM::LISTA luego R::(N::n) es ELEM::(ELEM::LISTA) ≡ ELEM::LISTA'

a)

1) CASE ZERO :: SUCC(ZERO) :: [ ]$_{NAT}$ OF { [ ] ⇝ FALSE | x::xs ⇝ isZERO(x) }



2) FOLDR 1 :: 2 :: 3 :: (λx:[NAT].x) [ ]$_{NAT}$ BASE ⇝ ZERO ; REC(HEAD, REC) ⇝ HEAD + REC



b) Tipo Nuevo: [T]

$$\frac{}{\Gamma \vdash [\,]_T : [T]}\ \text{T-VACIA}$$

$$\frac{\Gamma \vdash M : T \qquad \Gamma \vdash N : [T]}{\Gamma \vdash M :: N : [T]}\ \text{T-ADD}$$

$$\frac{\Gamma \vdash M : [T] \qquad \Gamma \vdash N : P \qquad \Gamma, h : T, t : [T] \vdash O : P}{\Gamma \vdash \text{case } M \text{ of } \{ [\,] \leadsto N \mid h :: t \leadsto O \} : P}\ \text{T-CASE}$$

$$\frac{\Gamma \vdash M : [T] \qquad \Gamma \vdash N : P \qquad \Gamma, h : T, r : P \vdash O : P}{\Gamma \vdash \text{foldr } M \text{ base} \leadsto N ;\ \text{rec}(h, r) \leadsto O : P}\ \text{T-FOLDR}$$

c) Recordar renombrar las ligadas.

$$\dfrac{\dfrac{\dfrac{}{\Gamma' \vdash x : \text{Bool}}\,\text{T-VAR} \quad \dfrac{\dfrac{}{\Gamma' \vdash x : \text{Bool}}\,\text{T-VAR} \quad \dfrac{}{\Gamma' \vdash y : [\text{Bool}]}\,\text{T-VAR}}{\Gamma' \vdash x :: y : [\text{Bool}]}\,\text{T-ADD}}{\Gamma' \vdash x :: (x :: y) : [\text{Bool}]}\,\text{T-ADD} \quad \dfrac{}{\Gamma' \vdash y : [\text{Bool}]}\,\text{T-VAR} \quad \dfrac{\dfrac{}{\Gamma'' \vdash y' : \text{Bool}}\,\text{T-VAR} \quad \dfrac{}{\Gamma'' \vdash x' : [\text{Bool}]}\,\text{T-VAR} \quad \dfrac{}{\Gamma'' \vdash [\,]_{\text{Bool}} : [\text{Bool}]}\,\text{T-VACIA}}{\Gamma'' = \Gamma', y' : \text{Bool}, x' : [\text{Bool}] \vdash \text{if } y \text{ then } x \text{ else } [\,]_{\text{Bool}} : [\text{Bool}]}\,\text{T-IF}}{\Gamma' = x : \text{Bool}, y : [\text{Bool}] \vdash \text{foldr } \underbrace{x :: x :: y}_{\text{Contexto}} \text{ base} \leadsto y ;\ \text{rec}(y', x') \leadsto \text{if } y' \text{ then } x' \text{ else } [\,]_{\text{Bool}} : [\text{Bool}]}\,\text{T-FOLDR}$$

Creo que es la de contexto

Cabeza de lista

Res recursión

Por def. vienen ligadas.

d) $V ::= \ldots \mid [\,]_\sigma \mid V :: V$

e) Congruencia: 2 ($M :: N$), 1 case, 1 fold

CÓMPUTO: 2 (CASE [], V::V), 2 FOLD([], V::V)

