Università degli Studi di Padova
Dipartimento di Matematica "Tullio Levi-Civita"
Corso di Laurea Magistrale in Informatica

Esame di Teoria dei Tipi

Teoria dei Tipi
Elaborato scritto - 01/09/2020
Eleonora Signor, 1237581

Indice

Capitolo 1

Introduzione

1.1 La triplice faccia della teoria dei tipi

La teoria dei tipi offre una base teorica a fondamento dello sviluppo di:

- **Matematica**: nella teoria degli insiemi;
- **Logica**: come fondamento dei connettivi logici e dei quantificatori, con trattazione mediate tecniche di *proof-theory* per dimostrarne la non falsità o non contradditorietà;
- **Informatica**: per la correttezza dei programmi, da una semantica operazionale a un certo tipo di operazioni.
  Con riferimento alla teoria degli insiemi, visto come linguaggio di programmazione funzionale, è possibile specificare con formule l'obiettivo di un programma e dimostrarne la correttezza attraverso la specifica.

La teoria dei tipi nasce per garantire la *Certified Proof Correctness*. Ovvero la correttezza dei programmi, volta a costruire gli assistenti automatici per le formalizzazioni.

1.2 Come nasce la teoria dei tipi

Gli errori di programmazione sono stati preponderanti alla nascita di metodi automatici, che assicurassero la correttezza del *software*. Alcuni di questi, degni di nota, sono stati:

- incidente nel lancio dell'Apollo 11;
- tragedie sanitarie: incidenti avvenuti tra il 1985-1987, in cui dei pazienti ricevettero una massiccia *overdose* di radiazioni e per la quali alcuni morirono;
- errori di vita civile: riserva di solo due cifre per il campo età all'interno dei *database*. Ecco che una signora danese ricevette per il suo 107-esimo compleanno, una mail, dalle autorità della scuola locale, per iscriversi alla prima elementare.

Per la matematica la correttezza delle dimostrazioni è irrilevante solo quando la soluzione è certa (come accade con il cubo di *Rubik*, dove so che la soluzione è corretta quando ognuno dei lati è uniformemente colorato); e in generale questo è difficile che accada.
Un'esempio di problema, dove la soluzione non è certa, è il Teorema dei Quattro Colori, risolto da un *computer* e la cui prova di correttezza della dimostrazione fu data dal *proof- assistant* Coq. Quest'ultimo basato sulla teoria dei tipi e intellegibile dall'essere umano.

Una citazione importante va al matematico Russo V.V. *Voevodsky*, vincitore della medaglia *Fields*. Esso si battè per la creazione di un *proof assistant*, per rendere le dimostrazioni da informali, per problemi complessi, a completamente formalizzate, con l'impiego della teoria dei tipi. I suoi studi trovano principale applicazione in campo algebrico e geometrico; ma i concetti emersi assunsero delle connotazioni più ampie. *Voevodsky*, difatti, si rese conto che formalizzare equivale a programmare. Ciò significa che la teoria dei tipi permette di vedere una dimostrazione come un programma.

Esiste la certezza assoluta per una certa teoria, esclusivamente, quando ha un numero di assiomi, accettati per fede, molto limitato. In quanto assiomatizzabile da un calcolatore.

In conclusione formalizzare in una teoria dei tipi (come quella degli insiemi) equivale a programmare un programma.

1.3 Il Paradosso di Russell

La base della teoria dei tipi, compresa quella di *Martin-Löf*, si deve a B. *Russell*. Siamo nel 1907 quando nasce la teoria dei tipi, sviluppata nei *Principia Mathematica* da B. *Russel* assieme ad A.N. *Whitehead*. Tale teoria, intesa come logica e non informatica, nasce come soluzione alternativa alla teoria degli insiemi, di allora, con lo scopo di fondare la matematica su un sistema formale accettabile e non contraddittorio.
Di seguito espongo un sistema contraddittorio della teoria degli insiemi.

Linguaggio *L* di una teoria degli insiemi *F*

- L linguaggio del primo ordine ($=$, &, $\rightarrow$, $\vee$, $\forall x$, $\exists x$), con l'aggiunta del predicato $\in$ "appartiene"
- variabili VAR $\ni$ {x, y, z, w,...}

dove x, y, z sono da intendersi come insiemi e x $\in$ y = "x appartiene a y".

All'interno di L c'è una teoria degli insiemi. Tra cui prende posto l'**assioma di comprensione di *Frege***, definito nel modo seguente:
Per ogni formula ϕ(x) vale che $\exists$z $\forall$y (y$\in$z $\Leftrightarrow$ ϕ(y)) [$\equiv$ $\exists$z z = {x | ϕ(x)}]

Teorema (o Paradosso) di *Russell*: la teoria F è contraddittoria.

Dimostrazione:
$\phi(x) = x \notin x$ ($\equiv \neg$ ($x \in x$))
Per l'assioma di comprensione $\exists z\ z = \{x \mid x \notin x\}$ ($\exists z\ \forall y\ (y \in z \Leftrightarrow y \notin y)$).
Ponendo $y=z$ ottengo che $z \in z \Leftrightarrow z \notin z$, che risulta una **contraddizione**.

L'assioma di comprensione è contraddittorio perchè permette di formare insiemi che non appartengono a se stessi.

Come correggere la contraddizione?
La soluzione accettabile è porre agli insiemi una **gerarchia di tipi**. In questo modo l'assioma di comprensione diventa

$$\exists z\ \forall y\ (y \in a \rightarrow (y \in z \Leftrightarrow \phi(y)) \equiv z = \{x \in a \mid \phi(x)\}$$

In questo modo non posso più creare il Paradosso di *Russell*.

Al momento questa teoria dei tipi non è utilizzata. Una sua evoluzione diretta è la teoria dei tipi di *Martin-Löf*.

L'idea di *Russell* fu dunque quella di costruire insiemi partendo da una gerarchia.

1.4 Idee principali nelle teorie di tipo moderne

Le teorie di tipo moderne (chiamate λ-calcolo tipato) nascono, nel corso degli anni '30, dalla combinazione della teoria di tipo di *Russell* con il λ-calcolo di *Church*.

1.4.1 Richiamo della teoria del λ-calcolo di *Church*

Ha origine dalla logica, è un linguaggio in grado di trattare le funzioni e rivolto alla loro formalizzazione. Consiste in un linguaggio formale, le cui componenti principali sono programmi chiamati termini (pensati come funzioni).
La grammatica è la seguente:

$$t := x \mid b_1(b_2) \mid \lambda x.t$$

Esempio di applicazione: $tg(x) \equiv \lambda x.tg(x)$

Regole di computazione di base

$$(\lambda x.t)(b) \rightarrow t[\frac{x}{b}] \qquad \frac{b_1 \rightarrow b_2 \qquad a_1 \rightarrow a_2}{b_1(a_1) \rightarrow b_2(a_2)} \qquad \frac{b_1 \rightarrow b_2}{\lambda x.b_1 \rightarrow \lambda x.b_2}$$

Si dice che un programma si riduca a un altro, cioè converge, solo se c'è una sequenza di riduzioni (applicazione di regole e/o assiomi), che connettono il primo programma con l'ultimo. Si parla, in questo modo, di **chiusura transitiva e simmetrica**, che si conclude quando il programma non è più riducibile. Quanto appena descritto può venire definito in

$t \twoheadrightarrow t'$ sse esiste un numero finito di passi per cui t si riduce a t', ovvero esiste $b_1 \dots b_m$ t.c. $t \to b_1 \to b_2 \dots \to b_m \to t'$.

Il λ calcolo permette di codificare qualsiasi programma scritto in qualunque linguaggio (imperativo, dichiarativo, Java, C++, BASIC, ...). Tuttavia tale linguaggio non codifica solo programmi che terminano, ma anche programmi che non lo fanno. Un esempio di applicazione, per quest'ultima categoria, è un programma con computazione infinita: $\lambda x.x(x)$.
$\lambda x.x(x)$ lo applichiamo a se stesso. Perció diventa $\Lambda \equiv (\lambda x.x(x))(\lambda x.x(x))$ che seguendo la computazione si riduce a

$$x(x)[\frac{x}{\lambda x.x(x)}] \equiv (\lambda x.x(x))(\lambda x.x(x))$$

Dunque esiste una catena di $(\mathrm{t}_i)_{i\in\mathbb{N}}$ di termini $\mathrm{t}_i \to \mathrm{t}_{i+1}$. Ciò significa che Λ non termina in qualunque linguaggio sia interpretato.

Λ risulta un buon metodo per rappresentare le funzioni, ma non è completo, rispetto all'intuizione matematica di funzione. È necessario, per questo, tipare le variabili; ovvero λx.x∈A→B(x∈A).

Il λ-calcolo tipato, nato dal λ-calcolo "puro", è anch'esso un linguaggio di programmazione. Essendo tipato può essere trattato come una teoria degli insiemi.

1.5 Che cosa è un tipo?

Per rispondere a questa domanda è necessario fornire la semantica intuitiva di tipo. Per farlo è utile pensare alla teoria dei tipi come paradigma di fondazione sia logico che matematico che informatico.

Sintassi in teoria dei tipi moderna	**Sintassi in teoria degli insiemi**	**Sintassi in un linguaggio logico/per una logica (anche predicativo)**	**Sintassi in un linguaggio di programmazione**
A *type*	A *set*	A *prop*	A *data type*
a∈A	a∈A	a∈A	a∈A

Tabella 1.1: Sintassi per i diversi paradigmi funzionali.

Per la sintassi:

- nella **teoria dei tipi moderna** a rappresenta un termine e A un tipo;
- nella sintassi in una **teoria degli insiemi** a è un elemento e A un insieme. Coincidendo con la corrispondenza originale in mente da *Russell*.
- nella sintassi in un **linguaggio logico** a rappresenta una dimostrazione di A, e A viene inteso come insieme o tipo delle sue dimostrazioni. Perciò a rappresenta un *proof-term* affermante come la proposizione di A sia vera.

- nella teoria in una sintassi di un **linguaggio di programmazione** a rappresenta un programma e A una specifica.

Dunque quando parliamo di tipo ci riferiamo a un insieme, una proposizione o *data type*, a seconda dell'applicazione di tipo che si ha in mente.

Dal punto di vista logico non si hanno solo proposizioni, ma anche predicati. Parlare solo di tipo non risulta quindi sufficiente. Per questo se si vuole rappresentare non una proposizione, ma un predicato A(x) si usa la sintassi **A(x) prop[x∈D]**.
Dalla logica si sa che i predicati ϕ(x) hanno x senza un dominio specifico, perchè la sintassi non determina che cosa è in x. Al seguito di tutto questo i predicati hanno una variabile che deve essere tipata come ϕ**(x) prop[x∈D]**.
Dunque (definizione di predicato)

$$\exists\ \mathbf{z} \qquad \mathbf{z=\{x\in a|\ \phi(x)\}} \qquad \equiv \qquad \mathbf{\phi(x)prop[x\in\ a]}$$

Quanto appena definito da origine al concetto di **tipo dipendente**, nel quale vengono tipate tutte le variabili che appartengono a una **famiglia di tipo**.

Le famiglie di tipo sono indispensabili per rappresentare il concetto di predicato. Di seguito ho riassunto in forma tabellare le diverse famiglie.

di tipo	**negli insiemi**	**in logica**	**dati dipendenti**
A(x) prop[x∈D]	A(x) set[x∈D]	A(x) prop[x∈D]	A(x) datatype[x∈D]

Tabella 1.2: Famiglia di tipi.

Il concetto di tipo dipendente è stato introdotto per la prima volta da *Martin-Löf*. *Russell* si era limitato a definire esclusivamente il concetto di funzione proposizionale dipendente da un tipo.

1.6 Esempi di tipi

A type	**A set**	**A prop**	**A data type**
N_1 singoletto	l'insieme singoletto	*tt* costante vero	tipo Unit
N_0 vuoto	l'insieme vuoto	$\perp$ costante falso	vuoto come *datatype*
B×C (tipo prodotto)	l'insieme prodotto cartesiano dell'insieme B con l'insieme C	B&C congiunzione della proposizione B e della proposizione C	tipo prodotto cartesiano (come in *set theory*)
B+C (tipo somma binaria)	l'insieme unione disgiunta dell'insieme B con l'insieme C	B∨C disgiunta della proposizione B e della proposizione C	tipo unione disgiunta con codifica
B→C	l'insieme delle funzioni dall'insieme B verso l'insieme C: A→B ≡ {f | f: B→C}	B→C, implicazione della proposizione B e della proposizione C	insieme delle funzioni dal *datatype* B al *datatype* C

Tabella 1.3: Esempi di tipi.

1.6.1 I tipi dipendenti

A(x)type[x∈B]
tipo prodotto dipendente $\prod_{x\in B} C(x)$
tipo somma dipendente disgiunta indiciata $\sum_{x\in B} C(x)$

A(x)set[x∈B]	**A(x)prop[x∈B]**	**A(x)datatype[x∈B]**
$\{f : B \to \coprod_{x\in B} C(x)\}$ $\coprod_{x\in B} C(x) =$ $\{b, c \vert b \in B \quad c \in C(b)\}$	$\forall x \in B \quad C(x)$	tipo prodotto indiciato come in *set theory* (non esiste un *datatype* specifico)
$\dot{\bigcup}_{x\in B} C(x)$ $\coprod_{x\in B} C(x) =$ $\{b, c \vert b \in B \quad c \in C(b)\}$	$\exists x \in B \quad C(x)$	non è primitivo, deriva sempre dalla *set theory*

Tabella 1.4: Tipi dipendenti.

Lo *slogan* principale della teoria dei tipi è quello di tipare le variabili in un linguaggio formale set teorico/computazionale.

Esiste anche il **Tipo uguaglianza**:

- intensionale: Id(B,c,d);
- estensionale: Eq(B,c,d).

Introdotte da *Martin-Löf*.
E i costrutti degli **Universi**, in cui U è universo di proposizioni e di insiemi.

1.7 Regole paradigmatiche per caratterizzare la teoria dei tipi

La teoria dei tipi è stata formalizzata usando la nozione di **giudizio**, dove si asserisce qualcosa come vero.
Ci sono quattro forme di giudizio (nelle quali Γ identifica il contesto):

- **A type[Γ]**: A è un tipo, possibilmente indicato da variabili nel contesto Γ, dipendente da Γ stesso. Rappresenta il giudizio di tipo.
- **A = B type[Γ]**: il tipo A dipendente da Γ è uguale al tipo B dipendente da Γ. Rappresenta il giudizio di uguaglianza di tipo.
- **a $\in$ A [Γ]**: a è un elemento del tipo A, possibilmente indiciato, ovvero dipendente da Γ e dalle sue variabili di contesto. Un esempio di tipo dipendente è l'array, che ha termini di funzioni che dipendono da Γ. Invece il termine non è dipendente quando si parla di funzione costante senza variabili.
- **a = b $\in$ A [Γ]**: a come elemento del tipo A dipende da Γ ed è uguale in modo definizionale/computazionale al termine b. Quest'ultimo, difatti, è elemento del tipo A dipendente da Γ.

All'interno di ogni singolo giudizio si lavora con la teoria dei tipi.
I giudizi sono esclusivamente asserzioni, dicono solo qualcosa quando è vero (non si usano i quantificatori). Essi limitano le frasi che si possono fare per codificare la Logica intuizionistica.

1.7.1 Simbolo $\in$

Il significato di a$\in$A in teoria dei tipi è differente da quello insiemistico. Espongo il concetto con un esempio trattato a lezione:

$$\mathbf{1} \in \mathbf{Nat}$$

- in ***set theory*** usuale $\in$ è tra insiemi. Nell'equazione sopra, 1 rappresenta lui stesso un'insieme e Nat l'insieme dei numeri Naturali. Risulta vero che $1 \equiv \{\varnothing\}$, poichè $0 \equiv \varnothing$.
- invece in **teoria dei tipi** (di *Martin-Löf* come di *Russell*) 1 rappresenta un elemento ma non un tipo e Nat il tipo dei Naturali. Vi è dunque la distinzione tra elemento e tipo (come esiste quella tra programmi e tipi).

1.7.2 Uguaglianza definizionale vs uguaglianza proposizionale

Specifico $\mathbf{a} = \mathbf{b} \in \mathbf{A}[\Gamma]$ come l'uguaglianza computazionale/definizionale, che viene data come primitiva e non va confusa con l'uguaglianza proposizionale/estensionale tra a e b.
L'uguaglianza proposizionale $a =_A b$ è rappresentata non da un giudizio, che asserisce solo ciò che è vero, ma bensì da un tipo Eq(A,a,b) che può anche essere senza termini e/o essere falso, dal punto di vista logico.

Visti come programmi, a e b rappresentano lo stesso programma. In λ-calcolo $a \to b$ oppure $b \to a$ (si riducono). Inoltre a e b possono essere sia termini finali che trovarsi in mezzo alla computazione.

1.7.3 Generazione di contesti

Esiste anche un quinto giudizio ausiliario (§2.1.1) *(F-c)*, che permette di generare i contesti. Tale giudizio, a differenza dei primi quattro, rimane immutato in ogni teoria dei tipi.

1.8 Esercizi

1) Dato il seguente termine, elencare quali sono le sue variabili libere e le sue variabili legate con i lambda termini relativi.

$$\lambda z.(((\lambda x.\lambda x.yx)x)(v\lambda z.\lambda w.v))$$

Soluzione

$\lambda z.(((\lambda x.\boldsymbol{\lambda x}.\mathbf{yx})\mathbf{x})(\mathbf{v}\lambda z.\lambda w.\mathbf{v}))$

- le variabili libere (colorate in verde) sono y, x e v
- le variabili legate con i relativi lambda termini (colorate in rosso) sono la x

2) Rinominare le variabili legate nel seguente termine in modo che non ci siano due variabili legate con lo stesso nome.

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

Soluzione

$x(\boldsymbol{\lambda x}.((\boldsymbol{\lambda x}.x)x))$

Le variabili legate sono le x all'interno delle parentesi tonde. Una possibile rinomina, per evitare che queste variabili legate abbiano lo stesso nome, è $x(\lambda x.((\lambda y.y)x))$, dove la x della parentesi più interna è stata sostituita con la y.

3) Evidenziare di due colori diversi quali sono le variabili libere e quali quelle legate.

$$(\lambda \mathbf{x}.(\mathbf{z}(\lambda \mathbf{z}.((\mathbf{xyz})\mathbf{x}))\mathbf{zx}))\mathbf{x}(\lambda \mathbf{x}.((\lambda \mathbf{y}.\mathbf{yy})(\lambda \mathbf{z}.\mathbf{zz})))$$

Soluzione

$(\lambda x.(z(\lambda z.((xyz)x))zx))x(\lambda x.((\lambda y.yy)(\lambda z.zz)))$

- le variabili libere (colorate in verde) sono z, y e x
- le variabili legate (colorate in rosso) sono x, y e z

4) Descrivere un termine del λ-calcolo, descritto in §1.4.1, che è convergente con almeno un passo di riduzione rispetto a una specifica strategia di riduzione deterministica.

Soluzione

Per definizione un termine t è convergente, rispetto a una strategia di riduzione deterministica, se esiste un numero finito n >= 1 di termini s_1,..., s_n tale che $s_1 \equiv t$ e $s_i \to_1 s_{i+1}$ per i = 1,..., n-1 e s_n non è riducibile ad alcun termine. Prendendo in considerazione una strategia di riduzione deterministica *call-by value*, usata per la semantica nei linguaggi di programmazione, allora un esempio di termine t del λ-calcolo, convergente in almeno un passo di riduzione, è:

$$t \equiv ((\lambda x.x)z)((\lambda y.y)w) \to_1 z((\lambda y.y)w) \to_1 z(w) \equiv s_n \equiv s$$

5) Descrivere due termini diversi del λ-calcolo, descritto in §1.4.1, che non sono convergenti, sempre rispetto a una strategia di riduzione deterministica.

Soluzione

Per definizione un termine t diverge (è non convergente), rispetto a una strategia di riduzione deterministica, se esiste una quantità numerabile di termini s_i al variare di i $\in$ Nat tale che $s_1 \equiv t$ e $s_i \to_1 s_{i+1}$, per ogni i $\in$ Nat (ossia esiste una lista infinita di passi computazioni a partire da t). Prendendo in considerazione una strategia di riduzione deterministica *call-by value*, usata per la semantica nei linguaggi di programmazione, allora un esempio di due termine diversi del λ-calcolo che non sono convergenti è:

$$(\lambda x\ x(x))(\lambda y\ y(y)) \to_1 (\lambda y\ y(y))(\lambda y\ y(y)) \to_1 \ldots \to_1 (\lambda y\ y(y))(\lambda y\ y(y)) \to_1 \ldots$$

Si riduce sempre a se stessa, a qualunque passo di computazione. Perciò ammette computazione infinita (diverge) non raggiungendo mai un valore finale.

6) Che relazione c'è tra il λ-calcolo puro con le regole di riduzione date in §1.4.1, rispetto a quello in cui, adottando la stessa sintassi di termini, imponiamo la seguente definizione di riduzione $\to_1^*$.

- per ogni termine t e b $(\lambda x.t)(b) \to_1^* t[\frac{x}{b}]$
- per ogni termine b, b_1, b_2 e a, a_1, a_2

$$R_I \, \frac{a_1 \to_1^* a_2 \qquad b_1 \to_1^* b_2}{a_1(b_1) \to_1^* a_2(b_2)} \qquad\qquad R_{II} \, \frac{t_1 \to_1^* t_2}{\lambda x.t_1 \to_1^* \lambda x.t_2}$$

Soluzione

Idea: devo provare che relazione esiste tra $\to_1^*$ e $\to$. Per cui verifico cosa accade per ($\to_1^* \subseteq \to_1$) e ($\to_1 \subseteq \to_1^*$)

Sia L(T) l'insieme dei λ termini che è possibile ridurre in forma normale, con la strategia di riduzione T. Dimostro che valgono le seguenti relazioni tra $\to_1^*$ *e* $\to$:

1. $L(\to_1^*) \nsubseteq L(\to_1)$
2. $L(\to_1^*) \subset L(\to_1)$

1. Si ha il λ termine $t \equiv ((\lambda x.x)z)((\lambda y.y)w)$
   Allora applicando la strategia di riduzione $\to$, ottengo

$$\frac{\lambda x.x \to z}{((\lambda x.x)z)((\lambda y.y)w) \to z((\lambda y.y)w)}$$

$$\frac{(\lambda y.y)w \to w}{z((\lambda y.y)w) \to z(w)}$$

   Dunque riesco a giungere a una forma normale.
   Cosa che non è possibile con la strategia $\to_1^*$. In quanto $((\lambda x.x)z)((\lambda y.y)w) \to_1^* z((\lambda y.y)w)$ che non è in forma normale. Per cui $(((\lambda x.x)z)((\lambda y.y)w) \not\to_1^*$.
   In conclusione risulta vero che $L(\to_1^*) \nsubseteq L(\to_1)$.

2. Per provare l'inclusione di $L(\to_1^*)$ in $L(\to_1)$ basta che dimostro, per una valutazione che usa entrambe le strategie, il sempre possibile rimpiazzo della regola R_I con la sua regola corrispondente in $\to_1$ (A_I + A_{II}) . Più formalmente significa provare che per ogni valutazione di un termine M, che usa la regola R_I, si può sempre ottenere una valutazione che usa solo regole della strategia $\to_1$.
   Per farlo procedo per induzione sul numero di volte n che la regola R_I viene utilizzata durante la valutazione di M.

   - (n=0): caso base. La regola R_I non viene mai utilizzata nella valutazione di M, e dunque M è già implicitamente dimostrato, usando l'ipotesi induttiva, con la strategia $\to_1$.
   - (n $\to$ n+1): caso induttivo. Premesse:
     (a) per n risulta vero che $L(\to_1^*) \subset L(\to_1)$, devo provare che vale anche per n+1;

(b) inoltre la valutazione di M utilizza almeno una volta la regola R_I;

(c) per ipotesi induttuva, esiste almeno una valutazione di M con solo regole della strategia $\rightarrow_1$.

Dunque ho M $\rightarrow ... \xrightarrow{R_I}{}^{*}_{1}$ M^I e voglio costruire una derivazione M $\rightarrow ... \xrightarrow{A_I}_1$ $M_1 \xrightarrow{A_{II}}_1 M^I$.
Le strategie di valutazione sono deterministiche, portano pertanto allo stesso risultato della sequenza di derivazione, per cui $R_I = A_I + A_{II}$ risulta vero. Inoltre, per ipotesi induttiva, è sempre possibile avere una valutazione con l'utilizzo di solo regole $\rightarrow_1$ ($A_I + A_{II}$ esistono). Perciò risulta corretto rimpiazzare $\rightarrow_1^*$ con $\rightarrow_1$.
In conclusione risulta vero che $L(\rightarrow_1^*) \subset L(\rightarrow_1)$.

Capitolo 2

Regole della teoria dei tipi

Lo scopo della teoria dei tipi è offrire un sistema formale in cui derivare, tramite regole e assiomi, giudizi nella forma:

$$A\ type[\Gamma] \qquad A = B\ type[\Gamma] \qquad a \in A\ [\Gamma] \qquad a = b \in A\ [\Gamma]$$

$$+\ ausiliaria \quad \Gamma\ cont$$

L'ultimo giudizio non è necessario, serve esclusivamente per imparare.
Quando si formula una nuova teoria dei tipi è bene impiegare il minor numero possibile di regole strutturali e di formazione di tipi e termini. Tali regole devono essere rivolte all'ottimizzazione e correttezza della teoria. Alcune di queste, come quelle di indebolimento e sostituzione in §2.1.5, sono irrinunciabili, la cui validità è sempre garantita e utilizzate nella derivazione di ogni teoria.

Se la teoria dei tipi è dipendente si ha bisogno di tutti i giudizi. Invece in una teoria dei tipi non dipendente, come quella dei tipi semplici, il giudizio $A = B\ type[\Gamma]$ può venire omesso.

2.1 Regole strutturali

Assioma unico: [ ] cont

Nel calcolo dei sequenti, in Logica classica, le derivazioni di giudizio, valide in una teoria dei tipi con solo le regole singoletto, diventano derivazioni di sequenti nella forma $\Gamma \dashv \mathrm{A}$ e unico assioma $\varphi \dashv \varphi$.

Di seguito illustro le principali regole di contesto comuni a tutte le teorie dei tipi.

2.1.1 Regole di formazione dei contesti

$$[\,]\ \text{cont} \qquad \text{dove} \qquad [\,] = \varnothing$$

$$\text{F-c)}\ \frac{\mathrm{A\ type}[\Gamma]}{\Gamma,\ \mathrm{x} \in \mathrm{A}}\ (\mathrm{x} \in \mathrm{A}) \notin \Gamma$$

2.1.2 Regole di assunzione delle variabili

$$\text{var)}\ \frac{\Gamma,\ \text{x} \in \text{A},\ \Delta\ \text{cont}}{\text{x} \in \text{A}[\Gamma,\text{x} \in \text{A},\ \Delta]}$$

2.1.3 Regole strutturali addizionali sull'uguaglianza

L'uguaglianza, in una teoria dei tipi, consiste in una relazione di equivalenza sia fra tipi che fra termini. Sono perciò valide le seguenti regole di uguaglianza tra tipi:

$$ref)\ \frac{A\ type[\Gamma]}{A = A\ type[\Gamma]} \qquad sym)\ \frac{A = B\ type[\Gamma]}{B = A\ type[\Gamma]}$$

$$tra)\ \frac{A = B\ type[\Gamma] \qquad B = C\ type[\Gamma]}{A = C\ type[\Gamma]}$$

E allo stesso modo anche le regole di uguaglianza definizionale/computazionale tra termini:

$$ref)\ \frac{a \in A\ [\Gamma]}{a = a \in A\ [\Gamma]} \qquad sym)\ \frac{a = b \in A\ [\Gamma]}{b = a \in A\ [\Gamma]}$$

$$tra)\ \frac{a = b \in A\ [\Gamma] \qquad b = c \in A\ [\Gamma]}{a = c \in A\ [\Gamma]}$$

2.1.4 Regole di conversione dell'uguaglianza per tipi uguali

L'appartenenza si conserva con l'uguaglianza di termini e tipi. Le regole da aggiungere, in una teoria dei tipi, per garantirlo sono:

$$conv)\ \frac{a \in A\ [\Gamma] \quad A = B\ type[\Gamma]}{a \in B\ [\Gamma]}$$

$$conv-eq)\ \frac{a = b \in A\ [\Gamma] \qquad A = B\ type[\Gamma]}{a = b \in B\ [\Gamma]}$$

2.1.5 Regole di indebolimento e di sostituzione

Indebolimento

$$ind-ty)\ \frac{A\ type[\Gamma] \quad \Gamma,\Delta\ cont}{A\ type[\Gamma,\Delta]} \qquad ind-ty-eq)\ \frac{A = B\ type[\Gamma] \quad \Gamma,\Delta\ cont}{A = B\ type[\Gamma,\Delta]}$$

$$ind-te)\ \frac{a \in A\ [\Gamma] \quad \Gamma,\Delta\ cont}{a \in A\ [\Gamma,\Delta]} \qquad ind-te-eq)\ \frac{a = b \in A\ [\Gamma] \quad \Gamma,\Delta\ cont}{a = b \in A\ [\Gamma,\Delta]}$$

Sostituzione

$$\begin{array}{c} C(x_1,...,x_n)\ type[\Gamma, x_1 \in A_1,...,x_n \in A_n(x_1,...,x_{n-1})] \\ sub-typ)\ \dfrac{a_1 \in A_1\ [\Gamma]\ ...a_n \in A_n(a_1,...,a_{n-1})\ [\Gamma]}{C(a_1,...,a_n)\ type[\Gamma]} \end{array}$$

$$\begin{array}{c} C(x_1,...,x_n)\ type[\Gamma, x_1 \in A_1,...,x_n \in A_n(x_1,...,x_{n-1})] \\ sub-eq-typ)\ \dfrac{a_1 = b_1 \in A_1\ [\Gamma]\ ...a_n = b_n \in A_n(a_1,...,a_{n-1})\ [\Gamma]}{C(a_1,...,a_n) = C(b_1,...,b_n)\ type[\Gamma]} \end{array}$$

$$C(x_1,...,x_n) = D(x_1,...,x_n)\ type[\Gamma, x_1 \in A_1, ..., x_n \in A_n(x_1,...,x_{n-1})]$$
$$sub-Eqtyp)\ \ \frac{a_1 \in A_1\ [\Gamma]\ ...a_n \in A_n(a_1,...,a_{n-1})\ [\Gamma]}{C(a_1,...,a_n) = D(a_1,...,a_n)\ type[\Gamma]}$$

$$C(x_1,...,x_n) = D(x_1,...,x_n)\ type[\Gamma, x_1 \in A_1, ..., x_n \in A_n(x_1,...,x_{n-1})]$$
$$sub-eq-Eqtyp)\ \ \frac{a_1 = b_1 \in A_1\ [\Gamma]\ ...a_n = b_n \in A_n(a_1,...,a_{n-1})\ [\Gamma]}{C(a_1,...,a_n) = D(b_1,...,b_n)\ type[\Gamma]}$$

$$c(x_1,...,x_n) \in C(x_1,...,x_n)\ [\Gamma, x_1 \in A_1, ..., x_n \in A_n(x_1,...,x_{n-1})]$$
$$sub-ter)\ \ \frac{a_1 \in A_1\ type[\Gamma]\ ...a_n \in A_n(a_1,...,a_{n-1})\ [\Gamma]}{c(a_1,...,a_n) \in C(a_1,...,a_n)\ type[\Gamma]}$$

$$c(x_1,...,x_n) = d(x_1,...,x_n) \in C(x_1,...,x_n)\ [\Gamma, x_1 \in A_1, ..., x_n \in A_n(x_1,...,x_{n-1})]$$
$$sub-eqter)\ \ \frac{a_1 \in A_1\ [\Gamma]\ ...a_n \in A_n(a_1,...,a_{n-1})\ [\Gamma]}{c(a_1,...,a_n) = d(a_1,...,a_n) \in C(a_1,...,a_n)\ [\Gamma]}$$

$$c(x_1,...,x_n) \in C(x_1,...,x_n)\ [\Gamma, x_1 \in A_1, ..., x_n \in A_n(x_1,...,x_{n-1})]$$
$$sub-eq-ter)\ \ \frac{a_1 = b_1 \in A_1\ [\Gamma]\ ...a_n = b_n \in A_n(a_1,...,a_{n-1})\ [\Gamma]}{c(a_1,...,a_n) = c(b_1,...,b_n) \in C(a_1,...,a_n)\ [\Gamma]}$$

$$c(x_1,...,x_n) = d(x_1,...,x_n) \in C(x_1,...,x_n)\ [\Gamma, x_1 \in A_1, ..., x_n \in A_n(x_1,...,x_{n-1})]$$
$$sub-eq-eqter)\ \ \frac{a_1 = b_1 \in A_1\ [\Gamma]\ ...a_n = b_n \in A_n(a_1,...,a_{n-1})\ [\Gamma]}{c(a_1,...,a_n) = d(b_1,...,b_n) \in C(a_1,...,a_n)\ [\Gamma]}$$

2.1.6 Regole proprie e derivabili

In una teoria formale ci sono due tipi di regole:

- **regole proprie del calcolo**, come lo sono le regole strutturali e quelle del singoletto;

- **regole derivabili**, come le regole di sostituzione, utili per abbreviare le derivazioni.

Una regola r) $\frac{J_1,...,J_n}{J}$ è ammissibile in un calcolo t sse i giudizi derivabili in $t+r$ sono gli stessi dei giudizi derivabili in t. Ciò comporta che l'aggiunta di una regola rt non cambia i giudizi che ne possono derivare.
Quando un assioma è ammissibili e derivabile questo coincide con un giudizio derivabile.

2.1.7 Nozione di contesto telescopico

Un giudizio, in teoria dei tipi dipendenti si esprime nella forma

$$\mathbf{A(x_1,...,x_n)[x_1 \in B_1,...,x_n \in B_n]}$$

e prende il nome di **contesto telescopico**. Questi presenta una dipendenza continua, esemplificata nel seguente giudizio

$$A(x_1, x_2, x_3)\ type[x \in C_1, x_2 \in C(x_1), x_3 \in C(x_1 x_2)..)$$

Si parla di contesti rigidi, ovvero senza possibilità di scambio. Come appare dall'esempio sotto.
$[x \in Nat, y \in Nat, z \in Mat(x, y)]$ cont $\Rightarrow$ **è derivabile**
$[y \in Nat, x \in Nat, z \in Mat(x, y)]$ cont $\Rightarrow$ **è derivabile**
$[y \in Nat, z \in Mat(x, y), x \in Nat]$ $\Rightarrow$ **non è un contesto**.

Per cui non esiste lo scambio arbitrario, e si deve porre attenzione alle dipendenza delle assunzioni, che provoca una sostituzione rigida.

2.1.8 Esempi di applicazione delle regole di sostituzione

Nota: attenzione all'ordine di sostituzione, si deve partire sempre da quello con meno dipendenze.

$$\text{sost}\ \frac{\text{c} \in [\text{C}, \Gamma] \qquad \text{b} \in [\text{B(c)}, \Gamma] \qquad \text{A(x,y) type}[\text{x} \in \text{C, y} \in \text{B(x)}]}{\text{A(c,b) type}[\Gamma]}$$

$$\text{sost}\ \frac{\text{c} \in [\text{C},\Gamma] \qquad \text{b} \in [\text{B(c)},\Gamma] \qquad \text{a(x,y)} \in \text{A(x,y)}[\text{x} \in \text{C, y} \in \text{B(x)}]}{\text{a(c,b)} \in \text{A(c,b)}\ [\Gamma]}$$

Se si ha un giudizio di tipo può venire usato il giudizio di uguaglianza tra termini e la sostituzione.

$$\text{sost-eq}\ \frac{\text{A(x) type}[\Gamma, \text{x} \in \text{C}] \qquad \text{c} = \text{e} \in \text{C}[\Gamma]}{\text{A(c)} = \text{A(e) type}[\Gamma]}$$

È dunque fondamentale, tra tipi dipendenti, il concetto di uguaglianza fra tipi. Allo stesso modo posso considerare il caso in cui si ha un elemento

$$\text{sost-eq}\ \frac{\text{a(x)} \in \text{A(x)}\ [\Gamma, \text{x} \in \text{C}] \qquad \text{c} = \text{e} \in \text{C}[\Gamma]}{\text{a(c)} = \text{a(e)} \in \text{A(c)}\ [\Gamma]}$$

Per poter affermare che A(e) = A(c) devo poterlo dedurre. Per farlo mi sono indispensabili le **regole di Conversione dell'uguaglianza del tipaggio (§2.1.9)**.

2.1.9 Regole di tipaggio

Regole di Conversione

$$conv)\frac{c \in C[\Gamma] \qquad C = D\ type[\Gamma]}{c \in D\ [\Gamma]}$$

Se due tipi sono uguali allora hanno gli stessi termini: $C = D \Rightarrow (c \in C \Leftrightarrow c \in D)$. L'uguaglianza fra tipi è per questo simmetrica.
Tuttavia non sempre l'unicità del tipaggio di un termine (il $\Leftrightarrow$) è garantito per ogni teoria. Nei casi trattati dal corso sì, in quanto verrà inteso che $C = D\ type[\Gamma]$ sse due tipi hanno gli stessi elementi (come già accade in *set theory*), ma può non essere sempre vero.

Regole di Conversione dell'uguaglianza

$$conv - eq)\frac{c = d \in C[\Gamma] \qquad C = D\ type[\Gamma]}{c = d \in D\ [\Gamma]}$$

Questa regola permette di convertire le uguaglianze nel tipaggio di un termine.

2.2 Il tipo singoletto

Il tipo singoletto risulta essere paradigmatico per gli altri tipi. Per definirlo impiegherò i giudizi nella forma $A\ type[\Gamma]$, $a \in A\ [\Gamma]$ e $a = b \in A\ [\Gamma]$. L'uguaglianza, invece, non verrà coinvolta, in quanto non può essere impiegata per definire un nuovo tipo, è difatti usata solo nelle derivazioni.

Innanzitutto come già visto in §2.1.1 ogni derivazione parte sempre dal contesto vuoto ([ ]).

$$\text{F-c)}\ \frac{\text{A type}[\Gamma]}{\Gamma,\ \text{x} \in \text{A}}\ (\text{x} \in \text{A}) \notin \Gamma$$

2.2.1 Regola di Formazione

$$\text{F-S)}\ \frac{\Gamma\ \text{cont}}{N_1\ \text{type}[\Gamma]}$$

La regola (F-S) permette di derivare vari giudizi e di dire che cosa è un tipo.

Con l'impiego solo delle regole *(F-c)* e *(F-S)* si possono derivare $[x_1 \in N_1 ... \ x_n \in N_1]$ cont e ottenere, così, contesti di una lista arbitraria di variabili diverse, appartenenti a N_1, come si vede dall'esempio seguente.

$$\text{F-c}\ \cfrac{\text{F-S}\ \cfrac{\text{F-c}\ \cfrac{[\ ]\ \text{cont}}{[x_1 \in N_1]\ \text{cont}}\ (x_1 \in N_1) \notin [\ ]}{N_1\ \text{type}\ [x_1 \in N_1]}}{[x_1 \in N_1,\ x_2 \in N_1]\ \text{cont}}\ (x_2 \in N_1) \notin x_1 \in N_1$$

2.2.2 Regole di Introduzione

$$\text{I-S)}\ \frac{\Gamma \text{ cont}}{* \in N_1\ [\Gamma]}$$

Sia N_1 in ogni contesto Γ, partendo da contesto [ ], la regola (I-S) permette di formare i termini, per mezzo dell'introduzione di un elemento costante $*$ in N_1.

Un esempio diretto della sua applicazione è

$$\text{I-S)}\ \frac{x_1 \in N_1 ... x_n \in N_1 \text{ cont}}{* \in N_1\ [x_1 \in N_1 ... x_n \in N_1]}$$

2.2.3 Regole di Eliminazione

$$\text{E-S)}\ \frac{\text{t} \in N_1\ [\Gamma] \qquad \text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{c} \in \text{M}(*)[\Gamma]}{El_{N1}(\text{t, c}) \in \text{M}(*)[\Gamma]}$$

El trattasi di costruttore di funzioni e $M[t] = M(z)[\frac{z}{t}]$.

2.2.4 Regole di Conversione

$$\text{C-S)}\ \frac{\text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{c} \in \text{M}(*)[\Gamma]}{El_{N1}(*, \text{c}) = \text{c} \in \text{M}(*)[\Gamma]}$$

La conversione rende possibile l'applicazione della regola di eliminazione (E-S) introducendo delle uguaglianze.

2.2.5 Regole di Uguaglianza

$$\text{eq-E-N}_1)\ \frac{\text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{t}_1 = \text{t}_2 \in N_1[\Gamma] \qquad \text{c}_1 = \text{c}_2 \in \text{M}(*)[\Gamma]}{\text{El}_{N1}(\text{t}_1, \text{c}_1) = \text{El}_{N1}(\text{t}_2, \text{c}_2) \in \text{M}(t_1)[\Gamma]}$$

Le regole *(S), (I-S), (E-S)* e *(C-S)* hanno una spiegazione computazionale, e riguardano la compatibilità tra tipi, ma non da caratterizzare il tipo dei tipi. Inoltre il tipo singoletto non è dipendente.

2.2.6 Eliminatore dipendente

La regola di eliminazione si può equivalentemente scrivere in un altro modo

$$\text{E-S}_{dip})\ \frac{\text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{c} \in \text{M}(*)[\Gamma]}{El_{N1}(\text{z, c}) \in \text{M(z)}[\Gamma, \text{z} \in N_1]}$$

Le regole *(E-S$_{dip}$) + (le regole di sostituzione) + (F-S) + (I-S)* permettono di verificare la validità di *(E-S)*.

$$\text{sost)}\ \frac{\text{t} \in N_1[\Gamma] \qquad \text{E-S}_{dip})\ \dfrac{\text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{c} \in \text{M}(*)[\Gamma]}{El_{N1}(\text{z, c}) \in \text{M(z)}[\Gamma]}}{El_{N1}(\text{t, c}) \in \text{M(t)}[\Gamma]}$$

Inoltre vale anche il viceversa, da *(E-S)* si riesce a ottenere *(E-S$_{dip}$)*.

2.2.7 Osservazioni sul tipo singoletto

L'eliminatore El_{N1}(z, c) rappresenta una funzione definita per ricorsione su N_1, difatti in *(C-S)* si ha che $\mathrm{El}_{N1}(z, c)[\frac{z}{*}] = \mathrm{El}_{N1}(*, c)$.
Supposto che se $* \in \mathrm{N}_1[\Gamma]$ in *(E-S)*, allora per la singola conversione vale che $\mathrm{El}_{N1}(*,c) = c \in M(*)$. Dunque El_{N1}(z, c) rappresenta un programa funzionale per ricorsione. Questi è definito su N_1, a partire da $c \in M(*)$, perciò $\mathrm{El}_{N1}(*, c) = c$.
La regola di eliminazione permette di definire un programma funzionale da N_1 a M(z) esclusivamente con $c \in M(*)$, ovvero definendo $*$ come **elemento canonico**. Inoltre non svolge solo il compito di ricorsione, ma anche d'induzione.

In

$$\frac{t \in \mathrm{N}_1[\Gamma]}{t = * \in \mathrm{N}_1[\Gamma]}$$

risulta vera l'uguaglianza definizionale?
No, non è vera. La regola di eliminazione consente di dare un valore al termine canonico, permettendo così di attribuire un valore a tutti i possibili termini del singoletto. Ma in generale questo non vale all'interno della teoria. Difatti l'uguaglianza definizionale è diversa da quella matematica e va intesa come computazionale e non proposizionale (come definito in §1.7.2).

2.3 Sanitary checks rules

Le *Sanitary checks* sono regole strutturali utili per abbreviare le derivazioni. Queste sono derivabili solo se la teoria dei tipi è corretta.

Assumendo *T*, come una teoria dei tipi di riferimento, le *Sanitary checks* sono le seguenti:

$$\frac{[\Gamma, \Delta]\ \mathrm{cont}}{\Gamma\ \mathrm{cont}}$$

Se **[Γ, Δ] cont** è derivabile in *T* allora anche **[Γ] cont** è derivabile in *T*.

$$\frac{\mathrm{J}_1,\ldots,\mathrm{J}_n}{\mathrm{J}}$$

Se $\mathbf{J}_i$ **con i = 1,...,n** in *T* sono derivabili allora anche **J** è derivabile in *T*.

$$\frac{A\ \mathrm{type}\ [\Gamma]}{\Gamma\ \mathrm{cont}}$$

Se **A type [Γ]** è derivabile in *T* allora anche Γ **cont** è derivabile in *T*.

$$\frac{A = B\ \mathrm{type}\ [\Gamma]}{A\ \mathrm{type}[\Gamma]} \quad \frac{A = B\ \mathrm{type}\ [\Gamma]}{B\ \mathrm{type}[\Gamma]}$$

Se **A = B type [Γ]** è derivabile in *T* allora anche **A type[Γ]** e **B type[Γ]** sono derivabili in *T*.

$$\frac{\mathrm{a} \in \mathrm{A}[\Gamma]}{\mathrm{A\ type}[\Gamma]}$$

Se $\mathbf{a} \in \mathbf{A}[\Gamma]$ è derivabile in T allora anche $\mathbf{A\ type}[\Gamma]$ è derivabile in T.

$$\frac{\mathrm{a} = \mathrm{b} \in \mathrm{A}[\Gamma]}{\mathrm{a} \in \mathrm{A}[\Gamma]} \quad \frac{\mathrm{a} = \mathrm{b} \in \mathrm{A}[\Gamma]}{\mathrm{b} \in \mathrm{A}[\Gamma]}$$

Se $\mathbf{a} = \mathbf{b} \in \mathbf{A}[\Gamma]$ è derivabile in T allora anche $\mathbf{a} \in \mathbf{A}[\Gamma]$ e $\mathbf{b} \in \mathbf{A}[\Gamma]$ sono derivabili in T.

2.4 Schema generale

Di seguito illustro uno schema generale, valido per ogni teoria dei tipi, di produzione di regole definenti un tipo, i suoi termini e l'uguaglianza.

