TRATAMIENTO DE TERMINOS QUE REPRESENTAN EL MISMO CONCEPTO EN EL CALCULO-λ

Carlos Lombardi y Enrique Vetere
(Universidad de Buenos Aires)

Abstract

In the classical λ-calculus, the α-*equivalence* relation is defined to identify the λ-terms that differ only in the names of its bound variable, which have therefore the same meaning.

The processes involved in computation are formally described in theλ-calculus by *reduction relations*, defined over the set of terms; and the most important among these relations is the β-reduction that describe the application of a function to an argument. We should be able to carry the reduction relations over the quotient set defined by the α-equivalence; in order to do so in a sound way, it is necessary to verify that the reduction results does not depend on the representatives of the involved α-equivalence classes chosen to operate.

This article focuses on the sound definition of reduction relations over α-equivalence classes, introducing a sufficient set of conditions that guarantee that a reduction relation defined over terms generate a sound definition on α-equivalence classes.

Resumen

El cálculo-λ es una teoría que enfatiza los aspectos computacionales de las funciones. Los términos del cálculo-λ se construyen a partir de variables, un constructor para denotar la aplicación de una función a un argumento y un símbolo ligador. Este símbolo ligador, que se utiliza para denotar funciones, indica la variable (llamada por esta razón "variable ligada") que será reemplazada en el cuerpo de la función al aplicarla a un argumento. Los términos que sólo difieren en sus variables ligadas representan el mismo concepto, y por lo tanto son identificados mediante una relación de equivalencia, llamada α-equivalencia.

Dentro del cálculo-λ se definen relaciones de reducción, que modelan los conceptos relacionados con el cómputo; siendo la más importante la β-reducción que describe formalmente la aplicación de una función a su argumento. Estas relaciones se definen sobre el conjunto de términos, siendo deseable exenderlas al conjunto cociente por la relación de α-equivalencia. Para ello, se requiere verificar que la definición no depende del representante elegido para operar, es decir, que el concepto de reducción definido no es alterado cuando los términos elegidos para operar son reemplazados por otros α-equivalentes. Este trabajo se centra en la correcta definición de relaciones de reducción sobre el conjunto de clases de α-equivalencia, proveyendo un conjunto de condiciones suficientes para garantizar que las relaciones de reducción definidas sobre términos definen relaciones de reducción correctas sobre el conjunto cociente.

Los resultados y las ideas que aquí se exponen surgieron durante el desarrollo, bajo la dirección de Alejandro Ríos, de nuestra tesis de licenciatura en Ciencias de la Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires.

Introducción

El cálculo-λ es una teoría acerca de las funciones. A diferencia de la formalización conjuntista, donde las funciones son modeladas como conjuntos de pares, el cálculo-λ enfatiza los aspectos computacionales, formalizando a las funciones como *reglas* que describen cómo obtener el resultado a partir de los argumentos.

Ciertas cualidades del cálculo-λ lo configuran como una herramienta adecuada para varios propósitos. Esquemáticamente:

- Con el cálculo-λ se pueden representar las funciones computables, esto es, las funciones para las cuales existe un programa que las calcula.

- En el cálculo-λ el *proceso de evaluación* es estudiado explícitamente, lo cual permite estudiar problemas que aparecen con frecuencia, como el orden de evaluación.

- Una extensión del cálculo-λ, el cálculo-λ tipado, permite trazar una analogía entre programas y demostraciones.

- Los lenguajes de programación funcional tienen su fundamento el cálculo-λ.

- El cálculo-λ permite estudiar explícitamente el tratamiento de las variables ligadas, cuestión que aparece en varios dominios, como en el cálculo de predicados.

El cálculo-λ consiste en un lenguaje, junto con ciertas operaciones y relaciones definidas sobre los términos del lenguaje. El lenguaje del cálculo-λ puro es extremadamente simple: se define a partir de un conjunto de variables, que se supone numerable y ordenado, mediante la aplicación de una cantidad (finita) de veces estas reglas:

(VAR) • Una variable es un término del cálculo-λ

(ABS) • Dado un término A y una variable x, el término $(\lambda x.A)$ es un término del cálculo-λ. A un término de esta forma se llama *abstracción*.

(APP) • Dados dos términos A y B, el término (AB). A un término de esta forma se llama *aplicación*.