CONGRUENCIA:

$$\frac{M \to M'}{M :: N \to M' :: N} \text{ C-MN1} \qquad \frac{N \to N'}{V :: N \to V :: N'} \text{ C-MN2}$$

$$\frac{M \to M'}{\text{CASE } M \text{ OF } \{[\,] \rightsquigarrow N \mid h :: \tau \rightsquigarrow O\} \to \text{CASE } M' \text{ OF } \{[\,] \rightsquigarrow N \mid h :: \tau \rightsquigarrow O\}} \text{ C-CASE}$$

$$\frac{M \to M'}{\text{FOLDR } M \text{ BASE} \rightsquigarrow N;\ \text{REC}(h, r) \rightsquigarrow O \to \text{FOLDR } M \text{ BASE} \rightsquigarrow N;\ \text{REC}(h, r) \rightsquigarrow O} \text{ C-FOLDR}$$

CÓMPUTO: DEF FOLDR/CASE PARA AMBOS VAL.

$$\frac{}{\text{FOLDR } [\,]_6 \text{ BASE} \rightsquigarrow N;\ \text{REC}(h, r) \rightsquigarrow O \to N} \text{ C-FOLDR-VACÍO}$$

$$\frac{}{\text{FOLDR } V_1 :: V_2 \text{ BASE} \rightsquigarrow N;\ \text{REC}(h, r) \rightsquigarrow O} \text{ C-FOLDR-LISTA}$$

$$\to O\{\overset{\rightsquigarrow \text{HEAD.}}{h} := V_1,\ \overset{\rightsquigarrow \text{RES RECURSIÓN EN COLA}}{r} := \text{FOLDR } V_2 \text{ BASE} \rightsquigarrow N;\ \text{REC}(h, r) \rightsquigarrow O\}$$

$$\frac{}{\text{CASE } [\,]_6 \text{ OF } \{[\,] \rightsquigarrow N \mid h :: \tau \rightsquigarrow O\} \to N} \text{ C-CASE-VACÍO}$$

$$\frac{}{\text{CASE } V_1::V_2 \text{ OF } \{[\,] \rightsquigarrow N \mid h::t \rightsquigarrow O\} \to O\{h := V_1,\ t := V_2\}} \text{ C-CASE-LISTA}$$

Ejercicio 23 ★

A partir de la extensión del ejercicio 22, definir una nueva extensión que incorpore expresiones de la forma map(M, N), donde N es una lista y M una función que se aplicará a cada uno de los elementos de N.
Importante: tener en cuenta las anotaciones de tipos al definir las reglas de tipado y semántica.

Sólo van en los términos anotados.

$M, N, O ::= \ldots \mid \text{MAP}(M, N)$

TIPADO:

$$\frac{\Gamma \vdash M : (\tau \to \rho) \qquad \Gamma \vdash N : [\tau]}{\Gamma \vdash \text{MAP}(M, N) : [\rho]} \text{ T-MAP}$$

SEMÁNTICA:

CONGRUENCIA:

$$\frac{M \to M'}{\text{MAP}(M, N) \to \text{MAP}(M', N)} \text{ C-MAP1} \qquad \frac{N \to N'}{\text{MAP}(V, N) \to \text{MAP}(V, N')} \text{ C-MAP2}$$

CÓMPUTO: PARA CADA VALOR DEF MAP.

Acá tipamos aunque sea semántico porque el MAP al ejecutar puede cambiar el tipo de salida

$$\frac{\Gamma \vdash V : (\tau \to \rho)}{\text{MAP}(V, [\,]_\tau) \to [\,]_\rho} \text{ C-MAP-VACIA}$$

Acá sí va xq en el término está anotado!

$$\frac{\Gamma \vdash V : (\tau \to \rho)}{\text{MAP}(V, V_1 :: V_2) \to V\,V_1 :: (\text{MAP}(V, V_2))} \text{ C-MAP-LISTA}$$

si es vacío cae en MAP$(V, [\,]_\tau)$

Ejercicio 24 ★

A partir de la extensión del ejercicio 22, agregaremos términos para representar listas por comprensión, con un selector y una guarda, de la siguiente manera: $[M \mid x \leftarrow S, P]$, donde x es el nombre de una variable que puede aparecer libre en los términos M y P. La semántica es análoga a la de Haskell: para cada valor de la lista representada por el término S, se sustituye x en P y, de resultar verdadero, se agrega M con x sustituido al resultado. Definir las reglas de tipado, el conjunto de valores y las reglas de semántica para esta extensión.

