

An implementable definition of α -equivalence in λ -calculus

Functional Programming Project

Fabio Brau

23 gennaio 2021

Indice

1	Introduction	1
2	Lambda Terms	1
2.1	Variabili libere e legate	2
2.2	Operatore di Sostituzione	3
3	α-equivalenza	4
4	Equivalenza Semantica	5

1 Introduction

This project's aim is to provide a formal definition of α -equivalence on the λ -terms without seeing it as a transitive closure of a relation and without assuming conventions on the name of the variables.

2 Lambda Terms

From an intuitive points of view, a λ -term is a representation of a mathematical function written as a combination of *variables*. It is quite surprising to notice that “a systematic notation for functions is lacking in ordinary mathematics” [Curry]. In fact, the meaning of $f(x)$, that is the usual accepted notation to indicate a function (*Euler Notation*), is not uniquely determined and has to be deduced from the context. Sometime we refers to the definition of a function of the variable x ; sometimes we refers to the evaluation of the function in a value (or in a point, a vector, a matrix, a set and so on) equal to x .

An unambiguous way to indicate that the function f depends on the variable x could be the notation: $x \mapsto f(x)$. And, we can use the notation $f(x)$ when we intend to evaluate f in the object x . The introduction of the symbol λ could help, in a typographic sense, by shortening the notations $x \mapsto f(x)$ in $\lambda x.f(x)$.

Definizione 1. Let be $V = \{v_1, v_2, \dots\}$ an infinite set of *variables*. The set of the λ -terms, indicated with Λ , is recursively defined as follow

- $V \subseteq \Lambda$, i.e., each variable is a λ -term;
- If M is a λ -term, then $\lambda x.M \in \Lambda$, for any variable x , is a λ -term called *abstraction*;
- If M, N are λ -terms then (MN) is a new λ -term called *application*.

It's compulsory to observe that the symbols $M, N, \dots, v_0, v_1, \dots$ have to be intended as names of λ -terms. The *assignment* of a name to a λ -term, indicated with $=$, has to be ever intend in the metalanguage. For example, in the formula $M = \lambda x.x$, the symbol M it is just the label of the λ -term $\lambda x.x$.

È evidente che in una lambda espressione il numero di volte in cui appare un simbolo di parentesi o il simbolo λ può essere non trascurabile. Assumendo che i due operatori del metalinguaggio siano associativi a sinistra possiamo introdurre la seguente notazione che fornisce una scrittura più concisa.

Notazione 1. Siano x, y variabili e sia M una λ -espressione, allora la notazione $\lambda xy.M$ è una scrittura abbreviata per $\lambda x.\lambda y.M$ che per associatività a sinistra rappresenta univocamente $\lambda x.(\lambda y.M)$. Analogamente, siano X, Y, Z lambda espressioni, la notazione XYZ rappresenta univocamente (per associatività a sinistra) $((XY)Z)$.

Definizione 2. Diremo che due lambda espressioni M, N sono *sintatticamente equivalenti*, e scriveremo $M \equiv N$, se possono essere mutualmente scritte una nell'altra utilizzando la Notazione 1.

2.1 Variabili libere e legate

Consideriamo la λ -espressione $M = \lambda x.xy$ e sia $z \neq x$. Non sorprende che M sia, in un qualche senso da definire, equivalente a $\lambda z.zy$, anche se le due λ -espressioni non sono sintatticamente equivalenti, i.e. $M \not\equiv \lambda z.zy$. Questo è dovuto al fatto che, intuitivamente, la variabile x appare legata a λ nella forma λx .

Definizione 3. Data una λ -espressione M , l'insieme delle *variabili libere* $\mathcal{F}(M)$ è costruito induttivamente come segue:

- $\mathcal{F}(x) = \{x\};$
- $\mathcal{F}(\lambda x.P) = \mathcal{F}(P) \setminus \{x\};$
- $\mathcal{F}(PQ) = \mathcal{F}(P) \cup \mathcal{F}(Q);$

dove x è una qualunque variabile e P, Q sono λ -espressioni. Analogamente, l'insieme delle *variabili legate* $\mathcal{B}(M)$ è definito ricorsivamente sulla costruzione di M come segue:

- $\mathcal{B}(x) = \emptyset;$
- $\mathcal{B}(\lambda x.P) = \mathcal{B}(P) \cup \{x\};$
- $\mathcal{B}(PQ) = \mathcal{B}(P) \cup \mathcal{B}(Q);$

Osserviamo che può accadere che per una λ -espressione M vale $\mathcal{B}(M) \cap \mathcal{F}(M) \neq \emptyset$. I seguenti esempi possono essere di chiarimento.

Esempio 1. La λ -espressione $x(\lambda x.xx)$ ha una sola variabile libera (x) e solo una variabile legata (x).

Definizione 4. Definiamo l'insieme dei *combinatori* $\Lambda^0 \subseteq \Lambda$ come

$$\Lambda^0 = \{M \in \Lambda : \mathcal{F}(M) = \emptyset\}. \quad (1)$$

2.2 Operatore di Sostituzione

Consideriamo una λ -espressione M contenente una variabile x . Vogliamo definire formalmente l'operazione di *sostituzione* e cioè l'operazione nel metalinguaggio che consiste nel sostituire alla variabile x una λ -espressione N .