Al conjunto de todos los términos del lenguaje lo denominaremos L.

Damos como ejemplo algunos elementos de L:

- x es el término formado por la aplicación de la regla (VAR) a partir de la variable x

- (xy) es el término obtenido por la aplicación de la regla (APP) a partir de los términos x e y, los cuales a su vez son obtenidos por la aplicación de la regla (VAR), respectivamente, a partir de las variables x e y

- $(\lambda z.(zw))$ se obtiene aplicando la regla (ABS) a partir de la variable z y el término (zw), obtenido a su ver por (APP)

- Se puede proceder de modo análogo para obtener, por ejemplo, los términos: $(\lambda x.(xw))$, $((\lambda z.x)(\_w.w))$, $((\lambda z.x)y)$, $(x(\lambda y.(yw)))$.

Al símbolo λ se lo llama *ligador* y su aparición distingue a las variables de un término en dos tipos: variables libres y variables ligadas. Por ejemplo, en el término:

$$(\lambda x.((\lambda w.(x(wz)))y))$$

Las variables x,w están ligadas (por sus respectivos λ), mientras que z,y están libres. Para un término dado A, se definen los conjuntos $FV(A)$ y $BV(A)$ que son, respectivamente, el conjunto de variables libres y el conjunto de variables ligadas de A.

¿En que sentido los términos del cálculo-λ representan funciones? Consideremos por ejemplo la función $f(x)=x+1$ definida sobre los números naturales. Se dice que f es una función de x, que, dado un número natural para x, da como resultado $x+1$. La formalización clásica de las funciones define a f como un conjunto de pares ordenados, en este caso, el conjunto

$$f=\{(x,x+1) \quad x \quad \mathbf{N}\}=\{(1,2),(2,3),(3,4),...\}$$

Sin embargo en el cálculo-λ nos interesa considerar a las funciones como reglas. Una notación contemporánea, más próxima a este concepto es

$$x \rightarrow x+1$$

Esta notación insinúa que la función es una regla que asigna, dado un valor para x, el resultado de calcular x+1. Apartándonos un poco del cálculo-λ *puro* que acabamos de presentar, podemos expresar la función mencionada como:

$$(\lambda x.x + 1)$$

La función definida por $(\lambda x.x+1)$ equivale en cierto modo a $x \to x+1$ en tanto define la regla a aplicar para calcular el resultado de la función dado un valor para x. Sin embargo, es importante destacar que $x \to x+1$ es una notación del meta-lenguaje, mientras que $(\lambda x.x+1)$ es un término *del lenguaje* en el cálculo-λ (extendido con algunas constantes[1]). Como consecuencia, el cálculo-λ permite describir funciones de orden superior. Por ejemplo, el término:

$$(\lambda f.(\lambda x.f(fx)))$$

puede verse como una función que, aplicada a una función f resulta en una función de x que aplica dos veces f al argumento. Por ejemplo, la función anterior aplicada a la función *succ* (sucesor) devuelve la función $(\lambda x.(succ(succx)))$, que en la notación anterior sería $x \to succ(succ(x))$. Otra consecuencia de de la forma en que el cálculo-λ modela las funciones es la posibilidad de describir funciones "anónimas", notemos que tanto la función descripta antes como el resultado de su aplicación son funciones, aunque no tienen un nombre que las identifica. El beneficio de estas cualidades puede apreciarse inmediatamente al trabajar con los lenguajes de programación funcionales, cuyo fundamento teórico es el cálculo-λ.

Ya vimos de qué forma las *abstracciones* representan funciones. La denominación *abstracción* para tales términos tiene el siguiente sentido: el término $\lambda x.x+1$ es la *abstracción* de la fórmula x+1 *respecto* de la variable x, es decir, es la transformación de la fórmula de cálculo x+1 en objeto matemático.

La otra regla importante para construir términos del cálculo-λ es (APP). Con esta regla se modelan los términos que representan la aplicación de una función a su argumento. Por ejemplo, si F es un término que representa una función, el término

$$(F4)$$

representa la aplicación de la función F al argumento 4 y se corresponde con notación habitual $F(4)$. De igual modo:

$$((\lambda x.x+1)4)$$

es la aplicación de la función ($\lambda x.x$+1) al argumento 4.

