Linguaggi di programmazione

Numeri naturali

In base a chi lo si chiede si avranno risposte diverse sul dare una definizione di numeri naturali

es

$3=\{\overbrace{\{\}}^{0},\overbrace{\{\{\}\}}^{1},\overbrace{\{\{\},\{\{\}\}\}}^{2}\}$ Von Neumann → definizione insiemistica

$3=\lambda xy\cdot x(x(xy))$ Alonzo Church → definizione ricorsiva

Immaginiamo di parlare con un marziano e cerchiamo gli di dare una definizione di successore nei numeri naturali (lui tenterà di smontarla, quindi bisogna fare attenzione) secondo gli assiomi di Peano

1. $0\in\mathbb{N}$
2. $n\in\mathbb{N}\to succ(n)\in\mathbb{N}$
3. $\nexists n$ t.c. $0=succ(n)$
4. $\forall n,m\ succ(n)=succ(m)\Rightarrow n=m$
5. $\forall S\subseteq\mathbb{N}\wedge\begin{cases}0\in S\\ n\in S\Rightarrow succ(n)\in S\end{cases}\Rightarrow S=\mathbb{N}$

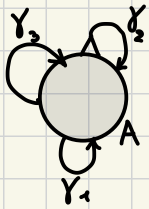
una proprietà e il sottoinsieme corrispondente sono in biezione (dal sottoinsieme posso ricavare la proprietà originale)

induzione

$$\forall P\ \ \frac{P(0)\quad P(n)\to P(n+1)}{\forall n\ \ P(n)}$$

Definire un'algebra

(A,Γ) dove $\Gamma=\{\gamma_1,\gamma_2,\gamma_3,\dots\}$



Ma posso anche definire operazioni che prendono elementi dall'esterno

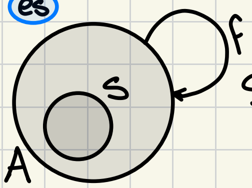
Chiusura di un'algebra

Sia $f: A^n\times K\to A$ una operazione su A con parametri esterni in $K=K_1\times\dots\times K_m$.

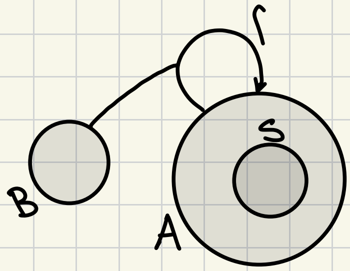
Un insieme $S\subseteq A$ si dice chiuso rispetto a f quando $a_1,\dots,a\in S\to f(a_1,\dots,a_n,K_1,\dots,K_m)\in S$

Ovvero è chiuso se per ogni input possibile che pesco da A e che proviene da S, questo rimane in f (quelli che provengono da A\S o dall'esterno non ci interessano

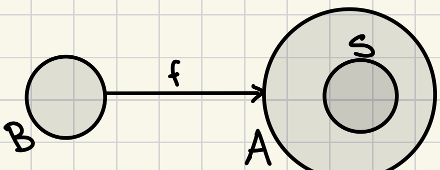
es



S è chiusa rispetto a f se $\forall x\in S\to f(x)\in S$



S è chiuso rispetto a S se $\forall x\in B, y\in A\to f(x,y)\in S$



S è chiuso rispetto a f se $\forall x\in B\to f(x)\in S$

applicando la def vedremo che, visto che possiamo usare qualsiasi elemento di S, la seconda parte dell'implicazione (visto che prende input solo da B) è vera se per qualsiasi input di f finisce in S

Algebra induttiva

Un'algebra (A,Γ) si dice induttiva quando:

- tutte le γ_i sono iniettive
- tutte le γ_i sono disgiunte ← un elemento non può essere immagine di due funzioni differenti
- $\forall S\subseteq A$ se S è chiuso rispetto a tutte le γ_i allora $S=A$ → principio di induzione
  ↳ si dice che A è la più piccola algebra contenuta in A

es $\mathbb{N}$

in $\mathbb{N}$ però abbiamo parlato dell'elemento 0 (ora non contemplato), definiamo quindi la seguente algebra

$(\mathbb{N}, \{\emptyset, succ\})$ dove $\mathbb{1} \xrightarrow{\emptyset} \mathbb{N}$, $\mathbb{N} \xrightarrow{succ} \mathbb{N}$