1. **Regole di Formazione:** identificate con il preambolo F-T, con T teoria dei tipi in esame e t' elemento canonico.
   Tali regole rispettano la forma $\dfrac{[\Gamma]\ \mathrm{cont}}{T\ \mathrm{type}[\Gamma]}$

2. **Regole di Introduzione:** identificate con il preambolo I-T, con T teoria dei tipi in esame e t' elemento canonico.
   Tali regole consistono nella forma introduttiva degli elementi canonici di T e rispettano la forma $\dfrac{[\Gamma]\ \mathrm{cont}}{\mathrm{t}' \in T[\Gamma]}$

3. **Regole di Eliminazione:** identificate con il preambolo E-T con T teoria dei tipi in esame e t' elemento canonico.
   Tali regole sono definenti E_T, a partire dagli elementi di T a valori in un tipo M(z) type[Γ, z $\in T$]. L'ipotesi valida è che siano dati degli elementi in M(z) sui valori canonici di T. Tali regole rispettano la forma

   $$\frac{\mathrm{t} \in T[\Gamma] \qquad \mathrm{M(z)\ type}[\Gamma,\ \mathrm{z} \in T] \qquad \mathrm{c} \in \mathrm{M(t')}[\Gamma]}{El_T(\mathrm{t},\ \mathrm{c}) \in \mathrm{M(t)}[\Gamma]}$$

4. **Regole di Conversione:** identificate con il preambolo C-T, con T teoria dei tipi in esame e t' elemento canonico.
   , che stabiliscono che gli eliminatori in (3) sono stabiliti per ricorsione a partire dalle ipotesi. Tali regole rispettano la forma

   $$\frac{\mathrm{M(z)\ type}[\Gamma,\ \mathrm{z} \in T] \qquad \mathrm{c} \in \mathrm{M(t')}[\Gamma]}{\mathrm{El}_T(\mathrm{t'},\ \mathrm{c}) = \mathrm{c} \in \mathrm{M(t')}[\Gamma]}$$

5. **Regole di Uguaglianza:** identificate con il preambolo eq-E-T, con T teoria dei tipi in esame e t' elemento canonico.
   Tali regole stabiliscono che i costruttori di T in (2) e (3) permettono l'uguaglianza definizionale dei termini da cui dipendono. Tali regole rispettano la forma

   $$\frac{\mathrm{M(z)\ type}[\Gamma,\ \mathrm{z} \in T] \qquad \mathrm{t} = \mathrm{s} \in T[\Gamma] \qquad \mathrm{c} = \mathrm{d} \in \mathrm{M(t')}[\Gamma]}{\mathrm{El}_T(\mathrm{t},\ \mathrm{c}) = \mathrm{El}_T(\mathrm{s},\ \mathrm{d}) \in \mathrm{M(t)}[\Gamma]}$$

Le regole (5) sono implicite in T, ma non ovvie dal punto di vista computazionale.

2.5 Uguaglianza definizionale

Definizione: se P_1 e P_2 programmi

$P_1,\ P_2$: $Nat^m \rightarrow Nat$ allora $P_1 = P_2$ *sse* $\forall\ n_1...n_m$ *e* $P_1(n_1...n_m) = P_2(n_1...n_m)$. Ma questo non è decidibile.

Ovvero le funzioni ricorsive per P_1 e P_2 non sono decidibili. A seguito di ciò non esiste un algoritmo in grado di decidere se due programmi P_1 e P_2 sono estensionalmente (proposizionalmente) uguali o meno.

Risulta però vero il concetto di **uguaglianza definizionale**/computazionale (in teoria dei tipi intensionali).
Dati $a \in A[\Gamma]$ e $b \in A[\Gamma]$ derivabili, nella nostra teoria T di *Martin-Löf*, esiste un algoritmo H $(a \in A[\Gamma], b \in A[\Gamma]) = \begin{cases} \textbf{sì} \text{ sse } a = b \in A[\Gamma] \text{ è derivabile in } T \\ \textbf{no} \text{ sse } a = b \in A[\Gamma] \text{ non ha derivazione in } T \end{cases}$
Il giudizio $a = b \in A[\Gamma]$ è decidibile, anche con J giudizio, in teoria dei tipi di *Martin-Löf*, derivabile. Per cui con H si scrive: Giudizi di $T \rightarrow \{0,1\}$
$H[J] = \begin{cases} \mathbf{1} \text{ sse } J \text{ è derivabile in } T \\ \mathbf{0} \text{ sse } J \text{ non è derivabile in } T \end{cases}$
H lavora come un *proof-assistant* (esempio *Coq*).

2.5.1 Applicazione dell'uguaglianza definizionale tra termini

Definizione: i termini *untyped* sono

$$t = x \mid * \mid El_{N1}(t_1,\, t_2)$$

Definizione: relazione di riduzione

$$\rightarrow_1 \subseteq \text{term x term}$$

$t_1 \rightarrow_1 t_2 \equiv t_1$ si riduce in un passo di computazione a t_2. Ecco che esiste una relazione che computa t_1 con t_2.
Le **relazioni di computazione**, dell'uguaglianza definizionale, le ho descritte in §2.6.

Definizione: t termine *untyped* è in forma normale (in NF) sse non esiste termine s tale che $t \rightarrow_1 s$. Ovvero non è più riducibile a nulla.
Le forme normali sono i valori assumibili dai programmi.

Definizione:**Teoria di Validità**
Dati $t \in A[\Gamma]$ e $s \in A[\Gamma]$ in T_1 allora, $t \rightarrow_1 s \Rightarrow t = s \in A[\Gamma]$ derivabile in T.

termini	termini in forma canonica	termini non in forma canonica o introduttiva
termini in NF	$*$	x, $El_{N1}(x, *)$
termini non in NF	/	$El_{N1}(*, x)$

Tabella 2.1: Termini a $\rightarrow_1$ di N_1.

I termini non in forma canonica derivano dalle regole di introduzione; invece quelli non in forma canonica vengono introdotte dall'eliminatore.

La chiusura riflessiva, simmetrica e transitiva delle derivazioni è proprio l'uguaglianza definizionale. Tale proprietà non vale esclusivamente per $\rightarrow_1$ ma per qualsiasi combinazione delle riduzioni in §2.6, con variazione però delle forma normali (che non sono altro che i risultati dei nostri programmi, derivanti dai diversi cammini di computazione).
Un esempio significativo di applicazione di strategie di computazione l'ho riportato in §2.7, esercizio 4. Utile per comprendere a cosa serve la relazione $\rightarrow_1$ e che ogni teoria $T \rightarrow_1$ non è deterministica.

Le definizioni seguenti sono definite sulla regole di semantica operazionale.

Definizione **Riducibilità**
Dati i termini t e s allora **t** ***Red***$_{NF}$ **s** sse s è in NF ed esistono $h_1...h_n$ (n>=1) tale che $h_1 \equiv t$, $h_n \equiv s$ e se n>1 $h_i \rightarrow h_{i+1}$ per i = 1 a n-1.

$$\mathbf{t}\ \boldsymbol{Red}_{NF}\ \mathbf{s} = \begin{cases} \mathbf{t} \equiv \text{s e t è in NF} \\ \text{esiste n>1, esistono } h_1...h_n \text{ termini tale che } \mathbf{t} \equiv h_1 \rightarrow_1 h_2...h_n \end{cases}$$

Definizione **Teorema di Confluenza**$\rightarrow_1$ per T computabile
Dato t (termine) e s_1 e s_2 (in NF) tale che **t** ***Red***$_{NF}$ **s_1** e **t** ***Red***$_{NF}$ **s_2** allora $s_1 \equiv s_2$ (coincide a meno di rinomina di variabile vincolante).
Quando t si riduce s_1 e in s_2 c'è l'unicità della forma normale.

Definizione **Teorema della forma normale (debole)**
Dato t termine della grammatica, esiste s termine in NF, tale che **t** ***Red***$_{NF}$ **s**. Allora esistono $t \equiv h_1 \rightarrow_1 ... \rightarrow_1 h_n$ con n>1 se t non è già in NF; oppure $t \equiv s$ se t è già in NF.
Questo significa che è sempre possibile rendere un programma convergente. Ma si può dire di più: $\nexists$ programmi che divergono.

Definizione **Teorema della forma normale (forte)**
Per ogni termine t, l'albero dei cammini di riduzione di t è ben formato (ovvero $\nexists$ un cammino di riduzioni $\rightarrow_1$ infinito).
In questo modo ogni strategia deterministica è convergente.
Con quanto appena enunciato sopra possiamo definire quanto segue.
Dato $t \in A[\Gamma]$ derivabile in T

$$NF(t_1) \equiv \begin{cases} \mathbf{t} \text{ se t è in NF} \\ \mathbf{s} \text{ se } \boldsymbol{t}\ \boldsymbol{Red}_{NF}\ \boldsymbol{s} \end{cases}$$

Dunque se t non é in NF, per il teorema normale, comunque esiste una riduzione in NF.
Sono così in grado di dimostrare che, dati a $\in$ A[Γ] e b $\in$ A[Γ], giudizi derivabili in T_i allora a = b $\in$ A[Γ] sse NF(a) $\equiv$ NF(b) sse

1. a e b sono in NF e quindi a $\equiv$ b
2. a non in NF, b in NF e **a** $\boldsymbol{Red_{NF}}$ **b**
3. a in NF, b non in NF e **b** $\boldsymbol{Red_{NF}}$ **a**
4. né a né b sono in NF esiste s in NF tale che **a** $\boldsymbol{Red_{NF}}$ **s** e **b** $\boldsymbol{Red_{NF}}$ **s**

Per i punti elencanti sopra trova validità la relazione **a** $\boldsymbol{Red_{NF}}$ **NF(a)** $\Rightarrow$ a = NF(a) $\in$ A[Γ] è derivabile (la forma normale è uguale al termine stesso).
Questo rende l'uguaglianza computabile, si è difatti in grado di dimostrare che esiste P programma tale che P(a) = NF(a), per ogni a termine *untyped* in T(incluso T_1).

In conclusione la computabilità dell'uguaglianza (definizionale) tra due termini, si riduce a computare le forme normali del primo termine con quelle del secondo e a verificare se sono identicamente la stessa (a meno di rinomina di variabili).

2.6 Semantica operazionale del singoletto

La relazione $\to_1$ viene definita all'interno dei termini con l'uso delle seguenti regole di riduzione:

- β_{N1}-red) $\mathrm{El}_{N1}(*, \mathrm{t}) \to_1 \mathrm{t}$
- red$_I$) $\dfrac{\mathrm{t}_1 \to_1 \mathrm{t}_2}{\mathrm{El}_{N1}(\mathrm{t}_1, \mathrm{c}) \to_1 \mathrm{El}_{N1}(\mathrm{t}_2, \mathrm{c})}$ red$_{II}$) $\dfrac{\mathrm{c}_1 \to_1 \mathrm{c}_2}{\mathrm{El}_{N1}(\mathrm{t}, \mathrm{c}_1) \to_1 \mathrm{El}_{N1}(\mathrm{t}, \mathrm{c}_2)}$
- red$_I$ e red$_{II}$ possono venire simulate da un'unica regola $\dfrac{\mathrm{t}_1 \to_1 \mathrm{t}_2 \qquad \mathrm{c}_1 \to_1 \mathrm{c}_2}{\mathrm{El}_{N1}(\mathrm{t}_1, \mathrm{c}_1) \to_1 \mathrm{El}_{N1}(\mathrm{t}_2, \mathrm{c}_2)}$

β_{N1}-*red* risulta valida per *(C-S)*, le regole di riduzione red$_I$ e red$_{II}$ per *(eq-E-S)*.

2.7 Esercizi

1) Data

$$\text{E-N}_{1prog})\ \frac{\mathrm{M(w)\ type\ [\Gamma,\ w \in N_1]} \qquad \mathrm{d \in M(*)\ type[\Gamma]}}{\mathrm{El}_{N1}(\mathrm{w, d}) \in \mathrm{M(w)\ type[\Gamma,\ w \in N_1]}}$$

dimostrare che in $\boldsymbol{T_1}$ la regola *E-N*$_{1prog}$ è derivabile. Al fine di ciò basta mostrare che se i giudizi premessa sono derivabili, allora lo è anche il giudizio di conclusione.

Soluzione

Per una maggiore comprensione delle derivazioni, ho ritenuto opportuno, ove necessario, spezzare l'albero in più parti.

In E-N_{1prog} il contesto è $\Sigma \equiv \Gamma$, w $\in$ N_1 che permette l'applicazione di E-S.

$$\text{E-S}\ \dfrac{\text{var}\ \dfrac{\text{s-checks}\ \dfrac{\dfrac{}{M(w)\ type[\Gamma,\ w \in N_1]}}{[\Gamma,\ w \in N_1]\ cont}}{w \in N_1[\Gamma,\ w \in N_1]} \qquad \dfrac{\mathbf{1}}{M(z)\ type[\Gamma,\ w \in N_1,\ z \in N_1]} \qquad \dfrac{}{d \in M(*)[\Gamma,\ w \in N_1]}}{El_{N1}(w,\ d) \in M(w)[\Gamma,\ w \in N_1]}$$

1

$$\text{sub-typ}\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{\text{s-checks}\ \dfrac{\dfrac{}{M(w)\ type[\Gamma,\ w \in N_1]}}{\Gamma,\ w \in N_1\ cont}}{N_1\ type\ [\Gamma,\ w \in N_1]} \quad (z \in N_1) \notin (\Gamma,\ w \in N_1)}{\Gamma,\ w \in N_1,\ z \in N_1\ cont}}{z \in N_1[\Gamma,\ w \in N_1,\ z \in N_1]} \qquad \text{ind-ty}\ \dfrac{\dfrac{}{M(w)\ type[\Gamma,\ w \in N_1]} \quad \text{F-c}\ \dfrac{\text{F-S}\ \dfrac{\text{s-checks}\ \dfrac{\dfrac{}{M(w)\ type[\Gamma,\ w \in N_1]}}{\Gamma,\ w \in N_1\ cont}}{N_1\ type[\Gamma,\ w \in N_1]} \quad (z \in N_1) \notin (\Gamma,\ w \in N_1)}{\Gamma,\ w \in N_1,\ z \in N_1\ cont}}{M(w)\ type[\Gamma,\ w \in N_1,\ z \in N_1]}}{M(z)\ type[\Gamma,\ w \in N_1,\ z \in N_1]}$$

Assumo che le premesse di *E-N_{1prog}* (*M(w) type [Γ,w $\in$ N_1]* e *d $\in$ M($*$)[Γ]*) siano valide, perciò è valido, dalla prova sopra, anche il giudizio di conclusione *El_{N1}(w, d) $\in$ M(w)[Γ, w $\in$ N_1]*, di conseguenza derivabile in T_1.

2) Dimostrare che la regola *E-S* è derivabile in una teoria dei tipi T_1, in cui si è rimpiazzata la regola di eliminazione *E-S* con la regola *E-N_{1prog}*, aggiungendovi le regole di indebolimento, sostituzione e di *Sanitary checks*.

$$\text{E-}N_{1prog})\ \dfrac{M(w)\ type\ [\Gamma,\ w \in N_1] \qquad d \in M(*)[\Gamma]}{El_{N1}(w,\ d) \in M(w)[\Gamma,\ w \in N_1]}$$

$$\text{E-S})\ \dfrac{t' \in N_1[\Gamma] \qquad M(w)\ type\ [\Gamma,\ w \in N_1] \qquad d \in M(*)[\Gamma]}{El_{N1}(t',\ d) \in M(t')[\Gamma]}$$

Soluzione

Idea: parto dalla regola di eliminazione E-S, vi applico la regola di sostituzione sub-typ giungendo così alle premesse di E-N_{1prog}

$$\text{sub-typ}\ \dfrac{\dfrac{}{t' \in N_1[\Gamma]} \qquad \text{E-}N_{1prog}\ \dfrac{\dfrac{}{M(w)\ type\ [\Gamma,\ w \in N_1]} \quad \dfrac{}{d \in D(*)[\Gamma]}}{El_{N1}(w,\ d) \in M(w)[\Gamma,\ w \in N_1]}}{El_{N1}(t',\ d) \in M(t')[\Gamma]}$$

Assumo che siano valide per costruzione le premesse di *E-N_{1prog}* (come dimostro nell'esercizio 1) e di *E-S*.

3) Sia T_1 la teoria dei tipi definita del tipo singoletto con le regole strutturali, definite in questo capitolo, incluse quelle di sostituzione e indebolimento. Allora stabilire se i seguenti termini sono tipabili come termini del tipo singoletto, secondo T_1 e quali sono uguali definizionalmente.

- $\mathrm{El}_{N1}(*, *)$
- $\mathrm{El}_{N1}(\mathrm{x}, *)$
- $\mathrm{El}_{N1}(*, \mathrm{y})$
- $\mathrm{El}_{N1}(\mathrm{x}, \mathrm{y})$
- $\mathrm{El}_{N1}(\mathrm{El}_{N1}(*, \mathrm{y}), \mathrm{El}_{N1}(\mathrm{x}, *))$

Soluzione

Per una maggiore comprensione delle derivazioni, ho ritenuto opportuno, ove necessario, spezzare l'albero in più parti.

1

$$\text{E-S}\ \dfrac{\text{I-S}\ \dfrac{[\ ]\ \mathrm{cont}}{* \in \mathrm{N1}[\ ]} \qquad \text{F-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}[\ ]} \quad (\mathrm{z} \in \mathrm{N}_1) \notin [\ ]}{\mathrm{z} \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{N}_1\ \mathrm{type}[\mathrm{z} \in \mathrm{N}_1]} \qquad \text{I-S}\ \dfrac{[\ ]\ \mathrm{cont}}{* \in \mathrm{N}_1[\ ]}}{\mathrm{El}_{N1}(*, *) \in N_1[\ ]}$$

Applicando la β N_1-*red* allora $\mathrm{El}_{N1}(*, *) \to_1 *$
$\mathrm{El}_{N1}(*, *)$ è uguale definizionalmente.

2

$$\text{E-S}\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}[\ ]} \quad (\mathrm{x} \in \mathrm{N}_1) \notin [\ ]}{\mathrm{x} \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{x} \in \mathrm{N}_1[\mathrm{x} \in \mathrm{N}_1]} \qquad \text{F-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}\ [\ ]} \quad (\mathrm{x} \in \mathrm{N}_1) \notin [\ ]}{\mathrm{x} \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{N}_1\ \mathrm{type}[\mathrm{x} \in \mathrm{N}_1]} \quad (\mathrm{z} \in \mathrm{N}_1) \notin \mathrm{x} \in \mathrm{N}_1}{\mathrm{x} \in \mathrm{N}_1,\ \mathrm{z} \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{N}_1\ \mathrm{type}[\mathrm{x} \in \mathrm{N}_1,\ \mathrm{z} \in \mathrm{N}_1]} \qquad \text{I-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}[\ ]} \quad (\mathrm{x} \in \mathrm{N}_1) \notin [\ ]}{\mathrm{x} \in \mathrm{N}_1\ \mathrm{cont}}}{* \in \mathrm{N}_1[\mathrm{x} \in \mathrm{N}_1]}}{\mathrm{El}_{N1}(\mathrm{x}, *) \in \mathrm{N}_1[\mathrm{x} \in \mathrm{N}_1]}$$

Applicando la β N_1-*red* allora $\mathrm{El}_{N1}(\mathrm{x}, *) \not\to_1$
$\mathrm{El}_{N1}(\mathrm{x}, *)$ non è uguale definizionalmente.

3

$$\text{E-S}\ \dfrac{\text{I-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}[\ ]} \quad (\mathrm{y} \in \mathrm{N}_1) \notin [\ ]}{\mathrm{y} \in \mathrm{N}_1\ \mathrm{cont}}}{* \in \mathrm{N}_1[\mathrm{y} \in \mathrm{N}_1]} \qquad \text{F-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}[\ ]} \quad (\mathrm{y} \in \mathrm{N}_1) \notin [\ ]}{\mathrm{y} \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{N}_1\ \mathrm{type}[\mathrm{y} \in \mathrm{N}_1]} \quad (\mathrm{z} \in \mathrm{N}_1) \notin \mathrm{y} \in \mathrm{N}_1}{\mathrm{y} \in \mathrm{N}_1,\ \mathrm{z} \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{N}_1\ \mathrm{type}[\mathrm{y} \in \mathrm{N}_1,\ \mathrm{z} \in \mathrm{N}_1]} \qquad \text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}[\ ]} \quad (\mathrm{y} \in \mathrm{N}_1) \notin [\ ]}{\mathrm{y} \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{y} \in \mathrm{N}_1[\mathrm{y} \in \mathrm{N}_1]}}{\mathrm{El}_{N1}(*, \mathrm{y}) \in \mathrm{N}_1[\mathrm{y} \in \mathrm{N}_1]}$$

Applicando la β N_1-*red* allora $\mathrm{El}_{N1}(*, y) \to_1 y$
$\mathrm{El}_{N1}(*, y)$ è uguale definizionalmente.

4

$$\text{E-S}\,\dfrac{\text{var}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{[\ ]\ \text{cont}}{N_1\ \text{type}\ [\ ]}}{x \in N_1\ \text{cont}}\,(x \in N_1) \notin [\ ]}{N_1\ \text{type}[x \in N_1]}}{x \in N_1,\ y \in N_1\ \text{cont}}\,\begin{matrix}(y \in N_1)\\ \notin x \in N_1\end{matrix}}{x \in N_1[x \in N_1,\ y \in N_1]} \quad \text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{[\ ]\ \text{cont}}{N_1\ \text{type}[\ ]}}{x \in N_1\ \text{cont}}\,(x \in N_1) \notin [\ ]}{N_1\ \text{type}[x \in N_1]}}{x \in N_1,\ y \in N_1\ \text{cont}}\,\begin{matrix}(y \in N_1)\\ \notin (x \in N_1)\end{matrix}}{N_1\ \text{type}[x \in N_1,\ y \in N_1]}}{x \in N_1,\ y \in N_1,\ z \in N_1\ \text{cont}}\,\begin{matrix}(z \in N_1) \notin\\ (x \in N_1,\ y \in N_1)\end{matrix}}{N_1\ \text{type}[x \in N_1,\ y \in N_1,\ z \in N_1]} \quad \text{var}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{[\ ]\ \text{cont}}{N_1\ \text{type}\ [\ ]}}{x \in N_1\ \text{cont}}\,(x \in N_1) \notin [\ ]}{N_1\ \text{type}[x \in N_1]}}{x \in N_1,\ y \in N_1\ \text{cont}}\,\begin{matrix}(y \in N_1)\\ \notin x \in N_1\end{matrix}}{y \in N_1[x \in N_1,\ y \in N_1]}}{\mathrm{El}_{N1}(x, y) \in N_1[x \in N_1,\ y \in N_1]}$$

Applicando la β N_1-*red* allora $\mathrm{El}_{N1}(x, y) \not\twoheadrightarrow_1$
$\mathrm{El}_{N1}(x, y)$ non è uguale definizionalmente.

5

$$\text{E-S}\,\dfrac{\dfrac{\mathbf{5}_A}{\mathrm{El}_{N1}(*, y) \in N_1[y \in N_1,\ x \in N_1]} \quad \dfrac{\mathbf{5}_B}{N_1\ \text{type}[y \in N_1,\ x \in N_1,\ z \in N_1]} \quad \dfrac{\mathbf{5}_C}{\mathrm{El}_{N1}(x, *) \in N_1[y \in N_1,\ x \in N_1]}}{\mathrm{El}_{N1}(\mathrm{El}_{N1}(*, y),\ \mathrm{El}_{N1}(x, *)) \in N_1[y \in N_1,\ x \in N_1]}$$

$\mathbf{5}_A$

$$\text{ind-te}\,\dfrac{\dfrac{}{\mathrm{El}_{N1}(*, y) \in N_1[y \in N_1]} \quad \text{F-c}\,\dfrac{\text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{[\ ]\ \text{cont}}{N_1\ \text{type}[\ ]}}{y \in N_1\ \text{cont}}\,(y \in N_1) \notin [\ ]}{N_1\ \text{type}[y \in N_1]}}{y \in N_1,\ x \in N_1\ \text{cont}}\,(x \in N_1) \notin y \in N_1}{\mathrm{El}_{N1}(*, y) \in N_1[y \in N_1,\ x \in N_1]}$$

$\mathbf{5}_B$

$$\text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{\text{F-c}\,\dfrac{\text{F-S}\,\dfrac{[\ ]\ \text{cont}}{N_1\ \text{type}[\ ]}}{y \in N_1\ \text{cont}}\,(y \in N_1) \notin [\ ]}{N_1\ \text{type}[y \in N_1]}}{y \in N_1,\ x \in N_1\ \text{cont}}\,(x \in N_1) \notin y \in N_1}{N_1\ \text{type}[y \in N_1,\ x \in N_1]}}{y \in N_1,\ x \in N_1,\ z \in N_1\ \text{cont}}\,(z \in N_1) \notin (y \in N_1,\ x \in N_1)}{N_1\ \text{type}[y \in N_1,\ x \in N_1,\ z \in N_1]}$$

5_C

$$
\text{ex-te}\ \dfrac{\text{ind-te}\ \dfrac{\dfrac{}{\mathrm{El}_{N1}(x,\ *) \in \mathrm{N}_1[x \in \mathrm{N}_1]} \qquad \text{F-c}\ \dfrac{\text{F-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}\ [\ ]}\quad (x \in \mathrm{N}_1) \notin [\ ]}{x \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{N}_1\ \mathrm{type}[x \in \mathrm{N}_1]}\quad (y \in \mathrm{N}_1) \notin x \in \mathrm{N}_1}{x \in \mathrm{N}_1,\ y \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{El}_{N1}(x,\ *) \in \mathrm{N}_1[x \in \mathrm{N}_1,\ y \in \mathrm{N}_1]} \qquad \text{F-c}\ \dfrac{\text{F-S}\ \dfrac{\text{F-c}\ \dfrac{\text{F-S}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{N}_1\ \mathrm{type}\ [\ ]}\quad (y \in \mathrm{N}_1) \notin [\ ]}{y \in \mathrm{N}_1\ \mathrm{count}}}{\mathrm{N}_1\ \mathrm{type}[y \in \mathrm{N}_1]}\quad (x \in \mathrm{N}_1) \notin y \in \mathrm{N}_1}{y \in \mathrm{N}_1,\ x \in \mathrm{N}_1\ \mathrm{cont}}}{\mathrm{El}_{N1}(x,\ *) \in \mathrm{N}_1[y \in \mathrm{N}_1,\ x \in \mathrm{N}_1]}
$$

Per le premesse $El_{N1}(*,\ y) \in N_1[y \in N_1]$ e $El_{N1}(x,\ *) \in N_1[x \in N_1]$ ho già dimostrato sopra (in 3 e 2) la loro tipabilità per il tipo singoletto.
Applicando la *red*$_I$ e β N_1-*red* allora $\mathrm{El}_{N1}(\mathrm{El}_{N1}(*,\ y),\ \mathrm{El}_{N1}(x,\ *)) \to_1 \mathrm{El}_{N1}(y,\ \mathrm{El}_{N1}(x,\ *))$.
Più nel dettaglio la riduzione è la seguente

$$
\mathrm{red}_I\ \dfrac{\beta\ \mathrm{N}_1\text{-red}\ \dfrac{}{\mathrm{El}_{N1}(*,\ y) \to_1 y}}{\mathrm{El}_{N1}(\mathrm{El}_{N1}(*,\ y),\ \mathrm{El}_{N1}(x,\ *)) \to_1 \mathrm{El}_{N1}(y,\ \mathrm{El}_{N1}(x,\ *))}
$$

$\mathrm{El}_{N1}(\mathrm{El}_{N1}(*,\ y),\ \mathrm{El}_{N1}(x,\ *))$ è uguale definizionalmente.

4) Dati i termini definiti dalla seguente grammatica relativa ai termini del tipo singoletto

$$\mathbf{t \equiv v \mid * \mid El_{N1}(t_1,\ t_2)}$$

con $\mathbf{v \in \{x,y,w,z\} \bigcup \{x_i \mid i \in Nat\}}$, ovvero considerando come variabili le ultime lettere dell'alfabeto inglese e poi tutte le variabili ottenute ponendo alla variabili x un indice che varia nei numeri naturali.
Sia $\to_1$ una relazione binaria tra questi termini *untyped* definita a partire dalle seguenti regole

$$\beta_{N1})\ \mathbf{red} \qquad \mathbf{El_{\mathit{Nat}}(*,c) \to_1 c}$$

$$\mathbf{red}_I)\ \dfrac{\mathbf{t_1 \to t_2}}{\mathbf{El_{N1}(t_1,\ c) \to_1 El_{N1}(t_2,\ c)}} \qquad \mathbf{red}_{II})\ \dfrac{\mathbf{c_1 \to c_2}}{\mathbf{El_{N1}(t,\ c_1) \to_1 El_{N1}(t,\ c_2)}}$$

- **Costruire l'albero dei cammini (ovvero sequenze) di passi di riduzione possibili fino a un termine in forma normale, ovvero non ulteriormente riducibile rispetto alla relazione $\to_1$ del termine**

$$\mathbf{El_{N_1}(El_{N1}(*,\ *),\ El_{N1}(*,\ x))}$$

- **Produrre un infinità di termini del tipo singoletto che non sono riducibili secondo la relazione di un passo di riduzione$\to_1$.**
  Dati due di questi termini , si riesce a dire che sono definionalmente uguali secondo le regole del tipo singoletto?

Soluzione

Idea: uso un albero di derivazione per mostrare ogni passo di derivazione di ogni cammino.
Se $\mathrm{w} = \mathrm{El}_{N1}(\mathrm{El}_{N1}(*,\ *),\ \mathrm{El}_{N1}(*,\ x))$ combino il lambda termine w con l'applicazione della strategia deterministica di riduzione ($\to_1$), con la quale il termine

si riduce eventualmente a forma normale (implicando la definizione di riducibilità).
β_{N1}-red:
$El_{N1}(*,\ *) \to_1 *$
$El_{N1}(*,\ x) \to_1 x$

$$\text{red}_I\ \dfrac{\beta\text{-N}_1\text{ red}\ \dfrac{\beta\text{-N}_1\text{ red}\ \dfrac{\text{x}}{\text{El}_{N1}(*,\text{ x})} \quad \dfrac{\text{x}}{\text{El}_{N1}(*,\text{ x})}\ \beta\text{-N}_1\text{ red}}{\text{El}_{NI}(*,\ \text{El}_{N1}(*,\text{ x}))}\ \text{red}_{II} \qquad \dfrac{\dfrac{\text{x}}{\text{El}_{N1}(*,\text{ x})}\ \beta\text{-N}_1\text{ red}}{\text{El}_{N1}(\text{El}_{N1}(*,*),\text{ x})}\ \text{red}_I}{\text{El}_{N1}(\text{El}_{N1}(*,\ *),\ \text{El}_{N1}(*,\text{ x}))}\ \text{red}_{II}$$

$\Rightarrow$ (w, (red$_{II}$, red$_I$, β_{N1}-red)) rappresenta un programma.

Termine t non più riducibile significa che è un termine *untyped* che è in forma normale, perchè non esiste alcun altro termine s tale che t $\to_1$ s. Dunque l'infinità di termini singoletto, non più riducibili rispettano la definizione data sopra corrisponde a

$$\text{t} \equiv \begin{cases} v \\ * \\ El_{N1}(t_1, t_2) \end{cases}$$

Dati due termini t' e t'' termini *untyped* non riesco a dire che sono definizionalmente uguali perchè già in forma normale. Difatti, per il teorema della forma normale forte, vale $\to_0$.

Capitolo 3

Tipo dei numeri Naturali

3.1 Regole di Formazione

$$\text{F-Nat)}\ \frac{\Gamma\ \text{cont}}{\text{Nat type}[\Gamma]}$$

3.2 Regole di Introduzione

$$\text{I}_1\text{-Nat)}\ \frac{\Gamma\ \text{cont}}{0 \in \text{Nat}[\Gamma]} \qquad \text{I}_2\text{-Nat)}\ \frac{\text{m} \in \text{Nat}[\Gamma]}{\text{succ(m)} \in \text{Nat}[\Gamma]}$$

3.3 Regole di Eliminazione

$$\text{E-Nat)}\ \frac{\text{t} \in \text{Nat}[\Gamma] \qquad \text{M(z) type}[\Gamma,\ \text{z} \in \text{Nat}] \qquad \begin{array}{c}\text{c} \in \text{M(0)}[\Gamma] \\ \text{e(x,y)} \in \text{M(succ(x))}[\Gamma,\ \text{x} \in \text{Nat},\ \text{y} \in \text{M(x)}]\end{array}}{\text{El}_{Nat}\text{(t,c,e)} \in \text{M(t)}[\Gamma]}$$

3.4 Regole di Conversione

$$\text{C}_1\text{-Nat)}\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{Nat}] \qquad \text{c} \in \text{M(0)}[\Gamma] \qquad \text{e(x,y)} \in \text{M(succ(x))}[\Gamma,\ \text{x} \in \text{Nat},\ \text{y} \in \text{M(x)}]}{\text{El}_{Nat}\text{(0,c,e)} = \text{c} \in \text{M(0)}[\Gamma]}$$

$$\text{C}_2\text{-Nat)}\ \frac{\text{m} \in \text{Nat}[\Gamma] \qquad \text{M(z) type}[\Gamma,\ \text{z} \in \text{Nat}] \qquad \begin{array}{c}\text{c} \in \text{M(0)}[\Gamma] \\ \text{e(x,y)} \in \text{M(succ(x))}[\Gamma,\ \text{x} \in \text{Nat},\ \text{y} \in \text{M(x)}]\end{array}}{\text{El}_{Nat}\text{(succ(m),c,e)} = \text{e(m, El}_{Nat}\text{(m,c,e))} \in \text{M(succ(m))}[\Gamma]}$$

3.5 Regole di Uguaglianza

$$\text{eq-Nat)}\ \frac{\text{t}_1 = \text{t}_2 \in \text{Nat}[\Gamma]}{\text{succ(t}_1\text{)} = \text{succ(t}_2\text{)} \in \text{Nat}[\Gamma]}$$

3.6 Introduzione ed Eliminatore dipendente

Le regole di formazione dei tipi e dei loro termini sono formulate, in modo, da rendere la regole di sostituzione per tipi e termini ammissibili.
Ad esempio la regola di introduzione del successore di un numero naturale, si può formulare come un esplicito programma funzionale visto come termine dipendente.

$$\mathrm{I_2\text{-}Nat}_{prog})\ \frac{\Gamma\ \mathrm{cont}}{\mathrm{succ(x)} \in \mathrm{Nat}[\Gamma,\ \mathrm{x} \in \mathrm{Nat}]}$$

Il medesimo discorso vale per la regola di eliminazione

$$\mathrm{E\text{-}Nat}_{dip})\ \frac{\mathrm{M(z)\ type}[\Gamma,\ \mathrm{z} \in \mathrm{Nat}] \qquad \mathrm{c} \in \mathrm{M(0)}[\Gamma] \qquad \mathrm{e(x,y)} \in \mathrm{M(succ(x))}[\Gamma,\ \mathrm{x} \in \mathrm{Nat},\ \mathrm{y} \in \mathrm{M(x)}]}{\mathrm{El}_{Nat}\mathrm{(z,c,e)} \in \mathrm{M(z)}[\Gamma,\ \mathrm{z} \in \mathrm{Nat}]}$$

(E-Nat) è equivalente a *(E-Nat$_{dip}$)*. Difatti la teoria T_{N1Nat}, in cui c'è *(E-Nat)*, è equivalente a $T^{\wedge}$ senza *(E-Nat)*, ma con *(E-Nat$_{dip}$)*, le regole di sostituzione e di *Sanitary check*.

3.7 Semantica operazionale dei numeri Naturali

La relazione $\to_1$ viene definita all'interno dei termini con l'uso delle seguenti regole di riduzione:

- β_{1Nat}-red) $\mathrm{El}_{Nat}(0,\ \mathrm{c},\ \mathrm{e}) \to_1 \mathrm{c}$
- β_{2Nat}-red) $\mathrm{El}_{Nat}(\mathrm{succ(m)},\ \mathrm{c},\ \mathrm{e}) \to_1 \mathrm{e}(\mathrm{m},\mathrm{El}_{Nat}(\mathrm{m},\ \mathrm{c},\ \mathrm{e}))$
- $\mathrm{Nat\text{-}red_1})\ \dfrac{\mathrm{t_1} \to_1 \mathrm{t_2}}{\mathrm{El}_{Nat}(\mathrm{t_1},\ \mathrm{c},\ \mathrm{e}) \to_1 \mathrm{El}_{Nat}(\mathrm{t_2},\ \mathrm{c},\ \mathrm{e})}$

  $\mathrm{Nat\text{-}red_2})\ \dfrac{\mathrm{c_1} \to_1 \mathrm{c_2}}{\mathrm{El}_{Nat}(\mathrm{t},\ \mathrm{c_1},\ \mathrm{e}) \to_1 \mathrm{El}_{Nat}(\mathrm{t},\ \mathrm{c_2},\ \mathrm{e})}$
- novità dei numeri Naturali rispetto al tipo singoletto $\mathrm{Nat\text{-}red})\ \dfrac{\mathrm{t_1} \to_1 \mathrm{t_2}}{\mathrm{succ(t_1)} \to_1 \mathrm{succ(t_2)}}$

3.8 Primitiva ricorsiva

Definizione

$\mathrm{Nat}^n \times \mathrm{Nat} \to \mathrm{Nat}$
Dati $\mathrm{g_0}$: $\mathrm{Nat}^m \to \mathrm{Nat}$ e $\mathrm{g_1}$: $\mathrm{Nat}^m \times \mathrm{Nat} \times \mathrm{Nat} \to \mathrm{Nat}$
$\mathrm{n_1 ... n}_m \in \mathrm{Nat}$ allora

$$\mathbf{rec(n_1...n}_m\mathbf{,\ 0)} \stackrel{def}{=} \mathbf{g}_0\mathbf{(n_1...n}_m\mathbf{)}$$
$$\mathbf{rec(n_1...n}_m\mathbf{,\ k+1)} \stackrel{def}{=} \mathbf{g}_0\mathbf{(n_1...n}_m\mathbf{,\ k,\ rec(n_1...n}_m\mathbf{,\ k))}$$

3.9 Addizione per ricorsione

Di seguito riporto un esercizio, svolto in aula, con lo scopo di comprendere come l'uguaglianza definizionale, tra numeri naturali, non coincide con l'uguaglianza aritmetica in matematica.

La somma, tra numeri naturali, viene definita usando l'eliminatore dipendente *E-Nat*$_{dip}$(§3.6). In questo modo si riesce a definire per la nostra teoria T_{1Nat}

$$\text{w + z} \in \text{Nat [w} \in \text{Nat, z} \in \text{Nat]}$$
come
$$\text{El}_{Nat}\text{(z, w, (x,y).succ(y))[w} \in \text{Nat, z} \in \text{Nat]}$$

Usando la nozione di primitiva ricorsiva e decidendo di ricorrere su z

$$\text{w + 0} \equiv \text{w}$$
$$\text{w + succ(z)} \equiv \text{succ(w+z)} \equiv \text{succ(y)}$$

Ecco che l'albero di derivazione assume la seguente forma:

$$\text{E-Nat}_{dip}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\ ]\ \text{cont}}{\text{Nat type [ ]}}}{\text{w} \in \text{Nat cont}}\ (\text{w} \in \text{Nat}) \notin [\ ]}{\text{Nat type[w} \in \text{Nat]}}}{\text{w} \in \text{Nat, z} \in \text{Nat cont}}\ \begin{matrix}(\text{z} \in \text{Nat}) \notin \\ \text{w} \in \text{Nat}\end{matrix}}{\text{Nat type[w} \in \text{Nat, z} \in \text{Nat]}} \qquad \text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\ ]\ \text{cont}}{\text{Nat type [ ]}}}{\text{w} \in \text{Nat cont}}\ (\text{w} \in \text{Nat}) \notin [\ ]}{\text{w} \in \text{Nat[w} \in \text{Nat]}} \qquad \text{I}_2\text{-Nat}_{prog}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\ ]\ \text{cont}}{\text{Nat type [ ]}}}{\text{w} \in \text{Nat cont}}\ (\text{w} \in \text{Nat}) \notin [\ ]}{\text{Nat type[w} \in \text{Nat]}}}{\text{w} \in \text{Nat, x} \in \text{Nat cont}}\ \begin{matrix}(\text{x} \in \text{Nat}) \notin \\ \text{w} \in \text{Nat}\end{matrix}}{\text{succ(y)} \in \text{Nat [w} \in \text{Nat, x} \in \text{Nat, y} \in \text{Nat]}}}{\text{El}_{Nat}\text{(z, w, (x,y).succ(y))} \in \text{Nat[w} \in \text{Nat, z} \in \text{Nat]}}$$

- Prova induttiva (minimo dei controlli da fare per verificare l'esattezza della derivazione)
  - Caso base: cosa accade se poniamo al posto di z lo 0?
    El$_{Nat}$(0, w, (x,y).succ(y)) $\to_1$ w per β_{1Nat}*-red* ($\equiv$ w + 0 = w)
  - Passo induttivo: ricorro su z (esempio succ(0) = 1)
    El$_{Nat}$(succ(0), w, (x,y).succ(y)) $\to_1$ succ(El$_{Nat}$(0, w, (x,y).succ(y)))
    per β_{1Nat}*-red* $\to_1$ succ(w)
    Dunque w + 1 = succ(w) $\in$ Nat

  $\Rightarrow$ Il programma fa effettivamente quello che dovrebbe.

3.9.1 Osservazioni sull'addizione

w $+_1$ z $\equiv$ El$_{Nat}$(z, w, (x,y), succ(y)) $\neq$ come NF da z.
Se sostituisco w con 0, allora 0 $+_1$ z $\equiv$ w $+_1$ z$[\frac{w}{0}]$ $\equiv$ El$_{Nat}$(z, 0, (x,y), succ(y)) è in NF (dunque non riesco più a ridurla ulteriormente).
Ecco che 0 $+_1$ z è un valore in NF $\neq$ da z, da cui è impossibile dimostrare che 0 $+_1$ z = z $\in$ Nat[z $\in$ Nat].
Questo non accade per w $+_1$ 0 = w (§3.9).
Perciò, se noi scriviamo la somma ricorrendo sul secondo membro, riusciamo a dire che *primo-membro* $+_1$ *0* = *primo-membro*, ma non che *0* $+_1$ *secondo-membro* = *secondo-membro*. In quanto non esiste alcuna sotto-strategia deterministica, per il secondo caso, per cui il programma si ferma.

In conclusione, l'uguaglianza definizionale di termini con variabili è diversa dall'uguaglianza aritmetica. Eccezione fatta per le espressioni chiuse, senza variabili, perchè il termine chiuso si riduce a un'unica NF, che si dimostra essere un numero arabico.

3.10 Esercizi

1) Dimostrare che le regole enunciate in §3, *I_2-Nat$_{prog}$* ed *E-Nat$_{dip}$*, sono ammissibili nel sistema di teoria dei tipi dei numeri Naturali.

Soluzione

- In $\mathrm{I_2\text{-}Nat}_{prog}$ il contesto è $\Sigma \equiv \Gamma$, $x \in$ Nat che permette l'applicazione di $\mathrm{I_2}$ Nat.

$$\alpha\text{-eq}\ \dfrac{\mathrm{I_2\text{-}Nat}\ \dfrac{\mathrm{var}\ \dfrac{\mathrm{F\text{-}c}\ \dfrac{\mathrm{F\text{-}Nat}\ \dfrac{\Gamma\ \mathrm{cont}}{\mathrm{Nat\ type}[\Gamma]}}{\Gamma,\ \mathrm{m} \in \mathrm{Nat\ cont}}\ (\mathrm{m} \in \mathrm{Nat}) \notin \Gamma}{\mathrm{m} \in \mathrm{Nat}[\Gamma,\ \mathrm{m} \in \mathrm{Nat}]}}{\mathrm{succ(m)} \in \mathrm{Nat}[\Gamma,\ \mathrm{m} \in \mathrm{Nat}]}}{\mathrm{succ(x)} \in \mathrm{Nat}[\Gamma,\ \mathrm{x} \in \mathrm{Nat}]}$$

Assumo che le premesse di *I_2-Nat$_{prog}$* (Γ *count*) siano valide, perciò è valido, dalla prova sopra, anche il giudizio di conclusione *succ(x)* $\in$ *Nat[*Γ*, x* $\in$ *Nat]*, di conseguenza derivabile in $T^{\wedge}$.

- In $\mathrm{E\text{-}Nat}_{dip}$ il contesto è $\Sigma \equiv \Gamma$, $z \in$ Nat, che permette l'applicazione di E-Nat.

$$\mathrm{E\text{-}Nat}\ \dfrac{\mathrm{var}\ \dfrac{\mathrm{F\text{-}c}\ \dfrac{\mathrm{F\text{-}Nat}\ \dfrac{\Gamma\ \mathrm{cont}}{\mathrm{Nat\ type}[\Gamma]}}{\Gamma,\ \mathrm{z} \in \mathrm{Nat\ cont}}\ (\mathrm{z} \in \mathrm{Nat}) \notin \Gamma}{\mathrm{z} \in \mathrm{Nat}[\Gamma,\ \mathrm{z} \in \mathrm{Nat}]} \quad \dfrac{}{\mathrm{M(z)\ type}[\Gamma,\mathrm{z} \in \mathrm{Nat}]} \quad \dfrac{}{\mathrm{c} \in \mathrm{M(0)}[\Gamma,\mathrm{z} \in \mathrm{Nat}]} \quad \dfrac{}{\mathrm{e(x,y)} \in \mathrm{M(succ(x))}[\Gamma,\ \mathrm{z} \in \mathrm{Nat},\ \mathrm{x} \in \mathrm{Nat},\ \mathrm{y} \in \mathrm{M(x)}]}}{\mathrm{El}_{Nat}\mathrm{(z,c,e)} \in \mathrm{M(z)}[\Gamma,\ \mathrm{z} \in \mathrm{Nat}]}$$

Assumo che le premesse di *E-Nat$_{dip}$* (*M(z) type[*Γ*, z* $\in$ *Nat], c* $\in$ *M(0)[*Γ*], e(x,y)* $\in$ *M(succ(x))[*Σ*, x* $\in$ *Nat, y* $\in$ *M(x)]*) siano valide e che Γ cont sia assioma ([ ] cont = [Γ] cont con $\Gamma = \varnothing$), perciò è valido, dalla prova sopra, anche il giudizio di conclusione *El$_{Nat}$(z,c,e)* $\in$ *M(z)[*Γ*, z* $\in$ *Nat]*, di conseguenza derivabile in $T^{\wedge}$.

2) Definire w + 2 ∈ Nat[w ∈ Nat], ove 2 è l'abbreviazione del termine ottenuto applicando 2 ≡ succ(succ(0)).