En el cálculo-λ los procesos relacionados con el cómputo se representan mediante relaciones denominadas *relaciones de reducción*, definidas sobre el conjunto de los términos.

La principal de estas relaciones es la llamada λ-reducción, que modela el concepto de evaluación entendida como aplicación de una función a un argumento. Antes de presentarla, debemos introducir otro importante concepto: el de sustitución.

Como en otros dominios, en el cálculo-λ se define una operación de reemplazo de variables por términos. En el cálculo-λ puro esta es una operación del meta-lenguaje que se denomina sustitución, y se anota:

$$A[x:=B]$$

Informalmente quiere decir: sustituir en el término A las apariciones libres de la variable x por el término B. Por ejemplo:

$$(\lambda x.(yx))[y:=(wz)]$$

es el término

$$(\lambda x.(wz)x)$$

Debido a la presencia del símbolo ligador λ esta operación requiere de ciertos cuidados para preservar el sentido de función:

- La sustitución no se realiza debajo de un λ que liga a la misma variable que se quiere sustituir. Es decir, $(\lambda x.A)[x:=B] = (\lambda x.A)$. La razón es que x, al estar ligada por el λ, es el argumento de la función cuyo cuerpo es A y no un término con sentido fuera de la función, y por lo tanto las apariciones libres de x en A están “escondidas” por el ligador.

- La sustitución puede ser problemática en casos como este:

$$(\lambda x.(x+y))[y:=x]$$

pues si se reemplaza ingenuamente se obtiene:

$$(\lambda x.(x+x))$$

lo cual es claramente incorrecto. Este problema se previene renombrando la variable del ligador antes de sustituir, de modo que se obtiene:

$$(\lambda w.(w+x))$$

β

Ahora estamos en condiciones de examinar la relación $\overset{\beta}{♦}$ que modela en el cálculo-λ el cómputo de la aplicación de una función a un argumento.

La relación $\overset{\beta}{♦}$ se define como la clausura contextual de la denominada *noción de reducción* β, definida por:

$$((\lambda x.A)B)\ \beta\ A[x:=B]$$

En otras palabras, $\overset{\beta}{♦}$ es el reemplazo de un subtérmino de la forma ((_*x*.*A*)*B*) por *A*[*x*:=*B*] en cualquier lugar de un término. A un término de la forma ((_*x*.*A*)*B*) se lo llama _-redex, y al resultado de *A*[*x*:=*B*] su _-contracto.

Esta noción modela el concepto intuitivo de evaluación de una función, como se ve en el siguiente ejemplo, que corresponde a la función *x* Æ *x*+1 aplicada al argumento 3. Intuitivamente, el término ((_*x*.*x*+1)3) se evalúa

$$((\lambda x.x+1)3)♦\ 3+1$$

donde ♦ indica el proceso de evaluación.

Notemos que podemos escribir:

$$((\lambda x.x+1)3)♦\ (x+1)[x:=3]$$

es decir, el resultado de evaluar la aplicación consiste en sustituir las apariciones libres del argumento, *x*, en el cuerpo de la función, *x*+1, por el argumento 3. Esto es exactamente lo que sanciona la noción β, de modo que la flecha ♦ puede reempla zarse por $\overset{\beta}{♦}$.

Pasemos a otro ejemplo, esta vez en el cálculo-λ puro. El siguiente es un caso correcto de β-reducción[2]:

$$(\lambda\ x.(\lambda\ z.zx)w\backslash)v \overset{\beta}{♦} (\lambda x.wx)v$$

por la aplicación de la noción β al subtérmino $(\lambda z.zx)w$, lo cual da como resultado $zx[z:=w]=wx$.

Notemos que a su vez el término obtenido por la reducción anterior puede reducirse otra vez:

$$(\lambda x.wx)v \xrightarrow{\beta} wv$$

Es habitual considerar la clausura reflexiva-transitiva de $\xrightarrow{\beta}$, que consiste en aplicar cero o más pasos de reducción, y se anota $\xrightarrow{\beta}{}^{*}$. Los dos pasos de reducción recién descriptos justifican que puede escribirse

$$(\lambda x.(\lambda z.zx)w)v \xrightarrow{\beta}{}^{*} wv$$

Notemos también que además de la secuencia de pasos de reducción que describimos:

$$(\lambda x.(\lambda z.zx)w)v \xrightarrow{\beta} (\lambda x.wx)v \xrightarrow{\beta} wv$$

también puede construirse otra secuencia partiendo del mismo término inicial

$$(\lambda x.(\lambda z.zx)w)v \xrightarrow{\beta} (\lambda z.zv)w \;\; (\beta, \xrightarrow{\beta})\; wv$$

alcanzándose en ambos casos el mismo resultado. Esta propiedad de $\xrightarrow{\beta}$, i.e. que distintos caminos de reducción confluyen al mismo resultado, es obviamente una propiedad deseable para todas las relaciones de reducción y se la estudia bajo el nombre de *confluencia*.