$M, N, O ::= \ldots \mid [M \mid x \leftarrow S, P]$

TIPADO:

$$\frac{\Gamma \vdash M : [\tau] \qquad \Gamma, x : \tau \vdash S : [\tau] \qquad \Gamma \vdash P : (\tau \to Bool)}{\Gamma \vdash [M \mid x \leftarrow S, P] : [\tau]} \text{ T-LC}$$

VALORES: quedan igual. Al final reduce a $V_1 :: V_2$, luego es la lista valores. A nivel programa no me interesa la lista en forma de comprensión

SEMÁNTICA:

CONGRUENCIA:

$$\frac{S \to S'}{[M \mid x \leftarrow S, P] \to [M \mid x \leftarrow S', P]} \text{ C-COMPRENSIÓN}$$

CÓMPUTO: ¿Debería considerar agregar operaciones de cómputo con $[\,]_\sigma$ y $V_1 :: V_2$ si justamente por comprensión genera listas? Termina siendo de forma $V_1 :: V_2$.

Exacto, no cambia nada más. El valor resultado es o bien $[\,]_\sigma$ o $V :: V$

~> me dijeron que no le de bola.

Ejercicio 25 *(Conectivos booleanos)*

Definir como macros (azúcar sintáctica) los términos **Not**, **And**, **Or**, **Xor**, que simulen desde la reducción los conectivos clásicos usuales, por ej. $And\ M\ N \twoheadrightarrow \mathsf{true} \Leftrightarrow M \twoheadrightarrow \mathsf{true}$ y $N \twoheadrightarrow \mathsf{true}$.

Notar que definir una macro no es lo mismo que hacer una extensión. Por ejemplo, definir el término $I_\sigma \stackrel{\text{def}}{=} \lambda x : \sigma . x$, que es la función identidad del tipo σ, es distinto de extender la sintaxis del lenguaje con términos de la forma $I(M)$, lo cual además requeriría agregar nuevas reglas de tipado y de evaluación.

~ ¿Cómo digo que son "evaluables" a nivel TRUE/FALSE? ~> Esto es a nivel cómputo. Asumimos que M N reducen a Bool

$And_{Bool} = \lambda a : Bool . \lambda b : Bool . \text{ if } a \text{ then } b \text{ else } a$

- if T then F else T : F
- if T then T else T : T
- if F then T else F : F
- if F then F else F : F

$Not_{Bool} = \lambda a : b .$

Ejercicio 27 ★

Se desea extender el Cálculo Lambda tipado con colas bidireccionales (también conocidas como *deque*). Se extenderán los tipos y términos de la siguiente manera:

$$\sigma ::= \cdots \mid \mathsf{Cola}_\sigma \qquad M ::= \cdots \mid \langle\rangle_\sigma \mid M \bullet M \mid \mathsf{próximo}(M) \mid \mathsf{desencolar}(M) \mid \mathsf{case}\ M\ \mathsf{of}\ \langle\rangle \rightsquigarrow M\,;\, c \bullet x \rightsquigarrow M$$

donde $\langle\rangle_\sigma$ es la cola vacía en la que se pueden encolar elementos de tipo σ; $M_1 \bullet M_2$ representa el agregado del elemento M_2 al **final** de la cola M_1; los observadores $\mathsf{próximo}(M_1)$ y $\mathsf{desencolar}(M_1)$ devuelven, respectivamente, el primer elemento de la cola (el primero que se encoló), y la cola sin el primer elemento (estos dos últimos solo tienen sentido si la cola no es vacía); y el observador $\mathsf{case}\ M_1\ \mathsf{of}\ \langle\rangle \rightsquigarrow M_2; c \bullet x \rightsquigarrow M_3$ permite operar con la cola en sentido contrario, accediendo al último elemento encolado (cuyo valor se ligará a la variable x en M_3) y al resto de la cola (que se ligará a la variable c en el mismo subtérmino). → ¿Cómo puedo tirar error, o eso es un "cuidado" cuando haga la macro?

