Rappresentabilità di Funzioni nel Lambda-calcolo Polimorfico

Alessio Marchetti

1 Introduzione

Il λ-calcolo è un sistema formale sviluppato negli anni '30 da Alonzo Church. Lo scopo originario era quello di fondare la matematica, e non è stato raggiunto in quanto si rivelò inconsistente, come dimostrato da Kleene e Rosser nel 1936. Un sottoinsieme del sistema si è comunque sviluppato per la sua capacità di esprimere computazioni mediante astrazione su variabili e sostituzione. Lo studio del λ-calcolo è dunque lo studio di entità dette λ-termini che svologono allo stesso tempo il ruolo di programmi e di dati su cui i programmi lavorano. Sui termini si andrà a considerare una relazione di ordine, detta riduzione, che rappresenta l'esecuzione dei programmi. All'interno del λ-calcolo, con opportune codifiche è possibile rappresentare i numeri naturali e le tutte funzioni calcolabili, parziali e totali. Poichè l'insieme delle funzioni calcolabili totali non è ricorsivamente enumerabile esistono dei termini la cui computazione non termina (diremo che non sono normalizzanti).

Il λ-calcolo tipato è una variante del λ-calcolo ad ogni termine è associata un'entità sintattica detta tipo. Esso ha origine nei lavori di Haskell Curry (1934) e di Church (1940). La riduzione in questo caso è ridefinita aggiungendo vincoli sul come è possibile comporre i termini in base al loro tipo, e in particolare rende chiara la distinzione tra dati e programmi. Come conseguenza si può dimostrare che questa variante ha la proprietà di normalizzazione, ovvero tutte le computazioni terminano e tutti i termini sono normalizzanti. I tipi sono studiati anche in ambito informatico per la verifica in modo automatico della presenza di alcuni errori che dovrebbero essere altrimenti cercati a mano dal programmatore.

In questa tesi ci occuperemo di una variante del calcolo detta sistema F, anche nota come λ-calcolo polimorfico o λ-calcolo del secondo ordine. Essa è stata sviluppata dal logico Jean-Yves Girard (1972) e dall'informatico John Charles Reynolds (1974). Il sistema F è essenzialmente una variante del λ-calcolo tipato in cui viene aggiunta una quantificazione universale sui tipi.

Anche per il sistema F vale la proprietà di normalizzazione. Troveremo dunque che le funzioni rappresentabili nel sistema F sono un sottoinsieme delle funzioni calcolabili totali, e ne daremo una caratterizzazione più precisa: esse sono esattamente le funzioni di cui l'aritmetica di Heyting del secondo ordine dimostra la totalità. Mostreremo quindi un esempio di funzione non rappresentabile nel sistema F e dedurremo dunque la consistenza dell'aritmetica di Peano del secondo ordine a partire dal risultato di normalizzazione.

Metteremo inoltre tale risultato in confronto con un risultato equivalente su un altra variante del λ-calcolo, detta sistema T di Gödel, per cui vale ugualmente la proprietà di normalizzazione. Nel sistema T infatti le funzioni rappresentabili sono esattamente quelle che l'aritmetica di Heyting del primo ordine dimostra essere totali.

2 Il λ-calcolo non tipato

Il λ-calcolo è un sistema formale per descrivere delle computazioni. [...]

In questa tesi ci occuperemo di studiare alcune proprietà di diversi tipi di λ-calcoli. Iniziamo dunque definendo gli elementi fondamentali del λ-calcolo più semplice, ovvero quello non tipato.

Definizione 2.1 I termini del λ-calcolo semplice si definiscono induttivamente come:

- Le variabili $x_1, x_2, \ldots$ sono termini.
- Se t e v sono termini, allora anche l'applicazione (tv) è un termine.
- Se t è un termine e x è una variabile, l'astrazione $(\lambda x.t)$ è un termine.

Dato un termine t, i sottotermini sono tutti i termini che appaiono nella costruzione induttiva di t.

L'idea intuitiva dietro questa definizione è quella che un termine del tipo $\lambda x.t$ corrisponde a un programma con input x e corpo t. Inoltre un'applicazione della forma tv rappresenta un programma t quando con input v. Un fatto interessante da notare è che i termini possono svolgere indistintamente il ruolo di programma e di dato su cui un programma opera.

Per comodità e leggibilità delle notazioni, ometteremo spesso le parentesi sottointendendo che l'applicazione si associa a sinistra (e quindi $xyz = (xy)z$) e l'astrazione si associa a destra, usando un singolo simbolo λ, per esempio $\lambda xyz.yyxz = \lambda x.(\lambda y.(\lambda z.yyxz))$. Vettori di variabili.

In modo simile a quanto si fa comunemente in logica è utile distinguere le occorrenze di una variabile in una formula tra occorrenze libere e legate. In particolare l'astrazione su una varibile lega tale variabile. Più formalmente:

Definizione 2.2 Un'occorrenza della variabile x in un termine t si dice legata se esiste un sottotermine del tipo $\lambda x.t'$ che la contiene. Si dice legata altrimenti.