4. La relación de α-equivalencia

Consideremos los términos

$$(\lambda x.x) \text{ y } (\lambda y.y)$$

Se nota inmediatamente que ambos representan el mismo concepto, que es la función identidad. Su aplicación a cualquier argumento resulta, en un paso de evaluación, en el mismo argumento, lo que se describe mediante la β-reducción observando que para cualquier término A se verifica

$$(\lambda\, x.x)A \xrightarrow{\beta} A \;\text{ y }\; (\lambda\, y.y)A \xrightarrow{\beta} A$$

La diferencia entre ellos es sólo sintáctica, y está en el nombre elegido para su variable ligada, *x* en el primer caso e *y* en el segundo.

En dominios diferentes al cálculo-_ también encontramos expresiones que sólo difieren en el nombre de variables ligadas, teniendo idéntico significado; consideremos las fórmulas de la lógica de primer orden

$$x\ Px \text{ y } \quad y\ Py$$

y las definiciones de funciones del estilo que comentamos en la primera sección

$$x \to x+1 \text{ y } y \to y+1$$

En el primer caso, el cuantificador universal hace las veces de símbolo ligador; en el segundo, la característica que denota que una variable está ligada es su aparición a la izquierda del símbolo $\to$.

Para asociar en el cálculo -λ a los términos que representan el mismo concepto, se define la relación de α-*equivalencia*, que notaremos como $\equiv_\alpha$, y que agrupa a los términos que sólo difieren en el nombre de una o varias de sus variables ligadas.

La α-equivalencia modela dentro del cálculo-λ una noción intuitivamente clara; a pesar de ello, la literatura presenta distintas definiciones para la relación $\equiv_\alpha$, véanse p.ej. las que se proponen en [**¡Error!No se encuentra el origen de la referencia.**ine, Bloo, o Rose]. En nuestra tesis utilizamos la definición que aparece en [Hindley and Seldin], que transcribimos a continuación

Definición 4.1 (cambio de variable ligada y $\equiv_\alpha$) *Sea un termino P que contiene una ocurrencia de* $\lambda\, x.M$, *y sea y FV(M). El acto de reemplazar este* $\lambda x.M$ *por*

$$\lambda y.M[x:=y]$$

se llama cambio de variable ligada (CVL, notación (*cvl*,♦)) *en P. Diremos que*

$$P \equiv_\alpha Q$$

si y sólo si Q ha sido obtenido a partir de P por un secuencia (posiblemente vacía) de cambios de variable ligada.

La α-equivalencia es, efectivamente, una relación de equivalencia[3] . Por lo tanto, quedan definidos los conceptos de *clase de* α*-equivalencia* y *conjunto cociente* del conjunto de términos *L* por la relación $\equiv_\alpha$. Para notar dicho conjunto cociente utilizamos la letra Λ.

Al trabajar en el cálculo-λ debería ser posible evitar cualquier distinción entre términos α-equivalentes; tal distinción no presenta ningún interés. Dicho en otras palabras, *debe ser posible considerar al cálculo-λ como un cálculo ya no sobre tér - minos, sino sobre clases de α-equivalencia*, considerándose a la sustitución y a las relaciones de reducción como definidas en Λ. Los términos pasan a ser en ésta visión, meros *representantes* de su clase de α-equivalencia. En su obra clásica sobre cálculo-λ, Barendregt lo expresa claramente

> (_) se considerará a los (λ-)términos módulo cambio de variables ligadas, o sea módulo $\equiv_\alpha$. Por lo tanto los λ-términos son representantes de los objetos en los que estamos interesados. [Barendregt, apéndice C, p. 578].

mientras que Hindley y Seldin indican que

> Los términos α-equivalentes tienen idéntica interpretación y juegan idénticos roles en cualquier aplicación del cálculo-λ. [Hindley and Seldin, comentario a la definición 1.16, p.9].

