Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Tesis de Licenciatura en Ciencias de la Computación

Una curiosa versión de λ_{dB} basada en "swappings": aplicación a traducciones entre cálculos de sustituciones explícitas con nombres e índices

Ariel Mendelzon
amendelzon@dc.uba.ar
L.U. 262/02

Director: Alejandro Ríos

Co-Director: Beta Ziliani

Abril - 2010

Resumen

Desde que Melliès demuestra que el cálculo $\lambda\sigma$ no posee la propiedad de preservación de la normalización fuerte (PSN) [Mel95], numerosos cálculos de sustituciones explícitas han visto la luz; el objetivo: solventar dicho inconveniente sin perder, a la vez, el abanico de propiedades que sí posee $\lambda\sigma$, e intentando ganar – también – confluencia sobre metatérminos.

En [Kes08, Kes09], Delia Kesner presenta λex, formalismo con variables nombradas que posee todas las propiedades esperables de un cálculo de sustituciones explícitas. El objetivo de esta tesis es introducir el cálculo de sustituciones explícitas con índices *à la de Bruijn* λrex, formalismo que resulta ser isomorfo a λex y que, por lo tanto, tiene exactamente las mismas propiedades. A su vez, y a diferencia de otros formalismos existentes, λrex exhibe una notación extremadamente sencilla. Todo esto lo convierte en el primer cálculo con índices, sustituciones unarias y notación muy simple en poseer un conjunto completo de propiedades deseables.

Junto con λrex, se presentan otros dos cálculos indexados e isomorfos a λx [Blo95, Blo97] y a λxgc [BR95]: λre y $\lambda\mathrm{re}_{\mathrm{gc}}$, respectivamente. Hasta donde sabemos, no se conocían al momento tales isomorfismos.

Abstract

Since Melliès showed that $\lambda\sigma$ does not have preservation of strong normalization (PSN) [Mel95], several explicit substitution calculi have been proposed; mainly trying to get PSN without losing other desired properties that $\lambda\sigma$ does have, as well as intending the achievement of confluence on open terms.

In [Kes08, Kes09], Delia Kesner introduces λex, a formalism with variable names that has all the desired properties one should expect from an explicit substitution calculi. The aim of this thesis is to present the explicit substitution calculi with *de Bruijn* indexes λrex, a formalism that turns out to be isomorphic to λex, and that consequently has all of its properties. Moreover, and in contrast to other existent formalisms, λrex has an extremely unsophisticated notation. All of this makes it the first indexed explicit substitution calculi with unary closures and a very simple notation to posess a complete set of desirable properties.

Besides λrex, two other indexed calculi that turn out to be isomorphic to λx [Blo95, Blo97] and λxgc [BR95] are introduced: λre and $\lambda\mathrm{re}_{\mathrm{gc}}$, respectively. As far as we known, no such isomorphisms have been achieved to the date.

A Beny y abuelito José.

Agradecimientos

A todos aquellos que, de una forma u otra, lucharon por una universidad pública, autónoma, gratuita y laica. Y a todos los que, día a día, continúan luchando.

A mi familia, por acompañarme en todo (o casi todo).

A "Lo' Piiibe", por hacer más amena la carrera. Y por muchas cosas más.

A Alejandro Ríos y Beta Ziliani, por la sabiduría, el esfuerzo y la paciencia aportados.

A Delia Kesner, por interesantes discusiones λ-ísticas y una admirable predisposición a responder consultas.

A Eduardo Bonelli, por sus atractivas propuestas de trabajo a futuro.

Al 160, indiscutible musa de la reescritura.

Finalmente, al famoso y nunca bien ponderado λ. Sin éste, esta tesis no hubiera sido posible.

Índice general

Introducción

Este trabajo es acerca de sustituciones explícitas, formalismo que toma fuerza entre los años 1991 y 1995, primero a partir de la introducción del novedoso cálculo $\lambda\sigma$ [ACCL91] y, más tarde, cuando Melliès [Mel95] prueba que éste resulta no poseer una propiedad sumamente importante (preservación de la normalización fuerte, o PSN), lo que genera un aluvión de nuevos cálculos que intentan solventar los problemas que van apareciendo a medida que crece el trabajo en el área.

La motivación principal detrás del campo de las sustituciones explícitas es la de estudiar, a nivel teórico y en profundidad, lo que ocurre cuando la operación de sustitución (uno de los pilares fundamentales del cálculo λ y de muchas áreas de la lógica) se incluye dentro del mismo lenguaje al que sirve, puesto que en el cálculo λ clásico la sustitución es en realidad una *metaoperación* que se encuentra fuera del lenguaje (*i.e.,* en un orden superior) y se utiliza de manera atómica.

Si bien han aparecido numerosos cálculos de sustituciones explícitas desde los años '90, recién en 2007 D. Kesner consigue el primer cálculo que posee toda una batería de propiedades deseables al mismo tiempo: λes [Kes07], simplificándolo más tarde para llegar a λex [Kes08, Kes09]. Ambos cálculos son formalismos con nombres de variables, quedando hasta el momento abierto el problema de conseguir un cálculo con *índices de de Bruijn* [dB72] que posea, al menos, las mismas bondades.

Presentamos, en este trabajo, tal cálculo. En particular, damos un formalismo que resulta ser *isomorfo* a λex y que, por ende, preserva exactamente las mismas propiedades. La idea para la concepción surge a partir de la imposibilidad que tuvimos durante las etapas iniciales del desarrollo de dar una traducción sencilla desde los términos de un cálculo con nombres y sustituciones explícitas unarias a un formalismo similar con índices de *de Bruijn*, tal y como detallaremos en el capítulo correspondiente.

En lo que a trabajos relacionados respecta, es importante destacar que la idea original y el desarrollo de la noción de "swapping" presentada aquí fue independiente de la noción análoga de "exchange" que se utiliza en el cálculo $\lambda_{\emptyset\mathrm{dB}}$ [Arb05]. Este cálculo está basado en el formalismo con nombres de variables y sin sustituciones $\lambda_\emptyset$, de Revesz [Rev85]. La motivación principal detrás de la concepción de $\lambda_{\emptyset\mathrm{dB}}$ fue la de obtener una versión indexada de $\lambda_\emptyset$, llegando a la idea de "exchange" como herramienta para lograrlo. El planteo que nosotros realizamos aquí en cuanto a la introducción de este operador es más profundo: creemos que la noción de "swapping" es una de las claves para obtener formalismos con índices de de Bruijn que se comporten correctamente al tratar la sustitución de manera explícita. Más aún, y desde esta óptica, pensamos que

la versión alternativa de λ_{dB} que mostramos en este trabajo es más "acertada" que la original. En esta dirección, el trabajo que se realiza en $\lambda_{\emptyset\mathrm{dB}}$ nos da una confirmación más de esta afirmación, dada la casual correspondencia entre los operadores introducidos.

Siguiendo con trabajos relacionados podemos mencionar, también, un cálculo de sustituciones explícitas cuyos fundamentos son similares al nuestro, que data de 1978 y que fue propuesto por la misma persona que inventó los índices a los que hacemos alusión: Nicolaas Govert de Bruijn. Existen dos diferencias muy importantes a nivel conceptual entre el cálculo que propone de Bruijn ($\lambda\xi\phi$ [dB78], resumido y modernizado con respecto a notación en [Les94]) y el que nosotros damos aquí. En primer lugar, y como punto más fuerte, el cálculo de de Bruijn no posee composición de sustituciones (característica clave para la obtención de *metaconfluencia*), mientras que el nuestro sí. Además, y como segunda desemejanza importante, los operadores de actualización de índices de $\lambda\xi\phi$ se encuentran en el lenguaje, mientras que en el que nosotros proponemos se ubican en el metalenguaje. Esta última divergencia se debe a que, en pos de conseguir un isomorfismo con λex, decidimos – con éxito – estudiar el cálculo con metaoperadores, sacrificando lo que podría ser una característica deseable – aunque veremos más adelante que quizás para este cálculo no lo sea.

Dadas las diferencias más importantes entre ambos cálculos, es interesante mencionar también que, en cuanto a forma, estos son completamente diferentes (característica que radica en los usos y costumbres notacionales de las épocas en que fueron concebidos). Como último detalle atractivo para el investigador interesado queremos agregar que, por mucho que buscamos, no encontramos un estudio profundo de $\lambda\xi\phi$ en la literatura.

Antes de cerrar esta introducción, comentamos brevemente el contenido de la tesis: en el capítulo 1 presentamos los formalismos que utilizamos durante el trabajo, junto con algunas herramientas aledañas. En los capítulos 2 y 3 presentamos, respectivamente, las ideas iniciales que llevaron a la concepción del cálculo y el cálculo *puro* (*i.e.,* sin sustituciones explícitas). Más adelante, en los capítulos 4 y 5, proponemos y estudiamos dos cálculos de sustituciones explícitas previos al cálculo principal y derivados del presentado en el capítulo 3 que, en particular, resultan muy interesantes por sí solos. Por último, en el capítulo 6, damos el cálculo estrella del trabajo – λrex –, junto con las pruebas del isomorfismo existente entre éste y λex. Además, agregamos un apartado especial dedicado a mostrar metaconfluencia. Cerramos luego con reflexiones sobre lo conseguido y posibles líneas de trabajo a futuro.

Capítulo 1

Presentación de formalismos

1.1 Introducción

En la presente sección introducimos los conceptos y formalismos necesarios para llevar adelante el desarrollo de la tesis. Muchos de éstos no son estrictamente necesarios para el trabajo per se; sin embargo, elegimos colocarlos aquí pues los consideramos muy importantes dentro de la teoría de la reescritura, y pueden ser de mucha utilidad para el lector poco familiarizado con el área.

Creemos importante recalcar que los cálculos que proponemos en este trabajo resultan poseer muchas de las propiedades que presentamos en esta sección. No obstante, no desarrollamos pruebas específicas para estas propiedades sobre nuestros cálculos, pues éstas surgen como corolario directo de varios isomorfismos presentados. De esta manera, y al igual que para aquellos formalismos que presentamos aquí y que son relevantes por sí mismos, basta con presentar de manera relativamente informal los conceptos.

Comenzamos con una introducción básica al concepto de *sistemas de reescritura abstractos*, junto con algunas propiedades deseables de éstos. Continuamos con la presentación del concepto de *sistemas de reescritura de términos* y aledaños. Por último, y aquí haremos especial énfasis – consecuencia del contenido de la tesis –, introducimos el λ-cálculo, junto con sus diferentes variantes a través de la historia del área, y centrándonos en los conceptos introductorios que hacen al objetivo perseguido con este trabajo.

1.2 Convenciones utilizadas

Presentamos aquí las convenciones de índole notacional que utilizaremos a lo largo de todo el trabajo.

Para hablar de números naturales, usaremos $\mathbb{N}$ para los naturales mayores o iguales a cero, y $\mathbb{N}_{>0}$ para los naturales mayores estrictos a cero.

Con respecto a conjuntos, usaremos las clásicas $\bullet \cup \bullet$ y $\bullet \cap \bullet$ para referirnos a las operaciones de unión e intersección, respectivamente. A su vez, la operación $\bullet - \bullet$ denotará la diferencia de conjuntos.

Cuando hagamos referencia a relaciones entre términos de algún cálculo, la igualdad ($\bullet = \bullet$) será *siempre* igualdad sintáctica. Cualquier otro tipo de relación de equivalencia será notada de manera especial (tal y como puede ser el caso de la α-congruencia, que introducimos más adelante).

En lo que a nombramiento de términos, variables y demás respecta, usaremos, generalmente, las letras $x, y, z, \ldots$ para referirnos a variables nombradas; $n, m, o, \ldots$ para hablar de variables numéricas; $i, j, k, \ldots$ para supra o subindicar funciones; y, finalmente $t, u, v, \ldots$ y $a, b, c, \ldots$ cuando hablemos de términos de algún cálculo nombrado o con índices, respectivamente.

1.3 Sistemas de reescritura abstractos

1.3.1 Definiciones básicas

El estudio de los sistemas de reescritura abstractos (ARS – Abstract Reduction Systems) surge de la necesidad de tratar la noción abstracta de reescritura. Es decir, el análisis de sistemas en los que la actividad principal es una operación paso por paso. Este puede ser el caso, por ejemplo, de un sistema algebraico, en donde uno parte de una expresión y desea "evaluarla" para llegar a una expresión más simple. Como ejemplo, supongamos la expresión algebraica $(3 + 2) \times 7$, en donde uno puede armar una reducción:

$$(3 + 2) \times 7 \to 5 \times 7 \to 35$$

Es importante destacar que en el estudio de estos sistemas no se asume una estructura particular para los elementos, sino que se centra en estudiar una relación binaria entre elementos de un conjunto posiblemente infinito. De esta manera, es posible obtener propiedades aplicables a toda una serie de teorías especializadas que parten de la reescritura abstracta. Tal es el caso, por ejemplo, de los sistemas de reescritura de términos, el λ-cálculo, etc.

Definimos a continuación el concepto de ARS.

Definición 1.1 (ARS)**.** Un sistema de reescritura abstracto (ARS) es un par (A, R), con A un conjunto y R una relación binaria sobre A. Es decir, $R \subseteq A \times A$. Cuando $(a, b) \in R$, con $a, b \in A$, notamos $a \to_R b$, y llamamos *R-reducto* de a a b y *R-expansión* de b a a. Siempre que el contexto sea lo suficientemente claro, podemos utilizar la notación $a \to b$.

Definición 1.2 (Composición de relaciones)**.** Dadas dos relaciones $R \subseteq A \times B$ y $S \subseteq B \times C$, definimos la composición de R y S como:

$$R \circ S = \{(a, c) : \exists\, b \in B : a \to_R b \wedge b \to_S c\}$$

Definición 1.3 (Relaciones inducidas)**.** Dada una relación $R \subseteq A \times A$, definimos las siguientes relaciones generadas por $\to_R$ (notada aquí $\to$).

- Identidad: $\overset{0}{\to} = \{(a, a) : a \in A\}$
- i-ésima composición ($i \in \mathbb{N}$): $\overset{i+1}{\to} = \overset{i}{\to} \circ \to$
- Clausura transitiva: $\overset{+}{\to} = \bigcup_{i>0} \overset{i}{\to}$

- Clausura reflexo-transitiva: $\xrightarrow{*} = \xrightarrow{0} \cup \xrightarrow{+}$ (también notada $\twoheadrightarrow$)
- Clausura reflexiva: $\xrightarrow{=} = \xrightarrow{0} \cup \rightarrow$
- Relación inversa: $\rightarrow^{-1} = \{(b, a) : a \rightarrow b\}$ (también notada $\leftarrow$)
- Clausura simétrica: $\leftrightarrow = \rightarrow \cup \leftarrow$
- Clausura simétrico-transitiva: $\xleftrightarrow{+} = (\leftrightarrow)^+$
- Clausura reflexo-simétrico-transitiva: $\xleftrightarrow{*} = (\leftrightarrow)^*$ (también notada $=_R$)

Nota. *Es importante destacar que, dada una relacion R, la P clausura de R es la menor relación con la propiedad P que contiene a R. De esta manera, por ejemplo, podríamos definir a la clausura reflexo-transitiva de R (*i.e., $\xrightarrow{*}_R$*) como la menor relación reflexiva y transitiva que contiene a R.*

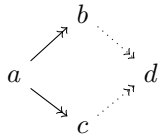
Mostramos aquí la terminología básica en lo que a sistemas de reducción abstractos se refiere.

Definición 1.4 (Terminología)**.** Sea (A, R) un ARS. Sean $a, b \in A$.

- a es **reducible** si y sólo si existe $b \in A : a \rightarrow b$.
- a **está en forma normal** si y sólo si no es reducible.
- b **es forma normal de** a si y sólo si $a \twoheadrightarrow b$ y b es forma normal. Si a tiene una única forma normal, notamos a ésta como $a\!\downarrow$.
- b es **sucesor directo** de a si y sólo si $a \rightarrow b$.
- b es **sucesor** de a si y sólo si $a \xrightarrow{+} b$.
- a y b **convergen** si y sólo si existe $c \in A : a \twoheadrightarrow c \wedge b \twoheadrightarrow c$. En dicho caso, notamos $a \downarrow b$.

1.3.2 Confluencia

En el estudio de los sistemas de reescritura abstractos, resulta de gran interés el análisis de la *confluencia*. Esto es: dados tres elementos $a, b, c \in A : a \twoheadrightarrow b \wedge a \twoheadrightarrow c$, ¿existe $d \in A : b \twoheadrightarrow d \wedge c \twoheadrightarrow d$? Gráficamente, lo representamos como:



Nota. *En el diagrama anterior, las líneas completas denotan un cuantificador universal (por ejemplo, $\forall\, a, b : a \twoheadrightarrow b$); y las punteadas, un cuantificador existencial (por ejemplo, $\exists\, b, d : b \twoheadrightarrow d$).*

Dicho en otras palabras: ¿es posible llegar siempre a un mismo elemento, sin importar el orden de reducción? Para el análisis de dicho problema se estudian diferentes problemas intermedios, que introducimos a continuación.

Definición 1.5 (Confluencia local)**.** Sea $R \subseteq A \times A$. Decimos que R es localmente confluente o **WCR** si y sólo si

$$\forall a, b, c \in A : a \to_R b \wedge a \to_R c \implies b \downarrow c$$

Definición 1.6 (Confluencia)**.** Sea $R \subseteq A \times A$. Decimos que R es confluente si y sólo si

$$\forall a, b, c \in A : a \twoheadrightarrow_R b \wedge a \twoheadrightarrow_R c \implies b \downarrow c$$

Observación 1.7. Si una relación $R \subseteq A \times A$ es confluente, entonces todo elemento de A tiene a lo sumo una forma normal bajo R. Esta conclusión proviene del hecho de observar que, por confluencia de R, se daría un absurdo si tuviéramos dos formas normales distintas b y c de un elemento a, pues sería posible volver a reducir en un elemento común d, contradiciendo el hecho de que b y c sean formas normales diferentes de a.

Definición 1.8 (Church-Rosser)**.** Sea $R \subseteq A \times A$. R se dice *Church-Rosser* o **CR** si y sólo si

$$\forall a, b \in A : a =_R b \implies a \downarrow b$$

Teorema 1.9. Sea $R \subseteq A \times A$. R es confluente si y sólo si R es CR.

Demostración. Una prueba puede encontrarse en [BN98], teorema 2.1.5. □

De aquí en más, utilizaremos los términos *Church-Rosser*, *CR* o *confluente* de manera indistinta.

Observación 1.10. Notar que CR $\implies$ WCR. Lo contrario (*i.e.,* WCR $\implies$ CR) no es cierto. Un contraejemplo puede encontrarse en [Ter03].

1.3.3 Terminación

En el campo de los ARSs, resulta también ser muy importante el estudio de la *terminación*. Esto es, la posibilidad de responder a preguntas tales como: ¿dado un ARS, es cierto que todos sus elementos tienen al menos una forma normal? ¿es cierto que para un ARS dado, no es posible obtener derivaciones infinitas? La posibilidad de responder preguntas como éstas puede dar certeza acerca de, por ejemplo, la finalización de una computación. Para introducir el problema, damos algunas definiciones y propiedades.

Definición 1.11 (Normalización débil)**.** Sea $R \subseteq A \times A$. Decimos que R es débilmente normalizante o **WN** si y sólo si todo elemento de A tiene forma normal bajo R.

Definición 1.12 (Normalización fuerte)**.** Sea $R \subseteq A \times A$. Decimos que R es fuertemente normalizante o **SN** si y sólo si no existen derivaciones infinitas. Es decir, no existen derivaciones de la forma

$$a_0 \to a_1 \to a_2 \to \dots$$

Observación 1.13. SN $\implies$ WN

Damos a continuación un resultado muy importante de la teoría de reescritura abstracta, que permite simplificar muchas pruebas para los problemas que se presentan.

Lema 1.14 (Newman)**.** Sea $R \subseteq A \times A$. Si R es SN y WCR, entonces R es CR.

Demostración. Dos pruebas diferentes pueden encontrarse en [Ter03]. □

Lema 1.15 (Existencia y unicidad de formas normales)**.** Si un ARS (A, R) es SN y WCR, entonces todo elemento de A tiene una única forma normal.

Demostración. Por observación 1.13, R es WN. Luego, todo elemento tiene al menos una forma normal. Además, por lema 1.14, R es CR. Entonces, por observación 1.7, las formas normales son únicas. □

1.3.4 Isomorfismo entre ARSs

Dado que será utilizado con frecuencia en este trabajo, introducimos aquí el concepto de isomorfismo entre dos sistemas de reescritura abstractos.

Informalmente, decimos que dos ARSs $(A, \to_A)$ y $(B, \to_B)$ son isomorfos (del griego, *isos* – igual y *morph* – forma) si y sólo si existe un *mapping* entre los objetos de A y de B, de forma tal que las relaciones a través de este mapping se preserven. Formalizamos este concepto a continuación:

Definición 1.16 (Isomorfismo entre dos ARSs)**.** Dados dos ARSs $(A, \to_A)$ y $(B, \to_B)$, decimos que son isomorfos si y sólo si existen dos funciones de traducción $\mathrm{w} : A \to B$ y $\mathrm{u} : B \to A$ tal que:

1. $\mathrm{w} \circ \mathrm{u} = \mathrm{Id}_B \ \wedge \ \mathrm{u} \circ \mathrm{w} = \mathrm{Id}_A$ (*i.e.,* las funciones w y u son biyectivas, y cada una es la recíproca de la otra)
2. $\forall\, a, a' \in A : a \to_A a' \implies \mathrm{w}(a) \to_B \mathrm{w}(a')$
3. $\forall\, b, b' \in B : b \to_B b' \implies \mathrm{u}(b) \to_A \mathrm{u}(b')$

La importancia que tiene la existencia de un isomorfismo entre dos sistemas de reescritura abstractos es la de preservación de propiedades entre uno y otro. Esto es: si los ARSs $(A, \to_A)$ y $(B, \to_B)$ son isomorfos, entonces $(A, \to_A)$ tiene la propiedad P si y sólo si $(B, \to_B)$ la tiene. En otras palabras, probar una propiedad para uno de los sistemas equivale a probarla para el otro.

1.4 Sistemas de reescritura de términos

1.4.1 Introducción

Los sistemas de reescritura de términos (TRS – Term Rewriting Systems) son sistemas de reescritura abstractos que trabajan sobre términos de primer orden, y cuyas relaciones están dadas por "reglas de reducción": reglas esquemáticas que definen qué términos se relacionan entre sí a través de pares de términos de primer orden que pueden ser instanciados de manera contextual.

A modo de ejemplo, podríamos pensar en términos de primer orden que representen fórmulas booleanas, y en reglas de reducción para evaluarlas:

$$\textbf{Términos:} \quad f ::= 0 \mid 1 \mid f \wedge f \mid f \vee f \mid \neg f$$

$$\textbf{Reglas de reducción:} \quad \begin{cases} 1 \wedge x \to x & (1) \\ x \wedge 0 \to 0 & (2) \\ 0 \vee x \to x & (3) \\ x \vee 1 \to 1 & (4) \\ \neg 1 \to 0 & (5) \\ \neg 0 \to 1 & (6) \end{cases}$$

Entonces, por ejemplo, podríamos tener la reducción:

$$1 \wedge 1 \to_{(1)} 1$$

reemplazando la x en la regla (1) por 1. De la misma manera, podemos reducir dentro de contextos más amplios (*i.e.,* no triviales):

$$\underbrace{(\neg 1)} \vee (\neg 0) \ \to_{(5)} \ \underbrace{0 \vee (\neg 0)} \ \to_{(3)} \ \underbrace{\neg 0} \ \to_{(6)} \ 1$$

Hecha esta pequeña introducción, pasamos a definir formalmente los sistemas de reescritura de términos.

1.4.2 Términos y contextos

En este apartado definimos los términos de primer orden sobre los que trabajan los TRSs.

Definición 1.17 (Signatura)**.** Una signatura Σ es un conjunto no vacío de símbolos de función $\{f, g, \ldots\}$. Cada uno de estos símbolos tiene asociado un natural $n \in \mathbb{N}$, que llamamos aridad. La función $\text{ar} : \Sigma \to \mathbb{N}$ nos da la aridad de un símbolo de función $f \in \Sigma$. Notamos tambien la aridad $\text{ar}(f)$ de un símbolo $f \in \Sigma$ como $f^{(n)}$.
En el ejemplo de las fórmulas booleanas, la signatura es:

$$\Sigma = \left\{ \wedge^{(2)}, \vee^{(2)}, \neg^{(1)}, 0^{(0)}, 1^{(0)} \right\}$$

Definición 1.18 (Términos para una signatura Σ y variables V)**.** Sea Σ una signatura y V un conjunto de variables. El conjunto de términos sobre Σ con variables V se nota $Ter(\Sigma, V)$, y está definido inductivamente como:

1. $V \subseteq Ter(\Sigma, V)$
2. $\forall f^{(n)} \in \Sigma, t_1, \ldots, t_n \in Ter(\Sigma, V) : f(t_1, \ldots, t_n) \in Ter(\Sigma, V)$

En un término de la forma $f(t_1, \ldots, t_n)$, llamamos *argumentos* o *parámetros* a los t_i, $1 \leq i \leq n$; y *símbolo raíz* o *símbolo cabeza* a f. En el ejemplo anterior, posibles términos son: $x, 0, 1, 1 \wedge 0, (\neg 0) \vee 1$.

Definimos ahora el conjunto de variables que posee un término.

Definición 1.19 (Variables de un término)**.** El conjunto de variables de un término $Var : Ter(\Sigma, V) \to \mathcal{P}(V)$ se define inductivamente como:

1. $Var(x) = \{x\}$
2. $Var(f(t_1, \ldots, t_n)) = \bigcup_{i=1}^{n} Var(t_i)$

A continuación definimos el concepto de "contexto". Un contexto no es más que un término con una o más ocurrencias de un símbolo especial $\Box$, que denominamos informalmente "agujero". Es decir, un contexto es un elemento del conjunto $Ter(\Sigma \cup \{\Box\}, V)$. Nos centraremos aquí – en pos de mantenernos dentro de los límites de la tesis en lo que a introducción respecta – en un tipo de contextos: aquellos que tienen sólo un agujero. Nos referiremos a éstos simplemente como "contextos", dado que en lo sucesivo no haremos referencia a contextos de más de un agujero.

Definición 1.20 (Contexto)**.** Un contexto C sobre los términos $Ter(\Sigma, V)$ se define inductivamente como:

1. $\Box$ es contexto (el contexto *vacío* o *trivial*).
2. $f(t_1, \ldots, t_{i-1}, C, t_{i+1}, \ldots, t_n)$ es contexto para todo $n \in \mathbb{N}_{>0}$, $f^{(n)} \in \Sigma$, $t_j \in Ter(\Sigma, V)$, $1 \leq i \leq n$, C contexto.

Definición 1.21 (Reemplazo en contextos)**.** Dado un contexto C sobre $Ter(\Sigma, V)$, y $t \in Ter(\Sigma, V)$, notamos $C[t]$ al reemplazo de $\Box$ por t en C. Dicho reemplazo se define inductivamente como:

1. $\Box[t] = t$
2. $f(t_1, \ldots, t_{i-1}, C, t_{i+1}, \ldots, t_n)[t] = f(t_1, \ldots, t_{i-1}, C[t], t_{i+1}, \ldots, t_n)$, con C contexto

1.4.3 Sustituciones

En este apartado presentamos el concepto de sustitución, formalismo indispensable para la definición de un sistema de reescritura de términos. Intuitivamente, una sustitución hace de "instanciador" de términos en el siguiente sentido: dado un término con variables, se asigna a cada variable algún término. De esta manera, es posible obtener términos distintos a partir de un término base. Esto es, el término resultante mantiene su estructura, pues el término base hace de "esqueleto". Más adelante veremos que esto sirve para "matchear" las reglas de reescritura al momento de realizar una reducción.

Definición 1.22 (Sustitución)**.** Dada una signatura Σ y un conjunto de variables V, una sustitución es una función $\sigma : V \to Ter(\Sigma, V)$ que cumple que:

$$\{x : x \in V \wedge \sigma(x) \neq x\} \text{ es finito}$$

La notación utilizada para denotar una sustitución σ es:

$$\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$$

con la semántica:

$$\begin{array}{lcll} \sigma(x_i) & = & t_i & \iff \quad x_i \in \{x_1, \ldots, x_n\} \\ \sigma(y) & = & y & \iff \quad y \notin \{x_1, \ldots, x_n\} \end{array}$$

Toda sustitución $\sigma : V \to Ter(\Sigma, V)$ se extiende naturalmente a una sustitución $\bar{\sigma} : Ter(\Sigma, V) \to Ter(\Sigma, V)$ de la siguiente manera:

$$\begin{array}{lcll} \bar{\sigma}(x) & = & \sigma(x) & \forall x \in V \\ \bar{\sigma}(f(t_1, \ldots, t_n)) & = & f(\bar{\sigma}(t_1), \ldots, \bar{\sigma}(t_n)) & \forall t_j \in Ter(\Sigma, V),\ f^{(n)} \in \Sigma \end{array}$$

De esta manera, definimos el conjunto de sustituciones para una signatura Σ y conjunto de variables V como:

$$Sus(\Sigma, V) = \{\sigma : V \to Ter(\Sigma, V) \ / \ \sigma \text{ es sustitución}\}$$

Nota. *A partir de aquí, usaremos $\bar{\sigma}$ y σ de manera indistinta.*

1.4.4 Identidades y reglas de reescritura

Introducimos aquí dos conceptos que nos permitirán definir los sistemas de reescritura de términos: las identidades y las reglas de reescritura. Los TRSs se definen a partir de reglas de reescritura, y estas últimas son casos especiales de identidades. Intuitivamente, una regla de reescritura es un par de términos que indican de qué manera es posible realizar una reducción dentro de un término (ver sección 1.4.1 para un ejemplo). Esto se hace encontrando una sustitución que haga que "una parte" de un término "unifique" con un "lado izquierdo" de una regla de reescritura, y reemplazando dicho subtérmino por el resultado de la misma sustitución aplicada al "lado derecho" de la regla de reescritura.

Definición 1.23 (Identidad)**.** Una identidad es un par $(s, t) \in Ter(\Sigma, V) \times Ter(\Sigma, V)$. La notamos $s \approx t$, y llamamos "lado izquierdo" o **lhs** a s y "lado derecho" o **rhs** a t.

Definición 1.24 (Regla de reescritura)**.** Una regla de reescritura es una identidad $l \approx r$, con $l, r \in Ter(\Sigma, V)$ tal que:

1. $l \notin V$
2. $Var(r) \subseteq Var(l)$

La notamos $l \to r$.

1.4.5 Relaciones de reescritura y TRSs

Como parte final de la introducción a los sistemas de reescritura de términos, presentamos el concepto de relación de reescritura y de TRS. Antes de hacerlo, damos algunas definiciones inherentes a relaciones binarias sobre términos de primer orden.

Definición 1.25. Dada una relación binaria $\equiv \subseteq Ter(\Sigma, V) \times Ter(\Sigma, V)$, decimos que:

1. $\equiv$ es **cerrada por sustituciones** si y sólo si para todo $\sigma \in Sus(\Sigma, V)$ y $s, t \in Ter(\Sigma, V)$ tenemos que $s \equiv t \implies \sigma(s) \equiv \sigma(t)$.
2. $\equiv$ es **cerrada por Σ-operaciones** si y sólo si para todo $f^{(n)} \in \Sigma$, $s_1, \ldots, s_n, t_1, \ldots, t_n \in Ter(\Sigma, V)$ tenemos que $s_i \equiv t_i$ para todo $i \in \{1, \ldots, n\} \implies f(s_1, \ldots, s_n) \equiv f(t_1, \ldots, t_n)$.

3. $\equiv$ es **compatible con Σ-operaciones** si y sólo si para todo $f^{(n)} \in \Sigma$, $i \in \{1, \ldots, n\}$, $s_1, \ldots, s_{i-1}, s, t, s_{i+1}, \ldots, s_n \in Ter(\Sigma, V)$ tenemos que $s \equiv t \implies f(s_1, \ldots, s_{i-1}, s, s_{i+1}, \ldots, s_n) \equiv f(s_1, \ldots, s_{i-1}, t, s_{i+1}, \ldots, s_n)$.

4. $\equiv$ es **compatible con Σ-contextos** si y sólo si para todo C contexto sobre $Ter(\Sigma, V)$, $s, t \in Ter(\Sigma, V)$ tenemos que $s \equiv t \implies C[s] \equiv C[t]$.

A continuación, mostramos un lema que relaciona algunos de los conceptos recién presentados.

Lema 1.26. Sea $\equiv \subseteq Ter(\Sigma, V) \times Ter(\Sigma, V)$.

1. $\equiv$ es compatible con Σ-operaciones si y sólo si $\equiv$ es compatible con Σ-contextos.

2. Si $\equiv$ es reflexiva y transitiva, entonces es compatible con Σ-operaciones si y sólo si es cerrada por Σ-operaciones.

Demostración. Una prueba puede encontrarse en [BN98], lema 3.1.11. □

Definición 1.27 (Relación de reescritura)**.** Una relación $\equiv \subseteq Ter(\Sigma, V) \times Ter(\Sigma, V)$ se dice **relación de reescritura** si y sólo si es compatible con Σ-operaciones y cerrada por sustituciones.

Definición 1.28 (Sistema de reescritura de términos)**.** Un sistema de reescritura de términos o TRS es un conjunto de reglas de reescritura $R \subseteq Ter(\Sigma, V) \times Ter(\Sigma, V)$. Un TRS define una relación, que notamos $\to_R$, como:

$$\forall s, t \in Ter(\Sigma, V) : s \to_R t \iff$$

$$(\exists l \to r \in R, C \text{ contexto}, \sigma \in Sus(\Sigma, V) : s = C[\sigma(l)] \wedge t = C[\sigma(r)])$$

Llamamos *redex* (por **re**ducible **ex**pression - expresión reducible) a s, y contracción de s a t.

Lema 1.29. $\to_R$ es una relación de reescritura. En particular, es la más chica que contiene a R.

Demostración. Una prueba del carácter de relación de reescritura de $\to_R$ puede encontrarse en [BN98], lema 3.1.10. La prueba es sencilla, y proviene de la definición de $\to_R$ de manera casi trivial.

Mostrar que además es la relación de reescritura más chica que contiene a R es un ejercicio sencillo, que parte de suponer que existe otra más chica y llegar a la conclusión de que son las mismas. □

Observación 1.30. Son también de reescritura $\overset{+}{\to}_R$, $\overset{*}{\to}_R$ y $\overset{*}{\leftrightarrow}_R$.

Demostración. Sencillas, por inducción en la cantidad de pasos de reducción. □

1.5 El cálculo λ

1.5.1 Introducción

El cálculo λ es un sistema concebido por Alonzo Church en los años '30. Originalmente, la idea de Church era la de conseguir una herramienta que formara parte de una teoría general y fundacional de la matemática. Sin embargo, en 1936, Kleene y Rosser mostraron que el sistema planteado era inconsistente. Actuando en consecuencia, Church aísla la parte del sistema encargado de la computación de funciones, obteniendo como resultado lo que hoy se conoce como el *cálculo* λ. Este cálculo resulta ser un modelo exitoso para las funciones computables (Kleene y Rosser prueban, también en 1936, que toda función recursiva es representable en el cálculo λ). A su vez, en 1937, Turing muestra que su formalismo (*i.e.,* la *máquina de Turing*) es equivalente en poder expresivo al cálculo λ.

El cálculo λ ha jugado un papel muy importante en áreas como lógica y computabilidad, y ha sentado las bases para lo que hoy conocemos como programación funcional y sistemas de tipos.

El concepto principal en el que se basa el cálculo λ es el de función. Esto es, matemáticamente, una relación funcional entre elementos de dos conjuntos. Desde el punto de vista computacional, podemos ver a una función como reglas mediante las cuales un *input* puede ser transformado en un *output*. Veamos, por ejemplo, la función "*succ*", que dado un número natural como *input*, nos provee como *output* a su sucesor (desde el punto de vista matemático, una relación funcional que asocia a cada natural n con su sucesor $n+1$). Tenemos, entonces:

$$succ(n) = n + 1$$

Podemos hacer aquí tres observaciones que sientan las bases de lo que es el cálculo λ: primero, que las funciones no necesitan ser nombradas explícitamente. Es decir, podríamos tener una notación genérica para el concepto de función. Por ejemplo:

$$n \to n + 1$$

Segundo, que el nombre de los parámetros (en este caso, n), es también irrelevante. Es decir, la expresión

$$x \to x + 1$$

denota exactamente la misma función. Por último, dada una función de más de un parámetro, podemos verla como una función de un parámetro que devuelve otra función. Por ejemplo, podemos ver a $x, y \to x + y$ (la función "suma") como:

$$x \to (y \to x + y)$$

Esto es, una función que dado un natural x, devuelve otra función de un parámetro que suma y a x (en este caso, x tendrá valor fijo).

Cabe destacar que la versión del cálculo λ que introducimos aquí es conocida como cálculo λ no tipado. Es decir, las expresiones no tienen asociados tipos. Existen versiones del cálculo λ que son tipadas (a saber, dos principales: cálculo λ tipado *à la Church* y *à la Curry*). Referimos al lector interesado a [BAG+92] para más detalles.

Presentadas ya las ideas intuitivas subyacentes al formalismo, pasamos a introducir el cálculo λ.

1.5.2 Términos

En el cálculo λ, existen dos operaciones básicas: aplicación y abstracción. La primera de ellas, aplicación, se nota como:

$$t\,u \qquad \text{con } t, u \text{ términos}$$

y denota la aplicación del parámetro u a la función t (usualmente notado en matemática como $t(u)$). Cabe destacar que el carácter no tipado del cálculo λ clásico hace posible que cualquier término pueda oficiar al mismo tiempo de función y de parámetro.

La segunda de las operaciones, abstracción, se nota como:

$$\lambda x.t \qquad \text{con } x \text{ una variable y } t \text{ un término}$$

y denota intuitivamente la función

$$x \to t[x]$$

donde $t[x]$ denota un término t dependiente de una variable x. En este caso, decimos que la abstracción *liga* a la variable x en t. Definiremos más adelante el concepto de variables libres y variables ligadas, que intuitivamente es: una variable está ligada si hay una abstracción que la liga; y está libre si no.

De esta manera, definimos los términos del cálculo λ, junto con los conceptos de variables libres y variables ligadas:

Definición 1.31 (Términos del cálculo λ)**.** El conjunto de términos del cálculo λ, denotado Λ, está dado por:

$$t ::= x \mid t\,t \mid \lambda x.t \qquad \text{con } x \in V$$

donde $V = \{x, y, z, \ldots\}$.

Nota. *La convención utilizada para la asociatividad y precedencia de los operadores de aplicación y abstracción es:*

- $t\,u\,v = (t\,u)\,v$ *(aplicación con prioridad a izquierda).*
- $\lambda x\,y.t = \lambda x.\lambda y.t = \lambda x.(\lambda y.t)$ *(abstracción con prioridad a derecha, junto con abuso de notación para simplificar la escritura).*
- $\lambda x.t\,u = \lambda x.(t\,u)$ *(la aplicación tiene mayor precedencia que la abstracción).*

Definición 1.32 (Variables libres de un término)**.** El conjunto de variables libres de un término de Λ, $\mathrm{FV} : \Lambda \to \mathcal{P}(V)$, se define inductivamente como:

$$\begin{aligned} \mathrm{FV}(x) &= \{x\} \\ \mathrm{FV}(t\,u) &= \mathrm{FV}(t) \cup \mathrm{FV}(u) \\ \mathrm{FV}(\lambda x.t) &= \mathrm{FV}(t) - \{x\} \end{aligned}$$

Definición 1.33 (Variables ligadas de un término)**.** El conjunto de variables ligadas de un término de Λ, $\mathrm{BV} : \Lambda \to \mathcal{P}(V)$, se define inductivamente como:

$$\begin{aligned} \mathrm{BV}(x) &= \emptyset \\ \mathrm{BV}(t\,u) &= \mathrm{BV}(t) \cup \mathrm{BV}(u) \\ \mathrm{BV}(\lambda x.t) &= \mathrm{BV}(t) \cup \{x\} \end{aligned}$$

1.5.3 Metasustitución

A la hora de "evaluar" una función, como puede ser por ejemplo el caso de $(x \to x * x)(3)$, es necesario realizar una sustitución: donde dice x, hay que colocar 3, de modo tal de obtener la expresión $3 * 3$. En el cálculo λ es necesaria dicha noción, formalizada mediante una operación atómica que denominamos *metasustitución*. La idea intuitiva de la metasustitución es el reemplazo de todas las variables libres x de un término t por otro término u. Lo formalizamos a continuación.

Definición 1.34 (Metasustitución)**.** Definimos la operación de metasustitución $\bullet\{\bullet := \bullet\} : \Lambda \times V \times \Lambda \to \Lambda$ inductivamente como:

$$\begin{array}{rcl} x\{x := v\} & = & v \\ y\{x := v\} & = & y \qquad \text{si } x \neq y \\ (t\,u)\{x := v\} & = & t\{x := v\}\ u\{x := v\} \\ (\lambda x.t)\{x := v\} & = & \lambda x.t \\ (\lambda y.t)\{x := v\} & = & \lambda y'.t\{y := y'\}\{x := v\} \\ & & \quad \text{con } y' \notin \mathrm{FV}(v) \cup \{x\} \cup (\mathrm{FV}(t) - \{y\}) \end{array}$$

Nota. *Esta definición fue tomada de [BG99]. Difiere de otras definiciones encontradas en la literatura en que posee un caso menos (los casos en los que se produce captura y en los que no, se condensan en el último de los casos presentados). Una* captura *ocurre cuando, al efectuarse una sustitución, alguna variable libre del término que sustituye es ligada por alguna abstracción del término en el que se efectúa la sustitución. Las capturas son indeseables, pues alteran la semántica esperada del término en cuestión. Para evitarlas, simplemente se efectúa un* renombre *de la variable ligada al sustituir dentro de una abstracción.*

1.5.4 α-equivalencia

Como mencionamos en la sección 1.5.1, el nombre de los parámetros es irrelevante. Por ejemplo, las expresiones:

$$\lambda x.x \qquad \text{y} \qquad \lambda y.y$$

denotan la función identidad, siendo la única diferencia el nombre de sus variables ligadas. Formalizamos esta noción presentando una relación conocida como α-equivalencia: decimos que dos términos son α-equivalentes si y sólo si uno puede ser obtenido a partir del otro a través del renombre de variables ligadas.

Definición 1.35 (α-equivalencia)**.** Definimos $=_\alpha \subseteq \Lambda \times \Lambda$ como la relación de equivalencia más chica que cumple:

$$\begin{array}{rcl} x =_\alpha x & & \\ t =_\alpha t' \wedge u =_\alpha u' & \Longrightarrow & t\,u =_\alpha t'\,u' \\ t =_\alpha t' \wedge y \notin \mathrm{FV}(t') - \{x\} & \Longrightarrow & \lambda x.t =_\alpha \lambda y.t'\{x := y\} \end{array}$$

Consideramos ahora los términos del cálculo λ cocientados por la α-equivalencia: $\Lambda/ =_\alpha$, y podremos utilizar la denominada *convención de Barendregt* o *convención de variables*: *"en cualquier contexto matemático, todas las variables ligadas*

de los términos son diferentes de todas las variables libres. Además, letras diferentes denotan variables diferentes."

Con esto en mente, pasamos a definir una metasustitución más sencilla, asumiendo la convención de Barendregt:

Definición 1.36 (Metasustitución con convención de variables)**.** Definimos la metasustitución $\bullet\{\bullet := \bullet\} : (\Lambda/=_\alpha) \times V \times (\Lambda/=_\alpha) \to (\Lambda/=_\alpha)$ inductivamente como:

$$\begin{aligned} x\{x := v\} &= v \\ y\{x := v\} &= y \\ (t\,u)\{x := v\} &= t\{x := v\}\ u\{x := v\} \\ (\lambda y.t)\{x := v\} &= \lambda y.t\{x := v\} \end{aligned}$$

Nota. *Cabe destacar que, durante todo el trabajo, haremos las aclaraciones pertinentes cuando utilicemos una u otra definición de la metasustitución, así como cuando estemos o no trabajando con la convención de variables.*

1.5.5 β-reducción

El cálculo λ tiene una única regla de reducción, denominada β. Dicha regla es la que implementa la "evaluación" de las funciones. La definimos a continuación.

Definición 1.37 (β-reducción)**.** Para todo $t, u \in \Lambda$, $\beta \subseteq \Lambda \times \Lambda$ se define como:

$$t \to_\beta u \iff \exists\, C \text{ contexto}, v, w \in \Lambda : t = C[(\lambda x.v)\, w] \wedge u = C[v\{x := w\}]$$

Para asegurarnos que es posible trabajar módulo α-equivalencia, existe un lema que nos dice que la β-reducción se comporta correctamente.

Lema 1.38. β "pasa bien al cociente" $(\Lambda/=_\alpha)$. Esto es,

$$\forall\, t, t', v \in \Lambda : t =_\alpha t' \wedge t \to_\beta v \implies (\exists\, v' \in \Lambda : v =_\alpha v' \wedge t' \to_\beta v')$$

Demostración. Una prueba puede encontrarse en [LV02], proposición 2.3.3. □

Con esta definición y lema, podemos definir el cálculo λ formalmente:

Definición 1.39 (Cálculo λ)**.** El cálculo λ es el sistema de reducción $(\Lambda/=_\alpha,\ \beta)$.

Concluimos la presentación del cálculo λ con dos propiedades muy importantes de éste.

Observación 1.40. El cálculo λ no es SN ni WN. Lo podemos ver con el siguiente contraejemplo:

$$(\lambda x.x\,x)\,(\lambda x.x\,x) \to_\beta (x\,x)\{x := (\lambda x.x\,x)\} = (\lambda x.x\,x)\,(\lambda x.x\,x) \to_\beta \ldots$$

Teorema 1.41. El cálculo λ es CR.

Demostración. Existen numerosas pruebas en la literatura. Por ejemplo, en [Bar84], sección 11.1. □

1.6 El cálculo λ con índices *à la de Bruijn*

1.6.1 Introducción

En [dB72], Nicolaas Govert de Bruijn presenta una novedosa versión del cálculo λ, que denominamos cálculo λ con índices *à la de Bruijn* (o simplemente λ_{dB}). La novedad proviene del hecho de que de Bruijn consigue una notación que evita tener que lidiar con la α-equivalencia. Esto es: términos α-equivalentes en el λ cálculo clásico son sintácticamente iguales con la nueva notación. La ventaja de esto radica en que la α-equivalencia es tediosa y costosa a la hora de implementar el cálculo λ.

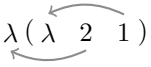
Para evitar la necesidad de trabajar módulo α-equivalencia, de Bruijn reemplaza las variables por *índices* naturales. Un índice representa la ocurrencia de una variable en un término, y su valor denota la cantidad de abstracciones que se deben atravesar para encontrar la abstracción que le corresponde a dicho índice. Por ejemplo, el término:

$$\lambda x.\lambda y.x\, y$$

se escribe en λ_{dB} como:

$$\lambda\lambda 2\, 1$$

En este caso, el índice 2 está ligado por la primera de las abstracciones, y el 1 por la segunda:



De esta manera, términos tales como $\lambda x.x$ y $\lambda y.y$ se escriben en λ_{dB} como $\lambda 1$.

Existe un caso especial a considerar para realizar las traducciones entre una y otra notación: cuando los términos poseen variables libres. En este caso, lo que se hace es predefinir un orden para las variables del λ cálculo clásico. Luego, el índice que le corresponderá a una variable x cuya posición en el orden de variables sea i será $i + j$, donde j es la cantidad de abstracciones que tiene por delante la variable. Esto es, el índice dependerá del contexto en donde aparezca la variable. Consideremos, por ejemplo, el término $(\lambda x.x\, y\, (\lambda z.z\, y))\, x\, z$. Supongamos que ordenamos las variables así: $[x, y, z]$. La única variable que sólo aparece libre en el término es y, cuya posición en el orden es 2. Las variables x y z aparecen tanto libres como ligadas, y sus posiciones son 1 y 3, respectivamente. La traducción a λ_{dB} es:

$$(\lambda\, 1\ \underset{[2+1]}{3}\ (\lambda\, 1\ \underset{[2+2]}{4}))\ \underset{[1+0]}{1}\ \underset{[3+0]}{3}$$

Hecha esta introducción informal, pasamos a definir formalmente el cálculo λ_{dB}.

1.6.2 Algunas definiciones

Antes de definir λ_{dB} vamos a necesitar definir un número de operaciones entre conjuntos de números naturales y números naturales, presentando luego ciertas propiedades sobre éstas. Tanto las definiciones como las propiedades fueron

tomadas de [KR98], excepto por la definición 7 y las propiedades 4, 7, 8 (caso de igualdad) y 9. En todos los casos, las demostraciones son sencillas, y las obviamos aquí.

Definición 1.42. Sean $N \subset \mathbb{N}$, $k \in \mathbb{N}$. Definimos:

1. $N - k = \{n - k : n \in N \wedge n > k\}$
2. $N + k = \{n + k : n \in N\}$
3. $N_{<k} = \{n : n \in N \wedge n < k\}$
4. $N_{\leq k} = \{n : n \in N \wedge n \leq k\}$
5. $N_{>k} = \{n : n \in N \wedge n > k\}$
6. $N_{\geq k} = \{n : n \in N \wedge n \geq k\}$
7. $N_{=k} = \{n : n \in N \wedge n = k\}$

Lema 1.43. Sean $N, M \subset \mathbb{N}$, $k, k' \in \mathbb{N}$. Luego,

1. $(N \cup M) - k = (N - k) \cup (M - k)$
2. $(N \cup M) + k = (N + k) \cup (M + k)$
3. $(N - k) - k' = N - (k + k')$
4. $N - k = N_{>k} - k$
5. $(N_{>k+1} - 1) + 1 = (N_{>k+1} + 1) - 1$
6. $(N + k) - 1 = N + (k - 1) \quad \text{si } k \geq 1$
7. $(N + 1) - (k + 1) = N - k$
8. $(N - 1)_{\,\theta\, k} = N_{\theta\, k+1} - 1 \quad \text{con } \theta \in \{<, \leq, >, \geq, =\}$
9. $(N \cup M)_{\,\theta\, k} = N_{\theta\, k} \cup M_{\theta\, k} \quad \text{con } \theta \in \{<, \leq, >, \geq, =\}$

1.6.3 Definición de λ_{dB} y propiedades

En esta sección definiremos el cálculo λ_{dB}, viendo cómo se implementa la β-reducción (*i.e.,* β_{dB}-reducción) con esta notación. Comenzamos por definir los términos del cálculo y el conjunto de variables libres.

Definición 1.44 (Términos de λ_{dB})**.** El conjunto de términos de λ_{dB}, que notamos Λ_{dB}, está dado por:

$$a ::= n \mid a\, a \mid \lambda a \qquad n \in \mathbb{N}_{>0}$$

Definición 1.45 (Variables libres de un término de Λ_{dB})**.** El conjunto de variables libres de un término, $\mathrm{FV} : \Lambda_{\mathrm{dB}} \to \mathcal{P}(\mathbb{N}_{>0})$, se define inductivamente como:

$$\begin{aligned} \mathrm{FV}(n) &= \{n\} \\ \mathrm{FV}(a\, b) &= \mathrm{FV}(a) \cup \mathrm{FV}(b) \\ \mathrm{FV}(\lambda a) &= \mathrm{FV}(a) - 1 \end{aligned}$$

Antes de definir la metasustitución para λ_{dB}, creemos importante introducir de manera intuitiva lo que ocurre, dado que más adelante haremos un cuestionamiento de esto mismo.

Supongamos que en un término a, queremos sustituir la variable libre n por el término b. Hacemos dos observaciones con respecto a la distribución de la metasustitución dentro de un término:

1. Recordemos que al cruzar una abstracción, la variable libre n pasa a ser representada por el índice $n+1$. Por lo tanto, al cruzar una abstracción, deberemos sumar uno al índice indicado en la metasustitución.

2. Una vez alcanzado el índice n con la metasustitución, éste deberá ser reemplazado por b. Observemos que esto no es tan trivial, pues el índice n nos indica que hemos cruzado $n-1$ abstracciones. Luego, será necesario incrementar en $n-1$ las variables libres del término b antes de realizar el reemplazo en cuestión. Para esto, de Bruijn introduce un nuevo metaoperador de *updating* de índices.

Definimos, ahora sí, la metasustitución y el metaoperador de *updating*.

Definición 1.46 (Metaoperador de *updating*)**.** Para todo $a, b \in \Lambda_{\mathrm{dB}}$, $n \in \mathbb{N}_{>0}$, el operador $\mathrm{U}_k^i : \Lambda_{\mathrm{dB}} \to \Lambda_{\mathrm{dB}}$, con $k \in \mathbb{N} \wedge i \in \mathbb{N}_{>0}$ se define inductivamente como:

$$\begin{array}{rcl} \mathrm{U}_k^i(n) & = & \begin{cases} n & \text{si } n \leq k \\ n+i-1 & \text{si } n > k \end{cases} \\ \mathrm{U}_k^i(a\,b) & = & \mathrm{U}_k^i(a)\,\mathrm{U}_k^i(b) \\ \mathrm{U}_k^i(\lambda a) & = & \lambda \mathrm{U}_{k+1}^i(a) \end{array}$$

Nota. *Intuitivamente, el índice k oficia de contador de abstracciones atravesadas; mientras que el índice i indica en cuánto deben incrementarse los índices del término afectado.*

Observación 1.47. Es trivial mostrar que $\forall a \in \Lambda_{\mathrm{dB}} : \mathrm{U}_0^1(a) = a$.

Definición 1.48 (Metasustitución para λ_{dB})**.** Para todo $a, b, c \in \Lambda_{\mathrm{dB}}$, $n, m \in \mathbb{N}_{>0}$, $\bullet\{\{\bullet \leftarrow \bullet\}\} : \Lambda_{\mathrm{dB}} \times \mathbb{N}_{>0} \times \Lambda_{\mathrm{dB}} \to \Lambda_{\mathrm{dB}}$ se define inductivamente como:

$$\begin{array}{rcl} m\{\{n \leftarrow c\}\} & = & \begin{cases} m & \text{si } m < n \\ \mathrm{U}_0^n(c) & \text{si } m = n \\ m-1 & \text{si } m > n \end{cases} \\ (a\,b)\{\{n \leftarrow c\}\} & = & a\{\{n \leftarrow c\}\}\, b\{\{n \leftarrow c\}\} \\ (\lambda a)\{\{n \leftarrow c\}\} & = & \lambda a\{\{n+1 \leftarrow c\}\} \end{array}$$

Nota. *El caso en que se decrementa en uno la variable afectada, se debe a que se asume que la operación proviene de una β_{dB}-reducción. Por ende, ha desaparecido una abstracción y ello debe ser reflejado. Todo debería quedar más claro con la siguiente definición.*

Definición 1.49 (β_{dB}-reducción)**.** Para todo $a, b \in \Lambda_{\mathrm{dB}}$, $\beta_{\mathrm{dB}} \subseteq \Lambda_{\mathrm{dB}} \times \Lambda_{\mathrm{dB}}$ se define como:

$$a \ \to_{\beta_{\mathrm{dB}}} \ b \iff \exists\, C \text{ contexto}, c, d \in \Lambda_{\mathrm{dB}} : a = C[(\lambda c)\, d] \wedge b = C[c\{\{1 \leftarrow d\}\}]$$

Una vez definida la metasustitución, podemos definir el cálculo.

Definición 1.50 (Cálculo λ_{dB})**.** El cálculo λ_{dB} es el sistema de reducción $(\Lambda_{\mathrm{dB}}\,,\ \beta_{\mathrm{dB}})$.

Teorema 1.51 (λ y λ_{dB} son isomorfos)**.** Los cálculos λ y λ_{dB} son isomorfos.

Demostración. Una prueba puede encontrarse en [KR98]. □

Corolario 1.52. Los cálculos λ y λ_{dB} poseen las mismas propiedades.

Demostración. Consecuencia directa del teorema 1.51. □

Nota. *Como ejemplo, y entre otras propiedades derivadas del isomorfismo, podemos mencionar que los cálculos λ y λ_{dB} son confluentes y no fuertemente normalizantes.*

Antes de cerrar la presentación del cálculo, enunciamos dos lemas que aparecen en [KR97], y que utilizaremos más adelante.