Inoltre, dato un termine t si definiscono induttivamente le sue variabili libere come:

- Se $t = x$ dove x è una variabile, allora è l'unica variabile libera d t.
- Se $t = uv$, allora le variabili libere di t sono tutte e sole le variabili libere che compaiono in u o in v.
- Se $t = \lambda x.u$, allora le variabili libere di t sono tutte le variabili libere di u con l'esclusione di x.

Nel seguito considereremo i termini modulo il rinominare le variabili legate. Questo corrisponde al fatto che in un programma è possibile rinominare i parametri formali delle funzioni (modificando consistentemente le loro occorrenze nei corpi di tali funzioni). In particolare la relazione di equivalenza che mette in relazione un termine con tutti i termini uguali a esso a meno del rinominare le variabili legate si chiama α-equivalenza. Da qui in avanti, per semplicità, abuseremo della notazione riferendoci alle classi di equivalenza con i loro elementi.

Con qualche accortezza per evitare la cattura delle variabili si può definire la sostituzione di un termine su una variabile.

Definizione 2.3 Se u e v sono termini e x è una variabile, allora la sostituzione di v su x in u è il termine $u[v/x]$ definito come:

- Se $u = x$ allora $u[v/x] = v$.
- Se $u = y$ con y una variabile distinta da x, allora $u[v/x] = y$.
- Se $u = tw$, allora $u[v/x] = (t[v/x])(w[v/x])$.
- Se $u = \lambda y.t$ e y è distinta da x e non appare libera in v, allora $u[v/x] = \lambda y.t[v/x]$.

A questo punto abbiamo presentato tutti gli strumenti per introdurre una relazione fondamentale sui termini del calcolo, ovvero la conversione.

Definizione 2.4 Dati due termini u e v, si dice che u si converte a v e scriverermo $u \rightsquigarrow_C v$, se v è ottenuto da u sostituendo un sottotermine nella forma $(\lambda x.u')u''$ con $u'[u''/x]$.

Si dice che u si riduce a v se esistono n termini $u_1, \ldots, u_n$ con $u_1 = u$ e $u_n = v$ tali che $u_1 \rightsquigarrow_C u_2 \rightsquigarrow_C \ldots \rightsquigarrow_C u_n$. In tal caso scriveremo $u \rightsquigarrow v$.

Notiamo che la definizione dice essenzialmente che $\rightsquigarrow$ è la chiusura transitiva di $\rightsquigarrow_C$. Tali nozioni di conversione e riduzione sono chiamati in letteratura anche β-conversione e β-riduzione. La più piccola relazione di equivalenza che contiene la β-conversione è detta β-equivalenza.

L'idea di conversione corrisponde all'esecuzione di un passo del programma corrispondente al termine che viene convertito. Notiamo però che volendo ridurre un termine, la successione delle conversioni non è univocamente determinata, come nel termine $((\lambda x.xx)y)((\lambda x.x)z)$.

Facciamo ora due esempi importanti di riduzione.

Esempio 2.5

$$\begin{gathered}(\lambda f.\lambda x.f(fx))(\lambda y.yy) \rightsquigarrow_C \\ \lambda x.(\lambda y.yy)((\lambda y.yy)x) \rightsquigarrow_C \\ \lambda x.(\lambda y.yy)(xx) \rightsquigarrow_C \\ \lambda x.xxxx.\end{gathered}$$

Esempio 2.6

$$\begin{gathered}\Omega = (\lambda x.xx)(\lambda x.xx) \rightsquigarrow_C \\ (\lambda x.xx)(\lambda x.xx) \rightsquigarrow_C \\ \ldots\end{gathered}$$

Notiamo che il primo esempio finisce con un termine che non può essere ulteriormente convertito, mentre nel secondo la successione di conversioni è infinita. Questa distinzione è importante e conduce alle seguenti definizioni.

Definizione 2.7 Un termine si dice in forma normale se non può essere ulteriormente convertito. Un termine si dice normalizzante se può essere convertito a un termine in forma normale.

Esistono comunque termini che pur essendo normalizzanti ammettono una successione infinita di conversioni:

Esempio 2.8

$$(\lambda x.\lambda y.y)\Omega z \rightsquigarrow_C z$$

ma convertendo ad ogni passo il termine Ω (si veda la sua definizione nell'esempio precedente), si ottiene che:

$$(\lambda x.\lambda y.y)\Omega z \rightsquigarrow_C$$
$$(\lambda x.\lambda y.y)\Omega z \rightsquigarrow_C$$
$$\dots$$

È allora utile introdurre le seguenti nozioni:

Definizione 2.9 Dato un termine t si definisce $\nu(t)$ il massimo numero di conversioni necessarie per portare t in forma normale, ossia

$$\nu(t) = \sup\{n \ \$ \ \exists u_1, \dots, u_n \text{ per cui}$$
$$t \rightsquigarrow_C u_1 \rightsquigarrow_C \dots \rightsquigarrow_C u_n \text{ e } u_n \text{ è in forma normale}\}.$$

Un termine t si dice fortemente normalizzante se $\nu(t) < \infty$.