Por otro lado, la sintaxis usual del cálculo-λ denota términos, y *las definiciones de la sustitución y las relaciones de reducción se hacen sobre términos*, esto es, sobre el conjunto *L*.

En resumen, operamos con términos para obtener conclusiones acerca de sus clases de α-equivalencia. Debemos asegurar la corrección de tal forma de operar; de otro modo, el cálculo-λ resultaría inadecuado como teoría.

Una forma de tratar esta cuestión consiste en definir una sintaxis distinta, que denote clases en lugar de términos, y realizar las definiciones sobre esta nueva sintaxis. De esta forma, se estaría operando directamente sobre clases de λ-equivalencia. Este es el caso de la notación de De Bruijn que se describe en [**¡Error!No se encuentra el origen de la referencia.**]; hay otras propuestas en este sentido entre las cuales mencionamos la que aparece en [Gordon].

Algunos autores consideran indispensable la utilización de una notación de este estilo para obtener una formalización correcta del cálculo-λ. Gérard Huet lo indica de esta forma

el incordio que resulta de arrastrar explícitamente la α-equivalencia a toda la teoría sintáctica del cálculo-λ elimina cualquier esperanza de desarrollar la teoría con el rigor que requiere su formalización. [**¡Error!No se encuentra el origen de la referencia.**, p.10].

Los resultados principales de nuestra tesis, que expondremos en la siguiente sección, apoyan un camino ortogonal al anterior, que consiste en realizar las definiciones estrictamente sobre términos, y demostrar que el resultado de pasar a clases estas definiciones resulta en operaciones y relaciones bien definidas sobre Δ. Esto se aparta de algunas presentaciones del cálculo-λ en las cuales la sustitución se define como una operación sobre clases de α-equivalencia (como en la presentación del cálculo-λ de [Barendregt]) o como una "operación no determinística" sobre *L* (como aparece en [Bloo]); o bien se restringen las definiciones a términos que no resulten problemáticos para la sustitución (este es el sentido de la llamada *convención de variables*, que aparece en [Barendregt, 2.1.12, p.26] y luego repetidamente en la literatura).

Respecto de la *sustitución* definida estrictamente sobre *L*, un lema que aparece recurrentemente en la literatura garantiza que al transladar la definición hecha sobre términos se obtiene una operación bien definida sobre Λ. Tal lema es el siguiente:

Lema 4.2 (la sustitución preserva la α-equivalencia) *Si* $A \equiv_\alpha A'$, $B \equiv_\alpha B'$ *entonces* $A[x:=B] \equiv_\alpha A'[x:=B']$

Demostración. Es el lema 1.2.27 [**¡Error! No se encuentra el origen de la referencia.** Lombardi y Vetere, p.28], la demostración está en pp.28-32.

Este lema indica que la sustitución preserva la _-equivalencia pues afirma que la _-equivalencia de los operadores garantiza la de los resultados.

En la sección siguiente describiremos una forma de garantizar que las *relaciones de reducción* pueden considerarse definidas sobre clases de α-equivalencia.

La β-reducción ya presentada es la principal de las relaciones de reducción en el cálculo-λ, pero no es la única. Existen otros conceptos relacionados con el cómputo que también son modelados dentro del cálculo-λ como relaciones de reducción.

Los resultados que presentaremos se aplican a relaciones de reducción en general, tratando a la β-reducción como un caso particular. Para ello tratamos a las relaciones de reducción partiendo del concepto de noción de reducción, y derivando a partir del mismo las relaciones que modelan la reducción en un paso y en cero o más pasos.

En principio consideraremos como *noción de reducción* a cualquier relación binaria sobre *L*, coincidiendo con la definición 3.1.2 [Barendregt **¡Error!No se encuentra el origen de la referencia.**, p.51]. Si ρ es una noción de reducción, definimos

- la *r-reducción en un paso*, notación $\overset{\rho}{\blacklozenge}$, como la clausura contextual de ρ.

- la ρ-*reducción en n pasos*, notación (ρ,), como la clausura reflexiva-transitiva de $\overset{\rho}{\blacklozenge}$.

