Clase Práctica 6 : Cálculo λ

Tomás Felipe Melli

July 10, 2025

Índice

1 Primera extensión : `pares`

Sintaxis: tipos y términos

Nos dan la siguiente sintaxis con sus tipos y términos:

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

Y nos piden definir como macro la función $\texttt{curry}_{\sigma,\tau,\psi}$ que sirve para currificar funciones que reciben pares como argumentos.

$$curry_{\sigma,\tau,\psi} \stackrel{def}{=} \lambda f : \sigma \times \tau \to \psi.\lambda x : \sigma.\lambda y : \tau.\ f\langle x, y\rangle$$

Reglas de tipado

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

$$\frac{\Gamma \vdash M : \sigma \times \tau}{\Gamma \vdash \pi_1(M) : \sigma}\ \text{T-}\pi_1$$

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

Y por tanto, extender el conjunto de valores y determinar las nuevas reglas de semántica. Las reglas de semántica son : o de **congruencia** o de **cómputo**. Las de congruencia sirven para decir qué parte de un término reduce primero cuando tenemos varias cosas que podrían reducirse. O sea, **qué reduce primero**. Por ello, estas tienen como premisa que un subtérmino reduce y nos dicen cómo meter la reducción dentro del término más grande. Las de cómputo nos dicen **a qué reduce**

Extendiendo valores

$$V ::= \ldots \mid \langle V, V \rangle$$

Reglas de semántica

$$\frac{}{\pi_1(\langle V_1, V_2 \rangle) \to V_1}\ \text{E-}\pi_1 - 1 \qquad \texttt{cómputo}$$

$$\frac{M \to M'}{\pi_1(M) \to \pi_1(M')}\ \text{E-}\pi_1 - 2 \qquad \texttt{congruencia}$$

$$\frac{}{\pi_2(\langle V_1, V_2 \rangle) \to V_2}\ \text{E-}\pi_2 - 1 \qquad \texttt{cómputo}$$

$$\frac{M \to M'}{\pi_2(M) \to \pi_2(M')}\ \text{E-}\pi_2 - 2 \qquad \texttt{congruencia}$$

$$\frac{M \to M'}{\langle M, N \rangle \to \langle M', N \rangle}\ \text{E-Par1} \qquad \texttt{congruencia}$$

$$\frac{N \to N'}{\langle V, N \rangle \to \langle V, N' \rangle}\ \text{E-Par2} \qquad \texttt{congruencia}$$

Qué problema introduce la siguiente regla ?

$$\pi_1(\langle M, N \rangle) \to M$$

El problema que introduce es que podemos aplicar o esa regla o la que reduce $M \to M'$ nos queda $\pi_1(\langle M', N \rangle)$ con lo cuál estaríamos rompiendo el determinismo. Como se ve en este ejemplo

$$\begin{tikzcd} \pi_1(\langle (\lambda x : Bool.\, x)\ true,\ false \rangle) \arrow[r, "MAL"] \arrow[dr] & (\lambda x : Bool.\, x)\ true \\ & \pi_1(\langle true,\ false \rangle) \end{tikzcd}$$

Nos piden a continuación verificar el siguiente juicio de tipado :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x:Nat \vdash x:Nat}\text{T-VAR} \qquad \dfrac{}{x:Nat\vdash true:Bool}\text{T-TRUE}}{x:Nat \vdash \langle x,true\rangle : Nat\times Bool}\text{T-PAR}}{\emptyset\vdash \lambda x:Nat.\langle x,true\rangle : Nat\to Nat\times Bool}\text{T-ABS} \qquad \dfrac{}{\emptyset\vdash 0:Nat}\text{T-ZERO}}{\emptyset \vdash (\lambda x:Nat.\langle x,true\rangle)\;0 : Nat\times Bool}\text{T-APP}}{\emptyset\vdash \pi_1((\lambda x:Nat.\langle x,True\rangle)\;0):Nat}\text{T-}\pi_1$$

Y reducir

$$\pi_1((\lambda x:Nat.\langle x,True\rangle)\;0) \xrightarrow[\text{E-}\pi_1,\ \beta]{} \pi_1(\langle 0,true\rangle) \xrightarrow[\text{E-}\pi_1\text{PAR}]{} 0$$

$$\pi_1(\pi_1(\lambda x:Nat,\langle\langle x,x\rangle,true\rangle)\;0) \xrightarrow[\text{E-}\pi_1,\ \text{E-}\pi_1,\ \beta]{} \pi_1(\pi_1(\langle\langle 0,0\rangle,true\rangle))$$

