Semantica denotazionale (definizione formale delle esecuzioni dei programmi C)

Titolo nota 24/09/2015

$f : A \to B_\perp$

$f[b/a]^{add} = g$ se $f(a) = \perp$

$$g(m) = \begin{cases} b & \text{se } m = a \\ f(m) & \text{altrimenti} \end{cases}$$

$f[b/a]^{mod} = g$ se $f(a) \neq \perp$

$$g(m) = \begin{cases} b & \text{se } m = a \\ f(m) & \text{altrimenti} \end{cases}$$

OPERAZIONI SUI "FRAME"

PILE DI "FRAME" (OPERAZIONI)

Titolo nota 24/09/2015

$$\Pi = \{\Omega\} \cup \{f.\pi \mid f : A \to B_\perp \text{ e } \pi \in \Pi\}$$

$$\pi\left[{}^{b}/_{a}\right]^{add} = f\left[{}^{b}/_{a}\right]^{aold} . \pi' \qquad \text{se } \pi = f.\pi' \text{ e } f(a) = \perp$$

$$\pi\left[{}^{b}/_{a}\right]^{mod} = \begin{cases} f\left[{}^{b}/_{a}\right]^{mod} . \pi' & \text{se } \pi = f.\pi' \text{ e } f(a) \neq \perp \\ f.\pi'\left[{}^{b}/_{a}\right]^{mod} & \text{se } \pi = f.\pi' \text{ e } f(a) = \perp \end{cases}$$

non si mette il caso Ω perché l'operazione di mod su Ω è indefinita

PILE DI "FRAME" (operativi)

Titolo nota 24/09/2015

$$\pi(a) = \begin{cases} \bot & \text{se } \pi = \Omega \\ f(a) & \text{se } \pi = f.\pi' \text{ e } f(a) \neq \bot \\ \pi'(a) & \text{se } \pi = f.\pi' \text{ e } f(a) = \bot \end{cases}$$

Valori (e non più come nelle operazioni precedenti)

Ambiente e Memoria come nuovo stato (entrambi più di "frame")

$$P = \{\Omega\} \cup \{\varphi.\rho \mid \varphi : Ide \to Loc_\bot \text{ e } \rho \in P\}$$

$$M = \{\Omega\} \cup \{\nu.\mu \mid \nu : Loc \to Val_\bot \text{ e } \mu \in M\}$$

φ: frame di ambiente

ν: frame di memoria

$\Big\|$ Stato

Titolo nota 24/09/2015

Exp → Ide | Num | Exp Op Exp | *Ide | & Ide Op → + | * | ...

Dec → Type Ide | Type Ide = Exp ;

Com → Ide = Exp ; | if (Exp) Com else Com |
while (Exp) Com | { ~~Dec_list~~ Com_list } | Block

Block → { Dec_list Com_list }

Com_list → Com | Com Com_list

Dec_list → Dec | Dec Dec_list

$Sem_e : Exp \times P \times M \to Val_\perp$

$Sem_d : Dec \times P \times M \to P \times M$

$Sem_c : Com \times P \times M \to M$

$Sem_{dl} : Dec\_list \times P \times M \to P \times M$

$Sem_{cl} : Com\_list \times P \times M \to M$

Tipi delle "funzioni semantiche"

$$Sem_e(x, \rho, \mu) = \mu(\rho(x))$$

$$Sem_e(n, \rho, \mu) = val(n)$$

$$Sem_e(e_1 \ op \ e_2, \rho, \mu) = v_1 \ o \ v_2$$

calcolo i valori delle espressioni e_1 e e_2 nello stato e applico l'operatore ai due valori.

o = operatore corrispondente al simbolo op

dove
- $v_1 = Sem_e(e_1, \rho, \mu)$
- $v_2 = Sem_e(e_2, \rho, \mu)$

o = operatore corrispondente al simbolo op

op simbolo sintattico per quell'operatore

definizione ricorsiva

Semantica delle dichiarazioni ($Sem_d : Dec \times P \times M \to P \times M$)

