Lógica Y Programación

Problemas de λ-Calculus y Lógica Combinatoria

UNIVERSIDAD DE GRANADA

Universidad de Granada

Lógica Y Programación

Curso

5to curso

Docente

Francisco Miguel García Olmedo

Autores

Alejandro Egea López
Nicolás Ramírez Rodiles

A fecha de
1 de diciembre de 2025

Índice

Ejercicio 1.

Exponga y desarrolle justificadamente el tema de la "Notación de de Bruijn".

Solución.

Introducción

La notación tradicional del cálculo λ usa nombres para las variables ligadas ($x, y, z, \dots$). Esta práctica introduce tres dificultades fundamentales:

1. **Captura accidental**: al sustituir una expresión por una variable libre, puede producirse captura si surgen colisiones de nombre con variables ligadas en el contexto.
2. **Conversión-α obligatoria**: para evitar capturas debe renombrarse sistemáticamente las variables ligadas.
3. **Complejidad en meta-demostraciones**: propiedades como la confluencia o la prueba de Church-Rosser se complican debido a la constante necesidad de renombrar.

La propuesta de N. G. de Bruijn consiste en *eliminar los nombres de variables ligadas* y reemplazarlos por **índices naturales** que indican cuántos ligadores separan la ocurrencia de su λ correspondiente. De este modo:

- desaparece por completo la conversión-α,
- no hay colisiones de nombres,
- las definiciones de sustitución, reducción β y η se vuelven puramente estructurales,
- las demostraciones metateóricas se simplifican sustancialmente.

El propio de Bruijn establece tres criterios para evaluar una notación:

(i) legibilidad humana,

(ii) claridad en discusión metalingüística,

(iii) utilidad para implementaciones automáticas.

La notación de índices sacrifica parcialmente el punto (i), pero destaca en (ii) y (iii).

Name-carrying expressions y árbol sintáctico

Antes de eliminar nombres, representamos aplicaciones mediante un símbolo especial:

$$A((M), (N)),$$

que corresponde a la aplicación usual MN. Esto permite un análisis uniforme de la estructura.

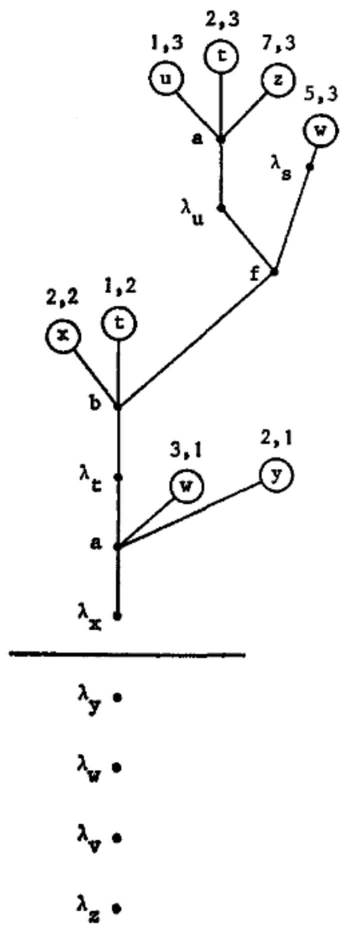
Consideremos ahora la siguiente expresión λ-cálculo (tomada del ejemplo del artículo):

$$\boxed{\lambda x\, a(\lambda b(x, t, f(\lambda u\, a(u, t, z),\ \lambda s\, w)), w, y)}$$

En forma estructurada "name-carrying", cada rama del árbol lleva etiquetas a, b, f según la construcción.

El artículo presenta el siguiente **árbol anotado** con *reference depth* y *level*. Los pares (d, ℓ) representan:

- $d =$ *reference depth*: número de ligadores λ desde la ocurrencia hasta su ligador,
- $\ell =$ *level*: número total de λ hasta la raíz.



