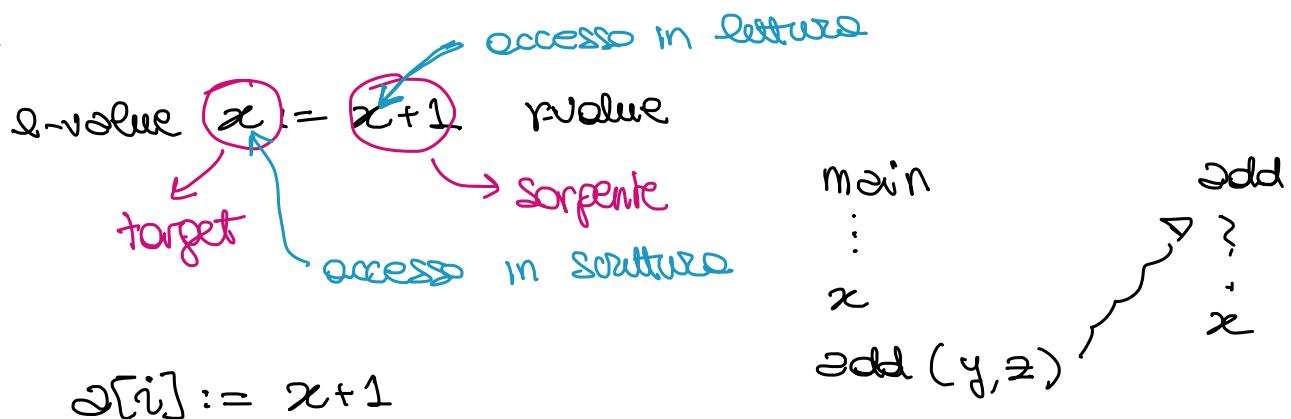


Espressioni → con id costanti }
Dichiarazioni → con id costanti } → Ambiente
(associazione
id - oggetto denotato)

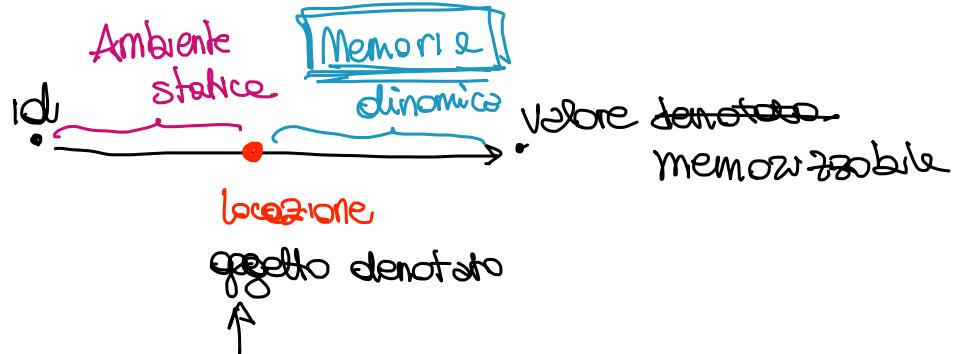
MD Concetto di tipo (tipo dell'oggetto dendato)

↳ Semionno stato che verifice i vincoli contestuali (vincoli di tipo)