Si osservi che se un termine è fortemente normalizzante allora è anche normalizzante, e non può essere convertito un numero infinito di volte.

Convertendo sempre la redex più a sinistra si ottiene comunque sempre una forma normale.

Un'importante proprietà di cui gode la riduzione è detta proprietà di Church-Rosser, che in particolare implica l'unicità del termine in forma normale a cui si converte un termine normalizzante.

Teorema 2.10 Sia t un termine. Se u e v sono termini per cui $t \rightsquigarrow u$ e $t \rightsquigarrow v$, allora esiste un quarto termine w tale che $u \rightsquigarrow w$ e $v \rightsquigarrow w$. Disegno del diamante.

2.1 Rappresentabilità nel λ-calcolo non tipato

Tra i termini del λ-calcolo ne possiamo individuare alcuni per metterli in corrispondenza con i numeri naturali.

Definizione 2.11 Dato un numero naturale n definiamo il corrispondente numerale $\overline{n}$ come il termine $\lambda f x.f^n x$, dove il simbolo $f^n x$ indica $f(f(\cdots f(x)\cdots))$.

In particolare $\overline{n}$ è un termine che presa una funzione f, la compone con se stessa n volte. Osserviamo che inoltre i numerali sono termini in forma normale, dunque per la proprietà di Church-Rosser sono anche termini distinti anche modulo la β-equivalenza.

Definizione 2.12 Data una funzione (eventualmente parziale) $\phi\colon \mathbb{N} \to \mathbb{N}$, si dice che un termine t rappresenta ϕ se per ogni coppia di naturali m e n vale che $\phi(n) = m$ se e solo se $t\overline{n} \rightsquigarrow \overline{m}$ e $t\overline{n}$ non è normalizzabile quando $\phi(n) = \bot$.

Facciamo ora alcuni esempi di funzioni rappresentabili:

Esempio 2.13 Il termine $A = \lambda pqfx.(pf)(qfx)$ rappresenta l'addizione. Per esempio:

$$A\overline{2}\ \overline{3} \rightsquigarrow \lambda fx.f^2(f^3x) = \lambda fx.f^5x = \overline{5}.$$

In modo simile esistono i termini $M = \lambda pqfx.q(pf)x$ e $E = \lambda pqfx.qpfx$ che rappresentano rispettivamente la moltiplicazione e l'esponenziazione. Questo ultimo termine è leggermente differente dagli altri due in quanto è l'unico in cui un numerale viene direttamente applicato ad un altro numerale. Vedremo che questa differenza sarà decisiva per la rappresentabilità in alcune varianti di λ-calcolo.

Teorema 2.14 Le funzioni rappresentabili nel λ-calcolo non tipato sono esattamente le funzioni calcolabili.

Un nodo cruciale nella dimostrazione del precedente teorema è l'esistenza di un combinatore di punto fisso, ovvero di un termine t tale che per ogni termine u vale che $tu = u(tu)$, modula la β-conversione. Un esempio di combinatore di punto fisso è il termine $Y = \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$, come si può facilmente verificare. Esso è noto come combinatore di punto fisso di Curry. Notiamo inoltre che tale termine non è normalizzante, ma più in generale possiamo dimostrare che nessun combinatore di punto fisso t può essere normalizzante. Infatti se indichiamo con t' la forma normale di tx, dove x è una variabile, allora anche t' è normalizzante. Inoltre vale che $t' = xt'$, modulo la β-conversione, ma entrambi questi termini sono in forma normale, e dunque abbiamo l'assurdo.

3 Il λ-calcolo tipato semplice

Nel λ-calcolo non tipato, esistono termini per cui la normalizzazione non corrisponde all'idea intuitiva di "semplificazione", come nell'esempio quello con Ω oppure nel caso ancora peggiore seguente.

Esempio 3.1

$$(\lambda x.xxx)(\lambda x.xxx) \rightsquigarrow$$
$$(\lambda x.xxx)(\lambda x.xxx)(\lambda x.xxx) \rightsquigarrow$$
$$\dots$$

Il problema alla base di questo comportamento è il fatto che non vi è distizione tra dati e porgrammi e in particolare è permessa l'applicazione di un termine a se stesso. Versioni più sofisticate del calcolo puntano dunque a introdurre dei vincoli sull'applicazione dei termini, ed è per questo motivo che si introducono i tipi. L'idea è quella di associare ad ogni termine un tipo, e permettere l'applicazione di termini solo se i loro tipi sono compatibili.

Definiamo allora una variante del λ-calcolo detta λ-calcolo tipato semplice.

Definizione 3.2 I tipi del $\lambda|$-calcolo tipato semplice sono definiti induttivamente come:

- $U_1, U_2, \dots$ sono tipi, detti variabili di tipo.
- Se U e V sono tipi, allora anche $(U \to V)$ è un tipo.