Soluzione

La ricorsione la faccio su w, usando lo schema di ricorsione primitiva, vale che w + 2 ≡ El_{Nat}(w, 2, (x,y).succ(y)) ∈ Nat[w ∈ Nat].

$$\text{E-Nat}_{dip}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{\text{w} \in \text{Nat cont}}\ (\text{w} \in \text{Nat}) \notin [\,]}{\text{Nat type}[\text{w} \in \text{Nat}]} \quad \text{I}_2\text{-Nat}\ \dfrac{\text{I}_2\text{-Nat}\ \dfrac{\text{I}_1\text{-Nat}\ \dfrac{[\,]\ \text{cont}}{0 \in \text{Nat}[\,]}}{1 \in \text{Nat}[\,]}}{2 \in \text{Nat}[\,]} \quad \text{I}_2\text{-Nat}_{prog}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{\text{x} \in \text{Nat cont}}\ (\text{x} \in \text{Nat}) \notin [\,]}{\text{succ(y)} \in \text{Nat}[\text{x} \in \text{Nat, y} \in \text{Nat}]}}{\text{El}_{Nat}(\text{w, 2, (x,y).succ(y))} \in \text{Nat}[\text{w} \in \text{Nat}]}$$

Dimostrazione di correttezza di El_{Nat}(w, 2, (x,y).succ(y)) ∈ Nat[w ∈ Nat]

- El_{Nat}(0, 2, (x,y).succ(y)) $\to_1$ 2 per β_{1Nat}*-red*
- El_{Nat}(succ(m), 2, (x,y).succ(y)) $\to_1$ succ(El_{Nat}(m, 2, (x,y).succ(y))) per β_{2Nat}*-red* ⇒ per m = 0 ≡ succ(El_{Nat}(0, 2, (x,y).succ(y))) $\to_1$ succ(2) ∈ Nat = 3 (dal punto precedente).

3) Definire l'operazione di addizione usando le regole del tipo dei numeri Naturali.

$$\text{x} + \text{y} \in \text{Nat}[\text{x} \in \text{Nat, y} \in \text{Nat}]$$

in modo tale che valga x + 0 = x ∈ Nat[x ∈ Nat]

Soluzione

La ricorsione la faccio sulla y, usando lo schema di ricorsione primitiva, vale che x + y ≡ El_{Nat}(y, x, (w,z).succ(z)) ∈ Nat[x ∈ Nat, y ∈ Nat].

$$\text{E-Nat}_{dip}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]}}{\text{x} \in \text{Nat cont}}\ (\text{x} \in \text{Nat}) \notin [\,]}{\text{Nat type}[\text{x} \in \text{Nat}]}}{\text{x} \in \text{Nat, y} \in \text{Nat cont}}\ \begin{matrix}(\text{y} \in \text{Nat}) \notin \\ \text{x} \in \text{Nat}\end{matrix}}{\text{Nat type}[\text{x} \in \text{Nat, y} \in \text{Nat}]} \quad \text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]}}{\text{x} \in \text{Nat cont}}\ (\text{x} \in \text{Nat}) \notin [\,]}{\text{x} \in \text{Nat}[\text{x} \in \text{Nat}]} \quad \text{I}_2\text{-Nat}_{prog}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{\text{x} \in \text{Nat cont}}\ (\text{x} \in \text{Nat}) \notin [\,]}{\text{Nat type}[\text{x} \in \text{Nat}]}}{\text{x} \in \text{Nat, w} \in \text{Nat cont}}\ \begin{matrix}(\text{w} \in \text{Nat}) \\ \notin \text{x} \in \text{Nat}\end{matrix}}{\text{succ(z)} \in \text{Nat}[\text{x} \in \text{Nat, w} \in \text{Nat, z} \in \text{Nat}]}}{\text{El}_{Nat}(\text{y, x, (w,z).succ(z))} \in \text{Nat}[\text{x} \in \text{Nat, y} \in \text{Nat}]}$$

Dimostrazione di correttezza di El_{Nat}(y, x, (w,z).succ(z)) ∈ Nat[x ∈ Nat, y ∈ Nat]

- El_{Nat}(0, x, (w,z).succ(z)) $\to_1$ x per β_{1Nat}*-red*
- El_{Nat}(succ(y), x, (w,z).succ(z)) $\to_1$ succ(El_{Nat}(y, x, (w,z).succ(z))) per β_{2Nat}*-red* ⇒ per y=0 ≡ succ(El_{Nat}(0, x, (w,z).succ(z))) $\to_1$ succ(x) ∈ Nat = x + 1 (dal punto precedente).

4) Definire l'operazione di addizione usando le regole del tipo dei numeri Naturali.

$$x + y \in \text{Nat}[x \in \text{Nat},\ y \in \text{Nat}]$$

in modo tale che valga $0 + y = y \in \text{Nat}[y \in \text{Nat}]$

Soluzione

La ricorsione la faccio sulla x, percui, usando lo schema di ricorsione primitiva, vale che $x + y \equiv \text{El}_{Nat}(x,\ y,\ (w,z).\text{succ}(z)) \in \text{Nat}[x \in \text{Nat},\ y \in \text{Nat}]$.

$$\text{ex-te}\ \dfrac{\dfrac{\mathbf{1}}{\text{El}_{Nat}(x,\ y,\ (w,z).\text{succ}(z)) \in \text{Nat}[y \in \text{Nat},\ x \in \text{Nat}]} \qquad \text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]} \quad (x \in \text{Nat}) \notin [\,]}{x \in \text{Nat cont}}}{\text{Nat type}[x \in \text{Nat}]} \quad (y \in \text{Nat}) \notin x \in \text{Nat}}{x \in \text{Nat},\ y \in \text{Nat cont}}}{\text{El}_{Nat}(x,\ y,\ (w,z).\text{succ}(z)) \in \text{Nat}[x \in \text{Nat},\ y \in \text{Nat}]}$$

1

$$\text{E-Nat}_{dip}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]} \quad (y \in \text{Nat}) \notin [\,]}{y \in \text{Nat cont}}}{\text{Nat type}\ [y \in \text{Nat}]} \quad (x \in \text{Nat}) \notin y \in \text{Nat}}{y \in \text{Nat},\ x \in \text{Nat cont}}}{\text{Nat type}[y \in \text{Nat},\ x \in \text{Nat}]} \quad \text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]} \quad (y \in \text{Nat}) \notin [\,]}{y \in \text{Nat cont}}}{y \in \text{Nat}[y \in \text{Nat}]} \quad \text{I}_2\text{-Nat}_{prog}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]} \quad (y \in \text{Nat}) \notin [\,]}{y \in \text{Nat cont}}}{\text{Nat type}[y \in \text{Nat}]} \quad (w \in \text{Nat}) \notin y \in \text{Nat}}{y \in \text{Nat},\ w \in \text{Nat cont}}}{\text{succ}(z) \in \text{Nat}[y \in \text{Nat},\ w \in \text{Nat},\ z \in \text{Nat}]}}{\text{El}_{Nat}(x,\ y,\ (w,z).\text{succ}(z)) \in \text{Nat}[y \in \text{Nat},\ x \in \text{Nat}]}$$

Dimostrazione di correttezza di $El_{Nat}(x,\ y,\ (w,z).succ(z)) \in Nat[y \in Nat,\ x \in Nat]$

- $\text{El}_{Nat}(0,\ y,\ (w,z).\text{succ}(z)) \to_1 y$ per β_{1Nat}-*red*
- $\text{El}_{Nat}(\text{succ}(x),\ y,\ (w,z).\text{succ}(z)) \to_1 \text{succ}(\text{El}_{Nat}(x,\ y,\ (w,z).\text{succ}(z)))$ per β_{2Nat}-*red* $\Rightarrow$ per $x=0 \equiv \text{succ}(\text{El}_{Nat}(0,\ y,\ (w,z).\text{succ}(z))) \to_1 \text{succ}(y) \in \text{Nat} = y + 1$ (dal punto precedente).

5) Un concetto importante del λ-cacolo è α-equivalenza (α-eq). Questa consiste in una forma base di equivalenza, definibile in termini *lambda*, dove una qualsiasi variabile legata può essere convertita in un'altra. Per esempio, λx.x e λy.y lambda sono termini α-*eq* che rappresentano la medesima funzione.
Nella dispensa ho utilizzato α-*eq* per provare la validità delle conclusioni , quando ne sussistevano le premesse per farlo.
Di seguito dimostro la validità di α-*eq* derivando la sua stessa regola.

$$\dfrac{M(x)\ \text{type}[x \in A]}{M(w)\ \text{type}[w \in A]}$$

Soluzione

Assumo, per ipotesi, che A sia un tipo non dipendente ($A \equiv \text{Nat}$). Vale però per qualsiasi tipo.

$$
\text{sub-typ}\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{\text{w} \in \text{Nat cont}}\ (\text{w} \in \text{Nat}) \notin [\,]}{\text{w} \in \text{Nat}[\text{w} \in \text{Nat}]} \qquad \text{ex-ty}\ \dfrac{\dfrac{\mathbf{1}}{\text{M(x) type}[\text{x} \in \text{Nat},\ \text{w} \in \text{Nat}]} \qquad \dfrac{\mathbf{2}}{\text{w} \in \text{Nat},\ \text{x} \in \text{Nat cont}}}{\text{M(x) type}[\text{w} \in \text{Nat},\ \text{x} \in \text{Nat}]}}{\text{M(w) type}[\text{w} \in \text{Nat}]}
$$

1

$$
\text{ind-ty}\ \dfrac{\dfrac{}{\text{M(x) type}[\text{x} \in \text{Nat}]} \qquad \text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{\text{x} \in \text{Nat cont}}\ (\text{x} \in \text{Nat}) \notin [\,]}{\text{Nat type}[\text{x} \in \text{Nat}]}}{\text{x} \in \text{Nat},\ \text{w} \in \text{Nat cont}}\ (\text{w} \in \text{Nat}) \notin \text{x} \in \text{Nat}}{\text{M(x) type}[\text{x} \in \text{Nat},\ \text{w} \in \text{Nat}]}
$$

2

$$
\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{\text{w} \in \text{Nat cont}}\ (\text{w} \in \text{Nat}) \notin [\,]}{\text{Nat type}[\text{w} \in \text{Nat}]}}{\text{w} \in \text{Nat},\ \text{x} \in \text{Nat cont}}\ (\text{x} \in \text{Nat}) \notin \text{w} \in \text{Nat}
$$

Capitolo 4

Tipo delle liste di un tipo

Il tipo delle liste di elementi costruisce un costruttore delle liste ed è definito dalle regole seguenti.

4.1 Regole di Formazione

$$\text{F-cost)}\ \frac{\text{A type } [\Gamma]}{\text{List(A) type}[\Gamma]}$$

4.2 Regole di Introduzione

$$\text{I}_1\text{-List)}\ \frac{\text{List(A) type } [\Gamma]}{\text{nil} \in \text{List(A)}[\Gamma]} \qquad \text{I}_2\text{-List)}\ \frac{\text{s} \in \text{List(A)}[\Gamma] \qquad \text{a} \in \text{A}[\Gamma]}{\text{cons(s,a)} \in \text{List(A)}[\Gamma]}$$

4.3 Regole di Eliminazione

$$\text{E-List)}\ \frac{\begin{array}{c}\text{M(z) type}[\Gamma,\ \text{z} \in \text{List(A)}]\\ \text{t} \in \text{List(A)}[\Gamma]\\ \text{c} \in \text{M(nil)}[\Gamma]\end{array} \qquad \text{e(x,w,y)} \in \text{M(cons(x,w))}[\Gamma,\ \text{x} \in \text{List(A)},\ \text{w} \in \text{A},\ \text{y} \in \text{M(x)}]}{\text{El}_{List}\text{(t,c,e)} \in \text{M(t)}[\Gamma]}$$

4.4 Regole di Conservazione

$$\text{C}_1\text{-List)}\ \frac{\begin{array}{c}\text{M(z) type}[\Gamma,\ \text{z} \in \text{List(A)}]\\ \text{c} \in \text{M(nil)}[\Gamma]\end{array} \qquad \text{e(x,w,y)} \in \text{M(cons(x,w))}[\Gamma,\ \text{x} \in \text{List(A)},\ \text{w} \in \text{A},\ \text{y} \in \text{M(x)}]}{\text{El}_{List}\text{(nil,c,e)} = \text{c} \in \text{M(nil)}[\Gamma]}$$

$$\text{C}_2\text{-List)}\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{List(A)}] \qquad \begin{array}{c}\text{s} \in \text{List(A)}[\Gamma]\\ \text{a} \in \text{A}[\Gamma]\\ \text{c} \in \text{M(nil)}[\Gamma]\end{array} \qquad \text{e(x,w,y)} \in \text{M(cons(x,w))}[\Gamma,\ \text{x} \in \text{List(A)},\ \text{w} \in \text{A},\ \text{y} \in \text{M(x)}]}{\text{El}_{List}\text{(cons(s,a),c,e)} = \text{e(s,a, E}_{List}\text{(s,c,e))} \in \text{M(cons(s,a))}[\Gamma]}$$

4.5 Regole di Uguaglianza

$$\text{eq-I}_1\text{-List)}\ \frac{A_1 = A_2\ type[\Gamma]}{List(A_1) = List(A_2)\ type(\Gamma)}$$

$$\text{eq-I}_2\text{-List)}\ \frac{s_1 = s_2 \in List(A)[\Gamma] \qquad a_1 = a_2 \in A[\Gamma]}{cons(s_1,a_1) = cons(s_2,a_2) \in List(A)(\Gamma)}$$

$$\text{eq-E-List)}\ \frac{\begin{array}{c} M(z)\ type[\Gamma,\ z \in List(A)] \\ t_1 = t_2 \in List(A)[\Gamma] \\ c_1 = c_2 \in M(nil)[\Gamma] \end{array} \qquad e_1(x,w,y) = e_2(x,w,y) \in M(cons(x,w))[\Gamma,\ x \in List(A),\ w \in A,\ y \in M(x)]}{El_{List}(t_1,\ c_1,\ e_1) = El_{List}(t_2,\ c_2,\ e_2) \in M(t_1)[\Gamma]}$$

4.6 Eliminatore dipendente

L'eliminatore ha anche la forma dipendente, conveniente da usare soprattutto per motivi pratici.

$$\text{E-List}_{dip})\ \frac{\begin{array}{c} M(z)\ type[\Gamma,\ z \in List(A)] \\ c \in M(nil)[\Gamma] \end{array} \qquad e(x,w,y) \in M(cons(x,w))[\Gamma,\ x \in List(A),\ w \in A,\ y \in M(x)]}{El_{List}(z,c,e) \in M(z)[\Gamma,\ z \in List(A)]}$$

Quando si scrive un programma è bene scrivere la sostituzione espressamente, come riporto sotto.

$$\text{E-List}_{dip})\ \frac{\begin{array}{c} D\ type[\Gamma,\ z \in List(A)] \\ c \in D(\frac{z}{nil})[\Gamma] \end{array} \qquad e(x,w,y) \in D(\frac{z}{cons(x,w)})[\Gamma,\ x \in List(A),\ w \in A,\ y \in D(\frac{z}{x})]}{El_{List}(z,c,e) \in D(z)[\Gamma,\ z \in List(A)]}$$

4.7 Semantica operazionale del tipo lista

La relazione $\to_1$ viene definita all'interno dei termini con l'uso delle seguenti regole di riduzione:

- β_{1List}-red) $El_{List}(nil,\ c,\ e) \to_1 c$
- β_{2List}-red) $El_{List}(cons(s,a),\ c,\ e) \to_1 e(s,a,El_{List}(s,\ c,\ e))$
- List-red$_1$) $\dfrac{t_1 \to_1 t_2}{El_{List}(t_1,\ c,\ e) \to_1 El_{List}(t_2,\ c,\ e)}$

  List-red$_2$) $\dfrac{c_1 \to_1 c_2}{El_{List}(t,\ c_1,\ e) \to_1 El_{List}(t,\ c_2,\ e)}$
- novità delle liste rispetto al tipo singoletto List-red$_I$) $\dfrac{s_1 \to_1 s_2}{cons(s_1,a) \to_1 cons(s_2,a)}$
- List-red$_{II}$) $\dfrac{a_1 \to_1 a_2}{cons(s,a_1) \to_1 cons(s,a_2)}$

4.8 Esercizi

1) Dati i tipi singoletto e delle liste è possibile definire un tipo dei numeri Naturali *Nat*?

Soluzione

Posso vedere una lista come una collezione di n singoletti. Ecco che posso definire il tipo dei numeri Naturali su questa lista, per ricorsione primitiva, nel modo sottostante:

$$0 = \text{nil}$$
$$1 = \text{cons}(\text{nil}, *)$$
$$n = \text{cons}(\text{cons}(n\text{-}1), *)$$

2) Dato un tipo A semplice, ovvero non dipendente, per esempio A = $\mathbf{N_1}$, se si vuole definire un'operazione

$$\mathbf{append_1(x`,y`) \in List(A)[x` \in List(A),\ y` \in A]}$$

tale che l'elemento y` venga posto alla fine della lista x`, allora basta definire $\mathbf{append_1}$ nel modo seguente

$$\mathbf{append_1(x`,y`) = cons(x`,y`) \in List(A)[x` \in List(A),\ y` \in A]}$$

Ma se si vuole definire un'operazione

$$\mathbf{append_2(x`,y`) \in List(A)[x` \in List(A),\ y` \in A]}$$

tale che y` venga posto all'inizio della lista x`, allora come occorre procedere?

Soluzione

Per riuscire a porre y` all'inizio della lista x`, devo usare la regola di eliminazione sulla lista x`. Per farlo, devo prima di tutto dare equazionalmente la definizione ricorsiva di append_2

$$\text{append}_2(\text{nil}, y`) = \text{cons}(\text{nil}, y`) = y`$$
$$\text{append}_2(\text{cons}(s, x`), y`) = \text{cons}(\text{append}_2(s, y`), x`)$$

Ora posso applicare la regola dell'eliminazione nel modo seguente:

- Premesse:
  - M(z) type[Γ, z ∈ List(A)] ≡ List(A) type[x` ∈ List(A), y` ∈ A] (*z ∈ List(A) risulta superfluo*)
  - t ∈ List(A)[Γ] ≡ x` ∈ List(A)[x` ∈ List(A), y` ∈ A]
  - c ∈ M(nil)[Γ] ≡ y` ∈ A[x` ∈ List(A), y` ∈ A]
  - e(x,w,y) ∈ M(cons(x,w))[Γ, x ∈ List(A), w ∈ A, y ∈ M(x)] ≡ cons(z,y) ∈ List(A)[x` ∈ List(A), y` ∈ A, y∈ A, z ∈ List(A)]
- Giudizio di conclusione:

– $El_{List}(t,c,e) \in M(t)[\Gamma] \equiv El_{List}(x', y', (x,y,z).cons(z,y)) \in List(A)[x' \in List(A), y' \in A]$

$$\text{E-List}\ \dfrac{\mathbf{1}\qquad \mathbf{2}\qquad \mathbf{3}\qquad \mathbf{4}}{El_{List}(x', y', (x,y,z).cons(z,y)) \in List(A)[x' \in List(A), y' \in A]}$$

1

$$\text{F-cost}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ type[\ ]}\qquad \text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ type[\ ]}\qquad \text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\dfrac{}{A\ type[\ ]}}{List(A)\ type[\ ]}}{x' \in List(A)\ cont}\ (x' \in List(A)) \notin [\ ]}{A\ type[x' \in List(A)]}}{x' \in List(A), y' \in A\ cont}\ (y' \in A) \notin x' \in List(A)}{A\ type[x' \in List(A), y' \in A]}}{List(A)\ type[x' \in List(A), y' \in A]}$$

2

La regola di scambio di contesto (ex-ty) può venire anche omessa.

$$\text{ex-ty}\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ type[\ ]}\qquad \text{F-c}\ \dfrac{\dfrac{}{A\ type[\ ]}}{y' \in A\ cont}\ (y' \in A) \notin [\ ]}{A\ type[y' \in A]}}{List(A)\ type[y' \in A]}}{y' \in A, x' \in List(A)\ cont}\ (x' \in List(A)) \notin y' \in A}{x' \in List(A)[y' \in A, x' \in List(A)]}\qquad \text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ type[\ ]}\qquad \text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\dfrac{}{A\ type[\ ]}}{List(A)\ type[\ ]}}{x' \in List(A)\ cont}\ (x' \in List(A)) \notin [\ ]}{A\ type[x' \in List(A)]}}{x' \in List(A), y' \in A\ cont}\ (y' \in A) \notin x' \in List(A)}{x' \in List(A)[x' \in List(A), y' \in A]}$$

3

$$\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ type[\ ]}\qquad \text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\dfrac{}{A\ type[\ ]}}{List(A)\ type\ [\ ]}}{x' \in List(A)\ cont}\ (x' \in List(A)) \notin [\ ]}{A\ type[x' \in List(A)]}}{x' \in List(A), y' \in A\ cont}\ (y' \in A) \notin x' \in List(A)}{y' \in A[x' \in List(A), y' \in A]}$$

4

$$
\text{I}_2\text{-List}\ \dfrac{\text{var}\ \dfrac{\dfrac{\mathbf{4'}}{\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{z} \in \text{List(A)},\ \text{y} \in \text{A},\ \text{z} \in \text{List(A) cont}}}{\text{z} \in \text{List(A)}[\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{y} \in \text{A},\ \text{z} \in \text{List(A)}]} \qquad \text{var}\ \dfrac{\dfrac{\mathbf{4'}}{\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{y} \in \text{A},\ \text{z} \in \text{List(A) cont}}}{\text{y} \in \text{A}[\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{y} \in \text{A},\ \text{z} \in \text{List(A)}]}}{\text{cons(z,y)} \in \text{List(A)}[\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{y} \in \text{A},\ \text{z} \in \text{List(A)}]}
$$

4'

$$
\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{\text{A type[ ]}} \quad \text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{\text{A type[ ]}} \quad \text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{\text{A type[ ]}} \quad \text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\dfrac{}{\text{A type[ ]}}}{\text{List(A) type[ ]}}}{\text{x}' \in \text{List(A) cont}}\ (\text{x}' \in \text{List(A)}) \notin [\ ]}{\text{A type}[\text{x}' \in \text{List(A)}]}}{\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A cont}}\ (\text{y}' \in \text{A}) \notin \text{x}' \in \text{List(A)}}{\text{A type}[\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A}]}}{\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{y} \in \text{A cont}}\ (\text{y} \in \text{A}) \notin (\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A})}{\text{List(A) type}[\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{y} \in \text{A}]}}{\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{y} \in \text{A},\ \text{z} \in \text{List(A) cont}}\ (\text{z} \in \text{List(A)}) \notin (\text{x}' \in \text{List(A)},\ \text{y}' \in \text{A},\ \text{y} \in \text{A})
$$

Dimostrazione di correttezza di $El_{List}(x', y', (x,y,z).cons(z,y)) \in List(A)[x' \in List(A), y' \in A]$

- $\text{El}_{List}(\text{nil}, \text{y}', (\text{x,y,z}).\text{cons(z,y)}) \to_1 \text{y}'$ per β_{1List}-*red*
- $\text{El}_{List}(\text{cons(s,x}'), \text{y}', (\text{x,y,z}).\text{cons(z,y)}) \to_1 \text{cons}(\text{El}_{List}(\text{x}', \text{y}', (\text{x,y,z}).\text{cons(z,y)}))$ per β_{2List}-*red* $\Rightarrow$ per $\text{x}' = \text{nil} \equiv \text{cons}(\text{El}_{List}(\text{nil}, \text{y}', (\text{x,y,z}).\text{cons(z,y)})) \to_1$ $\text{cons(nil, y}') \in \text{List(A)} = \text{y}'$ (dal punto precedente).

3) Definire l'operazione *append* di accostamento di una lista a un'altra di tipo A type[ ] usando le regole del tipo delle liste

$$\textbf{append(x,y)} \in \textbf{List(A)[x} \in \textbf{List(A), y} \in \textbf{List(A)]}$$

in modo tale che valga append(x,nil) $=$ x $\in$ List(A)[x $\in$ List(A)]

Soluzione

Per riuscire a poter concatenare la lista x alla lista y, devo usare la regola di eliminazione sulla lista y. Per farlo, devo prima di tutto definire equazionalmente la definizione ricorsiva di append

$$
\begin{gathered}
\text{append(x, nil)} = \text{cons(x, nil)} = \text{x} \\
\text{append(x, cons(y, a))} = \text{cons(x, append(y, a))}
\end{gathered}
$$

Ora posso applicare la regola dell'eliminazione nel modo seguente:

- Premesse:
  - $\text{M(z) type}[\Gamma, \text{z} \in \text{List(A)}] \equiv \text{List(A) type}[\text{x} \in \text{List(A)},\ \text{y} \in \text{List(A)}]$ ($z \in$ *List(A) risulta superfluo*)

- $t \in List(A)[\Gamma] \equiv y \in List(A)[x \in List(A),\ y \in List(A)]$
- $c \in M(nil)[\Gamma] \equiv x \in List(A)[x \in List(A),\ y \in List(A)]$
- $e(x,w,y) \in M(cons(x,w))[\Gamma,\ x \in List(A),\ w \in A,\ y \in M(x)] \equiv cons(z,v) \in List(A)[x \in List(A)\ y \in List(A),\ v \in A,\ z \in List(A)]$

- Giudizio di conclusione:
  - $El_{List}(t,c,e) \in M(t)[\Gamma] \equiv El_{List}(y,\ x,(u,v,z).cons(z,v)) \in List(A)[x \in List(A),\ y \in List(A)]$

$$\text{E-List}\ \dfrac{\mathbf{1}\qquad \mathbf{2}\qquad \mathbf{3}\qquad \mathbf{4}}{El_{List}(y,\ x,(u,v,z).cons(z,v)) \in List(A)[x \in List(A),\ y \in List(A)]}$$

1

$$\text{F-cost}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ type[\ ]} \qquad \text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ type[\ ]} \qquad \text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\dfrac{}{A\ type[\ ]}}{List(A)\ type[\ ]}}{x \in List(A)\ cont}\ (x \in List(A)) \notin [\ ]}{A\ type[x \in List(A)]}}{List(A)\ type[x \in List(A)]}}{x \in List(A),\ y \in List(A)\ cont}\ (y \in List(A)) \notin x \in List(A)}{A\ type[x \in List(A),\ y \in List(A)]}}{List(A)\ type[x \in List(A),\ y \in List(A)]}$$

2

$$\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ type[\ ]} \qquad \text{F-c}\ \dfrac{\text{F-cost}\ \dfrac{\dfrac{}{A\ type[\ ]}}{List(A)\ type[\ ]}}{x \in List(A)\ cont}\ (x \in List(A)) \notin [\ ]}{A\ type[x \in List(A)]}}{List(A)\ type[x \in List(A)]}}{x \in List(A),\ y \in List(A)\ cont}\ (y \in List(A)) \notin x \in List(A)}{y \in List(A)[x \in List(A),\ y \in List(A)]}$$

3

La regola di scambio di contesto (ex-ty) può venire anche omessa.

$$
\text{ex-ty}\;\dfrac{
\text{var}\;\dfrac{
\text{F-c}\;\dfrac{
\text{F-cost}\;\dfrac{
\text{ind-ty}\;\dfrac{
\dfrac{}{\text{A type}[\,]}\quad
\text{F-c}\;\dfrac{\text{F-cost}\;\dfrac{\dfrac{}{\text{A type}[\,]}}{\text{List(A) type}[\,]}}{y \in \text{List(A) cont}}\;(y \in \text{List(A)}) \notin [\,]
}{\text{A type}[y \in \text{List(A)}]}
}{\text{List(A) type}[y \in \text{List(A)}]}
}{y \in \text{List(A)},\ x \in \text{List(A) cont}}\;(x \in \text{List(A)}) \notin y \in \text{List(A)}
}{x \in \text{List(A)}[y \in \text{List(A)},\ x \in \text{List(A)}]}
\qquad
\text{F-c}\;\dfrac{
\text{F-cost}\;\dfrac{
\text{ind-ty}\;\dfrac{
\dfrac{}{\text{A type}[\,]}\quad
\text{F-c}\;\dfrac{\text{F-cost}\;\dfrac{\dfrac{}{\text{A type}[\,]}}{\text{List(A) type}[\,]}}{x \in \text{List(A) cont}}\;(x \in \text{List(A)}) \notin [\,]
}{\text{A type}[x \in \text{List(A)}]}
}{\text{List(A)type}[x \in \text{List(A)}]}
}{x \in \text{List(A)},\ y \in \text{List(A) cont}}\;(y \in \text{List(A)}) \notin x \in \text{List(A)}
}{x \in \text{List(A)}[x \in \text{List(A)},\ y \in \text{List(A)}]}
$$

4

$$
\text{I}_2\text{-List}\;\dfrac{
\text{var}\;\dfrac{\dfrac{\mathbf{4'}}{x \in \text{List(A)},\ y \in \text{List(A)},\ v \in A,\ z \in \text{List(A) cont}}}{z \in \text{List(A)}[x \in \text{List(A)},\ y \in \text{List(A)},\ v \in A,\ z \in \text{List(A)}]}
\quad
\text{var}\;\dfrac{\dfrac{\mathbf{4'}}{x \in \text{List(A)}\ y \in \text{List(A)},\ v\in A,\ z \in \text{List(A) cont}}}{v \in A[x \in \text{List(A)}\ y \in \text{List(A)},\ v \in A,\ z \in \text{List(A)}]}
}{\text{cons}(z,v) \in \text{List(A)}[x \in \text{List(A)},\ y \in \text{List(A)},\ v \in A,\ z \in \text{List(A)}]}
$$

4'

$$
\text{F-c}\;\dfrac{
\text{ind-ty}\;\dfrac{
\text{F-cost}\;\dfrac{\text{A type}[\,]}{\text{List(A) type}[\,]}
\quad
\text{F-c}\;\dfrac{
\text{ind-ty}\;\dfrac{
\dfrac{}{\text{A type}[\,]}\quad
\text{F-c}\;\dfrac{
\text{F-cost}\;\dfrac{
\text{ind-ty}\;\dfrac{
\dfrac{}{\text{A type}[\,]}\quad
\text{F-c}\;\dfrac{\text{F-cost}\;\dfrac{\dfrac{}{\text{A type}[\,]}}{\text{List(A) type}[\,]}}{x \in \text{List(A) cont}}\;(x \in \text{List(A)}) \notin [\,]
}{\text{A type}[x \in \text{List(A)}]}
}{\text{List(A) type}[x \in \text{List(A)}]}
}{x \in \text{List(A)},\ y \in \text{List(A) cont}}\;(y \in \text{List(A)}) \notin x \in \text{List(A)}
}{\text{A type}[x \in \text{List(A)},\ y \in \text{List(A)}]}
}{x \in \text{List(A)},\ y \in \text{List(A)},\ v \in A\ \text{cont}}\;(v \in A) \notin (x \in \text{List(A)},\ y \in \text{List(A)}))
}{\text{List(A) type}[x \in \text{List(A)},\ y \in \text{List(A)},\ v \in A]}
}{x \in \text{List(A)}\ y \in \text{List(A)},\ v \in A,\ z \in \text{List(A) cont}}\;(z \in \text{List(A)}) \notin (x \in \text{List(A)},\ y \in \text{List(A)},\ v \in A)
$$

Dimostrazione di correttezza di $El_{List}(y, x,(u,v,z).cons(z,v)) \in List(A)[x \in List(A), y \in List(A)]$

- $\mathrm{El}_{List}(\mathrm{nil}, x,(u,v,z).\mathrm{cons}(z,v)) \to_1 x$ per β_{1List}-*red*
- $\mathrm{El}_{List}(\mathrm{cons}(y,a), x,(u,v,z).\mathrm{cons}(z,v)) \to_1 \mathrm{cons}(\mathrm{El}_{List}(y, x, (u,v,z).\mathrm{cons}(z,v)))$ per β_{2List}-*red* $\Rightarrow$ per $y = \mathrm{nil} \equiv \mathrm{cons}(\mathrm{El}_{List}(\mathrm{nil}, x, (u,v,z).\mathrm{cons}(z,v))) \to_1$ $\mathrm{cons}(x, \mathrm{nil}) \in \mathrm{List}(A) = x$ (dal punto precedente).

Capitolo 5

Tipo della somma disgiunta

Il tipo somma disgiunta è un costruttore di tipo. Questi non è dipendente se da solo, lo diventa solo quando agisce su un tipo dipendente.
Anche con il tipo somma disgiunta si parla di tipo induttivo (accade già per N_1, Nat e List(A)).
Le regole del tipo della somma disgiunta sono le seguenti.

5.1 Regole di Formazione

$$\text{F-+)}\ \frac{\mathrm{B\ type\ [\Gamma]} \qquad \mathrm{C\ type\ [\Gamma]}}{\mathrm{B + C\ type\ [\Gamma]}}$$

5.2 Regole di Introduzione

$$\mathrm{I_1\text{-+})}\ \frac{\mathrm{b \in B[\Gamma]} \qquad \mathrm{B + C\ type[\Gamma]}}{\mathrm{inl(b) \in B + C[\Gamma]}} \qquad \mathrm{I_2\text{-+})}\ \frac{\mathrm{c \in C[\Gamma]} \qquad \mathrm{B + C\ type[\Gamma]}}{\mathrm{inr(c) \in B + C[\Gamma]}}$$

Nelle regole di Introduzione inl e inr sevono, rispettivamente, per introdurre un termine a sinistra e a destra.

5.3 Regole di Eliminazione

$$\text{E-+)}\ \frac{\begin{array}{c}\mathrm{M(z)\ type[\Gamma,\ z \in B + C]} \\ \mathrm{t \in B + C[\Gamma]}\end{array} \qquad \mathrm{e_B(x_1) \in M(inl(x_1))[\Gamma,\ x_1 \in B]} \qquad \mathrm{e_C(x_2) \in M(inr(x_2))[\Gamma,\ x_2 \in C]}}{\mathrm{El_+(t,e_B,e_C) \in M(t)[\Gamma]}}$$

5.4 Regole di Conservazione

$$\mathrm{C_1\text{-+})}\ \frac{\begin{array}{c}\mathrm{M(z)\ type[\Gamma,\ z \in B + C]} \\ \mathrm{b \in B[\Gamma]}\end{array} \qquad \mathrm{e_B(x_1) \in M(inl(x_1))[\Gamma,\ x_1 \in B]} \qquad \mathrm{e_C(x_2) \in M(inr(x_2))[\Gamma,\ x_2 \in C]}}{\mathrm{El_+(inl(b),e_B,e_C) = e_B(b) \in M(inl(b))[\Gamma]}}$$

$$\mathrm{C_2\text{-}+)}\ \frac{\begin{array}{c}\mathrm{M(z)\ type[\Gamma,\ z \in B + C]} \\ \mathrm{c \in C[\Gamma]}\end{array} \qquad \mathrm{e_B(x_1) \in M(inl(x_1))[\Gamma,\ x_1 \in B]} \qquad \mathrm{e_C(x_2) \in M(inr(x_2))[\Gamma,\ x_2 \in C]}}{\mathrm{El_+(inr(c),e_B,e_C) = e_C(c) \in M(inr(c))[\Gamma]}}$$

5.5 Regole di Uguaglianza

$$\mathrm{eq\text{-}F\text{-}+)}\ \frac{\mathrm{B_1 = B_2\ type[\Gamma]} \qquad \mathrm{C_1 = C_2\ type[\Gamma]}}{\mathrm{B_1 + C_1 = B_2 + C_2\ type(\Gamma)}}$$

5.6 Eliminatore dipendente

L'eliminatore ha, anche nel caso della somma disgiunta, la forma dipendente.

$$\mathrm{E_{\mathit{dip}}\text{-}+)}\ \frac{\mathrm{M(z)\ type[\Gamma,\ z \in B + C]} \qquad \mathrm{e_B(x_1) \in M(inl(x_1))[\Gamma,\ x_1 \in B]} \qquad \mathrm{e_C(x_2) \in M(inr(x_2))[\Gamma,\ x_2 \in C]}}{\mathrm{El_+(z,e_B,e_C) \in M(z)[\Gamma,z \in B + C]}}$$

Da sottolineare come con questo eliminatore non definisce solo che si ricorre su B + C, ma da, nascosto, anche un principio di induzione. Inoltre si possono vedere B + C come due proposizioni β e γ (come illustro in 5.8).

5.7 Semantica operazionale della somma disgiunta

La relazione $\rightarrow_1$ viene definita all'interno dei termini con l'uso delle seguenti regole di riduzione:

- β_{1+}-red) $\mathrm{El_+(inl(b),\ e_B,\ e_C) \rightarrow_1 e_B(b)}$
- β_{2+}-red) $\mathrm{El_+(inr(c),\ e_B,\ e_C) \rightarrow_1 e_C(c)}$
- +-red) $\dfrac{\mathrm{t_1 \rightarrow_1 t_2}}{\mathrm{El_+(t_1,\ e_B,\ e_C) \rightarrow_1 El_+(t_2,\ e_B,\ e_C)}}$
- novità della somma disgiunta rispetto al tipo singoletto +-red$_I$) $\dfrac{\mathrm{b_1 \rightarrow_1 b_2}}{\mathrm{inl(b_1) \rightarrow_1 inl(b_2)}}$
- +-red$_{II}$) $\dfrac{\mathrm{c_1 \rightarrow_1 c_2}}{\mathrm{inr(c_1) \rightarrow_1 inr(c_2)}}$

5.8 Osservazioni dal punto di vista logico

Le regole di *Eliminazione/Introduzione* partono dalle regole di congiunzione nella logica (nel caso in cui sia B che C siano proposizioni), inoltre nel caso dell'eliminazione si va a eliminare verso altre proposizioni.
Tale concetto è stato voluto da *Martin-Löf*, per riuscire a interpretare la logica con la teoria dei tipi.

5.8.1 La regola di Formazione

Se si hanno le proposizioni β prop[Γ] e γ prop[Γ] e si interpreta la somma come disgiunzione, allora la regola di *Formazione* diventa

$$\text{F-+)}\ \frac{\beta \text{ prop } [\Gamma] \qquad \gamma \text{ prop } [\Gamma]}{\text{B} \vee \text{C prop } [\Gamma]}$$

Questi ha influenza sia sulla regola *(E-$_{dip}$+)* che *(I-+)*.

La regola di Eliminazione

$$\text{E}_{dip}\text{-+)}\ \frac{\xi \text{ prop}[\Gamma] \qquad \xi \text{ è vero}[\Gamma,\text{supponiamo } \beta \text{ vero}] \qquad \xi \text{ è vero}[\Gamma,\gamma \text{ è vero}]}{\cancel{\text{El}_+(\text{z},\text{e}_B,\text{e}_C)} \in \xi \text{ vero}[\Gamma,\text{z} \in \beta \vee \gamma \text{ vero}]}$$

che, nel calcolo dei sequenti, equivale alla or a sinistra

$$\vee\text{-S)}\ \frac{\beta \vdash_\Gamma \xi \qquad \gamma \vdash_\Gamma \xi}{\beta \vee \gamma \vdash_\Gamma \xi}$$

Allo stesso modo le regole di *Introduzione* si possono vedere in modo molto semplice.

Le regole di Introduzione

$$\text{I}_1\text{-+)}\ \frac{\text{b} \in \beta[\Gamma] \qquad \beta + \gamma \text{ prop}[\Gamma]}{\text{inl(b)} \in \beta \vee \gamma}$$

b $\in \beta \equiv \beta$ vero
$\beta \vee \gamma \equiv \beta \vee \gamma$ vero
Questa, nel calcolo dei sequenti, equivale alla or a destra
Con Δ contesto, allora

$$\vee\text{-D)}\ \frac{\Delta \vdash_\Gamma \beta}{\Delta \vdash_\Gamma \beta \vee \gamma}$$

Lo stesso si può fare sulla seconda regola di *Introduzione*.

$$\text{I}_2\text{-+)}\ \frac{\text{c} \in \gamma[\Gamma] \qquad \beta + \gamma \text{ prop}[\Gamma]}{\text{inr(c)} \in \beta \vee \gamma}$$

c $\in \gamma \equiv \gamma$ vero
$\beta \vee \gamma \equiv \beta \vee \gamma$ vero
Questa, nel calcolo dei sequenti, equivale alla regola di riduzione naturale nella logica
Con Δ contesto, allora

$$\text{I}_2\text{-+)}\ \frac{\Delta \vdash_\Gamma \gamma}{\Delta \vdash_\Gamma \beta \vee \gamma}$$

5.9 Esercizi

1) Si scrivano le regole del tipo booleano come tipo semplice e si provi che è rappresentabile come $N_1 + N_1$.

Soluzione

Definisco il tipo Bool nel modo seguente Bool = true || false
inl(∗) ≡ true
inr(∗) ≡ false
Per definire regole e semantica operazionale uso che $N_1 + N_1$ = Bool.

- **Regole del tipo Bool:**
  - *Regole di Formazione*

$$\text{F-Bool)}\ \frac{\Gamma\ \text{cont}}{\text{Bool type}[\Gamma]}$$

  - *Regole di Introduzione*

$$\text{I}_1\text{-Bool)}\ \frac{\Gamma\ \text{cont}}{\text{true} \in \text{Bool}[\Gamma]} \qquad \text{I}_2\text{-Bool)}\ \frac{\Gamma\ \text{cont}}{\text{false} \in \text{Bool}[\Gamma]}$$

  - *Regole di Eliminazione*

$$\text{E-Bool)}\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{Bool}] \qquad \text{t} \in \text{Bool}[\Gamma] \qquad \begin{array}{c}\text{e}_B \in \text{M(true)}[\Gamma] \\ \text{e}_C \in \text{M(false)}[\Gamma]\end{array}}{\text{El}_{Bool}(\text{t,e}_B,\text{e}_C) \in \text{M(t)}[\Gamma]}$$

$$\text{E-Bool}_{dip})\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{Bool}] \qquad \begin{array}{c}\text{e}_B \in \text{M(true)}[\Gamma] \\ \text{e}_C \in \text{M(false)}[\Gamma]\end{array}}{\text{El}_{Bool}(\text{z,e}_B,\text{e}_C) \in \text{M(z)}[\Gamma,\text{z} \in \text{Bool}]}$$

  - *Regole di Conversione*

$$\text{C}_1\text{-Bool)}\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{Bool}] \qquad \text{e}_B \in \text{M(true)}[\Gamma] \qquad \text{e}_C \in \text{M(false)}[\Gamma]}{\text{El}_{Bool}(\text{true,e}_B,\text{e}_C) = \text{e}_B \in \text{M(true)}[\Gamma]}$$

$$\text{C}_2\text{-Bool)}\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{Bool}] \qquad \text{e}_B \in \text{M(true)}[\Gamma] \qquad \text{e}_C \in \text{M(false)}[\Gamma]}{\text{El}_{Bool}(\text{false,e}_B,\text{e}_C) = \text{e}_C \in \text{M(false)}[\Gamma]}$$

  - *Regole di Uguaglianza*

$$\text{eq-E-Bool)}\ \frac{\text{M(z) type}\ [\Gamma,\ \text{z} \in \text{Bool}] \qquad \text{t} = \text{t'} \in \text{Bool}[\Gamma] \qquad \begin{array}{c}\text{e}_B = \text{s} \in \text{M(true)}[\Gamma] \\ \text{e}_C = \text{u} \in \text{M(false)}[\Gamma]\end{array}}{\text{El}_{Bool}(\text{t,e}_B,\text{e}_C) = \text{El}_{Bool}(\text{t',s,u}) \in \text{M(t')}[\Gamma]}$$

- **Semantica operazionale del tipo Bool:**
  - β_{1Bool}-red) $\text{El}_{Bool}(\text{true, e}_B,\ \text{e}_C) \to_1 \text{e}_B$
  - β_{2Bool}-red) $\text{El}_{Bool}(\text{false, e}_B,\ \text{e}_C) \to_1 \text{e}_C$
  - IF-true) if true then M else N $\to_1$ M

– IF-false) if false then M else N $\rightarrow_1$ N

– Bool-red) $\dfrac{\mathrm{t}_1 \rightarrow_1 \mathrm{t}_2}{\mathrm{El}_{Bool}(\mathrm{t}_1,\ \mathrm{e}_B,\ \mathrm{e}_C) \rightarrow_1 \mathrm{El}_{Bool}(\mathrm{t}_2,\ \mathrm{e}_B,\ \mathrm{e}_C)}$

– IF) $\dfrac{\mathrm{M}_1 \rightarrow_1 \mathrm{M}_1^{\text{`}}}{\text{if } \mathrm{M}_1 \text{ then } \mathrm{M}_2 \text{ else } \mathrm{M}_3 \rightarrow_1 \text{if } \mathrm{M}_1^{\text{`}} \text{ then } \mathrm{M}_2 \text{ else } \mathrm{M}_3}$

Capitolo 6

Tipo dell'uguaglianza proposizionale

Più modi permettono di descrivere il tipo dell'uguaglianza, uno di questi è il tipo dell'uguaglianza proposizionale (o identità proposizionale) di *Martin-Löf*.
È un primo esempio di **tipo dipendente**, e se opportunamente abbinato al costruttore può costruire altri tipi dipendenti.
Inoltre questi è dipendente in modo primitivo e definito dalle regole seguenti.