Lema 1.53 (Lema 6 de [KR97])**.**

$$\forall a \in \Lambda_{\mathrm{dB}},\ i,j \in \mathbb{N}_{>0} : \mathrm{U}_0^i(\mathrm{U}_0^j(a)) = \mathrm{U}_0^{i+j-1}(a)$$

Lema 1.54 (Lema 10 de [KR97])**.**

$$\forall a,b \in \Lambda_{\mathrm{dB}},\ n,i \in \mathbb{N}_{>0},\ k \in \mathbb{N} : n \leq k+1 \implies$$

$$\mathrm{U}_k^i(a\{\{n \leftarrow b\}\}) = \mathrm{U}_{k+1}^i(a)\{\{n \leftarrow \mathrm{U}_{k-n+1}^i(b)\}\}$$

1.7 Sustituciones explícitas

1.7.1 Introducción y propiedades

Como se habrá podido apreciar en los cálculos λ y λ_{dB}, llamamos *metasustitución* a la operación de sustituir una variable en un término por otro. Esto se debe a que dicha operación se encuentra en el metalenguaje; es decir, no existe una construcción específica dentro del mismo lenguaje para denotar las operaciones de sustitución, ni, en consecuencia, reglas para evaluarlas. La metasustitución es atómica y bidireccional (*i.e.,* se utiliza la igualdad).

En [ACCL91] se presenta un cálculo con índices de *de Bruijn* – conocido como $\lambda\sigma$ – que incluye dentro del lenguaje la construcción de sustitución, junto con reglas para su evaluación. Una de las principales motivaciones para esta innovación es la de proveer un acercamiento entre la teoría y posibles implementaciones, puesto que si no se toman medidas a la hora de implementar, se puede provocar una explosión en el tamaño de los términos. Esto es, por supuesto, indeseable. Por otro lado, $\lambda\sigma$ introduce lo que se conoce como *composición de sustituciones*, mecanismo que logra evitar la obligación de esperar a que las sustituciones internas terminen de realizarse para poder ejecutar las externas. La composición de sustituciones resulta ser una característica clave para la obtención de *metaconfluencia*, una propiedad que es de suma utilidad en ciertas aplicaciones de estos cálculos, y que más adelante estaremos explicando.

El cálculo $\lambda\sigma$ resulta *simular* correctamente el cálculo λ_{dB}, ser *sound* (*i.e.,* dados dos términos $t, u \in \Lambda$, no es posible llegar de t a u si no ocurre que $t \twoheadrightarrow_\beta u$; en otras palabras: no agrega reducciones β inexistentes) y ser confluente, entre otras propiedades. En [Mel95], sin embargo, Melliès muestra un término que es SN en el λ cálculo tradicional, pero que bajo $\lambda\sigma$ admite una derivación infinita. Este fenómeno, conocido como *contraejemplo de Melliès*, representa la apertura de una época de búsqueda en el área; la búsqueda de un cálculo con sustituciones explícitas (con o sin índices de *de Bruijn*) que cumpliera toda una serie de propiedades, que enumeramos a continuación.

Sea λ_{Z} un cálculo con sustituciones explícitas, Λ_{Z} su conjunto de términos, Z su cálculo de sustituciones asociado (*i.e.,* aquellas reglas que se ocupan de evaluar las sustituciones) y $\mathrm{to}_{\mathrm{Z}} : \Lambda \to \Lambda_{\mathrm{Z}}$ y $\mathrm{from}_{\mathrm{Z}} : \Lambda_{\mathrm{Z}} \to \Lambda$ traducciones desde y hacia términos del λ cálculo tradicional. Las siguientes son algunas de las propiedades deseables de λ_{Z} (en particular, las que sirven al propósito de este trabajo):

- **(Sim) - Simulación**: Una propiedad básica de cualquier cálculo de sustituciones explícitas: el cálculo λ_{Z} simula al λ cálculo clásico. Esto es:

$$\forall t, u \in \Lambda : t \ \to_\beta \ u \implies \mathrm{to}_{\mathrm{Z}}(t) \ \twoheadrightarrow_{\lambda_{\mathrm{Z}}} \ \mathrm{to}_{\mathrm{Z}}(u)$$

- **(Snd) - Soundness**: Otra propiedad básica de cualquier cálculo de sustituciones explícitas: el cálculo λ_{Z} es *sound* con respecto al λ cálculo clásico. Esto es:

$$\forall t, u \in \Lambda_{\mathrm{Z}} : t, u \text{ en Z-f.n.} \wedge \ t \ \twoheadrightarrow_{\lambda_{\mathrm{Z}}} \ u \implies \mathrm{from}_{\mathrm{Z}}(t) \ \twoheadrightarrow_\beta \ \mathrm{from}_{\mathrm{Z}}(u)$$

- **($\mathbf{CR_Z}$) y ($\mathbf{SN_Z}$)**: El cálculo Z es confluente y fuertemente normalizante (y, como consecuencia de ambas, las Z-formas normales son únicas).

- **(CR)**: El cálculo λ_{Z} es confluente.

- **(MC) - Metaconfluencia**: El cálculo λ_{Z} extendido a términos abiertos (*i.e.,* términos que admiten variables que representan pruebas / partes de programas incompletos) es confluente. Esta propiedad es, en general, difícil de obtener en simultáneo con la propiedad que presentamos abajo.

- **(PSN) - Preservación de normalización fuerte**: Todo término fuertemente normalizante en el λ cálculo clásico también lo es en λ_{Z}. Esta propiedad es, históricamente, la más difícil de probar; y, al mismo tiempo, de obtener en simultáneo con **MC**.

Desde el contraejemplo de Melliès han aparecido numerosos cálculos de sustituciones explícitas – con nombres, índices, etc. – que intentan, generalmente sin éxito, obtener todas las propiedades anteriores. Mostramos a continuación algunos de ellos, con los que lidiaremos durante el trabajo.

1.7.2 El cálculo λx

En [Blo95, Blo97] se presenta un cálculo de sustituciones explícitas con nombres: λx. Dicho cálculo es uno de los más sencillos que pueden pensarse, y se obtiene a partir del agregado de la construcción de sustitución al lenguaje, y de la

orientación de las igualdades de la metasustitución del cálculo λ clásico para la evaluación de dichas sustituciones. Presentamos aquí este cálculo, extendiendo las nociones conocidas de variables libres, α-equivalencia y metasustitución para la sustitución explícita.

Nota. *Utilizamos la notación ya presentada $t\{x := u\}$ para denotar la clásica metasustitución; y la nueva notación (presentada en la siguiente definición) $t[x := u]$ para denotar sustitución explícita.*

Definición 1.55 (Términos del cálculo λx)**.** El conjunto de términos de λx, denotado Λx, está dado por:

$$t ::= x \mid t\,t \mid \lambda x.t \mid t[x := t] \qquad \text{con } x \in V$$

donde $V = \{x, y, z, \ldots\}$.

Definición 1.56 (Variables libres de un término)**.** El conjunto de variables libres de un término de Λx, $\mathrm{FV} : \Lambda\mathrm{x} \to \mathcal{P}(V)$, se define inductivamente como:

$$\begin{aligned}
\mathrm{FV}(x) &= \{x\} \\
\mathrm{FV}(t\,u) &= \mathrm{FV}(t) \cup \mathrm{FV}(u) \\
\mathrm{FV}(\lambda x.t) &= \mathrm{FV}(t) - \{x\} \\
\mathrm{FV}(t[x := u]) &= (\mathrm{FV}(t) - \{x\}) \cup \mathrm{FV}(u)
\end{aligned}$$

Definición 1.57 (Metasustitución)**.** Definimos la metasustitución, $\bullet\{\bullet := \bullet\} : \Lambda\mathrm{x} \times V \times \Lambda\mathrm{x} \to \Lambda\mathrm{x}$ inductivamente como:

$$\begin{aligned}
x\{x := v\} &= v \\
y\{x := v\} &= y \qquad \text{si } x \neq y \\
(t\,u)\{x := v\} &= t\{x := v\}\ u\{x := v\} \\
(\lambda x.t)\{x := v\} &= \lambda x.t \\
(\lambda y.t)\{x := v\} &= \lambda y'.t\{y := y'\}\{x := v\} \\
&\qquad \text{con } y' \notin \mathrm{FV}(v) \cup \{x\} \cup (\mathrm{FV}(t) - \{y\}) \\
(t[x := u])\{x := v\} &= t[x := u\{x := v\}] \\
(t[y := u])\{x := v\} &= t\{y := y'\}\{x := v\}\,[y' := u\{x := v\}] \\
&\qquad \text{con } y' \notin \mathrm{FV}(v) \cup \{x\} \cup (\mathrm{FV}(t) - \{y\})
\end{aligned}$$

Definición 1.58 (α-equivalencia)**.** Definimos $=_\alpha \subseteq \Lambda\mathrm{x} \times \Lambda\mathrm{x}$ como la relación de equivalencia más chica que cumple:

$$\begin{aligned}
x &=_\alpha x \\
t =_\alpha t' \wedge u =_\alpha u' &\implies t\,u =_\alpha t'\,u' \\
t =_\alpha t' \wedge y \notin \mathrm{FV}(t') - \{x\} &\implies \lambda x.t =_\alpha \lambda y.t'\{x := y\} \\
t =_\alpha t' \wedge u =_\alpha u' \wedge y \notin \mathrm{FV}(t') - \{x\} &\implies t[x := u] =_\alpha t'\{x := y\}\,[y := u']
\end{aligned}$$

Presentadas ya las definiciones (y extensiones) correspondientes, podemos apreciar las reglas del cálculo λx en la figura 1.1. Es fácil ver que se mantiene la clásica regla β, ahora llamada (Beta); y se agregan reglas para la distribución de las sustituciones explícitas dentro de los términos.

(Beta)	$(\lambda x.t)\, u$	$\rightarrow$	$t[x := u]$	
(App)	$(t\, u)[x := v]$	$\rightarrow$	$t[x := v]\, u[x := v]$	
(Lamb)	$(\lambda y.t)[x := v]$	$\rightarrow$	$\lambda y.t[x := v]$	$(y \neq x \wedge y \notin \mathrm{FV}(v))$
(Var)	$x[x := v]$	$\rightarrow$	v	
(VarR)	$y[x := v]$	$\rightarrow$	y	$(y \neq x)$

Figura 1.1: Reglas del cálculo λx

Llamamos x al conjunto de reglas (App), (Lamb), (Var) y (VarR). Dicho conjunto es el que conforma las reglas necesarias para la distribución de las sustituciones explícitas dentro de un término.

Llamamos, además, λx al conjunto de reglas (Beta) + x.

Las relaciones de reducción x-reducción y λx-reducción se definen como las clausuras contextuales de las reglas x y λx, respectivamente. Dicho esto, el cálculo λx se define como el sistema de reducción $(\Lambda\mathrm{x}, \lambda\mathrm{x})$.

El cálculo λx tiene, entre otras, las propiedades (Sim), (Snd), ($\mathrm{CR_x}$), ($\mathrm{SN_x}$), (CR) y (PSN). Para más detalles, referimos al lector a [Blo95, Blo97, BG99].

1.7.3 El cálculo λxgc

En [BR95] se presenta un cálculo de sustituciones explícitas muy similar a λx: λxgc. La diferencia radica en el agregado de una regla de *Garbage Collection*.

A modo de clarificar la situación, la idea de *Garbage Collection* es que el cálculo mismo se dé cuenta cuándo una sustitución aplicada a un término es vacua. Por vacua, queremos decir que la sustitución no afecta al término en cuestión. En el caso particular de λx, dicha vacuidad ocurre cuando, en $t[x := u]$, $x \notin \mathrm{FV}(t)$. En tal situación, bajo λx, la sustitución se distribuye dentro del término t. Al llegar a las variables, dado que son todas de la forma $y \neq x$, se reduce por la regla (VarR) a ese mismo y. Por ende, la idea para *Garbage Collection* es esta: si la variable x no está libre en t, el resultado de la sustitución es el mismo término t.

Es importante volver a recalcar que la *única* diferencia entre λx y λxgc es la regla de *Garbage Collection* (llamada GC). Por ende, las definiciones para términos, variables libres, metasustitución y α-equivalencia se mantienen iguales. En consecuencia, notaremos como Λx a los términos de λxgc, y la notación será la misma en todos los casos.

Las reglas del cálculo λxgc pueden verse en la figura 1.2. De manera análoga a lo acontecido para λx, llamamos xgc al conjunto de reglas x + (GC); y λxgc al conjunto de reglas (Beta) + xgc.

Las relaciones de reducción correspondientes a xgc y a λxgc se definen igual que para λx. Finalmente, llamamos λxgc al sistema de reducción $(\Lambda\mathrm{x}, \lambda\mathrm{xgc})$.

Es importante mencionar que el cálculo λxgc posee las mismas propiedades que λx, radicando su ventaja en una mejora de *performance* para posibles implementaciones (*i.e.,* la regla de *Garbage Collection* hace posible esto, ahorrando pasos innecesarios de reducción). Para el siguiente cálculo, sin embargo, la inclusión de esta regla no tiene sólo fines implementativos, sino que hace un aporte

(Beta)	$(\lambda x.t)\, u$	$\to$	$t[x := u]$	
(App)	$(t\, u)[x := v]$	$\to$	$t[x := v]\, u[x := v]$	
(Lamb)	$(\lambda y.t)[x := v]$	$\to$	$\lambda y.t[x := v]$	$(y \neq x \wedge y \notin \mathrm{FV}(v))$
(Var)	$x[x := v]$	$\to$	v	
(VarR)	$y[x := v]$	$\to$	y	$(y \neq x)$
(GC)	$t[x := v]$	$\to$	t	$(x \notin \mathrm{FV}(t))$

Figura 1.2: Reglas del cálculo λxgc

muy importante a la parte teórica (puntualmente, juega un rol primordial en la obtención de la propiedad MC).

1.7.4 El cálculo λex

El cálculo λex es el último cálculo de sustituciones explícitas con nombres que introduciremos. Dicho cálculo es presentado en [Kes08, Kes09], y representa un importante avance en el área. El motivo: se consigue un cálculo con todas las propiedades enumeradas anteriormente.

Cabe destacar que, a través del tiempo, existieron numerosos cálculos de sustituciones explícitas, pero ninguno logró al mismo tiempo tener todas las propiedades enumeradas en 1.7.1. Generalmente lo que ocurría era que, o bien se tenía PSN, o bien MC (ejemplos hay muchos: $\lambda\sigma_{\Uparrow}$ [CHL96], λ_{υ} [eaBBL^{+}95], $\lambda\mathrm{s}_e$ [KR97], por nombrar algunos). O, en algunos casos, se conseguían cálculos con ambas propiedades, pero que perdían simpleza en la notación (tal es el caso, por ejemplo, de λ_{ω} [DG01]); o bien llegaban a perder incluso la propiedad de simulación de la clásica β-reducción paso por paso (*i.e.,* había una simulación llamada *big-step*): en este aspecto podemos resaltar el caso de λ_{ζ} [Mn96].

El cálculo λex es un cálculo derivado de λxgc que agrega composición de sustituciones. La innovación consiste en una nueva característica: el agregado de una ecuación para cocientar los términos (al estilo de la α-equivalencia). Dicha ecuación permite intercambiar el orden de aplicación de dos sustituciones, siempre y cuando este intercambio no altere la semántica. La ecuación, llamada ecuación C, es:

$$t[x := u][y := v] \;=_{\mathrm{C}}\; t[y := v][x := u] \qquad \text{si } y \notin \mathrm{FV}(u) \wedge x \notin \mathrm{FV}(v)$$

Uno podría preguntarse aquí el porqué de la necesidad de una ecuación en vez de una regla de reducción. La respuesta se torna obvia cuando uno observa lo que ocurriría en caso de ser esto así: supongamos que tenemos $x_1, x_2, x_3, x_4, x_5 \in \mathbb{V} : x_i \neq x_j \iff i \neq j$. Es decir, tenemos cinco variables diferentes. Bajo el λ-cálculo clásico, podemos armar la siguiente reducción:

$$(\lambda x_4.(\lambda x_1.x_2)\, x_3)\, x_5 \;\to_{\beta}\; (\lambda x_4.x_2\{x_1 := x_3\})\, x_5 \;\to_{\beta}$$

$$x_2\{x_1 := x_3\}\{x_4 := x_5\} \underset{x_1 \neq x_2}{=} x_2\{x_4 := x_5\} \underset{x_4 \neq x_2}{=} x_2$$

que está armada a partir de un término fuertemente normalizante, como puede observarse. Ahora, si tuviéramos la ecuación C como una regla de reducción, podríamos armar una reducción infinita en λex:

$$(\lambda x_4.(\lambda x_1.x_2)\, x_3)\, x_5 \underset{\text{(Beta)}}{\to_{\lambda\text{ex}}} (\lambda x_4.x_2[x_1 := x_3])\, x_5 \underset{\text{(Beta)}}{\to_{\lambda\text{ex}}}$$

$$x_2[x_1 := x_3][x_4 := x_5] \underset{\text{(C), } x_4 \neq x_3 \wedge x_1 \neq x_5}{\to_{\lambda\text{x}}} x_2[x_4 := x_5][x_1 := x_3]$$

$$\underset{\text{(C), } x_4 \neq x_3 \wedge x_1 \neq x_5}{\to_{\lambda\text{x}}} x_2[x_1 := x_3][x_4 := x_5] \underset{\text{(C), } x_4 \neq x_3 \wedge x_1 \neq x_5}{\to_{\lambda\text{x}}} \cdots$$

perdiendo inmediatamente PSN. Este es el motivo subyacente de la inclusión de dicha ecuación. Intuitivamente, lo que ocurre es que las sustituciones son independientes. Por ende, puede intercambiarse libremente el orden de aplicación de las mismas cuantas veces se desee, sin alterar el resultado de la derivación. Es por eso que se decide decir, directamente, que los términos son "los mismos". En el caso en que las sustituciones no sean independientes, sí se utiliza una regla de reducción para realizar la composición. Esta nueva regla, (Comp), es el otro elemento distintivo de λex con respecto a λxgc.

Como último detalle, hacemos notar que λex no tiene la regla (VarR) de λx y λxgc, pues esta es simulable mediante la regla (GC).

Hecho este pequeño excurso, mostramos las reglas del cálculo λex en la figura 1.3, no sin antes mencionar que, al igual que para λxgc, los términos y definiciones para variables libres, metasustitución y α-equivalencia son las mismas. Llamamos, por ende, Λx al conjunto de términos de λex. Es importante aclarar que en la regla (Lamb) se asume por convención de variables (*i.e.,* α-equivalencia) la condicion $y \neq x \wedge y \notin \text{FV}(v)$. Lo análogo vale para la regla (Comp).

(EqC)	$t[x := u][y := v]$	$=$	$t[y := v][x := u]$	$(y \notin \text{FV}(u) \wedge x \notin \text{FV}(v))$
(Beta)	$(\lambda x.t)\, u$	$\to$	$t[x := u]$	
(App)	$(t\, u)[x := v]$	$\to$	$t[x := v]\, u[x := v]$	
(Lamb)	$(\lambda y.t)[x := v]$	$\to$	$\lambda y.t[x := v]$	
(Var)	$x[x := v]$	$\to$	v	
(GC)	$t[x := v]$	$\to$	t	$(x \notin \text{FV}(t))$
(Comp)	$t[x := u][y := v]$	$\to$	$t[y := v][x := u[y := v]]$	$(y \in \text{FV}(u))$

Figura 1.3: Reglas del cálculo λex

De manera similar a λx, llamamos x al conjunto de reglas (App), (Lamb), (Var), (GC) y (Comp); y Bx al conjunto de reglas (Beta) + x. Llamamos C (notada $=_{\text{C}}$) a la relación de equivalencia más pequeña y compatible con contextos generada por la ecuación C. Por último, las relaciones ex y λex son generadas por las relaciones x y Bx módulo α + C equivalencia, respectivamente. Es decir,

$$\forall\, t, t' \in \Lambda\text{x} : t \to_{\text{ex}} t' \iff (\exists\, s, s' \in \Lambda\text{x} : t =_{\alpha+C} s \wedge s \to_{\text{x}} s' \wedge s' =_{\alpha+C} t')$$
$$\forall\, t, t' \in \Lambda\text{x} : t \to_{\lambda\text{ex}} t' \iff (\exists\, s, s' \in \Lambda\text{x} : t =_{\alpha+C} s \wedge s \to_{\text{Bx}} s' \wedge s' =_{\alpha+C} t')$$

De esta forma, el cálculo λex es el sistema de reducción $(\Lambda\mathrm{x}, \lambda\mathrm{ex})$.

Como ya se mencionó anteriormente, este cálculo posee todas las propiedades mencionadas en la introducción de la presente sección, con el agregado de una nueva, *full composition*, que mencionaremos fugazmente más adelante, y que por ello presentamos a continuación.

Definición 1.59 (Full Composition)**.** Decimos que un cálculo de sustituciones explícitas $(\Lambda_\mathrm{Z}, \lambda_\mathrm{Z})$, con Z su cálculo de sustituciones explícitas asociado, posee la propiedad de *full composition* si y sólo si:

$$\forall\, t, u \in \Lambda_\mathrm{Z} : t[x := u] \ \xrightarrow{+}_\mathrm{z}\ t\{x := u\}$$

En donde $t[x := u]$ es un término con sustitución explícita, y $t\{x := u\}$ es la construcción que denota sustitución implícita. Cabe destacar que, si bien aquí utilizamos nombres de variables, esta propiedad aplica también a cálculos de sustituciones explícitas con índices. Informalmente, la noción de *full composition* es la de poder, a partir de cualquier término con sustitución explícita y mediante el subcálculo Z, llegar al término que resultaría de aplicar la sustitución instantánea análoga.

Lema 1.60 (Full Composition para λex)**.** El cálculo λex posee la propiedad de *full composition*. Es decir,

$$\forall\, t, u \in \Lambda\mathrm{x} : t[x := u] \xrightarrow{+}_\mathrm{ex} t\{x := u\}$$

Demostración. Ver lema 2.2 de [Kes09] □

Extensión de λex a términos abiertos

El cálculo λex posee, entre otras, la propiedad de *metaconfluencia*. Esto es: el cálculo extendido a términos con metavariables resulta ser confluente. Dicha extensión es, en el caso particular de λex, con metavariables *anotadas* o *decoradas*. Es decir, cada metavariable posee cierta información; en este caso, un conjunto de variables (que representará el conjunto de variables libres posibles de la metavariable).

En este pequeño apartado presentaremos el cálculo extendido, ya que lo usaremos en la sección 6.3.

A continuación definimos el conjunto de metatérminos sobre el que opera la mentada extensión.

Definición 1.61 (Conjunto de metatérminos del cálculo λex)**.** El conjunto de metatérminos de λex, denotado $\Lambda\mathrm{x}_\mathrm{op}$, está dado por:

$$t ::= x \mid t\,t \mid \lambda x.t \mid t[x := t] \mid \mathrm{X}_\Delta \qquad \text{con } x \in V,\ \mathrm{X} \in \mathbb{M},\ \Delta \in \mathcal{P}(V)$$

con $\mathbb{M}$ un conjunto infinito numerable de metavariables $\{\mathrm{X}, \mathrm{Y}, \mathrm{Z}, \ldots\}$ y Δ finito.

Dada esta definición, extendemos ahora las nociones de variables libres, metasustitución y α-equivalencia. Cabe destacar que sólo definiremos lo que ocurre para metavariables. El resto de los casos se mantienen igual, y las definiciones pueden encontrarse en la sección 1.7.2. Recalcamos, a su vez, el agregado de una noción de metasustitución dentro de un conjunto de variables.

Definición 1.62 (Metasustitución dentro de un conjunto de variables)**.** La metasustitución dentro de un conjunto de variables, $\bullet\{\bullet := \bullet\} : \mathcal{P}(V) \times V \times \Lambda\mathrm{x}_{\mathrm{op}} \to \mathcal{P}(V)$ se define como:

$$\begin{array}{lcll} \Delta\{x := v\} & = & (\Delta - \{x\}) \cup \mathrm{FV}(v) & \text{si } x \in \Delta \\ \Delta\{x := v\} & = & \Delta & \text{si } x \notin \Delta \end{array}$$

Nota. *Es fácil ver que para todo $\Delta \in \mathcal{P}(V)$, y para todo $x, y \in V$,*

$$\Delta\{x := y\} = \{z\{x := y\} : z \in \Delta\}$$

Definición 1.63 (Variables libres de un metatérmino)**.** Extendemos el conjunto de variables libres, $\mathrm{FV} : \Lambda\mathrm{x}_{\mathrm{op}} \to \mathcal{P}(V)$, como:

$$\mathrm{FV}(\mathrm{X}_\Delta) \quad = \quad \Delta$$

Definición 1.64 (Metasustitución)**.** Extendemos la metasustitución, $\bullet\{\bullet := \bullet\} : \Lambda\mathrm{x}_{\mathrm{op}} \times V \times \Lambda\mathrm{x}_{\mathrm{op}} \to \Lambda\mathrm{x}_{\mathrm{op}}$ como:

$$\mathrm{X}_\Delta\{x := v\} \quad = \quad \mathrm{X}_{\Delta\{x:=v\}}$$

Definición 1.65 (α-equivalencia)**.** Extendemos $=_\alpha \subseteq \Lambda\mathrm{x}_{\mathrm{op}} \times \Lambda\mathrm{x}_{\mathrm{op}}$ con el caso de metavariable:

$$\mathrm{X}_\Delta =_\alpha \mathrm{X}_\Delta$$

Definidas estas nociones, sólo resta decir que las relaciones x, Bx, ex y λex se definen de igual manera que antes, sólo que operando sobre metatérminos.

Como ya mencionamos, esta extensión resulta ser confluente y mantener el resto de las propiedades que ya poseía λex.

1.7.5 El cálculo λs

Este cálculo, presentado en [KR95], es el primero de tres cálculos de sustituciones explícitas con índices *à la de Bruijn* que mostraremos aquí. A grandes rasgos, es el cálculo más sencillo de este estilo que pueda pensarse, dado que se obtiene – de manera análoga a λx – agregando construcciones para sustituciones a los términos de Λ_{dB} (ver sección 1.6) y orientando las igualdades, tanto de la metasustitución como de la función de *updating*. Cabe destacar que λs no posee composición de sustituciones, motivo que lo lleva a no tener la propiedad MC. Una de las motivaciones de su presentación fue, justamente, introducir un formalismo sin composición que poseyera PSN, de modo tal de ver a futuro la posibilidad de admisión de una extensión que fuera efectivamente MC sin perder el resto de las bondades. Veamos el cálculo.

Definición 1.66 (Conjunto de términos de λs)**.** El conjunto de términos de λs, denotado Λs, está dado por:

$$a ::= n \mid a\, a \mid \lambda a \mid a\ \sigma^n\ a \mid \varphi^i_k\, a \qquad n, i \in \mathbb{N}_{>0},\ k \in \mathbb{N}$$

Nota. *El operador σ^i es la explicitación de la metasustitución de λ_{dB}, mientras que el operador φ^i_k es la explicitación de la función de* updating. *Para más detalles, referimos al lector a [KR95].*

Las reglas del cálculo λs pueden verse en la figura 1.4. Al conjunto completo de reglas se lo llama λs, mientras que el conjunto de reglas s está dado por $\lambda\mathrm{s} - \{\sigma\text{-generation}\}$. Como de costumbre, las relaciones de reducción λs y s se definen como la clausura contextual de los conjuntos de reglas correspondientes. El cálculo λs es el sistema de reducción $(\Lambda\mathrm{s}, \lambda\mathrm{s})$.

(σ-generation)	$(\lambda a)\, b$	$\to$	$a\ \sigma^1\, b$
(σ-λ-transition)	$(\lambda a)\ \sigma^i\, b$	$\to$	$\lambda(a\ \sigma^{i+1}\, b)$
(σ-app-transition)	$(a_1\ a_2)\ \sigma^i\, b$	$\to$	$(a_1\ \sigma^i\, b)\,(a_2\ \sigma^i\, b)$
(σ-destruction)	$n\ \sigma^i\, b$	$\to$	$\begin{cases} n-1 & \text{si } n > i \\ \varphi_0^i\, b & \text{si } n = i \\ n & \text{si } n < i \end{cases}$
(φ-λ-transition)	$\varphi_k^i\ (\lambda a)$	$\to$	$\lambda(\varphi_{k+1}^i\ a)$
(φ-app-transition)	$\varphi_k^i\ (a_1\ a_2)$	$\to$	$(\varphi_k^i\ a_1)\,(\varphi_k^i\ a_2)$
(φ-destruction)	$\varphi_k^i\ n$	$\to$	$\begin{cases} n+i-1 & \text{si } n > k \\ n & \text{si } n \leq k \end{cases}$

Figura 1.4: Reglas del cálculo λs

1.7.6 El cálculo $\lambda \mathrm{s}_e$

Este formalismo, introducido en [KR97], es una extensión de λs, concebida mediante el agregado de composición de sustituciones (a través de una serie de lemas de interacción entre metasustituciones y *updatings* para λ_{dB}). Originalmente, se esperaba que estos agregados lograran MC, manteniendo PSN. Pero, si bien se logra efectivamente obtener metaconfluencia, en [Gui00] se muestra que la extensión pierde la preservación de la normalización fuerte. A su vez, es hasta ahora un problema abierto $\mathrm{SNs_e}$.

Las reglas del cálculo $\lambda \mathrm{s}_e$ son las mismas que las del cálculo λs, más el agregado de las reglas mostradas en la figura 1.5. El conjunto de reglas $\lambda \mathrm{s}_e$ está dado por todas las reglas, y el subcálculo de sustituciones $\mathrm{s_e}$ se conforma mediante $\lambda \mathrm{s}_e - \{\sigma\text{-generation}\}$. Las relaciones de reducción correspondientes son las clausuras contextuales de los conjuntos de reglas, y el cálculo $\lambda \mathrm{s}_e$ es el sistema de reducción $(\Lambda \mathrm{s}, \lambda \mathrm{s}_e)$.

(σ-σ-transition)	$(a\ \sigma^i\, b)\ \sigma^j\, c$	$\to$	$(a\ \sigma^{j+1}\, c)\ \sigma^i\, (b\ \sigma^{j-i+1}\, c)$	(si $i \leq j$)
(σ-φ-transition 1)	$(\varphi_k^i\ a)\ \sigma^j\, b$	$\to$	$\varphi_k^{i-1}\, a$	(si $k < j < k+i$)
(σ-φ-transition 2)	$(\varphi_k^i\ a)\ \sigma^j\, b$	$\to$	$\varphi_k^i\, (a\ \sigma^{j-i+1}\, b)$	(si $k+i \leq j$)
(φ-σ-transition)	$\varphi_k^i\, (a\ \sigma^j\, b)$	$\to$	$(\varphi_{k+1}^i\ a)\ \sigma^j\, (\varphi_{k+1-j}^i\, b)$	(si $j \leq k+1$)
(φ-φ-transition 1)	$\varphi_k^i\, (\varphi_l^j\ a)$	$\to$	$\varphi_l^j\, (\varphi_{k+1-j}^i\ a)$	(si $l+j \leq k$)
(φ-φ-transition 2)	$\varphi_k^i\, (\varphi_l^j\ a)$	$\to$	$\varphi_l^{j+i-1}\, a$	(si $l \leq k < l+j$)

Figura 1.5: Nuevas reglas para el cálculo $\lambda \mathrm{s}_e$

1.7.7 El cálculo λt

En [KR98] se presenta un cálculo de sustituciones explícitas con índices *à la de Bruijn* muy similar a λs: λt. Dicho cálculo se basa en una presentación alternativa de λ_{dB}, cuyo sello principal es un metaoperador de *updating progresivo*. Por progresivo, queremos decir que la actualización de índices no se realiza al

momento de sustituir, sino que se va haciendo paulatinamente al ir atravesando abstracciones. Presentamos aquí este cálculo pues la mentada noción de *updating progresivo* fue de importante inspiración para lo que más adelante mostraremos.

Definición 1.67 (Conjunto de términos de λt)**.** El conjunto de términos de λt, denotado Λt, está dado por:

$$a ::= n \mid a\,a \mid \lambda a \mid a\,\varsigma^n\,a \mid \theta_k\,a \qquad n, i \in \mathbb{N}_{>0},\ k \in \mathbb{N}$$

Nota. *El operador ς^i es la versión explícita de la metasustitución correspondiente a la presentación alternativa λ_{dB}; el operador θ_k, su contraparte para la función de* updating progresivo*. Recomendamos la lectura de [KR98] para quien quiera ahondar más en el tema.*

Las reglas del cálculo pueden apreciarse en la figura 1.6, junto con las principales diferencias respecto de λs (*i.e.,* el *updating progresivo*). Llamamos al conjunto de todas las reglas λt, y t al conjunto de reglas conformado por $\lambda\text{t} - \{\varsigma - \mathit{generation}\}$. Al igual que para los cálculos anteriores, las relaciones de reducción están dadas por la clausura contextual de los conjuntos de reglas. El cálculo λt es el sistema de reducción $(\Lambda\text{t}, \lambda\text{t})$.

(ς-generation)	$(\lambda a)\,b$	$\to$	$a\,\varsigma^1\,b$
(ς-λ-transition)	$(\lambda a)\,\varsigma^i\,b$	$\to$	$\lambda(a\,\varsigma^{i+1}\,(\theta_o\,b))$
(ς-app-transition)	$(a_1\,a_2)\,\varsigma^i\,b$	$\to$	$(a_1\,\varsigma^i\,b)\,(a_2\,\varsigma^i\,b)$
(ς-destruction)	$n\,\varsigma^i\,b$	$\to$	$\begin{cases} n-1 & \text{si } n > i \\ b & \text{si } n = i \\ n & \text{si } n < i \end{cases}$
(θ-λ-transition)	$\theta_k\,(\lambda a)$	$\to$	$\lambda(\theta_{k+1}\,a)$
(θ-app-transition)	$\theta_k\,(a_1\,a_2)$	$\to$	$(\theta_k\,a_1)\,(\theta_k\,a_2)$
(θ-destruction)	$\theta_k\,n$	$\to$	$\begin{cases} n+1 & \text{si } n > k \\ n & \text{si } n \leq k \end{cases}$

Figura 1.6: Reglas del cálculo λt

Cabe destacar que λt tiene las propiedades (CRt), (SNt), (Sim), (Snd), (PSN) y (CR).

Capítulo 2

Ideas iniciales

2.1 Introducción

En este capítulo haremos una presentación de carácter informal e intuitivo acerca de las ideas preliminares más importantes que surgieron a lo largo del trabajo, y que luego llevaron a lo que finalmente desarrollamos; a saber: dos cálculos de sustituciones explícitas.

Comenzaremos con un formalismo relativamente *sui generis*, que parte de la idea de agregar términos con *agujeros* para denotar lugares en donde se ha de realizar la operación de sustitución, tomando como premisa una presunta falla conceptual en λ_{dB} (ver sección 1.6 de este trabajo).

En segundo y último lugar, mostramos un cálculo al estilo de λs (ver sección 1.7.5 de este trabajo, o bien [KR95]), para el que intentamos concebir una forma sencilla de composición (en contrapartida a las complicadas reglas que vemos en la extensión de λs, $\lambda\mathrm{s}_e$ - ver sección 1.7.6 de este trabajo, y remitirse a [KR97] para mayor introspección), sumado al agregado de *Garbage Collection* y de *updatings* en el metalenguaje. Es de importancia aclarar que la construcción de dicho cálculo estuvo guiada por un intento de acercamiento a λex (ver sección 1.7.4 de este trabajo, y [Kes08, Kes09] para más detalles); y que, si bien no tuvimos éxito con el cálculo per se, su planteo nos guió hasta lo que finalmente es nuestro trabajo.

Es importante destacar que, durante todo el capítulo, haremos uso de la notación genérica $a[n := b]$ para denotar la construcción de sustitución explícita con índices *à la de Bruijn*, con el objetivo de simplificar la lectura, y dado que no hay diferencias de índole semántica con otras notaciones existentes.

2.2 Un intento diferente: ¡agujeros!

La idea para este cálculo surgió de una observación con respecto a λ_{dB}. En este último formalismo, recordemos, la β-reducción se define como:

$$(\lambda a)\, b \to a\{\{1 \leftarrow b\}\}$$

Ahora bien: si pensamos la metasustitución como no atómica, tenemos que, en el lado derecho de la expresión de arriba, a no se modifica como consecuencia de la desaparición de la abstracción. La modificación correspondiente a este término

queda en manos de la sustitución, que realiza el decremento correspondiente de las variables. Conceptualmente, es raro que la sustitución tenga la *responsabilidad* de realizar el decremento de índices, pasando parte de esta tarea también a la función de *updating* (recordar que el índice i representa aumentar los índices en $i-1$, dada la remoción de una abstracción). Es decir, la sustitución nunca es un instrumento aislado del cálculo, sino que se asocia fuertemente a una β-reducción.

Con esta idea en mente, es que pensamos en un cálculo que lograra disociar el concepto de remoción de abstracciones del de sustitución, de forma tal de ver si la mentada disociación era una causa de la dificultad de obtención de metaconfluencia y preservación de la normalización fuerte al mismo tiempo.

Para lograr la implementación de esta separación de conceptos, concebimos un cálculo con dos tipos de índices: los clásicos y los *agujeros*. La función de los índices clásicos es la ya conocida, mientras que los agujeros ofician de "contextos" en los cuales puede sustituirse un término. Así, los términos, denotados por $\Lambda_\circ$, se definen como:

Definición 2.1 (Conjunto de términos $\Lambda_\circ$)**.** El conjunto de términos $\Lambda_\circ$ está dado por:

$$a ::= n \mid \mathring{n} \mid a\,a \mid \lambda a \mid a[\mathring{n} := a] \quad (n \in \mathbb{N}_{>0})$$

De la definición, podemos observar que existen las dos clases de índices, y que la sustitución explícita se realiza sobre agujeros, en vez de sobre índices clásicos. Las variables libres pueden ser tanto índices como agujeros, extendiéndose naturalmente la definición que se utiliza en el clásico λ_{dB} (ver sección 1.6). Introducimos dos nuevos metaoperadores para el trabajo con los términos. El primero – ϕ_k^i, $k \in \mathbb{N}_{>0}$, $i \in \mathbb{Z}$ – se encarga de hacer las actualizaciones de índices (permitiendo también decrementos), sin alterar los "agujeros". El segundo – Υ_i, $i \in \mathbb{N}_{>0}$ – , que denominamos de *generación de agujero*, cumple la función de, dado un término a, reemplazar todas las variables libres i en a por el agujero $\mathring{i}$; y a todos los agujeros mayores a $\mathring{i}$, incrementarlos en uno (para evitar *clashes*). A continuación definimos formalmente el operador de actualización de índices.

Definición 2.2 (Metaoperador ϕ_k^i)**.** Para todo $k \in \mathbb{N}_{>0}$, $i \in \mathbb{Z}$, $\phi_k^i : \Lambda_\circ \to \Lambda_\circ$ se define inductivamente como:

$$\begin{aligned}
\phi_k^i(n) &= \begin{cases} n+i & \text{si } n > k \wedge n+i > 0 \\ n & \text{si } n \leq k \vee n+i \leq 0 \end{cases} \\
\phi_k^i(\mathring{n}) &= \mathring{n} \\
\phi_k^i(a\,b) &= \phi_k^i(a)\ \phi_k^i(b) \\
\phi_k^i(\lambda a) &= \lambda \phi_{k+1}^i(a) \\
\phi_k^i(a[\mathring{n} := b]) &= \phi_k^i(a)\,[\mathring{n} := \phi_k^i(b)]
\end{aligned}$$

Por último, mostramos la definición del operador de generación de agujero.

Definición 2.3 (Metaoperador Υ_i)**.** Para todo $i \in \mathbb{N}_{>0}$, $\Upsilon_i : \Lambda_\circ \to \Lambda_\circ$ se define inductivamente como:

$$\begin{array}{rcl}
\Upsilon_i(n) & = & \begin{cases} n-1 & \text{si } n > i \\ \mathring{n} & \text{si } n = i \\ n & \text{si } n < i \end{cases} \\
\Upsilon_i(\mathring{n}) & = & \begin{cases} n \mathbin{\mathring{+}} 1 & \text{si } n > i \\ \mathring{n} & \text{si } n \leq i \end{cases} \\
\Upsilon_i(a\,b) & = & \Upsilon_i(a)\ \Upsilon_i(b) \\
\Upsilon_i(\lambda a) & = & \lambda \Upsilon_{i+1}(a) \\
\Upsilon_i(a[\mathring{n} := b]) & = & \begin{cases} \Upsilon_i(a)\,[n \mathbin{\mathring{+}} 1 := \phi_0^{-1}(b)] & \text{si } n > i \\ \Upsilon_i(a)\,[\mathring{n} := \phi_0^{-1}(b)] & \text{si } n \leq i \end{cases}
\end{array}$$

Además de la mentada separación de conceptos, otro interés fue el de concebir reglas de composición sencillas para todos los casos (*i.e.,* tanto sustituciones dependientes como independientes), y que a la vez pudieran evitar la necesidad de trabajar módulo una ecuación, como sucede en λex [Kes08, Kes09] (ver subsección 1.7.4). Como podrá apreciarse, esto sí se logró (aunque el cálculo en definitiva no funciona, como veremos más adelante).

Las reglas del sistema (al que no le pusimos nombre) pueden verse en la figura 2.1.

(Beta)	$(\lambda a)\,b$	$\to$	$\Upsilon_1(a)\,[\mathring{1} := b]$	
(App)	$(a\,b)[\mathring{n} := c]$	$\to$	$a[\mathring{n} := c]\,a[\mathring{n} := c]$	
(Lamb)	$(\lambda a)[\mathring{n} := c]$	$\to$	$\lambda a[n \mathbin{\mathring{+}} 1 := \phi_0^1(c)]$	
(Var)	$\mathring{n}[\mathring{n} := c]$	$\to$	c	
(GC)	$a[\mathring{n} := c]$	$\to$	a	$(\mathring{n} \notin \mathrm{FV}(a))$
($\mathrm{Comp_d}$)	$a[\mathring{n} := c][\mathring{m} := d]$	$\to$	$a[\mathring{m} := d][\mathring{n} := c[\mathring{m} := d]]$	$(\mathring{m} \in \mathrm{FV}(c))$
($\mathrm{Comp_i}$)	$a[\mathring{n} := c][\mathring{m} := d]$	$\to$	$a[\mathring{m} := d][\mathring{n} := c]$	$(\mathring{m} \notin \mathrm{FV}(c) \wedge m < n)$

Figura 2.1: Reglas del cálculo con "agujeros"

Ahora, si bien todo parecía funcionar correctamente, vimos que el clásico ejemplo de divergencia (*i.e.,* sustitución en β-reducto) no cerraba. El ejemplo concreto era:

$$((\lambda a)\,b)[\mathring{n} := c] \quad \to_{\mathrm{Beta}} \quad \Upsilon_1(a)\,[\mathring{1} := b][\mathring{n} := c] \tag{2.1}$$

$$((\lambda a)\,b)[\mathring{n} := c] \quad \twoheadrightarrow_{\mathrm{App,\ Lamb,\ Beta}} \quad \Upsilon_1(a)\,[n \mathbin{\mathring{+}} 2 := c][\mathring{1} := b[\mathring{n} := c]] \tag{2.2}$$

que, como mencionamos, resulta no cerrar, pues en (2.1) tenemos el índice $\mathring{n}$; mientras que en (2.2) figura $n \mathbin{\mathring{+}} 2$. Lo que ocurrió es que, en (2.1), la regla (Beta) provoca una independización del cruce de información (cosa que no ocurre en (2.2)). Es decir: la sustitución externa de (2.1) *nunca se enteró* de la nueva β-reducción que ocurrió. Así, se produce una inconsistencia entre los agujeros de un término y las sustituciones, haciéndose imposible cerrar el contraejemplo. Luego de ocurrido el inconveniente, decidimos tomar otro camino. Sin embargo,

nos parece interesante estudiar a futuro qué pasaría en el caso de implementar algún sistema de intercambio de información, al estilo de λ_ς [Mn96], que subsane el inconveniente aludido.

2.3 Un intento al estilo de λs

Como ya mencionamos en la introducción, la idea aquí fue intentar una aproximación a λex tratando de lograr, al mismo tiempo, un cálculo con índices cuyas reglas de reducción no involucraran actualizaciones complicadas de éstos, tal y como sí ocurre en λs_e.

Para aproximarnos a nuestro objetivo, tomamos una serie de decisiones de diseño:

1. Utilizar una versión extendida a $\mathbb{N}$ (incluyendo el 0) de la función de actualización de λ_{dB}: U_k^i, con $i, k \in \mathbb{N}$. La ventaja que traería aparejada la posesión de este operador extendido, que estaría en el metalenguaje, sería la de poder realizar decrementos de índices.

2. Manejar una actualización progresiva de índices, tal y como se hace en λt (sección 1.7.7 de este trabajo, y [KR98]). Así, por ejemplo, la regla de reducción que se aplicaría para cruzar una abstracción, quedaría de la forma:
$$\text{(Lambda)} \quad (\lambda a)[n := c] \to \lambda a[n+1 := \mathrm{U}_0^2(c)]$$
La ventaja de hacer esto, es que en cualquier término de la forma $a[n := b]$, b esté *siempre* listo para ser reemplazado en a, sin necesidad de una actualización previa. Luego, la regla (Var) quedaría como la vemos a continuación:
$$\text{(Var)} \quad n[n := c] \to c$$

3. Agregar una regla de *Garbage Collection*, de modo tal de facilitar la obtención de metaconfluencia, tal y como se hace en λex. La regla quedaría así:
$$\text{(GC)} \quad a[n := c] \to \mathrm{U}_n^0(a) \quad (n \notin \mathrm{FV}(a))$$
Como se puede apreciar, dado que no hay una regla que decremente las variables luego de la desaparición de una abstracción (producto a su vez de una β-reducción), es necesario hacer lo propio aquí, utilizando el metaoperador de actualización extendido. Decimos que no hay una regla para decrementar las variables pues consideramos un cálculo sin contrapartida de la regla (Var).

4. En caso de llegar a una regla de composición razonable, condicionarla a cierto orden entre los índices (como puede apreciarse en el cálculo con "agujeros", sección 2.2). Generalizando, la regla tendría la forma:
$$a[i := b][j := c] \to a'[j' := c'][i' := b'] \qquad \text{si } i\ \theta\ j, \text{ con } \theta \in \{<, >\}$$
De esta manera, pensamos, iba a ser posible evitar la ecuación que se necesita en λex para tener *full composition* (ver definición 1.59).

Como primer paso, y luego de un esbozo inicial, intentamos ver que continuaran valiendo las identidades a partir de las cuales surgen las reglas del cálculo λs_e (*i.e.,* lemas 5 a 10 de [KR97]), pues habíamos extendido la función de actualización. El resultado fue bastante bueno: los lemas 5 a 8 continuaban valiendo; mientras que los lemas 9 y 10 valían sólo para el caso estricto (*i.e.,* $l + j < k + 1$ y $n < k + 1$, respectivamente).

Con estos lemas en mano, intentamos definir de manera *completa* la distribución de la función de *updating* para el caso de sustitución (cosa que no está hecha en λs_e, y que nos daba un punto de partida interesante). Esto es, ver de qué forma definir:

$$\mathrm{U}_k^i(a[n := b])$$

Luego de algún trabajo, decidimos volver a utilizar la función de *updating* clásica y agregar un nuevo metaoperador que sirviera exclusivamente para el decremento de índices, pues nos ocurrió que para un caso particular, existía un término para el cual no podíamos resolver un *clash* que se producía (y para el que parece no haber solución). Bajo estas nuevas premisas, logramos definir de forma completa la distribución de los updatings dentro de una sustitución. La definición, para el lector interesado, es:

$$\mathrm{U}_k^i(a[n := b]) = \begin{cases} \mathrm{U}_{k+1}^i(a)[n := \mathrm{U}_{k-n+1}^i(b)] & \text{si } n \leq k+1 \\ \mathrm{U}_k^i(a)[n+i-1 := b] & \text{si } n > k+1 \end{cases}$$

El siguiente paso fue la definición de la composición de sustituciones, también en forma *completa*. Esto es, de qué forma manejar términos de la forma:

$$a[i := b][j := c]$$

Lamentablemente aquí, luego de varios intentos, llegamos a la conclusión de que no es posible definir tal cosa para todos los casos, pues se produce inevitablemente un *clash* entre índices. Ahora bien, "jugando" con algunos ejemplos, llegamos a intentar mapear directamente los términos de nuestro cálculo con los de λex. Es decir, buscamos una *traducción*. Para esto, nos inspiramos en la que se muestra en [KR98], utilizada con el fin de probar el isomorfismo existente entre los cálculos λ y λ_{dB}. Ahora bien, al tratar de definir la traducción para el caso de sustitución explícita, llegamos a la conclusión de que era imposible definirla de forma análoga a la original. Para ver por qué, veamos el caso de abstracción de la traducción mostrada en el citado artículo:

$$\mathrm{w}_{[x_1,\ldots,x_n]}(\lambda x.t) = \lambda \mathrm{w}_{[x,x_1,\ldots,x_n]}(t)$$

Lo que ocurre aquí es que la lista asociada a la traducción oficia de "indexadora" de variables. Es decir, el índice que le corresponde a una variable particular se obtiene buscando su posición en la lista. Al abstraer, todas las variables deben sumar uno a sus índices, y la variable abstraída pasa a ser representada por el índice 1. Esto es exactamente lo que hace la traducción mostrada. Ahora bien, veamos el caso de sustitución explícita:

$$\mathrm{w}_{[x_1,\ldots,x_n]}(t[x := u])$$

Para comprender la problemática recalquemos, como punto de partida, que este estilo de traducciones se basa fuertemente en el *binding* impuesto para un término a partir de la notación de *de Bruijn*. Ahora bien, en los términos $a[i := b]$,

además del *binder*, hay un índice que termina *imponiendo* el *binding*, por sobre la construcción original de clausura. Por ende, se hace necesario determinar el valor de i al traducir desde un término de la forma $t[x := u]$, de manera tal de insertar a x en dicha posición de la lista $[x_1, \ldots, x_n]$ (no podríamos obrar como en el caso de la abstracción, asignando 1 al índice siempre, pues no obtendríamos una biyección). Aquí nos preguntamos: ¿es necesario inspeccionar t para determinar el índice de x, cosa que no ocurre para el caso de abstracción? De ser necesario, ¿qué ocurriría si x no figura libre en t? ¿cómo sería posible obtener el índice correspondiente? ¿se hace necesario tener información sobre el "pasado" del término a traducir, en lo que a atravesar *binders* respecta? Si bien no respondimos a estas preguntas, ni ideamos una traducción funcional, el interrogante mismo nos permitió llegar a lo que finalmente es nuestro desarrollo, y que veremos en el capítulo siguiente.

Capítulo 3

Cálculo λr

3.1 Otra presentación de λ_{dB}: λr

3.1.1 Introducción

Esta presentación sirve como introducción a tres cálculos de sustituciones explícitas que mostraremos más adelante. La utilidad del desarrollo radica en la posibilidad de mostrar algunas propiedades que posteriormente guiarán la definición de los ya mentados cálculos. Como bien indica el nombre de la sección, mostramos aquí una versión alternativa al λ-cálculo clásico con índices *à la de Bruijn*.

3.1.2 Ideas intuitivas detrás de la presentación

Como ya mencionamos en 1.6, la idea principal detrás del λ-cálculo con índices *à la de Bruijn* [dB72] es la de eliminar la necesidad de lidiar con la α-equivalencia, de manera tal de simplificar las implementaciones. Así, uno obtiene un cálculo isomorfo al λ-cálculo clásico, pero con la ventaja de que términos α-equivalentes en éste tienen la misma representación con la notación de *de Bruijn*. Por ejemplo, los términos α-equivalentes $\lambda x.x$ y $\lambda y.y$ se representan en λ_{dB} como $\lambda 1$. De esta forma, uno normalmente se refiere a λ_{dB} como un cálculo sin nombres, o *namefree/nameless*.

A nuestro entender, esta noción de *namefree*-calculus resulta *ser cierta* en el caso de λ_{dB}, puesto que la sustitución se encuentra en el *metalenguaje*. Sin embargo, en cálculos con sustituciones explícitas e índices de *de Bruijn*, como por ejemplo λs [KR95], λs_e [KR97] o λt [KR98] – entre otros –, nos tomamos el atrevimiento de afirmar que esto no es cierto. Es decir, este tipo de cálculos no son ya *nameless*. Dicha aserción proviene del hecho de que estos cálculos tienen, además de las clásicas aplicación y abstracción, construcciones para denotar sustituciones que poseen un índice que indica qué variable sustituir. Por ende, y unificando notación, pasamos a tener términos del estilo:

$$a[i := b]$$

Aquí, si bien i no es un nombre per se, sino un índice, en realidad está cumpliendo la función de nombre, sólo que nominalmente oculto a la vista. Cumple la función de nombre justamente, pues está indicando qué variable del término a

debe reemplazarse por b. Esta noción de no-completamente-*namefree*-calculus surgió de la dificultad de definir una traducción desde λx de estos términos (dificultad que puede apreciarse, también, en [KR98]). Intuitivamente, lo que ocurre es que para el caso de sustitución explícita, a diferencia del caso de abstracción (en donde el binder **no** está nombrado), es necesario conocer de antemano la profundidad (en términos de índices) de la variable a sustituir, de forma tal de definir el índice a colocar en el término traducido (ver sección 2.3).

El problema mencionado derivó en un intento de lograr un isomorfismo con cálculos del estilo de λx [Blo95, Blo97], λxgc [BR95] o λex [Kes08, Kes09]. Para esto, el primer paso fue quitar los índices de los términos con sustituciones explícitas, de manera tal de lograr un cálculo verdaderamente *nameless*. Así, quedamos con términos de la forma:

$$a[b]$$

y con una Beta-regla que pasa de ser:

$$(\lambda a)\, b \to a[1 := b]$$

a tener la forma:

$$(\lambda a)\, b \to a[b]$$

siendo el significado de $a[b]$ el de reemplazar las ocurrencias libres de la variable 1 por b en a. Esta formulación, si bien sencilla, presenta dos problemas, surgidos del hecho de que, al abstraer un término, todas sus variables libres se ven incrementadas en uno. Veamos dichos inconvenientes:

1. La necesidad de incrementar las variables libres de b al atravesar una abstracción con la sustitución, dado que, al no tener un índice que indique la cantidad de abstracciones atravesadas, no es posible hacer un incremento acumulativo al eliminar la sustitución.

2. La necesidad de poseer una herramienta que logre efectivamente eliminar el requerimiento de poseer un índice en la sustitución, puesto que al atravesar una abstracción, las variables i cambian por $i + 1$, y en nuestra notación, no es posible expresar la sustitución de un término que no sea 1.

El primero de los problemas descriptos se soluciona con la introducción de un operador de incremento de índices que aumente las variables en uno, y que se aplique al atravesar una abstracción. El segundo es un poco más complicado. Para explicar lo que hacemos, cambiemos por un momento la notación de sustitución explícita:

$$a[b] \qquad \text{por} \qquad \sigma^b a$$

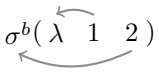
Veamos con un ejemplo la idea para la solución. Supongamos que tenemos el término:

$$(\lambda 1\, 2)[b]$$

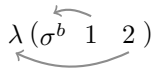
que en la notación temporaria describimos como:

$$\sigma^b(\lambda 1\, 2)$$

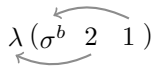
Notemos que, en este último término, la variable 1 "apunta" a la abstracción, y la variable 2, a la sustitución:



Cuando atravesamos la abstracción con la sustitución, se invierte el orden:



con lo que, de no hacer nada al respecto, estaríamos cambiando la semántica. Para solucionar esto, simplemente hacemos un intercambio de variables: la 1 la cambiamos por la 2, y viceversa. Tenemos, entonces:



recuperando – a priori – el significado original del término. Por ende, esto es justamente lo que hacemos al atravesar una abstracción: intercambiamos las variables libres 1 por 2 en el término abstraido, dejando el resto de las variables como están, e incrementando – como ya se mencionó – los índices del término a sustituir (b en el ejemplo) en uno. Cabe destacar que las variables libres i pasan a ser representadas por $i + 1$ al colocarse una abstracción más por delante. Es por esto que introducimos un nuevo operador (que llamaremos de *swap*, y notaremos $\updownarrow_i$, presentado en la siguiente sección) que se encarga de hacer esto mismo: intercambiar las variables libres i por $i+1$ en un término. De la misma manera, introducimos otro operador, que llamaremos de incremento de índices, y notaremos $\uparrow_i$, que cumple la función incrementar las variables libres de un término en uno.

Con las dos nociones presentadas (de *swap* y de incremento de índices), pareciera ser que – intuitivamente, al menos – recuperamos la semántica del λ-cálculo con índices *à la de Bruijn*, sin necesidad de utilizar índices en las sustituciones que indiquen dónde sustituir. Es decir, lograríamos un cálculo que, de explicitarse la sustitución, sería realmente *nameless*.

Formalizamos estas nociones intuitivas en la siguiente sección, y probamos más adelante que, efectivamente, la noción es correcta. Es decir, un cálculo de este estilo con sustitución implícita (*i.e.,* en el metalenguage) no es nada más ni nada menos que una presentación diferente para el λ-cálculo con índices *à la de Bruijn*.