Per comodità, in assenza di parentesi, l'associatività di $\to$ è a dstra: per esempio $U \to V \to W = (U \to (V \to W))$.

A questo punto ricostruiamo i termini associando a ciascuno di essi un relativo tipo.

- Per ogni tipo U, le variabili $x_1^U, x_2^U, \dots$ sono termini di tipo U.
- Se t e v sono termini di tipo rispettivamente $U \to V$ e U, allora l'applicazione (tv) è un termine di tipo V.
- Se t è un termine di tipo V e x è una variabile di tipo U, allora l'astrazione $(\lambda x.t)$ è un termine di tipo $U \to V$.

Per indicare che un termine t è di tipo U scriveremo anche t^U.

Si vede dunque che il tipo $U \to V$ corrisponde al tipo delle funzioni dai termini di tipo U ai termini di tipo V, e che l'applicazione è consentita solo quando "il dominio della funzione e il tipo dell'argomento coincidono".

In modo identico a quanto già fatto per il λ-calcolo semplice, è possibile definire le nozioni di conversione, riduzione e forma normale. Si noti che entrambe le relazione conservano il tipo dei termini.

3.1 Normalizzazione Forte per il Tipato Semplice

L'obiettivo di questa sezione è quella di dimostrare il seguente importante risultato:

Teorema 3.3 Tutti i termini del λ-calcolo tipato semplice sono fortemente normalizzanti.

Da questo fatto discende che l'espressività di questo calcolo è molto ridotta rispetto a quella del λ-calcolo tipato semplice. Per esempio non possiamo trovare nessun combinatore di punto fisso e vedremo che la classe di funzioni rappresentabili è anch'essa ridotta.

Definiamo come prima cosa la nozione di riducibilità.

Definizione 3.4 Sia U un tipo, e t un termine di tipo U. Definiamo induttivamente il suo insieme di riducibilità RED_U come:

- Se U è atomico, t è riducibile se e solo se è fortemente normalizzante.
- Se $T = V \to W$, t è riducibile se e solo se per ogni termine riducibile v di tipo V, il termine tv è riducibile di tipo W.

Osserviamo che sono termini riducibili tutte le variabili di tipo.

Definizione 3.5 Diciamo che un termine t è neutrale se è nella forma xy.

L'idea dietro alla neutralità è che se t è un termine neutrale e v un termine per cui $tv \rightsquigarrow_C u$, allora $u = t'v$ oppure $u = tv'$ dove t' e v' sono conversioni rispettivamente di t e v. In particolare non ci sono step di riduzione in cui v o un suo sottotermine viene sostituito in una variabile di t.

Dimostriamo ora che gli insiemi di riducibili godono di alcune proprietà, che saranno utili a dimostrare il teorema di questa sezione e torneranno anche utili nello studio della normalizzazione nel sistema F.

Proposizione 3.6

(CR1) Se $t \in \mathrm{RED}_U$, allora t è fortemente normalizzante.
(CR2) Se $t \in \mathrm{RED}_U$ e $t \rightsquigarrow u$, allora $u \in \mathrm{RED}_U$.
(CR3) Se t è neutrale di tipo U e per ogni t' per cui $t \rightsquigarrow_C t'$ vale che $t' \in \mathrm{RED}_U$, allora anche $t \in \mathrm{RED}_U$.

Notiamo che la prima proprietà indica che essere riducibile implica l'essere fortemente normalizzante. La seconda proprietà permette di conoscere la riducibilità di un termine data la riducibilità di un termine precedente in una catena di conversioni. Infine la terza proprietà permette di conoscere la riducibilità di un termne data quella delle sue conversioni.

Dimostrazione. La dimostrazione è per induzione sulla complessità dei tipi.

Iniziamo dal caso in cui il tipo U sia una variabile di tipo. Allora, poichè i riducibili di tipo U sono i termini fortemente normalizzanti, (CR1) è una tautologia. Se un termine t è fortemente normalizzante e $t \rightsquigarrow t'$, allora anche t' è fortemente normalizzante perché vale che $\nu(t') < \nu(t)$. Dunque anche (CR2) vale. Per (CR3), sia t un termine neutrale per cui tutte le conversioni sono fortemente normalizzanti. Allora vale che $\nu(t)$ è pari al massimo di $\nu(t')$ al variare di t' tra le conversioni di t, e dunque è finito.

Consideriamo adesso il tipo $U \to V$. Supponiamo che t sia un riducibile di tale tipo, e supponiamo che x sia una variabie d tipo U. Poichè x è neutrale e normale, essa è riducibile. Allora anche tx è riducibile, per la definizione di riducibilità sul tipo freccia. Osserviamo ora che $\nu(t) \leq \nu(tx)$, infatti ad ogni catena di conversioni $t \rightsquigarrow_C t_1 \rightsquigarrow \cdots \rightsquigarrow t_n$ possiamo associare la catena $tx \rightsquigarrow_C t_1 x \rightsquigarrow \cdots \rightsquigarrow t_n x$. Poichè $\nu(tx)$ è finito, t è fortemente normalizzante, e (CR1) è dimostrato.