6.1 Regole di Formazione

$$\text{F-Id)}\ \frac{\text{A type } [\Gamma] \qquad \text{a} \in \text{A}[\Gamma] \qquad \text{b} \in \text{A}[\Gamma]}{\text{Id(A,a,b) type}[\Gamma]}$$

6.2 Regole di Introduzione

$$\text{I-Id)}\ \frac{\text{a} \in \text{A}[\Gamma]}{\text{id(a)} \in \text{Id(A,a,a)}[\Gamma]}$$

6.3 Regole di Eliminazione

$$\text{E-Id)}\ \frac{\begin{array}{c}\text{M}(\text{z}_1,\text{z}_2,\text{z}_3)\text{ type}[\Gamma,\ \text{z}_1 \in \text{A},\ \text{z}_2 \in \text{A},\ \text{z}_3 \in \text{Id(A, z}_1\text{, z}_2)] \\ \text{a} \in \text{A}[\Gamma] \\ \text{b} \in \text{A}[\Gamma] \\ \text{t} \in \text{Id(A,a,b)}[\Gamma]\end{array} \qquad \text{e(x)} \in \text{M(x,x,Id(x))}[\Gamma,\ \text{x} \in \text{A}]}{\text{El}_{Id}\text{(t, (x).e(x))} \in \text{M(a,b,t)}[\Gamma]}$$

6.4 Regole di Conservazione

$$\text{C-Id)}\ \frac{\begin{array}{c}\text{M}(\text{z}_1,\text{z}_2,\text{z}_3)\text{ type}[\Gamma,\ \text{z}_1 \in \text{A},\ \text{z}_2 \in \text{A},\ \text{z}_3 \in \text{Id(A, z}_1\text{, z}_2)] \\ \text{a} \in \text{A}[\Gamma]\end{array} \qquad \text{e(x)} \in \text{M(x,x,Id(x))}[\Gamma,\ \text{x} \in \text{A}]}{\text{El}_{Id}\text{(Id(a), (x).e(x))} = \text{e(a)} \in \text{M(a,a,Id(a))}[\Gamma]}$$

6.5 Regole di Uguaglianza

$$\text{eq-F-Id)}\ \frac{A_1 = A_2\ type[\Gamma] \qquad a_1 = a_2 \in A[\Gamma] \qquad b_1 = b_2 \in A[\Gamma]}{Id(A_1,a_1,b_1) = Id(A_2,a_2,b_2)\ type[\Gamma]}$$

$$\text{eq-I-Id)}\ \frac{a_1 = a_2 \in A[\Gamma]}{id(a_1) = id(a_2) \in Id(A,a_1,a_1)[\Gamma]}$$

6.6 Eliminatore dipendente

L'eliminatore ha anche la forma dipendente.

$$\text{E-Id}_{dip})\ \frac{M(z_1,z_2,z_3)\ type[\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in Id(A,\ z_1,\ z_2)] \qquad e(x) \in M(x,x,Id(x))[\Gamma,\ x \in A]}{El_{Id}(z_3,\ (x).e(x)) \in M(z_1,\ z_2,\ z_3)[\Gamma, z_1 \in A,\ z_2 \in A,\ z_3 \in Id(A,z_1,\ z_2)]}$$

6.7 Lemma dell'Id

È derivabile la regola

$$\frac{a = b \in A[\Gamma]}{id(a) \in Id(A,a,b)}$$

Con $id(a) = id(b)$

Ovvero l'uguaglianza proposizionale rende uguali i termini, definizionalmente, ma non vale il contrario. Percui se $p \in Id(A,a,b)[\Gamma]$ è derivabile $\not\Rightarrow a = b \in A[\Gamma]$ è derivabile.
Formalmente si può trovare con il tipo singoletto, nel modo seguente: "? $\in Id(N_1,x,*)[x \in N_1] \not\Rightarrow x = * \in N_1[x \in N_1]$ (non è vero che x è uguale a $*$)"

Dimostrazione

$$\dfrac{\text{I-Id}\ \dfrac{\text{s-checks}\ \dfrac{\overline{a = b \in A[\Gamma]}}{a \in A[\Gamma]}}{id(a) \in Id(A,a,a)[\Gamma]} \qquad \dfrac{\mathbf{1}}{Id(A,a,a) = Id(A,a,b)\ type[\Gamma]}}{id(a) \in Id(A,a,b)[\Gamma]}\ \text{conv}$$

1

$$\dfrac{\text{ref}\ \dfrac{\text{s-checks}\ \dfrac{\text{s-checks}\ \dfrac{\overline{a = b \in A[\Gamma]}}{a \in A[\Gamma]}}{A\ type[\Gamma]}}{A = A\ type[\Gamma]} \qquad \text{ref}\ \dfrac{\text{s-checks}\ \dfrac{\overline{a = b \in A[\Gamma]}}{a \in A[\Gamma]}}{a = a \in A[\Gamma]} \qquad \overline{a = b \in A[\Gamma]}}{Id(A,a,a)[\Gamma] = Id(A,a,b)[\Gamma]}\ \text{eq-F-Id}$$

6.8 Esercizio di dimostrazione per induzione su addizione con zero

Nel dettaglio l'esercizio presentato in questa sezione, e svolto a lezione, viene risolto in §6.9, esercizio 5.

$$\text{Nat)}\ \varphi(0)\ \&\ \forall_{x\in Nat}(\varphi(\text{x}) \to \varphi(\text{succ(x)})) \to \forall_{z\in Nat}\ \varphi(\text{z})\ (\S 11).$$

Definizione **Somma dei numeri Naturali**
$\text{w} + \text{z} \in \text{Nat}[\text{w} \in \text{Nat},\text{z} \in \text{Nat}]$
$\text{w} + \text{z} \overset{def}{\equiv} \text{El}_{Nat}(\text{z,w,(x,y).succ(y)})$

$\text{w} + 0 = \text{w} \in \text{Nat}[\text{w} \in \text{Nat}]$ per β_{1Nat}*-red*
ma definizionalmente $0 + \text{z} \neq \text{z}$
$\text{El}_{Nat}(\text{z,0,(x,y).succ(y)})$ è in forma normale $\neq$ da z, anch'essa in forma normale
Per cui $0 + \text{z} = \text{z} \in \text{Nat}[\text{z} \in \text{Nat}]$, ricorrendo su z, non è derivabile in teoria dei tipi sui Naturali.

In realtà, si può risolvere $0 + \text{z} = \text{z} \in \text{Nat}[\text{z} \in \text{Nat}]$, non con l'uso dell'uguaglianza definizionale, ma con l'identità proposizionale, nel modo seguente.

Obiettivo: trovare pf $\in \forall_{z\in Nat}$ Id(Nat,0+z,z)

(N.B. So che $\forall_{w\in Nat}$ Id(Nat,w+0,w) [ ] $\equiv \lambda$w.id(w)
Invece pf $\in \forall_{z\in Nat}$ Id(Nat,0+z,z) va dimostrato per induzione.)

Uso il *proof-term* così definito:

$$\text{h}_2(\text{z}_1,\text{z}_2,\text{z}_3) \in \text{Id(Nat,succ(z}_1\text{),succ(z}_2\text{))}[\text{z}_1 \in \text{Nat},\text{z}_2 \in \text{Nat},\text{z}_3 \in \text{Id(Nat,z}_2,\text{z}_3)]$$

$$\text{E-Nat}_{dip}\ \cfrac{\overline{\text{Id(Nat,0+z,z)type}[\text{z} \in \text{Nat}]} \quad \overline{\text{id}(0) \in \text{Id(Nat,0+0,0)}[\ ]} \quad \overline{\text{h}_2(0+\text{x,x,y}) \in \text{Id(Nat,0+succ(x),succ(x))}[\text{x} \in \text{Nat},\text{y} \in \text{Id(Nat,0+x,x)}]}}{\text{I-}\prod\ \cfrac{\text{El}_{Nat}(\text{z,id(0),(x,y).h}_2(0+\text{x,x,y})) \in \text{Id(Nat,0+z,z)}[\text{z} \in \text{Nat}]}{?\ \ (\lambda z.(El_{Nat}(z,id(0),(x,y).h_2(0{+}x,x,y))) \in \forall_{z\in Nat}\ \text{Id(Nat,0+z,z)}[\ ]}}$$

Per la regola *(I-$\prod$)* fare riferimento a §8.8.2.

In conclusione: $\forall_{z\in Nat}$ Id(Nat,0+z,z) è vero

6.9 Esercizi

1) Si dimostri che esiste una funzione $\mathbf{h}_1$ che mi permetta di dimostrare la simmetria dell'uguaglianza. Si dia la prova che $\mathbf{z}_2 = \mathbf{z}_1$ non appena [Γ, $\mathbf{z}_1 \in$ A, $\mathbf{z}_2 \in$ A, $\mathbf{z}_3 \in$ Id(A,$\mathbf{z}_1$,$\mathbf{z}_2$)] con A type[Γ].

$$\mathbf{h_1(z_1,z_2,z_3) \in Id(A,z_2,z_1)[\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in Id(A,\ z_1,\ z_2)]}$$

Soluzione (*si chiede di dimostrare che h_1 è proof term nella forma indicata sopra*)

Assunzione dell'esercizio: Id(A, z_1, z_2) type[Γ, $z_1 \in A$, $z_2 \in A$]

$M(z_1,z_2,z_3) \equiv \mathrm{Id}(A,\ z_2,\ z_1)$
$h_1(z_1,z_2,z_3) \equiv \mathrm{El}_{Id}(z_3,(x).\mathrm{id}(x))$

$$\text{E-Id}_{dip}\ \dfrac{\dfrac{\mathbf{1}}{\mathrm{Id}(A,z_2,z_1)\ \text{type}[\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in \mathrm{Id}(A,\ z_1,\ z_2)]} \qquad \dfrac{\mathbf{2}}{\mathrm{id}(x) \in \mathrm{Id}(A,x,x)[\Gamma,\ x \in A]}}{h_1(z_1,z_2,z_3) \in \mathrm{Id}(A,z_2,z_1)[\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in \mathrm{Id}(A,\ z_1,\ z_2)]}$$

1

$$\text{F-id}\ \dfrac{\dfrac{\mathbf{1}_A}{A\ \text{type}\ [\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in \mathrm{Id}(A,z_1,z_2)]} \quad \dfrac{\mathbf{1}_B}{z_2 \in A[\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in \mathrm{Id}(A,z_1,z_2)]} \quad \dfrac{\mathbf{1}_C}{z_1 \in A[\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in \mathrm{Id}(A,z_1,z_2)]}}{\mathrm{Id}(A,z_2,z_1)\ \text{type}[\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in \mathrm{Id}(A,z_1,z_2)]}$$

$\mathbf{1}_A$

$$\text{ind-ty}\ \dfrac{\dfrac{}{A\ \text{type}\ [\Gamma]} \quad \text{F-c}\ \dfrac{\text{F-Id}\ \dfrac{\dfrac{\mathbf{1}'_A}{A\ \text{type}[\Gamma,\ z_1 \in A,\ z_2 \in A]} \quad \dfrac{\mathbf{1}'_B}{z_1 \in A\ [\Gamma,\ z_1 \in A,\ z_2 \in A]} \quad \dfrac{\mathbf{1}'_C}{z_2 \in A\ [\Gamma,\ z_1 \in A,\ z_2 \in A]}}{\mathrm{Id}(A,z_1,z_2)\text{type}[\Gamma,\ z_1 \in A,\ z_2 \in A]}}{\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in \mathrm{Id}(A,z_1,z_2)\ \text{cont}}\ \begin{matrix}(z_3 \in \mathrm{Id}(A,z_1,z_2)) \notin \\ (\Gamma,\ z_1 \in A,\ z_2 \in A)\end{matrix}}{A\ \text{type}\ [\Gamma,\ z_1 \in A,\ z_2 \in A,\ z_3 \in \mathrm{Id}(A,z_1,z_2)]}$$

$\mathbf{1}'_A$

$$\text{ind-ty}\ \dfrac{\dfrac{}{A\ \text{type}\ [\Gamma]} \quad \text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ \text{type}\ [\Gamma]} \quad \text{F-c}\ \dfrac{\dfrac{}{A\ \text{type}[\Gamma]}}{\Gamma,\ z_1 \in A\ \text{cont}}\ (z_1 \in A) \notin \Gamma}{A\ \text{type}[\Gamma,\ z_1 \in A]}}{\Gamma,\ z_1 \in A,\ z_2 \in A\ \text{cont}}\ (z_2 \in A) \notin (\Gamma,\ z_1 \in A)}{A\ \text{type}[\Gamma,\ z_1 \in A,\ z_2 \in A]}$$

$\mathbf{1}'_B$

$$\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ \text{type}\ [\Gamma]} \quad \text{F-c}\ \dfrac{\dfrac{}{A\ \text{type}[\Gamma]}}{\Gamma,\ z_1 \in A\ \text{cont}}\ (z_1 \in A) \notin \Gamma}{A\ \text{type}[\Gamma,\ z_1 \in A]}}{\Gamma,\ z_1 \in A,\ z_2 \in A\ \text{cont}}\ (z_2 \in A) \notin (\Gamma,\ z_1 \in A)}{z_1 \in A[\Gamma,\ z_1 \in A,\ z_2 \in A]}$$

$\mathbf{1}'_C$

$$\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{A\ \text{type}\ [\Gamma]} \quad \text{F-c}\ \dfrac{\dfrac{}{A\ \text{type}[\Gamma]}}{\Gamma,\ z_1 \in A\ \text{cont}}\ (z_1 \in A) \notin \Gamma}{A\ \text{type}[\Gamma,\ z_1 \in A]}}{\Gamma,\ z_1 \in A,\ z_2 \in A\ \text{cont}}\ (z_2 \in A) \notin (\Gamma,\ z_1 \in A)}{z_2 \in A[\Gamma,\ z_1 \in A,\ z_2 \in A]}$$

2

$$\text{I-id}\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\overline{\mathrm{A\ type}[\Gamma]}}{\Gamma,\ \mathrm{x} \in \mathrm{A\ cont}}\ (\mathrm{x} \in \mathrm{A}) \notin \Gamma}{\mathrm{x} \in \mathrm{A}[\Gamma,\ \mathrm{x} \in \mathrm{A}]}}{\mathrm{id(x)} \in \mathrm{Id(A,x,x)}[\Gamma,\ \mathrm{x} \in \mathrm{A}]}$$

Le derivazioni $\mathbf{1}_B$ e $\mathbf{1}_C$ sono simili a quelle in $\mathbf{1}_A$. Le ometto per evitare una sovradimensionalità del numero di derivazioni.

Verifico che Id(A, $\mathbf{z}_1 \in$ A, $\mathbf{z}_2 \in$ A)type[Γ, $\mathbf{z}_1 \in$ A, $\mathbf{z}_2 \in$ A] è una premessa valida rispetto al giudizio di conclusione Id(A, $\mathbf{z}_2 \in$ A, $\mathbf{z}_1 \in$ A)type[Γ, $\mathbf{z}_1 \in$ A, $\mathbf{z}_2 \in$ A, $\mathbf{z}_3 \in$ Id(A,$\mathbf{z}_1$,$\mathbf{z}_2$)]

$\Delta = \Gamma$, $z_1 \in$ A, $z_2 \in$ A, $z_3 \in$ Id(A,z_1,z_2)
$\theta =$ A, $z_1 \in$ A, $z_2 \in$ A

$$\text{sub-typ}\ \dfrac{\dfrac{(*)}{\mathrm{z}_2 \in \mathrm{A}[\Delta]} \quad \text{sub-typ}\ \dfrac{\dfrac{(*)}{\mathrm{z}_1 \in \mathrm{A}[\Delta,\ \mathrm{w} \in \mathrm{A}]} \quad \text{sub-typ}\ \dfrac{\dfrac{(*)}{\mathrm{w} \in \mathrm{A}[\Delta,\ \mathrm{w} \in \mathrm{A}]} \quad \text{ind-ty}\ \dfrac{\overline{\mathbf{Id}(\theta)\mathbf{type}[\Gamma,\ \mathbf{z}_1 \in \mathbf{A},\ \mathbf{z}_2 \in \mathbf{A}]} \quad \dfrac{(*)}{\Delta,\ \mathrm{w} \in \mathrm{A\ cont}}}{\mathrm{Id}(\theta)\mathrm{type}[\Delta,\ \mathrm{w} \in \mathrm{A}]}}{\mathrm{Id(A,\ w \in A,\ z_2 \in A)type}[\Delta,\ \mathrm{w} \in \mathrm{A}]}}{\mathrm{Id(A,\ w \in A,\ z_1 \in A)type}[\Delta,\ \mathrm{w} \in \mathrm{A}]}}{\mathrm{Id(A,\ z_2 \in A,\ z_1 \in A)type}[\Delta]}$$

Con il simbolo $(*)$ indico quelle derivazioni che terminano su un assioma, che per evitare ripetizioni nella derivazione ho omesso.

2) Si dimostri che esiste una funzione $\mathbf{h}_2$ che mi permetta di dimostrare che il successore preserva l'uguaglianza proposizionale. Si dia la prova che succ($\mathbf{z}_1$) = succ($\mathbf{z}_2$) non appena [$\mathbf{z}_1 \in$ Nat, $\mathbf{z}_2 \in$ Nat, $\mathbf{z}_3 \in$ Id(Nat,$\mathbf{z}_1$,$\mathbf{z}_2$)] con Nat type[Γ] non dipendente.

$$\mathbf{h}_2(\mathbf{z}_1,\ \mathbf{z}_2,\ \mathbf{z}_3) \in \mathbf{Id(Nat,\ succ(z_1),\ succ(z_2))}[\mathbf{z}_1 \in \mathbf{Nat},\ \mathbf{z}_2 \in \mathbf{Nat},\ \mathbf{z}_3 \in \mathbf{Id(Nat,z_1,z_2)}]$$

Soluzione

M(z_1,z_2,z_3) $\equiv$ Id(Nat,succ(z_1),succ(z_2))
h_2(z_1,z_2,z_3) $\equiv$ El_{Id}(z_3,(x).id(succ(x)))

$\Sigma \equiv z_1 \in$ Nat, $z_2 \in$ Nat, $z_3 \in$ Id(Nat,z_1,z_2)

$$\text{E-Id}_{dip}\ \dfrac{\dfrac{\mathbf{1}}{\mathrm{Id(Nat,succ(z_1),succ(z_2))type}\ [\Sigma]} \quad \dfrac{\mathbf{2}}{\mathrm{id(succ(x))} \in \mathrm{Id(Nat,succ(x),succ(x))}[\mathrm{x} \in \mathrm{Nat}]}}{\mathrm{h_2(z_1,\ z_2,\ z_3)} \in \mathrm{Id(Nat,\ succ(z_1),\ succ(z_2))}[\Sigma]}$$

1

$$\text{F-id}\ \dfrac{\dfrac{\mathbf{1}_A}{\mathrm{Nat\ type}\ [\Sigma]} \quad \dfrac{\mathbf{1}_B}{\mathrm{succ(z_1)} \in \mathrm{Nat}[\Sigma]} \quad \dfrac{\mathbf{1}_C}{\mathrm{succ(z_2)} \in \mathrm{Nat}[\Sigma]}}{\mathrm{Id(Nat,succ(z_1),succ(z_2))type}\ [\Sigma]}$$

$\mathbf{1}_A$

$$\text{ind-ty}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]} \qquad \text{F-c}\ \dfrac{\text{F-Id}\ \dfrac{\dfrac{\mathbf{1}^{\backprime}_A}{\text{Nat type}[z_1 \in \text{Nat},\ z_2 \in \text{Nat}]} \quad \dfrac{\mathbf{1}^{\backprime}_B}{z_1 \in \text{Nat}[z_1 \in \text{Nat},\ z_2 \in \text{Nat}]} \quad \dfrac{\mathbf{1}^{\backprime}_C}{z_2 \in \text{Nat}[z_1 \in \text{Nat},\ z_2 \in \text{Nat}]}}{\text{Id}(\text{Nat},z_1,z_2)\text{type}[z_1 \in \text{Nat},\ z_2 \in \text{Nat}]}}{\Sigma\ \text{cont}}\ \begin{array}{l}(z_3 \in \text{Id}(\text{Nat},z_1,z_2)) \notin \\ (z_1 \in \text{Nat},\ z_2 \in \text{Nat})\end{array}}{\text{Nat type}\ [\Sigma]}$$

$\mathbf{1}^{\backprime}_A$

$$\text{ind-ty}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]} \qquad \text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]} \qquad \text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{z_1 \in \text{Nat cont}}\ (z_1 \in \text{Nat}) \notin [\,]}{\text{Nat type}[z_1 \in \text{Nat}]}}{z_1 \in \text{Nat},\ z_2 \in \text{Nat cont}}\ (z_2 \in \text{Nat}) \notin z_1 \in \text{Nat}}{\text{Nat type}[z_1 \in \text{Nat},\ z_2 \in \text{Nat}]}$$

$\mathbf{1}^{\backprime}_B$

La regola di scambio di contesto (ex-te) può venire anche omessa.

$$\text{ex-te}\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]} \qquad \text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{z_2 \in \text{Nat cont}}\ (z_2 \in \text{Nat}) \notin [\,]}{\text{Nat type}[z_2 \in \text{Nat}]}}{z_2 \in \text{Nat},\ z_1 \in \text{Nat cont}}\ (z_1 \in \text{Nat}) \notin z_2 \in \text{Nat}}{z_1 \in \text{Nat}\ [z_2 \in \text{Nat},\ z_1 \in \text{Nat}]} \qquad \text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{z_1 \in \text{Nat cont}}\ (z_1 \in \text{Nat}) \notin [\,]}{\text{Nat type}[z_1 \in \text{Nat}]}}{z_1 \in \text{Nat},\ z_2 \in \text{Nat cont}}\ (z_2 \in \text{Nat}) \notin z_1 \in \text{Nat}}{z_1 \in \text{Nat}\ [z_1 \in \text{Nat},\ z_2 \in \text{Nat}]}$$

$\mathbf{1}^{\backprime}_C$

$$\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]} \qquad \text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{z_1 \in \text{Nat cont}}\ (z_1 \in \text{Nat}) \notin [\,]}{\text{Nat type}[z_1 \in \text{Nat}]}}{z_1 \in \text{Nat},\ z_2 \in \text{Nat cont}}\ (z_2 \in \text{Nat}) \notin z_1 \in \text{Nat}}{z_2 \in \text{Nat}[z_1 \in \text{Nat},\ z_2 \in \text{Nat}]}$$

2

$$\text{I-id}\ \dfrac{\text{I}_2\text{-Nat}\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]}}{x \in \text{Nat cont}}\ (x \in \text{Nat}) \notin [\,]}{x \in \text{Nat}[x \in \text{Nat}]}}{\text{succ}(x) \in \text{Nat}[x \in \text{Nat}]}}{\text{id}(\text{succ}(x)) \in \text{Id}(\text{Nat},\text{succ}(x),\text{succ}(x))[x \in \text{Nat}]}$$

Le derivazioni $\mathbf{1}_B$ e $\mathbf{1}_C$ sono simili a quelle in $\mathbf{1}_A$. Le ometto per evitare una sovradimensionalità del numero di derivazioni.

3) Si dimostri che se a = b $\in$ A[Γ] è derivabile nella teoria dei tipi, con le regole finora introdotte, allora esiste un *proof-term* tale che

$$\mathbf{pf} \in \mathbf{Id(A,a,b)}$$

è derivabile.

Soluzione (*lemma in §6.7*)

Un **pf** è un qualsiasi elemento di un qualsiasi tipo di uguaglianza proposizionale. Per essere conforme con le regole fornite in §6 definisco pf $\equiv$ id(a).

$$\text{conv}\ \dfrac{\text{I-Id}\ \dfrac{\text{s-checks}\ \dfrac{\overline{\mathrm{a = b \in A[\Gamma]}}}{\mathrm{a \in A[\Gamma]}}}{\mathrm{id(a) \in Id(A,a,a)[\Gamma]}} \qquad \text{eq-F-Id}\ \dfrac{\text{ref}\ \dfrac{\text{s-checks}\ \dfrac{\text{s-checks}\ \dfrac{\overline{\mathrm{a = b \in A[\Gamma]}}}{\mathrm{a \in A[\Gamma]}}}{\mathrm{A\ type[\Gamma]}}}{\mathrm{A = A\ type[\Gamma]}} \quad \text{ref}\ \dfrac{\text{s-checks}\ \dfrac{\overline{\mathrm{a = b \in A[\Gamma]}}}{\mathrm{a \in A[\Gamma]}}}{\mathrm{a = a \in A[\Gamma]}} \quad \overline{\mathrm{a = b \in A[\Gamma]}}}{\mathrm{Id(A,a,a) = Id(A,a,b) type[\Gamma]}}}{\mathrm{id(a) \in Id(A,a,b)[\Gamma]}}$$

4) Si dimostri che esiste un *proof-term* pf del tipo

$$\mathbf{pf} \in \mathbf{Id(N_1,x,*)[x \in N_1]}$$

Soluzione

Uso l'eliminatore dipendente del tipo $\mathrm{N_1}$.

$\mathrm{z \equiv x}$
$\mathrm{El_{N1}(z,c) \equiv pf \equiv El_{N1}(x,x.id(*))}$
$\mathrm{M(z) \equiv Id(N_1,x,*)}$
$\mathrm{c \in M(*)[\Gamma] \equiv id(*) \in Id(N_1,*,*)[x \in N_1]}$

$$\text{E-S}_{dip}\ \dfrac{\text{F-Id}\ \dfrac{\overline{\mathrm{N_1\ type[x \in N_1]}} \quad \overline{\mathrm{x \in N_1[x \in N_1]}} \quad \overline{\mathrm{* \in N_1[x \in N_1]}}}{\mathrm{Id(N_1,x,*)type[x \in N_1]}} \qquad \text{I-Id}\ \dfrac{\text{I-S}\ \dfrac{\mathrm{[\,]\ cont}}{\mathrm{* \in N_1[\,]}}}{\mathrm{id(*) \in Id(N_1,*,*)[\,]}}}{\mathrm{El_{N1}(x,x.id(*)) \in Id(N_1,x,*)[x \in N_1]}}$$

5) Si dimostri che esiste un *proof-term* pf tale che sia possibile definire l'addizione tra numeri naturali

$$\mathbf{x + y \in Nat[x \in Nat, y \in Nat]}$$

in modo tale che esistano dei *proof-term* $\mathbf{pf_1}$ e $\mathbf{pf_2}$ tali che

$$\mathbf{pf_1 \in Id(Nat,x+0,x)[x \in Nat]} \qquad \mathbf{pf_2 \in Id(Nat,0+y,y)[y \in Nat]}$$

Soluzione

1. $\text{pf}_1 \in \text{Id}(\text{Nat},x+0,x)[x \in \text{Nat}]$

   $x + y \overset{def}{\equiv} \text{El}_{Nat}(y,x,(w,z).\text{succ}(z))$
   $x + 0 = x \in \text{Nat}[x \in \text{Nat}]$ per β_{1Nat}-red

   $\forall_{x\in Nat}\ \text{Id}(\text{Nat},x+0,x)[\,] \equiv \lambda x.\text{id}(x)$
   $\forall_{x\in Nat}\ \text{Id}(\text{Nat},x+0,x)[\,] \overset{def}{\equiv} \prod_{x\in Nat}\text{Id}(\text{Nat},x+0,x)[\,]$

$$
\text{I-}\textstyle\prod\,\dfrac{\text{conv}\,\dfrac{\text{I-Id}\,\dfrac{\text{var}\,\dfrac{\text{F-c}\,\dfrac{\text{F-Nat}\,\dfrac{[\,]\ \text{cont}}{\text{Nat type}\,[\,]}}{x \in \text{Nat cont}}\ (x \in \text{Nat}) \notin [\,]}{x \in \text{Nat}[x \in \text{Nat}]}}{\text{id}(x) \in \text{Id}(\text{Nat},x,x)[x \in \text{Nat}]} \qquad \text{eq-F-Id}\,\dfrac{\text{ref}\,\dfrac{\text{ind-ty}\,\dfrac{\dfrac{(*)}{\text{Nat type}[\,]} \quad \dfrac{(*)}{x \in \text{Nat cont}}}{\text{Nat type}[x \in \text{Nat}]}}{\text{Nat} = \text{Nat type}[x \in \text{Nat}]} \quad \dfrac{\mathbf{1}}{x = x+0 \in \text{Nat}\ [x \in \text{Nat}]} \quad \text{ref}\,\dfrac{\dfrac{(*)}{x \in \text{Nat}\ [x \in \text{Nat}]}}{x = x \in \text{Nat}\ [x \in \text{Nat}]}}{\text{Id}(\text{Nat},x,x) = \text{Id}(\text{Nat},x+0,x)\ \text{type}[x \in \text{Nat}]}}{\text{id}(x) \in \text{Id}(\text{Nat},x+0,x)[x \in \text{Nat}]}}{\lambda x.\text{id}(x) \in \prod_{x\in Nat}\text{Id}(\text{Nat},x+0,x)[\,]}
$$

1

$\text{El}_{Nat}(0,c,e) \equiv \text{El}_{Nat}(0,x,(w,z).\text{succ}(z))$
$c \in M(0)[\Gamma] \equiv x \in \text{Nat}[x \in \text{Nat}]$

$\Delta \equiv x \in \text{Nat},\ w \in \text{Nat},\ z \in \text{Nat}$
$\Delta' \equiv z \in \text{Nat},\ x \in \text{Nat},\ w \in \text{Nat}$

$$
\text{sym}\,\dfrac{\text{C}_1\text{-Nat}\,\dfrac{\dfrac{(*)}{\text{Nat type}\ [x \in \text{Nat}]} \quad \dfrac{(*)}{x \in \text{Nat}\ [x \in \text{Nat}]} \quad \text{ex-te}\,\dfrac{\text{ind-te}\,\dfrac{\text{I}_2\text{-Nat}_{prog}\,\dfrac{[\,]\ \text{cont}}{\text{succ}(z) \in \text{Nat}\ [z \in \text{Nat}]} \quad \dfrac{\bigstar}{\Delta'\ \text{cont}}}{\text{succ}(z) \in \text{Nat}\ [\Delta']} \quad \dfrac{\bigstar}{\Delta\ \text{cont}}}{\text{succ}(z) \in \text{Nat}\ [\Delta]}}{x+0 = x \in \text{Nat}\ [x \in \text{Nat}]}}{x = x+0 \in \text{Nat}\ [x \in \text{Nat}]}
$$

Ho usato $(*)$ per concludere le derivazioni già svolte all'interno dell'albero.
$\bigstar$ derivazione già risolta negli esercizi precedenti. Questa prevede una combinazioni di istruzioni di indebolimento/assunzione di variabili/formazione di contesto per verificare l'assioma [ ] cont.

2. $\text{pf}_2 \in \text{Id}(\text{Nat},0+y,y)[y \in \text{Nat}]$

   Procedo per induzione. Uso l'eliminatore dipendente del tipo Nat.
   Svolgo la ricorsione su y (altrimenti su x cambierei il contesto e "uscirei" dalle richieste dell'esercizio), dunque

   - Premesse:
     - $M(z)\ \text{type}[\Gamma,z \in \text{Nat}] \equiv \text{Id}(\text{Nat},0+y,y)[y \in \text{Nat}]$
     - $c \in M(0)[\Gamma] \equiv \text{id}(0) \in \text{Id}(\text{Nat},0+0,0)[\,]$
     - $M(z) \equiv M(y)$ per α-eq (dimostrazione in §3.10 esercizio 5)

- $e(y,x) \in M(succ(y))[\Gamma, y \in Nat, x \in M(y)] \equiv$
  $h(0+y,y,x) \in Id(Nat,0+succ(y),succ(y))[\Gamma, y \in Nat, x \in Id(Nat,0+y,y)]$

- Conclusioni:
  - $El_{Nat}(z,c,e) \equiv El_{Nat}(y,id(0),(z,v).h(0+z,z,v))$
  - $M(z)[\Gamma, z \in Nat] \equiv Id(Nat,0+y,y)[y \in Nat]$

$$\text{E-Nat}_{dip}\ \dfrac{\dfrac{\mathbf{1}}{Id(Nat,0+y,y)type[y \in Nat]} \quad \dfrac{\mathbf{2}}{id(0) \in Id(Nat,0+0,0)[\ ]} \quad \dfrac{\mathbf{3}}{h(0+z,z,v) \in Id(Nat,0+succ(z),succ(z))[z \in Nat, v \in Id(Nat,0+z,z)]}}{El_{Nat}(y,id(0),(z,v).h_2(0+z,z,v))Id(Nat,0+y,y)[y \in Nat]}$$

1

$$\text{F-Id}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\ ]\ cont}{Nat\ type[\ ]} \quad (y \in Nat) \notin [\ ]}{y \in Nat\ cont}}{Nat\ type[y \in Nat]} \quad \dfrac{\heartsuit}{0+y \in Nat[y \in Nat]} \quad \text{var}\ \dfrac{\dfrac{(*)}{y \in Nat\ cont}}{y \in Nat[y \in Nat]}}{Id(Nat,0+y,y)type[y \in Nat]}$$

2

$$\text{conv}\ \dfrac{\text{I-Id}\ \dfrac{\text{I-Nat}\ \dfrac{[\ ]\ cont}{0 \in Nat[\ ]}}{id(0) \in Id(Nat,0,0)[\ ]} \quad \text{eq-F-Id}\ \dfrac{\text{ref}\ \dfrac{\text{F-Nat}\ \dfrac{[\ ]\ cont}{Nat\ type[\ ]}}{Nat = Nat\ type[\ ]} \quad \dfrac{\mathbf{2'}}{0 = 0+0 \in Nat[\ ]} \quad \text{ref}\ \dfrac{\dfrac{(*)}{0 \in Nat[\ ]}}{0 = 0 \in Nat[\ ]}}{Id(Nat,0,0) = Id(Nat,0+0,0)\ type[\ ]}}{id(0) \in Id(Nat,0+0,0)[\ ]}$$

2'

$El_{Nat}(0,c,e) \equiv El_{Nat}(0,0,(w,x).succ(x))$
$c \in M(0)[\Gamma] \equiv 0 \in Nat[\ ]$

$$\text{sym}\ \dfrac{\text{C}_1\text{-Nat}\ \dfrac{\text{F-Nat}\ \dfrac{[\ ]\ cont}{Nat\ type[\ ]} \quad \text{I-Nat}\ \dfrac{[\ ]\ cont}{0 \in Nat[\ ]} \quad \text{I}_2\text{-Nat}\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\dfrac{(*)}{Nat\ type[\ ]} \quad (w \in Nat) \notin [\ ]}{w \in Nat\ cont}}{Nat\ type[w \in Nat]} \quad (x \in Nat) \notin w \in Nat}{w \in Nat, x \in Nat\ cont}}{x \in Nat[w \in Nat, x \in Nat]}}{succ(x) \in Nat[w \in Nat, x \in Nat]}}{0+0 = 0 \in Nat[\ ]}}{0 = 0+0 \in Nat[\ ]}$$

3

$\Delta \equiv z \in Nat, v \in Id(Nat,0+z,z)$

$$\text{conv}\ \dfrac{\dfrac{\diamondsuit}{h_2(0+z,z,v) \in Id(Nat,succ(0+z),succ(z))[\Delta]} \quad \dfrac{\mathbf{3'}}{Id(Nat,succ(0+z),succ(z)) = Id(Nat,0+succ(z),succ(z))[\Delta]}}{h_2(0+z,z,v) \in Id(Nat,0+succ(z),succ(z))[\Delta]}$$

3'

$$\text{eq-F-Id}\ \dfrac{\text{ref}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\dfrac{\mathbf{3'}_A}{\mathrm{Id(Nat,0{+}z,z)type[z \in Nat]}}}{\Delta\ \mathrm{cont}}\ \mathrm{(y \in Id(Nat,0{+}z,z))} \notin \mathrm{z \in Nat}}{\mathrm{Nat\ type}[\Delta]}}{\mathrm{Nat = Nat\ type}[\Delta]} \qquad \dfrac{\mathbf{3'}_B}{\mathrm{succ(0{+}z) = 0{+}succ(z) \in Nat}[\Delta]} \qquad \text{ref}\ \dfrac{\text{I}_2\text{-Nat}\ \dfrac{\dfrac{\bigstar}{\mathrm{z \in Nat}[\Delta]}}{\mathrm{succ(z) \in Nat}[\Delta]}}{\mathrm{succ(z) = succ(z) \in Nat}[\Delta]}}{\mathrm{Id(Nat,succ(0{+}z),succ(z)) = Id(Nat,0{+}succ(z),succ(z))}[\Delta]}$$

3'$_A$

$$\text{F-Id}\ \dfrac{\text{F-Nat}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{Nat\ type}[\ ]}\ \mathrm{(z \in Nat)} \notin [\ ]}{\mathrm{z \in Nat\ cont}}}{\mathrm{Nat\ type[z \in Nat]}} \qquad \dfrac{\heartsuit}{\mathrm{0{+}z \in Nat[z \in Nat]}} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\mathrm{z \in Nat\ cont}}}{\mathrm{z \in Nat[z \in Nat]}}}{\mathrm{Id(Nat,0{+}z,z)type[z \in Nat]}}$$

3'$_B$

El_{Nat}(0,c,e) $\equiv$ El_{Nat}(0,succ(0+z),(w,u).succ(u))
c $\in$ M(0)[Γ] $\equiv$ succ(0+z) $\in$ Nat [Δ] $\equiv$ 0 + z + 1 $\in$ Nat [Δ] $\equiv$ 0 + succ(z) $\in$ Nat [Δ]
e(x,y) $\in$ M(succ(x))[Γ, x $\in$ Nat, y $\in$ M(x)] $\equiv$ succ(0+z) + succ(w) $\in$ Nat[Δ] $\equiv$ succ(succ(0+z) + w) $\in$ Nat[Δ] $\equiv$ succ(u) $\in$ Nat[Δ] $\equiv$ u + 1 $\in$ Nat[Δ] $\equiv$ succ(0+z+1+w) $\in$ Nat[Δ] $\equiv$ 0+z+1+w+1 $\in$ Nat[Δ] $\equiv$ 0+z+w+2 $\in$ Nat[Δ] $\equiv$ 0+z+2 $\in$ Nat[Δ]

$$\text{sym}\ \dfrac{\text{C}_2\text{-Nat}\ \dfrac{\dfrac{\bigstar}{\mathrm{Nat\ type}[\Delta]} \qquad \text{I}_2\text{-Nat}\ \dfrac{\text{ind-te}\ \dfrac{\dfrac{\heartsuit}{\mathrm{0{+}z \in Nat[z \in Nat]}} \quad \dfrac{\bigstar}{\Delta\ \mathrm{cont}}}{\mathrm{0{+}z \in Nat}[\Delta]}}{\mathrm{succ(0{+}z) \in Nat}[\Delta]} \qquad \text{I}_2\text{-Nat}\ \dfrac{\text{var}\ \dfrac{\dfrac{\bigstar}{\Delta,\ \mathrm{w \in Nat,\ u \in Nat\ cont}}}{\mathrm{u \in Nat}[\Delta,\ \mathrm{w \in Nat,\ u \in Nat}]}}{\mathrm{succ(u) \in Nat}[\Delta,\ \mathrm{w \in Nat,\ u \in Nat}]}}{\mathrm{0{+}succ(z) = succ(0{+}z) \in Nat}[\Delta]}}{\mathrm{succ(0{+}z) = 0{+}succ(z) \in Nat}[\Delta]}$$

Ho usato (∗) per concludere le derivazioni già svolte all'interno dell'albero.
♡ derivazione già risolta in §3.10, esercizio 3 (necessaria applicazione di α-*eq*).
♢ derivazione già risolta nell'esercizio 2, di questa sezione (necessaria applicazione di α-*eq*).
★ derivazione già risolta negli esercizi precedenti. Prevede una combinazioni di istruzioni di indebolimento/assunzione di variabili/formazione di contesto per verificare l'assioma [ ] cont.

Capitolo 7

Tipo somma indiciata forte

La somma indiciata forte è il potenziamento indiciato della **somma disgiunta binaria** (§5). Tipo induttivo, ovvero generato con il principio d'induzione della regola di eliminazione, di tipi dipendente.

Definizione set-teorica

$$\dot{\bigcup_{x \in B\ set}} \mathbf{C(x)} \qquad \mathbf{(C(x)\ set)\ x \in B}$$

$$\neq \quad \bigcup \mathbf{\{y : \exists\, x \in B \quad y \in C(x)\}}$$

$$\stackrel{def}{=} \mathbf{\{(b,c){:}\ b \in B\ e\ c \in C(b)\}}$$

È l'unione disgiunta di una famiglia di insiemi, definito dalle regole seguenti.

7.1 Regole di Formazione

$$\text{F-}\Sigma)\ \frac{\text{C(x) type } [\Gamma,\ x \in B]}{\sum_{x \in B} \text{C(x) type}[\Gamma]}$$

7.2 Regole di Introduzione

$$\text{I-}\Sigma)\ \frac{\text{b} \in \text{B}[\Gamma] \qquad \text{c} \in \text{C(b)}[\Gamma] \qquad \sum_{x \in B} \text{C(x) type}[\Gamma]}{<\text{b,c}> \in \sum_{x \in B} \text{C(x)}[\Gamma]}$$

7.3 Regole di Eliminazione

$$\text{E-}\Sigma)\ \frac{\text{M(z) type}[\Gamma,\ z \in \sum_{x \in B} \text{C(x)}] \qquad \begin{array}{c} t \in \sum_{x \in B} \text{C(x)}[\Gamma] \\ \text{e(x,y)} \in \text{M}(<\text{x,y}>)[\Gamma,\ x \in B,\ y \in C(x)] \end{array}}{\text{El}_\Sigma(\text{t,(x,y).e(x,y)}) \in \text{M(t)}[\Gamma]}$$

7.4 Regole di Conservazione

$$\text{C-}\Sigma)\ \frac{\mathrm{M(z)\ type}[\Gamma,\ z \in \sum_{x\in B} \mathrm{C(x)}] \qquad \begin{matrix} \mathrm{b} \in \mathrm{B}[\Gamma] \\ \mathrm{c} \in \mathrm{C(b)} \end{matrix} \qquad \mathrm{e(x,y)} \in \mathrm{M(<x,y>)}[\Gamma,\ x \in B,\ y \in C(x)]}{\mathrm{El}_\Sigma(<b,c>,e) = e(b,c) \in M(<b,c>)[\Gamma]}$$

7.5 Regole di Uguaglianza

$$\text{eq-F-}\Sigma)\ \frac{\mathrm{B}_1 = \mathrm{B}_2\ \mathrm{type}[\Gamma] \qquad \mathrm{C}_1(x) = \mathrm{C}_2(x)\ \mathrm{type}[\Gamma,\ x \in \mathrm{B}_1]}{\sum_{x\in B_1} \mathrm{C}_1(x) = \sum_{x\in B_2} \mathrm{C}_2(x)\ \mathrm{type}[\Gamma]}$$

$$\text{eq-I-}\Sigma)\ \frac{\sum_{x\in B} \mathrm{C(x)\ type}[\Gamma] \qquad \mathrm{b}_1 = \mathrm{b}_2 \in \mathrm{B}[\Gamma] \qquad \mathrm{c}_1 = \mathrm{c}_2 \in \mathrm{C(b_1)}[\Gamma]}{<b_1,c_1> = <b_2,c_2> \in \sum_{x\in B} \mathrm{C(x)}[\Gamma]}$$

$$\text{eq-E-}\Sigma)\ \frac{\mathrm{M(z)\ type}[\Gamma,\ z \in \sum_{x\in B} \mathrm{C(x)}] \qquad \begin{matrix} \mathrm{t}_1 = \mathrm{t}_2 \in \sum_{x\in B} \mathrm{C(x)}[\Gamma] \\ \mathrm{e_1(x,y) = e_2(x,y)} \in \mathrm{M(<x,y>)}[\Gamma,\ x \in B,\ y \in C(x)] \end{matrix}}{\mathrm{El}_\Sigma(t_1,e_1) = \mathrm{El}_\Sigma(t_2,e_2) \in M(t_1)[\Gamma]}$$

7.6 Eliminatore dipendente

L'eliminatore ha anche la forma dipendente.

$$\text{E-}\Sigma_{dip})\ \frac{\mathrm{M(z)\ type}[\Gamma,\ z \in \sum_{x\in B} \mathrm{C(x)}] \qquad \mathrm{e(x,y)} \in \mathrm{M(<x,y>)}[\Gamma,\ x \in B,\ y \in C(x)]}{\mathrm{El}_\Sigma(z,(x,y).e(x,y)) \in M(z)[\Gamma,\ z \in \sum_{x\in B} \mathrm{C(x)}]}$$

7.7 Semantica operazionale della somma indiciata forte

La relazione $\to_1$ viene definita all'interno dei termini con l'uso delle seguenti regole di riduzione:

- β_Σ-red) $\mathrm{El}_\Sigma(<b,c>,\ e) \to_1 e(b,c)$
- Σ-red) $\dfrac{t_1 \to_1 t_2}{\mathrm{El}_\Sigma\ (t_1,\ e) \to_1 \mathrm{El}_\Sigma\ (t_2,\ e)}$
- novità della somma indiciata forte rispetto al tipo singoletto
  $\Sigma\text{-red}_I)\ \dfrac{b_1 \to_1 b_2}{<b_1,\ c> \to_1 <b_2,\ c>}$
- $\Sigma\text{-red}_{II})\ \dfrac{c_1 \to_1 c_2}{<b,\ c_1 > \to_1 <b,\ c_2 >}$

7.8 Esercizi

1) Provare a scrivere le regole del prodotto cartesiano A × B di un tipo A con un tipo B.

Svolgimento

$A \times B \stackrel{def}{=} \sum_{x \in A} B$

- **Regole del tipo prodotto cartesiano:**
  - *Regole di Formazione*

$$\text{F-}\times)\ \frac{A\ type\ [\Gamma] \qquad B\ type\ [\Gamma]}{A \times B\ type[\Gamma]}$$

  - *Regole di Introduzione*

$$\text{I-}\times)\ \frac{a \in A[\Gamma] \qquad b \in B[\Gamma]}{<a,b> \in A \times B[\Gamma]}$$

  - *Regole di Proiezione*

$$\text{PJ}_1\text{-}\times)\ \frac{d \in A \times B[\Gamma]}{\pi_1 d \in A[\Gamma]}$$

$$\text{PJ}_2\text{-}\times)\ \frac{d \in A \times B[\Gamma]}{\pi_2 d \in B[\Gamma]}$$

  - *Regole di Uguaglianza delle proiezioni*

$$\text{PJ}_1\text{-eq})\ \frac{a \in A[\Gamma] \qquad b \in B[\Gamma]}{\pi_1(<a,b>) = a \in A[\Gamma]}$$

$$\text{PJ}_2\text{-eq})\ \frac{a \in A[\Gamma] \qquad b \in B[\Gamma]}{\pi_2(<a,b>) = b \in B[\Gamma]}$$

- **Semantica operazionale del tipo prodotto cartesiano:**
  - $\beta_\times$-red$_1$) $El_\Sigma(<a,b>, e) \to_1 e(a,b)$
  - $\beta_\times$-red$_2$) $\pi_1(<a,b>) \to_1 a$
  - $\beta_\times$-red$_3$) $\pi_2(<a,b>) \to_1 b$
  - $\times$-red) $\dfrac{t_1 \to_1 t_2}{El_\Sigma\ (t_1, e) \to_1 El_\Sigma\ (t_2, e)}$
  - $\times$-red$_I$) $\dfrac{a_1 \to_1 a_2}{<a_1, b> \to_1 <a_2, b>}$
  - $\times$-red$_{II}$) $\dfrac{b_1 \to_1 b_2}{<a, b_1> \to_1 <a, b_2>}$

2) Provare a scrivere le regole del tipo delle funzioni $A \to B[\Gamma]$ da un tipo A a un tipo B.