2 Segunda extensión : `uniones disjuntas`

Sintaxis: tipos y términos

$$\sigma ::= \ldots \mid \sigma + \sigma$$
$$M ::= \ldots \mid left_\sigma(M) \mid rigth_\sigma(M) \mid \texttt{case M of left(x)} \rightsquigarrow \texttt{M ; right(y)} \rightsquigarrow \texttt{M}$$

Donde $\sigma + \tau$ representa el tipo de la unión disjunta entre sigma y tau, similar al tipo `Either` σ τ en Haskell. Donde $left_\sigma(M)$ y $rigth_\sigma(M)$ inyectan un valor en la unión. Donde `case M of left(x)` $\rightsquigarrow$ `N ; right(y)` $\rightsquigarrow$ `O` efectúa un análisis de casos del término M comparándolo con los patrones de `left` y `right`.
En esta línea nos piden marcar subtérminos y anotaciones de tipos,

$$\sigma ::= \ldots \mid \sigma + \sigma$$
$$M ::= \ldots \mid left_\sigma(M) \mid rigth_\sigma(M) \mid \texttt{case M of left(x)} \rightsquigarrow \texttt{M ; right(y)} \rightsquigarrow \texttt{M}$$

Y enunciar las nuevas reglas de tipado y extender el conjunto de valores y las reglas de semántica de la nueva extensión.

Reglas de tipado

$$\frac{\Gamma \vdash M : \sigma}{\Gamma \vdash left_\sigma(M) : \tau + \sigma}\text{T-Left}$$

$$\frac{\Gamma \vdash M : \tau}{\Gamma \vdash left_\sigma(M) : \tau + \sigma}\text{T-Right}$$

$$\frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x : \sigma.M : \sigma \to \tau}\text{T-Abs}$$

$$\frac{\Gamma \vdash M : \sigma + \tau \qquad \Gamma, x : \sigma \vdash N : \rho \qquad \Gamma, y : \tau \vdash O : \rho}{\Gamma \vdash \texttt{case M of left(x)} \rightsquigarrow \texttt{N ; right(y)} \rightsquigarrow \texttt{O} : \rho}\text{T-Case}$$

Extensión de valores

$$V ::= \dots \mid left_\sigma(V) \mid right_\sigma(V)$$

Reglas de semántica

$$\frac{M \to M'}{left_\sigma(M) \to left_\sigma(M')}\text{ E-Left}$$

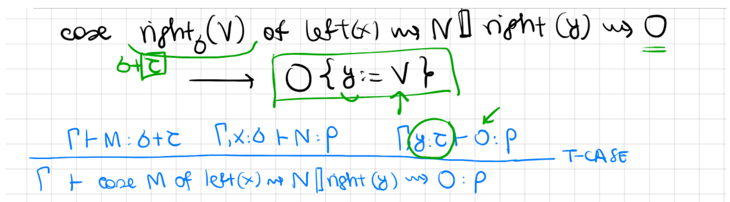
$$\frac{M \to M'}{right_\sigma(M) \to right_\sigma(M')}\text{ E-Right}$$

$$\frac{M \to M'}{\texttt{case M of left(x)} \rightsquigarrow \texttt{N ; right(y)} \rightsquigarrow \texttt{O} \to \texttt{case M' of left(x)} \rightsquigarrow \texttt{N ; right(y)} \rightsquigarrow \texttt{O}}\text{ E-Case}$$

$$\frac{}{\texttt{case } left_\sigma(V) \texttt{ of left(x)} \rightsquigarrow \texttt{N ; right(y)} \rightsquigarrow \texttt{O} \to \texttt{N} \{ \texttt{ x := V}\}}\text{ E-CaseLeft}$$

$$\frac{}{\texttt{case } right_\sigma(V) \texttt{ of left(x)} \rightsquigarrow \texttt{N ; right(y)} \rightsquigarrow \texttt{O} \to \texttt{O} \{ \texttt{ y := V}\}}\text{ E-CaseRight}$$

Detalle importante de por qué tenemos en la regla de tipado el $y : \tau$ a la derecha de todo :



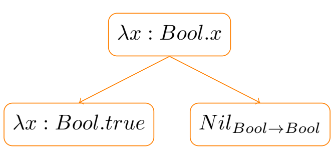
3 Tercera extensión : `árboles binarios`

3.1 `árbol binario`

Sintaxis: tipos y términos

$$\sigma ::= \dots \mid AB_\sigma$$
$$M, N, O ::= \dots \mid Nil_\sigma \mid Bin(M, N, O) \mid raiz(M) \mid der(M) \mid izq(M) \mid esNil(M)$$

Árbol de funciones



Este árbol sería

$\texttt{Bin(Bin(Nil}_{Bool\to Bool}\texttt{,}\lambda x : Bool.true\texttt{, Nil}_{Bool\to Bool}\texttt{), }\lambda x : Bool.x\texttt{, Nil}_{Bool\to Bool}\texttt{)}$

