Práctica N° 4 - Cálculo-λ: 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 λ con los tipos Bool, Nat y funciones.

Notación para este segmento de la materia:

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

Gramáticas a tener en cuenta:

- Términos
  $M ::= x \mid \lambda x\colon \sigma.\ M \mid M\ M \mid \mathsf{true} \mid \mathsf{false} \mid \mathsf{if}\ M\ \mathsf{then}\ M\ \mathsf{else}\ M \mid 0 \mid \mathsf{succ}(M) \mid \mathsf{pred}(M) \mid \mathsf{isZero}(M)$

  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
  $\sigma ::= \mathsf{Bool} \mid \mathsf{Nat} \mid \sigma \to \sigma$

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) $\lambda x\colon \mathsf{Bool}.\ \mathsf{if}\ 0\ \mathsf{then\ true\ else}\ 0\ \mathsf{succ(true)}$

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 $\lambda x\colon \mathsf{Nat}.\ \mathsf{succ}((\lambda x\colon \mathsf{Nat}.\ x)\ x)$.

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 $u\ x\ (y\ z)$?

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?
$(\lambda x\colon \mathsf{Bool} \to \mathsf{Nat} \to \mathsf{Bool}.\ \lambda y\colon \mathsf{Bool} \to \mathsf{Nat}.\ \lambda z\colon \mathsf{Bool}.\ x\ z\ (y\ z))\ u$

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) $\vdash \mathsf{if\ true\ then}\ 0\ \mathsf{else\ succ}(0)\ :\ \mathsf{Nat}$

b) $x : \mathsf{Nat}, y : \mathsf{Bool} \vdash \mathsf{if\ true\ then\ false\ else}\ (\lambda z\colon \mathsf{Bool}.\ z)\ \mathsf{true}\ :\ \mathsf{Bool}$

c) $\vdash \mathsf{if}\ \lambda x\colon \mathsf{Bool}.\ x\ \mathsf{then}\ 0\ \mathsf{else\ succ}(0)\ :\ \mathsf{Nat}$

d) $x : \mathsf{Bool} \to \mathsf{Nat}, y : \mathsf{Bool} \vdash x\ y\ :\ \mathsf{Nat}$

Ejercicio 7 ★

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

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

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

Ejercicio 8

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

a) $\vdash \mathsf{succ}(0)\ :\ \sigma$

b) $\vdash \mathsf{isZero}(\mathsf{succ}(0))\ :\ \sigma$

c) $\vdash \mathsf{if\ (if\ true\ then\ false\ else\ false)\ then}\ 0\ \mathsf{else\ succ}(0)\ :\ \sigma$

Ejercicio 9 *(Tipos habitados)* ★

Decimos que un tipo σ está *habitado* si existe un término M tal que el juicio $\vdash M : \sigma$ es derivable. En ese caso, decimos que M es un *habitante* de σ. Por ejemplo, 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) $(\sigma \to \tau \to \rho) \to \tau \to \sigma \to \rho$

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

Preguntas para pensar: ¿Con qué función ya conocida de Haskell se corresponde cada uno de los habitantes? ¿Hay tipos que no estén habitados? ¿Si se reemplaza $\to$ por $\Rightarrow$, las fórmulas habitadas son siempre tautologías? ¿Las tautologías son siempre fórmulas habitadas?

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

b) $\vdash (\lambda x\colon \sigma.\ x)(\lambda y\colon \mathsf{Bool}.\ 0) \ : \ \sigma$

c) $y\colon \tau \vdash \mathsf{if}\ (\lambda x\colon \sigma.\ x)\ \mathsf{then}\ y\ \mathsf{else}\ \mathsf{succ}(0) \ : \ \sigma$

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

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

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

g) $x\colon \sigma \vdash x\ \mathsf{true} \ : \ \sigma$

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

Ejercicio 11 *(Debilitamiento y fortalecimiento)*

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

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

Ejercicio 12 *(Regla de sustitución ★)*

Demostrar que si valen $\Gamma, x : \sigma \vdash M \ : \ \tau$ y $\Gamma \vdash N \ : \ \sigma$ entonces vale $\Gamma \vdash M\boldsymbol{\{x := \mathrm{N}\}} \ : \ \tau$.
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) $(\lambda y\colon \sigma.\ x\ (\lambda x\colon \tau.\ x))\boldsymbol{\{x := (\lambda y\colon \rho.\ x\ y)\}}$

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

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-λ.