Lectura precisa del árbol

Tomemos algunos nodos para ejemplificar:

- El nodo x tiene etiqueta $(2, 2)$:
  - reference depth $= 2$: está bajo dos λ antes de llegar a su ligador original λ_x,
  - level $= 2$: desde la ocurrencia hasta la raíz hay dos λ.
- El nodo t bajo $b(x, t)$ tiene $(1, 2)$:
  - depth = 1: está inmediatamente bajo λ_t,
  - level = 2: hay dos ligadores superiores en total.
- El nodo v etiquetado $(3, 1)$ indica que:
  - depth = 3: su ligador está tres λ más arriba,
  - level = 1: solo un λ separa esa rama de la raíz.

- Lo mismo ocurre para u, t, z, w en la rama derecha bajo f.

Este árbol es la base para convertir la expresión a notación sin nombres.

De nombres a índices: notación de de Bruijn

Para cada ocurrencia ligada, sustituimos su nombre por su *reference depth*.

- Las variables libres se conservan en una lista ordenada $(x_1, x_2, \dots)$,
- Los *level* sirven como información auxiliar para verificar la corrección formal, pero se omiten en la expresión final.

Ejemplos simples

$$\lambda x.\, x \;\longmapsto\; \lambda.\, 1$$

$$\lambda x.\, \lambda y.\, (x\, y) \;\longmapsto\; \lambda.\, \lambda.\, (2\, 1)$$

$$\lambda x.\, \lambda y.\, \lambda z.\, x \;\longmapsto\; \lambda.\, \lambda.\, \lambda.\, 3$$

Sintaxis de expresiones sin nombres

Usamos una gramática claramente indentada:

```
<NF>         ::= <Const>
               | <Index>
               | <NFList>
               | lambda.<NF>

<Const>      ::= a | b | c | ...         ; constantes simbólicas

<Index>      ::= 1 | 2 | 3 | ...         ; variable ligada por índice

<NFList>     ::= A(<NF>, <NF>)           ; aplicación
               | <NF> <NF>               ; concatenación de expresiones
```

Esta sintaxis es extremadamente regular: ya no aparecen nombres ligados, sólo índices.

Sustitución en notación de de Bruijn

La sustitución se define como:

$$S(Z_1, Z_2, \dots;\ Q),$$

donde Z_i sustituye a la variable libre cuyo índice es i.
Casos fundamentales:

- Si Q es una constante, permanece igual.
- Si $Q = k$ es un índice, entonces:
$$S(\dots, Z_k;\ k) = Z_k,$$
con ajuste de índices si la sustitución entra bajo una λ.

- Si $Q = A((Q_1),(Q_2))$, entonces:
$$S(Z;\ A((Q_1),(Q_2))) = A((S(Z;Q_1)),(S(Z;Q_2))).$$
- Si $Q = \lambda.\, R$, entonces:
$$S(Z_1, Z_2, \ldots;\ \lambda.\, R) = \lambda.\, S(Z_1', Z_2', \ldots;\ R),$$
donde cada Z_i' es la versión de Z_i con sus índices incrementados en uno (para preservar referencias correctas).

Reducción β

La regla de contracción β se expresa elegantemente:
$$A((\lambda.\, Q),(r)) \;\longrightarrow\; S(r, 2, 3, \ldots;\ Q),$$
es decir: sustituimos el índice 1 por r.

Ejemplos

$$(\lambda x.\, x)\ a \quad\rightsquigarrow\quad \lambda.\, 1\ a \;\longrightarrow\; a.$$
$$(\lambda x.\, \lambda y.\, x)\ y \quad\rightsquigarrow\quad A((\lambda.\, \lambda.\, 2),(y)) \;\longrightarrow\; \lambda.\, 1.$$
En ningún momento aparece conversión-α.

Reducción η

La regla extensional:
$$\lambda.\, A((Q), 1) \;\longrightarrow\; Q$$
siempre que Q no contenga ninguna referencia al índice 1 ligado por la λ.
Ejemplo:
$$\lambda x.\, f\, x \quad\longmapsto\quad \lambda.\, A((f), 1) \quad\longrightarrow\quad f.$$

Algunos comentarios sobre la reducción múltiple

De Bruijn desarrolla una teoría de *reducción múltiple* β_U, donde un conjunto U de símbolos de aplicación se reduce simultáneamente.