Para terminar con estas ideas preliminares, es importante agregar que un cálculo de sustituciones explícitas derivado de las ideas recién presentadas no sería el primer cálculo *nameless* en la historia, puesto que podemos encontrar cálculos de este tipo en la literatura. Tal es el caso, por ejemplo, de $\lambda\upsilon$ [eaBBL$^+$95]. A su vez, la noción de *nameless* que acabamos de discutir podría perderse en caso de agregar reglas condicionales que sí hablen de índices. De hecho, esto es lo que ocurre en nuestro caso particular al agregar *Garbage Collection* al primer cálculo de sustituciones explícitas derivado del que presentamos aquí (ver capítulo 5). Sin embargo, y aunque se "pierda" esta noción purista de *nameless* (si bien ésta se mantiene en lo que a estructura de términos refiere), el enfoque que tomamos resulta de suma utilidad para la obtención de lo que es el cálculo más relevante del trabajo: λrex (ver capítulo 6).

3.1.3 Definición de λr

Como se explicó en la sección anterior, los términos de λr son los mismos que los de λ_{dB}, siendo la diferencia entre los dos cálculos la manera en que se realiza la sustitución (que en estos cálculos está definida en el metalenguaje). Denotamos, por ende, al conjunto de términos de λr igual que al conjunto de términos de λ_{dB}, es decir: Λ_{dB}.

Definimos, a continuación, los dos metaoperadores que vamos a necesitar para la definición de la metasustitución, a saber: $\uparrow_i$ y $\updownarrow_i$.

El siguiente metaoperador se encarga de sumar uno a las variables libres del término al que es aplicado.

Definición 3.1 (Metaoperador $\uparrow_i$)**.** Para todo $i \in \mathbb{N}$, $\uparrow_i : \Lambda_{\mathrm{dB}} \to \Lambda_{\mathrm{dB}}$ se define inductivamente como:

$$\begin{aligned} \uparrow_i(n) &= \begin{cases} n & \text{si } n \leq i \\ n+1 & \text{si } n > i \end{cases} \\ \uparrow_i(a\,b) &= \uparrow_i(a)\ \uparrow_i(b) \\ \uparrow_i(\lambda a) &= \lambda\, \uparrow_{i+1}(a) \end{aligned}$$

Observación 3.2. Notar que $\uparrow_i(a) = \mathrm{U}_i^2(a)$.

El metaoperador que introducimos a continuación es el operador distintivo del cálculo λr, y se encarga de intercambiar las variables libres i por $i+1$ en el término al que es aplicado.

Definición 3.3 (Metaoperador $\updownarrow_i$)**.** Para todo $i \in \mathbb{N}_{>0}$, $\updownarrow_i : \Lambda_{\mathrm{dB}} \to \Lambda_{\mathrm{dB}}$ se define inductivamente como:

$$\begin{aligned} \updownarrow_i(n) &= \begin{cases} n & \text{si } n < i \ \vee\ n > i+1 \\ i+1 & \text{si } n = i \\ i & \text{si } n = i+1 \end{cases} \\ \updownarrow_i(a\,b) &= \updownarrow_i(a)\ \updownarrow_i(b) \\ \updownarrow_i(\lambda a) &= \lambda\, \updownarrow_{i+1}(a) \end{aligned}$$

Teniendo ya la definición de los anteriores metaoperadores, podemos dar la definición para la metasustitución. Antes de hacerlo, es importante destacar que la idea detrás de $a\{b\}$ es la de sustituir las variables libres 1 en a por b. Para poder hacer esto, y dado que luego de atravesar una abstracción, las variables libres i pasan a ser representadas por $i+1$, es necesario (como ya se explicó en la sección anterior) hacer el *swap* de la variable libre 1 por 2, de modo tal de reflejar que se atravesó una abstracción con la sustitución, y permitir seguir realizando la misma sobre los índices 1. A su vez, y como puede observarse más abajo, al atravesar una abstracción es necesario subir en uno los índices de las variables libres de b. Esto es indispensable, dado que al no poseer un índice dentro de la sustitución que indique la cantidad de abstracciones atravesadas, no es posible realizar una actualización acumulativa de índices al realizar el reemplazo de una variable por el término correspondiente. En este aspecto, podemos observar un comportamiento similar en el cálculo λt [KR98].

Definición 3.4 (Metasustitución para λr)**.** Para todo $a, b, c \in \Lambda_{\mathrm{dB}}$, $n \in \mathbb{N}_{>0}$, $\bullet\{\bullet\} : \Lambda_{\mathrm{dB}} \times \Lambda_{\mathrm{dB}} \to \Lambda_{\mathrm{dB}}$ se define inductivamente como:

$$\begin{array}{rcl} n\{c\} & = & \begin{cases} c & \text{si } n = 1 \\ n-1 & \text{si } n > 1 \end{cases} \\ (a\,b)\{c\} & = & a\{c\}\,b\{c\} \\ (\lambda a)\{c\} & = & \lambda\uparrow_1(a)\{\uparrow_0(c)\} \end{array}$$

Una vez definida la metasustitución, procedemos a definir la relación β_{r} y, posteriormente, el cálculo λr:

Definición 3.5 (β_{r}-reducción)**.** Se define la relación $\beta_{\mathrm{r}} \subseteq \Lambda_{\mathrm{dB}} \times \Lambda_{\mathrm{dB}}$, notada $\to_{\beta_{\mathrm{r}}}$ como:

$$(\forall a, b \in \Lambda_{\mathrm{dB}})\,(a \to_{\beta_{\mathrm{r}}} b \text{ sii } (\exists\ C \text{ contexto};\ c, d \in \Lambda_{\mathrm{dB}})\,(a = \mathrm{C}\,[(\lambda c)d] \wedge b = \mathrm{C}\,[c\{d\}]))$$

Definición 3.6 (Cálculo λr)**.** El cálculo λr es el sistema de reducción $(\Lambda_{\mathrm{dB}}, \beta_{\mathrm{r}})$

3.1.4 Algunos ejemplos del comportamiento de λr

A modo de ilustrar el comportamiento de λr, damos aquí algunos ejemplos.

- Selección del primer argumento:

$$(\lambda\lambda 2)\,3\,4 \to_{\beta_{\mathrm{r}}} ((\lambda 2)\{3\})\,4 \underset{\text{msust.}}{=} (\lambda\uparrow_1(2)\{\uparrow_0(3)\})\,4 \underset{\text{swap}}{=} (\lambda 1\{\uparrow_0(3)\})\,4 \underset{\text{incr.}}{=}$$

$$(\lambda 1\{4\})\,4 \underset{\text{msust.}}{=} (\lambda 4)\,4 \to_{\beta_{\mathrm{r}}} 4\{4\} \underset{\text{msust.}}{=} 3$$

- Selección del segundo argumento:

$$(\lambda\lambda 1)\,3\,4 \to_{\beta_{\mathrm{r}}} ((\lambda 1)\{3\})\,4 \underset{\text{msust.}}{=} (\lambda\uparrow_1(1)\{\uparrow_0(3)\})\,4 \underset{\text{swap}}{=} (\lambda 2\{\uparrow_0(3)\})\,4 \underset{\text{incr.}}{=}$$

$$(\lambda 2\{4\})\,4 \underset{\text{msust.}}{=} (\lambda 1)\,4 \to_{\beta_{\mathrm{r}}} 1\{4\} \underset{\text{msust.}}{=} 4$$

- Un paso de derivación de Ω:

$$(\lambda 1\,1)\,(\lambda 1\,1) \to_{\beta_{\mathrm{r}}} (1\,1)\{\lambda 1\,1\} \underset{\text{msust.}}{=} 1\{\lambda 1\,1\}\,1\{\lambda 1\,1\} \underset{\text{msust.}}{=} (\lambda 1\,1)\,(\lambda 1\,1)$$

- Función constante:

$$(\lambda 8)3 \to_{\beta_{\mathrm{r}}} 8\{3\} \underset{\text{msust.}}{=} 7$$

- Derivación de SKK a I (con $\mathrm{S} = \lambda\lambda\lambda 3\,1\,(2\,1)$, $\mathrm{K} = \lambda\lambda 2$ y $\mathrm{I} = \lambda 1$):

$$\mathrm{SKK} = (\lambda\lambda\lambda 3\,1\,(2\,1))\,(\lambda\lambda 2)\,(\lambda\lambda 2) \to_{\beta_{\mathrm{r}}} (\lambda\lambda 3\,1\,(2\,1))\{\lambda\lambda 2\}\,(\lambda\lambda 2) \underset{\text{msust.}}{=}$$

$$(\lambda\uparrow_1(\lambda 3\,1\,(2\,1))\{\uparrow_0(\lambda\lambda 2)\})\,(\lambda\lambda 2) \underset{\text{swap}}{=}$$

$$(\lambda(\lambda\uparrow_2(3\,1\,(2\,1)))\{\uparrow_0(\lambda\lambda 2)\})\,(\lambda\lambda 2) \underset{\text{swap}}{=}$$

$$(\lambda(\lambda 2\,1\,(3\,1))\{\uparrow_0(\lambda\lambda 2)\})\,(\lambda\lambda 2) \underset{\text{incr.}}{=} (\lambda(\lambda 2\,1\,(3\,1))\{\lambda\lambda 2\})\,(\lambda\lambda 2) \underset{\text{msust.}}{=}$$

$$(\lambda\lambda \uparrow_1(2\,1\,(3\,1))\{\uparrow_0(\lambda\lambda 2)\})\,(\lambda\lambda 2) \underset{\text{swap., incr.}}{=}$$
$$(\lambda\lambda(1\,2\,(3\,2))\{\lambda\lambda 2\})\,(\lambda\lambda 2) \underset{\text{msust.}}{=} (\lambda\lambda(\lambda\lambda 2)\,1\,(2\,1))\,(\lambda\lambda 2) \to_{\beta_{\mathrm{r}}}$$
$$(\lambda(\lambda\lambda 2)\,1\,(2\,1))\{\lambda\lambda 2\} \underset{\text{msust.}}{=} \lambda \uparrow_1((\lambda\lambda 2)\,1\,(2\,1))\{\uparrow_0(\lambda\lambda 2)\} \underset{\text{swap., incr.}}{=}$$
$$\lambda((\lambda\lambda 2)\,2\,(1\,2))\{\lambda\lambda 2\} \underset{\text{msust.}}{=} \lambda(\lambda\lambda 2)\{\lambda\lambda 2\}\,2\{\lambda\lambda 2\}\,(1\,2)\{\lambda\lambda 2\} \underset{\text{msust.}}{=}$$
$$\lambda(\lambda \uparrow_1(\lambda 2)\{\uparrow_0(\lambda\lambda 2)\})\;1((\lambda\lambda 2)\,1) \underset{\text{swap., incr.}}{=}$$
$$\lambda(\lambda(\lambda 3)\{\lambda\lambda 2\}\,)\,1((\lambda\lambda 2)\,1) \underset{\text{msust., swap., incr.}}{=}$$
$$\lambda(\lambda\lambda 2\,)\,1((\lambda\lambda 2)\,1) \to_{\beta_{\mathrm{r}}} \lambda(\lambda 2)\{1\}\,((\lambda\lambda 2)\,1) \underset{\text{msust.}}{=}$$
$$\lambda(\lambda \uparrow_1(2)\{\uparrow_0(1)\})\,((\lambda\lambda 2)\,1) \underset{\text{swap., incr.}}{=} \lambda(\lambda 1\{2\})\,((\lambda\lambda 2)\,1) \underset{\text{msust.}}{=}$$
$$\lambda(\lambda 2)\,((\lambda\lambda 2)\,1) \to_{\beta_{\mathrm{r}}} \lambda 2\{((\lambda\lambda 2)\,1)\} \underset{\text{msust.}}{=} \lambda 1 = \mathrm{I}$$

3.2 λ_{dB} y λr son el mismo cálculo

3.2.1 Comprendiendo el problema

Si quisiéramos probar que λ_{dB} y λr son el mismo cálculo, bastaría con ver que:

$$(\forall a, b \in \Lambda_{\mathrm{dB}})\,(a\{\{1 \leftarrow b\}\} = a\{b\})$$

Ahora bien, para probar esto, es necesario antes probar el caso general. Es decir:

$$(\forall a, b \in \Lambda_{\mathrm{dB}}, n \in \mathbb{N}_{>0})\,(a\{\{n \leftarrow b\}\} = a'\{b'\})$$

donde a' debería ser el resultado de una serie de *swaps* sobre a; y b', el resultado de una serie de subidas de índices de b. Esto, de forma tal de implementar la sustitución clásica de *de Bruijn*. Esta intuición proviene del hecho de que, para subir el índice en la sustitución clásica de *de Bruijn*, es necesario atravesar una abstracción. En nuestro cálculo, al atravesar un lambda, se realiza un swap y se suben los índices en el término de la sustitución. Por ende, para conocer la forma que debería tener a' en base a múltiples *swaps* e incrementos de índice, parecería bastar con comparar cuidosamente la reducción de un término utilizando ambas relaciones (β_{dB} y β_{r}). Veamos un ejemplo que clarifique la situación:

Sean $a, b \in \Lambda_{\mathrm{dB}}$. Tomemos el término $(\lambda\lambda\lambda a)\,b$. Veamos primero la reducción bajo β_{dB}:

$$\begin{array}{rcl} (\lambda\lambda\lambda a)\,b & \to_{\beta_{\mathrm{dB}}} & (\lambda\lambda a)\{\{1 \leftarrow b\}\} = \\ \lambda(\lambda a)\{\{2 \leftarrow b\}\} & = & \lambda\lambda a\{\{3 \leftarrow b\}\} \end{array}$$

Analicemos ahora la reducción bajo β_{r}:

$$\begin{array}{rcl}(\lambda\lambda\lambda a)\, b & \to_{\beta_{\mathrm{r}}} & (\lambda\lambda a)\{b\} = \\ \lambda \Uparrow_1(\lambda a)\{\uparrow_0(b)\} & = & \lambda(\lambda \Uparrow_2(a))\{\uparrow_0(b)\} = \\ \lambda\lambda \Uparrow_1(\Uparrow_2(a))\{\uparrow_0(\uparrow_0(b))\} & & \end{array}$$

Vemos así, que luego de la reducción y de atravesar las dos abstracciones que quedan en el término $(\lambda\lambda a)$, tenemos, bajo β_{dB} y β_{r} respectivamente, los términos:

$$\lambda\lambda a\{\{3 \leftarrow b\}\} \qquad \text{y} \qquad \lambda\lambda \Uparrow_1(\Uparrow_2(a))\{\uparrow_0(\uparrow_0(b))\}$$

En general, luego de atravesar $n-1$ abstracciones ($n \in \mathbb{N}_{>0}$), parecería que llegamos a los términos:

$$\underbrace{\lambda \cdots \lambda}_{n-1} a\{\{n \leftarrow b\}\} \qquad \text{y} \qquad \underbrace{\lambda \cdots \lambda}_{n-1} \Uparrow_1(\cdots \Uparrow_{n-1}(a) \cdots)\{\underbrace{\uparrow_0 (\cdots \uparrow_0}_{n-1}(b) \cdots)\}$$

En base a estos dos últimos términos generales, definimos dos nociones: una, la de anidamiento de *swaps*; otra, la de anidamiento de incrementos de índice. Cabe aclarar que estas definiciones existen sólo a modo de agregar claridad a las pruebas, siendo posible también lograr pruebas menos estrictas. Nuevamente, elegimos alargar un poco las pruebas para aportar claridad al desarrollo.

Definición 3.7 (Anidamiento de *swaps*)**.** Sean $i \in \mathbb{N}_{>0}$, $j \in \mathbb{N}$, $a \in \Lambda_{\mathrm{dB}}$. Definimos inductivamente $\Updownarrow_i^j \colon \Lambda_{\mathrm{dB}} \to \Lambda_{\mathrm{dB}}$ como:

$$\Updownarrow_i^j(a) \quad = \quad \begin{cases} a & \text{si } j = 0 \\ \Updownarrow_i^{j-1}(\Uparrow_{i+j-1}(a)) & \text{si } j > 0 \end{cases}$$

La idea intuitiva detrás de $\Updownarrow_i^j(a)$, es la de

$$\underbrace{\Uparrow_i (\Uparrow_{i+1} (\cdots \Uparrow_{i+j-1}}_{j \text{ swaps}}(a) \cdots))$$

En el caso del ejemplo anterior, el anidamiento de *swaps* resultante se describe como $\Updownarrow_1^2(a)$. Cabe destacar que el menor y mayor índice afectados por $\Updownarrow_i^j$ son, respectivamente, i e $i+j$.

Definición 3.8 (Anidamiento de incrementos de índice)**.** Sea $i \in \mathbb{N}$, $a \in \Lambda_{\mathrm{dB}}$. Definimos inductivamente $\Uparrow^i \colon \Lambda_{\mathrm{dB}} \to \Lambda_{\mathrm{dB}}$ como:

$$\Uparrow^i(a) \quad = \quad \begin{cases} a & \text{si } i = 0 \\ \Uparrow^{i-1}(\uparrow_0(a)) & \text{si } i > 0 \end{cases}$$

La idea intuitiva detrás de $\Uparrow^i(a)$, es la de

$$\underbrace{\uparrow_0 (\cdots \uparrow_0}_{i \text{ incrementos}}(a) \cdots)$$

En el caso del ejemplo anterior, el anidamiento de incrementos de índice resultante se describe como $\Uparrow^2(a)$.

A partir de estas dos definiciones, podemos dar el enunciado de la prueba que queremos realizar para mostrar que λ_{dB} y λr son iguales. Esto es:

$$(\forall a, b \in \Lambda_{\mathrm{dB}},\ n \in \mathbb{N}_{>0})\ \big(a\{\{n \leftarrow b\}\} = \Updownarrow_1^{n-1}(a)\{\Uparrow^{n-1}(b)\}\big)$$

Para poder demostrar esto último, es necesario probar antes una serie de lemas intermedios. Esto lo hacemos en la siguiente sección.

3.2.2 Lemas intermedios

A continuación, mostramos una serie de lemas necesarios para probar la igualdad entre λ_{dB} y λr.

Este primer lema sólo sirve a modo de confirmar la intuición: nos dice que una serie de incrementos de índice anidados se corresponde con la actualización clásica de *de Bruijn*.

Lema 3.9. Para todo $a \in \Lambda_{\mathrm{dB}},\ n \in \mathbb{N}_{>0},\ \Uparrow^{n-1}(a) = \mathrm{U}_0^n(a)$

Demostración. Por inducción en n.

- Caso base. $n = 1$. Luego, $\Uparrow^0(a) \underset{\text{def}}{=} a \underset{\text{O.1.47}}{=} \mathrm{U}_0^1(a)$.
- Paso inductivo. $n > 1$.

$$\Uparrow^{n-1}(a) \underset{\text{def}}{=} \Uparrow^{n-2}(\uparrow_0(a)) \underset{\text{O.3.2}}{=} \Uparrow^{n-2}(\mathrm{U}_0^2(a)) \underset{\text{HI}}{=} \mathrm{U}_0^{n-1}(\mathrm{U}_0^2(a)) \underset{\text{L.1.53}}{=} \mathrm{U}_0^n(a)$$

□

El siguiente lema dice que un *swap* anidado ($\Updownarrow_i^j$), aplicado a un índice mayor al máximo índice afectado por los *swaps*, no afecta al índice en cuestión.

Lema 3.10. Para todo $m, i \in \mathbb{N}_{>0},\ n \in \mathbb{N}$, si $m > n + i$, entonces $\Updownarrow_i^n(m) = m$.

Demostración. Por inducción en n.

- Caso base. $n = 0$. Luego, $\Updownarrow_i^0(m) \underset{\text{def}}{=} m$.
- Paso inductivo. $n > 0$. Tenemos $m > n + i$. Luego,

$$\Updownarrow_i^n(m) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(m)) \underset{\text{hip}}{=} \Updownarrow_i^{n-1}(m) \underset{\text{HI}}{=} m$$

□

El lema mostrado a continuación, nos dice que un *swap* anidado aplicado a un índice mayor o igual al menor índice afectado, y menor estricto al mayor índice afectado, tiene como resultado el incremento en uno de dicho índice. Es decir, éste es alcanzado por sólo uno de los *swaps*.

Lema 3.11. Para todo $m, i \in \mathbb{N}_{>0},\ n \in \mathbb{N}$, si $i \leq m < n + i$, entonces $\Updownarrow_i^n(m) = m + 1$.

Demostración. Por inducción en n.

- Caso base. $n = 0$. Luego, como $i \leq m$, no puede ocurrir que $m < n+i = i$. Vale vacuamente.
- Paso inductivo. $n > 0$. Tenemos $m < n + i$. Luego,

$$\Updownarrow_i^n(m) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(m))$$

Dos casos:

1. $m < n + i - 1 \implies \Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(m)) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(m) \underset{\text{HI}}{=} m + 1$
2. $m = n + i - 1 \implies \Updownarrow_1^{n-1}(\updownarrow_{n+i-1}(n + i - 1)) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(n + i) \underset{\text{L.3.10}}{=} n + i \underset{\text{hip}}{=} m + 1$

□

El siguiente y último de los lemas referentes a *swaps* anidados aplicados a índices, nos dice que si el índice al que se le aplica es igual al mayor índice afectado por la operación, y dicha operación empieza desde i, entonces el resultado es exactamente i. Dicho de otra forma, el índice $n+i$ va "bajando" hasta llegar a i.

Lema 3.12. Para todo $i \in \mathbb{N}_{>0},\ n \in \mathbb{N},\ \Updownarrow_i^n(n + i) = i$.

Demostración. Por inducción en n.

- Caso base. $n = 0$. Luego, $\Updownarrow_i^0(i) \underset{\text{def}}{=} i$
- Paso inductivo. $n > 0$. Luego,

$$\Updownarrow_i^n(n + i) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(n + i)) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(n + i - 1) \underset{\text{HI}}{=} i$$

□

Este lema dice cómo interactúan los swaps anidados con las aplicaciones (nos confirma, en realidad, que el comportamiento es el esperado).

Lema 3.13. Para todo $a, b \in \Lambda_{\text{dB}},\ n \in \mathbb{N},\ i \in \mathbb{N}_{>0},\ \Updownarrow_i^n(a)\ \Updownarrow_i^n(b) = \Updownarrow_i^n(a\, b)$

Demostración. Por inducción en n.

- Caso base. $n = 0$. Luego, $\Updownarrow_i^0(a)\ \Updownarrow_i^0(b) \underset{\text{def}}{=} a\, b \underset{\text{def}}{=} \Updownarrow_i^0(a\, b)$
- Paso inductivo. $n > 0$. Luego,

$$\Updownarrow_i^n(a)\ \Updownarrow_i^n(b) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(a))\ \Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(b)) \underset{\text{HI}}{=}$$
$$\Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(a)\ \updownarrow_{n+i-1}(b)) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(a\, b)) \underset{\text{def}}{=} \Updownarrow_i^n(a\, b)$$

□

El próximo, nos muestra qué es lo que ocurre cuando un *swap* anidado atraviesa una abstracción.

Lema 3.14. Para todo $a \in \Lambda_{\mathrm{dB}}, n \in \mathbb{N},\ i \in \mathbb{N}_{>0},\ \Updownarrow_i^n(\lambda a) = \lambda\ \Updownarrow_{i+1}^n(a)$

Demostración. Por inducción en n.

- Caso base. $n = 0$. Luego, $\Updownarrow_i^0(\lambda a) \underset{\text{def}}{=} \lambda a \underset{\text{def}}{=} \lambda\ \Updownarrow_{i+1}^0(a)$.
- Paso inductivo. $n > 0$. Luego,

$$\Updownarrow_i^n(\lambda a) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(\updownarrow_{n+i-1}(\lambda a)) \underset{\text{def}}{=} \Updownarrow_i^{n-1}(\lambda\ \updownarrow_{n+i}(a)) \underset{\text{HI}}{=}$$
$$\lambda\ \Updownarrow_{i+1}^{n-1}(\updownarrow_{n+i}(a)) \underset{\text{def}}{=} \lambda\ \Updownarrow_{i+1}^n(a)$$

□

El siguiente y último lema, nos muestra qué es lo que ocurre cuando una sustitución de un término con incrementos de índice atraviesa una abstracción que contiene un *swap* anidado dentro.

Lema 3.15. Para todo $a, b \in \Lambda_{\mathrm{dB}},\ n \in \mathbb{N}_{>0}$,

$$\lambda\ \Updownarrow_1^n(a)\{\Uparrow^n(b)\} = (\lambda\ \Updownarrow_2^{n-1}(a))\{\Uparrow^{n-1}(b)\}$$

Demostración. Por inducción en n.

- Caso base. $n = 1$. Luego,

$$\lambda\ \Updownarrow_1^1(a)\{\Uparrow^1(b)\} \underset{\text{def}}{=} \lambda\ \updownarrow_1(a)\{\uparrow_0(b)\} \underset{\text{Msust}}{=}$$
$$(\lambda a)\{b\} \underset{\text{def}}{=} (\lambda\ \Updownarrow_2^0(a))\{\Uparrow^0(b)\}$$

- Paso inductivo. $n > 1$. Luego,

$$\lambda\ \Updownarrow_1^n(a)\{\Uparrow^n(b)\} \underset{\text{def}}{=} \lambda\ \Updownarrow_1^{n-1}(\updownarrow_n(a))\{\Uparrow^{n-1}(\uparrow_0(b))\} \underset{\text{HI}}{=}$$
$$(\lambda\ \Updownarrow_2^{n-2}(\updownarrow_n(a)))\{\Uparrow^{n-2}(\uparrow_0(b))\} \underset{\text{def}}{=} (\lambda\ \Updownarrow_2^{n-1}(a))\{\Uparrow^{n-1}(b)\}$$

□

3.2.3 Igualdad de los cálculos

Teniendo ya los lemas auxiliares necesarios, podemos pasar a probar el lema principal que lleva a mostrar que λ_{dB} y λr son el mismo cálculo.

Lema 3.16 (Correspondencia entre metasustitución en λ_{dB} y $\lambda\mathrm{r}$)**.** Para todo $a, b \in \Lambda_{\mathrm{dB}},\ n \in \mathbb{N}_{>0},\ a\{\{n \leftarrow b\}\} = \Downarrow_1^{n-1}(a)\{\Uparrow^{n-1}(b)\}$

Demostración. Por inducción en a.

- $a = m \in \mathbb{N}_{>0}$. Luego,

$$a\{\{n \leftarrow b\}\} = m\{\{n \leftarrow b\}\} = \begin{cases} m-1 & \text{si } m > n \\ \mathrm{U}_0^n(b) & \text{si } m = n \\ m & \text{si } m < n \end{cases}$$

  Dividimos en casos:

  1. $m > n \implies \Downarrow_1^{n-1}(m)\{\Uparrow^{n-1}(b)\} \underset{\text{L.3.10}}{=} m\{\Uparrow^{n-1}(b)\} \underset{m\,>\,n\,\geq\,1}{=} m-1$
  2. $m = n \implies \Downarrow_1^{n-1}(n)\{\Uparrow^{n-1}(b)\} \underset{\text{L.3.12}}{=} 1\{\Uparrow^{n-1}(b)\} \underset{\text{def}}{=} \Uparrow^{n-1}(b) \underset{\text{L.3.9}}{=} \mathrm{U}_0^n(b)$
  3. $m < n \implies \Downarrow_1^{n-1}(m)\{\Uparrow^{n-1}(b)\} \underset{\text{L.3.11}}{=} m+1\{\Uparrow^{n-1}(b)\} \underset{m\,+\,1\,>\,1}{=} m$

  Luego,
$$m\{\{n \leftarrow b\}\} = \Downarrow_1^{n-1}(m)\{\Uparrow^{n-1}(b)\}$$

- $a = c\,d,\ c, d \in \Lambda_{\mathrm{dB}}$. Luego,

$$a\{\{n \leftarrow b\}\} = (c\,d)\{\{n \leftarrow b\}\} \underset{\text{def}}{=} c\{\{n \leftarrow b\}\}\,d\{\{n \leftarrow b\}\} \underset{\text{HI}}{=}$$
$$\Downarrow_1^{n-1}(c)\{\Uparrow^{n-1}(b)\}\ \Downarrow_1^{n-1}(d)\{\Uparrow^{n-1}(b)\} \underset{\text{def}}{=} (\Downarrow_1^{n-1}(c)\ \Downarrow_1^{n-1}(d))\{\Uparrow^{n-1}(b)\} \underset{\text{L.3.13}}{=}$$
$$\Downarrow_1^{n-1}(c\,d)\{\Uparrow^{n-1}(b)\} = \Downarrow_1^{n-1}(a)\{\Uparrow^{n-1}(b)\}$$

- $a = \lambda c,\ c \in \Lambda_{\mathrm{dB}}$. Luego,

$$a\{\{n \leftarrow b\}\} = (\lambda c)\{\{n \leftarrow b\}\} \underset{\text{def}}{=} \lambda c\{\{n+1 \leftarrow b\}\} \underset{\text{HI}}{=}$$
$$\lambda\ \Downarrow_1^{n}(c)\{\Uparrow^{n}(b)\} \underset{\text{L.3.15}}{=} (\lambda\ \Downarrow_2^{n-1}(c))\{\Uparrow^{n-1}(b)\} \underset{\text{L.3.14}}{=}$$
$$\Downarrow_1^{n-1}(\lambda c)\{\Uparrow^{n-1}(b)\} = \Downarrow_1^{n-1}(a)\{\Uparrow^{n-1}(b)\}$$

□

Corolario 3.17. Para todo $a, b \in \Lambda_{\mathrm{dB}},\ a\{\{1 \leftarrow b\}\} = a\{b\}$

Demostración. Usar lema 3.16 con $n = 1$. □

Nota. *Es interesante mencionar que, en el marco de las correcciones a esta tesis, Beta Ziliani corroboró este resultado utilizando el asistente para demostraciones* Coq[1].

Enunciamos y probamos, ahora sí, la igualdad entre λ_{dB} y $\lambda\mathrm{r}$. Para esto, basta el siguiente lema:

[1] utilizando Coq v8.2 (`http://coq.inria.fr`) con la extensión Ssreflect 1.1 (`http://www.msr-inria.inria.fr/events-news/ssreflect-1-1-release-1`)

Lema 3.18. Para todo $a, b \in \Lambda_{\mathrm{dB}}$, $a \ \to_{\beta_{\mathrm{dB}}} \ b$ sii $a \to_{\beta_{\mathrm{r}}} b$. Es decir, $\beta_{\mathrm{dB}} = \beta_{\mathrm{r}}$.

Demostración.

$$a \ \to_{\beta_{\mathrm{dB}}} \ b \iff (\exists\ \mathrm{C} \text{ contexto},\ c, d \in \Lambda_{\mathrm{dB}})\,(a = \mathrm{C}\,[(\lambda c)\,d] \wedge b = \mathrm{C}\,[c\{\{1 \leftarrow d\}\}])$$

$$\underset{\text{C.3.17}}{\iff} (\exists\ \mathrm{C} \text{ contexto},\ c, d \in \Lambda_{\mathrm{dB}})\,(a = \mathrm{C}\,[(\lambda c)\,d] \wedge b = \mathrm{C}\,[c\{d\}]) \iff a \to_{\beta_{\mathrm{r}}} b$$

□

Teorema 3.19 (Igualdad entre λ_{dB} y $\lambda\mathrm{r}$)**.** Los cálculos λ_{dB} y $\lambda\mathrm{r}$ son iguales.

Demostración. $\beta_{\mathrm{dB}} = \beta_{\mathrm{r}} \implies \lambda_{\mathrm{dB}} = (\Lambda_{\mathrm{dB}}, \beta_{\mathrm{dB}}) = (\Lambda_{\mathrm{dB}}, \beta_{\mathrm{r}}) = \lambda\mathrm{r}$ □

Capítulo 4

Cálculo λre

4.1 El cálculo de sustituciones explícitas λre

4.1.1 Introducción

En la presente sección introducimos un cálculo de sustituciones explícitas derivado del cálculo λr: λre. Éste surge a partir de un cambio: la inclusión en el cálculo del operador de metasustitución de λr (*i.e.,* $\bullet\{\bullet\}$).

Es importante destacar que, en este cálculo, no todos los operadores son explícitos: los operadores de incremento de índices ($\uparrow_i$) y de *swaping* ($\updownarrow_i$) siguen estando en el metalenguaje. Esta decisión tiene dos motivos subyacentes: uno, el de realizar un desarrollo paulatino de un cálculo de sustituciones explícitas que posea *todas* las buenas propiedades, con la ventaja de poder ubicar las posibles falencias en el diseño de una manera modular; dos: el de encontrar una correspondencia uno a uno (*i.e.,* un isomorfismo) con algún cálculo de sustituciones explícitas con nombres que ya posea buenas propiedades. De hecho, y como veremos más adelante, este cálculo resulta ser isomorfo al cálculo de sustituciones explícitas λx [Blo95, Blo97].

4.1.2 Algunos lemas

Como es de esperarse, los términos del cálculo λre serán los mismos que los del cálculo λr, con el agregado de términos de la forma $a[b]$, la versión explícita de $a\{b\}$ en λr. Ahora bien, como tenemos dos metaoperadores que deben funcionar sobre este nuevo conjunto de términos (a saber, $\updownarrow_i$ y $\uparrow_i$), es necesario extender su definición al caso de la sustitución explícita. Para esto, mostramos primero algunas propiedades sobre la interacción entre estos operadores y las sustituciones en λr, con el objetivo de internalizarlas luego dentro del cálculo como definiciones.

Antes de pasar a las propiedades, sin embargo, necesitamos definir el tamaño de un término $a \in \Lambda_{\mathrm{dB}}$. Esto, para luego mostrar que las operaciones de *swapping* no lo afectan. Es decir, que:

$$\forall a \in \Lambda_{\mathrm{dB}},\ i \in \mathbb{N}_{>0} : |a| = |\updownarrow_i(a)|$$

Pasamos, entonces, a definir el tamaño de un término.

Definición 4.1 (Tamaño de un término de λr)**.** El tamaño de un término de λr, $|\bullet| : \Lambda_{\mathrm{dB}} \to \mathbb{N}$ se define inductivamente como:

$$\begin{aligned} |n| &= 1 \\ |a\,b| &= |a| + |b| \\ |\lambda a| &= |a| + 1 \end{aligned}$$

Nota. $\forall a \in \Lambda_{dB} : |a| \geq 1$

Lema 4.2 (Preservación del tamaño por *swap*)**.** Para todo $a \in \Lambda_{\mathrm{dB}},\ i \in \mathbb{N}_{>0}$,

$$|\uparrow\!\downarrow_i(a)| = |a|$$

Demostración. Por inducción en a. Sea $i \in \mathbb{N}_{>0}$.

- $a = m \in \mathbb{N}_{>0}$. Luego,

$$|\uparrow\!\downarrow_i(a)| = |\uparrow\!\downarrow_i(m)| = \begin{cases} |m| = |a| & \text{si } n < i \ \vee \ n > i+1 \\ |i+1| = 1 = |m| = |a| & \text{si } n = i \\ |i| = 1 = |m| = |a| & \text{si } n = i+1 \end{cases}$$

- $a = c\,d,\ c, d \in \Lambda_{\mathrm{dB}}$. Luego,

$$\begin{aligned} |\uparrow\!\downarrow_i(a)| = |\uparrow\!\downarrow_i(c\,d)| \underset{\text{def}}{=} |\uparrow\!\downarrow_i(c)\ \uparrow\!\downarrow_i(d)| \underset{\text{def}}{=} \\ |\uparrow\!\downarrow_i(c)| + |\uparrow\!\downarrow_i(d)| \underset{\text{HI}}{=} |c| + |d| \underset{\text{def}}{=} |c\,d| = |a| \end{aligned}$$

- $a = \lambda c,\ c \in \Lambda_{\mathrm{dB}}$. Luego,

$$\begin{aligned} |\uparrow\!\downarrow_i(a)| = |\uparrow\!\downarrow_i(\lambda c)| \underset{\text{def}}{=} |\lambda\uparrow\!\downarrow_{i+1}(c)| \underset{\text{def}}{=} \\ 1 + |\uparrow\!\downarrow_{i+1}(c)| \underset{\text{HI}}{=} 1 + |c| \underset{\text{def}}{=} |\lambda c| = |a| \end{aligned}$$

□

Ahora sí, pasamos a enumerar y demostrar las propiedades que necesitaremos más adelante.

Este próximo lema nos muestra cuándo y cómo es posible intercambiar el orden de aplicación entre *swaps* y subidas de índice.

Lema 4.3 (Interacción entre subidas de índice y *swaps*)**.** Para todo $a \in \Lambda_{\mathrm{dB}},\ i \in \mathbb{N}_{>0},\ j \in \mathbb{N}$, si $j < i$, entonces $\uparrow\!\downarrow_{i+1}(\uparrow_j(a)) = \uparrow_j(\uparrow\!\downarrow_i(a))$.

Demostración. Por inducción en a. Sean $i \in \mathbb{N}_{>0},\ j \in \mathbb{N} : j < i$.

- $a = n \in \mathbb{N}_{>0}$. Dividimos en dos casos:

1. $n > j \implies \uparrow_{i+1}(\uparrow_j(n)) =$

   $$\uparrow_{i+1}(n+1) = \begin{cases} n+1 & \text{si } n+1 \notin \{i+1, i+2\} \iff n \notin \{i, i+1\} \\ i+2 & \text{si } n+1 = i+1 \iff n = i \\ i+1 & \text{si } n+1 = i+2 \iff n = i+1 \end{cases}$$

   Ahora,

   $$\uparrow_j(\uparrow_i(n)) = \begin{cases} \uparrow_j(n) \underset{n > j}{=} n+1 & n \notin \{i, i+1\} \\ \uparrow_j(i+1) \underset{j < i}{=} i+2 & n = i \\ \uparrow_j(i) \underset{j < i}{=} i+1 & n = i+1 \end{cases}$$

2. $n \leq j \underset{j < i}{\implies} n < i$
   Tenemos que $\uparrow_{i+1}(\uparrow_j(n)) = \uparrow_{i+1}(n) \underset{n < i}{=} n$
   Ahora, $\uparrow_j(\uparrow_i(n)) \underset{n < i}{=} \uparrow_j(n) \underset{n \leq j}{=} n$

- $a = b\,c,\ b, c \in \Lambda_{\text{dB}}$. Luego,

$$\uparrow_{i+1}(\uparrow_j(b\,c)) \underset{\text{def}}{=} \uparrow_{i+1}(\uparrow_j(b)\ \uparrow_j(c)) \underset{\text{def}}{=} \uparrow_{i+1}(\uparrow_j(b))\ \uparrow_{i+1}(\uparrow_j(c)) \underset{\text{HI}}{=}$$
$$\uparrow_j(\uparrow_i(b))\ \uparrow_j(\uparrow_i(c)) \underset{\text{def}}{=} \uparrow_j(\uparrow_i(b)\ \uparrow_i(c)) \underset{\text{def}}{=} \uparrow_j(\uparrow_i(b\,c))$$

- $a = \lambda b,\ b \in \Lambda_{\text{dB}}$. Luego,

$$\uparrow_{i+1}(\uparrow_j(\lambda b)) \underset{\text{def}}{=} \uparrow_{i+1}(\lambda\ \uparrow_{j+1}(b)) \underset{\text{def}}{=} \lambda\ \uparrow_{i+2}(\uparrow_{j+1}(b)) \underset{\text{HI}}{=}$$
$$\lambda\ \uparrow_{j+1}(\uparrow_{i+1}(b)) \underset{\text{def}}{=} \uparrow_j(\lambda\ \uparrow_{i+1}(b)) \underset{\text{def}}{=} \uparrow_j(\uparrow_i(\lambda b))$$

□

El lema que mostramos a continuación nos dice en qué situaciones es posible intercambiar el orden de aplicación de dos *swaps* anidados sobre un término.

Lema 4.4 (Independencia de *swaps*)**.** Para todo $a \in \Lambda_{\text{dB}},\ i, j \in \mathbb{N}_{>0}$, si $j \geq 2$, entonces $\uparrow_{i+j}(\uparrow_i(a)) = \uparrow_i(\uparrow_{i+j}(a))$.

Demostración. Por inducción en a. Sean $i, j \in \mathbb{N}_{>0} : j \geq 2$.

- $a = n \in \mathbb{N}_{>0}$. Dividimos en cuatro casos:

  1. $n < i$. Tenemos $n < i \wedge j \geq 2 \implies n < i + j$. Ahora,

     $\uparrow_{i+j}(\uparrow_i(n)) = \uparrow_{i+j}(n) = n$. Además, $\uparrow_i(\uparrow_{i+j}(n)) = \uparrow_i(n) = n$.

  2. $n > i + 1$. Tenemos $\uparrow_{i+j}(\uparrow_i(n)) =$

     $$\uparrow_{i+j}(n) = \begin{cases} n & \text{si } i+1 < n < i+j \\ n & \text{si } n > i+j+1 \\ i+j+1 & \text{si } n = i+j \\ i+j & \text{si } n = i+j+1 \end{cases}$$

Ahora,

$$\uparrow_i(\uparrow_{i+j}(n)) = \begin{cases} \uparrow_i(n) = n & \text{si } i+1 < n < i+j \\ \uparrow_i(n) = n & \text{si } n > i+j+1 \\ \uparrow_i(i+j+1) \underset{j \geq 2}{=} i+j+1 & \text{si } n = i+j \\ \uparrow_i(i+j) \underset{j \geq 2}{=} i+j & \text{si } n = i+j+1 \end{cases}$$

3. $n = i$. Tenemos $1 < j \implies i+1 < i+j$. Luego,

$$\uparrow_{i+j}(\uparrow_i(n)) = \uparrow_{i+j}(i+1) = i+1.$$

Además, $j \geq 2 \implies i < i+j$. Entonces,

$$\uparrow_i(\uparrow_{i+j}(n)) = \uparrow_i(i) = i+1.$$

4. $n = i+1$. Tenemos $j \geq 2 \implies (i < i+j) \wedge (i+1 < i+j)$. Entonces,

$$\uparrow_{i+j}(\uparrow_i(n)) = \uparrow_{i+j}(i) = i.$$

Además,

$$\uparrow_i(\uparrow_{i+j}(n)) = \uparrow_i(i+1) = i.$$

- $a = b\,c,\ b, c \in \Lambda_{\mathrm{dB}}$. Luego,

$$\uparrow_{i+j}(\uparrow_i(b\,c)) \underset{\text{def}}{=} \uparrow_{i+j}(\uparrow_i(b)\ \uparrow_i(c)) \underset{\text{def}}{=} \uparrow_{i+j}(\uparrow_i(b))\ \uparrow_{i+j}(\uparrow_i(c)) \underset{\text{HI}}{=}$$
$$\uparrow_i(\uparrow_{i+j}(b))\ \uparrow_i(\uparrow_{i+j}(c)) \underset{\text{def}}{=} \uparrow_i(\uparrow_{i+j}(b)\ \uparrow_{i+j}(c)) \underset{\text{def}}{=} \uparrow_i(\uparrow_{i+j}(b\,c))$$

- $a = \lambda b,\ b \in \Lambda_{\mathrm{dB}}$. Luego,

$$\uparrow_{i+j}(\uparrow_i(\lambda b)) \underset{\text{def}}{=} \uparrow_{i+j}(\lambda\ \uparrow_{i+1}(b)) \underset{\text{def}}{=} \lambda\ \uparrow_{i+j+1}(\uparrow_{i+1}(b)) \underset{\text{HI}}{=}$$
$$\lambda\ \uparrow_{i+1}(\uparrow_{i+j+1}(b)) \underset{\text{def}}{=} \uparrow_i(\lambda\ \uparrow_{i+j+1}(b)) \underset{\text{def}}{=} \uparrow_i(\uparrow_{i+j}(\lambda b))$$

□

Los siguientes dos lemas permiten conocer la forma en que un *swap* y un incremento de índices pueden distribuirse dentro de una sustitución. Son la base de las definiciones extendidas de los metaoperadores para el cálculo λre.

Lema 4.5 (Interacción entre metasustituciónes y $\uparrow_i$ en λr)**.** Para todo $a, b \in \Lambda_{\mathrm{dB}},\ i \in \mathbb{N}_{>0},\ \uparrow_i(a\{b\}) = \uparrow_{i+1}(a)\{\uparrow_i(b)\}$

Demostración. Por inducción en $|a|$. Sean $b \in \Lambda_{\mathrm{dB}},\ i \in \mathbb{N}_{>0}$.

- Caso base. $|a| = 1$. Luego, $a = n \in \mathbb{N}_{>0}$. Dividimos en dos casos:

1. $n = 1$. Luego, $\uparrow_i(1\{b\}) = \uparrow_i(b)$. Además,

$$\uparrow_{i+1}(1)\{\uparrow_i(b)\} \underset{i+1>1}{=} 1\{\uparrow_i(b)\} = \uparrow_i(b)$$

2. $n > 1$. Luego,

$$\uparrow_i(n\{b\}) = \uparrow_i(n-1) = \begin{cases} n-1 & \text{si } n-1 < i \iff n < i+1 \\ n-1 & \text{si } n-1 > i+1 \iff n > i+2 \\ i+1 & \text{si } n-1 = i \iff n = i+1 \\ i & \text{si } n-1 = i+1 \iff n = i+2 \end{cases}$$

Ahora,

$$\uparrow_{i+1}(n)\{\uparrow_i(b)\} = \begin{cases} n\{\uparrow_i(b)\} \underset{\text{hip}}{=} n-1 & \text{si } n < i+1 \\ n\{\uparrow_i(b)\} \underset{\text{hip}}{=} n-1 & \text{si } n > i+2 \\ (i+2)\{\uparrow_i(b)\} = i+1 & \text{si } n = i+1 \\ (i+1)\{\uparrow_i(b)\} = i & \text{si } n = i+2 \end{cases}$$

- Paso inductivo. $|a| > 1$. Tenemos dos opciones,

1. $a = c\,d,\ c, d \in \Lambda_{\text{dB}}$. Luego,

$$\uparrow_i((c\,d)\{b\}) \underset{\text{def}}{=} \uparrow_i(c\{b\}\ d\{b\}) \underset{\text{def}}{=} \uparrow_i(c\{b\})\ \uparrow_i(d\{b\}) \underset{\text{HI}}{=}$$
$$\uparrow_{i+1}(c)\{\uparrow_i(b)\}\ \uparrow_{i+1}(d)\{\uparrow_i(b)\} \underset{\text{def}}{=} (\uparrow_{i+1}(c)\ \uparrow_{i+1}(d))\{\uparrow_i(b)\} \underset{\text{def}}{=}$$
$$\uparrow_{i+1}(c\,d)\{\uparrow_i(b)\}$$

2. $a = \lambda c,\ c \in \Lambda_{\text{dB}}$. Luego,

$$\uparrow_i((\lambda c)\{b\}) \underset{\text{def}}{=} \uparrow_i(\lambda\ \uparrow_1(c)\{\uparrow_0(b)\}) \underset{\text{def}}{=} \lambda\ \uparrow_{i+1}(\uparrow_1(c)\{\uparrow_0(b)\}) \underset{\text{HI, L.4.2}}{=}$$
$$\lambda\ \uparrow_{i+2}(\uparrow_1(c))\{\uparrow_{i+1}(\uparrow_0(b))\} \underset{\text{L.4.3}}{=} \lambda\ \uparrow_{i+2}(\uparrow_1(c))\{\uparrow_0(\uparrow_i(b))\} \underset{\text{L.4.4}}{=}$$
$$\lambda\ \uparrow_1(\uparrow_{i+2}(c))\{\uparrow_0(\uparrow_i(b))\} \underset{\text{def}}{=} (\lambda\ \uparrow_{i+2}(c))\{\uparrow_i(b)\} \underset{\text{def}}{=} \uparrow_{i+1}(\lambda c)\{\uparrow_i(b)\}$$

□

Lema 4.6 (Interacción entre metasustituciónes y $\uparrow_i$ en λr)**.** Para todo $a, b \in \Lambda_{\text{dB}},\ i \in \mathbb{N},\ \uparrow_i(a\{b\}) = \uparrow_{i+1}(a)\{\uparrow_i(b)\}$

Observación 4.7. Del lema 1.54, tenemos que

$$\forall a, b \in \Lambda_{\text{dB}},\ n, i \in \mathbb{N}_{>0},\ k \in \mathbb{N} : n \leq k+1 \implies$$
$$\text{U}^i_k(a\{\{n \leftarrow b\}\}) = \text{U}^i_{k+1}(a)\{\{n \leftarrow \text{U}^i_{k-n+1}(b)\}\}$$

En particular, con $n = 1$, tenemos que:

$$\text{U}^i_k(a\{\{1 \leftarrow b\}\}) = \text{U}^i_{k+1}(a)\{\{1 \leftarrow \text{U}^i_k(b)\}\}$$

Demostración. (lema 4.6) Directa. Sean $a, b \in \Lambda_{\text{dB}},\ i \in \mathbb{N}$.

$$\uparrow_i(a\{b\}) \underset{\text{O.3.2}}{=} \text{U}^2_i(a\{b\}) \underset{\text{C.3.17}}{=} \text{U}^2_i(a\{\{1 \leftarrow b\}\}) \underset{\text{O.4.7}}{=}$$
$$\text{U}^2_{i+1}(a)\{\{1 \leftarrow \text{U}^2_i(b)\}\} \underset{\text{O.3.2}}{=} \uparrow_{i+1}(a)\{\{1 \leftarrow \uparrow_i(b)\}\} \underset{\text{C.3.17}}{=} \uparrow_{i+1}(a)\{\uparrow_i(b)\}$$

□

Teniendo los dos lemas anteriores (*i.e.,* de distribución de *swaps* e incrementos de índice dentro de una sustitución), podemos pasar a definir el cálculo λre. Esto lo hacemos en la siguiente sección.

4.1.3 Definiciones

A continuación, definimos los términos del cálculo λre:

Definición 4.8 (Conjunto de términos Λre)**.** El conjunto Λre está dado por:

$$a ::= n \mid a\,a \mid \lambda a \mid a[a] \qquad n \in \mathbb{N}_{>0}$$

Nota. *Como puede apreciarse a partir de la definición anterior, los términos de λre son los mismos que los de λ_{dB}, con el agregado de términos con sustitución explícita à la λr (*i.e., *$a[b]$). Dicho esto, decimos que un término $a \in \Lambda re$ es* puro *si y sólo si éste no posee ninguna sustitución explícita.*

Definición 4.9 (Variables libres de un término de λre)**.** Las variables libres de un término, $\mathrm{FV} : \Lambda re \to \mathcal{P}(\mathbb{N}_{>0})$, se definen inductivamente como:

$$\begin{aligned}
\mathrm{FV}(n) &= \{n\} \\
\mathrm{FV}(a\,b) &= \mathrm{FV}(a) \cup \mathrm{FV}(b) \\
\mathrm{FV}(\lambda a) &= \mathrm{FV}(a) - 1 \\
\mathrm{FV}(a[b]) &= (\mathrm{FV}(a) - 1) \cup \mathrm{FV}(b)
\end{aligned}$$

Como se mencionó anteriormente, dado que en λre tenemos, además de aplicaciones y abstracciones, términos de la forma $a[b]$, debemos extender las definiciones de *swaps* e incrementos de índice para estos. Para la extensión de incrementos de índice a la sustitución explícita, nos basamos en el lema 4.6, mostrado en la subsección anterior.

Definición 4.10 (Metaoperador $\uparrow_i$)**.** Para todo $i \in \mathbb{N}$, $\uparrow_i : \Lambda re \to \Lambda re$ se define inductivamente como:

$$\begin{aligned}
\uparrow_i(n) &= \begin{cases} n & \text{si } n \leq i \\ n+1 & \text{si } n > i \end{cases} \\
\uparrow_i(a\,b) &= \uparrow_i(a)\ \uparrow_i(b) \\
\uparrow_i(\lambda a) &= \lambda\, \uparrow_{i+1}(a) \\
\uparrow_i(a[b]) &= \uparrow_{i+1}(a)[\uparrow_i(b)]
\end{aligned}$$

Para la extensión de *swaps* a la sustitución explícita, nos basamos en el lema 4.5, también mostrado en la subsección anterior.

Definición 4.11 (Metaoperador $\updownarrow_i$)**.** Para todo $i \in \mathbb{N}_{>0}$, $\updownarrow_i : \Lambda re \to \Lambda re$ se define inductivamente como:

$$\begin{aligned}
\updownarrow_i(n) &= \begin{cases} n & \text{si } n < i \ \vee\ n > i+1 \\ i+1 & \text{si } n = i \\ i & \text{si } n = i+1 \end{cases} \\
\updownarrow_i(a\,b) &= \updownarrow_i(a)\ \updownarrow_i(b) \\
\updownarrow_i(\lambda a) &= \lambda\, \updownarrow_{i+1}(a) \\
\updownarrow_i(a[b]) &= \updownarrow_{i+1}(a)[\updownarrow_i(b)]
\end{aligned}$$

Una vez definidos los metaoperadores de *swap* e incremento de índices pasamos, ahora sí, a presentar las reglas del cálculo λre. Éstas pueden apreciarse en la figura 4.1.

(Beta)	$(\lambda a)\, b$	$\to$	$a[b]$	
(App)	$(a\, b)[c]$	$\to$	$a[c]\ b[c]$	
(Lamb)	$(\lambda a)[c]$	$\to$	$\lambda \uparrow_1(a)[\uparrow_0(c)]$	
(Var)	$1[c]$	$\to$	c	
(VarR)	$(n+1)[c]$	$\to$	n	$(n \in \mathbb{N}_{>0})$

Figura 4.1: Reglas del cálculo λre

Llamamos re al conjunto de reglas (Abs), (Lamb), (Var) y (VarR). Dicho conjunto conforma las reglas necesarias para la distribución de las sustituciones explícitas dentro de un término.

Llamamos, además, λre al conjunto de reglas (Beta) + re.

Antes de pasar a definir formalmente el cálculo λre, damos una explicación de sus reglas de reducción:

- Regla **(Beta)**: Es la regla equivalente a β_{r} en λr. Se ocupa de generar la sustitución explícita, que luego será distribuida en el término por el resto de las reglas.
- Regla **(App)**: Se encarga de distribuir una sustitución dentro de una aplicación.
- Regla **(Lamb)**: Tiene la función de definir lo que ocurre cuando una sustitución atraviesa una abstracción. Notar que es la misma definición que para el caso de la metasustitución en λr.
- Regla **(Var)**: Es la regla que realiza una sustitución. Es decir, cuando nos encontramos con un término de la forma $1[a]\,,\ a \in \Lambda\mathrm{re}$, lo que debe ocurrir, es que dicho 1 sea reemplazado por a (tal es el caso para la metasustitución en λr).
- Regla **(VarR)**: Es la contrapartida de la regla **(Var)**. Es decir, se encarga de decrementar los índices no afectados por una sustitución particular. Este decremento encuentra su raíz en el hecho de que se está quitando una abstracción (recordar la intuición acerca del cálculo λr).

Definimos, entonces, el cálculo λre:

Definición 4.12 (re-reducción)**.** Se define la relación $\mathrm{re} \subseteq \Lambda\mathrm{re} \times \Lambda\mathrm{re}$, notada $\to_{\mathrm{re}}$ como la clausura contextual de las reglas re.

Definición 4.13 (λre-reducción)**.** Se define la relación $\lambda\mathrm{re} \subseteq \Lambda\mathrm{re} \times \Lambda\mathrm{re}$, notada $\to_{\lambda\mathrm{re}}$ como la clausura contextual de las reglas λre.

Definición 4.14 (Cálculo λre)**.** El cálculo λre es el sistema de reducción $(\Lambda\mathrm{re}, \lambda\mathrm{re})$

4.1.4 No agregado de variables libres por reducción

Antes de cerrar la presentación del cálculo λre, damos dos lemas auxiliares y, posteriormente, otro lema más, que nos garantiza la inclusión de variables libres del reducto de un término en las variables libres de él mismo.

Lema 4.15. Para todo $a \in \Lambda\text{re}$, $i \in \mathbb{N}_{>0}$, tenemos que $\mathrm{FV}(\uparrow_i(a)) - (i+1) = \mathrm{FV}(a) - (i+1)$.

Demostración. Por inducción en a. Sea $i \in \mathbb{N}_{>0}$.