5. Relaciones de reducción sobre clases de α-equivalencia

En la sección anterior planteamos la necesidad de poder considerar a los términos como meros representantes de su clase de α-equivalencia respecto de las nociones y relaciones de reducción, que deben poder definirse correctamente sobre clases a partir de las definiciones dadas sobre términos.

La propiedad que presentamos a continuación formaliza esta cuestión respecto de una relación binaria arbitraria en *L*.

Definición 5.1 (Propiedad A7) *Sea* ρ *una relación definida en L. Decimos que* ρ *tiene la propiedad A7 si y sólo si*

$$A \,\rho\, B \quad A \equiv_\alpha A' \qquad B'.\ A' \,\rho\, B' \quad B \equiv_\alpha B'$$

para cualesquiera A,B,A'.

El que una relación ρ cumpla A7 garantiza que si desde un representante de una cierta clase de α-equivalencia (la clase de *A*) puede alcanzarse un representante de alguna clase (la clase de *B*), entonces desde cualquier otro representante de la misma clase de partida podrá alcanzarse mediante la misma relación algún representante de la misma clase de llegada; esto es, cualquier término α-equivalente a *A* estará relacionado con algún término α-equivalente a *B*.

Y viceversa: alcanza con que exista algún término en la clase de *A* (llamémoslo *A*'') relacionado con un representante de una cierta clase de llegada (al que llamaremos *B*'') para garantizar que *A* está relacionado con algún representante de la clase de *B*''; en términos de la propiedad tenemos $A'' \,\rho\, B''$ $A'' \equiv_\alpha A$ y por lo tanto existe algún *B* tal que $A \,\rho\, B$ $B'' \equiv_\alpha B$.

Respecto de toda relación que cumpla la propiedad A7, para analizar qué *clases* están relacionadas por ρ con la clase de *A*, es indistinto partir de *A* o de cualquier otro término en la misma clase[4] . Por lo tanto, los términos pueden considerarse meros representantes de su clase de α-equivalencia.

Si ρ es una noción de reducción, o una relación derivada de una noción, A7 garantiza que el conjunto de clases alcanzables a partir de una clase determinada mediante una noción, un paso de reducción o bien cero o más pasos de reducción (según el tipo de relación que sea ρ) es el mismo para cualquier representante de la clase de partida que se elija.

Para poder afirmar que el concepto de cómputo modelado por una noción de reducción puede definirse correctamente sobre clases, debe demostrarse que tanto la noción como sus relaciones derivadas (en particular las relaciones de reducción en uno y en n pasos) cumplen la propiedad A7. El análisis de la noción no es suficiente; en la sección 2.4 de [Lombardi y Vetere] presentamos y describimos ejemplos de nociones de reducción que cumplen A7, pero cuya reducción en un paso no tienen esta propiedad.

En la literatura sobre cálculo-λ resultados similares a A7 aparecen enunciados, y a veces demostrados, para la noción β o alguna relación derivada de ella.

En la tesis doctoral de Alejandro Ríos la propiedad aparece demostrada para la β-reducción en un paso; es su lema A.7 [Ríos, p.164] y motivó la denominación A7 que usamos en este trabajo.

En la tesis doctoral de Roel Bloo aparece una variación de A7, también para la β-reducción en un paso (proposición 2.18 [**¡Error!No se encuentra el origen de la referencia.**, p. 21]), precedida por este comentario

> Antes de tratar con propiedades de la β-reducción, debemos asegurarnos de que la α-equivalencia y la β-reducción no entren en conflicto (*do not clash* en el original) [**¡Error!No se encuentra el origen de la referencia.**o, p. 21] de donde queda claro que esta es la forma que Bloo indica para verificar que la β-reducción puede considerarse definida en Λ.

En la presentación que Krivine hace del cálculo-λ en [Krivine], se demuestra la propiedad A7 para la noción β (proposición 3 [Krivine, p.10]), indicando a continuación que

La proposición precedente indica que esta noción está definida sobre Λ [Krivine, p.10] .

En nuestro trabajo de licenciatura, preferimos caracterizar a A7 como una propiedad general aplicable a cualquier relación binaria sobre el conjunto de términos, y realizar un estudio de esta propiedad independiente de nociones o relaciones de reducción específicas.