Reglas de tipado

$$\frac{}{\Gamma \vdash Nil_\sigma : AB_\sigma}\text{ T-Nil}$$

$$\frac{\Gamma \vdash M : AB_\sigma \qquad \Gamma \vdash N : \sigma \qquad \Gamma \vdash O : AB_\sigma}{\Gamma \vdash Bin(M, N, O) : AB_\sigma}\text{ T-Bin}$$

$$\frac{\Gamma \vdash M : AB_\sigma}{\Gamma \vdash raiz(M) : AB_\sigma}\text{ T-Raiz}$$

$$\frac{\Gamma \vdash M : AB_\sigma}{\Gamma \vdash esNil(M) : AB_\sigma}\text{ T-EsNil}$$

$$\frac{\Gamma \vdash M : AB_\sigma}{\Gamma \vdash izq(M) : AB_\sigma}\text{ T-Izq}$$

$$\frac{\Gamma \vdash M : AB_\sigma}{\Gamma \vdash izq(M) : AB_\sigma}\text{ T-Der}$$

Extensión de valores

$V ::= \ldots \mid Nil_\sigma \mid Bin(V,V,V)$

Reglas de semántica

$$\frac{M \to M'}{Bin(M,N,O) \to Bin(M',N,O)} \text{ E-Bin1}$$

$$\frac{N \to N'}{Bin(V,N,O) \to Bin(V,N',O)} \text{ E-Bin2}$$

$$\frac{O \to O'}{Bin(V_1,V_2,O) \to Bin(V_1,V_2,O')} \text{ E-Bin3}$$

$$\frac{M \to M'}{raiz(M) \to raiz(M')} \text{ E-Raiz}$$

$$\frac{M \to M'}{raiz(Bin(V_1,V_2,V_3)) \to V_2} \text{ E-RaizBin}$$

No definimos reglas para $\texttt{raiz(Nil}_\sigma\texttt{)}$, $\texttt{izq(Nil}_\sigma\texttt{)}$, $\texttt{der(Nil}_\sigma\texttt{)}$ ya que perdemos progreso.

$$\frac{M \to M'}{izq(M) \to izq(M')} \text{ E-Izq}$$

$$\frac{M \to M'}{der(M) \to der(M')} \text{ E-Der}$$

$$\frac{\,'\,}{izq(Bin(V_1,V_2,V_3)) \to V_1} \text{ E-IzqBin}$$

$$\frac{}{der(Bin(V_1,V_2,V_3)) \to V_3} \text{ E-DerBin}$$

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

$$\frac{}{esNil(Bin(V_1,V_2,V_3)) \to false} \text{ E-esNilBin}$$

$$\frac{}{esNil(Nil_\sigma) \to true} \text{ E-esNilNil}$$

3.2 `árboles bis`

Sintaxis : tipos y términos

$$\sigma ::= \ldots \mid AB_\sigma$$
$$M,N,O ::= \ldots \mid Nil_\sigma \mid Bin(M,N,O) \mid \texttt{case M of Nil} \rightsquigarrow \texttt{N ; Bin(i,r,d)} \rightsquigarrow \texttt{O}$$

Reglas de tipado

$$\frac{}{\Gamma \vdash Nil_\sigma : AB_\sigma} \text{ T-Nil}$$

$$\frac{\Gamma \vdash M : AB_\sigma \qquad \Gamma \vdash N : \sigma \qquad \Gamma \vdash O : AB_\sigma}{\Gamma \vdash Bin(M,N,O) : AB_\sigma} \text{ T-Bin}$$

$$\frac{\Gamma \vdash M : AB_\tau \qquad \Gamma \vdash N : \sigma \qquad \Gamma, i : AB_\tau, r : \tau, d : AB_\tau \vdash O : \sigma}{\Gamma \vdash \texttt{case M of Nil} \rightsquigarrow \texttt{N ; Bin(i,r,d)} \rightsquigarrow \texttt{O} : \sigma} \text{ T-CaseAB}$$