Ideas centrales:

- se define la noción de *expresión U-correcta*,
- la sustitución conserva U-corrección (Teorema 10.1),
- la reducción múltiple y la sustitución poseen reglas claras de conmutación y composición (Teorema 11.1),
- reducciones múltiples para conjuntos distintos U, V conmutan entre sí (Teorema 11.2),
- todo ello conduce a una demostración pulcra del **teorema de Church-Rosser** en la notación sin nombres.

En la notación de índices no existe la conversión-α ni conflictos de nombres, por lo que la demostración se vuelve más transparente: la confluencia se logra por propiedades puramente estructurales.

Conclusión

La notación de de Bruijn resuelve de raíz los problemas de captura y renombrado al sustituir nombres ligados por índices estructurales. La sustitución y la reducción se vuelven definiciones limpias, algebraicas y mecanizables.

La aplicación a la reducción múltiple y a la prueba de Church-Rosser muestra que la notación no sólo simplifica cálculos locales, sino que también clarifica la teoría global del cálculo λ.

Ejercicio 2.

Demuestre que para todo λ-término N, $\lambda x.x\,K\,N \,\#\, \lambda x.x\,S\,N$. (Nota: Recuérdese que $S \equiv \lambda xyz.xz(yz)$ y $K \equiv \lambda xy.x$).

Solución. Supongamos, por reducción al absurdo, que $\lambda x.x\,K\,N = \lambda x.x\,S\,N$ y vamos a llegar a la contradicción de que $K = S$, lo cual es imposible porque $S\#K$ por clase.

En primer lugar, aplicamos la "regla 3" con $Z = K$, de modo que

$$\lambda x.x\,K\,N \;=\; \lambda x.x\,S\,N \Rightarrow (\lambda x.x\,K\,N)K \;=\; (\lambda x.x\,S\,N)K$$

Ahora bien, aplicando "β-conversión" y "sustitución 1" obtenemos que

$$(\lambda x.x\,K\,N)K = KKN \qquad \text{y} \qquad (\lambda x.x\,S\,N)K = KSN$$

tal que, por transitividad, obtenemos

$$KKN = KSN$$

Finalmente, basta con aplicar la propia definición de K, concluyendo por transitividad que

$$KKN = K \qquad \text{y} \qquad KSN = S$$

$$\Rightarrow K = S$$

lo cual es una contradicción.

Ejercicio 3.

Dibuje razonadamente el grafo $G_\beta(WWW)$, donde $W \equiv \lambda xy.xyy$.

Solución. Primero, es interesante recordar la definición del grafo de β-conversión de un λ-término tal que

> El grafo de β-conversión de un λ-término T (o grafo de reducción de un λ-término T), que denotaremos por $G_\beta(T)$, verifica:
>
> 1. Un λ-término M es un nodo de $G_\beta(T)$ si $\lambda \vdash T = M$.
> 2. Si M_1, M_2 son nodos distintos de $G_\beta(T)$, entonces $M_1 \not\equiv M_2$ (en particular, admite $M_1 =_\beta M_2$).
> 3. $n \geq 1$ aristas unen nodo M_1 a nodo M_2 (pudiendo ser $M_1 \equiv M_2$) si, y solo si, $\lambda \vdash M_1 = M_2$

Ejercicio 4.

Encuentre razonadamente un λ-término M tal que $G_\beta(M)$ sea exactamente:

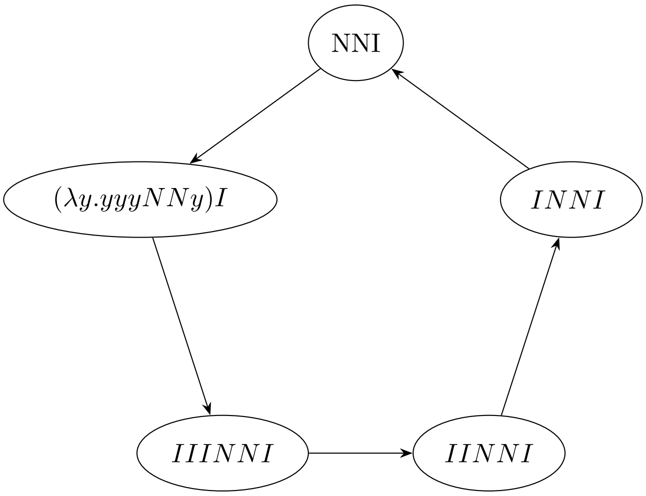
Solución. Sea el candidato $M \equiv NNI$ donde

$$N \equiv \lambda xy.yyyxxy \qquad \text{e} \qquad I \equiv \lambda x.x$$

y veamos que $G_\beta(M)$ es el grafo del enunciado. Sabiendo que $II \equiv (\lambda x.x)I = I$, tenemos que

$$\begin{aligned} NNI &\equiv (\lambda xy.yyyxxy)NI \\ &= (\lambda y.yyyNNy)I \\ &= IIINNI \\ &= IINNI \\ &= INNI \\ &= NNI \qquad \text{(¡hemos cerrado el ciclo!)} \end{aligned}$$

En efecto, tenemos cinco λ-términos no iguales (en el sentido de $\equiv$). Y $G_\beta(M)$ es



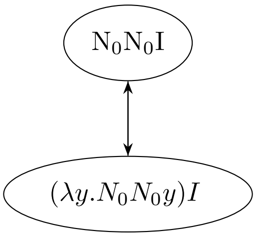
Vamos a explorar un poco más este λ-término que hemos propuesto. Presentamos las siguientes definiciones:

$$N_k = \lambda xy.\underbrace{y\cdots y}_{\text{k veces}}xxy \qquad \text{y} \qquad C_{k+2} = N_k N_k I \qquad \text{para } k \in \mathbb{N}$$

Así pues, estas definiciones nos conducen a plantear el siguiente lema.

> $G_\beta(C_{k+2})$ es un grafo de reducción cíclico de $k+2$ nodos para todo $k \in \mathbb{N}$.

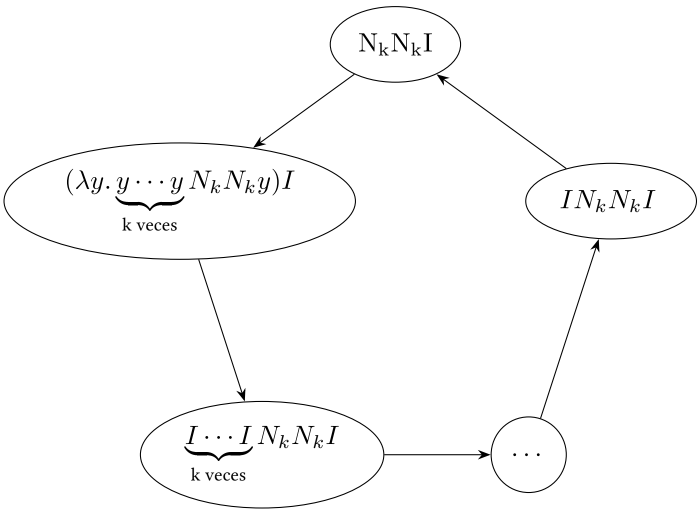
Por una parte, veamos el caso base $k = 0$ tal que $N_0 = \lambda xy.xxy$ y por ende, $C_2 = N_0N_0I$. Tras la primera parte del ejercicio, es inmediato comprobar que $(\lambda y.N_0N_0y)I = N_0N_0I$ se trata de $G_\beta(C_2)$ cíclico de 2 nodos. Cabe destacar que el caso base constituye un caso *degenerado*, puesto que *un ciclo de dos nodos no es más que el segmento que los une.*