- $a = n \in \mathbb{N}_{>0}$. Dividimos en tres casos:
  1. $n < i \vee n > i+1$. Luego, por definición, tenemos que
  $$\mathrm{FV}(\uparrow_i(n)) - (i+1) = \mathrm{FV}(n) - (i+1)$$
  2. $n = i$. Luego,
  $$\mathrm{FV}(\uparrow_i(i)) - (i+1) = \mathrm{FV}(i+1) - (i+1) = \{i+1\} - (i+1) = \emptyset$$
  Además,
  $$\mathrm{FV}(i) - (i+1) = \{i\} - (i+1) = \emptyset$$
  3. $n = i+1$. Luego,
  $$\mathrm{FV}(\uparrow_i(i+1)) - (i+1) = \mathrm{FV}(i) - (i+1) = \{i\} - (i+1) = \emptyset$$
  Además,
  $$\mathrm{FV}(i+1) - (i+1) = \{i+1\} - (i+1) = \emptyset$$
- $a = b\,c$. Luego,
$$\mathrm{FV}(\uparrow_i(b\,c)) - (i+1) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_i(b)\ \uparrow_i(c)) - (i+1) \underset{\text{def}}{=}$$
$$(\mathrm{FV}(\uparrow_i(b)) \cup \mathrm{FV}(\uparrow_i(c))) - (i+1) \underset{\text{HI, L.1.43.1}}{=} (\mathrm{FV}(b) \cup \mathrm{FV}(c)) - (i+1) \underset{\text{def}}{=}$$
$$\mathrm{FV}(b\,c) - (i+1)$$
- $a = \lambda b$. Luego,
$$\mathrm{FV}(\uparrow_i(\lambda b)) - (i+1) \underset{\text{def}}{=} \mathrm{FV}(\lambda\ \uparrow_{i+1}(b)) - (i+1) \underset{\text{def}}{=}$$
$$(\mathrm{FV}(\uparrow_{i+1}(b)) - 1) - (i+1) \underset{\text{L.1.43.3}}{=} \mathrm{FV}(\uparrow_{i+1}(b)) - (i+2) \underset{\text{HI}}{=}$$
$$\mathrm{FV}(b) - (i+2) \underset{\text{L.1.43.3}}{=} (\mathrm{FV}(b) - 1) - (i+1) \underset{\text{def}}{=} \mathrm{FV}(\lambda b) - (i+1)$$
- $a = b[c]$. Luego,
$$\mathrm{FV}(\uparrow_i(b[c])) - (i+1) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_{i+1}(b)[\uparrow_i(c)]) - (i+1) \underset{\text{def}}{=}$$
$$(\,(\mathrm{FV}(\uparrow_{i+1}(b)) - 1) \cup \mathrm{FV}(\uparrow_i(c))\,) - (i+1) \underset{\text{L.1.43.1/3}}{=}$$
$$(\,\mathrm{FV}(\uparrow_{i+1}(b)) - (i+2)\,) \cup (\,\mathrm{FV}(\uparrow_i(c)) - (i+1)\,) \underset{\text{HI}}{=}$$
$$(\,\mathrm{FV}(b) - (i+2)\,) \cup (\,\mathrm{FV}(c) - (i+1)\,) \underset{\text{L.1.43.1/3}}{=} (\,(\mathrm{FV}(b) - 1) \cup \mathrm{FV}(c)\,) - (i+1) \underset{\text{def}}{=}$$
$$\mathrm{FV}(b[c]) - (i+1)$$

□

Lema 4.16. Para todo $a \in \Lambda re,\ i \in \mathbb{N}$, tenemos que $\mathrm{FV}(\uparrow_i(a)) - (i+1) = \mathrm{FV}(a) - i$.

Demostración. Por inducción en a. Sea $i \in \mathbb{N}$.

- $a = n \in \mathbb{N}_{>0}$. Dividimos en dos casos:

  1. $n \leq i$. Luego,

  $$\mathrm{FV}(\uparrow_i(n)) - (i+1) \underset{\text{def}}{=} \mathrm{FV}(n) - (i+1) \underset{n \leq i}{=} \emptyset$$

  Además,

  $$\mathrm{FV}(n) - i \underset{n \leq i}{=} \emptyset$$

  2. $n > i$. Luego,

  $$\mathrm{FV}(\uparrow_i(n)) \ - \ (i+1) \underset{\text{def}}{=} \mathrm{FV}(n+1) \ - \ (i+1) \underset{\text{def}}{=}$$

  $$\{n+1\} \ - \ (i+1) \underset{n > i}{=} \{n-i\}$$

  Además,

  $$\mathrm{FV}(n) - i \underset{n > i}{=} \{n-i\}$$

- $a = b\,c$. Luego,

$$\mathrm{FV}(\uparrow_i(b\,c)) - (i+1) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_i(b)\ \uparrow_i(c)) - (i+1) \underset{\text{def}}{=}$$

$$(\mathrm{FV}(\uparrow_i(b)) \cup \mathrm{FV}(\uparrow_i(c))) - (i+1) \underset{\text{HI, L.1.43.1}}{=} (\mathrm{FV}(b) \cup \mathrm{FV}(c)) - i \underset{\text{def}}{=}$$

$$\mathrm{FV}(b\,c) - i$$

- $a = \lambda b$. Luego,

$$\mathrm{FV}(\uparrow_i(\lambda b)) - (i+1) \underset{\text{def}}{=} \mathrm{FV}(\lambda\ \uparrow_{i+1}(b)) - (i+1) \underset{\text{def}}{=}$$

$$(\mathrm{FV}(\uparrow_{i+1}(b)) \ - \ 1) \ - \ (i+1) \underset{\text{L.1.43.3}}{=} \mathrm{FV}(\uparrow_{i+1}(b)) \ - \ (i+2) \underset{\text{HI}}{=}$$

$$\mathrm{FV}(b) - (i+1) \underset{\text{L.1.43.3}}{=} (\mathrm{FV}(b) - 1) - i \underset{\text{def}}{=} \mathrm{FV}(\lambda b) - i$$

- $a = b[c]$. Luego,

$$\mathrm{FV}(\uparrow_i(b[c])) - (i+1) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_{i+1}(b)[\uparrow_i(c)]) - (i+1) \underset{\text{def}}{=}$$

$$(\,(\mathrm{FV}(\uparrow_{i+1}(b)) \ - \ 1) \cup \mathrm{FV}(\uparrow_i(c))\,) \ - \ (i+1) \underset{\text{L.1.43.1/3}}{=}$$

$$(\,\mathrm{FV}(\uparrow_{i+1}(b)) \ - \ (i+2)\,) \cup (\,\mathrm{FV}(\uparrow_i(c)) \ - \ (i+1)\,) \underset{\text{HI}}{=}$$

$$(\,\mathrm{FV}(b) \ - \ (i+1)\,) \cup (\,\mathrm{FV}(c) \ - \ i\,) \underset{\text{L.1.43.1/3}}{=} (\,(\mathrm{FV}(b) \ - \ 1) \cup \mathrm{FV}(c)\,) \ - \ i \underset{\text{def}}{=}$$

$$\mathrm{FV}(b[c]) \ - \ i$$

□

Damos, ahora sí, el lema que garantiza que una reducción no agrega variables libres al reducto.

Lema 4.17 (No agregado de variables libres por reducción)**.** Para todo $a, b \in \Lambda\mathrm{re}$, tenemos que:

$$a \to_{\lambda\mathrm{re}} b \implies \mathrm{FV}(b) \subseteq \mathrm{FV}(a)$$

Demostración. Por inducción en a. Sea $b \in \Lambda\mathrm{re} : a \to_{\lambda\mathrm{re}} b$.

- $a = n \in \mathbb{N}_{>0}$. No hay reducción posible. Luego, vale vacuamente.
- $a = c\,d$. Tenemos dos casos posibles:
  1. La reducción es interna. Veamos el caso en que $c \to_{\lambda\mathrm{re}} c'$. Tenemos que, entonces, $a = c\,d \to_{\lambda\mathrm{re}} c'\,d = b$. Luego,

     $$\mathrm{FV}(c'\,d) \underset{\text{def}}{=} \mathrm{FV}(c') \cup \mathrm{FV}(d) \underset{\text{HI}}{\subseteq}$$

     $$\mathrm{FV}(c) \cup \mathrm{FV}(d) \underset{\text{def}}{=} \mathrm{FV}(c\,d)$$

     El caso en que $d \to_{\lambda\mathrm{re}} d'$ es análogo.
  2. La reducción es en la raíz. La única posibilidad es que sea por la regla (Beta). Es decir, tenemos que $c\,d = (\lambda c')\,d \underset{\text{(Beta)}}{\to_{\lambda\mathrm{re}}} c'[d] = b$. Luego,

     $$\mathrm{FV}(c'[d]) \underset{\text{def}}{=} (\mathrm{FV}(c') - 1) \cup \mathrm{FV}(d) \underset{\text{def}}{=} \mathrm{FV}(\lambda c') \cup \mathrm{FV}(d) \underset{\text{def}}{=} \mathrm{FV}((\lambda c')\,d)$$

- $a = \lambda c$. La única posibilidad es que la reducción sea interna. Es decir, $c \to_{\lambda\mathrm{re}} c'$. La prueba es análoga al caso (1) de la aplicación.
- $a = c[d]$. Si la reducción es interna (es decir, $e \to_{\lambda\mathrm{re}} e'$, $e \in \{c, d\}$), procedemos de manera similar a los casos de aplicación y abstracción. Si la reducción es en la raíz, tenemos cuatro posibilidades.
  1. La reducción es por la regla (App). Es decir, tenemos

     $$c[d] = (c_1\,c_2)[d] \underset{\text{(App)}}{\to_{\lambda\mathrm{re}}} c_1[d]\ c_2[d] = b$$

     Luego,

     $$\mathrm{FV}(c_1[d]\ c_2[d]) \underset{\text{def}}{=} \mathrm{FV}(c_1[d]) \cup \mathrm{FV}(c_2[d]) \underset{\text{def}}{=}$$

     $$[(\mathrm{FV}(c_1) - 1) \cup \mathrm{FV}(d)] \cup [(\mathrm{FV}(c_2) - 1) \cup \mathrm{FV}(d)] \underset{\text{L.1.43.1}}{=}$$

     $$[(\mathrm{FV}(c_1) \cup \mathrm{FV}(c_2)) - 1] \cup \mathrm{FV}(d) \underset{\text{def}}{=} (\mathrm{FV}(c_1\,c_2) - 1) \cup \mathrm{FV}(d) \underset{\text{def}}{=}$$

     $$\mathrm{FV}((c_1\,c_2)[d])$$

  2. La reducción es por la regla (Var). Es decir, tenemos

     $$c[d] = 1[d] \underset{\text{(Var)}}{\to_{\lambda\mathrm{re}}} d = b$$

     Luego,

     $$\mathrm{FV}(d) = (\{1\} - 1) \cup \mathrm{FV}(d) \underset{\text{def}}{=} \mathrm{FV}(1[d])$$

3. La reducción es por la regla (VarR). Es decir, tenemos

$$c[d] = (n+1)[d] \underset{\text{(VarR)}}{\rightarrow_{\lambda\text{re}}} n = b,\ n \in \mathbb{N}_{>0}$$

Luego,

$$\text{FV}(n) = \{n\} = (\{n+1\} - 1) \subseteq$$
$$(\{n+1\} - 1) \cup \text{FV}(d) \underset{\text{def}}{=} \text{FV}((n+1)[d])$$

4. La reducción es por la regla (Lamb). Es decir, tenemos

$$c[d] = (\lambda c')[d] \underset{\text{(Lamb)}}{\rightarrow_{\lambda\text{re}}} \lambda\, \uparrow_1(c')[\uparrow_0(d)] = b$$

Luego,

$$\text{FV}(\lambda\, \uparrow_1(c')[\uparrow_0(d)]) \underset{\text{def}}{=} \text{FV}(\uparrow_1(c')[\uparrow_0(d)]) - 1 \underset{\text{def}}{=}$$
$$[(\text{FV}(\uparrow_1(c')) - 1) \cup \text{FV}(\uparrow_0(d))] - 1 \underset{\text{L.1.43.1/3}}{=}$$
$$(\text{FV}(\uparrow_1(c')) - 2) \cup (\text{FV}(\uparrow_0(d)) - 1) \underset{\text{L.4.16}}{=} (\text{FV}(\uparrow_1(c')) - 2) \cup \text{FV}(d) \underset{\text{L.4.15}}{=}$$
$$(\text{FV}(c') - 2) \cup \text{FV}(d) \underset{\text{L.1.43.3, def}}{=} (\text{FV}(\lambda c') - 1) \cup \text{FV}(d) \underset{\text{def}}{=} \text{FV}((\lambda c')[d])$$

□

4.2 Isomorfismo entre λx y λre

4.2.1 Introducción

Como mencionamos en la sección anterior, el cálculo λre resulta ser isomorfo al cálculo λx. Mostramos aquí tal isomorfismo. Hasta donde sabemos, este es el primer cálculo de sustituciones explícitas con índices *à la de Bruijn* isomorfo a λx.

Es de suma importancia recalcar que, tal y como se introdujo en la sección 1.3.4 de este mismo trabajo, la principal ventaja de poseer un isomorfismo entre dos cálculos (como es el que aquí mostramos), radica en que es posible concluir que las propiedades de los cálculos son las mismas. Esto es: si el cálculo A posee la propiedad PSN, y éste es isomorfo a otro cálculo B, es trivial mostrar que B tiene también PSN. Esto es sumamente ventajoso dado que, si bien ciertas propiedades básicas son sencillas de probar, otras, como puede ser el caso de PSN, son generalmente difíciles y engorrosas; en estos casos, un isomorfismo vuelve tal tarea trivial.

4.2.2 Traducciones entre Λx y Λre

Referimos al lector a la presentación de formalismos (puntualmente, a la sección 1.3.4), de manera tal de recordar que, para mostrar que dos cálculos son isomorfos, se necesita primero dar dos traducciones (que llamaremos w y u) entre los términos de ambos cálculos. Presentamos dichas traducciones a continuación, junto con algunos lemas que nos aseguran la corrección de las definiciones (*i.e.,* que las traducciones sean, efectivamente, funciones).

Nota. *La notación elegida para una lista cuyos elementos sean $x_1, \ldots, x_n$ es $[x_1, \ldots, x_n]$. Haremos también uso de la notación $\bar{x}_{1:n}$ para denotar la misma lista; de la notación $x : \bar{x}_{1:n}$ para denotar la lista $[x, x_1, \ldots, x_n]$; y de la notación $\bar{x}_{1:n} \bullet \bar{y}_{1:m}$ para denotar la lista $[x_1, \ldots, x_n, y_1, \ldots, y_m]$. Cabe destacar que, en caso de que $m > n$, $\bar{x}_{m:n}$ denotará la lista vacía (*i.e., $[\,]$*).*

Nota. *Traducciones similares a estas son presentadas en [KR98], con la excepción del caso de la sustitución explícita. Esto último, debido a que las traducciones referidas se hacen para mostrar el isomorfismo entre el λ-cálculo clásico y el λ-cálculo* à la de Bruijn. *En estos cálculos, la sustitución se encuentra en el metalenguaje. Cabe destacar, sin embargo, que en [KR98] sí se realiza una traducción desde el cálculo λx al cálculo λt (ambos con sustituciones explícitas), sin lograr la traducción inversa.*

Comenzamos con la traducción de términos de λx a términos de λre.

Definición 4.18 (Traducción desde Λx a Λre)**.** Para todo $t \in \Lambda\mathrm{x}$, $n \in \mathbb{N}$: $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, $\mathrm{w}_{[x_1,\ldots,x_n]} : \Lambda\mathrm{x} \to \Lambda\mathrm{re}$ se define inductivamente como:

$$\begin{array}{rcll} \mathrm{w}_{\bar{x}_{1:n}}(x) & = & \mathrm{mín}\,\{j : x_j = x\} & (x \in \{x_1, \ldots, x_n\}) \\ \mathrm{w}_{\bar{x}_{1:n}}(t\,u) & = & \mathrm{w}_{\bar{x}_{1:n}}(t)\ \mathrm{w}_{\bar{x}_{1:n}}(u) & \\ \mathrm{w}_{\bar{x}_{1:n}}(\lambda x.t) & = & \lambda \mathrm{w}_{x:\bar{x}_{1:n}}(t) & \\ \mathrm{w}_{\bar{x}_{1:n}}(t[x := u]) & = & \mathrm{w}_{x:\bar{x}_{1:n}}(t)[\mathrm{w}_{\bar{x}_{1:n}}(u)] & \end{array}$$

Nota. *Notar que $w_{\bar{x}_{1:n}}$ está bien definida, dado que:*

1. $FV(\lambda x.t) \subseteq \{x_1, \ldots, x_n\} \implies FV(t) \subseteq \{x, x_1, \ldots, x_n\}$
2. $FV(t[x := v]) \subseteq \{x_1, \ldots, x_n\} \implies FV(t) \subseteq \{x, x_1, \ldots, x_n\} \wedge FV(v) \subseteq \{x_1, \ldots, x_n\}$

A continuación, mostramos que cualquier par de términos α-equivalentes de λx tienen la misma traducción a λre. Es decir, términos α-equivalentes tienen la misma imagen bajo $\mathrm{w}_{\bar{x}_{1:n}}$. Se refiere al lector a la presentación del cálculo λx (sección 1.7.2 de este trabajo, def. 1.58), o bien a [BG99], para recordar la definición de α-congruencia en dicho cálculo.

Cabe destacar que, en los siguientes dos lemas, no hacemos uso de la convención de Barendregt dado que, de proceder de esa manera, estaríamos asumiendo como hipótesis lo que queremos probar. Es decir, si aceptamos que podemos dar un nombre *fresco* a cualquier variable ligada en un término dado, estamos asumiendo necesariamente que

$$t =_\alpha u \implies \mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{x}_{1:n}}(u)$$

que es exactamente lo que queremos probar. Por ende, vamos a hacer uso de la definición de metasustitución para λx que figura en la sección 1.7.2 (puntualmente, en la definición 1.57), tomada a su vez de [BG99].

El siguiente lema nos asegura que un cambio de nombres no afecta la traducción del término en cuestión, siempre y cuando hagamos el correspondiente cambio de nombre en la lista asociada. Cabe detacar que, en la inducción que hacemos a continuación, de ser posible aplicar hipótesis inductiva sobre un término t, es también posible hacerlo sobre $t\{x := y\}$, con x e y variables, pues al sustituir una variable por otra en un término, el tamaño del mismo no cambia.

Lema 4.19. Para todo $t \in \Lambda\mathrm{x},\ n \in \mathbb{N} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, tenemos que $\forall y \notin \{x_1, \ldots, x_n\},\ z \in \{x_1, \ldots, x_n\} : \mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{x}_{1:k-1} \bullet y \bullet \bar{x}_{k+1:n}}(t\{z := y\})$, con $k = \min\{j : x_j = z\}$.

Demostración. Por inducción en t. Sea $k = \min\{j : x_j = z\}$. Por claridad, llamamos

$$\bar{y} = \bar{x}_{1:k-1} \bullet y \bullet \bar{x}_{k+1:n}$$

Es decir, tenemos que ver que: $\mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{y}}(t\{z := y\})$

- $t = x \in \{x_1, \ldots, x_n\}$. Hay dos casos:

  1. $x \neq z$. Entonces,

  $$\mathrm{w}_{\bar{x}_{1:n}}(x) \underset{\text{def}}{=} \min\{j : x_j = x\} = l \underset{x \neq z}{\neq} k$$

  Además,

  $$\mathrm{w}_{\bar{y}}(x\{z := y\}) \underset{x \neq z}{=} \mathrm{w}_{\bar{y}}(x) \underset{y \neq x}{=} \min\{j : x_j = x \wedge j \neq k\} = l$$

  2. $x = z$. Entonces,

  $$\mathrm{w}_{\bar{x}_{1:n}}(x) \underset{\text{def}}{=} \min\{j : x_j = x = z\} \underset{\text{hip}}{=} k$$

  Además,

  $$\mathrm{w}_{\bar{y}}(x\{z := y\}) \underset{x = z}{=} \mathrm{w}_{\bar{y}}(y) \underset{y \notin \{x_1, \ldots, x_n\}}{=} k$$

- $t = t_1\, t_2$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1\, t_2) = \mathrm{w}_{\bar{x}_{1:n}}(t_1)\ \mathrm{w}_{\bar{x}_{1:n}}(t_2) \underset{\text{HI}}{=}$$

$$\mathrm{w}_{\bar{y}}(t_1\{z := y\})\ \mathrm{w}_{\bar{y}}(t_2\{z := y\}) =$$

$$\mathrm{w}_{\bar{y}}(t_1\{z := y\}\ t_2\{z := y\}) = \mathrm{w}_{\bar{y}}((t_1\, t_2)\{z := y\})$$

- $t = \lambda x.u$. Dado que no trabajamos módulo α-equivalencia, tenemos dos casos.

  1. $x = z$. Luego, tenemos que, como $z \neq y \implies x \neq y$. Ahora, por definición de metasustitución para este caso, tenemos:

  $$(\lambda x.u)\{z := y\} = \lambda x.u$$

  Sea $x' \notin \{x_1, \ldots, x_n, y\}$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.u) = \lambda \mathrm{w}_{x:\bar{x}_{1:n}}(u) \underset{\text{HI}}{=} \lambda \mathrm{w}_{x':\bar{x}_{1:n}}(u\{x := x'\}) \underset{\text{HI}}{=}$$

$$\lambda \mathrm{w}_{x':\bar{y}}(u\{x := x'\}\{z := y\}) \underset{x = z}{=} \lambda \mathrm{w}_{x':\bar{y}}(u\{x := x'\}) \underset{\text{HI}}{=}$$

$$\lambda \mathrm{w}_{x:\bar{y}}(u) = \mathrm{w}_{\bar{y}}(\lambda x.u)$$

2. $x \neq z$. Luego, por definición de metasustitución,

$$(\lambda x.u)\{z := y\} = \lambda x'.u\{x := x'\}\{z := y\} \text{ con } x' \notin \{x_1, \ldots, x_n, y\}$$

Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.u) = \lambda \mathrm{w}_{x:\bar{x}_{1:n}}(u) \underset{\text{HI}}{=} \lambda \mathrm{w}_{x':\bar{x}_{1:n}}(u\{x := x'\}) \underset{\text{HI}}{=}$$

$$\lambda \mathrm{w}_{x':\bar{y}}(u\{x := x'\}\{z := y\}) = \mathrm{w}_{\bar{y}}(\lambda x'.u\{x := x'\}\{z := y\})$$

- $t = u[x := v]$. Igual que para el caso de abstracción, tenemos dos casos.

1. $x = z$. Luego, tenemos que, como $z \neq y \implies x \neq y$. Ahora, por definición de metasustitución para este caso, tenemos:

$$u[x := v]\{z := y\} = u[x := v\{z := y\}]$$

Sea $x' \notin \{x_1, \ldots, x_n, y\}$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(u[x := v]) = \mathrm{w}_{x:\bar{x}_{1:n}}(u)[\mathrm{w}_{\bar{x}_{1:n}}(v)] \underset{\text{HI}}{=} \mathrm{w}_{x':\bar{x}_{1:n}}(u\{x := x'\})[\mathrm{w}_{\bar{x}_{1:n}}(v)] \underset{\text{HI}}{=}$$

$$\mathrm{w}_{x':\bar{y}}(u\{x := x'\}\{z := y\})[\mathrm{w}_{\bar{y}}(v\{z := y\})] \underset{x = z}{=}$$

$$\mathrm{w}_{x':\bar{y}}(u\{x := x'\})[\mathrm{w}_{\bar{y}}(v\{z := y\})] \underset{\text{HI}}{=}$$

$$\mathrm{w}_{x:\bar{y}}(u)[\mathrm{w}_{\bar{y}}(v\{z := y\})] = \mathrm{w}_{\bar{y}}(u[x := v\{z := y\}])$$

2. $x \neq z$. Luego, por definición de metasustitución,

$$u[x := v]\{z := y\} = u\{x := x'\}\{z := y\}\,[x' := v\{z := y\}]$$

$$\text{con } x' \notin \{x_1, \ldots, x_n, y\}$$

Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(u[x := v]) = \mathrm{w}_{x:\bar{x}_{1:n}}(u)[\mathrm{w}_{\bar{x}_{1:n}}(v)] \underset{\text{HI}}{=} \mathrm{w}_{x':\bar{x}_{1:n}}(u\{x := x'\})[\mathrm{w}_{\bar{x}_{1:n}}(v)] \underset{\text{HI}}{=}$$

$$\mathrm{w}_{x':\bar{y}}(u\{x := x'\}\{z := y\})[\mathrm{w}_{\bar{y}}(v\{z := y\})] =$$
$$\mathrm{w}_{\bar{y}}(u\{x := x'\}\{z := y\}\,[x' := v\{z := y\}])$$

□

El lema que acabamos de mostrar nos sirve para probar el siguiente lema.

Lema 4.20. Para todo $t, u \in \Lambda\mathrm{x},\ n \in \mathbb{N},\ \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, tenemos que $t =_\alpha u \implies \mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{x}_{1:n}}(u)$. Notar que $\mathrm{w}_{\bar{x}_{1:n}}(u)$ está bien definido, pues $t =_\alpha u \implies \mathrm{FV}(t) = \mathrm{FV}(u)$.

Demostración. Por inducción en t.

- $t = x \in \{x_1, \ldots, x_n\}$. Luego, $x =_\alpha u \underset{\text{def}}{\iff} u = x$. Trivialmente, tenemos $\mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{x}_{1:n}}(u)$.

- $t = t_1\, t_2$. Luego, $t =_\alpha u \underset{\text{def}}{\iff} u = u_1\, u_2 \wedge t_1 =_\alpha u_1 \wedge t_2 =_\alpha u_2$. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1\, t_2) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1)\ \mathrm{w}_{\bar{x}_{1:n}}(t_2) \underset{\text{HI}}{=} \mathrm{w}_{\bar{x}_{1:n}}(u_1)\ \mathrm{w}_{\bar{x}_{1:n}}(u_2) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(u_1\, u_2)$$

- $t = \lambda x.v$. Luego, $t =_\alpha u \underset{\text{def}}{\iff} u = \lambda y.p\{x := y\}\ ,\ v =_\alpha p \wedge y \notin \mathrm{FV}(p) - \{x\}$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.v) \underset{\text{def}}{=} \lambda \mathrm{w}_{x:\bar{x}_{1:n}}(v) \underset{\text{HI}}{=} \lambda \mathrm{w}_{x:\bar{x}_{1:n}}(p) \underset{\text{L.4.19}}{=}$$
$$\lambda \mathrm{w}_{y:\bar{x}_{1:n}}(p\{x := y\}) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(\lambda y.p\{x := y\})$$

- $t = r[x := v]$. Luego, $t =_\alpha u \iff u = p\{x := y\}\,[y := q]$, $r =_\alpha p \wedge v =_\alpha q \wedge y \notin \mathrm{FV}(p) - \{x\}$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(r[x := v]) \underset{\text{def}}{=} \mathrm{w}_{x:\bar{x}_{1:n}}(r)[\mathrm{w}_{\bar{x}_{1:n}}(v)] \underset{\text{HI}}{=} \mathrm{w}_{x:\bar{x}_{1:n}}(p)[\mathrm{w}_{\bar{x}_{1:n}}(q)] \underset{\text{L.4.19}}{=}$$
$$\mathrm{w}_{y:\bar{x}_{1:n}}(p\{x := y\})[\mathrm{w}_{\bar{x}_{1:n}}(q)] \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(p\{x := y\}\,[y := q])$$

□

Con este lema, podemos nuevamente usar la convención de variables de Barendregt cuando trabajemos sobre la traducción $\mathrm{w}_{\bar{x}_{1:n}}$. Además, hemos probado que la traducción $\mathrm{w}_{\bar{x}_{1:n}}$ es, efectivamente, función con respecto al cociente $\Lambda\mathrm{x}\,/ =_\alpha$.

Mostramos a continuación un lema que nos dice que agregar variables detrás de la lista $[x_1, \ldots, x_n]$ no altera el resultado de la traducción $\mathrm{w}_{\bar{x}_{1:n}}$.

Lema 4.21. Para todo $t \in \Lambda\mathrm{x} :\ \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, y para todo $\{y_1, \ldots, y_m\} \subset \mathbb{V}$, tenemos que $\mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(t)$.

Demostración. Por inducción en t.

- $t = x \in \{x_1, \ldots, x_n\}$. Trivialmente, tenemos:

$$\mathrm{w}_{\bar{x}_{1:n}}(x) = \text{mín}\,\{j : x_j = x\} \underset{x\ \in\ \{x_1,\ \ldots,\ x_n\}}{=} \mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(x)$$

- $t = t_1\, t_2$. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1\, t_2) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1)\ \mathrm{w}_{\bar{x}_{1:n}}(t_2) \underset{\text{HI}}{=}$$
$$\mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(t_1)\ \mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(t_2) = \mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(t_1\, t_2)$$

- $t = \lambda x.v$. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.v) \underset{\text{def}}{=} \lambda \mathrm{w}_{x:\bar{x}_{1:n}}(v) \underset{\text{HI}}{=}$$
$$\lambda \mathrm{w}_{x:\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(v) = \mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(\lambda x.v)$$

- $t = r[x := v]$. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(r[x := v]) \underset{\text{def}}{=} \mathrm{w}_{x:\bar{x}_{1:n}}(r)[\mathrm{w}_{\bar{x}_{1:n}}(v)] \underset{\text{HI}}{=}$$
$$\mathrm{w}_{x:\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(r)[\mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(v)] = \mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(r[x := v])$$

□

Dado este último lema, pasamos a definir una traducción w uniforme (*i.e.,* no dependiente de una lista de variables $\bar{x}_{1:n}$).

Definición 4.22 (Traducción uniforme desde Λx a Λre)**.** Dada una enumeración $[v_1, v_2, \ldots]$ de $\mathbb{V}$ (prefijada), definimos, para todo $t \in \Lambda\mathrm{x},\ n \in \mathbb{N} : \mathrm{FV}(t) \subseteq \{v_1, \ldots, v_n\}$, $\mathrm{w} : \Lambda\mathrm{x} \to \Lambda\mathrm{re}$ como:

$$\mathrm{w}(t) = \mathrm{w}_{\bar{v}_{1:n}}(t)$$

Nota. *Debido al lema 4.21, la definición es correcta, pues no depende de la lista elegida, siempre y cuando la enumeración de variables se mantenga fija. Además, la traducción es independiente del representante de la clase de α-equivalencia elegido. Es decir (por lema 4.20),* $t =_\alpha u \implies w(t) = w(u)$.

Observación. Para la traducción w (desde Λx a Λre), decidimos utilizar la definición dada en [KR98]. Notamos aquí, sin embargo, que con un pequeño cambio podríamos evitar la utilización de la función mínimo para el caso base, y restringir la lista de variables a variables distintas entre sí. Esto podría lograrse "eligiendo" la variable a agregar al principio de la lista en los casos de abstracción y sustitución explícita, realizando el correspondiente cambio de nombres en la recursión. Creemos que un cambio así podría significar una reducción en la complejidad de las pruebas correspondientes. Elegimos, sin embargo, utilizar la versión original de las traducciones, mostrando aquí cómo sería la versión alternativa, pero dejando las pruebas de correctitud para un futuro, en caso de ser necesario.

Definición 4.23 (Traducción alternativa desde Λx a Λre)**.** Para todo $t \in \Lambda\mathrm{x},\ n \in \mathbb{N} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, $\mathrm{w}_{[x_1,\ldots,x_n]} : \Lambda\mathrm{x} \to \Lambda\mathrm{re}$, con $\{x_1, \ldots, x_n\}$ variables distintas entre sí, se define inductivamente como:

$$\begin{array}{rcl}
\mathrm{w}_{\bar{x}_{1:n}}(x_i) & = & i \\
\mathrm{w}_{\bar{x}_{1:n}}(t\,u) & = & \mathrm{w}_{\bar{x}_{1:n}}(t)\ \mathrm{w}_{\bar{x}_{1:n}}(u) \\
\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.t) & = & \lambda \mathrm{w}_{y:\bar{x}_{1:n}}(t\{x := y\}),\ \text{con } y \notin \{x_1, \ldots, x_n\} \\
\mathrm{w}_{\bar{x}_{1:n}}(t[x := u]) & = & \mathrm{w}_{y:\bar{x}_{1:n}}(t\{x := y\})[\mathrm{w}_{\bar{x}_{1:n}}(u)],\ \text{con } y \notin \{x_1, \ldots, x_n\}
\end{array}$$

Pasamos ahora a definir la traducción inversa a w.

Definición 4.24 (Traducción desde Λre a Λx)**.** Para todo $a \in \Lambda\mathrm{re},\ n \in \mathbb{N} : \mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, $\mathrm{u}_{[x_1,\ldots,x_n]} : \Lambda\mathrm{re} \to \Lambda\mathrm{x}$, con $\{x_1, \ldots, x_n\}$ variables distintas entre sí, se define inductivamente como:

$$\begin{array}{rcll}
\mathrm{u}_{\bar{x}_{1:n}}(j) & = & x_j & (j \in \mathbb{N}_{>0} : j \leq n) \\
\mathrm{u}_{\bar{x}_{1:n}}(a\,b) & = & \mathrm{u}_{\bar{x}_{1:n}}(a)\ \mathrm{u}_{\bar{x}_{1:n}}(b) & \\
\mathrm{u}_{\bar{x}_{1:n}}(\lambda a) & = & \lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(a) & (x \notin \{x_1, \ldots, x_n\}) \\
\mathrm{u}_{\bar{x}_{1:n}}(a[b]) & = & \mathrm{u}_{x:\bar{x}_{1:n}}(a)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(b)] & (x \notin \{x_1, \ldots, x_n\})
\end{array}$$

Nota. *Notar que* $u_{\bar{x}_{1:n}}$ *está bien definida, dado que:*

1. $FV(\lambda a) \subseteq \{1, \ldots, n\} \implies FV(a) \subseteq \{1, \ldots, n+1\}$
2. $FV(a[b]) \subseteq \{1, \ldots, n\} \implies FV(a) \subseteq \{1, \ldots, n+1\} \wedge FV(b) \subseteq \{1, \ldots, n\}$

Al igual que para la traducción anterior, debemos mostrar que esta traducción es correcta. Para eso, queda ver que la elección de x para los casos de abstracción y sustitución explícita no afecta el resultado. Esto es: términos traducidos con diferentes elecciones de la variable ligada resultan ser α-equivalentes. Mostramos dicha aserción con dos lemas.

Lema 4.25. Para todo $a \in \Lambda\mathrm{re}$, $n \in \mathbb{N}$, $\{x_1, \ldots, x_n\}$ variables distintas tal que $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, tenemos que $\forall y \notin \{x_1, \ldots, x_n\}$, $1 \leq k \leq n$:
$\mathrm{u}_{\bar{x}_{1:n}}(a)\{x_k := y\} =_\alpha \mathrm{u}_{\bar{x}_{1:k-1} \bullet y \bullet \bar{x}_{k+1:n}}(a)$.

Demostración. Por inducción en a. Por claridad, llamamos

$$\bar{y} = \bar{x}_{1:k-1} \bullet y \bullet \bar{x}_{k+1:n}$$

Es decir, tenemos que ver que: $\mathrm{u}_{\bar{x}_{1:n}}(a)\{x_k := y\} =_\alpha \mathrm{u}_{\bar{y}}(a)$

- $a = j \in \{1, \ldots, n\}$. Tenemos dos casos:
  1. $j \neq k$. Entonces,

     $$\mathrm{u}_{\bar{x}_{1:n}}(j)\{x_k := y\} \underset{\text{def}}{=} x_j\{x_k := y\} \underset{j \neq k,\ \text{hip}}{=} x_j$$

     Además,

     $$\mathrm{u}_{\bar{y}}(j) \underset{j \neq k}{=} x_j$$

     Por definición de α-equivalencia, $x_j = x_j \implies x_j =_\alpha x_j$
  2. $j = k$. Entonces,

     $$\mathrm{u}_{\bar{x}_{1:n}}(k)\{x_k := y\} \underset{\text{def}}{=} x_k\{x_k := y\} \underset{\text{def}}{=} y$$

     Además,

     $$\mathrm{u}_{\bar{y}}(k) = y$$

- $a = a_1\, a_2$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(a_1\, a_2)\{x_k := y\} = (\mathrm{u}_{\bar{x}_{1:n}}(a_1)\ \mathrm{u}_{\bar{x}_{1:n}}(a_2))\{x_k := y\} =$$

$$\mathrm{u}_{\bar{x}_{1:n}}(a_1)\{x_k := y\}\ \mathrm{u}_{\bar{x}_{1:n}}(a_2)\{x_k := y\} \underset{\text{HI}}{=_\alpha} \mathrm{u}_{\bar{y}}(a_1)\ \mathrm{u}_{\bar{y}}(a_2) = \mathrm{u}_{\bar{y}}(a_1\, a_2)$$

- $a = \lambda b$. Sean $x, z \notin \{y, x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(\lambda b)\{x_k := y\} \underset{\text{def}}{=} (\lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(b))\{x_k := y\} \underset{x \neq x_k \,\wedge\, x \neq y}{=}$$

$$\lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(b)\{x_k := y\} \underset{\text{HI}}{=_\alpha} \lambda x.\mathrm{u}_{x:\bar{y}}(b)$$

  Además,

$$\mathrm{u}_{\bar{y}}(\lambda b) \underset{\text{def}}{=} \lambda z.\mathrm{u}_{z:\bar{y}}(b) \underset{\text{def}}{=_\alpha} \lambda x.\mathrm{u}_{z:\bar{y}}(b)\{z := x\} \underset{\text{HI}}{=_\alpha} \lambda x.\mathrm{u}_{x:\bar{y}}(b)$$

- $a = b[c]$. Sean $x, z \notin \{y, x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(b[c])\{x_k := y\} \underset{\text{def}}{=} (\mathrm{u}_{x:\bar{x}_{1:n}}(b)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c)])\{x_k := y\} \underset{x \notin \{x_1, \ldots, x_n, y\}}{=}$$

$$\mathrm{u}_{x:\bar{x}_{1:n}}(b)\{x := x\}\{x_k := y\}\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c)\{x_k := y\}] \underset{\text{def}}{=}$$

$$\mathrm{u}_{x:\bar{x}_{1:n}}(b)\{x_k := y\}\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c)\{x_k := y\}] \underset{\text{HI}}{=_\alpha}$$

$$\mathrm{u}_{x:\bar{y}}(b)\,[x := \mathrm{u}_{\bar{y}}(c)]$$

  Además,

$$\mathrm{u}_{\bar{y}}(b[c]) \underset{\text{def}}{=} \mathrm{u}_{z:\bar{y}}(b)\,[z := \mathrm{u}_{\bar{y}}(c)] \underset{\text{def}}{=_\alpha}$$

$$\mathrm{u}_{z:\bar{y}}(b)\{z := x\}\,[x := \mathrm{u}_{\bar{y}}(c)] \underset{\text{HI}}{=_\alpha} \mathrm{u}_{x:\bar{y}}(b)\,[x := \mathrm{u}_{\bar{y}}(c)]$$

□

El lema que acabamos de probar nos sirve para el siguiente lema, que muestra lo que mencionamos anteriormente sobre $\mathrm{u}_{\bar{x}_{1:n}}$.

Lema 4.26. Para todo $a, b \in \Lambda\mathrm{re}$, $n \in \mathbb{N}$, $\{x_1, \ldots, x_n\}$ variables distintas, $x, y \notin \{x_1, \ldots, x_n\} : \mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, tenemos que:

1. $\lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(a) =_\alpha \lambda y.\mathrm{u}_{y:\bar{x}_{1:n}}(a)$
2. $\mathrm{u}_{x:\bar{x}_{1:n}}(a)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(b)] =_\alpha \mathrm{u}_{y:\bar{x}_{1:n}}(a)\,[y := \mathrm{u}_{\bar{x}_{1:n}}(b)]$

Demostración. (1)

$$\lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(a) \underset{\text{def}}{=_\alpha} \lambda y.\mathrm{u}_{x:\bar{x}_{1:n}}(a)\{x := y\} \underset{\text{L.4.25}}{=} \lambda y.\mathrm{u}_{y:\bar{x}_{1:n}}(a)$$

□

Demostración. (2)

$$\mathrm{u}_{x:\bar{x}_{1:n}}(a)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(b)] \underset{\text{def}}{=_\alpha} \mathrm{u}_{x:\bar{x}_{1:n}}(a)\{x := y\}\,[y := \mathrm{u}_{\bar{x}_{1:n}}(b)] \underset{\text{L.4.25}}{=}$$

$$\mathrm{u}_{y:\bar{x}_{1:n}}(a)\,[y := \mathrm{u}_{\bar{x}_{1:n}}(b)]$$

□

Con este lema hemos probado, al igual que como hicimos para la traducción anterior, que la traducción $\mathrm{u}_{\bar{x}_{1:n}}$ es función con respecto al cociente $\Lambda\mathrm{x} / =_\alpha$.

Mostramos a continuación un lema para u análogo al lema 4.21, que nos dice que el agregado de variables atrás de la lista $\bar{x}_{1:n}$ no afecta el resultado de la traducción.

Lema 4.27. Para todo $a \in \Lambda\text{re}$, $\{x_1, \ldots, x_n\}$ variables distintas tal que $\text{FV}(a) \subseteq \{1, \ldots, n\}$, y para todo $\{y_1, \ldots, y_m\}$ variables distintas tal que $\{x_1, \ldots, x_n\} \cap \{y_1, \ldots, y_m\} = \emptyset$, tenemos que $\text{u}_{\bar{x}_{1:n}}(a) =_\alpha \text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(a)$

Demostración. Por inducción en a.

- $a = j \in \{1, \ldots, n\}$. Entonces,

$$\text{u}_{\bar{x}_{1:n}}(j) \underset{\text{def}}{=} x_j \underset{\text{def}}{=} \text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(j) \implies$$

$$\text{u}_{\bar{x}_{1:n}}(j) =_\alpha \text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(j)$$

- $a = a_1\, a_2$. Entonces,

$$\text{u}_{\bar{x}_{1:n}}(a_1\, a_2) \underset{\text{def}}{=} \text{u}_{\bar{x}_{1:n}}(a_1)\ \text{u}_{\bar{x}_{1:n}}(a_2) \underset{\text{HI}}{=_\alpha}$$

$$\text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(a_1)\ \text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(a_2) \underset{\text{def}}{=} \text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(a_1\, a_2)$$

- $a = \lambda b$. Sean $x, z \notin \{x_1, \ldots, x_n\}$. Entonces,

$$\text{u}_{\bar{x}_{1:n}}(\lambda b) \underset{\text{def}}{=} \lambda x.\text{u}_{x:\bar{x}_{1:n}}(b) \underset{\text{HI}}{=_\alpha}$$

$$\lambda x.\text{u}_{x:\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(b) \underset{\text{L.4.26}}{=_\alpha} \lambda z.\text{u}_{z:\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(b) \underset{\text{def}}{=} \text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(\lambda b)$$

- $a = b[c]$. Sean $x, z \notin \{x_1, \ldots, x_n\}$. Entonces,

$$\text{u}_{\bar{x}_{1:n}}(b[c]) \underset{\text{def}}{=} \text{u}_{x:\bar{x}_{1:n}}(b)\,[x := \text{u}_{\bar{x}_{1:n}}(c)] \underset{\text{HI}}{=_\alpha}$$

$$\text{u}_{x:\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(b)\,[x := \text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(c)] \underset{\text{L.4.26}}{=_\alpha} \text{u}_{z:\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(b)\,[z := \text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(c)] \underset{\text{def}}{=}$$

$$\text{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(b[c])$$

□

Dado este último lema, pasamos a definir una traducción u uniforme (*i.e.,* no dependiente de una lista de variables $\bar{x}_{1:n}$).

Definición 4.28 (Traducción uniforme desde Λre a Λx)**.** Dada una enumeración $[v_1, v_2, \ldots]$ de $\mathbb{V}$ (prefijada), definimos, para todo $a \in \Lambda\text{re}$, $n \in \mathbb{N} : \text{FV}(a) \subseteq \{1, \ldots, n\}$, $\text{u} : \Lambda\text{re} \to \Lambda\text{x}$ como:

$$\text{u}(a) = \text{u}_{\bar{v}_{1:n}}(a)$$

Nota. *Al igual que para la traducción w, debido al lema 4.27, la definición es correcta, pues no depende de la lista elegida, siempre y cuando la enumeración de variables se mantenga fija. De la misma forma, la traducción no depende de la variable elegida en los casos de abstracción y sustitución explítica, debido al lema 4.26. Recalcamos, igualmente, que en este caso, las traducciones obtenidas no son iguales, sino que pertenecen a la misma clase de α-equivalencia.*

4.2.3 Pruebas del isomorfismo

Habiendo presentado las traducciones w y u, y probado que son efectivamente función con respecto al cociente $\Lambda\mathrm{x} \,/\, =_\alpha$, pasamos ahora a mostrar que mediante éstas, podemos ver que los cálculos λx y λre son isomorfos. Para esto, recordemos que, según lo presentado en la sección 1.3.4, tenemos que ver que:

1. $\mathrm{w} \circ \mathrm{u} = \mathrm{Id}_{\Lambda\mathrm{re}} \wedge \mathrm{u} \circ \mathrm{w} = \mathrm{Id}_{\Lambda\mathrm{x}}$
2. $\forall t, u \in \Lambda\mathrm{x} : t \to_{\lambda\mathrm{x}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{re}} \mathrm{w}(u)$
3. $\forall a, b \in \Lambda\mathrm{re} : a \to_{\lambda\mathrm{re}} b \implies \mathrm{u}(a) \to_{\lambda\mathrm{x}} \mathrm{u}(b)$

Composición de las traducciones

Mostramos, a continuación, que la composición de las traducciones es, efectivamente, la función identidad. Para esto necesitamos dos lemas, junto con otros dos lemas auxiliares.

Veamos primero los dos lemas auxiliares. Éstos hablan sobre las variables libres de un término traducido.

Lema 4.29 (Variables libres del término $\mathrm{w}_{\bar{x}_{1:n}}(t)$)**.** Para todo $t \in \Lambda\mathrm{x} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, tenemos que $\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(t)) \subseteq \{1, \ldots, n\}$.

Demostración. Por inducción en t.

- $t = x \in \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(x)) \underset{\text{def}}{=} \mathrm{FV}(\text{mín}\,\{i : x_i = x\}) = \mathrm{FV}(k) = \{k\} \underset{\text{hip}}{\subseteq} \{1, \ldots, n\}$$

- $t = u\,v$. Luego,

$$\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(u\,v)) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(u)\ \mathrm{w}_{\bar{x}_{1:n}}(v)) \underset{\text{def}}{=}$$

$$\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(u)) \cup \mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(v)) \underset{\text{HI}}{\subseteq} \{1, \ldots, n\} \cup \{1, \ldots, n\} = \{1, \ldots, n\}$$

- $t = \lambda x.u$. Luego,

$$\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.u)) \underset{\text{def}}{=} \mathrm{FV}(\lambda \mathrm{w}_{x:\bar{x}_{1:n}}(u)) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{w}_{x:\bar{x}_{1:n}}(u)) - 1 \underset{\text{HI}}{\subseteq}$$

$$\{1, \ldots, n+1\} - 1 \underset{\text{def}}{=} \{1, \ldots, n\}$$

- $t = u[x := v]$. Luego,

$$\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(u[x := v])) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{w}_{x:\bar{x}_{1:n}}(u)[\mathrm{w}_{\bar{x}_{1:n}}(v)]) \underset{\text{def}}{=}$$

$$(\mathrm{FV}(\mathrm{w}_{x:\bar{x}_{1:n}}(u)) - 1) \cup \mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(v)) \underset{\text{HI}}{\subseteq} (\{1, \ldots, n+1\} - 1) \cup \{1, \ldots, n\} =$$

$$\{1, \ldots, n\}$$

□

Lema 4.30 (Variables libres del término $\mathrm{u}_{\bar{x}_{1:n}}(a)$)**.** Para todo $a \in \Lambda\mathrm{re} : \mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, $\{x_1, \ldots, x_n\}$ un conjunto de variables distintas, tenemos que $\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(a)) \subseteq \{x_1, \ldots, x_n\}$.

Demostración. Por inducción en a. Supongamos que $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$.

- $a = j \in \{1, \ldots, n\}$. Luego,

$$\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(j)) \underset{\text{def}}{=} \mathrm{FV}(x_j) = \{x_j\} \subseteq \{x_1, \ldots, x_n\}$$

- $a = b\,c$. Luego,

$$\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(b\,c)) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(b)\ \mathrm{u}_{\bar{x}_{1:n}}(c)) \underset{\text{def}}{=}$$

$$\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(b)) \cup \mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(c)) \underset{\text{HI}}{\subseteq} \{x_1, \ldots, x_n\} \cup \{x_1, \ldots, x_n\} = \{x_1, \ldots, x_n\}$$

- $a = \lambda b$. Sea $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(\lambda b)) \underset{\text{def}}{=} \mathrm{FV}(\lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(b)) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{u}_{x:\bar{x}_{1:n}}(b)) - \{x\} \underset{\text{HI}}{\subseteq}$$

$$\{x, x_1, \ldots, x_n\} - \{x\} = \{x_1, \ldots, x_n\}$$

- $a = b[c]$. Sea $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(b[c])) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{u}_{x:\bar{x}_{1:n}}(b)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c)]) \underset{\text{def}}{=}$$

$$(\mathrm{FV}(\mathrm{u}_{x:\bar{x}_{1:n}}(b)) - \{x\}) \cup \mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(c)) \underset{\text{HI}}{\subseteq}$$

$$(\{x, x_1, \ldots, x_n\} - \{x\}) \cup \{x_1, \ldots, x_n\} = \{x_1, \ldots, x_n\}$$

□

Ahora sí, vemos los dos lemas de composición.

Lema 4.31 ($\mathrm{w} \circ \mathrm{u} = \mathrm{Id}_{\Lambda\mathrm{re}}$)**.** Para todo $a \in \Lambda\mathrm{re} : \mathrm{w}(\mathrm{u}(a)) = a$

Demostración. Por inducción en a. Sea $[x_1, x_2, \ldots]$ la enumeración de $\mathbb{V}$ elegida para w y u, con $x_i \neq x_j \iff i \neq j$. Supongamos, sin pérdida de generalidad, que $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}, n \in \mathbb{N}$. Por definición, tenemos que $\mathrm{u}(a) = \mathrm{u}_{\bar{x}_{1:n}}(a)$. Luego, por lema 4.30, tenemos que $\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(a)) \subseteq \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{w}(\mathrm{u}(a)) = \mathrm{w}(\mathrm{u}_{\bar{x}_{1:n}}(a)) = \mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(a))$$

- $a = j \in \{1, \ldots, n\}$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(j)) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(x_j) \underset{x_i\text{s diferentes}}{=} j$$

- $a = b\,c$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(b\,c)) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(b))\ \mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(c)) \underset{\text{HI}}{=} b\,c$$

- $a = \lambda b$. Sea $x \notin \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(\lambda b)) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(\lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(b)) \underset{\text{def}}{=}$$

$$\lambda \mathrm{w}_{x:\bar{x}_{1:n}}(\mathrm{u}_{x:\bar{x}_{1:n}}(b)) \underset{\text{HI}}{=} \lambda b$$

- $a = b[c]$. Sea $x \notin \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(b[c])) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{x:\bar{x}_{1:n}}(b)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c)]) \underset{\text{def}}{=}$$

$$\mathrm{w}_{x:\bar{x}_{1:n}}(\mathrm{u}_{x:\bar{x}_{1:n}}(b))[\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(c))] \underset{\text{HI}}{=} b[c]$$

□

Lema 4.32 ($\mathrm{u} \circ \mathrm{w} = \mathrm{Id}_{\Lambda \mathrm{x}}$)**.** Para todo $t \in \Lambda \mathrm{x} : \mathrm{u}(\mathrm{w}(t)) =_\alpha t$

Demostración. Por inducción en t. Sea $[x_1, x_2, \ldots]$ la enumeración de $\mathbb{V}$ elegida para w y u, con $x_i \neq x_j \iff i \neq j$. Supongamos, sin pérdida de generalidad, que $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}, n \in \mathbb{N}$. Por definición, tenemos que $\mathrm{w}(t) = \mathrm{w}_{\bar{x}_{1:n}}(t)$. Luego, por lema 4.29, tenemos que $\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(t)) \subseteq \{1, \ldots, n\}$. Entonces,

$$\mathrm{u}(\mathrm{w}(t)) = \mathrm{u}(\mathrm{w}_{\bar{x}_{1:n}}(t)) = \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(t))$$

- $t = x_j \in \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(x_j)) \underset{x_i\text{s diferentes}}{=} \mathrm{u}_{\bar{x}_{1:n}}(j) \underset{\text{def}}{=} x_j$$

  Luego, tenemos que $\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(x_j)) = x_j \implies \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(x_j)) =_\alpha x_j$

- $t = u\,v$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(u\,v)) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(u))\ \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(v)) \underset{\text{HI}}{=_\alpha} u\,v$$

- $t = \lambda x.v$. Por α-equivalencia (en particular, por lema 4.26.1), podemos asumir que $x \notin \{x_1, \ldots, x_n\}$. Sea, además $y \notin \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.v)) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(\lambda \mathrm{w}_{x:\bar{x}_{1:n}}(v)) \underset{\text{def}}{=}$$

$$\lambda y.\mathrm{u}_{y:\bar{x}_{1:n}}(\mathrm{w}_{x:\bar{x}_{1:n}}(v)) \underset{\text{L. 4.26.1}}{=_\alpha} \lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(\mathrm{w}_{x:\bar{x}_{1:n}}(v)) \underset{\text{HI}}{=_\alpha} \lambda x.v$$

- $t = u[x := v]$. Por α-equivalencia (en particular, por lema 4.26.2), podemos asumir que $x \notin \{x_1, \ldots, x_n\}$. Sea, además $y \notin \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(u[x := v])) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{x:\bar{x}_{1:n}}(u)[\mathrm{w}_{\bar{x}_{1:n}}(v)]) \underset{\text{def}}{=}$$

$$\mathrm{u}_{y:\bar{x}_{1:n}}(\mathrm{w}_{x:\bar{x}_{1:n}}(u))\,[y := \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(v))] \underset{\text{L. 4.26.2}}{=_\alpha}$$

$$\mathrm{u}_{x:\bar{x}_{1:n}}(\mathrm{w}_{x:\bar{x}_{1:n}}(u))\,[x := \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(v))] \underset{\text{HI}}{=_\alpha} u[x := v]$$

□

Un paso de reducción en λx implica un paso de reducción en λre

Veremos a continación una serie de lemas auxiliares que nos permiten demostrar que, cuando un término de λx reduce a otro, la traducción del primero a λre reduce a la traducción del segundo. Este es el segundo de los pasos necesarios para mostrar el isomorfismo entre ambos cálculos.

Lema 4.33. Sea $\{x_1, \ldots, x_n\}$ un conjunto de variables. Sean, además, $y \in \{x_1, \ldots, x_n\}$, $x \notin \{x_1, \ldots, x_n\}$. Luego, $\mathrm{w}_{x:\bar{x}_{1:n}}(y) = \mathrm{w}_{\bar{x}_{1:n}}(y) + 1$.

Demostración. Directa. Por hipótesis, tenemos que $x \neq y$. Como $y \in \{x_1, \ldots, x_n\}$, tenemos que:

$$\mathrm{w}_{\bar{x}_{1:n}}(y) \underset{\text{def}}{=} \text{mín}\,\{j : x_j = y\} = k \in \mathbb{N}_{>0}$$

Ahora,

$$\mathrm{w}_{x:\bar{x}_{1:n}}(y) \underset{\text{def}}{=} \begin{cases} 1 & \text{si } x = y \\ \text{mín}\,\{j+1 : x_j = y\} = k+1 & \text{si } x \neq y \end{cases}$$

Recordemos que $x \neq y$. Luego, tenemos $\mathrm{w}_{x:\bar{x}_{1:n}}(y) = k + 1 = \mathrm{w}_{\bar{x}_{1:n}}(y) + 1$. □

El siguiente lema nos muestra cómo interactúan la traducción w y los *swaps*.

Lema 4.34 (Interacción entre la traducción w y $\updownarrow_i$)**.** Sea $t \in \Lambda\mathrm{x} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Sea $i \in \mathbb{N}_{>0} : i < n \wedge x_i \neq x_{i+1}$. Luego,

$$\updownarrow_i(\mathrm{w}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(t)) = \mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(t)$$

Demostración. Por inducción en t.

- $t = x \in \{x_1, \ldots, x_n\}$. Supongamos que $\mathrm{w}_{\bar{x}_{1:n}}(x) = k,\ 1 \leq k \leq n$. O sea, tenemos que $x_k = x$. Luego, $\updownarrow_i(\mathrm{w}_{\bar{x}_{1:n}}(t)) = \updownarrow_i(k)$. Tenemos cuatro casos:

  1. $k < i \implies \updownarrow_i(k) \underset{\text{def}}{=} k$. Luego,

  $$\mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(x_k) \underset{\text{def}}{=} \text{mín}\,\{j : x_j = x\} \underset{k\,<\,i}{=} k$$

  2. $k > i+1 \implies \updownarrow_i(k) \underset{\text{def}}{=} k$. Además, tenemos que $x_k \neq x_i \wedge x_k \neq x_{i+1}$. Luego,

  $$\mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(x_k) \underset{\text{def}}{=} \text{mín}\,\{j : x_j = x\} \underset{\substack{k\,>\,i+1\,\wedge \\ x_k \neq x_i\,\wedge\,x_k \neq x_{i+1}}}{=} k$$

  3. $k = i \implies \updownarrow_i(k) \underset{\text{def}}{=} i+1$. Tenemos que $x_k = x_i$. Recordemos, también, que $x_i \neq x_{i+1}$. Luego,

  $$\mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(x_i) \underset{\text{def}}{=} i+1$$

  4. $k = i+1 \implies \updownarrow_i(k) \underset{\text{def}}{=} i$. Tenemos que $x_k = x_{i+1}$. Luego,

  $$\mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(x_{i+1}) \underset{\text{def}}{=} i$$

  De aquí en más, llamamos $\bar{y}$ a $[x_1, \ldots, x_{i+1}, x_i, \ldots, x_n]$.