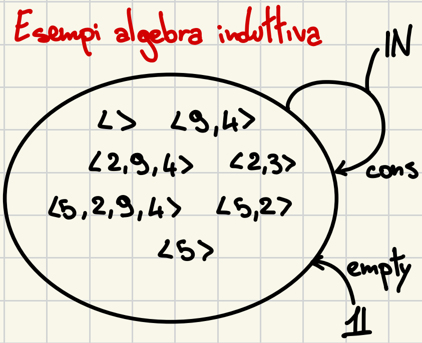
funzione che associa un insieme di un solo elemento all'elemento 0 in $\mathbb{N}$

Notiamo quindi che:

- tutte le γ_i sono iniettive 4.
- tutte le imagini di γ_i sono disgiunte 3. ($\emptyset$ punta a 0, mentre succ a tutti gli altri)
- $\mathbb{N}$ è chiuso rispetto a succ
- $\mathbb{N}$ è chiuso rispetto a $\emptyset$

Il costruire un'algebra induttiva mi garantisce che non esistono sottoalgebre in essa contenute. Inoltre tutte le algebre induttive sono isomorfismi

Esempi algebra induttiva

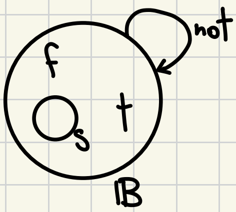


cons(2,⟨9,4⟩) = ⟨2,9,4⟩ ← con append non sarebbe stata induttiva, non disgiunta

L'insieme che contiene le liste finite è induttivo, mentre quello che contiene quelle finite e infinite poiché contiene l'algebra delle liste finite

↳ l'insieme vuoto non è chiuso rispetto a nessuna nullaria perché applicandola su nessun input, restituisce ⟨⟩, non contenuto in $\emptyset$. L'operazione produce direttamente un output, non si può dire che la prima parte dell'implicazione della def di chiusura è falsa, rimane solo da vedere se l'output è S

esistono algebre induttive senza operazioni nullarie? exp.



L'insieme $S = \emptyset$ é chiuso poiché non ci sono elementi in S su cui poter applicare la funzione, la prima parte dell'implicazio della definizione è sempre falsa (implicazione vera)

Chiuso ristetto a not
con un costruttore che pesca solo da fuori non lo era

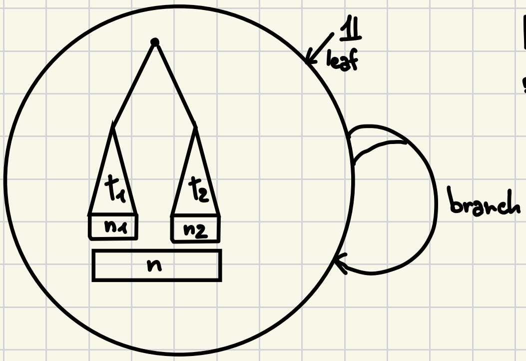
02/10

TEOREMA

Un albero binario con n foglie ha 2n-1 nodi

dim

Per dimostrarlo dovremo usare l'induzione strutturale



Nell'induzione strutturale assumiamo che la proprietà sia valida per t_1 e t_2

$$n = n_1 + n_2$$

radice

$$(2n_1-1)+(2n_2-1)+1 = 2\underbrace{(n_1+n_2)}_{n}-1$$

Espressione

link

Definiamo un linguaggio L come insieme di stringhe. Per descrivere la sintassi di linguaggi formali, usiamo la seguente sintassi

<simbolo> ::= <espressione>

es

$$M, N ::= 5 \mid 7 \mid M+N \mid M * N$$

assegnazione "secca"

dove se M e N sono espressioni è possibile costruire un'ulteriore espressione unendoli tramite "+" o "*"

Definiamo quindi una funzione eval: Exp → ℕ

es

elemento di Exp

eval(5) = 5 ← elemento di ℕ

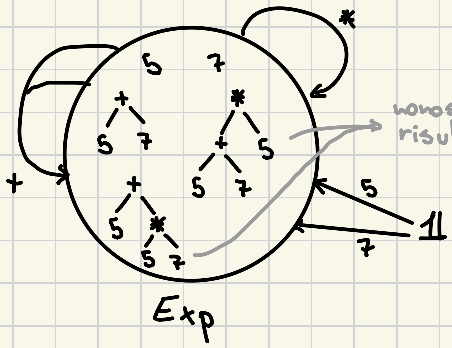
eval(7) = 7

eval(M+N) = eval(M) + eval(N)

eval(M*N) = eval(M) * eval(N)

eval(5+7*5) = ?

infatti in base a come considero l'espressione posso avere risultati diversi, dobbiamo quindi disambiguarla



nonostante scrivendolo possa sembrare che il risultato sia identico, in realtà sono due oggetti diversi

sicuramente non si tratta di una grammatica induttiva, ma ho usato l'induzione per definirla