1. Introducir las reglas de tipado para la extensión propuesta.
2. Definir el conjunto de valores y las nuevas reglas de reducción. Pueden usar los conectivos booleanos de la guía. ⇝ me dijeron que no es necesario. **No es necesario escribir las reglas de congruencia,** basta con indicar cuántas son.

   Pista: puede ser necesario mirar más de un nivel de un término para saber a qué reduce. → ej para prox y desencolar hago $\underbrace{(V_1 \cdot V_2)}_{\text{más de 1 anidamiento}} \cdot V_3$
3. Mostrar paso por paso cómo reduce la expresión: $\mathsf{case}\ \langle\rangle_{\mathsf{Nat}} \bullet \underline{1} \bullet \underline{0}\ \mathsf{of}\ \langle\rangle \rightsquigarrow \mathsf{próximo}(\langle\rangle_{\mathsf{Bool}}); c \bullet x \rightsquigarrow \mathsf{isZero}(x)$
4. Definir como macro la función $\texttt{último}_\tau$, que dada una cola devuelve el último elemento que se encoló en ella. Si la cola es vacía, puede colgarse o llegar a una forma normal bien tipada que no sea un valor. Dar un juicio de tipado válido para esta función (no es necesario demostrarlo).

?

1)

$$\frac{}{\Gamma \vdash \langle\rangle_\sigma : \mathrm{Cola}_\sigma}\ \text{T-VACIA}$$

$$\frac{\Gamma \vdash M : \mathrm{Cola}_\sigma \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash M \bullet N : \mathrm{Cola}_\sigma}\ \text{T-ADD}$$

$$\frac{\Gamma \vdash M : \mathrm{Cola}_\sigma}{\Gamma \vdash \mathrm{Proximo}(M) : \sigma}\ \text{T-PROXIMO}$$

$$\frac{\Gamma \vdash M : \mathrm{Cola}_\sigma}{\Gamma \vdash \mathrm{DESENCOLAR}(M) : \mathrm{Cola}_\sigma}\ \text{T-DESENCOLAR}$$

$$\frac{\Gamma \vdash M : \mathrm{Cola}_\sigma \qquad \Gamma \vdash N : \rho \qquad \Gamma, c : \mathrm{Cola}_\sigma, x : \sigma \vdash P : \rho}{\Gamma \vdash \mathrm{Case}\ M\ \mathrm{OF}\ \langle\rangle \rightsquigarrow N;\ c \cdot x \rightsquigarrow P : \rho}\ \text{T-CASE}$$

2) $V ::= \ldots \mid \langle\rangle_\sigma \mid V \cdot V$ ⇝ La forma de construir mi lista.

Reglas congruencia: 5.

Cómputo: 6. 2 case, Prox, Desencolar

$$\frac{}{\mathrm{Case}\ \langle\rangle_\sigma\ \mathrm{OF}\ \langle\rangle \rightsquigarrow N;\ C \cdot x \rightsquigarrow P \to N}\ \text{C-CASE-VACIA}$$

$$\frac{}{\mathrm{CASE}\ V_1 \cdot V_2\ \mathrm{OF}\ \langle\rangle \rightsquigarrow N;\ C \cdot x \rightsquigarrow P \to P\{C := V_2, x := V_1\}}\ \text{C-CASE-LISTA}$$

$$\frac{}{\text{PROXIMO}((v_1 \cdot v_2) \cdot v_3) \to \text{PROXIMO}(v_1 \cdot v_2)}\ \text{C-PROXIMO-LISTA}$$

$$\frac{}{\text{PROX}(\langle\rangle_\sigma \cdot v) \to v}\ \text{C-PROX-VACIO}$$

$$\frac{}{\text{DESENCOLAR}((v_1 \cdot v_2) \cdot v_3) \to \text{DESENCOLAR}(v_1 \cdot v_2) \cdot v_3}\ \text{C-DESENCOLAR-LISTA}$$