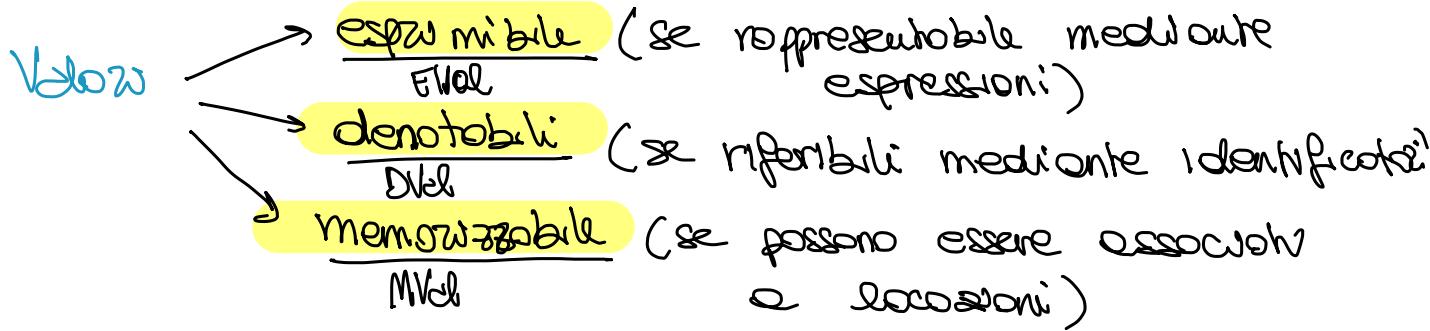
Memoria



↪ LOCAZIONE (costruzione delle celle di memoria)



Memoria =insieme di associazioni fra lezioni e
Vocabolario memoriazabili

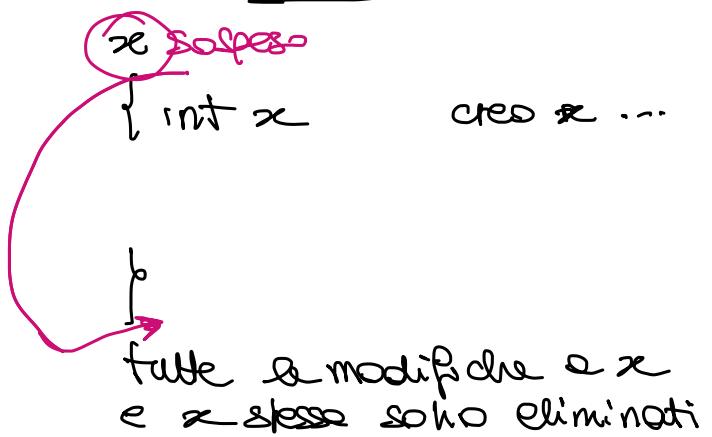


La memoria serve per catturare le trasformazioni irreversibili generate dal programma.

L'ambiente cattura trasformazioni reversibili

$$x := x + 1$$

$$x := x - 1$$



Mem insieme di memorie (Associazioni tra locazione e valore)

Loc insieme di locazioni

MVal insieme di valori memorizzabili

$$\sigma \in \text{Mem} \quad \sigma : \text{Loc} \rightarrow \text{MVal}$$

$$L \subseteq_f Loc$$

$$\nexists \sigma \in L. \sigma : l \mapsto \text{NG MVal}$$

$$\text{Mem}_L = L \rightarrow \text{MVal}$$

$$L \subseteq_f Loc$$

$$\text{Mem} = \bigcup_{L \subseteq_f Loc} \text{Mem}_L$$

insieme di associazioni da un qualsiasi insieme finito di Loc ad un insieme di valori (memorizzabili)

DVal = Eval \cup loc \rightsquigarrow NVal \cup loc

MVal = NVal

Aggiornamento di una memoria

$\sigma[\sigma']$

$\sigma, \sigma' \in \text{Mem}$

$\sigma : L$

(L è il dominio di σ)

$\sigma' : L'$

(L' dominio di σ')

$\sigma'' = \sigma[\sigma'] \in \text{Mem}$

$\forall l \in L \cup L' . \sigma''(l) =$

$\begin{cases} \sigma'(l) & l \in L' \\ \sigma(l) & l \in L \setminus L' \end{cases}$

$\sigma(l) & l \in L \setminus L'$

Aggiungiamo alle dichiarazioni la dichiarazione di identificatore ~~versabile~~

Dec : $D \rightarrow \dots | \underline{\text{var } x : \tau = e}$

dobbiamo aggiungere ai NPs
le tipo location

Tipi = {int, bool, intloc
boolloc}

Loc \rightarrow location di tipo τ

(intloc : loc per valori interi)

boolloc : loc per valori booleani)