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

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

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:

$$V := \lambda x\colon \sigma.\ M \mid \mathsf{true} \mid \mathsf{false} \mid 0 \mid \mathsf{succ}(V)$$

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) $\mathsf{succ}(\mathsf{succ}(0))$

Ejercicio 16 *(Programa, Forma Normal)* ★

Para el siguiente ejercicio, considerar el cálculo **sin** la regla $\mathsf{pred}(0) \to 0$

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 $(\lambda x\colon \mathsf{Bool}.\ x)\ \mathsf{true}$

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 $(\lambda f\colon \mathsf{Nat} \to \mathsf{Bool}.\ f\ 0)\ (\lambda x\colon \mathsf{Nat}.\ \mathsf{isZero}(x))$

VI $(\lambda f\colon \mathsf{Nat} \to \mathsf{Bool}.\ x)\ (\lambda x\colon \mathsf{Nat}.\ \mathsf{isZero}(x))$

VII $(\lambda f\colon \mathsf{Nat} \to \mathsf{Bool}.\ f\ \mathsf{pred}(0))\ (\lambda x\colon \mathsf{Nat}.\ \mathsf{isZero}(x))$

VIII $\mathsf{fix}\ (\lambda y\colon \mathsf{Nat}.\ \mathsf{succ}(y))$

Ejercicio 17 *(Determinismo)*

a) ¿Es cierto que la relación definida $\to$ es determinística (o 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 $M' = M''$?

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).

Ejercicio 19

Al agregar la siguiente regla para las abstracciones:

$$\textbf{Si } M \to M',\ \textbf{entonces: } \lambda x\colon \tau.\ M \to \lambda x\colon \tau.\ M' \qquad (\ \xi\ )$$

a) Repensar el conjunto de valores para respetar esta modificación, pensar por ejemplo si $(\lambda x\colon \mathsf{Bool}.\ (\lambda y\colon \mathsf{Bool}.\ y)\ \mathsf{true})$ 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
$(\lambda x\colon \mathsf{Nat} \to \mathsf{Nat}.\ x\ \underline{23})\ (\lambda x\colon \mathsf{Nat}.\ 0)$
¿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 siguiente reglas de tipado y semántica:

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

Si $M \to M'$, **entonces:** $M + N \to M' + N$ $(+_{c1})$
Si $N \to N'$, **entonces:** $V + N \to V + N'$ $(+_{c2})$
$V + 0 \to V$ $(+_0)$
$V_1 + \mathsf{succ}(V_2) \to \mathsf{succ}(V_1) + V_2$ $(+_{\mathsf{succ}})$

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:

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

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, 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 $(\sigma \to \tau \to \rho) \to ((\sigma \times \tau) \to \rho)$.

c) Definir reglas de semántica operacional manteniendo el determinismo y la preservación de tipos. **Importante:** no olvidar las reglas de congruencia.

d) Demostrar que la relación de reducción definida es determinística.

e) ¿Cómo se extiende el conjunto de los valores? ¿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-λ tipado con *uniones disjuntas*. Las gramáticas de los tipos y los términos se extienden de la siguiente manera:

$$\begin{array}{lll} \sigma & ::= & \ldots \mid \sigma + \sigma \\ M & ::= & \ldots \mid \mathsf{left}(M) \mid \mathsf{right}(M) \mid \mathsf{case}\, M \,\mathsf{of}\, \mathsf{left}(x) \leadsto M_1 \,[\!]\, \mathsf{right}(y) \leadsto M_2 \end{array}$$

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

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

b) Usando las reglas de tipado anteriores, 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) Definir reglas de semántica operacional manteniendo el determinismo y la preservación de tipos.

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

e) ¿Cómo se extiende el conjunto de los valores? ¿Se verifica la propiedad de progreso?

Ejercicio 22 ★

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

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

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

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

donde

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

- $\texttt{case}\ 0 :: \mathsf{succ}(0) :: [\,]_{\mathsf{Nat}}\ \ \texttt{of}\ \ \{[\,] \leadsto \mathsf{false} \mid x :: xs \leadsto \mathsf{isZero}(x)\} \twoheadrightarrow \mathsf{true}$
- $\texttt{foldr}\, \underline{1} :: \underline{2} :: \underline{3} :: (\lambda x \colon [\mathsf{Nat}].\ x)\ [\,]_{\mathsf{Nat}}\ \text{base} \leadsto 0;\ \text{rec}(head, rec) \leadsto head + rec \twoheadrightarrow \underline{6}$

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).

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

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.

Ejercicio 23 ★

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

[1] Notemos que si escribimos $\sigma \to \tau$ como τ^σ, 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. $And\ M\ N \twoheadrightarrow \mathsf{true} \Leftrightarrow M \twoheadrightarrow \mathsf{true}$ 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.

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 σ en cualquier contexto.

a) $head_\sigma : [\sigma] \to \sigma$ y $tail_\sigma : [\sigma] \to [\sigma]$ (asumir que $\perp_\sigma \stackrel{\text{def}}{=} \mathit{fix}\ \lambda x : \sigma.x$).

b) $iterate_\sigma{:}(\sigma \to \sigma) \to \sigma \to [\sigma]$ que dadas f y x genera la lista infinita $x :: f\ x :: f(f\ x) :: f(f(f\ x)) :: \ldots$

c) $zip_{\rho,\sigma} : [\rho] \to [\sigma] \to [\rho \times \sigma]$ que se comporta como la función homónima de Haskell.

d) $take_\sigma : \mathsf{Nat} \to ([\sigma] \to [\sigma])$ que se comporta como la función homónima de Haskell.