

## Práctica N° 4 - Cálculo- $\lambda$ : Tipado y Semántica Operacional

Los ejercicios marcados con el símbolo ★ constituyen un subconjunto mínimo de ejercitación. Sin embargo, aconsejamos fuertemente hacer todos los ejercicios.

A menos que se especifiquen las extensiones a utilizar, trabajaremos con el cálculo  $\lambda$  con los tipos Bool, Nat y funciones.

Notación para este segmento de la materia:

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

19 ★

Gramáticas a tener en cuenta:

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

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

- Tipos  
 $\tau ::= \text{Bool} \mid \text{Nat} \mid \tau \rightarrow \tau$

### 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) $x$                                 | i) $\lambda x: \text{Bool}. \text{succ}(x)$                                |
| b) $x x$                               | j) $\lambda x: \text{if true then Bool else Nat}. x$                       |
| c) $M$                                 | k) $\sigma$  |
| d) $M M$                               | l) Bool  |
| e) true false                          | m) $\text{Bool} \rightarrow \text{Bool}$                                   |
| f) true succ(false true)               | n) $\text{Bool} \rightarrow \text{Bool} \rightarrow \text{Nat}$            |
| g) $\lambda x. \text{isZero}(x)$       | ñ) $(\text{Bool} \rightarrow \text{Bool}) \rightarrow \text{Nat}$          |
| h) $\lambda x: \sigma. \text{succ}(x)$ | o) succ true   |
|  | p) $\lambda x: \text{Bool}. \text{if zero then true else zero succ(true)}$ |

#### Ejercicio 2

Mostrar un término que utilice al menos una vez **todas** las reglas de generación de la gramática de los términos y exhibir su *árbol sintáctico*.

#### Ejercicio 3 ★

- a) Marcar las ocurrencias del término  $x$  como **subtérmino** en  $\lambda x: \text{Nat}. \text{succ}((\lambda x: \text{Nat}. x) x)$ .
- b) ¿Ocurre  $x_1$  como subtérmino en  $\lambda x_1: \text{Nat}. \text{succ}(x_2)$ ? **NO**
- c) ¿Ocurre  $x (y z)$  como subtérmino en  $(u x)(y z)$ ? **NO**

#### Ejercicio 4 ★

Para los siguientes términos:

- a)  $u \ x \ (y \ z) \ (\lambda v : \text{Bool}. v \ y)$
- b)  $(\lambda x : \text{Bool} \rightarrow \text{Nat} \rightarrow \text{Bool}. \lambda y : \text{Bool} \rightarrow \text{Nat}. \lambda z : \text{Bool}. x \ z \ (y \ z)) \ u \ v \ w$
- c)  $w \ (\lambda x : \text{Bool} \rightarrow \text{Nat} \rightarrow \text{Bool}. \lambda y : \text{Bool} \rightarrow \text{Nat}. \lambda z : \text{Bool}. x \ z \ (y \ z)) \ u \ v$

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?  
 $(\lambda x : \text{Bool} \rightarrow \text{Nat} \rightarrow \text{Bool}. \lambda y : \text{Bool} \rightarrow \text{Nat}. \lambda z : \text{Bool}. x \ z \ (y \ z)) \ u$

### TIPADO

#### Ejercicio 5

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

#### 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)  $\vdash \text{if true then zero else succ(zero)} : \text{Nat}$
- b)  $x : \text{Nat}, y : \text{Bool} \vdash \text{if true then false else } (\lambda z : \text{Bool}. z) \ \text{true} : \text{Bool}$
- c)  $\vdash \text{if } \lambda x : \text{Bool}. x \ \text{then zero else succ(zero)} : \text{Nat}$
- d)  $x : \text{Bool} \rightarrow \text{Nat}, y : \text{Bool} \vdash x \ y : \text{Nat}$

#### Ejercicio 7 ★

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

$$\frac{\Gamma \vdash M : \tau}{\Gamma \vdash \lambda x : \sigma. M : \sigma \rightarrow \tau} \rightarrow_{i2}$$

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

#### Ejercicio 8

Determinar qué tipo representa  $\sigma$  en cada uno de los siguientes juicios de tipado.