Equazione ricorsive di domini

es

$\mathbb{N} \equiv \mathbb{1} + \mathbb{N}$

$L \equiv \mathbb{N} \cup (\mathbb{N} \times L)$

insieme di liste; isomorfo a

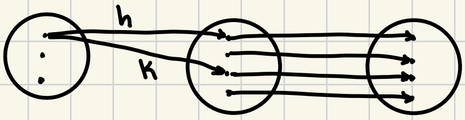
Brief sulla teoria delle categorie

Nella teoria delle categorie non si può usare la parola "elemento"

Come faccio quindi a descrivere un morfismo (funzione) f iniettivo (monomorfismo) non potendo parlare di elementi

RIS $h \circ f = K \circ f \Rightarrow h = K$

$$A \underset{K}{\overset{h}{\rightrightarrows}} B \xrightarrow{f} C$$



essendo f iniettiva ad ogni elemento viene associato un elemento diverso del codominio, dunque nella composizione $h \circ f$ e $K \circ f$ risulta che gli elementi di destinazione sono esattamente gli stessi allora $h = K$

Segnatura di un'algebra

Due algebre (A, Γ_A), (B, Γ_B) hanno la stessa segnatura se $\forall \gamma_A \in \Gamma_A \ \exists \gamma_B \in \Gamma_B$ con stessa arietà e se, sostituendo tutte le B in γ_B con A ottengo γ_A

es

<u>algebra D</u>

$f_D : A \times D \to D$

$g_D : \mathbb{1} \longrightarrow D$

$h_D : A \times B \times D \longrightarrow D$

<u>algebra di C</u>

$f_C : A \times C \longrightarrow C$

$g_C : \mathbb{1} \longrightarrow C$

$h_C : A \times B \times C \longrightarrow C$

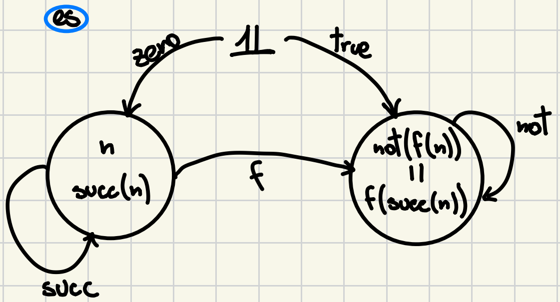
D e C hanno la stessa segnatura

Omomorfismo

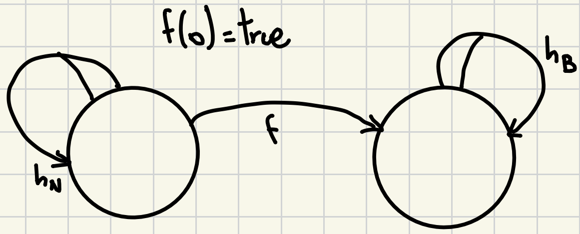
Due algebre con la stessa segnatura (A, Γ) e $(B, \Delta = \{\delta_1, \dots, \delta_K\})$, un omomorfismo è una funzione $A \to B$ tale che:

$$\forall i \quad f(\gamma_i(a_1, \dots, a_K)) = \delta_i(f(a_1), \dots, f(a_K))$$ ← rispetta le operazione

es



$f(0) = true$



$$f(h_N(m,n)) = h_B(f(m), f(n))$$

Se applico l'operazione da una parte o dall'altra il risultato non cambia, e lo devo fare per tutte le operazioni dell'algebra

TEOREMA

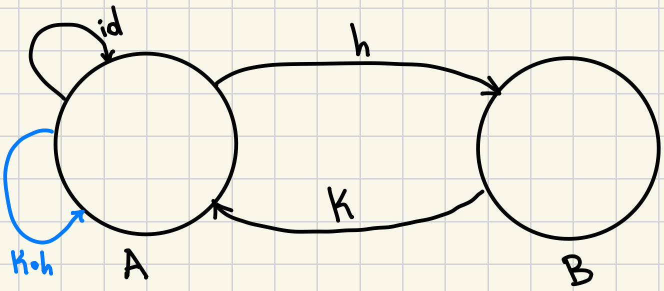
Prese due algebre induttive che hanno la stessa segnatura allora esiste un unico omomorfismo

<u>DEF</u> Isomorfismo

Un isomorfismo è un omomorfismo in cui la funzione è una biezione

LEMMA di Lambek

Avendo due algebre induttive esistono due omomorfismi (nelle due direzioni) e componendoli otteniamo un ulteriore omomorfismo