Se consideriamo un termine t di tipo $U \to V$ riducibile e un termine t' tale che $t \rightsquigarrow t'$, allora per ogni termnine u di tipo U vale che $tu \rightsquigarrow t'u$. Utilizzando l'ipotesi induttiva di (CR2) su V, otteniamo che anche $t'u$ è riducibile. Per cui anche t' è riducibile e (CR2) vale.

Supponiamo ora di avere t neutrale per cui tutte le sue conversioni siano riducibili. Sia u un riducibile di tipo U. L'obiettivo è mostrare che tu è riducibile. Per l'ipotesi induttiva per U, già sappiamo che u è fortemente normalizzabile, per cui possiamo ragionare per induzione su $\nu(u)$. Notiamo che per neutralità di t, tu si può convertire soltanto in $t'u$, con t' conversione di t, oppure in tu', con u' conversione di u. Nel primo caso sappiamo che t' è riducibile, e dunque anche $t'u$ lo è. Nel secondo caso possiamo osservare che $\nu(u') < \nu(u)$ e dunque per induzione otteniamo nuovamente che $t'u$ è riducibile. Poichè tu si converte soltanto a riducibili, è anch'esso riducibile, per ipotesi induttiva di (CR3). □

A questo punto dimostriamo un utile lemma.

Lemma 3.7 Se per tutti i termini riducibili u di tipo U, il termine $v[u/x]$ è riducibile, allora anche il termine $\lambda x.v$ è riducibile.

Dimostrazione. Supponiamo che $v[u/x]$ sia di tipo V, allora il termine $\lambda x.v$ è di tipo $U \to V$. Allora vogliamo dimostrare che per ogni termine riducibile u di tipo U vale che $(\lambda x.v)u$ è riducibile. Notiamo che v è riducibile, infatti x è riducibile di tipo U.

Ragioniamo per induzione sulla somma $\nu(v)+\nu(u)$ per dimostrare che tutte le conversoni di $(\lambda x.v)u$ sono riducibili. Il termine $(\lambda x.v)u$ si può convertire in:

- $v[u/x]$, che è riducibile per ipotesi.
- $(\lambda x.v')u$, con v' conversione di v. Allora si ha che v' è riducibile per (CR2), e vale $\nu(v') < \nu(v)$ e quindi per ipotesi induttiva $\lambda x.v'$ è riducibile.
- $(\lambda x.v)u'$, con u' conversione di u. In questo caso, similmente a prima, u' è riducibile, e vale $\nu(v') < \nu(v)$. Nuovamente $(\lambda x.v)u'$ è anch'esso riducibile per ipotesi induttiva.

Concludiamo per (CR3), che assicura che $\lambda x.v$ sia dunque riducibile. □

Adesso si dimostra una versione più forte del teorema.

Proposizione 3.8 Sia t un termine le cui variabili libere compaiono tra $x_1, \ldots, x_n = \underline{x}$, di tipo rispettivamente $U_1, \ldots, U_n$. Siano $u_1, \ldots, u_n = \underline{u}$ termini riducibili di tipo rispettivamente $U_1, \ldots, U_n$. Allora il termine $t[\underline{u}/\underline{x}]$ è riducibile. Intediamo per $t[\underline{u}/\underline{x}]$ la sostituzione $t[u_1/P]x_1 \cdots [u_n/P]x_n$.

Dimostrazione. Per induzione sulla complessità di t:

- Se $t = x_i$, allora la tesi è una tautologia.
- Se $t = wv$, allora per l'ipotesi induttiva $w[\underline{u}/\underline{x}]$ e $v[\underline{u}/\underline{x}]$ sono riducibili. Ne consegue che $t[\underline{u}/\underline{x}] = w[\underline{u}/\underline{x}]v[\underline{u}/\underline{x}]$ è riducibile.
- Se $t = \lambda y.w$ di tipo $V \to W$, allora per ipotesi induttiva, $t[\underline{u}/\underline{x}][v/y]$ è riducibile per tutti i termini v di tipo V. Allora per il lemma quale lemma? si ottiene che $\lambda y.w[\underline{u}/\underline{x}]$ è riducibile.

□

La dimostrazione del teorema di normalizzazione forte segue da quella della proposizione ponendo $\underline{u} = \underline{x}$.

3.2 Rappresentabilità per il Tipato Semplice

All'interno del λ-calcolo tipato semplice è possibile rappresentare i numerali, con gli stessi termini presentati per il caso non tipato. Tuttavia, la scelta del tipo per in numeri naturali non è unica. Infatti per ogni tipo U possiamo costruire per ogni naturale n il corrispondente numerale

$$\overline{n} = \lambda f^{U\to U}.\lambda x^U.f^n x$$

di tipo $\text{Int} = (U \to U) \to U \to U$. Avendo a disposizione i numerali, possiamo definire le funzioni rappresentabili in modo identico a quanto fatto con il λ-calcolo tipato semplice.