- a)  $\vdash \text{succ(zero)} : \sigma$
- b)  $\vdash \text{isZero(succ(zero))} : \sigma$
- c)  $\vdash \text{if (if true then false else false) then zero else succ(zero)} : \sigma$

#### Ejercicio 9 (Tipos habitados) ★

Decimos que un tipo  $\tau$  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  $\tau$ . Por ejemplo, dado un tipo  $\sigma$ , la identidad  $\lambda x : \sigma. x$  es un habitante del tipo  $\sigma \rightarrow \sigma$ . Demostrar que los siguientes tipos están habitados (para cualquier  $\sigma, \tau$  y  $\rho$ ):

- ✓ a)  $\sigma \rightarrow \tau \rightarrow \sigma$
- ↓ b)  $(\sigma \rightarrow \tau \rightarrow \rho) \rightarrow (\sigma \rightarrow \tau) \rightarrow \sigma \rightarrow \rho$

- ✓ c)  $(\sigma \rightarrow \tau \rightarrow \rho) \rightarrow \tau \rightarrow \sigma \rightarrow \rho$   
 ✓ d)  $(\tau \rightarrow \rho) \rightarrow (\sigma \rightarrow \tau) \rightarrow \sigma \rightarrow \rho$

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  $\rightarrow$  por  $\Rightarrow$ , las fórmulas habitadas son siempre tautologías? ¿Las tautologías son siempre fórmulas habitadas?

### Ejercicio 10 ★

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

- ✓ a)  $x: \sigma \vdash \text{isZero}(\text{succ}(x)) : \tau$   
 ✓ b)  $\vdash (\lambda x: \sigma. x)(\lambda y: \text{Bool}. \text{zero}) : \sigma$   
 ✗ c)  $y: \tau \vdash \text{if } (\lambda x: \sigma. x) \text{ then } y \text{ else } \text{succ}(\text{zero}) : \sigma$   
 ✓ d)  $x: \sigma \vdash x y : \tau$   
 ✓ e)  $x: \sigma, y: \tau \vdash x y : \tau$   
 ✓ f)  $x: \sigma \vdash x \text{ true} : \tau$   
 ✗ g)  $x: \sigma \vdash x \text{ true} : \sigma$   
 ✗ h)  $x: \sigma \vdash x x : \tau$

### Ejercicio 11 (Debilitamiento y fortalecimiento)

Demostrar las siguientes propiedades, procediendo por inducción en la derivación del juicio correspondiente:

- Si  $\Gamma \vdash M : \sigma$  es un juicio de tipado derivable y  $x$  es una variable que no aparece en  $\Gamma$ , entonces  $\Gamma, x: \tau \vdash M : \sigma$  es derivable para todo tipo  $\tau$ . Esta regla se conoce como *debilitamiento* o *weakening*.
- Si  $\Gamma, x: \tau \vdash M : \sigma$  es un juicio de tipado derivable tal que  $x$  no aparece libre en  $M$ , entonces  $\Gamma \vdash M : \sigma$  es derivable para todo tipo  $\tau$ . Esta regla se conoce como *fortalecimiento* o *strengthening*.
- Dar un contraejemplo para fortalecimiento en el caso en el que  $x$  aparece libre en  $M$ .

### 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  $M$ .

## SEMÁNTICA

### Ejercicio 13 ★

Sean  $\sigma, \tau, \rho$  tipos. Según la definición de sustitución, calcular:

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

Renombrar variables en ambos términos para que las sustituciones no cambien su significado.

### Ejercicio 14 (Conmutación de sustituciones)

Sean  $M, N$  y  $P$  términos del cálculo- $\lambda$ .

- a) Por inducción en la estructura del término  $M$ , demostrar que si  $x$  no aparece libre en  $P$  y  $x \neq y$ , entonces:

$$M\{x := N\}\{y := P\} = M\{y := P\}\{x := N\{y := P\}\}$$

- b) Dar un contraejemplo para la ecuación de arriba cuando  $x$  aparece libre en  $P$ .

### Ejercicio 15 (Valores) ★

Dado el conjunto de valores visto en clase:

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

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

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

### Ejercicio 16 (Programa, Forma Normal) ★

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

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**.