- $t = u\,v$. Luego,

$$\uparrow_i(\mathrm{w}_{\bar{x}_{1:n}}(u\,v)) \underset{\text{def}}{=} \uparrow_i(\mathrm{w}_{\bar{x}_{1:n}}(u)\ \mathrm{w}_{\bar{x}_{1:n}}(v)) \underset{\text{def}}{=}$$

$$\uparrow_i(\mathrm{w}_{\bar{x}_{1:n}}(u))\ \uparrow_i(\mathrm{w}_{\bar{x}_{1:n}}(v)) \underset{\text{HI}}{=} \mathrm{w}_{\bar{y}}(u)\ \mathrm{w}_{\bar{y}}(v) \underset{\text{def}}{=} \mathrm{w}_{\bar{y}}(u\,v)$$

- $t = \lambda x.u$. Luego,

$$\uparrow_i(\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.u)) \underset{\text{def}}{=} \uparrow_i(\lambda \mathrm{w}_{x:\bar{x}_{1:n}}(u)) \underset{\text{def}}{=}$$

$$\lambda\uparrow_{i+1}(\mathrm{w}_{x:\bar{x}_{1:n}}(u)) \underset{\text{HI}}{=} \lambda \mathrm{w}_{x:\bar{y}}(u) \underset{\text{def}}{=} \mathrm{w}_{\bar{y}}(\lambda x.u)$$

- $t = u[x := v]$. Luego,

$$\uparrow_i(\mathrm{w}_{\bar{x}_{1:n}}(u[x := v])) \underset{\text{def}}{=} \uparrow_i(\mathrm{w}_{x:\bar{x}_{1:n}}(u)[\mathrm{w}_{\bar{x}_{1:n}}(v)]) \underset{\text{def}}{=}$$

$$\uparrow_{i+1}(\mathrm{w}_{x:\bar{x}_{1:n}}(u))[\uparrow_i(\mathrm{w}_{\bar{x}_{1:n}}(v))] \underset{\text{HI}}{=} \mathrm{w}_{x:\bar{y}}(u)[\mathrm{w}_{\bar{y}}(v)] \underset{\text{def}}{=} \mathrm{w}_{\bar{y}}(u[x := v])$$

□

El siguiente lema muestra la interacción entre la traducción w y los incrementos de índice. Cabe destacar que podemos encontrar una prueba similiar en el lema 13 de [KR98], aunque ésta se hace para la función de actualización clásica de λ_{dB} (U^i_k), y sólo sobre los términos del conjunto Λ_{dB} (*i.e.,* no hay caso de sustitución explícita).

Lema 4.35 (Interacción entre la traducción w y $\uparrow_i$)**.** Sea $t \in \Lambda\mathrm{x}$: $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Sea $m \in \mathbb{N} : m \leq n$. Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{w}_{\bar{x}_{1:m} \bullet x \bullet \bar{x}_{m+1:n}}(t) = \uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(t))$$

Demostración. Por inducción en t. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m} \bullet x \bullet \bar{x}_{m+1:n}$. Entonces, tenemos que ver que:

$$\mathrm{w}_{\bar{z}}(t) = \uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(t))$$

- $t = y \in \{x_1, \ldots, x_n\}$. Por hipótesis, $y \neq x$. Hay dos casos:

  1. $y \in \{x_1, \ldots, x_m\}$. Entonces, $\mathrm{w}_{\bar{z}}(y) \underset{\text{def}}{=} k \leq m$. Luego,

$$\uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(y)) \underset{\text{def}}{=} \uparrow_m(k) \underset{\text{def, } k \leq m}{=} k$$

  2. $y \notin \{x_1, \ldots, x_m\}$. Entonces, $\mathrm{w}_{\bar{z}}(y) \underset{\text{def}}{=} k > m + 1$. Luego,

$$\uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(y)) \underset{\text{def}}{=} \uparrow_m(k-1) \underset{\text{def, } k-1 > m}{=} k$$

- $t = u\,v$. Luego,

$$\uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(u\,v)) \underset{\text{def}}{=} \uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(u)\ \mathrm{w}_{\bar{x}_{1:n}}(v)) \underset{\text{def}}{=} \uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(u))\ \uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(v)) \underset{\text{HI}}{=}$$

$$\mathrm{w}_{\bar{z}}(u)\ \mathrm{w}_{\bar{z}}(v) \underset{\text{def}}{=} \mathrm{w}_{\bar{z}}(u\,v)$$

- $t = \lambda y.u$. Por convención de variables, podemos asumir $y \neq x$. Luego,

$$\uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(\lambda y.u)) \underset{\text{def}}{=} \uparrow_m(\lambda \mathrm{w}_{y:\bar{x}_{1:n}}(u)) \underset{\text{def}}{=} \lambda \uparrow_{m+1}(\mathrm{w}_{y:\bar{x}_{1:n}}(u)) \underset{\text{HI}}{=}$$

$$\lambda \mathrm{w}_{y:\bar{z}}(u) \underset{\text{def}}{=} \mathrm{w}_{\bar{z}}(\lambda y.u)$$

- $t = u[y := v]$. Al igual que en el caso anterior, podemos asumir $y \neq x$. Luego,

$$\uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(u[y := v])) \underset{\text{def}}{=} \uparrow_m(\mathrm{w}_{y:\bar{x}_{1:n}}(u)[\mathrm{w}_{\bar{x}_{1:n}}(v)]) \underset{\text{def}}{=}$$

$$\uparrow_{m+1}(\mathrm{w}_{y:\bar{x}_{1:n}}(u))[\uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(v))] \underset{\text{HI}}{=} \mathrm{w}_{y:\bar{z}}(u)[\mathrm{w}_{\bar{z}}(v)] \underset{\text{def}}{=} \mathrm{w}_{\bar{z}}(u[y := v])$$

□

Veamos, ahora sí, el lema principal.

Lema 4.36 (Preservación de la reducción λx bajo w)**.** Para todo $t, u \in \Lambda\mathrm{x}$, tenemos que:

$$t \to_{\lambda\mathrm{x}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{re}} \mathrm{w}(u)$$

Demostración. Por inducción en t. Sin pérdida de generalidad, supongamos que $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Como $t \to_{\lambda\mathrm{x}} u$, tenemos que $\mathrm{FV}(u) \subseteq \mathrm{FV}(t)$. Sean $x_1, \ldots, x_n$ las primeras n variables de la enumeración de $\mathbb{V}$ elegida para w. Luego, tenemos que $\mathrm{w}(t) = \mathrm{w}_{\bar{x}_{1:n}}(t)$ y que $\mathrm{w}(u) = \mathrm{w}_{\bar{x}_{1:n}}(u)$.

- $t = x \in \{x_1, \ldots, x_n\}$. No hay reducción posible. Luego, vale vacuamente.

- $t = t_1\, t_2$. Tenemos dos casos posibles:

  1. La reducción es interna. Veamos el caso en que $t_1 \to_{\lambda\mathrm{x}} t_1'$. Tenemos que, entonces, $t = t_1\, t_2 \to_{\lambda\mathrm{x}} t_1'\, t_2 = u$. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1\, t_2) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1)\ \mathrm{w}_{\bar{x}_{1:n}}(t_2) \underset{\text{HI}}{\to_{\lambda\mathrm{re}}}$$

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1')\ \mathrm{w}_{\bar{x}_{1:n}}(t_2) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1'\, t_2)$$

     El caso en que $t_2 \to_{\lambda\mathrm{x}} t_2'$ es análogo.

  2. La reducción es en la raíz. La única posibilidad es que sea por la regla (Beta). Es decir, tenemos que $t_1\, t_2 = (\lambda x.t_1')\, t_2 \underset{\text{(Beta)}}{\to_{\lambda\mathrm{x}}} t_1'[x := t_2] = u$. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}((\lambda x.t_1')\, t_2) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(\lambda x.t_1')\ \mathrm{w}_{\bar{x}_{1:n}}(t_2) \underset{\text{def}}{=}$$

$$(\lambda \mathrm{w}_{x:\bar{x}_{1:n}}(t_1'))\, \mathrm{w}_{\bar{x}_{1:n}}(t_2) \underset{\text{(Beta)}}{\to_{\lambda\mathrm{re}}} \mathrm{w}_{x:\bar{x}_{1:n}}(t_1')[\mathrm{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=}$$

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1'[x := t_2])$$

- $t = \lambda x.t_1$. La única posibilidad es que la reducción sea interna. Es decir, $t_1 \to_{\lambda\mathrm{x}} t_1'$. La prueba es análoga al caso (1) de la aplicación.

- $t = t_1[x := t_2]$. Si la reducción es interna (es decir, $t_i \to_{\lambda\mathrm{x}} t_i',\ i \in \{1, 2\}$), procedemos de manera similar a los casos de aplicación y abstracción. Si la reducción es en la raíz, tenemos cuatro posibilidades.

1. La reducción es por la regla (App). Es decir, tenemos

$$t_1[x := t_2] = (t_1'\ t_1'')[x := t_2] \underset{\text{(App)}}{\rightarrow_{\lambda\text{x}}} t_1'[x := t_2]\ t_1''[x := t_2] = u$$

   Luego,

$$\text{w}_{\bar{x}_{1:n}}((t_1'\ t_1'')[x := t_2]) \underset{\text{def}}{=} \text{w}_{x:\bar{x}_{1:n}}(t_1'\ t_1'')[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=}$$

$$(\text{w}_{x:\bar{x}_{1:n}}(t_1')\ \text{w}_{x:\bar{x}_{1:n}}(t_1''))[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{(App)}}{\rightarrow_{\lambda\text{re}}}$$

$$\text{w}_{x:\bar{x}_{1:n}}(t_1')[\text{w}_{\bar{x}_{1:n}}(t_2)]\ \text{w}_{x:\bar{x}_{1:n}}(t_1'')[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=}$$

$$\text{w}_{\bar{x}_{1:n}}(t_1'[x := t_2])\ \text{w}_{\bar{x}_{1:n}}(t_1''[x := t_2]) \underset{\text{def}}{=} \text{w}_{\bar{x}_{1:n}}(t_1'[x := t_2]\ t_1''[x := t_2])$$

2. La reducción es por la regla (Var). Es decir, tenemos

$$t_1[x := t_2] = x[x := t_2] \underset{\text{(Var)}}{\rightarrow_{\lambda\text{x}}} t_2 = u$$

   Luego,

$$\text{w}_{\bar{x}_{1:n}}(x[x := t_2]) \underset{\text{def}}{=} \text{w}_{x:\bar{x}_{1:n}}(x)[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=}$$

$$1[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{(Var)}}{\rightarrow_{\lambda\text{re}}} \text{w}_{\bar{x}_{1:n}}(t_2)$$

3. La reducción es por la regla (VarR). Es decir, tenemos

$$t_1[x := t_2] = y[x := t_2] \underset{\text{(VarR)}}{\rightarrow_{\lambda\text{x}}} y = u, \text{ con } x \neq y$$

   Luego,

$$\text{w}_{\bar{x}_{1:n}}(y[x := t_2]) \underset{\text{def}}{=} \text{w}_{x:\bar{x}_{1:n}}(y)[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def, } x \neq y}{=}$$

$$(k+1)[\text{w}_{\bar{x}_{1:n}}(t_2)]\,, \text{con } k \in \mathbb{N}_{>0}$$

   Además,

$$(k+1)[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{(VarR)}}{\rightarrow_{\lambda\text{re}}} k \underset{\text{L. 4.33}}{=} \text{w}_{\bar{x}_{1:n}}(y)$$

4. La reducción es por la regla (Lamb). Es decir, tenemos

$$t_1[x := t_2] = (\lambda y.t_1')[x := t_2] \underset{\text{(Lamb)}}{\rightarrow_{\lambda\text{x}}} \lambda y.t_1'[x := t_2] = u$$

   Cabe destacar, al igual que para el caso anterior, que por convención de variables, asumimos $x \neq y \wedge y \notin \text{FV}(t_2)$. Luego,

$$\text{w}_{\bar{x}_{1:n}}((\lambda y.t_1')[x := t_2]) \underset{\text{def}}{=} \text{w}_{x:\bar{x}_{1:n}}(\lambda y.t_1')[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=}$$

$$(\lambda \text{w}_{y:x:\bar{x}_{1:n}}(t_1'))[\text{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{(Lamb)}}{\rightarrow_{\lambda\text{re}}}$$

$$\lambda \uparrow_1(\text{w}_{y:x:\bar{x}_{1:n}}(t_1'))[\uparrow_0(\text{w}_{\bar{x}_{1:n}}(t_2))] \underset{\text{L.4.34, } x \neq y}{=}$$

$$\lambda \text{w}_{x:y:\bar{x}_{1:n}}(t_1')[\uparrow_0(\text{w}_{\bar{x}_{1:n}}(t_2))] \underset{\text{L. 4.35, } y \notin \text{FV}(t_2)}{=}$$

$$\lambda \text{w}_{x:y:\bar{x}_{1:n}}(t_1')[\text{w}_{y:\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=}$$

$$\lambda \text{w}_{y:\bar{x}_{1:n}}(t_1'[x := t_2]) \underset{\text{def}}{=} \text{w}_{\bar{x}_{1:n}}(\lambda y.t_1'[x := t_2])$$

□

Un paso de reducción en λre implica un paso de reducción en λx

Al igual que para la subsección anterior, necesitamos algunos lemas auxiliares para poder mostrar lo inverso a lo ya expuesto. Es decir, que cuando un término de λre reduce a otro, la traducción del primero a λx va a un término que hace lo propio con respecto a la traducción del segundo. Con la demostración de esta afirmación, tenemos los elementos necesarios para afirmar fehacientemente que los cálculos en cuestión son isomorfos.

El siguiente lema nos muestra cómo interactúan la traducción u y los *swaps*.

Lema 4.37 (Interacción entre la traducción u y $\updownarrow_i$)**.** Sea $a \in \Lambda re$: $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$. Sea $i \in \mathbb{N}_{>0} : i < n$. Sean, además, $\{x_1, \ldots, x_n\}$ variables distintas. Luego,

$$\mathrm{u}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(a) =_\alpha \mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\updownarrow_i(a))$$

Demostración. Por inducción en a.

- $a = k \in \{1, \ldots, n\}$. Por definición, tenemos que $\mathrm{u}_{\bar{x}_{1:n}}(a) = x_k$. Tenemos tres casos:

  1. $k < i \vee k > i + 1$. Luego,

  $$\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\updownarrow_i(k)) \underset{\text{def}}{=} \mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(k) \underset{k < i \vee k > i+1}{=} x_k$$

  2. $k = i$. Luego,

  $$\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\updownarrow_i(i)) \underset{\text{def}}{=} \mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(i+1) \underset{\text{def}}{=} x_i = x_k$$

  3. $k = i + 1$. Luego,

  $$\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\updownarrow_i(i+1)) \underset{\text{def}}{=} \mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(i) \underset{\text{def}}{=} x_{i+1} = x_k$$

  Por definición de α-equivalencia, $x_k =_\alpha x_k$.

  De aquí en más, llamamos $\bar{y}$ a $[x_1, \ldots, x_{i+1}, x_i, \ldots, x_n]$.

- $a = b\,c$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}(b\,c) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(b)\ \mathrm{u}_{\bar{x}_{1:n}}(c) \underset{\text{HI}}{=_\alpha}$$
$$\mathrm{u}_{\bar{y}}(\updownarrow_i(b))\ \mathrm{u}_{\bar{y}}(\updownarrow_i(c)) \underset{\text{def}}{=} \mathrm{u}_{\bar{y}}(\updownarrow_i(b\,c))$$

- $a = \lambda b$. Sea $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}(\lambda b) \underset{\text{def}}{=} \lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(b) \underset{\text{HI}}{=_\alpha}$$
$$\lambda x.\mathrm{u}_{x:\bar{y}}(\updownarrow_{i+1}(b)) \underset{\text{def}}{=_\alpha} \mathrm{u}_{\bar{y}}(\lambda\, \updownarrow_{i+1}(b)) \underset{\text{def}}{=} \mathrm{u}_{\bar{y}}(\updownarrow_i(\lambda b))$$

- $a = b[c]$. Sea $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}(b[c]) \underset{\text{def}}{=} \mathrm{u}_{x:\bar{x}_{1:n}}(b)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c)] \underset{\text{HI}}{=_\alpha}$$
$$\mathrm{u}_{x:\bar{y}}(\updownarrow_{i+1}(b))\,[x := \mathrm{u}_{\bar{y}}(\updownarrow_i(c))] \underset{\text{def}}{=_\alpha} \mathrm{u}_{\bar{y}}(\updownarrow_{i+1}(b)[\updownarrow_i(c)]) \underset{\text{def}}{=} \mathrm{u}_{\bar{y}}(\updownarrow_i(b[c]))$$

□

El siguiente lema muestra la interacción entre la traducción u y los incrementos de índice. Cabe destacar que podemos encontrar una prueba similiar en el lema 15 de [KR98], aunque – al igual que lo que sucede con el lema 13 de ese mismo artículo – ésta se hace para la función de actualización clásica de λ_{dB} (U^i_k), y sólo sobre los términos del conjunto Λ_{dB} (*i.e.,* no hay caso de sustitución explícita).

Lema 4.38 (Interacción entre la traducción u y $\uparrow_i$)**.** Sea $a \in \Lambda\mathrm{re}$: $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$. Sean $\{x_1, \ldots, x_n\}$ variables distintas, y sea $m \in \mathbb{N} : m \leq n$. Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{x}_{1:m} \bullet x \bullet \bar{x}_{m+1:n}}(\uparrow_m(a)) =_\alpha \mathrm{u}_{\bar{x}_{1:n}}(a)$$

Demostración. Por inducción en a. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m} \bullet x \bullet \bar{x}_{m+1:n}$. Entonces, tenemos que ver que:

$$\mathrm{u}_{\bar{z}}(\uparrow_m(a)) =_\alpha \mathrm{u}_{\bar{x}_{1:n}}(a)$$

- $a = k \in \{1, \ldots, n\}$. Dos casos:

  1. $k \leq m$. Luego,

  $$\mathrm{u}_{\bar{z}}(\uparrow_m(k)) \underset{\text{def}}{=} \mathrm{u}_{\bar{z}}(k) \underset{k \leq m}{=} x_k = \mathrm{u}_{\bar{x}_{1:n}}(k)$$

  2. $k > m$. Luego,

  $$\mathrm{u}_{\bar{z}}(\uparrow_m(k)) \underset{\text{def}}{=} \mathrm{u}_{\bar{z}}(k+1) \underset{k+1 > m+1}{=} x_k = \mathrm{u}_{\bar{x}_{1:n}}(k)$$

  Por definición de α-equivalencia, $x_k =_\alpha x_k$.

- $a = b\,c$. Luego,

$$\mathrm{u}_{\bar{z}}(\uparrow_m(b\,c)) \underset{\text{def}}{=} \mathrm{u}_{\bar{z}}(\uparrow_m(b)\ \uparrow_m(c)) \underset{\text{def}}{=} \mathrm{u}_{\bar{z}}(\uparrow_m(b))\ \mathrm{u}_{\bar{z}}(\uparrow_m(c)) \underset{\text{HI}}{=_\alpha}$$

$$\mathrm{u}_{\bar{x}_{1:n}}(b)\ \mathrm{u}_{\bar{x}_{1:n}}(c) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(b\,c)$$

- $a = \lambda b$. Sea $y \notin \{x, x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{z}}(\uparrow_m(\lambda b)) \underset{\text{def}}{=} \mathrm{u}_{\bar{z}}(\lambda\ \uparrow_{m+1}(b)) \underset{\text{def}}{=} \lambda y.\mathrm{u}_{y:\bar{z}}(\uparrow_{m+1}(b)) \underset{\text{HI}}{=_\alpha}$$

$$\lambda y.\mathrm{u}_{y:\bar{x}_{1:n}}(b) \underset{\text{def}}{=_\alpha} \mathrm{u}_{\bar{x}_{1:n}}(\lambda b)$$

- $a = b[c]$. Sea $y \notin \{x, x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{z}}(\uparrow_m(b[c])) \underset{\text{def}}{=} \mathrm{u}_{\bar{z}}(\uparrow_{m+1}(b)[\uparrow_m(c)]) \underset{\text{def}}{=} \mathrm{u}_{y:\bar{z}}(\uparrow_{m+1}(b))\,[y := \mathrm{u}_{\bar{z}}(\uparrow_m(c))] \underset{\text{HI}}{=_\alpha}$$

$$\mathrm{u}_{y:\bar{x}_{1:n}}(b)\,[y := \mathrm{u}_{\bar{x}_{1:n}}(c)] \underset{\text{def}}{=_\alpha} \mathrm{u}_{\bar{x}_{1:n}}(b[c])$$

□

Veamos, ahora sí, el lema principal.

Lema 4.39 (Preservación de la reducción λre bajo u)**.** Para todo $a, b \in \Lambda re$, tenemos que:

$$a \to_{\lambda re} b \implies \mathrm{u}(a) \to_{\lambda x} \mathrm{u}(b)$$

Demostración. Por inducción en a. Sin pérdida de generalidad, supongamos que $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$. Como $a \to_{\lambda re} b$, tenemos que, por lema 4.17, $\mathrm{FV}(b) \subseteq \mathrm{FV}(a)$. Sean $x_1, \ldots, x_n$ las primeras n variables de la enumeración de $\mathbb{V}$ elegida para u. Luego, tenemos que $\mathrm{u}(a) = \mathrm{u}_{\bar{x}_{1:n}}(a)$ y que $\mathrm{u}(b) = \mathrm{u}_{\bar{x}_{1:n}}(b)$.

- $a = k \in \{1, \ldots, n\}$. No hay reducción posible. Luego, vale vacuamente.
- $a = c_1\, c_2$. Tenemos dos casos posibles:
  1. La reducción es interna. Veamos el caso en que $c_1 \to_{\lambda re} c_1'$. Tenemos que, entonces, $a = c_1\, c_2 \to_{\lambda re} c_1'\, c_2 = b$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}(c_1\, c_2) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(c_1)\ \mathrm{u}_{\bar{x}_{1:n}}(c_2) \underset{\text{HI}}{\to_{\lambda x}}$$

$$\mathrm{u}_{\bar{x}_{1:n}}(c_1')\ \mathrm{u}_{\bar{x}_{1:n}}(c_2) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(c_1'\, c_2)$$

     El caso en que $c_2 \to_{\lambda re} c_2'$ es análogo.
  2. La reducción es en la raíz. La única posibilidad es que sea por la regla (Beta). Es decir, tenemos que $c_1\, c_2 = (\lambda c_1')\, c_2 \underset{\text{(Beta)}}{\to_{\lambda re}} c_1'[c_2] = b$. Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}((\lambda c_1')\, c_2) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(\lambda c_1')\ \mathrm{u}_{\bar{x}_{1:n}}(c_2) \underset{\text{def}}{=}$$

$$(\lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(c_1'))\, \mathrm{u}_{\bar{x}_{1:n}}(c_2) \underset{\text{(Beta)}}{\to_{\lambda x}} \mathrm{u}_{x:\bar{x}_{1:n}}(c_1')\, [x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{def}}{=}$$

$$\mathrm{u}_{\bar{x}_{1:n}}(c_1'[c_2])$$

- $a = \lambda c$. La única posibilidad es que la reducción sea interna. Es decir, $c \to_{\lambda re} c'$. La prueba es análoga al caso (1) de la aplicación.
- $a = c_1[c_2]$. Si la reducción es interna (es decir, $c_i \to_{\lambda re} c_i',\ i \in \{1, 2\}$), procedemos de manera similar a los casos de aplicación y abstracción. Si la reducción es en la raíz, tenemos cuatro posibilidades.
  1. La reducción es por la regla (App). Es decir, tenemos

$$c_1[c_2] = (c_1'\, c_1'')[c_2] \underset{\text{(App)}}{\to_{\lambda re}} c_1'[c_2]\ c_1''[c_2] = b$$

     Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}((c_1'\, c_1'')[c_2]) \underset{\text{def}}{=} \mathrm{u}_{x:\bar{x}_{1:n}}(c_1'\, c_1'')\, [x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{def}}{=}$$

$$(\mathrm{u}_{x:\bar{x}_{1:n}}(c_1')\ \mathrm{u}_{x:\bar{x}_{1:n}}(c_1''))[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{(App)}}{\to_{\lambda x}}$$

$$\mathrm{u}_{x:\bar{x}_{1:n}}(c_1')\, [x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)]\, \mathrm{u}_{x:\bar{x}_{1:n}}(c_1'')\, [x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{def}}{=}$$

$$\mathrm{u}_{\bar{x}_{1:n}}(c_1'[c_2])\ \mathrm{u}_{\bar{x}_{1:n}}(c_1''[c_2]) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(c_1'[c_2]\ c_1''[c_2])$$

2. La reducción es por la regla (Var). Es decir, tenemos

$$c_1[c_2] = 1[c_2] \underset{\text{(Var)}}{\to_{\lambda\text{re}}} c_2 = b$$

Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}(1[c_2]) \underset{\text{def}}{=} \mathrm{u}_{x:\bar{x}_{1:n}}(1)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{def}}{=}$$

$$x[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{(Var)}}{\to_{\lambda\text{x}}} \mathrm{u}_{\bar{x}_{1:n}}(c_2)$$

3. La reducción es por la regla (VarR). Es decir, tenemos

$$c_1[c_2] = (k+1)[c_2] \underset{\text{(VarR)}}{\to_{\lambda\text{re}}} k = b, \text{ con } k \in \mathbb{N}_{>0}$$

Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}((k+1)[c_2]) \underset{\text{def}}{=} \mathrm{u}_{x:\bar{x}_{1:n}}(k+1)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{def, } k+1>1}{=}$$

$$x_k[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{(VarR), } x \neq x_k}{\to_{\lambda\text{x}}} x_k \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(k)$$

4. La reducción es por la regla (Lamb). Es decir, tenemos

$$c_1[c_2] = (\lambda c_1')[c_2] \underset{\text{(Lamb)}}{\to_{\lambda\text{re}}} \lambda\, \updownarrow_1(c_1')[\uparrow_0(c_2)] = b$$

Sean, además, $x, y \notin \{x_1, \ldots, x_n\}$, $x \neq y$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}((\lambda c_1')[c_2]) \underset{\text{def}}{=} \mathrm{u}_{x:\bar{x}_{1:n}}(\lambda c_1')\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{def}}{=}$$

$$(\lambda y.\mathrm{u}_{y:x:\bar{x}_{1:n}}(c_1'))[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{(Lamb), } y \notin \{x, x_1, \ldots, x_n\}}{\to_{\lambda\text{x}}}$$

$$\lambda y.\mathrm{u}_{y:x:\bar{x}_{1:n}}(c_1')\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{L.4.37}}{=}$$

$$\lambda y.\mathrm{u}_{x:y:\bar{x}_{1:n}}(\updownarrow_1(c_1'))\,[x := \mathrm{u}_{\bar{x}_{1:n}}(c_2)] \underset{\text{L.4.38}}{=}$$

$$\lambda y.\mathrm{u}_{x:y:\bar{x}_{1:n}}(\updownarrow_1(c_1'))\,[x := \mathrm{u}_{y:\bar{x}_{1:n}}(\uparrow_0(c_2))] \underset{\text{def}}{=}$$

$$\lambda y.\mathrm{u}_{y:\bar{x}_{1:n}}(\updownarrow_1(c_1')[\uparrow_0(c_2)]) \underset{\text{def}}{=}$$

$$\mathrm{u}_{\bar{x}_{1:n}}(\lambda\, \updownarrow_1(c_1')[\uparrow_0(c_2)])$$

□

Enunciamos y demostramos, finalmente, el teorema principal de esta sección.

Teorema 4.40 (Isomorfismo entre λx y λre)**.** Los cálculos λx y λre son isomorfos.

Demostración. Consecuencia directa de la definición de isomorfismo, de las traducciones expuestas (w y u), y de los lemas 4.31, 4.32, 4.36 y 4.39. □

Corolario 4.41. Los cálculos λre y λx tienen las mismas propiedades.

Demostración. Consecuencia directa del teorema 4.40 □

Tal y como mencionamos en la introducción a esta sección, y como reza el corolario 4.41, los cálculos λx y λre tienen las mismas propiedades. Esto es, hemos probado que nuestro cálculo posee (ver sección 1.7.2): (Sim), (Snd), ($\mathrm{CR_{re}}$), ($\mathrm{SN_{re}}$), (CR) y (PSN).

Notamos que, para el caso particular de las propiedades ($\mathrm{CR_{re}}$) y ($\mathrm{SN_{re}}$), habría que probar antes que los subcálculos de sustituciones explícitas son isomorfos, cosa que no hemos hecho aquí. Sin embargo, la prueba de esto es sencilla: basta con quitar los casos de reducción por la regla (Beta) en los lemas 4.36 y 4.39. Hacemos esta aclaración únicamente aquí, aunque también aplica para los cálculos que presentaremos en los capítulos 5 y 6.

Capítulo 5

Cálculo $\lambda\mathbf{re_{gc}}$

5.1 El cálculo de sustituciones explícitas $\lambda\mathbf{re_{gc}}$

5.1.1 Introducción

En la presente sección introducimos un cálculo de sustituciones explícitas derivado del cálculo λre: $\lambda\text{re}_{\text{gc}}$. Dicho cálculo surge a partir del agregado de una regla de *Garbage Collection* a λre, al estilo de λxgc [BR95], $\lambda\sigma_{\text{gc}}$ [Zil09] y λex [Kes08, Kes09], entre otros. La inclusión de esta regla responde a un intento de acercamiento a un cálculo con índices de *de Bruijn* que posea *todas* las buenas propiedades, como bien podría ser el caso de un cálculo isomorfo a λex. Hacemos notar, sin embargo, que no agregamos composición aquí. Esto último radica en ver si, con el agregado de *Garbage Collection*, llegamos a un cálculo cercano a λex, pero sin composición. Éste es el caso de, por ejemplo, λxgc [BR95], presentado en 1.7.3. Es importante volver a destacar que, si bien la regla de *Garbage Collection* no hace a la obtención de propiedades adicionales en λxgc, sí lo hace en λex. Puntualmente, juega un rol clave en la obtención de metaconfluencia y PSN en simultáneo (referimos al lector a [Kes09] para más detalles).

Antes de presentar $\lambda\text{re}_{\text{gc}}$, queremos destacar la introducción de un nuevo metaoperador en este cálculo. Dicho operador, que denominaremos de *decremento de índices*, sirve para la implementación de la regla de *Garbage Collection*. Como se describió en la sección 1.7.3, la idea de *Garbage Collection* es que el cálculo mismo se dé cuenta cuándo una sustitución aplicada a un término es vacua. En el caso particular de λre, esto pasa cuando, en $a[b]$, $1 \notin \text{FV}(a)$. Bajo λre, cuando esto mismo ocurre, la sustitución se distribuye dentro del término a. Al llegar a los índices, dado que son todos de la forma $n+1$ ($n \in \mathbb{N}_{>0}$), se los decrementa en uno mediante la regla (VarR) (*i.e.,* reducen a n). Luego, de manera análoga a λxgc, la idea para *Garbage Collection* es: si la variable 1 no está libre en a, el resultado de la sustitución es decrementar todas las variables libres de a en 1. Con este propósito, es que introducimos dentro del cálculo el nuevo operador, que definimos en la siguiente sección, donde además hacemos lo propio para $\lambda\text{re}_{\text{gc}}$.

5.1.2 Términos de $\lambda\text{re}_{\text{gc}}$ y *decremento de índices*

El cálculo que estamos presentando, $\lambda\text{re}_{\text{gc}}$, es una extensión del presentado en la anterior sección: λre. Dicha extensión, consiste en el agregado de una regla (*Garbage Collection*) al cálculo. Por este motivo, aclaramos que los términos sobre los que opera $\lambda\text{re}_{\text{gc}}$, son los mismos sobre los que trabaja λre, descriptos en la definición 4.8. Asimismo, las definiciones de variables libres, tamaño de términos, metaoperadores, etc., aplican de idéntica manera aquí. Luego, donde no haya diferencia entre las definiciones para ambos formalismos (*i.e.,* cuando las definiciones sean *iguales*), utilizaremos los nombres dados para λre. Por ejemplo, denotaremos al conjunto de términos de $\lambda\text{re}_{\text{gc}}$ como Λre.

Como mencionamos en la anterior sección, definimos a continuación el operador de *decremento de índices*, necesario para la implementación de la regla de *Garbage Collection* de $\lambda\text{re}_{\text{gc}}$. Es importante mencionar que definimos primero el metaoperador sobre los términos sobre los que opera λr (*i.e.,* Λ_{dB}), para luego extenderlo sobre los términos del cálculo que estamos presentando, previa prueba de propiedades sobre el cálculo con sustituciones implícitas, a modo de verificar la consistencia de lo que definimos.

Nota. *La definición del operador de decremento de índices, junto con algunas de sus propiedades, está inspirada en [RÓ93], y puede encontrarse también en [VARK07].*

Definición 5.1 (Metaoperador $\downarrow_i$)**.** Para todo $i \geq 1$, $\downarrow_i : \Lambda_{\text{dB}} \to \Lambda_{\text{dB}}$ se define inductivamente como:

$$\begin{array}{rcl} \downarrow_i(n) & = & \begin{cases} n & \text{si } n < i \\ indefinido & \text{si } n = i \\ n-1 & \text{si } n > i \end{cases} \\ \downarrow_i(a\,b) & = & \downarrow_i(a)\ \downarrow_i(b) \\ \downarrow_i(\lambda a) & = & \lambda \downarrow_{i+1}(a) \end{array}$$

Nota. *Notar que, $\forall a \in \Lambda_{dB} : \downarrow_i(a)$ está bien definido $\iff$ $i \notin FV(a)$. Intuitivamente, en el caso en que $i \in FV(a)$, lo que ocurre es que podemos estar provocando una captura en caso de realizar el decremento de índice, o bien estar generando un índice inválido (como ser el "índice" 0 para el caso particular de $\downarrow_1(a)$, si $1 \in FV(a)$).*

5.1.3 Algunos lemas sobre decremento de índices para λr

Los siguientes cuatro lemas permiten conocer la forma en que un decremento de índices puede distribuirse dentro de una sustitución en el cálculo λr. En particular, el último de los lemas presentados a continuación (lema 5.6), nos sirve para extender la definición de decremento de índices a los términos del cálculo $\lambda\text{re}_{\text{gc}}$.

Lema 5.2. Para todo $a \in \Lambda_{\text{dB}}$, $i, j \in \mathbb{N}_{>0}$, si $j \geq 2 \wedge i + j \notin \text{FV}(a)$, entonces $i + j \notin \text{FV}(\updownarrow_i(a))$.

Demostración. Por inducción en a.

- $a = n \in \mathbb{N}_{>0}$. Sabemos que $n \neq i + j$ por hipótesis. Entonces,

$$\mathrm{FV}(\uparrow_i(n)) = \left\{ \begin{array}{lcll} \mathrm{FV}(n) & = & \{n\} & \text{si } n < i \vee n > i+1 \\ \mathrm{FV}(i+1) & = & \{i+1\} & \text{si } n = i \\ \mathrm{FV}(i) & = & \{i\} & \text{si } n = i+1 \end{array} \right\} \not\ni i + j$$

- $a = b\,c$. Tenemos, por definición de FV, que $i + j \notin \mathrm{FV}(b) \cup \mathrm{FV}(c)$. Entonces,

$$\mathrm{FV}(\uparrow_i(b\,c)) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_i(b)\ \uparrow_i(c)) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_i(b)) \cup \mathrm{FV}(\uparrow_i(c))$$

  Por hipótesis inductiva, $i + j \notin \mathrm{FV}(\uparrow_i(b)) \wedge i + j \notin \mathrm{FV}(\uparrow_i(c))$. Luego, tampoco está en la unión.

- $a = \lambda b$. Por definición de FV, $i + j + 1 \notin \mathrm{FV}(b)$. Por hipótesis inductiva, $i + j + 1 \notin \mathrm{FV}(\uparrow_{i+1}(b))$. Luego,

$$\mathrm{FV}(\uparrow_i(\lambda b)) \underset{\text{def}}{=} \mathrm{FV}(\lambda\ \uparrow_{i+1}(b)) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_{i+1}(b)) - 1$$

  Ahora, $i + j \notin \mathrm{FV}(\uparrow_{i+1}(b)) - 1 \iff i + j + 1 \notin \mathrm{FV}(\uparrow_{i+1}(b))$

□

Observación 5.3. Para todo $a \in \Lambda_{\mathrm{dB}}$, $i \in \mathbb{N}_{>0}$, si $i \notin \mathrm{FV}(a)$, entonces $i + 1 \notin \mathrm{FV}(\uparrow_0(a))$. Esta observación puede verificarse con una sencilla inducción sobre a, y puede verse intuitivamente por el contrarrecíproco.

Lema 5.4. Para todo $a \in \Lambda_{\mathrm{dB}}$, $i \in \mathbb{N}_{>0}$, $j \in \mathbb{N}$, si $i \geq j + 2 \wedge i - 1 \notin \mathrm{FV}(a)$, entonces $\downarrow_i(\uparrow_j(a)) = \uparrow_j(\downarrow_{i-1}(a))$. Notar que, por hipótesis, $\downarrow_{i-1}(a)$ está bien definido.

Demostración. Por inducción en a.

- $a = n \in \mathbb{N}_{>0}$. Tenemos, por hipótesis, que $n \neq i - 1 \iff n + 1 \neq i$. Dividimos en dos casos:

  1. $n > j$. Tenemos, entonces, que $\downarrow_i(\uparrow_j(n)) \underset{\text{def}}{=} \downarrow_i(n+1)$, que está bien definido por hipótesis. Luego,

$$\downarrow_i(n+1) = \left\{ \begin{array}{ll} n+1 & \text{si } n+1 < i \iff n < i-1 \\ n & \text{si } n+1 > i \iff n > i-1 \end{array} \right.$$

     Además,

$$\uparrow_j(\downarrow_{i-1}(n)) = \left\{ \begin{array}{lll} \uparrow_j(n) & \underset{n\ >\ j}{=} & n+1 \quad \text{si } n < i-1 \\ \uparrow_j(n-1) & \underset{n\ -\ 1\ >\ i\ -\ 2\ \geq\ j}{=} & n \quad \text{si } n > i-1 \end{array} \right.$$

  2. $n \leq j$. Luego, por hipótesis, $i \geq j + 2 \implies i > j \geq n$. Entonces,

$$\downarrow_i(\uparrow_j(n)) \underset{n\ \leq\ j}{=} \downarrow_i(n) \underset{i\ >\ n}{=} n$$

     Además, por hipótesis, $i \geq j + 2 \iff i - 1 \geq j + 1 > n$. Luego,

$$\uparrow_j(\downarrow_{i-1}(n)) \underset{i\ -\ 1\ >\ n}{=} \uparrow_j(n) \underset{n\ \leq\ j}{=} n$$

     Notar que tanto $\downarrow_i(n)$ como $\downarrow_{i-1}(n)$ están bien definidos por hipótesis.

- $a = b\,c$. Luego,

$$\downarrow_i(\uparrow_j(b\,c)) \underset{\text{def}}{=} \downarrow_i(\uparrow_j(b)\ \uparrow_j(c)) \underset{\text{def}}{=} \downarrow_i(\uparrow_j(b))\ \downarrow_i(\uparrow_j(c)) \underset{\text{HI}}{=}$$
$$\uparrow_j(\downarrow_{i-1}(b))\ \uparrow_j(\downarrow_{i-1}(c)) \underset{\text{def}}{=} \uparrow_j(\downarrow_{i-1}(b)\ \downarrow_{i-1}(c)) \underset{\text{def}}{=}$$
$$\uparrow_j(\downarrow_{i-1}(b\,c))$$

- $a = \lambda b$. Luego,

$$\downarrow_i(\uparrow_j(\lambda b)) \underset{\text{def}}{=} \downarrow_i(\lambda\ \uparrow_{j+1}(b)) \underset{\text{def}}{=} \lambda\ \downarrow_{i+1}(\uparrow_{j+1}(b)) \underset{\text{HI}}{=}$$
$$\lambda\ \uparrow_{j+1}(\downarrow_i(b)) \underset{\text{def}}{=} \uparrow_j(\lambda\ \downarrow_i(b)) \underset{\text{def}}{=}$$
$$\uparrow_j(\downarrow_{i-1}(\lambda b))$$

□

Lema 5.5. Para todo $a \in \Lambda_{\text{dB}},\ i, j \in \mathbb{N}_{>0}$, si $j \geq 2 \wedge i + j \notin \text{FV}(a)$, entonces $\downarrow_{i+j}(\updownarrow_i(a)) = \updownarrow_i(\downarrow_{i+j}(a))$. Notar que, por hipótesis, $\downarrow_{i+j}(a)$ está bien definido.

Demostración. Por inducción en a.

- $a = n \in \mathbb{N}_{>0}$. Tenemos, por hipótesis, que $n \neq i + j$. Dividimos en dos casos:

  1. $n < i + j$. Luego, $\updownarrow_i(\downarrow_{i+j}(n)) = \updownarrow_i(n)$. Entonces,

$$\updownarrow_i(n) = \begin{cases} n & \text{si } n < i \vee n > i + 1 \\ i + 1 & \text{si } n = i \\ i & \text{si } n = i + 1 \end{cases}$$

  Ahora,

$$\downarrow_{i+j}(\updownarrow_i(n)) = \begin{cases} \downarrow_{i+j}(n) & \underset{n\ <\ i\ +\ j}{=} & n & \text{si } n < i \vee n > i + 1 \\ \downarrow_{i+j}(i+1) & \underset{j\ \geq\ 2}{=} & i + 1 & \text{si } n = i \\ \downarrow_{i+j}(i) & \underset{j\ \geq\ 2}{=} & i & \text{si } n = i + 1 \end{cases}$$

  2. $n > i + j$. Como $j \geq 2$, tenemos que $n - 1 > i + j - 1 \geq i + 1$. Luego,

$$\updownarrow_i(\downarrow_{i+j}(n)) \underset{\text{def}}{=} \updownarrow_i(n-1) \underset{n\ -\ 1\ >\ i\ +\ 1}{=} n - 1$$

  Por otro lado,

$$\downarrow_{i+j}(\updownarrow_i(n)) \underset{n\ -\ 1\ >\ i\ +\ 1}{=} \downarrow_{i+j}(n) \underset{\text{def}}{=} n - 1$$

- $a = b\,c$. Luego,

$$\downarrow_{i+j}(\updownarrow_i(b\,c)) \underset{\text{def}}{=} \downarrow_{i+j}(\updownarrow_i(b)\ \updownarrow_i(c)) \underset{\text{def}}{=} \downarrow_{i+j}(\updownarrow_i(b))\ \downarrow_{i+j}(\updownarrow_i(c)) \underset{\text{HI}}{=}$$
$$\updownarrow_i(\downarrow_{i+j}(b))\ \updownarrow_i(\downarrow_{i+j}(c)) \underset{\text{def}}{=} \updownarrow_i(\downarrow_{i+j}(b)\ \downarrow_{i+j}(c)) \underset{\text{def}}{=} \updownarrow_i(\downarrow_{i+j}(b\,c))$$

- $a = \lambda b$. Luego,

$$\downarrow_{i+j}(\updownarrow_i(\lambda b)) \underset{\text{def}}{=} \downarrow_{i+j}(\lambda\ \updownarrow_{i+1}(b)) \underset{\text{def}}{=} \lambda\ \downarrow_{i+j+1}(\updownarrow_{i+1}(b)) \underset{\text{HI}}{=}$$
$$\lambda\ \updownarrow_{i+1}(\downarrow_{i+j+1}(b)) \underset{\text{def}}{=} \updownarrow_i(\lambda\ \downarrow_{i+j+1}(b)) \underset{\text{def}}{=} \updownarrow_i(\downarrow_{i+j}(\lambda b))$$

□

El siguiente lema nos permite conocer de qué manera se distribuye un decremento de índices dentro de una sustitución, y es el que utilizaremos como argumento para la extensión de decremento de índices para $\lambda \text{re}_{\text{gc}}$.

Lema 5.6 (Interacción entre metasustituciónes y $\downarrow_i$ en λr)**.** Para todo $a, b \in \Lambda_{\text{dB}},\ i \in \mathbb{N}_{>0}$, si $i+1 \notin \text{FV}(a) \wedge i \notin \text{FV}(b)$, entonces, $\downarrow_i(a\{b\}) = \downarrow_{i+1}(a)\{\downarrow_i(b)\}$. Notar que $\downarrow_{i+1}(a)$ y $\downarrow_i(b)$ están bien definidos por hipótesis.

Demostración. Por inducción en a.

- $a = n \in \mathbb{N}_{>0}$. Por hipótesis, $n \neq i+1$. Dividimos en dos casos:
  1. $n = 1$. Tenemos que $\downarrow_i(1\{b\}) \underset{\text{def}}{=} \downarrow_i(b)$. Luego,
  $$\downarrow_{i+1}(1)\{\downarrow_i(b)\} \underset{1\ <\ i\ +\ 1}{=} 1\{\downarrow_i(b)\} \underset{\text{def}}{=} \downarrow_i(b)$$
  2. $n > 1$. Luego, $\downarrow_i(n\{b\}) \underset{\text{def}}{=} \downarrow_i(n-1)$.
  Notar que $i \neq n-1 \iff i+1 \neq n$, con lo que no hay indefinición. Entonces,
  $$\downarrow_i(n-1) = \begin{cases} n-1 & \text{si } n-1 < i \iff n < i+1 \\ n-2 & \text{si } n-1 > i \iff n > i+1 \end{cases}$$
  Además,
  $$\downarrow_{i+1}(n)\{\downarrow_i(b)\} = \begin{cases} n\{\downarrow_i(b)\} & \underset{n\ >\ 1}{=} & n-1 & \text{si } n < i+1 \\ n-1\{\downarrow_i(b)\} & \underset{n\ -\ 1\ >\ i\ \geq\ 1}{=} & n-2 & \text{si } n > i+1 \end{cases}$$

- $a = c\,d$. Luego,

$$\downarrow_i((c\,d)\{b\}) \underset{\text{def}}{=} \downarrow_i(c\{b\}\ d\{b\}) \underset{\text{def}}{=} \downarrow_i(c\{b\})\ \downarrow_i(d\{b\}) \underset{\text{HI}}{=}$$
$$\downarrow_{i+1}(c)\{\downarrow_i(b)\}\ \downarrow_{i+1}(d)\{\downarrow_i(b)\} \underset{\text{def}}{=} (\downarrow_{i+1}(c)\ \downarrow_{i+1}(d))\{\downarrow_i(b)\} \underset{\text{def}}{=}$$
$$\downarrow_{i+1}(c\,d)\{\downarrow_i(b)\}$$

- $a = \lambda c$. Luego,

$$\downarrow_i((\lambda c)\{b\}) \underset{\text{def}}{=} \downarrow_i(\lambda\ \updownarrow_1(c)\{\uparrow_0(b)\}) \underset{\text{def}}{=} \lambda\ \downarrow_{i+1}(\updownarrow_1(c)\{\uparrow_0(b)\}) \underset{\text{HI}}{=}$$

Observar que, por lema 5.2, $i+2 \notin \text{FV}(c) \implies i+2 \notin \text{FV}(\updownarrow_i(c))$ y que, por observación 5.3, $i \notin \text{FV}(b) \implies i+1 \notin \text{FV}(\uparrow_0(b))$. Además, por lema 4.2, $|\ \updownarrow_1(c)\ | = |c|$. Luego, es posible aplicar hipótesis inductiva. Tenemos, en consecuencia, que:

$$\underset{\text{HI}}{=} \lambda\ \downarrow_{i+2}(\updownarrow_1(c))\{\downarrow_{i+1}(\uparrow_0(b))\} \underset{\text{L.5.4}}{=}$$
$$\lambda\ \downarrow_{i+2}(\updownarrow_1(c))\{\uparrow_0(\downarrow_i(b))\} \underset{\text{L.5.5}}{=} \lambda\ \updownarrow_1(\downarrow_{i+2}(c))\{\uparrow_0(\downarrow_i(b))\} \underset{\text{def}}{=}$$
$$(\lambda\ \downarrow_{i+2}(c))\{\downarrow_i(b)\} \underset{\text{def}}{=} \downarrow_{i+1}(\lambda c)\{\downarrow_i(b)\}$$

□

5.1.4 Más definiciones

Como mencionamos anteriormente, basándonos en el lema 5.6, podemos extender la definición de decrementos de índice a los términos de $\lambda\text{re}_{\text{gc}}$ (Λre). Esto lo hacemos a continuación.

Definición 5.7 (Metaoperador $\downarrow_i$)**.** Para todo $i \geq 1$, $\downarrow_i : \Lambda\text{re} \to \Lambda\text{re}$ se define inductivamente como:

$$\begin{array}{rcl} \downarrow_i(n) & = & \begin{cases} n & \text{si } n < i \\ \text{indefinido} & \text{si } n = i \\ n-1 & \text{si } n > i \end{cases} \\ \downarrow_i(a\,b) & = & \downarrow_i(a)\ \downarrow_i(b) \\ \downarrow_i(\lambda a) & = & \lambda \downarrow_{i+1}(a) \\ \downarrow_i(a[b]) & = & \downarrow_{i+1}(a)[\downarrow_i(b)] \end{array}$$

Nota. *Cabe destacar que, al igual que para el decremento de índices para λr, $\downarrow_i(a)$ está bien definido si y sólo si $i \notin FV(a)$.*

Presentado ya el metaoperador de decremento de índices, pasamos a introducir las reglas del cálculo $\lambda\text{re}_{\text{gc}}$. Dichas reglas pueden verse en la figura 5.1.

(Beta)	$(\lambda a)\,b$	$\to$	$a[b]$	
(App)	$(a\,b)[c]$	$\to$	$a[c]\ b[c]$	
(Lamb)	$(\lambda a)[c]$	$\to$	$\lambda \Uparrow_1(a)[\uparrow_0(c)]$	
(Var)	$1[c]$	$\to$	c	
(VarR)	$(n+1)[c]$	$\to$	n	$(n \in \mathbb{N}_{>0})$
(GC)	$a[c]$	$\to$	$\downarrow_1(a)$	$(1 \notin \text{FV}(a))$

Figura 5.1: Reglas del cálculo $\lambda\text{re}_{\text{gc}}$

Nota. *La regla (GC) está bien definida, pues $\downarrow_1(a)$ está definido si y sólo si $1 \notin FV(a)$.*

Al igual que para λre, llamamos re_{gc} al conjunto de reglas (Var), (VarR), (App), (Lamb) y (GC). Es decir, el conjunto de reglas re_{gc} está conformado por aquellas que se encargan de distribuir y evaluar las sustituciones explícitas. Denominamos $\lambda\text{re}_{\text{gc}}$ al conjunto de reglas (Beta) + re_{gc}.

Es interesante destacar que, de manera análoga a los cálculos λx y λxgc, la diferencia entre λre y $\lambda\text{re}_{\text{gc}}$ es su regla de *Garbage Collection*. Es decir, hacemos ver al lector que $\text{re}_{\text{gc}} = \text{re} + (GC)$, y $\lambda\text{re}_{\text{gc}} = \lambda\text{re} + (GC)$.

Hacemos una pausa en el desarrollo para discutir la inclusión de la regla (VarR) en este cálculo. Dicha regla, como puede observarse en la figura 5.1, dice:

$$(n+1)[c] \to n \qquad n \in \mathbb{N}_{>0}$$

Ahora bien: en caso de no incluir dicha regla, ¿sería posible simuarla mediante la regla (GC)? La respuesta es: sí. Imaginemos que, dado un término $a[b] \in \Lambda\text{re}$, a tiene alguna variable libre mayor a uno. Es decir, $(n+1) \in \text{FV}(a)$, con $n \in \mathbb{N}_{>0}$. De igual manera que para el cálculo λre, podríamos aquí distribuir la sustitución $[b]$ dentro del término a, utilizando las reglas (App) y (Lamb). En algún momento, llegaremos a un término de la forma $(n+1)[b]$, con $n \in \mathbb{N}_{>0}$. Notemos que, dado el caso, y ante la falta de la regla (VarR), podemos utilizar la regla (GC), que en la mencionada situación se comportaría de igual manera. Es decir,

$$(n+1)[b] \underset{\text{(GC), } 1 \notin \text{FV}(n+1)}{\to_{\lambda\text{re}_{\text{gc}}}} \downarrow_1(n+1) \underset{n+1>1}{=} n$$

Dicho esto, mantenemos la regla (VarR) en el cálculo, sólo a fines de mostrar que éste es isomorfo a λxgc, cálculo que sí incluye la regla equivalente [BR95, p.3]. En la siguiente sección, mostraremos dicho isomorfismo.

Pasamos a definir formalmente el cálculo $\lambda\text{re}_{\text{gc}}$.

Definición 5.8 (re_{gc}-reducción)**.** Se define la relación $\text{re}_{\text{gc}} \subseteq \Lambda\text{re} \times \Lambda\text{re}$, notada $\to_{\text{re}_{\text{gc}}}$ como la clausura contextual de las reglas re_{gc}.

Definición 5.9 ($\lambda\text{re}_{\text{gc}}$-reducción)**.** Se define la relación $\lambda\text{re}_{\text{gc}} \subseteq \Lambda\text{re} \times \Lambda\text{re}$, notada $\to_{\lambda\text{re}_{\text{gc}}}$ como la clausura contextual de las reglas $\lambda\text{re}_{\text{gc}}$.

Definición 5.10 (Cálculo $\lambda\text{re}_{\text{gc}}$)**.** El cálculo $\lambda\text{re}_{\text{gc}}$ es el sistema de reducción $(\Lambda\text{re}, \lambda\text{re}_{\text{gc}})$

5.1.5 No agregado de variables libres por reducción

Al igual que para el cálculo λre, mostramos a continuación un lema que nos garantiza el no agregado de variables libres bajo la relación $\lambda\text{re}_{\text{gc}}$.

Observación 5.11. Como puede verse fácilmente (con una inducción, por ejemplo),

$$\forall a \in \Lambda\text{re} : 1 \notin \text{FV}(a) \implies \text{FV}(\downarrow_1(a)) = \text{FV}(a) - 1$$

Esto ocurre, intuitivamente, porque la operación $\downarrow_1(a)$ está restando uno a las variables libres de a. Además, notar que la operación de decremento de índices está bien definida por hipótesis. Esto es, $1 \notin \text{FV}(a)$.

Lema 5.12 (No agregado de variables libres por reducción)**.** Para todo $a, b \in \Lambda\text{re}$, tenemos que:

$$a \to_{\lambda\text{re}_{\text{gc}}} b \implies \text{FV}(b) \subseteq \text{FV}(a)$$

Demostración. Por inducción en a.

- $a = n \in \mathbb{N}_{>0}$. No hay reducción posible. Luego, vale vacuamente.
- $a = c\,d$. Análogo al lema 4.17, cambiando λre por $\lambda\text{re}_{\text{gc}}$.
- $a = \lambda c$. Análogo al lema 4.17, cambiando λre por $\lambda\text{re}_{\text{gc}}$.
- $a = c[d]$. Si la reducción es interna (es decir, $e \to_{\lambda\text{re}_{\text{gc}}} e'$, $e \in \{c, d\}$), procedemos de manera similar al caso de aplicación del lema 4.17, cuando la reducción es interna (1), y cambiando λre por $\lambda\text{re}_{\text{gc}}$. Si la reducción es en la raíz, tenemos dos posibilidades.

1. La reducción es por la regla (App), (Var), (VarR) o (Lamb). Cada uno de estos casos es análogo a su contraparte en el lema 4.17, cambiando λre por λre$_{\mathrm{gc}}$.
2. La reducción es por la regla (GC). Es decir, tenemos que $1 \notin \mathrm{FV}(c)$ y, por lo tanto,
$$c[d] \underset{(\mathrm{GC})}{\to_{\lambda\mathrm{re_{gc}}}} \downarrow_1(c) = b$$
Luego,
$$\mathrm{FV}(\downarrow_1(c)) \underset{\text{O.5.11}}{=} \mathrm{FV}(c) - 1 \subseteq (\mathrm{FV}(c) - 1) \cup \mathrm{FV}(d) \underset{\text{def}}{=} \mathrm{FV}(c[d])$$

□

5.2 Isomorfismo entre λxgc y λre$_{\mathbf{gc}}$

5.2.1 Introducción

El objetivo de esta sección es mostrar que los cálculos λxgc y λre$_{\mathrm{gc}}$ son isomorfos. Al igual que para λx, hasta ahora no conocíamos ningún cálculo con índices isomorfo a λxgc, y la posesión de dicho isomorfismo (tal y como explicamos en las secciones 1.3.4 y 4.2), hace que los cálculos tengan las mismas propiedades. Esto nos ahorra, tal y como sucedió para λre, varias pruebas difíciles sobre λre$_{\mathrm{gc}}$.