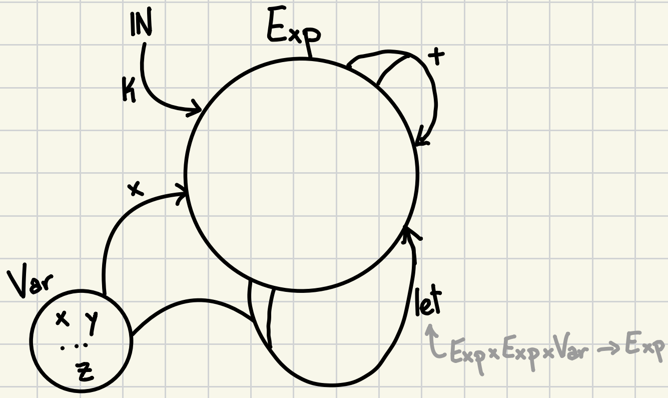
componendo h e K ottengo dunque, per quanto detto, un ulteriore omomorfismo, ma in A giá esiste un omomorfismo ovvero l'identitá. Per il teorema precedente peró puó esistere un solo omomorfismo dunque $id = K \circ h$
da ció si deduce che K e h sono inversi per cui la loro composizione é chiaramente biettiva, il che ci fa concludere che $K \circ h$ é un isomorfismo

Linguaggio Exp   link

es

$M, N, L ::= K \mid x \mid M+N \mid \text{let } x = M \text{ in } N$ ← MLStandard

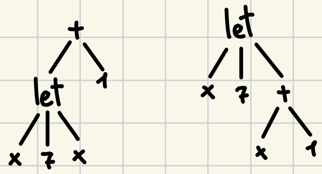
dove:
- K → costante
- x → variabile ∈Var
- let x = M in N → assegna a x il valore M all'interno di N



Vediamo alcuni esempi di uso:

let x=3 in x+x+2 ⇒ 8
let x=1+2 in 4 ⇒ 4
let x=let y=3 in y+1 in x+y ⇒ 7
let x=7 in x+1 ⇒ ??

quest'ultima espressione puó creare ambiguitá dato che puó essere doppiamente interpretata



DEF Variabili libere e legate

Una variabile x si dice libera si dice libera se data un'exp let x=M in N, x non compare in N
Si dice invece legata se invece compare in N

es

let x=5 in x+y   (x: legata, y: libera)

DEF free

Definiamo la funzione free: $Exp \longrightarrow \wp(Var)$ che restituisce l'insieme delle variabili libere di un'espressione

es

$free(\text{let } x=7 \text{ in } x+y) = \{y\}$
$free(K) = \emptyset$
$free(x) = \{x\}$
$free(M+N) = free(N) \cup free(M)$
$free(\text{let } x=M \text{ in } N) = free(M) \cup (free(N) - x)$ ← $free(\text{let } x=x \text{ in } x) = x$

Semantica operazionale

09/10

La semantica operazionale permette di rispondere alla domanda quanto fa, permettendoci di eliminare le ambiguità viste finora

Definisco quindi un'operazione per calcolare il valore, ma dipende anche dall'**ambiente** che si sta considerando

associazione tra variabile e valore

$M \in Exp \times \overset{E\in}{Env} \rightsquigarrow \overset{v\in}{Val}$ dove $\rightsquigarrow \subseteq Exp \times Exp \times Val$

ambiente in cui valutare Exp

Poiché ci sono casi in cui non viene restituito alcun risultato, si tratta quindi di una **relazione** e non di una funzione

Mi sto quindi chiedendo se $(M, E, v) \in \rightsquigarrow$

in questo ambiente

questa espressione

ha questo valore

es

$\left(\text{let } x=3 \text{ in } x+1, \begin{bmatrix} x \mapsto 7 \\ 7 \mapsto 12 \end{bmatrix}, 10\right) \in \rightsquigarrow$