Semantica statica per Dec

associa un ambiente statico

DI(var $x : \tau = e) = \{x\}$

FI(var $x : \tau = e) = F_I(e)$

Aggiungiamo queste def. a quelle già viste

Regole sem. statiche che oppungono (alle dichiarazioni)

$$\frac{\Delta \vdash e : \tau}{\Delta \vdash \text{var } x : \tau = e : [x \mapsto \tau \text{ in } e]}$$

Semantico statico delle espressioni identificatore può essere
una variabile

$$\Delta \vdash I : \tau \quad \text{se } \Delta(I) \text{ ed è, } \underline{\text{clock}}$$

Semantico dinamico

↪ combio per l'appunti della memoria

$$\Gamma = \Sigma \times \text{Mem} \xrightarrow{*} \Gamma' \times \text{Mem}$$

N.B.

Espressioni

$$pt \langle e, \sigma \rangle \rightarrow \langle e', \sigma' \rangle$$

La semantico dinamico delle espressioni si porta dentro le memorie in cui valutare l'espressione

$$\overbrace{\Gamma}^T$$

$$\Gamma = D \times \text{Mem} \xrightarrow{*} \text{Env} \times \text{Mem}$$

Dichiarazioni

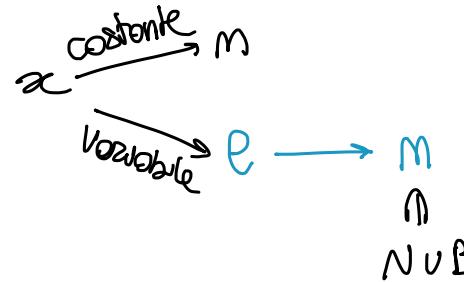
$$pr \langle d, \sigma \rangle \rightarrow \langle d', \sigma' \rangle$$

σ' può essere diverso da σ

- Regole da aggiungere alle espressioni:

$$\vdash \langle x, \sigma \rangle \rightarrow \langle m, \sigma' \rangle$$

$$\rho(x) = m \text{ oppure } \rho(x) = \rho_{\text{loc}} \\ (\text{come prima}) \quad \underline{\sigma(e) = m}$$



- Regole da aggiungere alle dichiarazioni:

$$\frac{}{\vdash \langle e, \sigma \rangle \rightarrow^* \langle k, \sigma' \rangle}$$

$$\vdash \langle \text{var } x : e = e, \sigma \rangle \rightarrow \langle [x \mapsto e], \sigma'[e \mapsto k] \rangle$$

$e \in \text{Loc}$ nuova locazione

Unica regola delle dichiarazioni

che puo' modificare la memoria

(Side effect)

Valutazione di una espressione con id. variabili

$$\text{Eval} : \text{Exp} \times \text{Mem} \rightarrow \text{Val} \times \text{Mem}$$

$$\text{Eval}(e, \sigma) = (k, \sigma') \text{ sse } \vdash \langle e, \sigma \rangle \rightarrow^* \langle k, \sigma' \rangle$$

→ Equivalenza di espressioni con id. variabili

$$\equiv \subseteq \text{Exp} \times \text{Exp}$$

$$\forall e_0, e_1 \in \text{Exp} \quad e_0 \equiv e_1 \quad \text{sse } \forall \sigma. \text{Eval}(e_0, \sigma) = \text{Eval}(e_1, \sigma)$$

$$2 + x - x \equiv \frac{2x}{x}$$

Elaborazione di dichiarazioni con id variabili

$\text{Elob} : \text{Dec} \times \text{Mem} \rightarrow \text{Env} \times \text{Mem}$

$\text{Elob}(d, \sigma) = (p, \sigma')$ se

$\vdash \langle d, \sigma \rangle \rightarrow^* \langle p, \sigma' \rangle$

da una dichiarazione ad un ambiente

Equivalente per dichiarazioni con id variabili

$\equiv \subseteq \text{Dec} \times \text{Dec}$

$\forall d_1, d_2 \in \text{Dec} \quad d_1 \equiv d_2 \iff \forall \sigma \in \text{Mem}$