Definizione 5 (Curry). Sia $M, N \in \Lambda$, definiamo l'operatore di *sostituzione senza cattura* che sostituisce la variabile x nella λ -espressione M con la λ -espressione N , restituendo una λ -espressione indicata da $M[x := N]$, in modo ricorsivo sulla costruzione delle λ -espressioni:

Caso 1 Se M è una variabile:

- Se $M = x$ allora $M[x := N] \equiv N;$
- Se $M = y \neq x$ allora $M[x := N] \equiv y;$

Caso 2 Se $M = M_1 M_2$ è una applicazione:

- $M[x := N] \equiv (M_1[x := N])(M_2[x := N]);$

Caso 3 Se M è una astrazione:

- Se $M = \lambda x.M_1$ allora $M[x := N] \equiv M;$

- Se $M = \lambda y.M_1$, con $x \neq y$, allora

$$M[x := N] \equiv \lambda z.M_1[y := z][x := N]$$

dove: $z = y$ se $x \notin \mathcal{F}(M_1)$ o $y \notin \mathcal{F}(N)$; z scelta tale che non compare né in M né in N altrimenti;

I primi due casi sono intuitivi, mentre il terzo è più delicato. Consideriamo $M = \lambda xy.x$ e $N = y$. Nella sostituzione $M[x := N]$, la variabile libera y presente in N verrebbe catturata, in quanto è presente in M come variabile legata, diventando anch'essa legata: Questo fenomeno si chiama *cattura* e rende necessario introdurre la nuova variabile z .

Lemma 1. *Valgono le seguenti proposizioni*

1. Se $x \in \mathcal{F}(M)$ e $y \notin \mathcal{F}(M)$, allora $M \equiv M[x := y][y := x]$;
2. Se $x \in \mathcal{F}(M)$ e $y \notin \mathcal{F}(M)$, allora $x \notin \mathcal{F}(M[x := y])$ e $y \in \mathcal{F}(M[x := y])$;

3 α -equivalenza

Intuitivamente una λ -espressione è ottenuta tramite astrazione e/o applicazione di λ -espressioni. Due λ -espressioni si dicono α equivalenti se possono essere riscritte una nell'altra a meno di sostituzione di variabili legate. Questa conversione, per quanto sia facilmente comprensibile a livello intuitivo, nasconde una serie di complicazioni. La seguente definizione è una riformulazione della definizione formale fornita da [Curry et al.].

Definizione 6 (α -equivalenza). Definiamo la relazione \equiv_α induttivamente simultaneamente alla costruzione della lambda espressioni. Preso Λ_0 come l'insieme delle λ -espressioni di rango 0 (costituite da una sola variabile, senza astrazioni o applicazioni), definiamo

$$\forall x, y \in \Lambda_0, \quad x \equiv_\alpha y \iff x = y.$$

Osserviamo che \equiv_α è di equivalenza su Λ_0 . Consideriamo ora l'insieme delle λ -espressioni di rango $k > 0$ definito ricorsivamente come $\Lambda_k = \hat{\Lambda}_k \cup \bar{\Lambda}_k$, dove

$$\begin{aligned} \hat{\Lambda}_k &= \{\lambda x.M : M \in \Lambda_{k-1}\} \\ \bar{\Lambda}_k &= \{(MN), (NM) : M \in \Lambda_{k-1}, N \in \Lambda_i, i < k\} \end{aligned}$$

Su questo insieme definiamo la relazione \equiv_α come

- Se $M, N \in \hat{\Lambda}_k$, con $M = \lambda x.M_1$ e $N = \lambda y.N_1$ allora

– Se $x = y$, allora

$$M \equiv_\alpha N \iff M_1 \equiv_\alpha N_1$$

– Se $x \neq y$, allora

$$M \equiv_\alpha N \iff N_1 \equiv_\alpha M_1[x := y] \quad \wedge \quad (y \notin \mathcal{F}(M_1) \wedge x \notin \mathcal{F}(N_1))$$

• Se $M, N \in \bar{\Lambda}_k$, con $M = M_1 M_2$ e $N = N_1 N_2$, allora

$$M \equiv_\alpha N \iff M_1 \equiv_\alpha N_1 \wedge M_2 \equiv_\alpha N_2$$

Per induzione forte e Lemma 1 si dimostra che \equiv_α è di equivalenza su Λ_k per ogni k . In conclusione, essendo $\Lambda = \cup_k \Lambda_k$, le relazioni di equivalenza si estendono ad una relazione di equivalenza sulle lambda espressioni.

Lemma 2. *La sostituzione è invariante per α -equivalenza. Formalmente, se $M \equiv_\alpha M'$ e se $N \equiv_\alpha N'$ allora*

$$M[x := N] \equiv_\alpha M'[x := N']$$

4 Equivalenza Semantica

Possiamo ora definire una prima versione della equivalenza semantica come segue

Definizione 7 (Teoria λ). Teoria del primo ordine sul linguaggio Λ con relazione di equivalenza semantica \doteq per cui valgono i seguenti assiomi

$$(\alpha) \quad M \equiv_\alpha N \Rightarrow M \doteq N;$$

$$(\beta) \quad (\lambda x.M)N \doteq M[x := N];$$

$$(\xi) \quad M \doteq N \Rightarrow \lambda x.M \doteq \lambda x.N;$$

$$(I) \quad \doteq \subseteq \Lambda \times \Lambda \text{ è di equivalenza};$$

$$(II) \quad M \doteq N \Rightarrow \quad ZM \doteq ZN \wedge MZ \doteq NZ;$$