5.2.2 Traducciones entre Λx y Λre

Como puede observarse en la presentación del cálculo λxgc [BR95], y al igual que para lo que ocurre entre λre y λre$_{\mathrm{gc}}$, los términos de λx y λxgc son los mismos. Por ende, utilizamos las traducciones ya dadas en la sección 4.2.2 para realizar las demostraciones correspondientes. Como ayuda para la memoria, remitimos al lector a las definiciones de $\mathrm{w}_{\bar{x}_{1:n}}$ (4.18), de $\mathrm{u}_{\bar{x}_{1:n}}$ (4.24), de w (4.22) y de u (4.28) antes de comenzar la lectura de las pruebas presentadas aquí.

5.2.3 Pruebas del isomorfismo

Recordemos de la sección 4.2.3 que, para mostrar el isomorfismo entre los cálculos, tenemos que dar pruebas para los siguientes tres puntos:

1. $\mathrm{w} \circ \mathrm{u} = \mathrm{Id}_{\Lambda\mathrm{re}} \wedge \mathrm{u} \circ \mathrm{w} = \mathrm{Id}_{\Lambda\mathrm{x}}$
2. $\forall t, u \in \Lambda\mathrm{x} : t \to_{\lambda\mathrm{xgc}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{re_{gc}}} \mathrm{w}(u)$
3. $\forall a, b \in \Lambda\mathrm{re} : a \to_{\lambda\mathrm{re_{gc}}} b \implies \mathrm{u}(a) \to_{\lambda\mathrm{xgc}} \mathrm{u}(b)$

Dado que las traducciones que utilizamos aquí son las mismas que utilizamos para el isomorfismo entre λre y λx, no es necesario volver demostrar que la composición de las traducciones da como resultado la función identidad. Remitimos al lector a la sección 4.2.3 para más detalles sobre las pruebas.

Pasamos, dicho esto último, a las pruebas de preservación de reducción por traducción.

Un paso de reducción en λxgc implica un paso de reducción en λre$_{gc}$

Además de los lemas presentados en la sección 4.2.3, antes de probar la mentada preservación de la reducción, necesitamos dos lemas más, que presentamos ahora.

El primero de los dos lemas muestra la preservación de no pertenencia de variables libres con la traducción w (*i.e.,* si x no está libre en t, entonces el índice que le corresponde a x en el término traducido tampoco está libre en la traducción de t).

Lema 5.13. Sea $t \in \Lambda x : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Sea $m \in \mathbb{N} : 1 \leq m \leq n+1$. Luego, para todo $x \notin \{x_1, \ldots, x_n\}$, tenemos que $m \notin \mathrm{FV}\big(\mathrm{w}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(t)\big)$.

Demostración. Por inducción en t. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}$. Entonces, tenemos que ver que para todo $x \notin \{x_1, \ldots, x_n\}$, $m \notin \mathrm{FV}(\mathrm{w}_{\bar{z}}(t))$.

- $t = y \in \{x_1, \ldots, x_n\}$. Por hipótesis, tenemos que $y \neq x$. Luego, $\mathrm{w}_{\bar{z}}(y) \underset{\text{def}}{=} min\,\{j : z_j = y\} = k \underset{\text{hip}}{\neq} m$. Entonces, $\mathrm{FV}(\mathrm{w}_{\bar{z}}(y)) = \{k\} \not\ni m$.

- $t = u\,v$. Luego,

  $$\mathrm{FV}(\mathrm{w}_{\bar{z}}(u\,v)) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{w}_{\bar{z}}(u)\ \mathrm{w}_{\bar{z}}(v)) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{w}_{\bar{z}}(u)) \cup \mathrm{FV}(\mathrm{w}_{\bar{z}}(v))$$

  Por hipótesis inductiva, $m \notin \mathrm{FV}(\mathrm{w}_{\bar{z}}(M))$, con $M \in \{u, v\}$. Luego, tampoco está en la unión.

- $t = \lambda y.u$. Por convención de variables, podemos asumir $y \neq x$. Luego,

  $$\mathrm{FV}(\mathrm{w}_{\bar{z}}(\lambda y.u)) \underset{\text{def}}{=} \mathrm{FV}(\lambda \mathrm{w}_{y:\bar{z}}(u)) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{w}_{y:\bar{z}}(u)) - 1$$

  Por hipótesis inductiva, $(m+1) \notin \mathrm{FV}(\mathrm{w}_{y:\bar{z}}(u))$.
  Luego, $m \notin \mathrm{FV}(\mathrm{w}_{y:\bar{z}}(u)) - 1$.

- $t = u[y := v]$. Al igual que en el caso anterior, podemos asumir $y \neq x$. Luego,

  $$\mathrm{FV}(\mathrm{w}_{\bar{z}}(u[y := v])) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{w}_{y:\bar{z}}(u)[\mathrm{w}_{\bar{z}}(v)]) \underset{\text{def}}{=}$$

  $$(\mathrm{FV}(\mathrm{w}_{y:\bar{z}}(u)) - 1) \cup \mathrm{FV}(\mathrm{w}_{\bar{z}}(v))$$

  Por hipótesis inductiva, $(m+1) \notin \mathrm{FV}(\mathrm{w}_{y:\bar{z}}(u)) \wedge m \notin \mathrm{FV}(\mathrm{w}_{\bar{z}}(v))$.
  Luego, $m \notin \mathrm{FV}(\mathrm{w}_{y:\bar{z}}(u)) - 1$ y, por ende, tampoco está en $(\mathrm{FV}(\mathrm{w}_{y:\bar{z}}(u)) - 1) \cup \mathrm{FV}(\mathrm{w}_{\bar{z}}(v))$

□

Observación 5.14. De manera análoga al lema 5.13, podemos probar que para todo $t \in \Lambda x : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, para todo $m \in \mathbb{N} : 1 \leq m \leq n+1$, y para todo $x \notin \{x_1, \ldots, x_{m-1}\} \wedge x \in \mathrm{FV}(t)$, se cumple que $m \in \mathrm{FV}\big(\mathrm{w}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(t)\big)$.

El siguiente y último de los lemas, muestra la relación entre los decrementos de índice y la traducción w.

Lema 5.15 (Interacción entre la traducción w y $\downarrow_i$)**.** Sea $t \in \Lambda\mathrm{x} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Sea $m \in \mathbb{N} : 1 \leq m \leq n+1$. Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(t) = \downarrow_m(\mathrm{w}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(t))$$

Demostración. Por inducción en t. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}$. Entonces, tenemos que ver que:

$$\mathrm{w}_{\bar{x}_{1:n}}(t) = \downarrow_m(\mathrm{w}_{\bar{z}}(t))$$

- $t = y \in \{x_1, \ldots, x_n\}$. Por hipótesis, $y \neq x$. Hay dos casos:
  1. $y \in \{x_1, \ldots, x_{m-1}\}$. Entonces, $\mathrm{w}_{\bar{x}_{1:n}}(y) \underset{\text{def}}{=} k < m$. Luego,
  $$\downarrow_m(\mathrm{w}_{\bar{z}}(y)) \underset{\text{hip}}{=} \downarrow_m(k) \underset{\text{def, } k < m}{=} k$$
  2. $y \notin \{x_1, \ldots, x_{m-1}\}$. Por lo tanto, $y \in \{x_m, \ldots, x_n\}$. Entonces, $\mathrm{w}_{\bar{x}_{1:n}}(y) \underset{\text{def}}{=} k \geq m$. Luego,
  $$\downarrow_m(\mathrm{w}_{\bar{z}}(y)) \underset{\text{hip, } x \neq y}{=} \downarrow_m(k+1) \underset{\text{def, } k+1 > m}{=} k$$
- $t = u\, v$. Luego,
$$\downarrow_m(\mathrm{w}_{\bar{z}}(u\, v)) \underset{\text{def}}{=} \downarrow_m(\mathrm{w}_{\bar{z}}(u)\ \mathrm{w}_{\bar{z}}(v)) \underset{\text{def}}{=} \downarrow_m(\mathrm{w}_{\bar{z}}(u))\ \downarrow_m(\mathrm{w}_{\bar{z}}(v)) \underset{\text{HI}}{=}$$
$$\mathrm{w}_{\bar{x}_{1:n}}(u)\ \mathrm{w}_{\bar{x}_{1:n}}(v) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(u\, v)$$
- $t = \lambda y.u$. Por convención de variables, podemos asumir $y \neq x$. Luego,
$$\downarrow_m(\mathrm{w}_{\bar{z}}(\lambda y.u)) \underset{\text{def}}{=} \downarrow_m(\lambda \mathrm{w}_{y:\bar{z}}(u)) \underset{\text{def}}{=} \lambda\, \downarrow_{m+1}(\mathrm{w}_{y:\bar{z}}(u)) \underset{\text{HI}}{=}$$
$$\lambda \mathrm{w}_{y:\bar{x}_{1:n}}(u) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(\lambda y.u)$$
- $t = u[y := v]$. Al igual que en el caso anterior, podemos asumir $y \neq x$. Luego,
$$\downarrow_m(\mathrm{w}_{\bar{z}}(u[y := v])) \underset{\text{def}}{=} \downarrow_m(\mathrm{w}_{y:\bar{z}}(u)[\mathrm{w}_{\bar{z}}(v)]) \underset{\text{def}}{=}$$
$$\downarrow_{m+1}(\mathrm{w}_{y:\bar{z}}(u))[\downarrow_m(\mathrm{w}_{\bar{z}}(v))] \underset{\text{HI}}{=} \mathrm{w}_{y:\bar{x}_{1:n}}(u)[\mathrm{w}_{\bar{x}_{1:n}}(v)] \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(u[y := v])$$

□

Dados los dos únicos lemas adicionales necesarios, pasamos a mostrar la preservación de la reducción λxgc bajo la traducción w. Cabe destacar que el único caso interesante es el de la reducción por *Garbage Collection* (GC), dado que el resto de los casos son análogos a los tratados en el lema 4.36.

Lema 5.16 (Preservación de la reducción λxgc bajo w)**.** Para todo $t, u \in \Lambda\mathrm{x}$, tenemos que:

$$t \to_{\lambda\mathrm{xgc}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{re}_{\mathrm{gc}}} \mathrm{w}(u)$$

Demostración. Por inducción en t. Sin pérdida de generalidad, supongamos que $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Como $t \to_{\lambda\mathrm{xgc}} u$, tenemos que $\mathrm{FV}(u) \subseteq \mathrm{FV}(t)$. Sean $x_1, \ldots, x_n$ las primeras n variables de la enumeración de $\mathbb{V}$ elegida para w. Luego, tenemos que $\mathrm{w}(t) = \mathrm{w}_{\bar{x}_{1:n}}(t)$ y que $\mathrm{w}(u) = \mathrm{w}_{\bar{x}_{1:n}}(u)$.

- $t = x \in \{x_1, \ldots, x_n\}$. No hay reducción posible. Luego, vale vacuamente.
- $t = t_1\, t_2$. Análogo a su contraparte en el lema 4.36, cambiando λx por λxgc y λre por λre$_{\mathrm{gc}}$.
- $t = \lambda x.t_1$. Análogo a su contraparte en el lema 4.36, cambiando λx por λxgc y λre por λre$_{\mathrm{gc}}$.
- $t = t_1[x := t_2]$. Si la reducción es interna (es decir, $t_i \to_{\lambda\mathrm{x}} t_i',\ i \in \{1, 2\}$), procedemos de manera similar al caso de aplicación del lema 4.36, cuando la reducción es interna (1). Si la reducción es en la raíz, tenemos dos posibilidades.

  1. La reducción es por la regla (App), (Var), (VarR) o (Lamb). Cada uno de estos casos es análogo a su contraparte en el lema 4.36, cambiando λx por λxgc y λre por λre$_{\mathrm{gc}}$.
  2. La reducción es por la regla (GC). Es decir, tenemos que $x \notin \mathrm{FV}(t_1)$ y, por lo tanto,
  $$t_1[x := t_2] \underset{(\mathrm{GC})}{\to_{\lambda\mathrm{xgc}}} t_1 = u$$
  Luego,
  $$\mathrm{w}_{\bar{x}_{1:n}}(t_1[x := t_2]) \underset{\mathrm{def}}{=} \mathrm{w}_{x:\bar{x}_{1:n}}(t_1)[\mathrm{w}_{\bar{x}_{1:n}}(t_2)] \underset{(\mathrm{GC}),\ \mathrm{L.5.13}}{\to_{\lambda\mathrm{re}_{\mathrm{gc}}}}$$
  $$\downarrow_1(\mathrm{w}_{x:\bar{x}_{1:n}}(t_1)) \underset{\mathrm{L.5.15}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1)$$

□

Un paso de reducción en λre$_{\mathbf{gc}}$ implica un paso de reducción en λxgc

Al igual que para la preservación de la reducción bajo la traducción w, vamos a necesitar aquí dos lemas adicionales previos a la demostración del lema principal. Dichos lemas son análogos a los presentados para w, radicando su diferencia en que éstos se prueban para la traducción u. Los presentamos a continuación, seguidos del lema principal.

El primero de los dos lemas muestra la preservación de no pertenencia de variables libres con la traducción u (análogo a lo hecho para w en la subsección anterior).

Lema 5.17. Sea $a \in \Lambda\text{re} : \text{FV}(a) \subseteq \{1, \ldots, n\}$. Sean $\{x_1, \ldots, x_n\}$ variables distintas, y sea $m \in \mathbb{N} : 1 \leq m \leq n+1$. Luego, si $m \notin \text{FV}(a)$, tenemos que para todo $x \notin \{x_1, \ldots, x_n\}$, $x \notin \text{FV}\big(\text{u}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(a)\big)$.

Demostración. Por inducción en a. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}$. Entonces, tenemos que ver que para todo $x \notin \{x_1, \ldots, x_n\}$, $m \notin \text{FV}(a) \implies x \notin \text{FV}(\text{u}_{\bar{z}}(a))$.

- $a = k \in \{1, \ldots, n\}$. Como $k \neq m$, tenemos que

$$\text{u}_{\bar{z}}(k) = x_i \neq x, \text{ con } 1 \leq i \leq n$$

  Luego, $\text{FV}(\text{u}_{\bar{z}}(k)) \subseteq \{x_1, \ldots, x_n\} \not\ni x$.

- $a = b\,c$. Luego,

$$\text{FV}(\text{u}_{\bar{z}}(b\,c)) \underset{\text{def}}{=} \text{FV}(\text{u}_{\bar{z}}(b)\ \text{u}_{\bar{z}}(c)) \underset{\text{def}}{=} \text{FV}(\text{u}_{\bar{z}}(b)) \cup \text{FV}(\text{u}_{\bar{z}}(c))$$

  Por hipótesis inductiva, $x \notin \text{FV}(\text{w}_{\bar{z}}(D))$, con $D \in \{b, c\}$. Luego, tampoco está en la unión.

- $a = \lambda b$. Sea $y \notin \{x, x_1, \ldots, x_n\}$. Luego,

$$\text{FV}(\text{u}_{\bar{z}}(\lambda b)) \underset{\text{def}}{=} \text{FV}(\lambda y.\text{u}_{y:\bar{z}}(b)) \underset{\text{def}}{=} \text{FV}(\text{u}_{y:\bar{z}}(b)) - \{y\}$$

  Como $m \notin \text{FV}(\lambda b) \implies (m+1) \notin \text{FV}(b)$, tenemos que, por hipótesis inductiva, $x \notin \text{FV}(\text{u}_{y:\bar{z}}(b))$. Luego, $x \notin \text{FV}(\text{u}_{y:\bar{z}}(b)) - \{y\}$.

- $a = b[c]$. Sea $y \notin \{x, x_1, \ldots, x_n\}$. Luego,

$$\text{FV}(\text{u}_{\bar{z}}(b[c])) \underset{\text{def}}{=} \text{FV}(\text{u}_{y:\bar{z}}(b)\,[y := \text{u}_{\bar{z}}(c)]) \underset{\text{def}}{=}$$

$$(\text{FV}(\text{u}_{y:\bar{z}}(b)) - \{y\}) \cup \text{FV}(\text{u}_{\bar{z}}(c))$$

  Como $m \notin \text{FV}(b[c]) \implies (m+1) \notin \text{FV}(b) \wedge m \notin \text{FV}(c)$, tenemos que, por hipótesis inductiva, $x \notin \text{FV}(\text{u}_{y:\bar{z}}(b)) \wedge x \notin \text{FV}(\text{u}_{\bar{z}}(c))$. Luego, $x \notin \text{FV}(\text{u}_{y:\bar{z}}(b)) - \{y\}$. Entonces, $x \notin (\text{FV}(\text{u}_{y:\bar{z}}(b)) - \{y\}) \cup \text{FV}(\text{u}_{\bar{z}}(c))$.

□

Observación 5.18. De manera análoga al lema 5.17, podemos probar que para todo $a \in \Lambda\text{re} : \text{FV}(a) \subseteq \{1, \ldots, n\}$, todo $\{x_1, \ldots, x_n\}$ conjunto de variables distintas, y todo $m \in \mathbb{N} : 1 \leq m \leq n+1$, ocurre que si $m \in \text{FV}(a)$, tenemos que para toda variable $x \notin \{x_1, \ldots, x_n\}$, $x \in \text{FV}\big(\text{u}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(a)\big)$.

El siguiente y último de los lemas auxiliares, nos da la relación entre los decrementos de índice y la traducción u.

Lema 5.19 (Interacción entre la traducción u y $\downarrow_i$)**.** Sea $a \in \Lambda\text{re} : \text{FV}(a) \subseteq \{1, \ldots, n\}$. Sean $\{x_1, \ldots, x_n\}$ variables distintas, y sea $m \in \mathbb{N} : 1 \leq m \leq n+1$. Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego, si $m \notin \text{FV}(a)$, tenemos que

$$\text{u}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(a) =_\alpha \text{u}_{\bar{x}_{1:n}}(\downarrow_m(a))$$

Demostración. Por inducción en a. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}$. Entonces, tenemos que ver que:

$$\text{u}_{\bar{z}}(a) =_\alpha \text{u}_{\bar{x}_{1:n}}(\downarrow_m(a))$$

- $a = k \in \{1, \ldots, n\}$. Como $k \neq m$, tenemos que

$$\text{u}_{\bar{z}}(k) = z_k = \begin{cases} x_k & \text{si } k < m \\ x_{k-1} & \text{si } k > m \end{cases}$$

  Ahora,

$$\text{u}_{\bar{x}_{1:n}}(\downarrow_m(k)) = \begin{cases} \text{u}_{\bar{x}_{1:n}}(k) = x_k & \text{si } k < m \\ \text{u}_{\bar{x}_{1:n}}(k-1) = x_{k-1} & \text{si } k > m \end{cases}$$

  Por definición de α-equivalencia, $x_k =_\alpha x_k \wedge x_{k-1} =_\alpha x_{k-1}$.

- $a = b\,c$. Luego,

$$\text{u}_{\bar{x}_{1:n}}(\downarrow_m(b\,c)) \underset{\text{def}}{=} \text{u}_{\bar{x}_{1:n}}(\downarrow_m(b)\ \downarrow_m(c)) \underset{\text{def}}{=} \text{u}_{\bar{x}_{1:n}}(\downarrow_m(b))\ \text{u}_{\bar{x}_{1:n}}(\downarrow_m(c)) \underset{\text{HI}}{=_\alpha}$$

$$\text{u}_{\bar{z}}(b)\ \text{u}_{\bar{z}}(c) \underset{\text{def}}{=} \text{u}_{\bar{z}}(b\,c)$$

- $a = \lambda b$. Sea $y \notin \{x, x_1, \ldots, x_n\}$.
  Recordemos que $m \notin \text{FV}(\lambda b) \implies (m+1) \notin \text{FV}(b)$. Luego,

$$\text{u}_{\bar{x}_{1:n}}(\downarrow_m(\lambda b)) \underset{\text{def}}{=} \text{u}_{\bar{x}_{1:n}}(\lambda\, \downarrow_{m+1}(b)) \underset{\text{def}}{=} \lambda y.\text{u}_{y:\bar{x}_{1:n}}(\downarrow_{m+1}(b)) \underset{\text{HI}}{=_\alpha}$$

$$\lambda y.\text{u}_{y:\bar{z}}(b) \underset{\text{def}}{=_\alpha} \text{u}_{\bar{z}}(\lambda b)$$

- $a = b[c]$. Sea $y \notin \{x, x_1, \ldots, x_n\}$.
  Al igual que para el caso anterior, tengamos presente que
  $m \notin \text{FV}(b[c]) \implies (m+1) \notin \text{FV}(b) \wedge m \notin \text{FV}(c)$. Luego,

$$\text{u}_{\bar{x}_{1:n}}(\downarrow_m(b[c])) \underset{\text{def}}{=} \text{u}_{\bar{x}_{1:n}}(\downarrow_{m+1}(b)[\downarrow_m(c)]) \underset{\text{def}}{=}$$

$$\text{u}_{y:\bar{x}_{1:n}}(\downarrow_{m+1}(b))\,[y := \text{u}_{\bar{x}_{1:n}}(\downarrow_m(c))] \underset{\text{HI}}{=_\alpha} \text{u}_{y:\bar{z}}(b)\,[y := \text{u}_{\bar{z}}(c)] \underset{\text{def}}{=_\alpha} \text{u}_{\bar{z}}(b[c])$$

□

Dados los dos lemas presentados arriba, mostramos, ahora sí, la preservación de la reducción λre$_{\text{gc}}$ bajo la traducción u. De la misma manera que para la preservación de la reducción λxgc bajo w, el único caso interesante es el de la reducción por *Garbage Collection* (GC) pues, siguiendo lo que acontece para la prueba del lema 5.16, los casos restantes son un reflejo del lema 4.39.

Lema 5.20 (Preservación de la reducción λre$_{gc}$ bajo u)**.** Para todo $a, b \in \Lambda re$, tenemos que:

$$a \to_{\lambda re_{gc}} b \implies u(a) \to_{\lambda xgc} u(b)$$

Demostración. Por inducción en a. Sin pérdida de generalidad, supongamos que $FV(a) \subseteq \{1, \ldots, n\}$. Como $a \to_{\lambda re_{gc}} b$ tenemos, por lema 5.12, que $FV(b) \subseteq FV(a)$. Sean $x_1, \ldots, x_n$ las primeras n variables de la enumeración de $\mathbb{V}$ elegida para u. Luego, tenemos que $u(a) = u_{\bar{x}_{1:n}}(a)$ y que $u(b) = u_{\bar{x}_{1:n}}(u)$.

- $a = k \in \{1, \ldots, n\}$. No hay reducción posible. Luego, vale vacuamente.
- $a = a_1\, a_2$. Análogo a su contraparte en el lema 4.39, cambiando λx por λxgc y λre por λre$_{gc}$.
- $a = \lambda a_1$. Análogo a su contraparte en el lema 4.39, cambiando λx por λxgc y λre por λre$_{gc}$.
- $a = a_1[a_2]$. Si la reducción es interna (es decir, $a_i \to_{\lambda re_{gc}} a_i',\ i \in \{1, 2\}$), procedemos de manera similar al caso de aplicación del lema 4.39, cuando la reducción es interna (1), y cambiando λx por λxgc y λre por λre$_{gc}$. Si la reducción es en la raíz, tenemos dos posibilidades.

  1. La reducción es por la regla (App), (Var), (VarR) o (Lamb). Cada uno de estos casos es análogo a su contraparte en el lema 4.39, cambiando λx por λxgc y λre por λre$_{gc}$.
  2. La reducción es por la regla (GC). Es decir, tenemos que $1 \notin FV(a_1)$ y, por lo tanto,
  $$a_1[a_2] \underset{(GC)}{\to_{\lambda re_{gc}}} \downarrow_1(a_1) = b$$
  Sea $x \notin \{x_1, \ldots, x_n\}$. Luego,
  $$u_{\bar{x}_{1:n}}(a_1[a_2]) \underset{def}{=} u_{x:\bar{x}_{1:n}}(a_1)\,[x := u_{\bar{x}_{1:n}}(a_2)] \underset{(GC),\ L.5.17}{\to_{\lambda xgc}}$$
  $$u_{x:\bar{x}_{1:n}}(a_1) \underset{L.5.19}{=} u_{\bar{x}_{1:n}}(\downarrow_1(a_1))$$

□

A continuación, enunciamos y probamos el isomorfismo entre λxgc y λre$_{gc}$, objetivo principal de la presente sección.

Teorema 5.21 (Isomorfismo entre λxgc y λre$_{gc}$)**.** Los cálculos λxgc y λre$_{gc}$ son isomorfos.

Demostración. Consecuencia directa de la definición de isomorfismo, de las traducciones expuestas (w y u), y de los lemas 4.31, 4.32, 5.16 y 5.20. □

Corolario 5.22. Los cálculos λre$_{gc}$ y λxgc tienen las mismas propiedades

Demostración. Consecuencia directa del teorema 5.21 □

El corolario 5.22 nos dice que el cálculo λre$_{gc}$ tiene las mismas propiedades que λxgc. Éste, a su vez, tiene las mismas que las de los cálculos λx y λre. Es decir, λre$_{gc}$ tiene: (Sim), (Snd), (CR$_{re_{gc}}$), (SN$_{re_{gc}}$), (CR) y (PSN).

Capítulo 6

Cálculo λrex

6.1 El cálculo de sustituciones explícitas λrex

6.1.1 Introducción

Presentamos aquí el último de los cálculos del trabajo, que deriva de $\lambda\text{re}_{\text{gc}}$, presentado en el capítulo anterior. El objetivo es la obtención de un cálculo con índices de *de Bruijn* que mantenga las propiedades del cálculo del que deriva, y que a su vez posea confluencia en términos abiertos.

Para el desarrollo, nos basamos en el cálculo λex [Kes08, Kes09], presentado en la sección 1.7.4, formalismo con nombres que posee *todas* las buenas propiedades, incluyendo la mencionada confluencia en términos abiertos. No casualmente, dicho cálculo es un derivado de λxgc, radicando la diferencia en el agregado de una regla de composición y del cocientado de los términos a través de una relación de equivalencia dada por una ecuación, aspecto novedoso en términos de este tipo de cálculos, como se mencionó en su introducción. Dado que el cálculo λex es λxgc más las mencionadas composición y ecuación, y que nuestro cálculo $\lambda\text{re}_{\text{gc}}$ es isomorfo a λxgc, procedemos como es de esperarse: encontrar la regla y ecuación análoga para extender $\lambda\text{re}_{\text{gc}}$ a un cálculo más cercano a λex. Más aún, el formalismo resultante resulta ser isomorfo a λex, tal y como veremos luego de presentarlo.

De esta manera, obtenemos un tercer cálculo isomorfo a un cálculo con nombres ya estudiado y, hasta donde sabemos, el primero con índices de *de Bruijn* y notación clásica que posee *todas* las propiedades esperables de un cálculo de este estilo.

A continuación pasamos a explicar la elección de las reglas del cálculo, siguiendo con su presentación; y continuando luego con las pruebas del isomorfismo con λex. Por último, cerramos con la extensión de λrex a términos abiertos y el isomorfismo existente entre dicha extensión y la extensión correspondiente de λex, de modo tal de mostrar metaconfluencia.

6.1.2 Discusión sobre las reglas de composición

Damos aquí una explicación de naturaleza intuitiva acerca de la regla de composición y de la ecuación que agregaremos a $\lambda\text{re}_{\text{gc}}$ en base a λex, de modo de

obtener el cálculo λrex. Cabe destacar que los términos sobre los que operará λrex serán los mismos que los de los cálculos λre y λre_{gc}: Λre.

La novedad del cálculo λex es la inclusión de una regla de composición y del cocientado de los términos del cálculo mediante (además de la ya clásica α-equivalencia) una nueva ecuación, llamada ecuación C. Como ya se introdujo anteriormente en la sección 1.7.4 de este trabajo, la regla de composición y la ecuación son de la forma:

$$\begin{array}{llll} t[x := u][y := v] & \to_{(\text{Comp})} & t[y := v][x := u[y := v]] & \text{si } y \in \text{FV}(u) \\ t[x := u][y := v] & =_{\text{C}} & t[y := v][x := u] & \text{si } y \notin \text{FV}(u) \wedge x \notin \text{FV}(v) \end{array}$$

Cabe destacar que, en el caso de la regla de composición, puede asumirse por convención de Barendregt que $x \neq y \wedge x \notin \text{FV}(v)$.

La idea para el cálculo λrex sería, por ende, agregar la regla de composición y la ecuación C al cálculo. Comencemos por la regla de composición. En un término de la forma $a[b][c]$, si quisiéramos invertir el orden de aplicación de las sustituciones, deberíamos ver cuándo el término b se ve afectado por c. Analicemos qué ocurre para $a = 1$.

$$1[b][c] \underset{(\text{Var})}{\to} b[c]$$

Aquí, la única manera en que el término c afecte a b, es si $1 \in \text{FV}(b)$. Esta será, justamente, la condición de la regla. Ahora bien, en el término $a[b][c]$, la sustitución que contiene a c está afectando (debido al binding impuesto por los índices de *de Bruijn*) a la variable libre 2 en a; y la sustitución con b, a la variable libre 1. Si fuéramos a aplicar la sustitución de c antes que la de b, tendríamos – a priori –, un término de la forma $a'[c'][b']$. En este término, la sustitución que contiene a c' está reemplazando las variables libres 1; y la que contiene a b', a las variables 2. Por ende, a' debería ser el resultado de realizar un *swap* de 1 por 2 sobre a, de modo tal de no modificar la semántica del término. Tenemos, entonces, el término $\updownarrow_1(a)[c'][b']$. Ahora bien: notemos aquí que la sustitución que tiene el término c' ha "atravesado" la sustitución con el término b'. Esto quiere decir, además del *swap* que esto provoca sobre a, que sus variables libres deben ser incrementadas en uno. Por ende, tenemos: $\updownarrow_1(a)[\uparrow_0(c)][b']$. Por último, observemos que la sustitución que contiene a c debería afectar a b, dada la condición de la composición. Como ya vimos para el caso $a = 1$, quedaría el término $b[c]$ en el lugar de b'. Esto es exactamente lo que hacemos, y con esto ya tenemos la regla de composición para nuestro cálculo:

$$a[b][c] \to_{(\text{Comp})} \updownarrow_1(a)[\uparrow_0(c)][b[c]] \qquad \text{si } 1 \in \text{FV}(b)$$

Cabría preguntarse por qué si hay que incrementar los índices de c al atravesar la sustitución, no habría que decrementar los de b. La respuesta es: la sustitución que tiene al término c aplicada a b ya se encarga de hacerlo. Podría uno preguntarse también qué ocurre con la segunda condición sobre la composición en λex: $x \notin \text{FV}(v)$, condición que se asume por convención de Barendregt. La respuesta es: no es necesario hacerlo aquí, dado que no necesitamos trabajar módulo α-equivalencia. Más aún, la condición se cumple "sola". Intuitivamente, se asume $x \notin \text{FV}(v)$ para que, al aplicar la regla, no tengamos que la sustitución con x afecte al término v, dado que en el término original no lo hacía. Decimos que la condición se cumple "sola" en nuestro cálculo, porque la regla

incrementa los índices de c. De esta manera, jamás ocurre que luego de aplicar la composición, $1 \in \mathrm{FV}(\uparrow_0(c))$.

Para la explicación de la ecuación análoga a la ecuación C de λex (que llamaremos ecuación D), podemos basarnos en la regla de composición. Supongamos por un momento que negamos la condición de ésta. Tendríamos, entonces,

$$a[b][c] \to_{(\mathrm{Comp})_{\mathrm{D}}} \updownarrow_1(a)[\uparrow_0(c)][b[c]] \qquad \text{si } 1 \notin \mathrm{FV}(b)$$

Ahora bien, dada dicha condición, tenemos que reduciendo por *Garbage Collection*,

$$\updownarrow_1(a)[\uparrow_0(c)][b[c]] \to_{(\mathrm{GC})} \updownarrow_1(a)[\uparrow_0(c)][\downarrow_1(b)]$$

Obviamente, no podemos quitar la condición de la regla de composición y dejar que la regla de *Garbage Collection* haga su magia. Esto se debe a que podríamos generar derivaciones inifinitas, reduciendo con la regla de composición cuantas veces queramos. Además, como se explicó en la introducción a λex (sección 1.7.4), tampoco podríamos hacer que la ecuación D sea una regla de reducción, pues ocurriría lo mismo. Dicho esto, nuestra ecuación D será:

$$a[b][c] =_{\mathrm{D}} \updownarrow_1(a)[\uparrow_0(c)][\downarrow_1(b)] \qquad \text{si } 1 \notin \mathrm{FV}(b)$$

Notar que $\downarrow_1(b)$ está bien definido por la condición de la ecuación. Notar también que, de manera análoga a lo acontecido para la regla de composición, no necesitamos hacer una asunción análoga a $x \notin \mathrm{FV}(v)$ de λex, pues la naturaleza de los índices de *de Bruijn* se encarga de solucionarnos el problema (recordar que $1 \notin \mathrm{FV}(\uparrow_0(c))$).

Por último, hacemos una observación importante sobre la ecuación D: es posible "volver" al término original. Supongamos que $1 \notin \mathrm{FV}(b)$. Luego,

$$a[b][c] \underset{1 \notin \mathrm{FV}(b)}{=_{\mathrm{D}}} \updownarrow_1(a)[\uparrow_0(c)][\downarrow_1(b)] \underset{1 \notin \mathrm{FV}(\uparrow_0(c))}{=_{\mathrm{D}}} \updownarrow_1(\updownarrow_1(a))[\uparrow_0(\downarrow_1(b))][\downarrow_1(\uparrow_0(c))] = a[b][c]$$

pues, como podemos verificar fácilmente:

$$\forall a \in \Lambda\mathrm{re} : \updownarrow_1(\updownarrow_1(a)) = a \ \wedge (1 \notin \mathrm{FV}(a) \implies \uparrow_0(\downarrow_1(a)) = a) \ \wedge \downarrow_1(\uparrow_0(a)) = a$$

6.1.3 Presentación del cálculo

Antes de presentar el cálculo, y como ya mencionamos en la sección anterior, los términos de λrex son los mismos que los de $\lambda\mathrm{re}_{\mathrm{gc}}$ y λre, siendo la diferencia la regla de composición y la ecuación. Por ende, toda definición y lema que sólo hable de términos y metaoperadores, sin utilizar reglas particulares de cada cálculo, son válidos aquí también. No así, por ejemplo, para el caso del no agregado de variables libres por reducción (y por la nueva ecuación), proposición que mostramos válida más adelante. Dicho esto, los términos del cálculo λrex los notaremos igual que para los cálculos anteriores: Λre.

Por último, antes de presentar el cálculo, es importante destacar que quitamos la regla (VarR), dado que el cálculo λex no tiene la regla equivalente. Recordamos de la presentación de $\lambda\mathrm{re}_{\mathrm{gc}}$ (sección 5.1) que, de todas formas, dicha regla puede simularse utilizando la regla (GC), con lo que no hay mayores diferencias.

Podemos ver entonces, hechas las aclaraciones pertinentes, las reglas y ecuaciones del cálculo λrex en la figura 6.1.

(EqD)	$a[b][c]$	$=$	$\uparrow_1(a)[\uparrow_0(c)][\downarrow_1(b)]$	$(1 \notin \mathrm{FV}(b))$
(Beta)	$(\lambda a)\, b$	$\to$	$a[b]$	
(App)	$(a\, b)[c]$	$\to$	$a[c]\ b[c]$	
(Lamb)	$(\lambda a)[c]$	$\to$	$\lambda\, \uparrow_1(a)[\uparrow_0(c)]$	
(Var)	$1[c]$	$\to$	c	
(GC)	$a[c]$	$\to$	$\downarrow_1(a)$	$(1 \notin \mathrm{FV}(a))$
(Comp)	$a[b][c]$	$\to$	$\uparrow_1(a)[\uparrow_0(c)][b[c]]$	$(1 \in \mathrm{FV}(b))$

Figura 6.1: Reglas del cálculo λrex

Llamamos $\mathrm{rex_p}$ al conjunto de reglas $\mathrm{re_{gc}}$+ (Comp) - (VarR); y $\lambda\mathrm{rex_p}$ al conjunto de reglas (Beta) + $\mathrm{rex_p}$.

Pasamos a definir formalmente el cálculo λrex.

Definición 6.1 ($=_\mathrm{D}$-equivalencia)**.** Se define la relación $=_\mathrm{D} \subseteq \Lambda\mathrm{re} \times \Lambda\mathrm{re}$ en base a las siguientes reglas de inferencia, con $a, b, c, a', b', c' \in \Lambda\mathrm{re}$:

$$\frac{}{a =_\mathrm{D} a}\ (\textsc{Refl}_\mathrm{D}) \qquad \frac{1 \notin \mathrm{FV}(b)}{a[b][c] =_\mathrm{D} \uparrow_1(a)[\uparrow_0(c)][\downarrow_1(b)]}\ (\textsc{EqD})$$

$$\frac{a =_\mathrm{D} a'}{a' =_\mathrm{D} a}\ (\textsc{Sym}_\mathrm{D}) \qquad \frac{a =_\mathrm{D} b \qquad b =_\mathrm{D} c}{a =_\mathrm{D} c}\ (\textsc{Trans}_\mathrm{D})$$

$$\frac{a =_\mathrm{D} a' \qquad b =_\mathrm{D} b'}{a\, b =_\mathrm{D} a'\, b'}\ (\textsc{App}_\mathrm{D}) \qquad \frac{a =_\mathrm{D} a'}{\lambda a =_\mathrm{D} \lambda a'}\ (\textsc{Abs}_\mathrm{D})$$

$$\frac{a =_\mathrm{D} a' \qquad b =_\mathrm{D} b'}{a[b] =_\mathrm{D} a'[b']}\ (\textsc{Sust}_\mathrm{D})$$

Es decir, $=_\mathrm{D}$ es la relación de equivalencia más chica y compatible con contextos generada por la ecuación (EqD). Aclaramos que, aunque no lo hayamos expuesto en la presentación, asumimos la existencia de un sistema de inferencia análogo que define la relación $=_\mathrm{C}$ para el cálculo λex, sólo que utilizando la ecuación (EqC). Desde el aspecto notacional, las reglas de inferencia que poseen la letra D como subíndice (por ejemplo ($\textsc{App}_\mathrm{D}$)), encuentran su contraparte en el sistema de inferencia para la relación de equivalencia $=_\mathrm{C}$ cambiando el subíndice D por C (por ejemplo, ($\textsc{Sust}_\mathrm{D}$) por ($\textsc{Sust}_\mathrm{C}$)).

Definición 6.2 ($\mathrm{rex_p}$-reducción)**.** Se define la relación $\mathrm{rex_p} \subseteq \Lambda\mathrm{re} \times \Lambda\mathrm{re}$, notada $\to_{\mathrm{rex_p}}$ como la clausura contextual de las reglas $\mathrm{rex_p}$.

Definición 6.3 ($\lambda\mathrm{rex_p}$-reducción)**.** Se define la relación $\lambda\mathrm{rex_p} \subseteq \Lambda\mathrm{re} \times \Lambda\mathrm{re}$, notada $\to_{\lambda\mathrm{rex_p}}$ como la clausura contextual de las reglas $\lambda\mathrm{rex_p}$.

Definición 6.4 (rex-reducción)**.** Se define la relación $\mathrm{rex} \subseteq \Lambda\mathrm{re} \times \Lambda\mathrm{re}$, notada $\to_{\mathrm{rex}}$ como:

$$\forall a, b \in \Lambda\mathrm{re} : a \to_{\mathrm{rex}} b \iff \big(\exists c, d \in \Lambda\mathrm{re} : a =_\mathrm{D} c \to_{\mathrm{rex_p}} d =_\mathrm{D} b\big)$$

Definición 6.5 (λrex-reducción)**.** Se define la relación $\lambda\text{rex} \subseteq \Lambda\text{re} \times \Lambda\text{re}$, notada $\to_{\lambda\text{rex}}$ como:

$$\forall a, b \in \Lambda\text{re} : a \to_{\lambda\text{rex}} b \iff \big(\exists c, d \in \Lambda\text{re} : a =_{\text{D}} c \to_{\lambda\text{rex}_{\text{p}}} d =_{\text{D}} b\big)$$

Definición 6.6 (Cálculo λrex)**.** El cálculo λrex es el sistema de reducción $(\Lambda\text{re}, \lambda\text{rex})$

6.1.4 No agregado de variables libres por reducción

Al igual que para los cálculos λre y $\lambda\text{re}_{\text{gc}}$, mostramos aquí que la relación λrex no agrega variables libres. Para mostrar esto, tenemos que ver que:

$$a \to_{\lambda\text{rex}} b \implies \text{FV}(b) \subseteq \text{FV}(a)$$

Ahora bien,

$$a \to_{\lambda\text{rex}} b \underset{\text{def}}{\iff} \big(\exists c, d \in \Lambda\text{re} : a =_{\text{D}} c \to_{\lambda\text{rex}_{\text{p}}} d =_{\text{D}} b\big)$$

Por ende, mostramos dos cosas: primero, que dos términos D-equivalentes tienen las mismas variables libres. Segundo, que la $\lambda\text{rex}_{\text{p}}$-reducción no agrega variables libres. De esta manera, habremos probado que la λrex-reducción no agrega variables libres.

Mostramos primero que dos términos D-equivalentes tienen las mismas variables libres.

Lema 6.7 (Igualdad de variables libres a través de $=_{\text{D}}$-conversión)**.** Para todo $a, b \in \Lambda\text{re}$, tenemos que

$$a =_{\text{D}} b \implies \text{FV}(a) = \text{FV}(b)$$

Demostración. Por inducción en la inferencia de $a =_{\text{D}} b$.

- La inferencia es por la regla (Refl), (Sym) o (Trans). Vale pues la igualdad es relación de equivalencia (*i.e.,* usar hipótesis inductiva e inferir por la misma regla para la igualdad de conjuntos).
- La inferencia es por la regla (App). Tenemos $a = c\,d =_{\text{D}} c'\,d' = b$, con $c =_{\text{D}} c' \wedge d =_{\text{D}} d'$. Luego,

$$\text{FV}(c\,d) \underset{\text{def}}{=} \text{FV}(c) \cup \text{FV}(d) \underset{\text{HI}}{=} \text{FV}(c') \cup \text{FV}(d') \underset{\text{def}}{=} \text{FV}(c'\,d')$$

- La inferencia es por la regla (Abs). Tenemos $a = \lambda c =_{\text{D}} \lambda c' = b$, con $c =_{\text{D}} c'$. Luego,

$$\text{FV}(\lambda c) \underset{\text{def}}{=} \text{FV}(c) - 1 \underset{\text{HI}}{=} \text{FV}(c') - 1 \underset{\text{def}}{=} \text{FV}(\lambda c')$$

- La inferencia es por la regla (Sust). Tenemos $a = c[d] =_{\text{D}} c'[d'] = b$, con $c =_{\text{D}} c' \wedge d =_{\text{D}} d'$. Luego,

$$\text{FV}(c[d]) \underset{\text{def}}{=} (\text{FV}(c) - 1) \cup \text{FV}(d) \underset{\text{HI}}{=} (\text{FV}(c') - 1) \cup \text{FV}(d') \underset{\text{def}}{=} \text{FV}(c'[d'])$$

- La inferencia es por la regla (EqD).
  Tenemos $a = c[d][e] =_{\mathrm{D}} \updownarrow_1(c)[\uparrow_0(e)][\downarrow_1(d)] = b$, con $1 \notin \mathrm{FV}(d)$. Luego,

$$\mathrm{FV}(\updownarrow_1(c)[\uparrow_0(e)][\downarrow_1(d)]) \underset{\text{def}}{=} (\,\mathrm{FV}(\updownarrow_1(c)[\uparrow_0(e)]) - 1\,) \cup \mathrm{FV}(\downarrow_1(d)) \underset{\text{def}}{=}$$

$$\{\,[\,(\,\mathrm{FV}(\updownarrow_1(c)) - 1\,) \cup \mathrm{FV}(\uparrow_0(e))\,] - 1\,\} \cup \mathrm{FV}(\downarrow_1(d)) \underset{\text{L.1.43.1/3}}{=}$$

$$(\,\mathrm{FV}(\updownarrow_1(c)) - 2\,) \cup (\,\mathrm{FV}(\uparrow_0(e)) - 1\,) \cup \mathrm{FV}(\downarrow_1(d)) \underset{\text{L.4.16}}{=}$$

$$(\,\mathrm{FV}(\updownarrow_1(c)) - 2\,) \cup \mathrm{FV}(e) \cup \mathrm{FV}(\downarrow_1(d)) \underset{\text{L.4.15}}{=}$$

$$(\,\mathrm{FV}(c) - 2\,) \cup \mathrm{FV}(e) \cup \mathrm{FV}(\downarrow_1(d)) \underset{\text{O.5.11}}{=}$$

$$(\,\mathrm{FV}(c) - 2\,) \cup \mathrm{FV}(e) \cup (\,\mathrm{FV}(d) - 1\,) \underset{\text{L.1.43.1/3}}{=}$$

$$\{\,[\,(\,\mathrm{FV}(c) - 1\,) \cup \mathrm{FV}(d)\,] - 1\,\} \cup \mathrm{FV}(e) \underset{\text{def}}{=}$$

$$(\,\mathrm{FV}(c[d]) - 1\,) \cup \mathrm{FV}(e) \underset{\text{def}}{=} \mathrm{FV}(c[d][e])$$

□

Como segundo paso, mostramos que la relación $\lambda\mathrm{rex}_{\mathrm{p}}$ no agrega variables libres.

Lema 6.8 (No agregado de variables libres por $\lambda\mathrm{rex}_{\mathrm{p}}$-reducción)**.** Para todo $a, b \in \Lambda\mathrm{re}$, tenemos que:

$$a \to_{\lambda\mathrm{rex}_{\mathrm{p}}} b \implies \mathrm{FV}(b) \subseteq \mathrm{FV}(a)$$

Demostración. Por inducción en a.

- $a = n \in \mathbb{N}_{>0}$. No hay reducción posible. Luego, vale vacuamente.
- $a = c\,d$. Análogo al lema 4.17, cambiando $\lambda\mathrm{re}$ por $\lambda\mathrm{rex}_{\mathrm{p}}$.
- $a = \lambda c$. Análogo al lema 4.17, cambiando $\lambda\mathrm{re}$ por $\lambda\mathrm{rex}_{\mathrm{p}}$.
- $a = c[d]$. Si la reducción es interna (es decir, $e \to_{\lambda\mathrm{rex}_{\mathrm{p}}} e',\ e \in \{c, d\}$), procedemos de manera similar al caso de aplicación del lema 4.17, cuando la reducción es interna (1), y cambiando $\lambda\mathrm{re}$ por $\lambda\mathrm{rex}_{\mathrm{p}}$. Si la reducción es en la raíz, tenemos tres posibilidades.
  1. La reducción es por la regla (App), (Var) o (Lamb). Cada uno de estos casos es análogo a su contraparte en el lema 4.17, cambiando $\lambda\mathrm{re}$ por $\lambda\mathrm{rex}_{\mathrm{p}}$.
  2. La reducción es por la regla (GC). Análogo a su contraparte en el lema 5.12, cambiando $\lambda\mathrm{re}_{\mathrm{gc}}$ por $\lambda\mathrm{rex}_{\mathrm{p}}$.
  3. La reducción es por la regla (Comp). Tenemos

$$c[d] = e[f][d] \underset{\text{(Comp)}}{\to_{\lambda\mathrm{rex}_{\mathrm{p}}}} \updownarrow_1(e)[\uparrow_0(d)][f[d]] = b$$

  con $1 \in \mathrm{FV}(f)$. Luego,

$$\mathrm{FV}(\updownarrow_1(e)[\uparrow_0(d)][f[d]]) \underset{\text{def}}{=} (\,\mathrm{FV}(\updownarrow_1(e)[\uparrow_0(d)]) - 1\,) \cup \mathrm{FV}(f[d]) \underset{\text{def}}{=}$$

$$(\,\mathrm{FV}(\Uparrow_1(e)[\uparrow_0(d)])-1\,)\cup[\,(\,\mathrm{FV}(f)-1\,)\cup\mathrm{FV}(d)\,]\underset{\text{def}}{=}$$

$$\{\,[\,(\,\mathrm{FV}(\Uparrow_1(e))-1\,)\cup\mathrm{FV}(\uparrow_0(d))\,]-1\,\}\cup[\,(\,\mathrm{FV}(f)-1\,)\cup\mathrm{FV}(d)\,]\underset{\text{L.1.43.1/3}}{=}$$

$$[\,(\,\mathrm{FV}(\Uparrow_1(e))-2\,)\cup(\,\mathrm{FV}(\uparrow_0(d))-1\,)\,]\cup[\,(\,\mathrm{FV}(f)-1\,)\cup\mathrm{FV}(d)\,]\underset{\text{L.4.16}}{=}$$

$$[\,(\,\mathrm{FV}(\Uparrow_1(e))-2\,)\cup\mathrm{FV}(d)\,]\cup[\,(\,\mathrm{FV}(f)-1\,)\cup\mathrm{FV}(d)\,]\underset{\text{L.4.15}}{=}$$

$$[\,(\,\mathrm{FV}(e)-2\,)\cup\mathrm{FV}(d)\,]\cup[\,(\,\mathrm{FV}(f)-1\,)\cup\mathrm{FV}(d)\,]\underset{\text{asoc. y conmut. unión}}{=}$$

$$(\,\mathrm{FV}(e)-2\,)\cup(\,\mathrm{FV}(f)-1\,)\cup\mathrm{FV}(d)$$

Además,

$$\mathrm{FV}(e[f][d])\underset{\text{def}}{=}(\,\mathrm{FV}(e[f])-1\,)\cup\mathrm{FV}(d)\underset{\text{def}}{=}$$

$$\{\,[\,(\,\mathrm{FV}(e)-1\,)\cup\mathrm{FV}(f)\,]-1\,\}\cup\mathrm{FV}(d)\underset{\text{L.1.43.1/3}}{=}$$

$$(\,\mathrm{FV}(e)-2\,)\cup(\,\mathrm{FV}(f)-1\,)\cup\mathrm{FV}(d)$$

□

Ahora sí, como consecuencia de los dos lemas anteriores, podemos formular y probar el lema que nos asegura el no agregado de variables libres por λrex-reducción.

Lema 6.9 (No agregado de variables libres por λrex-reducción)**.** Para todo $a, b \in \Lambda\mathrm{re}$, tenemos que:

$$a \to_{\lambda\mathrm{rex}} b \implies \mathrm{FV}(b) \subseteq \mathrm{FV}(a)$$

Demostración. Por definición, $a \to_{\lambda\mathrm{rex}} b \iff \big(\exists c, d \in \Lambda\mathrm{re} : a =_{\mathrm{D}} c \to_{\lambda\mathrm{rex_p}} d =_{\mathrm{D}} b\big)$. Luego,

$$\mathrm{FV}(b)\underset{\text{L.6.7}}{=}\mathrm{FV}(d)\underset{\text{L.6.8}}{\subseteq}\mathrm{FV}(c)\underset{\text{L.6.7}}{=}\mathrm{FV}(a)$$

□

6.2 Isomorfismo entre λex y λrex

6.2.1 Introducción

De manera análoga a lo escrito en las secciones 4.2 y 5.2, el objetivo de la presente sección es enunciar y demostrar el isomorfismo que existe entre los cálculos λex y λrex. Con este desarrollo confirmamos lo que, hasta donde tenemos conocimiento, es el primer cálculo con índices de *de Bruijn* y sustituciones explícitas que posee un conjunto deseable de *buenas* propiedades, a saber: (Sim), (Snd), ($\mathrm{CR_{rex}}$), ($\mathrm{SN_{rex}}$), (CR), (MC) y (PSN), entre otras. El caso particular de la metaconfluencia se verá en la sección 6.3, en donde también presentamos la extensión de λrex a términos abiertos.

6.2.2 Traducciones entre Λx y Λre

Al igual que para el isomorfismo entre λxgc y $\lambda\text{re}_{\text{gc}}$, las traducciones que utilizamos aquí son las mismas que para el isomorfismo entre λx y λre, dado que los términos de los tres cálculos son los mismos. Remitimos al lector, por lo tanto, a las definiciones 4.18, 4.24, 4.22 y 4.28 a modo de facilitar la lectura de las pruebas, puesto que no repetiremos dichas definiciones aquí.

6.2.3 Pruebas del isomorfismo

Repetimos de las secciones 4.2.3 y 5.2.3 que, para mostrar que λex y λrex son isomorfos, es necesario demostrar las siguientes aserciones:

1. $\mathrm{w} \circ \mathrm{u} = \mathrm{Id}_{\Lambda\mathrm{re}} \wedge \mathrm{u} \circ \mathrm{w} = \mathrm{Id}_{\Lambda\mathrm{x}}$
2. $\forall t, u \in \Lambda\mathrm{x} : t \to_{\lambda\mathrm{ex}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{rex}} \mathrm{w}(u)$
3. $\forall a, b \in \Lambda\mathrm{re} : a \to_{\lambda\mathrm{rex}} b \implies \mathrm{u}(a) \to_{\lambda\mathrm{ex}} \mathrm{u}(b)$

Remitimos al lector a la sección 4.2.3 para la revisión de la primera de las pruebas (*i.e.,* la composición de las traducciones da como resultado la función identidad), dado que no es necesario repetir dichas pruebas.

Pasamos, entonces, a las pruebas de preservación de reducción por traducción.

Un paso de reducción en λex implica un paso de reducción en λrex

A diferencia de lo desarrollado en las secciones 4.2.3 y 5.2.3, no necesitamos aquí ningún lema adicional a los ya presentados para probar la preservación de la reducción λex por la traducción w. Ahora bien, dado que en [Kes08, Kes09] la λex-reducción se define como:

$$\forall t, t' \in \Lambda\mathrm{x} : t \to_{\lambda\mathrm{ex}} t' \iff (\exists s, s' \in \Lambda\mathrm{x} : t =_{\mathrm{C}} s \to_{\mathrm{Bx}} s' =_{\mathrm{C}} t')$$

sí vamos a necesitar dos lemas intermedios para concluir la mentada preservación de la reducción bajo w. El primero nos asegura que la traducción preserva la relación de C-equivalencia. Es decir, términos C-equivalentes en λex resultan ser D-equivalentes al realizarse la traducción. El segundo y último nos dice que la traducción preserva la reducción sin considerar C-equivalencia. Esto es, todo par de términos relacionados mediante la relación Bx, al traducirse se relacionan mediante la relación $\mathrm{rex}_{\mathrm{p}}$. Veamos el primero de ellos.

Lema 6.10 (Preservación de la relación de equivalencia C bajo la traducción w)**.** Para todo $t, u \in \Lambda\mathrm{x}$, tenemos que:

$$t =_{\mathrm{C}} u \implies \mathrm{w}(t) =_{\mathrm{D}} \mathrm{w}(u)$$

Demostración. Por inducción en la inferencia de $t =_{\mathrm{C}} u$. Sin pérdida de generalidad, supongamos que $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Como $t =_{\mathrm{C}} u$, tenemos que $\mathrm{FV}(t) = \mathrm{FV}(u)$. Sean $x_1, \ldots, x_n$ las primeras n variables de la enumeración de $\mathbb{V}$ elegida para w. Luego, tenemos que $\mathrm{w}(t) = \mathrm{w}_{\bar{x}_{1:n}}(t)$ y que $\mathrm{w}(u) = \mathrm{w}_{\bar{x}_{1:n}}(u)$.

- La inferencia es por la regla ($\textsc{Refl}_{\mathrm{C}}$), ($\textsc{Sym}_{\mathrm{C}}$) o ($\textsc{Trans}_{\mathrm{C}}$). Vale trivialmente, pues la D-equivalencia es una relación de equivalencia (*i.e.,* usar hipótesis inductiva y concluir por la misma regla para la D-equivalencia).