$\text{Elob}(d_1, \sigma) = \text{Elob}(d_2, \sigma)$

Per modificare la memoria abbiamo bisogno di costrutti dedicati → comandi

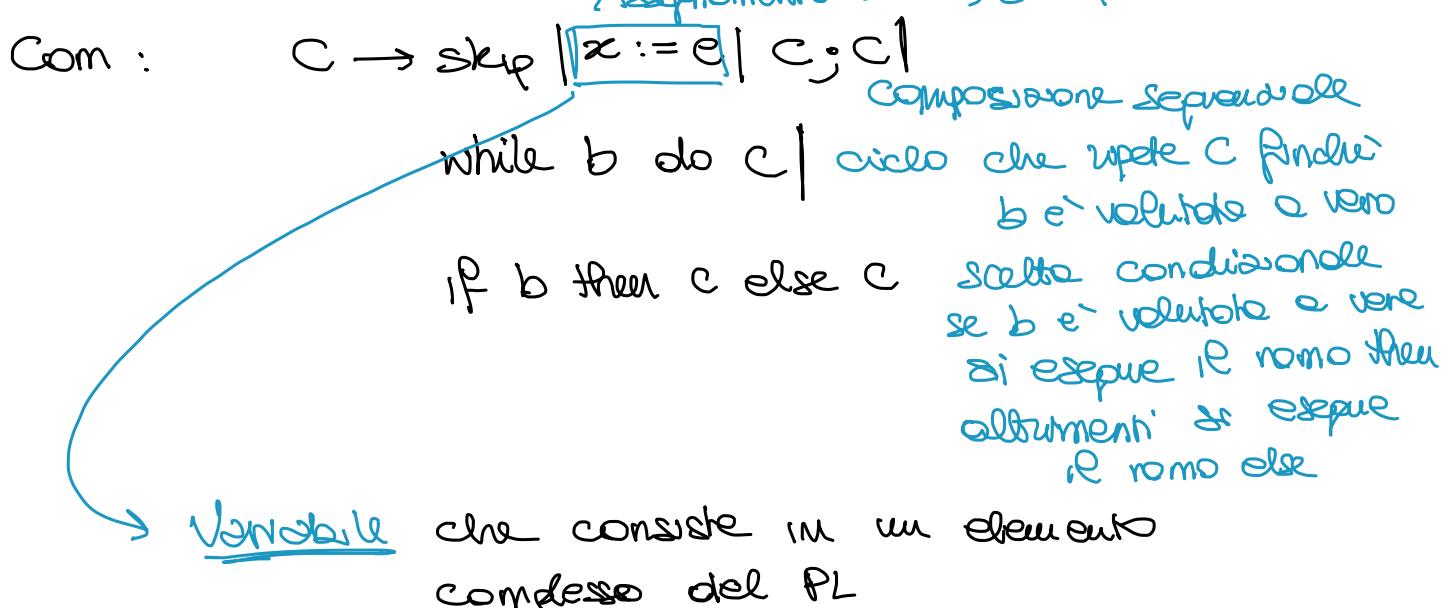
↪ strumenti che abbiamo nel PL per denotare richieste di modifica della memoria.

I comandi vengono eseguiti per ottenere la trasformazione della memoria denotata

→ Assegnamento (costrutto base di modifica della memoria)

→ Composizione (sequenziale)

→ Controllo del flusso di esecuzione (condizionale e iterativo)



- Elementi che lo caratterizzano
- Nome / identificatore
 - Tipo (insieme dei valori / operazioni sui valori) che caratterizza il contenuto della variabile
 - Valore (valore riferibile mediante l'occurrenza del nome)
- ↪ Indirizzo / Locazione
- ↪ Scope visibilità nell'ambiente (ambiente di riferimento)
- ↪ Tempo di vita misura in termini temporali dell'esistenza della variabile