REDUCE A C-DESENCOLAR-VACIO $\rightsquigarrow \langle\rangle_\sigma \cdot v_3$

$$\frac{}{\text{DESENCOLAR}(\langle\rangle_\sigma \cdot v_1) \to \langle\rangle_\sigma}\ \text{C-DESENCOLAR-VACIO}$$

RAZÓN POR LA QUE HACEMOS PRÓXIMO/DESENCOLAR CON EL VALOR VACÍO: necesito alguna forma de llegar al único elemento que queda, C-PROXIMO no me sirve xq reduce a Proximo(v1·v2) ¿Qué pasa si V1 es otra lista, o es []? ¿Tiene sentido hacer PROXIMO([]·V)? ¿A qué reduce? Para manejar ese caso, agrego C-PROX-VACIO. Análogo para DESENCOLAR.

$$\text{CASE}\ \overbrace{\langle\rangle_{\text{NAT}} \cdot \underline{1}}^{v_1} \cdot \overbrace{\underline{0}}^{v_2}\ \text{OF}\ \langle\rangle \rightsquigarrow \text{PROXIMO}(\langle\rangle_{\text{BOOL}})\ ;\ c \cdot x \rightsquigarrow \text{ISZERO}(x)$$

ESTA SON VALORES LA GUARDA.

Notar q tenía FN pero no trabó.

C-CASE-LISTA

$\to$ $\text{ISZERO}(0)\ \{x := 0,\ c := \langle\rangle_{\text{NAT}} \cdot 1\}$

ISZERO-0

$\to$ TRUE

EL FIX.

4) ÚLTIMO$_\sigma$ $= \lambda c{:}\text{Cola}_\sigma .\ \text{CASE}\ c\ \text{OF}\ \langle\rangle \rightsquigarrow \bot_\sigma\ ;\ c \cdot x \rightsquigarrow x$

$\Theta \rightsquigarrow \text{PROX}(\langle\rangle_\sigma) \rightsquigarrow$ ES F.N PERO NO ES VALOR.

$$\dfrac{\dfrac{}{a{:}\text{Cola}_\sigma \vdash a{:}\text{Cola}_\sigma}\ \text{T-VAR} \qquad \dfrac{}{c{:}\text{Cola}_\sigma \vdash \bot_\sigma : \sigma}\ ? \qquad \dfrac{}{c{:}\text{Cola}_\sigma, x{:}\sigma \vdash x{:}\sigma}\ \text{T-VAR}\ \checkmark}{\dfrac{a{:}\text{Cola}_\sigma \vdash \text{CASE}\ a\ \text{OF}\ \langle\rangle \rightsquigarrow \bot_\sigma\ ;\ c \cdot x \rightsquigarrow x : \sigma}{\emptyset \vdash \lambda a{:}\text{Cola}_\sigma .\ \text{CASE}\ a\ \text{OF}\ \langle\rangle \rightsquigarrow \bot_\sigma\ ;\ c \cdot x \rightsquigarrow x : \text{Cola}_\sigma \to \sigma}\ \text{T-ABS}}\ \text{T-CASE}$$

Ejercicio 3 - Cálculo Lambda Tipado

Se desea extender el cálculo lambda simplemente tipado para modelar **listas ordenadas**. Una lista ordenada se construye de manera similar a una lista común, pero sus observadores se comportan de manera diferente: la cabeza de la lista es siempre el mínimo elemento, y su cola es el resto de los elementos.

Estas listas tienen como precondición que el tipo de sus elementos cuente con una operación de comparación $<$ (no es necesario escribir nada al respecto, suponemos que vale).