Possiamo associare dei tipi anche ai termini che avevamo usato nel λ-calcolo semplice per rappresentare la somma e la moltiplicazione, infatti

$$A = \lambda p^{\text{Int}} q^{\text{Int}} f^{U\to U} x^U.(pf)(qfx)$$

è la versione tipata per l'addizione e

$$M = \lambda p^{\text{Int}} q^{\text{Int}} f^{U\to U} x^U.q(pf)x$$

lo è per la moltiplicazione.

Possiamo dare un ulteriore esempio di funzione rappresentabile, che è quella corrispondente all'*if/then/else*, ovvero la funzione condizionale $f(x, y, z)$ che vale y se x è non nullo e vale z altrimenti. Essa è rappresentata dal termine

$$C = \lambda p^{\text{Int}} q^{\text{Int}} r^{\text{Int}} f^{U\to U} x^U.p(\lambda y^U.qfx)(rfx)$$

Come importante conseguenza della proprietà di normalizzazione forte si ha una conseguente riduzione della classe delle funzioni rappresentabili, che devono essere per forza totali. Inoltre non tutte le funzioni totali sono rappresentabili, infatti se cosi fosse sarebbe possibile trovare una loro enumerazione con i termini del calcolo, ma ciò non è possibile perché l'insieme delle (codifiche delle) funzioni totali non è ricorsivamente enumerabile.

Per esempio il termine che avevamo usato per rappresentare l'esponenziazione non può essere tipato infatti l'applicazione di un numerale a un altro numerale, ovvero un termine di tipo Int a un altro termine di tipo Int, non è permessa dalle regole del λ-calcolo tipato semplice.

Si può dire di più, perché nessun termine può rappresentare l'esponenziazione e più in generale vale il seguente teorema:

Teorema 3.9 Le funzioni rappresentabili nel λ-calcolo tipato semplice sono esattamente le funzioni generate dalle costanti 0 e 1 e dalle funzioni di somma, moltiplicazione e condizionale.

Un verso è immediato, avendo già trovato i termini A, M e C. L'altro è da fare?

4 Il Sistema T di Gödel

Il grosso problema del calcolo tipato semplice è che i naturali non hanno un tipo unico, e non è possibile definire funzioni per ricorsione primitiva. Per ovviare a questo problema introduciamo un nuovo calcolo, il sistema T di Gödel, in cui vengono artificialmente inseriti tipi per gli interi, i booleani e alcuni termini che rappresentano delle funzioni basilari su di essi.

Definizione 4.1 I termini del Sistema T...

Inoltre si danno anche le regole per le conversioni.

Definizione 4.2 La riduzione per il Sistema T...

Si ha dunque che i termini O e S rappresentano rispettivamente lo zero e il successore, e T e F i valori booleani vero e falso. D rappresenta l'operatore *if/then/else*, mentre R è l'operatore di ricorsione primitiva.

4.1 Normalizzazione Forte per il Sistema T

Teorema 4.3 Tutti i termini del sistema T sono fortemente normalizzanti.

5 Il sistema F

5.1 Normalizzazione per il Sistema F

6 Aritmetiche di Peano e di Heyting

Definizione 6.1 Il linguaggio per la logica del secondo ordine è lo stesso di quello del primo ordine con l'aggiunta per ogni naturale n di numerabili simboli X^n, che chiameremo variabili di relazione.

Le formule atomiche sono $\bot$ e espressioni della forma $X^n(t_1, \ldots, t_n)$, dove i t_i sono termini del linguaggio.

Le formule sono definite induttivamente come:

- Le formule atomiche.
- Date ϕ e ψ formule, sono formule anche $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \to \psi$.
- Data una formula ϕ e una variabile x, sono formule anche $\forall x \phi$ e $\exists x \phi$.

- Data una formula ϕ e una variabile di relazione X, sono formule anche $\forall X\phi$ e $\exists X\phi$.

Si definisce inoltre la formula $\neg\phi$ come $\phi \to \bot$.

In modo naturale possiamo definire il concetto di variabili libere in una formula:

Definizione 6.2

- Le variabili libere di $X(t_1, \ldots, t_n)$ con X variabile di relazione n-aria, sono l'unione di tutte le variabili libere che compaiono nei termini t_i per ogni i e X.
- Le variabili libere di $r(t_1, \ldots, t_n)$ con r simbolo d relazione n-aria sono l'unione di tutte le variabili libere che compaiono nei termini t_i per ogni i.
- Le variabili libere di $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \to \psi$ sono l'unione delle variabili libere di ϕ e ψ.
- Le variabili libere di $\forall x\phi$ e $\exists x\phi$ con x variabile, sono le variabili libere di ϕ meno x.
- Le variabili libere di $\forall X\phi$ e $\exists X\phi$ con X variabile di relazione, sono le variabili libere di ϕ meno X.