- La inferencia es por la regla ($\textsc{App}_\textsc{C}$). Tenemos $t = t_1\, t_2 =_\textsc{C} t_1'\, t_2' = u$, con $t_1 =_\textsc{C} t_1' \wedge t_2 =_\textsc{C} t_2'$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1\, t_2) \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1)\ \mathrm{w}_{\bar{x}_{1:n}}(t_2) \underset{\text{HI, } (\textsc{App}_\textsc{D})}{=_\textsc{D}}$$

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1')\ \mathrm{w}_{\bar{x}_{1:n}}(t_2') \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1'\, t_2')$$

- La inferencia es por la regla ($\textsc{Abs}_\textsc{C}$). Tenemos $t = \lambda x.t_1 =_\textsc{C} \lambda x.t_1' = u$, con $t_1 =_\textsc{C} t_1'$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.t_1) \underset{\text{def}}{=} \lambda \mathrm{w}_{x:\bar{x}_{1:n}}(t_1) \underset{\text{HI, } (\textsc{Abs}_\textsc{D})}{=_\textsc{D}}$$

$$\lambda \mathrm{w}_{x:\bar{x}_{1:n}}(t_1') \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(\lambda x.t_1')$$

- La inferencia es por la regla ($\textsc{Sust}_\textsc{C}$).
  Tenemos $t = t_1[x := t_2] =_\textsc{C} t_1'[x := t_2'] = u$, con $t_1 =_\textsc{C} t_1' \wedge t_2 =_\textsc{C} t_2'$.

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1[x := t_2]) \underset{\text{def}}{=} \mathrm{w}_{x:\bar{x}_{1:n}}(t_1)[\mathrm{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{HI, } (\textsc{Sust}_\textsc{D})}{=_\textsc{D}}$$

$$\mathrm{w}_{x:\bar{x}_{1:n}}(t_1')[\mathrm{w}_{\bar{x}_{1:n}}(t_2')] \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1'[x := t_2'])$$

- La inferencia es por la regla (EQC).
  Tenemos $t = t_1[y := t_2][x := t_3] =_\textsc{C} t_1[x := t_3][y := t_2] = u$,
  con $x \neq y \wedge x \notin \mathrm{FV}(t_2) \wedge y \notin \mathrm{FV}(t_3)$. Cabe destacar que asumimos, por convención de Barendregt, que $\{x, y\} \cap \{x_1, \ldots, x_n\} = \emptyset$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(t_1[y := t_2][x := t_3]) \underset{\text{def}}{=} \mathrm{w}_{x:\bar{x}_{1:n}}(t_1[y := t_2])[\mathrm{w}_{\bar{x}_{1:n}}(t_3)] \underset{\text{def}}{=}$$

$$\mathrm{w}_{y:x:\bar{x}_{1:n}}(t_1)[\mathrm{w}_{x:\bar{x}_{1:n}}(t_2)][\mathrm{w}_{\bar{x}_{1:n}}(t_3)] =_\mathrm{D}$$

  Ahora, como $x \notin \mathrm{FV}(t_2)$, tenemos que, por lema 5.13, $1 \notin \mathrm{FV}(\mathrm{w}_{x:\bar{x}_{1:n}}(t_2))$.
  Por definición, podemos aplicar ecuación D (utilizando la regla (EQD)):

$$=_\mathrm{D} \updownarrow_1(\mathrm{w}_{y:x:\bar{x}_{1:n}}(t_1))[\uparrow_0(\mathrm{w}_{\bar{x}_{1:n}}(t_3))][\downarrow_1(\mathrm{w}_{x:\bar{x}_{1:n}}(t_2))] \underset{\text{L.4.34}}{=}$$

$$\mathrm{w}_{x:y:\bar{x}_{1:n}}(t_1)[\uparrow_0(\mathrm{w}_{\bar{x}_{1:n}}(t_3))][\downarrow_1(\mathrm{w}_{x:\bar{x}_{1:n}}(t_2))] \underset{\text{L.4.35, } y \notin \mathrm{FV}(t_3)}{=}$$

$$\mathrm{w}_{x:y:\bar{x}_{1:n}}(t_1)[\mathrm{w}_{y:\bar{x}_{1:n}}(t_3)][\downarrow_1(\mathrm{w}_{x:\bar{x}_{1:n}}(t_2))] \underset{\text{L.5.15, } x \notin \mathrm{FV}(t_2)}{=}$$

$$\mathrm{w}_{x:y:\bar{x}_{1:n}}(t_1)[\mathrm{w}_{y:\bar{x}_{1:n}}(t_3)][\mathrm{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=}$$

$$\mathrm{w}_{y:\bar{x}_{1:n}}(t_1[x := t_3])[\mathrm{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(t_1[x := t_3][y := t_2])$$

□

Presentamos a continuación el segundo de los ya mentados lemas auxiliares.

Lema 6.11 (Preservación de la reducción Bx bajo w)**.** Para todo $t, u \in \Lambda\mathrm{x}$, tenemos que:

$$t \to_{\mathrm{Bx}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{rex_p}} \mathrm{w}(u)$$

Demostración. Por inducción en t. Sin pérdida de generalidad, supongamos que $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Como $t \to_{\mathrm{Bx}} u$, tenemos que $\mathrm{FV}(u) \subseteq \mathrm{FV}(t)$. Sean $x_1, \ldots, x_n$ las primeras n variables de la enumeración de $\mathbb{V}$ elegida para w. Luego, tenemos que $\mathrm{w}(t) = \mathrm{w}_{\bar{x}_{1:n}}(t)$ y que $\mathrm{w}(u) = \mathrm{w}_{\bar{x}_{1:n}}(u)$.

- $t = x \in \{x_1, \ldots, x_n\}$. No hay reducción posible. Luego, vale vacuamente.
- $t = t_1\, t_2$. Análogo a su contraparte en el lema 4.36, cambiando λx por Bx y λre por $\lambda\mathrm{rex_p}$.
- $t = \lambda x.t_1$. Análogo a su contraparte en el lema 4.36, cambiando λx por Bx y λre por $\lambda\mathrm{rex_p}$.
- $t = t_1[x := t_2]$. Si la reducción es interna (es decir, $t_i \to_{\mathrm{Bx}} t_i',\ i \in \{1, 2\}$), procedemos de manera similar al caso de aplicación del lema 4.36, cuando la reducción es interna (1). Si la reducción es en la raíz, tenemos tres posibilidades.

  1. La reducción es por la regla (App), (Var) o (Lamb). Cada uno de estos casos es análogo a su contraparte en el lema 4.36, cambiando λx por Bx y λre por $\lambda\mathrm{rex_p}$.
  2. La reducción es por la regla (GC). Análogo a su contraparte en el lema 5.16, cambiando λxgc por Bx y $\lambda\mathrm{re_{gc}}$ por $\lambda\mathrm{rex_p}$.
  3. La reducción es por la regla (Comp). Tenemos

     $$t_1[x := t_2] = t_3[y := t_4][x := t_2] \underset{\text{(Comp)}}{\to_{\mathrm{Bx}}} t_3[x := t_2][y := t_4[x := t_2]] = u$$

     con $x \in \mathrm{FV}(t_4)$. Cabe destacar que, por convención de Barendregt, asumimos $x \neq y \wedge y \notin \{x_1, \ldots, x_n\}$. Luego,

     $$\mathrm{w}_{\bar{x}_{1:n}}(t_3[y := t_4][x := t_2]) \underset{\text{def}}{=} \mathrm{w}_{x:\bar{x}_{1:n}}(t_3[y := t_4])[\mathrm{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{def}}{=}$$

     $$\mathrm{w}_{y:x:\bar{x}_{1:n}}(t_3)[\mathrm{w}_{x:\bar{x}_{1:n}}(t_4)][\mathrm{w}_{\bar{x}_{1:n}}(t_2)] \underset{\text{O.5.14, } x \in \mathrm{FV}(t_4),\ \text{(Comp)}}{\to_{\lambda\mathrm{rex_p}}}$$

     $$\uparrow_1(\mathrm{w}_{y:x:\bar{x}_{1:n}}(t_3))[\uparrow_0(\mathrm{w}_{\bar{x}_{1:n}}(t_2))][\mathrm{w}_{x:\bar{x}_{1:n}}(t_4)[\mathrm{w}_{\bar{x}_{1:n}}(t_2)]] \underset{\text{L.4.34}}{=}$$

     $$\mathrm{w}_{x:y:\bar{x}_{1:n}}(t_3)[\uparrow_0(\mathrm{w}_{\bar{x}_{1:n}}(t_2))][\mathrm{w}_{x:\bar{x}_{1:n}}(t_4)[\mathrm{w}_{\bar{x}_{1:n}}(t_2)]] \underset{\text{L.4.35, } y \notin \mathrm{FV}(t_2)}{=}$$

     $$\mathrm{w}_{x:y:\bar{x}_{1:n}}(t_3)[\mathrm{w}_{y:\bar{x}_{1:n}}(t_2)][\mathrm{w}_{x:\bar{x}_{1:n}}(t_4)[\mathrm{w}_{\bar{x}_{1:n}}(t_2)]] \underset{\text{def}}{=}$$

     $$\mathrm{w}_{x:y:\bar{x}_{1:n}}(t_3)[\mathrm{w}_{y:\bar{x}_{1:n}}(t_2)][\mathrm{w}_{\bar{x}_{1:n}}(t_4[x := t_2])] \underset{\text{def}}{=}$$

     $$\mathrm{w}_{y:\bar{x}_{1:n}}(t_3[x := t_2])[\mathrm{w}_{\bar{x}_{1:n}}(t_4[x := t_2])] \underset{\text{def}}{=}$$

     $$\mathrm{w}_{\bar{x}_{1:n}}(t_3[x := t_2][y := t_4[x := t_2]])$$

□

Presentados los dos lemas auxiliares veamos, ahora sí, la preservación de la reducción λex bajo w.

Lema 6.12 (Preservación de la reducción λex bajo w)**.** Para todo $t, u \in \Lambda\mathrm{x}$, tenemos que:

$$t \to_{\lambda\mathrm{ex}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{rex}} \mathrm{w}(u)$$

Demostración. Por definición,

$$t \to_{\lambda\mathrm{ex}} u \underset{\text{def}}{\iff} (\exists t', u' \in \Lambda\mathrm{x} : t =_{\mathrm{C}} t' \to_{\mathrm{Bx}} u' =_{\mathrm{C}} u)$$

Ahora, por lema 6.10, $\mathrm{w}(t) =_{\mathrm{D}} \mathrm{w}(t') \wedge \mathrm{w}(u) =_{\mathrm{D}} \mathrm{w}(u')$; además, por lema 6.11, $\mathrm{w}(t') \to_{\lambda\mathrm{rex_p}} \mathrm{w}(u')$. Por lo tanto, utilizando la definición de λrex-reducción:

$$\mathrm{w}(t) =_{\mathrm{D}} \mathrm{w}(t') \to_{\lambda\mathrm{rex_p}} \mathrm{w}(u') =_{\mathrm{D}} \mathrm{w}(u) \iff \mathrm{w}(t) \to_{\lambda\mathrm{rex}} \mathrm{w}(u)$$

□

Un paso de reducción en λrex implica un paso de reducción en λex

Al igual que para la preservación de la reducción λex bajo w, no vamos a necesitar ningún lema adicional a los presentados en las secciones 4.2.3 y 5.2.3. De todas maneras, de forma análoga a lo presentado en el apartado anterior, sí vamos a requerir dos lemas auxiliares para probar lo que queremos. Uno de ellos nos garantiza la preservación de la relación de D-equivalencia a través de la traducción u (*i.e.,* términos D-equivalentes se traducen en términos C-equivalentes). El otro hace lo propio para las relaciones de reducción $\lambda\mathrm{rex_p}$ y Bx.

Mostramos, entonces, el primero de los lemas auxiliares.

Lema 6.13 (Preservación de la relación de equivalencia D bajo la traducción u)**.** Para todo $a, b \in \Lambda\mathrm{re}$, tenemos que:

$$a =_{\mathrm{D}} b \implies \mathrm{u}(a) =_{\mathrm{C}} \mathrm{u}(b)$$

Demostración. Por inducción en la inferencia de $a =_{\mathrm{D}} b$. Sin pérdida de generalidad, supongamos que $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$. Como $a =_{\mathrm{D}} b$, tenemos que, por lema 6.7, $\mathrm{FV}(a) = \mathrm{FV}(b)$. Sean $x_1, \ldots, x_n$ las primeras n variables de la enumeración de $\mathbb{V}$ elegida para u. Luego, tenemos que $\mathrm{u}(a) = \mathrm{u}_{\bar{x}_{1:n}}(a)$ y que $\mathrm{u}(b) = \mathrm{u}_{\bar{x}_{1:n}}(b)$.

- La inferencia es por la regla ($\textsc{Refl}_{\mathrm{D}}$), ($\textsc{Sym}_{\mathrm{D}}$) o ($\textsc{Trans}_{\mathrm{D}}$). Vale trivialmente, pues la C-equivalencia es una relación de equivalencia (*i.e.,* usar hipótesis inductiva y concluir por la misma regla para la C-equivalencia).

- La inferencia es por la regla ($\textsc{App}_{\mathrm{D}}$). Tenemos $a = a_1\, a_2 =_{\mathrm{D}} a'_1\, a'_2 = b$, con $a_1 =_{\mathrm{D}} a'_1 \wedge a_2 =_{\mathrm{D}} a'_2$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(a_1\, a_2) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(a_1)\ \mathrm{u}_{\bar{x}_{1:n}}(a_2) \underset{\text{HI, } (\textsc{App}_{\mathrm{C}})}{=_{\mathrm{C}}}$$

$$\mathrm{u}_{\bar{x}_{1:n}}(a'_1)\ \mathrm{u}_{\bar{x}_{1:n}}(a'_2) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(a'_1\, a'_2)$$

- La inferencia es por la regla (Abs_D). Tenemos $a = \lambda a_1 =_\text{D} \lambda a_1' = b$, con $a_1 =_\text{D} a_1'$. Sea $x \notin \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(\lambda a_1) \underset{\text{def}}{=} \lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(a_1) \underset{\text{HI, } (\text{Abs}_\text{C})}{=_\text{C}}$$

$$\lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(a_1') \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(\lambda a_1')$$

- La inferencia es por la regla (Sust_D). Tenemos $a = a_1[a_2] =_\text{D} a_1'[a_2'] = b$, con $a_1 =_\text{D} a_1' \wedge a_2 =_\text{D} a_2'$. Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(a_1[a_2]) \underset{\text{def}}{=} \mathrm{u}_{x:\bar{x}_{1:n}}(a_1)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_2)] \underset{\text{HI, } (\text{Sust}_\text{C})}{=_\text{C}}$$

$$\mathrm{u}_{x:\bar{x}_{1:n}}(a_1')\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_2')] \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(a_1'[a_2'])$$

- La inferencia es por la regla (EqD).
  Tenemos $a = a_1[a_2][a_3] =_\text{D} \uparrow_1(a_1)[\uparrow_0(a_3)][\downarrow_1(a_2)] = b$, con $1 \notin \text{FV}(a_2)$. Sean $x, y \notin \{x_1, \ldots, x_n\}$, $x \neq y$. Entonces,

$$\mathrm{u}_{\bar{x}_{1:n}}(a_1[a_2][a_3]) \underset{\text{def}}{=} \mathrm{u}_{x:\bar{x}_{1:n}}(a_1[a_2])\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_3)] \underset{\text{def}}{=}$$

$$\mathrm{u}_{y:x:\bar{x}_{1:n}}(a_1)\,[y := \mathrm{u}_{x:\bar{x}_{1:n}}(a_2)][x := \mathrm{u}_{\bar{x}_{1:n}}(a_3)] =_\text{C}$$

  Ahora, como $1 \notin \text{FV}(a_2)$, tenemos que, por lema 5.17, $x \notin \text{FV}(\mathrm{u}_{x:\bar{x}_{1:n}}(a_2))$. Además, por lema 4.30, $\text{FV}(\mathrm{u}_{\bar{x}_{1:n}}(a_3)) \subseteq \{x_1, \ldots, x_n\}$. Ergo, como $y \notin \{x_1, \ldots, x_n\}$, tenemos que $y \notin \text{FV}(\mathrm{u}_{\bar{x}_{1:n}}(a_3))$. Por definición, podemos aplicar ecuación C (utilizando la regla (EqC)):

$$=_\text{C} \mathrm{u}_{y:x:\bar{x}_{1:n}}(a_1)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_3)][y := \mathrm{u}_{x:\bar{x}_{1:n}}(a_2)] \underset{\text{L.4.37}}{=}$$

$$\mathrm{u}_{x:y:\bar{x}_{1:n}}(\updownarrow_1(a_1))\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_3)][y := \mathrm{u}_{x:\bar{x}_{1:n}}(a_2)] \underset{\text{L.4.38}}{=}$$

$$\mathrm{u}_{x:y:\bar{x}_{1:n}}(\updownarrow_1(a_1))\,[x := \mathrm{u}_{y:\bar{x}_{1:n}}(\uparrow_0(a_3))][y := \mathrm{u}_{x:\bar{x}_{1:n}}(a_2)] \underset{\text{L.5.19, } 1 \notin \text{FV}(a_2)}{=}$$

$$\mathrm{u}_{x:y:\bar{x}_{1:n}}(\updownarrow_1(a_1))\,[x := \mathrm{u}_{y:\bar{x}_{1:n}}(\uparrow_0(a_3))][y := \mathrm{u}_{\bar{x}_{1:n}}(\downarrow_1(a_2))] \underset{\text{def}}{=}$$

$$\mathrm{u}_{y:\bar{x}_{1:n}}(\updownarrow_1(a_1)[\uparrow_0(a_3)])\,[y := \mathrm{u}_{\bar{x}_{1:n}}(\downarrow_1(a_2))] \underset{\text{def}}{=}$$

$$\mathrm{u}_{\bar{x}_{1:n}}(\updownarrow_1(a_1)[\uparrow_0(a_3)][\downarrow_1(a_2)])$$

□

Mostramos ahora el segundo de los lemas auxiliares que necesitamos.

Lema 6.14 (Preservación de la reducción $\lambda\text{rex}_\text{p}$ bajo u)**.** Para todo $a, b \in \Lambda\text{re}$, tenemos que:

$$a \to_{\lambda\text{rex}_\text{p}} b \implies \mathrm{u}(a) \to_{\text{Bx}} \mathrm{u}(b)$$

Demostración. Por inducción en a. Sin pérdida de generalidad, supongamos que $\text{FV}(a) \subseteq \{1, \ldots, n\}$. Como $a \to_{\lambda\text{rex}_\text{p}} b$, tenemos que, por lema 6.8, $\text{FV}(b) \subseteq \text{FV}(a)$. Sean $x_1, \ldots, x_n$ las primeras n variables de la enumeración de $\mathbb{V}$ elegida para u. Luego, tenemos que $\mathrm{u}(a) = \mathrm{u}_{\bar{x}_{1:n}}(a)$ y que $\mathrm{u}(b) = \mathrm{u}_{\bar{x}_{1:n}}(b)$.

- $a = k \in \{1, \ldots, n\}$. No hay reducción posible. Luego, vale vacuamente.
- $a = a_1\, a_2$. Análogo a su contraparte en el lema 4.39, cambiando λx por Bx y λre por $\lambda\mathrm{rex}_\mathrm{p}$.
- $a = \lambda a_1$. Análogo a su contraparte en el lema 4.39, cambiando λx por Bx y λre por $\lambda\mathrm{rex}_\mathrm{p}$.
- $a = a_1[a_2]$. Si la reducción es interna (es decir, $a_i \to_{\lambda\mathrm{rex}_\mathrm{p}} a_i',\ i \in \{1, 2\}$), procedemos de manera similar al caso de aplicación del lema 4.39, cuando la reducción es interna (1). Si la reducción es en la raíz, tenemos tres posibilidades.
  1. La reducción es por la regla (App), (Var) o (Lamb). Cada uno de estos casos es análogo a su contraparte en el lema 4.39, cambiando λx por Bx y λre por $\lambda\mathrm{rex}_\mathrm{p}$.
  2. La reducción es por la regla (GC). Análogo a su contraparte en el lema 5.20, cambiando λxgc por Bx y $\lambda\mathrm{re}_\mathrm{gc}$ por $\lambda\mathrm{rex}_\mathrm{p}$.
  3. La reducción es por la regla (Comp). Tenemos

$$a_1[a_2] = a_3[a_4][a_2] \underset{\text{(Comp)}}{\to_{\lambda\mathrm{rex}_\mathrm{p}}} \uparrow_1(a_3)[\uparrow_0(a_2)][a_4[a_2]] = b$$

  con $1 \in \mathrm{FV}(a_4)$. Sean además, $x, y \notin \{x_1, \ldots, x_n\}$, $x \neq y$. Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}(a_3[a_4][a_2]) \underset{\text{def}}{=} \mathrm{u}_{x:\bar{x}_{1:n}}(a_3[a_4])\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_2)] \underset{\text{def}}{=}$$

$$\mathrm{u}_{y:x:\bar{x}_{1:n}}(a_3)\,[y := \mathrm{u}_{x:\bar{x}_{1:n}}(a_4)][x := \mathrm{u}_{\bar{x}_{1:n}}(a_2)] \underset{\text{O.5.18, } 1 \in \mathrm{FV}(a_4),\ \text{(Comp)}}{\to_{\mathrm{Bx}}}$$

$$\mathrm{u}_{y:x:\bar{x}_{1:n}}(a_3)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_2)][y := \mathrm{u}_{x:\bar{x}_{1:n}}(a_4)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_2)]] \underset{\text{L.4.37}}{=}$$

$$\mathrm{u}_{x:y:\bar{x}_{1:n}}(\updownarrow_1(a_3))\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_2)][y := \mathrm{u}_{x:\bar{x}_{1:n}}(a_4)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_2)]] \underset{\text{L.4.38}}{=}$$

$$\mathrm{u}_{x:y:\bar{x}_{1:n}}(\updownarrow_1(a_3))\,[x := \mathrm{u}_{y:\bar{x}_{1:n}}(\uparrow_0(a_2))][y := \mathrm{u}_{x:\bar{x}_{1:n}}(a_4)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(a_2)]] \underset{\text{def}}{=}$$

$$\mathrm{u}_{x:y:\bar{x}_{1:n}}(\updownarrow_1(a_3))\,[x := \mathrm{u}_{y:\bar{x}_{1:n}}(\uparrow_0(a_2))][y := \mathrm{u}_{\bar{x}_{1:n}}(a_4[a_2])] \underset{\text{def}}{=}$$

$$\mathrm{u}_{y:\bar{x}_{1:n}}(\updownarrow_1(a_3)[\uparrow_0(a_2)])\,[y := \mathrm{u}_{\bar{x}_{1:n}}(a_4[a_2])] \underset{\text{def}}{=}$$

$$\mathrm{u}_{\bar{x}_{1:n}}(\updownarrow_1(a_3)[\uparrow_0(a_2)][a_4[a_2]])$$

□

Veamos, ya probados los lemas auxiliares, la preservación de la reducción λrex bajo u.

Lema 6.15 (Preservación de la reducción λrex bajo u)**.** Para todo $a, b \in \Lambda\text{re}$, tenemos que:

$$a \to_{\lambda\text{rex}} b \implies \mathrm{u}(a) \to_{\lambda\text{ex}} \mathrm{u}(b)$$

Demostración. Por definición 6.5,

$$a \to_{\lambda\text{rex}} b \underset{\text{def}}{\iff} \left(\exists a', b' \in \Lambda\text{re} : a =_{\mathrm{D}} a' \to_{\lambda\text{rex}_{\mathrm{p}}} b' =_{\mathrm{D}} b\right)$$

Ahora, por lema 6.13, $\mathrm{u}(a) =_{\mathrm{C}} \mathrm{u}(a') \wedge \mathrm{u}(b) =_{\mathrm{C}} \mathrm{u}(b')$; además, por lema 6.14, $\mathrm{u}(a') \to_{\mathrm{Bx}} \mathrm{u}(b')$. Por lo tanto, utilizando la definición de λex-reducción:

$$\mathrm{u}(a) =_{\mathrm{C}} \mathrm{u}(a') \to_{\mathrm{Bx}} \mathrm{u}(b') =_{\mathrm{C}} \mathrm{u}(b) \iff \mathrm{u}(a) \to_{\lambda\text{ex}} \mathrm{u}(b)$$

□

A continuación enunciamos y probamos el isomorfismo entre λex y λrex.

Teorema 6.16 (Isomorfismo entre λex y λrex)**.** Los cálculos λex y λrex son isomorfos.

Demostración. Consecuencia directa de la definición de isomorfismo, de las traducciones expuestas (w y u), y de los lemas 4.31, 4.32, 6.12 y 6.15. □

Corolario 6.17. Los cálculos λrex y λex tienen las mismas propiedades

Demostración. Consecuencia directa del teorema 6.16 □

Debido al corolario 6.17, el cálculo λrex tiene todas las propiedades mencionadas al principio de esta misma sección; sin embargo, aún no hemos probado metaconfluencia. Para mostrar dicha propiedad, debemos primero extender el cálculo λrex a términos abiertos; y, posteriormente, mostrar la existencia de un isomorfismo entre éste y la extensión correspondiente del cálculo λex. Esto lo hacemos en la siguiente sección, para así cerrar el trabajo.

6.3 Metaconfluencia de λrex

6.3.1 Introducción

Tal y como mencionamos sobre el final de la sección anterior, el objetivo de esta sección es el de mostrar que el cálculo λrex posee la propiedad de *metaconfluencia*. Esto es: la extensión a términos abiertos (*i.e.,* términos que admiten metavariables representando, por ejemplo, pruebas incompletas o programas sin terminar) de λrex es confluente. Para lograr probar metaconfluencia para λrex, seguiremos la misma línea que en el curso de todo el trabajo: mostraremos que existe un isomorfismo entre las extensiones a términos abiertos de λex y λrex.

Para probar el isomorfismo, es necesario primero dar la extensión de λrex, trabajo que haremos en la siguiente subsección. Una vez extendido el cálculo, podremos desarrollar las pruebas cuyo último objetivo es el de mostrar el mentado isomorfismo. Dichas demostraciones son análogas a las vistas durante todo el trabajo; y cabe destacar, además, que dado que los lemas expuestos aquí son extensiones de los ya mostrados en las secciones 4.2, 5.2 y 6.2, sólo escribimos los casos correspondientes a metatérminos.

Con esta parte del desarrollo damos por cerrado el presente trabajo, dando posteriormente las conclusiones pertinentes.

6.3.2 Extensión de λrex a términos abiertos

Definición 6.18 (Conjunto de términos $\Lambda\mathrm{re}_{\mathrm{op}}$)**.** El conjunto $\Lambda\mathrm{re}_{\mathrm{op}}$ está dado por:

$$a ::= n \mid a\,a \mid \lambda a \mid a[a] \mid \mathrm{X}_{\Delta} \qquad n \in \mathbb{N}_{>0},\, \mathrm{X} \in \mathbb{M},\, \Delta \in \mathcal{P}(\mathbb{N}_{>0})$$

con $\mathbb{M}$ un conjunto infinito numerable de metavariables $\{\mathrm{X}, \mathrm{Y}, \mathrm{Z}, \ldots\}$ y Δ finito.

Dado que hemos extendido el conjunto de términos para contemplar términos abiertos, debemos asimismo extender la noción de variables libres, así como también el accionar de los metaoperadores $\updownarrow_i$, $\uparrow_i$ y $\downarrow_i$. A continuación, comenzamos extendiendo la definición de variables libres. Para aportar claridad y comodidad al lector, y de la misma forma que hemos hecho en capítulos anteriores, repetimos aquí las definiciones completas.

Definición 6.19 (Variables libres de un elemento de $\Lambda\mathrm{re}_{\mathrm{op}}$)**.** Las variables libres de un término, $\mathrm{FV} : \Lambda\mathrm{re}_{\mathrm{op}} \to \mathcal{P}(\mathbb{N}_{>0})$, se define inductivamente como:

$$\begin{array}{rcl} \mathrm{FV}(n) & = & \{n\} \\ \mathrm{FV}(a\,b) & = & \mathrm{FV}(a) \cup \mathrm{FV}(b) \\ \mathrm{FV}(\lambda a) & = & \mathrm{FV}(a) - 1 \\ \mathrm{FV}(a[b]) & = & (\mathrm{FV}(a) - 1) \cup \mathrm{FV}(b) \\ \mathrm{FV}(\mathrm{X}_{\Delta}) & = & \Delta \end{array}$$

Para la extensión de los metaoperadores $\updownarrow_i$, $\uparrow_i$ y $\downarrow_i$, conjeturamos primero la forma que deberían tener y, posteriormente, probamos la aserción sobre los elementos de $\Lambda\mathrm{re}$. De esta forma, estamos de alguna manera verificando que la definición sea coherente. Mostramos a continuación los tres lemas que nos permiten realizar las mencionadas extensiones; a saber: la forma en que mutan las variables libres de un término al aplicársele alguno de los metaoperadores.

Lema 6.20 (Estado de las variables libres luego de la aplicación de $\uparrow_i$)**.** Para todo $a \in \Lambda\mathrm{re}$, $i \in \mathbb{N}$, tenemos que

$$\mathrm{FV}(\uparrow_i(a)) = \mathrm{FV}(a)_{\leq i} \cup \big(\mathrm{FV}(a)_{>i} + 1\big)$$

Demostración. Por inducción en a. Sea $i \in \mathbb{N}$.

- $a = n \in \mathbb{N}_{>0}$. Luego,

$$\mathrm{FV}(\uparrow_i(n)) = \begin{cases} \{n+1\} & \text{si } n > i \\ \{n\} & \text{si } n \leq i \end{cases}$$

  Dividimos en casos:

  1. $n > i \implies \mathrm{FV}(n)_{\leq i} \cup \big(\mathrm{FV}(n)_{>i} + 1\big) \underset{n\,>\,i}{=} \emptyset \cup (\{n\} + 1) \underset{\mathrm{def}}{=} \{n+1\}$
  2. $n \leq i \implies \mathrm{FV}(n)_{\leq i} \cup \big(\mathrm{FV}(n)_{>i} + 1\big) \underset{n\,\leq\,i}{=} \{n\} \cup (\emptyset + 1) = \{n\}$

- $a = b\,c$. Luego,

$$\mathrm{FV}(\uparrow_i(b\,c)) \underset{\mathrm{def}}{=} \mathrm{FV}(\uparrow_i(b)\ \uparrow_i(c)) \underset{\mathrm{def}}{=} \mathrm{FV}(\uparrow_i(b)) \cup \mathrm{FV}(\uparrow_i(c)) \underset{\mathrm{HI}}{=}$$

$$\mathrm{FV}(b)_{\leq i} \cup \big(\mathrm{FV}(b)_{>i} + 1\big) \cup \mathrm{FV}(c)_{\leq i} \cup \big(\mathrm{FV}(c)_{>i} + 1\big) \underset{\text{L.1.43.9/2}}{=}$$
$$(\mathrm{FV}(b) \cup \mathrm{FV}(c))_{\leq i} \cup ((\mathrm{FV}(b) \cup \mathrm{FV}(c))_{>i} + 1) \underset{\text{def}}{=}$$
$$\mathrm{FV}(b\,c)_{\leq i} \cup \big(\mathrm{FV}(b\,c)_{>i} + 1\big)$$

- $a = \lambda b$. Luego,

$$\mathrm{FV}(\uparrow_i(\lambda b)) \underset{\text{def}}{=} \mathrm{FV}(\lambda \uparrow_{i+1}(b)) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_{i+1}(b)) - 1 \underset{\text{HI}}{=}$$
$$\Big(\mathrm{FV}(b)_{\leq i+1} \cup \big(\mathrm{FV}(b)_{>i+1} + 1\big)\Big) - 1 \underset{\text{L.1.43.1}}{=}$$
$$\Big(\mathrm{FV}(b)_{\leq i+1} - 1\Big) \cup \big(\big(\mathrm{FV}(b)_{>i+1} + 1\big) - 1\big) \underset{\text{L.1.43.5}}{=}$$
$$\Big(\mathrm{FV}(b)_{\leq i+1} - 1\Big) \cup \big(\big(\mathrm{FV}(b)_{>i+1} - 1\big) + 1\big) \underset{\text{L.1.43.8}}{=}$$
$$(\mathrm{FV}(b) - 1)_{\leq i} \cup ((\mathrm{FV}(b) - 1)_{>i} + 1) \underset{\text{def}}{=} \mathrm{FV}(\lambda b)_{\leq i} \cup \big(\mathrm{FV}(\lambda b)_{>i} + 1\big)$$

- $a = b[c]$. Luego,

$$\mathrm{FV}(\uparrow_i(b[c])) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_{i+1}(b)[\uparrow_i(c)]) \underset{\text{def}}{=} (\mathrm{FV}(\uparrow_{i+1}(b)) - 1) \cup \mathrm{FV}(\uparrow_i(c)) \underset{\text{HI}}{=}$$
$$\Big(\Big(\mathrm{FV}(b)_{\leq i+1} \cup \big(\mathrm{FV}(b)_{>i+1} + 1\big)\Big) - 1\Big) \cup \mathrm{FV}(c)_{\leq i} \cup \big(\mathrm{FV}(c)_{>i} + 1\big) \underset{\text{L.1.43.1/5}}{=}$$
$$\Big(\mathrm{FV}(b)_{\leq i+1} - 1\Big) \cup \big(\big(\mathrm{FV}(b)_{>i+1} - 1\big) + 1\big) \cup \mathrm{FV}(c)_{\leq i} \cup \big(\mathrm{FV}(c)_{>i} + 1\big) \underset{\text{L.1.43.8}}{=}$$
$$(\mathrm{FV}(b) - 1)_{\leq i} \cup ((\mathrm{FV}(b) - 1)_{>i} + 1) \cup \mathrm{FV}(c)_{\leq i} \cup \big(\mathrm{FV}(c)_{>i} + 1\big) \underset{\text{L.1.43.2}}{=}$$
$$(\mathrm{FV}(b) - 1)_{\leq i} \cup \mathrm{FV}(c)_{\leq i} \cup \big(\big((\mathrm{FV}(b) - 1)_{>i} \cup \mathrm{FV}(c)_{>i}\big) + 1\big) \underset{\text{L.1.43.9}}{=}$$
$$((\mathrm{FV}(b) - 1) \cup \mathrm{FV}(c))_{\leq i} \cup (((\mathrm{FV}(b) - 1) \cup \mathrm{FV}(c))_{>i} + 1) \underset{\text{def}}{=}$$
$$\mathrm{FV}(b[c])_{\leq i} \cup \big(\mathrm{FV}(b[c])_{>i} + 1\big)$$

□

Lema 6.21 (Estado de las variables libres luego de la aplicación de $\updownarrow_i$)**.** Para todo $a \in \Lambda\mathrm{re}$, $i \in \mathbb{N}_{>0}$, tenemos que

$$\mathrm{FV}(\updownarrow_i(a)) = \mathrm{FV}(a)_{<i} \cup \mathrm{FV}(a)_{>i+1} \cup (\mathrm{FV}(a)_{=i} + 1) \cup \big(\mathrm{FV}(a)_{=i+1} - 1\big)$$

Demostración. Por inducción en a. Sea $i \in \mathbb{N}_{>0}$.

- $a = n \in \mathbb{N}_{>0}$. Luego,

$$\mathrm{FV}(\updownarrow_i(n)) = \begin{cases} \{n\} & \text{si } n < i \vee n > i+1 \\ \{i+1\} & \text{si } n = i \\ \{i\} & \text{si } n = i+1 \end{cases}$$

  Dividimos en casos:

1. $n < i \implies$

$$\mathrm{FV}(n)_{<i} \cup \mathrm{FV}(n)_{>i+1} \cup (\mathrm{FV}(n)_{=i} + 1) \cup \big(\mathrm{FV}(n)_{=i+1} - 1\big) =$$
$$\{n\} \cup \emptyset \cup (\emptyset + 1) \cup (\emptyset - 1) = \{n\}$$

2. $n > i + 1 \implies$

$$\mathrm{FV}(n)_{<i} \cup \mathrm{FV}(n)_{>i+1} \cup (\mathrm{FV}(n)_{=i} + 1) \cup \big(\mathrm{FV}(n)_{=i+1} - 1\big) =$$
$$\emptyset \cup \{n\} \cup (\emptyset + 1) \cup (\emptyset - 1) = \{n\}$$

3. $n = i \implies$

$$\mathrm{FV}(n)_{<i} \cup \mathrm{FV}(n)_{>i+1} \cup (\mathrm{FV}(n)_{=i} + 1) \cup \big(\mathrm{FV}(n)_{=i+1} - 1\big) =$$
$$\emptyset \cup \emptyset \cup (\{i\} + 1) \cup (\emptyset - 1) = \{i + 1\}$$

4. $n = i + 1 \implies$

$$\mathrm{FV}(n)_{<i} \cup \mathrm{FV}(n)_{>i+1} \cup (\mathrm{FV}(n)_{=i} + 1) \cup \big(\mathrm{FV}(n)_{=i+1} - 1\big) =$$
$$\emptyset \cup \emptyset \cup (\emptyset + 1) \cup (\{i + 1\} - 1) = \{i\}$$

- $a = b\,c$. Luego,

$$\mathrm{FV}(\uparrow_i(b\,c)) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_i(b)\ \uparrow_i(c)) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_i(b)) \cup \mathrm{FV}(\uparrow_i(c)) \underset{\text{HI}}{=}$$
$$\mathrm{FV}(b)_{<i} \cup \mathrm{FV}(b)_{>i+1} \cup (\mathrm{FV}(b)_{=i} + 1) \cup \big(\mathrm{FV}(b)_{=i+1} - 1\big) \cup$$
$$\mathrm{FV}(c)_{<i} \cup \mathrm{FV}(c)_{>i+1} \cup (\mathrm{FV}(c)_{=i} + 1) \cup \big(\mathrm{FV}(c)_{=i+1} - 1\big) \underset{\text{L.1.43.9/1/2}}{=}$$
$$(\mathrm{FV}(b) \cup \mathrm{FV}(c))_{<i} \cup (\mathrm{FV}(b) \cup \mathrm{FV}(c))_{>i+1} \cup$$
$$((\mathrm{FV}(b) \cup \mathrm{FV}(c))_{=i} + 1) \cup ((\mathrm{FV}(b) \cup \mathrm{FV}(c))_{=i+1} - 1) \underset{\text{def}}{=}$$
$$\mathrm{FV}(b\,c)_{<i} \cup \mathrm{FV}(b\,c)_{>i+1} \cup (\mathrm{FV}(b\,c)_{=i} + 1) \cup \big(\mathrm{FV}(b\,c)_{=i+1} - 1\big)$$

- $a = \lambda b$. Luego,

$$\mathrm{FV}(\uparrow_i(\lambda b)) \underset{\text{def}}{=} \mathrm{FV}(\lambda\ \uparrow_{i+1}(b)) \underset{\text{def}}{=} \mathrm{FV}(\uparrow_{i+1}(b)) - 1 \underset{\text{HI}}{=}$$
$$\big(\mathrm{FV}(b)_{<i+1} \cup \mathrm{FV}(b)_{>i+2} \cup \big(\mathrm{FV}(b)_{=i+1} + 1\big) \cup \big(\mathrm{FV}(b)_{=i+2} - 1\big)\big) - 1$$
$$\underset{\text{L.1.43.1/5/8}}{=}$$
$$(\mathrm{FV}(b)-1)_{<i} \cup (\mathrm{FV}(b)-1)_{>i+1} \cup ((\mathrm{FV}(b) - 1)_{=i} + 1) \cup ((\mathrm{FV}(b) - 1)_{=i+1} - 1) \underset{\text{def}}{=}$$
$$\mathrm{FV}(\lambda b)_{<i} \cup \mathrm{FV}(\lambda b)_{>i+1} \cup (\mathrm{FV}(\lambda b)_{=i} + 1) \cup \big(\mathrm{FV}(\lambda b)_{=i+1} - 1\big)$$

- $a = b[c]$. Este caso es muy similar a los anteriores y al caso correspondiente en el lema anterior. Por la simpleza de la demostración y la complejidad de la escritura, lo obviamos aquí.

□

Lema 6.22 (Estado de las variables libres luego de la aplicación de $\downarrow_i$)**.** Para todo $a \in \Lambda\text{re}$, $i \in \mathbb{N}_{>0}$, tenemos que, si $i \notin \mathrm{FV}(a)$, entonces

$$\mathrm{FV}(\downarrow_i(a)) = \mathrm{FV}(a)_{<i} \cup \big(\mathrm{FV}(a)_{>i} - 1\big)$$

Demostración. Por inducción en a. Sea $i \in \mathbb{N}_{>0}$.

- $a = n \in \mathbb{N}_{>0}$. Por hipótesis, tenemos que $n \neq i$. Luego,

$$\mathrm{FV}(\downarrow_i(n)) = \begin{cases} \{n-1\} & \text{si } n > i \\ \{n\} & \text{si } n < i \end{cases}$$

  Dividimos en casos:

  1. $n > i \implies \mathrm{FV}(n)_{<i} \cup \big(\mathrm{FV}(n)_{>i} - 1\big) \underset{n\,>\,i}{=} \emptyset \cup (\{n\} - 1) \underset{n\,>\,i}{=} \{n-1\}$
  2. $n < i \implies \mathrm{FV}(n)_{<i} \cup \big(\mathrm{FV}(n)_{>i} - 1\big) \underset{n\,<\,i}{=} \{n\} \cup (\emptyset - 1) = \{n\}$

- $a = b\,c$. Luego,

$$\mathrm{FV}(\downarrow_i(b\,c)) \underset{\text{def}}{=} \mathrm{FV}(\downarrow_i(b)\ \downarrow_i(c)) \underset{\text{def}}{=} \mathrm{FV}(\downarrow_i(b)) \cup \mathrm{FV}(\downarrow_i(c)) \underset{\text{HI}}{=}$$

$$\mathrm{FV}(b)_{<i} \cup \big(\mathrm{FV}(b)_{>i} - 1\big) \cup \mathrm{FV}(c)_{<i} \cup \big(\mathrm{FV}(c)_{>i} - 1\big) \underset{\text{L.1.43.9/1}}{=}$$

$$(\mathrm{FV}(b) \cup \mathrm{FV}(c))_{<i} \cup ((\mathrm{FV}(b) \cup \mathrm{FV}(c))_{>i} - 1) \underset{\text{def}}{=}$$

$$\mathrm{FV}(b\,c)_{<i} \cup \big(\mathrm{FV}(b\,c)_{>i} - 1\big)$$

- $a = \lambda b$. Luego,

$$\mathrm{FV}(\downarrow_i(\lambda b)) \underset{\text{def}}{=} \mathrm{FV}(\lambda\ \downarrow_{i+1}(b)) \underset{\text{def}}{=} \mathrm{FV}(\downarrow_{i+1}(b)) - 1 \underset{\text{HI}}{=}$$

$$\big(\mathrm{FV}(b)_{<i+1} \cup \big(\mathrm{FV}(b)_{>i+1} - 1\big)\big) - 1 \underset{\text{L.1.43.1}}{=}$$

$$\big(\mathrm{FV}(b)_{<i+1} - 1\big) \cup \big(\big(\mathrm{FV}(b)_{>i+1} - 1\big) - 1\big) \underset{\text{L.1.43.8}}{=}$$

$$(\mathrm{FV}(b) - 1)_{<i} \cup ((\mathrm{FV}(b) - 1)_{>i} - 1) \underset{\text{def}}{=} \mathrm{FV}(\lambda b)_{<i} \cup \big(\mathrm{FV}(\lambda b)_{>i} - 1\big)$$

- $a = b[c]$. Luego,

$$\mathrm{FV}(\downarrow_i(b[c])) \underset{\text{def}}{=} \mathrm{FV}(\downarrow_{i+1}(b)[\downarrow_i(c)]) \underset{\text{def}}{=} (\mathrm{FV}(\downarrow_{i+1}(b)) - 1) \cup \mathrm{FV}(\downarrow_i(c)) \underset{\text{HI}}{=}$$

$$\big(\big(\mathrm{FV}(b)_{<i+1} \cup \big(\mathrm{FV}(b)_{>i+1} - 1\big)\big) - 1\big) \cup \mathrm{FV}(c)_{<i} \cup \big(\mathrm{FV}(c)_{>i} - 1\big) \underset{\text{L.1.43.1}}{=}$$

$$\big(\mathrm{FV}(b)_{<i+1} - 1\big) \cup \big(\big(\mathrm{FV}(b)_{>i+1} - 1\big) - 1\big) \cup \mathrm{FV}(c)_{<i} \cup \big(\mathrm{FV}(c)_{>i} - 1\big) \underset{\text{L.1.43.8}}{=}$$

$$(\mathrm{FV}(b) - 1)_{<i} \cup ((\mathrm{FV}(b) - 1)_{>i} - 1) \cup \mathrm{FV}(c)_{<i} \cup \big(\mathrm{FV}(c)_{>i} - 1\big) \underset{\text{L.1.43.1}}{=}$$

$$(\mathrm{FV}(b) - 1)_{<i} \cup \mathrm{FV}(c)_{<i} \cup \big(\big((\mathrm{FV}(b) - 1)_{>i} \cup \mathrm{FV}(c)_{>i}\big) - 1\big) \underset{\text{L.1.43.9}}{=}$$

$$((\mathrm{FV}(b) - 1) \cup \mathrm{FV}(c))_{<i} \cup (((\mathrm{FV}(b) - 1) \cup \mathrm{FV}(c))_{>i} - 1) \underset{\text{def}}{=}$$

$$\mathrm{FV}(b[c])_{<i} \cup \big(\mathrm{FV}(b[c])_{>i} - 1\big)$$

□

Dados estos tres lemas auxiliares, pasamos ahora a extender las definiciones de los metaoperadores.

Para la extensión del metaoperador de incremento de índices, nos basamos en el lema 6.20.

Definición 6.23 (Metaoperador $\uparrow_i$)**.** Para todo $i \in \mathbb{N}$, $\uparrow_i : \Lambda\mathrm{re}_{\mathrm{op}} \to \Lambda\mathrm{re}_{\mathrm{op}}$ se define inductivamente como:

$$\begin{array}{rcl}
\uparrow_i(n) & = & \begin{cases} n & \text{si } n \leq i \\ n+1 & \text{si } n > i \end{cases} \\
\uparrow_i(a\,b) & = & \uparrow_i(a)\ \uparrow_i(b) \\
\uparrow_i(\lambda a) & = & \lambda\, \uparrow_{i+1}(a) \\
\uparrow_i(a[b]) & = & \uparrow_{i+1}(a)[\uparrow_i(b)] \\
\uparrow_i(\mathrm{X}_\Delta) & = & \mathrm{X}_{\Delta'} \qquad \text{con } \Delta' = \Delta_{\leq i} \cup (\Delta_{>i} + 1)
\end{array}$$

Para la extensión del metaoperador de *swap*, utilizamos lo que dice el lema 6.21.

Definición 6.24 (Metaoperador $\updownarrow_i$)**.** Para todo $i \in \mathbb{N}_{>0}$, $\updownarrow_i : \Lambda\mathrm{re}_{\mathrm{op}} \to \Lambda\mathrm{re}_{\mathrm{op}}$ se define inductivamente como:

$$\begin{array}{rcl}
\updownarrow_i(n) & = & \begin{cases} n & \text{si } n < i \ \vee\ n > i+1 \\ i+1 & \text{si } n = i \\ i & \text{si } n = i+1 \end{cases} \\
\updownarrow_i(a\,b) & = & \updownarrow_i(a)\ \updownarrow_i(b) \\
\updownarrow_i(\lambda a) & = & \lambda\, \updownarrow_{i+1}(a) \\
\updownarrow_i(a[b]) & = & \updownarrow_{i+1}(a)[\updownarrow_i(b)] \\
\updownarrow_i(\mathrm{X}_\Delta) & = & \mathrm{X}_{\Delta'} \qquad \text{con } \Delta' = \Delta_{<i} \cup \Delta_{>i+1} \cup (\Delta_{=i} + 1) \cup (\Delta_{=i+1} - 1)
\end{array}$$

Para la extensión del último de los metaoperadores, el de decremento de índices, seguimos al lema 6.22.

Definición 6.25 (Metaoperador $\downarrow_i$)**.** Para todo $i \geq 1$, $\downarrow_i : \Lambda\mathrm{re}_{\mathrm{op}} \to \Lambda\mathrm{re}_{\mathrm{op}}$ se define inductivamente como:

$$\begin{array}{rcl}
\downarrow_i(n) & = & \begin{cases} n & \text{si } n < i \\ \text{indefinido} & \text{si } n = i \\ n-1 & \text{si } n > i \end{cases} \\
\downarrow_i(a\,b) & = & \downarrow_i(a)\ \downarrow_i(b) \\
\downarrow_i(\lambda a) & = & \lambda\, \downarrow_{i+1}(a) \\
\downarrow_i(a[b]) & = & \downarrow_{i+1}(a)[\downarrow_i(b)] \\
\downarrow_i(\mathrm{X}_\Delta) & = & \begin{cases} \mathrm{X}_{\Delta'} \quad \text{con } \Delta' = \Delta_{<i} \cup (\Delta_{>i} - 1) & \text{si } i \notin \Delta \\ \text{indefinido} & \text{si } i \in \Delta \end{cases}
\end{array}$$

Cabe destacar que las relaciones de reducción y la relación de D-equivalencia para la extensión a términos abiertos de λrex se definen de manera análoga a lo hecho para el cálculo λrex "puro"; la diferencia radica en que la definición se hace sobre los elementos de $\Lambda\mathrm{re}_{\mathrm{op}}$. Hacemos especial énfasis en la definición de la relación de D-equivalencia para metatérminos, puesto que uno podría preguntarse qué es lo que ocurre: dado que el sistema de inferencia es el mismo (sólo que aplicado a metatérminos), tenemos que, dadas dos metavariables, éstas son D-equivalentes si y sólo si son iguales (utilizando para esto la regla de inferencia ($\textsc{Refl}_{\mathrm{D}}$)).

6.3.3 No agregado de variables libres por reducción

Así como en la presentación de λre, $\lambda\mathrm{re}_{\mathrm{gc}}$ y λrex mostramos que la reducción no agrega variables libres, debemos mostrarlo también aquí. Para ello, extendemos a continuación las pruebas de los lemas auxiliares que necesitamos. En dichas extensiones, sólo mostraremos el caso en que el término es una metavariable, dado que el resto de los casos se mantiene igual.

Damos primero la extensión del lema 4.15.

Lema 6.26. Para todo $a \in \Lambda\mathrm{re}_{\mathrm{op}}$, $i \in \mathbb{N}_{>0}$, tenemos que $\mathrm{FV}(\updownarrow_i(a)) - (i+1) = \mathrm{FV}(a) - (i+1)$.

Demostración. Por inducción en a. Sea $i \in \mathbb{N}_{>0}$. Basta ver el caso en que $a = \mathrm{X}_\Delta$. Tenemos:

$$\mathrm{FV}(\updownarrow_i(\mathrm{X}_\Delta)) - (i+1) \underset{\text{def}}{=} (\Delta_{<i} \cup \Delta_{>i+1} \cup (\Delta_{=i}+1) \cup (\Delta_{=i+1}-1)) - (i+1) \underset{\text{L.1.43.1}}{=}$$
$$(\Delta_{<i} - (i+1)) \cup (\Delta_{>i+1} - (i+1)) \cup ((\Delta_{=i}+1) - (i+1)) \cup$$
$$((\Delta_{=i+1}-1) - (i+1)) = \Delta_{>i+1} - (i+1) \underset{\text{L.1.43.4}}{=} \Delta - (i+1) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{X}_\Delta) - (i+1)$$

□

Seguimos con la extensión del lema 4.16.

Lema 6.27. Para todo $a \in \Lambda\mathrm{re}$, $i \in \mathbb{N}$, tenemos que $\mathrm{FV}(\uparrow_i(a)) - (i+1) = \mathrm{FV}(a) - i$.

Demostración. Por inducción en a. Sea $i \in \mathbb{N}$. Basta ver el caso en que $a = \mathrm{X}_\Delta$. Tenemos:

$$\mathrm{FV}(\uparrow_i(\mathrm{X}_\Delta)) - (i+1) \underset{\text{def}}{=} (\Delta_{\leq i} \cup (\Delta_{>i}+1)) - (i+1) \underset{\text{L.1.43.1}}{=}$$
$$(\Delta_{\leq i} - (i+1)) \cup ((\Delta_{>i}+1) - (i+1)) = \emptyset \cup ((\Delta_{>i}+1) - (i+1)) \underset{\text{L.1.43.7}}{=}$$
$$(\Delta_{>i} - i) \underset{\text{L.1.43.4}}{=} \Delta - i \underset{\text{def}}{=} \mathrm{FV}(\mathrm{X}_\Delta) - i$$

□

Por último, hacemos una extensión informal a la observación 5.11.

Observación 6.28. Como ya vimos en la observación mencionada, es fácil ver que también para metatérminos se tiene la propiedad:

$$\forall a \in \Lambda\mathrm{re}_{\mathrm{op}} : 1 \notin \mathrm{FV}(a) \implies \mathrm{FV}(\downarrow_1(a)) = \mathrm{FV}(a) - 1$$

Para la demostración, usar directamente las definiciones.

Dadas las extensiones de estos dos lemas y la observación anterior, podemos afirmar que el cálculo λrex sobre términos abiertos tiene la propiedad de no agregado de variables libres por reducción. Es decir,

$$\forall a, b \in \Lambda\mathrm{re}_{\mathrm{op}} : a \to_{\lambda\mathrm{rex}} b \implies \mathrm{FV}(b) \subseteq \mathrm{FV}(a)$$

Esta afirmación proviene del hecho de observar que, en las demostraciones de esta propiedad para λrex (*i.e.,* lemas 6.7, 6.8 y 6.9), sería sólo necesario agregar el caso en que el término es una metavariable para el lema 6.8. Ahora bien, para el caso de metavariable la propiedad vale vacuamente, pues no hay reducción posible. Más aún, el resto de los casos sigue cumpliéndose en todos los lemas, pues valen los lemas auxiliares para $\Lambda\mathrm{re}_{\mathrm{op}}$, como mostramos recién.

6.3.4 Isomorfismo entre las extensiones a términos abiertos de λex y λrex

Traducciones entre $\Lambda\mathrm{x}_{\mathrm{op}}$ y $\Lambda\mathrm{re}_{\mathrm{op}}$

Dado que tanto $\Lambda\mathrm{x}_{\mathrm{op}}$ como $\Lambda\mathrm{re}_{\mathrm{op}}$ se componen respectivamente de los términos de Λx y Λre más el agregado de metavariables anotadas, se vuelve necesario, para mostrar el isomorfismo entre ambas extensiones, extender también las traducciones ya mostradas en la sección 4.2.2 (*i.e.,* w y u), de manera tal de contemplar el caso de metavariable.

A su vez, se hace indispensable mostrar que ambas traducciones continúan siendo función, motivo por el cual extenderemos, luego de cada definición, los lemas correspondientes.

Al igual que como hicimos en el transcurso de todo el trabajo, mostramos las definiciones completas con el sólo propósito de aportar claridad y comodidad.

Comenzamos con la traducción w.

Definición 6.29 (Traducción desde $\Lambda\mathrm{x}_{\mathrm{op}}$ a $\Lambda\mathrm{re}_{\mathrm{op}}$)**.** Para todo $t \in \Lambda\mathrm{x}_{\mathrm{op}},\ n \in \mathbb{N} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, $\mathrm{w}_{[x_1,\ldots,x_n]} : \Lambda\mathrm{x}_{\mathrm{op}} \to \Lambda\mathrm{re}_{\mathrm{op}}$ se define inductivamente como:

$$\begin{array}{rcll}
\mathrm{w}_{\bar{x}_{1:n}}(x) & = & \mathrm{mín}\,\{j : x_j = x\} & (x \in \{x_1, \ldots, x_n\}) \\
\mathrm{w}_{\bar{x}_{1:n}}(t\,u) & = & \mathrm{w}_{\bar{x}_{1:n}}(t)\ \mathrm{w}_{\bar{x}_{1:n}}(u) & \\
\mathrm{w}_{\bar{x}_{1:n}}(\lambda x.t) & = & \lambda \mathrm{w}_{x:\bar{x}_{1:n}}(t) & \\
\mathrm{w}_{\bar{x}_{1:n}}(t[x := u]) & = & \mathrm{w}_{x:\bar{x}_{1:n}}(t)[\mathrm{w}_{\bar{x}_{1:n}}(u)] & \\
\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_{\Delta}) & = & \mathrm{X}_{\Delta'} \qquad \text{con } \Delta' = \{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta\} &
\end{array}$$

Seguimos ahora con la extensión de los lemas 4.19 y 4.20, lemas que muestran la condición de función de la traducción $\mathrm{w}_{\bar{x}_{1:n}}$. Mostramos, tanto en estos lemas como en los que siguen, sólo el caso de metavariable, puesto que el resto se mantiene igual.

Lema 6.30. Para todo $t \in \Lambda\mathrm{x}_{\mathrm{op}},\ n \in \mathbb{N} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, tenemos que $\forall y \notin \{x_1, \ldots, x_n\},\ z \in \{x_1, \ldots, x_n\} : \mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{x}_{1:k-1} \bullet y \bullet \bar{x}_{k+1:n}}(t\{z := y\})$, con $k = \mathrm{mín}\,\{j : x_j = z\}$.