Por otra parte, veamos el caso para un $k > 0$ arbitrario tal que $N_k = \lambda xy.\underbrace{y\cdots y}_{\text{k veces}}xxy$, es decir

$$\begin{aligned}
C_{k+2} = N_kN_kI = (\lambda xy.\underbrace{y\cdots y}_{\text{k veces}}xxy)N_kI = & \\
&= (\lambda y.\underbrace{y\cdots y}_{\text{k veces}}N_kN_ky)I \\
&= \underbrace{I\cdots I}_{\text{k veces}}N_kN_kI \\
&= \underbrace{I\cdots I}_{\text{k-1 veces}}N_kN_kI \\
&= \ldots \\
&= IN_kN_kI \\
&= N_kN_kI
\end{aligned}$$

En conclusión, es inmediato observar que hay $k+2$ λ-términos no iguales (en el sentido de $\equiv$) y así podemos hacer que $G_\beta(C_{k+2})$ sea un policiclo de cualquier número **finito** de nodos mayor o igual que 2.



Ejercicio 5.

Sea el λ-término:

$$G \equiv \lambda yx.x(yx) \qquad \text{y} \qquad M \equiv (\lambda xy.y(xxy))(\lambda xy.y(xxy)).$$

1. Demuestre que M es un punto fijo de G.
2. Demuestre que si el combinador N es un punto fijo de G, entonces N es un operador de punto fijo.
3. Demuestre que M es un combinador de punto fijo.
4. Demuestre que si M es un combinador de punto fijo, entonces $M = GM$.

Solución.

(1) Si aplicamos el Teorema del Punto Fijo al λ-término G, sean y tal que $y \notin FV(G)$, $W = \lambda z.G(zz)$ y $X = WW$, entonces obtenemos que $X = M$ es punto fijo de G. En efecto,

$$\begin{aligned}
X &\equiv \\
&\equiv WW \\
&= (\lambda yx.x(yyx))(\lambda yx.x(yyx)) \qquad \text{por } (\dagger) \\
&\equiv (\lambda xy.y(xxy))(\lambda xy.y(xxy)) \qquad \text{por } \alpha\text{-congruencia}
\end{aligned}$$

donde $(\dagger)$ resulta por

$$\begin{aligned}
W &\equiv \\
&\equiv \lambda y.G(yy) \\
&\equiv \lambda y.(\lambda yx.x(yx))(yy) \\
&= \beta\lambda y.((\lambda x.x(yx))[y := yy]) \qquad \text{por } \beta\text{-conversión} \\
&\equiv \lambda y.\lambda x.x(yyx) \qquad \text{por sustitución (1)} \\
&= \lambda yx.x(yyx)
\end{aligned}$$

(2) Supongamos que el combinador N es un punto fijo de G, es decir, $N = GN$. Por esto,

$$\begin{aligned}
N &= \\
&= GN \\
&\equiv (\lambda yx.x(yx))N \\
&= \lambda x.((x(yx))[y := N]) \qquad \text{por } \beta\text{-conversión} \\
&\equiv \lambda x.x(Nx) \qquad \text{por sustitución 1}
\end{aligned}$$

Y concluimos que, dado F un λ-término,

$$\begin{aligned}
NF &= \\
&= (\lambda x.x(Nx))F \\
&= (x(Nx))[x := F] \qquad \text{por } \beta\text{-conversión} \\
&= F(NF) \qquad \text{por sustitución 1}
\end{aligned}$$

que es precisamente lo que se quería demostrar.

(3) Vamos a presentar una forma inmediata de resolver el ejercicio y otra más didáctica.

Por una parte, M es claramente un combinador porque ninguna variable ocurra libre y, además, es punto fijo de G por el apartado (1). Tenemos que verifica las hipótesis del apartado (2) y por tanto, M es un operador (o combinador) de punto fijo.

Por otra parte, si denotamos por $W := \lambda xy.y(xxy)$ tenemos que

$$\begin{aligned} M =& \\ &\equiv WW \\ &\equiv (\lambda xy.y(xxy))W \\ &= \lambda y.((y(xxy))[x := W]) \qquad \text{por } \beta\text{-conversión} \\ &\equiv \lambda y.y(WWy) \qquad \text{por sustitución 1} \\ &\equiv \lambda y.y(My) \qquad \text{denotamos por } (\dagger) \end{aligned}$$

Y llegamos a que, dado F un λ-término,

$$\begin{aligned} MF =& \\ &= (\lambda y.y(My))F \qquad \text{por } (\ddagger) \\ &= (y(My))[y := F] \qquad \text{por } \beta\text{-conversión} \\ &\equiv F(MF) \qquad \text{por sustitución 1} \end{aligned}$$

es decir, que M es un combinador de punto fijo.