$(5, [E], 7) \notin \rightsquigarrow$

Passiamo ora a una notazione più snella

deduzione

$\underbrace{E \vdash M \rightsquigarrow v}_{\text{sequente}}$ che si legge "M deduce E con valore v"

deduzione: $A, A \longrightarrow B \vdash B$

piove

se piove porto l'ombrello

deduco che ho l'ombrello

conseguenza logica $\neq$ deduzione

Ambiente

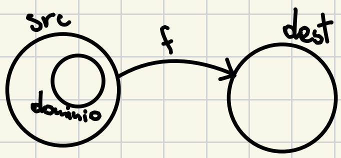
L'ambiente è una funzione parziale a dominio finito da variabile a valore

funzione non definita per alcuni elementi del dominio

$E: Var \xrightarrow{fin} Val$

dominio finito

Scriviamo gli ambienti come un insieme di coppie



$f: src \longrightarrow dest$

$$(E_1 E_2)(x) = \begin{cases} E_2(x) & \text{se } x \in dom(E_2) \\ E_1(x) & \text{altrimenti} \end{cases}$$

es

$E_1 = \{(z,3),(y,9)\}$

$E_2 = \{(z,4)\}$

$(E_1E_2)(y) = 9$

$(E_1E_2)(z) = 4$

Regola di inferenza eager

① $E \vdash k \leadsto k$ const

② $E \vdash x \leadsto E(x)$ x non esiste non esiste nessuna tripla var

↳ quanto fa x nell'ambiente E?

③
$$\frac{E \vdash M \leadsto v_1 \qquad E \vdash N \leadsto v_2}{E \vdash M+N \leadsto v} \; (\text{dove } v = v_1 + v_2)$$
condizioni laterali

conclusione

plus

④
valuta M con E e mettilo in v — valuta N con E insieme a (x,v) e mettilo in v'
$$\frac{E \vdash M \leadsto v \qquad E\{(x,v)\} \vdash N \leadsto v'}{E \vdash \text{let } x=M \text{ in } N \leadsto v'}$$
let

es $\varnothing \vdash \text{let } x=3 \text{ in } ((\text{let } x=(\text{let } y=2 \text{ in } x+y) \text{ in } x+7)+x)$

adesso applichiamo le regole di inferenza dall'esterno all'interno

per prima uso quella del let, spostando la parentesi sarebbe stata quella del +

$$\dfrac{\varnothing \vdash 3 \leadsto 3 \qquad \dfrac{\dfrac{\dfrac{(x,3) \vdash 2 \leadsto 2 \qquad \dfrac{(x,3)(y,2) \vdash x \leadsto 3 \qquad (x,3)(y,2) \vdash y \leadsto 2}{(x,3)(y,2) \vdash x+y \leadsto 5}\,\text{plus}}{(x,3) \vdash \text{let } y=2 \text{ in } x+y \leadsto 5}\,\text{let} \qquad (x,3)(x,5) \vdash x+7 \leadsto 12}{(x,3) \vdash \text{let } x=(\text{let } y=2 \text{ in } x+y) \text{ in } x+7 \leadsto 12}\,\text{let} \qquad (x,3) \vdash x \leadsto 3}{\{(x,3)\} \vdash (\text{let } x=(\text{let } y=2 \text{ in } x+y) \text{ in } x+7)+x \leadsto 15}\,\text{plus}}{\varnothing \vdash \text{let } x=\underbrace{3}_{M} \text{ in } \underbrace{((\text{let } x=(\text{let } y=2 \text{ in } x+y) \text{ in } x+7)+x)}_{N} \leadsto 15}\,\text{let}$$

L'approccio eager però ha uno svantaggio, infatti in un caso del tipo let x=  in 7 viene valutato prima il "mostro" nonostante in questo caso non sia assolutamente necessario

Come soluzione viene usato l'approccio lazy

Semantica lazy-dinamica

Rispetto alle regole di inferenza eager cambiano quella del let e quella della variabile

①
$$\frac{E \vdash M \leadsto v}{E \vdash x \leadsto v} \; (\text{se } Ex = M)$$
var

oss $\varnothing \vdash \text{let } x=y \text{ in let } y=7 \text{ in } x$ in lazy dinamico fa 7 ma dovrebbe essere non definito

②
$$\frac{E(x,M) \vdash N \leadsto v}{E \vdash \text{let } x=M \text{ in } N \leadsto v}$$
let

es let x=2 in let y=x+1 in let x=7 in y

eager

$$\dfrac{\varnothing \vdash 2 \leadsto 2 \qquad \dfrac{\dfrac{(x,2) \vdash x \leadsto 2 \qquad (x,2) \vdash 1 \leadsto 1}{(x,2) \vdash x+1 \leadsto 3} \qquad \dfrac{(x,2)(y,3) \vdash 7 \leadsto 7 \qquad (x,2)(y,3)(x,7) \vdash y \leadsto 3}{(x,2)(y,3) \vdash \text{let } x=7 \text{ in } y \leadsto 3}}{(x,2) \vdash \text{let } y=x+1 \text{ in let } x=7 \text{ in } y \leadsto 3}}{\varnothing \vdash \text{let } x=2 \text{ in let } y=x+1 \text{ in let } x=7 \text{ in } y \leadsto 3}$$