Svolgimento

- **Regole del tipo delle funzioni:**
  - *Regole di Formazione*

$$\text{F-}\!\to) \ \frac{\text{A type } [\Gamma] \qquad \text{B type } [\Gamma]}{\text{A} \to \text{B type}[\Gamma]}$$

  - *Regole di Introduzione*

$$\text{I-}\!\to) \ \frac{\text{b(x)} \in \text{B}[\Gamma,\text{x}\in\text{A}]}{\lambda\text{x}^A.\text{b(x)} \in \text{A} \to \text{B}[\Gamma]}$$

  - *Regole di Eliminazione*

$$\text{E-}\!\to) \ \frac{\text{f} \in \text{A}\to\text{B}[\Gamma] \qquad \text{a} \in \text{A}[\Gamma]}{\text{Ap(f,a)} \in \text{B}[\Gamma]}$$

  - *Regole di Conversione*

$$\text{C-}\!\to) \ \frac{\text{b(x)} \in \text{B}[\Gamma, \text{x} \in \text{A}] \qquad \text{a} \in \text{A}[\Gamma]}{\text{Ap}(\lambda\text{x}^A.\text{b(x),a}) = \text{b(a)} \in \text{B}[\Gamma]}$$

  - *Regole di Uguaglianza*

$$\text{eq-I-}\!\to) \ \frac{\text{b}_1\text{(x)} = \text{b}_2\text{(x)} \in \text{B}[\Gamma, \text{x} \in \text{A}]}{\lambda\text{x}^A.\text{b}_1\text{(x)} = \lambda\text{x}^A.\text{b}_2\text{(x)} \in \text{A} \to \text{B}[\Gamma]}$$

$$\text{eq-E-}\!\to) \ \frac{\text{f}_1 = \text{f}_2 \in \text{A}\to\text{B}[\Gamma] \qquad \text{a}_1 = \text{a}_2 \in \text{A}[\Gamma]}{\text{Ap}(\text{f}_1,\text{a}_1) = \text{Ap}(\text{f}_2,\text{a}_2) \in \text{B}[\Gamma]}$$

- **Semantica operazionale del tipo delle funzioni:**
  - $\beta_{\to}$-red) $\text{Ap}(\lambda\text{x}^A.\text{b(x), a}) \to_1 \text{b(a)}$
  - $\to$-red$_1$) $\dfrac{\text{f}_1 \to_1 \text{f}_2}{\text{Ap}(\text{f}_1, \text{a}) \to_1 \text{Ap}(\text{f}_2, \text{a})}$
  - $\to$-red$_2$) $\dfrac{\text{a}_1 \to_1 \text{a}_2}{\text{Ap}(\text{f}, \text{a}_1) \to_1 \text{Ap}(\text{f}, \text{a}_2)}$
  - $\to$-red) $\dfrac{\text{b}_1 \to_1 \text{b}_2}{\lambda\text{x}^A.\text{b}_1 \to \lambda\text{x}^A.\text{b}_2}$

7.9 Rappresentazione del tipo prodotto cartesiano

Il **tipo somma indiciata forte** è un potenziamento, con tipi dipendenti, del tipo prodotto cartesiano.
Allora ripartendo dal concetto di somma indiciata forte

$$\dot{\bigcup_{x \in B\ set}} \mathrm{C(x)} \qquad \mathrm{(C(x)\ set)\ x} \in \mathrm{B} \stackrel{def}{=} \{\mathrm{(b,c)}: \mathrm{b} \in \mathrm{B\ e\ c} \in \mathrm{C(b)}\}$$

Se $\mathrm{C(x)} \equiv \mathrm{C'}$ dove C' non varia $\quad \Rightarrow \quad \stackrel{def}{=} \{\mathrm{(b,c)}: \mathrm{b} \in \mathrm{B\ e\ c} \in \mathrm{C(b)}\} \simeq \mathrm{B} \times \mathrm{C'}$

$$\mathrm{B} \times \mathrm{C} \stackrel{def}{=} \sum_{x \in B\ set} \mathrm{C}$$

Supposto che

$$\text{F-}\Sigma\ \frac{\mathrm{B\ type}[\Gamma] \qquad \text{ind-ty}\ \dfrac{\mathrm{C\ type}[\Gamma]}{\mathrm{C\ type}[\Gamma,\ \mathrm{x} \in \mathrm{B}]}}{\mathrm{B} \times \mathrm{C} \stackrel{def}{=} \sum \in_{x \in B} \mathrm{C}}$$

Il tipo prodotto cartesiano è definito dalle seguenti regole.

7.9.1 Regole di Formazione

$$\text{F-}\times)\ \frac{\mathrm{B\ type}\ [\Gamma] \qquad \mathrm{C\ type}\ [\Gamma]}{\mathrm{B} \times \mathrm{C\ type}[\Gamma]}$$

7.9.2 Regole di Introduzione

$$\text{I-}\times)\ \frac{\mathrm{b} \in \mathrm{B}[\Gamma] \qquad \mathrm{c} \in \mathrm{C}[\Gamma]}{<\mathrm{b,c}> \in \mathrm{B} \times \mathrm{C}[\Gamma]}$$

7.9.3 Regole di Proiezione

$$\mathrm{PJ_1\text{-}}\times)\ \frac{\mathrm{d} \in \mathrm{B} \times \mathrm{C}[\Gamma]}{\pi_1 \mathrm{d} \in \mathrm{B}[\Gamma]}$$

$$\mathrm{PJ_2\text{-}}\times)\ \frac{\mathrm{d} \in \mathrm{B} \times \mathrm{C}[\Gamma]}{\pi_2 \mathrm{d} \in \mathrm{C}[\Gamma]}$$

7.9.4 Regole di Uguaglianza delle proiezioni

$$\mathrm{PJ_1\text{-}eq})\ \frac{\mathrm{b} \in \mathrm{B}[\Gamma] \qquad \mathrm{c} \in \mathrm{C}[\Gamma]}{\pi_1(<\mathrm{b,c}>) = \mathrm{b} \in \mathrm{B}[\Gamma]}$$

$$\mathrm{PJ_2\text{-}eq})\ \frac{\mathrm{b} \in \mathrm{B}[\Gamma] \qquad \mathrm{c} \in \mathrm{C}[\Gamma]}{\pi_2(<\mathrm{b,c}>) = \mathrm{c} \in \mathrm{C}[\Gamma]}$$

7.9.5 Semantica operazionale del prodotto cartesiano

- $\beta_\times$-red$_1$) $\mathrm{El}_\Sigma(<b,c>, e) \to_1 e(b,c)$
- $\beta_\times$-red$_2$) $\pi_1(<b,c>) \to_1 b$
- $\beta_\times$-red$_3$) $\pi_2(<b,c>) \to_1 c$
- $\times$-red) $\dfrac{t_1 \to_1 t_2}{\mathrm{El}_\Sigma\ (t_1, e) \to_1 \mathrm{El}_\Sigma\ (t_2, e)}$
- $\times$-red$_I$) $\dfrac{b_1 \to_1 b_2}{<b_1, c> \to_1 <b_2, c>}$
- $\times$-red$_{II}$) $\dfrac{c_1 \to_1 c_2}{<b, c_1> \to_1 <b, c_2>}$

7.9.6 Correttezza del prodotto cartesiano

Dimostrazione

Dimostro che $B \times C \stackrel{def}{=} \sum_{x\in B\ set} C$, trovando *($PJ_1$-$\times$)* e *($PJ_2$-$\times$)* usando *(E-$\Sigma_{dip}$)*.

$$M(z) \equiv B$$
$$\sum_{x\in B} C \equiv B \times C$$

Assumo derivabili B type[Γ] e C type[Γ]

1. Sulla prima proiezione

   $\mathrm{El}_\Sigma(z,(x,y).x) \equiv \pi_1 z$

   La prima proiezione sulla coppia canonica $M(<x,y>)$ è x

$$\text{E-}\Sigma_{dip}\ \dfrac{\text{ind-ty}\ \dfrac{\overline{B\ type[\Gamma]} \qquad \text{F-c}\ \dfrac{\text{F-}\times\ \dfrac{\overline{B\ type[\Gamma]} \quad \overline{C\ type[\Gamma]}}{\sum_{x\in B} C\ type[\Gamma]}}{\Gamma, z \in \sum_{x\in B} C\ cont}\ (z \in \sum_{x\in B} C) \notin \Gamma}{B\ type[\Gamma, z \in \sum_{x\in B} C]} \qquad \text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\overline{C\ type[\Gamma]} \quad \text{F-c}\ \dfrac{\overline{B\ type[\Gamma]}}{\Gamma, x \in B\ cont}\ (x \in B) \notin \Gamma}{C\ type[\Gamma, x \in B]}}{\Gamma, x \in B, y \in C\ cont}\ (y \in C) \notin (\Gamma, x \in B)}{x \in B[\Gamma, x \in B, y \in C]}}{\pi_1 z \in B[\Gamma, z \in \sum_{x\in B} C]}$$

   In conclusione

$$\pi_1 z \stackrel{def}{=} \mathrm{El}_\Sigma(z,(x,y).x) \in B[\Gamma, z \in \sum_{x\in B} C]$$

2. Sulla seconda proiezione

   $El_{\Sigma}(z,(x,y).y) \equiv \pi_2 z$
   La seconda proiezione sulla coppia canonica M(<x,y>) è y

$$\text{E-}\Sigma_{dip}\ \frac{\dfrac{(*)}{\text{B type}[\Gamma,\ z \in \sum_{x\in B} \text{C}]} \qquad \text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{\text{C type}[\Gamma]} \quad \text{F-c}\ \dfrac{\dfrac{}{\text{B type}[\Gamma]}}{\Gamma,\text{x} \in \text{B cont}}\ (\text{x} \in \text{B}) \notin \Gamma}{\text{C type}[\Gamma,\text{x} \in \text{B}]}}{\Gamma,\text{x}\in\text{B},\text{y}\in\text{C cont}}\ (\text{y}\in\text{C})\notin(\Gamma,\ \text{x}\in\text{B})}{\text{y} \in \text{C}[\Gamma,\text{x}\in\text{B},\text{y}\in\text{C}]}}{\pi_2\text{z} \in \text{C}[\Gamma,\text{z} \in \sum_{x\in B} \text{C}]}$$

   In conclusione

$$\pi_2 z \overset{def}{=} El_{\Sigma}(z,(x,y).y) \in C[\Gamma, z \in \sum_{x\in B} C]$$

   Ho concluso con $(*)$ i sequenti già risolti, per evitare ripetizioni.

Per verificare che effettivamente $\pi_1\mathbf{z}$ e $\pi_2\mathbf{z}$ siano *(PJ_1-$\times$)* e *(PJ_2-$\times$)*, utilizzo le riduzioni $\beta_{\times}$-*red*$_1$ e $\beta_{\times}$-*red*$_2$.
La riduzione $El_{\Sigma}(<b,c>, e) \to_1 e(b,c)$ riscritta con sostituzioni diventa $El_{\Sigma}(<b,c>, e) \to_1 e[\frac{x}{b}, \frac{y}{c}]$

Allora applicata a una coppia <b,c>:

- $\pi_1(<b,c>) \overset{def}{=} El_{\Sigma}(z,(x,y).x) = El_{\Sigma}(<b,c>,(x,y).x) \to_1 x[\frac{x}{b}, \frac{y}{c}] \to b$
- $\pi_2(<b,c>) \overset{def}{=} El_{\Sigma}(z,(x,y).y) = El_{\Sigma}(<b,c>,(x,y).y) \to_1 y[\frac{x}{b}, \frac{y}{c}] \to c$

Entrambe le $\beta_{\times}$-*red* sono verificate. Percui la definizione $B \times C \overset{def}{=} \sum_{x\in B} C$ è giustificata.

7.9.7 Esempi

L'unione disgiunta non è l'unione insiemistica e questo lo si vede chiaramente dalla relazione sotto
$\bigcup_{x\in Nat} Nat \equiv \{(x,y) \bullet x \in Nat, y \in Nat\} \equiv Nat \times Nat \neq \bigcup_{x\in Nat} Nat\ \{y\text{: } y \in Nat,$ per un certo $x \in Nat\}$
La prima parte della relazione, se la famiglia è costante ingloba il prodotto cartesiano.

Un esempio di unioni genuine, che si possono formare con tipi che non coincidono con il prodotto cartesiano, è $\sum_{x\in Nat} Mat(x \bullet x)$, ove Mat sono le matrici

quadratiche.
A livello insiemisitico potremmo dover proiettare, data una coppia <x,k> (con x elementi e k una certa matrice), sull'indice o sull'elemento di cui stiamo parlando. Dunque dovremmo poter avere a disposizione le due proiezioni *(PJ_1-$\times$)* e *(PJ_2-$\times$)* anche su famiglie non costanti.

Per individuare correttamente $\pi_1(z)$ e $\pi_2(z)$ assumo che $\sum_{x\in B} C(x)\ type[\Gamma]$ sia derivabile. Tale ipotesi è vincolante per l'ottenimento della famiglia.

Definisco $\omega = z \in \sum_{x\in B} C(x)$

1. Per la prima proiezione

$$
\text{E-}\Sigma_{dip}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{B\ type[\Gamma]} \qquad \text{F-c}\ \dfrac{\dfrac{}{\sum_{x\in B} C(x) type[\Gamma]}}{\Gamma,\ \omega\ cont}\ \omega \notin \Gamma}{B\ type[\Gamma,\ \omega]} \qquad \text{var}\ \dfrac{\text{F-c}\ \dfrac{\dfrac{}{C(x)\ type[\Gamma, x \in B]}}{\Gamma, x \in B, y \in C(x)\ cont}\ (y \in C(x)) \notin (\Gamma, x \in B)}{x \in B[\Gamma, x \in B, y \in C(x)]}}{El_\Sigma(z,(x,y).x) \in B[\Gamma,\ \omega]}
$$

In conclusione

$$\pi_1 z \overset{def}{=} El_\Sigma(z,(x,y).x) \in B[\Gamma,\omega]$$

2. Per la seconda proiezione
$\pi_2 z \overset{def}{=} El_\Sigma(z,(x,y).y) \in C(\pi_1(z))[\Gamma,\omega]$

Allora usando che $M(z) \equiv C(\pi_1 z) \Rightarrow M(<x,y>) \equiv C(\pi_1 <x,y>)$

$$
\text{E-}\Sigma_{dip}\ \dfrac{\dfrac{\mathbf{1}}{C(\pi_1 z)\ type[\Gamma,\ \omega]} \qquad \dfrac{\mathbf{2}}{y \in C(\pi_1 <x,y>)[\Gamma, x \in B, y \in C(x)]}}{El_\Sigma(z,(x,y).y) \in C(\pi_1 z)[\Gamma,\ \omega]}
$$

1

$$
\text{sub-typ}\ \dfrac{\text{PJ}_1\text{-}\times\ \dfrac{\dfrac{(*)}{z \in \sum_{x\in B} C(x)[\Gamma,\omega]}}{\pi_1 z \in B[\Gamma,\omega]} \qquad \text{ex-ty}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{C(x)\ type[\Gamma, x \in B]} \qquad \dfrac{(*)}{\Gamma, x \in B, \omega\ cont}}{C(x)\ type[\Gamma, x \in B, \omega]} \qquad \text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{B\ type[\Gamma]} \qquad \text{F-c}\ \dfrac{\text{F-}\Sigma\ \dfrac{\dfrac{}{C(x)\ type[\Gamma,\ x \in B]}}{\sum_{x\in B} C(x)\ type[\Gamma]}}{\Gamma,\omega\ cont}\ \omega \notin \Gamma}{B\ type[\Gamma,\omega]}}{\Gamma,\omega,\ x \in B\ cont}\ (x \in B) \notin (\Gamma,\omega)}{C(x)\ type[\Gamma,\omega,\ x \in B]}}{C(\pi_1 z)\ type[\Gamma,\omega]}
$$

2

$$\text{conv}\ \dfrac{\text{var}\ \dfrac{\dfrac{(*)}{\Gamma,x \in B,y \in C(x)\ cont}}{y \in C(x)[\Gamma,x \in B,y \in C(x)]} \qquad \text{sub-eq-typ}\ \dfrac{\dfrac{\mathbf{2}^A}{x = \pi_1 <x,y>\in B[\Gamma,x \in B,y \in C(x)]} \qquad \dfrac{\mathbf{2}^B}{C(w)\ type[\Gamma,x \in B,y \in C(x),w \in B]}}{C(x) = C(\pi_1 <x,y>)\ type[\Gamma,x \in B,y \in C(x)]}}{y \in C(\pi_1 <x,y>)[\Gamma,x \in B,y \in C(x)]}$$

$\mathbf{2}^A$

$$\text{sym}\ \dfrac{\text{ind-te-eq}\ \dfrac{\text{C-}\Sigma\ \dfrac{\dfrac{(*)}{B\ type[\Gamma]} \quad \dfrac{(*)}{x \in B[\Gamma]} \quad \dfrac{(*)}{y \in C(x)} \quad \dfrac{(*)}{x \in B[\Gamma,\ x \in B,\ y \in C(x)]}}{\pi_1 <x,y> = x \in B[\Gamma]} \qquad \dfrac{(*)}{\Gamma,x \in B,y \in C(x)\ cont}}{\pi_1 <x,y> = x \in B[\Gamma,x \in B,y \in C(x)]}}{x = \pi_1 <x,y>\in B[\Gamma,x \in B,y \in C(x)]}$$

$\mathbf{2}^B$

$$\text{ex-ty}\ \dfrac{\text{ind-ty}\ \dfrac{\alpha\text{-eq}\ \dfrac{\dfrac{}{C(x)\ type[\Gamma,x \in B]}}{C(w)type[\Gamma,w \in B]} \qquad \dfrac{(*)}{\Gamma,w \in B,x \in B,y \in C(x)\ cont}}{C(w)\ type[\Gamma,w \in B,x \in B,y \in C(x)]} \qquad \dfrac{(*)}{\Gamma,x \in B,y \in C(x),w \in B\ cont}}{C(w)\ type[\Gamma,x \in B,y \in C(x),w \in B]}$$

In conclusione vale $\pi_2 z$

Ho concluso con **(∗)** una derivazione per semplificare o evitare ripetizioni nella derivazione stessa.

7.10 Usi del tipo somma indiciata per rappresentare l'assioma di separazione e di quantificazione universale

Il tipo $\sum_{x \in B} \mathbf{C(x)}$ viene usato per rappresentare:

1. **uso *set* teoretico:** l'unione indiciata disgiunta insiemistica (C(x) set[x ∈ B]);
2. **uso *set* teoretico:** con assioma di separazione;
3. **uso logico:** proposizionale.

Il punto (1) l'ho già spiegato nella prima parte di questo capitolo, ora presento le implicazioni di (2) e (3).

7.10.1 L'assioma di separazione

L'assioma di separazione dice che dato B, insieme, esiste l'insieme ottenuto per separazione da B tramite φ(x), definito come {x ∈ B | φ(x)} con φ(x) predicato. Perciò per ogni y insieme, y & {x ∈ B | φ(x)} ⇔ φ(y) vale (come già enunciato in §1.3).
Si può, dunque, simulare l'insieme degli y per cui vale φ(y) nel momento in cui si ha una proposizione

$$\text{F-}\Sigma\ \frac{\varphi(\mathrm{x})\ \cancel{prop}_{\mathbf{type}}[\Gamma,\mathrm{x} \in \mathrm{B}]}{\sum\limits_{x\in B} \varphi(\mathrm{x})\ \cancel{type}_{\mathbf{set}}[\Gamma]}$$

dove, nella regola sopra, prop viene pensato come un type.

Dimostrazione

- (⇐) da $\sum\limits_{x\in B} \varphi$(x) significa avere b ∈ B + pf ∈ φ(b), e quindi b soddisfa φ(x). Ecco che non è solo b ma <b,pf> ∈ {x ∈ B | φ(x)} con pf ∈ φ(b) $_{I-\Sigma}$ ⇐ φ(b).

- (⇒) z ∈ {x ∈ B | φ(x)} con z che sarebbe una coppia e {x ∈ B | φ(x)} ≡ $\sum\limits_{x\in B} \varphi$(x) ⇏ φ(z) vale, in quanto z è una coppia tipata.
Ma posso tuttavia dimostrare che $\varphi(\pi_1 z)$ vale usando le proiezioni. Ecco che π_2(z) ∈ $\varphi(\pi_1$z). π_2(z) è cosi *proof term* per cui $\varphi(\pi_1$z) vale.

Questo ha permesso a *Martin-Löf* di inglobare la *set theory* all'interno della teoria dei tipi.

7.10.2 Proposizionale

$$\text{F-}\Sigma\ \frac{\mathrm{C(x)\ type}[\Gamma,\mathrm{x} \in \mathrm{B}_{set}] \equiv \varphi(\mathrm{x})\ \mathrm{prop}}{\sum\limits_{x\in B} \varphi(\mathrm{x})\ \cancel{type}_{\mathbf{prop}}[\Gamma] \stackrel{def}{=} \exists_{x\in B}\varphi(x)}$$

È la **quantificazione esistenziale**, in logica, di φ(x) che varia sull'insieme B.

Giustificazione

ϕ prop$[\Gamma, x_1 \in \varphi_1, ..., x_n \in \varphi_n]$
Sotto ipotesi φ_i prop[Γ]   i = 1..n
e φ prop[Γ] con prop ≡ type (ovvero tipi/*set* delle loro dimostrazioni)

Definizione ***proof-term***
t ∈ φ[Γ] dove φ è prop

Se devo dire che pf ∈ $\phi[\Gamma, x_1 \in \varphi_1, ..., x_n \in \varphi_n] \equiv \phi$ è vero$[\Gamma, \varphi_1$ vero,...,φ_n vero] (nessuno degli elementi del contesto dipende dal precedente) ≡ $\varphi_1, ..., \varphi_n$ $\vdash_\Gamma \phi$.

Sappiamo che la logica lavora con variabili *untyped*, dunque noi ora aggiungiamo in $\exists_{x \in B}\ \varphi$(x) l'informazione sul tipo.

Per quanto assunto sopra, la regola di *Introduzione* viene nel modo sequente riscritta

Assumendo che $\exists_{x \in B}\ \varphi$(x) type[Γ] sia ben formato

$$\text{I-}\Sigma\ \frac{\mathrm{b} \in \mathrm{B}_{set}[\Gamma] \qquad \mathrm{c} \in \varphi(\mathrm{b})[\Gamma]}{\cancel{<b, c>} \in \sum_{x \in B} \mathrm{C(x)}[\Gamma]}$$

$\mathrm{c} \in \varphi(\mathrm{b})[\Gamma] \equiv \varphi(\mathrm{b})$ è vero $[\Gamma,\mathrm{x}_1 \in \varphi_1,...,\mathrm{x}_n \in \varphi_n] \equiv \gamma_1,...,\gamma_n \vdash \varphi(\mathrm{b})$
$\sum\limits_{x \in B} \mathrm{C(x)}[\Gamma] \equiv \exists\ \mathrm{x} \in \mathrm{B}\ \varphi(\mathrm{x})$ vero[Γ]

In conclusione, riscritta, la regola di *Introduzione*, con il calcolo dei sequenti diventa

$$\text{I-}\Sigma\ \frac{\gamma_1,...,\gamma_n \vdash_\Gamma \varphi(\mathrm{b})}{\gamma_1,...,\gamma_n \vdash_\Gamma \exists\ \mathrm{x} \in \mathrm{B}\ \varphi(\mathrm{x})}$$

Che non è altro che la regola a destra dell'esiste.

Ora definisco la regola dell'*Eliminatore dipendente*.
Per farlo suppongo di avere un'altra proposizione definita come segue

$$\frac{\phi\ \mathrm{prop}[\Gamma]}{\phi\ \mathrm{prop}[\Gamma,\mathrm{z} \in \sum_{x \in B} \varphi(x)]}$$

$$\text{E-}\Sigma_{dip}\ \frac{\phi\ \mathrm{prop}[\Gamma] \qquad \mathrm{e(x,y)} \in \phi[\Gamma,\mathrm{x} \in \mathrm{B},\mathrm{y} \in \varphi(\mathrm{x})]}{\gamma \in \phi[\Gamma,\mathrm{z} \in \exists\ \mathrm{x} \in \mathrm{B}\ \varphi(\mathrm{x})]}$$

$\mathrm{e(x,y)} \equiv \varphi$ vero
$\phi[\Gamma,\mathrm{x} \in \mathrm{B},\mathrm{y} \in \varphi(\mathrm{x})] \equiv \phi$ è vero$[\Gamma,\mathrm{x} \in \mathrm{B},\varphi(\mathrm{x})$ vero]
$\gamma \in \phi \equiv \phi$ vera
$\mathrm{z} \in \exists\ \mathrm{x} \in \mathrm{B}\ \varphi(\mathrm{x}) \equiv \exists\ \mathrm{x} \in \mathrm{B}\ \varphi(\mathrm{x})$ vero

In conclusione, riscritta, la regola dell'*Eliminatore dipendente*, con il calcolo dei sequenti diventa

$$\text{E-}\Sigma_{dip}\ \frac{\gamma_1,...,\gamma_n\ \varphi(\mathrm{x})\ \text{vero} \vdash_{\Gamma,x \in B} \phi}{\gamma_1,...,\gamma_n,\ \exists\ \mathrm{x} \in \mathrm{B}\ \varphi(\mathrm{x}) \vdash_\Gamma \phi}$$

Che non è altro che la regola a sinistra dell'esiste.

7.10.3 Conclusioni sugli usi del tipo della somma indiciata disgiunta

Per

1. $\sum_{x \in B} C(x)$ è un *set* (dove sia x ∈ B che C(x) sono rispettivamente dei *set*) ⇒ B × C è il prodotto cartesiano

2. $\sum_{x \in B} \varphi(x)$ dove φ è una proposizione ⇒ rappresenta l'assioma di separazione $\{x \in B \mid \varphi(x)\}$
   Esempio: $\sum_{z \in List(Nat)}$ Id(Nat,lh(z),2) ⇒ rappresenta il sottoinsieme delle liste di lunghezza 2

3. $\exists\, x \in B\ \varphi(x) \Rightarrow \sum_{x \in B} \varphi(x)$, ove $\varphi(x)$ e $\sum_{x \in B} \varphi(x)$ sono proposizioni

Questo serve per:

- capire in che modo formalizzare;
- comprendere il significato della teoria dei tipi che sto leggendo.

7.11 Congiunzione come tipo prodotto cartesiano

Il tipo prodotto cartesiano permette di interpretare la congiunzione della logica.

Date due proposizioni φ prop[Γ] e ψ prop[Γ]

$$\varphi \,\&\, \psi \stackrel{def}{=} \varphi \times \psi$$

L'uguaglianza vale in quanto φ prop[Γ] ≡ φ type[Γ], che è il tipo delle sue dimostrazioni

La regola di *Introduzione* del prodotto cartesiano è la seguente

$$\text{I-}\times\ \frac{\mathrm{a} \in \varphi[\Gamma] \qquad \mathrm{b} \in \psi[\Gamma]}{< a, b > \in \varphi \times \psi\ [\Gamma]}$$

⇒ a ∈ φ[Γ] ≡ φ true[Γ]
⇒ b ∈ ψ[Γ] ≡ ψ true[Γ]
⇒ $\varphi \times \psi \equiv \varphi \,\&\, \psi$
⇒ $< a, b > \in \varphi \times \psi$ [Γ] ≡ $\varphi \,\&\, \psi$ vero[Γ]

(1) Dunque φ true[Γ] e ψ true[Γ] ⇒ $\varphi \,\&\, \psi$ true[Γ].

Con la regole di *Eliminazione* riesco a provare anche il verso opposto dell'implicazione. In quanto d ∈ $\varphi \times \psi$[Γ] è come dire $\varphi \,\&\, \psi$ true[Γ], per cui, grazie alle regole di Proiezione del prodotto cartesiano

$$\text{PJ}_1\text{-}\times\ \frac{\mathrm{d} \in \varphi \,\&\, \psi[\Gamma]}{\pi_1 \mathrm{d} \in \varphi\ [\Gamma]}$$

$\mathrm{d} \in \varphi \ \& \ \psi[\Gamma] \equiv \varphi \ \& \ \psi \ \mathrm{true}[\Gamma]$
$\pi_1\mathrm{d} \in \varphi \ [\Gamma] \equiv \varphi \ \mathrm{vero}[\Gamma]$

$$\mathrm{PJ}_2\text{-}\times \ \frac{\mathrm{d} \in \varphi \ \& \ \psi[\Gamma]}{\pi_2\mathrm{d} \in \psi \ [\Gamma]}$$

$\mathrm{d} \in \varphi \ \& \ \psi[\Gamma] \equiv \varphi \ \& \ \psi \ \mathrm{true}[\Gamma]$
$\pi_2\mathrm{d} \in \varphi \ [\Gamma] \equiv \psi \ \mathrm{vero}[\Gamma]$

(2) Dunque con le *regole di Eliminazione* si è mostrato come se $\varphi \ \& \ \psi$ true$[\Gamma]$ $\Rightarrow \varphi$ true$[\Gamma]$ e ψ true$[\Gamma]$.

Quanto definito in (1) e (2) permettono di concludere che date φ prop$[\Gamma]$ e ψ prop$[\Gamma]$ (interpretate come tipi delle loro dimostrazioni), è corretto definire $\varphi \ \& \ \psi \overset{def}{=} \varphi \times \psi$.
Questo perchè una formula è vera quando φ true$[\Gamma]$ sse *(def)* $\exists$ pf $\in \varphi[\Gamma]$. Tale definizione di *proof-term* rende vera entrambe le proprietà (1) e (2), e quindi si può ben interpretare la congiunzione come prodotto cartesiano.

In conclusione $\varphi \ \& \ \psi \overset{def}{=} \varphi \times \psi \equiv \sum_{x \in \varphi} \psi$.

Nella teoria dei tipi semplici $\varphi \ \& \ \psi \overset{def}{=} \varphi \times \psi$ era già stato introdotto da *Curry-Howard*.

Capitolo 8

Tipo delle funzioni

Il tipo delle funzione è definito dalle regole seguenti.

8.1 Regole di Formazione

$$\text{F-}\rightarrow)\ \frac{\text{B type } [\Gamma] \qquad \text{C type } [\Gamma]}{\text{B} \rightarrow \text{C type}[\Gamma]}$$

8.2 Regole di Introduzione

$$\text{I-}\rightarrow)\ \frac{\text{c(x)} \in \text{C}[\Gamma,\text{x} \in \text{B}]}{\lambda\text{x}^{B}.\text{c(x)} \in \text{B} \rightarrow \text{C}[\Gamma]}$$

c è una meta-variabile, con il quale si indica un termine che può dipendere o meno da x.

8.3 Regole di Eliminazione

$$\text{E-}\rightarrow)\ \frac{\text{f} \in \text{B} \rightarrow \text{C}[\Gamma] \qquad \text{b} \in \text{B}[\Gamma]}{\text{Ap(f,b)} \in \text{C}[\Gamma]}$$

Il tipo non è induttivo, per cui la regola di eliminazione non definisce un ricorsivo, nè principi di induzione.

8.4 Regole di Conservazione

$$\text{C-}\rightarrow)\ \frac{\text{c(x)} \in \text{C}[\Gamma,\ \text{x} \in \text{B}] \qquad \text{b} \in \text{B}[\Gamma]}{\text{Ap}(\lambda\text{x}^{B}.\text{c(x)},\text{b}] = \text{c(b)}\in \text{C}[\Gamma]}$$

8.5 Regole di Uguaglianza

$$\text{eq-E-}\!\!\rightarrow)\ \frac{\mathrm{f}_1 = \mathrm{f}_2 \in \mathrm{B} \rightarrow \mathrm{C}[\Gamma] \qquad \mathrm{b}_1 = \mathrm{b}_2 \in \mathrm{B}[\Gamma]}{\mathrm{Ap}(\mathrm{f}_1,\mathrm{b}_1) = \mathrm{Ap}(\mathrm{f}_2,\mathrm{b}_2) \in \mathrm{C}[\Gamma]}$$

$$\text{eq-I-}\!\!\rightarrow)\ \frac{\mathrm{c}_1(\mathrm{x}) = \mathrm{c}_2(\mathrm{x}) \in \mathrm{C}[\Gamma,\mathrm{x} \in \mathrm{B}]}{\lambda \mathrm{x}^B.\mathrm{c}_1(\mathrm{x}) = \lambda \mathrm{x}^B.\mathrm{c}_2(\mathrm{x}) \in \mathrm{B} \rightarrow \mathrm{C}[\Gamma]}$$

La regola (eq-I-→) prende anche il nome di ξ-rule ed è una regola difficile da modellare.

8.6 Semantica operazionale del tipo funzione

La relazione $\rightarrow_1$ viene definita all'interno dei termini con l'uso delle seguenti regole di riduzione:

- $\beta_{\rightarrow}$-red) $\mathrm{Ap}(\lambda \mathrm{x}.\mathrm{c}(\mathrm{x}),\mathrm{b}) \rightarrow_1 \mathrm{c}(\mathrm{b})$
  per la programmazione: $\mathrm{Ap}(\lambda \mathrm{x}.\mathrm{c},\mathrm{b}) \rightarrow_1 \mathrm{c}[\frac{x}{b}]$
- $\rightarrow\text{-red}_1)\ \dfrac{\mathrm{f}_1 \rightarrow_1 \mathrm{f}_2}{\mathrm{Ap}(\mathrm{f}_1,\mathrm{b}) \rightarrow_1 \mathrm{Ap}(\mathrm{f}_2,\mathrm{b})}$
- $\rightarrow\text{-red}_2)\ \dfrac{\mathrm{b}_1 \rightarrow_1 \mathrm{b}_2}{\mathrm{Ap}(\mathrm{f},\mathrm{b}_1) \rightarrow_1 \mathrm{Ap}(\mathrm{f},\mathrm{b}_2)}$
- novità della funzioni rispetto al tipo singoletto $\rightarrow\text{-red})\ \dfrac{\mathrm{c}_1 \rightarrow_1 \mathrm{c}_2}{\lambda \mathrm{x}.\mathrm{c}_1 \rightarrow_1 \lambda \mathrm{x}.\mathrm{c}_2}$

8.7 Osservazioni dal punto di vista logico

8.7.1 La regola di Introduzione

Questo tipo è importante perchè c'è di mezzo la logica. Se difatti prendo la regola di *Introduzione* e ci metto le preposizioni

$$\text{F-}\!\!\rightarrow)\ \frac{\beta\ \mathrm{prop}[\Gamma] \qquad \gamma\ \mathrm{prop}[\Gamma]}{\beta \rightarrow \gamma\ \mathrm{prop}[\Gamma]}$$

Ove $\beta \rightarrow \gamma$ è un'implicazione logica.

$\beta \rightarrow \gamma$ *(implicazione)* $\quad = \quad$ $\beta \rightarrow \gamma$ *(tipo)*
Questo risulta vero perchè (I-→) afferma che se si ha

$$\text{I-}\!\!\rightarrow)\ \frac{\mathrm{c}(\mathrm{x}) \in \gamma[\Gamma,\Delta,\mathrm{x} \in \beta]}{\cancel{\lambda \mathrm{x}.\mathrm{c}(\mathrm{x})} \in \beta \rightarrow \gamma[\Gamma,\Delta]}$$

$\mathrm{c}(\mathrm{x}) \in \gamma \equiv \gamma$ vero
$\mathrm{x} \in \beta \equiv \beta$ vero
$\beta \rightarrow \gamma \equiv \beta \rightarrow \gamma$ vero

Che non è altro che la regola, nel calcolo dei sequenti, dell'implica a destra

$$\rightarrow\text{-D})\ \frac{\varphi_1, ..\varphi_n, \beta \vdash_\Gamma \gamma}{\varphi_1, ..\varphi_n \vdash_\Gamma \beta \rightarrow \gamma}$$

8.7.2 La regola di Eliminazione (Modus Pones)

$$\text{E-}\rightarrow)\ \frac{\mathrm{f} \in \beta \rightarrow \gamma[\Gamma] \qquad \mathrm{b} \in \beta[\Gamma]}{? \in \gamma[\Gamma]}$$

f $\in \beta \rightarrow \gamma \equiv \beta \rightarrow \gamma$ vero
b $\in \beta \equiv \beta$ vero
? $\in \gamma \equiv \gamma$ vero
Che non è altro che la regola, nel calcolo dei sequenti, del Modus Pones

$$\text{MP})\ \frac{\gamma_1, ..\gamma_n \vdash_\Gamma \beta \rightarrow \gamma \qquad \gamma_1, ..\gamma_n \vdash_\Gamma \beta}{\gamma_1, ..\gamma_n \vdash_\Gamma \gamma}$$

8.8 Tipo prodotto dipendente

Il tipo prodotto dipendente o indiciato (si chiama anche $\prod$), come già riscontrato per il tipo funzione, è un tipo non induttivo. Inoltre è un potenziamento espressivo, sempre, del **tipo dello spazio delle funzioni** (tipo funzioni è un caso particolare).
Le regole del prodotto indiciato dipendente, che agisce su un tipo dipendente, sono le seguenti.

8.8.1 Regole di Formazione

$$\text{F-}\prod)\ \frac{\mathrm{B\ type}\ [\Gamma] \qquad \mathrm{C(x)\ type}\ [\Gamma, \mathrm{x} \in \mathrm{B}]}{\prod_{x \in B} \mathrm{C(x)\ type}[\Gamma]}$$

8.8.2 Regole di Introduzione

$$\text{I-}\prod)\ \frac{\mathrm{c(x)} \in \mathrm{C(x)}[\Gamma, \mathrm{x} \in \mathrm{B}]}{\lambda \mathrm{x}^B.\mathrm{c(x)} \in \prod_{x \in B} \mathrm{C(x)}[\Gamma]}$$

8.8.3 Regole di Eliminazione

$$\text{E-}\prod)\ \frac{\mathrm{f} \in \prod_{x \in B} \mathrm{C(x)}[\Gamma] \qquad \mathrm{b} \in \mathrm{B}[\Gamma]}{\mathrm{Ap(f,b)} \in \mathrm{C(b)}[\Gamma]}$$

8.8.4 Regole di Conservazione

$$\text{C-}\prod)\ \frac{\mathrm{c(x)} \in \mathrm{C(x)}[\Gamma,\ \mathrm{x} \in \mathrm{B}] \qquad \mathrm{b} \in \mathrm{B}[\Gamma]}{\mathrm{Ap}(\lambda \mathrm{x}^B.\mathrm{c(x)},\mathrm{b}] = \mathrm{c(b)} \in \mathrm{C(b)}[\Gamma]}$$

8.8.5 Regole di Uguaglianza

$$\text{eq-E-}\prod)\ \frac{f_1 = f_2 \in \prod_{x\in B} C(x)[\Gamma] \qquad b_1 = b_2 \in B[\Gamma]}{Ap(f_1,b_1) = Ap(f_2,b_2) \in C(b_1)[\Gamma]}$$

$$\text{eq-I-}\prod)\ \frac{c_1(x) = c_2(x) \in C(x)[\Gamma,x \in B]}{\lambda x^B.c_1(x) = \lambda x^B.c_2(x) \in \prod_{x\in B} C(x)[\Gamma]}$$

La regola (eq-I-$\prod$) prende anche il nome di ξ-rule ed è una regola difficile da modellare.

$$\text{eq-F-}\prod)\ \frac{B_1 = B_2 \text{ type}[\Gamma] \qquad C_1(x) = C_2(x) \text{ type}[\Gamma,x \in B_1]}{\prod_{x\in B} C_1(x) = \prod_{x\in B} C_2(x)[\Gamma]}$$

8.8.6 Osservazioni dal punto di vista logico

La regola di Formazione

Se prendo la regola di *Formazione* e ci metto le proposizioni, questa diventa

$$\text{F-}\prod)\ \frac{B \text{ set } [\Gamma] \qquad \varphi(x) \text{ prop } [\Gamma,x \in B]}{\forall_{x\in B}\ \varphi(x) \text{ prop}[\Gamma]}$$

che rende vera la seguente definizione

$$\forall_{x\in B}\ \varphi(x) \stackrel{def}{=} \prod_{x\in B} \varphi(x)$$

dalle regole *(I-$\prod$)* e *(E-$\prod$)*.

La regola di Introduzione

$$\text{I-}\prod)\ \frac{c(x) \in \varphi(x)[\Gamma,x \in B]}{? \in \prod_{x\in B} \varphi(x)[\Gamma]}$$

che definisce nella *set-theory* la regola del $\forall$. Difatti, la regola, con le opportune sostituzioni diventa

$$\text{I-}\prod)\ \frac{\varphi(x) \text{ vero}[\Gamma,\Delta,x \in B]}{\forall_{x\in B}\ \varphi(x) \text{ vero}[\Gamma,\Delta]}$$

con $\Delta \equiv [x_1 ... \varphi_1 ... x_n ... \varphi_n]$ e φ_i prop[Γ]
Nel calcolo dei sequenti equivale al per ogni a destra

$$\forall\text{-D})\ \frac{\Delta \vdash_{\Gamma,x\in B} \varphi(x)}{\Delta \vdash_\Gamma \forall_{x\in B}\ \varphi(x)}\ x \notin fv(\Delta)$$

La regola di Eliminazione

$$\text{E-}\prod)\ \frac{\mathrm{f} \in \forall_{x \in B}\ \varphi(\mathrm{x})[\Gamma] \qquad \mathrm{b} \in \mathrm{B}[\Gamma]}{?\ \in \varphi(\mathrm{b})[\Gamma]}$$

che con le opportune sostituzioni diventa, nel calcolo dei sequenti, una regola di eliminazione della riduzione naturale

$$\text{E-}\prod)\ \frac{\Delta \vdash \forall_{x \in B}\ \varphi(x) \qquad \mathrm{b} \in \mathrm{B}[\Gamma]}{\Delta \vdash \varphi(\mathrm{b})}$$

con $\Delta \equiv$ tutte assunzioni di prop$[\Gamma]$

Per concludere, il prodotto dipendente definisce sia:

1. L'**implicazione**

   $$\beta \to \gamma \equiv \prod_{x \in B} \gamma$$

   dove $\to$ indica famiglia costante

2. La **quantificazione esistenziale**

   $$\forall_{x \in B}\ \varphi(x) \equiv \prod_{x \in B} \varphi(x)$$

Capitolo 9

Tipo vuoto

Il tipo vuoto (N_0, *empty-set*) è un tipo induttivo generato dalle sue regole di introduzione (che non ci sono).
Il tipo N_0 è definito dalle seguenti regole.

9.1 Regole di Formazione

$$\text{F-N}_0)\ \frac{\Gamma \text{ cont}}{N_0 \text{ type } [\Gamma]}$$

9.2 Regole di Introduzione

$N_0[\Gamma] = \varnothing \Rightarrow$ non ci sono valori canonici

Non ci sono le regole di Introduzione, in quanto N_0 si vuole che sia un tipo vuoto, e di conseguenza non serve introdurre elementi.

9.3 Regole di Eliminazione

$$\text{E-N}_0)\ \frac{t \in N_0[\Gamma] \qquad M(z) \text{ type}[\Gamma, z \in N_0]}{El_{N0}(t) \in M(t)[\Gamma]}$$

9.4 Regole di Conservazione

Non esiste alcuna regola di Conversione, a causa della mancanza di ipotesi dell'eliminatore sui termini M(0).

9.5 Regole di Uguaglianza

$$\text{eq-E-N}_0)\ \frac{t_1 = t_2 \in N_0[\Gamma] \qquad M(z) \text{ type}[\Gamma, z \in N_0]}{El_{N0}(t_1) = El_{N0}(t_2) \in M(t_1)[\Gamma]}$$

Non serve nessun'altra ipotesi, non essendoci alcun elemento canonico.
Inoltre questo comporta la non necessità delle altre regole di uguaglianza.

9.6 Eliminatore dipendente

$$\text{E-N}_{0dip})\ \frac{\text{M(z) type}[\Gamma,\text{z} \in \text{N}_0]}{\text{El}_{N0}(\text{z}) \in \text{M(z)}[\Gamma,\text{z} \in \text{N}_0]}$$

9.7 Semantica operazionale del tipo funzione

La relazione $\to_1$ viene definita all'interno dei termini con l'uso delle seguenti regole di riduzione:

- β_{N_0}-red) NON ESISTE
- per far interagire N_0 con altri tipi (altrimenti N_0 da solo non fa nulla) $\text{N}_0\text{-red})\ \dfrac{\text{t}_1 \to_1 \text{t}_2}{\text{El}_{N0}(\text{t}_1) \to_1 \text{El}_{N0}(\text{t}_2)}$

La regola *(N_0-red)* è associata all'uguaglianza dell'eliminazione.

9.8 Osservazioni dal punto di vista logico

Il tipo vuoto serve in logica per interpretare il falso

$$\bot \stackrel{def}{\equiv} \mathbf{N}_0$$

$\bot$ è vero sse $\exists$ t $\in \text{N}_0$

$$\text{E-N}_0\ \frac{\text{t} \in \text{N}_0[\Gamma] \qquad \text{ind-ty}\ \dfrac{\bot \text{ è vero}[\Gamma] \qquad \varphi \text{ prop}[\Gamma]}{\varphi \text{ prop}[\Gamma,\text{z} \in \text{N}_0]}}{\text{El}_{N0}(\text{t}) \in \varphi[\Gamma]}$$

Quella appena definita rappresenta la regola $\dfrac{\bot \text{ vero}[\Gamma]}{\varphi \text{ vero}[\Gamma]}$

Inoltre, con la regola di *Eliminazione dipendente*, definita in §9.6, posso dire che, per ogni φ prop[Γ]
$\Rightarrow$M(z) type[Γ,z $\in \text{N}_0$] $\equiv \varphi$(z) è prop
$\Rightarrow\text{El}_{N0}$(z) $\in$ M(z)[Γ,z $\in \text{N}_0$] $\equiv \varphi$ è vero[Γ,Δ,N_0 vero] $\equiv \bot \vdash \varphi$ che è l'**assioma del falso** in logica *($\bot$-ax)*
In conclusione, dunque, la regola di *Eliminazione* di N_0 rappresenta *($\bot$-ax)* in calcolo dei sequenti/regola del falso *quodlibet* in deduzione naturale.

Capitolo 10

La logica della teoria dei tipi di *Martin-Löf*

Molti tipi, come visto nei capitoli precedenti hanno un significato logico.

Ma qual'è la logica che viene affrontata in type-theory?

I tipi descritti fino a ora sono sufficienti per interpretare le formule, con predicati dipendenti dal tipo, della logica **predicativa con l'uguaglianza**. Tuttavia la logica che la teoria dei tipi, di questi tipi, rende valida è la **Logica intuizionistica** e non quella classica.

Quale è la logica valida in teoria dei tipi?