Extensión de valores

$V ::= \ldots \mid Nil_\sigma \mid Bin(V,V,V)$

Reglas de semántica

$$\frac{M \to M'}{Bin(M,N,O) \to Bin(M',N,O)} \text{ E-Bin1}$$

$$\frac{N \to N'}{Bin(V,N,O) \to Bin(V,N',O)} \text{ E-Bin2}$$

$$\frac{O \to O'}{Bin(V_1,V_2,O) \to Bin(V_1,V_2,O')} \text{ E-Bin3}$$

$$\frac{M \to M'}{\texttt{case M of Nil} \leadsto \texttt{N ; Bin(i,r,d)} \leadsto \texttt{O} \to \texttt{case M' of Nil} \leadsto \texttt{N ; Bin(i,r,d)} \leadsto \texttt{O}} \text{ E-CaseAB}$$

$$\frac{}{\texttt{case Nil}_\sigma \texttt{ of Nil} \leadsto \texttt{N ; Bin(i,r,d)} \leadsto \texttt{O} \to \texttt{N}} \text{ E-CaseNil}$$

$$\frac{}{\texttt{case Bin(V}_1\texttt{,V}_2\texttt{,V}_3\texttt{) of Nil} \leadsto \texttt{N ; Bin(i,r,d)} \leadsto \texttt{O} \to \texttt{O}\{\texttt{i := V}_1\texttt{, r := V}_2\texttt{, d := V}_3\}} \text{ E-CaseBin}$$

Nos piden reducir el siguiente término

`case if`$(\lambda x : Bool.x)$ `True then Bin(Nil`$_{Nat}$`,` $\underline{1}$`, Nil`$_{Nat}$`) else Nil`$_{Nat}$ `of Nil` $\leadsto$ `False ; Bin(i,r,d)` $\leadsto$ `isZero(r)`

Recordemos las siguiente reglas de cómputo :

$$\frac{}{\texttt{if true then N else O} \to \texttt{N}} \text{ E-IfTrue}$$

$$\frac{}{\texttt{if false then N else O} \to \texttt{O}} \text{ E-IfFalse}$$

y de congruencia :

$$\frac{M \to M'}{\texttt{if M then N else O} \to \texttt{if M' then N else O}} \text{ E-If}$$

Con esto, ...

$\xrightarrow{E-CaseAB,\ E-If,\ \beta}$ `case if True then Bin(Nil`$_{Nat}$`,` $\underline{1}$`, Nil`$_{Nat}$`) else Nil`$_{Nat}$ `of Nil` $\leadsto$ `False ; Bin(i,r,d)` $\leadsto$ `isZer`

$\xrightarrow{E-CaseAB,\ E-IfTrue}$ `case Bin(Nil`$_{Nat}$`,` $\underline{1}$`, Nil`$_{Nat}$`) of Nil` $\leadsto$ `False ; Bin(i,r,d)` $\leadsto$ `isZero(r)`

$\xrightarrow{E-CaseBin}$ `isZero(r)` $\{$ `i := Nil`$_{Nat}$ `, r :=` $\underline{1}$ `, d := Nil`$_{Nat}\}$ `= isZero(`$\underline{1}$`) = isZero(succ(zero))`

$\xrightarrow{E-isZero_n}$ `false`

Definir macros para raiz, der, izq y esNil

`esNil`$_\sigma$ `=`λ`x : AB`$_\sigma$ `. case x of Nil` $\leadsto$ `true ; Bin(i,r,d)` $\leadsto$ `false`

`raiz`$_\sigma$ `=`λ`x : AB`$_\sigma$ `. case x of Nil` $\leadsto \bot_\sigma$ `; Bin(i,r,d)` $\leadsto$ `r` `usa fix (punto fijo)`

`izq`$_\sigma$ `=`λ`x : AB`$_\sigma$ `. case x of Nil` $\leadsto \bot_{AB_\sigma}$ `; Bin(i,r,d)` $\leadsto$ `i`

Definir una nueva extensión que incorpore expresiones de la forma `map(M,N)`, donde N es un árbol y M una función que se aplicará a cada elemento de N

Extendemos términos

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

Reglas de tipado

$$\frac{\Gamma \vdash M : \sigma \to \rho \qquad \Gamma \vdash N : AB_\sigma}{\Gamma \vdash map(M,N) : AB_\rho} \text{ T-Map}$$

Extendemos valores

$V ::= \ldots \mid$ `no se agregan valores`

Reglas de semántica

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

$$\frac{N \to N'}{\texttt{map(V,N)} \to \texttt{map(V,N')}}\ \text{E-Map2}$$

$$\frac{}{\texttt{map(V,Bin(V}_1\texttt{,V}_2\texttt{,V}_3\texttt{))} \to \texttt{Bin(map(V,V}_1\texttt{),V V}_2\texttt{, map(V,V}_3\texttt{))}}\ \text{E-MapBin}$$

$$\frac{\vdash V : \tau \to \rho}{\texttt{map(V,Nil}_\tau\texttt{)} \to \texttt{Nil}_\rho}\ \text{E-MapNil}$$

Como comentario :

$\texttt{map(}\lambda x\ \texttt{: Bool. zero, Nil}_{Bool}\texttt{)} \overset{E-MapNil}{\to} Nil_{Nat}$
$\vdash \lambda x : Bool.zero : Bool \to Nat$