

CuCh machine
Linguaggi di Programmazione
a.a. 2019-2020

Edoardo De Matteis
1746561

Mirko Giacchini
matricola

Indice

1	Introduzione	1
2	Sintassi	1
3	Semantica operativa	2
3.1	Dynamic eager	2
3.2	Dynamic lazy	2
3.3	Static eager	3
3.4	Static lazy	3
4	Osservazioni	4
5	Alberi binari di Church	5

1 Introduzione

Una **Curry-Church** machine implementa un interprete minimale di un linguaggio funzionale non tipato.

2 Sintassi

syntax.sml

$$\begin{aligned} FUN ::= & Const\ k \mid Var\ x \\ & \mid Sum(M, N) \mid Fn(x, M) \\ & \mid Let(x, M, N) \mid App(M, N) \end{aligned} \tag{1}$$

$$K ::= 0 \mid 1 \mid \dots \tag{2}$$

$$X ::= A \mid \dots \mid Z \mid a \mid \dots \mid z \mid \dots \tag{3}$$

$$ENV : VAR \rightarrow VAR \times FUN \times ENV \tag{4}$$

$$find : ENV \times FUN \rightarrow (FUN \times ENV) \cup EXC \quad (5)$$

In ENV nel codominio il prodotto cartesiano presenta ENV poichè necessario nelle valutazioni con scoping statico, nel mondo dinamico non è necessario e semplicemente lo si ignora. EXC è l'insieme delle eccezioni.

3 Semantica operativa

$$\mapsto \subseteq ENV \times FUN \times VAL \equiv ENV \vdash FUN \mapsto VAL \quad (6)$$

3.1 Dynamic eager

dynamic_eager.sml

$$\frac{}{E \vdash Const\ k \mapsto Const\ k}$$

$$\frac{}{E \vdash Var\ x \mapsto v} \quad E(x) = v$$

$$\frac{E \vdash M \mapsto v1 \quad E \vdash N \mapsto v2}{E \vdash Sum(M, N) \mapsto v} \quad (v=v1+v2)$$

$$\frac{}{E \vdash Fn(x, M) \mapsto (x, M)}$$

$$\frac{E \vdash M \mapsto (x, M') \quad E \vdash N \mapsto v \quad E(x, v) \vdash M' \mapsto v'}{E \vdash App(M, N) \mapsto v'}$$

$$\frac{E \vdash M \mapsto v \quad E(x, v) \vdash N \mapsto v'}{E \vdash Let(x, M, N) \mapsto v'}$$

3.2 Dynamic lazy

dynamic_lazy.sml

$$\frac{}{E \vdash Const\ k \mapsto Const\ k}$$

$$\frac{E \vdash M \mapsto v}{E \vdash Var\ x \mapsto v} \quad E(x) = M$$

$$\frac{E \vdash M \mapsto v1 \quad E \vdash N \mapsto v2}{E \vdash Sum(M, N) \mapsto v} \quad (v=v1+v2)$$

$$\overline{E \vdash Fn(x, M) \mapsto (x, M)}$$

$$\frac{E \vdash M \mapsto (x, M') \quad E(x, N) \vdash M' \mapsto v}{E \vdash App(M, N) \mapsto v}$$

$$\frac{E(x, M) \vdash N \mapsto v}{E \vdash Let(x, M, N) \mapsto v}$$

3.3 Static eager

static_eager.sml

$$\overline{E \vdash Const\ k \mapsto Const\ k}$$

$$\frac{E \vdash M \mapsto v}{E \vdash Var\ x \mapsto v} \quad E(x) = M$$

$$\frac{E \vdash M \mapsto v1 \quad E \vdash N \mapsto v2}{E \vdash Sum(M, N) \mapsto v} \quad (v=v1+v2)$$

$$\overline{E \vdash Fn(x, M) \mapsto (x, M, E)}$$

$$\frac{E \vdash M \mapsto (x, M', E') \quad E \vdash N \mapsto v \quad E'(x, v) \vdash M' \mapsto v'}{E \vdash App(M, N) \mapsto v'}$$

$$\frac{E \vdash M \mapsto v \quad E(x, v) \vdash N \mapsto v'}{E \vdash Let(x, M, N) \mapsto v'}$$

3.4 Static lazy

static_lazy.sml

$$\overline{E \vdash Const\ k \mapsto Const\ k}$$

$$\frac{E' \vdash M \mapsto v}{E \vdash Var\ x \mapsto v} \quad E(x) = (M, E')$$

$$\frac{E \vdash M \mapsto v1 \quad E \vdash N \mapsto v2}{E \vdash Sum(M, N) \mapsto v} \quad (v=v1+v2)$$

$$\overline{E \vdash Fn(x, M) \mapsto (x, M, E)}$$

$$\frac{E \vdash M \mapsto (x, M', E') \quad E'(x, N) \vdash M' \mapsto v}{E \vdash App(M, N) \mapsto v}$$

$$\frac{E(x, M, E) \vdash N \mapsto v}{E \vdash Let(x, M, N) \mapsto v}$$

4 Osservazioni

Un programma che mette in mostra le differenze tra una valutazione eager ed una lazy è il programma

$$let\ x = x\ in\ x$$

che in una semantica static eager (come in SML) dà errore perchè x non è definita. In una semantica dynamic lazy va invece in loop

$$\frac{\dots}{\langle (x, x) \rangle \vdash x \mapsto} \\ \frac{\langle (x, x) \rangle \vdash x \mapsto}{\emptyset \vdash Let(x, x, x) \mapsto}$$

Inoltre in SML non è possibile eseguire il più piccolo transinfinito

$$\omega = (fn\ x \Rightarrow x\ x)(fn\ x \Rightarrow x\ x)$$

per via del sistema dei tipi, FN non essendo tipato non presenta questo problema e si entra in loop come si può dimostrare ad esempio con valutazione eager statica

$$\frac{\emptyset \vdash Fn(x, x\ x) \mapsto (x, x\ x, \emptyset) \quad \emptyset \vdash Fn(x, x\ x) \mapsto (x, x\ x, \emptyset) \quad \frac{\dots}{\langle (x, x\ x) \rangle \vdash x\ x \mapsto} \quad \frac{\langle (x, x\ x) \rangle \vdash x \mapsto}{\langle (x, x\ x) \rangle \vdash x\ x \mapsto}}{\emptyset \vdash Fn(x, x\ x)\ Fn(x, x\ x) \mapsto}$$

Un'esecuzione interessante è quella di $\langle (x, k) \rangle \vdash App(Fn(x, x), x)$ per la cattura di x .

$$\frac{\dots}{\langle (x, 3)(x, x) \rangle \vdash x \mapsto} \\ \frac{\langle (x, 3) \rangle \vdash Fn(x, x) \mapsto (x, x) \quad \langle (x, 3)(x, x) \rangle \vdash x \mapsto}{\langle (x, 3) \rangle \vdash App(Fn(x, x), x) \mapsto}$$

$$\frac{\langle(x, 3)\rangle \vdash Fn(y, y) \mapsto (y, y) \quad \langle(x, 3)(y, y)\rangle \vdash x \mapsto 3}{\langle(x, 3)\rangle \vdash App(Fn(y, y), x) \mapsto 3}$$