Dipende dalla teoria dei tipi (dalla teoria dei tipi di *Martin-Löf* si rende valida solo quella intuizionistica).
Di seguito do la definizione per induzione sulle formule. In queste il contesto tipa le variabili libere della formula.

$$(\bot)^I[\Gamma] \stackrel{def}{\equiv} \mathrm{N}_0 \text{ type}[\Gamma]$$
$$(tt)^I[\Gamma] \stackrel{def}{\equiv} \mathrm{N}_1 \text{ type}[\Gamma]$$
$$(\varphi \vee \psi)^I[\Gamma] \stackrel{def}{\equiv} \varphi^I + \psi^I \text{ type}[\Gamma]$$
$$(\varphi \;\&\; \psi)^I[\Gamma] \stackrel{def}{\equiv} \varphi^I \times \psi^I \text{ type}[\Gamma]$$
$$(\varphi \to \psi)^I[\Gamma] \stackrel{def}{\equiv} \textstyle\prod_{z\in\varphi^I} \psi^I \text{ type}[\Gamma]$$
$$(\neg\, \varphi)^I[\Gamma] \stackrel{def}{\equiv} \varphi^I \to \bot^I \text{type}[\Gamma] \stackrel{def}{\equiv} \varphi^I \to \mathrm{N}_0 \text{ type}[\Gamma]$$
$$(\forall_{x\in A}\; \varphi(\mathrm{x}))[\Gamma] \stackrel{def}{\equiv} \prod_{x\in A} (\varphi(\mathrm{x}))^I \text{ type}[\Gamma]$$
$$(\mathrm{t} = \mathrm{s})^I[\Gamma] \stackrel{def}{\equiv} \mathrm{Id(A,t,s)} \text{ type}[\Gamma] \Rightarrow \textit{unico predicato atomico}$$
$$(\exists_{x\in A}\; \varphi(\mathrm{x}))^I[\Gamma] \stackrel{def}{\equiv} \sum_{x\in A} (\varphi(\mathrm{x}))^I \text{ type}[\Gamma]$$

Da tenere presente che per noi vale

$$\boldsymbol{\varphi(\mathrm{x}) \text{ type}[\mathrm{x} \in \mathrm{A}]}$$

e in generale una formula φ(x) prop[x $\in$ A], con prop che dipendono da tipi e A tipi.
Invece sono formule non valide $\cancel{\forall \varphi(x)}$ $\cancel{\exists \varphi(x)}$, in quanto le variabili senza tipo non esistono in teoria dei tipi.

Lemma per ogni formula interpretata, il giudizio di tipo che la interpreta è derivabile nella teoria dei tipi.

Definizione

Con il giudizio α **prop[Γ]** (sintassi della logica del primo ordine), con α formula del linguaggio predicativo con l'uguaglianza a variabili tipate, si intende il giudizio α^I **type[Γ]** (in teoria dei tipi), ove questo è la traduzione di α nel linguaggio di una formula predicativa con l'uguaglianza.

Va ricordato che α prop[Γ] $\equiv$ predicato che è una formula proposizionale su Γ

α **prop** (secondo il solito uso di proposizione) **sse** α **prop[Γ]**

Definizione

Introduciamo il concetto di **dimostrazioni logiche**.
Date le proposizioni ϕ prop[Γ] e α_i prop[Γ] per i = 1,..,n derivabili in teoria dei tipi.
Introduciamo il **giudizio ϕ true[Γ,α_1 true,...,α_n true]**. In questo modo, si definisce il **contesto spuro**, dove Γ è il solito contesto e α_1 true,...,α_n true è un contesto nuovo di assunzioni di proposizioni.

Introdotto il giudizio diciamo che è derivabile se esiste un termine (detto *proof-term*) pf tale che pf $\in \phi[\Gamma$,x$_1 \in \alpha_1$,...,x$_n \in \alpha_n]$ che è derivabile in una teoria dei tipi.

Definizione
(Per ricollegare il calcolo del primo ordine con la logica del primo anno)

Preso un sequente, in formule della logica predicativa con l'uguaglianza e variabili, nel linguaggio L, tipate, vale che
$(\Sigma \vdash_\Gamma \Delta)^I \stackrel{def}{\equiv} (\Delta^\vee)^I$ **type** [Γ**,x** $\in (\Sigma^\&)^I$]
Per il quale vale una definizione per induzione, dei termini:

- $\Sigma^\&$
  - $[\ ]^{\&} \equiv tt$
  - $[\Sigma^I,\varphi\ ]^{\&} \equiv (\Sigma^\&)\ \&\ \varphi$
- $\Delta^\vee$
  - $[\ ]^{\vee} \equiv \bot$
  - $[\ \Delta^I,\varphi\ ]^{\vee} \equiv (\Delta^\vee) \vee \varphi$

Definizione

Un sequente $\sum \vdash_\Gamma \Delta$ è valido in teoria dei tipi sse $(\Delta^\vee)^I$ true $[\Gamma,(\sum^{\&})^I$ true$]$ è derivabile.
Ovvero se esiste un pf tale che pf $\in (\Delta^\vee)^I[\Gamma$,x $\in (\sum^{\&})^I]$.
Si può dimostrare $(\sum \vdash_\Gamma \Delta)^I$ è derivabile sse $(\Delta^\vee)^I$ type$[\Gamma]$ è derivabile.

Teorema i sequenti derivabili, del calcolo della deduzione naturale *DNI*,[1] sono validi in teoria dei tipi.

Non sono validi quelli della Logica classica. Perchè? Per il Principio del terzo escluso.

Il Principio del terzo escluso non è valido in teoria dei tipi perchè per ogni φ noi dovremmo dedurre, in teoria dei tipi, un **pf** $\in$ $(\varphi \vee \neg\varphi)^I$ $[\,] \equiv \varphi^I + \neg\varphi^I$ $[\,]$.

DNI + Principio del terzo escluso $\equiv$ calcolo dei sequenti della Logica classica prediativa con l'uguaglianza

$\Rightarrow$ Logica classica = Logica intuizionistica + la derivabilità $\vdash \varphi \vee \neg\varphi$

Il Principio del terzo escluso è la derivabilità $\vdash \varphi \vee \neg\varphi$, per la formula φ. Ma $\vdash \varphi \vee \neg\varphi$ non è valido in teoria dei tipi perchè esiste G, formula, per cui $\vdash$ G $\vee \neg$G non è valido.

Idea della Dimostrazione

Metto assieme il teorema di *Gödel* con quello della forma normale forte.

$$\text{pf} \in \text{G} \vee \neg\text{G} = \text{G}^I + \neg\text{G}^I$$

Questa va ridotta in forma normale, che corrisponde a una delle due forme seguenti:

- NF(pf) = inl(pf_1) con $\text{pf}_1 \in$ G[ ]
- NF(pf) = inr(pf_2) con $\text{pf}_2 \in$ G[ ]

Ma *Gödel* ha dimostrato che nè $\vdash$ G[ ] e neanche $\vdash \neg$G[ ] sono derivabili ($\Leftrightarrow$ non valido in teoria dei tipi.)

10.1 Esercizi

1) Supposto A type [Γ], B type[Γ] e R(x,y) type[$\Gamma, x \in A, y \in B$], dimostrare che il seguente principio di scelta, detto comunemente assioma di scelta

$$\forall_{x\in A}\ \exists_{y\in B} R(x,y) \rightarrow \exists_{f\in A\rightarrow B}\ \forall_{x\in A}\ R(x,f(x))\ \textbf{true}[\Gamma]$$

[1] Logica Intuizionistica.

Soluzione

Devo verificare che pf $\in \forall_{x\in A}\ \exists_{y\in B}$R(x,y) $\to \exists_{f\in A\to B}\ \forall_{x\in A}\ R(x,f(x))\ [\Gamma]$ sia derivabile in teoria dei tipi.
In logica predicativa con l'uguaglianza diventa:

$\Rightarrow$ pf $\in \prod_{z\in(\prod_{x\in A}\ \sum_{y\in B} R(x,y))} \sum_{f\in A\to B} \prod_{x\in A}\ R(x,f(x))\ [\Gamma]$

Ora posso applicare *(I-$\prod$)*:
l'istanza deve rispettare le regole di introduzione del tipo prodotto dipendente, somma indiciata e del tipo funzione, dunque

- prima applicazione, considerando $\prod_{z\in(\prod_{x\in A}\ \sum_{y\in B} R(x,y))}$ C(z) $[\Gamma]$: λz.c(z)
- seconda applicazione considerando $\prod_{z\in(\prod_{x\in A}\ \sum_{y\in B} R(x,y))} \sum_{f\in A\to B}$ C(f) $[\Gamma]$: λz.$<\pi_1 z, \pi_2 z>$.c(f), perchè per ogni z, z $\in$ f e z $\in$ C
- terza applicazione, considerando $\prod_{z\in(\prod_{x\in A}\ \sum_{y\in B} R(x,y))} \sum_{f\in A\to B}$ C(f) $[\Gamma]$ devo individuare un elemento che appartenga a B: λz.$<\pi_1(Ap(z,x))$, $\pi_2(Ap(z,x))>$.c(f)
- quarta applicazione, considerando $\prod_{z\in(\prod_{x\in A}\ \sum_{y\in B} R(x,y))} \to \sum_{f\in A\to B} \prod_{x\in A}\ R(x,f(x))\ [\Gamma]$: λz.$<\lambda\ x.\pi_1(Ap(z,x)),\ \lambda\ x.\pi_2(Ap(z,x))>$

$\theta = \Gamma$,z $\in \prod_{x\in A} \sum_{y\in B} R(x,y)$

$$\text{I-}\prod\ \dfrac{\text{I-}\sum\ \dfrac{\mathbf{1}\qquad \mathbf{2}\qquad \mathbf{3}}{<\lambda\ x.\pi_1(Ap(z,x)),\ \lambda\ x.\pi_2(Ap(z,x))> \in \sum_{f\in A\to B}\prod_{x\in A} R(x,f(x))\ [\theta]}}{\lambda z.<\lambda\ x.\pi_1(Ap(z,x)),\ \lambda\ x.\pi_2(Ap(z,x))> \in \prod_{z\in(\prod_{x\in A}\ \sum_{y\in B} R(x,y))}\sum_{f\in A\to B}\prod_{x\in A} R(x,f(x))\ [\Gamma]}$$

1

$$\text{I-}\!\to\ \dfrac{\text{PJ}_1\text{-}\times\ \dfrac{\text{E-}\prod\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{\text{A type}[\Gamma]}\quad \text{F-c}\ \dfrac{\text{F-}\prod\ \dfrac{\dfrac{}{\text{A type}[\Gamma]}\qquad \dfrac{}{\sum_{y\in B} R(x,y)\ \text{type}[\Gamma,\text{x} \in \text{A}]}}{\prod_{x\in A}\sum_{y\in B} R(x,y)\ \text{type}[\Gamma]}}{\theta\ \text{cont}}\ (\text{z} \in \prod_{x\in A}\sum_{y\in B} R(x,y)) \notin \Gamma}{\text{A type}[\theta]}}{\theta,\text{x} \in \text{A cont}}\ (\text{x} \in \text{A}) \notin \theta}{\text{z} \in \prod_{x\in A}\sum_{y\in B} R(x,y)[\theta,\text{x} \in \text{A}]}\qquad \text{var}\ \dfrac{\dfrac{(*)}{\theta,\text{x} \in \text{A cont}}}{x \in A[\theta,\text{x} \in \text{A}]}}{Ap(z,x) \in \sum_{y\in B} R(x,y)[\theta,\text{x} \in \text{A}]}}{\pi_1(Ap(z,x)) \in \text{B}[\theta,\text{x} \in \text{A}]}}{\lambda\ x.\pi_1(Ap(z,x)) \in \text{A} \to \text{B}[\theta]}$$

2

R(x,Ap($\lambda\ x.\pi_1(Ap(z,x))$,x)) $\to_1$ R(x, $\pi_1(Ap(z,x))$ con $\beta_{\to} - red$

$$
\text{I-}\prod\,\dfrac{\text{conv}\,\dfrac{\text{PJ}_2\text{-}\times\,\dfrac{\text{E-}\prod\,\dfrac{\dfrac{(*)}{z \in \prod_{x\in A} \sum_{y \in B} \mathrm{R(x,y)}[\theta, x \in A]} \qquad \dfrac{(*)}{\mathrm{x} \in \mathrm{A}\ [\theta, x \in A]}}{Ap(z,x) \in \sum_{y\in B} \mathrm{R(x,y)}[\theta, x \in A]}}{\pi_2(Ap(z,x)) \in \mathrm{R}(\mathrm{x},\ \pi_1(Ap(z,x))[\theta, x \in A]} \qquad \dfrac{}{\mathrm{R}(\mathrm{x},\ \pi_1(Ap(z,x)) = \mathrm{R}(\mathrm{x},\mathrm{Ap}(\lambda\ x.\pi_1(Ap(z,x)),\mathrm{x}))[\theta, x \in A]}}{\pi_2(Ap(z,x)) \in \mathrm{R}(\mathrm{x},\mathrm{Ap}(\lambda\ x.\pi_1(Ap(z,x)),\mathrm{x}))[\theta, x \in A]}}{\lambda\ x.\pi_2(Ap(z,x)) \in \prod_{x\in A} \mathrm{R}(\mathrm{x},\mathrm{Ap}(\lambda\ x.\pi_1(Ap(z,x)),\mathrm{x}))[\theta]}
$$

3

$$
\text{F-}\sum\,\dfrac{\text{F-}\prod\,\dfrac{\dfrac{}{\mathrm{A\ type}\ [\theta, f \in A \to B]} \qquad \dfrac{}{R(x, f(x))\ [\theta, f \in A \to B, x \in A]}}{\prod_{x\in A} R(x, f(x))\ [\theta, f \in A \to B]}}{\sum_{f \in A \to B} \prod_{x \in A} R(x, f(x))\ [\theta]}
$$

Ho usato $(*)$ per concludere le derivazioni già svolte all'interno dell'albero.

Capitolo 11

Principi di induzione per tipi induttivi

Di seguito tratto i principi di induzione, associati alle regole di *Eliminazione* dei cosiddetti tipi induttivi T, per renderne la ragione del nome; quando il tipo M(z) type[Γ,z $\in T$], verso cui si elimina, è in realtà una φ(z) prop[Γ,z $\in T$].

Tipi induttivi: N_0, N_1, List(A), Nat, B + C, $\sum_{x \in B}$ C(x), Id(A,a,b)

11.1 Proprietà per i tipi induttivi

Per poter enunciare il principio di induzione, ci servono delle **proprietà di manipolazione dei contesti**.

- $\sum$ serve per **raggruppare le assunzioni in un contesto**:
  c(x,y) $\in$ C(x,y)[x $\in$ A,y $\in$ B(x)] derivabile $\Rightarrow$ c(π_1z,π_2z) $\in$ C(π_1z,π_2z)[z $\in \sum_{x \in A}$B(x)] è derivabile
  Si può anche andare nella direzione opposta, dunque
  d(z) $\in$ D(z)[z $\in \sum_{x \in A}$B(x)] derivabile $\Rightarrow$ d($<$x,y$>$) $\in$ D($<$x,y$>$)[x $\in$ A,y $\in$ B(x)] derivabile

- $\prod$ serve per **togliere le assunzioni in un contesto**:
  c(x,y) $\in$ C(x,y)[x $\in$ A,y $\in$ B(x)] derivabile $\Rightarrow$ λy.c(x) $\in \prod_{y \in B(x)}$C(x,y)[x $\in$ A]
  Per cui posso sempre togliere l'ultima assunzione.

Notazione: quando ho φ(z) $\rightarrow \varphi^I$(z)

Di seguito enuncio i vari tipi di induzione, con spiegazione per il principio dei numeri Naturali.

11.2 Principio di induzione numeri Naturali

$$\text{E-Nat}_{dip})\ \frac{\mathrm{M(z)type}[\Gamma,\ z \in \mathrm{Nat}] \qquad \begin{array}{c} c \in \mathrm{M(0)}[\Gamma] \\ \mathrm{e(x,y)} \in \mathrm{M(succ(x))}[\Gamma, x \in \mathrm{Nat}, y \in \mathrm{M(x)}] \end{array}}{\mathrm{El}_{Nat}(\mathrm{z,c,e}) \in \mathrm{M(z)}[\Gamma, z \in \mathrm{Nat}]}$$

È da tale regola che si ricavano i principi di induzione.

Esempio

$$\text{E-Nat}_{dip})\ \frac{\varphi(\mathrm{z})\mathrm{prop}[\Gamma,\ z \in \mathrm{Nat}] \qquad \begin{array}{c} c \in \varphi(0)[\Gamma] \\ \mathrm{e(x,y)} \in \varphi(\mathrm{succ(x)})[\Gamma, z \in \mathrm{Nat}, y \in \mathrm{M(x)}] \end{array}}{\mathrm{El}_{Nat}(\mathrm{z,c,e}) \in \varphi(\mathrm{z})[\Gamma, z \in \mathrm{Nat}]}$$

$\Leftrightarrow \lambda z.\mathrm{El}_{Nat}(\mathrm{z,c,e}) \in \forall_{z \in Nat}\ \varphi(\mathrm{z})$ (usando $\prod$ per togliere le assunzioni di contesto)
A questo punto posso raggrupparlo in un unico termine, non solo con la manipolazione del contesto, ma anche trasformando le assunzioni della regola in variabili
$\lambda z.\mathrm{El}_{Nat}(\mathrm{z,w,(x,y)e(x,y)}) \in \forall_{z \in Nat}\varphi(z)[\Gamma, \mathrm{w} \in \varphi(0); \mathrm{z}_f \in \forall_{x \in Nat}\ (\varphi(\mathrm{x}) \rightarrow \varphi(\mathrm{succ(x)}))]$
$\Rightarrow \lambda z.\mathrm{El}_{Nat}(\mathrm{z,w,(x,y).Ap(Ap(z}_f\mathrm{,x),y)}) \in \forall_{z \in Nat}\varphi(z)[\Gamma, \mathrm{w} \in \varphi(0) \times/\&\ \mathrm{z}_f \in \forall_{x \in Nat}\ (\varphi(\mathrm{x}) \rightarrow \varphi(\mathrm{succ(x)}))]$

Assumendo che, quando astraggo, φ true$[\Gamma, \Psi$ true$]$ derivabile $\Leftrightarrow \Psi \rightarrow \varphi$ true$[\Gamma]$

In questo modo arrivo ad avere il principio di induzione che rende valido l'eliminatore dei numeri Naturali

$$\mathrm{Nat})\ \varphi(0)\ \&\ \forall_{x \in Nat}(\varphi(\mathrm{x}) \rightarrow \varphi(\mathrm{succ(x)}) \rightarrow \forall_{z \in Nat}\ \varphi(\mathrm{z})\ \mathrm{true}[\Gamma]$$

è valido e segue dalla *(E-Nat$_{dip}$)*

11.3 Principio di induzione su N_0

$$\mathrm{N0})\ \perp \mathrm{true} \rightarrow \varphi[\Gamma]$$

è valido per ogni φ prop$[\Gamma]$

11.4 Principio di induzione su N_1

$$\mathrm{N}_1)\ \varphi(*) \rightarrow \forall_{x \in N_1}\ \varphi(\mathrm{x})\ \mathrm{true}[\Gamma]$$

è valido per ogni $\varphi(z)$ prop$[\Gamma, z \in \mathrm{N}_1]$

11.5 Principio di induzione su List(A)

Supponiamo A type[Γ] derivabile

$$\text{List(A))}\ \varphi(nil)\ \&\ \forall_{x\in List(A)}\ \forall_{w\in A}(\varphi(x) \rightarrow \varphi(cons(x,w))) \rightarrow \forall_{z\in List(A)}\ \varphi(\text{z})\ \text{true}[\Gamma]$$

è derivabile

11.6 Principio di induzione su B+C

φ(z) prop[Γ, z ∈ B + C] derivabile

$$\text{B+C)}\ \forall_{x_1\in B}\ \varphi(inl(x_1))\ \&\ \forall_{x_2\in C}\ \varphi(inr(x_2)) \rightarrow \forall_{z\in B+C}\ \varphi(z)\ \text{true}[\Gamma]$$

è derivabile

11.7 Principio di induzione su $\sum_{x\in B}$ C(x)

Supponiamo φ(z) prop[Γ,z ∈ B + C]

$$\Sigma)\ \forall_{x\in B}\ \forall_{y\in C(x)}\ \varphi(<x,y>) \rightarrow \forall_{z\in\ \sum_{x\in B}C(x)}\ \varphi(\text{z})\ \text{true}[\Gamma]$$

è derivabile
con $\forall_{z\in\ \sum_{x\in B}C(x)}$ famiglia di insiemi = unione indiciata disgiunta

11.8 Principio di induzione su Id(A,a,b)

È un tipo induttivo, grande novità, introdotta da *Martin-Löf*, all'interno della teoria dei tipi.

Dato $\varphi(z_1, z_2, z_3)$ prop[Γ,z_1 ∈ A,z_2 ∈ A,z_3 ∈ Id(A,z_1,z_2)] derivabile

$$\text{Ind)}\ \forall_{x\in A}\ \varphi(x,x,id(x)) \rightarrow \forall_{z_1\in A}\ \forall_{z_2\in A}\ \forall_{z_3\in Id(A,z_1,z_2)}\ \varphi(z_1,z_2,z_3)\ \text{true}[\Gamma]$$

è derivabile

Tale principio non è formulato su φ(z) type[Γ,$z \in Id(A,a,b)$] con A e a,b fissati, ma dipende da tre parametri z_1, z_2 e z_3, come accade nella regola del *(E-Id)*.

In conclusione, ho mostrato, come i **principi di induzione** servono per dimostrare le proposizioni; e le **regola di eliminazione** non determinano solo ricorsori, ma anche regole di induzione.

11.9 Esercizi

1) Dimostrare che i principi di induzione sono validi in teoria dei tipi.

Soluzione

- N_0) $\perp$ true $\rightarrow \varphi[\Gamma]$

Devo verificare che pf $\in \perp \rightarrow \varphi[\Gamma] \equiv$ pf $\in \prod_{z\in\perp} \varphi[\Gamma] \equiv$ pf $\in \prod_{z\in N_0} \varphi[\Gamma]$ sia derivabile in teoria dei tipi.

pf $\in \prod_{z\in N_0} \varphi[\Gamma]$ applicando *(I-$\prod$)* diventa $\mathrm{El}_{N_0}(z) \in \varphi[\Gamma, z \in N_0]$

Supposto che φ type$[\Gamma]$ sia derivabile, allora

$$\text{E-N}_{0dip}\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{\varphi \text{ type}[\Gamma]} \qquad \text{F-c}\ \dfrac{\text{F-N}_0\ \dfrac{\Gamma \text{ cont}}{N_0 \text{ type}[\Gamma]}}{\Gamma, z \in N_0 \text{ cont}}\ (z \in N_0) \notin \Gamma}{\varphi \text{ type}[\Gamma, z \in N_0]}}{\mathrm{El}_{N_0}(z) \in \varphi[\Gamma, z \in N_0]}$$

- Nat) $\varphi(0)$ & $\forall_{x\in Nat}(\varphi(\mathrm{x}) \rightarrow \varphi(\mathrm{succ(x)}) \rightarrow \forall_{z\in Nat}\ \varphi(z)$ true$[\Gamma]$

Devo verificare che pf $\in \prod_{y\in(\varphi(0)\times\prod_{x\in Nat}(\prod_{s\in\varphi(x)} \varphi(succ(x))))} \prod_{z\in Nat}\varphi(z)[\Gamma]$
in teoria dei tipi; che equivale a
$\lambda z.\mathrm{El}_{Nat}(\mathrm{z,w,Ap(Ap(k,x),y)}) \in \prod_{z\in Nat}\varphi(z)[\Gamma,\ y \in (< w, k >\in \varphi(0) \times \prod_{x\in Nat}(\prod_{s\in\varphi(x)} \varphi(succ(x))))]$
Applico *(I-$\prod$)* e ottengo
$\mathrm{El}_{Nat}(\mathrm{z,w,Ap(Ap(k,x),y)}) \in \varphi(z)[\Gamma, y \in (< w, k >\in \varphi(0)\times\prod_{x\in Nat}(\prod_{s\in\varphi(x)} \varphi(succ(x)))),$
$z \in Nat]$

$\Delta \equiv \Gamma,\ y \in (< w, k >\in \varphi(0) \times \prod_{x\in Nat}(\prod_{s\in\varphi(x)} \varphi(succ(x))))$

Supposto che φ(z) type$[\Delta$, z $\in$ Nat$]$ sia derivabile, allora

$$\text{E-Nat}_{dip}\ \dfrac{\dfrac{}{\varphi(z) \text{ type}[\Delta, z \in Nat]} \qquad \dfrac{\mathbf{1}}{w \in \varphi(0)[\Delta]} \qquad \dfrac{\mathbf{2}}{\mathrm{Ap(Ap(k,x),y)} \in \varphi(succ(x))[\Delta,\ x \in Nat,\ y \in \varphi(x)]}}{\mathrm{El}_{Nat}(\mathrm{z,w,Ap(Ap(k,x),y)}) \in \varphi(z)[\Delta, z \in Nat]}$$

1

$$
\text{var}\;\dfrac{\text{F-c}\;\dfrac{\text{I-}\times\;\dfrac{\dfrac{}{w \in \varphi(0)\ \text{type}[\Gamma]}\qquad\dfrac{}{k \in \prod_{x\in Nat}(\prod_{s\in\varphi(x)} \varphi(succ(x)))\ \text{type}[\Gamma]}}{< w,k > \in \varphi(0)\times\prod_{x\in Nat}(\prod_{s\in\varphi(x)} \varphi(succ(x)))\ \text{type}[\Gamma]}}{\Delta\ \text{cont}}\;\begin{array}{l}(y \in (< w,k > \in \varphi(0)\times\prod_{x\in Nat}\\ (\prod_{s\in\varphi(x)} \varphi(succ(x))))) \notin \Gamma\end{array}}{\text{w} \in \varphi(0)[\Delta]}
$$

2

$$
\text{E-}\prod\;\dfrac{\text{E-}\prod\;\dfrac{\text{var}\;\dfrac{\text{F-c}\;\dfrac{\alpha\text{-eq}\;\dfrac{\dfrac{}{\varphi(z)\ \text{type}[\Delta,\text{z} \in \text{Nat}]}}{\varphi(x)\ \text{type}[\Delta,\ \text{x} \in \text{Nat}]}}{\Delta,\ \text{x} \in \text{Nat},\ \text{y} \in \varphi(x)\ \text{cont}}\;\begin{array}{l}(\text{y} \in \varphi(x)) \notin\\ (\Delta,\ \text{x} \in \text{Nat})\end{array}}{\text{k} \in \prod_{x\in Nat}\prod_{s\in\varphi(x)} \varphi(\text{succ(x)})\ [\Delta,\ \text{x} \in \text{Nat},\ \text{y} \in \varphi(x)]}\qquad \text{var}\;\dfrac{\dfrac{(*)}{\Delta,\ \text{x} \in \text{Nat},\ \text{y} \in \varphi(x)\ \text{cont}}}{\text{x} \in \text{Nat}\ [\Delta,\ \text{x} \in \text{Nat},\ \text{y} \in \varphi(x)]}}{\text{Ap(k,x)} \in \prod_{s\in\varphi(x)} \varphi(\text{succ(x)})\ [\Delta,\ \text{x} \in \text{Nat},\ \text{y} \in \varphi(x)]}\qquad \text{var}\;\dfrac{\dfrac{(*)}{\Delta,\ \text{x} \in \text{Nat},\ \text{y} \in \varphi(x)\ \text{cont}}}{\text{y} \in \varphi(x)[\Delta,\ \text{x} \in \text{Nat},\ \text{y} \in \varphi(x)]}}{\text{Ap(Ap(k,x),y)} \in \varphi(succ(x))[\Delta,\ \text{x} \in \text{Nat},\ \text{y} \in \varphi(x)]}
$$

Ho usato $(*)$ per concludere le derivazioni già svolte all'interno dell'albero.
La validità di α-eq l'ho già dimostrata in §3.10, esercizio 5.

- N_1) $\varphi(*) \to \forall_{x\in N_1}\ \varphi(\text{x})$ true$[\Gamma]$

Devo verificare che pf $\in \prod_{z\in\varphi(*)} \forall_{x\in N_1}\ \varphi(\text{x})[\Gamma] \equiv$ pf $\in \prod_{z\in\varphi(*)} \prod_{x\in N_1}$ $\varphi(\text{x})[\Gamma]$ in teoria dei tipi; che equivale a
λx.El$_{N1}$(x,z) $\in \prod_{x\in N_1}\ \varphi(\text{x})[\Gamma,\ \text{z} \in \varphi(*)]$
Applico *(I-$\prod$)* e ottengo
El$_{N1}$(x,z) $\in \varphi(\text{x})[\Gamma,\ \text{z} \in \varphi(*),\ \text{x} \in \text{N}_1]$

Supposto che $*$ appartenga a $\varphi(*)$ per costruzione e $\varphi(\text{z})$ type$[\Delta,\ \text{z} \in \text{N}_1]$ sia derivabile, allora

$$
\text{E-S}_{dip}\;\dfrac{\alpha - eq\;\dfrac{\dfrac{}{\varphi(\text{z})[\Gamma,\ \text{x} \in \varphi(*),\ \text{z} \in \text{N}_1]}}{\varphi(\text{x})[\Gamma,\ \text{z} \in \varphi(*),\ \text{x} \in \text{N}_1]}\qquad \text{var}\;\dfrac{\text{F-c}\;\dfrac{\text{s-checks}\;\dfrac{\dfrac{}{* \in \varphi(*)\ \text{type}[\Gamma]}}{\varphi(*)\ \text{type}[\Gamma]}}{\Gamma,\ \text{z} \in \varphi(*)\ \text{cont}}\;(\text{z} \in \varphi(*)) \notin \Gamma}{\text{z} \in \varphi(*)[\Gamma,\ \text{z} \in \varphi(*)]}}{\text{El}_{N1}(\text{x,z}) \in \varphi(\text{x})[\Gamma,\ \text{z} \in \varphi(*),\ \text{x} \in \text{N}_1]}
$$

La validità di α-eq l'ho già dimostrata in §3.10, esercizio 5.

- List(A)) $\varphi(nil)\ \&\ \forall_{x\in List(A)}\ \forall_{w\in A}(\varphi(x) \to \varphi(cons(x,w))) \to \forall_{z\in List(A)}$ $\varphi(z)$ true$[\Gamma]$

Devo verificare che pf $\in \varphi(nil) \times \prod_{x\in List(A)} \prod_{w\in A} (\prod_{z'\in\varphi(x)} \varphi(cons(x,w)))$ $\to \prod_{z\in List(A)} \varphi(z)\ [\Gamma]$
pf $\in \prod_{z''\in(\varphi(nil)\times\prod_{x\in List(A)}\prod_{w\in A}(\prod_{z'\in\varphi(x)}\varphi(cons(x,w))))} \prod_{z\in List(A)} \varphi(z)\ [\Gamma]$ in teoria dei tipi; che equivale a
$\lambda z.$El$_{List}$(z,c,(Ap(Ap(Ap(k,x),w),y)) $\in \prod_{z\in List(A)} \varphi(z)\ [\Gamma,\ \text{z}'' \in (< c,k > \in$ $\varphi(nil) \times \prod_{x\in List(A)}\prod_{w\in A} (\prod_{z'\in\varphi(x)}\varphi(cons(x,w))))]$
Applico *(I- $\prod$)* e ottengo

El_{List}(z,c,Ap(Ap(Ap(k,x),w),y)) $\in \varphi(z)$ [Γ, z'' $\in$ ($< c,k >\in \varphi(nil) \times \prod_{x\in List(A)}\prod_{w\in A} (\prod_{z'\in\varphi(x)}\varphi(cons(x,w))))$, z $\in$ List(A)]

$\Delta \equiv \Gamma$, z'' $\in (< c,k >\in \varphi(nil)\times\prod_{x\in List(A)}\prod_{w\in A} (\prod_{z'\in\varphi(x)}\varphi(cons(x,w))))$

Supposto che φ(z) type[Δ, z $\in$ List(A)] e A type[Γ] siano derivabili, allora

$$\text{E-List}_{dip}\ \dfrac{\dfrac{}{\varphi(z)\ [\Delta, z \in \mathrm{List(A)}]} \qquad \dfrac{\mathbf{1}}{c \in \varphi(nil)[\Delta]} \qquad \dfrac{\mathbf{2}}{\mathrm{Ap(Ap(Ap(k,x),w),y))} \in \varphi(cons(x,w))[\Delta,\ x \in \mathrm{List(A)},\ w \in A,\ y \in \varphi(x)]}}{\mathrm{El}_{List}(z,c,\mathrm{Ap(Ap(Ap(k,x),w),y)} \in \varphi(z)\ [\Delta, z \in \mathrm{List(A)}]}$$

1

$$\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{I-}\times\ \dfrac{\dfrac{}{c \in \varphi(nil)\ \mathrm{type}[\Gamma]} \qquad \dfrac{}{k \in \prod_{x\in List(A)}\prod_{w\in A} (\prod_{z'\in\varphi(x)}\varphi(cons(x,w)))\ \mathrm{type}[\Gamma]}}{< c,k >\in \varphi(nil) \times \prod_{x\in List(A)}\prod_{w\in A} (\prod_{z'\in\varphi(x)}\varphi(cons(x,w)))\ \mathrm{type}[\Gamma]} \quad (z'' \in (< c,k >\in \varphi(nil) \times \prod_{x\in List(A)} \prod_{w\in A} (\prod_{z'\in\varphi(x)}\varphi(cons(x,w)))) \notin \Gamma}{\Delta\ \mathrm{cont}}}{c \in \varphi(nil)[\Delta]}$$

2

$\zeta \equiv \Delta$, x $\in$ List(A), w $\in$ A, y $\in \varphi(x)$

$$\text{E-}\prod\ \dfrac{\text{E-}\prod\ \dfrac{\dfrac{\mathbf{2'}}{\mathrm{Ap(k,x)} \in \prod_{w\in A} \prod_{z'\in\varphi(x)} \varphi(cons(x,w))[\zeta]} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\zeta\ \mathrm{cont}}}{w \in A\ [\zeta]}}{\mathrm{Ap(Ap(k,x),w)} \in \prod_{z'\in\varphi(x)} \varphi(cons(x,w))[\zeta]} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\zeta\ \mathrm{cont}}}{y \in \varphi(x)\ [\zeta]}}{\mathrm{Ap(Ap(Ap(k,x),w),y)} \in \varphi(cons(x,w))[\zeta]}$$

2'

$$\text{E-}\prod\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{ind-ty}\ \dfrac{\alpha-eq\ \dfrac{\dfrac{}{\varphi(z)\ \mathrm{type}[\Delta,\ z \in \mathrm{List(A)}]}}{\varphi(x)\ \mathrm{type}[\Delta,\ x \in \mathrm{List(A)}]} \qquad \text{F-c}\ \dfrac{\dfrac{}{A\ \mathrm{type}[\Delta,\ x \in \mathrm{List(A)}]}}{\Delta,\ x \in \mathrm{List(A)},\ w \in A\ \mathrm{cont}}\ (w \in A) \notin (\Delta,\ x \in \mathrm{List(A)})}{\varphi(x)\ \mathrm{type}[\Delta,\ x \in \mathrm{List(A)},\ w \in A]} \quad (y \in \varphi(x)) \notin (\Delta,\ x \in \mathrm{List(A)},\ w \in A)}{\zeta\ \mathrm{cont}}}{k \in \prod_{x\in List(A)} \prod_{w\in A} \prod_{z'\in\varphi(x)} \varphi(cons(x,w))[\zeta]} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\zeta\ \mathrm{cont}}}{x \in \mathrm{List(A)}[\zeta]}}{\mathrm{Ap(k,x)} \in \prod_{w\in A} \prod_{z'\in\varphi(x)} \varphi(cons(x,w))[\zeta]}$$

Ho usato ($*$) per concludere le derivazioni già svolte all'interno dell'albero.

- B+C) $\forall_{x_1\in B}\ \varphi(inl(x_1))\ \&\ \forall_{x_2\in C}\ \varphi(inr(x_2)) \to \forall_{z\in B+C}\ \varphi(z)$ true[Γ]

Devo verificare che pf $\in \prod_{x_1\in B} \varphi(inl(x_1)) \times \prod_{x_2\in C} \varphi(inr(x_2)) \to \prod_{z\in B+C} \varphi(z)$ [Γ] $\equiv$ pf $\in \prod_{z'\in(\prod_{x_1\in B}\varphi(inl(x_1))\times\prod_{x_2\in C}\varphi(inr(x_2)))} \prod_{z\in B+C} \varphi(z)$ [Γ] in teoria dei tipi; che equivale a
λz.El_+(z,Ap(k,x_1),Ap(w,x_2)) $\in \prod_{z\in B+C} \varphi(z)$ [$\Gamma, z' \in (< k,w >\in \prod_{x_1\in B} \varphi(inl(x_1))\times \prod_{x_2\in C} \varphi(inr(x_2)))$]
Applico *(I-$\prod$)* e ottengo
El_+(z,Ap(k,x_1),Ap(w,x_2)) $\in \varphi(z)$ [$\Gamma, z' \in (< k,w >\in \prod_{x_1\in B} \varphi(inl(x_1)) \times \prod_{x_2\in C} \varphi(inr(x_2))),\ z \in B + C$]

$\Delta \equiv \Gamma, z` \in (< k, w >\in \prod_{x_1 \in B} \varphi(inl(x_1)) \times \prod_{x_2 \in C} \varphi(inr(x_2)))$

Supposto che φ(z) type[Δ, z $\in$ B + C], B type[Δ] e C type[Δ] siano derivabili, allora

$$\text{E-+}\ \frac{\dfrac{}{\varphi(z)\ \text{type}[\Delta, z \in B + C]} \qquad \dfrac{\mathbf{1}}{\text{Ap(k,x}_1) \in \varphi(inl(x_1))\ [\Delta, x_1 \in B]} \qquad \dfrac{\mathbf{2}}{\text{Ap(w,x}_2) \in \varphi(inr(x_2))\ [\Delta, x_2 \in C]}}{\text{El}_+\text{(z,Ap(k,x}_1\text{),Ap(w,x}_2)) \in \varphi(z)\ [\Delta, z \in B + C]}$$

1

$$\text{E-}\prod\ \frac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\overline{\text{B type}[\Delta]}}{\Delta, x_1 \in B \text{ cont}}\ (x_1 \in B) \notin \Delta}{\text{k} \in \prod_{x_1 \in B} \varphi(inl(x_1))\ [\Delta, x_1 \in B]} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\Delta, x_1 \in B \text{ cont}}}{\text{x}_1 \in \text{B}\ [\Delta, x_1 \in B]}}{\text{Ap(k,x}_1) \in \varphi(inl(x_1))\ [\Delta, x_1 \in B]}$$

2

$$\text{E-}\prod\ \frac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\overline{\text{C type}[\Delta]}}{\Delta, x_2 \in C \text{ cont}}\ (x_2 \in C) \notin \Delta}{\text{w} \in \prod_{x_2 \in C} \varphi(inr(x_2))\ [\Delta, x_2 \in C]} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\Delta, x_2 \in C \text{ cont}}}{\text{x}_2 \in \text{C}\ [\Delta, x_2 \in C]}}{\text{Ap(w,x}_2) \in \varphi(inr(x_2))\ [\Delta, x_2 \in C]}$$

Ho usato $(*)$ per concludere le derivazioni già svolte all'interno dell'albero.

- $\sum$) $\forall_{x \in B}\ \forall_{y \in C(x)}\ \varphi(< x, y >) \rightarrow \forall_{z \in\ \sum_{x \in B} C(x)} \varphi$(z) true[$\Gamma$]

Devo verificare che pf $\in \prod_{x \in B} \prod_{y \in C(x)} \varphi(< x, y >) \rightarrow \prod_{z \in \sum_{x \in B} C(x)} \varphi(z)$ [Γ] $\equiv$ pf $\in \prod_{z` \in (\prod_{x \in B} \prod_{y \in C(x)} \varphi(<x,y>))} \prod_{z \in \sum_{x \in B} C(x)} \varphi(z)$ [Γ] che in teoria dei tipi; equivale a
$\lambda z.El_{\Sigma}(z, Ap(Ap(z`, x), y)) \in \prod_{z \in \sum_{x \in B} C(x)} \varphi(z)[\Gamma$, z ` $\in (\prod_{x \in B} \prod_{y \in C(x)} \varphi(< x, y >))$
Applico *(I-$\prod$)* e ottengo
$El_{\Sigma}(z, Ap(Ap(z`, x), y)) \in \varphi(z)[\Gamma$, z ` $\in (\prod_{x \in B} \prod_{y \in C(x)} \varphi(< x, y >))$, z $\in \sum_{x \in B} C(x)]$

$\Delta \equiv \Gamma$, z ` $\in (\prod_{x \in B} \prod_{y \in C(x)} \varphi(< x, y >))$

Supposto che φ(z) type[Δ, z $\in \sum_{x \in B} C(x)$] e C(x) type[Δ, x $\in$ B] siano derivabili, allora

$$\text{E-}\Sigma\ \dfrac{\dfrac{}{\varphi(z)[\Delta,\ z \in \sum_{x\in B} C(x)]} \qquad \text{E-}\prod\ \dfrac{\dfrac{\mathbf{1}}{\mathrm{Ap(z`,x)} \in \prod_{y\in C(x)} \varphi(<x,y>)[\Delta,\ x \in B,\ y \in C(x)]} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\Delta,\ x \in B,\ y \in C(x)\ \mathrm{cont}}}{y \in C(x)[\Delta,\ x \in B,\ y \in C(x)]}}{\mathrm{Ap(Ap(z`,x),y)} \in \varphi(<x,y>)[\Delta,\ x \in B,\ y \in C(x)]}}{El_{\Sigma}(z, Ap(Ap(z`,x),y)) \in \varphi(z)[\Delta,\ z \in \sum_{x\in B} C(x)]}$$

1

$$\text{E-}\prod\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\dfrac{}{C(x)\ \mathrm{type}[\Delta,\ x \in B]}}{\Delta,\ x \in B,\ y \in C(x)\ \mathrm{cont}}\ (y \in C(x)) \notin (\Delta,\ x \in B)}{z` \in \prod_{x\in B} \prod_{y\in C(x)} \varphi(<x,y>)[\Delta,\ x \in B,\ y \in C(x)]} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\Delta,\ x \in B,\ y \in C(x)\ \mathrm{cont}}}{x \in B\ [\Delta,\ x \in B,\ y \in C(x)]}}{\mathrm{Ap(z`,x)} \in \prod_{y\in C(x)} \varphi(<x,y>)[\Delta,\ x \in B,\ y \in C(x)]}$$

Ho usato $(*)$ per concludere le derivazioni già svolte all'interno dell'albero.

- Ind) $\forall_{x\in A}\ \varphi(x,x,id(x)) \rightarrow \forall_{z_1\in A}\ \forall_{z_2\in A}\ \forall_{z_3\in Id(A,z_1,z_2)}\ \varphi(z_1,z_2,z_3)$ true$[\Gamma]$

Devo verificare che pf $\in \prod_{x\in A} \varphi(x,x,id(x)) \rightarrow \prod_{z_1\in A} \prod_{z_2\in A} \prod_{z_3\in Id(A,z_1,z_2)} \varphi(z_1,z_2,z_3)$ $[\Gamma]$ $\equiv$ pf $\in \prod_{z\in(\prod_{x\in A}\varphi(x,x,id(x)))} \prod_{z_1\in A} \prod_{z_2\in A} \prod_{z_3\in Id(A,z_1,z_2)} \varphi(z_1,z_2,z_3)$ $[\Gamma]$ in teoria dei tipi; che equivale a
$\lambda z_1.\lambda z_2.\lambda z_3.El_{Id}(z_3, Ap(z,x)) \in \prod_{z_1\in A} \prod_{z_2\in A} \prod_{z_3\in Id(A,z_1,z_2)} \varphi(z_1,z_2,z_3)$ $[\Gamma,\ z \in (\prod_{x\in A}\varphi(x,x,id(x)))]$
Applico *(I-$\prod$)* e ottengo
$El_{Id}(z_3, Ap(z,x)) \in \varphi(z_1,z_2,z_3)$ $[\Gamma,\ z \in (\prod_{x\in A}\varphi(x,x,id(x))),\ z_1 \in A,\ z_2 \in A,\ z_3 \in Id(A,z_1,z_2)]$

$\Delta \equiv \Gamma,\ z \in (\prod_{x\in A}\varphi(x,x,id(x)))$

Supposto che $\varphi(z_1,z_2,z_3)$ $[\Delta,\ z_1 \in A,\ z_2 \in A,\ z_3 \in Id(A,z_1,z_2)]$ e A type$[\Delta]$ siano derivabili, allora

$$\text{E-Id}\ \dfrac{\dfrac{}{\varphi(z_1,z_2,z_3)\ [\Delta,\ z_1 \in A,\ z_2 \in A,\ z_3 \in Id(A,z_1,z_2)]} \qquad \text{E-}\prod\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\dfrac{}{A\ \mathrm{type}[\Delta]}}{\Delta,\ x \in A\ \mathrm{cont}}\ (x \in A) \notin \Delta}{z \in \prod_{x\in A} \varphi(x,x,id(x))\ [\Delta,\ x \in A]} \qquad \text{var}\ \dfrac{\dfrac{(*)}{\Delta,\ x \in A\ \mathrm{cont}}}{x \in A[\Delta,\ x \in A]}}{\mathrm{Ap(z,x)} \in \varphi(x,x,id(x))\ [\Delta,\ x \in A]}}{El_{Id}(z_3, Ap(z,x)) \in \varphi(z_1,z_2,z_3)\ [\Delta,\ z_1 \in A,\ z_2 \in A,\ z_3 \in Id(A,z_1,z_2)]}$$

Ho usato $(*)$ per concludere le derivazioni già svolte all'interno dell'albero.