Demostración. Por inducción en t. Sea $k = \mathrm{mín}\,\{j : x_j = z\}$. Por claridad, llamamos

$$\bar{y} = \bar{x}_{1:k-1} \bullet y \bullet \bar{x}_{k+1:n}$$

Es decir, tenemos que ver que: $\mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{y}}(t\{z := y\})$. Basta ver el caso en que $t = \mathrm{X}_{\Delta}$. Recordemos que dado que, por definición, $\mathrm{FV}(\mathrm{X}_{\Delta}) = \Delta$, tenemos que $\Delta \subseteq \{x_1, \ldots, x_n\}$. Ahora,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_{\Delta}) \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta'}$$

$$\text{con } \Delta' = \{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta\}$$

A su vez,

$$\mathrm{w}_{\bar{y}}(\mathrm{X}_{\Delta}\{z := y\}) \underset{\mathrm{def}}{=} \mathrm{w}_{\bar{y}}\big(\mathrm{X}_{\Delta\{z:=y\}}\big) \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta''}$$

$$\text{con } \Delta'' = \{\mathrm{w}_{\bar{y}}(x) : x \in \Delta\{z := y\}\}$$

Es decir, basta ver que $\Delta' = \Delta''$. Ahora bien, dado que ya mostramos que se cumple que

$$\mathrm{w}_{\bar{x}_{1:n}}(x) = \mathrm{w}_{\bar{y}}(x\{z := y\}) \qquad \forall x \in \{x_1, \ldots, x_n\}$$

tenemos que

$$\Delta'' = \{\mathrm{w}_{\bar{y}}(x) : x \in \Delta\{z := y\}\} \underset{\text{def}}{=}$$
$$\{\mathrm{w}_{\bar{y}}(x\{z := y\}) : x \in \Delta\} \underset{\Delta \subseteq \{x_1, \ldots, x_n\}}{=}$$
$$\{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta\} = \Delta'$$

□

Lema 6.31. Para todo $t, u \in \Lambda\mathrm{x}_{\mathrm{op}}$, $n \in \mathbb{N}$, $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, tenemos que $t =_\alpha u \implies \mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{x}_{1:n}}(u)$. Notar que $\mathrm{w}_{\bar{x}_{1:n}}(u)$ está bien definido, pues $t =_\alpha u \implies \mathrm{FV}(t) = \mathrm{FV}(u)$.

Demostración. Por inducción en t. Basta ver el caso en que $t = \mathrm{X}_\Delta$. Luego, $t =_\alpha u \underset{\text{def}}{\iff} t = u = \mathrm{X}_\Delta$. Trivialmente se cumple la propiedad. □

Extendemos ahora el lema 4.21, de manera tal de asegurar que la definición de la traducción uniforme w (ver 4.22) se mantenga válida.

Lema 6.32. Para todo $t \in \Lambda\mathrm{x}_{\mathrm{op}}$: $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, y para todo $\{y_1, \ldots, y_m\} \subset \mathbb{V}$, tenemos que $\mathrm{w}_{\bar{x}_{1:n}}(t) = \mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(t)$.

Demostración. Por inducción en t. Basta ver el caso en que $t = \mathrm{X}_\Delta$. Recordar nuevamente que $\Delta \subseteq \{x_1, \ldots, x_n\}$. Entonces,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta) \underset{\text{def}}{=} \mathrm{X}_{\Delta'}$$

$$\text{con } \Delta' = \{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta\}$$

Además,

$$\mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(\mathrm{X}_\Delta) \underset{\text{def}}{=} \mathrm{X}_{\Delta''}$$

$$\text{con } \Delta'' = \{\mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(x) : x \in \Delta\}$$

Basta mostrar que $\Delta' = \Delta''$. Ahora, como ya vimos que

$$\mathrm{w}_{\bar{x}_{1:n}}(x) = \mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(x) \qquad \forall x \in \{x_1, \ldots, x_n\}$$

tenemos que, como $\Delta \subseteq \{x_1, \ldots, x_n\}$,

$$\Delta' = \{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta\} = \{\mathrm{w}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(x) : x \in \Delta\} = \Delta''$$

□

De esta forma hemos mostrado que, efectivamente, la traducción w se mantiene bien definida para la extensión a términos abiertos (definición que no mostramos aquí, pues no difiere más que en el dominio y codominio).

Seguimos ahora con el trabajo análogo para la traducción u.

Definición 6.33 (Traducción desde $\Lambda\mathrm{re}_{\mathrm{op}}$ a $\Lambda\mathrm{x}_{\mathrm{op}}$)**.** Para todo $a \in \Lambda\mathrm{re}_{\mathrm{op}}$, $n \in \mathbb{N} : \mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, $\mathrm{u}_{[x_1,\ldots,x_n]} : \Lambda\mathrm{re}_{\mathrm{op}} \to \Lambda\mathrm{x}_{\mathrm{op}}$, con $\{x_1, \ldots, x_n\}$ variables distintas entre sí, se define inductivamente como:

$$\begin{array}{rcll}
\mathrm{u}_{\bar{x}_{1:n}}(j) & = & x_j & (j \in \mathbb{N}_{>0} : j \leq n) \\
\mathrm{u}_{\bar{x}_{1:n}}(a\, b) & = & \mathrm{u}_{\bar{x}_{1:n}}(a)\ \mathrm{u}_{\bar{x}_{1:n}}(b) & \\
\mathrm{u}_{\bar{x}_{1:n}}(\lambda a) & = & \lambda x.\mathrm{u}_{x:\bar{x}_{1:n}}(a) & (x \notin \{x_1, \ldots, x_n\}) \\
\mathrm{u}_{\bar{x}_{1:n}}(a[b]) & = & \mathrm{u}_{x:\bar{x}_{1:n}}(a)\,[x := \mathrm{u}_{\bar{x}_{1:n}}(b)] & (x \notin \{x_1, \ldots, x_n\}) \\
\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta) & = & \mathrm{X}_{\Delta'} & \text{con } \Delta' = \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta\}
\end{array}$$

Extendemos ahora la prueba del lema 4.25, que a su vez hace que el lema 4.26 se mantenga válido para metatérminos. Como consecuencia del último, tenemos que la extensión de la traducción $\mathrm{u}_{\bar{x}_{1:n}}$ – mostrada arriba – mantiene su carácter de función.

Lema 6.34. Para todo $a \in \Lambda\mathrm{re}_{\mathrm{op}}$, $n \in \mathbb{N}$, $\{x_1, \ldots, x_n\}$ variables distintas tal que $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, tenemos que $\forall y \notin \{x_1, \ldots, x_n\}$, $1 \leq k \leq n$: $\mathrm{u}_{\bar{x}_{1:n}}(a)\{x_k := y\} =_\alpha \mathrm{u}_{\bar{x}_{1:k-1} \bullet y \bullet \bar{x}_{k+1:n}}(a)$.

Demostración. Por inducción en a. Por claridad, llamamos

$$\bar{y} = \bar{x}_{1:k-1} \bullet y \bullet \bar{x}_{k+1:n}$$

Es decir, tenemos que ver que: $\mathrm{u}_{\bar{x}_{1:n}}(a)\{x_k := y\} =_\alpha \mathrm{u}_{\bar{y}}(a)$. Basta ver el caso en que $a = \mathrm{X}_\Delta$. Recordemos que, por definición, $\Delta \subseteq \{1, \ldots, n\}$. Ahora bien,

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta)\{x_k := y\} \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta'}\{x_k := y\} \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta'\{x_k := y\}}$$

$$\text{con } \Delta' = \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta\}$$

además,

$$\mathrm{u}_{\bar{y}}(\mathrm{X}_\Delta) \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta''}$$

$$\text{con } \Delta'' = \{\mathrm{u}_{\bar{y}}(j) : j \in \Delta\}$$

Para mostrar que $\mathrm{X}_{\Delta''} =_\alpha \mathrm{X}_{\Delta'\{x_k := y\}}$, basta - por definición - ver que $\Delta'' = \Delta'\{x_k := y\}$. Ahora bien, hemos visto ya que

$$\mathrm{u}_{\bar{y}}(j) = \mathrm{u}_{\bar{x}_{1:n}}(j)\{x_k := y\} \qquad \forall j \in \{1, \ldots, n\}$$

Por lo tanto, como $\Delta \subseteq \{1, \ldots, n\}$, tenemos que

$$\Delta'\{x_k := y\} = (\{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta\})\{x_k := y\} \underset{\mathrm{def}}{=}$$

$$\{\mathrm{u}_{\bar{x}_{1:n}}(j)\{x_k := y\} : j \in \Delta\} = \{\mathrm{u}_{\bar{y}}(j) : j \in \Delta\} = \Delta''$$

□

Para finalizar con la presentación de las traducciones sobre términos abiertos, mostramos la extensión del lema 4.27, que nos permite asegurar – de manera análoga a lo mostrado para w – que la definición de la traducción u uniforme se mantiene válida.

Destacamos también que, dado que la definición de la traducción uniforme para metatérminos no difiere más que en dominio y codominio, no la repetimos aquí.

Lema 6.35. Para todo $a \in \Lambda\mathrm{re}_{\mathrm{op}}$, $\{x_1, \ldots, x_n\}$ variables distintas tal que $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, y para todo $\{y_1, \ldots, y_m\}$ variables distintas tal que $\{x_1, \ldots, x_n\} \cap \{y_1, \ldots, y_m\} = \emptyset$, tenemos que $\mathrm{u}_{\bar{x}_{1:n}}(a) =_\alpha \mathrm{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(a)$

Demostración. Por inducción en a. Basta ver el caso en que $a = \mathrm{X}_\Delta$. Recordemos que, por definición, $\Delta \subseteq \{1, \ldots, n\}$. Tenemos que

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta) \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta'}$$

$$\text{con } \Delta' = \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta\}$$

Además,

$$\mathrm{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(\mathrm{X}_\Delta) \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta''}$$

$$\text{con } \Delta'' = \{\mathrm{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(j) : j \in \Delta\}$$

Por definición de α-equivalencia para metatérminos, es suficiente ver que $\Delta' = \Delta''$. Ahora, como ya mostramos que

$$\mathrm{u}_{\bar{x}_{1:n}}(j) = \mathrm{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(j) \qquad \forall j \in \{1, \ldots, n\}$$

tenemos que, como $\Delta \subseteq \{1, \ldots, n\}$,

$$\Delta' = \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta\} = \{\mathrm{u}_{\bar{x}_{1:n} \bullet \bar{y}_{1:m}}(j) : j \in \Delta\} = \Delta''$$

□

Probando, una vez más, el isomorfismo

Una vez extendidas las traducciones w y u, y probado que mantienen su condición de función con respecto al cociente $\Lambda\mathrm{x}_{\mathrm{op}} \,/\, =_\alpha$, las utilizamos ahora para mostrar que las extensiones a términos abiertos de λex y λrex son isomorfas. Para esto, recordemos por última vez en el trabajo que, de acuerdo con lo introducido en la sección 1.3.4, debemos probar:

1. $\mathrm{w} \circ \mathrm{u} = \mathrm{Id}_{\Lambda\mathrm{re}_{\mathrm{op}}} \wedge \mathrm{u} \circ \mathrm{w} = \mathrm{Id}_{\Lambda\mathrm{x}_{\mathrm{op}}}$
2. $\forall t, u \in \Lambda\mathrm{x}_{\mathrm{op}} : t \to_{\lambda\mathrm{ex}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{rex}} \mathrm{w}(u)$
3. $\forall a, b \in \Lambda\mathrm{re}_{\mathrm{op}} : a \to_{\lambda\mathrm{rex}} b \implies \mathrm{u}(a) \to_{\lambda\mathrm{ex}} \mathrm{u}(b)$

De manera análoga a lo realizado en la sección inmediatamente anterior, sólo mostramos los casos de metavariable para las pruebas que ya hemos dado a lo largo del desarrollo; y sólo hacemos una mención para aquellos lemas que ni siquiera es necesario extender.

Composición de las traducciones

Extendemos aquí las pruebas de los lemas 4.29, 4.30, 4.31 y 4.32, de manera tal de comprobar que las nuevas traducciones conforman una biyección.

Lema 6.36 (Variables libres del término $\mathrm{w}_{\bar{x}_{1:n}}(t)$)**.** Para todo $t \in \Lambda\mathrm{x}_{\mathrm{op}}$: $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, tenemos que $\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(t)) \subseteq \{1, \ldots, n\}$.

Demostración. Por inducción en t. Basta ver el caso en que $t = \mathrm{X}_\Delta$. Como ya vimos en el lema para $\Lambda\mathrm{x}$, tenemos que

$$\mathrm{w}_{\bar{x}_{1:n}}(x) \in \{1, \ldots, n\} \qquad \forall x \in \{x_1, \ldots, x_n\}$$

Luego, como $\Delta \subseteq \{x_1, \ldots, x_n\}$,

$$\mathrm{FV}(\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta)) \underset{\mathrm{def}}{=} \{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta\} \subseteq \{1, \ldots, n\}$$

□

Lema 6.37 (Variables libres del término $\mathrm{u}_{\bar{x}_{1:n}}(a)$)**.** Para todo $a \in \Lambda\mathrm{re}_{\mathrm{op}} : \mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, y para todo $\{x_1, \ldots, x_n\}$ conjunto de variables distintas, tenemos que $\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(a)) \subseteq \{x_1, \ldots, x_n\}$.

Demostración. Por inducción en a. Basta ver el caso en que $a = \mathrm{X}_\Delta$. Como ya vimos en el lema para $\Lambda\mathrm{re}$, tenemos que

$$\mathrm{u}_{\bar{x}_{1:n}}(j) \in \{x_1, \ldots, x_n\} \qquad \forall j \in \{1, \ldots, n\}$$

Luego, como $\Delta \subseteq \{1, \ldots, n\}$,

$$\mathrm{FV}(\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta)) \underset{\mathrm{def}}{=} \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta\} \subseteq \{x_1, \ldots, x_n\}$$

□

Lema 6.38 ($\mathrm{w} \circ \mathrm{u} = \mathrm{Id}_{\Lambda\mathrm{re}_{\mathrm{op}}}$)**.** Para todo $a \in \Lambda\mathrm{re}_{\mathrm{op}} : \mathrm{w}(\mathrm{u}(a)) = a$

Demostración. Por inducción en a. Basta ver el caso en que $a = \mathrm{X}_\Delta$. Recordando lo mostrado en el lema 4.31, y por lema 6.37, tenemos que, si $\mathrm{FV}(\mathrm{X}_\Delta) = \Delta \subseteq \{1, \ldots, n\}$, entonces:

$$\mathrm{w}(\mathrm{u}(\mathrm{X}_\Delta)) = \mathrm{w}(\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta)) = \mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta))$$

Ahora, como mostramos anteriormente,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(j)) = j \qquad \forall j \in \{1, \ldots, n\}$$

Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta)) \underset{\mathrm{def}}{=} \mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_{\Delta'}) \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta''}$$

$$\text{con } \Delta' = \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta\} \quad \text{y} \quad \Delta'' = \{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta'\}$$

Es decir, tenemos

$$\Delta'' = \{\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{u}_{\bar{x}_{1:n}}(j)) : j \in \Delta\} \underset{\Delta \subseteq \{1, \ldots, n\}}{=} \{j : j \in \Delta\} = \Delta$$

□

Lema 6.39 ($\mathrm{u} \circ \mathrm{w} = \mathrm{Id}_{\Lambda \mathrm{x}_{\mathrm{op}}}$)**.** Para todo $t \in \Lambda \mathrm{x}_{\mathrm{op}} : \mathrm{u}(\mathrm{w}(t)) =_\alpha t$

Demostración. Por inducción en t. Basta ver el caso en que $t = \mathrm{X}_\Delta$. Recordando lo mostrado en el lema 4.32, y por lema 6.36, tenemos que, si $\mathrm{FV}(\mathrm{X}_\Delta) = \Delta \subseteq \{x_1, \ldots, x_n\}$, entonces:

$$\mathrm{u}(\mathrm{w}(\mathrm{X}_\Delta)) = \mathrm{u}(\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta)) = \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta))$$

Ahora, como mostramos anteriormente,

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(x)) = x \qquad \forall x \in \{x_1, \ldots, x_n\}$$

Luego,

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta)) \underset{\text{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_{\Delta'}) \underset{\text{def}}{=} \mathrm{X}_{\Delta''}$$

$$\text{con } \Delta' = \{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta\} \quad \text{y} \quad \Delta'' = \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta'\}$$

Es decir, tenemos

$$\Delta'' = \{\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{w}_{\bar{x}_{1:n}}(x)) : x \in \Delta\} \underset{\Delta \subseteq \{x_1, \ldots, x_n\}}{=} \{x : x \in \Delta\} = \Delta$$

Y, además, por definición, $\mathrm{X}_A = \mathrm{X}_B \implies \mathrm{X}_A =_\alpha \mathrm{X}_B$. □

Hemos comprobado aquí que las traducciones extendidas a metatérminos continúan conformando una biyección. A continuación, damos las extensiones de las pruebas de preservación de la reducción sobre términos abiertos, en ambas direcciones.

Un paso de reducción en λex implica un paso de reducción en λrex sobre metatérminos

En esta sección extendemos aquellos lemas que nos permitirán mostrar que:

$$\forall t, u \in \Lambda \mathrm{x}_{\mathrm{op}} : t \to_{\lambda\mathrm{ex}} u \implies \mathrm{w}(t) \to_{\lambda\mathrm{rex}} \mathrm{w}(u)$$

Los lemas son: 4.34, 4.35, 5.13 y 5.15. Volvemos a destacar que sólo mostramos el caso de metavariable, pues el resto se mantiene igual.

Comenzamos con la extensión del lema 4.34.

Lema 6.40 (Interacción entre la traducción w y $\uparrow_i$)**.** Sea $t \in \Lambda \mathrm{x}_{\mathrm{op}}$: $\mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Sea $i \in \mathbb{N}_{>0} : i < n \wedge x_i \neq x_{i+1}$. Luego,

$$\uparrow_i(\mathrm{w}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(t)) = \mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(t)$$

Demostración. Por inducción en t. Basta ver el caso en que $t = \mathrm{X}_\Delta$. Recordemos que $\Delta \subseteq \{x_1, \ldots, x_n\}$. Tenemos:

$$\uparrow_i(\mathrm{w}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(\mathrm{X}_\Delta)) \underset{\text{def}}{=} \uparrow_i(\mathrm{X}_{\Delta'}) \underset{\text{def}}{=} \mathrm{X}_{\Delta''}$$

$$\text{con } \Delta' = \left\{\mathrm{w}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(x) : x \in \Delta\right\} \quad \text{y}$$

$$\Delta'' = \Delta'_{<i} \cup \Delta'_{>i+1} \cup (\Delta'_{=i} + 1) \cup \left(\Delta'_{=i+1} - 1\right)$$

Además,

$$\mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\mathrm{X}_\Delta) \underset{\text{def}}{=} \mathrm{X}_{\Delta'''}$$

$$\text{con } \Delta''' = \left\{ \mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(x) : x \in \Delta \right\}$$

Por otro lado, sabemos que

$$\updownarrow_i(\mathrm{w}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(x)) = \mathrm{w}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(x) \quad \forall x \in \{x_1,\ldots,x_n\}$$

Ahora bien, como $\Delta \subseteq \{x_1,\ldots,x_n\}$, tenemos que

$$\Delta''' = \left\{ \updownarrow_i(\mathrm{w}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(x)) : x \in \Delta \right\} =_{\text{def}}$$

$$\{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta \wedge \mathrm{w}_{\bar{x}_{1:n}}(x) < i\} \cup \{\mathrm{w}_{\bar{x}_{1:n}}(x) : x \in \Delta \wedge \mathrm{w}_{\bar{x}_{1:n}}(x) > i+1\} \cup$$
$$\{\mathrm{w}_{\bar{x}_{1:n}}(x)+1 : x \in \Delta \wedge \mathrm{w}_{\bar{x}_{1:n}}(x) = i\} \cup \{\mathrm{w}_{\bar{x}_{1:n}}(x)-1 : x \in \Delta \wedge \mathrm{w}_{\bar{x}_{1:n}}(x) = i+1\} =$$
$$\Delta'_{<i} \cup \Delta'_{>i+1} \cup (\Delta'_{=i}+1) \cup \left(\Delta'_{=i+1}-1\right) = \Delta''$$

□

Como segundo paso, extendemos el lema 4.35.

Lema 6.41 (Interacción entre la traducción w y $\uparrow_i$)**.** Sea $t \in \Lambda\mathrm{x}_{\mathrm{op}}$: $\mathrm{FV}(t) \subseteq \{x_1,\ldots,x_n\}$. Sea $m \in \mathbb{N} : m \leq n$. Sea, además, $x \notin \{x_1,\ldots,x_n\}$. Luego,

$$\mathrm{w}_{\bar{x}_{1:m} \bullet x \bullet \bar{x}_{m+1:n}}(t) = \uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(t))$$

Demostración. Por inducción en t. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m} \bullet x \bullet \bar{x}_{m+1:n}$. Entonces, tenemos que ver que:

$$\mathrm{w}_{\bar{z}}(t) = \uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(t))$$

Basta ver el caso en que $t = \mathrm{X}_\Delta$. Nuevamente, tenemos $\Delta \subseteq \{x_1,\ldots,x_n\}$. Luego,

$$\mathrm{w}_{\bar{z}}(\mathrm{X}_\Delta) =_{\text{def}} \mathrm{X}_{\Delta'}$$
$$\text{con } \Delta' = \{\mathrm{w}_{\bar{z}}(y) : y \in \Delta\}$$

Como $\Delta \subseteq \{x_1,\ldots,x_n\}$ y la propiedad vale para todo $y \in \{x_1,\ldots,x_n\}$, tenemos que:

$$\Delta' = \{\uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(y)) : y \in \Delta\} =_{\text{def}}$$
$$\{\mathrm{w}_{\bar{x}_{1:n}}(y) : y \in \Delta \wedge \mathrm{w}_{\bar{x}_{1:n}}(y) \leq m\} \cup \{\mathrm{w}_{\bar{x}_{1:n}}(y)+1 : y \in \Delta \wedge \mathrm{w}_{\bar{x}_{1:n}}(y) > m\} =_{\text{def}}$$
$$\Delta''_{\leq m} \cup (\Delta''_{>m}+1)$$
$$\text{con } \Delta'' = \{\mathrm{w}_{\bar{x}_{1:n}}(y) : y \in \Delta\}$$

Ahora bien,

$$\uparrow_m(\mathrm{w}_{\bar{x}_{1:n}}(t)) =_{\text{def}} \mathrm{X}_{\Delta'''}$$
$$\text{con } \Delta''' = \Delta''_{\leq m} \cup (\Delta''_{>m}+1) = \Delta'$$

□

A continuación, mostramos la extensión del lema 5.13.

Lema 6.42. Sea $t \in \Lambda\mathrm{x}_{\mathrm{op}} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Sea $m \in \mathbb{N} : 1 \leq m \leq n+1$. Luego, para todo $x \notin \{x_1, \ldots, x_n\}$, tenemos que $m \notin \mathrm{FV}\big(\mathrm{w}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(t)\big)$.

Demostración. Por inducción en t. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}$. Entonces, tenemos que ver que para todo $x \notin \{x_1, \ldots, x_n\}$, $m \notin \mathrm{FV}(\mathrm{w}_{\bar{z}}(t))$. Basta ver el caso en que $t = \mathrm{X}_\Delta$. Recordar, una vez más, que $\Delta \subseteq \{x_1, \ldots, x_n\}$. Tenemos:

$$\mathrm{FV}(\mathrm{w}_{\bar{z}}(\mathrm{X}_\Delta)) \underset{\text{def}}{=} \mathrm{FV}(\mathrm{X}_{\Delta'}) \underset{\text{def}}{=} \Delta'$$

$$\text{con } \Delta' = \{\mathrm{w}_{\bar{z}}(x) : x \in \Delta\}$$

Como sabemos que $\forall y \in \{x_1, \ldots, x_n\} : m \notin \mathrm{FV}(\mathrm{w}_{\bar{z}}(y))$ y que $\Delta \subseteq \{x_1, \ldots, x_n\}$, tenemos necesariamente que $m \notin \Delta'$. □

Hacemos extensiva la observación 5.14.

Observación 6.43. Como ya vimos en la mentada observación, es análogo al lema anterior mostrar que para todo $t \in \Lambda\mathrm{x}_{\mathrm{op}} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$, para todo $m \in \mathbb{N} : 1 \leq m \leq n+1$, y para todo $x \notin \{x_1, \ldots, x_{m-1}\} \wedge x \in \mathrm{FV}(t)$, se cumple que $m \in \mathrm{FV}\big(\mathrm{w}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(t)\big)$.

Por último, extendemos el lema 5.15.

Lema 6.44 (Interacción entre la traducción w y $\downarrow_i$)**.** Sea $t \in \Lambda\mathrm{x}_{\mathrm{op}} : \mathrm{FV}(t) \subseteq \{x_1, \ldots, x_n\}$. Sea $m \in \mathbb{N} : 1 \leq m \leq n+1$. Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(t) = \downarrow_m(\mathrm{w}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(t))$$

Demostración. Por inducción en t. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}$. Entonces, tenemos que ver que:

$$\mathrm{w}_{\bar{x}_{1:n}}(t) = \downarrow_m(\mathrm{w}_{\bar{z}}(t))$$

Basta ver el caso en que $t = \mathrm{X}_\Delta$. Tenemos que $\Delta \subseteq \{x_1, \ldots, x_n\}$, y que, por el lema 6.42, $m \notin \mathrm{FV}(\mathrm{w}_{\bar{z}_x}(\mathrm{X}_\Delta))$, con lo que el decremento de índices está bien definido. Luego,

$$\mathrm{w}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta) \underset{\text{def}}{=} \mathrm{X}_{\Delta'}$$

$$\text{con } \Delta' = \{\mathrm{w}_{\bar{x}_{1:n}}(y) : y \in \Delta\}$$

Como la propiedad vale para todo $y \in \{x_1, \ldots, x_n\}$, tenemos que:

$$\Delta' = \{\downarrow_m(\mathrm{w}_{\bar{z}}(y)) : y \in \Delta\} \underset{\text{def}}{=}$$

$$\{\mathrm{w}_{\bar{z}}(y) : y \in \Delta \wedge \mathrm{w}_{\bar{z}}(y) < m\} \cup \{\mathrm{w}_{\bar{z}}(y) - 1 : y \in \Delta \wedge \mathrm{w}_{\bar{z}}(y) > m\} =$$

$$\Delta''_{<m} \cup (\Delta''_{>m} - 1)$$

$$\text{con } \Delta'' = \{\mathrm{w}_{\bar{z}}(y) : y \in \Delta\}$$

Además,

$$\downarrow_m(\mathrm{w}_{\bar{z}}(\mathrm{X}_\Delta)) \underset{\text{def}}{=} \downarrow_m(\mathrm{X}_{\Delta''}) \underset{\text{def}}{=} \mathrm{X}_{\Delta'''}$$

$$\text{con } \Delta''' = \Delta''_{<m} \cup (\Delta''_{>m} - 1) = \Delta'$$

□

Finalizadas las extensiones de los lemas necesarios, podemos afirmar que la propiedad de preservación de la reducción λex bajo la traducción w se mantiene para términos abiertos. Esta observación proviene del hecho de analizar que, en los lemas 6.10, 6.11 y 6.12, no es necesario agregar casos para metavariable, y que los casos ya mostrados siguen valiendo al cumplirse los lemas anteriores sobre metatérminos. En verdad, sí se hace necesario agregar el caso para metavariable en el lema 6.11, aunque vemos rápidamente que la propiedad se cumple vacuamente, pues no hay reducción posible para el término X_Δ.

Un paso de reducción en λrex implica un paso de reducción en λex sobre metatérminos

En esta última sección realizamos la extensión de aquellos lemas que nos servirán para mostrar que la preservación de la reducción λrex bajo la traducción u se mantiene para metatérminos. De esta manera, estaremos probando la última de las condiciones necesarias del isomorfismo; y, así, mostrando que efectivamente las extensiones a términos abiertos de los cálculos λex y λrex son isomorfas. Por ende, y basándonos en [Kes09], habremos probado que λrex admite una extensión a términos abiertos que es confluente sin perder el resto de las propiedades.

Extendemos, entonces, las pruebas para los lemas 4.37, 4.38, 5.17 y 5.19, mostrando, al igual que durante todo este último tramo del trabajo, sólo el caso de metavariable.

Comenzamos con el lema 4.37.

Lema 6.45 (Interacción entre la traducción u y $\uparrow_i$)**.** Sea $a \in \Lambda\mathrm{re}_{\mathrm{op}}$: $\mathrm{FV}(a) \subseteq \{1, \ldots, n\}$. Sea $i \in \mathbb{N}_{>0} : i < n$. Sean, además, $\{x_1, \ldots, x_n\}$ variables distintas. Luego,

$$\mathrm{u}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(a) =_\alpha \mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\uparrow_i(a))$$

Demostración. Por inducción en a. Basta ver el caso en que $a = \mathrm{X}_\Delta$. Por definición, $\Delta \subseteq \{1, \ldots, n\}$. Luego,

$$\mathrm{u}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(\mathrm{X}_\Delta) \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta'}$$

$$\text{con } \Delta' = \left\{\mathrm{u}_{[x_1,\ldots,x_i,x_{i+1},\ldots,x_n]}(j) : j \in \Delta\right\}$$

Además,

$$\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\uparrow_i(\mathrm{X}_\Delta)) \underset{\mathrm{def}}{=} \mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\mathrm{X}_{\Delta''}) = \mathrm{X}_{\Delta'''}$$

$$\text{con } \Delta'' = \Delta_{<i} \cup \Delta_{>i+1} \cup (\Delta_{=i} + 1) \cup (\Delta_{=i+1} - 1) \quad \text{y}$$
$$\Delta''' = \left\{\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(j) : j \in \Delta''\right\}$$

Ahora bien, como vale la propiedad para todo $j \in \{1, \ldots, n\}$, y $\Delta \subseteq \{1, \ldots, n\}$, tenemos que

$$\Delta' = \left\{\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(\uparrow_i(j)) : j \in \Delta\right\} \underset{\mathrm{def}}{=}$$

$$\left\{\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(j) : j \in \Delta \wedge j < i\right\} \cup$$

$$\left\{\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(j) : j \in \Delta \wedge j > i+1\right\} \cup$$

$$\left\{\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(j+1) : j \in \Delta \wedge j = i\right\} \cup$$

$$\left\{\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(j-1) : j \in \Delta \wedge j = i+1\right\} =_{\text{def}}$$

$$\left\{\mathrm{u}_{[x_1,\ldots,x_{i+1},x_i,\ldots,x_n]}(j) : j \in \Delta''\right\} = \Delta'''$$

Por definición de α-equivalencia, $\mathrm{X}_{\Delta'} = \mathrm{X}_{\Delta'''} \implies \mathrm{X}_{\Delta'} =_\alpha \mathrm{X}_{\Delta'''}$. □

Seguimos con el lema 4.38.

Lema 6.46 (Interacción entre la traducción u y $\uparrow_i$)**.** Sea $a \in \Lambda\mathrm{re}_{\mathrm{op}}$: $\mathrm{FV}(a) \subseteq \{1,\ldots,n\}$. Sean $\{x_1,\ldots,x_n\}$ variables distintas, y sea $m \in \mathbb{N} : m \leq n$. Sea, además, $x \notin \{x_1,\ldots,x_n\}$. Luego,

$$\mathrm{u}_{\bar{x}_{1:m} \bullet x \bullet \bar{x}_{m+1:n}}(\uparrow_m(a)) =_\alpha \mathrm{u}_{\bar{x}_{1:n}}(a)$$

Demostración. Por inducción en a. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m} \bullet x \bullet \bar{x}_{m+1:n}$. Entonces, tenemos que ver que:

$$\mathrm{u}_{\bar{z}}(\uparrow_m(a)) =_\alpha \mathrm{u}_{\bar{x}_{1:n}}(a)$$

Basta ver el caso en que $a = \mathrm{X}_\Delta$. Recordar que $\Delta \subseteq \{1,\ldots,n\}$. Ahora,

$$\mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_\Delta) =_{\text{def}} \mathrm{X}_{\Delta'}$$

$$\text{con } \Delta' = \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta\}$$

Como $\Delta \subseteq \{1,\ldots,n\}$ y la propiedad vale para todo $j \in \{1,\ldots,n\}$, tenemos que:

$$\Delta' = \{\mathrm{u}_{\bar{z}}(\uparrow_m(j)) : j \in \Delta\} =_{\text{def}}$$

$$\{\mathrm{u}_{\bar{z}}(j) : j \in \Delta \wedge j \leq m\} \cup \{\mathrm{u}_{\bar{z}}(j+1) : j \in \Delta \wedge j > m\} =_{\text{def}} \{\mathrm{u}_{\bar{z}}(j) : j \in \Delta''\}$$

$$\text{con } \Delta'' = \Delta_{\leq m} \cup (\Delta_{>m} + 1)$$

Ahora bien,

$$\mathrm{u}_{\bar{z}}(\uparrow_m(\mathrm{X}_\Delta)) =_{\text{def}} \mathrm{u}_{\bar{z}}(\mathrm{X}_{\Delta''}) =_{\text{def}} \mathrm{X}_{\Delta'}$$

Por definición de α-equivalencia, $\mathrm{X}_{\Delta'} =_\alpha \mathrm{X}_{\Delta'}$. □

Continuamos con el anteúltimo de los lemas, extensión del 5.17.

Lema 6.47. Sea $a \in \Lambda\mathrm{re}_{\mathrm{op}}$: $\mathrm{FV}(a) \subseteq \{1,\ldots,n\}$. Sean $\{x_1,\ldots,x_n\}$ variables distintas, y sea $m \in \mathbb{N} : 1 \leq m \leq n+1$. Luego, si $m \notin \mathrm{FV}(a)$, tenemos que para todo $x \notin \{x_1,\ldots,x_n\}$, $x \notin \mathrm{FV}\big(\mathrm{u}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(a)\big)$.

Demostración. Por inducción en a. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}$. Entonces, tenemos que ver que para todo $x \notin \{x_1,\ldots,x_n\}$, $m \notin \mathrm{FV}(a) \implies x \notin \mathrm{FV}(\mathrm{u}_{\bar{z}}(a))$. Basta ver el caso en que $a = \mathrm{X}_\Delta$, recordando también que $\Delta \subseteq \{1,\ldots,n\}$. Sea $x \notin \{x_1,\ldots,x_n\}$. Tenemos:

$$\mathrm{FV}(\mathrm{u}_{\bar{z}}(\mathrm{X}_\Delta)) =_{\text{def}} \mathrm{FV}(\mathrm{X}_{\Delta'}) =_{\text{def}} \Delta'$$

$$\text{con } \Delta' = \{\mathrm{u}_{\bar{z}}(j) : j \in \Delta\}$$

Como sabemos que $\forall j \in \{1,\ldots,n\}, j \neq m : x \notin \mathrm{FV}(\mathrm{u}_{\bar{z}}(j))$ y que $\Delta \subseteq \{1,\ldots,n\} \wedge m \notin \Delta$, tenemos necesariamente que $x \notin \Delta'$. □

Antes de pasar al último de los lemas, extendemos también la observación 5.18.

Observación 6.48. De manera análoga al lema 6.47, podemos probar que para todo $a \in \Lambda\mathrm{re}_{\mathrm{op}} : \mathrm{FV}(a) \subseteq \{1, \ldots, n\}$, todo $\{x_1, \ldots, x_n\}$ conjunto de variables distintas, y todo $m \in \mathbb{N} : 1 \leq m \leq n+1$, ocurre que si $m \in \mathrm{FV}(a)$, tenemos que para toda variable $x \notin \{x_1, \ldots, x_n\}$, $x \in \mathrm{FV}\big(\mathrm{u}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(a)\big)$.

Dada esta observación, terminamos las extensiones con el último de los lemas, prolongación del 5.19.

Lema 6.49 (Interacción entre la traducción u y $\downarrow_i$)**.** Sea $a \in \Lambda\mathrm{re}_{\mathrm{op}} : \mathrm{FV}(a) \subseteq \{1, \ldots, n\}$. Sean $\{x_1, \ldots, x_n\}$ variables distintas, y sea $m \in \mathbb{N} : 1 \leq m \leq n+1$. Sea, además, $x \notin \{x_1, \ldots, x_n\}$. Luego, si $m \notin \mathrm{FV}(a)$, tenemos que

$$\mathrm{u}_{\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}}(a) =_\alpha \mathrm{u}_{\bar{x}_{1:n}}(\downarrow_m(a))$$

Demostración. Por inducción en a. Para mayor comodidad en la notación, llamamos $\bar{z}$ a $\bar{x}_{1:m-1} \bullet x \bullet \bar{x}_{m:n}$. Entonces, tenemos que ver que:

$$\mathrm{u}_{\bar{z}}(a) =_\alpha \mathrm{u}_{\bar{x}_{1:n}}(\downarrow_m(a))$$

Basta ver el caso en que $a = \mathrm{X}_\Delta$, recordando por última vez que $\Delta \subseteq \{1, \ldots, n\}$. A su vez, como $m \notin \mathrm{FV}(a)$, $\downarrow_m(a)$ está bien definido. Sea $x \notin \{x_1, \ldots, x_n\}$. Tenemos, entonces:

$$\mathrm{u}_{\bar{z}}(\mathrm{X}_\Delta) \underset{\mathrm{def}}{=} \mathrm{X}_{\Delta'}$$

$$\text{con } \Delta' = \{\mathrm{u}_{\bar{z}}(j) : j \in \Delta\}$$

Como la propiedad vale para todo $j \in \{1, \ldots, n\}$, $j \neq m$, y sabemos que $m \notin \Delta$, tenemos que:

$$\Delta' = \{\mathrm{u}_{\bar{x}_{1:n}}(\downarrow_m(j)) : j \in \Delta\} \underset{\mathrm{def}}{=}$$

$$\{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta \wedge j < m\} \cup \{\mathrm{u}_{\bar{x}_{1:n}}(j) - 1 : j \in \Delta \wedge j > m\} =$$

$$\{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta''\}$$

$$\text{con } \Delta'' = \Delta_{<m} \cup (\Delta_{>m} - 1)$$

Además,

$$\mathrm{u}_{\bar{x}_{1:n}}(\downarrow_m(\mathrm{X}_\Delta)) \underset{\mathrm{def}}{=} \mathrm{u}_{\bar{x}_{1:n}}(\mathrm{X}_{\Delta''}) \underset{\mathrm{def}}{=} \{\mathrm{u}_{\bar{x}_{1:n}}(j) : j \in \Delta''\} = \Delta'$$

Por definición de α-equivalencia, $\mathrm{X}_{\Delta'} =_\alpha \mathrm{X}_{\Delta'}$. □

Terminadas las extensiones a los lemas, podemos afirmar – de manera análoga a lo hecho para la traducción w sobre términos abiertos –, que la preservación de la reducción λrex bajo la traducción u se mantiene en metatérminos. Esto, al igual que en la sección anterior, proviene del hecho de observar que, en los lemas 6.13, 6.14 y 6.15, no es necesario agregar el caso de metavariable; y que con sólo ser válidos los lemas que acabamos de probar, la propiedad se mantiene (salvo, claro, para el lema 6.14, en el que se podría agregar el caso de metavariable – caso que se cumple vacuamente por no haber reducción posible).

De esta manera, terminamos con las pruebas del isomorfismo entre las extensiones a términos abiertos de λex y λrex enunciando y probando el teorema principal.

Teorema 6.50 (Isomorfismo entre las extensiones a términos abiertos de λex y λrex)**.** Las extensiones a términos abiertos de los cálculos λex y λrex son isomorfas.

Demostración. Consecuencia directa de la definición de isomorfismo, de las traducciones expuestas (w y u), y de los lemas 4.31, 6.38, 4.32, 6.39, 6.12 y 6.15, junto con las extensiones de los lemas que los hacen seguir siendo válidos para metatérminos. □

Corolario 6.51. Las extensiones a términos abiertos de los cálculos λrex y λex tienen las mismas propiedades.

Demostración. Consecuencia directa del teorema 6.50 □

Como reza el corolario 6.51, los cálculos $(\Lambda x_{op}, \lambda ex)$ y $(\Lambda re_{op}, \lambda rex)$ poseen las mismas propiedades. Por ende, ahora sí, podemos afirmar que logramos obtener: (Sim), (Snd), (CR_{rex}), (SN_{rex}), (CR), (MC) y (PSN) simultáneamente.

Conclusiones

Durante el curso de esta tesis hemos propuesto, desarrollado y estudiado tres nuevos cálculos de sustituciones explícitas, basados todos en una presentación alternativa del ya clásico λ_{dB} [dB72].

Creemos que el trabajo, cuyo último logro fue la obtención de λrex, formalismo isomorfo a λex [Kes08, Kes09], representa un gran avance en lo que hace al área de sustituciones explícitas con índices *à la de Bruijn*. Dicha afirmación encuentra justificativo en tres puntos principales, a saber:

- La respuesta a la largamente formulada pregunta acerca de la existencia de un cálculo con índices que pudiera conseguir al mismo tiempo simulación de la β-reducción en un paso, soundness, confluencia, preservación de la normalización fuerte y metaconfluencia, sin requerir, para esto, de la introducción de complicadas construcciones sintácticas que poca naturalidad guardan con respecto al λ cálculo clásico.

- La invención de un medio de traducción sencillo e intuitivo entre cálculos con nombres y cálculos con índices, que simplifica el trazado de puentes directos para futuros desarrollos en cualquiera de las dos notaciones.

- El asentamiento de una base teórica a partir de la cual pueden buscarse adaptaciones de otros cálculos de sustituciones explícitas con índices à la de Bruijn, con el fin de conseguir formalismos que posean todas las buenas propiedades esperables, y que sean a su vez viables desde el punto de vista práctico/implementativo. Tal puede ser el caso, por ejemplo, de un cálculo al estilo de $\lambda\sigma$ [ACCL91].

En parcial detrimento de lo antedicho, es importante mencionar que, tal y como sucede para λex, la necesidad de trabajar módulo D-equivalencia en el cálculo λrex complica las cosas a la hora de concebir posibles implementaciones. A su vez, el uso de sustituciones unarias es una clara limitación desde el punto de vista de la *performance*, aunque quizás sea un paso intermedio indispensable para lograr un acercamiento a buenas soluciones desde esta óptica.

Como balance de lo recién expuesto creemos que, si bien no hemos conseguido un cálculo con absolutamente todas las bondades esperables, especialmente desde el punto de vista práctico (*i.e.,* la necesidad de trabajar módulo D-equivalencia es una clara barrera), sí hemos abierto un abanico de posibilidades de investigación en el ámbito de las sustituciones explícitas con índices à la de Bruijn que – creemos – se separa de manera relativamente radical de lo ya existente y estudiado a la fecha. Siguiendo con esta línea de razonamiento, pensamos que dicha apertura representa el camino a seguir para la concepción de

nuevos formalismos indexados que consigan lo que, desde hace 20 años, buscan incansablemente quienes trabajan en el área.

Trabajo futuro

Como cierre del presente trabajo, damos aquí una lista tentativa de lo que, a nuestro criterio, podrían ser ramas de estudio en un futuro no muy distante.

En primer lugar, y dado que la principal crítica a λex es la necesidad de trabajar módulo C-equivalencia – crítica, debido al isomorfismo existente, trasladable directamente a λrex –, creemos importante el estudio de alternativas que permitan eliminar la necesidad de lidiar con relaciones de equivalencia entre términos. Para esto, vemos – a priori – dos caminos posibles:

- Intentar eliminar la ecuación C del cálculo λrex, dejando sólo la regla de composición, pero sin condicionarla a sustituciones dependientes; esto, de manera tal de conservar las bondades de la ecuación y, en consiguiente, mantener MC. Ahora bien: para lograr esto sin perder a la vez PSN, habría que cambiar la condición de la regla de composición por una que sólo permita derivar si una composición es "mayor" o "menor" que otra (*i.e.,* sólo se puede ir, pero no volver). Dicho orden podría obtenerse mediante el establecimiento de un orden total entre los términos del cálculo. Si bien no tenemos una propuesta concreta, pensamos que podría ser un camino viable.

- Concebir un cálculo con sustituciones explícitas n-arias derivado de λrex, que permita cambiar la ecuación C por una regla de composición al estilo de $\lambda\sigma$. Puntualmente, vemos dos posibilidades para atacar la problemática. La primera tendría, como punto de partida, el agregado de sustituciones n-arias a λrex, basándose principalmente en una noción extendida de *swapping* para la implementación de la regla (Lamb) en dichas sustituciones. La segunda sería la contrapartida de la primera. Esto es, tomar un formalismo con buenas propiedades y sustituciones n-arias, tal como $\lambda\sigma_{\Uparrow}$ [CHL96], para luego modificarlo agregándole nuestra noción de *swap*. Creemos, sin embargo, que la primera de las opciones es la más viable, puesto que podría realizarse un seguimiento de las propiedades a medida que va mutando el cálculo, de forma tal de verificar la preservación de éstas.

Surge, en segundo lugar, la cuestión acerca de la explicitación de los meta-operadores $\updownarrow_i$, $\uparrow_i$ y $\downarrow_i$. Creemos innecesario realizar este trabajo, puesto que parte del mérito de λrex radica en haber quitado la actualización de índices del lenguaje, mostrando su carácter "accesorio". Esto es: representan más un medio que un fin en sí mismos. Más aún, al realizar un esbozo de la cantidad de reglas que quedaría en un cálculo con todos los operadores explícitos, llegamos a un número que ronda las 30. Con esta cifra en mente, se vuelven extremadamente tediosas las pruebas por inducción en pasos de reducción, análisis de pares críticos, etc.

Por último, y como otras alternativas interesantes, podemos mencionar:

- A nivel conceptual, estudiar la relación existente – si la hubiera – entre la técnica de *swapping* (utilizada como fundación de los cálculos presentados aquí) y *nominal logic* o *nominal rewriting* ([Pit03] y [FG07], entre otros),

de manera de arribar a una mejor comprensión de la lógica subyacente a dichas tecnologías.

- Analizar la factibilidad de implementar λrex sobre asistentes de pruebas (como puede ser, por ejemplo, Coq[1]), unificación de alto orden [DHK00], etc.
- Agregar una regla η para el cálculo λrex, y estudiar su comportamiento. A priori, este trabajo parecería ser sencillo de plantear, dado que las principales herramientas para la implementación de una regla η ya están dadas en λrex (puntualmente, el operador de decremento de índices $\downarrow_i$).
- Estudiar un cálculo que fusione los conceptos subyacentes a λex [Kes08, Kes09], λr y $\lambda_\emptyset$ [Rev85], de manera tal de conseguir un formalismo sin sustituciones que posea buenas propiedades. En esta dirección, lo más interesante a priori sería analizar una extensión de $\lambda_{\emptyset \mathrm{dB}}$ [Arb05].

De todo lo antedicho, pensamos que la opción más interesante y prometedora para plantear y estudiar en el corto y mediano plazo es una extensión de λrex a un cálculo con sustituciones n-arias (teniendo en mente la eliminación de la ecuación D como objetivo principal). A continuación damos algunas de las ideas que estuvimos barajando para esto.

Ideas para extender λrex a un cálculo con sustituciones n-arias

La característica distintiva de los cálculos λr (capítulo 3), λre (capítulo 4), $\lambda\mathrm{re}_{\mathrm{gc}}$ (capítulo 5) y λrex (capítulo 6) es la de eliminar el uso de índices dentro de la sustitución, reemplazándolos por un artilugio relativamente novedoso en lo que a índices de *de Bruijn* respecta: la operación de *swap*. Esta operación se hace necesaria al momento de atravesar una abstracción con una sustitución, dado que el *binding* de las variables cambia. Hicimos mención recién a la posibilidad de extender λrex a un cálculo del estilo de $\lambda\sigma_{\Uparrow}$, formalismo que posee MC pero no PSN. Nuestra idea es hacer justamente esto; analicemos, por lo tanto, la forma que tiene la regla (Lamb) en $\lambda\sigma_{\Uparrow}$ (remitimos al lector a [CHL96] para más información sobre el cálculo):

$$\text{(Lamb)} \qquad (\lambda a)[s] \to \lambda a[\Uparrow(s)]$$

siendo la semántica de $\Uparrow(s)$ intuitivamente $1 \cdot (s \circ \uparrow)$. Es decir, se "aumentan" los "índices" de la sustitución, y a la vez se incrementan las variables libres de los términos de s en 1. Esto es muy similar a lo que ocurre en un cálculo con sustituciones unarias al estilo de λs [KR95], sólo que trabajando con sustituciones n-arias. Visto esto, nuestra propuesta es la de eliminar la necesidad de "aumentar los índices de la sustitución", acercándonos así a la idea detrás de los cálculos planteados en este trabajo. ¿Cómo implementar esto? La respuesta debería ser ya evidente: ¡haciendo *swaps* sobre a! Es decir, reemplazar la regla por una del estilo:

$$\text{(Lamb)} \qquad (\lambda a)[s] \to \lambda \Updownarrow(a)[\Uparrow(s)]$$

en donde la semántica de $\Updownarrow(a)$ sea la de *swapear* los índices de a para reflejar el nuevo *binding*; y la de $\Uparrow(s)$ pase a ser $s \circ \uparrow$. De esta manera, estaríamos – a

[1] `http://coq.inria.fr`

priori – implementando la misma idea subyacente a nuestros cálculos, sólo que sobre sustituciones n-arias.

Dado que estas ideas están todavía en su etapa preliminar, restan todavía muchos problemas que resolver antes de poder plantear un cálculo completo. Enumeramos los que creemos más relevantes:

1. Definir una parametrización sencilla (y correcta) para $\Updownarrow(a)$, dado que diferentes sustituciones deberían generar *swaps* diferentes, dependiendo de los términos afectados.

2. Idear reglas de composición que se comporten correctamente de acuerdo a la semántica de los operadores.

3. Buscar una regla de *Garbage Collection* análoga a la existente en $\lambda\mathrm{re}_{\mathrm{gc}}$ y λrex.

4. Analizar la factibilidad y conveniencia de plantear un cálculo inicial con algunos de los operadores ubicados en el metalenguaje, de forma tal de analizar un formalismo intermedio que resulte más sencillo de trabajar.

Recalcamos nuevamente el carácter preliminar de las ideas expuestas aquí. Sin embargo, estamos actualmente trabajando en su refinación, con el objetivo de arribar a un cálculo concreto que podamos estudiar formalmente; y con la esperanza de conseguir un cálculo de sustituciones explícitas que posea *absolutamente todas* las largamente añoradas propiedades.

Bibliografía

[ACCL91] Martín Abadi, Luca Cardelli, Pierre-L. Curien, and Jean-J. Levy. Explicit substitutions. *Journal of Functional Programming*, 1:31–46, 1991.

[Arb05] Ariel Arbiser. *Explicit Substitution Systems and Subsystems*. PhD thesis, Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, 2005.

[BAG+92] Henk Barendregt, S. Abramsky, D. M. Gabbay, T. S. E. Maibaum, and H. P. Barendregt. Lambda calculi with types. In *Handbook of Logic in Computer Science*, pages 117–309. Oxford University Press, 1992.

[Bar84] Hendrik P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North Holland, Amsterdam, 1984.

[BG99] Roel Bloo and Herman Geuvers. Explicit substitution on the edge of strong normalization. *Theor. Comput. Sci.*, 211(1-2):375–395, 1999.

[Blo95] Roel Bloo. Preservation of strong normalisation for explicit substitution. Technical report, 1995.

[Blo97] Roel Bloo. *Preservation of Termination for Explicit Substitution*. PhD thesis, Eindhoven University, 1997.

[BN98] Franz Baader and Tobias Nipkow. *Term rewriting and all that*. Cambridge University Press, New York, NY, USA, 1998.

[BR95] Roel Bloo and Kristoffer H. Rose. Preservation of strong normalisation in named lambda calculi with explicit substitution and garbage collection. In *CSN-95: Computing Science in the Netherlands*, pages 62–72, 1995.

[CHL96] Pierre-L. Curien, Thérèse Hardin, and Jean-J. Lévy. Confluence properties of weak and strong calculi of explicit substitutions. *Journal of the ACM*, 43(2):362–397, 1996.

[dB72] Nicolaas G. de Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the church-rosser theorem. *Indagationes Mathematicae*, 34:381–392, 1972.

[dB78] Nicolaas G. de Bruijn. A namefree λ calculus with facilities for internal definition of expressions and segments. *Technical Report TH-Report 78-WSK-03, Department of Mathematics, Technical University of Eindhoven*, 1978.

[DG01] René David and Bruno Guillaume. A λ-calculus with explicit weakening and explicit substitution. *Mathematical. Structures in Comp. Sci.*, 11(1):169–206, 2001.

[DHK00] Gilles Dowek, Thérèse Hardin, and Claude Kirchner. Higher order unification via explicit substitutions. *Inf. Comput.*, 157(1-2):183–235, 2000.

[eaBBL$^+$95] Zine el-abidine Benaissa, Daniel Briaud, Pierre Lescanne, Jocelyne Rouyer-Degli, and Projet Eureca. $\lambda\upsilon$, a calculus of explicit substitutions which preserves strong normalisation. 1995.

[FG07] Maribel Fernández and Murdoch J. Gabbay. Nominal rewriting. *Inf. Comput.*, 205(6):917–965, 2007.

[Gui00] Bruno Guillaume. The λs_e-calculus does not preserve strong normalisation. *Journal of Functional Programming*, 10(04):321–325, 2000.

[Kes07] Delia Kesner. *Lecture Notes in Computer Science*, chapter The theory of calculi with explicit substitutions revisited, pages 238–252. Springer Berlin / Heidelberg, 2007.

[Kes08] Delia Kesner. Perpetuality for full and safe composition (in a constructive setting). In *ICALP '08: Proceedings of the 35th international colloquium on Automata, Languages and Programming, Part II*, pages 311–322, Berlin, Heidelberg, 2008. Springer-Verlag.

[Kes09] Delia Kesner. A theory of explicit substitutions with safe and full composition. *Logical Methods in Computer Science*, 5(3:1):1–29, 2009.

[KR95] Fairouz Kamareddine and Alejandro Ríos. A lambda-calculus à la de bruijn with explicit substitutions. In *PLILP '95: Proceedings of the 7th International Symposium on Programming Languages: Implementations, Logics and Programs, Lecture Notes in Computer Science 982*, pages 45–62, 1995.

[KR97] Fairouz Kamareddine and Alejandro Ríos. Extending a λ-calculus with explicit substitution which preserves strong normalisation into a confluent calculus on open terms. *Journal of Functional Programming*, 7(4):395–420, 1997.

[KR98] Fairouz Kamareddine and Alejandro Ríos. Bridging de bruijn indices and variable names in explicit substitutions calculi. *Logic Journal of the IGPL*, 6(6):843–874, 1998.

[Les94] Pierre Lescanne. De bruijn's $\lambda\xi\phi$, an early calculus of explicit substitutions. 1994.

[LRD95] Pierre Lescanne and Jocelyne Rouyer-Degli. Explicit substitutions with de bruijn's levels. In *RTA '95: Proceedings of the 6th International Conference on Rewriting Techniques and Applications*, pages 294–308, London, UK, 1995. Springer-Verlag.

[LV02] Carlos Lombardi and Enrique Vetere. Estudio de relaciones de reducción en el cálculo-λ puro. Tesis de Licenciatura en Ciencias de la Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, 2002.

[Mel95] Paul-André Melliès. Typed lambda-calculi with explicit substitutions may not terminate. In *TLCA '95: Proceedings of the Second International Conference on Typed Lambda Calculi and Applications, Lecture Notes in Computer Science 902*, pages 328–334, 1995.

[Mn96] César A. Muñoz. Confluence and preservation of strong normalisation in an explicit substitutions calculus. In *LICS '96: Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science*, page 440, Washington, DC, USA, 1996. IEEE Computer Society.

[Pit03] Andrew M. Pitts. Nominal logic, a first order theory of names and binding. *Inf. Comput.*, 186(2):165–193, 2003.

[RÓ3] Alejandro Ríos. *Contributions à l'étude des Lambda-calculus avec Substitutions Explicites.* PhD thesis, Université Paris 7, 1993.

[Rev85] Gyorgy Revesz. Axioms for the theory of lambda-conversion. *SIAM Journal on Computing*, 14(2):373–382, 1985.

[Rit99] Eike Ritter. Characterising explicit substitutions which preserve termination. In *TLCA*, pages 325–339, 1999.

[Ter03] Terese. *Term Rewriting Systems.* Cambridge University Press, 2003.

[VARK07] Daniel L. Ventura, Mauricio Ayala-Rincón, and Fairouz Kamareddine. Explicit substitutions calculi with explicit eta rules. 2007.

[Zil09] Beta Ziliani. $\lambda\sigma_{\mathrm{gc}}$: Un cálculo basado en $\lambda\sigma$ con garbage collection. Tesis de Licenciatura en Ciencias de la Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, 2009.