lazy-dinamico

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(x,2)(y,x+1)(x,7) \vdash x \leadsto 7 \qquad (x,2)(y,x+1)(x,7) \vdash 1 \leadsto 1}{(x,2)(y,x+1)(x,7) \vdash x+1 \leadsto 8}}{(x,2)(y,x+1)(x,7) \vdash y \leadsto 8}}{(x,2)(y,x+1) \vdash \text{let } x=7 \text{ in } y \leadsto 8}}{\varnothing(x,2) \vdash \text{let } y=x+1 \text{ in let } x=7 \text{ in } y \leadsto 8}}{\varnothing \vdash \text{let } x=2 \text{ in let } y=x+1 \text{ in let } x=7 \text{ in } y \leadsto 8}$$

diverso

Introducendo il lazy ci siamo esposti al problema dello scoping (in questo caso è dinamico). La semantica lazy-dinamica è considerata "sbagliata" poichè ci dà un risultato diverso dalle altre 3 combinazioni di metodologie

Semantica lazy-statica

Rispetto alle regole di inferenza eager cambiano quella del let e quella della variabile

①

$$\frac{E \vdash M \leadsto v}{E' \vdash x \leadsto v} \ (\text{se } E'(x) = (M, E)) \qquad \textbf{var}$$

②

$$\frac{E(x, M, E) \vdash N \leadsto v}{E \vdash \text{let } x = M \text{ in } N \leadsto v} \qquad \textbf{let} \qquad E : Var \xrightarrow{fin} Exp \times Env$$

oss in questo caso let x= [illegible] in x+x+x+x+x la semantica lazy è molto lenta (devo ricalcolare ogni volta x)

es

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\emptyset \vdash 2 \leadsto 2}{(x,2,\emptyset) \vdash x \leadsto 2} \quad (x,2,\emptyset) \vdash 1 \leadsto 1}{(x,2,\emptyset) \vdash x+1 \leadsto 3}}{E(x,7,E) \vdash y \leadsto 3}}{\underbrace{(x,2,\emptyset)(y, x+1, (x,2,\emptyset))}_{E} \text{ let } x=7 \text{ in } y \leadsto 3}}{(x,2,\emptyset) \vdash \text{let } y = x+1 \text{ in let } x=7 \text{ in } y \leadsto 3}}{\emptyset \vdash \text{let } x=2 \text{ in let } y=x+1 \text{ in let } x=7 \text{ in } y \leadsto 3}$$

TEOREMA

Semantica eager e lazy-statica sono equivalenti

	statico	dinamico
lazy	O	X
eager	O	O

il linguaggio Exp è troppo semplice per poter cogliere la differenza tra eager statico e dinamico

Linguaggio Fun

$M,N ::= 5 \mid x \mid M+N \mid \text{let } x=M \text{ in } N \mid \text{fn } x \Rightarrow M \mid MN$

↳ funzione M applicata a argomento N

es

$\underbrace{(\text{fn } x \Rightarrow x+1)}_{M}\underbrace{5}_{N} \rightsquigarrow 6$

$(\text{fn } x \Rightarrow x+1)(\text{fn } z \Rightarrow z)$

restituisce la funzione alla quale se passo in input una funzione restituisce una funzione

$[(\text{fn } x \Rightarrow [\text{fn } y \Rightarrow \underset{z \Rightarrow z+1}{y}\underset{53}{(x+1)}])\, 52](\text{fn } z \Rightarrow z+1) \rightsquigarrow 54$

si possono non solo avere valori in output ma anche intere funzioni

$(\text{fn } z \Rightarrow z)(\text{fn } x \Rightarrow x7) \rightsquigarrow \text{fn } x \Rightarrow x7$

oss esistono termini che non hanno una semantica (legali da scrivere ma che non danno alcun risultato) come ad esempio $5(\text{fn } x \Rightarrow x+1)$

oss Fun non é nē commutativo né associativo

Semantica eager dinamica

Inoltre a differenza di Exp, $Env: var \xrightarrow{fin} val$ dove $val = \{5, 42, \dots\} \cup (var \times Fun)$

$$E \vdash \text{fn } x \Rightarrow M \rightsquigarrow (x, M)$$

$\text{fn } x \Rightarrow x+1 \rightarrow \overbrace{(x, x+1)}^{\text{chiusura}}$

$$\frac{E \vdash M \rightsquigarrow \overset{\text{chiusura}}{(x, M')} \qquad E \vdash N \rightsquigarrow v \qquad E(x,v) \vdash M' \rightsquigarrow v'}{E \vdash MN \rightsquigarrow v'}$$