La sostituzione di termini nelle variabili è la sostituzione standard, con l'attenzione di evitare la cattura delle variabili. Per sostituire invece relazioni al posto di variabili di relazione ci appoggeremo al concetto di specie.

Definizione 6.3 Sia ϕ una formula e $x_1, \ldots, x_n$ delle variabili individuali, allora l'espressione $\lambda x_1, \ldots, x_n.\phi$ è una specie di arietà n. Si noti che le variabili x_i possono apparire o non apparire in ϕ.

Le variabili libere di $\lambda x_1, \ldots, x_n.\phi$ sono le variabili libere di ϕ meno le variabili x_i.

Abbrevieremo inoltre l'espressione $\lambda x_1, \ldots x_n.X(x_1, \ldots, x_n)$ con semplicemente X e se $\underline{x} = x_1, \ldots, x_n$, abbrevieremo $\lambda x_1, \ldots, x_n.\phi$ con $\lambda \underline{x}.\phi$.

Definiamo inoltre induttivamente la sostituzione di una specie di arietà n $\lambda \underline{x}.\phi$ in una variabile di relazione n-aria X:

- $\bot[\lambda \underline{x}.\phi/X] = \bot$.
- $r(t_1, \ldots, t_n)[\lambda \underline{x}.\phi/X] = r(t_1, \ldots, t_n)$ quando r è una relazione oppure una variabile di relazione diversa da X.

- $(X(\underline{t})[\lambda \underline{x}.\phi/X] = \phi[\underline{t}/\underline{x}]$.
- $(\eta \to \psi)[\lambda \underline{x}.\phi/X] = \eta[\lambda \underline{x}.\phi/X] \to \psi[\lambda \underline{x}.\phi/X]$ e equivalentemente per $\eta \vee \psi$ e $\eta \wedge \psi$.
- $(\forall x \eta)[\lambda \underline{x}.\phi/X] = \forall x \eta[\lambda \underline{x}.\phi/X]$ per tutte le variabili individuali x che non appaiono libere in $\lambda \underline{x}.\phi$. Equvalentemente per $\exists x \eta$.
- $(\forall Y \eta)[\lambda \underline{x}.\phi/X] = \forall Y \eta[\lambda \underline{x}.\phi/X]$ con Y variabile di relazione diversa da X e Y che non appare libera in $\lambda \underline{x}.\phi$.

A questo punto presentiamo le regole della deduzione naturale per la logica del secondo ordine: regole per la deduzione, p.308

Aggiungendo queste regole alle regole della deduzione naturale per la logica classica del primo ordine, si ottiene il sistema per la logica classica del secondo ordine. Equivalentemente, aggiungendole alle regole per la logica intuizionista del primo ordine si ottiene il sistema per la logica intuizionista del secondo ordine.

Notiamo che è possibile dimostrare in entrambi i tipi di logica il principio di comprensione:

$$\exists Y \ \forall x \ (\phi(x) \leftrightarrow x \in Y).$$

per ogni formula ϕ.

Inoltre molti dei connettivi presentati sono ridondanti: infatti è possibile definirli tutti in termini dei soli $\to$ e $\forall$ (su variabili individuali e di relazione). In particolare: È corretto usare =?

- $\bot = \forall X.X$.
- $\phi \vee \psi = \forall X((\phi \to X) \to (\psi \to X) \to X)$.
- $\phi \wedge \psi = \forall X((\phi \to \psi \to X) \to X)$.
- $\exists x \phi = \forall R(\forall x(\phi \to R) \to R)$.
- $\exists X \phi = \forall R(\forall X(\phi \to R) \to R)$.

Il prossimo passo è quello di mettere in evidenza un rapporto che sussiste tra le proposizioni derivabili dalla logica intuizionista e la logica classica.

Definizione 6.4 Data una formula ϕ definiamo induttivamente la sua traduzione di Gödel $k(\phi)$ come:

- $\neg\neg\phi$ se ϕ è atomica.

- $\neg\neg(k(\eta) \to k(\psi))$ se $\phi = \eta \to \psi$, e in modo di equivalentemente si definisce per gli altri connettivi binari.
- ...

Introduciamo adesso le aritmetiche del secondo ordine. Utilizziamo un linguaggio che ha come unica costante 0, il simbolo di funzione successore S e una relazione binaria di uguaglianza $=$.

A questo punto possiamo dare il seguente risultato:

Proposizione 6.5 La proposizione ϕ è un teorema della logica classica se e solo se $k(\phi)$ è un teorema della logica intuizionista. Dimostrazione?

Consideriamo i seguenti assiomi per l'uguaglianza:

(U1) $\forall a(a = a)$;
(U2) $\forall ab(a = b \to b = a)$;
(U3) $\forall abc(a = b \to b = c \to a = c)$;
(U4) $\forall ab(a = b \to Sa = Sb)$,

in cui i primi tre sono i consueti assiomi per una relazione di equivalenza e l'ultimo è una sorta di passo induttivo.