2) Si dimostri che esiste un *proof-term* del tipo

pf $\in$ Id(A,x,z)[x $\in$ A,y $\in$ A,z $\in$ A,w$_1$ $\in$ Id(A,x,y),w$_2$ $\in$ Id(A,y,z)]

Soluzione

Utilizzo $\prod$ per togliere le assunzioni in un contesto (§11.1):

$\Rightarrow \lambda w_2.id(w_2) \in \prod_{w_2\in Id(A,y,z)}$ Id(A,x,z)[x $\in$ A,y $\in$ A,z $\in$ A,w_1 $\in$ Id(A,x,y)]
$\Rightarrow \lambda z.\lambda w_2.id(w_2) \in \prod_{z\in A}\prod_{w_2\in Id(A,y,z)}$ Id(A,x,z)[x $\in$ A,y $\in$ A,w_1 $\in$ Id(A,x,y)]
$\Rightarrow \lambda w_1.\lambda z.\lambda w_2.id(w_2) \in \prod_{w_1\in Id(A,x,y)}\prod_{z\in A}\prod_{w_2\in Id(A,y,z)}$ $\in$ Id(A,x,z)[x $\in$ A,y $\in$ A]
$\Rightarrow \lambda w_1.(El_{Id}(w_1,(x).\lambda z.\lambda w_2.w_2)) \in \prod_{w_1\in Id(A,x,y)}\prod_{z\in A}\prod_{w_2\in Id(A,y,z)}$ $\in$ Id(A,x,z)[x $\in$ A,y $\in$ A]
Applico *(I-$\prod$)*, per poter poi derivare con *(*E-Id$_{dip}$), e ottengo
El$_{Id}$(w_1, (x).$\lambda z.\lambda w_2.w_2$) $\in \prod_{z\in A}\prod_{w_2\in Id(A,y,z)}$ Id(A,x,z)[x $\in$ A,y $\in$ A,w_1 $\in$ Id(A,x,y)]

- Premesse:
  - M(z_1,z_2,z_3) $\equiv$ M(x,y,w_1)
  - M(z_1,z_2,z_3) type[$\Gamma, z_1\in A, z_2\in A, z_3\in Id(A,z_1,z_2)$] $\equiv \prod_{z\in A}\prod_{w_2\in Id(A,y,z)}$ Id(A,x,z)[x $\in$ A,y $\in$ A,w_1 $\in$ Id(A,x,y)]
  - e(x) $\in$ M(x,x,id(x))[$\Gamma, x\in A$] $\equiv$ $\lambda z.\lambda w_2.w_2 \in \prod_{z\in A}\prod_{w_2\in Id(A,x,z)}$ Id(A,x,z)[x $\in$ A]
- Conclusione:
  - El$_{Id}$(z_3, (x).e(x)) $\equiv$ El$_{Id}$(w_1, (x).$\lambda z.\lambda w_2.w_2$)
  - M(z_1,z_2,z_3) type[Γ, z_1 $\in$ A, z_2 $\in$ A, z_3 $\in$ Id(A,z_1,z_2)] $\equiv \prod_{z\in A}\prod_{w_2\in Id(A,y,z)}$ Id(A,x,z)[x $\in$ A,y $\in$ A,w_1 $\in$ Id(A,x,y)]

$$\text{E-Id}_{dip}\ \dfrac{\dfrac{\mathbf{1}}{\prod_{z\in A}\prod_{w_2\in Id(A,y,z)} \mathrm{Id(A,x,z)[x \in A,y \in A,w_1 \in Id(A,x,y)]}} \qquad \dfrac{\mathbf{2}}{\lambda z.\lambda w_2.w_2 \in \prod_{z\in A}\prod_{w_2\in Id(A,x,z)} \mathrm{Id(A,x,z)[x \in A]}}}{\mathrm{El}_{Id}(\mathrm{w_1,\ (x)}.\lambda z.\lambda w_2.w_2) \in \prod_{z\in A}\prod_{w_2\in Id(A,y,z)} \mathrm{Id(A,x,z)[x \in A,y \in A,w_1 \in Id(A,x,y)]}}$$

1

$\Delta \equiv$ x $\in$ A,y $\in$ A,w_1 $\in$ Id(A,x,y), z $\in$ A

$$\text{F-}\prod\ \dfrac{\text{ind-ty}\ \dfrac{\dfrac{}{\mathrm{A\ type[\ ]}} \quad \text{F-c}\ \dfrac{\text{F-Id}\ \dfrac{\dfrac{}{\mathrm{A\ type[x\in A,y\in A]}} \quad \dfrac{}{\mathrm{x\in A[x\in A,y\in A]}} \quad \dfrac{}{\mathrm{y\in A[x\in A,y\in A]}}}{\mathrm{Id(A,x,y)\ type[x\in A,y\in A]}} \quad \begin{array}{c}(\mathrm{w_1\in Id(A,x,y)})\\ \notin (\mathrm{x\in A,y\in A})\end{array}}{\mathrm{x\in A,y\in A,w_1\in Id(A,x,y)\ cont}}}{\mathrm{A\ type[x\in A,y\in A,w_1\in Id(A,x,y)]}} \qquad \dfrac{\mathbf{1`}}{\prod_{w_2\in Id(A,y,z)} \mathrm{Id(A,x,z)}[\Delta]}}{\prod_{z\in A}\prod_{w_2\in Id(A,y,z)} \mathrm{Id(A,x,z)[x\in A,y\in A,w_1\in Id(A,x,y)]}}$$

1`

$$\text{F-}\prod\ \dfrac{\text{F-Id}\ \dfrac{\dfrac{}{\mathrm{A\ type}[\Delta]} \quad \dfrac{}{\mathrm{y\in A}[\Delta]} \quad \dfrac{}{\mathrm{z\in A}[\Delta]}}{\mathrm{Id(A,y,z)\ type}[\Delta]} \qquad \text{F-Id}\ \dfrac{\dfrac{}{\mathrm{A\ type}[\Delta,\ \mathrm{w_2\in Id(A,y,z)}]} \quad \dfrac{}{\mathrm{x\in A}[\Delta,\ \mathrm{w_2\in Id(A,y,z)}]} \quad \dfrac{}{\mathrm{z\in A}[\Delta,\ \mathrm{w_2\in Id(A,y,z)}]}}{\mathrm{Id(A,x,z)}[\Delta,\ \mathrm{w_2\in Id(A,y,z)}]}}{\prod_{w_2\in Id(A,y,z)} \mathrm{Id(A,x,z)}[\Delta]}$$

2

$$\text{I-}\prod\ \dfrac{\text{I-}\prod\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Id}\ \dfrac{\dfrac{}{\mathrm{A\ type[x\in A,\ z\in A]}} \quad \dfrac{}{\mathrm{x\in A\ [x\in A,\ z\in A]}} \quad \dfrac{}{\mathrm{z\in A\ [x\in A,\ z\in A]}}}{\mathrm{Id(A,x,z)[x\in A,\ z\in A]}} \quad \begin{array}{c}(\mathrm{w_2\in Id(A,x,z)})\notin\\ (\mathrm{x\in A,\ z\in A})\end{array}}{\mathrm{x\in A,\ z\in A,\ w_2\in Id(A,x,z)\ cont}}}{\mathrm{w_2\in Id(A,x,z)[x\in A,\ z\in A,\ w_2\in Id(A,x,z)]}}}{\lambda w_2.w_2 \in \prod_{w_2\in Id(A,x,z)} \mathrm{Id(A,x,z)[x\in A,\ z\in A]}}}{\lambda z.\lambda w_2.w_2 \in \prod_{z\in A}\prod_{w_2\in Id(A,x,z)} \mathrm{Id(A,x,z)[x\in A]}}$$

Capitolo 12

Tipi Universo à la Tarski

Il tipo Universo, **schema di tipi**, non è induttivo ed è rivolto, in questo caso, alla teoria dei tipi di *Martin-Löf* intensionale.
Esiste il tipo Universo à la *Tarski* e alla *Russell.* Il primo, che espongo nel capitolo, più preciso ma pesante nella sintassi, più facile da capire e da modellare (soprattutto per fare delle semantiche nella teoria dei tipi), per cui migliore quando si programma con un *proof-assistant*; e il secondo più semplice da utilizzare nelle dimostrazioni, ma meno preciso.

Caratteristica del tipo Universo è che permette la ripetizione delle costruzioni per generare una gerarchia di universi, legati fra di loro

$$\mathrm{U}_0,\ \mathrm{U}_1, \ldots,\ \mathrm{U}_n \qquad \text{con n} \in \mathrm{Nat}$$

Nella prima versione *Martin-Löf* permetteva che U_0 appartenesse a U_0, generando però una contraddizione.

$$\mathbf{U}_0 \notin \mathbf{U}_0 \quad \rightarrow \quad \mathbf{U}_0 \in \mathbf{U}_0 \ \textbf{è una contraddizione}$$

Universo è per questo un contenitore, di tipi, predicativo, ovvero $\mathrm{U}_0 \notin \mathrm{U}_0$.

Fondamentale

Essenziale, per avere commutatività, è non permettere che $\hat{U_0} \in \mathrm{U}_0$ (un codice U_0 dentro a U_0) altrimenti, come già detto, si ricade nella contraddizione. Per cui si dichiara come "*small*" o piccoli tutti i tipi con codice in U_0; e invece come "*large*" i tipi costruiti con U_0, ad esempio $\mathrm{U}_0 \rightarrow \mathrm{U}_0$ type[ ].
Ecco che per la natura della *regola di Eliminazione* e dal fatto che contiene esclusivamente *small* e non se stesso, posso affermare che l'Universo è **predicativo e non induttivo**.

Il tipo del **Primo Universo ($\mathbf{U}_0$) alla Tarski** è definito dalle regole seguenti, dove **T** indica la decodifica.

12.1 Regole di Formazione

$$\text{F-Uno)}\ \frac{\Gamma \ \text{cont}}{\mathrm{U}_0 \ \text{type}[\Gamma]}$$

U_0 (come N_0, N_1, ecc..) è una costante di tipo.

12.2 Regole di Introduzione

Per le regole di Introduzione U_0 funge da contenitore di tipi, tramite codifica, ed è chiuso rispetto ai tipi introdotti, nei capitolo precedenti. Inoltre le regole di Introduzione usano quelle di Eliminazione.

$$\mathrm{I_1\text{-}Uno)}\ \frac{\Gamma\ \mathrm{cont}}{\hat{N_0} \in \mathrm{U_0}[\Gamma]} \qquad \mathrm{I_2\text{-}Uno)}\ \frac{\Gamma\ \mathrm{cont}}{\hat{N_1} \in \mathrm{U_0}[\Gamma]} \qquad \mathrm{I_3\text{-}Uno)}\ \frac{\Gamma\ \mathrm{cont}}{\hat{Nat} \in \mathrm{U_0}[\Gamma]}$$

$$\mathrm{I_4\text{-}Uno)}\ \frac{\mathrm{c(x)} \in \mathrm{U_0}[\Gamma,\mathrm{x} \in \mathbf{T}(\mathrm{b})] \qquad \mathrm{b} \in \mathrm{U_0}[\Gamma]}{\hat{\sum}_{x\in b}\ \mathrm{c(x)} \in \mathrm{U_0}[\Gamma]}$$

$$\mathrm{I_5\text{-}Uno)}\ \frac{\mathrm{c(x)} \in \mathrm{U_0}[\Gamma,\mathrm{x} \in \mathbf{T}(\mathrm{b})] \qquad \mathrm{b} \in \mathrm{U_0}[\Gamma]}{\hat{\prod}_{x\in b}\ \mathrm{c(x)} \in \mathrm{U_0}[\Gamma]}$$

$$\mathrm{I_6\text{-}Uno)}\ \frac{\mathrm{b} \in \mathrm{U_0}[\Gamma] \qquad \mathrm{c} \in \mathrm{U_0}[\Gamma]}{\mathrm{b}\ \hat{+}\ \mathrm{c} \in \mathrm{U_0}[\Gamma]} \qquad \mathrm{I_7\text{-}Uno)}\ \frac{\mathrm{b} \in \mathrm{U_0}[\Gamma]}{\hat{List}(b) \in \mathrm{U_0}[\Gamma]}$$

In (I_7-Uno) b rappresenta la codifica di un tipo in un contenitore e $\hat{List}(b)$ il codice delle liste associate a b.

$$\mathrm{I_8\text{-}Uno)}\ \frac{\mathrm{b} \in \mathrm{U_0}[\Gamma] \qquad \mathrm{c} \in \mathbf{T}(\mathrm{b})[\Gamma] \qquad \mathrm{d} \in \mathbf{T}(\mathrm{b})[\Gamma]}{\hat{Id}(\mathrm{b,c,d}) \in \mathrm{U_0}[\Gamma]}$$

12.3 Regole di Eliminazione

$$\mathrm{E\text{-}Uno)}\ \frac{\mathrm{a} \in \mathrm{U_0}[\Gamma]}{\mathbf{T}(\mathrm{a})\ \mathrm{type}[\Gamma]}$$

T *indica la decodifica del codice a, e trasforma il codice in un tipo.*

12.4 Regole di Conversione

$$\mathrm{C_1\text{-}Uno)}\ \frac{\Gamma\ \mathrm{cont}}{\mathbf{T}(\hat{N_0}) = \mathrm{N_0}\ \mathrm{type}[\Gamma]} \qquad \mathrm{C_2\text{-}Uno)}\ \frac{\Gamma\ \mathrm{cont}}{\mathbf{T}(\hat{N_1}) = \mathrm{N_1}\ \mathrm{type}[\Gamma]}$$

$$\mathrm{C_3\text{-}Uno)}\ \frac{\Gamma\ \mathrm{cont}}{\mathbf{T}(\hat{Nat}) = \mathrm{Nat}\ \mathrm{type}[\Gamma]}$$

$$\mathrm{C_4\text{-}Uno)}\ \frac{\mathrm{c(x)} \in \mathrm{U_0}[\Gamma,\mathrm{x} \in \mathbf{T}(\mathrm{b})] \qquad \mathrm{b} \in \mathrm{U_0}[\Gamma]}{\mathbf{T}(\hat{\sum}_{x\in b} c(x)) = \sum_{x\in \mathbf{T}(b)} \mathrm{T(c(x))}\ \mathrm{type}[\Gamma]}$$

$$\mathrm{C_5\text{-}Uno)}\ \frac{\mathrm{c(x)} \in \mathrm{U_0}[\Gamma,\mathrm{x} \in \mathbf{T}(\mathrm{b})] \qquad \mathrm{b} \in \mathrm{U_0}[\Gamma]}{\mathbf{T}(\hat{\prod}_{x\in b} c(x)) = \prod_{x\in \mathbf{T}(b)} \mathrm{T(c(x))}\ \mathrm{type}[\Gamma]}$$

$$\text{C}_6\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma] \qquad \text{c} \in \text{U}_0[\Gamma]}{\mathbf{T}(\text{b} \mathbin{\hat{+}} \text{c}) = \mathbf{T}(\text{b}) + \mathbf{T}(\text{c})\ \text{type}[\Gamma]} \qquad \text{C}_7\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma]}{\mathbf{T}(\hat{List}(\text{b})) = \text{List}(\mathbf{T}(\text{b}))\ \text{type}[\Gamma]}$$

$$\text{C}_8\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma] \qquad \text{c} \in \mathbf{T}(\text{b})[\Gamma] \qquad \text{d} \in \mathbf{T}(\text{b})[\Gamma]}{\mathbf{T}(\hat{Id}(\text{b,c,d})) = \text{Id}(\mathbf{T}(\text{b}),\text{c,d})\ \text{type}[\Gamma]}$$

12.5 Regole di Uguaglianza

$$\text{eq-I}_4\text{-Uno)}\ \frac{\text{c}_1(\text{x}) = \text{c}_2(\text{x}) \in \text{U}_0[\Gamma,\text{x} \in \mathbf{T}(\text{b})] \qquad \text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma]}{\hat{\sum}_{x\in b_1} \text{c}_1(\text{x}) = \hat{\sum}_{x\in b_2} \text{c}_2(\text{x}) \in \text{U}_0[\Gamma]}$$

$$\text{eq-I}_5\text{-Uno)}\ \frac{\text{c}_1(\text{x}) = \text{c}_2(\text{x}) \in \text{U}_0[\Gamma,\text{x} \in \mathbf{T}(\text{b})] \qquad \text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma]}{\hat{\prod}_{x\in b_1} \text{c}_1(\text{x}) = \hat{\prod}_{x\in b_2} \text{c}_2(\text{x}) \in \text{U}_0[\Gamma]}$$

$$\text{eq-I}_6\text{-Uno)}\ \frac{\text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma] \qquad \text{c}_1 = \text{c}_2 \in \text{U}_0[\Gamma]}{\text{b}_1 \mathbin{\hat{+}} \text{c}_1 = \text{b}_2 \mathbin{\hat{+}} \text{c}_2 \in \text{U}_0[\Gamma]}$$

$$\text{eq-I}_7\text{-Uno)}\ \frac{\text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma]}{\hat{List}(\text{b}_1) = \hat{List}(\text{b}_2) \in \text{U}_0[\Gamma]}$$

$$\text{eq-I}_8\text{-Uno)}\ \frac{\text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma] \qquad \text{c}_1 = \text{c}_2 \in \mathbf{T}(\text{b})[\Gamma] \qquad \text{d}_1 = \text{d}_2 \in \mathbf{T}(\text{b})[\Gamma]}{\hat{Id}(\text{b}_1,\text{c}_1,\text{d}_1) = \hat{Id}(\text{b}_2,\text{c}_2,\text{d}_2) \in \text{U}_0[\Gamma]}$$

12.6 Semantica operazionale tipo Primo Universo

Negli universi normalizzare i termini è più difficile e richiede una relazione di riduzione che coinvolge anche i tipi, vale dunque il **teorema di forte normalizzazione** (§2.5.1).
La relazione $\rightarrow_1$ viene definita all'interno dei termini con l'uso delle seguenti regole di riduzione:

- β_{Uno}-C$_1$-red) $\mathbf{T}(\hat{N}_0) \rightarrow_1 \text{N}_0$
- β_{Uno}-C$_2$-red) $\mathbf{T}(\hat{N}_1) \rightarrow_1 \text{N}_1$
- β_{Uno}-C$_3$-red) $\mathbf{T}(\hat{Nat}) \rightarrow_1 \text{Nat}$
- β_{Uno}-C$_4$-red) $\mathbf{T}(\hat{\sum}_{x\in b} c(x)) \rightarrow_1 \sum_{x\in\mathbf{T}(b)} \mathbf{T}(c(x))$
- β_{Uno}-C$_5$-red) $\mathbf{T}(\hat{\prod}_{x\in b} c(x)) \rightarrow_1 \prod_{x\in\mathbf{T}(b)} \mathbf{T}(c(x))$
- β_{Uno}-C$_6$-red) $\mathbf{T}(\text{b} \mathbin{\hat{+}} \text{c}) \rightarrow_1 \mathbf{T}(\text{b}) + \mathbf{T}(\text{c})$
- β_{Uno}-C$_7$-red) $\mathbf{T}(\hat{List}(\text{b})) \rightarrow_1 \text{List}(\mathbf{T}(\text{b}))$

- β_{Uno}-C$_8$-red) $\mathbf{T}(\hat{Id}(b,c,d)) \rightarrow_1 \mathrm{Id}(\mathbf{T}(b),c,d)$
- Uno-red) $\dfrac{a_1 \to a_2}{\mathbf{T}(a_1) \rightarrow_1 \mathbf{T}(a_2)}$
- novità del tipo Universo rispetto al tipo singoletto
  Uno-I$_4$-red$_I$) $\dfrac{c_1 \rightarrow_1 c_2 \qquad b_1 \rightarrow_1 b_2}{\hat{\sum}_{x\in b_1} c_1(x) \rightarrow_1 \hat{\sum}_{x\in b_2} c_2(x)}$
- *Uno*-I$_5$-red$_I$) $\dfrac{c_1 \rightarrow_1 c_2 \qquad b_1 \rightarrow_1 b_2}{\hat{\prod}_{x\in b_1} c_1(x) \rightarrow_1 \hat{\prod}_{x\in b_2} c_2(x)}$
- *Uno*-I$_6$-red$_I$) $\dfrac{b_1 \rightarrow_1 b_2}{b_1 \hat{+} c \rightarrow_1 b_2 \hat{+} c}$ *Uno*-I$_6$-red$_{II}$) $\dfrac{c_1 \rightarrow_1 c_2}{b \hat{+} c_1 \rightarrow_1 b \hat{+} c_2}$
- *Uno*-I$_7$-red$_I$) $\dfrac{d_1 \rightarrow_1 d_2}{\hat{List}(d_1) \rightarrow_1 \hat{List}(d_2)}$
- *Uno*-I$_8$-red$_I$) $\dfrac{t_1 \rightarrow_1 t_2}{\hat{Id}(t_1,w,z) \rightarrow_1 \hat{Id}(t_2,w,z)}$ *Uno*-I$_8$-red$_{II}$) $\dfrac{w_1 \rightarrow_1 w_2}{\hat{Id}(t,w_1,z) \rightarrow_1 \hat{Id}(t,w_2,z)}$
  Uno-I$_8$-red$_{III}$) $\dfrac{z_1 \rightarrow_1 z_2}{\hat{Id}(t,w,z_1) \rightarrow_1 \hat{Id}(t,w,z_2)}$

12.7 Esercizio su Universo: 0 diverso da 1 proposizionalmente

Il sequente esercizio è utile a comprendere come la logica si interpreta all'interno della teoria dei tipi.
Inoltre mette in luce, come l'uso degli Universi, permette di dimostrare alcuni fatti banali dell'aritmetica, e in questo caso, l'assioma di Peano.

Voglio dimonstrare che ¬Id(Nat,0,1) è derivabile, ovvero che esiste pf tale che

$$\mathbf{pf} \in \mathbf{Id(Nat,0,1)} \to \mathbf{N_0[\ ]}$$

Questo perchè $N_0 \equiv \perp$ e $\mathrm{Id(Nat,0,1)} \equiv 0 =_{Nat} 1$
Che è equivalente a trovare (con *(I-→)*)

$$\mathbf{pf(z)} \in \mathbf{N_0[z} \in \mathbf{Id(Nat,0,1)]}$$

Ricordo che l'uguaglianza è Id e non 0 = 1 ∈ Nat (0 è una NF ≠ succ(0) = 1 che è una NF) che difatti non è derivabile.

L'unico modo per trovare pf è solo utilizzando un Universo ($\mathbf{U_0}$).

$$\begin{array}{c} U_0 \\ \hat{U_0} \in U_1 \\ \hat{U_1} \in U_2 \\ \ldots \end{array}$$

Nella teoria dei tipi di *Martin-Löf* è possibile sempre definire U_0 e poi mettere il codice di U_0 dento a U_1, il codice di U_1 dentro a U_2, e cosi via, senza mai far collassare gli universi.
Al netto di ciò, in questo esercizio, basterebbe usare un Universo U_k e non solo U_0.

12.7.1 Lemma 1

Definisco la funzione *isZero*, indispensabile per capire se z è 0 oppure 1. Dunque

$$\mathbf{isZero(z)[z \in Nat]}$$

Idea: pensare a U_0 come se fosse Bool (in realtà U_0 ne contiene molti di più).

- vero: $\hat{N}_1 \to T(\hat{N}_1) = N_1 = tt^I$
- falso: $\hat{N}_0 \to T(\hat{N}_0) = N_0 = \bot^I$

È vero che z è zero, per cui nel caso base va messo $\hat{N}_1$
Il successore di qualcosa non è zero, e dunque va messo $\hat{N}_0$

$$\text{E-Nat}_{dip}\ \dfrac{\text{F-Uno}\ \dfrac{[\,]\ \text{cont}}{U_0\ \text{type}[\,]} \qquad \text{I}_2\text{-Uno}\ \dfrac{[\,]\ \text{cont}}{\hat{N}_1 \in U_0[\,]} \qquad \text{ind-te}\ \dfrac{\text{I}_1\text{-Uno}\ \dfrac{[\,]\ \text{cont}}{\hat{N}_0 \in U_0[\,]} \qquad \text{F-c}\ \dfrac{\text{F-Uno}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}\ [\,]}}{\text{x} \in \text{Nat cont}}\ (\text{x} \in \text{Nat}) \notin [\,]}{U_0\ \text{type}[\text{x} \in \text{Nat}]}\ \begin{matrix}(\text{y} \in U_0)\\ \notin \text{x} \in \text{Nat}\end{matrix}}{\text{x} \in \text{Nat, y} \in U_0\ \text{cont}}}{\hat{N}_0 \in U_0[\text{x} \in \text{Nat, y} \in U_0]}}{\text{El}_{Nat}(\text{z},\hat{N}_1,(\text{x,y}).\hat{N}_0) \in U_0[\text{z} \in \text{Nat}]}$$

1. Verifico che *isZero*$(0) = \hat{N}_1 \in U_0[\,]$ (*per farlo uso la regola di Conversione dei Naturali*)

$$\text{C}_1\text{-Nat}\ \dfrac{\text{F-Uno}\ \dfrac{[\,]\ \text{cont}}{U_0\ \text{type}[\,]} \qquad \text{I}_2\text{-Uno}\ \dfrac{[\,]\ \text{cont}}{\hat{N}_1 \in U_0[\,]} \qquad \dfrac{(*)}{\hat{N}_0 \in U_0[\text{x} \in \text{Nat, y} \in U_0]}}{\text{El}_{Nat}(0,\hat{N}_1,(\text{x,y}).\hat{N}_0) = \hat{N}_1 \in U_0\ [\,]}$$

2. Verifico che *isZero*$(1) = \hat{N}_0 \in U_0[\,]$

$$\text{C}_2\text{-Nat}\ \dfrac{\text{F-Uno}\ \dfrac{[\,]\ \text{cont}}{U_0\ \text{type}[\,]} \qquad \text{I}_1\text{-Nat}\ \dfrac{[\,]\ \text{cont}}{0 \in \text{Nat}\ [\,]} \qquad \text{I}_2\text{-Uno}\ \dfrac{[\,]\ \text{cont}}{\hat{N}_1 \in U_0[\,]} \qquad \dfrac{(*)}{\hat{N}_0 \in U_0[\text{x} \in \text{Nat, y} \in U_0]}}{\text{El}_{Nat}(\text{succ}(0),\hat{N}_1,(\text{x,y}).\hat{N}_0) = \hat{N}_0 \in U_0\ [\,]}$$

Uso $(*)$ per indicare le derivazioni già svolte all'interno dello stesso esercizio.

Dunque

$$\textbf{\textit{isZero}(z) equivale a El}_{Nat}(\mathbf{z},\hat{N}_1,(\mathbf{x,y}).\hat{N}_0)$$
$$\textbf{con \textit{isZero}(0)} = \hat{N}_1 \in \mathbf{U_0[\,]}\ \textbf{e \textit{isZero}(1)} = \hat{N}_0 \in \mathbf{U_0[\,]}$$

12.7.2 Lemma 2

$h_{isZero}(z_1,z_2,z_3) \in \mathrm{Id}(U_0,\mathrm{isZero}(z_1),\mathrm{isZero}(z_2))[z_1 \in \mathrm{Nat}, z_2 \in \mathrm{Nat}, z_3 \in \mathrm{Id}(\mathrm{Nat},z_1,z_2)]$
è derivabile.
$h_{isZero}(z_1,z_2,z_3)$ è equivalente a $\mathrm{El}_{Id}(z_3,(x).\mathrm{id}(\mathrm{isZero}(x))$ (derivazione presente in §6.9 esercizio 2) è anch'essa derivabile.

Corollario

$h_{isZero}(0,1,z) \in \mathrm{Id}(U_0,\mathrm{isZero}(0),\mathrm{isZero}(1))[z \in \mathrm{Id}(\mathrm{Nat},0,1)]$ è derivabile.
Inoltre visto che *isZero*$(0) = \hat{N}_1$ e *isZero*$(1) = \hat{N}_0$ anche

$$\mathbf{h}_{isZero}\mathbf{(0,1,z)\ Id(U_0,}\hat{N}_1,\hat{N}_0\mathbf{)[z} \in \mathbf{Id(Nat,0,1)]}$$

è derivabile.

$$\text{eq-F-Id}\ \dfrac{\dfrac{\mathbf{1}}{U_0 = U_0\ \mathrm{type}[z \in \mathrm{Id}(\mathrm{Nat},0,1)]} \qquad \dfrac{\mathbf{2}}{\mathrm{isZero}(0) = \hat{N}_1 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]} \qquad \dfrac{\mathbf{3}}{\mathrm{isZero}(1) = \hat{N}_0 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]}}{\mathrm{Id}(U_0,\mathrm{isZero}(0),\mathrm{isZero}(1)) = \mathrm{Id}(U_0,\hat{N}_1,\hat{N}_0)[z \in \mathrm{Id}(\mathrm{Nat},0,1)]}$$

1

$$\text{ref}\ \dfrac{\text{F-Uno}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Id}\ \dfrac{\text{F-Nat}\ \dfrac{[\ ]\ \mathrm{cont}}{\mathrm{Nat\ type}\ [\ ]} \qquad \text{I}_1\text{-Nat}\ \dfrac{[\ ]\ \mathrm{cont}}{0 \in \mathrm{Nat}\ [\ ]} \qquad \text{I}_2\text{-Nat}\ \dfrac{\dfrac{(*)}{0 \in \mathrm{Nat}\ [\ ]}}{1 \in \mathrm{Nat}\ [\ ]}}{\mathrm{Id}(\mathrm{Nat},0,1)\ \mathrm{type}\ [\ ]} \quad (z \in \mathrm{Id}(\mathrm{Nat},0,1)) \notin [\ ]}{z \in \mathrm{Id}(\mathrm{Nat},0,1)\ \mathrm{cont}}}{U_0\ \mathrm{type}[z \in \mathrm{Id}(\mathrm{Nat},0,1)]}}{U_0 = U_0\ \mathrm{type}[z \in \mathrm{Id}(\mathrm{Nat},0,1)]}$$

2

isZero$(0) = \hat{N}_1 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)] \equiv \mathrm{El}_{Nat}(0,\ \hat{N}_1,(x,y).\hat{N}_0) = \hat{N}_1 \in U_0$ $[z \in \mathrm{Id}(\mathrm{Nat},0,1)]$

$$\text{C}_1\text{-Nat}\ \dfrac{\dfrac{(*)}{U_0\ \mathrm{type}\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]} \qquad \text{I}_2\text{-Uno}\ \dfrac{\dfrac{(*)}{z \in \mathrm{Id}(\mathrm{Nat},0,1)\ \mathrm{cont}}}{\hat{N}_1 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]} \qquad \text{I}_1\text{-Uno}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Uno}\ \dfrac{\text{F-c}\ \dfrac{\text{F-Nat}\ \dfrac{\dfrac{(*)}{z \in \mathrm{Id}(\mathrm{Nat},0,1)\ \mathrm{cont}}}{\mathrm{Nat\ type}[z \in \mathrm{Id}(\mathrm{Nat},0,1)]} \quad \begin{matrix}(x \in \mathrm{Nat}) \notin \\ z \in \mathrm{Id}(\mathrm{Nat},0,1)\end{matrix}}{z \in \mathrm{Id}(\mathrm{Nat},0,1),\ x \in \mathrm{Nat}\ \mathrm{cont}}}{U_0\ \mathrm{type}[z \in \mathrm{Id}(\mathrm{Nat},0,1),\ x \in \mathrm{Nat}]} \quad \begin{matrix}(y \in U_0) \notin (z \in \\ \mathrm{Id}(\mathrm{Nat},0,1),\ x \in \mathrm{Nat})\end{matrix}}{z \in \mathrm{Id}(\mathrm{Nat},0,1),\ x \in \mathrm{Nat},\ y \in U_0\ \mathrm{cont}}}{\hat{N}_0 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1),\ x \in \mathrm{Nat},\ y \in U_0]}}{\mathrm{El}_{Nat}(0,\ \hat{N}_1,(x,y).\hat{N}_0) = \hat{N}_1 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]}$$

3

isZero$(1) = \hat{N}_0 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)] \equiv \mathrm{El}_{Nat}(\mathrm{succ}(0),\ \hat{N}_1,(x,y).\hat{N}_0) = \hat{N}_0$ $\in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]$

$$\text{C}_2\text{-Nat}\ \dfrac{\text{I}_1\ \text{Nat}\ \dfrac{\dfrac{(*)}{z \in \mathrm{Id}(\mathrm{Nat},0,1)\ \mathrm{cont}}}{0 \in \mathrm{Nat}\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]} \qquad \dfrac{(*)}{U_0\ \mathrm{type}\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]} \qquad \text{I}_1\text{-Uno}\ \dfrac{\dfrac{(*)}{z \in \mathrm{Id}(\mathrm{Nat},0,1)\ \mathrm{cont}}}{\hat{N}_0 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]} \qquad \dfrac{(*)}{\hat{N}_0 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1),\ x \in \mathrm{Nat},\ y \in U_0]}}{\mathrm{El}_{Nat}(\mathrm{succ}(0),\ \hat{N}_1,(x,y).\hat{N}_0) = \hat{N}_0 \in U_0\ [z \in \mathrm{Id}(\mathrm{Nat},0,1)]}$$

Ho usato $(*)$ per concludere le derivazioni già svolte all'interno dell'albero.

Ecco che se $0 = 1$ siamo vicino alla contraddizione, in quanto usando che *isZero*$(0) = \hat{N}_1$ e *isZero*$(1) = \hat{N}_0$, allora anche $\hat{N}_0 = \hat{N}_1$ nell'Universo. Ma $\hat{N}_0$ è *false* e $\hat{N}_1$ è *true* e se *true* fosse uguale a *false* avremmo una logica inconsistente.

12.7.3 Lemma 3

Serve per arrivare ad avere una contraddizione.

q_{U_0}(x,y,z) $\in$ (T(x) $\to$ T(y)) $\times$ (T(y) $\to$ T(x))[x $\in$ U$_0$,y $\in$ U$_0$,z $\in$ Id(U$_0$,x,y)] è derivabile, equivalente a El$_{Id}$(z,(x).< λw.w,λw.w>) $\in$ (T(x) $\to$ T(y)) $\times$ (T(x) $\to$ T(y)).

$\Delta \equiv$ x $\in$ U$_0$,y $\in$ U$_0$,z $\in$ Id(U$_0$,x,y)

$$
\text{E-Id}_{dip}\ \dfrac{
\text{F-}\times\ \dfrac{\dfrac{}{(T(x) \to T(y))\ type[\Delta]} \quad \dfrac{}{(T(x) \to T(y))\ type[\Delta]}}{(T(x) \to T(y)) \times (T(x) \to T(y))\ type[\Delta]}
\qquad
\text{I-}\times\ \dfrac{
\text{I-}\to\ \dfrac{\text{var}\ \dfrac{\text{F-c}\ \dfrac{\dfrac{}{T(x)\ type[x \in U_0]}}{x \in U_0, w \in T(x)\ cont}\ (w \in T(x)) \notin x \in U_0}{w \in T(x)[x \in U_0, w \in T(x)]}}{\lambda w.w \in T(x) \to T(x)[x \in U_0]}
\qquad
\dfrac{(*)}{\lambda w.w \in T(x) \to T(x)[x \in U_0]}
}{< \lambda w.w, \lambda w.w> \in (T(x) \to T(x)) \times (T(x) \to T(x))[x \in U_0]}
}{El_{Id}(z,(x).< \lambda w.w, \lambda w.w>) \in (T(x) \to T(y)) \times (T(x) \to T(y))[\Delta]}
$$

Ho usato $(*)$ per concludere le derivazioni già svolte all'interno dell'albero.

Corollario

Da q_{U_0}(x,y,z) $\in$ (T(x) $\to$ T(y)) $\times$ (T(y) $\to$ T(x))[Δ] sostituendo x con $\hat{N}_1$ e y con $\hat{N}_0$ ottengo

$$q_{U_0}(\hat{N}_1,\hat{N}_0,z) \in (T(\hat{N}_1) \to T(\hat{N}_0)) \times (T(\hat{N}_0) \to T(\hat{N}_1))[z \in Id(Nat,0,1)]$$

che è derivabile. Inoltre mediante sostituzione trovo che

$$
\text{eq-F-}\times\ \dfrac{
\text{eq-F-}\prod\ \dfrac{\text{C}_2\text{-Uno}\ \dfrac{\Gamma\ cont}{T(\hat{N}_1) = N_1\ type[\Gamma]} \quad \text{C}_1\text{-Uno}\ \dfrac{\Gamma\ cont}{T(\hat{N}_0) = N_0\ type[\Gamma]}}{T(\hat{N}_1) \to T(\hat{N}_0) = N_1 \to N_0}
\qquad
\text{eq-F-}\prod\text{-F}\ \dfrac{\text{C}_1\text{-Uno}\ \dfrac{\Gamma\ cont}{T(\hat{N}_0) = N_0\ type[\Gamma]} \quad \text{C}_2\text{-Uno}\ \dfrac{\Gamma\ cont}{T(\hat{N}_1) = N_1\ type[\Gamma]}}{T(\hat{N}_0) \to T(\hat{N}_1) = N_0 \to N_1}
}{(T(\hat{N}_1) \to T(\hat{N}_0)) \times (T(\hat{N}_0) \to T(\hat{N}_1)) = (N_1 \to N_0) \times (N_0 \to N_1)}
$$

$$\mathbf{q_{U_0}(\hat{N}_1,\hat{N}_0,z) \in (N_1 \to N_0) \times (N_0 \to N_1)}$$

è derivabile.

Corollario finale

Ap($\pi_1 q_{U_0}(\hat{N}_1, \hat{N}_0, z`)$), $*$)$\in$ N$_0$[z` $\in$ Id(U$_0$,$\hat{N}_1$,$\hat{N}_0$)]
Dunque $\perp$ true [Id(U$_0$,$\hat{N}_1$,$\hat{N}_0$) true] è derivabile.

Inoltre h$_{isZero}$(0,1,z) $\in$ Id(U$_0$,$\hat{N}_1$,$\hat{N}_0$)[z $\in$ Id(Nat,0,1)] è equivalente a Id(U$_0$,$\hat{N}_1$,$\hat{N}_0$) true[z $\in$ Id(Nat,0,1)] che è derivabile.
Per cui mettendo assieme, usando la logica, che Id(U$_0$,$\hat{N}_1$,$\hat{N}_0$) true[z $\in$ Id(Nat,0,1)] e $\perp$ true [Id(U$_0$,$\hat{N}_1$,$\hat{N}_0$) true] si ottiene $\perp$ true [Id(Nat,0,1)] che è ancora derivabile.

Il *proof-term* non è altro che prendere Ap($\pi_1 q_{U_0}(\hat{N}_1, \hat{N}_0, z`)$),$*$)$\in$ N$_0$[z` $\in$ Id(U$_0$,$\hat{N}_1$,$\hat{N}_0$)] e comporlo con h$_{isZero}$(0,1,z) $\in$ Id(U$_0$,$\hat{N}_1$,$\hat{N}_0$)[z $\in$ Id(Nat,0,1)]

$$\mathbf{Ap}(\pi_1 q_{U_0}(\hat{N}_1, \hat{N}_0, h_{isZero}(\mathbf{0{,}1{,}z})),*) \in \mathbf{N_0[z \in Id(Nat{,}0{,}1)]}$$
$$\lambda\mathbf{z.Ap}(\pi_1 q_{U_0}(\hat{N}_1, \hat{N}_0, h_{isZero}(\mathbf{0{,}1{,}z})),*) \in \neg\mathbf{Id(Nat{,}0{,}1)}$$

Tale *proof-term*, non è altro che la codifica del *proof-assistant*, della dimostrazione che ¬Id(Nat,0,1) true[ ].

12.8 Peculiarità della teoria dei tipi di *Martin-Löf*

Per teoria dei tipi di *Martin-Löf* si intende

$$\mathrm{N_0,N_1,\ Nat,\ List,\ B + C,\ Id(A,a,b),\ } \sum_{x\in A} \mathrm{B(x)},\ \prod_{x\in A} \mathrm{B(x),\ U_0} + \text{altri universi}$$

Questa è la teoria **MLTT**.
La teoria **MLTT** è **aperta**, ovvero si intende anche l'estensione della teoria specifica, con i tipi elencati, con altri tipi induttivi, secondo lo schema delle regole date in §2.4, in modo predicativo. In tale teoria deve sempre valere il teorema della forma normale §2.5.1 (la confluenza), e in generale la correttezza dei 4 giudizi deve essere decidibile (*decidable type checking*).

$$\mathbf{MLTT}_0 \stackrel{def}{=} \mathbf{MLTT} - \text{Universo e con solo i tipi elencati sopra}$$
$$\mathbf{MLTT}_1 \stackrel{def}{=} \mathbf{MLTT}_0 + \mathrm{U_0}\ \text{universo}$$

A titolo di esempio, la teoria $\mathbf{MLTT}_1$ è quella spiegata nella teoria del corso.

12.8.1 Necessità di un Universo per formalizzare l'aritmetica

Se prendiamo la teoria $\mathbf{MLTT}_0$ (senza l'Universo)

$$\equiv \mathrm{N_0,\ N_1,\ List(A),\ Nat,\ B + C,\ Id(A,a,b),\ } \sum_{x\in A} \mathrm{B(x)},\ \prod_{x\in A} \mathrm{B(x)}$$

risulta **inadeguata** per formalizzare la matematica, perchè non dimostra che i Naturali sono formati da infiniti numeri. In particolare non è in grado di dimostrare che $0 \neq 1$ proposizionalmente. Formalmente significa

¬ Id(Nat,0,1) true[ ] non è derivabile in $\mathbf{MLTT}_0$ (ma lo è con l'Universo)

Al massimo in **MLTT**(*Martin-Löf's style theory*) si può formalizzare in *Costructive Math* = *Math* (matematica usuale che si può dimostrare con la Logica intuizionisitica) + Logica intuizionistica.
Invece *Classic Math* = *Math* + Logica intuizionistica + $\varphi \vee \neg\varphi$ true[Γ] per ogni φ
⇒ Logica intuizionistica + $\varphi \vee \neg\varphi$ true[Γ] per ogni φ = Logica classica
⇒ $\varphi \vee \neg\varphi$ true[Γ] per ogni φ non vale in **MLTT** (come già accennato all'interno di §10).
In **MLTT** non si può formalizzare la *Classic Math*, al massimo solo quella *Costructive*.

Dimostrazione del perchè $\mathbf{MLTT_0}$, senza Universo, è indeguata

Per dimostrare l'aritmetica intuizionistica bisogna almeno essere in MLTT_1, ci vuole almeno U_0 Universo.

Dimostro che $MLTT_0$ non è sufficiente, con il seguente esempio

$$\mathbf{0 \neq 1\ prop \equiv \neg Id(Nat,0,1)}$$

1. **Costruire un contromodello interno di $\mathbf{MLTT}_0$**, che renda valido tutte le regole di $\mathbf{MLTT}_0$, con l'assunzione che $\mathbf{MLTT}_0$ sia consistente. Ma $(\neg\mathrm{Id}(\mathrm{Nat},0,1))^J = \mathrm{N}_0$ type [ ] = $\perp$ type [ ].

   Idea: $(\ )^J$ tipi di $\mathrm{MLTT}_0 \rightarrow \{\mathrm{N}_0,\mathrm{N}_1\} \equiv$ Bool (ovvero ogni tipo viene interpretato come proposizione *true* o *false*)
   Ciò significa che interpreto $(\mathrm{B\ type}[\Gamma])^J$, all'interno di $\mathbf{MLTT}_0$ stesso, come $\mathrm{B}^J\ \mathrm{type}[\mathrm{z} \in \mathrm{N}^J]$
   Inoltre interpreto un *proof-term* $(\mathrm{pf} \in \mathrm{B\ type}[\Gamma])^J \stackrel{def}{=} * \in \mathrm{B}^J[\Gamma]$ sse $\mathrm{B}^J = \mathrm{N}_1$ (*interpretazione parziale*).

   Con tale interpretazione si ottiene il seguente **Teorema**
   $\mathrm{pf} \in \mathrm{B}[\Gamma]$ è derivabile $\rightarrow * \in \mathrm{B}^J[\Gamma^J]$ è derivabile in $\mathbf{MLTT}_0$. Questo vale anche per i tipi, dunque se $\mathrm{B\ type}[\Gamma]$ è derivabile $\rightarrow \mathrm{B}^J\ \mathrm{type}[\Gamma^J]$ è derivabile.

2. **Interpretazione**

$$(\mathrm{N}_0[\Gamma])^J \stackrel{def}{=} \mathrm{N}_0[\Gamma^J]$$
$$(\mathrm{N}_1[\Gamma])^J \stackrel{def}{=} \mathrm{N}_1[\Gamma^J]$$
$$(\mathrm{List(A)}[\Gamma])^J \stackrel{def}{=} \mathrm{A}^J[\Gamma^J]$$
$$(\mathrm{Nat}[\Gamma])^J \stackrel{def}{=} \mathrm{N}_1[\Gamma^J]$$
$$(\mathrm{Id(A,a,b)}[\Gamma])^J \stackrel{def}{=} \mathrm{A}^J[\Gamma^J]$$

   Con questo modello allora se scrivo $\mathrm{Id(A,a,b)}^J$ viene interpretato come $\mathrm{Nat}^J = \mathrm{N}_1$
   Inoltre possiamo derivare $* \in \mathrm{Id(A,a,b)}^J$ in $\mathbf{MTT}_0$.
   Se $Id(A,a,b)^J$ è abitata, ed è vero, non può però essere abitata la negazione.
   Formalmente la negazione viene interpretata come

$$\neg\ \mathbf{Id(Nat,0,1)} \stackrel{def}{=} (\mathbf{Id(Nat,0,1)} \rightarrow \mathbf{N}_0)^J$$

   Qui va data l'interpretazione dello spazio di funzioni, caso particolare di Π

$$(\prod_{x \in A} \mathrm{B})^J = \begin{cases} \mathbf{N}_0 \text{ se } A^J = N_1 \text{ e } B^J = N_0 \\ \mathbf{N}_1 \text{ altrimenti} \end{cases}$$

   Pensate come proposizioni
   $(\prod_{x \in A} \mathrm{B})^J \equiv \mathrm{A}^J \rightarrow \mathrm{B}^J \equiv$

$tt \rightarrow tt \equiv tt$
$\perp \rightarrow tt \equiv tt$
$\perp \rightarrow \perp \equiv tt$
$tt \rightarrow \perp \equiv \perp$

Dunque $\neg$ Id(Nat,0,1) $\stackrel{def}{=}$ (Id(Nat,0,1)$\rightarrow$ $\mathrm{N}_0)^J = \prod_{x \in Id(Nat,0,1)^J} \mathrm{N}_0 = \prod_{x \in N_1} \mathrm{N}_0 = \mathrm{N}_0 = \mathrm{N}_0(\perp)$

In questo modo ho dimostrato che non è vero che $\mathbf{0 \neq 1}$ proposizionalemente.

Per completezza do anche le interpretazioni di $\sum$ e $+$

$$(\sum_{x \in A} \mathrm{B})^J = \begin{cases} \mathbf{N}_1 \text{ se } A^J = N_1 \text{ e } B^J = N_1 \\ \mathbf{N}_0 \text{ altrimenti} \end{cases}$$

Pensate come proposizioni
$(\sum_{x \in A} \mathrm{B})^J \equiv \mathrm{A}^J \times \mathrm{B}^J \equiv$
$tt \times tt \equiv tt$
$\perp \times tt \equiv \perp$
$\perp \times \perp \equiv \perp$
$tt \rightarrow \perp \equiv \perp$

$$(\mathrm{A} + \mathrm{B})^J = \begin{cases} \mathbf{N}_0 \text{ se } A^J = B^J = N_0 \\ \mathbf{N}_1 \text{ altrimenti} \end{cases}$$

Pensate come proposizioni
$(\mathrm{A} + \mathrm{B})^J \equiv \mathrm{A}^J \vee \mathrm{B}^J \equiv$
$\perp \vee \perp \equiv \perp$
$\perp \vee tt \equiv tt$
$tt \vee tt \equiv tt$
$tt \vee \perp \equiv tt$

In questo modo abbiamo costruito (tipi + termini di $\mathbf{MLTT}_0)^J \rightarrow \mathbf{MLTT}_0$. Che rappresenta, difatti, uno schiacciamento dei tipi tt (N_1) o $\perp$ (N_0), pensando ai tipi come proposizioni, riuscendo a provare che 0 =1 è vero, in quanto tutti i tipi dei Naturali vengono interpretati unicamente dal singoletto.