(4) Supongamos que N es un combinador de punto fijo, es decir, para todo F λ-término $NF = F(NF)$. Tomando el G del enunciado tenemos que

$$\begin{aligned} GM =& \\ &\equiv (\lambda yx.x(yx))N \\ &= \lambda x.((x(yx))[y := N]) \qquad \text{por } \beta\text{-conversión} \\ &\equiv \lambda x.x(Nx) \qquad \text{por sustitución 1} \end{aligned}$$

Ahora bien, si tomamos $T = GN$ llegamos a que

$$\begin{aligned} TF =& \\ &= GNF \qquad \text{por regla 3} \\ &= (\lambda x.x(Nx))F \\ &= (x(Nx))[x := F] \qquad \text{por } \beta\text{-conversión} \\ &= F(NF) \qquad \text{por sustitución 1} \end{aligned}$$

Finalmente, aplicamos la hipótesis obteniendo que $TF = F(NF) = NF$, y de nuevo, por la regla 3, tenemos que $T = N$.

Como partíamos de que $T = GN$, efectivamente llegamos a que $N = GN$, es decir, N es punto fijo de G.

Ejercicio 6.

Considere el combinador:

$$Y \equiv \lambda y.(\lambda x.y(xx))(\lambda x.y(xx))$$

y demuestre que $GY = Y$.

Solución. Probemos que Y es un combinador de punto fijo para que, aplicando el Ejercicio 5 (d), concluyamos que $Y = GY$. Es decir, debemos probar que para cada F λ-término se tiene que $YF = F(YF)$, que es inmediato por β-conversión tal que

$$\begin{aligned} YF &\equiv \\ &\equiv (\lambda y.(\lambda x.y(xx))(\lambda x.y(xx)))F \\ &= ((\lambda x.y(xx))(\lambda x.y(xx)))[y := F] \qquad \text{por } \beta\text{-conversión} \\ &\equiv (\lambda x.y(xx))[y := F](\lambda x.y(xx))[y := F] \qquad \text{por sustitución 6} \\ &\equiv (\lambda x.F(xx))(\lambda x.F(xx)) \qquad \text{por sustitución 1} \\ &\equiv \omega\omega \end{aligned}$$

Y ahora bastaría continuar reduciendo $YF = \omega\omega$ tal que

$$\begin{aligned} \omega\omega &= \\ &= F(xx)[x := \omega] \qquad \text{por } \beta\text{-conversión} \\ &\equiv F(\omega\omega) \qquad \text{por sustitución 1} \\ &\equiv F(YF) \qquad \text{por el cálculo previo} \end{aligned}$$

Por lo tanto, hemos probado que Y es combinador de punto fijo, con lo cual afirmamos que $Y = GY$.

Ejercicio 7.

Considere la sucesión de combinadores $\{Y^n\}_n$ definida para todo número natural n como sigue:

$$Y^n = \begin{cases} Y, & \text{si } n = 0, \\ Y^{n-1}G, & \text{si } n > 0. \end{cases}$$

Demuestre que para todo $n \geq 0$, Y^n es un combinador de punto fijo.

Solución. Razonamos por inducción sobre $n \geq 0$. Por una parte, el caso base $\boxed{n = 0}$ es $Y^0 = Y$, que es claramente un combinador de punto fijo por el Ejercicio 6.

Por otra parte, veamos el caso inductivo $\boxed{S(k) \Rightarrow S(k+1)\, \forall k > 0}$ donde la hipótesis de inducción es $S(k) =$ «Y^kes un combinador de punto fijo». Queremos demostrar que $S(k+1)$, y procedemos de la siguiente manera

$$Y^{k+1} \overset{def.}{=} Y^k G \overset{H.I.}{=} G(Y^k G) \overset{def.}{=} G(Y^{k+1})$$

En efecto, como $Y^{k+1} = G(Y^{k+1})$, entonces Y^{k+1} es un combinador de punto fijo como se quería demostar, completando la inducción.