Encontramos un conjunto de tres condiciones, aplicables *sobre las nociones de reducción*, con la siguiente propiedad: que una noción cumpla con el conjunto de las tres condiciones es suficiente para garantizar que *tanto ella como todas sus relaciones derivadas* cumplen A7.

Una de las tres condiciones que la noción debe cumplir es la misma A7; a continuación presentamos las otras dos y formalizamos lo recién expuesto:

Definición 5.2 (Monotonía de variables libres) *Sea* ρ *una relación definida en L. Decimos que* ρ *tiene* monotonía de variables libres *(abreviada MFV) si y sólo si*

$$A \rho B \quad FV(A) \quad FV(B)$$

Definición 5.3 (Propiedad A6) *Sea* ρ *una relación definida en L. Decimos que* ρ *tiene la propiedad A6 si y sólo si*

$$A \rho B \qquad x,T \;\; C,D \,.\, C \equiv_\alpha A[x:=T] \quad D \equiv_\alpha B[x:=T] \quad C \rho D$$

Proposición 5.4 (Propiedad A7 para relaciones derivadas) *Sea una relación* ρ *que cumple MFV, A6 y A7. Entonces las relaciones de reducción en uno y en n pasos que resultan de considerar a* ρ *una noción de reducción (esto es, la clausura contextual de* ρ *y la clausura reflexiva-transitiva de la anterior) cumplen A7.*

Demostración. Esta proposición coincide con la conjunción del corolario 2.2.10 más la proposición 2.2.11 de [**¡Error!No se encuentra el origen de la referencia.** Lombardi y Vetere]; en la sección 2.2 (pp.46-57) se elaboran las demostraciones.

Este tratamiento generaliza al que se encuentra en la literatura en dos sentidos.

Por un lado, el análisis de la propiedad A7 está hecho *a partir de un concepto de reducción arbitrario*, en vez de estudiarse específicamente la β-reducción.

Por otro, es suficientre demostrar las tres propiedades sobre la noción para obtener los resultados deseados sobre las relaciones derivadas; *un estudio de la noción es suficiente* para mostrar que el concepto de cómputo que modela queda correctamente definido sobre clases.

Como consecuencia de esta doble generalización, el resultado para la β-reducción se obtiene como un caso particular del planteo exhibido en esta sección, siendo suficiente demostrar las propiedades MFV, A6 y A7 para la noción β. Formalmente

Proposición 5.5 *La noción* β *tiene las propiedades MFV, A6 y A7.*

Demostración. Proposiciones 2.3.1, 2.3.2 y 2.3.3 en [Lombardi y Vetere, pp. 57-59].

Proposición 5.6 *Las relaciones* $\overset{\beta}{\blacklozenge}$ *de* β*-reducción en un paso, y* $\overset{\beta}{\blacklozenge}{}^{*}$ *de* β*-reducción en n pasos, tienen la propiedad A7.*

Demostración. Corolario de las proposiciones 5.5 y 5.4. garantizando la proposición 5.6 que la β-reducción queda bien definida sobre clases de α-equivalencia.

La reducción del análisis a la noción de reducción reduce efectivamente el trabajo necesario; las demostraciones directas de A7 que elaboramos para la β-reducción en un paso, así como la desarrollada por Ríos para su tesis, resultan mucho más largas de las demostraciones para la noción que exhibimos en la tesis.

El procedimiento formalizado por la proposición 5.4 puede aplicarse a cualquier noción que modele un concepto significativo para el cómputo, para garantizar que el concepto de reducción que queda definido en el cálculo-λ es correcto respecto de su definición sobre clases de α-equivalencia. En nuestra tesis lo utilizamos para la η-reducción, que junto con la β-reducción modelan el principio de extensionalidad.

Consecuentemente, proponemos revisar el concepto de *noción de reducción*, considerando como tal ya no cualquier relación binaria sobre los términos del cálculo-λ, sino solamente aquellas que cumplen MFV, A6 y A7.

6. Conclusiones y trabajo futuro

Los resultados expuestos nos permiten, por un lado, hacer las definiciones sobre el conjunto de términos, principalmente el concepto de β-reducción; y por otro, considerar al cálculo-λ como un cálculo sobre el conjunto de clases. Dicho de otro modo, es posible operar con términos, y a la vez considerarlos como representantes de su clase de α-equivalencia. La propiedad A7 nos garantiza que este procedimiento es correcto.