Se extienden los tipos y expresiones de la siguiente manera:

$$\tau ::= \cdots \mid [\tau]^<$$
$$M ::= \cdots \mid [\,]^<_\tau \mid M ::^< M \mid \texttt{head}^<(M) \mid \texttt{tail}^<(M)$$

donde:

- $[\tau]^<$ es el tipo de las listas ordenadas con elementos de tipo τ.
- $[\,]^<_\tau$ es una lista ordenada vacía que admite valores de tipo τ.
- $M_1 ::^< M_2$ es la lista resultante de agregar el elemento M_1 a la lista M_2[1].
- $\texttt{head}^<(M)$ denota el mínimo elemento de la lista M.
- $\texttt{tail}^<(M)$ es la lista ordenada con todos los elementos de M excepto el mínimo (si M tiene más de una aparición de su mínimo elemento, se elimina una de ellas).

a) Introducir las reglas de tipado para la extensión propuesta.

b) Definir el conjunto de valores y las nuevas reglas de reducción en un paso. Tener en cuenta que una lista vacía no tiene cabeza ni cola.

Pista: puede ser necesario mirar más de un nivel de un término para saber a qué reduce.

c) Mostrar paso por paso cómo reduce la expresión:

$\mathsf{head}^<((\lambda x:\mathsf{Nat}.\underline{2})\mathsf{zero} ::^< \underline{1} ::^< [\,]^<_{\mathsf{Nat}})$

Se pueden considerar dadas las siguientes reglas:

$$\frac{M_1 \to M'}{M_1 < M_2 \to M' < M_2}(\text{E-}<_1) \qquad \frac{M_2 \to M'}{V < M_2 \to V < M'}(\text{E-}<_2)$$

$$\frac{}{V < \mathsf{zero} \to \mathsf{False}}(\text{E-}<_F) \qquad \frac{}{\mathsf{zero} < \mathsf{Succ}(V) \to \mathsf{True}}(\text{E-}<_T)$$

$$\frac{}{\mathsf{Succ}(V_1) < \mathsf{Succ}(V_2) \to V_1 < V_2}(\text{E-}<_S)$$

[1] Notar que M_1 no necesariamente es la cabeza de la lista, y que $::^<$ es un constructor, no es responsable de ordenar los elementos en la lista. El orden aparece al observarla. Por ejemplo, $\underline{1} ::^< \underline{2} ::^< [\,]^<_{\mathsf{Nat}}$ y $\underline{2} ::^< \underline{1} ::^< [\,]^<_{\mathsf{Nat}}$ son dos valores de tipo $[\mathsf{Nat}]^<$ que tienen la misma cabeza y la misma cola.

3)

$T ::= \dots \mid [T]^<$

$M ::= \dots \mid [\,]^<_T \mid M ::^< M \mid \mathrm{HEAD}^<(M) \mid \mathrm{TAIL}^<(M)$

La cabeza se sustenta al menor.

Ej: $M_1 ::^< M_2 \leadsto \underline{1} ::^< \underline{2} ::^< [\,]_{NAT} \equiv \underline{2} ::^< \underline{1} ::^< [\,]_{NAT}$

ya HEAD = 1 TAIL $2 ::^< [\,]_{NAT}$

a)

$$\frac{}{\Gamma \vdash [\,]^<_T : [T]^<}\text{T-VACÍA} \qquad \frac{\Gamma \vdash M : T \quad \Gamma \vdash N : [T]^<}{\Gamma \vdash M ::^< N : [T]^<}\text{T-ADD}$$

$$\frac{\Gamma \vdash M : [T]^<}{\Gamma \vdash \mathrm{HEAD}^<(M) : T}\text{T-HEAD} \qquad \frac{\Gamma \vdash M : [T]^<}{\Gamma \vdash \mathrm{TAIL}^<(M) : [T]^<}\text{T-TAIL}$$

b) $V ::= \dots \mid [\,]^<_T \mid V ::^< V$ (cada op con estos 2 val)