es

$$\dfrac{\emptyset \vdash \text{fn } x \Rightarrow x+1 \rightsquigarrow (x, x+1) \qquad \dfrac{(z,(x,x+1)) \vdash z \rightsquigarrow (x,x+1) \qquad \dfrac{E \vdash \text{fn } y \Rightarrow y5 \rightsquigarrow (y, y5) \qquad E \vdash \text{fn } z \Rightarrow 3 \rightsquigarrow (z,3) \qquad \dfrac{E' \vdash y \rightsquigarrow (z,3) \qquad E' \vdash 5 \rightsquigarrow 5 \qquad E'(z,5) \vdash 3 \rightsquigarrow 3}{\overbrace{E(y,(z,3))}^{E'} \vdash y5 \rightsquigarrow 3}}{\overbrace{(z,(x,x+1))}^{E} \vdash (\text{fn } y \Rightarrow y5)(\text{fn } z \Rightarrow 3) \rightsquigarrow 3} \qquad \dfrac{E(x,3) \vdash x \rightsquigarrow 3 \qquad E(x,3) \vdash 1 \rightsquigarrow 1}{E(x,3) \vdash x+1 \rightsquigarrow 4}}{(z,(x,x+1)) \vdash z[(\text{fn } y \Rightarrow y5)(\text{fn } z \Rightarrow 3)] \rightsquigarrow 4}}{\emptyset \vdash \text{let } z=(\text{fn } x \Rightarrow x+1) \text{ in } z[(\text{fn } y \Rightarrow y5)(\text{fn } z \Rightarrow 3)] \rightsquigarrow 4}$$

$$\dfrac{\emptyset \vdash 3 \rightsquigarrow 3 \qquad \dfrac{(x,3) \vdash \text{fn } z \Rightarrow z+x \rightsquigarrow (z, z+x) \qquad \dfrac{E \vdash 7 \rightsquigarrow 7 \qquad \dfrac{E' \vdash y \rightsquigarrow (z, z+x) \qquad E' \vdash 5 \rightsquigarrow 5 \qquad \dfrac{E'(z,5) \vdash z \rightsquigarrow 5 \qquad E'(z,5) \vdash x \rightsquigarrow 7}{E'(z,5) \vdash z+x \rightsquigarrow 12}}{\overbrace{E(x,7)}^{E'} \vdash y5 \rightsquigarrow 12}}{\overbrace{(x,3)(y,(z,z+x))}^{E} \vdash \text{let } x=7 \text{ in } y5 \rightsquigarrow 12}}{(x,3) \vdash \text{let } y=(\text{fn } z \Rightarrow z+x) \text{ in let } x=7 \text{ in } y5 \rightsquigarrow 12}}{\emptyset \vdash \text{let } x=3 \text{ in let } y=(\text{fn } z \Rightarrow z+x) \text{ in let } x=7 \text{ in } y5 \rightsquigarrow 12}$$

Semantica eager statica

$$E \vdash \text{fn } x \Rightarrow M \rightsquigarrow (x, M, E)$$

$$\frac{E \vdash M \rightsquigarrow (x, M', E') \qquad E \vdash N \rightsquigarrow v \qquad E'(x,v) \vdash M' \rightsquigarrow v'}{E \vdash MN \rightsquigarrow v'}$$

es

$$\dfrac{\emptyset \vdash 3 \leadsto 3 \qquad \dfrac{(x,3) \vdash \text{fn } z \Rightarrow z+x \leadsto (z, z+x, (x,3)) \qquad \dfrac{E \vdash 7 \leadsto 7 \qquad \dfrac{E' \vdash y \leadsto (z, z+x, (x,3)) \qquad E' \vdash 5 \leadsto 5 \qquad \dfrac{(x,3)(z,5) \vdash z \leadsto 5 \qquad (x,3)(z,7) \vdash x \leadsto 3}{(x,3)(z,5) \vdash z+x \leadsto 8}}{\overbrace{E(x,7)}^{E'} \vdash y5 \leadsto 8}}{\overbrace{(x,3)(y,(z,z+x),(x,3))}^{E} \vdash \text{let } x-7 \text{ in } y5 \leadsto 8}}{(x,3) \vdash \text{let } y=((\text{fn } z \Rightarrow z+x) \text{ in let } x=7 \text{ in } y5 \leadsto 8}}{\emptyset \vdash \text{let } x=3 \text{ in let } y=(\text{fn } z \Rightarrow z+x) \text{ in let } x=7 \text{ in } y5 \leadsto 8}$$