Questa interpretazione non funziona più con l'Universo, perchè

$$\mathbf{U}_0^J = \mathbf{N}_1 \Leftrightarrow \mathbf{MLTT}_1 \textbf{ è inconsistente, poichè } (\perp)^J = (tt)^J$$

Invece U_0, essendo Bool, contiene sia il codice di $\hat{N_1}$ che il codice di $\hat{N_0}$.

12.9 Principi di scelta per estrazione di programmi da dimostrazioni

La **MLTT** ha una peculiare proprietà, dipendente dal fatto che abbiamo identificato la quantificazione esistenziale con l'unione indiciata.

$$\exists_{x\in A}\ \varphi(x) \overset{def}{\equiv} \sum_{x\in A} \varphi(x)$$

Da notarsi che quanto descritto successivamente è interno a **MLTT**. Difatti, per altre teorie, può verificarsi la derivazione in una teoria T_1 e l'estrazione della funzione in un'altra T_2.

12.9.1 Regola di scelta

Vale, in questa teoria, l'**estrazione dei programmi** (intesi come funzioni) da dimostrazioni di validità di **specifiche date**.
Ciò significa che se ho una relazione R(x,y) prop[Γ,x ∈ A,y ∈ B] e ho dimostrato $\forall_{x\in A}\ \exists_{y\in B}$ R(x,y) true[Γ] derivato in **MLTT**, allora
⇒ esiste f ∈ A → B[Γ] in **MLLT** tale che $\forall_{x\in A}$ R(x,Ap(f,x)) true[Γ] derivabile in **MLTT**.

Esempio

Supponiamo di aver dimostrato (*specifica proprietà di un programma*) che ∀x ∈ Nat ∃y ∈ Nat B_p(x,y) (= il programma P su input x si ferma al più in y passi) true[Γ], che è equivalente a un'equazione ∀x ∈ Nat ∃y ∈ Nat ax + bx = c
Percui
⇒ esiste b ∈ Nat → Nat[Γ] tale che ∀x ∈ Nat B_p(x,Ap(b,x)) true[Γ] (*restituisce in quanti passi il programma computa*) derivabile, che equivale in termini di equazione a f tale che ax + Ap(f,x) = c ∀x ∈ Nat

Questo permette di risolvere delle disuguaglianze o riuscire a provare l'arrestabilità di un programma, sempre su un certo y, dopo n passi, ed estrarne l'algoritmo che calcola il *bound*/la funzione che da la soluzione dell'equazione.

Suppongo pf ∈ ∀x ∈ Nat ∃y ∈ Nat B_p(x,y)[Γ]
Si deriva b $\overset{def}{=}$ λx.π_1(Ap(pf,x)) ∈ Nat → Nat[Γ] è derivabile
λx.π_2(Ap(pf,x)) ∈ ∀x ∈ Nat B_p(x,Ap(b,x))[Γ] è derivabile

12.9.2 Assioma di scelta

La proprietà della regola della scelta, può venire rafforzata nel modo seguente.

Data una relazione derivabile in **MLTT** vale che → (l'implica) diventa un'implicazione interna
$\forall_{x\in A}\ \exists_{y\in B}$ R(x,y) → $\exists_{f\in A\to B}\ \forall_{x\in A}$ R(x,Ap(f,x)) true[Γ] è derivabile in **MLTT**.

In generale l'assioma di scelta e la regola non sono equivalenti.

12.10 Inconsistenza di MLTT con codice del Primo Universo in se stesso

Non esiste una versione impredicativa di **MLTT** (includendo almeno U_0) che mantenga

$$\exists_x \in \varphi(\mathbf{x}) \equiv \sum_{x \in A} \varphi(\mathbf{x})$$

- ***T* è predicativa:** con univeso di oggetti *small* $\mathrm{U}_0 \to \mathrm{U}_0$ non è *small.* Dunque non si può considerare **MLTT** + $\hat{U}_0 \in \mathrm{U}_0$ (versione à la *Tarski*), oppure $\mathrm{U}_0 \in \mathrm{U}_0$ nella versione di *Russell.*

- ***T* è impredicativa:** almeno con gli esempi introdotti in letteratura. Non solo $\mathrm{U}_0 \in \mathrm{U}_0$, ma se data la premessa $\mathrm{b(x)} \in \mathrm{U}_0[\Gamma,\mathrm{x} \in \mathrm{A}]$, allora

$$\frac{\mathrm{b(x)} \in \mathrm{U}_0[\Gamma,\mathrm{x} \in \mathrm{A}]}{\prod_{x \in A} \mathrm{b(x)} \in \mathrm{U}_0[\Gamma]} \qquad \frac{\mathrm{b(x)} \in \mathrm{U}_0[\Gamma,\mathrm{x} \in \mathrm{A}]}{\sum_{x \in A} \mathrm{b(x)} \in \mathrm{U}_0[\Gamma]}$$

  Difatti se le regole sopra vengono aggiunte a **MLTT**, la teoria diventa **inconsistente**.
  Una teoria si dice inconsistente, quando dimostra il falso, ecco che ogni teoria di *Martin-Löf* è necessario sia predicativa.

12.10.1 Come mai la teoria diventa inconsistente?

Per vedere che $\hat{U}_0 \in \mathrm{U}_0$ diventa inconsistente, si formalizza in **MLTT** + $\hat{U}_0 \in \mathrm{U}_0$ il Paradosso di *Mirimanoff*, in teoria degli insiemi usuali della Matematica

"esiste W *set* tale che W = {X *set* | X è ben fondato}"

In teoria degli insiemi $\nexists$ W $\equiv$ {X *set* | X ben fondato}, ovvero non è possibile da dimostrare l'esistenza di W come insieme.
Per questo W si può scrivere, in teoria degli insiemi, unicamente come una classe o collezione.

Dimostrazione

Definizione: X è *set* ed è ben fondato sse non esiste una cantena discentende $(\mathrm{y}_i)_{i \in Nat}$ con y_i *set* tale che $\mathrm{x} \in \mathrm{y}_0 \in \mathrm{y}_i ... \in \mathrm{y}_i \in \mathrm{y}_{i+1}$ $\forall \mathrm{i} \in \mathrm{Nat}$ (sono finite ecco che è ben fondato).

Suppongo W *set*, allora si dimostra come *lemma cruciale* che W è ben fondato. In quanto se esistesse una cantena $(\mathrm{y}_i)_{i \in Nat}$ $\mathrm{w} \in \mathrm{y}_0 \in \mathrm{y}_i ... \in \mathrm{y}_i \in \mathrm{y}_{i+1}$ per il fatto che y_i è un elemento di w allora è ben fondato. Ma riusciremmo ad avere che la catena $(\mathrm{y}_i)_{i>=1}$ è infinita e questo porterebbe a una contraddizione.

Corollario: Ma allora se W è un insieme e ben fondato, allora w $\in$ W.
Ma ancora abbiamo trovato una contraddizione, perchè grazie alla valenza del corollario possiamo dire che esiste $(\mathrm{y}_i^{\backslash})_{i \in Nat}$ tale che $\mathrm{y}_i^{\backslash} \equiv \mathrm{W}$ per cui $\mathrm{w} \in \mathrm{w} \in \mathrm{w} ... \in \mathrm{w}$ e questo mi premette nuovamente di dire che W non è ben fondato.

Conclusione: non esiste W = {X *set* | X è ben fondato}.

In teoria dei tipi W viene simultato

$$\Omega \equiv W \stackrel{def}{=} \{(\mathbf{X},<) \textbf{ tipi} \mid (\mathbf{X},<) \textbf{ è ben fondato}\}$$

in relazione x < y prop[x,y ∈ X]

Ancora riesco a dimostrare che Ω è ben fondato e che porta a una contraddizione. Ecco che si è dimostrato che in **MLTT** + $\hat{U}_0 \in U_0$ è inconsistente, ovvero ⊥ true [ ] in tale teoria.

Anche altre quantificazioni, come quelle illustrate sopra, allo stesso modo possono essere dimostrate inconsistenti.
Tutto questo rende evidente come l'uso dei paradossi permette di vedere i limiti dei sistemi formali dal punto di vista logico.

12.11 Esercizi

1) Dimostrare in teoria dei tipi con un universo, con almeno il codice del tipo vuoto e del tipo singoletto, ad esempio con U_0, si può derivare

$$\neg \mathbf{Id(Nat,0,1)\ true[\,]}$$

Soluzione

Devo dimostrare che esiste pf ∈ ¬Id(Nat,0,1)[ ] ≡ pf(z) ∈ N_0[z ∈ Id(Nat,0,1)], come già dimostrato in §12.7. Per completezza, di seguito, mostro la derivabilità di ¬ Id(Nat,0,1) type[ ].

$\neg$ Id(Nat,0,1) type[ ] ≡ Id(Nat,0,1) → ⊥ type[ ] ≡ Id(Nat,0,1) →N_0 type[ ] ≡ $\prod_{z\in Id(Nat,0,1)}$ N_0 type[ ]

$$\text{F-}\textstyle\prod\ \dfrac{\text{F-Id}\ \dfrac{\text{F-Nat}\ \dfrac{[\,]\ \text{cont}}{\text{Nat type}[\,]} \quad \text{I}_1\text{-Nat}\ \dfrac{[\,]\ \text{cont}}{0 \in \text{Nat}[\,]} \quad \text{I}_2\text{-Nat}\ \dfrac{\text{I}_1\text{-Nat}\ \dfrac{[\,]\ \text{cont}}{0 \in \text{Nat}[\,]}}{1 \in \text{Nat}[\,]}}{\text{Id(Nat,0,1) type}[\,]} \quad \text{F-N}_0\ \dfrac{\text{F-c}\ \dfrac{\dfrac{(*)}{\text{Id(Nat,0,1) type}[\,]}}{z \in \text{Id(Nat,0,1) cont}}\ (z \in \text{Id(Nat,0,1)}) \notin [\,]}{\text{N}_0\ \text{type}[z \in \text{Id(Nat,0,1)}]}}{\prod_{z\in Id(Nat,0,1)} \text{N}_0\ \text{type}[\,]}$$

Ho usato (∗) per concludere le derivazioni già svolte all'interno dell'albero.

Appendice A

Tipi principali e regole

A.1 Tipo singoletto

A.1.1 Regola di Formazione

$$\text{F-S)}\ \frac{\Gamma \text{ cont}}{N_1 \text{ type}[\Gamma]}$$

A.1.2 Regole di Introduzione

$$\text{I-S)}\ \frac{\Gamma \text{ cont}}{* \in N_1\ [\Gamma]}$$

A.1.3 Regole di Eliminazione

$$\text{E-S)}\ \frac{\text{t} \in N_1\ [\Gamma] \qquad \text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{c} \in \text{M}(*)[\Gamma]}{El_{N1}(\text{t, c}) \in \text{M}(*)[\Gamma]}$$

A.1.4 Regole di Conversione

$$\text{C-S)}\ \frac{\text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{c} \in \text{M}(*)[\Gamma]}{El_{N1}(*, \text{c}) = \text{c} \in \text{M}(*)[\Gamma]}$$

A.1.5 Eliminatore dipendente

$$\text{E-S}_{dip})\ \frac{\text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{c} \in \text{M}(*)[\Gamma]}{El_{N1}(\text{z, c}) \in \text{M(z)}[\Gamma, \text{z} \in N_1]}$$

A.1.6 Regole di Uguaglianza

$$\text{eq-E-N}_1)\ \frac{\text{M(z) type}[\Gamma, \text{z} \in N_1] \qquad \text{t}_1 = \text{t}_2 \in N_1[\Gamma] \qquad \text{c}_1 = \text{c}_2 \in \text{M}(*)[\Gamma]}{\text{El}_{N1}(\text{t}_1, \text{c}_1) = \text{El}_{N1}(\text{t}_2, \text{c}_2) \in \text{M}(t_1)[\Gamma]}$$

A.2 Tipo dei numeri Naturali

A.2.1 Regole di Formazione

$$\text{F-Nat)}\ \frac{\Gamma\ \text{cont}}{\text{Nat type}[\Gamma]}$$

A.2.2 Regole di Introduzione

$$\text{I}_1\text{-Nat)}\ \frac{\Gamma\ \text{cont}}{0 \in \text{Nat}[\Gamma]} \qquad \text{I}_2\text{-Nat)}\ \frac{\text{m} \in \text{Nat}[\Gamma]}{\text{succ(m)} \in \text{Nat}[\Gamma]}$$

A.2.3 Regole di Eliminazione

$$\text{E-Nat)}\ \frac{\text{t} \in \text{Nat}[\Gamma] \qquad \text{M(z) type}[\Gamma,\ \text{z} \in \text{Nat}] \qquad \begin{array}{c}\text{c} \in \text{M(0)}[\Gamma] \\ \text{e(x,y)} \in \text{M(succ(x))}[\Gamma,\ \text{x} \in \text{Nat},\ \text{y} \in \text{M(x)}]\end{array}}{\text{El}_{Nat}\text{(t,c,e)} \in \text{M(t)}[\Gamma]}$$

A.2.4 Regole di Conversione

$$\text{C}_1\text{-Nat)}\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{Nat}] \qquad \text{c} \in \text{M(0)}[\Gamma] \qquad \text{e(x,y)} \in \text{M(succ(x))}[\Gamma,\ \text{x} \in \text{Nat},\ \text{y} \in \text{M(x)}]}{\text{El}_{Nat}\text{(0,c,e)} = \text{c} \in \text{M(0)}[\Gamma]}$$

$$\text{C}_2\text{-Nat)}\ \frac{\text{m} \in \text{Nat}[\Gamma] \qquad \text{M(z) type}[\Gamma,\ \text{z} \in \text{Nat}] \qquad \begin{array}{c}\text{c} \in \text{M(0)}[\Gamma] \\ \text{e(x,y)} \in \text{M(succ(x))}[\Gamma,\ \text{x} \in \text{Nat},\ \text{y} \in \text{M(x)}]\end{array}}{\text{El}_{Nat}\text{(succ(m),c,e)} = \text{e(m, El}_{Nat}\text{(m,c,e))} \in \text{M(succ(m))}[\Gamma]}$$

A.2.5 Regole di Uguaglianza

$$\text{eq-Nat)}\ \frac{\text{t}_1 = \text{t}_2 \in \text{Nat}[\Gamma]}{\text{succ(t}_1\text{)} = \text{succ(t}_2\text{)} \in \text{Nat}[\Gamma]}$$

A.2.6 Introduzione ed Eliminatore dipendente

$$\text{I}_2\text{-Nat}_{prog}\text{)}\ \frac{\Gamma\ \text{cont}}{\text{succ(x)} \in \text{Nat}[\Gamma,\ \text{x} \in \text{Nat}]}$$

$$\text{E-Nat}_{dip}\text{)}\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{Nat}] \qquad \text{c} \in \text{M(0)}[\Gamma] \qquad \text{e(x,y)} \in \text{M(succ(x))}[\Gamma,\ \text{x} \in \text{Nat},\ \text{y} \in \text{M(x)}]}{\text{El}_{Nat}\text{(z,c,e)} \in \text{M(z)}[\Gamma,\ \text{z} \in \text{Nat}]}$$

A.3 Tipo delle liste di un tipo

$$\text{F-cost)}\ \frac{\text{A type }[\Gamma]}{\text{List(A) type}[\Gamma]}$$

A.3.1 Regole di Introduzione

$$\text{I}_1\text{-List)}\ \frac{\text{List(A) type }[\Gamma]}{\text{nil} \in \text{List(A)}[\Gamma]} \qquad \text{I}_2\text{-List)}\ \frac{\text{s} \in \text{List(A)}[\Gamma] \qquad \text{a} \in \text{A}[\Gamma]}{\text{cons(s,a)} \in \text{List(A)}[\Gamma]}$$

A.3.2 Regole di Eliminazione

$$\text{E-List)}\ \frac{\begin{array}{c}\text{M(z) type}[\Gamma,\ \text{z} \in \text{List(A)}]\\ \text{t} \in \text{List(A)}[\Gamma]\\ \text{c} \in \text{M(nil)}[\Gamma]\end{array} \qquad \text{e(x,w,y)} \in \text{M(cons(x,w))}[\Gamma,\ \text{x} \in \text{List(A)},\ \text{w} \in \text{A},\ \text{y} \in \text{M(x)}]}{\text{El}_{List}\text{(t,c,e)} \in \text{M(t)}[\Gamma]}$$

A.3.3 Regole di Conservazione

$$\text{C}_1\text{-List)}\ \frac{\begin{array}{c}\text{M(z) type}[\Gamma,\ \text{z} \in \text{List(A)}]\\ \text{c} \in \text{M(nil)}[\Gamma]\end{array} \qquad \text{e(x,w,y)} \in \text{M(cons(x,w))}[\Gamma,\ \text{x} \in \text{List(A)},\ \text{w} \in \text{A},\ \text{y} \in \text{M(x)}]}{\text{El}_{List}\text{(nil,c,e)} = \text{c} \in \text{M(nil)}[\Gamma]}$$

$$\text{C}_2\text{-List)}\ \frac{\text{M(z) type}[\Gamma,\ \text{z} \in \text{List(A)}] \qquad \begin{array}{c}\text{s} \in \text{List(A)}[\Gamma]\\ \text{a} \in \text{A}[\Gamma]\\ \text{c} \in \text{M(nil)}[\Gamma]\end{array} \qquad \text{e(x,w,y)} \in \text{M(cons(x,w))}[\Gamma,\ \text{x} \in \text{List(A)},\ \text{w} \in \text{A},\ \text{y} \in \text{M(x)}]}{\text{El}_{List}\text{(cons(s,a),c,e)} = \text{e(s,a, E}_{List}\text{(s,c,e))} \in \text{M(cons(s,a))}[\Gamma]}$$

A.3.4 Regole di Uguaglianza

$$\text{eq-I}_1\text{-List)}\ \frac{\text{A}_1 = \text{A}_2\ \text{type}[\Gamma]}{\text{List(A}_1\text{)} = \text{List(A}_2\text{) type(}\Gamma\text{)}}$$

$$\text{eq-I}_2\text{-List)}\ \frac{\text{s}_1 = \text{s}_2 \in \text{List(A)}[\Gamma] \qquad \text{a}_1 = \text{a}_2 \in \text{A}[\Gamma]}{\text{cons(s}_1\text{,a}_1\text{)} = \text{cons(s}_2\text{,a}_2\text{)} \in \text{List(A)(}\Gamma\text{)}}$$

$$\text{eq-E-List)}\ \frac{\begin{array}{c}\text{M(z) type}[\Gamma,\ \text{z} \in \text{List(A)}]\\ \text{t}_1 = \text{t}_2 \in \text{List(A)}[\Gamma]\\ \text{c}_1 = \text{c}_2 \in \text{M(nil)}[\Gamma]\end{array} \qquad \text{e}_1\text{(x,w,y)} = \text{e}_2\text{(x,w,y)} \in \text{M(cons(x,w))}[\Gamma,\ \text{x} \in \text{List(A)},\ \text{w} \in \text{A},\ \text{y} \in \text{M(x)}]}{\text{El}_{List}\text{(t}_1\text{, c}_1\text{, e}_1\text{)} = \text{El}_{List}\text{(t}_2\text{, c}_2\text{, e}_2\text{)} \in \text{M(t}_1\text{)}[\Gamma]}$$

A.3.5 Eliminatore dipendente

$$\text{E-List}_{dip}\text{)}\ \frac{\begin{array}{c}\text{M(z) type}[\Gamma,\ \text{z} \in \text{List(A)}]\\ \text{c} \in \text{M(nil)}[\Gamma]\end{array} \qquad \text{e(x,w,y)} \in \text{M(cons(x,w))}[\Gamma,\ \text{x} \in \text{List(A)},\ \text{w} \in \text{A},\ \text{y} \in \text{M(x)}]}{\text{El}_{List}\text{(z,c,e)} \in \text{M(z)}[\Gamma,\ \text{z} \in \text{List(A)}]}$$

A.4 Tipo della somma disgiunta

A.4.1 Regole di Formazione

$$\text{F-+)}\ \frac{\mathrm{B\ type\ [\Gamma]} \qquad \mathrm{C\ type\ [\Gamma]}}{\mathrm{B + C\ type\ [\Gamma]}}$$

A.4.2 Regole di Introduzione

$$\text{I}_1\text{-+)}\ \frac{\mathrm{b} \in \mathrm{B}[\Gamma] \qquad \mathrm{B + C\ type}[\Gamma]}{\mathrm{inl(b)} \in \mathrm{B + C}[\Gamma]} \qquad \text{I}_2\text{-+)}\ \frac{\mathrm{c} \in \mathrm{C}[\Gamma] \qquad \mathrm{B + C\ type}[\Gamma]}{\mathrm{inr(c)} \in \mathrm{B + C}[\Gamma]}$$

A.4.3 Regole di Eliminazione

$$\text{E-+)}\ \frac{\begin{array}{c}\mathrm{M(z)\ type}[\Gamma,\ \mathrm{z} \in \mathrm{B + C}] \\ \mathrm{t} \in \mathrm{B + C}[\Gamma]\end{array} \qquad \mathrm{e}_B(\mathrm{x}_1) \in \mathrm{M(inl(x_1))}[\Gamma,\ \mathrm{x}_1 \in \mathrm{B}] \qquad \mathrm{e}_C(\mathrm{x}_2) \in \mathrm{M(inr(x_2))}[\Gamma,\ \mathrm{x}_2 \in \mathrm{C}]}{\mathrm{El}_+(\mathrm{t,e}_B,\mathrm{e}_C) \in \mathrm{M(t)}[\Gamma]}$$

A.4.4 Regole di Conservazione

$$\text{C}_1\text{-+)}\ \frac{\begin{array}{c}\mathrm{M(z)\ type}[\Gamma,\ \mathrm{z} \in \mathrm{B + C}] \\ \mathrm{b} \in \mathrm{B}[\Gamma]\end{array} \qquad \mathrm{e}_B(\mathrm{x}_1) \in \mathrm{M(inl(x_1))}[\Gamma,\ \mathrm{x}_1 \in \mathrm{B}] \qquad \mathrm{e}_C(\mathrm{x}_2) \in \mathrm{M(inr(x_2))}[\Gamma,\ \mathrm{x}_2 \in \mathrm{C}]}{\mathrm{El}_+(\mathrm{inl(b)},\mathrm{e}_B,\mathrm{e}_C) = \mathrm{e}_B(\mathrm{b}) \in \mathrm{M(inl(b))}[\Gamma]}$$

$$\text{C}_2\text{-+)}\ \frac{\begin{array}{c}\mathrm{M(z)\ type}[\Gamma,\ \mathrm{z} \in \mathrm{B + C}] \\ \mathrm{c} \in \mathrm{C}[\Gamma]\end{array} \qquad \mathrm{e}_B(\mathrm{x}_1) \in \mathrm{M(inl(x_1))}[\Gamma,\ \mathrm{x}_1 \in \mathrm{B}] \qquad \mathrm{e}_C(\mathrm{x}_2) \in \mathrm{M(inr(x_2))}[\Gamma,\ \mathrm{x}_2 \in \mathrm{C}]}{\mathrm{El}_+(\mathrm{inr(c)},\mathrm{e}_B,\mathrm{e}_C) = \mathrm{e}_C(\mathrm{c}) \in \mathrm{M(inr(c))}[\Gamma]}$$

A.4.5 Regole di Uguaglianza

$$\text{eq-F-+)}\ \frac{\mathrm{B}_1 = \mathrm{B}_2\ \mathrm{type}[\Gamma] \qquad \mathrm{C}_1 = \mathrm{C}_2\ \mathrm{type}[\Gamma]}{\mathrm{B}_1 + \mathrm{C}_1 = \mathrm{B}_2 + \mathrm{C}_2\ \mathrm{type}(\Gamma)}$$

A.4.6 Eliminatore dipendente

$$\text{E}_{dip}\text{-+)}\ \frac{\mathrm{M(z)\ type}[\Gamma,\ \mathrm{z} \in \mathrm{B + C}] \qquad \mathrm{e}_B(\mathrm{x}_1) \in \mathrm{M(inl(x_1))}[\Gamma,\ \mathrm{x}_1 \in \mathrm{B}] \qquad \mathrm{e}_C(\mathrm{x}_2) \in \mathrm{M(inr(x_2))}[\Gamma,\ \mathrm{x}_2 \in \mathrm{C}]}{\mathrm{El}_+(\mathrm{z,e}_B,\mathrm{e}_C) \in \mathrm{M(z)}[\Gamma,\mathrm{z} \in \mathrm{B + C}]}$$

A.5 Tipo dell'uguaglianza proposizionale

A.5.1 Regole di Formazione

$$\text{F-Id)}\ \frac{\text{A type }[\Gamma] \qquad \text{a} \in \text{A}[\Gamma] \qquad \text{b} \in \text{A}[\Gamma]}{\text{Id(A,a,b) type}[\Gamma]}$$

A.5.2 Regole di Introduzione

$$\text{I-Id)}\ \frac{\text{a} \in \text{A}[\Gamma]}{\text{id(a)} \in \text{Id(A,a,a)}[\Gamma]}$$

A.5.3 Regole di Eliminazione

$$\text{E-Id)}\ \frac{\begin{array}{c}\text{M(z}_1\text{,z}_2\text{,z}_3\text{) type}[\Gamma,\ \text{z}_1 \in \text{A},\ \text{z}_2 \in \text{A},\ \text{z}_3 \in \text{Id(A, z}_1\text{, z}_2\text{)}] \\ \text{a} \in \text{A}[\Gamma] \\ \text{b} \in \text{A}[\Gamma] \\ \text{t} \in \text{Id(A,a,b)}[\Gamma]\end{array} \qquad \text{e(x)} \in \text{M(x,x,Id(x))}[\Gamma,\ \text{x} \in \text{A}]}{\text{El}_{Id}\text{(t, (x).e(x))} \in \text{M(a,b,t)}[\Gamma]}$$

A.5.4 Regole di Conservazione

$$\text{C-Id)}\ \frac{\begin{array}{c}\text{M(z}_1\text{,z}_2\text{,z}_3\text{) type}[\Gamma,\ \text{z}_1 \in \text{A},\ \text{z}_2 \in \text{A},\ \text{z}_3 \in \text{Id(A, z}_1\text{, z}_2\text{)}] \\ \text{a} \in \text{A}[\Gamma]\end{array} \qquad \text{e(x)} \in \text{M(x,x,Id(x))}[\Gamma,\ \text{x} \in \text{A}]}{\text{El}_{Id}\text{(Id(a), (x).e(x))} = \text{e(a)} \in \text{M(a,a,Id(a))}[\Gamma]}$$

A.5.5 Regole di Uguaglianza

$$\text{eq-F-Id)}\ \frac{\text{A}_1 = \text{A}_2\ \text{type}[\Gamma] \qquad \text{a}_1 = \text{a}_2 \in \text{A}[\Gamma] \qquad \text{b}_1 = \text{b}_2 \in \text{A}[\Gamma]}{\text{Id(A}_1\text{,a}_1\text{,b}_1\text{)} = \text{Id(A}_2\text{,a}_2\text{,b}_2\text{) type}[\Gamma]}$$

$$\text{eq-I-Id)}\ \frac{\text{a}_1 = \text{a}_2 \in \text{A}[\Gamma]}{\text{id(a}_1\text{)} = \text{id(a}_2\text{)} \in \text{Id(A,a}_1\text{,a}_1\text{)}[\Gamma]}$$

A.5.6 Eliminatore dipendente

$$\text{E-Id}_{dip}\text{)}\ \frac{\text{M(z}_1\text{,z}_2\text{,z}_3\text{) type}[\Gamma,\ \text{z}_1 \in \text{A},\ \text{z}_2 \in \text{A},\ \text{z}_3 \in \text{Id(A, z}_1\text{, z}_2\text{)}] \qquad \text{e(x)} \in \text{M(x,x,Id(x))}[\Gamma,\ \text{x} \in \text{A}]}{\text{El}_{Id}\text{(z}_3\text{, (x).e(x))} \in \text{M(z}_1\text{, z}_2\text{, z}_3\text{)}[\Gamma\text{,z}_1 \in \text{A},\ \text{z}_2 \in \text{A},\ \text{z}_3 \in \text{Id(A,z}_1\text{, z}_2\text{)}]}$$

A.6 Tipo somma indiciata forte

A.6.1 Regole di Formazione

$$\text{F-}\Sigma)\ \frac{\mathrm{C(x)\ type\ [\Gamma,\ x \in B]}}{\sum_{x\in B} \mathrm{C(x)\ type[\Gamma]}}$$

A.6.2 Regole di Introduzione

$$\text{I-}\Sigma)\ \frac{\mathrm{b \in B[\Gamma]} \qquad \mathrm{c \in C(b)[\Gamma]} \qquad \sum_{x\in B} \mathrm{C(x)\ type[\Gamma]}}{\mathrm{<b,c>} \in \sum_{x\in B} \mathrm{C(x)[\Gamma]}}$$

A.6.3 Regole di Eliminazione

$$\text{E-}\Sigma)\ \frac{\mathrm{M(z)\ type[\Gamma,\ z \in} \sum_{x\in B} \mathrm{C(x)]} \qquad \begin{array}{c} \mathrm{t} \in \sum_{x\in B} \mathrm{C(x)[\Gamma]} \\ \mathrm{e(x,y) \in M(<x,y>)[\Gamma,\ x \in B,\ y \in C(x)]} \end{array}}{\mathrm{El_{\Sigma}(t,(x,y).e(x,y)) \in M(t)[\Gamma]}}$$

A.6.4 Regole di Conservazione

$$\text{C-}\Sigma)\ \frac{\mathrm{M(z)\ type[\Gamma,\ z \in} \sum_{x\in B} \mathrm{C(x)]} \qquad \begin{array}{c} \mathrm{b \in B[\Gamma]} \\ \mathrm{c \in C(b)} \end{array} \qquad \mathrm{e(x,y) \in M(<x,y>)[\Gamma,\ x \in B,\ y \in C(x)]}}{\mathrm{El_{\Sigma}(<b,c>,e) = e(b,c) \in M(<b,c>)[\Gamma]}}$$

A.6.5 Regole di Uguaglianza

$$\text{eq-F-}\Sigma)\ \frac{\mathrm{B_1 = B_2\ type[\Gamma]} \qquad \mathrm{C_1(x) = C_2(x)\ type[\Gamma,\ x \in B_1]}}{\sum_{x\in B_1} \mathrm{C_1(x)} = \sum_{x\in B_2} \mathrm{C_2(x)\ type[\Gamma]}}$$

$$\text{eq-I-}\Sigma)\ \frac{\sum_{x\in B} \mathrm{C(x)\ type[\Gamma]} \qquad \mathrm{b_1 = b_2 \in B[\Gamma]} \qquad \mathrm{c_1 = c_2 \in C(b_1)[\Gamma]}}{\mathrm{<b_1,c_1> = <b_2,c_2>} \in \sum_{x\in B} \mathrm{C(x)[\Gamma]}}$$

$$\text{eq-E-}\Sigma)\ \frac{\mathrm{M(z)\ type[\Gamma,\ z \in} \sum_{x\in B} \mathrm{C(x)]} \qquad \begin{array}{c} \mathrm{t_1 = t_2} \in \sum_{x\in B} \mathrm{C(x)[\Gamma]} \\ \mathrm{e_1(x,y) = e_2(x,y) \in M(<x,y>)[\Gamma,\ x \in B,\ y \in C(x)]} \end{array}}{\mathrm{El_{\Sigma}(t_1,e_1) = El_{\Sigma}(t_2,e_2) \in M(t_1)[\Gamma]}}$$

A.6.6 Eliminatore dipendente

$$\text{E-}\Sigma_{dip})\ \frac{\mathrm{M(z)\ type[\Gamma,\ z \in} \sum_{x\in B} \mathrm{C(x)]} \qquad \mathrm{e(x,y) \in M(<x,y>)[\Gamma,\ x \in B,\ y \in C(x)]}}{\mathrm{El_{\Sigma}(z,(x,y).e(x,y)) \in M(z)[\Gamma,\ z \in} \sum_{x\in B} \mathrm{C(x)]}}$$

A.7 Tipo prodotto cartesiano

A.7.1 Regole di Formazione

$$\text{F-}\times)\ \frac{\text{B type } [\Gamma] \qquad \text{C type } [\Gamma]}{\text{B} \times \text{C type}[\Gamma]}$$

A.7.2 Regole di Introduzione

$$\text{I-}\times)\ \frac{\text{b} \in \text{B}[\Gamma] \qquad \text{c} \in \text{C}[\Gamma]}{<\text{b,c}> \in \text{B} \times \text{C}[\Gamma]}$$

A.7.3 Regole di Proiezione

$$\text{PJ}_1\text{-}\times)\ \frac{\text{d} \in \text{B} \times \text{C}[\Gamma]}{\pi_1\text{d} \in \text{B}[\Gamma]}$$

$$\text{PJ}_2\text{-}\times)\ \frac{\text{d} \in \text{B} \times \text{C}[\Gamma]}{\pi_2\text{d} \in \text{C}[\Gamma]}$$

A.7.4 Regole di Uguaglianza delle proiezioni

$$\text{PJ}_1\text{-eq})\ \frac{\text{b} \in \text{B}[\Gamma] \qquad \text{c} \in \text{C}[\Gamma]}{\pi_1(<\text{b,c}>) = \text{b} \in \text{B}[\Gamma]}$$

$$\text{PJ}_2\text{-eq})\ \frac{\text{b} \in \text{B}[\Gamma] \qquad \text{c} \in \text{C}[\Gamma]}{\pi_2(<\text{b,c}>) = \text{c} \in \text{C}[\Gamma]}$$

A.8 Tipo delle funzioni

A.8.1 Regole di Formazione

$$\text{F-}\!\to)\ \frac{\text{B type } [\Gamma] \qquad \text{C type } [\Gamma]}{\text{B} \to \text{C type}[\Gamma]}$$

A.8.2 Regole di Introduzione

$$\text{I-}\!\to)\ \frac{\text{c(x)} \in \text{C}[\Gamma,\text{x} \in \text{B}]}{\lambda\text{x}^{B}.\text{c(x)} \in \text{B} \to \text{C}[\Gamma]}$$

A.8.3 Regole di Eliminazione

$$\text{E-}\!\to)\ \frac{\text{f} \in \text{B} \to \text{C}[\Gamma] \qquad \text{b} \in \text{B}[\Gamma]}{\text{Ap(f,b)} \in \text{C}[\Gamma]}$$

A.8.4 Regole di Conservazione

$$\text{C-}\!\to)\ \frac{\text{c(x)} \in \text{C}[\Gamma,\ \text{x} \in \text{B}] \qquad \text{b} \in \text{B}[\Gamma]}{\text{Ap}(\lambda\text{x}^{B}.\text{c(x)},\text{b}] = \text{c(b)}\in \text{C}[\Gamma]}$$

A.8.5 Regole di Uguaglianza

$$\text{eq-E-}\!\to)\ \frac{\text{f}_1 = \text{f}_2 \in \text{B} \to \text{C}[\Gamma] \qquad \text{b}_1 = \text{b}_2 \in \text{B}[\Gamma]}{\text{Ap}(\text{f}_1,\text{b}_1) = \text{Ap}(\text{f}_2,\text{b}_2) \in \text{C}[\Gamma]}$$

$$\text{eq-I-}\!\to)\ \frac{\text{c}_1\text{(x)} = \text{c}_2\text{(x)} \in \text{C}[\Gamma,\text{x} \in \text{B}]}{\lambda\text{x}^{B}.\text{c}_1\text{(x)} = \lambda\text{x}^{B}.\text{c}_2\text{(x)} \in \text{B} \to \text{C}[\Gamma]}$$

A.9 Tipo prodotto dipendente

A.9.1 Regole di Formazione

$$\text{F-}\prod)\ \frac{\text{B type }[\Gamma] \qquad \text{C(x) type }[\Gamma,\text{x} \in \text{B}]}{\prod_{x\in B} \text{C(x) type}[\Gamma]}$$

A.9.2 Regole di Introduzione

$$\text{I-}\prod)\ \frac{\text{c(x)} \in \text{C(x)}[\Gamma,\text{x} \in \text{B}]}{\lambda \text{x}^B.\text{c(x)} \in \prod_{x\in B} \text{C(x)}[\Gamma]}$$

A.9.3 Regole di Eliminazione

$$\text{E-}\prod)\ \frac{\text{f} \in \prod_{x\in B} \text{C(x)}[\Gamma] \qquad \text{b} \in \text{B}[\Gamma]}{\text{Ap(f,b)} \in \text{C(b)}[\Gamma]}$$

A.9.4 Regole di Conservazione

$$\text{C-}\prod)\ \frac{\text{c(x)} \in \text{C(x)}[\Gamma,\ \text{x} \in \text{B}] \qquad \text{b} \in \text{B}[\Gamma]}{\text{Ap}(\lambda \text{x}^B.\text{c(x),b}] = \text{c(b)}\in \text{C(b)}[\Gamma]}$$

A.9.5 Regole di Uguaglianza

$$\text{eq-E-}\prod)\ \frac{\text{f}_1 = \text{f}_2 \in \prod_{x\in B} \text{C(x)}[\Gamma] \qquad \text{b}_1 = \text{b}_2 \in \text{B}[\Gamma]}{\text{Ap(f}_1\text{,b}_1) = \text{Ap(f}_2\text{,b}_2) \in \text{C(b}_1)[\Gamma]}$$

$$\text{eq-I-}\prod)\ \frac{\text{c}_1\text{(x)} = \text{c}_2\text{(x)} \in \text{C(x)}[\Gamma,\text{x} \in \text{B}]}{\lambda \text{x}^B.\text{c}_1\text{(x)} = \lambda \text{x}^B.\text{c}_2\text{(x)} \in \prod_{x\in B} \text{C(x)}[\Gamma]}$$

$$\text{eq-F-}\prod)\ \frac{\text{B}_1 = \text{B}_2 \text{ type}[\Gamma] \qquad \text{C}_1\text{(x)} = \text{C}_2\text{(x) type}[\Gamma,\text{x} \in \text{B}_1]}{\prod_{x\in B} \text{C}_1\text{(x)} = \prod_{x\in B} \text{C}_2\text{(x)}[\Gamma]}$$

A.10 Tipo vuoto

A.10.1 Regole di Formazione

$$\text{F-N}_0)\ \frac{\Gamma\ \text{cont}}{\text{N}_0\ \text{type}\ [\Gamma]}$$

A.10.2 Regole di Eliminazione

$$\text{E-N}_0)\ \frac{\text{t} \in \text{N}_0[\Gamma] \qquad \text{M(z) type}[\Gamma,\text{z} \in \text{N}_0]}{\text{El}_{N0}(\text{t}) \in \text{M(t)}[\Gamma]}$$

A.10.3 Regole di Uguaglianza

$$\text{eq-E-N}_0)\ \frac{\text{t}_1 = \text{t}_2 \in \text{N}_0[\Gamma] \qquad \text{M(z) type}[\Gamma,\text{z} \in \text{N}_0]}{\text{El}_{N0}(\text{t}_1) = \text{El}_{N0}(\text{t}_2) \in \text{M}(\text{t}_1)[\Gamma]}$$

A.10.4 Eliminatore dipendente

$$\text{E-N}_{0dip})\ \frac{\text{M(z) type}[\Gamma,\text{z} \in \text{N}_0]}{\text{El}_{N0}(\text{z}) \in \text{M(z)}[\Gamma,\text{z} \in \text{N}_0]}$$

A.11 Tipi Primo Universo

A.11.1 Regole di Formazione

$$\text{F-Uno)}\ \frac{\Gamma\ \text{cont}}{\text{U}_0\ \text{type}[\Gamma]}$$

A.11.2 Regole di Introduzione

$$\text{I}_1\text{-Uno)}\ \frac{\Gamma\ \text{cont}}{\hat{N}_0 \in \text{U}_0[\Gamma]} \qquad \text{I}_2\text{-Uno)}\ \frac{\Gamma\ \text{cont}}{\hat{N}_1 \in \text{U}_0[\Gamma]} \qquad \text{I}_3\text{-Uno)}\ \frac{\Gamma\ \text{cont}}{\hat{Nat} \in \text{U}_0[\Gamma]}$$

$$\text{I}_4\text{-Uno)}\ \frac{\text{c(x)} \in \text{U}_0[\Gamma,\text{x} \in \mathbf{T}(\text{b})] \qquad \text{b} \in \text{U}_0[\Gamma]}{\hat{\sum}_{x\in b}\ \text{c(x)} \in \text{U}_0[\Gamma]}$$

$$\text{I}_5\text{-Uno)}\ \frac{\text{c(x)} \in \text{U}_0[\Gamma,\text{x} \in \mathbf{T}(\text{b})] \qquad \text{b} \in \text{U}_0[\Gamma]}{\hat{\prod}_{x\in b}\ \text{c(x)} \in \text{U}_0[\Gamma]}$$

$$\text{I}_6\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma] \qquad \text{c} \in \text{U}_0[\Gamma]}{\text{b} \hat{+} \text{c} \in \text{U}_0[\Gamma]} \qquad \text{I}_7\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma]}{\hat{List}(b) \in \text{U}_0[\Gamma]}$$

$$\text{I}_8\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma] \qquad \text{c} \in \mathbf{T}(\text{b})[\Gamma] \qquad \text{d} \in \mathbf{T}(\text{b})[\Gamma]}{\hat{Id}(\text{b,c,d}) \in \text{U}_0[\Gamma]}$$

A.11.3 Regole di Eliminazione

$$\text{E-Uno)}\ \frac{\text{a} \in \text{U}_0[\Gamma]}{\mathbf{T}(\text{a})\ \text{type}[\Gamma]}$$

A.11.4 Regole di Conversione

$$\text{C}_1\text{-Uno)}\ \frac{\Gamma\ \text{cont}}{\mathbf{T}(\hat{N}_0) = \text{N}_0\ \text{type}[\Gamma]} \qquad \text{C}_2\text{-Uno)}\ \frac{\Gamma\ \text{cont}}{\mathbf{T}(\hat{N}_1) = \text{N}_1\ \text{type}[\Gamma]}$$

$$\text{C}_3\text{-Uno)}\ \frac{\Gamma\ \text{cont}}{\mathbf{T}((\hat{Nat})\ ) = \text{Nat type}[\Gamma]}$$

$$\text{C}_4\text{-Uno)}\ \frac{\text{c(x)} \in \text{U}_0[\Gamma,\text{x} \in \mathbf{T}(\text{b})] \qquad \text{b} \in \text{U}_0[\Gamma]}{\mathbf{T}(\hat{\sum}_{x\in b}c(x)) = \sum_{x\in\mathbf{T}(b)}\text{T(c(x)) type}[\Gamma]}$$

$$\text{C}_5\text{-Uno)}\ \frac{\text{c(x)} \in \text{U}_0[\Gamma,\text{x} \in \mathbf{T}(\text{b})] \qquad \text{b} \in \text{U}_0[\Gamma]}{\mathbf{T}(\hat{\prod}_{x\in b}c(x)) = \prod_{x\in\mathbf{T}(b)}\text{T(c(x)) type}[\Gamma]}$$

$$\text{C}_6\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma] \qquad \text{c} \in \text{U}_0[\Gamma]}{\mathbf{T}(\text{b} \hat{+} \text{c}) = \mathbf{T}(\text{b}) + \mathbf{T}(\text{c})\ \text{type}[\Gamma]} \qquad \text{C}_7\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma]}{\mathbf{T}(\hat{List}(\text{b})) = \text{List}(\mathbf{T}(\text{b}))\ \text{type}[\Gamma]}$$

$$\text{C}_8\text{-Uno)}\ \frac{\text{b} \in \text{U}_0[\Gamma] \qquad \text{c} \in \mathbf{T}(\text{b})[\Gamma] \qquad \text{d} \in \mathbf{T}(\text{b})[\Gamma]}{\mathbf{T}(\hat{Id}(\text{b,c,d})) = \text{Id}(\mathbf{T}(\text{b}),\text{c,d})\ \text{type}[\Gamma]}$$

A.11.5 Regole di Uguaglianza

$$\text{eq-I}_4\text{-Uno)}\ \frac{\text{c}_1(\text{x}) = \text{c}_2(\text{x}) \in \text{U}_0[\Gamma,\text{x} \in \mathbf{T}(\text{b})] \qquad \text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma]}{\hat{\sum}_{x \in b_1} \text{c}_1(\text{x}) = \hat{\sum}_{x \in b_2} \text{c}_2(\text{x}) \in \text{U}_0[\Gamma]}$$

$$\text{eq-I}_5\text{-Uno)}\ \frac{\text{c}_1(\text{x}) = \text{c}_2(\text{x}) \in \text{U}_0[\Gamma,\text{x} \in \mathbf{T}(\text{b})] \qquad \text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma]}{\hat{\prod}_{x \in b_1} \text{c}_1(\text{x}) = \hat{\prod}_{x \in b_2} \text{c}_2(\text{x}) \in \text{U}_0[\Gamma]}$$

$$\text{eq-I}_6\text{-Uno)}\ \frac{\text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma] \qquad \text{c}_1 = \text{c}_2 \in \text{U}_0[\Gamma]}{\text{b}_1 \mathbin{\hat{+}} \text{c}_1 = \text{b}_2 \mathbin{\hat{+}} \text{c}_2 \in \text{U}_0[\Gamma]}$$

$$\text{eq-I}_7\text{-Uno)}\ \frac{\text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma]}{L\hat{i}st(\text{b}_1) = L\hat{i}st(\text{b}_2) \in \text{U}_0[\Gamma]}$$

$$\text{eq-I}_8\text{-Uno)}\ \frac{\text{b}_1 = \text{b}_2 \in \text{U}_0[\Gamma] \qquad \text{c}_1 = \text{c}_2 \in \mathbf{T}(\text{b})[\Gamma] \qquad \text{d}_1 = \text{d}_2 \in \mathbf{T}(\text{b})[\Gamma]}{\hat{Id}(\text{b}_1,\text{c}_1,\text{d}_1) = \hat{Id}(\text{b}_2,\text{c}_2,\text{d}_2) \in \text{U}_0[\Gamma]}$$