Titolo nota 24/09/2015

$Dec \to Type\ Ide\ ; \mid Type\ Ide = Exp\ ;$

$succloc : M \to Loc$

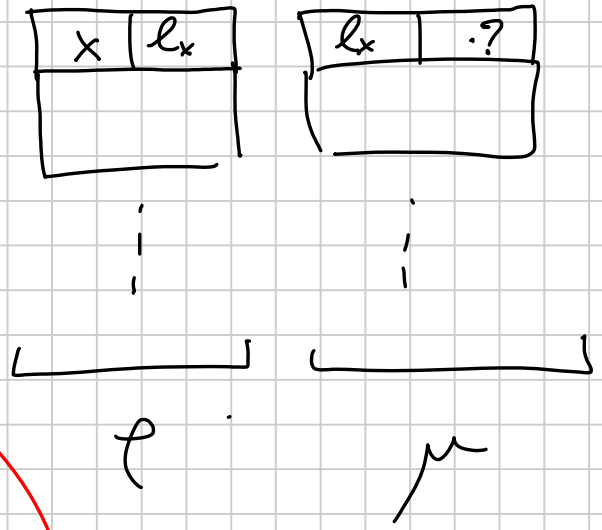
$$Sem_d(\underline{T}\ x\ ;, \rho, \mu) = (\rho[l/x]^{add}, \mu[?/l]^{add})$$

dove

$$l = succloc(\mu)$$

applico succloc alla memoria μ, ottengo la prima locazione libera che chiamo l.

$? \in Val$ significa che non so quale valore $\in Val$ ci sia in memoria



$$Sem_d(\overline{T}\ x = e;, \rho, \mu) = (\rho[l/x]^{add}, \mu[v/l]^{add})$$

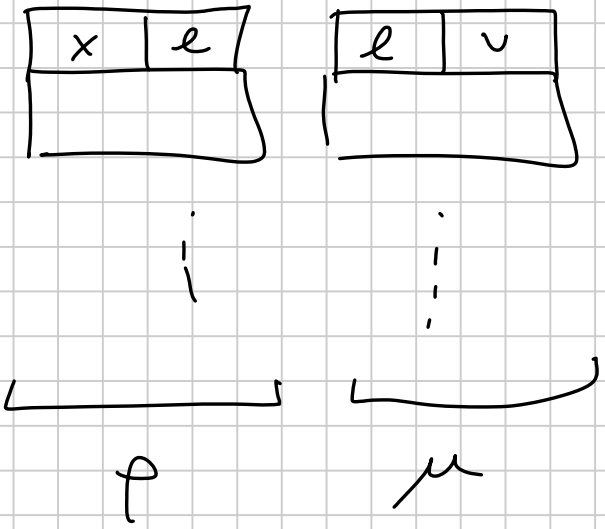
dove

$l = succloc(\mu)$

$v = Sem_e(e, \rho, \mu)$

calcolo il valore della espressione e.

e lo chiamo $\bar{v}$



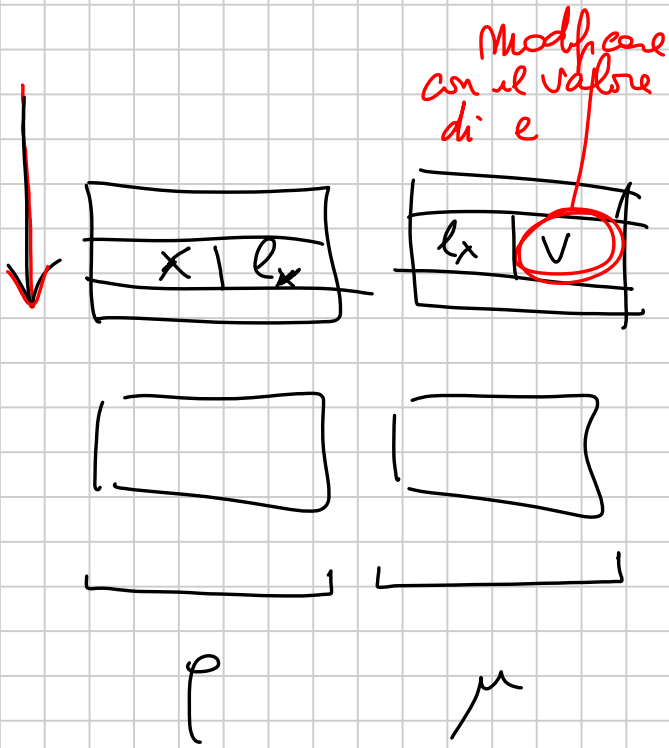
Semantica dei comandi $Sem_c : Com \times P \times M \to M$

Titolo nota 24/09/2015

$$Sem_c(x = e;, \rho, \mu) = \mu\left[\frac{v}{\rho(x)}\right]^{mod}$$

dove $v = Sem_e(e, \rho, \mu)$

calcolo il valore della espressione e e lo chiamo v



Titolo nota 24/09/2015

$Sem_c(\text{if } (e)\ c_1 \text{ else } c_2, \rho, \mu) =$

$\equiv$

$Sem_c(c_1, \rho, \mu)$?

Sì!