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

### Ejercicio 17 (Determinismo)

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

### Ejercicio 18

- a) ¿Da lo mismo evaluar  $\text{succ}(\text{pred}(M))$  que  $\text{pred}(\text{succ}(M))$ ? ¿Por qué?  
 b) ¿Es verdad que para todo término  $M$  vale  $\text{isZero}(\text{succ}(M)) \twoheadrightarrow \text{false}$ ? Si no lo es, ¿para qué términos vale?  
 c) ¿Para qué términos  $M$  vale  $\text{isZero}(\text{pred}(M)) \twoheadrightarrow \text{true}$ ? (Hay infinitos).

### Ejercicio 19

Al agregar la siguiente regla para las abstracciones:

**Si**  $M \rightarrow M'$ , **entonces:**  $\lambda x: \tau. M \rightarrow \lambda x: \tau. M' \quad (\xi)$

- a) Repensar el conjunto de valores para respetar esta modificación, pensar por ejemplo si  $(\lambda x: \text{Bool. } (\lambda y: \text{Bool. } y) \text{ true})$  es o no un valor.  
 b) ¿Qué reglas deberían modificarse para no perder el determinismo?  
 c) Utilizando el cálculo modificado y los valores definidos, reducir la siguiente expresión  $(\lambda x: \text{Nat} \rightarrow \text{Nat. } x \underline{23}) (\lambda x: \text{Nat. } \text{pred}(\text{succ}(\text{zero})))$   
 ¿Qué se puede concluir entonces? ¿Es una buena idea agregar esta regla?

## 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 siguientes reglas de tipado y semántica:

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

$$\begin{array}{ll} \text{Si } M \rightarrow M', \text{ entonces: } M + N \rightarrow M' + N & (+_{c1}) \\ \text{Si } N \rightarrow N', \text{ entonces: } V + N \rightarrow V + N' & (+_{c2}) \\ V + \text{zero} \rightarrow V & (+_0) \\ V_1 + \text{succ}(V_2) \rightarrow \text{succ}(V_1) + V_2 & (+_{\text{succ}}) \end{array}$$

### Ejercicio 20 (Pares, o productos) ★

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

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

donde  $\sigma \times \tau$  es el tipo de los pares cuya primera componente es de tipo  $\sigma$  y cuya segunda componente es de tipo  $\tau$ ,  $\langle M, N \rangle$  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  $\sigma$ ,  $\tau$  y  $\rho$ , exhibir habitantes de los siguientes tipos:

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

✓ 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?

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

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

$$\begin{array}{ll} \tau & ::= \dots \mid \tau + \tau \\ M & ::= \dots \mid \text{left}_\tau(M) \mid \text{right}_\tau(M) \mid \text{case } M \text{ of } \text{left}(x) \rightsquigarrow M_1 \parallel \text{right}(y) \rightsquigarrow M_2 \end{array}$$

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

- a) Definir reglas de tipado para los nuevos constructores de términos.  
b) Usando las reglas de tipado anteriores, y dados los tipos  $\sigma$ ,  $\tau$  y  $\rho$ , exhibir habitantes de los siguientes tipos:

- I) Inyecciones:  $\sigma \rightarrow (\sigma + \tau)$  y  $\tau \rightarrow (\sigma + \tau)$ .
- II) Análisis de casos:  $(\sigma + \tau) \rightarrow (\sigma \rightarrow \rho) \rightarrow (\tau \rightarrow \rho) \rightarrow \rho$ .
- III) Conmutatividad:  $(\sigma + \tau) \rightarrow (\tau + \sigma)$ .
- IV) Asociatividad:  $((\sigma + \tau) + \rho) \rightarrow (\sigma + (\tau + \rho))$  y  $(\sigma + (\tau + \rho)) \rightarrow ((\sigma + \tau) + \rho)$ .
- V) Distributividad del producto sobre la suma:  $(\sigma \times (\tau + \rho)) \rightarrow ((\sigma \times \tau) + (\sigma \times \rho))$  y  $((\sigma \times \tau) + (\sigma \times \rho)) \rightarrow (\sigma \times (\tau + \rho))$ .