Aggiungiamo ancora tre assiomi di Peano:

(P1) $\forall ab(Sa = Sb \to a = b)$;
(P2) $\forall a(Sa = 0 \to \bot)$;
(P3) $\forall a\ \mathrm{Int}(a)$,

dove $\mathrm{Int}(a) = \forall X(\forall b(X(b) \to X(Sb)) \to X(0) \to X(a))$ serve a sostituire lo schema di induzione.

Definizione 6.6 Gli assiomi (U1-4) e (P1-3) utilizzati con il sistema della logica classica del secondo ordine definiscono l'aritmetica di peano del secondo ordine PA2. Quando invece sono utilizzati con il sistema della logica intuizionista del secondo ordine definiscono l'aritmetica di Heyting del secondo ordine HA2.

Utilizzando la quantificazione al secondo ordine è anche possibili definire dei predicati per la somma e il prodotto (e anche per le funzioni primitive ricorsive).

Consideriamo T una teoria nel linguaggio dell'aritmetica. Sia inoltre $\mathcal{U}$ la funzione universale, ovvero una formula primitiva ricorsiva tale che $\mathcal{U}(e, n, m)$ è vera se e solo se il programma con codifica e eseguito con input n ha output m.

Diciamo che una funzione è dimostrabilmente totale in T se esiste un programma con codifica e tale che

$$T \vdash \forall n \; \exists! m \; \mathcal{U}(e, n, m).$$

Aggiungere precisazioni per funzioni con più di una variabile?

Notiamo che la formula da dimostrare ha complessità Π^0_2.

Vale il seguente teorema:

Teorema 6.7 Le funzioni dimostrabilmente totali in PA2 sono esattamente le funzioni dimostrabilmente totali in HA2. Dimostrazione?

7 Rappresentabilità in T

7.1 Codifica dei termini

Lo scopo di questa sezione è quello di interpretare il sistema T all'interno delle teorie dell'aritmetica. Per fare ciò serve come prima cosa trovare una codifica con i numeri di Gödel per i termini del calcolo.

Definizione 7.1 Dato un tipo U, gli si associa un numero naturale...
Idem con i termini

Abbiamo a questo punto delle formule che esprimono la conversione, la riduzione, la normalizzazione di un termine, la forte normalizzazione di un termine.

7.2 Le funzioni rappresentabili sono dimostrabilmente totali in PA

Notiamo che le funzioni rappresentabili nel sistema T sono calcolabili. Infatti, dati una funzione f rappresentata da un termine t e un numero naturale n, per la tesi di Church, disponiamo di un algoritmo di calcolo. Si scrive il termine $t\overline{n}$, lo si riduce in forma normale t'. Essa sarà il numerale corrispondente al numero naturale $m = f(n)$.

Lemma 7.2 I termini normali di tipo int sono tutti e soli i numerali. Capire dove mettere questo lemma.

Vogliamo dunque dimostrare il seguente teorema:

Teorema 7.3 Tutte le funzioni rappresentabili dal sistema T sono dimostrabilmente totali in PA.

L'idea di dimostrazione consiste nel ripercorrere la dimostrazione della normalizzazione forte su un singolo termine. Per fare ciò serve esprimere con un predicato la riducibilità di un termine, e poi ragionare per induzione sulle varie riducibilità.

7.3 Le funzioni dimostrabilmente totali in PA sono rappresentabili

8 Rappresentabilità in F

Nel sistema F è presente un tipo corrispondente ai numeri naturali, ovvero il tipo

$$\text{Int} = \Pi X.X \to (X \to X) \to X$$

dove si hanno i termini corrispondenti allo zero e al successore rispettivamente uguali a

$$O = \Lambda X.\lambda x.\lambda f.x$$

$$S = \lambda n.\Lambda X.\lambda x.\lambda f.f(nXxf).$$

Possiamo scrivere allora i numerali come le forme normali di $S^n O$ per ogni n naturale. A questo punto dimostriamo il lemma:

Lemma 8.1 I numerali sono tutti e soli i termini in forma normale di tipo Int.

In realtà la costruzione fatta per i numeri naturali a partire dai costruttori zero e successore può essere generalizzata a qualunque tipo di dato algebrico.

In modo equivalente a quanto già fatto con le altre versioni del λ-calcolo è possibile definire la nozione di funzione rappresentabile, e poi dare una caratterizzazione di tali funzioni.

Teorema 8.2 Le funzioni dimostrabilmente totali in PA2 sono tutte e sole le funzioni rappresentabili nel sistema F.

Iniziamo con la freccia più semplice, ovvero $\Leftarrow$. Come nel caso del sistema T, la dimostrazione della forte normalizzazione di un termine, può essere interpretata in PA2 come una dimostrazione della totalità della funzione corrispondente a tale termine. Infatti per la dimostrazione sono stati utilizzati due principi:

- Lo schema di comprensione, necessario a dimostrare che i riducibili parametrici sono candidati di riducibilità.
- Il principio di induzione.

Notiamo che tuttavia non è possibile esprimere la riducibilità in generale, ma solo per specifici termini.