$\begin{cases} \mu' \text{ dove } \mu' = Sem_c(c_1, \rho, \mu) & \text{se } Sem_e(e, \rho, \mu) = 1 \\ \mu' \text{ dove } \mu' = Sem_c(c_2, \rho, \mu) & \text{se } Sem_e(e, \rho, \mu) = \emptyset \end{cases}$

Vero

falso

Titolo nota 24/09/2015

$$Sem_C(\text{while } (e)\ c, \rho, \mu) = \begin{cases} \mu & \text{se } Sem_E(e, \rho, \mu) = \emptyset \\ \mu'' & \text{se } Sem_E(e, \rho, \mu) = 1 \end{cases}$$

$\text{while } (e)\ c \in Cmd$

dove

$\mu' = Sem_C(c, \rho, \mu)$

valuto il comando c, ottengo una nuova memoria che chiamo μ'

$\mu'' = Sem_C(\text{while } (e)\ c, \rho, \mu')$

rivaluto l'intero $\text{while } (e)\ c$ e ottengo (spero di ottenere) una nuova memoria che chiamo μ''

(perché potrebbe anche non finire)

Blocco

$Sem_{dl} : Dec\_list \times P \times M \to P \times M$

Titolo nota 24/09/2015

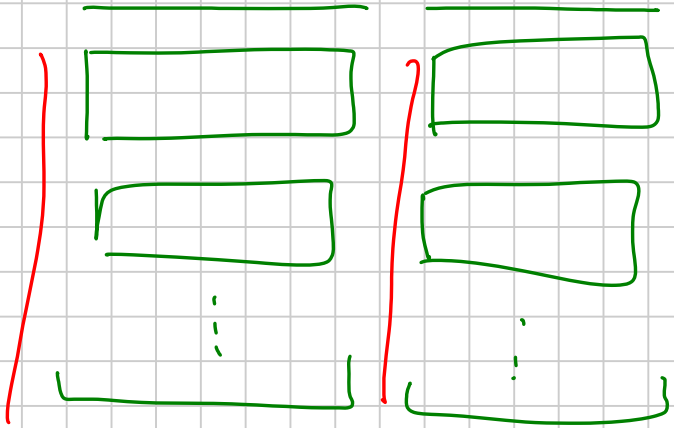
$Sem_c(\{dl\ c\}, \rho, \mu) =$

dove

$\to (\varphi.\rho\ \nu.\mu) = Sem_{dl}(dl, \omega.\rho, \omega.\mu)$

ω ambiente e memoria a cui ho aggiunto un frame vuoto in testa

delle dichiarazioni dl ottengo un nuovo ambiente e una nuova memoria in cui sono cambiati solamente i frame in testa che diamo rispettivamente φ e ν.



$\omega(n) = l$

Blocco

$Sem_{dc} : Dec\_list \times P \times M \to P \times M$

Titolo nota 24/09/2015

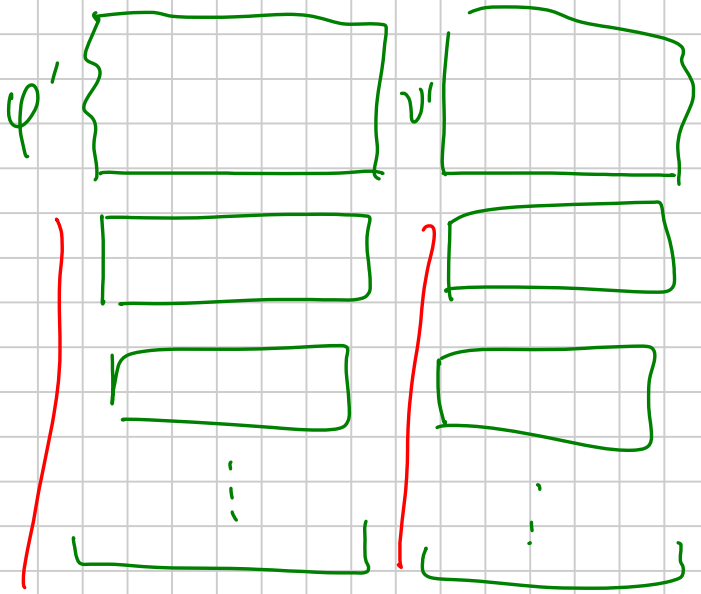
$Sem_c(\{dc\ cl\}, \rho, \mu) = \mu'$

dove

$(\varphi.\rho\ \nu.\mu) = Sem_{dc}(dc, \omega.\rho, \omega.\mu)$

$\nu'.\mu' = Sem_{cl}(cl, \varphi.\rho, \nu.\mu)$

nuova memoria che ottengo valutando i comandi cl a partire dallo stato $(\varphi.\rho, \nu.\mu)$



$\omega(n) = 1$

$Exp \rightarrow \ldots \mid *Id \mid \&Id$

$Cmd \rightarrow \ldots \mid *Id = Exp;$