Ejercicio 8.

Encuentre razonadamente el CL-término $(\lambda xy.xyy)_{CL}$.

Solución. En primer lugar, escribimos el λ-término como un CL-término de tipo '[x].M' (que usaremos solo permitir la conversión $\lambda \to CL$, pero que no es un CL-término en sí mismo):

$$(\lambda xy.xyy)_{CL} \to [xy].xyy,$$

y, por definición de abstracción de múltiples variables, obtenemos que

$$[xy].xyy \equiv [x]([y].xyy)$$

de modo que ya podemos proceder a aplicar las reglas de abstracción (a), (b), (c), (f) (las que se requieran). A modo de recordatorio:

Definition 2.18 (Abstraction). Para todo término de CL llamémoslo M y toda variable x, se define por inducción un término de CL llamado $[x].M$ del siguiente modo:

(a) $[x].M \equiv KM$ si $x \notin FV(M)$.

(b) $[x].x \equiv I$.

(c) $[x].Ux \equiv U$ si $x \notin FV(U)$.

(f) $[x].UV \equiv S([x].U)([x].V)$ si no se aplica ninguna de las reglas anteriores.

*Cabe destacar que la referencia prescinde de las reglas (d) y (e); y que en [x].UV usando la regla (f), V es siempre la parte derecha de la aplicación más externa del término.

Vamos a operar según las reglas (*las reglas se enuncian en orden de aplicación):

$$
\begin{aligned}
[x].([y].xyy) \equiv & \\
& \equiv [x].(S([y].xy)([y].y)) \qquad \text{por regla (f)} \\
& \equiv [x].(S(S([y].x)([y].y))I) \qquad \text{por regla (f) y (b)} \\
& \equiv [x].(S(S(Kx)(I))I) \qquad \text{por regla (a) y (b)} \\
& \equiv S([x].S(S(Kx)(I)))([x].I) \qquad \text{por regla (f)} \\
& \equiv S(S([x].S)([x].(S(Kx)(I))))KI \qquad \text{por regla (f) y (a)} \\
& \equiv S(S(KS)([x].(S(Kx)(I))))KI \qquad \text{por regla (a) y (f)} \\
& \equiv S(S(KS)(S([x].S(Kx))([x].I)))KI \qquad \text{por regla (f)} \\
& \equiv S(S(KS)(S(S([x].S)([x].Kx))KI))KI \qquad \text{por regla (f) y (a)} \\
& \equiv S(S(KS)(S(S(KS)K)KI))KI \qquad \text{por regla (a) y (c)}
\end{aligned}
$$

Y por último, podemos aplicar las propias definiciones de los combinadores S, K, I para simplificar la expresión tal que

$$
\begin{aligned}
S(S(KS)(S(S(KS)K)KI))KI \equiv & \\
& \equiv \text{FALTA TERMINAR ESTA SIMPLIFACIÓN}
\end{aligned}
$$

Ejercicio 9.

Esquematice la relación entre el sistema λ y la lógica combinatoria.

Solución.

Referencias

[1] De Bruijn, N. G. (1972). *Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church–Rosser theorem. Indagationes Mathematicae (Proceedings), 75*(5), 381–392. https://doi.org/10.1016/1385-7258(72)90034-0

[2] Venturini Zilli, M. (1984). *Reduction graphs in the lambda calculus. Theoretical Computer Science, 29*(3), 251–275. https://doi.org/10.1016/0304-3975(84)90002-1

[3] Hindley, J. R., & Seldin, J. P. (2008). *Lambda-calculus and combinators: An introduction* (2nd ed.). Cambridge University Press. (Secciones relevantes: Section 2C, *Abstraction in CL*, pp. 26–29; Chapter 9, *Correspondence between* λ *and CL*, pp. 92–106.)