Espressione ω

L'espressione ω è una espressione non valutabile in alcuna semantica del linguaggio Fun

non serve che riscrivo tanto è lo stesso ambiente

$$\dfrac{\emptyset \vdash \text{fn } x \Rightarrow xx \rightsquigarrow (x,xx) \qquad \emptyset \vdash \text{fn } x \Rightarrow xx \rightsquigarrow (x,xx) \qquad \dfrac{(x,(x,xx)) \vdash x \rightsquigarrow (x,xx) \quad (x,(x,xx)) \vdash x \rightsquigarrow (x,xx) \quad (x,(x,xx)) \vdash xx \rightsquigarrow \omega}{(x,(x,xx)) \vdash xx \rightsquigarrow \omega}}{\emptyset \vdash (\text{fn } x \Rightarrow xx)(\text{fn } x \Rightarrow xx) \rightsquigarrow \omega}$$

Non è valutabile poiché, come possiamo vedere, va in loop. Però espressioni del tipo let $x = \omega$ in 7 possono essere valutabili tramite una semantica lazy (in eager dovrei interagire con ω e mi porterebbe in loop)
L'omega permette di introdurre la ricorsione, che rende il linguaggio Fun turing completo

Semantica lazy statica

$$E \vdash \text{fn } x \Rightarrow M \rightsquigarrow (x,M,E)$$

$$\dfrac{E \vdash M \rightsquigarrow (x,M',E') \qquad E'(x,(N,E)) \vdash M' \rightsquigarrow v}{E \vdash MN \rightsquigarrow v}$$

Curryficazione

La curryficazione consente nel trasformare una funzione che prende più argomenti in una sequenza di funzioni, ognuna delle quali prende un solo argomento

$$A \times B \to C \quad \Rightarrow \quad A \to (B \to C)$$

es 5+7
$P \equiv \text{fn } x \Rightarrow [\text{fn } y \Rightarrow x+y]$
$(P5)7$
↳ da implementare

Grazie a questo strumento è possibile semplificare il linguaggio Fun rimuovendo let $x = M$ in N, infatti:
let $x = M$ in $N \equiv (\text{fn } x \Rightarrow N)M$

Numeri di Church

Nel λ-calcolo i numeri possono essere rappresentati come funzioni. In generale in numero n:
$n \equiv \lambda f.\lambda x.f^n(x)$
↳ applica f a x n volte

es SML
$2 \equiv \underbrace{\text{fn } x \Rightarrow \text{fn } x(xy)}_{\text{DUE}}$
adesso dobbiamo costruire una funzione eval per poterlo valutare

eval $= (\text{fn } z \Rightarrow z(\text{fn } x \Rightarrow x+1)0)$

verifichiamo

EAGER STATICA

eval due $= (\text{fn } z \Rightarrow z(\text{fn } x \Rightarrow x+1)0)$ DUE $=$
$= \text{DUE}(\text{fn } x \Rightarrow x+1)0 =$
$= (\text{fn } x \Rightarrow \text{fn } y \Rightarrow x(xy))(\text{fn } x \Rightarrow x+1)0 = (\text{fn } y \Rightarrow (\text{fn } x \Rightarrow x+1)((\text{fn } x \Rightarrow x+1)\ y))0 =$

$= (\text{fn } x \Rightarrow x+1)((\text{fn } x \Rightarrow x+1)\ 0) = (\text{fn } x \Rightarrow x+1)1 = 2$

vediamo ora la funzione per trovare il successore di un numero di Church
succ $= \text{fn } z \Rightarrow \text{fn } x \Rightarrow \text{fn } y \Rightarrow x(z\ x\ y)$

applichiamola

succ DUE =

$= (\text{fn } z \Rightarrow \text{fn } x \Rightarrow \text{fn } y \Rightarrow x(z\ x\ y))\ \text{DUE} =$

$= \text{fn } x \Rightarrow \text{fn } y \Rightarrow x(\text{DUE}\ x\ y) =$

$= \text{fn } x \Rightarrow \text{fn } y \Rightarrow x(\underbrace{(\text{fn } x \Rightarrow \text{fn } y \Rightarrow x(x y))\ x\ y}) =$

$\hookrightarrow (\text{fn } x \Rightarrow \text{fn } y \Rightarrow x(x\ y))\ x\ y = \text{fn } y \Rightarrow x(x\ y)\ y = x(x\ y)$

$= \text{fn } x \Rightarrow \text{fn } y \Rightarrow x(x(x y)) \equiv 3$