VI) Ley de los exponentes<sup>1</sup>:  $((\sigma + \tau) \rightarrow \rho) \rightarrow ((\sigma \rightarrow \rho) \times (\tau \rightarrow \rho))$  y  $((\sigma \rightarrow \rho) \times (\tau \rightarrow \rho)) \rightarrow ((\sigma + \tau) \rightarrow \rho)$ .

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.

## Ejercicio 22 ★

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

$$\tau ::= \dots \mid [\tau]$$

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

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

donde

- $[]_{\sigma}$  es la lista vacía cuyos elementos son de tipo  $\sigma$ ;
- $M :: N$  agrega  $M$  a la lista  $N$ ;
- $\text{case } M \text{ of } \{ [] \rightsquigarrow N \mid h :: t \rightsquigarrow O \}$  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;
- $\text{foldr } M \text{ base } \rightsquigarrow N; \text{rec}(h, r) \rightsquigarrow O$  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,

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

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).

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

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.

## Ejercicio 23 ★

A partir de la extensión del ejercicio 22, definir una nueva extensión que incorpore expresiones de la forma  $\text{map}(M, N)$ , 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.

## 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.

<sup>1</sup>Notemos que si escribimos  $\sigma \rightarrow \tau$  como  $\tau^{\sigma}$ , las dos expresiones nos dicen que  $\rho^{\sigma+\tau} \longleftrightarrow \rho^{\sigma} \times \rho^{\tau}$ .

### 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.  $And\ M\ N \rightarrow true \Leftrightarrow M \rightarrow true\ y\ N \rightarrow true$ .

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  $\sigma$ , 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.

### Ejercicio 26

Definir las siguientes funciones en Cálculo Lambda con Listas (visto en el ejercicio 22). Pueden definirse como macros o como extensiones al cálculo.

**Nota:** en este ejercicio usamos la notación  $M : \sigma$  para decir que la expresión  $M$  a definir debe tener tipo  $\sigma$  en cualquier contexto.

- $head_\sigma : [\sigma] \rightarrow \sigma$  y  $tail_\sigma : [\sigma] \rightarrow [\sigma]$  (asumir que  $\perp_\sigma \stackrel{\text{def}}{=} \mu x : \sigma. x$ ).
- $iterate_\sigma : (\sigma \rightarrow \sigma) \rightarrow \sigma \rightarrow [\sigma]$  que dadas  $f$  y  $x$  genera la lista infinita  $x :: f\ x :: f(f\ x) :: f(f(f\ x)) :: \dots$
- $zip_{\rho, \sigma} : [\rho] \rightarrow [\sigma] \rightarrow [\rho \times \sigma]$  que se comporta como la función homónima de Haskell.
- $take_\sigma : \text{Nat} \rightarrow ([\sigma] \rightarrow [\sigma])$  que se comporta como la función homónima de Haskell.

### 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:

$$\sigma ::= \dots \mid Cola_\sigma \quad M ::= \dots \mid \langle \rangle_\sigma \mid M \bullet M \mid \text{próximo}(M) \mid \text{desencolar}(M) \mid \text{case } M \text{ of } \langle \rangle \rightsquigarrow M; c \bullet x \rightsquigarrow M$$

donde  $\langle \rangle_\sigma$  es la cola vacía en la que se pueden encolar elementos de tipo  $\sigma$ ;  $M_1 \bullet M_2$  representa el agregado del elemento  $M_2$  al **final** de la cola  $M_1$ ; los observadores  $\text{próximo}(M_1)$  y  $\text{desencolar}(M_1)$  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  $\text{case } M_1 \text{ of } \langle \rangle \rightsquigarrow M_2; c \bullet x \rightsquigarrow M_3$  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).

- Introducir las reglas de tipado para la extensión propuesta.
- Definir el conjunto de valores y las nuevas reglas de reducción. Pueden usar los conectivos booleanos de la guía. **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.

- Mostrar paso por paso cómo reduce la expresión:  $\text{case } \langle \rangle_{\text{Nat}} \bullet \underline{1} \bullet 0 \text{ of } \langle \rangle \rightsquigarrow \text{próximo}(\langle \rangle_{\text{Bool}}); c \bullet x \rightsquigarrow \text{isZero}(x)$
- Definir como macro la función  $\text{último}_\tau$ , 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).