Congruencia:

$$\frac{M \to M'}{M ::^< N \to M' ::^< N}\text{E-ADD1} \qquad \frac{N \to N'}{V ::^< N \to V ::^< N'}\text{E-ADD2}$$

$$\frac{M \to M'}{\mathrm{HEAD}^<(M) \to \mathrm{HEAD}(M')}\text{E-HEAD} \qquad \frac{M \to M'}{\mathrm{TAIL}^<(M) \to \mathrm{TAIL}(M')}\text{E-TAIL}$$

Cómputo: Al definir head/tail con vacía pierdo progreso

$$\frac{}{\mathrm{HEAD}^<(V_1 ::^< [\,]_T) \to V_1}\text{C-HEAD-EMPTY}$$

$$\frac{}{\mathrm{HEAD}^<(V_1 ::^< (V_2 ::^< V_3)) \to \text{if} (V_1 < V_2 \text{ THEN } (\mathrm{HEAD}(V_1 ::^< V_3)) \text{ ELSE } (\mathrm{HEAD}(V_2 ::^< V_3))}\text{C-HEAD-LISTA}$$

$$\frac{}{\mathrm{TAIL}^<(V_1 ::^< [\,]_T) \to [\,]_T}\text{C-TAIL-EMPTY}$$

$$\frac{}{\mathrm{TAIL}(V_1 ::^< (V_2 ::^< V_3)) \to \text{IF } (\mathrm{HEAD}(V_1 ::^< (V_2 ::^< V_3)) == V_1 \text{ THEN } (V_2 ::^< V_3) \text{ ELSE } V_1 ::^< (\mathrm{TAIL}(V_2 ::^< V_3))}\text{C-TAIL-LISTA}$$

if $v_1 < v_2$ then $v_2 ::^< \mathrm{tail}(v_1 ::^< v_3)$ else $v_1 ::^< \mathrm{tail}(v_2 ::^< v_3)$

c) $\mathrm{HEAD}^<((\lambda x:NAT.\,2)\,\mathrm{ZERO} ::^< \underline{1} ::^< [\,]^<_{NAT})$

$\xrightarrow{\{E\text{-}HEAD\}\,(B)} \mathrm{HEAD}^<(\underline{2} ::^< \underline{1} ::^< [\,]^<_{NAT})$

$\xrightarrow{\{C\text{-}HEAD\text{-}LISTA\}}$ if $(\underline{2} < \underline{1})$ THEN $(\mathrm{HEAD}^<(\underline{2} ::^< [\,]_{NAT}))$ ELSE $(\mathrm{HEAD}^<(\underline{1} ::^< [\,]_{NAT}))$

$\xrightarrow{\text{Reescribo}}$ if $(\mathrm{Succ}(\mathrm{Succ}(\mathrm{zero})) < \mathrm{Succ}(\mathrm{zero}))$ THEN $(\mathrm{HEAD}^<(\mathrm{Succ}(\mathrm{Succ}(\mathrm{zero})) ::^< [\,]_{NAT})$ ELSE $(\mathrm{HEAD}^<(\mathrm{Succ}(\mathrm{zero})) ::^< [\,]_{NAT})$

$\xrightarrow{\{E\text{-}IF\}\,\{E\text{-}<_S\}}$ IF $(\mathrm{Succ}(\mathrm{zero}) < \mathrm{zero})$ THEN …

$\xrightarrow{E\text{-}<_F,\ E\text{-}IF}$ IF FALSE THEN …

$\xrightarrow{\{E\text{-}IFFALSE\}} \mathrm{HEAD}^<((\mathrm{Succ}(\mathrm{zero})) ::^< [\,]_{NAT})$

$\xrightarrow{\{C\text{-}HEADEMPTY\}} \mathrm{Succ}(\mathrm{zero}) \equiv \underline{1}$