Creemos que uno de los puntos importantes de este enfoque reside en la adecuación intuitiva de los conceptos definidos. En primer lugar, el lenguaje se define inductivamente con un sencillo conjunto de reglas, y todos los términos son válidos. En segundo lugar, la operación de sustitución es una función total, i.e. el dominio de la sustitución es *L* y su resultado está unívocamente determinado por sus argumentos. Por último, se tienen relaciones de reducción definidas para todos los términos de *L*, es decir, lo esperado intuitivamente.

Una vez definidos los conceptos, tal como describimos, de modo que resultan adecuados a la intuición; se presenta otra cuestión a considerar: sería inaceptable obtener resultados con distinto significado (i.e. no α-equivalentes) cuando se computa el resultado de dos aplicaciones (de funciones a sus argumentos) que tienen el mismo significado (i.e. α-equivalentes). Los resultados expuestos en este trabajo proveen un puente entre ambas cuestiones, en un marco absolutamente formal.

Ciertos aspectos de este trabajo merecen una indagación mayor. Utilizando las herramientas obtenidas, puede encararse la verificación de las relaciones de reducción definidas en el cálculo-λ extendido con constantes. También podría resultar interesante la generalización de estos resultados a orden superior, esto es, a un lenguaje donde el concepto de ligador está definido de manera abstracta, y para el cual el cálculo-λ y otros cálculos sean ejemplos particulares.

Bibliografía

Barendregt, H. P. (1984). *The Lambda Calculus: Its Syntax and Semantics*. Elsevier, Amsterdam.

Bloo, R. (1997). *Preservation of Strong Normalisation for Explicit Substitution.* PhD thesis, Eindhoven University of Technology.

Bruijn, N. G. de (1972). “Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the church-rosser theorem”. *Indag. Math.*, 34, pp.381-392.

Gordon, A. D. (1994). “Amechanisation of name-carrying syntax up to alpha-conversion”. *Proceedings of the 6th int. workshop on Higher Order Logic Theorem Proving and its Applications*, vol. 780 of *LNCS*, pp. 414-426, Vancouver, Springer-Verlag.

Hindley,J. R., J. P. Seldin (1986). *Introduction to combinators and lambda-calculus*. London Mathematical Society, London.

Huet, G. (1992). "Constructive computation theory part 1. Notes de cours". http://pauillac.inria.fr/bin/psearch/publioscope? huet.

Krivine, J. L . (1990). *Lambda-calcul types et modèles*. Masson, Paris.

Carlos Lombardi, C. E. Vetere (2002). *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.

Ríos, A. (1993). *Contributions à l'etude des Lambda-calculs avec Substitutions Explicites*. PhD thesis, Université Paris VII.

Rose, K. H. (1996). *Operational Reduction Models for Functional Programming Languages*. PhD thesis, University of Copenhagen, Universitetsparken 1, DK-2100 København Ø.

Notas:

[1] Desde el punto de vista formal, la extensión del cálculo-λ con los números naturales como constantes no es necesaria. Church, quien inventó el cálculo-λ, definió términos, que hoy llamamos *numerales de Church*, para representar a los números naturales. El número 1 se representa con el término $\lambda f.(\lambda x.(fx))$, el 2 con el término $\lambda f.(\lambda x.(f(fx)))$, el 3 con $\lambda f.(\lambda x.(f(f(fx))))$, etc. En general, el número n se representa con $\lambda f.(\lambda x.(f^n x))$, donde f^n debe entenderse como "repetir n veces el término f". De igual modo existen términos para representar las funciones sucesor, suma, producto, etc. Usaremos de todos modos la notación del cálculo-λ extendido para ilustrar más fácilmente los conceptos.

[2] Eliminamos, por claridad, algunos paréntesis redundantes.

[3] Una demostración exhaustiva aparece en [**¡Error!No se encuentra el origen de la referencia.**, lemas 1.2.20 a 1.2.22, págs. 19 a 21]

[4] En la sección 2.5 de [**¡Error!No se encuentra el origen de la referencia.**] se define formalmente cómo construir relaciones sobre clases a partir de relaciones definidas sobre términos, demostrándose que A7 garantiza que la relación sobre clases resultante queda bien definida y que por lo tanto queda justificada la práctica de operar sobre términos para obtener conclusiones acerca de clases.